Arix Corporation
I/O Processor Module
Hardware Spec, ver 3.0

Curt Berg, 18 Nov 1988

CONTENTS

Document overview

This document specifies the IOPM hardware functions and operation. In addition it contains a cost estimate, and a preliminary MTBF calculation. The intended audience for this document is people within the company involved in the project.

Document history

Ver 0.1 First released draft

Ver 1.0 Addition : - Section about Console support
- Section about system wide addressing in respect to the IOPM

Changes : - System bus interface will check that responses come from the correct source, and the command buffer can no longer be over written by unsolicited commands on the System bus.

Ver 2.0 Addition : - Device bus functions specified

Changes : - Memory cycle will be 250nS, 100 nS burst cycles
- Memory uses page mode devices to simulate nibble mode
- Local bus cycle size will not be checked against port size
- Write accelerator can store two read commands

Ver 2.1 Addition : - Device bus signal description

Changes : - CIO deleted

Ver 2.2 Changes : - Free running clock added

Ver 3.0 Updated for release. No major changes

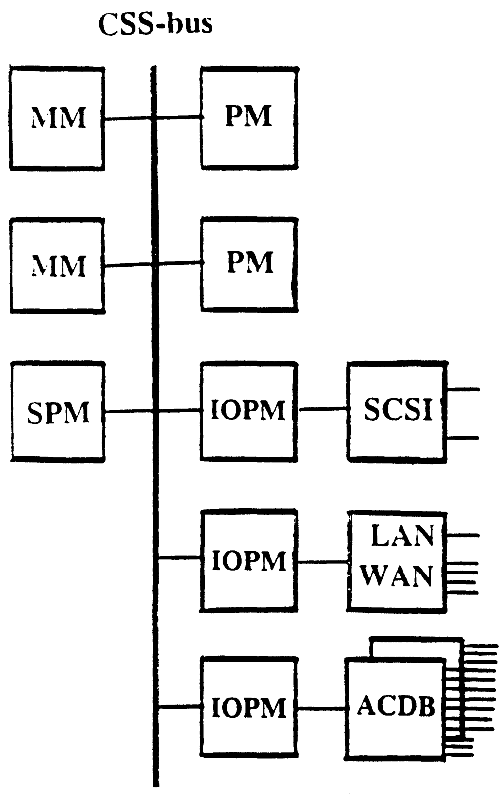
Related documents :
System 3000 Hardware Description, 4 May 1987
Generic I/O Controller Proposal, ver 2.0, Sep 17 1987
Motorola 68030 Users Manual

1. Introduction

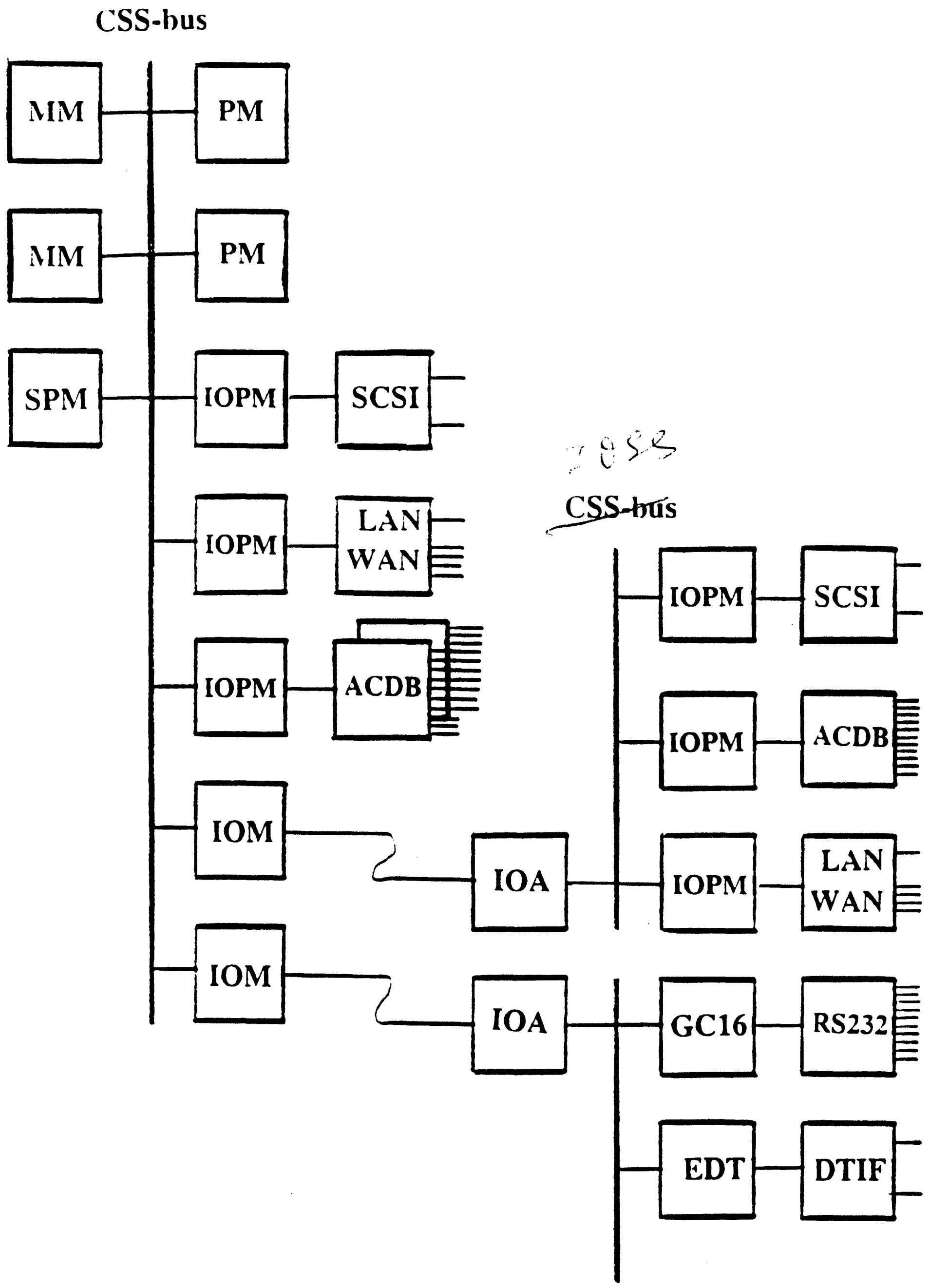
The I/O Processor Module (IOPM) is a high performance controller for the Arix System 90 based on a 68030 micro-processor. The controller is generic in the sense that it is a general purpose controller where specific device interfaces are located on separate device boards. The IOPM is intended to support at least three classes of device boards: 1. Communication device boards, 2. Disk/Tape device boards, 3. Networking device boards. The IOPM interfaces to the System bus and can be installed in either the main card cage or in an extension card cage. See figure 1.

The IOPM can also be configured to a Console mode whereby it will have additional privileges and capabilities on the System bus. A minimum set of features have been added to the IOPM so that in the future a device board can be designed that can serve both as service processor and device board in a small system.

The IOPM is will be designed for auto insertion. The board will have 3 connectors : 1. 150 pin female System bus connector, 2. 80 pin female Arbiter connector, 3. 96 pin male DIN device board interface connector.



SYSTEM 90 BLOCK DIAGRAM
with IOPM based I/O sub-system



SYSTEM 90 BLOCK DIAGRAM
with IOPM based I/O sub-system

2. IOPM hardware description overview

This section introduces the architecture of the IOPM and describes the concept and function of the IOPM.

The main sections of the IOPM are 20 MHz 68030 processor, 1 MByte of parity protected DRAM, System bus interface, 32-bit local bus, and a device board interface. (See Fig 2.) Other features include, diagnostic EPROM, Interrupt control, a 16 bit free running counter, status LED's and Control/Status registers. The architecture is notable for it's simplicity, the high performance processor, and the minimal amount of device specific functions. The device board interface is a general purpose interface which supports DMA to local memory, and DMA to main memory via the System bus interface.

