1990 Data Book/Handbook

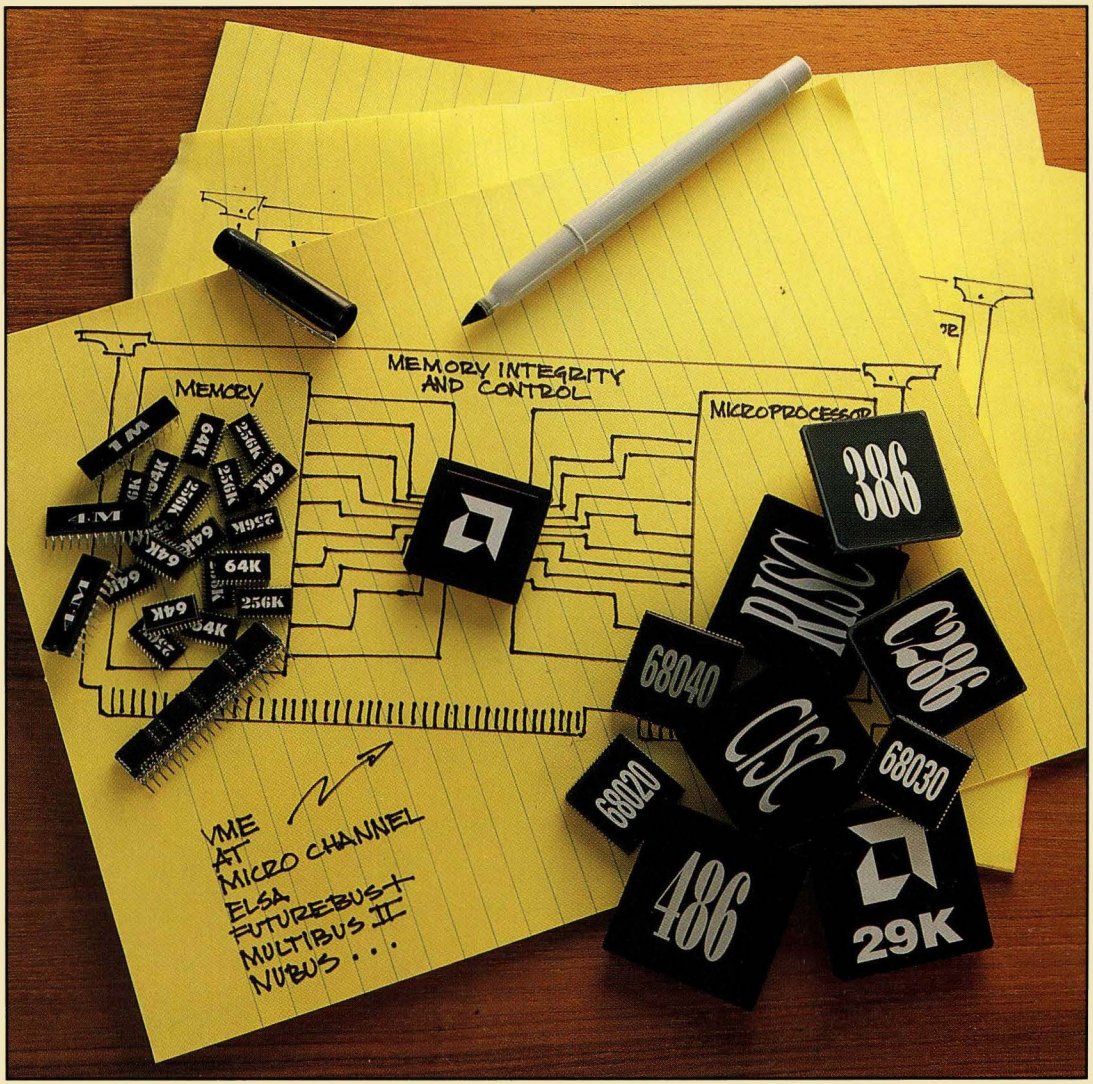
Dynamic Memory Design

AMD

Dynamic Memory Design

1990 Data Book/Handbook

Advanced
Micro
Devices





Advanced Micro Devices

Dynamic Memory Design

1990 Data Book/Handbook

© 1990 Advanced Micro Devices

Advanced Micro Devices reserves the right to make changes in its products without notice in order to improve design or performance characteristics.

This publication neither states nor implies any warranty of any kind, including but not limited to implied warranties of merchantability or fitness for a particular application. AMD assumes no responsibility for the use of any circuitry other than the circuitry in an AMD product.

The information in this publication is believed to be accurate in all respects at the time of publication, but is subject to change without notice. AMD assumes no responsibility for any errors or omssions, and disclaims responsibility for any consequences resulting from the use of the information included herein. Additionally, AMD assumes no responsibility for the functioning of undescribed features or parameters.

Trademarks

Am29000 and ADAPT29K are trademarks of Advanced Micro Devices, Inc.
PAL is a registered trademark of Advanced Micro Devices, Inc.
PS/2 and Micro Channel are trademarks of IBM Corporation
PC-AT is a registered trademark of IBM Corporation
Xenix and OS/2 are trademarks of Microsoft Corporation
Multibus is a registered trademark of Intel Corporation
Macintosh is a registered trademark of Apple Computer, Inc.
SmartModel is a registered trademark of Logic Automation, Inc.

Dynamic memory subsystems are increasingly becoming the main system performance bottleneck in low, medium, and high-end computation and communication systems. Performance advances in microprocessor and bus technologies are outpacing memory throughput improvements. AMD's Dynamic Memory Management Products are designed to reduce memory system overhead penalties through high integration. Sub-micron CMOS technology and configurable functionality provide system-specific performance enhancement with low static and dynamic power consumption.



Fred J. Roeder
Vice President
Standard Logic Division

PREFACE

A pioneer in the field of Dynamic Memory Management since 1981, AMD offers a complete solution for the design of today's sophisticated, high-speed memory systems—a family of CMOS DRAM Management building blocks comprising the following:

- Dynamic Memory Controllers
- Error Detection and Correction (EDC) Circuits
- Multiple Bus Exchange (MBE)
- DRAM Drivers

These versatile memory management circuits can ease the design task precipitated by new and faster microprocessors, including RISC microprocessors, the increased demands for more system memory, and the requirements arising when designing DRAMs into smaller systems. They also offer design flexibility for expanding the basic 16-bit system to 32- or 64-bit word widths, and beyond.

This handbook/databook provides descriptions of the memory management circuits including specifications and gives specific examples of how to design dynamic memory systems, using these high-speed CMOS building blocks. All necessary functions are available to the system designer so that he can obtain the best cost/performance ratio to satisfy his memory-system design.

A Product Selector Guide appears on page viii following the Table of Contents.

Chapter 1 gives an overview of the three principle memory management building blocks.

Chapter 2 contains a collection of material to aid the user in designing his memory subsystem. It includes

- A discussion of DRAM types, special access modes and refresh types,
- Two chapters from Clearpoint Research Corporation's "Designer's Guide to Add-on Memory,"
- Error detection and correction system architectures and capabililties,
- And finally, a brief overview of system buses.

Chapter 3 presents four application notes describing different interface designs using the Am29C688 4M Configurable Dynamic Memory Controller/Driver(CDMC).

Chapter 4 comprises two application notes demonstrating the capabilities of the Am29C660 32-bit Error Detection and Correction (EDC) Circuit and the Am29C668 CDMC when used with the IBM PC-AT and PS/2 bus architectures.

Chapter 5 is a collection of article reprints: two detailing the 29C668 CDMC circuit, another presenting a clock-generator circuit using the Am2971A Programmable Event Generator (PEG), and one describing a demonstration board using the Am95C71 Video Data Compression/Expansion Processor (VCEP) and the Am29C668 CDMC.

Chapter 6 contains memory management data sheets as listed in the table of contents.

Chapter 7 shows packaging and physical dimensions.

The Appendix is a brief discussion of the behavioral simulation models from Logic Automation, Inc.

TABLE OF CONTENTS

Chapter 1 Dynamic Memory Design Overview

AMD's System Design Methodology 1-3
Family Overview 1-3
Dynamic Memory Control 1-3
Error Detection and Correction(EDC) 1-3
Multiple Bus Exchange(MBE) 1-4
DRAM Drivers 1-4

Chapter 2 Memory System Architectures

Introduction 2-3
DRAM Basics 2-4
DRAM Types and Accesses 2-4
DRAM Refresh Types 2-6
Understanding Memory Design 2-8
For the Memory Novice: An Overview 2-8
Memory Design for Improved System Performance 2-11
Bus Efficiency 2-11
Memory Technologies and High Performance 2-13
New Technology 2-16
Design Integrity 2-17
Designing for Reliability 2-19
The Probability of Errors 2-19
Error Protection 2-19
Running Bare 2-20
Parity Generation and Checking 2-20
Error Detection and Correction 2-20
The Mechanics of EDC 2-20
Parity vs EDC: A Comparison 2-21
The Importance of the EDC Word Length 2-21
EDC on Array Cards 2-22
Redundancy 2-22
Mean Time to Repair 2-23
Error Detection and Correction Architectures 2-24
Fly-By 2-24
Flow-Through 2-25
Refresh With Scrubbing 2-25
Am29C660 CMOS Cascadable 32-Bit EDC Circuit 2-27
Program to Evaluate Am29C660 Multiple Error Detection Capability 2-28
Memory Reliabilities With and Without the 29C66 EDC Circuit 2-30

Chapter 2 Memory System Architectures (Continued)

System Buses 2-31
VMEbus 2-31
Multibus I and II 2-32
Nu Bus 2-32
AT Bus 2-32
Micro Channel 2-33
EISA 2-33
Q-Bus 2-33
MicroVAX II 2-33
MicroVAX 3000 2-33
Futurebus+ 2-34

Chapter 3 Microprocessor Interfaces to the Am29C688 Configurable Dynamic Memory Controller/Driver

Introduction 3-2
Am29C668 CDMC to Am29000 Streamlined Instruction Processor Interface 3-3
Am29C668 CDMC to 80C286 Microprocessor Interface 3-37
Am29C668 CDMC to 80386 Microprocessor Interface 3-53
Am29C668 CDMC to 68020 Microprocessor Interface 3-71

Chapter 4 EDC Memory Board Designs

Introduction 4-2
IBM PC-AT Plug-in Memory Card with EDC 4-3
IBM PS/2 12-Mbyte Memory Board with EDC 4-25

Chapter 5 Special Applications and Article Reprints

Introduction 5-2
Configurable DRAM Controller Enhances System Performance 5-3
Four-Megabit DRAM Controller Offers Burst Addressing 5-11
Expand or Shrink Clock Cycles to the System's Needs 5-15
High-Speed VCEP Demonstration Board Using the Am29C668 CDMC 5-19

Chapter 6 Product Data Sheets (Numerical Index)

Am2925A	60-MHz Microcycle-Length Programmable Clock Generator	6-3
Am2960/-1/A	16-Bit Cascadable EDC Circuit	6-24
Am2964B	64K Dynamic Memory Controller	6-66
Am2965/66	8-Bit DRAM Driver	6-80
Am2968A	256K Dynamic Memory Controller/Driver	6-88
Am2971A	100-MHz Enhanced Programmable Event Generator (PEG™)	6-112
Am2976	11-Bit DRAM Driver	6-141
Am29368	1M Dynamic Memory Controller/Driver	6-150
Am29C60/-1/A	16-Bit Cascadable EDC Circuit	6-172
Am29C660/A-E	32-Bit Cascadable EDC Circuit	6-189
Am29C668/-1	4M Configurable Dynamic Memory Controller/Driver	6-222
Am29C827A/8/A	10-Bit CMOS Bus Buffer	6-266
Am29C983/A	9-Bit x 4-Port Multiple Bus Exchange	6-275
Am29C985	9-Bit x 4-Port Multiple Bus Exchange with Parity Checker/Generator	6-291
673104A	1M Dynamic RAM Controller/Driver	6-305
SN74S409/-2	256K Dynamic RAM Controller/Driver	6-327

Chapter 7 Physical Dimensions 7-3

Appendix Behavioral Simulation Models from Logic Automation, Inc. A-3

PRODUCT SELECTOR GUIDE

DRAM Controllers

Am29C668	4M Configurable Dynamic Memory Controller/Driver, 33 ns	6-222
Am29C668-1	4M Configurable Dynamic Memory Controller/Driver, 29 ns	6-222
Am29368	1M Dynamic Memory Controller/Driver	6-150
Am2968A	256K Dynamic Memory Controller/Driver	6-88
Am2964B	64K Dynamic Memory Controller	6-66
673104A	1M Dynamic RAM Controller/Driver	6-305
SN74S409	256K Dynamic RAM Controller/Driver	6-327
SN74S409-2	256K Dynamic RAM Controller/Driver, High Speed	6-327

Error Detection and Correction Circuits

Am29C660	32-Bit CMOS Cascadable EDC (36 ns Error Detect)	6-189
Am29C660A	32-Bit CMOS Cascadable EDC (30 ns Error Detect)	6-189
Am29C660B	32-Bit CMOS Cascadable EDC (25 ns Error Detect)	6-189
Am29C660C	32-Bit CMOS Cascadable EDC (16 ns Error Detect)	6-189
Am29C660D	32-Bit CMOS Cascadable EDC (12 ns Error Detect)	6-189
Am29C660E*	32-Bit CMOS Cascadable EDC	
Am29C60	16-Bit CMOS Cascadable EDC (32 ns Error Detect)	6-172
Am29C60-1	16-Bit CMOS Cascadable EDC (25 ns Error Detect)	6-172
Am29C60A	16-Bit CMOS Cascadable EDC (20 ns Error Detect)	6-172
Am2960	16-Bit Cascadable EDC (32 ns Error Detect)	6-24
Am2960-1	16-Bit Cascadable EDC (25 ns Error Detect)	6-24
Am2960A	16-Bit Cascadable EDC (18 ns Error Detect)	6-24

Multiple Bus Exchanges

Am29C983	9-Bit x 4-Port MBE	6-275
Am29C983A*	9-Bit x 4-Port MBE, High Speed	6-275
Am29C985*	9-Bit x 4-Port MBE with Parity Checker/Generater	6-291

DRAM Drivers

Am2976	11-Bit DRAM Driver	6-141
Am2966	8-Bit DRAM Driver, Non-inverting	6-80
Am2965	8-Bit DRAM Driver, Inverting	6-80
Am29C827A	10-Bit CMOS Bus Buffer, Non-inverting	6-266
Am29C828A	10-Bit CMOS Bus Buffer, Inverting	6-266

Programmable Clock

Am2971A	100-MHz Enhanced Programmable Event Generator (PEG™)	6-112
Am2925A	60-MHz Microcycle Length Programmable Clock Generator	6-3

* in development

CHAPTER 1
Dynamic Memory Design Overview

AMD's System Design Methodology 1-3
Family Overview 1-3
Dynamic Memory Control 1-3
Error Detection and Correction(EDC) 1-3
Multiple Bus Exchange(MBE) 1-4
DRAM Drivers 1-4

DYNAMIC MEMORY DESIGN OVERVIEW

AMD's SYSTEM DESIGN METHODOLOGY

AMD's new CMOS Dynamic Memory Management family, featuring maximum performance and flexibility, offers a complete system solution for memory-system design. This family contains functions that are generally applicable for a wide range of memory requirements over the entire computing spectrum, from powerful desktop PCs and workstations through superminis, mainframes and telecommunication applications. In each system area, the AMD solution achieves maximum performance and reliability at minimum cost and with minimum device count.

All necessary functions are available to the system designer so that he can tailor the memory subsystem to his specific requirements:

- Complete address-path and refresh control,
- Controlled edge-rate drive for DRAM inputs,
- Programmable timing generation,
- Error Detection and Correction (EDC),
- System data-bus interface.

The AMD DRAM Management building blocks offer design flexibility in a variety of applications–expansion from the basic 16-bit system to 32- or 64-bit systems, and beyond, with or without EDC protection.

Today's high-performance "burst" microprocessors running at speeds above 16 MHz are fully supported. Read/write features include full support of byte writing, selectable access-timing options, burst-mode access support, page-mode-access support, static-column-access support, true bank interleaving, and selectable output-drive configurations. Using AMD's proprietary cache-access mode, designers can take full advantage of the performance benefits of page-mode DRAMs by continually comparing bank and row addresses during subsequent accesses.

Selectable refresh options include standard row refresh and CAS-before-RAS refresh, which is supported by some DRAMs. In a system employing EDC logic for memory-system integrity, refresh with scrubbing that prevents accumulation of soft errors is fully supported. Configurable row, column, and bank refresh counters and timing logic provide for a built-in EDC initialization mode, during which a known value is written to every memory location before starting normal operation.

Every member of the Dynamic Memory Management family is processed using AMD's advanced submicron CMOS technology. Full qualification and reliability data is available upon request.

The driving force behind this application handbook/databook is the determination to provide immediate answers to most common memory design and application questions. Full application support is vital throughout the complete design cycle from conception, through working prototypes, to production.

FAMILY OVERVIEW

Dynamic Memory Control

AMD's newest and most sophisticated DRAM controller is the Am29C668 4M Configurable Dynamic Memory Controller/Driver. This device provides the logic necessary to access and refresh 64K, 256K, 1M and 4M x n DRAMs. New features of the Am29C668 include support for burst-mode microprocessor accessing, automatic access timing, support for page-mode DRAMs, and selectable output-drive configurations. The Am29C668 can directly drive two banks of 39 DRAMs (32-bit word plus seven check bits) or four banks of 22 DRAMs (16-bit word plus six check bits) with its proprietary low-ground-bounce, low-undershoot outputs.

Other DRAM controllers offered by AMD include the Am29368 and 673104A 1M DRAM Controller/Drivers, the Am2968A and SN74S409 for 256K DRAMs, and the Am2964B for 64K DRAMs, the first monolithic DRAM controller offered from any manufacturer.

Error Detection and Correction (EDC)

AMD's newest EDC circuit is the CMOS Cascadable 32-bit Am29C660 that uses an industry-standard modified Hamming code to generate check bits and detect and correct hard and soft errors. The Am29C660 may be used with any memory technology including DRAM, SRAM, EPROM, EEPROM, Flash, and other types that exhibit increased soft error rates due to very small cell geometries.

The Am29C660 is currently the world's fastest and lowest power 32-bit EDC circuit. It is currently available in five speed grades; the fastest detects errors in 12 ns and corrects them in 18 ns maximum under worst-case operating conditions. The device is available in industry-standard 68-pin PLCC and ceramic PGA packaging in both commercial and military versions.

A full range of both bipolar and CMOS 16-bit EDC circuits is also available. These are the industry standard Am29C60 and Am2960 EDC families. The Am29C60A detects errors in 20 ns and corrects them in 25 ns maximum. It requires the lowest power in the industry.

Multiple Bus Exchange (MBE)

These devices are general-purpose, high-speed, digital cross-point switches, designed to improve interbus communications. The Am29C983 and Am29C983A are 9-bit x 4-port MBEs with input and output latches on all TTL compatible I/O ports. Any port may serve as either a source or destination. Differing sets of I/Os may communicate concurrently with one another. All outputs have 48-mA drive capability for efficiently driving high capacitive and inductive buses. The Am29C985 9-bit x 4-port MBE incorporates parity-check and generation capabilities. More detailed application information on the MBE may be obtained from the Multiple Bus Exchange Handbook/Databook.

DRAM Drivers

All of these devices offer proprietary edge-rate-controlled outputs to reduce output undershoots, overshoots and ground bounce. Skew times between outputs are also minimized. The Am29C827A and 29C828A are 48 mA bus buffers that may also be used to drive DRAM address and control inputs. These 10-bit wide devices are well suited for driving 1-Mbit x 1 and 1Mbit x 4 DRAMs. The 11-bit Am2976 helps minimize skews on the address lines on 4-Mbit x 1 DRAMS. The Am2965 and Am2966 are 8-bit DRAM drivers with industry-standard pinouts.

CHAPTER 2
Memory System Architectures

Introduction 2-3
DRAM Basics 2-4
DRAM Types and Accesses 2-4
DRAM Refresh Types 2-6
Understanding Memory Design 2-8
For the Memory Novice: An Overview 2-8
Memory Design for Improved System Performance 2-11
Bus Efficiency 2-11
Memory Technologies and High Performance 2-13
New Technology 2-16
Design Integrity 2-17
Designing for Reliability 2-19
The Probability of Errors 2-19
Error Protection 2-19
Running Bare 2-20
Parity Generation and Checking 2-20
Error Detection and Correction 2-20
The Mechanics of EDC 2-20
Parity vs EDC: A Comparison 2-21
The Importance of the EDC Word Length 2-21
EDC on Array Cards 2-22
Redundancy 2-22
Mean Time to Repair 2-23
Error Detection and Correction Architectures 2-24
Fly-By 2-24
Flow-Through 2-25
Refresh With Scrubbing 2-25
Am29C660 CMOS Cascadable 32-Bit EDC Circuit 2-27
Program to Evaluate Am29C660 Multiple Error Detection Capability 2-28
System Buses 2-31
VMEbus 2-31
Multibus I and II 2-32
Nu Bus 2-32
AT Bus 2-32
Micro Channel 2-33
EISA 2-33
Q-Bus 2-33
MicroVAX II 2-33
MicroVAX 3000 2-33
Futurebus+ 2-34

MEMORY SYSTEM ARCHITECTURES

INTRODUCTION

Dynamic memory systems differ extensively; they use different types and densities of DRAMs with varying access modes, timing requirements, refresh options, and architectural organizations. The DRAMs are organized in different word lengths, and may support parity or error detection and correction (EDC) with additional memory overhead. Different board layouts, control circuitry, packaging, and bus protocols are also used.

The memory-subsystem design is directly related to the price/performance of the entire computer system. Low-end machines generally have the main memory located on the motherboard. They provide for add-on memory that is accessed by a local memory bus or the system backplane. High-end systems often have separate memory boards that may be added in large quantities depending upon the required configuration.

This chapter contains a collection of material intended to give the new memory designer, as well as the seasoned professional, information to help make memory-subsystem design easier. There is a section on DRAM basics including discussions of special DRAM types, special access modes, and refresh types.

Two sections are chapters selected from *The Designer's Guide to Add-on Memory* from Clearpoint Inc. "Understanding Memory Design" covers the fundamentals of how data is accessed and stored, the different system and component technologies available to accomplish this now and in the future, and a general overview of system integrity. "Designing for Reliability" explains the different options available for detecting and correcting hard and soft memory errors.

Three related industry trends strongly support the case for increased EDC capability in today's high-performance systems, from desktop machines through mainframes. The first trend arises from the need to support sophisticated operating systems and applications software. The result is a requirement for more dynamic memory. With each additional bank of memory, there is an increased probability of soft errors, which increases the need for system protection.

The second trend is driven more by the business aspects of computer design. To remain competitive in today's marketplace, data-processing systems require higher density DRAMs to provide more data storage capacity in the same or less amount of real estate, at reasonable prices. As memory-feature sizes are decreased to meet higher density requirements, the probability of both hard and soft errors increases; smaller capacitive cells are more susceptible to bit complementing due to alpha-particle bombardment and electrical noise.

The third and last trend emanates from the user's desire to have a reliable system: one that crashes only rarely, if at all. Today's computer-literate consumers are demanding the security provided by EDC to prevent the microprocessor from attempting to execute or transfer erroneous instructions or data.

DRAM BASICS

DRAMs store data in 1-bit cells. One or more cells may be accessed in one data transfer, depending on the organization of the DRAM. Cells are arranged in square grids, with each cell having a specific row/column grid position identified by a bit address consisting of a row address and a column address.

DRAM row/column addresses are multiplexed on one set of address pins. Row addresses are latched on the falling edge of a Row Address Strobe $\overline{RAS}$ and column addresses are latched on the falling edge of a Column Address Strobe $\overline{CAS}$. A Write Enable WE signal is used to indicate whether a cycle is Read or Write: Low during Write cycles, High during Read cycles.

DRAM cells are capacitors while static RAMs store bits in transistor cells, where voltages change only during Write cycles. Thus, voltages are *static*, hence the term *static RAM.* On the other hand, voltage levels "leak away" over time from DRAM capacitor cells, which therefore require refreshing at regular intervals to maintain adequate voltage levels. The term *dynamic RAM* comes from this constantly changing cell voltage.

Although static RAM access times are faster and interfaces are easier to design, DRAMS offer the clear advantages of small cell size, small package size, and lower cost.

DRAM Types and Accesses

There are a number of special-access DRAMs available that help reduce memory access time when used in a particular access mode. The special-access mode is a feature available to the user in addition to the normal RAS/MSEL/CAS fundamental accesses. The Multiplexer Select MSEL is a dual-function input to a DRAM controller, used to determine whether the address to the DRAM is a row or column address.

Special-access DRAMs normally command a premium price. However, they can more than compensate for this by appreciably reducing the memory cycle time and enhancing the system performance. The basic choice is dictated by the system configuration and its application, the main objective being enhancement of the overall system performance at a given cost.

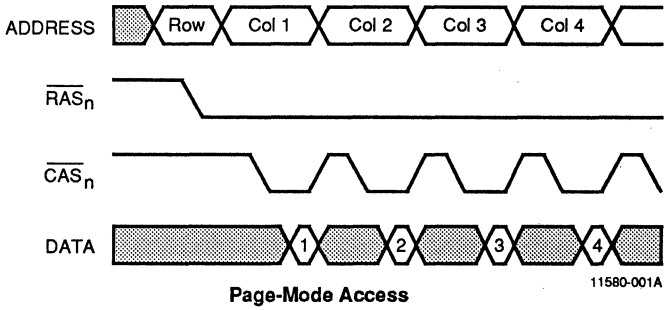
The following special-access DRAMs are discussed here.

- Page Mode
- Enhanced Page Mode or Fast Page Mode
- Static Column Mode
- Nibble or Ripple Mode

Page-Mode Access—performed with regular DRAMs

Page-mode access provides for fast random access of locations within a page, i.e., DRAM row, by saving the $\overline{RAS}$ precharge time for every access within the page. It starts as a normal access with a $\overline{RAS}$ time for the initial access on the page. All subsequent accesses on the same page require only the assertion of the $\overline{CAS}$ input. At the end of the initial access, $\overline{CAS}$ is deactivated while $\overline{RAS}$ is held active. For subsequent accesses within the page, a new column address is placed on the address inputs of the DRAM and $\overline{CAS}$ is asserted, thus initiating the page-mode access. The access time is calculated from the active edge of $\overline{CAS}$. A $\overline{RAS}$ precharge delay is only incurred for

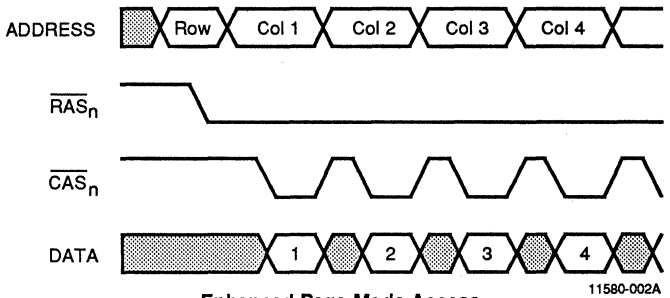
accesses on pages other than the current one, i.e., whenever $\overline{RAS}$ is deactivated. Page-mode accesses may be non-sequential; i.e., as long as the row address is unchanged, *any* column address may be selected in any order.



Page-Mode Access

Enhanced Page-Mode Access—requires page mode DRAMs

Enhanced page-mode access is similar to page-mode access, in that it provides fast random accesses to locations within a page by eliminating the $\overline{RAS}$ precharge time for page accesses after the initial access. However, it is faster than a standard page-mode access because the next access is started as soon as a new column is placed on the DRAM address lines, rather than starting from the $\overline{CAS}$ active time. $\overline{CAS}$ still latches the column data and acts as an output enable, but the page access starts from the column address change, rather from the active edge of $\overline{CAS}$.

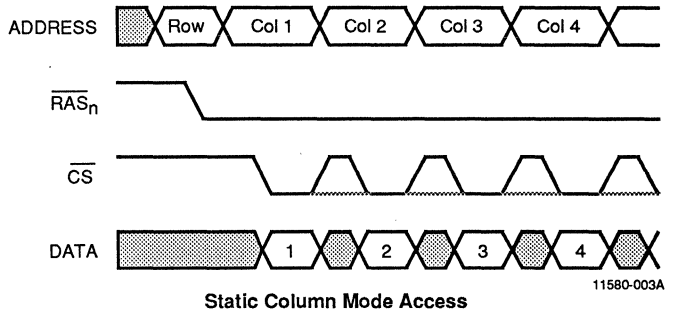


Enhanced Page-Mode Access

Static Column Mode Access—requires static-column DRAMs

Static-column-mode access is also similar to page-mode access, in that it provides fast random accesses to locations within a previously accessed page. It is even faster than enhanced page-mode access; since $\overline{CAS}$ is not toggled during Read accesses, $\overline{CAS}$ precharge time is eliminated. The column access is started as soon as a new address is placed on the DRAM address inputs. The column address is not latched, but must be valid for the duration of the access. Chip Select $\overline{CS}$ acts as an output enable only; it does not latch the column address.

As in page-mode access, the $\overline{RAS}$ access delay is incurred only on the first access on a page. Subsequent accesses on the same page require access time from the column-access change. A $\overline{RAS}$ precharge delay is incurred when an access is to a page other than the current one, i.e., whenever $\overline{RAS}$ is deactivated.

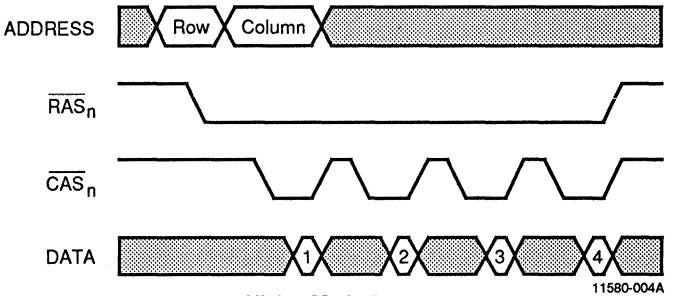


Static Column Mode Access

Nibble-Mode Access—requires nibble-mode DRAMs

The nibble-mode access provides for fast random access of four locations in modulo-4 order, i.e, 2,3,4,1 or 4,1,2,3, etc., with only one address from the system. The remaining addresses are generated internally by the DRAM. This frees up the address bus while the memory is being accessed. The falling edge of $\overline{CAS}$ initiates the next access.

As in the page-mode access, the $\overline{RAS}$ access delay is incurred only on the first access of the the nibble; the subsequent three accesses require only $\overline{CAS}$ access time. A $\overline{RAS}$ precharge delay is incurred between nibble accesses, i.e., whenever $\overline{RAS}$ is deactivated.



Nibble Mode Access

DRAM Refresh Types

To maintain data integrity, i.e., prevent bits from changing state, all DRAMs must be refreshed within a fixed time, usually 4 ms. Hence, all rows need to be accessed at least once in 4 ms. Refreshing a DRAM row refreshes all the locations in that row.

Interleaved and Burst Refreshes

DRAM refreshes may be interleaved between memory accesses every so often to meet the above condition. This is called interleaved refresh. Another option, called burst refresh, is to refresh all the locations in a continuous burst before the maximum time between refreshes.

An intermix of the above operations may also be performed, in which case a fixed number of burst refresh cycles may be performed between fixed intervals of time.

RAS-Only Refresh

The simplest type of refresh operation, called RAS-only refresh, is performed by placing the row address on the address input lines and activating $\overline{RAS}$. It can be performed on all types of DRAMs. All the banks of the DRAM array can be refreshed simultaneously using this method.

When operating on more than one bank of DRAMs, the $\overline{RAS}$ inputs of all the banks can be staggered by a clock cycle. This type of refresh timing is called staggered refresh timing. Staggered refresh helps reduce ground bounce and overshoot/undershoot generation associated with driving high-capacitive and inductive DRAM loads. It requires less power than refreshing all banks at once.

CAS-Before-RAS Refresh

Using CAS-before-RAS refresh, the row-refresh address is generated internally by the DRAM rather than generated by an external DRAM controller. The active edge of $\overline{CAS}$ increments the on-chip refresh counter; $\overline{RAS}$ then initiates the actual refresh operation. This type of a refresh operation can be performed only by DRAMs supporting this feature.

Hidden Refresh

A third type of refresh is called a hidden refresh. The $\overline{CAS}$ signal holds the data active from a memory access while a row-refresh address is placed on the address inputs and a $\overline{RAS}$ signal is activated to perform a refresh. Hidden refresh has minimum system impact, since all or most of the refresh cycle is overlapped with an access to another memory or I/O device. This type of a refresh operation can be performed only by DRAMs supporting this feature.

Refresh with Scrubbing

A fourth type of refresh, called refresh with scrubbing is performed on DRAM arrays in systems using error detection and correction. In this type of operation, an error detection and correction operation is performed during a refresh cycle. If the memory array has four banks of DRAMs, a $\overline{RAS}$-only refresh is performed on the corresponding rows of three banks; simultaneously, a Read/Modify/Write R/M/W cycle is performed in the other bank. The location undergoing error detection and correction is systematically cycled through the entire memory so all locations are checked for errors.

The refresh with scrubbing operation detects and corrects all single-bit soft errors. This reduces the probability of accumulating single-bit soft errors that result in multiple-bit errors, which are uncorrectable. (See further discussion on page 2-25)

Understanding Memory Design

The performance of memory is a function of many things, some related to the memory board design and some not. The purpose of this section is to identify the system design elements that affect the performance of memory. While providing a brief overview of how systems operate with memory, this section will also focus on the broad category of bus efficiency. This explanation of the basic elements of memory will include a description of major design issues such as the width of the data bus, the clock speed, bus protocols, array vs. on-board memory control, and ECL (emitter coupled logic) vs. TTL (transistor-transistor logic).

FOR THE MEMORY NOVICE: AN OVERVIEW

In simple terms, the bus on a computer system is a set of electrical connectors or wires attached to the computer backplane (printed circuit board, generally in the back of the computer cabinet). The bus connects to the various elements of the computer system (Central Processing Unit, system memory, disk storage, printers, terminals, etc.) to allow the transfer of electrical signals between the different parts of the system. Typical transactions include the transfer of instructions from the CPU to read data from memory (find it for processing by CPU); to write data (take processed data from CPU back to memory) or to enter data from a terminal or transfer it to disk storage.

The term "bus" is often used broadly, to mean both the hardware (the physical connectors and cable on the computer's backplane) as well as any resulting design constraints (the size of the data bus or the bus protocol). The bus hardware is usually made up of electrical "paths" — designated pins on the connectors — dedicated to transferring data, addresses and/or clock signals along these "paths."

The signal is transmitted according to one of two major timing methods, or protocols: synchronous or asynchronous. The choice of bus protocol determines which timing method is used by the CPU in recognizing electrical signal changes on the bus, thereby coordinating when the various parts of the computer "communicate."

Depending on the bus width and clock cycle time, data is transferred at varying rates. The total configuration of bus width, cycle time and protocol is the major determinant of bus performance.

This chapter explains these elements and how they impact memory performance.

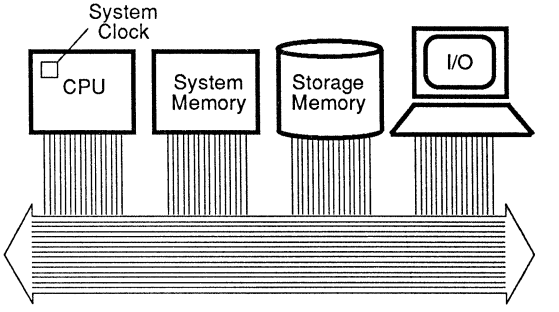
Width of the Bus

The width of the bus is simply the number of signals on the bus that are dedicated to transferring information. It is usually expressed in bit transfer capability, i.e., 8-bit, 16-bit or 32-bit wide bus. Ideally, the width of the bus should equal the width of the internal processor data word. Otherwise, buffering or multiplexing is necessary to compensate for the different data word size. With most of the new high-performance processors like the 68030, the 80386 and the MicroVAX offering a 32-bit word size on the processor, it is much more common to see 32-bit wide buses. One of the many differences between the MicroVAX I and II is the width of the bus going from the 16-bit wide Q-bus to a 32-bit wide local memory bus.

Clock Speed

Many computers have a system-wide clock that sends out high-frequency "ticks" (or cycles) by which all internal system events are coordinated, including transactions occurring on the bus. The clock frequency is expressed in megaHertz (MHz), each frequency unit defining one clock signal cycle. Clock speed is also dependent on the physical length of the bus, because of noise and transmission line requirements. Typical clock speeds range from an IBM PC's slow 4.7 MHz (or 4.7 million cycles/second) to 33 MHz in a VAX 8650.

Many system designers specify their system clocks to run well beyond the capabilities of current hardware to facilitate future growth and upward compatibility. In asynchronous bus machines (see next section for explanation), clocks may run at different speeds in different parts



System Bus—16 Bits Wide

Reprinted with permission from Clearpoint, "The Designer's Guide to Add-In Memory", Third Edition.
© 1989 Clearpoint Research Corp.

of the machine. This technique allows individual hardware devices to benefit from faster speeds as their individual design requirements permit. However, the limitations of other slower hardware may prevent the speed improvement from being fully reflected in the overall system performance.

Designing a bus for extremely fast clock rates takes attention to details such as signal transmission theory and adequate signal termination. If a bus is well designed, it should be able to improve as fast as the processors can. Many upgraded systems, such as the VAX 8650 upgrade from the 8600, amount to little more than running at a faster clock speed. If all the devices and the bus are capable of running with a faster clock, this is no problem. In some low-end systems, such as the PDP 11/73 and 11/83, customers simply changed the crystal on their systems to achieve substantial improvements in performance.

Bus Bandwidth

Bus bandwidth refers to a bus's maximum capacity for transmitting data. In simple terms, it is analogous to determining the volume of water through a pipe if you know the speed of the water and the pipe's diameter. Bus bandwidth is determined by its data transfer cycle time (x bytes/nanosecond) and its width (16 bits or 32 bits), and it is expressed as total data transferred per second (i.e., 1/2 kilobyte/second). The significance of a bus's bandwidth is that a bus with comparatively slow cycle time but with a wide bus width (or vice versa) can still have a competitive data transfer rate. For example, the Micro-VAX II has a wide 32-bit memory bus and a relatively slow 400 ns memory cycle time, but still is fast enough to keep up with the processor.

Bus Protocols

Synchronous Bus

This term refers to a timing method for synchronizing the transmission of data over the bus via regular signals determined by the system clock. The clock signal determines the precise and regularly occurring moment that the bus hardware will recognize and act on a signal level change (see timing diagram 1 at right). In a synchronous bus, the clock speed is perhaps the single most important determinant of bus performance.

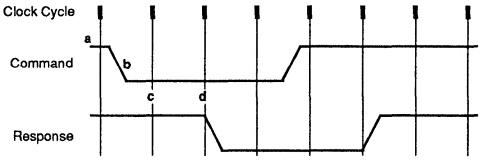
Asynchronous Bus

In contrast, an asynchronous bus has no system-wide clock. The bus hardware can recognize and act on a signal state change the moment it occurs (see diagram 2). Whether a bus should be synchronous or asynchronous is often hotly debated, each viewpoint with valid points to offer. Much of the debate focuses on the relative merits of the Multibus II (synchronous) vs. the VMEbus (asynchronous); however, the arguments are relevant to all buses.

The synchronous buses (like the Multibus II and the BI) are generally considered to yield higher performance than the asynchronous buses. In general, the design criteria for a "handshaking scheme," acknowledging that a device is ready to transfer data is greatly simplified in a synchronous environment. On the other hand, a disadvantage of synchronous buses is that a change in a signal cannot be recognized until the next clock edge (see diagram 3). A small increase in the speed of another device on the bus will have no overall system performance benefit unless it is at least one full clock cycle faster.

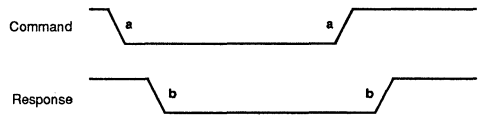
There are other pluses of a synchronous bus. One reason is that it is easier to implement concurrent cycles (several cycles taking place over the bus concurrently) on a synchronous bus. Likewise, the greater degree of control enforced by the timing specifications generally results in fewer compatibility problems down the road.

1) Synchronous Bus - (a) At leading edge of each clock signal, other signal paths are sampled for a state change (i.e. high to low). (b) A signal change that occurs during a clock cycle cannot be sampled until (c) the next clock cycle. (d) The resulting signal change (response) caused by the command signal is implemented at the next clock cycle.



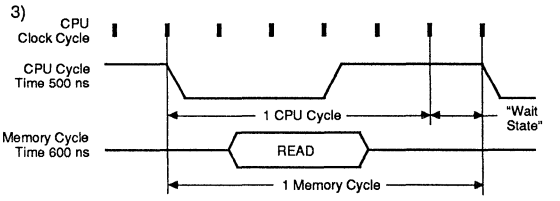
Summary: Signal levels cause state changes.

2) Asynchronous Bus - There is no system clock. (a) Upon a change in the command signal, (b) the response signal reflects a state change.



Summary: Signal edges cause state changes.

3)



Summary: CPUs can incur "wait states" (wasted CPU time) when the memory cycle is longer than the CPU cycle.

Synchronous and Asynchronous Protocols

Asynchronous buses are advantageous for other reasons. First, there is no waiting for clock edges before signal changes are acknowleged on the bus. As a result it is easier to take advantage of faster memory chips. The difference between the time when a device on the bus is ready to respond and the time a clock cycle strobes that response is called "clock latency." Since asynchronous buses do not have clock latency, there can be a real advantage to speeding up access times.

Other disadvantages include the increased possibility of design complexity resulting in performance inefficiencies and design errors. An asynchronous protocol is only beneficial if the designer can effectively mix different cycle times to enhance performance. In addition, a bus that is too complex may cause compatibility problems down the road. Anoher disadvantage of an asynchronous bus occurs when devices that plug into the system are synchronous (i.e. processors, peripheral interfaces, input/output devices). The bus signals must be synchronized to the local device clock, resulting in an additional time-consuming layer of activity and slowed performance.

Memory Processing Performance

Bus efficiency and microprocessor speeds are critical to system performance, but memory plays a significant role also. Its design is just as critical to overall performance.

Read Access Time — The most common measure of memory board performance is access time. Generically, access time is defined as the time from when the processor or other device makes a request for data at a given address to the time the memory board responds that it is ready to send data. This definition is subject to a fair amount of vendor variation. DEC, for example, defines access time on the Q-Bus in such a way that many vendors offer memory boards with access times faster than 100 ns. This is despite the fact that the memory chips used on those boards have access times of 150 ns.

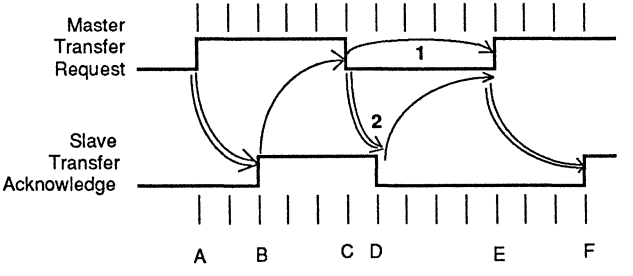
This paradox is a definitional quirk; the boards send the response signal (TRPLY) to the processor in advance of the data actually being ready. Since by the time the processor or other device is really ready to receive data, the memory board will be ready to send, this is allowable. The statistic is meaningful for comparison purposes with other Q-Bus memories, but is totally meaningless for comparisons across buses. In addition, it has little to do with performance on the Q-Bus. Almost all memories with access times less than 150 ns perform nearly identically on the Q-Bus.

Write access time is defined as the time from when a device sends data to memory until the next time it can send data. From the memory board's perspective, the data must be latched and a signal sent back on the bus indicating the data has been received. Since this is very straightforward, most write access times are very fast and do not vary considerably among vendors.

Cycle Time — The time from when a device makes a request for data until the next time a request for data will be acted upon by the memory board is defined as the read cycle time. Frequently cycle time is a better indicator of actual memory board performance than access time, especially in block transfers where the memory board is likely to be the constraining device.

In many cases design engineers optimize for access time and the result is a very slow cycle time. Which is more important depends on the application: I/O-intensive applications with lots of sequential reading and writing need a fast cycle time, while processor intensive tasks with highly random reads and writes depend more on access time.

Write cycle time is similarly defined as the time from a request to send data until the next time the memory board is able to write data.



The timing diagram above illustrates both access time and cycle time. Access time is defined as the slave receiving all signals at A until the slave sends back a response at B. Cycle time is from B to F including the full time of the subsequent access from E to F. Path 1 and path 2 indicate two popular "handshaking"schemes for master and slave devices.

Read-Modify-Write — The time from a read request until the memory board has latched the modified data and released the bus, including the time the processor needs to perform the operation, is defined as read-modify-write. Since this is highly processor-dependent, it is not often used as an indicator of memory board performance per se. It is a useful indicator for system-level comparisons.

MEMORY DESIGN FOR IMPROVED SYSTEM PERFORMANCE

BUS EFFICIENCY

Over the years, techniques have evolved that increase bus efficiency, i.e., decrease the incidence of bus transactions encountering wait states. These techniques show up over and over, in combinations or individually. It is important to understand the principles by which they work and interact with each other.

Multiplexing Data and Address

Multiplexing, or muxing as it is frequently abbreviated, is the alternate use of the same bus signal lines for data and address. The purpose is cost reduction: by sharing lines, the total number of signal lines is reduced by the lesser of the number of address or data bits. On the Q-bus, for example, 22 address bits and 16 data bits are muxed on 22 signal lines. The overhead of muxing can be considerable: additional signal lines are needed to enable the coordination or handshaking to inform devices what type of information is currently on the bus. Typically, multiplexing entails a performance penalty because the same lines have to perform two jobs. In transferring information the address must first be transmitted, followed by the data. If more than one consecutive piece of data must be transferred, this method quickly becomes inefficient, although block mode transfers can compensate to some degree. Multiplexing is now widely used on very high-performance buses like the Multibus II and the BI, so it is not necessarily considered too slow.

Interleaving

Interleaving refers to the practice of moving sequential words in different memory arrays so that the transfer of subsequent words can begin immediately. Since the overhead of access times can be overlapped, the average transfer time in sequential accesses can be substantially reduced. Interleaving can be performed on a single board, where the memory banks are composed of two autonomous halves; or across boards in a system or array. Many different interleaving schemes are currently in use, from the two-way interleaving on the VAX 780 and IBM RT PC to 8-way interleaving on some Data General systems. However, increased interleaving does not result in a linear increase in performance.

Transfer Methods

More efficient transfer of data and instructions can speed up system performance significantly. Since most transfers involve sequential addresses, methods for moving consecutive words or blocks at one time are particularly beneficial.

Prefetching and Pipelining

Prefetching refers to the CPU's ability to anticipate data accesses and start data retrieval before it is requested. A common form of prefetching is to start two to eight accesses in parallel so that the second (to the nth) access is proceeding simultaneously with the period where the first address is valid. The first access is subject to normal access times (150-300 ns) while subsequent accesses appear to be 10-20 ns apart. Hence, it is possible to transfer eight words in the time it would normally take to transfer two.

Pipelining is similar to prefetching, but it usually prefetches instructions rather than data.

Page and Block Mode Transfers

Both of these methods of transfer pump large bursts of data to and from sequential addresses. Instead of saving up a series of sequential addresses which will then be transferred consecutively, the CPU gives one instruction with a beginning address and the transfer takes that address plus the following page or block of addresses during the transfer. This technique is particularly useful in cases where the system is accessing the disk and multiple consecutive disk accesses would dramatically impair system performance.

Direct Memory Access (DMA)

DMA, an architectural feature of most buses, allows information to be read from disk and written to memory (or vice versa), via dedicated bus signal paths without interrupting the CPU. Some systems also add a DMA controller between the main memory and I/O devices, i.e., disk drives. An electronic device containing data buffers and logic circuits, the controller can control data transfer operations in place of the CPU, permitting simultaneous use of I/O devices and the system processor. In large database operations, DMA can be especially beneficial, particularly when used in conjunction with a memory cache.

Caching Memory

Caches have become one of the most widely used techniques to improve the performance of systems in general. Caches can be located on the processor, the memory or the peripherals (such as a disk drive), and serve as quickly accessible storage for interim processing results, soon-to-be-executed instructions or blocks of sequential addresses.

A cache memory on the processor is defined as a small (usually 16 KB to 64 KB byte capacity), very high speed (50 ns access time or less) memory that is tightly coupled with the processor. It is usually implemented in Static RAM or ECL to achieve faster access times. The use algorithm, unlike main memory, is implemented in firmware rather than in the operating system. The use algorithm is selected to achieve the maximum cache hit rate without tremendous overhead in "swapping." The optimal size of a cache is really an economic decision: it is driven by the difference in cost between main memory and cache memory. The larger the cache gets, the less likely it is to see enough performance increase to justify the dollar cost of additional cache memory. Hence, caches are generally quite small.

Caches on the disk controller are currently popular, now that a megabyte of memory can be condensed into one or two square inches. Implemented in DRAM, they are really just an extension of main memory with a different use algorithm. In sequential DMA, a cached disk controller can achieve extremely fast transfer rates.

Local Memories

Local memories are being used principally because of multiprocessing. In multiprocessing systems, there are often two or more processors operating concurrently from one memory. While the multiple processors can provide significant "number-crunching" functionality, the performance enhancement can be diminished as the system bus is overloaded by processor–memory transfers. To mitigate this problem, most multiprocessor architectures allow for each processor to have its own local memory on a separate bus.

Some systems completely eliminate system memory in favor of local memories. The difficulty in maintaining local and system memories has to do with control of processes, synchronization and concurrency. The benefits of better performance have made the solution of these problems a necessity for multiprocessing operating systems.

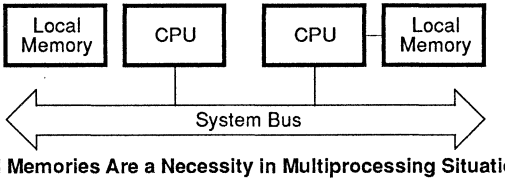
Dual-Ported Memories

Dual-ported memories are similar in many ways to local memories except that they can answer on either a local bus or the system bus. The advantage of a dual-ported memory is that it both increases bus efficiency by taking traffic off the system bus, and it has a faster access time than transfers on the system bus.

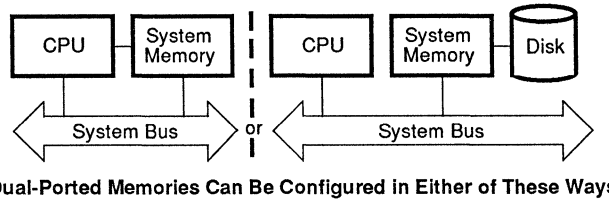
Dual-ported memory is typically situated on both the system bus and a local bus to the processor. However, there are dual-ported memories that sit on a local bus to a disk controller or other high-speed peripheral like an array processor. The principle of operation is essentially the same where ever it sits. Dual-ported memories are generally much more expensive than standard single-ported memory because of the considerable additional logic required.

Array versus On-Board Memory Control

One of the most common features of today's high performance systems, from the DEC VAX 8800 down to the IBM RT PC, is the use of separate memory controllers and array cards. The memory controller is that portion of a memory that performs addressing, timing, refresh



Local Memories Are a Necessity in Multiprocessing Situations.



Dual-Ported Memories Can Be Configured in Either of These Ways

control and arbitration, and EDC or parity generation and checking. By locating the memory controller on a separate card from the memory array, or simply directly on the processor card, expansion and manufacture of memory cards is greatly simplified. In addition, with the rapid advance in memory chip density, memory controllers are not obsoleted along with the array cards. The big advantage of separating the memory controller from the array is that the private memory bus can be greatly simplified from the full system bus. Typically it is much shorter; hence, the problems with noise are substantially reduced.

The second big advantage is cost: the memory controller accounts for one-third to one-half the cost of a memory card (with an on-board memory controller). When large numbers of arrays are required, eliminating the repetition of the memory controller can amount to considerable savings. While some recent buses still use on-board memory controllers (such as the VAXBI), the trend in system design is toward separating the controller from the array. The Sun 3/1XX series, for example, uses the high performance VMEbus for a system bus, but the memory hangs on a private memory bus with the memory controller on the processor card. Similar designs are also used in the RT PC and the VAX 8500, indicating the wide range of system sizes that are going in this direction.

ECL vs. TTL

Emitter Coupled Logic (ECL) is not a new idea — many RCA and CDC systems from the sixties pioneered the use of ECL for achieving better performance. ECL eliminates transistor storage time as a speed limiting characteristic, permitting much higher performance than is possible with TTL (transistor-transistor logic) circuits. ECL is now being used by DEC in the VAX 8600, 8700 and 8800 series; this has re-ignited interest by other manufacturers as well.

There are several problems with ECL that make it more difficult to work with than TTL. Careful attention must be paid to signal line lengths, due to the high speed and impedance characteristics of an ECL gate. Since propagation delays on a long signal line are a substantial part of the timing on an ECL differential amplifier, the design engineer must take this into account in laying out the board.

A second problem arises due to the power requirements of ECL. Compare the power needed for a VAX 8600 (a 4 MIPS machine) with a VAX 8300 (close to 2 MIPS), and you will find substantially more than twice the draw. ECL is the main reason; additional cooling is also necessary to keep the machine operating in a tighter temperature range.

ECL is generally only used in the busiest circuits. To achieve high densities in the system memory, standard dynamic RAMs are the technology of choice. ECL RAMs are too expensive and too hot to use for the large 8–128 MB arrays that are now commonly available for system memories. Instead, the memory controller is designed using ECL, and ECL-to-TTL conversion and buffering are performed on the arrays.

Memory Technologies and High Performance

Amid the hoopla of high performance microprocessors and "war of the buses," the advances in memory technology often are ignored. In fact, memory has come a long way since the core memories of the sixties. Semiconductor technology averages a fourfold increase in density approximately every 2-3 years. These improvements in densities and processes often pave the way for the microprocessors to follow. This section gives a brief review of the memory technologies currently in use, those in process and what that means for high-performance computers.

What Is Memory?

Memory is that part of the computer system from which the CPU reads and writes information which it uses in the execution of programs. Memory technologies are characterized by volatility, write-ability, and semiconductor technology. Non-volatile memory maintains its storage after power has been turned off. Write-ability is the extent to which data can be changed, and the means used to change data at a given location. Semiconductor technology refers to the actual semiconductor and substrate used for manufacturing the memory device.

Non-volatile memories are generally used for programs or data that must remain intact through power and system failures. Read-Only Memories (ROM), Erasable, and Electronically Erasable Programmable ROMs (EPROMs and EEPROMs) are sub-categories of non-volatile memories. Bubble memories and the newer FRAMs (Ferro-electric Random Access Memories) are still other types that offer non-volatility.

Volatile RAMs are either static or dynamic; dynamic (DRAMs) memories need to be "refreshed" periodically to maintain a charge in their cells. Every millisecond or so the system is put on hold while the cells are all recharged. Static RAMs (SRAMs) are more convenient (since they require no refreshing), however they require twice the number of transistors as DRAMs for the same storage capacity.

Within volatile RAMs, the major semiconductor technologies in use include N-channel Metal Oxide Semiconductor (NMOS), Complementary Metal Oxide Semiconductor (CMOS), Emitter-Coupled Logic (ECL), and Gallium-Arsenide RAMs.

GaAs RAMs are not currently widely available, although the military is funding considerable research in this area. ECL SRAMs are only used in cache memories because of their high speed and very high cost. NMOS has been the principal DRAM technology for many years, and is now being replaced at the high end by advances in CMOS technology. Megabit DRAMs are now almost exclusively CMOS.

CMOS is both faster and uses less power than NMOS. With density and cost approaching NMOS levels, CMOS is fast becoming the technology of choice.

The Meaning of Specifications — Most vendors quote typical, average or maximum specifications for their boards. "Typical" is supposed to be the speed for an operation characteristic of normal usage. "Maximum" is defined as the most it will ever be in any operation. "Average" is sometimes used synonymously with "typical" and sometimes to reflect an average of buffered and non-buffered operations. This is relevant when some operations will occur out of a buffer on the memory board rather than through the memory chips. What is truly "average" of course depends on what a normal mix of buffered and non-buffered accesses should be. Maximum times are supposed to be based on the worst-case timings of all the devices on a board. Since this number may be impossibly long, most vendors simply calculate a number they are confident will always be greater than actual usage.

Changes in the specified speed of a board do not always translate into actual improvement in system performance. This is because the memory board may not be the bottleneck in your system. If another area of the system is the binding constraint, speeding up the memory card will have little additional value. It may still be worthwhile to select the faster card, however, because over the life of the system it is likely that faster hardware will be added.

In a number of cases, the advertising of faster memory is truly misleading. The Q-Bus is a good example. Vendors achieve very fast access times by a variety of means, some in clear violation of the Q-Bus specifications. (Hanging logic directly on the bus without the appropriate drivers is not permitted, for example.) Regardless of how fast the access time is, if the processor or other device is not ready to receive data, the reduction in time is meaningless. This is especially true if cycle time was sacrificed to achieve these spectacularly low numbers.

Another example is the VAX 8600. Faster memory on the 8600 is feasible by definition, but it has little to do with actual system performance. Some vendors who advertise memory faster than DEC's "prove their case" by asking customers to switch off the cache on the processor. Some customers are actually impressed that their memory could perform faster if their system were non-functional.

In general, cached processors have done a lot to simplify the world of the memory vendor. When only 20% or 30% of the accesses are to main memory, the speed of memory is rarely the most important attribute.

Techniques Used to Improve Performance

The simplest way to speed up access time on a memory board is to use faster DRAMs. Most 64K and 256K DRAMs on boards today have a 150 ns access time. For a premium, there are good quantities of 100 and 120 ns parts available. The megabit DRAMs are yielding even faster parts; 100 and 120 ns access times are more common than slower devices. As die sizes shrink, the smaller circuits become faster and faster.

The access time of the memory chip is by far the largest component of the access time on the memory board. Hence the largest proportional gains in speed are achieved at this level. To achieve a gain of 30-50 ns through better design optimization and faster memory controller logic is very difficult.

At the memory controller level, faster memories are achieved through a variety of techniques. By focusing on the longest chain of logical gates, the design engineer attempts to whittle this down to the absolute minimum. He then can try using faster logic like CMOS (Complementary Metal Oxide Semiconductor) or ALS (Advanced Low-power Schottky) that pare each gate to the minimum interval. The trade-off at this level is between more expensive, faster parts and less-expensive, older, commodity-type logic.

Beyond this, there are various ways to "cheat" on timing, some of which are innocuous and some of which play with the ultimate reliability of the circuit under worst case environmental conditions. On the Q-Bus, for example, the bus is specified to be functional with backplanes up to 50 feet long. Since this is a rarity, to say the least, some designers have been known to make assumptions that the actual maximum is somewhat less than that. Also common is ignoring DEC requirements for specific drivers on all bus interfaces. If it works, many customers don't particularly care.

Another way to save time is by cutting margin to the bone. All devices have worst-case and actual performance

specifications. By skimping on worst- case timings, most circuits can achieve considerable improvements. If the circuit never fails, is this imprudent? The problem comes when the circuit only fails rarely or under adverse environmental conditions.

To some degree the customer is dependent upon the manufacturer for prudent design decisions. There is no substitute for adequate testing and design verification at the alpha and beta test stage to reveal a sound design. Testing at elevated temperatures (55–60° C) is also a good preventative measure, since device timings are at their slowest at higher temperatures. High temperature failures are the most common way to find out if an engineer "pushed the envelope" a bit too far.

Other techniques that are sometimes used on board level products are borrowed from system level techniques: interleaving and caching. On-board interleaving is sometimes implemented on a single card even when it is not specified in the system architecture. While the benefits of interleaving are not as great over a system bus, it can still boost performance.

Caching or buffering is another popular approach. Digital has implemented a two-stage cache on the MicroVAX 3X00; 1 Kb of 90 ns cycle on-chip cache, and 64 Kb of 180 ns cycle on-board cache. Intel has implemented memory board-based caching in their Multibus II product line.

Clearpoint uses a 64-bit buffer in an EDC chip set that improves access times on sequential reads and writes. The basic idea is to have a modest-sized buffer on the memory card that latches consecutive addresses each time data is accessed. Then if the subsequent operation calls for consecutive data, no additional access to the DRAMs is necessary. Since the buffer can usually be accessed in one-half to one-third of the time of a DRAM access, this considerably improves performance. Providing a large cache on the memory in addition to a cached processor, however, is not likely to yield incremental results.

Density

After performance, density is the most important feature of a memory card. Customers always want more memory in fewer slots with less power consumption. Greater density reduces the cost of additional memory, since the last megabyte on a board is always a good deal less expensive than the first. What follows is a compendium of techniques used to increase density.

DRAM Capacity

The most basic means of increasing density is to use DRAMs with a higher capacity. The DRAM manufacturers have obliged by coming up with denser memory chips year after year. Currently, a new generation of memory boards is entering the market based upon megabit DRAMs. This will supersede the 256K DRAMs that are now the bread and butter, just as the 256K DRAM replaced most of the 64K DRAM product. The inevitability of this progress is now old hat. The only issue is when the crossover occurs.

Typically sales begin on the highest density product as soon as a product can be delivered. With the 256Kb DRAMs this was practically instantaneous since the device is pin-compatible with the 64Kb DRAM. The megabit DRAM took longer since it is an 18-pin device instead of 16-pin like the 256Kb and 64Kb DRAMs. Demand shifts slowly, typically crossing over when the 4x density device costs 6-7 times as much. The crossover occurs before the price has decreased to 4x because the denser parts obviate 3 PC boards full of interface logic for the same density. In addition they are more reliable (because of fewer parts), use less power and free up slots.

Curiously, demand for the less dense parts continues for years after they are non-economic for design-ins. 64K DRAMs are still widely used even though they now cost more than 1/2 as much as 256K DRAMs. Once a part becomes more of a specialty item than a commodity, the price begins to rise.

DRAM Packaging

DRAMs are available in a variety of packages, each of which allows for a different packaging density. The most common package by far is still the Dual In-line Pinpackage (DIP), which looks like a standard IC with a row of pins emanating from each lengthwise edge. The popularity of this package is primarily the result of history — most computer products are still designed around DIPs, on standard printed circuit boards using through-hole technology, and soldered over a wave solder machine. DIPs are easy to handle and insert, their height is minimal, and they have been widely used in the past.

Surface Mount Devices (SMDs) offer much greater packaging density than DIPs and have been growing in popularity for this reason. Originally developed and popularized in the Far East for applications in toys and small appliances, SMDs require less than one-half the surface area of a DIP for the same capacity. In addition, the leads do not penetrate the PC board so that devices can be mounted on both sides of a card if the system will allow this. The leads are folded so that they lay flat either underneath the device or to the side of it.

Manufacturing of SMD-based boards requires a whole new set of equipment very different from that used in conventional assembly. Because the devices are so much smaller and the placement on the board is not

guided by insertion into holes, automatic assembly is preferred. The "pick-and-place" machines that are used for this are very sophisticated, incorporating the latest in robotics technology. SMDs are glued into place using a solder paste; instead of being waved, the boards are passed through a hot vapor chamber that melts the solder and makes the connections.

Eventually, SMDs should pass DIPs and conventional packaging in popularity, once all the manufacturing wrinkles are ironed out and the cost comes down. Right now, it is still a relatively expensive package and the assembly costs are still greater than manual assembly. Because of the tight tolerances, only small PC boards are currently used with SMD assembly. On larger cards, such as DEC's 16 MB 8600 memory, DEC chose to use SMD technology on the small daughter cards rather than attempt their use on the full-size PC board. Clearpoint uses SMDs on boards as large as the VAXBI form factor, as well as on the smaller cards for the MicroVAX 2000, Apollo DN 4000 and daughter card for the VAX 8800.

SMDs have spawned another package that has some of the advantages of SMD without any penalty except cost: Single In-line Pin packages or Single In-line Memory Modules (SIPs and SIMMs). SIPs are essentially small PC cards with a row of SMDs mounted on one or both sides, with normal insertion pins along one edge of the PC card. The SIP is then mounted on edge on a standard board with the pins penetrating the PC card just like any other conventional IC. With SIPs it is possible to achieve densities up to three to four times what is possible with DIPs; however, the height is sometimes greater than any other device on the card. In systems that allow for wide spacing among cards, this is not a problem. In most small systems these days, however, it is a concern.

Another issue is cost. When SIPs were first introduced, the military was the principal customer and prices were several times comparable product in DIPs. Now that the market has expanded and several of the major DRAM vendors have entered, the price is lower, but still 50% to 75% greater than DIPs.

The latest package to hit the market is ZIPs — Zig-zag In-line Pin Packages. ZIPs essentially take a normal DIP package and stand it on edge, with both rows of pins emanating from the same edge in a zig-zag pattern. ZIPs have a lower profile than SIPs, but still allow for twice the density of DIPs. The manufacturing cost is nearly identical to DIPs, however the pricing reflects the smaller market size since it is a new product.

Mitsubishi introduced the product in 1985 to little initial interest. The MicroVAX II created a niche for the product because ZIPs were the most cost-effective way to pack 8 MB of memory on a single card. Suddenly, demand soared and now most DRAM vendors are offering the package. As cost settles in to the DIP range, the ZIP market should continue to expand. The only difficulty seems to be keeping the devices in place as they pass over the wave-solder machine.

In the long run, it is difficult to tell whether SMDs will win or whether advances like ZIPs will extend the life of conventional technology. An often overlooked fact is that the wafers are now becoming a much larger fraction of the size of the package. Megabit DRAM SMDs are not much smaller than DIPs.

NEW TECHNOLOGY

A number of other developments have made greater densities possible. Custom and semi-custom Very Large Scale Integration (VLSI) devices are now popping up in many applications. VLSI has declined in cost substantially over the last two to four years. CAD/CAE tools now allow the rapid design and simulation of very large circuits. Clearpoint, for example, has a two-chip set for EDC control, comprised of 2400 and 5200 gates per device. The result is that what used to take 50 - 100 ICs to implement in standard logic now can be done with one or two VLSI chips. The space freed-up on the card can now be used for even more memory chips, or some additional functionality.

Some prognosticators are forecasting that FRAMs (pronounced F-RAM) will obsolete DRAMs. FRAM stands for Ferroelectric Random Access Memory. It is a new memory technology, based on an old discovery dating back to 1921. Promising significant enhancements over standard DRAMs – such as non-volatility, radiation-hardness and density – it could be the perfect memory technology.

The ferroelectric effect specifically refers to the tendency for certain crystalline materials to polarize spontaneously when an electrical field is applied, *and to remain polarized after the field is removed.* If the electrical field is reversed, the polarization is also reversed. The result is that the crystalline material can act as a capacitor with two distinct polarizations dependent on voltage levels. Since no current is required for the ferroelectric material to retain its polarization, it can act as a completely non-volatile digital memory capacitor. Storing either 1's or 0's in a ferroelectric element, the FRAM can be read by sensing the interaction of an applied field with the element's polarization.

Recently, two companies have announced breakthroughs in materials and processing necessary for commercially viable FRAMs: Ramtron Corp. (Colorado Springs, CO) and Krysalis Corp. (Albuquerque, NM). Ramtron currently produces a 256-bit non-volatile static

RAM which has a thin film of lead zirconate titanate (PZT) deposited over conventional semiconductor memory circuitry to form ferroelectric capacitors as part of the memory cells. The ferroelectric capacitors only come into play in the event of a power interruption. Krysalis likewise uses a conventional CMOS silicon wafer over which a thin film of ceramic ferroelectric material is deposited. The film is delineated to form non-volatile memory cells. In contrast to Ramtron's process, Krysalis' ferroelectric cells will serve as the primary storage element.

Both companies are still in early stages of development and promise much in years to come. Already a possible replacement for EEPROMs, they could eventually supplant SRAMs and DRAMs if they deliver the promised speeds and longevity. But don't hold your breath. FRAMs aren't forecast for widespread distribution until 1992.

Another development is the widespread use of Programmable Logic Arrays (PLAs or PALs), PROMs and EPROMs in situations that formerly used conventional logic. As hardware and software expertise merges, so do the approaches to engineering solutions. Programmable devices allow for very dense packaging combined with incredible flexibility. It is not uncommon today for EPROMs or PLAs to serve as upgrades to a board to enable compatibility with the latest developments on a bus. Whereas previously boards might be returned for extensive ECOs and retesting, replacing a socketed PROM in the field is a simple and inexpensive procedure.

DESIGN INTEGRITY

Engineers refer to design integrity wistfully as the "right stuff": the invisible glue that brings coherence and consistency to any system or product. Most seasoned engineers sit back and tell you it's like a work of art, "you know it when you see it." For the layperson, a few pointers would be helpful.

Is It the Solution to the Right Problem?

Any design or system is an attempt to solve a problem or a set of problems simultaneously. It should be very obvious when looking at a design "solution" what the problem is. Frequently, however, a design is borrowed from another setting and haphazardly applied to the given problem. If it works, little attempt at modification or optimization is made.

Examples abound in the computer world. The Q-Bus was originally a stop-gap inexpensive microcomputer bus intended for very low-end applications like process control and single-user systems. It is now the system bus on the MicroVAX 3X00. While it may be expedient for DEC to maintain a performance gap between its high- and low-end systems, the continued tweaking of the Q-Bus to apply it to a high-performance environment leaves a lot to be desired from the customer's point of view.

Was It Designed to Grow in Predictable Ways?

A design should be dynamic; it should be adaptable to the changing environment that is inevitable in most computer systems. Many of these changes are predictable, such as the continuing increase in memory chip densities. Does the design plan for growth? Are constraints built in because of compromises that will undoubtedly become bottlenecks in the future?

Are There Hints of Capabilities Well Beyond Those Required?

This is a corollary to the above question. Usually a growth path will be revealed by huge margins or extra capabilities on a board that are not warranted by the current environment. If these extras are costless or nearly so, they hint at the forward possibilities a customer should expect.

Other indications are the use of advanced technologies that solve old problems in different ways than normally done. VLSI in a unique application, a novel test process or reliability feature, all are evidence of original thought beyond what is called for in a design. It is also evidence of the vendor's capability to solve more difficult problems as they arise in the future.

Is It a Clean Design?

There is no substitute for the "right stuff" — a well-conceived design or an intelligently packaged system shows the attention to detail that is necessary in today's market. Compare several designs and look for the differences.

If Compromises or Trade-offs Are Made, Do They Make Sense?

Often compromises are made to solve one problem, creating new problems or costing too much somewhere else. Look for the details that don't make sense: special instructions where there should not be any; inconsistent performance measurements that are optimized for unusual environments; advertising key features of a product with no mention of those attributes that are known to be trade-offs of the feature. How many times, however, have you bought something that sounded too good to be true only to find out the "gotchas" when you took delivery?

Designed for Reliability: Keep It Simple

One of the most common phenomena in today's market is a very advanced design that never seems to work. A

good design should show evidence of addressing reliability as a central problem. How is the system likely to fail? Where are the design decisions that address these failure modes? Is it easy to test and stress these failure potentials in the quality control process?

What Happens When You Scratch the Surface?

Usually the design represents a coalescence of many competing concerns and constraints. In many ways, it is an optimization problem in many dimensions. The result is, when you scratch the surface (or formally, do sensitivity analysis on each of the variables) you expect to be forced back to the optimization point reached by the design engineer. Needless to say, sometimes this does not happen. Even a layperson can ask questions that delve into these design decisions, revealing the layers of the problem below. If you find questions that are not asked, and choices that seem to have been decided randomly, you are not looking at "the right stuff."

The High Performance Design

The result of a careful design is a product that stands the test of time. Understanding the design decisions for high performance put you, the buyer, in the driver's seat, right where you belong.

Designing for Reliability

Reliability is of paramount concern to all customers. The lack of consistently reliable product has spawned a growth industry within the computer market for highly reliable systems, such as Tandem and Stratus. Since many decisions on the system level have important ramifications for reliability at the memory board level, this chapter is devoted to clarifying the issue.

THE PROBABILITY OF ERRORS

There are two types of memory errors in dynamic RAMs: soft errors due to radiation-induced bit switching, and hard errors due to the unexpected deterioration of a memory chip. Soft errors cause no lasting damage to the memory board, but they do corrupt programs or data. Hard errors demand physical repairs. The principal cause of soft errors is alpha particle radiation from trace levels of uranium and thorium in the plastic packaging of a DRAM. Other cosmic sources of high-energy radiation also can cause soft errors. The energy necessary to displace electrons from an individual memory cell is a function of the size of the cell and the charge it carries.

As the cell size decreases by a factor of four, as in the change from a 64Kb to a 256Kb DRAM, the charge per cell only decreases slightly. The probability of a highly energetic particle hitting the device is a function of the size of the device. Since the 256Kb DRAM die is only slightly larger than the 64Kb die, the soft error rate only increases perhaps 50% with a 4X increase in density. The result is that a 256Kb DRAM is somewhat less reliable than a 64Kb on a per device basis, but more reliable on a per cell basis.

Hard failures occur when devices on the memory board fail. Memory chips have been known to fail partially or completely; all ICs in the bus interface logic also have some probability of failure. ICs in general exhibit a "bathtub" distribution for failures over time (see chart below), with a high initial failure rate while undetected IC defects are uncovered, then a low failure rate for a long time until it increases again because of wear-out.

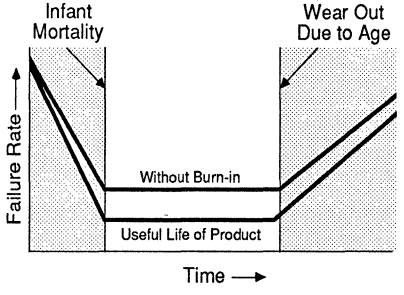
The probability of memory errors has been estimated by memory chip manufacturers, EDC chip set manufacturers, and various users. The estimates vary, not surprisingly, depending on what the writer is trying to sell. Robert McEliece, a professor at Caltech, wrote in *Scientific American* ("The Reliability of Computer Memories", January, 1985 252:1, 88–95) about memory reliability with no obvious predisposition. He used the figure for expected soft failure rate of a single memory cell of 1 every 1,000,000 years. In a 1 megabyte memory board with 8.4 million cells, this translates into a mean time before failure (MTBF) of 43 days.

Texas Instruments calculates the soft error rate MTBF based on their 64Kb DRAMs somewhat more optimistically; for an 8 megabyte system the MTBF was 33.4 days. Advanced Micro Devices, Inc., (AMD) a manufacturer of an EDC chip, estimates that a 16 MB system would have a MTBF of 13 days. Memory chip manufacturers measure reliability in terms of FITs—expected errors per billion device-hours. The 64Kb DRAM has a soft error rate of approximately 500 FIT, the 256Kb about 730, and the megabit DRAMs something close to 1000. Texas Instruments estimates hard errors to be roughly 1/5 to 1/3 as likely as soft errors.

Memory board manufacturers buy very few lots of memory chips in which the number of failures for 10,000 chips over a 72-hour burn-in is less than one; it is sometimes over five. While this represents the high initial failure of the early portion of this bathtub distribution of failures over time, it still implies a hard failure rate of about 1400 FIT (based on 1 per 10,000).

ERROR PROTECTION

System hardware reliability is achieved by including some amount of overhead for identifying when errors have occurred and either halting the system or providing some measure of fault tolerance. The method by which the system addresses error protection is usually consistent throughout the hardware and software. The chain is only as strong as the weakest link: if the hardware is designed to identify parity errors and the software has no provision for interrupts, the parity checking is basically worthless.



ICs in General Exhibit a "Bathtub" Distribution of Failure vs. Time, with a High Initial Failure Rate

Reprinted with permission from Clearpoint, "The Designer's Guide to Add-In Memory", Third Edition.
© 1989 Clearpoint Research Corp.

The three basic categories of error protection that are in use today are: parity generation and checking; error detection and correction (EDC); and complete redundancy. A fourth category — none of the above — is still seen occasionally. Parity checking is the most common low-end measure, since the cost is quite modest, performance is only mildly affected, and the software to support it is reasonably simple. EDC is the most common level of protection for minicomputers and mainframes, and seems to be migrating to both ends of the spectrum as hardware becomes less expensive and more reliable. Complete redundancy is the means used by "fault tolerant" systems that need to remain operational even when components of the system die. It is more than twice as expensive, since not only is twice as much hardware needed but the system software environment is considerably more complicated.

RUNNING BARE

The big question here is "Why?" Some OEMs still pay for reliable hardware and then fail to implement software that will take advantage of it. There is no more dismal feeling than having put in a week of work entering a database to find out that the system corrupted it days ago, and you've been copying bad files as back-ups. "Why didn't I know?" you may ask. Because your system was too cheap to put in any error protection.

"But I spent $100,000 on this system!" There's one born every minute.

PARITY GENERATION AND CHECKING

Parity checking would at least eliminate the above problem. Parity checking involves storing a bit with every byte of information that indicates the internal consistency of that byte. Generally this is as simple as determining whether there is an odd number of ones in the byte, and storing a 1 in the parity bit if there is. Then, every time that byte is accessed, transferred, stored, etc., the parity bit is compared with the byte to make sure it is still consistent. If not, a parity error is generated, and the system is generally halted and the location of the error is backed out.

How does the system know where the parity error was found? Through the Control and Status Register (CSR). The memory controller has a register that latches or stores the address of the word currently being accessed. When the system halts, the CSR can be interrogated using basic machine language commands to reveal the row and column address. If the system is restarted and halts again at the same address, this usually means that a memory chip has died and needs to be replaced. Byte parity uses a parity bit for every byte of data. In a 1 megabyte memory board, there would be 128 64Kb DRAMs used to store data and 16 used to store parity bits. This, and some additional logic, are the principal overhead of parity generation. Regardless of current prices, this is a very small price to pay.

ERROR DETECTION AND CORRECTION

Error detection and correction (EDC), also called ECC for error checking and correction, goes one step further by correcting single-bit errors that would otherwise halt the system. Double- and multiple-bit errors are also detected (unlike some multiple-bit parity errors) and treated like parity errors by the system.

EDC is valuable since the majority of errors are single-bit. EDC allows the continued operation of a system with only an entry on an error log to indicate that an error actually occurred. Hard errors due to memory chip failure are also corrected by the system, since each memory chip stores at most one bit from each word. Since hard errors are usually permanent, when the system is operating with a failed memory chip on board, it is essentially offering only parity protection for additional errors.

THE MECHANICS OF EDC

To the layperson, EDC seems almost too good to be true. Ask a casual observer to devise an error correction scheme, and he is likely to come up with two error-correction bits for every bit of data: store all bits three times; if one bit is different, go with the majority rule. This system requires three times as much storage as no protection, and gives incorrect information for double-bit errors. How, then, can an EDC system work that requires no more storage than parity memory (1 bit per byte), corrects all single-bit errors, and detects all double-bit and most multiple-bit errors?

EDC works by storing an error correction code (ECC) with each word that both identifies where a failure has occurred and corrects the error. The word size upon which error correction is performed depends on the specific memory design tradeoffs; typically, it is the word size of the system (usually 16 or 32 bits). But, it can use an ECC word size of 64 bits as well. Corrections can be made on either four 16 bit or two 32 bit words simultaneously.

In simple terms, the number of bits required to identify the location of an error in a word of N bits is log(2) of N. To understand why this is so, imagine trying to tell someone else where the error is with yes or no answers. They would ask "Is it in the first half?" then "Is it in the second quarter?" etc. Each time the possible locations would be halved.

	Check Bits Required			64Kb Chips Required for 1MB of Memory	
Word Size (bits)	EDC	Parity*	Word Size (bits)	EDC	Parity*
8	5	1	8	208	144
16	6	2	16	176	144
32	7	4	32	156	144
64	8	8	64	144	144

*Assumes one parity bit per byte.

Number of Bits Required for EDC and Parity

Additionally, two bits are required no matter how many bits are in the word in order to allow for errors in the check bits and to correctly diagnose double- or multiple-bit errors.

While not delving too deeply into the mathematics, it is obvious that the log(2)N formula also has to include the check bits. Using the 64-bit word as an example, assume k check bits are required. Then $k \geq \log(2)\ (64+k)$.

Correctly diagnosing multiple-bit errors requires an additional bit. Operation-ally, only odd combinations of one's (1, 3, or 5) in the ECC are used to diagnose problems; even combinations indicate that double- or multiple-bit errors have occurred. If only one bit is a 1, this implies that the checkbit is in error. If 3 bits or 5 bits are one's, then the designated data bit is in error. If 2, 4 or 6 bits are one's, then there has been a double-bit error (or even multiple-bit error).

PARITY vs. EDC: A COMPARISON

The effect of EDC on reliability is substantial. Whereas with parity a single-bit error causes an interrupt, with EDC it takes two errors within a word to cause an interrupt. The MTBF due to two soft errors in a word is about 600 million years. McEliece estimates the MTBF of a one megabyte EDC board due to soft errors to be 63 years; for a parity memory the MTBF would be 35.7 days. There are, of course, more probable ways of crashing an EDC memory. Only the memory array is protected against hard errors. The failure of any of the interface ICs is uncorrectable. Even so, the MTBF of an EDC memory card is at least an order of magnitude greater than a parity memory; AMD estimates that a MTBF factor of 50 to 60 is expected.

The protection against hard errors is very dramatic: it is possible to pull a memory chip off the board and have the board continue to operate. This is because memory arrays are organized in such a way that each memory chip stores at most one data bit in a word. If one of the memory chips dies, it can only effect a single-bit error in a word, and is therefore correctable. (Note: this is only true with x1 DRAMs; 64K x 4 or 256K x 4 organizations can store more than one bit per word in a single DRAM.)

THE IMPORTANCE OF THE ECC WORD LENGTH

The chart above shows the number of check bits required for EDC compared to the number of parity bits required for byte parity checking. This corresponds directly to the number of memory chips that are required for a given capacity of memory. For example, an EDC 1 MB memory using a 32-bit word would require 3 more bits than parity on each row. Since there are 4 rows of 32 bits (using 64K DRAMs), this amounts to 12 extra memory chips required: 156 compared to 144 for parity.

However, the same number of memory chips are required for parity memory and for EDC when error correction is performed on a word of 64 bits (see chart above). Unless EDC is implemented with 64-bit word, EDC is more costly than parity in number of DRAMS required to support it.

Performance is another attribute of the board that can suffer when EDC is implemented. The logic to perform EDC is understandably much more complicated; additional gates require more time for execution. However, with a 64-bit error correction scheme, the overhead of error checking is only incurred once every four 16-bit words or once every two 32-bit words, in sequential accesses. Since 64 bits are latched simultaneously into a much faster register than the memory chip, the register acts as a small cache. As a result, there is a performance enhancement which effectively offsets the increased time required for the additional logic.

Overall, EDC can translate into a significant improvement in error protection, and, as long as the EDC is implemented with a 64-bit word, the cost and performance tradeoff is insignificant.

Double-bit Error Detection and Correction: If One Is Good, Aren't Two Better?

Recently, some IC vendors have implemented logic in an EDC chip set that detects and corrects double-bit errors. This seems like a substantial improvement over single-bit correction and double-bit detection; in fact, the likelihood of two soft errors in one location is incredibly small and the benefit beset with operating negatives.

If an IC experiences a double-bit hard error, an EDC that automatically corrects a double-bit error is probably masking a bad IC that should be replaced. In addition, multiple-bit errors are never logged and the data integrity is jeopardized. For certain applications (satellites, for instance), where continuing operation is more important than complete data integrity, double-bit EDC will make a significant contribution. But, many system managers would prefer the assurance of data integrity.

EDC ON ARRAY CARDS

EDC Functionality

Most systems that implement EDC use separate memory controllers and arrays; the controller board implements the EDC while the arrays simply contain the extra storage to hold the check bits. Likewise, parity memory controllers have a CSR designed for parity. They only latch the address of data in the case of bad parity. Should an EDC memory card be placed in the array, the controller CSR only provides information on double- or multiple-bit errors that halt the system. To monitor the error correction on the memory array, a separate CSR is needed.

Currently, there are some separate controller/array systems that do not have EDC on the controller, but for which vendors sell EDC memory. This is now being advertised in the MicroVAX II marketplace. Since EDC works in a fundamentally different way than parity, customers should look very carefully at what they are buying. It is possible to decrease system reliability if the controller board does not provide adequate EDC functionality.

EDC Diagnostics

Another problem is diagnostics. If the system memory diagnostics are designed to test parity, they will yield little useful information on the functionality of an EDC array. Vendors of the EDC array offer a mechanical switch to toggle when diagnostics are being run, turning off the EDC. The result is that there is really no way to test the EDC array on the system; a custom tester is required to tell if the EDC is even working as advertised.

Worse, some boards only reveal single-bit errors using an LED, with no error logging or clearing through software. This requires opening the system box periodically to check for single-bit errors. If a soft error has occurred, the system must be powered down; the switch toggled; the cabinet closed; and the system powered up again before it is operational as an EDC memory.

Any realist knows that the probability of crashing a system increases by several orders of magnitude each time it is manually altered. An EDC system that requires a great deal of manual intervention is clearly more trouble than it is good. For those who live by the adage "If it ain't broke, don't fix it," make sure you know what you are buying.

REDUNDANCY

Redundancy is currently the ultimate error protection available for hardware. Many major vendors now offer systems in which all boards are "duplexed" — i.e., redundant — and the crash of either will not bring the system to a halt. Stratus Computer even has the system automatically call up remotely to order a replacement for a failing board. Customers only find out there was a failure when a replacement module shows up on their dock.

This is great, but curiously the systems still crash. Usually the reason is software: all the redundant hardware in the world cannot eliminate crashes due to bugs in system software. The operating system environments for redundant systems are necessarily more complex, resulting in a somewhat greater vulnerability to unpleasant encounters with wayward applications. Once an environment is stable, however, the added hardware reliability can reduce downtime to minutes a year.

Redundant systems are more than twice as expensive as simple EDC, because of the complexity of the operating environment. Still, with hardware rapidly declining in price relative to five years ago, the redundancy may be worth the cost.

Measurement of Reliability

Many people try to quantify reliability, and make meaningful comparisons based on these numbers. Because reliability involves statistical probabilities, talking in terms of individual boards or systems is meaningless. It is always important to remember that just because you can measure something, that does not mean that it is what you are looking for. The favorite joke on this runs:

A woman walking down the street one night comes across a man looking on the ground underneath a street light.

She asks him what he is up to, and he replies that he is looking for his lost car keys.

"Where did you lose them?" she asks.

"Over next to my car," he answers, pointing to his car up the street.

"So why are you looking over here?" the woman asks, incredulously.

"Because it's too dark to see anything over there."

Mean Time Between Failures

So goes the saga of measuring reliability. MTBFs are easy to compute, but splitting hairs over differences is really looking at the wrong issue. MTBFs are computed based upon the cumulative probabilities of failure due to wear-out of ICs and PC boards in a normal operating environment. When the probability of an individual component failing is very small (1 in a million years, or thereabouts), the probability of 1,000 or 5,000 such devices failing is approximated by the sum of the individual probabilities. To compute an MTBF, the manufacturer adds up the individual probabilities for all the ICs on the board, the PC board itself (based on the number of holes and layers), and each solder connection. This cumulative probability is the reciprocal of the MTBF.

The cumulative probabilities are very low, as one would expect. A memory vendor computes MTBFs from information provided by the suppliers of the components. Not that the suppliers have any conflict of interest, but they like to measure their component's reliability under circumstances that show it "in a good light." Usually, this means tightly controlled temperature and environment, complete static protection, and elimination of failures due to "abnormal" circumstances.

The good news is that the probability of failure in these circumstances is very low. The bad news is that "abnormal" circumstances are the cause of 99% of the failures. Poor static control in handling, improper soldering, conductive dust build-up, inadequate QC at the component level, poor quality printed circuit boards, corrosive desoldering fluids, incorrectly labelled ICs and discretes, out-of-revision programmable devices, improper packaging and shipping, and incorrect installation, to name a few, are all abnormal circumstances.

MEAN TIME TO REPAIR

Another index of reliability is the MTTR, since downtime is a function of both the MTTR and the MTBF. MTTR is simply the time it takes to repair a failure on a board. It is computed by estimating the probability of each failure mode and multiplying this by the expected time for repairing that failure. While this may be useful for the vendor to compute his labor costs in repair, it has very little to do with what a customer will experience.

Repair time depends primarily on whether a customer can perform on-site repair or whether the board must be returned to the factory. Most customers return all boards for repair, since this is really the best way of guaranteeing that it is done correctly. For customers willing to repair failed memory chips, the only issue is whether the memory is socketed or soldered-in. On-site repair of soldered-in memory is never a good idea.

MTTR then becomes dependent on vendor response time. If the vendor has implemented true 24-hour advance replacement, this becomes the MTTR.

REFERENCES AND SITE VISITS

In reality, there is no substitute for references and first-hand experience in determining vendor reliability. The vendor knows what his reliability problems are, and his larger customers will too. Vendors have been known to use a lot of poetic license in describing their return rates, so asking will not generally reveal the skeletons. Large customers, however, are usually willing to share their results. If a customer has a large sample size (200–500 boards) and a long history with the vendor you can usually assume they have seen some dirty laundry.

THE BOTTOM LINE

Reliability is an important factor in any system configuration. But the real bottom line is understanding what can or can't be done to ensure the maximum reliability for your investment. How much reliability is enough? It is probably a question that can never be answered entirely. However, understanding the risks is half the battle. With adequate knowledge, a reasonable decision can be made.

ERROR DETECTION AND CORRECTION SYSTEM ARCHITECTURES

Once the decision to use EDC has been made, a designer must choose how this capability will be implemented within his memory system. In general, there are two distinct implementations of EDC, commonly referred to as *Fly-By* and *Flow-Through* (Figure 2-1).

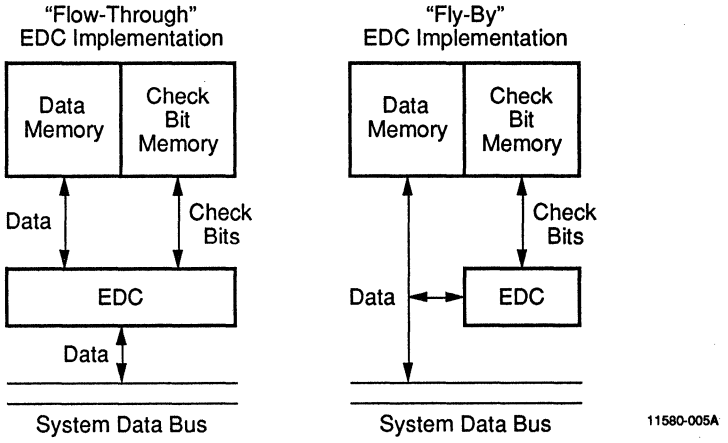
Fly-By

Fly-By, which is another term for a Check-Only Configuration, checks all words read from the memory to the bus for errors. If an error is discovered, it is flagged by the EDC and a recovery routine is initiated to complement the bit-in-error before the full data word is presented as valid. The erroneous word is gated off the bus and the corrected data substituted.

While this error-checking routine is being undertaken by the EDC circuit, the data words are simultaneously sent directly to the bus. The system always assumes that no error has occurred, a correct assumption in the majority of cases. This implementation provides maximum speed for most memory accesses; the cycle will occur without any delay from the EDC, since most words read from memory contain no errors.

However, if an error is detected, the CPU uses the Error or Multiple Error flags from the EDC to either interrupt the memory cycle or stretch it by adding wait states; this causes a significant throughput delay. At the designer's option, the corrected data may be written back to the memory if the EDC correction logic is enabled. In some instances, one may not wish to write the corrected data back to memory, such as in the case of a system employing memory "scrubbing" during refresh cycles (a more detailed explanation of memory scrubbing follows). In this case, the system automatically corrects the error later.

Figure 2-1



EDC Implementations

The Flow-Through method assumes each word is erroneous and completes a correction cycle during every memory access. This simplifies system timing because each memory cycle is identical. The Fly-By implementation only interrupts the memory cycle if an error is detected, and a recovery routine is initiated. This provides maximum speed for most memory accesses, since there is no EDC delay if an error has not occurred.

Commonly, a penalty of one wait state is the result of a correctable error. For non-correctable errors and instruction fetches, all high-performance processors have some form of pipelining or prefetch buffering. Since the corrupted instruction, at best, only gets to the decode stage of the processor before an interrupt is asserted, a recovery routine is executed before the error is encountered. For data accesses, invalid data is read into the processor. However, after insertion of an interrupt, the processor will *not* crash due to invalid data.

Using the Fly-By implementation, data Reads proceed as fast with EDC as without. Slowdown occurs only if there is an error. Even if the memory system has an error every hour, this would only occur once every 3-4 *billion* memory cycles. So even with a high error rate, EDC in the Fly-By configuration has essentially zero impact on memory-system speed. Fly-By can be implemented with the 680XX and 80X86 processor families in addition to many other CPU and system-bus configurations.

Flow-Through

A Flow-Through EDC implementation places the EDC directly between the memory and system data bus. Using this configuration, also called "Correct Always," all words from memory are assumed to contain errors and are routed first through an EDC circuit before reaching the bus. The EDC also assumes every word has an error and completes a correction cycle, which simplifies system timing. In this implementation, the impact of a memory error is negligible, for the EDC corrects the error without having to interrupt the system cycle. The EDC correction cycle runs at the same speed whether or not the data contains an error. All memory access cycles are slower due to the additional time required by the EDC. Usually, the Flow-Through configuration is used with microprocessors that have ample memory-timing budgets, or in systems that do not support bus-cycle restarts.

The increased speeds of today's EDC products are continually making this EDC configuration more practical for high-end systems, since error detection and correction has become an insignificant part of the total memory-cycle time. The Am29C660D, for example, detects and corrects an error in a 32-bit data word in a mere 18 ns. This is a five times improvement in speed over circuits available several years ago.

Summary

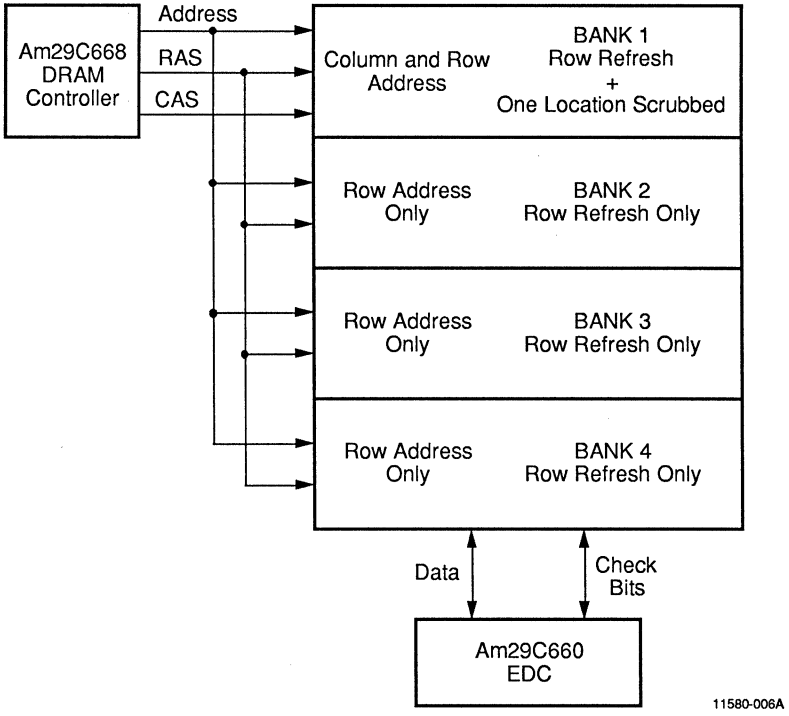
The two different EDC implementations have specific pros and cons. A Fly-By system requires more complicated timing, but will ultimately have the best performance. A Flow-Through system has simpler timing, but lower performance. The Am29C660 may be used to implement either Fly-By or Flow-Through EDC configurations.

Refresh with Scrubbing

"Memory Scrubbing" is a housekeeping operation implemented in hardware for checking the DRAM memory for errors during normal refresh operations. Hidden from the CPU, it is performed while no other memory accesses are being attempted.

On each refresh-with-scrubbing cycle, one memory word is read, checked for errors, and corrected if necessary before being written back to memory. If several banks of memory are refreshed simultaneously, all but one of the banks undergoes a standard row-refresh cycle, while a single selected word is scrubbed in one bank only (Figure 2-2). Today's sophisticated DRAM controllers, such as the Am29C668, contain scrubbing counters that keep track of the appropriate row, column and bank location of the word to be scrubbed. They also include EDC initialization logic that writes a known value to all memory locations during power-up. For a 16-Mword memory (2^{24} locations)

Figure 2-2



Memory Scrubbing

During scrubbing, one location of memory is checked for errors during normal refresh cycles. In this case, a single address location within bank #1 is scrubbed while the active row is refreshed in all four banks. The Am29C668 contains row, column, and bank counters that scrub all memory locations in succession via the Am29C660 EDC.

employing megabit DRAMs and one refresh every 16 µs, scrubbing the entire memory takes four and one half minutes, regardless of the word width.

When an error occurs, a Read/Modify/Write R/M/W cycle is performed. The duration of a R/M/W cycle is longer than a normal Read or Write cycle. During refresh operations, a row in each bank is accessed by asserting the Row Address Strobe RAS line. This refreshes all locations in that row. If an error is detected, a Write operation is performed within the refresh cycle; wait states may be required to extend the cycle. However, system reliability is increased because soft errors cannot accumulate in areas of memory that are not frequently accessed.

When performing RAS-only refresh without scrubbing, all four RAS lines are activated, but the CAS lines remain inactive. A refresh with scrubbing cycle activates all four RAS lines and a single CAS line. Correctable errors detected during scrubbing cycles are not reported to the CPU.

AM29C660 CMOS CASCADABLE 32-BIT ERROR DETECTION AND CORRECTION CIRCUIT

The Am29C660 EDC contains the logic necessary to generate check bits on a 32-bit data field according to a modified Hamming code algorithm. It can also correct the data word when check bits are supplied. It detects all single and double-bit errors as well as some triple-bit errors. Error conditions are flagged on the ERROR and MULTERROR outputs of the EDC. It can correct all single bit errors. For 32-bit words, seven check-bits are used. Check bits are generated in the "generate mode" to correspond to the contents of the Data Input latch. They are stored in a separate section of memory specifically set aside for them.

Each time a word is subsequently read, the EDC regenerates new check bits and compares them to the original check bits from memory. This is accomplished by XORing the two sets of check bits to create "syndrome" bits. If the two sets of check bits are identical, the syndrome bits will all be zero and no error has occurred. If the two sets differ, the EDC detects an error by generating non-zero syndrome bits.

The EDC then determines which bit has been wrongly complemented (in the "correct mode") and corrects the error by changing the bit back to its original value. The EDC can detect and correct errors in the data word and the original check bit representation if necessary. In addition, gross error conditions, such as the occurrence of all ones or all zeroes in the data and check bits can be detected.

The Am29C660 is expandable to operate in the 64-bit mode. Two devices may be cascaded to generate the eight necessary check bits for a 64-bit word. In both 32 and 64-bit modes, error syndrome bits are made available on separate outputs for error logging. Here, the locations of errors are stored separately for diagnostic purposes, so the locations of malfunctioning DRAMs can be determined.

The Am29C660 also includes two diagnostic modes in which diagnostic data may be forced into portions of the chip via software to simplify device testing and to execute system diagnostic functions. Using this feature, "dummy" data words and check-bit representations may be loaded into their respective input latches to generate different error detection and correction operations.

PROGRAM TO EVALUATE AM29C660 MULTIPLE ERROR DETECTION CAPABILITY

A program was written to evaluate the error-detection capability of the Am29C660. It simulates all the possible combinations of bit errors. The number of possible errors for a given number of bits in error is a combinatorial function of the number of bits in the word and the number of bits in error. If m is the number of bits in error and n is the total number of bits in the code word, then the total number of words is calculated as follows:

$$\frac{n!}{m!\,(n-m)!}$$

where $n! = 1 \times 2 \times 3 \times \ldots \times n$.

The program generates the data and check bits with the specified number of bits in error. The program calculates the check bits for the data and the calculated check bits are XOR-ed with the check bits generated in the previous step. If the resulting syndrome bits are zeros, the errors were not detected. If the syndrome bits are not zeros, they must be compared against the syndromes for a single-bit error through a look-up table. If the syndrome bits match a single-bit error, the word would be "miscorrected." The EDC thinks the error is a single-bit error and attempts to correct it. If none of the previous conditions are satisfied, the error is detected. This process is performed for all the possible combinations.

Table 2-1 summarizes the results of the simulation. The time needed to execute the program limited the number of bits in error that could be simulated. From the table, it can be seen that for any odd number of errors, all the errors are detected; but some are miscorrected. For an even number of bits in error, none are miscorrected; but some are not detected.

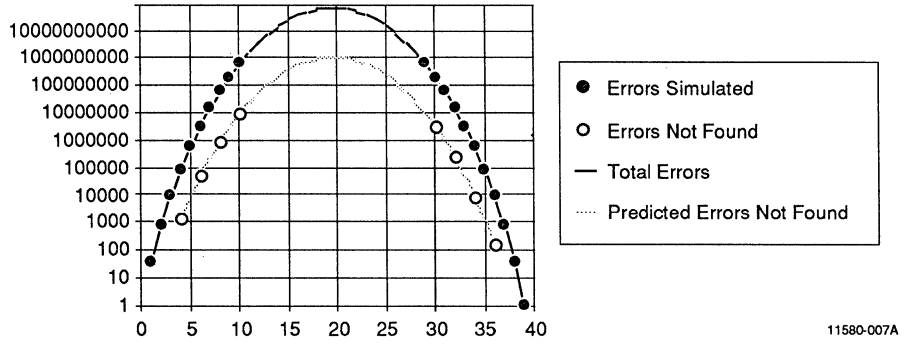
Table 2-1

Number of Bits in Error	Number Detected	Number Not Detected	Number Miscorrected	Number of Possible Errors
1	39	0	0	39
2	741	0	0	741
3	3619	0	5520	9139
4	80871	1380	0	82251
5	224577	0	351180	575757
6	3212143	50480	0	3262623
7	6012217	0	9368720	15380937
8	60560888	962860	0	61523748
9	82758472	0	129156660	211915132
10	625814596	9930800	0	635745396
29	248334700	0	387410696	635745396
30	208604844	3310288	0	211915132
31	24031156	0	37492592	61523748
32	15140312	240625	0	15380937
33	1275240	0	1987383	3262623
34	566845	8912	0	575757
35	32075	0	50176	82251
36	8983	156	0	9139
37	273	0	468	74
38	39	0	0	39
39	1	0	0	1

Two plots that extrapolate these trends were generated from the data. The first plot, Figure 2-3, is for errors not detected and only applies to data with an even number of errors. The percentage of errors not detected is approximately 1.6 % between 4 and 36 even-bit errors inclusive. There are no errors miscorrected for even-bit errors. The second plot, Figure 2-4, shows the number of errors simulated and the number of errors that were miscorrected and only applies to an odd number of errors. The percentage of odd-bit errors that are miscorrected is approximately 61% for words with 3 to 37 errors inclusive. For odd-bit errors, all errors are detected.

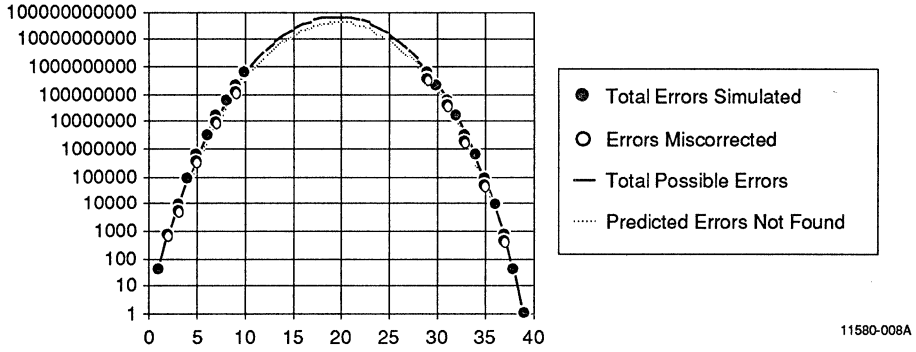
For nibble memories, assuming a single failure of one DRAM, the probability of detecting the error is 92.5%; the probability of not detecting an error is 1.5%; and the probability of miscorrecting an error is 6.0%. This assumes that the number of 1-, 2-, 3- and 4-bit errors are equally distributed, which should be the case for a total failure.

Figure 2-3



Number of Errors Not Detected for Even Bits in Error

Figure 2-4



Number of Errors Miscorrected for Odd Number of Bits in Error

MEMORY RELIABILITIES WITH AND WITHOUT THE AM29C660 EDC CIRCUIT

DRAM manufacturers express MTBF in terms of failures in time (FITs). One FIT represents one error in one billion (10^9) hours of operation. Toshiba, a major supplier of 1-Mbit DRAMs, claims a soft-error rate of 252 FITs (one failure every 3.97×10^6 hours). Clearpoint, a manufacturer of add-in memory boards for a variety of systems and buses, estimates the soft-error rate at approximately 1000 FITs (one failure every 10^6 hours). The actual value is somewhere between these two figures. An analysis using both FIT rates follows.

The MTBF in hours for each separate DRAM is $\frac{10^9}{\text{FITs}}$

The memory system MTBF in hours is $\frac{\text{DRAM MTBF}}{\text{\# of DRAMs}}$

The following table summarizes the MTBF for different memory sizes without EDC assuming a FIT rate of 252.

# of Memory Chips	Memory Size(Mbyte)	Memory MTBF(years)
32	4	14.1
64	8	7.1
96	12	4.7
128	16	3.5
160	20	2.8
192	24	2.4

The following table assumes a FIT rate of 1000 and no EDC.

# of Memory Chips	Memory Size(Mbyte)	Memory MTBF
32	4	3.6 yrs
64	8	1.8 yrs
96	12	1.2 yrs
128	16	326 days
160	20	260 days
192	24	217 days

With EDC, all single-bit errors are detected and corrected. The probability of a fatal two-bit error occurring depends upon several system considerations. The Am29C668 4-Mbit Configurable DRAM controller can be used to "scrub" the memory during refresh cycles. This insures the data integrity of seldom-accessed memory locations and increases the MTBF. If scrubbing is not used, the MTBF for the system depends upon when two soft errors occur in the same word. The approximation for the occurrence of two soft errors in the same word is given by the approximation to the birthday paradox*:

$$\text{(MTBF using 39 chips without EDC)} \times \sqrt{\frac{\pi * \text{Number of Memory words}}{2}}$$

*The birthday paradox is that, on the average, only 24 people need be asked their birthday before two people are found to have been born on the same day of the year. This is a well known problem in statistics.

The Am29C660 adds an overhead of seven bits for every 32-bit word. For a 32-bit EDC system without scrubbing, the system MTBF, assuming a FIT rate of 252, is:

# of Memory Chips	Memory Size(Mbyte)	Memory MTBF (Years)
39	4	14,907
78	8	10,541
117	12	8,607
156	16	7,454
195	20	6,667
234	24	6,086

The following table assumes a FIT rate of 1000 and EDC and no scrubbing.

# of Memory Chips	Memory Size(Mbyte)	Memory MTBF (Years)
39	4	3,757
78	8	2,656
117	12	2,168
156	16	1,878
195	20	1,680
234	24	1,534

These numbers, however, are overly optimistic because these tables disregard any hard failures. The Am29C660 and the extra memory chips also impact the reliability of the system, because there are more memories subject to failure. The actual increase in MTBF is more on the order of 50 or 60, *which increases the memory reliability in the worst case from 217 days to 30 years.* If scrubbing is used, the MTBF is increased even more.

An EDC requires slightly greater overhead than a parity system, seven check bits versus four parity bits, but the EDC offers a dramatic increase in reliability. Parity only detects errors. EDC, therefore, is a very valuable tool in increasing system reliability.

SYSTEM BUSES

The section of this chapter, "Understanding Memory Design," discusses system-bus efficiency, particularly as it relates to memory, referring to examples such as Q-Bus and Multibus II. It is important that the designer be familiar with the various available system buses, since he may need to design interface circuitry to meet a strict set of specifications. There are many choices: some buses are designed for specific systems from DEC, IBM etc.; others are vendor independent and offer open standards. Choosing the right bus is rarely easy. The designer must consider many factors: e.g., board size, connector type, arbitration methods, protocols, available semiconductor technologies. A brief overview of the most popular buses, by no means a complete listing, is given here.

The VMEbus

The VMEbus was developed by Motorola, in association with other companies, to provide an open architecture. Perhaps the most popular bus among OEMs, the VME offers 8-,16-, or 32-bit data and 16-, 24-, or 32-bit addressing and a 40 Mbyte/s bandwidth. The VMEbus is rapidly becoming the choice for military applications.

Numerous products are offered for use with the VMEbus, including almost all processors, memories and memory boards, controllers, error-detection and correction circuits, and other support products. The asynchronous VMEbus provides for block-mode and unaligned transfers; it is extremely flexible with minimal compatibility problems. Present plans include a 256-bit data path, 2-3 Gbyte/s bandwidth, 64-bit addressing and scalability.

VERSAbus was VME's predecessor, designed primarily for 68000-based systems. It is in limited use today.

Multibus* I and II

Multibus I is one of the most popular single-board computer buses and boasts an extremely large installed base in both the military and OEM markets. Developed by Intel, it offers a simple architecture: 16-bit data, 24-bit addressing, asynchronous operation and requires no multiplexing. It is still an excellent choice for 8- and 16-bit applications; however the growth path stops here. Intel was forced to meet the versatility demands of more sophisticated systems; the result is Multibus II.

Multibus II is a synchronous bus with five levels of embedded sub-buses. The main parallel-system bus is 32 bits wide and operates at 40 Mbyte/s; the local memory bus is even faster. Combined with a serial bus, Multibus II architecture offers a broad range of bandwidths in one specification. Synchronous buses are usually tightly specified to keep compatibility problems to a minimum and to make system design easier. However, the five levels of sub-buses, all with different clock speeds, complicate the design. To solve this problem, Intel offers a bus-interface chip set to standardize bus communication.

Multibus II offers a broad capability for tightly coupled, synchronous operation of many processors and shared devices to avoid the problems of centralized arbitration.

The NuBus**

The NuBus, another bus that supports 8-,16- and 32-bit addressing, is simple, flexible and easy to use. It is best known for its application in the Macintosh II***. NuBus has only one address space, as compared to three required by both Multibus II and VMEbus; only four control lines are required to define a transaction, compared to many more for Multibus and VMEbus. NuBus supports a 37.5 Mbyte/s data-transfer rate, as well as several types of DMA transactions.

A NuBus overhaul in 1990 may provide twice the current performance for 32-bit systems with a transfer rate to 80 Mbyte/s and improved specifications that will enhance its position in both the workstation and industrial markets.

AT Bus

The IBM PC, originally based on the 8088, is probably the most popular system ever built. There have been two major enhancements, the XT and the AT. The latter is based on the 80286 and offers a 16-bit data bus vs only an 8-bit bus on the XT and the PC. The AT architecture is adequate for single-user machines and continues to be a strong contender in the work-station marketplace. However, the AT bus is still only 16 bits wide, but the machine performance continues to improve with faster processors and peripherals. Most work-station vendors now offer some degree of compatibility through networking to provide for PC applications on larger machines.

---

* Multibus is a registered trademark of Intel Corporation
** NuBus is a trademark of Texas Instruments Incorporated
*** Macintosh II is a registered trademark of Apple Computer Incorporated

Micro Channel*

Micro Channel currently provides a low-end standard to improve upon the performance of the AT bus. It resides only in the PS/2* Model 70 and 80 systems. (See detailed discussion in the Application Note, "IBM PS/2 12-Mbyte Memory Board with Error Detection and Correction", Chapter 4). Micro Channel is an IBM-proprietary bus.

The Micro Channel architecture uses combinations of 8-, 16- and 32-bit connectors to implement a complex bus arbitration scheme for sharing address, data, and control lines without conflict. Bandwidth is 20 Mbyte/s.

EISA (Extended Industry Standard Architecture)

Based on the AT bus, EISA was designed by a consortium of computer vendors and is intended to be the answer to Micro Channel. It targets single-CPU environments and supports multiple-bus masters. EISA serves as a shared resource in a network of PCs and segments its architecture into memory and I/O buses. Bandwidth is 33 Mbyte/s.

The Q-Bus

For many years, DEC's Q-Bus has been the most popular OEM bus for low-end applications, such as process control and single-user systems. Originally designed for the LSI-11 microcomputers, it is still used extensively throughout the MicroVAX computer family.

The asynchronous Q-Bus is inexpensive and simple to use. To save signal lines, address and data are multiplexed. The original 16 address lines have been increased to 22 bits to provide a 4-Mbyte address space. The data bus remains only 16 bits wide.

The Q-Bus specifies that DEC (or equivalent) line drivers be used on any bus interface. However, some designers have found ways to hang logic directly on the bus with no ill effects. Some years ago, DEC added block-mode DMA to speed data transfer. An on-board memory controller enables add-in vendors to offer a wide variety of memory types that can easily be used with the Q-Bus.

MicroVAX II

Another successful DEC bus is the MicroVAX II, which uses the Q-Bus for I/O but has a separate memory bus to reduce bus traffic and speed memory access. DMA is handled over the Q-Bus to the memory controller, which is on the processor card. The data is then transferred to the memory via the memory bus. Unlike the early MicroVAX I, the memory bus has a full 32-bit data path.

The memory bus is synchronous with a fixed 400-ns cycle time, which means that all memory boards perform the same; faster memories buy nothing. Parity is the standard level of error protection, providing error detection only, no correction. To exceed the 16-Mbyte maximum configuration of the MicroVAX II, some vendors offer RAM-disk expansion on the Q-Bus.

MicroVAX 3000

The MicroVAX 3000 Series adds many improvements: speed, error correction and memory addressability. DEC continues to expand its bus architectures to support the VAX product line.

*Micro Channel and PS/2 are trademarks of IBM Corporation

Futurebus+

The IEEE specification for Futurebus+, the bus for 64- and 128-bit, and even 256-bit, systems, was due for completion in the fourth quarter '89. Using backplane transceiver logic (BTL), designed specifically for driving a backplane bus, Futurebus+ solves the transmission line problems that limit the speed for buses based on TTL. It boasts speeds of 400 to 3200 Mbyte/s, depending on the size of the system, 32 to 256 bits. Futurebus+ uses an asynchronous bus-interface protocol that is completely independent of the processor family or technology used to implement the system. The bus contains extensive monitoring, diagnostic and error-detection facilities.

Futurebus+ also features a distributed arbitration scheme that expedites the building of fault tolerance into a system. It is an open-architecture bus that also includes a broadcast/broadcall facility for interacting back and forth with multiple boards. The bus supports the implementation of a variety of caching methodologies within a single shared-memory multiprocessing system.

There will probably be very few Futurebus-only systems appearing in the marketplace for some time. Most people will experiment by using Futurebus to upgrade VME or Multibus II systems. Bridges or links will give users a variety of I/O, peripheral and low-cost CPU cards for tasks that do not require the full power of Futurebus. Many companies are exploring ways to bridge the gap between their buses and Futurebus.

Opinions vary on the role the VMEbus will play in the future. Most board vendors, for the time being, will probably provide bridges as stepping stones to link VME with Futurebus+. Intel and the Multibus Manufacturer's Group plan to link Multibus II to Futurebus+.

Users must be able to upgrade their installed systems by connecting them via cable to the new high-performance system, thus preserving their investments in both hardware and software.

REFERENCES

Electronic Engineering Times , CMP Publications, August 14 and September 11, 1989.

The Designer's Guide to Add-In Memory , Third Edition 1989, Chapter Five, *The Final Step—An Industry Survey* , Clearpoint Research Corporation, Hopkinton, MA.

CHAPTER 3
Microprocessor Interfaces to the Am29C668 4M Configurable Dynamic Memory Controller/Driver

Introduction 3-2
Am29C668 CDMC to Am29000 Streamlined Instruction Processor Interface 3-3
Am29C668 CDMC to 80C286 Microprocessor Interface 3-37
Am29C668 CDMC to 80386 Microprocessor Interface 3-53
Am29C668 CDMC to 68020 Microprocessor Interface 3-71

INTRODUCTION

This chapter includes application notes describing four different interface designs utilizing the Am29C668 4M Configurable Dynamic Memory Controller/Driver (CDMC) and four well-known microprocessors–the Am29000, the 80C286, the 80386, and the 68020. The CDMC acts as the address controller between the microprocessor and the dynamic memory array, providing control for 4M, 1M, 256K and 64K dynamic RAMs.

Each interface was designed to provide maximum performance at reasonable cost. Each is as general as possible so that the user may tailor his implementation to a specific memory system. Possible changes are discussed with associated system requirements and implications. A block diagram, timing analysis, and logic equations necessary to implement each design are included.

Am29C668 Configurable Dynamic Memory Controller to Am29000 Streamlined Instruction Processor Interface

by Douglas Lee, Applications Specialist

INTRODUCTION

The interface between the Am29C668 4-Mbit Configurable Dynamic Memory Controller (CDMC) and the Am29000 Streamlined Instruction Processor was designed for maximum performance, while using relatively inexpensive DRAMs. This design uses 100-ns fast-page-mode DRAMs, yet achieves single-cycle burst accesses at 20 MHz. It also uses a minimum number of devices to reduce the required board space. This design is as general as possible so that users may tailor their implementations to specific memory systems. A block diagram, timing analyses and logic equations necessary to implement the design are included.

Distinctive Characteristics

- Am29C668 4-Mbit Configurable Dynamic Memory Controller/Driver with Auto Timing
- 20-MHz Am29000 Streamlined Instruction Processor
- 100-ns Fast-Page-Mode 1 Mbit x 1 DRAMs. Also supports 256 Kbit x 4 Fast-Page-Mode DRAMs or 256 Kbit x 1 Fast-Page-Mode DRAMs.
- Single Am29C668 Controls Two Banks of Interleaved Memory.
- 8-Mbyte Dynamic Memory per Am29C668.
- Four-Cycle Initial Access on Read Cycles, Three-Cycle Initial Access on Write Cycle, Single-Cycle Burst Accesses for Read and Write Cycles. Three-Cycle Initial Read Access Within a Page, Two-Cycle Initial Write Access Within a Page.
- Supports Instruction Burst Restart, Two-Cycle Access.
- Supports Byte Writes.
- Supports 4-Gbyte Address Space Each for Instruction Memory, Data Memory and I/O. Separate Instruction and Data Memory, With Instruction Memory Accessible Via the Data Bus to Load Programs.
- Compatible with the Adapt29K™ In-Circuit Diagnostic System by Decoding Option Bits OPT[2:0].

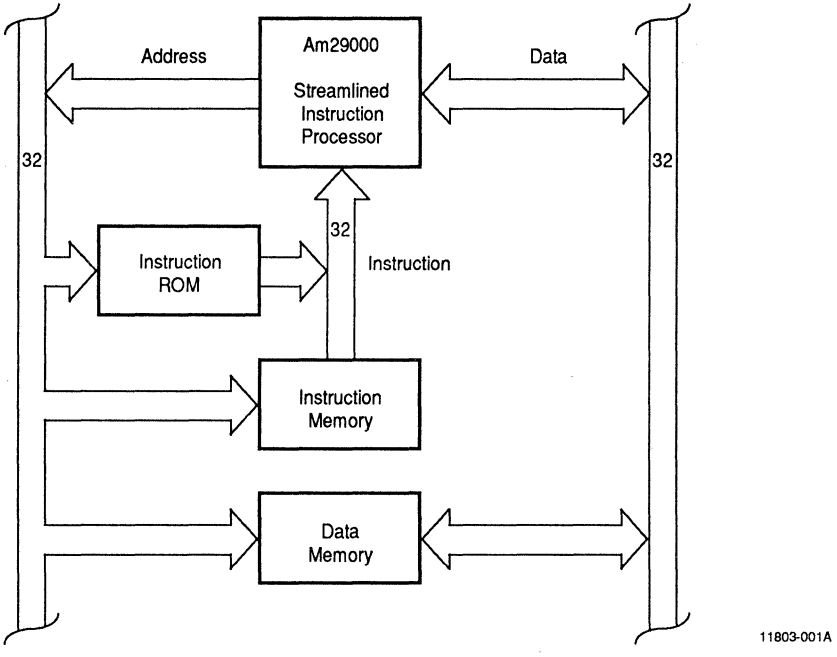


Figure 1. Am29000 System Diagram

Am29000 Overview

When compared with static RAM (SRAM), dynamic RAM can provide far more memory at lower cost and power in the available board space. The main penalty in using DRAM is a loss of speed in the initial memory-access time. Burst-access performance can be maintained by using bank interleaving and fast-page-mode DRAMs. Fortunately, the Am29000 provides features that help compensate for a slower initial access time of system memory.

The Am29000 has an external Harvard architecture with separate data and instruction buses (Figure 1) so that the processor can fetch instructions and data simultaneously. With slower memories, it becomes important to maintain separate instruction and data spaces to increase the probability of instruction and data accesses occurring simultaneouly, while decreasing the probability of a data access preempting an instruction burst. The Am29000 also has burst-mode loads, stores and instruction accesses to provide maximum memory bandwidth.

The Am29000 branch target cache (BTC) stores the first four instructions after a successful branch. The BTC is two-way, set associative, with 16 blocks per set and a block size of four words; there are 512 bytes of storage or 128 words. When a branch is taken, the first four instructions come from the BTC if the branch target address is in the cache. At the same time, the first instruction following those in the cache is accessed. The first three cycles of the initial memory access are hidden by the execution of the instructions in the BTC.

The large register file of the Am29000 in effect provides a data cache for the most frequently used operands. This significantly reduces the number of times that memory needs to be accessed for data as compared with other microprocessors. Also, the Am29000 load and store operations may be overlapped with the execution of other instructions, which again reduces the impact of a slower initial-access-time memory system.

MEMORY ARCHITECTURE OVERVIEW

To obtain good memory throughput while maintaining reasonable cost, 100-ns fast-page-mode 1-MBit DRAMs are used. The Am29000 can accept data every cycle. If additional cycles are needed, the memory controller holds the appropriate ready signal ($\overline{IRDY}$ or $\overline{DRDY}$) inactive. The processor bus interface waits until the ready signal is asserted. For this memory design, the 20-MHz Am29000 completes the initial access to memory in four cycles for Read accesses, three cycles for Write accesses. If the access is to a page that has been previously accessed, a Read access is completed in three cycles and a Write in two cycles. During Write accesses, the ready signal is asserted one cycle earlier than in the Read access because the data is latched from the bus. The memory still requires the same number of cycles to complete the actual memory access, but the system bus is freed to start another access.

There are four major types of speciality-mode DRAMs: fast-page-mode, static-column, nibble-mode and video DRAMs (VRAMs). Nibble-mode DRAMs can access four bits of data in a modulo-4 fashion, but are not applicable to this design because the length of a burst is indeterminate. VRAMs are attractive, since they have an on-chip shift register that permits concurrent data and instruction accesses; however they have cost and availability disadvantages. Static-column DRAMs (SCDRAMs) have a simpler interface since only the column address is changed to access another location in memory. However, this advantage presents a drawback. Since a fast-page-mode DRAM latches the row and column address, the address may change much sooner than an address in an SCDRAM, for which the column address must remain stable until the data is latched externally. This means that the page-mode DRAM can effectively overlap the address propagation delay with the memory access time, thereby giving better performance for comparable-speed DRAMs. In addition, to obtain the 20 MHz throughput using 100-ns SCDRAMs, each bank must be controlled by one Am29C668, thereby increasing control logic, cost and required board space.

Fast-page-mode DRAMs appear to the processor as if they are fast cache memories during accesses within the page. The page size for a 1-Mbit DRAM is 1024 bits or 1 Kbits. The memory discussed here is 32 bits wide and two banks are interleaved; therefore the page size is 8 Kbytes. The Am29C668 detects accesses within the same page via the on-chip cache-mode operation. When a new address is latched, it is compared with the previous row and bank address; if the addresses are the same, $\overline{CH}$ is asserted. The memory state machine immediately begins the next access. An access outside the page, a page miss, causes the memory controller to perform the $\overline{RAS}$ precharge for the DRAMs followed by a normal memory access. Figure 2 shows the timing for an initial access, cache hit and cache miss. The total access time on a page miss requires seven cycles, one for decoding, two cycles for the $\overline{RAS}$ precharge and four for the data access. Shorter memory-access times result when using the cache-mode method than when using normal DRAM accesses. The actual performance of the memory system depends upon the instruction mix of the programs executed.

Each Am29C668 controls a memory array consisting of two banks. Each bank contains 4 Mbytes or 1 Mword (32 bits) of memory. This gives a maximum size of 8 Mbytes

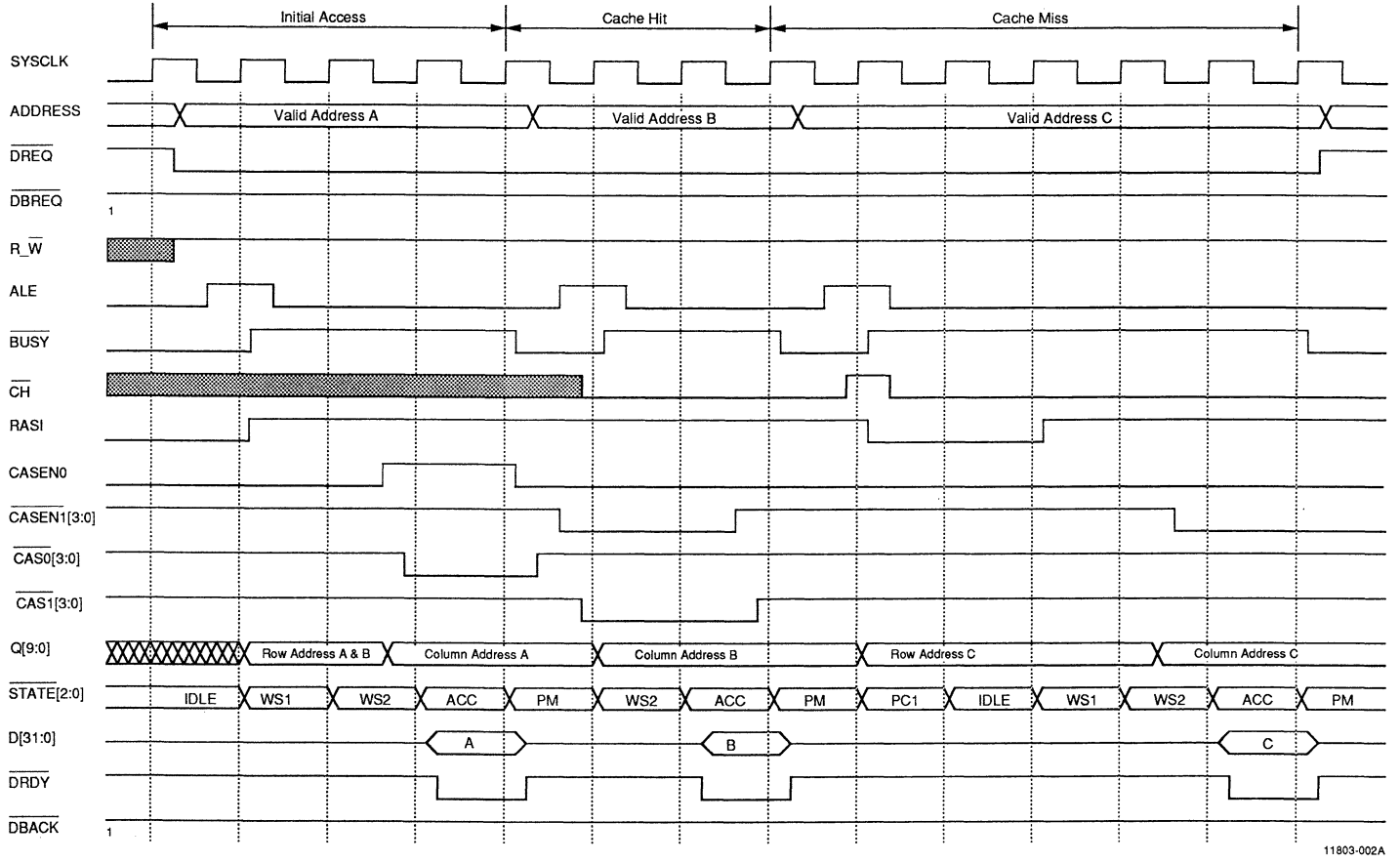


Figure 2. Cache-Mode Timings

or 2 Mwords of memory per Am29C668 controller. For smaller memory systems, 256 Kbits x 4 or 256 Kbits x 1 DRAMs can be used as long as $\overline{CAS}$ access time t_{cac} = 25 ns. This results in a 2-Mbyte memory size or 512 Kwords per Am29C668. With 256 Kbits x 4 DRAMs, four banks of memory can be supported so that the system can be upgraded from 2 Mbytes to 4 Mbytes.

Supported Access Methods

This memory design supports simple and burst-mode accesses of the Am29000. Pipelined accesses are not supported. The memory is divided into instruction and data memory. The instruction memory permits both data Reads and data Writes and is accessible as data so that programs can be loaded into memory via the data bus. This is the most general implementation. Alternative implementations for instruction memory are discussed in the section "Variations on This Design."

Data accesses of instruction memory are given priority over instruction accesses so that a store to instruction memory, while simultaneously executing from instruction memory, will not result in a deadlock. Data memory supports only data accesses.

Figures 3a and 3b show the timing waveforms for an instruction burst access. An instruction burst access is initiated when Instruction Request $\overline{IREQ}$, Instruction Request Type $\overline{IREQT}$, and the address A[31:0] are asserted during the first half of the clock cycle. If an exception occurs, for example a translation lookaside buffer miss or a jump followed by a jump, the Bus Invalid $\overline{BINV}$ is asserted during the second half of the bus cycle. Instruction Burst Request $\overline{IBREQ}$ is also asserted during this part of the cycle. If the request is valid, the memory system accesses the data and asserts Instruction Ready $\overline{IRDY}$ in the cycle when the data is valid. The processor requests burst-mode instruction accesses by asserting $\overline{IBREQ}$. The memory responds to an $\overline{IBREQ}$ request with the Instruction Burst Acknowledge signal $\overline{IBACK}$. In the cycle after $\overline{IBACK}$ is asserted, the Am29000 can start an access to data memory. The memory control logic cannot assert $\overline{IBACK}$ until the address and all the necessary control signals have been latched. During instruction bursts, $\overline{IBREQ}$ may be temporarily deasserted when the prefetch buffer is filled. The memory continues to assert $\overline{IBACK}$, indicating that it can restart the instruction burst. If $\overline{IBREQ}$ is asserted by the processor, the instruction burst starts in the next cycle. If a new request is initiated, the memory deasserts $\overline{IBACK}$.

Figure 4 shows the timing waveforms for a data access. Data accesses are similar to instruction accesses, with a few exceptions. The Am29000 initiates a data access by driving Data Request $\overline{DREQ}$, Data Request Type DREQT[1:0], the address A[31:0] and the Option bits OPT[2:0]. All instruction accesses are 32 bits. Data accesses can be word (32 bits), half word (16 bits) or byte (8 bits). The length of the access is decoded from the OPT[2:0] and the two least significant address bits A[1:0]. The OPT[2:0] bits also distinguish between accesses to memory, instruction ROM (as data), cache control and the ADAPT29K. The Data Request Type outputs DREQT[1:0] distinguish between data memory, input/output, and coprocessor accesses. The Data Ready $\overline{DRDY}$ and Data Burst Acknowledge $\overline{DBACK}$ function the same as the equivalent signals in the instruction control bus. Data Burst Request $\overline{DBREQ}$ functions the same as $\overline{IBREQ}$, except that the Am29000 does not suspend burst-mode data accesses; therefore, this memory design does not support data-burst restart.

Refresh Cycles

To retain data, dynamic memories must be refreshed periodically to restore the charge on the memory-cell storage capacitors. For 1-Mbit DRAMs, all 512 rows of memory must be refreshed every 8 ms. There are three different methods for performing refresh cycles: burst, forced and hidden; each has its advantages and disadvantages. The best method is determined by the instruction mix, system hardware and performance requirements.

The burst-refresh method refreshes all 512 rows sequentially and works especially well in systems with long idle times between memory accesses. The main disadvantage is that an access to memory may be delayed for long periods during the refresh cycles, greatly impacting system response time. This would definitely not be an acceptable method for real-time systems.

The forced or distributed-refresh method periodically inserts refresh cycles. If refreshes are interspersed between memory accesses, the memory-access time is not greatly impacted, since there is a low probability of refresh-request and memory-request contention. One refresh request is generated every 15.6 μs = 8 ms/512 rows. This method is preferable to burst refresh in most systems.

Hidden refresh has the lowest system impact since all or most of the refresh cycle is overlapped with an access to another memory or I/O device. There are times, however, when the system continually accesses the same memory and does not permit hidden refreshes to be performed. If this happens, a forced-refresh cycle must be used. There are conceivable situations where hidden refresh would not perform as well as forced refresh; however, for most general applications, hidden refresh is the best choice.

This design utilizes forced refreshes instead of hidden refreshes, for several reasons. Additional logic is needed to keep track of hidden-refresh cycles. This logic must

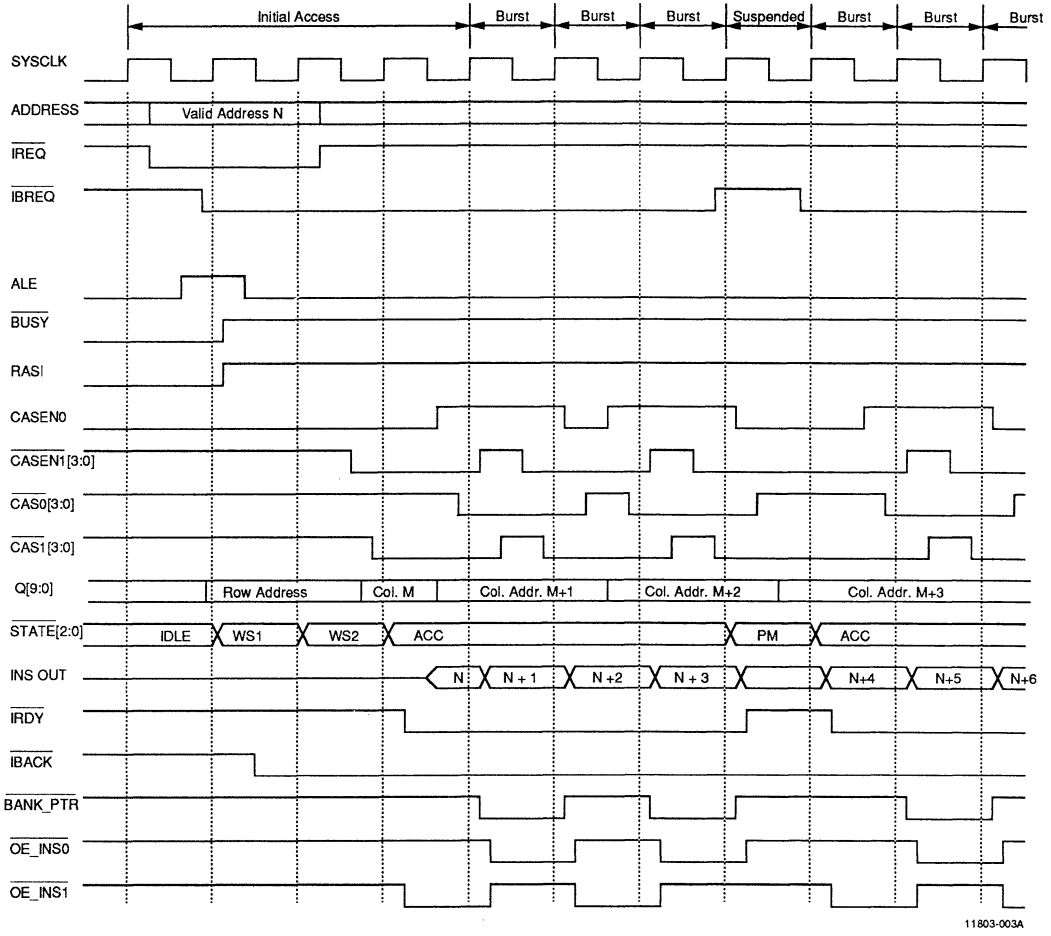


Figure 3a. Instruction Burst Access

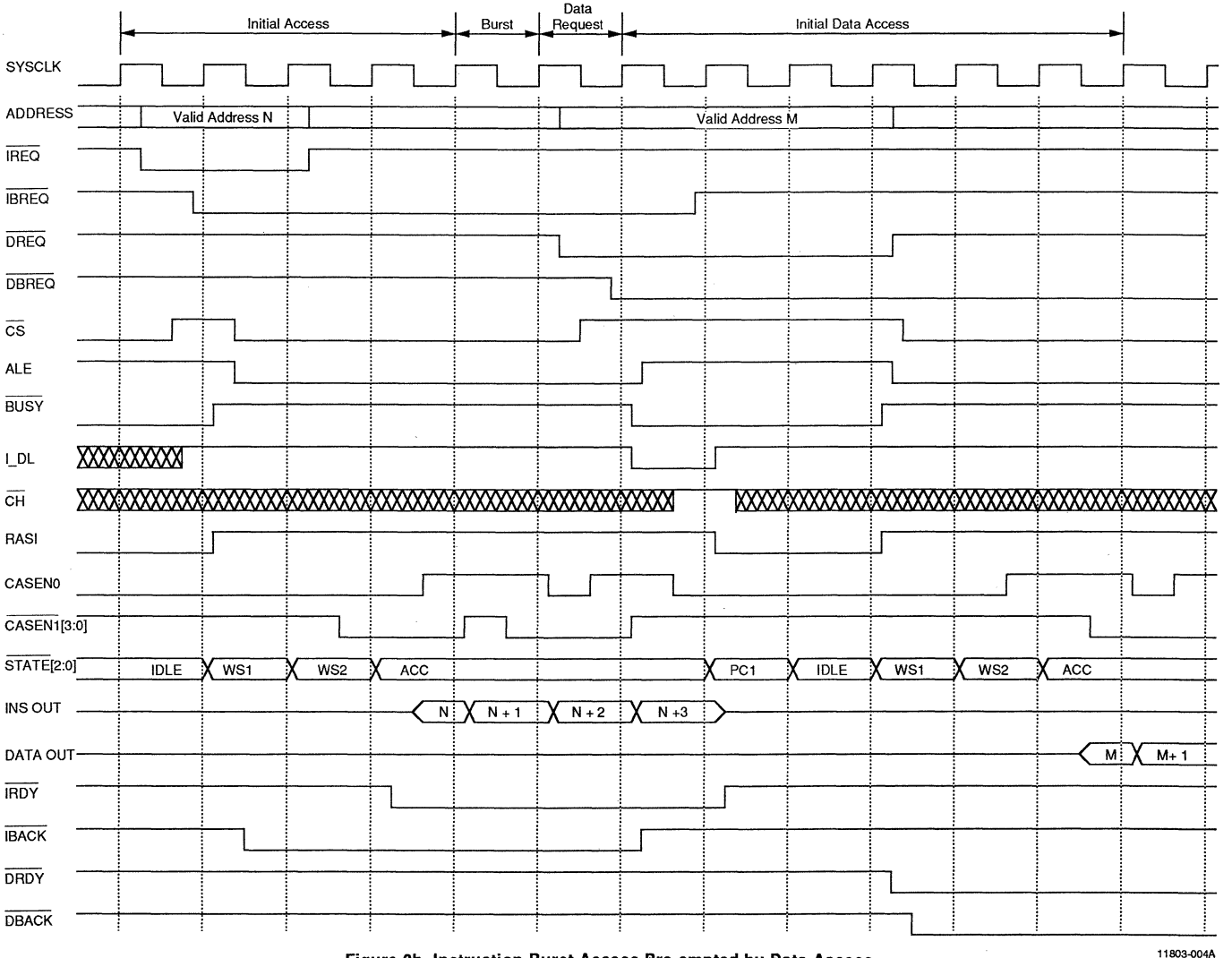


Figure 3b. Instruction Burst Access Pre-empted by Data Access

11803-004A

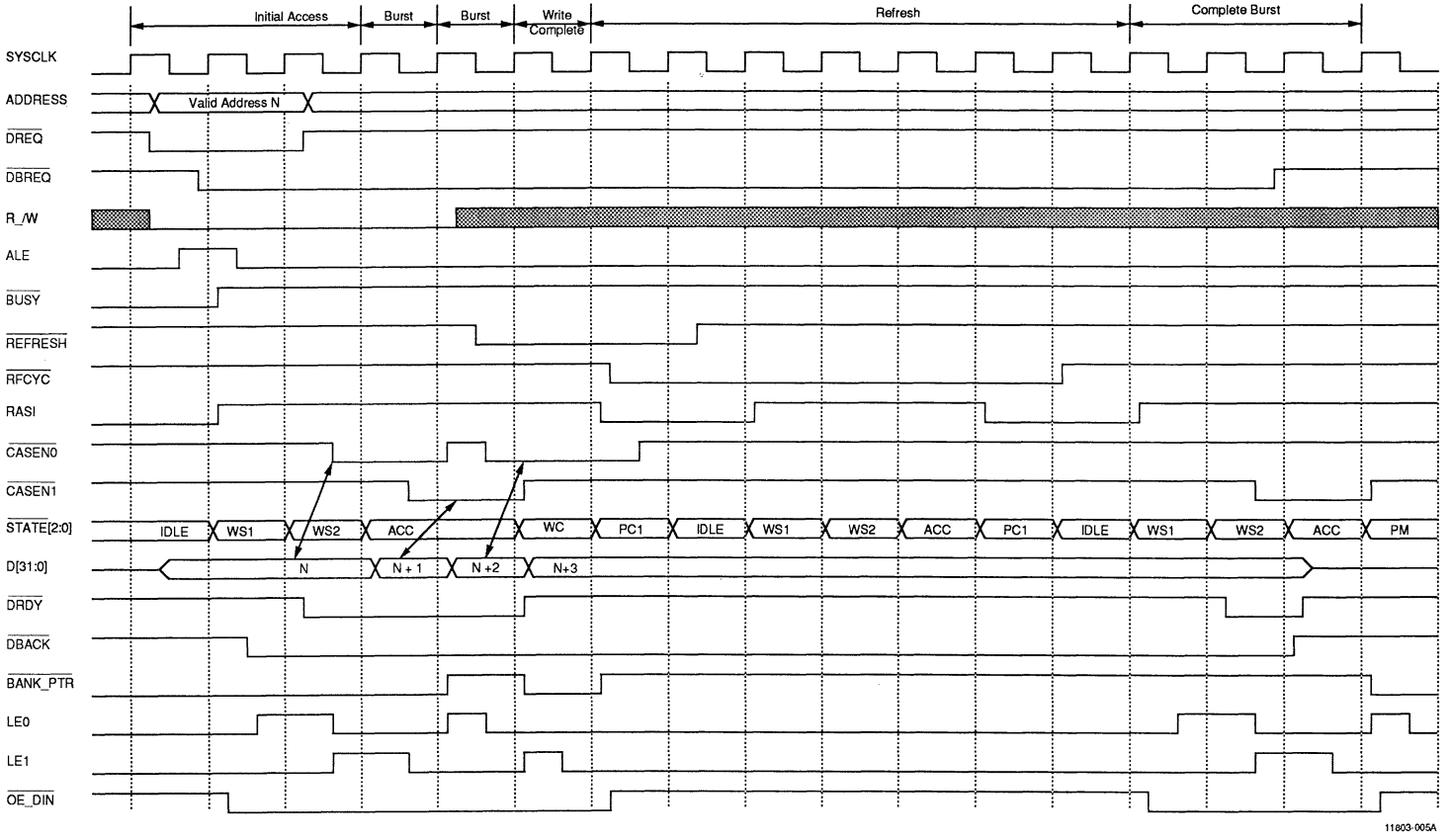


Figure 4. Burst Data Write and Refresh

suppress the forced-refresh request after a hidden-refresh cycle is performed and must force a refresh when no hidden refresh is performed. This adds extra devices and consumes more board space, money and power. Determining when to use hidden refresh is also difficult with the Am29000. For example, if an instruction burst is suspended, a hidden refresh should not occur due to a data access to another part of the system. This is because the instruction burst may have been suspended and could restart on any subsequent cycle. If the hidden refresh starts, the processor must wait a minimum of 11 cycles until the next instruction is read. Since instruction-burst suspension can occur frequently, due to filling the prefetch buffer, this situation can have a major impact on system performance. In these situations, hidden refresh does not guarantee major savings, therefore, the extra effort to implement it is not justified.

A refresh cycle is identical to a normal access, except that the $\overline{CAS}$ outputs to the DRAMs must be suppressed. The Am29C668 suppresses the $\overline{CAS}$ outputs in the refresh mode. $\overline{REFRESH}$ is asserted by the Timer PAL every 9.8 µs. The refresh interval is determined by maximum $\overline{CAS}$ active time (10 ms) as explained in the *Timer PAL* section. If a simple memory access is in progress, the access is completed before the refresh cycle begins. If a burst memory access is in progress at the time a refresh is requested, the burst is suspended. After the refresh access is complete, the burst restarts automatically. If both a memory-access request and refresh request occur during the same cycle, the refresh request is given priority. This is done to meet the refresh requirements of the DRAM.

FUNCTIONAL DESCRIPTION

The main block diagram for this design, including the control logic, buffers and memory array are shown in Figure 5a. Figure 5b is the detailed diagram of the control logic. Figure 5c shows the connections for the Am29C983A Multiple Bus Exchange for one bank of memory.

Am29C668 Configurable Dynamic Memory Controller (CDMC)

The Am29C668 generates the $\overline{RAS}$, $\overline{CAS}$ and address signals to the DRAM array. No external drivers are needed. Additionally, the Am29C668 generates the row addresses during $\overline{RAS}$-only refreshes from the internal-refresh row-address counter.

The Am29C668 must be programmed, via an I/O access, before any memory accesses may occur. The Am29C668 occupys a 256-Kbyte or 64-Kword address space, that can be reduced to only 1 byte by adding an additional decoder in parallel with the Request PAL. The actual decoding is left up to the user since it is system dependent. Address bits A[12:3] contain the value to be loaded into the configuration register. For this design, the lower 13 address bits are 730H. The options selected are: two banks $\overline{RAS}$ and $\overline{CAS}$ configuration, $\overline{CAS}$ byte decoding, $\overline{RAS}$-only refresh, 1-Mbit memory size, cache mode and external timing. The input AC[10] must be tied Low to place the Am29C668 in normal-mode operation, since AC[10] is only used with 4-Mbit DRAMs.

When the Am29C668 is configured to support two banks of memory, the $\overline{RAS}_n$ outputs are divided into two sets: $\overline{RAS}$[1:0] and $\overline{RAS}$[3:2]. Each set normally controls one bank of memory to reduce the capacitive loading each driver sees, thus minimizing the propagation delay. In this application, however, the $\overline{RAS}_n$ outputs of both banks must be Low simultaneously to support interleaved burst-mode accesses. This is generally not possible using other DRAM controllers and would normally require one controller per bank. However,with the Am29C668 in the two-bank configuration, $\overline{RAS}$[0] controls bank 0 and $\overline{RAS}$[1] controls bank 1. Both $\overline{RAS}$[0] and $\overline{RAS}$[1] go Low when RASI is asserted; the SEL[1:0] are not used and both must be tied Low to insure proper operation. In this way, both banks can be active at the same time and still be controlled by one Am29C668. This unique feature saves board space, power and cost. To control which bank is accessed, the $\overline{CAS}_n$ inputs to each bank are strobed separately; this requires an external buffer. Since an external buffer (Am2966) is also used to drive $\overline{WE}$, no additional parts are required.

The Am29C668 supports byte decoding through the $\overline{CASEN}$[3:0] inputs. The Option bits OPT[2:0] and the two least significant address bits A[1:0] are decoded to generate four byte-enable signals $\overline{BE}$[3:0], that are connected directly to the $\overline{CASEN}$[3:0] inputs of the Am29C668. The CASEN0 output of the $\overline{CAS}$-Enable PAL is connected to the Am29C668 CASIEN input. CASIEN controls when the $\overline{CAS}$[3:0] outputs are enabled and the $\overline{BE}$[3:0] outputs control which outputs are active. In addition, the $\overline{BE}$[3:0] outputs are used with the state variables STATE[2:0] to generate $\overline{CASEN1}$[3:0] from the $\overline{CAS}$-Enable PAL. The $\overline{CASEN1}$[3:0] outputs control the $\overline{CAS}$ inputs to bank 1. These outputs are buffered with an Am2966 to control overshoot and undershoot. During byte writes, the bytes that are not selected perform a $\overline{RAS}$-only refresh on the current row address. All Reads are words regardless of the values of OPT[2:0] and A[1:0]. CASEN0 and $\overline{CASEN1}$[3:0] are also used during page-mode accesses. Strobing CASEN0 causes the Am29C668 $\overline{CAS}$[3:0] outputs to pulse and access the DRAMs during a page-mode access (for bank 0). Similarly the $\overline{CASEN1}$[3:0] are pulsed to control the page-mode access for bank 1. The $\overline{CASEN1}$[3:0] outputs from

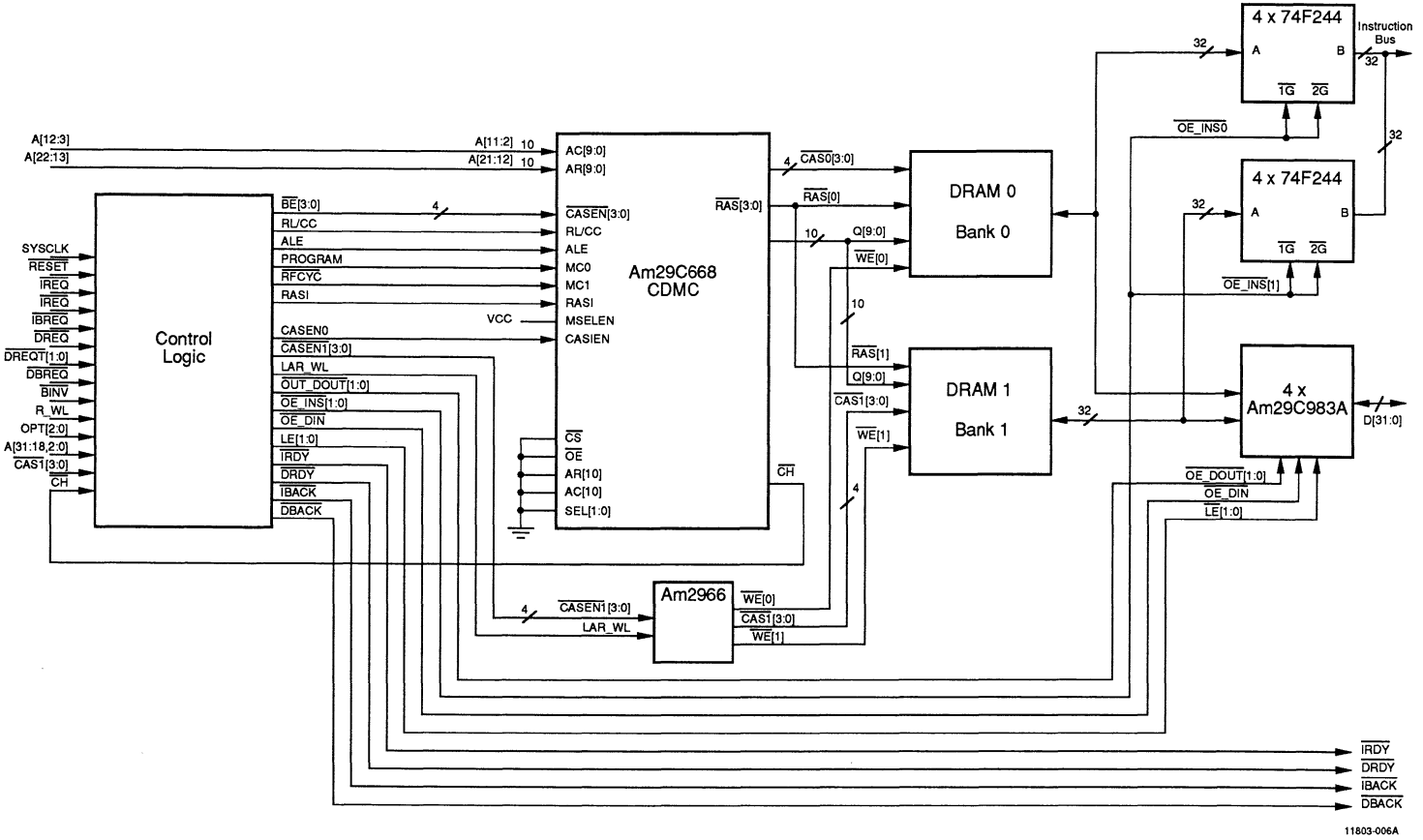


Figure 5a. Top Level Schematic

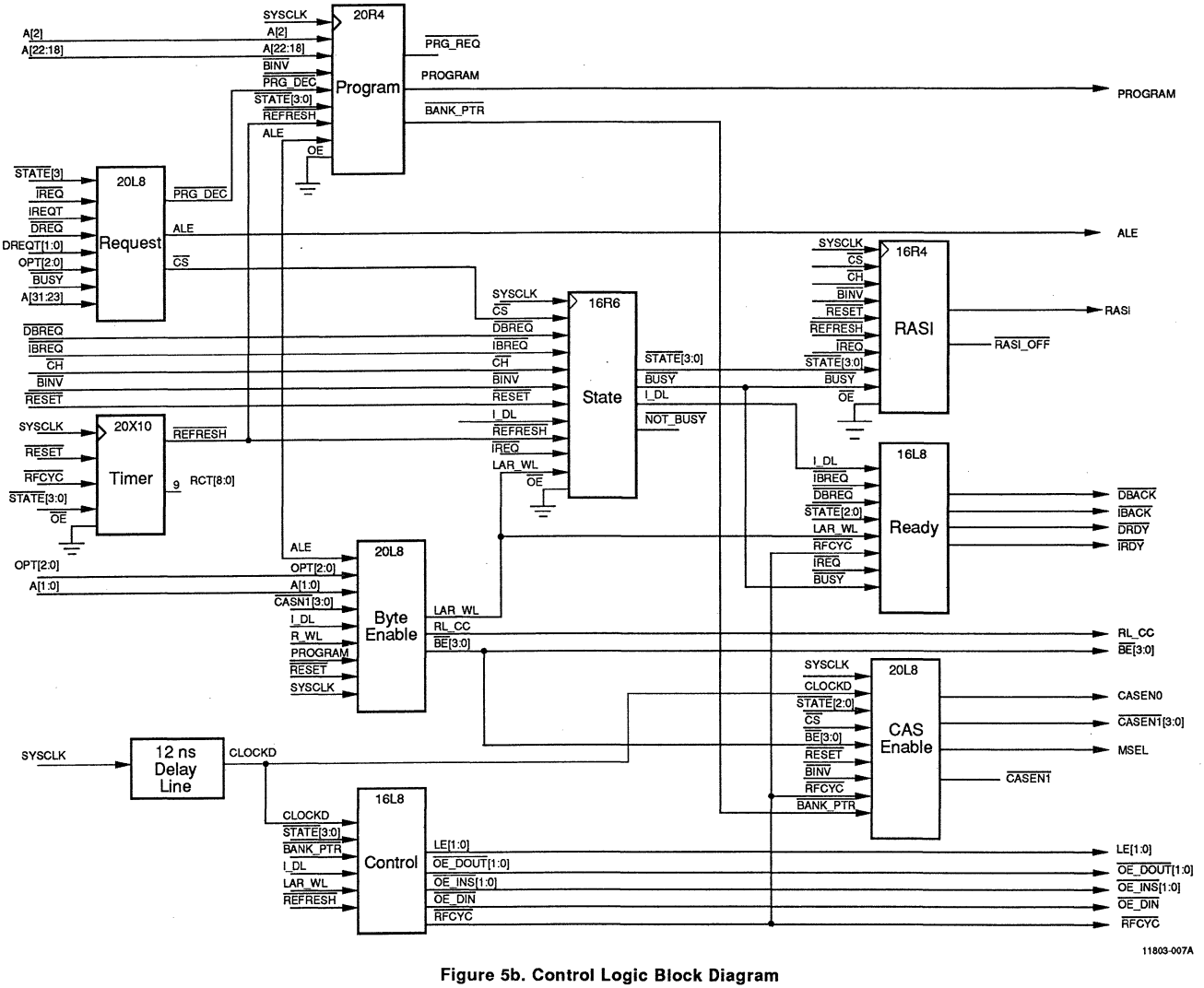


Figure 5b. Control Logic Block Diagram

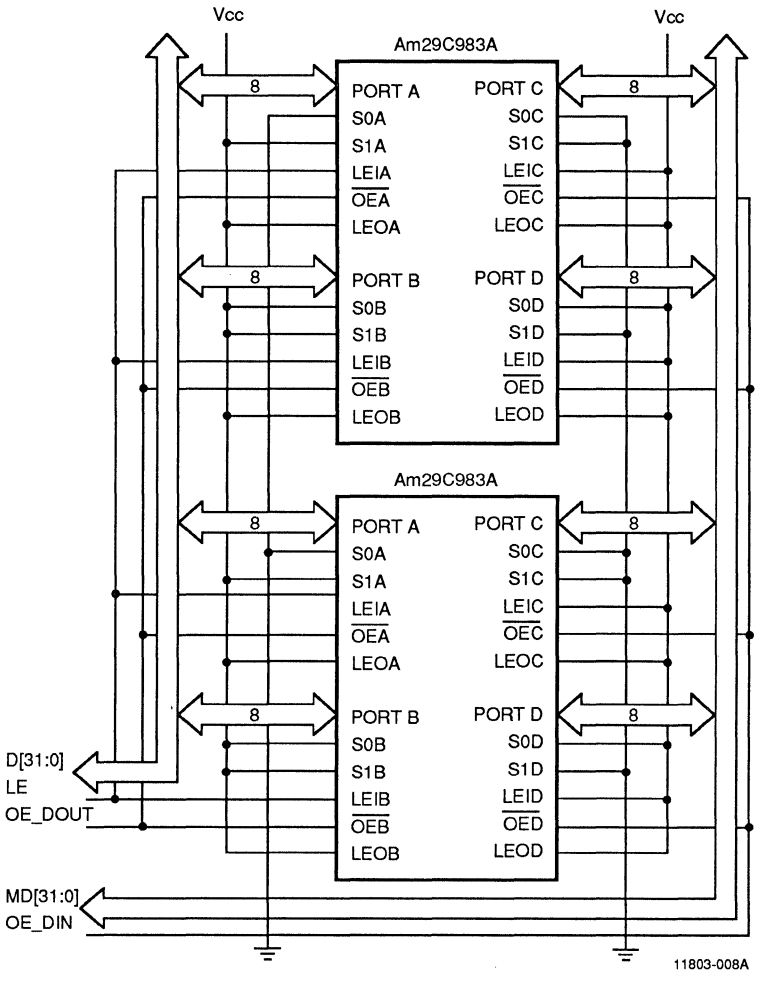


Figure 5c. MBE Connections

the $\overline{CAS}$ Enable PAL are suppressed during refresh, since the external buffer does not automatically perform this function. The Am29C668 suppresses the $\overline{CAS}_n$ outputs in the $\overline{RAS}$-only refresh mode.

The Multiplexer Select input MSEL controls the multiplexing of the addresses to the DRAMs. When MSEL is Low, the row address is selected; when MSEL is High, the column address is selected.

The Address-Latch-Enable signal ALE is generated by the Request PAL. When ALE is High, the address latch of the Am29C668 is transparent. When ALE is Low, the address is latched.

The Register Load/Column Clock RL/CC input of the Am29C668 is a dual-function input. In the initialize mode (MC[1:0] = 11), this input is the register-load signal. On the rising edge of RL/CC, inputs AC[10:0] are stored in the configuration register. In the Read/Write mode (MC[1:0] = 10), RL/CC functions as the column-clock signal to the Am29C668. The High-to-Low edge of RL/CC increments the column counter. ALE must be Low for the counter to increment. If ALE is High, the latch is transparent and the counter does not function properly. The counter function is used during burst-mode accesses. Since both memory banks share the same address bus, the address cannot be incremented until after the $\overline{CAS}_n$ outputs to bank 1 are asserted. The four $\overline{CAS}_n$ outputs to bank 1 are ORed together in the Byte-Enable PAL to generate the RL/CC input to the Am29C668. By using the $\overline{CAS}_n$ outputs, the Column-Address-to-$\overline{CAS}$ hold time is guaranteed.

The Am29C668 has on-chip comparators for implementing cache-mode operation. When properly configured, the Am29C668 compares the previously accessed row and bank address with the current row and bank address. If they are the same, the Cache Hit $\overline{CH}$ output is asserted. The memory-control logic can then access the memory in a fast-page or static-column mode. In this design, cache-mode accesses require three cycles instead of up to six. If the access is to a different row, the control logic must insert a $\overline{RAS}$ precharge and then perform a normal access. This takes six cycles to complete. SEL[1:0] must be tied Low to insure proper comparison of the bank address.

Request PAL

The Request PAL decodes the valid address and control signals for the memory access. It also decodes some of the bus control signals and the upper address bits for an I/O access. The following inputs are used:

$\overline{IREQ}$	Instruction Request, from Am29000
IREQT	Instruction Request Type from Am29000
$\overline{DREQ}$	Data Request, from Am29000
DREQT[1:0]	Data Request Type, distinguishes between memory, I/O and co-processor accesses, from Am29000
OPT[2:0]	Option bits, from Am29000
$\overline{BUSY}$	Memory Busy, asserted when memory is being used, from State PAL
$\overline{STATE}$[3]	State Variable 3, from the State PAL
A[31:23]	Address bits, from Am29000

The following outputs are generated by the Request PAL:

$\overline{PRG_DEC}$	Program Decode, partial decoding of program request
ALE	Address Latch Enable, connected to Am29C668
$\overline{CS}$	Chip Select, Indicates valid memory request signals

The Request PAL's main function is to decode a memory access. The Chip Select $\overline{CS}$ is generated when a valid memory request is generated by the Am29000. The memory occupies the same address space for both data and instruction memory. This is not required and may be changed so that the instruction and data memory occupy different address spaces. For instruction memory accesses, IREQT is decoded. IREQT distinguishes between ROM and RAM accesses. In this implementation, ROM and RAM can occupy the same address space. The 32 address bits are decoded, which provides for a full 4-Gbyte address space.

For data-memory accesses, DREQT[1:0] and OPT[2:0] are decoded to determine the type of memory access (see Tables 2a and 2b). By decoding DREQT[1:0] and OPT[2:0], compatibility with the ADAPT29K is maintained. Also, the memory does not falsely interpret accesses to I/O devices or the coprocessor as memory accesses.The $\overline{CS}$ output does not decode the $\overline{BINV}$ signal. The State PAL uses both $\overline{CS}$ and $\overline{BINV}$ to determine if the memory request is valid.

Table 2a. Decoding of DREQT[1:0]

DREQT[1:0]	Meaning
00	Instruction/Data-Memory Access
01	Input/Output Access
1X	Coprocessor Transfer

Table 2b. Decoding of OPT[2:0] Based on DREQT[1:0]

DREQT[1:0]	OPT[2:0]	Meaning
0X	000	Word Length Access
0X	001	Byte Access
0X	010	Half-Word Access
XX	011	Reserved
00	100	Instruction ROM Access (as data)
00	101	Cache Control
00	110	ADAPT29K Accesses
XX	111	Reserved

The ALE signal from the Request PAL is connected to the Am29C668 ALE input. When ALE is High, the Am29C668 address latch is transparent. The Low-going edge of ALE latches the value in the address latch. In most implementations, ALE is $\overline{CS}$ inverted. However, this is not possible in this case, since ALE must be generated 28 ns, at most, after the rising edge of the SYLCLK. For a 20-MHz Am29000, the control signals require a maximum of 16 ns to be valid leaving only 12 ns to generate ALE. If ALE were $\overline{CS}$ inverted, it would require two PAL delays to generate, or 15 ns minimum; therefore, ALE cannot be $\overline{CS}$ inverted. Directly decoding ALE is not feasible either because of the number of address bits that must be decoded. Thus, in this design, ALE is $\overline{BUSY}$ inverted plus an additional term to provide for an instruction-burst restart. $\overline{BUSY}$ is generated by the State PAL and indicates when the memory is being accessed. The additional term is necessary because the memory system supports instruction-burst restart. At the end of an instruction burst, the memory controller enters the page-mode state and waits for the burst to restart or for a new instruction request. ALE must be held Low in this state to insure that the instruction burst can restart. If a new instruction request is received, ALE must be active in that cycle. Since $\overline{BUSY}$ is still asserted, there must be an additional term that so that ALE can be asserted when $\overline{BUSY}$ is asserted. $\overline{STATE}$[3] indicates that the memory is in the page-mode state. If $\overline{IREQ}$ is asserted during this state, ALE is also asserted so that the new instruction can begin without delay.

Program Decode $\overline{PRG_DEC}$ from the Request PAL generates a partial decoding of a programming request, because of the large number of address bits that must be decoded. Since the configuration register occupies a 256-Kbyte block, A[31:18] must be decoded. This cannot be done simply and efficiently in a single PAL. By using a two-stage decoder, Request and Program PALs, the programming requests are easily decoded.

Program PAL

The Program PAL is used to decode the lower address bits and $\overline{REFRESH}$ signal for an I/O access to load the Am29C668 configuration register. This PAL also maintains the Bank-Pointer signal $\overline{BANK_PTR}$ that determines which bank is currently active. The following inputs are used:

SYSCLK	20-MHz System Clock
$\overline{STATE}$[3:0]	State Variables, from State PAL
A[2]	Address Bit 2, from Am29000
A[22:18]	Address, from Am29000
$\overline{BINV}$	Bus Invalid, from Am29000
ALE	Address Latch Enable, from Request PAL
$\overline{PRG_DEC}$	Program Decode, from Request PAL
$\overline{REFRESH}$	Refresh Request, from Timer PAL

The following outputs are generated:

$\overline{PRG_REQ}$	Program Request, Signals Valid Decoding of Program Request
PROGRAM	Program, Connects to Am29C668 MC0 Input
$\overline{BANK_PTR}$	Bank Pointer, Indicates Currently Active Bank

The $\overline{PRG_REQ}$ signal is a combinatorial output and is asserted when A[22:18] are the same as the addresses of the configuration register. Currently the Am29C668 occupies 256 Kbyte of I/O address space. This can be reduced to only one byte by adding an address decoder in parallel with the Request PAL. This decoder would decode the lower 18 address bits A[17:0]. By using this parallel decoding scheme to detect valid I/O requests, the full 32-bit address is decoded and all of the 4-Gbyte I/O address space can be used.

PROGRAM is a registered output, asserted if $\overline{PRG_REQ}$ and $\overline{PRG_DEC}$ are valid and $\overline{BINV}$ and $\overline{REFRESH}$ are deasserted. It is connected to the Am29C668 MC0 input. When MC1 and MC0 are both one, the Am29C668 configuration register is loaded on the rising edge of the register load RL/CC signal, generated by the Byte-Enable PAL. Programming requires two clocks as shown in Figure 6. The programming logic does not distinguish between I/O Reads and Writes and configures the Am29C668 CDMC on either valid access. $\overline{REFRESH}$ must be deasserted so that the State PAL does not assert $\overline{RFCYC}$, the MC1 input to the Am29C668.

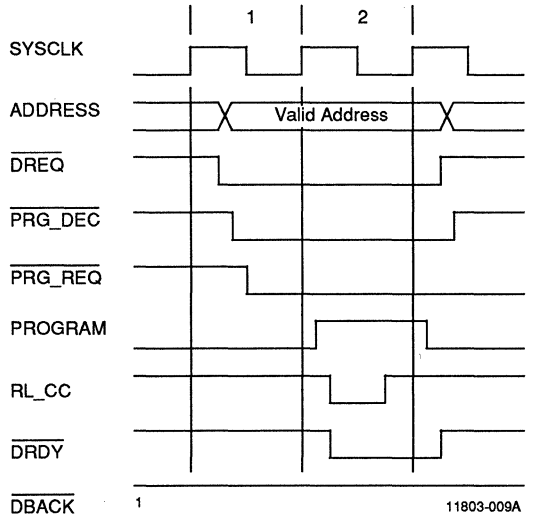


Figure 6. Program Cycle

The bank-pointer signal $\overline{\text{BANK_PTR}}$ indicates the currently active bank, which is initially determined by the value of A[2] during the initial access. If A[2] is 0, bank 0 is active; if A[2] is1, bank 1is active. A[2] is used because consecutive words must be in different banks for burst accesses to occur with no wait states. $\overline{\text{BANK_PTR}}$ is toggled in the access state.

Byte-Enable PAL

This PAL generates the $\overline{\text{BE}}$ and latched-Read/Write signals used during data-memory accesses. It generates the register RL/CC signal used during programming and burst-mode accesses. The following inputs are used:

ALE	Address Latch Enable, from Request PAL
OPT[2:0]	Option, from Am29000
A[1:0]	Least Significant Address Bits 1 and 0, from Am29000
$\overline{\text{CAS1}}$[3:0]	$\overline{\text{CAS}}$ Outputs to Bank 1, from CAS-Enable PAL
I_DL	Instruction Active High, Data Active Low, from Request PAL
R_WL	Read Active High, Write Active Low, from Am29000
PROGRAM	Program, from Program PAL
$\overline{\text{RESET}}$	System Reset Signal
SYSCLK	20-MHz System Clock
RL/CC	Register Load/Column Clock
LAR_WL	Latched Read Active High, Write Active Low
$\overline{\text{BE}}$[3:0]	Byte Enable

The RL/CC signal is a dual-function output. During programming cycles, it loads the Am29C668 configuration register on its rising edge. During memory accesses, it increments the Am29C668 column counter on its falling edge. This function is used during burst accesses to increment the column address.

The Byte-Enable PAL latches the Read/Write R_WL signal generated by the Am29000. The latched signal is LAR_WL. When ALE is High, the latch is transparent; and when ALE is Low, the value is latched. R_WL is undefined during instruction accesses. LAR_WL is forced High during instruction accesses because all instruction accesses are Read cycles.

The byte-enable outputs $\overline{\text{BE}}$[3:0] are generated by decoding the option bits OPT[2:0] and the two least significant address bits A[1:0]. See Table 3. The Am29000 supports both Big Endian and Little Endian addressing. Big Endian numbers the bytes and half words from the most significant bit to the least significant bit, while Little Endian numbers from the least significant to the most significant. This design assumes Big Endian, as does all current 29K software. If a Little Endian implementation is required, the variable BYTEORDER must be changed from 0 to 1 in the PAL equations.

Table 3. Byte-Enable Decoding.

OPT[2:0]	A[1:0]	$\overline{\text{BE}}$[3:0]	Access Type
000	XX	0000	Word
001	00	0111	Byte 0
001	01	1011	Byte 1
001	10	1101	Byte 2
001	11	1110	Byte 3
010	00	0011	Half Word 0
010	10	1100	Half Word 1

Note: Big Endian assumed.

CAS-Enable PAL

This PAL generates the CAS-Enable signals that control the $\overline{\text{CAS}}$ outputs to the memory. The following inputs are used:

SYSCLK	20-MHz SystemClock
CLOCKD	Delayed SYSCLK
$\overline{\text{STATE}}$[2:0]	State Variables, from State PAL Indicates Current Memory Cycle

$\overline{BE}$[3:0]	Byte Enable, from the Byte-Enable PAL
$\overline{CS}$	Chip Select, from Request PAL
$\overline{RFCYC}$	Refresh Cycle, from State PAL
$\overline{BANK_PTR}$	Bank Pointer, from Control PAL Indicates Which Bank is Currently Active
$\overline{RESET}$	System-Reset Signal
$\overline{BINV}$	Bus Invalid, from Am29000

The following outputs are generated:

CASEN0	CAS Enable for Bank 0
$\overline{CASEN1}$[3:0]	CAS Enable for Bank 1
$\overline{CASEN1}$	Used to Determine Which Bank has Completed its Access.
MSEL	Multiplexer Select, Connected to the Am29C668 MSEL Input.

CASEN0 is the input to the Am29C668 CASIEN. When CASEN0 is asserted, the $\overline{CAS}$[3:0] outputs of the Am29C668 are enabled. Since the auto-timing is used, the CASIEN input is an external override for controlling the $\overline{CAS}$ outputs.

The $\overline{CASEN1}$[3:0] outputs are the inputs to theAm2966 external buffer and control the $\overline{CAS}$ inputs of bank 1. Each output controls one byte. During refresh cycles, the $\overline{CASEN1}$[3:0] outputs are always deasserted. The Am29C668 deasserts the $\overline{CAS}$ outputs during refresh cycles, but the external buffer's $\overline{CAS}$ outputs follow the inputs. Therefore all the $\overline{CASEN1}$[3:0] outputs are suppressed.

$\overline{CASEN1}$ is used to determine if CASEN0 or $\overline{CASEN1}$[3:0] is deasserted on the rising edge of SYSCLK. This assures maximum $\overline{CAS}$ precharge time since the CASEN0 and $\overline{CASEN1}$[3:0] outputs turn off on the rising edge of SYSCLK. If this were not done, some of the $\overline{CAS}$ precharge time would be lost waiting for $\overline{BANK_PTR}$ and $\overline{STATE}$[3:0].

MSEL controls address multiplexing to the DRAMs. When MSEL is Low, the row address is selected, when MSEL is High, the column address is selected.

Control PAL

The Control PAL generates the signals to enable and disable the output drivers and control the data latches. The following inputs are used:

SYSCLK	20-MHz System Clock
$\overline{STATE}$[3:0]	State Variables, from State PAL
I_DL	Instruction or Data Cycle Indicator, from Request PAL
LAR_WL	Latched Read or Write Cycle Indicator, from Byte-Enable PAL
$\overline{REFRESH}$	Refresh Request, from Timer PAL
$\overline{BANK_PTR}$	Bank Pointer, from Program PAL

The following outputs are generated:

$\overline{OE_INS}$[1:0]	Output Enable for Instruction-Bus Buffers
$\overline{OE_DOUT}$[1:0]	Output Enable for Data-Bus Buffers
$\overline{OE_DIN}$	Output Enable Data Input
LE[1:0]	Latch Enables for Data-Bus Latches
$\overline{RFCYC}$	Refresh Cycle, Connected to Am29C668 MC1 Input and Used to Indicate That a Refresh Cycle is in Progress

The memory connects to two separate data and instruction buses. While the instruction bus is Read only, the data bus is bidirectional and four Am29C983A Multiple Bus Exchangers (MBE) are used as interface. By using the input latches in the MBE, single-cycle Write cycles can be performed. The data is removed from the bus one cycle before it is written to memory; thus two Write cycles can be overlapped. Since the instruction bus is Read-only, simple 74F244 buffers are used for interfacing.

The $\overline{OE_INS}$[1:0] outputs enable the output drivers of the instruction bus. The instruction buffers are only turned on when the memory is in the access state and the bank is active as indicated by $\overline{BANK_PTR}$. $\overline{RFCYC}$ must also be deasserted to insure that the buffers do not drive the bus during refresh cycles. The buffers do not turn on until at least two cycles after the access is initiated, providing ample time for the previous device controlling the bus to get off.

The $\overline{OE_DOUT}$[1:0] outputs perform the same function as the $\overline{OE_INS}$[1:0] outputs. The data drivers are only turned on when the memory is in the access state, the bank is active as indicated by $\overline{BANK_PTR}$ and $\overline{RFCYC}$ is deasserted. Similarly, the data buffers do not turn on until at least two cycles after the access is initiated.

The $\overline{OE_DIN}$ input controls the outputs driving the data to both banks of DRAMs. The drivers are turned on as early as possible to insure the data set-up time is guaranteed.

The LE[1:0] outputs control the input data latches. The data from the Am29000 is valid 20 ns after the rising edge of the system clock and is held for 4 ns after the next rising edge of the system clock. The LE outputs of this PAL

cannot be generated synchronously from SYSCLK because, in the worst case, LE would be deasserted 4 ns after data is invalid, violating the Am29C983A setup and hold times. If a synchronous design were used, a faster clock signal would be needed. Since the system clock is already 20 MHz, the logical choice is a 40-MHz clock signal, twice the SYSCLK frequency. This design would be difficult because the skew between the two clock signals must be carefully controlled. A simpler and therefore better design is to use a delay line to generate a delayed clock signal CLOCKD, so that LE[1:0] are combinatorial outputs that are valid only while CLOCKD is High. SYSCLK is delayed long enough to insure that $\overline{\text{BANK_PTR}}$ is valid and the LE[1:0] outputs are not falsely asserted.

The refresh cycle output $\overline{\text{RFCYC}}$, which is only asserted during refresh cycles, is connected to the Am29C668 MC1 input. When $\overline{\text{RFCYC}}$ is asserted, the Am29C668 drives the current refresh row address on its Q[10:0] outputs. The internal counter is updated at the end of each refresh cycle to insure that all rows are refreshed. $\overline{\text{RFCYC}}$ is asserted during the IDLE cycle to guarantee the MC1-to-RASI set-up time. $\overline{\text{RFCYC}}$ is deasserted during the PC1 cycle to meet the MC1-to-RASI hold time.

Ready PAL

This PAL generates the ready and burst acknowledgement signals for both instruction and data accesses. The following inputs are used:

I_DL	Instruction or Data Indicator, from Request PAL
$\overline{\text{IREQ}}$	Instruction Request from Am29000
$\overline{\text{IBREQ}}$	Instruction Burst Request, from Am29000
$\overline{\text{DBREQ}}$	Data Burst Request, from Am29000
$\overline{\text{STATE}}$[2:0]	State Variables, from State PAL
$\overline{\text{CS}}$	Chip Select, from Request PAL
LAR_WL	Latched Read/Write Signal, from Byte Enable PAL
$\overline{\text{RFCYC}}$	Refresh Cycle Indicator, from State PAL
$\overline{\text{BUSY}}$	Busy from State PAL, Indicates When Memory is Accessed

The following outputs are generated:

$\overline{\text{IBACK}}$	Instruction Burst Acknowledge
$\overline{\text{DBACK}}$	Data Burst Acknowledge
$\overline{\text{IRDY}}$	Instruction Ready, Indicates Valid Instruction on the Instruction bus
$\overline{\text{DRDY}}$	Data Ready, Indicates Valid Data on the Data Bus

The $\overline{\text{DRDY}}$ and $\overline{\text{DBACK}}$ outputs are used to inform the processor about the status of data-memory accesses. $\overline{\text{DRDY}}$ is asserted at the completion of data Read or Write cycles. $\overline{\text{DBACK}}$ is asserted when the memory supports burst-mode accesses. When $\overline{\text{DBACK}}$ is asserted, the Am29000 is freed to start an instruction access if one is pending. Therefore, all the address and control lines needed by the memory in subsequent cycles must be latched. This is insured by not asserting $\overline{\text{DBACK}}$ until ALE is deasserted. $\overline{\text{DBACK}}$ is not deasserted as the result of a refresh request. If a burst access is currently in progress, it is completed. The burst is delayed by not asserting $\overline{\text{DRDY}}$. The refresh cycle is performed and then the burst is resumed. If this were not done, a refresh request would terminate the burst access. The processor would then have to try to restart the burst access, causing more contention for the address bus. $\overline{\text{DRDY}}$ is also asserted during any cycle when PROGRAM is active.

The $\overline{\text{IRDY}}$ and $\overline{\text{IBACK}}$ outputs function the same as $\overline{\text{DRDY}}$ and $\overline{\text{DBACK}}$ except they are used for instruction cycles rather than data cycles. Because the memory supports instruction burst re-start, $\overline{\text{IBACK}}$ must be asserted as long as the memory can restart the suspended burst. $\overline{\text{IBACK}}$ is latched until a new cycle is started when the Am29000 asserts $\overline{\text{IREQ}}$. $\overline{\text{IBACK}}$ is deasserted as soon as $\overline{\text{IREQ}}$ is asserted, preventing $\overline{\text{IBACK}}$ from being falsely asserted.

To support multiple devices, the same control signals from different devices are externally OR-ed together to obtain the appropriate control signals to the Am29000. It is difficult to implement the transfer of control by selectively driving the control lines with three-state buffers as is commonly done in slower memory systems. Wire OR-ing with open-collector drivers is similarly impractical. Using an SSI gate or PAL is the only practical method to support multiple devices on the same bus.

Timer PAL

The Timer PAL generates the refresh requests. The following inputs are used:

SYSCLK	20-MHz System Clock
$\overline{\text{RESET}}$	System Reset; Initializes Counter
$\overline{\text{RFCYC}}$	Refresh Request; Signals Refresh-Memory Access
$\overline{\text{STATE}}$[3:0]	State Variables, from State PAL

The following outputs are used:

RCT[8:0]	Counter
$\overline{\text{REFRESH}}$	Forced Refresh Request

The DRAMs have a maximum $\overline{\text{CAS}}$ active time of 10 ms. If $\overline{\text{CAS}}$ is active longer than 10 ms, the memory could be corrupted; therefore, the refresh interval should be less than 10 ms to ensure that the maximum $\overline{\text{CAS}}$ active time is not violated. If the system can always guarantee that $\overline{\text{CAS}}$ is not active longer than 10 ms, the refresh interval can be extended to 15.6 µs. This reduces the refresh overhead from 3.5 to 2.2%.

The Timer PAL helps implement the forced refreshes along with the State PAL. The period for the Timer PAL is selected by the value initialized in the counter. This value is set to 195 resulting in a refresh request cycle time of 9.8 µs = 196 x 50 ns (an extra cycle is included since the counter decrements to 0 before resetting). The initial value of the count is determined by the location of INIT in the logic equations (see the PAL Equations section). $\overline{\text{REFRESH}}$ is asserted when the counter decrements to zero and until the memory finishes the memory refresh. This is indicated by $\overline{\text{RFCYC}}$ asserted and the memory in the access state.

There are several alternate methods to implement the refresh timer. A 555 timer could be used and would require less board space and cost less; however, this solution requires asynchronous arbitration. Another alternative is to use a spare DMA channel to implement the refresh requests similar to the PC-AT* and PS/2* systems. This is practical only if there are spare DMA channels available.

RASI PAL

This PAL generates the Row Address Strobe Input RASI for the Am29C668, that causes the appropriate $\overline{\text{RAS}}_n$ output to be asserted and start the internal timing chain. The following inputs are used:

SYSCLK	20-MHz System Clock
$\overline{\text{STATE}}$[3:0]	State Variables, from State PAL
$\overline{\text{BUSY}}$	Busy, from State PAL
$\overline{\text{REFRESH}}$	Refresh Request, from Timer PAL
$\overline{\text{BINV}}$	Bus Invalid, from Am29000
$\overline{\text{RESET}}$	System Reset Signal
$\overline{\text{CS}}$	Chip Select, from Request PAL
$\overline{\text{CH}}$	Cache Hit Signal, from Am29C668
$\overline{\text{IREQ}}$	Instruction Request, from Am29000

The following outputs are generated:

$\overline{\text{RASI_OFF}}$	Indicates Certain Conditions where RASI Must be Deasserted
RASI	Row Address Strobe Input, Connected to Am29C668 RASI input.

The RASI output is connected to the Am29C668 RASI input and is used to control the $\overline{\text{RAS}}_n$ outputs and to start all memory accesses, Read/Write and refresh cycles. Once RASI is asserted, it remains asserted until $\overline{\text{RASI_OFF}}$ is asserted. $\overline{\text{RAS_OFF}}$ is required because there were not enough product terms to implement RASI directly. Keeping RASI asserted provides for fast-page-mode accesses. The $\overline{\text{RAS}}$[0] output is the only Am29C668 $\overline{\text{RAS}}$ output used in this design; it is generated by inverting RASI.

State PAL

The State PAL is responsible for arbitrating between memory accesses. It also implements the memory state machine. The following inputs are used:

SYSCLK	20-MHz System Clock
$\overline{\text{CS}}$	Chip Select, from Request PAL
$\overline{\text{DBREQ}}$	Data Burst Request, from Am29000
$\overline{\text{IBREQ}}$	Instruction Burst Request, from Am29000
$\overline{\text{IREQ}}$	Instruction Request, from Am29000
$\overline{\text{BINV}}$	Bus Invalid, from Am29000
LAR_WL	Latched Read or Write Signals, from Byte Enable PAL
$\overline{\text{CH}}$	Cache Hit, from Am29C668
$\overline{\text{RESET}}$	System Reset
$\overline{\text{REFRESH}}$	Refresh, from Timer PAL

The following outputs are generated:

$\overline{\text{STATE}}$[3:0]	State Variables, Indicate Current Memory State
$\overline{\text{BUSY}}$	Busy, Asserted When Memory is Being Accessed
$\overline{\text{NOT_BUSY}}$	Not Busy, Used to Indicate When $\overline{\text{BUSY}}$ Should be Deasserted
I_DL	Instruction Cycle Active High, Data Cycle Active Low

*PC-AT and PS/2 are trademarks of IBM Corporation.

The I_DL output is used to indicate the type of memory access currently being performed. When I_DL is High, an instruction cycle is being performed; when it is Low, a data cycle is performed. I_DL is also Low when an instruction access is initiated outside of this modules address space so that the control logic will not mistakenly restart an instruction burst.

This PAL controls the memory. Figure 7 shows the memory state diagram. There are seven memory states:

IDLE	Idle State or Arbitration
WS1	Wait State One
WS2	Wait State Two
ACC	Memory Access
WC	Write Complete
PC1	Pre-charge State One
PM	Page Mode

The state machine uses four state variables $\overline{\text{STATE}}$[3:0]. The lower three state variables $\overline{\text{STATE}}$[2:0] can be used by the other PALs to determine the present state of the memory. This is possible if the PAL does not need to distinguish between the IDLE and PM states as they have the same lower three bits. This is possible in the Ready PAL for instance.

The memory always starts in the IDLE state (Figure 7). On RESET, the state machine goes to IDLE and remains there until a refresh request or memory-access request. When a valid memory request is generated by the Am29000, the state machine goes from the IDLE state to WS1. In this state, RASI is asserted to start the internal timing chain in the Am29C668. From WS1, the state machine goes unconditionally to WS2. In the second half of WS2, the appropriate $\overline{\text{CASEN1}}$[3:0] or CASEN0 signals are asserted. This guarantees the $\overline{\text{RAS}}$-to-$\overline{\text{CAS}}$ timing. The memory finally goes to the ACC state, completing the initial memory access. If a burst access is requested and there are no pending memory accesses or refresh accesses, the memory remains in the ACC state. When the burst is completed, the state machine unconditionally goes to the PM state. In this state, RASI is held High and the $\overline{\text{CASEN1}}$[3:0], CASEN0 and BANK_PTR outputs are not changed. Instruction bursts can be restarted in this state in the cycle after $\overline{\text{IBREQ}}$ is asserted. Other accesses may be initiated in only three cycles if they are to the currently active page.

A refresh cycle requested by the timer PAL asserting $\overline{\text{REFRESH}}$ is given priority over memory accesses. If a refresh request is received while the memory is accessed, the refresh request waits until the current memory access is completed. The memory state machine precharges $\overline{\text{RAS}}$ by going to states PC1 and IDLE. In both these states, $\overline{\text{RASI}}$ is held Low, thereby precharging $\overline{\text{RAS}}$. The $\overline{\text{RAS}}$-only refresh cycle follows. The refresh cycle is identical to a memory access, except that $\overline{\text{CAS}}$ is suppressed to the DRAMs. Finally another precharge $\overline{\text{RAS}}$ cycle occurs. If there are no outstanding memory requests, the state machine remains in the IDLE state. If the refresh request occurs during a burst access, the burst access is suspended by the memory-control logic and restarted after the refresh access is complete.

To prevent any possible deadlock, an instruction burst is the only burst access that may be interrupted by another memory request. If a data-memory request is generated while an instruction burst is in progress, the state machine deasserts $\overline{\text{BUSY}}$, thereby asserting the Address Latch Enable ALE input. This adds one extra cycle to the data access, which is not critical since this situation only occurs when loading a program into the same memory that the Am29000 is using for execution. This occurs infrequently.

A $\overline{\text{BUSY}}$ signal indicates to the control logic that the memory is currently being accessed. It is asserted during data-memory accesses and is not deasserted until the memory access/burst is complete. $\overline{\text{BUSY}}$ is also asserted during instruction accesses, but may be deasserted if a data request is pending. $\overline{\text{BUSY}}$ is always asserted until the state machine reaches the ACC state; the current access must always be completed. The output $\overline{\text{NOT_BUSY}}$ is used to indicate when $\overline{\text{BUSY}}$ must be deasserted, because it was not possible to implement $\overline{\text{BUSY}}$ with the available product terms in a single output.

Data Buffers

Am29C983A Multiple Bus Exchanges for the data bus are used in this design to minimize the number of devices used. Since early Write cycles ($\overline{\text{WE}}$ asserted before $\overline{\text{CAS}}$) are always performed, the output driver of the DRAM is always off when data is written to memory; therefore, the input and output pins of the DRAMs can be tied together to reduce the number of traces. Also, transceivers can now be used; otherwise, separate data input and output buffers would be needed. Because only the input latches are used, all the latch-enable outputs are tied High making these output latches always transparent. The MBEs also provide for byte swapping and for support of byte parity via a 9-bit data path. Byte swapping is useful in interfacing to 16- or 8-bit devices or for translating from Big to Little Endian. Parity can be used to enhance memory reliability by detecting single-bit errors. Neither of these features are used in this design.

The instruction bus is separate from the data bus, therefore it requires its own buffer. Since the instruction bus is a Read-only bus, 74F244 buffers can be used. By using

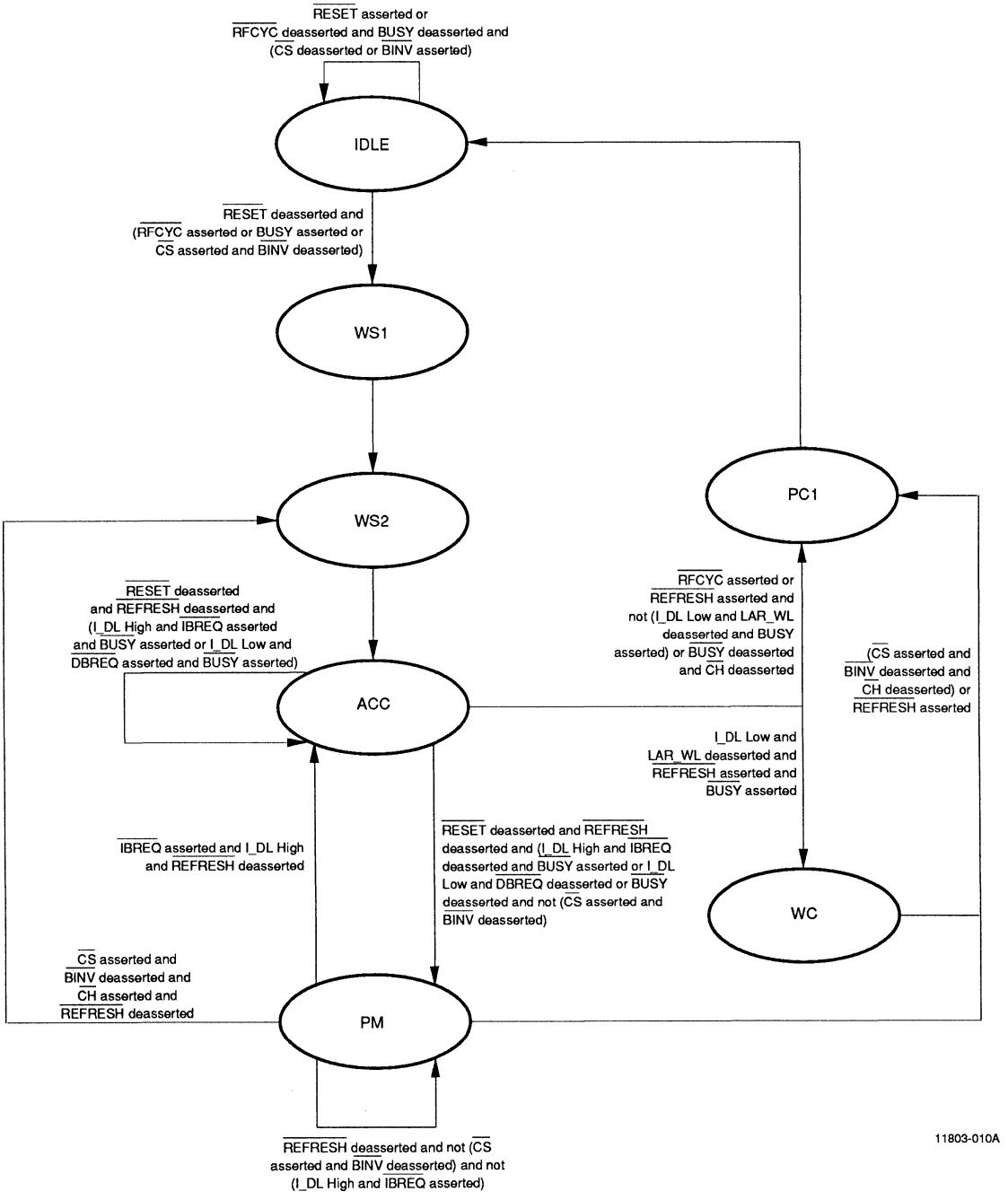


Figure 7. State Diagram

buffers on both instruction and data buses, other devices or memories can be attached to the same bus.

Data Memory Control

The data memory is organized like the instruction memory. However, the data memory is only accessible from the data bus; therefore, instruction buffers are not needed. The RASI and State PALs can be combined by eliminating the I_DL output and the $\overline{IREQ}$ input. The Ready and Control PALs can also be combined by eliminating the inputs and outputs associated with instruction accesses. The rest of the PALs are the same, except that the input I_DL is tied Low and $\overline{IBREQ}$ and $\overline{IREQ}$ are tied High.

TIMING ANALYSES

This design must meet many timing constraints. There are two different modes of operation, simple and burst accesses. The timing for simple accesses is discussed first. Note: all timings in this section are in nanoseconds unless otherwise stated.

Decoding is performed during the first cycle of an access. This timing is constrained by the cache-mode accesses. The following timing is required:

SYLCLK to Address	16	Am29000 Parameter 6
Address to ALE	10	PAL20L8-10 Delay
ALE to $\overline{CH}$	16	Am29C668 Parameter 34
$\overline{CH}$ Set-up	7	PAL16R6-7 t_{su}
Total	49	

Therefore, the Request PAL must be a PAL20L8-10 and the State PAL must be a PAL16R8-7, to meet the minimum set-up time for the State PAL. ALE is valid a minimum of 27 ns before RASI is asserted to insure that the row address is valid before $\overline{RAS}$ is asserted.

The next three cycles are the actual memory access. The slowest memory access occurs during data-memory Reads because the Am29C983A is slower than the 74F244. The load on the $\overline{RAS}$ output is 32 x 7 pF = 224 pF. 250 pF is assumed to account for other capacitive effects.

SYLCLK to RASI	7	PAL16R6-7 Delay
RASI to $\overline{RAS}$	21	Am29C668 Parameter 20
DRAM (tacc)	100	DRAM Access from $\overline{RAS}$
Buffer Delay	9	Am29C983A Delay
Data Set-up	8	Am29000 Parameter 9A
Total	145	

2 SYSCLK cycles	100	Am29000 Parameter 1
SYSCLK High Max	26	Am29000 Parameter 1A
SYSCLK to CASEN	10	PAL20L8-10 Delay
CASEN to $\overline{CAS_n}$	18	Am2966 Delay
DRAM (t_{cac})	25	DRAM Access from $\overline{CAS}$
Buffer Delay	9	Am29C983A Delay
Data Set-up	8	Am29000 Parameter 9A
Total	196	
SYSCLK Cycle	50	Am29000 Parameter 1
SYSCLK High Max	26	Am29000 Parameter 1A
SYSCLK to MSEL	10	PAL20L8-10 Delay
MSEL to Q_n	31	Am29C668 Parameter 19
DRAM (t_{aa})	50	DRAM Access from Column Address
Buffer Delay	9	Am29C983A Delay
Data Set-up	8	Am29000 Parameter 9A
Total	184	

The first timing is the access time from $\overline{RAS}$; the second timing is from $\overline{CAS}$ and the third is from the column address. This provides for the simple accesses to complete in four cycles, one cycle for address decode and three cycles for the memory access. The $\overline{RAS}$ precharge time for 100-ns DRAMs is 90 ns, therefore two cycles are more than sufficient.

The $\overline{IRDY}$ and $\overline{DRDY}$ signals require the following timing:

SYSCLK to ACC_ST	7	PAL16R6-7 Delay
ACC_ST to $\overline{DRDY}$	10	PAL16L8-10 Delay
External NOR gate	7.5	PAL16L8-7 Delay
$\overline{DRDY}$ Set-up	16	Am29000 Parameter 9B
Total	40.5	

For $\overline{IBACK}$ and $\overline{DBACK}$ the following timing applies:

SYSCLK to $\overline{IREQ}$	16	Am29000 Parameter 6
$\overline{IREQ}$ to $\overline{IBACK}$	10	PAL16L8-10 Delay
External NOR gate	7.5	PAL16L8-7 Delay
$\overline{IBACK}$ Set-up	15	Am29000 Parameter 9B
Total	48.5	

This means that a PAL16L8-10 is required for the Ready PAL if the external device that generates the ready and burst acknowledge signals to the Am29000 is a PAL16L8-7.

A burst access must take less than two cycles to meet the single-cycle burst requirement. The load on the outputs of the Am2966 is assumed to be 70 pF.

SYSCLK High Max	26	Am29000 Parameter 1A
SYSCLK to $\overline{\text{CASEN1}}$	10	PAL20L8-10 Delay
$\overline{\text{CASEN1}}$ to $\overline{\text{CAS}}$	18	Am2966 Delay
DRAM (t_{cac})	25	DRAM Access from $\overline{\text{CAS}}$
Buffer Delay	9	Am29C983A Delay
Data Set-up	8	Am29000 Parameter 9A
Total	96	

During burst accesses, the column address must meet the DRAMs setup time of 0 ns. The timing is for the address is:

SYSCLK High Max	26	Am29000 Parameter 1A
SYSCLK to $\overline{\text{CASEN1}}$	10	PAL20L8-10 Delay
$\overline{\text{CASEN1}}$ to $\overline{\text{CAS}}$	18	2966 Delay
$\overline{\text{CAS}}$ to RL_CC	10	PAL20L8-10 Delay
RL_CC to Q_n	30	Am29C668 Parameter 27
Total	94	

The $\overline{\text{CAS}}$ to bank 0 requires:

One Processor Cycle Min	50	Am29000 Parameter 1
Pre-Charge Cycle	26	1/2 SYSCLK Cycle
SYSCLK to CASIEN0	10	PAL20L8-10 Delay
CASIEN to $\overline{\text{CAS}}$	12	Am29C668 Parameter 26
Total	98	

Therefore, the address is valid 4 ns before $\overline{\text{CAS}}$. This is the difference between the slowest address to the fastest $\overline{\text{CAS}}$. For $\overline{\text{CAS}}$ precharge during the fast-page-mode accesses, $\overline{\text{CAS}}$ must be deasserted for at least 15 ns for most DRAMs. Some manufactures make DRAMs with $\overline{\text{CAS}}$-precharge time t_{cp} of 10 ns. Because the Am29C668 has symmetric outputs, the rise and fall times are the same. The outputs of the PAL driving the CASIEN input of the Am29C668 do not have symmetric rise and fall times. The t_{cp}, therefore, is shorter than the minimum clock-High time of the Am29000, 24 ns parameter 1A. For the minimum tcp to be violated, the skew between the rise and fall times of the PAL would have to be greater than 9 ns. This would never be the case; therefore the $\overline{\text{CAS}}$ precharge can be met under worst-case conditions.

For programming cycles, the address must be decode in one cycle:

SYSCLK to Address	16	Am29000 Parameter 6
Address to $\overline{\text{PRG_REQ}}$	10	PAL16L8-10 Delay
PROGRAM Set up	10	PAL20R4-10 t_{su}
Total	36	

The delay line used to generate CASEN0, $\overline{\text{CASEN1}}$[3:0] and LE[1:0] must delay SYSCLK long enough for $\overline{\text{BANK_PTR}}$ and $\overline{\text{BINV}}$ to reach their final values. If this delay is too long, the data buffers can latch the wrong data on Write cycles. Therefore, the Program PAL should be a PAL20R4-10. $\overline{\text{BANK_PTR}}$ is valid 8 ns after the rising edge of SYSCLK and BINV is valid 9 ns after the falling edge of SYSCLK. Therefore, the delay line must be at least 9 ns. Because most delay lines are accurate to ±2 ns, the delay line should be at least 11 ns. The maximum delay is computed as follows:

SYSCLK	50	Am29000 Parameter 1
SYSCLK to Data Hold time	4	AM29000 Parameter 20
SYSCLK High Max	-26	Am29000 Parameter 1A
CLOCKD to LE	-10	PAL20L8-10 t_{su}
Data Hold Time	-2.5	Am29C983A Parameter 14
Total	15.5	Maximum Delay

Therefore, a nominal value of 12 ns would be best for the delay line and the Control PAL should be a PAL16L8-10.

PARTS LIST

Part	Count
PAL16L8-10	2
PAL16R4-7	1
PAL16R6-7	1
PAL20L8-10	3
PAL20R4-10	1
PAL20X10A	1
Am2966	1
Am29C668	1
74F244	8
Am29C983A	4
Memories	64
12 ns Delay Line	1
Total	88

A discrete memory controller, described in the *Am29000 32-Bit Streamlined Instruction Processor Memory Design Handbook* Chapter 6, requires 113 devices including memory devices. The design, described here, represents a 22.1 % reduction in parts. The memory controller in utilizing the Am29C668 requires 12 devices while the discrete design requires 25 or 208 % more. The Am29C668 offers a high degree of integration, which reduces system cost, power consumption, board space and design time. However, the discrete design has one advantage in that it can operate in a 25-MHz system providing up to a 25 % performance improvement. The Am29C668 can also be used in higher speed systems, but the control logic must be changed.

VARIATIONS ON THIS DESIGN

One change to this design is to permit only data-Write accesses to instruction memory. This eliminates the output buffers on the data bus, reducing the required number of parts. The Am29C983As could be replaced with 74F373s to save cost.

This design is limited to 20 MHz due to the control logic. $\overline{\text{IBREQ}}$ and $\overline{\text{DBREQ}}$ are used by the control logic and are valid very late in the cycle. For faster versions of the Am29000, $\overline{\text{IBREQ}}$ and $\overline{\text{DBREQ}}$ do not allow for sufficient setup time to be registered by the PALs. However, the Am29C668 can be used in 30-MHz systems, presently the fastest Am29000, with a different control logic. Two banks of SCDRAMs with one Am29C668 controlling each bank provides maximum performance. Using 70-ns SCDRAMs, the initial cycle requires five cycles and burst accesses require one cycle.

