iAPX 43204, iAPX 43205
FAULT TOLERANT
BUS INTERFACE AND MEMORY CONTROL UNITS

- **Software Transparent Detection And Recovery From Any Single Point Failure**
- **Supports Up To 31 Processors For A Large Performance Range**
- **Configure From 1 To 8 Buses For High Bandwidth And Fault Tolerance**
- **Single, Dual, and Quad Redundant Configurations Tailor System Designs to Meet A Spectrum of Fault Tolerance And Cost Objectives**
- **VLSI System Simplifies Design With Low TTL Count**
- **Dynamic RAM Refresh with Error Correction and Scrubbing**

The 43204 Bus Interface Unit (BIU) and 43205 Memory Control Unit (MCU) are two VLSI devices that support the construction of fault tolerant, multiple processor 432 systems. Together they support: multiprocessor arbitration, dynamic RAM control, and ECC with a minimal amount of TTL. Fault tolerant systems can be built that tolerate the failure of any single component or bus. The BIU and MCU detect the failure and automatically switch to a redundant processor, bus, or memory. Hardware failures are completely masked from application software.

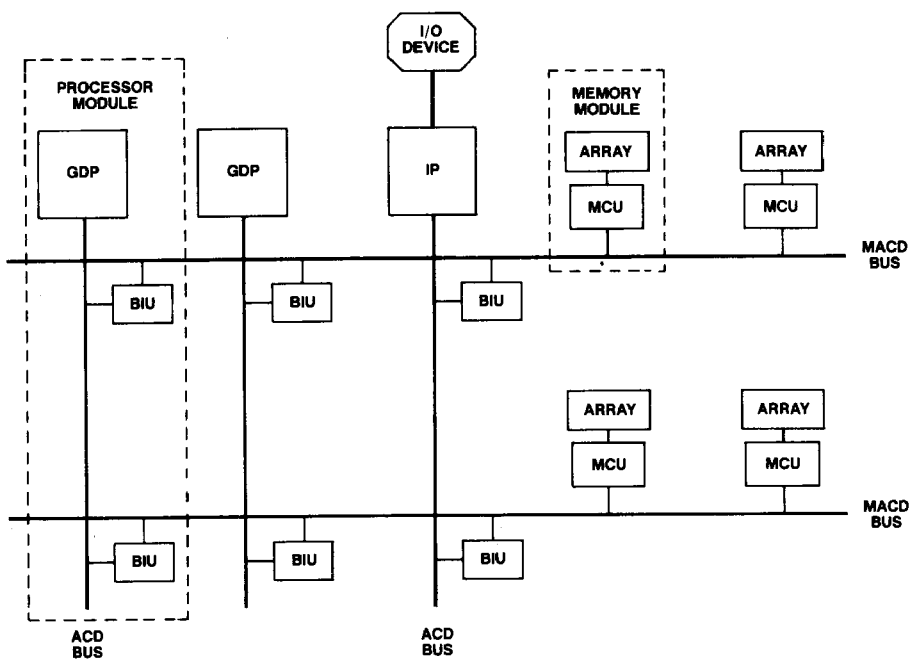


Figure 1. A 2 Bus System with 2 General Data Processors and 1 I/O Subsystem.

October 1983
ORDER NUMBER: 210963-002

INTRODUCTION

The first phase of the iAPX 432 program introduced two processor types: the General Data Processor (GDP) and the Interface Processor (IP). The GDP was implemented with two VLSI components: iAPX 43201 and iAPX 43202. The IP was implemented as a single VLSI component: the iAPX 43203. These three VLSI components implement the *processor architecture* for the iAPX 432. System builders have constructed multiple processor systems by surrounding the VLSI processors with discrete logic, which provided the interface to shared memory and the interprocessor communication paths. The method for interconnecting iAPX 432 processors and memories was unique for each system, since no standard had been defined.

This data sheet describes a pair of VLSI components: the iAPX 43204 Bus Interface Unit (BIU) and the iAPX 43205 Memory Control Unit (MCU) that form a unifying *interconnect architecture* for building iAPX 432 systems. Together, these components form the basis for constructing multiple-processor fault-tolerant iAPX 432 systems.

The iAPX 432 together with the BIU/MCU interconnect architecture provide:

- *Integrated fault tolerance.* The VLSI interconnect components (BIU/MCU) integrate all the detection and recovery logic required to build a system that can tolerate any single component failure.
- *Software transparent fault tolerance.* Hardware performs all fault detection and recovery functions transparent to application software. The machine never stops.
- *Configurability.* The BIU and MCU support a range of fault tolerance and performance options to meet a diverse set of cost, performance, and reliability needs.
- *Standard VLSI solution.* Very little external logic is required.
- *Reliable software.* The iAPX 432 system's "need to know" (capability) addressing confines errors, protecting the system from errant software.

The object-based architecture of the iAPX 432 provides a robust and flexible environment for cooperating, concurrent software systems. The iAPX 432 processors use a cooperating, self-dispatching mechanism to automatically share the workload between the available processors. The number of processors available in the system is transparent to software.

The BIU and the MCU extend the logical flexibility and robustness of the iAPX 432 processors into the physical implementation of iAPX 432 systems. The BIU and MCU allow the iAPX 432 hardware to modularly and transparently extend the processing power (from 1 to 63 modules of processors or memories), bus bandwidth (1 to 8 backplane buses), and fault-tolerant capabilities of the system. Figure 1 shows an example of a two bus three processor (2 GDPs + 1 IP) system.

As Figure 2 shows, an iAPX 432 system based on the interconnect architecture may be expanded gracefully. A system with one processor and one memory may be built with a single memory bus. Transparent multiprocessing may be achieved by simply adding processor modules. When additional memory is required, memory modules may be added onto the single memory bus. When more memory bandwidth is required, an additional memory bus(es) can be added. None of these alternative systems require any change to application software.

In an iAPX 432 system, each processor is unaware of the manner in which the memory address space is actually implemented. Hardware located in the BIUs determines how processor addresses are mapped to buses and memory systems.

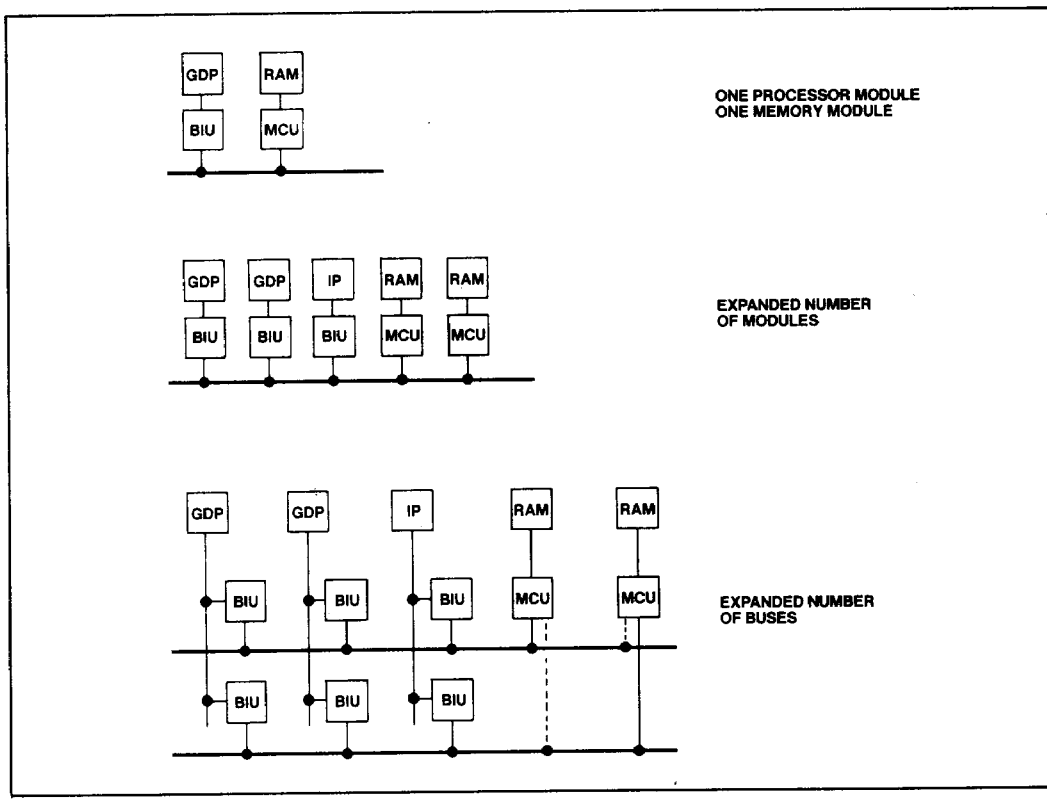


Figure 2. Modular Expansion

BUS INTERFACE UNIT

The Bus Interface Unit or BIU provides the switching function of the IAPX 432 interconnect system. That is, it accepts access requests from an iAPX 432 processor and, based on the physical address, it decides which memory bus(es) will be used to perform the access. The BIU is also responsible for arbitrating the usage of the memory bus. Finally, the BIU is responsible for propagating error information throughout the sytem.

MEMORY CONTROL UNIT

The Memory Control Unit or MCU interfaces memory storage arrays to the memory bus. The storage arrays will typically be constructed with high-density dynamic RAM (DRAM) components. All types of DRAMs are supported: 16K, 64K, 256K, even partially good components. The MCU manages the storage array as a logical collection of 32 data bits, 7 bits of error correcting code (ECC), and an optional spare bit. The MCU can automatically *refresh* the dynamic storage array. In addition, the MCU can *scrub* single-bit errors from the storage array as a background task. Scrubbing is accomplished by periodically reading the storage array, correcting all single-bit errors, and detecting and reporting all double-bit errors. The MCU accepts variable length data requests from the memory bus and performs the necessary access sequencing to read or write the data into the storage array. A modest amount of external logic is required to interface the MCU to the storage array RAMs — for simple configurations, as few as 12 external TTL packages are required.

MEMORY BUS

The memory bus (sometimes referred to as the MACD bus) provides the principal communication path, carrying all memory access requests and interprocessor communication. The memory bus connects BIUs to MCUs. Each node in the interconnect system tracks each operation on the memory bus to which it is attached. Thus, unlike most bus protocols, each BIU and MCU keeps track of all outstanding requests on the bus — not just the ones made by the BIU or MCU itself. Control for the bus is fully distributed; there is no centralized bus controller.

INTEGRATED FAULT TOLERANCE

BIUs and MCUs also form the basis for building *fault-tolerant* iAPX 432 systems. *Functional Redundancy Checking (FRC)* provides the low-level hardware support on which hardware fault-tolerant modules are constructed. In Figure 3, notice that a redundant processor module is formed by *replication* of the VLSI GDP and BIUs. A redundant memory module is formed by duplicating the VLSI MCU. The unshaded GDPs, BIUs, and MCUs act as *masters*. The shaded components act as *checkers*, which observe their master and report any disagreement they detect in the values the master produced.

When any error occurs, a special *error reporting network* notifies all nodes in the system of the discrepancy. Figure 4 illustrates the flow of error information in the interconnect system. In phase 1, an error is detected at a node in the interconnect system. The example illustrates an error detected at BIU(2,1); i.e., the BIU on memory bus 2 in processor

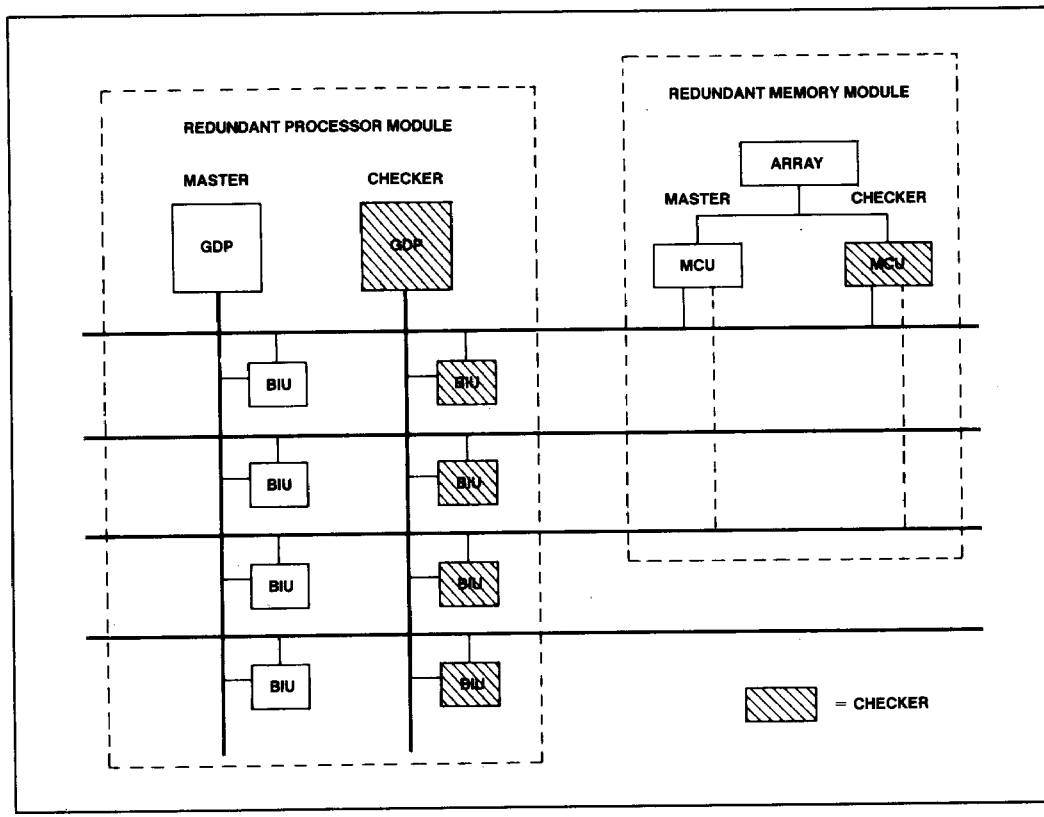


Figure 3. FRC Configuration Pairing

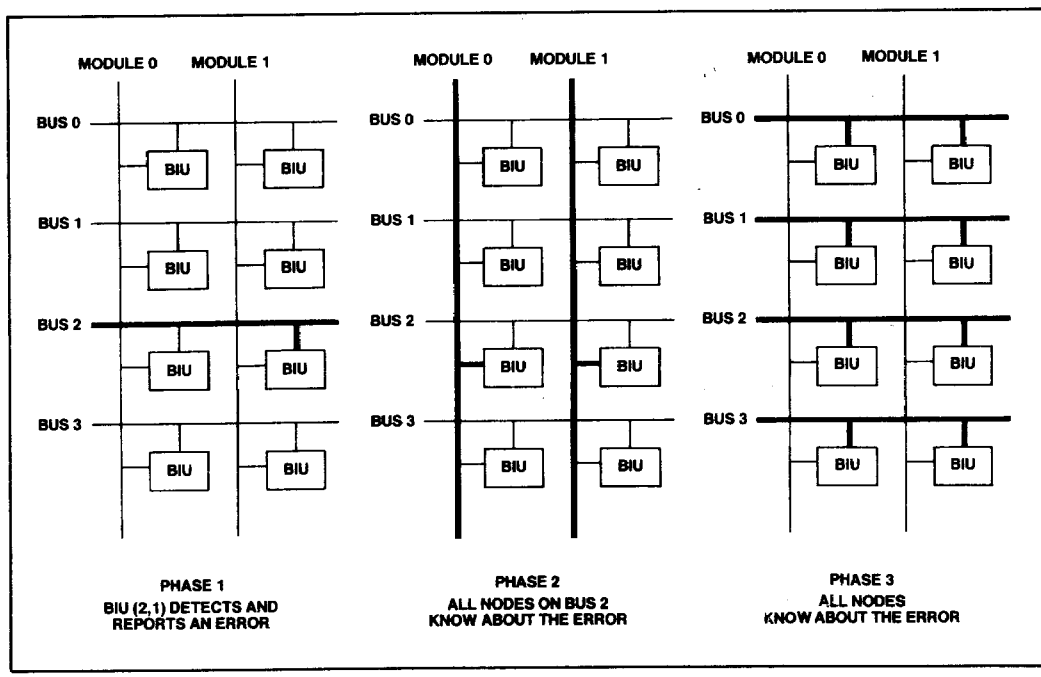


Figure 4. Three Phase Error Reporting Mechanism

module 1. The detecting component reports the error to all components attached to the same bus (a bold line indicates an active error reporting path). At this point, if all error reporting nodes are intact, all nodes have received the error message. In phase 2, all components that received the phase 1 error message *rebroadcast* the message along their module paths. Finally, in phase 3, each component that has received an error message rebroadcasts the message along its bus path. This second rebroadcast ensures that all nodes receive the error message even if a single module or bus error report line has failed. At the end of phase 3, all interconnect components in the system have been informed of the error. The actual error reporting paths are separate from, but run parallel to, the MACD and ACD busses so that error reports may propagate even if a bus is inoperative. In addition, the reporting paths may be duplicated to remove any single-point dependency in delivering an error report.

RECOVERY

The recovery process begins after an error report message has been broadcast around the system. Recovery is a distributed operation — each node in the system reads the error report message and decides what recovery actions need to be taken.

For recovery to be successful, there must be redundant resources available in the system. There are three redundancy mechanisms in the BIU and MCU: *bus retry buffers, ECC, and module shadowing*. The first two are useful in recovering from transient errors, while module shadowing allows recovery from permanent errors.

Figure 5 illustrates how every module in the system may be paired with another self-checking module of the same type. This pair of self-checking modules operates in lock step and provides a complete and current backup for all state information in the module. This mechanism is known as *module shadowing* because a shadow is ready to fill in if the primary fails, or vice versa. Fault detection and recovery is performed totally transparent to both application and system software. When the recovery is complete, system software is notified that a failure occurred. Figure 6 shows sample module failures and automatic hardware recovery.

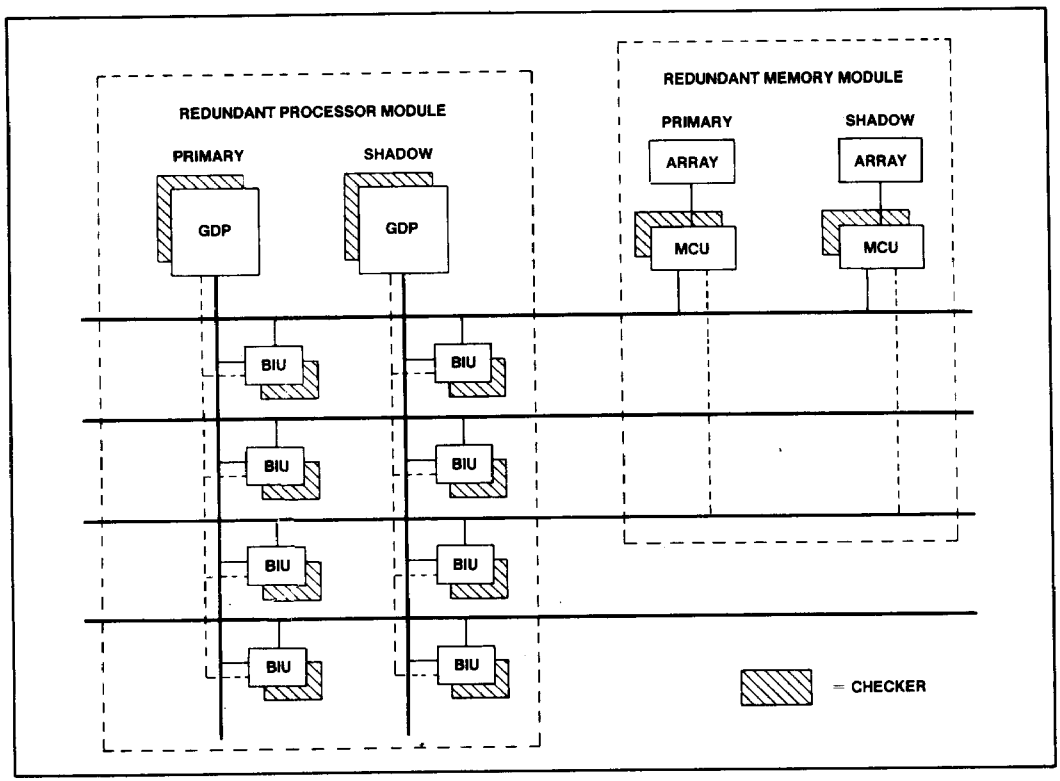


Figure 5. QMR Configuration Pairing

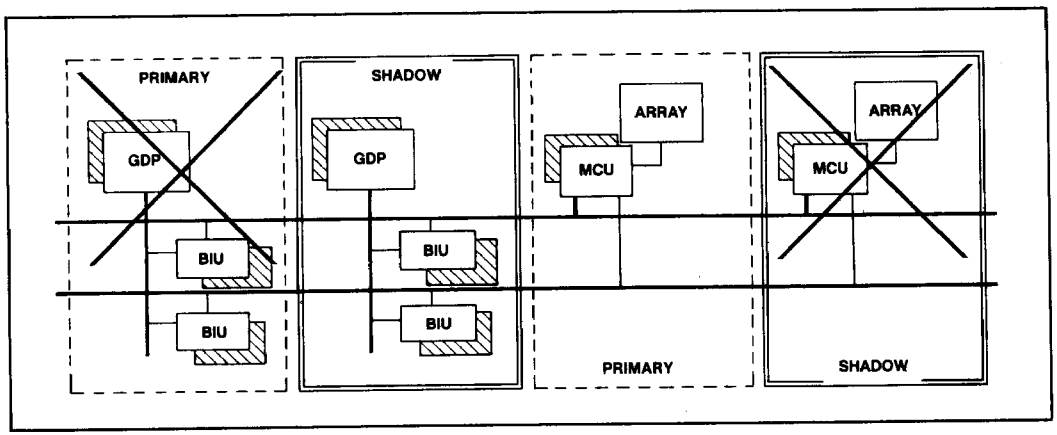


Figure 6A. Module Failures Are Detected

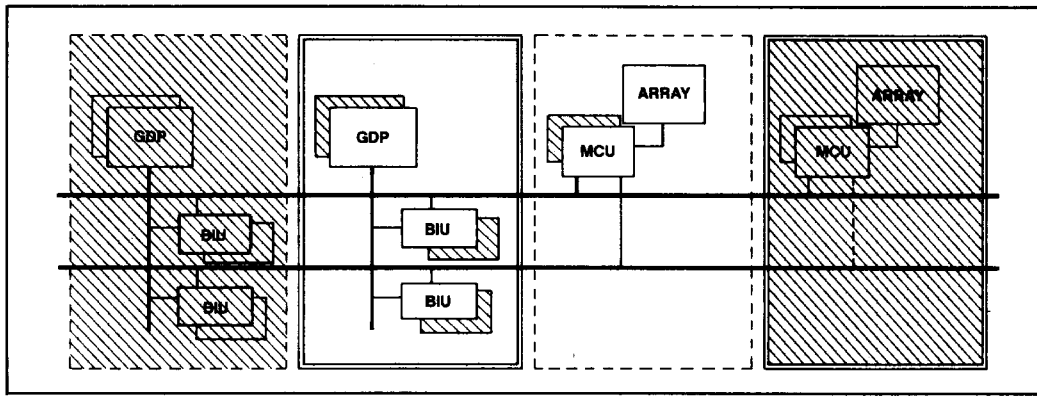


Figure 6B. Failed Modules Are Disabled

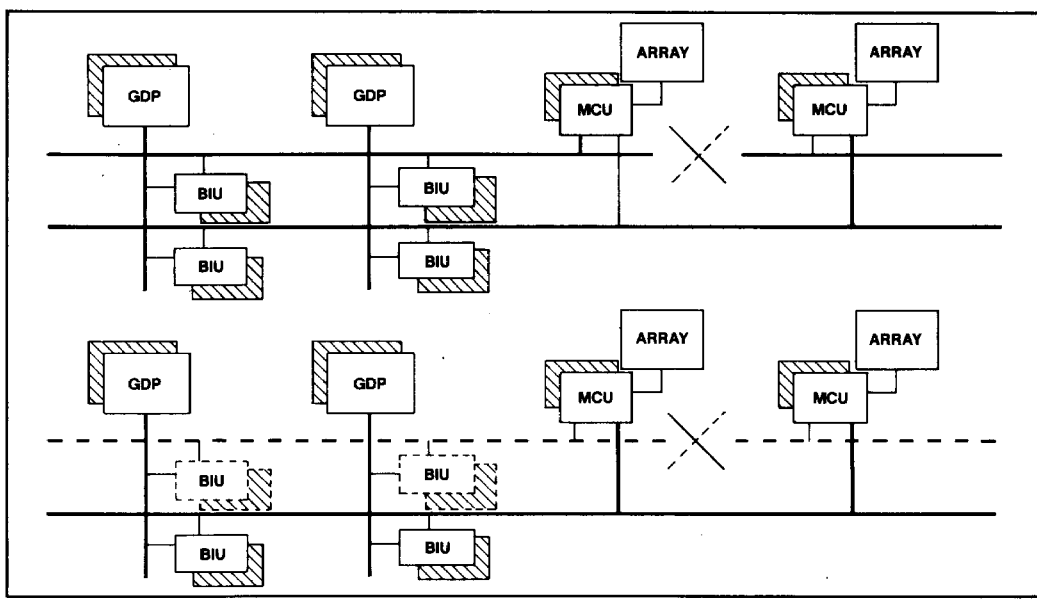


Figure 7. Bus Reconfiguration

A fault-tolerant module is also called a QMR module (Quad Modular Redundant) because most components (except memory) are replicated four times. There are two self-checking modules and each of these has a master and a checker.

Each memory bus in the system may be paired with another memory bus. Memory modules are physically connected to both buses although logically they are attached to only one bus at a time. During normal operation the buses run independently. Both contribute to the total memory bandwidth available in the system. If a bus fails, the memory modules attached to that bus will automatically switch to the other bus which is still operating. Figure 7 illustrates how the BIU and MCU reconfigure the system when a bus fails.

CONFIGURABLE FAULT TOLERANCE

Figure 8 illustrates the range of alternatives available to system designers when they build iAPX 432 systems. The most fault-tolerant systems are built from a QMR configuration of processors that can tolerate any single component failure without crashing the system. BIUs and MCUs provide full hardware error detection and recovery transparent to software.

The lowest cost configurations can be built using basic processor modules without FRC or QMR. This type of configuration will crash if a component fails, but can be made "self-healing" by adding intelligent software to the I/O subsystem. Unlike QMR, self-healing does not protect against system crashes, but it does allow the system to recover from a failure in a short period of time. The "healing" takes place in 3 steps. First, a watchdog timer in an I/O subsystem alerts I/O subsystem software that the central system has failed. Second, the I/O subsystem checks BIU/MCU error logging registers and runs diagnostics to identify which resource (e.g., processor, bus or memory) has failed. Third, the I/O subsystem reinitializes the system using the configuration control within the BIU and MCU to configure out the failed resource. The system is up and running without human intervention after only a short period of down time.

The basic configuration is the lowest cost alternative, but for some applications it is desirable to have a very high degree of confidence that calculations are performed correctly. A QMR system will do this since all components have a checker that alerts the system whenever a mistake is made. However, a QMR configuration may be overkill for some applications that can tolerate an occasional system failure, as long as the computations are correct when they do complete. FRC configurations offer an alternative in between the basic and QMR approaches. Adding a second set of checker components to each module improves the error detection capabilities of the system providing "high confidence" computing. No single hardware failure will go undetected and corrupt the results of a critical computation. FRC insures that any error is caught before it can propagate to another module in the system. FRC alone does not provide automatic hardware recovery like a QMR system, but it does detect errors as soon as they occur so that the system does not become corrupted. It is then the responsibility of system software to implement a "self-healing" strategy where the faulty resource is disabled and the system reinitialized.

The software configurability of a BIU/MCU system allows a system to use a combination of the above strategies. For example, software can configure a system as a full QMR system in the morning for critical applications, and then switch to an FRC only system in the afternoon. This doubles the system throughput (twice as many processors are working in parallel) without making any hardware changes.

