IP Modules
Draft Standard

VITA 4-1995

Draft 1.0.d.0
April 7, 1995

This standard is being prepared under the VITA Standards Organization (VSO) and is unapproved.

Do not specify or claim conformance to this document.

VITA Standards Organization
10229 North Scottsdale Road, Suite B
Scottsdale AZ 85253
Voice: 602-951-8866
Fax: 602-951-0720

GreenSpring Computers, Inc
1204 O'Brien Drive
Menlo Park CA 94025
Voice: 415-327-1200
Fax: 415-327-3808

Abstract

This standard defines a versatile module, known as an “IP module.” These modules provide a convenient method of implementing a wide range of I/O, control, interface, slave processor, analog and digital functions. IP modules, about the size of a traditional business card, mount parallel with a host Carrier board, which provides host processor or primary bus interfacing, as well as mechanical means for connecting the IP module’s I/O to the outside world. Typical Carriers include standalone processors, DSP based carriers, as well as desktop buses and VME based boards. This specification includes mechanical, host bus electrical, and logical definition of I/O space, memory space, identification space, interrupts, DMA, and reset functions. Two physical sizes, two fixed clock rates, and multiple data width sizes to 32-bits are defined.

Scope of the Standard

This document define the following elements of IP Modules:

- Electrical Interface of the Logic connector, including voltage levels, loading, pin definition, cycle types, timing diagrams, and state diagrams;
- Mechanical size and mounting of the IP Modules, for single-size and double-size;
- Mechanical and electrical requirements of the IP Module Carrier Board;
- The mechanical I/O connector;
- The minimum contents for the ID space;
- Permissible feature combinations.

The following elements are beyond the scope of this Standard:

- I/O pin assignment;
- I/O system wiring;
- Regulatory compliance issues;
- System power;
- Host system architecture, processor, instruction set and byte ordering;
- System and application software issues;
- Definitions of any individual modules.

Foreword

As VME became the industrial bus of choice in the 80s, the standard IC moved from a 20-pin MSI DIP to a VLSI high pin count surface mount device. Although the functionality of standard boards increased dramatically, the average selling price remained at about $2500. This high per-slot cost prompted manufacturers to continue to place as much functionality on each board as possible, to keep the total slot and total system costs as low as possible. This trend increased the basic functionality granularity (the smallest purchasable function) 10 to 50 fold. It was no longer practical to purchase separately a small number of serial lines or a small amount of memory. Many manufacturers attempted to remedy this problem by offering proprietary "daughter modules" for their processor boards. At least 20 different such proprietary offerings were on the market by the end of the 1980s.

At this time GreenSpring Computers, under the technical direction of Kim Rubin, created a module with the specific design and intent that it become an open, widely accepted standard. GreenSpring Computers introduced the module and its specification as an open standard at BUSCON in 1988.

When the modules were introduced, GreenSpring incorporated features that were found in only a few, if any, other modules:

- Up to four modules fit on one VME or PC-AT board, for high modularity
- Bus and processor independent; supported both Motorola and Intel byte ordering
- Simple, synchronous interface made Modules and Carriers easy to design
- Rugged mechanical mounting, for ease of use and high reliability
- Defined ID space for configuration management
- Low cost

The concept caught on, and by 1994 over 80 companies were designing and offering products to this standard, on at least seven standard buses. GreenSpring called their modules "IndustryPacks®," but each manufacturer was free to use whatever trade name they wished. The modules are called "IP Modules," a term that is now in the public domain. This standard is a direct result of that work.

In 1991 Motorola Computer Group incorporated the IP Module specification into its 68040 based MVME162, implementing four IP Module positions in a single VME slot. This helped propel the acceptance of the specification. For the remainder of this document, the modules will be identified as "IP Modules." Manufacturers and users are encouraged to use this term for the generic identification of these modules.

At the start of 1994 a formal standards committee was put together under the VSO, the VITA Standards Organization, which had received accreditation as a standards organization under ANSI. Most of the committee's work has been done via internet, which has proven effective. The purpose of the committee is to validate and clarify the existing specification. In particular, timing details for 32 MHz and DMA operation are being clarified. Also, the specification was reformatted, with numbering added for easier reference and formality.

1.

Description

The IP Module is a versatile electronic module that provides a convenient level of modularity for implementing a wide range of I/O, control, interface, analog and digital functions.

Up to two IP Modules can be mounted on a 3U (single high) VMEbus board, up to four IP Modules on a 6U (double high) VMEbus or "C" size VXIbus board. IP Modules also fit conveniently on Macintosh II™ Nubus boards. In these busses, mounted IP Modules meet all applicable bus specifications. Figure 1 shows a typical IP Module mounting arrangement (in this case, on a 3U VME board).

The IP Module specification is comprehensive. Memory is supported, as are interrupts and DMA. The basic IP Module interface—8 MHz, single-size—supports a continuous data rate of eight MB/second as four megatransfers per second using 16-bit word transfer size. Peak continuous data rate for a 32 MHz, 32-bit Module is 64 MB/second as 16 megatransfers per second using a 32-bit data transfer width. Each IP Module includes a configuration (ID) PROM. This ID information may be used by software for autoconfiguration.

IP Modules come in two sizes, single-size and double-size. The double-size appears electrically and mechanically as two adjacent IP Modules. This Specification describes the single-size IP Module unless explicitly stated otherwise.

Two connectors are used on each IP Module. One is dedicated to control of the IP Module, and is fully specified. This connector contains the Logic Interface, which is described in this document. This interface is generally thought of as a "bus," although not all signals are actually bussed. The other connector is provided for the IP Module's specific function. It's interface is called the I/O Interface. All 50 signals in the I/O Interface are defined by each IP Module. The connectors are an industry standard part. The connector pins are gold plated and fully shrouded; the connectors are keyed, assuring reliable, repeatable field installation.

Each IP Module communicates with its Carrier board via a single 50 pin connector. This connector contains address, data, clock, control and power lines. This "bus" is called the VITA-4 Logic Interface. This document is the specification for that interface.

All data transfers between the Carrier board and the IP Module are done synchronously. This provides a simple, reliable design. CMOS logic levels are used to keep power dissipation of both IP Modules and Carrier boards to a minimum.

Features:

- Simple, reliable electrical design (synchronous transfers)
- Simple, reliable mechanical design (shrouded, keyed connector)
- Mechanically compatible with 3U VME, 6U VME, Nubus, ISA, PCI and VXI
- ID PROM permits autoconfiguration, compatible with VXI and NuBus
- word (128 byte) I/O space per IP Module
- Up to 8 MB memory space per single-size IP Module
- Byte or word addressing
- Two interrupts per single-size IP Module
- Two DMA channels per single-size IP Module
- 64 Mbyte/second peak continuous data rate
- 32-bit data width and 32 MHz clock options

Table 1 Signal Identification

Function	Name	No of Pins	Class
Data Bus	D00..D15	16	data
Address	A1..A6	6 (Note 1)	address
Reset	Reset*	1	control
I/O Select	IOSel*	1	control
Memory Select	MemSel*	1	control
Read Int Vector	IntSel*	1	control
Clock	CLK	1	clock
Module Identification	IDSel*	1	control
Data Direction	R/W*	1	control
Data Acknowledge	Ack*	1	control
Byte Select	BS0* BS1*	2	control
Interrupt Request	IntReq0* IntReq1*	2	control
DMA Request	DMAReq0* DMAReq1*	2	control
DMA Acknowledge	DMAck*	1	control
DMA Termination	DMAEnd*	1	control
IP Module Error	Error*	1	control
Function Strobe	Strobe*	1	option
Ground	GND	4	power
+5 Volts	+5V	2	power
+12 Volts	+12V	1	power
–12 Volts	–12V	1	power
Reserved		2	reserve
Total		50	

An asterisk (*) after the signal name means that the line is active low. All other signals are active high.

Note 1: During memory access cycles, the 16 data lines are used for the high order address lines (A7..A22).

2. Signal Descriptions

The following paragraphs discuss each of the signals used in the IP Module Logic Interface.

2.1. D00..D15 [Data Bus]

The 16 lines of the data bus are used to read and write data between the Carrier board and the IP Module. These lines are used for I/O, memory, DMA, interrupt vectors, and ID data. During memory addressing, these lines carry the high order 16 bits of the address (A7..A22). Data transfer is synchronous with CLK. During some transfers only the low order eight (D0..D7) lines are used. See the figures at the end of this document for timing details.