PAL Equations

The following are the logic equations for the PAL devices. They are written in PLPL.

```
" March 6, 1989.
32-Bit Memory Design for the 29K.  This device generates and latches the byte enable
signals.  Opt[2:0] and A[1:0] are decoded to determine which bytes are
being accessed.  ALE is used to latch the values."

DEVICE      BYTE_ENABLE (P20L8)

PIN   ALE = 1 (INPUT COMBINATORIAL)
      OPT[2:0] = 2:4 (INPUT COMBINATORIAL)
      A[1:0] = 5:6 (INPUT COMBINATORIAL)
      /CASN1[3:0] = 7:10 (INPUT COMBINATORIAL)
      I_DL = 11 (INPUT COMBINATORIAL)
      R_WL = 13 (INPUT COMBINATORIAL)
      PROGRAM = 14 (INPUT COMBINATORIAL)
      /RESET = 16 (INPUT COMBINATORIAL)
      SYSCLK = 23 (INPUT COMBINATORIAL)

      RL_CC = 15 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      LAR_WL = 17 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /BE[3:0] = 21:18 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE BYTEORDER = 0;   "ASSUMES BIG ENDIAN"

BEGIN
ENABLE (BE[3:0],RL_CC,LAR_WL);

BE[0] =  /RESET * ALE * (R_WL + /OPT[2] * /OPT[1] * /OPT[0] +
      /OPT[2] * /OPT[1] * OPT[0] * /A[1] * /A[0] * BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * /A[1] * BYTEORDER +
      /OPT[2] * /OPT[1] * OPT[0] * A[1] * A[0] * /BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * A[1] * /BYTEORDER) +
      BE[0] * /ALE;
BE[1] = /RESET * ALE * (R_WL + /OPT[2] * /OPT[1] * /OPT[0] +
      /OPT[2] * /OPT[1] * OPT[0] * /A[1] * A[0] * BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * /A[1] * BYTEORDER +
      /OPT[2] * /OPT[1] * OPT[0] * A[1] * /A[0] * /BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * A[1] * /BYTEORDER) +
      BE[1] * /ALE;
BE[2] = /RESET * ALE * (R_WL + /OPT[2] * /OPT[1] * /OPT[0] +
      /OPT[2] * /OPT[1] * OPT[0] * A[1] * /A[0] * BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * A[1] * BYTEORDER +
      /OPT[2] * /OPT[1] * OPT[0] * /A[1] * A[0] * /BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * /A[1] * /BYTEORDER) +
      BE[2] * /ALE;
BE[3] = /RESET * ALE * (R_WL + /OPT[2] * /OPT[1] * /OPT[0] +
      /OPT[2] * /OPT[1] * OPT[0] * A[1] * A[0] * BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * A[1]* BYTEORDER +
      /OPT[2] * /OPT[1] * OPT[0] * /A[1] * /A[0] * /BYTEORDER +
      /OPT[2] * OPT[1] * /OPT[0] * /A[1]* /BYTEORDER) +
      BE[3] * /ALE;

/RL_CC = /PROGRAM * /(CASN1[0] + CASN1[1] + CASN1[2] + CASN1[3]) +
      PROGRAM * /SYSCLK;

/LAR_WL = ALE * (I_DL + /I_DL * R_WL) + LAR_WL * /ALE;
END.
```

```
"March 6, 1989  29K MEMORY SYSTEM.  This PAL will generate the /CAS Enable signals
to the external buffer and the Am29C668."
DEVICE      CAS_ENABLE  (P20L8)

PIN   SYSCLK = 1 (INPUT COMBINATORIAL)
      CLOCKD = 2 (INPUT COMBINATORIAL)
      /STATE[2:0] = 3:5 (INPUT COMBINATORIAL)
      /BE[3:0] = 6:9 (INPUT COMBINATORIAL)
      /CS = 10 (INPUT COMBINATORIAL)
      /RFCYC = 11 (INPUT COMBINATORIAL)
      /BINV = 13 (INPUT COMBINATORIAL)
      /RESET = 14 (INPUT COMBINATORIAL)
      /BANK_PTR = 23 (INPUT COMBINATORIAL)
      MSEL = 22 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      CASEN0 = 21 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /CASEN1[3:0] = 20:17 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /CASEN1 = 16 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE IDLE = #B000,    WS1 = #B010,      WS2 = #B110,      ACC = #B100,
      PC1 = #B001,      PM = #B000, WC  = #B101,
      IDLE_ST = /STATE[2] * /STATE[1] * /STATE[0],
      WS1_ST  = /STATE[2] *  STATE[1] * /STATE[0],
      WS2_ST  =  STATE[2] *  STATE[1] * /STATE[0],
      ACC_ST  =  STATE[2] * /STATE[1] * /STATE[0],
      PC1_ST  = /STATE[2] * /STATE[1] *  STATE[0],
      PM_ST   = /STATE[2] * /STATE[1] * /STATE[0],
      WC_ST   =  STATE[2] * /STATE[1] *  STATE[0];

BEGIN
ENABLE (CASEN0,CASEN1,CASEN1[3:0]), MSEL;
MSEL = (SYSCLK * WSI_ST + MSEL * /RESET */PLI_ST);
/CASEN0 = WS2_ST * BANK_PTR * /SYSCLK + ACC_ST * /SYSCLK +
      CASEN0 * /(SYSCLK * CASEN1 + PC1_ST +
            PM_ST * CS * /BINV * /CLOCKD * /SYSCLK);

CASEN1 = /SYSCLK * BANK_PTR *
            (CASEN1[0] + CASEN1[1] + CASEN1[2] + CASEN1[3]) * ACC_ST +
      PM_ST * /CASEN0 * /SYSCLK + SYSCLK * CASEN1;

CASEN1[0] = /RESET * /RFCYC * BE[0] * WS2_ST * /BANK_PTR * /SYSCLK +
      /RESET * /RFCYC * BE[0] * ACC_ST * /SYSCLK +
      /RESET * PM_ST * CASEN1[0] * /(CS * /BINV * /CLOCKD * /SYSCLK) +
      /RESET * CASEN1 * CASEN1[0] * ACC_ST;
CASEN1[1] = /RESET * /RFCYC * BE[1] * WS2_ST * /BANK_PTR * /SYSCLK +
      /RESET * /RFCYC * BE[1] * ACC_ST * /SYSCLK +
      /RESET * PM_ST * CASEN1[1] * /(CS * /BINV * /CLOCKD * /SYSCLK) +
      /RESET * CASEN1 * BE[1] * ACC_ST;
CASEN1[2] = /RESET * /RFCYC * BE[2] * WS2_ST * /BANK_PTR * /SYSCLK +
      /RESET * /RFCYC * BE[2] * ACC_ST * /SYSCLK +
      /RESET * PM_ST * CASEN1[2] * /(CS * /BINV * /CLOCKD * /SYSCLK) +
      /RESET * CASEN1 * BE[2] * ACC_ST;
CASEN1[3] = /RESET * /RFCYC * BE[3] * WS2_ST * /BANK_PTR * /SYSCLK +
      /RESET * /RFCYC * BE[3] * ACC_ST * /SYSCLK +
      /RESET * PM_ST * CASEN1[3] * /(CS * /BINV * /CLOCKD * /SYSCLK) +
      /RESET * CASEN1 * BE[3] * ACC_ST;
END.
```

```
"March 6, 1989
Memory Design for the 29K.  This PAL will generate the signals that control the
memory data and instruction buffers.  It will also generate /RFCYC that indicates
when the control logic is performing a memory refresh cycle."

DEVICE      CONTROL (P16L8)

PIN   CLOCKD = 1 (INPUT COMBINATORIAL)
      /STATE[3:0] = 2:5 (INPUT COMBINATORIAL)
      I_DL = 6 (INPUT COMBINATORIAL)
      LAR_WL = 7 (INPUT COMBINATORIAL)
      /REFRESH = 8 (INPUT COMBINATORIAL)
      /BANK_PTR = 9 (INPUT COMBINATORIAL)

      /OE_DOUT[1:0] = 19:18 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /OE_INS[1:0] = 17:16 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      LE[1:0] = 15:14 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /RFCYC = 13 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /OE_DIN = 13 (OUTPUT ACTIVE_LOW COMBINATORIAL)
;

DEFINE IDLE = #B0000,   WS1 = #B0010,     WS2 = #B0110,     ACC = #B0100,
      PC1 = #B0001,     PM = #B1000,      WC  = #B0101,     UN1 = #B0011,
      UN2 = #B0111,     UN3 = #B1001,     UN4 = #B1010,     UN5 = #B1011,
      UN6 = #B1100,     UN7 = #B1101,     UN8 = #B1110,     UN9 = #B1111,
      IDLE_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WS1_ST  = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      WS2_ST  = /STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      ACC_ST  = /STATE[3] *  STATE[2] * /STATE[1] * /STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PM_ST   =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WC_ST   = /STATE[3] *  STATE[2] * /STATE[1] *  STATE[0];


BEGIN

ENABLE (OE_DOUT[1:0],OE_INS[1:0],LE[1:0]);

OE_DOUT[0] = /RFCYC * /I_DL * LAR_WL *  BANK_PTR * ACC_ST;
OE_DOUT[1] = /RFCYC * /I_DL * LAR_WL * /BANK_PTR * ACC_ST;

OE_INS[0] = /RFCYC * I_DL *  BANK_PTR * ACC_ST;
OE_INS[1] = /RFCYC * I_DL * /BANK_PTR * ACC_ST;

OE_DIN = /RFCYC * /I_DL * /LAR_WL * (WS1_ST + WS2_ST + ACC_ST + PM_ST);

/LE[0] = WS1_ST * BANK_PTR * /LAR_WL +
      ACC_ST * /BANK_PTR * /LAR_WL * CLOCKD;
/LE[1] = WS1_ST * /BANK_PTR * /LAR_WL +
      CLOCKD * ACC_ST * BANK_PTR * /LAR_WL * CLOCKD;

RFCYC = IDLE_ST * REFRESH * /RESET + RFCYC * /RESET * /PC1_ST;

END.
```

```
"January 30, 1989.
29K memory design.  This PAL decodes the addresses for programming request.  It will
also keep track of which bank is currently active through the output /BANK_PTR."

DEVICE      Program (P20R4)

PIN    SYSCLK = 1 (CLOCK)
       /STATE[3:0] = 2:5 (INPUT COMBINATORIAL)
       A[2] = 6 (INPUT COMBINATORIAL)
       ALE = 7 (INPUT COMBINATORIAL)
       /BINV = 8 (INPUT COMBINATORIAL)
       /REFRESH = 9 (INPUT COMBINATORIAL)
       /PRG_DEC = 10 (INPUT COMBINATORIAL)
       A[22:18] = 11,14,21:23 (INPUT COMBINATORIAL)
       /OE = 13 (CONTROL)

       /PRG_REQ = 15 (OUTPUT ACTIVE_LOW COMBINATORIAL)
       /BANK_PTR = 20 (OUTPUT ACTIVE_LOW REGISTERED)
       PROGRAM = 19 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE IDLE = #B0000, WS1 = #B0010, WS2 = #B0110, ACC = #B0100,
       PC1 = #B0001, PM  = #B1000, WC  = #B0101, UN1 = #B0011,
       UN2 = #B0111, UN3 = #B1001, UN4 = #B1010, UN5 = #B1011,
       UN6 = #B1100, UN7 = #B1101, UN8 = #B1110, UN9 = #B1111,
       IDLE_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
       WS1_ST  = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
       WS2_ST  = /STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
       ACC_ST  = /STATE[3] *  STATE[2] * /STATE[1] * /STATE[0],
       PC1_ST  = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
       PM_ST   =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
       WC_ST   = /STATE[3] *  STATE[2] * /STATE[1] *  STATE[0],
       VALID_ADDR = A[22] * A[21] * A[20] * A[19] * A[18];

BEGIN

ENABLE (PRG_REQ);

PRG_REQ = VALID_ADDR;

/PROGRAM := PRG_REQ * /BINV * /REFRESH * PRG_DEC;

BANK_PTR := ALE * /A[2] +
       /ALE * BANK_PTR * /(ACC_ST + WC_ST) +
       (ACC_ST + WC_ST) * /BANK_PTR;

END.
```

```
"March 6, 1989
29K Memory Design.  This PAL will generate the RASI input to the Am29C668.
"

DEVICE      RASI (P16R4)

PIN   SYSCLK = 1 (CLOCK)
      /STATE[3:0] = 2:5 (INPUT COMBINATORIAL)
      /BUSY = 6 (INPUT COMBINATORIAL)
      /REFRESH = 7 (INPUT COMBINATORIAL)
      /BINV = 8 (INPUT COMBINATORIAL)
      /RESET = 9 (INPUT COMBINATORIAL)
      /CS = 12 (INPUT COMBINATORIAL)
      /CH = 13 (INPUT COMBINATORIAL)
      /IREQ = 19 (INPUT COMBINATORIAL)
      /OE = 11 (CONTROL)

      /RASI_OFF = 18 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /RASI = 17 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE IDLE = #B0000, WS1 = #B0010, WS2 = #B0110,     ACC = #B0100,
      PC1 = #B0001,   PM = #B1000,  WC  = #B0101,     UN1 = #B0011,
      UN2 = #B0111,   UN3 = #B1001, UN4 = #B1010,     UN5 = #B1011,
      UN6 = #B1100,   UN7 = #B1101, UN8 = #B1110,     UN9 = #B1111,
      IDLE_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WS1_ST  = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      WS2_ST  = /STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      ACC_ST  = /STATE[3] *  STATE[2] * /STATE[1] * /STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PM_ST   =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WC_ST   = /STATE[3] *  STATE[2] * /STATE[1] *  STATE[0];

BEGIN
ENABLE (RASI_OFF);

/RASI := /(ACC_ST * REFRESH +
      ACC_ST * /BUSY * CS * /BINV * /CH +
      PM_ST * REFRESH +
      PM_ST * CS * /BINV * /CH * /BUSY +
      PM_ST * CS * /BINV * /CH * BUSY * IREQ +
      IDLE_ST * /BUSY * /REFRESH * /(CS * /BINV) +
      RASI_OFF);

RASI_OFF = RESET + WC_ST + PC1_ST;

END.
```

```
"March 6, 1989
32-bit Memory for the 29K.  This PAL will generate the control signals back to the
Am29000.  It assumes that the control signals are externally combined to obtain the
signal connected to the Am29000."

DEVICE              READY (P16L8)

PIN    I_DL = 1 (INPUT COMBINATORIAL)
       /IBREQ = 2 (INPUT COMBINATORIAL)
       /DBREQ = 3 (INPUT COMBINATORIAL)
       /STATE[2:0] = 4:6 (INPUT COMBINATORIAL)
       LAR_WL = 7 (INPUT COMBINATORIAL)
       /IREQ = 8 (INPUT COMBINATORIAL)
       /RFCYC = 9 (INPUT COMBINATORIAL)
       /BUSY = 11 (INPUT COMBINATORIAL)

       /DRDY = 19 (OUTPUT ACTIVE_LOW COMBINATORIAL)
       /IRDY = 18 (OUTPUT ACTIVE_LOW COMBINATORIAL)
       /DBACK = 17 (OUTPUT ACTIVE_LOW COMBINATORIAL)
       /IBACK = 16 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE IDLE = #B000,     WS1 = #B010,      WS2 = #B110,      ACC = #B100,
       PC1 = #B001,      PM = #B000, WC  = #B101,
       IDLE_ST = /STATE[2] * /STATE[1] * /STATE[0],
       WS1_ST  = /STATE[2] *  STATE[1] * /STATE[0],
       WS2_ST  =  STATE[2] *  STATE[1] * /STATE[0],
       ACC_ST  =  STATE[2] * /STATE[1] * /STATE[0],
       PC1_ST  = /STATE[2] * /STATE[1] *  STATE[0],
       PM_ST   = /STATE[2] * /STATE[1] * /STATE[0],
       WC_ST   =  STATE[2] * /STATE[1] *  STATE[0];
BEGIN

ENABLE (DBACK,DRDY,IRDY,IBACK);

IRDY = /RFCYC * I_DL * ACC_ST;

IBACK = /RFCYC * I_DL * IBREQ * (WS1_ST + WS2_ST) +
       IBACK * /IREQ * I_DL * BUSY;

DRDY = /RFCYC * /I_DL * /LAR_WL * WS2_ST +
       /RFCYC * /I_DL * /LAR_WL * ACC_ST * BUSY +
       /RFCYC * /I_DL * LAR_WL * ACC_ST;

DBACK = /RFCYC * /I_DL * DBREQ * (WS1_ST + WS2_ST) +
       DBACK * DBREQ * /I_DL * BUSY;

END.
```

```
"March 6, 1989
Memory System for the Am29000.  This PAL decodes valid data and instruction memory
requests.  It will also partially decode programming requests."

DEVICE      Request (P20L8)

PIN   /IREQ = 1 (INPUT COMBINATORIAL)
      IREQT = 2 (INPUT COMBINATORIAL)
      /DREQ = 3 (INPUT COMBINATORIAL)
      DREQT[1:0] = 4,5 (INPUT COMBINATORIAL)
      OPT[2:0] = 6:8 (INPUT COMBINATORIAL)
      /BUSY = 9 (INPUT COMBINATORIAL)
      /STATE[3] = 10 (INPUT COMBINATORIAL)
      A[31:23] = 11,13,14,16:20,23 (INPUT COMBINATORIAL)

      /PRG_DEC = 22 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /CS = 21 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      ALE = 15 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE
      IOREQ = DREQ * DREQT[0] * /DREQT[1],
      DATAREQ = DREQ * /DREQT[0] * /DREQT[1] * /OPT[2] *
            (/OPT[1] * /OPT[0] + /OPT[1] * OPT[0] + OPT[1] * /OPT[0]),
      INSREQ = IREQ * /IREQT,
      MEMADDR = /A[31] * /A[30] * /A[29] * /A[28] * /A[27] * /A[26] *
                  /A[25] * /A[24] * /A[23];

BEGIN

ENABLE (PRG_DEC,ALE,CS);

/ALE = /BUSY + STATE[3] * INSREQ;

PRG_DEC = IOREQ;

CS = MEMADDR * (INSREQ + DATAREQ);

END.
```

```
"March 20, 1989.
This is the refresh timer for the 29K design.  This PAL will uses a 20 MHz clock to
generate a refresh request every 9.8 µs.  Refresh will remain asserted until /RFCYC
and the memory is accessed.  The initial value can be set by changing the constant
associated with INIT.  If the initial value is to be a 1, then INIT * 1 should be in
the RCT[n] equation, if the initial value is to be a 0 then INIT * 1 should be in
the XOR(RCT[n]) equation.  RCT[0] is initialized to 1 and RCT[4] is initialize to
0."

DEVICE            TIMER (P20X10)

PIN   SYSCLK = 1 (CLOCK)
      /RESET = 2 (INPUT COMBINATORIAL)
      /RFCYC = 3 (INPUT COMBINATORIAL)
      /STATE[3:0] = 4:7 (INPUT COMBINATORIAL)
      /OE = 13 (CONTROL)

      /RCT[7:0] = 22:15 (OUTPUT ACTIVE_LOW REGISTERED)
      /REFRESH = 14 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE IDLE = #B0000,   WS1 = #B0010,     WS2 = #B0110,     ACC = #B0100,
      PC1 = #B0001,     PM = #B1000,      WC  = #B0101,     UN1 = #B0011,
      UN2 = #B0111,     UN3 = #B1001,     UN4 = #B1010,     UN5 = #B1011,
      UN6 = #B1100,     UN7 = #B1101,     UN8 = #B1110,     UN9 = #B1111,
      IDLE_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WS1_ST  = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      WS2_ST  = /STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      ACC_ST  = /STATE[3] *  STATE[2] * /STATE[1] * /STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PM_ST   =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WC_ST   = /STATE[3] *  STATE[2] * /STATE[1] *  STATE[0],
      START_REFRESH = /RCT[7] * /RCT[6] * /RCT[5] * /RCT[4] *
            /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0],
      INIT = /RCT[7] * /RCT[6] * /RCT[5] * /RCT[4] * /RCT[3] *
            /RCT[2] * /RCT[1] * /RCT[0];

BEGIN

REFRESH := START_REFRESH * /RESET + REFRESH * /RESET;
XOR(REFRESH) := RFCYC * ACC_ST * /RESET;

RCT[0] := /RCT[0] + INIT * 1;
XOR(RCT[0]) := INIT * 0;

RCT[1] := /RCT[0] + INIT * 1;
XOR(RCT[1]) := RCT[1] + INIT * 0;

RCT[2] := /RCT[1] * /RCT[0] + INIT * 0;
XOR(RCT[2]) := RCT[2] + INIT * 1;

RCT[3] := /RCT[2] * /RCT[1] * /RCT[0] + INIT * 0;
XOR(RCT[3]) := RCT[3] + INIT * 1;

RCT[4] := /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0] + INIT * 0;
XOR(RCT[4]) := RCT[4] + INIT * 1;
```

```
RCT[5] := /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0] + INIT * 0;
XOR(RCT[5]) := RCT[5] + INIT * 1;

RCT[6] := /RCT[5] * /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0] +
            INIT * 1;
XOR(RCT[6]) := RCT[6] + INIT * 0;

RCT[7] := /RCT[6] * /RCT[5] * /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0]+
      INIT * 1;
XOR(RCT[7]) := RCT[7] + INIT * 0;

END.
```

```
"March 6, 1989
32-Bit Memory for the 29K.  This PAL implements the state machine for this design.
It will indicate what the current state of the memory is."

DEVICE      MEMORY_STATE_MACHINE (P16R6)

PIN   SYSCLK = 1 (CLOCK)
      /CS = 2 (INPUT COMBINATORIAL)
      /DBREQ = 3 (INPUT COMBINATORIAL)
      LAR_WL = 4 (INPUT COMBINATORIAL)
      /IBREQ = 5 (INPUT COMBINATORIAL)
      /CH = 6 (INPUT COMBINATORIAL)
      /IREQ = 7 (INPUT COMBINATORIAL)
      /RESET = 8 (INPUT COMBINATORIAL)
      /REFRESH = 9 (INPUT COMBINATORIAL)
      /BINV = 19 (INPUT COMBINATORIAL)
      /OE = 11 (CONTROL)

      /STATE[3:0] = 18:15 (OUTPUT ACTIVE_LOW REGISTERED)
      /I_DL = 14 (OUTPUT ACTIVE_LOW REGISTERED)
      /BUSY = 13 (OUTPUT ACTIVE_LOW REGISTERED)
      /NOT_BUSY = 12 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE IDLE = #B0000,   WS1 = #B0010,     WS2 = #B0110,     ACC = #B0100,
      PC1 = #B0001,     PM = #B1000,      WC  = #B0101,     UN1 = #B0011,
      UN2 = #B0111,     UN3 = #B1001,     UN4 = #B1010,     UN5 = #B1011,
      UN6 = #B1100,     UN7 = #B1101,     UN8 = #B1110,     UN9 = #B1111,
      IDLE_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WS1_ST  = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      WS2_ST  = /STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      ACC_ST  = /STATE[3] *  STATE[2] * /STATE[1] * /STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PM_ST   =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      WC_ST   = /STATE[3] *  STATE[2] * /STATE[1] *  STATE[0];

BEGIN
ENABLE(NOT_BUSY);

BUSY := /BUSY * (IDLE_ST + PM_ST + ACC_ST) * CS * /BINV * /RESET +
      BUSY * /NOT_BUSY;

NOT_BUSY = RESET +
      WS2_ST * I_DL * /LAR_WL * /DBREQ +
      ACC_ST * I_DL * /DBREQ +
      ACC_ST * /I_DL * CS * /BINV +
      PM_ST * /I_DL * CS * /BINV +
      PM_ST * IREQ * /CS;

I_DL := (IDLE_ST + PM_ST + ACC_ST) * /BUSY * /IREQ * CS * /BINV +
      (PM_ST) *  BUSY * IREQ * /CS +
      (IDLE_ST + ACC_ST) * /CS * /BUSY +
      I_DL * (WS1_ST + WS2_ST +
            ACC_ST * BUSY + PC1_ST * BUSY + IDLE_ST * BUSY) +
      RESET;

CASE (STATE[3:0]) BEGIN
```

```
      IDLE) BEGIN
            IF (/RESET * (REFRESH + CS * /BINV + BUSY)) THEN
                  STATE[3:0] := WS1;
            IF (/REFRESH * /BUSY * (/CS + BINV)) THEN
                  STATE[3:0] := IDLE;
            END;
      WS1) BEGIN
            IF (RESET) THEN STATE[3:0] := IDLE;
            ELSE STATE[3:0] := WS2;
           END;
      WS2) BEGIN
            IF (RESET) THEN STATE[3:0] := IDLE;
            IF (/RESET) THEN STATE[3:0] := ACC;
           END;
      ACC) BEGIN
            IF (RESET) THEN STATE[3:0] := IDLE;
            IF (/RESET * REFRESH) THEN
                  IF (I_DL * /LAR_WL * BUSY) THEN STATE[3:0] := WC;
                  ELSE STATE[3:0] := PC1;
            IF (/RESET * /REFRESH * BUSY *
                        (/I_DL * IBREQ + I_DL * DBREQ)) THEN
                  STATE[3:0] := ACC;
            IF (/RESET * /REFRESH *
                  (/I_DL * /IBREQ * BUSY + I_DL * /DBREQ) THEN
                  STATE[3:0] := PM;
            IF (/RESET * /REFRESH * /BUSY * CH * CS * /BINV) THEN
                  STATE[3:0] := WS2;
            IF (/RESET * /REFRESH * /BUSY * /CH * CS * /BINV) THEN
                  STATE[3:0] := PC1;
            IF (/RESET * /REFRESH * /BUSY * /(CS * /BINV)) THEN
                  STATE[3:0] := PM;
             END;
      PC1) STATE[3:0] := IDLE;
      PM) BEGIN
            IF (/RESET * (REFRESH + (/BUSY + IREQ) * CS * /CH * /BINV)) THEN
                  STATE[3:0] := PC1;
            IF (/RESET * /REFRESH * (/BUSY + IREQ) * CS * CH * /BINV) THEN
                  STATE[3:0] := WS2;
            IF (/RESET * /REFRESH * BUSY * IBREQ * /I_DL) THEN
                  STATE[3:0] := ACC;
            IF (/RESET * /REFRESH * /(CS * /BINV * (/BUSY + IREQ) *
                  /(IBREQ * /I_DL * BUSY)) THEN
                  STATE[3:0] := PM;
            IF (RESET) THEN STATE[3:0] := IDLE;
          END;
      WC) IF (RESET) THEN STATE[3:0] := IDLE;
            ELSE STATE[3:0] := PC1;
      UN1,UN2,UN3,UN4,UN5,UN6,UN7,UN8,UN9) STATE[3:0] := IDLE;
          END;
END;  "CASE"

END.
```

Am29C668 Configurable Dynamic Memory Controller to 80C286 Microprocessor Interface

by Douglas Lee, Applications Specialist

INTRODUCTION

The interface between the Am29C668 4-MBit Configurable Dynamic Memory Controller/Driver (CDMC) and the AMD 80C286 microprocessor was designed to provide maximum performance at reasonable cost. This design is as general as possible so that the user may tailor his implementation to a specific memory system. Possible changes to the design are discussed with associated system requirements and implications. A block diagram, timing analyses and logic equations necessary to implement the design are included. This design requires a minimum number of external devices to perform the interface and glue functions: two PAL™ devices (one PAL16R8, one PAL20R4), a 555 timer and an inverter.

Distinctive Characteristics

- Am29C668 4-MBit Configurable Dynamic Memory Controller/Driver, With Selectable Auto-Timing or External-Timing Mode
- 20-MHz 80C286 Microprocessor (may be upgraded to 25 MHz)
- 85-ns Fast-Page-Mode 1 Mbit x 1 DRAM
- One Wait-State Initial Accesses With Zero Wait-State Subsequent Page-Mode Accesses
- 8-Mbyte Dynamic Memory per Am29C668

MEMORY ARCHITECTURE OVERVIEW

To obtain the maximum memory throughput but still maintain a reasonable cost, 85-ns fast page-mode 1-MBit DRAMs are used. The 80C286 requires a minimum of two processor cycles per access. If additional cycles are needed, the external logic holds $\overline{\text{READY}}$ inactive and the processor inserts wait states until $\overline{\text{READY}}$ is asserted. For this memory design, the 20-MHz 80C286 completes the initial access to memory in one wait state (three processor cycles total). The subsequent accesses within the page are performed with no wait states (two processor cycles).

The page size for a 1-Mbit DRAM is 1024 bits or 1 Kbits. This memory is 16 bits wide, therefore the page size is 2 Kbytes. The Am29C668 detects accesses within the same page via on-chip cache-mode operation; consequently, page-mode DRAMs appear to the processor as if they are fast cache memories. When a new address is latched, it is compared with the previous row and bank address; if the addresses are to the same row and bank, the Cache Hit signal $\overline{\text{CH}}$ is asserted. The memory state machine immediately begins the next access. An access outside the page, a page miss, causes the memory controller to perform the $\overline{\text{RAS}}$ precharge for the DRAMs. The total access time on a page miss requires five processor-clock cycles, one for decoding, two cycles for the $\overline{\text{RAS}}$ precharge and two for the data access. This method of accessing memory results in shorter access times than memories using normal DRAM accesses. For certain systems, this memory can result in near-zero wait-state accesses. Only static RAMs can guarantee zero wait-state accesses. The actual performance of the memory depends upon the instruction mix of the programs executed.

The memory array consists of four banks. Each bank contains 2 Mbytes or 1 Mword (16-bits) of memory that provides a maximum memory size of 8 Mbytes or 4 Mwords per Am29C668 controller.

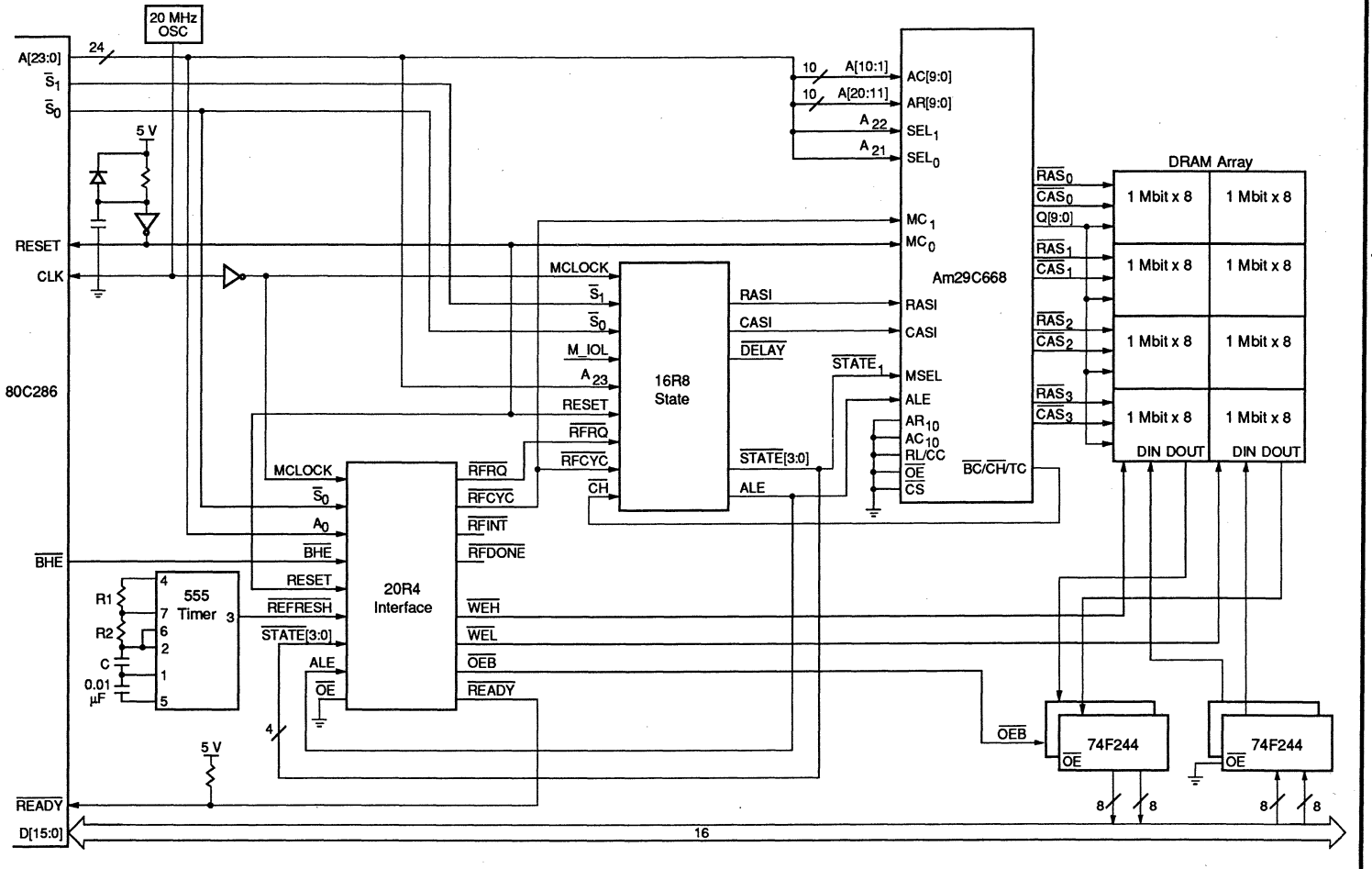
A 20-MHz system was selected since high performance is achieved with 85-ns DRAMs. For a 25-MHz processor using the same memory architecture, a single-wait-state memory would require 60-ns DRAMs. The State PAL would have to be implemented using a PAL16L8-7 device to meet the faster clock rate and shorter set-up times. This would be very expensive. For such a system, a cache could be a more cost-effective solution than using fast expensive DRAMs.

The 80C286 generates its internal processor clock from an external oscillator. The external clock oscillates at twice the speed of the 80C286 processor clock. The external crystal also provides the clock for the memory control logic. The oscillator signal is inverted to obtain MCLOCK for the memory control logic. The 80C286 20-MHz clock is referred to as the processor clock or CLK and the memory controller 40-MHz clock is called the memory clock or MCLOCK to avoid confusion.

FUNCTIONAL DESCRIPTION

The primary data paths and functional elements are shown in Figure 1. The following discussion describes each subsection of the block diagram, including the control logic, timer, buffers and memory array.

PAL is a registered trademark of Advanced Micro Devices.



03552-001A

Figure 1. Interface Block Diagram

Am29C668 CDMC

The Am29C668 CDMC generates the $\overline{RAS}$, $\overline{CAS}$ and address signals to the DRAM array; no external drivers are needed. Additionally, the Am29C668 generates the row addresses during $\overline{RAS}$-only refreshes from its internal row-refresh counter. The Am29C668 timing is controlled externally by the State PAL and the Interface PAL.

The State PAL generates the RASI, CASI, MSEL and $\overline{ALE}$ inputs to the Am29C668. RASI controls the $\overline{RAS}_n$ outputs. When RASI is asserted, the proper $\overline{RAS}_n$ outputs are enabled. CASI similarly controls the $\overline{CAS}_n$ outputs. During refresh and reset, the $\overline{CAS}_n$ outputs are always suppressed by the Am29C668. MSEL controls the multiplexing of the row and column address. When MSEL is Low, the row address is output on the Q[9:0] outputs of the Am29C668; when MSEL is High, the column address is output. If the Am29C668 is in the read/write mode, the address is taken from the address latch; in the refresh mode, the address is taken from the row-refresh counter.

The Interface PAL controls the Am29C668 mode of operation: read/write, refresh or reset. The Refresh Cycle output $\overline{RFCYC}$ is connected to the MC_1 input of the Am29C668. When $\overline{RFCYC}$ is active Low, the Am29C668 performs refresh cycles. When $\overline{RFCYC}$ is not active, i.e., High, the Am29C668 is either in read/write mode or reset. If RESET is asserted, the control logic forces the Am29C668 into the reset mode, resetting the refresh counters to zero and reconfiguring the CDMC to the default mode. The memory controller initiates "wake up" cycles to the DRAMs until RESET is deasserted. If RESET and $\overline{RFCYC}$ are deasserted, the Am29C668 is in the read/write mode.

The $\overline{CH}$ signal from the Am29C668 is used to determine if the current address in the input latch has the same row and bank address as the previous memory access. If the current access is to the same bank and row, $\overline{CH}$ is asserted and a page-mode access is initiated. If $\overline{CH}$ is deasserted, $\overline{RAS}$ must be precharged and a normal access occurs.

For this application, the Am29C668 is used in the default configuration, with external timing, 4-bank configuration, 1-Mbit DRAMs, $\overline{CAS}$ bank decoding, $\overline{RAS}$-only refresh and cache mode. It is possible to reconfigure the Am29C668 to provide additional features not used in this application (see the Am29C668 data sheet).

Interface PAL

This PAL generates the ready signal to the 80C286, refresh signals to the control logic, the output enable to the buffers, and write enables to the DRAMs. The following inputs are used:

MCLOCK	Inverted 40-MHz system clock
$\overline{S0}$	Status, from 80C286
A[0]	Address bit 0, from 80C286
$\overline{BHE}$	Bus High Enable, from 80C286
RESET	System Reset
$\overline{REFRESH}$	Refresh, from 555 timer
$\overline{STATE}$[3:0]	State variables, from State PAL
ALE	Address Latch Enable, from State PAL

The following outputs are generated:

$\overline{READY}$	Ready, to 80C286
$\overline{RFINT}$	Refresh Intermediate, used to synchronize refresh requests
$\overline{RFRQ}$	Refresh Request, to State PAL
$\overline{RFCYC}$	Refresh Cycle, Indicates if access is a refresh cycle
$\overline{RFDONE}$	Refresh Done
$\overline{WEL}$	Write Enable Low, to DRAM array
$\overline{WEH}$	Write Enable High, to DRAM array
$\overline{OEB}$	Output Enable Buffer, to Data Output Buffers

The $\overline{READY}$ output is a three-state output and requires an external pull-up resistor to keep it deasserted. When $\overline{READY}$ is asserted, it signals the 80C286 that the current memory access is completed.

$\overline{OEB}$ enables the outputs of the data drivers during processor reads. The memory has separate data input and output lines since all write cycles are late write cycles.

$\overline{WEL}$ and $\overline{WEH}$ control the write cycles to the memory data bytes; $\overline{WEL}$ controls the lower memory byte and $\overline{WEH}$ controls the upper byte. $\overline{WEL}$ is asserted during write cycles when A[0] is Low; $\overline{WEH}$ is asserted during write cycles when $\overline{BHE}$ is Low. Since the status bit must be decoded to determine if the access is a read or write cycle, $\overline{CAS}_n$ to the DRAMs may occur before $\overline{WEL}$ or $\overline{WEH}$ is asserted, enabling the DRAMs output drivers. This requires that the data input and data output lines be separate.

$\overline{RFINT}$, $\overline{RFRQ}$, $\overline{RFCYC}$ and $\overline{RFDONE}$ control the refresh cycles. The 555 timer controls the refresh interval and generates refresh requests by driving the $\overline{REFRESH}$ input Low. This signal is asynchronous to the memory clock, therefore it must be synchronized. Two internal

D-flip-flops, $\overline{RFINT}$ and $\overline{RFRQ}$, are used to synchronize the refresh requests to avoid any metastability. $\overline{RFDONE}$ is used to prevent $\overline{REFRESH}$, which is active for a long time, from generating multiple refresh cycles. Once the refresh cycle is initiated, $\overline{RFDONE}$ is asserted and remains so until $\overline{REFRESH}$ is deasserted. $\overline{RFCYC}$ is connected to the Am29C668 MC_1 input. Since $\overline{RFCYC}$ is normally High, the Am29C668 is normally in the read/write mode.

The system reset, RESET, is tied directly to the MC_0 input of the Am29C668. When RESET is asserted, the memory may be corrupted due to a violation of the DRAM timing requirements. It is assumed, for this application, that RESET indicates that the system is initializing or that an unrecoverable failure has occurred; therefore, the memory contents may be unreliable. If the application requires that memory be retained during system resets, a new signal MC_0 must be generated by the control logic. This signal goes active when RESET is asserted and the state machine is in the IDLE, PC1, PC2A or PC2B state. MC_0 must remain active until the state machine reaches the page-mode PM state. When RESET goes active, $\overline{RFRQ}$ is asserted and $\overline{RFCYC}$ is deasserted. The memory control logic asserts RASI and holds it asserted for four MCLOCK cycles. The falling edge of RASI, while MC_0 and MC_1 are High, resets the Am29C668 to the default configuration; all internal refresh counters are initialized to zero. The memory control logic continues to generate reset cycles until RESET is deasserted. The reset cycles also generate the "wake up" cycles to the DRAMs.

State PAL

The State PAL arbitrates between memory accesses and refresh cycles, generates the control signals to the Am29C668 and contains the state machine that controls the memory accesses. The following inputs are used:

MCLOCK	Inverted 40 MHz system clock
$\overline{S0}$, $\overline{S1}$	Status, from 80C286
M_IOL	Memory High, I/O Low, from 80C286
A[23]	Address bit 23, from 80C286
RESET	System Reset signal
$\overline{CH}$	Cache Hit, from Am29C668
$\overline{RFRQ}$	Refresh request signal, from the Interface PAL
$\overline{RFCYC}$	Refresh Cycle, from the Interface PAL

The following outputs are generated:

$\overline{STATE}$[3:0]	State variables for the state machine
RASI	Row Address Strobe Input, input to the Am29C668
CASI	Column Address Strobe Input, input to the Am29C668
ALE	Address Latch Enable, input to the Am29C668
$\overline{DELAY}$	Delay Cycle, used to synchronize memory accesses

The state variables $\overline{STATE}$[3:0] are used to keep track of the current status of the memory accesses. There are 16 states used, including six undefined states, as shown in the state diagram (Figure 2). If the state machine powers up in one of these six undefined states, the state machine goes to the IDLE state on the next clock. Figure 3 shows the timing of an initial access followed by a fast-page-mode access and an off-page access.

When a valid memory request occurs, the state machine begins a long initial access. RASI is asserted and the state machine goes to the WS1 state. From the WS1 state, the controller unconditionally goes to WS2A, WS2B, ACCA and then ACCB. In the WS2A state, the MSEL input of the Am29C668 is asserted so that the column address can propagate to the DRAMs. This input is tied to the state machine output $\overline{STATE}$[1]. In the WS2B state, CASI is asserted. These timings insure that the DRAM address setup and hold times are met relative to the falling edges of $\overline{RAS}$ and $\overline{CAS}$.

From the ACCB state, the memory controller goes to the PM state. The controller waits here until there is a refresh request or a valid memory request. If there is a memory request, $\overline{CH}$ is used to determine if the access is to the currently active page. If the access is to the same page, then the memory controller goes to the WS2B, ACCA and ACCB states, completing the access in just two cycles. If the access is outside the current page, a $\overline{RAS}$ precharge cycle must be performed followed by a long access. The $\overline{RAS}$ precharge cycle is performed by deasserting RASI and going from the PM state to PC1, PC2A, PC2B and then to the IDLE state. From the IDLE state, the memory controller performs a long initial access.

Refresh cycles perform the same accesses as memory accesses with a few exceptions. If a refresh request occurs while the memory controller is in the IDLE state, the refresh access waits one cycle to allow the row-refresh address adequate setup time relative to $\overline{RAS}$. The refresh access goes from the IDLE state to WS1, WS2A, WS2B states, then to the PM state. The memory controller must then perform a $\overline{RAS}$ precharge in the same manner it is performed during page-mode misses.

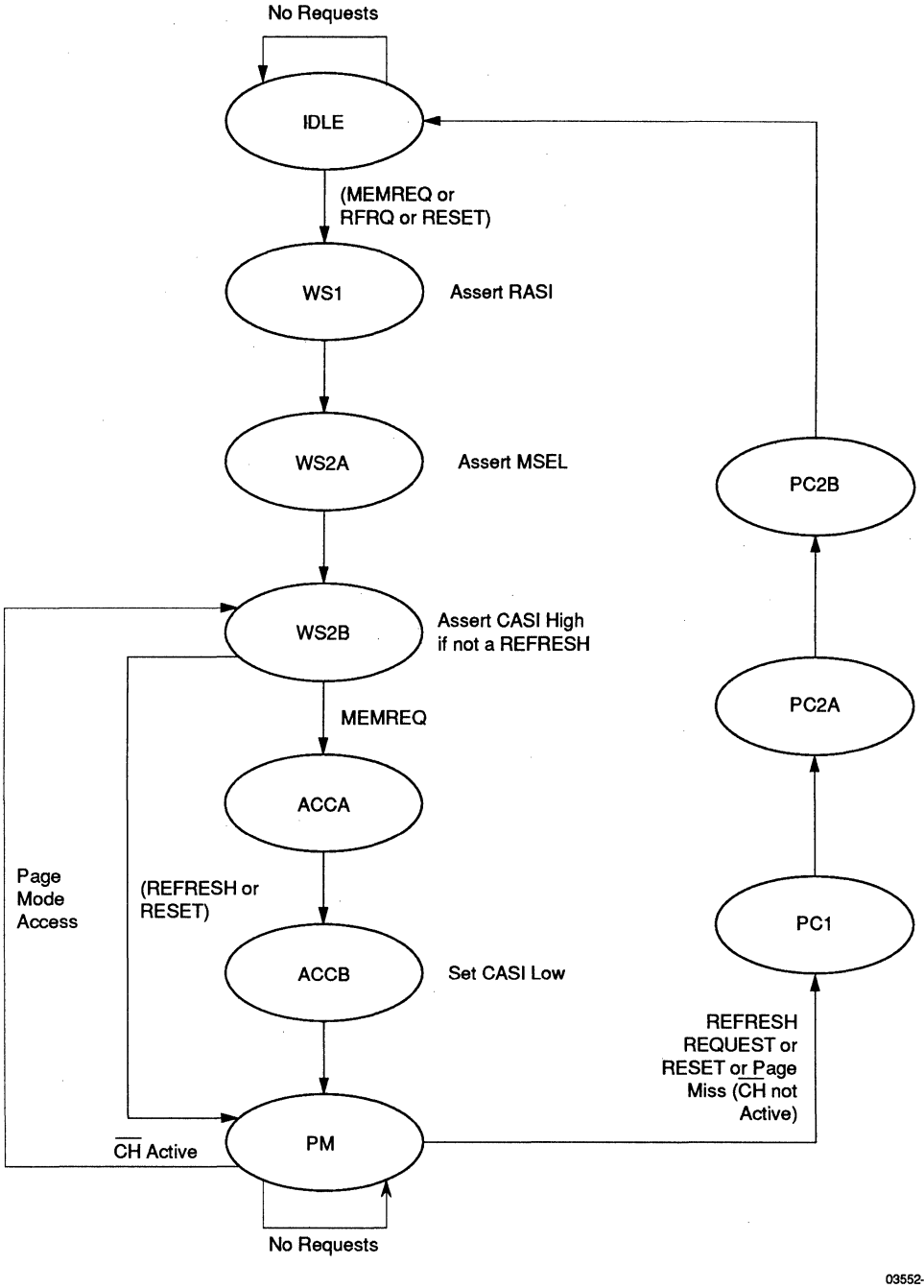


Figure 2. State Diagram

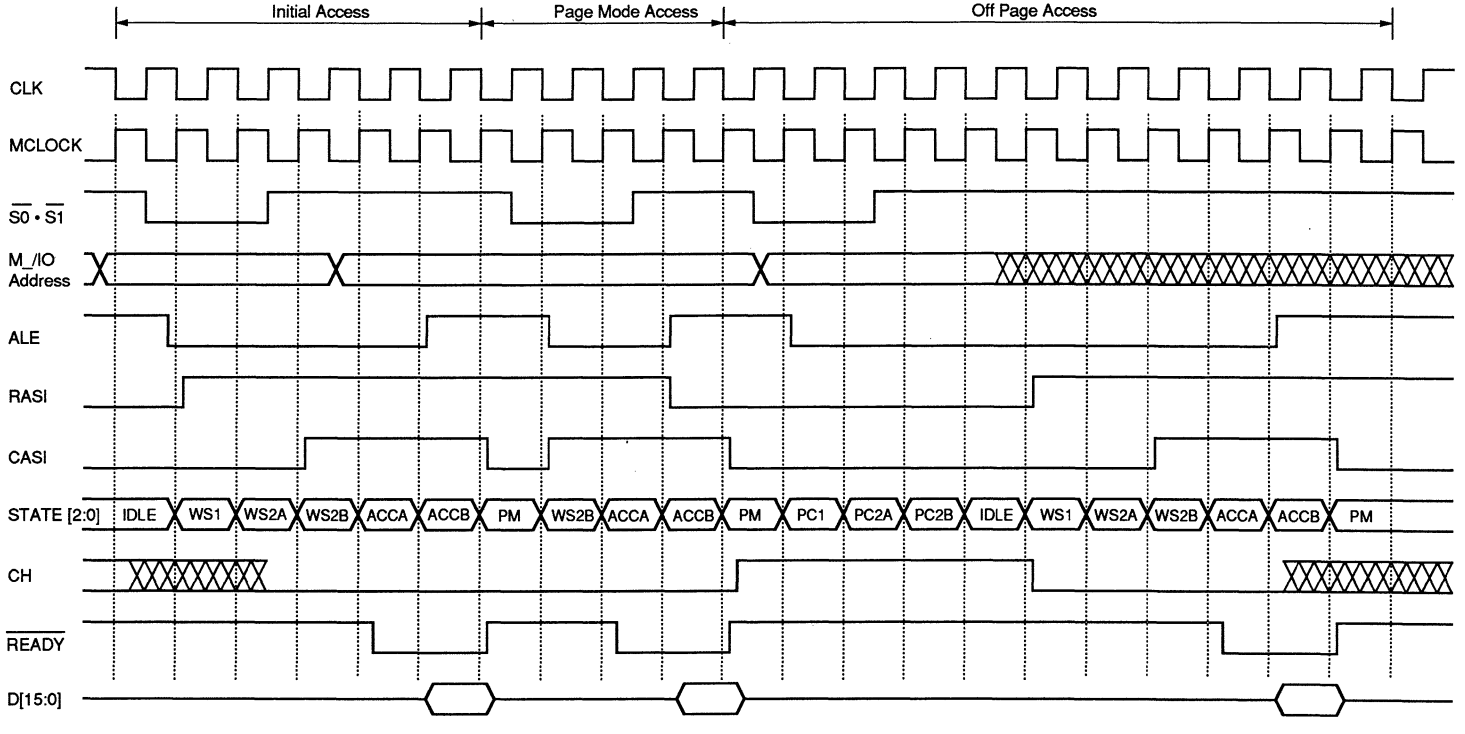


Figure 3. Timing Diagram

Sometimes, a memory access is requested during a refresh cycle. Since the refresh interval is controlled by the 555 timer, the refresh is asynchronous to the 80C286 memory requests. Under certain conditions, the refresh may start at the beginning of a bus cycle; all memory requests occur at the end of the bus cycle. If the 80C286 requests a memory access during this off-cycle refresh, the memory and the processor will be out of synchronization. This causes the memory controller to assert $\overline{\text{READY}}$ and data one cycle early and to remove them one cycle early. This situation is avoided by using the $\overline{\text{DELAY}}$ output to re-synchronize the memory and processor. If the memory request occurs during the wrong state, $\overline{\text{DELAY}}$ is asserted and the state machine delays one cycle. This re-synchronizes the state machine and the processor. The timing for re-synchronization is shown in Figure 4.

The ALE output from the State PAL is connected to the ALE input of the Am29C668; this signal controls the address latch of the CDMC. When ALE is High, the address latch is transparent and the address is latched on the falling edge of ALE. ALE goes Low when there is a valid memory request, and remains Low until the memory state machine is in the ACCA state, signalling that the memory access is about to complete. Asserting ALE High one cycle before the access completes provides enough time for the next address to be compared with the current address. This insures that $\overline{\text{CH}}$ is valid when it is evaluated by the state machine.

Refresh Cycles

To retain data, dynamic memories must be refreshed periodically to restore the charge on their storage capacitors. For 1-Mbit DRAMs, all 512 rows of memory must be refreshed every 8 ms. There are different methods to perform the refresh cycles: burst, forced and hidden; each has its advantages and disadvantages. The best method is determined by the instruction mix, system hardware and performance requirements.

The first method is burst-refresh cycles that refresh all 512 rows in one sequence. It works well in systems where there are long idle times between memory accesses. The main disadvantage of this system is an access to memory may be delayed for long periods during the refresh cycles, greatly impacting system throughput. This would definitely not be an acceptable method for real-time systems.

Another method is to periodically insert refresh cycles; this is called forced or distributed refresh. If refreshes are interspersed between memory accesses, the memory throughput and access time is not greatly impacted because there is a lower probability of refresh request and memory request contention. One refresh request is generated every 15.6 µs = 8 ms/512 rows. This method is preferable to burst refresh in most systems.

It would be better if the refresh cycles occurred when the processor was accessing other memory or I/O. This type of refreshing is called hidden refresh since all or most of the refresh cycle is overlapped with another access. There are times, however, when the system continually accesses the same memory page and prevents the performance of hidden refreshes. If this happens, a forced refresh cycle must be performed. Hidden refreshing has the lowest system impact since all or most of the refresh cycle is overlapped with an access to another memory or I/O device. There are conceivable situations where hidden refresh would not perform as well as forced refresh. However, for most general applications, hidden refreshing offers the best performance.

This design utilizes forced refreshes instead of hidden refreshes. There are several reasons this method was selected. Additional logic is needed to keep track of when hidden refresh cycles are performed. This logic must suppress the forced refresh request after a hidden refresh cycle is performed and must force a refresh when no hidden refresh is performed. As a result, extra devices must be added that consume more board space, money and power.

A refresh cycle is identical to a normal access, except that the $\overline{\text{CAS}}$ outputs to the DRAMs are suppressed. The Am29C668 suppresses the $\overline{\text{CAS}}$ outputs in the refresh mode. $\overline{\text{REFRESH}}$ is asserted by the 555 timer every 10 µs to insure that the DRAMs maximum $\overline{\text{RAS}}$ active time is not violated. If a memory access is in process, the access is completed before the refresh cycle begins. If both a memory access and memory request occur during the IDLE state, the refresh request is given priority to insure that the refresh requirements of the DRAM are met.

555 Timer

A 555 timer is used to generate the refresh requests. The refresh period T is determined by the values of R1, R2 and C.

$T = 0.693\ [R1 + (2 \times R2)]C,$

When R1 = 15 kΩ, R2 = 4.7 kΩ, and C = 470 pF, T = 7.9 µs.

This meets the required refresh interval if 5% tolerance resistors and a 20% tolerance capacitor are used. Therefore, the maximum refresh overhead is 3.2%.

Normally the refresh interval for forced refresh is 15.8 µs. In this application, the refresh timer performs two functions: it generates the refresh request to maintain the data in the DRAMs and it times the $\overline{\text{RAS}}$ Low signal to

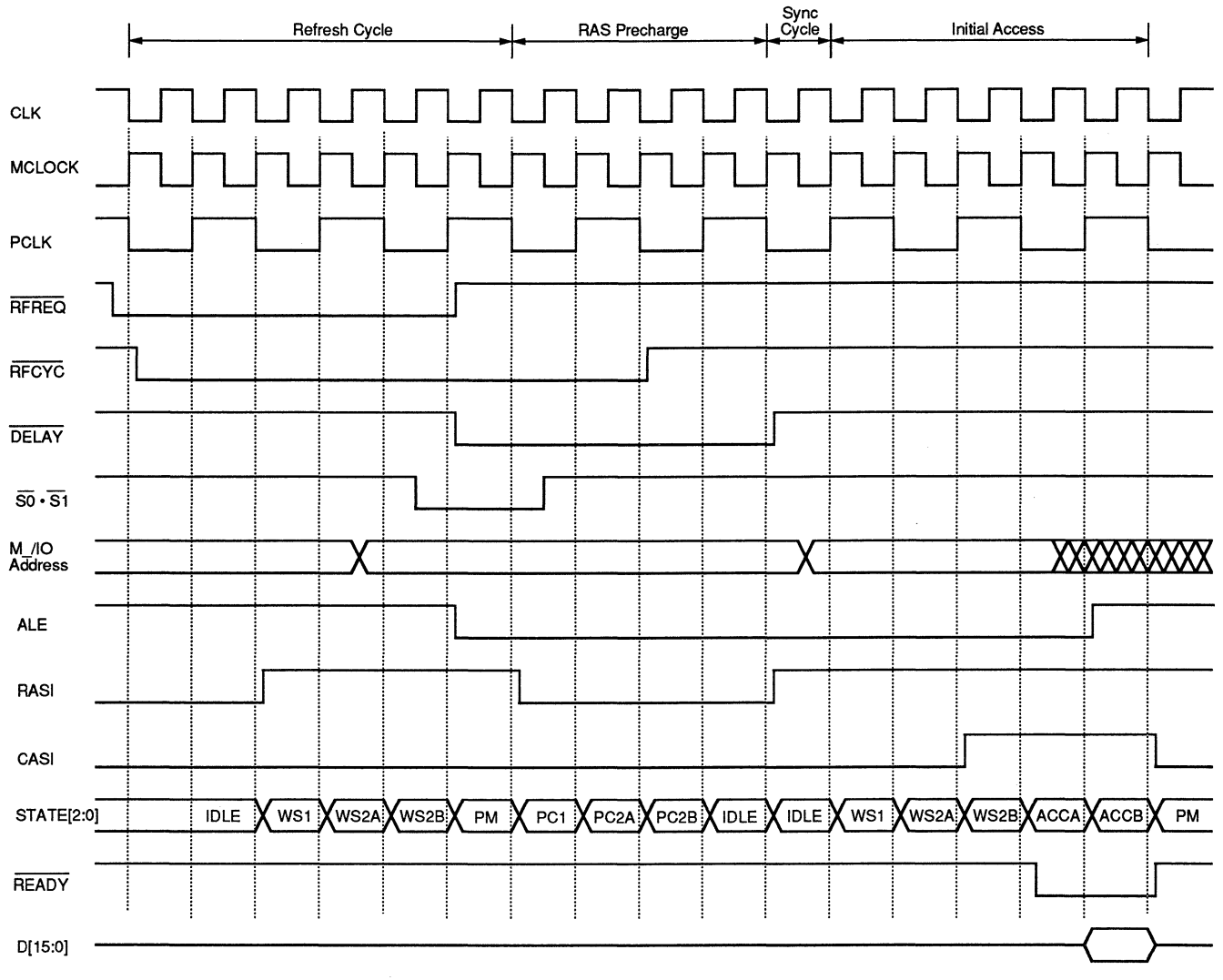


Figure 4. Refresh Timing Diagram with Re-synchronization

guarantee that the maximum $\overline{RAS}$ Low time is not exceeded. The DRAMs have two specified maximum $\overline{RAS}$ Low times: t_{RAS} (10 µs) the maximum $\overline{RAS}$ Low time during a normal access and t_{RASP} (100 µs) the maximum $\overline{RAS}$ Low time during fast-page-mode accesses. If the system can guarantee that every initial access is always followed by a fast-page-mode access, the refresh-timer interval can be 15.8 µs. In most systems, however, this cannot be guaranteed. Therefore, the refresh timer must interrupt the memory every 10 µs to guarantee that the DRAM t_{RAS} is not violated.

The 555 timer could be replaced with an external counter and logic to generate the refresh request. The counter could be synchronized to the processor so that all refresh requests terminate in the correct relationship to the processor clock. Another advantage of the counter implementation is that the exact refresh interval can be more precisely controlled, reducing the refresh overhead to a maximum of 2.5%. The major drawback is the increased cost and board space of the counter and logic over that of the 555 timer.

Data Buffers

Simple data buffers are used to minimize the propagation delay. Since the DRAMs have separate inputs and outputs, two 74F244s are used on the data input lines and two on the data output lines. The 74F244s on the data inputs are permanently enabled by tying their $\overline{OE}$ inputs Low. The $\overline{OE}$ inputs to the data output buffers are generated by the State PAL output $\overline{OEB}$.

Timing Analysis

There are two different DRAM parameters that must be examined to determine the access time. The first is $\overline{RAS}$-to-$\overline{CAS}$ delay t_{RCD}. If t_{RCD} is greater than the specified t_{RCD} (max), the access is controlled by $\overline{CAS}$ access time t_{CAC}. The second parameter is $\overline{RAS}$-to-column-address delay t_{RAD}. If t_{RAD} is greater than t_{RAD} (max), the access is controlled by the access time from column address t_{AA}. If both t_{RCD} and t_{RAD} are less than their specified maximums, the access time is determined by $\overline{RAS}$, t_{RAC}. If both t_{RCD} and t_{RAD} are greater than their maximums, the one yielding the slowest access time determines the memory access time. The hold time of the 80C286 does not affect timing shown here because it is hidden by the turn-off time of the data driver.

In this design, MSEL is asserted one memory-clock cycle after RASI, and CASI is asserted two memory clock cycles after RASI (see Figure 5). The capacitive load on $\overline{RAS}_n$ is approximately

7 pF/DRAM x 16 DRAMs = 112 pF.

For margin, assume 120 pF. The delay from RASI to $\overline{RAS}_n$ is therefore,

26 ns - (350 - 120) pF x 2.5 ns/50 pF = 14.5 ns or approximately 15 ns.

The loading and delay for $\overline{CAS}_n$ are the same; therefore, t_{RCD} is two memory-clock cycles or 50 ns. The load on the Q_n outputs is

5 pF/DRAM x 16 DRAMS/bank x 4 banks = 320 pF.

Assuming 350 pF for margin, the delay from MSEL to Q_n is 26 ns. Therefore,

t_{RAD} = 1 MCLOCK cycle + MSEL-to-Q_n delay - RASI- to-$\overline{RAS}_n$ delay = 25 + 26 - 15 = 36 ns.

Since t_{RAD} (max) = 40 ns and t_{RCD} (max) = 60 ns, the access timing is determined by t_{RAC}. The access requires:

1 MCLOCK Cycle	25 ns
74ALS04 Inverter Delay	5 ns
MCLOCK to RASI	8 ns
RASI to $\overline{RAS}_n$	15 ns
DRAM Access	85 ns
Buffer Delay	7 ns
Data Setup	3 ns
Total Access Time	148 ns

With a total access time of 148 ns, the access is completed in three processor cycles (150 ns). Therefore, the State PAL must be a PAL16R8-10. MSEL is asserted on the cycle after RASI; this insures that the row-address hold time is met. CASI asserted one cycle after MSEL guarantees that the column-address set up time is met.

Consecutive accesses within the same page are limited by the $\overline{CAS}$ access time:

1 MCLOCK Cycle	25 ns
74ALS04 Inverter	5 ns
MCLOCK to CASI	8 ns
CASI to $\overline{CAS}_n$	15 ns
DRAM Page Mode Access	30 ns
Buffer Delay	7 ns
Data Setup	3 ns
Total Access Time	93 ns

With a total access time of 93 ns, the access is completed in two cycles with a margin of (2 x 50) - 93 = 7 ns minimum. Table 1 summarizes the number of processor cycles required for the initial memory access for different DRAM and processor speeds. Table 2 gives the number of processor cycles required for fast-page-mode accesses for different DRAM and processor speeds.

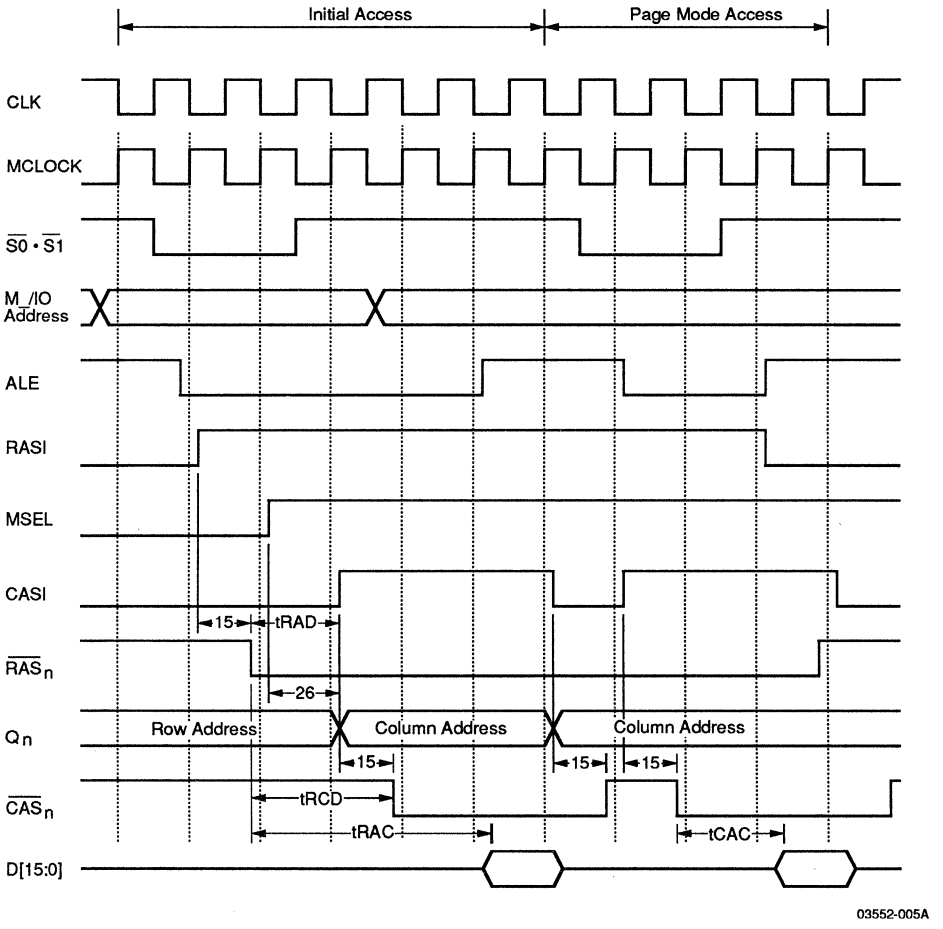


Figure 5. Timing Analyses

Table 1. Processor Cycles for Initial Access for Different Processor and Memory Speeds

Processor Speed (MHz)	Memory Speed (ns) 60	85	100	120
16	3	3	3	3
20	3	3	4	4
25*	3	4	4	5

* Assumes State PAL is PAL16R8-7.

Table 2. Processor Cycles for Page-Mode Access for Different Processor and Memory Speeds

Processor Speed (MHz)	Memory Speed (ns)* 60	85	100	120
16	2	2	2	2
20	2	2	2	2
25**	2	2	2	2

* 2-Cycle access requires zero wait state.
** Assumes State PAL is PAL16R8-7.

PAL EQUATIONS

```
"Am29C668 to AMD 80C286 Interface                    April 10, 1989"

DEVICE      INTERFACE (P20R4)

PIN   MCLOCK = 1 (CLOCK)
      /STATE[3:0] = 2:5 (INPUT COMBINATORIAL)
      RESET = 6 (INPUT COMBINATORIAL)
      /REFRESH = 7 (INPUT COMBINATORIAL)
      ALE = 8 (INPUT COMBINATORIAL)
      A[0]= 9 (INPUT COMBINATORIAL)
      /BHE = 10 (INPUT COMBINATORIAL)
      /S0 = 11 (INPUT COMBINATORIAL)
      /OE = 13 (CONTROL)

      /READY = 22 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /RFDONE = 21 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /RFINT = 20 (OUTPUT ACTIVE_LOW REGISTERED)
      /RFRQ = 19 (OUTPUT ACTIVE_LOW REGISTERED)
      /RFCYC = 18 (OUTPUT ACTIVE_LOW REGISTERED)
      /OEB = 17 (OUTPUT ACTIVE_LOW REGISTERED)
      /WEL = 16 (OUTPUT ACTIVE_LOW COMBINATORIAL)
      /WEH = 15 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE  IDLE_ST =  STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      WS1_ST  =  STATE[3] *  STATE[2] *  STATE[1] *  STATE[0],
      WS2A_ST =  STATE[3] *  STATE[2] * /STATE[1] *  STATE[0],
      WS2B_ST =  STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      ACCA_ST =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      ACCB_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      PM_ST   = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] *  STATE[1] *  STATE[0],
      PC2A_ST = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      PC2B_ST =  STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      IDLE = #b1110,    WS1  = #b1111,    WS2A = #b1101,    WS2B = #b1001,
      ACCA = #b1000,    ACCB = #b0000,    PM   = #b0001,    PC1  = #b0011,
      PC2A = #b0010,    PC2B = #b1010;

BEGIN
ENABLE (WEL,WEH,RFDONE);

ENABLE (READY) = (ACCA_ST + ACCB_ST) * /RFCYC * /RESET;
READY = 1;

"These two signals synchronize the refresh request from the 555 timer."

RFINT := REFRESH * /RFDONE + RESET;
RFRQ := RFINT + RFRQ * /(PM_ST * RFCYC);

RFCYC := IDLE_ST * RFRQ * /RFDONE * /RESET +
      PC2B_ST * RFRQ * /RFDONE * /RESET +
      RFCYC * (WS1_ST + WS2A_ST + WS2B_ST + PM_ST);
```

```
RFDONE := RFCYC * WS1_ST + RFDONE * REFRESH * /RESET;

WEL = /A[0] * ALE * S0 + WEL * /ALE * /RESET;
WEH = BHE * ALE * S0 + WEH * /ALE * /RESET;

OEB := /WEH * /WEL * /RFCYC * /RESET * (WS2A_ST + WS2B_ST + ACCA_ST);

END.

TEST_VECTORS
IN    MCLOCK,STATE[3:0],RESET,REFRESH,ALE,A[0],BHE,S0,OE;
OUT   READY,RFDONE,RFINT,RFRQ,RFCYC,OEB,WEL,WEH;
BEGIN
C 1 1 1 0 0 0 1 0 0 0 1    Z L L L L L L L;
C 1 1 1 1 0 0 1 1 0 0 0 1    Z L H L L L L L;  "Refresh Cycle"
C 1 1 1 1 0 0 1 1 0 0 0 1    Z L H H L L L L;
C 1 1 1 1 0 0 1 1 0 0 0 1    Z L H H H L L L;
C 1 1 1 1 1 0 1 1 0 0 0 1    Z H H H H L L L;
C 1 1 0 1 0 1 1 1 0 0 0 1    Z H L H H L L L;
C 1 0 0 1 0 1 1 1 0 1 1 1    Z H L H H L H H;
C 0 0 0 1 0 1 0 0 1 1 1 1    Z H L L H L H H;  "Write Cycle"
C 0 0 1 1 0 1 0 0 1 1 1 1    Z H L L L L H H;
C 0 0 1 0 0 1 0 0 1 1 1 1    Z H L L L L H H;
C 1 0 1 0 0 1 0 0 1 1 1 1    Z H L L L L H H;
C 1 1 1 1 0 0 1 0 0 1 1 1    Z H L L L L H H;
C 1 1 1 1 1 0 1 0 0 1 1 1    Z H L L L L H H;
C 1 1 0 1 0 0 0 0 0 1 1 1    Z L L L L L H H;
C 1 0 0 1 0 0 0 0 0 0 0 1    Z L L L L L H H;
C 1 0 0 0 0 0 0 0 1 0 0 1    H L L L L L H H;
C 0 0 0 0 0 0 0 1 1 0 0 1    H L L L L L L L;
C 0 0 0 1 0 0 1 1 0 0 1    Z L L L L L L L;  "Read Cycle"
C 1 0 0 1 0 0 0 0 1 0 0 1    Z L L L L H L L;
C 1 0 0 0 0 0 0 0 1 0 0 1    H L L L L H L L;
C 0 0 0 0 0 0 0 0 0 0 0 1    H L L L L L L L;
C 0 0 0 0 1 0 0 0 0 0 0 1    Z L L L L L L L;
END.
```

```
"Am29C668 to AMD 80C286 20 MHz Interface.
APRIL 19, 1989"

DEVICE      STATE (P16R8)

PIN   MCLOCK = 1 (CLOCK)
      /S0 = 2 (INPUT COMBINATORIAL)
      /S1 = 3 (INPUT COMBINATORIAL)
      M_IOL = 4 (INPUT COMBINATORIAL)
      /CH = 5 (INPUT COMBINATORIAL)
      A[23] = 6 (INPUT COMBINATORIAL)
      /RFRQ = 7 (INPUT COMBINATORIAL)
      /RFCYC = 8 (INPUT COMBINATORIAL)
      RESET = 9 (INPUT COMBINATORIAL)
      /OE = 11 (CONTROL)

      /STATE[3:0] = 19:16 (OUTPUT ACTIVE_LOW REGISTERED)
      /RASI = 15 (OUTPUT ACTIVE_LOW REGISTERED)
      CASI = 14 (OUTPUT ACTIVE_LOW REGISTERED)
      /ALE = 13 (OUTPUT ACTIVE_LOW REGISTERED)
      /DELAY = 12 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE  READ = /A[23] * M_IOL * S1,
      WRITE = /A[23] * M_IOL * S0,
      MEMREQ = READ + WRITE,
      IDLE_ST =  STATE[3] *  STATE[2] *  STATE[1] * /STATE[0],
      WS1_ST  =  STATE[3] *  STATE[2] *  STATE[1] *  STATE[0],
      WS2A_ST =  STATE[3] *  STATE[2] * /STATE[1] *  STATE[0],
      WS2B_ST =  STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      ACCA_ST =  STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      ACCB_ST = /STATE[3] * /STATE[2] * /STATE[1] * /STATE[0],
      PM_ST   = /STATE[3] * /STATE[2] * /STATE[1] *  STATE[0],
      PC1_ST  = /STATE[3] * /STATE[2] *  STATE[1] *  STATE[0],
      PC2A_ST = /STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      PC2B_ST =  STATE[3] * /STATE[2] *  STATE[1] * /STATE[0],
      IDLE = #b1110,    WS1  = #b1111,    WS2A = #b1101,    WS2B = #b1001,
      ACCA = #b1000,    ACCB = #b0000,    PM   = #b0001,    PC1  = #b0011,
      pc2a = #b0010,    PC2B = #b1010,    UNUSED1 = #b1100, UNUSED2 = #b0100,
      UNUSED3 = #b0101, UNUSED4 = #b0111, UNUSED5 = #b0110, UNUSED6 = #b1011;

BEGIN

ALE := MEMREQ +   ALE * /(ACCA_ST * /RFCYC);

DELAY := MEMREQ * /ALE * (IDLE_ST * RFRQ * /RFCYC +
                  STATE[1] * STATE[0] + STATE[3] * /STATE[2]) +
      DELAY * /(STATE[2] * /RFCYC);

CASE (STATE[3:0]) BEGIN

IDLE) IF (ALE * /DELAY + MEMREQ * /RFRQ + RFCYC + RFRQ * RESET) THEN
STATE[3:0] := WS1;
      ELSE STATE[3:0] := IDLE;
```

```
WS1)   STATE[3:0] := WS2A;

WS2A)  STATE[3:0] := WS2B;

WS2B)  IF (RFRQ) then STATE[3:0] := PM;
       ELSE STATE[3:0] := ACCA;

ACCA)  STATE[3:0] := ACCB;

ACCB)  STATE[3:0] := PM;

PM)    IF (RFCYC + RFRQ + /CH * MEMREQ) THEN STATE[3:0] := PC1;
       ELSE
             IF (CH * MEMREQ) THEN STATE[3:0] := WS2B;
             ELSE STATE[3:0] := PM;

PC1)   STATE[3:0] := PC2A;

PC2A)  STATE[3:0] := PC2B;

PC2B)  STATE[3:0] := IDLE;

UNUSED1,UNUSED2,UNUSED3,UNUSED4,UNUSED5,UNUSED6)  STATE[3:0] := IDLE;

END "CASE";

/CASI := WS2A_ST * /RFRQ + CASI * /ACCB_ST;

RASI := IDLE_ST * /(ALE + MEMREQ * /RFRQ + RFCYC + RFRQ * RESET) +
       PM_ST * (/CH * MEMREQ + RFRQ) +
       /STATE[2] * STATE[1];

END.

TEST_VECTORS
IN MCLOCK,S[1:0],M_IOL,A[23],RFRQ,RFCYC,RESET,CH,OE;
OUT STATE[3:0],RASI,CASI,ALE,DELAY;
BEGIN
0 0 0 0 1 0 0 0 0 1       L L L L L H L L;
C 0 0 0 1 0 0 0 0 1       L L L H L L L L;
C 0 0 0 1 0 0 0 0 1       L L L H L L L L;
C 0 0 0 1 0 0 0 1 0 1     L L L H L L L L;  "SYSTEM RESET"
C 0 0 0 1 0 0 1 0 1       L L L H L L L L;
C 0 0 0 1 1 0 1 0 1       L L H H H L L L;
C 0 0 0 1 1 0 1 0 1       L L H L H L L L;
C 0 0 0 1 1 0 1 0 1       H L H L H L L L;
C 0 0 0 1 1 0 1 0 1       H H H L H L L L;
C 0 0 0 1 1 0 1 0 1       H H H H L L L L;
C 0 0 0 1 1 0 0 0 1       H H L H L L L L;
C 0 0 0 1 1 0 0 0 1       H L L H L L L L;
C 0 0 0 1 1 0 0 0 1       L L L H L L L L;
C 0 0 0 1 1 0 0 0 1       L L H H H L L L;
C 0 0 0 1 0 0 0 0 1       L L H L H L L L;
C 0 0 0 1 0 0 0 0 1       H L H L H L L L;
C 0 0 0 1 0 0 0 0 1       H H H L H L L L;  "IDLE"
```

```
C 0 0 0 0 1 0 0 0 1       H H H L H L L L;  "REFRESH REQUEST"
C 0 0 0 0 1 1 0 0 1       H H H H L L L L;  "START REFRESH CYCLE"
C 0 0 0 0 1 1 0 0 1       H H L H L L L L;
C 0 0 0 0 1 1 0 0 1       H L L H L L L L;
C 1 0 1 0 1 1 0 0 1       L L L H L L H H;  "MEMORY REQUEST"
C 1 0 1 0 1 1 0 0 1       L L H H H L H H;
C 0 0 0 0 0 1 0 0 1       L L H L H L H H;
C 0 0 0 0 0 0 0 0 1       H L H L H L H H;
C 0 0 0 0 0 0 0 0 1       H H H L H L H H;  "WAIT ONE CYCLE TO SYNCHRONIZE"
C 0 0 0 0 0 0 0 0 1       H H H L L L H L;  "MEMORY ACCESS"
C 0 0 0 0 0 0 0 0 1       H H H H L L H L;
C 0 0 0 0 0 0 0 0 1       H H L H L L H L;
C 0 0 0 0 0 0 0 0 1       H L L H L H H L;
C 0 0 0 0 0 0 0 0 1       H L L L L H H L;
C 0 0 0 0 0 0 0 0 1       L L L L L H L L;
C 0 0 0 0 0 0 0 0 1       L L L H L L L L;  "PAGE MODE WAIT"
END.
```

Am29C668 Configurable Dynamic Memory Controller to 80386 Microprocessor Interface

by Douglas Lee, Applications Specialist

INTRODUCTION

The interface between the Am29C668 4-Mbit Configurable Dynamic Memory Controller (CDMC) and the Intel 80386 microprocessor was designed for maximum performance; therefore, lowering the total device count was a secondary concern. It is possible to interface the Am29C668 to the 80386 with only one PAL®, but this would make many assumptions about the system implementation which are not generally applicable. This design is as general as possible so that the user may tailor his implementation to a specific memory system. Possible changes to the design are discussed with associated system requirements and implications. A block diagram, timing analyses and logic equations necessary to implement the design are included. This design requires a minimum number of external devices to perform the interface and glue functions: three PAL devices (one 20L8, one 22V10 and one 20X10A), one 74LS240 (six inverters) and four 74F245 bidirectional transceivers.

Distinctive Characteristics

- Am29C668 4-Mbit Configurable Dynamic Memory Controller/Driver with Auto Timing
- 16-MHz 80386 Microprocessor
- 120-ns Fast Page Mode 1 Mbit x 1 DRAM
- Two Wait-State Initial Accesses With Zero Wait-State Subsequent Page-Mode Accesses
- 4-Mbyte Dynamic Memory Expandable Up To 16 Mbyte per Am29C668
- Supports Up To 4 Gbytes of Physical Memory
- Supports Address Pipelining
- Performs Hidden Refresh Cycles to Maximize Memory Throughput

MEMORY ARCHITECTURE OVERVIEW

To obtain the maximum memory throughput but still maintain a reasonable cost, 120-ns fast page-mode DRAMs are used. The 80386 requires a minimum of two processor cycles per access. If additional cycles are needed, the memory controller holds $\overline{RDY}$ inactive. The processor inserts wait states until $\overline{RDY}$ is asserted. The 16-MHz 80386 completes the initial access to memory in two wait states (four cycles total). The subsequent accesses within the page are performed with no wait states (two cycles total).

Page-mode DRAMs appear to the processor as if they are fast cache memories during page accesses. The page size for a 1-Mbit DRAM is 1024 bits or 1 Kbits. The memory is 32 bits wide, therefore the page size is 4 Kbytes. The Am29C668 detects accesses within the same page via on-chip cache-mode operation. When a new address is latched, it is compared with the previous address; if the row and bank addresses are the same (the upper 12 bits) then the Cache Hit signal $\overline{CH}$ is asserted. The memory state machine immediately begins the next access without deasserting $\overline{RAS}$. An access outside the page results in a five-cycle, three-wait-state access, two cycles for the $\overline{RAS}$ precharge and three for the data access. The actual performance enhancement of the memory depends upon the instruction mix of the program executed.

The memory array consists of four banks; each bank contains 4 Mbytes or 1 Mword (32 bits) of memory. This gives a maximum size of 16 Mbytes or 4 Mwords of memory.

A 16-MHz system was selected since high performance may be achieved even with relatively slow DRAMs. As processor speeds increase toward 20 and 25 MHz, the number of wait states increases until this memory architecture becomes impractical. Faster microprocessors demand faster DRAMs, static RAM caches, bank interleaving, or a combination of three or more other exotic architectures, implemented at considerable cost. These topics are beyond the scope of this application note.

FUNCTIONAL DESCRIPTION

The primary data paths and functional elements are shown in Figure 1. The following discussion describes each subsection of the block diagram, including the control logic, buffers and memory array.

Am29C668 CDMC

The Am29C668 generates the $\overline{RAS}$, $\overline{CAS}$ and address signals to the DRAM array; no external drivers are needed. Additionally, the Am29C668 generates the row addresses during refreshes from its internal refresh row-address counter. The Am29C668, operating in the auto-timing mode, generates the $\overline{RAS}$-to-$\overline{CAS}$ and the $\overline{RAS}$-to-address timing internally.

The Am29C668 must be programmed before any memory accesses can occur. This is accomplished through a dummy I/O access to the I/O address E190H. The con-

trol logic does not distinguish between I/O Reads and Writes to the CDMC configuration register. The input AC[10] must be tied Low to insure proper configuration. The address bits A[11:2] are written into the Am29C668 configuration register. The options selected are: a 4-bank $\overline{RAS}$ and $\overline{CAS}$ configuration, $\overline{CAS}$ byte decoding, $\overline{RAS}$-only refresh, 1-Mbyte memory size, cache mode and auto timing.

The Am29C668 supports byte decoding through the CASEN[3:0] inputs. Byte-enable outputs $\overline{BE}$[3:0] from the 80386 are connected to these inputs and only the selected bytes are accessed. The unselected bytes perform a $\overline{RAS}$-only refresh on the current row.

The auto-timing mode generates the $\overline{RAS}$, $\overline{CAS}$ and multiplexed address inputs to the DRAMs. This timing is optimized for 100-ns DRAMs. The $\overline{CAS}$n outputs are further qualified by the CASIEN input (auto timing with external override). With RASI active, CASIEN is pulsed; this accesses the DRAM in fast-page mode. During these fast-page-mode accesses, the DRAMs are accessed with no wait states inserted. The $\overline{CH}$ signal from the Am29C668 is used to determine if the current address in the input latch has the same row as the previous address. If the fast-page-mode accesses are to the same bank and row, $\overline{CH}$ is asserted and a page mode access is initiated. If $\overline{CH}$ is deasserted, the $\overline{RAS}$ must be precharged and a normal access occurs.

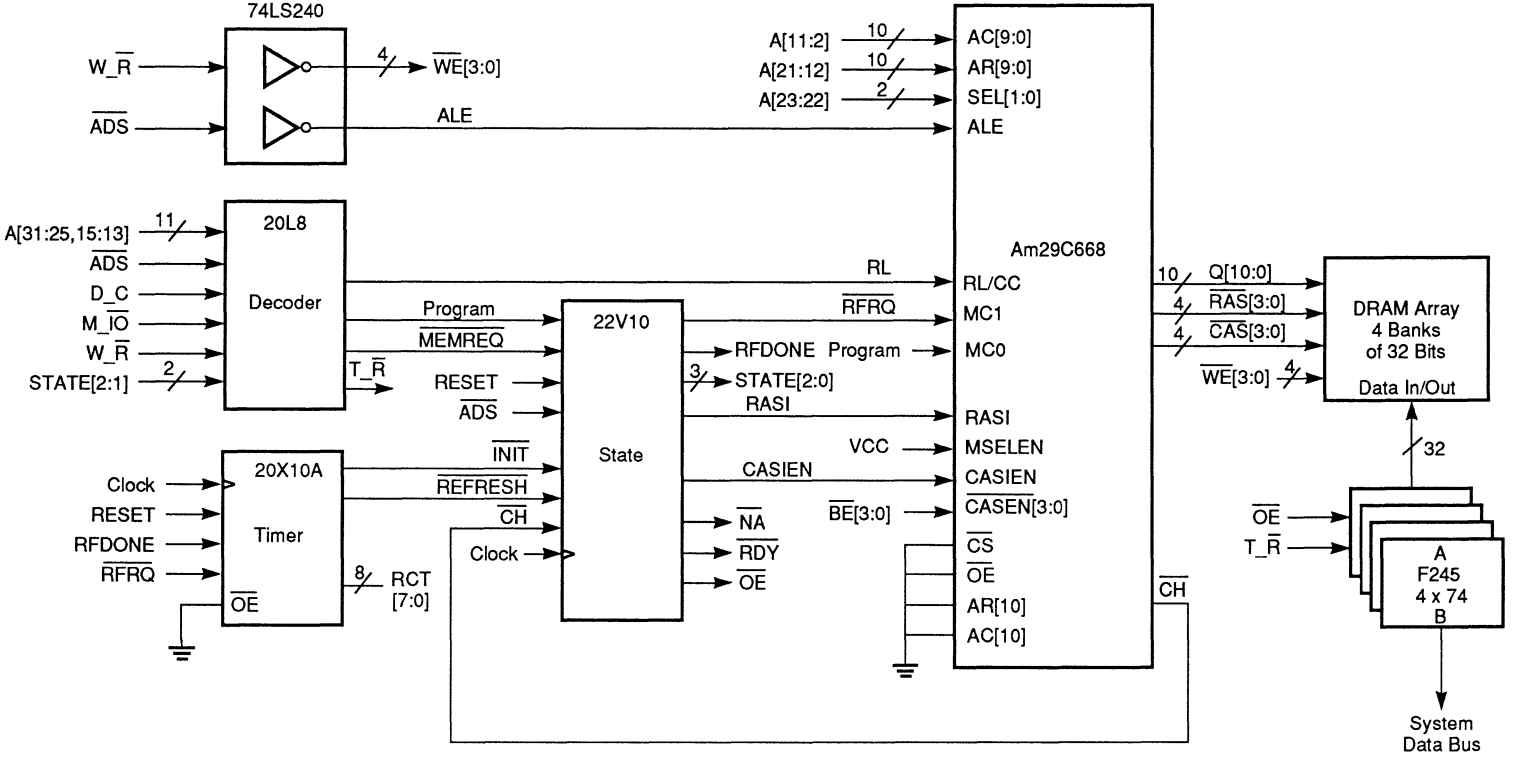


Figure 1. Interface Block Diagram.

11588-001B

Decoder PAL

The Decoder PAL decodes valid memory accesses. The following inputs are used:

A[31:24]	System address, for memory access decoding
A[15:13]	System address, for I/O access decoding
$\overline{ADS}$	Address Status, indicates when address and control signals are valid
M_$\overline{IO}$	Memory active High, I/O active Low
W_$\overline{R}$	Write active High, Read active Low
D_$\overline{C}$	Data active High, Control active Low
STATE[2:1]	Upper two state variables

The following outputs are generated:

$\overline{MEMREQ}$	Request for memory access
RL	Register load signal
PROGRAM	Program cycle
T_$\overline{R}$	Transmit active High, Receive active Low, drives the 74F245

The Decoder PAL decodes the upper bit of the address and M_$\overline{IO}$ to see if the memory is being accessed. If the address and control signals are valid, $\overline{MEMREQ}$ is asserted. The lower address bits A[23:2] are used by the Am29C668 to select the proper bank and memory word. Bits A[11:2] are the column address and are connected to the Am29C668 inputs AC[9:0]; A[21:12] are the row address and are connected to the Am29C668 inputs AR[9:0]. Input AR[10] of the Am29C668 must be tied Low to insure the proper evaluation of $\overline{CH}$. This input is used only with 4-Mbit DRAMs. Bits A[23:22] determine which bank is selected and are connected to SEL[1:0] of the Am29C668. By decoding all the address bits, this design supports a physical memory of up to 4 Gbytes (4,294,967,296 bytes). If a smaller memory space is used, fewer address bits must be decoded and the decoder logic could be combined in one 20-pin PAL, thus saving board space and cost.

The Am29C668 is programmed by providing a dummy output to an I/O address. The upper three bits of the I/O space A[15:13] are decoded along with D_$\overline{C}$ and M_$\overline{IO}$. If the access is valid, PROGRAM is asserted and is input to the Control PAL and to the MC0 input of the Am29C668. The Am29C668 stores the lower 11 bits of the address in the configuration register. By decoding all 16 address bits during I/O operations, this design supports the full 64-Kbyte I/O address space of the 80386.

The Register Load RL signal is used to load the Am29C668 configuration register. It is asserted when the processor executes an I/O to the configuration-register address space. RL is asserted after PROGRAM is asserted and $\overline{ADS}$ is deasserted to insure the correct setup of MC0 relative to RL.

The decoder T_$\overline{R}$ output signal controls the flow of data through each of the four 74F245 bidirectional transceivers. When the signal is High, data is sent from port A to port B; when Low, data is sent from port B to port A. This signal is latched when the memory state machine is in the SW2 or ACC state. In all other states, the latch is transparent and the output is W_$\overline{R}$ inverted. During pipelined cycles, W_$\overline{R}$ can change at least 27 ns before the access is completed. If T_$\overline{R}$ were just W_$\overline{R}$ inverted, the transceiver would change the flow of data too soon and the data would be corrupted. By latching T_$\overline{R}$, data remains valid until the access is completed.

State PAL

The State (Control) PAL arbitrates between accesses, generates the bus control signals to the 80386 and contains the state machine that controls the memory accesses. The following inputs are used:

CLOCK	16-MHz system clock
$\overline{REFRESH}$	Refresh request signal from the Timer PAL
PROGRAM	Program cycle signal from the Decoder PAL
$\overline{MEMREQ}$	Memory cycle request from the Decoder PAL
RESET	System reset signal
$\overline{ADS}$	Address Status; indicates when address and control signals are valid
$\overline{CH}$	Cache Hit; signals access within active page
$\overline{INIT}$	Initialize, resets RFDONE

The following outputs are generated:

$\overline{OE}$	Output enable for the $\overline{RDY}$ and $\overline{NA}$ signals
CASIEN	Column Address Strobe Input Enable
RFDONE	Hidden Refresh performed
$\overline{RDY}$	Ready signal to the 80386, signals termination of valid access
$\overline{NA}$	Next Address to the 80386, enables address pipeline
$\overline{RFRQ}$	Refresh Request, signals refresh cycles to the Am29C668
STATE[2:0]	State variables for the state machine

Figure 2 is the state diagram for the memory controller state machine. Figure 3 gives the timing diagram for accesses to the memory.

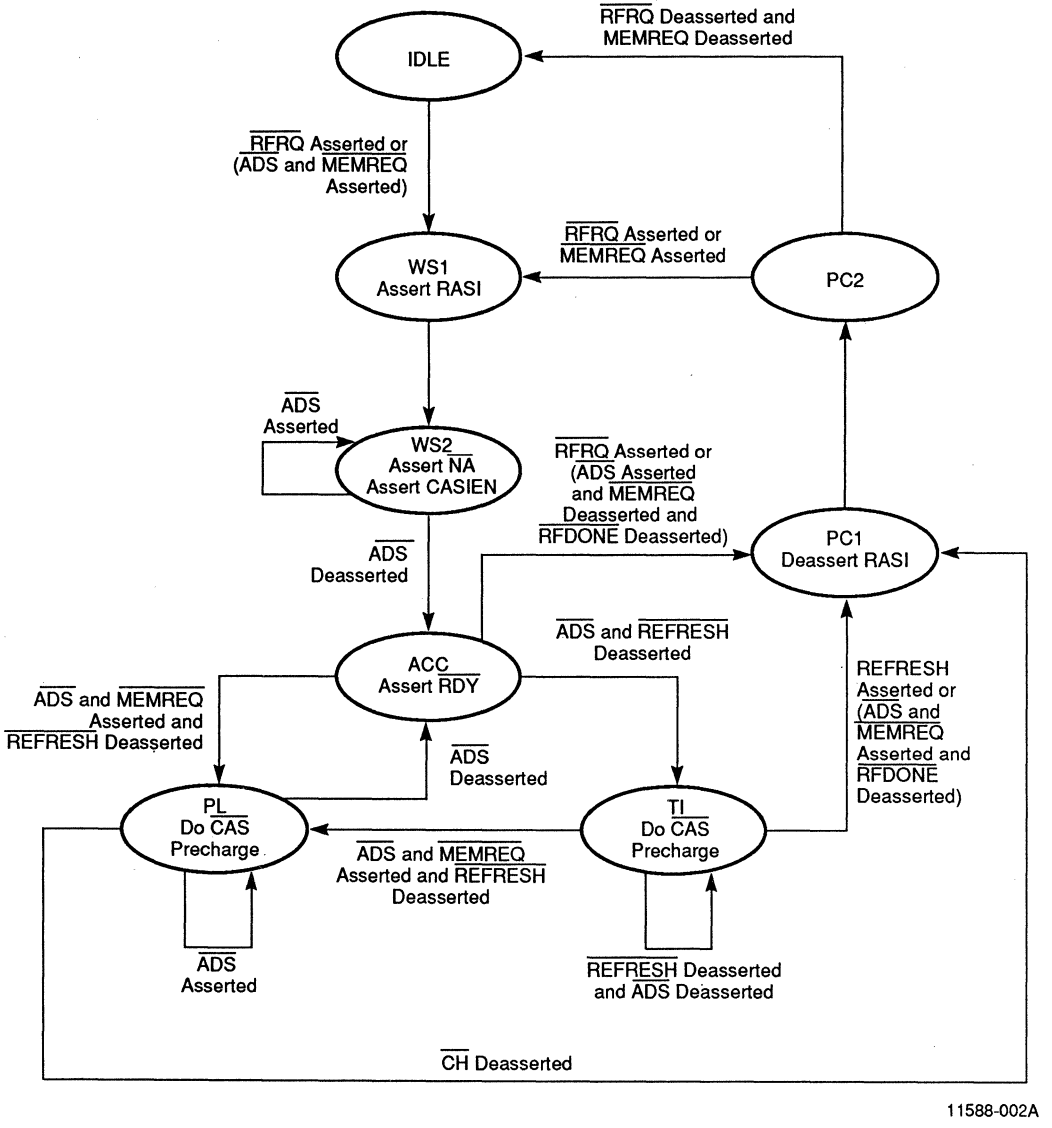
Memory Accesses

The output enable signal $\overline{OE}$ enables the output buffers and is asserted whenever the memory is being accessed or a program cycle is being performed. $\overline{OE}$ ac-

tive with $\overline{ADS}$ deasserted is also used to enable the outputs of $\overline{RDY}$ and $\overline{NA}$ generated by the State PAL. If a refresh cycle is being performed, signaled by $\overline{RFRQ}$ asserted, $\overline{OE}$ is suppressed.

Memory accesses are initiated by asserting $\overline{ADS}$. As a result, the state machine goes to the first wait state WS1and RASI is asserted. In this state, $\overline{CAS}$n is suppressed by deasserting CASIEN so that the column addresses have ample time to become valid. From WS1, the state machine always goes to the second wait state WS2. Here CASIEN is asserted as long as $\overline{ADS}$ is inactive to insure that the write data is set up before $\overline{CAS}$ is asserted. In this state, $\overline{NA}$ is asserted so that the next address and bus control signals can be driven on the bus. When $\overline{ADS}$ is deasserted, the state machine goes to the ACC state. In this state, $\overline{RDY}$ is asserted to signal the 80386 that the access has been completed.

If there is a pending refresh request, it is given priority. The state machine goes to PC1 and PC2 to precharge $\overline{RAS}$ and then executes the refresh cycle. If there is a memory request to a different page, $\overline{RAS}$ is precharged and a normal access performed. Accesses to the same page, signaled by $\overline{CH}$ asserted, causes the state machine to go to PL and then to ACC. If no accesses are pending, signaled by $\overline{MEMREQ}$ deasserted, the state machine enters the TI state. It remains in this state until a refresh request or a valid memory request is received. On a refresh request, $\overline{RAS}$ is precharged and the refresh cycle is performed. If an access to the same page is initiated while the state machine is in TI, the state machine goes to PL. This is done to insure that $\overline{ADS}$ is deasserted, signaling that the previous cycle has completed before the access is initiated. $\overline{ADS}$ is active for multiple cycles, only if $\overline{NA}$ is asserted more than one cycle before $\overline{RDY}$. If $\overline{NA}$ is always asserted one cycle before $\overline{RDY}$, the state machine can go directly to ACC and save one access cycle. If an access is initiated to another page, the $\overline{RAS}$ precharge occurs followed by a normal access.



11588-002A

Figure 2. State Diagram

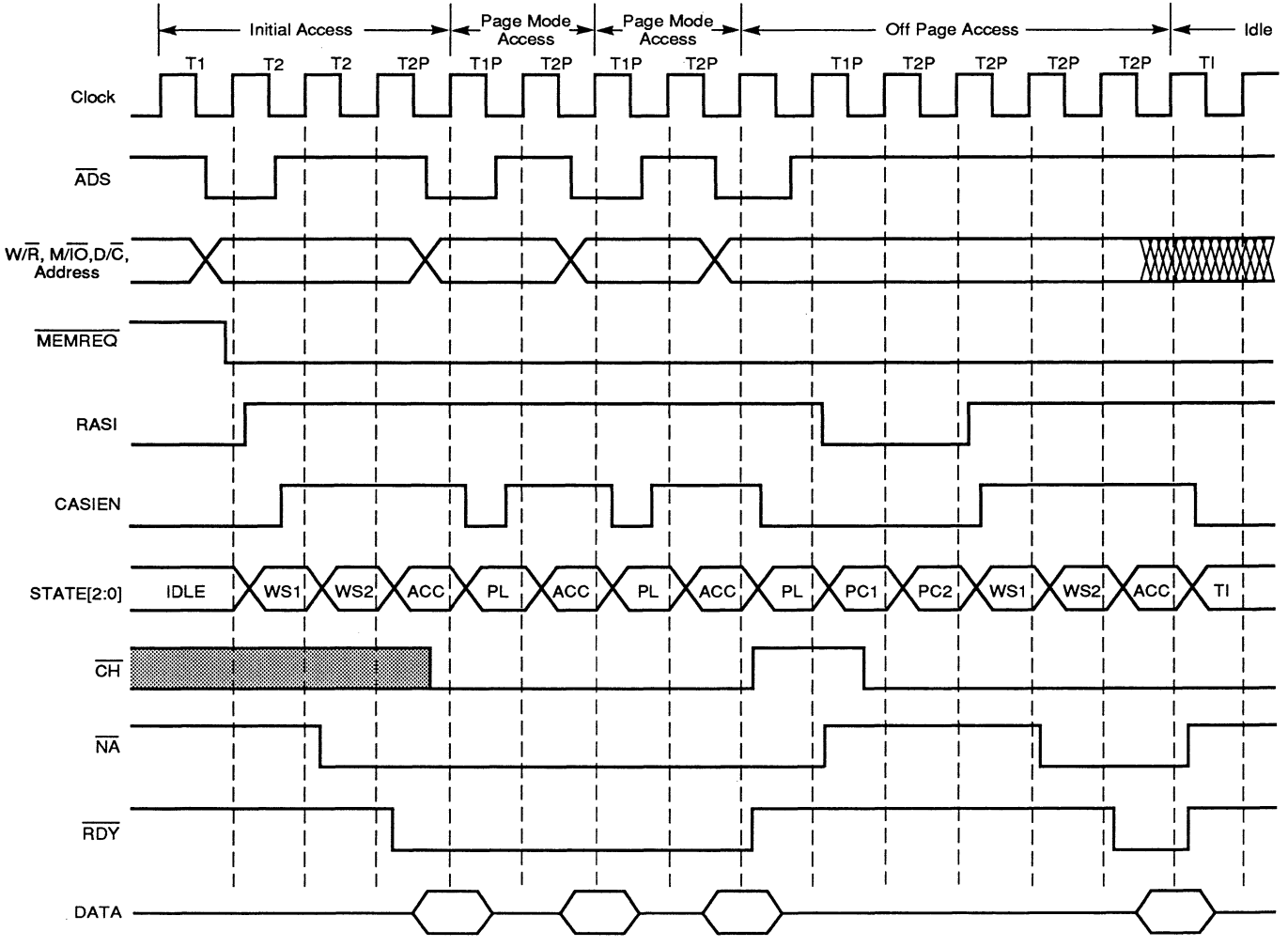


Figure 3. Timing Diagram

11588-003A

Refresh Cycles

Dynamic memories must be refreshed periodically to restore the charge on the storage capacitor to retain the data. For 1-Mbit DRAMs, all 512 rows of memory must be refreshed every 8 ms. There are several different methods to perform refresh cycles: burst, forced and hidden. Each method has its advantages and disadvantages. The best method is determined by the instruction mix, system hardware and performance requirements.

The first method is burst refresh cycles that refresh all 512 rows at once. It works well in systems where there are long idle times between memory accesses. The main disadvantage is that an access to memory may be delayed for long periods during the refresh cycles, greatly impacting system throughput. This would definitely not be an acceptable method for real-time systems.

Another method is to periodically insert refresh cycles; this is called forced refresh. If refreshes are interspersed between memory accesses, the memory throughput and access time is not greatly impacted because there is a lower probability of refresh request and memory request contention. One refresh request is generated every 15.6 µs = 8 ms/512 rows. This method is preferable to burst refresh in most systems.

Even better would be for the refresh cycles to occur when the processor is accessing other memory or I/O. This type of refreshing is called hidden refresh since all or most of the refresh cycle is overlapped with another access. There are times, however, when the system continually accesses the same memory page and prevents hidden refreshes from being performed. If this happens, a forced refresh cycle must be performed. Hidden refreshing has the lowest system impact since all or most of the refresh cycle is overlapped with an access to another memory or I/O device. There are conceivable situations where hidden refresh would not perform as well as forced refresh. However, for most general applications, hidden refreshing offers the best performance.

A refresh cycle is identical to a normal access, except that the $\overline{CAS}$n outputs are suppressed by the Am29C668. The control logic performs two different types of refresh cycles, hidden and forced. A hidden refresh is performed when $\overline{ADS}$ is asserted and MEMREQ is deasserted. RFDONE is set to indicate that a hidden refresh has been performed. When RFDONE is asserted, $\overline{REFRESH}$ is suppressed since no forced refresh is needed. RFDONE is reset by $\overline{INIT}$, ensuring that only one refresh cycle is performed every 15 µs. To guarantee the setup time of MC1 relative to RASI, $\overline{RFRQ}$ is asserted one cycle before the refresh cycle begins.

There are times, however, when the system continually accesses the same memory and prevents hidden refreshes from being performed. If this happens, $\overline{REFRESH}$ is asserted by the Timer PAL every 15 µs. The state machine gives higher priority to refresh accesses than memory accesses. A refresh cycle is performed during the next memory access.

Timer PAL

Timer PAL Inputs:

CLOCK	System 16-MHz clock
RFDONE	Refresh Done; asserted when hidden refresh is performed
RESET	System Reset; initializes counter
$\overline{RFRQ}$	Refresh Request; signals refresh memory access
RCT[7:0]	Counter
$\overline{REFRESH}$	Forced Refresh Request
$\overline{INIT}$	Initialize; resets counter and RFDONE in State PAL

The Timer PAL helps implement the hidden and forced refreshes along with the State PAL. When the State PAL detects an access outside the memory, it generates a refresh request if one has not been performed within the last 15 µs. RFDONE is set to indicate that a hidden refresh cycle has been performed. If no hidden refresh has been performed in 15 µs, the Timer PAL generates a forced refresh by asserting $\overline{REFRESH}$. The State PAL performs a memory refresh cycle on the next available memory access. The Timer PAL asserts $\overline{INIT}$ for one clock cycle every 15 µs to reset RFDONE and reinitialize the counter.

The timing for the Timer PAL is selected by the value initialized in the counter. This value is set to 247 resulting in a refresh request cycle time of 15.3 µs. The initial value is set by changing the terms 0 * INIT or 1 * INIT. $\overline{REFRESH}$ is asserted 11 cycles before $\overline{INIT}$ when no hidden refresh has been performed. This timing may be changed by altering the value of START_REFRESH.

There are several alternate methods to implement the refresh timer. A 555 timer could be used to save board space and cut cost. The problem with this solution is the need for asynchronous arbitration. This requires additional logic. Another alternative is to use a spare DMA channel to implement the refresh requests similar to the IBM PC-AT and PS/2* systems. Memory throughput may not be as high as the current design because the bus is unavailable during refresh accesses; whereas with hidden refreshes, accesses outside the memory spaces occur concurrently.

*PC-AT and PS/2 are registered trademarks of IBM Corporation.

Data Buffers

Simple data buffers are used to minimize the propagation delay. The $\overline{OE}$ is generated by the State PAL and T_$\overline{R}$ is generated by the Decoder PAL. Four 74F245s are used in this design since 32 bits are used.

Timing Analysis

The initial access to memory requires four processor cycles to complete. Each processor cycle is a minimum of 62 ns at 16 MHz. The hold time of the 80C386 does not affect timing shown here because it is hidden by the turn-off time of the data driver. The initial access requires:

T1 Cycle	62 ns
Clock to RASI	12 ns
RASI to $\overline{RAS}$n	26 ns
DRAM Access	120 ns
Buffer Delay	7 ns
Data Setup	10 ns
Total Access Time	237 ns

With a total access time of 237 ns, the access is completed in four processor cycles with a (4 x 62) - 237 = 11 ns margin. The second consecutive access requires:

½ T1 Cycle Precharge	31 ns
Clock Low to CASIEN	15 ns
CASIEN to $\overline{CAS}$n	26 ns
DRAM Page Mode Access	30 ns
Buffer Delay	7 ns
Data Setup	10 ns
Total Access Time	119 ns

With a total access time of 119 ns, the access is completed in two cycles with a margin of (2 x 62) - 119 = 5 ns. Non-consecutive accesses to the same page (an idle bus state between memory accesses) results in an extra wait state inserted. This wait state is necessary to insure that $\overline{ADS}$ is deasserted, signaling a valid memory access, before the access is completed. If $\overline{NA}$ is always asserted one cycle before $\overline{RDY}$, the access may be completed without the wait state because $\overline{ADS}$ will always be deasserted before the completion of the access.

The $\overline{RAS}$ precharge time for 120-ns DRAMs is 90 ns, therefore two processor cycles (62 x 2 = 124 ns) are sufficient.

ALE is always valid 13 ns before the active High clock pulse. For four banks of 32 bits with parity, 36 x 4 = 144 DRAMs, the capacitive load on the address outputs will be 720 pF = 5 pF/DRAM x 144 DRAMs. The Am29C668 requires 45 ns to drive the address lines; therefore the addresses are valid 32 ns after the active High clock pulse. This is sufficient to meet the address setup time to $\overline{RAS}$. CASIEN is not asserted until wait state WS_1, allowing more than enough time for the column address to become valid. On consecutive accesses to the same page, the address is active 32 ns after the rising edge of the clock and this is more than 32 ns before the fastest $\overline{CAS}$n.

The signal $\overline{MEMREQ}$ must meet the setup time for the State PAL, which is 13 ns for the 22V10. Since the address is not valid until 40 ns after the rising edge of the clock, only 9 ns remain for the address decoding. This means that the Decoder PAL must be a 16L8-7 and the State PAL is a 22V10-15 in order to satisfy the system timings.

Refresh cycles require six processor cycles: two cycles for precharge and four cycles for the memory refresh. Table 1 shows the number of cycles needed to access memory for different processor and memory speeds.

Table 1. Access Cycles for Different Processor and Memory Speeds

Memory Speed (ns)	Processor Speed-ns 16 MHz		20 MHz		25 MHz	
	Initial	Cache Mode	Initial	Cache Mode	Initial	Cache Mode
120	4	2	5	3	6	3
100	4	2	5	3	5	3
85	4	2	4	3	5	3

PAL Equations

The following are the logic equations for the three PAL devices. They are written in PLPL software.

```
"       November 28, 1988
Am29C668 to 80386 Decoder PAL. This PAL will decode the accesses to memory and
generate the /MEMREQ and ALE signals for the 29C668. Program is used to configure
the 29C668. A[2:21] address the long words (32-bits), A[23:24] selects the bank."

DEVICE   Decoder (P20L8)

PIN  A[31:24] = 1:8 (INPUT COMBINATORIAL)
     A[15:12] = 9:11 (INPUT COMBINATORIAL)
     ADS = 14 (INPUT COMBINATORIAL)
     M_IOL = 19 (INPUT COMBINATORIAL)
     W_RL = 16 (INPUT COMBINATORIAL)
     STATE[2:1] = 17:18 (INPUT COMBINATORIAL)
     D_CL = 23 (INPUT COMBINATORIAL)

     /MEMREQ = 22 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     PROGRAM = 21 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     T_RL = 20 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     RL = 15 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE PROGADDR = A[15] * A[14] * A[13] * A[12],
       MEMADDR =  /A[31] * /A[30] * /A[29] * /A[28] * /A[27] * /A[26] * /A[25];

BEGIN
ENABLE (MEMREQ,PROGRAM,RL,T_RL); ENABLE (W_RL,STATE[2:1], M_IOL) = 0;

"MEMREQ AND PROGRAM CAN BE OCCUPY ANY SPACE BY CHANGING THE DECODING."
MEMREQ = M_IOL * MEMADDR;
/PROGRAM = /M_IOL * PROGADDR * D_CL;

/RL = ADS * PROGRAM;
/T_RL = /W_RL * /(/STATE[2] * STATE[1]) + T_RL * /STATE[2] * STATE[1];

END.
TEST_VECTORS
IN A[15:13], A[31:25], ADS, M_IOL, D_CL, W_RL, STATE[2:1];
OUT MEMREQ, PROGRAM, RL, T_RL;
BEGIN
"                        M              M P
A A A A A A A A A A A A A I D W S S      E R   T
1 1 1 1 3 3 2 2 2 2 2 2 2 D O C R T T      M G R R
5 4 3 2 1 0 9 8 7 6 5 S L L L 2 1      R M L L
-------------------------------------------------"
1 1 1 1 0 0 0 0 0 0 1 0 1 1 0 0 0      L L L H;
1 1 1 1 0 0 0 0 0 0 1 1 1 1 0 0 1      L L L H;
1 1 1 1 0 0 0 0 0 0 0 1 1 1 1 0 1      H L L H;
1 1 1 1 0 0 0 0 0 0 0 1 0 1 1 1 1      L H L L;
1 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 1      L H H L;
0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 1 0 1      L L L L;
0 1 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 1      L L L L;
END.
```

```
"      November 28, 1988
Am29C668 to 80386 Decoder PAL. This PAL generates REFRESH every 15 µsec if a
hidden refresh has not been performed. /INIT is generated every 15 µsec to reset
RFDONE in the State PAL and to reinitialize the counter. The timing of this device
can be altered by changing the initial value of the counter. "

DEVICE   TIMER (20X10A)

PIN  CLOCK = 1 (CLOCK)
     RFDONE = 2 (INPUT COMBINATORIAL)
     RESET = 3 (INPUT COMBINATORIAL)
     /RFRQ = 4 (INPUT COMBINATORIAL)
     OE = 13 (CONTROL)

     RCT[7:0] = 23:16 (OUTPUT ACTIVE_HIGH REGISTERED)
     /REFRESH = 15 (OUTPUT ACTIVE_LOW REGISTERED)
     /INIT = 14 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE
START_REFRESH=/RCT[7]*/RCT[6]*/RCT[5]*/RCT[4]*RCT[3]*/RCT[2]*RCT[1]* RCT[0];

BEGIN

INIT := /RCT[7]*/RCT[6]*/RCT[5]*/RCT[4]*/RCT[3]*/RCT[2]*/RCT[1]*RCT[0] + RESET;

REFRESH  := /RFDONE *START_REFRESH + REFRESH */RFRQ;

RCT[0]   :=/RCT[0]*/INIT + 1 * INIT;

RCT[1]   :=/RCT[0]*/INIT;
XOR(RCT[1])  := RCT[1]*/INIT + 1 * INIT;

RCT[2]   :=/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[2])  := RCT[2]*/INIT + 0 * INIT;

RCT[3]   :=/RCT[2]*/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[3])  := RCT[3]*/INIT + 1 * INIT;

RCT[4]   :=/RCT[3]*/RCT[2]*/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[4])  := RCT[4]*/INIT + 1 * INIT;

RCT[5]   :=/RCT[4]*/RCT[3]*/RCT[2]*/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[5])  := RCT[5]*/INIT + 1 * INIT;

RCT[6]   :=/RCT[5]*/RCT[4]*/RCT[3]*/RCT[2]*/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[6])  := RCT[6]*/INIT + 1 * INIT;

RCT[7]   :=/RCT[6]*/RCT[5]*/RCT[4]*/RCT[3]*/RCT[2]*/RCT[1]*/RCT[0]*/INIT;
XOR(RCT[7])  := RCT[7]*/INIT + 1 * INIT;

END.
```

```
TEST_VECTORS
IN CLOCK, RESET, RFDONE,OE,RFRQ;
OUT RCT[7:0],INIT, REFRESH;
BEGIN
"
C R R                          R
L E F   R        R R R R R R R F
O S D   F        C C C C C C C N R
C E N O R        T T T T T T T I S
K T E E Q        7 6 5 4 3 2 1 0 T H
---------------------------------------"
0 0 0 1 0        H H H H H H H L L;
C 1 0 1 0        H H H H H H L H L;    "RESET THE TIMER"
C 0 0 1 0        H H H H H L H H L L;  "INITIAL COUNT VALUE"
C 0 0 1 0        H H H H H L H L L L;  "DECREMENTS"
C 0 0 1 0        H H H H H L L H L L;
P 0 0 0 0        0 0 0 0 1 1 1 1 1 1;  "PRELOAD VALUE TO TEST"
0 0 0 1 0        H H H H L L L L L L;  "THAT COUNTER DECREMENTS"
C 0 0 1 0        H H H L H H H H L L;  "AT BOUNDARIES"
P 0 0 0 0        0 0 0 1 1 1 1 1 1 1;
0 0 0 1 0        H H H L L L L L L L;
C 0 0 1 0        H H L H H H H H L L;
P 0 0 0 0        0 0 1 1 1 1 1 1 1 1;
0 0 0 1 0        H H L L L L L L L L;
C 0 0 1 0        H L H H H H H H L L;
P 0 0 0 0        0 1 1 1 1 1 1 1 1 1;
0 0 0 1 0        H L L L L L L L L L;
C 0 0 1 0        L H H H H H H H L L;
P 0 0 0 0        1 1 1 1 1 1 1 0 1 1;  "PRELOAD 1"
0 0 0 1 0        L L L L L L L H L L;
C 0 0 1 0        L L L L L L L L H L;  "TEST THE INIT SIGNAL"
C 0 0 1 0        H H H H H L H H L L;
P 0 0 0 0        1 1 1 1 0 0 1 1 1 1;  "PRELOAD 12"
0 0 0 1 0        L L L L H H L L L L;
C 0 0 1 0        L L L L H L H H L L;  "TEST REFRESH SIGNAL"
C 0 0 1 0        L L L L H L H L L H;
C 0 0 1 1        L L L L H L L H L L;
P 0 0 0 0        1 1 1 1 0 0 1 1 1 1;  "PRELOAD 12"
0 0 1 1 0        L L L L H H L L L L;
C 0 1 1 0        L L L L H L H H L L;  "TEST REFRESH SIGNAL"
C 0 1 1 0        L L L L H L H L L L;
END.
```

```
"      November 28, 1988
Am29C668 to 80386 CONTROL PAL. This PAL contains the memory state machine. It also
generates the CAS Input Enable (CASIEN), the RAS Input (RASI), Refresh Request
(/RFRQ), the Refresh Done (RFDONE) and the Ready (/RDY) and Next Address (/NA)
signals to the processor."

DEVICE   State (P22V10)

PIN  CLOCK = 1 (INPUT COMBINATORIAL)
     ADS = 2 (INPUT COMBINATORIAL)
     /REFRESH = 3 (INPUT COMBINATORIAL)
     PROGRAM = 4 (INPUT COMBINATORIAL)
     CH = 5 (INPUT COMBINATORIAL)
     /MEMREQ = 6 (INPUT COMBINATORIAL)
     RESET = 7 (INPUT COMBINATORIAL)
     INIT = 8 (INPUT COMBINATORIAL)

     CASIEN = 14 (OUTPUT ACTIVE_HIGH COMBINATORIAL)
     /RDY = 15 (OUTPUT ACTIVE_LOW REGISTERED)
     /RFRQ = 16 (OUTPUT ACTIVE_LOW REGISTERED)
     STATE[2:0] = 17:19 (OUTPUT ACTIVE_HIGH REGISTERED)
     RASI = 20 (OUTPUT ACTIVE_HIGH REGISTERED)
     /NA = 21 (OUTPUT ACTIVE_LOW REGISTERED)
     RFDONE = 22 (OUTPUT ACTIVE_HIGH REGISTERED)
     OE = 23 (OUTPUT ACTIVE_LOW COMBINATORIAL);

"State machine definitions."
DEFINE
  IDLE= /STATE[2] * /STATE[1] * /STATE[0],
  WS1 = /STATE[2] * /STATE[1] *  STATE[0],
  WS2 = /STATE[2] *  STATE[1] * /STATE[0],
  ACC = /STATE[2] *  STATE[1] *  STATE[0],
  PL =  STATE[2] *   STATE[1] *  STATE[0],
  TI =  STATE[2] *   STATE[1] * /STATE[0],
  PC1 =  STATE[2] * /STATE[1] * /STATE[0],
  PC2 =  STATE[2] * /STATE[1] *  STATE[0],
  IDLE_ST = #B000,
  WS1_ST = #B001,
  WS2_ST = #B010,
  ACC_ST = #B011,
  PL_ST = #B111,
  TI_ST = #B110,
  PC1_ST = #B100,
  PC2_ST = #B101;

BEGIN

ENABLE (RASI,CASIEN,RFRQ,STATE[2:0],RFDONE,OE);
ENABLE (RDY,NA) = OE;

OE = (WS1 + WS2 + ACC + PL + PROGRAM) * /RFRQ;

CASIEN = RASI * /RFRQ * (/ADS * WS1 + ADS * WS2 + ACC +
  PL * /CLOCK);

IF (RESET) THEN BEGIN     "On RESET initialize all variables"
  STATE[2:0] := IDLE_ST;
  RFRQ := 0;
  RASI := 0;
  RFDONE := 0;
END;
```

```
CASE (STATE[2:0]) BEGIN

IDLE_ST) IF (RFRQ * /PROGRAM) THEN BEGIN
    STATE[2:0] := WS1_ST;
    RFRQ := 1;
    RASI := 1;
    RFDONE := RFDONE * /INIT;
  END;
  ELSE IF(ADS * MEMREQ * /REFRESH) THEN BEGIN
    STATE[2:0] := WS1_ST;
    RFDONE := RFDONE * /INIT;
    RASI := 1;
  END;
  ELSE BEGIN
    RASI := 0;
    STATE[2:0] := IDLE_ST;
    IF (PROGRAM * ADS * /REFRESH) THEN RDY = 1;
    RFRQ := ADS * /PROGRAM * /MEMREQ * /RFDONE +
      REFRESH;       "TURN ON RFRQ 1 CYCLE EARLY"
    RFDONE := RFDONE * /INIT;
  END;
WS1_ST) BEGIN
    STATE[2:0] := WS2_ST;
    RFRQ := RFRQ;
    RFDONE := RFRQ * /REFRESH * /INIT;
    RASI := 1;
    IF (/RFRQ * ADS) THEN BEGIN
      NA := 1;
    END;
  END;
WS2_ST) BEGIN
    RASI := 1;
    RFRQ := RFRQ;
    RFDONE := RFDONE * /INIT;
    IF (ADS) THEN BEGIN
      STATE[2:0] := ACC_ST;
      IF (/RFRQ) THEN BEGIN
        NA := 1;
        RDY := 1;
      END;
    END;
    ELSE BEGIN
      STATE[2:0] := WS2_ST;
      IF (/RFRQ) THEN BEGIN
        NA := 1;
      END;
    END;
  END;
ACC_ST) BEGIN
    RFRQ := RFRQ;
    RFDONE := RFDONE * /INIT;
    IF (RFRQ + REFRESH + ADS * CH * MEMREQ + ADS * /MEMREQ * /RFDONE)
      THEN BEGIN
      STATE[2:0] := PC1_ST;
      RASI := 0;
  END;
  ELSE IF (ADS * CH * MEMREQ) THEN BEGIN
    STATE[2:0] := PL_ST;
    NA := 1;
    RASI := 1;
    RDY := 1;
```

```
  END;
  ELSE IF (ADS + ADS * /MEMREQ) THEN BEGIN
    STATE[2:0] := TI_ST;
    RASI := 1;
  END;
  END;
PL_ST) BEGIN
    RFRQ := RFRQ;
    NA = 1;
    RASI := 1;
    RFDONE := RFDONE * /INIT;
    IF (ADS) THEN BEGIN
      STATE[2:0] := ACC_ST;
      RDY = 1;
    END;
    ELSE BEGIN
      STATE[2:0] := PL_ST;
      RDY = 0;
    END;
  END;
TI_ST) BEGIN
    RFDONE := RFDONE * /INIT;
    RFRQ := RFRQ;
    IF (REFRESH + ADS * MEMREQ * CH + ADS * /MEMREQ * /RFDONE)
        THEN BEGIN
      STATE[2:0] := PC1_ST;
      RASI := 0;
    END;
    ELSE IF (ADS * MEMREQ * CH) THEN BEGIN
      STATE[2:0]:= PL_ST;
      NA : = 1;
      RASI := 1;
    END;
    ELSE BEGIN
      STATE[2:0] := TI_ST;
      RASI := 1;
    END;
  END;
PC1_ST) BEGIN
    STATE[2:0] := PC2_ST;
    RASI := 0;
    RFRQ := REFRESH + /MEMREQ * /RFDONE * /RFRQ;
    RFDONE := RFDONE * /INIT;
  END;
PC2_ST) BEGIN
    RFRQ := RFRQ;
    RFDONE := RFDONE * /INIT;
    IF (/RFRQ * /MEMREQ) THEN BEGIN
      STATE[2:0] := IDLE;
      RASI := 0;
    END;
    ELSE BEGIN
      STATE[2:0] := WS1_ST;
      RASI := 1;
    END;
  END;
END; "CASE"

END.
```

```
TEST_VECTORS
IN CLOCK,ADS,REFRESH,PROGRAM,CH,MEMREQ,RESET,INIT;
OUT RASI,CASIEN,RFRQ,RFDONE,STATE[2:0],RDY,NA,OE;
BEGIN
"                        C
              M R        A   R
    R P   E E I        R S R F
C A F R   M S N        A I F D S S S R
L D S G C R E I        S E R N T T T D N O
K S H M H Q T T        I N Q E 2 1 0 Y A E
-------------------------------------------"
C 0 0 0 X 0 1 0        L L L L L L L Z Z L;   "SYSTEM RESET"
C 0 0 0 X 0 0 0        L L L L L L L Z Z L;   "IDLE STATE"
C 1 0 1 X 0 0 0        L L L L L L L L L H;   "PROGRAMMING CYCLE"
C 0 0 1 X 0 0 0        L L L L L L L H L H;   "ASSERT RDY TO 386"
C 0 0 0 X 0 0 0        L L L L L L L Z Z L;
C 1 0 0 X 1 0 0        H L L L L L H L L H;   "MEMORY ACCESS"
C 0 0 0 X 1 0 0        H H L L L L H L L H;   "ASSERT /NA TO PIPELINE ADDR."
C 0 0 0 0 1 1 0 0      H H L L L H H H H H;   "INITIAL ACCESS TERMINATES"
1 1 0 0 1 1 0 0        H L L L H H H H H H;   "PIPELINED ACCESS"
C 0 0 0 0 1 0 0        H H L L L H H H H H;   "PIPELINED ACCESS TERMINATED."
C 1 0 0 0 1 0 0        L L L L H L L Z Z L;   "PAGE MISS"
C 0 1 0 1 1 0 0        L L H L H L H Z Z L;   "PRECHARGE. REFRESH REQUEST"
C 0 1 0 X 1 0 0        H L H L L L H Z Z L;   "REFRESH ACCESS"
C 0 1 0 X 1 0 0        H L H L L H L Z Z L;
C 0 1 0 X 1 0 0        H L H L L H H Z Z L;
C 0 0 0 X 1 0 0        L L H L H L L Z Z L;   "PRECHARGE."
C 0 0 0 X 1 0 0        L L L L H L H Z Z L;   "PRECHARGE"
C 0 0 0 X 1 0 0        H H L L L L H L L H;   "MEMORY ACCESS"
C 0 0 0 X 1 0 0        H H L L L H L L H H;   "ASSERT /NA TO PIPELINE ADDR."
C 0 0 0 X 0 0 0        H H L L L H H H H H;   "ACCESS TERMINATES"
C 0 0 0 X 0 0 0        H L L L H H L Z Z L;   "NO PENDING ACCESS."
C 0 0 0 X 0 0 0        H L L L H H L Z Z L;   "TI STATE"
C 1 0 0 1 1 0 0        H H L L H H H L H H;   "MEMORY ACCESS"
C 0 0 0 X 0 0 0        H H L L L H H H H H;   "ACCESS TERMINATES"
C 0 0 0 X 1 0 0        H L L L H H L Z Z L;   "NO PENDING ACCESS."
C 0 0 0 X 0 0 0        H L L L H H L Z Z L;   "TI STATE"
C 1 0 0 0 1 0 0        L L L L H L L Z Z L;   "PRECHARGE."
C 0 0 0 X 1 0 0        L L L L H L H Z Z L;   "PRECHARGE."
C 0 0 0 X 1 0 0        H H L L L L H L L H;   "MEMORY ACCESS"
C 0 0 0 X 1 0 0        H H L L L H L L H H;   "ASSERT /NA TO PIPELINE ADDR."
C 0 0 0 X 0 0 0        H H L L L H H H H H;   "ACCESS TERMINATES"
C 0 0 0 X 0 0 0        H L L L H H L Z Z L;   "NO PENDING ACCESS."
C 0 0 0 X 0 0 0        H L L L H H L Z Z L;   "TI STATE"
C 0 1 0 1 0 0 0        L L L L H L L Z Z L;   "PRECHARGE. REFRESH REQUEST"
C 0 1 0 1 0 0 0        L L H L H L H Z Z L;
C 0 1 0 X 0 0 0        H L H L L L H Z Z L;   "REFRESH ACCESS"
C 0 1 0 X 0 0 0        H L H L L H L Z Z L;
C 0 1 0 X 0 0 0        H L H L L H H Z Z L;
C 0 0 0 X 0 0 0        L L H L H L L Z Z L;   "PRECHARGE."
C 0 0 0 X 0 0 0        L L L L H L H Z Z L;   "PRECHARGE. REFRESH REQUEST"
C 0 0 0 X 0 0 0        L L L L L L L Z Z L;   "IDLE"
C 0 0 0 X 0 0 0        L L L L L L L Z Z L;   "START REFRESH ACCESS"
C 0 1 0 X 0 0 0        L L H L L L L Z Z L;
C 0 1 0 X 0 0 0        H L H L L L H Z Z L;   "REFRESH ACCESS"
C 0 1 0 X 0 0 0        H L H L L H L Z Z L;
C 0 1 0 X 0 0 0        H L H L L H H Z Z L;
C 0 0 0 X 0 0 0        L L H L H L L Z Z L;   "PRECHARGE."
C 0 0 0 X 0 0 0        L L L L H L H Z Z L;   "PRECHARGE. REFRESH REQUEST"
C 0 0 0 X 0 0 0        L L L L L L L Z Z L;   "IDLE"
C 1 0 0 X 1 0 0        H L L L L L H L L H;   "DELAYED MEMORY ACCESS"
```

```
C 1 0 0 X 1 0 0        H L L L L H L L L H;
C 1 0 0 X 1 0 0        H L L L L H L L H H;
C 1 0 0 X 1 0 0        H L L L L H L L H H;    "ASSERT /NA TO PIPELINE ADDR."
C 0 0 0 1 1 0 0        H H L L L H H H H H;    "INITIAL ACCESS TERMINATES"
C 1 0 0 X 0 0 0        L L L L H L L Z Z L;    "HIDDEN REFRESH"
C 1 0 0 X 0 0 0        L L H L H L H Z Z L;    "PRECHARGE. REFRESH REQUEST"
C 0 0 0 X 1 0 0        H L H L L L H Z Z L;    "REFRESH ACCESS"
C 0 0 0 X 1 0 0        H L H H L H L Z Z L;
C 0 0 0 X 1 0 0        H L H H L H H Z Z L;
C 0 0 0 X 1 0 0        L L H H H L L Z Z L;    "PRECHARGE."
C 0 0 0 X 1 0 0        L L L H H L H Z Z L;    "PRECHARGE"
C 0 1 0 X 0 0 0        L L L H L L L Z Z L;
END.
```

Am29C668 Configurable Dynamic Memory Controller to 68020 Microprocessor Interface

by Douglas Lee, Applications Specialist

INTRODUCTION

The interface between the Am29C668 4 MBit Configurable Dynamic Memory Controller (CDMC) and the Motorola 68020 microprocessor was designed for maximum performance; therefore, lowering the total device count was a secondary concern. It is possible to interface the Am29C668 to the 68020 with fewer PAL® devices, but this would make many assumptions about the system implementation which are not generally applicable. This design is as general as possible so that the user may tailor his implementation to a specific memory system. Possible changes to the design are discussed with associated system requirements and implications. A block diagram, timing analyses and logic equations necessary to implement the design are included. This design requires a minimum number of external devices to perform the interface and glue functions: three PAL devices (one 16L8, one 22V10 and one 20X10), a decoder, and four 74F245 bidirectional transceivers.

Distinctive Characteristics

- Am29C668 4-Mbit Configurable Dynamic Memory Controller/Driver with Auto Timing
- 20-MHz 68020 Microprocessor
- 120-ns Fast Page Mode 1 Mbit x 1 DRAMs
- Two Wait-State Initial Accesses With Zero Wait-State Subsequent Page-Mode Accesses
- 4-Mbyte Dynamic Memory Expandable Up To 8-Mbyte per Am29C668

MEMORY ARCHITECTURE OVERVIEW

To obtain the maximum memory throughput but still maintain a reasonable cost, 120-ns fast page-mode 1-Mbit DRAMs are used. The 68020 requires a minimum of three processor cycles per access. If additional cycles are needed, the memory controller holds $\overline{\text{DSACK}}$[1:0] inactive. The processor inserts wait states until $\overline{\text{DSACK}}$[1:0] are asserted. The 20 MHz 68020 completes the initial access to memory in two wait states (five cycles total). The subsequent accesses within the page are performed with no wait states (three cycles total).

Page-mode DRAMs appear to the processor as if they are fast cache memories during page accesses. The page size for a 1-Mbit DRAM is 1024 bits or 1 Kbits. The 120-ns page-mode DRAM is 32 bits wide, therefore the page size is 4 Kbytes. The Am29C668 detects accesses within the same page via on-chip cache-mode operation. When a new address is latched, it is compared with the previous address; if the row and bank addresses are the same, the Cache Hit signal $\overline{\text{CH}}$ is asserted. The memory state machine immediately begins the next access. An access outside the page, a page miss, causes the memory controller to perform the $\overline{\text{RAS}}$ precharge for the DRAMs. The total access time on a page miss requires seven cycles, one for decoding, two cycles for the $\overline{\text{RAS}}$ precharge and four for the data access. This method of accessing memory results in shorter access times than memories using normal DRAM accesses. For certain systems, this memory can result in near-zero wait-state accesses. Only static RAMs can guarantee zero-wait-state accesses. The actual performance of the memory depends upon the instruction mix of the programs executed.

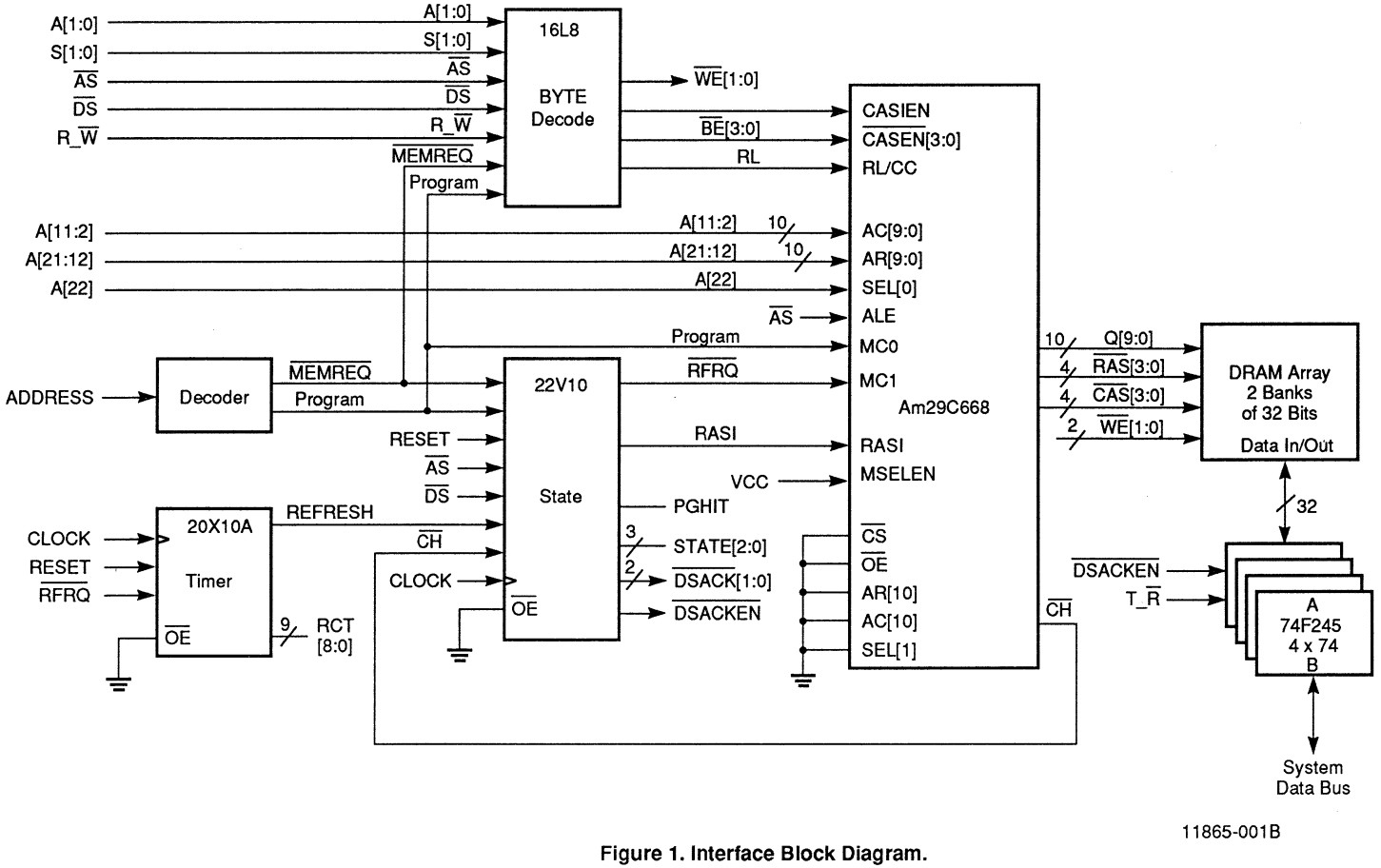
The memory array consists of two banks, each containing 4 Mbytes or 1 Mword (32 bits) of memory. This gives a maximum size of 8 Mbytes or 2 Mwords of memory per Am29C668 controller.

A 20-MHz system was selected since high performance may be achieved even with 120-ns DRAMs. As processor speeds increase toward 25 and 30 MHz, the number of wait states increases until this memory architecture becomes impractical. Faster microprocessors demand faster DRAMs, static RAM caches, bank interleaving, specialty mode DRAMs, or more exotic architectures, implemented at considerable cost. These topics are beyond the scope of this application note.

FUNCTIONAL DESCRIPTION

The primary data paths and functional elements are shown in Figure 1. The following discussion describes each subsection of the block diagram, including the control logic, buffers and memory array.

*PAL is a registered trademark of Advanced Micro Devices, Inc.



11865-001B

Figure 1. Interface Block Diagram.

Am29C668 CDMC

The Am29C668 generates the $\overline{RAS}$, $\overline{CAS}$ and address signals to the DRAM array; no external drivers are needed. Additionally, the Am29C668 generates the row addresses during $\overline{RAS}$-only refreshes from its internal refresh row-address counter. The Am29C668, operating in the auto-timing mode, generates the $\overline{RAS}$-to-$\overline{CAS}$ and the $\overline{RAS}$-to-address timing internally.

The Am29C668 must be programmed before any memory accesses can occur. This is accomplished through a dummy memory access because the 68020 memory maps all I/O devices. The Am29C668 occupies a 4-Kbyte or 1-Kword address space. The actual decoding is left up to the user since it is highly system dependent. The address bits A[11:2] contain the value to be loaded into the configuration register. For this design, the lower 12 address bits are 198H. The options selected are: two banks $\overline{RAS}$ and $\overline{CAS}$ configuration, $\overline{CAS}$ byte decoding, $\overline{RAS}$-only refresh, 1-Mbit DRAM size, cache mode and auto-timing. If the part is used in only one configuration, only a single byte of address space is needed. The input AC[10] must be tied Low to place the Am29C668 in normal-mode operation. AC[10] is only used with 4-Mbit DRAMs.

When the Am29C668 is configured for two banks, $\overline{RAS}$[1:0] are connected to the $\overline{RAS}$ inputs of bank 0, and $\overline{RAS}$[3:2] are connected to bank 1 to minimize the capacitive load each output must drive. $\overline{CAS}$[3] controls byte 3 of both banks and similarly $\overline{CAS}$[2] controls byte 2; $\overline{CAS}$[1] controls byte 1 and $\overline{CAS}$[0] controls byte 0. This provides for byte accesses to memory.

The Am29C668 supports byte decoding through the $\overline{CASEN}$[3:0] inputs. The two least significant address bits A[1:0] and the Size bits SIZ[1:0] from the 68020 are decoded to generate four byte-enable signals, $\overline{BE}$[3:0]. These byte-enable signals are connected to $\overline{CASEN}$[3:0] and only the selected bytes are accessed. The bytes that are not selected perform a $\overline{RAS}$-only refresh on the current row address. (Note: $\overline{BE}$[3] enables bits 31 to 24, $\overline{BE}$[2] enables bits 23 to 16, $\overline{BE}$[1] enables bits 15 to 8, $\overline{BE}$[0] enables bits 7 to 0.)

The auto-timing mode generates the $\overline{RAS}$, $\overline{CAS}$ and multiplexed address inputs to the DRAMs. This timing is optimized for 100-ns DRAMs. The $\overline{CH}$ signal from the Am29C668 is used to determine if the current address in the input latch has the same row and bank addresses as the previous access. If the access is to the active row, $\overline{CH}$ is asserted and a page-mode access is initiated. If $\overline{CH}$ is deasserted, $\overline{RAS}$ must be precharged and a normal access occurs.

The Address Strobe signal $\overline{AS}$ from the 68020 is input directly to the Address Latch Enable ALE of the Am29C668. When $\overline{AS}$ is deasserted, the address latch of the Am29C688 is transparent. When $\overline{AS}$ is asserted, the address is latched.

Byte Decoder PAL

The Byte Decoder PAL generates the byte-enable, write-enable, register load and output-enable signals. The following inputs are used:

A[1:0]	System address bit 1 and 0, for byte decoding.
SIZ[1:0]	Size from 68020, indicates the number of bytes remaining to be transferred.
$\overline{AS}$	Address Strobe, signals valid address and control signals.
$\overline{DS}$	Data Strobe, indicates valid data on the bus during write cycles, or signals memory to drive data on the bus during read cycles.
R_$\overline{W}$	Read active High, Write active Low, from 68020.
PROGRAM	Program cycle, from decoder.
$\overline{MEMREQ}$	Memory Access Request, from decoder.

The following outputs are generated:

$\overline{WE}$[1:0]	Write Enable, enables writing to DRAMs.
$\overline{BE}$[3:0]	Byte Enable, selects active bytes during access.
$\overline{DSACKEN}$	Data and Size Acknowledge Enable, enables $\overline{DSACK}$ and bus drivers during valid memory accesses and programming cycles.
RL	Register load signal, input to Am29C668.

Table 1. Byte Enable Decoding for the 68020

A1	A0	SIZ1	SIZ0	$\overline{BE3}$	$\overline{BE2}$	$\overline{BE1}$	$\overline{BE0}$	Access Type
0	0	0	1	1	1	1	0	
0	1	0	1	1	1	0	1	
1	0	0	1	1	0	1	1	Byte
1	1	0	1	0	1	1	1	
0	0	1	0	1	1	0	0	
0	1	1	0	1	0	0	1	
1	0	1	0	0	0	1	1	Word
1	1	1	0	0	1	1	1	
0	0	1	1	1	0	0	0	
0	1	1	1	0	0	0	1	
1	0	1	1	0	0	1	1	3 Bytes
1	1	1	1	0	1	1	1	
0	0	0	0	0	0	0	0	
0	1	0	0	0	0	0	1	
1	0	0	0	0	0	1	1	Long Word
1	1	0	0	0	1	1	1	

Note: 0 is 0 V and 1 is 5 V.

The Write Enable $\overline{WE}$[1:0] signals are input to the DRAM array to control read and write accesses. When active, a write access is performed and when inactive, a read access is performed. One signal is generated per bank to reduce the capacitive load on the output driver. All write cycles performed are "early" write cycles ($\overline{WE}$ active before $\overline{CAS}$) that prevent the DRAM output driver from turning on. As a result, the Data In DIN and Data Out DOUT pins of the DRAMs can be tied together, reducing the number of routes on the PC board.

Byte Enable outputs $\overline{BE}$[3:0] are used to select the bytes to be accessed and to control the assertion of the $\overline{CAS}_n$ outputs. They are generated from the address bits A[1:0] and the SIZ[1:0] inputs from the 68020. Table 1 gives this decoding scheme.

The Register Load RL signal is used to load the Am29C668 configuration register. It is asserted when the processor performs an access to the configuration-register address space. RL is asserted after PROGRAM is asserted and $\overline{AS}$ is asserted to ensure the correct setup of MC0.

State PAL

The State PAL arbitrates between memory accesses and refresh cycles, generates the bus-control signals to the 68020 and contains the state machine that controls the memory accesses. The following inputs are used:

CLOCK 20-MHz system clock.

$\overline{REFRESH}$ Refresh request signal from the Timer PAL.

PROGRAM Program cycle signal from the Decoder PAL.

$\overline{MEMREQ}$ Memory cycle request from the Decoder PAL.

RESET System reset signal.

$\overline{AS}$ Address Strobe, indicates when address signals are valid.

$\overline{DS}$ Data Strobe, indicates when data is valid.

$\overline{CH}$ Cache Hit; signals access within active page.

The following outputs are generated:

RASI Row Address Strobe Input, input to the Am29C668.

PGHIT PGHIT, signals when an access to the same page is detected.

$\overline{DSACKEN}$ $\overline{DSACK}$ Enable, enables $\overline{DSACK}$ output.

$\overline{DSACK}$[1:0] Data Size and Acknowledge signal to the 68020, signals termination of valid access and size of memory port.

$\overline{RFRQ}$ Refresh Request, signals refresh cycles to the Am29C668.

$\overline{STATE}$[2:0] State variables for the state machine.

Data and Size Acknowledge Enable $\overline{DSACKEN}$ is output to the four 74F245s and to the state PAL. This signal enables the output drivers of the bus transceivers. $\overline{DSACKEN}$ also enables the output $\overline{DSACK}$. This signal is necessary since $\overline{DSACK}$[1:0] are asserted during both program and memory accesses. During program cycles, the output buffers are enabled, but the data is not written to memory.

$\overline{DSACK}$[1:0] are generated by the state PAL so that the memory can drive both inputs Low to signal the processor that it is a 32-bit device and that the memory access is complete. $\overline{DSACK}$[1:0] are enabled by $\overline{DSACKEN}$ so that multiple drivers can be connected to these inputs. $\overline{DSACK}$[1:0] must be pulled up through a resistor to maintain the proper logic levels when not being driven.

CASIEN is used as an external override in the auto-timing mode. This input is also used during fast-page mode accesses. CASIEN is strobed by $\overline{DS}$, causing the $\overline{CAS}_n$ outputs to pulse, and thereby access the DRAMs. $\overline{DS}$ is ANDed with $\overline{MEMREQ}$ to generate CASIEN. $\overline{DS}$ is used to guarantee that the data is valid before $\overline{CAS}_n$ is asserted during Write cycles.

Figure 2 is the state diagram for the memory-controller state machine. Figure 3 gives the timing diagram for accesses to the memory. The following is a list of the state-machine states:

IDLE	Idle State, waiting for memory request or refresh request.
SW1	System Wait 1, memory access in progress.
SW2	System Wait 2, memory access in progress.
ACC	Access cycle, data valid during this cycle.
PM	Page Mode Wait, waiting for memory request or refresh request with RASI asserted.
PC1	Precharge cycle 1, $\overline{RAS}$ Precharge cycle 1.
PC2	Precharge cycle 2, $\overline{RAS}$ Precharge cycle 2.
PC12	Precharge cycle 1or 2, $\overline{RAS}$ Precharge cycle 1 or 2 depending upon the type of access.

Memory Accesses

Memory accesses are initiated by the 68020. The 68020 drives a valid address and a read/write signal onto the bus. Address Strobe $\overline{AS}$ is then asserted by the 68020 to signal that a memory access has begun; it is valid a maximum of 25 ns after the clock goes Low. Therefore, there is no setup time for $\overline{AS}$ relative to the clock. This means that RASI must be an asynchronous signal. RASI is not asserted until the clock is active to ensure that there is always one full cycle before the state machine begins. $\overline{AS}$ is connected directly to the ALE input of the Am29C668.

Once RASI is asserted, the state machine proceeds through states SW1, SW2 and finally ACC where the data is valid. If a refresh access is pending, the state machine goes to state PC1 on the next cycle. The memory performs the RAS precharge cycles, PC1 and PC2, and then performs the refresh access. If no refresh request is pending, the memory goes to state PM.

In the PM state, RASI is still asserted, but the $\overline{CAS}_n$ outputs are disabled because the $\overline{CASEN}_n$ inputs are deasserted. When $\overline{AS}$ is asserted, the address of the current access is compared with the previous access. If the access is to the same row, the Am29C668 asserts $\overline{CH}$. The state machine goes to state ACC and the access is completed in three cycles, no wait states inserted. If the access is to another page, a page miss, the state machine goes to the PC12 state. The state machine then goes to the IDLE state to guarantee the $\overline{RAS}$ precharge time; a normal access is then completed.

If $\overline{REFRESH}$ is asserted while the state machine is in the PM state, the state machine goes to PC12. From PC12, it goes to the PC2 state, thus completing the $\overline{RAS}$ precharge. The state machine then goes directly to the SW1 state, completes a normal access and the $\overline{RAS}$ precharge cycle. After a refresh cycle, the memory always goes to the IDLE state.

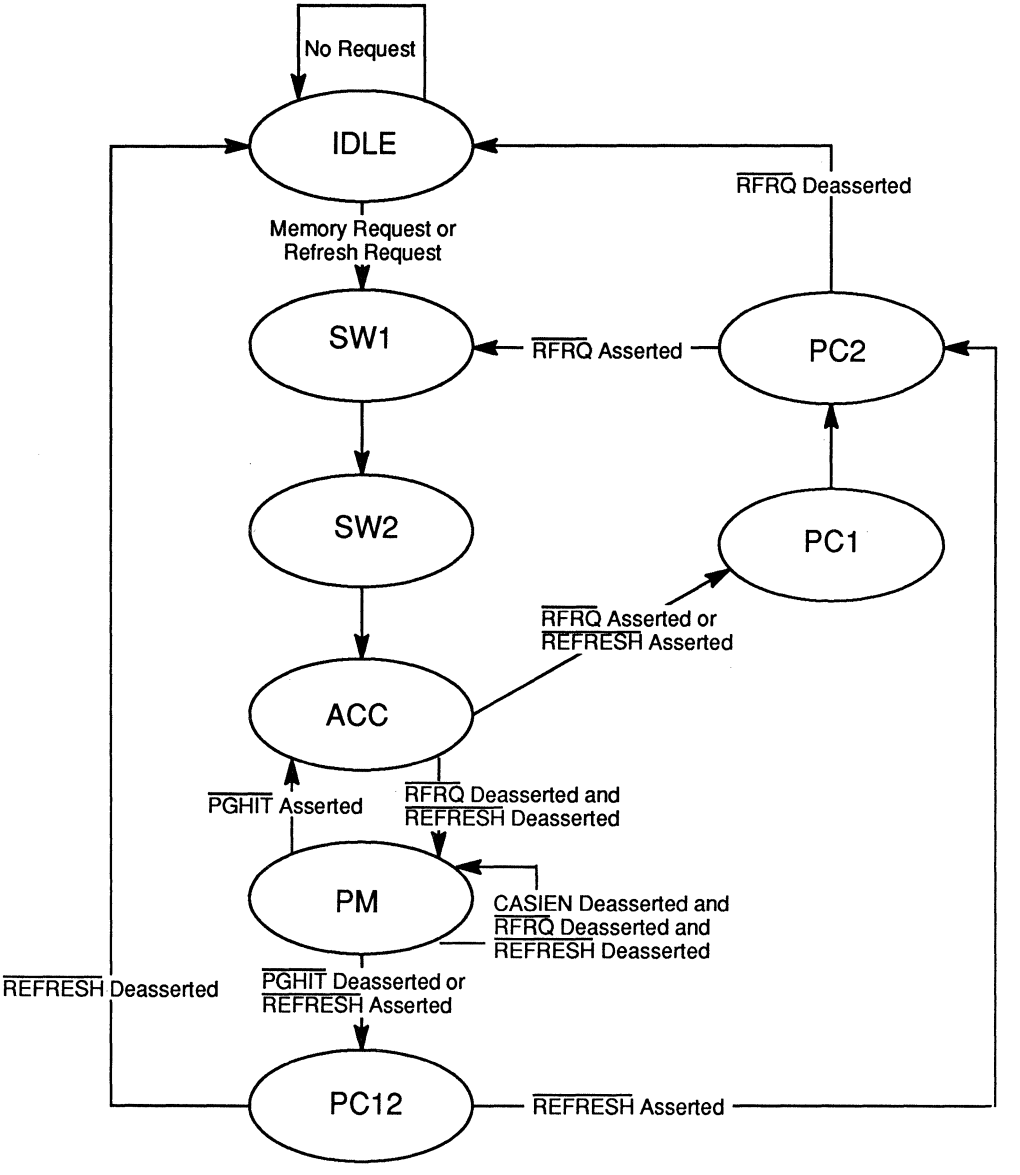


Figure 2. State Diagram

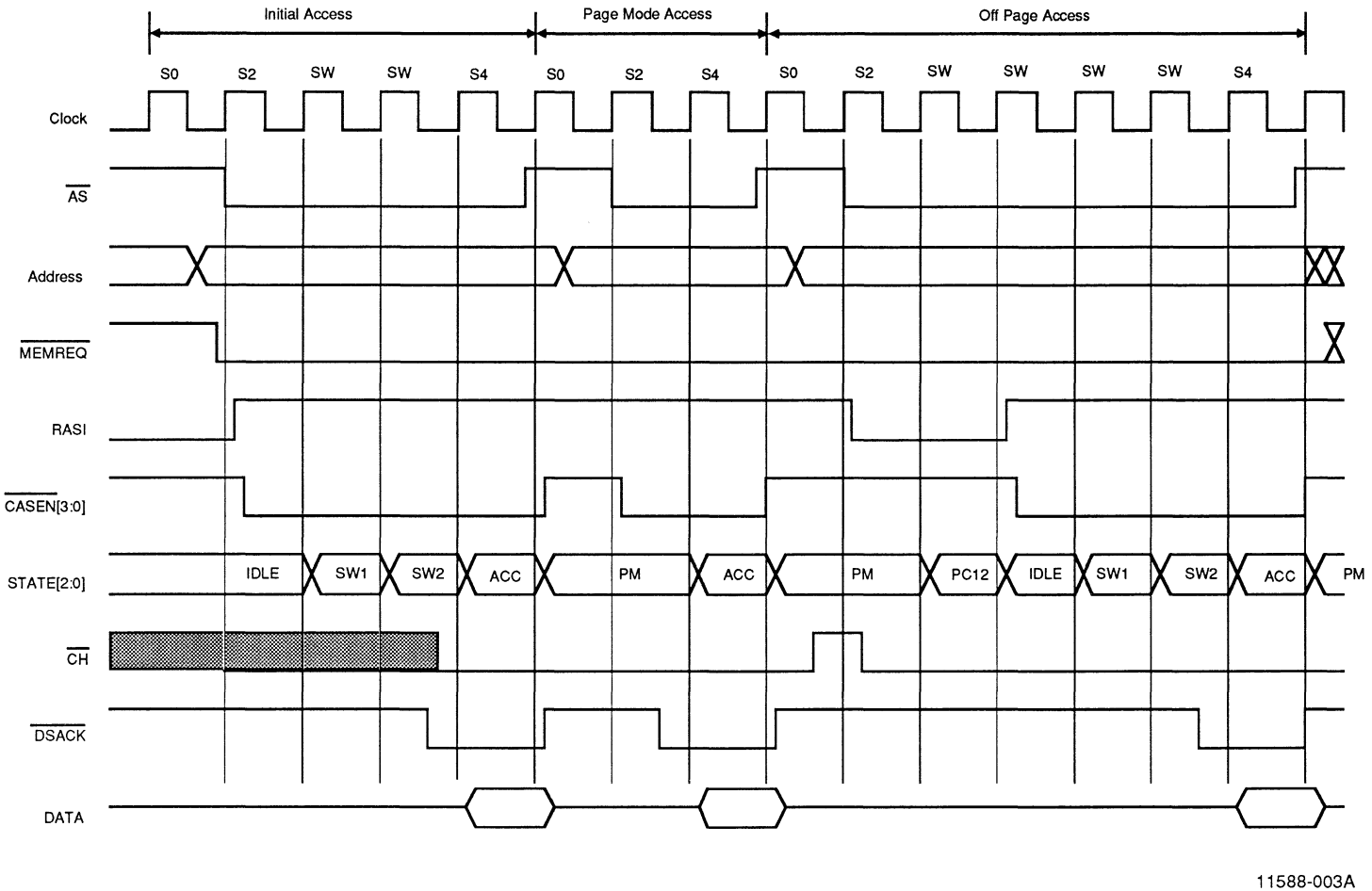


Figure 3. Timing Diagram

Refresh Cycles

Dynamic memories must be refreshed periodically to restore the charge on the storage capacitor to retain the data. For 1-Mbit DRAMs, all 512 rows of memory must be refreshed every 8 ms. There are several different methods to perform refresh cycles: burst, forced and hidden. Each method has its advantages and disadvantages. The best method is determined by the instruction mix, system hardware and performance requirements.

The first method is burst refresh cycles that refresh all 512 rows at once. It works well in systems where there are long idle times between memory accesses. The main disadvantage is that an access to memory may be delayed for long periods during the refresh cycles, greatly impacting system throughput. This would definitely not be an acceptable method for real-time systems.

Another method is to periodically insert refresh cycles; this is called forced or distributed refresh. If refreshes are interspersed between memory accesses, the memory throughput and access time is not greatly impacted, because there is a lower probability of refresh request and memory request contention. One refresh request is generated every 15.6 µs = 8 ms/512 rows. This method is preferable to burst refresh in most systems.

Even better would be for the refresh cycles to occur when the processor is accessing other memory or I/O. This type of refreshing is called hidden refresh since all or most of the refresh cycle is overlapped with another access. There are times, however, when the system continually accesses the same memory page and prevents hidden refreshes from being performed. If this happens, a forced refresh cycle must be performed. Hidden refreshing has the lowest system impact since all or most of the refresh cycle is overlapped with an access to another memory or I/O device. There are conceivable situations where hidden refresh would not perform as well as forced refresh. However, for most general applications, hidden refreshing offers the best performance.

This design utilizes forced refreshes instead of hidden refreshes. There are several reasons this method was selected. Additional logic is needed to keep track of hidden refresh cycles. This logic must suppress the forced refresh request after a hidden refresh cycle is performed and must force a refresh when no hidden refresh is performed. This method adds extra devices and consumes more board space, money and power.

A refresh cycle is identical to a normal access, except that the $\overline{CAS}$ outputs to the DRAMs must be suppressed. The Am29C668 suppresses the $\overline{CAS}$ outputs in the refresh mode. $\overline{REFRESH}$ is asserted by the Timer PAL every 15.25 µs. If a memory access is in process, the access is completed before the refresh cycle begins. If both a memory access and memory request occur during the idle state, the refresh request is given priority to ensure that the DRAM refresh requirements are met.

The $\overline{RFRQ}$ is asserted $^1/_2$ cycle before RASI is asserted to insure that the MC1-to-RASI set-up time is met. If RASI is asserted too soon, the proper row address for the refresh cycle will not be valid when $\overline{RAS}$ is asserted to the DRAMs.

Figure 4 shows the timing for a refresh cycle that begins during an idle cycle. Figure 5 shows the timing for a refresh cycle that begins during a page-mode cycle.

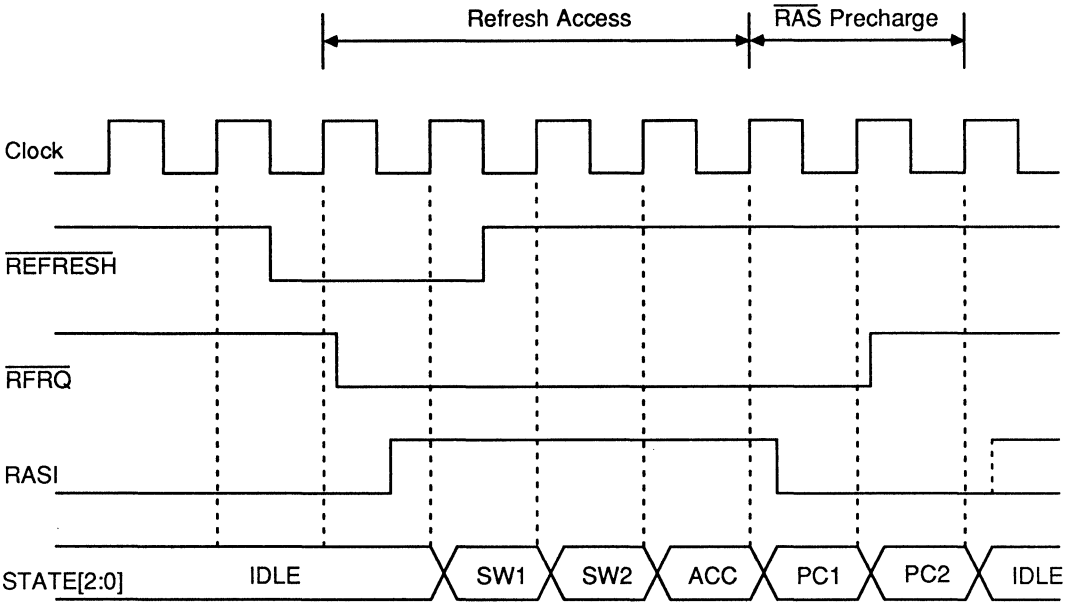


Figure 4. Refresh Timing from Initial Idle State

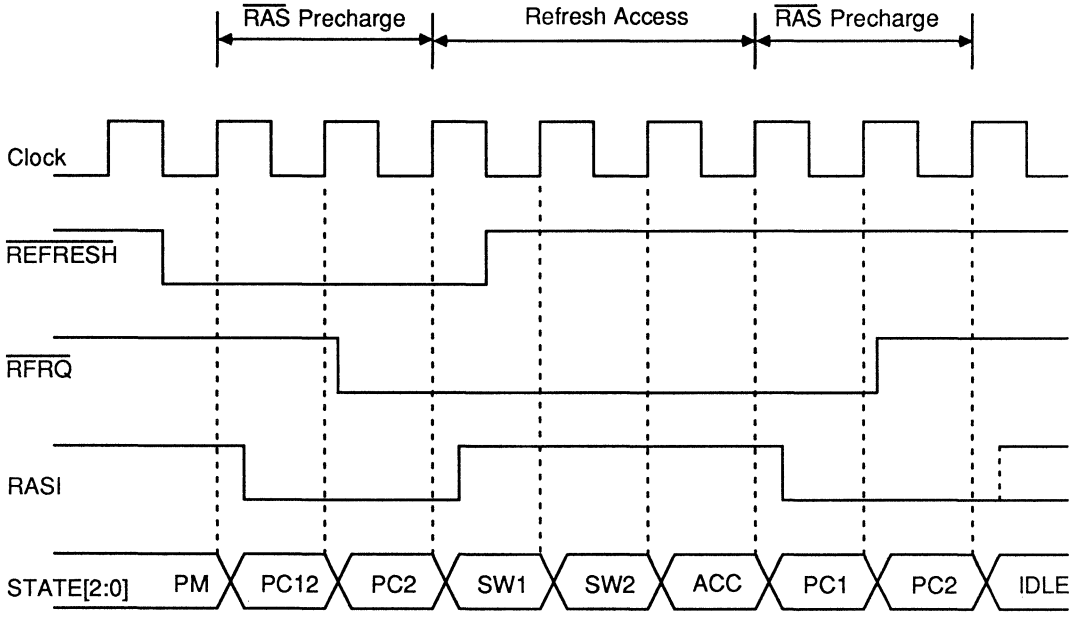


Figure 5. Refresh Timing from Initial Page Mode State

Timer PAL

Timer PAL Inputs:

CLOCK	System 20 MHz clock.
RESET	System Reset; initializes counter.
$\overline{RFRQ}$	Refresh Request; signals refresh memory access.

Timer PAL Outputs:

RCT[8:0]	Counter.
$\overline{REFRESH}$	Forced Refresh Request.

The Timer PAL helps implement the forced refreshes along with the State PAL. The period for the Timer PAL is selected by the value initialized in the counter. This value is set to 304, resulting in a refresh request cycle time of 15.25 µs = 305 x 50 ns (an extra cycle is included since the counter decrements to 0 before resetting). The initial value is determined by the location of INIT in the logic equation. $\overline{REFRESH}$ is asserted when the counter decrements to zero. $\overline{REFRESH}$ is asserted until the State PAL asserts $\overline{RFRQ}$, signaling that the refresh cycle has started.

There are several alternate methods to implement the refresh timer. A 555 timer could be used to save board space and cut cost. The problem with this solution is the need for asynchronous arbitration. The necessary logic could be included in the Byte Decoder PAL if it is expanded to a 24-pin device to accommodate all the logic. Another alternative is to use a spare DMA channel to implement the refresh requests similar to the IBM PC-AT and PS/2* systems.

Data Buffers

Simple data buffers are used to minimize the propagation delay. The $\overline{OE}$ input is generated by the Byte-Decoder-PAL output $\overline{DSACKEN}$ and T_$\overline{R}$ is connected to R_$\overline{W}$ generated by the 68020. Four 74F245s are used in this design. Since this design is synchronous to the processor clock, no latches are required for the data input/output.

Timing Analysis

The initial access to memory requires five processor cycles to complete. Each processor cycle is a minimum of 50 ns at 20 MHz. The access requires:

S0, S1 Cycles	50 ns
Clock to RASI	15 ns
RASI to $\overline{RAS}$n	21 ns
DRAM Access	120 ns
Buffer Delay	7 ns
Data Setup	5 ns
Total Access Time	218 ns

With a total access time of 218 ns, the access is completed in five processor cycles. The data must be valid 5 ns before the falling edge of the clock. This provides for four and a half cycles: 4 x 50 + 20 (parameter 2 of the 68020) = 220 ns to complete the access, allowing for a timing margin of 220 - 218 = 2 ns. Therefore, the State PAL must be a 22V10-15. The consecutive accesses within the same page require:

S0, S1 Cycles	50 ns
$\overline{DS}$ to CASIEN	10 ns
CASIEN to $\overline{CAS}$n	17 ns
DRAM Page Mode Access	30 ns
Buffer Delay	7 ns
Data Setup	5 ns
Total Access Time	119 ns

With a total access time of 119 ns, the access is completed in 2.5 cycles with a margin of (2 x 50 + 20) - 119 = 1 ns minimum. Since this is a worst-case timing analysis, this margin is sufficient.

The Timer PAL operates at a frequency of 20 MHz. For the Byte Decoder PAL, the important parameter is $\overline{BE}$[1:0], which must be valid soon enough for the $\overline{CAS}$ outputs to guarantee the data hold time of the DRAMs. Data Setup $\overline{DS}$ is 38 ns minimum and the data is held on the bus for 10 ns minimum after $\overline{DS}$ is deasserted. The data hold time required for the DRAMs is 20 ns. The propagation delay from $\overline{CASEN}$ to $\overline{CAS}$ is 17 ns for a load of 112 pF. This leaves 11 ns from $\overline{DS}$ to $\overline{BE}$n (38 + 10 - 20 - 17 = 11). Therefore, a D-speed PAL is needed for the Byte Decoder. Figure 6 shows the timing for the byte-enable signals.

The $\overline{RAS}$ precharge time for 120 ns DRAMs is 90 ns, therefore two processor cycles (50 x 2 = 100 ns) are sufficient.

*PC-AT and PS/2 are registered trademarks of IBM Corporation.

Table 2 shows the number of cycles needed to access memory for different processor and memory speeds for this architecture. The table shows that at 25 MHz, even the fastest DRAMs, 70 ns access times, require 1 wait state when accessed in the fast-page mode. Therefore, a different memory architecture would be needed to obtain higher memory and system performance at this processor speed.

Table 2. Access Cycles for Different Processor and Memory Speeds

Memory Speed (ns)	Processor Speed-ns 16 MHz		20 MHz		25 MHz	
	Initial	Cache Mode	Initial	Cache Mode	Initial	Cache Mode
120	5	3	5	3	6	4
100	5	3	5	3	6	4
80	4	3	5	3	5	4
70	4	3	4	3	5	4

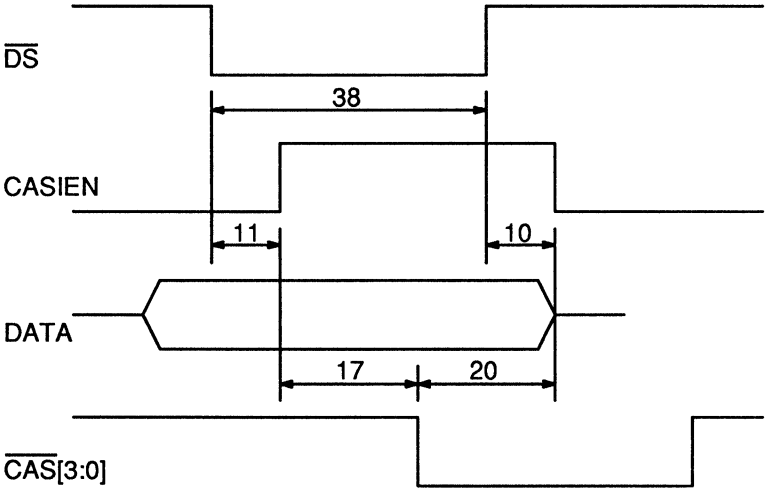


Figure 6. Byte Enable Timing Diagram

PAL Equations

The following are the logic equations for the three PAL devices. They are written in PLPL.

```
"February 15, 1988
Byte Decoder PAL.  This PAL generates the byte enable signals for the Am29C668.
It will also generate the Write Enable Signals to the DRAMs and the CASIEN signal
to the Am29C668.

Note:  BE[0] is the same as UUD, BE[1] = UMD, BE[2] = LMD, BE[3] = LLD in Motorola
notation."

DEVICE BYTE_DECODER (P16L8)

PIN A[1:0] = 1:2 (INPUT COMBINATORIAL)
     S1 = 3 (INPUT COMBINATORIAL)
     S0 = 4 (INPUT COMBINATORIAL)
     /AS = 5 (INPUT COMBINATORIAL)
     /DS = 6 (INPUT COMBINATORIAL)
     R_WL = 8 (INPUT COMBINATORIAL)
     /MEMREQ = 9 (INPUT COMBINATORIAL)
     PROGRAM = 11 (INPUT COMBINATORIAL)
     /WE[1:0] = 19:18 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     /BE[3:0] = 17:14 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     RL = 13 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     CASIEN = 12 (OUTPUT ACTIVE_LOW COMBINATORIAL);

BEGIN
ENABLE (BE[3:0], RL, WE[1:0], DSACKEN);

BE[0] = /A[1] * /A[0];
BE[1] = /A[1] * A[0] + /A[1] * /S0 + /A[1] * S1;
BE[2] = A[1] * /A[0] + /A[1] * /S1 * /S0 +
  /A[1] * S1 * S0 + /A[1] * A[0] * S1;
BE[3] = /S1 * /S0 + A[1] * A[0] + A[1] * S1 +
  S1 * S0 * A[0];

/RL = PROGRAM * AS;

WE[1] = R_WL;
WE[0] = R_WL;

/CASIEN = MEMREQ * DS;

END.
TEST_VECTORS

IN A[1:0],S1,S0;
OUT BE[3:0];

BEGIN
"
A A S S        B B B B
               E E E E
1 0 1 0        3 2 1 0
------------------------"
0 0 0 1        H L L L;  "BYTE ACCESSES"
0 1 0 1        L H L L;
1 0 0 1        L L H L;
1 1 0 1        L L L H;
0 0 1 0        H H L L;  "WORD ACCESSES"
0 1 1 0        L H H L;
1 0 1 0        L L H H;
```

```
1 1 1 0         L L L H;
0 0 1 1         H H H L;  "3-BYTE ACCESSES"
0 1 1 1         L H H H;
1 0 1 1         L L H H;
1 1 1 1         L L L H;
0 0 0 0         H H H H;  "LONG WORD ACCESSES"
0 1 0 0         L H H H;
1 0 0 0         L L H H;
1 1 0 0         L L L H;
END.
```

```
"December 5, 1988

Timer PAL for Am29C668 to 68020 Interface.  This PAL will generate refresh request
for the DRAM controller."

DEVICE   TIMER (P20XRP10)

PIN CLOCK = 1 (CLOCK)
     RESET = 2 (INPUT COMBINATORIAL)
     /RFRQ = 3 (INPUT COMBINATORIAL)
     /OE = 13 (CONTROL)
     /RCT[8:0] = 23:15 (OUTPUT ACTIVE_LOW REGISTERED)
     /REFRESH = 14 (OUTPUT ACTIVE_LOW REGISTERED);

DEFINE
START_REFRESH = /RCT[8] * /RCT[7] * /RCT[6] * /RCT[5] * /RCT[4] *
    /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0],
INIT = /RCT[8] * /RCT[7] * /RCT[6] * /RCT[5] * /RCT[4] * /RCT[3] *
    /RCT[2] * /RCT[1] * /RCT[0];

BEGIN
REFRESH := START_REFRESH * /RESET + REFRESH * /RFRQ * /RESET;

RCT[0] := /RCT[0];
XOR(RCT[0]) := INIT;
RCT[1] :=/RCT[0];
XOR(RCT[1]) := RCT[1] + INIT;
RCT[2] :=/RCT[1] * /RCT[0];
XOR(RCT[2]) := RCT[2] + INIT;
RCT[3] :=/RCT[2] * /RCT[1] * /RCT[0];
XOR(RCT[3]) := RCT[3]  + INIT;
RCT[4] :=/RCT[3] * /RCT[2] * /RCT[1] * /RCT[0] + INIT;
XOR(RCT[4]) := RCT[4];
RCT[5] :=/RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0]  + INIT;
XOR(RCT[5]) := RCT[5];
RCT[6] :=/RCT[5] * /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0];
XOR(RCT[6]) := RCT[6] + INIT;
RCT[7] :=/RCT[6] * /RCT[5] * /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] * /RCT[0];
XOR(RCT[7]) := RCT[7] + INIT;
RCT[8] :=/RCT[7] * /RCT[6] * /RCT[5] * /RCT[4] * /RCT[3] * /RCT[2] * /RCT[1] *
    /RCT[0];
XOR(RCT[8]) := RCT[8];

END.
TEST_VECTORS
IN CLOCK, RESET, OE, RFRQ;
OUT RCT [8:0], REFRESH;
BEGIN
"
C R                        R
L E   R     R R R R R R R R F
O S   F     C C C C C C C C R
C E O R     T T T T T T T T S
K T E Q     8 7 6 5 4 3 2 1 0 H
---------------------------------"
0 0 1 0     L L L L L L L L L;
C 1 1 0     H L L H H L L L L;  "RESET THE TIMER"
C 0 1 0     H L L H L H H H H;  "INITIAL COUNT VALUE"
C 0 1 0     H L L H L H H H L;  "DECREMENTS"
C 0 1 0     H L L H L H H L H;
P 0 0 0     0 0 0 0 0 0 0 1 1;  "PRELOAD VALUE TO TEST"
```

```
0 0 1 0     H H H H H H H L L L;   "THAT COUNTER DECREMENTS"
C 0 1 0     H H H H H H L H H L;   "AT BOUNDARIES"
P 0 0 0     0 0 0 0 0 0 1 1 1 1;   "PRELOAD VALUE TO TEST"
0 0 1 0     H H H H H H L L L L;   "THAT COUNTER DECREMENTS"
C 0 1 0     H H H H H L H H H L;   "AT BOUNDARIES"
P 0 0 0     0 0 0 0 0 1 1 1 1 1;   "PRELOAD VALUE TO TEST"
0 0 1 0     H H H H H L L L L L;   "THAT COUNTER INCREMENTS"
C 0 1 0     H H H H L H H H H L;   "AT BOUNDARIES"
P 0 0 0     0 0 0 0 1 1 1 1 1 1;   "PRELOAD VALUE TO TEST"
0 0 1 0     H H H H L L L L L L;   "THAT COUNTER DECREMENTS"
C 0 1 0     H H H L H H H H H L;   "AT BOUNDARIES"
P 0 0 0     0 0 0 1 1 1 1 1 1 1;
0 0 1 0     H H H L L L L L L L;
C 0 1 0     H H L H H H H H H L;
P 0 0 0     0 0 1 1 1 1 1 1 1 1;
0 0 1 0     H H L L L L L L L L;
C 0 1 0     H L H H H H H H H L;
P 0 0 0     0 1 1 1 1 1 1 1 1 1;
0 0 1 0     H L L L L L L L L L;
C 0 1 0     L H H H H H H H H L;
P 0 0 0     1 1 1 1 1 1 1 1 0 1;   "PRELOAD #1"
0 0 1 0     L L L L L L L L H L;
C 0 1 0     L L L L L L L L L L;   "TEST THE INIT SIGNAL"
C 0 1 0     H L L H H L L L L H;
C 0 1 1     H L L H L H H H H L;
```

```
"February 15, 1988
State PAL for Am29C668 to 68020 Interface.  This PAL contains the state machine
that controls the memory.  It will arbitrate between refresh and memory accesses
and will generate the control signals for the Am29C668 and 68020 processor."

DEVICE STATE (P22V10)

PIN  CLOCK = 1 (CLOCK)
     CLK = 2 (INPUT COMBINATORIAL)
     /AS = 3 (INPUT COMBINATORIAL)
     /DS = 4 (INPUT COMBINATORIAL)
     /MEMREQ = 5 (INPUT COMBINATORIAL)
     /REFRESH = 6 (INPUT COMBINATORIAL)
     PROGRAM = 7 (INPUT COMBINATORIAL)
     RESET = 8 (INPUT COMBINATORIAL)
     /CH = 9 (INPUT COMBINATORIAL)
     /DSACKEN = 23 (OUTPUT_ACTIVE_LOW COMBINATORIAL)
     RASI = 15 (OUTPUT ACTIVE_HIGH COMBINATORIAL)
     /PGHIT = 22 (OUTPUT ACTIVE_LOW COMBINATORIAL)
     STATE[2:0] = 21:19 (OUTPUT ACTIVE_HIGH REGISTERED)
     /RFRQ = 18 (OUTPUT ACTIVE_LOW REGISTERED)
     /DSACK[1:0] = 17:16 (OUTPUT ACTIVE_LOW COMBINATORIAL);

DEFINE IDLE =    /STATE[2] * /STATE[1] * /STATE[0],
     SW1 =     /STATE[2] * /STATE[1] *  STATE[0],
     SW2 =     /STATE[2] *  STATE[1] *  STATE[0],
     ACC =      STATE[2] *  STATE[1] *  STATE[0],
     PM =       STATE[2] * /STATE[1] *  STATE[0],
     PC1 =      STATE[2] *  STATE[1] * /STATE[0],
     PC12 =     STATE[2] * /STATE[1] * /STATE[0],
     PC2 =     /STATE[2] *  STATE[1] * /STATE[0],
     IDLE_ST = #b000,
     SW1_ST =  #b001,
     SW2_ST =  #b011,
     ACC_ST =  #b111,
     PM_ST =   #b101,
     PC1_ST =  #b110,
     PC2_ST =  #b010,
     PC12_ST = #b100;

BEGIN
ENABLE (RASI, PGHIT, STATE[2:0], RFRQ, DSACKEN);
ENABLE (DSACK[1:0]) = DSACKEN;

RASI = IDLE * RFRQ * /CLK +
  IDLE * AS * MEMREQ * CLK * /REFRESH + IDLE * RASI +
  SW1 + SW2 + ACC + PM * /(/CH * AS * MEMREQ * CLK) * RASI;

DSACKEN = (MEMREQ + PROGRAM) * DS;

PGHIT = PM * AS * MEMREQ * CH * CLK * RASI +
  PM * PGHIT * /CLK;

RFRQ = /RESET * REFRESH * /PROGRAM * (IDLE * /RASI + PC1 + PC12) +
  RFRQ * /RESET * /PC1;

DSACK[1] = SW2 * /RFRQ * /CLK + ACC * /RFRQ + PM * /CLK * PGHIT +
  PROGRAM * AS;
DSACK[0] = SW2 * /RFRQ * /CLK + ACC * /RFRQ + PM * /CLK * PGHIT +
  PROGRAM * AS;
```

```
IF (RESET) THEN BEGIN
  STATE[2:0] := IDLE_ST;
  RFRQ := 0;
END;

CASE (STATE[2:0]) BEGIN
  IDLE_ST) IF (RASI) THEN BEGIN
      STATE[2:0] := SW1_ST;
      END;
      ELSE BEGIN
          STATE[2:0] := IDLE_ST;
      END;
  SW1_ST) BEGIN
      STATE[2:0] := SW2_ST;
      END;
  SW2_ST) BEGIN
      STATE[2:0] := ACC_ST;
      END;
  ACC_ST) BEGIN
      IF (REFRESH + RFRQ) THEN STATE[2:0] := PC1_ST;
      ELSE STATE[2:0] := PM_ST;
      END;
  PM_ST) BEGIN
      IF (PGHIT) THEN STATE[2:0] := ACC_ST;
      ELSE IF (/RASI) THEN
          STATE[2:0] := PC12_ST;
      ELSE IF (REFRESH) THEN BEGIN
          STATE[2:0] := PC12_ST;
      END;
      ELSE STATE[2:0] := PM_ST;
      END;
  PC1_ST) BEGIN
      STATE[2:0] := PC2_ST;
      END;
  PC12_ST) BEGIN
      IF (REFRESH) THEN STATE[2:0] := PC2_ST;
      ELSE STATE[2:0] := IDLE_ST;
      END;
  PC2_ST) BEGIN
      IF (RFRQ) THEN STATE[2:0] := SW1_ST;
      ELSE STATE[2:0] := IDLE_ST;
      END;
END; "CASE"

END.
TEST_VECTORS

IN CLK,CLOCK,AS,DS,MEMREQ,REFRESH,PROGRAM,RESET,CH,DSACKEN;

OUT RASI,PGHIT,RFRQ,STATE[2:0],DSACK[1:0];

BEGIN
"              R   D       P         D D
  C     M R P E   S     R G R S S S S S
C L     R F R S   K     A H F T T T C C
L C A D E S G E C E     S I R       K K
K K S S Q H M T H N     I T Q 2 1 0 1 0
-----------------------------------------"
C C 0 0 0 0 0 1 X 0     L L L L L L Z Z;  "IDLE STATE"
C C 1 1 0 0 1 0 X 1     L L L L L L H H;  "PROGRAM CYCLE"
C C 1 1 0 0 1 0 X 1     L L L L L L H H;
```

```
0 0 0 0 0 0 0 0 X 0     L L L L L L Z Z;  "END OF PROGRAM CYCLE"
C C 0 0 0 0 0 0 X 0     L L L L L L Z Z;  "IDLE CYCLE"
0 0 1 1 1 0 0 0 X 1     L L L L L L L L;  "START OF ACCESS"
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;
0 0 1 1 1 0 0 0 X 1     H L L L L L L L;
C C 1 1 1 0 0 0 X 1     H L L L L H L L;  "STATE SW1"
1 1 1 1 1 0 0 0 X 1     H L L L H H L L;  "STATE SW2"
0 0 1 1 1 0 0 0 X 1     H L L L H H H H;  "STATE SW2"
1 1 1 1 1 0 0 0 X 1     H L L H H H H H;  "ACCESS"
0 0 1 1 1 0 0 0 X 1     H L L H H H H H;
0 0 0 0 1 0 0 0 X 0     H L L H H H Z Z;  "ACCESS TERMINATED"
C C 0 0 1 0 0 0 X 0     H L L H L H Z Z;  "PAGE MODE STATE"
0 0 1 1 1 0 0 0 1 1     H L L H L H L L;  "START OF ACCESS"
1 1 1 1 1 0 0 0 1 1     H H L H L H L L;
0 0 1 1 1 0 0 0 1 1     H H L H L H H H;
1 1 1 1 1 0 0 0 1 1     H L L H H H H H;  "ACCESS"
0 0 1 1 1 0 0 0 1 1     H L L H H H H H;
0 0 0 0 1 0 0 0 1 0     H L L H H H Z Z;
C C 0 0 1 0 0 0 X 0     H L L H L H Z Z;
1 1 1 1 1 0 0 0 X 1     H H L H L H L L;  "ACCESS REQUEST"
1 1 1 1 1 1 0 0 X 1     H H L H L H L L;  "REFRESH REQUEST"
0 0 1 1 1 1 0 0 X 1     H H L H L H H H;  "ACCESS"
1 1 1 1 1 1 0 0 1 1     H L L H H H H H;  "ACCESS"
0 0 0 0 1 1 0 0 1 0     H L L H H H Z Z;  "ACCESS"
C C 0 0 1 1 0 0 X 0     L L L H H L Z Z;  "PC1 STATE"
C C 1 1 0 1 0 0 X 0     L L H L H L Z Z;  "PC2 STATE"
C C 1 1 0 1 0 0 X 0     H L H L L H Z Z;  "SW1 STATE"
C C 1 1 0 0 0 0 X 0     H L H L H H Z Z;  "SW2 STATE"
C C 1 1 0 0 0 0 X 0     H L H H H H Z Z;  "ACC STATE"
C C 1 1 0 0 0 0 X 0     L L H H H L Z Z;  "PC1"
C C 1 1 0 0 0 0 X 0     L L L L H L Z Z;  "PC2"
C C 1 1 0 0 0 0 X 0     L L L L L L Z Z;  "IDLE"
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;
1 1 1 1 1 1 0 0 X 1     H L L L L L L L;
0 0 1 1 1 1 0 0 X 1     H L L L L L L L;
C C 1 1 1 1 0 0 X 1     H L L L L H L L;  "STATE SW1"
1 1 1 1 1 1 0 0 X 1     H L L L H H L L;  "STATE SW2"
0 0 1 1 1 1 0 0 X 1     H L L L H H H H;  "STATE SW2"
1 1 1 1 1 1 0 0 X 1     H L L H H H H H;  "ACCESS"
0 0 1 1 1 1 0 0 X 1     H L L H H H H H;
C C 0 0 1 1 0 0 X 0     L L L H H L Z Z;  "PC1 STATE"
C C 0 0 1 1 0 0 X 0     L L H L H L Z Z;  "PC2 STATE"
C C 0 0 0 1 0 0 X 0     H L H L L H Z Z;  "SW1 STATE"
C C 0 0 0 0 0 0 0 X 0   H L H L H H Z Z;  "SW2 STATE"
C C 0 0 0 0 0 0 0 X 0   H L H H H H Z Z;  "ACC STATE"
C C 0 0 0 0 0 0 0 X 0   L L H H H L Z Z;  "PC1"
C C 0 0 0 0 0 0 0 X 0   L L L L H L Z Z;  "PC2"
C C 0 0 0 0 0 0 0 X 0   L L L L L L Z Z;  "IDLE"
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;
1 1 1 1 1 1 0 0 X 1     H L L L L L L L;
0 0 1 1 1 1 0 0 X 1     H L L L L L L L;
C C 1 1 1 1 0 0 X 1     H L L L L H L L;  "STATE SW1"
1 1 1 1 1 1 0 0 X 1     H L L L H H L L;  "STATE SW2"
0 0 1 1 1 1 0 0 X 1     H L L L H H H H;  "STATE SW2"
1 1 1 1 1 1 0 0 X 1     H L L H H H H H;  "REFRESH ACCESS"
0 0 1 1 1 1 0 0 X 1     H L L H H H H H;
0 0 0 0 1 1 0 0 1 0     H L L H H H Z Z;
C C 0 0 1 1 0 0 X 0     L L L H H L Z Z;  "PC1"
C C 0 0 1 1 0 0 X 0     L L H L H L Z Z;  "PC2"
C C 0 0 0 1 0 0 X 0     H L H L L H Z Z;  "SW1 STATE"
C C 0 0 0 0 0 0 0 X 0   H L H L H H Z Z;  "SW2 STATE"
C C 0 0 0 0 0 0 0 X 0   H L H H H H Z Z;  "ACC STATE"
```

```
C C 0 0 0 0 0 0 X 0     L L H H H L Z Z;  "PC1"
C C 0 0 0 0 0 0 X 0     L L L L H L Z Z;  "PC2"
C C 0 0 0 0 0 0 X 0     L L L L L L Z Z;  "IDLE"
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;
1 1 1 1 1 1 0 0 X 1     H L L L L L L L;
0 0 1 1 1 1 0 0 X 1     H L L L L L L L;
C C 1 1 1 1 0 0 X 1     H L L L L H L L;  "STATE SW1"
1 1 1 1 1 1 0 0 X 1     H L L L H H L L;  "STATE SW2"
0 0 1 1 1 1 0 0 X 1     H L L L H H H H;  "STATE SW2"
1 1 1 1 1 1 0 0 X 1     H L L H H H H H;  "ACCESS"
0 0 1 1 1 1 0 0 X 1     H L L H H H H H;
0 0 0 0 1 0 0 0 1 0     H L L H H H Z Z;
C C 0 0 1 0 0 0 X 0     H L L H L H Z Z;  "PAGE MISS"
0 0 1 1 1 0 0 0 0 1     H L L H L H L L;
1 1 1 1 1 0 0 0 0 1     L L L H L H L L;
1 1 1 1 1 0 0 0 1 1     L L L H L H L L;
C C 1 1 1 0 0 0 1 1     L L L H L L L L;  "PC12"
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;  "IDLE"
0 0 1 1 1 0 0 0 X 1     H L L L L L L L;
1 1 1 1 1 0 0 0 X 1     H L L L L H L L;  "STATE SW1"
0 0 1 1 1 0 0 0 X 1     H L L L L H L L;
1 1 1 1 1 0 0 0 X 1     H L L L H H L L;  "STATE SW2"
0 0 1 1 1 0 0 0 X 1     H L L L H H H H;  "STATE SW2"
1 1 1 1 1 0 0 0 X 1     H L L H H H H H;  "ACCESS"
0 0 1 1 1 0 0 0 X 1     H L L H H H H H;
0 0 0 0 1 0 0 0 X 0     H L L H H H Z Z;  "ACCESS TERMINATED"
C C 0 0 1 0 0 0 X 0     H L L H L H Z Z;  "PAGE MODE STATE"
C C 0 0 1 1 0 0 X 0     L L L H L L Z Z;  "REFRESH REQUEST, PC12"
C C 0 0 1 1 0 0 X 0     L L H L H L Z Z;  "PC2"
C C 0 0 1 1 0 0 X 0     H L H L L H Z Z;  "SW1"
C C 0 0 1 0 0 0 X 0     H L H L H H Z Z;  "SW2"
C C 0 0 1 0 0 0 X 0     H L H H H H Z Z;  "ACC"
C C 0 0 1 0 0 0 X 0     L L H H H L Z Z;  "PC1"
C C 0 0 1 0 0 0 X 0     L L L L H L Z Z;  "PC2"
C C 0 0 1 0 0 0 X 0     L L L L L L Z Z;  "IDLE"
1 1 0 0 1 0 0 0 X 0     L L L L L L Z Z;
1 1 0 0 1 1 0 0 X 0     L L L L L L Z Z;  "REFRESH AND ACCESS"
0 0 1 1 1 1 0 0 X 1     L L L L L L L L;  "DURING SAME CYCLE"
1 1 1 1 1 1 0 0 X 1     L L H L L L L L;  "RFRQ ASSERTED"
0 0 1 1 1 1 0 0 X 1     H L H L L L L L;  "RASI ASSERTED"
C C 1 1 1 0 0 0 X 1     H L H L L H L L;
C C 1 1 1 0 0 0 X 1     H L H L H H L L;
C C 1 1 1 0 0 0 X 1     H L H H H H L L;  "RFRQ COMPLETES"
C C 1 1 1 0 0 0 X 1     L L H H H L L L;  "/RAS PRECHARGE"
C C 1 1 1 0 0 0 X 1     L L L L H L L L;
1 1 1 1 1 0 0 0 X 1     H L L L L L L L;  "ACCESS STARTS"
0 0 1 1 1 0 0 0 X 1     H L L L L L L L;
C C 1 1 1 0 0 0 X 1     H L L L L H L L;
1 1 1 1 1 0 0 0 X 1     H L L L H H L L;
0 0 1 1 1 0 0 0 X 1     H L L L H H H H;
C C 1 1 1 0 0 0 X 1     H L L H H H H H;  "ACCESS COMPLETE"
C C 0 0 0 0 0 0 X 0     H L L H L H Z Z;
END.
```

CHAPTER 4
EDC Memory-Board Designs

Introduction 4-2
IBM PC-AT Plug-in Memory Card with EDC 4-3
IBM PS/2 12-Mbyte Memory Board with EDC 4-25

INTRODUCTION

The following application notes describe evaluation boards that demonstrate the capabilities of the Am29C660C 32-bit Error Detection and Correction circuit (EDC) and the Am29C668 4M Configurable Dynamic Memory Controller (CDMC). The first board is IBM PC-AT compatible, the second is Micro-Channel and PS/2 compatible.

As systems require larger and larger memories, it is imperative to protect the memory from soft errors that occur when a single bit is complemented due to noise, alpha particles, or some other event. While single-bit errors are the most common, double- and multiple-bit errors sometimes occur. The Am29C660 EDC detects and corrects all single-bit errors and detects all double- and some multiple-bit errors. The Am29C668 CDMC can control large memories, up to four banks of 4-Mbit DRAMs, and can drive the $\overline{RAS}_n$, $\overline{CAS}_n$ and address lines without external drivers or damping resistors.

IBM PC-AT Plug-in Memory Card with Error Detection and Correction (EDC)

by Douglas Lee, Applications Specialist

INTRODUCTION

This PC-AT compatible evaluation board demonstrates the capabilities the Am29C668 Configurable Dynamic Memory Controller (CDMC) and the Am29C660C 32-bit Error Detection and Correction Circuit (EDC). As memory size and density increase, it becomes more important to protect the memory from soft errors. The EDC detects and corrects all single-bit errors and detects all double and some triple-bit errors. When a word is accessed, it is checked for errors and if an error is found, the corrected data is written back to memory as well as to the data bus. This board also performs memory "scrubbing", which is the detection and correction of single-bit errors during normal refresh cycles hidden from the microprocessor to maintain the integrity of seldom-accessed memory locations. Scrubbing the memory prevents accumulation of single-bit errors which, in turn, avoids most double-bit errors.

Note: an operating system that utilizes the upper 15 Mbytes in the PC-AT* address space, such as Xenix** or OS/2**, is required to effectively utilize the board. A simple memory test is provided.

Detailed schematics are included, beginning on page 17.

Distinctive Characteristics

- Corrects all single-bit errors, detects all double-bit and some triple-bit errors.
- 12 Mbyte of dynamic RAM(I M x I-bit packages).
- Am29C668 CDMC and Am29C660C EDC packaged in plastic leaded chip carriers for maximum component density.
- Supports memory scrubbing during refresh.
- Designed for 10 MHz systems.
- A Dynamic Memory Timing Controller implemented using Programmable Array Logic (PAL™) devices and delay lines.
- System Data Interface consists of two Am29C983s and one Am29C823.
- 32-bit internal data bus with 7-bit check bit and 7-bit syndrome bus.
- Syndrome latch for diagnostic and test purposes.
- Occupies the second through the 13th megabyte PC-AT address space memory block.

*PC-AT is a registered trademark of IBM Corporation.
**Xenix and OS/2 are trademarks of Microsoft Corporation.

DETAILED DESCRIPTION

The primary data paths and functional elements are shown in Figure I. The following discussion describes each section of the block diagram in detail. Components not appearing in the block diagram but existing on the schematic are also discussed.

Edge Card Connectors

This board can only be used in a PC-AT backplane with the dual-connector I/O channel. Interrupt Request IRQ3 is used by the board to signal the detection of a multiple-bit error to the system microprocessor. Jumper W2 can be removed to prevent IRQ3 from being driven onto the backplane. All signals used from the edge connector are listed on page 6.

Memory Decoder

The 20L10B, chip U6, provides memory decoding for the board. It decodes the upper four bits, Local Addresses LA[23:20], to generate the four internal address bits, DMCA[23:20]. Table 1 shows the address translation. This address translation was chosen since most ATs contain 1 Mbyte of RAM on the mother board. A gap from the first to the second megabyte is required because the setup parameters of the host system (non-volatile RAM) are destroyed when the PC ROM routine BIOS detects RAM that was not properly installed by the "setup" program.

Table 1. Address Translation for the Memory Board

LA				DMCA				
23	22	21	20	23	22	21	20	
0	0	1	0	0	0	0	0	1st Row
0	0	1	1	0	0	0	1	
0	1	0	0	0	0	1	0	
0	1	0	1	0	0	1	1	
0	1	1	0	0	1	0	0	2nd Row
0	1	1	1	0	1	0	1	
1	0	0	0	0	1	1	0	
1	0	0	1	0	1	1	1	
1	0	1	0	1	0	0	0	3rd Row
1	0	1	1	1	0	0	1	
1	1	0	0	1	0	1	0	
1	1	0	1	1	0	1	1	

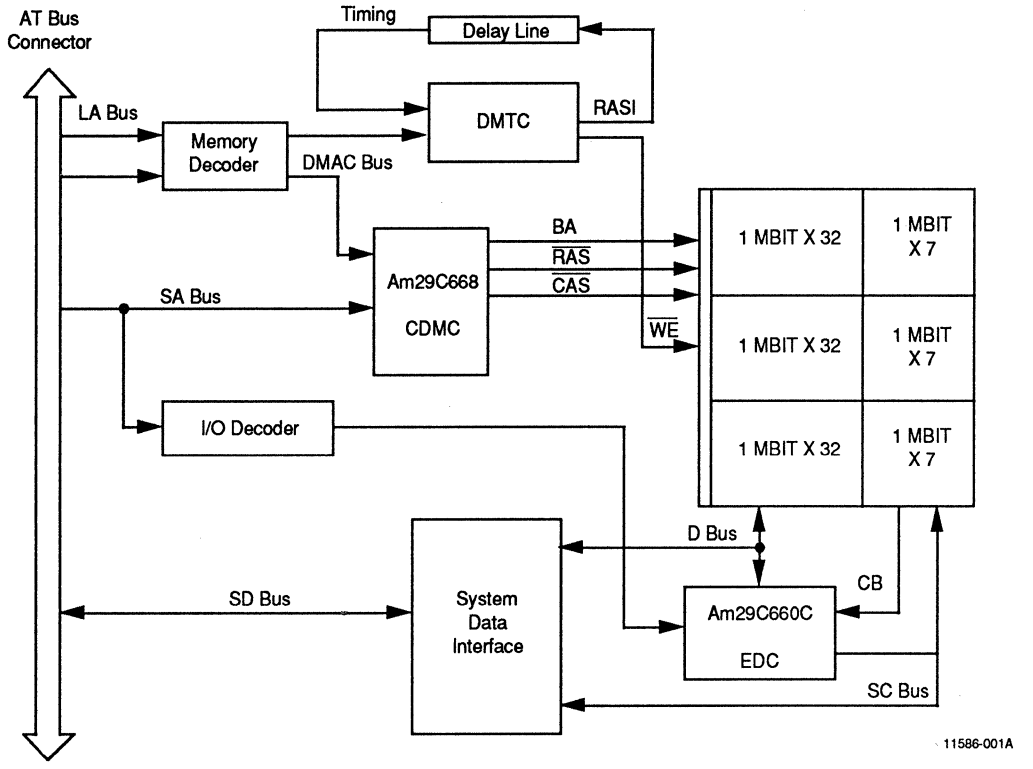


Figure 1. Block Diagram

The PAL U6 also generates the Cycle Request $\overline{\text{CYCREQ}}$ and Read/Write Request $\overline{\text{RWRQ}}$ signals when a valid memory access occurs. $\overline{\text{CYCREQ}}$ remains active until the processor completes its data transfer and $\overline{\text{RWRQ}}$ remains active until the full memory cycle is completed. $\overline{\text{CYCREQ}}$ and $\overline{\text{RWRQ}}$ are latched on Address Bus Latch Enable BALE going Low. The 16-Bit Memory Chip Select $\overline{\text{MEMCS16}}$ is asserted during valid memory accesses and goes to a high-impedance state at all other times. The 16-bit Enable Memory Chip Select $\overline{\text{ENMCS16}}$ controls the output signal of $\overline{\text{MEMCS16}}$.

Refresh Request $\overline{\text{RFRQ}}$ is generated during refresh cycles initiated by the processor driving $\overline{\text{REFRESH}}$ active. The Forced Refresh High FRH signal becomes active when $\overline{\text{REFRESH}}$ is High and is used to detect the falling edge of $\overline{\text{REFRESH}}$. This is required because $\overline{\text{REFRESH}}$ is active longer than the memory refresh cycle.

I/O Channel Ready $\overline{\text{IOCHRDY}}$ is used to signal the processor that valid data is ready during Reads and to signal completions of Writes. $\overline{\text{IOCHRDY}}$ is a 3-state output, enabled by $\overline{\text{CYCREQ}}$ when a valid memory access occurs. This board is designed to work in 10 MHz systems. If the board is used in faster systems (> 12.5 MHz), a faster PAL may be necessary to assert $\overline{\text{IOCHRDY}}$ inactive so that the processor can detect the signal and insert wait states. This timing is machine dependent.

I/O Decoder

The I/O address decoding is provided by a combinatorial PAL, U7, and jumper W1. The Diagnostic Latch Enable LEDIAG and Syndrome Output Enable $\overline{\text{SOE}}$ are generated to select either the diagnostic latch or the syndrome latch respectively. The base address of the I/O ports is selectable by jumper W1 to be 320 h (Hex) or 220 h. Table 2 details this decoding scheme.

Table 2. I/O Decoding Scheme

System Address SA														Jumper	Port	
11	10	9	8	7	6	5	4	3	2	1	0	$\overline{IOW}$	$\overline{IOR}$	W1	Decoded	Address
0	0	1	0	0	0	1	0	0	0	0	0	1	0	in	$\overline{SOE}$	220 h
0	0	1	1	0	0	1	0	0	0	0	0	1	0	out	$\overline{SOE}$	320 h
0	0	1	0	0	0	1	0	0	0	1	0	0	1	in	LEDIAG	222 h
0	0	1	1	0	0	1	0	0	0	1	0	0	1	out	LEDIAG	322 h

Dynamic Memory Timing Control

The timing control for this board was implemented using PALs and delay lines for increased flexibility. The following subsections describe the signals and their functions.

RASI, Mode Control and End of Cycle

The Row Address Strobe Input RASI, Mode Control MC_n and End of Cycle $\overline{EOC}$ signals are generated by U8 in the Control Logic. RASI is used to initiate the timing sequence and to signal the CDMC, Unit 1, to generate the $\overline{RASn}$ signal to the appropriate bank of memory. The $\overline{EOC}$ signal is generated to signify the end of any memory cycle. This signal is used to reset the timing-tap outputs and place the internal logic into the initial state for the next memory access. The timing taps Tn are registered by a 20RA10, Unit 13. This technique provides shorter cycle times. Using a conventional design, it would be necessary to wait for the delay line to clear before the next cycle could start. The $\overline{DONE}$ signal is generated by the asynchronous PAL, U9, in the Control Logic to indicate that the required eight DRAM wake-up cycles have been completed and that the memory is ready for initialization. The MC_1 and MC_0 inputs of the CDMC determine which of the four operating modes will be used. Table 3 shows the decoding.

Table 3. Mode Control Decoding

MC_1	MC_0	Mode
0	0	Refresh without scrubbing.
0	1	Refresh with scrubbing or initialize.
1	0	Read/Write mode.
1	1	Reset.

Latched Error, Initialization and Interrupt -3 Signals

Device U9 generates the $\overline{LATCHED_ERR}$ signal to indicate that an error has occurred on a Read or Read/Modify/Write cycle. This signal is set by the $\overline{EOC}$ signal and is sampled on the rising edge of timing tap T2 . Every memory access is assumed to be a long cycle unless $\overline{LATCHED_ERR}$ is false. This assures correct and concise logic implementation. If $\overline{LATCHED_ERR}$ were conditionally set instead of reset, much more complicated logic would be required, since another signal is needed to indicate when $\overline{LATCHED_ERR}$ is valid.

Device U9 also latches the Initialize $\overline{INIT}$ signal, which is generated by the rising edge of RESET. $\overline{INIT}$ remains active until Terminal Count TC is received from the CDMC indicating that all the memory locations have been initialized. Counter[3:0] and $\overline{DONE}$ count the wake-up cycles and indicate when eight have been completed. Timing tap T8 indicates the end of a wake-up cycle and causes Counter[3:0] to increment. Counter[3:0] is initialized to 0 and, when it reaches 7, all eight cycles have been completed. $\overline{DONE}$ is asserted and the counting is inhibited. $\overline{DONE}$ remains active until $\overline{INIT}$ is deactivated.

Memory-Board Interrupt $\overline{INTR3}$ signals that a multiple error has occurred at T4. $\overline{INTR3}$ goes to Interrupt Request 3 IRQ3 on the backplane via jumper W2 and is cleared by an access to the syndrome latch (signal $\overline{SOE}$). This signal can be disabled by removing Jumper W2. In systems that support bus retry registering, $\overline{INTR3}$ is not required.

Pulsed CASI

The Column Address Strobe Input CASI is a pulsed CAS line, used when connecting the data-in lines to the data-out lines on the DRAMs. The Interface Controller PAL, U11, generates this signal from the registered timing-tap signals received from U13, a 20RA10 PAL. Figure 2 shows how the pulsed CASI signal is produced. This only applies during Read/Modify/Write and Refresh with Scrubbing cycles. Note that at time A in the diagram, the DRAM outputs are 3-stated so that the Am29C660C can drive the data bus.

Miscellaneous Logic Functions

Miscellaneous logic functions are performed by the PAL20L8B Interface Controller, U12. $\overline{WE}[2:0]$ writes the data into the DRAMs. One signal per bank is necessary to drive the large capacitive load (273 pF). These signals have 39 Ω series damping resistors to control over- and under-shoot. Latch Enable Out/Generate LEO_GEN is a dual-purpose signal. When active High, it enables the output latches of the Am29C660C.

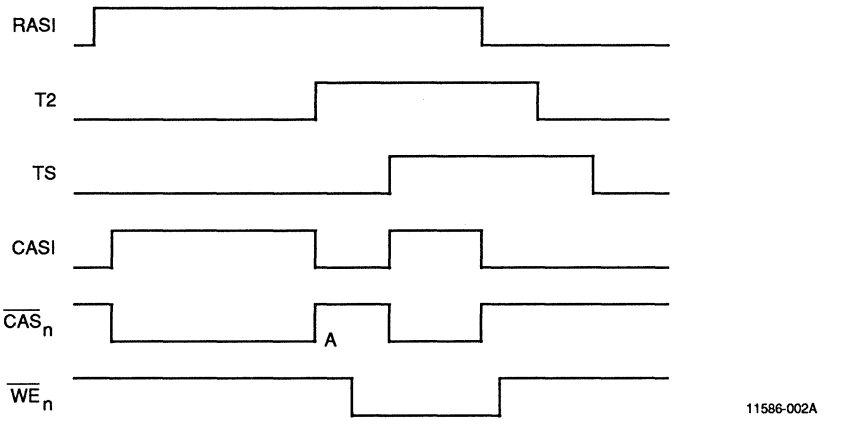


Figure 2. Idealized Timing Diagram for Pulsed CASI Signal

When active Low, it causes check bits to be generated for the data in the input latch of the Am29C660C. The Latch Enable Input LEI controls the latching of data into the 29C660C. When LEI is High, the input latch is transparent; data is latched when LEI is Low. Byte Latch Enable $\overline{LEB}$ controls the data latch from the internal data bus to the system bus. When $\overline{LEB}$ is High, the latch is transparent. S_AND_$\overline{LEB}$ is used to condition the output enables from the system bus.

Delay Lines

The delay lines, U14 through U17, provide the timing reference signals. RASI initiates the timing sequence. See Delay-Line Tap Calculations, page 15, for the tap-timing calculations of the timing configuration currently installed on the board. According to the board timing requirements, each timing signal must be reset at the beginning of each cycle. Device U13 is used to register the timing taps. Since it has separate clocks for each of its 10 output registers, this PAL saves board space over discrete logic. The outputs are reset on $\overline{EOC}$ going active. Using the technique of registering the timing-tap output shortens the memory cycle time.

Interface Control

PAL U10 generates the Output Enable Byte Low $\overline{OEBL}$ and the Output Enable Byte High $\overline{OEBH}$ signals, used by the respective 29C983 Multiple Bus Exchange device, Units 3 and 4, to output the proper data word (16 bits). When address bit SA_1 is Low, the lower word (16-bits) is selected; when High, the upper word is selected. In addition, PAL U10 generates the Output Enable signals, $\overline{OE}$[3:0], to the EDC and to PAL U11 that control byte selection in byte and word writes. SA_{0-1} and $\overline{SBHE}$ are decoded so that the selected byte or word is written to memory.

PAL U11 generates Syndrome Latch Enable $\overline{SYN_EN}$ and Output Byte Enable $\overline{OE_BYTE}$[3:0]. Syndrome Latch Enable $\overline{SYN_EN}$ is an active Low signal that latches the syndrome bits when an error is detected. Device U11 drives the $\overline{OE_BYTE}$[3:0] inputs of the 29C983 latched transceivers that provide the system data-bus interface. These signals gate the AT-bus data lines to and from the various byte-wide data bits of the internal data bus on the board (the D bus). Note from the PAL equations that the gating signals are conditioned by S_AND_$\overline{LEB}$ to ensure proper latching of the data from the DRAMs. LG enables the latches.

Configurable Dynamic Memory Controller and Buffers

The Am29C668 CDMC, U1, supplies the DRAM array with multiplexed row and column address signals,as well as $\overline{RAS}_n$ and $\overline{CAS}_n$ lines. Timing inputs to this device are provided by the delay lines. The Am29C668 is used in the Am29368-compatible mode for I Mbit DRAMs. During Initialization, the Am29C668 generates initialization cycles until the entire memory is written with data and check bits. When the initialization is complete, TC is asserted High. $\overline{RAS}$-only Refresh is used when no memory scrubbing is selected; jumper W3 is connected. Note: AC_{10} is not connected since 1 Mbit DRAMs are used. AC_{10} is the most significant bit of the column latch and would be output on Q_{10}. Since 1 Mbit DRAMs only use Q_{0-9}, this address bit would be lost and the data would be written into the wrong location. 1 Mbit DRAMs use AC_{0-9} for the column address and AR_{0-9} for the row address.

Error Detection and Correction Circuit

Device U2, a high-speed Am29C660C EDC, generates check bits during a write and verifies the data and check bits during a read. Separate error ($\overline{ERR}$) and multiple-bit error ($\overline{MERR}$) signals are output if one or more errors are detected. If $\overline{MERR}$ is asserted, IRQ_3 is active if enabled by jumper W2. Jumper W4 selects input signal CODEID0 to U2. If W4 is installed, a 32-bit slice is selected (the chip may operate in 32- or 64-bit mode); if W2 is not installed, the chip operates in internal-control mode. The internal-control mode provides access to the diagnostic registers in the Am29C660 and aids in debugging and testing the board. See the Am29C660C data sheet for further details.

The Am29C660C internal diagnostic latch is available from the I/O channel. To write the diagnostic register, a word is written from the AT microprocessor to the address selected by the I/O decoder and associated jumper. See I/O Decoder section.

The Output Enable Syndrome/Check Bit $\overline{OESC}$ pin on the EDC is grounded to eliminate the T7-to-OES delay in the PALs and the $\overline{OESC}$-to-SC bus output-enable delay in the Am29C660C; this results in improved performance. This could not be done in an application where the CB bus and SC bus are tied together.

The Am29C660C must initialize the memory to a known state on Reset so that proper check bits are present for byte or word writes. On Reset, the data currently in the input latch is used to initialize the memory. If a known pattern must be written into memory, it may be done after the initialization in software.

System Data Bus Interface

The Am29C983s, U3 and U4, provide byte routing to and from the AT bus and internal D bus. The control signals for these devices are driven by the interface controller PALs, U10, U11 and U12.

Note that in the documentation for the PAL, the equations include a $\overline{MEMW}$ term. This term was included to prevent D-bus contention during a read without error cycle after T2.

Syndrome Latch (Syndrome Logic)

An Am29C823 9-bit latch, U5, stores the check bits when an error is detected by the EDC at timing tap T3. The contents of this latch can then be read by the microprocessor from the I/O channel in an interrupt routine. Decoding the syndrome latch bits reveals information about the error. Refer to the Am29C660C product specification for details. The error signal is qualified by MC1, RASI, and T3 so that the syndrome latch is only updated when errors occur during a Read or Write operation. This makes diagnostics easier and prevents glitches on $\overline{EN}$ of U5.

DRAM Array

The DRAM array consists of three banks of two blocks: the 32-bit data block and 7-bit check-bit block. The array is organized as three rows of 39 bits by 1 Mbit chips, or 117 components. Total user memory is 12 Mbytes. Note that the address space occupied on the AT bus is the 12 Mbytes immediately above the lowest megabyte of the 16 Mbytes available. By using a pulsed $\overline{CAS}$ signal to the chips, the data-in pins can be tied to the data-out pins on the DRAMs. The power pin on the 20-pin zip-pack memory chip is pin 15, the ground pin is 4, and no-connects are 8, 9, and 10.

JUMPER AND CONFIGURATION INFORMATION

A summary of options that can be modified by the user is given below. See the previous text for detailed information concerning these options.

W1: selects the board I/O decode address
(default = in)

W2: IRQ_3 enable/disable to AT I/O channel
(default = in)

W3: determines the RM_2 input to the PAL DMTC
(default = in)

W4: determines $CODEID_0$ input to the EDC
(default = in)

EDGE CONNECTOR PIN NAMES

Pin #	Signal Name	I/O	Pin #	Signal Name	I/O
A1	—	—	B1	GND	—
A2	SD7	I/O	B2	RESET	I
A3	SD6	I/O	B3	+5 VDC	—
A4	SD5	I/O	B4	—	—
A5	SD4	I/O	B5	—	—
A6	SD3	I/O	B6	—	—
A7	SD2	I/O	B7	—	—
A8	SD1	I/O	B8	—	—
A9	SD0	I/O	B9	—	—
A10	IOCHRDY	O	B10	GND	—
A11	AEN	I	B11	—	—
A12	SA19	I	B12	—	—
A13	SA18	I	B13	IOW	I
A14	SA17	I	B14	IOR	I
A15	SA16	I	B15	—	—
A16	SA15	I	B16	—	—
A17	SA14	I	B17	—	—
A18	SA13	I	B18	—	—
A19	SA12	I	B19	REFRESH	I
A20	SA11	I	B20	—	—
A21	SA10	I	B21	—	—
A22	SA9	I	B22	—	—
A23	SA8	I	B23	—	—
A24	SA7	I	B24	—	—
A25	SA6	I	B25	IRQ3	O
A26	SA5	I	B26	—	—
A27	SA4	I	B27	—	—
A28	SA3	I	B28	BALE	I
A29	SA2	I	B29	+5 VDC	—
A30	SA1	I	B30	—	—
A31	SA0	I	B31	GND	—
C1	SBHE	I	D1	MEMCS16	O
C2	LA23	I	D2	IOCS16	O
C3	LA22	I	D3	—	—
C4	LA21	I	D4	—	—
C5	LA20	I	D5	—	—
C6	—	—	D6	—	—
C7	—	—	D7	—	—
C8	—	—	D8	—	—
C9	MEMR	I	D9	—	—
C10	MEMW	I	D10	—	—
C11	SD8	I/O	D11	—	—
C12	SD9	I/O	D12	—	—
C13	SD10	I/O	D13	—	—
C14	SD11	I/O	D14	—	—
C15	SD12	I/O	D15	—	—
C16	SD13	I/O	D16	+5 VDC	—
C17	SD14	I/O	D17	—	—
C18	SD15	I/O	D18	GND	—

Note: A1-A31 and C1-C81 are on the component side, B1-B31 and D1- D18 are on the solder side.