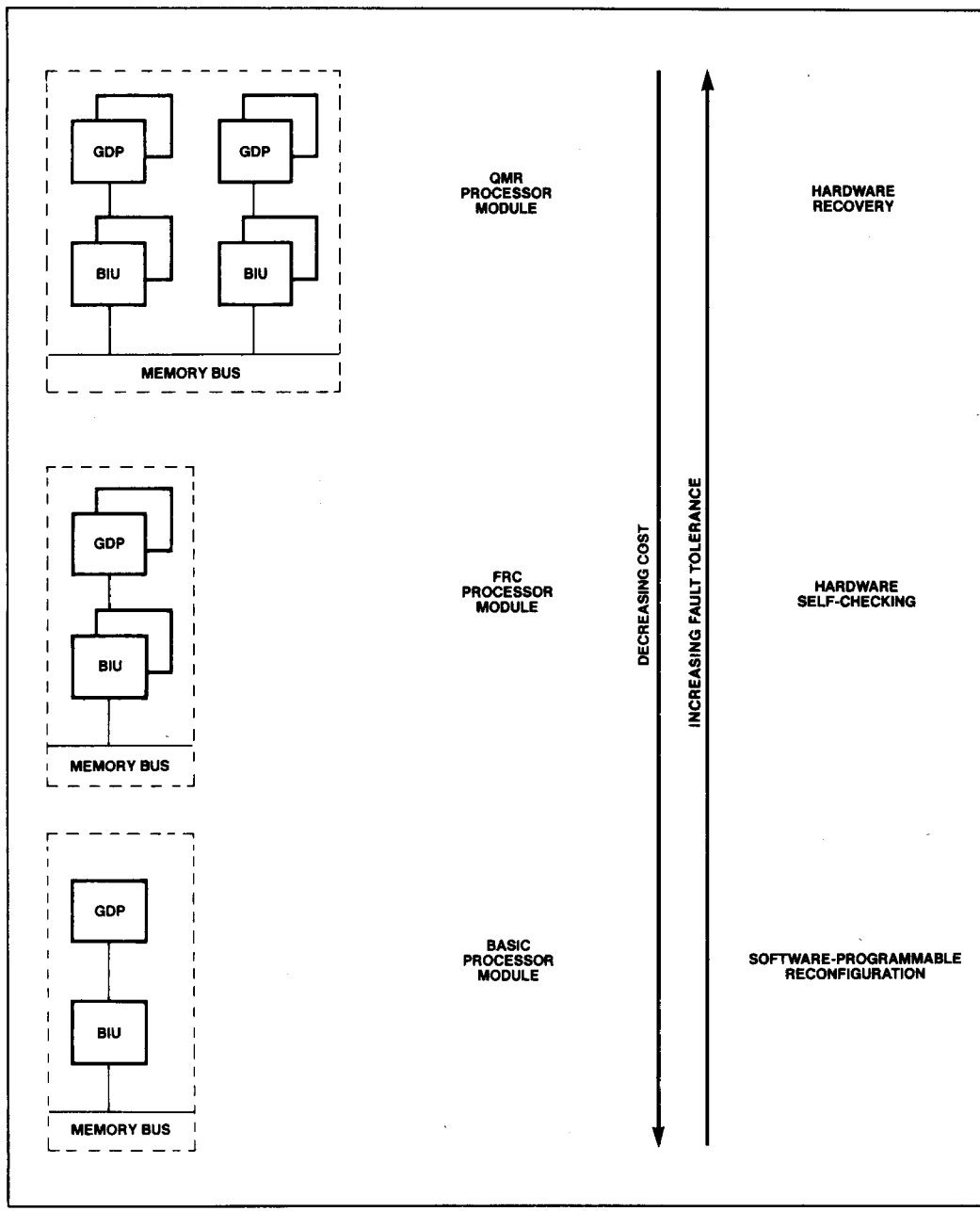


Figure 8. Fault-Tolerant Alternatives

FAULT-TOLERANT SYSTEM DESIGN RESPONSIBILITIES

The interconnect architecture and the VLSI components provide a stable base for developing fault-tolerant iAPX 432 systems. The iAPX 432 interconnect components address the issues concerning fault tolerance which are encountered when constructing the iAPX 432 central system. *A number of system-wide issues remain the responsibility of the iAPX 432 system designer.* These issues include:

- A fault-tolerant I/O system
- Fault-tolerant power supplies and distribution method
- A fault-tolerant method for clock generation and distribution
- The electrical and physical provisions for on-line repair

SUMMARY

The iAPX 432 interconnect architecture provides a standard VLSI method for constructing multiple processor VLSI computer systems. The iAPX 432 interconnect architecture is implemented by a pair of VLSI components, the Bus Interface Unit (BIU) and the Memory Control Unit (MCU). Together with iAPX 432 processors, these components permit the construction of modular, extensible, multiprocessor computer systems. The components are designed to support the construction of fully fault-tolerant iAPX 432 systems. However, there is no penalty in performance or in cost for those applications that do not require fault tolerance.

The 432 fault-tolerant mechanisms are designed to provide a flexible and complete solution to the problems of fault-tolerant hardware. For basic systems (those without checkers for error detection or QMR for recovery), a user may decide to use only a few detection mechanisms and provide recovery only for transient errors. This functionality comes at no additional cost in the VLSI interconnect system. To reduce maintenance cost and increase system availability, a system may use all of the detection mechanisms (i.e. may add checker components) but may not add any extra recovery capability (i.e. may not marry self-checking modules into a fault-tolerant QMR module). Continuous operation is available to the user who adds the extra recovery capabilities.

None of the fault-tolerant mechanisms reduce system performance. Systems that do not require the highest level of fault tolerance are not penalized in any way (cost, size, or performance) for the unused fault-tolerant capabilities. Increased levels of fault tolerance are achieved by replicating the iAPX 432 VLSI components. The hardware fault tolerance in the iAPX 432 is transparent to application software. The system's fault-tolerant capabilities may be changed without any changes to the application software system.

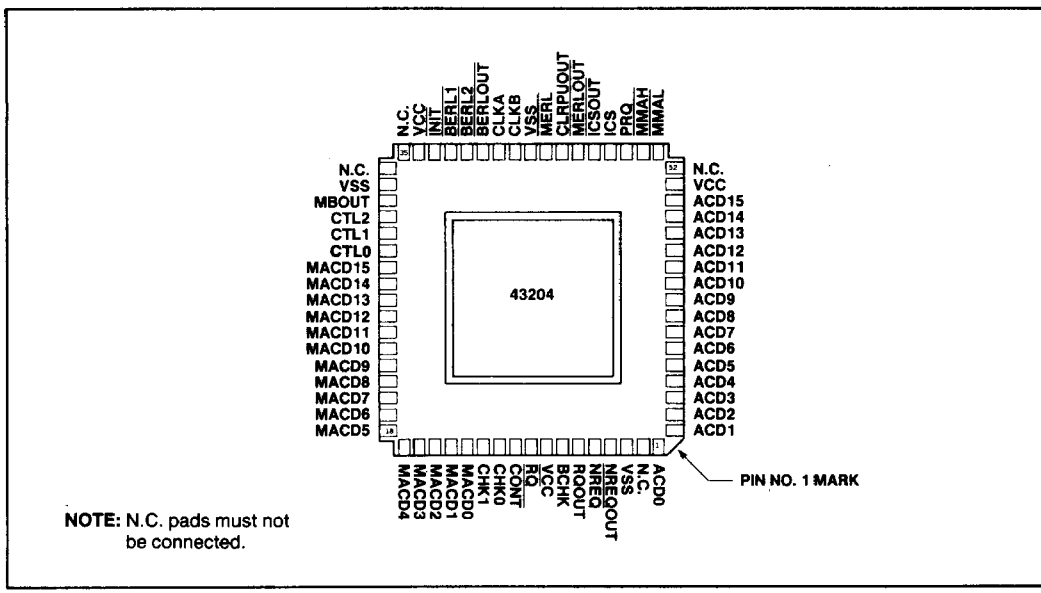


Figure 9. 43204 Pin Configuration

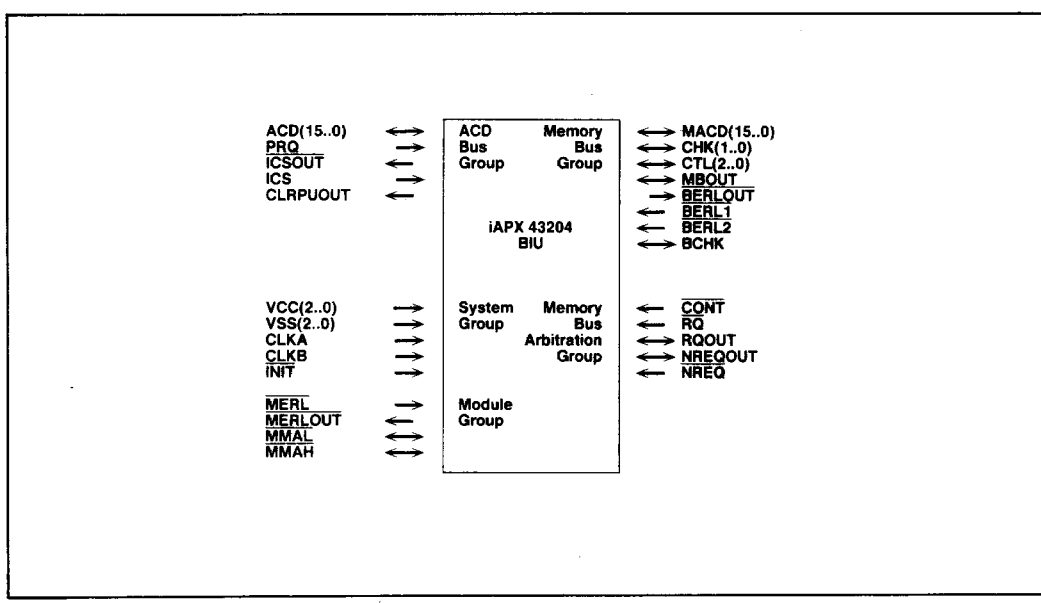


Figure 10. iAPX 43204 BIU Logic Symbol

Table 1. iAPX 43204 BIU Pin Description

Symbol	Type	Name and Function
Memory Bus Group		
MACD15 . . MACD0	I/O*	These 16 bidirectional signals carry physical memory addresses, control information (access length and type), and data to and from the memory bus.
CHK1..CHK0	I/O*	These 2 bidirectional signals carry parity bit check information which detects errors in transfers on MACD15 . . MACD0 and CTL2 . . CTL0. The 2 parity check bits are computed to satisfy the following equations (X = Exclusive OR): MACD15 X MACD13 X MACD11 X MACD9 X MACD7 X MACD5 X MACD3 X MACD1 X CTL1 X CHK1 = 0 MACD14 X MACD12 X MACD10 X MACD8 X MACD6 X MACD4 X MACD2 X MACD0 X CTL2 X CTL0 X CHK0 = 1 The BIU and MCU generate and check even parity (an even number of ones) across the 10 odd-numbered MACD, CTL, and CHK signals, and odd parity (an odd number of ones) across the 11 even-numbered MACD, CTL, and CHK signals.
CTL2..CTL0	I/O*	The 3 MACD bus control signals carry a code that controls the sequencing of the memory bus.
MBOUT	I/O**	MBOUT controls the direction of external buffers for the MACD, CHK, and CTL signals. When MBOUT is asserted, it indicates that the buffers must be directed to carry information outbound from the component to the memory bus.
$\overline{\text{BERLOUT}}$	0	$\overline{\text{BERLOUT}}$ supplies bit-serial bus error messages when the component detects a memory bus error, a storage array error, or a memory module error.
$\overline{\text{BERL1}}$, $\overline{\text{BERL2}}$,	I I	$\overline{\text{BERL1}}$ and $\overline{\text{BERL2}}$ are duplicate paths on which the component receives bit-serial bus error messages from the memory bus. When duplicated paths are not required, these two pins must be supplied with the same bus error report information.
BCHK	I/O*	BCHK provides a mechanism which checks that external buffers are operating. BCHK is toggled once each clock cycle by the component that is driving it. In an FRC pair, the master component drives BCHK. The checker component in the FRC pair receives BCHK. Routing BCHK from the master component, through one buffer in each external buffer package, and to the checker component, forms a serial network. If the oscillating BCHK signal fails to traverse the external buffer network, the buffer path is suspect and a bus error will be signalled. Buffer checking can be disabled by interconnect register programming.

Table 1. iAPX 43204 BIU Pin Description (Continued)

Symbol	Type	Name and Function
Memory Bus Arbitration Group		
$\overline{CONT}$	I	The $\overline{CONT}$ input indicates if the external arbitration network has detected that two or more simultaneous requests have been made for the use of the memory bus. When contention is indicated, all contending components will perform a binary arbitration sequence (based on each component's unique 6-bit module ID) to decide which component will be granted first use of the memory bus.
$\overline{RQ}$	I	The $\overline{RQ}$ input indicates if any agent is requesting the use of the memory bus. There are three valid combinations for $\overline{RQ}$ and $\overline{CONT}$: **$\overline{RQ}$ $\overline{CONT}$ Interpretation** 1 1 No request 0 1 One BIU is making a request 0 0 Two or more BIUs are making a request The MCU does not generate any memory bus requests. The MCU tracks the action of $\overline{RQ}$ and $\overline{CONT}$, and the CTL(2 . . 0) signals to determine when it is allowable to use the memory bus to reply to a request.
RQOUT	I/O**	When asserted, RQOUT signals that the component requires the use of the memory bus. RQOUT is intended to drive an external open-collector inverter which is wire-ORed to form a combined $\overline{RQ}$ line. The RQOUT signal from all BIUs attached to a memory bus must be logically combined to form a contention signal. (Contention occurs when two or more BIUs issue RQOUT simultaneously). The logic to detect contention among BIUs must be supplied by the customer.
NREQOUT	I/O*	The NREQOUT signal indicates that the component has received a new request from its associated processor. NREQOUT is intended to drive an external open-collector inverter, which is wire-ORed (with the same signal from other BIUs on the memory bus) to form $\overline{NREQ}$.
$\overline{NREQ}$	I	$\overline{NREQ}$ is an input that signals the beginning of a new time-ordered request cycle in which a request from one or more processors must be managed.

Table 1. iAPX 43204 BIU Pin Description (Continued)

Symbol	Type	Name and Function
Module Group		
$\overline{\text{MERL}}$	I	The $\overline{\text{MERL}}$ input accepts bit-serial module error messages. See the $\overline{\text{BERLOUT}}$ pin description for the format of the serial error messages.
$\overline{\text{MERLOUT}}$	O	The $\overline{\text{MERLOUT}}$ output broadcasts bit-serial module error messages to all BIUs contained within the same module (attached to the same processor).
$\overline{\text{MMAL}}$	I/O+	$\overline{\text{MMAL}}$ operates in the same manner as $\overline{\text{MMAH}}$ except that when $\overline{\text{MMAL}}$ is asserted it indicates that the lower addressed portion of a multiple module access is in progress on the memory bus. The two BIUs that are cooperating in a multiple module access observe both $\overline{\text{MMAH}}$ and $\overline{\text{MMAL}}$. Both signals are deasserted after each BIU has completed its portion of the access on the memory bus to which it is connected. In read accesses, after both signals are deasserted, the BIU with the lower addressed portion of the access presents data to the processor first. The BIU with the higher addressed portion of the access tracks the other BIU by counting the number of bytes returned to the processor (noting ICS, Interconnect Status, see below). When the BIU with the lower-addressed portion of the access completes its transfer, the next BIU begins automatically. Multiple module read accesses which begin at an odd addressed byte boundary cause the two cooperating BIUs to simultaneously return data to the processor. At the address boundary for which they share access responsibility, the low BIU returns its last byte on ACD7 . . ACD0, and the high BIU returns its first byte on ACD15 . . ACD8. After $\overline{\text{MMAH}}$ and $\overline{\text{MMAL}}$ have been deasserted, one (or both) of the BIUs may reassert the signals, each to indicate that its portion of the multiple module access encountered an error. This indication will be returned to the processor during error significance time for ICS.
$\overline{\text{MMAH}}$	I/O+	When asserted, $\overline{\text{MMAH}}$ indicates that one of the BIUs in a module is performing the high order address part of a multiple module access. A multiple module access occurs when a processor request spans an address range such that two memory buses, each connected via a different BIU must be engaged. When it is deasserted, $\overline{\text{MMAH}}$ indicates that the high portion of the access has been completed on the memory bus.

Table 1. iAPX 43204 BIU Pin Description (Continued)

Symbol	Type	Name and Function
ACD Bus Group		The ACD Bus Group contains the set of signals with which a compatible iAPX 432 processor connects to the BIU. See the iAPX 43201/43202 General Data Processor Data Sheet (Order Number 590125) and the iAPX 43203 Interface Processor Data Sheet (Order Number 590130) for information about compatible iAPX 432 processors.
ACD15 . . ACD0	I/O	These 16 signals form the processor-to-BIU communication path that carries all memory and interconnect accesses.
PRQ	I	PRQ indicates the start of a processor request to the BIU.
$\overline{\text{ICSOUT}}$	O	$\overline{\text{ICSOUT}}$ is intended to drive an external open-collector inverter to form ICS. All BIUs in a processor module contribute to the wired-OR ICS signal.
ICS	I	ICS supplies interconnect status to both the BIU and its associated iAPX 432 processor. ICS carries information on errors, data synchronization, and interprocessor communication to the processor. It is also monitored by each BIU for coordinating multiple module accesses.
CLRPUOUT	O	CLRPUOUT is intended to drive an open collector inverter and form a wired-OR $\overline{\text{CLR}}$ signal to the associated iAPX 432 processor. All BIUs in a processor module contribute to the wired-OR $\overline{\text{CLR}}$ signal. Using CLRPUOUT, a BIU can synchronize the FRC master and checker processor components.
System Group		
VCC2..VCC0		Three VCC pins supply 5-volt power to the BIU/MCU. All three pins must be connected. The three VCC pins are not connected together inside the component.
VSS2..VSS0		Three VSS pins provide ground to the BIU/MCU. All three pins must be connected. The three VSS pins are not connected together inside the component.
CLKA	I	CLKA is a square-wave clock for the BIU/MCU. CLKA must operate continuously to preserve the operating state of the component.
CLKB	I	CLKB is a square-wave clock for the BIU/MCU. CLKB is the same frequency as CLKA but lags CLKA by 90 degrees. CLKB must operate continuously to preserve the operating state of the component.
$\overline{\text{INIT}}$	I	$\overline{\text{INIT}}$ is a signal that causes the BIU/MCU to initialize. In addition, $\overline{\text{INIT}}$ is used to enable external logic which provides configuration information to the component.

Legend:
I = Input signal
O = Output signal
I/O = Input/Output signal
* = FRC errors cause module error
** = FRC errors cause bus error
$\underline{+}$ = External passive pullup required (10K Ohms)
$\overline{}$ = Asserted low

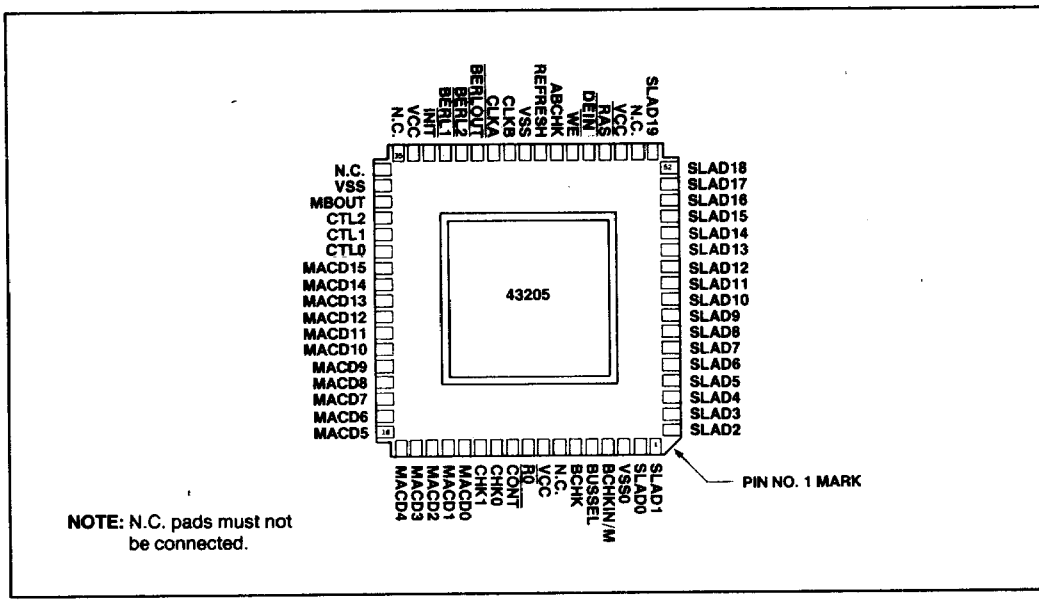


Figure 11. 43205 Pin Configuration

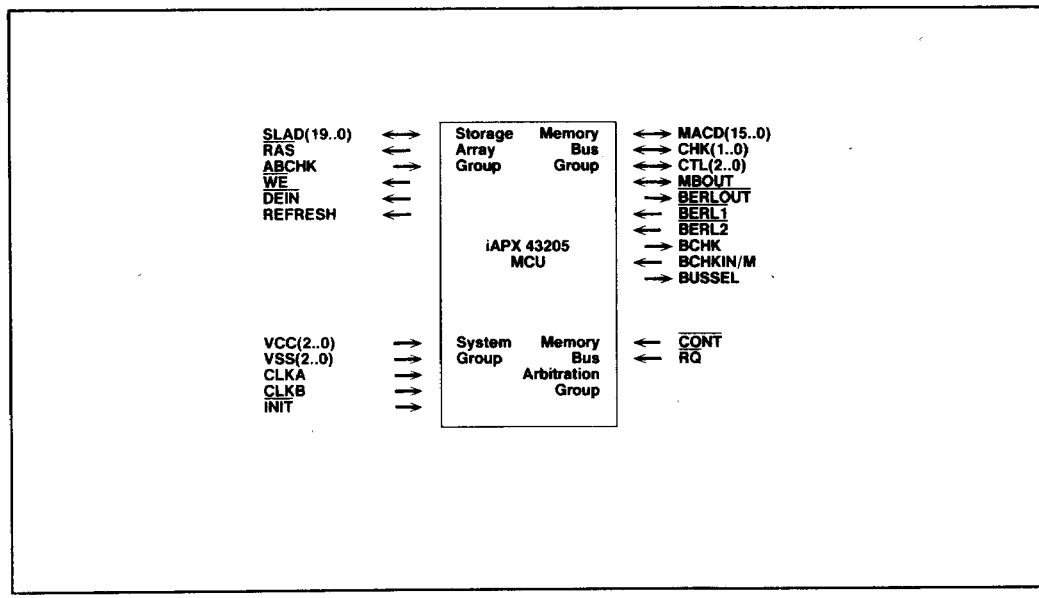


Figure 12. iAPX 43205 MCU Logic Symbol

Table 2. iAPX 43205 MCU Pin Description

Symbol	Type	Name and Function
Memory Bus Group		
MACD15 . . MACD0	I/O*	These 16 bidirectional signals carry physical memory addresses, control information (access length and type), and data to and from the memory bus.
CHK1..CHK0	I/O*	These 2 bidirectional signals carry parity bit check information which detects errors in transfers on MACD15 . . MACD0 and CTL2 . . CTL0. The 2 parity check bits are computed to satisfy the following equations (X = Exclusive OR): MACD15 X MACD13 X MACD11 X MACD9 X MACD7 X MACD5 X MACD3 X MACD1 X CTL1 X CHK1 = 0 MACD14 X MACD12 X MACD10 X MACD8 X MACD6 X MACD4 X MACD2 X MACD0 X CTL2 X CTL0 X CHK0 = 1 The BIU and MCU generate and check even parity (an even number of ones) across the 10 odd-numbered MACD, CTL, and CHK signals, and odd parity (an odd number of ones) across the 11 even-numbered MACD, CTL, and CHK signals.
CTL2..CTL0	I/O*	The 3 MACD bus control signals carry a code that controls the sequencing of the memory bus.
MBOUT	I/O**	MBOUT controls the direction of external buffers for the MACD, CHK, and CTL signals. When MBOUT is asserted, it indicates that the buffers must be directed to carry information outbound from the component to the memory bus.
$\overline{\text{BERLOUT}}$	O	$\overline{\text{BERLOUT}}$ supplies bit-serial bus error messages when the component detects a memory bus error, a storage array error, or a memory module error.
$\overline{\text{BERL1}}$ $\overline{\text{BERL2}}$	I I	$\overline{\text{BERL1}}$ and $\overline{\text{BERL2}}$ are duplicate paths on which the component receives bit-serial bus error messages from the memory bus. When duplicated paths are not required, these two pins must be supplied with the same bus error report information.
BCHK	O*	BCHK provides a mechanism which checks that external buffers are operating. BCHK is toggled once each clock cycle by the component. By routing BCHK through one buffer in each external buffer package to BCHKIN/M, a serial network is formed. If the oscillating BCHK signal fails to traverse the external buffer network, BCHKIN/M will detect the error and signal a bus error. The BCHK signal does not toggle when the component is being initialized by either the $\overline{\text{INIT}}$ signal or an internal initialization request. Buffer checking can be disabled by a parameter acquired by the MCU during initialization. The MCU will disable buffer checking after it detects a permanent module error.
BCHKIN/M	I	BCHKIN/M checks the oscillating BCHK signal after it has been routed through each of the external buffers for the memory bus. If any errors are detected, a module error will be signalled. During initialization, the BCHKIN/M pin accepts the MASTER information. If it is high during initialization, then the component will become the master of an FRC pair of components; otherwise it will become a checker.

Table 2. iAPX 43205 MCU Pin Description (Continued)

Symbol	Type	Name and Function
Memory Bus Group		
BUSSEL	O	BUSSEL controls which of two memory buses (normal or backup) the MCU is to use. The middle bit of an internal 3-bit normal bus identifier (ID) is logically combined with an internal bus state code to produce BUSSEL. The bus state code records the state and health of both the normal and backup buses. When the component switches to an alternate bus (changes the bus state code), BUSSEL is changed accordingly.
Memory Bus Arbitration Group		
$\overline{CONT}$	I	The $\overline{CONT}$ input indicates if the external arbitration network has detected that two or more simultaneous requests have been made for the use of the memory bus. When contention is indicated, all contending components will perform a binary arbitration sequence (based on each component's unique 6-bit module ID) to decide which component will be granted first use of the memory bus.
$\overline{RQ}$	I	The $\overline{RQ}$ input indicates if any agent is requesting the use of the memory bus. There are three valid combinations for $\overline{RQ}$ and $\overline{CONT}$: **$\overline{RQ}$ $\overline{CONT}$ Interpretation** 1 1 No request 0 1 One BIU is making a request 0 0 Two or more BIUs are making a request The MCU does not generate any memory bus requests. The MCU tracks the action of $\overline{RQ}$ and $\overline{CONT}$, and the CTL(2 . . 0) signals to determine when it is allowable to use the memory bus to reply to a request.
Storage Array Group		
SLAD19. . SLAD0	I/O*	The 20 SLAD signals form the communication path between the MCU and its associated storage array. The SLAD bus multiplexes addresses to the storage array with data (32 bits) and ECC (7 bits) which are to be read from or written to the array.
$\overline{RAS}$	O*	When $\overline{RAS}$ is asserted, it indicates the start of a storage array cycle. $\overline{RAS}$ may combine with external sequencing logic to control the operation of the storage array.
ABCHK	I	ABCHK is an input used to verify the external RAM control logic. WE and CAS are used to generate the ABCHK signal. In addition, the functionality of the external buffers associated with the storage array may be validated by routing the oscillating BCHK signal through each of the buffers in a similar manner as on the MACD bus side of the MCU. If an error is detected, ABCHK can be corrupted and in this fashion report the error. An alternate method of checking the storage array buffers is to use buffer packages with no more than four buffers per package so that the special ECC protection in the MCU may detect buffer failures.
$\overline{WE}$	O*	When $\overline{WE}$ is asserted, the MCU indicates that a write operation is to be performed in the storage array.

Table 2. iAPX 43205 MCU Pin Description (Continued)

Symbol	Type	Name and Function
Memory Bus Arbitration Group		
$\overline{\text{DEIN}}$	O*	When $\overline{\text{DEIN}}$ is asserted, the MCU indicates that the SLAD19 . . SLAD0 signals are ready to accept information from the storage array into the MCU.
REFRESH	O*	When REFRESH is asserted the MCU indicates that the storage array cycle is a refresh cycle. In systems with multiple bank dynamic RAM storage arrays, the REFRESH signal may be used to command the storage array sequencing logic to perform an appropriate cycle (e.g., RAS-only refresh for all banks). In a storage array with a single bank of dynamic RAMs the REFRESH signal need not be used.
System Group		
VCC2..VCC0		Three VCC pins supply 5-volt power to the BIU/MCU. All three pins must be connected. The three VCC pins are not connected together inside the component.
VSS2..VSS0		Three VSS pins provide ground to the BIU/MCU. All three pins must be connected. The three VSS pins are not connected together inside the component.
CLKA	I	CLKA is a square-wave clock for the BIU/MCU. CLKA must operate continuously to preserve the operating state of the component.
CLKB	I	CLKB is a square-wave clock for the BIU/MCU. CLKB is the same frequency as CLKA but lags CLKA by 90 degrees. CLKB must operate continuously to preserve the operating state of the component.
$\overline{\text{INIT}}$	I	$\overline{\text{INIT}}$ is a signal that causes the BIU/MCU to initialize. In addition, $\overline{\text{INIT}}$ is used to enable external logic which provides configuration information to the component.

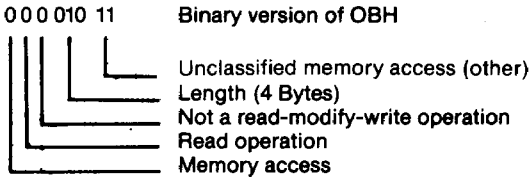
Legend:
I = Input signal
O = Output signal
I/O = Input/Output signal
* = FRC errors cause module error
** = FRC errors cause bus error
$\overline{}$ = Asserted low

iAPX 43204 FUNCTIONAL DESCRIPTION

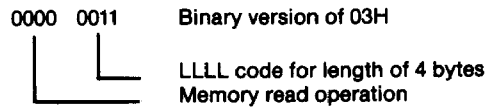
This section describes how the iAPX 43204 Bus Interface Unit operates through a set of functional diagrams that trace the operation of the pins clock-by-clock. To understand the notation on the various waveforms, refer to Figure 13. It illustrates the general operation of the BIU as it accepts a memory read request from a processor, forwards the request to a Memory Control Unit (MCU), and returns the reply data to the processor.

The cycle numbers (0, 1, 2, . . .) at the top of each diagram enumerate the clock cycles. When a common group of signals is plotted, its value is displayed inside a data waveform. For example, the BIU signals MACD15 . . . MACDO are all high in cycles 0 through 4. In cycle 5, the MACD bus carries the value 0300H and in cycle 6 the value 0000H.