When an IOPM is installed into a system cabinet some thought must be given to the priority the particular controller will have. If device board attached to an IOPM performs high speed DMA operation to/from the system bus (via the IOPM local bus) it should be configured as a high priority module on the system bus.

IOPM Block Diagram

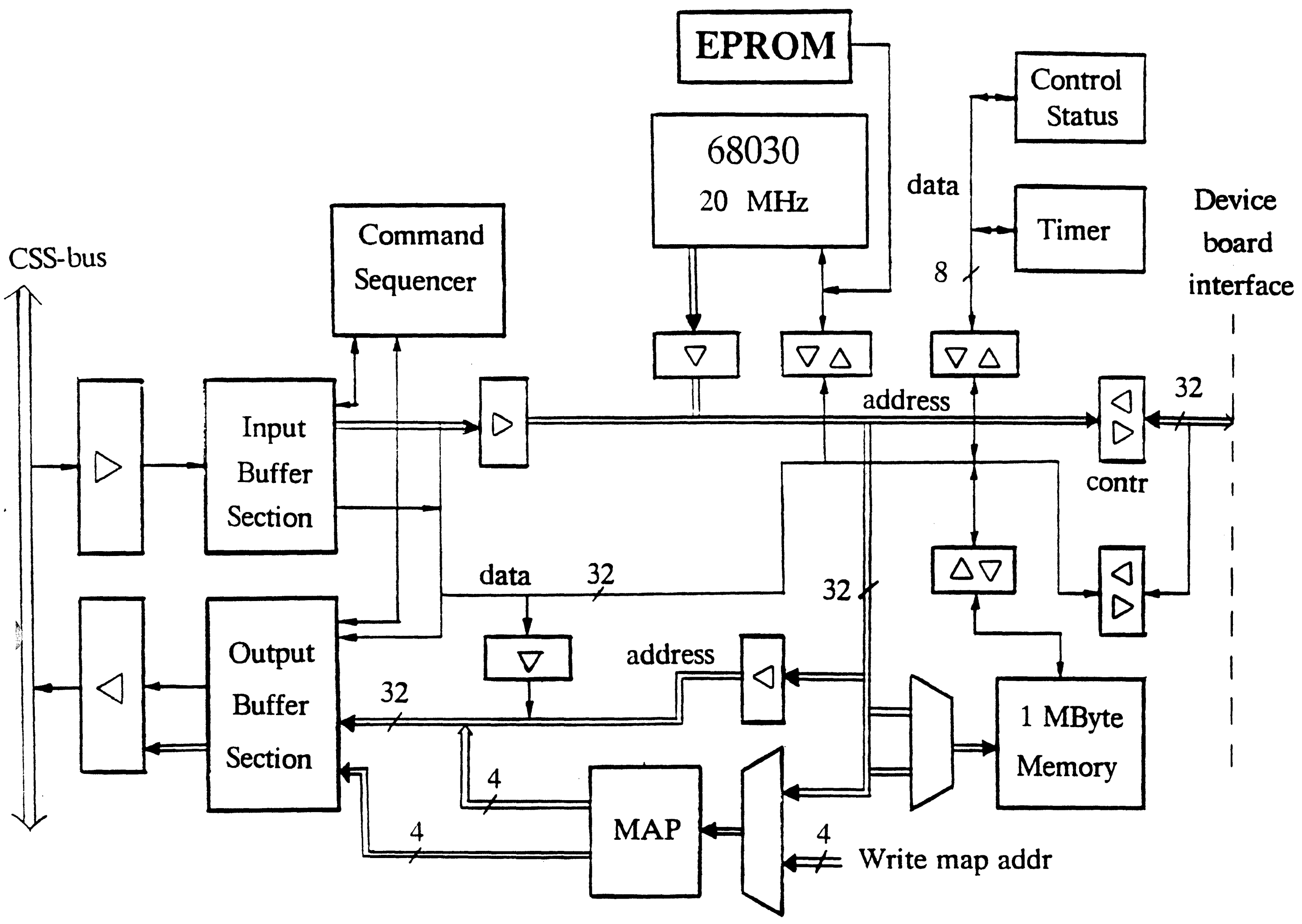


Fig. 1

2.1 System bus interface section

The System bus interface section connects the slower non-multiplexed 32-bit local bus with the faster time multiplexed 64 bit System bus. This interface has a 15 entry address translation map to translate 32 bit local addresses to 36-bit System bus addresses. From the System bus side the interface accepts 1,2,3 or 4 byte write commands, and 1,2,3,4,8 or 16 byte read commands (8 and 16 byte reads commands must only be sent to targets on the local bus that support burst read operation, e.g. local memory).

From the IOPM bus side the System bus interface has a two deep command buffer ("write accelerator",) necessary to accommodate high data transfer rates. When the System bus interface sends a read command over the System bus, write commands will still be sent while the System bus interface waits for the responses for the read command.

The System bus output interface section can send responses over the System bus to an initiator even though a command request is pending (command back-off). It is essential that responses to incoming read commands are sent with higher priority than commands to avoid deadlock in the system.

The System bus interface supports 68030 atomic operation (system wide locking) with the condition that the 68030's first bus cycle of the atomic operation is an operation in the System bus address range.

The IOPM System 90 board ID will be 84 Hex.

2.2 Processor

The IOPM has a 20 MHz 68030 which supports burst fill of it's internal data and instruction caches. The 68030's internal caches helps reduce local bus traffic, thus performance will not significantly degrade when part of the local bus bandwidth is used for DMA. Burst filling of the internal data cache will also greatly improve performance when moving large blocks of data, and reading data from the stack. The processor also has an internal MMU that will give some system protection, and simplify system wide addressing. The processor and will run synchronously with the 20 MHz System bus clock.

2.3 IOPM local bus

The IOPM local bus connects the System bus interface, the 68030, local memory, and the device bus interface. At any time, either the 68030, the System bus interface (also called Command Sequencer), or the device interface is bus master, and thereby has permission to perform local bus operations. The device bus interface has highest priority, and is guaranteed a maximum DMA latency of about 750 nS. Various potential deadlock situation exists. To avoid poor performance and deadlock situations a bus master can be suspended when performing a bus operation for lack of a resource being available. The suspended bus cycle will not continue until the requested resource becomes available.

The local bus supports various transfer sizes including burst read and burst write operations (the device board is the only bus master which can initiate a burst write cycle). Dynamic bus sizing is not supported over the System bus and consequently not on the local bus. (The only exception is when the IOPM's 68030 accesses the diag bag or local EPROM which are on the 68030 bus not the IOPM local bus).

2.4 Memory, refresh

The memory is 1MByte, byte parity protected, organized 256 kbit x 32. When 4 Mbit DRAMs become available the RAM array can be 4 MByte, organized 1 Mbit x 32. The memory will support burst read/write (nibble mode operation) for filling the 68030's internal caches and for high speed DMA's. A memory cycle will be 250 nS, and following memory cycles during a burst will be 100 nS/cycle. A 250 nS refresh cycle will be performed every 12.8 uS and will take 3 % of the memory bandwidth.

2.5 Device board interface

The device board interface is a synchronous 32 bit address/data multiplexed interface and has many similarities with the 68030 processor bus. It will support various size operations including 16 byte burst reads and writes. Device boards can elect to DMA data to local or main memory. The device boards can read and write the IOPM local memory with the same cycle time as the 68030. The device board interface has three interrupt control lines, which by it can interrupt the 68030 at three interrupt levels including NMI.

2.6 EPROM, Control/Status registers

The IOPM will have 8/16/32kByte diagnostic EPROM. The EPROM can only be read by the 68030. There will be Control/Status registers for various control and monitor functions and 10 status LED's.

3. System wide addressing

The IOPM will support a 36 bit system wide addressing scheme, where the upper four bits A35-A32 of address can be considered physical address in the main card cage. The next four bits A31-A28 can be considered physical address in an extension card cage if the card cage address A35-A32 is addressing an IOM connected to a System bus expansion card cage. If a System bus address is not addressing an IOM, address bits A31-A28 are considered regular board offset. The board offset A31-A28 is ignored by the IOPM, so the system can address F000_0000 - FFFF_FFFF of the IOPM address map on the IOPM local bus.