2.2. A1..A6 [Address]

The six lines of the address bus are used for addressing I/O locations. These lines are the low order six lines during memory addressing. Only the Carrier board drives the address lines. The address lines are interpreted as "word" addresses. Byte writes are accomplished by using the Byte Select (BSx*) lines. A1 is used during interrupt acknowledge cycles and DMA acknowledge cycles to select which interrupt or DMA channel is being acknowledged. A1 through A5 are used to select which of 32 bytes are being read during Read ID cycles. A6 should be low during ID space reads (for compatibility with planned support of serial EEPROM devices).

The address lines are not used for selecting IP Modules. The four select lines (IOSel*, MemSel*, IntSel* and IDSel*) are used for this purpose. This may be interpreted as "geographic addressing," since the select lines are not bussed.

2.3. Reset*

The active low Reset* signal is driven from the Carrier board to the IP Modules to reset the circuitry to a known state.

Reset* must be driven following power up for at least 200 milliseconds. There is no maximum time limit. When Reset* is asserted to the IP Modules they must terminate any cycle in progress, remove any interrupt requests and DMA requests, and disable future interrupt and DMA requests (until enabled by software). Reset may be asserted asynchronously, but must be de-asserted synchronously.

Reset* must also be asserted by the Carrier board if the power supplies fall below minimum specifications of the carrier board or host bus. +5 must be monitored; monitoring of other supplies is optional. This is required particularly so that the Reset* signal may be used by an IP Module to safely switch to standby or fail-safe operation. Thus the Reset* signal is a logical OR of "Reset" and "Power-Fail" functions.

Unless an IP Module has no local registers it must respond to Reset*. If a IP Module has non-volatile memory, that memory need not be reset. However the IP Module's documentation must clearly state which locations and registers are affected by Reset*.

A IP Module may take some time following the end of Reset until it is capable of responding . This time may be needed to load programmable logic, perform a self-test, internal calibration, internal initiation or other function. The IP Module's documentation must clearly state the maximum time necessary for the module to

initialize, and what functions, if any, are available prior to the completion of the local initialization.

2.4. IOSel* [I/O Select]

The active low IOSel* signal is one of four "select" lines driven by the Carrier board to enable the IP Module. This line is used for input or output cycles. These cycles use only the six address lines A1.. A6. Each IP Module thus occupies 64 words, or 128 bytes in the system I/O space.

Data width on I/O cycles is a function of the IP Module itself, and is generally fixed. If only eight bits are used by an IP Module, they must be the low order (D0..D7) data lines.

IOSel* is unique to each IP Module; it is not bussed. A IP Module need not respond to IOSel* if it has no I/O functions.

The byte select lines must be driven valid by Carrier boards. IP Modules are not obligated to examine the byte select lines during accesses in the I/O space, however.

A IP Module need not respond to IOSel* if it has no functions in the I/O space. It need not decode any address lines that it does not need. Thus functions (registers) within an IP Module's I/O space may appear in multiple places within the 64 word IO space.

IOSel* is also used in I/O DMA cycles. For these cycles, DMAck* is asserted simultaneously with IOSel*. I/O DMA cycles are compelled. Each cycle is in response to a DMA Request from the IP Module. See DMA Section for more information.

2.5. MemSel* [Memory Select]

The active low MemSel* signal is one of four "select" lines driven by the Carrier board to enable the IP Module. This line is used for memory read or write cycles. These cycles use up to 22 address lines to address up to eight MB per IP Module. The synchronous protocol uses the data lines to carry the high order 16 address signals during the first clock of the memory access cycle.

Memory write cycles use the two byte select signals (BS0*, BS1*). IP Modules, during memory read cycles may ignore the byte selects by driving both bytes, if desired. Carrier boards must drive the byte select signals valid on all memory cycles.

MemSel* is unique to each IP Module; it is not bussed. A IP Module need not respond to MemSel* if it has no memory.

MemSel* is also used in Memory DMA cycles. For these cycles, DMAck* is asserted simultaneously with MemSel*. Memory DMA cycles are not compelled. These cycles indicate that the source of the cycle is a host DMA controller. See DMA Section for more information.

2.6. IntSel* [Read Int Vector]

The active low IntSel* signal is one of four "select" lines driven by the Carrier board to enable the IP Module. This line is used for reading the IP Module's interrupt vector during an interrupt acknowledge cycle. A1 is used to indicate if the acknowledge is for IntReq0* (A1 is Low) or IntReq1* (A1 is High). The interrupt vector may be eight or 16 bits, but typically is eight.

IntSel* is unique to each IP Module and is not bussed. A IP Module need not respond to IntSel* if it has no interrupt requests asserted. However an IP Module should not remove interrupt requests without a specific action from the host Carrier board.

2.7. CLK [Clock]

This signal is a 50 percent duty cycle 8.0 MHz or 32.0 MHz clock driven to the IP Module by the Carrier board. All data transfers are synchronous to this clock. The signals involved in a data transfer are changed relative to the rising edge of the clock. Signals generated by the Carrier board are typically sampled by the IP Modules on the next rising edge of the clock. Signals generated by the IP Modules are sampled by the Carrier board on the rising edge of the clock. See Figure 9 for timing details.

Recommendation 2-1: It is recommended that Carrier board designers use a series terminated CLK driver to avoid reflections on this key signal. For traditional .062 inch thick FR4 circuit boards, six to eight layers with internal power and ground planes and typical design rules of .007 trace and .008 gap, 33Ω has been found to be a good value. A separate series termination resistor for each IP Module is recommended. Carrier board designers may wish to consider that in such a design the IP Module will see the CLK edge delayed slightly from the edge at the output of the CLK driver.

Recommendation 2-2: It is recommended that IP Module designers keep in mind that the CLK signal may not be as clean as desired. Also that there is likely to be some skew between the edge received by the IP Module and the edge known to the Carrier board. Therefore, conservative timing design is recommended.

Low level timing is not the same for 8 MHz and 32 MHz interfaces. Also, the basic cycle state machine is slightly different for 8 MHz and 32 MHz operation. IP Modules must be specified for operation at 8 MHz, 32 MHz or for both frequencies. All Modules must be nominally functional at 8 MHz, including correct reading of the ID PROM.

2.8. IDSel* [IP Module Identification]

The active low IDSel* signal is one of four "select" lines driven by the Carrier board to enable the IP Module. This line is used for reading the 32 byte ROM that contains the IP Module identification information. The address lines A1..A5 are used to select the ID byte. IP Module identification information contained in the ROM permits software to perform autoconfiguration.

IDSel* is unique to each IP Module and is not bussed. All IP Modules must respond to IDSel*. The configuration ROM contains three classes of information: (i) fixed identification data defined by this Specification, (ii) IP Module dependent data, defined for each IP Module and set by the manufacturer, and (iii) remaining space may be used by the customer, if desired. See Section "ID PROM Data" for more information.

Double-size IP Modules need to have an ID PROM on only their “A” side.

2.9. R/W* [Data Direction]

The R/W* signal is driven by the Carrier board to indicate the direction of data flow on the data lines. High indicates a Read cycle (IP Module drives data lines). Low indicates a Write cycle (Carrier board drives data lines). R/W* may change

asynchronously during idle cycles. It must, however, meet the synchronous timing requirements during all select, wait and hold cycles.

R/W* has the same directional meaning for the data bus for standard cycles and for DMA cycles. It is possible to support a cycle where one IP Module is writing data to another IP Module during the same cycle; however to do this the R/W* line to each IP Module must be driven separately, and not bussed. Typically, IP Modules read and write data to the host Carrier board only.

2.10. Ack* [Data Acknowledge]

Ack* is asserted by the IP Module to terminate each data transfer. Each IP Module may continuously drive Ack* high or low, or it may tri-state this signal when de-asserted to save power. The minimum data transfer takes two clock cycles (one for the select, one for the acknowledge). This permits a maximum continuous data rate of eight MB/second for each module. Ack* is driven relative to the rising edge of CLK. See Figure 9 for timing details.

A IP Module may hold off asserting Ack*, effecting "wait states." The Carrier board may hold a select line asserted on read or write cycles, creating "hold states." After the IP Module asserts Ack* during hold states, the IP Module then holds Ack* asserted as a "hold acknowledgment."

Ack* is not bussed.

There is no fixed maximum amount of time by which an IP Module must assert Ack*. However the individual IP Module's specification must clearly state access time, or "wait states." In some cases the number of wait states may be a function of external events. If there is a chance that data could be lost then the IP Module must have a maximum wait time. If no Ack* occurs within this time, then the IP Module will never Ack*. This permits timeout logic elsewhere in the system to take effect, permitting software to execute a re-try, or otherwise take specific action.