Notice that in Figure 13, the BIU receives a processor request (PRQ) in cycle 0. The ACD bus carries the information 0B00H in cycle 1 (the specification field on the high-byte and addresses A7 . . . A0 on the low-order byte) and 0000H in cycle 2 (the address bits A23 . . . A8) for the access. Reorganizing the information, it is apparent that the BIU has been provided a processor request with a specification field of OBH and a 24-bit physical address of 000000H. Referring to the GDP or IP data sheets, the specification field can be decoded as follows:



Once the processor has presented the request to the BIU, the BIU presents ICS (Interconnect Status) to the processor indicating that the processor is to wait for the data to be returned by the BIU and the memory system. In cycle 4, the BIU issues the NREQOUT signal (not shown) and observes NREQ and RQ immediately. Though not shown, no contention with any other processor is observed on the CONT pin. In cycle 5, the BIU asserts MBOUT to control its external TTL buffers to drive the memory bus and presents a two-cycle memory read request message. The MACD bus carries the memory bus specification code (high-order byte) and the physical memory address bits (A7 . . . A0) in cycle 5, and the remaining 16 address bits (A23 . . . A8) in cycle 6. Referring to the Memory Bus appendix of the Interconnect ARM, the memory bus specification field of 03H is decoded as follows:



In cycles 10 and 11, the MCU that serviced the request presents the data that it read from the storage array, least significant bytes first. The returned bytes in the example are: 2AH, 03H, 9AH, 8CH. In cycles 15 and 16, the BIU presents the same data to the processor and indicates the availability of each byte with ICS.

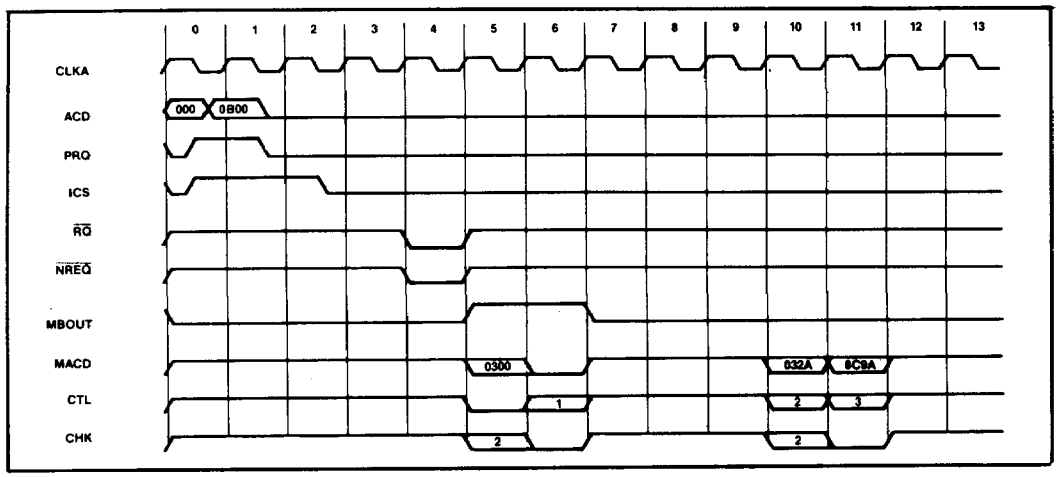


Figure 13. 4-Byte Memory Read

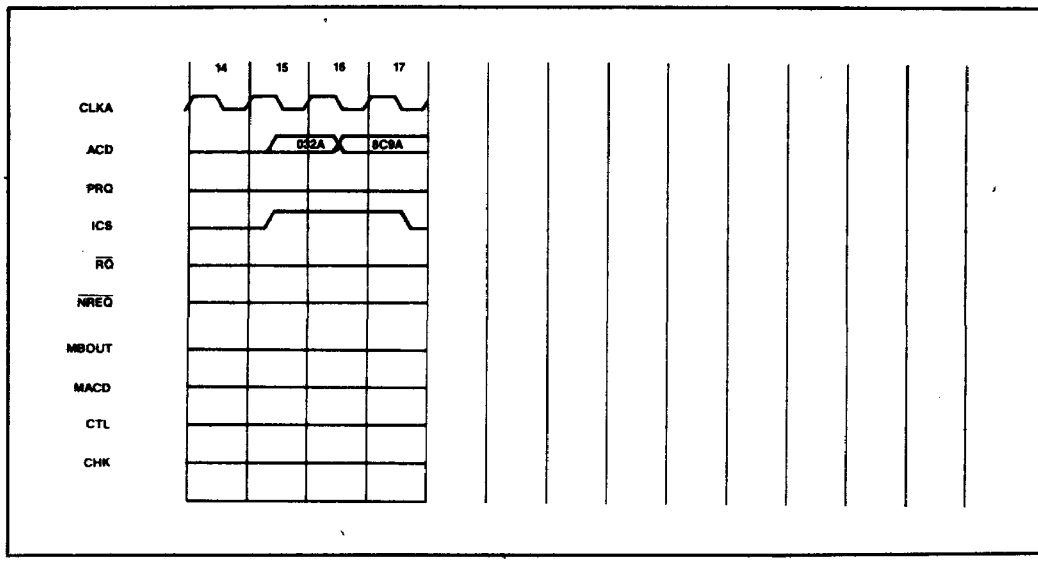


Figure 13. 4-Byte Memory Read (continued)

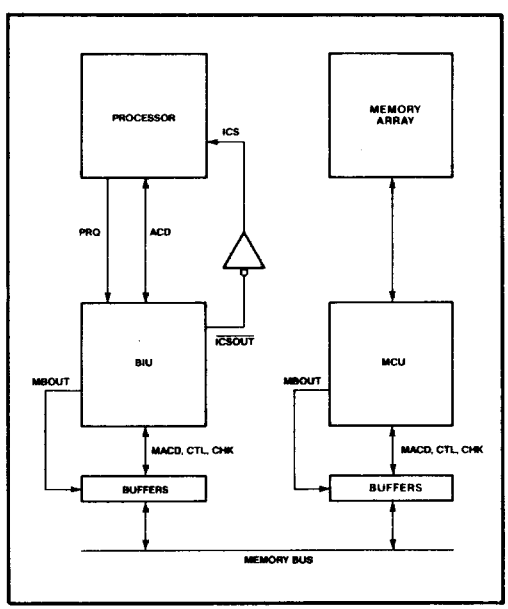


Figure 14. Hardware Configuration for Memory Read/Write Examples

Figure 15 illustrates the steps that occur when a processor requests a 4-byte write at physical memory address 0. The hardware system for this example includes one processor, one BIU, one memory bus, one MCU, and one storage array (see Figure 15). In cycle 0, the processor emits the processor request signal (PRQ=1) along with the specification field and the low-order address byte on the ACD signals. The specification field (high-byte of the first double-byte) in this example is 4BH and the low-order address byte (low-byte of the first double-byte) is 00H. In cycle 1, the processor emits the high- and mid-order address bytes on the ACD signals. In this example, the high- and mid-order address ;bytes are 00H. The BIU deasserts ICSOUT (not shown) which, after an external logical inversion, provides the ICS signal (Interconnect Status) to the processor. The deasserted ICS is used to stretch the processor access cycle (generate wait states). The processor presents the first double-byte of data to be written in cycle 4, but because ICS is deasserted, the processor must stretch the data into cycle 5 before the BIU will accept it. In cycle 5, the processor provides the second double byte of data to be wirtten (0000H). Notice that the BIU deasserts ICS to hold the processor until the entire request is actually satisfied by the MCU and memory array.

In cycle 4, the BIU presents the request to the memory bus arbitration logic. Since only one processor is in the system, the NREQ (output) and RQ input signals are asserted but no contention (CONT signal is not shown) is detected. Thus, the write access proceeds immediately to the memory bus (MACD, CTL, and CHK signals). In cycles 5 through 8, MBOUT is asserted by the BIU to direct external buffers to pass the BIU request to the memory bus. The first two double-bytes of the request contain a specification byte and three address bytes. The information in these two double-bytes is a modified version of that received from the processor ACD bus. The high-order byte of the memory address is contained in the low-order byte of the first double-byte. The mid- and low-order bytes of the memory address are contained, respectively, in the high-order and low-order bytes of the second double byte. MBOUT remains asserted while the two double-bytes of the write request are provided by the BIU. The MCU performs the access and returns a write reply (CTL=6) in cycle 10. The BIU asserts ICS for the processor in cycle 14, allowing the processor to escape from the stretched write cycle that it had entered earlier. In this case, no error occurred during the access. However, had there been an error, the level of ICS in cycle 15 would indicate the error to the processor.

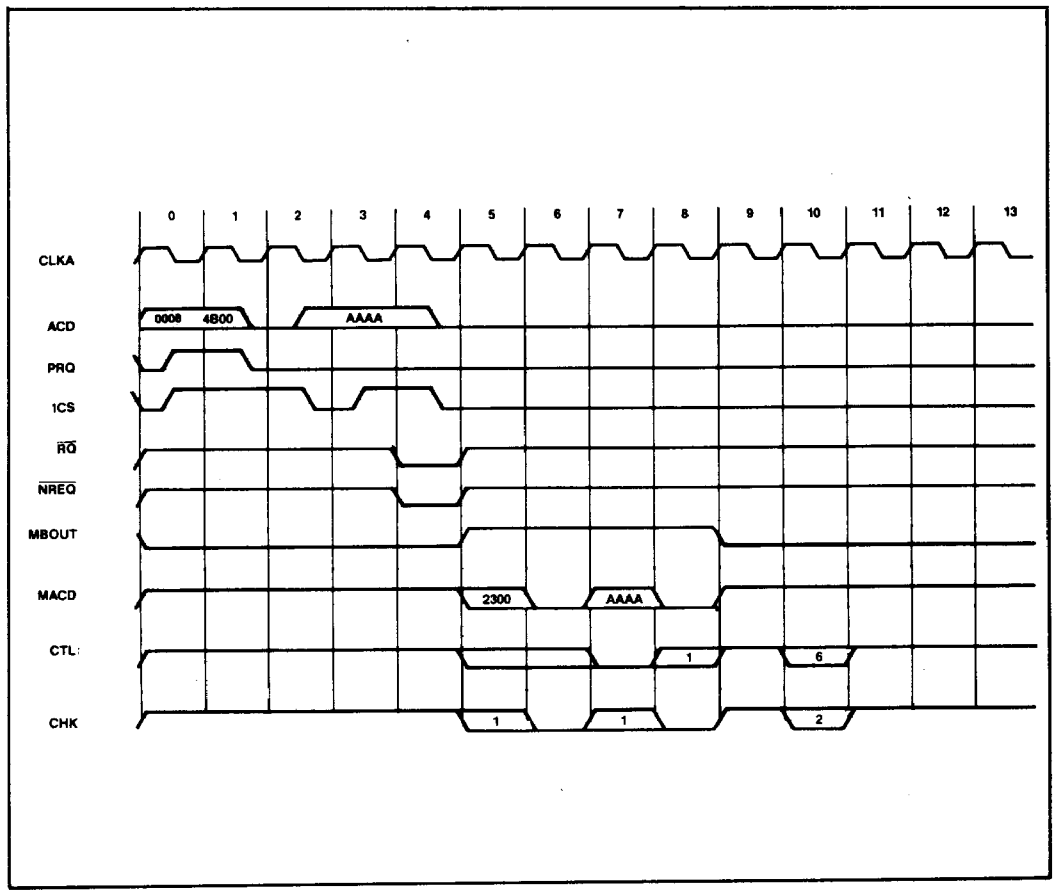


Figure 15. 4-Byte Memory Write

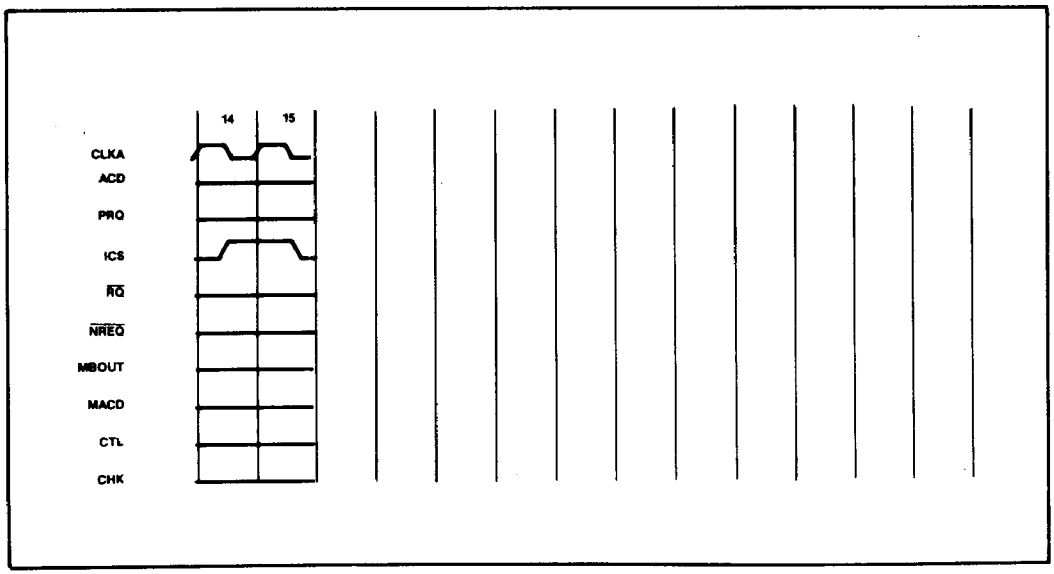


Figure 15. 4-Byte Memory Write (continued)

MULTIPLE MODULE ACCESS READ— 2-WAY INTERLEAVING

The Figures 16 and 17 illustrate the interconnect system's ability to support interleaving of memory requests among two memory busses. The hardware configuration pictured in Figure 16 contains one processor, two BIUs, two memory busses, two MCUs, and two storage arrays. The two BIUs are initialized to perform 2-way interleaving of memory requests based on bit 6 of the physical address that the processor provides. Specifically, that portion of a request with bit 6 of the physical address equal to zero is serviced by memory bus 0. That portion of a request with bit 6 of the physical address equal to one is serviced by bus 1.

In the example that follows, an 8-byte read at physical memory address 00003DH is requested. As Figure 16 indicates, the two BIUs each recognize the portion of the request that is supported by the bus to which they are connected. The BIU on bus 0 (BIU0) services the first 3 bytes of the access (03DH . . . 03FH) and the BIU on bus 1 (BIU1) services the remaining 5 bytes (040H . . . 044H). In the interleaving process, the BIUs reorder the physical address bits, based on the selected interleaving, to present requests to the MCU in a linear address space. A BIU transforms its portion of the request by replacing address bit 23 of the physical address with address bit 6, and shifting original address bits 23 . . . 7 to the right by one position. The reordered result (6,23 . . . 7,5 . . . 0) is forwarded to the respective memory bus. The BIU1 translation of physical address 000040H to memory bus 2 address 800000H illustrates the address interleaving operation.

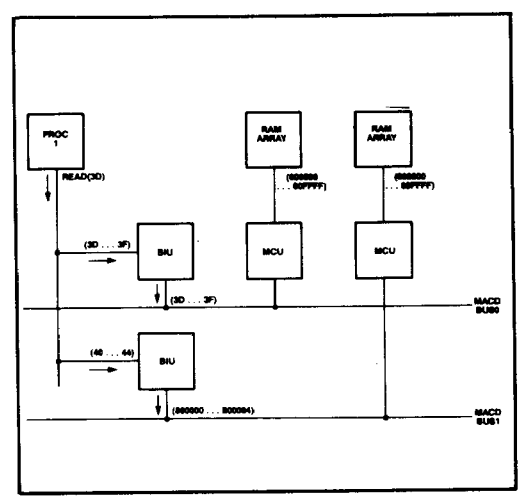


Figure 16. Hardware Configuration for MMA Read

Referring to Figure 17 for this operation, notice that the processor emits its request just as in earlier examples. However, in this case, two BIUs recognize that they each must participate in the request. Thus, BIU0 asserts MMAL and BIU1 asserts MMAH indicating that a multiple module access is required to complete this access. The two BIUs then present their requests to their respective memory busses after performing the address interleaving as described above and each access continues independently. Once both accesses have completed, the BIUs cooperate to return the data to the processor. BIU0 provides the first 3 bytes and BIU1 returns the final 5 bytes. Notice, in cycle 23, that BIU0 provides the low byte (25H) and BIU1 provides the high-byte (F5H). Thereafter, BIU1 provides its remaining bytes with the byte significance that is required by the processor. By this cooperation, the processor is unaware that two BIUs were involved in the operation.

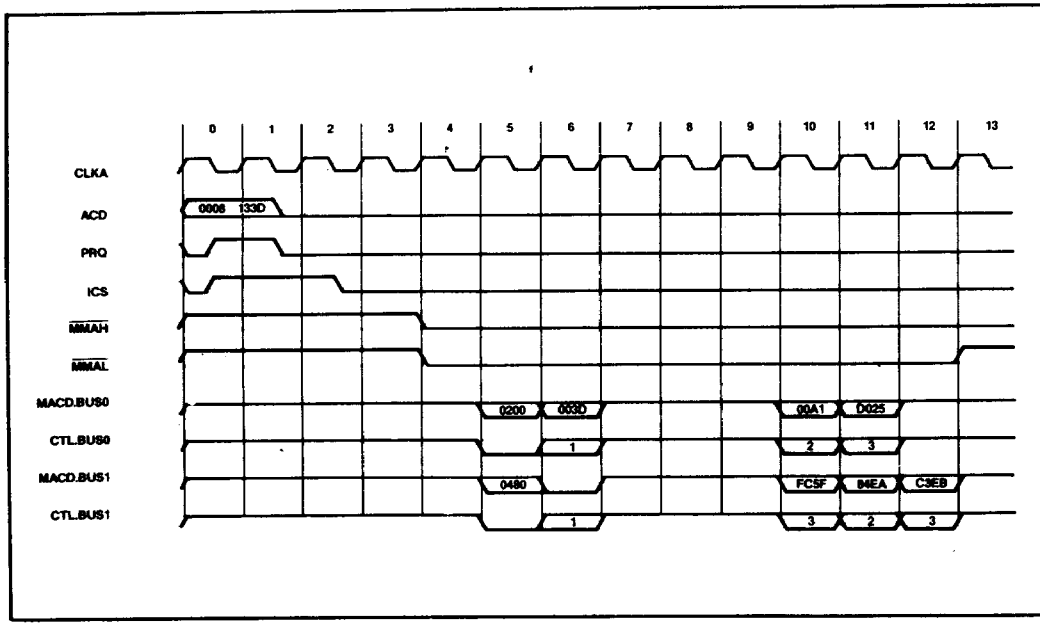


Figure 17. Multiple Module Access Read—2-Way Interleaving

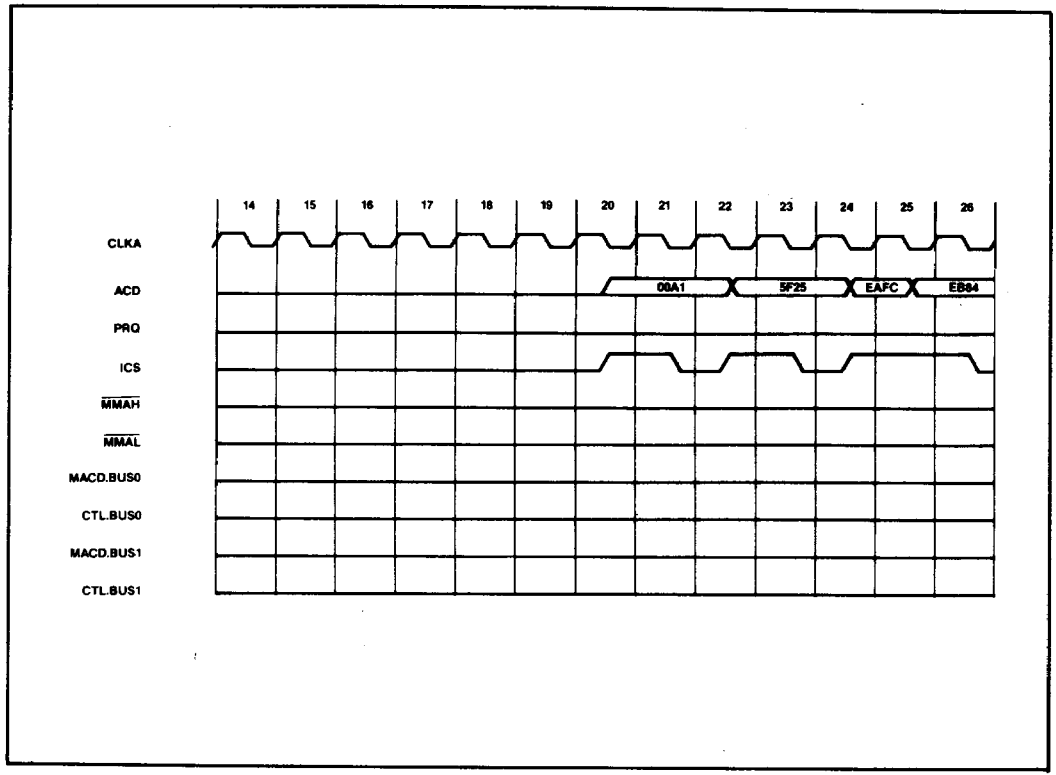


Figure 17. Multiple Module Access Read—2-Way Interleaving (cont)

MULTIPLE MODULE ACCESS WRITE—2-WAY INTERLEAVING

The following multiple module access is also performed on the same hardware configuration shown in Figure 16 with modified interleaving characteristics. In this case, the system is initialized with 2-way interleaving on address bit 7 (see Figure 19). Therefore, the 8-byte multiple module access write to physical address 00007DH is transformed into two memory bus requests: bytes 000040H . . . 000044H on bus 0 and bytes 80003DH . . . 80003FH on bus 1. Again, the $\overline{\text{MMAH}}$ and $\overline{\text{MMAL}}$ signals coordinate the multiple module access and each bus operates independently (see Figure 20).

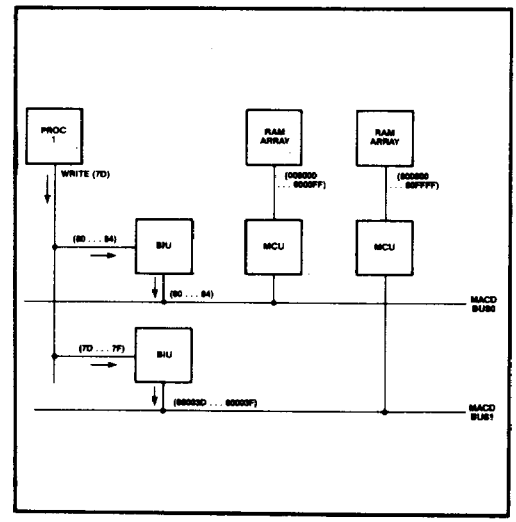


Figure 18. Hardware Configuration for MMA Write

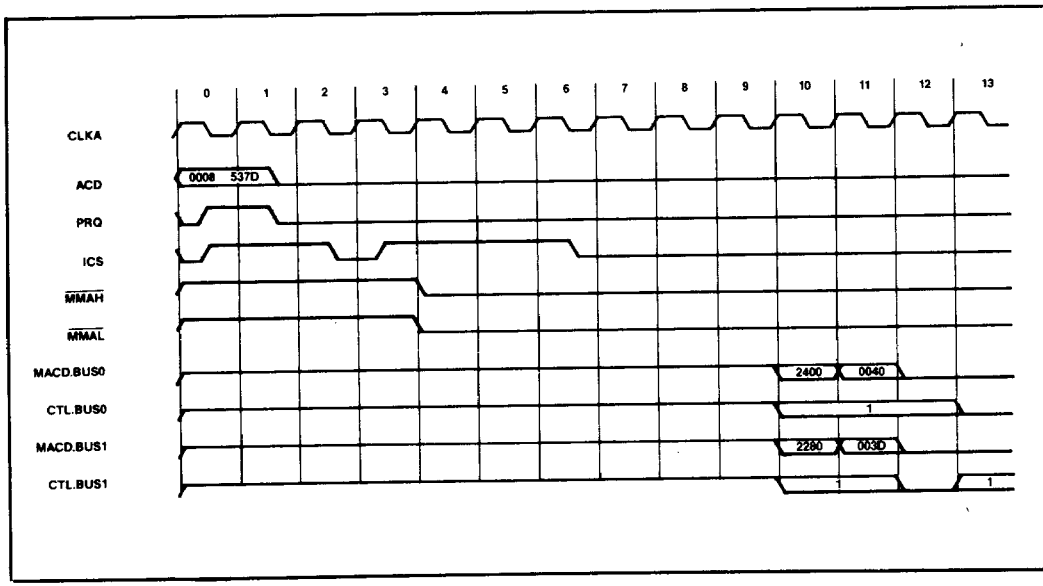


Figure 19. Multiple Module Access Write—2-Way Interleaving

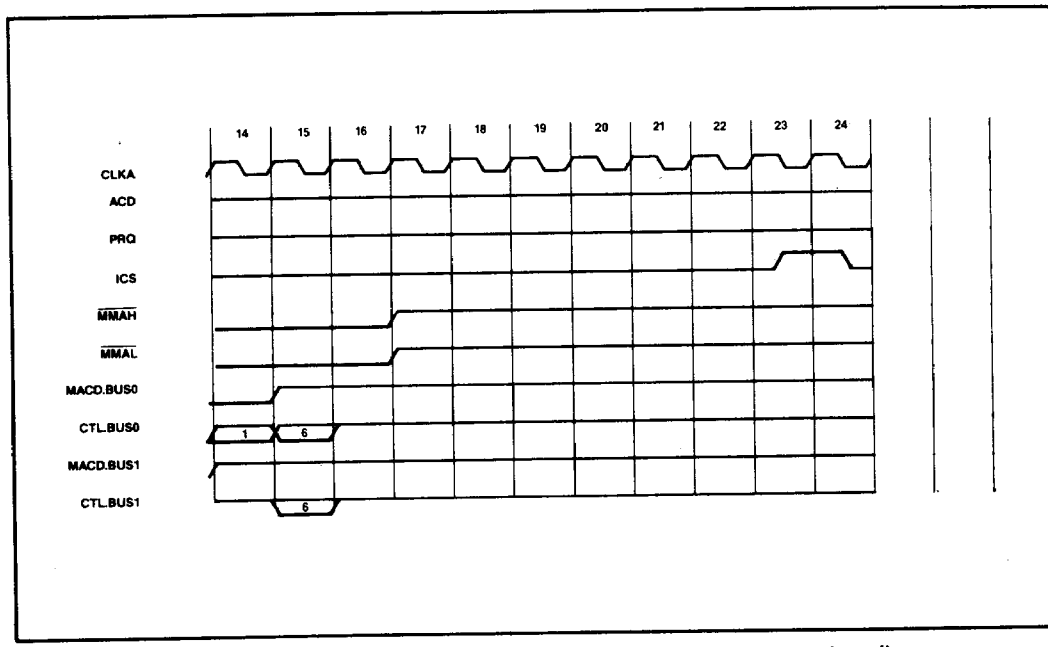


Figure 19. Multiple Module Access Write—2-Way Interleaving (Continued)

GLOBAL INTERPROCESSOR COMMUNICATION (IPC) BUS BLURB

Figure 20 illustrates the signaling of a global interprocessor communication message (IPC). The IPC is called a bus blurb since it is broadcast to all BIUs on the memory bus but does not require that any replies be generated. The delivery of the message is guaranteed by a memory-based communication object; the bus blurb only serves as the low-level notification. A programmer invokes a global IPC by the instruction BROADCAST TO PROCESSORS. The processor performs the instruction by writing a value (00H) to the IPC interconnect register (address 02H) in the associated BIU. In cycle 0, the processor emits a specification field (C7H). In cycles 0 and 1, the processor provides the interconnect address (000002H). In cycle 2, the processor provides the destination processor ID (0000H represents all processors, the global ID). Notice that the BIU stretches the processor (ICS=0) until the delivery of the global IPC is completed (cycle 10).