4. System bus interface

This section is a detailed description of the various blocks contained in the System bus interface section.

The bus interface on the IOPM will comply with the System bus spec rev 2.5.

4.1 System bus interface overview

The main sections of the System bus interface are the input buffer section, the output buffer section, the command sequencer, and the address translation map.

The input buffer section monitors the System bus activity and latches error free transmissions (commands and responses) sent to the slot the IOPM is installed in. It checks incoming commands and responses for errors and sends ACKs and NACKs accordingly. The input buffer section also maintains the Ready Counter, and executes write control commands.

The output buffer section stores outgoing commands and responses, and sends them according to the System bus protocol. It ensures that responses are sent with higher priority than commands to avoid system deadlock situations. In addition it acquires and maintains the system lock for the 68030. The lock is released when the atomic operation completed by the 68030 and all entries in the output buffer associated with this locked operation are flushed.

The command sequencer executes the incoming read and write commands from the system bus and if necessary stores responses in the output buffer to be sent back to the initiator of the command. It takes commands from the 68030 or a device board addressed to the System bus, sends them via the output buffer section, and ensures that responses gets back to the initiator.

The address translation map translates IOPM local 32 bit addresses to 36 bit System bus addresses for commands to the system bus. This map has 15 entries and will normally only be programmed once during system boot.

4.2 Supported System bus commands

The System bus interface section will support the following incoming commands.

Type field bit	command
543210P	(P is even Parity bit)
0000000	Error response
0000101	Error data response
0000110	Error data 1 response
0001001	Data response
0110000	Read 4 bytes
0110011	Read 1 byte
0110101	Read 2 bytes
0110110	Read 3 bytes
0111001	Read 8 bytes
0111010	Read 16 bytes
1010000	Write 4 bytes
1010011	Write 1 byte
1010101	Write 2 bytes
1010110	Write 3 bytes
1100101	Module disable
1100110	Module enable
1101001	Interface disable
1101010	Interface enable

Notice that 32 byte read commands are not supported by the IOPM. The IOPM will return an NACK if sent a non supported read command. An incoming 8 or 16 byte read command must address a target that supports burst reads or an error response will be returned. Generally size checking is not performed on the local IOPM bus.

The offset range FFFF_FF00 to FFFF_FFFF of all System bus modules contains control and status registers. The only registers supported in this address range by the IOPM is the write only interrupt register and the board ID. The ID can be read with a byte read at location FFFF_FFFF or a long word read at location FFFF_FFFC (ID is when fourth byte in the long word).

4.3 Input buffer section

The input buffer section consists of a receive synchronizer, a command decoder, a command source address latch, a Ready Counter, and an input pipeline register.

4.3.1 Receive synchronizer

The receive synchronizer isolates and synchronizes all signals coming from the System bus. All signals on the System bus are received with 74F374s. Signals from arbiter are not synchronized before used.

4.3.2 Command decode and source latch

The command decoder decodes incoming commands and responses when the ACTIVE signal is asserted, the destination address correct, and matches the slot address of this IOPM. If the incoming command is a control write and no error is detected then the command is performed within two clock cycles by the command decoder, an ACK is sent, and the Ready Counter is incremented. If the command decoded is a response and no error was detected, then the data and type are latched in the pipeline register, and an ACK is sent. If the transfer was a command and no error was detected, the data, type, and source address is latched, and an ACK is sent. If any error is detected on incoming commands or responses a NACK is sent. If an error is detected on an incoming command or the Ready Counter was decremented without a valid command received, then the Ready Counter is immediately incremented. When a valid command is received the command decoder will not accept an other incoming commands (except control writes) until the Ready Counter is incremented by the Command Sequencer, indicating that the latched command has been executed. If the input section ever sends a NACK this error condition is latched in the error status register, and an NMI to the 68030 is generated.

The command decoder also checks responses to make sure that they were sent from the correct source. If not, the response will be NACKed. This section also maintains count of how much valid information is in the input pipeline register. The count information is used by the Command Sequencer to execute commands or return responses. The command sequencer is normally responsible for decrementing this counters. A bit in a control register can be set to flush the information in the input and output pipeline buffers.

4.3.3 Input pipeline register

The input pipeline register (9 x 29C520) stores incoming commands and responses. It can store one command and one response (response can be two partial responses). If the command is a read or write command the pipeline register stores the command data, type, and the source address. The type is used by the command sequencer to determine what type command or response was received. The source address is used as destination address when storing responses in the output pipeline. The data in the input pipeline registers can be read by diagnostics as 32 bit I/O registers.

4.3.4 Ready Counter

The Ready Counter indicates to the arbiter that the IOPM is ready to receive a single command. The Ready Counter is implemented with a single JK-flop, which initially is set on power up or board reset. The arbiter clears this bit when it grants someone on the System bus permission to send a command to this IOPM's. If the Ready Counter is decremented and the command decoder doesn't decode a valid command or it was a control write command the Ready counter will immediately be increment. The Ready Counter is normally incremented by the command sequencer when it has executed the command latched in the input pipeline register. A command is considered executed when the response is latched in the output buffer section, or if a write command when the write is acknowledged on the local bus.

4.3.5 Input section System bus protocol

During the first System bus clock cycle of an input command or response the data, type, source, destination and ACTIVE signal are driven by the source on the System bus. If the transmission is a command, the Ready Counter of the destination is decremented by the System bus arbiter during this cycle. During the second clock cycle the command is decoded and checked for errors. If no errors where detected then at the trailing edge of the second clock cycle the data, type, and source address (if read or write command) is latched. During the third clock cycle an ACK or NACK is sent, and if the Ready Counter was decremented and no valid read or write command was latched the Ready Counter is incremented.

4.3.6 Control Writes and System bus Reset

The MODULE ENABLE and the INTERFACE ENABLE bits in the input buffer section will be asserted at power-up time or IOPM board reset if the IOPM is in Console mode. If not in Console mode theses bits will instead be cleared. Incoming control writes, can at any time assert or negate the MODULE and INTERFACE ENABLE bits. An IOPM in Console mode send control writes over the System bus.

A System bus reset will cause a board reset when not in Console mode. When in Console mode the IOPM can instead generate a System bus resets by setting a bit in the Configuration Control Register.

4.4 Output buffer section

The output buffer section consists of an output pipeline register a System bus sequencer, and output buffers.

4.4.1 Output pipeline register

The output pipeline register stores any two outgoing commands and up to 16 bytes of response data. Outgoing commands contains, command type, command data (32 bit address and 32 bit data), destination addresses, 2 bits of local address , and a command initiator bit. The 2 bit of local address, the command type and the initiator bit is passed to the Command Sequencer when a read command is sent on the system bus. The Command Sequencer uses this information when later returning responses to the initiator of the command.

When the Command Sequencer executes read commands responses are put in the output pipeline. If responses are <= 4 bytes the Command Sequencer fills 8 bytes of "data" into the pipeline register at the time. Both "halves" of the output pipeline register are then filled with the same data. 8 or 16 byte responses (burst read) fills 4 bytes of data at the time into the pipeline register. If a bus error happens during a read burst operation, the burst will stop, and an error response will be sent.

4.4.2 Output buffers

The output buffer synchronizes the outputs from the output pipeline register, generates error checking signals, and drives the System bus using 74F241 bus drivers. The bus can only be driven after the INTERFACE_ENABLE bit is asserted. The bus is only driven when permission is granted by the arbiter.

4.4.3 System bus sequencer

The System bus sequencer controls the sending of commands and responses stored in the pipeline register. It maintains count of the content currently in the pipeline register, and keeps information about if the content is part of an atomic sequence. The sequencer handles atomic sequences by acquiring the system lock and ensure that all parts of an atomic operation are flushed from the pipeline before releasing the system lock. When a read command is send over the System bus, information about the expected responses are also sent to the Command Sequencer. The command sequencer uses this information to deliver the responses in the correct order and to the appropriate initiator.

The bus sequencer will send commands and responses over the System bus according to the system bus specification. The bus sequencer will always attempt to send responses with higher priority than commands. A pending command request will be displaced by a response request as soon as a response is entered into the pipeline (command back off). Commands can be send every forth clock cycle, and response every third clock cycle.