OBSERVATION 2-1: Without this limit, it might be possible for the system timeout and the IP Module's Ack* to occur simultaneously, losing that data.

2.11. BS0*..BS1* [Byte Select]

The two byte select lines are used to indicate which of the two data bytes are valid. While IP Modules may ignore the byte select lines for some cycles, Carrier boards must drive BS0* and BS1* valid on all cycles. IP Modules must observe the byte select lines if they implement 16-bit memory. BS0* selects the low, or odd, byte (D0..D7). BS1* selects the high, or even, byte (D8..D15). Both strobes together select both bytes.

The two byte strobes may change asynchronously during idle cycles. They must however meet the synchronous timing requirements during all select, wait and hold cycles.

OBSERVATION 2-2: Big endian byte ordering (Motorola 68K and VMEbus convention) is compatible with IP Modules. This is the convention used throughout this document. Big endian uses even byte addresses (e.g. 0) for the high byte (D8..D15). Carrier boards and software must be careful to document how they map byte addresses. Generally speaking: (1) word addresses for 16-bit devices are even numbered; (2) 8-bit devices are on odd addresses (big endian byte ordering), (3) machines that implement 32-bit busses with no dynamic sizing (e.g.

NuBus) will typically map 8-bit IP Modules to every fourth byte address and 16-bit IP Modules in every other word.

OBSERVATION 2-3: Little endian byte ordering (Intel x86 family convention) is compatible with IP Modules, although this convention is not used in the document. Little endian uses even byte addresses (e.g. 0) for the low byte (D0..D7). Carrier boards in little endian systems should always map data lines from the host bus or processor to like numbered data lines on the IP Module, e.g. D0 to D0. Data strobes must be driven appropriate for the actual byte lane(s) used. Carrier boards and software must be careful to document how they map byte addresses. Generally speaking: (1) word addresses for 16-bit devices will not change; (2) 8-bit Modules on odd addresses (using big endian convention) will change to even addresses (under little endian byte ordering).

2.12. IntReq0*..IntReq1* [Interrupt Request]

The IP Module specification permits each module two dedicated interrupts. IntReq0* and IntReq1* are not bussed. These signals, if used, may by be driven by the IP Module either open drain or by conventional (two state) MOS or CMOS components. Synchronization to CLK is not required by the IP Module. This permits VLSI components on the IP Module to have their interrupt request outputs directly wired to the connector pins. The Carrier board must provide passive pullup resistors and synchronization, if needed.

Reset* forces any pending interrupt requests to be removed, and interrupts to be disabled until explicitly re-enabled by software. This means that all IP Modules, must provide a way to prevent external signals from generating an interrupt until some access to the IP Module from the host occurs. Ideally there is a control register bit that enables IP Module interrupts.

2.13. DMAReq0*..DMAReq1* [DMA Request]

The IP Module specification permits each module two dedicated DMA channels. DMAReq0* and DMAReq1* are not bussed. These signals, if used, are driven by the IP Module as conventional (two state) logic to indicate that the IP Module wishes to have a DMA cycle performed. These signals are synchronized to CLK by the IP Module. Carrier boards must have passive pull-ups on these signals to support IP Modules that have no DMA.

See DMA section for more detail.

DMAReqx* is removed by the IP Module, if there is not an additional DMA request pending, by the falling (asserted) edge of the Ack* signal driven by the IP Module at the end of the DMA acknowledge cycle. Thus the Carrier board can sample a valid DMAReqx* on the same clock edge that it finds Ack* to be valid. A IP Module may delay generating Ack* with wait states in order to be sure to have DMAReqx* again valid by this time. Or it may automatically remove DMAReqx* each cycle, then re-assert as soon as it can determine that an additional request is pending.

The DMAReqx* lines request a compelled, I/O DMA cycle. They may not be used to request a DMA cycle to memory on the module.

When the Module requests a DMA cycle, it cannot directly indicate data width nor data direction desired. This information must have been decided in advance of the request.

A IP Module may not request a DMA cycle until such action has specifically been enabled on the Module by the host processor following Reset.

2.14. DMAck* [DMA Acknowledge]

The IP Module specification supports two types of DMA cycles, I/O DMA and Memory DMA. In both cases the DMA cycle is a modification of the standard I/O or standard Memory cycle. The signal DMAck* is driven by the Carrier board to indicate that the cycle is a DMA cycle. This signal is timed the same as the A1..A6 Address Bus that is driven at the same time.

I/O DMA cycles are compelled. They occur only in response to the Module requesting such a cycle by asserting DMAReq0* or DMAReq1*. The Carrier board drives IOSel*, DMAck* and A1 low or high to respond. For each I/O DMA cycle there is a previous, corresponding DMA request. A1 is driven low in response to DMAReq0*, and high in response to DMAReq1*.

Memory DMA cycles are not compelled. They are similar to standard memory access cycles. The address lines are defined the same as for standard memory cycles. Memory DMA cycles do not occur in response to DMAReqx*. A IP Module will typically chose to interpret Memory DMA cycles in the same way as it interprets standard Memory cycles. The Carrier board will generally drive the DMAck* line on DMA Memory cycles when the cycle has been initiated by a DMA controller, rather than the host processor.

> **OBSERVATION 2-4: DMAck* is not really an "acknowledge" signal when used with Memory DMA cycles, since it is not in response to a DMA request. It serves primarily in this case to identify to the IP Module that the source of the cycle is a DMA controller. Modules may use this information to process the cycle differently than a standard memory cycle, if that is appropriate to the function of the Module. In this case, the DMAck* signal may be viewed as a "select modifier."**

All DMA cycles may be either reads or writes. The R/W* indicates data bus direction in the same way as for non-DMA cycles. DMA cycles may be 8-, 16- or (on double-size Modules) 32-bits wide.

2.15. DMAEnd* [DMA Termination]

This bidirectional, open-drain line is used to terminate I/O DMA transfers. Both IP Modules and Carriers that support DMA must recognize DMAEnd* as an input. IP Modules that support DMA may optionally drive DMAEnd* as an output. Carrier boards that support DMA must drive DMAEnd*. DMAEnd* is driven open-drain by both the Carrier board and the IP Module.

If the transfer ends because a terminal count has been reached the DMA controller asserts DMAEnd*. A IP Module asserting this open drain signal forces DMA termination. The Carrier board and/or the IP Module may assert DMAEnd* to indicate either abnormal termination or normal termination of the DMA transfers. No additional transfers for that DMA channel should occur, nor any DMA requests for that channel should occur, until the host software has recognized the termination and appropriately responded.

OBSERVATION 2-5: The purpose of DMAEnd* is to assure that the DMA elements of Modules and Carrier boards stay synchronized relative to DMA "in progress" status. The source of the DMAEnd* signal (either the Carrier board's DMA controller or the IP Module) would typically generate an interrupt to the host. If DMAEnd* is used to indicate a premature termination of a transfer then both ends of the transfer are able to take immediate, appropriate action. If DMAEnd* is used to signal a normal termination (for example, DMAEnd* occurring

simultaneously with a previously programmed terminal count) then chained DMA, or perhaps continuing normal operation on a communications channel can occur automatically without fear that an error might be lost or overrun in the system.

DMAEnd* is driven during with the final DMA cycle. The Carrier board asserts DMAEnd* during the clock cycle following the Select cycle. If there are no Hold states or Wait states then this would be during the Acknowledge state. If there are Wait states then DMAEnd* would be asserted by the Carrier board prior to the Acknowledge state.

The IP Module asserts DMAEnd* so that it is valid during its Acknowledge state. It is thus timed similar to valid read data.

OBSERVATION 2-6: If a IP Module needs to know if the Carrier is asserting DMAEnd*, even if the Module itself is asserting DMAEnd*, then it may insert at least one Wait state and sample DMAEnd* during this Wait state, prior to asserting DMAEnd* itself.

RECOMMENDATION 2-7: IP Modules, particularly at 32 MHz, should not sample DMAEnd* until the cycle following Select in order that the signal has sufficient time to rise to a valid logic high, since it might have been driven low during the previous cycle.

Both Carrier boards that support DMA and IP Modules that support DMA must have a 1.2 KΩ pullup resistor to +5 volts on this signal line. Modules that do not support DMA should not connect to this signal line. DMAEnd* must be driven by a 24 mA high sink current driver.

2.16. Error* [IP Module Error]

A IP Module may assert this open-drain signal asynchronously to signal a non-recoverable error. This signal may not be used to indicate invalid bus protocol, time out, nor as a pre-initialization indicator. It may be used to indicate component failure, unrecoverable self-test failures, or serious, hard-wired configuration errors.