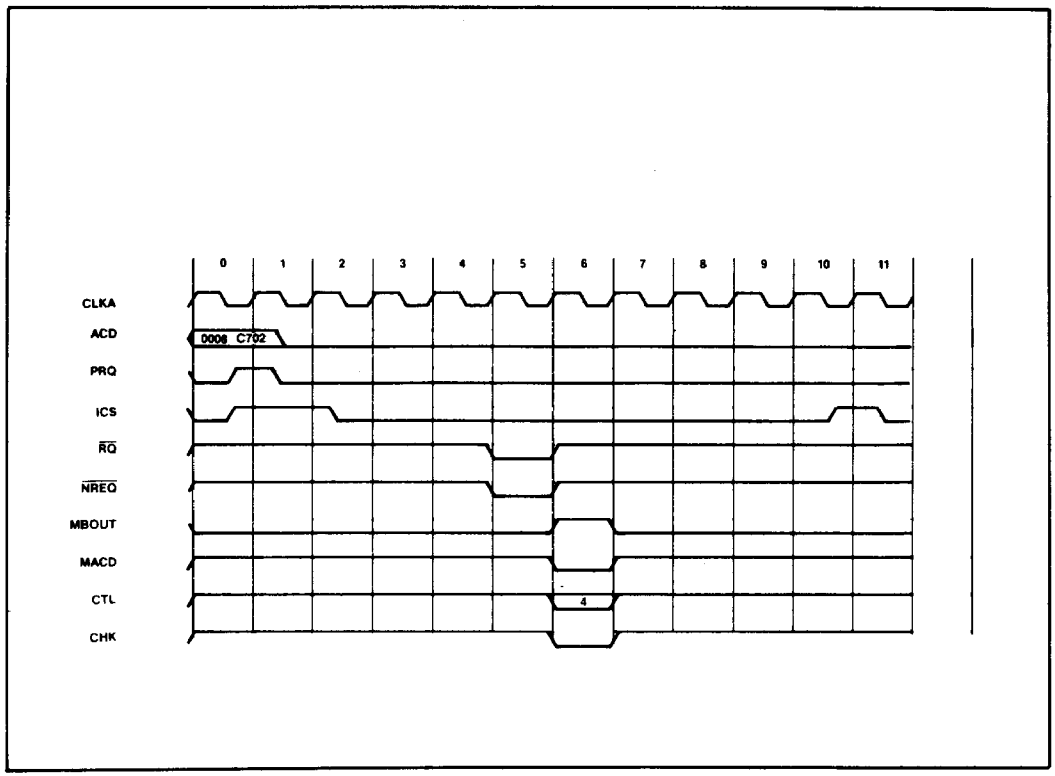


Figure 20. Global Interprocessor Communication (IPC) Bus Blurb

MACD CORRUPTION AND PARITY ERROR

Figure 22 illustrates the process of detecting and reporting a memory bus parity error to the serial error reporting network. This reporting method is common to all errors that are detected by the interconnect system. The error reporting paths for this example are illustrated in Figure 21.

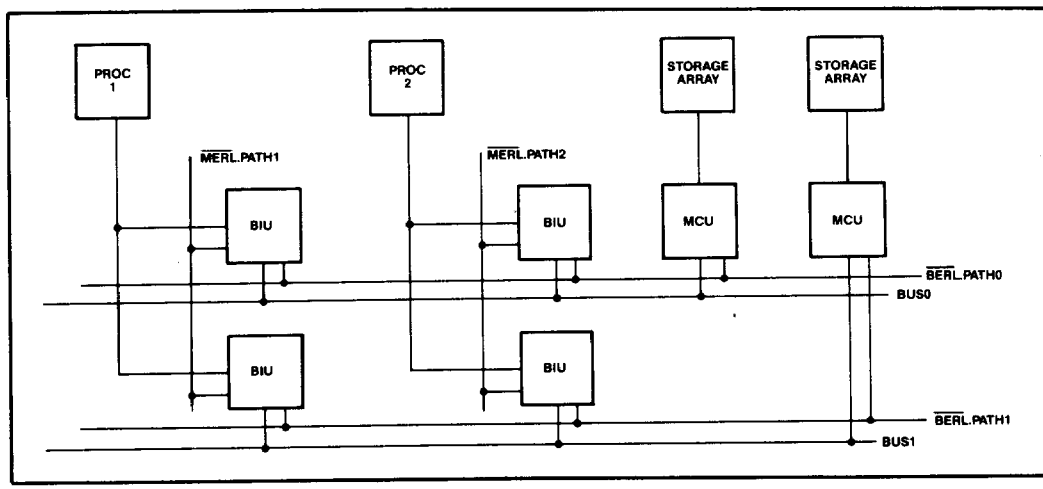
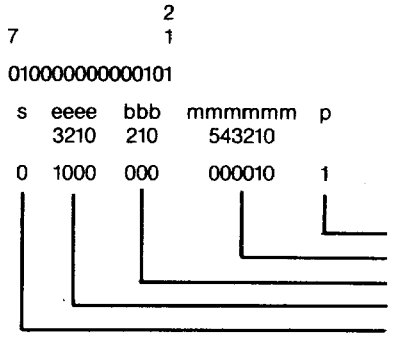


Figure 21. Hardware Configuration for Error Reporting

PHASE 1—DETECTING AND REPORTING THE PARITY ERROR ON BERL

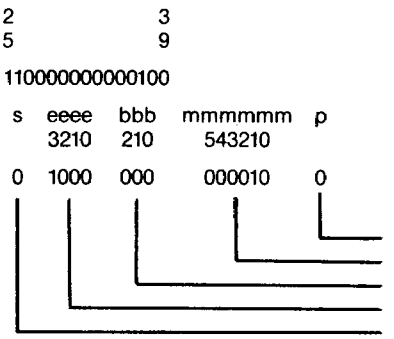
In cycle 3, bit 1 of the memory bus (MACD signals) is corrupted with a zero causing the MACD data to change from FFFFH to FFFDH. This causes a parity error on the memory bus that must be reported to the interconnect system. The reporting process begins in cycles 5 and 6 when the BERL1 signal (driven by BERLOUT) carries the start code of the serial error report. In cycles 7 through 21, the message is propagated along the bus error report path associated with bus 0. The message sent is (complement of the levels on the BERL1 signals):



The error report codes are described in the iAPX 432 Interconnect Architecture Reference Manual, order no. 172487 (see BIU Register 00—Error Report Log).

PHASE 2—BROADCASTING THE ERROR ON MERL

Once the full serial message has propagated along the memory bus, each BIU that received it repeats the message along the module error report line path (MERL1 and MERL2). This time, the error report message is:



PHASE 3—REBROADCASTING THE ERROR ON ALL BERL PATHS

In cycle 42, the message is again rebroadcast, this time along all the bus error report line paths. The message sent is:

```
4          5
4          8
1100000000000100
s  eeee   bbb  mmmmmm  p
   3210   210   543210
1  1000   000   000010  0
```

Cycle numbers

Binary serial error message on BERL1.PATH0

Reformatted serial error report message

Parity bit
Module number
Bus number
Error code
Sequence

Once this three-step reporting process is completed, all modules have been informed of the error.

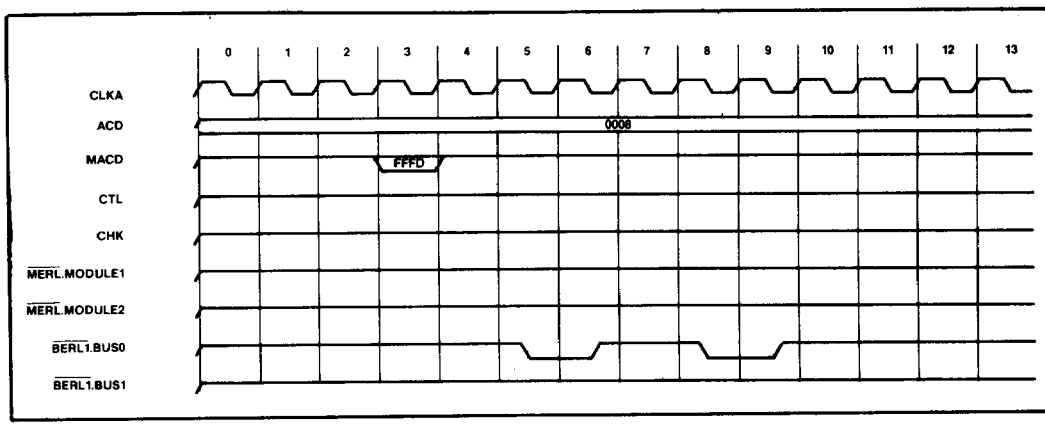


Figure 22. MACD Corruption Causing Parity Error

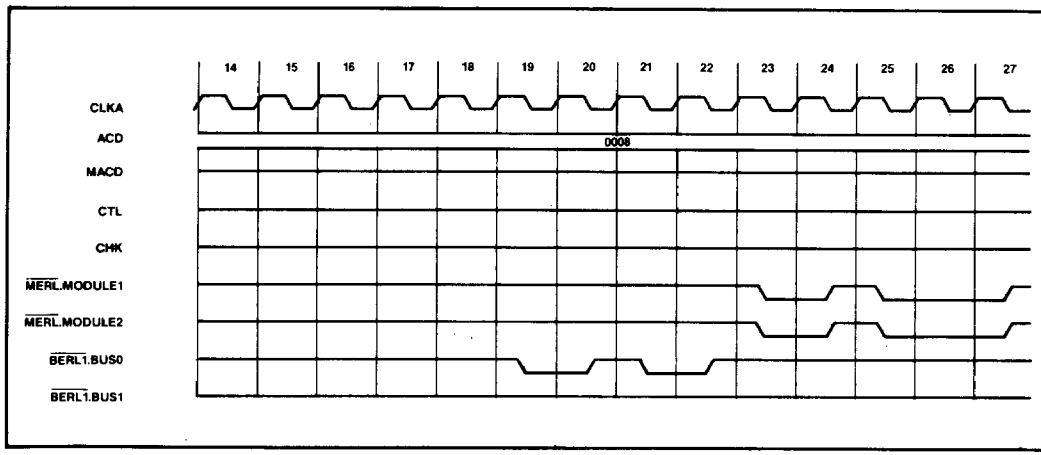


Figure 22. MACD Corruption Causing Parity Error (continued)

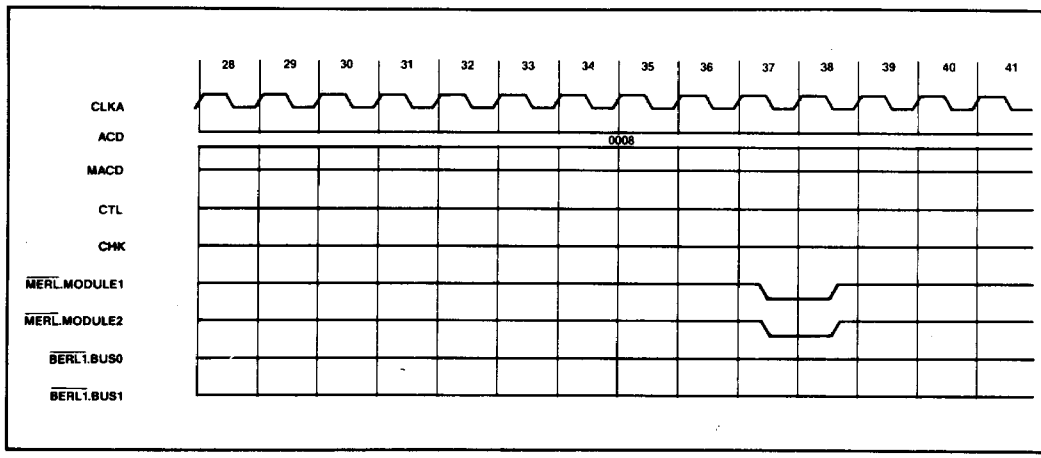


Figure 22. MACD Corruption Causing Parity Error (continued)

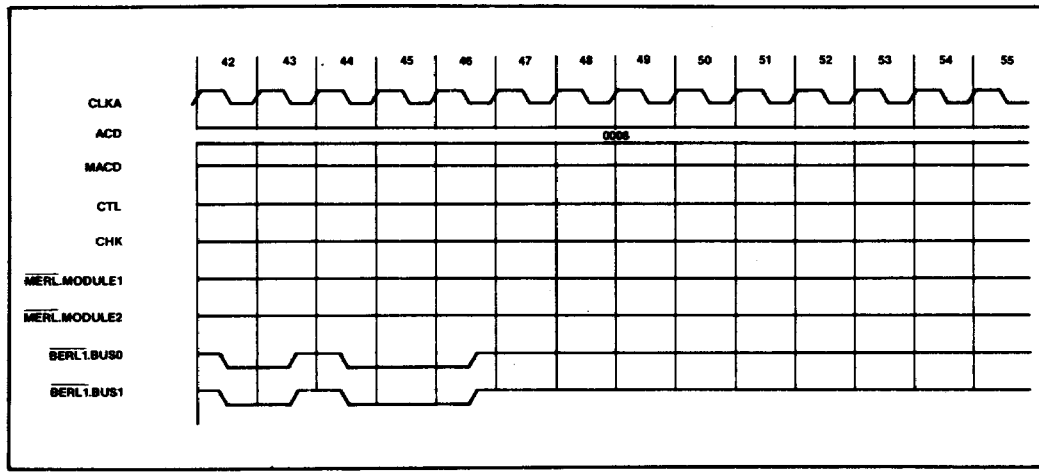


Figure 22. MACD Corruption Causing Parity Error (continued)

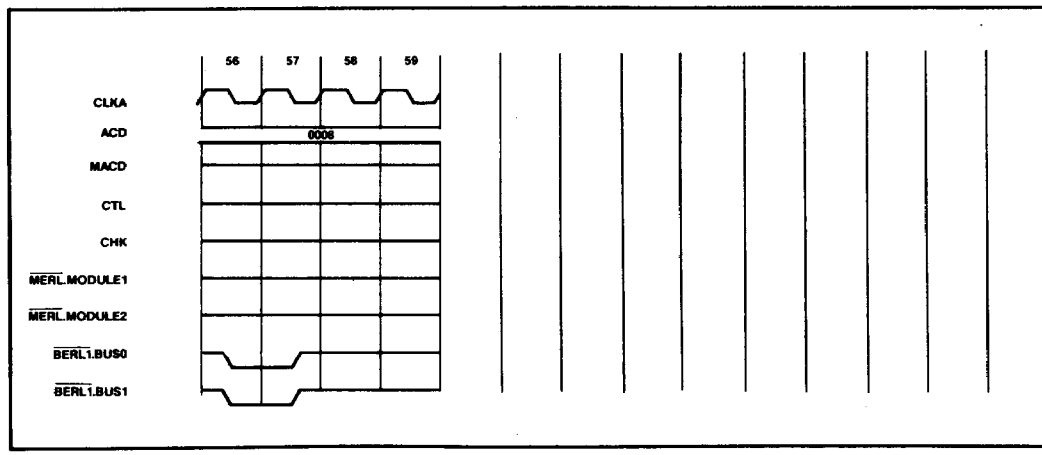


Figure 22. MACD Corruption Causing Parity Error (continued)

TEST DETECTION COMMAND

The Test Detection command is a special command that requests a BIU to test the detection hardware contained inside the component. It checks all the FRC circuitry, memory bus parity generator/checkers, and buffer checking logic. The hardware configuration for this example is the same as that for the previous example (see Figure 21). When the command has been performed an error report message is generated, just as in the previous example. When the test is successful, the message indicates "no error." When any of the detection circuitry has failed, the message indicates "module error." In this example, the detection circuitry is operating properly and phase 1 of the serial error reporting sequence contains the "no error" message starting in cycle 5 (see Figure 23).

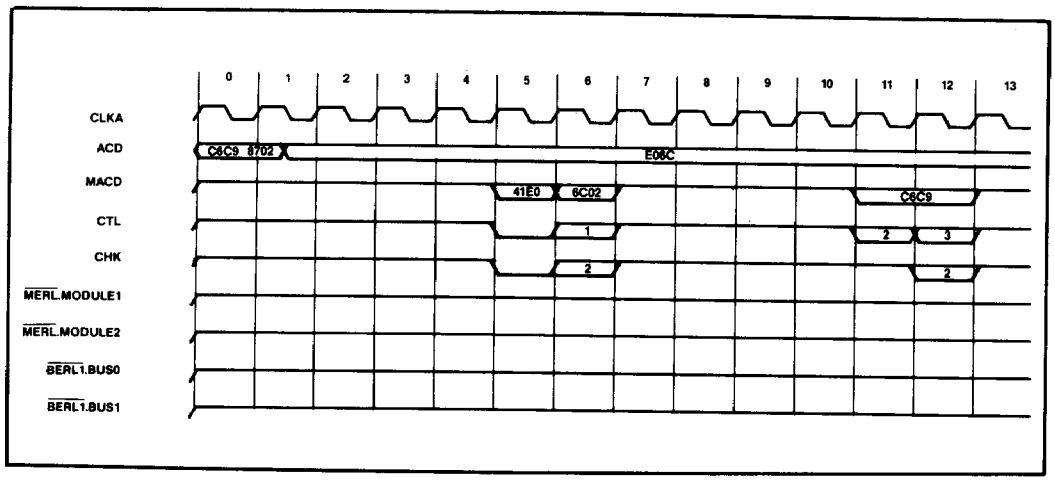


Figure 23. Test Detection Command

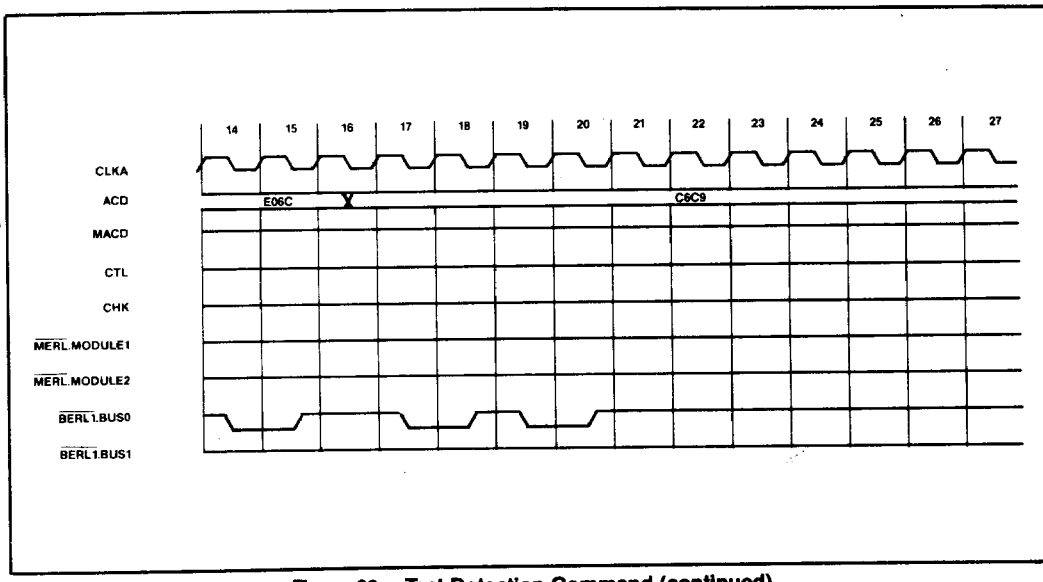


Figure 23. Test Detection Command (continued)

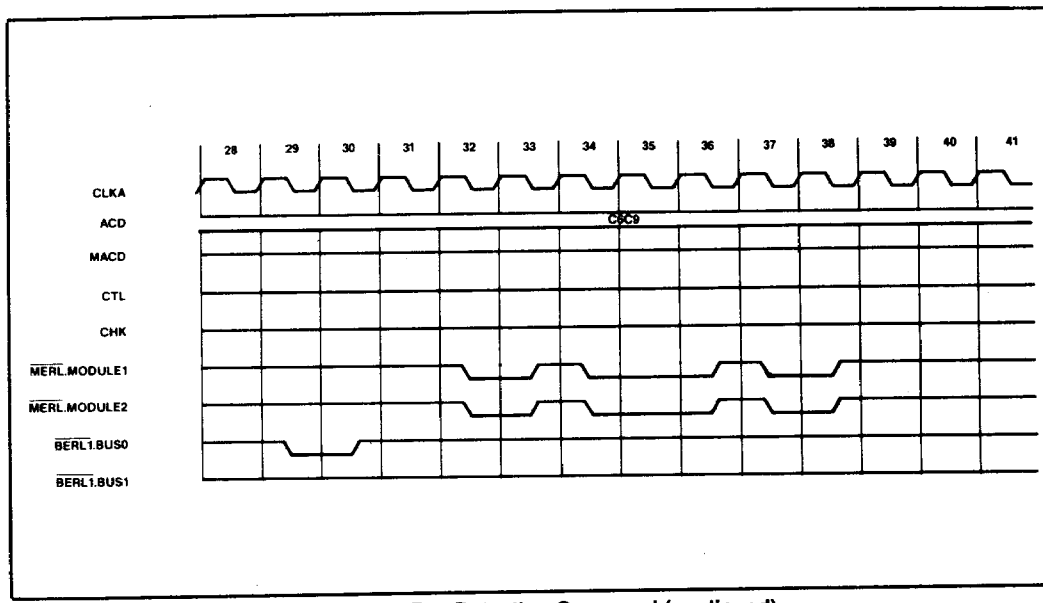


Figure 23. Test Detection Command (continued)

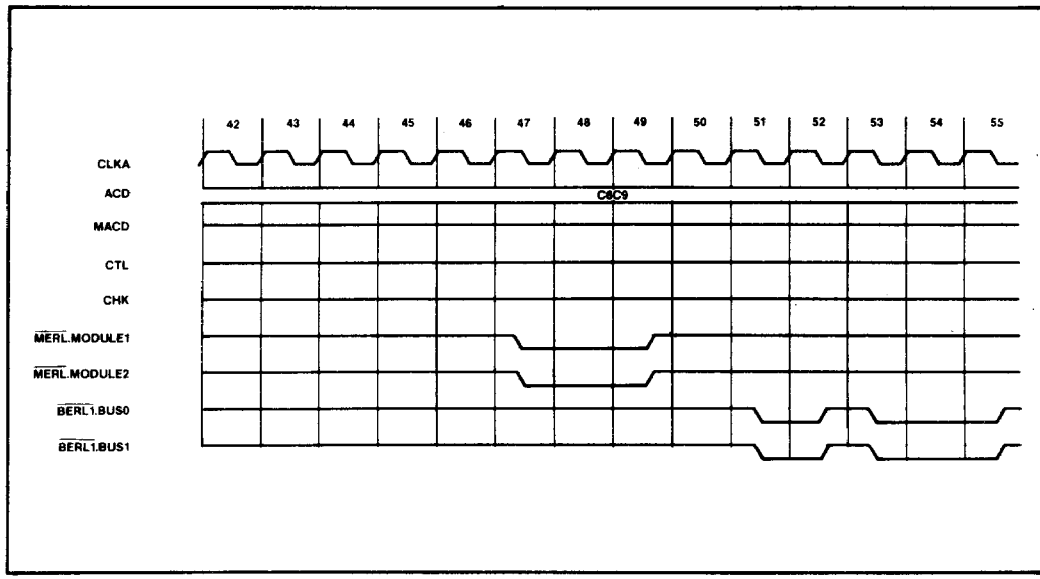


Figure 23. Test Detection Command (continued)

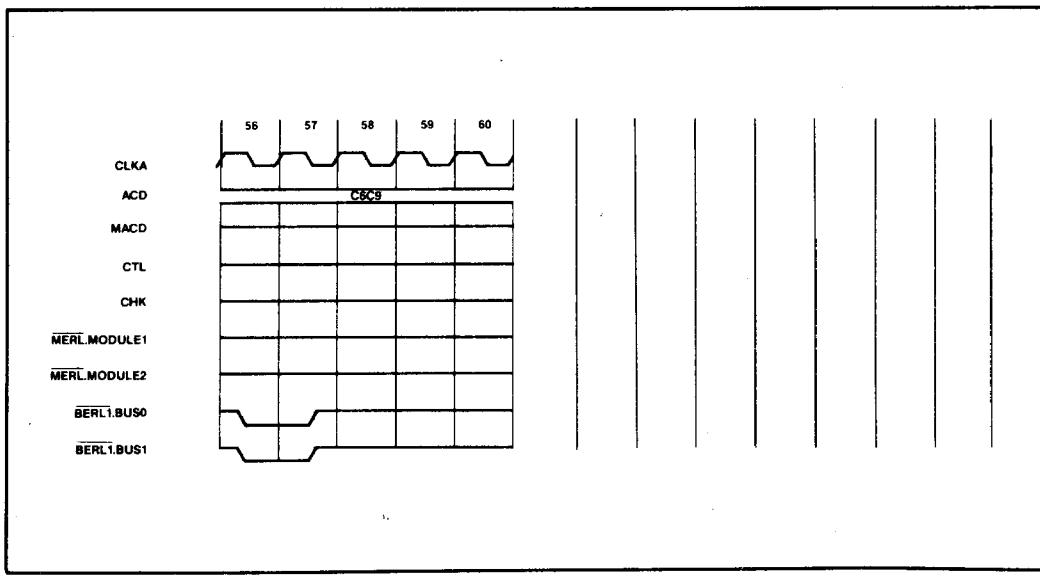


Figure 23. Test Detection Command (continued)

MACD BUS ARBITRATION

Figure 24 illustrates two processors as they simultaneously issue memory requests. Since the requests are both serviced by the same memory bus and MCU, arbitration is performed to select the first request that should be presented. The hardware configuration consists of two processors (processor modules 2 and 6), two BIUs, one memory bus, one MCU, and one storage array (see Figure 25).

In cycle 0, both processors issue a request indicated by the respective processor request signals, PRQ.PROC2 and PRQ.PROC6. In cycle 4, the associated BIUs present their requests to the memory bus (NREQ=0, RQ=0) and each notes that another BIU is also contending (CONT=0) for the use of the memory bus. The BIUs resolve the contention by examining the individual bits of their respective logical IDs, beginning with the most significant bit. In this case, the default logical ID is a bit-reversed version of the module ID. In cycle 4, each BIU notes contention. In cycle 5, each BIU still notes contention. In cycle 6, the BIU serving module 2 wins the arbitration and issues its request to the memory bus. Overlapped with the first request, the BIU in module 6 reissues its request for the memory bus (RQ=0 in cycle 9) and detects no contention. However, each BIU monitors the memory bus and the BIU in module 6 waits for the current bus activity to complete before its request can be placed on the bus.

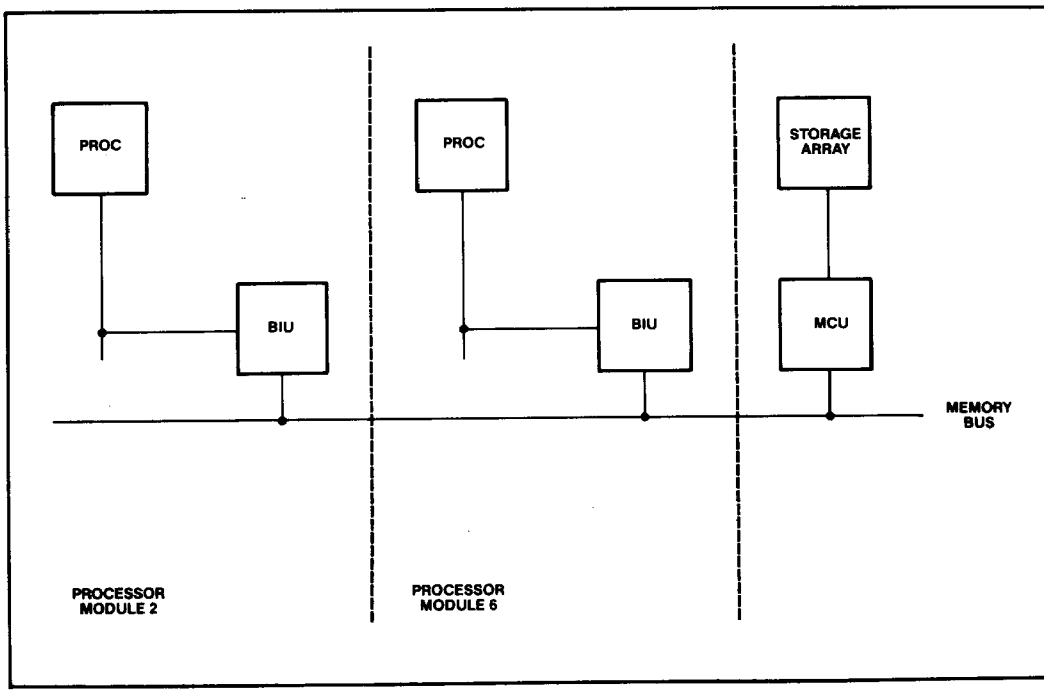


Figure 25. Hardware Configuration for Memory Bus Arbitration

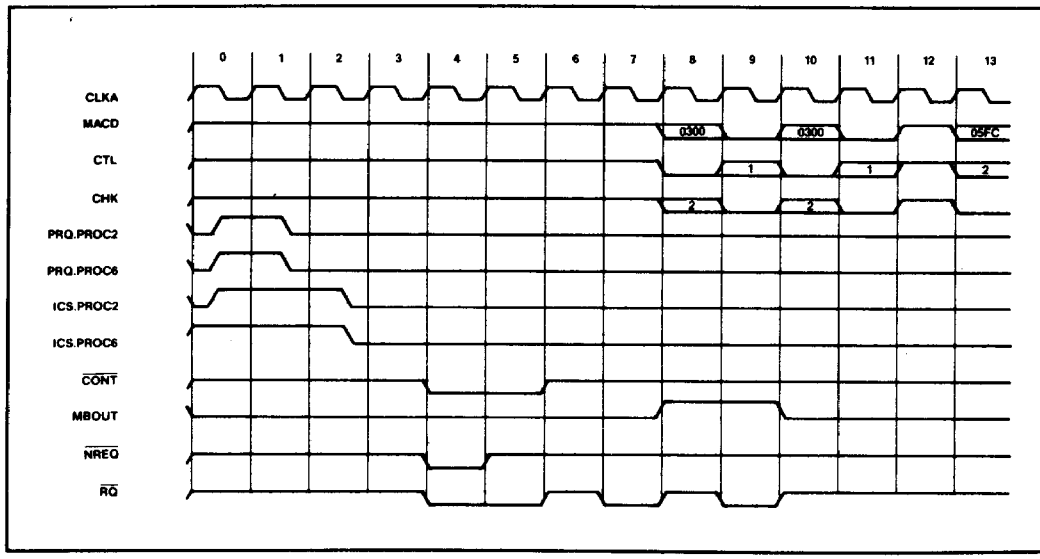


Figure 24. MACD Bus Arbitration

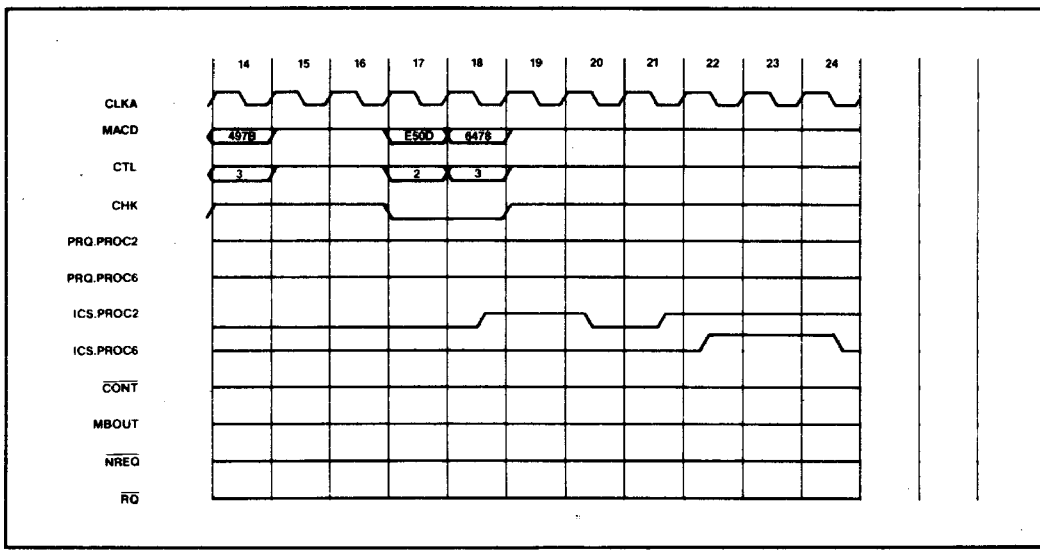


Figure 24. MACD Bus Arbitration (continued)

Table 3. iAPX 43204 Clock Edge Table

Signal	Pin(s)	Input Sample		Output Drive	
		Clock	Edge	Clock	Edge
Inputs					
MERL	44	CLKA	Rising	n/a	
BERL1	38	CLKA	Rising	n/a	
BERL2	39	CLKA	Rising	n/a	
ICS	48	CLKA	Rising	n/a	
INIT	37	CLKA	Rising	n/a	
RQ	9	CLKA	Falling	n/a	
CONT	10	CLKA	Falling	n/a	
NREQ	5	CLKA	Falling	n/a	
PRQ	49	CLKB	Rising	n/a	
Outputs					
MERLOUT	46	n/a		CLKA	Rising
BERLOUT	40	n/a		CLKA	Rising
ICSOUT	47	n/a		CLKB	Rising
CLRPUOUT	45	n/a		CLKB	Rising
Input/Output					
MMAH	50	CLKA	Rising	CLKA	Rising
MMAL	51	CLKA	Rising	CLKA	Rising
ACD15 . . . 0	1, 54-68	CLKB	Rising	CLKB	Falling
MACD15 . . . 0	13-28	CLKA	Rising	CLKB	Rising
CTL2 . . . 0	29-31	CLKA	Rising	CLKB	Rising
CHK1 . . . 0	11, 12	CLKA	Rising	CLKB	Rising
MBOUT	32	CLKA	Rising	CLKA	Rising
NREQOUT	4	CLKA	Falling	CLKA	Falling
RQOUT	6	CLKA	Falling	CLKA	Falling
BCHK	7	CLKA	Rising	CLKB	Rising

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Range ..... 0°C to 70°C
Storage Temperature .....−65°C to +150°C
Voltage on Any Pin with
Respect to Ground ..... −1.0V to +7V
Power Dissipation ..... 2.2 Watt

**NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

iAPX 43204 DC Characteristics

Symbol	Description	Min	Max	Units
Vilc	Clock input low voltage	−0.5	+0.8	V.
Vihc	Clock input high voltage	3.2	Vcc+0.5	V.
Vil	Input low voltage	−0.5	+0.8	V.
Vih	Input high voltage	2.0	VCC+0.5	V.
Vol	Output low voltage	—	0.45	V.
Voh	Output high voltage	2.4	VCC	V.
Ili	Input leakage current (measured at Vin=VCC)	—	±10	μA.
Ilo	Output leakage current (measured at 0.45 V. ≤ Vout ≤ VCC)	—	±10	μA.
Ioh	Output high current (measured at 2.6 V.)	−2	—	mA.
Iol	Output low current (measured at 0.45 V.)	4	—	mA.
Icc	Power supply current (sum of VCC0, VCC1, VCC2)	—	400	mA.