The bus sequencer monitors a wait response bit, to ensure that an other read command is not sent over the System bus until the response from previous read command is received and delivered. This ensures that the input response buffer in the input pipeline is not over written. The bus sequencer is capable of sending write commands when it is waiting for responses from the System bus.

If the bus sequencer sends a command over the System bus and no ACK or a NACK is received, then this error condition is latched in the error status register and an NMI to the 68030 is generated. Through one of the status registers the 68030 can also check if any commands are stuck in the pipeline register.

If the board is in an extension cabinet the command sequencer will assert BUS_IOX_ACTIVE instead of ACTIVE when driving the System bus, except in loop back mode. In loop back mode an IOPM can send transmissions to itself by asserting ACTIVE instead of BUS_XMIT_STROBE.

4.5 Command sequencer

The command sequencer has two main functions: 1. Execute incoming commands from the System bus and if necessary send responses to the initiator, 2. Monitor the local bus and transfer commands in the System bus address space from the local bus on to the System bus and if necessary return responses to the initiator. The command sequences can for diagnostic purposes be disabled from executing incoming commands.

Commands can be received by the input buffer section when the Ready Counter indicates ready. The IOPM local bus arbiter ensures that a command in the input pipeline, will not be executed by the command sequencer until the output pipeline response section is empty ensuring space for a possible response.

4.5.1 Incoming write commands

The IOPM local bus arbiter gives the Command Sequencer access to the local bus when a command is detected in the input buffer and the local bus becomes available. When the command is executed (operation acknowledged) the Ready Counters is incremented.

4.5.2 Incoming read commands and out going responses

After the command sequencer has acquired the local bus it will run a local bus read cycle. If the command is an 8 or 16 byte read command a burst read cycle will be performed. If a bus error is detected a response of type error will be returned. If no error is detected responses of type data will be sent. Incoming 8 and 16 byte read commands are always 8 and 16 byte aligned. 8 byte read commands are performed as burst cycles on the local bus but only the first eight bytes received are stored in the output response buffer. The ready counter will be incremented when all responses are latched in the output pipeline.

4.6 Address translation map

The Address translation map translates addresses for all local bus read and write operation directed to the System bus. The address translation map contains 15 entries, and translates local 32 bit addresses to a 36 bit System bus addresses. The map is a 16 byte RAM where bit 28 - 31 of the local address represent the address to the map RAM (map entry hex F is not used). The 8 bit content of the RAM represent the mapped System bus address bit 28 - 31 and the 4 bit destination address. The map entries can be read and written in the local I/O address space.

4.6.1 Outgoing write commands

Write commands on the local bus targeted for the System bus are latched in the output pipeline register if the pipeline is not already full. If full the initiator of the write command can be suspended, freeing up the local bus for other operations. Latched in the pipeline during a write is the local bus data is, the mapped address, the destination address, and the command type.

When the CSS_CONT_WR bit in the control register is set and the IOPM is in Console mode the command sequencer will generate control write commands instead of write commands to the System bus. The address translation map is still used to generate the destination address. The control commands are encoded according to the following table:

Operation	will generate
Write word to odd address	Module disable
Write word to even address	Module enable
Write byte to odd address	Interface disable
Write byte to even address	Interface enable

4.6.2 Outgoing read commands

Local bus read commands in the System bus address space are latched in the output pipeline register if not already full. If full the initiator might get suspended. If the burst request signal is asserted on the local bus a 16 byte read command will be issued and A2-A3 of the local address forced to 0 (since 16 byte read commands on the System bus must be 16 byte aligned). If the 68030 is bus owner and does not assert the CIOUT* or the CBREQ* signal the read command will be encoded as a 4 byte read command. This enables the 68030 to cache the particular location. When the read command is latched in the pipeline, as is the mapped address, the destination address, the command type, local address bit 2-3, and a command initiator bit.

The output pipeline might contain two read commands at the same time. When a read command is sent on the System bus the Command Sequencer is passed information about the response size and the intended receiver. A read command is not sent until the previous read response is received and delivered.

4.6.3 Incoming responses

If the command sequencer is waiting for a 16 byte response all the 16 bytes are be received before the command sequencer returns any data to the initiator. For responses < 16 bytes, local address bit A2 determines what half of the pipeline register will be enabled on to the local bus. For 16 byte responses the command sequencer looks at the latched local bus address (A2-A3) to determine in what order to return the 4 long words stored in the input pipeline register (68030 might request burst transfers that are not 16 byte aligned).

If the response is of type error or error data a bus error is returned when the response is delivered and the input response pipeline is flushed. In the case only one response type error or data was received when two partial responses were expected, then the read command initiator will time out. The input pipeline must then be flushed before being used again.

5. IOPM local bus arbiter

The IOPM local bus arbiter coordinates all bus activities on the IOPM local bus. There are three possible bus masters : 1. Device board. 2. Command Sequencer, 3. 68030. The local bus arbiter gives ownership to one of these three bus masters. With a few exceptions, most bus ownership switches will take zero clock cycles. The local bus arbiter can suspend a bus master if the accesses a resource not available at this time (e.g.Command buffer full) or if the bus master is waiting for a response. The local bus arbiter also keeps track of who is suspended and continues the suspended bus masters bus cycle whenever the requested resource becomes available. To keep the local bus access time (DMA latency) to a minimum for the device board, a 68030 burst read operation from local memory will be terminated early (See 68030 Manual section 7) when the local bus is requested by the device board or the Command Sequencer.

5.1 Bus priorities

The 68030 is normally the owner of the IOPM bus and has the lowest bus priority. The device board interface section has the highest priority, and the Command Sequencer has the second highest priority.

There are three different causes for a bus ownership switch : 1. Bus Request by higher priority bus master 2. Bus cycle can be suspended, 3. Suspended bus cycle can continue. The IOPM local bus arbiter controls all ownership switching.

The device bus has the highest priority on the local bus and when requesting the bus it will be granted the bus when the current bus operation (cycle or burst) terminates. When the device board is signaled to suspend its bus cycle, it will be forced off the bus. When the IOPM local bus arbiter later signals continue the device board continues it's suspended bus cycle.

5.2 Bus timeout

The IOPM has two bus masters with time-out. The Command Sequencer which has a time-out value of 64 uS, timed from the time the command is received until it is executed. When timeout occurs this error condition is latched in the error status register in a 68030 NMI is generated. The 68030 has a bus cycle time-out value of one second. If a 68030 bus cycle times out a berror exception is generated, the error condition is latched in the error status register and an NMI is generated.

5.3 Suspended operation

A bus master can be suspended when it accesses a resource not available at the time. A burst cannot be suspended once the first part of the burst has been delivered. Following conditions will or may temporary suspend the operation of a bus master.

Device board will be suspended if :

1. Accessing the System bus interface and the command buffer is full,
   and someone else can perform a local bus cycle.

2. Waiting for a System bus response and someone else can perform a local bus cycle.

68030 will be suspended if :

1. Accessing the device board and the device board signals that the bus cycle will be suspended, and someone else can perform a bus cycle.
2. Accessing the System bus interface and the command buffer is full, and someone else can perform a bus cycle.
3. Accessing the command buffer and the device board is waiting for the command buffer to become non full.
4. Waiting for a System bus response and someone else can perform a bus cycle.

Command sequencer cannot be suspended. Software must ensure that commands from System bus does not access device boards that can suspend the initiators bus operation.

6. Device board interface

The device board interface uses a 96 pin DIN connector to implement a synchronous address data multiplexed interface. The interface is a general purpose interface supporting, DMA, burst read/write operations and interrupts. Most signals are synchronous to the 20 MHz clock supplied at the interface.

A device board that only transfers data to local memory, or is a pure slave does not need to support some of the more sophisticated features of the device board interface.

6.1 Supported functions

Following functions are supported when the IOPM is bus master :

- 1,2,3,or 4 byte read or write operations.
- Burst read operations. Always 16 bytes.
- Atomic operations. (68030 initiated only).
- Suspension of bus cycle if 68030 is initiator.