Less serious errors should be signaled using an interrupt.

2.17. Strobe* [Function Strobe]

This uncommitted line may be used as an optional input to or output from a IP Module. It is reserved for a digital strobe or clock signal related to the Module's functionality. For example, Strobe* could be used for a sample clock to an A/D converter from a local timer, or a communication line clock signal.

Neither Carrier boards nor Modules are required to use the Strobe* line. Modules that can optionally drive the Strobe* line must provide a mechanism to disable the Strobe* driver.

OBSERVATION 2-8: The Strobe* line may be used for signals that are time critical. Such signals would not be able to use interrupts or other defined communication mechanisms between the Carrier and the Module.

OBSERVATION 2-9: The Strobe* line may be used for two Modules to communicate with each other without using their I/O wiring or the other defined Module to Carrier communication mechanisms. Such use is beyond the scope of this Specification, however.

RECOMMENDATION 2-10: Carrier boards could provide a passive pullup resistor on the Strobe* lines, and a simple shunt block that would permit end customers to connect the Strobe* line of one IP Module to another Module, if desired, via a shunt.

RECOMMENDATION 2-11: IP Modules that use the Strobe* line should provide the option of implementing the same functionality via an I/O line. This provides compatibility with Carriers that do not connect to the Strobe* line, and provides a way to communicate with Modules not mounted on the same Carrier.

2.18. GND [Ground]

Four pins are assigned in the four corners of the logic connector for ground. This is the zero volt reference for the logic signals on this connector, and the return path for the power supplies for both digital and analog functions on the IP Modules (that are powered through the logic connector).

Minimum electrical noise in the ground is desirable. This increases noise immunity on digital signals, improves the accuracy of analog functions, and minimizes radiated RF emissions. This specification encourages ground plane construction within both Carrier boards and IP Modules.

External power supplies may be connected to IP Modules via the I/O connector. A Carrier board may provide power to the IP Module logic connector either by wiring it directly from the host machine, regulating or filtering the host supply, connecting to external power or on-board batteries, or other techniques.

OBSERVATION 2-12: System grounding of power supplies, sensors, and external cabling is beyond the scope of this specification. Significant care should be exercised by hardware manufacturers, system integrators, and end users to meet applicable safety and performance requirements.

2.19. +5V [+5 Volts]

This is the primary supply for digital logic functions on the IP Module. Two pins provide for up to two Amperes of current maximum from this supply.

OBSERVATION 2-13: Power supply regulation, conducted RF noise, and power dissipation are beyond the scope of this specification. However good regulation and the use of RF filtering is a recommended practice. Similarly, power dissipation on IP Modules should be kept to a minimum.

Please see additional discussion under GND, above.

2.20. +12V, -12V [+12 Volts, -12 Volts]

These two power supplies are primarily used to power analog and communication functions on IP Modules. One pin is available for each voltage, providing a maximum current of 1 Ampere each.

OBSERVATION 2-14: Power supply regulation, conducted RF noise, and power dissipation are beyond the scope of this Specification. However good regulation and the use of RF filtering is a recommended practice. Similarly, power dissipation on IP Modules should be kept to a minimum.

These two power supplies are not required to be supplied by the Carrier boards, but are strongly recommended. Carrier boards that do not provide these voltages must be clearly marked.

IP Modules that require these, or other voltages, may chose to have them provided (required or optionally) via the I/O connector.

Please see additional discussion under GND.

2.21. Reserved

These two pins may be used in future revisions to support additional features. Carrier boards should connect to these pins with passive pull-up resistors.

3. Cycle Types

The following table shows allowable data transfer cycle types and their implementation using the Carrier board driven control signals. This table is for single-size (16-bit) Modules. See Section 32-bit Operation for more information on double-size 32-bit cycle types.

Table 2 Cycle Types

Cycle Type	R/W*	**IOSel***	**MemSel***	**IntSel***	**IDSel***	DMAck*
Input	H	**L**	H	H	H	H
Output	L	**L**	H	H	H	H
Memory Read	H	H	**L**	H	H	H
Memory Write	L	H	**L**	H	H	H
Interrupt Ack	H	H	H	**L**	H	H
ID Read	H	H	H	H	**L**	H
I/O DMA	H/L	**L**	H	H	H	**L**
Memory DMA	H/L	H	**L**	H	H	**L**

Note that exactly one of the four IP Module select lines (IOSel, MemSel*, IntSel* or IDSel*, shown in bold) is asserted for each cycle type.*

The following table shows allowable memory addressing by cycle type. This table is for single-size (16-bit) Modules. See Section 32-bit Operation for more information on addressing on double-size 32-bit Modules.

Table 3 Address Spaces by Cycle Type

Cycle Type	**Address Space (size)**	**Address Lines Used**
Input	64 words	A1..A6
Output	64 words	A1..A6
Memory Read	Up to 8 Mbytes	A1..A22 Note 1
Memory Write	Up to 8 Mbytes	A1..A22 Note 2
Interrupt Ack	2 vectors	A1
ID Read	32 bytes	A1..A5 Note 3
I/O DMA	2 channels	A1 Note 4
Memory DMA	Up to 8 Mbytes	A1..22

Note 1: Byte memory reads are supported by the Carrier board: the IP Modules always read either eight or 16 bits wide fixed.

Note 2: Memory write cycles use BS0 and BS1* to select the lower byte, upper byte or word (16 bit) access. The minimum and maximum address spaces for memory read cycles and write cycles is the same.*

Note 3: Format I ID reads are on the lower, byte (D0..D7). Format II ID reads are 16-bits wide. The ID space takes up 32 words.

Note 4: I/O DMA cycles use A1 to indicate which channel is being acknowledged.

4. Addressing Conventions

4.1. A0 Usage

The IP Module Logic Interface does not use A0. A0 as used by the host is normally used to select the high or low 8-bit byte out of a 16-bit word. Traditionally A0 is the least significant address line that can address individual bytes. (This is no longer true on all processors, such as DSPs.) The Logic Interface uses a 16-bit data bus. 16-bit wide memory (RAM) uses the byte selects for byte writes to fully support eight and 16 bit memory operations from the host and/or DMA.

Address notation follows the industry convention. This is sometimes called "byte addressing." A0 is thus implicitly present in all written addresses.

This specification uses the convention that the low order byte in a word is called address one, or "odd." Byte-wide I/O addresses are normally on the low order byte, thus occupy consecutive odd addresses. This is consistent with “big endian” architectures, such as the Motorola 68K family and the VMEbus. IP Modules are also compatible with little endian systems.

See discussion under I/O Space and Memory Space, and under the BSx* lines, for more details.

4.2. Words versus Bytes

This specification uses the convention that data width for registers is fixed in the design of the Module. These registers are typically accessed in the I/O Space. Accesses within the Module’s Memory Space are assumed to access memory, and that both byte and word addressing is supported. These conventions are not fixed requirements of this specification, however. Modules may support flexible register data width, although this is rare. Modules may also support fixed memory data width, although this is not recommended.

A more common situation is the placing of registers in the memory space. In this case, access width may be fixed, even though access in the memory space. This occurs, for example, when an IC has more than 64 registers, or an IC provides fixed address register decoding in an address space that is larger than 64 words. Another case of fixed data width in the memory space can occur when a Module uses registers suitable for high-speed DMA (for example FIFO data registers), but wishes to make them available for non-compelled DMA access.

5. I/O Space

The six address lines give 2^6 or 64 I/O locations. Each location may be 16 bits wide. In many cases only eight bits will be used. When only eight bits are used they must be the low order (D0..D7). It is not necessary that all of the bits read or written (8 or 16) be meaningful, but it is necessary to drive either eight or 16 of the data lines.

Passive pullup resistors on the Carrier board guarantee that undriven lines are always read as a logical one.

Each IP Module should verify that only the IOSel* is asserted each cycle, to avoid confusion with any future definitions of advanced cycle types that might be implemented by asserting multiple select lines.

Each IP Module uses 128 bytes out of the I/O space of the host. Some Carrier boards use A7 to select the ID ROM space of each IP Module. In this case each IP Module uses 256 bytes out of the host's I/O space. This permits 256 IP Modules in one 64 KB I/O space.

OBSERVATION 5-15: Nubus and some other busses are 32-bits wide only. One likely mapping is that each IP Module occupies the low 16 (or 8) bits out of each 32-bit long word. Carrier boards on these busses that use this mapping must make this explicit in all documentation. Address maps given for IP Modules may be converted to Nubus addresses by multiplying the IP Module address by two and adding two.