All DC parameters are guaranteed over the following conditions:

VSS2..VSS0 = 0 Volts
VCC2..VCC0 = 5.0 Volts ± 5%
The absolute value of the differential DC voltage between any of the VCC pins (VCC2..VCC0) must be less than 0.1 Volts. This is normally guaranteed by connecting the three VCC pins to the same printed circuit power trace.

iAPX 43204 AC Characteristics

Ambient temperatrue range of 0°C to 70°C

Symbol	Description	5 MHz		7 MHz		8 MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_r, t_f	Clock rise and fall times	-	13	-	11	0	10	nsec
t_1, t_2, t_3, t_4	Clock pulse width	37	250	25	250	24	250	nsec
t_{cy}	Clock cycle time	200	1000	143	1000	125	1000	nsec
t_{cd}	Clock to signal delay time	-	70	-	60	-	55	nsec
t_{dh}	Clock to signal hold time	20	-	17	-	15	-	nsec
t_{en}	Clock to signal output enable time	20	-	17	-	15	-	nsec
t_{df}	Clock to signal data float time	-	50	-	44	-	40	nsec
t_{dc}	Signal to clock setup time	30	-	26	-	24	-	nsec
t_{ie}	Initialization period	20	100	20	100	20	100	t_{cy}

All AC parameters are guaranteed over the following conditions:

Ambient temperature range of 0 degrees Centigrade to 70 degrees Centigrade

VSS2 . . . VSS0 = 0 Volts

VCC2 . . . VCC0 = 5.0 Volts ± 10%

100 picoFarad external load capacitor on all output pins

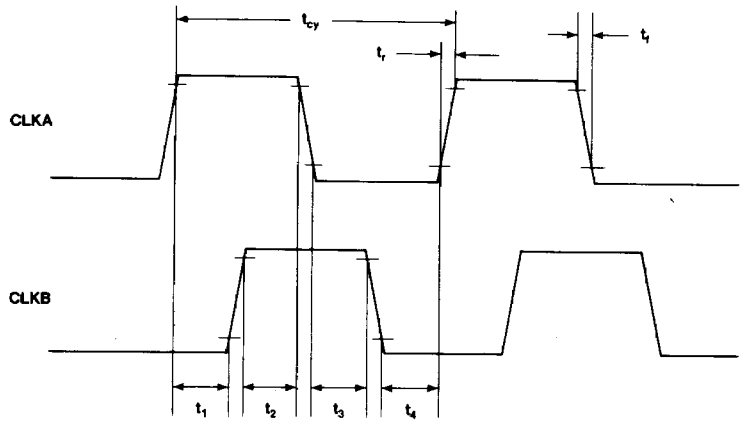
iAPX 43204 Capacitance Data

Conditions: Ta = 25°C
VCC = 5.0 Volts, GND = 0.0 Volts
f(test) = 1.0 MHz
Inputs held at 0.0 Volts
All outputs in high impedance state
All input/output pins are classified as outputs

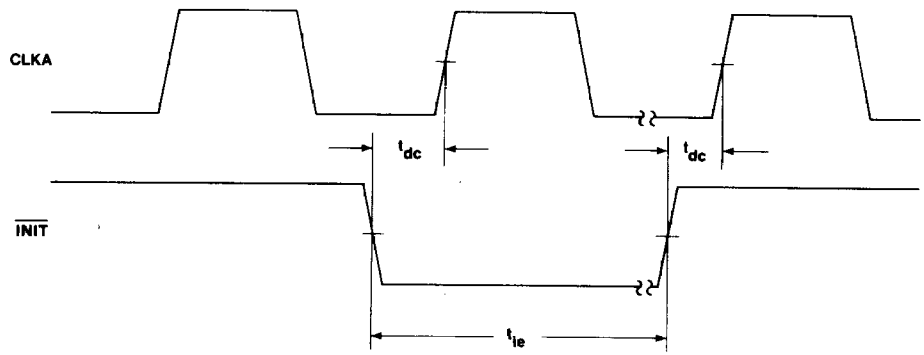
Symbol	Description	Max	Units
Cin	Input Capacitance	6	pF
Cout	Output Capacitance	12	pF

WAVEFORMS

iAPX 43204 Clock Input Specification

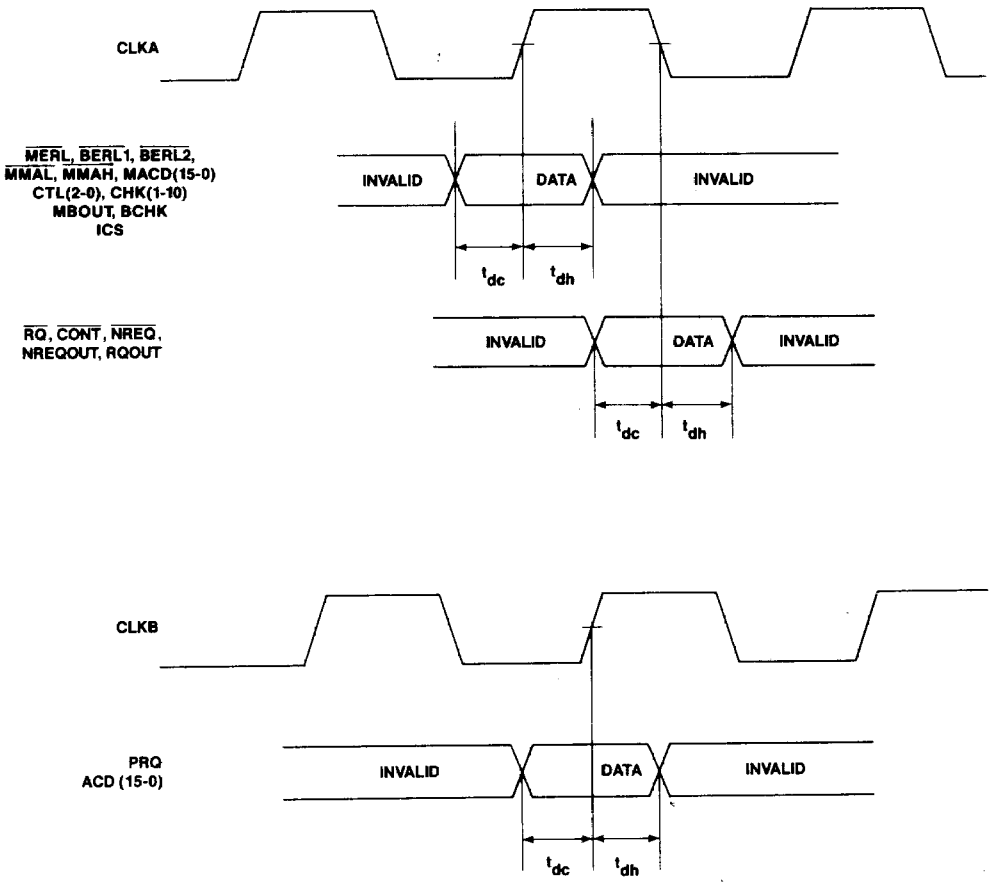


iAPX 43204 Initialization Timing



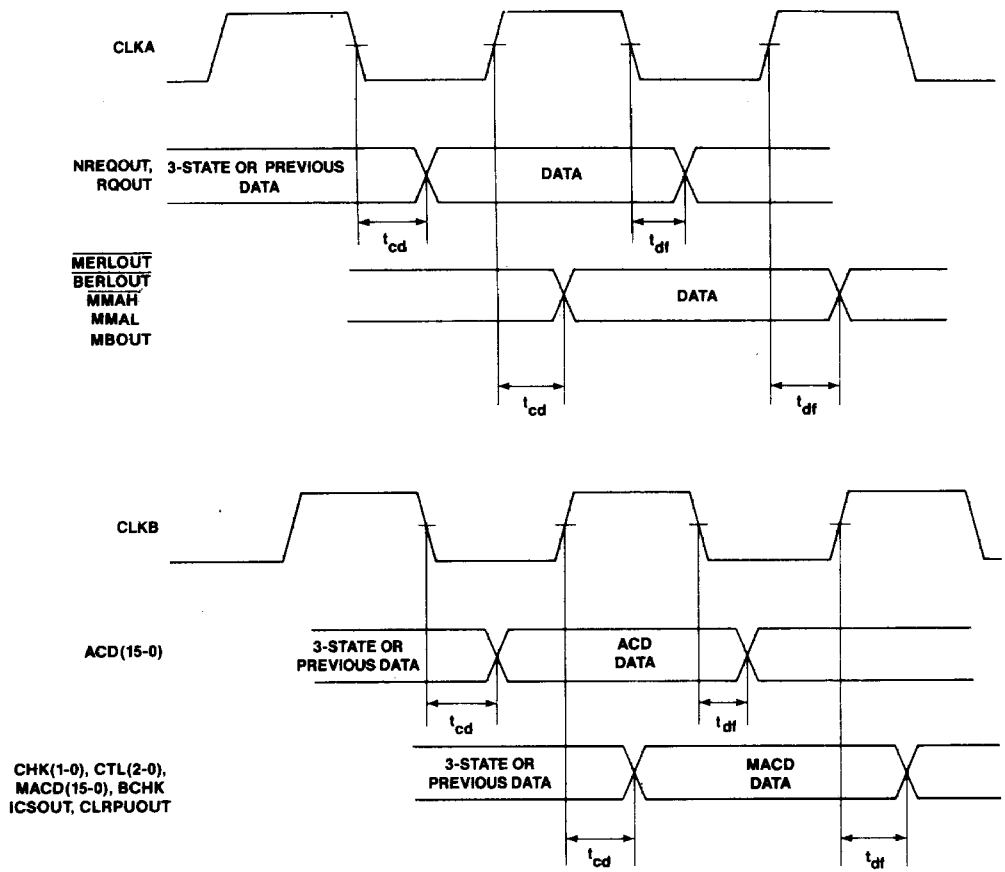
WAVEFORMS (Continued)

iAPX 43204 Input Timing Specification



WAVEFORMS (Continued)

iAPX 43204 Output Timing Specification



iAPX 43205 FUNCTIONAL DESCRIPTION

This section describes how the iAPX 43205 Memory Control Unit (MCU) operates. It contains a set of diagrams that present the clock-by-clock activity of the pins on an MCU component. To understand the notation on the waveforms, refer to Figure 26. It illustrates a 32-bit memory read operation of physical memory array address 00000H.

The cycle numbers (0, 1, 2, . . .) at the top of each waveform enumerate the clock cycles. When a common group of signals is plotted, its value is displayed inside a data waveform. For example, the MACD signals (MACD15 . . . MACD0) are all high in cycle 0, carry the hexadecimal value 0300H in cycle 1, and are all low in cycle 3. The MACD signals (MACD15 . . . MACD0) carry the values 0300H in cycle 1 and 0000H in cycle 2. The first double byte of the access contains the specification field (03H—the high byte of the first double byte) and the low-order address byte (00H—the low byte of the first double byte) of the memory address. The second double byte of the access contains the high- (00H) and mid-order (00H) address bytes. This example illustrates a double byte read performed at physical memory address 00000H. Also, notice that the SLAD pins (SLAD19 . . . SLAD0) are represented with a 5-digit hexadecimal value.

Consider the SLAD group in the example. In the last half of cycle 3 and during cycle 4, the SLAD group carries the value 00000H, the storage array address for the access. In cycle 5 the array returns read data of 42200H, and in cycle 6 the array responds with DC664H. In this example, the storage array is physically 20-bits wide, and the least significant physical bit is a spare bit. Thus, to interpret the actual data present on the SLAD wires, a logical shift is necessary to align the data as it will be presented on the MACD signals.

Original SLAD Information

Cycle 5	4	2	2	0	0	Hexadecimal
	0100	0010	0010	0000	0000	Binary
					↕	Spare Bit
Cycle 6	D	C	6	6	4	Hexadecimal
	1101	1100	0110	0110	0100	Binary

Cycle 5 contains the least significant information, and cycle 6 contains the most significant information. The least significant bit of cycle 5, the spare bit, is not used in this case. Thus, the actual data and ECC information transferred on the SLAD signals is formed by concatenating the low- and high-order SLAD data groups and deleting the low-order spare bit. The 7 ECC bits occupy the high-order 7 bits of the result, and the 32 data bits occupy the low-order portion.

Original SLAD Information

D	C	6	6	4	4	2	2	0	0	
1101	1100	0110	0110	0100	0100	0010	0010	0000	0000	
									↕	
6	0	31							0	Spare bit
ECC								Data		

Spare Bit Removed

6	E	3	3	2	2	1	1	0	0
110	1110	0011	0011	0010	0010	0001	0001	0000	0000
6	0	31							0
ECC								Data	

The 32-bit value 33221100H is returned to the MACD bus with the low-order 16 bits in cycle 7 and the high-order 16 bits in cycle 8. The 7-bit ECC value is checked by the MCU and is not transferred to the MACD signals.

These waveforms are accurate cycle-to-cycle representations of MCU function. They are not meant to serve as diagrams of exact component timing. For example, the diagrams depict MACD data being issued from the MCU at the functional clock boundaries for the rising edge of CLKA. The MCU actually sources MACD information on the rising edge of CLKB. Always refer to the precise electrical specifications, AC specifications, and timing diagrams when detailed timing information is required. The functional diagrams are intended to succinctly summarize the functions of the MCU.

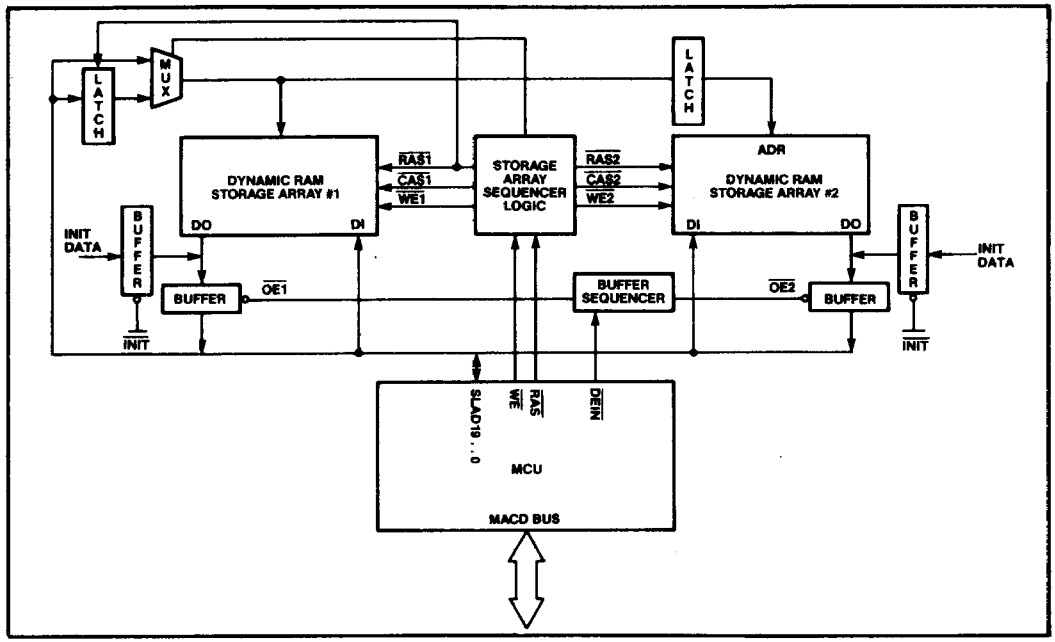


Figure 27. MCU and External Hardware Configuration

HARDWARE CONFIGURATION FOR MEMORY READ AND WRITE OPERATIONS

Figure 27 is the basis for demonstrating MCU memory read and write operations. In each memory read and memory write example, the appropriate command is presented to the MCU from the MACD (memory) bus and the MCU performs the operation to completion. Each memory operation assumes the following hardware configuration for the logic external to the MCU. Notice that external logic accepts the RAS, WE, and DEIN signals and produces the precise timing signals which are required by the associated dynamic RAM storage array. In the diagrams which follow, only signals which are present on the MCU signal pins are displayed.

MEMORY READ OPERATIONS

Figure 26 (Normal Aligned Read) illustrates a memory read operation that returns two double bytes (4 bytes) from the memory subsystem, typical for iAPX 432 GDP instruction fetch cycles.

Figure 28 (Normal 2-byte Read) illustrates a memory read operation that returns a single double byte operand.

Figure 29 (Normal Nonaligned Read) illustrates a 4-byte memory read operation that is not aligned to a 4-byte boundary, the natural boundary for instruction fetches and 32-bit operands. In this case, the MCU must perform two separate storage array accesses to acquire the 4 bytes.

Figure 30 (4-byte Read with Extended Access) illustrates a programmable option on the MCU that accommodates storage arrays that require longer memory access cycles. With the extended access option, the MCU extends RAS one cycle and delays DEIN one cycle to allow the external array sequencing logic to extend the storage array cycle.

Figure 31 (Staged Read with Correctable Error) demonstrates how an MCU can stage (hold) memory data until all ECC detection *and* correction is performed. Without selecting the staging option, a MCU would normally present the memory data it acquired to the MACD bus as in other memory read cycles. However, should an error be detected, the MCU would signal (BERLOUT) to inform a BIU that data it had been given was invalid, and the BIU would retry the access.

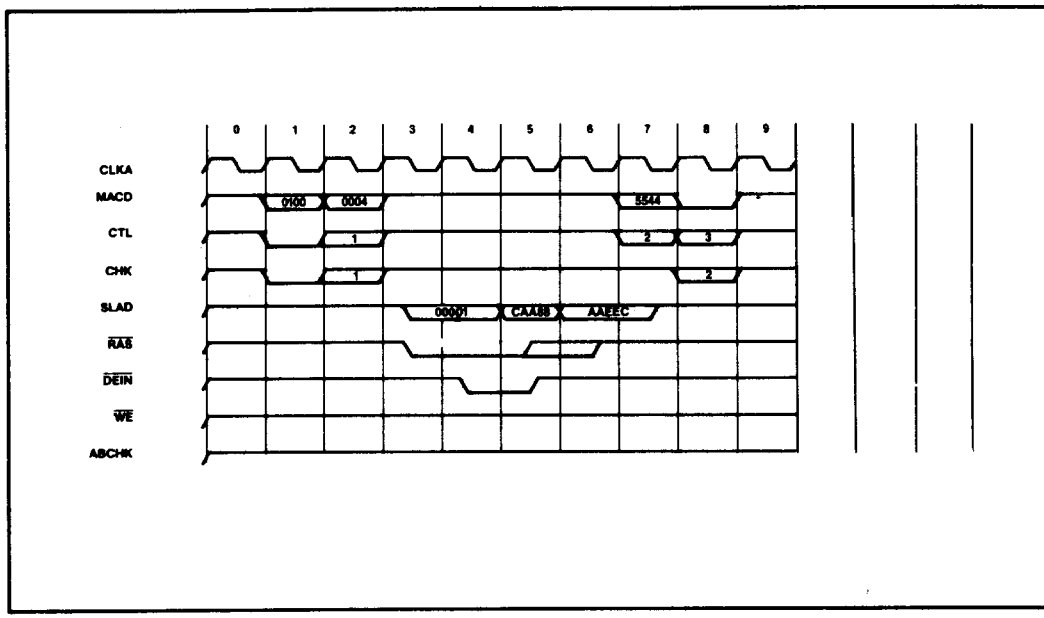


Figure 28. Normal 2-byte Read

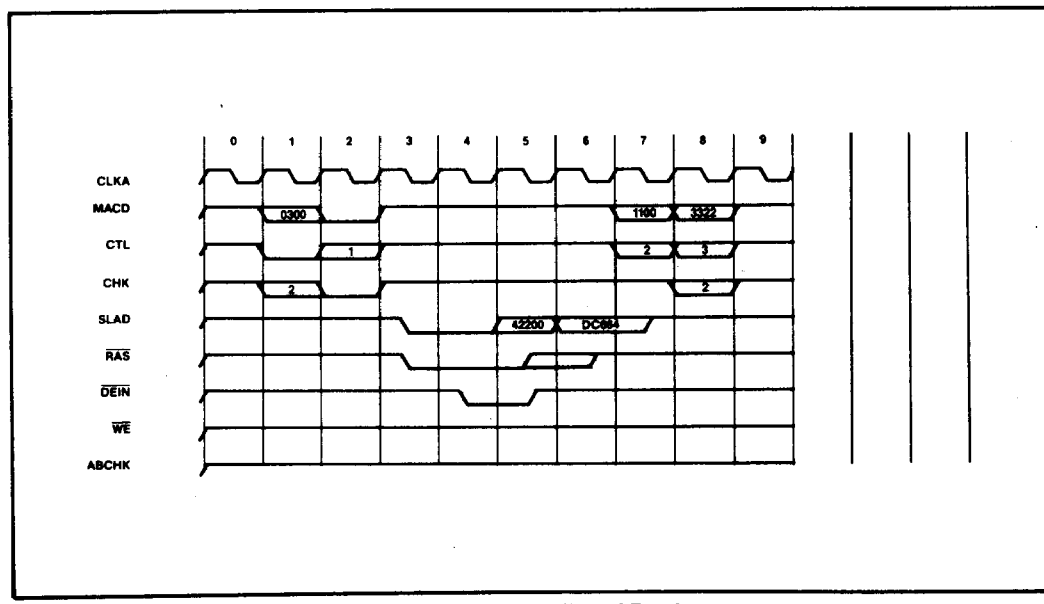


Figure 26. Normal Aligned Read

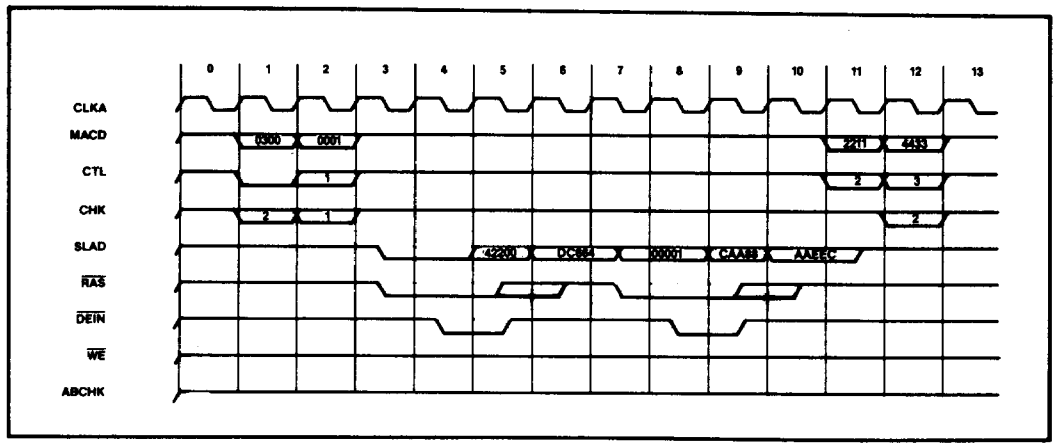


Figure 29. Normal Nonaligned Read

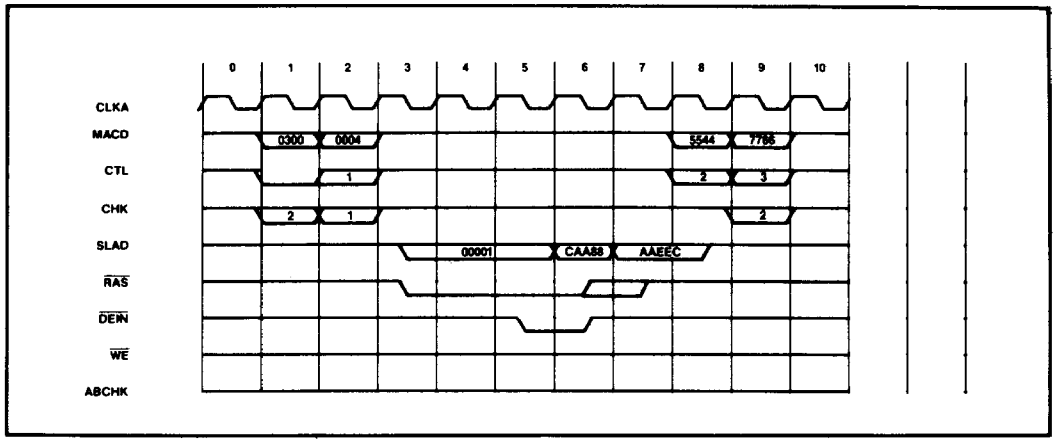


Figure 30. 4-Byte Read with Extended Access

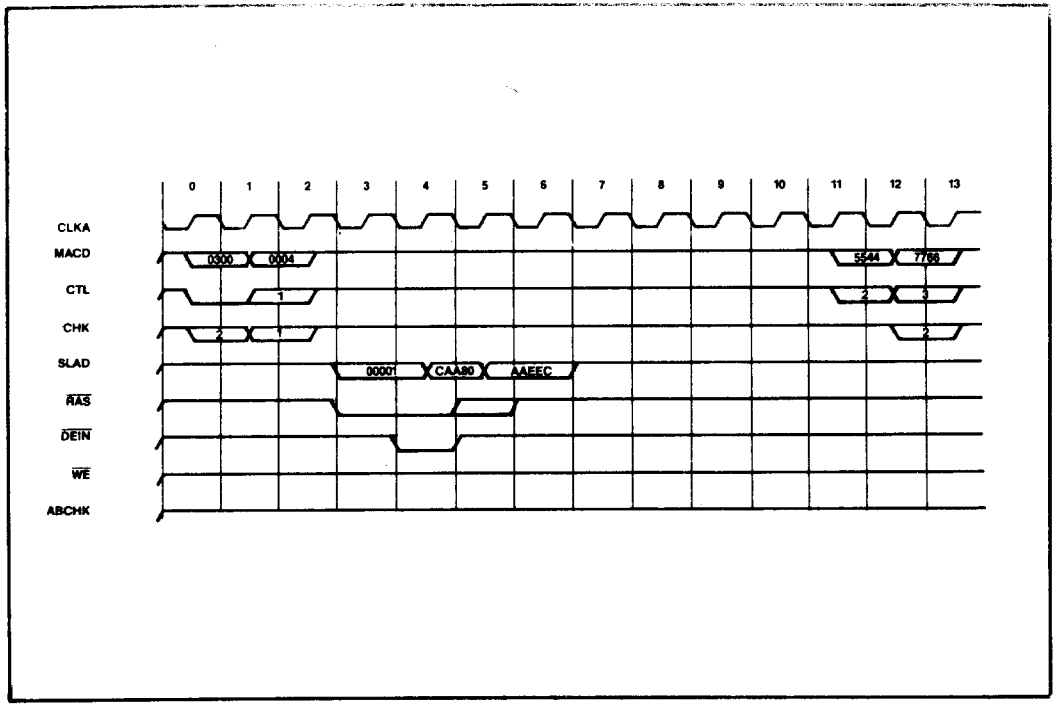


Figure 31. Staged Read with Correctable Error

MEMORY WRITE OPERATIONS

Figure 32 (Normal 2-byte Write) illustrates a memory write operation that stores a single double byte operand.