Following functions are supported when the device board is bus master:

- 1,2,3,or 4 byte read/ or write operations.
- Burst read or write operations. Always 16 bytes. (Burst write to IOPM memory only)
- Suspension of bus cycle if System bus is target.

Following additional functions are supported over the device board interface :

- Interrupt the IOPM at three different interrupt levels.
- Reset device board.
- Reset IOPM. Only used in Console mode.
- Set the IOPM to Console mode.
- LED indicators can be driven by device board.

Dynamic bus sizing is not supported over the interface. Therefore software and hardware has to ensure that all transfers are alinged on the correct boundaries.

If locations on a device board is going to be cashable by the 68030, the device board must return 32 bits of data on reads independent of the SIZE1, SIZE0, and A1-A0 signals.

Burst read operations can start on any 4 byte aligned address, and cannot be suspended once the first long word of the burst read have been delivered. If a bus-error occur during a burst, the burst transfer is terminated with the transfer that had the bus error.

The 68030 can perform atomic operations over the device board interface. For the operation to be atomic the first read cycle must be in the device board address space that supports atomic operation, and the device board must not suspend the 68030 once the first read of the atomic operation has been acknowledged. The remaining cycles of the atomic operation must be to the device board address space or the IOPM local RAM or the operation is not guaranteed to be atomic.

The device board can at any time request to become bus master. Once bus master the device board may perform as many operations as it find suitable. Note that the Command Sequencer time-out is 64 uS, and the Command Sequencer should under normal circumstances execute an incoming command within about 6 uS.

Burst write operations by a device board must only be performed with the IOPM memory as target. The burst must be aligned on a 4 byte address, and the DEV.SIZE0 and DEV.SIZE.ONE signals must indicate a long word operation.

Only bus operation performed by a device board in the System bus address space can be suspended. The device board can be suspended, waiting for a response or waiting for a free entry in the command buffer.

6.2 Classes of device boards

Not all device boards will have to support the more sophisticated features available on the device bus. We will therefore describe three classes of device boards: I. Device boards that are slaves only, II. Device boards that supports DMA but cannot be suspended, III. Device boards that supports DMA to main memory.

Non of the three classes have to support burst operations as master or slave. The burst operations are purely a feature to increase the throughput.

The first class of device boards that are slaves only are the simplest. This class will probably not support any suspension of initiators, but will instead let the initiator wait until ready to perform the operation. This class cannot become suspended since it cannot become bus master.

The second class supports DMA but only to local memory, and can therefore not be suspended. This class might elect to suspend an initiator, when slave because internal data busses might be used, or for any other reason, when a DMA operation must take place before any other operation can be performed.

The third class can perform DMA to main memory and must therefore support being suspended. It might however elect to suspend or not suspend an initiator when itself is suspended.

6.3 Device interface signals

Following signals make up the device board interface (out is from IOPM to device board):
Power and ground signals are not listed.

Signal name	direction	tristate	function
DEV.AD[0..31]	BI	yes	address and data multiplexed lines
DEV.CYC.ST.OUT*	OUT	yes	Cycle start out
DEV.CYC.ST.IN*	IN	yes	Cycle start in
DEV.SIZE[0..1]	BI	yes	transfer size encoding bits
DEV.RW	BI	yes	Read/Write
DEV.BURST.REQ*	BI	yes	Burst request
DEV.RMC*	OUT	yes	Read Modify Command
DEV.STERM.IN*	IN	yes	Synchronous termination in
DEV.STERM.OUT*	OUT	yes	Synchronous termination out
DEV.MEM.ACCESS*	OUT	no	IOPM memory cycle starts
DEV.BURST.ACK*	BI	yes	Burst Acknowledge
DEV.BUS.ERROR*	BI	yes	Bus error
DEV.BUS.REQ*	IN	no	Bus request
DEV.BUS.ACK*	OUT	no	Bus acknowledge
DEV.SUSP.OUT*	OUT	no	Suspend device board
DEV.SUSP.IN*	IN	no	Suspend Initiator
DEV.INT4*	IN	no	Interrupt Request level 4
DEV.INT5*	IN	no	Interrupt Request level 5
DEV.NMI*	IN	no	Non Maskable Interrupt
DEV.RST.OUT*	out	no	Reset of device board
DEV.RST.IN*	In	no	Reset of IOPM
DEV.LED1*	IN	no	LED1 ON
DEV.LED2*	IN	no	LED2 ON
DEV.CLK*	OUT	no	Clock

6.4 Device Interface signal description

DEV.AD[0..31] = Device Address Data
Address data multiplexed lines. The current bus master drive the address until the bus cycle starts, then switches to drive or receive data depending on operation. The bus is the driven one clock cycle after the cycle is terminated. Both IOPM and device boards ensures that there is no bus fight on these lines turning on the drivers slower than turning them off.

DEV.CYCLE.ST.OUT = Cycle Start Out
This synchronous (with respect to the negative edge of clock) control line starts a bus cycle when the IOPM is bus master. Will be negated if IOPM is suspended.

CYCLE.ST.IN = Cycle Start In
This synchronous control line (negative edge of CLK) starts a bus cycle when the device board is bus master. When a device board becomes bus master it is free to start the bus cycle when ever ready to do so.

Bus cycle control lines :
DEV.SIZE[0..1]
DEV.RW
DEV.BURST.REQ*
DEV.RMC*
These bidirectional control (except for DEV.RMC), driven by the current bus master indicates what type of bus cycle is in progress. They must be valid when DEV.CYCLE.ST.IN* or DEV.CYCLE.ST.OUT* is asserted and device board is not suspended.

Cycle Termination signals :
DEV.STERM.IN* = Synchronous termination of bus cycle in
DEV.STERM.OUT* = Synchronous cycle termination of bus cycle out
DEV.BURST.ACK* = Burst Acknowledge
DEV.BUS.ERROR*
DEV.MEM.ACCESS* = IOPM memory access starts
These signals terminates a bus cycle, and are driverr by the target. A bus cycle is always terminated by DEV.STERM.OUT* or DEV.STERM.IN*. If DEV.BURST.ACK* is asserted during a burst cycle, and DEV.BUS.ERROR* is negated (valid on read cycles only), the burst will continue until four long words are transferred. Burst cycles (4 long words) are considered one bus operation and cannot be suspended once started. BUS.ERROR on a read burst cycle will terminate the burst early. DEV.MEM.ACCESS* is a redundant signal that can aid device boards performing burst writes to IOPM memory, giving an early warning of when the DEV.STERM.OUT* is going to be asserted. Using this will give more time to prepare the following long words of a burst transfer.

Bus ownership control signals :
DEV.BUS.REQ* = Bus request

DEV.BUS.ACK* = Bus acknowledge
DEV.SUSP.OUT = Suspend device board
DEV.SUSP.IN = Suspend Initiator
The BUS.REQ* and BUS.ACK* signals are unidirectional control signals synchronous with respect to the positive edge of CLK. Once bus owner a device board can elect to use the bus for any length of time by holding the BUS.REQ* line asserted. The device board becomes bus master, and can drive the bus on the clock cycle after when the DEV.BUS.ACK* is asserted. The device board will receive one bus acknowledge for every bus operation it has requested permission to performed. If the BUS.REQ* is asserted when the last DEV.STERM.IN* in a bus operation is asserted then the device board will continue to be bus master. DEV.SUSP.OUT suspends the device board. The device board must hold the CYCLE.ST.IN* signal asserted, but release the AD and the control lines on the clock cycle after DEV.SUSP.OUT* was asserted.

The suspended cycle will continue on the clock cycle after the DEV.SUSP.OUT* is negated. The device board may elect to suspend a bus cycle when the device board is a bus slave by asserting the DEV.SUSP.IN* signal. The full bus cycle will be rerun when the device board negates DEV.SUSP.IN* and the current cycle ends.

Interrupt control lines :
DEV.INT4* = Request level 4 interrupt
DEV.INT5* = Request level 5 interrupt
DEV.NMI* = Non Maskable Interrupt
These are unidirectional asynchronous interrupt control lines that can be driven by the device board.