PAL SOURCE CODE LISTINGS

The assembler used is PLPL. Functional test vectors are used to verify these equations, but are not listed to conserve space.

```
DEVICE          IO_DECODER (P20L8)                "U7"

PIN             SA[11:0] = 1:11, 13 (INPUT Combinatorial)
                /IOW = 14 (INPUT Combinatorial)
                /IOR = 23 (INPUT Combinatorial)
                A320_220L = 16 (INPUT Combinatorial)
                AEN = 17 (INPUT Combinatorial)

                /SOE = 22 (OUTPUT Active_Low Combinatorial)
                /LEDIAG = 21 (OUTPUT Active_Low Combinatorial)
                /IOCS16 = 20 (OUTPUT Active_Low Combinatorial)
                /ENIOCS = 19 (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE (SOE,LEDIAG,ENIOCS);
ENABLE (AEN,A320_220L) = 0;
ENABLE (IOCS16) = ENIOCS;

CASE (SA[11:0])
  BEGIN
  #B001000100000) BEGIN
                SOE =  /A320_220L * IOR * /IOW * /AEN;
                IOCS16 = /A320_220L * /AEN * IOR;
                ENIOCS = /A320_220L * /AEN;
                END;
  #B001000100010) BEGIN
                LEDIAG = /A320_220L * IOW * /IOR * /AEN;
                IOCS16 = /A320_220L * /AEN * IOW;
                ENIOCS = /A320_220L * /AEN;
                END;
  #B001100100000) BEGIN
                SOE = A320_220L * IOR * /IOW * /AEN;
                IOCS16 = A320_220L * /AEN * IOR;
                ENIOCS = A320_220L * /AEN;
                END;
  #B001100100010) BEGIN
                LEDIAG = A320_220L * IOW * /IOR * /AEN;
                IOCS16 = A320_220L * /AEN * IOW;
                ENIOCS = A320_220L * /AEN;
                END;
END;

END.
```

```
DEVICE          MEMORY_DECODER3 (P20L10)          "U6"

PIN             LA[23:21] = 1:3 (INPUT Combinatorial)
                /REFRESH = 4 (INPUT Combinatorial)
                /MEMR = 5 (INPUT Combinatorial)
                /MEMW = 6 (INPUT Combinatorial)
                BALE = 7 (INPUT Combinatorial)
                /EOC = 8 (INPUT Combinatorial)
                /INIT = 9 (INPUT Combinatorial)
                T1 = 10 (INPUT Combinatorial)
                T4 = 11 (INPUT Combinatorial)
                /LATCHED_ERR = 13 (INPUT Combinatorial)

                IOCHRDY = 23 (OUTPUT Active_Low Combinatorial)
                /CYCREQ = 22 (OUTPUT Active_Low Combinatorial)
                /MEMCS16 = 21 (OUTPUT Active_Low Combinatorial)
                /ENMCS16 = 20 (OUTPUT Active_Low Combinatorial)
                /RWRQ = 19 (OUTPUT Active_Low Combinatorial)
                /RFRQ = 18 (OUTPUT Active_Low Combinatorial)
                /FRH = 17 (OUTPUT Active_Low Combinatorial)
                DMAC[23:21] = 16:14 (OUTPUT Active_Low Combinatorial);

DEFINE VALID_ADDRESS = LA[22] * /LA[21] + /LA[23] * LA[21] + LA[23] * /LA[22];

BEGIN

ENABLE (DMAC[23:21],CYCREQ,IOCHRDY,ENMCS16,RWRQ,RFRQ,FRH);
ENABLE(MEMCS16) = ENMCS16;  ENABLE (IOCHRDY) = CYCREQ;

/DMAC[21] = /LA[21];        /DMAC[22] = /LA[22] * /LA[21] + LA[22] * LA[21];
/DMAC[23] = LA[23] * LA[22] + LA[23] * LA[21];

CYCREQ = VALID_ADDRESS * BALE * MEMW * /REFRESH * /INIT +
  VALID_ADDRESS * BALE * MEMR * /REFRESH * /INIT +
  /BALE * CYCREQ * (MEMW + MEMR) * /INIT;

MEMCS16 = VALID_ADDRESS * /REFRESH;
ENMCS16 = VALID_ADDRESS * /REFRESH;

/IOCHRDY = (MEMR + MEMW) * CYCREQ *
  (MEMW * T1 * /LATCHED_ERR + T4 * (LATCHED_ERR + /MEMW));

RWRQ = VALID_ADDRESS * BALE * MEMW * /REFRESH * /INIT +
  VALID_ADDRESS * BALE * MEMR * /REFRESH * /INIT +
  RWRQ * (/EOC + CYCREQ) * /INIT;

RFRQ = REFRESH * FRH * /RWRQ * /EOC * /INIT + RFRQ * /EOC * /INIT;

FRH = /(REFRESH * /FRH + RFRQ * T4);

END.
```

```
DEVICE          ARBITER (P16L8)                    "U8"

PIN             T2 = 1 (INPUT Combinatorial)
                T8 = 2 (INPUT Combinatorial)
                T9 = 3 (INPUT Combinatorial)
                T10 = 4 (INPUT Combinatorial)
                /RWRQ = 5 (INPUT Combinatorial)
                /RFRQ = 6 (INPUT Combinatorial)
                /LATCHED_ERR = 7 (INPUT Combinatorial)
                /INIT = 8 (INPUT Combinatorial)
                RM2 = 9 (INPUT Combinatorial)
                /MEMW = 11 (INPUT Combinatorial)
                Done = 13 (INPUT Combinatorial)

                /EOC = 18 (OUTPUT Active_Low Combinatorial)
                /RASI = 17 (OUTPUT Active_Low Combinatorial)
                MC1 = 16 (OUTPUT Active_Low Combinatorial)
                MC0 = 15 (OUTPUT Active_Low Combinatorial)
                /R_WL = 14 (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE (RASI,EOC,MC0,MC1,R_WL);
ENABLE (Done) = 0;

EOC = INIT * T8 +
  RFRQ * (/RM2 * T8 + RM2 * T10) +
    RWRQ * /RFRQ *
    (/R_WL * T10 + R_WL * /LATCHED_ERR * T8 +
    R_WL * LATCHED_ERR * T10) +
    EOC * T8;

RASI = INIT * T2 +
  /INIT * /RFRQ * /RWRQ +
    RFRQ * /RM2 * T2 * /MC0 * /MC1 +
    RFRQ * RM2 * T9 * MC0 * /MC1 +
    RWRQ * /RFRQ * R_WL * (/LATCHED_ERR * T2 +
    LATCHED_ERR * T9) +
    RWRQ * /RFRQ * /R_WL * T9;

/MC0 = INIT + RFRQ * RM2 * (/T9 + /RASI);

/MC1 = /(RFRQ * /RM2 * (/T2 + /RASI) +
  RFRQ * RM2 * (/T9 + /RASI) + INIT * DONE);

R_WL = MEMW + R_WL * /EOC;

END.
```

```
DEVICE          MISC (P20L8)                    "U12"

PIN             T2 = 1 (INPUT Combinatorial)
                T3 = 2 (INPUT Combinatorial)
                T6 = 3 (INPUT Combinatorial)
                T7 = 4 (INPUT Combinatorial)
                T9 = 5 (INPUT Combinatorial)
                /CYCREQ = 6 (INPUT Combinatorial)
                /LATCHED_ERR = 7 (INPUT Combinatorial)
                /INIT = 8 (INPUT Combinatorial)
                RM2 = 9 (INPUT Combinatorial)
                R_WL = 10 (INPUT Combinatorial)
                RASI = 11 (INPUT Combinatorial)
                /RWRQ = 13 (INPUT Combinatorial)
                /RFRQ = 23 (INPUT Combinatorial)
                /LATCHED_MERR = 14 (INPUT Combinatorial)

                /WE[2:0] = 22:20 (OUTPUT Active_Low Combinatorial)
                /LEO_GENL = 19 (OUTPUT Active_Low Combinatorial)
                /LEI = 18 (OUTPUT Active_Low Combinatorial)
                LEB = 17 (OUTPUT Active_Low Combinatorial)
                /S_AND_NOT_LEB = 16 (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE (LEB,LEI,LEO_GENL,WE[2:0],S_AND_NOT_LEB);

/LEB = RWRQ * /RFRQ * R_WL * RASI *
  (/T2 * /LATCHED_ERR + /T3 * LATCHED_ERR);

LEI = INIT +  /INIT * /RFRQ * /RWRQ +
  RWRQ * /RASI + RWRQ * T7 * /LEO_GENL * /R_WL +
    RWRQ * T7 * /LEO_GENL * R_WL * LATCHED_ERR +
    RWRQ * T7 * R_WL * /LATCHED_ERR +
    RFRQ * /RM2 + RFRQ * RM2 * /RASI + RFRQ * RM2 * T7 * /LEO_GENL;

LEO_GENL = /INIT *(RFRQ * RM2 +
  RWRQ * /RFRQ * (/R_WL + R_WL * LATCHED_ERR)) * T3 * /T9;

WE[2] = INIT + RFRQ * RM2 * T6 * /T9 * /LATCHED_MERR +
  RWRQ * /RFRQ * (R_WL * LATCHED_ERR + /R_WL) * T6 * /T9 * /LATCHED_MERR;

WE[1] = INIT + RFRQ * RM2 * T6 * /T9 * /LATCHED_MERR +
  RWRQ * /RFRQ * (R_WL * LATCHED_ERR + /R_WL) * T6 * /T9 * /LATCHED_MERR;

WE[0] = INIT + RFRQ * RM2 * T6 * /T9 * /LATCHED_MERR +
  RWRQ * /RFRQ * (R_WL * LATCHED_ERR + /R_WL) * T6 * /T9 * /LATCHED_MERR;

S_AND_NOT_LEB = /INIT * RASI * /T2 * (RWRQ * /RFRQ + RFRQ) +
  RWRQ * /RFRQ * R_WL * RASI * (/T2 * /LATCHED_ERR + /T3 * LATCHED_ERR);

END.
```

```
DEVICE          Output_Enable (P20L8)           "U10"

PIN             T2 = 1 (INPUT Combinatorial)
                T7 = 2 (INPUT Combinatorial)
                T8 = 3 (INPUT Combinatorial)
                T10 = 4 (INPUT Combinatorial)
                SA0 = 5 (INPUT Combinatorial)
                SA1 = 6 (INPUT Combinatorial)
                /SBHE = 7 (INPUT Combinatorial)
                /CYCREQ = 8 (INPUT Combinatorial)
                /LATCHED_ERR = 9 (INPUT Combinatorial)
                /INIT = 10 (INPUT Combinatorial)
                RM2 = 11 (INPUT Combinatorial)
                R_WL = 13 (INPUT Combinatorial)
                /RWRQ = 14 (INPUT Combinatorial)
                /RFRQ = 23 (INPUT Combinatorial)

                /OEBH = 15 (OUTPUT Active_Low Combinatorial)
                /OEBL = 16 (OUTPUT Active_Low Combinatorial)
                /OEH1 = 17 (OUTPUT Active_Low Combinatorial)
                /OEH0 = 18 (OUTPUT Active_Low Combinatorial)
                /OEL1= 19 (OUTPUT Active_Low Combinatorial)
                /OEL0 = 20 (OUTPUT Active_Low Combinatorial)
                /OES = 21 (OUTPUT Active_Low Combinatorial);

DEFINE          B_WL = /SBHE + SA0;

BEGIN

ENABLE(OEBH,OEBL,OEH1,OEH0,OEL1,OEL0);

OEBH = /INIT * RWRQ * /RFRQ * R_WL * T7 * CYCREQ * SA1;

OEBL = /INIT * RWRQ * /RFRQ * R_WL * T7 * CYCREQ * /SA1;

OEL0 = INIT + (RFRQ * RM2 * T2 * /T10) +
  RWRQ * /RFRQ * /R_WL * (B_WL * (SA1 + SA0) + /B_WL * SA1) * T2 * /T10 +
    RWRQ * /RFRQ * R_WL * LATCHED_ERR * T2 * /T10;

OEL1 = INIT + (RFRQ * RM2 * T2 * /T10) +
  RWRQ * /RFRQ * /R_WL * (B_WL * (SA1 + /SA0) + /B_WL * SA1) * T2 * /T10 +
    RWRQ * /RFRQ * R_WL * LATCHED_ERR * T2 * /T10;

OEH0 = INIT + (RFRQ * RM2 * T2 * /T10) +
  RWRQ * /RFRQ * /R_WL * (B_WL * (/SA1 + SA0) + /B_WL * /SA1) * T2 * /T10 +
    RWRQ * /RFRQ * R_WL * LATCHED_ERR * T2 * /T10;

OEH1 = INIT + (RFRQ * RM2 * T2 * /T10) +
  RWRQ * /RFRQ * /R_WL * (B_WL * (/SA1 + /SA0) + /B_WL * /SA1) * T2 * /T10 +
    RWRQ * /RFRQ * R_WL * LATCHED_ERR * T2 * /T10;

END.
```

```
DEVICE          INTERFACE (P20L8)                "U11"

PIN             /OE0 = 1 (INPUT Combinatorial)
                /OE1 = 2 (INPUT Combinatorial)
                /OE2 = 3 (INPUT Combinatorial)
                /OE3 = 4 (INPUT Combinatorial)
                S_AND_NOT_LEB = 5 (INPUT Combinatorial)
                /MEMW = 6 (INPUT Combinatorial)
                LEO_GENL = 7 (INPUT Combinatorial)
                LEDIAG = 8 (INPUT Combinatorial)
                T2 = 9 (INPUT Combinatorial)
                CAS = 10 (INPUT Combinatorial)
                TS = 11 (INPUT Combinatorial)
                /ERR = 13 (INPUT Combinatorial)
                RASI = 23 (INPUT Combinatorial)
                MC1 = 14 (INPUT Combinatorial)
                T3 = 16 (INPUT Combinatorial)

                LG = 22 (OUTPUT Active_Low Combinatorial)
                /OE_BYTE3 = 21 (OUTPUT Active_Low Combinatorial)
                /OE_BYTE2 = 20 (OUTPUT Active_Low Combinatorial)
                /OE_BYTE1 = 19 (OUTPUT Active_Low Combinatorial)
                /OE_BYTE0 = 18 (OUTPUT Active_Low Combinatorial)
                CASI = 17 (OUTPUT Active_Low Combinatorial)
                /SYN_EN = 15 (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE (OE_BYTE0,OE_BYTE1,OE_BYTE2,OE_BYTE3,LG,CASI,SYN_EN);
ENABLE (T3) = 0;

OE_BYTE0 = (/OE0 * S_AND_NOT_LEB * MEMW) + LEDIAG;

OE_BYTE1 = (/OE1 * S_AND_NOT_LEB * MEMW) + LEDIAG;

OE_BYTE2 = /OE2 * S_AND_NOT_LEB * MEMW;

OE_BYTE3 = /OE3 * S_AND_NOT_LEB * MEMW;

/LG = MEMW + LEDIAG;

/CASI = (/T2 * CAS) + (TS * CAS);

SYN_EN = ERR * MC1 * RASI * /T3;

END.
```

```
DEVICE          SAMPLE (P20RA10)                    "U9"

PIN             /PRE_LOAD = 1 (CONTROL)
                /ERR = 2 (INPUT Combinatorial)
                INT2 = 3 (INPUT Combinatorial)
                /MERR = 4 (INPUT Combinatorial)
                INT4 = 5 (INPUT Combinatorial)
                /SOE = 6 (INPUT Combinatorial)
                TC = 7 (INPUT Combinatorial)
                SYS_RESET = 8 (INPUT Combinatorial)
                /EOC = 9 (INPUT Combinatorial)
                T8 = 10 (INPUT Combinatorial)
                /OE = 13 (CONTROL)

                /LATCHED_ERR = 23 (OUTPUT Active_Low Registered)
                /INTR3 = 22 (OUTPUT Active_Low Registered)
                /INIT = 21 (OUTPUT Active_Low Registered)
                /Done = 20 (OUTPUT Active_Low Registered)
                /Counter[0:2] = 19:17 (OUTPUT Active_Low Registered)
                /LATCHED_MERR = 16 (OUTPUT Active_Low Registered);

BEGIN
ENABLE(LATCHED_ERR,INTR3,Done,Counter[0:2],INIT,LATCHED_MERR);

LATCHED_ERR = ERR;              CLOCK_PT(LATCHED_ERR) = INT2;
  PRESET(LATCHED_ERR) = EOC;

INTR3 = MERR + INTR3;           CLOCK_PT(INTR3) = INT4;
  RESET(INTR3) = SOE;
LATCHED_MERR = MERR;            CLOCK_PT(LATCHED_MERR) = INT4;
  RESET(LATCHED_MERR) = EOC;

CLOCK_PT (INIT)= SYS_RESET;     RESET(INIT) = TC * EOC;   INIT = 1;

RESET(Counter[2:0]) = /INIT;    CLOCK_PT(Counter[2:0]) = T8;

IF (/Done = 1) THEN
  CASE (Counter[2:0]) BEGIN
                0) Counter[2:0] = 1;
                1) Counter[2:0] = 2;
                2) Counter[2:0] = 3;
                3) Counter[2:0] = 4;
                4) Counter[2:0] = 5;
                5) Counter[2:0] = 6;
                6) Counter[2:0] = 7;
                7) Counter[2:0] = 0;
  END;

Done = Counter[2] * Counter[1] * Counter[0] + Done * INIT;
  RESET(Done) = /INIT;          CLOCK_PT(Done) = T8;
END.
```

```
DEVICE          TIMER (P20RA10)                    "U13"

PIN             /PRE_LOAD = 1 (CONTROL)
                INT1 = 2 (INPUT Combinatorial)
                INT2 = 3 (INPUT Combinatorial)
                INT3 = 4 (INPUT Combinatorial)
                INT4 = 5 (INPUT Combinatorial)
                INT6 = 6 (INPUT Combinatorial)
                INT7 = 7 (INPUT Combinatorial)
                INT8 = 8 (INPUT Combinatorial)
                INT9 = 9 (INPUT Combinatorial)
                INT10 = 10 (INPUT Combinatorial)
                /EOC = 11 (INPUT Combinatorial)
                /OE = 13 (CONTROL)

                T1  = 23 (OUTPUT Active_Low Registered)
                T2  = 22 (OUTPUT Active_Low Registered)
                T3  = 21 (OUTPUT Active_Low Registered)
                T4  = 20 (OUTPUT Active_Low Registered)
                T6  = 19 (OUTPUT Active_Low Registered)
                T7  = 18 (OUTPUT Active_Low Registered)
                T8  = 17 (OUTPUT Active_Low Registered)
                T9  = 16 (OUTPUT Active_Low Registered)
                T10 = 15 (OUTPUT Active_Low Registered);

BEGIN

ENABLE(T1,T2,T3,T4,T6,T7,T8,T9,T10);

/T1  = 1;       CLOCK_PT(T1) = INT1;            PRESET(T1) = EOC;
/T2  = 1;       CLOCK_PT(T2) = INT2;            PRESET(T2) = EOC;
/T3  = 1;       CLOCK_PT(T3) = INT3;            PRESET(T3) = EOC;
/T4  = 1;       CLOCK_PT(T4) = INT4;            PRESET(T4) = EOC;
/T6  = 1;       CLOCK_PT(T6) = INT6;            PRESET(T6) = EOC;
/T7  = 1;       CLOCK_PT(T7) = INT7;            PRESET(T7) = EOC;
/T8  = 1;       CLOCK_PT(T8) = INT8;            PRESET(T8) = EOC;
/T9  = 1;       CLOCK_PT(T9) = INT9;            PRESET(T9) = EOC;
/T10 = 1;       CLOCK_PT(T10) = INT10;          PRESET(T10) = EOC;

END.
```

DELAY LINE TAP CALCULATIONS

Derivation of the tap outputs is included here. The calculated time is adjusted to the nearest tap of the delay line equal to or greater than the calculated time.

	Calculated Time (ns)
MSEL - RASI to MUX SELECT	
t_{RAH} (DRAM) min	15.0
t_{SKEW} (Qn to $\overline{RAS}$n) 29C668 max	6.0
Total	21.0
$\overline{CAS}$ - RASI to $\overline{CAS}$	
MSEL	21.0
t_{SKEW} ($\overline{CAS}$n to Qn) 29C668 max	-2.0
t_{ASC} DRAM min	0.0
$-t_{PD}$ ($\overline{CAS}$ to CASI) 20L8B min	-6.0
Total	13.0
INT7 - Data Valid to 29C660C	
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	29.0
t_{ACC} DRAM max	120.0
t_{SU} (Data In) 29C660C min	3.0
$-t_{PD}$ (INT7 to T7) 20RA10 min	-7.0
Total	145.0
INT2 - $\overline{ERROR}$ from 29C660C	
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	29.0
t_{ACC} DRAM max	120.0
t_{PD} (Data In to $\overline{ERROR}$) 29C660C max	16.0
t_{SU} ($\overline{ERROR}$) 20RA10 min	13.0
Total	178.0
INT3 - Corrected Data from 29C660C	
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	29.0
t_{ACC} DRAM max	120.0
t_{PD} (Data In to Data Out) 29C660C max	24.0
$-t_{PD}$ (INT3 to T3) 20RA10 min	-7.0
Total	166.0
INT4 - $\overline{MERR}$ from 29C660C and $\overline{IOCHRDY}$ for R/M/W	
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	29.0
t_{ACC} DRAM max	120.0
t_{PD} (Data In to $\overline{MULT}$ ERROR) 29C660C max	20.0
t_{SU} ($\overline{MERR}$) 20RA10 min	13.0
Total	182.0

	Calculated Time (ns)
INT 1 - $\overline{IOCHRDY}$ for Read without Error	
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	29.0
t_{ACC} DRAM max	120.0
t_{PD} (Data In to Data Out) 29C660C max	24.0
t_{PD} (Data Out to System Data) 29C983 max	14.0
$-t_{PD}$ (T1 to IOCHRDY) 20L10B min	-6.0
$-t_{PD}$ (INT1 to T1) 20RA10 min	-7.0
Total	174.0
INT 6 - Corrected Data and Check Bits (R/M/W)	
INT3	166.0
tSKEW (T6 to T3) 20RA10 max	0.5
tPD (T3 to LEO $\overline{GEN}$) 20L8B max	15.0
tPD (LEO $\overline{GEN}$ to SCn) 29C660C max	18.0
tDS DRAM min	0.0
Total	199.5
TS - Pulsed $\overline{CAS}$	
INT6	199.5
t_{PD} (INT6 to T6) 20RA10 max	20.0
t_{PD} (T6 to $\overline{WE}$) 20L8B max	15.0
$-t_{PD}$ (TS to CASI) 20L8B min	-6.0
$-t_{PD}$ (CASI to $\overline{CAS}$n) 29C668 min	-16.0
t_{WCS} DRAM	0.0
Total	228.5
INT9 - End of $\overline{WE}_n$ and RASI (R/M/W)	
t_S	222.5
t_{PD} (TS to CASI) 20L8B max	15.0
t_{PD} (CASI to $\overline{CAS}$n) 29C668 max	16.0
t_{WCH} ($\overline{WE}$ Pulse Width) DRAM min	25.0
$-t_{PD}$ (T9 to $\overline{WE}$) 20L8B min	-6.0
$-t_{PD}$ (INT9 to T9) 20RA10 min	-7.0
Total	261.5
INT8 - End of Read without Error	
INT2	178.0
t_{RP} RAM min	90.0
Total	268.0
INT10 - End of R/M/W Cycle	
INT9	261.5
t_{RP} RAM min	90.0
Total	351.5

Signal	Should Be (ns)	Is (ns)	Note
MSEL	21.0	30.0	
$\overline{CAS}$	22.0	30.0	= MSEL - 8
INT7	145.0	150.0	
INT2	178.0	180.0	
INT3	166.0	170.0	
INT4	182.0	190.0	
INT1	174.0	180.0	
INT6	203.5	210.0	= INT3 + 33.5
t_s	239.0	240.0	= INT6 + 29
INT8	270.0	270.0	= INT2 + 90
INT9	273.0	280.0	= TS + 33
INT10	370.0	370.0	= INT9 + 90

Notes:

1. Taps dependent or related to other taps are indicated with a comment in the "explanation" column.
2. Timing figures are based on Am29C660C data.

TEST SOFTWARE OPERATING INSTRUCTIONS

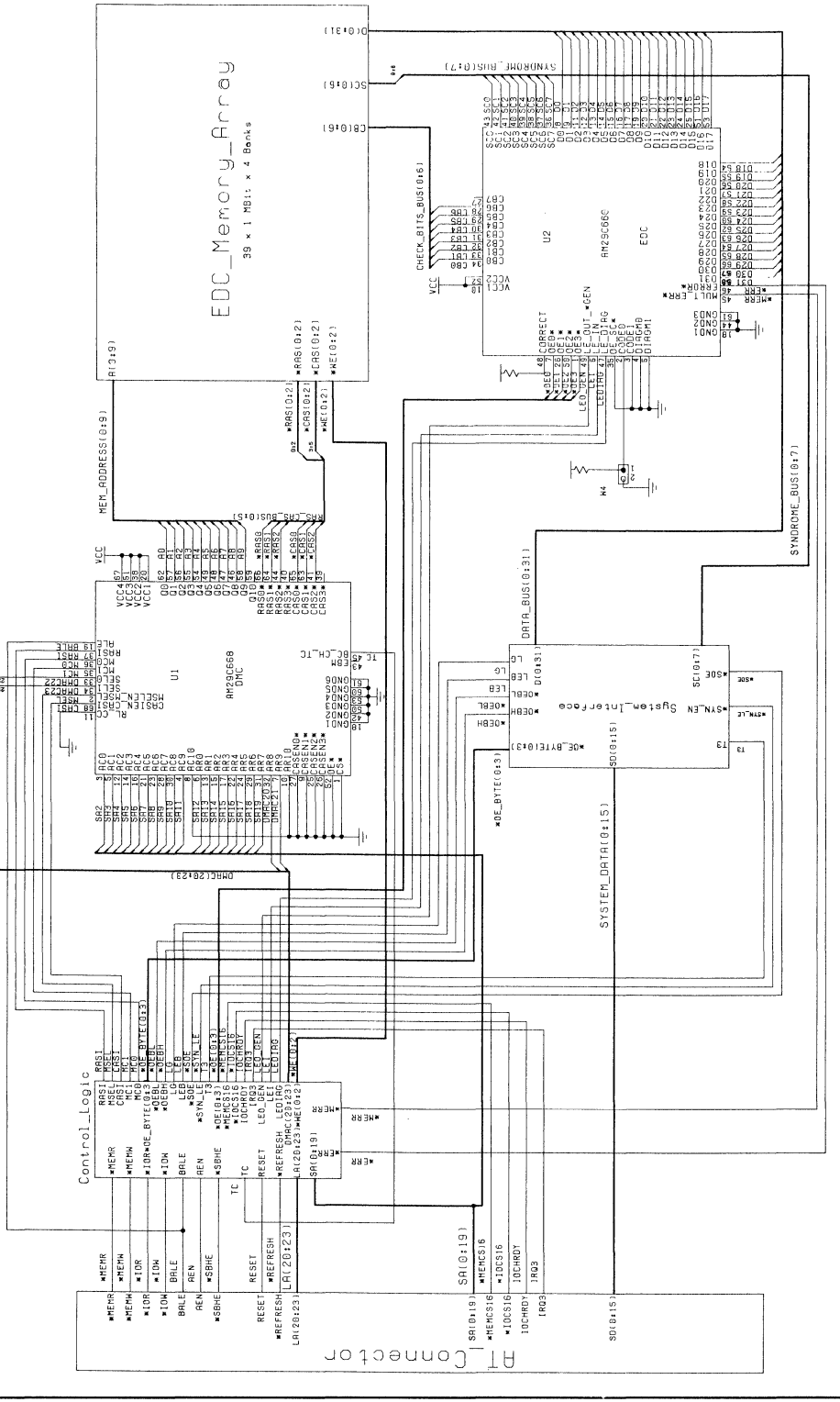
1. Put the "80286 Protected Mode" diskette in drive A and boot it. When "End RAM test" is displayed, remove this diskette out and install the "Test Programs" diskette in drive A. Hit return to get the "#" prompt.
2. Type "msbinin memrwcyc.exe t/n" and (enter). When the display clears and the "#" prompt reappears at the lower left of the screen, type "t" and (enter) to run the program. Selections must be entered exactly as they appear (capitalization is important). Note that the backspace key may be used to correct typing mistakes at the "#" prompt only. (Control-alt-del) also works. (Control-s), (Control-q), (Control-scroll-lock) do not work.
3. To terminate the program, press the (Sys-req) key (the memory test programs return a compare error count and terminate when the space bar is hit; infinite scope loop" programs require the (Sys-req) key to terminate. This returns the "#" prompt to the lower left screen. The program may be stopped and restarted repeatedly. To clean up the display, type "fresh" at the prompt; this also clears out some system tables.
4. Other test programs may be similarly run by replacing the above command line with "msbinin program.exe t/n".

PARTS LIST

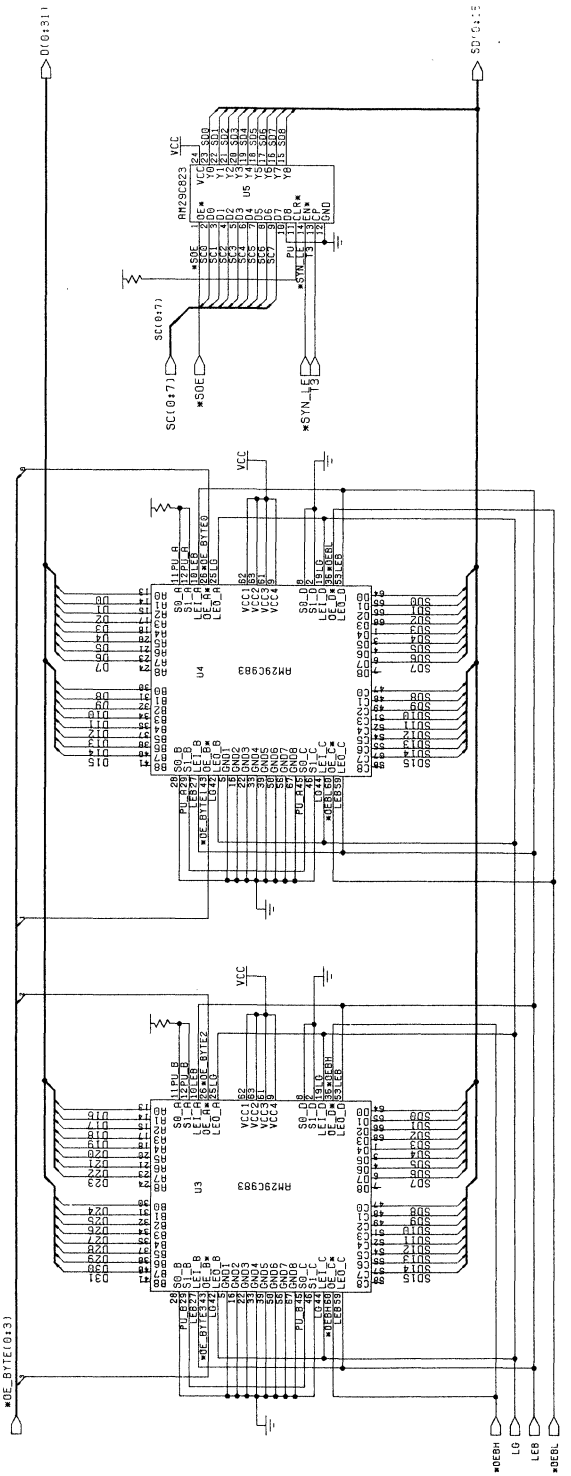
Unit #	Device	Description
U1	Am29C668	Configurable Dynamic Memory Controller
U2	Am29C660C	32-Bit Cascadable Error Detection and Correction Circuit
U3	Am29C983	Multiple Bus Exchange
U4	Am29C983	Multiple Bus Exchange
U5	Am29C823	Syndrome Latch for diagnostic use
U6	Am20L10B	Memory Decoder
U7	Am20L8B	I/O Decoder
U8	Am16L80	Control Logic
U9	Am20RA10	Asynchronous PAL
U10	Am20L8B	Interface Controller
U11	Am20L8B	Interface Controller
U12	Am20L8B	Interface Controller
U13	Am20RA10	Timing Tap Outputs
U14	Delay Line	System Timing Generation
U15	Delay Line	System Timing Generation
U16	Delay Line	System Timing Generation
U17	Delay Line	System Timing Generation

Description	Quantity per Board
CAPACITOR, 22 uF	7
CAPACITOR, 1.0 uF	1
CAPACITOR, 0.33 uF	76
CAPACITOR, 0.1 uF	27
CAPACITOR, 0.01 uF	1
RESISTOR PACK, 10 PIN SIP, 1 k	1
RESITORS, 39 Ω	3
DRAM, ZIP PACK, 1 M x 1	117
AM29C660C	1
AM29C668	1
AM29C983	2
AM29C823	1
AM16L8D	1
AM20L8B	4
AM20L10B	1
AM20RA10-20	2
DELAY LINE, 10 ns, DIP-14	4
SOCKET, 14 PIN DIP	4
SOCKET, 20 PIN DIP	1
SOCKET, 24 PIN DIP	9
SOCKET, 68 PIN PGA/PLCC CONV.	4

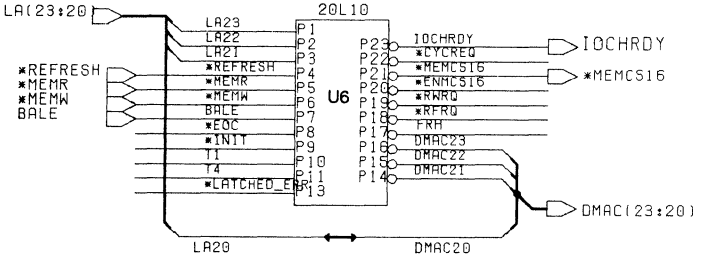
SCHEMATICS

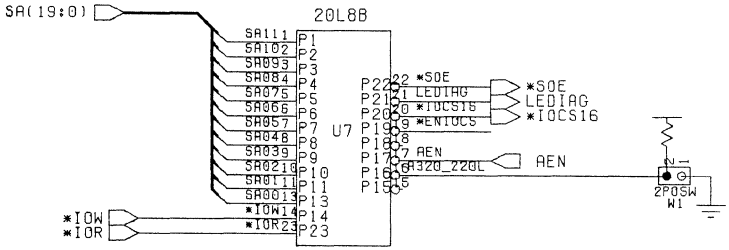


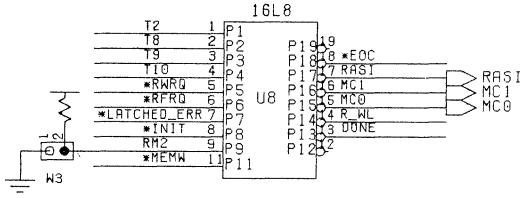
Top Level Schematic

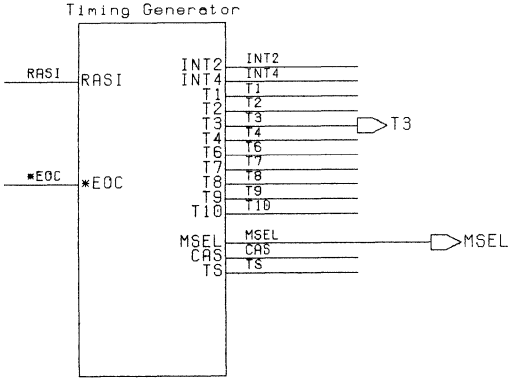


System Data Interface

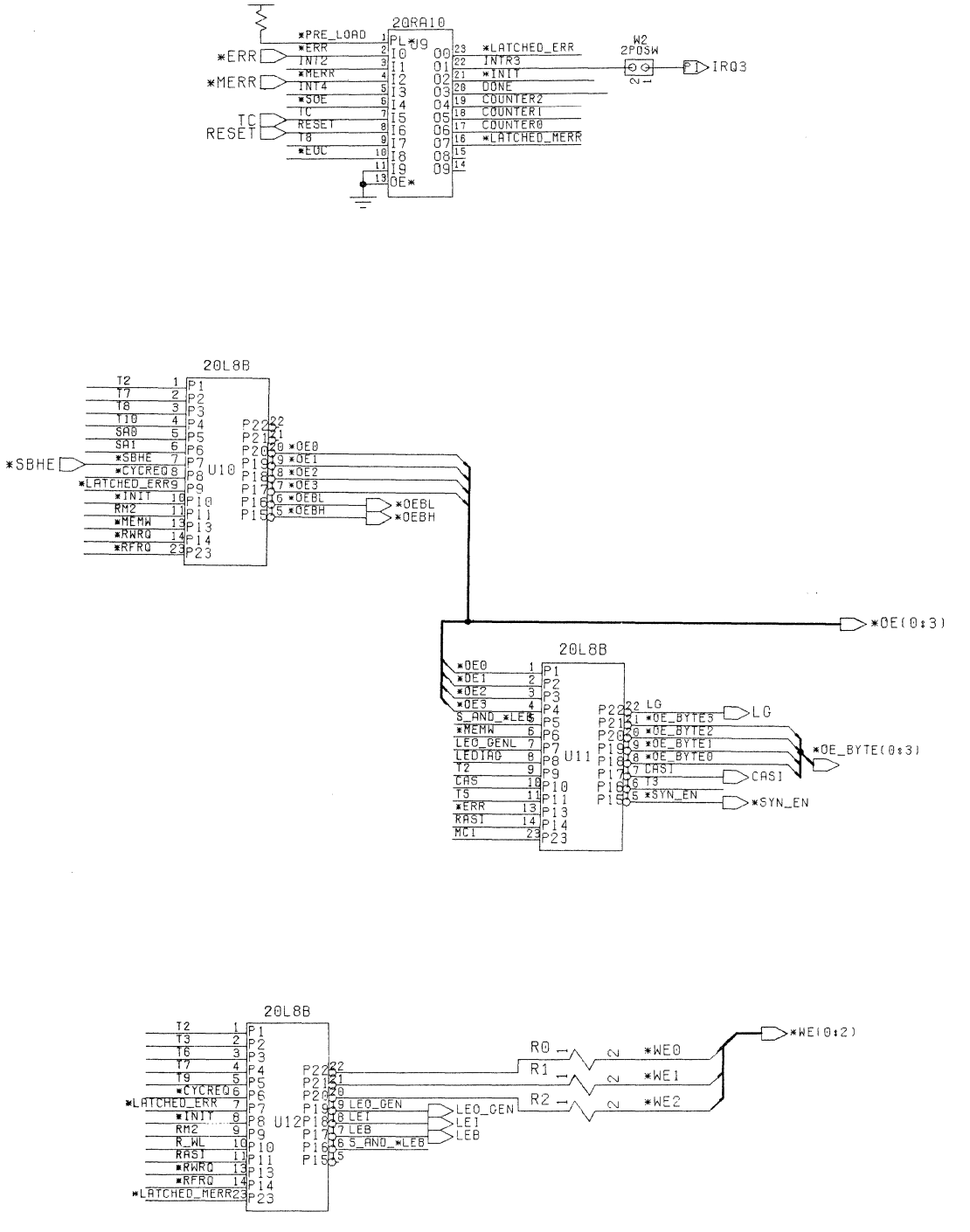




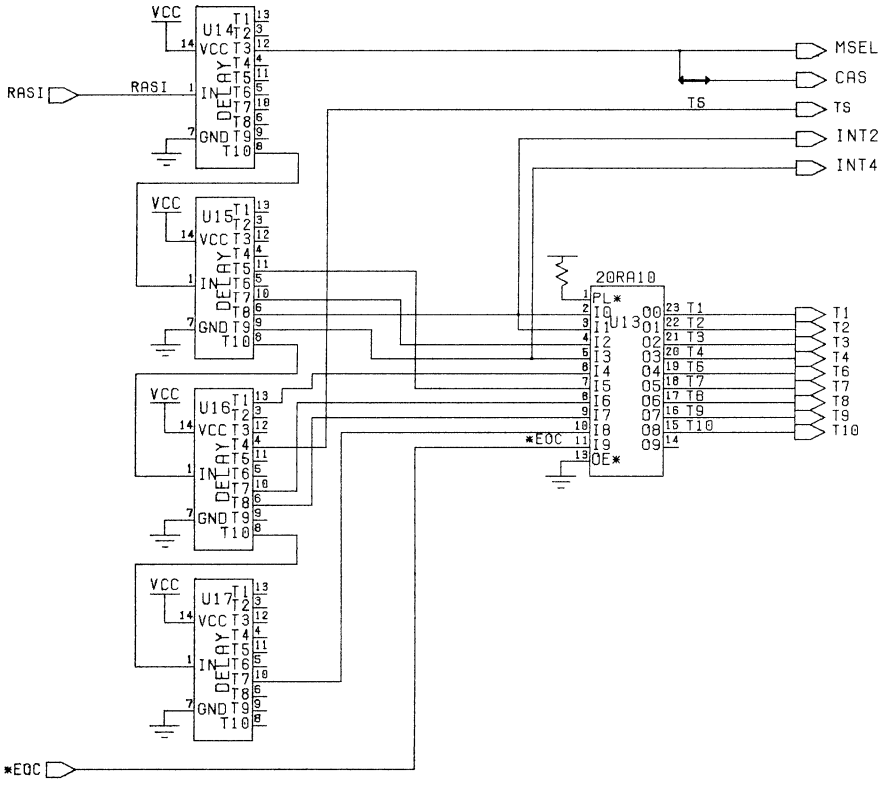




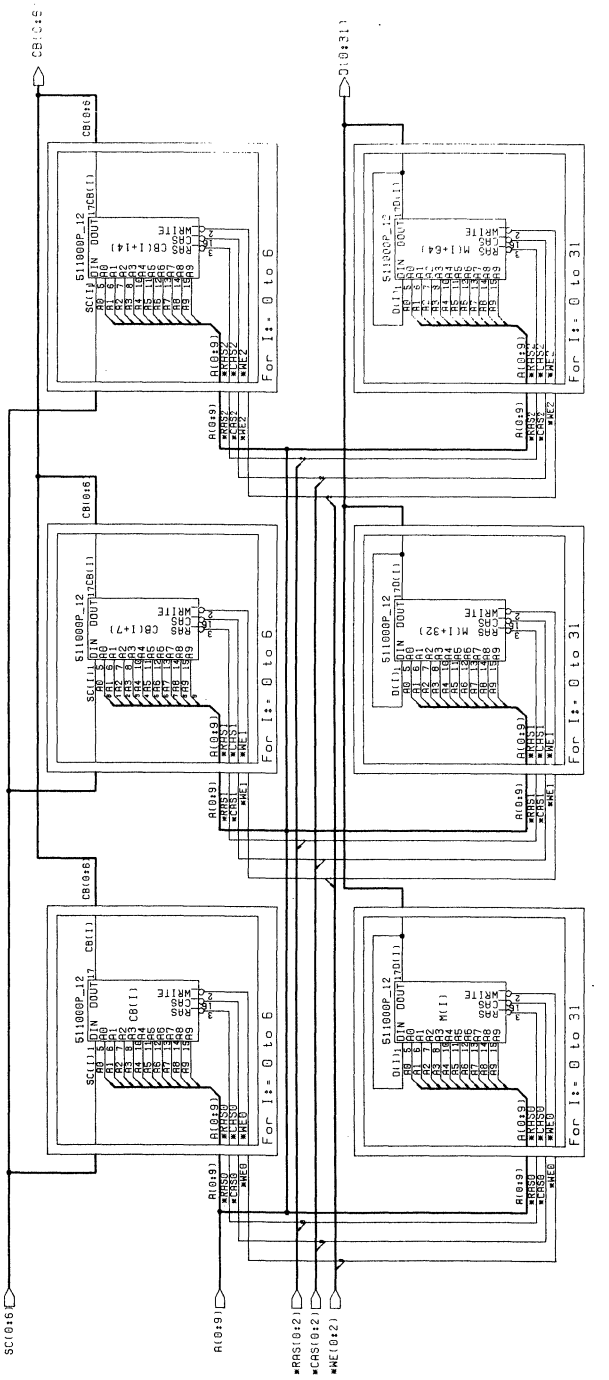
Timing Generation/Control Logic



Interface Controller Logic



Timing Generator



Memory Configuration

IBM PS/2* 12-Mbyte Memory Board with Error Detection and Correction (EDC)

by Douglas Lee, Applications Specialist

INTRODUCTION

This Micro-Channel-compatible evaluation board demonstrates the capabilities of the Am29C668 4M Configurable Dynamic Memory Controller (CDMC) and the Am29C660C Error Detection and Correction Circuit (EDC). As newer systems and software demand much larger memories, it becomes increasingly more important to protect the memory from soft errors, thereby increasing system reliability. Soft errors occur when a single bit is complemented due to noise, alpha particles or some other event. The most common error is a single-bit error where one bit of a memory word is incorrect. The Am29C660 EDC detects and corrects all single-bit errors and detects all double and some multiple-bit errors. When a word is accessed, it is checked for errors; if an error is found, the corrected data is written back. This board also performs memory "scrubbing," which is the detection and correction of single-bit errors during refresh to maintain the integrity of seldom-accessed memory locations. Scrubbing the memory prevents accumulation of single-bit errors. Double-bit errors result when two single-bit errors occur in the same word. Since the probability of this happening is quite low, scrubbing memory prevents most double-bit errors.

The Am29C668 CDMC is capable of controlling large memories, up-to-four banks of 4-Mbit DRAMS, and driving the $\overline{RAS}_n$, $\overline{CAS}_n$ and address lines without external drivers or damping resistors. It automatically generates the addresses needed for normal row refresh and refresh with scrubbing. The CDMC also has many features not utilized in this design, but appropriate for other systems, e.g, this design does not require reconfiguration of the CDMC through a simple I/O interface (see CDMC discussion, page 5).

Distinctive Characteristics

- 12 Mbytes of dynamic RAM (1M x 1-bit packages). 12 DRAM modules and 9 zip packages are used for maximum component density.
- Am29C668 4M Configurable Dynamic Memory Controller/Driver.
- Am29C660C high-speed 32-bit Error Detection and Correction Circuit corrects all single-bit errors, detects all double and some multiple-bit errors.
- One Wait State at 16 MHz with 120-ns DRAMs. Zero Wait States at 16 MHz with 70-ns DRAMs. One Wait State at 20 MHz with 85-ns DRAMs. Supports both basic transfer cycles and matched memory cycles.
- Supports memory scrubbing during refresh for improved reliability.
- The ability to relocate memory and I/O space through the Programmable Option Select (POS) registers. All options are software configurable through the POS registers.
- Dynamic Memory Timing Controller implemented using Programmable Array Logic (PAL®) devices and delay lines.
- Am29C688 used in Am29368-compatible mode with logic to reconfigure the Am29C668.
- 32-bit internal data bus with 7-bit check bit and 7-bit syndrome bus.
- Syndrome latch for diagnostic and test purposes.
- Direct interface with PS/2 Model 70 and 80 systems.

A BRIEF OVERVIEW OF THE MICRO-CHANNEL ARCHITECTURE

The Micro-Channel bus used in IBM PS/2 systems provides for three different add-in cards: 16-bit, 16-bit with auxillary video extension and 32-bit. This board is designed for 32-bit systems and fits only in the current IBM PS/2 Models 70 and 80 systems. The Micro Channel supports two types of bus accesses: Matched Memory Cycles and Basic Transfer Cycles. The Basic Transfer Cycle is supported by all PS/2 models. It permits at least 200-ns minimum cycle time with wait states of at least 100 ns. A card designed to support this type of access can be used in any of the PS/2 models. Matched Memory Cycles are only supported in 80386 machines, currently Models 70 and 80; cycle time is dependent upon the processor cycle time. Each access is a minimum of three processor cycles; however, additional wait states may be added. Cards designed to support this type of access cannot be used in all machines. This design supports Matched Memory Accesses, because it provides for the highest performance. Table 1 shows the number of wait states for specific memory access times and processor speeds for Read accesses. If only Basic Transfer Cycles are used, all Read accesses require one wait state (300 ns cycle time) to complete. Table 2 shows the read-access time (Status Valid to Read Data Valid) for different speed DRAMs.

* PS/2 is a registered trademark of IBM Corporation.

Table 1. Number of Wait States for Processor Speed and Memory Access Time

Memory Access Time-ns	Processor Clock 16 MHz	20 MHz
120	1	2
100	1	2
85	1	1
80	1	1
70	0	1

Table 2. Read Access Times For Different DRAMs from Status Active

DRAM Access Time-ns	Memory Board Access Time-ns
120	188
100	168
85	153
80	148
70	138

DETAILED DESCRIPTION

The primary data paths and functional elements are shown in Figure 1. The following discussion describes each section of the block diagram in detail. Components not appearing in the block diagram but existing on the schematic are also discussed.

Edge Card Connectors

This board can only be used in a 32-bit Micro Channel backplane. Interrupt Request $\overline{IRQ3}$ is used by the board to signal the detection of a multiple-bit error to the system processor. The interrupt can be disabled by writing a zero to bit 0 of POS Register 104. All signals used from the edge connector are listed on pages 8 and 9.

Dynamic Memory Timing Control (DMTC)

The timing controller for this board was implemented using PAL devices and delay lines for increased flexibility and performance. The following subsections describe the signals and their function.

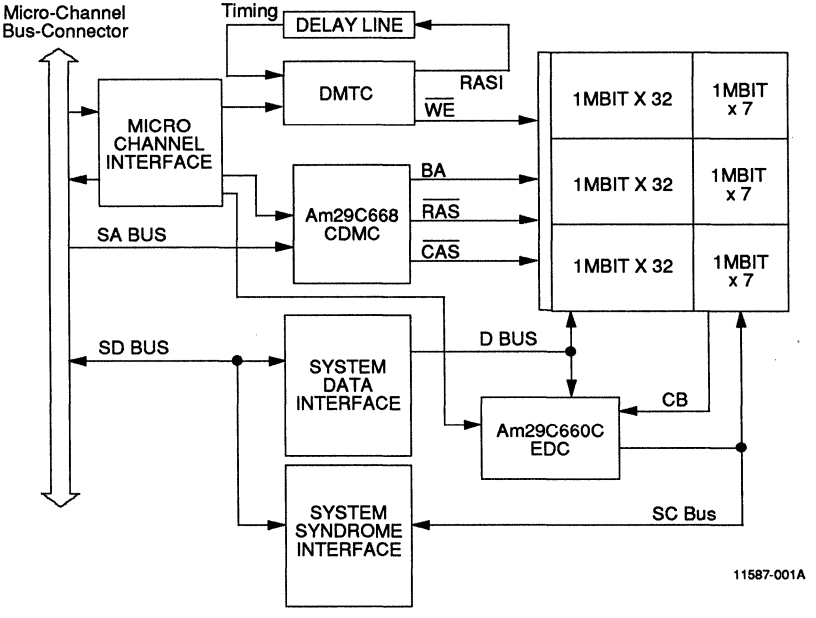


Figure 1. Block Diagram

I/O Channel Ready Logic

I/O Channel Ready $\overline{\text{IOCHRDY}}$ is used to signal the processor that valid data is ready during Reads and to signal completions of Writes. When $\overline{\text{IOCHRDY}}$ is pulled High, the processor inserts wait states until $\overline{\text{IOCHRDY}}$ is asserted. This signal is very important since it must be deasserted before Command Signal $\overline{\text{CMD}}$ is asserted by the system. If this does not happen, the system does not insert wait states and data is lost or corrupted.

There are two different types of extended cycles during a basic transfer cycle: synchronous and asynchronous. Synchronous extended cycles insert only one wait state. $\overline{\text{IOCHRDY}}$ is asserted within 30 ns of $\overline{\text{CMD}}$ going active. During asynchronous extended cycles, $\overline{\text{IOCHRDY}}$ is asserted 60 ns at most, before Read data is valid. To insert a wait state during a matched memory cycle, $\overline{\text{IOCHRDY}}$ is asserted 45 ns, at most, before Read data is valid. As the processor speed increases, this time proportionally decreases, i. e., a 20-MHz machine has 33 ns before Read data is valid. For this board to work in faster machines, $\overline{\text{IOCHRDY}}$ is asserted when the data is valid. In this manner, the board functions properly in any system with only a minor speed penalty in slower machines. For basic transfer cycles, the board uses synchronous extended cycles to maximize the memory bandwidth.

If a cycle has not completed, a wait state must be inserted regardless of the next type of access. The signal BBar and Busy are used to handle this logic. When the system initiates a memory access, signaled by $\overline{\text{CMD}}$ going active, BBar is active. BBar remains active until the end of the board memory cycle, signaled by End of Cycle $\overline{\text{EOC}}$. When the system ends its memory access by deactivating $\overline{\text{CMD}}$, Busy goes active and remains active until the end of the board memory cycle. If any other access to the board is attempted while Busy is High, $\overline{\text{IOCHRDY}}$ is deasserted and wait states inserted until the board's memory cycle terminates.

RASI, Mode Selection and End of Cycle

The Row Address Strobe Input RASI, Mode Control MC_n and End of Cycle $\overline{\text{EOC}}$ signals are generated by U4. RASI is used to initiate the timing sequence and to signal the Am29C668 to generate the $\overline{RAS}_n$ signals to the appropriate bank of memory. Two different sets of mode signals are generated, AC[2:0] and MC[1:0]. The AC[2:0] signals are used for internal control within the DMTC. An encoding scheme for the memory state was selected to minimize inputs to the PAL devices. If a fully decoded scheme were used, six signals instead of three would be required. There is no speed penalty since the memory state must be latched for the duration of the memory cycle and the encoding and latching are all performed by one PAL. Table 3 shows the decoding of AC[2:0].

Table 3. AC[2:0] Decoding

AC[2:0]	Mode
000	No Operation (Idle)
001	Long Write (32-bits)
010	Write
011	Not Allowed
100	Read
101	Refresh without Scrubbing
110	Refresh with Scrubbing
111	Initialize

MC_1 and MC_0 control the type of memory access for the CDMC. Table 4 shows the decoding of MC_1 and MC_0.

Table 4. MC_1 and MC_0 Decoding

MC_1	MC_0	Mode
0	0	Refresh without scrubbing
0	1	Refresh with scrubbing or initialize
1	0	Read/Write mode
1	1	Reset Refresh Counter

The $\overline{\text{EOC}}$ signal is generated to signify the end of any memory cycle. This signal also resets AC[2:0]. When AC[2:0] = 000, the End of Timing (EOT) becomes active and resets the timing-tap output to ensure that there can be no glitch on RASI. If $\overline{\text{EOC}}$ resets both AC[2:0] and the timing taps, the timing taps may be reset before AC[2:0] is reset. The RASI logic goes High until AC[2:0] is reset, resulting in a glitch on RASI and consequently on $\overline{RAS}_n$ to the DRAMs.

Latched Error, Initialization and Interrupt Signals

There are two different cycle lengths: a short cycle used by Read without Error, Refresh, Long Write (32-bits) and Initialize and a long cycle used by Read with Error, Read/Modify/Write and Scrubbing. All cycle lengths except Read cycles are known at the beginning. Because of this, careful attention must be paid to the timing and logic used during Read cycles. The EDC generates the $\overline{\text{ERROR}}$ signal when a single or multiple-bit error is detected during a Read, Read/Modify/Write or Scrubbing cycle. $\overline{\text{LErr}}$ is used by the rest of the board to determine if a Read cycle is long or short. PAL20RA10, U5, samples $\overline{\text{ERROR}}$ at Timing Tap T2 and, if $\overline{\text{ERROR}}$ is false, signal $\overline{\text{LErr}}$ is deasserted. $\overline{\text{LErr}}$ is preset by $\overline{\text{EOC}}$. This logic assumes that every Read cycle is a long cycle unless $\overline{\text{ERROR}}$ is false at T2. This assures correct and concise logic implementation. If $\overline{\text{LErr}}$ were conditionally asserted instead of deasserted, much more complicated logic would be required, since another signal is needed to indicate when $\overline{\text{LErr}}$ is valid.

Device U5 also latches the Initialize $\overline{\text{INIT}}$ signal, which is generated by the Board Enable $\overline{\text{BDENBL}}$ going active. $\overline{\text{INIT}}$ remains active until Terminal Count TC is received from the Am29C668 indicating that all the memory locations have been initialized. Counter[0:3] counts the wake-up cycles. When eight wake-up cycles are completed, $\overline{\text{DONE}}$ is asserted signaling the DMTC that it can begin initializing memory. At the end of a wake-up cycle, Counter[0:3] is incremented. Counter[0:3] is initialized to 0 and when the count reaches 7, $\overline{\text{DONE}}$ is asserted and the counter is inhibited. $\overline{\text{DONE}}$ remains active until $\overline{\text{INIT}}$ is deactivated.

Memory-Board Interrupt $\overline{\text{INTR}}$ signals that a multiple error has occurred at time T4. $\overline{\text{INTR}}$ goes to Interrupt Request $\overline{\text{IRQ3}}$ on the backplane and is cleared by an access to the syndrome latch signal $\overline{\text{SynLE}}$. This signal can be disabled by writing a zero to bit 0 of POS register 104. Registering $\overline{\text{INTR3}}$ would not be required in systems that support bus retry. The Channel Check signal $\overline{\text{ChCk}}$ can also be generated by writing a one to bit 3 of POS Register 104. Device U5 generates SetChCk when a multiple error is detected and $\overline{\text{ChCk}}$ is enabled. This signal is normally disabled.

Pulsed CASI, Write Enable and Miscellaneous Logic Functions

The Column Address Strobe Input CASI is a pulsed CAS line used when connecting the data-in lines to the data-out lines on the DRAMs. The Interface Controller PAL U11 generates this signal from the registered timing-tap signals from the Micro Channel Interface EPB2001, U13. Figure 2 shows how the pulsed-CASI signal is produced. This only applies during Read/Modify/Write and Refresh-with-Scrubbing cycles. Note that at time A in the diagram, the DRAM outputs are three-stated so that the Am29C660 can drive the data bus.

Write Enable $\overline{\text{WE}}$ is used to write the valid data into the memory. One $\overline{\text{WE}}$ signal is used per bank to drive the high capacitive load. The total delay must be calculated since the load capacitance is greater than the load specified in the data sheet (CL = 50 pF). The load capacitance of the DRAMs is (4 x 60)+ (3 x 5) = 255 pF, and the internal resistance of the PAL is assumed to be 4 Ω during High-to-Low transitions. The maximum High-to-Low transition time is calculated from 4.0 V to 0.8 V. The final output voltage is 0.5 V; therefore, the maximum High-to-Low transition time is:

$$V_{OUT} = (4.0 - 0.5)\exp\left(\frac{-t}{4\ \Omega \times 255\ pF}\right) + 0.5$$

$$t\ (\text{High to Low}) = (4\ \Omega)(255\ pF)(-1)\ \ln\left(\frac{0.8 - 0.5}{4.0 - 0.5}\right) \approx 2.5\ ns$$

This is added to the worst-case t_{pd} (15 ns) to get the worst-case delay. The minimum High-to-Low transition is calculated from 2.4 to 0.8 V with a final voltage of 0.3 V:

$$V_{OUT} = (2.4 - 0.3)\exp\left(\frac{-t}{50\ \Omega \times 255\ pF}\right) + 0.3$$

$$t\ (\text{High to Low}) = (4\ \Omega)(255\ pF)(-1)\ \ln\left(\frac{0.8 - 0.3}{2.4 - 0.3}\right) \approx 1.5\ ns$$

The minimum Low-to-High transition is from 0.8 V to 2.4 V with a final voltage of 4.0 V. The internal resistance of the PAL is approximately 50 Ω:

$$V_{OUT} = (0.8 - 4.0)\exp\left(\frac{-t}{50\ \Omega \times 255\ pF}\right) + 4.0$$

$$t\ (\text{Low to High}) = (50\ \Omega)\ (255\ pF)\ (-1)\ \ln\left(\frac{2.4 - 4.0}{0.8 - 4.0}\right) \approx 9\ ns$$

Miscellaneous logic functions are performed by U6, a combinatorial PAL device. Latch Enable Output or Generate LEO_GenL is a dual-purpose signal; when active High, it enables the output latches of the

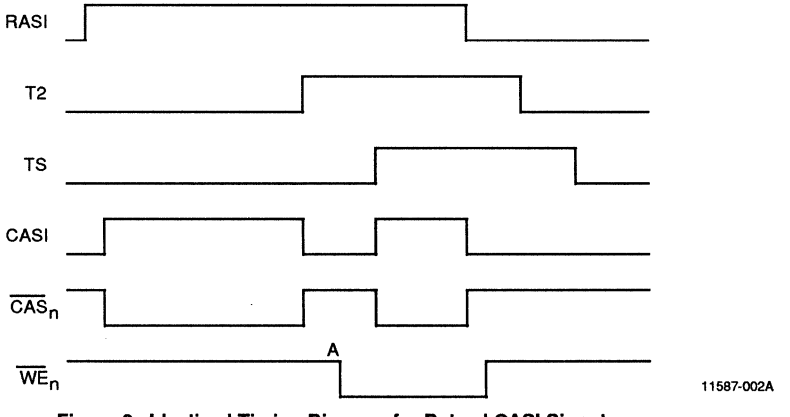


Figure 2. Idealized Timing Diagram for Pulsed CASI Signal

Am29C660; when it is active Low, the Am29C660 generates check bits for the data in the input latch. The Latch Enable Input signal LEI controls the latching of data into the Am29C660. When LEI is High, the input latch is transparent; when LEI is Low, data is latched. The Latch Enable Bus signal LEB controls the data latch from the internal data bus to the system bus. When LEB is High, the latch is transparent. S_AND_$\overline{\text{LEB}}$ is used to condition the output enables from the system bus.

Delay Lines

Delay lines D1 through D4 provide the timing reference signals; RASI initiates the timing sequence. See page 18-19 for the tap-timing calculations of the timing configuration currently installed on the board. The board is designed for 120-ns DRAMs. According to the board timing requirements, each timing signal must be reset at the beginning of each cycle. A PAL20RA10, U9, is used to register the timing taps. Since it has separate clocks for each of its 10 output registers, this PAL saves board space over a discrete-logic implementation. The outputs are reset by $\overline{\text{EOC}}$ going active. Registering the timing taps allows shorter cycle times, since it is not necessary to wait for the delay line to clear.

Interface Control

PAL U7 generates Output Enable System Data $\overline{\text{OE_SD}}$[0:3], Output Enable EDC $\overline{\text{OE_EDC}}$[0:3] and LG signals. Output Enable System Data controls the gating of data from the system bus to the memory. These signals control the flow of data during Writes to memory and to the EDC diagnostic register. For diagnostic Writes, the lower word (16 bits) from the data bus is input to the Am29C660. For Writes, the data is controlled by the $\overline{\text{BE}}$[0:3] signals generated by the system board. Output Enable EDC determines which data bytes the EDC supplies during Write and Read/Modify/Write cycles. The EDC supplies the unaltered bytes during a Write and provides the corrected bytes during a Read/Modify/Write cycle. LEY controls the input data latches of the bus transceivers. When the signal is active, the data latch is transparent; when it is inactive, the system data is latched.

PAL U8 generates $\overline{\text{OE_BD}}$[0:3], Syndrome Output Enable, Syndrome Latch Enable and Diagnostic Latch Enable. Syndrome Output Enable $\overline{\text{SYNOE}}$, an active-Low signal, enables the Am29C823 to drive the data from the syndrome latch onto the system data bus. Syndrome Latch Enable $\overline{\text{SYN_LE}}$ is an active-Low signal that latches the syndrome bits when an error is detected. $\overline{\text{OE_BD}}$[0:3] drives the $\overline{\text{OE_C}}$ and $\overline{\text{OE_D}}$ inputs of the transceivers that provide the system data-bus interface. These signals gate the data lines to and from the various byte-wide data bits of the internal data bus on the board, the D bus. Note from the PAL equations that the gating signals are conditioned by S_AND_$\overline{\text{LEB}}$ to ensure proper latching of the data from the DRAMs. LEY enables the latches on Writes; $\overline{\text{LEB}}$ enables the latches on Reads.

Configurable Dynamic Memory Controller

The Am29C668 Configurable Dynamic Memory Controller U1 supplies the DRAM array with multiplexed address, $\overline{\text{RAS}}_n$ and $\overline{\text{CAS}}_n$ signals. Timing inputs to this device are provided by the delay lines registered by U9. The Am29C668 is used in the Am29C368-compatible mode and can be reconfigured by writing data to the configuration registers.

During Initialization, the Am29C668 generates initialization cycles until the entire memory is written with data and check bits. When the initialization is complete, TC is asserted High signaling the DMTC that the initialization is complete. $\overline{\text{RAS}}$-only Refresh is used when no memory scrubbing is selected. Note: AC10 and AR10 are not connected since 1-Mbit DRAMs are used; 1-Mbit DRAMs use only 20 address bits.

Error Detection and Correction Circuit

The high-speed Am29C660C EDC U2 is used in the correct-always mode, i.e., data is always corrected before it is output to the bus. The fly-by mode, where the processor is interrupted when errors occur, cannot be used with PS/2 systems because the 80286/386 microprocessors do not support bus retry. In systems that support bus retry, data is read from the board as soon as it is accessed from memory. This saves 24 ns t_{pd} Data In to Data Correct for the C-speed part, during memory Reads. Memory Write times are not changed.

This device generates check bits during a Write and verifies the data and check bits during a Read. Separate error $\overline{\text{ERROR}}$ and multiple-bit error $\overline{\text{MULT_ERROR}}$ signals are output. If $\overline{\text{MULT_ERROR}}$ is asserted, then IRQ3 is active if enabled by IRQOE. Code ID generates input signal CODE ID0 to U2. If Code ID is active, a 32-bit slice is selected (the chip may operate in 32- or 64-bit mode); if Code ID is not active, the chip operates in internal-control mode. Using the internal-control mode, the user can access the diagnostic registers in the Am29C660C and more easily debug and test the board. See the Am29C660 data sheet for further details.

The Am29C660 internal diagnostic latch is available from the I/O channel. Data is written to the diagnostic latch through an I/O address specified through the POS registers. Bits 2 and 3 of POS Register 102 select four different addresses for the diagnostic latch. A 16-bit data word is written to the register to configure the part. Consult the Am29C660 data sheet for further information.

The $\overline{OESC}$ pin on the EDC is grounded to eliminate the OESC-to-SC bus output-enable delay in the Am29C660 with a resulting improvement in performance. This could not be done in an application where the CB bus and SC bus are tied together.

The Am29C660 must initialize the memory to a known state on reset. On $\overline{BdEnbl}$ going active, the data currently in the input latch is used to initialize the memory. If a known pattern must be written into memory, it may be done after the hardware initialization in software.

System Data Bus Interface

The 74F543s, U15 to U18, provide data latching between the system bus and internal D bus. The control signals for these devices are driven by the interface controller PAL devices, U7 and U8. Note that in the documentation for the PAL, the equations include a $\overline{MemWr}$ term, included to prevent D-bus contention during a Read-without-Error cycle after T2. The latches in the 74F543 are used to latch the Write data and free the bus. If simple transceivers are used, the data on the bus must be held until the Write data set-up time for the DRAMs is satisfied, adding to the access time. By using the 74F543s, 100 ns are saved during Read/Modify/Write cycles by latching the data and releasing the bus after the Read access is completed. This is not a problem on Long Write cycles since the data can be written directly to memory.

Syndrome Register (Syndrome Logic)

An Am29C823 register U12 stores the syndrome bits when an error is detected by the Am29C660C at timing tap T3. The contents of this register can then be read by the microprocessor from the I/O channel in an interrupt routine. Decoding the syndrome register bits reveals information about the error that occurred (see Am29C660 data sheet). The error signal is qualified by $\overline{MemRd}$ and $\overline{MemWr}$ so that the syndrome latch can only be updated when errors occur during a Read or Write operation. $\overline{INIT}$ is connected to the $\overline{CLR}$ of the Am29C823 to clear the latch on power-up and system reset.

DRAM Array

The DRAM array consists of two rows of two blocks: the 32-bit data block and 7-bit check-bit block. It is organized as three rows of 39 bits by 1 Mbit devices, or 117 components. Total user memory is 12 Mbytes. By using a pulsed $\overline{CAS}$ signal to the chips, the data-in pins can be tied to the data-out pins on the DRAMs. This facilitates routing on the PC board by minimizing the number of traces to the DRAM array. Four 9-bit memory modules and three zip packages are used per memory bank. The first four bits in each module are data bits. The last bit in each module and the three zip packages are used to store the check bits. The ninth bit of each module has separate data-in and data-out lines, while the rest of the module has common data-in and data-out lines. The check bits require separate data-in and data-out lines, since $\overline{OES}$ is tied Low to minimize the delay from check-bit generation to write back. This design uses 120-ns DRAMs to minimize cost. Using faster memories lowers the access times and reduces the number of wait states needed (See Tables 1 and 2).

Micro-Channel Bus Interface

Devices U13 and U14 perform most of the Micro-Channel interface. U14 is a comparator that compares the upper eight address bits with the eight bits set in POS register 105. If the upper bytes match and MADE24 is inactive, AddressValid 32 is active. U13 is a user-configurable Adapter Interface device, the EPB2001, designed specifically for the PS/2 Micro Channel. It decodes the lower 24 bits of the address, Address Valid 32, $\overline{MIO}$, $\overline{S0}$ and $\overline{S1}$ and signal-valid memory accesses and I/O accesses. $\overline{CdDS16}$ and $\overline{CdDS32}$ are also generated by this device. $\overline{CdDS16}$ is generated during an access to the syndrome latch, Am29C660's diagnostic register, Am29C668 configuration register or to memory. $\overline{CdDS32}$ is only asserted during memory accesses. Both $\overline{CdDS16}$ and $\overline{CdDS32}$ are generated during memory accesses to indicate that the memory supports both 16- and 32-bit transfers.

Unit 13 also contains all the POS registers. Bits 0 to 2 of POS register 104 are output on POS I/O 0 to 2 and are used to drive the DMTC inputs IRQEN, RM and CodeID. By writing the appropriate values to these bits, the board is configured (Figure 3). $\overline{MemRd}$ and $\overline{MemWr}$ generated by the DMTC logic could have been generated from the Micro-Channel Interface device, U13. This was not done because it is faster to begin Read cycles when status becomes valid, rather than wait for U13 to generate them. Generating these signals via the DMTC logic saves 55 ns on basic transfer cycles and 82 ns on matched memory cycles at 16 MHz.

POS CONFIGURATION INFORMATION

Figure 3 shows the mapping of control bits in the POS registers. The use of POS registers eliminates the need for jumpers and helps the user to easily resolve conflicts in memory and I/O mapping. This design provides maximum flexibility when reconfiguring the board via this interface.

POS-register 104 is used to configure the modes of the board. If Bit 0 is zero, interrupts from the board are disabled; a one enables the interrupts. Bit 1 determines the type of refresh, one for scrubbing and zero for non-scrubbing. Bit 2 determines the mode of the

Am29C660C. If Bit 2 is zero, the board is in normal operating mode; if one, the board is running diagnostics and using the values in the diagnostic register to control the operation of the Am29C660C. Bit 4 is the fast bit for indicating the processor speed, zero for 16-MHz systems and one for faster systems. This bit is used in generating the wait states for the board.

POS-register 102 is used to determine the I/O address space. Bits 0:2 select one of eight address spaces for the CDMC, Syndrome and Diagnostic Latch. The total I/O space for the board is 4 Kbytes. Since an I/O-mapped scheme is used to reconfigure the Am29C668, it occupies the lower 2 Kbytes I/O address space. The lower 11 bits of the address, bits A 00 to A 10, are the configuration data used by the Am29C668. The CDMC could have been mapped to a single I/O address with the configuration data written on the data bus, but this would require a multiplexer to select between the data and address bus, which adds extra board space, control logic and delay to the system. Using this I/O-mapped scheme, the Micro-Channel interface can drive $\overline{\text{Cd DS}}$ 16 and $\overline{\text{Cd DS}}$ 32, saving additional logic. The Syndrome Latch I/O address is byte 0 and the Diagnostic Latch I/O address is byte 1 of the upper 2 Kbytes of the I/O space.

POS-register 106 contains the upper eight bits of the memory board. These eight bits are compared with the upper eight bits on the system bus to determine if the access is the same address space as the memory.

POS-registers 100 and 101 contain the board ID for identifying the card during setup. POS-register 102 bit 0 is used as the board-enable signal. This bit is reset by ChReset or by the processor writing a zero to this bit during a card-setup cycle. While this bit is zero, the board does not respond to any access. This bit can only be reset by the processor during card-setup cycles and cannot be set during normal I/O Writes to this register. POS-register-105 bit 7 is the channel-check flag, the state of this bit is output on the bus through the $\overline{\text{ChCk}}$ pin. This bit is set by ChReset or by writing a one to the location. The bit is reset by asserting $\overline{\text{SetChk}}$ or by writing a zero to the location.

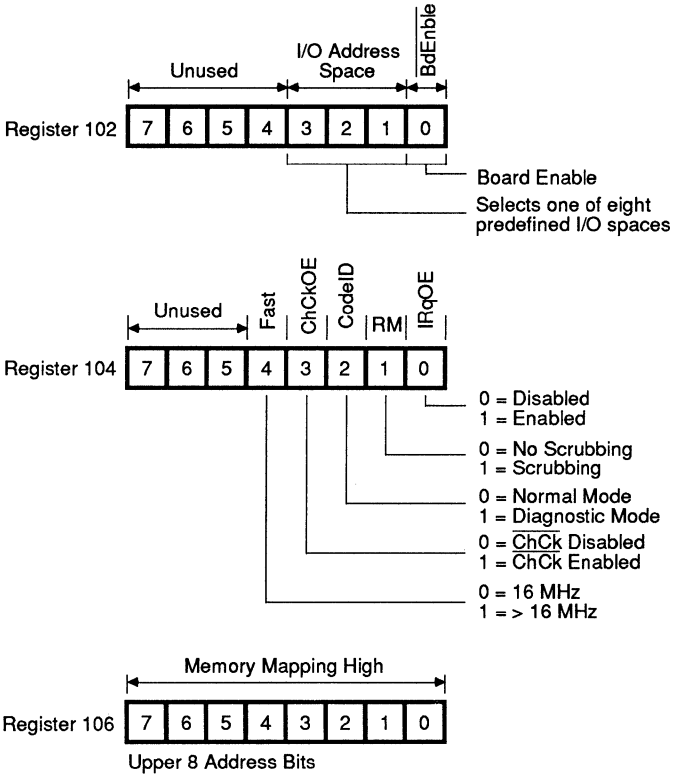


Figure 3. POS Registers Bit Map

EDGE CONNECTOR PIN NAMES

Pin #	Signal Name	I/O	Pin #	Signal Name	I/O
AM4	-	-	BM4	GND	-
AM3	MMC CMD	I	BM3	-	-
AM2	GND	-	BM2	MMCR	O
AM1	MMC	I	BM1	-	-
A1	CD SETUP	I	B1	-	-
A2	MADE 24	I	B2	-	-
A3	GND	-	B3	GND	-
A4	A 11	I	B4	-	-
A5	A 10	I	B5	GND	-
A6	A 09	I	B6	A 23	I
A7	+5 V	-	B7	A 22	I
A8	A 08	I	B8	A 21	I
A9	A 07	I	B9	GND	-
A10	A 06	I	B10	A 20	I
A11	+5 V	-	B11	A 19	I
A12	A 05	I	B12	A 18	I
A13	A 04	I	B13	GND	-
A14	A 03	I	B14	A 17	I
A15	+5 V	-	B15	A 16	I
A16	A 02	I	B16	A 15	I
A17	A 01	I	B17	GND	-
A18	A 00	I	B18	A 14	I
A19	-	-	B19	A 13	I
A20	ADL	I	B20	A 12	I
A21	-	-	B21	GND	-
A22	-	-	B22	-	-
A23	-	-	B23	IRQ 03	O
A24	-	-	B24	-	-
A25	-	-	B25	GND	-
A26	-	-	B26	-	-
A27	-	-	B27	-	-
A28	-	-	B28	-	-
A29	-	-	B29	GND	-
A30	-	-	B30	RESERVED	-
A31	+5 V	-	B31	RESERVED	-
A32	S0	I	B32	CHCK	O
A33	S1	I	B33	GND	-
A34	M/IO	I	B34	CMD	I
A35	-	-	B35	-	-
A36	CD CHRDY	O	B36	CD SFDBK	O
A37	D 00	I/O	B37	GND	-
A38	D 02	I/O	B38	D 01	I/O
A39	+5 V	-	B39	D 03	I/O
A40	D 05	I/O	B40	D 04	I/O
A41	D 06	I/O	B41	GND	-
A42	D 07	I/O	B42	CHRESET	I
A43	GND	-	B43	RESERVED	-
A44	-	-	B44	RESERVED	-
A45	REFRESH	I	B45	GND	-
A46	-	-	B46	-	-
A47	-	-	B47	-	-
A48	+5 V	-	B48	D 08	I/O
A49	D 10	I/O	B49	D 09	I/O
A50	D 11	I/O	B50	GND	-

Pin #	Signal Name	I/O	Pin #	Signal Name	I/O
A51	D 13	I/O	B51	D 12	I/O
A52	-	-	B52	D 14	I/O
A53	RESERVED	-	B53	D 15	I/O
A54	-	-	B54	GND	-
A55	$\overline{\text{Cd DS}}$ 16	O	B55	-	-
A56	+5 V	-	B56	-	-
A57	-	-	B57	-	-
A58	-	-	B58	GND	-
A59	RESERVED	-	B59	RESERVED	-
A60	RESERVED	-	B60	RESERVED	-
A61	GND	-	B61	RESERVED	-
A62	RESERVED	-	B62	RESERVED	-
A63	RESERVED	-	B63	GND	-
A64	RESERVED	-	B64	D 16	I/O
A65	-	-	B65	D 17	I/O
A66	D 19	I/O	B66	D 18	I/O
A67	D 20	I/O	B67	GND	-
A68	D 21	I/O	B68	D 22	I/O
A69	+5 V	-	B69	D 23	I/O
A70	D 24	I/O	B70	RESERVED	-
A71	D 25	I/O	B71	GND	-
A72	D 26	I/O	B72	D 27	I/O
A73	+5 V	-	B73	D 28	I/O
A74	D 30	I/O	B74	D 29	I/O
A75	D 31	I/O	B75	GND	-
A76	RESERVED	-	B76	$\overline{\text{BE}}$ 0	I
A77	-	-	B77	$\overline{\text{BE}}$ 1	I
A78	$\overline{\text{BE}}$ 3	I	B78	$\overline{\text{BE}}$ 2	I
A79	-	-	B79	GND	-
A80	$\overline{\text{Cd DS}}$ 32	O	B80	-	-
A81	-	-	B81	A 24	I
A82	A 26	I	B82	A 25	I
A83	A 27	I	B83	GND	-
A84	A 28	I	B84	A 29	I
A85	+5 V	-	B85	A 30	I
A86	RESERVED	-	B86	A 31	I
A87	RESERVED	-	B87	GND	-
A88	RESERVED	-	B88	RESERVED	-
A89	GND	-	B89	RESERVED	-

Note: Side A is the component side, Side B is on the solder side.

I = Signal Input to the board.
O = Signal Output from the board.
I/O = Signal Input to and Output from the board.
- = Not Applicable.

PAL SOURCE CODE LISTINGS

The assembler used was PLPL. Functional test vectors were used to verify these equations, but are not listed to conserve space.

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  Channel_Ready (P22V10)          "U3"

PIN    Fast = 1 (INPUT Combinatorial)
       AC[2:0] = 2:4 (INPUT Combinatorial)
       LongWord = 5 (INPUT Combinatorial)
       /MemReq = 6 (INPUT Combinatorial)
       /Cmd = 7 (INPUT Combinatorial)
       /CdDS16 = 8 (INPUT Combinatorial)
       /MMCmd = 9 (INPUT Combinatorial)
       /MMC = 10 (INPUT Combinatorial)
       EoC = 11 (INPUT Combinatorial)
       /BdEnbl = 13 (INPUT Combinatorial)
       T1 = 14 (INPUT Combinatorial)

       ALE = 23 (OUTPUT Active_High Combinatorial)
       /IOChRdy = 22 (OUTPUT Active_Low Combinatorial)
       /MMCr = 21(OUTPUT Active_Low Combinatorial)
       /MemWr = 20 (OUTPUT Active_Low Combinatorial)
       /MemRd = 19 (OUTPUT Active_Low Combinatorial)
       BBar = 18 (OUTPUT Active_High Combinatorial)
       Busy = 17 (OUTPUT Active_High Combinatorial)
       Delayed = 16 (OUTPUT Active_High Combinatorial);

DEFINE Read =      /AC[0] * /AC[1] *  AC[2],
       Write =     /AC[0] *  AC[1] * /AC[2],
       Long_Write =  AC[0] * /AC[1] * /AC[2];

BEGIN
ENABLE (Busy,LongWord,BBar);          ENABLE (T1,BdEnbl) = 0;
ENABLE (IOChRdy) = BdEnbl;

IOChrdy = CdDS16 * Delayed + Fast * /T1 +
      /Fast * /Delayed * /Cmd + /Fast * /Delayed * /MMCmd + /Fast * Delayed * /T1;
Delayed = Busy * (S0 + S1) + Delayed * /Eoc * MemReq;

BBar = Cmd * MemReq + BBar * /EoC;
Busy = BBar * /Cmd + Busy * /EoC;

ALE = /Cmd * /MMCmd * /Busy;
MemRd = (Cmd + MMCmd) * Read;
MemWr = (Cmd + MMCmd) * (Write + Long_Write);

MMCr = MMC * MemReq * /Delayed;

END.
```

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  RASI  (P22V10)          "U4"

PIN   /Cmd = 1 (INPUT Combinatorial)
      LongWord = 2 (INPUT Combinatorial)
      /MemReq = 3 (INPUT Combinatorial)
      /S0 = 4 (INPUT Combinatorial)
      /S1 = 5 (INPUT Combinatorial)
      /FR = 6 (INPUT Combinatorial)
      RM = 7 (INPUT Combinatorial)
      /Init = 8 (INPUT Combinatorial)
      T2 = 9 (INPUT Combinatorial)
      T8 = 10 (INPUT Combinatorial)
      T9 = 11 (INPUT Combinatorial)
      T10 = 13 (INPUT Combinatorial)
      /Latched_Err = 14 (INPUT Combinatorial)
      Done = 15 (INPUT Combinatorial)
      Busy = 16 (INPUT Combinatorial)

      AC[0:2] = 21:23 (OUTPUT Active_High Combinatorial)
      MC1 = 20 (OUTPUT Active_High Combinatorial)
      MC0 = 19 (OUTPUT Active_High Combinatorial)
      RASI = 18 (OUTPUT Active_High Combinatorial)
      /EoC = 17 (OUTPUT Active_Low Combinatorial);

DEFINE Read =      /AC[0] * /AC[1] *  AC[2],
       Write =     /AC[0] *  AC[1] * /AC[2],
       Long_Write =  AC[0] * /AC[1] * /AC[2];

BEGIN

ENABLE (RASI,EoC,MC0,MC1,EoT,AC[0:2]);
ENABLE (Done,Busy,Latched_Err) = 0;

AC[0] = (LongWord * MemWr) * /Cmd + (Refresh * /RM + Init) * /Busy + AC[0] * /EoC;
AC[1] = (/LongWord * MemWr) * /Cmd + (Refresh * RM + Init) * /Busy + AC[1] * /EoC;
AC[2] = (MemRd + Refresh + Init) * /Busy + AC[1] * /EoC;

EoC = Init * T8 + Refresh * /RM * T8 + Refresh * RM * T10 +
      Write * T10 + Long_Write * T8 + Read * (/Latched_Err * T8 + Latched_Err * T10) +
      EoC * T8;

RASI = (Init * /T2 + Refresh * /RM * /T2 * /MC0 * /MC1 +
        Refresh * RM * /T9 * MC0 * /MC1 +
        (MemRd + Read) * (/Latched_Err * /T2 + Latched_Err * /T9) +
        (MemWr + Write) * /T9 + MemWr * LongWord * /T2) * /Busy;

MC0 = Init + Refresh * RM * (/T9 + /RASI) + DMCSel;
MC1 = /(Refresh * /RM * (/T2 + /RASI) + Refresh * RM * (/T9 + /RASI) + Init * Done);

END.
```

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  SAMPLE (P20RA10)        "U5"

PIN    /PRE_LOAD = 1  (CONTROL)
       INT2 = 2  (INPUT Combinatorial)
       INT4 = 3  (INPUT Combinatorial)
       ChCkOE = 4  (INPUT Combinatorial)
       /EoC = 5  (INPUT Combinatorial)
       /SynSel = 6  (INPUT Combinatorial)
       TC = 7  (INPUT Combinatorial)
       /BdEnbl = 8  (INPUT Combinatorial)
       /Err = 9  (INPUT Combinatorial)
       /MErr = 10  (INPUT Combinatorial)
       IRqOE = 11  (INPUT Combinatorial)
       /OE = 13  (CONTROL)

       /LErr = 23  (OUTPUT Active_Low Registered)
       /Intr = 22  (OUTPUT Active_Low Registered)
       /INIT = 21  (OUTPUT Active_Low Registered)
       /Done = 20  (OUTPUT Active_Low Registered)
       /Counter[0:2] = 19:17  (OUTPUT Active_Low Registered)
       /LMErr = 16  (OUTPUT Active_Low Registered)
       /SetChCk = 15  (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE(LErr,Intr,Done,Counter[0:2],INIT,LMErr);

LErr = Err; CLOCK_PT(LErr) = INT2;  PRESET(LErr) = EoC;

Intr = MErr + Intr;      CLOCK_PT(Intr) = INT4;  RESET(Intr) = SynSel;
ENABLE(Intr) = IRqOE*BdEnbl;
LMErr = MErr;      CLOCK_PT(LMErr) = INT4; RESET(LMErr) = EoC;

CLOCK_PT (INIT)= BdEnbl;       RESET(INIT) = TC * EoC;        INIT = 1;

RESET(Counter[2:0]) = /INIT;  CLOCK_PT(Counter[2:0]) = EoC;
IF (/Done = 1) THEN CASE (Counter[2:0]) BEGIN
       0) Counter[2:0] = 1;
       1) Counter[2:0] = 2;
       2) Counter[2:0] = 3;
       3) Counter[2:0] = 4;
       4) Counter[2:0] = 5;
       5) Counter[2:0] = 6;
       6) Counter[2:0] = 7;
       7) Counter[2:0] = 0;
    END;
Done = Counter[2] * Counter[1] * Counter[0] + Done * INIT;
CLOCK_PT(Done) = EoC;     RESET(Done) = /INIT;

SetChCk = ChCkOE * LMErr;
END.
```

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  MISC (P22P10)           "U6"

PIN   AC[0:2] = 3:1 (INPUT Combinatorial)
      RASI = 4 (INPUT Combinatorial)
      T2 = 5 (INPUT Combinatorial)
      T3 = 6 (INPUT Combinatorial)
      T5 = 7 (INPUT Combinatorial)
      T6 = 8 (INPUT Combinatorial)
      T7 = 9 (INPUT Combinatorial)
      T9 = 10 (INPUT Combinatorial)
      CAS = 11 (INPUT Combinatorial)
      TS = 13 (INPUT Combinatorial)
      /LErr = 14 (INPUT Combinatorial)
      /LMErr = 15 (INPUT Combinatorial)

      /S_and_not_LEB = 23 (OUTPUT Active_Low Combinatorial)
      /LEO_GenL = 22 (OUTPUT Active_Low Combinatorial)
      LEI = 21 (OUTPUT Active_High Combinatorial)
      /LEB = 20 (OUTPUT Active_Low Combinatorial)
      CASI = 19 (OUTPUT Active_High Combinatorial)
      /WE[0:2] = 16:18 (OUTPUT Active_Low Combinatorial);

DEFINE Read =       /AC[0] * /AC[1] *  AC[2],
       Write =      /AC[0] *  AC[1] * /AC[2],
       Long_Write =  AC[0] * /AC[1] * /AC[2],
       Refresh =     AC[0] * /AC[1] *  AC[2],
       Scrub =      /AC[0] *  AC[1] *  AC[2],
       Init =        AC[0] *  AC[1] *  AC[2];

BEGIN

ENABLE (LEB,LEI,LEO_GenL,S_and_not_LEB,WE[0:2],CASI);

LEB = Read * RASI * (/T2 * /LErr + /T3 * LErr);

LEI = Read * RASI * /T7 + Read * LErr * /LEO_GenL * /T9 +
      Write * RASI * /T7 + Write * /LEO_GenL * /T9 + Long_Write * RASI * /T5 +
      Scrub * RASI * /T7 + Scrub * /LEO_GenL * /T9;

LEO_GenL = (Scrub + Write + Read * LErr) * T3 * /T9 + Long_Write * RASI * /T2 + Init;

S_and_not_LEB = RASI * /T2 * (Read + Write + Long_Write + Refresh + Scrub) +
    Read * RASI * (/T2 * /LErr + /T3 * LErr);

CASI = /T2 * CAS + TS * CAS;

WE[0] = Init + Long_Write * T5 * /T2 + (Scrub + Read * LErr + Write) * T6 * /T9 * /LMErr;
WE[1] = Init + Long_Write * T5 * /T2 + (Scrub + Read * LErr + Write) * T6 * /T9 * /LMErr;
WE[2] = Init + Long_Write * T5 * /T2 + (Scrub + Read * LErr + Write) * T6 * /T9 * /LMErr;

END.
```

```
"Douglas Lee                                                August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  INTERFACE (P22P10)              "U7"

PIN    /BE[0:3] = 1:4 (INPUT Combinatorial)
       T2 = 5 (INPUT Combinatorial)
       T10 = 6 (INPUT Combinatorial)
       /MemWr = 7 (INPUT Combinatorial)
       S_and_not_LEB = 8 (INPUT Combinatorial)
       AC[0:2] = 11:9 (INPUT Combinatorial)
       /Latched_Err = 13 (INPUT Combinatorial)
       LEDiag = 14 (INPUT Combinatorial)

       /LEY = 23 (OUTPUT Active_Low Combinatorial)
       /OE_SD[0:3] = 19:22 (OUTPUT Active_Low Combinatorial)
       /OE_EDC[0:3] = 18:15 (OUTPUT Active_Low Combinatorial);

DEFINE Read =      /AC[0] * /AC[1] *  AC[2],
       Write =     /AC[0] *  AC[1] * /AC[2],
       Scrub =     /AC[0] *  AC[1] *  AC[2],
       Init =       AC[0] *  AC[1] *  AC[2];

BEGIN

ENABLE(OE_SD[0:3],OE_EDC[0:3],LG);

OE_EDC[0] = Init + Scrub * T2 * /T10 + Write * /BE[0] * T2 * /T10 +
       Read * Latched_Err * T2 * /T10 + OE_EDC[0] * /T10;

OE_EDC[1] = Init + Scrub * T2 * /T10 + Write * /BE[1] * T2 * /T10 +
       Read * Latched_Err * T2 * /T10 + OE_EDC[1] * /T10;

OE_EDC[2] = Init + Scrub * T2 * /T10 + Write * /BE[2] * T2 * /T10 +
       Read * Latched_Err * T2 * /T10 + OE_EDC[2] * /T10;

OE_EDC[3] = Init + Scrub * T2 * /T10 + Write * /BE[3] * T2 * /T10 +
       Read * Latched_Err * T2 * /T10 + OE_EDC[3] * /T10;

OE_SD[0] = LEDiag + S_and_not_LEB * (Write + Long_Write) * /OE_EDC[0];

OE_SD[1] = LEDiag + S_and_not_LEB * (Write + Long_Write) * /OE_EDC[1];

OE_SD[2] = S_and_not_LEB * (Write + Long_Write) * /OE_EDC[2];

OE_SD[3] = S_and_not_LEB * (Write + Long_Write) * /OE_EDC[3];

LEY = MemWr + LEDiag;

END.
```

```
"Douglas Lee                                                August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  Output_Enable (P22P10)          "U8"

PIN   /BE[0:3] = 4:1 (INPUT Combinatorial)
      T7 = 5 (INPUT Combinatorial)
      /MemRd = 6 (INPUT Combinatorial)
      /MemWr = 7 (INPUT Combinatorial)
      /Err = 8 (INPUT Combinatorial)
      /SynSel = 9 (INPUT Combinatorial)
      /DiagSel = 10 (INPUT Combinatorial)
      /DMCSel = 11 (INPUT Combinatorial)
      /IOWr = 13 (INPUT Combinatorial)
      /IORd = 14 (INPUT Combinatorial)

      LEDiag = 23 (OUTPUT Active_High Combinatorial)
      /SynLE = 22 (OUTPUT Active_Low Combinatorial)
      /SynOE = 21 (OUTPUT Active_Low Combinatorial)
      RL = 20 (OUTPUT Active_Low Combinatorial)
      /OE_Bd[0:3] = 16:19 (OUTPUT Active_Low Combinatorial);

BEGIN

ENABLE (OE_Bd[0:3],SynLE,SynOE,RL,LEDiag);
ENABLE (IORd) = 0;

OE_Bd[0] = MemRd * T7 * BE[0];

OE_Bd[1] = MemRd * T7 * BE[1];

OE_Bd[2] = MemRd * T7 * BE[2];

OE_Bd[3] = MemRd * T7 * BE[3];

SynLE = (MemRd + MemWr) * Err;

SynOE = IORd * SynSel;

LEDiag = IOWr * DiagSel;

RL = DMCSel * IOWr;

END.
```

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  TIMER (P20RA10)          "U9"

PIN    /PRE_LOAD = 1 (CONTROL)
       Int1or2 = 2 (INPUT Combinatorial)
       Int3 = 3 (INPUT Combinatorial)
       Int5 = 4 (INPUT Combinatorial)
       Int6 = 5 (INPUT Combinatorial)
       Int7 = 6 (INPUT Combinatorial)
       Int8 = 7 (INPUT Combinatorial)
       Int9 = 8 (INPUT Combinatorial)
       Int10 = 9 (INPUT Combinatorial)
       /EoT = 10 (INPUT Combinatorial)
       /OE = 13 (CONTROL)

       T1  = 23 (OUTPUT Active_Low Registered)
       T2  = 22 (OUTPUT Active_Low Registered)
       T3  = 21 (OUTPUT Active_Low Registered)
       T5  = 20 (OUTPUT Active_Low Registered)
       T6  = 19 (OUTPUT Active_Low Registered)
       T7  = 18 (OUTPUT Active_Low Registered)
       T8  = 17 (OUTPUT Active_Low Registered)
       T9  = 16 (OUTPUT Active_Low Registered)
       T10 = 15 (OUTPUT Active_Low Registered);

BEGIN

ENABLE(T1,T2,T3,T6,T7,T8,T9,T10);

/T1  = 1;   CLOCK_PT(T1) = Int1or2; PRESET(T1) = EoT;
/T2  = 1;   CLOCK_PT(T2) = Int1or2; PRESET(T2) = EoT;
/T3  = 1;   CLOCK_PT(T3) = Int3;    PRESET(T3) = EoT;
/T5  = 1;   CLOCK_PT(T5) = Int5;    PRESET(T5) = EoT;
/T6  = 1;   CLOCK_PT(T6) = Int6;    PRESET(T6) = EoT;
/T7  = 1;   CLOCK_PT(T7) = Int7;    PRESET(T7) = EoT;
/T8  = 1;   CLOCK_PT(T8) = Int8;    PRESET(T8) = EoT;
/T9  = 1;   CLOCK_PT(T9) = Int9;    PRESET(T9) = EoT;
/T10 = 1;   CLOCK_PT(T10) = Int10;  PRESET(T10) = EoT;

END.
```

```
"Douglas Lee                                                    August 9, 1988
32-Bit Error Detection and Correction Board for the Micro Channel."

DEVICE                  Latch (16L8)            "U15"

PIN   /BE[3:0] = 1:4 (INPUT Combinatorial)
      Refresh = 5 (INPUT Combinatorial)
      ALE = 6 (INPUT Combinatorial)
      AC[2:0] = 7:10 (INPUT Combinatorial)

      /BEI[3:0] = 19:16 (OUTPUT Active_Low Combinatorial)
      LongWord = 15 (OUTPUT Active_Low Combinatorial)
      /FRH = 14 (OUTPUT Active_Low Combinatorial)
      /FR = 13 (OUTPUT Active_Low Combinatorial);

DEFINE Refresh =   AC[0] * /AC[1] *  AC[2],
      Scrub =     /AC[0] *  AC[1] *  AC[2];

BEGIN

ENABLE (BEI[0:3],LongWord,FRH,FR);

BEI[0] = BE[0] * /Cmd + BEI[0] * Cmd;
BEI[1] = BE[1] * /Cmd + BEI[1] * Cmd;
BEI[2] = BE[2] * /Cmd + BEI[2] * Cmd;
BEI[3] = BE[3] * /Cmd + BEI[3] * Cmd;

/LongWord = BE[3] * BE[2] * BE[1] * BE[0] + BEI[3] * BEI[2] * BEI[1] * BEI[0];

FRH = /(Refresh * /FRH + Refresh + Scrub);

FR = Refresh * FRH;