Figure 33 (Normal Aligned Write) illustrates a memory write operation that stores a 4-byte operand aligned to a 4-byte boundary.

Figure 34 (Normal Nonaligned Write) illustrates a 4-byte memory write operation that is not aligned to a 4-byte boundary. In this case, the MCU must perform two separate storage array accesses to store the 4 bytes.

Figure 35 (4-byte Write with Extended Access) illustrates a programmable option on the MCU that accommodates storage arrays which require longer memory access cycles. With the extended access option, the MCU extends RAS one clock cycle and delays DEIN one clock cycle to allow the external array sequencing logic to extend the storage array cycle.

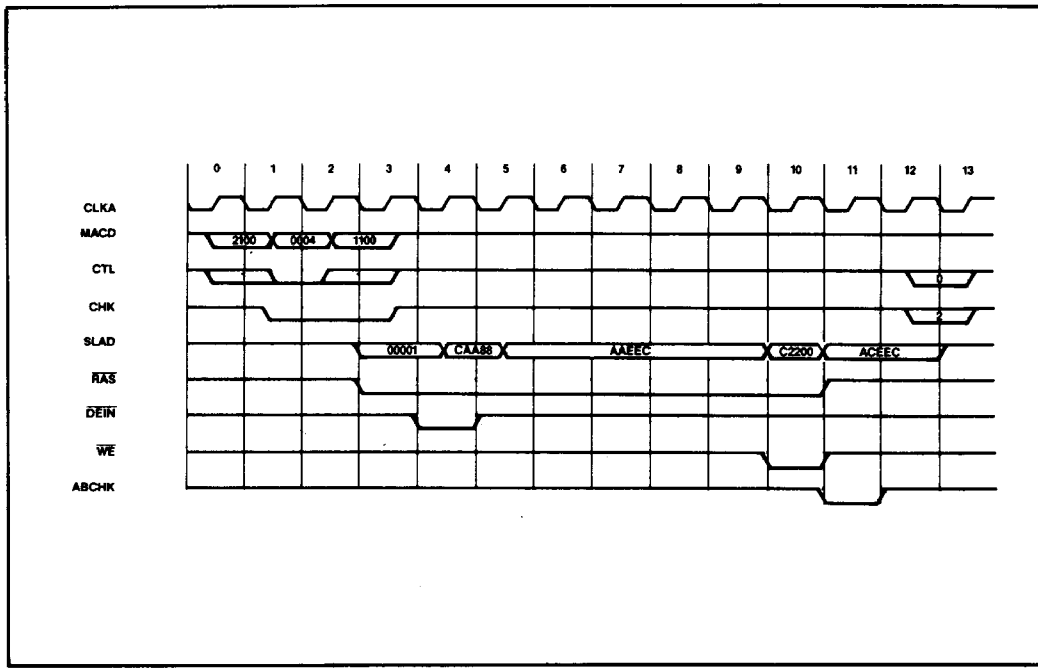


Figure 32. Normal 2-byte Write

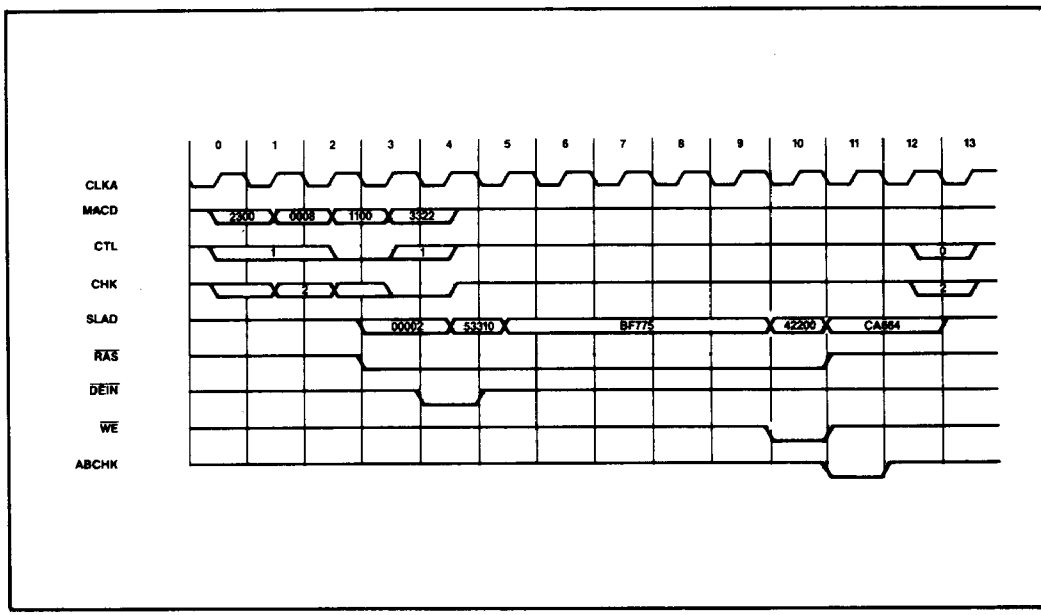


Figure 33. Normal Aligned Write

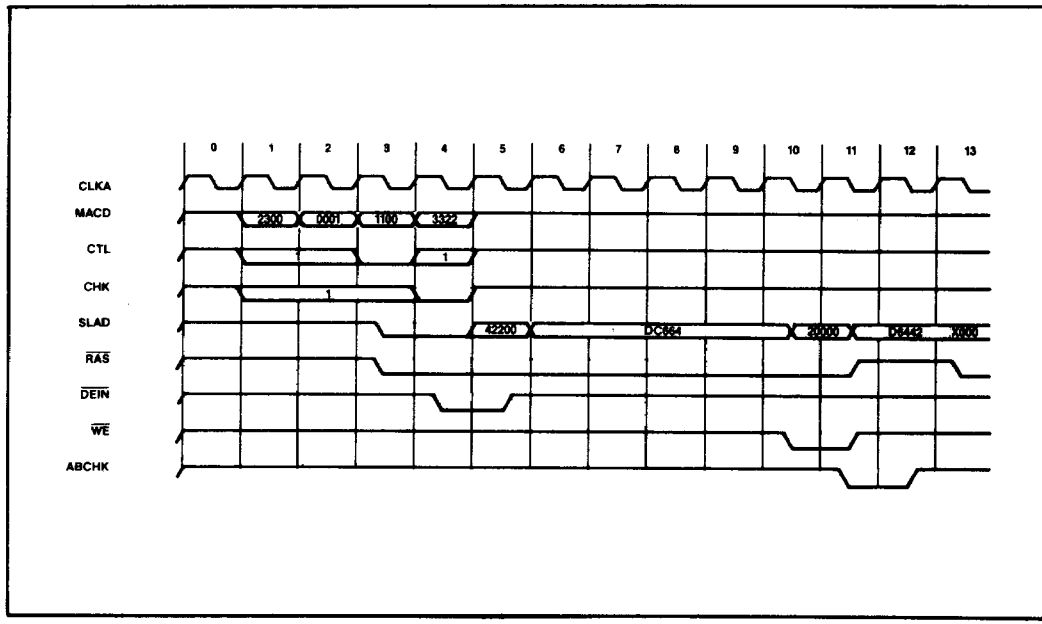


Figure 34. Normal Nonaligned Write

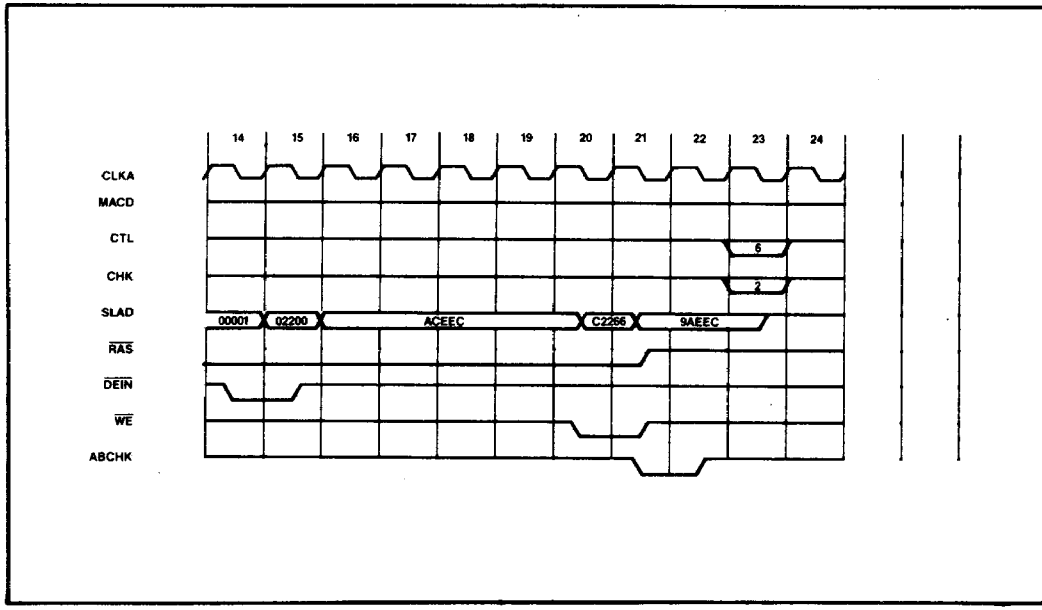


Figure 34. Normal Nonaligned Write (continued)

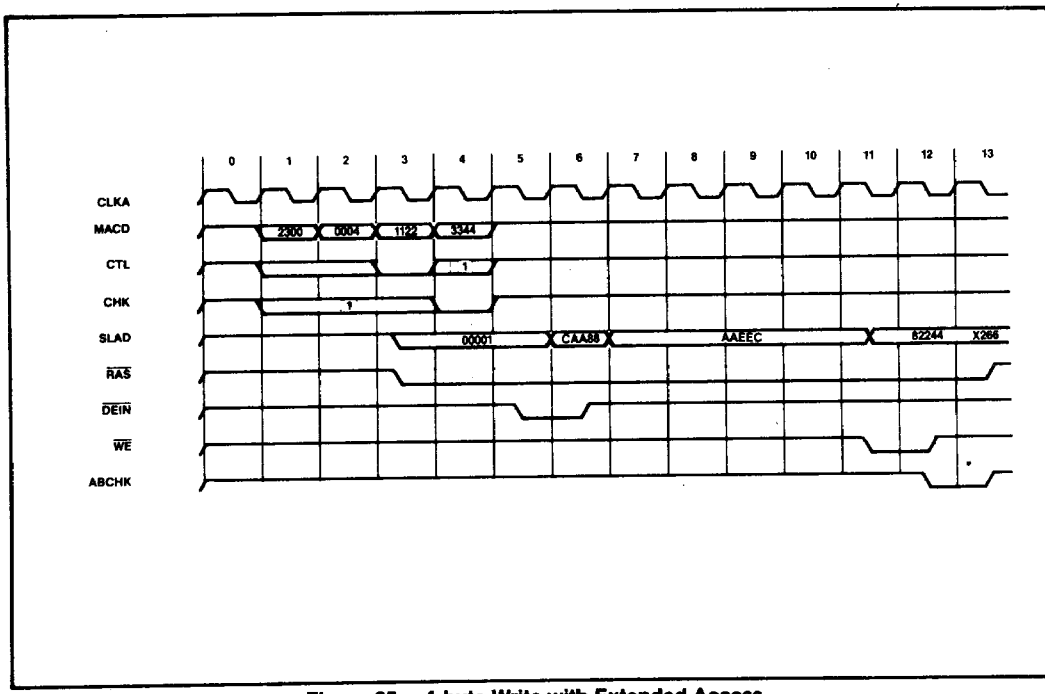


Figure 35. 4-byte Write with Extended Access

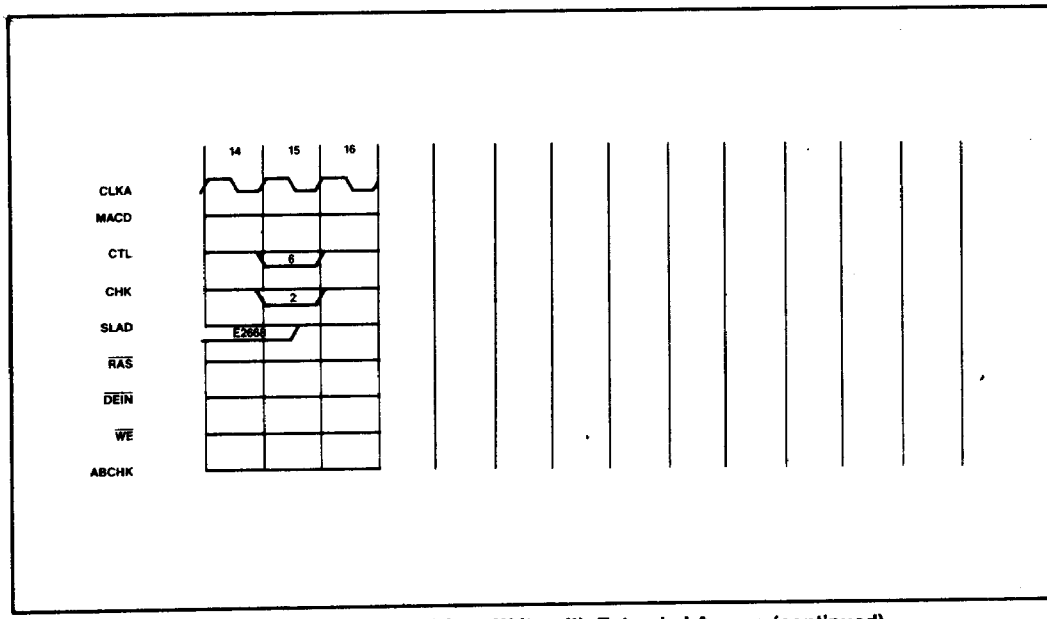


Figure 35. 4-byte Write with Extended Access (continued)

MEMORY REFRESHING OPERATIONS

Figures 36 and 37 illustrate an MCU performing two types of storage array refreshing operations. The MCU performs a *Normal Refresh Cycle* to satisfy standard dynamic RAM refresh requirements. External logic must use the REFRESH signal to generate the appropriate storage control signals (e.g., RAS-only refresh). The MCU performs a *Scrub Refresh Cycle* to cleanse the memory array of latent ECC errors by periodically reading the storage array and writing back a corrected version of data for any correctable errors it detects. Each of these diagrams utilize the same hardware configuration as the memory read and write operations described earlier. However, the MCU performs the refreshing operations independently of any commands from the MACD bus.

INTERCONNECT REGISTER OPERATIONS

Figures 38 and 39 illustrate the reading and writing of interconnect registers that are located on the MCU. Refer to the iAPX 432 Interconnect Architecture Reference Manual for a detailed description of specific interconnect registers. These diagrams illustrate general interconnect register operations. Later, specific commands are demonstrated that are invoked by interconnect register operations.

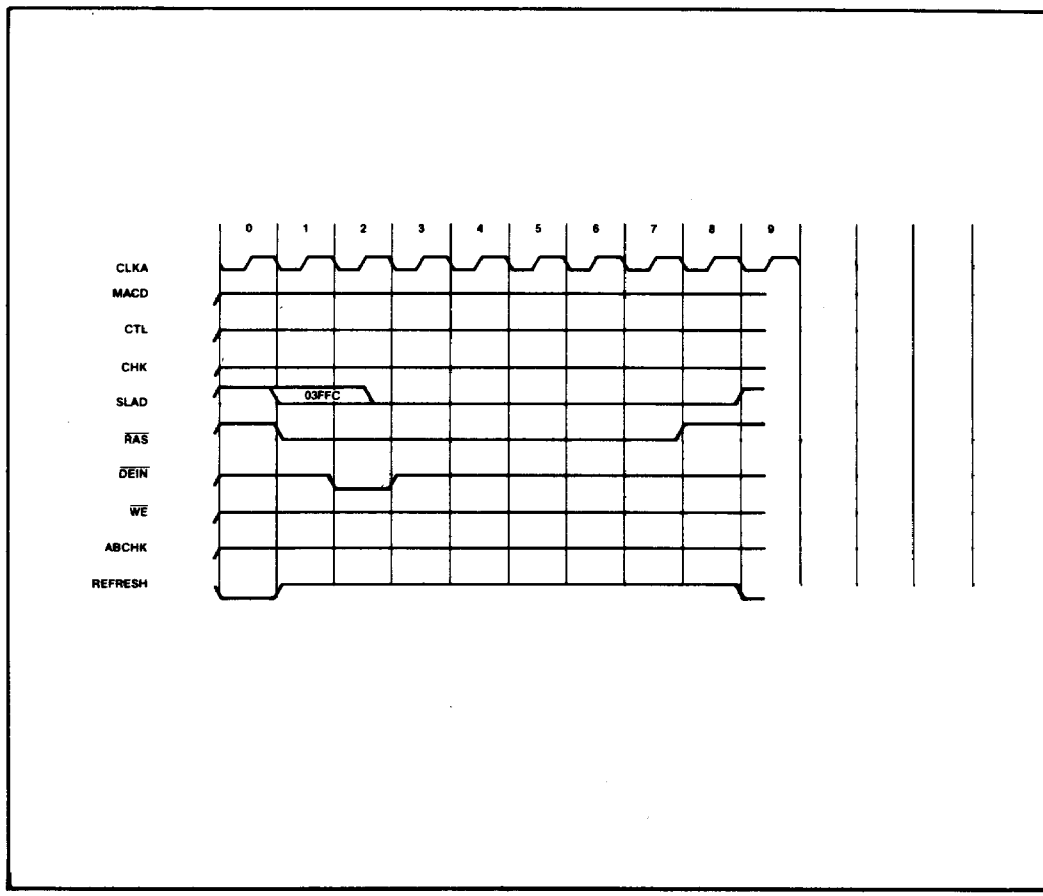


Figure 36. Normal Refresh Cycle

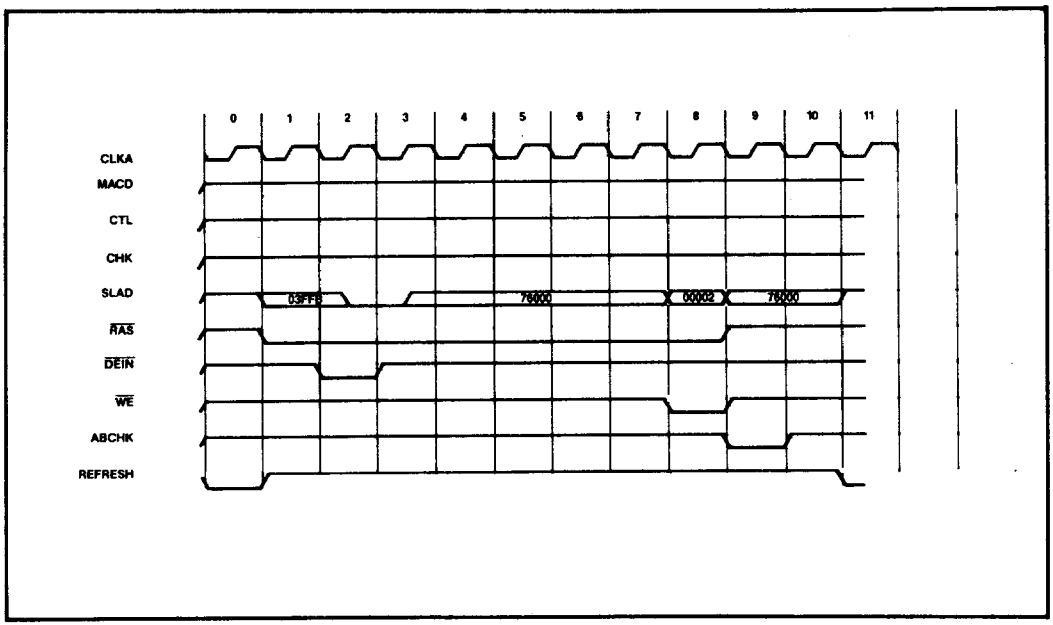


Figure 37. Scrub Refresh Cycle

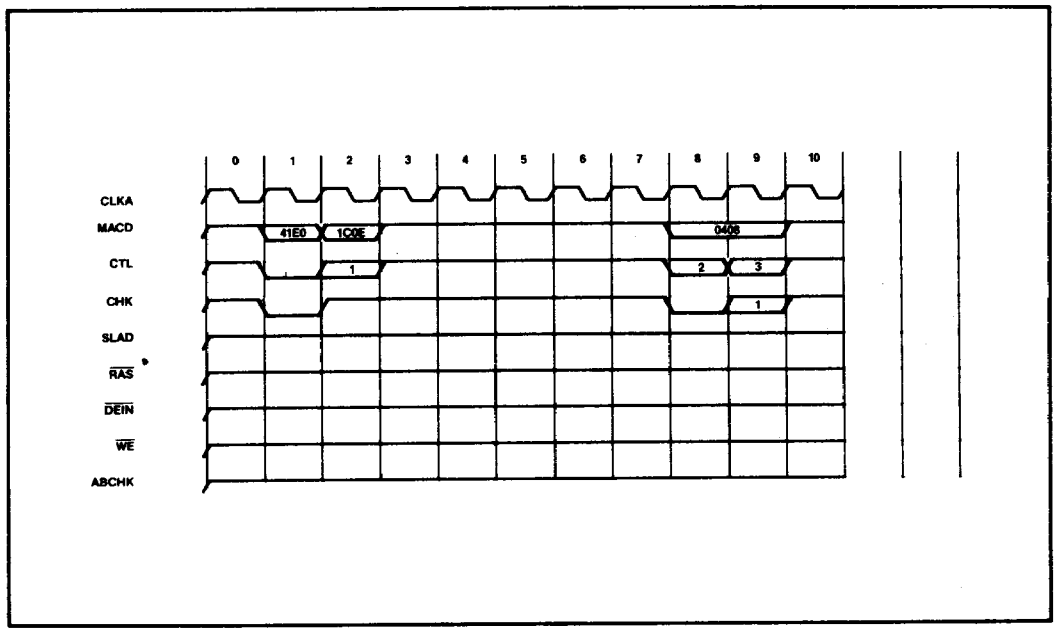


Figure 38. Interconnect Register Read

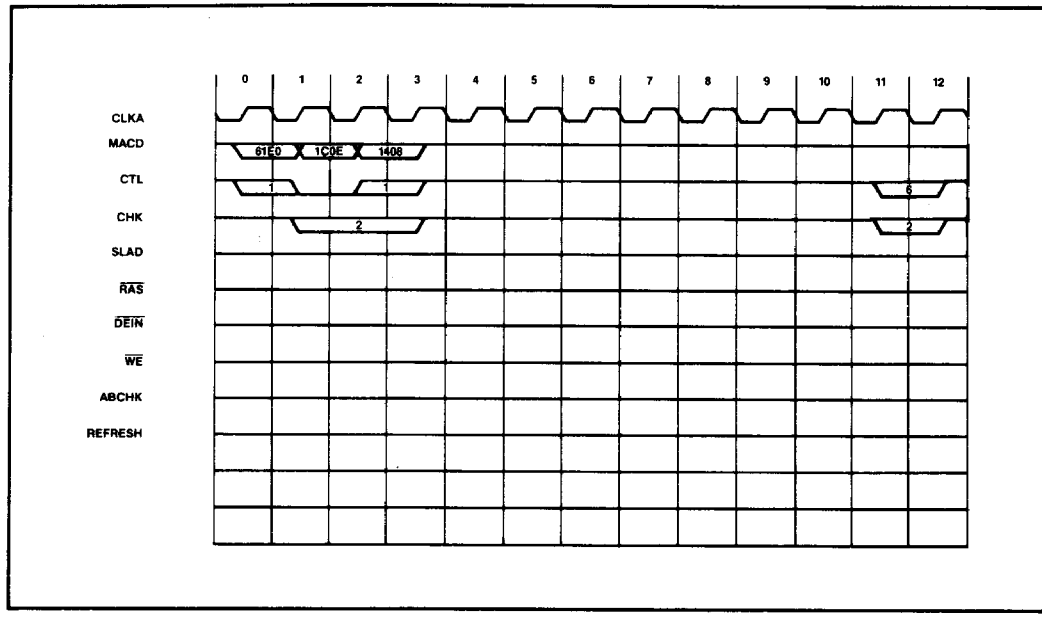


Figure 39. Interconnect Register Write

TEST DETECTION COMMAND

The Test Detection command is used to check that the detection circuits in the MCU are operating correctly. The command is invoked by an interconnect register write to the Test Detection register address (see the iAPX 432 Interconnect Architecture Reference Manual). The iAPX 432 instruction MOVE TO INTERCONNECT, executed on a GDP, causes the interconnect write cycle. The MCU reports the status of the test via the three-phase error report method described earlier. (See BERLOUT pin description.) If the test found no errors, the "no error" message is sent. If the test encountered an error, the "module error" message is sent.

Two Test Detection executions are shown in Figures 40 and 41, one each for the master and checker components in a FRC pair. Each of the diagrams consist of five parts. The two components operate identically except for the manner in which they check the BUSSEL signal (see cycles 10 and 11). The master operates the memory bus detection circuits with BUSSEL=0. The checker issues the complement, BUSSEL=1. External logic checks the two BUSSEL signals and reports the disagreement on the BCHKIN/M input.