DEV.RST.OUT* = Reset of device board
DEV.RST.IN* = Reset of IOPM (Normally only used in console mode)

DEV.LED1* = LED1 ON
DEV.LED2* = LED2 ON
Controls the two lowest LEDs on the front of the IOPM .

DEV.CLK* = Clock signal
Must be terminated on device board and received with an equivalent load to four 74AS1804 loads. With this arrangement it is estimated that the maximum clock skew between clocks on the IOPM and the device board is less then 5 nS.

7. 68030 Processor

The IOPM will have a 20 MHz 68030 running synchronous with the System bus. Other than the 68030, the IOPM hardware will not differentiate between supervisor and user mode.

The 68030 bus operation is slightly different from the 68020. When the 68030 reads a cacheable entry it expects as many bytes back as the device port size, even if it has signaled a byte read operation. Since the System bus, and the IOPM bus does not support dynamic bus sizing only target that can return 32 bits independent of read command must be allowed to be cached (e.g. local memory and memory modules).

The normal 68030 BR*, BACK*, and BGACK* signals will not used to arbitrate for the local bus. When the 68030 is not bus master it is disconnected from the IOPM local bus and held in a wait state until the local bus arbiter makes the 68030 bus master.

OCS, ECS, CIIN, REFILL, STATUS, CDIS, MMUDIS signals will not be used in the design. The CIOUT* will be used to encode the System bus read command type. A four byte read will always be issued if CIOUT* is not asserted and CBREQ* is not asserted (This allows main memory to be cached in the 68030). The IOPM hardware will be designed such that a IOPM can operate without the 68030 installed.

7.1 Power up and Reset

On power up and reset the EPROM will be mapped so the 68030 can fetch it's PC and PSW from location 0. To proceed and access the System bus the ERPOM has to be remapped. This is done by setting the DIS_EPROM bit in a control register. A 68030 reset instruction will not reset any part of the IOPM board.

On power-up the processor will be held reset (if not in Console mode) until a module enable control command is send to the IOPM. Only module disable control commands from the System bus, and an DEV.RST.IN* can reset the IOPM.

7.2 Bus errors

When a bus cycle is terminated with bus error a normal bus error exception will be taken. The cause of the bus error can be found in the error status register. A bus cycle that times out will also be terminated with a bus error.

7.3 Atomic operations : TAS, CAS, CAS 2 instructions

The TAS, CAS, and CAS 2 instructions will generate special Read Modify Write operations. These RMW operations will be treated as atomic operation in respect to other atomic operations on the System bus if and only if the first read cycle of the atomic operation is in the System bus address space. The 68030 can also perform atomic operations over the device board interface. All bus cycles must then be restricted to the IOPM memory and device board. The first bus access of the atomic operation must be to the device board. Note that these atomic operations will increase device board DMA latency time.

The 68030 also uses RMW cycles to do table walks. These operations are not normally atomic unless the tables are kept in main memory. There is no support for atomic operations from the device board or on local memory.

Atomic operation can hold up the System bus interface for an extended period of time while waiting for the lock to be acquired. During this time no device board DMA to the System bus can be performed, so DMA involving main memory must have adequate buffering.

7.4 68030 Interrupts

The 68030 interrupts will be auto-vectored for all seven interrupt levels. Two different mechanisms exists for generating interrupts. The first one is used by the device board. A hardware signal directly generates the particular level interrupt. When the interrupt is serviced the software has to take action to negate the signal that generated the interrupt. The second scheme is used by processor modules, the 68030, or the device board, and is implemented as an I/O register. When a particular bit in the register is set a corresponding level interrupt will be generated. The 68030 is normally responsible for clearing the bits.

Level	Interrupt cause	Comment
7	Register and hardware	All errors will cause NMI
6	Register	Software or Hardware, not assigned
5	Device Board Interrupt	
4	Device Board Interrupt	
3	Register	Software or Hardware, not assigned
2	Register	Software or Hardware, not assigned
1	Register	Software or Hardware, not assigned
0	No interrupt	

8. Local memory

The local memory will be 1 MByte parity protected organized 256k x 32. The memory will support nibble mode operation implemented with fast page mode memory devices. The memory will support both read and write burst operation. Normal read or write cycles will take 250 nS, and consecutive burst cycles will take 100 nS/cycle. Memory reads with parity errors will cause a bus error on the IOPM bus. Read operations always reads 32 bits independent of SIZE1, SIZE0. Burst cycles by the 68030 will be terminated early if the device board is requesting use of the local bus. This will minimize the DMA latency for DMA devices. The upper half can be protected from access by anybody else than the 68030, by setting the MEM.PROTECT bit in the configuration control register.

8.1 Memory refresh

A local memory refresh cycle will be performed every 12.8 uS. Refresh is transparent to the local bus cycle, and regular 250 nS memory accesses can be stretched to 500 nS if a refresh just started when a memory access from the local bus started.

9. Control/Status Registers

The IOPM will have various Control/Status registers for controlling and diagnosing the IOPM hardware.

9.1 Error Status Register

Any bit set in this register will cause an 68030 NMI exception. The CLEAR.ERROR* bit the Configuration Control register is used to clear these error conditions. All bits in the error register will be cleared on power-up or board reset.

68030_TIMEOUT* This bit is set when a 68030 bus cycle is longer then about one S.
When this bit is set the device board will be forced of the
bus, the command sequencer will be disable and a 68030 bus
error will be generated.

CS_TIMEOUT* This bit is set when the command sequencer did not finish a
command in 80 uS. The command sequencer will be disabled.

NACK_SENT* This bit indicates that the input buffer section NACKed an incoming
System bus transmission.

NO_ACK* This bit is set when the output buffer section sent a
command or response on the System bus and didn't receive an ACK.

NACK_RCVED* This bit is set when the output buffer section sent a command
or response and received a NACK.

MEM_PERROR* Parity error detected on memory read cycle was detected.

MEM_PRO_ERR* Memory protection error. Asserted if Command Sequencer or
Device board accesses the upper half of the local memory
when the protection bit in the Configuration Control register
is set.

DEV.BERROR* This bit is set when a transmission over the device board had
the DEV.BERROR* signal asserted.

9.2 Status Registers One

The Status register one is a read only register.

EXPANSION* Indicates that the IOPM is in an expansion cabinet.

INTERFACE_ENABLE Indicated that the interface section to the System bus
is enabled and the IOPM can now talk to the System bus.

CONSOLE	Indicates that the board is also service processor module and has certain privileges.
SLOT[0..3]*	Complimented slot number on the System bus.
CMD.2.OUT	Two commands are in the System bus output section.
CMD.1.OUT	One or more commands are in the System bus output section.
RX.CMD	A command from the System bus is in the input buffer.
DIAG.BD	Diag bag present. Active low.
INT.7	Interrupt register level 7 bit is set.
RX.FREEZE	System bus freeze signal is asserted. Active low.
DEV.NMI	Device board NMI is asserted. Active low.
DEV.SUSP	Device board is suspended.

9.3 Diagnostic Pipeline Registers

Four of the eight pipeline register in the System bus input section are readable for diagnostic purpose. The content reflects what was last entered into the particular pipeline register. The registers are 32 bit :

1. COMMAND.DATA, 2. COMMAND.ADDRESS, 3. Last incoming response lower long word, 4. Last incoming response upper long word.

9.4 Control Registers

There are 16 map register whereof 15 are used to translate IOPM bus addresses to System addresses.

MAP[0..F]	Sixteen 8-bit entries for the address translation map. Entry hex F is not used as a map. MSB of this register is System bus address 35, and LSB is System bus address bit 28.

A parity polarity register for the System bus output section, enables diagnostic to check parity generation and detection in the System bus interface section.

CSS_PAR[0..7]	Controls the sent parity on the System bus. If a bit is set then the wrong parity is generated for that particular System bus byte. This register is cleared on reset.

Configuration Control register :

MEM_PAR_POL[0..3]	When set, writes to local RAM will generate wrong parity. Used for diagnostic purposes only. Cleared on reset.
CSS_CONT_WR	When set, write on IOPM bus in the System bus address range will generate a System bus control write.