END.
```

DELAY LINE TAP CALCULATIONS

Derivation of the tap outputs is included here. The calculated time is adjusted to the nearest tap of the delay line (10-ns intervals) equal to or greater than the calculated time. The board is designed for 120-ns DRAMs.

	120 ns	100 ns	85 ns
MSEL - RASI to MUX SELECT			
t_{RAH} (DRAM) min	15.0	15.0	15.0
t_{SKEW} (Qn to $\overline{RAS}$n) 29C668 max	6.0	6.0	6.0
Total	21.0	21.0	21.0
$\overline{CAS}$ - RASI to $\overline{CAS}$			
MSEL	21.0	21.0	21.0
t_{SKEW} ($\overline{CAS}$n to Qn) 29C668 max	-2.0	-2.0	-2.0
t_{ASC} DRAM min	0.0	0.0	0.0
-t_{PD} ($\overline{CAS}$ to CASI) 22P8B min	-6.0	-6.0	-6.0
Total	13.0	13.0	13.0
INT5 - Valid Check Bits on Long Write			
t_{PD} (RASI to LEO_GenL) 22P10 max	15.0	15.0	15.0
t_{PD} (LEO_GenL to SC) 29C660C max	18.0	18.0	18.0
-t_{PD} (T5 to WE) 22P10 min	-7.0	-7.0	-7.0
Total	26.0	26.0	26.0
INT7 - Data Valid to 29C660C			
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	27.0	27.0	27.0
t_{ACC} DRAM max	120.0	100.0	85.0
-t_{PD} (INT7 to T7) 20RA10 min	-7.0	-7.0	-7.0
Total	140.0	120.0	105.0
INT2 - $\overline{ERROR}$ from 29C660C			
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	27.0	27.0	27.0
t_{ACC} DRAM max	120.0	100.0	85.0
t_{PD} (Data In to $\overline{ERROR}$) 29C660C max	16.0	16.0	16.0
t_{SU} ($\overline{ERROR}$) 20RA10 min	13.0	13.0	13.0
Total	176.0	156.0	141.0
INT3 - Corrected Data from EDC			
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	27.0	27.0	27.0
t_{ACC} DRAM max	120.0	100.0	85.0
t_{PD} (Data In–Data Out) 29C660C max	24.0	24.0	24.0
-t_{PD} (INT3 to T3) 20RA10 min	-7.0	-7.0	-7.0
Total	164.0	144.0	129.0

	120 ns	100 ns	85 ns
INT4 - $\overline{MERR}$ from 29C660C and $\overline{IOCHRDY}$ for R/M/W			
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	27.0	27.0	27.0
t_{ACC} DRAM max	120.0	100.0	85.0
t_{PD} (Data In to $\overline{MULT\ ERROR}$) 29C660C max	20.0	20.0	20.0
t_{SU} ($\overline{MERR}$) 20RA10 min	13.0	13.0	13.0
Total	180.0	160.0	145.0
INT 1 - $\overline{IOCHRDY}$ for Read without Error			
t_{PD} (RASI to $\overline{RAS}$n) 29C668 max	27.0	27.0	27.0
t_{ACC} DRAM max	120.0	100.0	85.0
t_{PD} (Data In to Data Out) 29C660C max	24.0	24.0	24.0
t_{PD} (Data Out to System Data) 29C983 max	14.0	14.0	14.0
-t_{PD} (T1 to IOCHRDY) 20L10B min	-6.0	-6.0	-6.0
-t_{PD} (INT1 to T1) 20RA10 min	-7.0	-7.0	-7.0
Total	172.0	152.0	137.0
INT 6 - Corrected Data and Check Bits (R/M/W)			
INT3	164.0	144.0	129.0
t_{SKEW} (T6 to T3) 20RA10 max	0.5	0.5	0.5
t_{PD} (T3 to LEO $\overline{GEN}$) 20L8B max	15.0	15.0	15.0
t_{PD} (LEO $\overline{GEN}$ to SCn) 29C660C max	18.0	18.0	18.0
t_{DS} DRAM min	0.0	0.0	0.0
Total	197.5	177.5	162.5
TS - Pulsed $\overline{CAS}$			
INT6	197.5	177.5	162.5
t_{PD} (INT6 to T6) 20RA10 max	20.0	20.0	20.0
t_{SKEW} ($\overline{WE}$n to CASI) 20L8B	3.0	3.0	3.0
-t_{PD} (CASI to $\overline{CAS}$n) 29C668 min	-15.0	-15.0	-15.0
t_{WCS} DRAM	0.0	0.0	0.0
Total	205.5	185.5	170.5
INT9 - End of $\overline{WE}$n and RASI (R/M/W)			
t_S	205.5	185.5	170.5
t_{PD} (TS to CASI) 20L8B max	15.0	15.0	15.0
t_{PD} (CASI to $\overline{CAS}$n) 29C668 max	31.0	31.0	31.0
t_{WCH} ($\overline{WE}$ Pulse Width) DRAM min	25.0	25.0	25.0
-t_{PD} (T9 to $\overline{WE}$) 20L8B min	-7.0	-7.0	-7.0
-t_{PD} (INT9 to T9) 20RA10 min	-7.0	-7.0	-7.0
Total	262.5	242.5	227.5

	120 ns	100 ns	85 ns
INT8 - End of Read without Error			
INT2	176.0	156.0	141.0
t_{RP} RAM min	90.0	80.0	70.0
Total	266.0	236.0	211.0
INT10 - End of R/M/W Cycle			
INT9	262.5	242.5	227.5
t_{RP} RAM min	90.0	80.0	70.0
Total	352.5	322.5	297.5

Signal	Should Be (ns)	Is (ns)	Note
MSEL	21.0	30.0	
CAS	22.0	30.0	= MSEL - 8
INT5	26.0	30.0	
INT7	140.0	140.0	
INT2	176.0	180.0	
INT3	164.0	170.0	
INT4	180.0	180.0	
INT1	172.0	180.0	
INT6	203.5	210.0	= INT3 + 33.5
TS	215.5	220.0	= INT6 + 8
INT8	270.0	270.0	= INT2 + 90
INT9	278.0	280.0	= TS + 57
INT10	370.0	370.0	= INT9 + 90

Notes:

1. Table for 120 ns DRAMs
2. Tap which are dependent or related to other taps are indicated with a comment in the "explanation" column.
3. Timing figures are based on Am29C660C data.

PARTS LIST

Unit #	Device	Description
U1	Am29C668	Configurable Dynamic Memory Controller
U2	Am29C660C	32-Bit Error Detection and Correction Circuit
U3	AmPAL16L8	Combinatorial PAL
U4	AmPAL22V10	Combinatorial PAL
U5	PAL20RA10	Asynchronous PAL
U6	AmPAL22P10	Misc. Logic Functions
U7	AmPAL22P10	Interface Controller
U8	AmPAL22P10	Interface Controller
U9	PAL20RA10	Asynchronous PAL
U10	AmPAL16L8	Combinatorial PAL

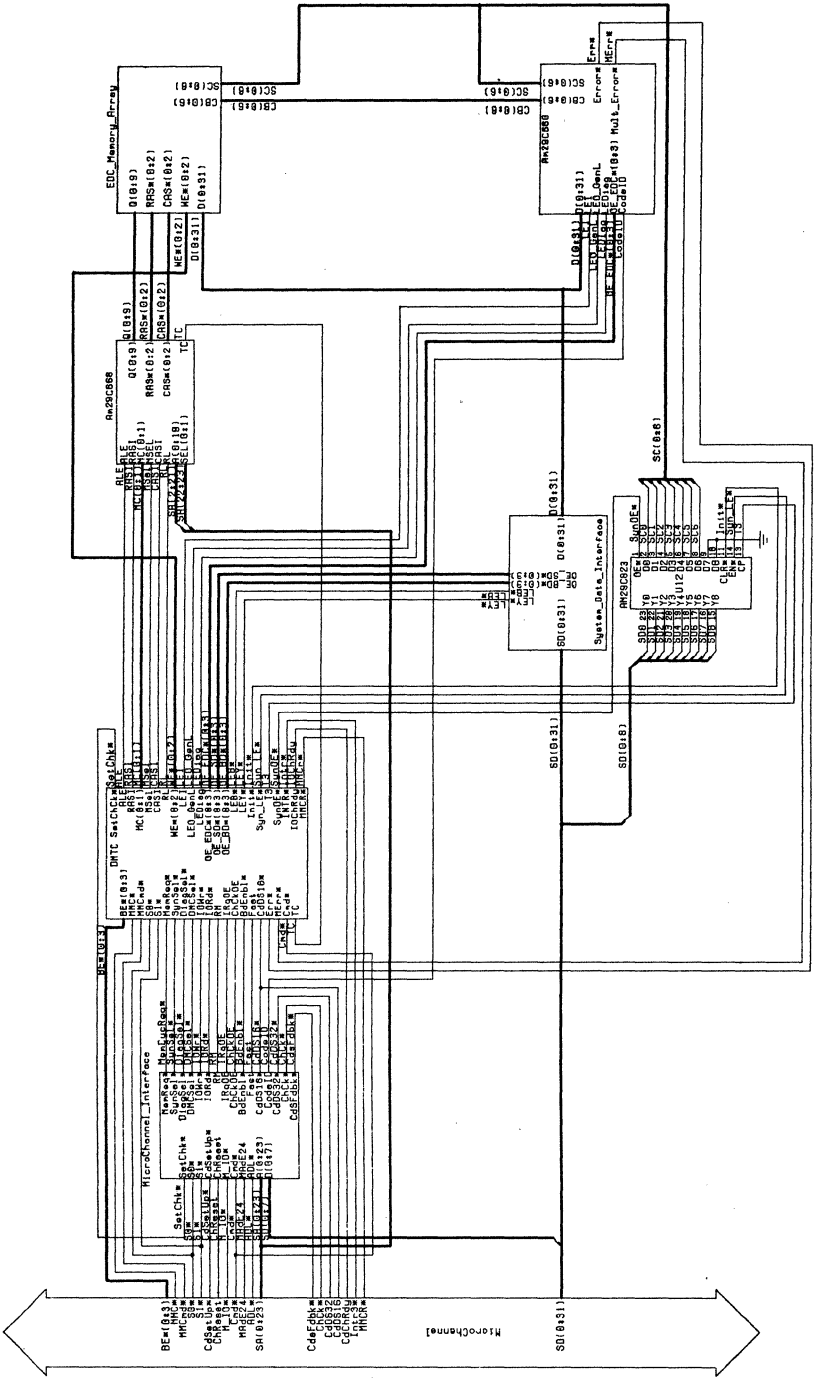
PARTS LIST (CONT.)

Unit #	Device	Description
U12	Am29C823	Syndrome Register
U13	EPB200I	Micro Channel Interface
U14	74ALS688	8-Bit Comparator
U15 -18	74F543	Octal Registered Transceiver

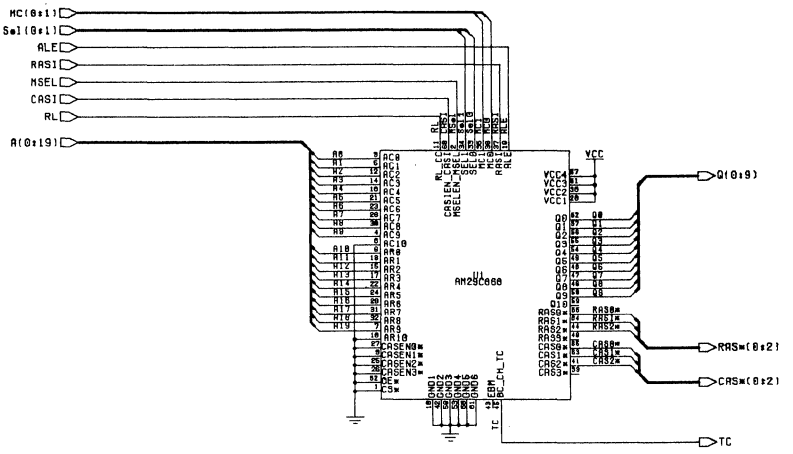
Description	Quantity per Board
CAPACITOR, 22 µF	7
CAPACITOR, 1.0 µF	1
CAPACITOR, 0.33 µF	9
CAPACITOR, 0.1 µF	28
CAPACITOR, 0.01 µF	1
RESISTOR PACK, 10 PIN SIP, 1 kΩ	1
DRAM MODULES, 1 M x 9	12
DRAM, ZIP PACK, 1 M x 1	9
Am29C660C	1
Am29C668	1
74F543	4
Am29C823	1
Am16L8	1
Am20RA10-20	2
Am22P10B	3
Am22V10-15	2
DELAY LINE, 10 ns, DIP-14	4
SOCKET, 14-PIN DIP	4
SOCKET, 24-PIN DIP	8
SOCKET, 68-PIN PGA/PLCC CONV.	4

SCHEMATICS

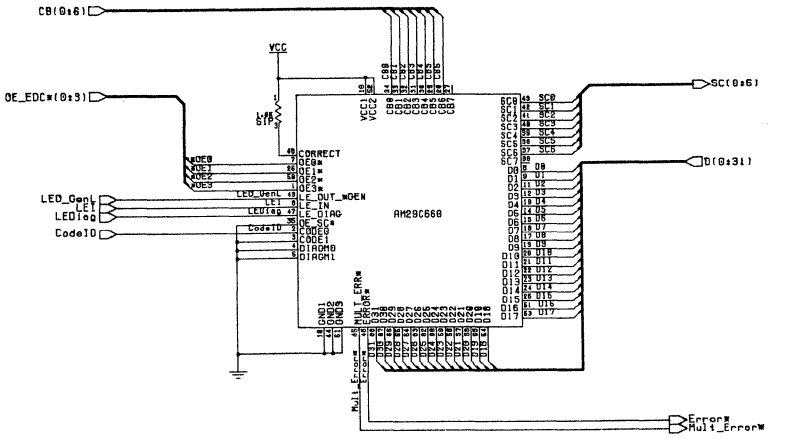
Detailed schematics follow.



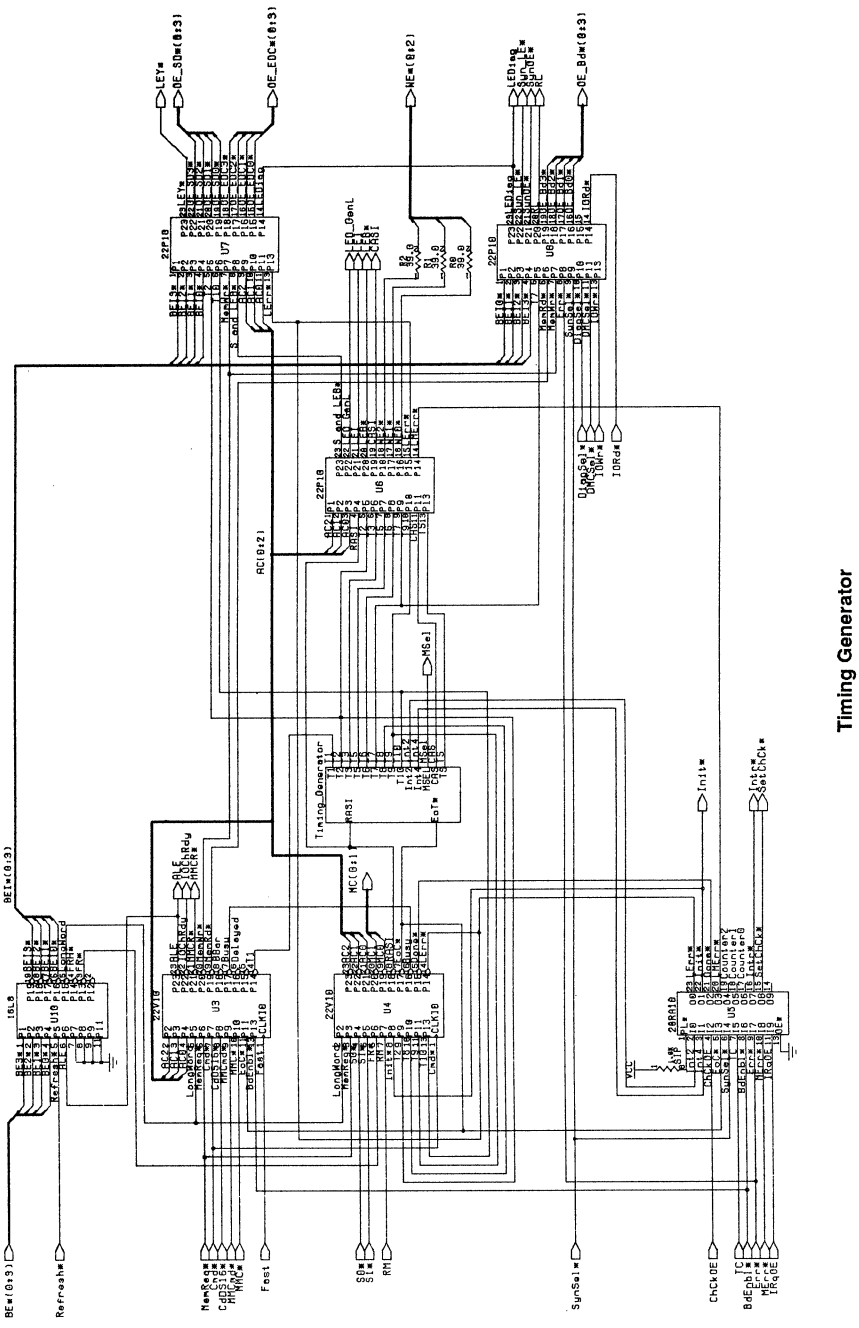
Top Level Schematic

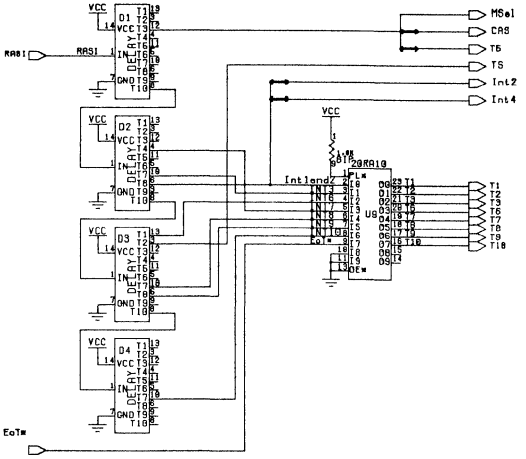


Am29C668 Dynamic Memory Controller

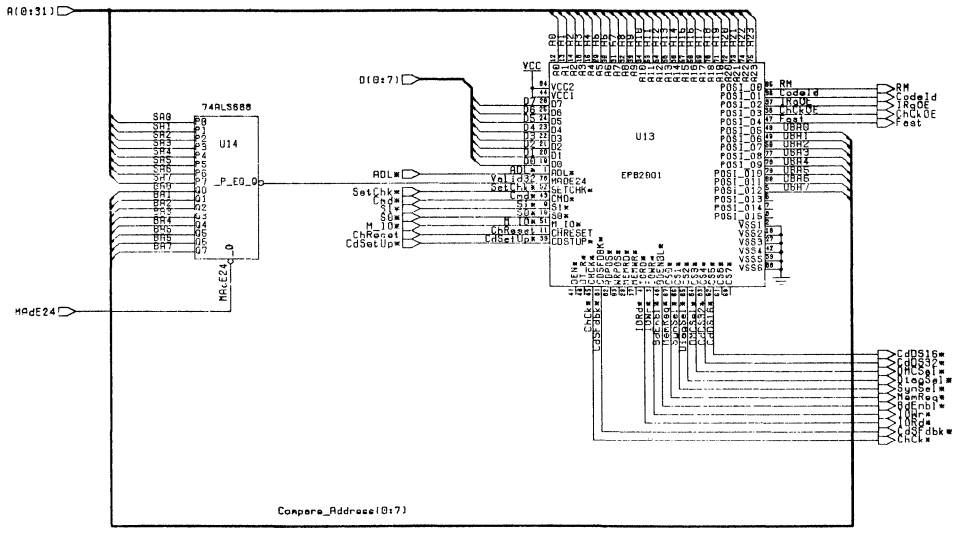


Error Detection and Correction

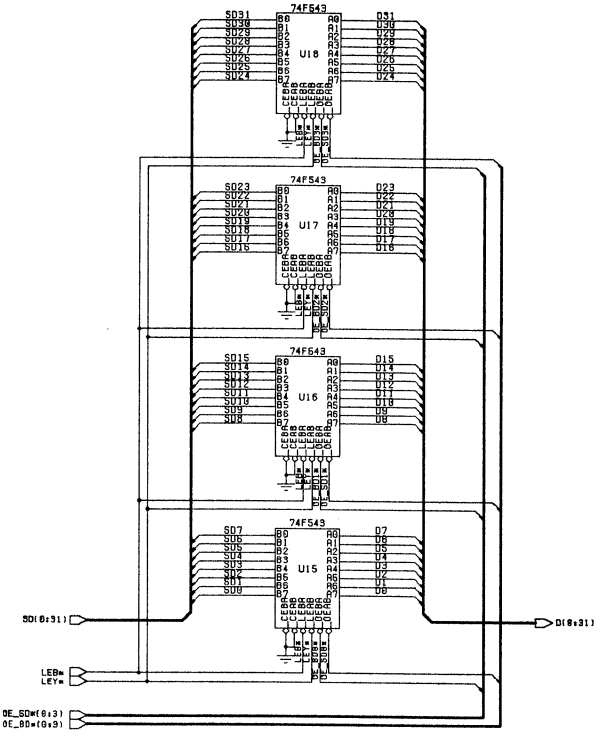

Timing Generator



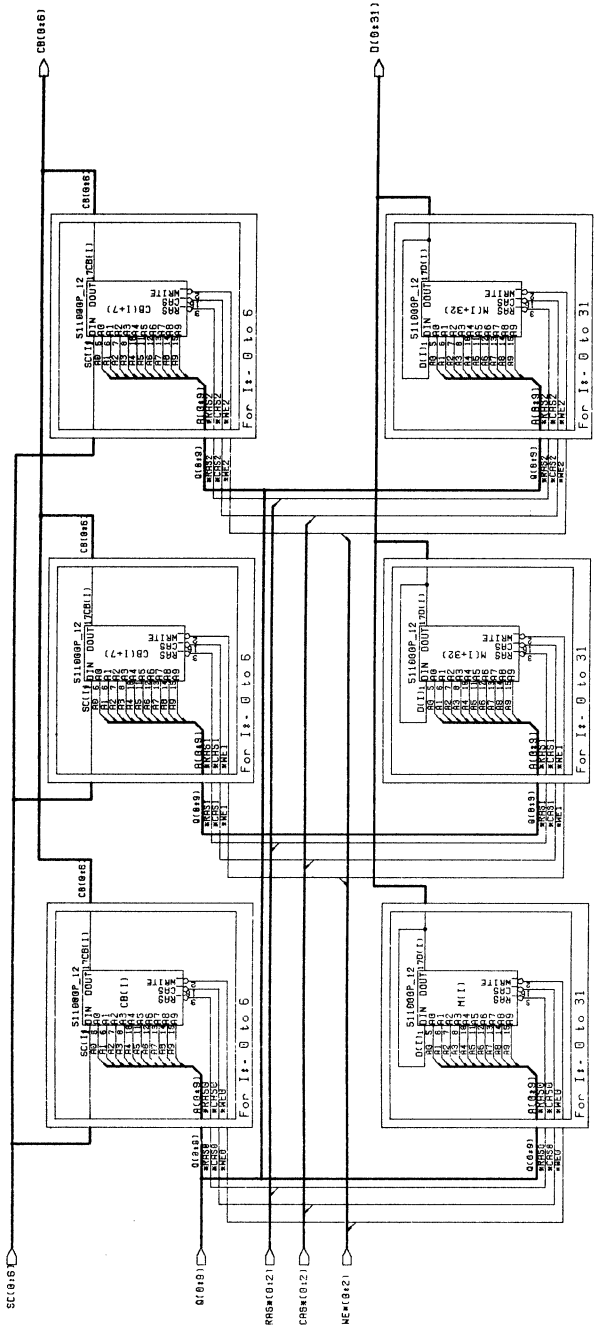
Micro Channel Interface



System Data Interface



EDC Memory Array



Dynamic Memory Timing Controller

CHAPTER 5
Special Applications and Article Reprints

Introduction 5-2
Configurable DRAM Controller Enhances System Performance 5-3
Four-Megabit DRAM Controller Offers Burst Addressing 5-11
Expand or Shrink Clock Cycles to the System's Needs 5-15
High-Speed VCEP Demonstration Board Using the Am29C668 CDMC 5-19

INTRODUCTION

This chapter contains three article reprints, two of which were originally printed in European publications and translated for use in this handbook. These two articles discuss the Am29C668 Configurable Dynamic Memory Controller (CDMC) in detail. The third article is a reprint from Electronic Design describing a clock-generator circuit for adaptive clocking using the Am2971A Programmable Event Generator (PEG) and an AmPAL18P8 PAL device.

A special application article describes a demonstration board using the Am95C71 Video-Data Compression/Expansion Processor (VCEP) and the Am29C668 CDMC. The board requires a dedicated memory buffer to hold compressed images, which is designed using DRAMs controlled by the Am29C668 CDMC.

Configurable DRAM Controller Enhances System Performance

by Percy R. Aria, Senior Product Planner

INTRODUCTION

With today's evolution of fast processors and RISC architectures, memory speed is a major factor in system throughput. Very fast memory in the form of static RAM looks attractive at first glance, but as the memory size increases, it becomes far less attractive due to prohibitive cost per bit compared to dynamic RAM. DRAM requires more complex control compared to SRAM, but cost per bit and high densities make DRAM very attractive for a wide range of applications.

The Am29C668 4-Mbit Configurable Dynamic Memory Controller greatly simplifies DRAM design. Because the Am29C668 is highly integrated as well as configurable, it can be used with virtually any processor in any system architecture. A block diagram of the Am29C668 is shown in Figure 1.

OPERATING MODES

The Am29C668 has two basic modes of operation, Read/Write and refresh; the timing diagram is shown in Figure 2. In the Read/Write mode, the Am29C668 latches the column, row and bank addresses. It then multiplexes the row and column addresses to the DRAMs under the control of either internally generated timing signals (auto-timing mode) or externally generated input signals.

The row address is latched in the DRAMs by the active (Low-going) edge of the Row Address Strobe $\overline{RAS}$ output, which follows the active (High-going) edge of the Row Address Strobe Input RASI. The address lines are then switched to the column address by either an internally generated signal if auto timing is selected or by pulling the Multiplexer Select MSEL signal active High if external timing is selected.

Read/Write Mode Optimization

The Read/Write mode of the Am29C668 may be optimized for the shortest DRAM access time in three different ways, depending on the system environment, software requirements and hardware configuration. These are Burst/Block-Mode Access, Cache-Mode Access and Bank-Interleave Mode Access.

Burst/Block-Mode Access

In this access mode, the Am29C668 can operate with processors that request burst accesses. Burst/block transfer is used by a high-performance processor to fill

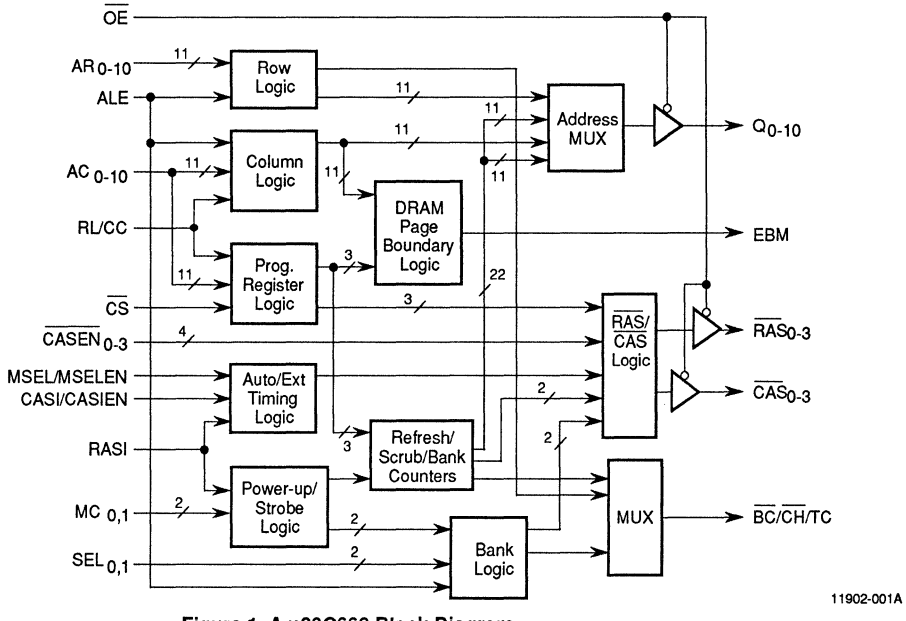


Figure 1. Am29C668 Block Diagram

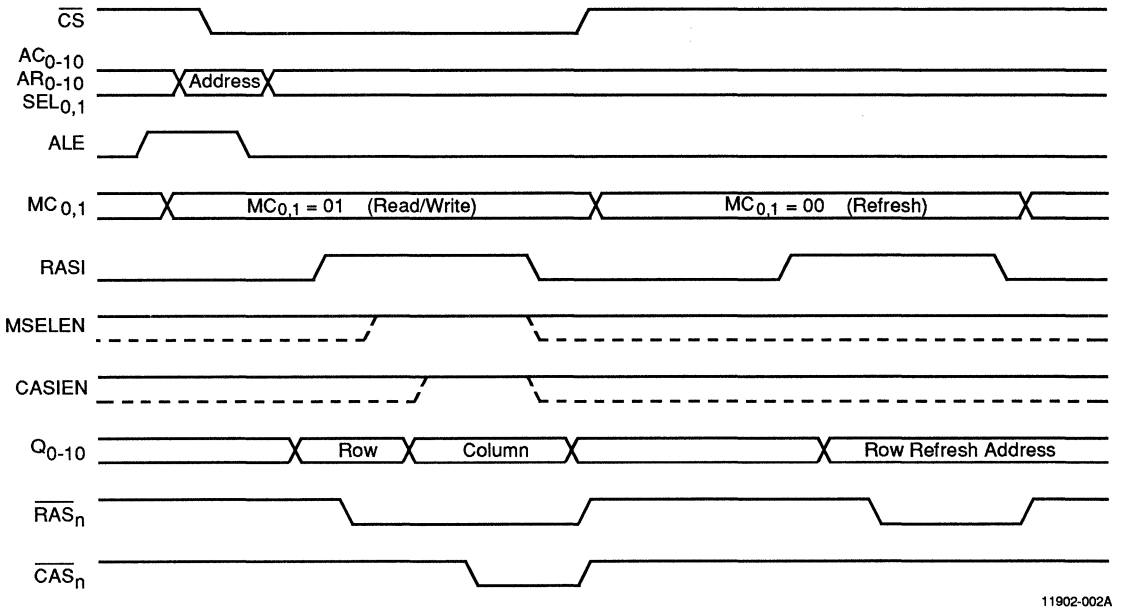


Figure 2. Read/Write and Refresh Operations

transfer is used by a high-performance processor to fill the cache, when a cache miss is detected; it is also used to support data pipelining.

During a burst/block access, the processor generates an address and expects to access consecutive locations starting at that address. The Am29C668 latches the address from the processor for the initial access to the DRAMs. The higher order address lines are latched as the row address and the lower order address lines are latched as the column address. The on-chip incrementer in the column-address logic is then used to generate subsequent addresses for consecutive accesses.

The Column Clock CC signal is used to increment the column-address logic. The RASI input is held active during the entire burst operation and only Column Address Strobe Input / Column Address Strobe Input Enable (CASI/CASIEN), depending on whether external or internal timing is used, is toggled to latch the column address for access to the DRAMs.

The column-address incrementer is designed to increment on the High-to-Low edge of the CC signal and CASI/CASIEN is an active High signal; therefore, for simple timing, the CC and CASI/CASIEN inputs may be tied together. As a result, the column address increments for the next access at the end of the DRAM access, when the CASI/CASIEN signal is deactivated. Figure 3 shows the timing for this type of access.

Additional support for the burst/block access consists of a programmable burst, limited only by the DRAM page size, and DRAM page-boundary detection logic. Two registers, the Burst Count Register and the Mask Register are loaded with the appropriate values via the AC_{0-10} address lines. The contents of the burst-count register are compared with the contents of the column-address incrementer on a bit-by-bit basis; the contents of the mask register determine which bits take part in the comparison. When a match of the comparison occurs, the Am29C668 asserts an End Burst/Block Mode (EBM) signal and the processor is expected to terminate the current burst access, generate a new address and re-establish a subsequent burst access.

This feature also has a self-aligning property at bit boundaries. For example, if the burst count is set at 16 and the initial access in a burst starts at 10, the first burst consists of six accesses and all subsequent bursts will consist of 16 accesses as shown in Figure 4. A detailed diagram of the programmable-register logic and EBM generation is shown in Figure 5.

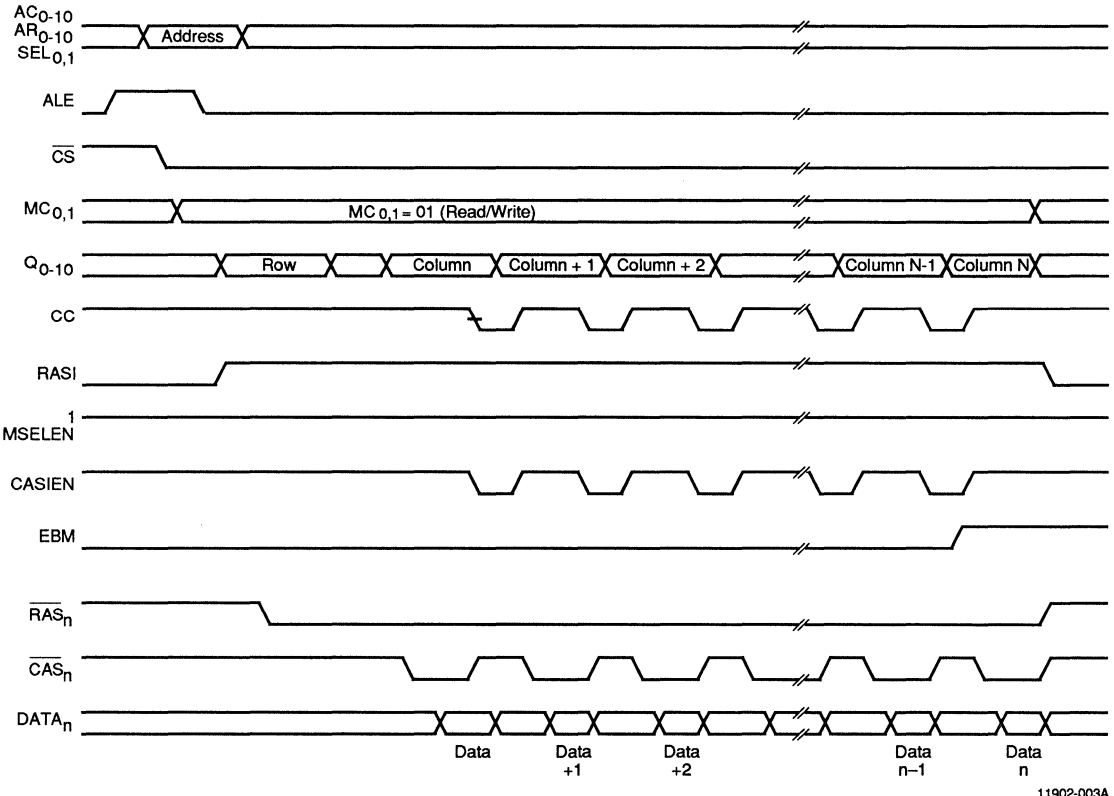


Figure 3. Burst Mode Access Ended by the Am29C668 (Auto Timing with External Override)

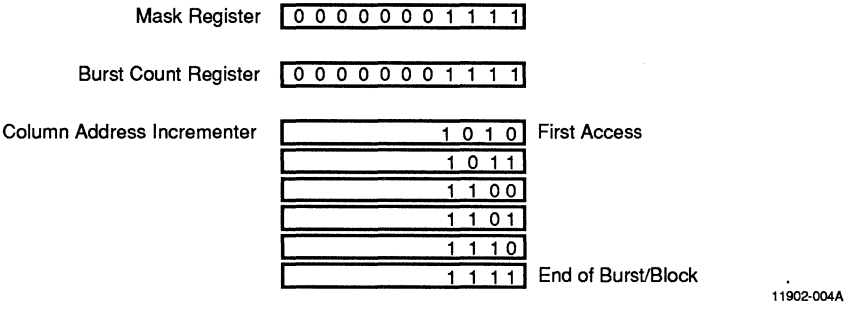


Figure 4. Burst-Mode Self-Alignment Example

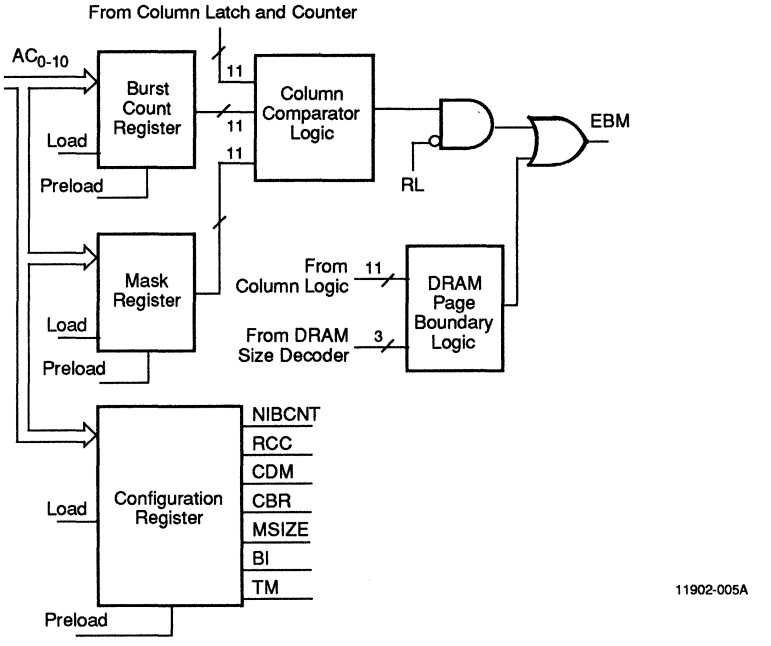


Figure 5. Programmable Register Logic and EBM Generation

Cache-Mode Access

In this mode, the Am29C668 can open a pseudo cache, the size of the DRAM page, and make fast random accesses to locations within the page. This access mode eliminates the $\overline{RAS}$ precharge time and the $\overline{RAS}$ access time from the DRAM cycle time, once the first access is completed. The Am29C668 therefore enhances the performance of the processor and the DRAMs, since these accesses appear to be normal accesses to the processor but are actually much faster page accesses.

On-chip comparators on the Am29C668 row and bank logic compare the row and bank addresses of consecutive accesses. When a match occurs, the Cache Hit $\overline{CH}$ signal is activated and the $\overline{RAS}$ strobe to the DRAMs is not deactivated at the end of the access, since there is no change in the row address for the next access. The new column address is latched into the DRAMs with the $\overline{CAS}$ strobe. Figure 6 shows a timing diagram for this type of access. The data from the DRAM is then available after the $\overline{CAS}$ access time of the DRAMs. The $\overline{CH}$ signal remains active as long as there is a match on the comparison of the row and bank addresses of subsequent accesses. When a mismatch occurs, the $\overline{CH}$ signal is deactivated and the external timing generator deactivates the RASI signal, which in turn latches the new row and bank addresses for future comparison. The $\overline{RAS}$ signal to the DRAMs is also deactivated and goes through precharge before starting the new access.

Bank-Interleave Mode Access

In this access mode, the Am29C668 can save the $\overline{RAS}$ precharge time between consecutive accesses to different DRAM banks, by overlapping the $\overline{RAS}$ precharge time of the previous access with the access time of the current access. This type of access therefore makes the $\overline{RAS}$ precharge time between accesses transparent; hence improves overall system performance. Typically, 100-ns access-time DRAMs have 80-ns $\overline{RAS}$ precharge time.

To take advantage of the bank-interleave mode access, the two LSBs of the processor address are tied to the SEL_{0-1} lines, which indicate the bank to be selected. Since a program flow is usually consecutive, this translates to accessing different DRAM banks for consecutive processor addresses.

In this mode, the Am29C668 makes a comparison of the consecutive DRAM bank addresses. When a mismatch occurs, the Bank Interleave BI signal goes active and the $\overline{RAS}$ strobe, generated by an external timing generator, to the new bank is generated right away, while the $\overline{RAS}$ strobe to the previous bank is going through a precharge. If, however, a match occurs, the BI signal goes inactive. This means that the current access is to the same bank as the previous access, in which case the $\overline{RAS}$ strobe is deactivated and goes through precharge before being activated for the current access.

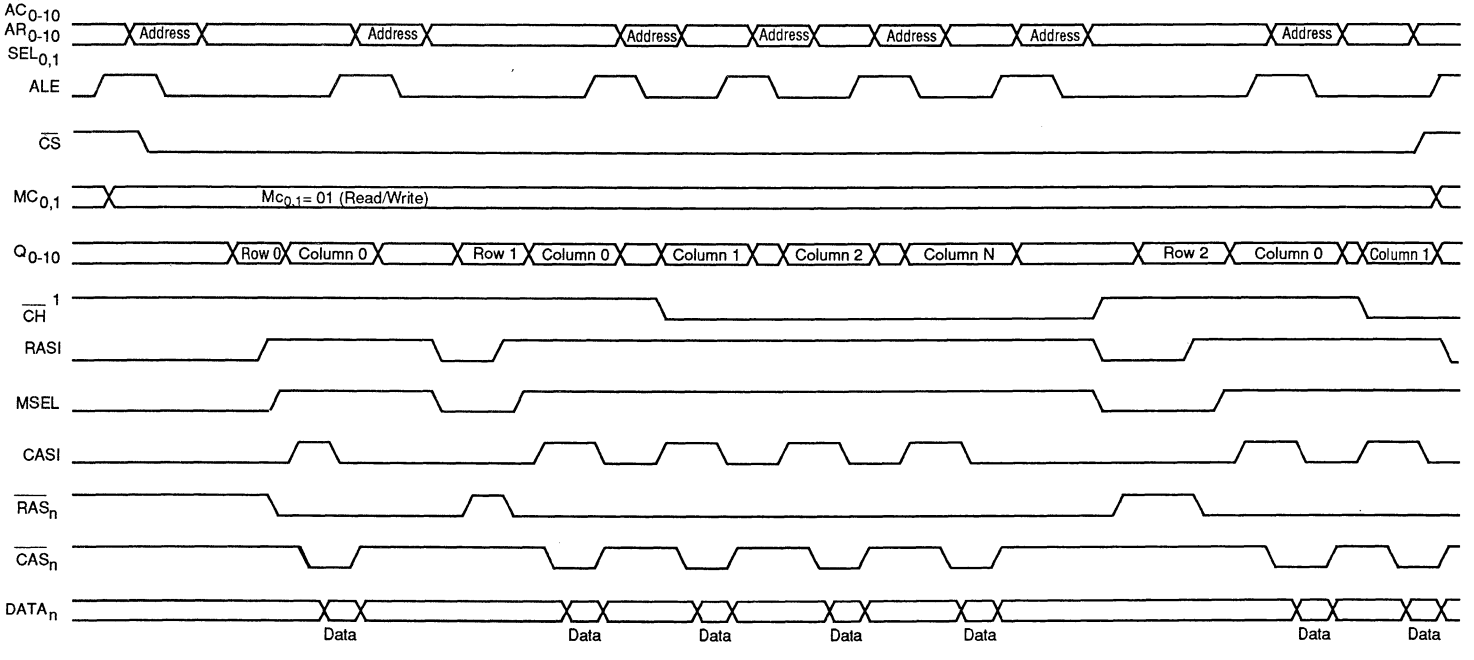


Figure 6. Cache-Mode Access with Page Mode DRAMs (External Timing)

11902-006A

Refresh Options

The Am29C668 has two refresh options for non-Error Detecting and Correcting (EDC) systems: $\overline{RAS}$-only refresh and $\overline{CAS}$-before-$\overline{RAS}$ refresh. The Am29C668 also offers a scrubbing option for EDC systems.

$\overline{RAS}$ -Only Refresh

During the $\overline{RAS}$-only refresh, the Am29C668 generates the appropriate refresh address and strobes the corresponding $\overline{RAS}$ to the DRAM banks to perform refresh. The Am29C668 refresh counter is incremented at the end of the refresh cycle to generate the address for the next refresh cycle.

$\overline{CAS}$ -Before-$\overline{RAS}$ Refresh

The Am29C668 also supports $\overline{CAS}$-before-$\overline{RAS}$ refresh, if this feature is available on the DRAMs. A bit in the Am29C668 Configuration Register, located within the Programmable Register Logic, is programmed to support this feature. The Am29C668 automatically activates the $\overline{CAS}$ and $\overline{RAS}$ strobes in the correct sequence during refresh. To the system timing generator, the input timing looks the same as the $\overline{RAS}$-only refresh timing if auto timing is used, or like the normal access timing if external timing is used. The Configuration Register is shown in Figure 7.

Refresh with 'Scrubbing'

The Am29C668 supports memory scrubbing in EDC systems. Scrubbing is a method of performing error detection and correction during refresh operations hidden from the processor. Scrubbing performs a read/modify/write operation on one location in the memory while refreshing the corresponding row on all the DRAMs. The Am29C668 has row, column and bank refresh counters, the lengths of which are automatically adjusted depending on the DRAM size being used.

By correcting single-bit errors, scrubbing prevents accumulation of multiple-bit soft errors, which are uncorrectable and cause system failure and down time. Another advantage to scrubbing is that during normal accesses, if an error is found, the data can be corrected and placed on the system bus but need not be written to the DRAMs. The scrubbing operation can write the corrected data back to the memory during the refresh operation. This helps improve the system performance, since the corrected data need not be written back to the memory when an error is found during normal accesses.

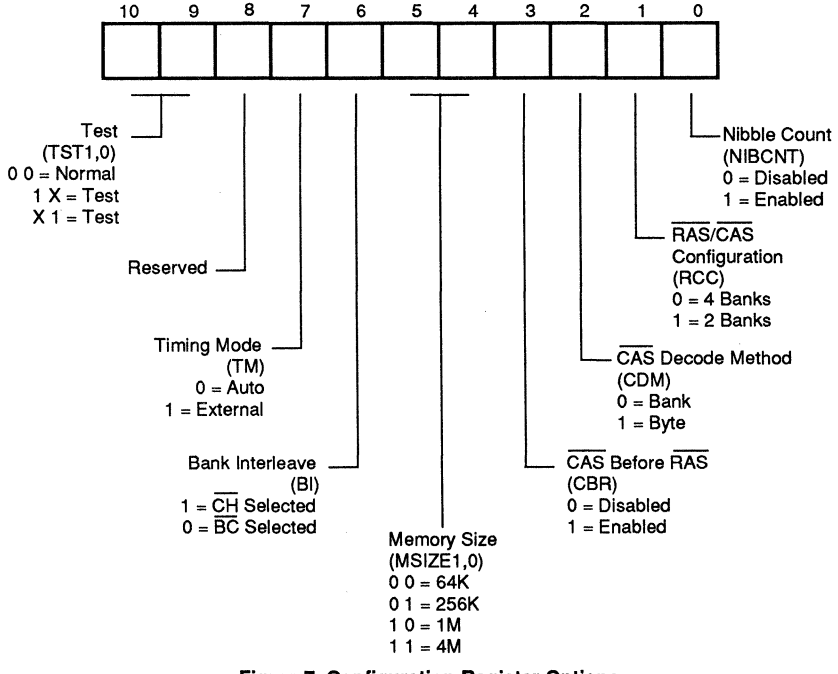


Figure 7. Configuration Register Options

Timing Options

Auto-Timing

In the auto-timing mode, the Am29C668 can be configured to generate its own internal MSEL and CASI timing signals for the output DRAM strobes ($\overline{RAS}_n$ and $\overline{CAS}_n$) from the input RASI signal. The auto-timing is optimized for 100-ns DRAMs.

External Timing

An external timing mode is also available so that the user can externally generate the MSEL and CASI timing signals for a specific application or DRAM.

Auto-Timing With External Override

A third option consists of auto-timing with an external override. In this mode, the MSEL and CASI inputs are defined as MSEL Enable MSELEN and CASI Enable CASIEN, respectively, and are ANDed with their respectively generated internal signals. This option can be used effectively for nibble-mode DRAMs, when the initial access can be made using auto-timing. The RASI signal is kept active and the CASIEN signal can simply be toggled to make the remaining three accessess of the nibble. A timing diagram of this type of an access is shown in Figure 8. A similar method can be used in the Burst/Block Access Mode and the Cache Access Mode.

OTHER DISTINCTIVE FEATURES

Nibble-Access Support

The incrementer on the Am29C668 can be configured to perform a modulo-four (nibble) count. A modulo-four burst access may be performed with a single address from the processor. Figure 9 shows a timing diagram for the nibble-mode access. This makes ordinary page-mode DRAMs look like nibble-mode DRAMs to the processor. Auto timing with external override is particularly well suited for this type of operation.

Configurable Drive Capability

The Am29C668 can be configured to drive two or four banks of DRAMs by programming a bit in the Configuration Register. By configuring it for two banks the drive capability of the $\overline{RAS}$ and $\overline{CAS}$ strobes is doubled.

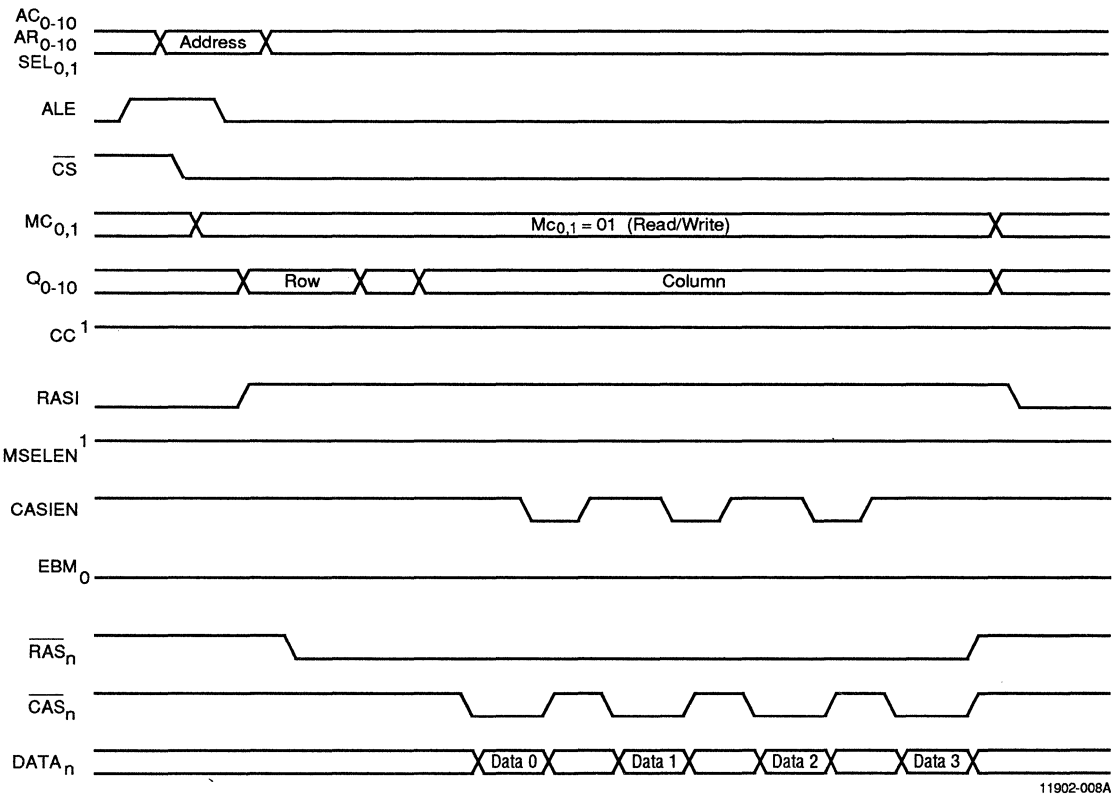


Figure 8. Nibble Mode Access with Nibble Mode DRAMs (Auto-Timing with External Override)

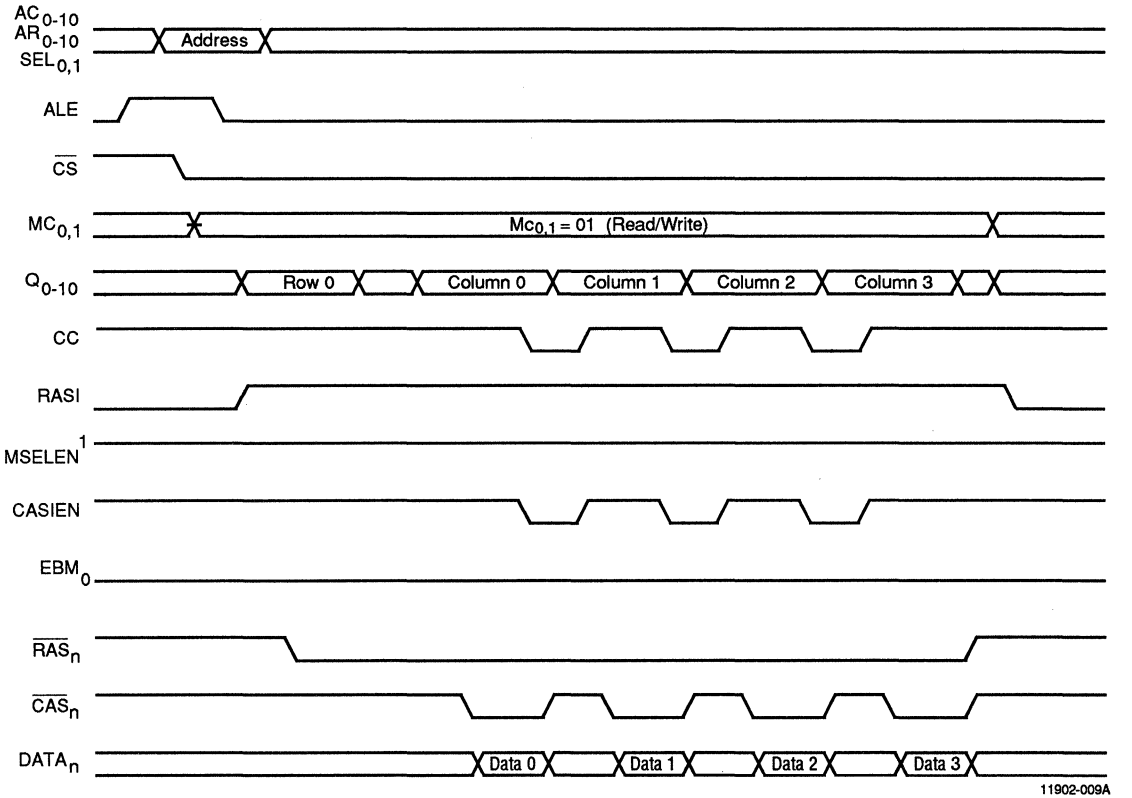


Figure 9. Nibble Mode Access with Page Mode DRAMs (Auto-Timing with External Override)

The Am29C668 has controlled outputs that limit the overshoot and undershoot to within acceptable limits of the DRAMs. The Am29C668 can directly drive four banks of 16-bit data word with EDC (22 bits total) or two banks of 32-bit data word with EDC (39 bits total). The Am29C668 can drive up to four banks of any data word size beyond 16 bits if used with external drivers.

Selectable $\overline{\text{CAS}}$ Decode Scheme

Two $\overline{\text{CAS}}$ decode schemes, available on the Am29C668, can be selected via a bit in the Configuration Register. The $\overline{\text{CAS}}$ can be decoded by bank to perform regular word, burst/block, cache, and bank-interleave accesses. The $\overline{\text{CAS}}$ can also be decoded on a byte boundary to perform byte operations (Figure 10).

When bank $\overline{\text{CAS}}$ decode is selected, $\overline{\text{CAS}}$ is decoded internally by the Am29C668. When byte $\overline{\text{CAS}}$ decode is selected, $\overline{\text{CAS}}$ is decoded externally and is placed on the four $\overline{\text{CAS}}$ Enable ($\overline{\text{CASEN}}_{0\text{-}3}$) inputs for the four bytes of a 32-bit data word.

CONCLUSION

The Am29C668 innovative architecture, versatility and advanced features provide improved memory throughput and enhanced system performance in virtually all DRAM-control applications. The auto-timing feature and drive capability make the Am29C668 an attractive, highly integrated dynamic memory-control solution for a wide range of applications.

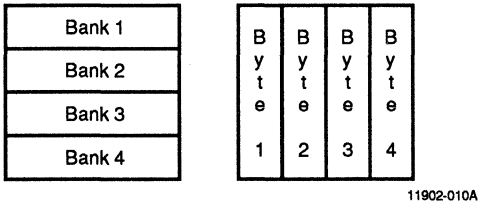


Figure 10. Bank-vs-Byte-Selective Memory Accessing

Four-Megabit DRAM Controller Offers Burst Addressing

by Bo Molander, Senior Field Applications Engineer

The advantages of dynamic random-access memory (DRAM) are lowest cost per bit compared to other semiconductor memories, and small packaging that requires minimum board space. Conversely, the best known drawback of DRAMs is that they must be refreshed every second or fourth millisecond to preserve the data stored in the array. In addition, to take advantage of the small package, memory row and column addresses must be multiplexed onto the input pins. Over the years, a number of different DRAM controllers have been introduced that make these drawbacks transparent to the DRAM user.

Initially, simple building blocks were offered, each device performing only a part of the total DRAM-control function. In time, however, more integrated solutions were presented. Soon, two major pathways were chosen: in the first, no timing-generation support was included in the memory controller; in the second, timing delays for the $\overline{RAS}$ and $\overline{CAS}$ strobes were generated within the controller. Both DRAM-controller types found applications dictated by system requirements. In the past, DRAM controllers with no internal timing generation were used extensively in systems with very fast memories requiring the shortest possible access times. A 1-Mbit DRAM controller, such as the Am29368, can be tailored to a specific memory system according to the number of memory devices, associated capacitance and the resulting propagation delays. For example, the Am29368 DRAM controller can be used very efficiently with 85-ns DRAMs.

A DRAM controller with internal timing generation provides a more compact solution with fewer devices. Unfortunately, speed and flexibility are sometimes sacrificed. These older DRAM controllers are simply too slow to keep up with today's dynamic memories that have access times from 60 to 100 ns.

The limitations of these early memory controllers have driven the system designer to solving memory-control requirements with PAL® devices or to other similar programmable devices. A simple memory-control function is easily implemented with a couple of PAL devices; but, more sophisticated features, like block transfers and burst addressing, increase the required number of PAL devices. The final memory controller could simply occupy too much board space.

The Am29C668 Configurable Dynamic Memory Controller/Driver (CDMC), shown in Figure 1, gives the memory-system designer a new alternative. It is fabricated using CMOS technology to provide very high speed and low power consumption, and can directly control DRAMs up to 4 Mbit in size. Timing generation can be provided either internally or externally, depending on the designer's requirements. The Am29C668 offers a very high drive capability — one device can directly drive four 16-bit wide banks of DRAMs plus the extra bits required for error detection and correction, or two 32-bit-wide banks plus check bits, without external buffers (Figure 2). As a result, memory-control systems can now be built with a minimum number of devices. Also, minimum access times can be achieved through burst addressing and by reading data from different memory banks using bank interleaving. The four independent CAS signals of the Am29C668 can be used to control four separate banks of DRAMs, or to determine the access of a certain byte within a 32-bit word.

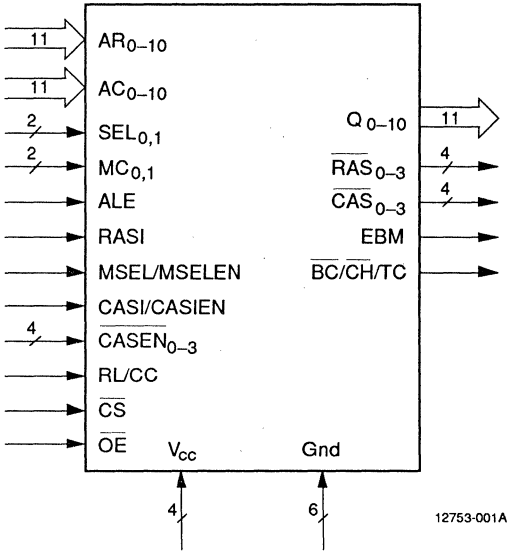


Figure 1. Am29C668 Logic Symbol

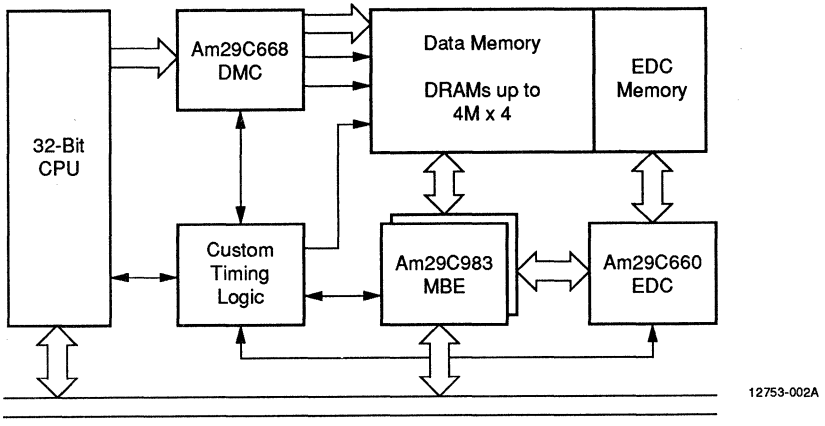


Figure 2. 32-Bit DRAM System

A BASIC SYSTEM

In a typical microprocessor system, the Column and Row Address inputs AC_{0-10} and AR_{0-10} of the Am29C668 are connected to the address bus coming from the microprocessor; the multiplexed Address outputs Q_{0-10} of the CDMC are connected to the memory address inputs. Since the CDMC only generates addresses and control signals to the memory banks, the critical data path is not affected. The following control signals must be connected to the memory banks: Address Latch Enable ALE that latches the address from the processor, Chip Select $\overline{CS}$, and Output Enable $\overline{OE}$. The $\overline{OE}$ signal can be used to control the output drivers, i.e., to determine whether they are to be active or in a high-impedance state. This feature provides for building larger systems with more than one Am29C668.

The RASI input is used to start an access cycle, independent of the timing mode selected, internal (Auto-Timing) or external. Selection of Auto or external timing generation is made via the dual-function Register Load/Column Clock RL/CC input and the Timing Mode TM bit in the configuration register. The Mode Control pins $MC_{0,1}$ are used to specify one of four modes: Read/Write, refresh with scrubbing, refresh without scrubbing, or reset.

Programmability

The configuration register, located within the Programmable Registers and Logic block of the Am29C668 (Figure 3), is loaded from the column-address bus with the input signal RL/CC. It can be programmed to select a number of options including DRAM size: 64K, 256K, 1 Mbit or 4 Mbit. With this wide selection, the CDMC can be used with the mass-produced DRAMs available today, as well as with the larger DRAMs of tomorrow. The refresh and scrubbing counters are automatically set to the correct count when the DRAM size is selected. The Column Address Strobe outputs $\overline{CAS}_n$ can be selected as bank or byte select. Also, many DRAMs support $\overline{RAS}$-before-$\overline{CAS}$ refresh, in which a refresh counter internal to the DRAM is used to select the next row in the memory array to be refreshed. The Am29C668 can be programmed to support this refresh scheme. Normally, the row address for refreshing is taken from a counter within the CDMC. Also, burst addressing is selectable when required for use with nibble-mode or page-mode DRAMs. For nibble-mode DRAMs, burst addresses are generated for four words at a time.

Timing Generation

As mentioned earlier, internal or external timing generation can be used with the Am29C668 CDMC. Internal timing generation (auto-timing mode) offers optimum performance when used with 100-ns memories (500-pF load); options exist for controlling $\overline{RAS}_n$ and $\overline{CAS}_n$ through gating with external signals. The auto-timing mode is selected via the TM bit and the memory-access cycle starts when RASI is activated. The row address is immediately available at the memory address inputs; at a specific time later, the four $\overline{RAS}_n$ signals go active so that the DRAM can latch this address. The row addresses presented to the DRAM change to column addresses, controlled by the internal timing delay and gated with the Multiplexer Select MSEL input, or in case of external timing generation, controlled only by MSEL. After a delay, the $\overline{CAS}_n$ outputs from the Am29C668 go active, signaling to the DRAM that the $\overline{CAS}_n$ addresses are stable.

The $\overline{CAS}$ timing is either generated internally in the auto-timing mode and enabled with the CASI input, or

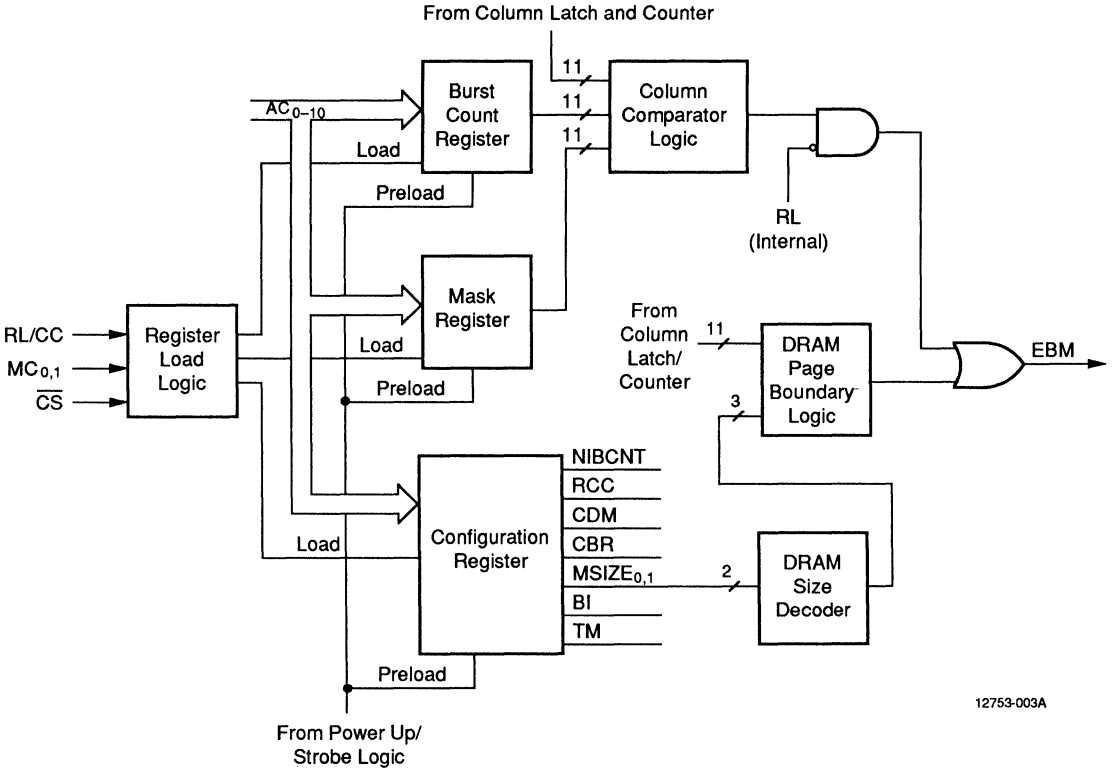


Figure 3. Programmable Registers and Logic

controlled only by the CASI input in the external-timing mode. Optimum timing varies with the speed of the DRAMs. Internal timing generation can be used with 100-ns memories without external drivers. The CDMC can be used with faster memories, in the 60-to-90 ns range, using delay lines or PAL devices for external timing control.

Memory-System Refresh

The Am29C668 supports three different types of refresh: normal, $\overline{CAS}$ before $\overline{RAS}$, and refresh with scrubbing. During normal refresh, the $MC_{0,1}$ inputs initiate the refresh cycle. The internal refresh counter, automatically adjusted for the size of the DRAM, outputs a refresh address. At the same time, all four $\overline{RAS}$ signals go active, signaling to the DRAM that a refresh cycle has begun.

$\overline{CAS}$-before-$\overline{RAS}$ refreshing, used with DRAMs that have on-chip refresh counters, is accomplished by changing the normal order that $\overline{RAS}_n$ and $\overline{CAS}_n$ are presented to the memory. This refresh support is selected by enabling the $\overline{CAS}$-before-$\overline{RAS}$ bit in the configuration register; the Am29C668 then outputs all four $\overline{CAS}_n$ signals, followed by the four $\overline{RAS}_n$, indicating to the DRAM that a $\overline{CAS}$-before-$\overline{RAS}$ refresh cycle has started.

The refresh with scrubbing mode is used when an error detection and correction (EDC) device, such as the Am29C660, is used in the system. One word is read from the memory system during a refresh, passed through the EDC and written back, completely transparent to the microprocessor. In this way, the entire memory bank can be automatically cleared from erroneous bits.

Burst Addressing

One of the most significant improvements in the Am29C668 over older-generation memory controllers is the availability of burst-addressing protocol for the memory banks. Burst addressing is used by some very fast RISC microprocessors, such as the Am29000, and in most cache-memory systems. During burst addressing, the microprocessor, or cache controller, sends only the first address when a consecutive block of data is being accessed. The memory system then latches this address and, via handshaking, sends or accepts data at address n, address n+1, n+2, etc. All the addresses,

except the first, must be generated outside the processor; therefore, a counter is required in the memory system. The Am29C668 includes all the necessary latches and counters to support burst-mode addressing, making external logic unnecessary.

Burst addressing has two distinctive advantages over traditional schemes. First, it reduces memory access time because the row address normally does not change during a burst operation when data is read sequentially. The $\overline{RAS}$ signal, therefore, need not be deactivated and then reactivated, saving both $\overline{RAS}$ precharge and $\overline{RAS}$ access time. The second advantage is that the microprocessor address bus is available for the other tasks during a burst access.

A slightly different way to perform burst operations is through block transfers. A predetermined number of words, set by the Am29C668 on-chip counter, are transferred. This mode is used, for example, by DEC's Q-bus. In another version of burst addressing, used by Motorola's 68030, the microprocessor always sends one address and expects four words (a nibble) back. This mode, also supported by the Am29C668, can be used with nibble-mode or standard page-mode DRAMs.

Overlapping Accesses

One method to reduce access time in a memory system is to use two or more memory banks and overlap accesses between banks, i.e., bank interleaving where the Read/Write cycle to bank n+1 starts before the access to bank n is finished. This technique saves $\overline{RAS}$ precharge time t_p which is normally in the 80–90 ns range for 100-to-120-ns DRAMs (Figure 4). To use overlapping, or bank interleaving, the two lower address bits from the microprocessor are connected to the SEL_{0-1} inputs of the Am29C668 and the accesses are then distributed among all four banks.

Special DRAM Types

The Am29C668 CDMC supports other special DRAM types — static column, ripple and page mode. For these DRAM types, the RL/CC input on the Am29C668 is used with the signals described above to control memory system accesses.

CONCLUSION

The Am29C668 CDMC offers an integrated, flexible solution to DRAM control. While the device is designed specifically to support fast, modern 32-bit microprocessors, it can be used with all DRAM types due to its built-in special functions. The combination of features, low power consumption and high speed, makes the Am29C668 a natural choice in any microprocessor system.

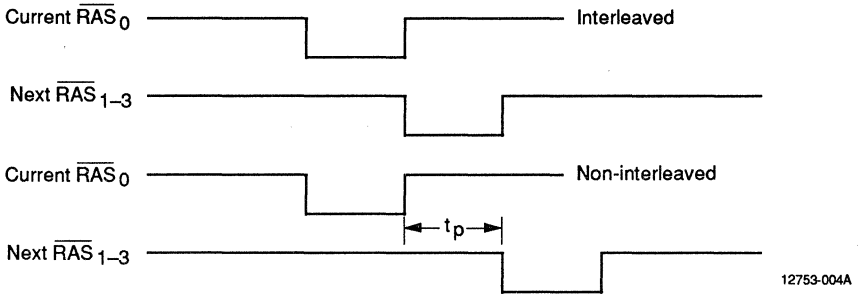


Figure 4. Am29C668 Enhancement Bank-Interleave Feature

Expand or shrink clock cycles to the system's needs

Rick Purvis and Jenny Yee
Advanced Micro Devices Inc., 901 Thompson Pl., Sunnyvale, CA 94088; (408) 732-2400.

In a microcoded system, tailoring the length of each microcycle to an operation makes the most of system performance. The clock cycle should be as short as possible and lengthened only when necessary to accommodate slow paths. Controlling clock cycle length dynamically revs up system performance; microcoded CPUs and I/O processors are examples.

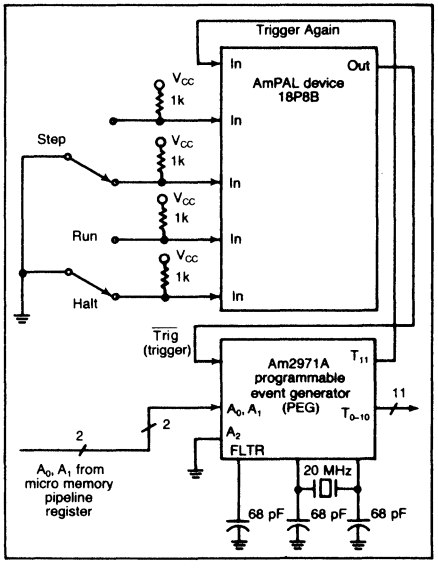
Sophisticated system designs, however, can't be built with the frequencies and waveforms from conventional clock generators like the Am2925A. But a clock-generator circuit (*Fig. 1*) consisting of an Am2971A programmable event generator (PEG) and an AmPAL18P8 PAL device can give the user much more flexibility in controlling both frequency and waveforms (see "A programmable event generator," p. 95). How many times have you needed just one more clock edge, 10 ns sooner or later? The PEG circuit can supply it. In fact this technique can also be applied to most general-purpose microprocessor systems. What's more, the event generator chip can serve any application that requires very small, fast state machines.

Choose 80, 100, or 120-ns system clock cycles or sequence up to twenty-six 11-bit machine states on the fly by using adaptive clocking.

The PEG-based design, though similar in function to the Am2925A, has more flexibility because it has 11 independent, user-defined outputs (T_{0-10}) and an internal clock of 100 MHz. The result is a 10-ns resolution between output edges and user-definable cycle lengths such as 80, 100, or 120 ns, selected by event generator inputs A_0 and A_1. The circuit's PAL device supplies debounced run/halt and step functions, and an input multiplexer for the PEG trigger input.

Primarily, the event generator operates as a 100-MHz synchronous state machine to produce output system-clocking waveforms at around 10 MHz. An external 20-MHz crystal controls the 100-MHz internal clock. The circuit can free run or single step. With the run/halt switch in the run position, the PEG state machine receives feedback trigger (T_{11}) pulses to generate continuous system clocks. When in halt, each step-switch activation directly triggers the PEG through the Trig input to generate one system clock cycle.

The PEG circuit can implement three independent state machines—one for each of the selectable cycle lengths (*Fig. 2*). When triggered by the falling edge (depending on the event generator's polarity setting) of a Trig signal, the state machine, in accordance with the cycle-length inputs A_1 and A_0, be-



1. With a clock-generator circuit consisting of an Am2971A programmable event generator (PEG) and an AmPAL18P8 PAL device, a user has the flexibility for controlling both frequency and waveforms. Often the designer needs just one more clock edge 10 ns sooner or later; the PEG circuit can supply it.

Reprinted with permission from Electronic Design, Vol. 35, No. 28;
©Copyright Hayden Publishing Co., Inc., November 27, 1987.

gins operation at the event generator's 100-MHz internal rate. Each current state points to the next state, in turn. System-clock outputs from the circuit come from the PEG outputs T_{0-10}.

The event generator must be in a stopped mode to be triggerable, however. Therefore, it stops after generating each system clock cycle, waiting for a trigger signal from either the T_{11} feedback signal in the free-running mode or the next step transition. In the free-running mode, sequencing continues until the event generator encounters a programmed stop bit.

State machine 1: 80-ns system clock cycle (A1, A0 = 0)

Current state:	0	1	2	3	4	5	
Next state:	1	2	3	4	5	6	
Outputs T_0:	1	1	1	1	0	0	
T_1:	1	1	1	0	0	0	
T_2:	1	1	0	0	0	0	
T_3:	1	0	0	0	0	1	
.							
.							
T_{11}:	0	1	1	1	1	0	
Stop bit:	0	0	0	0	0	1	

State machine 2: 100-ns system clock cycle (A1, A0 = 1)

Current state:	8	9	10	11	12	13	14	15
Next state:	9	10	11	12	13	14	15	16
Outputs T_0:	1	1	1	1	1	0	0	0
T_1:	1	1	1	1	0	0	0	0
T_2:	1	1	1	0	0	0	0	0
T_3:	1	1	0	0	0	0	0	0
.								
.								
T_{11}:	0	1	1	1	1	1	1	0
Stop bit:	0	0	0	0	0	0	0	1

State machine 3: 120-ns system clock cycle (A1, A2 = 3)

Current state:	16	17	18	19	20	21	22	23	24	25
Next state:	17	18	19	20	21	22	23	24	25	26
Outputs T_0:	1	1	1	1	1	1	0	0	0	0
T_1:	1	1	1	1	1	0	0	0	0	0
T_2:	1	1	1	1	0	0	0	0	0	0
T_3:	1	1	1	0	0	0	0	0	0	0
.										
.										
T_{11}:	0	1	1	1	1	1	1	1	1	0
Stop bit:	0	0	0	0	0	0	0	0	0	1

2. The PEG circuit can implement three independent state machines: one for each of the selectable cycle lengths. In the free-running mode, sequencing continues until the event generator encounters a programmed stop bit.

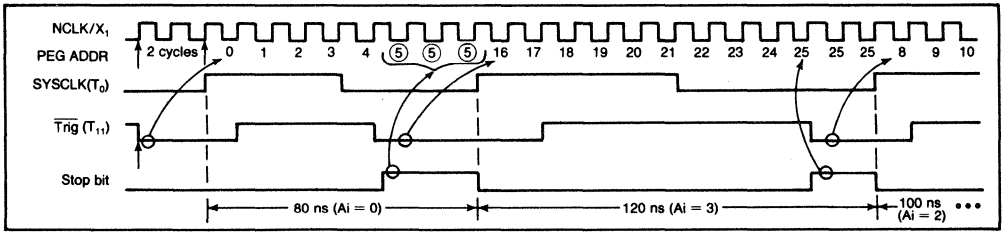
In the timing diagrams for free-running operation (*Fig. 3*), the internal event-generator clock is $NCLK/X_1$, where N is the PEG's programmable clock multiplier. Since the internal clock frequency can be 1, 5, 10, 5/2, or 5/4-times the input clock or crystal frequency, a 20-MHz crystal and a 5x clock setting supplies the 100-MHz internal clocking. The contents of the output register is the current-state entry. States with the stop bit set (such as PEG address 5) remain in the output register until T_{11} triggers the event generator again.

Also, two internal PEG cycles must occur after a trigger until the first state of the next state machine appears in the output register. This two-cycle delay after triggering results from internal event-generator pipeline delays. In this application of the event generator, that means the last state of a particular state machine repeats for three internal clock cycles before the PEG can trigger again. That explains why T_0 shows a 50-50 duty cycle (*Fig. 3 again*), though a high exists in states 0, 1, 2, 3, and low only for states 4, 5 *(Fig. 2 again)*.

The details of designing the clock generator (*Fig. 1 again*) serve as a good example of that procedure. First, however, consider the design of a simple continuously running 10-MHz system clock with 50-50 duty-cycle output (*Fig. 4*). Though the 100-MHz (10-ns cycle) internal PEG clock (labeled $NCLK/X_1$) is not available outside the PEG, its timing wave is a useful design construct. The T_0 SYSCLK output requires 10 of these internal event generator cycles; thus, T_0 is high for the first five $NCLK/X_1$ clock cycles and low for the next five.

Arbitrarily assign all start addresses to state 8 and have each successive state point to the next. This means that state 8's next state is 9, and so on until state 17. State 17's next state is 8 to achieve a loop back to the beginning of the state machine.

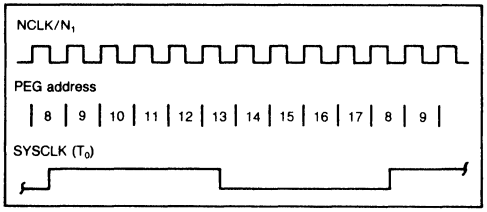
Thus far, the display is that of a trivial divide-by-10 counter. Adding a single-step capability would enhance the design. Each Step input would trigger a single T_0 sys-



3. In free-running operation, the internal PEG clock is $NCLK/X_1$, where N is the programmable clock multiplier. Since the internal clock frequency can be 1, 5, 10, 5/2, or 5/4-times the input clock or crystal frequency, a 20-MHz crystal and a 5x setting supplies 100-MHz internal clocking.

tem-clock cycle with the PEG stopping at the end of each T_0 cycle.

Then add the halt/run switch and AmPAL18P8B. Now, the circuit becomes a 10-MHz generator that can operate both in free-running and single-step modes. Similar to the condition in state 5 in *Fig. 2*, in the 100-MHz generator, both the Stop bit and T_{11} are active during state 15. The PEG stops, and its output remain in state 15 for two additional cycles before going to state 8, when triggered by the T_{11} feedback.



4. In the timing diagram of a simple, continuously running 10-MHz system clock with 50-50 duty cycle output, state 8 through 17 represents one cycle. Thus, T_0 is high for the first five clock cycles.

The final embellishment adds state machines for 80 and 120 cycles. It consists of merely drawing additional T_0 and T_{11} waveforms, adding the two stop bits, and remapping the starting addresses. The result is the clock generator of *Fig. 1*, with the PAL device fully programmed. Outputs T_{0-10} are available for user-defined clock waveforms.

Note: It's helpful to program unused event generator outputs to match the next state address to enable PEG state sequencing to be observed. Otherwise, the blind debugging of a state machine could prove difficult.□

Rick Purvis is a field application engineer with Advanced Micro Devices in Austin.

Jenny Yee, formerly with Advanced Micro Devices, is currently a compiler library manager at VLSI Technology Inc. in San Jose, Calif.

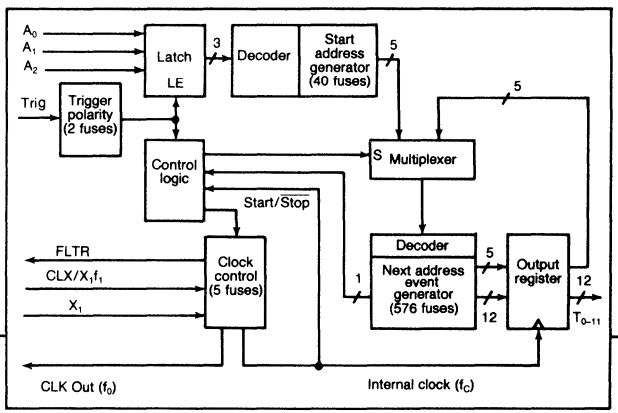
A programmable event generator

The Am2971A programmable event generator (PEG) is a fast registered PROM optimized for state-machine and timing-generation applications. It can serve as a digital substitute for analog delay lines or as a general-purpose programmable timing-waveform generator. Besides supplying 12 outputs, it can implement state machines having up to 32 states. It operates at any internal clock frequency from dc to 100 MHz. The clock control is a programmable phase-locked loop that can operate at up to 100 MHz from a 20-MHz source.

The PEG consists of four main functional blocks (see the figure): clock control; start-address generator; next-address/event generator; and control logic. The next-address/event generator is a PROM containing 32 words, 18 bits each: a stop bit, 5 bits of next-state information, and 12 bits for outputs T_{0-11}. That PROM and the output register form a classical registered-PROM state-machine architecture, with 5 bits of internal next-address feedback.

The start-address generator is a mapping PROM for the three external inputs A_{0-2}. One of eight start addresses may be selected to begin the timing sequence. The procedure for programming both event generator PROMs is the same as for any conventional PROM. The control logic makes it all work together.

To initiate a sequence, a programmable positive/negative edge transition at the Trig input latches the three inputs A_0 to A_2, which the logic then maps through the start-address generator to one of eight 5-bit starting addresses for the state machine. Each successive internal clock strobes both output data and the next address into the output register. This operation continues until the PEG encounters a stop bit or the other edge of Trig (programmable option). The company supports the event generator with design software (AmPEGPDS0) and programming hardware (AmPEGASUS0).



High-Speed VCEP Demonstration Board Using the Am29C668 Configurable Dynamic Memory Controller

by Vineet Dujari, Applications Manager

VCEP OVERVIEW

The Am95C71 Video Data Compression/Expansion Processor (VCEP), Figure 1, performs CCITT(T.4 and T.6)-compatible video-data processing at very high speeds for applications in low-cost real-time document storage and retrieval systems.

Using patented hardware-based compression/expansion techniques, the VCEP features throughput averaging 60 to 80 Mbit/s, i.e., over six pages per second, for CCITT standard documents. It has a dual-bus interface for source and destination memories; however, the VCEP can easily be used in low-cost single-bus systems. A simple register-based interface, controlled by any microprocessor, is used to program the device. A set of six registers contain parameters such as page width, page length, and the required coding algorithm. Normally, these registers are written only once during initialization; the device is controlled during normal operation by using the command/status register.

The hardware interface for the VCEP is straightforward. The device registers are accessed using standard chip-select and Read/Write control signals. Two on-chip input(source) and output(destination) FIFOs, l6 locations deep, provide for burst-mode data transfer from external memory. When the VCEP is enabled, I/O data-request signals indicate which data FIFO needs service and external control logic activates the appropriate data strobe. The FIFOs can also be accessed, using the register interface, in a low-end system to avoid additional control logic.

DEMONSTRATION BOARD DESCRIPTION

The VCEP demonstration board, Figure 2, plugs into a PC-AT* slot and drives an NEC multisync video monitor. The device operates in a dual-bus configuration in this application; compressed-image data is fed onto the board from the PC-AT bus and the expanded-image data is retrieved via the image-data bus. Since the VCEP is very fast, it requires no frame buffer to hold the expanded image; it can drive the video monitor directly. It performs image expansion in real time, 60 times a second, to display a flicker-free image. Alternatively, the external logic can provide a different coded image for each frame and an animation sequence can be displayed.

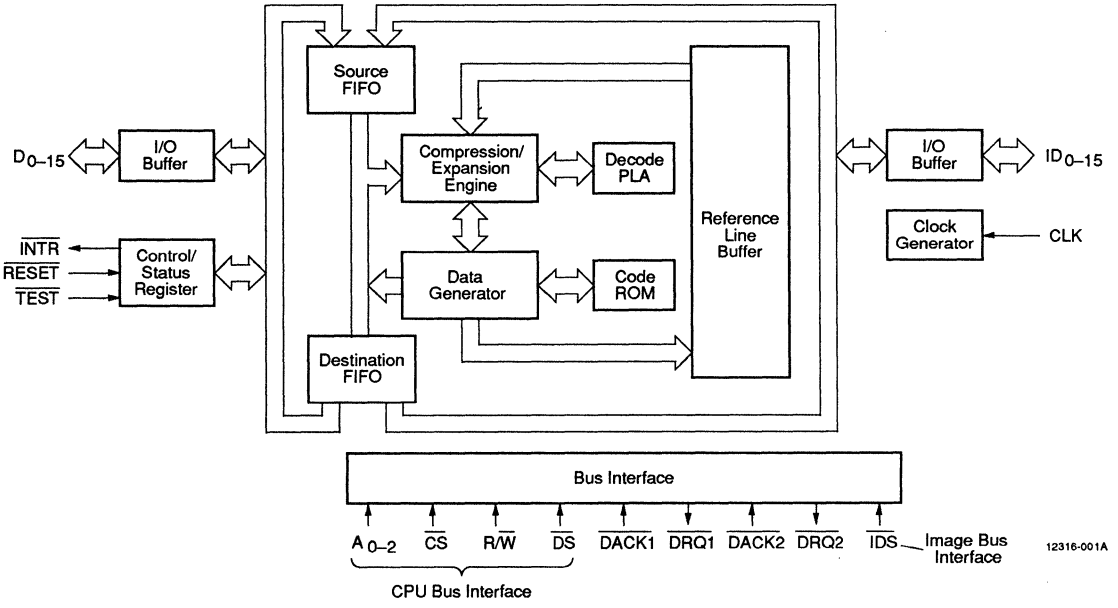


Figure 1. VCEP Block Diagram

* PC-AT is a registered trademark of IBM Incoporated.

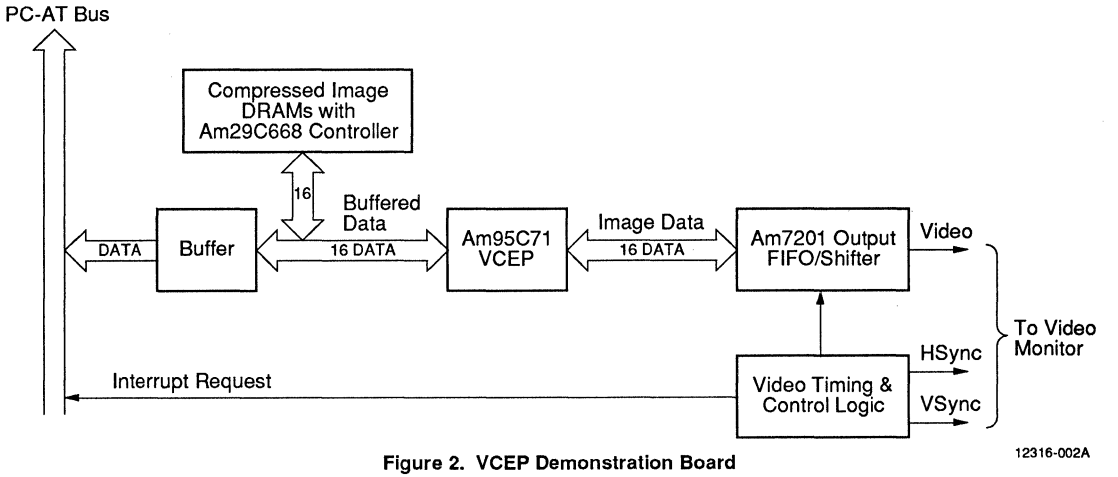


Figure 2. VCEP Demonstration Board

Compressed Image Memory

This board requires a dedicated memory buffer to hold the compressed images because the bandwidth of the PC-AT bus is not fast enough to supply the image data at the required rate. However, in other systems, the main memory could be used to hold the coded-image data.

The compressed-image memory is designed using DRAMs; the array is controlled by an Am29C668 Configurable Dynamic Memory Controller (CDMC). Three types of memory cycles are performed: DRAM refresh when the PC-AT $\overline{\text{REFRESH}}$ signal is active, memory Write when the PC-AT $\overline{\text{DACK5}}$ and $\overline{\text{IOW}}$ signals are active, and memory Read when the $\overline{\text{VDRQ}}$ signal from the VCEP is active and VCEP operation is enabled, i.e. the $\overline{\text{AT}}$ control signal from the board-control register is inactive.

The Am29C668 CDMC contains a refresh counter, autotiming logic, and a column-address counter for burst-mode accesses. In this application, compressed image data is always accessed sequentially and the column counter is used to generate sequential memory addresses.

The compressed-image memory is accessed starting at any page boundary, either by the PC-AT or the VCEP. The starting page address is loaded into the page-address register/counter (AmPAL22V10) by the CPU. This operation also loads the starting address into the memory-controller. The row-address latch is also loaded from the page-address register/counter, and the column-address latch is loaded with zeroes. Subsequent memory accesses increment the CDMC internal column counter, using the $\overline{\text{CAS0}}$ signal fed into the Am29C668 via the column-counter control signal RL/CC. When a page boundary is reached, the End Burst Mode EBM signal becomes active. This causes the external page-address register/counter to be incremented, and the starting address of the next page is loaded into the Am29C668 address latch.

Figure 3 shows the details of the PC-AT interface. The memory-control PAL® device is used to generate the RAS-Input RASI and Mode-Control signals $MC_{1\text{-}0}$ for the CDMC. The $MC_{1\text{-}0}$ specify one of four modes of operation of the Am29C668. Depending on the specific signals, the CDMC generates timing for refresh or Read/Write cycles, and outputs the refresh row address or the multiplexed memory-location address. Refresh and Write cycles are initiated by the PC-AT; hence they are guaranteed to be mutually exclusive. However, when the VCEP is enabled, Read and Refresh requests can become active simultaneously; the memory-control PAL device arbitrates this situation.

Video/Image Data Control

The video control logic consists of a horizontal counter, a vertical counter and a 16-bit video-data shift register. The values from these counters are decoded to generate the horizontal and vertical synchronization control signals and the load control signal for the shift register. The vertical control signal is also used to generate an interrupt to the CPU (Figure 4).

Display resolution is 768 pixels per line and 496 lines per frame. The horizontal-dot clock rate is 32 MHz. After accounting for the blanking times, the effective VCEP data throughput is 22.85 Mbit/s. A 512-deep FIFO, comprising two Am7201s, is connected between the VCEP image-data bus and video shift register to ensure that the image data is available even when the compression ratio in some local area of the document is not too good.

The image-data control PAL device transfers a word from the VCEP to the external FIFO when the Image Data Request $\overline{IDRQ}$ signal becomes active and FIFO-Full $\overline{FF}$ signal is inactive. This expanded image is then loaded into the video shift register and shifted out.

CONCLUSION

This demonstration board illustrates the high-speed operation of the VCEP and the Am29C668 CDMC. However, as mentioned before, the VCEP is capable of much higher throughput: 70 to 80 Mbit/s vs 23 Mbit/s required by the multisync monitor. Throughput rates vary depending on image complexity and coding choice. Running at the maximum clock rate of 20 MHz, the Am95C71 typically handles MMR coding at 65–75 Mbit/s, MR coding (k = 4) at 70–80 Mbit/s, and MH coding at 75–85 Mbit/s.

The VCEP can be designed to perform high-speed image compression/expansion in document-storage and retrieval systems. Using this device, retrieval systems can store large amounts of compressed data that can be quickly retrieved in real time.

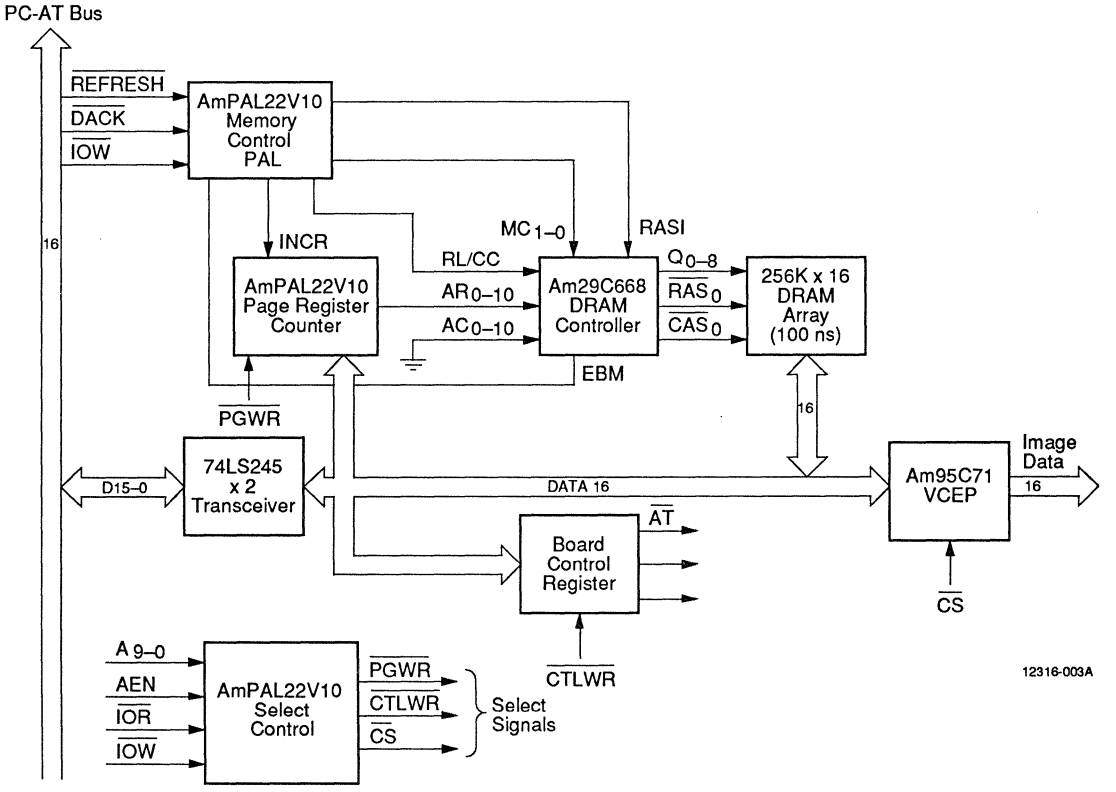


Figure 3. PC-AT Interface

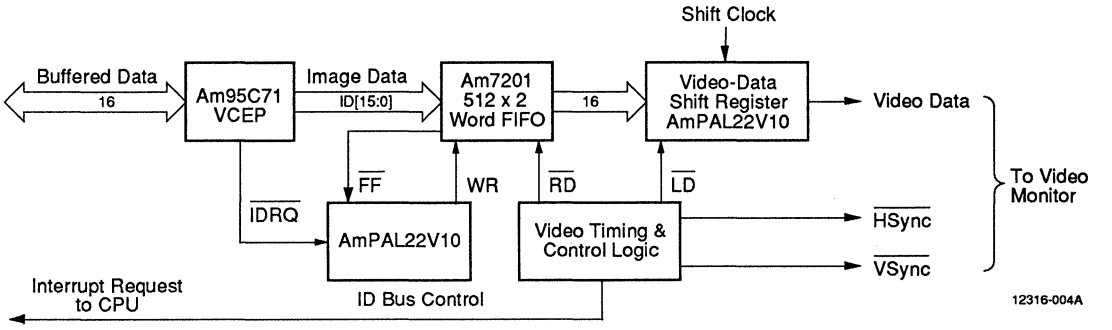


Figure 4. Video Interface Detail

CHAPTER 6
Product Data Sheets

Am2925A	60-MHz Microcycle-Length Programmable Clock Generator	6-3
Am2960/-1/A	16-Bit Cascadable EDC Circuit	6-24
Am2964B	64K Dynamic Memory Controller	6-66
Am2965/66	8-Bit DRAM Driver	6-80
Am2968A	256K Dynamic Memory Controller/Driver	6-88
Am2971A	100-MHz Enhanced Programmable Event Generator (PEG™)	6-112
Am2976	11-Bit DRAM Driver	6-141
Am29368	1M Dynamic Memory Controller/Driver	6-150
Am29C60/-1/A	16-Bit Cascadable EDC Circuit	6-172
Am29C660/A-E	32-Bit Cascadable EDC Circuit	6-189
Am29C668/-1	4M Configurable Dynamic Memory Controller/Driver	6-222
Am29C827A/8/A	10-Bit CMOS Bus Buffer	6-266
Am29C983/A	9-Bit x 4-Port Multiple Bus Exchange	6-275
Am29C985	9-Bit x 4-Port Multiple Bus Exchange with Parity Checker/Generator	6-291
673104A	1M Dynamic RAM Controller/Driver	6-305
SN74S409/-2	256K Dynamic RAM Controller/Driver	6-327

Final

Am2925A

Clock Generator and Microcycle Length Controller

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- **Crystal controlled oscillator**
  - Stable operation from 1-50 MHz
- **TTL controlled oscillator**
  - Stable operation from 1-60 MHz
- **Four microcode controlled clock outputs**
  Allows clock cycle length control for 15–30% increase in system throughput. Microcode selects one of eight clock patterns from 3 to 10 oscillator cycles in length.
- **System controls for $\overline{\text{Run}}/\overline{\text{Halt}}$ and Single Step**
  Switch debounced inputs provide flexible halt controls
- **Slim 0.3″ 24-pin package**
  LSI complexity in minimum board area

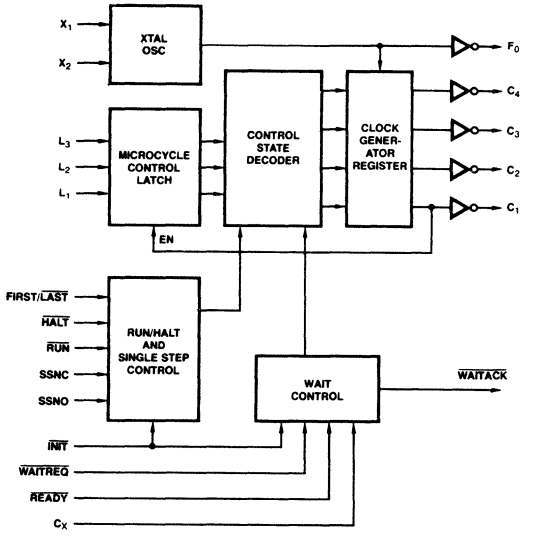
GENERAL DESCRIPTION

The Am2925A is a single-chip general purpose clock generator/driver. It is crystal controlled, and has microprogrammable clock cycle length to provide significant speed-up over fixed clock cycle approaches and meet a variety of system speed requirements. The Am2925A generates four different simultaneous clock output waveforms tailored to meet the needs of Am2900 and other bipolar and MOS microprocessor based systems. One-of-eight cycle lengths may be generated under microprogram control using the Cycle Length inputs L_1, L_2, and L_3.

The Am2925A oscillator runs at frequencies up to 50 MHz crystal input or 60 MHz TTL input. A buffered oscillator output, $F_{0,}$ is provided for external system timing in addition to the four microcode controlled clock outputs C_1, C_2, C_3 and C_4.

System control functions include Run, Halt, Single-Step, Initialize and Ready/Wait controls. In addition, the FIRST/$\overline{\text{LAST}}$ input determines where a halt occurs and the C_X input determines the end point timing of wait cycles. $\overline{\text{WAITACK}}$ indicates that the Am2925A is in a wait state.

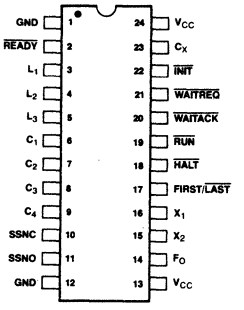
BLOCK DIAGRAM



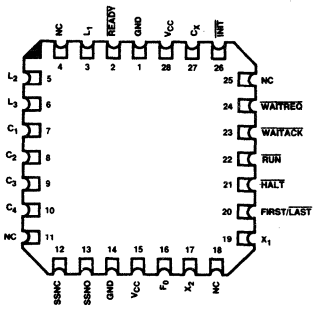
BD002590

Publication# 03367 Rev. C Amendment /0
Issue Date: January 1990

CONNECTION DIAGRAMS
Top View



CD004820



CD004810

Note: Pin 1 is marked for orientation.

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

A. Device Number
B. Speed Option (if applicable)
C. Package Type
D. Temperature Range
E. Optional Processing

AM2925 A D C —

E. OPTIONAL PROCESSING
Blank = Standard Processing

D. TEMPERATURE RANGE
C = Commercial (0 to +70°C)

C. PACKAGE TYPE
D = 24-Pin SLIM Ceramic DIP (CD3024)
L = 28-Pin Ceramic Leadless Chip Carrier (CLT028)
X = Dice

B. SPEED OPTION
A = 60 MHz

A. DEVICE NUMBER/DESCRIPTION
Am2925A
Clock Generator and Microcycle Length Controller

Valid Combinations	
AM2925A	DC, LC, XC

Valid Combinations

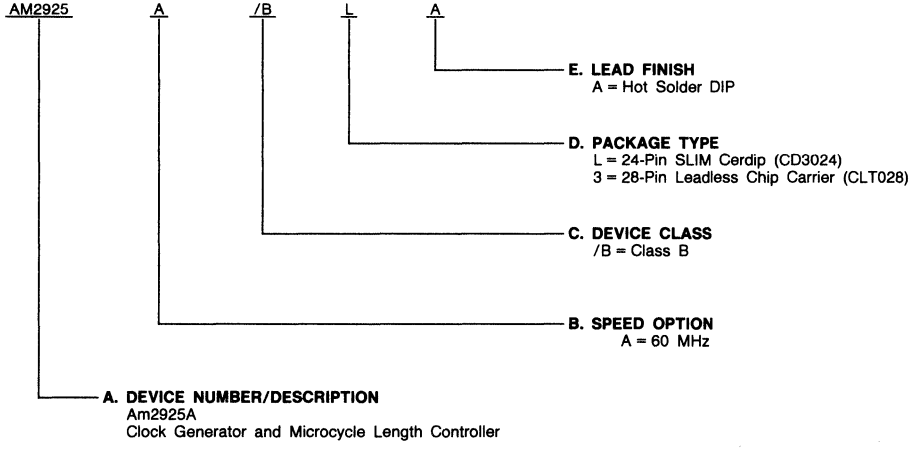
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. CPL (Controlled Products List) products are processed in accordance with MIL-STD-883C, but are inherently non-compliant because of package, solderability, or surface treatment exceptions to those specifications. The order number (Valid Combination) for APL products is formed by a combination of: **A. Device Number**

B. Speed Option (if applicable)
C. Device Class
D. Package Type
E. Lead Finish

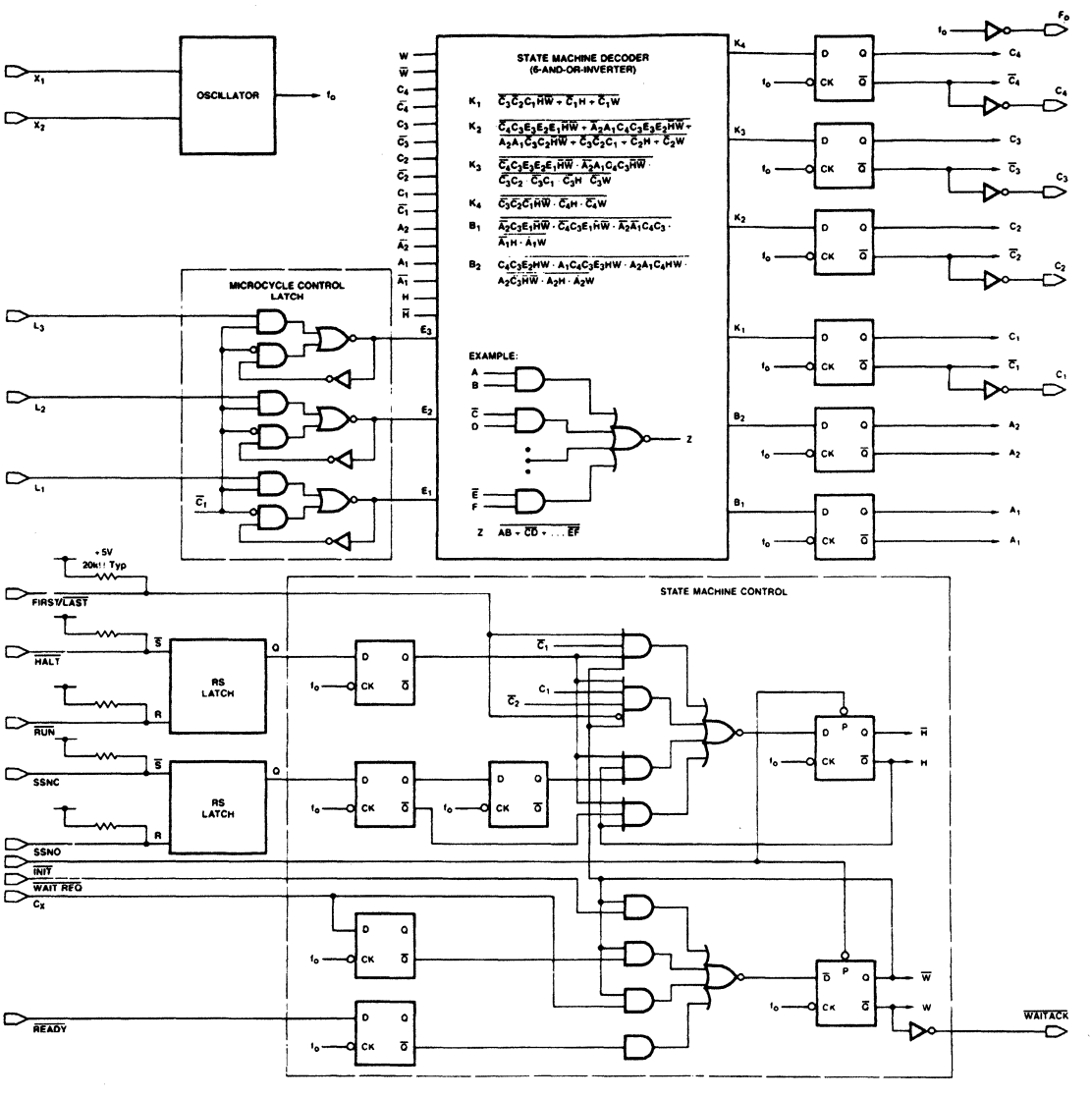


Valid Combinations	
AM2925A	/BLA, /B3A

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Am2925A LOGIC DIAGRAM



BD002542

PIN DESCRIPTION

$C_1 - C_4$ System Clock Outputs (Outputs)
These outputs are all active during every system clock cycle. Their timing is determined by clock cycle length controls, L_1, L_2, and L_3.

$L_1 - L_3$ Clock Cycle Length Control Inputs (Inputs)
These inputs receive the microcode bits that select the microcycle lengths. They form a control word which selects one of the eight microcycle waveform patterns F_3 through F_{10}.

F_0 Buffered Oscillator Output (Output)
F_0 internally generates all of the timing edges for outputs C_1, C_2, C_3, C_4 and $\overline{\text{WAITACK}}$. F_0 rises just prior to all of the C_1, C_2, C_3, C_4 transitions.

$\overline{\text{HALT}}$, $\overline{\text{RUN}}$ Debounce Inputs (Inputs)
These inputs determine whether the output clocks run or not. A LOW input on $\overline{\text{HALT}}$ ($\overline{\text{RUN}}$ = HIGH) will stop all clock outputs.

FIRST/$\overline{\text{LAST}}$ $\overline{\text{HALT}}$ Time Control Input (Input)
A HIGH input in conjunction with a $\overline{\text{HALT}}$ command a will cause halt to occur when C_4 = LOW and $C_1 = C_2 = C_3$ = HIGH (see Clock Waveforms). A LOW input causes a $\overline{\text{HALT}}$ to occur when $C_1 = C_2 = C_3$ = LOW and C_4 = HIGH.

SSNO, SSNC Single-Step Control Inputs (Inputs)
These debounced inputs allow system clock cycle single stepping while $\overline{\text{HALT}}$ is activated LOW.

$\overline{\text{WAITREQ}}$ $\overline{\text{WAIT REQUEST}}$ (Input)
When LOW this input will cause the outputs to halt during the next oscillator cycle after the C_X input goes LOW.

C_X Wait Cycle Control Input (Input)
The clock outputs respond to a wait request one oscillator clock cycle after C_X goes LOW. C_X is normally tied to any one of C_1, C_2, C_3 or C_4.

$\overline{\text{WAITACK}}$ $\overline{\text{WAIT ACKNOWLEDGE}}$ (Output)
When LOW, this output indicates that all clock outputs are in the "WAIT" state.

$\overline{\text{READY}}$ $\overline{\text{READY}}$ (Input)
The $\overline{\text{READY}}$ active LOW input is used to continue normal clock output patterns after a wait stage.

$\overline{\text{INIT}}$ $\overline{\text{INITIALIZE}}$ (Input)
This input is intended for use during power-up initialization of the system. When LOW all clock outputs free run regardless of the state of the Halt, Single Step, Wait Request and Ready inputs.

X_1, X_2 External Crystal Connections (Input/Output)
X_1 can also be driven by a TTL frequency source.

FUNCTIONAL DESCRIPTION

The Am2925A is a dynamically programmable general-purpose clock generator/driver. It can be logically separated into three parts. There is an oscillator, a state machine decoder and a state machine control section.

The oscillator is a linear inverting amplifier which may be configured with a minimum of external parts as a 1st harmonic* crystal oscillator, 3rd harmonic* crystal oscillator, L-C oscillator or used to buffer an external clock. The buffered, inverted output of this oscillator is available as F_0.

The state machine takes microcode information from the Microcycle Length "L" inputs L_1, L_2 and L_3 and counts the fundamental frequency of the internal oscillator, F_0, to create the clock outputs, C_1, C_2, C_3 and C_4.

The clock outputs have a characteristic wave shape relationship for each microcycle length. For example, C_1 is always LOW only on the last F_0 clock period of a microcycle, and C_4 is always LOW on the first. C_3 has an approximately 50% duty cycle, and C_2 is HIGH for all but the last two periods.

The current state of the machine is contained in a register, part of which is the Clock Generator Register. C_1, C_2, C_3 and C_4 are the outputs of this register. These outputs and the outputs of the Microcycle Control Latch are fed into a set of combinatorial logic to generate the next state. On each falling edge of the internal clock, the next state is entered into the current state register. The Microcycle Control Latch is latched when C_1 is HIGH. This means that it will be loaded during the last state of each microcycle ($C_1 = C_2 = C_3$ = LOW, C_4 = HIGH). This internal latch selects one of eight possible microcycle lengths, F_3 to F_{10}.

The state machine control logic, which determines the mode of operation of the state machine, is intended to be connected to a front panel. There are four basic modes of operation of the Am2925A comprised of Run, Halt, Wait and Single Step.

System Timing

In the typical computer, the time required to execute different instructions varies. However, the time allotted to each instruction is the time that it takes to execute the longest instruction. The Am2925A allows the user to dynamically vary the time allotted for each instruction, thereby allowing the user to realize a higher throughput.

This section will cover several aspects of the Am2925A. The first topic to be covered is the oscillator section which is responsible for providing the basis of all system timing. Second will be how to operate the Am2925A; last will be an example of an Am2925A in a 16-bit microprogrammed machine.

*It is understood that the terms "fundamental mode" and "3rd overtone" are generally regarded as more technically correct, but "1st harmonic" and "3rd harmonic" are used here because of their more generally accepted usage.

Oscillator

The Am2925A contains an inverting, linear amplifier which is intended to form the basis of a crystal oscillator. In designing this oscillator it is necessary to consider several factors related to the application.

The first consideration is the desired frequency accuracy. This may be subdivided into several areas. An oscillator is considered stable if it is insensitive to variations in temperature and supply voltage, and if it is unaffected by individual component changes and aging. The design of the Am2925A is such that the degree to which these goals are met is determined primarily by the choice of external components. Various types of crystals are available and the manufacturers' literature should be consulted to determine the appropriate type. For good temperature stability, zero temperature coefficient capacitors should be used (Type NPO). For extreme temperature stability, an oven must be used or some other form of temperature compensation applied.

Absolute frequency accuracy must also be considered. The resonant frequency varies with load capacitance. It is therefore important to match the load specified by the crystal manufacturer for a standard crystal (usually 32 pF), or to specify the load when ordering a special crystal. It should then be possible to determine from the crystal characteristics the load tolerance to maintain a given accuracy. If the "set-on" error due to load tolerance is unacceptable, a trimmer capacitor should be incorporated for fine adjustment.

The mechanism by which a crystal resonates is electromechanical. This resonance occurs at a fundamental frequency (1st harmonic) and at all odd harmonics of this frequency (even harmonic resonance is not mechanically possible). Unless otherwise constrained crystal oscillators operate at their fundamental frequency. However, crystals are not generally available with fundamental frequencies above 20-25 MHz. At higher frequencies, an overtone oscillator must be used. In this case, the crystal is designed to oscillate efficiently at one of its odd harmonic frequencies and additional components are included in the oscillator circuit to prevent it oscillating at lower harmonics.

Where a high degree of accuracy or stability is not required, the amplifier may be configured as an L-C oscillator. It may also be driven from an external clock source if operation is required in synchronous with that source.

1st Harmonic (Fundamental) Oscillator

The circuit of a typical 1st harmonic oscillator is shown in Figure 1. The crystal load is comprised of the two 68 pF capacitors in series. This 34 pF approximates the standard 32 pF crystal load. If a closer match is required then one of the capacitors should be replaced with a parallel combination of a fixed capacitor and a trimmer. The nominal value of the combination should be 60 pF to provide proper crystal loading.

A typical crystal specification for use in this circuit is:

Frequency Range: 5–20 MHz
Resonance: Parallel Mode
Load: 32 pF
Stability: .01% or to match systems requirements
Case: H-17 — for smaller size
Temp Range: −30 to +70°C
Note: Frequency will change over temperature.

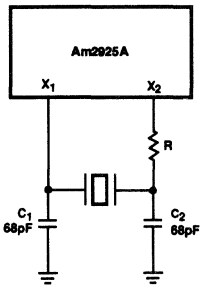


Figure 1. Connections for 5–20 MHz

It is good practice to ground the case of the crystal to eliminate stray pick-up and keep all connections as short as possible.

Note: At fundamental frequencies below 5 MHz it is possible for the oscillator to operate at the 3rd harmonic. To prevent this a resistor should be added in series with the X_2 pin as shown in the circuit diagram.

The resistor value should match the impedance of C_2:

$$R = X_{C_2} = \frac{1}{2\pi f C_2}$$

3rd Harmonic Oscillator

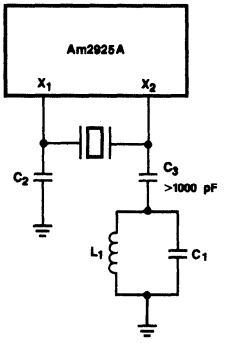
At frequencies greater than 20 MHz, the crystal can be operated at its 3rd harmonic. A typical circuit is shown in Figure 2. Two additional components are included; an inductor, L_1, and a capacitor, C_3. The purpose of the capacitor is to block the d.c. path through the inductor and thereby maintain the correct amplifier bias. C_3 should be large ($\geqslant$1000 pF).

The inductor forms a parallel tuned circuit with C_1. This circuit has its resonance set between the 1st and 3rd harmonics of the crystal and is used to prevent the oscillator operating at the 1st harmonic. In the 1st harmonic oscillator (Figure 1), the crystal appears as an inductor and forms a π-network with the two capacitors, thus providing the necessary phase shift for oscillation. In the 3rd harmonic oscillator, L_1 and C_1 are chosen such that at the 3rd harmonic the impedance of circuit is equivalent to that of the capacitor C_2 in the 1st harmonic oscillator (Figure 3b). Thus, the same π-network is formed (Figure 3c) and oscillation is possible. At the 1st harmonic the tuned circuit appears as an inductor (Figure 3a), the π-network is not formed and oscillation is not possible.

The following specification is typical for a crystal to be used in a 3rd harmonic oscillator.

Frequency Range: Above 20 MHz
Resonance: Parallel Mode
Load: 32 pF
Stability: .01% or to match systems requirements
Case: H-17 — for smaller size
Temp Range: −30 to +70°C
Note: Frequency will change over temperature.

Again it is good practice to ground the crystal case and keep connections short.

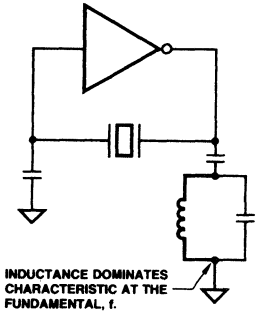


DF000852

C_1	C_2	L	f_0 (Max.)
68	82	$\frac{1150}{F_0{}^2}$	20-27
33	45	$\frac{2079}{F_0{}^2}$	27-48
18	25	$\frac{3800}{F_0{}^2}$	48-54

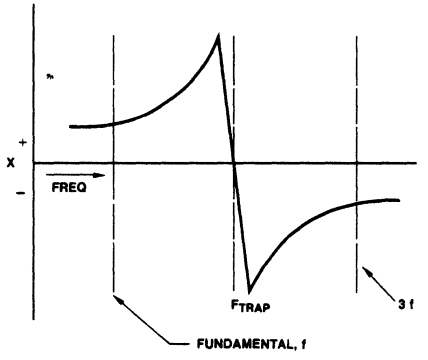
Units:
C_1, C_2 = pF; L = μH; f_0 = MHz

Figure 2. Connections for Frequencies above 20 MHz



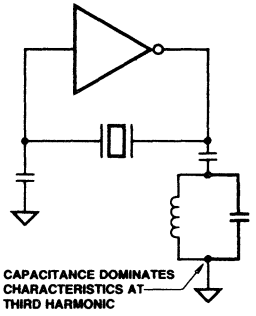
DF000820

a) Fundamental Equivalent



DF000840

b) Trap Impedance



DF000830

c) 3rd Harmonic Equivalent

Figure 3. Forcing Third Harmonic Oscillation

Design Procedure

(1) Assume C_1 = 82 pF and C_2 = 68 pF (this gives a sensible inductor value). L_1 is calculated according to the formula:

$$L_1 = \frac{1151}{f_o^2}$$

f_o = Operating frefquency in MHz
L_1 in μH

This sets the resonant frequency of the L-C combination at 0.52 f_o.

(2) Select the closest standard value inductor for L_1. Using this value calculate C_1 such that the resulting crystal load at the 3rd harmonic is 32 pF.

$$C_1 = 60 + \frac{25330}{L_1 \cdot f_o^2} \qquad C_1 \text{ in pF.}$$

Choose the closest standard capacitor value to this.

Using standard values both the resonant frequency of the L-C circuit (f_r) and the crystal load are non-optimal. This will cause a slight error in the oscillating frequency. If this is not permissible, C_1 may be a fixed capacitor in parallel with a trimmer such that the range of adjustment includes the calculated value for C_1. This is then set to give the desired frequency. In either case the approximate inductor value will cause the resonant frequency to the L-C circuit to change. This frequency, f_r, may be computed and should remain approximately midway between the 1st and 3rd harmonic.

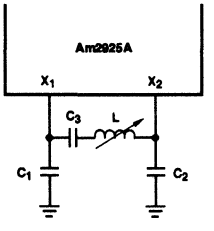
$$f_r = \frac{159}{\sqrt{L_1 C_1}}$$

f_r in MHZ
L_1 in μH
C_1 in pF

L-C Oscillator

The Am2925A can be operated as an L-C tuned oscillator (Figure 4) and will perform as a stable oscillator within the restrictions of the chosen frequency determining components (i.e., inductor and capacitors). The circuit chosen is a classical π-network with DC loop isolation. The Am2925A oscillator is a DC biased linear amplifier. This DC bias is necessary and therefore C_3 is included to block the DC path through the inductor. If a variable slug tuned inductor is used, a moderate range of frequency adjustment tuneability (approximately 2:1) can be achieved. The range can be enhanced by switching the two resonant capacitors (C_1, C_2) to larger or smaller values. The specific frequency of operation can be determined by the formula:

$$f = \frac{1}{2\pi\sqrt{LC}}$$ (where C is C_1 and C_2 in series).

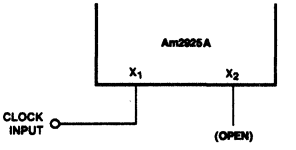


DF000872

Figure 4. L-C Tuned Oscillator

External Clock Drive

The Am2925A can be driven by an external TTL clock signal. This is accomplished by directly coupling the TTL clock signal into X_1, the oscillator input. The X_2 pin is an output and should be left unloaded in this configuration.



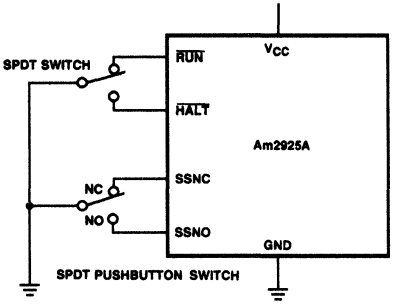
DF000882

Figure 5. External Clock Drive

Am2925A Control Inputs

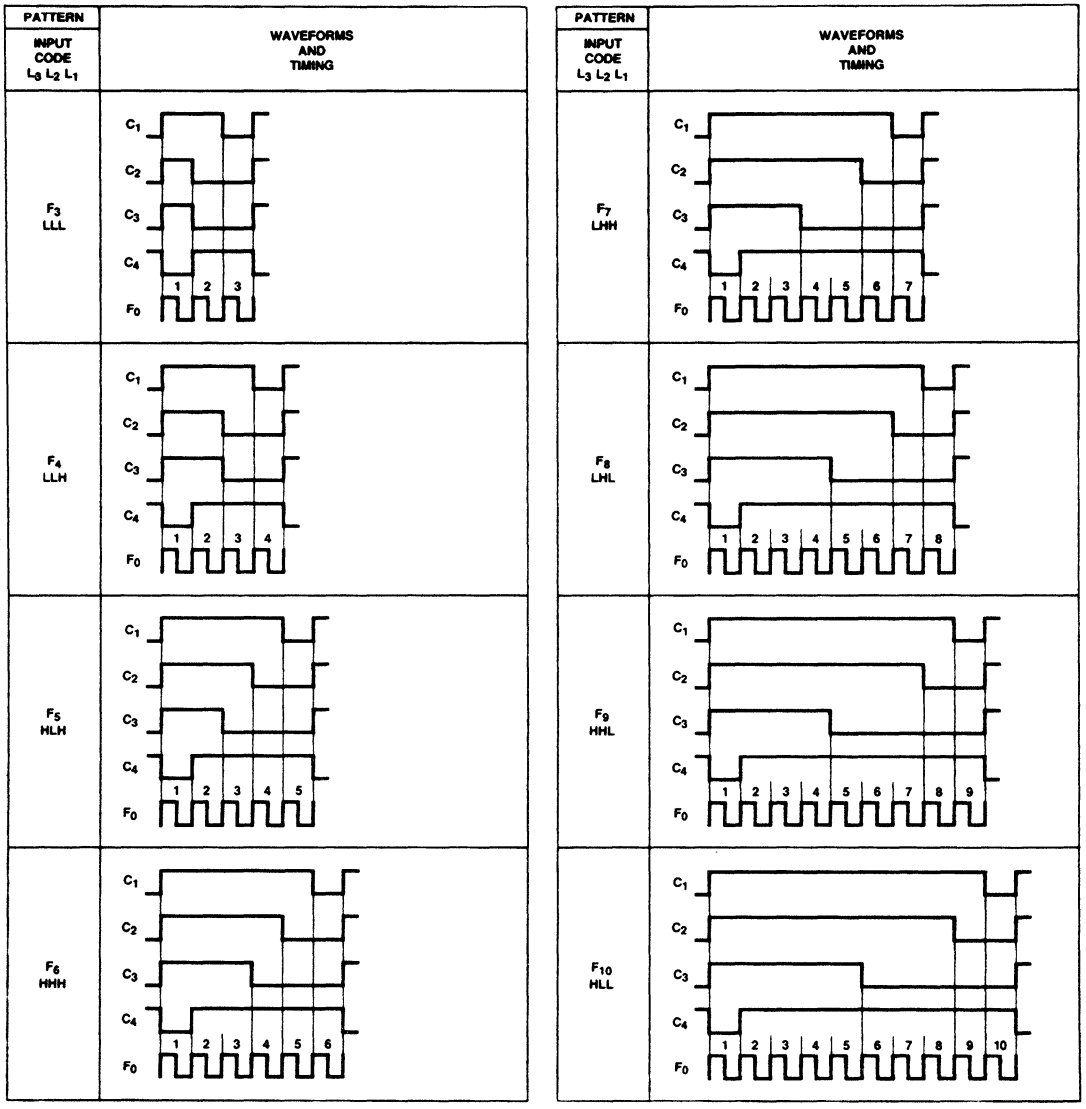
The control inputs fall into two categories, microcycle length control and clock control. Microcycle length control is provided via the "L" inputs which are intended to be connected to the microprogram memory. The "L" inputs are used to select one of eight cycle lengths ranging from three oscillator cycles for pattern F_3 to ten oscillator cycles for pattern F_{10}. This information is always loaded at the end of the microcycle into the Microcycle Control Latch. The Microcycle Latch performs the function of a pipeline register for the microcycle length microcode bits. Therefore, the cycle length goes in the same microword as the instruction that it is associated with.

The clock control inputs are used to synchronize the microprogram machine with the external world and I/O devices. Inputs like $\overline{RUN}$, $\overline{HALT}$, SSNO and SSNC, which start and stop execution, are meant to be connected to switches on the front panel of the microprogrammed machine (see Figure 6). These inputs have internal pull-up resistors and are connected to an R-S flip-flop in order to provide switch debouncing. The FIRST/$\overline{LAST}$ input is used to determine at what point of the microcycle the Am2925A will halt when $\overline{HALT}$ or a SINGLE STEP is initiated. In most applications the user wires this input HIGH or LOW depending on his design.



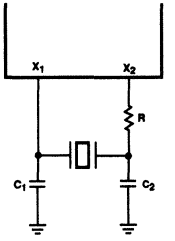
DF000862

Figure 6. Switch Connection for $\overline{RUN}$/$\overline{HALT}$ and Single Step



WF003080

Figure 7. Am2925A Clock Waveforms

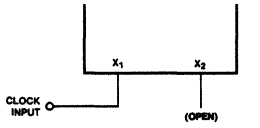


AF001770

$$R = X_{C_2} = \frac{1}{Tf C_2} \text{ for } 1-6 \text{ MHz}$$

$R = 0$ for 6 – 20 MHz

a) Fundamental Oscillator

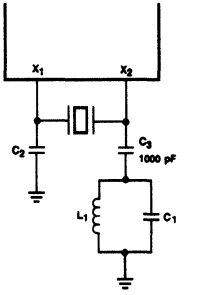


AF001761

b) External Clock Drive

C_1	C_2	L	f_0 (Max.)
68	82	$\frac{1150}{F_0{}^2}$	20-27
33	45	$\frac{2079}{F_0{}^2}$	27-48
18	25	$\frac{3800}{F_0{}^2}$	48-54

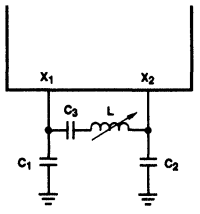
Units:
C_1, C_2 = pF; L = μH; f_0 = MHz



AF001741

($f_0 \geq 20$ MHz)

c) 3rd Harmonic Oscillator



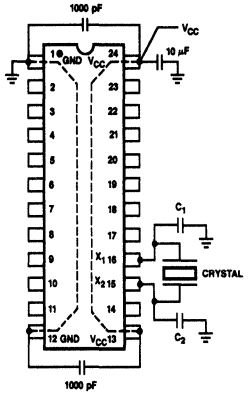
AF001751

$$f_0 = \frac{1}{2\pi\sqrt{2LC}}$$

$C_1 = C_2 = C$

$X_{C_3} < X_L$

d) L-C Oscillator



PF001070

e) Typical External Connections

Design Considerations

1. Oscillator external connections should be less than 1″ long — wirewrap is not recommended.
2. V_{CC} and GND connections should be less than ½″ long to power plane.
3. Supply decoupling includes both high frequency and bulk storage elements.
4. The same considerations apply for 3rd overtone configurations.

Figure 8. Am2925A Oscillator Applications

(External Component Calculations Summary)

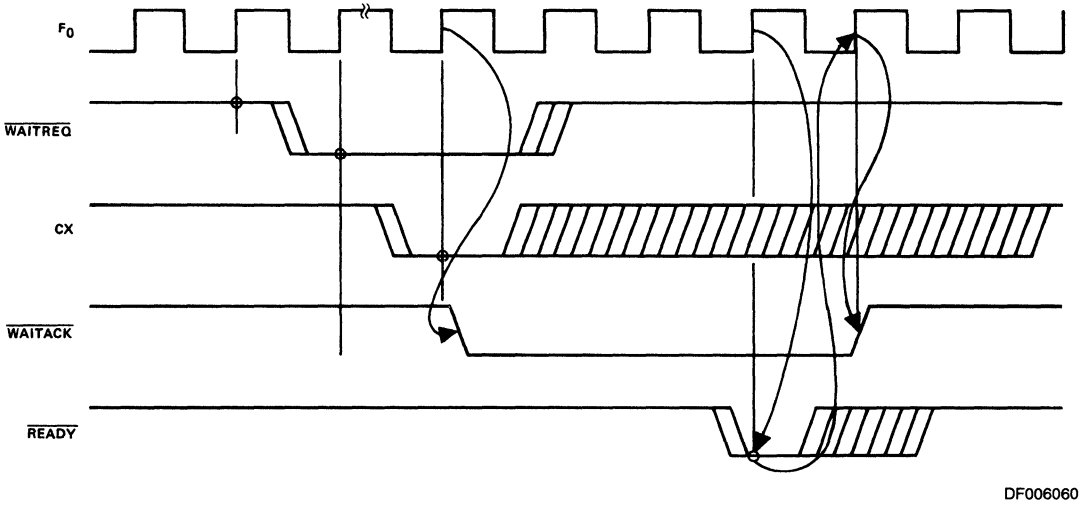


Figure 9. Am2925A WAIT/READY Timing

When HALT is held LOW (RUN = HIGH), the state machine will start the halt mode on the last (C_1 = LOW) or the first (C_4 = LOW) state of the microcycle as determined by the FIRST/LAST input. When RUN goes LOW (HALT = HIGH), the state machine will resume the run mode.

The WAITREQ, C_X, READY and WAITACK signals are used to synchronize other parts of a computer system (memory, I/O devices) to the CPU by dynamically stretching the microcycle. For example, the CPU may access a slow peripheral that requires the data remain on the data bus for several microseconds in which case the peripheral pulls the WAITREQ line LOW. The C_X input lets the designer specify when the WAITREQ line is sampled in the microcycle. This has a direct impact on how much time the peripheral has to respond in order to request a wait cycle (see Figure 9). The READY line is used by the peripheral to signal when it is ready to resume execution of the rest of the microcycle. The WAITACK line goes LOW on the next oscillator cycle after the C_X input goes LOW and remains LOW until the second oscillator cycle after READY goes LOW.

The SSNO and SSNC inputs are used to initiate the SINGLE STEP mode. These debounced inputs allow a single microcycle to occur while in the halt mode. SSNO (normally open) and SSNC (normally closed) are intended to be connected to a momentary SPDT switch. After SSNO has been LOW for one clock edge, the state machine will change to the run mode. The microcycle will end on the first or last state of the microcycle depending on the state of the FIRST/LAST.

AC Timing Signal References

Set-up and hold times in registers and latches are measured relative to the clock signals that drive them. In the Am2925A, the crystal oscillator provides a free running clock signal that drives all the registers on the devices. This clock is provided for the user through the buffered output of F_0. Therefore, F_0 is used as the reference for set-up, hold and clock to output times. However for the Microcontrol Latch, the set-up and hold times are referenced to the C_1 output which is the buffered version of the latch enable. This reference is appropriate for the Microcontrol Latch because in a typical application this latch is considered part of the pipeline register which is also driven by one of the "C" outputs.

Clock Outputs

There are four clock outputs provided for the user which have different duty cycles. The user must make a decision as to which one best fits his purposes. For example, in a three-address architecture, with the Am2903A (Figure 10), the C_3 clock (approximately 50% duty cycle) could be used to drive the clock input while C_2 (always LOW last two oscillator cycles) drives Instruction Enable. This guarantees, for microcycle lengths greater than four, that the internal RAM data latches of the Am2903A are closed and the destination address is multiplexed onto the B address bus before the RAM begins the Write cycle (Figure 11).

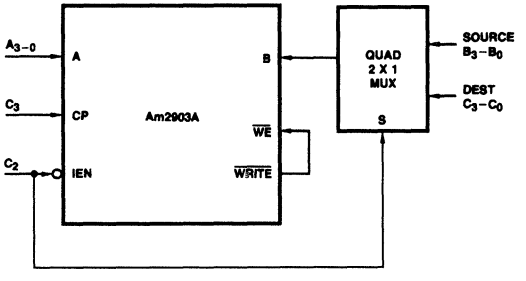


Figure 10. Am2903A Three-Address Architecture

APPLICATIONS

16-Bit Machine with Am2925A

The block diagram in Figure 12 shows a 16-bit microprogrammed machine which uses an Am2925A to generate system timing. The design decisions include oscillator frequency and clock pattern selections.

Selecting the Crystal

In order to pick the oscillator frequency, a detailed timing analysis of the machine must be done in order to determine the execution length of every operation to be performed. For each operation there will be several delay paths, which usually include the ALU and the microprogram control.

Table 1 is an example of two of these paths. PATH 1 is a path through the Am2910A (Figure 12) for a microprogram Conditional Jump Subroutine. PATH 2 is a data flow path through the Am2903A for an Add instruction. Therefore, if the operation were an Add with a Conditional Jump Subroutine the maximum delay would be 170 ns. If there were a Program Control Unit also, then delays through it would have to be considered.

After the execution times all of the instruction types have been calculated, the oscillator frequency can be selected. It is desirable to minimize the difference between the most commonly used instructions and multiples of the oscillator period. In this way the most efficient use can be made of the variable microcycle scheme.

For example, in the hypothetical machine in Figure 12 there are four instruction types (most machines will have more). Table 2 is a table which lists each instruction type, corresponding execution time, and anticipated percentage of the typical instruction stream for each instruction. Several possible frequencies are shown which contain the next highest multiple of the corresponding oscillator period for each instruction. 50 MHz is the best choice because it comes closest to matching instructions A and C which compose 90% of the typical instruction stream.

Fixed Bandwidth Buses

For those designs that require a data bus with fixed bandwidth and fixed time slots for each memory access, the designer should consider using cycle lengths which are a multiple of the shortest cycle length, i.e., cycle lengths 3, 6 and 9 or cycle lengths 4 and 8.

The design could further require that the bus be accessed only during the shortest cycle length. Therefore, by using multiple cycle lengths it can be predicted when the CPU will access the bus and for how long, thereby maintaining the fixed bandwidth.

Performance Comparison

Estimated performance can be calculated directly from Table 2. For a fixed microcycle machine the longest instruction execution time would have to be used for all instructions, yielding an average instruction time of 200 ns. With a variable microcycle machine the average instruction time is the sum of the products for each instruction, of the percentage of the instruction stream and the next highest multiple. The average instruction for the example machine with a 50 MHz crystal is:

$(0.6 \times 140 + .08 \times 180 + .3 + 200 + .02 \times 200) = 162.8$ ns

This represents a 19% increase in system performance without requiring any other system speed-ups and without requiring faster devices.

TABLE 1. DELAY PATH TOTALS FOR AN ADD AND A CONDITION JUMP SUBROUTINE

Device No.	Device Path	Path 1	Path 2
Am27S27	CP – Q	27	27
Am2904	INST – CT	58	–
Am2903A	I/AB – GP	–	50
Am2910A	CC – Y	30	–
Am2902A	GP – CN + Z	–	7
Am27S27	TS	55	–
Am2903A	CN – Z	–	35
Am2904	TSZ	–	17
Total	ns	170	136

TABLE 2. INSTRUCTION TIME ANALYSIS

Instruction Type	A	B	C	D	Units
Execution Time	140	180	184	200	ns
Percentage of Instruction Stream	60%	8%	30%	2%	%
Closest Multiple Oscillator Period					
20 MHz P = 50	150 (3P)	200 (4P)	200 (4P)	200 (4P)	ns
25 MHz P = 40	160 (4P)	200 (5P)	200 (5P)	200 (5P)	ns
30 MHz P = 33	167 (5P)	200 (6P)	200 (6P)	200 (6P)	ns
33 MHz P = 30	150 (5P)	180 (6P)	210 (7P)	210 (7P)	ns
40 MHz P = 25	150 (6P)	200 (8P)	200 (8P)	200 (8P)	ns
50 MHz P = 20	140 (7P)	180 (9P)	200 (10P)	200 (10P)	ns

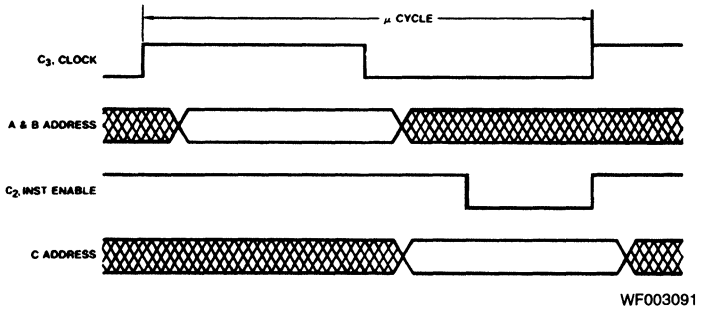


Figure 11. Am2903A Three-Address Operation

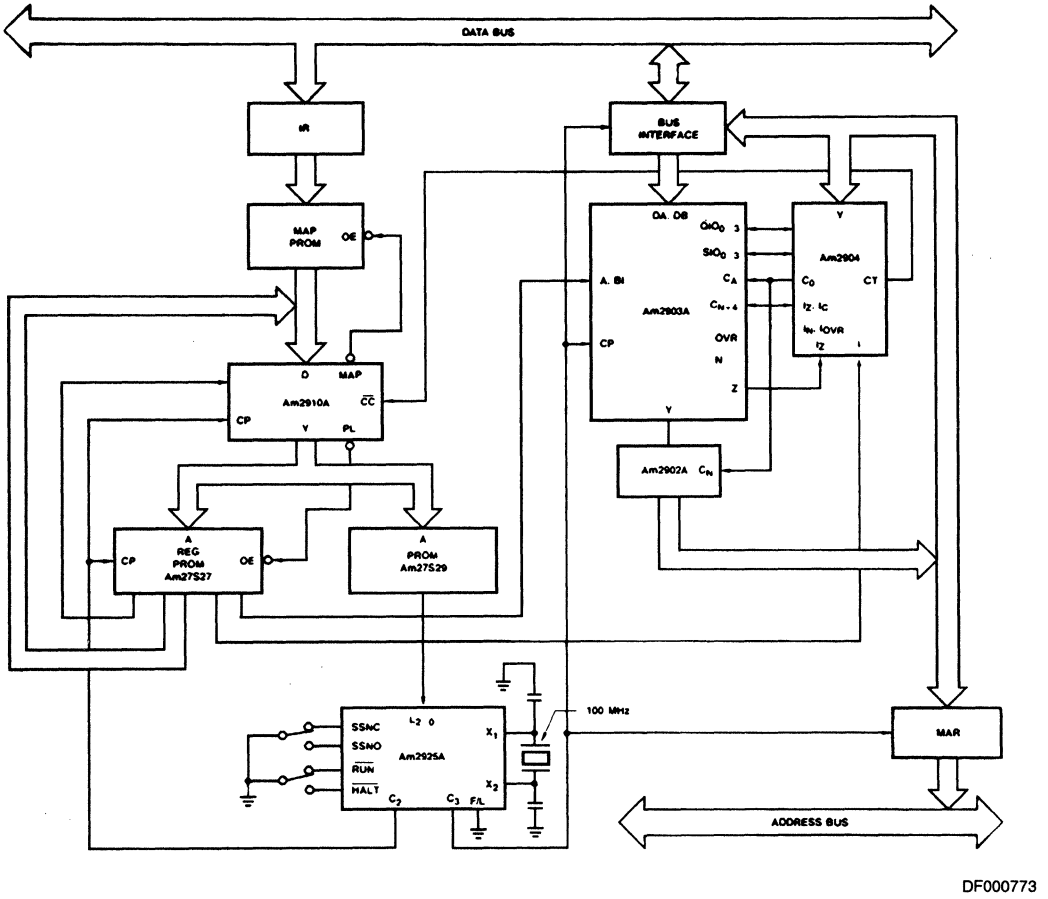


Figure 12. 16-Bit Microprogrammed Machine

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65 to +150°C
Ambient Temperature Under Bias ..... −55 to +125°C
Supply Voltage to Ground Potential
Continuous ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... −0.5 V to +V_{CC} Max.
DC Input Voltage ..... −0.5 to +5.5 V
DC Output Current, Into Outputs ..... 30 mA
DC Input Current ..... −30 to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Temperature (T_A) ..... 0 to +70°C
Supply Voltage (V_{CC}) ..... +4.75 to +5.25 V

Military (M) Devices
Temperature (T_C) ..... −55 to +125°C
Supply Voltage (V_{CC}) ..... +4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges unless otherwise specified

Parameter Symbol	Parameter Description	Test Conditions (Note 1)				Min.	Max.	Units
V_{OH}	Output HIGH Voltage	V_{CC} = Min. V_{IN} = V_{IH} or V_{IL}		I_{OH} = −1.0 mA		2.5		Volts
V_{OL}	Output LOW Voltage	V_{CC} = Min. V_{IN} = V_{IH} or V_{IL}		$\overline{WAITACK}$ C_i	I_{OL} = 4.0 mA		0.4	Volts
					I_{OL} = 8.0 mA		0.45	
					I_{OL} = 12 mA		0.5	
				F_0	I_{OL} = 16 mA		0.5	Volts
V_{IH}	Input HIGH Level (Note 2)	Guaranteed input logical HIGH voltage for all inputs				2.0		Volts
V_{IL}	Input LOW Level (Note 2)	Guaranteed input logical LOW voltage for all inputs			MIL		0.7	Volts
					COM'L		0.8	
V_I	Input Clamp Voltage (Note 2)	V_{CC} = Min., I_{IN} = −18 mA					−1.5	Volts
I_{IL}	Input LOW Current	V_{CC} = Max. V_{IN} = 0.4 V		$\overline{READY}$, $\overline{INIT}$, L_1, L_2, L_3			−0.4	mA
				$\overline{WAITREQ}$, X_1 (See Figure 13)			−0.8	mA
				SSNO, SSNC, $\overline{RUN}$, $\overline{HALT}$			−1.0	mA
				C_X			−1.2	mA
				FIRST/$\overline{LAST}$			−1.5	mA
I_{IH}	Input HIGH Current	V_{CC} = Max. V_{IN} = 2.7 V		$\overline{READY}$, $\overline{INIT}$, L_1, L_2, L_3			20	μA
				$\overline{WAITREQ}$			50	μA
				SSNO, SSNC, $\overline{RUN}$, $\overline{HALT}$			−500	μA
				C_X			70	μA
				FIRST/$\overline{LAST}$			−750	μA
				X_1 (See Figure 13)			500	μA
I_I	Input HIGH Current	V_{CC} = Max.	V_{IN} = 5.5 V	$\overline{READY}$, $\overline{INIT}$, L_1, L_2, L_3			100	μA
			V_{IN} = V_{CC}	SSNO, SSNC, $\overline{RUN}$, $\overline{HALT}$			100	μA
			V_{IN} = 5.5 V	$\overline{WAITREQ}$, C_X			1.0	mA
			V_{IN} = V_{CC}	FIRST/$\overline{LAST}$			1.0	mA
			V_{IN} = 4.0 V	X_1 (See Figure 13)			1.0	mA
I_{SC}	Output Short Circuit Current (Note 3)	V_{CC} = Max.				−30	−85	mA
I_{CC}	Power Supply Current (Note 4)	V_{CC} = Max.					120	mA

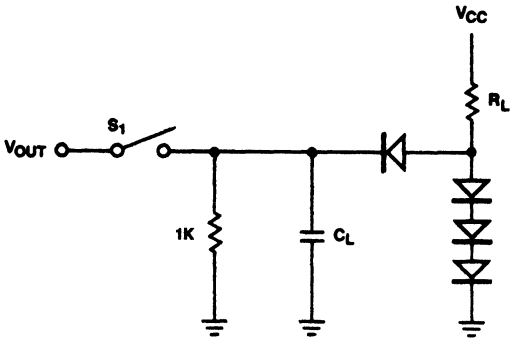
Notes: 1. For conditions shown as Min. or Max. use the appropriate value specified under Operating Ranges for the applicable device type.
2. Does not apply to X_1 and X_2.
3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
4. I_{CC} varies with temperature and oscillation frequency as shown in Figure 14. The parameters specified (worst case) apply to f_0 = 0, +25°C, C_1 = C_2 = C_3 = LOW, C_4 = HIGH, X_1 = 2.4 V, X_2 = OPEN and F_0 = LOW. The variations shown in Figure 14 apply to typical values.

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
———	MUST BE STEADY	WILL BE STEADY
———		
	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
	DON'T CARE; ANY CHANGE PERMITTED	CHANGING; STATE UNKNOWN
	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

KS000010

SWITCHING TEST CIRCUITS

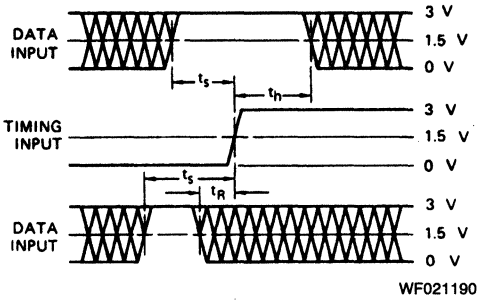


TC003490

Normal Outputs

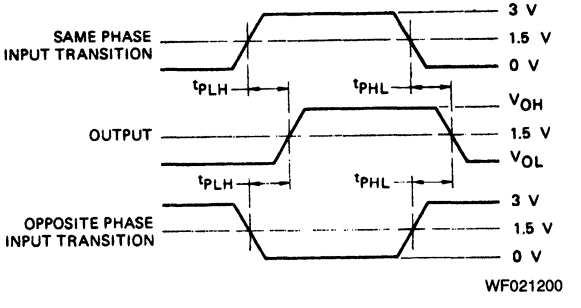
Notes: 1. C_L = 50 pF includes scope probe, wiring, and stray capacitances without device in test fixture.
2. S_1 is open during function test and all AC tests, except output enable tests.
3. C_L = 15 pF for t_{pd} for X_1 to F_0 tests.
4. Programmable loads are used for automatic testing.

SWITCHING TEST WAVEFORMS

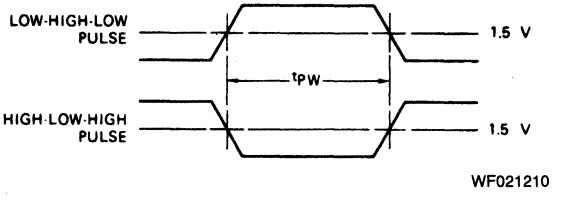


Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched are is don't care condition.

Setup, Hold, and Release Times



Propagation Delay



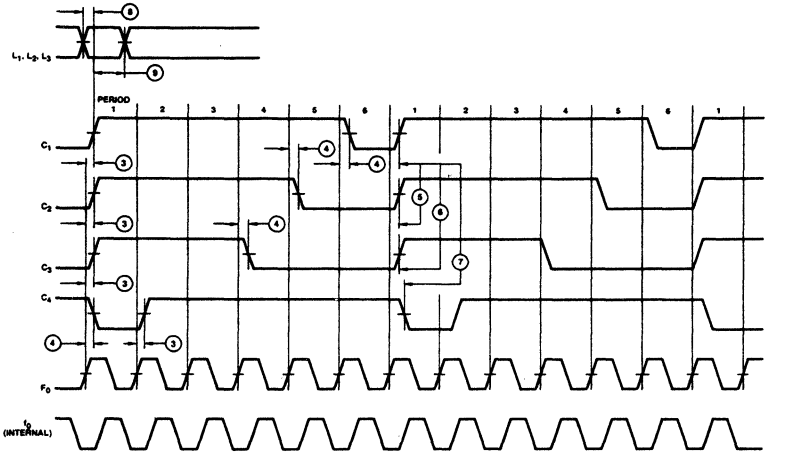
Pulse Width

SWITCHING CHARACTERISTICS over operating ranges unless otherwise specified

No.	Parameter Symbol	Parameter Description		Test Conditions	Am2925A Min.	Am2925A Max.	Units
1	t_{MAX1}	F_0 Frequency (TTL Input) (Notes 1, 7)	C	C_L = 15 pF, R_L = 280 Ω	60		MHz
			M		55		
2	t_{MAX2}	F_0 Frequency (Crystal Input) (Notes 1, 7)	C		50		MHz
			M		45		
3	t_{OFFSET}	F_0(↓) to C_1, C_2, C_3, C_4 or $\overline{WAITACK}$ (↓)	C M	C_L = 50 pF, R_L = 2.0 kΩ		8.5	ns
4	t_{OFFSET}	F_0(↓) to C_1, C_2, C_3, C_4 or $\overline{WAITACK}$ (↑)	C			12	ns
			M				
5	t_{SKEW}	C_1 (↓) to C_2 (↓)	C M			5	ns
6	t_{SKEW}	C_1 (↓) to C_3 (↓)	C M			5	ns
7	t_{SKEW}	C_1 (↓) to C_4 (↑) Opposite Transition	C M			11	ns
8	t_S	L_1, L_2, L_3 to C_1 (↓) (Note 8)	C		15		ns
			M				
9	t_H	L_1, L_2, L_3 to C_1 (↓) (Note 8)	C M		0		n s
10	t_S	C_X to F_0 (↓) (Notes 2, 8)	C M		15		ns
11	t_H	C_X to F_0 (↓) (Notes 2, 8)	C M		0		ns
12	t_S	$\overline{WAITREQ}$ to F_0 (↓) (Notes 3, 8)	C M		15		ns
13	t_H	$\overline{WAITREQ}$ to F_0 (↓) (Notes 3, 8)	C M		0		ns
14	t_S	$\overline{READY}$ to F_0 (↓) (Notes 3, 8)	C M		15		ns
15	t_H	$\overline{READY}$ to F_0 (↓) (Notes 3, 8)	C M		0		ns
16	t_S	$\overline{RUN}$, $\overline{HALT}$ (↑) to F_0 (↓) (Notes 3, 4, 8)	C M		15		ns
17	t_H	SSNC, SSNO to F_0 (↓) (Notes 3, 4, 8)	C M		15		ns
18	t_S	First/$\overline{LAST}$ to F_0 (↓) (Notes 5, 8)	C		20		ns
			M				
19	t_S	$\overline{INIT}$ (↑) to F_0 (↓) (Notes 3, 8)	C		20		ns
			M				
20	t_{PWL}	$\overline{INIT}$ LOW Pulse Width	C		12		ns
			M				
21	t_{PLH}	$\overline{INIT}$ to $\overline{WAITACK}$	C			17	ns
			M				
22	t_{PLH}	Propagation Delay (Notes 6, 9) X_1 to F_0	C	C_L = 15 pF, R_L = 280 Ω		16	ns
			M				
23	t_{PHL}		C			13	ns
			M				

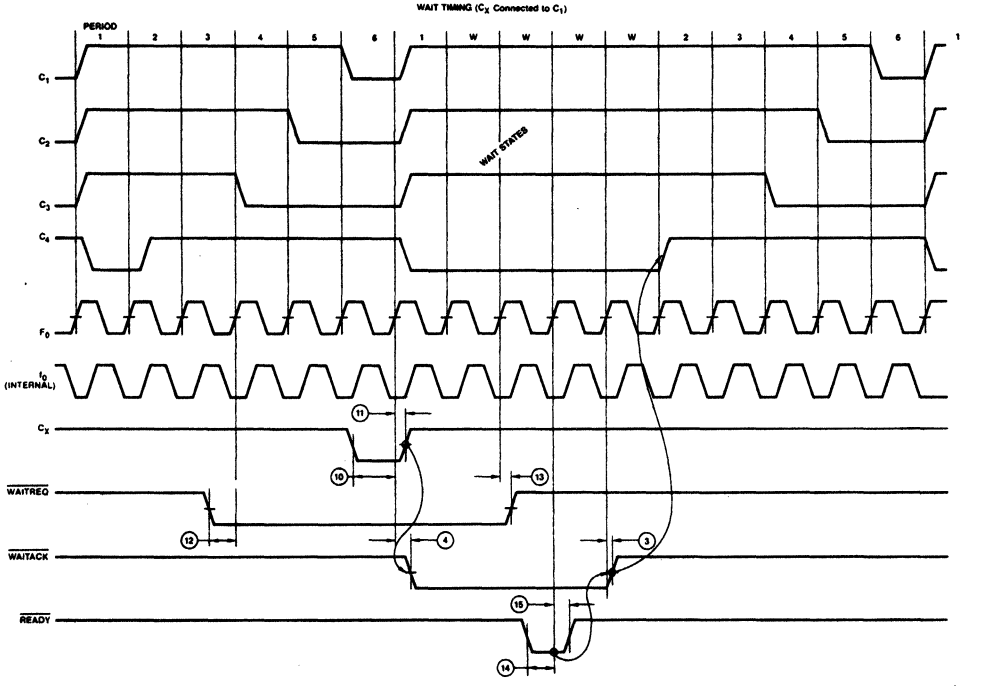
Notes:
1. The frequency guarantees apply with C_X connected to C_1, C_2, C_3, C_4 or HIGH. The C_X input load must be considered part of the 50 pF/2.0 kΩ clock output loading.
2. These set-up and hold times apply to the F_0 LOW-to-HIGH transition of the period in which C_X goes LOW.
3. These inputs are synchronized internally. Failure to meet t_S may cause a $1/F_0$ delay but will not cause incorrect operation.
4. These inputs are "debounced" by an internal R-S flip-flop and are intended to be connected to manual break-before-make switches.
5. FIRST/LAST normally wired HIGH or LOW.
6. Reference point of T offset has been moved forward which has increased T offsets.
7. f_{MAX} not tested.
8. Setup and Hold not tested.
9. Tested at 50 pF system load correlated to 15 pF.

SWITCHING WAVEFORMS



WF003070

Normal Cycle Without Wait States (Pattern F_6 Shown)



WF003061

Wait Timing (C_X Connected to C_1)

Am2925A General Test Notes

Automatic tester hardware and handler hardware add additional round trip A.C. delay to test measurements. Actual propagation delay testing may incorporate a correlation factor to negate the additional delay.

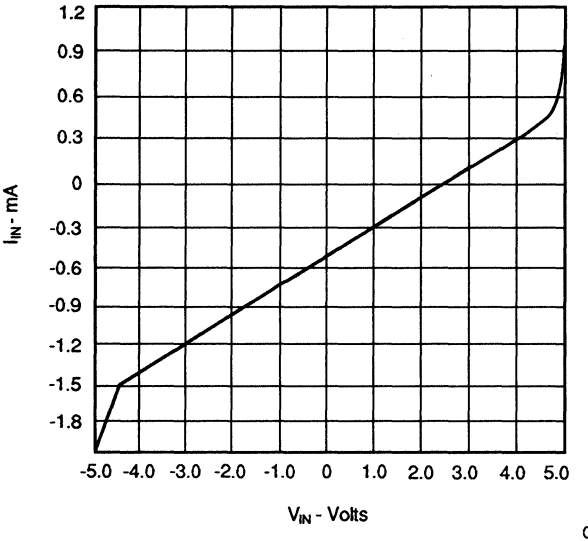
Function testing is done with input LOW less than V_{IL}, and input HIGH greater than V_{IH}. Single trip point at the approximate threshold voltage is used to determine output logic level.

Setup and Hold tests are not performed due to tester accuracy limitation. They are guaranteed by correlation.

T_{PLH} and T_{PHL}, X_1 to F_0, are tested with C_L = 50 pF and are extended limits, rather than 15 pF as specified in the Test Conditions column of the A.C. table. 50 pF is used because it is the minimum test system load. This parameter is guaranteed by correlation to worst-case A.C. parameter.

Programmable loads are used for automatic testing. A.C. loads specified in the datasheet are used for bench testing. Programmable loads, which simulate datasheets loads, are used during production testing.

TYPICAL PERFORMANCE CURVES



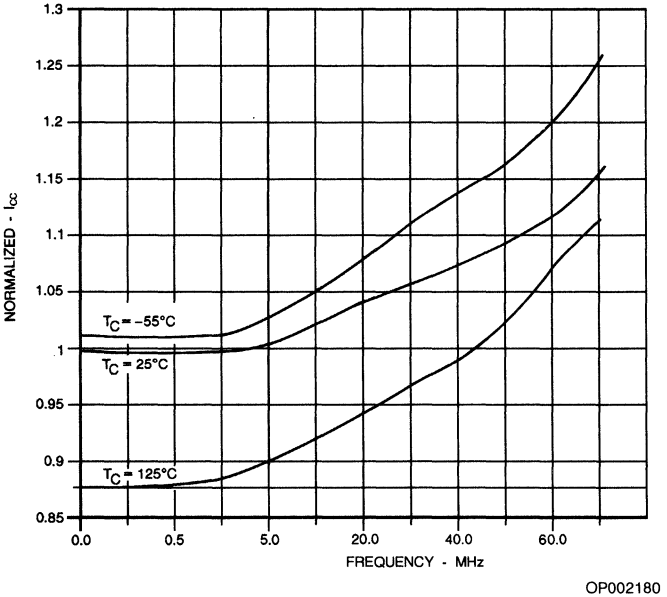
03367-001A

OP002780

X_1 is not a TTL input. It is a crystal connection to an inverting linear oscillator amplifier, and is specified primarily for test convenience.

Figure 13. Am2925A X_1 Input Characteristics (Typical, V_{CC} = 5.0 V)

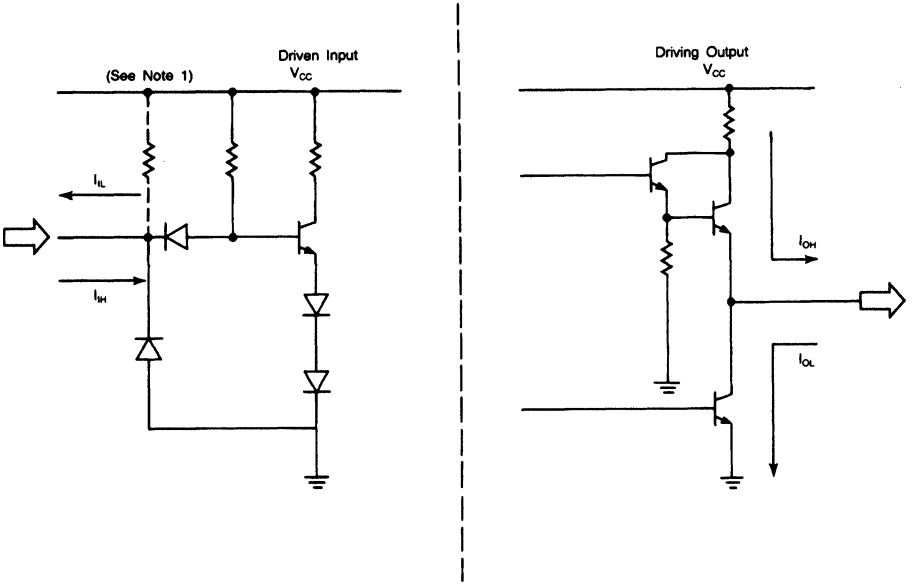
TYPICAL PERFORMANCE CURVES



OP002180

Figure 14. Am2925A I_{CC} Normalized vs Frequency (V_{CC} = 5.5 V)

INPUT/OUTPUT CURRENT DIAGRAM



IC000880

Notes: 1. This resistor exists for input pins 10, 11, 17, 18 and 19. This resistor does not exist in the input circuit structure of the other pins.

Am2960/Am2960-1/Am2960A

Cascadable 16-Bit Error Detection and Correction Unit

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Boosts Memory Reliability
  Corrects all single-bit errors. Detects all double and some triple-bit errors.
- Very High Speed
  Perfect for MOS microprocessor, minicomputer, and main-frame systems.

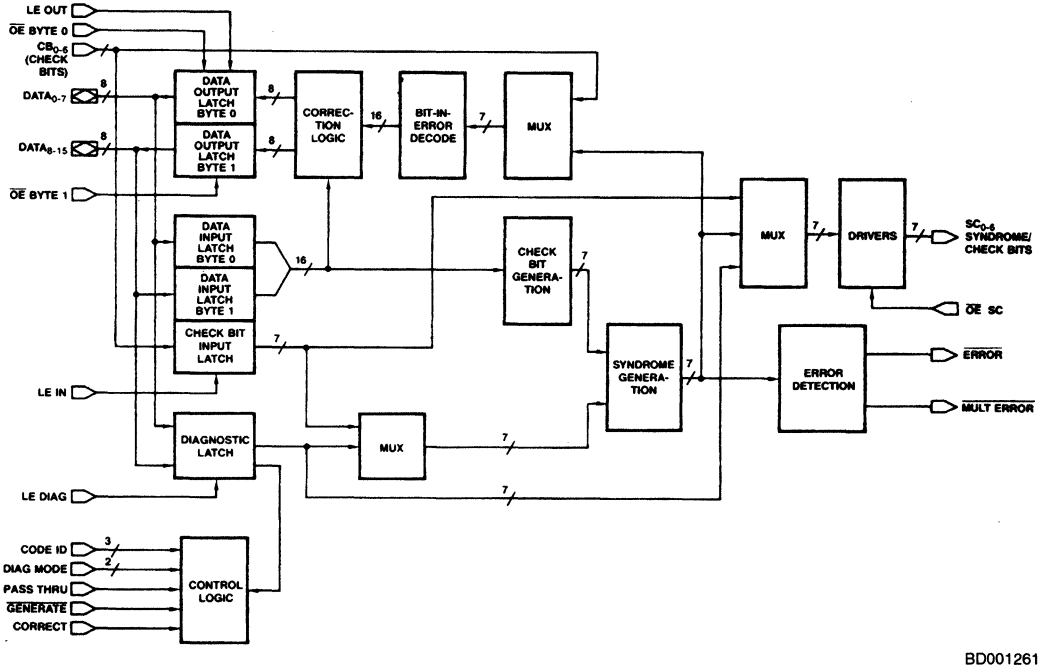
  High performance systems can use the Am2960 EDC in check-only mode to avoid memory system slowdown.
- Handles Data Words From 8 Bits to 64 Bits
  The Am2960 EDC cascades: 1 EDC for 8 bits or 16 bits, 2 for 32 bits, 4 for 64 bits.
- Easy Byte Operations
  Separate byte enables on the data out latch simplify the steps and cut the time required for byte writes.
- Diagnostics Built-In
  The processor may completely exercise the EDC under software control to check for proper operation of the EDC.

GENERAL DESCRIPTION

The Am2960 Error Detection and Correction Unit (EDC) contains the logic necessary to generate check bits on a 16-bit data field according to a modified Hamming Code, and to correct the data word when check bits are supplied. Operating on data read from memory, the Am2960 will correct any single-bit error and will detect all double and some triple-bit errors. For 16-bit words, 6 check bits are used. The Am2960 is expandable to operate on 32-bit words (7 check bits) and 64-bit words (8 check bits). In all configurations, the device makes the error syndrome available on separate outputs for data logging.

The Am2960 also features two diagnostic modes, in which diagnostic data can be forced into portions of the chip to simplify device testing and to execute system diagnostic functions.

BLOCK DIAGRAM

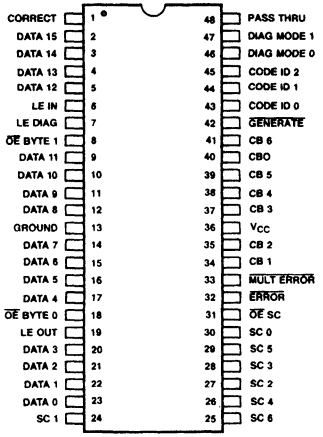


BD001261

Publication# 03565 Rev. E Amendment /0
Issue Date: January 1990

CONNECTION DIAGRAMS
Top View

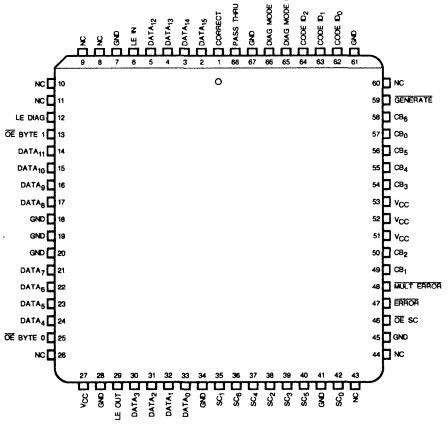
DIPs*



CD001421

*Also available in Flatpack. Pinout is identical to DIPs.

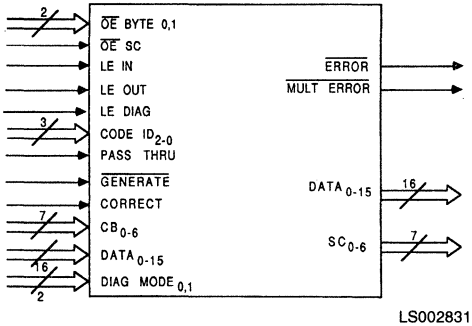
PLCC



CD010232

Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



LS002831

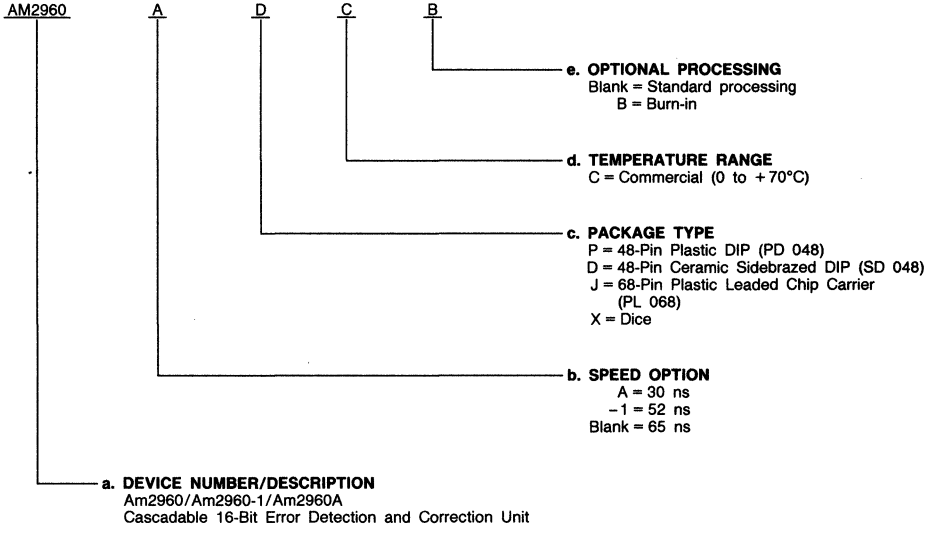
RELATED AMD PRODUCTS

Part No.	Description
Am29368	1M Dynamic Memory Controller
Am29C60A	16-Bit CMOS EDC
Am2968A	256K Dynamic Memory Controller
Am2969	Memory Timing Controller w/EDC Control
Am2971A	Programmable Event Generator
Am29C660	32-Bit CMOS EDC
Am29C668	4M – 64K Configurable Dynamic Memory Controller/Driver
Am29C983	9-Bit x 4-Port Multiple Bus Exchange

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM2960	DC, DCB, PC, PCB, XC, JC, JCB
AM2960A	
AM2960-1	

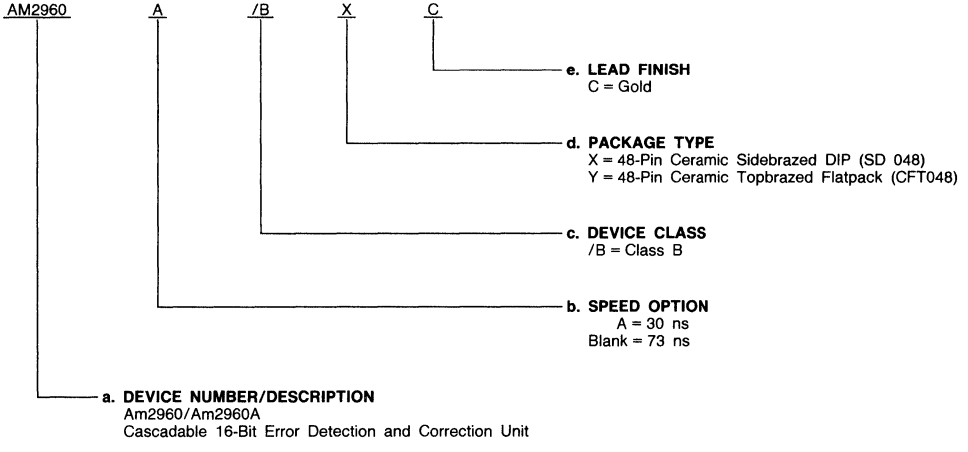
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM2960	/BXC, /BYC
AM2960A	

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consists of Subgroups:
1, 2, 3, 7, 8, 9, 10, 11

PIN DESCRIPTION

$DATA_{0-15}$ Data (Input/Output (16))
These bidirectional data lines provide input to the Data Input Latch and Diagnostic Latch, and receive output from the Data Output Latch. $DATA_0$ is the least significant bit; $DATA_{15}$ the most significant.

CB_{0-6} Check Bits (Input (7))
The check bit lines are used to input check bits for error detection. Also used to input syndrome bits for error correction in 32 and 64-bit configurations.

LE IN Latch Enable – Data Input Latch (Input)
Controls latching of the input data. When HIGH, the Data Input Latch and Check Bit Input Latch follow the input data and input check bits. When LOW, the Data Input Latch and Check Bit Input Latch are latched to their previous state.

$\overline{\text{GENERATE}}$ Generate Check Bits (Input)
When this input is LOW, the EDC is in the Check Bit Generate Mode. When HIGH, the EDC is in the Detect Mode or Correct Mode.

In the Generate Mode the circuit generates the check bits or partial check bits specific to the data in the Data Input Latch. The generated check bits are placed on the SC outputs.

In the Detect or Correct Modes the EDC detects single and multiple errors, and generates syndrome bits based upon the contents of the Data Input Latch and Check Bit Input Latch. In Correct Mode, single-bit errors are also automatically corrected – corrected data is placed at the inputs of the Data Output Latch. The syndrome result is placed on the SC outputs and indicates in a coded form the number of errors and the bit-in-error.

SC_{0-6} Syndrome/Check Bits (Output (7))
These seven lines hold the check/partial-check bits when the EDC is in Generate Mode, and will hold the syndrome/partial syndrome bits when the device is in Detect or Correct Modes. These are 3-state outputs.

$\overline{\text{OE}}$ SC Output Enable, Syndrome/Check Bits (Input)
Syndrome/Check Bits. When LOW, the 3-state output lines SC_{0-6} are enabled. When HIGH, the SC outputs are in the high impedance state.

$\overline{\text{ERROR}}$ Error Detected (Output)
When the EDC is in Detect or Correct Mode, this output will go LOW if one or more syndrome bits are asserted, meaning there are one or more bit errors in the data or check bits. If no syndrome bits are asserted, there are no errors detected and the output will be HIGH. In Generate Mode, $\overline{\text{ERROR}}$ is forced HIGH. (In a 64-bit configuration, $\overline{\text{ERROR}}$ must be externally implemented.)

$\overline{\text{MULT}}$ $\overline{\text{ERROR}}$ Multiple Errors Detected (Output)
When the EDC is in Detect or Correct Mode, this output if LOW indicates that there are two or more bit errors that have been detected. If HIGH, this indicates that either one or no errors have been detected. In Generate mode, $\overline{\text{MULT}}$ $\overline{\text{ERROR}}$ is forced HIGH. (In a 64-bit configuration, $\overline{\text{MULT}}$ $\overline{\text{ERROR}}$ must be externally implemented.)

CORRECT Correct (Input)
When HIGH, this signal allows the correction network to correct any single-bit error in the Data Input Latch (by complementing the bit-in-error) before putting it into the Data Output Latch. When LOW, the EDC will drive data directly from the Data Input Latch to the Data Output Latch without correction.

LE OUT Latch Enable – Data Output Latch (Input)
Controls the latching of the Data Output Latch. When LOW, the Data Output Latch is latched to its previous state. When HIGH, the Data Output Latch follows the output of the Data Input Latch as modified by the correction logic network. In Correct Mode, single-bit errors are corrected by the network before loading into the Data Output Latch. In Detect Mode, the contents of the Data Input Latch are passed through the correction network unchanged into the Data Output Latch. The inputs to the Data Output Latch are unspecified if the EDC is in Generate Mode.

$\overline{\text{OE}}$ BYTE 0, 1 Output Enable Bytes 0, 1 (Input)
These lines control the 3-state outputs for each of the two bytes of the Data Output Latch. When LOW, these lines enable the Data Output Latch, and when HIGH these lines force the Data Output Latch into the high impedance state. The two enable lines can be separately activated to enable only one byte of the Data Output Latch at a time.

PASS THRU Pass Thru (Input)
This line, when HIGH, forces the contents of the Check Bit Input Latch onto the Syndrome/Check Bit outputs (SC_{0-6}) and the unmodified contents of the Data Input Latch onto the inputs of the Data Output Latch.

DIAG $MODE_{0-1}$ Diagnostic Mode Select (Input)
These two lines control the initialization and diagnostic operation of the EDC.

CODE ID_{2-0} Code Identification (Input)
These three bits identify the size of the total data word to be processed and which 16-bit slice of larger data words a particular EDC is processing. The three allowable data word sizes are 16, 32 and 64 bits and their respective modified Hamming codes are designated 16/22, 32/39 and 64/72. Special CODE ID input 001 (ID_2, ID_1, ID_0) is also used to instruct the EDC that the signals CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT and PASS THRU are to be taken from the Diagnostic Latch, rather than from the input control lines.

LE DIAG Diagnostic Latch Enable (Input)
Diagnostic Latch. When HIGH, the Diagnostic Latch follows the 16-bit data on the input lines. When LOW, the outputs of the Diagnostic Latch are latched to their previous states. The Diagnostic Latch holds diagnostic check bits, and internal control signals for CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT and PASS THRU.

FUNCTIONAL DESCRIPTION

EDC Architecture

The EDC Unit is a powerful 16-bit cascadable slice used for check bit generation, error detection, error correction and diagnostics.

As shown in the block diagram, the device consists of the following:

- Data Input Latch
- Check Bit Input Latch
- Check Bit Generation Logic
- Syndrome Generation Logic
- Error Detection Logic
- Error Correction Logic
- Data Output Latch
- Diagnostic Latch
- Control Logic

Data Input Latch

16 bits of data are loaded from the bidirectional DATA lines under control of the Latch Enable input, LE IN. Depending on the control mode, the input data is either used for check bit generation or error detection/correction.

Check Bit Input Latch

Seven check bits are loaded under control of LE IN. Check bits are used in the Error Detection and Error Correction modes.

Check Bit Generation Logic

This block generates the appropriate check bits for the 16 bits of data in the Data Input Latch. The check bits are generated according to a modified Hamming code.

Syndrome Generation Logic

In both Error Detection and Error Correction modes, this logic block compares the check bits read from memory against a newly generated set of check bits produced for the data read in from memory. If both sets of check bits match, then there are no errors. If there is a mismatch, then one or more of the data or check bits is in error.

The syndrome bits are produced by an exclusive-OR of the two sets of check bits. If the two sets of check bits are identical (meaning there are no errors) the syndrome bits will be all zeroes. If there are errors, the syndrome bits can be decoded to determine the number of errors and the bit-in-error.

Error Detection Logic

This section decodes the syndrome bits generated by the Syndrome Generation Logic. If there are no errors in either the input data or check bits, the $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ outputs remain HIGH. If one or more errors are detected, $\overline{\text{ERROR}}$ goes LOW. If two or more errors are detected, both $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ go LOW.

Error Correction Logic

For single errors, the Error Correction Logic complements (corrects) the single data bit in error. This corrected data is loadable into the Data Output Latch, which can then be read onto the bidirectional data lines. If the single error is one of the check bits, the correction logic does not place corrected check bits on the syndrome/check bit outputs. If the corrected check bits are needed, the EDC must be switched to Generate Mode.

Data Output Latch

The Data Output Latch is used for storing the result of an error correction operation. The latch is loaded from the correction logic under control of the Data Output Latch Enable, LE OUT. The Data Output Latch may also be loaded directly from the Data Input Latch under control of the PASS THRU control input.

The Data Output Latch appears as two 8-bit (byte) latches which may be enabled independently for reading onto the bidirectional data lines.

Diagnostic Latch

This is a 16-bit latch loadable from the bidirectional data lines under control of the Diagnostic Latch Enable, LE DIAG. The Diagnostic Latch contains check bit information in one byte and control information in the other byte. The Diagnostic Latch is used for driving the device when in Internal Control Mode, or for supplying check bits when in one of the Diagnostic Modes.

Control Logic

The control logic determines the specific mode the device operates in. Normally the control logic is driven by external control inputs. However, in Internal Control Mode, the control signals are instead read from the Diagnostic Latch.

GENERAL OPERATION

The EDC contains the logic necessary to generate check bits on a 16-bit data field according to a modified Hamming code. Operating on data read from memory, the EDC will correct any single-bit error, and will detect all double and some triple-bit errors. The Am2960 may be configured to operate on 16-bit data words (with 6 check bits), 32-bit data words (with 7 check bits) and 64-bit data words (with 8 check bits). In fact the EDC can be configured to work on data words from 8 to 64 bits. In all configurations, the device makes the error syndrome bits available on separate outputs for error data logging.

Code and Byte Specification

The EDC may be configured in several different ways and operates differently in each configuration. It is necessary to indicate to the device what size data word is involved and which bytes of the data word it is processing. This is done with input lines CODE $ID_{2\text{-}0}$, as shown in Table I. The three modified Hamming codes referred to in Table I are:

- 16/22 – 16 data bits
  – 6 check bits
  – 22 bits in total.
- 32/39 code – 32 data bits
  – 7 check bits
  – 39 bits in total.
- 64/72 code – 64 data bits
  – 8 check bits
  – 72 bits in total.

CODE ID input 001 (ID_2, ID_1, ID_0) is a special code used to operate the device in Internal Control Mode (described later in this section).

Control Mode Selection

The device control lines are $\overline{\text{GENERATE}}$, CORRECT, PASS THRU, DIAG $MODE_{0\text{-}1}$ and CODE $ID_{2\text{-}0}$. Table 3 indicates the operating modes selected by various combinations of the control line inputs.

Diagnostics

Table 2 shows specifically how DIAG $MODE_{0\text{-}1}$ select between normal operation, initialization, and one of two diagnostic modes.

The Diagnostic Modes allow the user to operate the EDC under software control in order to verify proper functioning of the device.

Check and Syndrome Bit Labeling

The check bits generated in the EDC are designated as follows:

- 16-bit configuration – CX, C0, C1, C2, C4, C8;
- 32-bit configuration – CX, C0, C1, C2, C4, C8, C16;
- 64-bit configuration – CX, C0, C1, C2, C4, C8, C16, C32.

Syndrome bits are similarly labeled SX through S32. There are only 6 syndrome bits in the 16-bit configuration, 7 for 32 bits, and 8 syndrome bits in the 64-bit configuration.

Initialize Mode

The outputs of the Data Input Latch are forced to zero (and the Data Input Latches are latched in zero upon removal of the Initialized mode).

The inputs of the Data Output Latch are forced to zeroes. The check bit outputs (SC) are generated to correspond to the all-zero data. $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ are forced HIGH in the Initialize Mode.

Initialize Mode is useful after power up when RAM contents are random. The EDC may be placed in initialize mode and its' outputs written in to all memory locations by the processor.

TABLE 1. HAMMING CODE AND SLICE IDENTIFICATION

CODE ID_2	CODE ID_1	CODE ID_0	Hamming Code and Slice Selected
0	0	0	Code 16/22
0	0	1	Internal Control Mode
0	1	0	Code 32/39, Bytes 0 and 1
0	1	1	Code 32/39, Bytes 2 and 3
1	0	0	Code 64/72, Bytes 0 and 1
1	0	1	Code 64/72, Bytes 2 and 3
1	1	0	Code 64/72, Bytes 4 and 5
1	1	1	Code 64/72, Bytes 6 and 7

TABLE 2. DIAGNOSTIC MODE CONTROL

DIAG $MODE_1$	DIAG $MODE_0$	Diagnostic Mode Selected
0	0	**Non-diagnostic mode.** The EDC functions normally in all modes.
0	1	**Diagnostic Generate.** The contents of the Diagnostic Latch are substituted for the normally generated check bits when in the Generate Mode. The EDC functions normally in the Detect or Correct modes.
1	0	**Diagnostic Detect/Correct.** In the Detect or Correct Mode, the contents of the Diagnostic Latch are substituted for the check bits normally read from the Check Bit Input Latch. The EDC functions normally in the Generate Mode.
1	1	**Initialize.** The outputs of the Data Input Latch are forced to zeroes (and the Data Input Latches are latched in zero upon removal of the Initialize Mode), inputs of the Data Output Latches are forced to zeroes, and the check bits generated correspond to the all-zero data.

HAMMING CODE SELECTION

The Am2960 EDC uses a modified Hamming Code that allows 1) the EDC to be cascaded; 2) all double errors to be detected; 3) the gross error conditions of all 0s or 1s to be detected.

The error correction code can be selected independently of the processor with the exception of diagnostics software.

Diagnostic software run by a processor to check out the EDC system must know specifically which code is being used. This is only a problem when the EDC replaces an existing MSI implementation on an existing computer. In this case, the computer's software must first determine which of two codes (the old one used by the MSI implementation or the new one used by the EDC) is used by the computer's memory system.

This is easily determined by writing a test data word into memory and then examining whether the generated check bits are typical of the old or the new code. From then on the software runs only the diagnostic appropriate for the code used on that particular computer's memory system.

TABLE 3. OPERATING MODES

Operating Mode	Diagnostic Mode**		$\overline{\text{GENERATE}}$	
	DM_1	DM_0	0	1
Normal	0	0	Generate	Correct*
Diagnostic Generate	0	1	Diagnostic Generate	Correct*
Diagnostic Correct	1	0	Generate	Diagnostic Correct*
Initialize	1	1	Initialize	Initialize
Pass Thru	When PASS THRU is asserted, the Operating Mode is defaulted to the Pass Thru Mode.			

*Correct if the CORRECT Input is HIGH, Detect if the CORRECT Input is LOW.

**In Code $ID_{2\text{-}0}001(ID_2, ID_1, ID_0)$ DM_1 and DM_0 are taken from the Diagnostic Latch.

FUNCTIONAL EQUATIONS

The following equations and tables describe in detail how the output values of the AM2960 EDC are determined as a function of the value of the inputs and the internal states. Be sure to carefully read the following definitions of symbols before examining the tables.

Definitions

D_i ← ($DATA_i$ if LE IN is HIGH or the output of bit i of the Data Input Latch if LE IN is LOW)
C_i ← (CB_i if LE IN is HIGH or the output of bit i of the Check Bit Latch if LE IN is LOW)
DL_i ← Output of bit i of the Diagnostic Latch
S_i ← Internally generated syndromes (same as outputs of SC_i if outputs enabled)
PA ← D0⊕D1⊕D2⊕D4⊕D6⊕D8⊕D10⊕D12
PB ← D0⊕D1⊕D2⊕D3⊕D4⊕D5⊕D6⊕D7
PC ← D8⊕D9⊕D10⊕D11⊕D12⊕D13⊕D14⊕D15
PD ← D0⊕D3⊕D4⊕D7⊕D9⊕D10⊕D13⊕D15
PE ← D0⊕D1⊕D5⊕D6⊕D7⊕D11⊕D12⊕D13
PF ← D2⊕D3⊕D4⊕D5⊕D6⊕D7⊕D14⊕D15
PG_1 ← D0⊕D4⊕D6⊕D7
PG_2 ← D1⊕D2⊕D3⊕D5
PG_3 ← D8⊕D9⊕D11⊕D14
PG_4 ← D10⊕D12⊕D13⊕D15

Error Signals

$$\overline{\text{ERROR}} \leftarrow \overline{(\overline{S6} \cdot (\overline{ID_1} + \overline{ID_2}))} \cdot \overline{S5} \cdot \overline{S4} \cdot \overline{S3} \cdot \overline{S2} \cdot \overline{S1} \cdot \overline{S0} + \text{GENERATE} + \text{INITIALIZE} + \text{PASSTHRU}$$

$$\overline{\text{MULT ERROR}} \text{ (16 and 32-Bit Modes)} \leftarrow \overline{(\overline{S6} \cdot ID_1) \oplus S5 \oplus S4 \oplus S3 \oplus S2 \oplus S1 \oplus S0)} \text{ (ERROR)} + \text{TOME} + \text{GENERATE} + \text{PASSTHRU} + \text{INITIALIZE}$$

$$\overline{\text{MULT ERROR}} \text{ (64-Bit Modes)} \leftarrow \overline{\text{TOME}} + \text{GENERATE} + \text{PASSTHRU} + \text{INITIALIZE}$$

TABLE 4. TOME (Three or More Errors)*

			S0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
			**S6	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
			S5	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
			S4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
S1	S2	S3																	
0	0	0		0	0	0	1	0	1	1	1	0	1	1	1	0	0	0	0
0	0	1		0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
0	1	0		0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	1		1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	0	0		0	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	0	1		0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
1	1	0		1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1		0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

*(S6, S5,...S0 are internal syndromes except in Modes 010, 100, 101, 110, 111 (CODE ID_2, ID_1, ID_0). In these modes the syndromes are input over the Check-Bit lines. S6←C6, S5←C5, ...S1←C1, S0←C0.
**The S6 internal syndrome is always forced to 0 in CODE ID 000.

SC Outputs

Tables 5 to 9 show how outputs SC_{0-6} are generated in each control mode for various CODE IDs (internal control mode not applicable).

TABLE 5. GENERATE MODE (Check Bits)

GENERATE Mode (Check Bits)	CODE ID_{2-0}						
	000	010	011	100	101	110	111
SC_0←	$PG_2 \oplus PG_3$	$PG_1 \oplus PG_3$	$PG_2 \oplus PG_4 \oplus CB_0$	$PG_2 \oplus PG_3$	$PG_2 \oplus PG_3$	$PG_1 \oplus PG_4$	$PG_1 \oplus PG_4$
SC_1←	PA	PA	$PA \oplus CB_1$	PA	PA	PA	PA
SC_2←	$\overline{PD}$	$\overline{PD}$	$PD \oplus CB_2$	$\overline{PD}$	PD	PD	PD
SC_3←	$\overline{PE}$	$\overline{PE}$	$PE \oplus CB_3$	$\overline{PE}$	PE	PE	PE
SC_4←	PF	PF	$PF \oplus CB_4$	PF	PF	PF	PF
SC_5←	PC	PC	$PC \oplus CB_5$	PC	PC	PC	PC
SC_6←	1	PB	$PC \oplus CB_6$	PB	PB	PB	PB

TABLE 6. DETECT AND CORRECT MODES (Syndromes)

Detect and Correct Modes (Syndromes)	CODE ID_{2-0}						
	000	010	011*	100	101	110	111
SC_0←	$PG_2 \oplus PG_3 \oplus C0$	$PG_1 \oplus PG_3 \oplus C0$	$PG_2 \oplus PG_4 \oplus CB_0$	$PG_2 \oplus PG_3 \oplus C0$	$PG_2 \oplus PG_3$	$PG_1 \oplus PG_4$	$PG_1 \oplus PG_4$
SC_1←	$PA \oplus C1$	$PA \oplus C1$	$PA \oplus CB_1$	$PA \oplus C1$	PA	PA	PA
SC_2←	$\overline{PD} \oplus C2$	$\overline{PD} \oplus C2$	$PD \oplus CB_2$	$\overline{PD} \oplus C2$	PD	PD	PD
SC_3←	$\overline{PE} \oplus C3$	$\overline{PE} \oplus C3$	$PE \oplus CB_3$	$\overline{PE} \oplus C3$	PE	PE	PE
SC_4←	$PF \oplus C4$	$PF \oplus C4$	$PF \oplus CB_4$	$PF \oplus C4$	PF	PF	PF
SC_5←	$PC \oplus C5$	$PC \oplus C5$	$PC \oplus CB_5$	$PC \oplus C5$	PC	PC	PC
SC_6←	1	$PB \oplus C6$	$PC \oplus CB_6$	PB	PB	$PB \oplus C6$	$PB \oplus C6$

*In CODE ID_{2-0} 011 the Check-Bit Latch is forced transparent; the Data Latch operates normally.

TABLE 7. DIAGNOSTIC READ MODE

Diagnostic Read Mode	CODE ID_{2-0}						
	000	010	011*	100	101	110	111
SC_0←	$PG_2 \oplus PG_3 \oplus DL_0$	$PG_1 \oplus PG_3 \oplus DL_0$	$PG_2 \oplus PG_4 \oplus CB_0$	$PG_2 \oplus PG_3 \oplus DL_0$	$PG_2 \oplus PG_3$	$PG_1 \oplus PG_4$	$PG_1 \oplus PG_4$
SC_1←	$PA \oplus DL_1$	$PA \oplus DL_1$	$PA \oplus CB_1$	$PA \oplus DL_1$	PA	PA	PA
SC_2←	$\overline{PD} \oplus DL_2$	$\overline{PD} \oplus DL_2$	$PD \oplus CB_2$	$\overline{PD} \oplus DL_2$	PD	PD	PD
SC_3←	$\overline{PE} \oplus DL_3$	$\overline{PE} \oplus DL_3$	$PE \oplus CB_3$	$\overline{PE} \oplus DL_3$	PE	PE	PE
SC_4←	$PF \oplus DL_4$	$PF \oplus DL_4$	$PF \oplus CB_4$	$PF \oplus DL_4$	PF	PF	PF
SC_5←	$PC \oplus DL_5$	$PC \oplus DL_5$	$PC \oplus CB_5$	$PC \oplus DL_5$	PC	PC	PC
SC_6←	1	$PB \oplus DL_6$	$PC \oplus CB_6$	PB	PB	$PB \oplus DL_6$	$PB \oplus DL_7$

*In CODE ID_{2-0} 011 the Check-Bit Latch is forced transparent; the Data Latch operates normally.

TABLE 8. DIAGNOSTIC WRITE MODE

Diagnostic Write Mode	CODE ID_{2-0}						
	000	010	011*	100	101	110	111
$SC_0 \leftarrow$	DL_0	DL_0	CB_0	DL_0	1	1	1
$SC_1 \leftarrow$	DL_1	DL_1	CB_1	DL_1	1	1	1
$SC_2 \leftarrow$	DL_2	DL_2	CB_2	DL_2	1	1	1
$SC_3 \leftarrow$	DL_3	DL_3	CB_3	DL_3	1	1	1
$SC_4 \leftarrow$	DL_4	DL_4	CB_4	DL_4	1	1	1
$SC_5 \leftarrow$	DL_5	DL_5	CB_5	DL_5	1	1	1
$SC_6 \leftarrow$	1	DL_6	CB_6	1	1	DL_6	DL_7

*In CODE ID_{2-0} 011 the Check-Bit Latch is forced transparent; the Data Input Latch operates normally.

TABLE 9. PASS THRU MODE

PASS THRU Mode	CODE ID_{2-0}						
	000	010	011*	100	101	110	111
$SC_0 \leftarrow$	C0	C0	CB_0	C0	1	1	1
$SC_1 \leftarrow$	C1	C1	CB_1	C1	1	1	1
$SC_2 \leftarrow$	C2	C2	CB_2	C2	1	1	1
$SC_3 \leftarrow$	C3	C3	CB_3	C3	1	1	1
$SC_4 \leftarrow$	C4	C4	CB_4	C4	1	1	1
$SC_5 \leftarrow$	C5	C5	CB_5	C5	1	1	1
$SC_6 \leftarrow$	1	C6	CB_6	1	1	C6	C6

*In CODE ID_{2-0} 011 the Check-Bit Latch is forced transparent; the Data Input Latch operates normally.

Data Correction

Tables 10 to 16 shows which data output bits are corrected (inverted) depending upon the syndromes and the CODE ID position. Note that the syndromes that determine data correction are in some cases syndromes input externally via the CB inputs and in some cases syndromes generated internally by that EDC (S_i are the internal syndromes and are the same as the value of the SC_i output of that EDC if enabled).

The tables show the number of data bit inverted (corrected) if any for the CODE ID and syndrome combination.

TABLE 10. CODE ID_{2-0} = 000*

	S5	0	0	0	0	1	1	1	1
	S4	0	0	1	1	0	0	1	1
	S3	0	1	0	1	0	1	0	1
S2	S1								
0	0	–	–	–	5	–	11	14	–
0	1	–	1	2	6	8	12	–	–
1	0	–	–	3	7	9	13	15	–
1	1	–	0	4	–	10	–	–	–

*Unlisted S combinations are no correction.

TABLE 11. CODE ID_{2-0} = 010*

	CB_6	0	0	0	0	1	1	1	1
	CB_5	1	1	1	1	0	0	0	0
	CB_4	0	0	1	1	0	0	1	1
	CB_3	0	1	0	1	0	1	0	1
CB_2	CB_1								
0	0	–	11	14	–	–	–	–	5
0	1	8	12	–	–	–	1	2	6
1	0	9	13	15	–	–	–	3	7
1	1	10	–	–	–	–	0	4	–

*Unlisted CB combinations are no correction.

TABLE 12. CODE ID_{2-0} = 011*

S6	0	0	0	0	1	1	1	1
S5	0	0	0	0	1	1	1	1
S4	0	0	1	1	0	0	1	1
S3	0	1	0	1	0	1	0	1
S2 S1								
0 0	–	–	–	5	–	11	14	–
0 1	–	1	2	6	8	12	–	–
1 0	–	–	3	7	9	13	15	–
1 1	–	0	4	–	10	–	–	–

*Unlisted S combinations are no correction.

TABLE 13. CODE ID_{2-0} = 100*

CB_0	0	0	0	0	1	1	1	1
CB_6	0	0	0	0	1	1	1	1
CB_5	1	1	1	1	0	0	0	0
CB_4	0	0	1	1	0	0	1	1
CB_3	0	1	0	1	0	1	0	1
CB_2CB_1								
0 0	–	11	14	–	–	–	–	5
0 1	8	12	–	–	–	1	2	6
1 0	9	13	15	–	–	–	3	7
1 1	10	–	–	–	–	0	4	–

*Unlisted CB combinations are no correction.

TABLE 14. CODE ID_{2-0} = 101*

CB_0	0	0	0	0	1	1	1	1
CB_6	0	0	0	0	1	1	1	1
CB_5	0	0	0	0	1	1	1	1
CB_4	0	0	1	1	0	0	1	1
CB_3	0	1	0	1	0	1	0	1
CB_2CB_1								
0 0	–	–	–	5	–	11	14	–
0 1	–	1	2	6	8	12	–	–
1 0	–	–	3	7	9	13	15	–
1 1	–	0	4	–	10	–	–	–

*Unlisted CB combinations are no correction.

TABLE 15. CODE ID_{2-0} = 110*

CB_0	0	0	0	0	1	1	1	1
CB_6	1	1	1	1	0	0	0	0
CB_5	0	0	0	0	1	1	1	1
CB_4	0	0	1	1	0	0	1	1
CB_3	0	1	0	1	0	1	0	1
CB_2CB_1								
0 0	–	–	–	5	–	11	14	–
0 1	–	1	2	6	8	12	–	–
1 0	–	–	3	7	9	13	15	–
1 1	–	0	4	–	10	–	–	–

*Unlisted CB combinations are no correction.

TABLE 16. CODE ID_{2-0} = 111*

CB_0	0	0	0	0	1	1	1	1
CB_6	1	1	1	1	0	0	0	0
CB_5	1	1	1	1	0	0	0	0
CB_4	0	0	1	1	0	0	1	1
CB_3	0	1	0	1	0	1	0	1
CB_2CB_1								
0 0	–	11	14	–	–	–	–	5
0 1	8	12	–	–	–	1	2	6
1 0	9	13	15	–	–	–	3	7
1 1	10	–	–	–	–	0	4	–

*Unlisted CB combinations are no correction.

16-BIT DATA WORD CONFIGURATION

The 16-bit format consists of 16 data bits, 6 check bits and is referred to as 16/22 code (see Figure 5.)

The 16-bit configuration is shown in Figure 6.

Generate Mode

In this mode, check bits will be generated that correspond to the contents of the Data Input Latch. The check bits generated are placed on the outputs SC_{0-5} (SC_6 is a logical one, or high).

Check bits are generated according to a modified Hamming code. Details of the code for check bit generation are contained in Table 17. Each check bit is generated as either an XOR or XNOR of eight of the 16 data bits as indicated in the table. The XOR function results in an even parity check bit, the XNOR is an odd parity check bit.

Figure 1 shows the data flow in the Generate Mode.

Detect Mode

In this mode the device examines the contents of the Data Input Latch against the Check Bit Input Latch, and will detect all single-bit errors, all double-bit errors and some triple-bit errors. If one or more errors are detected, $\overline{ERROR}$ goes LOW. If two or more errors are detected, $\overline{MULT\ ERROR}$ goes LOW. Both error indicators are HIGH if there are no errors.

Also available on device outputs SC_{0-5} are the syndrome bits generated by the error detection step. The syndrome bits may be decoded to determine if a bit error was detected and, for single-bit errors, which of the data or check bits is in error. Table 18 gives the chart for decoding the syndrome bits generated by the 16-bit configuration (as an example, if the syndrome bits SX/S0/S1/S2/S4/S8 were 101001, this would be decoded to indicate that there is a single-bit error at data bit 9). If no error is detected, the syndrome bits will all be zeroes.

In Detect Mode, the contents of the Data Input Latch are driven directly to the inputs of the Data Output Latch without correction.

Correct Mode

In this mode, the EDC functions the same as in Detect Mode except that the correction network is allowed to correct (complement) any single-bit error of the Data Input Latch before putting it onto the inputs of the Data Output Latch. (See Figure 2.) If multiple errors are detected, the output of the correction network is unspecified. If the single-bit error is a check bit, there is no automatic correction. If check bit correction is desired, this can be done by placing the device in Generate Mode to produce a correct check bit sequence for the data in the Data Input Latch.

Pass Thru Mode

In this mode, the unmodified contents of the Data Input Latch are placed on the inputs of the Data Output Latch, and the contents of the Check Bit Input Latch are placed on outputs SC_{0-5}. $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ are forced HIGH in this mode.

Diagnostic Latch

The Diagnostic Latch serves both for diagnostic uses and internal control uses. It is loaded from the DATA lines under the control of LE DIAG. Table 19 shows the loading definitions for the DATA lines.

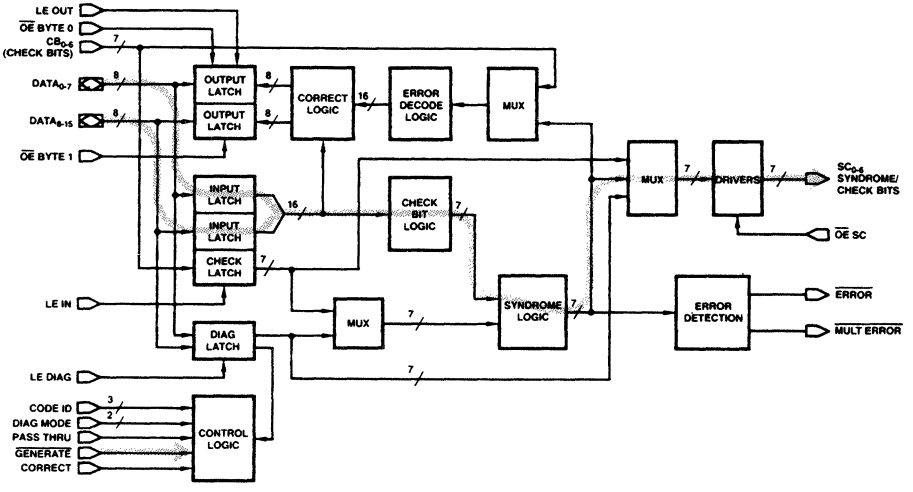
Diagnostic Generate/Diagnostic Detect/ Diagnostic Correct

These are special diagnostic modes selected by DIAG $MODE_{0-1}$ where either normal check bit inputs or outputs are substituted for by check bits loaded into the Diagnostic Latch (see Table 3 for details). Figures 3 and 4 illustrate the flow of data during the two diagnostic modes.

Internal Control Mode

This mode is selected by CODE ID_{2-0} input 001 (ID_2, ID_1, ID_0). When in Internal Control Mode, the EDC takes the CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT and PASS THRU control signals from the internal Diagnostic Latch rather than from the external input lines.

Table 19 gives the format for loading the Diagnostic Latch.



DF000280

Figure 1. Check Bit Generation Data Path

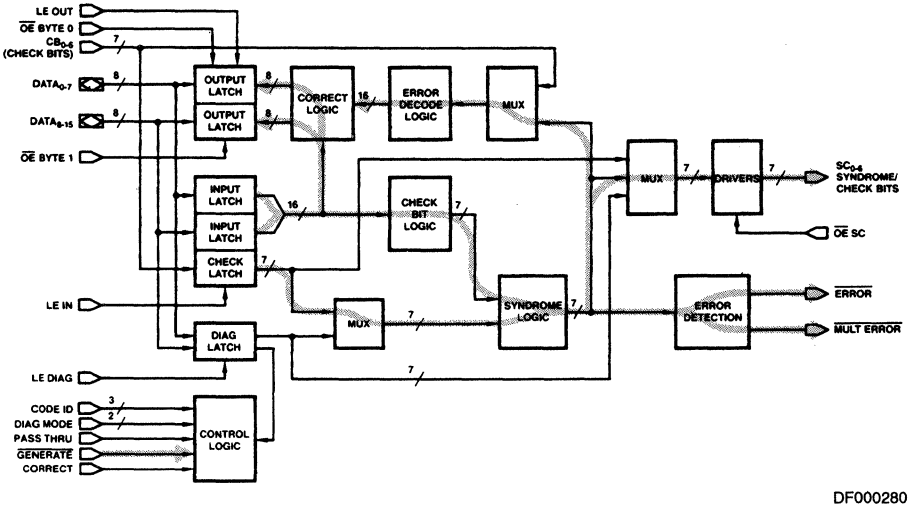


Figure 2. Error Detection and Correction Data Path

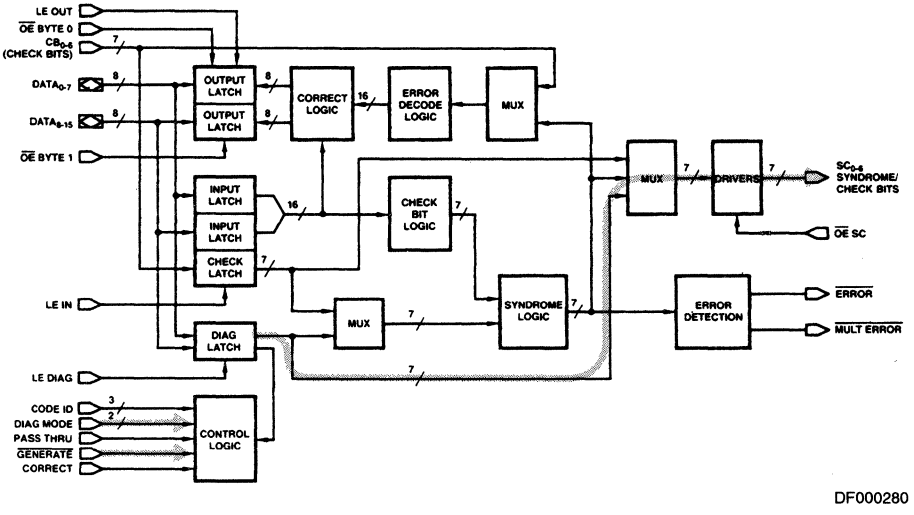


Figure 3. Diagnostic Check Bit Generation Data Path

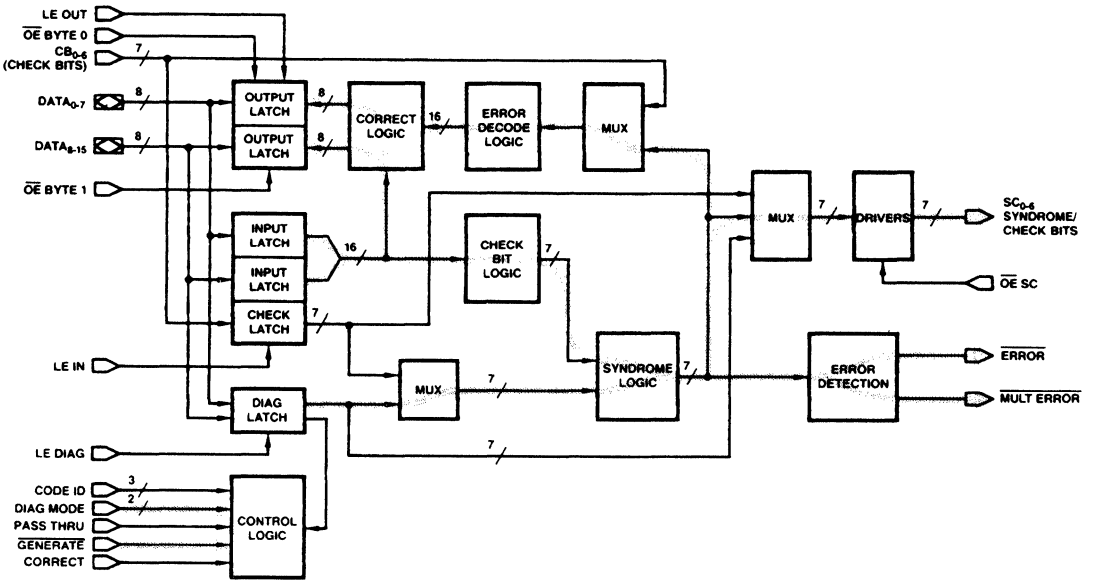
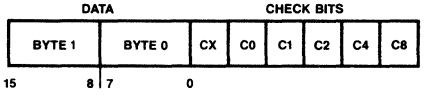


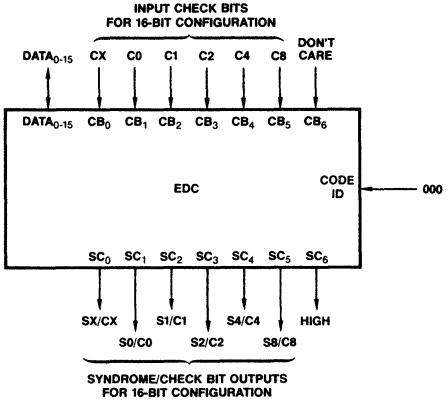
Figure 4. Diagnostic Detect and Correct Data Path



DF000220

Uses Modified Hamming Code 16/22
- 16 data bits
- 6 check bits
- 22 bits in total

Figure 5. 16-Bit Data Format