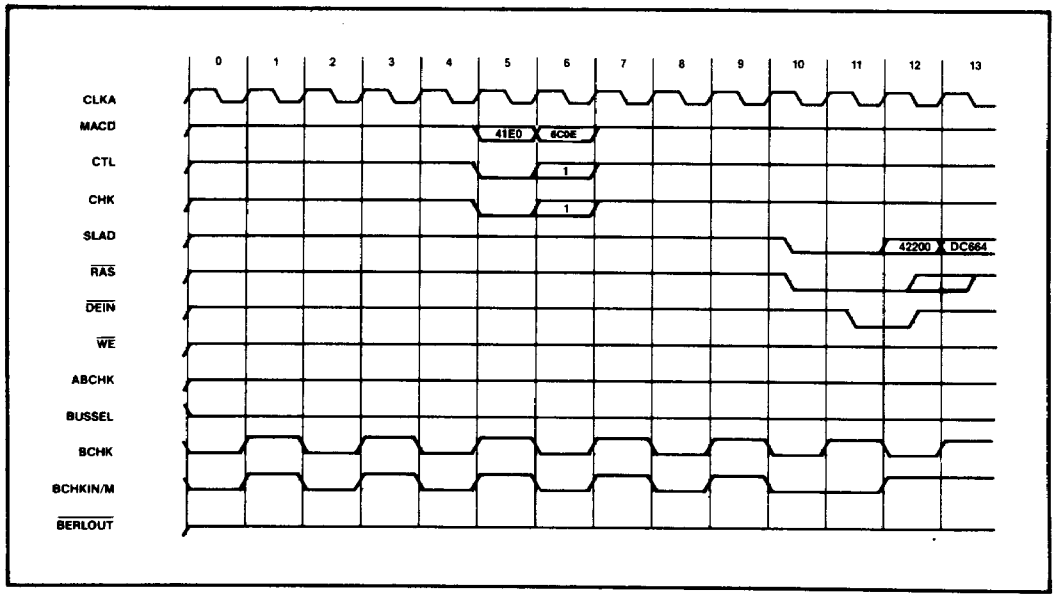


Figure 40. Test Detection—Master

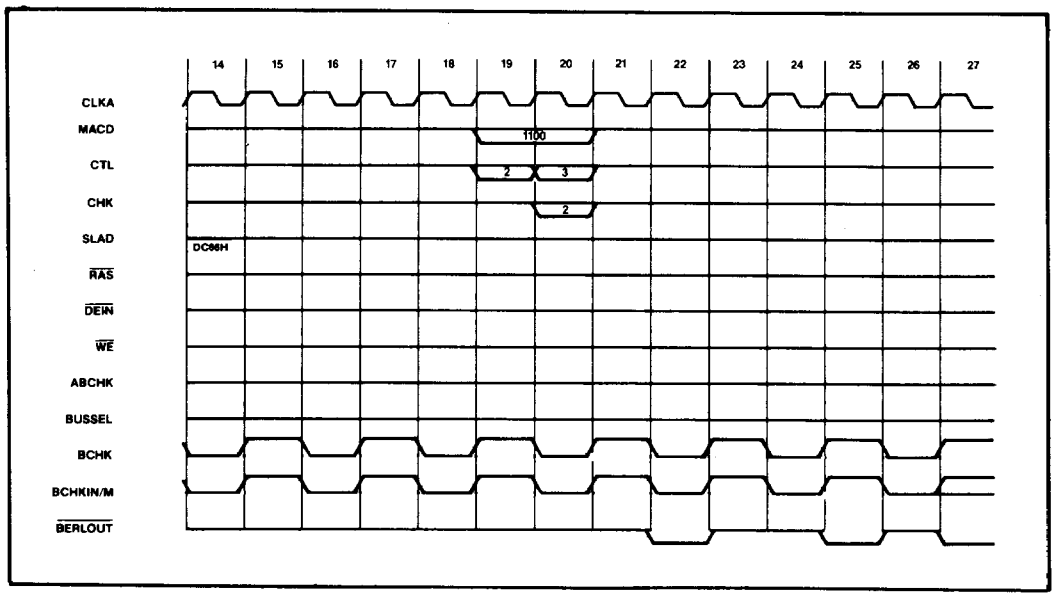


Figure 40. Test Detection—Master (continued)

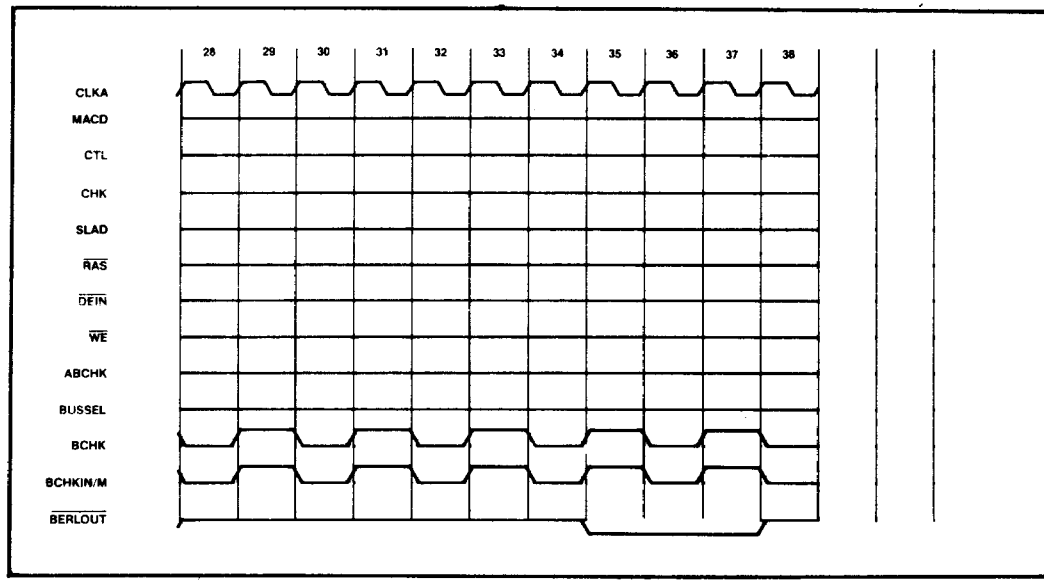


Figure 40. Test Detection—Master (continued)

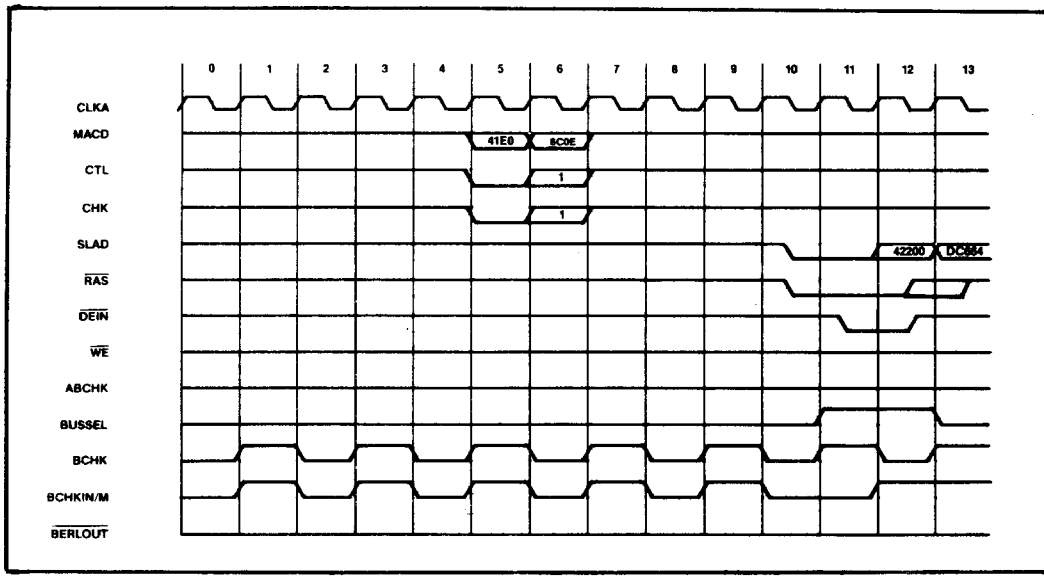


Figure 41. Test Detection—Checker

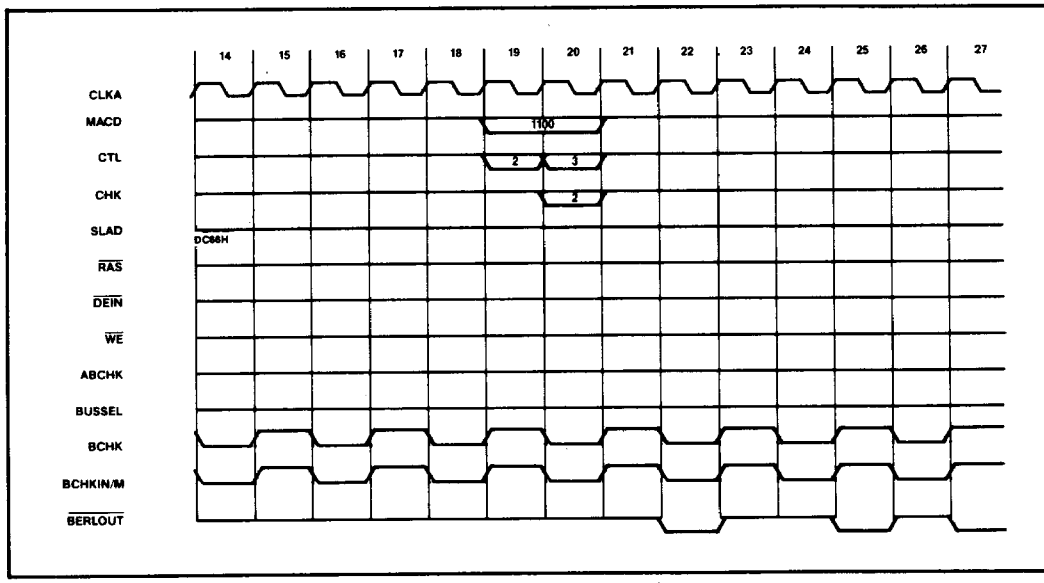


Figure 41. Test Detection—Checker (continued)

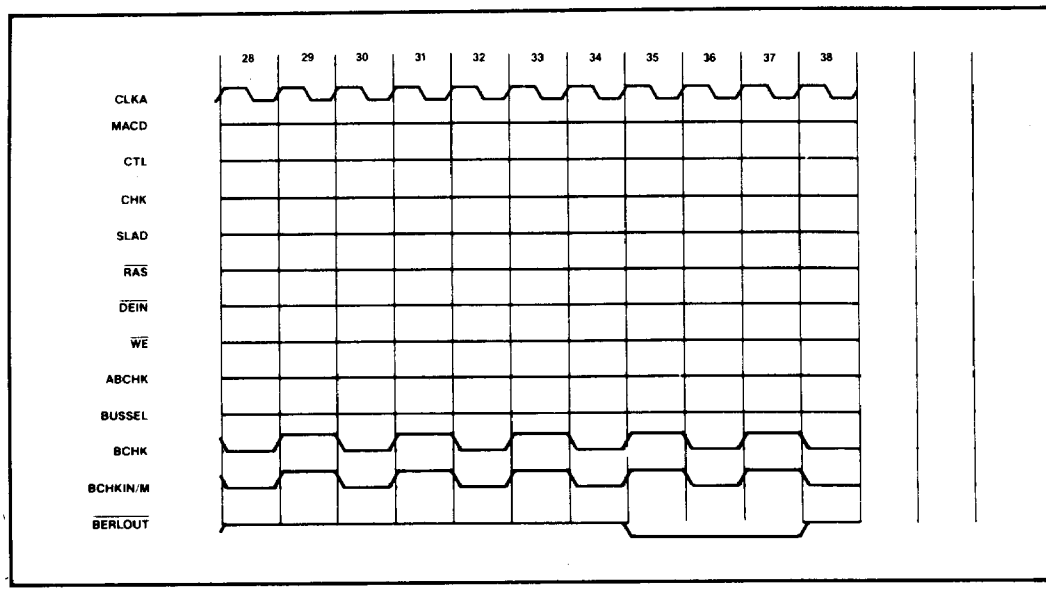


Figure 41. Test Detection—Checker (continued)

CLEAR MEMORY COMMAND

The Clear Memory command is invoked by an interconnect register write to the Clear Memory register. The MCU performs the command by writing zero data and appropriate ECC information to all storage array locations. The same hardware configuration that was used for memory read and write cycles is assumed. The MCU returns a reply to the requester as soon as it accepts the command but will not perform any further requests until it has cleared the memory. As seen in Figure 42, the MCU issues storage array write cycles beginning at the highest storage array address (in this case, 03FFFH) and decrements through remaining addresses (03FFEH, 03FFDH, . . .). Notice that the ECC and zero data information changes for each successive cycle (B000000000H, D200000000H, A600000000H, etc.). This occurs because ECC is computed across data *and* address information. This process continues until all storage locations have been cleared. Only the first few cycles of the process are demonstrated.

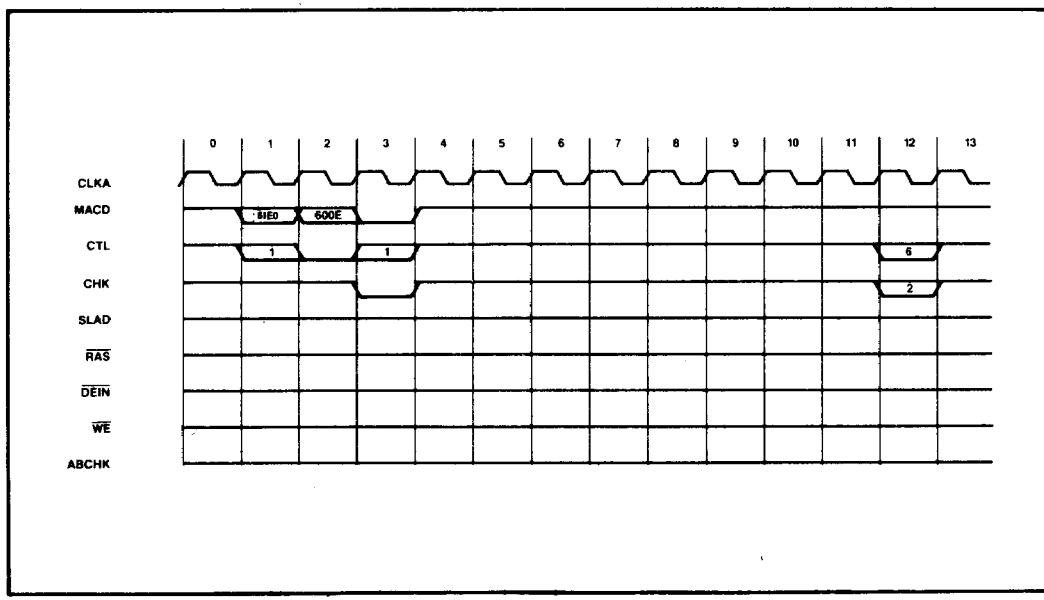


Figure 42. Clear Memory Command

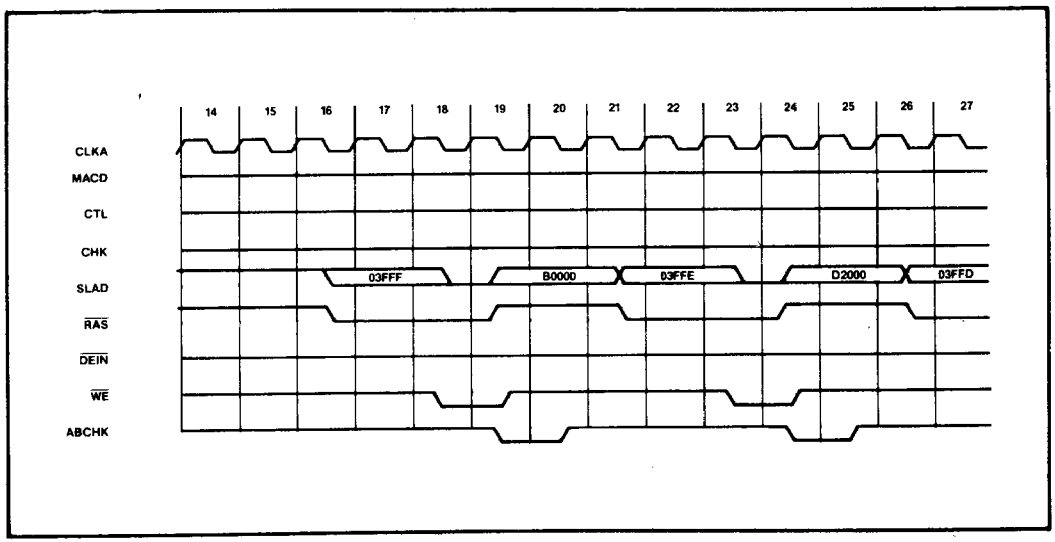


Figure 42. Clear Memory Command (continued)

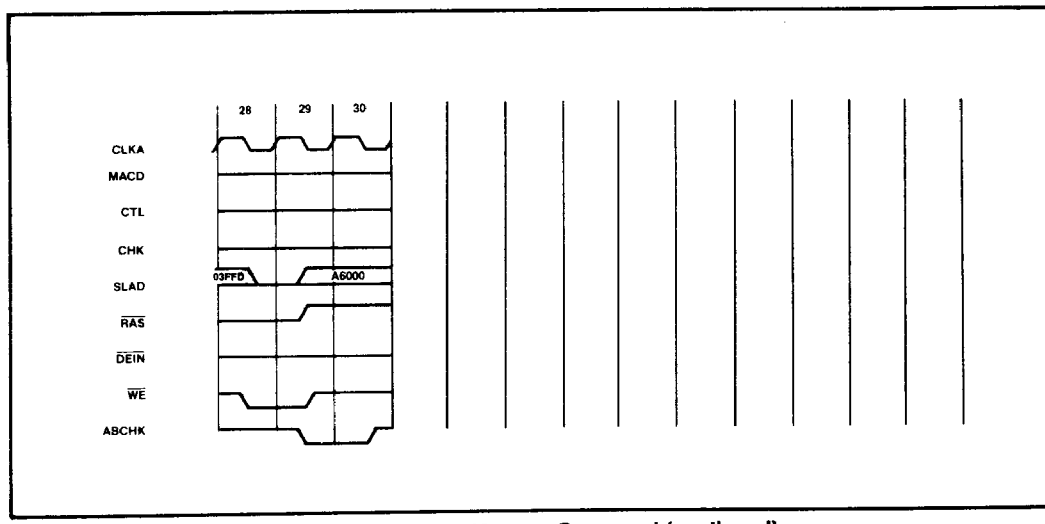


Figure 42. Clear Memory Command (continued)

Table 4. iAPX 43205 Clock Edge Table

Signal	Pin(s)	Input Sample		Output Drive	
		Clock	Edge	Clock	Edge
Inputs					
INIT	37	CLKA	Rising	n/a	
BERL1	38	CLKA	Rising	n/a	
BERL2	39	CLKA	Rising	n/a	
RQ	9	CLKA	Falling	n/a	
CONT	10	CLKA	Falling	n/a	
BCHKIN/M	4	CLKA	Rising	n/a	
ABCHK	45	CLKA	Rising	n/a	
Outputs					
RAS	48	n/a		CLKA	Falling
DEIN	47	n/a		CLKA	Falling
WE	46	n/a		CLKA	Falling
REFRESH	44	n/a		CLKA	Falling
BERLOUT	40	n/a		CLKA	Rising
BCHK	6	n/a		CLKB	Rising
BUSSEL	5	n/a		CLKA	Rising
Input/Output					
MACD15 . . . 0	13-28	CLKA	Rising	CLKB	Rising
CTL2 . . . 0	29-31	CLKA	Rising	CLKB	Rising
CHK1 . . . 0	11-12	CLKA	Rising	CLKB	Rising
MBOUT	32	CLKA	Rising	CLKA	Rising
SLAD19 . . . 0	1-2, 51-68	CLKA	Rising	CLKA	Falling

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Range ..... 0°C to 70°C
Storage Temperature .....−65°C to +150°C
Voltage on Any Pin with
Respect to Ground ..... −1.0V to +7V
Power Dissipation ..... 2.5 Watt

**NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

iAPX 43205 DC Characteristics

Symbol	Description	Min	Max	Units
Vilc	Clock input low voltage	−0.5	+0.8	V.
Vihc	Clock input high voltage	3.2	Vcc+0.5	V.
Vil	Input low voltage	−0.5	+0.8	V.
Vih	Input high voltage	2.0	VCC+0.5	V.
Vol	Output low voltage	—	0.45	V.
Voh	Output high voltage SLAD19..0 Other outputs	 2.6 2.4	 VCC VCC	 V. V.
Ili	Input leakage current (measured at Vin=VCC Volts)	—	±10	μA.
Ilo	Output leakage current (measured at 0.45 Volts ≤ Vout ≤ VCC Volts)	—	±10	μA.
Ioh	Output high current (measured at Vout=2.6 V.)	−2	—	mA.
Iol	Output low current (measured at Vout=0.45 V.)	4	—	mA.
Icc	Power supply current (sum of VCC0, VCC1, VCC2)	—	450	mA.

All DC parameters are guaranteed over the following conditions:

VSS2..VSS0 = 0 Volts
VCC2..VCC0 = 5.0 Volts ±10%
The absolute value of the differential DC voltage between any of the VCC pins (VCC2..VCC0) must be less than 0.1 Volts. This is normally guaranteed by connecting the three VCC pins to the same printed circuit power trace.

iAPX 43205 AC Characteristics

Ambient temperature range of 0°C to 70°C

Symbol	Description	5 MHz		7 MHz		8 MHz		Unit
		Min	Max	Min	Max	Min	Max	
t_r, t_f	Clock rise and fall times	-	13	-	11	-	10	nsec
t_1, t_2, t_3, t_4	Clock pulse width	37	250	25	250	24	250	nsec
t_{cy}	Clock cycle time	200	1000	143	1000	125	1000	nsec
t_{cd}	Clock to signal delay time	-	70	-	60	-	55	nsec
t_{dh}	Clock to signal hold time	20	-	17	-	15	-	nsec
t_{en}	Clock to signal output enable time	20	-	17	-	15	-	nsec
t_{df}	Clock to signal data float time	-	50	-	44	-	40	nsec
t_{dc}	Signal to clock setup time	30	-	26	-	24	-	nsec
t_{mc}	MACD input setup time	30	-	26	-	24	-	nsec
t_{ie}	Initialization period	40	100	40	100	40	100	t_{cy}

All AC parameters are guaranteed over the following conditions:

Ambient temperature range of 0 degrees Centigrade to 70 degrees Centigrade

VSS2 . . . VSS0 = 0 Volts
VCC2 . . . VCC0 = 5.0 Volts ± 10%

100 picoFarad external load capacitor on all output pins

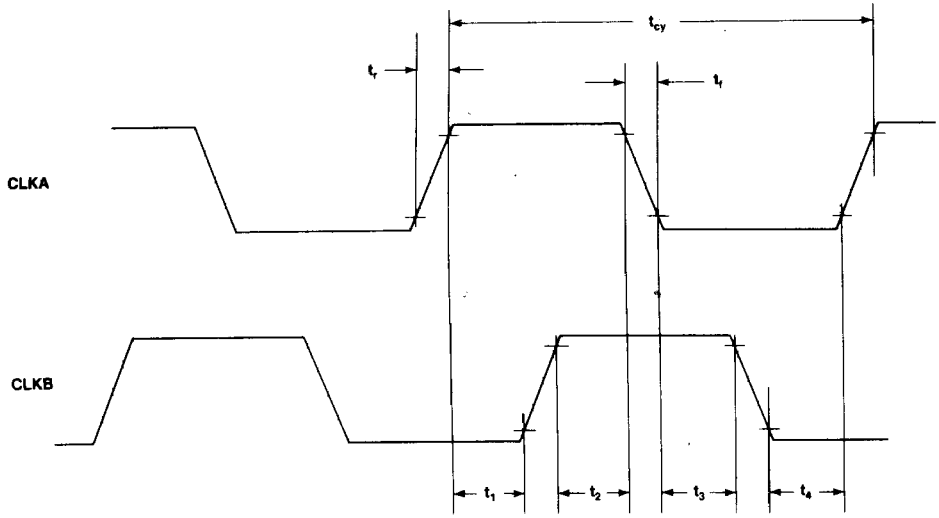
iAPX 43205 Capacitance Data

Conditions: Ta = 25°C
VCC = 5.0 Volts, GND = 0.0 Volts
f(test) = 1.0 MHz
Inputs held at 0.0 Volts
All outputs in high impedance state
All input/output pins are classified as outputs

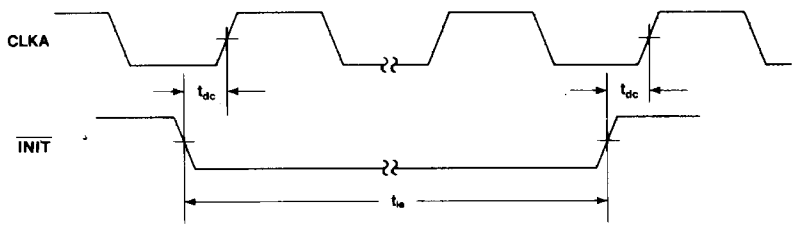
Symbol	Description	Max	Units
Cin	Input Capacitance	6	pF
Cout	Output Capacitance	12	pF

WAVEFORMS

Clock Input Timing Specification

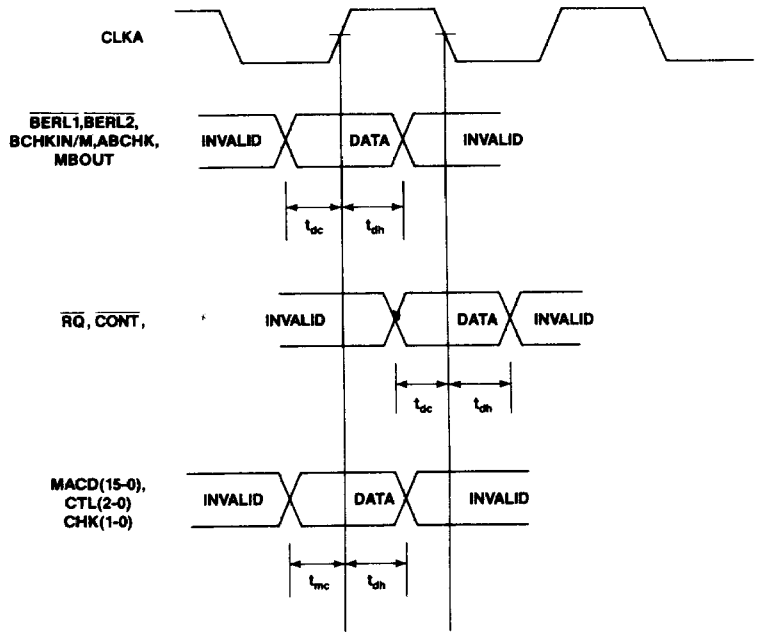


Initialization Timing Specification



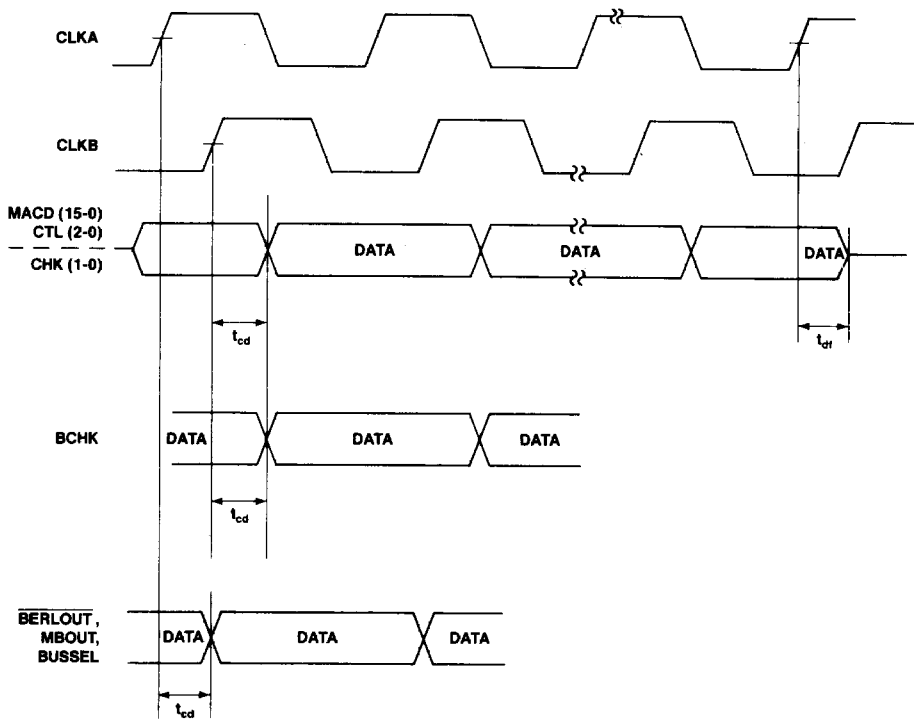
WAVEFORMS (Continued)

iAPX 43205 Input Timing Specification



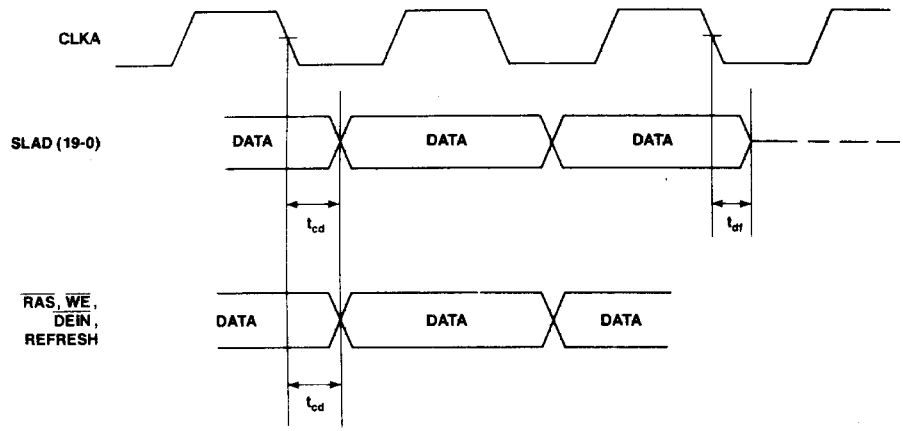
WAVEFORMS (Continued)

iAPX 43205 Output Timing Specification

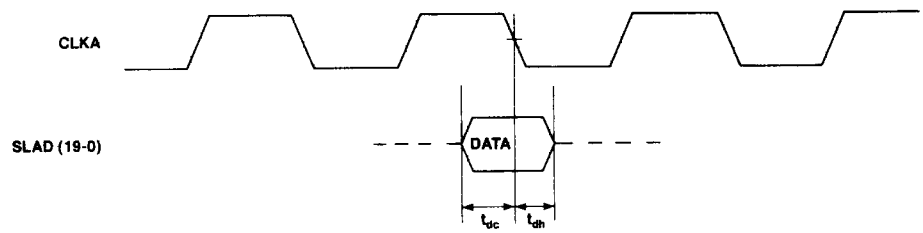


WAVEFORMS (Continued)

Storage Array Bus Output Timing Specification



Storage Array Bus Input Timing Specification



HARDWARE INTERFACING

This section presents examples of hardware that interfaces iAPX 432 processors (the iAPX 43201/43202 General Data Processor (GDP) and the iAPX 43203 Interface Processor (IP), the iAPX 43204 Bus Interface Unit (BIU), and the iAPX 43205 Memory Control Unit (MCU) to one another (see Figure 44). These examples present some alternatives for building iAPX 432 systems using the interconnect components. A wide variety of systems may be built with the interconnect components and this list of examples explores only a few dimensions of the design space.