OBSERVATION 5-16: This specification does not enforce any particular mapping by Carrier boards between the host(s) and the IP Module(s). Busses and processors 64 bits wide, for example, may use unconventional mappings. These unconventional mappings may not be software compatible across different systems, however.

6. Memory Space

The IP Module format permits IP Modules to contain substantial memory. The fully synchronous data transfer cycles make it easy to multiplex the data bus with the high order 16 address lines. The minimum data transfer cycle is two clocks, permitting a maximum continuous data transfer rate of basic IP Modules (8 MHz, 16-bit access) of eight MB/sec. 32-bit, 32 MHz Modules will support a peak write rate of 64 MB/sec, and a peak read rate of 42.67 MB/sec.

A IP Module may have only memory, only I/O, or both memory and I/O. During memory cycles the first clock cycle is used to transmit the memory address on the data lines. The second clock cycle is then used to move eight or 16 bits of data. More clock cycles may be used if needed by the IP Module by not asserting Ack* until it is ready to terminate the cycle.

The dedicated six low address lines plus the 16 address lines that share the data bus produce a maximum address space of 2^{22} words or eight MB per module. Unlike the 64 words of I/O space, each memory IP Module takes a variable amount of memory. Different Carrier boards implement memory addressing differently. Memory data width may be either eight or 16 bits, although 16 bits is more common. Details of the memory (size, type, speed, width) are contained in the configuration ROM on each IP Module. This permits software to perform autoconfigurations, as well as automatic memory diagnostics on power up.

Each IP Module should verify that only the MemSel* is asserted each cycle, to avoid confusion with any future definitions of advanced cycle types that might be implemented by asserting multiple select lines.

OBSERVATION 6-17: Nubus and some other busses are 32-bits wide only. One possible mapping is that each IP Module occupies the low 16 (or 8) bits out of each 32-bit long word. This mapping may be inappropriate for some types of memory. Note however that this mapping is common on Nubus, even for code (which is copied into contiguous memory prior to being executed.) Carrier boards on these busses that use this mapping must make this explicit in all documentation. Address maps given for IP Modules may be converted to Nubus addresses by multiplying the IP Module address by two and adding two.

OBSERVATION 6-18: This specification does not enforce any particular mapping by Carrier boards between the host(s) and the IP Module(s). Busses and processors 64 bits wide, for example, may use unconventional mappings. These unconventional mappings may not be software compatible across different systems, however.

7. Direct Memory Access

7.1. I/O DMA Cycles

Direct memory access (DMA) control is handled primarily by circuitry on the Carrier board. DMA features may be optionally supported by IP Modules and Carrier boards. For DMA cycles to occur both the Carrier and IP Module must support DMA.

DMA cycles are available in two type: I/O DMA and Memory DMA. Only the I/O DMA is compelled with a DMA request from the IP Module. In general, throughout this Specification, unless otherwise stated, “DMA” means I/O DMA only.

Prior to any (I/O) DMA cycles occurring, the host software must program the Carrier-based DMA controller and enable DMA on the IP Module. Reset* will halt any DMA activity until both the Carrier board and the IP Module are programmed by the host software. A IP Module initiates DMA activity by asserting one of its two DMAReqx* lines low, synchronous to the IP Module CLK. The DMA controller on the Carrier board responds by generating a DMA acknowledge cycle, which is a modification of a standard I/O data transfer cycle. The IP Module removes the DMA request line prior to the end of the cycle, unless it has another request ready, then it may leave the request line asserted.

DMA cycles may effectively be throttled by either the IP Module or the DMA controller on the Carrier board. If the IP Module does not wish to have another DMA cycle occur yet, it simply does not reassert its request line. If the DMA controller wishes to postpone the DMA cycle (perhaps because it is allocating bus bandwidth among competing requesters) it simply delays generating the DMA acknowledge cycle.

Either the IP Module or the Carrier board’s DMA controller may drive DMAEnd*. The DMA controller would typically drive DMAEnd* when either a terminal count was reached or there was an error condition that prevents further DMA cycles from being effective (such as a host memory error or bus error). The IP Module would typically drive DMAEnd* when it recognized a logical end of data (as might happen with variable length data communication packets) or when it encountered an error condition (for example a SCSI error in the middle of a sector). When DMAEnd* is asserted the host is interrupted and must interpret the cause of the termination and take appropriate action. IP Modules that support DMA must recognize DMAEnd* Carrier boards that support DMA must recognize DMAEnd*. However neither IP Modules nor Carrier boards are obligated to generate DMAEnd*. It is recommended that DMA controllers do generate DMAEnd*, at least for normal terminal count conditions. DMAEnd* applies only to the channel being accessed in that cycle.

Note that a Carrier board’s DMA controller that recognizes DMAEnd* may not need to be explicitly reprogrammed by the host processor if it has the ability to respond appropriately internally. For example, the DMA controller may support “chained” DMA operations, or it may have its own internal processor.

There are two DMA channels per single-size IP Module. There are four channels on a double-size IP Module. The request lines are non-multiplexed: DMAReq0* and DMAReq1* for the channel 0 and channel 1 respectively. The acknowledge cycle encodes the DMA channel number into A1. For 32-bit DMA, the request lines should be from the A side of the double-size Module. For 16-bit DMA on a double

(I/O) DMA cycles are a modification of standard I/O cycles. DMAck* is driven, similarly to an address line (A1..A6 timing), while IOSel* is asserted during the Select state. A1 is the only address line driven valid. A1 low indicates the cycle is responding to DMA request channel 0 (DMAReq0*) while A1 high indicates the cycle is responding to DMA request channel 1 (DMAReq1*). IP Modules should ignore address lines A2..A6 during DMA cycles. I/O-DMA cycles have the same data width options, eight through 32 bits, as standard I/O cycles. Unless otherwise stated, the cycle has similar characteristics to standard I/O cycles.

Because DMAReqx* is valid when Ack* is asserted (on the same rising CLK edge) it is possible to implement “back-to-back” DMA cycles.

See Section 18 for permissible combinations of DMA support.

7.2. Memory DMA Cycles

Memory DMA cycles are very similar to regular Memory cycles.

A variation on standard memory cycles is called a Memory DMA cycle. Unlike standard I/O DMA cycles, the address lines are used as in a standard Memory cycle. A1 is a memory address lines; it is not used to indicate a DMA channel. Indeed, since the DMA Request lines are used only for requesting I/O DMA channels, there are no identifiable “Memory DMA” channels.

On Memory DMA cycles, the Carrier drives DMAck* true along with MemSel*. Timing for DMAck* is similar to an address line A1..A6. DMAEnd* may optionally be driven by either the IP Module or the Carrier Board to indicate the last Memory DMA cycle. Note that the function of DMAEnd* on Memory DMA cycles is different than for I/O DMA cycles, since there are already no associated DMA requests across the IP Module interface. Data width and other characteristics are the same as for standard Memory cycles.

See Section 18 for permissible combinations of DMA support.

8. Interrupt Acknowledge

When an IP Module needs service it asynchronously asserts IntReq0* or IntReq1*, either open-drain or with a standard driver. These signals are not bussed.

The Carrier board responds by doing an interrupt acknowledge cycle. The select line IntSel* is asserted along with A1. A1 low responds to IntReq0*. A1 high responds to IntReq1*. The IP Module places its vector as valid data on the data lines (8-bit vectors use the low data lines D0..D7) and asserts Ack*. The interrupt request signal is removed either during the interrupt acknowledge cycle or later, when the IP Module has been serviced by a software routine.

IP Modules that have only a single interrupt should use IntReq0*.

Neither Modules nor Carrier boards are required to support interrupts. Interrupt support is strongly recommended.

9. ID PROM Data

There are two basic permissible formats for the required ID PROM on all IP Modules. The first format, Format I, dates back to the earliest modules, uses an 8-bit wide data path and 8-bit Manufacturer ID field.

Format II is newer. It uses 16-bit wide data and a 24 bit Manufacturer ID field.

Each IP Module must contain a configuration ID PROM whose size is at least 12 x 8. This ID PROM is read by the IDSel* line, and address lines A1 through A5. Type I ID PROM data is placed on the low order, D0..D7 data lines. Type II data is place on all 16 data lines. Double-size IP Modules need only have an ID PROM on the A side. Data is read when R/W* is high and IDSel* is asserted. A6 should be low during ID PROM reads.

The ID PROM contains three classes of information:

- Fixed identification data in one of two formats,
- IP Module dependent data, defined for each Module type and set by the manufacturer,
- Remaining space may be used by a system integrator or end user, if the PROM data is accessible to them on the Module.