DF000210

Figure 6. 16-Bit Configuration

SYNDROME DECODE TO BIT-IN-ERROR 8-BIT MODE

Syndrome Bits S0	S1	S4 S2	0 0	1 0	0 1	1 1
0	0		*	C4	C2	5
0	1		C1	3	TM	7
1	0		C0	2	1	6
1	1		TM	4	0	TM

* – no errors detected
TM – two or more errors

TABLE 17. 16-BIT MODIFIED HAMMING CODE – CHECK BIT ENCODE CHART

Generated Check Bits	Parity	Participating Data Bits															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X

The check bit is generated as either an XOR or XNOR of the eight data bits noted by an "X" in the table.

TABLE 18. SYNDROME DECODE TO BIT-IN-ERROR

Syndrome Bits SX S0 S1	S8 S4 S2								
		0 0 0	1 0 0	0 1 0	1 1 0	0 0 1	1 0 1	0 1 1	1 1 1
0 0 0		*	C8	C4	T	C2	T	T	M
0 0 1		C1	T	T	15	T	13	7	T
0 1 0		C0	T	T	M	T	12	6	T
0 1 1		T	10	4	T	0	T	T	M
1 0 0		CX	T	T	14	T	11	5	T
1 0 1		T	9	3	T	M	T	T	M
1 1 0		T	8	2	T	1	T	T	M
1 1 1		M	T	T	M	T	M	M	T

* – no errors detected
Number – location of the single bit-in-error
T – two errors detected
M – three or more errors detected

TABLE 19. 16-BIT DIAGNOSTIC LATCH LOADING FORMAT

Data Bit	Internal Function
0	Diagnostic Check Bit X
1	Diagnostic Check Bit 0
2	Diagnostic Check Bit 1
3	Diagnostic Check Bit 2
4	Diagnostic Check Bit 4
5	Diagnostic Check Bit 8
6, 7	Don't Care
8	CODE ID 0
9	CODE 1D 1
10	CODE ID 2
11	DIAG MODE 0
12	DIAG MODE 1
13	CORRECT
14	PASS THRU
15	Don't Care

32-BIT DATA WORD CONFIGURATION

The 32-bit format consists of 32 data bits and 7 check bits and is referred to as 32/39 code (see Figure 7).

The 32-bit configuration is shown in Figure 8.

The upper EDC (Slice 0/1) handles the least significant bytes 0 and 1 – the external DATA lines 0 to 15 are connected to the same numbered inputs of the upper device. The lower EDC (Slice 2/3) handles the most significant bytes 2 and 3–the external DATA lines for bits 16 to 31 are connected to inputs $DATA_0$ through $DATA_{15}$ respectively.

The valid syndrome and check bit outputs are those of Slice 2/3 as shown in the diagram. In Correct Mode these must be read into Slice 0/1 via the CB inputs and are selected by the MUX as inputs to the bit-in-error decoder (see block diagram). This requires external buffering and output enabling of the check bit lines, as shown. The $\overline{OE}$ SC signal can be used to control enabling of check bit inputs – when syndrome outputs are enabled, the external check bit inputs will be disabled.

The valid $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ outputs are those of the Slice 2/3. The $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ outputs of Slice 0/1 are unspecified. All of the latch enables and control signals must be input to both of the devices.

Generate Mode

In this mode, check bits will be generated that correspond to the contents of the Data Input Latch. The check bits generated are placed on the outputs SC_{0-6} of Slice 2/3.

Check bits are generated according to a modified Hamming code. Details of the code for check bit generation are contained in Table 23. Check bits are generated as either an XOR or XNOR of 16 of the 32 data bits as indicated in the table. The XOR function results in an even parity check bit, the XNOR in an odd parity check bit.

Detect Mode

In this mode, the device examines the contents of the Data Input Latch against the Check Bit Input Latch, and will detect all single-bit errors, all double-bit errors, and some triple-bit errors. If one or more errors are detected, $\overline{ERROR}$ goes LOW. If two or more errors are detected, $\overline{MULT\ ERROR}$ goes LOW. Both error indicators are HIGH if there are no errors. The valid $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ signals are those of Slice 2/3 – those of Slice 0/1 are undefined.

Also available on Slice 2/3 outputs SC_{0-6} are the syndrome bits generated by the error detection step. The syndrome bits may be decoded to determine if a bit error was detected and,

Also available on Slice 2/3 outputs SC_{0-6} are the syndrome bits generated by the error detection step. The syndrome bits may be decoded to determine if a bit error was detected and, for single-bit errors, which of the data or check bits is in error. Table 20 gives the chart for decoding the syndrome bits generated for the 32-bit configuration (as an example, if the syndrome bits SX/S0/S1/S2/S4/S8/S16 were 0010011, this would be decoded to indicate that there is a single-bit error at data bit 25). If no error is detected, the syndrome bits will be all zeroes.

In Detect Mode, the contents of the Data Input Latch are driven directly to the inputs of the Data Output Latch without corrections.

Correct Mode

In this mode, the EDC functions the same as in Detect Mode except that the correction network is allowed to correct (complement) any single-bit error of the Data Input Latch before putting it onto the inputs of the Data Output Latch. If multiple errors are detected, the output of the correction network is unspecified. If the single-bit error is a check bit, there is no automatic correction; if desired, this would be done by placing the device in Generate Mode to produce a correct check bit sequence for the data in the Data Input Latch.

For data correction, both Slices 0/1 and 2/3 require access to the syndrome bits on Slice 2/3's outputs SC_{0-6}. Slice 2/3 has access to these syndrome bits through internal data paths, but for Slice 0/1 they must be read through the inputs CB_{0-6}. The device connections for this are shown in Figure 8. When in Correct Mode, the SC outputs must be enabled so that they are available for reading in through the CB inputs.

Pass Thru Mode

In this mode, the unmodified contents of the Data Input Latch are placed on the inputs of the Data Output latch, and the contents of the Check Bit Input Latch are placed on outputs SC_{0-6} of Slice 2/3. $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ are forced HIGH in this mode.

Internal Control Mode

This mode is selected by CODE ID_{2-0} input 001 (ID_2, ID_1, ID_0). When in Internal Control Mode, the EDC takes the CODE ID_{2-0} DIAG $MODE_{0-1}$, CORRECT and PASS THRU control signals from the Internal Diagnostic Latch rather than from the external input lines.

Table 22 gives the format for loading the Diagnostic Latch.

TABLE 20.
SYNDROME DECODE TO BIT-IN-ERROR

Syndrome Bits				S16	0	1	0	1	0	1	0	1
				S8	0	0	1	1	0	0	1	1
				S4	0	0	0	0	1	1	1	1
SX	S0	S1	S2									
0	0	0	0		*	C16	C8	T	C4	T	T	30
0	0	0	1		C2	T	T	27	T	5	M	T
0	0	1	0		C1	T	T	25	T	3	15	T
0	0	1	1		T	M	13	T	23	T	T	M
0	1	0	0		C0	T	T	24	T	2	M	T
0	1	0	1		T	1	12	T	22	T	T	M
0	1	1	0		T	M	10	T	20	T	T	M
0	1	1	1		16	T	T	M	T	M	M	T
1	0	0	0		CX	T	T	M	T	M	14	T
1	0	0	1		T	M	11	T	21	T	T	M
1	0	1	0		T	M	9	T	19	T	T	31
1	0	1	1		M	T	T	29	T	7	M	T
1	1	0	0		T	M	8	T	18	T	T	M
1	1	0	1		17	T	T	28	T	6	M	T
1	1	1	0		M	T	T	26	T	4	M	T
1	1	1	1		T	0	M	T	M	T	T	M

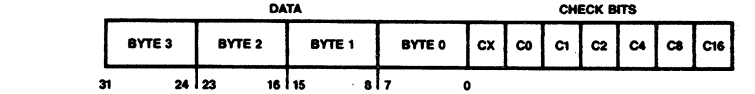
* – no errors detected
Number – number of the single bit-in-error
T – two errors detected
M – three or more errors detected

Diagnostic Latch

The Diagnostic Latch serves both for diagnostic uses and internal control uses. It is loaded from the DATA lines under the control of LE DIAG. Table 22 shows the loading definitions for the DATA lines.

Diagnostic Generate/Diagnostic Detect/ Diagnostic Correct

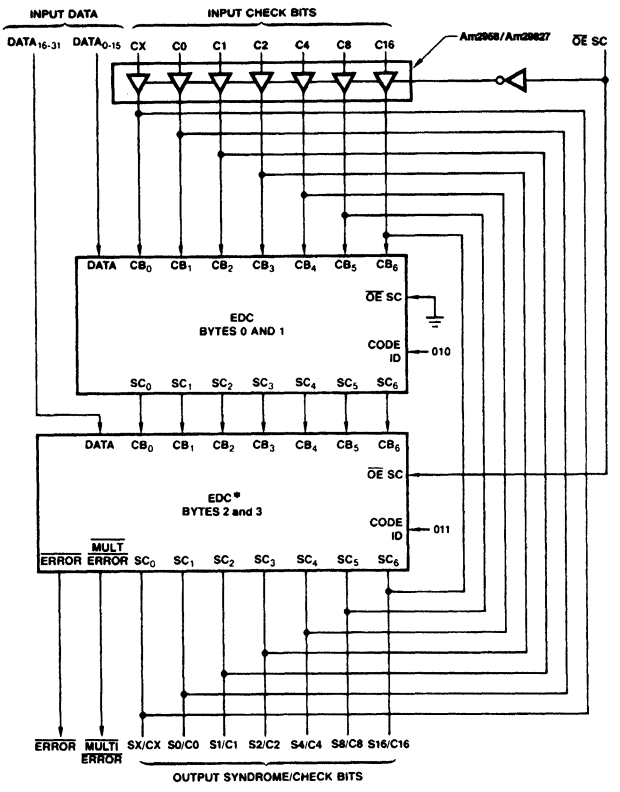
These are special diagnostic modes selected by DIAG $MODE_{0-1}$ where either normal check bit inputs or outputs are substituted for by check bits from the Diagnostic Latch (see Table 2 for details). Figures 3 and 4 illustrate the flow of data during the two diagnostic modes.



DF000460

Uses Modified Hamming Code 32/39
– 32 data bits – 7 check bits – 39 bits in total

Figure 7. 32-Bit Data Format



DF000111

*Check Bit Latch is Forced Transparent in this Code ID Combination for this Slice.

Figure 8. 32-Bit Configuration

TABLE 21.
KEY AC CALCULATIONS FOR THE 32-BIT CONFIGURATION

32-Bit Propagation Delay		Component Delay from Am2960 AC Specifications
From	To	
DATA	Check Bits Out	(Data to SC) + (CB to SC, CODE ID 011)
DATA In	Corrected DATA Out	(DATA to SC) + (CB to SC, CODE ID 011) + (CB to DATA, CODE ID 010)
DATA	Syndromes Out	(DATA to SC) + (CB to SC, CODE ID 011)
DATA	$\overline{\text{ERROR}}$ for 32 Bits	(DATA to SC) + (CB to $\overline{\text{ERROR}}$, CODE ID 011)
DATA	$\overline{\text{MULT ERROR}}$ for 32 Bits	(DATA to SC) + (CB to $\overline{\text{MULT ERROR}}$, CODE ID 011)

TABLE 22.
32-BIT DIAGNOSTIC LATCH LOADING FORMAT

Data Bit	Internal Function
0	Diagnostic Check Bit X
1	Diagnostic Check Bit 0
2	Diagnostic Check Bit 1
3	Diagnostic Check Bit 2
4	Diagnostic Check Bit 4
5	Diagnostic Check Bit 8
6	Diagnostic Check Bit 16
7	Don't Care
8	Slice 0/1 – CODE ID 0
9	Slice 0/1 – CODE ID 1
10	Slice 0/1 – CODE ID 2
11	Slice 0/1 – DIAG MODE 0
12	Slice 0/1 – DIAG MODE 1
13	Slice 0/1 – CORRECT
14	Slice 0/1 – PASS THRU
15	Don't Care
16-23	Don't Care
24	Slice 2/3 – CODE ID 0
25	Slice 2/3 – CODE ID 1
26	Slice 2/3 – CODE ID 2
27	Slice 2/3 – DIAG MODE 0
28	Slice 2/3 – DIAG MODE 1
29	Slice 2/3 – CORRECT
30	Slice 2/3 – PASS THRU
31	Don't Care

TABLE 23.
32-BIT MODIFIED HAMMING CODE – CHECK BIT ENCODE CHART

Generated Check Bits	Parity	Participating Data Bits															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)	X				X		X	X	X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								

Generated Check Bits	Parity	Participating Data Bits															
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
CX	Even (XOR)		X	X	X		X					X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X

The check bit is generated as either an XOR or XNOR of the sixteen data bits noted by an "X" in the table.

64-BIT DATA WORD CONFIGURATION

The 64-bit format consists of 64 data bits, 8 check bits and is referred to as 64/72 code (see Figure 9.).

The configuration to process 64-bit format is shown in Figure 6. In this configuration, a portion of the syndrome generation and error detection is implemented externally of the EDCs in MSI. For error correction, the syndrome bits generated must be read back into all four EDCs through the CB inputs. This necessitates the check bit buffering shown in the connection diagram of Figure 10. The $\overline{OE}$ SC signal can control the check bit enabling; when syndrome bit outputs are enabled, the external check bit lines will be disabled so that the syndrome bits may be read onto the CB inputs.

The error detection signals for the 64-bit configuration differ from the 16- and 32-bit configurations. The $\overline{ERROR}$ signal functions the same: it is LOW if one or more errors are detected, and HIGH if no errors are detected. The DOUBLE ERROR signal is HIGH if and only if a double-bit error is detected; it is LOW otherwise. All of the $\overline{MULT\ ERROR}$ outputs of the four devices are valid. $\overline{MULT\ ERROR}$ is LOW for all three ERROR cases and some DOUBLE ERROR combinations. (See TOME definition in Functional Equations section.) It is HIGH if either zero or one errors are detected.

This is a different meaning for $\overline{MULT\ ERROR}$ than in other configurations.

Generate Mode

In this mode, check bits will be generated that correspond to the contents of the Data Input Latch. The check bits generated appear at the outputs of the XOR gates as indicated in Figure 10.

Check bits are generated according to a modified Hamming code. Details of the code for check bit generation are contained in Table 25. Check bits are generated as either an XOR or XNOR of 32 of the 64 bits as indicated in the table. The XOR function results in an even parity check bit, the XNOR in an odd parity check bit.

Detect Mode

In this mode, the device examines the contents of the Data Input Latch against the Check Bit Input Latch, and will detect all single-bit errors, all double-bit errors and some triple-bit errors. If one or more errors are detected, $\overline{ERROR}$ goes LOW. If exactly two errors are detected, $\overline{DOUBLE\ ERROR}$ goes HIGH. If three or more errors are detected, $\overline{MULT\ ERROR}$ goes LOW – the $\overline{MULT\ ERROR}$ output of any of the four EDCs may be used.

Available as XOR gate outputs are the generated syndrome bits (see Figure 10). The syndrome bits may be decoded to determine if a bit error was detected and, for single-bit errors, which of the data or check bits is in error. Table 26 gives the chart for encoding the syndrome bits generated for the 64-bit configuration (as an example, if the syndrome bits SX/S1/S2/S4/S8/S16/S32 were 00100101, this would be decoded to indicate that there is a single-bit error at data bit 41). If no error is detected, the syndrome bits will all be zeroes.

In Detect Mode, the contents of the Data Input Latch are driven directly to the inputs of the Data Output Latch without corrections.

Correct Mode

In this mode, the EDC functions the same as in Detect Mode except that the correction network is allowed to correct (complement) any single-bit error of the Data Input Latch before putting it onto the inputs of the Data Output Latch. If multiple errors are detected, the output of the correction network is unspecified. If the single-bit error is a check bit, there is no automatic correction. Check bit correction can be done by placing the device in generate mode to produce a correct check bit sequence for the data in the Data Input Latch.

To perform the correction step, all four slices require access to the syndrome bits which are generated externally of the devices. This access is provided by reading the syndrome bits in through the CB inputs, where they are selected as inputs to the bit-in-error decoder by the multiplexer (see Block Diagram). The device connections for this are shown in Figure 10. When in Correct Mode, the SC outputs must be enabled so that the syndrome bits are available at the CB inputs.

Pass Thru Mode

In this mode, the unmodified contents of the Data Input Latch are placed on the inputs of the Data Output Latch, and the contents of Check Bit Input Latch are passed through the external XOR network and appear inverted at the XOR gate outputs labeled CX to C32 (see Figure 10).

Diagnostic Latch

The Diagnostic Latch serves both for diagnostic uses and internal control uses. It is loaded from the DATA lines under the control of LE DIAG. Table 27 shows the loading definitions for the DATA lines.

Diagnostic Generate/Diagnostic Detect/ Diagnostic Correct

These are special diagnostic modes selected by DIAG $MODE_{0-1}$ where either normal check bit inputs or outputs are substituted for by check bits from the Diagnostic Latch. See Table 2 for details.

Internal Control Mode

This mode is selected by CODE ID_{2-0}, input 001 (ID_2, ID_1, ID_0).

When in Internal Control Mode, the EDC takes the CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT and PASS THRU signals from the internal Diagnostic Latch rather than from the external control lines. Table 27 gives format for loading the Diagnostic Latch.

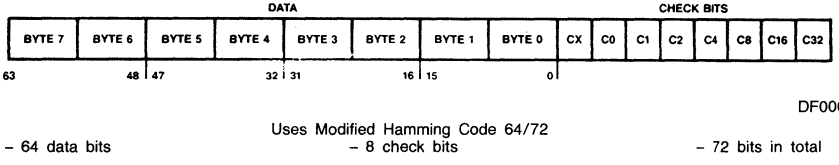
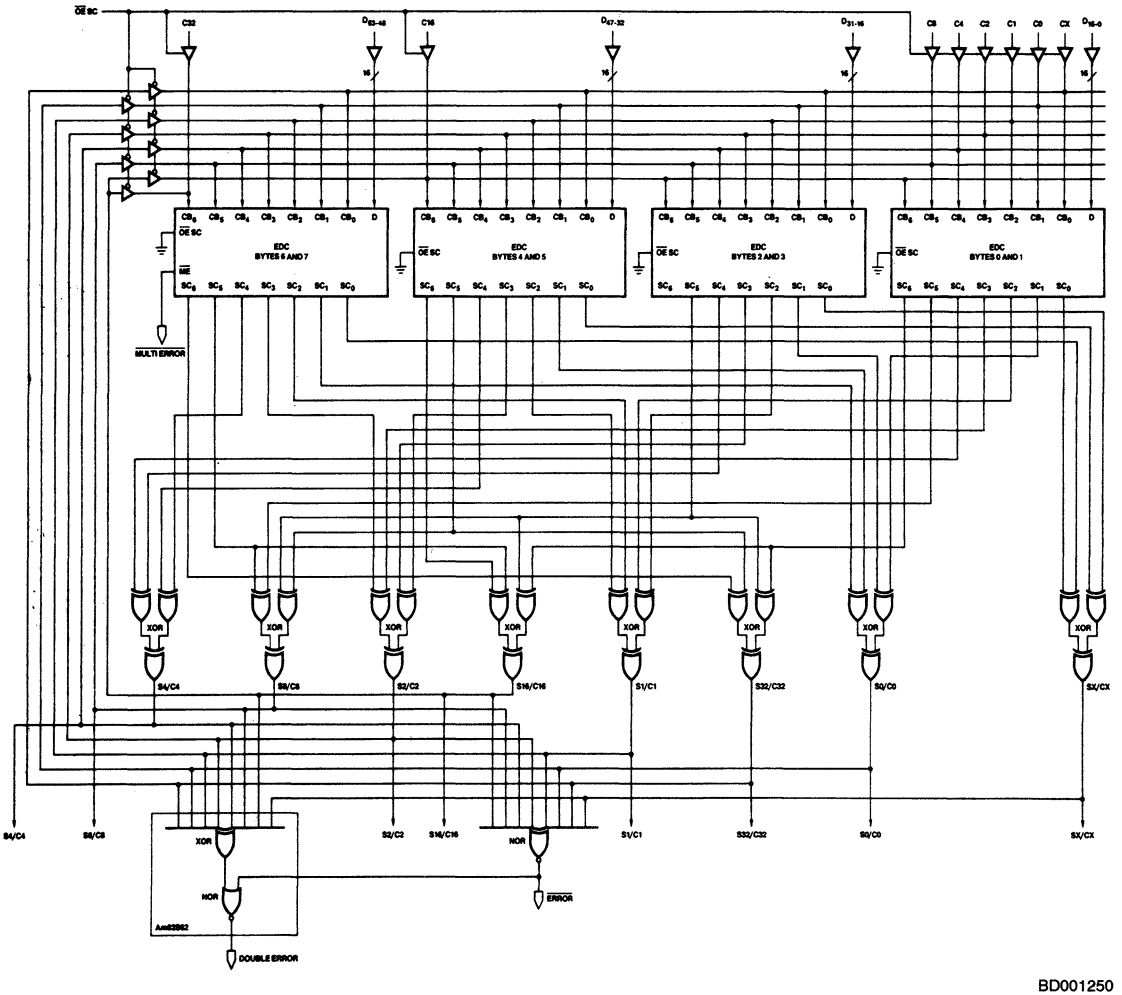


Figure 9. 64-Bit Data Format



Notes: 1. In Pass Thru Mode the Contents of the Check Latch Appear on the XOR Outputs Inverted.
2. In Diagnostic Generate Mode the Contents of the Diagnostic Latch Appear on the XOR Outputs Inverted.

Figure 10. 64-Bit Data Configuration

TABLE 24.
KEY AC CALCULATIONS FOR THE 64-BIT CONFIGURATION

64-Bit Propagation Delay		Component Delays from Am2960 AC Specifications, plus MSI
From	To	
DATA	Check Bits Out	(DATA to SC) + (XOR Delay)
DATA In	Corrected DATA Out	(DATA to SC) + (XOR Delay) + (Buffer Delay) + (CB to DATA, CODE ID 1xx)
DATA	Syndromes	(DATA to SC) + (XOR Delay)
DATA	$\overline{\text{ERROR}}$ for 64 Bits	(DATA to SC) + (XOR Delay) + (NOR Delay)
DATA	$\overline{\text{MULT ERROR}}$ for 64 Bits	(DATA to SC) + (XOR Delay) + (Buffer Delay) + (CB to $\overline{\text{MULT ERROR}}$, CODE ID 1xx)
DATA	DOUBLE ERROR for 64 Bits	(DATA to SC) + (XOR Delay) + (XOR/NOR Delay)

TABLE 25. 64-BIT MODIFIED HAMMING CODE - CHECK BIT ENCODE CHART

Generated Check Bits	Parity	Participating Data Bits															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								
C32	Even (XOR)	X	X	X	X	X	X	X	X								

Generated Check Bits	Parity	Participating Data Bits															
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X
C32	Even (XOR)									X	X	X	X	X	X	X	X

TABLE 25. (Cont'd.)
64-BIT MODIFIED HAMMING CODE – CHECK BIT ENCODE CHART

Generated Check Bits	Parity	Participating Data Bits															
		32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
CX	Even (XOR)	X				X		X	X			X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								
C32	Even (XOR)									X	X	X	X	X	X	X	X

Generated Check Bits	Parity	Participating Data Bits															
		48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
CX	Even (XOR)	X				X		X	X			X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X
C32	Even (XOR)	X	X	X	X	X	X	X	X								

The check bit is generated as either an XOR or XNOR of the 32 data bits noted by an "X" in the table.

TABLE 26. 64-BIT SYNDROME DECODE TO BIT-IN-ERROR

Syndrome Bits					S32	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
					S16	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
					S8	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
					S4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
SX	S0	S1	S2																		
0	0	0	0		*	C32	C16	T	C8	T	T	M	C4	T	T	M	T	46	62	T
0	0	0	1		C2	T	T	M	T	43	59	T	T	53	37	T	M	T	T	M
0	0	1	0		C1	T	T	M	T	41	57	T	T	51	35	T	15	T	T	31
0	0	1	1		T	M	M	T	13	T	T	29	23	T	T	7	T	M	M	T
0	1	0	0		C0	T	T	M	T	40	56	T	T	50	34	T	M	T	T	M
0	1	0	1		T	49	33	T	12	T	T	28	22	T	T	6	T	M	M	T
0	1	1	0		T	M	M	T	10	T	T	26	20	T	T	4	T	M	M	T
0	1	1	1		16	T	T	0	T	M	M	T	T	M	M	T	M	T	T	M
1	0	0	0		CX	T	T	M	T	M	M	T	T	M	M	T	14	T	T	30
1	0	0	1		T	M	M	T	11	T	T	27	21	T	T	5	T	M	M	T
1	0	1	0		T	M	M	T	9	T	T	25	19	T	T	3	T	47	63	T
1	0	1	1		M	T	T	M	T	45	61	T	T	55	39	T	M	T	T	M
1	1	0	0		T	M	M	T	8	T	T	24	18	T	T	2	T	M	M	T
1	1	0	1		17	T	T	1	T	44	60	T	T	54	38	T	M	T	T	M
1	1	1	0		M	T	T	M	T	42	58	T	T	52	36	T	M	T	T	M
1	1	1	1		T	48	32	T	M	T	T	M	M	T	T	M	T	M	M	T

* – no errors detected
Number – the number of the single bit-in-error
T – two errors detected
M – more than two errors detected

TABLE 27. 64-BIT DIAGNOSTIC LATCH LOADING FORMAT

Data Bit	Internal Function
0	Diagnostic Check Bit X
1	Diagnostic Check Bit 0
2	Diagnostic Check Bit 1
3	Diagnostic Check Bit 2
4	Diagnostic Check Bit 4
5	Diagnostic Check Bit 8
6, 7	Don't Care
8	Slice 0/1 – CODE ID 0
9	Slice 0/1 – CODE ID 1
10	Slice 0/1 – CODE ID 2
11	Slice 0/1 – DIAG MODE 0
12	Slice 0/1 – DIAG MODE 1
13	Slice 0/1 – CORRECT
14	Slice 0/1 – PASS THRU
15	Don't Care
16-23	Don't Care
24	Slice 2/3 – CODE ID 0
25	Slice 2/3 – CODE ID 1
26	Slice 2/3 – CODE ID 2
27	Slice 2/3 – DIAG MODE 0
28	Slice 2/3 – DIAG MODE 1
29	Slice 2/3 – CORRECT
30	Slice 2/3 – PASS THRU
31	Don't Care
32-37	Don't Care
38	Diagnostic Check Bit 16
39	Don't Care
40	Slice 4/5 – CODE ID 0
41	Slice 4/5 – CODE ID 1
42	Slice 4/5 – CODE ID 2
43	Slice 4/5 – DIAG MODE 0
44	Slice 4/5 – DIAG MODE 1
45	Slice 4/5 – CORRECT
46	Slice 4/5 – PASS THRU
47	Don't Care
48-54	Don't Care
55	Diagnostic Check Bit 32
56	Slice 6/7 – CODE ID 0
57	Slice 6/7 – CODE ID 1
58	Slice 6/7 – CODE ID 2
59	Slice 6/7 – DIAG MODE 0
60	Slice 6/7 – DIAG MODE 1
61	Slice 6/7 – CORRECT
62	Slice 6/7 – PASS THRU
63	Don't Care

APPLICATIONS

System Design Considerations

High Performance Parallel Operation

For maximum memory system performance, the EDC should be used in the Check-Only configuration shown in Figure 11. With this configuration, the memory system operates as fast with EDC as it would without.

On reads from memory, data is read out from the RAMs directly to the data bus (same as in a non-EDC system). At the same time, the data is read into the EDC to check for errors. If an error exists, the EDC's error flags are used to interrupt the CPU and/or to stretch the memory cycle. If no error is detected, no slowdown is required.

If an error is detected, the EDC generates corrected data for the processor. At the designer's option the correct data may be written back into memory; error logging and diagnostic routines may also be run under processor control.

The Check-Only configuration allows data reads to proceed as fast with EDC as without. Only if an error is detected is there any slowdown. But even if the memory system had an error every hour this would mean only one error every 3-4 *billion* memory cycles. So even with a very high error rate, EDC in a Check-Only configuration has essentially zero impact on memory system speed.

On writes to memory, check bits must be generated before the full memory word can be written into memory. But using the Am2961/62 Data Bus Buffers allows the data word to be buffered on the memory board while check bits are generated. This makes the check bit generate time transparent to the processor.

EDC in the Data Path

The simplest configuration for EDC is to have the EDC directly in the data path, as shown in Figure 12 (Correct-Always Configuration). In this configuration, data read from memory is always corrected prior to putting the data on the data bus. The advantages are simpler operation and no need for mid-cycle interrupts. The disadvantage is that memory system speed is slowed by the amount of time it takes for error correction on every cycle.

Usually the Correct-Always Configuration will be used with MOS microprocessors which have ample memory timing budgets. Most high performance processors will use the high performance parallel configuration shown in Figure 11 (Check-Only Configuration).

Scrubbing Avoids Double Errors

Single-bit errors are by far the most common in a memory system, and are always correctable by the EDC.

Double-bit memory errors are far less frequent than single-bit (50 to 1, or 100 to 1) and are always detected by the EDC, but not corrected.

In a memory system, soft errors occur only one at a time. A double-bit error in a data word occurs when a single soft error is left uncorrected and is followed by another error in the data word hours, days, or weeks after the first.

"Scrubbing" memory periodically avoids almost all double-bit errors. In the scrubbing operation, every data word in memory is periodically checked by the EDC for single-bit errors. If one is found, it is corrected and the data word written back into memory. Errors are not allowed to pile up, and most double-bit errors are avoided.

The scrubbing operation is generally done as a background routine when the memory is not being used by the processor. If memory is scrubbed frequently, errors that are detected and corrected during processor accesses need not be immediately written back into memory. Instead, the error will be corrected in memory during scrubbing. This reduces the time delay involved in a processor access of an incorrect memory word.

Correction of Double-Bit Errors

In some cases, double-bit memory errors can be corrected! This is possible when one of the two bit errors is a hard error.

When a double bit error is detected, the data word should be checked to determine if one of the errors is a hard error. If so, the hard error bit may be corrected by inverting it, leaving only a single, correctable error. The time for this operation is negligible, since it will occur infrequently.

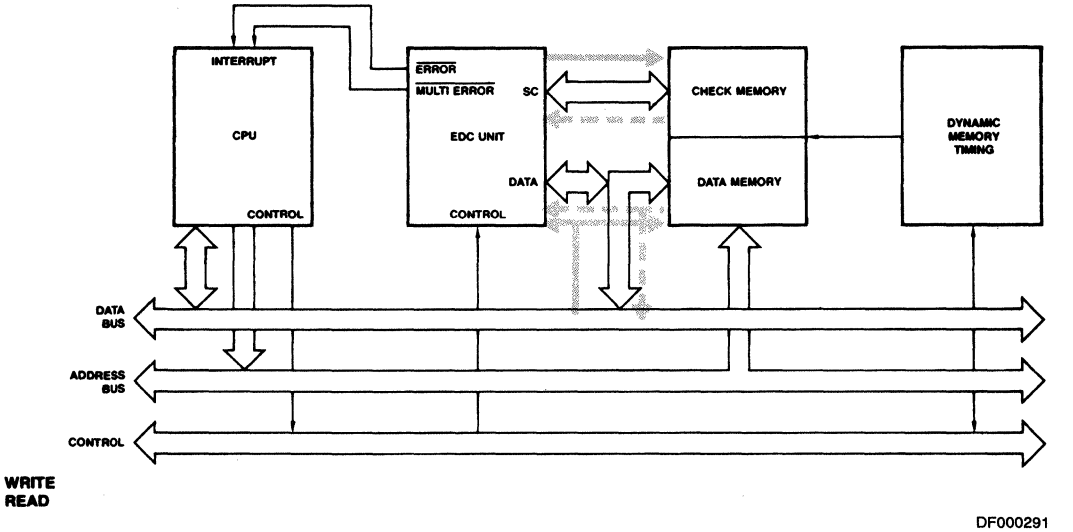


Figure 11. Check-Only Configuration

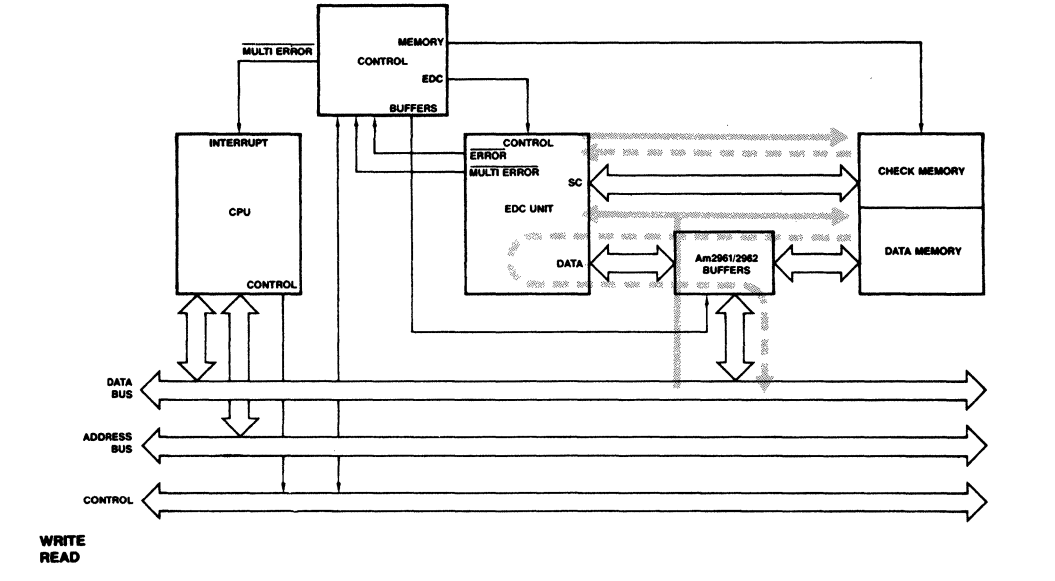


Figure 12. Correct-Always Configuration

The procedure after detection of a double error is as follows:

- Invert the data bits read from memory.
- Write the inverted data back into the same memory word.
- Re-read the memory location and XOR the newly read out value with the old. If there is no hard error, then the XOR result will be all 1's. If there is a hard error, it will have the same bit value regardless of what was written in. So it will show as a 0 after the XOR operation.
- Invert the hard error bit (this will "correct" it) leaving only one error in the data.
- The EDC can then correct the single bit error.
- Rewrite the correct data word into memory. This does not change the hard error, but does eliminate the soft error. So the next memory access will find only a single-bit, correctable error.

Example of Double-Bit Error Correction When One is a Hard Error

	16 data bits	6 check bits
1. Data Read from Memory (D_1)	1111111100000011	011010
2. EDC detects a multiple error. Syndromes:		011000
3. Syndrome decode indicates a double-bit error.		
4. Invert the bits read from memory (D_1)	0000000011111100	100101
5. Write D_1 back to the same memory location		
6. Read back the memory location (D_2)	0000000011111101	100101
7. XOR D_1 and D_2	1111111111111110	111111
8. So the last data bit is the hard error. Use this to modify D_1	1111111100000010	011010
9. Pass the modified D_1 through the EDC. The EDC detects a single bit correctable error and outputs corrected data:	1111111100000000	011010
10. Write the corrected data back to memory to fix the soft error.		

Error Logging and Preventative Maintenance

The effectiveness of preventative maintenance can be increased by logging information on errors detected by the EDC. This is called error logging.

The EDC provides syndromes when errors are detected. The syndromes indicate which bit is in error. In most memory systems, each individual RAM supplies only one bit of the memory word. So the syndrome and data word address specify which RAM was in error.

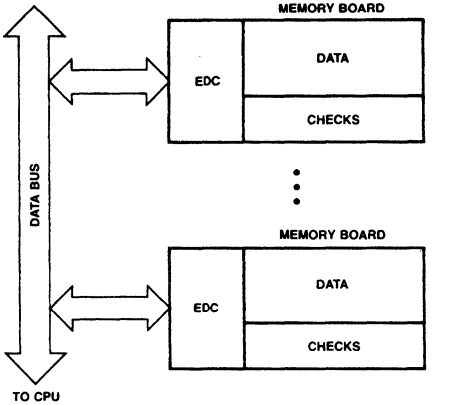
Typically a permanent/hard RAM failure is preceded by a period of time where the RAM displays an increasing frequency of intermittent, soft errors. Error logging statistic can be used to detect an increasing intermittent error frequency so that the RAM can be replaced before a permanent failure occurs.

Error logging also records the location of already hard failed RAMs. With EDC a hard failure will not halt system operation. EDC can always correct single-bit errors even if it is a hard error. EDC can also correct double-bit errors where one is hard and one soft (see "Correction of Double Bit Errors" Section). The ability to continue operation despite hard errors can greatly reduce the need for emergency field maintenance. The hard-failed RAMs can be instead replaced at low cost during a regularly-scheduled preventative maintenance session.

Reducing Check-Bit Overhead

Memory word widths need not be the same as the data word width of the processor. There is a substantial reduction in check bit overhead if wider memory words are used:

Memory Word		Check
# Data Bits	# Check Bits	Bit Overhead
8	5	38%
16	6	27%
32	7	14%
64	8	11%



DF000160

Figure 13. EDC Per Board

This reduction in check bit overhead lowers cost and increases the amount of data that can be packed on to each board.

The trade-off is that when writing data pieces into memory that are narrower than the memory word width, more steps are

required. These steps are exactly the same as those described in Byte Write in the Applications section. No penalty exists for reads from memory.

EDC Per Board vs. EDC Per System

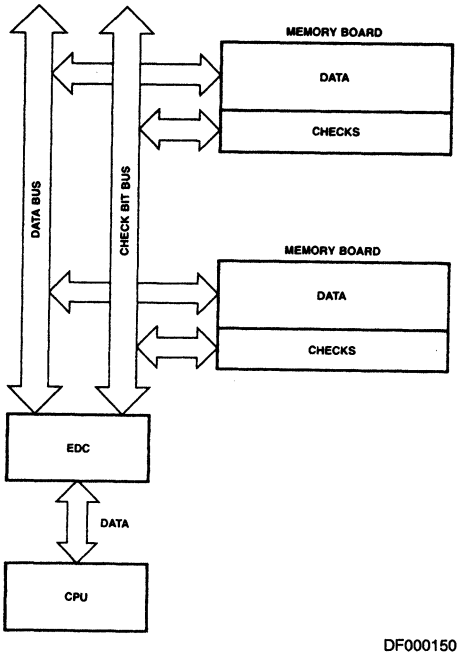
The choice of an EDC per system or per board depends on the economics and the architecture of the system.

Certainly the cheaper approach is to have only one EDC per system and this is a viable solution if only one memory location is accessed at a time.

This solution does require that the system has both data and check bit lines (see Figure 14). This makes retrofitting a system difficult and creates complications if static or ROM memory, which do not require check bits, are mixed in with dynamic RAM.

If the system has an advanced architecture, it is quite likely that it is necessary to simultaneously access memory locations on different memory boards (see Figure 13). Architectural features that require this are interleaved memory, cache memory, and DMA that is done simultaneously with processor memory accesses. EDC per board is a simpler system from a design standpoint.

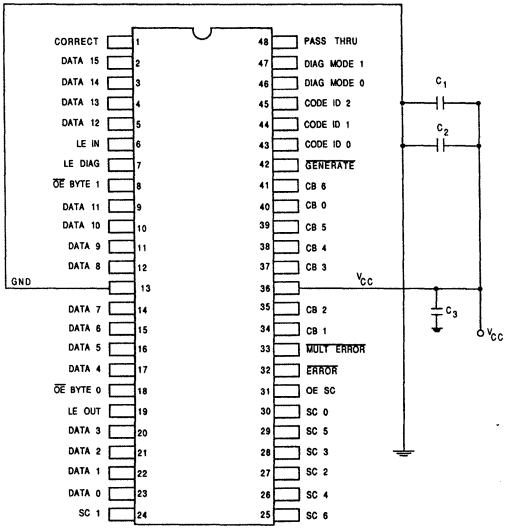
The EDC is designed to work efficiently in either the per system or per board configurations.



DF000150

Figure 14. EDC Per System

DEVICE DECOUPLING
V_{CC} AND GROUND PIN CONNECTIONS



TC003940

Note: 1. $C_1 = 1.0\ \mu F$, $C_2 = C_3 = 0.01\ \mu F$
The C_1, C_2, and C_3 capacitors should be used to shunt Low – and high-frequency noise from V_{CC}. Do not replace with one capacitor. Place capacitors as close to the device as possible.

DESIGNING FOR A FASTER SYSTEM ENVIRONMENT

Am2960A Designs

When using the Am2960A in a high performance memory system, certain MSI glue logic will not be fast enough. The following device conversions are therefore recommended:

Am2960/Am2960-1 Design		Am2960A Design/Upgrade	
EDC Bus Buffer:	Am2961/Am2962	EDC Bus Buffer:	Am29845/Am29846
Cascaded Design, CB_{0-6} Driver:	Am2958	Cascaded Design, CB_{0-6} Driver:	Am29827/Am29828

Am2960/Am2960-1 to Am2960A Design Upgrades

When upgrading a system from the Am2960/Am2960-1 to the Am2960A special care must be taken to ensure that MSI glue logic specifications are not violated. For example, if latches are used in the data path, set up and hold times of the latches must also be upgraded to match the performance of the Am2960A. This may require replacement of currently designed-in latches to faster versions or alternate solutions.

A set up and hold time specification of 2.5 ns or less is recommended for these latches.

Byte Write

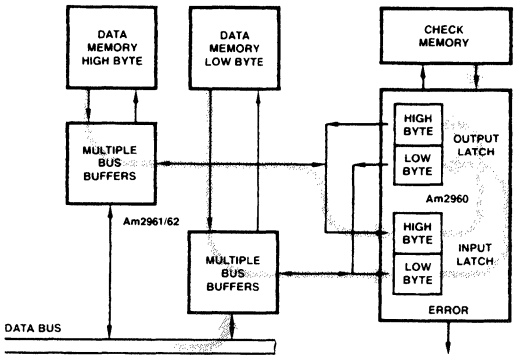
Byte operations are increasingly common for 16-bit and 32-bit processors. These complicate memory operations because check bits are generated for a complete 16-bit or 32-bit or 64-bit memory word – not for a single byte.

To write a byte into memory with EDC requires the following steps:

- Latch the byte into the Am2961/62 bus buffers (Figure 15)
- Read the complete data word from memory (Figure 15)
- Correct the complete data word if necessary (Figure 15)
- Insert the byte to be written into the data word (Figure 16)
- Generate new check bits for the entire data word (Figure 16)
- Store the data word back into memory (Figure 16)

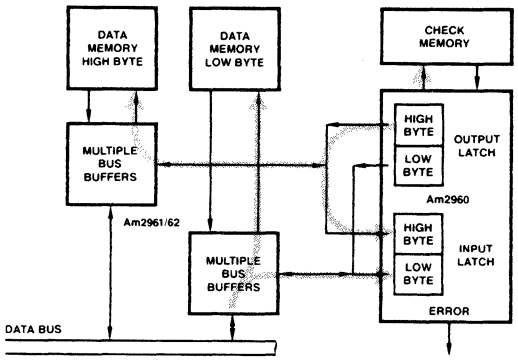
(In fact, these steps must be taken for any piece of data being written into memory that is not as wide as a full memory word.)

The Am2960 EDC is designed with the intent of keeping byte operations simple in EDC systems. The EDC has separate output enables for each byte in the Data Output Latch. As shown in Figures 15 and 16, this allows the data word to be read from memory, the new byte to be inserted among the old, and new check bits to be generated using less time and less hardware than if separate byte enables were not available.



AF000730

Figure 15. Byte Write, Phase 1: Read Out the Old Word and Correct



AF000730

Figure 16. Byte Write, Phase 2: Insert the New Byte, Generate Checks and Write Into Memory

Diagnostics

EDC is used to boost the reliability of the overall system. It is necessary to also be able to check the operation of the EDC itself. For this reason the EDC has an internal control mode, a diagnostic latch, and two diagnostic modes.

To check that the EDC is functioning properly, the processor can put the EDC under software control by setting CODE ID_{2-0} to 001. This puts the EDC into Internal Control Mode. In Internal Control Mode the EDC is controlled by the contents of the Diagnostic Latch which is loaded from the DATA inputs under processor control.

The EDC is set into CORRECT Mode. The processor loads in a known set of check bits into the Diagnostic Latch, a known set of data bits into the Data In Latch, and forces data errors. The output of the EDC (syndromes, error flags, corrected data) is then compared against the expected responses. By exercising the EDC with a string of data/check combinations and comparing the output against the expected responses, the EDC can be fully checked out.

Eight Bit Data Word

Eight bit MOS microprocessors can use EDC too. Only five check bits are required. The EDC configuration for eight bits is shown in Figure 17. It operates as does the normal 16-bit configuration with the upper byte fixed at 0.

Check bit overhead for 8-bit data words can be reduced two ways. See the sections "Single Error Correction Only" and "Reducing Check Bit Overhead."

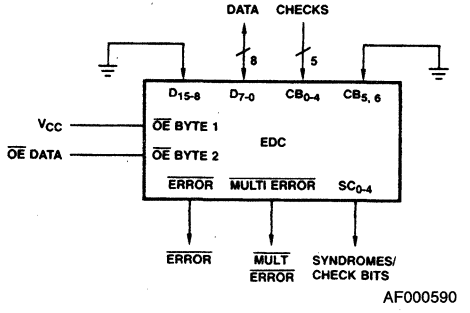


Figure 17. 8-Bit Configuration

Other Word Widths

EDC on data words other than 8, 16, 32, of 64 bits can be accomplished with the AM2960. In most cases the extra data bits can be forced to a constant, and EDC will proceed as normal. For example, a 24-bit data word is shown in Figure 18.

Single Error Correction Only

The EDC normally corrects all single-bit errors *and* detects all double-bit and some triple-bit errors. To save one check bit per word, the ability to detect double bit errors can be sacrificed – single errors are still detected and corrected.

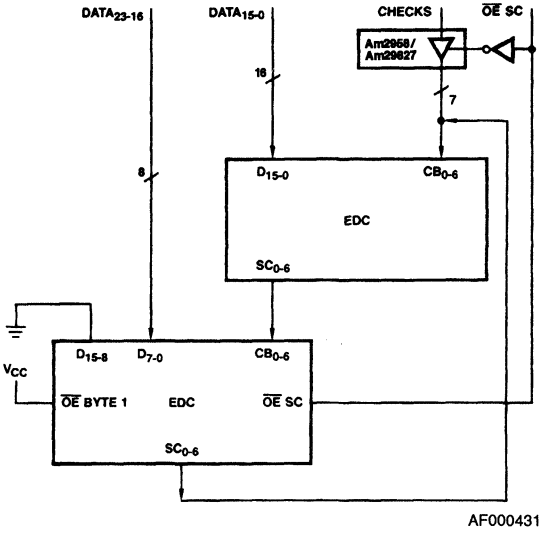


Figure 18. 24-Bit Configuration

Data Bits	Check Bits Required	
	Single Error Correction Only	Single Error Correct & Double Error Detect
8	4	5
16	5	6
32	6	7
64	7	8

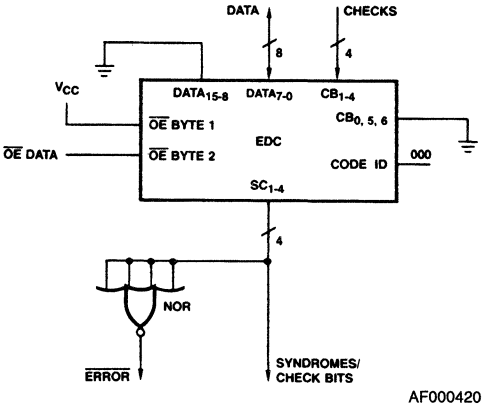


Figure 19. 8-Bit Single Error Correction Only

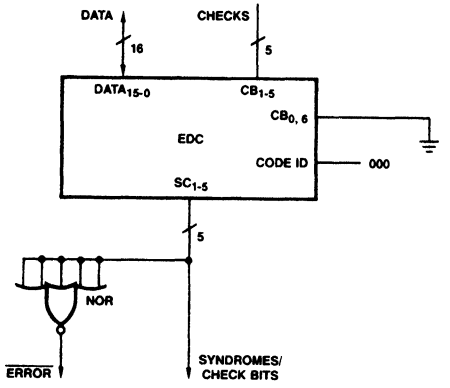
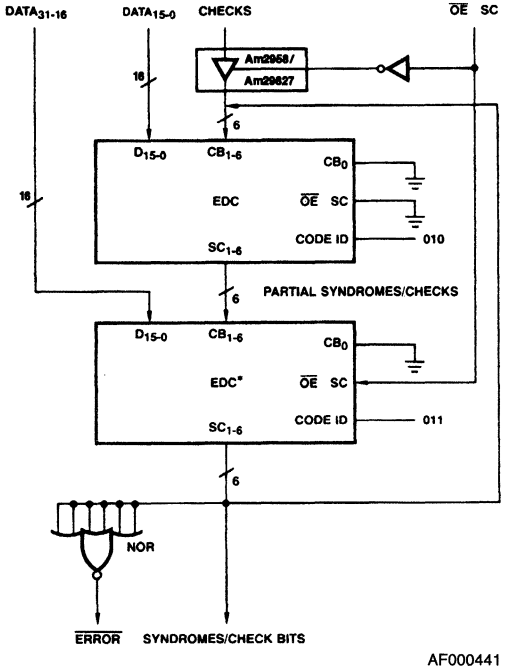


Figure 20. 16-Bit Single Error Correction Only



* The Code ID Combination for this Slice Forces the Check Bit Latch Transparent.

Figure 21. 32-Bit Single Correction Only

Figures 19, 20, 21, 22 show single error correction only configurations for 8, 16, 32, and 64-bit data words respectively.

Check Bit Correction

The EDC detects single bit errors whether the error is a data bit or a check bit. Data bit errors are automatically corrected by the EDC. To generate corrected check bits once a single check bit error is detected, the EDC need only be switched to GENERATE mode (data in the DATA INPUT LATCH is valid).

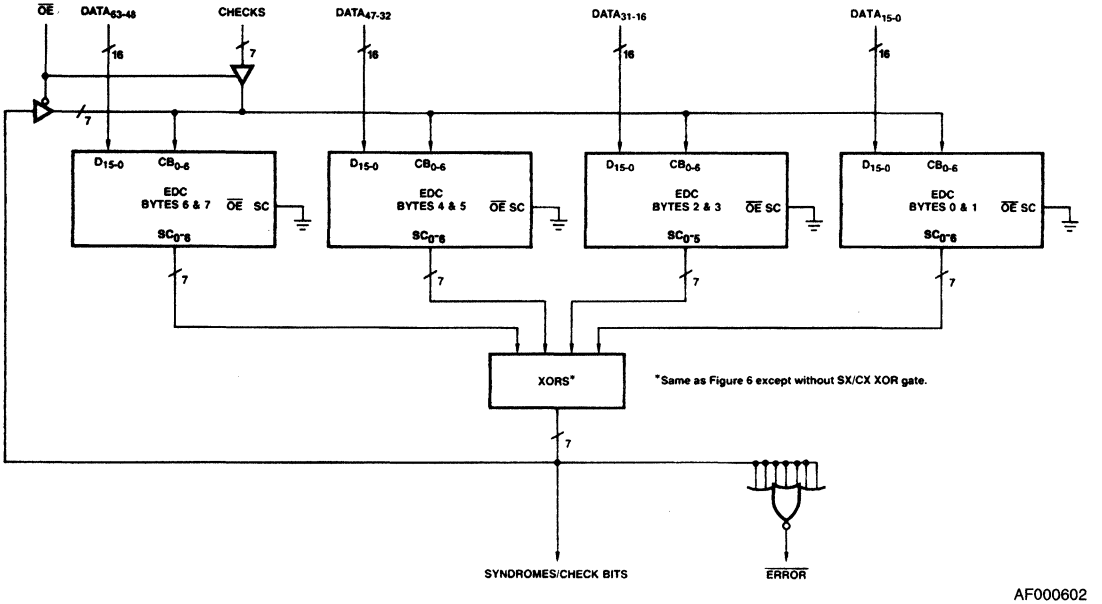
The syndromes generated by the EDC may be decoded to determine whether the single bit error is a check bit.

In many memory systems, a check bit error will be ignored on the memory read and corrected during a periodic "scrubbing" of memory (see section in System Design Considerations).

Multiple Errors

The bit-in-error decode logic uses syndrome bits S0 through S32 to correct errors, SX is only used in developing the multiple error signal. This means that some multiple errors will cause a data bit to be inverted.

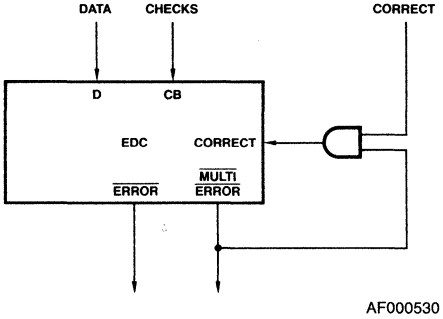
For example, in the 16-bit mode if data bits 8 and 13 are in error the syndrome 111100 (SX, S0, S1, S2, S4, S8) is produced. This is flagged a double error by the error detection logic, but the bit-in-error decoder only receives syndrome 11100 (S0, S1, S2, S4, S8), which it decodes as a single error in data bit 0 and inverts that bit. If it is desired to inhibit this inversion, the multiple error output may be connected to the correct input as in Figure 23. This will inhibit correction when a multiple error occurs. Extra time delay may be introduced in the data to correct data path when this is done.



AF000602

Notes: 1. In Pass Thru Mode the Contents of the Check Latch Appear on the XOR Outputs Inverted.
2. In Diagnostic Generate Mode the Contents of the Diagnostic Latch appear on the XOR Outputs Inverted.

Figure 22. 64-Bit Single Error Correction Only



AF000530

Figure 23. Inhibition of Data Modification

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Temperature (Case)
Under Bias ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continous ..... −0.5 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... −0.5 V to V_{CC} Max.
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Ambient Temperature (T_A) ..... 0°C to +70°C
Supply Voltage ..... +4.75 V to +5.25 V

Military (M) Devices
Case Temperature (T_C) ..... −55°C to +125°C
Supply Voltage ..... +4.5 V to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Descriptions	Test Conditions (Note 1)			Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OH} = −0.8 mA	COM'L	2.4		V
				MIL	2.4		
V_{OL}	Output LOW Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 8 mA			0.5	V
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for all Inputs (Note 5)			2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for all Inputs (Note 5)				0.8	V
V_I	Input Clamp Voltage	V_{CC} = Min., I_{IN} = −18 mA				−1.5	V
I_{IL}	Input LOW Current	V_{CC} = Max. V_{IN} = 0.5 V	$DATA_{0-15}$ (Note 3)			−410	μA
			All Other Inputs			−360	
I_{IH}	Input HIGH Current	V_{CC} = Max. V_{IN} = 2.7V	$DATA_{0-15}$ (Note 3)			70	μA
			All Other Inputs			50	
I_I	Input HIGH Current	V_{CC} = Max., V_{IN} = 5.5 V				1.0	mA
I_{OZH} I_{OZL}	Off State (High Impedance) Output Current	V_{CC} Max.	$DATA_{0-15}$	V_O = 2.4		70	μA
				V_O = 0.5		−410	
			SC_{0-6}	V_O = 2.4		50	
				V_O = 0.5		−50	
I_{OS}	Output Short-Circuit Current (Note 2)	V_{CC} = V_{CC} Max. + 0.5 V, V_O = 0.5 V		Am2960/2960-1	−25	−85	mA
				Am2960A	−20		
I_{CC}	Power Supply Current (Note 4)	V_{CC} = Max.	C Devices	Am2960/2960-1		400	mA
				Am2960A		365	
			M Devices	Am2960		400	mA
				Am2960A		400	
		V_{CC} = 5.0 V	T_A = 25°C	Am2960/2960-1		290	mA
				Am2960A		275	

Notes: 1. For conditions shown as Min. or Max., use the appropriate value specified under Operating Range for the applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.
3. These are three-state outputs internally connected to TTL inputs. Input Characteristics are measured with outputs disabled (High Impedance).
4. Worst case I_{CC} is at minimum temperature. Typical I_{CC} (V_{CC} = 5.0 V, T_A = 25°C) represents nominal units and is not tested.
5. These input levels provide zero noise immunity and should only be tested in a static, noise-free environment.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Notes 1 and 2)

The following table specifies the guaranteed device performance over the commercial operating range of 0°C to +70°C (ambient), with V_{CC} 4.75 to 5.25 V. All inputs switching is between 0 V and 3 V at 1 V/ns and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

		Data Path Description		Am2960		Am2960-1		Am2960A	
No.	Param.	From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
1	t_{PD}	$DATA_{0-15}$ (Note 3)	SC_{0-6}		32		28		25
			$DATA_{0-15}$		65		52		30
			$\overline{ERROR}$		32		25		18
			$\overline{MULT\ ERROR}$		50		50		23
2	t_{PD}	CB_{0-6} (CODE ID_{2-0} 000, 011)	SC_{0-6}		28		23		18
			$DATA_{0-15}$		56		50		25
			$\overline{ERROR}$		29		23		15
			$\overline{MULT\ ERROR}$		47		47		20
3	t_{PD}	CB_{0-6} (CODE ID_{2-0} 010, 100, 101, 110, 111)	SC_{0-6}		28		28		18
			$DATA_{0-15}$		45		34		23
			$\overline{ERROR}$		29		29		15
			$\overline{MULT\ ERROR}$		34		34		20
4	t_{PD}	$\overline{GENERATE}$	SC_{0-6}		35		35		27
			$DATA_{0-15}$		63		63		33
			$\overline{ERROR}$		36		36		18
			$\overline{MULT\ ERROR}$		55		55		23
5	t_{PD}	CORRECT (Not Internal Control Mode)	SC_{0-6}		–		–		–
			$DATA_{0-15}$		45		45		24
			$\overline{ERROR}$		–		–		–
			$\overline{MULT\ ERROR}$		–		–		–
6	t_{PD}	DIAG MODE (Not Internal Control Mode)	SC_{0-6}		50		50		26
			$DATA_{0-15}$		78		78		31
			$\overline{ERROR}$		59		59		17
			$\overline{MULT\ ERROR}$		75		75		23
7	t_{PD}	PASS THRU (Not Internal Control Mode)	SC_{0-6}		36		36		19
			$DATA_{0-15}$		44		44		23
			$\overline{ERROR}$		29		29		15
			$\overline{MULT\ ERROR}$		46		46		18
8	t_{PD}	CODE ID_{2-0}	SC_{0-6}		61		61		29
			$DATA_{0-15}$		90		90		34
			$\overline{ERROR}$		60		60		24
			$\overline{MULT\ ERROR}$		80		80		27

Notes: See notes following end of tables continued on next page.

SWITCHING CHARACTERISTICS over **COMMERCIAL** operating range (Notes 1 and 2) (Cont'd.)

No.	Param.	Data Path Description		Am2960		Am2960-1		Am2960A	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
9	t_{PD}	LE IN (From latched to transparent)	SC_{0-6}		39		39		21
			$DATA_{0-15}$		72		72		27
			$\overline{ERROR}$		39		39		18
			$\overline{MULT\ ERROR}$		59		59		21
10	t_{PD}	LE OUT (From latched to transparent)	SC_{0-6}		–		–		–
			$DATA_{0-15}$		31		31		19
			$\overline{ERROR}$		–		–		–
			$\overline{MULT\ ERROR}$		–		–		–
11	t_{PD}	LE DIAG (From latched to transparent; Not Internal Control Mode)	SC_{0-6}		45		45		20
			$DATA_{0-15}$		78		78		27
			$\overline{ERROR}$		45		45		18
			$\overline{MULT\ ERROR}$		65		65		21
12	t_{PD}	Internal Control Mode: LE DIAG (From latched to transparent)	SC_{0-6}		67		67		29
			$DATA_{0-15}$		96		96		30
			$\overline{ERROR}$		66		66		24
			$\overline{MULT\ ERROR}$		86		86		27
13	t_{PD}	Internal Control Mode: $DATA_{0-15}$ (Via Diagnostic latch)	SC_{0-6}		67		67		28
			$DATA_{0-15}$		96		96		35
			$\overline{ERROR}$		66		66		24
			$\overline{MULT\ ERROR}$		86		86		27
14	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE IN	6		6		3	
15	t_{HOLD}			7		7		5	
16	t_{SET}	CB_{0-6} (Notes 4, 5)		5		5		3	
17	t_{HOLD}			6		6		4	
18	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE OUT	44		34		20	
19	t_{HOLD}			5		5		4	
20	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 000, 011)		35		35		15	
21	t_{HOLD}			0		0		0	
22	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 010, 100, 101, 110, 111)		27		27		15	
23	t_{HOLD}			0		0		0	
24	t_{SET}	$\overline{GENERATE}$ (Notes 4, 5)		42		42		20	
25	t_{HOLD}			0		0		0	
26	t_{SET}	CORRECT (Notes 4, 5)		26		26		15	
27	t_{HOLD}			1		1		1	
28	t_{SET}	DIAG MODE (Notes 4, 5)		69		69		20	
29	t_{HOLD}			0		0		0	
30	t_{SET}	PASS THRU (Notes 4, 5)		26		26		7	
31	t_{HOLD}			0		0		0	
32	t_{SET}	$CODE\ ID_{2-0}$ (Notes 4, 5)		81		81		22	
33	t_{HOLD}			0		0		0	

Notes: See notes following table continued on next page.

SWITCHING CHARACTERISTICS over **COMMERCIAL** operating range (Notes 1 and 2) (Cont'd.)

No.	Param.	Data Path Description		Am2960		Am2960-1		Am2960A	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
34	t_{SET}	LE IN (Notes 4, 5)	LE OUT	51		51		24	
35	t_{HOLD}			5		5		4	
36	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE DIAG	6		6		3	
37	t_{HOLD}			8		8		6	
38	t_{EN}	$\overline{OE}$ BYTE 0,1 ENABLE (Note 6)	$DATA_{0-15}$		30		30		14
39	t_{DIS}				30		30		23
40	t_{EN}	$\overline{OE}$ SC DISABLE (Note 6)	SC_{0-6}		30		30		7
41	t_{DIS}				30		30		21
42	t_{PW}	MINIMUM PULSE WIDTH: LE IN, LE OUT, LE DIAG		15		15		12	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations.
3. Data In or LE In to Correct Data Out measurement requires timing as shown in Figure 26.
4. Set-up and Hold times relative to Latch Enables (Latching up data).
5. Set-up and Hold times are not tested, but are guaranteed by characterization.
6. Output disable tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.

SWITCHING CHARACTERISTICS over **MILITARY** operating range (for APL Products, Group A, Subgroups 7, 8, 9, 10, 11 are tested unless otherwise noted) (Notes 1 and 2)

The following table specifies the guaranteed device performance over the Military operating range of −55°C to + 125°C (case), with V_{CC} 4.5 to 5.5 V. All inputs switching is between 0 V and 3 V at 1 V/ns and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

No.	Param.	Data Path Description		Am2960		Am2960A	
		From Input	To Output	Min.	Max.	Min.	Max.
1	t_{PD}	$DATA_{0-15}$ (Note 3)	SC_{0-6}		35		25
			$DATA_{0-15}$		73		30
			$\overline{ERROR}$		36		20
			$\overline{MULT\ ERROR}$		56		23
2	t_{PD}	CB_{0-6} (CODE ID_{2-0} 000, 011)	SC_{0-6}		30		20
			$DATA_{0-15}$		61		28
			$\overline{ERROR}$		31		16
			$\overline{MULT\ ERROR}$		50		20
3	t_{PD}	CB_{0-6} (CODE ID_{2-0} 010, 100, 101, 110, 111)	SC_{0-6}		30		22
			$DATA_{0-15}$		50		25
			$\overline{ERROR}$		31		16
			$\overline{MULT\ ERROR}$		37		20
4	t_{PD}	$\overline{GENERATE}$	SC_{0-6}		38		27
			$DATA_{0-15}$		69		33
			$\overline{ERROR}$		41		19
			$\overline{MULT\ ERROR}$		62		24
5	t_{PD}	CORRECT (Not Internal Control Mode)	SC_{0-6}		–		–
			$DATA_{0-15}$		49		25
			$\overline{ERROR}$		–		–
			$\overline{MULT\ ERROR}$		–		–
6	t_{PD}	DIAG MODE (Not Internal Control Mode)	SC_{0-6}		58		26
			$DATA_{0-15}$		89		32
			$\overline{ERROR}$		65		20
			$\overline{MULT\ ERROR}$		90		24
7	t_{PD}	PASS THRU (Not Internal Control Mode)	SC_{0-6}		39		22
			$DATA_{0-15}$		51		25
			$\overline{ERROR}$		34		16
			$\overline{MULT\ ERROR}$		54		20
8	t_{PD}	CODE ID_{2-0}	SC_{0-6}		69		31
			$DATA_{0-15}$		100		35
			$\overline{ERROR}$		68		26
			$\overline{MULT\ ERROR}$		90		29

Notes: See notes following tables continued on next pages.

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

No.	Param.	Data Path Description		Am2960		Am2960A	
		From Input	To Output	Min.	Max.	Min.	Max.
9	t_{PD}	LE IN (From latched to transparent)	SC_{0-6}		44		24
			$DATA_{0-15}$		82		27
			$\overline{ERROR}$		43		19
			$\overline{MULT\ ERROR}$		66		23
10	t_{PD}	LE OUT (From latched to transparent)	SC_{0-6}		–		–
			$DATA_{0-15}$		33		19
			$\overline{ERROR}$		–		–
			$\overline{MULT\ ERROR}$		–		–
11	t_{PD}	LE DIAG (From latched to transparent; Not Internal Control Mode)	SC_{0-6}		50		27
			$DATA_{0-15}$		88		27
			$\overline{ERROR}$		49		19
			$\overline{MULT\ ERROR}$		72		23
12	t_{PD}	Internal Control Mode: LE DIAG (From latched to transparent)	SC_{0-6}		75		31
			$DATA_{0-15}$		106		35
			$\overline{ERROR}$		74		26
			$\overline{MULT\ ERROR}$		96		29
13	t_{PD}	Internal Control Mode: $DATA_{0-15}$ (Via Diagnostic latch)	SC_{0-6}		75		32
			$DATA_{0-15}$		106		35
			$\overline{ERROR}$		74		24
			$\overline{MULT\ ERROR}$		96		28
14	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE IN	7		3	
†15	t_{HOLD}			7		5	
16	t_{SET}	CB_{0-6} (Notes 4, 5)		5		3	
†17	t_{HOLD}			7		4	
18	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE OUT	50		20	
†19	t_{HOLD}			5		4	
20	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 000, 011)		38		15	
†21	t_{HOLD}			0		0	
22	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 010, 100, 101, 110, 111)		30		15	
†23	t_{HOLD}			0		0	
24	t_{SET}	$\overline{GENERATE}$ (Notes 4, 5)		46		20	
†25	t_{HOLD}			0		0	
26	t_{SET}	CORRECT (Notes 4, 5)		28		15	
†27	t_{HOLD}			1		1	
28	t_{SET}	DIAG MODE (Notes 4, 5)		84		20	
†29	t_{HOLD}			0		0	
30	t_{SET}	PASS THRU (Notes 4, 5)		30		7	
†31	t_{HOLD}			0		0	
32	t_{SET}	$CODE\ ID_{2-0}$ (Notes 4, 5)		89		22	
†33	t_{HOLD}			0		0	

Notes: See notes continued on next page.

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

No.	Param.	Data Path Description		Am2960		Am2960A	
		From Input	To Output	Min.	Max.	Min.	Max.
34	t_{SET}	LE IN (Notes 4, 5)	LE OUT	59		24	
†35	t_{HOLD}			5		4	
36	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE DIAG	7		3	
†37	t_{HOLD}			9		6	
38	t_{EN}	$\overline{OE}$ BYTE 0,1 ENABLE (Note 6)	$DATA_{0-15}$		35		14
39	t_{DIS}				35		23
40	t_{EN}	$\overline{OE}$ SC DISABLE (Note 6)	SC_{0-6}		35		7
41	t_{DIS}				35		21
42	t_{PW}	MINIMUM PULSE WIDTH: LE IN, LE OUT, LE DIAG		15		15	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations.
3. Data In or LE In to Correct Data Out measurement requires timing as shown in Figure 26.
4. Set-up and Hold times relative to Latch Enables (Latching up data).
5. Setup and Hold times are not tested, but are guaranteed by characterization.
6. Output disable tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.

† = Not Included in Group A Tests

SWITCHING TEST CIRCUITS

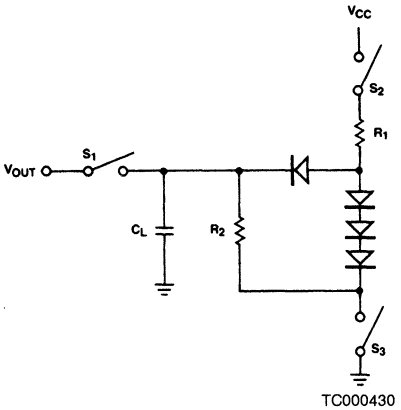


Figure 24. Three-State Outputs

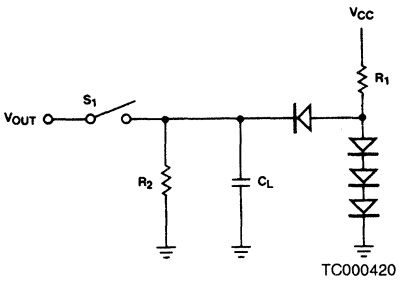


Figure 25. Normal Outputs

Notes: 1. C_L = 50pF includes scope probe, wiring and stray capacitances without device in test fixture.
2. S_1, S_2, S_3 are closed during function test and all A.C. tests, except output enable tests.
3. S_1 and S_3 are closed while S_2 is open for t_{PZH} test.
S_1 and S_2 are closed while S_3 is open for t_{PZL} test.
4. R_2 = 1K for three-state output.
R_2 is determined by the I_{OH} at V_{OH} = 2.4V for non-three-state outputs.
5. R_1 is determined by I_{OL} (MIL) with V_{CC} = 5.0V minus the current to ground through R_2.
6. C_L = 5.0 pF for output disable tests.

TEST OUTPUT LOADS

Pin #	Pin Label	Test Circuit	R_1	R_2
–	D_0-D_{15}	Fig. 24	430Ω	1KΩ
24-30	SC_0-SC_6	Fig. 24	430Ω	1KΩ
32	$\overline{\text{ERROR}}$	Fig. 25	470Ω	3kΩ
33	$\overline{\text{MULTERROR}}$	Fig. 25	470Ω	3KΩ

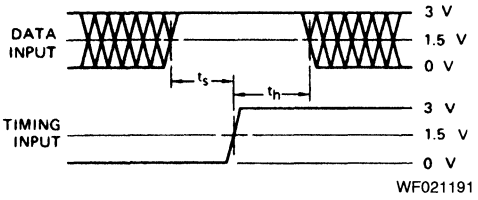
Notes on Testing

Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Insure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400mA in 5-8ns. Inductance in the ground cable may allow the ground pin at the device to rise by 100's of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 3$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. Changing the CODE ID inputs can cause loss of data in some of the Am2960 internal latches. Specifically, the entire checkbit latch and bits 6 and 7 of the diagnostic latch are indeterminate after a change in CODE ID inputs.

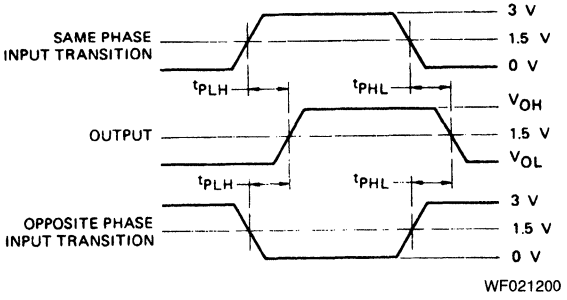
Logic simulations should store "x" (i.e., "don't care") in these bits after CODE ID change. Test programs should reload these registers before they are used.

SWITCHING TEST WAVEFORMS

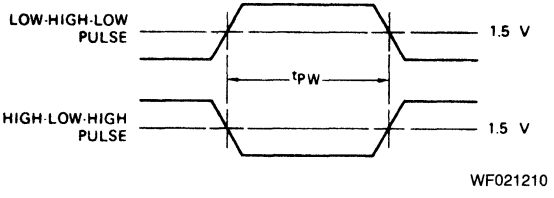


Notes: 1. Diagram show for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched are is don't care condition.

Setup and Hold Times



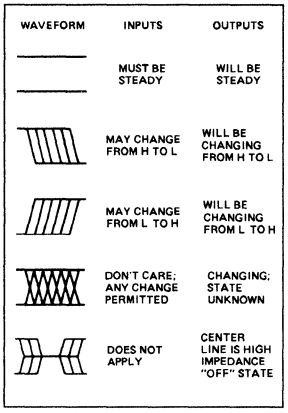
Propagation Delay



Pulse Width

SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS



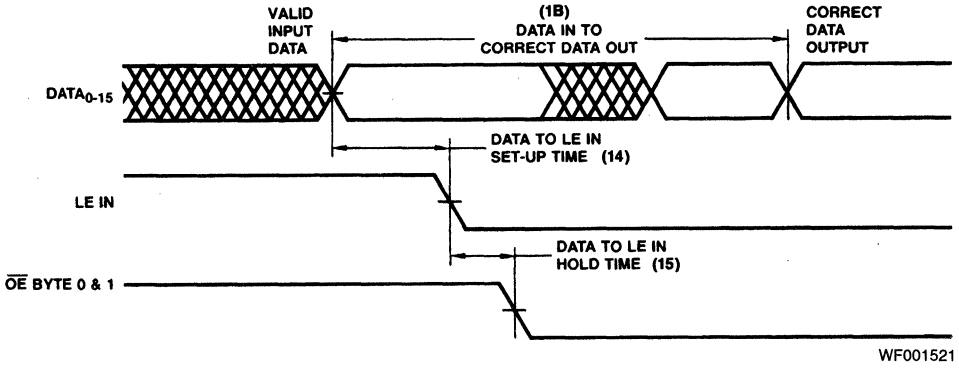
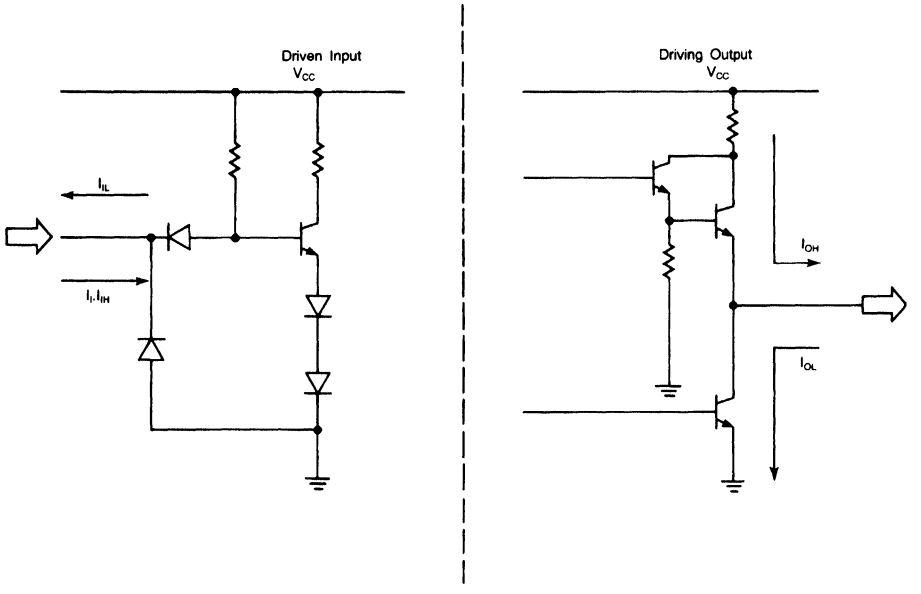


Figure 26. DATA IN/LE IN to Correct DATA OUT

INPUT/OUTPUT CIRCUIT DIAGRAMS



IC000883

Final

Am2964B

Dynamic Memory Controller

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Dynamic Memory Controller for 16K and 64K MOS dynamic RAMs
- 8-Bit Refresh Counter for refresh address generation, has clear input and terminal count output
- Refresh Counter terminal count selectable at 256 or 128
- Latch input $\overline{RAS}$ Decoder provides 4 $\overline{RAS}$ outputs, all active during refresh
- Dual 8-Bit Address Latches plus separate $\overline{RAS}$ Decoder Latches
- Burst mode, distributed refresh or transparent refresh mode determined by user

GENERAL DESCRIPTION

The Am2964B Dynamic Memory Controller (DMC) replaces a dozen MSI devices by grouping several unique functions. Two 8-bit latches capture and hold the memory address. These latches and a clearable, 8-bit refresh counter feed into an 8-bit, 3-input, Schottky speed MUX for output to the dynamic RAM address lines.

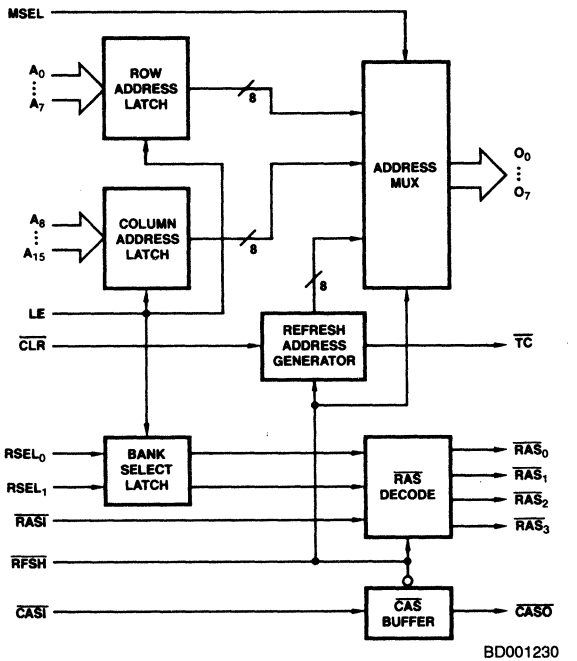
The same silicon chip also includes a special $\overline{RAS}$ decoder and $\overline{CAS}$ buffer. Placing these functions on the same chip minimizes the time skew between output functions which would otherwise be separate MSI chips, and therefore allows a faster memory cycle time by the amount of skew eliminated.

The $\overline{RAS}$ Decoder allows upper addresses to select one-of-four banks of RAM by determining which bank receives a $\overline{RAS}$ input. During refresh ($\overline{RFSH}$ = LOW) the decoder mode is changed to four-of-four and all banks of memory receive a $\overline{RAS}$ input for refresh in response to a $\overline{RASI}$ active LOW input. $\overline{CAS}$ is inhibited during refresh.

Burst mode refresh is accomplished by holding $\overline{RFSH}$ LOW and toggling $\overline{RASI}$.

A_{15} is a dual function input which controls the refresh counter's range. For 64K RAMs it is an address input. For 16K RAMs it can be pulled to +12V through 1KΩ to terminate the refresh count at 128 instead of 256.

BLOCK DIAGRAM



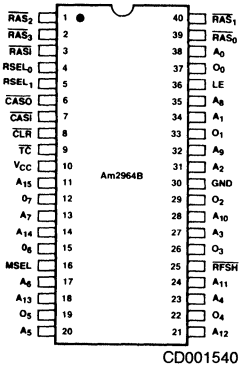
BD001230

IMOX is a trademark of Advanced Micro Devices, Inc.

Publication# 03527 Rev. C Amendment /0
Issue Date: January 1990

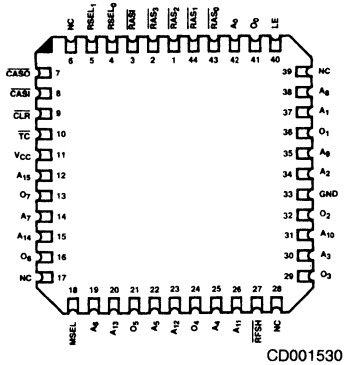
CONNECTION DIAGRAM
Top View

DIP



CD001540

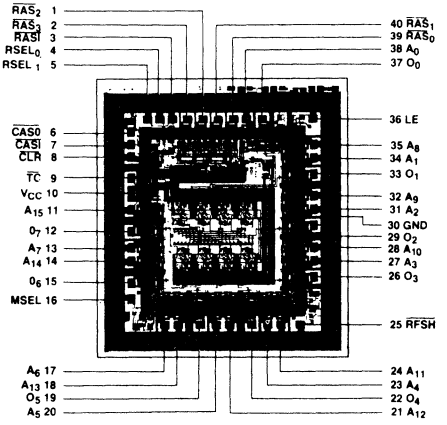
Leadless Chip Carrier



CD001530

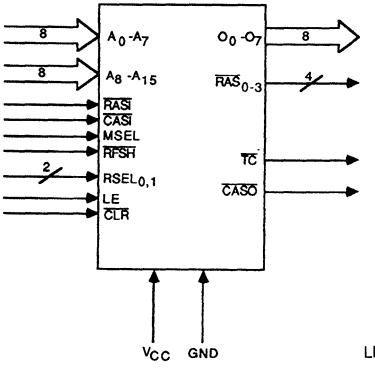
Note: Pin 1 is marked for orientation.

METALLIZATION AND PAD LAYOUT



DIE SIZE 0.156" x 0.143"

LOGIC SYMBOL

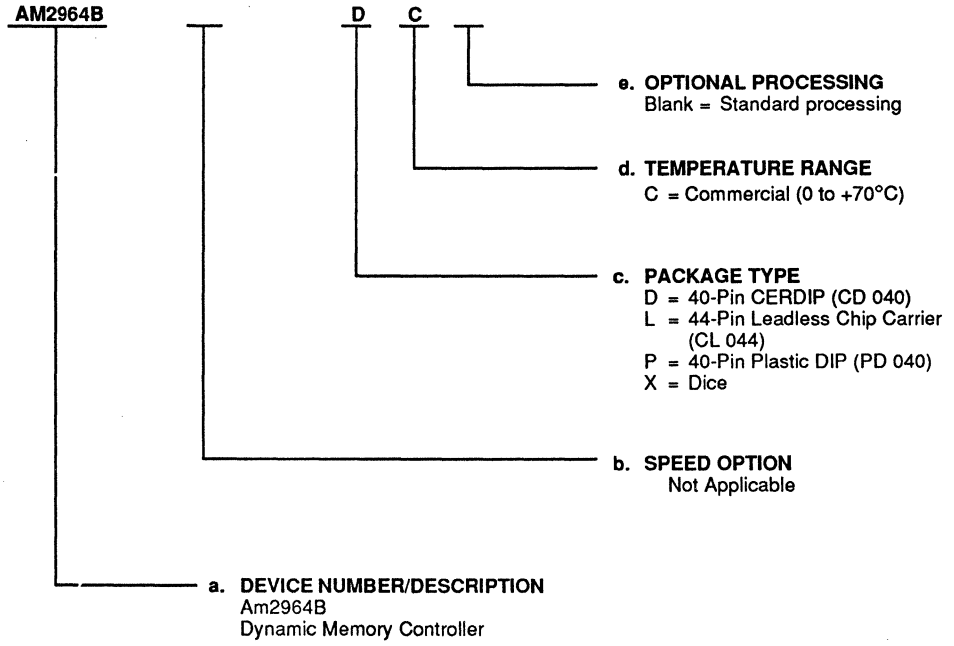


ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. **Device Number**
b. **Speed Option (if applicable)**
c. **Package Type**
d. **Temperature Range**
e. **Optional Processing**



Valid Combinations	
AM2964B	DC, LC, PC, XC

Valid Combinations

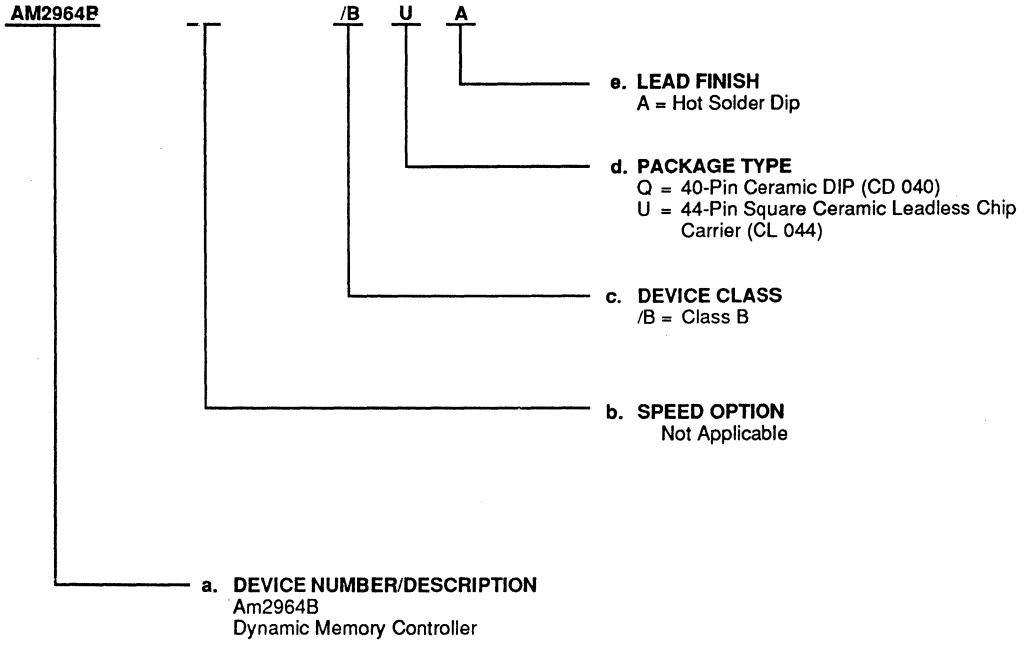
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM2964B	/BQA, /BUA

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations.

PIN DESCRIPTION

Pin No.	Name	I/O	Description
	$A_0 - A_7$	I	The low order Address inputs are used to latch eight Row Address inputs for the RAM. These inputs drive the outputs $O_0 - O_7$ when MSEL is HIGH.
	$A_8 - A_{15}$	I	The high order Address inputs are used to latch eight Column Address inputs for the RAM. These inputs drive the outputs $O_0 - O_7$ when MSEL is LOW.
11	A_{15}	I	A_{15} is a dual input. With normal TTL level inputs A_{15} acts as address input A_{15} for 64K RAMs. If A_{15} is pulled up to +12V through a 1KΩ resistor, the terminal count output. $\overline{TC}$, will go LOW every 128 counts (for 16K RAMs) instead of every 256 counts.
	$O_0 - O_7$	O	The RAM address outputs. The eight-bit width is designed for dynamic RAMs up to 64K.
16	MSEL	I	The Multiplexer-SELect input determines whether low order or high order address inputs appear at the multiplexer outputs $O_0 - O_7$. When MSEL is HIGH the low order address latches ($A_0 - A_7$) are connected to the outputs. When MSEL is LOW the high order address latches are connected to the outputs.
25	$\overline{RFSH}$	I	The Refresh control input. When active LOW the $\overline{RFSH}$ input switches the address output multiplexer to output the inverted contents of the 8-bit refresh counter. $\overline{RFSH}$ LOW also inhibits the $\overline{CAS}$ buffer and changes the mode of the $\overline{RAS}$ decoder from one-of-four to four-of-four so that all four $\overline{RAS}$ decoder outputs, $\overline{RAS_0}$, $\overline{RAS_1}$, $\overline{RAS_2}$ and $\overline{RAS_3}$, go LOW in response to a LOW input at $\overline{RASI}$. This action refreshes one row address in each of the four $\overline{RAS}$ decoded memory banks. The refresh counter is advanced at the end of each refresh cycle by the LOW-to-HIGH transition of $\overline{RFSH}$ or $\overline{RASI}$ (whichever occurs first). In burst mode refresh, $\overline{RFSH}$ may be held LOW and refresh accomplished by toggling $\overline{RASI}$.
9	$\overline{TC}$	O	The Terminal Count output. A LOW output at $\overline{TC}$ indicates that the refresh counter has been sequenced through either 128 or 256 refresh addresses depending on A_{15}. The $\overline{TC}$ output remains active LOW until the refresh counter is advanced by the rising edge of $\overline{RASI}$ or $\overline{RFSH}$.
8	$\overline{CLR}$	I	The refresh counter Clear input. An active LOW input at $\overline{CLR}$ resets the refresh counter to all LOW (refresh address output to all HIGH).
36	LE	I	The address latch enable input. An active HIGH input at LE causes the two 8-bit address latches and the 2-bit $\overline{RAS}$ Select input latch to go transparent, accepting new input data. A LOW input on LE latches the input data which meets set-up and hold time requirements.
4, 5	$RSEL_0$ and $RSEL_1$	I	The $\overline{RAS}$ decoder Select inputs. Data (latched) at these inputs (normally higher order addresses) is decoded by the $\overline{RAS}$ Decoder to "$\overline{RAS}$ Select" one of four banks of memory with $\overline{RAS_0}$, $\overline{RAS_1}$, $\overline{RAS_2}$ or $\overline{RAS_3}$.
3	$\overline{RASI}$	I	The Row Address Strobe Input. During normal memory cycles the selected $\overline{RAS}$ Decoder output $\overline{RAS_0}$, $\overline{RAS_1}$, $\overline{RAS_2}$ or $\overline{RAS_3}$ will go active LOW in response to an active LOW input at $\overline{RASI}$. During refresh ($\overline{RFSH}$ = LOW), all $\overline{RAS}$ outputs go LOW in response to $\overline{RASI}$ = LOW.
39, 40, 1, 2	$\overline{RAS_0}$, $\overline{RAS_1}$, $\overline{RAS_2}$, $\overline{RAS_3}$	O	Row Address Strobe outputs ($\overline{RAS_i}$). Each provides a Row Address Strobe for one of the four banks of memory. Each will go active LOW only when selected by $RSEL_0$ and $RSEL_1$ and only when $\overline{RASI}$ goes active LOW. All RAS_{0-3} outputs go active low in response $\overline{RASI}$ when $\overline{RFSH}$ goes LOW.
7	$\overline{CASI}$	I	The Column Address Strobe. An active LOW input at $\overline{CASI}$ will result in an active LOW output at $\overline{CASO}$, unless a refresh cycle is in progress ($\overline{RFSH}$ = LOW).
6	$\overline{CASO}$	O	The Column Address Strobe output. The active LOW $\overline{CASO}$ output strobes the Column Address into the dynamic RAM. $\overline{CASO}$ is inhibited during refresh ($\overline{RFSH}$ = LOW).

$\overline{RAS}$ OUTPUT FUNCTION TABLE

$\overline{RFSH}$	$\overline{RASI}$	$RSEL_1$	$RSEL_0$	$\overline{RAS_0}$	$\overline{RAS_1}$	$\overline{RAS_2}$	$\overline{RAS_3}$
L	H	X	X	H	H	H	H
L	L	X	X	L	L	L	L
H	H	X	X	H	H	H	H
H	L	L	L	L	H	H	H
H	L	L	H	H	L	H	H
H	L	H	L	H	H	L	H
H	L	H	H	H	H	H	L

$\overline{CASO}$ FUNCTION TABLE

$\overline{RFSH}$	$\overline{CASI}$	$\overline{CASO}$
H	L	L
H	H	H
L	X	H

ADDRESS OUTPUT FUNCTION TABLE

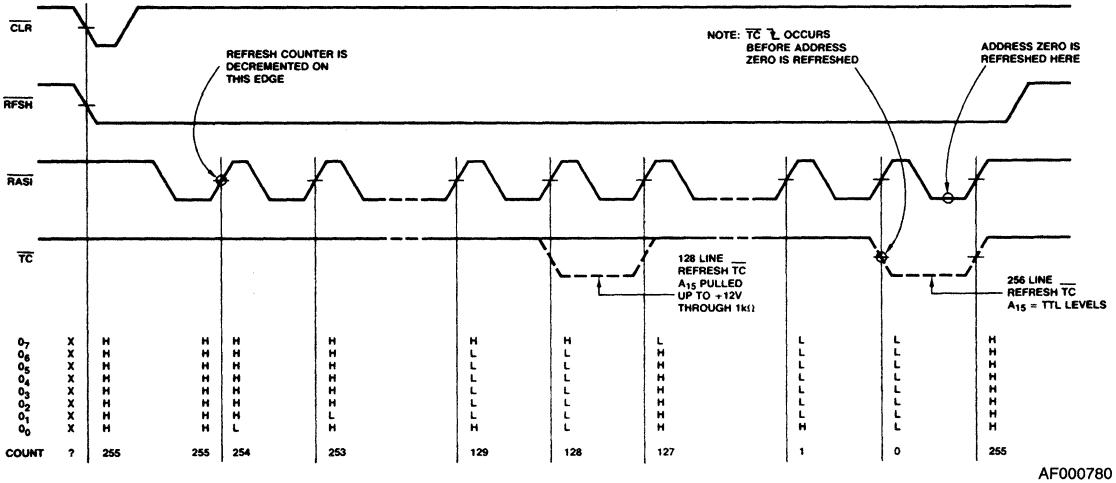
MSEL	$\overline{RFSH}$	O_0-O_7
H	H	A_0-A_7
L	H	A_8-A_{15}
X	L	Refresh Address

REFRESH ADDRESS COUNTER FUNCTION TABLE

A_{15}	$\overline{CLR}$	$\overline{RFSH}$	$\overline{RASi}$	$\overline{TC}$	REFRESH COUNT	FUNCTION
X	L	X	X	X	FF_H	Clear Counter
X	H	↓	X	X	NC	Output Refresh Address No Change for Counter
X	H	↑	L	X	Count – 1	Return to Memory Cycle Mode and Decrement Counter
X	H	L	↓	X	NC	Output all RAS_i to RAM No Change for Counter
X	H	L	↑	X	Count – 1	Return RAS_i to HIGH and Decrement Counter
L or H	H	X	X	L	00_H	Terminal Count for 256 Line Refresh
+12V*	H	X	X	L	00_H and 80_H	Terminal Count for 128 Line Refresh

* Through 1KΩ resistor.

BURST REFRESH TIMING



AF000780

The timing shown assumes that burst mode applications may power-down the Am2964B with the RAM. Therefore the counter is cleared prior to executing the refresh sequence.

FUNCTIONAL DESCRIPTION

Architecture

The Dynamic Memory Controller (DMC) provides address multiplexing, refresh address generation and $\overline{RAS}/\overline{CAS}$ control for the MOS dynamic RAM memories of any data width. The eight bit address path is designed for 64K RAMs and can be used with 16K RAMs.

Sixteen address input latches and two $\overline{RAS}$ Select latches (for higher order addresses) allow the DMC to control up to 256K words of memory (with 64K RAMs) by using the internal $\overline{RAS}$ decoder to select from one-of-four banks of RAMs.

Speed With Minimum Skew

The DMC provides Schottky speed in all of the critical paths. In addition, time skew between the Address, $\overline{RAS}$ and $\overline{CAS}$ paths is minimized (and specified) by placing these function on the same chip. The inclusion of the $\overline{CAS}$ buffer allows matching of its propagation delay, plus provides the $\overline{CAS}$ inhibit function during $\overline{RAS}$ – only refresh.

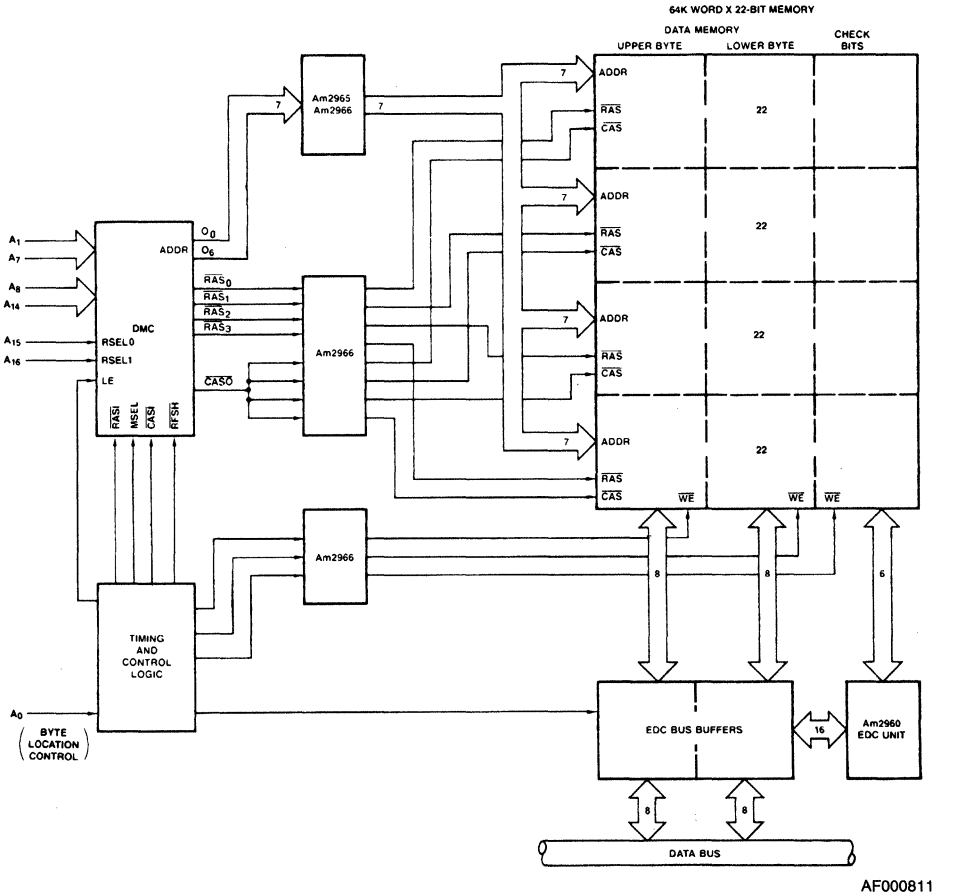
Input Latches

The eighteen input latches are transparent when LE is HIGH and latch the input data meeting set-up and hold time requirements when LE goes LOW. In systems with separate address and data buses, LE may be permanently enabled HIGH.

Refresh Counter

The 8-bit refresh counter provides both 128 and 256 line refresh capability. Refresh control is external to allow maximum user flexibility. Transparent (hidden), burst, synchronous or asynchronous refresh modes are all possible.

The refresh counter is advanced at the LOW-to-HIGH transition of $\overline{RFSH}$ (or $\overline{RASI}$). This assures a stable counter output for the next refresh cycle. The counter will continue to cycle through 256 addresses unless reset to zero by $\overline{CLR}$. This actually causes all outputs to go HIGH since the output MUX is inverting. (Address inputs to outputs are non-inverting since both the input latches and output MUX are inverting).



Address and $\overline{RAS}/\overline{CAS}$ drivers each drive 22 RAM inputs at each output. Timing skew is minimized by using one device for address lines and one device for $\overline{RAS}/\overline{CAS}$, spreading the $\overline{CAS}$ loading over four drivers to equalize the capacitive load on each driver.

Figure 1. Dynamic Memory Control with Error Detection and Correction

Refresh Terminal Count

The refresh counter also provides a Terminal Count output for burst mode refresh applications. $\overline{TC}$ normally occurs at count 255 (0_0 to 0_7 all LOW when $\overline{RFSH}$ is LOW). $\overline{TC}$ can be made to occur at count 127 for 128 line burst mode refresh by pulling A_{15} up to +12V through a 1KΩ±10% resistor. The counter actually cycles through 256 with $\overline{TC}$ determined by A_{15}. Otherwise, A_{15} functions as an address input when driven at normal TTL levels.

Three Input 8-Bit Address Multiplexer

The address MUX is 8-bits wide (for 64K RAMs) and has three data sources: the lower address input latch (A_0 to A_7), the upper address input latch (A_8 to A_{15}) and the internal refresh counter. The lower address latch is selected when MSEL is HIGH. This is normally the Row address. The upper address latch is selected when MSEL is LOW. This is normally the Column address. The third source, the refresh counter, is selected when $\overline{RFSH}$ is LOW and overrides MSEL.

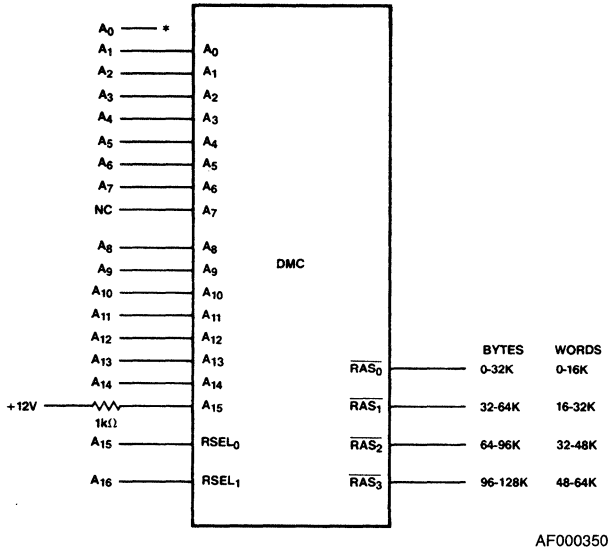
When $\overline{RFSH}$ goes LOW, the MUX selects the refresh counter address and $\overline{CASO}$ is inhibited. Also, the $\overline{RAS}$ Decoder function is changed from one-of-four to four-of-four so all $\overline{RAS}$ outputs $\overline{RAS_0}$-$\overline{RAS_3}$ go LOW to refresh all banks of memory when $\overline{RASI}$ goes LOW. When RFSH is HIGH only one $\overline{RAS}$ output goes low. This is determined by the $\overline{RAS}$ Select inputs, $RSEL_0$ and $RSEL_1$. In either case the $\overline{RAS}$ Decoder output timing is controlled by $\overline{RASI}$ to make sure the refresh count appears at 0_0-0_7 before $\overline{RAS_0}$-$\overline{RAS_3}$ go LOW. This assures meeting Row address Set-up time requirement of the RAM (t_{ASR}).

Maximum Performance System

The typical organization of a maximum performance 16-bit system including Error Detection and Correction is shown in Figure 1. Delay lines provide the most accurate timing and are recommended for $\overline{RAS}$/MSEL/$\overline{CAS}$ timing in this type of system.

Controlling 16K RAMs or Smaller Systems

16K RAMs require seven address inputs and 128 line refresh. Also, A_0 is often used to designate upper or lower byte transactions in 16-bit systems. These modifications are shown in Figure 2.



*A_0 Controls Byte Select Logic

Figure 2. Word Organized Memory Using 16K RAMs

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Ambient Temperature under Bias ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continous ..... −0.5V to +7.0V
DC Voltage Applied to Outputs For
High Output State ..... −0.5V to V_{CC} Max
DC Input Voltage ..... −0.5V to +5.5V
Output Current, Into Outputs 20mA
DC Input Current ..... −30mA to +5.0mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Temperature ..... 0°C to +70°C
Supply Voltage ..... +4.75V to +5.25V

Military (M) Devices
Temperature ..... −55°C to +125°C
Supply Voltage ..... +4.5V to +5.5V

Operating ranges define those limits over which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges unless otherwise specified

Parameters	Descriptions	Test Conditions (Note 1)		Min	Typ (Note 2)	Max	Units
V_{OH}	Output HIGH Voltage	V_{CC} = MIN, V_{IN} = V_{IH} or V_{IL}, I_{OH} = −1mA	$\overline{TC}$	2.5			Volts
			Others	3.0			Volts
V_{OH}	Output HIGH Voltage	V_{CC} = MIN, V_{IN} = V_{IH} or V_{IL}, I_{OH} = −15mA	All outputs except $\overline{TC}$	2.0			Volts
V_{OL}	Output LOW Voltage	V_{CC} = MIN, V_{IN} = V_{IH} or V_{IL}	All outputs except $\overline{TC}$, I_{OL} = 16mA			0.5	Volts
			$\overline{TC}$, I_{OL} = 8mA			0.5	Volts
V_{IH}	Input HIGH level	Guaranteed input logical HIGH voltage for all inputs		2.0			Volts
V_{IL}	Input LOW level	Guaranteed input logical LOW voltage for all inputs				0.8	Volts
V_I	Input Clamp Voltage	V_{CC} = MIN, I_{IN} = −18mA				−1.5	Volts
I_{IL}	Input LOW Current	V_{CC} = MAX, V_{IN} = 0.4V	$\overline{RAS}$			−3.2	mA
			$\overline{CAS}$, MSEL, $\overline{RFSH}$			−1.6	mA
			A_0 −A_{15}, $\overline{CLR}$ $RSEL_{0,1}$, LE			−0.4	mA
I_{IH}	Input HIGH Current	V_{CC} = MAX, V_{IN} = 2.7V	$\overline{RAS}$			100	µA
			$\overline{CAS}$, MSEL, $\overline{RFSH}$			50	µA
			A_0-A_{15}, $\overline{CLR}$ $RSEL_{0,1}$, LE			20	µA
I_I	Input HIGH Current	V_{CC} = MAX, V_{IN} = 5.5V	$\overline{RAS}$			2.0	mA
			$\overline{CAS}$, MSEL, $\overline{RFSH}$			1.0	mA
		V_{CC} = MAX, V_{IN} = 5.5V	A_0-A_{15}, $\overline{CLR}$ $RSEL_{0,1}$, LE			0.1	mA
I_{SC}	Output Short Circuit Current	V_{CC} = MAX (Note 3)		−40		−100	mA
I_{CC}	Power Supply Current (Note 4)	25°C, 5V			122		mA
		0°C to 70°C	COM'L			173	mA
		70°C					
		−55°C to +125°C	MIL			164	mA
		+125°C				150	mA
I_T	A_{15} Enable Current	A_{15} connected to +12V through 1KΩ±10%				5	mA

Notes: 1. For conditions shown as MIN or MAX, use the appropriate value specified under Operating Range for the applicable device type.
2. Typical limits are at V_{CC} = 5.0V, 25°C ambient and maximum loading.
3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
4. I_{CC} is worst case when the Address inputs are latched HIGH, the refresh counter is at terminal count (255), $\overline{RAS}$ and $\overline{CAS}$ are HIGH and all other inputs are LOW.

SWITCHING CHARACTERISTICS over operating range for $C_L = 50pF$

Am2964B (Notes 5, 6)

Parameter		Description	Test Conditions	Typ	COMMERCIAL		MILITARY		Units
					Min	Max	Min	Max	
1	t_{PD}	A_i to O_i Delay	C_L = 50pF	14		19		23	ns
2	t_{PHL}	$\overline{RASI}$ to $\overline{RAS_i}$ ($\overline{RFSH}$ = H)		14		20		23	ns
3	t_{PHL}	$\overline{RASI}$ to $\overline{RAS_i}$ ($\overline{RFSH}$ = L)		14		20		23	ns
4	t_{PD}	MSEL to O_i		17	9		9		ns
5	t_{PD}	MSEL to O_i		17		21		25	ns
6	t_{PHL}	$\overline{CASI}$ to $\overline{CASO}$ ($\overline{RFSH}$ = H)		12		17		19	ns
7	t_{PHL}	$RSEL_i$ to $\overline{RAS_i}$ (LE = H, $\overline{RASI}$ = L)		15		20		24	ns
8	t_{PLH}	$\overline{RFSH}$ to $\overline{TC}$ ($\overline{RASI}$ = L)		30		40		50	ns
9	t_{PLH}	$\overline{RASI}$ to $\overline{TC}$ ($\overline{RFSH}$ = L)		25		35		40	ns
10	t_{PW}	$\overline{RASI}$ = L ($\overline{RFSH}$ = L)		10	50		50		ns
11	t_{PW}	$\overline{RASI}$ = H ($\overline{RFSH}$ = L)		10	50		50		ns
12	t_{PD}	$\overline{RFSH}$ to O_i ($\overline{RASI}$ = X)		17		21		25	ns
13	t_{PHL}	$\overline{RFSH}$ to $\overline{RAS_i}$ ($\overline{RASI}$ = L)		19		26		29	ns
14	t_{PW}	$\overline{CLR}$ = L		10	30		35		ns
15	t_{PLH}	$\overline{RFSH}$ to $\overline{CASO}$ ($\overline{RASI}$ = L $\overline{CASI}$ = L, Note 7)		16		21		25	ns
16	t_{PD}	LE to O_i		25		35		40	ns
17	t_{PHL}	LE to $\overline{RAS_i}$		30		40		45	ns
18	t_{PLH}	$\overline{CLR}$ to $\overline{TC}$		35		45		56	ns
19	t_{PLH}	$\overline{CLR}$ to O_i ($\overline{RFSH}$ = L)		31		44		54	ns
20	t_S	A_i to LE Set-Up Time		0	5		5		ns
21	t_H	A_i to LE Hold Time		5	12		15		ns
22	t_S	$RSEL_i$ to LE Set-Up Time		0	5		5		ns
23	t_H	$RSEL_i$ to LE Hold Time		10	17		25		ns
24	t_S	$\overline{CLR}$ Recovery Time		10	16		18		ns
25	t_{SKEW}	O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = H, Note 8)		2		5		6	ns
26	t_{SKEW}	O_i to $\overline{CASO}$ (Note 8)		6		8		8	ns
27	t_{SKEW}	O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = L, Note 9)		6		8		10	ns
28	t_{SKEW}	O_i to $\overline{RAS_i}$ (MSEL = ⌝⌞, Note 10)		1		5		5	ns

Notes: 5. Minimum spec limits for t_{PW}, t_S and t_H are minimum system operating requirements. Limits for t_{SKEW} and t_{PD} are guaranteed test limits for the device.

6. All AC parameters are specified at the 1.5V level.

7. $\overline{RFSH}$ inhibits $\overline{CASO}$ during refresh. Specification is for $\overline{CASO}$ inhibit time.

8. O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = HIGH) skew is guaranteed maximum difference between fastest $\overline{RASI}$ to $\overline{RAS_i}$ delay and slowest A_i to O_i delay within a single device. O_i to $\overline{CASO}$ skew is maximum difference between fastest $\overline{CASI}$ to $\overline{CASO}$ delay and slowest MSEL to O_i delay within a single device. See application section entitled Memory Cycle Timing for correlation to System Timing requirements.

9. O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = LOW) skew is guaranteed maximum difference between fastest $\overline{RASI}$ to $\overline{RAS_i}$ delay and slowest $\overline{RFSH}$ to O_i delay within a single device. See application section on Refresh Timing for correlation to system refresh timing requirements.

10. O_i to $\overline{RAS_i}$ (MSEL = ⌝⌞) skew is guaranteed maximum difference between fastest MSEL ⌝⌞ to O_i delay and slowest $\overline{RASI}$ to $\overline{RAS_i}$ delay within a single device.

SWITCHING CHARACTERISTICS over operating range for $C_L = 50pF$

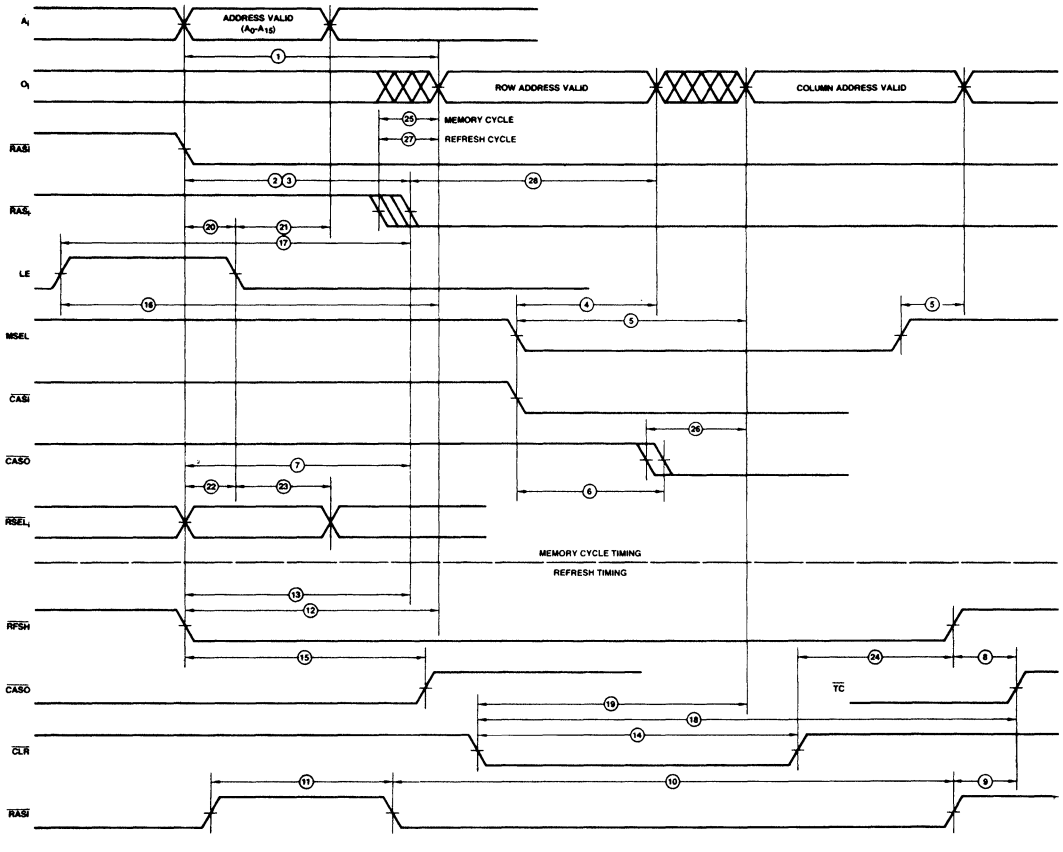
Am2964B (Notes 5, 6)

Parameter		Description	Test Conditions	Typ	COMMERCIAL		MILITARY		Units
					Min	Max	Min	Max	
1	t_{PD}	A_i to O_i Delay	C_L = 150pF	20		25		30	ns
2	t_{PHL}	$\overline{RASI}$ to $\overline{RAS_i}$ ($\overline{RFSH}$ = H)		18		24		27	ns
3	t_{PHL}	$\overline{RASI}$ to $\overline{RAS_i}$ ($\overline{RFSH}$ = L)		18		24		27	ns
4	t_{PD}	MSEL to O_i		23	12		12		ns
5	t_{PD}	MSEL to O_i		23		27		31	ns
6	t_{PHL}	$\overline{CASI}$ to $\overline{CASO}$ ($\overline{RFSH}$ = H)		17		24		26	ns
7	t_{PHL}	$RSEL_i$ to $\overline{RAS_i}$ (LE = H, $\overline{RASI}$ = L)		19		27		30	ns
8	t_{PLH}	$\overline{RFSH}$ to $\overline{TC}$ ($\overline{RASI}$ = L)		34		45		55	ns
9	t_{PLH}	$\overline{RASI}$ to $\overline{TC}$ ($\overline{RFSH}$ = L)		32		45		55	ns
10	t_{PW}	$\overline{RASI}$ = L ($\overline{RFSH}$ = L)		10	50		50		ns
11	t_{PW}	$\overline{RASI}$ = H ($\overline{RFSH}$ = L)		10	50		50		ns
12	t_{PD}	$\overline{RFSH}$ to O_i ($\overline{RASI}$ = X)		21		27		30	ns
13	t_{PHL}	$\overline{RFSH}$ to $\overline{RAS_i}$ ($\overline{RASI}$ = L)		25		33		36	ns
14	t_{PW}	$\overline{CLR}$ = L		10	30		35		ns
15	t_{PLH}	$\overline{RFSH}$ to $\overline{CASO}$ ($\overline{RASI}$ = L $\overline{CASI}$ = L, Note 7)		21		27		31	ns
16	t_{PD}	LE to O_i		30		40		50	ns
17	t_{PHL}	LE to $\overline{RAS_i}$		34		45		54	ns
18	t_{PLH}	$\overline{CLR}$ to $\overline{TC}$		39		55		60	ns
19	t_{PLH}	$\overline{CLR}$ to O_i ($\overline{RFSH}$ = L)		38		50		62	ns
20	t_S	A_i to LE Set-Up Time		0	5		5		ns
21	t_H	A_i to LE Hold Time		5	12		12		ns
22	t_S	$RSEL_i$ to LE Set-Up Time		0	5		5		ns
23	t_H	$RSEL_i$ to LE Hold Time		10	17		25		ns
24	t_S	$\overline{CLR}$ Recovery Time		10	16		18		ns
25	t_{SKEW}	O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = H, Note 8)		3		6		7	ns
26	t_{SKEW}	O_i to $\overline{CASO}$ (Note 8)		6		8		8	ns
27	t_{SKEW}	O_i to $\overline{RAS_i}$ ($\overline{RFSH}$ = L, Note 9)		6		9		10	ns
28	t_{SKEW}	O_i to $\overline{RAS_i}$ ($\overline{MSEL}$ = ⌐¯, Note 10)		1		5		5	ns

Notes on Testing

Incoming test procedures on this device should be carefully planned, taking into account the high complexity and power levels of the part. The following notes may be useful.

1. Insure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400mA in 5-8ns. Inductance in the ground cable may allow the ground pin at the device to rise by 100s of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leq 0.4V$ and $V_{IH} \geq 2.4V$ for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. To assist in testing, AMD offers complete documentation on our test procedures.



WF001990

Am2964B Dynamic Memory Controller Timing

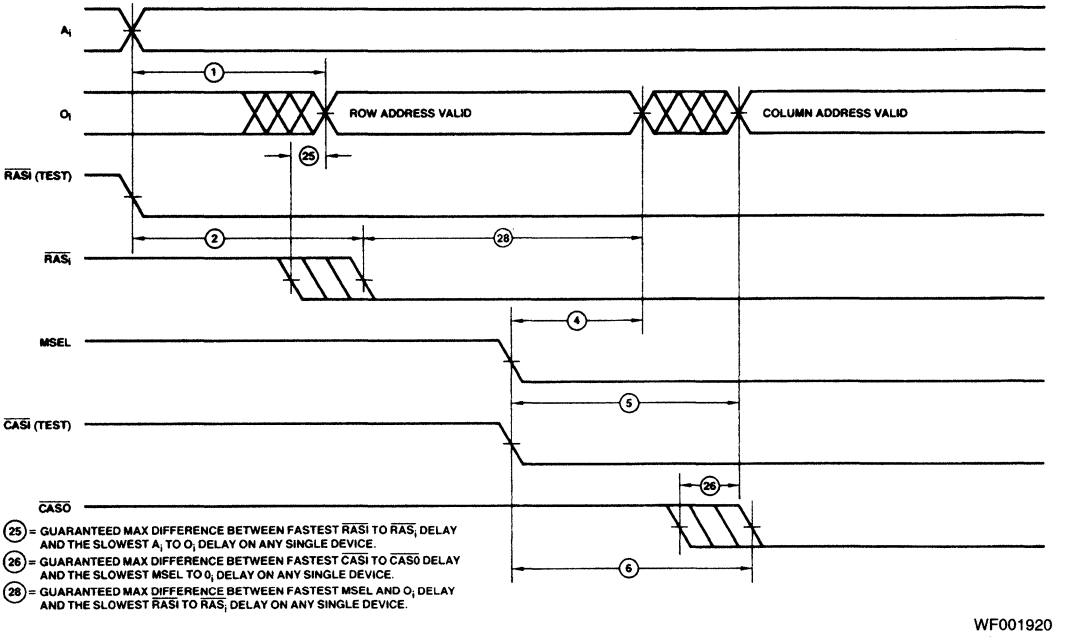
MEMORY CYCLE TIMING

The relationship between DMC specifications and system timing requirements are shown in Figure 3. T_1, T_2 and T_3 represent the minimum timing requirements at the DMC inputs to guarantee that RAM timing requirements are met and that maximum system performance is achieved.

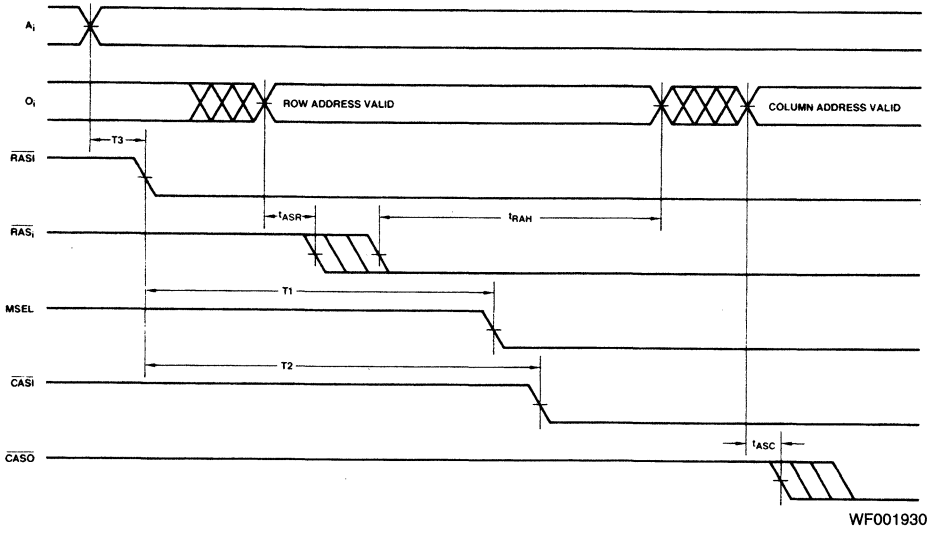
The minimum requirement for T_1, T_2 and T_3 are as follows:

$T_1MIN = t_{RAH} + t_{28}$
$T_2MIN = T_1 + t_{26} + t_{ASC}$
$T_3MIN = t_{ASR} + t_{25}$

See RAM data sheet for applicable values for t_{RAH}, t_{ASC} and t_{ASR}.



a. Specifications Applicable to Memory Cycle Timing

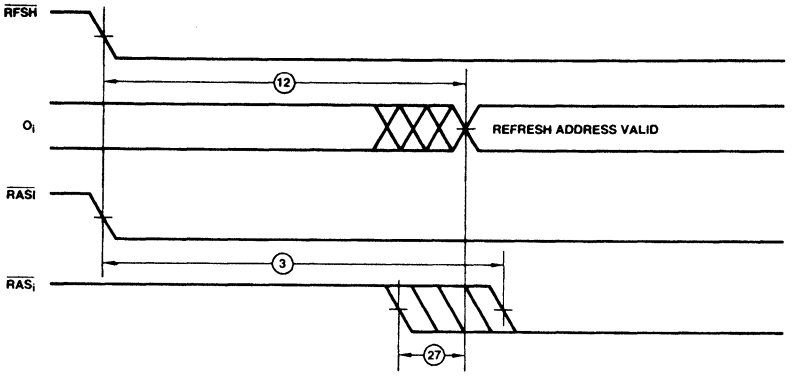


b. Desired System Timing

Figure 3. Memory Cycle Timing

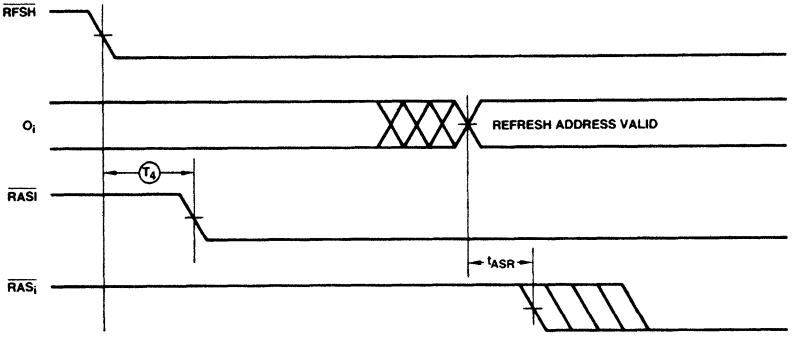
REFRESH CYCLE TIMING

The timing relationships for refresh are shown in Figure 4.
T_4 minimum is calculated as follows: $T_4 = t_{ASR} + t_{27}$



WF001940

a. Test Waveforms



(27) = GUARANTEED MAX DIFFERENCE BETWEEN FASTEST $\overline{RASI}$ TO $\overline{RAS_i}$ DELAY AND SLOWEST $\overline{RFSH}$ TO O_i DELAY ON ANY SINGLE DEVICE.

WF001880

b. Desired System Timing

Figure 4. Refresh Timing

Final

Am2965/Am2966

Octal Dynamic Memory Drivers with Three-State Outputs

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- **Controlled rise and fall characteristics**
  Internal resistors provide symmetrical drive to HIGH and LOW states, eliminating need for external series resistor.
- **Output swings designed to drive 16K and 64K RAMs**
  V_{OH} guaranteed at V_{CC} −1.15V. Undershoot going LOW guaranteed at less than 0.5V.
- **Large capacitive drive capability**
  35mA min source or sink current at 2.0V. Propagation delays specified for 50pF and 500pF loads.
- **Pin-compatible with 'S240 and 'S244**
  Non-inverting Am2966 replaces 74S244; inverting Am2965 replaces 74S240. Faster than 'S240/244 under equivalent load.
- **No-glitch outputs**
  Outputs forced into OFF state during power up and down. No glitch coming out of three-state.

GENERAL DESCRIPTION

The Am2965 and Am2966 are designed and specified to drive the capacitive input characteristics of the address and control lines of MOS dynamic RAMs. The unique design of the lower output driver includes a collector resistor to control undershoot on the HIGH-to-LOW transition. The upper output driver pulls up to V_{CC} − 1.15V to be compatible with MOS memory and is designed to have a rise time symmetrical with the lower output's controlled fall time. This allows optimization of Dynamic RAM performance.

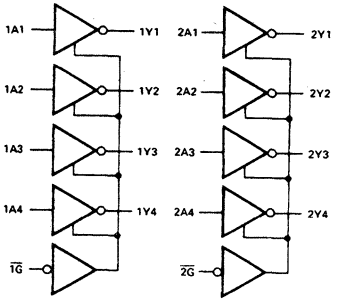
The Am2965 and Am2966 are pin-compatible with the popular 'S240 and 'S244 with identical 3-state output enable controls. The Am2965 has inverting drivers and the Am2966 has non-inverting drivers.

The inclusion of an internal resistor in the lower output driver eliminates the requirement for an external series resistor, therefore reducing package count and the board area required. The internal resistor controls the output fall and undershoot without slowing the output rise.

These devices are designed for use with the Am2964 Dynamic Memory Controller where large dynamic memories with highly capacitive input lines require additional buffering. Driving eight address lines or four $\overline{RAS}$ and four $\overline{CAS}$ lines with drivers on the same silicon chip also provides a significant performance advantage by minimizing skew between drivers. Each device has specified skew between drivers to improve the memory access worst case timing over the min and max t_{PD} difference of unspecified devices.

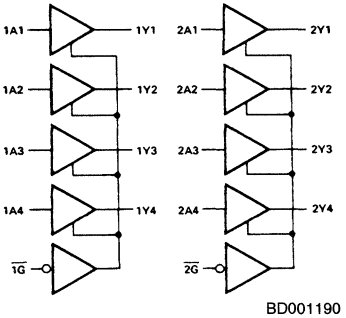
BLOCK DIAGRAM

Am2965



BD001180

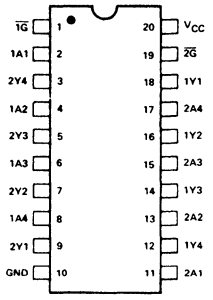
Am2966



BD001190

Publication# 05409 Rev. B Amendment /0
Issue Date: January 1990

CONNECTION DIAGRAM
Top View



CD001440

Note: Pin 1 is marked for orientation

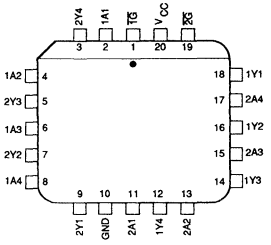
Am2965

Inputs		Outputs
$\overline{G}$	A	Y
H	X	Z
L	H	L
L	L	H

Am2966

Inputs		Outputs
$\overline{G}$	A	Y
H	X	Z
L	L	L
L	H	H

Plastic Leaded Chip Carrier
Leadless Chip Carrier

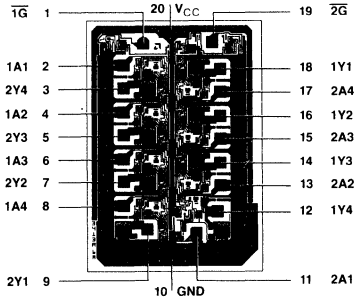


05409-001A

CD012030

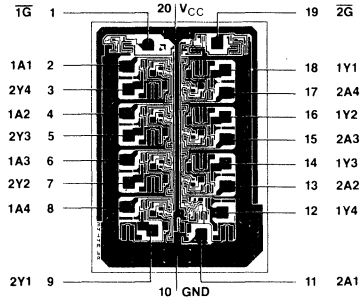
METALLIZATION AND PAD LAYOUT

Am2965



DIE SIZE 0.094″ x 0.060″

Am2966



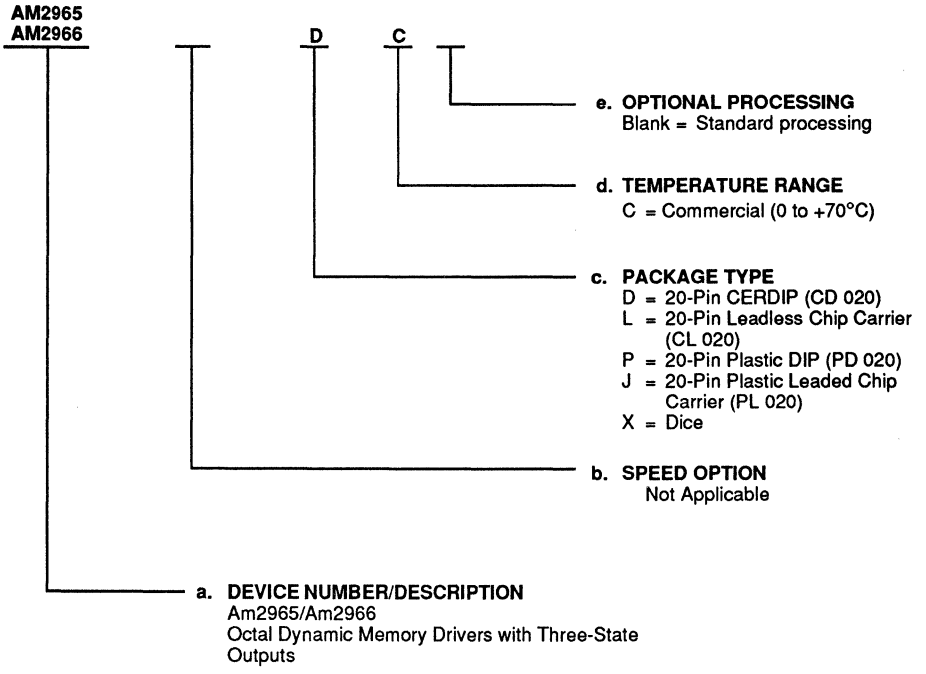
DIE SIZE 0.094″ x 0.066″

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. **Device Number**
b. **Speed Option (if applicable)**
c. **Package Type**
d. **Temperature Range**
e. **Optional Processing**



Valid Combinations	
AM2965 AM2966	DC, JC, LC, PC, XC

Valid Combinations

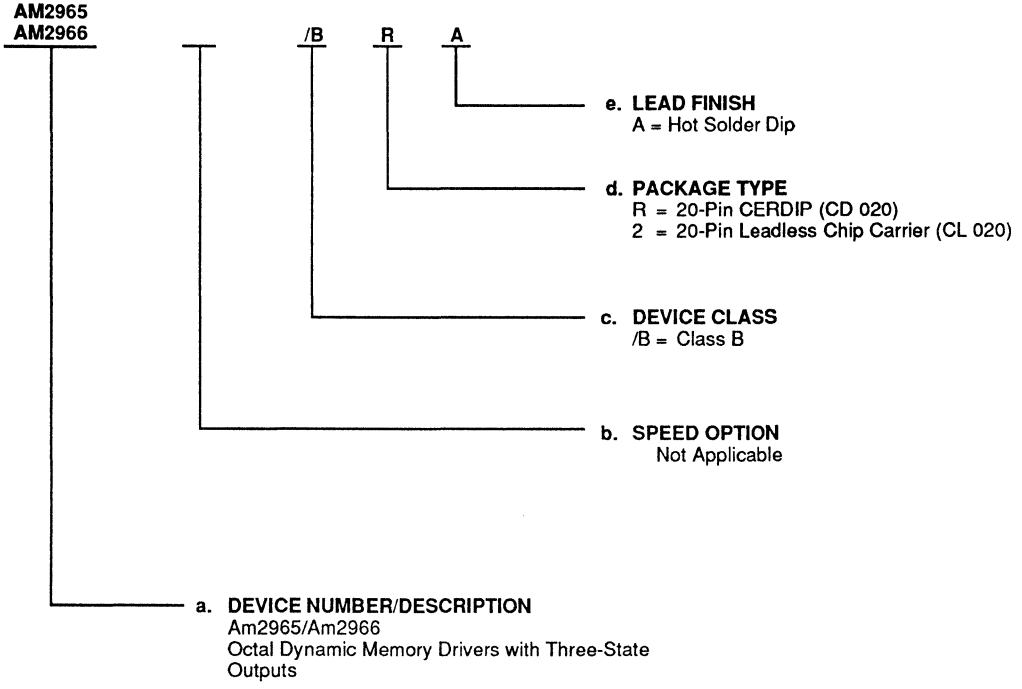
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of:

a. **Device Number**
b. **Speed Option (if applicable)**
c. **Package Type**
d. **Temperature Range**
e. **Optional Processing**

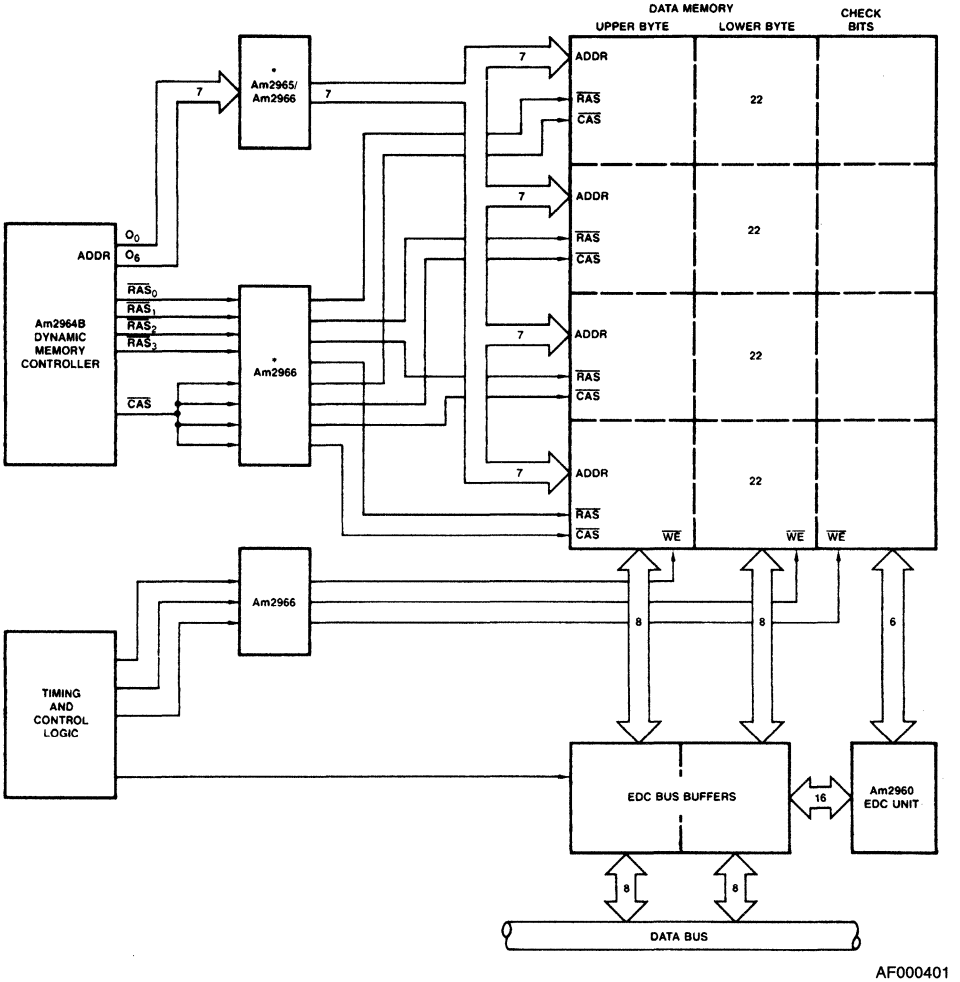


Valid Combinations	
AM2965 AM2966	/B2A, /BRA

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations.

APPLICATION



AF000401

*Address and $\overline{RAS}/\overline{CAS}$ drivers each drive 22 RAM inputs at each output. Timing skew is minimized by using one device for address lines and one device for $\overline{RAS}/\overline{CAS}$, spreading the $\overline{CAS}$ loading over four drivers to equalize the capacitive load on each driver.

DYNAMIC MEMORY CONTROL WITH ERROR DETECTION AND CORRECTION

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Temperature (Ambient)
Under Bias ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continous ..... −0.5V to +7.0V
DC Voltage Applied to Outputs For
High Output State ..... −0.5V to V_{CC} Max
DC Input Voltage ..... −0.5V to +7.0V
DC Output Current, Into Outputs 200mA
DC Input Current ..... −30mA to +5.0mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Temperature ..... 0°C to +70°C
Supply Voltage ..... +4.75V to +5.25V

Military (M) Devices
Temperature ..... −55°C to +125°C
Supply Voltage ..... +4.5V to +5.5V

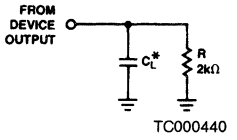
Operating ranges define those limits over which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges unless otherwise specified

Parameters	Descriptions		Test Conditions (Note 1)		Min	Typ (Note 2)	Max	Units
V_{OH}	Output HIGH Voltage		V_{CC} = MIN, V_{IN} = V_{IH} or V_{IL}	I_{OH} = −1mA	V_{CC} −1.15	V_{CC} −0.7V		Volts
V_{OL}	Output LOW Voltage		V_{CC} = MIN, V_{IN} = V_{IH} or V_{IL}	I_{OL} = 1mA			0.5	Volts
				I_{OL} = 12mA			0.8	
V_{IH}	Input HIGH Level		Guaranteed input logical HIGH voltage for all inputs		2.0			Volts
V_{IL}	Input LOW Level		Guaranteed input logical LOW voltage for all inputs				0.8	Volts
V_I	Input Clamp Voltage		V_{CC} = MIN, I_{IN} = −18mA				−1.2	Volts
I_{IL}	Input LOW Current		V_{CC} = MAX, V_{IN} = 0.4V	DATA			−200	μA
				$\overline{1G}$, $\overline{2G}$			−400	
I_{IH}	Input HIGH Current		V_{CC} = MAX, V_{IN} = 2.7V				20	μA
I_I	Input HIGH Current		V_{CC} = MAX, V_{IN} = 7.0V				0.1	mA
I_{OZH}	Off-State Current		V_O = 2.7V				100	μA
I_{OZL}	Off-State Current		V_O = 0.4V				−200	μA
I_{OL}	Output Sink Current		V_{OL} = 2.0V		50			mA
I_{OH}	Output Source Current		V_{OH} = 2.0V		−35			mA
I_{SC}	Output Short Circuit Current (Note 3)		V_{CC} = MAX		−60 (see I_{OH}		−200	mA
I_{CC}	Supply Current	Am2965	All Outputs HIGH	V_{CC} = MAX Outputs Open		24	50	mA
			All Outputs LOW			86	125	
			All Outputs Hi-Z			86	125	
		Am2966	All Outputs HIGH	V_{CC} = MAX Outputs Open		53	75	
			All Outputs LOW			92	130	
			All Outputs Hi-Z			116	150	

Notes: 1. For conditions shown as MIN or MAX, use the appropriate value specified under Operating Range for the applicable device type.
2. Typical limits are at V_{CC} = 5.0V, 25°C ambient and maximum loading.
3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.

SWITCHING TEST CIRCUIT



*t_{pd} specified at C = 50 and 500pF.

Figure 1. Capacitive Load Switching.

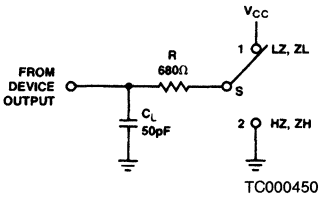
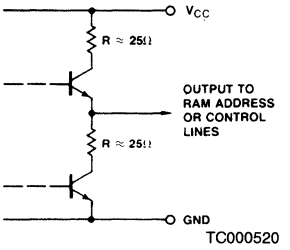


Figure 2. Three-State Enable/Disable.

TYPICAL OUTPUT DRIVER



SWITCHING CHARACTERISTICS (T_A = +25°C, V_{CC} = 5.0V)

Parameters	Description	Test Conditions		Min	Typ	Max	Units
t_{PLH}	Propagation Delay Time from LOW-to-HIGH Output	Figure 1 Test Circuit Figure 3 Voltage Levels and Waveforms	C_L = 0pF		6	(Note 4)	ns
			C_L = 50pF	6	9	15	
			C_L = 500pF	18	22	30	
t_{PHL}	Propagation Delay Time from HIGH-to-LOW Output		C_L = 0pF		4	(Note 4)	ns
			C_L = 50pF	5	7	15	
			C_L = 500pF	18	22	30	
t_{PLZ}	Output Disable Time from LOW, HIGH	Figures 2 and 4, S = 1			11	20	ns
t_{PHZ}		Figures 2 and 4, S = 2			6.5	12	
t_{PZL}	Output Enable Time from LOW, HIGH	Figures 2 and 4, S = 1			12	20	ns
t_{PZH}		Figures 2 and 4, S = 2			12	20	
t_{SKEW}	Output-to-Output Skew	Figures 1 and 3, C_L = 50pF			±0.5	±3.0 (Note 5)	ns
V_{ONP}	Output Voltage Undershoot	Figures 1 and 3, C_L = 50pF			0	−0.5	Volts

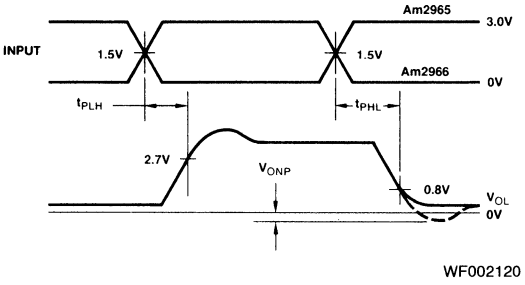
SWITCHING CHARACTERISTICS over operating ranges unless otherwise specified (Note 6)

Parameters	Description	Test Conditions		COMMERCIAL		MILITARY		Units
				Min	Max	Min	Max	
t_{PLH}	Propagation Delay Time LOW-to-HIGH Output	Figures 1 and 3	C_L = 50pF	4	17	4	20	ns
			C_L = 500pF	18	35	18	40	
t_{PHL}	Propagation Delay Time HIGH-to-LOW Output	Figures 1 and 3	C_L = 50pF	4	17	4	20	ns
			C_L = 500pF	18	35	18	40	
t_{PLZ}	Output Disable Time from LOW, HIGH	Figures 2 and 4	S = 1		24		24	ns
t_{PHZ}			S = 2		16		16	
t_{PZL}	Output Enable Time from LOW, HIGH	Figures 2 and 4	S = 1		28		28	ns
t_{PZH}			S = 2		28		28	
V_{ONP}	Output Voltage Undershoot	Figures 1 and 3, C_L = 50pF			−0.5		−0.5	Volts

Notes: 4. Typical time shown for reference only – not tested.
5. Time Skew specification is guaranteed by design but not tested.
6. AC performance over the operating temperature range is guaranteed by testing defined in Group A, Subgroup 9.
7. T_C = −55 to +125°C for Flatpak versions.

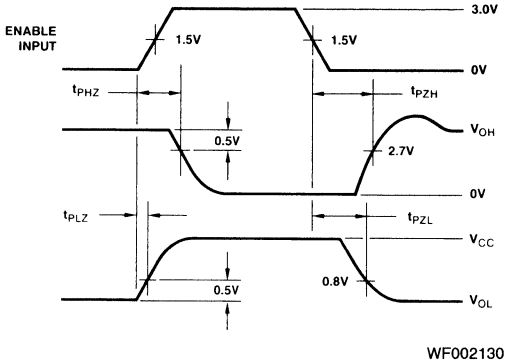
TYPICAL SWITCHING CHARACTERISTICS

SWITCHING TEST WAVEFORMS



$t_r = t_f = 2.5ns$
f = 2.5MHz
t_{pw} = 200ns

Figure 3. Output Drive Levels.



$t_r = t_f = 2.5ns$
f = 1MHz
t_{pw} = 800ns

Figure 4. Three-State Control Levels.

The RAM Driver symmetrical output design offers significant improvement over a standard Schottky output by providing a balanced drive output impedance (≈25Ω both HIGH and LOW), and by pulling up to MOS V_{OH} levels (V_{CC} – 1.5V). External resistors, not required with the RAM Driver, protect standard Schottky drivers from error causing undershoot but also slow the output rise by adding to the internal R.

The RAM Driver is optimized to drive LOW at maximum speed based on safe undershoot control and to drive HIGH with a symmetrical speed characteristic. This is an optimum approach, because the dominant RAM loading characteristic is input capacitance.

The curves shown below provide performance characteristics typical of both the inverting (Am2965) and non-inverting (Am2966) RAM Drivers.

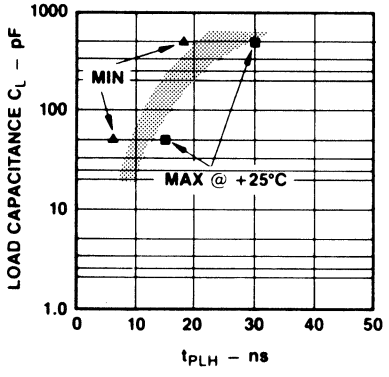


Figure 5. t_{PLH} for V_{OH} = 2.7 Vol vs. C_L.

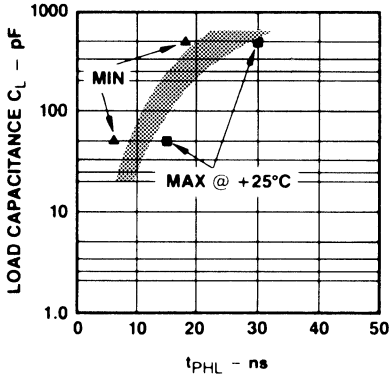


Figure 6. t_{PHL} for V_{OL} = 0.8 Volts vs. C_L.

The curves above depict the typical tPLH and tPHL for the RAM Driver outputs as a function of load capacitance. The minimums and maximums are shown for worst case design. The typical band is provided as a guide for intermediate capacitive loads.

Final

Am2968A

256K Dynamic Memory Controller/Driver

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

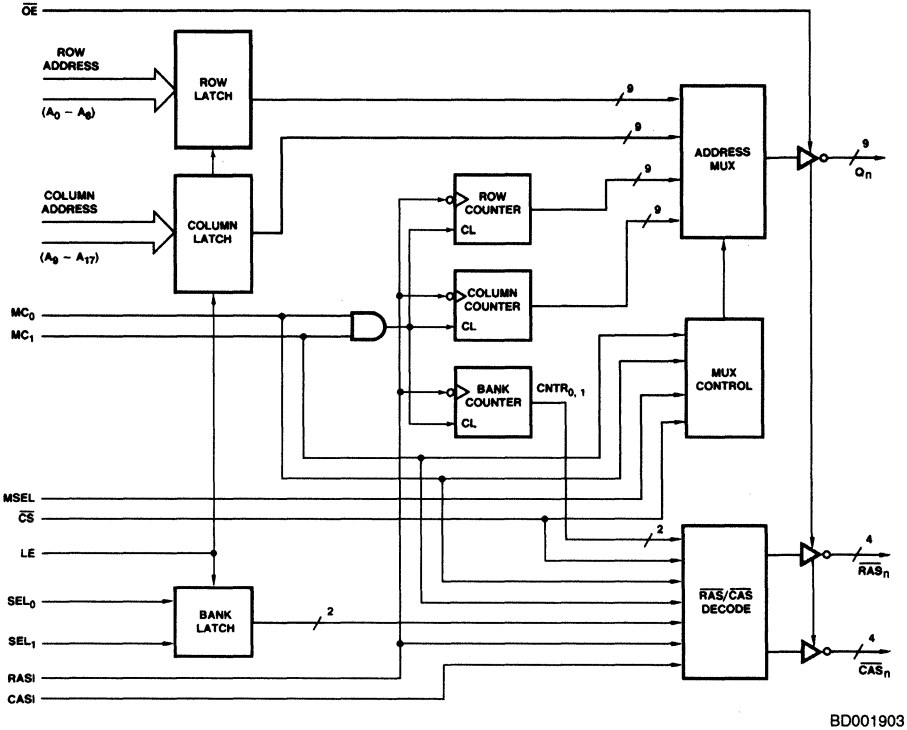
- Provides control for 16K, 64K, and 256K dynamic RAMs
- Outputs directly drive up to 88 DRAMs, with a guaranteed worst-case limit on the undershoot
- Highest-order two address bits select one of four banks of RAMs
- Separate output enable for multi-channel access to memory on surface-mount packages
- Supports scrubbing operations and other specialty access modes
- Upgrade path to Am29368 1M DRAM Controller

GENERAL DESCRIPTION

The Am2968A Dynamic Memory Controller/Driver (DMC) is intended to be used with today's high performance memory systems. The DMC acts as the address controller between any processor and dynamic memory array, using its two 9-bit address latches to hold the Row and Column addresses for any DRAM up to 256K. These latches, and the two Row/Column refresh address counters, feed into a 9-bit, 4-input MUX for output to the dynamic RAM address lines. A 2-bit bank select latch for the two high-order address bits is provided to select one each of the four $\overline{RAS}_n$ and $\overline{CAS}_n$ outputs.

The Am2968A has two basic modes of operation, read/write and refresh. In refresh mode, the two counters cycle through the refresh addresses. If memory scrubbing is not being implemented, only the Row Counter is used, generating up to 512 addresses to refresh a 512-cycle-refresh 256K DRAM. When memory scrubbing is being performed, both the Row and Column counters are used to perform read-modify-write cycles. In this mode all $\overline{RAS}_n$ outputs will be active while only one $\overline{CAS}_n$ is active at a time.

BLOCK DIAGRAM



BD001903

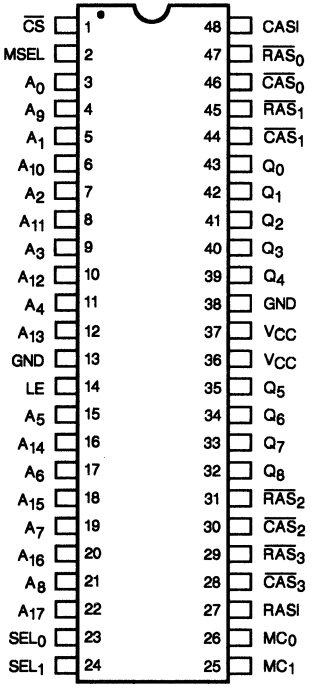
Publication# 04564 Rev. D Amendment /0
Issue Date: January 1990

RELATED AMD PRODUCTS

Part No.	Description
Am29C668	4M Configurable Dynamic Memory Controller/Driver
Am29C660D	12ns 32-Bit Cascadable EDC
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port Multiple Bus Exchange w/Parity
Am29C60A	16-Bit Cascadable EDC, High Speed
Am29368	1M Dynamic Memory Controller/Driver
Am2971A	100MHz Enhanced Programmable Event Generator
Am2976	11-Bit DRAM Driver
Am2965/6	8-Bit DRAM Driver (Inverting, Non-inverting)
Am29C827A	10-Bit Buffer
Am29C828A	10-Bit Buffer (Inverting)

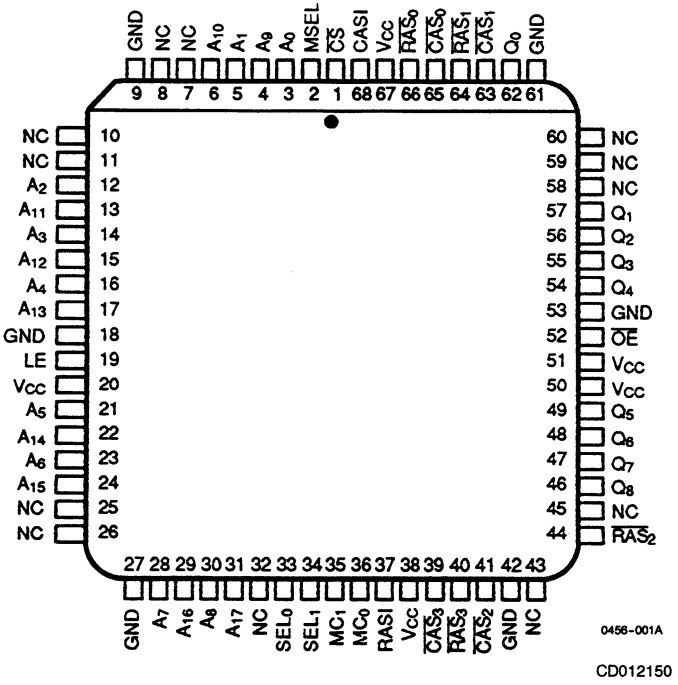
CONNECTIONS DIAGRAMS
Top View

DIPs*



CD005053

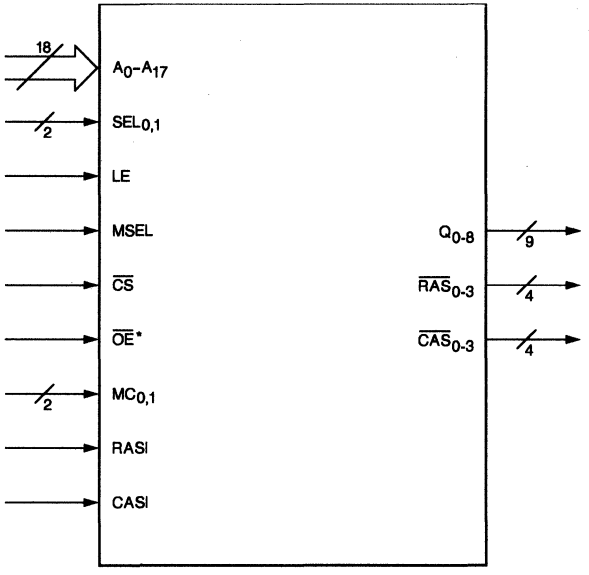
PLCC



0456-001A

CD012150

LOGIC DIAGRAM



LS002841

Die Size: 0.205″ x 0.256″ Gate Count: 300

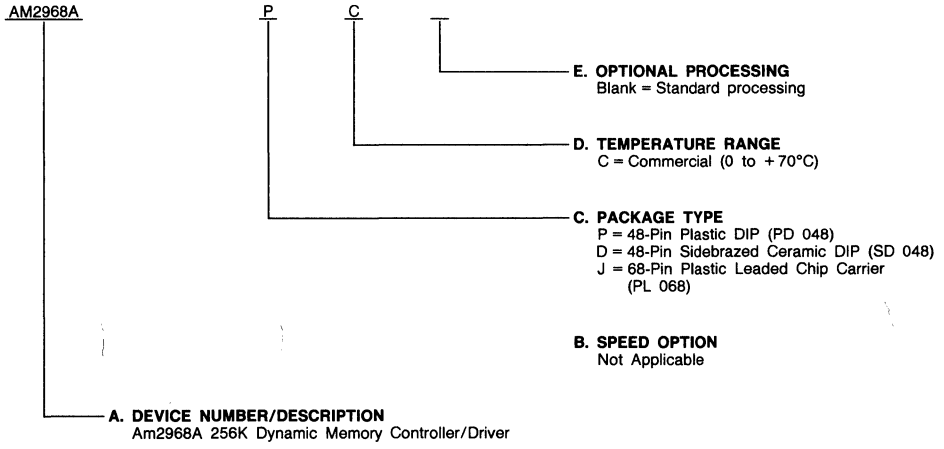
Parameter	CERDIP	PDIP	PLCC	Units
θ_{JA}	37	55	31	35
θ_{JC}	10	N/A	6	N/A

* Available only on surface mount packages.

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **A. Device Number**
B. Speed Option (if applicable)
C. Package Type
D. Temperature Range
E. Optional Processing



Valid Combinations	
AM2968A	PC, DC, JC

Valid Combinations

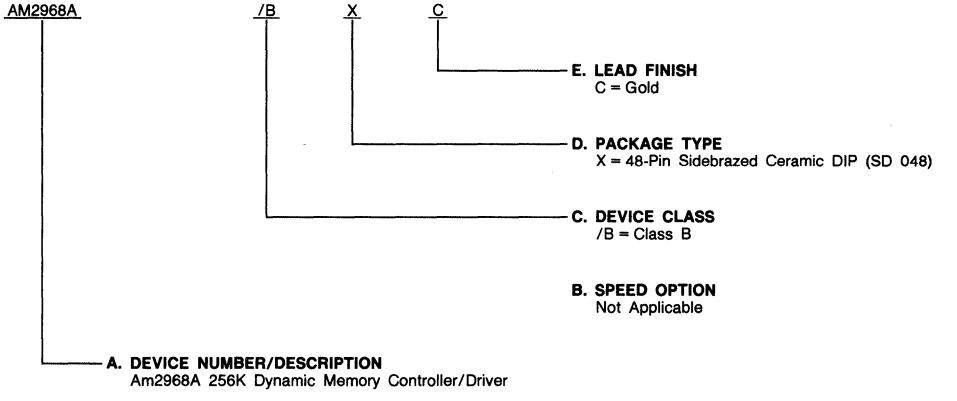
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

ORDERING INFORMATION (Cont'd.)

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. CPL (Controlled Products List) products are processed in accordance with MIL-STD-883C, but are inherently non-compliant because of package, solderability, or surface treatment exceptions to those specifications. The order number (Valid Combination) for APL products is formed by a combination of: **A. Device Number**

B. Speed Option (if applicable)
C. Device Class
D. Package Type
E. Lead Finish



Valid Combinations	
AM2968A	/BXC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consist of 1, 2, 3, 9, 10, 11

PIN DESCRIPTION

$A_0 - A_{17}$ Address Inputs (Inputs 18)
$A_0 - A_8$ are latched in as the nine-bit Row Address for the RAM. These inputs drive $Q_0 - Q_8$ when the Am2968A is in the Read/Write mode and MSEL is LOW. $A_9 - A_{17}$ are latched in as the Column Address, and will drive $Q_0 - Q_8$ when MSEL is HIGH and the DMC is in the Read/Write mode. The addresses are latched with the Latch Enable (LE) signal.

$\overline{CAS}_{0-3}$ Column Address Strobe (Outputs 4, Active LOW)
During normal Read/Write cycles the two select bits (SEL_0, SEL_1) determine which $\overline{CAS}_n$ output will go active following CASI going HIGH. When memory scrubbing is performed, only the $\overline{CAS}_n$ signal selected by $CNTR_0$ and $CNTR_1$ will be active (see $\overline{CAS}$ Output Function Table). For non-scrubbing cycles, all four $\overline{CAS}_n$ outputs remain HIGH.

CASI Column Address Strobe (Input, Active HIGH)
This input going active will cause the selected $\overline{CAS}_n$ output to be forced LOW.

$\overline{CS}$ Chip Select (Input, Active LOW)
This active-LOW input is used to select the DMC. When $\overline{CS}$ is active, the Am2968A operates normally in all four modes. When $\overline{CS}$ goes HIGH, the device will not enter the Read/Write mode. This allows more than one Am2968A DMC to control the same memory, thus providing an easy method for expanding the memory size.

LE Latch Enable (Input, Active HIGH)
This active-HIGH input causes the Row, Column, and Bank Select latches to become transparent, allowing the latches to accept new input data. A LOW input on LE latches the input data, assuming it meets the setup and hold time requirements.

MC_{0-1} Mode Control (Inputs 2)
These inputs are used to specify which of the four operating modes the DMC should be using. The description of the four operating modes is given in Table 1.

MSEL Multiplexer Select (Input)
This input determines whether the Row or Column Address will be sent to the memory address inputs. When MSEL is HIGH the Column Address is selected, while the Row Address is selected when MSEL is LOW. The address may come from either the address latch or refresh address counter depending on $MC_{0,1}$.

$\overline{OE}$ Output Enable (Input, Active LOW, Three-State)
This active-LOW input enables/disables the output signals. When $\overline{OE}$ is HIGH, the outputs of the DMC enter the high-impedance state. $\overline{OE}$ is only available on the surface-mount packages.

Q_{0-8} Address Outputs (Outputs 9)
These address outputs will feed the DRAM address inputs, and provide drive for memory systems up to 500 picofarads in capacitance.

$\overline{RAS}_{0-3}$ Row Address Strobe (Outputs 4, Active LOW)
Each one of the Row Address Strobe outputs provides a $\overline{RAS}_n$ signal to one of the four banks of dynamic memory. Each will go LOW only when selected by SEL_0 and SEL_1 and only after RASI goes HIGH. All four go LOW in response to RASI in either of the Refresh modes.

RASI Row Address Strobe (Input Active High)
During normal memory cycles, the decoded $\overline{RAS}_n$ output ($\overline{RAS}_0$, $\overline{RAS}_1$, $\overline{RAS}_2$, or $\overline{RAS}_3$) is forced LOW after receipt of RASI. In either Refresh mode, all four $\overline{RAS}_n$ outputs will go LOW following RASI going HIGH.

SEL_{0-1} Bank Select (Inputs 2)
These two inputs are normally the two higher-order address bits, and are used in the Read/Write mode to select which bank of memory will be receiving the $\overline{RAS}_n$ and $\overline{CAS}_n$ signals after RASI and CASI go HIGH.

FUNCTIONAL DESCRIPTION

Architecture

The Am2968A provides all the required data and refresh addresses needed by the dynamic RAM memory. In normal operation, the Row and Column addresses are multiplexed to the dynamic RAM by using MSEL, with the corresponding $\overline{RAS}_n$ and $\overline{CAS}_n$ signals activated to strobe the addresses into the RAM. High capacitance drivers on the outputs allow the DMC to drive four banks of 16-bit words, including a 6-bit checkword, for a total of 88 DRAMs.

Table 1. MODE CONTROL FUNCTION

MC_1	MC_0	Operating Mode
0	0	Refresh without Scrubbing. Refresh cycles are performed with only the Row Counter being used to generate addresses. In this mode, all four $\overline{RAS}_n$ outputs are active while the four $\overline{CAS}_n$ signals are kept HIGH.
0	1	Refresh with Scrubbing/Initialize. During this mode, refresh cycles are done with both the Row and Column counters generating the addresses. MSEL is used to select between the Row and Column counter. All four $\overline{RAS}_n$ go active in response to RASI, while only one $\overline{CAS}_n$ output goes LOW in response to CASI. The Bank Counter keeps track of which $\overline{CAS}_n$ output will go active. This mode is also used on system power-up so that the memory can be written with a known data pattern.
1	0	Read/Write. This mode is used to perform Read/Write cycles. Both the Row and Column addresses are latched and multiplexed to the address output lines using MSEL. SEL_0 and SEL_1 are decoded to determine which $\overline{RAS}_n$ and $\overline{CAS}_n$ will be active.
1	1	Clear Refresh Counter. This mode will clear the three refresh counters (Row, Column, and Bank) on the HIGH-to-LOW transition of RASI, putting them at the start of the refresh sequence. In this mode, all four $\overline{RAS}_n$ are driven LOW upon receipt of RASI so that DRAM wake-up cycles may be performed.

Table 2. ADDRESS OUTPUT FUNCTION

$\overline{CS}$	MC_1	MC_0	MSEL	Mode	MUX Output
0	0	0	X	Refresh without Scrubbing	Row Counter Address
	0	1	1	Refresh with Scrubbing	Column Counter Address
			0		Row Counter Address
	1	0	1	Read/Write	Column Address Latch
			0		Row Address Latch
	1	1	X	Clear Refresh Counter	Zero
1	0	0	X	Refresh without Scrubbing	Row Counter Address
	0	1	1	Refresh with Scrubbing	Column Counter Address
			0		Row Counter Address
	1	0	X	Read/Write	Zero
	1	1	X	Clear Refresh Counter	Zero

Table 3. $\overline{RAS}$ OUTPUT FUNCTION

RASI	$\overline{CS}$	MC_1	MC_0	SEL_1	SEL_0	Mode	$\overline{RAS}_0$	$\overline{RAS}_1$	$\overline{RAS}_2$	$\overline{RAS}_3$
0	X	X	X	X	X	X	1	1	1	1
1	0	0	0	X	X	Refresh without Scrubbing	0	0	0	0
		0	1	X	X	Refresh with Scrubbing	0	0	0	0
		1	0	0	0	Read/Write	0	1	1	1
				0	1		1	0	1	1
				1	0		1	1	0	1
				1	1		1	1	1	0
		1	1	X	X	Clear Refresh Counter	0	0	0	0
	1	0	0	X	X	Refresh without Scrubbing	0	0	0	0
		0	1			Refresh with Scrubbing	0	0	0	0
		1	0			Read/Write	1	1	1	1
		1	1			Clear Refresh Counter	0	0	0	0

Table 4. $\overline{CAS}$ OUTPUT FUNCTION

Inputs						Internal		Outputs			
CASI	$\overline{CS}$	MC_1	MC_0	SEL_1	SEL_0	$CNTR_1$	$CNTR_0$	$\overline{CAS}_0$	$\overline{CAS}_1$	$\overline{CAS}_2$	$\overline{CAS}_3$
1	0	0	0	X	X	X	X	1	1	1	1
		0	1	X	X	0	0	0	1	1	1
						0	1	1	0	1	1
						1	0	1	1	0	1
						1	1	1	1	1	0
		1	0	0	0	X	X	0	1	1	1
				0	1			1	0	1	1
				1	0			1	1	0	1
				1	1			1	1	1	0
		1	1	X	X	X	X	1	1	1	1
	1	0	0	X	X	X	X	1	1	1	1
		0	1	X	X	0	0	0	1	1	1
						0	1	1	0	1	1
						1	0	1	1	0	1
						1	1	1	1	1	0
		1	0	X	X	X	X	1	1	1	1
		1	1								
0	X	X	X	X	X	X	X	1	1	1	1

Input Latches

For those systems where addresses and data are multiplexed onto a single bus, the DMC has latches to hold the address information. The twenty input latches (Row, Column, and Bank Select) are transparent when Latch Enable (LE) is HIGH and will latch the input data meeting setup and hold time requirements when LE goes LOW. For systems where the processor has separate address and data buses, LE may be permanently enabled HIGH.

Refresh Counters

The two 9-bit refresh counters make it possible to support 128, 256, and 512 line refresh. External control over which type of refresh is to be performed allows the user maximum flexibility when choosing the refreshing scheme. Transparent (hidden), burst, synchronous or asynchronous refresh modes are all possible.

The refresh counters are advanced at the HIGH-to-LOW transition of RASI. This assures a stable counter output for the next refresh cycle.

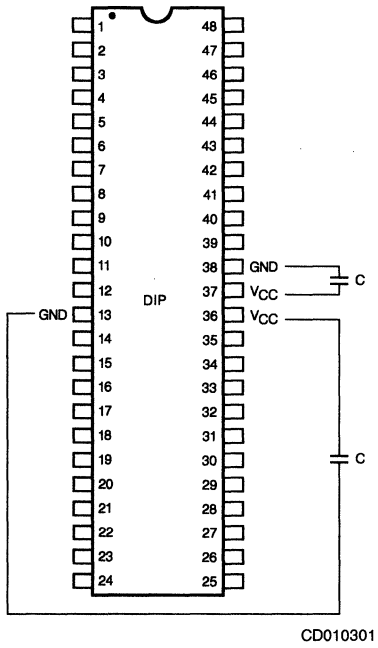
Refresh with Error Correction

The Am2968A makes it possible to correct single-bit errors in parallel with performing dynamic RAM refresh cycles. This "scrubbing" of memory can be done periodically as a background routine when the memory is not being used by the processor. In a memory scrubbing cycle ($MC_{1,0} = 01$), the Row Address is strobed into all four banks with all four $\overline{RAS}_n$ outputs going LOW.

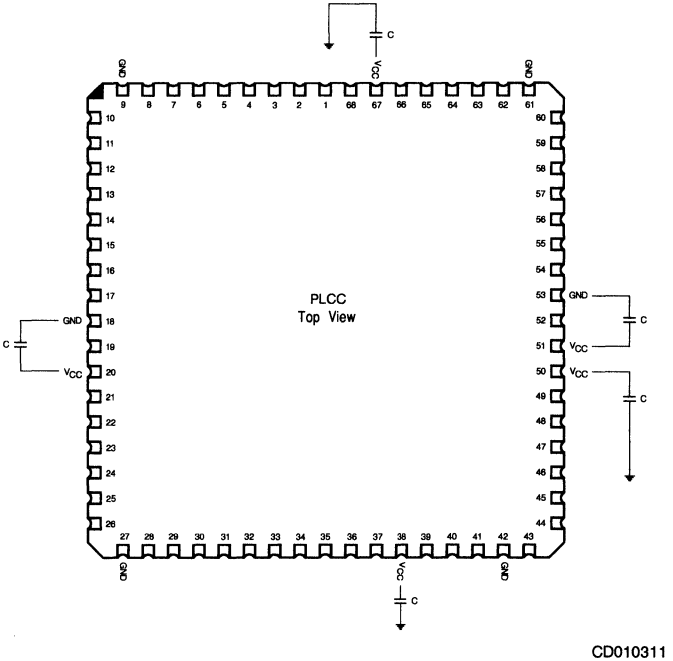
The Column Address is strobed into a single bank with the activated $\overline{CAS}_n$ output being selected by the Bank Counter. This type of cycle is used to simultaneously refresh the addressed row in all banks and read and correct (if necessary) one word in memory; thereby reducing the overhead associated with Error Detection and Correction. When doing refresh with memory scrubbing, both the Row and Column counters are multiplexed to the dynamic RAM address lines by using MSEL. Using the Refresh with Memory Scrubbing mode implies the presence of an error correcting facility such as the Am29C60A EDC unit. When doing refresh without scrubbing, all four $\overline{RAS}_n$ still go LOW but the $\overline{CAS}_n$ outputs are all driven HIGH so as not to activate the output lines of the memory.

Decoupling

Due to the high switching speeds and high drive capability of the Am2968A, it is necessary to decouple the device for proper operation. 1μF multilayer ceramic capacitors are recommended for decoupling (see Figure 1a). It is important to mount the capacitors as close as possible to the power pins (V_{CC}, GND) to minimize lead inductance and noise. A ground plane is recommended.



CD010301



CD010311

Figures 1a
Decoupling Connection Diagrams

V_{ONP}

The guaranteed maximum undershoot voltage of the Am2968A is −0.5 volts. V_{ONP} is measured with respect to ground (Fig. 1b). Note that the ground of the capacitive load must be the same as for the V_{CC} pin(s). As loading increases, V_{ONP} will approach zero.

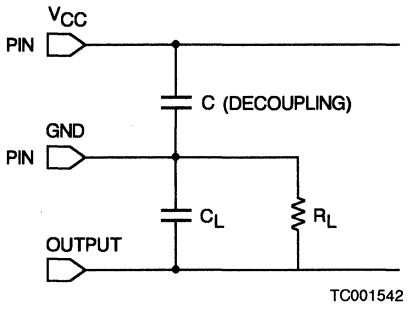
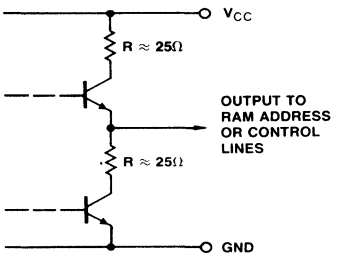


Figure 1b

The RAM Driver symmetrical output design offers significant improvement over a standard Schottky output by providing a balanced drive output impedance (≈25Ω both HIGH and LOW), and by pulling up to MOS V_{OH} levels. External resistors, not required with the RAM Driver, protect standard Schottky drivers from error causing undershoot but also slow the output rise by adding to the internal R.

The RAM Driver is optimized to drive LOW at maximum speed based on safe undershoot control and to drive HIGH with a symmetrical speed characteristic. This is an optimum approach because the dominant RAM loading characteristic is input capacitance.

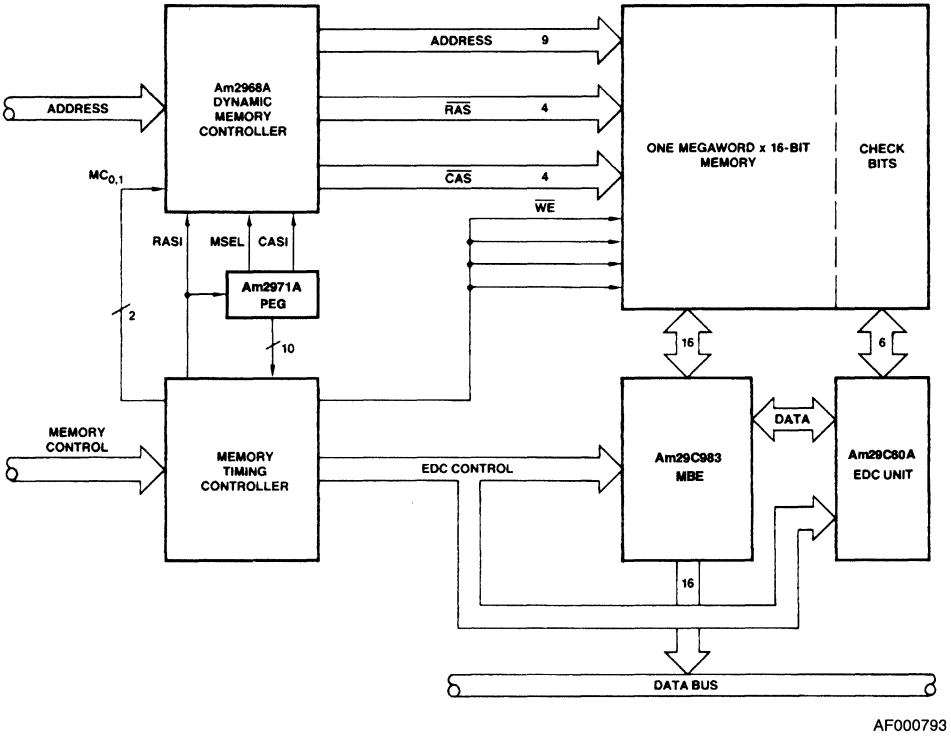
TYPICAL OUTPUT DRIVER



APPLICATIONS

Timing Control

To obtain optimum performance and maximum design flexibility, the timing and control logic for the memory system has been kept a separate function. For systems implementing Error Detection and Correction, the Am29C60A EDC unit may be used in 16-bit systems. The Am29C983 MBE serves as a data bus buffer.



AF000793

Figure 2a. One Megaword Dynamic Memory with Error Detection and Correction

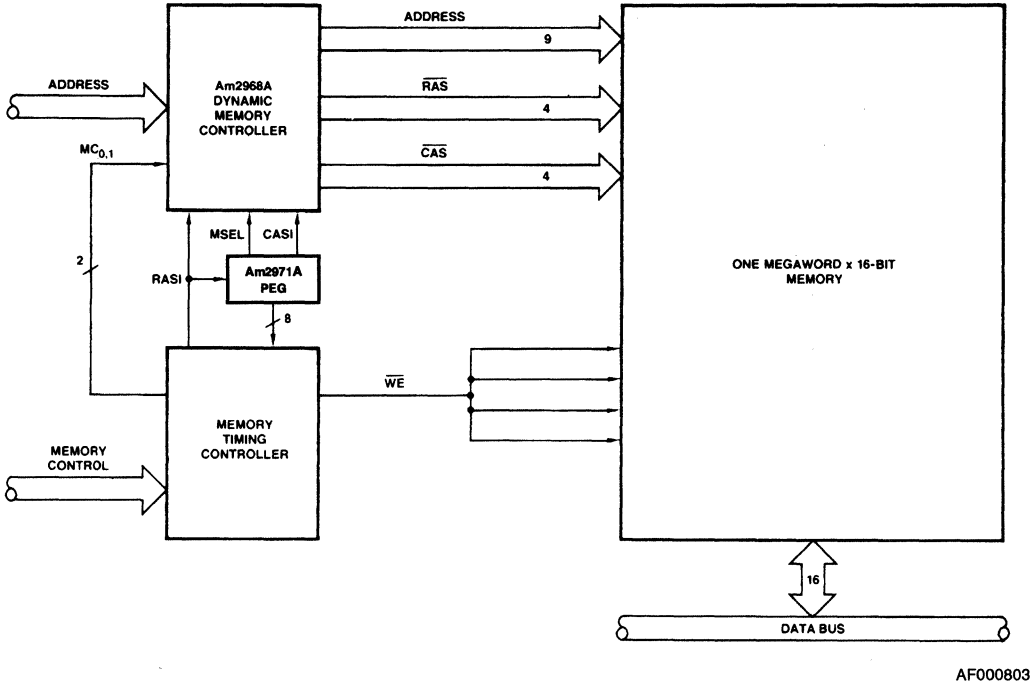


Figure 2b. One Megaword Dynamic Memory

Memory Expansion

With a 9-bit address path, the Am2968A can control up to one megaword memory when using 256K dynamic RAMs. If a larger memory size is desired, the DMC's chip select ($\overline{CS}$) makes it easy to double the memory size by using two Am2968As. Memory can be increased in one megaword increments by adding another DMC unit. A four-megaword memory system implementing EDC is shown in Figure 3.

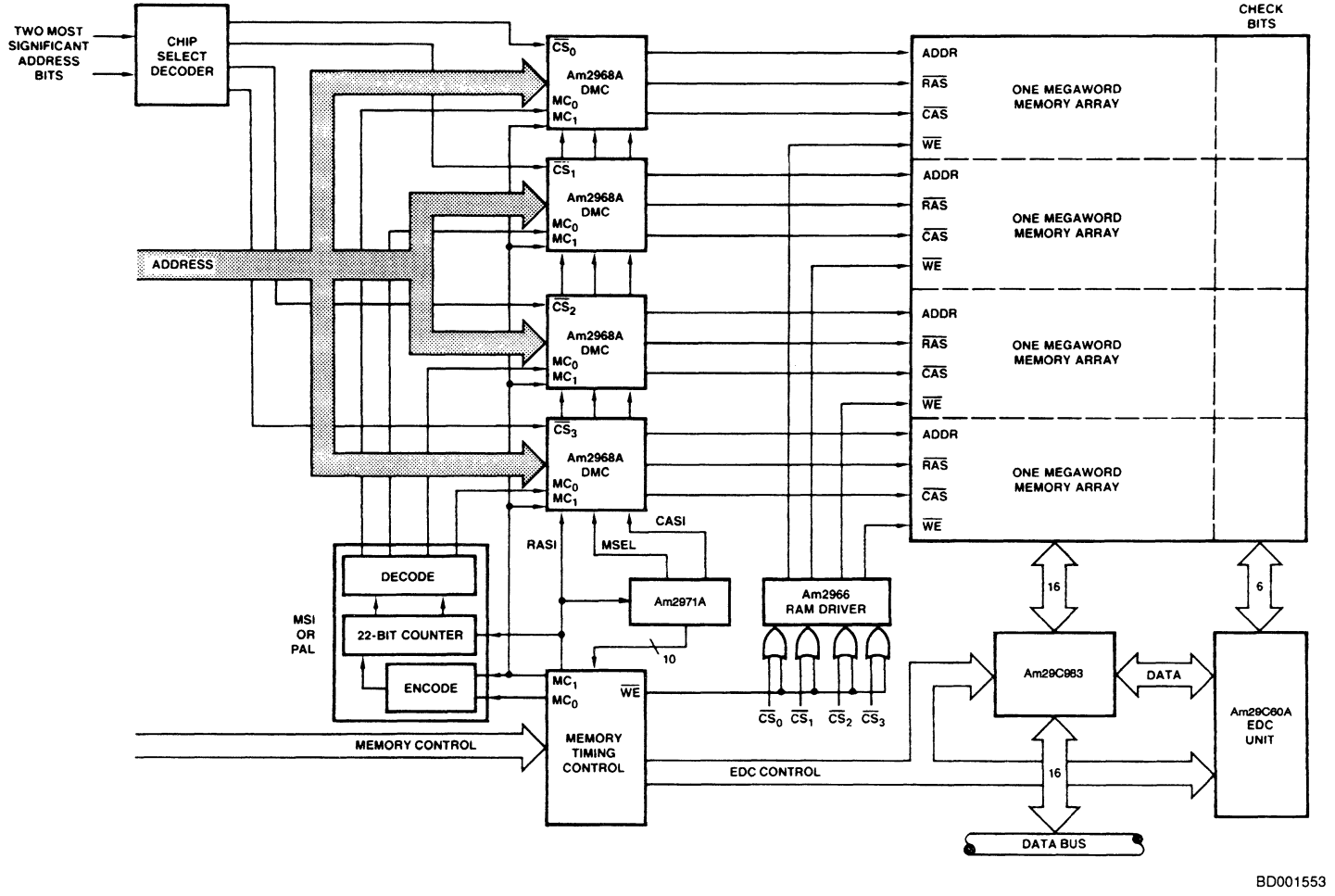


Figure 3. Four Megaword Error Correcting Memory

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Ambient Temperature with
Power Applied ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continuous ..... −0.5 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... −0.5 V to +V_{CC} max
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
T_A (Ambient) ..... 0 to +70°C
V_{CC} ..... 5.0 V ±10%
Min ..... 4.50 V
Max ..... 5.50 V

Military* (M) Devices
T_C (Case) ..... −55 to +125°C
V_{CC} ..... 5.0 V ±10%
Min ..... 4.50 V
Max ..... 5.50 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

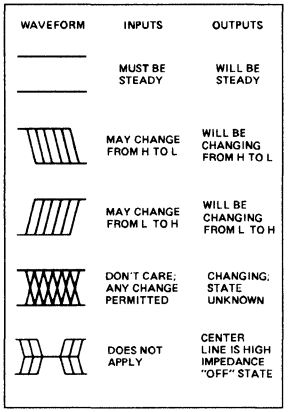
*Military Product 100% tested at T_C = 25°C, +125°C, and −55°C.

DC CHARACTERISTICS over operating ranges unless otherwise specified; Included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted.

Parameters	Descriptions	Test Conditions (Note 1)		Min.	Typ.	Max.	Units
V_{OH}	Output HIGH Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}, I_{OH} = −1 mA	COMM	2.7			Volts
			MIL	2.5			
V_{OL}	Output LOW Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 1 mA			0.5	Volts
			I_{OL} = 12 mA			0.8	
V_{IH}	Input HIGH Level	Guaranteed input logical-HIGH voltage for all inputs		2.0			Volts
V_{IL}	Input LOW Level	Guaranteed input logical-LOW voltage for all inputs				0.8	Volts
V_I	Input Clamp Voltage	V_{CC} = Min., I_{IN} = −18 mA				−1.2	Volts
I_{IL}	Input LOW Current	V_{CC} = Max., V_{IN} = 0.4 V				−400	μA
I_{IH}	Input HIGH Current	V_{CC} = Max., V_{IN} = 2.4 V				20	μA
I_I	Input HIGH Current	V_{CC} = Max., V_{IN} = 5.5 V				100	μA
I_{OZH}	Off-State Current	V_O = 2.4 V				50	μA
I_{OZL}	Off-State Current	V_O = 0.4 V				−50	μA
I_{OL}	Output Sink Current	V_{OL} = 2.0 V		45			mA
I_{SC}	Output Short-Circuit Current	V_{CC} = Max. (Note 2)		−60	−95	−275	mA
I_{CC}	Power Supply Current	V_{CC} = Max.	25°C, 5 V		230		mA
			0°C to +70°C			280	
			−55°C to 125°C			295	

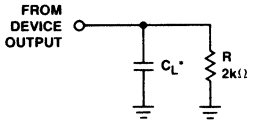
Notes: 1. For conditions shown as Min or Max, use the appropriate value specified under Operating Range for the applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.

KEY TO SWITCHING WAVEFORMS



KS000010

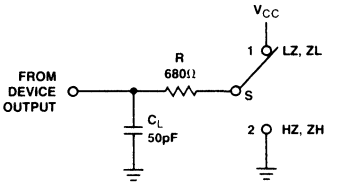
SWITCHING TEST CIRCUITS



TC000490

*t_{pd} specified at C_L = 50 and 500 pF

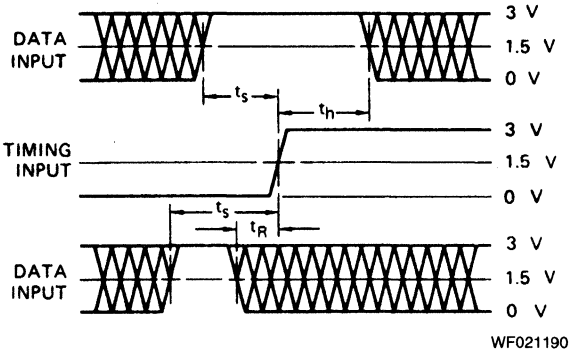
A. Capacitive Load Switching



TC000510

B. Three-State Enable/Disable (for PLCC only)

SWITCHING TEST WAVEFORMS

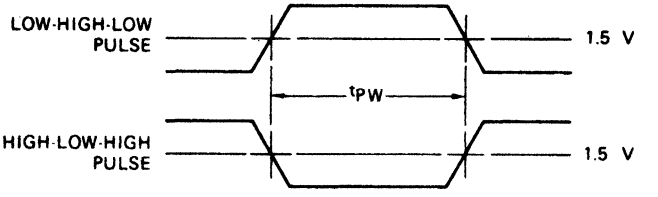


WF021190

A. Setup, Hold, and Release Times

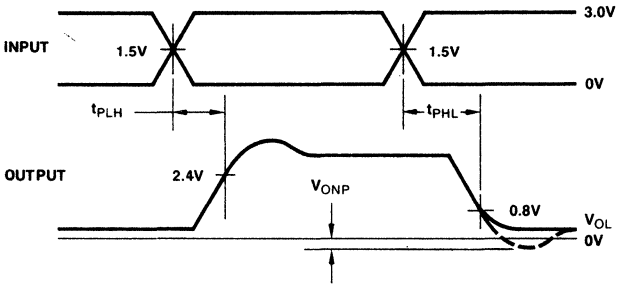
Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched are "don't care" condition.

SWITCHING TEST WAVEFORM (Cont'd.)



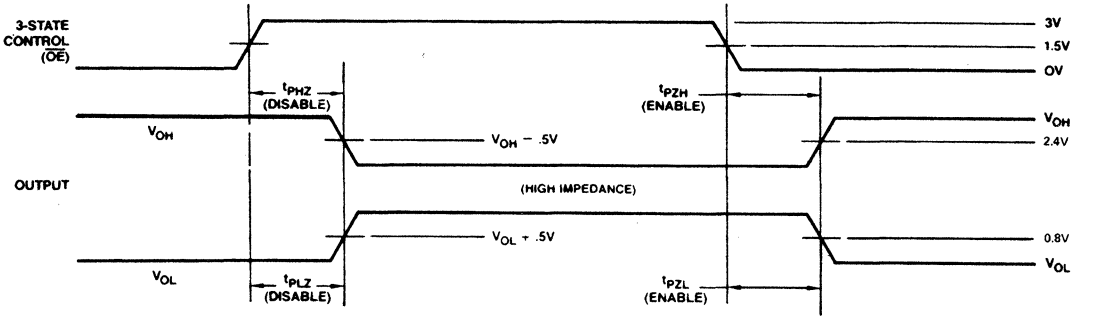
WF021210

B. Pulse Width



WF002122

C. Output Drivers Levels



WFR02941

Note: Decoupling is needed for all AC tests

D. Three-State Control Levels
(for Surface-mount packages only)

GENERAL TEST NOTES

Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Insure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.

2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.

3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5-8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by hundreds of millivolts momentarily.

4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 4$ V for AC tests.

5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.

6. Automatic tester hardware and handler add additional round trip A.C. delay to test measurements. Actual propagation delay testing may incorporate a correlation factor to negate the additional delay.

SWITCHING CHARACTERISTICS over operation range for C_L = 50 pF; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

Parameter Symbol		Parameter Description	Test Conditions		COMMERCIAL AND MILITARY			Units
					Typ.	Min.	Max.	
1	t_{PD}	A_n to Q_n Delay	Fig. A and C		12	3	20	ns
2	t_{PD}	RASI to $\overline{RAS}_n$			10	3	18	ns
3	t_{PD}	CASI to $\overline{CAS}_n$			8	3	17	ns
4	t_{PD}	MSEL to Q_n			12	3	20	ns
5	t_{PD}	MC_n to Q_n			15	5	24	ns
6	t_{PD}	LE to $\overline{RAS}_n$			15		25	ns
7	t_{PD}	LE to $\overline{CAS}_n$			14		24	ns
8	t_{PD}	MC_n to $\overline{RAS}_n$			14	3	21	ns
9	t_{PD}	MC_n to $\overline{CAS}_n$			12	3	19	ns
10	t_{PD}	LE to Q_n			15	5	25	ns
11	t_{PWL}	RASI, CASI			10	20		ns
12	t_{PWH}	RASI, CASI			10	20		ns
13	t_S	A_n to LE			1	5		ns
14	t_H	A_n to LE			1	5		ns
15	t_{PD}	$\overline{CS}$ to Q_n			16		23	ns
16	t_{PD}	$\overline{CS}$ to $\overline{RAS}_n$			12		20	ns
17	t_{PD}	$\overline{CS}$ to $\overline{CAS}_n$			11		19	ns
18	t_{PD}	SEL_n to $\overline{RAS}_n$			12		20	ns
19	t_{PD}	SEL_n to $\overline{CAS}_n$			11		18	ns
20	t_S	SEL_n to LE			1	5		ns
21	t_H	SEL_n to LE			1	5		ns
22	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 10)			10		17	ns
23	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 00, 01)			10		17	ns
24	t_{SKEW}	Q_n to $\overline{RAS}_n$			2		10	ns
25	t_{SKEW}	Q_n to $\overline{CAS}_n$			12		17	ns
26	t_H	MC_1 to RASI	C_L = 50 pF			5		ns
27	t_S	$\overline{CS}$ to RASI	C_L = 50 pF			5		ns
28	t_S	SEL_1 to RASI	C_L = 50 pF			5		ns
	t_{PLZ}	Output Disable Time	Fig. D and B	S = 1	15		22	ns
	t_{PHZ}	From LOW, HIGH (Note 1)		S = 2	13		20	ns
	t_{PZL}	Output Enable Time	Fig. D and B	S = 1	13		19	ns
	t_{PZH}	From LOW, HIGH (Note 1)		S = 2	14		21	ns
†	V_{ONP}	Output Undershoot Voltage (Note 2)	Fig. A and C		0		–0.5	V

Notes:

1. Three-state ($\overline{OE}$) applies only to PLCC package.
2. Not tested in production. Guaranteed by characterization data.

† = Not included in Group A testing

SWITCHING CHARACTERISTICS over operating range for C_L = 150 pF; included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted (Note 3)

Parameter		Description	Test Conditions	Typ.	COMMERCIAL AND MILITARY		Units
					Min.	Max.	
1	t_{PD}	A_n to Q_n Delay		16	9	24	ns
2	t_{PD}	RASI to $\overline{RAS}_n$		15	9	23	ns
3	t_{PD}	CASI to $\overline{CAS}_n$		14	9	22	ns
4	t_{PD}	MSEL to Q_n		17	9	26	ns
5	t_{PD}	MC_n to Q_n		18	10	28	ns
6	t_{PD}	LE to $\overline{RAS}_n$		20		28	ns
7	t_{PD}	LE to $\overline{CAS}_n$		19		27	ns
8	t_{PD}	MC_n to $\overline{RAS}_n$		19	9	25	ns
9	t_{PD}	MC_n to $\overline{CAS}_n$		17	9	23	ns
10	t_{PD}	LE to Q_n		20	10	27	ns
11	t_{PWL}	RASI, CASI		10	20		ns
12	t_{PWH}	RASI, CASI		10	20		ns
13	t_S	A_n to LE	Fig. A and C	1	5		ns
14	t_H	A_n to LE		1	5		ns
15	t_{PD}	$\overline{CS}$ to Q_n		19		27	ns
16	t_{PD}	$\overline{CS}$ to $\overline{RAS}_n$		14		22	ns
17	t_{PD}	$\overline{CS}$ to $\overline{CAS}_n$		14		22	ns
18	t_{PD}	SEL_n to $\overline{RAS}_n$		15		23	ns
19	t_{PD}	SEL_n to $\overline{CAS}_n$		14		22	ns
20	t_S	SEL_n to LE		1	5		ns
21	t_H	SEL_n to LE		1	5		ns
22	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 10)		10		15	ns
23	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 00,01)		10		17	ns
24	t_{SKEW}	Q_n to $\overline{RAS}_n$		2		8	ns
25	t_{SKEW}	Q_n to $\overline{CAS}_n$		15		17	ns
26	t_H	MC_1 to RASI	C_L = 150 pF		5		ns
27	t_S	$\overline{CS}$ to RASI	C_L = 150 pF		5		ns
28	t_S	SEL_1 to RASI	C_L = 150 pF		5		ns
†	V_{ONP}	Output Undershoot Voltage		0		−0.5	V

Note: 3. Production AC testing at 150pF load is not do ne. Performance at 150pF load is guaranted by characterization data and correlation to the 50pF and 500pF measurements.

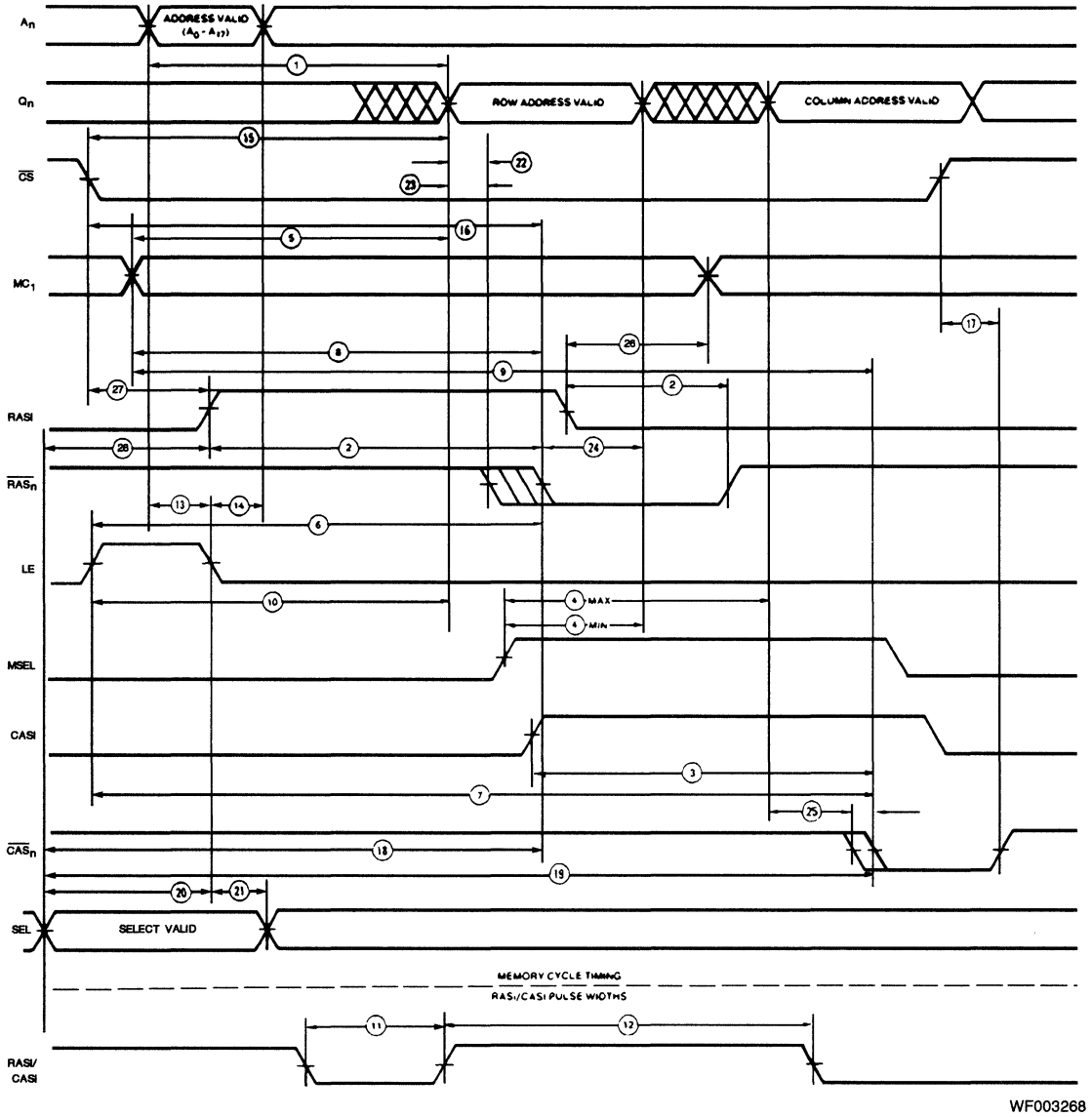
† = Not included in Group A tests

SWITCHING CHARACTERISTICS over operating range for C_L = 500 pF. Included in Group A, Subgroup 9, 10, 11 tests unless otherwise noted

Parameter		Description	Test Conditions	Typ.	COMMERCIAL AND MILITARY		Units
					Min.	Max.	
1	t_{PD}	A_n to Q_n Delay		29	12	40	ns
2	t_{PD}	RASI to $\overline{RAS}_n$		28	12	40	ns
3	t_{PD}	CASI to $\overline{CAS}_n$		26	12	37	ns
4	t_{PD}	MSEL to Q_n		29	12	42	ns
5	t_{PD}	MC_n to Q_n		30	12	44	ns
6	t_{PD}	LE to $\overline{RAS}_n$		32		46	ns
7	t_{PD}	LE to $\overline{CAS}_n$		31		45	ns
8	t_{PD}	MC_n to $\overline{RAS}_n$		30	12	40	ns
9	t_{PD}	MC_n to $\overline{CAS}_n$		28	12	40	ns
10	t_{PD}	LE to Q_n		32	12	46	ns
11	t_{PWL}	RASI, CASI		10	20		ns
12	t_{PWH}	RASI, CASI		10	20		ns
13	t_S	A_n to LE	Fig. A and C	1	5		ns
14	t_H	A_n to LE		1	5		ns
15	t_{PD}	$\overline{CS}$ to Q_n		30		45	ns
16	t_{PD}	$\overline{CS}$ to $\overline{RAS}_n$		27		40	ns
17	t_{PD}	$\overline{CS}$ to $\overline{CAS}_n$		26		38	ns
18	t_{PD}	SEL_n to $\overline{RAS}_n$		31		42	ns
19	t_{PD}	SEL_n to $\overline{CAS}_n$		28		41	ns
20	t_S	SEL_n to LE		1	5		ns
21	t_H	SEL_n to LE		1	5		ns
22	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 10)		10		18	ns
23	t_{SKEW}	Q_n to $\overline{RAS}_n$ (MC_n = 00,01)		10		18	ns
24	t_{SKEW}	Q_n to $\overline{RAS}_n$		2		8	ns
25	t_{SKEW}	Q_n to $\overline{CAS}_n$		15		20	ns
26	t_H	MC_1 to RASI	C_L = 500 pF		5		ns
27	t_S	$\overline{CS}$ to RASI	C_L = 500 pF		5		ns
28	t_S	SEL_1 to RASI	C_L = 500 pF		5		ns
†	V_{ONP}	Output Undershoot Voltage		0		-0.5	V

† = Not included in Group A tests

SWITCHING WAVEFORMS



WF003268

Am2968A Dynamic Memory Controller Timing

MEMORY CYCLE TIMING

The relationship between DMC specifications and system timing requirements are shown in Figure 4. T_1, T_2 and T_3 represent the minimum timing requirements at the DMC inputs to guarantee that RAM timing requirements are met and that maximum system performance is achieved.

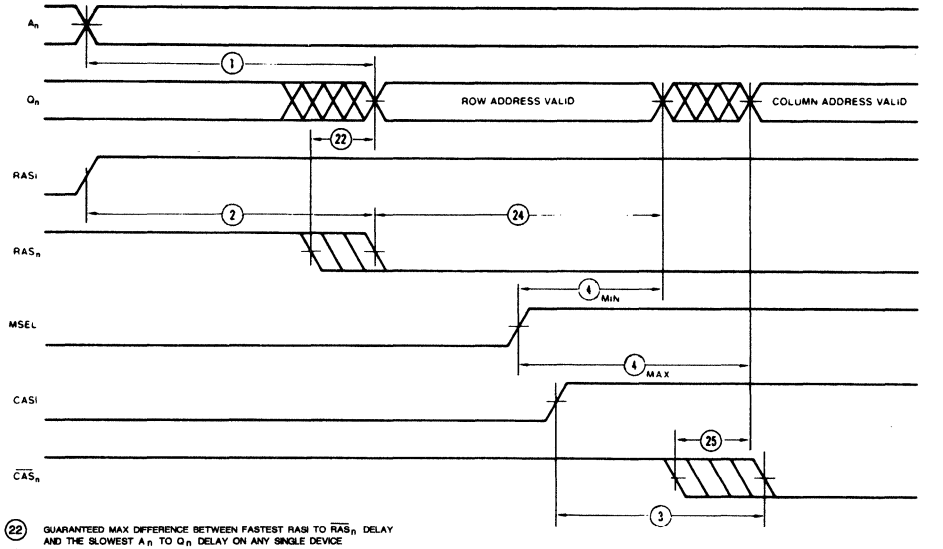
The minimum requirement for T_1, T_2 and T_3 are as follows:

T_1 Min. = $t_{ASR} + t_{22}$
T_2 Min. = $t_{RAH} + t_{24}$
T_3 Min. = $T_2 + t_{25} + t_{ASC}$

See RAM data sheet for applicable values for t_{RAH}, t_{ASC} and t_{ASR}.

Figure 4. Memory Cycle Timing

a. Specifications Applicable to Memory Cycle Timing (MC_n = 1, 0)



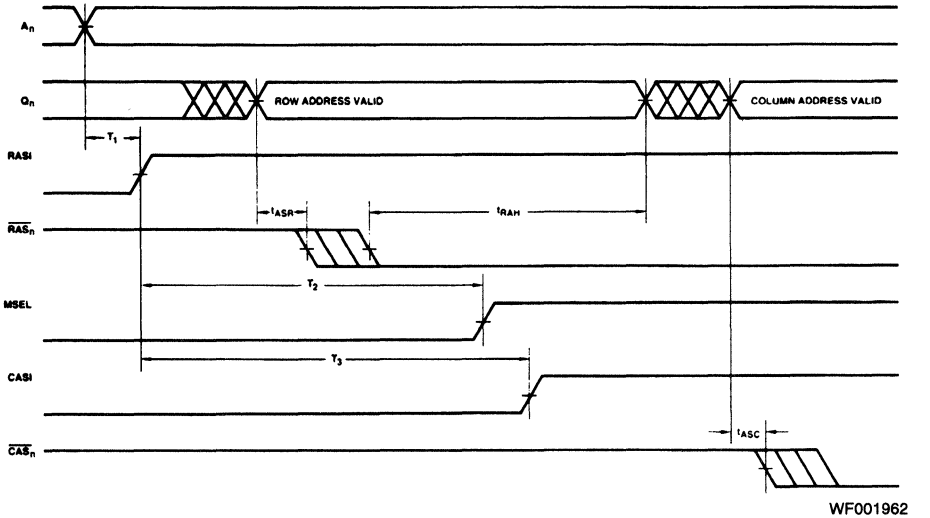
(22) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST RASI TO $\overline{RAS}_n$ DELAY AND THE SLOWEST A_n TO Q_n DELAY ON ANY SINGLE DEVICE

(24) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST MSEL TO Q_n DELAY AND THE SLOWEST RASI TO RAS_n DELAY ON ANY SINGLE DEVICE

(25) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST CASI TO $\overline{CAS}_n$ DELAY AND THE SLOWEST MSEL TO Q_n DELAY ON ANY SINGLE DEVICE

WF003285

b. Desired System Timing



WF001962

REFRESH CYCLE TIMING

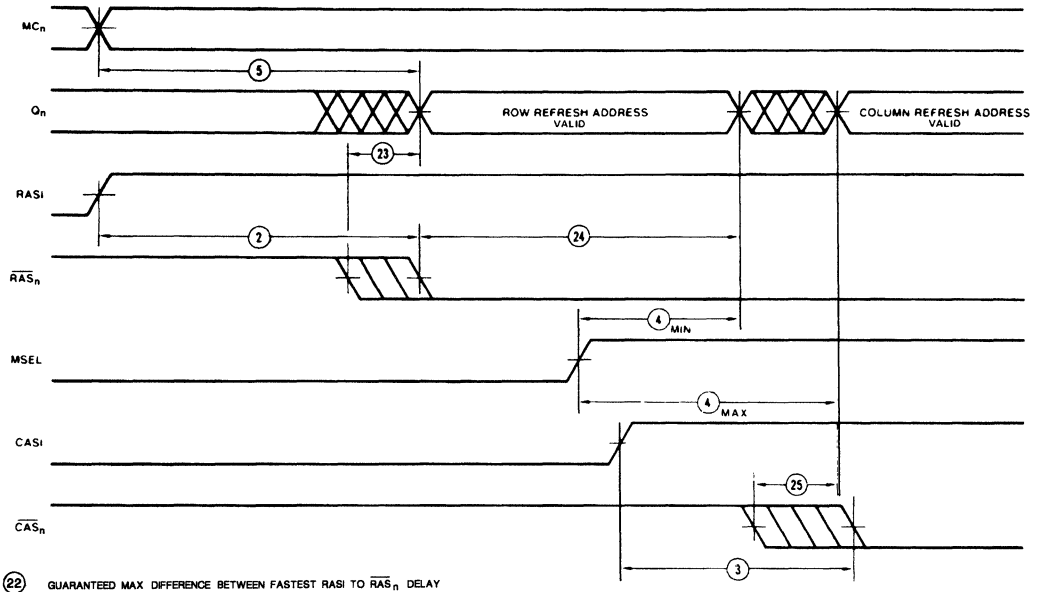
The timing relationships for refresh are shown in Figure 5.

T_4 minimum is calculated as follows:

T_4 Min. = $t_{ASR} + t_{23}$

Figure 5. Refresh Cycle Timing

a. Specifications Applicable to Refresh Cycle Timing (MC_n = 00, 01)



(22) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST RASI TO $\overline{RAS}_n$ DELAY AND THE SLOWEST A_n TO Q_n DELAY ON ANY SINGLE DEVICE

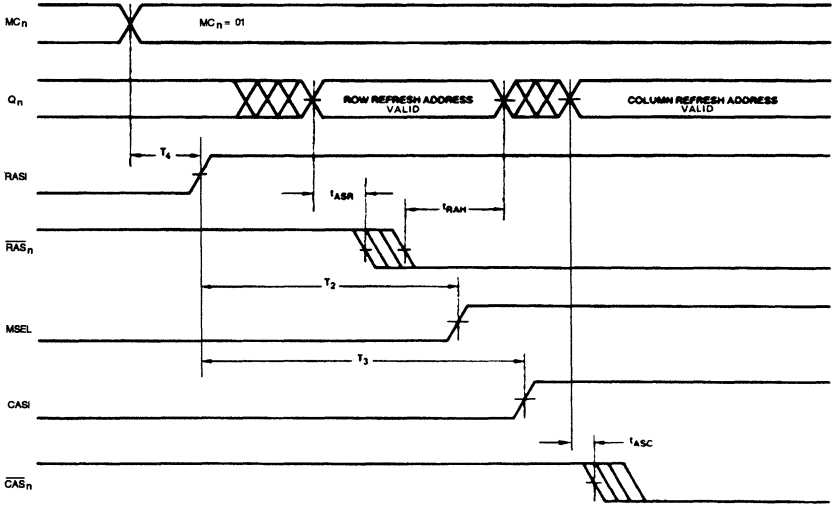
(23) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST RASI TO $\overline{RAS}_n$ DELAY AND THE SLOWEST MC_n TO Q_n DELAY ON ANY SINGLE DEVICE

(24) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST MSEL TO Q_n DELAY AND THE SLOWEST RASI TO $\overline{RAS}_n$ DELAY ON ANY SINGLE DEVICE

(25) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST CASI TO CAS_n DELAY AND THE SLOWEST MSEL TO Q_n DELAY ON ANY SINGLE DEVICE

WF003273

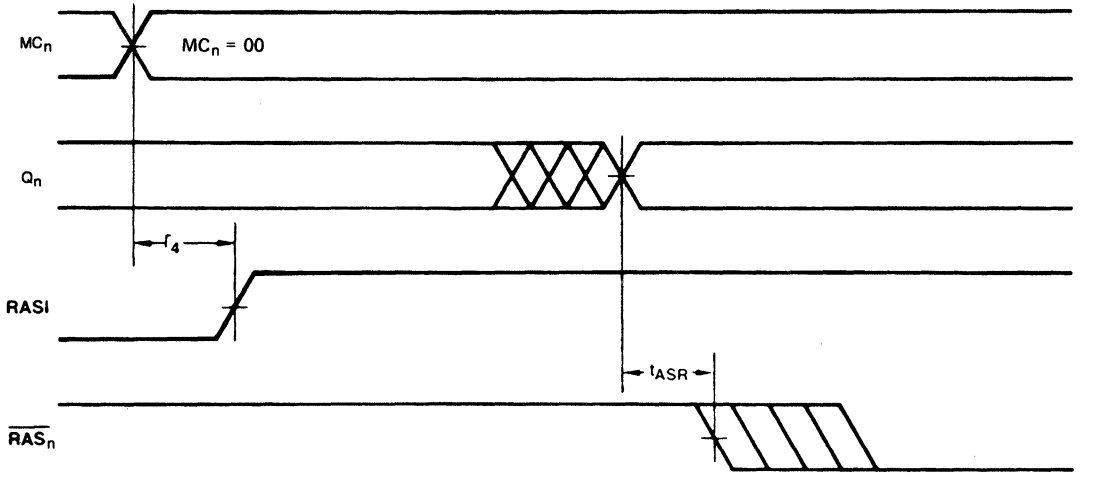
b. Desired Timing: Refresh with Scrubbing



WF001983

Figure 5. Refresh Cycle Timing (Cont'd.)

c. Desired Timing: Refresh without Scrubbing



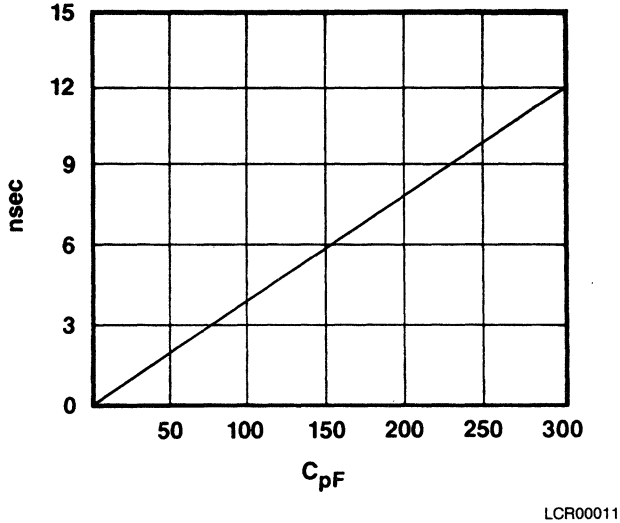
WF003253

NANOSECONDS VERSUS PICOFARADS

To help calculate how the AC performance of the DMC will vary for capacitive loads other than 50, 150, and 500pF refer to the table below.

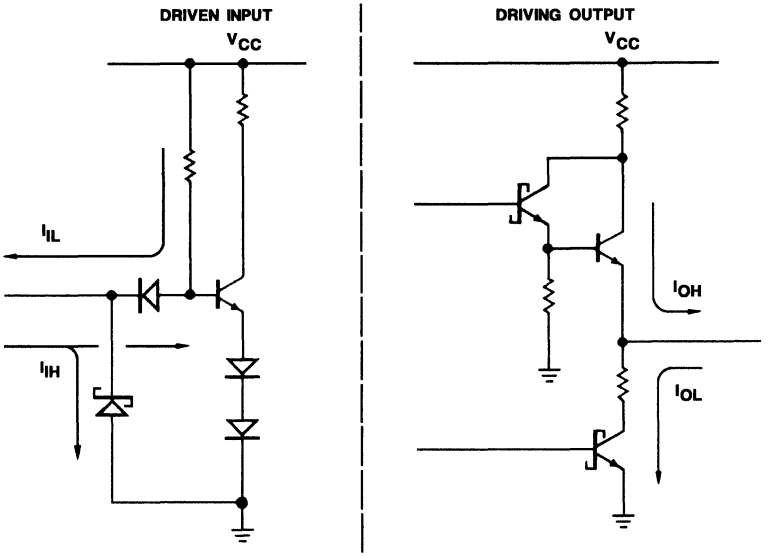
Example: For a system capacitive load of 250pF, add the delay associated with 100pF from the table to the AC specs done at 150pF.

Change in Propagation Delay versus Loading Capacitance (TYPICAL)



LCR00011

INPUT/OUTPUT CURRENT DIAGRAM



IC000791

Final

Am2971A

Enhanced Programmable Event Generator (PEG)™

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

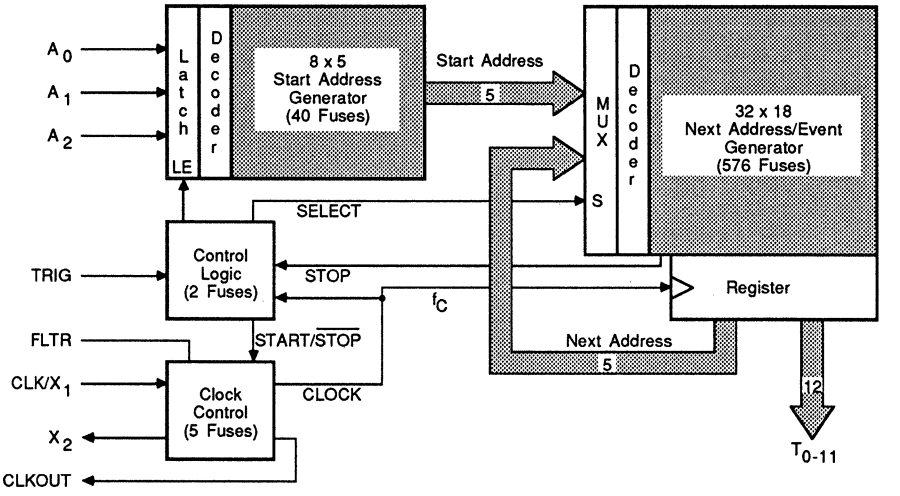
- **Generates arbitrarily defined output sequences on 12 parallel outputs**
- **Timing resolution down to 10 ns**
- **Internal frequency-multiplying Phase-Locked Loop (PLL)**
- **Crystal-controlled on-chip oscillator**
- **Programmable trigger polarity and STOP function**

GENERAL DESCRIPTION

The PEG is a versatile source of 12 simultaneous timing sequences. It can act as a digital substitute for multiple tapped delay lines or as a general-purpose user-programmable waveform generator.

Timing is derived from an external TTL source or an on-chip crystal oscillator, combined with an on-chip programmable frequency-multiplying PLL and clock divider. This achieves excellent timing resolution, down to 10 ns, from low-cost stable frequency sources of 10 MHz or less. The PEG uses platinum-silicide fuse technology and is programmed similar to any other AMD PROM.

BLOCK DIAGRAM

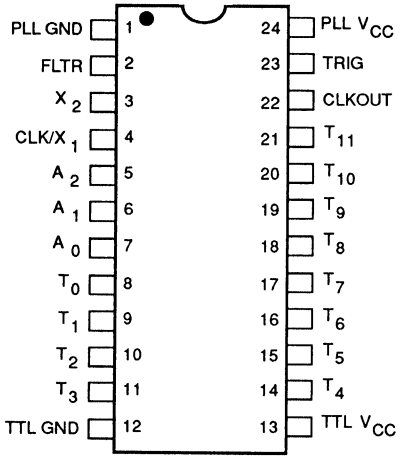


05280-001A

Publication# 05280 Rev. E Amendment /0
Issue Date: January 1990

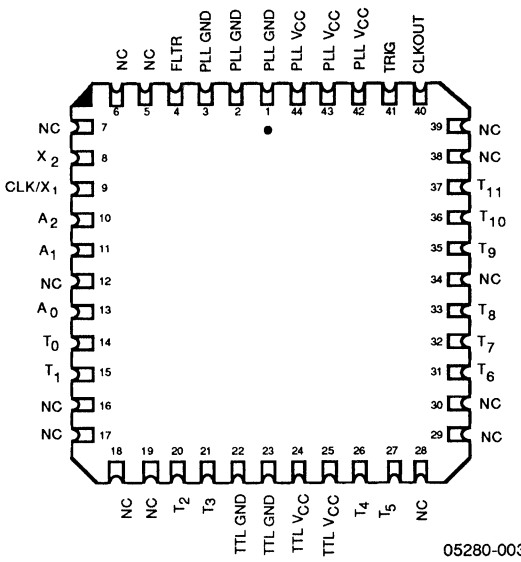
CONNECTION DIAGRAMS

DIP



05280-002A

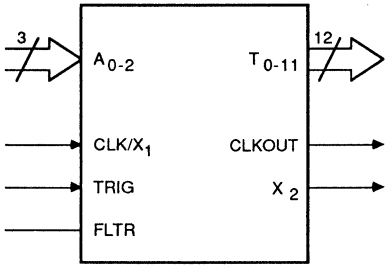
LCC*



05280-003A

* Top View, JEDEC type-C package (NC = No Connection)

LOGIC SYMBOL



Approximate Gate Count: 100

Die Size: 0.173" x 0.257"

05280-004A

THERMAL CHARACTERISTICS

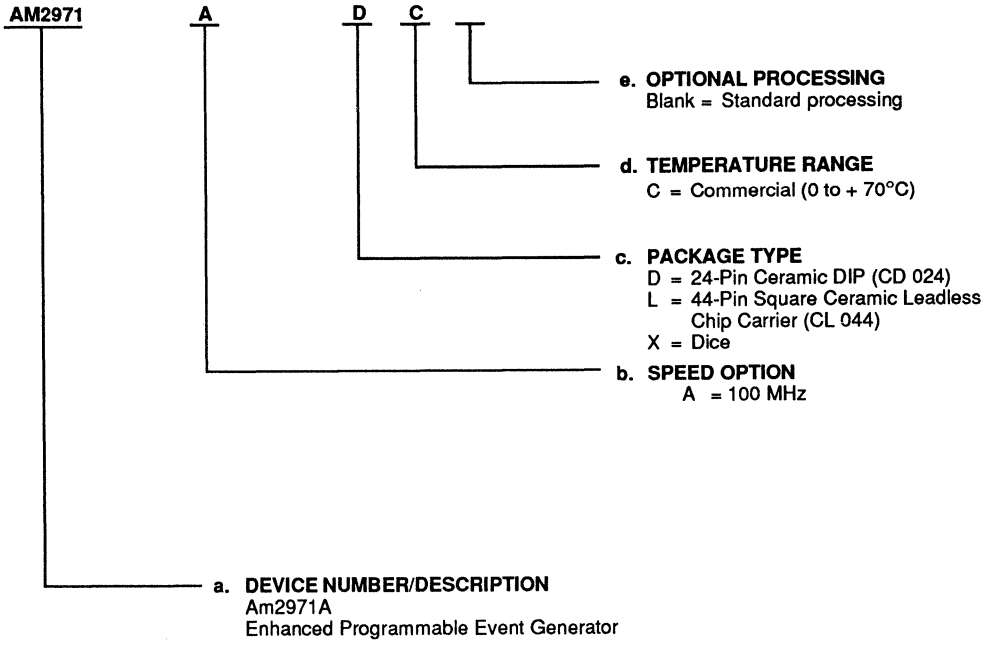
	24-pin Ceramic DIP	44-pin Ceramic LCC	Unit
θ_{JC} Max.	11	15	°C/W
θ_{JA} Max.	49	75	°C/W

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM2971A	DC, LC, XC

Valid Combinations

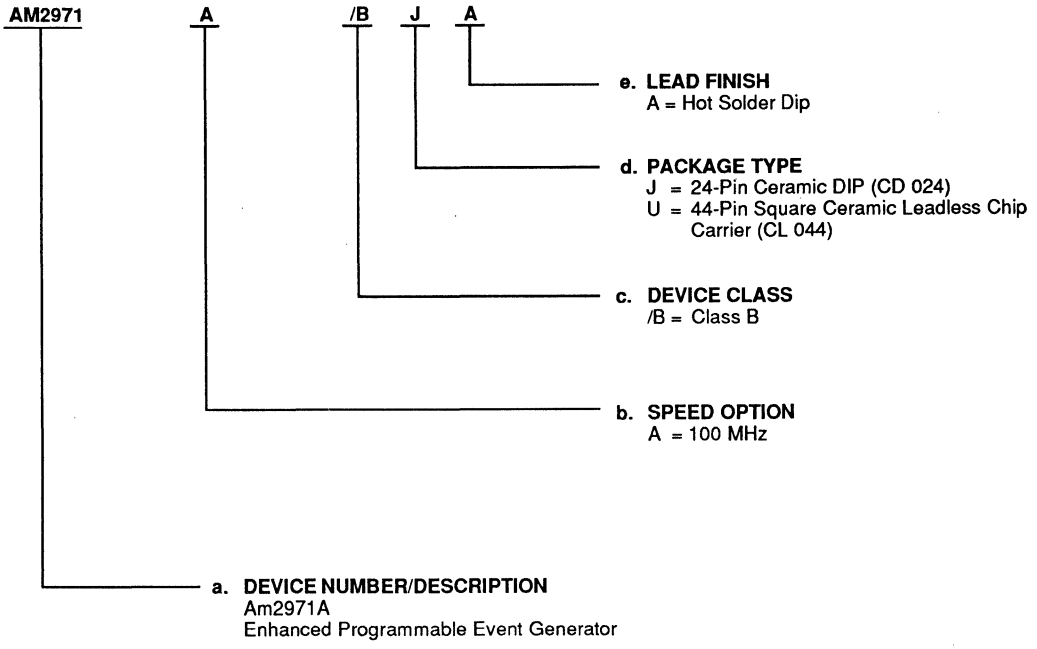
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM2971A	/BJA, /BUA

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

A_0–A_2
Addresses (Inputs)

These three bits access the Start Address Generator which contains eight user-programmed start locations. Each cycle starts at the location pointed to by the Start Address Generator word selected by the A_0–A_2 inputs. In the Program Mode, these inputs are unused and may be allowed to float.

CLK/X_1 and X_2
Clock/Crystal (Input/Output)

A TTL-level clock may be applied to the CLK/X_1 input, with the X_2 output left floating, or an AT-cut parallel resonant crystal may be connected between these two pins.

CLKOUT
Output Clock (Output)

CLKOUT is a clock output pin which may be used for system reference. The output frequency for CLKOUT (f_O) is fuse-programmable to be either 0.5, 1, or 2 times the input frequency. This output is not valid in the Bypass Mode. In the Program Mode a high-voltage pulse is applied to CLKOUT to blow selected fuses.

FLTR
Filter

This pin is used to connect a 0.47-µF filter capacitor between the Phase-Locked Loop and ground when an external crystal is used or the PLL is selected. When clocking the PEG with an external TTL source greater than 10 MHz in the Bypass Mode, this pin should be tied LOW. See Figure 8 for proper device decoupling with the PLL Bypassed.

T_0–T_{11}
Timing Outputs (Outputs; Active HIGH)

These are the twelve timing outputs which follow a user-programmed timing pattern. They are registered for glitch-free operation. In the Program and Verify Modes, T_0–T_{10} function as address inputs to access each individual fuse. T_0–T_5 serve as Row Address inputs, and T_6–T_{10} serve as Column Address inputs (see Table 6). After power-up, these outputs are all LOW. T_{11} functions as data input in the Program Mode and as data output in the Verify Mode.

TRIG
Trigger (Input)

The timing cycle of the PEG can be started by either the rising or falling edge of the start (TRIG) pulse; the polarity is defined as a fuse option (fuse #621) in the TRIGGER POLARITY block. The trailing edge of the start (TRIG) pulse stops the timing sequence if the STOP TRIG fuse (fuse #622) is left unprogrammed (0).

POWER, GROUND
TTL/PLL Power Pair

There are two sets of V_{CC} and ground pins. One power pair is used by the PLL (Phase-Locked Loop) and the internal ECL circuitry. The other power pair is used by the remainder of the chip (TTL). Surface-mount packages have additional supply connections. All power and grounds must be connected regardless of mode of operation.

FUNCTIONAL DESCRIPTION

The leading edge of the trigger pulse (polarity is fuse-programmable) causes the continuously running internal clock to step through the on-chip PROM addresses, starting at one of eight fuse-programmed locations selected by the A_{0-2} inputs.

Each addressed PROM location generates a fuse-programmed 12-bit pattern on the T_{0-11} outputs, and internally generates the fuse-programmable next PROM address as well as a fuse-programmable STOP bit, if desired. Since there is no program counter, there is an almost infinite number of ways of programming the PEG for any desired output pattern. The user will most likely choose an ascending address sequence, but this is only one of many arbitrary choices.

The address sequence can loop but cannot execute conditional jumps.

The sequence of operations stops either as a result of the trailing edge of the trigger pulse (if so enabled by a fuse) or by the programmable STOP bit. A new sequence can only be started after the previous sequence has stopped.

The internal clock frequency, f_C (see Operational Description for an explanation of all internal and external signal frequencies), is derived from and is proportional to the frequency on the X_1 input, which is either an external TTL signal or the resonant frequency of a crystal connected between X_1 and X_2. Controlled by programmable fuses, the frequency on X_1 is either used directly or is first multiplied by a factor of 1.25, 2.5, 5, or 10 to generate the internal clock frequency. A clock output is available; its frequency (fuse-programmable) is either half, double, or equal to the frequency on X_1. This output is not valid in the Bypass Mode.

Operational Description

Frequency Definitions

To avoid confusion, the definitions of the various frequencies associated with the PEG are given below:

f_I = This is the user's Input Frequency into the CLK/X_1 pin.

f_O = This is the PEG's Output Frequency at the CLKOUT pin. A CLKOUT signal is valid only when the PLL is used.

f_C = This is the Internal Clock Frequency, which is gated into the event generator state machine. When a timing sequence has been stopped, there is no f_C.

f_A = This is the Internal Altered Input Frequency, which is equivalent to f_C in value. This frequency is always generated, but it is not gated to the state machine (thus becoming f_C) unless a sequence is started.

f_{PLL} = This is the Phase-Locked Loop Frequency (f_I x 5 or f_I x 10).

TRIG-to-Output Delay

Operation of the PEG is initiated by a transition (of programmed polarity) on the TRIG input. This transition starts a series of internal events which lead to the clocking of the T_0–T_{11} output registers and to programmed changes on these outputs.

There are two possible conditions:

If the TRIG transition is synchronous with the frequency on X_1 (i.e., X_1 is a TTL clock signal and TRIG is synchronized with it), then the TRIG-to-output delay can be well-controlled, but the designer must analyze the timing and programming relationship carefully, as described below.