DIS_EPROM When set the EPROM is disabled and the IOPM can talk to the System bus.

EXPANSION_LOOPB Normally when a IOPM is in an expansion cabinet all commands and responses are sent all the way to the main cabinet even if the destination is in the expansion cabinet. This bit when set enables a IOPM to loop back on the System expansion bus. This bit is only set when the IOPM is installed in an expansion cabinet and when performing self test.

DIS.CS When set disables the Command Sequencer from becoming bus master. For diagnostic use only.

DO_CSS_RESET When set and in CONSOLE mode the System bus reset line is asserted.

ENA.DEV.BD Enable the device board to become bus master.

DEV_RESET.OUT* When set will generate a device board reset.

FLUSH_.OUT Flushes the output pipeline.

FLUSH_.IN Flushes the input pipeline.

CLEAR.ERROR* When set clear all error bits in the error register. Active low.

FREEZE Asserts the System bus freeze signal.

PROTECT.RAM When set protects the upper half of the local RAM from being accessed by anybody else than the 68030.

LED control register :

FAIL_LED* This bit turns off the red fail indicator.

PASS_LED When set illuminates the PASS LED.

SOFT.LED1 When set illuminates the SOFT.LED1.

SOFT.LED2 When set illuminates the SOFT.LED2.

Interrupt control register : This register controls the interrupt levels of pending interrupts to the 68030.

INT.1 Generates a pending level one interrupt.

INT.2	Generates a pending level two interrupt.
INT.3	Generates a pending level three interrupt.
INT.6	Generates a pending level six interrupt.
INT.7	Generates a pending level seven interrupt.

9.5 IOPM LED indicators

There is a set of LEDs visible through the PCB retaining plate on the front edge of the PC board. There are ten LEDs in total. The first two are diagnostic status indicators and the others are activity indicators.

There are two diagnostic status indicators which appear on the front edge of the card above the retaining screw. These indicate gross diagnostic status of the module. The top diagnostic status indicator is red and the bottom is green. A red indicator means the board has been diagnosed bad by diagnostics or that diagnostics have not run. The green indicator indicates that the board has passed diagnostics. The power-up or reset state on these indicators will be red(on) and green(off).

Following is a brief description of what the LEDs indicate, from top to bottom.

Name	Color	Function
FAIL	Red	Software controlled, turned on when a failure is detected or after PON.
PASS	Green	Software controlled (turned on after passing self test).
ACTIVE	Red	Turned on every time the IOPM asserts BUS_ACTIVE on the System bu
CS.CMD	Red	Turned on when Command Sequencer detects an incoming command.
SUSPENDED	Red	Turned on when someone is suspended by the local bus controller.
DEV.REQ	Red	Turned on when the device board asserts bus request.
SOFT.LED1	Red	Software controlled.
SOFT.LED2	Red	Software controlled.
DEV.LED1	Red	Device board status indicator
DEV.LED2	Red	Device board status indicator

10. Free running counter

The IOPM has a word wide read only free running counter. This counter is implemented with a ripple counter not synchroniced with the read access and can therefore return a false value if read when changing value. The counter should therefore be read until two consecutive reads return the same counter value. Each counter tick is 12.8 uS.

11. Console Support

This section describes how a IOPM could be used to implement what we call a Console. A Console is a IOPM and a device board that implements a subset of the functions of a

Service Processor Module and a Real World Interface. In addition it would also have a device interface. In small systems this subset should be sufficient for the Console to act as a SPM, and thereby cost reduced the minimum systems cost. This section describes the functions added to the IOPM to support a Console. It also lists the functions of an SPM that will not be supported and the implications this will have.

Following is a list of the functions added to the IOPM to support Console mode:
- Device board signal to select between IOPM normal or Console mode.
- In Console mode the IOPM can issue System bus reset.
- In Console mode the IOPM can perform control write commands to the System bus.
- Device board can reset the IOPM.

Following functions will not be supported :
- Bus watching of NACKs and missing ACKs.
- Diagnostic functions to send illegal System bus commands, command with errors, emulate other slot #, etc.
- System bus source address is used as offset into the interrupt dispatcher.

All other functions of the SPM, and Real World Interface can be implemented on the device board or in software on the IOPM.

There are some system implications by only implementing a subset of the SPM functions. First it cannot diagnose a system as well as a SPM could. Secondly the interrupt dispatcher, if any would reside on the device board could not be made totally compatible with the SPM interrupt dispatcher because the SPM uses the System bus source address as an offset into the interrupt vector table. This means that the kernel would need to change to use some other scheme for reading its interrupt vector.

12. Performance of bus cycles

Following are the estimated timing for various read and write cycles and estimated times for bus owner ship switches.

All accesses to local memory can be stretched 250 nS if the access starts just after a refresh has started.

68030 as bus local bus mater :

Read/write of long word from/to local memory : 250 nS
Burst read (4 long words) from local memory : 550 nS

For accesses to System bus time estimated timing assumes write accelerator empty, no contention on System bus or on the memory board.
Read of long word from main memory : 1100 nS
Burst read (4 long words) from main memory : 1550 nS

Write long word to main memory : 200 nS
Read or write to device board : Minimum 250 nS.

Device board is bus master :

Read/write of long word from/to local memory : 250 nS
Burst read (4 long words) from local memory : 550 nS
Write long word to main memory : 200 nS

For the following discussion we assume that the target IOPM performs only local memory accesses, and uses 80 % of the local bus bandwidth, 60 % of the bus accesses are burst, 40 % are long word accesses. No contention on System bus is assumed.

68030 bus master :
Read long word from another IOPM's local memory : min 1150 nS, average 1240 nS
Burst read from another IOPM's local memory : min 1800 nS, average 1890 nS
Write multiple (large copy) long words to another IOPM's local memory : min 600 nS, average 690 nS

For the following discussion we assume that the local 68030 performs only local memory accesses, and uses 80 % of the local bus bandwidth, 60 % of the bus accesses are burst, 40 % are long word accesses. No contention on System bus or in main memory is assumed. Device board is initiator of bus cycle. Keep in mind that if the 68030 did some other operations, like accessing device board, or System bus, some of the max time can get dramatically increased.

Times indicated are from device board bus request to the device board completes the bus cycle :

Read/write long word from/to IOPM local memory : Min = 350 nS, average 400 nS, max 750 nS.
Max time is when 68030 just started a burst read and a refresh cycle is performed before the device board can perform it's bus cycle.

Write to write accelerator, when not full : Min 300 nS, average 350 nS, max 700 nS.

Read burst from main memory : Min 1650 nS, average 1800 nS, max 2200 nS.

13. Power-up and reset

The IOPM implements five types of resets, 1. Power On Normal (PON), 2. Board reset, 3. Device Board reset out, 4. Device Board reset in, 5. 68030 Processor Reset (module disable control command).

Each reset is dependent on various conditions described below.

13.1 Power on reset

When the IOPM is powered on a circuit detects acceptable voltage levels and keeps the entire board reset for an additional 100 mS. Whenever the voltage falls below the acceptable voltage the board will be held in a reset condition.

13.2 Board Reset

There are three conditions that will reset the entire board. 1. A power on reset, 2. When not in Console mode and the System bus signal is asserted, 3. The Device board reset in signal is asserted.

13.3 Device Board reset out

The device board reset out is asserted when the Device board reset out bit in the configuration control register is set. This bit is asserted by program control or at power-on or when a board reset is performed.

13.4 Device Board reset in

If the device board contains logic for a Console then the device board may generate a reset signal to the IOPM and the System bus.

13.5 68030 Processor reset

The 68030 is reset on a board reset, or when not in Console mode and the module has not yet been enabled by the SPM. Once enabled a module disable control command will again reset the 68030 until an other module enable command is received.

14. Clock distribution

The IOPM will use the 20 MHz BCLK* signal as it's main source of clock information. This signal will be loaded by six 74AS1804 (or equivalent) loads. The 74AS1804 will drive the clock lines on the board. These clock lines must be routed without stubs and will have DC terminations.

20 MHz clocks signals on the device board shall propagate through the same number of gates (1 buffer delay on device board) as the main clock on the IOPM. The clock skew between IOPM clock and device board clock is estimated to < 5 nS.