Data formats are shown in the two Tables below.

RECOMMENDATION 9-19: Write cycles into the ID space are undefined by this Specification. It is recommended that Carrier boards provide the capability to write into the ID space. Read cycles to addresses where A6 is high are undefined by this Specification. It is recommended that Carrier boards provide the capability to access this portion of the ID space.

OBSERVATION 9-20: Data in the ID PROM space may be provided by a dedicated PROM component, or by logic in an FPLD or other means.

ID PROM access must take less time than 4.0 µsec, once the IP Module is initialized.

OBSERVATION 9-21: Note that some IP Modules take many milliseconds to initialize following Reset (for example, loading Xilinx® devices). Host systems should be able to tolerate a long delay following Reset prior to reading IP Module's ID PROMs. IP Module manufacturers must state the longest time required following the end of Reset prior to functionality.

9.1. ID PROM Data Format I

Table 4 ID PROM Data Format I

Address	Description	Contents
3F 2*nn+1	User Space	
2*nn-1 19	IP Module Specific Space	
17	CRC	
15	Number of bytes used	= nn
13	Driver ID, high byte	
11	Driver ID, low byte	
0F	Reserved	00
0D	Revision	
0B	Model Number	
09	Manufacturer ID	
07	ASCII ”C” or ASCII “H”	43 or 48
05	ASCII ”A”	41
03	ASCII ”P”	50
01	ASCII ”I”	49

The fixed data space contains twelve bytes.

The low four bytes contain the ASCII text "IPAC” for 8 MHz or “IPAH.” “IPAC” is used for 8 MHz only design; “IPAH” is used on Modules that run at 32 MHz. These four ASCII bytes at the base of the ID space identify the Module as installed, and containing a Format I ID PROM.

Table 4 assumes big endian byte ordering. Thus the consecutive bytes of the ID PROM are on the low byte of the data bus and occupy consecutive ODD byte addresses. Other architectures—while fully compatible with IP Modules—may use a different host addressing scheme. For example little endian architecture machines will place these bytes locations on consecutive EVEN byte addresses. Some 32-bit NuBus implementations, for example, might place these bytes on every fourth byte address.

Byte 09 of Format I is the Manufacturer ID. Up until this Specification is approved by VSO, these numbers are assigned by GreenSpring Computers, Inc, at no charge. Requests should be made in writing, delivered via mail or fax, and include:

- Contact Name
- Company name and address
- Phone and fax
- Indication if product(s) are for internal use only or will be sold on the open market
- Indication if the products are Modules or Carrier boards.
- Optionally, a short description of the product may be included.

GreenSpring may be contacted at:

GreenSpring Computers, Inc.
1204 O’Brien Drive
Menlo Park CA 94025
voice: 415.327.1200
fax: 415.327.3808

When this Specification is approved by the VSO, the VSO or its designate shall assign Manufacturer ID numbers. For contacting VSO, see the front of this document.

8-bit Manufacturer ID numbers FF and 00 are never used. Numbers in the range 01 to 0F may be used by anyone as "unregistered" or for prototype IP Modules.

The next two bytes, Model Number and Revision, are set by the manufacturer. Byte 0F is reserved for future extension. It is set to 00.

RECOMMENDATION 9-22: Interpretation of the Revision field is up to each individual manufacturer. A recommended format is to start with (hexadecimal) $A1. The first hex letter increments for major product revisions that impact the product's specifications or performance. The digit increments for minor revisions.

OBSERVATION 9-23: The Revision field could be used by systems integrators for configuration management. By comparing against a minimum revision level or a known fixed revision it is possible for the system software to assure that only compatible Modules will be installed during maintenance or system upgrades.

The next two bytes contain a sixteen bit field reserved to identify an appropriate software driver. The number 0000 means no driver is specified. The interpretation of this field is beyond the scope of this Specification.

The byte at address $15 in Table 4 contains the number of bytes used in the ID PROM for fixed data and IP Module specific data. If no IP Module specific data is used then this number is (hex) $0C, for the (decimal) twelve bytes of fixed data. The first byte in the ID PROM that is free for user data is at address 2*nn+1, where nn is the number in the byte at address 15. There is no requirement that any user space be provided.

An 8-bit CRC checksum is used in byte $17 to verify that the information in the ID PROM is being read correctly. This CRC covers only the number of bytes set by the manufacturer: this number is at address $15 (discussed in the previous paragraph). The CRC used is an industry standard. It is the low eight bits of the FCS as described in CCITT T.30 (Fascicle VII.3) section 5.3.7. Byte $17 is first set to zero for the generation of the CRC.

Sample programs to generate and check the CRC may be found in the Appendix to this Specification. These programs are not validated and are provided as examples only.

9.2. ID PROM Data Format II

Table 5 ID PROM Data Format II

Address	Description	Contents
3E nn+2	User Space	
nn 20	IP Module Specific Space	
18	16-bit CRC	
16	Number of bytes used, 16-bit quantity	= nn
14	16-bit Flags	
12	Reserved for Driver ID, high word	
10	Reserved for Driver ID, low word	
0E	Reserved	0000
0C	Revision, 2 bytes	
0A	Model Number, 2 bytes	
08	Manufacturer ID, low 16 bits of 24	yyyy
06	Manufacturer ID, high 8 bits of 24	00yy
04	ASCII “4 <space>”	3420
02	ASCII “TA”	5441
00	ASCII ”VI”	5649

The fixed data space contains twelve 16-bit words.

The low three words contain the ASCII text "VITA4 .” Note that the ”V” is in the high byte; the “I” is in the low byte. These six ASCII characters at the base of the ID space identify the Module as installed, and containing a Format II ID PROM.

The two words at addresses 06 and 08 contain a 24-bit IEEE Manufacturer ID number. The high eight bits in the ID PROM are filled with zero. Note that this Manufacturer ID number is different than the one contained in Type I format ID PROMs. These numbers were originally assigned by IEEE to form part of a 48-bit unique node identifier “MAC address” as specified in IEEE Std 802.3 (ISO / IEC 8802-3) Section 3.2.3. IEEE calls this number an “Organizationally Unique Identifier.”

IEEE may be contacted at:

Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York NY 10017-2394

Organizationally Unique Identifiers may be obtained by contacting the IEEE Registration Authority at:

Institute of Electrical and Electronics Engineers, Inc.
Registration Authority, IEEE Standards Department
PO Box 1331
445 Hoes Lane
Piscataway NJ 08855-1331
voice: 908.562.3813
fax: 908.562.1571

At the time of this writing, there was a US$1000 fee for each Organizationally Unique Identifier. See also IEEE Std 802-1990.

The next two words, Model Number and Revision, are defined and set by the manufacturer.

RECOMMENDATION 9-24: Interpretation of the 16-bit Revision field is up to each individual manufacturer. A recommended format is to start with ASCII A1 (hex $4131). The first ASCII letter increments for major product revisions that impact the product's specifications or performance. The ASCII digit increments for minor revisions.

OBSERVATION 9-25: The Revision field could be used by systems integrators for configuration management. By comparing against a minimum revision level or a known fixed revision it is possible for the system software to assure that only compatible Modules will be installed during maintenance or system upgrades.

The 16-bit reserved field should be set to 0000.

The next two words contain a 32-bit field reserved to identify an appropriate software driver. The number 0000 0000 means no driver is specified. The interpretation of this field is beyond the scope of this Specification.

The next 16-bits are reserved for Flags. The bits are currently defined:

- D0 = 0, identifies this revision of the specification for the ID PROM,
- D1 = 1 for 8 MHz functionality supported, D1 = 0 for 8 MHz operation not recommended.
- D2=1 for 32 MHz functionality supported, D2 = 0 for 32 MHz operation not supported or not recommended.
- Bits D3..D15 must be set to zero.

Note that the ID PROM itself must always be readable at 8 MHz. D1 = 0 means that the manufacturer does not recommend that the Module be operated at 8 MHz for its intended function. D1=1 means that the module is specified at 8 MHz.

A Module may be designed for 8 MHz operation only, or for both 8 and 32 MHz operation, or primarily for 32 MHz operation.

The word at address $16 in Table 5 contains the number of bytes used in the ID PROM for fixed data and IP Module specific data. If no IP Module specific data is used then this number is (hex) $1A, for the (decimal) 26 bytes of fixed data, including the CRC. The first word in the ID PROM that is free for user data is at address nn. where nn is the number in the word at address $16. There is no requirement that any user space be provided.