In the more normal case where TRIG is asynchronous to the frequency on X_1, the TRIG-to-output delay can be described very simply, but has an unavoidable uncertainty of one internal clock period.

Trigger Asynchronous

Start Delay

The delay from the active trigger edge to the first possible change of output pattern on T_0–T_{11} is the sum of:

1) Propagation delays in the trigger circuit plus output driver,
2) Up to one clock period of f_A due to the asynchronous relationship between TRIG and f_A, and
3) One clock period of f_A (used internally to prevent metastable operation).

Stop Delay

A timing sequence can be stopped either by the trailing edge of the start (TRIG) pulse (if so enabled by leaving fuse #622 unprogrammed) or by a programmed STOP bit in the Next Address/Event Generator fuse block.

The timing sequence stops when it detects either or both of these conditions. If stopped by a programmed STOP bit, the outputs remain at the level that is programmed in the same address location as the STOP bit. If stopped from the TRIG input, there is a delay equivalent to the starting delay.

Trigger Synchronous

Detailed Analysis of the Start and Stop Timing Sequences

The operation of the TRIG function can best be described by a synchronous state machine which uses f_A as the clock. All transitions occur on the rising clock edge. Figure 1 is the state diagram, Figure 2 the equivalent timing diagram. The state diagram uses the terms "active" and "inactive" edges of TRIG, since the actual polarity of TRIG is user-programmable.

State A is the idle state, after a reset or after operation has stopped for more than two periods of f_A. In state A, Select and f_C are HIGH (i.e., the Next Address/Event Generator PROM is addressed from the Start Address Generator PROM), but the output registers are not clocked; they retain their previous value.

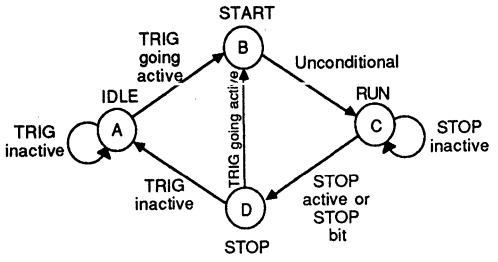
When TRIG goes active, the next rising edge of f_A forces the state machine into state B and causes f_C to be equal to f_A. The output registers are still not clocked.

The next rising edge of f_A forces the state machine into state C, clocks the output from the Next Address/Event Generator PROM into the output register and forces Select LOW. Subsequent cycles use the registered "next address" output as an address to the Next Address/Event Generator PROM. State C lasts until a STOP condition is encountered.

When a STOP condition is encountered, the next rising edge of f_C forces the state machine into state D. If TRIG goes active while the state machine is in state D, the next rising edge of f_A will cause it to go to state B; otherwise it will go to state A.

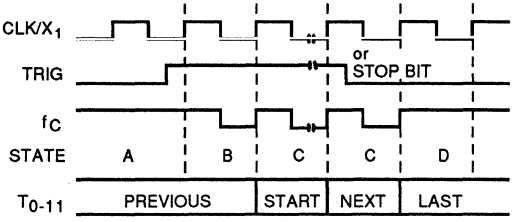
The shortest possible sequence is A-B-C-D-A, staying one f_A period in states B, C, D.

The fastest possible retrigger goes C-D-B-C, staying one f_A period in states D and B, at frequencies below 50 MHz. At frequencies greater than 50 MHz, the PEG will stay two clock periods in state D.



05280-005A

Figure 1. State Machine Diagram



05280-006A

Figure 2. State Machine Timing (Bypass Mode)

Start And Stop Timing Synchronous With f_I

The following paragraphs describe in detail the timing relationship and requirements between TRIG and the output changes on T_0–T_{11}, provided that TRIG is synchronous with f_I.

Bypass Mode ($f_C = f_I$)

TRIG must change from inactive to active for a specified setup time before the rising edge on X_1 and must stay active at least until 5 ns after the next subsequent rising edge on X_1. If the inactive-going edge of TRIG is programmed as a STOP condition, the shortest legitimate TRIG pulse will cause the state machine to cycle from state A to B to C to D, clocking the output register twice (first with the code accessed by the start address, then with the code accessed by the "next address" bits). If TRIG lasts additional X_1 clock periods, the state machine will spend this additional time in state C and the PEG will step through more codes.

PLL x5 or x10 Mode

When the f_{PLL} output is used as f_A, either 5 or 10 internal clock cycles occur for every f_I clock cycle. Since f_A is phase-synchronized to the rising edge of X_1, the timing analysis is very similar to the previous one, with f_A substituted for periods on X_1. The setup times associated with f_I also apply to f_A, since f_A is in phase with f_I. If the TRIG setup time with respect to X_1 exceeds one f_{PLL} period, then the state machine may trigger on the earlier internal clock. Because of the spread of guaranteed device parameters, an uncertainty is introduced. Thus, using the "STOP-from-TRIG" feature to generate a predetermined number of output sequences could result in an incorrect number of transitions (either more or less).

÷2 or ÷4 Mode

In this mode the rising transitions of f_A are no longer uniquely related to the rising transitions of f_I or f_{PLL}, since the ÷2 counter can be in either of two starting states and the ÷4 counter can be in any one of four possible starting states.

The START and STOP sequences, therefore, have an additional unpredictable delay of either a 0 or 1/2 period of f_A (if ÷2 is chosen), or either a 0, 1/4, 1/2, or 3/4 period of f_A (if ÷4 if chosen).

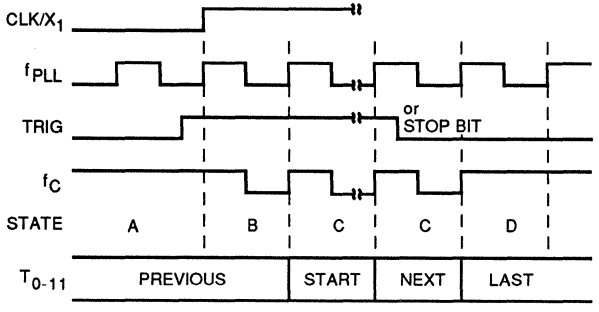


Figure 3. PLL Mode

05280-007A

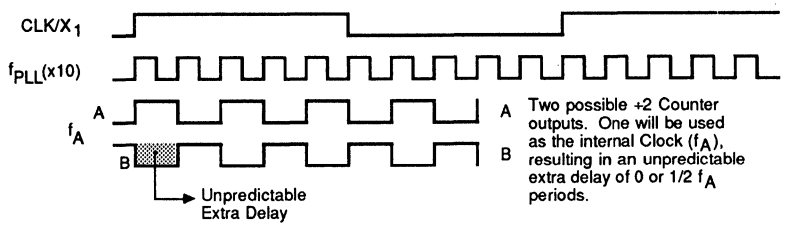


Figure 4. ÷2 Mode 05280-008A

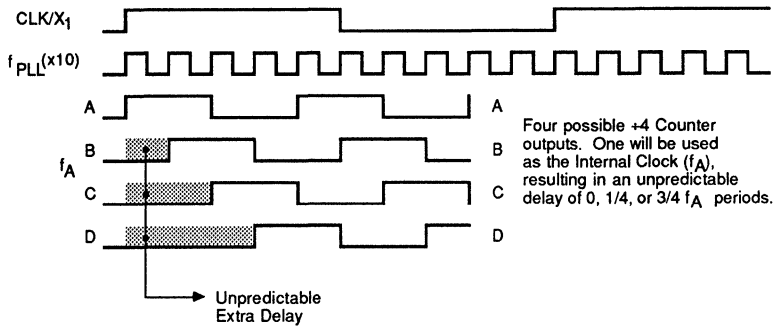


Figure 5. ÷4 Mode 05280-009A

Output Skew and Jitter

The twelve timing waveform outputs (T_0–T_{11}) are synchronized internally in an output register in order to minimize output skew.

The guaranteed maximum value for the remaining skew is specified by parameters 6–10 in the Switching Characteristics Table (depending upon the PEG version used and the number and type of transitions).

See Switching Characteristics for tighter skew specifications of certain outputs. More closely matched outputs should be used for more critical timing.

Any oscillator, and especially a Phase-Locked Loop, exhibits a certain amount of jitter — random phase modulation of the internal clock. Such jitter affects all outputs together (synchronously).

Jitter is typically less than ±1.0 ns for the ceramic DIP and Flatpack, and less than ±0.5 ns for the LCC package.

Output Event Resolution

Each of the twelve timing waveform outputs (T_0–T_{11}) can be programmed to change on any rising edge of the internal clock frequency (f_C), with the following restrictions:

The Am2971A has a maximum operating frequency of 100 MHz with a TTL source, all outputs in use. An output resolution of 10 ns for transitions of the same output or between transitions of different outputs is obtainable when a maximum of nine outputs (any nine) are switched simultaneously. If more than nine outputs are used, resolution within the same output is 20 ns, and 10 ns between outputs. If a crystal is used to clock the Am2971A, all outputs may be used, but only six outputs (any six) can be programmed to switch simultaneously. The output resolution with six outputs switching can be 10 ns within or between outputs. At internal frequencies less than 85 MHz, there are no programming restrictions (all outputs can switch simultaneously).

Table 1. Output Resolution (Between Successive Transitions of the Same Output)

Number of outputs switching simultaneously		12, 11, 10	9, 8, 7	6, 5, 4, 3, 2, 1
TTL Clock Source	PLL	20 ns	10 ns	10 ns
	Bypass			
Crystal Clock Source		Not Allowed	Not Allowed	10 ns

Oscillator

The Am2971A contains an inverting linear amplifier which can be used as a crystal oscillator. Various types of crystals are available, and the manufacturers' literature should be consulted to determine the appropriate type.

Crystal frequency varies with load capacitance. It is therefore important to match the load specified by the crystal manufacturer for a standard crystal (usually 32 pF), or to specify the load when ordering a special crystal.

The circuit of a typical 1st-harmonic oscillator is shown in Figure 6. The crystal load is comprised of the two 68-pF capacitors effectively in series. This 34 pF approximates the standard 32-pF crystal load. If a closer match is required, one of the capacitors should be replaced with a parallel combination of a fixed capacitor and a trimmer.

A typical crystal specification for use in this circuit is:

Frequency Range: 2–20 MHz
Resonance: AT, Parallel Resonant Mode
Load: 32 pF
Stability: to match system requirements

In order to eliminate stray pick-up, it is good practice to ground the case of the crystal and to keep all connections as short as possible.

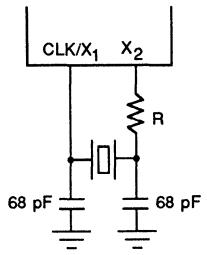
At fundamental frequencies below 6 MHz, the crystal might accidently operate in 3rd-harmonic mode. To prevent this, a resistor should be added in series with the X_2 pin as shown in the circuit diagram (Figure 6).

The resistor value should equal the impedance of C:

$$R = XC = \frac{1}{2\pi f^{*}C} = \frac{2342\ \Omega}{f(MHz)}$$ Example: R = 390 Ω for 6 MHz

Design Considerations

1) Oscillator external connections must be less than 1" long—wirewrap is not recommended.
2) V_{CC} and GND connections to power plane should be less than 1/2" long.
3) Effective supply decoupling over a broad frequency range is mandatory (Reference Figures 7 and 8).



05280-010A

Figure 6. Am2971A Crystal Oscillator Circuit

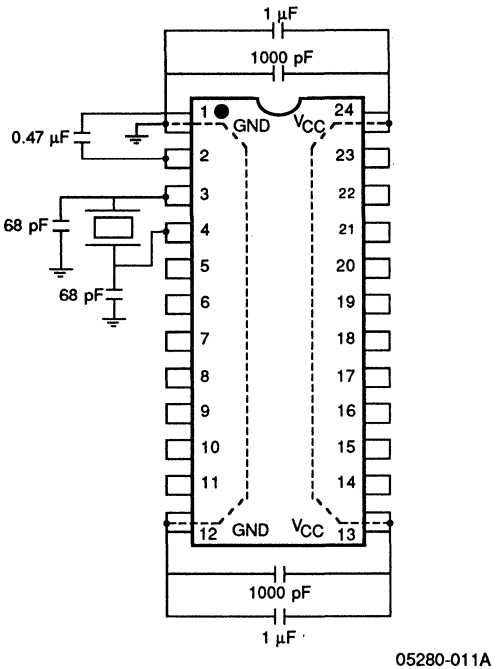


Figure 7. Am2971A Recommended Layout and Decoupling (Crystal Input)

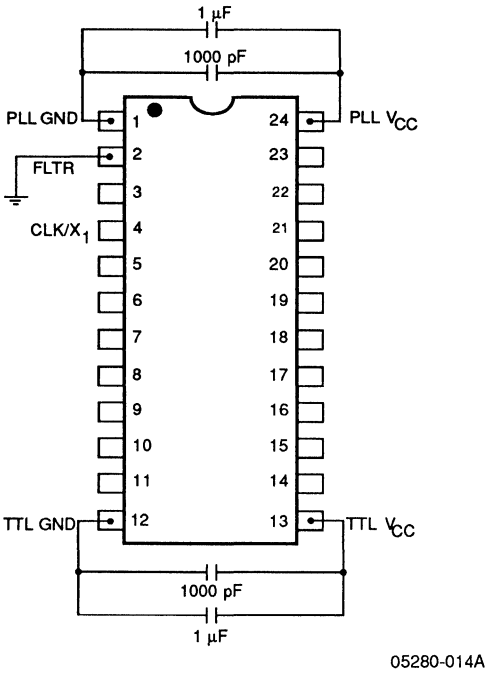


Figure 8. PEG Decoupling With PLL Bypassed

Bypass Mode Decoupling

When using the PEG with a direct TTL source above 10 MHz with the PLL bypassed, the decoupling shown in Figure 8 is mandatory.

The decoupling of the FLTR (filter) pin is necessary at frequencies higher than 10 MHz because the PLL will attempt to lock onto the incoming clock signal at CLK/X_1. This can cause anomalies in device operation and increase device jitter. Grounding the FLTR pin will isolate the PLL from the incoming clock and permit proper operation of the PEG.

PROGRAMMING

JEDEC Fuse Map

Table 6 shows the JEDEC fuse map and describes the fuse-addressing mechanism. For programming purposes the fuses are addressed by using T_0–T_{10} as inputs.

Each of the 622 fuses is addressed individually and programmed (output HIGH) or left unprogrammed (output LOW) by the equivalent level on T_{11}.

The fuses are addressed by a row/column matrix:

Input signals on pins T_0–T_5 define the row (T_0 = LSB, T_5 = MSB).

Input signals on pins T_6–T_{10} define the column (T_6 = LSB, T_{10} = MSB).

(Table 6 uses decimal notation for rows and columns. Note that there is no direct relationship between the row and column addresses and the fuse number. The fuse number is only used to refer to a particular fuse.)

In the Next Address/Event Block, each row describes the fuses used in one event. The bits accessed by columns 0 to 4 represent the Next Address column (address 0 accesses the LSB of the next address), the bits accessed by columns 5 through 16 represent the 12 output levels (column address 5 accesses the bit that uses T_0 as an output), and column address 17 accesses the STOP bit.

In the Start Address Block, the eight starting addresses are programmed by row addresses 32–39 (row address 32 programs the 5-bit word that defines the starting address selected by a 0 on the A_0– A_2 inputs, row address 39 programs the word that defines the starting address selected by a 7 on the A_0–A_2 inputs).

The Control Fuses (Reference Tables 2, and 6, and Figure 9) are accessed by row address 40 and column addresses 23–29. The functions of these fuses are described below:

Fuse #616: 0 = Divide-by-2 divider is not selected
1 = Divide-by-2 divider is selected (#617 must be zero)

Fuse #617: 0 = Divide-by-4 divider is not selected
1 = Divide-by-4 divider is selected (#616 must be zero)

Fuse #618: 0 = f_A is generated from PLL output
1 = $f_A = f_I$

Fuse #619: 0 = PLL multiplies by 10
1 = PLL multiplies by 5

Fuse #620: 0 = $f_O = f_{PLL}$ divided by 10
1 = $f_O = f_{PLL}$ divided by 5

Fuse #621: 0 = Start on rising edge of TRIG
1 = Stop on falling edge of TRIG

Fuse #622: 0 = Stop on trailing edge of TRIG (polarity defined by #621)
1 = No stop on trailing edge of TRIG

Table 2. Clock Logic Control Fuses

Fuse Pattern					Internal Clock		CLKOUT	
#616	#617	#618	#619	#620	f_A	(fc)	f_O	
0	0	0	0	0	10	X f_I	1	X f_I
1	0	0	0	0	5	X f_I	1	X f_I
0	1	0	0	0	2.5	X f_I	1	X f_I
0	0	0	0	1	10	X f_I	2	X f_I
1	0	0	0	1	5	X f_I	2	X f_I
0	1	0	0	1	2.5	X f_I	2	X f_I
0	0	0	1	0	5	X f_I	0.5	X f_I
1	0	0	1	0	2.5	X f_I	0.5	X f_I
0	1	0	1	0	1.25	X f_I	0.5	X f_I
0	0	0	1	1	5	X f_I	1	X f_I
1	0	0	1	1	2.5	X f_I	1	X f_I
0	1	0	1	1	1.25	X f_I	1	X f_I
X	X	1	0	0	1	X f_I	1	X f_I
X	X	1	0	1	1	X f_I	2	X f_I
X	X	1	1	0	1	X f_I	0.5	X f_I
X	X	1	1	1	1	X f_I	1	X f_I
1	1	X	X	X	Illegal code			

Note: f_I is the frequency on the X_1 input.

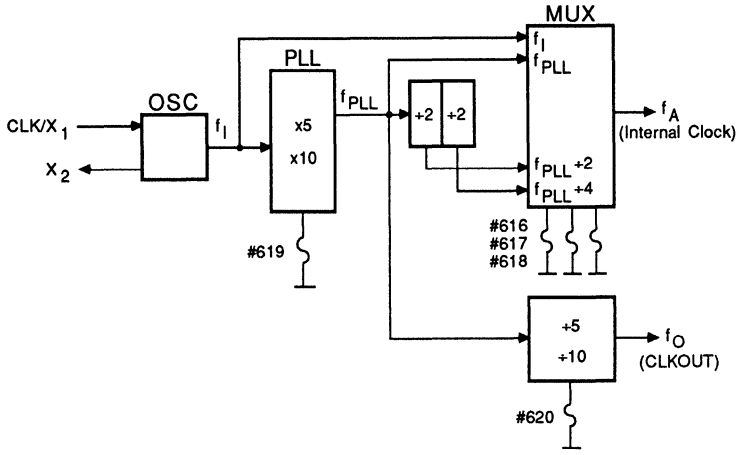


Figure 9. Clock Logic Control Fuse Locations

05280-012A

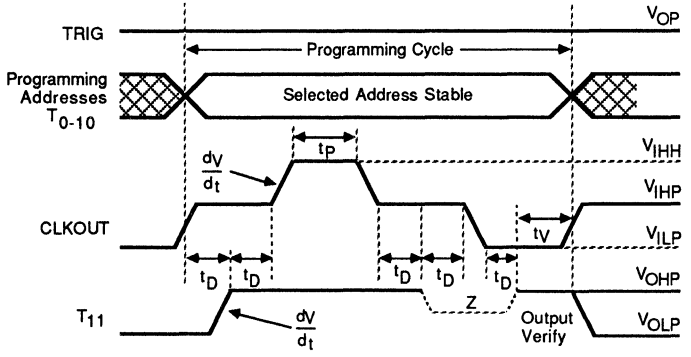


Figure 10. Programming and Verify Mode Timing

05280-013A

Programming Procedure

The following section describes the hardware requirements for programming the PEG.

This information is supplied for the designer of programming equipment. Normally the user of a PEG will utilize commercially available programming equipment and therefore has no need to study these pages (programming equipment information can be found in Table 3).

The PEG, like all AMD bipolar PROMs and PALs®, is programmed by selectively blowing platinum-silicide fuse links, one link at a time.

The table of programming parameters specifies not only the required voltage levels, but also the delays between the various actions (rise times for T_{11} and CLKOUT), and the length of the programming pulse t_P.

During the program and verify operation, TRIG must be pulled to V_{OP} (15 V) which disables the T_0–T_{11} and CLKOUT outputs. The fuse to be programmed (or verified) is selected by applying Row and Column addresses (see Figure 10) to T_0–T_{10}. (T_0 is the LSB of the Row address, T_6 is the LSB of the Column address). The selected fuse is programmed (blown, changed from 0 to 1) by a logic HIGH on T_{11} and a V_{IHH} pulse (12 V, 40–100 µs) on CLKOUT. It is common practice to verify each fuse-programming operation by three-stating the signal driving T_{11} and then applying a LOW level to CLKOUT. A blown link is indicated by a HIGH output on T_{11}.

Most links will open within the specified programming time. Occasionally a link might be stronger and require an additional programming pulse of longer duration (4 to 10 ms).

After the link has been verified, programming proceeds to the next link. After all links have been programmed, the entire array should be verified. An unprogrammed fuse (0) is indicated by a LOW on T_{11}; a programmed fuse (1) by a HIGH on T_{11}.

All unprogrammed (unblown) fuses are 0.

An unprogrammed fuse in the Next Event/Address Block generates a LOW on the outputs and a 0 as the next address. (The user can take advantage of this fact to "repair" a program error. For example: If address 31 had been left unprogrammed, any erroneously programmed location (0–3) can be moved to address 31 by programming the additional "1's" in the Next-Address field of the preceding word).

Table 3 is a list of recommended suppliers for Am2971A programming support. Each supplier is required to complete qualification by AMD to ensure high programming yields. An asterisk indicates a certified supplier (qualification complete). Consult your AMD Sales Representative to determine the current status of vendors noted as TBD or for other available models.

Table 3. Programming Equipment Information

Supplier and Location	Programmer Model(s)	Personality Modules	Socket Adaptors	Development Software
*Digelec, Inc. Ocean, NJ 201-493-2420	u803B	FAM 12	DA 42	—
	860	—	—	—
Stag Microsystems Santa Clara, CA 800-227-8836 800-222-7824	ZM 2200	—	—	—
Varix, Inc. Dallas, TX 214-437-0777	SP 0300	—	—	—
	GP 1140			
*AMD, Inc. Sunnyvale, CA 800-538-8450	Data I/O Model 29/B	UniPak™ 2/2B	AmPEGASUS™	AmPEGPDS
Inlab, Inc. Broomfield, CO 800-237-6759	Inlab 28	—	—	A.C.E.
Kontron, Inc. Mountain View, CA 415-965-7020	EPP80	UPM/B	—	—
Minato, KK Tokyo, Japan	TBD	TBD	TBD	—
Data I/O, Inc. Redmond, WA 206-881-6444	UniSite™ 40	—	—	—

*Certified supplier (qualification complete).

UniPak and UniSite are trademarks of Data I/O Corporation.

Table 4. Programming Procedure

Step	Item	Description
1	Determine location of fuses which are to be blown	
2	Set TRIG = V_{OP}	This disables T_0–T_{11} as outputs and prepares the chip for programming.
3	Set CLKOUT = V_{IHP}	This will three-state T_{11} and cause it to float up to V_{IHP}.
4	Set T_{11} = V_{IHP} after t_D	This prepares T_{11} to accept the programming current which will be gated through CLKOUT.
5	Set CLKOUT = V_{IHH} after t_D for programming time, t_{PF}	This gates the programming current to T_{11}.
6	Set CLKOUT = V_{IHP}	This removes the programming current from CLKOUT.
7	Remove applied voltage from T_{11}	This sets up T_{11} as output for Program Verification.

Table 5. Programming Verification Procedure

Step	Item	Description
1	Set TRIG = V_{OP} after t_D	TRIG remains at V_{OP} during the entire programming/verify cycle.
2	Set CLKOUT = V_{ILP} after t_D	This enables T_{11} as an output.
3a	Verify T_{11} = V_{OHP} (Note 1)	This condition occurs if programming has been successful.
3b	Verify T_{11} = V_{OLP} (Note 2)	This condition occurs if programming has been unsuccessful.

Notes: 1. If verify indicates programming has been successful, proceed to the next fuse and program using the programming steps of the previous table.

2. If verify indicates programming has been unsuccessful, return to the same fuse and re-attempt programming using the programming time t_{PS}.

Table 6. JEDEC Fuse Numbers*

FUSE																												BLOCK
COLUMN																												
	LSB	NEXT ADDRESS			MSB	LSB	CURRENT EVENT T_0–T_{11}											MSB	STOP BIT	LSB				MSB					
ROW	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	—	—	—	—	—	—	—	—	—	—	—	—	NEXT ADDRESS/ EVENT GENERATOR FUSES
1	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	—	—	—	—	—	—	—	—	—	—	—	—	
2	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	—	—	—	—	—	—	—	—	—	—	—	—	
3	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	—	—	—	—	—	—	—	—	—	—	—	—	
4	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	—	—	—	—	—	—	—	—	—	—	—	—	
5	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	—	—	—	—	—	—	—	—	—	—	—	—	
6	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	—	—	—	—	—	—	—	—	—	—	—	—	
7	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	—	—	—	—	—	—	—	—	—	—	—	—	
8	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	—	—	—	—	—	—	—	—	—	—	—	—	
9	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	—	—	—	—	—	—	—	—	—	—	—	—	
10	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	—	—	—	—	—	—	—	—	—	—	—	—	
11	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	—	—	—	—	—	—	—	—	—	—	—	—	
12	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	—	—	—	—	—	—	—	—	—	—	—	—	
13	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	—	—	—	—	—	—	—	—	—	—	—	—	
14	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	—	—	—	—	—	—	—	—	—	—	—	—	
15	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	—	—	—	—	—	—	—	—	—	—	—	—	
16	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	—	—	—	—	—	—	—	—	—	—	—	—	
17	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	—	—	—	—	—	—	—	—	—	—	—	—	
18	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	—	—	—	—	—	—	—	—	—	—	—	—	
19	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	—	—	—	—	—	—	—	—	—	—	—	—	
20	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	—	—	—	—	—	—	—	—	—	—	—	—	
21	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	—	—	—	—	—	—	—	—	—	—	—	—	
22	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	—	—	—	—	—	—	—	—	—	—	—	—	
23	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	—	—	—	—	—	—	—	—	—	—	—	—	
24	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	—	—	—	—	—	—	—	—	—	—	—	—	
25	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	—	—	—	—	—	—	—	—	—	—	—	—	
26	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	—	—	—	—	—	—	—	—	—	—	—	—	
27	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	—	—	—	—	—	—	—	—	—	—	—	—	
28	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	—	—	—	—	—	—	—	—	—	—	—	—	
29	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	—	—	—	—	—	—	—	—	—	—	—	—	
30	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	—	—	—	—	—	—	—	—	—	—	—	—	
31	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	—	—	—	—	—	—	—	—	—	—	—	—	
32	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	576	577	578	579	580	—	—	—	—	—	—	—	START ADDRESS FUSES
33	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	581	582	583	584	585	—	—	—	—	—	—	—	
34	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	586	587	588	589	590	—	—	—	—	—	—	—	
35	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	591	592	593	594	595	—	—	—	—	—	—	—	
36	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	596	597	598	599	600	—	—	—	—	—	—	—	
37	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	601	602	603	604	605	—	—	—	—	—	—	—	
38	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	606	607	608	609	610	—	—	—	—	—	—	—	
39	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	611	612	613	614	615	—	—	—	—	—	—	—	
40	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	616	617	618	619	620	621	622	CLOCK CONTROL & TRIGGER POLARITY FUSES**

* Row and Column numerals in this table are decimal number representations of the Binary Row and Column address signals applied to T_5–T_0 and T_{10}–T_6 respectively to program or verify each fuse individually. In this table, the reader should not confuse the use of T_{10}–T_0 as address inputs to program or verify fuses and the use of T_{11}–T_0 as Timing Outputs providing 12 programmable, registered output waveforms during normal operation (the output, T_{11}, provides the fuse condition during program or verify).

** Includes: $f_{internal}$ CLK, CLKOUT, TRIG Polarity and Stop Bit.

PROGRAMMING PARAMETERS (T_A, T_C = +25°C)

Parameter Symbol	Parameter Description		Min.	Typ.	Max.	Unit
V_{IHH}	Control Pin Extra HIGH Level		11	12	13	V
V_{OP}	Program Voltage at 15–200 mA		14	15	16	V
V_{IHP}	Input HIGH Level During Programming and Verify		2.4	5	5.5	V
V_{ILP}	Input LOW Level During Programming and Verify		0	0.3	0.5	V
V_{CCP}	V_{CC} During Programming @ I_{CC} = 250 mA	COM'L	5	5.2	5.5	V
		MIL	5	5.2	5.5	
dV_{T11}/dt	Rate of Output Enable Voltage Change (T_{11} Rising Edge)		20		250	V/µs
dV_{CLKOUT}/dt	Rate of Fuse Enable Voltage Change (CLKOUT Rising Edge)		100		1000	V/µs
t_P	Programming Time First Attempt, t_{PF}		40	50	100	µs
	Programming Time Subsequent Attempts, t_{PS}		4	5	10	ms
t_D	Delays Between Various Level Changes		100	200	1000	ns
t_V	Period During which Timing Output, T_n is Valid for Program Verification				500	ns
V_{OHP}	Output HIGH Level During Programming and Verify		2.5	5	5.5	V
V_{OLP}	Output LOW Level During Programming and Verify		0	0.3	0.4	V

Note: 1. Parameters are not tested, but are guaranteed by design.

Table 7. PEG Summary

Parameter Description	Am2971A
Max. Operating Frequency	100 MHz
Max. 12-Output Skews:	
LOW-to-HIGH	2.5 ns
HIGH-to-LOW	2.5 ns
Opposite	6.5 ns
Max. Matched Output Skews: *	Down to:
LOW-to-HIGH	1.0 ns
HIGH-to-LOW	1.0 ns
Opposite	6.5 ns
I_{CC}, Maximum	425 mA
On-Chip Crystal Oscillator	Yes
Minimum Output Resolution:	
Between Outputs	10 ns
Same Output	10 ns

* See AC Switching Characteristics

PEG SUPPORT

AmPEGASUS

AMD's PEG Adaptor Socket and Universal Software (Am-PEGASUS) is a passive programmer adaptor socket made for use in conjunction with Data I/O Corporation's Model 29/29A/29B Universal Programmer equipped with a UniPak 2/2A/2B adaptor. This passive unit makes use of the generic 2K x 8 PROM socket by reassigning the PROM configuration into a PEG pin configuration and programming algorithm. The socket draws its power from the Model 29 and contains protective circuitry to provide additional safeguards for the UniPak. Additional information can be obtained from the AmPEGASUS User's Manual, PID# 09241A.

AmPEGPDS

AMD's PEG Programming Development Software (AmPEGPDS) is a software tool designed to aid the user in creating fuse map in the JEDEC standard for programming a PEG device. The main purpose of the software is to create and translate the input specification into a format that can be accepted by the programmer. The input specification is created by the designer using AmPEGPDS as an editor. AmPEGPDS is available on a standard 5-1/4" floppy disk, included in both of the PEG Application Kits described as follows.

PEG Application Kits

The **PEG Starter Kit** includes:

- Am2971A PEG Data Sheet
- AmPEGPDS
- AmPEGPDS Software User's Manual
- Two PEG Unprogrammed Samples
- Applications Articles
- Am2971A Product Description

The PEG Starter Kit is available free of charge from any AMD Sales Representative.

The **PEG Design Kit** includes the **PEG Starter Kit** plus:

- AmPEGASUS Programming Adaptor Socket (customer to specify DIP or LCC)
- AmPEGASUS Translation Software
- AmPEGASUS User's Manual

Consult your AMD Sales Representative for pricing information on the PEG Design Kit.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	–65 to +150°C
Ambient Temperature with Power Applied	–55 to +125°C
Supply Voltage to Ground Potential Continous (TLL and PLL V_{CC})	0 to +7.0 V
DC Voltage Applied to Outputs for HIGH Output State	0 to + V_{CC} Max.
DC Input Voltage	–0.5 to +5.5 V
DC Input Current	–18 to +5.0 mA

Stresses above those listed under Absolute Maximum Ratings may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +70°C
TLL and PLL Supply Voltage (V_{CC})	5.0 V ± 10%
Min.	4.5 V
Max.	5.5 V

Military* (M) Devices

Case Temperature (T_C)	–55 to +125°C
TLL and PLL Supply Voltage (V_{CC})	5.0 V ± 10%
Min.	4.5 V
Max.	5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

** Military Product 100% tested at T_C = +25°C, +125°C, and –55°C.*

DC CHARACTERISTICS over COMMERCIAL operating range unless otherwise specified
(for APL Products, Group A, Subgroups 1, 2, 3, 7, and 8 are tested unless otherwise noted)

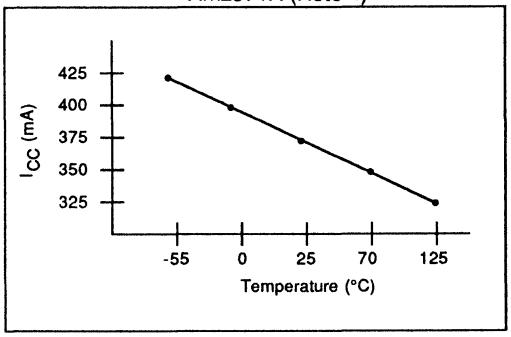
Parameter Symbol	Parameter Description	Test Conditions (Note 1)*	Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = Min. I_{OH} = –1 mA V_{IN} = V_{IH} or V_{IL}	2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = Min. I_{OL} = 8 mA V_{IN} = V_{IH} or V_{IL}		0.4	V
V_{IH}	Input HIGH Voltage	Guaranteed Input HIGH Voltage for all Inputs (Note 2)	2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input LOW Voltage for all Inputs (Note 2)		0.8	V
V_{IHC}	Input HIGH Voltage to CLK/X_1	Guaranteed Input HIGH Voltage for all Inputs	3.0		V
V_{ILC}	Input LOW Voltage to CLK/X_1	Guaranteed Input LOW Voltage for all Inputs		0.8	V
V_I	Input Clamp Voltage	V_{CC} = Min. I_{IN} = –18 mA (Note 2)		–1.2	V
I_{IH}	Input HIGH Current	V_{CC} = Min. CLK/X_1 V_{IN} = 3.0 V		700	µA
		V_{CC} = Max. A_0–A_2 and TRIG V_{IN} = 2.7 V		20	
I_{IL}	Input LOW Current	V_{CC} = Max. CLK/X_1 V_{IN} = 0.5 V		–500	µA
		V_{CC} = Max. A_0–A_2 and TRIG V_{IN} = 0.5 V		–250	
I_I	Input Current	V_{CC} = Min. CLK/X_1 V_{IN} = 4.0 V		1.2	mA
		V_{CC} = Max. A_0–A_2 V_{IN} = 5.5 V		100	µA
		V_{CC} = Max. TRIG V_{IN} = V_{CC} – 0.5 V		100	
I_{SC}	Output Short-Circuit Current	V_{CC} = Max. T_0–T_{11}, CLKOUT V_{OUT} = 0 V	–15	–100	mA
I_{CC}	Power Supply Current	V_{CC} = Max.		425	mA
		V_{CC} = 5.0 V T_C = 25°C (Note 4)		370	

*Key: C = COM'L Devices
M = MIL Devices

Notes:
1. For conditions shown as Min. or Max., use appropriate value specified under Operating Range for the applicable device type.
2. Does not apply to CLK/X_1 and X_2.
3. No more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.
4. I_{CC} varies with temperature and oscillation frequency. Worst-case I_{CC} is at minimum temperature. Typical I_{CC} (V_{CC} = 5.0 V, T_A, T_C = +25°C) represents nominal units and is not tested. See the following graph.

Typical Power Supply Current (Nominal Unit)

Am2971A (Note 4)



05280-015A

SWITCHING CHARACTERISTICS over operating ranges unless otherwise specified (for APL Products, Group A, Subgroups 9, 10, and 11 are tested unless otherwise noted)

Clock Control Logic: PLL Bypassed

No.	Parameter Symbol	Parameter Description	Test Conditions*		Am2971A Min.	Am2971A Max.	Unit
1	f_I	Direct TTL Clock Source, Input Frequency at CLK/X_1 with PLL Multiplier set to "x1" (Fuse #618 = 0)	(Notes 1, 4)	C	0	100	MHz
				M	0	100	
		Crystal Clock Source (Note 6). Input Frequency at CLK/X_1 and X_2 with PLL Multiplier set to "x1" (Fuse #618 = 1)		C	0	100	
				M	0	100	
2	t_{RCO}	Rise Time of Clock Out (CLKOUT) Signal	(Note 1)	C		10	ns
				M		10	
3	t_{FCO}	Fall Time of Clock Out (CLKOUT) Signal	(Note 1)	C		10	ns
				M		10	
4	t_{RTO}	Rise Time of T_n Outputs	(Note 1)	C		8	ns
				M		8	
5	t_{FTO}	Fall Time of T_n Outputs	(Note 1)	C		8	ns
				M		8	
6	t_{SKEWLH}	Skew Time between T_n Outputs. All Outputs Switching ↑	C_L = 50 pF (Note 5)	C		2.5	ns
				M		2.5	
6a	t_{SKEWHL}	Skew Time between T_n Outputs. All Outputs Switching ↓	C_L = 50 pF (Note 5)	C		2.5	ns
				M		2.5	
7	t_{SKEW8}	Skew Time between Matched T_n Outputs. Eight Outputs Switching ↑ or ↓ ($T_{0-3, 5, 6, 10, 11}$)	C_L = 50 pF (Note 5)	C		2	ns
				M		2	
8	t_{SKEW6}	Skew Time between Matched T_n Outputs. Six Outputs Switching ↑ or ↓ ($T_{1-3, 5, 10, 11}$)	C_L = 50 pF (Note 5)	C		1.5	ns
				M		1.5	
9	t_{SKEW4}	Skew Time between Matched T_n Outputs. Four Outputs Switching ↑ or ↓ ($T_{2, 3, 10, 11}$)	C_L = 50 pF (Note 5)	C		1	ns
				M		1.5	
10	t_{SKEW}	Skew Time between T_n Outputs. All Outputs Switching Mixed Transition	C_L = 50 pF (Note 5)	C		6.5	ns
				M		6.5	
11	$t_{SET\ TCA}$	TRIG Active to CLK/X_1 ↑ Setup Time to Start an Output (T_n) Sequence		C	8		ns
				M	8		
12	$t_{SET\ TCI}$	TRIG Inactive to CLK/X_1 ↑ Setup Time to Terminate an Output (T_n) Sequence		C	4		ns
				M	4		

SWITCHING CHARACTERISTICS — Clock Control Logic: PLL Bypassed (Cont'd.)

No.	Parameter Symbol	Parameter Description	Test Conditions*		Am2971A Min.	Am2971A Max.	Unit
13	$t_{PD\ CTO}$	Propagation Delay from an Input Clock Edge (CLK/X_1) ↑ to an Active T_n Output		C	10	22	ns
				M	10	22	
14	$t_{SET\ AT}$	A_{0-2} Inputs to TRIG ↑ Setup Time	(Note 2)	C	1		ns
				M	2		
15	$t_{HOLD\ AT}$	A_{0-2} Inputs to TRIG ↑ Hold Time	(Note 2)	C	8		ns
				M	8		
16	t_{RCVRY} (Min.)	Chip Recovery/Reset Time between New (TRIG Active) Timing Sequences when Halting via STOP Bits	(Note 1)	C		$2/f_I$	ns
				M		$2/f_I$	
		Chip Recovery/Reset Time between New (TRIG Active) Timing Sequences when Halting via TRIG Inactive		C		$2/f_I$	
				M		$2/f_I$	
17	$t_{PD\ TTA}$	Propagation Delay from TRIG Active to Start of First T_n Output Signals	(Note 3)	C	$18+1/f_I$	$30+1/f_I$	ns
				M	$18+1/f_I$	$30+1/f_I$	
18	$t_{PD\ TTI}$	Propagation Delay from TRIG Inactive to Completion of Last T_n Output Signals	(Note 3)	C	$14+1/f_I$	$26+1/f_I$	ns
				M	$14+1/f_I$	$26+1/f_I$	
19	$t_{PWH\ T}$	TRIG Input Pulse Width in HIGH State	(Note 3)	C	13		ns
				M	13		
20	$t_{PWL\ T}$	TRIG Input Pulse Width in LOW State	(Note 3)	C	13		ns
				M	13		
21	$t_{PRD\ TO}$	T_n Output Period/Timing Resolution between T_n Outputs	(Note 3)	C	$1/f_I$		ns
				M	$1/f_I$		

*Key: C = COM'L Devices
M = MIL Devices

SWITCHING CHARACTERISTICS — Clock Control Logic: PLL Selected (Cont'd.)

No.	Parameter Symbol	Parameter Description	Test Conditions*		Am2971A Min.	Am2971A Max.	Unit
1	f_I	Direct TTL Clock Source. Input Frequency at CLK/X_1 with PLL Multiplier set to "x10" (Fuse #619 = "0")	(Notes 1, 4)	C	1	10	MHz
				M	1	10	
		Direct TTL Clock Source. Input Frequency at CLK/X_1 with PLL Multiplier set to "x5" (Fuse #619 = "1")		C	2	20	
				M	2	20	
		Crystal Clock Source (Note 6). Input Frequency at CLK/X_1 and X_2 with PLL Multiplier set to "x10" (Fuse #619 = "0")		C	1	10	
				M	1	10	
		Crystal Clock Source (Note 6). Input Frequency at CLK/X_1 and X_2 with PLL Multiplier set to "x5" (Fuse #619 = "1")		C	2	20	
				M	2	20	
2	t_{RCO}	Rise Time of Clock Out (CLKOUT) Signal	(Note 1)	C		10	ns
				M		10	
3	t_{FCO}	Fall Time of Clock Out (CLKOUT) Signal	(Note 1)	C		10	ns
				M		10	
4	t_{RTO}	Rise Time of T_n Outputs	(Note 1)	C		8	ns
				M		8	
5	t_{FTO}	Fall Time of T_n Outputs	(Note 1)	C		8	ns
				M		8	
6	t_{SKEWLH}	Skew Time between T_n Outputs. All Outputs Switching ↑	C_L = 50 pF (Note 5)	C		2.5	ns
				M		2.5	
6a	t_{SKEWHL}	Skew Time between T_n Outputs. All Outputs Switching ↓	C_L = 50 pF (Note 5)	C		2.5	ns
				M		2.5	
7	t_{SKEW8}	Skew Time between Matched T_n Outputs. Eight Outputs Switching ↑ or ↓ ($T_{0-3, 5, 6, 10, 11}$)	C_L = 50 pF (Note 5)	C		2.0	ns
				M		2.0	
8	t_{SKEW6}	Skew Time between Matched T_n Outputs. Six Outputs Switching ↑ or ↓ ($T_{1-3, 5, 10, 11}$)	C_L = 50 pF (Note 5)	C		1.5	ns
				M		1.5	
9	t_{SKEW4}	Skew Time between Matched T_n Outputs. Four Outputs Switching ↑ or ↓ ($T_{2, 3, 10, 11}$)	C_L = 50 pF (Note 5)	C		1.0	ns
				M		1.5	
10	t_{SKEW}	Skew Time between Tn Outputs. All Outputs Switching Mixed Transition	C_L = 50 pF (Note 5)	C		6.5	ns
				M		6.5	

SWITCHING CHARACTERISTICS — Clock Control Logic: PLL Selected (Cont'd.)

No.	Parameter Symbol	Parameter Description	Test Conditions*		Am2971A Min.	Am2971A Max.	Unit
11	$t_{SET\ TCA}$	TRIG Active to CLK/X_1 (↑) Setup Time to Start an Output (T_n) Sequence		C	8		ns
				M	8		
12	$t_{SET\ TCI}$	TRIG Inactive to CLK/X_1 (↑) Setup Time to Terminate an Output (T_n) Sequence		C	4		ns
				M	4		
13a	$t_{PD\ CTO}$	Propagation Delay from an Input Clock Edge (CLK/X_1) (↑) to an Active T_n Output when "/1" has been Selected (Fuses #616 and 617 = "0")		C	10	20	ns
				M	10	26	
13b	$t_{PD\ CTO}$	Propagation Delay from an Input Clock Edge (CLK/X_1) (↑) to an Active T_n Output when "/2" has been Selected (Fuses #616 = "1" and 617 = "0")	(Note 3)	C	10+ 1/2f_C	20+ 1/2f_C	ns
				M	10+ 1/2f_C	26+ 1/2f_C	
13c	$t_{PD\ CTO}$	Propagation Delay from an Input Clock Edge (CLK/X_1) (↑) to an Active T_n Output when "/4" has been Selected (Fuses #616 = "0" and Fuse 617 = "1")	(Note 3)	C	10+ 3/4f_C	20+ 3/4f_C	ns
				M	10+ 3/4f_C	26+ 3/4f_C	
14	$t_{SET\ AT}$	A_{0-2} Inputs to TRIG (↑) Setup Time	(Note 2)	C	1.0		ns
				M	2.0		
15	$t_{HOLD\ AT}$	A_{0-2} Inputs to TRIG (↑) Hold Time	(Note 2)	C	8		ns
				M	8		
16	t_{RCVRY} (Min.)	Chip Recovery/Reset Time between New (TRIG Active) Timing Sequence when Halting via STOP Bits	(Note 1)	C		2/f_C	ns
				M		2/f_C	
		Chip Recovery/Reset Time between New (TRIG Active) Timing Sequence when Halting via TRIG Inactive		C		2/f_C	
				M		2/f_C	
17a	$t_{PD\ TTA}$	Propagation Delay from TRIG Active to Start of First T_n Output Signal when "/1" has been Selected (Fuses #616 and 617 = "0")	(Note 3)	C	18+ 1/f_C	28+ 1/f_C	ns
				M	18+ 1/f_C	34+ 1/f_C	
17b	$t_{PD\ TTA}$	Propagation Delay from TRIG Active to Start of First T_n Output Signal when "/2" has been Selected (Fuse #616 = "1" and Fuse 617 = "0")	(Note 3)	C	18+ 1/2f_C	28+ 1/2f_C	ns
				M	18+ 1/2f_C	34+ 1/2f_C	

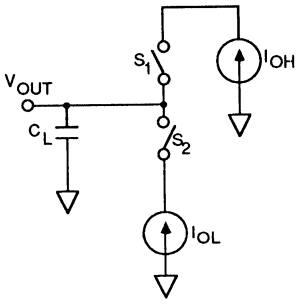
SWITCHING CHARACTERISTICS — Clock Control Logic: PLL Selected (Cont'd.)

No.	Parameter Symbol	Parameter Description	Test Conditions*		Am2971A Min.	Am2971A Max.	Unit
17c	$t_{PD\,TTA}$	Propagation Delay from TRIG Active to Start of First T_n Output Signal when "/4" has been Selected (Fuse #616 = "0" and Fuse 617 = "1")	(Note 3)	C	18+ 3/4f_C	28+ 3/4f_C	ns
				M	18+ 3/4f_C	34+ 3/4f_C	
18	$t_{PD\,TTI}$	Propagation Delay from TRIG Inactive to Completion of Last T_n Output Signals	(Note 3)	C	14+ 1/f_C	24+ 1/f_C	ns
				M	14+ 1/f_C	30+ 1/f_C	
19	$t_{PWH\,T}$	TRIG Input Pulse Width in HIGH State when "/1" has been Selected (Fuses #616 and 617 = "0")	(Note 3)	C	13		ns
				M	13		
20	t_{PWLT}	TRIG Input Pulse Width in LOW State	(Note 3)	C	13		ns
				M	13		
21	$t_{PRD\,TO}$	T_n Output Period/Timing Resolution between T_n Outputs	(Note 3)	C	1/f_C		ns
				M	1/f_C		
22	$t_{PD\,CTC}$	Propagation Delay from an Input Clock Edge (CLK/X_1) (↑) to an Output Clock Edge (Fuses #616 and 617 = "0")	(Note 1)	C	10	19	ns
				M	10	21	

*Key: C = COM'L Devices
M = MIL Devices

Notes:
1. Not production tested due to automatic test equipment limitation; guaranteed by characterization.
2. Only A_2 is tested.
3. Not tested; calculated from other parameters.
4. f_I input clock duty cycle should be 50% ± 10%.
5. Not tested. Skew times shown each case are guaranteed by characterization for specific outputs listed with any number of outputs switching.
6. The Am2971A has a maximum operating frequency of 100 MHz with a TTL source, all outputs in use, but only nine outputs (any nine) may be programmed to switch simultaneously. If a crystal is used to clock the Am2971A, all outputs may be used, buy only six outputs (any six) can be programmed to switch simultaneously.

SWITCHING TEST CIRCUIT



A. Outputs

05280-016A

Notes: 1. C_L = 50 pF. The load capacitance includes scope probe, wiring, and stray capacitance without the device in the test fixture.

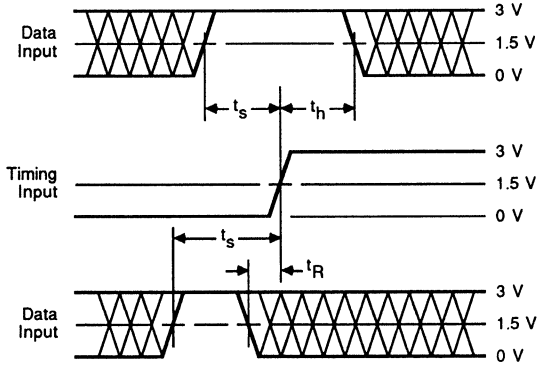
2. S_1 and S_2 are open during all DC and functional testing.

3. During AC testing, switches are set as follows:

   1) For V_{OUT} > 1.5 V, S_1 is closed and S_2 open

   2) For V_{OUT} < 1.5 V, S_1 is open and S_2 closed

SWITCHING TEST WAVEFORMS

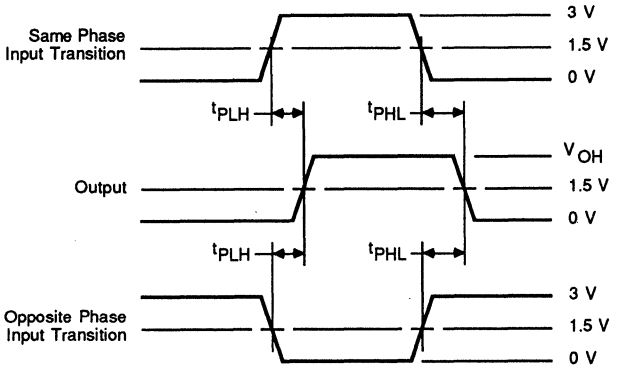


A. Setup, Hold, and Release Times

05280-017A

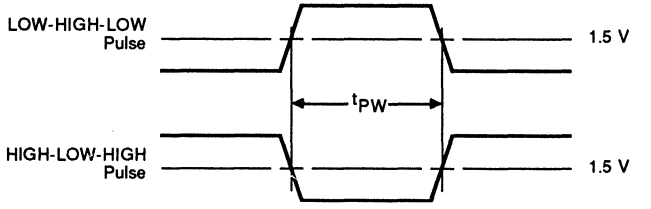
Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.

2. Cross-hatched area is don't care condition.



05280-018A

B. Propagation Delay



05280-019A

C. Pulse Width

General Test Notes

Automatic tester hardware and handler hardware add additional round-trip AC delay to test measurements. Actual propagation delay testing may incorporate a correlation factor to negate the additional delay.

Function testing is done with input LOW less than V_{IL}, and input HIGH greater than V_{IH}. A single trip point at the approximate threshold voltage is used to determine output logic level.

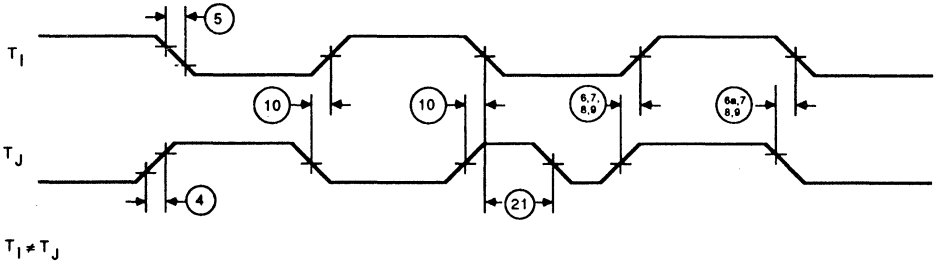
Some Setup and Hold tests are not performed due to tester accuracy limitation. They are guaranteed by correlation.

AC loads specified in this data sheet are used for bench testing. Programmable loads, which simulate data sheet loads, are used during automatic production testing.

KEY TO SWITCHING WAVEFORMS

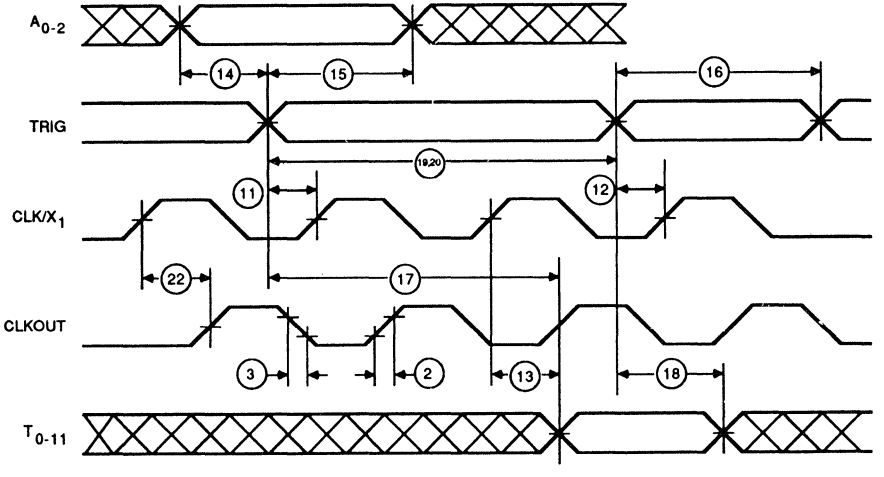
WAVEFORM	INPUTS	OUTPUTS
	Must Be Steady	Will Be Steady
	May Change from H to L	Will Be Changing from H to L
	May Change from L to H	Will Be Changing from L to H
	Don't Care Any Change Permitted	Changing State Unknown
	Does Not Apply	Center Line is High Impedence "Off" State

SWITCHING WAVEFORMS



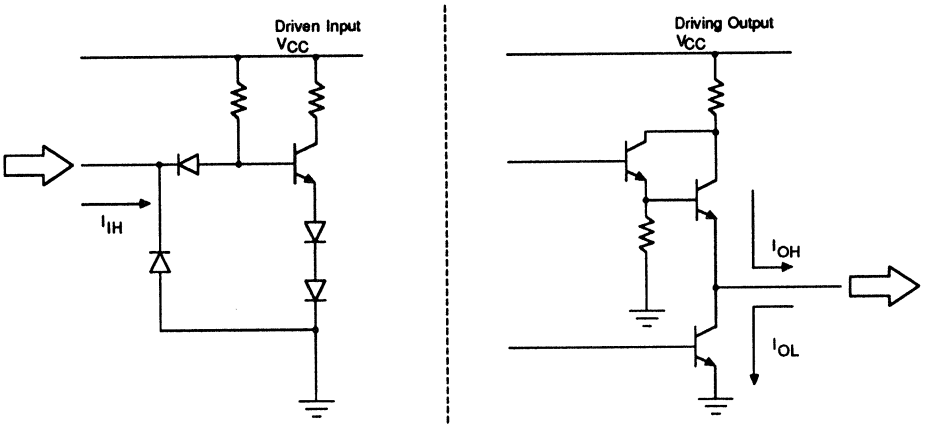
Rise Time/Fall Time /Skews

05280-023A

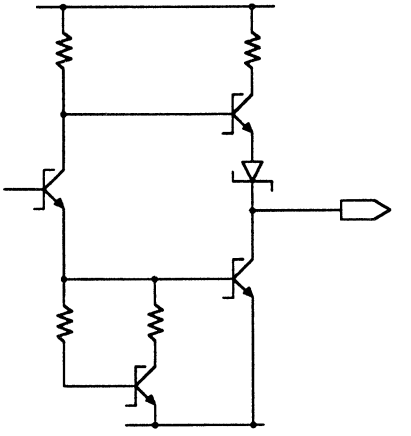


05280-022A

INPUT/OUTPUT CIRCUIT DIAGRAMS



05280-020A



Output Configuration for CLKOUT

05280-021A

Final

Am2976

Eleven-Bit Dynamic Memory Driver

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

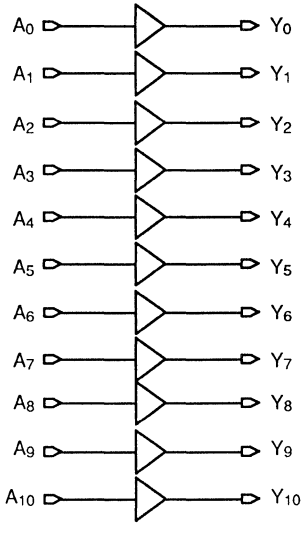
- Internal resistors eliminate need for external series resistors and reduce package count
- Guaranteed maximum undershoot of –0.5 V on HIGH-to-LOW transition
- Glitch-free outputs during power-up and power-down
- Simple "flow-through" pinout
- Large capacitive drive capability (up to 88 DRAMs) designed to drive 16K, 64K, 256K, 1M and 4M dynamic-memory arrays
- 24-pin, 0.3-inch space-saving plastic dual In-line package and 28-pin plastic leaded chip carrier

GENERAL DESCRIPTION

The Am2976 is an 11-bit bipolar Dynamic RAM Driver designed to drive the capacitive input characteristics of the address and control lines of 64K, 256K, 1M and 4M MOS dynamic RAMs. This device has non-inverting drivers and is functionally similar to the Am2966, but contains eleven drivers to accommodate wider address paths. A significant performance advantage of the increase in drivers per device is the ability to combine more signals onto a single device, thus minimizing skew time between drivers (i.e., $\overline{RAS}$ and $\overline{CAS}$).

The architecture of the lower output driver includes an internal collector-resistor to control undershoot (not to exceed –0.5 V) on the HIGH-to-LOW transition. This design also allows for control of the output fall time without slowing the output rise. The upper output driver pulls up to 2.7 V minimum to be compatible with MOS memory and is designed to have a rise time symmetrical with the low output's controlled fall times. The output impedance, 25 ohms nominal, is identical in both the logic HIGH and LOW states — this value was selected to match the intrinsic impedance of a PC board trace. The inclusion of internal resistors eliminates the need for additional external series resistors, therefore reducing package count and saving board area.

LOGIC DIAGRAM

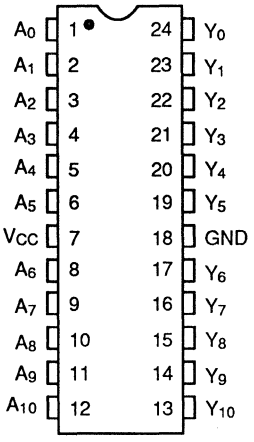


06024-001A

Publication# 06024 Rev. D Amendment /0
Issue Date: January 1990

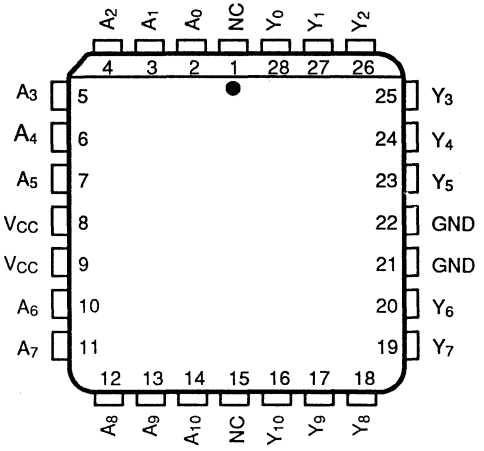
CONNECTION DIAGRAMS

DIP



06024-002A

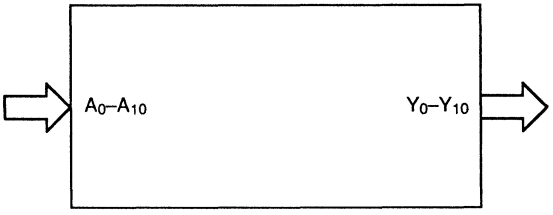
PLCC



06024-003A

Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



VCC = Power Supply
GND = Ground

06024-004A

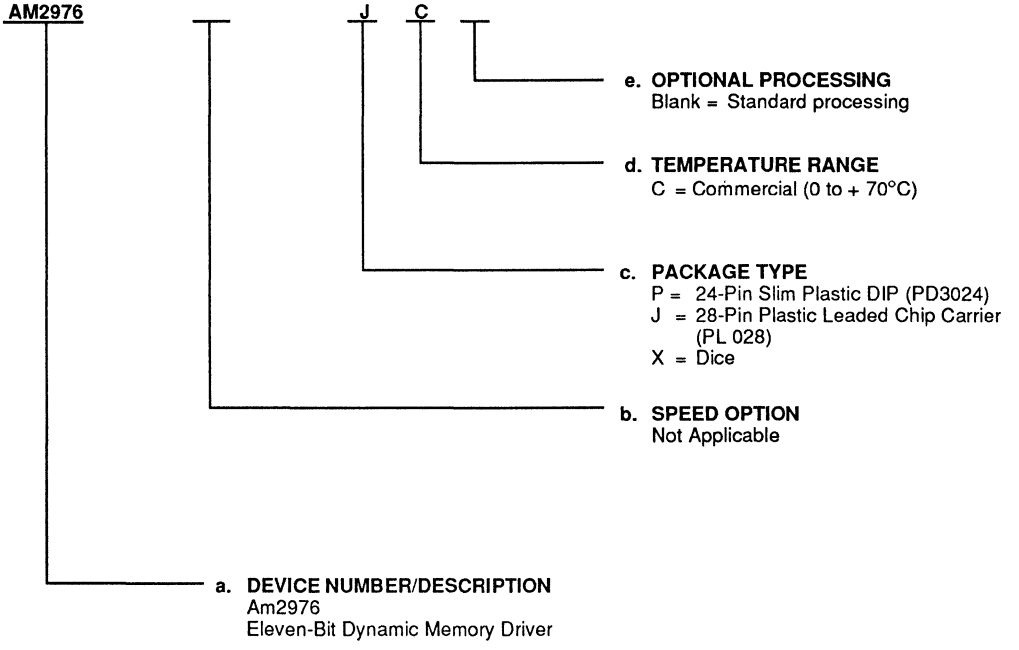
RELATED AMD PRODUCTS

Part No.	Description
Am29368	1M Dynamic Memory Controller/Driver
Am29C60A	High Speed CMOS Cascadable 16-Bit EDC
Am29C660D	12 ns CMOS Cascadable 32-Bit EDC
Am29C668	4M Configurable Dynamic Memory Controller/Driver
Am2968A	256K Dynamic Memory Controller/Driver
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port Multiple Bus Exchange with Parity
Am2965/6	8-Bit Dynamic RAM Driver Inverting/Non-Inverting

ORDERING INFORMATION
Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM2976	PC, JC, XC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, and to check on newly released valid combinations.

PIN DESCRIPTION

A_0–A_{10}
Driver (Address) Input (Inputs — 11)

These eleven pins are the inputs to the driver.

Y_0–Y_{10}
Driver (General Control) Output (Outputs — 11)

These eleven pins are the outputs from the driver.

V_{CC}, GND
Power, Ground Power Pair

TTL power and ground pins. The chip carrier package has two of each and the DIP has one of each.

FUNCTIONAL DESCRIPTION

Functional Description of the Am2976 is summarized on page one under General Description. For a Typical Output Driver diagram and a Function Table, refer to Figure 1 and Table 1, which follow.

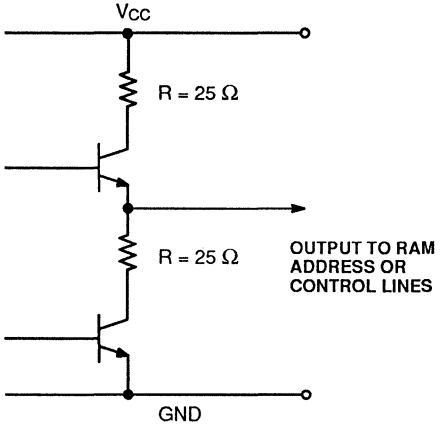


Figure 1. Typical Output Driver

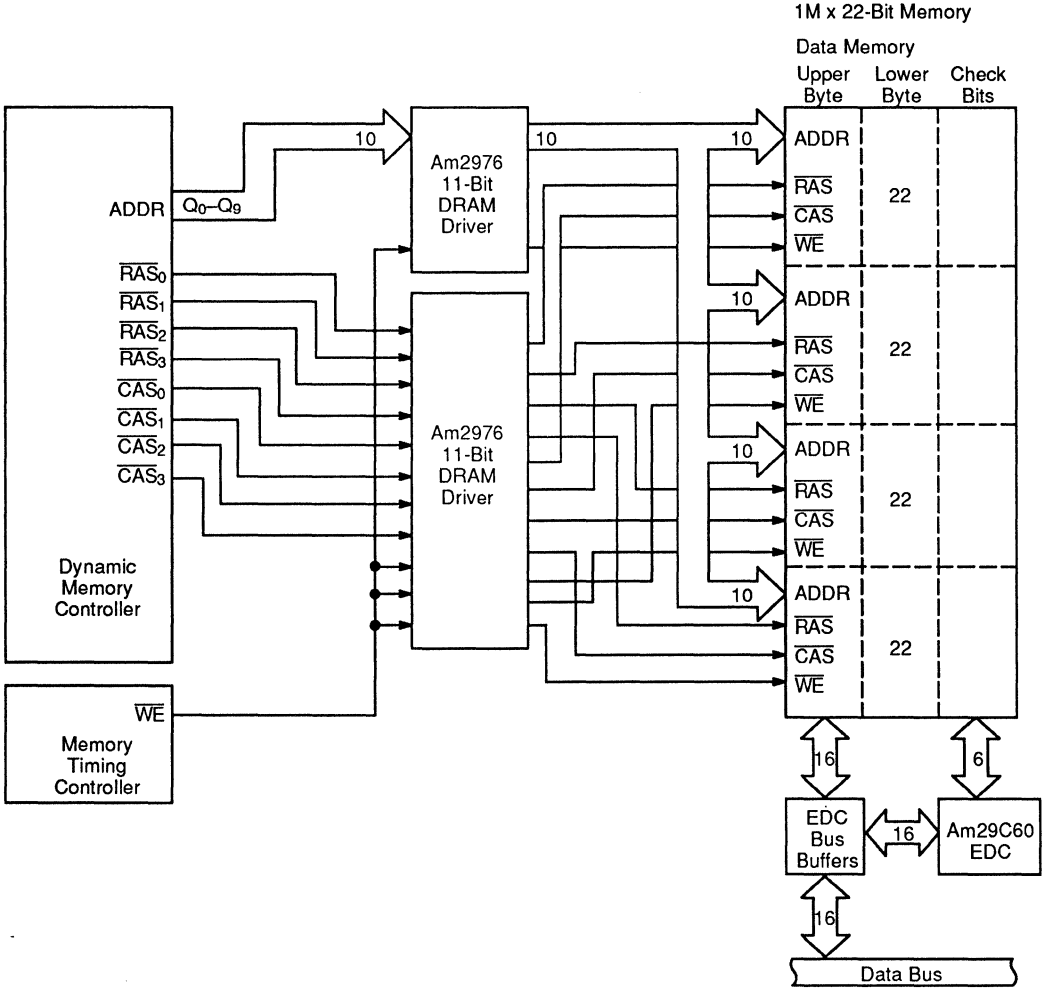
Table 1. Function Table

Inputs	Outputs
A_0–A_{10}	Y_0–Y_{10}
H	H
L	L

Key: H = HIGH
L = LOW

APPLICATIONS

The Am2976 is designed for use with a Dynamic Memory Controller — the Am29C668, for example — where large dynamic memories with highly capacitive input lines require additional buffering. A fully utilized memory system consists of four banks of dynamic RAMs (see Figure 2). Each bank uses identical address inputs but has individual $\overline{RAS}$, $\overline{CAS}$, and $\overline{WE}$ inputs. Only two devices are required to drive ten address lines, four $\overline{RAS}$ lines, four $\overline{CAS}$ lines, and four $\overline{WE}$ lines. Referencing Figure 2, note that the $\overline{RAS}$ and the $\overline{CAS}$ inputs to the memory array come from one device — thus minimizing the skew between $\overline{RAS}$ and $\overline{CAS}$ lines — optimizing the system's performance.



06024-006A

Figure 2. Am2976 Application Diagram

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	–65 to +150°C
Ambient/Case Temperature with Power Applied	–55 to +125°C
Supply Voltage with Respect to Ground	–0.5 to +7.0 V
DC Voltage Applied to Outputs for HIGH Output State	–1.5 to V_{CC} Max.
DC Input Voltage	–0.5 to +7.0 V
DC Output Short-Circuit Current	200 mA
DC Input Current	–30 to +5.0 mA

Stresses above those listed under Absolute Maximum Ratings may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +70°C
Supply Voltage	5.0 V ± 10%

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating ranges unless otherwise specified (Note 1)

Parameter Symbol	Parameter Description	Test Conditions		Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} =Min., V_{IN} = V_{IH}; I_{OH} = –1 mA		2.8		V
V_{OL}	Output LOW Voltage	V_{CC} = Min.,	I_{OL} = 1 mA		0.5	V
		V_{IN} = V_{IH}	I_{OL} = 12 mA		0.8	
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for all Inputs		2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for all Inputs			0.8	V
V_I	Input Clamp Voltage	V_{CC} = Min., I_{IN} = –18 mA			–1.5	V
I_{IL}	Input LOW Current	V_{CC} = Max., V_{IN} = 0.4 V			–0.4	mA
I_{IH}	Input HIGH Current	V_{CC} = Max., V_{IN} = 2.4 V			20	µA
I_I	Input HIGH Current	V_{CC} = Max., V_{IN} = 5.5 V			100	µA
I_{SC}	Output Short-Circuit Current	V_{CC} = Max. (Note 2)		–75	–275	mA
I_{CC}	Power Supply Current		T_A = 0 to +70°C		90	mA

Notes:

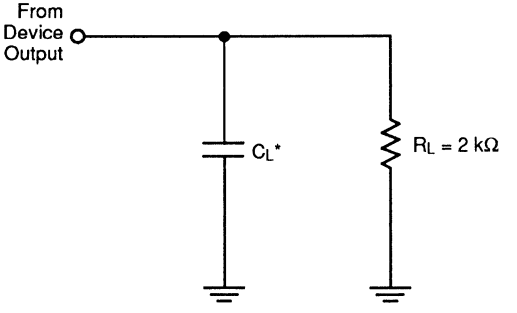
1. For conditions shown as Min. or Max., use appropriate value as specified under Operation Ranges for applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	Must Be Steady	Will Be Steady
	May Change from H to L	Will Be Changing from H to L
	May Change from L to H	Will Be Changing from L to H
	Don't Care Any Change Permitted	Changing State Unknown
	Does Not Apply	Center Line is High Impedance "Off" State

KS000010

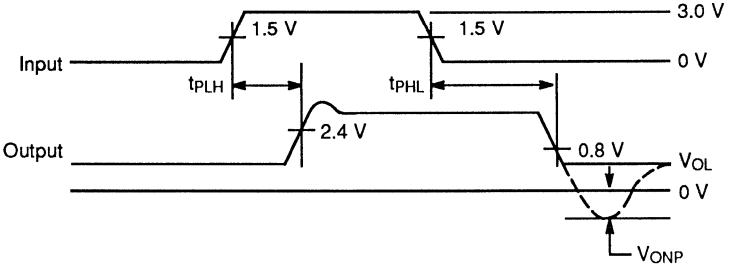
SWITCHING TEST CIRCUIT



* t_{PD} specified at C_L = 50 and 500 pF.

06024-007A

SWITCHING TEST WAVEFORM

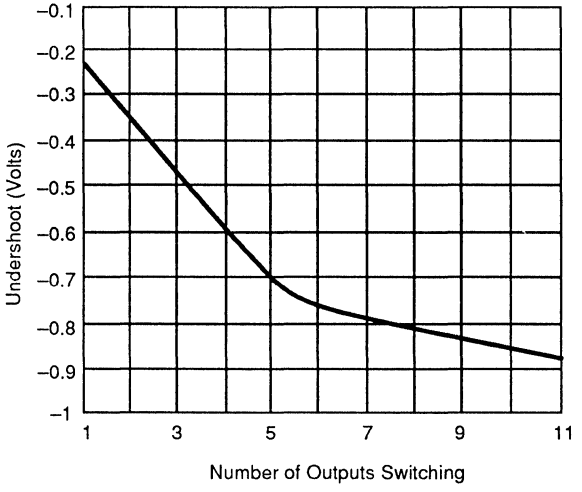


06024-008A

SWITCHING CHARACTERISTICS over operating range unless otherwise specified

No.	Parameter Symbol	Parameter Description		1 Output Switching		11 Outputs Switching		Unit
				Min.	Max.	Min.	Max.	
C_L = 50 pF								
1	t_{PLH}	Propagation Delay from A_i to Y_i, LOW-to-HIGH Transition	C Devices		16		22	ns
2	t_{PHL}	Propagation Delay from A_i to Y_i HIGH-LOW Transition	C Devices		12		14	ns
3	t_{SKEW}	Same Transition, Output to Output (Note 1)	C Devices		5		12	ns
4	t_{SKEW}	Opposite Transition, Output to Output (Note 1)	C Devices		9		16	ns
5	V_{ONP}	Output Voltage Undershoot (Note 2)	C Devices		–0.5		–1.0	V
C_L = 500 pF								
6	t_{PLH}	Propagation Delay from A_i to Y_i, LOW-HIGH Transition	C Devices		29		43	ns
7	t_{PHL}	Propagation Delay from A_i to Y_i, HIGH-LOW Transition	C Devices		33		36	ns
8	t_{SKEW}	Same Transition, Output to Output (Note 1)	C Devices		5		12	ns
9	t_{SKEW}	Opposite Transition, Output to Output (Note 1)	C Devices		9		16	ns
10	V_{ONP}	Output Voltage Undershoot (Note 2)	C Devices		–0.5		–0.5	V

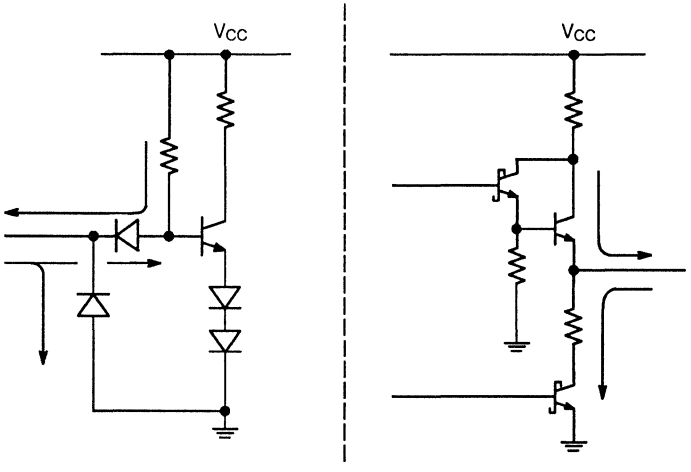
TYPICAL PERFORMANCE CURVE



06024-009A

Undershoot vs. Outputs Switching

INPUT/OUTPUT CURRENT DIAGRAM



06024-010A

Final

Am29368

1 Megabit Dynamic Memory Controller/Driver (DMC)

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

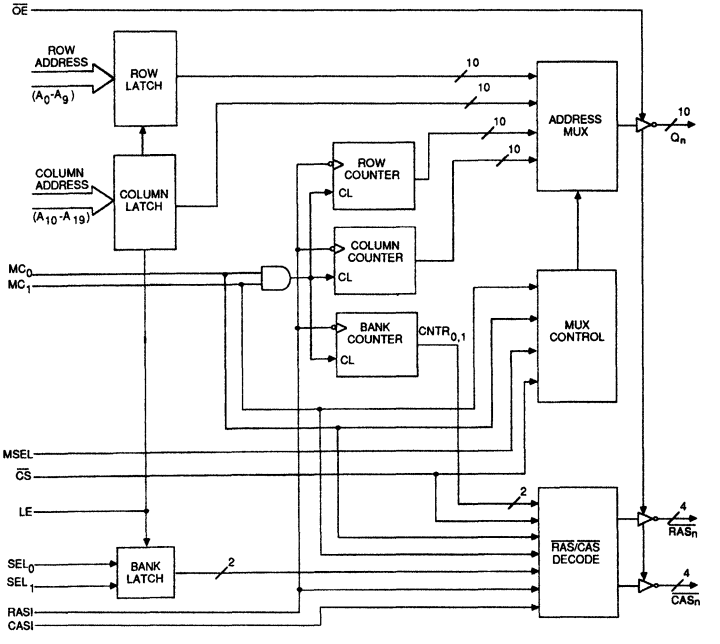
- Provides control for 16K, 64K, and 256K and 1-megabit dynamic RAMs
- Outputs directly drive up to 88 DRAMs, with a guaranteed worst-case limit on the undershoot
- Highest-order two address bits select one of four banks of RAMs
- Separate output enable for multi-channel access to memory
- Supports scrubbing operations and other specialty access modes
- Upgradable from Am2968A 256K DRAM Controller

GENERAL DESCRIPTION

The Am29368 Dynamic Memory Controller/Driver (DMC) is intended to be used with today's high performance memory systems. The DMC acts as the address controller between any processor and dynamic memory array, using its two 10-bit address latches to hold the Row and Column addresses for any DRAM up to 1 megabit. These latches, and the two Row/Column refresh address counters, feed into a 10-bit, 4-input MUX for output to the dynamic RAM address lines. A 2-bit bank select latch for the two high-order address bits is provided to select one each of the four $\overline{RAS}_n$ and $\overline{CAS}_n$ outputs.

The Am29368 has two basic modes of operation, read/write and refresh. In refresh mode, the two counters cycle through the refresh addresses. If memory scrubbing is not being implemented, only the Row Counter is used, generating up to 1024 addresses to refresh a 1024-cycle-refresh 1-megabit DRAM. When memory scrubbing is being performed, both the Row and Column counters are used to perform read-modify-write cycles. In this mode all $\overline{RAS}_n$ outputs will be active while only one $\overline{CAS}_n$ is active at a time.

BLOCK DIAGRAM



BD007002

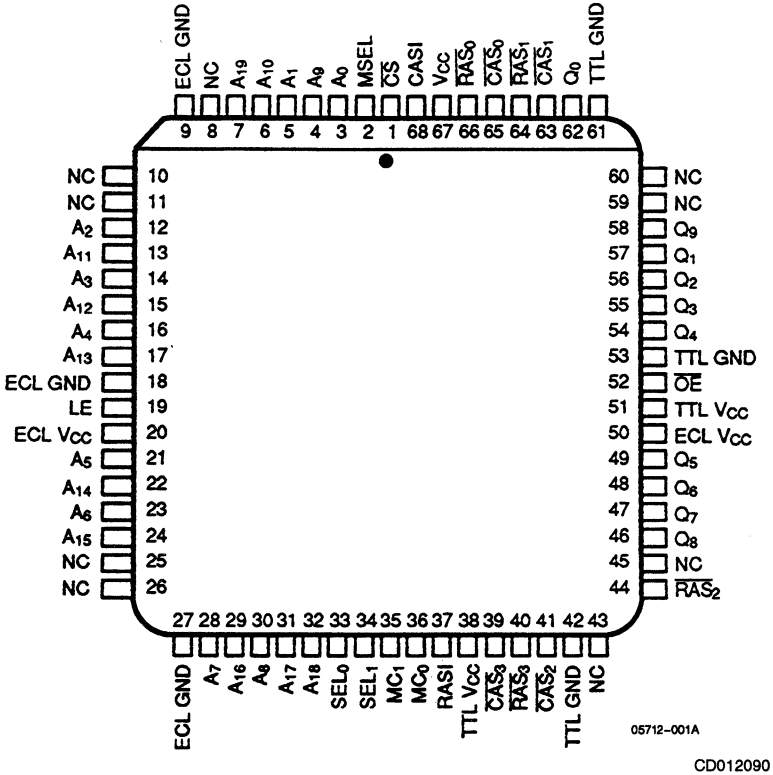
Publication# 05712 Rev. C Amendment /0
Issue Date: January 1990

RELATED AMD PRODUCTS

Part No.	Description
Am29C668	4M Configurable Dynamic Memory Controller/Driver
Am29C660D	12ns 32-Bit Cascadable EDC
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port Multiple Bus Exchange w/Parity
Am29C60A	16-Bit Cascadable EDC, High Speed
Am2968A	256K Dynamic Memory Controller/Driver
Am2971A	100MHz Enhanced Programmable Event Generator
Am2976	11-Bit DRAM Driver
Am2965/6	8-Bit DRAM Driver (Inverting, Non-inverting)
Am29C827A	10-Bit Buffer
Am29C828A	10-Bit Buffer (Inverting)

CONNECTIONS DIAGRAMS
Top View

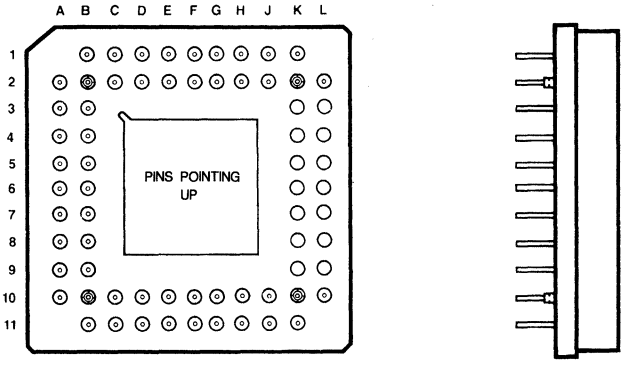
PLCC



CD012090

CONNECTIONS DIAGRAMS (Cont'd.)

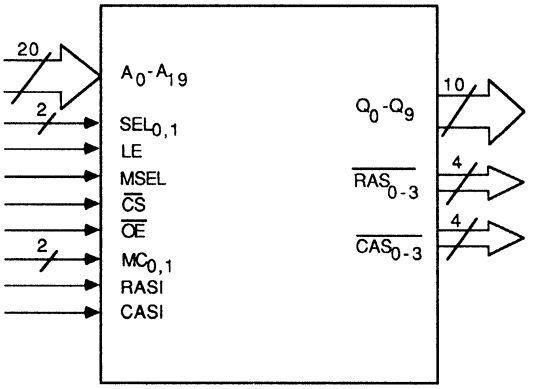
PGA



PO001772

PIN DESIGNATIONS							
(SORTED BY PIN NAME)				(SORTED BY PIN NUMBER)			
PIN NAME	**PIN NO.**	**PIN NAME**	**PIN NO.**	**PIN NO.**	**PIN NAME**	**PIN NO.**	**PIN NAME**
A_0	E-2	**NC**	A-10	**A-2**	GND	**G-1**	V_{CC}
A_1	D-2	**NC**	B-2	**A-3**	A_2	**G-2**	$\overline{RAS_0}$
A_2	A-3	**NC**	B-3	**A-4**	A_3	**G-10**	SEL_1
A_3	A-4	**NC**	B-10	**A-5**	A_4	**G-11**	SEL_0
A_4	A-5	**NC**	B-11	**A-6**	GND	**H-1**	$\overline{CAS_0}$
A_5	A-8	**NC**	C-1	**A-7**	LE	**H-2**	$\overline{RAS_1}$
A_6	A-9	**NC**	C-11	**A-8**	A_5	**H-10**	MC_0
A_7	D-10	**NC**	D-11	**A-9**	A_6	**H-11**	MC_1
A_8	E-11	**NC**	K-2	**A-10**	NC	**J-1**	$\overline{CAS_1}$
A_9	D-1	$\overline{OE}$	K-5	**B-1**	A_{19}	**J-2**	Q_0
A_{10}	C-2	Q_0	J-2	**B-2**	NC	**J-10**	V_{CC}
A_{11}	B-4	Q_1	K-3	**B-3**	NC	**J-11**	RASI
A_{12}	B-5	Q_2	L-3	**B-4**	A_{11}	**K-1**	GND
A_{13}	B-6	Q_3	K-4	**B-5**	A_{12}	**K-2**	NC
A_{14}	B-8	Q_4	L-4	**B-6**	A_{13}	**K-3**	Q_1
A_{15}	B-9	Q_5	L-7	**B-7**	V_{CC}	**K-4**	Q_3
A_{16}	E-10	Q_6	K-7	**B-8**	A_{14}	**K-5**	$\overline{OE}$
A_{17}	F-11	Q_7	L-8	**B-9**	A_{15}	**K-6**	V_{CC}
A_{18}	F-10	Q_8	K-8	**B-10**	NC	**K-7**	Q_6
A_{19}	B-1	Q_9	L-2	**B-11**	NC	**K-8**	Q_8
CASI	F-1	**RASI**	J-11	**C-1**	NC	**K-9**	GND
$\overline{CAS_0}$	H-1	$\overline{RAS_0}$	G-2	**C-2**	A_{10}	**K-10**	$\overline{RAS_3}$
$\overline{CAS_1}$	J-1	$\overline{RAS_1}$	H-2	**C-10**	GND	**K-11**	$\overline{CAS_3}$
$\overline{CAS_2}$	L-10	$\overline{RAS_2}$	L-9	**C-11**	NC	**L-2**	Q_9
$\overline{CAS_3}$	K-11	$\overline{RAS_3}$	K-10	**D-1**	A_9	**L-3**	Q_2
CS	F-2	SEL_0	G-11	**D-2**	A_1	**L-4**	Q_4
GND	A-2	SEL_1	G-10	**D-10**	A_7	**L-5**	GND
GND	A-6	V_{CC}	L-6	**D-11**	NC	**L-6**	V_{CC}
GND	C-10	V_{CC}	B-7	**E-1**	MSEL	**L-7**	Q_5
GND	K-1	V_{CC}	J-10	**E-2**	A_0	**L-8**	Q_7
GND	L-5	V_{CC}	G-1	**E-10**	A_{16}	**L-9**	$\overline{RAS_2}$
GND	K-9	V_{CC}	K-6	**E-11**	A_8	**L-10**	$\overline{CAS_2}$
LE	A-7			**F-1**	CASI		
MC_0	H-10			**F-2**	CS		
MC_1	H-11			**F-10**	A_{18}		
MSEL	E-1			**F-11**	A_{17}		

LOGIC DIAGRAM



LS002881

Die Size: 0.205" x 0.256" Gate Count: 325

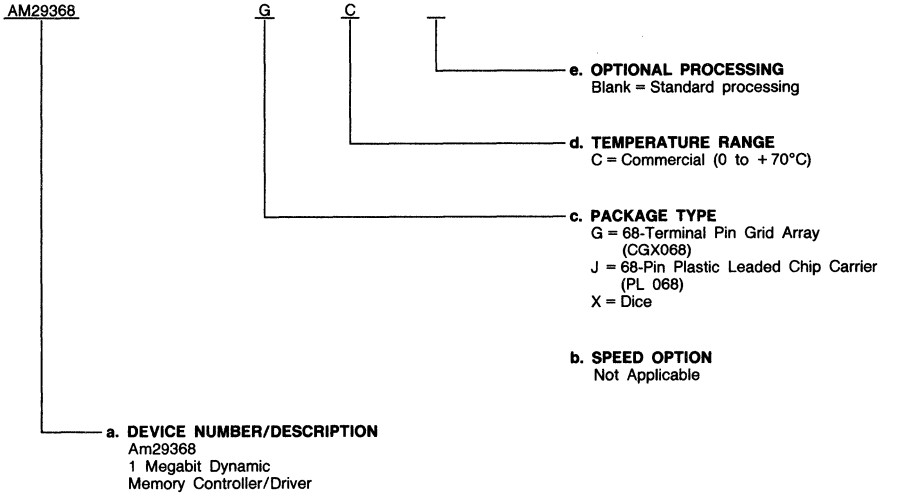
Parameter	PGA	PLCC	Units
θ_{JA}	34	35	°C/Watt
θ_{JC}	N/A	N/A	

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29368	GC, JC, XC

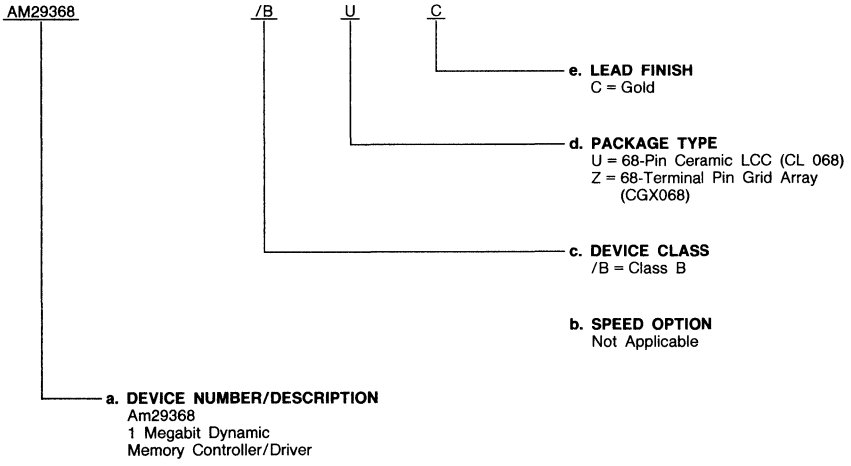
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

ORDERING INFORMATION (Cont'd.)

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements.The order number (Valid Combination) for APL products is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29368	/BUC, /BZC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consist of 1, 2, 3, 7, 8, 9, 10, 11

PIN DESCRIPTION

$A_0 - A_{19}$ Address Inputs (Input (20))
$A_0 - A_9$ are latched in as the 10-bit Row Address for the RAM. These inputs drive $Q_0 - Q_9$ when the Am29368 is in the Read/Write mode and MSEL is LOW. $A_{10} - A_{19}$ are latched in as the Column Address, and will drive $Q_0 - Q_9$ when MSEL is HIGH and the DMC is in the Read/Write mode. The addresses are latched with the Latch Enable (LE) signal.

$\overline{CAS}_{0-3}$ Column Address Strobe (Output (4))
During normal Read/Write cycles the two select bits (SEL_0, SEL_1) determine which $\overline{CAS}_n$ output will go active following CASI going HIGH. When memory scrubbing is performed, only the $\overline{CAS}_n$ signal selected by $CNTR_0$ and $CNTR_1$ will be active (see $\overline{CAS}$ Output Function Table). For non-scrubbing cycles, all four $\overline{CAS}_n$ outputs remain HIGH.

CASI Column Address Strobe (Input (1))
This input going active will cause the selected $\overline{CAS}_n$ output to be forced LOW.

$\overline{CS}$ Chip Select (Input (1))
This active-LOW input is used to select the DMC. When $\overline{CS}$ is active, the Am29368 operates normally in all four modes. When $\overline{CS}$ goes HIGH, the device will not enter the Read/Write mode. This allows more than one Am29368 DMC to control multiple memory banks, thus providing an easy method for expanding the memory size.

LE Latch Enable (Input (1))
This active-HIGH input causes the Row, Column, and Bank Select latches to become transparent, allowing the latches to accept new input data. A LOW input on LE latches the input data, assuming it meets the setup and hold time requirements.

MC_{0-1} Mode Control (Input (2))
These inputs are used to specify which of the four operating modes the DMC should be using. The description of the four operating modes is given in Table 1.

MSEL Multiplexer Select (Input (1))
This input determines whether the Row or Column Address will be sent to the memory address inputs. When MSEL is HIGH the Column Address is selected, while the Row Address is selected when MSEL is LOW. The address may come from either the address latch or refresh address counter depending on $MC_{0,1}$.

$\overline{OE}$ Output Enable (Input (1))
This active-LOW input enables/disables the output signals. When $\overline{OE}$ is HIGH, the outputs of the DMC enter the high-impedance state.

Q_{0-9} Address Outputs (Outputs (10))
These address outputs will feed the DRAM address inputs, and provide drive for memory systems up to 500 picofarads in capacitance.

$\overline{RAS}_{0-3}$ Row Address Strobe (Output (4))
Each one of the Row Address Strobe outputs provides a $\overline{RAS}_n$ signal to one of the four banks of dynamic memory. Each will go LOW only when selected by SEL_0 and SEL_1 and only after RASI goes HIGH. All four go LOW in response to RASI in either of the Refresh modes.

RASI Row Address Strobe (Input (1))
During normal memory cycles, the decoded $\overline{RAS}_n$ output ($\overline{RAS}_0$, $\overline{RAS}_1$, $\overline{RAS}_2$, or $\overline{RAS}_3$) is forced LOW after receipt of RASI. In either Refresh mode, all four $\overline{RAS}_n$ outputs will go LOW following RASI going HIGH.

SEL_{0-1} Bank Select (Input (2))
These two inputs are normally the two higher-order address bits, and are used in the Read/Write mode to select which bank of memory will be receiving the $\overline{RAS}_n$ and $\overline{CAS}_n$ signals after RASI and CASI go HIGH.

FUNCTIONAL DESCRIPTION

Architecture

The Am29368 provides all the required data and refresh addresses needed by the dynamic RAM memory. In normal operation, the Row and Column addresses are multiplexed to the dynamic RAM by using MSEL, with the corresponding $\overline{RAS}_n$ and $\overline{CAS}_n$ signals activated to strobe the addresses into the RAM. High capacitance drivers on the outputs allow the DMC to drive four banks of 16-bit words, including a 6-bit checkword, for a total of 88 DRAMs.

TABLE 1. MODE CONTROL FUNCTION

MC_1	MC_0	Operating Mode
0	0	Refresh without Scrubbing. Refresh cycles are performed with only the Row Counter being used to generate addresses. In this mode, all four $\overline{RAS}_n$ outputs are active while the four $\overline{CAS}_n$ signals are kept HIGH.
0	1	Refresh with Scrubbing/Initialize. During this mode, refresh cycles are done with both the Row and Column counters generating the addresses. MSEL is used to select between the Row and Column counter. All four $\overline{RAS}_n$ go active in response to RASI, while only one $\overline{CAS}_n$ output goes LOW in response to CASI. The Bank Counter keeps track of which $\overline{CAS}_n$ output will go active. This mode is also used on system power-up so that the memory can be written with a known data pattern.
1	0	Read/Write. This mode is used to perform Read/Write cycles. Both the Row and Column addresses are latched and multiplexed to the address output lines using MSEL. SEL_0 and SEL_1 are decoded to determine which $\overline{RAS}_n$ and $\overline{CAS}_n$ will be active.
1	1	Clear Refresh Counter. This mode will clear the three refresh counters (Row, Column, and Bank) on the HIGH-to-LOW transition of RASI, putting them at the start of the refresh sequence. In this mode, all four $\overline{RAS}_n$ are driven LOW upon receipt of RASI so that DRAM wake-up cycles may be performed.

TABLE 2. ADDRESS OUTPUT FUNCTION

$\overline{CS}$	MC_1	MC_0	MSEL	Mode	MUX Output
0	0	0	X	Refresh without Scrubbing	Row Counter Address
	0	1	1	Refresh with Scrubbing	Column Counter Address
			0		Row Counter Address
	1	0	1	Read/Write	Column Address Latch
			0		Row Address Latch
	1	1	X	Clear Refresh Counter	Zero
1	0	0	X	Refresh without Scrubbing	Row Counter Address
	0	1	1	Refresh with Scrubbing	Column Counter Address
			0		Row Counter Address
	1	0	X	Read/Write	Zero
	1	1	X	Clear Refresh Counter	Zero

TABLE 3. $\overline{RAS}$ OUTPUT FUNCTION

RASI	$\overline{CS}$	MC_1	MC_0	SEL_1	SEL_0	Mode	$\overline{RAS}_0$	$\overline{RAS}_1$	$\overline{RAS}_2$	$\overline{RAS}_3$
0	X	X	X	X	X	X	1	1	1	1
1	0	0	0	X	X	Refresh without Scrubbing	0	0	0	0
		0	1	X	X	Refresh with Scrubbing	0	0	0	0
		1	0	0	0	Read/Write	0	1	1	1
				0	1		1	0	1	1
				1	0		1	1	0	1
				1	1		1	1	1	0
		1	1	X	X	Clear Refresh Counter	0	0	0	0
	1	0	0	X	X	Refresh without Scrubbing	0	0	0	0
		0	1			Refresh with Scrubbing	0	0	0	0
		1	0			Read/Write	1	1	1	1
		1	1			Clear Refresh Counter	0	0	0	0

TABLE 4. $\overline{CAS}$ OUTPUT FUNCTION

Inputs						Internal		Outputs			
CASI	$\overline{CS}$	MC_1	MC_0	SEL_1	SEL_0	$CNTR_1$	$CNTR_0$	$\overline{CAS}_0$	$\overline{CAS}_1$	$\overline{CAS}_2$	$\overline{CAS}_3$
1	0	0	0	X	X	X	X	1	1	1	1
		0	1	X	X	0	0	0	1	1	1
						0	1	1	0	1	1
						1	0	1	1	0	1
						1	1	1	1	1	0
		1	0	0	0	X	X	0	1	1	1
				0	1			1	0	1	1
				1	0			1	1	0	1
				1	1			1	1	1	0
		1	1	X	X	X	X	1	1	1	1
	1	0	0	X	X	X	X	1	1	1	1
		0	1	X	X	0	0	0	1	1	1
						0	1	1	0	1	1
						1	0	1	1	0	1
						1	1	1	1	1	0
		1	0	X	X	X	X	1	1	1	1
		1	1								
0	X	X	X	X	X	X	X	1	1	1	1

Input Latches

For those systems where addresses and data are multiplexed onto a single bus, the DMC has latches to hold the address information. The twenty input latches (Row, Column, and Bank Select) are transparent when Latch Enable (LE) is HIGH and will latch the input data meeting setup and hold time requirements when LE goes LOW. For systems where the processor has separate address and data buses, LE may be permanently enabled HIGH.

Refresh Counters

The two 10-bit refresh counters make it possible to support 128, 256, and 512, and 1024 line refresh. External control over which type of refresh is to be performed allows the user maximum flexibility when choosing the refreshing scheme. Transparent (hidden), burst, synchronous or asynchronous refresh modes are all possible.

The refresh counters are advanced at the HIGH-to-LOW transition of RASI. This assures a stable counter output for the next refresh cycle.

Refresh with Error Correction

The Am29368 makes it possible to correct single-bit errors in parallel with performing dynamic RAM refresh cycles. This "scrubbing" of memory can be done periodically as a background routine when the memory is not being used by the processor. In a memory scrubbing cycle ($MC_{1,0} = 01$), the Row Address is strobed into all four banks with all four $\overline{RAS}_n$ outputs going LOW.

The Column Address is strobed into a single bank with the activated $\overline{CAS}_n$ output being selected by the Bank Counter. This type of cycle is used to simultaneously refresh the addressed row in all banks and read and correct (if necessary) one word in memory; thereby reducing the overhead associated with Error Detection and Correction. When doing refresh with memory scrubbing, both the Row and Column counters are multiplexed to the dynamic RAM address lines by using MSEL. Using the Refresh with Memory Scrubbing mode implies the presence of an error correcting facility such as the Am2960A EDC unit. When doing refresh without scrubbing, all four $\overline{RAS}_n$ still go LOW but the $\overline{CAS}_n$ outputs are all driven HIGH so as not to activate the output lines of the memory.

Decoupling

Due to the high switching speeds and high drive capability of the Am29368, it is necessary to decouple the device for proper operation. 1μF multilayer ceramic capacitors are recommended for decoupling (see Figures 1.1 & 1.2). It is important to mount the capacitors as close as possible to the power pins (V_{CC}, GND) to minimize lead inductance and noise. A ground plane is recommended.

It is strongly recommended that the Am29368 be directly surface mounted whenever possible. Should a PLCC, LCC, or PGA socket be required, a one-time-insertion-only socket with minimal lead lengths is necessary for proper device functioning.

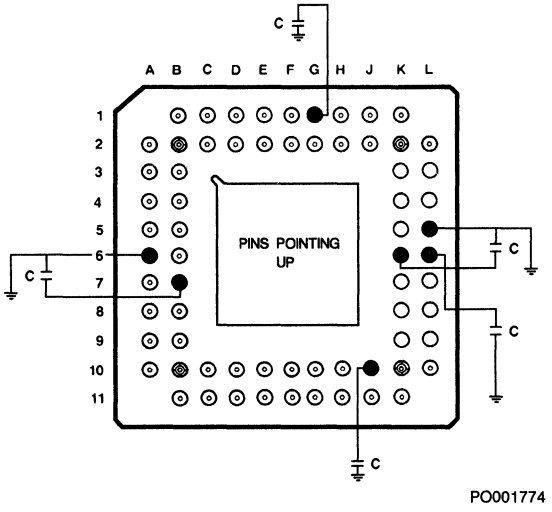


Figure 1.1. PGA Decoupling Connection Diagram

Note: PGA package has a different footprint from LCC or PLCC in a socket.

PGA Hook-Ups

V_{CC}	GND
K6	L5
L6	GP
G1	GP
J10	GP
B7	A6

GP = Ground Plane

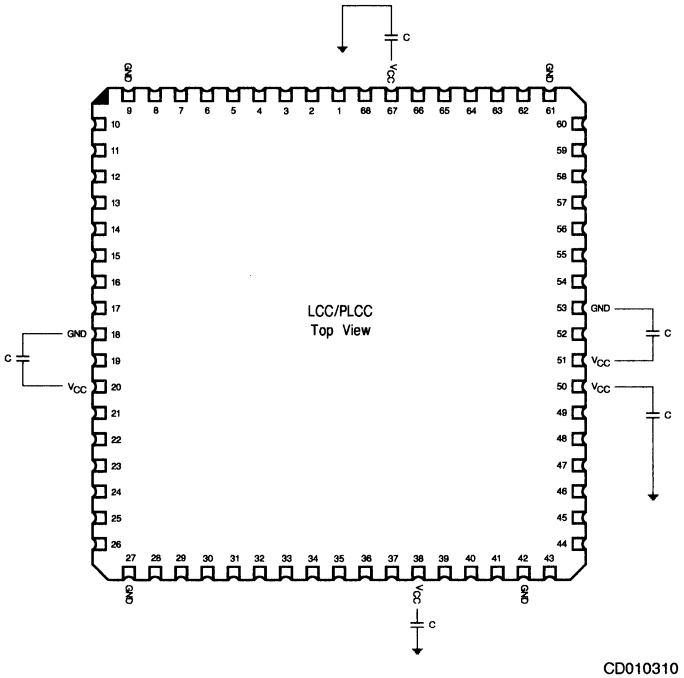


Figure 1.2. LCC/PLCC Decoupling Connection Diagram

V_{ONP}

The guaranteed maximum undershoot voltage of the Am29368 is −1.5 volts. V_{ONP} is measured with respect to ground (see Figure 1.3). Note that the ground of the capacitive load must be the same as for the V_{CC} pin(s). As loading increases, V_{ONP} will approach zero.

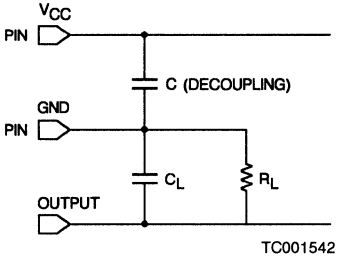
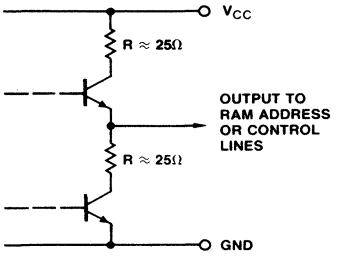


Figure 1.3. V_{ONP} with Respect to Ground

The RAM Driver symmetrical output design offers significant improvement over a standard Schottky output by providing a balanced drive output impedance (≈25 Ω both HIGH and LOW), and by pulling up to MOS V_{OH} levels. External resistors, not required with the RAM Driver, protect standard Schottky drivers from error causing undershoot but also slow the output rise by adding to the internal R.

The RAM Driver is optimized to drive LOW at maximum speed based on safe undershoot control and to drive HIGH with a symmetrical speed characteristic. This is an optimum approach because the dominant RAM loading characteristic is input capacitance.

TYPICAL OUTPUT DRIVER



TC000500

Memory Expansion

With a 10-bit address path, the Am29368 can control up to a four megaword memory when using 1M dynamic RAMs. If a larger memory size is desired, the DMC's chip select ($\overline{CS}$) makes it easy to double the memory size by using two Am29368s. Memory can be increased in four megaword increments by adding another DMC unit.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Ambient Temperature with
Power Applied ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continuous ..... −0.5 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... −0.5 V to +V_{CC} max
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
T_A (Ambient) ..... 0 to +70°C
V_{CC} ..... 5.0 V ±10%
Min ..... 4.50 V
Max ..... 5.50 V

Military* (M) Devices
T_C (Case) ..... −55 to +125°C
V_{CC} ..... 5.0 V ±10%
Min ..... 4.50 V
Max ..... 5.50 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

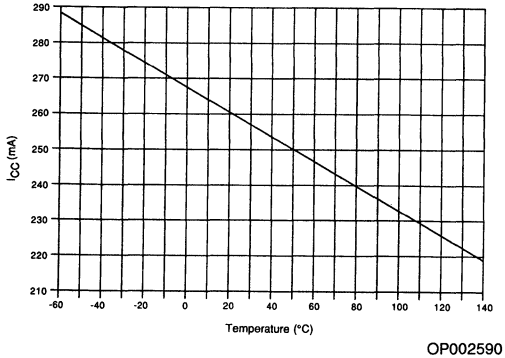
*Military Product 100% tested at T_C = +25°C, +125°C, and −55°C.

DC CHARACTERISTICS over operating ranges unless otherwise specified; Included in Group A, Subgroup 1, 2, 3 tests unless otherwise noted.

Parameters	Descriptions	Test Conditions (Note 1)		Min.	Typ.	Max.	Units
V_{OH}	Output HIGH Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}, I_{OH} = −1 mA	COMM	2.7			Volts
			MIL	2.5			
V_{OL}	Output LOW Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 1 mA			0.5	Volts
			I_{OL} = 12 mA			0.8	
V_{IH}	Input HIGH Level	Guaranteed input logical-HIGH voltage for all inputs		2.0			Volts
V_{IL}	Input LOW Level	Guaranteed input logical-LOW voltage for all inputs				0.8	Volts
V_I	Input Clamp Voltage	V_{CC} = Min., I_{IN} = −18 mA				−1.2	Volts
I_{IL}	Input LOW Current	V_{CC} = Max., V_{IN} = 0.4 V				−400	µA
I_{IH}	Input HIGH Current	V_{CC} = Max., V_{IN} = 2.4 V				20	µA
I_I	Input HIGH Current	V_{CC} = Max., V_{IN} = 5.5 V				100	µA
I_{OZH}	Off-State Current	V_O = 2.4 V				50	µA
I_{OZL}	Off-State Current	V_O = 0.4 V				−50	µA
I_{OL}	Output Sink Current	V_{OL} = 2.0 V		45			mA
I_{SC}	Output Short-Circuit Current	V_{CC} = Max. (Note 2)		−60	−95	−275	mA
I_{CC}	Power Supply Current	V_{CC} = Max.	25°C, 5 V		260		mA
			0°C to +70°C			375	
			−55°C to 125°C			415	

Notes: 1. For conditions shown as Min or Max, use the appropriate value specified under Operating Range for the applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.

TYPICAL I_{CC} Change with Temperature



OP002590

SWITCHING CHARACTERISTICS over operating range unless otherwise specified

Light Capacitive Loading (Small System)

No.	Parameter Symbol	Parameter Description	Test Conditions		Min.	Max.	Units
ADDRESS/RASI/CASI/LE LINES (Note 1)							
1	t_{PD}	A_n to Q_n	C_L = 50 pF		3	20	ns
2	t_{PD}	MSEL to Q_n	C_L = 50 pF		3	20	ns
3	t_{PD}	MC_n to Q_n	C_L = 50 pF		5	24	ns
4	t_{PD}	LE to Q_n	C_L = 50 pF		5	25	ns
5	t_{PD}	$\overline{CS}$ to Q_n	C_L = 50 pF			23	ns
6	t_S	A_n/SEL_n to LE (Note 2)	C_L = 50 pF		5		ns
7	t_H	A_n/SEL_n to LE (Note 2)	C_L = 50 pF		5		ns
8	t_H	MC_1 to RASI	C_L = 50 pF		5		ns
9	t_S	$\overline{CS}$ to RASI	C_L = 50 pF		5		ns
10	t_S	SEL_n to RASI	C_L = 50 pF		5		ns
11	t_{PWL}	RASI, CASI	C_L = 50 pF		20		ns
12	t_{PWH}	RASI, CASI	C_L = 50 pF		20		ns
$\overline{RAS_n}/\overline{CAS_n}$ LINES (Notes 1 and 3)							
13	t_{PD}	RASI to $\overline{RAS_n}$	C_L = 50 pF		3	18	ns
14	t_{PD}	CASI to $\overline{CAS_n}$	C_L = 50 pF		3	17	ns
15	t_{PD}	LE to $\overline{RAS_n}$	C_L = 50 pF			25	ns
16	t_{PD}	LE to $\overline{CAS_n}$	C_L = 50 pF			24	ns
17	t_{PD}	MC_n to $\overline{RAS_n}$	C_L = 50 pF		3	21	ns
18	t_{PD}	MC_n to $\overline{CAS_n}$	C_L = 50 pF		3	19	ns
19	t_{PD}	$\overline{CS}$ to $\overline{RAS_n}$	C_L = 50 pF			20	ns
20	t_{PD}	$\overline{CS}$ to $\overline{CAS_n}$	C_L = 50 pF			19	ns
21	t_{PD}	SEL_n to $\overline{RAS_n}$	C_L = 50 pF			20	ns
22	t_{PD}	SEL_n to $\overline{CAS_n}$	C_L = 50 pF			18	ns
23	t_{SKEW}	[t_{PD} (RASI to $\overline{RAS_n}$) − t_{PD} (A_n to Q_n)] (MC_n = 10)	C_L = 200 pF		−4	11	ns
			C_L = 350 pF		2	11	
24	t_{SKEW}	[t_{PD} (RASI to $\overline{RAS_n}$) − t_{PD} (MC_n to Q_n)] (MC_n = 00, 01)	C_L = 200 pF		−5	11	ns
			C_L = 350 pF		0	11	
25	t_{SKEW}	[t_{PD} (MSEL to $\overline{Q_n}$) − t_{PD} (RASI to $\overline{RAS_n}$)]	C_L = 200 pF		−15	−2	ns
			C_L = 350 pF		−15	−8	
26	t_{SKEW}	[t_{PD} (CASI to $\overline{CAS_n}$) − (MSEL to Q_n)]	C_L = 200 pF		−3	11	ns
			C_L = 350 pF		2	11	
THREE-STATE OUTPUTS/UNDERSHOOT (Note 4)							
27	t_{PLZ}	Output Disable Time from LOW, HIGH	C_L = 50 pF	S = 1		22	ns
28	t_{PHZ}			S = 2		20	
29	t_{PZL}	Output Enable Time from LOW, HIGH	C_L = 50 pF	S = 1		19	ns
30	t_{PZH}			S = 2		21	
31	V_{ONP}	Output Undershoot Voltage (Note 5)	C_L = 50 pF			−1.5	V

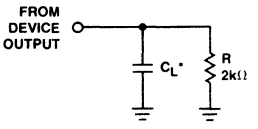
Notes: See notes following the Heavy Capacitive Loading Switching Characteristics table.

Heavy Capacitive Loading (Large Systems)

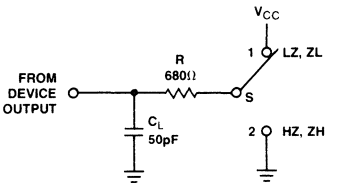
No.	Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Units
ADDRESS/RASI/CASI/LE LINES (Note 1)						
1	t_{PD}	A_n to Q_n	C_L = 500 pF	12	40	ns
2	t_{PD}	MSEL to Q_n	C_L = 500 pF	12	42	ns
3	t_{PD}	MC_n to Q_n	C_L = 500 pF	12	44	ns
4	t_{PD}	LE to Q_n	C_L = 500 pF	12	46	ns
5	t_{PD}	$\overline{CS}$ to Q_n	C_L = 500 pF		45	ns
6	t_S	A_n/SEL_n to LE	C_L = 500 pF	5		ns
7	t_H	A_n/SEL_n to LE (Note 2)	C_L = 500 pF	5		ns
8	t_H	MC_1 to RASI (Note 2)	C_L = 500 pF	5		ns
9	t_S	$\overline{CS}$ to RASI	C_L = 500 pF	5		ns
10	t_S	SEL_n to RASI	C_L = 500 pF	5		ns
11	t_{PWL}	RASI, CASI	C_L = 500 pF	20		ns
12	t_{PWH}	RASI, CASI	C_L = 500 pF	20		ns
$\overline{RAS_n}/\overline{CAS_n}$ LINES (Notes 1 and 3)						
13	t_{PD}	RASI to $\overline{RAS_n}$	C_L = 200 pF	10	25	ns
			C_L = 350 pF	11	33	
14	t_{PD}	CASI to $\overline{CAS_n}$	C_L = 200 pF	10	24	ns
			C_L = 350 pF	11	31	
15	t_{PD}	LE to $\overline{RAS_n}$	C_L = 200 pF		31	ns
			C_L = 350 pF		38	
16	t_{PD}	LE to $\overline{CAS_n}$	C_L = 200 pF		30	ns
			C_L = 350 pF		37	
17	t_{PD}	MC_n to $\overline{RAS_n}$	C_L = 200 pF	10	27	ns
			C_L = 350 pF	11	34	
18	t_{PD}	MC_n to $\overline{CAS_n}$	C_L = 200 pF	10	29	ns
			C_L = 350 pF	11	37	
19	t_{PD}	$\overline{CS}$ to $\overline{RAS_n}$	C_L = 200 pF		25	ns
			C_L = 350 pF		32	
20	t_{PD}	$\overline{CS}$ to $\overline{CAS_n}$	C_L = 200 pF		24	ns
			C_L = 350 pF		31	
21	t_{PD}	SEL_n to $\overline{RAS_n}$	C_L = 200 pF		26	ns
			C_L = 350 pF		34	
22	t_{PD}	SEL_n to $\overline{CAS_n}$	C_L = 200 pF		25	ns
			C_L = 350 pF		33	
23	t_{SKEW}	[t_{PD} (RASI to $\overline{RAS_n}$) − t_{PD} (A_n to Q_n)] (MC_n = 10)	C_L = 200 pF	−17	−3	ns
			C_L = 350 pF	−13	−3	
24	t_{SKEW}	[t_{PD} (RASI to $\overline{RAS_n}$) − t_{PD} (MC_n to Q_n)] (MC_n = 00, 01)	C_L = 200 pF	−16	−1	ns
			C_L = 350 pF	−16	−1	
25	t_{SKEW}	[t_{PD} (MSEL to $\overline{Q_n}$) − t_{PD} (RASI to $\overline{RAS_n}$)]	C_L = 200 pF	−6	11	ns
			C_L = 350 pF	−6	6	
26	t_{SKEW}	[t_{PD} (CASI to $\overline{CAS_n}$) − (MSEL to Q_n)]	C_L = 200 pF	−18	−5	ns
			C_L = 350 pF	−13	−5	

Notes:
1. Reference Figures A and C apply to all parameters except Parameters 8, 9, and 10.
2. Hold times are not tested, but are guaranteed by characterization data. Not included in Group A testing.
3. C_L = 200 pF loading corresponds to 4 banks, 22 bits (16 data bits + 6 check bits).
   C_L = 350 pF loading corresponds to 4 banks, 39 bits (32 data bits + 7 check bits). For additional loading information or to calculate t_{PD} between specified loads, see section labled "NANOSECONDS VERSUS PICOFARADS" following the Switching Waveforms. All parameters with a 200 pF load are not tested but are guaranteed by characterization data. Not included in Group A testing.
4. Not included in Group A testing. Reference Figures B and D apply.
5. V_{ONP} is not production tested but is guaranteed by characterization data. Limit specified is for all outputs switching simultaneously with minimum specified loading. As loading increases, V_{ONP} will approach zero.

SWITCHING TEST CIRCUITS



TC000490

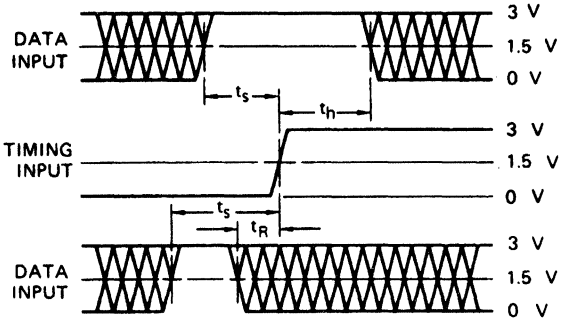


TC000510

*t_{pd} specified at C_L = 50 and 500 pF for Q_n
C_L = 50 and 350 pF for $\overline{RAS_n}$, $\overline{CAS_n}$

A. Capacitive Load Switching **B. Three-State Enable/Disable**

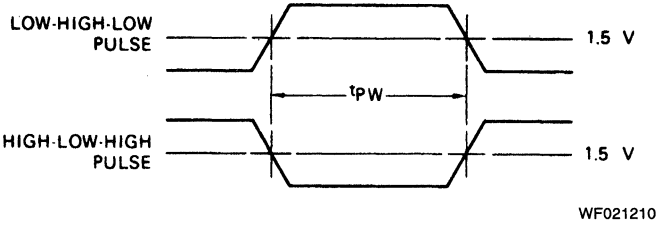
SWITCHING TEST WAVEFORMS



WF021190

A. Setup, Hold, and Release Times

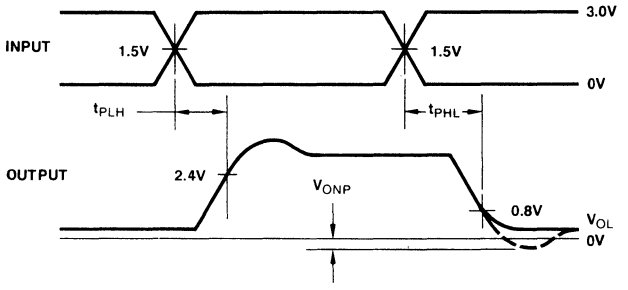
Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched are "don't care" condition.



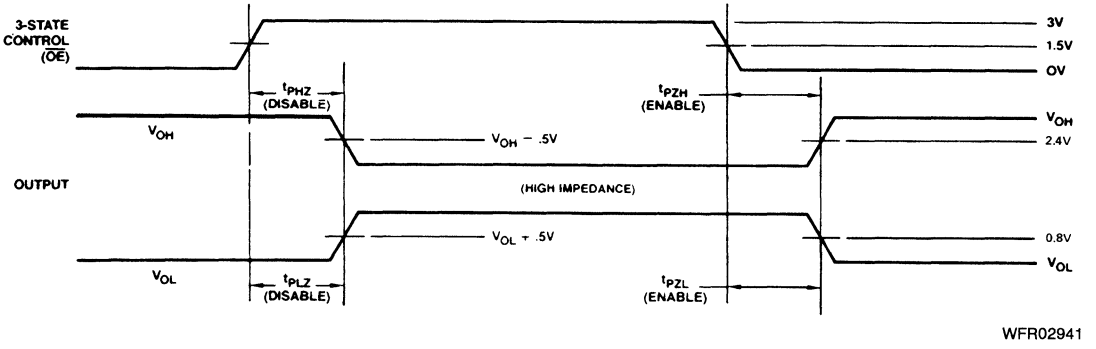
WF021210

B. Pulse Width

SWITCHING TEST WAVEFORMS (Cont'd.)



C. Output Drivers Levels



Note: Decoupling is needed for all AC tests

D. Three-State Control Levels

General Test Notes

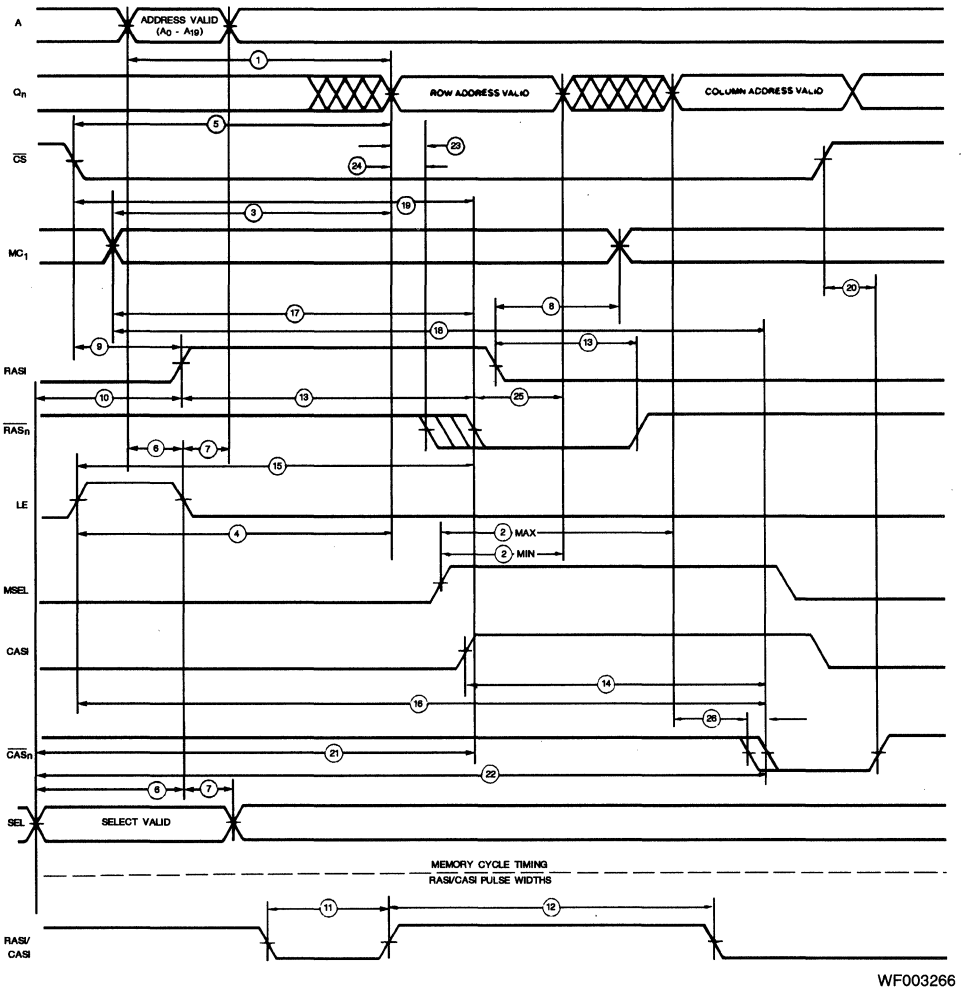
Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Insure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5-8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by 100's of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 4$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. Automatic tester hardware and handler add additional round trip A.C. delay to test measurements. Actual propagation delay testing may incorporate a correlation factor to negate the additional delay.

SWITCHING WAVEFORMS
KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	MUST BE STEADY	WILL BE STEADY
	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
	DON'T CARE; ANY CHANGE PERMITTED	CHANGING; STATE UNKNOWN
	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

KS000010



WF003266

Am29368 Dynamic Memory Controller Timing

Memory Cycle Timing

The relationship between DMC specifications and system timing requirements are shown in Figure 4. T_1, T_2 and T_3 represent the minimum timing requirements at the DMC inputs to guarantee that RAM timing requirements are met and that maximum system performance is achieved.

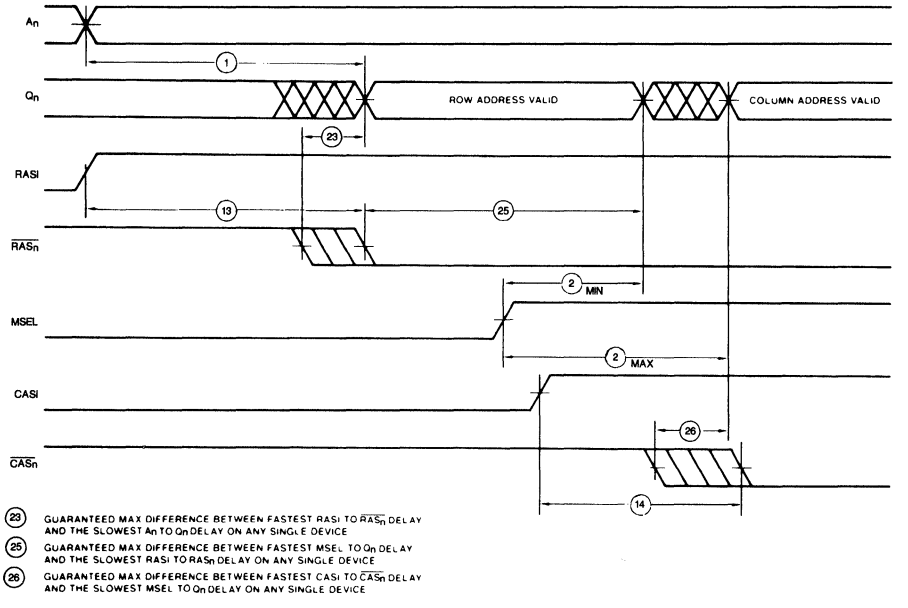
The minimum requirement for T_1, T_2 and T_3 are as follows:

T_1 Min. = $t_{ASR} + t_{23}$
T_2 Min. = $t_{RAH} + t_{25}$
T_3 Min. = $T_2 + t_{26} + t_{ASC}$

See RAM data sheet for applicable values for t_{RAH}, t_{ASC} and t_{ASR}.

Figure 4. Memory Cycle Timing

a. Specifications Applicable to Memory Cycle Timimg (MC_n = 1, 0)



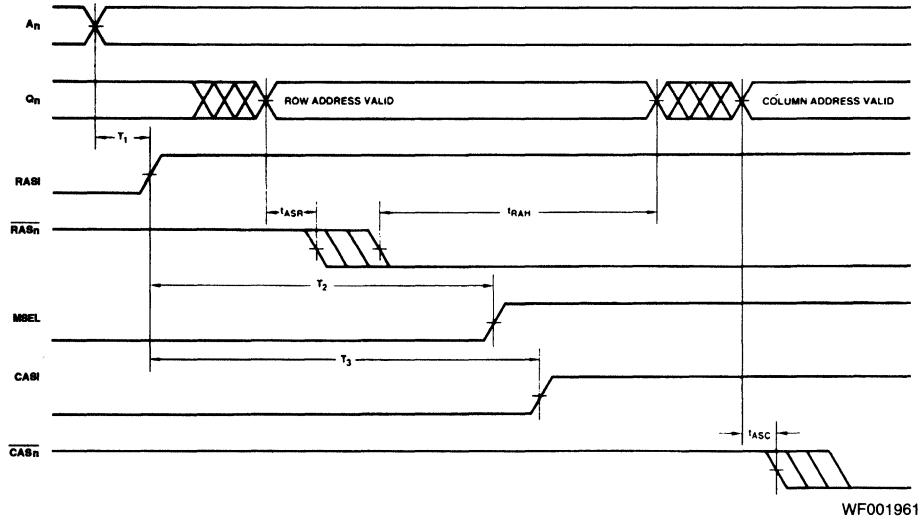
(23) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST RASI TO $\overline{RAS_n}$ DELAY AND THE SLOWEST An TO Qn DELAY ON ANY SINGLE DEVICE

(25) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST MSEL TO Qn DELAY AND THE SLOWEST RASI TO RAS_n DELAY ON ANY SINGLE DEVICE

(26) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST CASI TO $\overline{CAS_n}$ DELAY AND THE SLOWEST MSEL TO Qn DELAY ON ANY SINGLE DEVICE

WF003284

b. Desired System Timing



WF001961

Refresh Cycle Timing

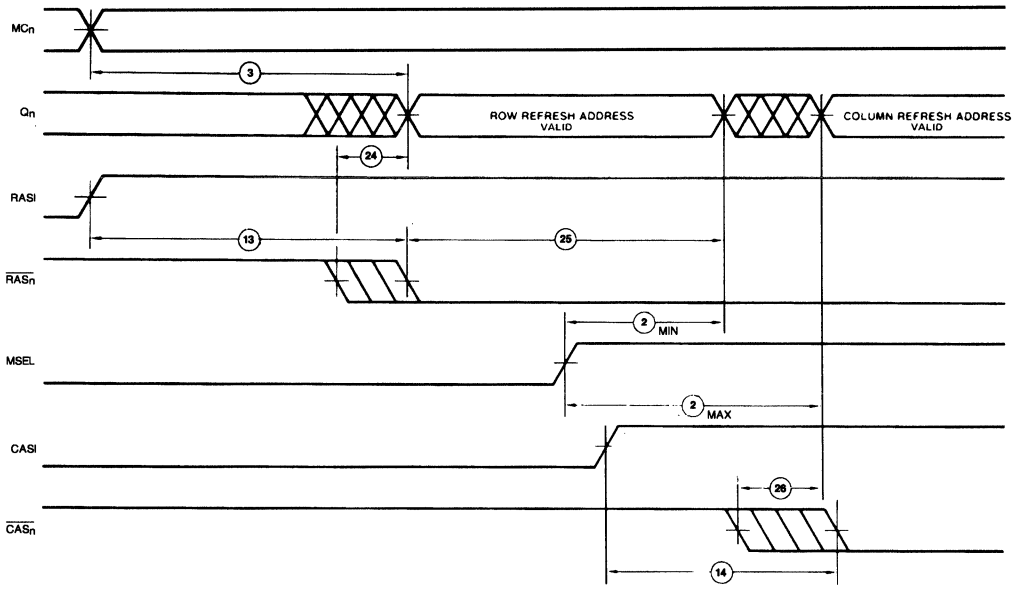
The timing relationships for refresh are shown in Figure 5.

T_4 minimum is calculated as follows:

T_4 Min. = $t_{ASR} + t_{24}$

Figure 5. Refresh Cycle Timing

a. Specifications Applicable to Refresh Cycle Timing (MC_n = 00, 01)



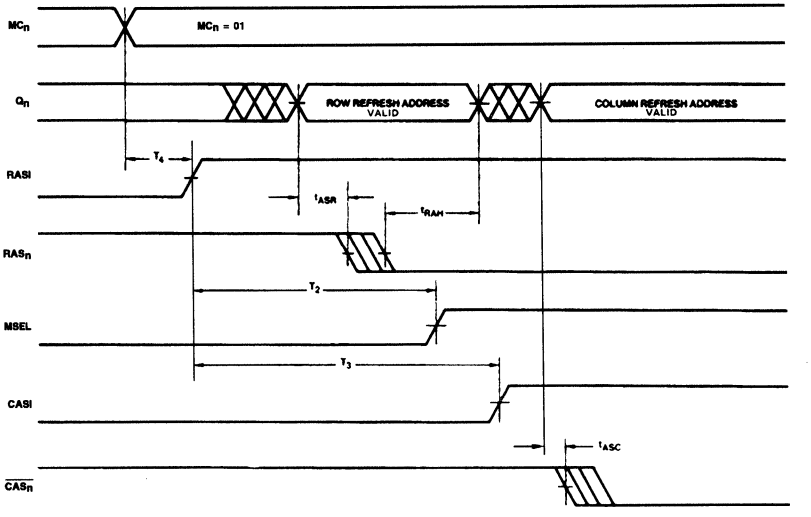
(24) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST RASI TO $\overline{RAS}_n$ DELAY AND THE SLOWEST MC_n TO Q_n DELAY ON ANY SINGLE DEVICE

(25) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST MSEL TO Q_n DELAY AND THE SLOWEST RASI TO $\overline{RAS}_n$ DELAY ON ANY SINGLE DEVICE

(26) GUARANTEED MAX DIFFERENCE BETWEEN FASTEST CASI TO CAS_n DELAY AND THE SLOWEST MSEL TO Q_n DELAY ON ANY SINGLE DEVICE

WF003274

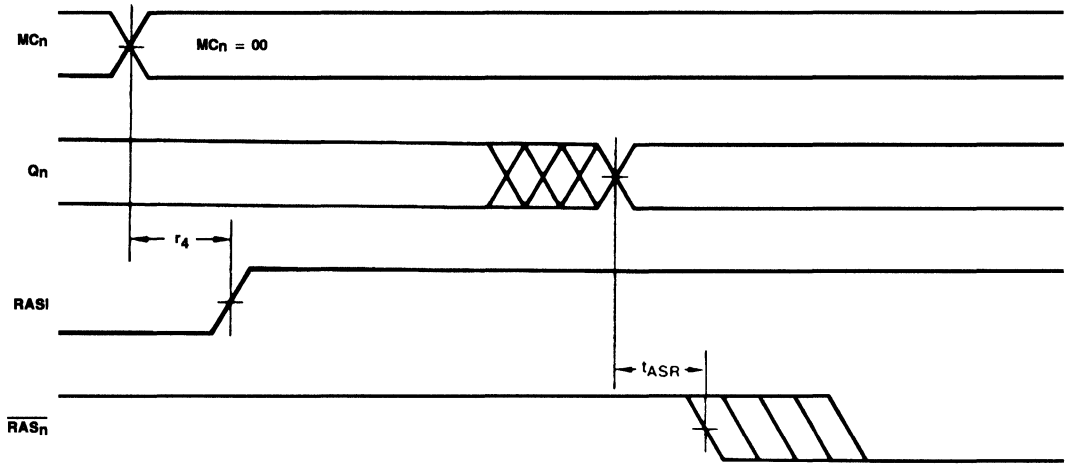
b. Desired Timing: Refresh with Scrubbing



WF001982

Figure 5. Refresh Cycle Timing (Cont'd.)

c. Desired Timing: Refresh without Scrubbing



WF003252

TYPICAL PERFORMANCE CURVES

NANOSECONDS VERSUS PICOFARADS

The Switching Characteristics Tables specify the minimum and maximum propagation delays for effective capacitive loads of 50 pF and 500 pF on Address Lines and 50 pF, 200 pF, and 350 pF on $\overline{RAS_n}$ and $\overline{CAS_n}$ Lines. The upper limits represent the maximum calculated load for the following conditions:

Address Lines: 500 pF = 16 data bits + 6 check bits, four banks

500 pF = 32 data bits + 7 check bits, two banks

$\overline{RAS_n}/\overline{CAS_n}$: 200 pF = 16 data bits + 6 check bits, four banks

350 pF = 32 data bits + 7 check bits, four banks

A more comprehensive analysis of loading is given in the table below.

16-Bit Systems (plus 6 Check Bits for Error Detection and Correction)

DMC Output	No. of DRAMS	Line Drive	Total Required Drive for Indicated Banks			Specified Data Sheet Load
			1	2	4	
Q_n (Address)	16 + 6 = 22	5 pF	100 pF	220 pF	440 pF	500 pF
$\overline{RAS_n}$		8 pF	176 pF	176 pF	176 pF	200 pF
$\overline{CAS_n}$		8 pF	176 pF	176 pF	176 pF	200 pF

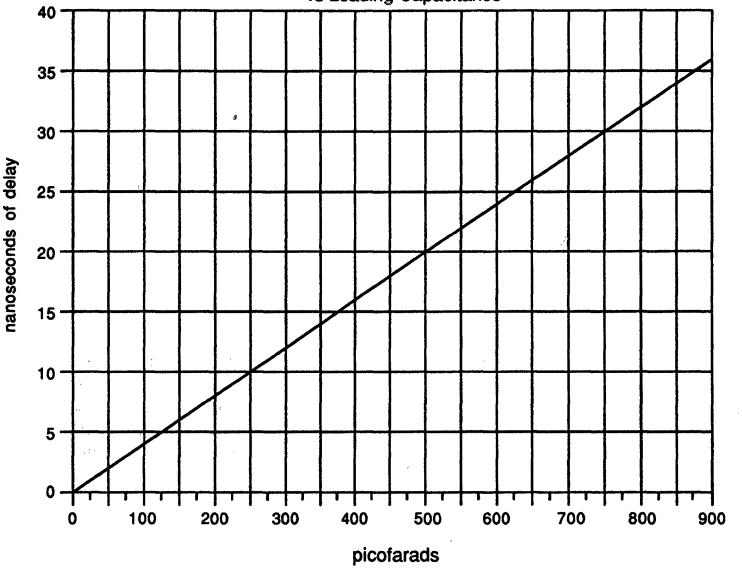
32-Bit Systems (plus 7 Check Bits for Error Detection and Correction)

DMC Output	No. of DRAMS	Line Drive	Total Required Drive for Indicated Banks			Specified Data Sheet Load
			1	2	4	
Q_n (Address)	32 + 7 = 39	5 pF	195 pF	390 pF	780 pF	500 pF
$\overline{RAS_n}$		8 pF	312 pF	312 pF	312 pF	350 pF
$\overline{CAS_n}$		8 pF	312 pF	312 pF	312 pF	350 pF

To calculate propagation delays at loads other than those which are specified, the following graph has been provided. For example, to calculate a system capacitive load of 275 pF, add the delay associated with 75 pF from the graph to the 200 pF $\overline{RAS_n}/\overline{CAS_n}$ delay as specified in the Switching Characteristics Tables. Likewise, add the delay associated with 225 pF from the graph to the 50 pF Q_n (Address) delay in the same AC Tables. This provides a 275 pF load for $\overline{RAS_n}$, $\overline{CAS_n}$, and Q_n lines.

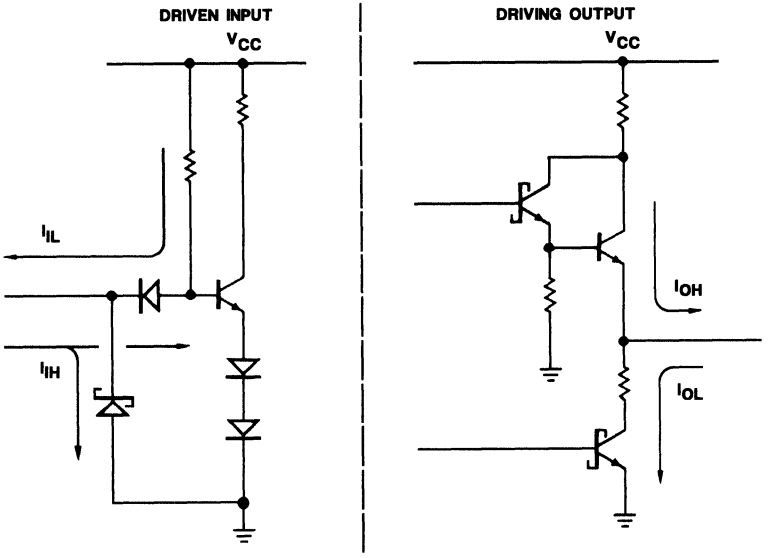
For 32-bit systems using four banks, the Address line delays can be adjusted in a similar fashion to compensate for loads exceeding the 500 pF specified. The specified 350 pF load is sufficient to drive four 32-bit banks $\overline{RAS_n}/\overline{CAS_n}$ without additional delay.

TYPICAL Change in Propagation Delay
vs Loading Capacitance



OP002600

INPUT/OUTPUT CIRCUIT DIAGRAM



IC000791

Final

Am29C60/Am29C60-1/Am29C60A

CMOS Cascadable 16-Bit Error Detection and Correction Unit

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Boosts Memory Reliability
  Corrects all single-bit errors. Detects all double- and some triple-bit errors. Reliability of dynamic RAM systems is increased more than sixty-fold.
- Very High Speed, Low Power
  Perfect for MOS microprocessors, minicomputers, main-frame systems, and engineering workstations.

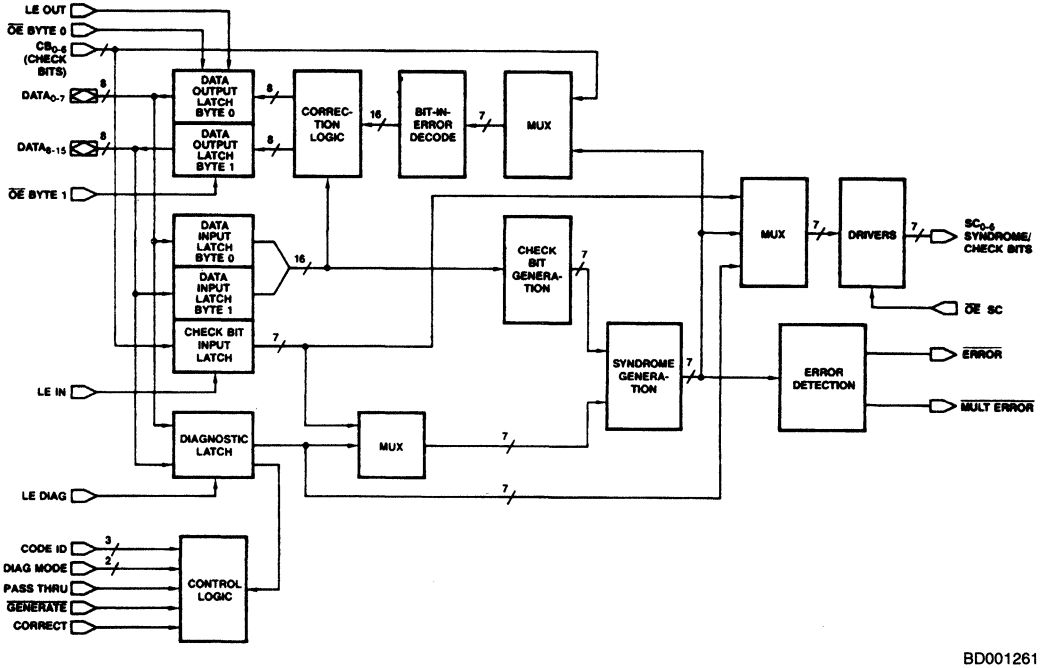
  High-performance systems can use the Am29C60 EDC in check-only mode to avoid memory system slowdown.
- Handles Data Words From 8 Bits to 64 Bits
  The Am29C60 EDC cascades: one EDC for 8 bits or 16 bits, two for 32 bits, four for 64 bits.
- Easy Byte Operations
  Separate byte enables on the data output latch simplify the steps and cut the time required for byte writes.
- Built-In Diagnostics
  The processor may completely exercise the EDC under software control to check for proper operation of the EDC.
- Compatible with the bipolar Am2960 Family.

GENERAL DESCRIPTION

The Am29C60 Error Detection and Correction Unit (EDC) contains the logic necessary to generate check bits on a 16-bit data field according to a modified Hamming Code, and to correct the data word when check bits are supplied. Operating on data read from memory, the Am29C60 corrects any single-bit errors and detects all double- and some triple-bit errors. For 16-bit words, 6 check bits are used. The Am29C60 is expandable to operate on 32-bit words (7 check bits) and 64-bit words (8 check bits). In all configurations, the device makes the error syndrome available on separate outputs for data logging.

The Am29C60 also features two diagnostic modes in which diagnostic data can be forced into portions of the chip to simplify device testing and to execute system diagnostic functions.

BLOCK DIAGRAM

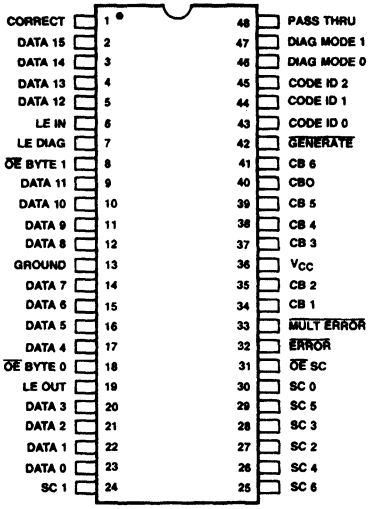


BD001261

Publication# 09544 Rev. C Amendment /0
Issue Date: January 1990

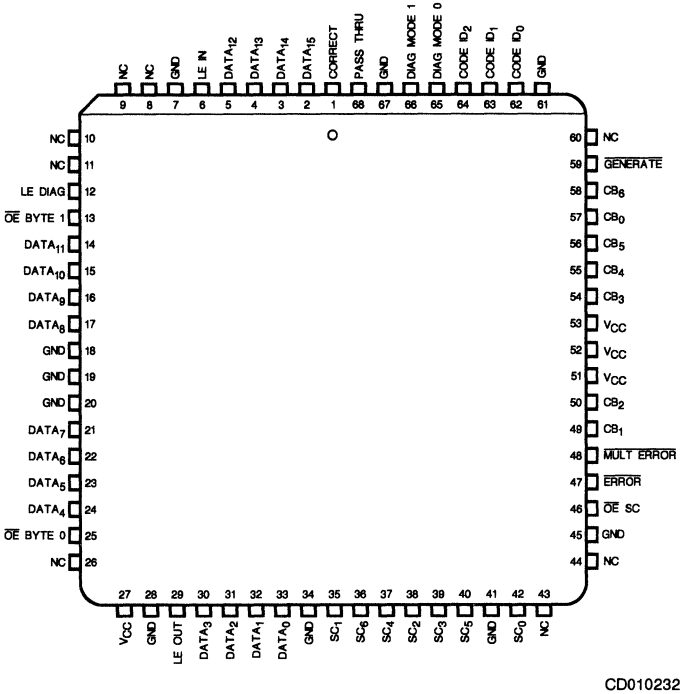
CONNECTION DIAGRAMS
Top View

DIPs



CD001421

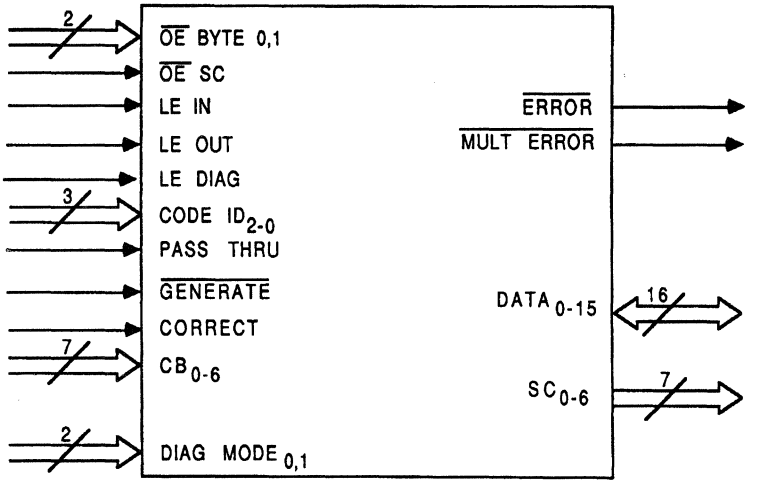
PLCC



CD010232

Note: Pin 1 is marked for orientation.

LOGIC SYMBOL



LS002833

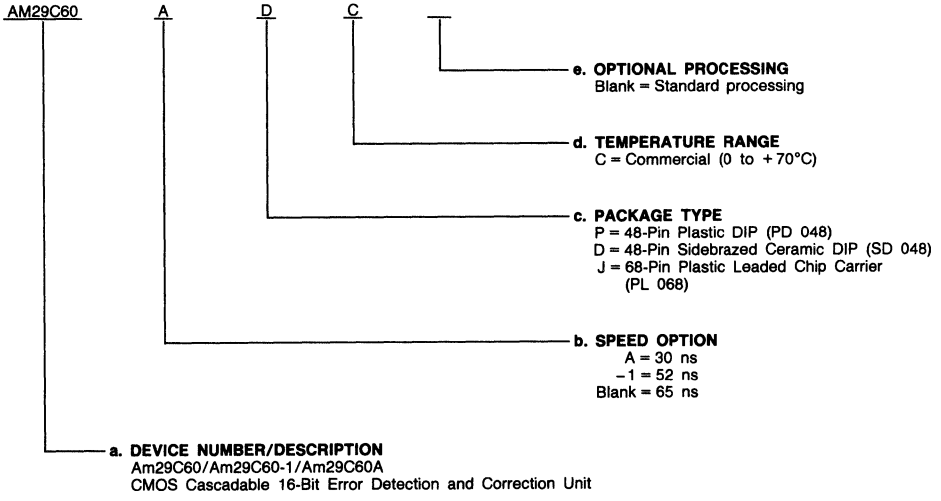
RELATED AMD PRODUCTS

Part No.	Description
Am29C668	4M Configurable Dynamic Memory Controller/Driver
Am29C660D	12 ns 32-Bit Cascadable EDC
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port Multiple Bus Exchange w/Parity
Am29368	1M Dynamic Memory Controller/Driver
Am2968A	256K Dynamic Memory Controller/Driver
Am2971A	100 MHz Enhanced Programmable Event Generator
Am2976	11-Bit DRAM Driver
Am2965/6	8-Bit DRAM Driver (Inverting, Non-inverting)

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29C60	DC, PC, XC, JC
AM29C60-1	
AM29C60A	

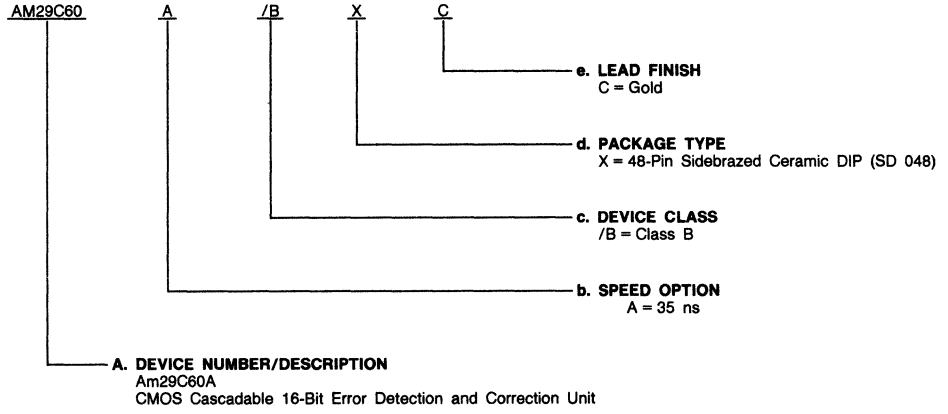
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29C60A	/BXC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consists of Subgroups
1, 2, 3, 7, 8, 9, 10, 11

PIN DESCRIPTION

CB_{0-6} Check Bits (Input)
The check bit lines are used to input check bits for error detection. Also used to input syndrome bits for error correction in 32- and 64-bit configurations.

CODE ID_{2-0} Code Identification (Input)
These three bits identify the size of the total data word to be processed and which 16-bit slice of larger data words a particular EDC is processing. The three allowable data word sizes are 16, 32, and 64 bits, and their respective modified Hamming codes are designated 16/22, 32/39, and 64/72. Special CODE ID input 001 (ID_2, ID_1, ID_0) is also used to instruct the EDC to take the signals CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT, and PASS THRU from the Diagnostic Latch, rather than from the input control lines.

CORRECT Correct (Input)
When HIGH, this signal allows the correction network to correct any single-bit error in the Data Input Latch (by complementing the bit-in-error) before putting it into the Data Output Latch. When the signal is LOW, the EDC drives data directly from the Data Input Latch to the Data Output Latch without correction.

$DATA_{0-15}$ Data (Input/Output; Three State)
These bidirectional data lines provide input to the Data Input Latch and Diagnostic Latch, and receive output from the Data Output Latch. $DATA_0$ is the least significant bit; $DATA_{15}$ the most significant.

DIAG $MODE_{0-1}$ Diagnostic Mode Select (Input)
These two lines control the initialization and diagnostic operation of the EDC.

$\overline{\text{ERROR}}$ Error Detected (Output)
When the EDC is in Detect or Correct Mode, this output goes LOW if one or more syndrome bits are asserted, indicating one or more bit errors in the data or check bits. If no syndrome bits are asserted, there are no errors detected and the output will be HIGH. In Generate Mode, $\overline{\text{ERROR}}$ is forced HIGH. In a 64-bit configuration, $\overline{\text{ERROR}}$ must be externally implemented.

$\overline{\text{GENERATE}}$ Generate Check Bits (Input)
When this input is LOW, the EDC is in the Check Bit Generate Mode. When HIGH, the EDC is in the Detect Mode or Correct Mode.

In the Generate Mode, the circuit generates the check bits or partial check bits specific to the data in the Data Input Latch. The generated check bits are placed on the SC outputs.

In the Detect or Correct Modes, the EDC detects single and multiple errors and generates syndrome bits based on the contents of the Data Input Latch and Check Bit Input Latch. In Correct Mode, single-bit errors are also automatically corrected; corrected data is placed at the inputs of the Data Output Latch. The syndrome result is placed on the SC outputs, and indicates in a coded form the number of errors and the bit-in-error.

LE DIAG Diagnostic Latch Enable (Input)
When this input is HIGH, the Diagnostic Latch follows the 16-bit data on the input lines. When it is LOW, the outputs of the Diagnostic Latch are latched to their previous states. The Diagnostic Latch holds diagnostic check bits and internal control signals for CODE ID_{2-0}, DIAG $MODE_{0-1}$, CORRECT, and PASS THRU.

LE IN Latch Enable – Data Input Latch (Input)
This input controls latching of the input data. When HIGH, the Data Input Latch and Check Bit Input Latch follow the input data and input check bits. When LOW, the Data Input Latch and Check Bit Input Latch are latched to their previous state.

LE OUT Latch Enable – Data Output Latch (Input)
This input controls the latching of the Data Output Latch. When it is LOW, the Data Output Latch is latched to its previous state. When it is HIGH, the Data Output Latch follows the output of the Data Input Latch as modified by the correction logic network. In Correct Mode, single-bit errors are corrected by the network before loading into the Data Output Latch. In Detect Mode, the contents of the Data Input Latch are passed unchanged through the correction network into the Data Output Latch. The inputs to the Data Output Latch are unspecified if the EDC is in Generate Mode.

$\overline{\text{MULT ERROR}}$ Multiple Errors Detected (Output)
When the EDC is in Detect or Correct Mode, this output, if LOW, indicates that two or more bit errors have been detected. If HIGH, either one or no errors have been detected. In Generate Mode, $\overline{\text{MULT ERROR}}$ is forced HIGH. In a 64-bit configuration, $\overline{\text{MULT ERROR}}$ must be externally implemented.

$\overline{\text{OE}}$ BYTE 0, 1 Output Enable Bytes 0, 1 (Input)
These lines control the three-state outputs for each of the two bytes of the Data Output Latch. When LOW, these lines enable the Data Output Latch, and when HIGH, these lines force the Data Output Latch into the high-impedance state. The two enable lines can be separately activated to enable only one byte of the Data Output Latch at a time.

$\overline{\text{OE}}$ SC Output Enable, Syndrome/Check Bits (Input)
When this input is LOW, the three-state output lines SC_{0-6} are enabled. When the input is HIGH, the SC outputs are in the high-impedance state.

PASS THRU Pass Thru (Input)
This line, when HIGH, forces the contents of the Check Bit Input Latch onto the Syndrome/Check Bit outputs (SC_{0-6}) and the unmodified contents of the Data Input Latch onto the inputs of the Data Output Latch.

SC_{0-6} Syndrome/Check Bits (Output; Three State)
These seven lines hold the check/partial-check bits when the EDC is in Generate Mode, and hold the syndrome/partial syndrome bits when the device is in Detect or Correct Modes. These are three-state outputs.

FUNCTIONAL DESCRIPTION

The CMOS Am29C60 and Am29C60-1 are direct parametric and functional replacements for the bipolar Am2960 and Am2960-1 16-bit EDC units used for check-bit generation, error detection, error correction, and diagnostics. The CMOS versions save system power with no loss in performance. The Am29C60A is a performance upgrade of the Am29C60 but is not a parametric equivalent to the bipolar Am2960A.

All three CMOS EDC circuits contain proprietary output buffers to decrease the on-chip-generated noise caused by current changes through fast logic. This minimizes "ground bounce" and ensures proper chip performance of these high-speed CMOS solutions in the system environment.

Please refer to EDC Product Specifications Booklet (Literature #03565E/0) for detailed functional description and applications information.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65°C to +150°C
Temperature (Case)
Under Bias ..... −55°C to +125°C
Supply Voltage to Ground Potential
Continous ..... −0.5 V to +7.0 V
DC Voltage Applied to Outputs For
High Output State ..... −0.5 V to V_{CC} Max.
DC Input Voltage ..... −0.5 V to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Ambient Temperature (T_A) ..... 0°C to +70°C
Supply Voltage ..... +4.75 V to +5.25 V

Military (M) Devices
Case Temperature (T_C) ..... −55°C to +125°C
Supply Voltage ..... +4.5 V to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating range unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions (Note 1)			Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OH} = −300 µA		V_{CC} − 0.2		V
			MIL I_{OH} = −12 mA		2.4		
			COM'L I_{OH} = −15 mA		2.4		
V_{OL}	Output LOW Voltage	V_{CC} = Min., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 300 µA			0.2	V
			MIL I_{OL} = 8 mA			0.5	
			COM'L I_{OL} = 16 mA			0.5	
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for all Inputs (Note 5)			2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for all Inputs (Note 5)				0.8	V
I_{IL}	Input LOW Current	V_{CC} = Max. V_{IN} = 0.5 V	$DATA_{0-15}$ (Note 3)			−10	µA
			All Other Inputs			−10	
I_{IH}	Input HIGH Current	V_{CC} = Max. V_{IN} = 2.7 V	$DATA_{0-15}$ (Note 3)			10	µA
			All Other Inputs			10	
I_I	Input HIGH Current	V_{CC} = Max.,				10	µA
I_{OZH} I_{OZL}	Off State (High-Impedance) Output Current	V_{CC} Max.	$DATA_{0-15}$	V_O = 2.4		40	µA
				V_O = 0.5		−40	
			SC_{0-6}	V_O = 2.4		40	
				V_O = 0.5		−40	
I_{OS}	Output Short-Circuit Current (Note 2)	V_{CC} = Max. V_O = 0 V			−30	−140	mA
I_{CC}	Power Supply (Note 4)	V_{CC} = Max.	COM'L	29C60, -1		60	mA
				29C60A		50	
			MIL	29C60A		50	
		V_{CC} = 5.0 V (Note 6)	T_A = +25°C	29C60, -1		45	
				29C60A		30	
I_{CCQC}	Quiescent Power Supply (CMOS)	V_{CC} = Max.	COM'L	29C60, -1		25	mA
				29C60A		5	
			MIL	29C60A		5	
		V_{CC} = 5.0 V (Note 6)	T_A = +25°C	29C60, -1		5	
				29C60A		2	

Notes: 1. For conditions shown as Min. or Max., use the appropriate value specified under Operating Range for the applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.
3. These are three-state outputs internally connected to TTL inputs. Input characteristics are measured with outputs disabled (high impedance).
4. Worst-case I_{CC} is at minimum temperature. Test conditions:
C_L = 50 pF, f = 10 MHz, V_{IN} = 50% duty cycle for all inputs at 3.4 V and 0.4 V, $\overline{OE}$ = GND.
5. These input levels provide zero noise immunity and should only be tested in a static, noise-free environment.
6. Not production tested. Typical I_{CC} (V_{CC} = 5.0 V and T_A = 25°C) represents nominal units.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Notes 1 and 2)

The following table specifies the guaranteed device performance over the commercial operating range of 0°C to + 70°C (ambient), with V_{CC} 4.75 to 5.25 V. All input switching is between 0 V and 3 V at 1 V/ns, and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

No.	Parameter Symbol	Data Path Description		Am29C60		Am29C60-1		Am29C60A	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
1	t_{PD}	$DATA_{0-15}$ (Note 3)	SC_{0-6}		32		28		20
			$DATA_{0-15}$		65		52		30
			$\overline{ERROR}$		32		25		20
			$\overline{MULT\ ERROR}$		50		50		23
2	t_{PD}	CB_{0-6} (CODE ID_{2-0} 000, 011)	SC_{0-6} (Note 7)		28		23		14
			$DATA_{0-15}$		56		50		25
			$\overline{ERROR}$		29		23		20
			$\overline{MULT\ ERROR}$		47		47		23
3	t_{PD}	CB_{0-6} (CODE ID_{2-0} 010, 100, 101, 110, 111)	SC_{0-6} (Note 7)		28		28		17
			$DATA_{0-15}$		45		34		25
			$\overline{ERROR}$		29		29		20
			$\overline{MULT\ ERROR}$		34		34		23
4	t_{PD}	$\overline{GENERATE}$	SC_{0-6} (Note 7)		35		35		17
			$DATA_{0-15}$		63		63		25
			$\overline{ERROR}$ (Note 7)		36		36		16
			$\overline{MULT\ ERROR}$		55		55		17
5	t_{PD}	CORRECT (Not Internal Control Mode)	SC_{0-6}		–		–		–
			$DATA_{0-15}$		45		45		20
			$\overline{ERROR}$		–		–		–
			$\overline{MULT\ ERROR}$		–		–		–
6	t_{PD}	DIAG MODE (Not Internal Control Mode)	SC_{0-6} (Note 7)		50		50		22
			$DATA_{0-15}$		78		78		27
			$\overline{ERROR}$ (Note 7)		59		59		19
			$\overline{MULT\ ERROR}$ (Note 7)		75		75		21
7	t_{PD}	PASS THRU (Not Internal Control Mode)	SC_{0-6}		36		36		22
			$DATA_{0-15}$		44		44		25
			$\overline{ERROR}$		29		29		18
			$\overline{MULT\ ERROR}$		46		46		21
8	t_{PD}	CODE ID_{2-0}	SC_{0-6}		61		61		23
			$DATA_{0-15}$		90		90		28
			$\overline{ERROR}$		60		60		25
			$\overline{MULT\ ERROR}$		80		80		28

Notes: See notes at end of this section.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C60		Am29C60-1		Am29C60A	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
9	t_{PD}	LE IN (From latched to transparent)	SC_{0-6}		39		39		22
			$DATA_{0-15}$		72		72		32
			$\overline{ERROR}$ (Note 7)		39		39		22
			$\overline{MULT\ ERROR}$		59		59		25
10	t_{PD}	LE OUT (From latched to transparent)	SC_{0-6}		–		–		–
			$DATA_{0-15}$		31		31		13
			$\overline{ERROR}$		–		–		–
			$\overline{MULT\ ERROR}$		–		–		–
11	t_{PD}	LE DIAG (From latched to transparent; not Internal Control Mode)	SC_{0-6}		45		45		22
			$DATA_{0-15}$		78		78		32
			$\overline{ERROR}$		45		45		22
			$\overline{MULT\ ERROR}$		65		65		25
12	t_{PD}	Internal Control Mode: LE DIAG (From latched to transparent)	SC_{0-6}		67		67		28
			$DATA_{0-15}$		96		96		38
			$\overline{ERROR}$		66		66		28
			$\overline{MULT\ ERROR}$		86		86		31
13	t_{PD}	Internal Control Mode: $DATA_{0-15}$ (Via Diagnostic latch)	SC_{0-6}		67		67		28
			$DATA_{0-15}$		96		96		38
			$\overline{ERROR}$		66		66		28
			$\overline{MULT\ ERROR}$		86		86		31
14	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE IN	6		6		5	
15	t_{HOLD}			7		7		3	
16	t_{SET}	CB_{0-6} (Notes 4, 5)		5		5		5	
17	t_{HOLD}			6		6		3	
18	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE OUT	44		34		24	
19	t_{HOLD}			5		5		2	
20	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 000, 011)		35		35		21	
21	t_{HOLD}			0		0		0	
22	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 010, 100, 101, 110, 111)		27		27		21	
23	t_{HOLD}			0		0		0	
24	t_{SET}	$\overline{GENERATE}$ (Notes 4, 5)		42		42		26	
25	t_{HOLD}			0		0		0	
26	t_{SET}	CORRECT (Notes 4, 5)		26		26		22	
27	t_{HOLD}			1		1		0	
28	t_{SET}	DIAG MODE (Notes 4, 5)		69		69		22	
29	t_{HOLD}			0		0		0	
30	t_{SET}	PASS THRU (Notes 4, 5)		26		26		22	
31	t_{HOLD}			0		0		0	
32	t_{SET}	$CODE\ ID_{2-0}$ (Notes 4, 5)		81		81		25	
33	t_{HOLD}			0		0		0	

Notes: See notes at end of this section.

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C60		Am29C60-1		Am29C60A	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.
34	t_{SET}	LE IN (Notes 4, 5)	LE OUT	51		51		28	
35	t_{HOLD}			5		5		0	
36	t_{SET}	DATA $_{0-15}$ (Notes 4, 5)	LE DIAG	6		6		3	
37	t_{HOLD}			8		8		5	
38	t_{EN}	$\overline{OE}$ BYTE 0, 1 ENABLE (Note 6)	$DATA_{0-15}$		30		30		14
39	t_{DIS}				30		30		23
40	t_{EN}	$\overline{OE}$ SC DISABLE (Note 6)	SC_{0-6}		30		30		16
41	t_{DIS}				30		30		21
42	t_{PW}	MINIMUM PULSE WIDTH: LE IN, LE OUT, LE DIAG		15		15		12	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations.
3. Data IN or LE IN to Correct Data Out measurement requires timing as shown in the Switching Waveforms.
4. Setup and Hold times relative to Latch Enables (Latching up data).
5. Setup and Hold times are not tested, but are guaranteed by characterization.
6. Output disable tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.
7. Not production tested. Guaranteed by characterization.

SWITCHING CHARACTERISTICS over **MILITARY** operating range (for APL Products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted) (Notes 1 and 2)

The following table specifies the guaranteed device performance over the Military operating range of −55°C to +125°C (case), with V_{CC} 4.5 to 5.5 V. All input switching is between 0 V and 3 V at 1 V/ns and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

No.	Parameter Symbol	Data Path Description		Am29C60A	
		From Input	To Output	Min.	Max.
1	t_{PD}	$DATA_{0-15}$ (Note 3)	SC_{0-6}		22
			$DATA_{0-15}$		35
			$\overline{ERROR}$		24
			$\overline{MULT\ ERROR}$		27
2	t_{PD}	CB_{0-6} ($CODE\ ID_{2-0}$ 000, 011)	SC_{0-6} (Note 7)		17
			$DATA_{0-15}$		28
			$\overline{ERROR}$		24
			$\overline{MULT\ ERROR}$		27
3	t_{PD}	CB_{0-6} ($CODE\ ID_{2-0}$ 010, 100, 101, 110, 111)	SC_{0-6} (Note 7)		19
			$DATA_{0-15}$		28
			$\overline{ERROR}$		24
			$\overline{MULT\ ERROR}$		27
4	t_{PD}	$\overline{GENERATE}$	SC_{0-6} (Note 7)		20
			$DATA_{0-15}$		28
			$\overline{ERROR}$ (Note 7)		21
			$\overline{MULT\ ERROR}$		25
5	t_{PD}	CORRECT (Not Internal Control Mode)	SC_{0-6}		–
			$DATA_{0-15}$		25
			$\overline{ERROR}$		–
			$\overline{MULT\ ERROR}$		–
6	t_{PD}	DIAG MODE (Not Internal Control Mode)	SC_{0-6} (Note 7)		25
			$DATA_{0-15}$		28
			$\overline{ERROR}$ (Note 7)		21
			$\overline{MULT\ ERROR}$ (Note 7)		24
7	t_{PD}	PASS THRU (Not Internal Control Mode)	SC_{0-6}		25
			$DATA_{0-15}$		28
			$\overline{ERROR}$		21
			$\overline{MULT\ ERROR}$		24
8	t_{PD}	$CODE\ ID_{2-0}$	SC_{0-6}		26
			$DATA_{0-15}$		31
			$\overline{ERROR}$		28
			$\overline{MULT\ ERROR}$		31

Notes: See notes at end of this section.

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C60A	
		From Input	To Output	Min.	Max.
9	t_{PD}	LE IN (From latched to transparent)	SC_{0-6}		24
			$DATA_{0-15}$		37
			$\overline{ERROR}$		26
			$\overline{MULT\ ERROR}$		29
10	t_{PD}	LE OUT (From latched to transparent)	SC_{0-6}		–
			$DATA_{0-15}$		16
			$\overline{ERROR}$		–
			$\overline{MULT\ ERROR}$		–
11	t_{PD}	LE DIAG (From latched to transparent; Not Internal Control Mode)	SC_{0-6}		24
			$DATA_{0-15}$		37
			$\overline{ERROR}$		26
			$\overline{MULT\ ERROR}$		29
12	t_{PD}	Internal Control Mode: LE DIAG (From latched to transparent)	SC_{0-6}		30
			$DATA_{0-15}$		43
			$\overline{ERROR}$		32
			$\overline{MULT\ ERROR}$		35
13	t_{PD}	Internal Control Mode: $DATA_{0-15}$ (Via Diagnostic latch)	SC_{0-6}		30
			$DATA_{0-15}$		43
			$\overline{ERROR}$		32
			$\overline{MULT\ ERROR}$		35
14	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE IN	5	
†15	t_{HOLD}			3	
16	t_{SET}	CB_{0-6} (Notes 4, 5)		5	
†17	t_{HOLD}			3	
18	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE OUT	27	
†19	t_{HOLD}			2	
20	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 000, 011)		24	
†21	t_{HOLD}			0	
22	t_{SET}	CB_{0-6} (Notes 4, 5) (CODE ID 010, 100, 101, 110, 111)		24	
†23	t_{HOLD}			0	
24	t_{SET}	$\overline{GENERATE}$ (Notes 4, 5)		29	
†25	t_{HOLD}			0	
26	t_{SET}	CORRECT (Notes 4, 5)		25	
†27	t_{HOLD}			0	
28	t_{SET}	DIAG MODE (Notes 4, 5)		25	
†29	t_{HOLD}			0	
30	t_{SET}	PASS THRU (Notes 4, 5)		25	
†31	t_{HOLD}			0	
32	t_{SET}	$CODE\ ID_{2-0}$ (Notes 4, 5)		28	
†33	t_{HOLD}			0	

Notes: See notes at end of this section.

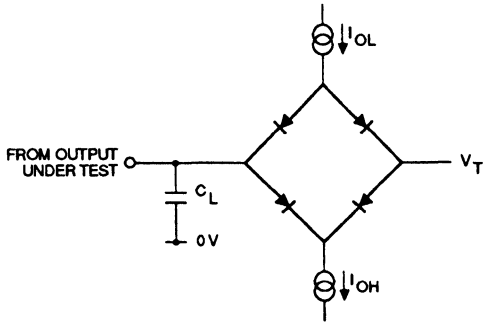
SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C60A	
		From Input	To Output	Min.	Max.
34	t_{SET}	LE IN (Notes 4, 5)	LE OUT	30	
†35	t_{HOLD}			0	
36	t_{SET}	$DATA_{0-15}$ (Notes 4, 5)	LE DIAG	5	
†37	t_{HOLD}			3	
38	t_{EN}	$\overline{OE}$ BYTE 0,1 ENABLE (Note 6)	$DATA_{0-15}$		28
39	t_{DIS}				25
40	t_{EN}	$\overline{OE}$ SC DISABLE (Note 6)	SC_{0-6}		28
41	t_{DIS}				25
42	t_{PW}	MINIMUM PULSE WIDTH: LE IN, LE OUT, LE DIAG		12	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations.
3. Data IN or LE IN to Correct Data Out measurement requires timing as shown in the Switching Waveforms.
4. Setup and Hold times relative to Latch Enables (Latching up data).
5. Setup and Hold times are not tested, but are guaranteed by characterization.
6. Output disable tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.
7. Not production tested. Guaranteed by characterization.

† = Not Included in Group A Tests.

SWITCHING TEST CIRCUIT



AF004810

Notes: 1. $C_L = 50$ pF for all tests except output enable/disable (includes scope probe, wiring, and stray capacitance without device in test fixture).
2. $C_L = 5$ pF for output enable/disable tests.
3. $V_T = 1.5$ V.

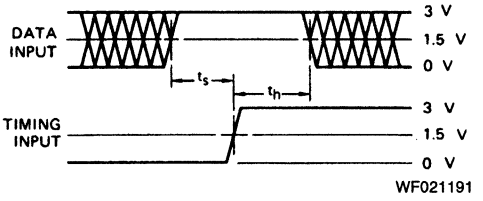
Notes on Testing

Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Ensure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5-8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by hundreds of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 3$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. Changing the CODE ID inputs can cause loss of data in some of the Am29C60 internal latches. Specifically, the entire checkbit latch and bits 6 and 7 of the diagnostic latch are indeterminate after a change in CODE ID inputs.

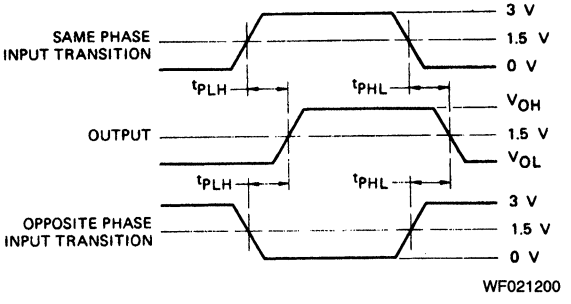
Logic simulations should store "x" (i.e., "don't care") in these bits after CODE ID change. Test programs should reload these registers before they are used.

SWITCHING TEST WAVEFORMS

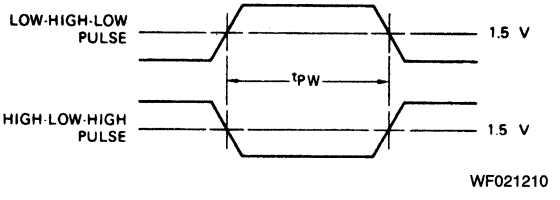


Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched area is don't care condition.

Setup and Hold Times



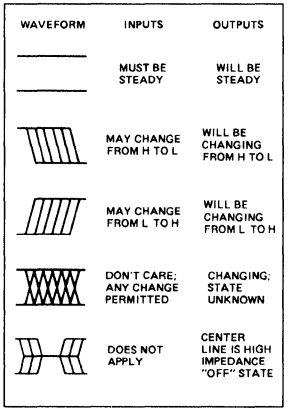
Propagation Delay



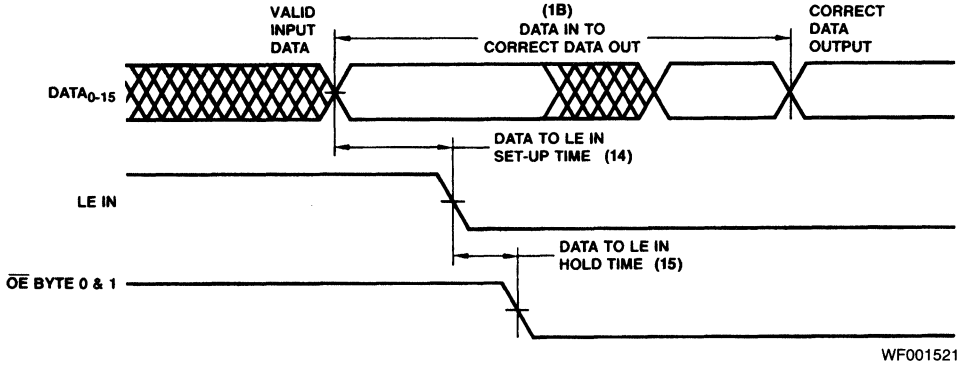
Pulse Width

SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS

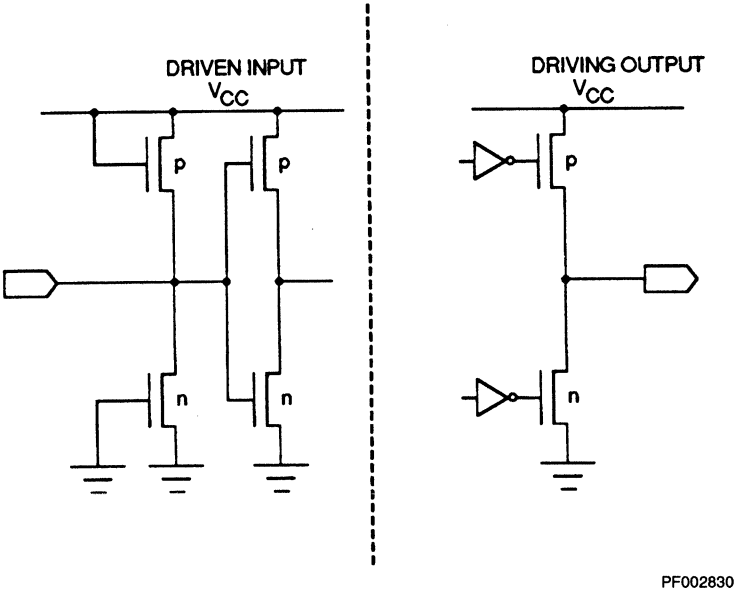


KS000010



DATA IN/LE IN to Correct DATA OUT

EQUIVALENT INPUT/OUTPUT CIRCUIT DIAGRAMS



PF002830

Final

Am29C660

CMOS Cascadable 32-Bit Error Detection and Correction Circuit

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Improves memory reliability
  - Corrects all single-bit errors. Detects all double- and some triple-bit errors
- Very high-speed error detection and correction
  - Down to 12 ns data-in to error detection
- Low-power CMOS process
- Cascadable for data words up to 64 bits
- Simplified byte operations
  - Separate byte enables on the Data Output Latch for fast byte writes
- Built-in diagnostics
  - Proper EDC operation can be verified by the CPU via software control
- Detects gross error conditions of all 1's or all 0's

GENERAL DESCRIPTION

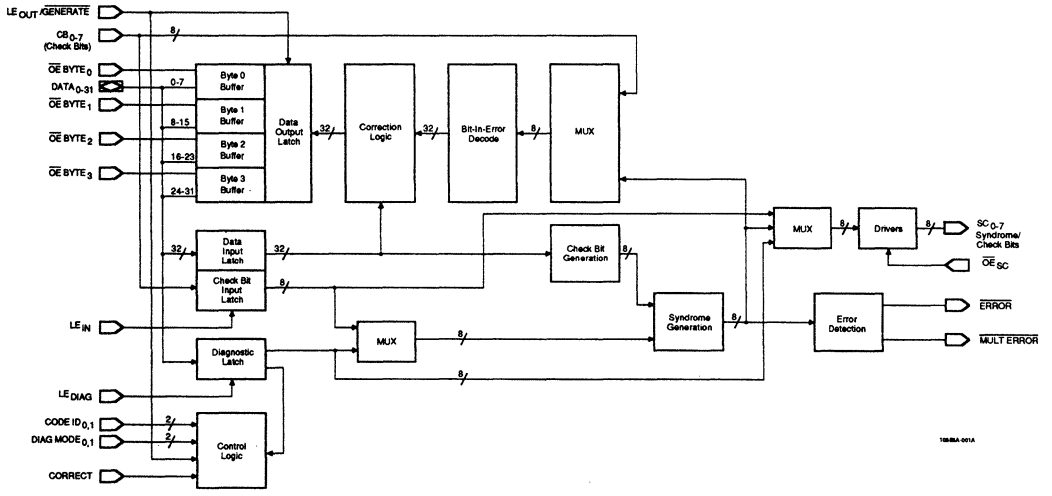
The Am29C660 CMOS Cascadable 32-Bit Error Detection and Correction Circuit (EDC) contains the logic necessary to generate check bits on a 32-bit data field according to a modified Hamming Code, and to correct the data word when check bits are supplied. Operating on data read from memory, the Am29C660 detects and corrects all single-bit errors and detects all double- and some triple-bit errors. For 32-bit words, 7 check bits are used.

The Am29C660 is expandable to operate on 64-bit data words (8 check bits). In both configurations, the device makes error syndromes available on separate outputs for error logging.

The Am29C660 also features two diagnostic modes in which diagnostic data can be forced into portions of the chip to simplify device testing and to execute system diagnostic functions.

When used with the Am2968A/Am29368/Am29C668 Dynamic Memory Controllers, the Am29C660 can perform AMD's invented memory "scrubbing" operation to provide highest data integrity.

BLOCK DIAGRAM



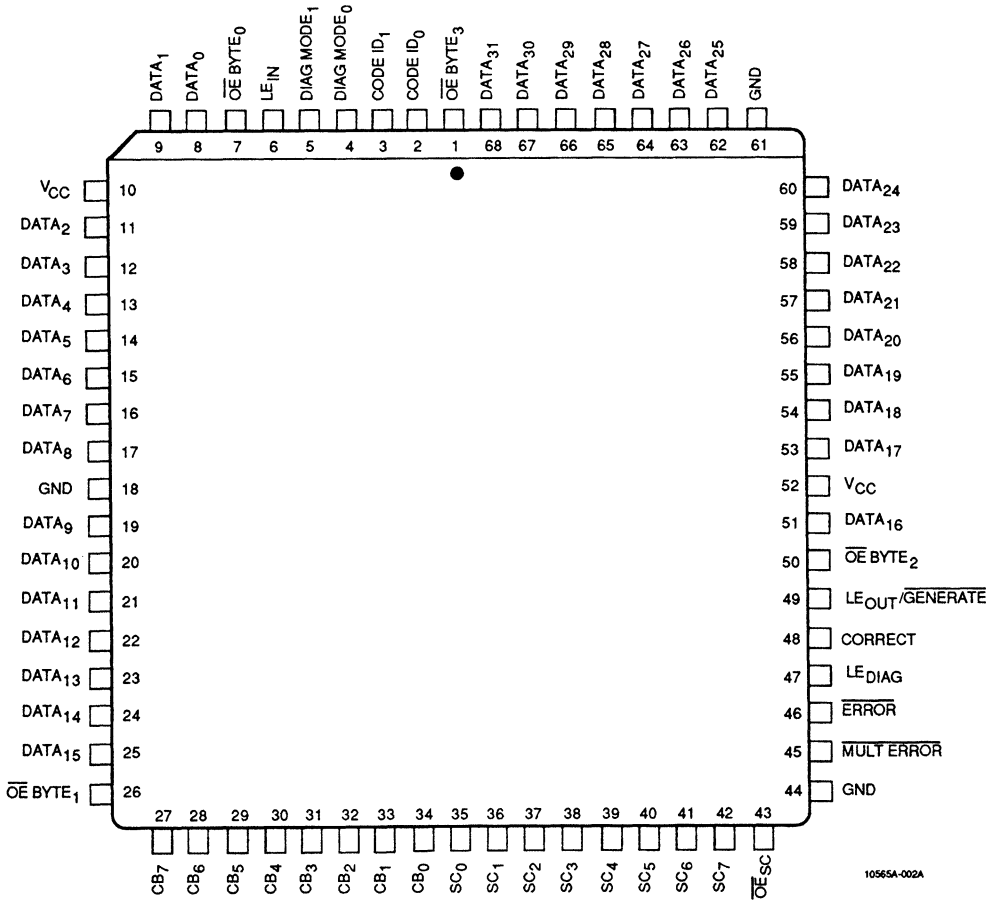
BD007960

RELATED AMD PRODUCTS

Part No.	Description
Am29C668	4M Configurable Dynamic Memory Controller/Driver
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port Multiple Bus Exchange w/Parity
Am29C60A	16-Bit Cascadable EDC, High Speed
Am29368	1M Dynamic Memory Controller/Driver
Am2968A	256K Dynamic Memory Controller/Driver
Am2971A	100 MHz Enhanced Programmable Event Generator
Am2976	11-Bit DRAM Driver
Am2965/6	8-Bit DRAM Driver (Inverting, Non-inverting)

CONNECTION DIAGRAMS
Top View

PLCC

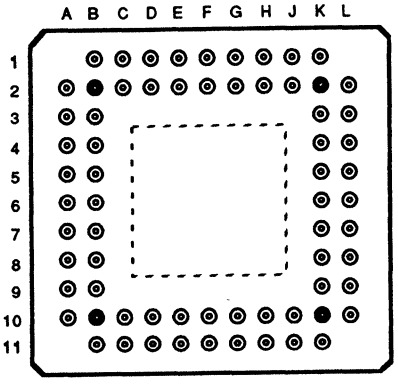


CD011500

Note: Pin 1 is marked for orientation (PLCC only).

PGA

(Pins facing up)



10565A-003A

CD011560

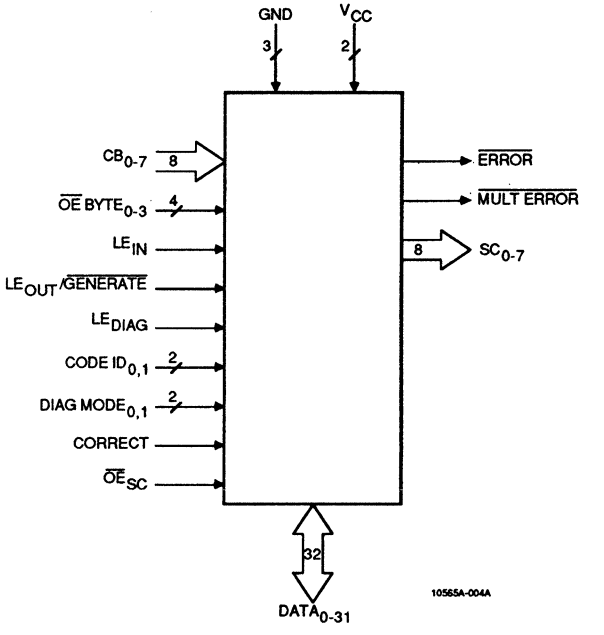
PGA PIN DESIGNATIONS
(Sorted by Pin No.)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-2	$DATA_1$	B-9	$DATA_{26}$	F-10	V_{CC}	K-4	CB_3
A-3	$DATA_0$	B-10	GND	F-11	$DATA_{17}$	K-5	CB_1
A-4	LE_{IN}	B-11	$DATA_{24}$	G-1	$DATA_{11}$	K-6	SC_0
A-5	DIAG $MODE_0$	C-1	$DATA_4$	G-2	$DATA_{10}$	K-7	SC_2
A-6	CODE ID_0	C-2	$DATA_3$	G-10	$\overline{OE}$ $BYTE_2$	K-8	SC_4
A-7	$DATA_{31}$	C-10	$DATA_{22}$	G-11	$DATA_{16}$	K-9	SC_6
A-8	$DATA_{29}$	C-11	$DATA_{23}$	H-1	$DATA_{13}$	K-10	GND
A-9	$DATA_{27}$	D-1	$DATA_6$	H-2	$DATA_{12}$	K-11	$\overline{MULT\ ERROR}$
A-10	$DATA_{25}$	D-2	$DATA_5$	H-10	CORRECT	L-2	CB_6
B-1	$DATA_2$	D-10	$DATA_{20}$	H-11	$LE_{OUT}/\overline{GENERATE}$	L-3	CB_4
B-2	V_{CC}	D-11	$DATA_{21}$	J-1	$DATA_{15}$	L-4	CB_2
B-3	$\overline{OE}$ $BYTE_0$	E-1	$DATA_8$	J-2	$DATA_{14}$	L-5	CB_0
B-4	DIAG $MODE_1$	E-2	$DATA_7$	J-10	$\overline{ERROR}$	L-6	SC_1
B-5	CODE ID_1	E-10	$DATA_{18}$	J-11	LE_{DIAG}	L-7	SC_3
B-6	$\overline{OE}$ $BYTE_3$	E-11	$DATA_{19}$	K-1	$\overline{OE}$ $BYTE_1$	L-8	SC_5
B-7	$DATA_{30}$	F-1	$DATA_9$	K-2	CB_7	L-9	SC_7
B-8	$DATA_{28}$	F-2	GND	K-3	CB_5	L-10	$\overline{OE}_{SC}$

(Sorted by Pin Name)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
L-5	CB_0	D-1	$DATA_6$	C-11	$DATA_{23}$	H-11	$LE_{OUT}/\overline{GENERATE}$
K-5	CB_1	E-2	$DATA_7$	B-11	$DATA_{24}$	K-11	$\overline{MULT\ ERROR}$
L-4	CB_2	E-1	$DATA_8$	A-10	$DATA_{25}$	B-3	$\overline{OE}$ $BYTE_0$
K-4	CB_3	F-1	$DATA_9$	B-9	$DATA_{26}$	K-1	$\overline{OE}$ $BYTE_1$
L-3	CB_4	G-2	$DATA_{10}$	A-9	$DATA_{27}$	G-10	$\overline{OE}$ $BYTE_2$
K-3	CB_5	G-1	$DATA_{11}$	B-8	$DATA_{28}$	B-6	$\overline{OE}$ $BYTE_3$
L-2	CB_6	H-2	$DATA_{12}$	A-8	$DATA_{29}$	L-10	$\overline{OE}_{SC}$
K-2	CB_7	H-1	$DATA_{13}$	B-7	$DATA_{30}$	K-6	SC_0
A-6	CODE ID_0	J-2	$DATA_{14}$	A-7	$DATA_{31}$	L-6	SC_1
B-5	CODE ID_1	J-1	$DATA_{15}$	A-5	DIAG $MODE_0$	K-7	SC_2
H-10	CORRECT	G-11	$DATA_{16}$	B-4	DIAG $MODE_1$	L-7	SC_3
A-3	$DATA_0$	F-11	$DATA_{17}$	J-10	$\overline{ERROR}$	K-8	SC_4
A-2	$DATA_1$	E-10	$DATA_{18}$	B-10	GND	L-8	SC_5
B-1	$DATA_2$	E-11	$DATA_{19}$	F-2	GND	K-9	SC_6
C-2	$DATA_3$	D-10	$DATA_{20}$	K-10	GND	L-9	SC_7
C-1	$DATA_4$	D-11	$DATA_{21}$	J-11	LE_{DIAG}	B-2	V_{CC}
D-2	$DATA_5$	C-10	$DATA_{22}$	A-4	LE_{IN}	F-10	V_{CC}

LOGIC SYMBOL



LS003250

Die Size: 0.174″ x 0.176″
Gate Count: 1670

Package Information

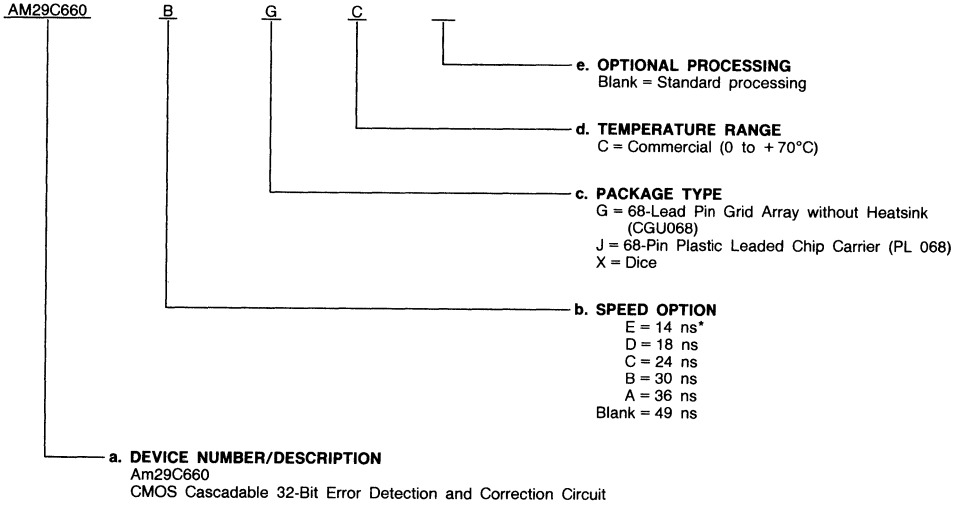
Parameter	PGA	PLCC	Unit
θ_{JA}	34	35	°C/W
θ_{JC}	–	N/A	°C/W

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29C660	GC, JC, XC
AM29C660A	
AM29C660B	
AM29C660C	
AM29C660D	
AM29C660E*	

*in development

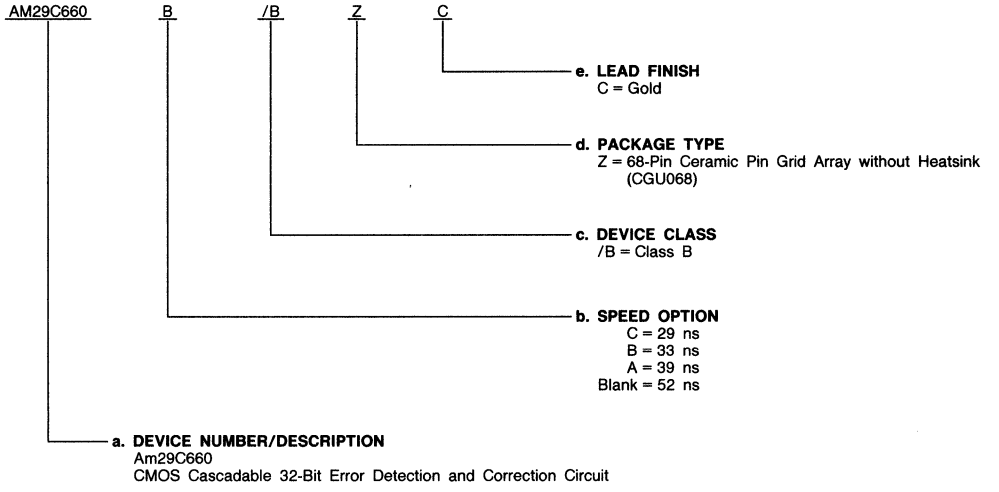
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29C660	/BZC
AM29C660A	
AM29C660B	
AM29C660C	

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consist of Subgroups
1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

CB_{0-7} Check Bits (Inputs)
These eight check bit lines are used to input bits for error detection. They are also used to input syndrome bits for error correction in the 64-bit configuration.

CODE $ID_{0,1}$ Code Identification (Inputs)
These two code identification inputs identify the size of the total data word to be processed. The two allowable data word sizes are 32 and 64 bits and their respective modified Hamming Codes are designated 32/39 and 64/72. The CODE ID inputs are also used to instruct the EDC that the CODE $ID_{0,1}$, DIAG $MODE_{0,1}$, and CORRECT signals are to be taken from the Diagnostic Latch, rather than from the input control lines (Reference Table 1).

CORRECT Correct (Input; Active HIGH)
This signal allows the correction network to correct any single-bit error in the Data Input Latch before putting it into the Data Output Latch by complementing the bit-in-error. When the signal is LOW, the EDC routes data directly from the Data Input Latch to the Data Output Latch without correction.

$DATA_{0-31}$ Data (Inputs/Outputs; Three-State)
These bidirectional data lines provide input to the Data Input Latch and Diagnostic Latch, and receive output from the Data Output Latch. $DATA_0$ is the least significant bit and $DATA_{31}$ is the most significant bit.

DIAG $MODE_{0,1}$ Diagnostic Mode Select (Inputs; Active HIGH)
These two lines control the initialization and diagnostic operation of the EDC circuit (Reference Table 2).

$\overline{\text{ERROR}}$ Error Detection Flag (Output; Active LOW)
When the EDC is in the Detect or Detect/Correct Mode, this output goes LOW if one or more syndrome bits are non-zero, indicating one or more data and/or check bits are in error. If all syndrome bits are zero, there are no errors detected and the output will be HIGH. In the Generate Mode, $\overline{\text{ERROR}}$ is forced HIGH.

GND (3) 0-V Power Supply
These pins are the 0-V power supply to the EDC circuit. All grounds must be connected during device operation.

LE_{DIAG} Diagnostic Latch Enable (Input)
This is the latch enable for the Diagnostic Latch. When HIGH, the Diagnostic Latch follows the 32-bit data on the input lines. When LOW, the outputs of the Diagnostic Latch are latched to their previous states. The Diagnostic Latch holds diagnostic check bits and internal control signals for CODE $ID_{0,1}$, DIAG $MODE_{0,1}$, and CORRECT.

LE_{IN} Latch Enable — Data Input Latch (Input)
This input controls latching of the input data. When HIGH, the Data Input Latch and Check Bit Input Latch follow the input data and input check bits, respectively. When LOW, the Data Input Latch and Check Bit Input Latch are latched to their previous states.

$LE_{OUT}/\overline{\text{GENERATE}}$ Latch Enable — Data Output Latch (Input)/$\overline{\text{Generate Check Bits}}$ (Input; Active LOW)
This is a multifunction pin. When it is LOW, it operates in the Check Bit Generate Mode. In this mode, the device generates the check bits or partial check bits specific to the data in the Data Input Latch. The generated check bits are placed on the syndrome/check bit outputs. The Data Output Latch is latched to its previous state when this pin is LOW.

When HIGH, the device is in the Detect or Detect/Correct Mode. In the Detect Mode, the device detects single and multiple errors, and generates syndrome bits specific to the data in the Data Input Latch and Check Bit Input Latch. In the Detect/Correct Mode, single-bit errors are automatically corrected, with the corrected data placed at the inputs of the Data Output Latch. The syndrome result is placed on the syndrome/check bit outputs and indicates, in a coded form, the number of errors and the specific bit in error. When HIGH, the Data Output Latch follows the output of the Data Input Latch as modified by the correction logic network.

In the Detect/Correct Mode, single-bit errors are corrected by the network before being loaded into the Data Output Latch. In the Detect Mode, the contents of the Data Input Latch are passed through the correction network unchanged into the Data Output Latch. The Data Output Latch is disabled with its contents unchanged if the EDC is in the Generate Mode.

$\overline{\text{MULT ERROR}}$ $\overline{\text{Multiple Error Detection Flag}}$ (Output; Active LOW)
When LOW in the Detect or Detect/Correct Mode, this output indicates that two or more bit errors have been detected. If HIGH, either one or no errors have been detected. In the Generate Mode, $\overline{\text{MULT ERROR}}$ is forced HIGH.

$\overline{OE}$ $BYTE_{0-3}$ Output Enable Bytes 0 – 3 (Inputs; Active LOW)
These lines control the three-state output buffers for each of the four bytes of the Data Output Latch. When LOW, they enable the output buffer of the Data Output Latch. When HIGH, they force the Data Output Latch buffer into the high-impedance state. Any number of bytes of the Data Output Latch is easily activated by separately selecting any of the four enable lines.

$\overline{OE}_{SC}$ $\overline{\text{Output Enable}}$, Syndrome/Check Bits (Input; Active LOW)
When in the LOW state, the three-state output lines SC_{0-7} are enabled. When this input is HIGH, the syndrome/check bit outputs are in the high-impedance state.

SC_{0-7} Syndrome/Check Bits (Outputs; Three-State)
These eight three-state outputs contain the check/partial-check bits when the EDC is in the Generate Mode. They also contain the syndrome/partial-syndrome bits when the device is in the Detect or Detect/Correct Modes.

V_{CC} (2) + 5-V Positive Power Supply Voltage
These pins are the positive + 5-V power supply to the EDC Circuit. All V_{CC} pins must be connected during device operation.

FUNCTIONAL DESCRIPTION

EDC Architecture

The Am29C660 EDC Circuit is a powerful 32-bit cascadable slice used for check bit generation, error detection, error correction, and diagnostics.

As shown in the block diagram, the device consists of the following:

- Data Input Latch
- Check Bit Input Latch
- Check Bit Generation Logic
- Syndrome Generation Logic
- Error Detection Logic
- Error Correction Logic
- Data Output Latch and Output Buffers
- Diagnostics Latch
- Control Logic

Data Input Latch

The Latch Enable Input, LE_{IN}, controls the loading of 32 bits of data into the Data Input Latch. Depending upon the control mode, the input data is either used for check bit generation or error detection/correction.

Check Bit Input Latch

Eight check bits are loaded under the control of LE_{IN}. Check bits are used in the Error Detection and Error Detection/Correction Modes.

Check Bit Generation Logic

This block generates the appropriate check bits for the 32 bits of data in the Data Input Latch. The check bits are generated according to a modified Hamming Code.

Syndrome Generation Logic

In both the Error Detection and Error Detection/Correction Modes, this logic block compares the check bits read from the memory against a newly generated set of check bits produced for the data read in from the memory. If both sets of check bits match, then there are no errors. If there is a mismatch, then one or more of the data and/or check bits is in error.

The syndrome bits are produced by an Exclusive-OR of the two sets of check bits. If the two sets of check bits are identical (meaning there are no errors), the syndrome bits will be all zeros. If there is a single-bit error, the syndrome bits can be decoded to determine the bit-in-error.

Error Detection Logic

This logic block decodes the syndrome bits generated by the Syndrome Generation Logic. If there are no errors in either the input data or check bits, the $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ outputs remain HIGH. If one or more errors are detected, $\overline{ERROR}$ goes LOW. If two or more errors are detected, both $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ go LOW.

Error Correction Logic

For single-bit errors, the Error Correction Logic corrects (by complementing) the single data bit in error. This corrected data is loaded into the Data Output Latch, which can then be read onto the bidirectional data lines. If the single-bit error is one of the check bits, the correction logic does not place corrected check bits on the syndrome/check bit outputs. If the corrected check bits are needed, the EDC must be switched to the Generate Mode.

Data Output Latch and Output Buffers

The Data Output Latch is used for storing the result of an error correction operation. The latch is loaded from the correction logic under control of the Data Output Latch Enable, LE_{OUT}/$\overline{GENERATE}$. The Data Output Latch may also be directly loaded from the Data Input Latch while in the Pass-Thru Mode.

Four data bytes in the Data Output Latch may be enabled independently (by $\overline{OE}$ $BYTE_{0-3}$) for reading onto the bidirectional data bus. This feature facilitates byte operations.

Diagnostics Latch

This is a 32-bit latch loadable from the bidirectional data lines under the control of the Diagnostic Latch Enable, LE_{DIAG}. The Diagnostic Latch contains check bit information in one byte and control information in the other bytes. The Diagnostic Latch is used for driving the device when in the Internal Control Mode, or for supplying check bits when in one of the Diagnostic Modes.

Control Logic

The Control Logic specifies the mode in which the EDC is operating. Normally, the control logic is driven by the external control inputs. However, in the Internal Control Mode, the control signals are taken from the Diagnostic Latch. Since LE_{OUT} and $\overline{GENERATE}$ are controlled by the same pin, the latching action (LE_{OUT} from HIGH to LOW) of the Data Output Latch causes the EDC to go into the Generate Mode.

Detailed Operational Description

The Am29C660 contains the logic necessary to generate check bits on a 32-bit data field according to a modified Hamming Code. Operating on data read from memory, the EDC will correct any single-bit error, and will detect all double- and some triple-bit errors. The Am29C660 may be configured to operate on 32-bit data words (with 7 check bits) and 64-bit data words (with 8 check bits). In either configuration, the device makes the error syndrome bits available on separate outputs for error logging.

Code and Byte Specification

The EDC Circuit may be configured in several different ways, and operates differently in each configuration. It is necessary to indicate to the device what size data word is involved and which bytes of the data word it is processing. This is done with the input lines CODE $ID_{0,\ 1}$ as shown in Table 1. The two modified Hamming codes referred to in Table 1 are:

1. 32/39 Code: 32 data bits, 7 check bits (39 bits in total)
2. 64/72 Code: 64 data bits, 8 check bits (72 bits in total)

TABLE 1. CODE ID DECODE

CODE ID_1	CODE ID_0	Hamming Code and Slice Selected
0	0	Code 32/39, 32-Bit Data Word
0	1	Internal Control Mode
1	0	Code 64/72, Lower 32-Bit Slice (0-31)
1	1	Code 64/72, Upper 32-Bit Slice (32-63)

Control Mode Selection

There are nine operating modes of the Am29C660. Eight of these modes are selected as shown in Tables 2 and 3. Table 2 is the Diagnostic Mode Control Decode Table, and Table 3 is the Mode Control Decode Table. The Diagnostic Mode pins, DIAG $MODE_{0,\ 1}$, define the five basic areas of operation. $\overline{GENERATE}$ and CORRECT further divide the operations into eight functions. The ninth mode is the Internal Control Mode which is selected by the CODE ID inputs as shown in Table 1.

TABLE 2. DIAGNOSTIC MODE CONTROL DECODE

DIAG $MODE_1$	DIAG $MODE_0$	CORRECT	Diagnostic Mode Selected
0	0	X	Normal EDC Function Mode or Non-Diagnostic Mode
0	1	X	Diagnostic Generate Mode
1	0	X	Diagnostic Detect Mode and Diagnostic Detect/Correct Mode
1	1	1	Initialize Mode
1	1	0	Pass-Thru Mode

TABLE 3. MODE CONTROL DECODE

Operating Mode	DIAG MODE 1	DIAG MODE 0	$LE_{OUT}/\overline{GENERATE}$	CORRECT	Contents of Data Output Latch	$SC_{0\text{-}7}$ and $\overline{OE}_{SC}$ = LOW	$\overline{ERROR}$ and $\overline{MULT\ ERROR}$
Generate	0 1	0 0	0	X	*	Check bits generated from Data Input Latch	High
Detect	0 0	0 1	1	0	Data Input Latch	Generated from Data Input/Check Bit Latches	Valid
Detect/Correct	0 0	0 1	1	1	Data Input Latch with single bit error detected	Generated From Data Input/Check Bit Latches	Valid
Pass-Thru	1	1	1 or ↓	0	Data Input latch	Check Bit Latch	HIGH
Diagnostic Generate	0	1	0	X	—	Check bits from Diagnostic Latch	—
Diagnostic Detect	1	0	1	0	Data Input latch	Data Input check bits from Diagnostic Latch	Valid