15. Connectors

The IOPM board has three different connectors: 1. 150 pin female System bus connector, 2. 80 pin female Arbiter connector, 3. 96 pin male device board connector.

For signal description of system bus and Arbiter connector see System 3000 Hardware Description. The device board connector signals are described in section "Device board interface".

15.1 Connector J1 Pinouts

Pin	Row A	Row B	Row C
1	COMMON	BUS_DATA_PAR[0]	BUS_DATA[07]
2	COMMON	BUS_DATA[06]	BUS_DATA[05]
3	COMMON	BUS_DATA[04]	BUS_DATA[03]
4	COMMON	BUS_DATA[02]	BUS_DATA[01]
5	COMMON	BUS_DATA[00]	BUS_DATA_PAR[1]
6	COMMON	BUS_DATA[17]	BUS_DATA[16]
7	COMMON	BUS_DATA[15]	BUS_DATA[14]
8	COMMON	BUS_DATA[13]	BUS_DATA[12]
9	COMMON	BUS_DATA[11]	BUS_DATA[10]
10	COMMON	BUS_DATA_PAR[2]	BUS_DATA[27]
11	COMMON	BUS_DATA[26]	BUS_DATA[25]
12	COMMON	BUS_DATA[24]	BUS_DATA[23]
13	COMMON	BUS_DATA[22]	BUS_DATA[21]
14	COMMON	BUS_DATA[20]	BUS_DATA_PAR[3]
15	COMMON	BUS_DATA[37]	BUS_DATA[36]
16	COMMON	BUS_DATA[35]	BUS_DATA[34]
17	COMMON	BUS_DATA[33]	BUS_DATA[32]
18	COMMON	BUS_DATA[31]	BUS_DATA[30]
19	COMMON	BUS_DATA_PAR[4]	BUS_DATA[47]
20	COMMON	BUS_DATA[46]	BUS_DATA[45]
21	COMMON	BUS_DATA[44]	BUS_DATA[43]
22	COMMON	BUS_DATA[42]	BUS_DATA[41]
23	COMMON	BUS_DATA[40]	BUS_DATA_PAR[5]
24	COMMON	BUS_DATA[57]	BUS_DATA[56]
25	COMMON	BUS_DATA[55]	BUS_DATA[54]
26	COMMON	BUS_DATA[53]	BUS_DATA[52]
27	COMMON	BUS_DATA[51]	BUS_DATA[50]
28	COMMON	BUS_DATA_PAR[6]	BUS_DATA[67]
29	COMMON	BUS_DATA[66]	BUS_DATA[65]
30	COMMON	BUS_DATA[64]	BUS_DATA[63]
31	COMMON	BUS_DATA[62]	BUS_DATA[61]
32	COMMON	BUS_DATA[60]	BUS_DATA_PAR[7]
33	COMMON	BUS_DATA[77]	BUS_DATA[76]
34	COMMON	BUS_DATA[75]	BUS_DATA[74]
35	COMMON	BUS_DATA[73]	BUS_DATA[72]
36	COMMON	BUS_DATA[71]	BUS_DATA[70]
37	COMMON	BUS_SRC[3]	BUS_SRC[3]*
38	COMMON	BUS_SRC[2]	BUS_SRC[2]*
39	COMMON	BUS_SRC[1]	BUS_SRC[1]*
40	COMMON	BUS_SRC[0]	BUS_SRC[0]*
41	COMMON	BUS_DEST[3]	BUS_DEST[3]*
42	COMMON	BUS_DEST[2]	BUS_DEST[2]*
43	COMMON	BUS_DEST[1]	BUS_DEST[1]*
44	COMMON	BUS_DEST[0]	BUS_DEST[0]*
45	COMMON	BUS_TYPE_PARITY	BUS_TYPE[5]
46	COMMON	BUS_TYPE[4]	BUS_TYPE[3]
47	COMMON	BUS_TYPE[2]	BUS_TYPE[1]
48	COMMON	BUS_TYPE[0]	BUS_DATAPAR_VLD
49	COMMON	BUS_IOX_ACTIVE	BUS_ACTIVE
50	COMMON	BUS_ACK	BUS_NACK

15.2 Connector J2 Pinouts

Pin	Row A	Row B	Row C
2		+5V	
4	+12V	N/C	+12V
5	ARB_CLOCK*	COMMON	EXPANSION*
6	ARB_READY_DEC*	COMMON	BUS_SLOT[3]
7	ARB_DEST[3]	COMMON	BUS_SLOT[2]
8	ARB_DEST[2]	COMMON	BUS_SLOT[1]
9	ARB_DEST[1]	COMMON	BUS_SLOT[0]
10	ARB_DEST[0]	COMMON	BUS_IOA_SLOT[3]
11	ARB_GRANT*	COMMON	BUS_IOA_SLOT[2]
12	ARB_BURST*	COMMON	BUS_IOA_SLOT[1]
13	ARB_REQUEST*	COMMON	BUS_IOA_SLOT[0]
14	ARB_RESP*	COMMON	BUS_C14
15	ARB_MODIFY*	COMMON	BUS_C15
16	ARB_LOCK*	COMMON	COMMON
17	ARB_READY1*	COMMON	BUS_FREEZE*
18	ARB_READY3*	COMMON	COMMON
19	ARB_GRANTERR*	COMMON	BUS_RESET*
20	BUS_A20	COMMON	COMMON
21	BUS_A21	COMMON	BUS_C21
22	BUS_A22	COMMON	BUS_C22
23	BUS_A23	COMMON	BUS_C23
24	BUS_A24	COMMON	BUS_C24
25	+12VAUX	+12VAUX	+12VAUX
26	BUS_A26	COMMON	BUS_C26
27	-12VAUX	-12VAUX	-12VAUX
28	BUS_A28	COMMON	BUS_C28
29	-12V	N/C	-12V
31		+5VAUX	

15.3 Connector P3 Pinouts

Pin 32 of the P3 connector is the pin closest to J2.

Pin	Row A	Row B	Row C
1	+12V	+5V	+12V
2	COMM	DEV.AD00	DEV.AD01
3	+5V	DEV.AD02	DEV.AD03
4	+5V	DEV.AD04	DEV.AD05
5	COMM	DEV.AD06	DEV.AD07
6	COMM	DEV.AD08	DEV.AD09
7	+5V	DEV.AD10	DEV.AD11
8	COMM	DEV.AD12	DEV.AD13
9	COMM	DEV.AD14	DEV.AD15
10	COMM	DEV.AD16	DEV.AD17
11	COMM	DEV.AD18	DEV.AD19
12	COMM	DEV.AD20	DEV.AD21
13	COMM	DEV.AD22	DEV.AD23
14	COMM	DEV.AD24	DEV.AD25
15	COMM	DEV.AD26	DEV.AD27
16	COMM	DEV.AD28	DEV.AD29
17	COMM	DEV.AD30	DEV.AD31
18	DEV.LED1*	DEV.LED2*	DEV.CONSOLE*
19	COMM		
20	COMM	DEV.BUS.REQ*	DEV.BUS.ACK*
21	COMM	DEV.SUSP.OUT*	DEV.MEM.ACCESS*
22	COMM	DEV.SUSP.IN*	DEV.CYC.ST.OUT*
23	COMM	DEV.CYC.ST.IN*	DEV.SIZE0
24	COMM	DEV.SIZE1	DEV.BURST.ACK*
25	COMM	DEV.BURST.REQ*	DEV.RMC*
26	COMM	DEV.R/W	DEV.BERROR*
27	COMM	DEV.STERM.OUT*	DEV.STERM.IN*
28	COMM	DEV.INT.5*	DEV.INT.4*
29	COMM	DEV.RST.OUT*	DEV.NMI*
30	COMM	DEV.CLK*	DEV.RST.IN*
31	-12V	+5V	-12V
32	+5V	+5V	+5V

16. Power consumption

The IOPM will use 5V only. It is estimated that this board will consume 12 A at 5V. The power consumption for the device boards is limited to 35 W. The current limits for the device boards are 8 A at 5V and 2 A at +-12V.