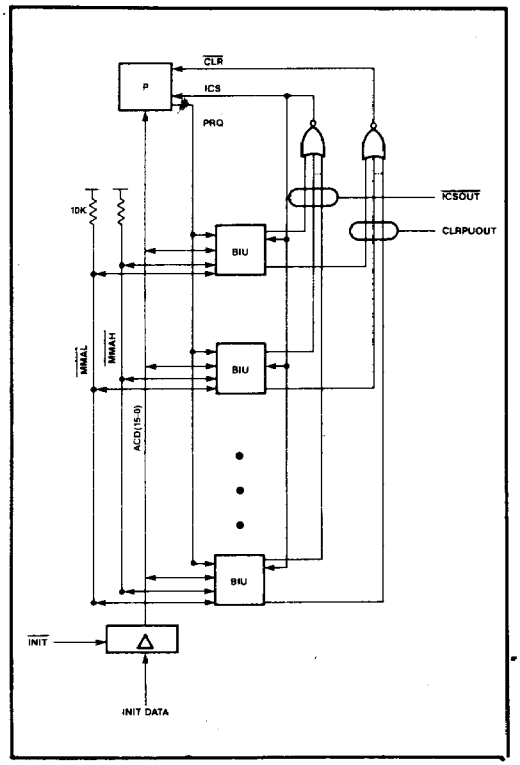


Figure 44. Interfacing iAPX 432 Processors to the BIU

Figure 45 illustrates the connection of a BIU to the serial error reporting networks. Here, the BIU connects to the module error report line (MERL) and has a duplicated bus error report line (BERL1 and BERL2). Each error report line is driven by open collector inverters so that the BIUs along either the module or bus axes may contribute error messages in wired-OR fashion. Two inverters are required to drive each error report line, one is a standard inverter and the other is an open collector version. The example shown also duplicates the bus error report lines so that bus error messages may be delivered even if one of the inverter paths fails. Should duplicated bus error report lines not be required, one of the two inverter groups could be deleted and the BERL1 and BERL2 signals connected together.

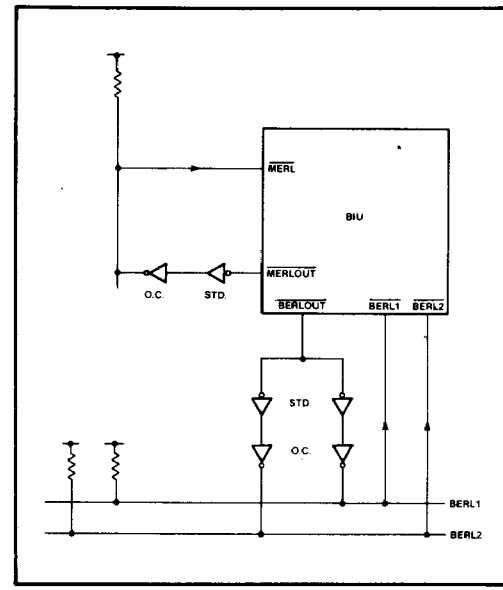


Figure 45. Interfacing a BIU to Error Reporting Network

The 432 hardware system designer must provide an external arbitration network which examines the BIU's arbitration signals, along with other BIUs in the system, to determine when it is permissible to use the memory bus. Two alternatives for this function are presented in Figures 46 and 47. The first method requires the fewest backplane signal lines since it utilizes multilevel *analog* signalling to arbitrate for the bus. The second method requires more backplane signals and unique wiring for each arbitrating unit but is fully *digital*.

With either method, each BIU produces two output signals, RQOUT and NREQOUT, which are activated to indicate that the BIU requires the use of the memory bus. Each BIU requires that an external logic network examine all the NREQOUTs and RQOUTs to identify when a request is being made by one or more BIUs. Three inputs to the BIU (NREQ, RQ and CONT) are generated by the external logic. NREQ (New Request) is activated to signal that a new time-ordering cycle has occurred. RQ (request) signifies that one or more RQOUTs are active and CONT (contention) signifies that more than one RQOUT is active.

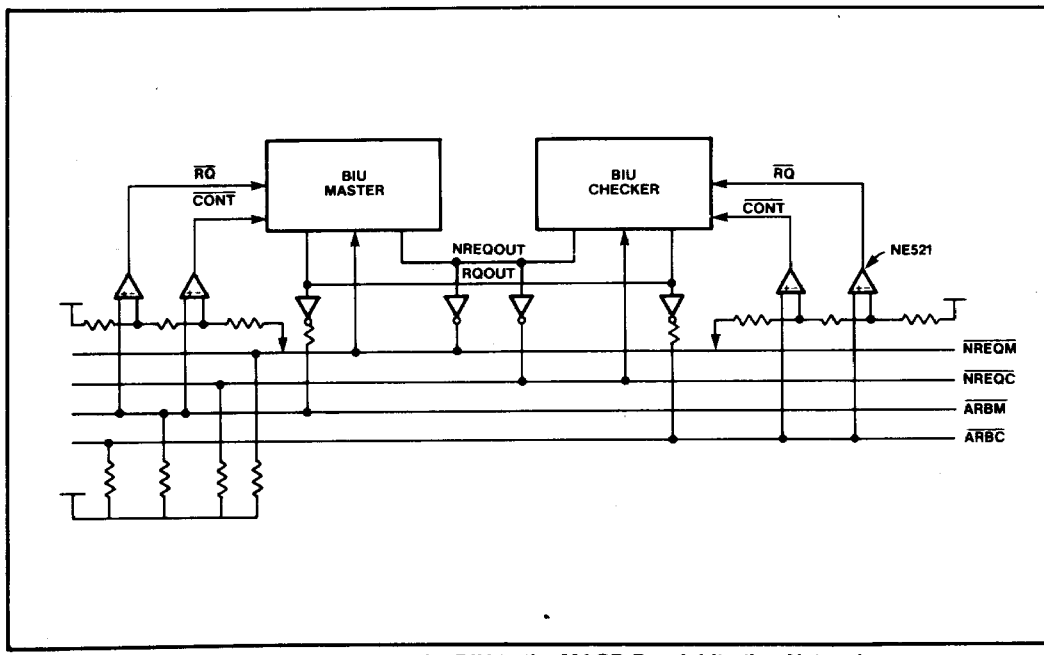


Figure 46. Interfacing the BIU to the MACD Bus Arbitration Network (Analog Method)

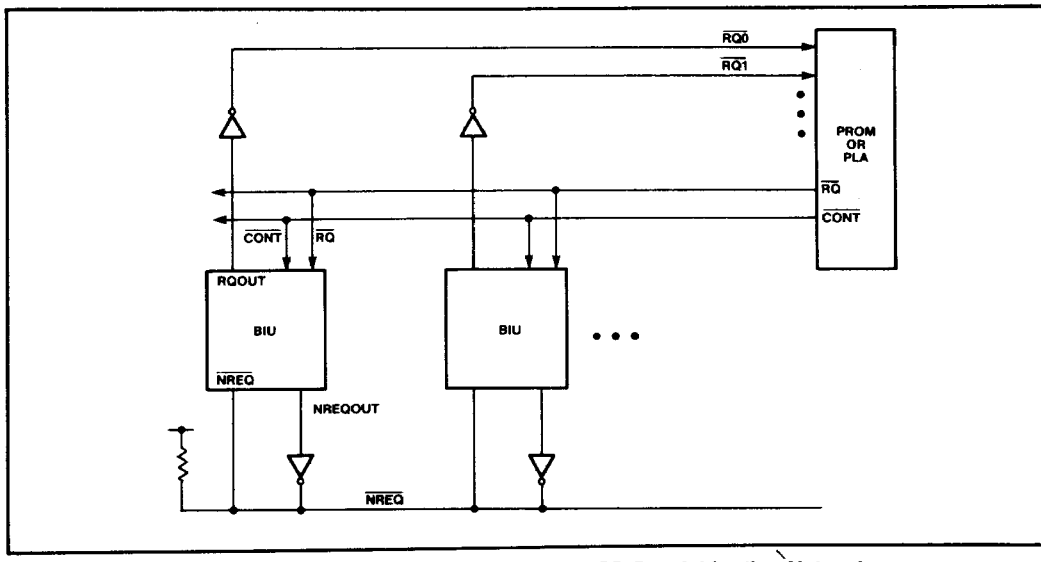


Figure 47. Interfacing the BIU to the MACD Bus Arbitration Network (Digital Method)

Figure 48 shows the required bus transceivers which connect a BIU to a memory bus. In addition, the drawing suggests how the oscillating BCHK signal may be used to check that the external buffers are operating correctly.

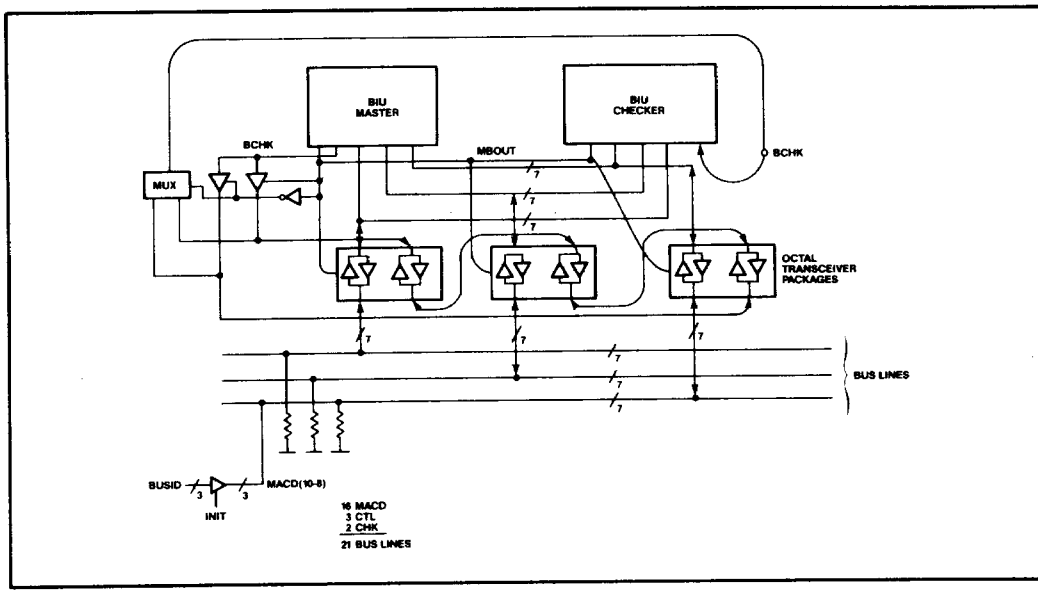


Figure 48. Interfacing the BIU to the MACD BUS

Figure 49 illustrates how an MCU, an external RAM storage array, and external sequencing logic form a memory subsystem. In this example, economical dynamic RAM components form the storage arrays and the array sequencing logic, under control of the MCU, provides the precise signals required to manage the arrays. Figures 50 and 51 detail the array sequencing logic and the timing of signals which coordinate the actions of the storage array.

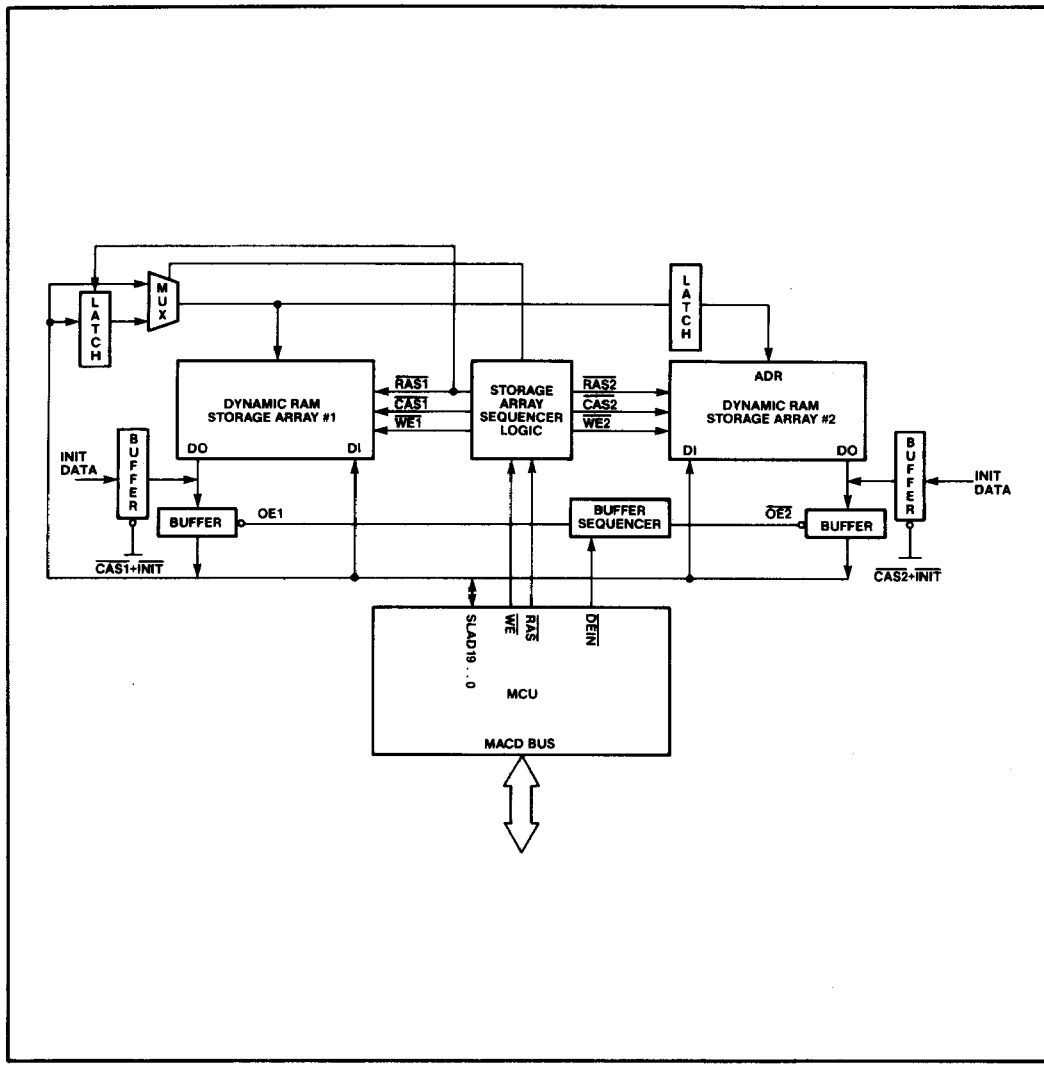


Figure 49. Interfacing the MCU to the Storage Array

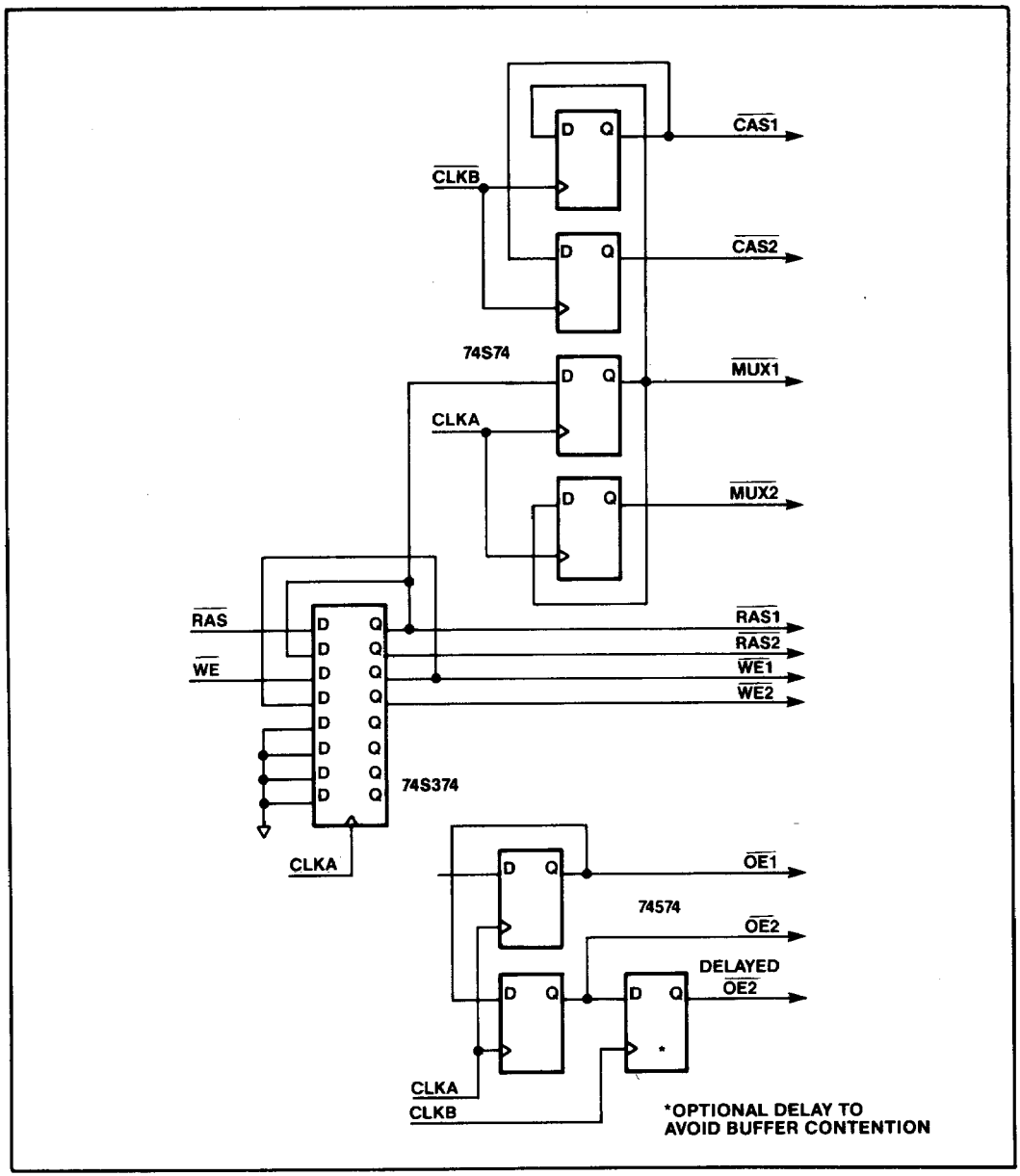


Figure 50. Sequencing Logic for the Storage Array

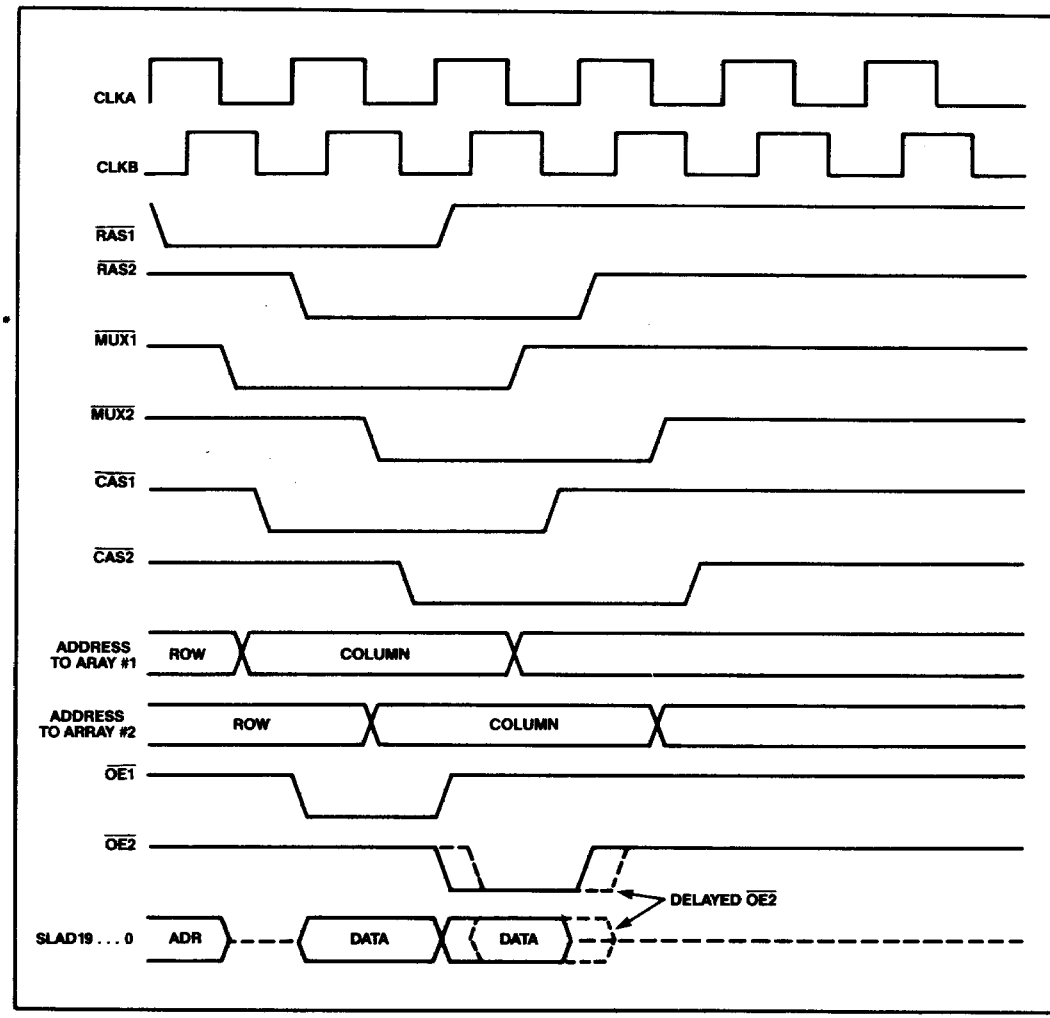


Figure 51. Timing Diagram for the Storage Array Interface

External hardware must be employed to permit an MCU to attach to its normal or backup memory bus. There are several facets to this requirement. Naturally, individual *bus transceivers* must be used to allow the MCU, through its BUSSEL (bus select) output signal, to choose which memory bus will carry its address, control, data, and check information. In addition, the MCU must access *error report* information and *arbitration* information from the correct memory bus. These requirements are met by the sort of network illustrated in Figure 52. Notice that these requirements are unique to the MCU. One BIU is required for *each* bus that a processor wishes to attach to, no such steering is required for a BIU. The next example highlights some of the considerations when extending these requirements to fault tolerant systems.

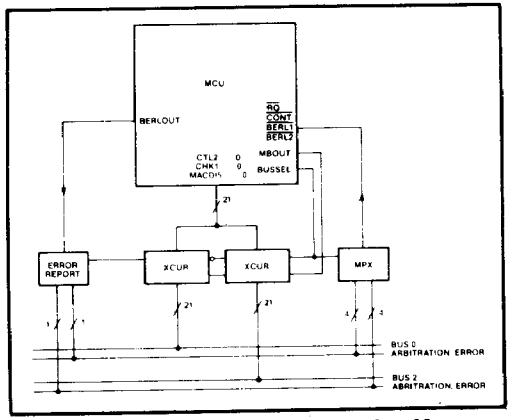


Figure 52. Interfacing the MCU to Two Memory Busses

Fault tolerant MCU configurations require that special external logic enable the transceivers which connect the MCU to its assigned memory bus. This may be done with a scheme illustrated in Figure 53. Two MCUs are employed in a FRC configuration. At their FRC interface, all the MACD, CTL, CHK signals as well as the MBOUT direction signal are compared for error. The master and checker MCUs each develop a version of the BUSSEL signal. Since it is not possible to FRC the BUSSEL signal and guarantee a correctly operating version, external hardware must be employed to develop a fault tolerant version of the bus selection function. The individual BUSSELs must be checked by external fault tolerant

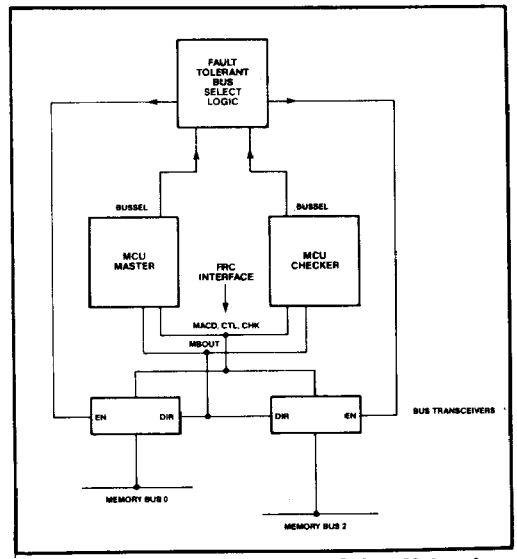


Figure 53. Fault Tolerant Bus Select Network

logic which enables only one of the two sets of memory bus transceivers when the BUSSELs are both active. If there is any discrepancy in the BUSSELs, the external logic must disable *both* of the bus transceiver sets so that the malfunctioning MCU cannot corrupt either memory bus.

This same technique must also be applied to other signals which must be routed to/from the currently assigned memory bus. The BERLOUT error report signal must only be output to the correctly selected bus. The $\overline{RQ}$, $\overline{CONT}$, $\overline{BERL1}$, and $\overline{BERL2}$ signals must only be received from the correctly selected bus.

Special logic is also required to check those signals which are not FRCed in a fault tolerant configuration. The BCHK output pins and the BCHKIN/M input pins of the master and checker MCUs may be used to detect errors in the external logic. In Figure 54, a PROM is used as the error detector. The inputs to the PROM may come from a variety of sources, depending on the particular hardware configuration. In this example, the PROM observes two sets of signals, one set from the master and another from

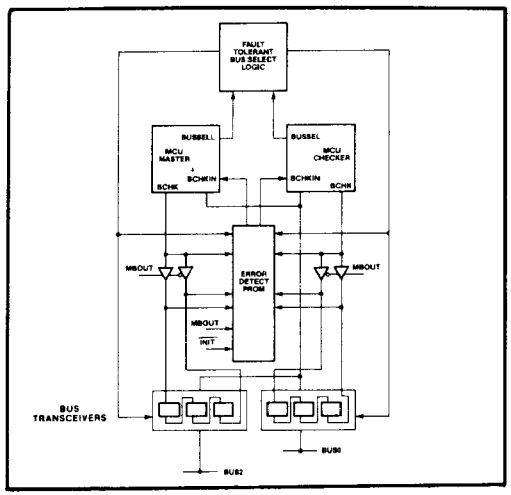


Figure 54. Detecting Errors in External MCU Logic

the checker. Notice that the oscillating BCHK signal is routed through the bus transceivers in different directions depending on the value of the MBOUT signal. The fault tolerant versions of the BUSSEL signal select collection of signals to be checked. The $\overline{\text{INIT}}$ input to the PROM provides a convenient way to establish master/checker roles during initialization since BCHKIN/M carries mastership information at that time.

PACKAGE

The 43204 and 43205 are packaged in 68-pin, leadless JEDEC type A hermetic chip carriers. Figure 55 illustrates the package, and Figures 9 and 11 show the pinouts.

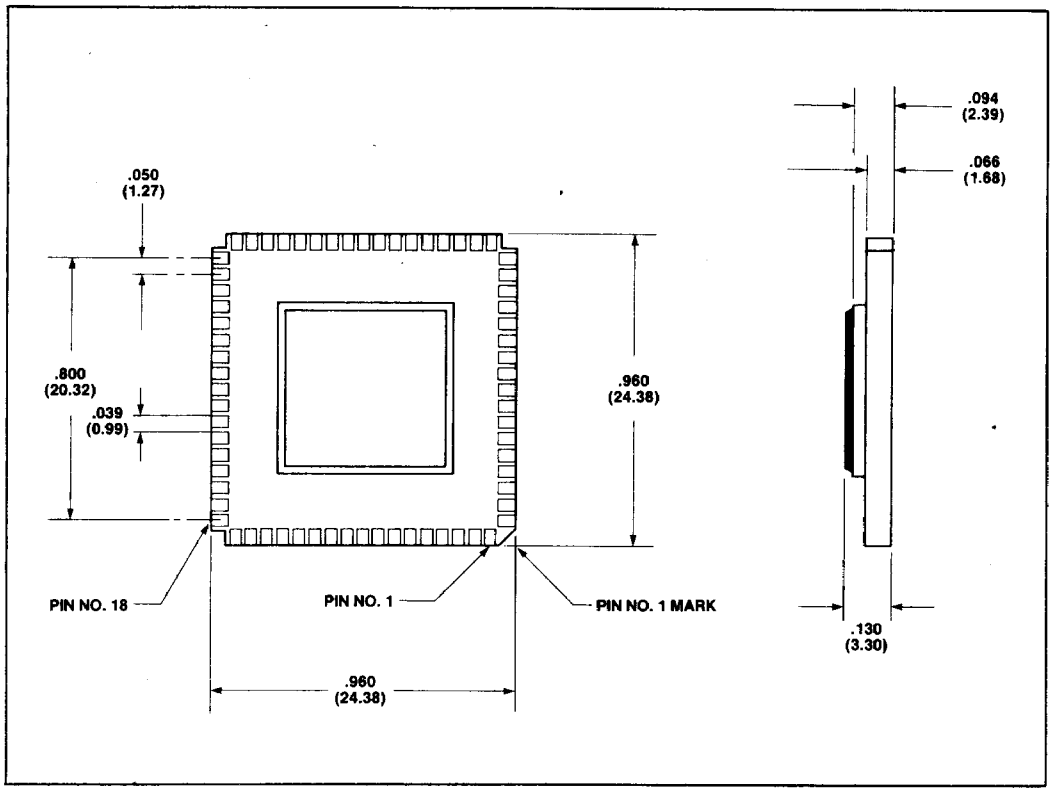


Figure 55. 43204 and 43205 JEDEC Type A Package