Diagnostic Detect/Correct	1	0	1	1	Data Input Latch with single-bit error detected	Data Input check bits from Diagnostic Latch	Valid
Initialize	1	1	1 or ↓	1	Data Input Latch set to all "0"s	Check bits generated from Data Input Latch	—
Internal Control	CODE $ID_{0,\ 1} = 10$ Control Signals CODE $ID_{0,\ 1}$ DIAG $MODE_{0,\ 1}$ and CORRECT are taken from the Diagnostic Latch						

*In the Generate Mode, data is read into the EDC circuit and the check bits are generated. The same data is written to the memory along with the check bits. Since the Data Output Latch is not used in the Generate Mode, LE_{OUT} being LOW (since it is tied to Generate) does not affect the writing of check bits.

The Generate Mode is used to display the check bits on the SC_{0-7} outputs. The Error Detect Mode provides an indication of an error or multiple errors on the $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ outputs. Single-bit errors are not corrected in this mode. The syndrome bits are provided on the SC_{0-7} outputs. In the Diagnostic Detect Mode, the syndrome bits are generated by comparing the internally generated check bits from the Data Input Latch with check bits stored in the Diagnostic Latch (as opposed to those in the Check Bit Latch).

The Detect/Correct Mode is similar to the Detect Mode except that single-bit errors will be corrected (by complementing) and made available as input to the Data Output Latch. In the Diagnostic Detect/Correct Mode, single-bit errors will be corrected as determined by the syndrome bits, which are in turn generated from the check bits corresponding to the data in the Data Input Latch and the check bits in the Diagnostic Latch.

In the Initialize Mode check bits are generated for all zero data bits. The Data Input Latch is held to a logic zero and is made available as input to the Data Output Latch.

In the Internal Control Mode, the control signals CODE $ID_{0,\ 1}$, DIAG $MODE_{0,\ 1}$, and CORRECT are taken from the Diagnostic Latch and their respective control inputs are disregarded.

Check and Syndrome Bits

The Am29C660 provides either check or syndrome bits on the three-state outputs SC_{0-7}. Check bits are generated from the Data Input Bits. Syndrome bits are an Exclusive-OR of the check bits generated from the data read from the memory and the check bits read from the memory with the stored data.

Syndrome bits can be decoded to determine the single bit-in-error or to indicate a double- or triple-bit error.

The check bits generated by the Am29C660 are designated as follows:

- 32-bit configuration: CX, C0, C1, C2, C4, C8, C16
- 64-bit configuration: CX, C0, C1, C2, C4, C8, C16, C32
- Pin Name: SC_0, SC_1, SC_2, SC_3, SC_4, SC_5, SC_6, SC_7.

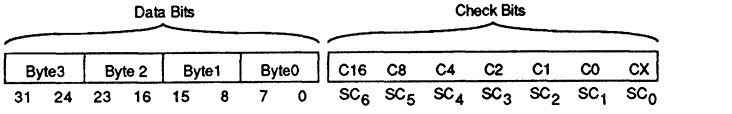
The syndrome bits generated by the Am29C660 are designated as follows:

- 32-bit configuration: SX, S0, S1, S2, S4, S8, S16
- 64-bit configuration: SX, S0, S1, S2, S4, S8, S16, S32
- Pin Name: SC_0, SC_1, SC_2, SC_3, SC_4, SC_5, SC_6, SC_7.

32-Bit Word Configuration

Data Field Format

The 32-bit format consists of 32 data bits and 7 check bits and is referred to as the 32/39 code. The format is shown in Figure 1.

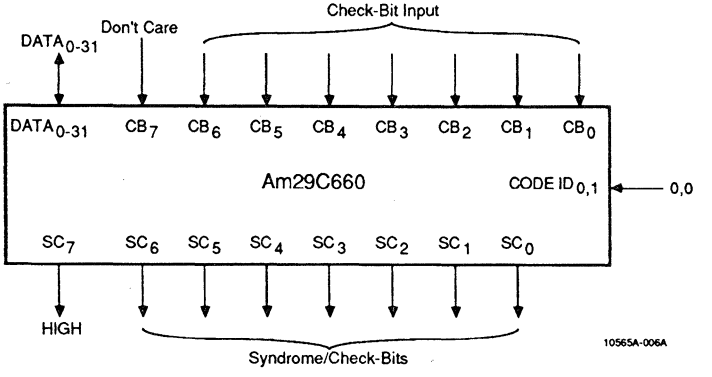


TB001210

Figure 1. 32-Bit Data Word Format

Chip Configuration

A single Am29C660 EDC Circuit connected as shown in Figure 2 is all that is necessary to perform single-bit error correction and double-bit error detection on a 32-bit data field. In this configuration, only seven check bits are required. Therefore, CB_7 is a "don't care" and CODE $ID_{0,1}$ = 00.



LD001600

Figure 2. 32-Bit Configuration

Generate Mode

In this mode, check bits will be generated that correspond to the contents of the Data Input Latch. The check bits generated are placed on the outputs SC_{0-6}. SC_7 is a logic "1" or HIGH.

Check bits are generated according to a modified Hamming Code. Details of the code for check bit generation are shown in Tables 4-1 and 4-2. Check bits are generated as either an XOR or XNOR of 16 of the 32 bits as indicated in the table. The XOR function results in an even parity check bit, the XNOR in an odd parity check bit.

Figure 3 shows the data flow in the Generate Mode.

TABLE 4-1. 32-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 0 – 15)

Generated Check Bits	Parity	Participating Data Bits															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)	X				X		X	X	X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								

TABLE 4-2. 32-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 16 – 31)

Generated Check Bits	Parity	Participating Data Bits															
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
CX	Even (XOR)		X	X	X		X					X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X

The check bit is generated as either an XOR or XNOR of the sixteen data bits noted by an "X" in the table.

Detect Mode

In this mode the device will compare the check bits generated from the contents of the Data Input Latch with the contents of the Check Bit Latch, and will detect all single- and double-bit errors and some triple-bit errors. If one or more errors are detected, $\overline{\text{ERROR}}$ goes LOW. If two or more errors are detected, $\overline{\text{MULT ERROR}}$ and $\overline{\text{ERROR}}$ go LOW. Both error indicators are HIGH if there is no error.

The syndrome bits which are generated during error detection are available on the outputs SC_{0-6}. SC_7 remains HIGH. The syndrome bits may be decoded to determine if a bit error was detected and for a single-bit error, which of the data or check bits is in error. Table 5 shows the syndrome bit decoding for the 32-bit data word configuration. If no error is detected, the syndrome bits will all be zeros (except SC_7 which is tied to a logical "1").

TABLE 5. SYNDROME BIT DECODING MATRIX

Syndrome Bits				S16	0	1	0	1	0	1	0	1
				S8	0	0	1	1	0	0	1	1
				S4	0	0	0	0	1	1	1	1
SX	S0	S1	S2									
0	0	0	0		*	C16	C8	T	C4	T	T	30
0	0	0	1		C2	T	T	27	T	5	M	T
0	0	1	0		C1	T	T	25	T	3	15	T
0	0	1	1		T	M	13	T	23	T	T	M
0	1	0	0		C0	T	T	24	T	2	M	T
0	1	0	1		T	1	12	T	22	T	T	M
0	1	1	0		T	M	10	T	20	T	T	M
0	1	1	1		16	T	T	M	T	M	M	T
1	0	0	0		CX	T	T	M	T	M	14	T
1	0	0	1		T	M	11	T	21	T	T	M
1	0	1	0		T	M	9	T	19	T	T	31
1	0	1	1		M	T	T	29	T	7	M	T
1	1	0	0		T	M	8	T	18	T	T	M
1	1	0	1		17	T	T	28	T	6	M	T
1	1	1	0		M	T	T	26	T	4	M	T
1	1	1	1		T	0	M	T	M	T	T	M

* = No errors detected
Number = The bit number of the single bit-in-error
T = Two errors detected
M = More than two errors detected

Detect/Correct Mode

In this mode, the EDC functions the same way as in the Detect Mode except that the correction network is enabled to correct, by complementing, any single-bit error of the Data Input Latch before placing it on the inputs of the Data Output Latch (see Figure 4). If a multiple error is detected, the output of the correction network is undefined. If a single-bit error occurs to a check bit there is no automatic correction. If a check bit correction is desired, it can be done by placing the device in the Generate Mode to produce the correct check bit sequence for the data in the Data Input Latch.

Pass-Thru Mode

In this mode, the unmodified contents of the Data Input Latch are placed on the inputs of the Data Output Latch, and the contents of the Check Bit Latch are placed on the outputs SC_{0-6} (SC_7 is a logical "1"). $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ are forced HIGH in this mode (inactive).

Diagnostic Generate Mode

This is one of the three special Diagnostic Modes selected by the control inputs DIAG $MODE_{0,\ 1}$. This mode is similar to the normal Generate Mode except that the check bits are not generated from the contents of the Data Input Latch. They are instead taken directly from the contents of the Diagnostic Latch. Table 6 shows the Diagnostic Latch coding format. Figures 5 and 6 illustrate the flow of data during the two diagnostic modes.

TABLE 6. 32-BIT DIAGNOSTIC LATCH CODING FORMAT

Data Bit	Internal Function
0	Diagnostic Check Bit X
1	Diagnostic Check Bit 0
2	Diagnostic Check Bit 1
3	Diagnostic Check Bit 2
4	Diagnostic Check Bit 4
5	Diagnostic Check Bit 8
6	Diagnostic Check Bit 16
7	Don't Care
8	CODE ID_0
9	CODE ID_1
10	DIAG $MODE_0$
11	DIAG $MODE_1$
12	CORRECT
13-31	Don't Care

Diagnostic Detect, Diagnostic Detect/Correct Modes

These two special Diagnostic Modes are also selected by the control inputs DIAG $MODE_{0,\ 1}$. They are similar to the normal Detect and Detect/Correct Modes except that the check bits are taken from the Diagnostic Latch rather than from the Check Bit Input Latch.

Initialize Mode

In this mode, the inputs of the Data Output Latch are forced to all zeros. The syndrome bit outputs SC_{0-6} are generated to correspond to the all-zero data on the Data Output Latch. SC_7 is tied HIGH. The $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ outputs are forced HIGH in this mode. The Initialize Mode is useful after power-up when RAM contents are random. The EDC may be

placed in this mode and its outputs written into all memory addresses under processor control.

Internal Control Mode

This mode is selected by the external control inputs CODE $ID_{0,1}$. When in the Internal Control Mode, the Am29C660 takes the CODE $ID_{0, 1}$, DIAG $MODE_{0, 1}$, and CORRECT control signals from the Diagnostic Latch rather than from the external input lines.

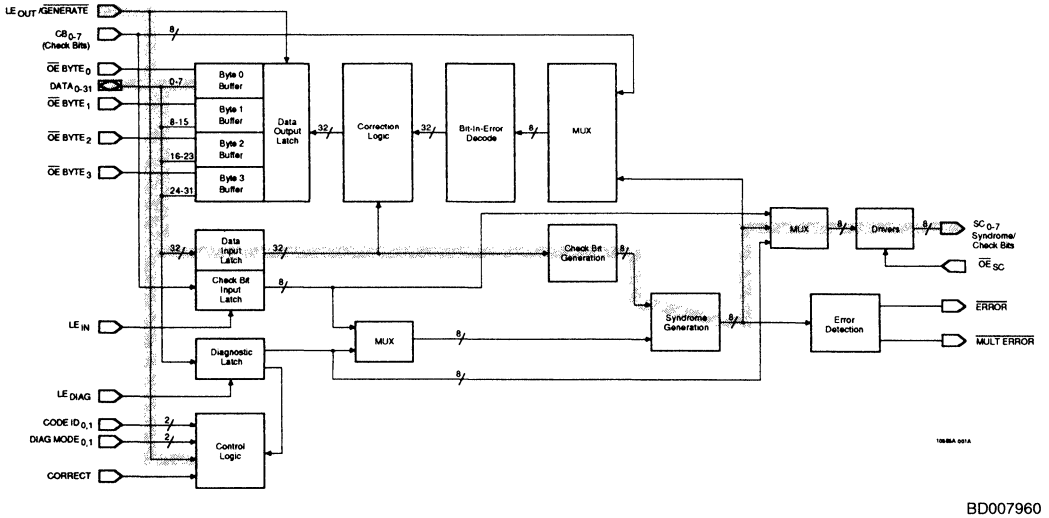


Figure 3. Check Bit Generation Data Path

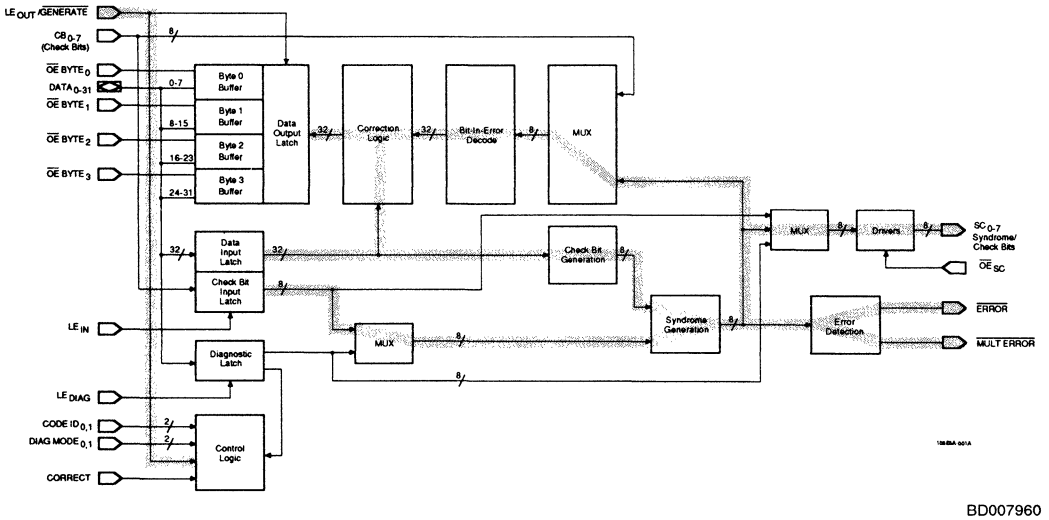


Figure 4. Error Detection and Correction Data Path

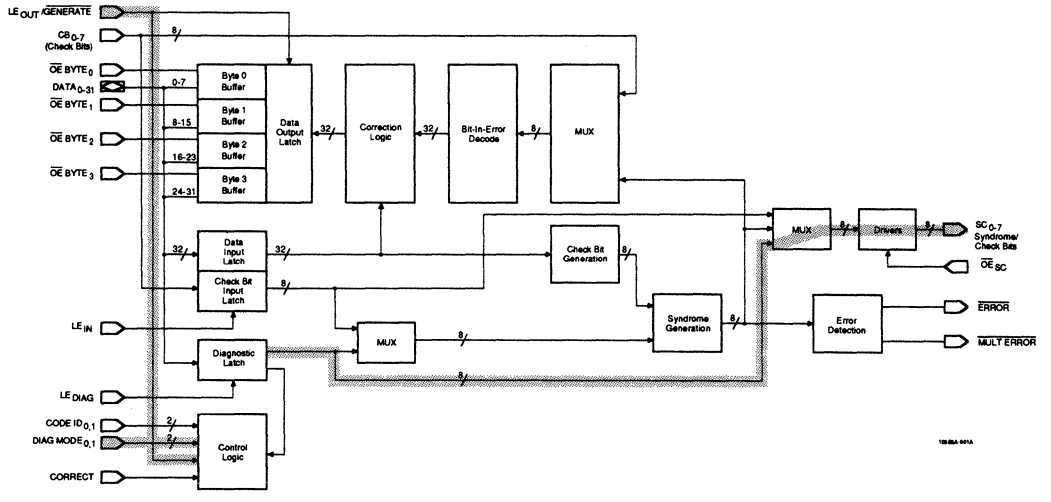


Figure 5. Diagnostic Check Bit Generation Data Path

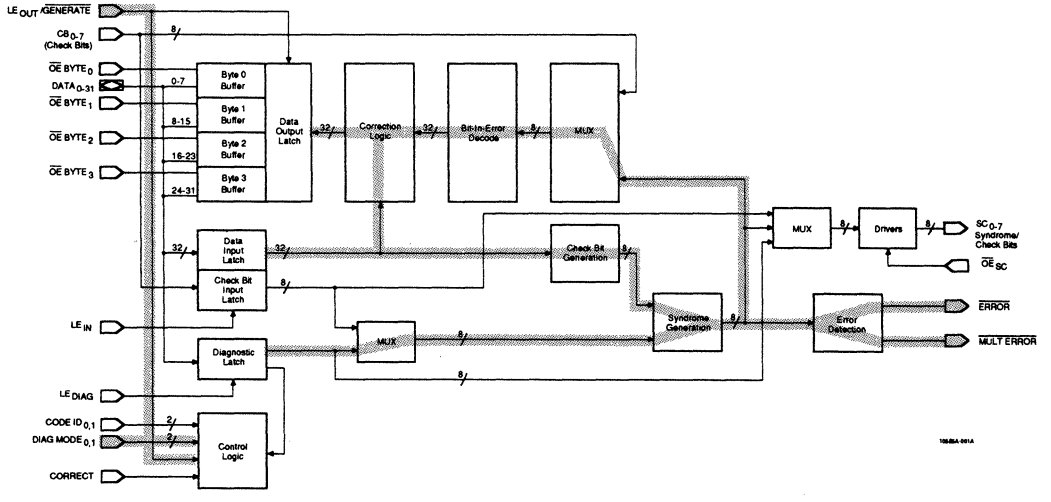
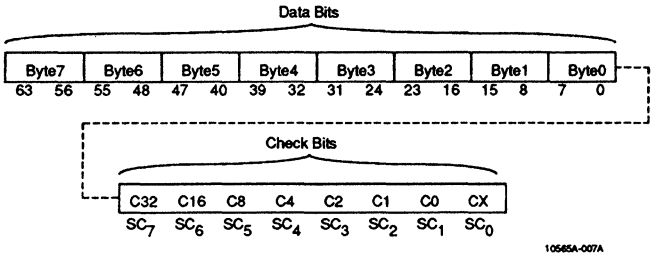


Figure 6. Diagnostic Detect and Correct Data Path

64-Bit Data Word Configuration

Data Field Format

The 64-bit format consists of 64 data bits and 8 check bits and is referred to as the 64/72 code. The format is shown in Figure 7.

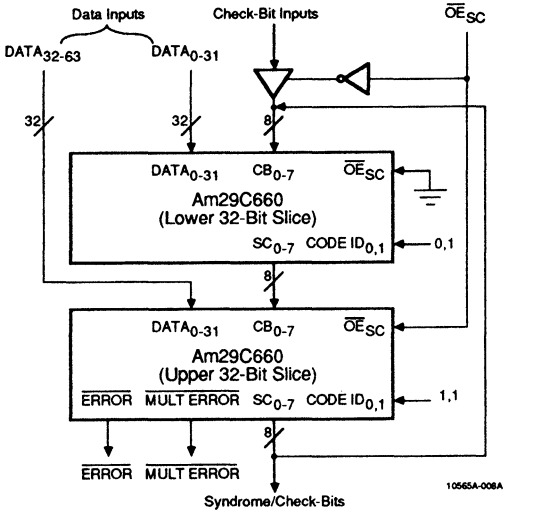


TB001220

Figure 7. 64-Bit Data Word Format

Chip Configuration

Two Am29C660 EDC Circuits connected as shown in Figure 8, provide all the necessary logic for all single-bit error detection/correction, all double-bit error detection, and some triple-bit error detection on a 64-bit data field. In this configuration, eight check bits are required. CODE $ID_{0,\ 1}$ control signal inputs distinguish the upper 32 bits from the lower 32 bits as shown in Table 1.



LD001610

Figure 8. 64-Bit Configuration

The valid syndrome bits and the $\overline{ERROR}$ and $\overline{MULT\ ERROR}$ output signals are taken from the upper EDC slice. The lower EDC slice has its $\overline{OE}_{SC}$ input tied to ground (logic "0") and its SC_{0-7} outputs connected to the respective CB_{0-7} inputs of the upper EDC slice. The 32 most significant data bits, $DATA_{32-63}$, are connected to the data lines of the upper EDC slice. All the latch enables and control signals must be input to both the upper and the lower EDC slices.

In the Detect/Correct Mode, the valid syndrome bits from the upper EDC slice must be read into the lower EDC slice via the check bit inputs of the lower EDC slice. They are selected by an internal MUX as inputs to the bit-in-error decoder (see Block Diagram). External buffering and output enabling of the check bit lines is required as shown in Figure 8. The $\overline{OE}_{SC}$ signal to the upper EDC slice can be used to control the enabling of the check bits to the lower EDC slice. The external check bits to the lower EDC slice are disabled.

Generate Mode

In this mode, check bits will be generated that correspond to the contents of the Data Input Latch. The generated check bits are placed on the SC_{0-7} outputs of the upper EDC slice.

Check bits are generated according to a modified Hamming Code. Details of the code for check bit generation are shown in Tables 7-1 through 7-4. Check bits are generated as either an XOR or XNOR of 32 of the 64 bits. The XOR function results in an even parity check bit, the XNOR in an odd parity check bit.

TABLE 7-1. 64-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 0 – 15)

Generated Check Bits	Parity	Participating Data Bits															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								
C32	Even (XOR)	X	X	X	X	X	X	X	X								

TABLE 7-2. 64-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 16-31)

Generated Check Bits	Parity	Participating Data Bits															
		16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
CX	Even (XOR)		X	X	X		X			X	X		X			X	
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X
C32	Even (XOR)									X	X	X	X	X	X	X	X

TABLE 7-3. 64-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 32 – 47)

Generated Check Bits	Parity	Participating Data Bits															
		32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
CX	Even (XOR)	X				X		X	X			X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)	X	X	X	X	X	X	X	X								
C32	Even (XOR)									X	X	X	X	X	X	X	X

TABLE 7-4. 64-BIT CONFIGURATION CHECK BIT ENCODING (Data Bits 48-63)

Generated Check Bits	Parity	Participating Data Bits															
		48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
CX	Even (XOR)	X				X		X	X			X		X	X		X
C0	Even (XOR)	X	X	X		X		X		X		X		X			
C1	Odd (XNOR)	X			X	X			X		X	X			X		X
C2	Odd (XNOR)	X	X				X	X	X				X	X	X		
C4	Even (XOR)			X	X	X	X	X	X							X	X
C8	Even (XOR)									X	X	X	X	X	X	X	X
C16	Even (XOR)									X	X	X	X	X	X	X	X
C32	Even (XOR)	X	X	X	X	X	X	X	X								

The check bit is generated as either an XOR or XNOR of the 32 data bits noted by an "X" in the table.

Detect Mode

In this mode, the device will compare the check bits generated from the contents of the Data Input Latch with the contents of the Check Bit Latch. All single- and double-bit errors and some triple-bit errors will be detected. If one or more errors are detected, $\overline{\text{ERROR}}$ goes LOW. If two or more errors are detected, both $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ will go LOW. Both error indicators will be HIGH if no errors are detected. The valid $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ signals are from the upper EDC slice.

The syndrome bits which are generated during error detection are available on the outputs SC_{0-7} of the upper EDC slice. The syndrome bits may be decoded to determine if a bit error was detected, and for a single-bit error, which of the data or check bits is in error. Table 8 shows the syndrome bit decoding for the 64-bit data word configuration. If no error is detected, the syndrome bits will all be zeros.

In the Detect Mode, the contents of the Data Input Latch are driven directly into the inputs of the Data Output Latch without correction.

TABLE 8. 64-BIT SYNDROME BIT DECODING MATRIX

Syndrome Bits																		
				S32	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
				S16	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
				S8	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
				S4	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
SX	S0	S1	S2																	
0	0	0	0		*	C32	C16	T	C8	T	T	M	C4	T	T	M	T	46	62	T
0	0	0	1		C2	T	T	M	T	43	59	T	T	53	37	T	M	T	T	M
0	0	1	0		C1	T	T	M	T	41	57	T	T	51	35	T	15	T	T	31
0	0	1	1		T	M	M	T	13	T	T	29	23	T	T	7	T	M	M	T
0	1	0	0		C0	T	T	M	T	40	56	T	T	50	34	T	M	T	T	M
0	1	0	1		T	49	33	T	12	T	T	28	22	T	T	6	T	M	M	T
0	1	1	0		T	M	M	T	10	T	T	26	20	T	T	4	T	M	M	T
0	1	1	1		16	T	T	0	T	M	M	T	T	M	M	T	M	T	T	M
1	0	0	0		CX	T	T	M	T	M	M	T	T	M	M	T	14	T	T	30
1	0	0	1		T	M	M	T	11	T	T	27	21	T	T	5	T	M	M	T
1	0	1	0		T	M	M	T	9	T	T	25	19	T	T	3	T	47	63	T
1	0	1	1		M	T	T	M	T	45	61	T	T	55	39	T	M	T	T	M
1	1	0	0		T	M	M	T	8	T	T	24	18	T	T	2	T	M	M	T
1	1	0	1		17	T	T	1	T	44	60	T	T	54	38	T	M	T	T	M
1	1	1	0		M	T	T	M	T	42	58	T	T	52	36	T	M	T	T	M
1	1	1	1		T	48	32	T	M	T	T	M	M	T	T	M	T	M	M	T

* = No errors detected
Number = The bit number of the single bit-in-error
T = Two errors detected
M = More than two errors detected

Detect/Correct Mode

In this mode, the EDC functions the same way as in the Detect Mode except that the correction network is enabled to correct, by complementing, any single-bit error in the Data Input Latch before placing the data on the inputs of the Data Output Latch. If a multiple error is detected, the output of the correction network is undefined. If a single-bit error occurs to a check bit, there is no automatic correction. If a check bit correction is desired, it can be done by placing the device in the Generate Mode to produce the correct check bit sequence for the data in the Data Input Latch.

For data correction, both the upper EDC slice and the lower EDC slice require access to the syndrome bits SC_{0-7} of the upper EDC slice. The upper EDC slice has access to these syndrome bits through an internal data path. The syndrome bits must be read through the CB_{0-7} inputs for the lower EDC slice as shown in Figure 8. In the Detect/Correct Mode, the valid SC_{0-7} outputs of the upper EDC slice must be enabled so that they are available for reading in through the CB_{0-7} inputs of the lower EDC slice.

Pass-Thru Mode

In this mode, the unmodified contents of the Data Input Latch are placed on the inputs of the Data Output Latch and the contents of the Check Bit Latch are placed on the outputs SC_{0-7} of the upper EDC slice. $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ are forced HIGH in this mode (inactive).

Diagnostic Generate Mode

This is one of the three special Diagnostic Modes selected by the control inputs DIAG $MODE_{0,\ 1}$. This mode is similar to the normal Generate Mode except that the check bits are not generated from the contents of the Data Input Latch. They are instead taken directly from the contents of the Diagnostic Latch. Table 9 shows the Diagnostic Latch coding format.

TABLE 9. 64-BIT DIAGNOSTIC LATCH CODING FORMAT

Data Bit	Internal Function
0	Diagnostic Check Bit X
1	Diagnostic Check Bit 0
2	Diagnostic Check Bit 1
3	Diagnostic Check Bit 2
4	Diagnostic Check Bit 4
5	Diagnostic Check Bit 8
6	Diagnostic Check Bit 16
7	Diagnostic Check Bit 32
8	CODE ID_0, Lower 32-Bit Slice
9	CODE ID_1, Lower 32-Bit Slice
10	DIAG $MODE_0$, Lower 32-Bit Slice
11	DIAG $MODE_1$, Lower 32-Bit Slice
12	CORRECT, Lower 32-Bit Slice
13-31	Don't Care
32-39	Don't Care
40	CODE ID_0, Upper 32-Bit Slice
41	CODE ID_1, Upper 32-Bit Slice
42	DIAG $MODE_0$, Upper 32-Bit Slice
43	DIAG $MODE_1$, Upper 32-Bit Slice
44	CORRECT, Upper 32-Bit Slice
45-63	Don't Care

Diagnostic Detect, Diagnostic Detect/Correct Modes

These two special Diagnostic Modes are also selected by the control inputs DIAG $MODE_{0,\ 1}$. These modes are similar to the normal Detect and Detect/Correct Modes except that the check bits are taken from the Diagnostic Latch, rather than from the Check Bit Input Latch.

Initialize Mode

In this mode the inputs of the Data Output Latch are forced to all zeros. The syndrome outputs SC_{0-7} are generated to correspond to the all-zero data on the Data Output Latch. The $\overline{\text{ERROR}}$ and $\overline{\text{MULT ERROR}}$ outputs are forced HIGH in this mode. The Initialize Mode is useful after power-up when RAM contents are random. The EDC may be placed in this mode in order to write its outputs into all memory addresses under processor control.

Internal Control Mode

This mode is selected by the external control inputs CODE $ID_{0,\ 1}$. When in the Internal Control Mode, the Am29C660 takes the CODE $ID_{0,\ 1}$, DIAG $MODE_{0,\ 1}$, and CORRECT control signals from the Diagnostic Latch rather than from the external input lines.

TABLE 10. KEY AC CALCULATIONS FOR THE 64-BIT CONFIGURATION

64-Bit Propagation Delay		Component Delays from Am29C660 AC Specifications	Example for Am29C660D
From	To		
$DATA_{0-31}$	Check Bit Outputs	(Data to SC, CODE ID 10) + (Check Bit to SC, CODE ID 11)	14 + 11 = 25 ns
$DATA_{0-31}$	Corrected Data Outputs	(Data to SC, + CODE ID 10) + (Check Bit to SC, CODE ID 11) + (Check Bit to Data, CODE ID 10)	14 + 11 + 12 = 37 ns
$DATA_{0-31}$	Syndrome Outputs	(Data to SC, CODE ID 10) + (Check Bit to SC, CODE ID 11)	14 + 11 = 25 ns
$DATA_{0-31}$	$\overline{\text{ERROR}}$ for 64 Bits	(Data to SC, CODE ID 10) + (Check Bit to $\overline{\text{ERROR}}$, CODE ID 11)	14 + 10 = 24 ns
$DATA_{0-31}$	$\overline{\text{MULT ERROR}}$ for 64 Bits	(Data to SC, CODE ID 10) + (Check Bit to $\overline{\text{MULT ERROR}}$, CODE ID 11)	14 + 12 = 26 ns

APPLICATIONS

System Design Considerations

To obtain optimum performance and maximum design flexibility, AMD's Dynamic Memory Management System has been divided into functional building blocks. For 32-bit error detecting/correcting systems, these building blocks include the Am29C660 EDC Circuit, Am29C668 4M Configurable Dynamic Memory Controller/Driver, and the Am2971A 100-MHz Enhanced Programmable Event Generator or delay lines as the timing reference. Together these chips can perform traditional EDC, Flow-Thru or Fly-By, or AMD's Scrubbing EDC cycle.

High-Performance Parallel Operation

For maximum memory system performance, the EDC should be used in the Check-Only (or Fly-By) configuration shown in Figure 9. With this configuration, the memory system operates as fast with EDC as it would without.

On reads from memory, data is read out from the DRAMs directly to the data bus (same as in a non-EDC system). At the same time, the data is read into the EDC to check for errors. If an error exists, the EDC's error flags ($\overline{\text{ERROR}}$, $\overline{\text{MULT ERROR}}$) are used to interrupt the CPU and/or to stretch the memory cycle. If no error is detected, no slowdown is required.

If an error is detected, the EDC generates corrected data for the processor. At the designer's option, the correct data may be written back into memory; error logging and diagnostic routines may also be run under processor control.

The Check-Only configuration allows data reads to proceed as fast with EDC as without. Only if an error is detected is there any slowdown, but even if the memory system had an error every hour this would mean only one error every 3 – 4 billion memory cycles. Therefore, even with a very high error rate, EDC in a Check-Only configuration has essentially zero impact on memory system speed.

On writes to memory, check bits must be generated before the full memory word can be written into memory. Using the Am29C983 Multiple Bus Exchange allows the data word to be buffered on the memory board while check bits are generated. This makes the check bit generate time transparent to the processor.

EDC in the Data Path

The simplest configuration for the Am29C660 is to have the EDC circuit directly in the data path, as shown in Figure 10 (Flow-Thru configuration). In this configuration, data read from memory is always corrected prior to putting the data on the data bus. The advantages are simpler operation and no need for mid-cycle interrupts. The disadvantage is that memory system speed is slowed by the amount of time it takes for error correction on every cycle.

Usually the Flow-Thru configuration will be used with MOS microprocessors which have ample memory timing budgets. Most high-performance processors will use the high-performance parallel configuration shown in Figure 9 (Fly-By).

Memory Scrubbing During Refresh

Scrubbing is an error correction technique that examines the entire memory during system refresh cycles, thus causing little or no impact on performance yet providing high data reliability since errors cannot accumulate.

Single-bit errors are by far the most common in a dynamic memory system, and are always correctable by the EDC. Double-bit errors occur far less frequently than single-bit errors (100-to-1 or greater) and are always detected by the EDC, but not corrected.

In a memory system, soft errors occur only one at a time. A double-bit error in a data word occurs when a single soft error is left uncorrected and is followed by another error in the data word within hours, days, or weeks after the first occurence.

Scrubbing the memory periodically avoids almost all double-bit errors. In the scrubbing operation, every data word in memory is periodically checked by the EDC for single-bit errors. If one is found, it is corrected and the correct data word is written back into memory. Errors are not allowed to accumulate, and thus most double-bit errors are avoided.

The scrubbing operation is generally done as a background routine when the memory is not being used by the processor. If memory is scrubbed frequently, errors that are detected and corrected during processor accesses need not be immediately written back into memory. Instead, the error will be corrected in memory during scrubbing. This reduces the time delay involved in a processor access of an incorrect memory word.

On each refresh-with-scrub cycle, one memory location is read, checked for errors, and if necessary, corrected before being written back into memory. For a sixteen-megaword memory (2^{24} locations) with one refresh every 16 microseconds, the AMD Memory Management chip set will scrub the entire memory of single-bit errors every minute. If multiple-bit

errors are encountered during a scrub cycle, $\overline{WE}$ is suppressed.

With the occurance of an error, a read/modify/write (R/M/W) cycle is performed. The duration of a R/M/W cycle is typically longer than a normal read or write cycle. During refresh operations, a row in each bank is accessed by energizing the $\overline{RAS}$ line. This refreshes all the locations in that row. If an error is detected, a write operation is performed in that bank at the location of the error. This is accomplished by energizing the $\overline{CAS}$ line in that bank for that location. The entire checking operation is performed within the refresh cycle. A wait state may need to be issued to extend the cycle should an error be discovered. However, the system reliability will be increased because soft errors will not be able to accumulate in areas of memory that are not frequently accessed.

When performing refresh without scrubbing, all four $\overline{RAS}$ lines go LOW but the $\overline{CAS}$ lines remain HIGH or inactive. A refresh with scrubbing will activate all four $\overline{RAS}$ lines as before and a single $\overline{CAS}$ line. Errors that are detected during scrubbing cycles do not cause interrupts or bus-error ($\overline{BERR}$) signal assertions.

Figure 11 shows a sixteen-megaword memory system with error detection and correction.

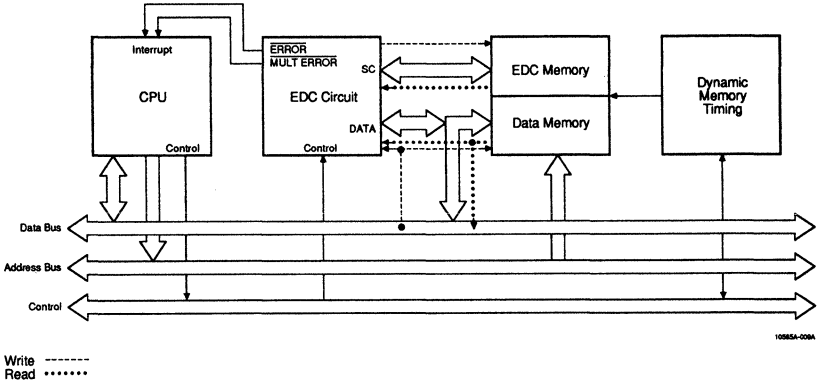


Figure 9. Check-Only Configuration

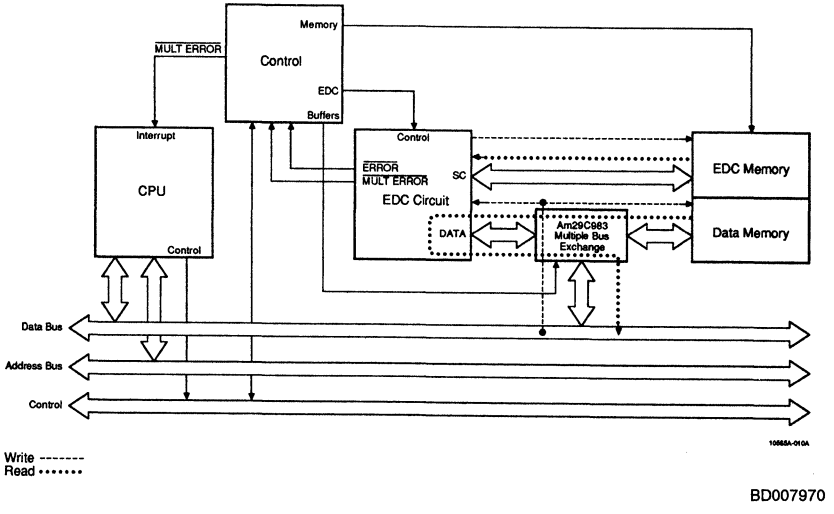
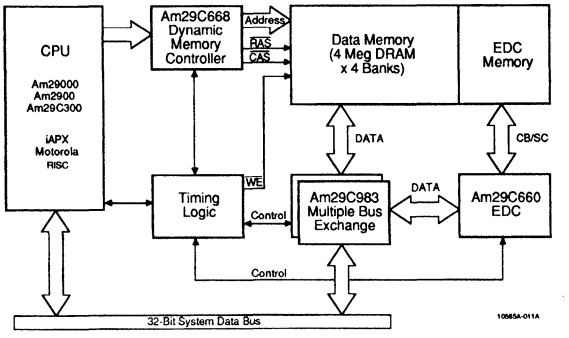
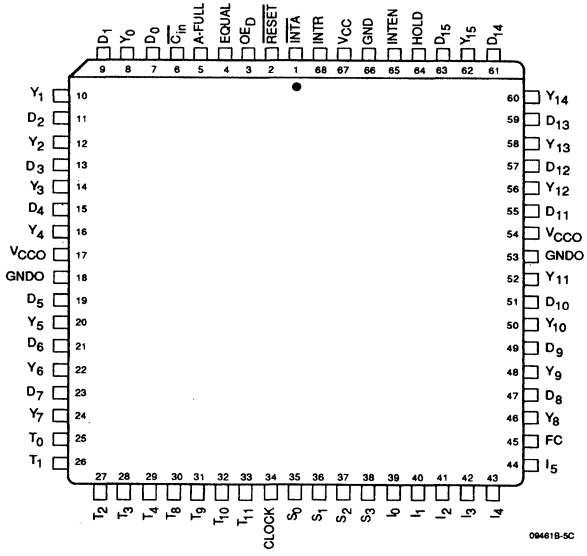


Figure 10. Correct-Always Configuration



BD007991

Figure 11. Sixteen-Megaword Memory System with EDC



CD011510

Figure 12. Device Decoupling — V_{CC} and Ground Pin Connections

Note: $C_1 = 1.0\ \mu F$, $C_2 = 0.1\ \mu F$, $C_3 = 0.01\ \mu F$
The C_1, C_2, C_3 capacitors should be used to shunt low- and high-frequency noise from V_{CC}. Do not replace with one capacitor. Place capacitors as close to the device as possible.

Table 11. 32-Bit Word/Check Bit Examples

Example 32-Bit Word								Corresponding Check Bits	
D_{31}							D_0	C_X	C_{16}
0101	0101	0101	0101	0101	0101	0101	0101	0011000	
1010	1010	1010	1010	1010	1010	1010	1010	0011000	
0001	0000	1100	0111	0111	1101	0111	1111	1101110	
0000	0011	0011	1101	1000	0101	0100	0000	1110011	
1111	1111	1111	0000	0000	0000	1111	1110	0101001	

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65 to +150°C
Temperature (Case)
Under Bias ..... −55 to +125°C
Supply Voltage to Ground Potential
Continous ..... −0.5 to +7.0 V
DC Voltage Applied to Outputs For
HIGH Output State ..... −0.5 V to V_{CC} Max.
DC Input Voltage ..... −0.5 to +5.5 V
DC Input Current ..... −30 mA to +5.0 mA

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Ambient Temperature (T_A) ..... 0 to +70°C
Supply Voltage ..... +4.75 to +5.25 V

Military (M) Devices
Case Temperature (T_C) ..... −55 to +125°C
Supply Voltage ..... +4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating range unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions (Note 1)			Min.	Max.	Unit
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for all Inputs (Note 3)			2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for all Inputs (Note 3)				0.8	V
I_{IH}	Input HIGH Current	V_{CC} = Max., V_{IN} = V_{CC}				5.0	µA
I_{IL}	Input LOW Current	V_{CC} = Max., V_{IN} = GND				−5.0	µA
V_{OH}	Output HIGH Voltage	V_{CC} = Min.	I_{OH} = −300 µA		V_{CC}−0.2		V
			MIL I_{OH} = −8 mA		2.4		
			COM'L I_{OH} = −15 mA		2.4		
V_{OL}	Output LOW Voltage	V_{CC} = Min.	I_{OL} = 300 µA			0.2	V
			MIL I_{OL} = 8 mA			0.5	
			COM'L I_{OL} = 15 mA			0.5	
I_{OZ}	Off-State (High-Impedance) Output Current	V_{CC} = Max.		V_O = 0 V		−10	µA
				V_O = V_{CC} (Max.)		10	
I_{OS}	Output Short-Circuit Current	V_{CC} = Min., V_O = 0 V (Note 2)			−30		mA
I_{CCQ}	Quiescent Power Supply Current (CMOS Inputs)	V_{CC} = Max., V_{CC}−0.2 V ≤ V_{IN}, V_{IN} ≤ 0.2V f_{OP} = 0				5.0	mA
I_{CCT}	Quiescent Input Power Supply Current (per Input @ TTL HIGH) (Note 4)	V_{CC} = Max., V_{IN} = 3.4 V, f_{OP} = 0				0.5	mA/ Input
I_{CCD}	Dynamic Power Supply Current	V_{CC} = Max., V_{CC}−0.2 V ≤ V_{IN}, V_{IN} ≤ 0.2 V Outputs Open, $\overline{OE}$ = LOW		MIL		6.5	mA/ MHz
				COM'L		6	
I_{CC}	Total Power Supply Current (Note 5)	V_{CC} = Max., f_{OP} = 10 MHz Outputs open, $\overline{OE}$ = LOW 50% Duty cycle V_{CC}−0.2 V ≤ V_{IN}, V_{IN} ≤ 0.2 V		MIL		70	mA
				COM'L		65	
		V_{CC} = Max., f_{OP} = 10 MHz Outputs open, $\overline{OE}$ = LOW 50% Duty cycle V_{IH} = 3.4 V, V_{IL} = 0.4 V		MIL		70	
				COM'L		65	

Notes:
1. For conditions shown as Min. or Max., use appropriate value specified under Operating Range for the applicable device type.
2. Not more than one output should be shorted at a time. Duration of the short-circuit test should not exceed one second.
3. These input levels provide zero noise immunity and should only be static tested in a noise-free environment.
4. I_{CCT} is derived by measuring the total current with all the inputs tied together at 3.4 V, subtracting out I_{CCQ}, then dividing by the total number of inputs.
5. Total Power Supply Current is the sum of the Quiescent Current and the Dynamic Current (at either CMOS or TTL input levels). For all conditions, the Total Power Supply Current can calculated by using the following equation:

I_{CC} = I_{CCQ} + I_{CCT} (N_T x D_H) + I_{CCD} (f_{OP}).
D_H = Data duty cycle TTL HIGH period (V_{IN} = 3.4 V).
N_T = Number of dynamic inputs driven at TTL levels.
f_{OP} = Operating frequency in Megahertz.

SWITCHING CHARACTERISTICS over COMMERCIAL operating ranges (Notes 1 and 2)

The following table specifies the guaranteed device performance over the Commercial operating range of 0°C to +70°C (ambient), with V_{CC} 4.75 to 5.25 V. All input switching is between 0 V and 3 V at 1 V/ns, and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

No.	Parameter Symbol	Data Path Description		Am29C660		Am29C660A		Am29C660B		Am29C660C	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
1	t_{PD}	$DATA_{0-31}$ (Note 3)	SC_{0-7}		37		27		25		18
			$DATA_{0-31}$ (Note 2)		49		36		30		24
			$\overline{ERROR}$		40		30		25		16
			$\overline{MULT\ ERROR}$		45		33		27		20
2	t_{PD}	CB_{0-7} (CODE ID 00, 11)	SC_{0-7}		22		16		14		14
			$DATA_{0-31}$		46		34		30		21
			$\overline{ERROR}$		26		19		17		13
			$\overline{MULT\ ERROR}$		31		23		20		16
3	t_{PD}	CB_{0-7} (CODE ID 10)	SC_{0-7}		22		16		16		15
			$DATA_{0-31}$		30		20		18		16
4	t_{PD}	$\overline{GENERATE}$	SC_{0-7}		29		21		21		18
			$\overline{ERROR}$		30		25		23		9
			$\overline{MULT\ ERROR}$		30		25		23		11
5	t_{PD}	CORRECT (Not Internal Control Mode)	$DATA_{0-31}$		31		23		23		16
6	t_{PD}	$DIAG\ MODE_{0,1}$ (Not Internal Control Mode)	SC_{0-7}		23		17		17		16
			$DATA_{0-31}$		35		26		26		26
			$\overline{ERROR}$		27		20		20		11
			$\overline{MULT\ ERROR}$		33		24		24		20
7	t_{PD}	$CODE\ ID_{0,1}$	SC_{0-7} (Note 6)		25		18		18		18
			$DATA_{0-31}$		35		26		26		23
			$\overline{ERROR}$		29		21		21		17
			$\overline{MULT\ ERROR}$		35		26		26		21
8	t_{PD}	LE_{IN} (From Latched to Transparent)	SC_{0-7}		37		27		27		22
			$DATA_{0-31}$ (Note 2)		51		38		38		28
			$\overline{ERROR}$		41		30		30		19
			$\overline{MULT\ ERROR}$		45		33		33		22
9	t_{PD}	LE_{OUT} (From Latched to Transparent)	$DATA_{0-31}$		17		12		12		12
10	t_{PD}	LE_{DIAG} (From Latched to Transparent; Not Internal Control Mode)	SC_{0-7} (Note 6)		21		15		15		15
			$DATA_{0-31}$		38		29		29		24
			$\overline{ERROR}$		26		19		19		15
			$\overline{MULT\ ERROR}$		30		22		22		19

SWITCHING CHARACTERISTICS over COMMERCIAL operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C660		Am29C660A		Am29C660B		Am29C660C	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
11	t_{PD}	Internal Control Mode: LE_{DIAG} (From Latched to Transparent)	SC_{0-7} (Note 6)		22		16		16		16
			$DATA_{0-31}$		42		32		32		22
			$\overline{ERROR}$		26		19		19		16
			$\overline{MULT\ ERROR}$		33		24		24		18
12	t_{PD}	Internal Control Mode: $DATA_{0-31}$ (Via Diagnostic Latch)	SC_{0-7}		22		16		16		16
			$DATA_{0-31}$ (Note 2)		42		32		32		25
			$\overline{ERROR}$		27		20		20		13
			$\overline{MULT\ ERROR}$		34		25		25		16
13	t_{SET}	$DATA_{0-31}$ (Note 4)	LE_{IN}	6		5		4		3	
14	t_{HOLD}			4		4		4		4	
15	t_{SET}	CB_{0-7} (Note 4)		5		5		4		2	
16	t_{HOLD}			4		4		4		4	
17	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{OUT}	30		23		19		6	
18	t_{HOLD}			0		0		0		0	
19	t_{SET}	CB_{0-7} (Note 4) (CODE ID 00, 11)		20		15		15		14	
20	t_{HOLD}			0		0		0		0	
21	t_{SET}	CB_{0-7} (CODE ID 10) (Note 4)		20		15		15		8	
22	t_{HOLD}			0		0		0		0	
23	t_{SET}	CORRECT (Note 4)		16		11		11		8	
24	t_{HOLD}			0		0		0		0	
25	t_{SET}	DIAG $MODE_{0,1}$ (Note 4)		23		17		17		17	
26	t_{HOLD}			0		0		0		0	
27	t_{SET}	CODE $ID_{0,1}$ (Note 4)		23		17		17		10	
28	t_{HOLD}			0		0		0		0	
29	t_{SET}	LE_{IN} (Notes 4 & 6)		31		25		20		19	
30	t_{HOLD}			0		0		0		0	
31	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{DIAG}	6		5		4		3	
32	t_{HOLD}			3		3		3		3	
33	t_{EN}	$\overline{OE}$ $BYTE_{0-3}$ (Notes 5 & 6)	$DATA_{0-31}$		27		23		23		15
34	t_{DIS}				23		19		19		13
35	t_{EN}	$\overline{OE}_{SC}$ (Notes 5 & 6)	SC_{0-7}		28		24		24		17
36	t_{DIS}				24		20		20		13
37	t_{PW}	Minimum Pulse Width: LE_{IN}, LE_{OUT}, LE_{DIAG}		12		9		9		6	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations and are not tested in production.
3. Data In or LE_{IN} to Correct Data Out measurement requires timing as shown in the Switching Waveforms.
4. Setup and Hold times relative to Latch Enables (Latching up data).
5. Output disable tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.
6. Not production tested. Guaranteed by characterization.

SWITCHING CHARACTERISTICS over operating ranges (for APL Products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted) (Notes 1 and 2)

No.	Parameter Symbol	Data Path Description		Am29C660D		Am29C660E*		Unit
		From Input	To Output	Min.	Max.	Min.	Max.	
1	t_{PD}	$DATA_{0-31}$ (Note 3)	SC_{0-7}		14			ns
			$DATA_{0-31}$ (Note 2)		18			ns
			$\overline{ERROR}$		12			ns
			$\overline{MULT\ ERROR}$		15			ns
2	t_{PD}	CB_{0-7} (CODE ID 00, 11)	SC_{0-7}		11			ns
			$DATA_{0-31}$		16			ns
			$\overline{ERROR}$		10			ns
			$\overline{MULT\ ERROR}$		12			ns
3	t_{PD}	CB_{0-7} (CODE ID 10)	SC_{0-7}		12			ns
			$DATA_{0-31}$		12			ns
4	t_{PD}	$\overline{GENERATE}$	SC_{0-7}		14			ns
			$\overline{ERROR}$		7			ns
			$\overline{MULT\ ERROR}$		8			ns
5	t_{PD}	CORRECT (Not Internal Control Mode)	$DATA_{0-31}$		12			ns
6	t_{PD}	DIAG $MODE_{0,1}$ (Not Internal Control Mode)	SC_{0-7}		12			ns
			$DATA_{0-31}$		20			ns
			$\overline{ERROR}$		8			ns
			$\overline{MULT\ ERROR}$		15			ns
7	t_{PD}	CODE $ID_{0,1}$	SC_{0-7} (Note 6)		14			ns
			$DATA_{0-31}$		18			ns
			$\overline{ERROR}$		13			ns
			$\overline{MULT\ ERROR}$		16			ns
8	t_{PD}	LE_{IN} (From Latched to Transparent)	SC_{0-7}		17			ns
			$DATA_{0-31}$ (Note 2)		21			ns
			$\overline{ERROR}$		14			ns
			$\overline{MULT\ ERROR}$		17			ns
9	t_{PD}	LE_{OUT} (From Latched to Transparent)	$DATA_{0-31}$		9			ns
10	t_{PD}	LE_{DIAG} (From Latched to Transparent; Not Internal Control Mode)	SC_{0-7} (Note 6)		12			ns
			$DATA_{0-31}$		18			ns
			$\overline{ERROR}$		12			ns
			$\overline{MULT\ ERROR}$		14			ns

*In development

SWITCHING CHARACTERISTICS over operating ranges (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C660D		Am29C660E*		Unit
		From Input	To Output	Min.	Max.	Min.	Max.	
11	t_{PD}	Internal Control Mode: LE_{DIAG} (From Latched to Transparent)	SC_{0-7} (Note 6)		12			ns
			$DATA_{0-31}$		17			ns
			$\overline{ERROR}$		12			ns
			$\overline{MULT\ ERROR}$		14			ns
12	t_{PD}	Internal Control Mode: $DATA_{0-31}$ (Via Diagnostic Latch)	SC_{0-7}		12			ns
			$DATA_{0-31}$ (Note 2)		19			ns
			$\overline{ERROR}$		10			ns
			$\overline{MULT\ ERROR}$		12			ns
13	t_{SET}	$DATA_{0-31}$ (Note 4)	LE_{IN}	3				ns
14†	t_{HOLD}			3				ns
15	t_{SET}	CB_{0-7} (Note 4)		2				ns
16†	t_{HOLD}			3				ns
17	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{OUT}	5				ns
18†	t_{HOLD}			0				ns
19	t_{SET}	CB_{0-7} (Note 4) (CODE ID 00, 11)		11				ns
20†	t_{HOLD}			0				ns
21	t_{SET}	CB_{0-7} (CODE ID 10) (Note 4)		6				ns
22†	t_{HOLD}			0				ns
23	t_{SET}	CORRECT (Note 4)		6				ns
24†	t_{HOLD}			0				ns
25	t_{SET}	DIAG $MODE_{0,1}$ (Note 4)		13				ns
26†	t_{HOLD}			0				ns
27	t_{SET}	CODE $ID_{0,1}$ (Note 4)		8				ns
28†	t_{HOLD}			0				ns
29	t_{SET}	LE_{IN} (Notes 4 & 6)		14				ns
30†	t_{HOLD}			0				ns
31	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{DIAG}	3				ns
32†	t_{HOLD}			3				ns
33	t_{EN}	$\overline{OE}$ $BYTE_{0-3}$ (Notes 5 & 6)	$DATA_{0-31}$	4	7			ns
34	t_{DIS}			6	8			ns
35	t_{EN}	$\overline{OE}_{SC}$ (Notes 5 & 6)	SC_{0-7}	3	8			ns
36	t_{DIS}			4	10			ns
37	t_{PW}	Minimum Pulse Width: LE_{IN}, LE_{OUT}, LE_{DIAG}		5				ns

Notes: 1. $C_L = 50$ pF.
2. These parameters are combinational propagation delay calculations, and not tested in production.
3. Data In or LE_{IN} to Correct Data Out measurement requires timing as shown in the Switching Waveforms.
4. Setup and Hold times relative to Latch Enables (Latching up data).
5. Output tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.
6. Not production tested. Guaranteed by characterization.

† = Not Included in Group A Tests.

*In development

SWITCHING CHARACTERISTICS over **MILITARY** operating range (for APL Products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted) (Notes 1 and 2)

The following table specifies the guaranteed device performance over the Military and Extended-Commercial operating ranges of −55°C to +125°C (case), with V_{CC} 4.5 to 5.5 V and 4.75 to 5.25 V, respectively. All input switching is between 0 V and 3 V at 1 V/ns and measurements are made at 1.5 V. All outputs have maximum DC load. All units are in nanoseconds (ns).

No.	Parameter Symbol	Data Path Description		Am29C660		Am29C660A		Am29C660B		Am29C660C	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
1	t_{PD}	$DATA_{0-31}$ (Note 3)	SC_{0-7}		40		30		28		22
			$DATA_{0-31}$ (Note 2)		52		39		33		29
			$\overline{ERROR}$		44		33		28		21
			$\overline{MULT\ ERROR}$		48		36		30		24
2	t_{PD}	CB_{0-7} (CODE ID 00, 11)	SC_{0-7}		25		19		17		17
			$DATA_{0-31}$		49		37		33		23
			$\overline{ERROR}$		29		22		20		16
			$\overline{MULT\ ERROR}$		34		26		23		18
3	t_{PD}	CB_{0-7} (CODE ID 10)	SC_{0-7}		25		19		19		17
			$DATA_{0-31}$		33		23		23		18
4	t_{PD}	$\overline{GENERATE}$	SC_{0-7}		32		24		24		20
			$\overline{ERROR}$		33		28		26		10
			$\overline{MULT\ ERROR}$		33		28		26		12
5	t_{PD}	CORRECT (Not Internal Control Mode)	$DATA_{0-31}$		34		26		26		17
6	t_{PD}	$DIAG\ MODE_{0,1}$ (Not Internal Control Mode)	SC_{0-7}		26		20		20		18
			$DATA_{0-31}$		38		29		29		29
			$\overline{ERROR}$		30		23		23		12
			$\overline{MULT\ ERROR}$		36		27		27		23
7	t_{PD}	$CODE\ ID_{0,1}$	SC_{0-7} (Note 6)		28		21		21		21
			$DATA_{0-31}$		38		29		29		26
			$\overline{ERROR}$		32		24		24		20
			$\overline{MULT\ ERROR}$		38		29		29		24
8	t_{PD}	LE_{IN} (From Latched to Transparent)	SC_{0-7}		40		30		30		24
			$DATA_{0-31}$ (Note 2)		54		41		41		32
			$\overline{ERROR}$		44		33		33		21
			$\overline{MULT\ ERROR}$		48		36		36		25
9	t_{PD}	LE_{OUT} (From Latched to Transparent)	$DATA_{0-31}$		20		15		15		13
10	t_{PD}	LE_{DIAG} (From Latched to Transparent; Not Internal Control Mode)	SC_{0-7} (Note 6)		26		18		18		18
			$DATA_{0-31}$		42		32		32		27
			$\overline{ERROR}$		29		22		22		17
			$\overline{MULT\ ERROR}$		33		25		25		21

SWITCHING CHARACTERISTICS over MILITARY operating range (Cont'd.)

No.	Parameter Symbol	Data Path Description		Am29C660		Am29C660A		Am29C660B		Am29C660C	
		From Input	To Output	Min.	Max.	Min.	Max.	Min.	Max.	Min.	Max.
11	t_{PD}	Internal Control Mode: LE_{DIAG} (From Latched to Transparent)	SC_{0-7} (Note 6)		25		19		19		19
			$DATA_{0-31}$		47		35		35		25
			$\overline{ERROR}$		29		22		22		18
			$\overline{MULT\ ERROR}$		36		27		27		21
12	t_{PD}	Internal Control Mode: $DATA_{0-31}$ (Via Diagnostic Latch)	SC_{0-7}		25		19		19		18
			$DATA_{0-31}$ (Note 2)		47		35		35		29
			$\overline{ERROR}$		30		23		23		14
			$\overline{MULT\ ERROR}$		37		28		28		18
13	t_{SET}	$DATA_{0-31}$ (Note 4)	LE_{IN}	6		5		4		3	
14†	t_{HOLD}			4		4		4		4	
15	t_{SET}	CB_{0-7} (Note 4)		5		5		4		2	
16†	t_{HOLD}			4		4		4		4	
17	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{OUT}	36		27		23		7	
18†	t_{HOLD}			0		0		0		0	
19	t_{SET}	CB_{0-7} (Note 4) (CODE ID 00, 11)		24		18		18		16	
20†	t_{HOLD}			0		0		0		0	
21	t_{SET}	CB_{0-7} (CODE ID 10) (Note 4)		24		18		18		10	
22†	t_{HOLD}			0		0		0		0	
23	t_{SET}	CORRECT (Note 4)		20		14		14		9	
24†	t_{HOLD}			0		0		0		0	
25	t_{SET}	DIAG $MODE_{0,1}$ (Note 4)		28		20		20		19	
26†	t_{HOLD}			0		0		0		0	
27	t_{SET}	CODE $ID_{0,1}$ (Note 4)		28		20		20		12	
28†	t_{HOLD}			0		0		0		0	
29	t_{SET}	LE_{IN} (Notes 4 & 6)		37		28		23		21	
30†	t_{HOLD}			0		0		0		0	
31	t_{SET}	$DATA_{0-31}$ (Notes 4 & 6)	LE_{DIAG}	6		5		4		3	
32†	t_{HOLD}			3		3		3		3	
33	t_{EN}	$\overline{OE}$ $BYTE_{0-3}$ (Notes 5 & 6)	$DATA_{0-31}$		29		25		25		17
34	t_{DIS}				25		21		21		15
35	t_{EN}	$\overline{OE}_{SC}$ (Notes 5 & 6)	SC_{0-7}		30		27		27		18
36	t_{DIS}				26		22		22		15
37	t_{PW}	Minimum Pulse Width: LE_{IN}, LE_{OUT}, LE_{DIAG}		15		12		12		6	

Notes: 1. $C_L = 50$ pF.
2. Certain parameters are combinational propagation delay calculations and are not tested in production.
3. Data In or LE_{IN} to Correct Data Out measurement requires timing as shown in the Switching Waveforms.
4. Setup and Hold times relative to Latch Enables (Latching up data).
5. Output tests specified with $C_L = 5$ pF and measured to 0.5 V change of output voltage level. Testing is performed at $C_L = 50$ pF and correlated to $C_L = 5$ pF.
6. Not production tested. Guaranteed by characterization.

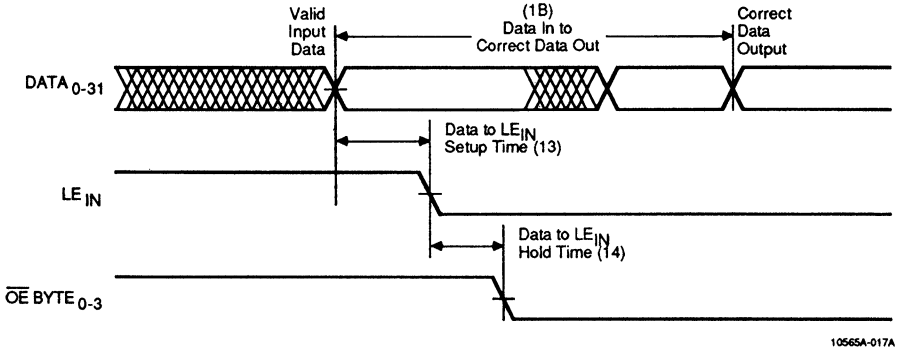
† = Not Included in Group A Tests.

SWITCHING WAVEFORMS

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
———	MUST BE STEADY	WILL BE STEADY
———		
⟋⟋⟋	MAY CHANGE FROM H TO L	WILL BE CHANGING FROM H TO L
⟋⟋⟋	MAY CHANGE FROM L TO H	WILL BE CHANGING FROM L TO H
⟋⟋⟋	DON'T CARE; ANY CHANGE PERMITTED	CHANGING; STATE UNKNOWN
⟋⟋⟋	DOES NOT APPLY	CENTER LINE IS HIGH IMPEDANCE "OFF" STATE

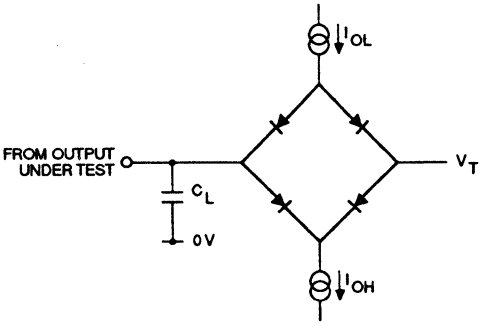
KS000010



WF026320

$DATA_{0-31}/LE_{IN}$ to Correct Data Out

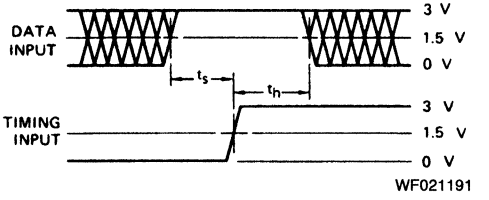
SWITCHING TEST CIRCUIT



AF004810

Notes: 1. C_L = 50 pF for all tests except output enable/disable (includes scope probe, wiring, and stray capacitance without device in test fixture).
2. C_L = 5 pF for output enable/disable tests.
3. V_T = 1.5 V.

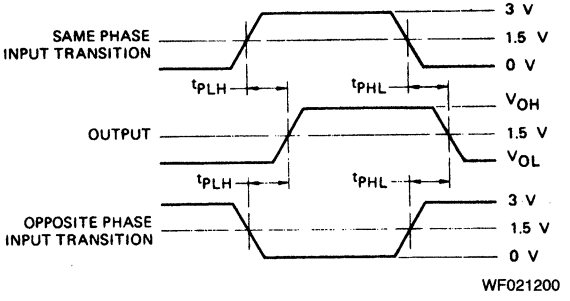
SWITCHING TEST WAVEFORMS



WF021191

Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross-hatched area is don't care condition.

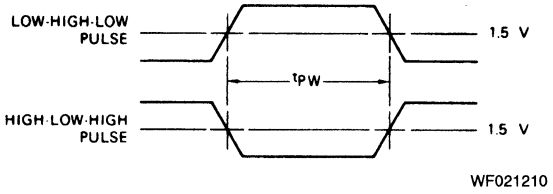
Setup and Hold Times



WF021200

Propagation Delay

SWITCHING TEST WAVEFORMS (Cont'd.)



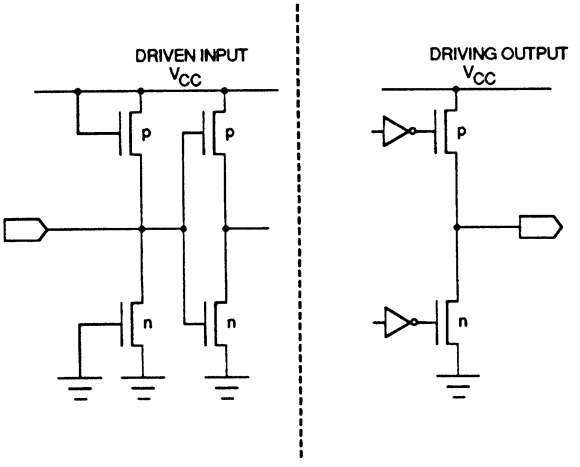
Pulse Width

Notes on Testing

Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Ensure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5-8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by hundreds of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leqslant 0$ V and $V_{IH} \geqslant 3$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. Changing the CODE ID inputs can cause loss of data in some of the Am29C660 internal latches. Specifically, the entire checkbit latch and bits 6 and 7 of the diagnostic latch are indeterminate after a change in CODE ID inputs. Logic simulations should store "x" (i.e., "don't care") in these bits after CODE ID change. Test programs should reload these registers before they are used.
7. Proper device grounding is critical when device testing. Multi-layer performance boards with radial decoupling between power and ground planes is recommended. Wiring unused interconnect pins to the ground plane is recommended. The ground plane must be sustained from the performance board to the device under test interface board. To minimize inductance, heavy-guage stranded wire with twisted pairs should be used for power wiring.

EQUIVALENT INPUT/OUTPUT CIRCUIT DIAGRAMS



Am29C668

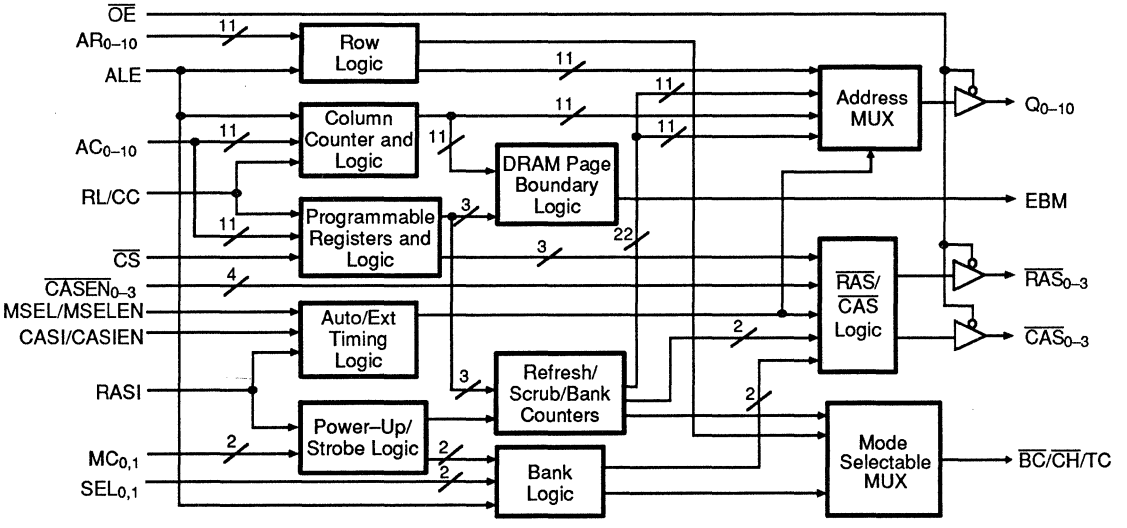
4M Configurable Dynamic Memory Controller/Driver

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Provides control for 4M, 1M, 256K, 64K dynamic RAMs
- Programmable Burst/Block Access support for Am29000,68000 family, iAPX family
- Proprietary "Cache" Mode supports Page Mode accessing
- Single-chip Bank Interleaving saves precharge time
- Nibble mode support (for Page Mode or Nibble Mode DRAMs)
- Selectable Address and Strobe autotiming or external timing
- Supports "Scrubbing" with refresh when used in an EDC system
- Supports CAS before RAS refresh
- Byte and Bank CAS Decoding
- Selectable 2 or 4 bank drive
- Outputs directly drive up to 88 DRAMs, with a guaranteed worst-case limit on the undershoot and overshoot
- Low-power advanced sub-micron CMOS process
- User configurable to replace Am2968A and Am29368 DMCs

BLOCK DIAGRAM



11068–001B

Publication # 11068 Rev. B Amendment /0
Issue Date: March 1990

GENERAL DESCRIPTION

The Am29C668 4M Configurable Dynamic Memory Controller/Driver (CDMC) is designed for high performance memory systems. The CDMC acts as the address controller between the processor and the dynamic memory array. It uses its 11-bit row latch and 11-bit-column latch and counter to hold the row and column addresses, respectively, for multiplexing these to any DRAM size up to 4M. These latches and counter and the row/column refresh counter are used to directly drive the address lines of the DRAM array. The output of the 2-bit bank latch is decoded to select the bank to be accessed.

The Am29C668 has two basic modes of operation, read/write and refresh. In the read/write mode, the Am29C668 latches the row, column, and bank addresses and multiplexes them to the DRAM array. This multiplexing occurs under the control of the internally-generated timing strobes in the Auto Timing Mode, or the externally-generated MSEL in External Timing Mode. The read/write mode of the Am29C668 may be optimized for the shortest memory access time, through burst/block access, "cache" mode access, nibble mode access, or bank interleaving.

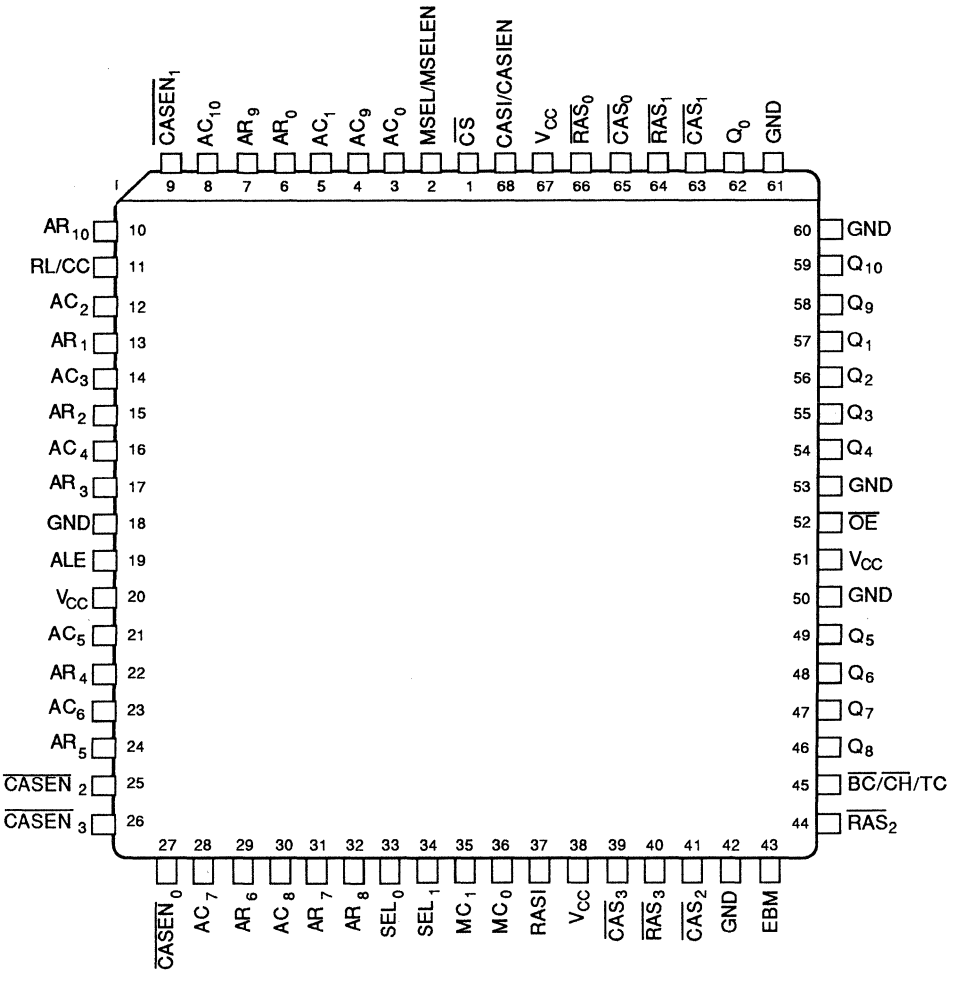
In the refresh mode, the refresh address is generated by the Am29C668 refresh counter. This counter is automatically adjusted for different DRAM sizes. If memory scrubbing is not being implemented, only the row counter is used to generate the row address for refresh. When memory scrubbing is being performed in EDC systems, both the row and column address counters are used.

RELATED AMD PRODUCTS

Part No.	Description
Am29368	1M Dynamic Memory Controller/Driver
Am29C60A	High Speed CMOS Cascadable 16-Bit EDC
Am29C660D	12 ns CMOS Cascadable 32-Bit EDC
Am2968A	256K Dynamic Memory Controller/Driver
Am2976	11-Bit Dynamic RAM Driver
Am29C983	9-Bit x 4-Port Multiple Bus Exchange
Am2965/6	8-Bit Dynamic RAM Driver Inverting/Non-Inverting
Am29C983A	9-Bit x 4-Port Multiple Bus Exchange, High Speed
Am29C985	9-Bit x 4-Port MBE with Parity
Am29C827A	10-Bit 48 mA Bus Buffer

CONNECTION DIAGRAMS
Top View

PLCC



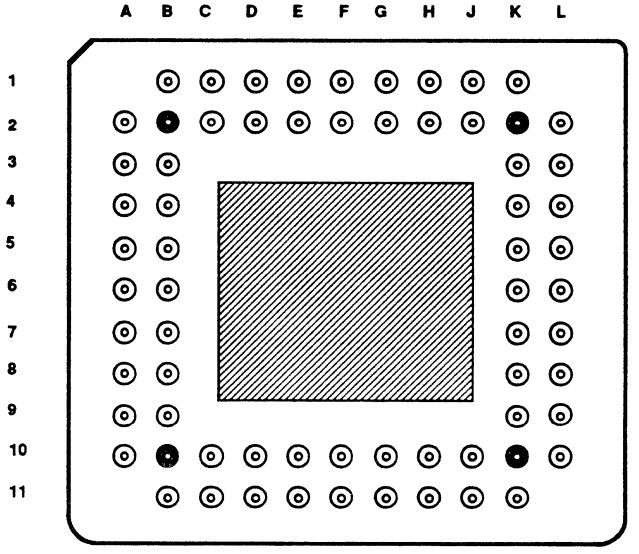
11068–002A

Note: Pin1 is marked for orientation (PLCC only).

CONNECTION DIAGRAM

Top View (Pins Pointing **Down**)

PGA*



11068–003A

*Pinout matches socketed PLCC pinout and footprint.

PGA PIN DESIGNATIONS

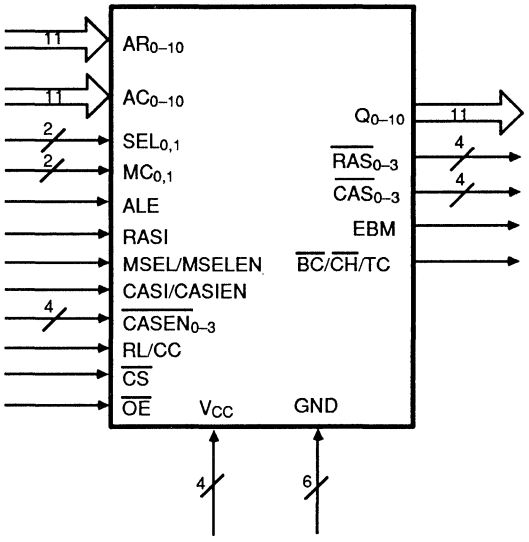
(Sorted by Pin Number)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
A-2	AR_{10}	**B-9**	$\overline{CASEN_2}$	**F-10**	MC_0	**K-4**	Q_1
A-3	AC_2	**B-10**	AC_7	**F-11**	MC_1	**K-5**	Q_3
A-4	AC_3	**B-11**	$\overline{CASEN_0}$	**G-1**	V_{CC}	**K-6**	GND
A-5	AC_4	**C-1**	AR_9	**G-2**	CASI/CASIEN	**K-7**	V_{CC}
A-6	GND	**C-2**	AC_{10}	**G-10**	V_{CC}	**K-8**	Q_5
A-7	V_{CC}	**C-10**	AC_8	**G-11**	RASI	**K-9**	Q_7
A-8	AR_4	**C-11**	AR_6	**H-1**	$\overline{CAS_0}$	**K-10**	$\overline{BC/CH}$/TC
A-9	AR_5	**D-1**	AC_1	**H-2**	$\overline{RAS_0}$	**K-11**	EBM
A-10	$\overline{CASEN_3}$	**D-2**	AR_0	**H-10**	$\overline{RAS_3}$	**L-2**	GND
B-1	$\overline{CASEN_1}$	**D-10**	AR_8	**H-11**	$\overline{CAS_3}$	**L-3**	Q_9
B-2	RL/CC	**D-11**	AR_7	**J-1**	$\overline{CAS_1}$	**L-4**	Q_2
B-3	AR_1	**E-1**	AC_0	**J-2**	$\overline{RAS_1}$	**L-5**	Q_4
B-4	AR_2	**E-2**	AC_9	**J-10**	GND	**L-6**	$\overline{OE}$
B-5	AR_3	**E-10**	SEL_1	**J-11**	$\overline{CAS_2}$	**L-7**	GND
B-6	ALE	**E-11**	SEL_0	**K-1**	GND	**L-8**	Q_6
B-7	AC_5	**F-1**	$\overline{CS}$	**K-2**	Q_0	**L-9**	Q_8
B-8	AC_6	**F-2**	MSEL/MSELEN	**K-3**	Q_{10}	**L-10**	$\overline{RAS_2}$

(Sorted by Pin Name)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
E-1	**AC_0**	C-11	**AR_6**	K-11	**EBM**	L-8	**Q_6**
D-1	**AC_1**	D-11	**AR_7**	A-6	**GND**	K-9	**Q_7**
A-3	**AC_2**	D-10	**AR_8**	J-10	**GND**	L-9	**Q_8**
A-4	**AC_3**	C-1	**AR_9**	K-1	**GND**	L-3	**Q_9**
A-5	**AC_4**	A-2	**AR_{10}**	K-6	**GND**	K-3	**Q_{10}**
B-7	**AC_5**	B-6	**ALE**	L-2	**GND**	H-2	**$\overline{RAS_0}$**
B-8	**AC_6**	K-10	**$\overline{BC/CH}$/TC**	L-7	**GND**	J-2	**$\overline{RAS_1}$**
B-10	**AC_7**	H-1	**$\overline{CAS_0}$**	F-10	**MC_0**	L-10	**$\overline{RAS_2}$**
C-10	**AC_8**	J-1	**$\overline{CAS_1}$**	F-11	**MC_1**	H-10	**$\overline{RAS_3}$**
E-2	**AC_9**	J-11	**$\overline{CAS_2}$**	F-2	**MSEL/MSELEN**	G-11	**RASI**
C-2	**AC_{10}**	H-11	**$\overline{CAS_3}$**	L-6	**$\overline{OE}$**	B–2	**RL/CC**
D-2	**AR_0**	B-11	**$\overline{CASEN_0}$**	K-2	**Q_0**	E-11	**SEL_0**
B-3	**AR_1**	B-1	**$\overline{CASEN_1}$**	K-4	**Q_1**	E-10	**SEL_1**
B-4	**AR_2**	B-9	**$\overline{CASEN_2}$**	L-4	**Q_2**	A-7	**V_{CC}**
B-5	**AR_3**	A-10	**$\overline{CASEN_3}$**	K-5	**Q_3**	G-1	**V_{CC}**
A-8	**AR_4**	G-2	**CASI/CASIEN**	L-5	**Q_4**	G-10	**V_{CC}**
A-9	**AR_5**	F-1	**$\overline{CS}$**	K-8	**Q_5**	K-7	**V_{CC}**

LOGIC SYMBOL



Die Size: 0.233" x 0.165" Gate Count: 3600 11068–004A

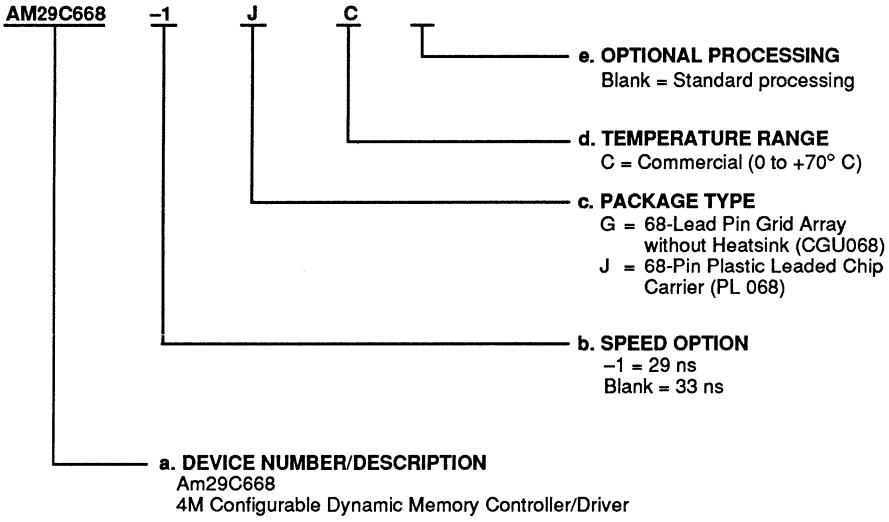
Package Information

Parameter	PGA	PLCC	Units
θ_{JA}	34	35	°C/W
θ_{JC}	–	N/A	

ORDERING INFORMATION
Standard Products

AMD standard products are available in several packages and operating ranges. The ordering number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29C668	JC, GC
AM29C668-1	

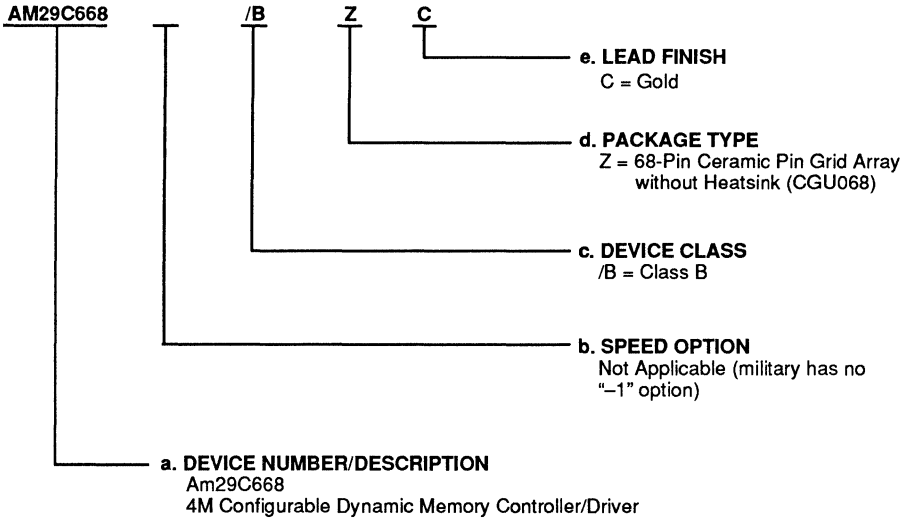
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

ORDERING INFORMATION
APL Products

AMD standard products are available in several packages and operating ranges. The ordering number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29C668	/BZC

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11

PIN DESCRIPTION

AC_{0-10} and AR_{0-10}

Column and Row Address Inputs (Inputs(22))

The address on these lines is latched by the LOW going edge of the Address Latch Enable (ALE) signal. AC_{0-10} are connected to the lower side of the system address bus and are driven on the output address lines Q_{0-10} when the MSEL (Multiplexer Select) signal is HIGH. AR_{0-10} are connected to the upper side of the system address bus and are driven on the output address lines Q_{0-10} when the MSEL signal is LOW.

AC_{0-7} and AR_{0-7} are used for 64K DRAMs.
AC_{0-8} and AR_{0-8} are used for 256K DRAMs.
AC_{0-9} and AR_{0-9} are used for 1M DRAMs.
AC_{0-10} and AR_{0-10} are used for 4M DRAMs.

ALE

Address Latch Enable (Input; Active HIGH)

This input causes the Row Latch, Column Latch and Counter, and the Bank Latch to become transparent allowing the latches to accept new input data. A LOW input on ALE latches the input data, assuming it meets specified set-up and hold requirements.

$\overline{BC}/\overline{CH}/TC$

$\overline{Bank\ Compare}/\overline{Cache\ Hit}$/Terminal Count (Outputs; Active LOW/LOW/HIGH)

This is a triple function output, dependent upon the Mode Control inputs and the Bank Interleave (BI) Bit in the Configuration Register. If the Mode Control inputs are 1,0 ($MC_{0,1}$ = 10) the Am29C668 is in the Refresh-With-Scrubbing or Initialize mode and this output acts as the Terminal Count (TC).

As **Terminal Count**, this output goes active (HIGH) when the Refresh Counter has gone through an entire count. The Refresh Counter is user configured for DRAM size (64K, 256K, 1M, 4M) and number of banks (2 or 4), which are programmable via the Memory Size Bits and the $\overline{RAS}/\overline{CAS}$ Configuration Bit, respectively, in the Configuration Register (Reference Figure 6). The TC signal is used to indicate the end of initialization in an Error Detection and Correction (EDC) system.

In the read/write mode ($MC_{0,1}$ = 01), this output acts as either the $\overline{Bank\ Compare}$ ($\overline{BC}$) signal if BI = 0, or as the $\overline{Cache\ Hit}$ ($\overline{CH}$) signal if BI = 1. As **$\overline{BC}$**, this output goes active (LOW) when the current memory access is to the same bank as the previous memory access and remains active until a memory access to a different bank is requested. This signal is used by the external timing generator during bank interleaving to either immediately activate the RASI input if two consecutive accesses are to two different banks (saving RAS precharge time) or delay the RASI input if two consecutive accesses are to the same bank (allowing the current $\overline{RAS}_n$ output to go through a precharge cycle before it is reactivated).

As **$\overline{Cache\ Hit}$**, this output goes active (LOW) when the current memory access is to the same row and the same bank as the previous access. The $\overline{CH}$ signal is used to facilitate Fast Page Mode or Static Column accesses.

$\overline{CAS}_{0-3}$

$\overline{Column\ Address\ Strobe}$ (Outputs (4); Active LOW; Three State)

Each $\overline{CAS}_n$ output will go active when selected by $SEL_{0,1}$ in a bank-wise $\overline{CAS}$ decoding method (CDM = 0, reference Figure 6) or when selected by $\overline{CASEN}_{0-3}$ in a byte-wise $\overline{CAS}$ decoding implementation. This will occur only when CASI goes active in the External Timing Mode or when CASIEN and the internally generated $\overline{CAS}$ go active in the Auto Timing Mode.

Each output provides a $\overline{CAS}_n$ signal to one of four banks of the dynamic memory, if four banks are used. If two banks are used, each bank can use 2 $\overline{CAS}_n$ signals to reduce the capacitive load on each. The number of banks (2 or 4) is programmable via the $\overline{RAS}/\overline{CAS}$ Configuration Bit in the Configuration Register (reference Table 4).

The $\overline{CAS}_n$ outputs contain pull-up resistors which ensure a logical HIGH (inactive) when in the high impedence state.

$\overline{CASEN}_{0-3}$

$\overline{Column\ Address\ Strobe\ Enable}$ (Inputs (4); Active LOW)

When a byte-wise method is used for $\overline{CAS}$ decoding these four inputs are decoded externally to handle byte operations. The timing generation may be Auto or External. Only those $\overline{CAS}_n$ outputs will be activated whose corresponding $\overline{CASEN}_n$ inputs are selected by the external byte decode circuit.

When a bank-wise method is used for $\overline{CAS}$ decoding these inputs are not used.

CASI/CASIEN

Column Addess Strobe Input/Column Address Strobe Input Enable (Input; Active HIGH)

This is a dual function input. In the External Timing mode this input is used as **CASI**. With a bank-wise $\overline{CAS}$ decoding method, the internally decoded $\overline{CAS}_n$ output is forced LOW after CASI goes active. When used as CASI with a byte-wise decoding method, the selected $\overline{CAS}_n$ output is forced LOW depending upon the externally decoded $\overline{CASEN}_n$ inputs after CASI goes active.

In Auto Timing Mode this input is used as **CASIEN**. With a bank-wise $\overline{CAS}$ decoding method, the decoded $\overline{CAS}_n$ output is forced LOW, if both the internally generated $\overline{CAS}$ and the CASIEN signals are active. This input is used to delay the $\overline{CAS}_{0-3}$ outputs from going active if de-

sired, resulting in a longer auto timing access sequence. This input is generally not used as CASIEN with a byte-wise $\overline{CAS}$ decoding implementation.

$\overline{CS}$

Chip Select (Input; Active LOW)

This input is used to enable the Am29C668. When active, the Am29C668 operates normally in all four modes. When $\overline{CS}$ goes inactive (HIGH), the device will not enter the Read/Write mode.

EBM

End Burst/Block Mode (Output Active HIGH)

This output is only used in the burst/block mode of data transfer. It indicates to the processor that the Am29C668 cannot perform any more data transfers in the burst/block mode for one of two reasons. Either the DRAM page boundary is reached (in which case a new row address is required from the processor), or a programmed allowable number of transfers has been completed.

GND (6) 0-V Power Supply

These pins are the 0-V power supply for the Am29C668. All grounds must be connected for proper device operation.

$MC_{0,1}$ Mode Control (Inputs (2))

These inputs specify one of four modes of operation of the Am29C668. Operating modes are described in Table 1.

MSEL/MSELEN

Multiplexer Select/Multiplexer Select Enable (Input; Active HIGH)

This is a dual function input. In the External Timing mode this input is used as **MSEL**. When MSEL is HIGH the column address is selected. When MSEL is LOW the row address is selected.

In the Auto Timing Mode this input acts as **MSELEN**. When MSELEN is HIGH and the internally generated MSEL is active the column address is selected. When MSELEN is LOW or the internally generated MSEL is inactive the row address is selected. MSELEN is used to delay the address change from row to column, if desired, resulting in a longer auto timing access sequence.

The address may come form either the address latches and counter or the refresh address counter depending upon $MC_{0,1}$. The MSEL/MSELEN input is only applicable in the Read/Write or Refresh with Scrubbing Modes.

$\overline{OE}$

Output Enable (Input; Active LOW)

This input enables/disables the output signals. When $\overline{OE}$ is inactive (HIGH), all address outputs of the Am29C668 enter a high impedence state and the $\overline{RAS}_n$ and $\overline{CAS}_n$ outputs are pulled inactive (HIGH).

$Q_{0\text{-}10}$

Address Outputs (Outputs(11);Three State)

These edge rate controlled outputs drive the dynamic memory address inputs. The drivers on these lines are able to drive high capacitive loads, which are specified at 350pF. Greater capacitive loads may also be driven, however. See section labeled "Typical Change in Propagation Delay vs Loading Capacitance" following the AC Characteristics.

$\overline{RAS}_{0\text{-}3}$

Row Address Strobe (Outputs (4); Active LOW; Three State)

Each Row Address Strobe output provides a $\overline{RAS}_n$ signal to one of four memory banks. Each will go low when selected by $SEL_{0,1}$ and only when RASI goes HIGH. All four go LOW in response to RASI in the refresh modes.

When a 2 bank $\overline{RAS}/\overline{CAS}$ configuration is selected (RCC = 1), RAS_0 and RAS_1 are tied together internally, as are RAS_2 and RAS_3. This reduces the capacitive loading on the $\overline{RAS}_n$ outputs in a two bank system (reference Table 4).

In four bank mode, the $\overline{RAS}_n$ outputs are decoded with $SEL_{0,1}$. In two bank mode these outputs are decoded with SEL_0. In this case SEL_1, should be tied LOW.

The RAS_n outputs contain pull-up resistors which ensure a logical HIGH (inactive) when in the high impedence state.

RASI

Row Address Strobe Input (Input; Active HIGH)

During normal memory cycles, the decoded $\overline{RAS}_n$ outputs ($\overline{RAS}_0$,$\overline{RAS}_1$,$\overline{RAS}_2$,$\overline{RAS}_3$) as determined by $SEL_{0,1}$ and the RCC bit in the Configuration Register are forced LOW after RASI goes active HIGH. During refresh, all four RAS_n outputs go LOW after RASI goes active HIGH. If auto timing is enabled, the HIGH going edge of RASI also initiates the internal timing cycle and its LOW going edge terminates the internal timing cycle.

RL/CC

Register Load/Column Clock (Input)

This is a dual function pin which depends upon the Mode Control inputs ($MC_{0,1}$). If $MC_{0,1}$ =11, the Am29C668 is in the Reset Mode and this pin acts as the Register Load signal. If $MC_{0,1}$ = 01 the Am29C668 is in the Read/Write Mode and this input acts as the Column Clock signal.

When used as **Register Load**, the LOW-to-HIGH edge of the signal loads either the Burst Count Register, the Mask Register, or the Configuration Register via the $AC_{0\text{-}10}$ Address Inputs. (Reference Figure 5).

When used as **Column Clock**, the HIGH-to-LOW edge of the signal increments the Column Counter during burst and nibble mode accessing.

$SEL_{0,1}$
Bank Select (Inputs (2))

These two inputs are the highest-order address bits when the Am29C668 is used in the normal access mode or in the burst/block access mode. They are the two lowest-order address bits when the Am29C668 is used in the bank interleave mode. In both cases $SEL_{0,1}$ are used in the Read/Write Mode to select which bank of memory will receive the $\overline{RAS}_n$ and $\overline{CAS}_n$ signals when RASI and CASI (or the internally generated CAS in the auto-timing mode) go active HIGH. The $\overline{CAS}_n$ signals will not be decoded from $SEL_{0,1}$ if a byte-wise $\overline{CAS}$ decoding scheme is selected. In two bank mode, only SEL_0 is used. SEL_1 should be tied LOW.

V_{CC} (4) + 5-V
Positive Power Supply Voltage

These inputs provide the power necessary to operate the Am29C668. All power supply inputs must be connected for proper device operation.

Table 1. Mode Control Function

MC_0	MC_1	Operating Mode
0	0	**Refresh Without Scrubbing (a more detailed description can be found in the section entitled "Refresh Modes")** a) $\overline{RAS}$–Only Refresh: Refresh cycles are performed with only the row refresh counter being used to increment addresses. In this mode, all four $\overline{RAS}_n$ outputs are active while the four $\overline{CAS}_n$ outputs are held inactive. b) $\overline{CAS}$ Before $\overline{RAS}$ Refresh: Refresh addresses are generated internally by the DRAMs. In this mode, all four $\overline{CAS}_n$ outputs are active followed by all four $\overline{RAS}_n$ outputs going active. This new type of refresh is selected via the CBR-bit in the Configuration Register. In this mode, RASI controls the $\overline{CAS}_n$ outputs and CASI controls the $\overline{RAS}_n$ outputs.
1	0	**Refresh With Scrubbing/Initialize (EDC Systems)** This mode may be used only in systems with Error Detection and Correction (EDC) capability. In this mode, refresh cycles are performed with both the row and column refresh counters generating the addresses. MSEL is used to select between the row and column addresses. All four $\overline{RAS}_n$ signals go active in response to RASI and one $\overline{CAS}_n$ output goes active in response to CASI. The CAS_n output is decoded from the bank refresh counter. The remaining three $\overline{CAS}_n$ outputs are left inactive, while their respective banks undergo normal refresh. This mode is also used to initialize the memory array by writing a known data pattern and corresponding check bits.
0	1	**Read/Write** This mode is used to perform read/write operations. The row address is taken from the row latch and the column address is taken from the column latch and counter. $SEL_{0,1}$ are decoded to determine which $\overline{RAS}_n$ and $\overline{CAS}_n$ will be active.
1	1	**Reset/Configuration** This mode is used to clear the refresh counters and the Register Logic. These operations are performed on the HIGH-to-LOW transition of RASI. This mode is used to load the configuration, burst count, and mask registers.

Table 2. Address Output Function

$\overline{CS}$	MC_1	MC_0	Internal MSEL	Mode	Address Multiplexer Output
0	0	0	X	Refresh W/O Scrubbing	Row Counter
	0	1	1	Refresh with Scrubbing	Column Counter
			0		Row Counter
	1	0	1	Read/Write	Column Latch
			0		Row Latch
	1	1	X	Reset	All Zero
1	0	0	X	Refresh W/O Scrubbing	Row Counter
	0	1	1	Refresh with Scrubbing	Column Counter
			0		Row Counter
	1	0	X	Read/Write	All Zero
	1	1	X	Reset	All Zero

X = Don't care

Table 3. $\overline{RAS}$ Output Function

Inputs						Outputs			
Internal RASI	$\overline{CS}$	MC	SEL*	RCC**	MODE	$\overline{RAS}_n$			
		1 0	1 0			3	2	1	0
0	X	X X	X X	X	No operation	1	1	1	1
1	0	0 0	X X	X	Refresh W/O Scrubbing	0	0	0	0
		0 1	X X	X	Refresh with Scrubbing	0	0	0	0
		1 0	0 0	0	Read/Write	1	1	1	0
			X 0	1		1	1	0	0
			0 1	0		1	1	0	1
			X 1	1		0	0	1	1
			1 0	0		1	0	1	1
			X 0	1		1	1	0	0
			1 1	0		0	1	1	1
			X 1	1		0	0	1	1
		1 1	X X	X	Reset	0	0	0	0
	1	0 0	X X	X	Refresh W/O Scrubbing	0	0	0	0
		0 1			Refresh with Scrubbing	0	0	0	0
		1 0			Read/Write	1	1	1	1
		1 1			Reset	0	0	0	0

* After Internal RASI is asserted, changing $SEL_{0,1}$ will not effect the $\overline{RAS}_n$ decoding until Internal RASI is deasserted.
** Reference Figure 6.

Table 4. $\overline{RAS}/\overline{CAS}$ Configuration Decode*

RCC	Mode	$\overline{RAS}/\overline{CAS}$ CONFIGURATION		
0	4-Bank	RAS_0	CAS_0	BANK 0
		RAS_1	CAS_1	BANK 1
		RAS_2	CAS_2	BANK 2
		RAS_3	CAS_3	BANK 3
1	2-Bank	RAS_0	CAS_0	BANK 0
		RAS_1	CAS_1	
		RAS_2	CAS_2	BANK 1
		RAS_3	CAS_3	

*CDM = 0

Table 5. $\overline{CAS}$ Output Function

Inputs								Outputs			
CDM**	Internal CASI†	$\overline{CS}$	CBR**	MC	SEL*	Internal Counter	RCC**	$\overline{CAS}_n$			
				1 0	1 0	1 0		3	2	1	0
0	1	0	1	0 0	X X	X X	X	0	0	0	0
			0					1	1	1	1
			1	0 1	X X	X X	X	1	1	1	1
			0			0 0		0	1	1	1	0
								1	1	1	0	0
						0 1		0	1	1	0	1
								1	0	0	1	1
						1 0		0	1	0	1	1
								1	1	1	0	0
						1 1		0	0	1	1	1
								1	0	0	1	1
			X	1 0	0 0	X X		0	1	1	1	0
								1	1	1	0	0
					0 1			0	1	1	0	1
								1	0	0	1	1
					1 0			0	1	0	1	1
								1	1	1	0	0
					1 1			0	0	1	1	1
								1	0	0	1	1
				1 1	X X	X X	X	1	1	1	1	1
		1	1	0 0	X X	X X	X	0	0	0	0	0
			0					1	1	1	1	1
			1	0 1	X X	X X	X	1	1	1	1	1
			0			0 0		0	1	1	1	0
								1	1	1	0	0
						0 1		0	1	1	0	1
								1	0	0	1	1
						1 0		0	1	0	1	1
								1	1	1	0	0
						1 1		0	0	1	1	1
								1	0	0	1	1
			X	1 0	X X	X X	X	1	1	1	1	1
				1 1								
	0	X	X	X X	X X	X X	X	1	1	1	1	1
1	1	X	1	0 0	X X	X X	X	0	0	0	0	0
			0					1	1	1	1	1
			X	0 1	X X	X X	X	1	1	1	1	1
				1 X	X X	X X	X	1	1	1	1	1
			X	X X	X X	X X	X	1	1	1	1	1
	0		X	X X	X X	X X	X	1	1	1	1	1

† In the external timing mode, Internal CASI follows the CASI input. In Autotiming Mode, this signal is generated internally and is enabled by the CASIEN inputs.

* After Internal RASI is asserted, changing $SEL_{0,1}$ will not effect the $\overline{CAS}_n$ decoding until Internal RASI is deasserted.

** Reference Figure 6.

FUNCTIONAL DESCRIPTION*

General Description

The Am29C668 4M Configurable Dynamic Memory Controller/Driver provides the controls required to operate dynamic RAMs up to 4Mbit x n. Manufactured in submicron CMOS technology, the Am29C668 performs the address control and generation function and strobe control and generation for 64K, 256K, 1M or 4M DRAMs. The Am29C668 controls the address to the DRAMs from the processor in the read/write mode and it generates and controls the address to the DRAMs in the refresh mode. The Am29C668 also generates the row and column address strobe signals in the read/write and refresh modes.

The Am29C668 has on-chip series damping resistors on its driver outputs to restrict the output signals to +0.8-V overshoot and -1.0V undershoot maximum (See Switching Waveforms).

Logic Overview

The functional blocks of the Am29C668 can be summarized as follows (reference block diagram):

Row Logic
Column Counter and Logic
Bank Logic
Programmable Registers and Logic
Auto/External Timing Logic
Power-Up/Strobe Logic
DRAM Page Boundary Logic
Refresh/Scrubbing/Bank Counters
Address Multiplexer
$\overline{RAS}/\overline{CAS}$ Logic

Row Logic

This block (Figure 1) consists of a Row Latch, a Register, and a Comparator. The 11-bit Row Latch holds the DRAM row address. It is transparent when the Address Latch Enable signal is HIGH, and the address is latched on the LOW-going edge of ALE.

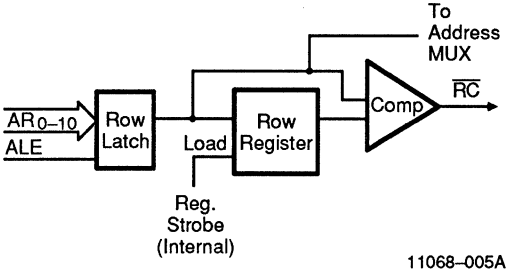


Figure 1. Row Logic

The 11-bit Row Register holds the row address of the previous DRAM access. The register is clocked at the beginning of every access where $MC_{0,1} = 01$ by the HIGH - going edge of the RASI input.

The 11-bit Row Comparator compares the row address of the current access (contents of the Row Latch) with the row address of the previous access (contents of the Row Register) and generates the Row Compare ($\overline{RC}$) signal. The $\overline{RC}$ and $\overline{BC}$ signals are ORed to generate a $\overline{CH}$ signal (Figure 2). $\overline{CH}$ is LOW if the current row and bank addresses are the same as the previous row and bank addresses, respectively. $\overline{CH}$ is high if they are not. This indicator is used by the external timing generator during Cache Mode accesses. The RASI input is held active (HIGH) if consecutive accesses are to the same row in the same bank, saving precharge time and access time on the current $\overline{RAS}_n$. The RASI input is deactivated if consecutive accesses are to different rows or banks, thereby ending the "cache" access.

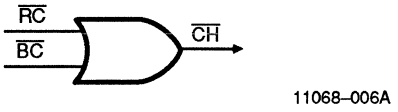


Figure 2. $\overline{\text{Cache Hit}}$ Generation

Column Counter and Logic

The block (Figure 3) consists of the Column Latch and Counter. The 11-bit loadable counter holds the DRAM column address. The counter is transparent when ALE is HIGH and the address is loaded on the LOW-going edge of ALE. The HIGH-to-LOW edge of the signal increments the Column Counter. ALE must be LOW in order to increment the counter.

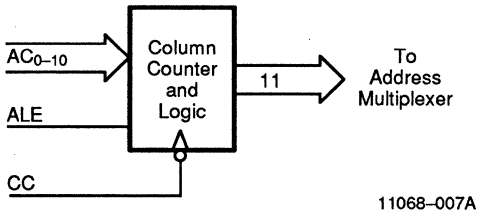


Figure 3. Column Counter and Logic

If the Nibble Count bit of the Configuration Register is enabled (NIBCNT = 1), then only the two LSBs of the Column Latch are clocked, generating a modulo four nibble count (Reference Nibble Mode section).

* Modes of operation are described beginning on page 22.

Bank Logic

This block (Figure 4) contains the Bank Latch, Bank Register, and Bank Comparator. The 2-bit Bank Latch holds the DRAM bank address. The latch is transparent when ALE is HIGH and latches the address on the LOW-going edge of ALE.

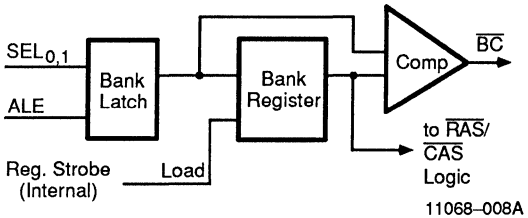


Figure 4. Bank Logic

The 2-bit Bank Register holds the bank address of the previous DRAM access. The register is clocked at the beginning of every access by the HIGH-going edge of RASI in Read/Write mode only.

The 2-bit Bank Comparator compares the bank address of the current access (contents of the Bank Latch) and the bank address of the previous access (contents of the Bank Register) and generates the $\overline{BC}$ signal. $\overline{BC}$ is LOW if the current bank address is the same as the previous bank address. $\overline{BC}$ is used by the external timing generator during bank interleaving to either activate the RASI input if two consecutive accesses are to two different banks (saving $\overline{RAS}$ precharge time on the current $\overline{RAS}_n$) or to delay the RASI input if two consecutive accesses are to the same bank (so that the current $\overline{RAS}_n$ output can go through precharge before it is reactivated).

Programmable Registers and Logic

This block (Figure 5) consists of the Configuration Register, Burst Count Register, Mask Register, Column Comparator Logic, Register Load Logic, and DRAM Size Decoder.

In order to load the 11-bit Configuration Register, a device reset ($MC_{0,1}$=11 with RASI ↓) must occur followed by switching MC_0, or MC_1 to 0 (to end reset operation). Then the Configuration Register is loaded via the column address bus (AC_{0-10}) by the High-going edge of RL/CC signal (↓) with $MC_{0,1}$=11. The Configuration Register is programmed to select the options shown in Figure 6 (reference Figure 15).

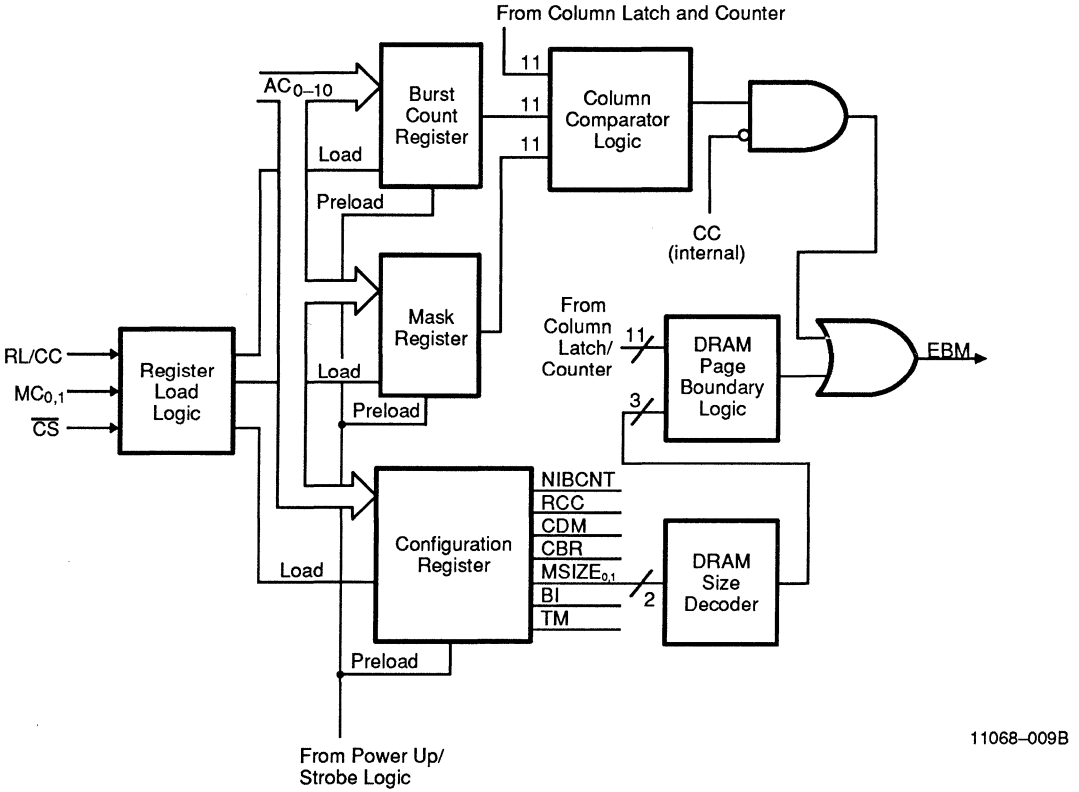


Figure 5. Programmable Registers and Logic

The 11-bit Burst Count Register is loaded via AC_{0-10} by the HIGH-going edge of RL through the Register Load Logic. This register is preloaded with all 1's (for a maximum burst count) in the Reset mode after power-up, and is only used in the Burst/Block mode of access, if a programmed number of accesses is required. This register is loaded with the maximum number of transfers to occur during any burst/block access. This number is dependent on the specifics of the system (i.e... page size or processor type).

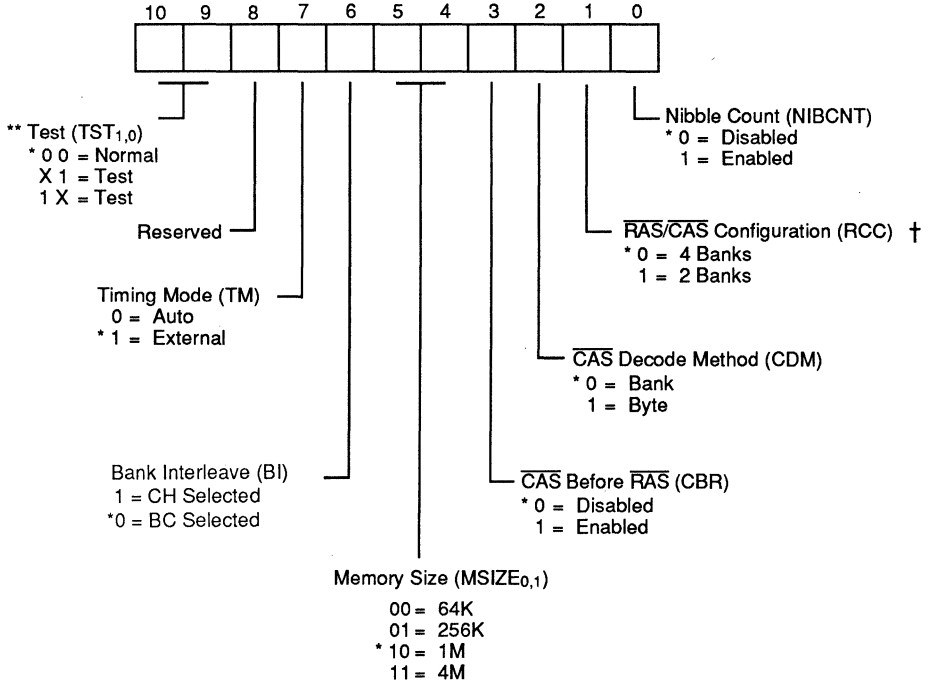
The 11-bit Mask Register is loaded via AC_{0-10} by the HIGH-going edge of the RL signal through the Register Load Logic. This register is preloaded with all 1's (for all bits to be compared) in the Reset Mode after power-up, and is only used in the Burst/Block mode of access, if a programmed number of accesses is required.

This register allows a burst to be made which is shorter than the page size of the memory. When the Mask Register is loaded with a "1" in a bit location, the corresponding bit in the Column Latch and the Burst Count Register is compared. This register also aligns the Column Latch and Counter for succeeding bursts of full length if they occur immediately after termination of the prior burst. The Mask Register is loaded with 00000001111 for a 16-bit maximum burst.

The Column Comparator Logic compares the contents of the Column Latch and Counter with that of the Burst Count Register (which contains the end of burst count value). The HIGH bits in the Mask Register determine which of the 11-bits of the Column Latch and Counter and the Burst Count Register are compared. This logic is used only in the Burst/Block mode of access. Reference Figure 6a.

The Register Load Logic loads the Burst Count, Mask and Configuration Registers via the address bus (AC_{0-10}) dependent upon the state of the Register Loading diagram in Figure 7. RL/CC Decoder and Register Load Logic are shown in Figure 8.

The DRAM Size Decoder determines the DRAM size being used. The $MSIZE_{1,0}$ bits in the Configuration Register are used to decode the size of the DRAMs being used as shown in Figure 6.



* Default. The Am29C668 will power up in Am29368 mode if the user does not reprogram the configuration register.

† Reference Table 4.

** These bits are used during factory testing only.

11068–010A

Figure 6. Configuration Register Options

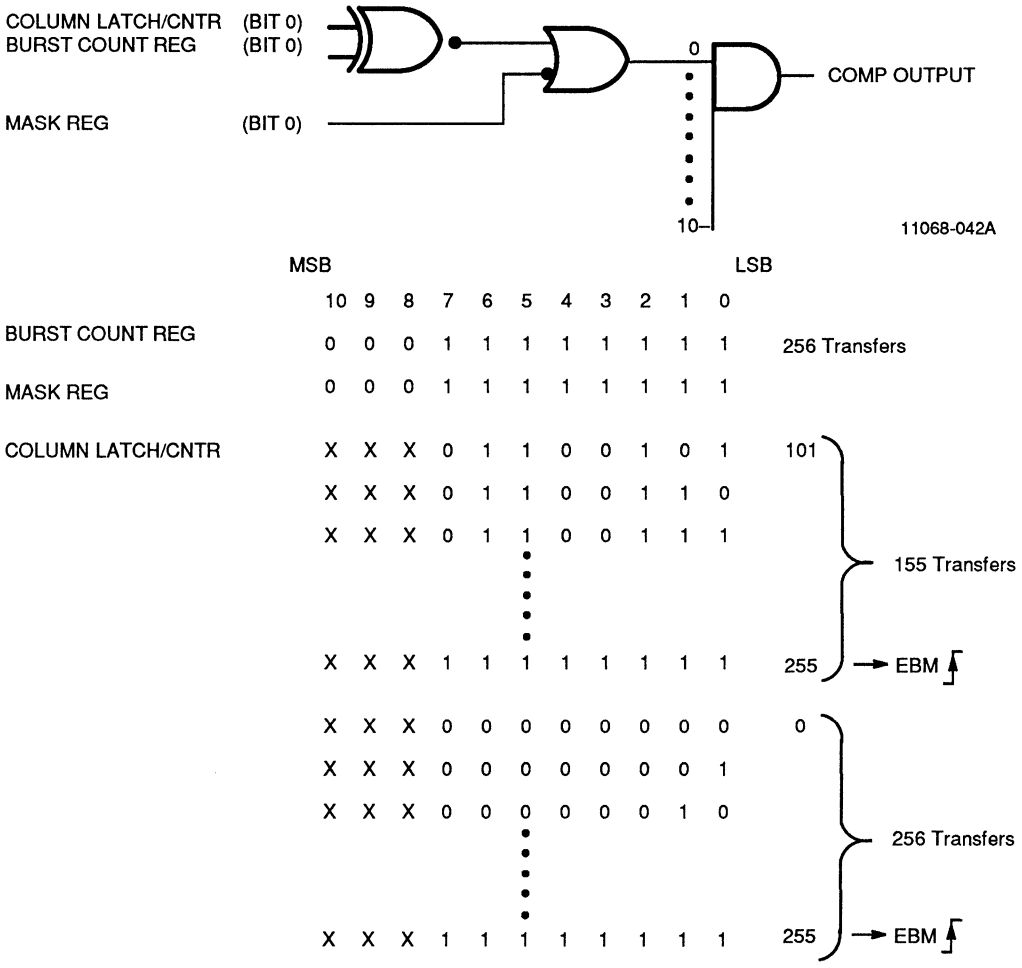


Figure 6a. Programmable Burst Logic with 256-word Burst Length Example

In this example, the self alignment feature of the Am29C668 is shown. The first burst is terminated on the DRAM page boundary by the EBM output. All subsequent bursts are then set at 256 transfers, which has been programmed via the Burst Count and Mask registers.

Auto/External Timing Logic

When Auto Timing mode is selected via the Timing Mode (TM) bit in the Configuration Register (TM = 0), this circuit generates internal timing delays between RASI-MSEL and MSEL-CASI. These delays are optimized for use with 100ns DRAMs.

In the Auto Timing mode the CASI/CASIEN input acts as CASIEN. In this mode internal $\overline{CAS}$ is generated from the active (HIGH) edge of RASI and is deactivated when RASI goes inactive by the Auto Timing Circuit. This internally generated $\overline{CAS}$ is gated with the CASIEN input to generate the $\overline{CAS}_n$ outputs. This gating circuit allows the Auto-Timing to be externally overridden (Figure 9). It is used for specialty DRAM accesses.

In the External Timing mode (TM = 1), the internal $\overline{CAS}$ signal follows the externally generated CASI input.

In the Auto Timing Mode the MSEL/MSELEN input acts as MSELEN. In this mode, internal MSEL is generated from the active (HIGH) edge of RASI, and is deactivated when RASI goes inactive, by the Auto Timing Circuit. This internally generated MSEL is gated with the MSELEN input to generate the internal MSEL signal. This feature is used to extend row address hold time via external control (overriding the Auto Timing feature).

In the External Timing mode (TM = 1), the internal MSEL signal follows the externally generated MSEL input.

The Auto Timing mode allows 4 banks of 16-bit data plus 6 EDC check bits or 2 banks of 32-bit data plus 7 EDC check bits comprised of 100ns DRAMs to be operated without external drivers.

Power-up/Strobe Logic

This block automatically presets the Am29C668 to the default condition upon power-up (Figure 6). This circuit also generates all the internal control signals for the Refresh Counter, Configuration, Burst Count, and Mask Registers, the Register Load Logic, and the Bank Register.

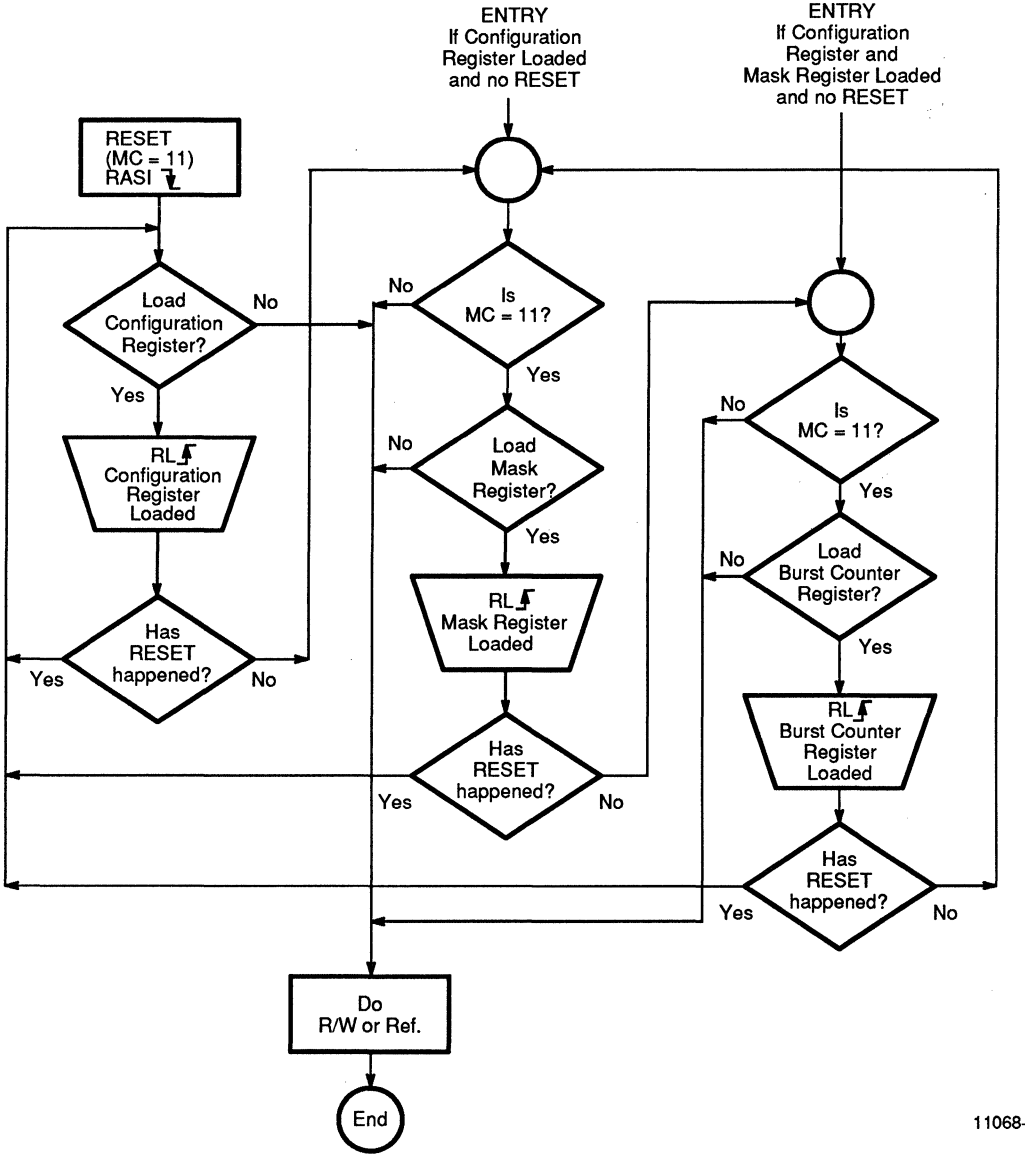


Figure 7. Register Loading (See paragraph on page 20)

Figure 7. Register Loading. The Configuration Register must be loaded before the Mask and Burst Count Registers may be loaded. Once the Configuration Register is loaded, the Register Load Logic will toggle between loading the Mask Register and Burst Count Register. The Configuration Register may only be loaded immediately after a reset. The Mask Register and Burst Count Register are only used in the Burst/Block access mode, in other modes only the Configuration Register need be loaded if the user wishes to alter its default mode indicated in Figure 6.

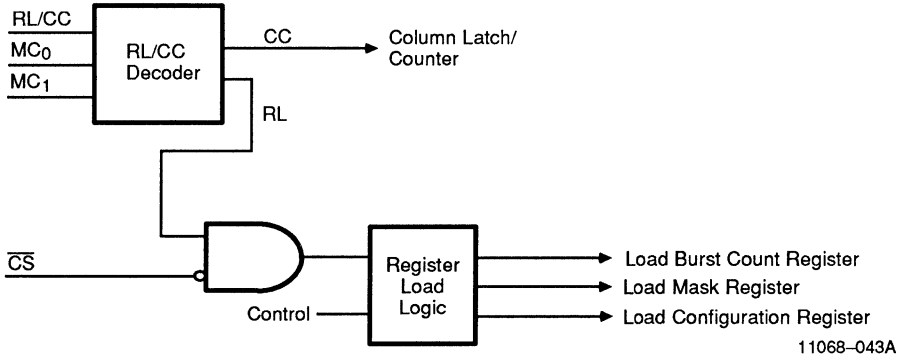


Figure 8. Register Load Logic and RL/CC Decoder (Register Logic)

DRAM Page Boundary Logic

This logic block indicates to the processor when a page boundary on the DRAM is reached. It monitors the contents of the Column Latch/Counter and, depending upon the outputs of the DRAM Size Decoder, signals an End of Burst/Block Mode (EBM) when a page boundary is reached. A page boundary condition is reached when the contents of the Column Latch/Counter equals the DRAM page boundary address.

This logic is used only in the Burst/Block mode of access.

Refresh/Scrubbing/Bank Counters

This block (Figure 10) contains the 11-bit Row Refresh Counter, 11-bit Column Refresh Counter, and 2-bit Bank Refresh Counter. All three counters are synchronous and are reset when $MC_{0,1}$ = 11 and RASI transitions from LOW to HIGH. These counters are clocked when $MC_{0,1}$ = 00 or 10 and RASI transitions from HIGH to LOW.

The size of the Row and Column Refresh Counters are automatically adjusted for the DRAM size being used. This is done by selecting the proper Row Counter output to go to the low order Column Counter input and similarly selecting the proper Column Counter output to go to the low order Bank Counter input. This selection is determined by the outputs of the DRAM Size Decoder with the help of a multiplexer.

The $\overline{RAS}/\overline{CAS}$ Configuration Bit of the Configuration Register selects which bit of the Bank Counter is used for the Terminal Count (TC) output, depending upon whether 2 or 4 banks of DRAM are used.

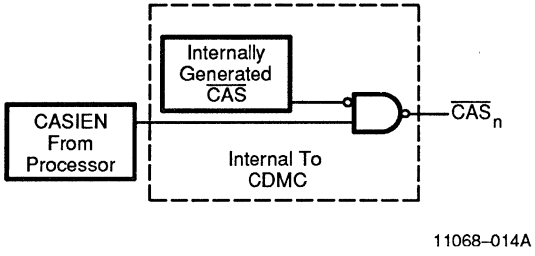


Figure 9. Auto Timing With External Override

Address Multiplexer

This block is an 11-bit, four input multiplexer which selects the address to the DRAMs. Its four address inputs are the row latch output, column latch/counter output, row refresh counter output, and column refresh counter output. The $MC_{0,1}$, internal MSEL, and $\overline{CS}$ input signals are decoded to select one of the four addresses.

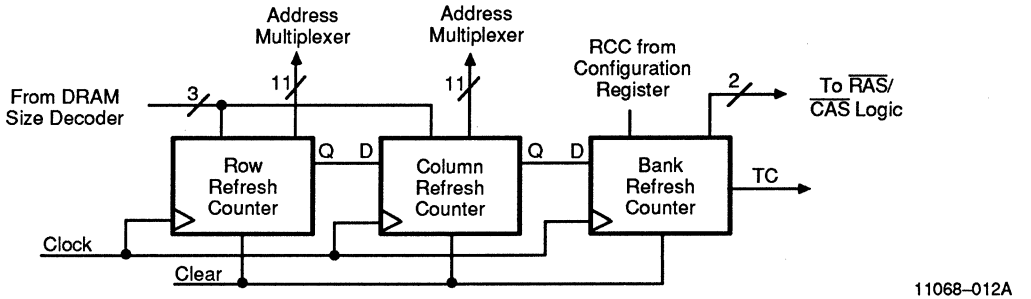


Figure 10. Refresh/Scrubbing/Bank Counters

RAS/CAS Logic

This block (Figure 11) contains the CAS Before RAS Logic, RAS and CAS Decode Logic, CAS Enable Logic, and CAS Multiplexer.

The CAS Before RAS Logic switches the RAS and CAS lines to the RAS and CAS Decode Logic if the CBR selection bit in the Configuration Register is set (1). This allows a CAS Before RAS refresh to be accomplished without altering the order of the RASI and CASI input strobes. Refresh With Scrubbing ($MC_{0,1}$ = 10) is not allowed when the CBR bit is set (1).

The RAS and CAS Decode Logic decodes the internal RAS and CAS timing signals to generate the four $\overline{RAS}_n$ and four $\overline{CAS}_n$ output signals.

The CAS Enable Logic is used if a byte-wise CAS decoding method is selected. Byte enables are decoded externally and are connected to the $\overline{CASEN}_{0-3}$ inputs. In the CAS Enable Logic all the $\overline{CASEN}_{0-3}$ signals are individually gated with the internal CAS signal to generate the $\overline{CAS}_n$ proper outputs.

The CAS Multiplexer is a 4-bit, two input multiplexer. It selects one set of $\overline{CAS}_n$ signals to the output depending upon the CAS decode method being used (selected by the CDM bit in the Configuration Register) and the operating mode (selected by $MC_{0,1}$).

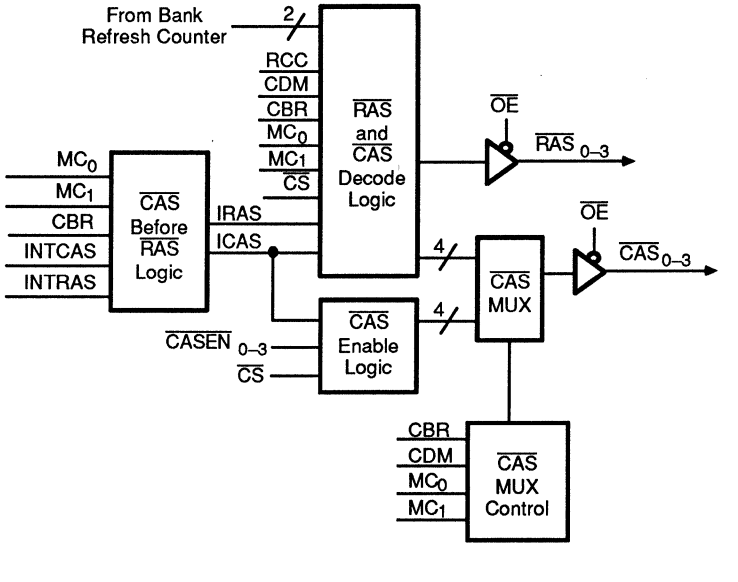


Figure 11. RAS/CAS Logic

MODES OF OPERATION

The Am29C668 has two basic modes of operation, read/write and refresh.

Read/Write Modes

In the read/write mode the Am29C668 latches the row, column and bank addresses and multiplexes them to the DRAM array under the control of the internally generated timing strobes in the Auto Timing Mode or the externally generated MSEL in the External Timing Mode. The timing option is selected via the Timing Mode (TM) bit in the Configuration Register (Figure 6).

The row address is latched in the DRAMs by the active (LOW-going) edge of the $\overline{RAS}_n$ output, which follows the active (HIGH-going) edge of the RASI input. The address lines are then switched to column address by either an internally generated signal in the Auto Timing Mode or by pulling MSEL active HIGH in the External Timing Mode. The column address is latched in the DRAMs on the active (LOW-going) edge of the $\overline{CAS}_n$ output, which follows either an internally generated signal in the Auto Timing Mode or the active (HIGH-going) edge of the CASI input in the External Timing Mode.

The read/write mode of the Am29C668 may be optimized for the shortest memory cycle time, through burst/block accesses, nibble mode accesses, "cache" mode accesses, or bank interleaving.

Burst/Block Mode

When a burst/block access is requested by the processor, the Am29C668 latches the initial row, column, and bank addresses. Subsequent column addresses are generated internally by the Am29C668, allowing consecutive memory locations to be accessed at high speed without the processor actually generating each memory location address. This type of transfer can be used by high performance processors to fill their on-chip or external cache when a cache miss is encountered.

During a burst access the CC input of the Am29C668 is toggled after the initial row, column, and bank addresses have been latched. While the RASI input is held high by the processor, each high-to-low transition of CC increments the column address for the next memory access.

The burst access will continue until a programmed number of accesses (which is stored in the Burst Count Register) has been completed or a page boundary is reached. Both conditions are indicated to the processor by the EBM output.

Nibble Mode

For Nibble mode accesses the Nibble Count bit (NIBCNT) in the Configuration Register is set to "1". This bit enables only the two least significant bits of the Column Latch and Counter to be clocked, allowing the Column Latch and Counter to perform a modulo four count when making a nibble burst access (as in the case of the 68030 processor) using non-nibble DRAMs.

When "nibble" DRAMs are used nibble accesses are accomplished by toggling CASI (Reference Figure 25).

Cache Mode

This mode allows the efficient use of page mode and fast page mode DRAMs by comparing back-to-back row and bank addresses from the processor.

In the "cache" mode of access of the Am29C668 the $\overline{RAS}_n$ output is held active (LOW) and any location in that row is accessed by only changing the column address. This makes the entire row look like a cache, since any access in that row can be made at high speed. To select the cache access mode, the Bank Interleave (BI) bit in the Configuration Register is set to "1". The row and bank addresses of consecutive accesses are compared by the Am29C668. If the row and bank addresses of consecutive accesses match, $\overline{CH}$ goes active (LOW) and signals the timing generator not to deactivate the RASI input but only to toggle the CASI/CASIEN input. If the row and bank addresses of consecutive accesses do not match, the $\overline{CH}$ signal goes inactive (HIGH) and informs the timing generator to deactivate the RASI input and start a new RASI cycle after the current cycle has gone through a $\overline{RAS}$ precharge cycle. When the RASI input is activated, its HIGH-going edge loads the row and bank registers with the contents of the row and bank latches, respectively, saving the current values for the next comparison.

Bank Interleave Mode

The Am29C668 can be configured to support on-chip bank interleaving by connecting the two LSBs of the processor address to $SEL_{0,1}$ and resetting the Bank Interleave (BI) bit in the Configuration Register to "0". Accesses being made to consecutive locations will be in adjacent banks, allowing the $\overline{RAS}_n$ strobe for the new bank to be activated as soon as the $\overline{RAS}_n$ strobe for the previous bank is deactivated and is precharging. This reduces the memory cycle time and improves memory throughput. The Bank Compare ($\overline{BC}$) signal indicates to the external timing generator whether the current access is to a different bank than the previous access and therefore whether bank interleaving is possible. The $\overline{BC}$ signal goes active (LOW) when the present access is to the same bank as the previous access.

The Bank Interleave Mode may not be used in conjunction with the Burst/Block, Nibble, or Cache Modes. The number of memory banks is set with the RCC bit in the Configuration Register.

Refresh Modes

Normal Refresh

In the normal refresh mode, the refresh address is generated by the Am29C668 refresh counter. The row refresh counter is used to generate the row address. All corresponding rows in all four banks are refreshed simultaneously by generating all four $\overline{RAS}_n$ outputs in response to the RASI input. Hence, the entire memory may be refreshed by stepping through the row refresh counter once. The row refresh counter is incremented to the next refresh address by the inactive (LOW going) edge of the RASI input.

Refresh With Scrubbing

When memory scrubbing is performed in systems employing error detection and correction (EDC), the row, column, and bank refresh counters are used. In this case, all four corresponding rows are refreshed and one location of one row is "scrubbed", ie...a read/modify/write cycle is performed. An entire memory array can be scrubbed by stepping through the row, column, and bank address counters once. The Am29C668 has four independent $\overline{CAS}_n$ outputs allowing a single bit to be accessed during refresh cycles.

$\overline{CAS}$ Before $\overline{RAS}$ Refresh

This is a feature of some dynamic RAMs. The DRAM on-chip refresh counter is updated and a refresh cycle performed by generating a $\overline{CAS}$ strobe before the $\overline{RAS}$ strobe. This refresh support is selected by enabling (setting to "1") the $\overline{CAS}$ Before $\overline{RAS}$ bit in the Configuration Register. Refresh with scrubbing ($MC_{0,1}$ = 10) is not allowed when the CBR bit is set (1).

In this mode, RASI controls the $\overline{CAS}_n$ outputs and CASI controls the $\overline{RAS}_n$ outputs. This allows a $\overline{CAS}$ before $\overline{RAS}$ refresh to be accomplished without altering the order of the RASI and CASI input strobes. When this mode is set, memory "scrubbing" ($MC_{0,1}$ = 10) is prohibited.

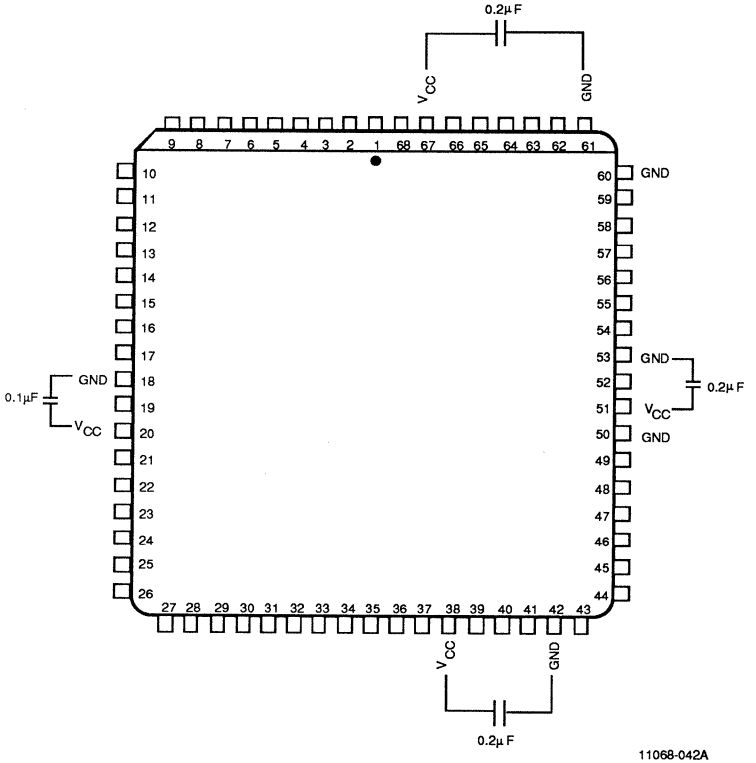


Figure 12. Device Decoupling – V_{CC} and Ground Pin Connections

Note:

Due to the high switching speeds and high drive capability of the Am29C668, it is necessary to decouple the device for proper operation. Multilayer ceramic capacitors are recommended. It is important to mount the capacitors as close as possible to the power pins (V_{CC}, GND) to minimize lead inductance and noise. A ground plane is strongly recommended. A wire wrapped board without power and ground planes is not recommended.

It is strongly recommended that this part be directly surface mounted whenever possible. Should a PLCC, or PGA socket be required, a one-time-insertion-only socket with minimal lead length is necessary for proper device function. The socket lead inductance should be 8nH maximum per pin.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature	-65 to +150°C
Ambient Operating Temperature	-55 to +125°C
Maximum V_{CC}	-0.5 to +7.0 V
DC Voltage Applied to Any Pin	-0.5 to V_{CC} +0.3 V

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+4.50 to +5.50 V

Military (M)

Case Temperature (T_C)	-55 to +125°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over COMMERCIAL operating range unless otherwise specified (for APL Products, Group A, Subgroups 1,2,3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions (Note 1)		Min	Max	Unit
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for all Inputs (Note 2)		2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for all Inputs (Note 2)			0.8	V
I_{IH}	Input HIGH Current	V_{CC} = Max., V_{IN} = V_{CC}			5.0	µA
I_{IL}	Input LOW Current	V_{CC} = Max., V_{IN} = GND			–5.0	µA
V_{OH}	Output HIGH Voltage	V_{CC} = Min. I_{OH} = –10 mA		2.7		V
V_{OL}	Output LOW Voltage	V_{CC} = Min., I_{OL} = 10 mA			0.5	V
V_{ON}	Output Undershoot ↓ Voltage (Note 4)	CL = 50pF			–1.0	V
V_{OP}	Output Overshoot ↓ Voltage (Note 4)	CL = 50pF			0.8	V
V_{OP}	Output Overshoot ↑ Voltage (Note 4)	CL = 50pF			0.8	V
I_{OZ}	Off–State (High Impedence) Output Current; Q Outputs	V_{CC} = Max.	V_O = 0V		–10	µA
			V_O = V_{CC} (Max)		10	
I_{CCQ}	Quiescent Power Supply Current (CMOS Inputs)	V_{CC} = Max. 4.3V ≤ V_{IN} V_{IN} ≤ 0.2V f_{OP} = 0			5.0	mA
I_{CCT}	Quiescent Input Power Supply Current (@ TTL HIGH)	V_{CC} = Max. V_{IN} = 2.4 f_{OP} = 0			25	mA
I_{CCD}	Dynamic Power Supply Current (Note 5)	V_{CC} = Max. 4.3 V ≤ V_{IN} , V_{IN} ≤ 0.2V C_L = 150pF $\overline{OE}$ = LOW	MIL		7	mA/ MHz
			COM'L		7	
I_{CC}	Total Power Supply Current (Notes 3 and 5)	V_{CC} = Max., f_{OP} = 10 MHz $\overline{OE}$ = LOW 50% Duty cycle, C_L = 150pF 4.3 V ≤ V_{IN} , V_{IN} ≤ 0.2V	MIL		100	mA
			COM'L		100	
		V_{CC} = Max., f_{OP} = 10 MHz $\overline{OE}$ = LOW 50% Duty cycle, C_L = 150pF V_{IN} = 3.4, V_{IL} = 0.4 V	MIL		100	
			COM'L		100	

Notes:

1. For conditions shown as Min. or Max., use appropriate value specified under Operating Range for the applicable device type.
2. Tested with limited test pattern.
3. Total Power Supply Current is the sum of the Quiescent Current and the dynamic current (at either CMOS or TTL input levels). For all conditions, the Total Power Supply Current can be calculated by using the following equation:

   $I_{CC} = I_{CCQ} + I_{CCD}(f_{op})$ (CMOS Inputs), $I_{CC} = I_{CCT} + I_{CCD}(f_{op})$ (TTL Inputs), f_{op} = Operating Frequency in Megahertz

   During device characterization, two addresses, one $\overline{RAS}_n$ and one $\overline{CAS}_n$ output were toggled at f_{OP} = 10 MHz during I_{CC} measurement.
4. V_{ON} and V_{OP} are not production tested but are guaranteed by characterization data for surface mounted devices with proper capacitive decoupling. Limits specified are for all outputs switching simultaneously with minimum specified loading. As loading increases, V_{ON} and V_{OP} will approach zero. Reference Switching Waveforms.
5. Not tested in production. Guaranteed by characterization data.

PRELIMINARY

SWITCHING CHARACTERISTICS over operating range unless otherwise specified

Capacitive Loading = 350pF for all Q_n, $\overline{RAS}_n$, and $\overline{CAS}_n$; 100pF for all other parameters (minimum tester load)

No.	Parameter Symbol	Parameter Description	Test Conditions	Commercial				Military		Units
				29C668		29C668-1		29C668		
				Min	Max	Min	Max	Min	Max	
COMMON PARAMETERS										
1	t_{PD}	AC_n/AR_n to Q_n	ALE = 1	4	33	4	29			ns
2	t_{PD}	MC_n to Q_n		6	37	6	34			ns
3	t_{PD}	ALE to Q_n		5	34	5	29			ns
4	t_{PD}	$\overline{CS}$ to Q_n		4	34	4	34			ns
5	t_S	AC_n/AR_n to ALE ↓ Set Up Time		4		4				ns
6	t_H	AC_n/AR_n to ALE ↓ Hold Time		4		4				ns
7	t_S	SEL_n to ALE ↓ Set Up Time		4		4				ns
8	t_H	SEL_n to ALE ↓ Hold Time		4		4				ns
9	t_S	MC_n to RASI ↑ Set Up Time		0		0				ns
10	t_H	MC_n to RASI ↓ Hold Time		4		4				ns
11	t_{PD}	$\overline{CS}$ ↓ to RASI ↑		4		4				ns
12	t_{PD}	$\overline{CS}$ ↑ to RASI ↓		4		4				ns
13	t_S	SEL_n to RASI ↑ Set Up Time	ALE = 1	4		4				ns
14	t_H	SEL_n to RASI ↑ Hold Time	ALE = 1	4		4				ns
15	t_{PWL}	RASI, CASI, CASIEN Pulse Width LOW	Note 2	10		10				ns
16	t_{PWH}	RASI, CASI, CASIEN PulseWidth HIGH	Note 2	10		10				ns
EXTERNAL TIMING MODE										
17	t_{PD}	RASI to $\overline{RAS}_n$		2	30	2	26			ns
18	t_{PD}	CASI to $\overline{CAS}_n$		3	30	3	26			ns
19	t_{PD}	MSEL to Q_n		4	30	4	26			ns
AUTO TIMING MODE										
20	t_{PD}	RASI to $\overline{RAS}_n$			30		26			ns
21	t_{PD}	$\overline{RAS}_n$ to Q_n (row address)	MSELEN = 1	15		15				ns
22	t_{PD}	RASI to $\overline{CAS}_n$	CASIEN = 1		88		88			ns
23	t_{PD}	$\overline{RAS}_n$ to $\overline{CAS}_n$	CASIEN = 1	25	66	25	66			ns
24	t_{PD}	Q_n to $\overline{CAS}_n$ (column address)	MSELEN = 1 CASIEN = 1	4		4				ns
25	t_{PD}	MSELEN to Q_n (column address)			33		31			ns
26	t_{PD}	CASIEN to $\overline{CAS}_n$			30		26			ns

SWITCHING CHARACTERISTICS (Continued)

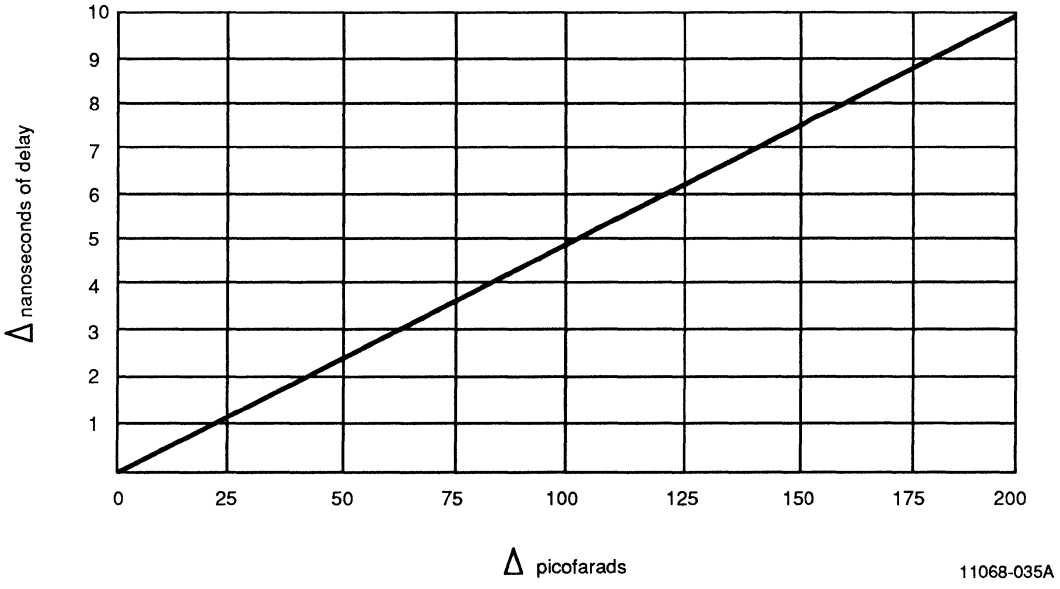
Capacitive Loading = 350pF for all Q_n, $\overline{RAS}_n$, and $\overline{CAS}_n$; 100pF for all other parameters (minimum tester load)

No.	Parameter Symbol	Parameter Description	Test Conditions	Commercial				Military		Units
				29C668		29C668-1		29C668		
				Min	Max	Min	Max	Min	Max	
SPECIALTY MODES										
27	t_{PD}	CC↓ to Q_n			33		30			ns
28	t_{PD}	CC↓ to EBM			18		17			ns
29	t_{PW}	CC Pulse Width LOW or HIGH	Note 2	10		10				ns
30	t_{PD}	SEL_n to $\overline{BC}$	ALE = 1		14		12			ns
31	t_{PD}	ALE to $\overline{BC}$			14		13			ns
32	t_{PD}	AR_n to $\overline{CH}$	ALE = 1		17		16			ns
33	t_{PD}	SEL_n to $\overline{CH}$	ALE = 1		14		13			ns
34	t_{PD}	ALE to $\overline{CH}$			18		16			ns
35	t_{PD}	RASI↓ to TC			19		19			ns
36	t_S	AC_n/AR_n to RL↑ Set Up Time		0		0				ns
37	t_H	AC_n/AR_n to RL↑ Hold Time		10		10				ns
38	t_S	MC_n to RL↑ Set Up Time		5		5				ns
39	t_H	MC_n to RL↓ Hold Time		2		2				ns
40	t_{PW}	RL Pulse Width LOW or HIGH	Note 2	10		10				ns
41	t_{PD}	$\overline{CASEN}_n$ to $\overline{CAS}_n$			35		31			ns
OUTPUT SKEWS										
42	t_{SKEW}	$\{t_{PD}(AC_n/AR_n$ to $Q_n) - t_{PD}$(RASI to $\overline{RAS}_n)\}$	MC_n = 01		23		20			ns
43	t_{SKEW}	$\{t_{PD}(MC_n$ to $Q_n) - t_{PD}$(RASI to $\overline{RAS}_n)\}$	MC_n = 00,01		23		21			ns
44	t_{SKEW}	$\{t_{PD}$(MSEL to $Q_n - t_{PD}$(RASI to $\overline{RAS}_n)\}$	Note 3		18		16			ns
45	t_{SKEW}	$\{t_{PD}$(MSEL to $Q_n - t_{PD}$(CASI to $\overline{CAS}_n)\}$			16		16			ns
THREE STATE OUTPUTS										
46	t_{PLZ}	Output Disable Time from LOW	Note 2		20		20			ns
47	t_{PHZ}	Output Disable Time from HIGH			25		25			ns
48	t_{PZL}	Output Enable Time from LOW			27		27			ns
49	t_{PZH}	Output Enable Time from HIGH			26		26			ns

Notes:

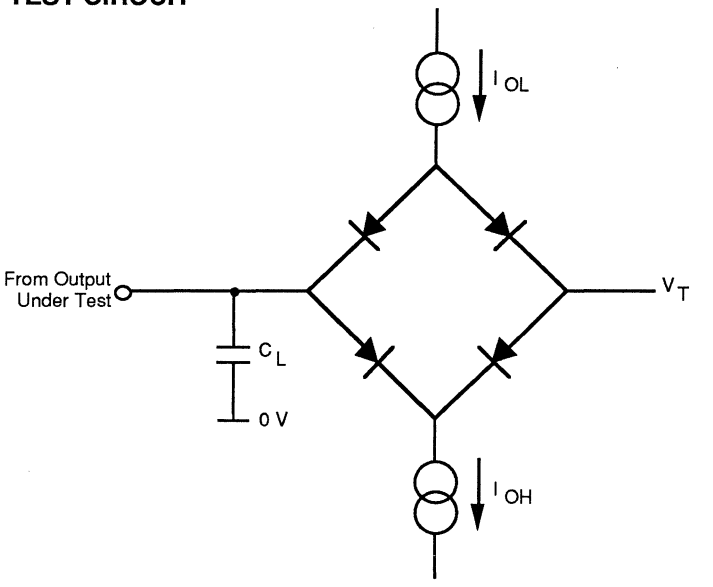
1. For additional loading information or to calculate t_{PD} between specified loads see section labeled "TYPICAL Change in Propagation Delay vs. Loading Capacitance" following the Switching Waveforms.
2. Not included in Group A testing. Not production tested. Guaranteed by characterization data.
3. Worst case for any given device.

TYPICAL Change in Propagation Delay vs Loading Capacitance



11068-035A

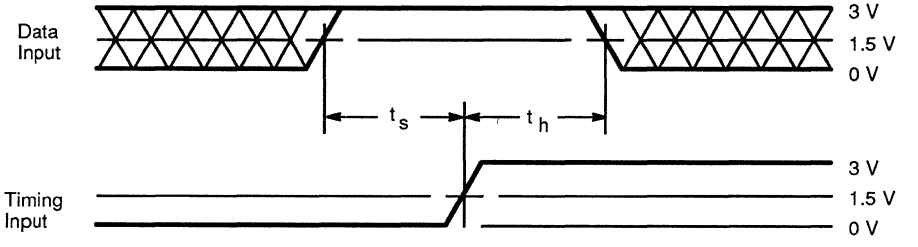
SWITCHING TEST CIRCUIT



11068–036A

Notes: 1. C_L is specified in Switching Characteristics Table.
2. V_T = 1.5V.

SWITCHING TEST WAVEFORMS



11068–037A

Notes: 1. Diagram shown for HIGH data only. Output transition may be opposite sense.
2. Cross–hatched area is don't care condition.

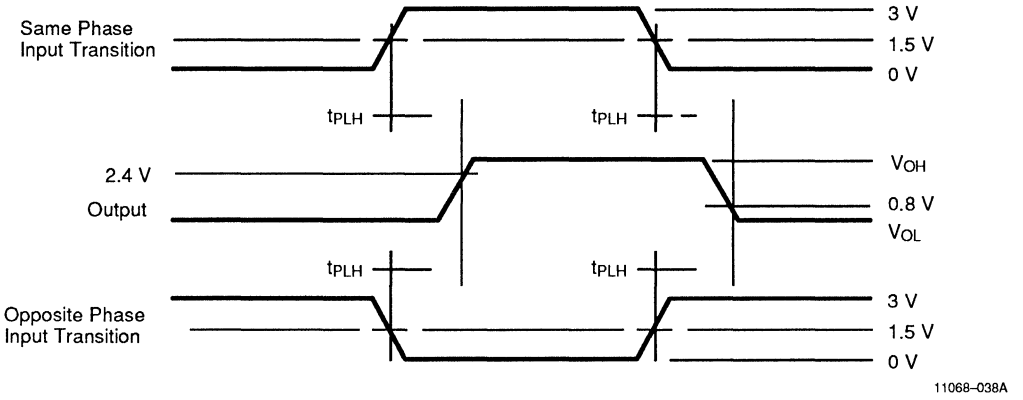
Setup and Hold Times

Notes on Testing

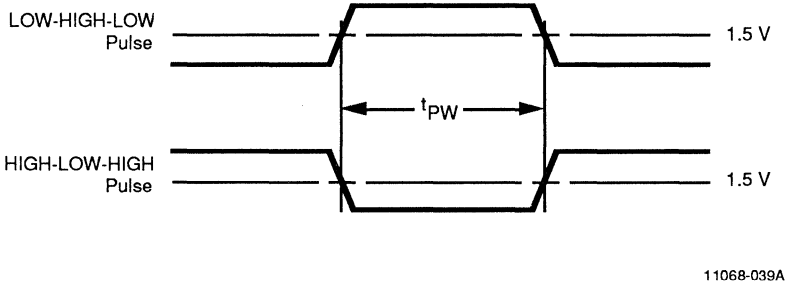
Incoming test procedures on this device should be carefully planned, taking into account the complexity and power levels of the part. The following notes may be useful.

1. Ensure the part is adequately decoupled at the test head. Large changes in V_{CC} current as the device switches may cause erroneous function failures due to V_{CC} changes.
2. Do not leave inputs floating during any tests, as they may start to oscillate at high frequency.
3. Do not attempt to perform threshold tests at high speed. Following an input transition, ground current may change by as much as 400 mA in 5-8 ns. Inductance in the ground cable may allow the ground pin at the device to rise by hundreds of millivolts momentarily.
4. Use extreme care in defining input levels for AC tests. Many inputs may be changed at once, so there will be significant noise at the device pins and they may not actually reach V_{IL} or V_{IH} until the noise has settled. AMD recommends using $V_{IL} \leq 0$ V and $V_{IH} \geq 3$ V for AC tests.
5. To simplify failure analysis, programs should be designed to perform DC, Function, and AC tests as three distinct groups of tests.
6. Proper device grounding is critical when device testing. Multi-layer performance boards with radial decoupling between power and ground planes is recommended. Wiring unused interconnect pins to the ground plane is recommended. The ground plane must be sustained from the performance board to the device under test interface board. To minimize inductance, heavy-gauge stranded wire with twisted pairs should be used for power wiring.

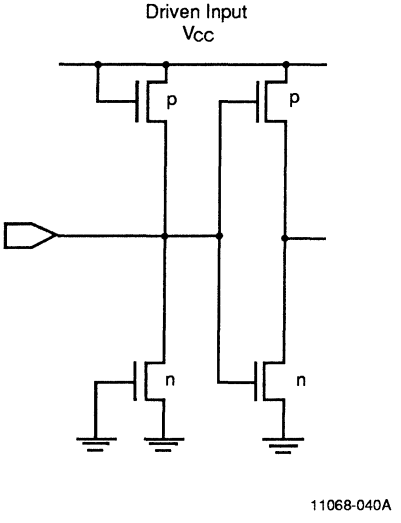
SWITCHING WAVEFORMS (Continued)



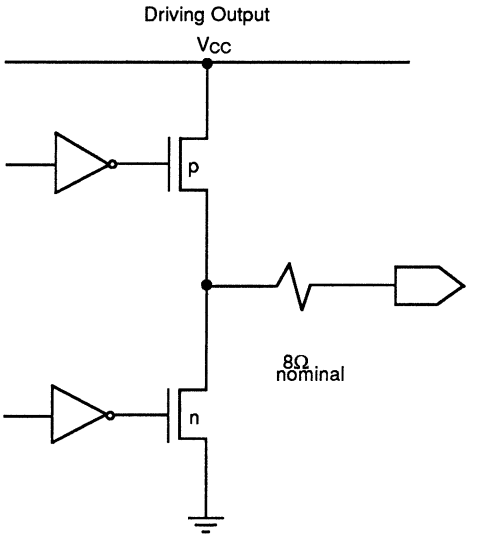
Propagation Delay



Input Pulse Width

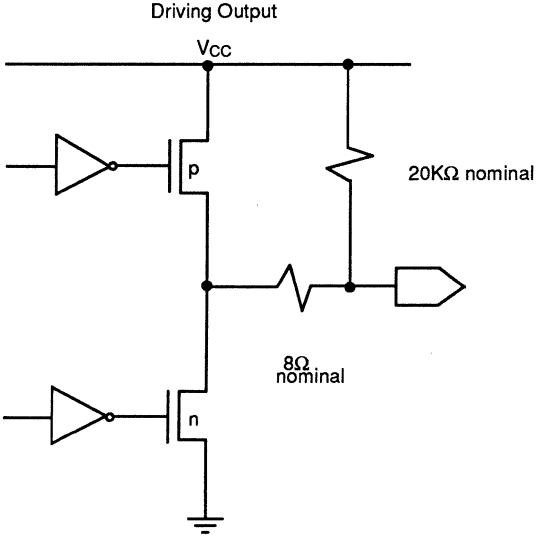


Equivalent Input Circuit



11068-044A

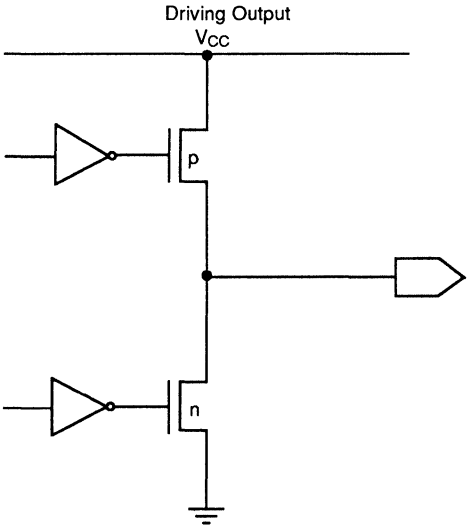
Equivalent Output Circuit (Q_n Outputs)



11068-045A

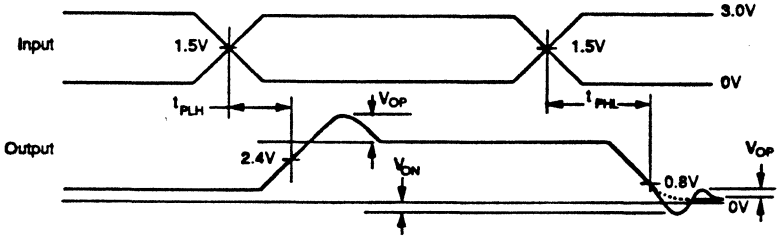
Equivalent Output Circuit ($\overline{RAS}_n$, $\overline{CAS}_n$ Outputs)

SWITCHING WAVEFORMS (Continued)



11068-046A

Equivalent Output Circuit
(All outputs except Q_n, $\overline{RAS}_n$, $\overline{CAS}_n$)



11068–041A

Output Drivers Level

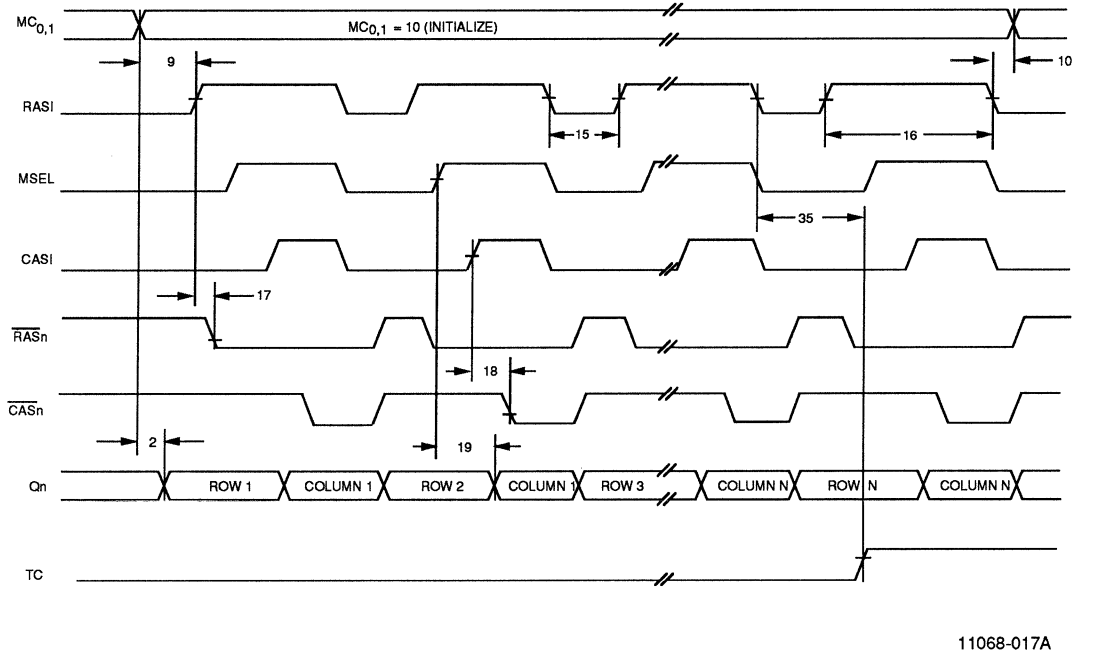


Figure 13. EDC Initialization with External Timing

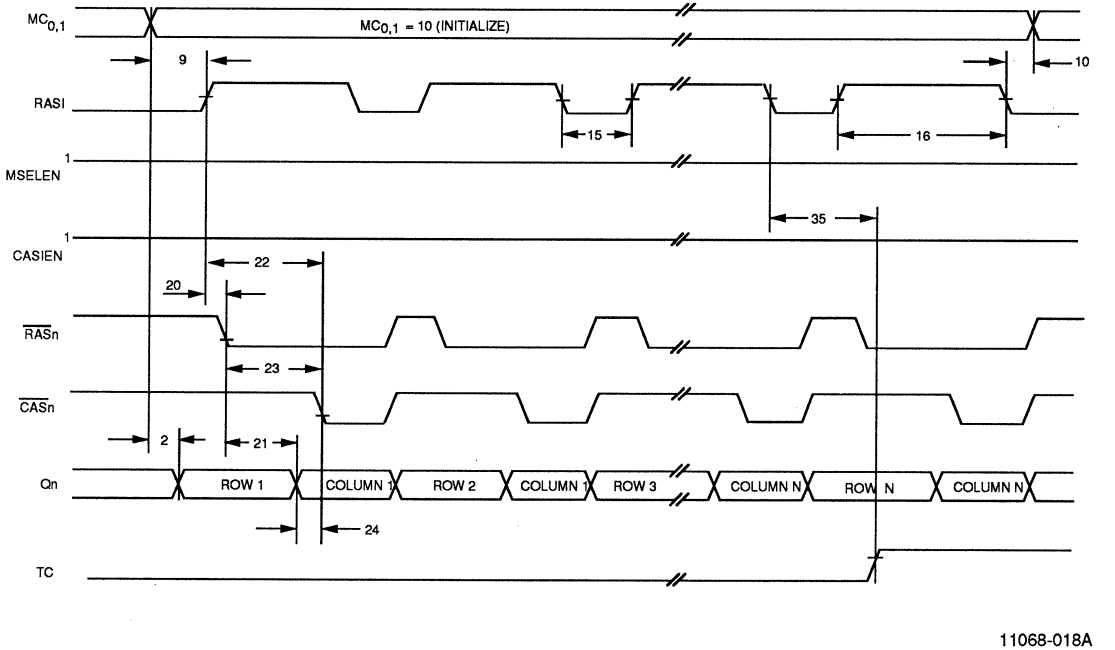


Figure 14. EDC Initialization with Auto-Timing

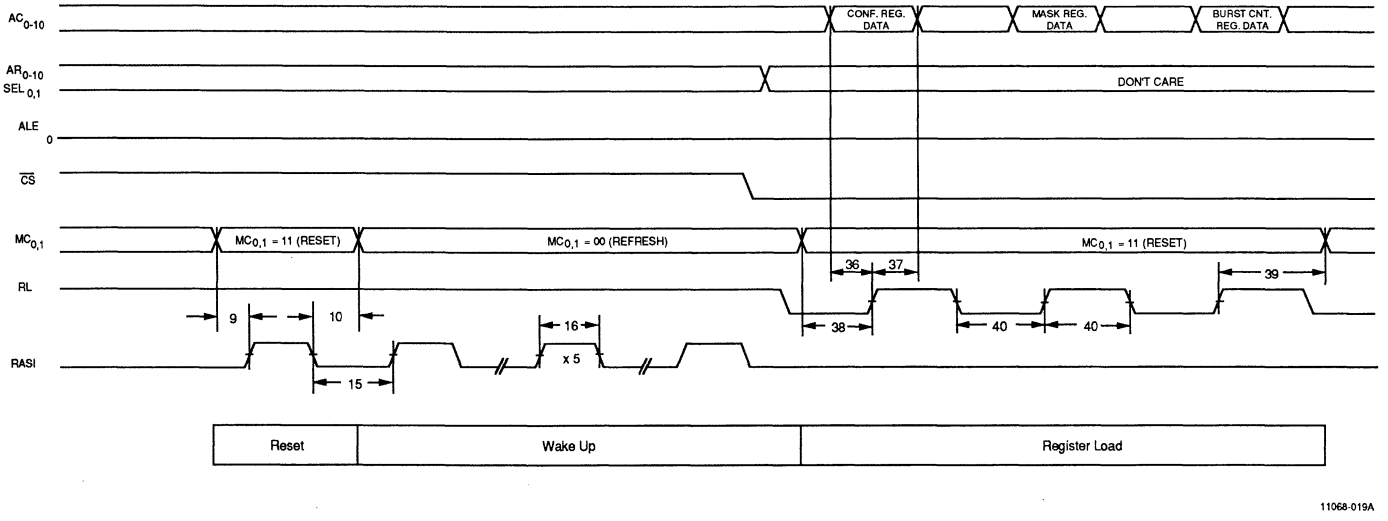
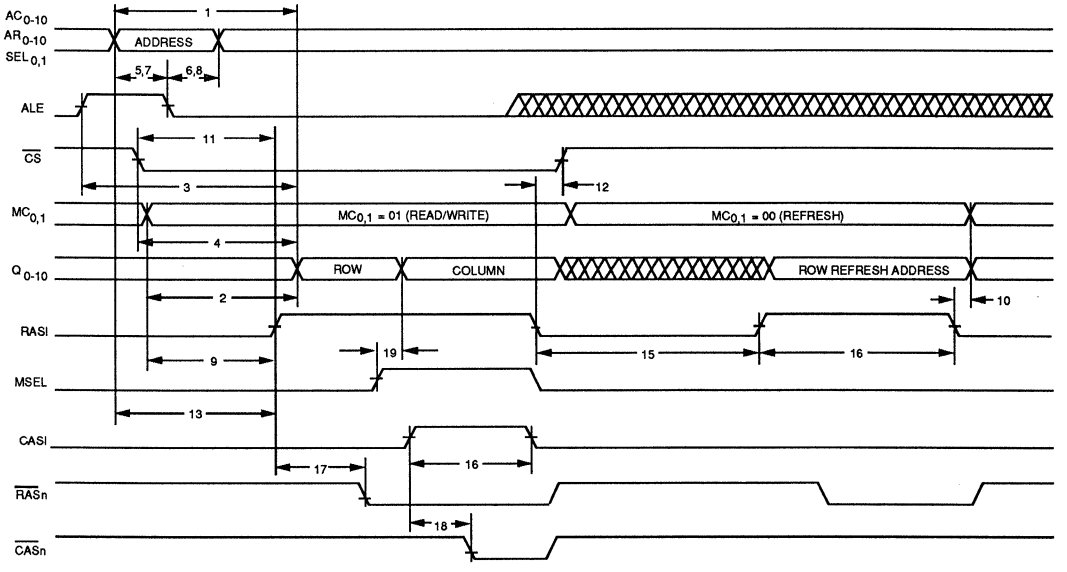
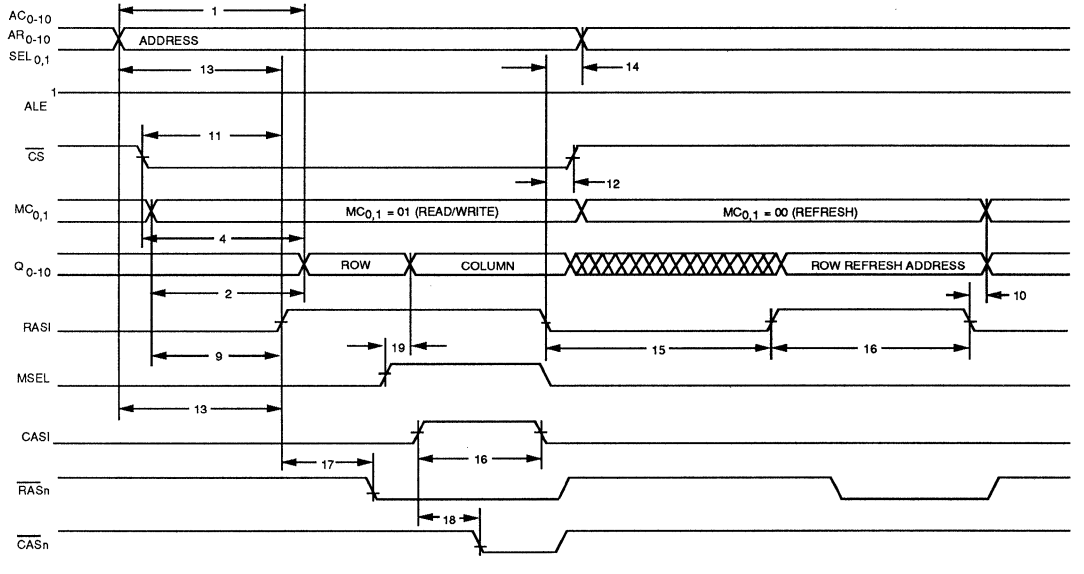


Figure 15. Reset, Wake Up, and Register Load Operation



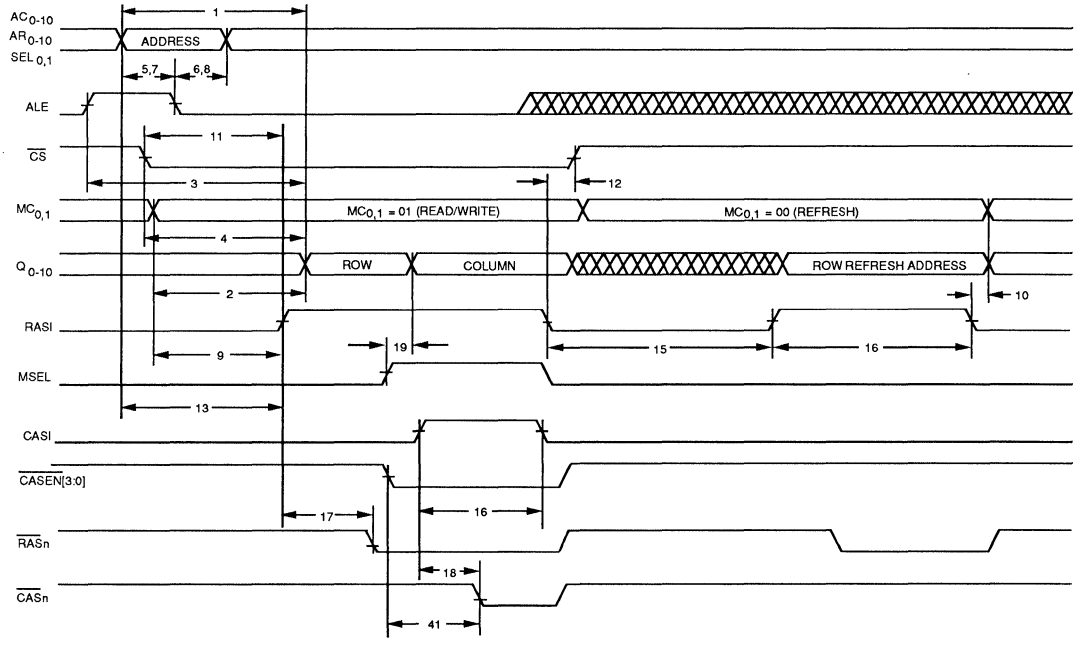
11068-020A

Figure 16. Standard Read/Write, Refresh Accesses with External Timing



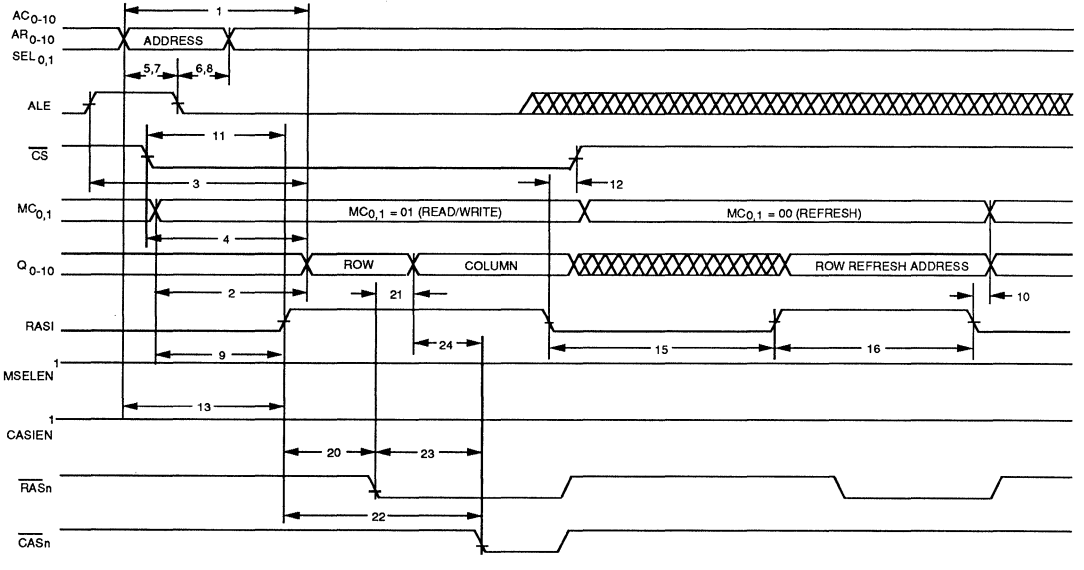
11068-021A

Figure 17. Standard Read/Write, Refresh Accesses with ALE = 1



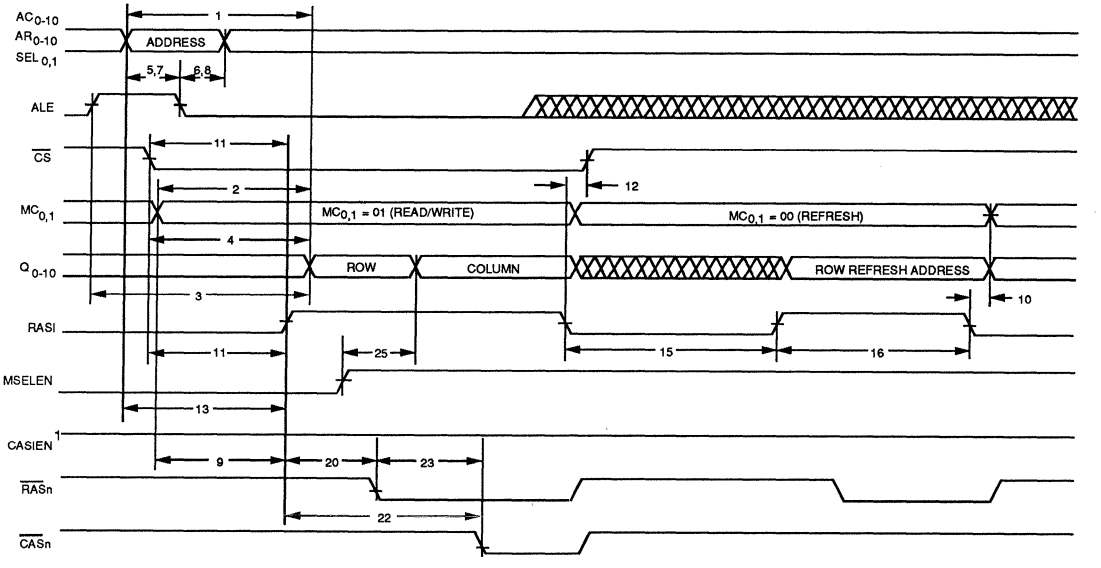
11068-022A

Figure 18. Standard Read/Write, Byte Access; Refresh (External Timing)



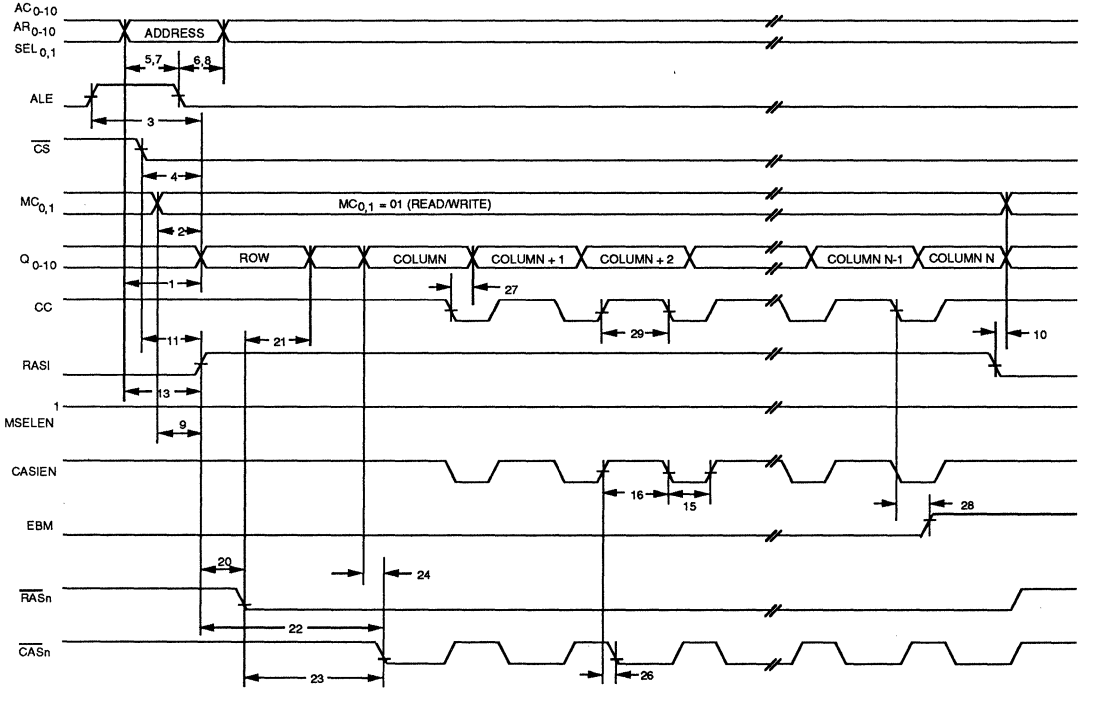
11068-023A

Figure 19. Standard Read/Write, Refresh Accesses with Auto-Timing



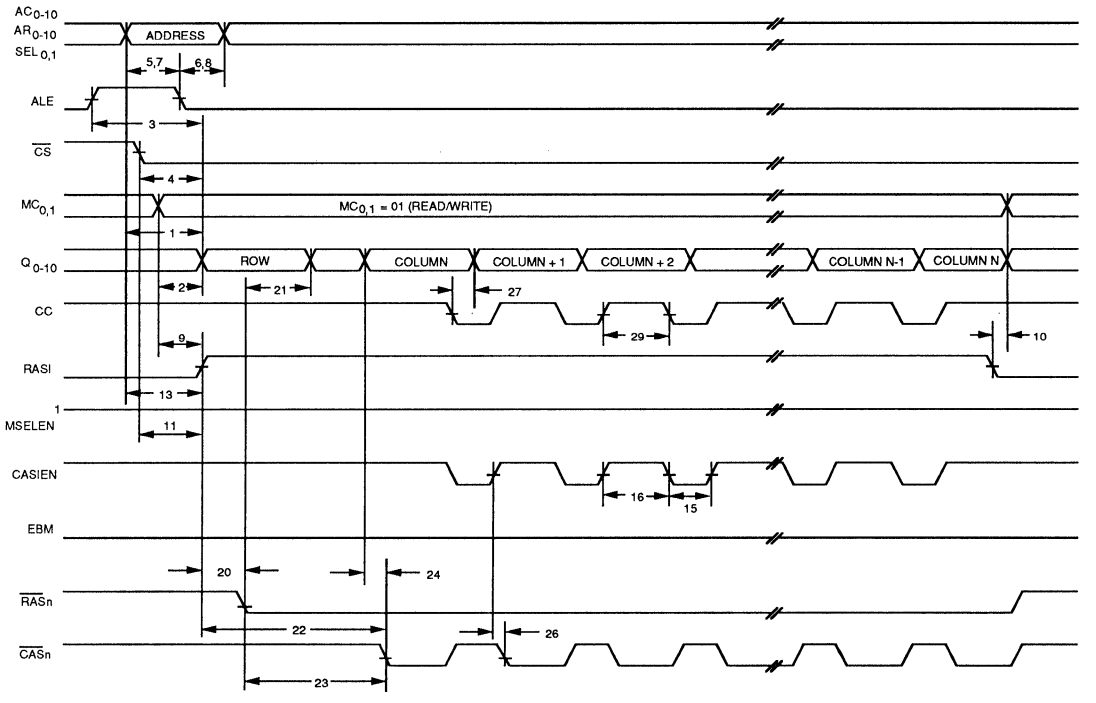
11068-024A

Figure 20. Read/Write, with Auto-Timing with External Override on MSELEN



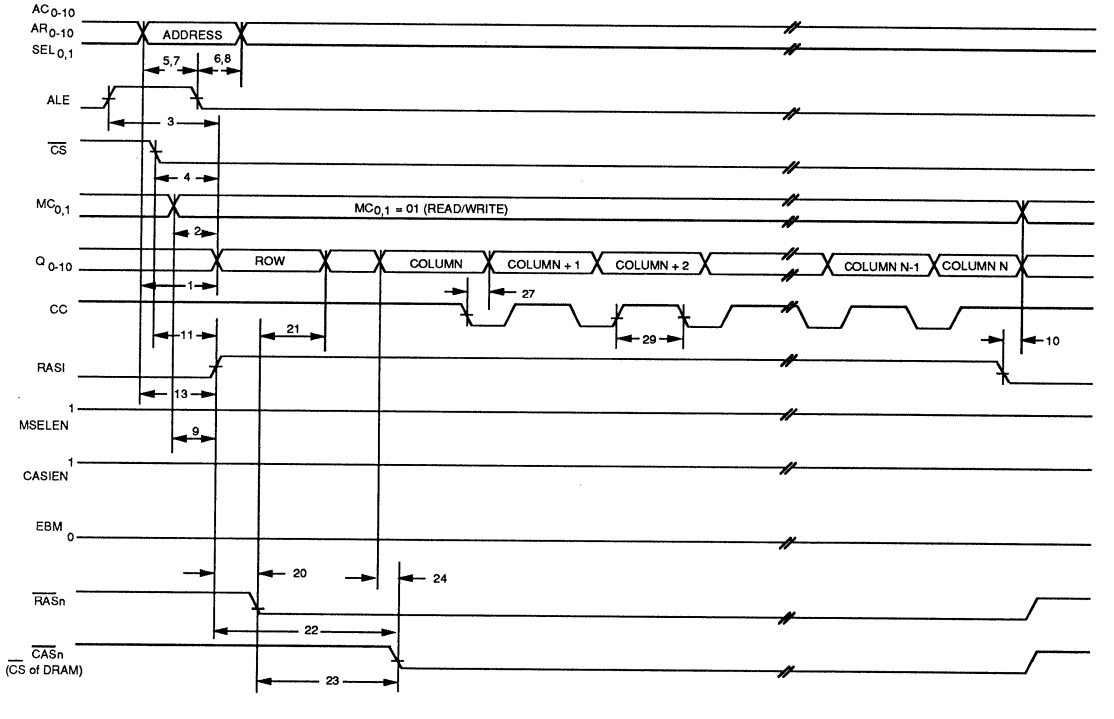
11068-025A

Figure 21. Burst Mode Access Ended by the Am29C668 (Auto-Timing with External Override)



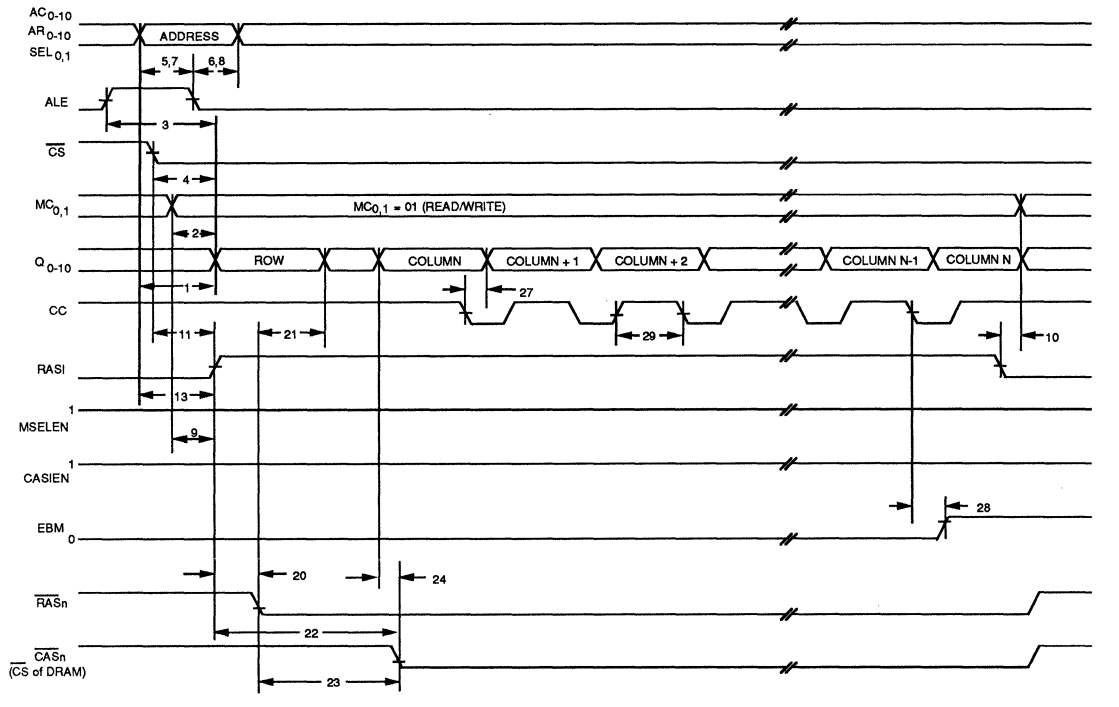
11068-026A

Figure 22. Burst Mode Access with Static Column DRAMs Ended by the Microprocessor (Auto-Timing)



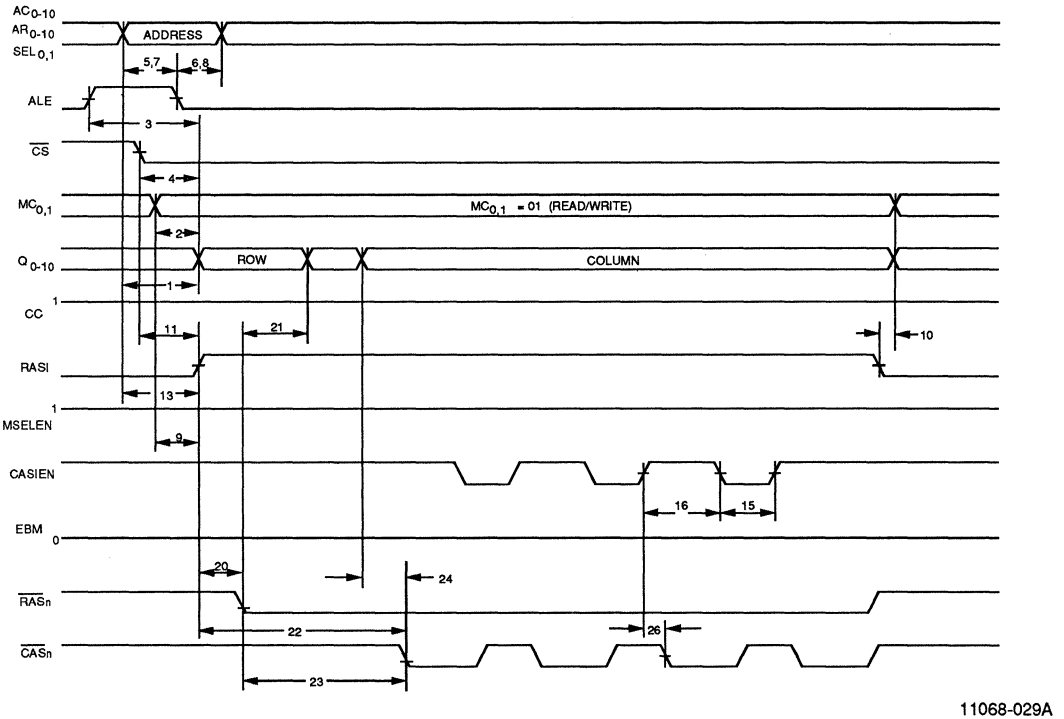
11068-027A

Figure 23. Burst Mode Access Ended by the Microprocessor (Auto-Timing with External Override)



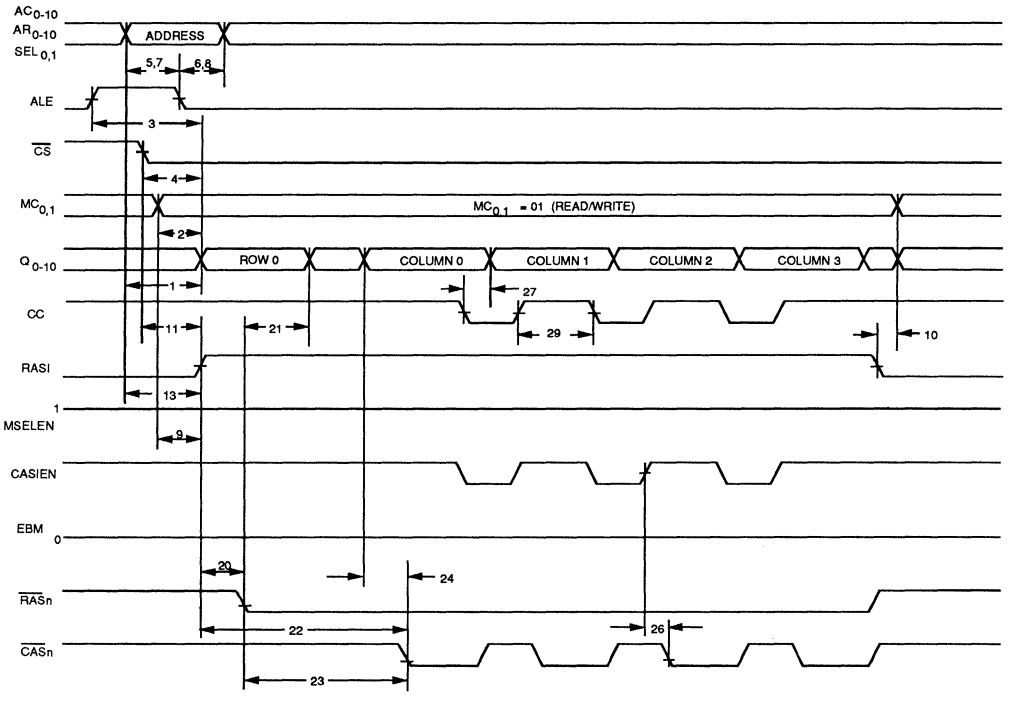
11068-028A

Figure 24. Burst Mode Access with Static Column DRAMs Ended by the Am29C668 (Auto-Timing)



11068-029A

Figure 25. Burst Mode Access with Nibble Mode DRAMs (Auto-Timing with External Override)



11068-030A

Figure 26. Nibble Mode Access with Page Mode DRAMs (Auto-Timing with External Override)

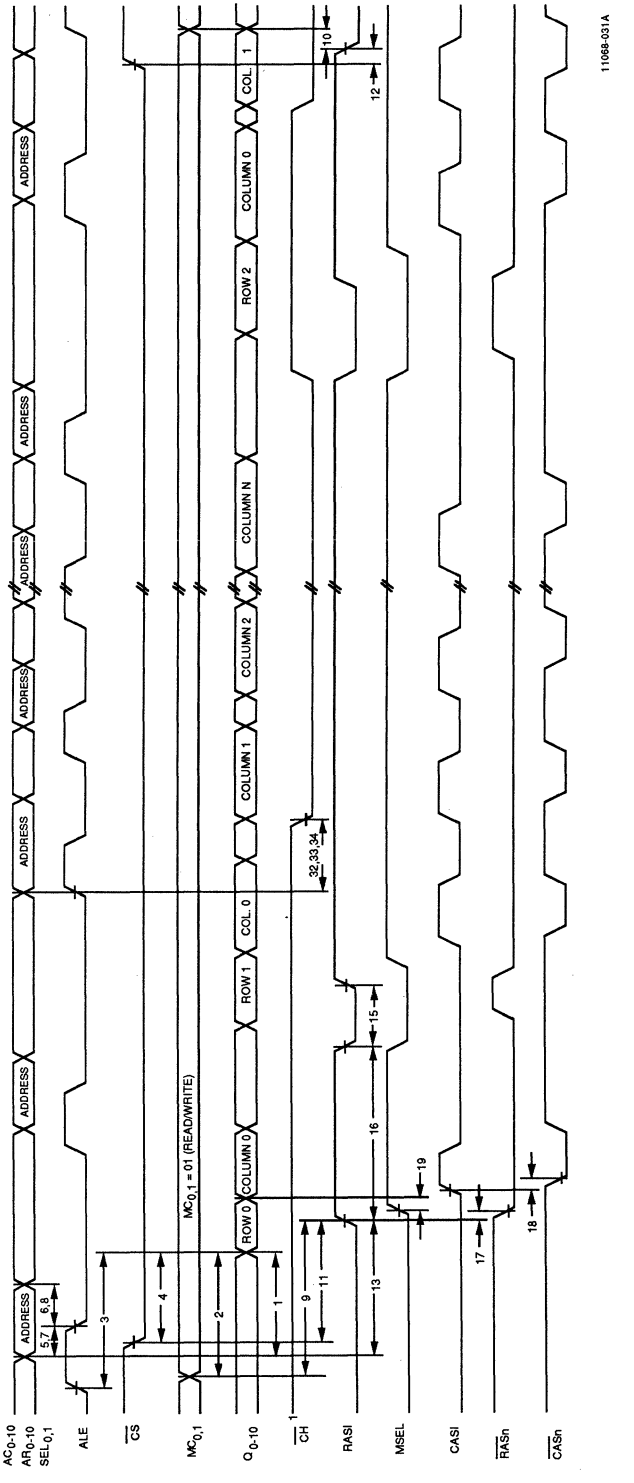


Figure 27. "Cache" Mode Access with Page Mode DRAMs (External Timing)

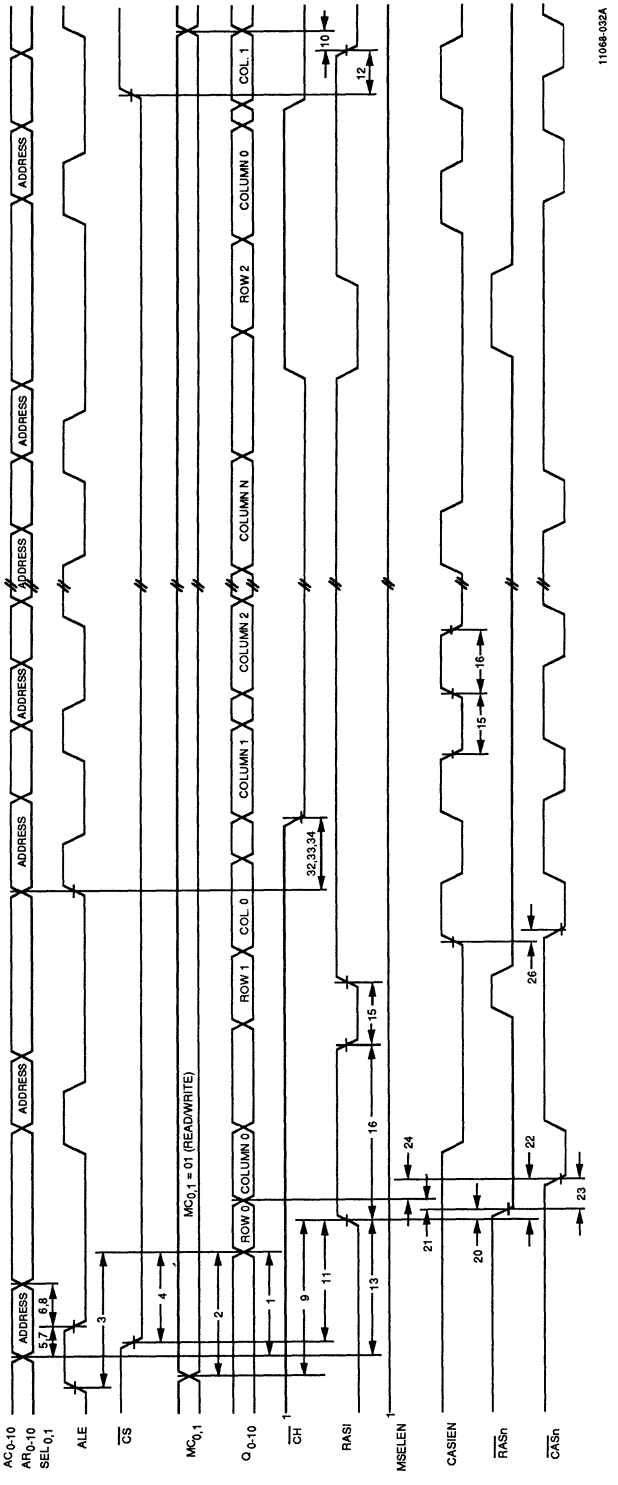


Figure 28. "Cache" Mode Access with Page Mode DRAMs (Auto-Timing with External Override)

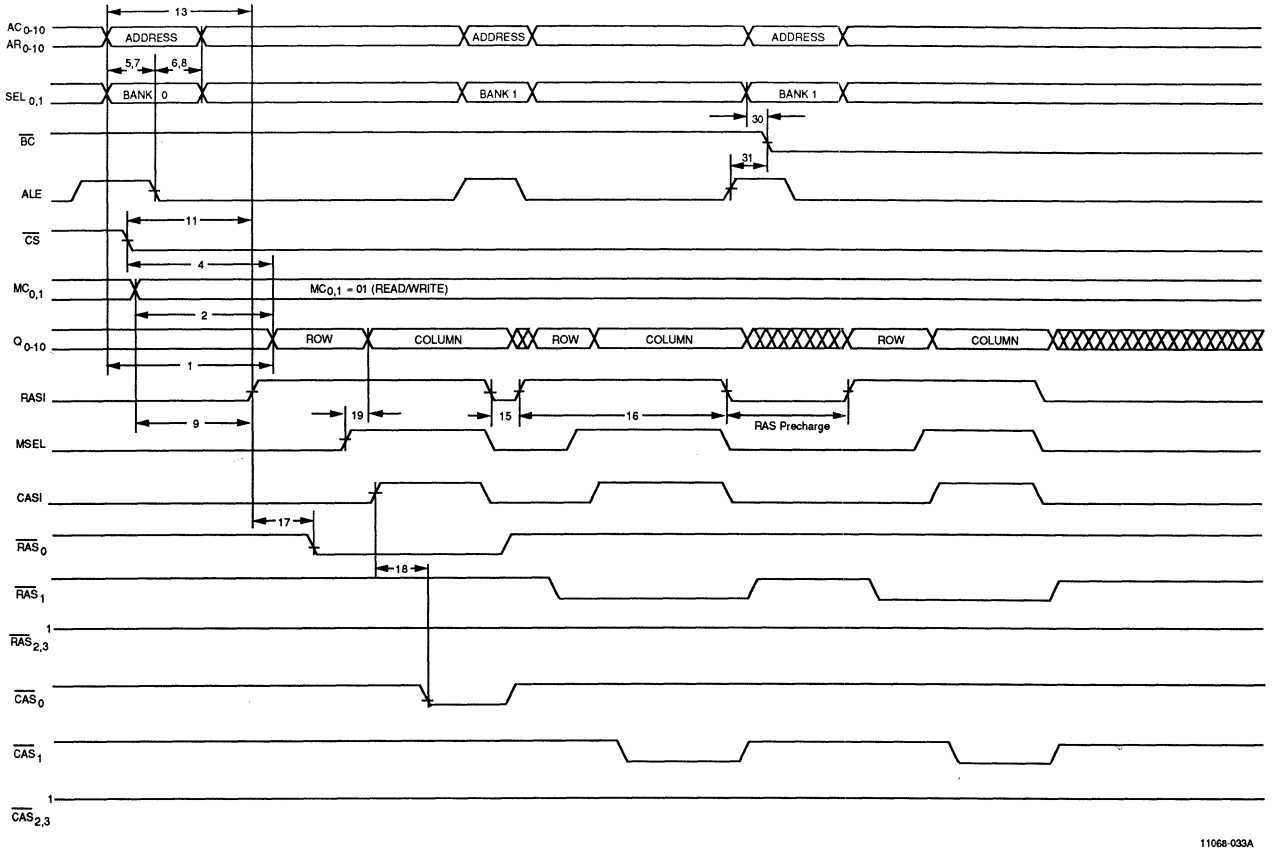


Figure 29. Bank Interleaved Accesses with External Timing

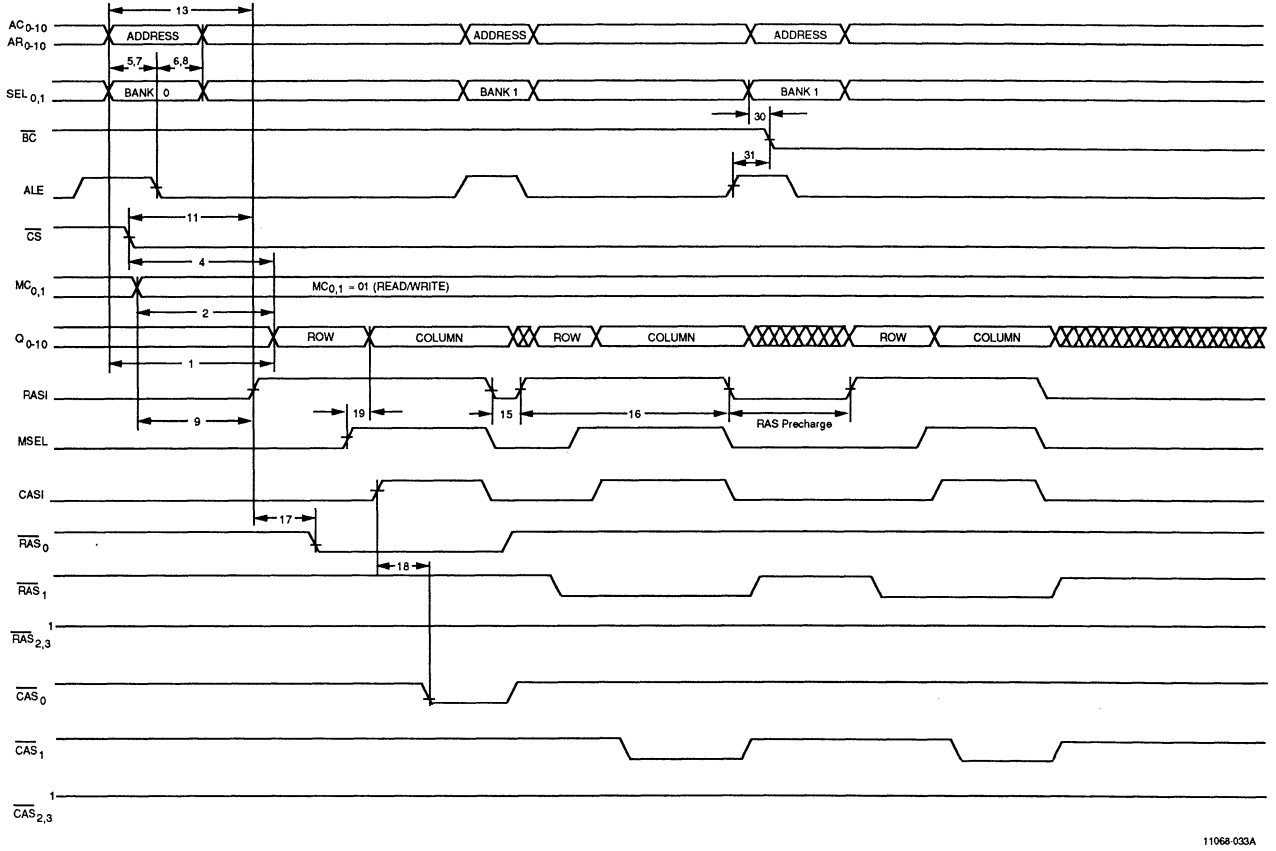


Figure 30. Bank Interleaved Accesses with Auto Timing

Preliminary

Am29C827A/Am29C828A

High-Performance CMOS Bus Buffers

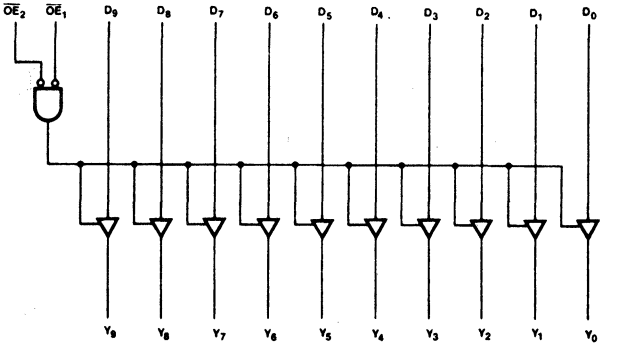
Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- High-speed CMOS buffers and inverters
  - D-Y delay = 4 ns typical
- Low standby power
- JEDEC FCT-compatible specs
- Very high output drive
  - I_{OL} = 48 mA Commercial, 32 mA Military
- 200-mV typical hysteresis on data input ports
- Proprietary edge-rate controlled outputs
- Power-up/down disable circuit provides for glitch-free power supply sequencing
- Ideal for driving 1Mbit x 1 and 1Mbit x 4 DRAM address inputs.

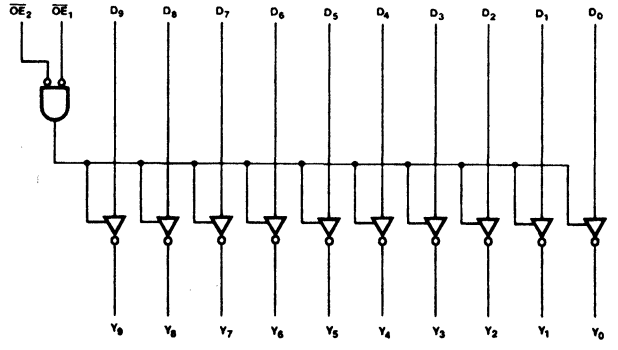
BLOCK DIAGRAMS

Am29C827A (Noninverting)



BD001092

Am29C828A (Inverting)



BD001093

Publication# 11228 Rev. B Amendment /0
Issue Date: January 1990

GENERAL DESCRIPTION

The Am29C827A and Am29C828A CMOS Bus Buffers provide high-performance bus interface buffering for wide address/data paths or buses carrying parity. Both devices feature 10-bit wide data paths and NORed output enables for maximum control flexibility. The Am29C827A has non-inverting outputs, while the Am29C828A has inverting outputs. Each device has data inputs with 200-mV typical input hysteresis to provide improved noise immunity. The Am29C827A and Am29C828A are produced with AMD's exclusive CS11SA CMOS process, and feature typical propagation delays of 4 ns, as well as an output current drive of 48 mA.

The Am29C827A and Am29C828A incorporate AMD's proprietary edge-controlled outputs in order to minimize simultaneous switching noise (ground bounce). By controlling the output transient currents, ground bounce and ouput ringing have been greatly reduced. A modified AMD output provides a stable, usable voltage level in less time than a non-controlled output.

Additionally, speed degradation due to increasing number of outputs switching is reduced. Together, these benefits or edge-rate control result in significant increase in system performance despite a minor increase in device propagation delay.*

A unique I/O circuitry provides for high-impedance outputs during power-off and power-up/down sequencing, thus providing glitch-free operation for card-edge and other active bus applications.

The Am29C827A and Am29C828A are available in the standard package options: DIPs, PLCCs, and SOICs.

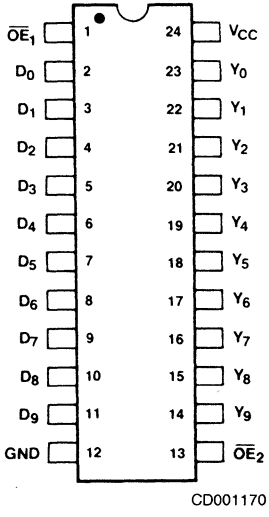
*For more details refer to a Minimization of Ground Bounce Through Output Edge-Rate Control Application Note (PID #10181A)

CONNECTION DIAGRAMS

Top View

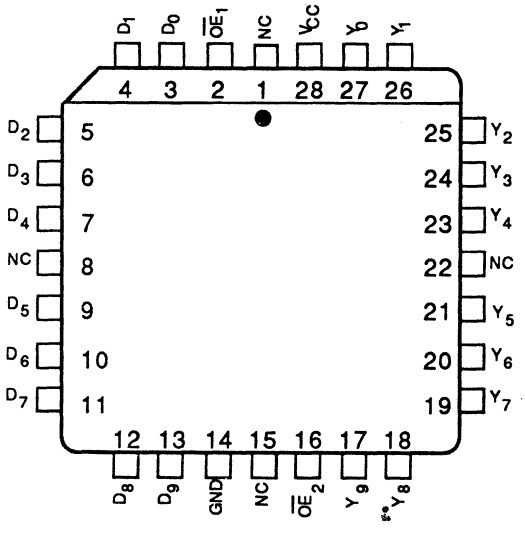
Am29C827A/Am29C828A

DIPs*



CD001170

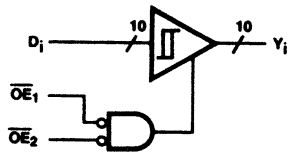
PLCC



*Also available in Small Outline package; pinout identical to DIPs.

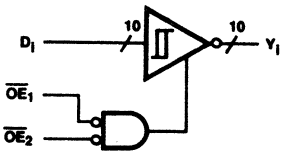
LOGIC SYMBOLS

Am29C827A



LS000391

Am29C828A



LS000393

FUNCTION TABLES

Am29C827A

Inputs			Outputs	
$\overline{OE}_1$	$\overline{OE}_2$	D_i	Y_i	Function
L	L	H	H	Transparent
L	L	L	L	Transparent
X	H	X	Z	Hi-Z
H	X	X	Z	Hi-Z

Am29C828A

Inputs			Outputs	
$\overline{OE}_1$	$\overline{OE}_2$	D_i	Y_i	Function
L	L	H	L	Transparent
L	L	L	H	Transparent
X	H	X	Z	Hi-Z
H	X	X	Z	Hi-Z

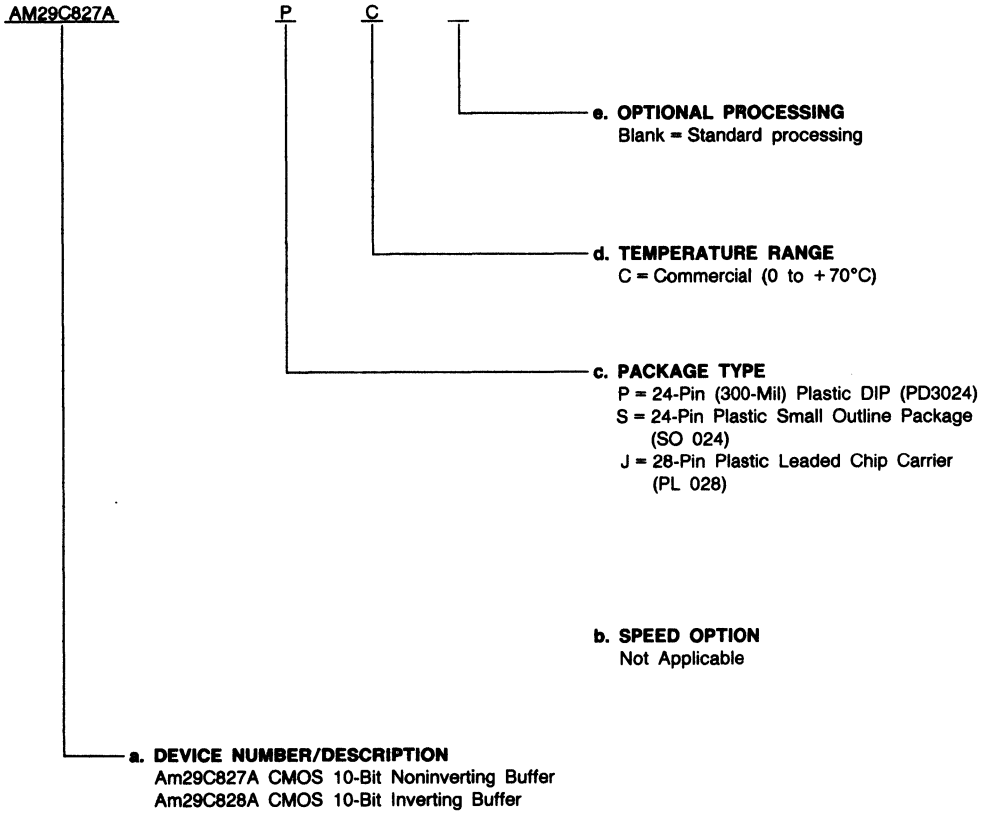
H = HIGH
L = LOW
X = Don't Care
Z = Hi-Z

ORDERING INFORMATION

Standard Products

AMD products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29C827A	PC, SC, JC
AM29C828A	

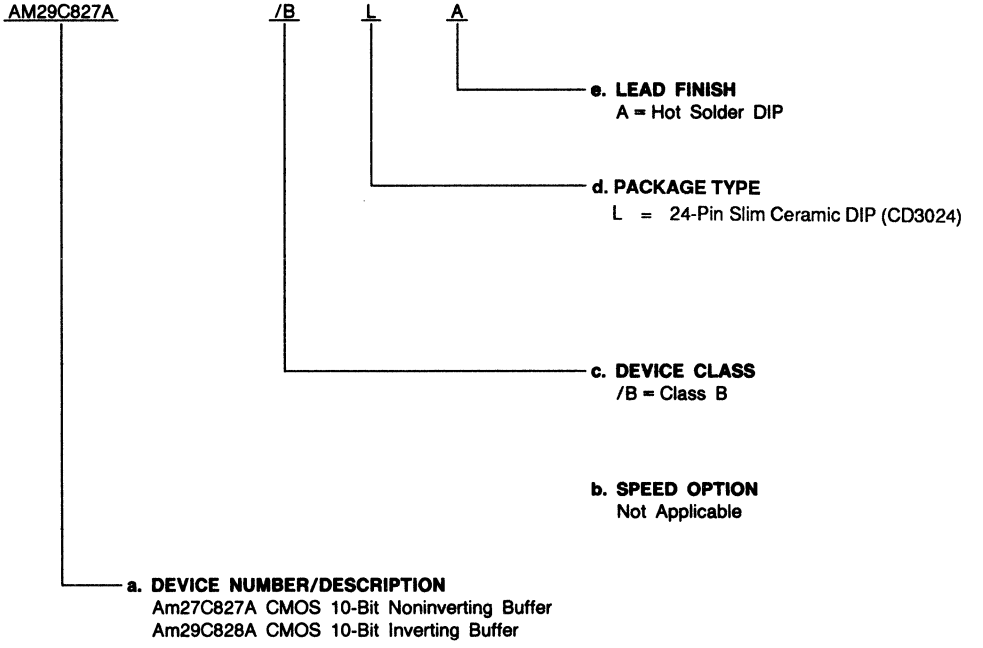
Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released valid combinations, and to obtain additional data on AMD's standard military grade products.

MILITARY ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of: **a. Device Number**
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29C827A	/BLA
AM29C828A	

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check for newly released valid combinations.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

$\overline{OE}_I$ Output Enables (Input, Active LOW)
When $\overline{OE}_1$ and $\overline{OE}_2$ are both LOW, the outputs are enabled. When either one or both are HIGH, the outputs are in the Hi-Z state.

D_I Data Inputs (Input)
D_i are the 10-bit data inputs.

Y_I Data Output (Output)
Y_i are the 10-bit data outputs.

ABSOLUTE MAXIMUM RATINGS

Storage Temperature ..... −65 to +150°C
Supply Voltage to Ground Potential
Continuous ..... −0.5 V to +6.0 V
DC Output Voltage ..... −0.5 V to +6.0 V
DC Input Voltage ..... −0.5 V to +6.0 V
DC Output Diode Current: Into Output ..... +50 mA
Out of Output ..... −50 mA
DC Input Diode Current: Into Input ..... +20 mA
Out of Input ..... −20 mA
DC Output Current per Pin:
I_{SINK} ..... +70 mA
I_{SOURCE} ..... −30 mA
Total DC Ground Current .$(n \times I_{OL} + m \times I_{CCT})$ mA (Note 1)
Total DC V_{CC} Current .... $(n \times I_{OH} + m \times I_{CCT})$ mA (Note 1)

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices
Temperature (T_A) ..... 0 to +70°C
Supply Voltage (V_{CC}) ..... +4.5 V to +5.5 V

Military (M) Devices
Temperature (T_A) ..... −55 to +125°C
Supply Voltage (V_{CC}) ..... +4.5 V to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

PRELIMINARY

DC CHARACTERISTICS over operating range unless otherwise specified (for APL Products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions			Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IH} or V_{IL}	I_{OH} = −15 mA		2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IH} or V_{IL}	MIL I_{OL} = 32 mA			0.5	V
			COM'L I_{OL} = 48 mA				
V_{IH}	Input HIGH Voltage	Guaranteed Input Logical HIGH Voltage for All Inputs (Note 2)			2.0		V
V_{IL}	Input LOW Voltage	Guaranteed Input Logical LOW Voltage for All Inputs (Note 2)				0.8	V
V_I	Input Clamp Voltage	V_{CC} = 4.5 V, I_{IN} = −18 mA				−1.2	V
I_{IL}	Input LOW Current	V_{CC} = 5.5 V, V_{IN} = GND				−10	μA
		V_{CC} = 5.5 V, V_{IN} = 0.4 V				−5	
I_{IH}	Input HIGH Current	V_{CC} = 5.5 V, V_{IN} = 2.7 V				5	μA
		V_{CC} = 5.5 V, V_{IN} = 5.5 V				10	
I_{OZH}	Output Off-State Current (High Impedance)	V_{CC} = 5.5 V, V_O = 5.5 V or 2.7 V (Note 3)				+10	μA
I_{OZL}		V_{CC} = 5.5 V, V_O = 0.4 V or GND (Note 3)				−10	μA
I_{SC}	Output Short-Circuit Current	V_{CC} = 5.5 V, V_O = 0 V (Note 4)			−60		mA
I_{CCQ}	Static Supply Current	V_{CC} = 5.5 V, Outputs Open	V_{IN} = V_{CC} or GND	MIL		1.5	mA
				COM'L		1.2	
I_{CCT}			V_{IN} = 3.4 V	Data Input		1.5	mA/Bit
				$\overline{OE}_1$, $\overline{OE}_2$		3.0	
I_{CCD}†	Dynamic Supply Current	V_{CC} = 5.5 V (Note 5)				275	μA/MHz/Bit

Notes: 1. n = number of outputs, m = number of inputs.
2. Input thresholds are tested in combination with other DC parameters or by correlation.
3. Off-state currents are only tested at worst-case conditions of V_{OUT} = 5.5 V or 0.0 V.
4. Not more than one output should be shorted at a time. Duration should not exceed 100 milliseconds.
5. Measured at a frequency ⩽ 10 MHz with 50% duty cycle.

† Not included in Group A tests.

SWITCHING CHARACTERISTICS over operating range unless otherwise specified (for APL Products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions*	COMMERCIAL		MILITARY		Units
			Min.	Max.	Min.	Max.	
t_{PLH}	Data (D_i) to Output (Y_i) Am29C827A (Noninverting) (Note 1)	C_L = 50 pF R_1 = 500 Ω R_2 = 500 Ω	1.0	6.5	1.0	7.5	ns
t_{PHL}			1.0	6.5	1.0	7.5	ns
t_{PLH}	Data (D_i) to Output (Y_i) Am29C828A (Inverting) (Note 1)		1.0	6.5	0.5	7.5	ns
t_{PHL}			1.0	6.5	0.5	7.5	ns
t_{ZH}	Output Enable Time $\overline{OE}$ to Y_i		1.0	9	1.0	11	ns
t_{ZL}			3.0	12	3.0	14	ns
t_{HZ}	Output Disable Time $\overline{OE}$ to Y_i		2.0	8	2.0	9	ns
t_{LZ}			3.0	8	2.0	9	ns

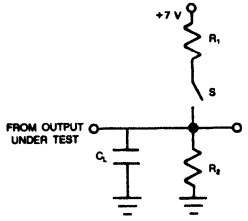
*See Test Circuit and Waveforms.

Notes: 1. For more details refer to a Minimization of Ground Bounce Through Output Edge-Rate Control Application Note (PID #10181A).

PRELIMINARY

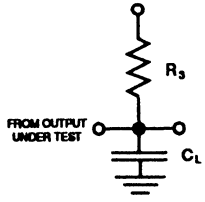
SWITCHING TEST CIRCUITS

THREE-STATE OUTPUTS



TC002682

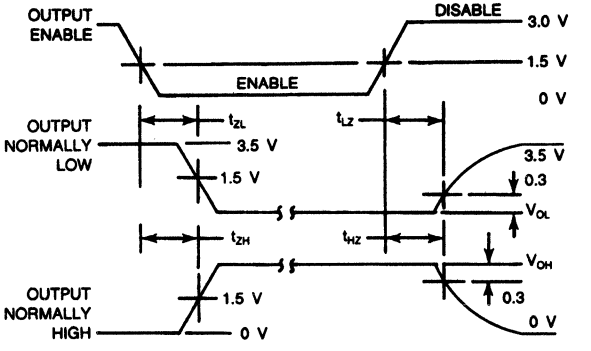
OPEN-DRAIN OUTPUTS



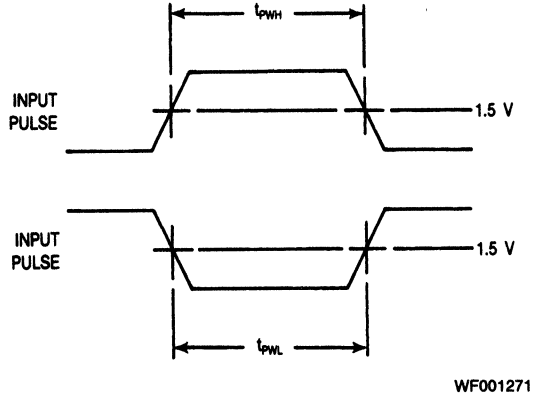
SWITCH POSITIONS FOR PARAMETER TESTING

Parameter	S Position
t_{PLH}	OPEN
t_{PHL}	OPEN
t_{HZ}	OPEN
t_{ZH}	OPEN
t_{LZ}	CLOSED
t_{ZL}	CLOSED

SWITCHING TEST WAVEFORMS



Enable and Disable Times



Pulse Width

Preliminary

Am29C983/Am29C983A

9-Bit x 4-Port Multiple Bus Exchange

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- **Four bidirectional I/O ports with latches**
  - Replaces several bidirectional latches and transceivers
  - Permits multiple bus communication
  - Allows two independent communication channels
  - TTL compatibility
- **9 bit-wide ports to handle byte parity**
- **Two selection inputs per port**
  - Independent port interconnect control
  - Increased flexibility in data routing
- **Matched port decoding**
  - Simplifies external decode logic
  - Easily cascadable for wider buses
- **Latches for incoming and outgoing data**
  - Independent controls permit selective data capture
  - Ideal for stored operation
  - Readback feature for system diagnostics
- **Glitch-free outputs during power-up/down**
  - No power-up sequencing needed
  - Ideal for card-edge interface
- **48 mA output drive**
  - High-capacitance bus driving
- **High-performance CMOS**
  - Low stand-by power consumption
- **Two speeds available**
  Am29C983
  - 9 ns (typ) port-to-port delay
  - 10 ns (typ) select-to-port delay
  Am29C983A
  - 6 ns (typ) port-to-port delay
  - 7 ns (typ) select-to-port delay
- **Available in 68-pin PLCC and PGA packages**
  - Significant savings in board space

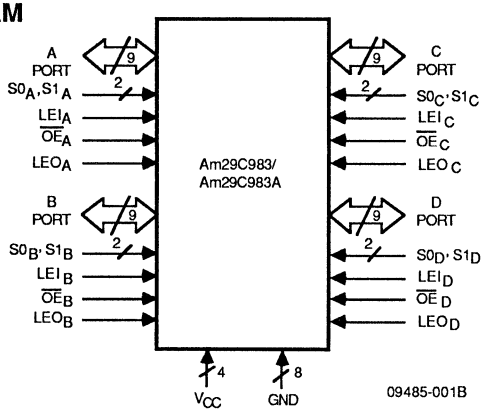
GENERAL DESCRIPTION

The Am29C983/A is a high-speed Multiple Bus Exchange device. It is organized as four 9-bit wide TTL-compatible I/O ports with Output Enable control for each port. Any port can serve either as a source (Input) port or as a destination (Output) port. When the output drivers of a port are disabled (high-impedance state), the port serves as a source port. When the drivers are enabled, the port serves as a destination port. Source port selection is made by two independent Select inputs at each port. This organization offers flexibility in implementing the Am29C983/A as a digital cross-point switch for multiple bus communication in a multiprocessing environment.

Each I/O port has an input latch to capture incoming data and an output latch to capture outgoing data. All input and output latches are independently controlled by active-HIGH Latch Enable inputs. This feature can be used to perform stored operation for byte-word compression and expansion to communicate between buses of different widths.

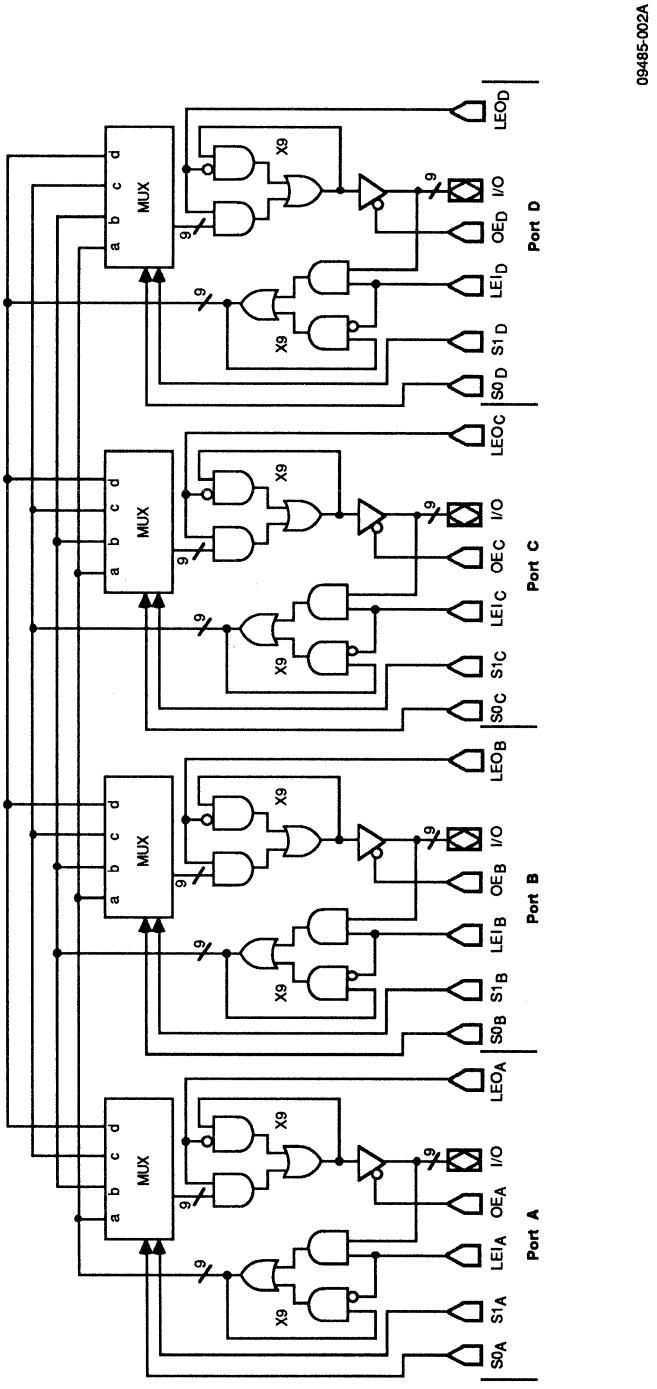
Independent port control permits cascading of Am29C983/As for wider buses. All I/O ports go into high-impedance state upon power-down. This feature makes the device ideally suited for card-edge applications.

SIMPLIFIED BLOCK DIAGRAM

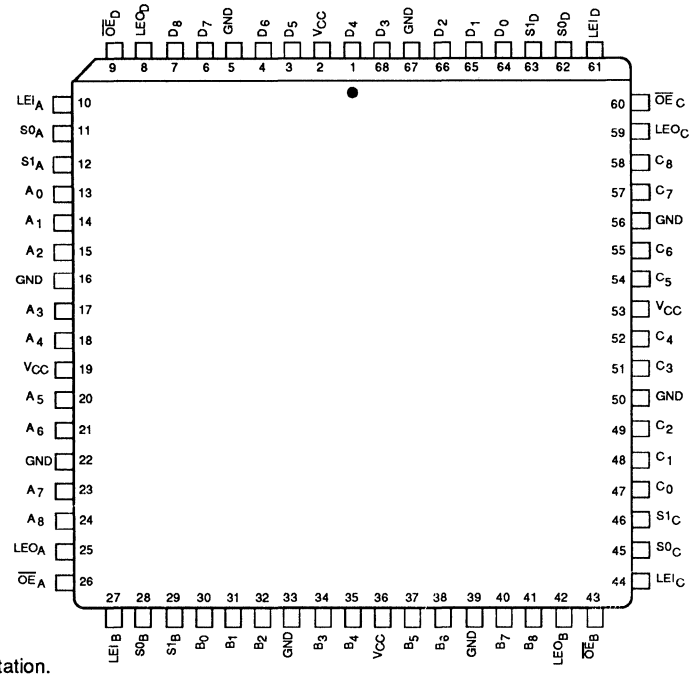


Publication# 09485 Rev. C Amendment /0
Issue Date: February 1990

DETAILED BLOCK DIAGRAM



CONNECTION DIAGRAM
PLCC
(Top View)



Note:
Pin 1 is marked for orientation.

PGA
(Bottom View)

	A	B	C	D	E	F	G	H	J	K	L
1		D_8	GND	D_5	D_4	GND	D_2	D_0	$S0_D$	$\overline{OE}_C$	
2	$\overline{OE}_D$	LEO_D	D_7	D_6	V_{CC}	D_3	D_1	$S1_1$	LEI_D	LEO_D	C_8
3	$S0_A$	LEI_A								C_7	GND
4	A_0	$S1_A$								C_6	C_5
5	A_2	A_5								C_4	V_{CC}
6	A_3	GND								C_3	GND
7	A_4	V_{CC}								C_1	C_2
8	A_5	A_6								$S1_7$	C_0
9	GND	A_7								LEI_C	$S0_C$
10	A_8	LEO_A	LEI_B	$S1_B$	B_1	B_3	V_{CC}	B_6	B_7	LEO_B	$\overline{OE}_8$
11		$\overline{OE}_A$	$S0_B$	B_0	B_2	GND	B_4	B_5	GND	B_8	

09485-013A

Note:
Notch indicates orientation.

PIN DESIGNATIONS

(Sorted by Pin Name)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-4	A_0	K-11	B_8	C-2	D_7	B-2	LEO_D
B-5	A_1	L-8	C_0	B-1	D_8	B-11	$\overline{OE}_A$
A-5	A_2	K-7	C_1	A-9	GND	L-10	$\overline{OE}_B$
A-6	A_3	L-7	C_2	B-6	GND	K-1	$\overline{OE}_C$
A-7	A_4	K-6	C_3	C-1	GND	A-2	$\overline{OE}_D$
A-8	A_5	K-5	C_4	F-1	GND	A-3	$S0_A$
B-8	A_6	L-4	C_5	F-11	GND	C-11	$S0_B$
B-9	A_7	K-4	C_6	J-11	GND	L-9	$S0_C$
A-10	A_8	K-3	C_7	L-3	GND	J-1	$S0_D$
D-11	B_0	L-2	C_8	L-6	GND	B-4	$S1_A$
E-10	B_1	H-1	D_0	B-3	LEI_A	D-10	$S1_B$
E-11	B_2	G-2	D_1	C-10	LEI_B	K-8	$S1_C$
F-10	B_3	G-1	D_2	K-9	LEI_C	H-2	$S1_D$
G-11	B_4	F-2	D_3	J-2	LEI_D	B-7	V_{CC}
H-11	B_5	E-1	D_4	B-10	LEO_A	E-2	V_{CC}
H-10	B_6	D-1	D_5	K-10	LEO_B	G-10	V_{CC}
J-10	B_7	D-2	D_6	K-2	LEO_C	L-5	V_{CC}

(Sorted by Pin Number)

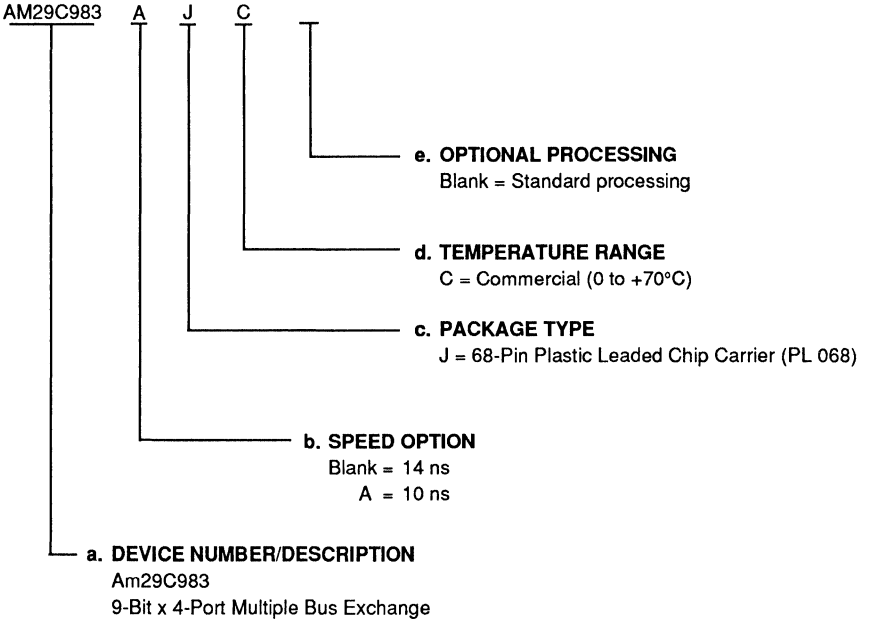
PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-2	$\overline{OE}_D$	B-9	A_7	F-10	B_3	K-4	C_6
A-3	$S0_A$	B-10	LEO_A	F-11	GND	K-5	C_4
A-4	A_0	B-11	$\overline{OE}_A$	G-1	D_2	K-6	C_3
A-5	A_2	C-1	GND	G-2	D_1	K-7	C_1
A-6	A_3	C-2	D_7	G-10	V_{CC}	K-8	$S1_C$
A-7	A_4	C-10	LEI_B	G-11	B_4	K-9	LEI_C
A-8	A_5	C-11	$S0_B$	H-1	D_0	K-10	LEO_B
A-9	GND	D-1	D_5	H-2	$S1_D$	K-11	B_8
A-10	A_8	D-2	D_6	H-10	B_6	L-2	C_8
B-1	D_8	D-10	$S1_B$	H-11	B_5	L-3	GND
B-2	LEO_D	D-11	B_0	J-1	$S0_D$	L-4	C_5
B-3	LEI_A	E-1	D_4	J-2	LEI_D	L-5	V_{CC}
B-4	$S1_A$	E-2	V_{CC}	J-10	B_7	L-6	GND
B-5	A_1	E-10	B_1	J-11	GND	L-7	C_2
B-6	GND	E-11	B_2	K-1	$\overline{OE}_C$	L-8	C_0
B-7	V_{CC}	F-1	GND	K-2	LEO_C	L-9	$S0_C$
B-8	A_6	F-2	D_3	K-3	C_7	L-10	$\overline{OE}_B$

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
AM29C983	JC
AM29C983A	

Valid Combinations

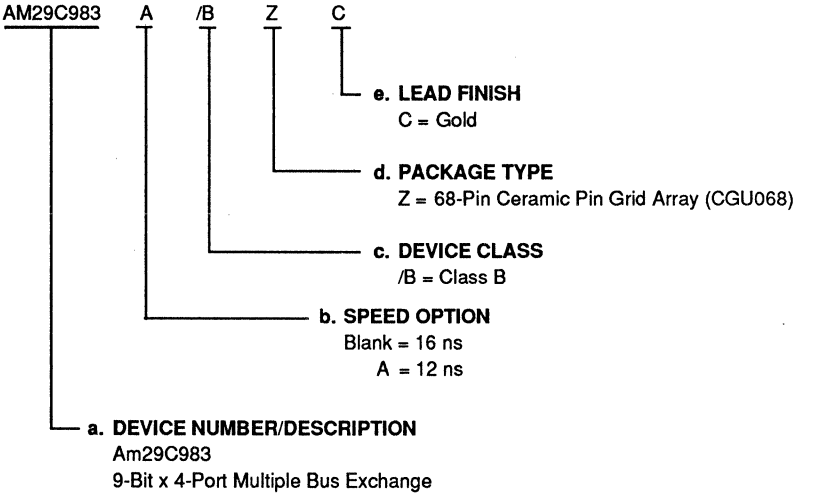
Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

ORDERING INFORMATION

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Device Class
d. Package Type
e. Lead Finish



Valid Combinations	
AM29C983	/BZC
AM29C983A	

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check on newly released combinations.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

A_I, B_I, C_I, AND D_I (I = 0 THROUGH 8)
DATA BUS I/O PORTS (INPUT/OUTPUT)

These four groups of nine I/O pins are defined as the A, B, C, and D ports respectively. Each port serves as a source (Input) or as a destination (Output).

SI_A, SI_B, SI_C, AND SI_D (I = 0, 1)
SOURCE PORT SELECT (INPUTS)

Each pair of inputs determines the source of data for the corresponding I/O port when used as a destination port.

LEI_A, LEI_B, LEI_C, AND LEI_D
INPUT LATCH ENABLE (INPUTS; ACTIVE HIGH)

Each LEI input controls a 9-bit wide latch on the input side of the corresponding I/O port. The latches are transparent when LEI is HIGH and are latched when LEI is LOW.

LEO_A, LEO_B, LEO_C, AND LEO_D
OUTPUT LATCH ENABLE (INPUTS; ACTIVE HIGH)

Each LEO input controls a 9-bit wide latch on the output side of the corresponding I/O port. The latches are transparent when LEO is HIGH and are latched when LEO is LOW.

$\overline{OE}_A$, $\overline{OE}_B$, $\overline{OE}_C$, AND $\overline{OE}_D$
OUTPUT ENABLE (INPUTS; ACTIVE LOW)

Each $\overline{OE}$ input controls the bus drivers of the corresponding I/O port. When $\overline{OE}$ is LOW, data at the output of the Output latches is passed to the bus. When $\overline{OE}$ is HIGH, the bus outputs are in high-impedance state.

FUNCTIONAL DESCRIPTION

The Am29C983/A Multiple Bus Exchange consists of four 9-bit I/O ports. Each port has a 9-bit Input latch to capture incoming data and a 9-bit Output latch to capture outgoing data. There are five control inputs associated with each port: two Select inputs for source port selection, two Latch Enable inputs (active HIGH) to control Input and Output latches, and an active LOW Output Enable line to control the bus driver at the I/O port.

Port Selection and Control

Each port is independently controlled by means of these five control inputs. If the output drivers of a port are disabled (high-impedance state), that port is an input and can be used as a source port. Incoming data can be captured in the Input latch. At the same time, the data at one of the four internal buses can be transferred to the Output latch under the control of the appropriate Select inputs. If the output drivers are enabled, the port serves as a destination port, transporting the data at the output of its Output latch to the external bus connected to the I/O port. Independent control of Input and Output latches and output drivers permits stored operation at any port.

Multiple Bus Communication