An 16-bit CRC checksum is used in word $18 to verify that the information in the ID PROM is being read correctly. This CRC covers only the number of bytes set by the manufacturer: this number is at address $16 (discussed in the previous paragraph). The CRC used is an industry standard. It is the low 16 bits of the FCS as described in CCITT T.30 (Fascicle VII.3) section 5.3.7. Word $18 is first set to zero for the generation of the CRC.

Sample programs to generate and check the CRC may be found in the Appendix to this Specification. These programs are not validated and are provided as examples only.

10. 8 MHz Data Transfer Cycle Details

This section describes basic cycle operation between the Carrier board and the IP Module. This section applies specifically to 8 MHz operation. See following sections for 32 MHz operation. Since 32 MHz operation is described in this document as a variation on the basic 8 MHz operation, this section is still applicable to 32 MHz operation, with the modifications described in the following sections.

All data transfers are performed synchronously to the 8 MHz CLK. Signals are changed following the rising edge of CLK. Interface signals are typically sampled on the rising edge of CLK.

The IP Module generates signals back to the Carrier board each cycle: Ack* and, for read cycles, Data. For DMA cycles, the signals DMAReqx* and DMAEnd* are also generated valid each cycle. For interrupts, the signals IntReqx* are generated by the IP. However these two interrupt requests, unlike all of other signals, may be generated and removed asynchronous to the clock.

The fastest and simplest data transfer cycle uses two clock periods. During the first clock period the Carrier board asserts one of four *select* lines (IOSel*, MemSel*, IntSel*, IDSel*) and drives the address bus valid. During the second clock period, the IP Module asserts Ack* to *terminate* the cycle. Data is nominally transferred at the end of this clock period. The basic cycle may be extended by the IP Module by inserting *wait* states between the select and termination clock cycles. On reads, the Carrier board may extend the cycle—causing data from the IP Module to be held valid—by inserting *hold* states. If no cycle is in progress the clock cycles are identified as *idle* states.

Typical data transfer cycles are shown in Figure 2 through Figure 9. The states are identified in smaller type above each clock period.

It is convenient to associate each clock cycle (clock period) with one of five *states: select, terminate, wait, hold* or *idle*. These are printed in italics is this section, and are used in the timing and state diagram figures near the end of this Specification.

The two states *select* and *terminate* are required for all data transfers. These two states are always one clock cycle long each. *Wait, hold* and *idle* states are optional.

See Figure 11, which shows the Data Transfer State Diagram.

The *idle* state is exited when the Carrier board asserts one of the four select lines. The select line may be asserted for as little as one clock cycle.

Any number of *wait* cycles may be inserted between the *select* cycle and the *terminate* cycle by the IP Module by not asserting Ack*. There is no explicit provision for cycle timeout. Even if an IP Module supports a specific select line it is not required to always assert Ack* in response to that select line. (If an IP Module is not driving an interrupt request it does not have to respond to IntSel*, for example.) Conversely, an invalid address to an IP Module does not require that IP Module to fail to assert Ack*.

There is no fixed maximum amount of time by which an IP Module must assert Ack*. However the IP Module literature must clearly state access time, or "wait states." In some cases the number of wait states may be a function of external events, in which case a maximum time must be stated.

RECOMMENDATION 10-26: IP Modules' maximum Ack* time should generally be not be more than a few microseconds.

If a Carrier board wishes to force an IP Module to prematurely terminate a cycle the Carrier board must assert Reset*.

The Carrier board may extend any read or write cycle by holding the select line asserted. There is no specified maximum number of *hold* states. If an IP Module requires a limit to the number of *hold* states in order for it to function properly, it should state that limit in its literature.

RECOMMENDATION 10-27: Hold states should not exceed four microseconds (32 hold states).

The Carrier board does not have to hold data valid on write cycles on any clocks past the *terminate* state. That is, the Carrier board may hold the select line true after Ack*, for its convenience, introducing hold states. The IP Module acknowledges these hold states by holding Ack* true; however the internal write cycle on the IP Module has been completed, and so data is ignored.

OBSERVATION 10-28: A single data transfer cycle may have either *wait* states, or *hold* states, or both, or neither.

11. 32-bit Access

IP Module may be accessed using 32-bit wide data if the IP Module is a double-size.

32-bit double-size IP Modules are fully compatible with the remainder of this specification. Such an IP Module is mechanically a double-size, thus using two Logic Interface connectors. The “A” connector is the low 16-bits, and the “B” connector is the upper 16-bits. The IP Module may be compared to two independent 16-bit IP Modules side by side. Double-size IP Module mechanical dimensions are shown in Figure 13.

OBSERVATION 11-29: Note that the Carrier board may not bus the data lines in this case.

OBSERVATION 11-30: The Carrier may easily support two 16-bit IP Modules or one 32-bit IP Module in the same physical space with minimal mechanical or electrical impact (more bus transceivers will generally be needed to support 32-bit operation, however). The logic on the Carrier would enable bus drivers appropriately, typically using either data width information from the host, or derived information from the host-driven address, or bits from a mode register in order to make the 16/32 bit width decision.

Access to the 32-bit mode consists of the Carrier asserting two matched select lines simultaneously (for example both IOSel* or both MemSel*). The Carrier must wait until both A and B ACK* are asserted simultaneously by the IP Module before terminating the cycle.

This simple protocol guarantees that there is no mistake between 16 and 32-bit accesses, yet maintains transparent compatibility. Should a Carrier attempt a 16-bit access to a 32-bit IP Module, it will respond correctly—enabling only 16 data lines. Should a Carrier attempt a 32-bit access to a double-size 16-bit IP Module, then the access will never complete, because the IP Module will drive only one of the two ACK* lines.

32-bit IP Modules that are in the memory space must be able to perform 8, 16 and 32-bit accesses in any aligned combination. 32-bit IP Modules in the I/O space may limit access to specific widths.

12. 8 MHz Timing Details

This section describes timing details of 8 MHz operation. See the following section for 32 MHz operation.

All signals used on the IP Module Logic Interface are measured relative to the rising edge of 8 MHz CLK. To minimize skew between the Carrier board and the IP Modules the CLK must be driven by a high current driver that meets both TTL and CMOS level specifications. The CLK signal should be terminated on the Carrier board near the IP Modules.

See Figure 12 for the Timing Diagram.

Ack* is held asserted for one clock cycle minimum, and is both asserted and deasserted within the time period specified from the rising clock edge. Ack* must be asserted during *hold* cycles as a hold acknowledge. Ack* must then be negated when the select line is detected deasserted on the next rising clock edge.

Memory cycles use the data bus to transmit the high order address lines (A7..A22) to the IP Module on the first clock cycle. On both read and write cycles the data bus is driven during the *select* state with address. Wait states inserted by the IP Module do not extend the address drive. For read cycles, data is valid at the end of the *termination* state. For write cycles, the data is valid during the termination state and during any wait states.

No "strobe" lines are used on the IP Module logic interface, in the sense that there are asynchronous strobe lines on the VMEbus. On output cycles IOSel*, low address (A1..A6), data and optionally the two byte selects are driven simultaneously. On memory write cycles MemSel*, low and high address (A1..A22), and the two byte selects are driven simultaneously on the first clock, followed by valid data on the remaining clock cycles. The byte selects, like the low address lines, must remain valid until the end of the termination and hold cycles.

RECOMMENDATION 12-31: It is recommended that Carrier boards not drive irrelevant information on the data and address lines between valid data transfers. However, during idle states, the data, address, R/W*, and byte selects are "don't cares." This means that these lines may be setup asynchronously ahead of the synchronous select line, and also released asynchronously after the select line is released, or after the terminate state.

Signal lines which must be driven by the Carrier board during select and wait states must also be driven during hold states. That is, if the Carrier board holds a select line, it must also hold relevant write, byte select and address lines.

OBSERVATION 12-32: Designers of IP Modules should note that there is no sequencing requirement, either asynchronous or synchronous for deasserting signals at the first idle state. For example, within one IP Module clock cycle the write line may change state before the select line. If these lines are directly connected to an LSI component its timing requirements may be violated. Similarly, address lines may change while write or select is asserted. LSI components that implement a function (such as clearing an interrupt bit) as a side effect of a register read require particular attention. In some cases it may be necessary to add a register or latch in the address path on the IP Module.

RECOMMENDATION 12-33: This specification currently requires that only one of the four select lines, MemSel*, IOSel*, IntSel*, or IDSel* is active at once. However, extensions have been proposed that use two select lines in combination.

Therefore it is recommended that IP Modules currently in design do not respond when more than one select line is asserted.