Four internal buses serve as pathways for port-to-port connection. By proper choice of source select codes for the ports, the Am29C983 can be configured in different modes for multiple bus communication. In one mode of operation, two ports can be selected as source ports and the other two as destination ports; thus, two independent bidirectional communication channels are established. In another mode, one port can be selected as the source, and one or more of the other ports can serve as destination ports. Any port not intended as a destination port can be disabled (high-impedance state) by means of its Output Enable control.

Input and Output Latches

The presence of Input and Output latches offers significant flexibility in using the Am29C983. Any port can be chosen as the source port to store incoming data in its Input latch. This can then be connected to one or more destination ports. The outgoing data can be further stored in the Output latches for later use; thus there are two stages of data storage between any two ports. This feature can be used in simple store and forward applications, as well as in more sophisticated applications for byte-word compression and expansion. Moreover, the data stored in the Input latch of a port can be "read back" to the same port by choosing it as the destination (its Output latch is transparent). This feature can be used for diagnostics in multiple bus communication.

TRUTH TABLES

A. Port Source Selection

$S1_n$	$S0_n$	Source
L	L	A Bus
L	H	B Bus
H	L	C Bus
H	H	D Bus

B. Latch Operation

LEI_n or LEO_n	Mode
H	Transparent
L	Latched

C. I/O Port Controls

LEO_n	$\overline{OE}_n$	I/O	Source of Data
L	L	Out	Contents of Output Latch
H	L	Out	Selected Source Port
X	H	In	

Key: n = A, B, C, or D
L = LOW
H = HIGH
X = Don't Care

APPLICATIONS

Microprocessor-Memory Interface

This application illustrates the use of matched port decoding and Input/Output latches of the Am29C983 Multiple Bus Exchange (MBE) in a simple yet powerful microprocessor-to-DRAM interface (see Figure 1).

MBE1 is used as an address latch to capture the 16-bit address into the A and B port Input latches from the multiplexed address/data bus. It multiplexes upper and lower bytes of the address to directly drive the DRAM array. MBE2 is configured as an 18-bit wide bidirectional latch to drive the memory data bus directly, or through another Am29C983 (MBE3) configured as another bidirectional latch for isolation from data bus activity.

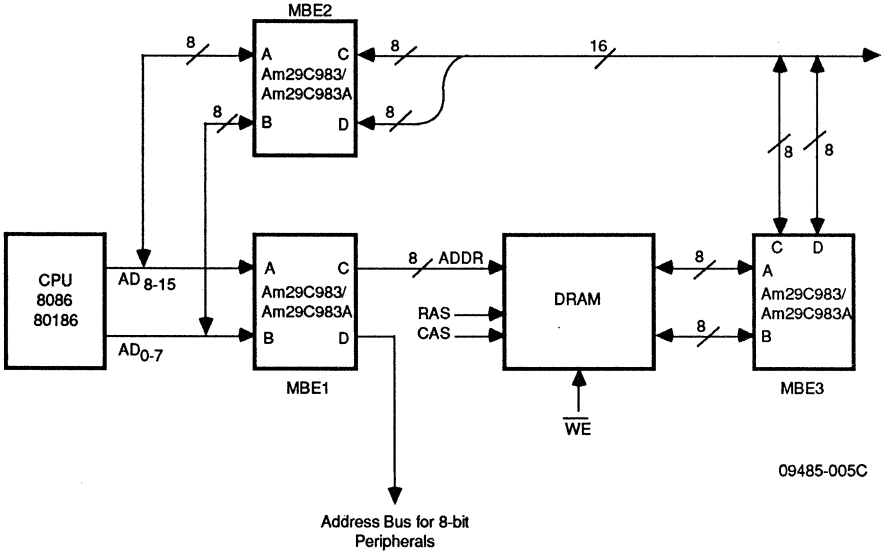


Figure 1. Microprocessor-Memory Interface

Note:
The Am29C983 has 9-bit wide data paths, so parity can be carried along the data.

Digital BOXCAR Integrator

This application illustrates use of the Am29C983 Multiple Bus Exchange (MBE) in a digital "boxcar" integrator. This integrator uses repetitive sampling and a background subtraction scheme to obtain accurate readings of periodic signals in the presence of noise (see Figure 2).

The Am29C983 MBEs are used to route and store a running sum of digitized data, signal as well as background. B port Input latches can be used for signal, and A port Input latches for background. Successive summation is performed by routing A or B latch data to the D port. Background subtraction is performed by connecting the A port to the C port, and the B port to the D port, and routing C port data to the ALU via the 2-to-1 multiplexer. The result is then routed via the C port of the Am29C983 MBE to the barrel shifter for integer division to convert running sums to averages. The output of the barrel shifter can be used to drive an output device, such as a pen plotter, after D/A conversion.

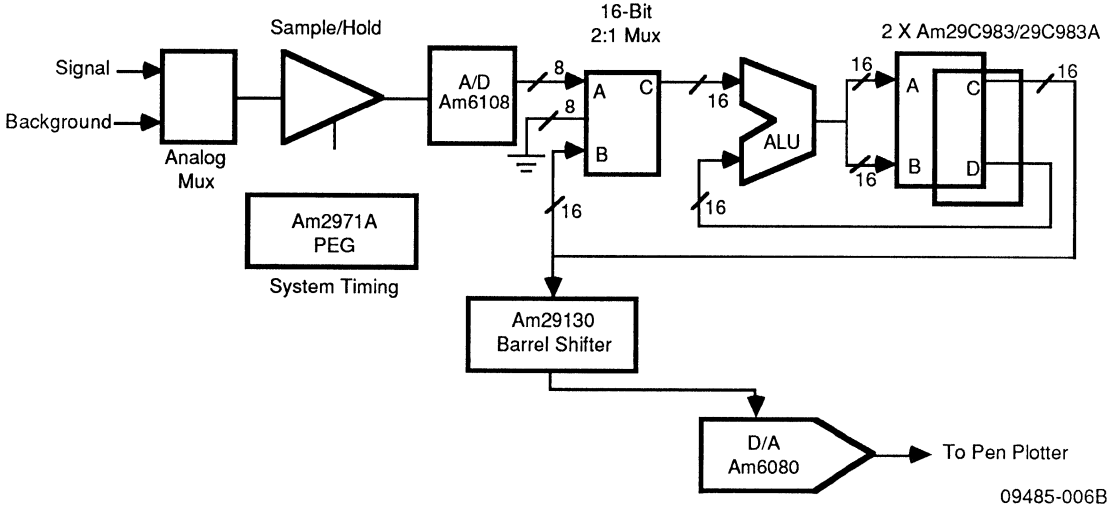
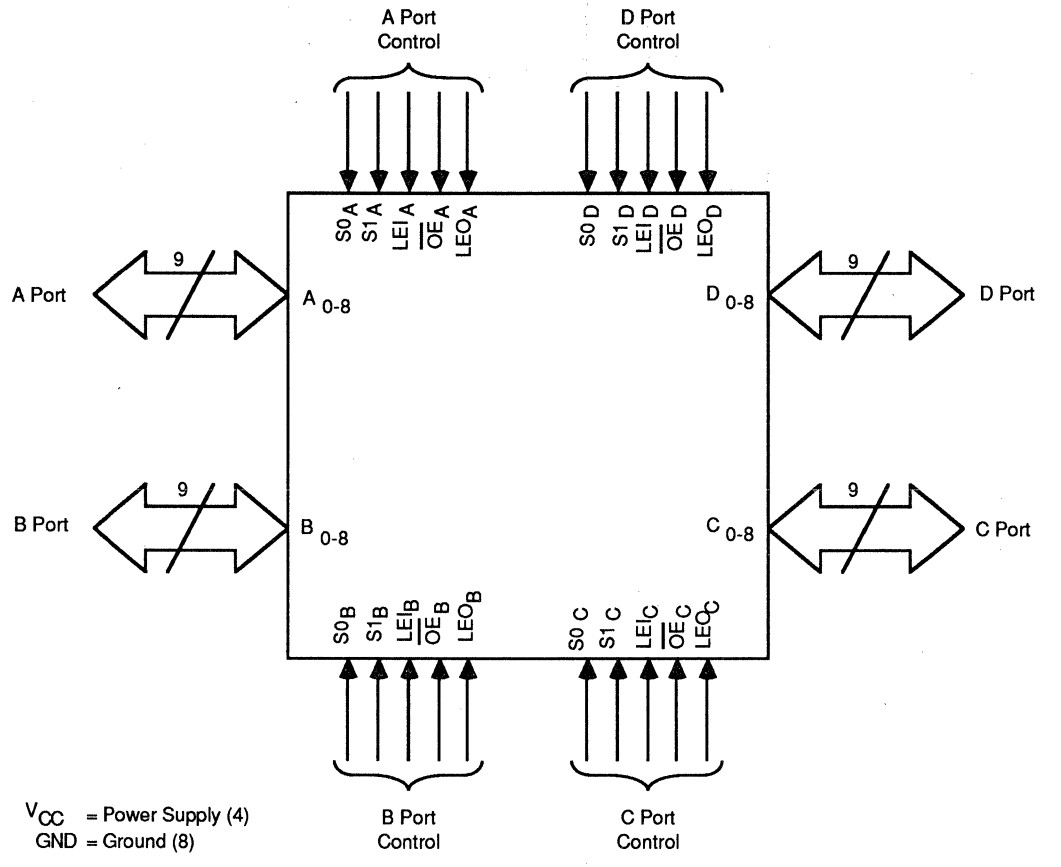


Figure 2. Digital Boxcar Integrator

LOGIC SYMBOL



09485-004A

ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V_{CC})	–0.5 to 6.0 V
DC Input Diode Current	
(I_{IK}) (V_{IN} < 0 V)	–20 mA
(V_{IN} > V_{CC} if applicable)	+20 mA
DC Input Voltage (V_{IN})	–0.5 to V_{CC} +0.5 V
DC Output Diode Current	
(I_{OK}) (V_{OUT} < 0 V)	–50 mA
(V_{OUT} > V_{CC} if applicable)	+50 mA
DC Output Current per Output Pin:	
I_{SINK}	+70 mA
I_{SOURCE}	–30 mA
DC Output Voltage (V_{OUT})	–0.5 to 0.7 V
Total DC Ground Current (I_{GND})	1750 mA
Total DC V_{CC} Current (I_{CC})	575 mA
Storage Temperature	–65 to +150°C

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V

Military (M) Devices

Temperature (T_A)	–55 to +125°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating range unless otherwise specified (for APL products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

AM29C983

Parameter Symbol	Parameter Description	Test Conditions		Min	Max	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = 4.5 V V_{IN} = V_{IL} or V_{IH}	I_{OH} = –15 mA	2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = 4.5 V V_{IN} = V_{IL} or V_{IH}	COM'L: I_{OL} = 48 mA MIL: I_{OL} = 32 mA		0.5	V
V_{IH}	Input HIGH Voltage	(Note 1)		2.0		V
V_{IL}	Input LOW Voltage	(Note 1)			0.8	V
V_{IC}	Input Clamp Voltage	V_{CC} = 4.5 V, I_{IN} = –18 mA			–1.2	V
I_{IL}	Input LOW Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 0 V			–10	µA
		V_{CC} = 5.5 V, V_{IN} = 0.4 V			–5	µA
I_{IH}	Input HIGH Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 2.7 V			5	µA
		V_{CC} = 5.5 V, V_{IN} = 5.5 V			10	µA
I_{OZL}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 0.4 V			–15	µA
		V_{CC} = 5.5 V, V_{OUT} = 0 V			–20	µA
I_{OZH}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 2.7 V			15	µA
		V_{CC} = 5.5 V, V_{OUT} = 5.5 V			20	µA
I_{SC}	Output Short-Circuit Current	V_{CC} = 5.5 V, V_{OUT} = 0 V (Note 2)		–60		mA
I_{CCQ}	Quiescent Power Supply Current (Note 4)	V_{CC} = 5.5 V, V_{IN} = 5.5 V or GND Outputs Open	MIL		1.5	mA
			COM'L		1.2	mA

DC CHARACTERISTICS (Continued)

Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Unit
I_{CCT}	Power Supply Current TTL Input HIGH (Note 4)	V_{CC} = 5.5 V, V_{IN} = 3.4 V Other Inputs at V_{CC} or GND	MIL	1.5	mA/ Input
			COM'L	1.3	
I_{CCD}†	Dynamic Power Supply Current (Note 4)	V_{CC} = 5.5 V, Outputs Open One Output Toggling (Note 3)		500	µA/ MHz/Bit

AM29C983A

Parameter Symbol	Parameter Description	Test Conditions		Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IL} or V_{IH}	I_{OH} = –15 mA	2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IL} or V_{IH}	COM'L: I_{OL} = 48 mA MIL: I_{OL} = 32 mA		0.5	V
V_{IH}	Input HIGH Voltage	(Note 1)		2.0		V
V_{IL}	Input LOW Voltage	(Note 1)			0.8	V
V_{IC}	Input Clamp Voltage	V_{CC} = 4.5 V, I_{IN} = –18 mA			–1.2	V
I_{IL}	Input LOW Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 0 V			–10	µA
		V_{CC} = 5.5 V, V_{IN} = 0.4 V			–5	µA
I_{IH}	Input HIGH Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 2.7 V			5	µA
		V_{CC} = 5.5 V, V_{IN} = 5.5 V			10	µA
I_{OZL}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 0.4 V			–15	µA
		V_{CC} = 5.5 V, V_{OUT} = 0 V			–20	µA
I_{OZH}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 2.7 V			15	µA
		V_{CC} = 5.5 V, V_{OUT} = 5.5 V			20	µA
I_{SC}	Output Short-Circuit Current	V_{CC} = 5.5 V, V_{OUT} = 0 V (Note 2)		–60		mA
I_{CCQ}	Quiescent Power Supply Current (Note 4)	V_{CC} = 5.5 V, V_{IN} = 5.5 V or GND Outputs Open	MIL		1.5	mA
			COM'L		1.2	mA
I_{CCT}	Power Supply Current TTL Input HIGH (Note 4)	V_{CC} = 5.5 V, V_{IN} = 3.4 V Other Inputs at V_{CC} or GND	MIL		1.5	mA/ Input
			COM'L		1.3	
I_{CCD}†	Dynamic Power Supply Current (Note 4)	V_{CC} = 5.5 V, Outputs Open One Output Toggling (Note 3)			500	µA/ MHz/Bit

Notes:
1. Input thresholds are tested in combination with other DC parameters or by correlation.
2. Not more than one output shorted at a time. Duration of short-circuit test not to exceed 100 milliseconds.
3. Measured at a frequency of < 10 MHz with 50% duty cycle. Unused inputs are at V_{CC} or GND.
4. Calculation of total device I_{CC}: $I_{CC} = I_{CCQ} + I_{CCT} \times M_T \times D_H + I_{CCD} \times ((C_L + 91) + 91) \times f \times N$

   Where C_L = Load Capacitance in pF per output
   f = Frequency in MHz
   N = Average number of outputs switching
   M_T = Number of inputs at logic HIGH
   D_H = Duty cycle for each input HIGH

†Not included in Group A tests.

SWITCHING CHARACTERISTICS over operating range unless other specified (for APL products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted)

AM29C983

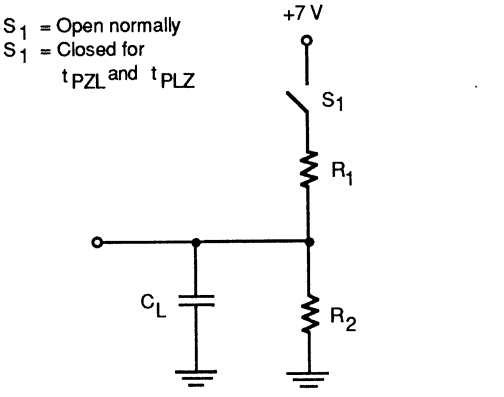
No.	Parameter Symbol	Parameter Description	Test Conditions	COM'L.		MIL.		Unit
				Min	Max	Min	Max	
1	t_{PLH}	Propagation Delay Port to Port	C_L = 50 pF R_1 = 500 Ohms R_2 = 500 Ohms	4	14	3	16	ns
2	t_{PHL}	LEI = HIGH, LEO = HIGH		4	14	3	16	ns
3	t_{PLH}	Propagation Delay Select Input to Port LEO = HIGH		5	18	4	20	ns
4	t_{PHL}			5	18	4	20	ns
5	t_{PLH}	Propagation Delay LEI to Port		5	18	4	20	ns
6	t_{PHL}	LEO = HIGH		5	18	4	20	ns
7	t_{PLH}	Propagation Delay		3	14	3	16	ns
8	t_{PHL}	LEO to Port		3	14	3	16	ns
9	t_{PZH}	Output Enable Time		1	14	1	16	ns
10	t_{PZL}	$\overline{OE}$ to Port		2	14	2	16	ns
11	t_{PHZ}	Output Disable Time		0	12	0	14	ns
12	t_{PLZ}	$\overline{OE}$ to Port Data Output		0	12	0	14	ns
13	t_s	Port to LEI Setup		2		3		ns
14	t_h	Port to LEI Hold		3		4		ns
15	t_s	Port to LEO Setup		4.5		5.5		ns
16	t_h	Port to LEO Hold		1.5		2.5		ns
17	t_s	Select to LEO Setup		6		7		ns
18	t_h	Select to LEO Hold		0		1		ns
19	t_s	LEI to LEO Setup		6		7		ns
20	t_h	LEI to LEO Hold		0		1		ns
21	t_{PWH}[†]	LEI, LEO Pulse Width HIGH		6		7		ns

SWITCHING CHARACTERISTICS (Continued)

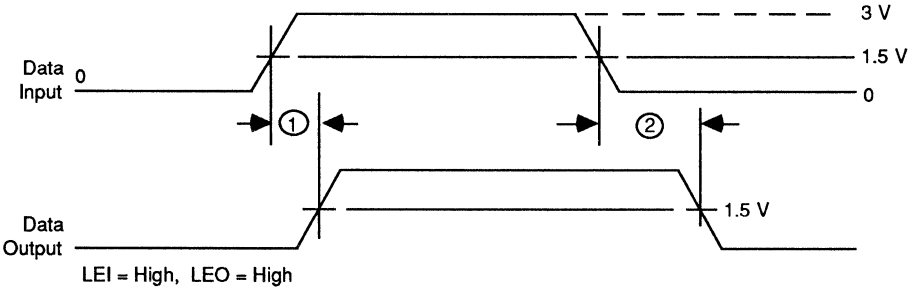
AM29C983A

No.	Parameter Symbol	Parameter Description	Test Conditions	COM'L.		MIL.		Unit
				Min	Max	Min	Max	
1	t_{PLH}	Propagation Delay Port to Port	C_L = 50 pF R_1 = 500 Ohms R_2 = 500 Ohms	3	10	2	12	ns
2	t_{PHL}	LEI = HIGH, LEO = HIGH		3	10	2	12	ns
3	t_{PLH}	Propagation Delay Select Input to Port LEO = HIGH		3	11	3	13	ns
4	t_{PHL}			4	11	3	13	ns
5	t_{PLH}	Propagation Delay LEI to Port		2	12	2	14	ns
6	t_{PHL}	LEO = HIGH		3	12	2	14	ns
7	t_{PLH}	Propagation Delay		2	10	2	12	ns
8	t_{PHL}	LEO to Port		3	10	2	12	ns
9	t_{PZH}	Output Enable Time		1	10	1	12	ns
10	t_{PZL}	$\overline{OE}$ to Port		1	10	1	12	ns
11	t_{PHZ}	Output Disable Time		0	9	0	11	ns
12	t_{PLZ}	$\overline{OE}$ to Port Data Output		0	9	0	11	ns
13	t_s	Port to LEI Setup		2		3		ns
14	t_h	Port to LEI Hold		3		4		ns
15	t_s	Port to LEO Setup		4.5		5.5		ns
16	t_h	Port to LEO Hold		1.5		2.5		ns
17	t_s	Select to LEO Setup		6		7		ns
18	t_h	Select to LEO Hold		0		1		ns
19	t_s	LEI to LEO Setup		6		7		ns
20	t_h	LEI to LEO Hold		0		1		ns
21	t_{PWH} †	LEI, LEO Pulse Width HIGH		6		7		ns

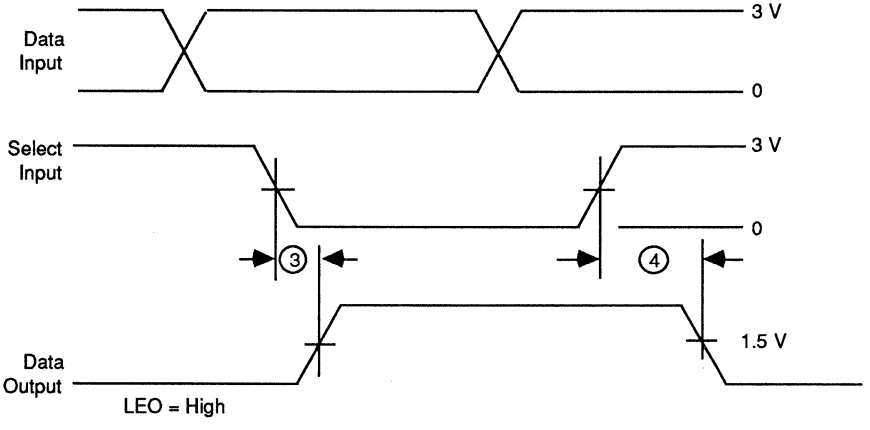
SWITCHING TEST CIRCUIT



SWITCHING TEST WAVEFORMS



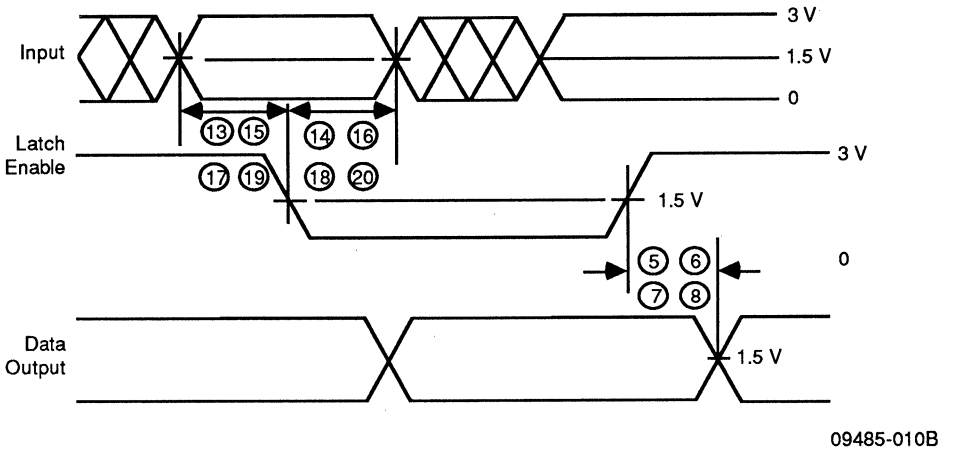
Propagation Delay—Port-to-Port 09485-008B



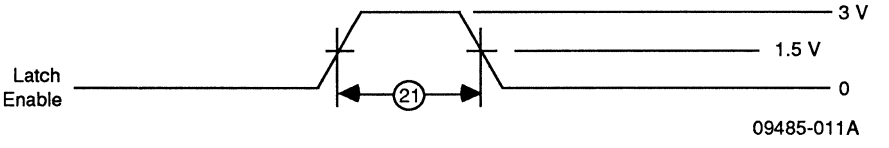
09485-009B

Propagation Delay—Select-to-Port

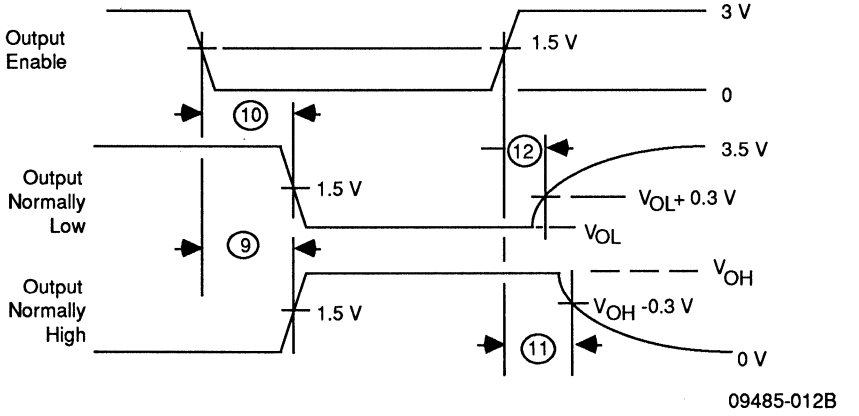
SWITCHING TEST WAVEFORMS (Continued)



Input and Output Latch Propagation Delay, Setup and Hold Times



Minimum Latch Enable Pulse Width



Enable and Disable Times

Preliminary

Am29C985

9-Bit x 4-Port Multiple Bus Exchange with Parity

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- **Four bidirectional I/O ports**
  - Replaces several bidirectional latched transceivers
  - Permits multiple bus communication
  - Allows two independent communication channels
  - TTL compatibility
- **9 bit-wide ports to handle byte parity**
- **Parity check/generate at all ports**
  - Odd parity
- **Additional output bus check**
  - Compares bus with driver inputs
- **Two selection inputs per port**
  - Independent port interconnect control
  - Increased flexibility in data routing
- **Matched port decoding**
  - Simplifies external decode logic
  - Easily cascadable for wider buses
- **Power-Up/Down disable**
  - No power-up sequencing needed
  - Ideal for card-edge interface
- **48 mA output drive**
  - High-capacitance bus driving
- **High-performance CMOS**
  - Low stand-by power consumption
  - 6 ns (typ.) port-to-port delay
  - 7 ns (typ.) select-to-port delay
- **Available in 68-pin PLCC, LCC and PGA packages**
  - Significant savings in board space
- **Proprietary output circuit minmizes ground bounce**
- **3-State during power off condition**

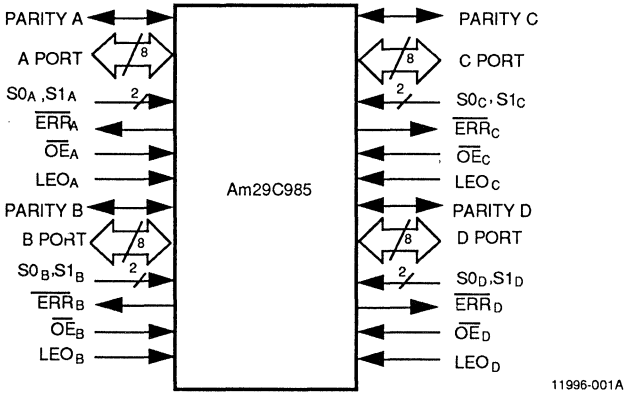
GENERAL DESCRIPTION

The Am29C985 is a high-speed Multiple Bus Exchange device. It is organized as four 9-bit wide TTL-compatible I/O ports with Output Enable control for each port. Any port can serve either as a source (Input) port or as a destination (Output) port. When the output drivers of a port are disabled (high-impedance state), the port serves as a source port. When the drivers are enabled, the port serves as a destination port. Source port selection is made by two independent Select inputs at each port. This organization offers flexibility in implementing the Am29C985 as a digital cross-point switch for multiple bus communication in a multiprocessing environment.

The Am29C985 incorporates parity check and generation capabilities on all four output ports. Each output port is capable of generating odd parity on byte-wide input data. Accordingly, parity check is accomplished at each output on incoming 9 bit parity data. A unique comparison scheme also performs a bus check by comparing the data driven onto the bus with the input data received at the internal multiplexers thus detecting stuck bus bits.

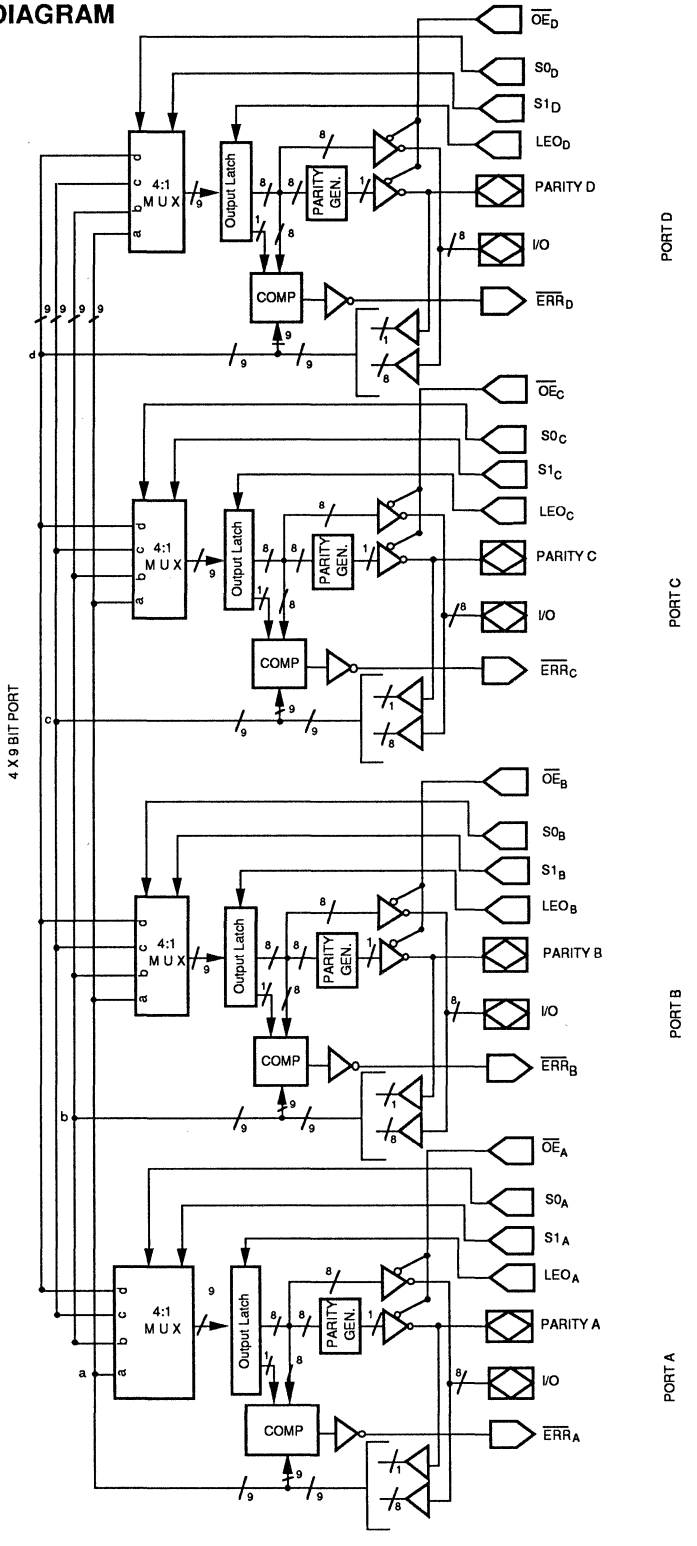
Each I/O port has an output latch to capture outgoing data. All output latches are independently controlled by active HIGH Output Latch Enable inputs. This feature can be used to perform stored operation for byte-word compression and expansion to communicate between buses of different widths. Independent port control permits cascading of Am29C985s for wide buses. All I/O ports go into high impedance state upon power down. This feature makes the device ideally suited for card-edge applications.

SIMPLIFIED BLOCK DIAGRAM



Publication # 11996 Rev. A Amendment /0
Issue Date: January 1990

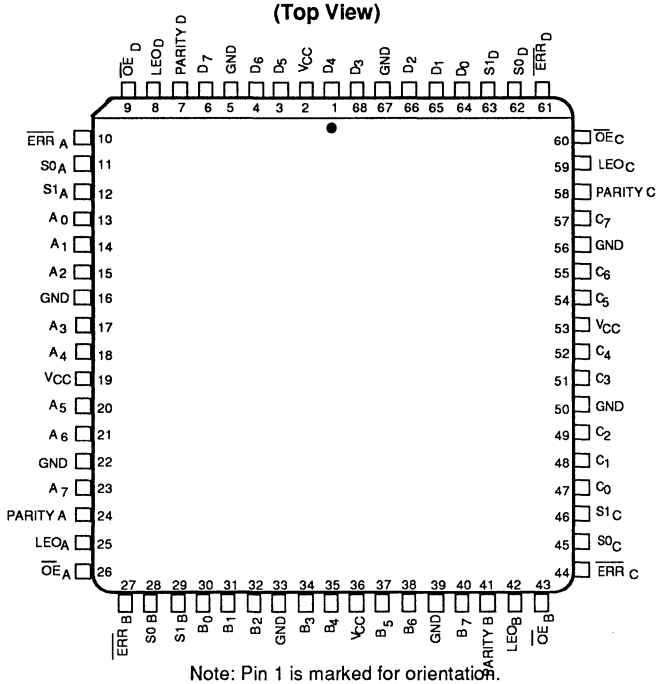
DETAILED BLOCK DIAGRAM



CONNECTION DIAGRAMS

PLCC*

(Top View)



11996A-003A

Note: Pin 1 is marked for orientation.

*Also available in 68-pin LCC; pinout identical to PLCC.

PGA

(Bottom View)

	A	B	C	D	E	F	G	H	J	K	L
1		PARITY D	GND	D_5	D_4	GND	D_2	D_0	$S0_D$	$\overline{OE}_C$	
2	$\overline{OE}_D$	LEO_D	D_7	D_6	V_{CC}	D_3	D_1	$S1_D$	$\overline{ERR}_D$	LEO_C	C_8
3	$S0_A$	$\overline{ERR}_A$								C_7	GND
4	A_0	$S1_A$							PARITY C		C_5
5	A_2	A_5								C_4	V_{CC}
6	A_3	GND								C_3	GND
7	A_4	V_{CC}								C_1	C_2
8	A_5	A_6								$S1_C$	C_0
9	GND	A_7								$\overline{ERR}_C$	$S0_C$
10	PARITY A	LEO_A	$\overline{ERR}_B$	$S1_B$	B_1	B_3	V_{CC}	B_6	B_7	LEO_B	$\overline{OE}_B$
11		$\overline{OE}_A$	$S0_B$	B_0	B_2	GND	B_4	B_5	GND	PARITY B	

Note: Notch indicates orientation.

11996A-004A

PGA PACKAGE

PIN DESIGNATIONS

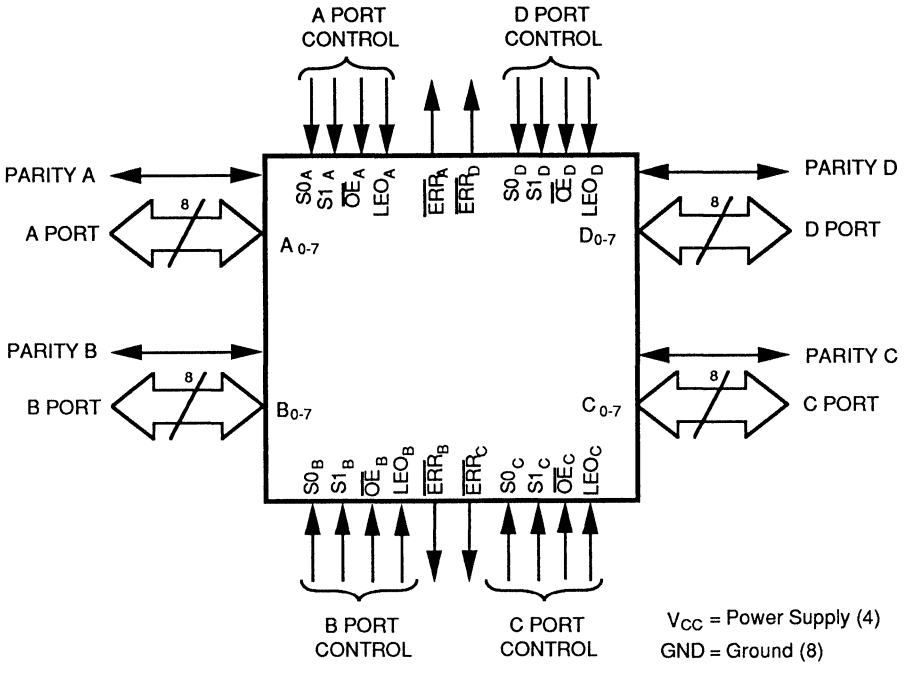
(Sorted by Pin Name)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-4	A_0	K-11	PARITY B	C-2	D_7	B-2	LEO_D
B-5	A_1	L-8	C_0	B-1	PARITY D	B-11	$\overline{OE}_A$
A-5	A_2	K-7	C_1	A-9	GND	L-10	$\overline{OE}_B$
A-6	A_3	L-7	C_2	B-6	GND	K-1	$\overline{OE}_C$
A-7	A_4	K-6	C_3	C-1	GND	A-2	$\overline{OE}_D$
A-8	A_5	K-5	C_4	F-1	GND	A-3	$S0_A$
B-8	A_6	L-4	C_5	F-11	GND	C-11	$S0_B$
B-9	A_7	K-4	C_6	J-11	GND	L-9	$S0_C$
A-10	PARITY A	K-3	C_7	L-3	GND	J-1	$S0_D$
D-11	B_0	L-2	PARITY C	L-6	GND	B-4	$S1_A$
E-10	B_1	H-1	D_0	B-3	$\overline{ERR}_A$	D-10	$S1_B$
E-11	B_2	G-2	D_1	C-10	$\overline{ERR}_B$	K-8	$S1_C$
F-10	B_3	G-1	D_2	K-9	$\overline{ERR}_C$	H-2	$S1_D$
G-11	B_4	F-2	D_3	J-2	$\overline{ERR}_D$	B-7	V_{CC}
H-11	B_5	E-1	D_4	B-10	LEO_A	E-2	V_{CC}
H-10	B_6	D-1	D_5	K-10	LEO_B	G-10	V_{CC}
J-10	B_7	D-2	D_6	K-2	LEO_C	L-5	V_{CC}

(Sorted by Pin Number)

PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME	PIN NO.	PIN NAME
A-2	$\overline{OE}_D$	B-9	A_7	F-10	B_3	K-4	C_6
A-3	$S0_A$	B-10	LEO_A	F-11	GND	K-5	C_4
A-4	A_0	B-11	$\overline{OE}_A$	G-1	D_2	K-6	C_3
A-5	A_2	C-1	GND	G-2	D_1	K-7	C_1
A-6	A_3	C-2	D_7	G-10	V_{CC}	K-8	$S1_C$
A-7	A_4	C-10	$\overline{ERR}_B$	G-11	B_4	K-9	$\overline{ERR}_C$
A-8	A_5	C-11	$S0_B$	H-1	D_0	K-10	LEO_B
A-9	GND	D-1	D_5	H-2	$S1_D$	K-11	PARITY B
A-10	PARITY A	D-2	D_6	H-10	B_6	L-2	PARITY C
B-1	PARITY D	D-10	$S1_B$	H-11	B_5	L-3	GND
B-2	LEO_D	D-11	B_0	J-1	$S0_D$	L-4	C_5
B-3	$\overline{ERR}_A$	E-1	D_4	J-2	$\overline{ERR}_D$	L-5	V_{CC}
B-4	$S1_A$	E-2	V_{CC}	J-10	B_7	L-6	GND
B-5	A_1	E-10	B_1	J-11	GND	L-7	C_2
B-6	GND	E-11	B_2	K-1	$\overline{OE}_C$	L-8	C_0
B-7	V_{CC}	F-1	GND	K-2	LEO_C	L-9	$S0_C$
B-8	A_6	F-2	D_3	K-3	C_7	L-10	$\overline{OE}_B$

LOGIC SYMBOL

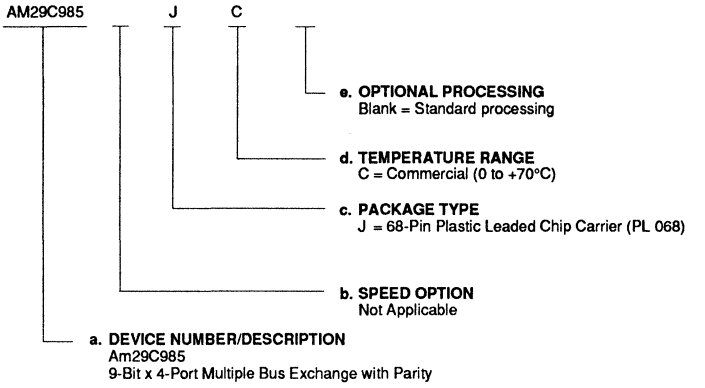


ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. **Device Number**
b. **Speed Option** (if applicable)
c. **Package Type**
d. **Temperature Range**



Valid Combinations

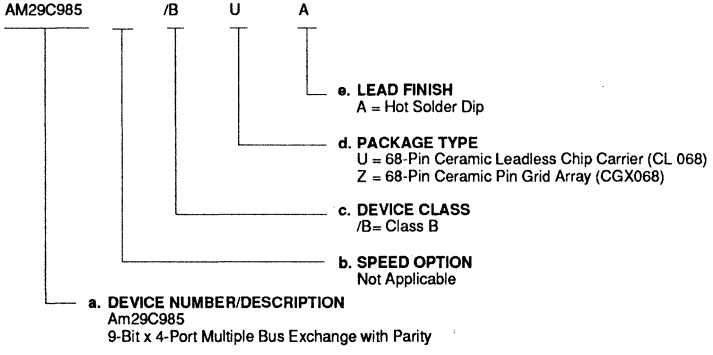
Valid Combinations	
AM29C985	JC

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products

APL Products

AMD products for Aerospace and Defense applications are available in several packages and operating ranges. APL (Approved Products List) products are fully compliant with MIL-STD-883C requirements. The order number (Valid Combination) for APL products is formed by a combination of:

a. **Device Number**
b. **Speed Option** (if applicable)
c. **Device Class**
d. **Package Type**
e. **Lead Finish**



Valid Combinations

Valid Combinations	
AM29C985	/BUA, /BZA

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations or to check on newly released combinations.

Group A Tests

Group A tests consist of Subgroups 1, 2, 3, 7, 8, 9, 10, 11.

PIN DESCRIPTION

A_i, B_i, C_i, and D_i (i = 0 through 7) Data Bus I/O Ports (Input/Output)
These four groups of eight I/O pins are defined as the A, B, C, and D ports respectively. Each port serves as a source (Input) or as a destination (Output).

PARITY A,PARITY B, PARITY C and PARITY D Parity Flag (Input/Output, Three-state)
As an input, parity and port are combined and checked for odd parity. As an output, parity is an active output indicating odd parity for port.

Si_A, Si_B, Si_C, and Si_D (i = 0, 1) Source Port Select (Inputs)
Each pair of inputs determines the source of data for the corresponding I/O port when used as a destination port.

$\overline{ERR}_A$, $\overline{ERR}_B$, $\overline{ERR}_C$ and $\overline{ERR}_D$ ERROR (Output, open drain)
Each output pin is used to flag Parity/Bus errors. Error is indicated by a LOW output.

LEO_A, LEO_B, LEO_C, and LEO_D Output Latch Enable (Inputs; Active HIGH)
Each LEO input controls a 9-bit wide latch on the output side of the corresponding I/O port. The latches are transparent when LEO is HIGH and are latched when LEO is LOW.

$\overline{OE}_A$, $\overline{OE}_B$, $\overline{OE}_C$, and $\overline{OE}_D$ Output Enable (Inputs; Active LOW)
Each $\overline{OE}$ input controls the bus drivers of the corresponding I/O port. When $\overline{OE}$ is LOW, data at the output of the Output latches is passed to the bus. When $\overline{OE}$ is HIGH, the bus outputs are in high-impedance state.

Am29C985 OPERATIONAL DESCRIPTION

Parity and bus checking are provided on the Am29C985. Parity checking and generation are both performed at the output. In order to preserve parity coverage through the part, the data driven onto the bus, including the generated parity, is compared to the data passing through the multiplexer, including the old parity. This has two effects: The comparison of the parity bits acts as a parity check. Also bus errors will be detected if the bus data does not agree with the data being driven.

Minimization of Ground Bounce through Output Edge-Rate Control

The Am29C985 incorporates AMD's proprietary edge controlled outputs in order to minimize simultaneous switching noise (ground bounce). By controlling the output transient currents, ground bounce and output ringing have been greatly reduced. A modified AMD output provides a stable, usable voltage level in less time than a controlled output.

Additionally, speed degradation due to increasing number of outputs switching is reduced. Together, these benefits of edge-rate control result in significant increase in system performance despite a minor increase in specified device propagation delay.

Power-Up/Down Disable

The Am29C985 contains a unique power up/down circuit to provide glitch free outputs during power-supply sequencing. This power-up circuit ensures that at low V_{CC} values (typically 0–2.0 V), the outputs are disabled and in 3-state. At V_{CC} values above this threshold, the outputs will remain disabled and not glitch to an active state if the appropriate output-enable inputs are conditioned for 3-state functionality. At V_{CC} values above the disable circuitry threshold, if the output-enable inputs are conditioned active (outputs enabled), the outputs will respond to a steady state input value. Additionally, the outputs will exhibit high impedance characteristics under power conditioning.

Input/Output Structures

Typical CMOS devices on the market today have maximum DC I/O voltage ratings that prevent some card edge applications, due to the uncertainty of the I/O voltage with respect to V_{CC}. This uncertainty occurs when extracting or replacing a card into a powered-on connector or when a powered-off device is sitting on an active bus. Under these conditions, the maximum rating of –0.5 V to V_{CC} + 0.5 V may be violated. This rating is derived from the presence of a parasitic diode from the input or output to V_{CC}. To prevent forward biasing the diode with an active signal, the 0.5 V limit above V_{CC} was adopted.

AMD has addressed this situation with unique input and output structures. These structures on the Am29C985 use an n-channel pull-up transistor. This results in a stacked n-channel output buffer and a proprietary ESD input cell.

These circuit modifications result in a maximum DC I/O voltage rating of –0.5 V to 7.0 V. The maximum rating is no longer a function of the V_{CC} voltage, thus allowing 3-state functionality under power off condition.

In addition, another benefit gained is that the n-channel pull-up reduces the output HIGH-level voltage for a lightly loaded output to 4.0 V, at V_{CC} = 5.0 Volts. This reduces switching noise and cross-talk associated with typical CMOS full rail-to-rail travel.

FUNCTIONAL DESCRIPTION

The Am29C985 Multiple Bus Exchange consists of four 9-bit I/O ports. Each port has a 9-bit output latch to capture outgoing data. There are four control pins associated with each port: two Select inputs for source port selection, one Output Latch Enable input (active HIGH) to control Output latches, and an active LOW Output Enable line to control the bus driver at the I/O port.

Port Selection and Control

Each port is independently controlled by these four control inputs. If the output drivers of a port are disabled (high-impedance state), that port is an input and can be used as a source port. At the same time, the data at one of the four internal buses can be transferred to the Output latch under the control of the appropriate Select inputs. If the output drivers are enabled, the port serves as a destination port, transporting the data at the output of its Output latch to the external bus connected to the I/O port. Independent control of the Output latch permits stored operation at any port.

Parity and Bus checking

In the Am29C985, parity checking and recognition are both performed at the output. To preserve parity coverage through the part, the data driven onto the bus, including the regenerated parity, is compared to the data passing through the switch, including the old parity. This has two effects: the comparison of parity bits acts as a parity check. Also bus errors will be detected if the bus data does not agree with data being driven to the output buffer.

Error Outputs

$\overline{ERR}$ pins are active LOW, open drain outputs. This allows easy combination of multiple bytes. When passing non-parity data through the part the output will have correct odd parity, but an error may be indicated due to the uncertainty of the 9th bit. Under this condition it is up to the user to ignore the error.

Multiple Bus Communication

Four internal buses serve as pathways for port-to-port connection. By proper choice of source select codes for the ports, the Am29C985 can be configured in different modes for multiple bus communication. In one mode of operation, two ports can be selected as source ports and the other two as destination ports; thus, two independent bidirectional communication channels are established. In another mode, one port can be selected as the source, and one or more of the other ports can serve as destination ports. Any port not intended as a destination port can be disabled (high-impedance state) by its Output Enable control.

TRUTH TABLES

A. Port Source Selection

$S1_n$	$S0_n$	Source
L	L	A Bus
L	H	B Bus
H	L	C Bus
H	H	D Bus

B. Output Latch Operation

LEO_n	Mode
H	Transparent
L	Latched

C. I/O Port Controls

LEO_n	$\overline{OE}_n$	I/O	Source of Data
L	L	Out	Contents of Output Latch
H	L	Out	Selected Source Port
X	H	In	

Key: n = A, B, C, or D
L = LOW
H = HIGH
X = Don't Care

ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V_{CC})	–0.5 to 6.0 V
DC Input Diode Current	
(I_{IK}) (V_{IN} < 0 V)	–20 mA
(V_{IN} > V_{CC} if applicable)	+20 mA
DC Input Voltage (V_{IN})	–0.5 to 7.0 V
DC Output Diode Current	
(I_{OK}) (V_{OUT} < 0 V)	–50 mA
(V_{OUT} > V_{CC} if applicable)	+50 mA
DC Output Current per Output Pin:	
I_{SINK}	+70 mA
I_{SOURCE}	–30 mA
DC Output Voltage (V_{OUT})	–0.5 to 7.0 V
Total DC Ground Current (I_{GND})	1750 mA
Total DC V_{CC} Current (I_{CC})	575 mA
Storage Temperature	–65 to +150°C

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices	
Temperature (T_A)	0 to +70°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V
Military (M) and Extended Commercial (E) Devices	
Temperature (T_A)	–55 to +125°C
Supply Voltage (V_{CC})	+4.5 to +5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS over operating range unless otherwise specified (for APL products, Group A, Subgroups 1, 2, 3 are tested unless otherwise noted)

Parameter Symbol	Parameter Description	Test Conditions		Min.	Max.	Unit
V_{OH}	Output HIGH Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IL} or V_{IH}	I_{OH} = –15 mA	2.4		V
V_{OL}	Output LOW Voltage	V_{CC} = 4.5 V, V_{IN} = V_{IL} or V_{IH}	I_{OL} = 48 mA COM'L		0.5	V
			I_{OL} = 32 mA MIL			
V_{IH}	Input HIGH Voltage	(Note 1)		2.0		V
V_{IL}	Input LOW Voltage	(Note 1)			0.8	V
V_{IC}	Input Clamp Voltage	V_{CC} = 4.5 V, I_{IN} = –18 mA			–1.2	V
I_{IL}	Input LOW Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 0 V			–10	µA
		V_{CC} = 5.5 V, V_{IN} = 0.4 V			–5	µA
I_{IH}	Input HIGH Current (Select Inputs)	V_{CC} = 5.5 V, V_{IN} = 2.7 V			5	µA
		V_{CC} = 5.5 V, V_{IN} = 5.5 V			10	µA
I_{OZL}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 0.4 V			–15	µA
		V_{CC} = 5.5 V, V_{OUT} = 0 V			–20	µA
I_{OZH}	Off-State Leakage Current (I/O Ports)	V_{CC} = 5.5 V, V_{OUT} = 2.7 V			15	µA
		V_{CC} = 5.5 V, V_{OUT} = 5.5 V			20	µA
I_{SC}	Output Short-Circuit Current	V_{CC} = 5.5 V, V_{OUT} = 0 V (Note 2)		–60		mA
I_{CCQ}	Quiescent Power Supply Current (Note 4)	V_{CC} = 5.5 V, V_{IN} = 5.5 V or GND Outputs Open	MIL		TBD	mA
			COM'L		TBD	mA
I_{CCT}	Power Supply Current TTL Input HIGH (Note 4)	V_{CC} = 5.5 V, V_{IN} = 3.4 V Other Inputs at V_{CC} or GND	MIL		TBD	mA/ Input
			COM'L		TBD	

DC CHARACTERISTICS (Cont'd.)

Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Unit
I_{CCD}†	Dynamic Power Supply Current (Note 4)	V_{CC} = 5.5 V, Outputs Open One Output Toggling (Note 3)		TBD	µA MHz/Bit
C_{PD}†	Power Dissipation Capacitance	V_{CC} = 5.5 V (Note 5)		TBD	pF/bit

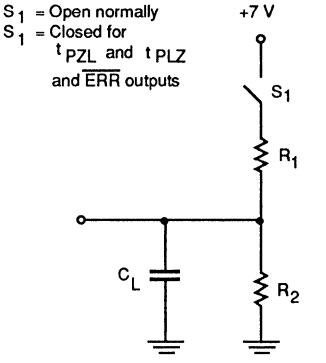
SWITCHING CHARACTERISTICS over operating range unless other specified (for APL products, Group A, Subgroups 9, 10, 11 are tested unless otherwise noted)

No.	Parameter Symbol	Parameter Description	Test Conditions	COM'L.		MIL.		Unit
				Min.	Max.	Min.	Max.	
1	t_{PLH}	Propagation Delay Port to Port	C_L = 50 pF R_1 = 500 Ohms R_2 = 500 Ohms	3	11	2	13	ns
2	t_{PHL}	LEO = HIGH		3	11	2	13	ns
3	t_{PLH}	Propagation Delay Select Input to Port		4	11	3	13	ns
4	t_{PHL}	LEO = HIGH		4	11	3	13	ns
5	t_{PLH}	Propagation Delay Port to Parity		4	14	3	16	ns
6	t_{PHL}	LEO = HIGH		4	14	3	16	ns
7	t_{PLH}	Propagation Delay		3	9	2	11	ns
8	t_{PHL}	LEO to Port		3	9	2	11	ns
9	t_{PZH}	Output Enable Time		3	8	2	10	ns
10	t_{PZL}	$\overline{OE}$ to Port		3	8	2	10	ns
11	t_{PHZ}	Output Disable Time		3	6	2	8	ns
12	t_{PLZ}	$\overline{OE}$ to Port		3	6	2	8	ns
13	t_{PLH}†	Propagation Delay		3	14	2	16	ns
14	t_{PHL}†	Port to $\overline{ERR}$		3	9	2	11	ns
15	t_{PLH}	Propagation Delay		5	15	4	17	ns
16	t_{PHL}	Select to Parity		5	15	4	17	ns
17	t_s	Port to LEO Setup		4.5		5.5		ns
18	t_h	Port to LEO Hold		0		1		ns
19	t_s	Select to LEO Setup		6.0		7.0		ns
20	t_h	Select to LEO Hold		0		1		ns
21	t_{PWH}†	LEO Pulse Width HIGH		3		4		ns

Notes:
1. Input thresholds are tested in combination with other DC parameters or by correlation.
2. Not more than one output shorted at a time. Duration of short-circuit test not to exceed 100 ms.
3. Measured at a frequency of < 10 MHz with 50% duty cycle. Unused inputs are at V_{CC} or GND.
4. Calculation of total device I_{CC}: $I_{CC} = I_{CCQ} + I_{CCT}D_HN_T + I_{CCD}(f_{CP}/2 + f_iN_i)$
   Where: D_H = Duty cycle for each TTL input HIGH
   N_T = Number of inputs at D_H
   f_{CP} = Clock frequency for clocked devices (Zero for non-clocked devices)
   f_i = Input frequency of the i^{th} input
   N_i = Number of inputs at f_i
5. C_{PD} in pF is calculated from I_{CCD} measurements using the formula
   $C_{PD} = I_{CCD}/V_{CC}$
   where I_{CCD} is expressed in µA/MHz/bit.

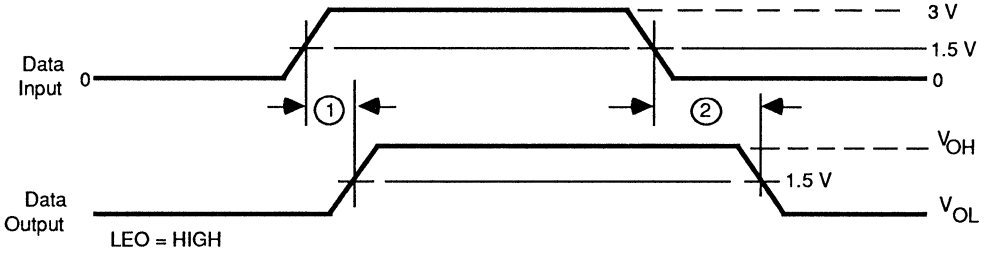
†Not included in Group A tests.

SWITCHING TEST CIRCUIT



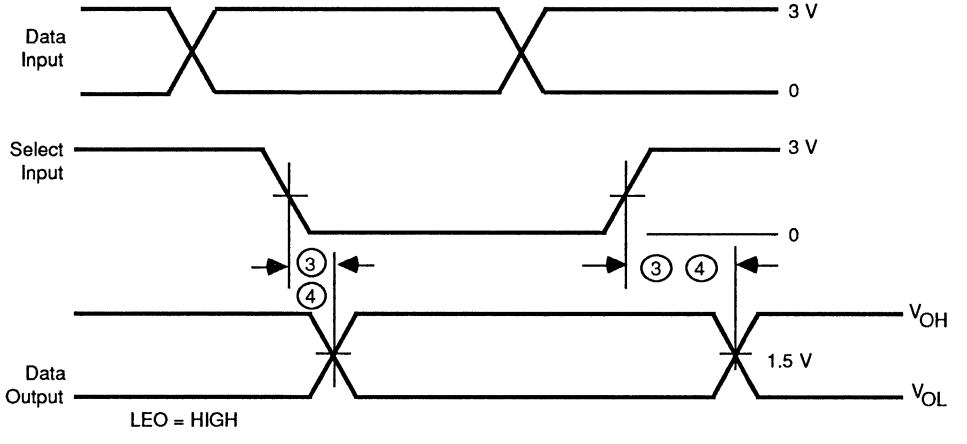
11996-006A

SWITCHING TEST WAVEFORMS



Propagation Delay—Port-to-Port

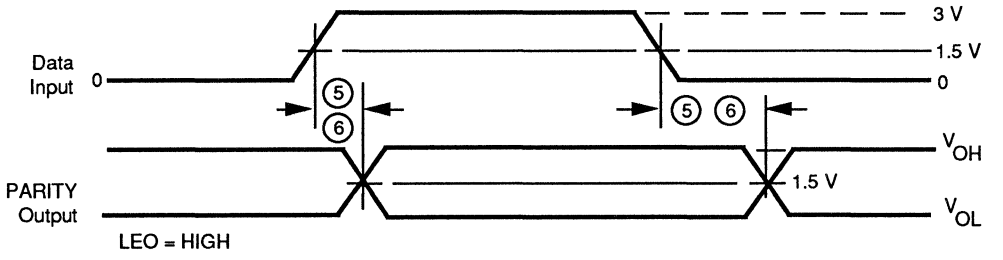
11996-007A



Propagation Delay—Select-to-Port

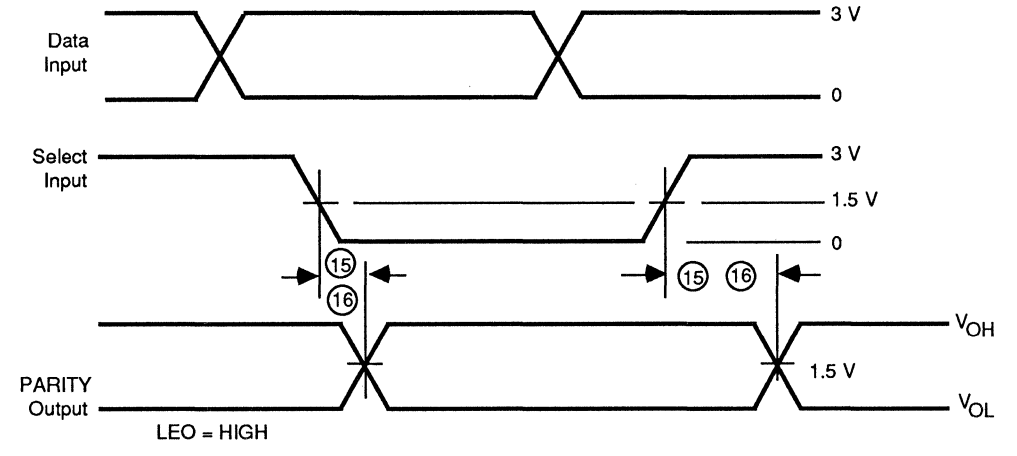
11996-008A

SWITCHING TEST WAVEFORMS (Cont'd.)



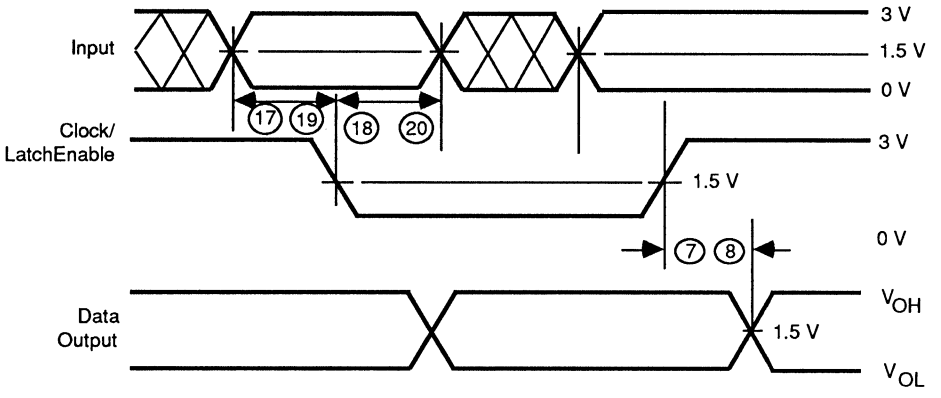
Propagation Delay — Port to Parity

11996-009A

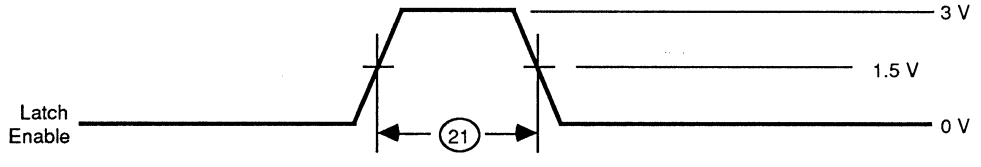


Propagation Delay — Select to Parity

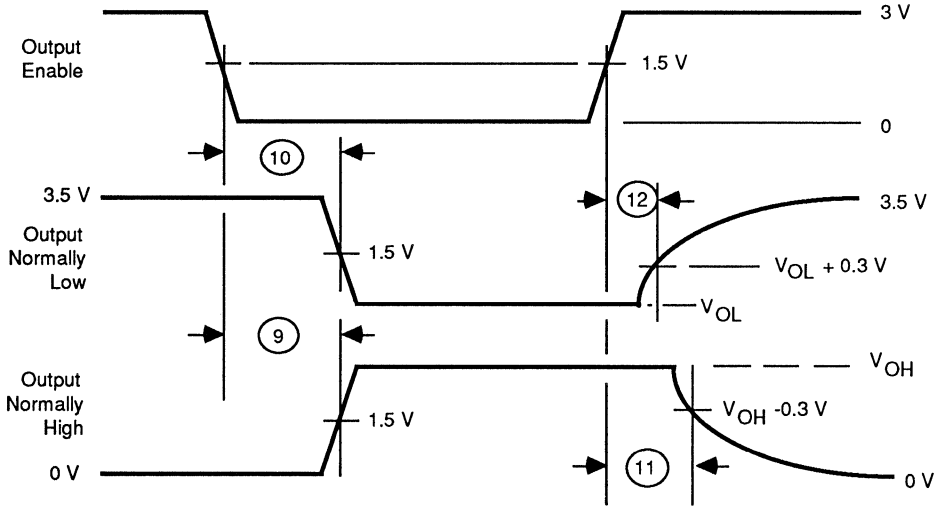
11996-010A



Output Latch Propagation Delay, Setup and Hold Times

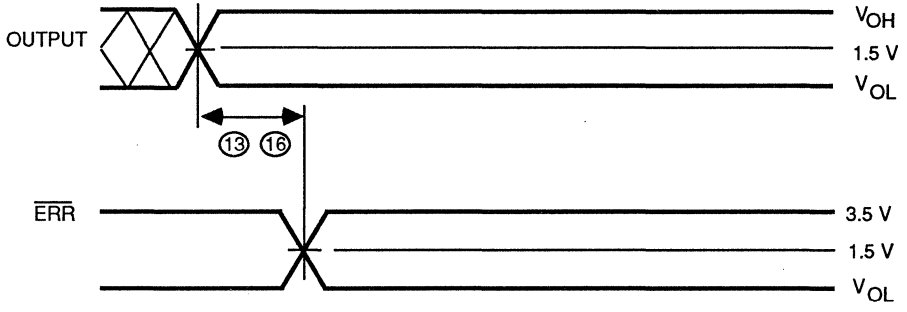


Minimum Latch Enable



Enable and Disable Times

SWITCHING TEST WAVEFORMS (Cont'd.)



11996-014A

Propagation Delay — Output Port to $\overline{ERR}$

Final

673104A

1-Megabit Dynamic RAM Controller/Driver

Advanced Micro Devices

DISTINCTIVE CHARACTERISTICS

- Supports up to 1 M DRAMs
- Capable of addressing up to 16 M bytes
- On-chip capacitive-load drivers capable of driving up to 88 DRAMs with 30-nsec typical address propagation delay and 128 DRAMs with 35-nsec typical address propagation delay
- $\overline{\text{RASIN}}$ to $\overline{\text{RAS}}$ delay of 23 nsec max ($\overline{\text{RAS}}$ driving 32 DRAMs)
- Max and Min skews are specified to simplify system design
- Four $\overline{\text{CASIN}}$ inputs and four $\overline{\text{CAS}}$ outputs simplify byte addressing
- An Auto-Access mode with extended $\overline{\text{CAS}}$ capability takes advantage of full performance of 120- and 150-nsec DRAMs
- An output series resistor reduces undershoot

GENERAL DESCRIPTION

The 673104A 1-Megabit Dynamic RAM Controller/Driver is designed for high performance memory systems. It acts as the address controller between any of today's fast microprocessors and a dynamic memory array. It uses its 10-bit row latch and 10-bit column latch to hold the row and column addresses, respectively, for multiplexing these to any 256K or 1 M DRAM.

The 673104A may directly drive up to 128 DRAMs with its ground bounce controlled outputs. Four separate $\overline{\text{RAS}}_n$ and four separate $\overline{\text{CAS}}_n$ outputs allow addressing up to 16 M bytes.

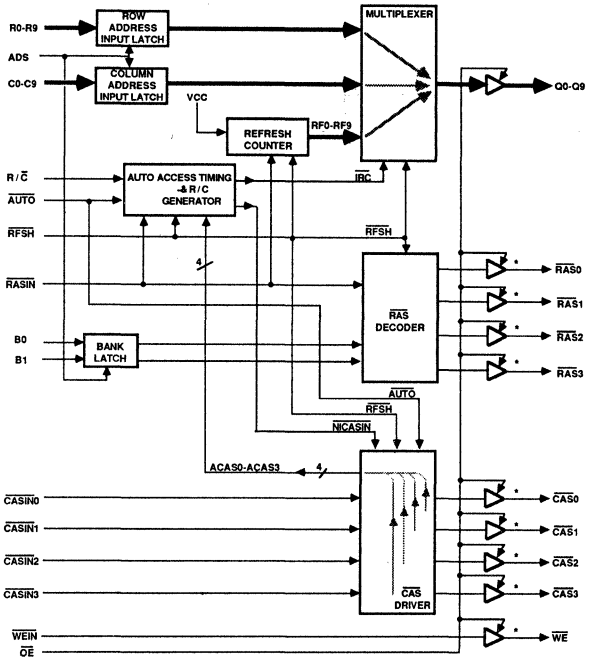
The 673104A has three modes of operation. These are Externally Controlled Access (ECA) in which external $\overline{\text{RASIN}}$, $\overline{\text{CASIN}}_n$, and R/$\overline{\text{C}}$ inputs control read/write accesses; Auto Access (AA) in which the access control signals are generated internally; and Refresh in which successive refresh addresses are generated by an internal refresh counter.

The 673104A is available in an industry standard 68-pin PLCC package.

Publication # 14060 Rev. A Amendment /0
Issue Date: January 1990

PAL® is a registered trademark of Advanced Micro Devices, Inc.

BLOCK DIAGRAM



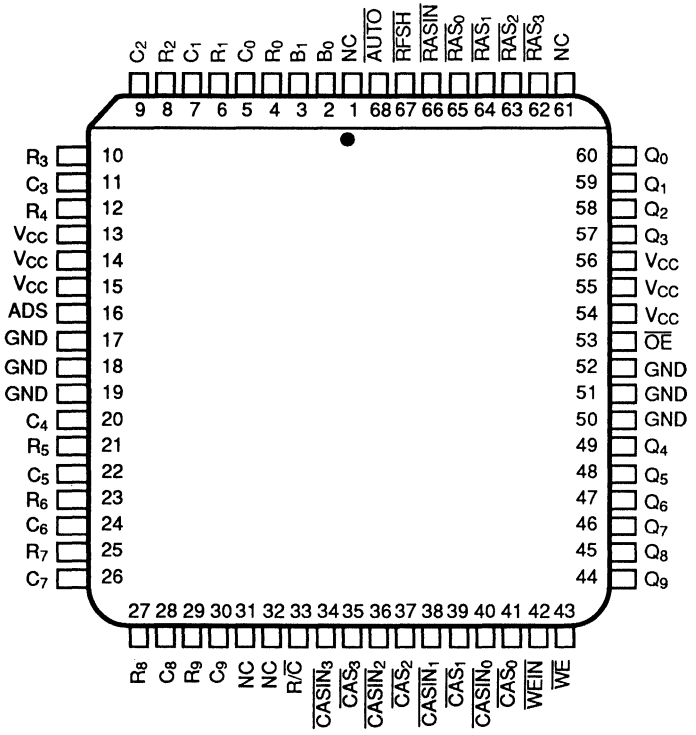
* Indicates that there is a 3-KΩ pull-up resistor on these outputs when they are disabled

Figure 1. 673104A Functional Block Diagram

14060-001A

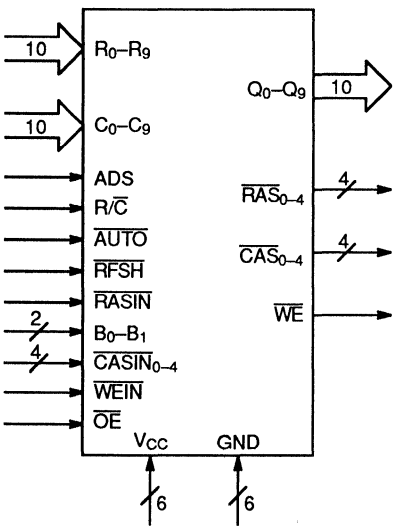
CONNECTION DIAGRAMS

673104ANL
68-Pin PLCC



14060-002A

LOGIC SYMBOL



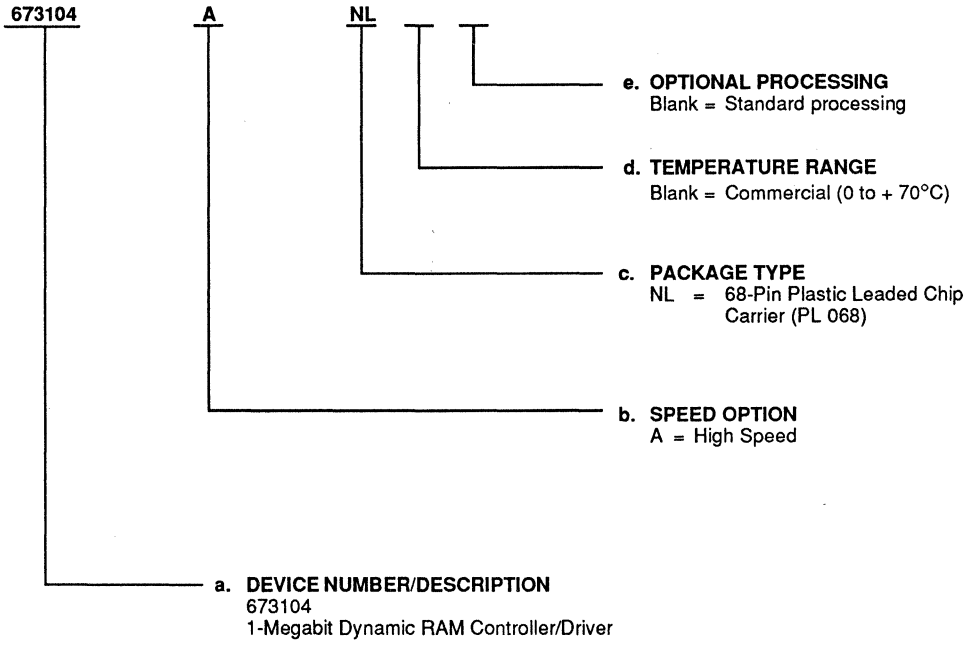
14060-003A

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
673104A	NL

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Only one configuration is available for this device.

PIN DESCRIPTION

V_{CC}, GND

V_{CC} – GND = 5 V ± 10%

The supply pins have been assigned to the center of the package to reduce voltage drops, both DC and AC. A low inductance connection between the ground pin and a solid ground plane will minimize fluctuations in the ground level of the device that may occur when the address outputs switch from HIGH to LOW simultaneously. A 1-µF multilayer ceramic capacitor in parallel with a low voltage tantalum capacitor, both connected close to the V_{CC} and GND pins, will properly decouple the device. All GND and V_{CC} must be connected for proper device operation.

R_0–R_9

Row Address Inputs

C_0–C_9

Column Address Inputs

B_0–B_1

Bank-Pair Select Inputs

Strobed by ADS. Decoded to enable one of the $\overline{RAS}$ outputs when $\overline{RASIN}$ goes LOW in the access modes.

Q_0–Q_9

Multiplexed Address Outputs

Selected from the row address input latch, the column address input latch, or the refresh counter.

$\overline{RASIN}$

Row Address Strobe Input

Drives the selected $\overline{RAS}_m$ output in the access modes and all $\overline{RAS}$ outputs in the Refresh mode.

ADS

Address (Latch) Strobe Input

Strobes input row address, column address, and Bank Select inputs into the respective latches when HIGH; latches on HIGH-to-LOW transition.

$\overline{OE}$

Output Enable

When $\overline{OE}$ is LOW the address and control outputs are enabled. When $\overline{OE}$ is HIGH the address outputs are in high-impedance and the control outputs are pulled HIGH.

R/$\overline{C}$

Row/Column Select Input

In the Externally-Controlled-Access, it is used to select either the row address input latch or the column address input latch onto the address outputs. In the Refresh mode, when $\overline{AUTO}$ is HIGH, it is used to select between the refresh address (R/$\overline{C}$ HIGH) and the column address (R/$\overline{C}$ LOW). When $\overline{AUTO}$ is LOW R/$\overline{C}$ is disabled.

$\overline{CASIN}_{0-3}$

Column Address Strobe Inputs

In the Externally-Controlled-Access mode the $\overline{CASIN}_n$ directly drives $\overline{CAS}_n$ output. In the Auto-Access mode, it is used to enable the corresponding $\overline{CAS}_n$ output (See $\overline{CAS}_{0-3}$ description).

$\overline{WEIN}$

Write Enable Input

$\overline{WE}$

Write Enable Output

$\overline{CAS}_{0-3}$

Column Address Strobe Outputs

In the Externally-Controlled-Access mode the $\overline{CAS}$ outputs follow the $\overline{CAS}$ inputs. In the Auto-Access mode the $\overline{CASIN}$ inputs are used to enable the $\overline{CAS}$ outputs, but the $\overline{CAS}$ outputs are asserted LOW, with proper delay from the $\overline{RAS}$ output, by the $\overline{RASIN}$ signal via the Auto-Access timing generator. In the Auto-Access mode, the $\overline{CAS}_n$ goes HIGH only when the corresponding $\overline{CASIN}_n$ goes HIGH. Extending the $\overline{CAS}_n$ LOW duration while $\overline{RASIN}$ and $\overline{RAS}_n$ go HIGH satisfies the precharge requirement of the dynamic RAMs.

$\overline{RAS}_{0-3}$

Row Address Strobe Outputs

When $\overline{RFSH}$ is HIGH the selected row address strobe output (decoded from signal B_0, B_1) follows the $\overline{RASIN}$ input. When $\overline{RFSH}$ is LOW all $\overline{RAS}$ outputs go LOW together following $\overline{RASIN}$ going LOW.

$\overline{AUTO}$

Auto-Access Input

When $\overline{AUTO}$ is LOW the Auto-Access mode is selected (see Auto-Access mode description).

$\overline{RFSH}$

Refresh Input

When $\overline{RFSH}$ is LOW the Refresh mode is selected (see Refresh mode description).

FUNCTIONAL DESCRIPTION

The 673104A is an LSI device, provided in a 68-pin package, which performs most of the functions needed to control and address Dynamic RAMs. Twenty-two address inputs, ten address outputs, four $\overline{RAS}$ outputs, and four $\overline{CASIN}$-$\overline{CAS}$ input-output pairs allow the 673104A to directly address 16 M bytes. The four $\overline{CASIN}_n$ inputs and the four $\overline{CAS}_n$ outputs simplify individual byte access in 32-bit wide memory arrays (see Figure 2).

The 673104A has three operating modes:

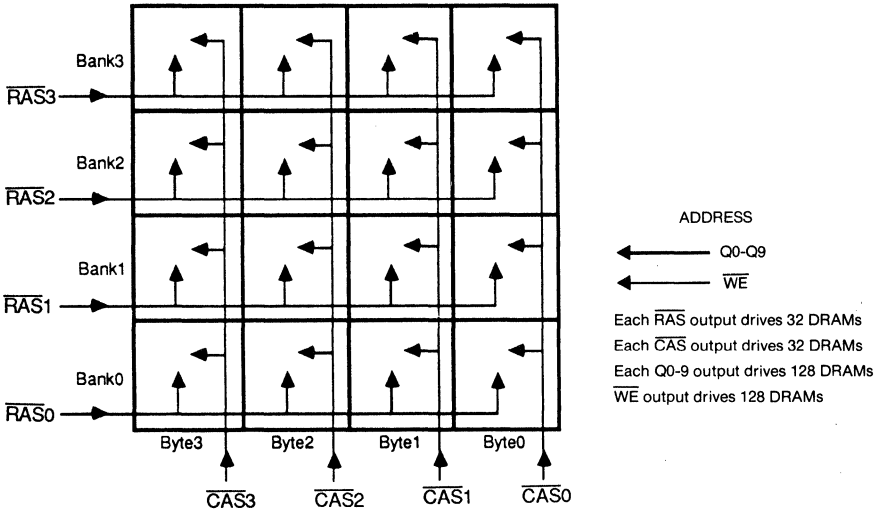
- Externally Controlled Access (ECA)
- Auto-Access (AA)
- Refresh (RFSH)

The Externally-Controlled-Access mode gives the system direct control over the $\overline{RAS}_n$ outputs, the $\overline{CAS}_n$ outputs, and Row/Column multiplexing. It also supports PAGE mode access, NIBBLE mode access and static column mode access.

The Auto-Access mode provides on-chip delays that automatically control the timing delays between $\overline{RAS}_n$ signals, address multiplexing, and $\overline{CAS}_n$ signals. In the Auto-Access mode $\overline{CASIN}_{0-3}$ inputs serve as enables for the respective $\overline{CAS}_{0-3}$ outputs, allowing the access of any byte of the memory array (for 32-bit wide memory arrays organized in four bytes). In this mode $\overline{CAS}_{0-3}$ outputs go HIGH only when the respective $\overline{CASIN}_{0-3}$ inputs go HIGH, and the address switches back to row address only when $\overline{CASIN}_{0-3}$ go HIGH. This feature allows extension of the $\overline{CAS}$ LOW time and column address time while $\overline{RASIN}$ and $\overline{RAS}_n$ can go HIGH to satisfy the precharge requirements of the dynamic RAMs.

When the Refresh mode is selected ($\overline{RFSH}$ is LOW) an on-chip refresh counter provides the refresh address; with $\overline{AUTO}$ HIGH and R/$\overline{C}$ LOW the column address is forced onto the address output multiplexer, facilitating an access of a particular memory location while refreshing a row. This feature may be useful when implementing error detection and correction scrubbing.

The 673104A can drive sixteen banks of DRAMs. $\overline{RAS}_n$ control signals are used to select four banks, while leaving the other twelve banks in standby. The four $\overline{CAS}_n$ outputs enable the selection of one bank out of four. The address lines and the $\overline{WE}$ signal can be connected to all sixteen banks. In a 32-bit wide byte-oriented, memory array the $\overline{RAS}_n$ signals select one out of four banks while the $\overline{CAS}_n$ signals select the bytes, as shown in Figure 2.



14060-004A

Figure 2. 673104A Addressing Four Banks of 32-bit Memory Array Organized in Four Bytes

ABSOLUTE MAXIMUM RATINGS

Supply Voltage, V_{CC}	–0.5 to +7.0 V
Storage Temperature Range	–65 to +150°C
Input Voltage	–1.5 to 5.5 V
Output Current	150 mA
Lead Temperature (soldering, 10 seconds)	300°C

Stresses above those listed under Absolute Maximum Ratings may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

OPERATING RANGES

Commercial (C) Devices

Ambient Temperature (T_A)	0 to +75°C
Supply Voltage (V_{CC})	4.5 to 5.5 V

Operating ranges define those limits between which the functionality of the device is guaranteed.

OPERATING CONDITIONS

Parameter Symbol	Parameter Description	Figure/ Number	Min.	Max.	Unit
t_{ASA}	Address setup time to ADS LOW	8/1	18		ns
t_{ADS}	Address strobe pulse width HIGH		26		ns
t_{AHA}	Address hold time from ADS LOW	8/2	10		ns
Externally Controlled Access Parameter					
t_{ADHAR}	ADS LOW hold from $\overline{RASIN}$ HIGH	3/7	0		ns
t_{BSR}	Bank select setup to $\overline{RASIN}$ LOW (ADS = HIGH)	3/8	10		ns
t_{BSH}	Bank select hold from $\overline{RASIN}$ HIGH (ADS = HIGH)	3/9	10		ns
t_{SSR}	Address strobe HIGH setup to $\overline{RASIN}$ LOW (B_0, B_1 STABLE)	3/10	20		ns
t_{AUHE}	$\overline{AUTO}$ hold from $\overline{RASIN}$ HIGH	3/4	55		ns
t_{AUSRC}	$\overline{AUTO}$ HIGH setup to R/$\overline{C}$ LOW	5/1	25		ns
t_{AUHRC}	$\overline{AUTO}$ HIGH hold to R/$\overline{C}$ HIGH	5/2	10		ns
t_{AUSCA}	$\overline{AUTO}$ HIGH setup to $\overline{CASIN}$ LOW	4/1	45		ns
t_{AUHCA}	$\overline{AUTO}$ HIGH hold from $\overline{CASIN}$ HIGH	4/2	0		ns
t_{AUS}	$\overline{AUTO}$ setup to $\overline{RASIN}$ LOW	3/3	0		ns
t_{RFSR}	$\overline{RFSH}$ HIGH setup to $\overline{RASIN}$ LOW (to guarantee t_{ASR} = 0)	3/1	10		ns
t_{RFHR}	$\overline{RFSH}$ HIGH hold from $\overline{RASIN}$ HIGH	3/2	10		ns
Automatic Access Parameter					
t_{ADHAR}	ADS LOW hold from $\overline{RASIN}$ HIGH	3/7	0		ns
t_{BSR}	Bank select setup to $\overline{RASIN}$ LOW (ADS = HIGH)	6/12	10		ns
t_{BSH}	Bank select hold from $\overline{RASIN}$ HIGH (ADS = HIGH)	3/9	10		ns
t_{ASRL}	Address setup to $\overline{RASIN}$ LOW (ADS = HIGH) ($t_{ASRL} = t_{d2}$ max to guarantee t_{ASR} = 0)	6/4	34		ns
t_{AUS}	$\overline{AUTO}$ setup to $\overline{RASIN}$ LOW	6/5	0		ns
t_{RFSR}	$\overline{RFSH}$ HIGH setup to $\overline{RASIN}$ LOW (to guarantee t_{ASR} = 0)	6/6	10		ns
t_{SSR}	Address strobe HIGH to $\overline{RASIN}$ LOW (B_0, B_1 STABLE)	6/7	20		ns
t_{CASR}	$\overline{CASIN}_{0-1}$ setup to $\overline{RASIN}$ LOW	6/8	–30		ns
t_{AUH}	$\overline{AUTO}$ hold from $\overline{RASIN}$ HIGH	6/9	50		ns
t_{AUCH}	$\overline{AUTO}$ LOW hold from $\overline{CASIN}$ HIGH	6/10	0		ns
Refresh Parameter					
t_{AUHRF}	$\overline{AUTO}$ LOW hold from $\overline{RFSH}$ HIGH (R/$\overline{C}$ LOW)	7/1	10		ns
t_{RCSRF}	R/$\overline{C}$ HIGH setup to $\overline{RFSH}$ LOW ($\overline{AUTO}$ HIGH)	7/2	20		ns
t_{AUSRF}	$\overline{AUTO}$ LOW setup to $\overline{RFSH}$ LOW (R/$\overline{C}$ LOW)	7/3	20		ns
t_{RCHRF}	R/$\overline{C}$ HIGH hold from $\overline{RFSH}$ HIGH ($\overline{AUTO}$ HIGH)	7/4	10		ns
t_{RFH}	$\overline{RASIN}$ HIGH during refresh	7/10	30		ns
t_{RFQS}	$\overline{RFSH}$ LOW setup to $\overline{RASIN}$ LOW (to guarantee t_{RFS})	7/14	34		ns

DC CHARACTERISTICS (V_{CC} = 5 V ± 10%, 0°C ≤ T_A ≤ 75°C)

Parameter Symbol	Parameter Description	Test Conditions	Min.	Max.	Unit
V_{IC}	Input clamp voltage	I_{IN} = −18 mA, V_{CC} = Min.		−1.2	V
I_{IH}	Input high current	V_{IN} = 2.7 V, V_{CC} = Max.		50	µA
I_{CTL}	Output load current for $\overline{RAS}$, $\overline{CAS}$, $\overline{WE}$	V_{OUT} = 0.4 V, V_{CC} = Max. Chip deselect		−2.5	mA
I_{IL}	Input low current except for $\overline{RFSH}$	V_{IN} = 0.4 V, V_{CC} = Max.		−250	µA
I_{ILRF}	Input low current for $\overline{RFSH}$	V_{IN} = 0.4 V, V_{CC} = Max.		−500	µA
V_{IL}	Input low threshold (Note 1)			0.8	V
V_{IH}	Input high threshold (Note 1)		2.0		V
V_{OL1}	Output low voltage	I_{OUT} = 1 mA, V_{CC} = Min.		0.5	V
V_{OL2}	Output low voltage	I_{OUT} = 12 mA, V_{CC} = Min.		0.8	V
V_{OH}	Output high voltage	I_{OUT} = −1 mA, V_{CC} = Min.	2.4		V
I_{OH}	Output source current (Note 2)	V_{OUT} = 0.8 V, V_{CC} = Min.	−50		mA
I_{OL}	Output sink current (Note 2)	V_{OUT} = 2.4 V, V_{CC} = Min.	40		mA
I_{OZ}	Three-state output current (address output)	0.4 V ≤ V_{OUT} ≤ 2.7 V V_{CC} = Max., Chip deselect	−50	50	µA
I_{CC}	Supply current	V_{CC} = Max.		280	mA

Notes:

1. These are absolute voltage levels with respect to the ground pins on the device and includes all overshoots due to system or tester noise. Do not attempt to test these values without suitable equipment.
2. This test is provided as a monitor of driver output source and sink current capability. Caution should be exercised in testing this parameter. One output should be tested at a time and test duration should not exceed one second.

SWITCHING CHARACTERISTICS (Note 3)

Parameter Symbol	Externally Controlled Access Parameter	Figure/ Number	Min.	Max.	Unit
t_{RHA}	Row addresses remaining valid from R/$\overline{C}$ LOW	5/3	0		ns
t_{RPDL}	$\overline{RASIN}$ to $\overline{RAS}$ LOW delay	3/11		23	ns
t_{RPDH}	$\overline{RASIN}$ to $\overline{RAS}$ HIGH delay	3/12		33	ns
t_{APD}	Address input to output delay	9/2		60	ns
t_{WPDL}	$\overline{WEIN}$ to $\overline{WE}$ LOW delay			50	ns
t_{WPDH}	$\overline{WEIN}$ to $\overline{WE}$ HIGH delay			45	ns
t_{CPDL}	$\overline{CASIN}$ to $\overline{CAS}$ LOW delay	4/3		28	ns
t_{CPDH}	$\overline{CASIN}$ to $\overline{CAS}$ HIGH delay	4/4		40	ns
t_{RCC}	Column select to column address valid	5/5		50	ns
t_{RCR}	Row select to row address valid	5/6		53	ns
t_{d1}	($\overline{CASIN}$ to $\overline{CAS}$ LOW delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)		−5	10	ns
t_{d2}	(Address input to output delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)			34	ns
t_{d3}	(Address input to output delay)-($\overline{CASIN}$ to $\overline{CAS}$ LOW delay)		0	28	ns
t_{d4}	Skew between address output lines			12	ns
t_{d5}	($\overline{RASIN}$ to $\overline{RAS}$ HIGH delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)		−10	10	ns
t_{d6}	($\overline{CASIN}$ to $\overline{CAS}$ LOW delay)-($\overline{CASIN}$ to $\overline{CAS}$ HIGH delay)		−12	12	ns
t_{SPD}	ADS HIGH to address output valid	9/1		64	ns
t_{RCHC}	Column addresses remaining valid from R/$\overline{C}$ HIGH	5/4	0		ns
t_{d7}	$t_{RPDL} - t_{RHA}$			16	ns
t_{d8}	$t_{RCC} - t_{CPDL}$			27	ns
Auto Access Parameter					
t_{RICL}	$\overline{RASIN}$ to $\overline{CAS}$ LOW delay	6/20		83	ns
t_{RCDL}	$\overline{RAS}$ to $\overline{CAS}$ LOW delay	6/23	30	73	ns
t_{RPDL}	$\overline{RASIN}$ to $\overline{RAS}$ LOW delay	6/13		23	ns
t_{RPDH}	$\overline{RASIN}$ to $\overline{RAS}$ HIGH delay	6/14		33	ns
t_{APD}	Address input to output delay	6/16		60	ns
t_{WPDL}	$\overline{WEIN}$ to $\overline{WE}$ LOW delay			50	ns
t_{WPDH}	$\overline{WEIN}$ to $\overline{WE}$ HIGH delay			45	ns
t_{CPDH}	$\overline{CASIN}$ to $\overline{CAS}$ HIGH delay	6/22		40	ns
t_{RAH}	Row address hold time from $\overline{RAS}$ LOW	6/17	15		ns
t_{d2}	(Address input to output delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)			34	ns
t_{SPD}	ADS HIGH to address output valid	6/19		64	ns

SWITCHING CHARACTERISTICS (Cont'd.)

Parameter Symbol	Auto Access Parameter	Figure/ Number	Min.	Max.	Unit
t_{ASC}	Column address setup to $\overline{CAS}$ LOW	6/21	0		ns
t_{CAHC}	Column address remaining valid from $\overline{CASIN}_{0-3}$ HIGH	6/18	5		ns
t_{ASR}	Row address valid before $\overline{RAS}$ LOW	6/24	0		ns
t_{d5}	($\overline{RASIN}$ to $\overline{RAS}$ HIGH delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)		–10	10	ns
Refresh Parameter					
t_{RFLCT}	$\overline{RFSH}$ LOW to refresh address valid ($\overline{AUTO}$ LOW or R/$\overline{C}$ HIGH)	7/5		50	ns
t_{RFPDL}	$\overline{RASIN}$ LOW to $\overline{RAS}$ LOW delay during refresh	7/11		26	ns
t_{RFPDH}	$\overline{RASIN}$ HIGH to $\overline{RAS}$ HIGH delay during refresh	7/12		38	ns
t_{RFAH}	Refresh address held from $\overline{RASIN}$ HIGH ($\overline{RFSH}$ LOW)	7/7	0		ns
t_{RHNC}	$\overline{RASIN}$ HIGH to new refresh address valid	7/9		72	ns
t_{RFAHR}	Refresh address held from $\overline{RFSH}$ HIGH	7/6	0		ns
t_{RFS}	Refresh address valid to $\overline{RAS}$ LOW (provided t_{RFQS} is satisfied)	7/13	0		ns
t_{RFRDH}	$\overline{RFSH}$ HIGH to $\overline{RAS}$ HIGH (for three banks, $\overline{RASIN}$ = LOW)	7/15		45	ns
t_{d9}	($\overline{RASIN}$ to $\overline{RAS}$ HIGH delay)-($\overline{RASIN}$ to $\overline{RAS}$ LOW delay)		–9	16	ns
Three-State Parameter					
t_{AZL}	$\overline{OE}$ LOW to address output LOW	10/1		50	ns
t_{AZH}	$\overline{OE}$ LOW to address output HIGH	10/2		60	ns
t_{ALZ}	$\overline{OE}$ HIGH to address output Hi-Z from LOW	10/3		35	ns
t_{AHZ}	$\overline{OE}$ HIGH to address output Hi-Z from HIGH	10/4		25	ns
t_{CTZL}	$\overline{OE}$ LOW to control output LOW	10/5		50	ns
t_{CTZH}	$\overline{OE}$ LOW to control output HIGH	10/6		50	ns
t_{CTLZ}	$\overline{OE}$ HIGH to control output Hi-Z from LOW	10/7		35	ns
t_{CTHZ}	$\overline{OE}$ HIGH to control output Hi-Z from HIGH	10/8		30	ns

Note:

3. Output load capacitance is typical for four banks of 32 DRAMs with trace capacitance. The values are Q_{0-9} C_L = 800 pF, $\overline{RAS}_{0-3}$ C_L = 250 pF, $\overline{WE}$ C_L = 800 pF, $\overline{CAS}_{0-3}$ C_L = 300 pF.

Externally-Controlled-Access Mode (ECA)

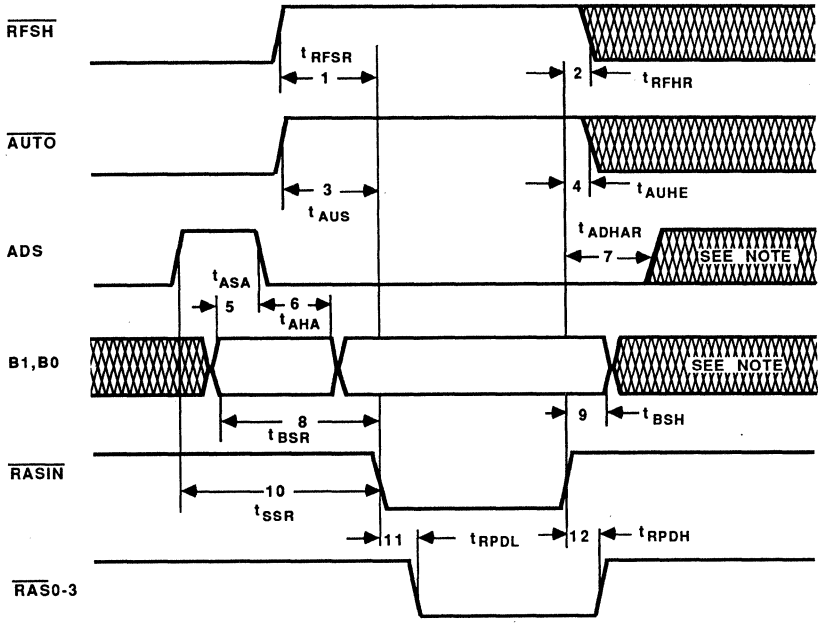
In this mode, selected when $\overline{AUTO}$ and $\overline{RFSH}$ are held HIGH, the 673104A serves as a straightforward multiplexer and driver for the address and control signals to the DRAMs. The $\overline{RAS}_n$ output selected by the B_0 and B_1 inputs follows the $\overline{RASIN}$ input, and each of the $\overline{CAS}$ outputs follows its corresponding $\overline{CASIN}$ input. When R/$\overline{C}$ is HIGH the row address is enabled onto the Q_{0-9} outputs. When R/$\overline{C}$ is LOW the column address latch is enabled onto Q_{0-9} outputs.

The $\overline{RASIN}$ — $\overline{RAS}$, $\overline{CASIN}$ — $\overline{CAS}$, and R/$\overline{C}$ — Q_{0-9} control paths are independent to allow the system designer maximum flexibility and support of special DRAM access and refresh modes such as NIBBLE mode, PAGE mode, $\overline{CAS}$ before $\overline{RAS}$, etc.

To allow tighter timing of the sequence of control signals to the Dynamic RAM, several difference timing parameters have been specified for the Externally Controlled Access mode. These parameters specify the maximum difference between the various "control channels" of the device. In particular, using switching characteristics t_{d7} and t_{d8} is very useful when designing the delay from $\overline{RASIN}$ going LOW to R/$\overline{C}$ and the delay between R/$\overline{C}$ going LOW to $\overline{CASIN}$ going LOW (see Applications).

Bank Select (Strobed by ADS)		Enabled $\overline{RAS}_n$
B_1	B_0	
0	0	$\overline{RAS}_0$
0	1	$\overline{RAS}_1$
1	0	$\overline{RAS}_2$
1	1	$\overline{RAS}_3$

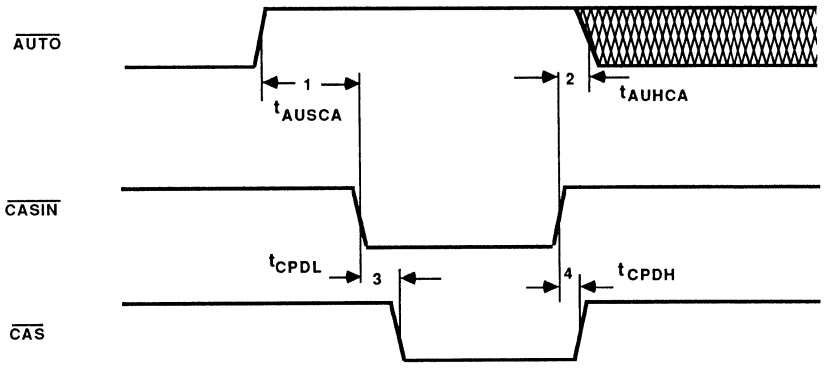
Table 1. Memory Bank Decode



14060-005A

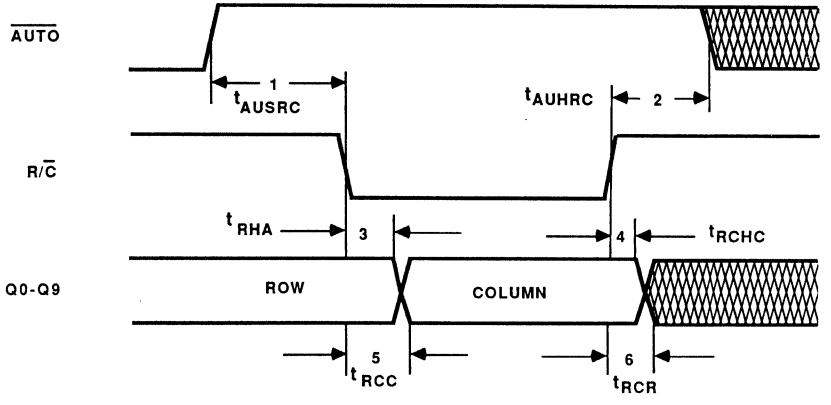
Figure 3. Externally-Controlled-Access – $\overline{RASIN}$-to-$\overline{RAS}$ Timing

Note: To prevent glitches on the $\overline{RAS}_{0-3}$ outputs, operating conditions t_{BSH} or t_{ADHAR} must be satisfied.



14060-006A

Figure 4. Externally-Controlled-Access-$\overline{\text{CASIN}}$ to $\overline{\text{CAS}}$ Timing



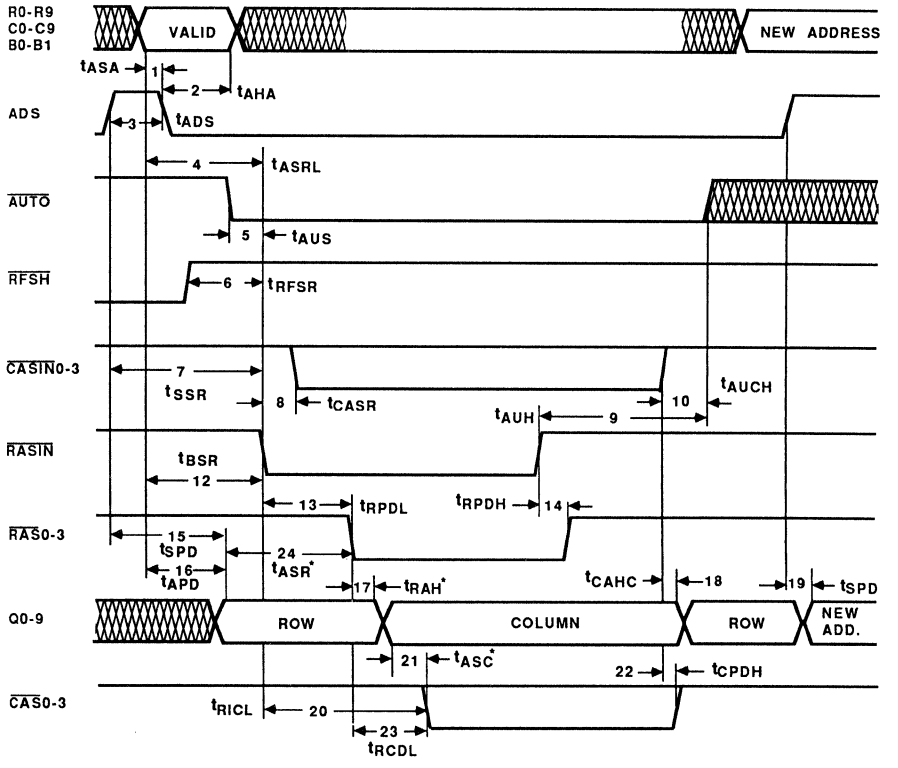
14060-007A

Figure 5. Externally-Controlled-Access R/$\overline{\text{C}}$ Timing

Note: t_{RCC} will be met only if the column address is available t_{APD} before it appears on Q_{0-9} outputs or if it is latched by ADS.

Auto-Access Mode (AA)

In the Auto-Access mode the 673104A provides the system designer with built-in delays and sequencing to accommodate DRAMs with 150 nanoseconds and faster access time. The Auto-Access mode is selected when $\overline{AUTO}$ is held LOW and $\overline{RFSH}$ is held HIGH. The R/$\overline{C}$ input is disabled, and $\overline{RASIN}$ going LOW initiates the sequence of control signals to access the DRAMs. The $\overline{CASIN}_{0-3}$ inputs are used as enables for the respective $\overline{CAS}_n$ outputs. A LOW on a $\overline{CASIN}_n$ input enables the $\overline{CAS}_n$ output to be driven LOW with the internally generated delay from $\overline{RAS}$. Each $\overline{CAS}_n$ output goes HIGH only when the corresponding $\overline{CASIN}_n$ input goes HIGH, and the address switches back to row address only when all $\overline{CASIN}$ go HIGH. This feature allows extension of the $\overline{CAS}$ LOW time and the column address time, while $\overline{RASIN}$ and $\overline{RAS}_m$ can go HIGH to satisfy precharge requirements of the dynamic RAMs. The R/$\overline{C}$ input is disabled in this mode.



* Indicates Dynamic RAM parameters.

14060-008A

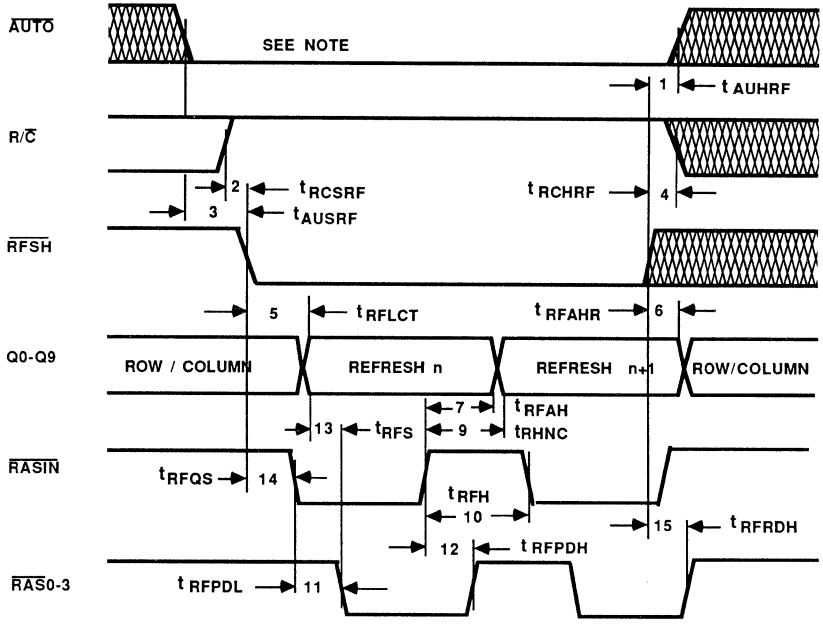
Figure 6. Auto-Access (AA) Timing

Refresh Mode (RFSH)

When $\overline{RFSH}$ is held LOW the refresh counter contents are enabled onto the Q_{0-9} address outputs, provided either R/$\overline{C}$ is held HIGH, or $\overline{AUTO}$ is held LOW, or both conditions exist. In this mode all four $\overline{RAS}$ outputs follow the $\overline{RASIN}$ input signal. The refresh counter increments the refresh address when either $\overline{RASIN}$ or $\overline{RFSH}$ goes HIGH while the other is LOW. When $\overline{AUTO}$ is LOW the $\overline{CASIN}_{0-3}$ inputs are isolated and $\overline{CAS}_{0-3}$ are held HIGH. Also, when $\overline{AUTO}$ is LOW the R/$\overline{C}$ input is isolated from the output multiplexer, and the refresh address appears at the Q_{0-9} outputs.

When $\overline{AUTO}$ is HIGH, pulling R/$\overline{C}$ LOW enables the column address onto the Q_{0-9} outputs. Also, each of the $\overline{CAS}$ outputs follows its respective $\overline{CASIN}$ input. This feature may be used when implementing error correction and detection "scrubbing" for four-bank memory arrays. "Scrubbing" is a term describing cyclic error correction of soft errors in the memory array, done within the refresh cycles. On every refresh cycle one location the memory array is accessed and the data in that location goes, if necessary, through a correction cycle (a read-modify-write memory cycle). The 673104A provides the facilities to force a column address onto the Q_{0-9} address outputs and to assert $\overline{CAS}_{0-3}$ outputs within a refresh cycle to allow scrubbing. A column counter and a bank counter need to added externally to provide the column addresses for scrubbing.

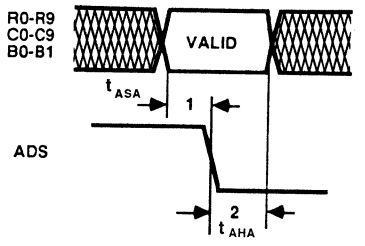
The refresh counter is a 10-bit counter that resets to 0 on power-up and rolls-over to 0 at 1023.



14060-008A

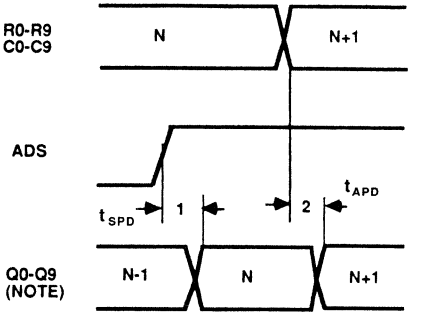
Figure 7. Refresh Timing

Note: In the REFRESH mode, $\overline{AUTO}$ must be LOW or R/$\overline{C}$ must be HIGH to guarantee the refresh address on the Q_{0-9} outputs.



14060-010A

Figure 8. Address Setup and Hold Time to ADS



14060-011A

Figure 9. Address Input/Output Propagation Delay

Note: Row or Column Address ($\overline{RFSH}$ = HIGH).

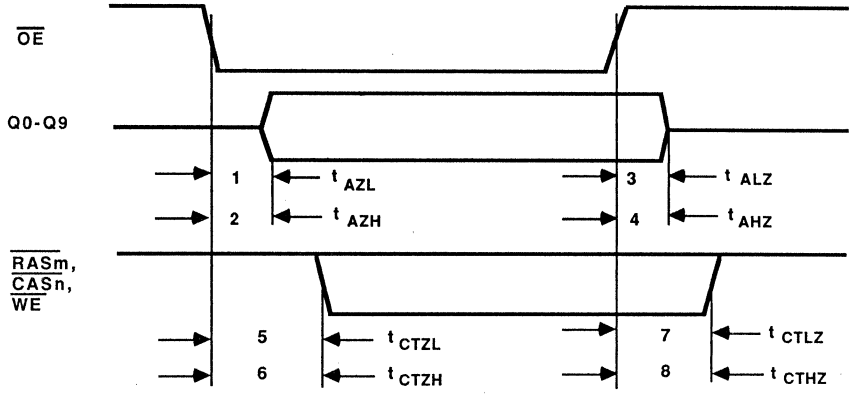


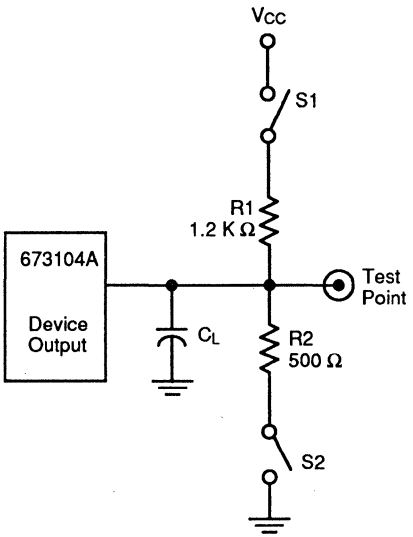
Figure 10. High-Z Timing

KEY TO SWITCHING WAVEFORMS

WAVEFORM	INPUTS	OUTPUTS
	Must Be Steady	Will Be Steady
	May Change from H to L	Will Be Changing from H to L
	May Change from L to H	Will Be Changing from L to H
	Don't Care Any Change Permitted	Changing State Unknown
	Does Not Apply	Center Line is High Impedance "Off" State

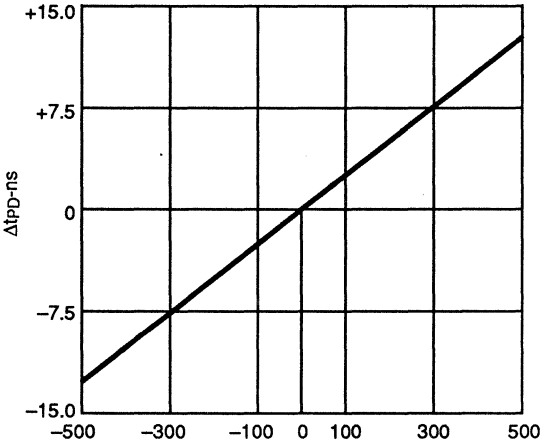
KS000010

SWITCHING TEST CIRCUIT



14060-009A

Note: Input pulse 0 V to 3.0 V, $t_R = t_F = 2.5$ ns, f = 1.0 MHz, $t_{PW} = 200$ns.
Input reference point on AC measurements is 1.5 V.
Output reference points are 2.4 V for HIGH and 0.8 V for LOW.



14060-012A

Change in Propagation Delays vs. Change in Loading Capacitance Relative to the Specified Load

Address Outputs

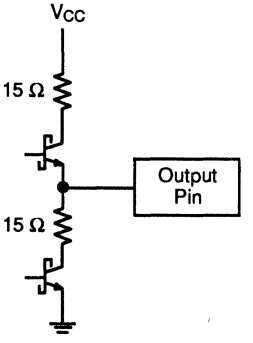
Test	S_1	S_2	C_L	Measured At
t_{PD}	Open	Closed	800 pF	0.8 V, 2.4 V
t_{PZH}	Closed	Closed	800 pF	2.4 V
t_{PHZ}	Open	Closed	15 pF	V_{OH} −0.5 V
t_{PZL}	Closed	Closed	800 pF	0.8 V
t_{PLZ}	Closed	Open	15 pF	V_{OL} +0.5 V

Control Outputs

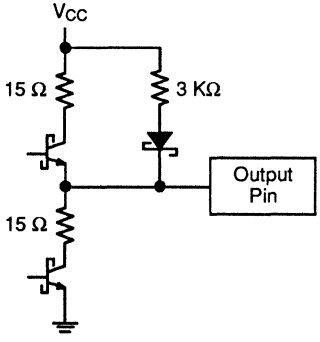
Test	S_1	S_2	C_L	Measured At
t_{PD}	Open	Closed	C_L	0.8 V, 2.4 V
t_{PZH}	Open	Closed	C_L	2.4 V
t_{PHZ}	Open	Closed	15 pF	V_{OH} −0.5 V
t_{PZL}	Open	Open	C_L	0.8 V
t_{PLZ}	Open	Open	15 pF	V_{OL} +0.5 V

Where C_L = 250 pF for $\overline{RAS}$, 300 pF for $\overline{CAS}$, and 800 pF for $\overline{WE}$.

Address Driver Output Stage



Control Driver Output Stage

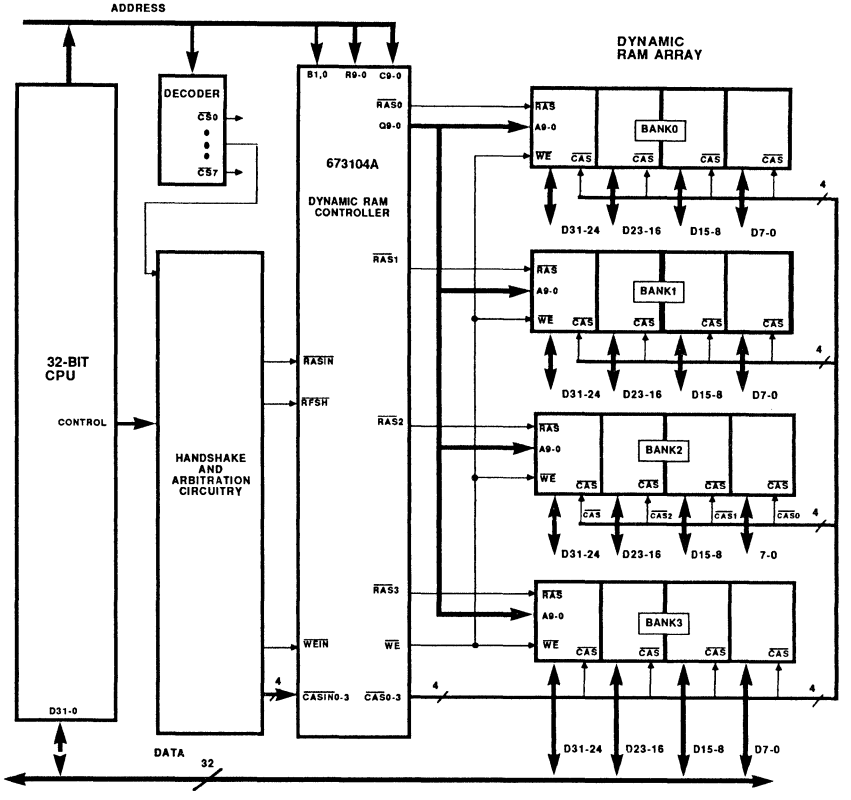


14060-013A

APPLICATIONS

Microprocessor Interface

The 673104A Dynamic RAM Controller provides the address and control signals required to access and refresh dynamic RAMs. When interfaced to a 32-bit microprocessor, some external logic is required to generate a refresh clock as well as to perform access/refresh arbitration and interface handshake functions. For some microprocessors external logic is required also to decode the address and signal to arrive at four data strobes. A hidden refresh (refresh which is transparent to the system) scheme may be implemented in the interface circuitry taking advantage of "free" system time to refresh to the memory, and falling back to "forced" refresh when hidden refresh cannot be performed.



14060-014A

Figure 11. A CPU Interfaced to the 673104A Driving 16 M Bytes of Dynamic Memory

Determining System Performance (Auto-Access)

When determining system performance the dynamic RAM parameters must be considered as well as the controller's propagation delays. For both read and write cycles the access time for the dynamic RAM is t_{RAC} ($\overline{RAS}$ access time) from $\overline{RAS}$ going LOW or t_{CAC} ($\overline{CAS}$ access time) from $\overline{CAS}$ going LOW. Since the $\overline{RAS}$ and $\overline{CAS}$ coming out of the controller are initiated by the $\overline{RASIN}$, the controller-memory performance is measured from the $\overline{RASIN}$ HIGH-to-LOW transition (see Figure 12).

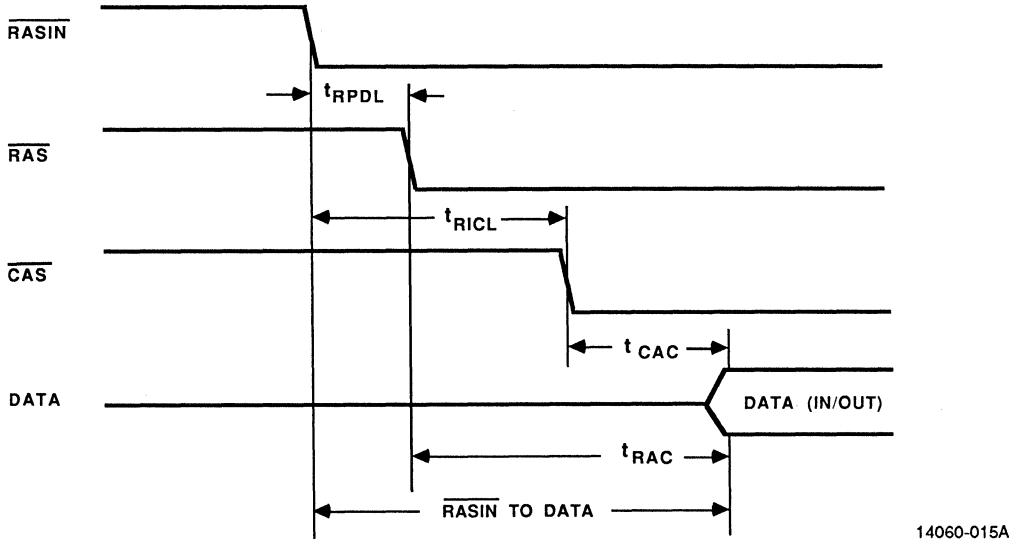


Figure 12. Access Time from $\overline{RASIN}$

The time from $\overline{RASIN}$ to data is calculated to be the longer of:

$t_{RICL} + t_{CAC}$ ($\overline{RASIN}$ to $\overline{CAS}$ + $\overline{CAS}$ to data)
$t_{RPDL} + t_{RAC}$ ($\overline{RASIN}$ to $\overline{RAS}$ + $\overline{RAS}$ to data)

Table 2 illustrates the access times from $\overline{RASIN}$ achieved for various dynamic RAM speeds.

Table 2. Access Times from $\overline{RASIN}$ for Various Memory Speeds

Controller/Memory	Parameter				
	t_{RAC}	t_{RPDL}	t_{CAC}	t_{RICL}	Access Time from $\overline{RASIN}$
673104A/HM256-12	120	23	60	83	143
673104A/HM256-15	150	23	75	83	173
673104A/MB8265A-10	100	23	50	83	133
673104A/MB8265A-12	120	23	60	83	143
673104A/IMS2620-10	100	23	60	83	143
673104A/IMS2620-12	120	23	70	83	153

673104A Parameters

t_{RICL} – $\overline{RASIN}$ LOW to $\overline{CAS}$ LOW delay

t_{RPDL} – $\overline{RASIN}$ LOW to $\overline{RAS}$ LOW delay

DRAM Parameters

t_{RAC} – Access time from $\overline{RAS}$ LOW

t_{CAC} – Access time from $\overline{CAS}$ LOW

Using the Externally Controlled Access

In the Externally Controlled Access mode $\overline{RASIN}$ controls the selected $\overline{RAS}_n$ output, $\overline{CASIN}_{0-3}$ control $\overline{CAS}_{0-3}$ outputs respectively and R/$\overline{C}$ controls the address multiplexer. The system designer may create, using the $\overline{RASIN}$, $\overline{CASIN}$ and R/$\overline{C}$ inputs, the required control signal sequence for the specific system being designed. Special dynamic RAM access modes such as Nibble mode and Page mode access cycles may be performed simply by toggling the appropriate control inputs. Special skew timing specifications have been specified to allow tighter timing control as outlined in the following examples. Both following examples relate to the scheme depicted in Figure 13.

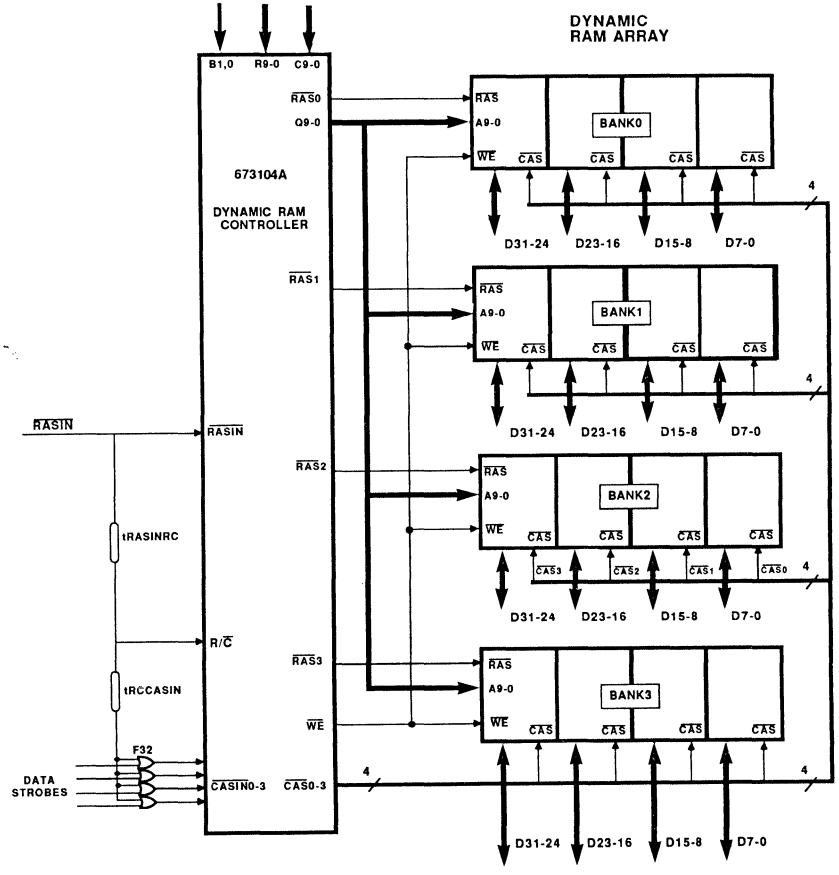


Figure 13. The 673104A in the Externally-Controlled Access Mode

Externally Controlled Access (ECA) (Cont'd.)

Example 1: Computing $\overline{RASIN}$ to R/$\overline{C}$ Delay

The delay between $\overline{RASIN}$ going LOW to R/$\overline{C}$ going LOW ($t_{RASINRC}$) which is required in order to satisfy the dynamic RAMs' row address hold time (t_{RAH}) is computed as follows:

$t_{RASINRC} = t_{RAH}$ (min) + t_{d7}

Where:

t_{RAH} (min) – Row address hold time (dynamic RAM parameter)

t_{d7} (max) = t_{RPDL}-t_{RHA}

t_{RPDL} – $\overline{RASIN}$ to $\overline{RAS}$ LOW delay

t_{RHA} – Row address held valid from R/$\overline{C}$ LOW

Example 2: Computing R/$\overline{C}$ to $\overline{CASIN}$ Delay

The delay between R/$\overline{C}$ going LOW and $\overline{CASIN}$ going LOW ($t_{RCCASIN}$) which is required in order to satisfy the dynamic RAMs' column address setup (t_{ASC}) is computed as follows:

$t_{RCCASIN} = t_{ASC}$ (min) + t_{d8} + t_{PD}F32(max)

Where:

t_{ASC} (min) – Column address setup (dynamic RAM parameter)

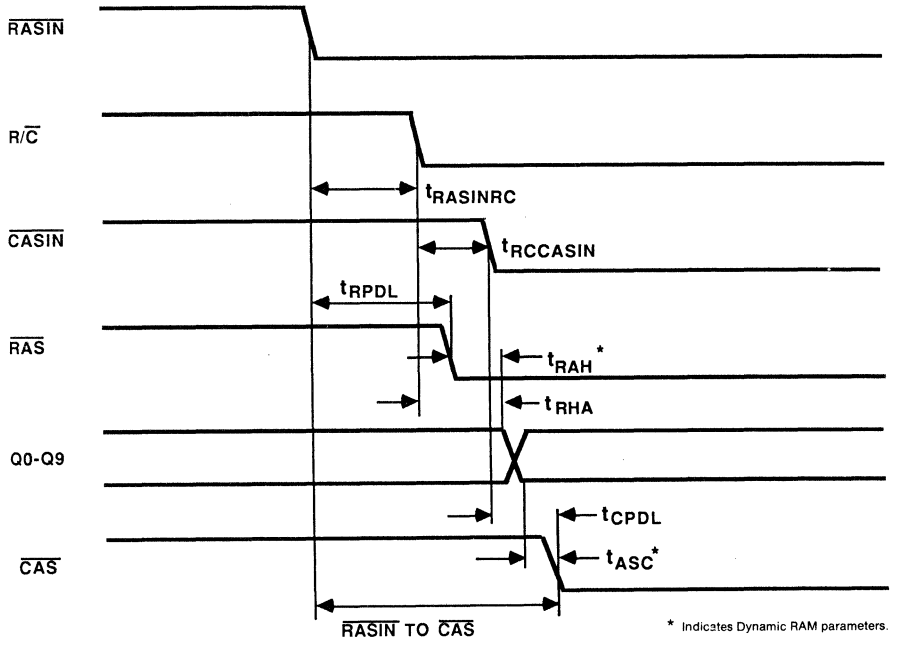
t_{d8} (max) = t_{RCC}-t_{CPDL}

t_{RCC} – R/$\overline{C}$ low to column address valid

t_{CPDL} – $\overline{CASIN}$ to $\overline{CAS}$ LOW delay

t_{PD}F32 (max) – Propagation delay of the OR gate used to validate $\overline{CASIN}$

Better system performance may be achieved using the t_{d7}, t_{d8} switching parameters to calculate $t_{RASINRC}$ and $t_{RCCASIN}$ than when using the t_{RPDL}, t_{RCDL}, t_{RCC} and t_{RHA} parameters (see Externally Controlled Access switching parameters).



* Indicates Dynamic RAM parameters.

14060-017A

Figure 14. Externally Controlled Access Timing

Final

SN74S409-2/DP8409A-2
SN74S409/DP8409A
256K Dynamic RAM Controller/Driver

Advanced Micro Devices

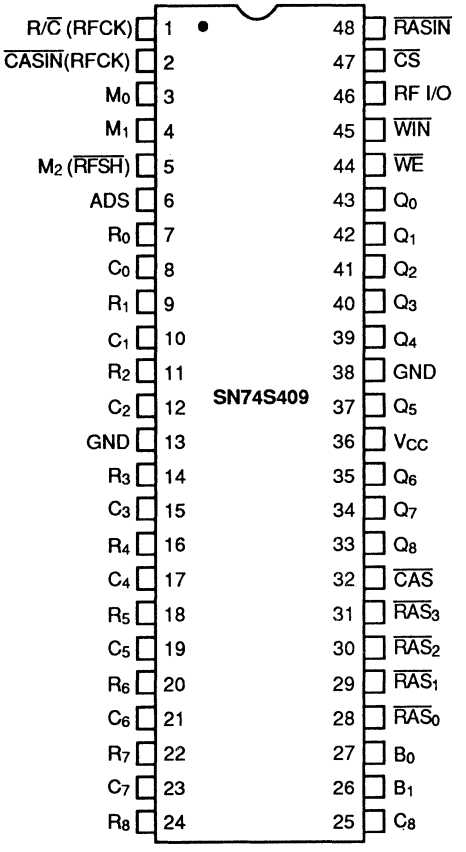
DISTINCTIVE CHARACTERISTICS

- **All DRAM drive functions on one chip have on-chip high-capacitance load drivers (specified up to 88 DRAMs)**
- **Drives directly all 16K, 64K and 256K DRAMs; capable of addressing up to 1M words**
- **Propagation delays of 25 nsec typical at 500 pF load**
- **Supports READ, WRITE and READ-MODIFY-WRITE cycles**
- **Eight modes of operation support externally-controlled and automatic access and refresh, as well as special memory initialization access**
- **On-chip 9-bit refresh counter with selectable End-of-Count (127, 255 or 511)**
- **Direct replacement for National DP8409, DP8409A**

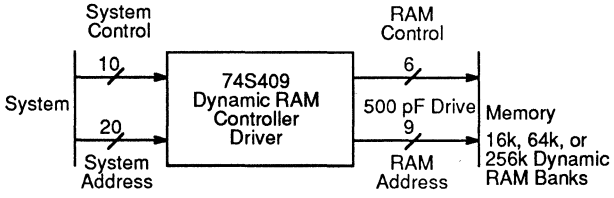
OPERATING MODES

0	Externally-controlled fresh
1	Auto refresh — forced
2	Automatic burst refresh
3a	All-$\overline{RAS}$ auto write
3b	Externally-controlled All-$\overline{RAS}$ write
4	Externally-controlled access
5	Auto access, slow t_{RAH}, hidden refresh
6	Auto access, fast t_{RAH}
7	Set end of count

PIN CONFIGURATION



13099-002A



Interface Between System and DRAM Banks

13099-001A

Publication # 13099 Rev. A Amendment /0
Issue Date: January 1990

BLOCK DIAGRAM

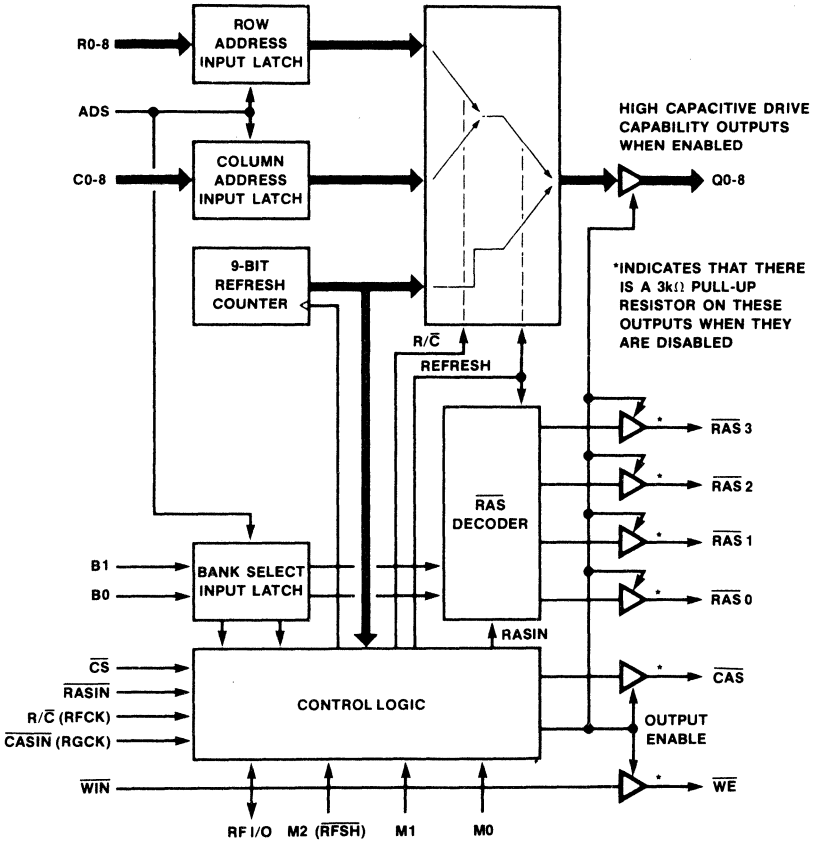


Figure 1. 74S409 Functional Block Diagram

GENERAL DESCRIPTION

The 74S409 is a Multi-Mode Dynamic RAM Controller/Driver capable of directly driving up to 88 DRAMs. 20 address lines to the 74S409 allow it to address up to 1M words and it can drive 16K, 64K and 256K DRAMs. Since the 74S409 is a one-chip solution (including capacitive-load drivers), it minimizes propagation delay skews, and saves board space.

The 74S409's 8 operating modes offer externally-controlled or on-chip automatic access and refresh. An on-chip refresh counter makes refreshing (either externally or automatically controlled) less complicated; and automatic memory initialization is both simple and fast.

The 74S409 is a 48-pin DRAM Controller/Driver with 9 multiplexed address outputs and 6 control signals. It consists of two 9-bit address latches, a 9-bit refresh counter, and control logic. The 74S409 timing parameters are specified when driving the typical load capitance of 88 DRAMs, including trace capacitance.

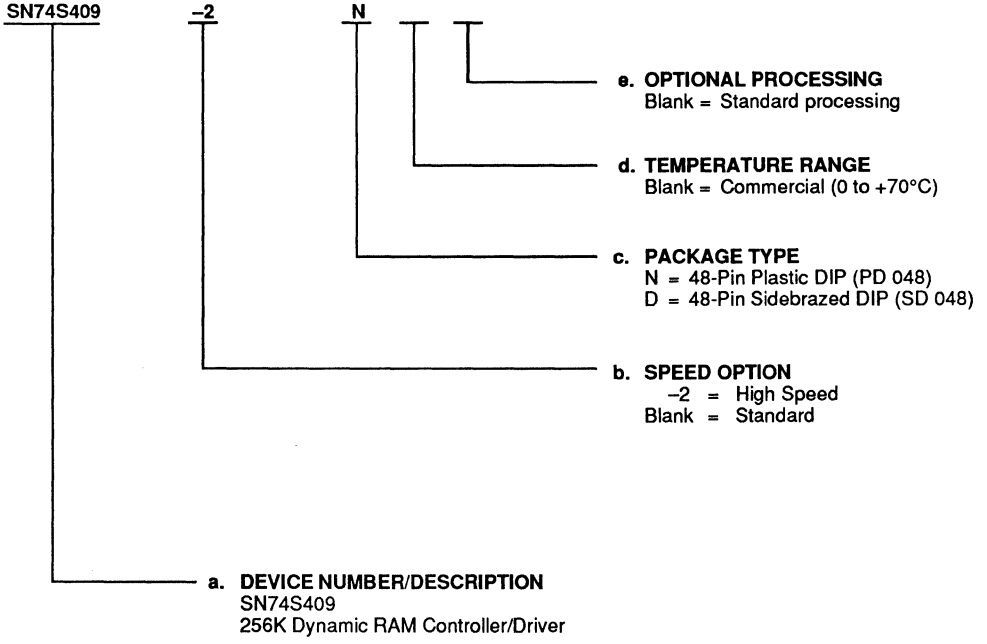
The 74S409 can drive up to 4 banks of DRAMs, with each bank comprised of 16Ks, 64Ks or 256Ks. Control signal outputs $\overline{CAS}$ and $\overline{WE}$ are provided with the same driving capability. Each $\overline{RAS}$ output drives one bank of DRAMs so that the four $\overline{RAS}$ outputs are used to select the banks, while $\overline{CAS}$, $\overline{WE}$ and the multiplexed addresses can be connected to all the banks of DRAMs. This leaves the non-selected banks in the standby mode (less than one tenth of the operating power) with the respective data outputs in three-state. Only the bank with its associated $\overline{RAS}$ low will be written to or read from, except in mode 3 where all $\overline{RAS}$ signals go low to allow fast memory initialization.

ORDERING INFORMATION

Standard Products

AMD standard products are available in several packages and operating ranges. The order number (Valid Combination) is formed by a combination of:

a. Device Number
b. Speed Option (if applicable)
c. Package Type
d. Temperature Range
e. Optional Processing



Valid Combinations	
SN74S409	N, D
SN74S409-2	

Valid Combinations

Valid Combinations list configurations planned to be supported in volume for this device. Consult the local AMD sales office to confirm availability of specific valid combinations, to check on newly released combinations, and to obtain additional data on AMD's standard military grade products.

Pin Description

V_{CC} GND, GND—V_{CC} = 5V ± 5%. The three supply pins have been assigned to the center of the package to reduce voltage drops, both DC and AC. There are also two ground pins to reduce the low level noise. The second ground pin is located two pins from V_{CC}, so that decoupling capacitors can be inserted directly next to these pins. It is important to adequately decouple this device, due to the high switching currents that will occur when all 9 address bits change in the same direction simultaneously. A recommended solution is a 1-μF multilayer ceramic capacitor in parallel with a low-voltage tantalum capacitor, both connected close to pins 36 and 38 to reduce lead inductance.

BANK SELECT (STROBED BY ADS)		ENABLED $\overline{RAS}_n$
B1	B0	
0	0	$\overline{RAS}_0$
0	1	$\overline{RAS}_1$
1	0	$\overline{RAS}_2$
1	1	$\overline{RAS}_3$

Table 1. Memory Bank Decode

R0-R8: Row Address Inputs.

C0-C8: Column Address Inputs.

B0, B1: Bank Select Inputs—Strobed by ADS. Decoded to enable one of the $\overline{RAS}$ outputs when $\overline{RASIN}$ goes low, in modes 4-6. In mode 7 B0, B1 are used to define End-of-Count (see table 3), and select mode 3a or 3b.

Q0-Q8: Multiplexed Address Outputs—Selected from the Row Address Input Latch, the Column Address Input Latch, or the Refresh Counter.

$\overline{RASIN}$: Row Address Strobe Input—Enables selected $\overline{RAS}_n$ output when M2 ($\overline{RFSH}$) is high (modes 4-6), and all $\overline{RAS}_n$ outputs in modes 0 and 3. $\overline{RASIN}$ input is disabled in modes 1 and 2.

R/$\overline{C}$ (RFCK)—In Auto-Refresh Mode this pin is the external Refresh Clock Input: one refresh cycle has to be performed each clock period. In all other modes it is Row/$\overline{Column}$ Select Input, selecting either the row or column address input latch onto the output bus.

$\overline{CASIN}$ (RGCK)—In modes 1, 2 and 3a, this pin is the $\overline{RAS}$ Generator Clock input. In all other modes it is $\overline{CASIN}$ (Column Address Strobe Input), which inhibits $\overline{CAS}$ output when high in Modes 3b and 4. In Mode 6 it can be used to prolong $\overline{CAS}$ output.

ADS: Address (Latch) Strobe Input—Strobes Input Row Address, Column Address, and Bank Select Inputs into respective latches when high; latches on High-to-Low transition.

$\overline{CS}$: Chip Select Input—three-state's the Address Outputs and puts the control signal into a high-impedance logic "1" state when high (unless refreshing in one of the Refresh Modes). Enables all outputs when low.

M0, M1, M2 ($\overline{RFSH}$): Mode Control Inputs—These 3 control pins determine the 8 major modes of operation of the 74S409 as depicted in Table 2.

RF I/O $\overline{RFRQ}$ — This I/O pin functions as a Reset Counter Input when set low from an external open-collector gate, or as a flag output. The flag goes active-low in Modes 0, 2 and 3a when the End-of-Count output is at 127, 255, or 511 (see Table 3). In Auto-Refresh Mode (mode 5) it is the $\overline{Refresh}$ $\overline{Request}$ ($\overline{RFRQ}$) output.

$\overline{WIN}$: Write Enable Input.

$\overline{WE}$: Write Enable Output—Buffered output from $\overline{WIN}$.

$\overline{CAS}$: Column Address Strobe Output—In Modes 3a, 5, and 6, $\overline{CAS}$ transitions low following valid column address. In Modes 3b and 4, it goes low after R/$\overline{C}$ goes low, or follows $\overline{CASIN}$ going low if R/$\overline{C}$ is already low. $\overline{CAS}$ is high during refresh.

$\overline{RAS}$ 0-3: Row Address Strobe Outputs—When M2($\overline{RFSH}$) is high (modes 4-6), the selected row address strobe output (decoded from signals B0, B1) follows the $\overline{RASIN}$ input. When M2 ($\overline{RFSH}$) is low (modes 0-3) all $\overline{RAS}_n$ outputs go low together following $\overline{RASIN}$ going low in modes 0 and 3 and automatically in modes 1 and 2.

Input Addressing

The address block consists of a row-address latch, a column-address latch, and a resettable refresh counter.

The address latches are fall-through when ADS is high and latch when ADS goes low. If the address bus contains valid address until after the valid address time, ADS can be permanently high. Otherwise ADS must go low while the address is still valid.

In normal memory-access operation, $\overline{RASIN}$ and R/$\overline{C}$ are initially high. When the address inputs are enabled into the address latches (modes 3-6) the row addresses appear on the Q outputs. The Address Strobe also inputs the bank-select address, (B0 and B1). If $\overline{CS}$ is low, all outputs are enabled. When $\overline{CS}$ goes high, the address outputs go three-state and the control outputs first go high through a low impedance, and then are held by an on-chip high impedance. This allows output paralleling with other 74S409s for multi-addressing. All outputs go active about 50ns after the chip is selected again. If $\overline{CS}$ is high, and a refresh cycle begins, all the outputs become active until the end of the refresh cycle.

Drive Capability

The 74S409 has timing parameters that are specified with up to 600pF loads for $\overline{CAS}$ and $\overline{WE}$, 500pF loads for Q_0-Q_8, and 150pF loads for $\overline{RAS}_n$ outputs. In a typical memory system this is equivalent to about 88 5V-only DRAMs, with trace lengths kept to a minimum. Therefore, the chip can drive four banks each of 16 or 22 bits, or two banks of 32 or 39 bits, or one bank of 64 or 72 bits.

Less loading will slightly reduce the timing parameters, and more loading will increase the timing parameters, according to the graph of Figure 14. The AC performance parameters are specified with the typical load capacitance of 88 DRAMs. This graph can be used to extrapolate the variations expected with other loading.

74S409 Driving Any 16K, 64K or 256K DRAMs

The 74S409 can drive any 16K, 64K, or 256K DRAMs. The on-chip 9-bit counter with selectable End-of-Count can support refresh of 128, 256 and 512 rows, while the 9 address and 4 $\overline{RAS}_n$ outputs can address 4 banks of 16K, 64K or 256K DRAMs.

Read, Write, and Read-Modify-Write Cycles

The output signal, $\overline{WE}$, determines what type of memory access cycle the memory will perform. If $\overline{WE}$ is kept high while $\overline{CAS}$ goes low, a read cycle occurs. If $\overline{WE}$ goes low before $\overline{CAS}$ goes low, a write cycle occurs and data at DI (DRAM input data) is written into the DRAM as $\overline{CAS}$ goes low. If $\overline{WE}$ goes low later than t_{CWD} after $\overline{CAS}$ goes low, first a read occurs and DO (DRAM output data) becomes valid; then data DI is written into the same address in the DRAM when $\overline{WE}$ goes low. In this read-modify-write case, DI and DO cannot be linked together. The type of cycle is therefore controlled by $\overline{WE}$, which follows $\overline{WIN}$.

Power-Up Initialize

When V_{CC} is first applied to the 74S409, an internal pulse clears the refresh counter, the internal control flip-flops, and sets the End-of-Count of the refresh counter to 127 (which may be changed via Mode 7). As V_{CC} increases to about 2.3 volts, it holds the output control signals at a level of one Schottky diode-drop below V_{CC}, and the output address to three-state. As V_{CC} increases above 2.3 volts, control of these outputs is granted to the system.

74S409 Functional Modes Description

The 74S409 operates in 8 different functional modes selected by signals M_0,M_1,M_2. Mode 3 splits further to modes 3a and 3b determined by signals B_0,B_1 in mode 7.

Mode 0 and mode 1 are generally used as Refresh modes for mode 4 and mode 5 respectively, and therefore will be described as mode-pairs 0,4 and 1,5.

Mode 6 is a fast access made for very fast DRAMs and mode 7 is used only to determine choice of mode 3a or 3b and for setting End-of-Count for the refresh modes.

MODE	($\overline{RFSH}$) M2	M1	M0	MODE OF OPERATION	CONDITIONS
0	0	0	0	Externally-controlled refresh	RF I/O = $\overline{EOC}$
1	0	0	1	Auto refresh—forced	RF I/O = Refresh request ($\overline{RFRQ}$)
2	0	1	0	Automatic burst refresh	RF I/O = $\overline{EOC}$
3a*	0	1	1	All-$\overline{RAS}$ auto write	RF I/O = $\overline{EOC}$; all $\overline{RAS}$ active
3b*	0	1	1	Externally-controlled All-$\overline{RAS}$ write	All-$\overline{RAS}$ active
4	1	0	0	Externally-controlled access	Active $\overline{RAS}$ defined by Table 2
5	1	0	1	Auto access, slow t_{RAH}, hidden refresh	Active $\overline{RAS}$ defined by Table 2
6	1	1	0	Auto access, fast t_{RAH}	Active $\overline{RAS}$ defined by Table 2
7	1	1	1	Set end of count; determines mode 3a or 3b	See Table 3 for Mode 7

*Mode 3a is selected by setting B_0,B_1 to 01, 00, or 10 in mode 7.
*Mode 3b is selected by setting B_1,B_0 to 11 in mode 7.

Table 2. 74S409 Mode Select Options

Mode 0 — Externally-Controlled Refresh
Mode 4 — Externally-Controlled Access

Modes 0 and 4 facilitate external control of all timing parameters associated with the DRAMs. These modes are independent modes of operation though generally used together in the same application as shown in Figure 2.

Mode 0—Externally-Controlled Refresh

In this mode the input address latches are disabled from the address outputs and the refresh counter is enabled. All $\overline{RAS}$ outputs go low following $\overline{RASIN}$ and refresh the enabled row in all four banks. $\overline{CASIN}$ and R/$\overline{C}$ inputs are not used and $\overline{CAS}$ is inhibited. The refresh counter increments when either $\overline{RASIN}$ or M2 ($\overline{RFSH}$) switch high while the other is still low.

RF I/O goes low when the count equals End-of-Count (as set in mode 7), and $\overline{RASIN}$ is low. The 9-bit counter will always roll-over to zero at 512, regardless of End-of-Count. However, the counter can be reset at any time by driving RF I/O low through an external open-collector.

During refresh, $\overline{RASIN}$ and M2 ($\overline{RFSH}$) can transition low simultaneously because the refresh counter becomes valid on the output bus t_{RFLCT} after $\overline{RFSH}$ goes low, which is a shorter time than t_{RFPDL}. This means the counter address is valid on the Q outputs before $\overline{RAS}$ occurs on all $\overline{RAS}$ outputs, strobing the counter address into that row of all the DRAMs (see Figure 2.). To perform externally-controlled burst refresh, $\overline{RFSH}$ initially can again have the same edge as $\overline{RASIN}$, but then maintains a low state, since $\overline{RASIN}$ going low-to-high increments the counter (performing the burst refresh).

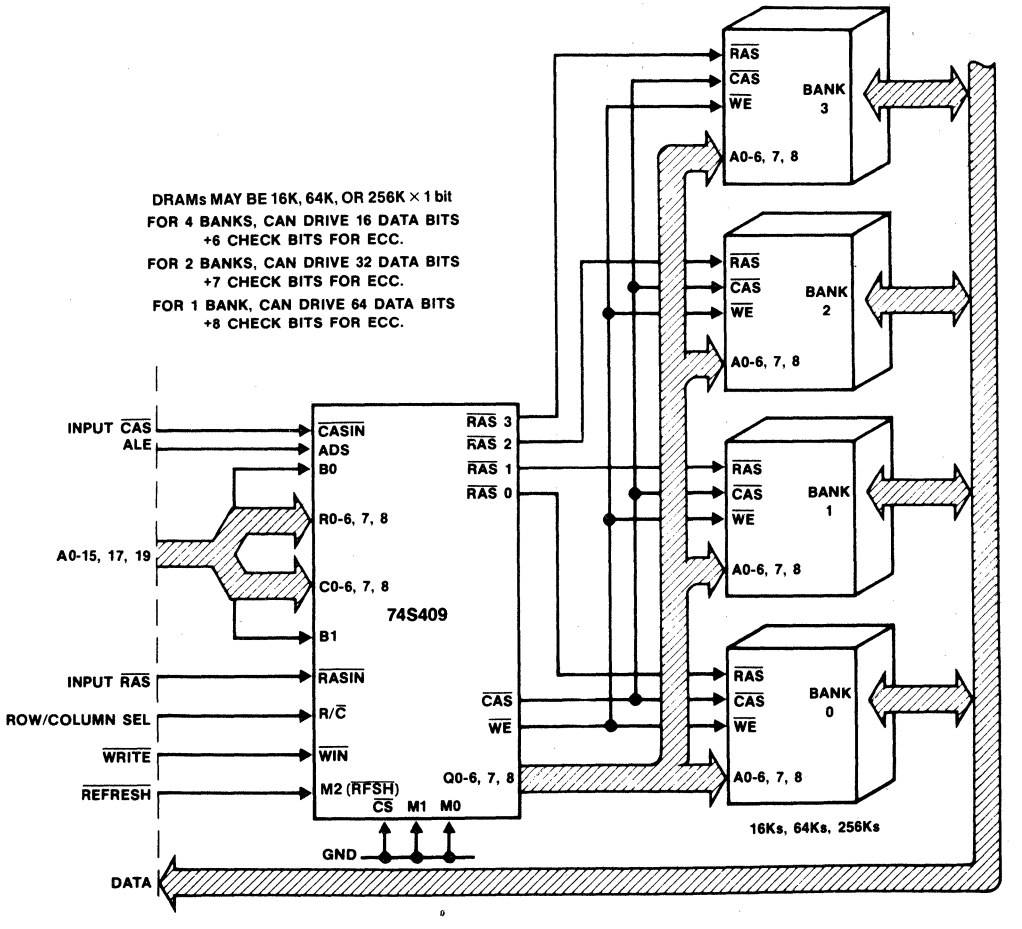


Figure 2. Typical Application of 74S409 Using Externally-Controlled Access and Refresh in Modes 0 and 4

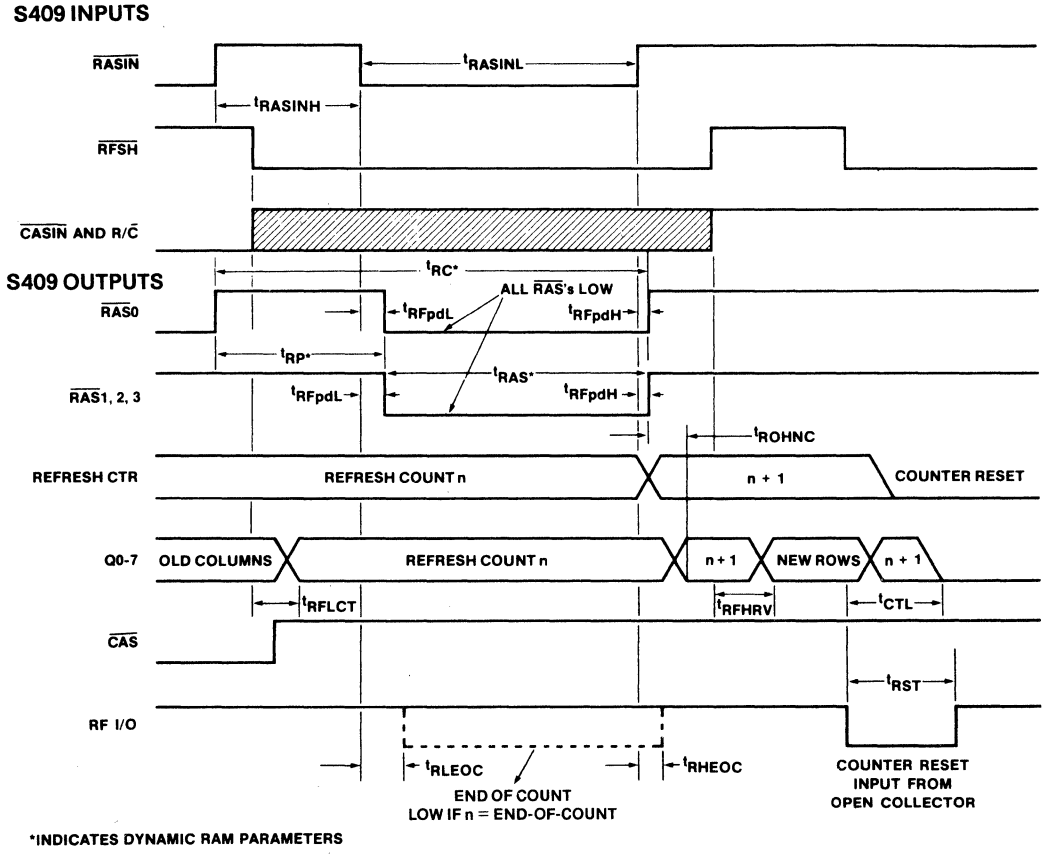


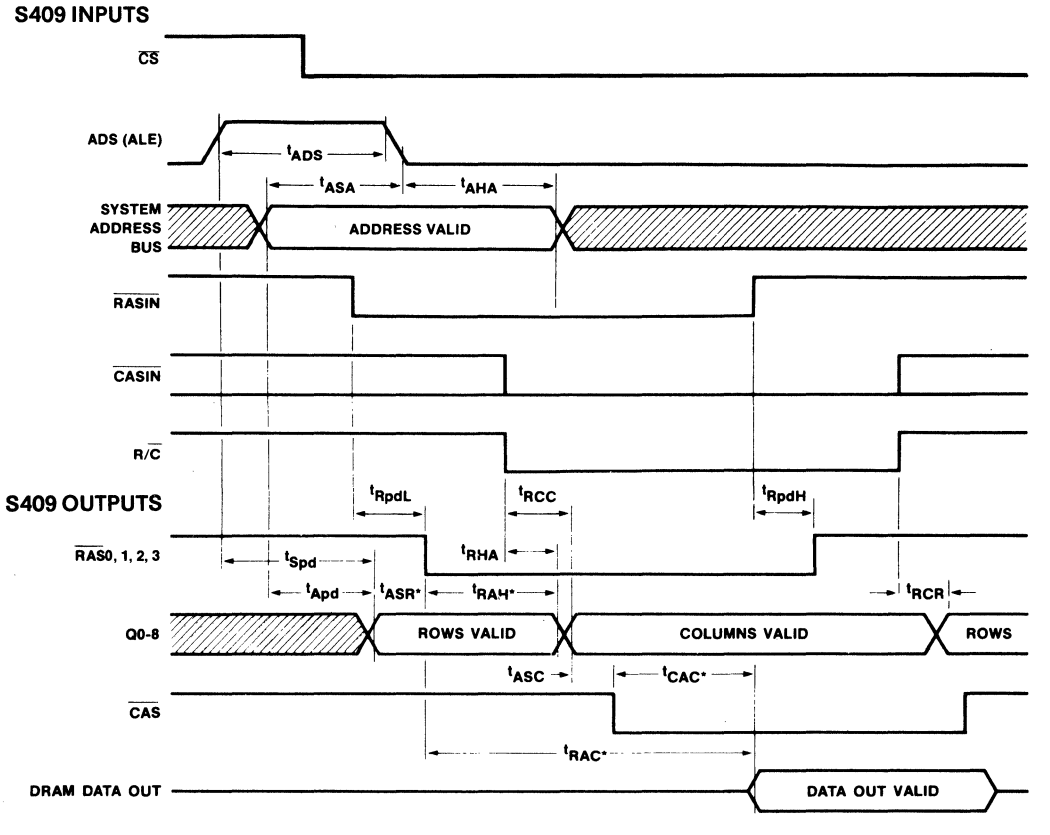
Figure 3. External Control Refresh Cycle (Mode 0)

Mode 4 — Externally-Controlled Access

This mode facilitates externally controlling all access-timing parameters associated with the DRAMs. Figures 4 and 5 show the timing for read and write cycles.

Output Address Selection

In this mode $\overline{CS}$ has to be low at least 50 nsec before the outputs will be valid. With R/$\overline{C}$ high, the row address latch contents are transfered to the multiplexed address bus output Q0-Q8. $\overline{RASIN}$ can go low after the row addresses have been set up on Q0-Q8, and enables one $\overline{RAS}$ output selected by signals B0, B1 to strobe the Q outputs into the desired bank of memory. After the row-address hold-time of the DRAMs, R/$\overline{C}$ can go low so that about 40 nsec later, the column address appears on the Q output.



*INDICATES DYNAMIC RAM PARAMETERS

Figure 4. Read Cycle Timing (Mode 4)

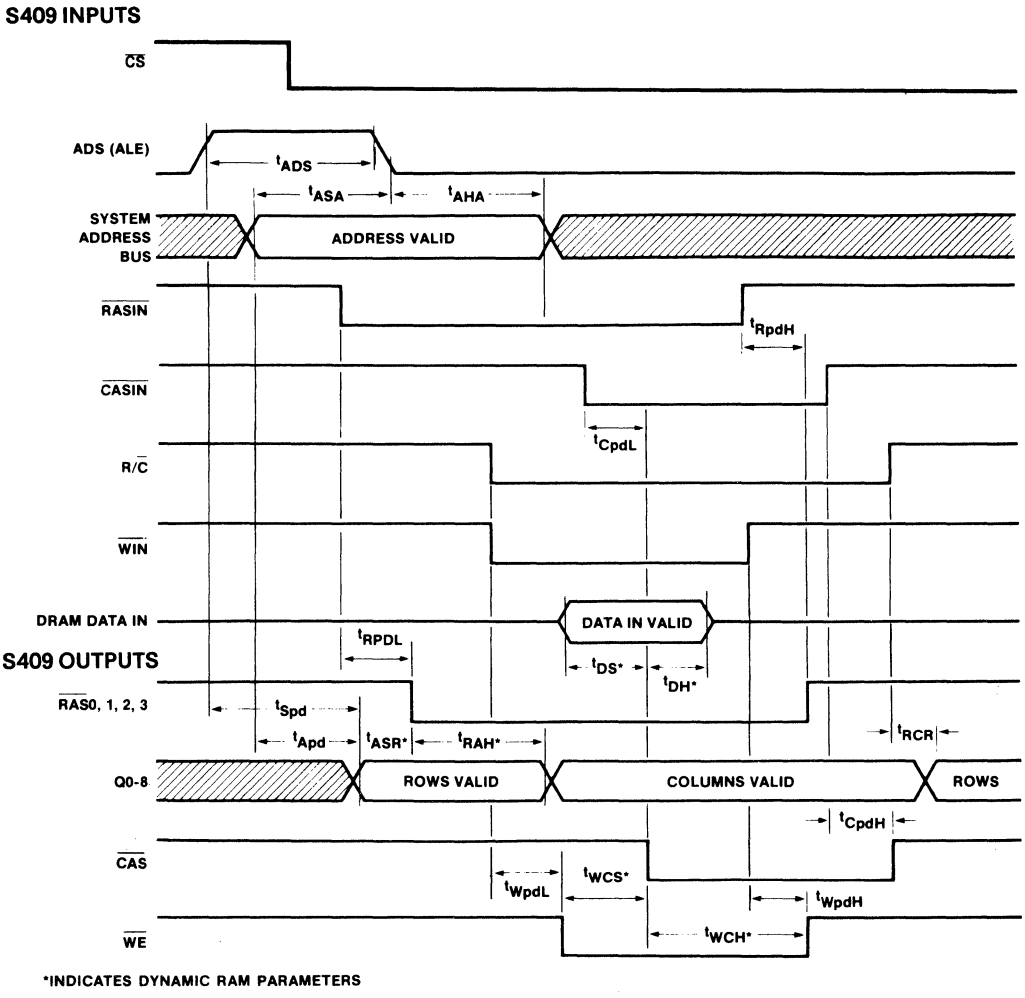


Figure 5. Write Cycle Timing (Mode 4)

Automatic $\overline{CAS}$ Generation

In a normal memory access cycle $\overline{CAS}$ can be derived from inputs $\overline{CASIN}$ or R/$\overline{C}$. If $\overline{CASIN}$ is high, then R/$\overline{C}$ going low switches the address output drivers from rows to columns. $\overline{CASIN}$ then going low causes $\overline{CAS}$ to go low approximately 40 ns later, allowing $\overline{CAS}$ to occur at a predictable time (see Figure 5). For maximum system speed, $\overline{CASIN}$ can be kept low, since $\overline{CAS}$ will automatically occur approximately 60 ns after R/$\overline{C}$ goes low (see Figure 4). Most DRAMs have a column address set-up time before $\overline{CAS}$ (t_{ASC}) of 0 ns or −10 ns. In other words, a t_{ASC} greater than 0 ns is safe. This feature reduces timing-skew problems, thereby improving access time of the system.

Fast Memory Access

For faster access time, R/$\overline{C}$ can go low a time delay (t_{RPDL} + t_{RAH} − t_{RHA}) after $\overline{RASIN}$ goes low, where t_{RAH} is the Row-Address hold-time of the DRAM, and $\overline{CASIN}$ can go low t_{RCC} − t_{CPOL} + t_{ASC} (min.) after R/$\overline{C}$ goes low (see t_{DiF1}, t_{DiF2} switching characteristics).

Mode 1 – Automatic Forced Refresh
Mode 5 – Automatic Access with Hidden Refresh

Mode 1 and Mode 5 are generally used together incorporating the advantages of the "hidden refresh" performed in mode 5 with the possibiity to force a refresh by changing to mode 1. An advantage of the Automatic Access over the Externally-Controlled Access is the reduced memory access time, due to the fact that the output control signals are derived internally from one input signal ($\overline{RASIN}$).

Hidden and Forced Refresh

Hidden Refresh is a term describing memory refresh performed when the system does not access the portion of memory controlled by the 74S409 ($\overline{CS}$ = 1). A hidden refresh will occur once per Refresh Clock (RFCK) cycle provided $\overline{CS}$ went high and $\overline{RASIN}$ went low. If no hidden refresh occurred while RFCK was high, the RF I/O ($\overline{RFRQ}$) goes low immediately after RFCK goes low, indicating to the system when a forced refresh is required. The system must allow a forced refresh to take place while RFCK is low by driving M2 ($\overline{RFSH}$) low, thereby changing mode of operation to Mode 1.

The Refresh Request on RF I/O ($\overline{RFRQ}$) is terminated as soon as $\overline{RAS}$ goes low, indicating to the system that the foced refresh has been done. The system should then drive M2 ($\overline{RFSH}$) high, changing the mode of operation back to Mode 5 (see Figure 6).

Mode 1 – Automatic Forced Refresh

In Mode 1, the R/$\overline{C}$ (RFCK) pin functions as RFCK (refresh cycle clock) instead of R/$\overline{C}$, and $\overline{CAS}$ remains high. If RFCK is kept permanently high then whenever M2 ($\overline{RFSH}$) goes low, an externally-controlled refresh will occur and all RAS outputs will follow $\overline{RASIN}$, strobing the refresh counter contents to the DRAMs. The RF I/O pin will always output high, but can be set low externally through an open-collector driver, to reset the refresh counter.

If RFCK is an input clock, one and only one refresh cycle must take place every RFCK cycle. If a hidden refresh does not occur while RFCK is high, in Mode 5, then RF I/O (Refresh Request) goes low immediately after RFCK goes low, indicating to the system that a forced refresh is required. The system must allow a forced refresh to take place while RFCK is low The Refresh Request signal on RF I/O may be connected to a Hold or Bus Request input to the system. The system acknowledges the Hold or Bus Request when ready, and outputs Hold Acknowledge or Bus Request Acknowledge. If this is connected to the M2 ($\overline{RFSH}$) pin, a forced-refresh cycle will be initiated by the S409, and $\overline{RAS}$ will be internally generated on all four $\overline{RAS}$ outputs, strobing the refresh counter contents on the address ouputs into all the DRAMs. An external $\overline{RAS}$ Generator Clock (RGCK) is requred for this function. It is fed to the $\overline{CASIN}$ (RGCK) pin, and may be up to 10 MHz. Whenever M2 goes low (inducing a forced refresh), $\overline{RAS}$ remains high for one to two periods of RGCK, depending on when M2 goes low relative to the high-to-low triggering edge of RGCK; $\overline{RAS}$ then goes low for two periods, performing a refresh on all banks. In order to obtain the minimum delay from M2 going low to $\overline{RAS}$ going low, M2 should go low $t_{\overline{RFSRG}}$ before the next falling edge of RGCK. The Refresh Request on RF I/O is terminated as $\overline{RAS}$ begins, so that by the time the system has acknowledged the removal of the request and disabled its Acknowledge, (i.e., M2 goes high), Refresh $\overline{RAS}$ will have ended, and normal operations can begin again in the Automatic Access mode (Mode 5). If it is desired that Refresh $\overline{RAS}$ end in less than 2 periods of RGCK from the time $\overline{RAS}$ went low, then M2 may go high earlier than t_{FRQH} after RF I/O goes high and $\overline{RAS}$ will go high t_{RFRH} after M2.

Mode 5 – Automatic Access with Hidden Refresh

In this mode all address outputs, $\overline{RAS}$ and $\overline{CAS}$ are initiated from $\overline{RASIN}$ making the DRAM access appear similar to static RAM access. The hidden refresh feature enables DRAM refresh accomplished with no time-loss to the system.

Provided the input address is valid as ADS goes low, $\overline{RASIN}$ can go low any time after ADS. This is because the selected $\overline{RAS}$ occurs typically 27 ns later, by which time the row address is already valid on the address output of the 74S409. The Address Set-Up time (t_{ASR}), is 0 ns on most DRAMs. The 74S409 in this mode (with ADS and $\overline{RASIN}$ edges simultaneously applied) produces a minimum t_{ASR} of 0 ns. This is true provided the input address was valid t_{ASR} before ADS went low (see Figure 7).

Next, the row address is disabled t_{RAH} after $\overline{RAS}$ goes low (30 ns minimum); in most DRAMs, t_{RAH} minimum is less than 30 ns. The column address is then set up and (t_{ASC} later,) $\overline{CAS}$ occurs. The only other control input required is $\overline{WIN}$. When a write cycle is required, $\overline{WIN}$ must go low at least 30 ns before $\overline{CAS}$ is output low.

This gives a total typical delay from: input address valid to $\overline{RASIN}$ (15 ns); to $\overline{RAS}$ (27 ns); to rows held (50 ns); to columns valid (25 ns); to $\overline{CAS}$ (23 ns) = 140 ns (that is, 125 ns from $\overline{RASIN}$). All of these typical figures are for heavy capacitive loading, of approximately 88 DRAMs.

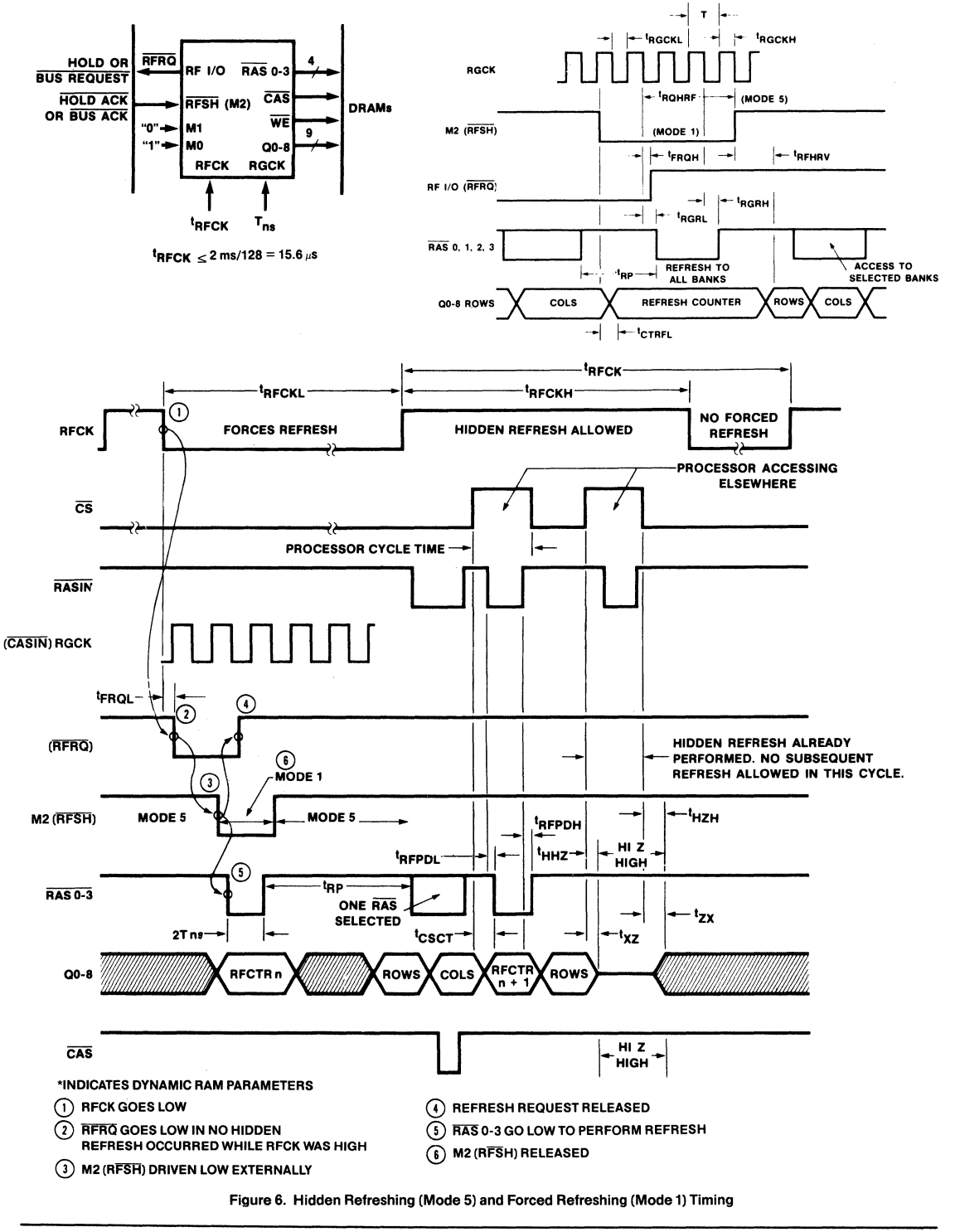
Refreshing

In this mode R/$\overline{C}$ (RFCK) functions as Refresh Clock and $\overline{CASIN}$ (RGCK) functions as $\overline{RAS}$ Generator Clock.

One refresh cycle must occur during each refresh clock period, and then the refresh address must be incremented before the next refresh cycle. As long as 128 rows are refreshed every 2 ms (one row every 16 μs), all 16K and 64K DRAMs will be correctly refreshed. The cycle time of RFCK must, therefore, be less than 16 μs. RFCK going high sets an internal refresh-request flipflop. First the 74S409 will attempt to perform a hidden refresh so that the system thruput will not be affected. If, during the time RFCK is high, $\overline{CS}$ on the 74S409 goes high and $\overline{RASIN}$ occurs, a hidden refresh will occur. In this case, $\overline{RASIN}$ should be considered a common read/write strobe. In other words, if the processor is accessing elsewhere (other than the DRAMs) while RFCK is high, the 74S409 will perform a refresh. The refresh counter is enabled to the address outputs whenever $\overline{CS}$ goes high with RFCK high, and all $\overline{RAS}$ outputs follow $\overline{RASIN}$. If a hidden refresh is taking place as RFCK goes low, the refresh continues. At the start of the hidden refresh, the refresh-request flipflop is reset so on further refresh can occur until the next RFCK period starts with the positive-going edge of RFCK (see Figure 6). $\overline{RASIN}$ should go low at least 20 ns before RFCK goes low, to ensure occurrence of the hidden refresh.

To determine the probability of a hidden refresh occurring, goes low, (and the internal-request flipflop has not been for 8μs, then the system has 20 chances to not select the 74S409. If during this time a hidden refresh did not occur, then the 74S409 forces a refresh while RFCK is low, but the system chooses when the refresh takes place. After RFCK goes low, (and the internal-request flip-flop has not been reset), RF I/O goes low indicating that a refresh is requested to the system. Only when the system acknowledges this request by setting M2 ($\overline{RFSH}$) low does the 74S409 initiate a forced refresh (which is performed automatically). Refer to Mode 1, and Figure 6. The internal refresh request flipflop is then reset.

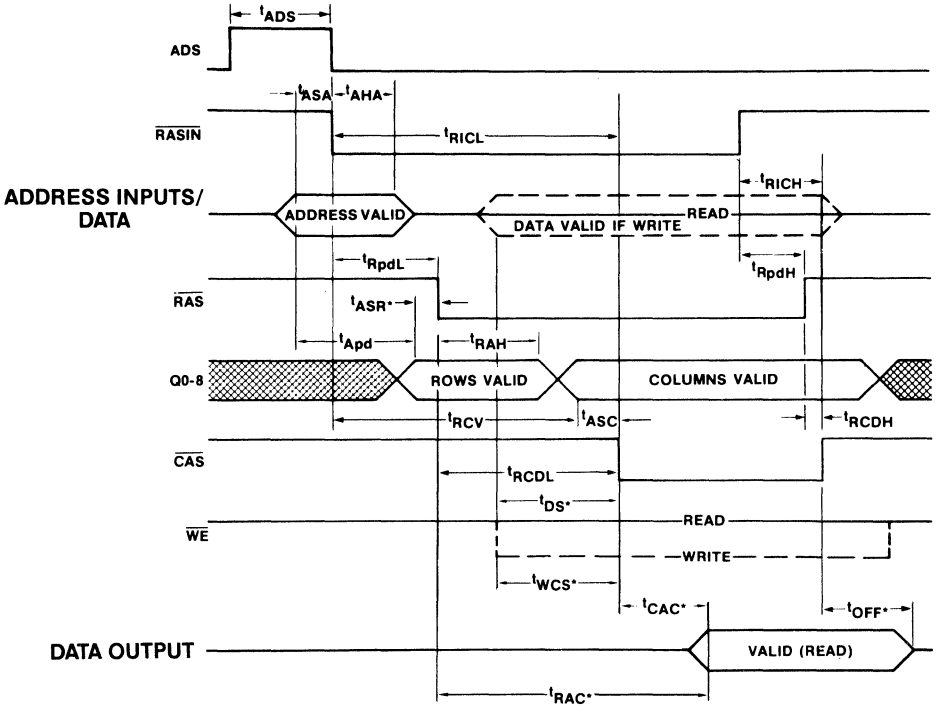
Figure 6 illustrates the refresh alternatives in Mode 5. If a hidden refresh has occurred and $\overline{CS}$ again goes high before RFCK goes low, the chip is deselected. All the control signals go high-impedance high (logic "1") and the address outputs go three-state until $\overline{CS}$ again goes low. This mode (combined with Mode 1) allows very fast access, and automatic refreshing (possibly not even slowing down the system), with no extra ICs. Careful system design can, and should, provide a higher probability of hidden refresh occurring. The duty cycle of RFCK need not be 50 percent; in fact, the low-time should be designed to be a minimum. This is determined by the worst-case time (required by the system) to respond to the 74S409's forced-refresh request.



*INDICATES DYNAMIC RAM PARAMETERS

1. RFCK GOES LOW
2. RFRQ GOES LOW IN NO HIDDEN REFRESH OCCURRED WHILE RFCK WAS HIGH
3. M2 (RFSH) DRIVEN LOW EXTERNALLY
4. REFRESH REQUEST RELEASED
5. RAS 0-3 GO LOW TO PERFORM REFRESH
6. M2 (RFSH) RELEASED

Figure 6. Hidden Refreshing (Mode 5) and Forced Refreshing (Mode 1) Timing



*INDICATES DYNAMIC RAM PARAMETERS

Figure 7. Mode 5 Timing

Mode 2 – Automatic Burst Refresh

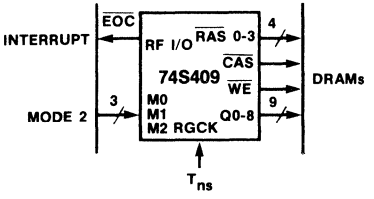
This mode is normally used before and/or after a DMA operation to ensure that all rows remain refreshed, provided the DMA transfer takes less than 2 ms (see Figure 8). When the 74S409 enters this mode, $\overline{CASIN}$ (RGCK) becomes the $\overline{RAS}$ Generator Clock (RGCK), and $\overline{RASIN}$ is disabled. $\overline{CAS}$ remains high, and RF I/O goes low when the refresh counter has reached the selected End-of-Count and the last $\overline{RAS}$ has ended. RF I/O then remains low until the Auto-Burst Refresh mode is terminated. RF I/O can therefore be used as an interrupt to indicate the End-of-Burst condition.

The signal on all four $\overline{RAS}$ outputs is just a divide-by-four of RGCK; in other words, if RGCK has a 100 ns period, $\overline{RAS}$ is high and low for 200 ns each cycle. The refresh counter increments at the end of each $\overline{RAS}$, starting from the count it contained when the mode was entered. If this was zero then for a RGCK with a 100 ns period with End-of Count set to 127, RF I/O will go low after 128 x 0.4μs, or 51.2μs. During this time, the system may be performing operations that do not involve DRAM. If all rows need to be burst refreshed, the refresh counter may be cleared by setting RF I/O low externally before the burst begins.

Burst-mode refreshing is also useful when powering down systems for long periods of time, but with data retention still required while the DRAMs are in standby. To maintain valid refreshing, power can be applied to the 74S409 (set to Mode 2), causing it to perform a complete burst refresh. When end-of-bust occurs (after 26 μs), power can then be removed from the 74S409 for 2 ms, consuming an average power of 1.3% of normal operating power. No control signal glitches occur when switching power to the 74S409.

Mode 3a – All-$\overline{RAS}$ Automatic Write

Mode 3a is useful at system initialization, when the memory is being cleared (i.e., with all-zeroes in the data field and the corresponding check bits for error detection and correction). This requires writing the same data to each location of memory (every row of each column of each bank). All $\overline{RAS}$ outputs are activated, as in refresh, and so are $\overline{CAS}$ and $\overline{WE}$. To write to all four banks simultaneously, every row is strobed in each column, in sequence, until data has been written to all locations. The refresh counter is used to address the rows, and $\overline{RAS}$ is low for two RGCK cycles and high for two cycles.



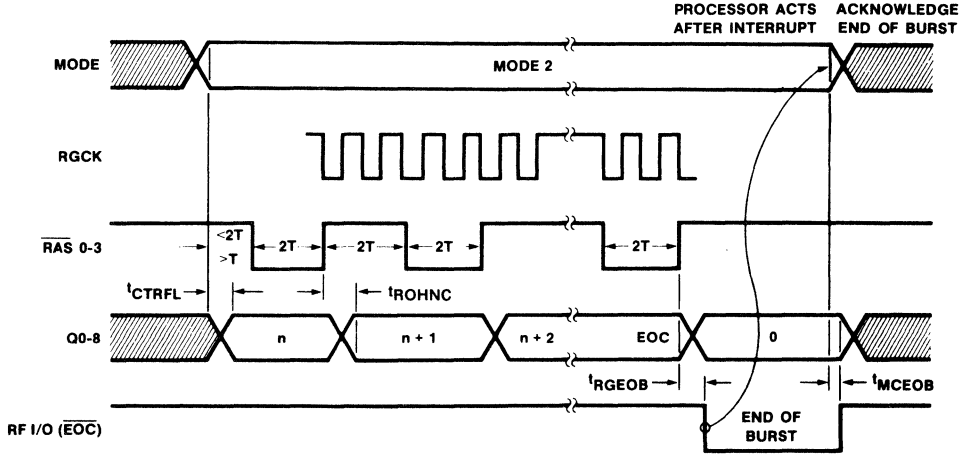
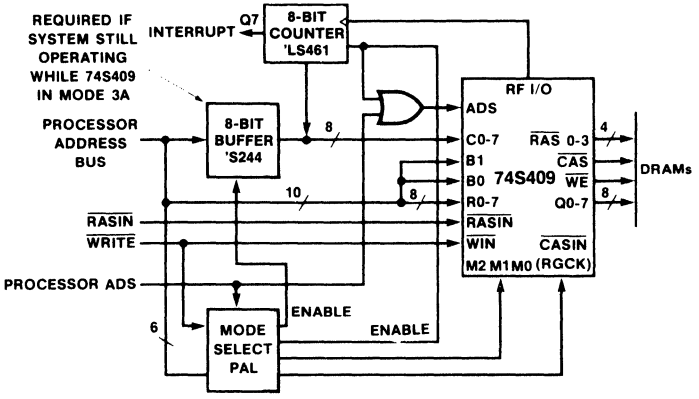


Figure 8. Auto-Burst Mode, Mode 2

To select this mode, B1 and B0 must have previously been set to 00, 01, or 10 in Mode 7, depending on the DRAM size. For example, for 16K DRAMs, B1 and B0 are 00. For 64K DRAMs, B1 and B0 are 01.

In this mode, R/$\overline{C}$ is disabled, $\overline{WE}$ is permanently enabled low, and $\overline{CASIN}$ (RGCK) becomes RGCK. RF I/O goes low whenever the refresh counter is 127, 255, or 511 (as set by End-of-Count in Mode 7), and the $\overline{RAS}$ outputs are active.



74S409 Extra Circuitry Required for All-$\overline{RAS}$ Auto Write Mode, Mode 3a

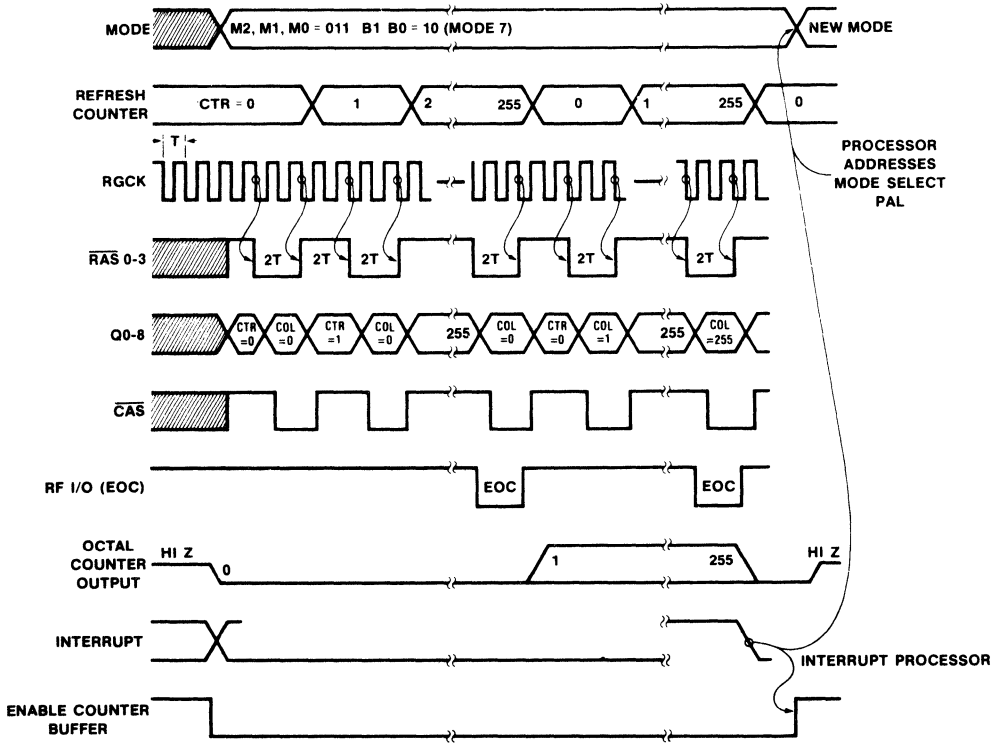


Figure 9. 74S409 All-$\overline{RAS}$ Auto Write Mode, Mode 3a, Timing Waveform

Mode 3b — Externally-Controlled All-$\overline{RAS}$ Write

To select this mode, B1 and B0 must first have been set to 11 in Mode 7. This mode is useful at system initialization, but under processor control. The memory address is provided by the processor, which also performs the incrementing. All four $\overline{RAS}$ outputs follow $\overline{RASIN}$ (supplied by the processor), strobing the row address into the DRAMs. R/$\overline{C}$ can now go low, while $\overline{CASIN}$ may be used to control $\overline{CAS}$ (as in the Externally-Controlled Access mode), so that $\overline{CAS}$ strobes the column address contents into the DRAMs. At this time $\overline{WE}$ should be low, causing the data to be written into all four banks of DRAMs. At the end of the write cycle, the input address is incremented and latched by the 74S409 for the next write cycle. This method is slower than Mode 3a, since the processor must perform the incrementing and accessing. Thus the processor is occupied during RAM initialization, and is not free for other initialization operations. However, initialization sequence timing is under system control, which may provide some system advantage.

Mode 4 — Externally-Controlled Access

Mode 4 is described in with mode 0 in section "Mode 0 and Mode 4."

Mode 5 — Automatic Access with Hidden Refresh

See description of mode 0 and mode 5.

Mode 6 — Fast Automatic Access

The Fast Automatic Access mode can only be used with fast DRAMs which have t_{RAH} of 10 nsec-15nsec. The typical $\overline{RASIN}$ to $\overline{CAS}$ delay is 105nsec. In this mode $\overline{CAS}$ can be extended after $\overline{RAS}$ goes high to extend the data output valid time. This feature is useful in applications with short cycles where $\overline{RAS}$ has to be terminated as soon as possible to meet the precharge (t_{RP}) requirements of the DRAM.

Mode 6 timing is illustrated in Figures 10 and 11. Provided that the input address is valid as ADS goes low, $\overline{RASIN}$ can go low any time after ADS. This is because the selected $\overline{RAS}$ occurs typically 27 ns later, by which time the row address is already valid on the address output of the 74S409. The Address Set-Up time (t_{ASR}), is 0 ns on most DRAMs. The 74S409 in this mode (with ADS and $\overline{RASIN}$ edges simultaneously applied) produces a minimum t_{ASR} of 0 ns. This is true provided the input address was valid t_{ASA} before ADS went low (see Figure 10).

Next, the row address is disabled t_{RAH} after $\overline{RAS}$ goes low (20 ns minimum); the column address is then set up and t_{ASC} later, $\overline{CAS}$ occurs. The only other control input required is $\overline{WIN}$. When a write cycle is required, $\overline{WIN}$ must go low at least 30 ns before $\overline{CAS}$ is output low.

This gives a total typical delay from: input address valid to $\overline{RASIN}$ (15 ns); to $\overline{RAS}$ (27 ns): to rows valid (50 ns); to columns valid (25 ns); to $\overline{CAS}$ (23 ns) = 140 ns (that is, 125 ns from $\overline{RASIN}$). All of these typical figures are for heavy capacitive loading, of approximately 88 DRAMs.

This mode is therefore extremely fast. The external timing is greatly simplified for the memory system designer: the only system signal required is $\overline{RASIN}$.

In this mode, the R/$\overline{C}$ (RFCK) pin is not used, but $\overline{CASIN}$ (RGCK) is used as $\overline{CASIN}$ to allow an extended $\overline{CAS}$ after $\overline{RAS}$ has already terminated. Refer to Figure 11.

Mode 7 — Set End-of-Count (3a, 3b select)

The End-of-Count can be externally selected in Mode 7, using ADS to strobe in the respective value of B1 and B0 (see Table 3). With B1 and B0 the same $\overline{EOC}$ is 127; with B1 = 0 and B0 = 1, $\overline{EOC}$ is 255; and with B1 = 1 and B0 = 0, $\overline{EOC}$ is 511. This selected value of $\overline{EOC}$ will be used until the next Mode 7 selection. At power-up the $\overline{EOC}$ is automatically set to 127 (B1 and B0 set to 11).

When B_1,B_2 are set to 11 in mode 7, mode 3b will be selected if mode 3 is selected (M_2, M_1, M_0 = 0, 1, 1). If B_1,B_2 is set to 00, 01 or 10 then mode 3a will be selected.

BANK SELECT (STROBED BY ADS)		END OF COUNT SELECTED
B1	B0	
0	0	127
0	1	255
1	0	511
1	1	127

Table 3. Mode 7

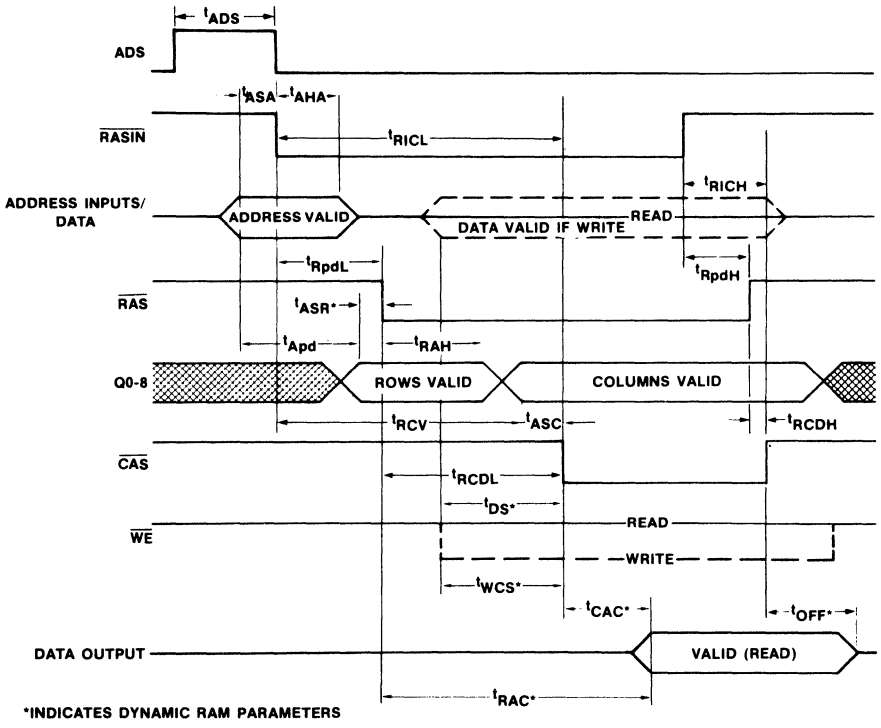


Figure 10. Mode 6 Timing ($\overline{\text{CASIN}}$ High)

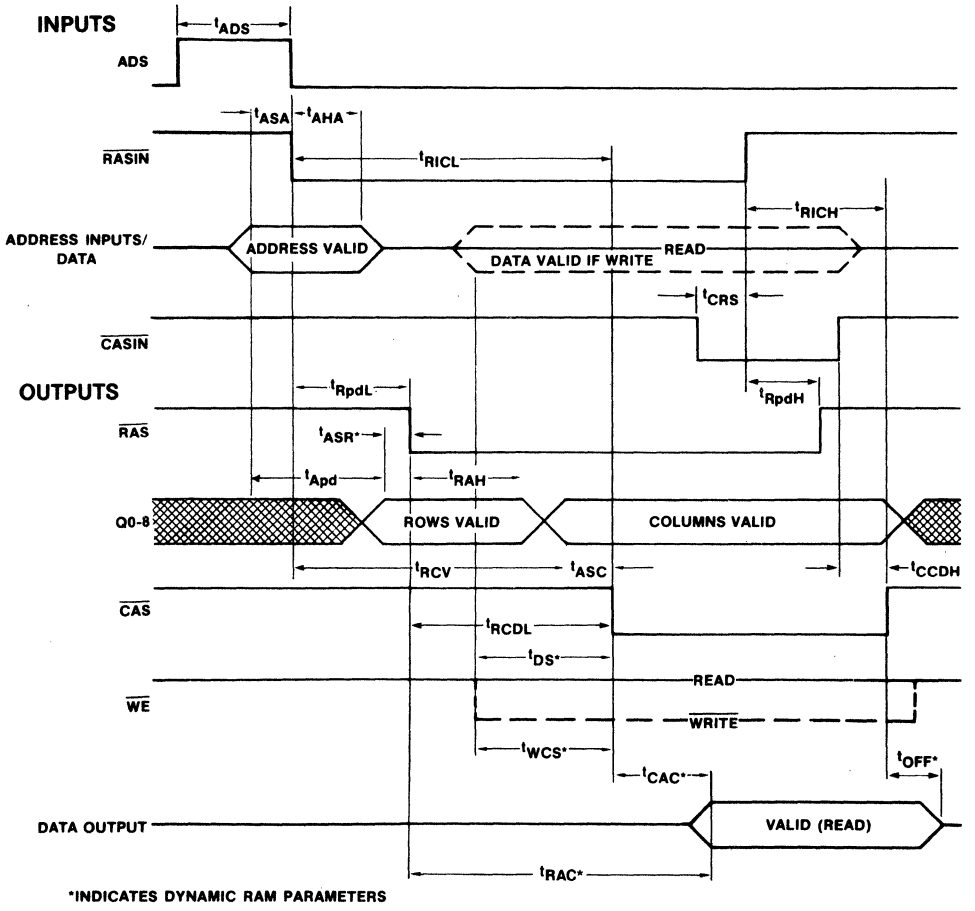


Figure 11. Mode 6 Timing, Extended $\overline{CAS}$

Switching Characteristics: (Cont'd)

SYMBOL	ACCESS PARAMETER	TEST CONDITIONS	'S409 MIN	'S409 TYP	'S409 MAX	'S409-2 MIN	'S409-2 TYP	'S409-2 MAX	UNIT
	THREE-STATE PARAMETER								
t_{ZH}	$\overline{CS}$ low to address output high from Hi	Figures 6, 12 R1 = 3.5k, R2 = 1.5k		35	60		35	60	ns
t_{HZ}	$\overline{CS}$ high to address output Hi-Z from high	C_L = 15pF, Figures 6,12 R2 = 1k, S1 Open		20	40		20	40	ns
t_{ZL}	$\overline{CS}$ low to address output low from Hi-Z	Figures 6, 12 R1 = 3.5k, R2 = 1.5k		35	60		35	60	ns
t_{LZ}	$\overline{CS}$ high to address output Hi-Z from low	C_L = 15pF, Figures 6,13 R1 = 1k, S2 Open		25	50		25	50	ns
t_{HZH}	$\overline{CS}$ low to control output ($\overline{WE}$, $\overline{CAS}$, ($\overline{RASO-3}$) high from Hi-Z high	Figures 6,12 R2 = 750Ω, S1 open		50	80		50	80	ns
t_{HHZ}	$\overline{CS}$ high to control output ($\overline{WE}$, $\overline{CAS}$, ($\overline{RASO-3}$) Hi-Z high from high	C_L = 15pF R2 = 750Ω, S1 open		40	75		40	75	ns
t_{HZL}	$\overline{CS}$ low to control output ($\overline{WE}$, $\overline{CAS}$, ($\overline{RASO-3}$) low from Hi-Z high	Figure 12 S1, S2 Open		45	75		45	75	ns
t_{LHZ}	$\overline{CS}$ high to control output ($\overline{WE}$, $\overline{CAS}$, ($\overline{RASO-3}$) Hi-Z high from low	C_L = 15pF, Figure 12 R2 = 750Ω, S1 open		50	80		50	80	ns

*Internally the device contains a 3K resistor in series with a Schottky Diode to V_{CC}.

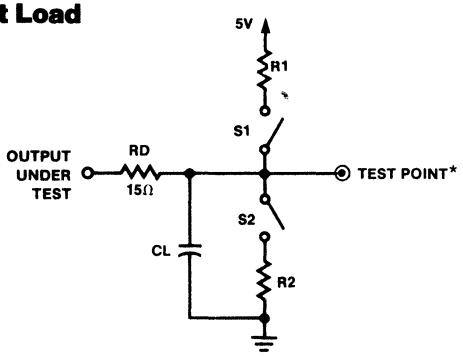
Note 1: Output load capacitance is typical for 4 banks of 22 DRAMs or 88 DRAMs including trace capacitance. These values are: Q0-Q8. C_L = 500pF; RAS0-RAS3, C_L = 150pF; CAS C_L = 600pF unless otherwise noted.

Note 2: All typical values are for T_A = 25°C and V_{CC} = 5.0V.

Note 3: This test is provided as a monitor of Driver output source and sink current capability. Caution should be exercised in testing this parameter. In testing these parameters, a 15Ω resistor should be placed in series with each output under test. One output should be tested at a time and test time should not exceed 1 second.

Note 4: Input pulse 0V to 3.0V, $t_R = t_F$ = 2.5 ns, f = 2.5 MHz. t_{PW} = 200 ns. Input reference point on AC measurements is 1.5V. Output reference points are 2.7V for High and 0.8V for Low.

Test Load



R1, R2 = 4.7K EXCEPT AS SPECIFIED.

* The "TEST POINT" is driven by the output under test, and observed by instrumentation.

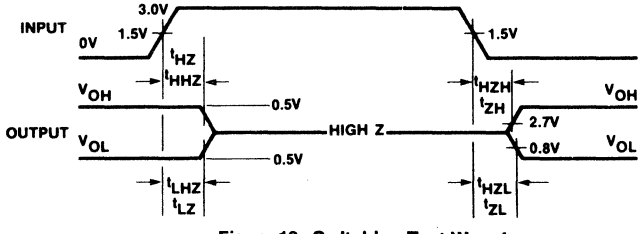


Figure 12. Switching Test Waveforms

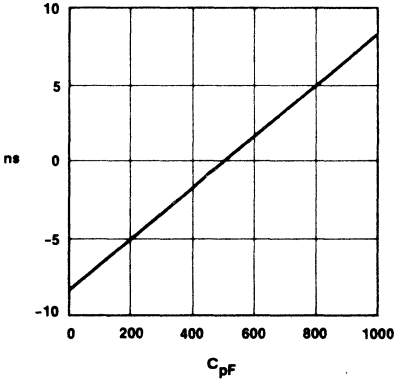


Figure 13. Change in Propagation Delay vs Loading Capacitance Relative to a 500 pF Load

Applications

The 74S409 Dynamic RAM Controller provides all the address and control signals necessary to access and refresh dynamic RAMs. Since the 74S409 is not compatible with a specific bus or microprocessor, an interface is often necessary between the 74S409 and the system. A general application using PAL to implement the interface and two additional chips to provide refresh clock and chip select is shown in Figure 14.

The 74S409 operating modes may vary from application to application. For efficient refresh it is recommended to use mode 1 and mode 5 to take advantage of the hidden (transparent) refresh with forced refresh backup.

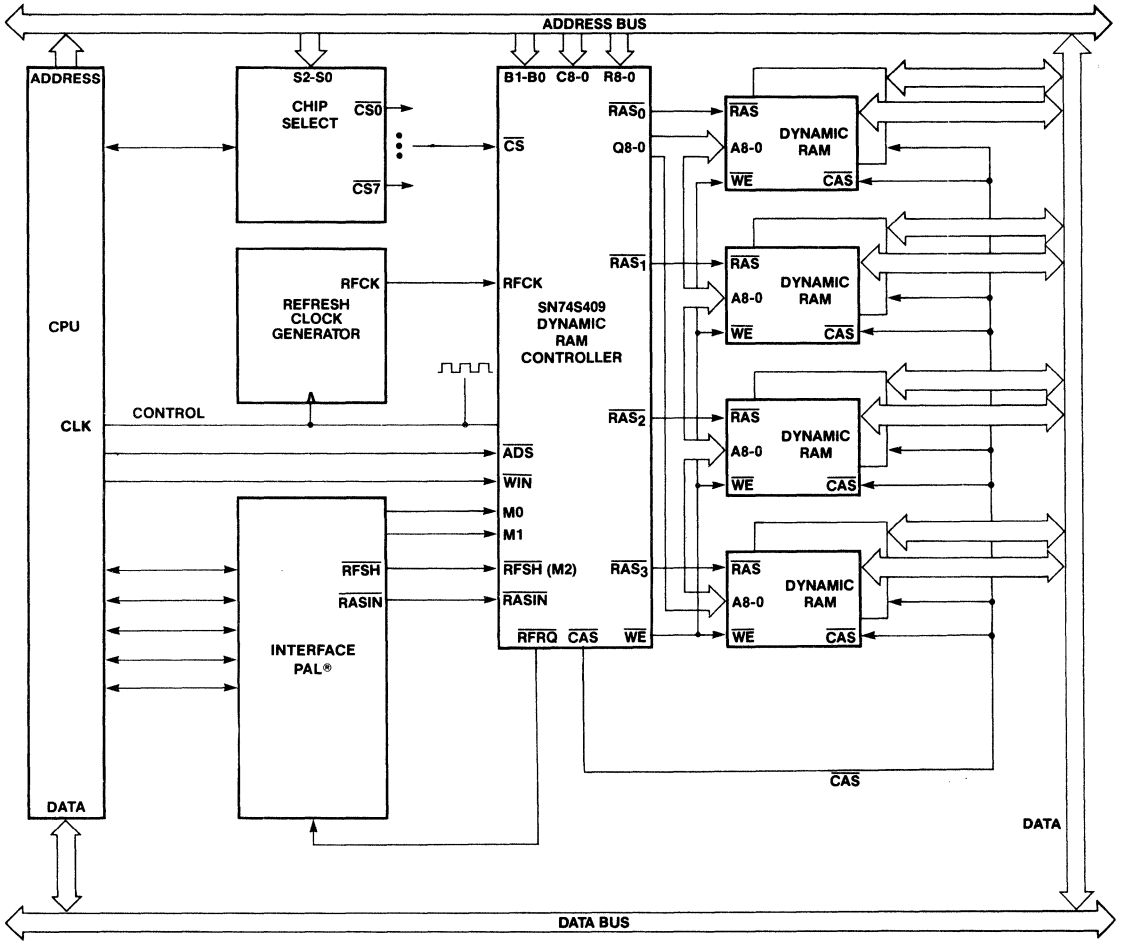


Figure 14. 74S409 in General Application

ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V_{CC})	−0.5 to +7.0 V
Storage Temperature Range	−65°C to +150°C
Input Voltage	−1.5 V to +5.5 V
Output Current	150 mA
Lead Temperature (soldering, 10 seconds)	300°C

Stresses above those listed under ABSOLUTE MAXIMUM RATINGS *may cause permanent device failure. Functionality at or above these limits is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability. Absolute maximum ratings are for system design reference; parameters given are not tested.*

OPERATING RANGES

Commercial (C) Devices

Supply Voltage (V_{CC})	+4.75 to +5.25 V
Operating Free Air Temperature (T_A)	0 to +75°C

Operating ranges define those limits between which the functionality of the device is guaranteed.

DC CHARACTERISTICS (V_{CC} = 5.0 V ± 5.0%, 0°C ≤ T_A ≤ 75°C, Typicals are for V_{CC} = 5 V, T_A = 25°C)

Parameter Symbol	Parameter Description	Test Conditions	Min.	Typ.	Max.	Unit
V_C	Input clamp voltage	V_{CC} = Min., I_C = −12 mA		−0.8	−1.2	V
I_{IH1}	Input high current for ADS, R/C only	V_{IN} = 2.5 V		2.0	100	µA
I_{IH2}	Input high current for other inputs, except RF I/O	V_{IN} = 2.5 V		1.0	50	µA
I_IRSI	Output load current for RF I/O	V_{IN} = 0.5 V, output high		−1.5	−2.5	mAV
I_ICTL	Output load current for $\overline{RAS}$, $\overline{CAS}$, $\overline{WE}$	V_{IN} = 0.5 V, chip deselect		−1.5	−2.5	mA
I_{IL1}	Input low current for ADS, R/$\overline{C}$ only	V_{IN} = 0.5 V		−0.1	−1.0	mA
I_{IL2}	Input low current for other inputs, except RF I/O	V_{IN} = 0.5 V		−0.05	−0.5	mA
V_{IL}**	Input low threshold				0.8	V
V_{IH}**	Input high threshold		2.0			V
V_{OL1}	Output low voltage, except RF I/O	I_{OL} = 20 mA		0.3	0.5	V
V_{OL2}	Output low voltage for RF I/O	I_{OL} = 10 mA		0.3	0.5	V
V_{OH1}	Output high voltage, except RF I/O	I_{OH} = −1 mA	2.4	3.5		V
V_{OH2}	Output high voltage for RF I/O	I_{OL} = −100 µA	2.4	3.5		V
I_{1D}	Output high drive current, except RF I/O	V_{OUT} = 0.8 V (Note 3)		−200		mA
I_{0D}	Output low drive current, except RF I/O	V_{OUT} = 2.7 V (Note 3)		200		mA
I_{OZ}	Three-state output current (address outputs)	0.4 V ≤ V_{OUT} ≤ 2.7 V, CS = 2.0 V, Mode 4	−50	1.0	50	µA
I_{CC}	Supply current	V_{CC} = Max		250	325	mA
C_{IN}	Input capacitance ADS, R/$\overline{C}$	T_A = 25°C		8		pF
C_{IN}	Input capacitance all other inputs	T_A = 25°C		5		pF

**These are absolute voltages with respect to pins 13 or 38 on the device and include all overshoots due to system or tester noise. Do not attempt to test these values without suitable equipment.

SWITCHING CHARACTERISTICS (V_{CC} = 5.0 V ± 5.0%, 0°C ≤ T_A ≤ 75°C) See Figure 12 for test load (switches S1 and S2 are closed unless otherwise specified) typicals are for V_{CC} = 5V, T_A = 25°C.

Parameter Symbol	Access Parameter Description	Test Conditions	'S409			'S409-2			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
t_{RHA}	Row address held from column select	Figure 4	10			10			ns
t_{RICL}	$\overline{RASIN}$ to $\overline{CAS}$ output delay (Mode 5)	Figures 7, 10	95	125	160	75	100	130	ns
t_{RICL}	$\overline{RASIN}$ to $\overline{CAS}$ output delay (Mode 6)	Figures 7, 10, 11	80	105	140	65	90	115	ns
t_{RICH}	$\overline{RASIN}$ to $\overline{CAS}$ output delay (Mode5)	Figures 7, 10	36	48	60	36	48	60	ns
t_{RICH}	$\overline{RASIN}$ to $\overline{CAS}$ output delay (Mode 6)	Figures 7, 10, 11	50	63	80	50	63	80	ns
t_{RCDL}	$\overline{RAS}$ to $\overline{CAS}$ output delay (Mode 5)	Figures 7, 10		98	125		75	100	ns
t_{RCDL}	$\overline{RAS}$ to $\overline{CAS}$ output delay (Mode 6)	Figures 7, 10, 11		78	105		65	85	ns
t_{RCDH}	$\overline{RAS}$ to $\overline{CAS}$ output delay (Mode 5)	Figures 7, 10		27	40		27	40	ns
t_{RCDH}	$\overline{RAS}$ to $\overline{CAS}$ output delay (Mode 6)	Figures 7, 10		40	65		40	65	ns
t_{CCDH}	$\overline{CASIN}$ to $\overline{CAS}$ output delay (Mode 6)	Figure 11	40	54	70	40	54	70	ns
t_{RCV}	$\overline{RASIN}$ to column address valid (Mode 5)	Figures 7, 10		90	120		80	105	ns
t_{RCV}	$\overline{RASIN}$ to column address valid (Mode 6)	Figures 7, 10, 11		75	105		70	90	ns
t_{RPDL}	$\overline{RASIN}$ to $\overline{RAS}$ delay	Figures 4, 5, 7, 10, 11	20	27	35	20	27	35	ns
t_{RPDH}	$\overline{RASIN}$ to $\overline{RAS}$ delay	Figures 4, 5, 7, 10, 11	15	23	32	15	23	32	ns
t_{APDL}	Address input to output low delay	Figures 4, 5, 7, 10, 11		25	40		25	40	ns
t_{APDH}	Address input to output high delay	Figures 4, 5, 7, 10, 11		25	40		25	40	ns
t_{SPDL}	Address strobe to address output low	Figures 4, 5		40	60		40	60	ns
t_{SPDH}	Address strobe to address output high	Figures 4, 5		40	60		40	60	ns
t_{WPDL}	$\overline{WIN}$ to $\overline{WE}$ output delay	Figure 5	15	25	30	15	25	30	ns
t_{WPD}	$\overline{WIN}$ to $\overline{WE}$ output delay	Figure 5	15	30	60	15	30	60	ns
t_{CRS}	$\overline{CASIN}$ setup time to $\overline{RASIN}$ high (Mode 6)	Figure 11	35			35			ns

SWITCHING CHARACTERISTICS (Cont'd.)

Parameter Symbol	Access Parameter Description	Test Conditions	'S409			'S409-2			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
t_{CPDL}	$\overline{CASIN}$ to $\overline{CAS}$ delay (R/C low in Mode 4)	Figure 5	32	41	58	32	41	58	ns
t_{CPDH}	$\overline{CASIN}$ to $\overline{CAS}$ delay	Figure 5	25	39	50	25	39	50	ns
t_{RCC}	Column select to column address valid	Figure 4		40	58		40	58	ns
t_{RCR}	Row select to row address valid	Figures 4, 5		40	58		40	58	ns
t_{RAH}	Row address hold time (Mode 5)	Figures 7, 10	30			20			ns
t_{RAH}	Row address hold time (Mode 6)	Figures 7, 10, 11	20			12			ns
t_{ASC}	Column address setup time (Mode 5)	Figures 7, 10	8			3			ns
t_{ASC}	Column address setup time (Mode 6)	Figures 7, 10, 11	6			3			ns
t_{DIF1}	Maximum ($t_{RPDL} - t_{RHA}$) (Mode 4)				15			15	ns
t_{DIF2}	Maximum $t_{RCC} - t_{CPDL}$) (Mode 4)				15			15	ns

Parameter Symbol	Refresh Parameter Description	Test Conditions	'S409			'S409-2			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
t_{FRQL}	RFCK low to forced $\overline{RFRQ}$ low	C_L = 50 pF, Figure 6		20	34		20	34	ns
t_{RFQH}	RGCK low to force $\overline{RFRQ}$ high	C_L = 50 pF, Figure 6		50	75		50	75	ns
t_{RGRL}	RGCK low to $\overline{RAS}$ low	Figure 6	36	65	95	36	65	95	ns
t_{RGRH}	RGCK low to $\overline{RAS}$ high	Figure 6	40	60	85	40	60	85	ns
t_{RFRH}	$\overline{RFSH}$ high to $\overline{RAS}$ high (encoding forced RFSH)	See Mode 1 description	55	80	110	55	80	110	ns
t_{CSCT}	$\overline{CS}$ high to $\overline{RFSH}$ counter valid	Figure 6		55	70		55	70	ns
t_{CTL}	RF I/O low to counter outputs all low	Figure 3			100			100	ns
t_{RFPDL}	$\overline{RASIN}$ to $\overline{RAS}$ delay during refresh	Figures 3, 6	35	50	70	35	50	70	ns
t_{RFPDH}	$\overline{RASIN}$ to $\overline{RAS}$ delay during refresh	Figures 3, 6	30	40	55	30	40	55	ns
t_{RFLCT}	$\overline{RFSH}$ low to counter address valid	$\overline{CS}$ = X, Figures 3,6,8		47	60		47	60	ns
t_{RFHRV}	$\overline{RFSH}$ high to row address valid	Figures 3, 6		45	60		45	60	ns

SWITCHING CHARACTERISTICS (Cont'd.)

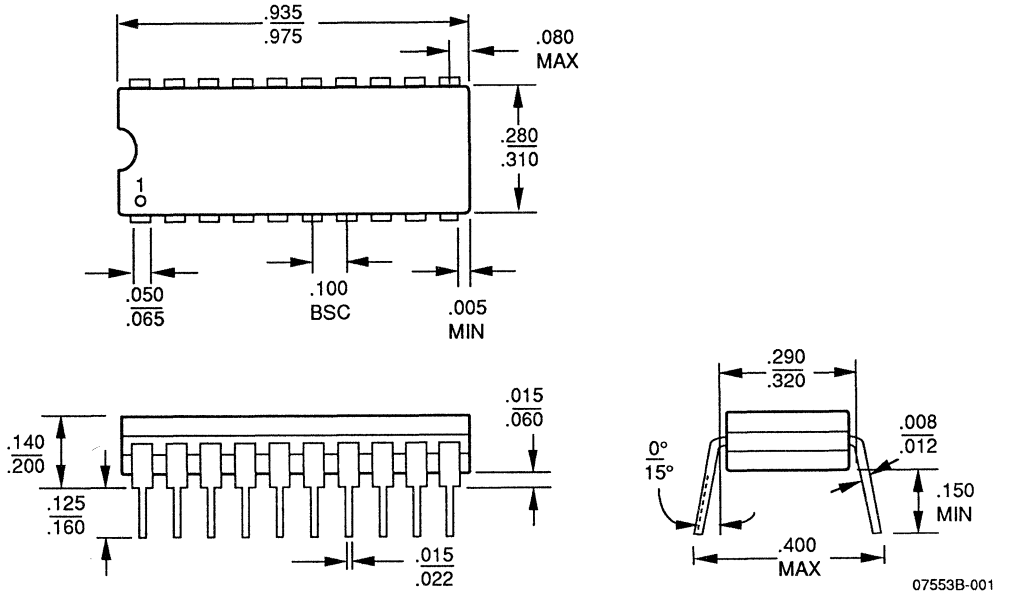
Parameter Symbol	Refresh Parameter Description	Test Conditions	'S409			'S409-2			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
t_{ROHNC}	$\overline{RAS}$ high to new count valid	Figures 3, 8		30	55		30	55	ns
t_{RLEOC}	$\overline{RASIN}$ low to end-of-count low	C_L = 50 pF, Figure 3			80			80	ns
t_{RHEOC}	$\overline{RASIN}$ high to end-of-count high	C_L = 50 pF, Figure 3			80			80	ns
t_{RGEOB}	RGCK low to end-of-burst low	C_L = 50 pF, Figure 8			95			95	ns
t_{MCEOB}	Mode change to end-of-burst high	C_L = 50 pF, Figure 8			75			75	ns

Parameter Symbol	Operating Parameter Description	Test Conditions	'S409			'S409-2			Unit
			Min.	Typ.	Max.	Min.	Typ.	Max.	
t_{ASA}	Address setup time to ADS	Figures 4, 5, 7, 10, 11	15			15			ns
t_{AHA}	Address hold time from ADS	Figures 4, 5, 7, 10, 11	15			15			ns
t_{ADS}	Address strobe pulse width	Figures 4, 5, 7, 10, 11	30			30			ns
$t_{RASINL,H}$	Pulse width of $\overline{RASIN}$ during refresh	Figure 3	50			50			ns
t_{RST}	Counter reset pulse width	Figure 3	70			70			ns
$t_{RFCKL,H}$	Minimum pulse width of RFCK	Figure 6	100			100			ns
T	Period of $\overline{RAS}$ generator clock	Figure 6	100			100			ns
t_{RGCKL}	Minimum pulse width low of RGCK	Figure 6	35			35			ns
t_{RGCKH}	Minimum pulse width high of RGCK	Figure 6	35			35			ns
t_{CSRL}	$\overline{CS}$ low to access $\overline{RASIN}$ low	See Mode 5 description	10			10			ns
t_{RFSRG}	$\overline{RFSH}$ low set-up to RGCK low (Mode 1)	See Mode 1 description	35			35			ns
t_{RQHRF}	$\overline{RFSH}$ hold time from $\overline{RFRQ}$ (RF I/O)	Figure 6	2T			2T			ns

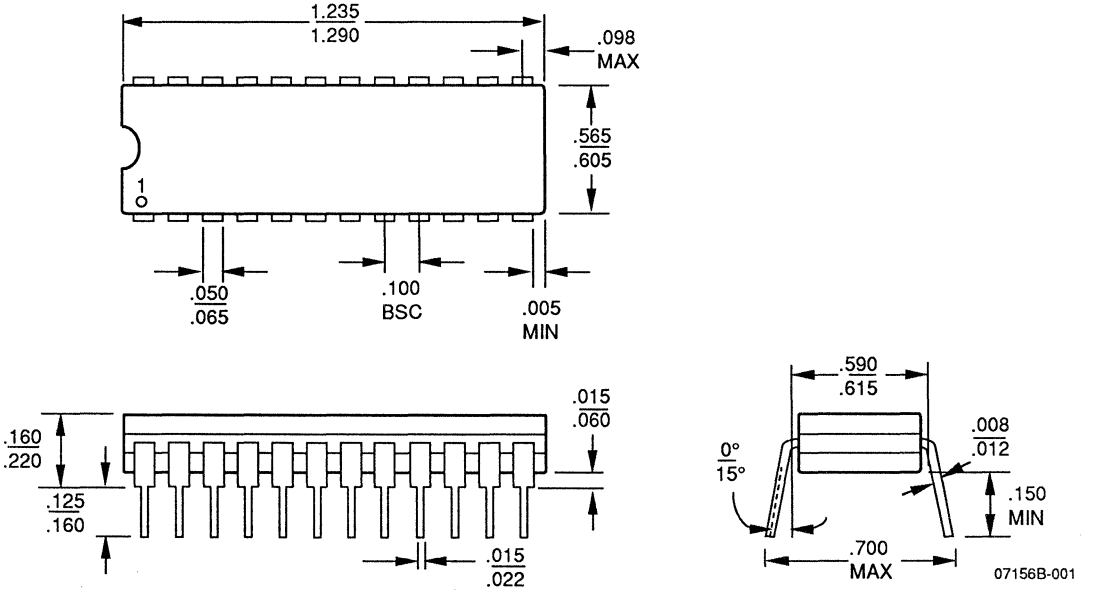
CHAPTER 7
Physical Dimensions

PHYSICAL DIMENSIONS*

CD020

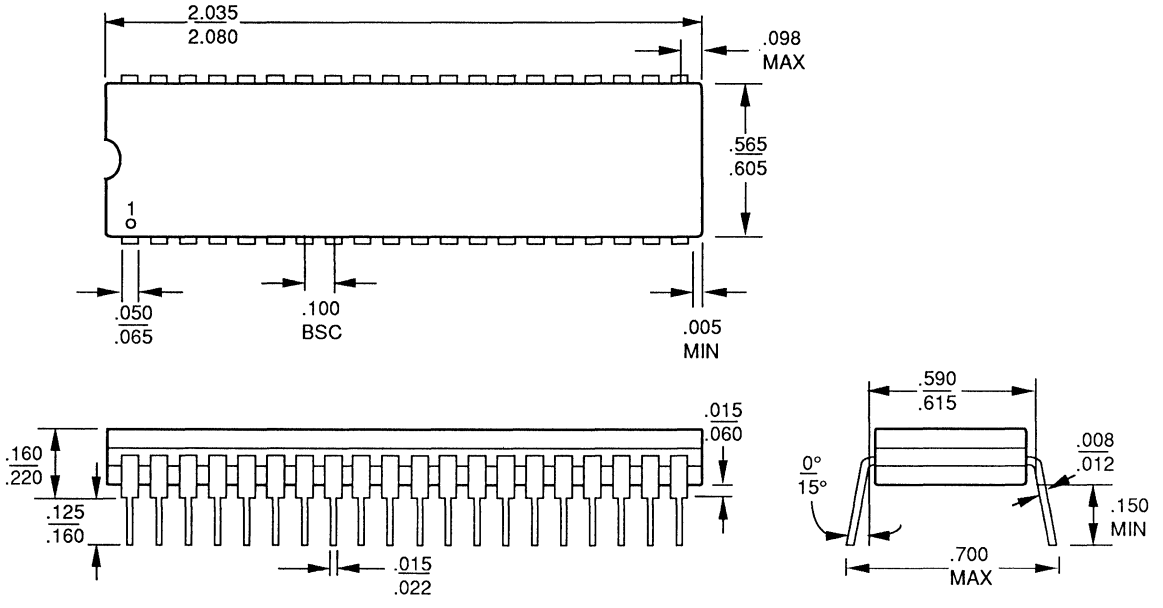


CD 024



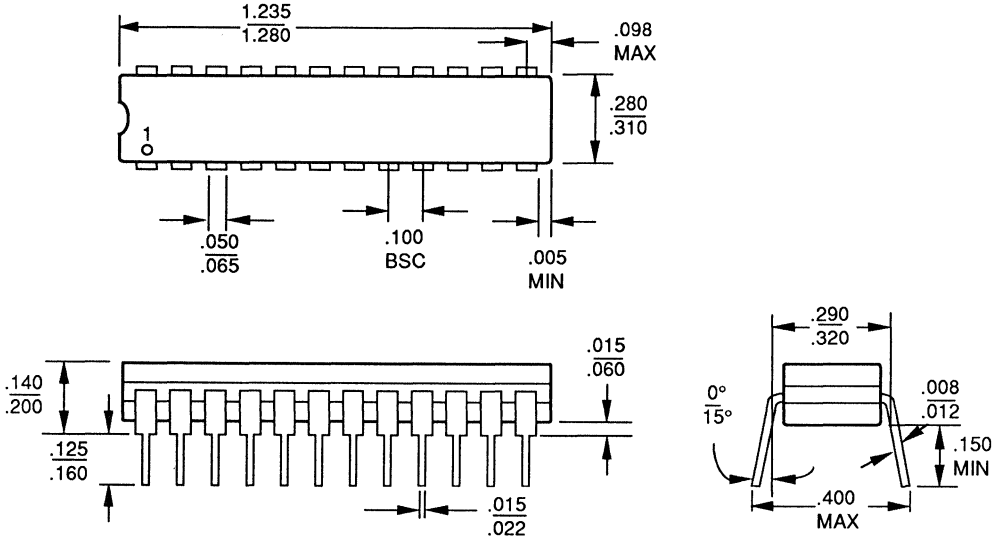
*For reference only. All measurements measured in inches. BSC is an ANSI standard for Basic Space Centering.

CD 040



06824C

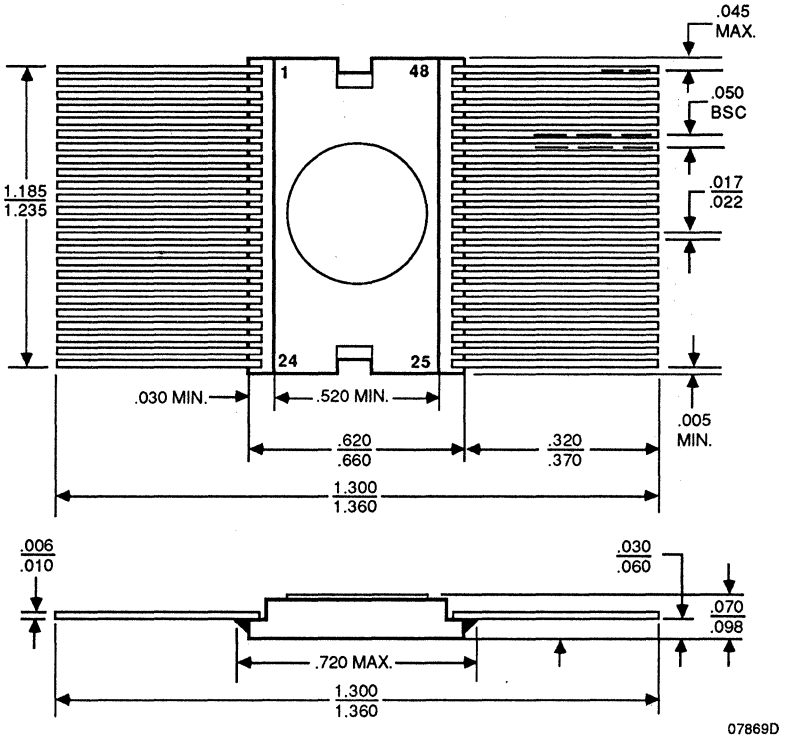
CD3024 **



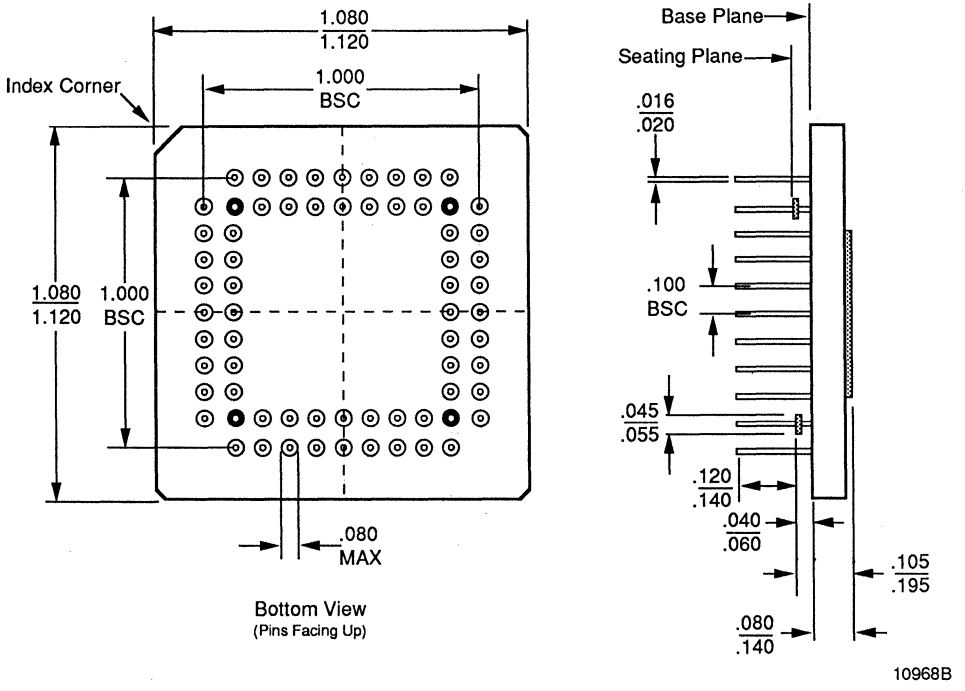
06850C

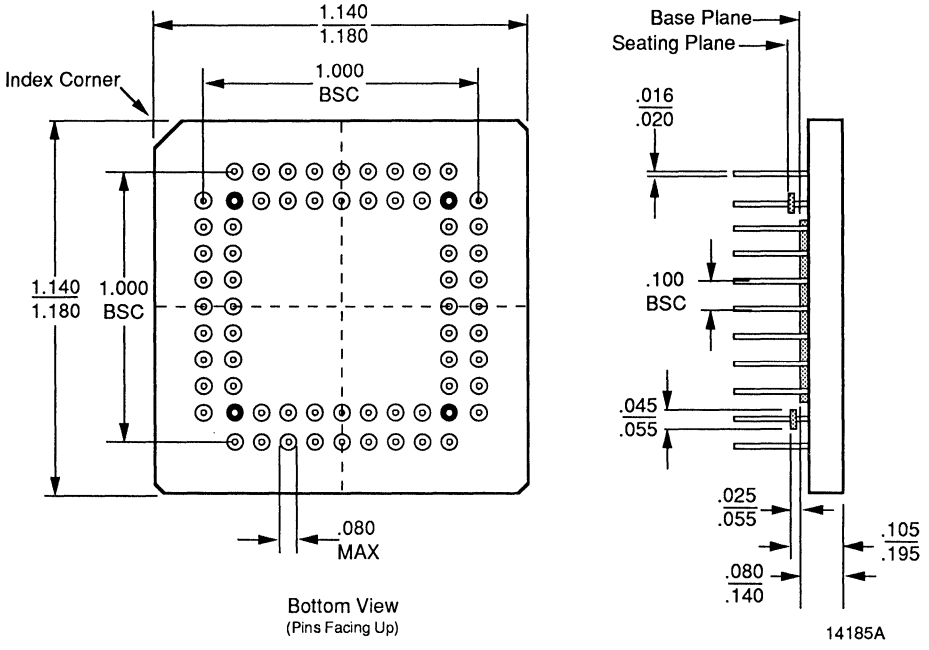
**1/2-Pin Version also available

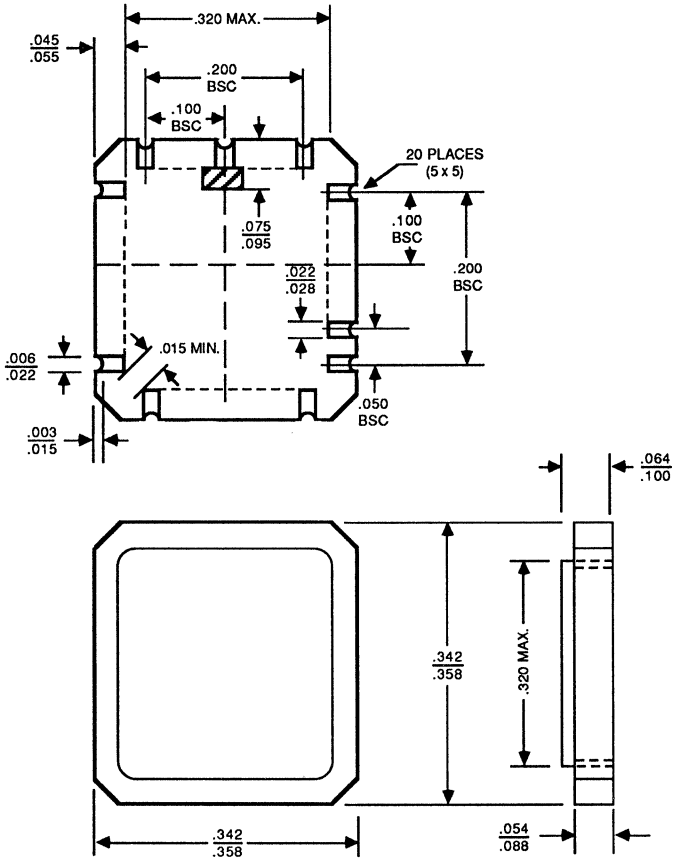
CFT048

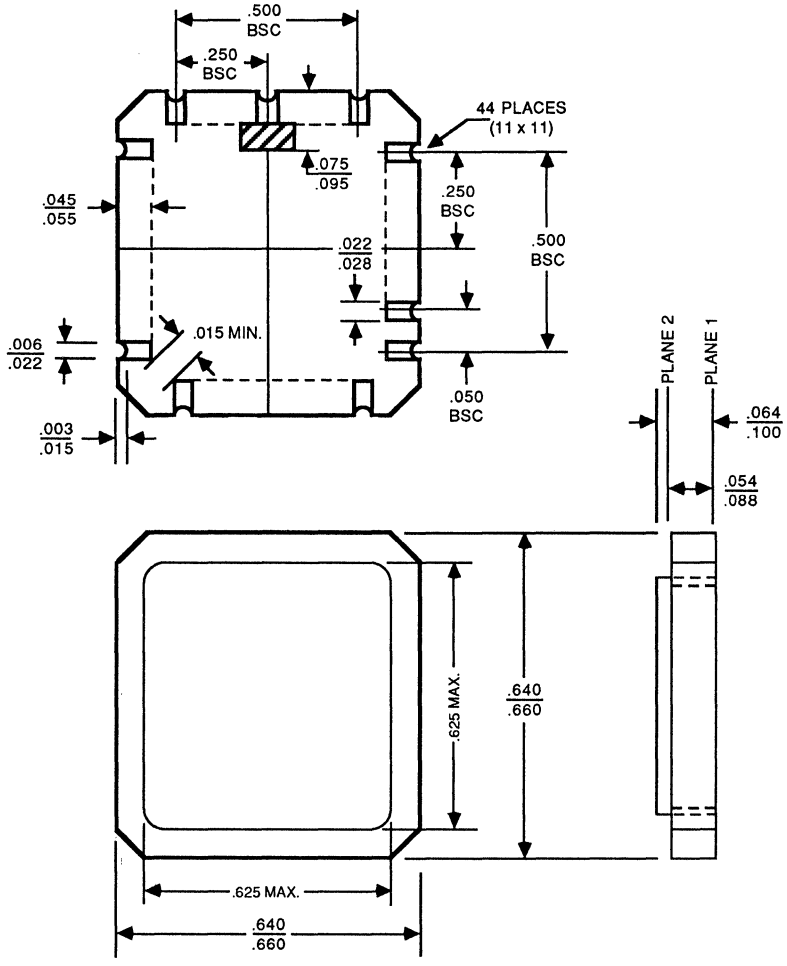


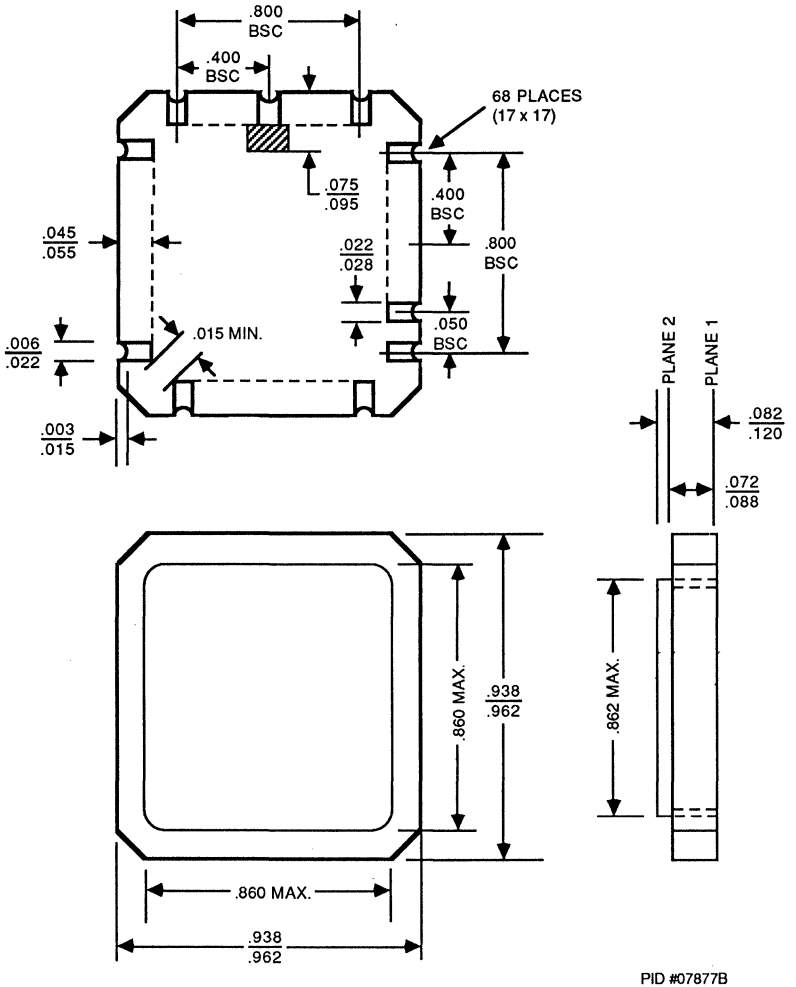
CGU068



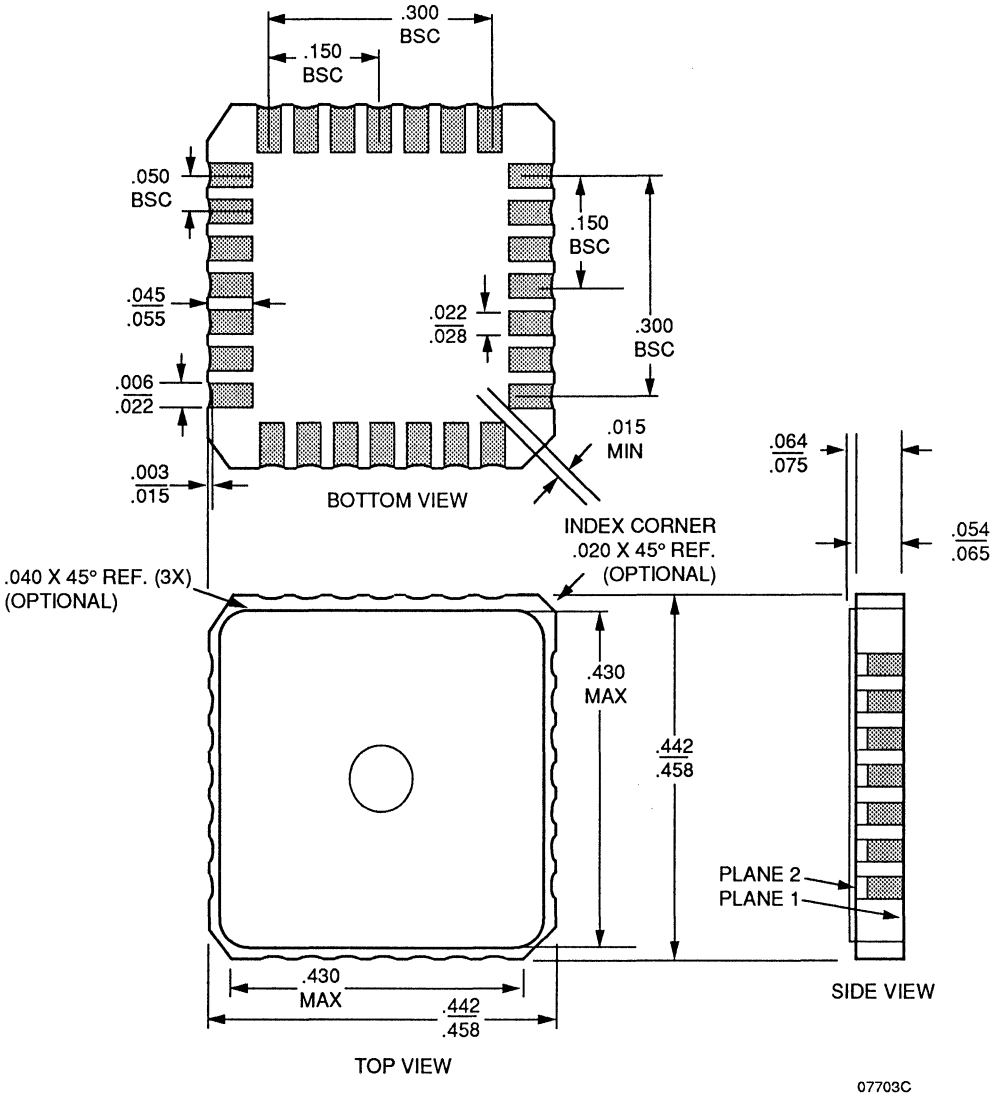

1.140
1.180
Index Corner
1.000
BSC
1.140
1.180
1.000
BSC
.080
MAX
Bottom View
(Pins Facing Up)
Base Plane
Seating Plane
.016
.020
.100
BSC
.045
.055
.025
.055
.080
.140
.105
.195
14185A


.320 MAX.
.045
.055
.200
BSC
.100
BSC
20 PLACES
(5 x 5)
.075
.095
.100
BSC
.200
BSC
.022
.028
.006
.022
.015 MIN.
.050
BSC
.003
.015
.064
.100
.342
.358
.320 MAX.
.342
.358
.054
.088
PID #07318C

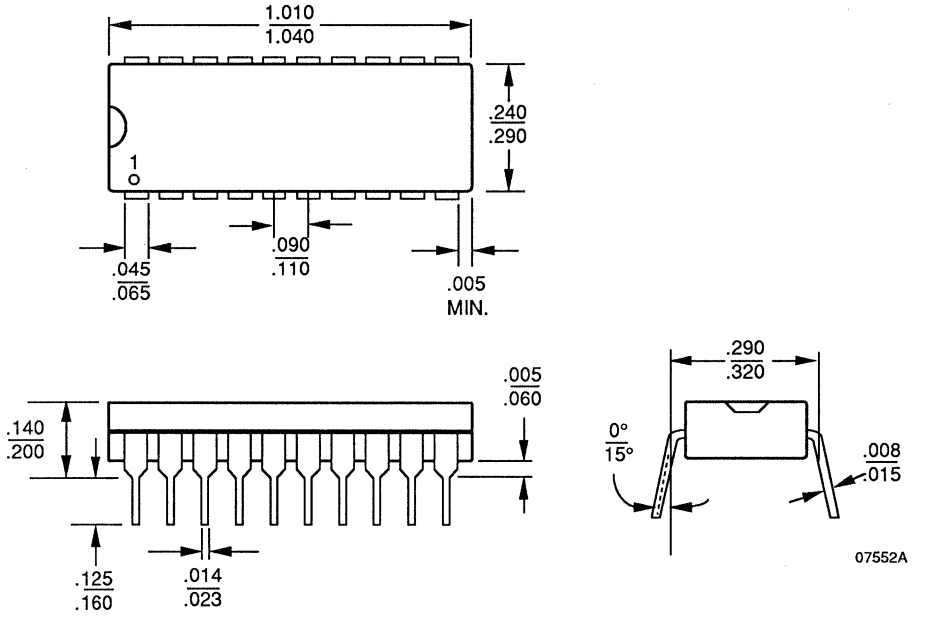

.500
BSC
.250
BSC
44 PLACES
(11 x 11)
.075
.095
.250
BSC
.500
BSC
.045
.055
.022
.028
.015 MIN.
.006
.022
.050
BSC
.003
.015
PLANE 2
PLANE 1
.064
.100
.054
.088
.625 MAX.
.640
.660
.625 MAX.
.640
.660
PID #06825E


.800
BSC
.400
BSC
68 PLACES
(17 x 17)
.075
.095
.400
BSC
.045
.055
.022
.028
.800
BSC
.050
BSC
.006
.022
.015 MIN.
.003
.015
PLANE 2
PLANE 1
.082
.120
.072
.088
.860 MAX.
.938
.962
.862 MAX.
.860 MAX.
.938
.962

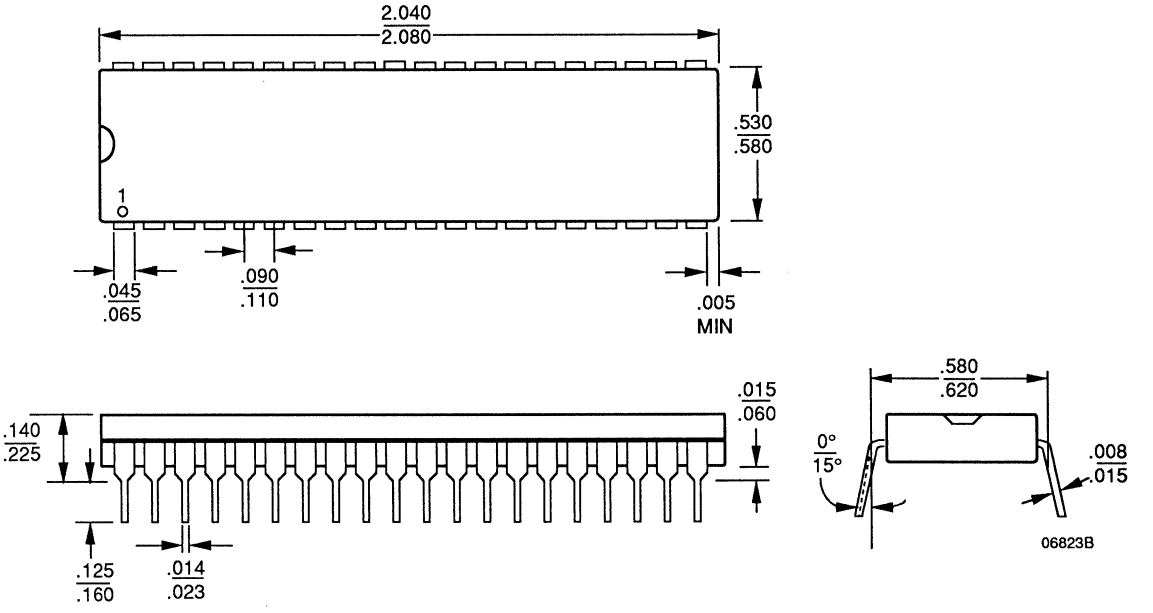
PID #07877B


.300
BSC
.150
BSC
.050
BSC
.045
.055
.006
.022
.003
.015
.022
.028
.150
BSC
.300
BSC
.015
MIN
BOTTOM VIEW
INDEX CORNER
.020 X 45° REF.
(OPTIONAL)
.040 X 45° REF. (3X)
(OPTIONAL)
.064
.075
.054
.065
.430
MAX
.442
.458
PLANE 2
PLANE 1
.430
MAX
.442
.458
TOP VIEW
SIDE VIEW
07703C

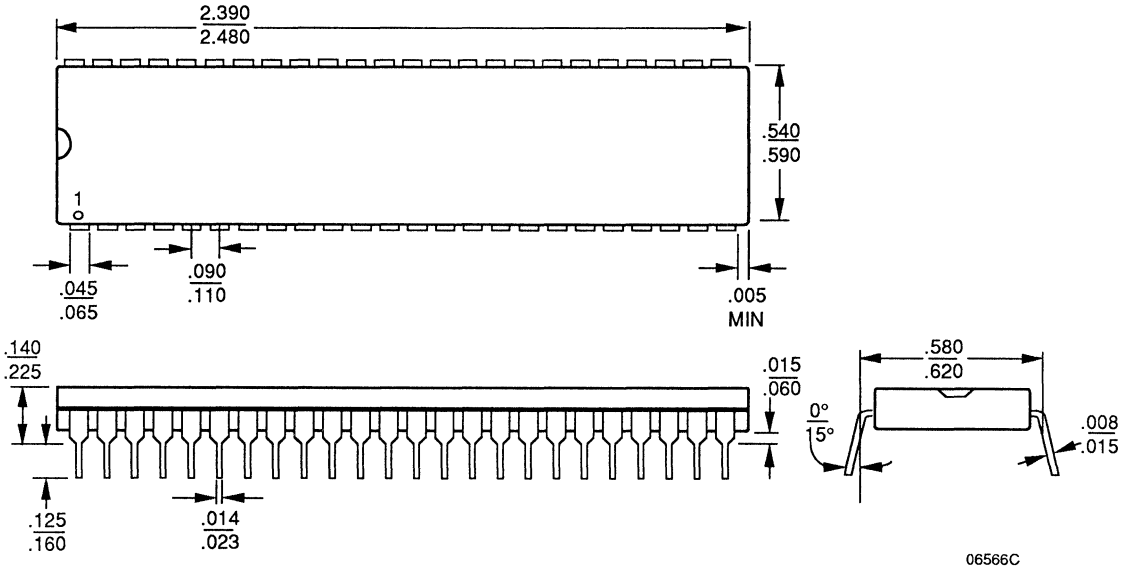
PD 020


1.010
1.040
.240
.290
1
.045
.065
.090
.110
.005
MIN.
.005
.060
.140
.200
.125
.160
.014
.023
.290
.320
0°
15°
.008
.015
07552A

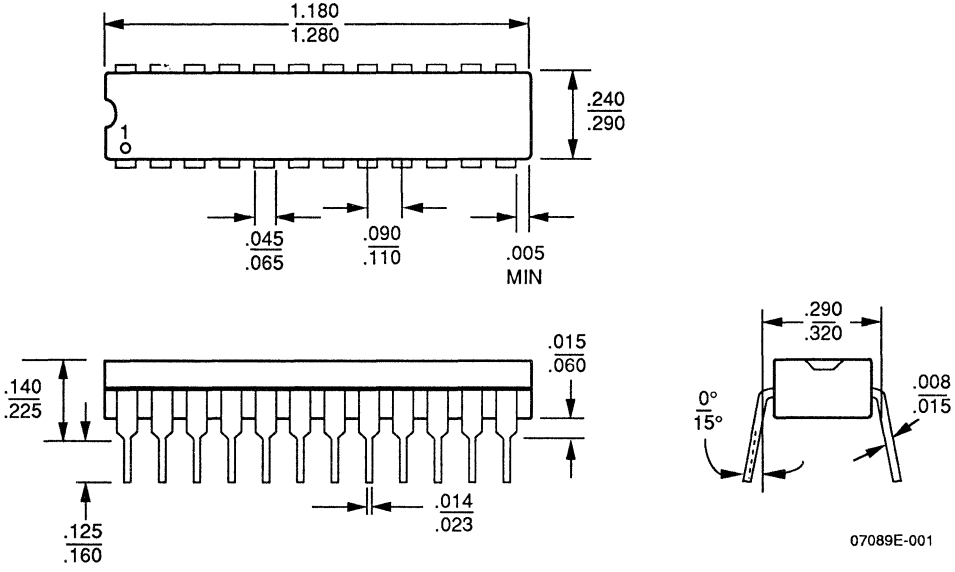
PD 040


2.040
2.080
.530
.580
1
.045
.065
.090
.110
.005
MIN
.015
.060
.140
.225
.125
.160
.014
.023
.580
.620
0°
15°
.008
.015
06823B

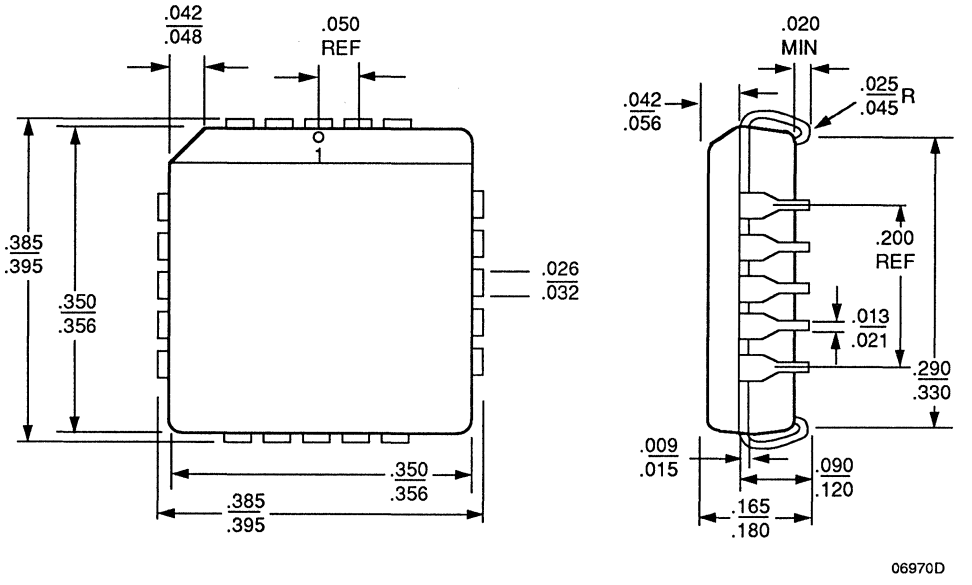
PD 048



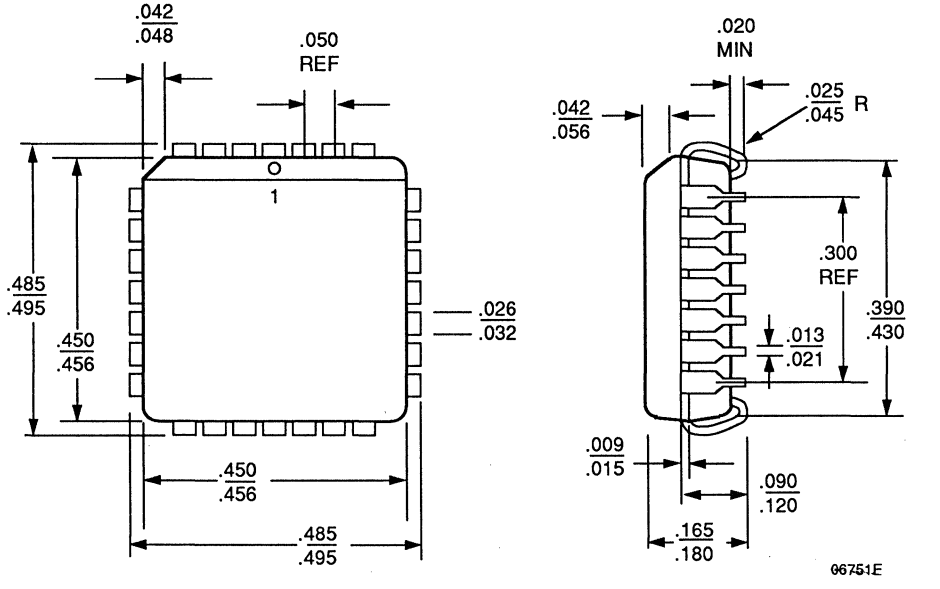
PD3024



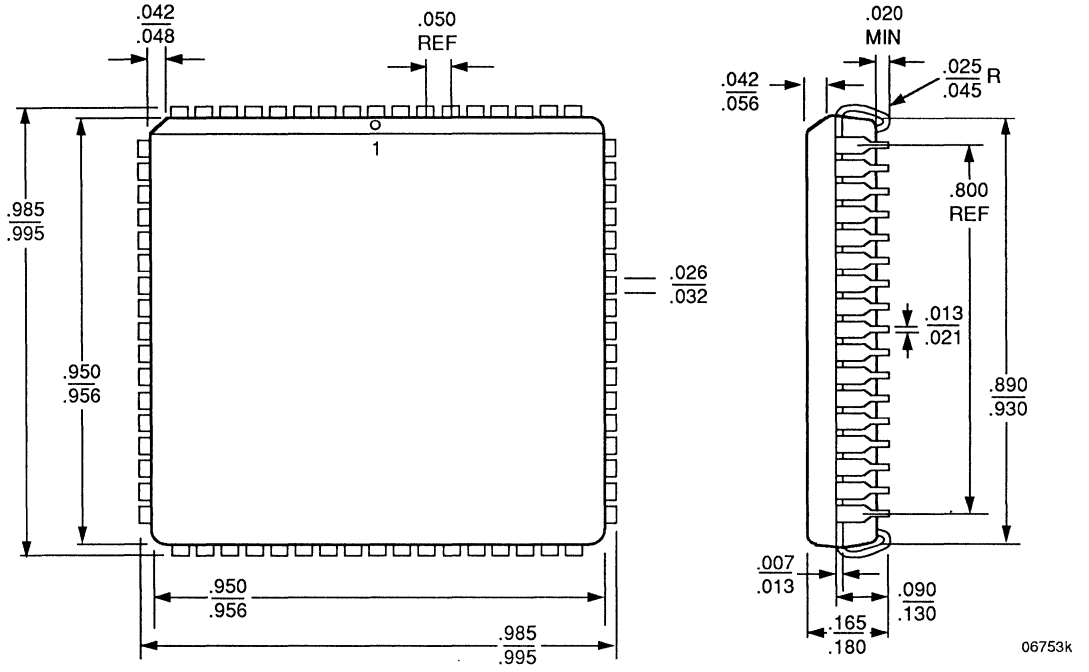
PL 020


.042
.048
.050
REF
1
.385
.395
.350
.356
.026
.032
.350
.356
.385
.395
.020
MIN
.042
.056
.025
.045
R
.200
REF
.013
.021
.290
.330
.009
.015
.090
.120
.165
.180
06970D

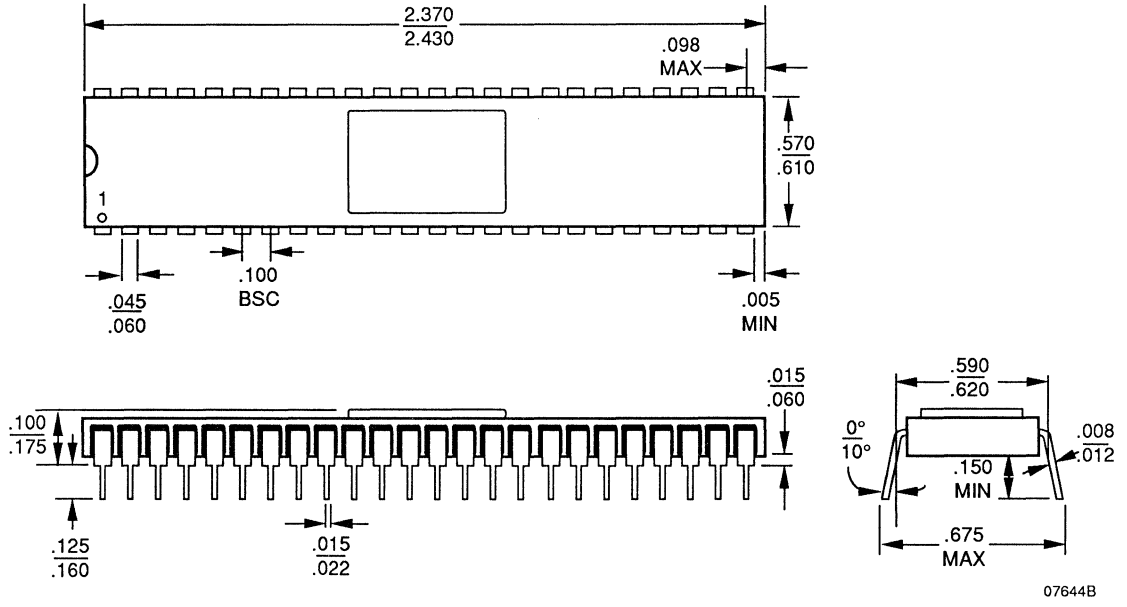
PL 028


.042
.048
.050
REF
1
.485
.495
.450
.456
.026
.032
.450
.456
.485
.495
.020
MIN
.042
.056
.025
.045
R
.300
REF
.390
.430
.013
.021
.009
.015
.090
.120
.165
.180
06751E

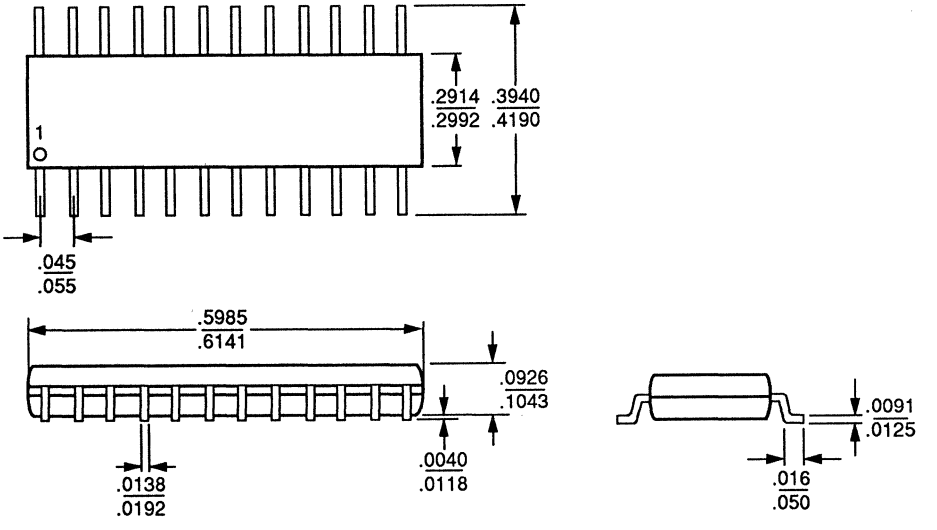
PL 068


.042
.048
.050
REF
.020
MIN
.042
.056
.025
.045
R
1
.985
.995
.800
REF
.026
.032
.013
.021
.950
.956
.890
.930
.007
.013
.090
.130
.950
.956
.165
.180
.985
.995
06753k

SD 048


2.370
2.430
.098
MAX
.570
.610
1
.100
BSC
.045
.060
.005
MIN
.015
.060
.100
.175
.590
.620
0°
10°
.008
.012
.150
MIN
.125
.160
.015
.022
.675
MAX
07644B

SO 024



09310B

APPENDIX
Behavioral Simulation Models from Logic Automation, Inc.

BEHAVIORAL SIMULATION MODELS FROM LOGIC AUTOMATION, INC.

- Fast and Accurate
- Extensive Error Messages
- Over 5000 Device Models
- Compatible with Leading Simulators
- SmartModel* Windows, a System Emulation Capability for Viewing and Changing Register Contents

SmartModels are behavioral language-simulation models with built-in expert assistance, used for board- and system-level simulation. Models of thousands of devices are available ranging from complex microprocessors to memories, PLDs, and TTL logic. Simulation provides several benefits to the user including faster design time, lower prototype costs, and higher quality product.

SmartModels increase designer productivity with extensive messages including usage checks and timing checks. The SmartModel usage checks look for undefined interrupts, uninitialized registers, illegal conditions—any misuse of the component that is likely to slow or stall the design process. These are reported as thoroughly as possible, pinpointing the error by documenting the design sheet, part instance, pin name, and the time of occurrence, so the error can be eliminated immediately.

SmartModel timing checks look for violations of timing specifications like set-up, hold, and recovery. Messages cite the required specification as well as the violation, along with the pin location and simulation time. The result is that the designer need not interrupt the system verification to search component data books for specifications. The necessary data is right there, built into the simulation models.

Dynamic Memory Management devices supported include:

- Am29C668, Am29C668-1
- Am29C660, Am29C660A, Am29C660B, Am29C660C, Am29C660D
- Am29C983, Am29C983A
- Am29C985

Host Systems supported now include:

- Mentor Graphics
- Valid Logic
- Gateway Design Automation
- HHB Systems
- Vantage
- Hewlett Packard
- AT&T (proprietary)

*SmartModel is a registered trademark of Logic Automation, Inc.

Host Systems under development include:

- Daisy
- Teradyne
- Viewlogic
- Silicon Compiler Systems

The following factory contacts at Logic Automation may be contacted for price and availability information:

Corporate Headquarters:
Logic Automation Incorporated
Rick Denker
19500 NW Gibbs Drive
P.O. Box 310
Beaverton, OR 97075
Phone: (503) 690-6900
FAX: (503) 690-6906
Electronic Bulletin Board: (503) 690-6907

European Sales Office:
Logic Automation (Europe), Ltd.
Jeff Dean
Farley Hall, London Road
Bracknell
Berkshire RG12 5EU
United Kingdom
Phone: 44 (0) 344 863230
FAX: 44 (0) 344 863999
Telex: 849999 (NET NYN G)

Sales Offices

North American

ALABAMA		(205) 882-9122
ARIZONA		(602) 242-4400
CALIFORNIA,		
Culver City		(213) 645-1524
Newport Beach		(714) 752-6262
Roseville		(916) 786-6700
San Diego		(619) 560-7030
San Jose		(408) 452-0500
Woodland Hills		(818) 992-4155
CANADA, Ontario,		
Kanata		(613) 592-0060
Willowdale		(416) 224-5193
COLORADO		(303) 741-2900
CONNECTICUT		(203) 264-7800
FLORIDA,		
Clearwater		(813) 530-9971
Ft. Lauderdale		(305) 776-2001
Orlando (Casselberry)		(407) 830-8100
GEORGIA		(404) 449-7920
ILLINOIS,		
Chicago (Itasca)		(312) 773-4422
Naperville		(312) 505-9517
KANSAS		(913) 451-3115
MARYLAND		(301) 796-9310
MASSACHUSETTS		(617) 273-3970
MICHIGAN		(313) 347-1522
MINNESOTA		(612) 938-0001
NEW JERSEY,		
Cherry Hill		(609) 662-2900
Parsippany		(201) 299-0002
NEW YORK,		
Liverpool		(315) 457-5400
Poughkeepsie		(914) 471-8180
Rochester		(716) 272-9020
NORTH CAROLINA		(919) 878-8111
OHIO,		
Columbus (Westerville)		(614) 891-6455
OREGON		(503) 245-0080
PENNSYLVANIA		(215) 398-8006
SOUTH CAROLINA		(803) 772-6760
TEXAS,		
Austin		(512) 346-7830
Dallas		(214) 934-9099
Houston		(713) 785-9001
UTAH		(801) 264-2900

International

BELGIUM, Bruxelles	TEL	(02) 771-91-42
	FAX	(02) 762-37-12
	TLX	846-61028
FRANCE, Paris	TEL	(1) 49-75-10-10
	FAX	(1) 49-75-10-13
	TLX	263282F
WEST GERMANY,		
Hannover area	TEL	(0511) 736085
	FAX	(0511) 721254
	TLX	922850
München	TEL	(089) 4114-0
	FAX	(089) 406490
	TLX	523883
Stuttgart	TEL	(0711) 62 33 77
	FAX	(0711) 625187
	TLX	721882
HONG KONG,	TEL	852-5-8654525
Wanchai	FAX	852-5-8654335
	TLX	67955AMDAPHX
ITALY, Milan	TEL	(02) 3390541
		(02) 3533241
	FAX	(02) 3498000
	TLX	843-315286
JAPAN,		
Kanagawa	TEL	462-47-2911
	FAX	462-47-1729
Tokyo	TEL	(03) 346-7550
	FAX	(03) 342-5196
	TLX	J24064AMDTKOJ
Osaka	TEL	06-243-3250
	FAX	06-243-3253

International (Continued)

KOREA, Seoul	TEL	822-784-0030
	FAX	822-784-8014
LATIN AMERICA,		
Ft. Lauderdale	TEL	(305) 484-8600
	FAX	(305) 485-9736
	TLX	5109554261 AMDFTL
NORWAY, Hovik	TEL	(03) 010156
	FAX	(02) 591959
	TLX	79079HBCN
SINGAPORE	TEL	65-3481188
	FAX	65-3480161
	TLX	55650 AMDMMI
SWEDEN,		
Stockholm	TEL	(08) 733 03 50
(Sundbyberg)	FAX	(08) 733 22 85
	TLX	11602
TAIWAN	TEL	886-2-7213393
	FAX	886-2-7723422
	TLX	886-2-7122066
UNITED KINGDOM,		
Manchester area	TEL	(0925) 828008
(Warrington)	FAX	(0925) 827693
	TLX	851-628524
London area	TEL	(0483) 740440
(Woking)	FAX	(0483) 756196
	TLX	851-859103

North American Representatives

CANADA		
Burnaby, B.C.		
DAVETEK MARKETING		(604) 430-3680
Calgary, Alberta		
DAVETEK MARKETING		(403) 291-4984
Kanata, Ontario		
VITEL ELECTRONICS		(613) 592-0060
Mississauga, Ontario		
VITEL ELECTRONICS		(416) 676-9720
Lachine, Quebec		
VITEL ELECTRONICS		(514) 636-5951
IDAHO		
INTERMOUNTAIN TECH MKTG, INC		(208) 888-6071
ILLINOIS		
HEARTLAND TECH MKTG, INC		(312) 577-9222
INDIANA		
Huntington - ELECTRONIC MARKETING CONSULTANTS, INC		(317) 921-3450
Indianapolis - ELECTRONIC MARKETING CONSULTANTS, INC		(317) 921-3450
IOWA		
LORENZ SALES		(319) 377-4666
KANSAS		
Merriam – LORENZ SALES		(913) 384-6556
Wichita – LORENZ SALES		(316) 721-0500
KENTUCKY		
ELECTRONIC MARKETING CONSULTANTS, INC		(317) 921-3452
MICHIGAN		
Birmingham - MIKE RAICK ASSOCIATES		(313) 644-5040
Holland – COM-TEK SALES, INC		(616) 392-7100
Novi – COM-TEK SALES, INC		(313) 344-1409
MISSOURI		
LORENZ SALES		(314) 997-4558
NEBRASKA		
LORENZ SALES		(402) 475-4660
NEW MEXICO		
THORSON DESERT STATES		(505) 293-8555
NEW YORK		
East Syracuse – NYCOM, INC		(315) 437-8343
Woodbury – COMPONENT CONSULTANTS, INC		(516) 364-8020
OHIO		
Centerville – DOLFUSS ROOT & CO		(513) 433-6776
Columbus – DOLFUSS ROOT & CO		(614) 885-4844
Strongsville – DOLFUSS ROOT & CO		(216) 238-0300
PENNSYLVANIA		
DOLFUSS ROOT & CO		(412) 221-4420
PUERTO RICO		
COMP REP ASSOC, INC		(809) 746-6550
UTAH, R^2 MARKETING		(801) 595-0631
WASHINGTON		
ELECTRA TECHNICAL SALES		(206) 821-7442
WISCONSIN		
HEARTLAND TECH MKTG, INC		(414) 792-0920

Advanced Micro Devices reserves the right to make changes in its product without notice in order to improve design or performance characteristics. The performance characteristics listed in this document are guaranteed by specific tests, guard banding, design and other practices common to the industry. For specific testing details, contact your local AMD sales representative. The company assumes no responsibility for the use of any circuits described herein.

Advanced Micro Devices, Inc. 901 Thompson Place, P.O. Box 3453, Sunnyvale, CA 94088, USA
Tel: (408) 732-2400 • TWX: 910-339-9280 • TELEX: 34-6306 • TOLL FREE: (800) 538-8450
APPLICATIONS HOTLINE TOLL FREE: (800) 222-9323 • (408) 749-5703

© 1990 Advanced Micro Devices, Inc.
2/2/90
B-32M-2/90-0 Printed in USA



ADVANCED MICRO DEVICES, INC.
901 Thompson Place
P.O. Box 3453
Sunnyvale,
California 94088-3453
(408) 732-2400
TWX: 910-339-9280
TELEX: 34-6306
TOLL-FREE
(800) 538-8450

APPLICATIONS HOTLINE
(800) 222-9323
(408) 749-5703

Printed in USA
11580A
B-31M-3/90-0