13. 32 MHz Operation

IP Modules are suitable for high speed I/O applications. This section defines a high speed mode of 32 MHz.

The major differences of 32 MHz high speed operation from basic 8 MHz operation are:

- Clock speed is increased to 32 MHz;
- Bus Turnaround states are added;
- Timing is more restrictive;
- Bus loading is more restrictive;
- Hold states are eliminated;
- Data on I/O write cycles is not valid during Select;
- ID PROM indicates the speed change.

Detailed timing requirement, cycle timing diagrams and state diagrams for 32 MHz operation are provided in the Figures section near the end of this document.

Rising clock to high impedance maximum time is 17 nsec.

Rising clock to turn-on plus data-valid is the same as for data valid (17 nsec).

The 32 MHz CLK is a 50% nominal duty cycle signal. The frequency accuracy is 0.1%. Rise and fall time shall not exceed three nanoseconds, as measured by Test Condition 1.

The state machine for 32 MHz operation is different than for eight MHz operation to permit a bus Turnaround state. This new state—which is an enforced idle state—occurs:

- during memory cycles, following the Select state;
- between a read cycle and a memory cycle.

A maximum number of permitted wait states by an IP Module at 32 MHz is 127 clock cycles.

High speed IP Modules should be identified in their literature with the phrase, “32-MHz IP Module.” Such IP Modules must permit access to their ID PROM at the standard eight MHz clock. Ideally, all but the relevant high speed portions of the IP Module would also work at the 8 MHz clock.

32-MHz IP Modules must indicate their clock speed capability in their ID PROM. See the ID PROM section in this document for more details.

14. Test Conditions

This section describes test conditions for use by board and system designers and integrators to assure that products and systems are compatible with this specification. Due to the relatively tight timing requirements for 32 MHz operation (compared to 8 MHz operation) it is appropriate to specify standard measurement conditions.

14.1. Standard Test Conditions

The Test Conditions listed in this subsection apply to all measurements of both 8 MHz and 32 MHz operation, unless otherwise stated.

14.1.1. **500 MHz oscilloscope and probes is assumed. The use of faster or slower measurement equipment may provide usable results subject to suitable interpretation.**

14.1.2. **All measurements are to be made on the IP Module connector pins. This Condition also applies to the ground reference.**

14.1.3. **Supply voltages during measurements shall be within 1% of the nominal voltage, measured with a DC meter at the IP Module connector pins.**

14.1.4. **Voltage thresholds for driven signals (from the Module or Carrier under Test) shall be 0.4 volts maximum logic low; 2.4 volts minimum logic high. Voltage thresholds for received signals (to the Module or Carrier under Test) shall be 0.8 volts minimum logic low; 2.0 volts maximum logic high.**

14.1.5. **Threshold voltage for the rising edge of CLK shall be at 1.5 volts.**

14.2. 32 MHz Specific Test Conditions

The Test Conditions listed in this subsection apply to specific measurements of 32 MHz operation, as stated.

14.2.1. **This test condition is for measuring worst case maximum data valid delays. The Carrier shall be fully populated with one Module under Test and the remaining IP Module slots filled with standard dummy modules as specified in 14.2.2. For testing Modules, a Carrier board with four total IP Module slots should be used.**

14.2.2. **A standard dummy load module shall consist of a single-size IP Module that provides 24 pF of capacitive loading on all lines, plus a 330Ω resistor to 1.5 volts on all appropriate signal lines. The resistor load may not be appropriate on lines that are not driven in the test setup and could produce invalid levels on these lines.**

14.2.3. **This test condition is for measuring worst case minimum data hold times. For Module testing, the Carrier shall be populated only with the Module under Test. For Carrier board testing, only a single module, selected for its minimum necessary loading, shall be used.**

14.2.4. **Maximum driver turn-off times should be measured using Test Condition 14.2.1. The driver may assumed to be off when a valid logic high passes in the negative direction through 2.0 volts, or when a valid logic low signal passes in the positive direction through 0.8 volts, whichever time is longer if both logic levels are applicable. The line being measured must have a resistive load connected for all 14.2.2 Modules. If noise is observed on the**

signal line, the last such transition through the specified voltage shall be used.

15. Pin Assignment

The pin spacing in the connector is two rows, 0.100 inch apart, with the pin to pin spacing within the row as 0.050 inches. The hole pattern in the printed circuit boards is four rows, 0.075 apart, with hole spacing within the row of 0.100. Pins are numbered in a similar format as a conventional D connector. A four layer printed circuit board using 12 mil trace/12 mil gap design rules normally has no trouble wiring these connectors.

Table 6 shows the correspondence of signal names to pin numbers.

Table 6 Pin Assignment of Signal Names

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
1	GND	2	CLK	26	GND	27	+ 5V
3	Reset*	4	D0	28	R/W*	29	IDSel*
5	D1	6	D2	30	DMAReq0*	31	MemSel*
7	D3	8	D4	32	DMAReq1*	33	IntSel*
9	D5	10	D6	34	DMAck*	35	IOSel*
11	D7	12	D8	36	Reserved	37	A1
13	D9	14	D10	38	DMAEnd*	39	A2
15	D11	16	D12	40	Error*	41	A3
17	D13	18	D14	42	IntReq0*	43	A4
19	D15	20	BS0*	44	IntReq1*	45	A5
21	BS1*	22	- 12V	46	Strobe*	47	A6
23	+12V	24	+ 5V	48	Ack*	49	Reserved
25	GND			50	GND		

An acceptable 50 pin connector is made by AMP, Inc. The shrouded header, AMP part no 173280-3, or equivalent, is used on the Carrier board. The plug, AMP part no 173279-3, or equivalent is used on the IP Modules. This same connector pair is used for the I/O interconnection. The header and plug have a molded shell in the shape of a "D", which effectively makes the connectors keyed against improper insertion. The pins and sockets are gold plated, rated at one Ampere per pin, and specified for a minimum of 200 insertion cycles. An acceptable connector is fully defined in AMP Catalog 87-797, Product Specification 108-5203, Instruction Sheet IS 292J and Test Report PRP No. 8445.

AMP Incorporated
Harrisburg, PA 17105.
voice: 717.564.0100.

AMP is a trademark of AMP, Inc.

Compatible connectors are also made by Taiconn, their Series 0131-50SB As of this writing, both manufacturers were considering manufacturing compatible surface mount connectors.

Taiconn U.S.A.
20400 Plummer Street
Northridge CA 91324
voice: 818.993.1138
fax: 818.993.1153

16. Electrical Interface

There are two different electrical interface specifications for IP Modules. There is one specification for 8 MHz Modules, and a second for 32 MHz Modules. Since all 32 MHz Modules must support at least ID PROM access at 8 MHz, the 32 MHz electrical specification is a more restrictive subset of the 8 MHz specification.

Modules that meet the pin loading requirements of the 32 MHz specification, but only operate at 8 MHz, may be identified as “Mixed speed bus compatible.” They may be used on Carriers that bus some of the Module’s logic signals, but support different clock rates to different modules.

The IP Module interface is specified for CMOS voltage levels and drive characteristics. Specifically, the logic family 74HCTxxx is the original basis for this specification. As of the writing of this revision, there is no universally accepted CMOS level and drive standard. (CMOS "families" include FCT, VCT, ACT, FACT, plus CMOS buffers and logic with a range of levels and edge speeds.)

16.1. 8 MHz Loading

Loading for one standard 8 MHz IP Module is defined by the following:

- CLK: 6.0 mA (logic low) maximum DC current in parallel with 30 pF
- All other signals: 3.0 mA (logic low) maximum DC current in parallel with 30 pF
- For DMAEnd* signal, see specifications under the following subsection, “32 MHz Loading.”

These loads have been selected to avoid the use of buffers on most 8 MHz IP Modules. The capacitive loading represents six typical CMOS or bipolar inputs, or alternatively two high current bi-directional buffers plus one standard input. The current loading for all signals except the CLK represents 12 low power bipolar inputs, eight "F" series parts inputs, or four "F" series enable inputs.

Standard CMOS or low power bipolar components on the Carrier board can drive four standard IP Modules. (A 24 mA driver is required for the CLK line.) The timing specifications also assume a standard component (not a bus driver) driving four IP Module loads. Skew, rise and fall times are taken into account.

Ack* may be driven by one standard 74HCTxxx, 74LSxxx, 74ALSxxx, 74Fxxx gate, or one low power bipolar or CMOS PLD (16V8, 22V10 typical) output, or compatible device.

All signals have a 10K ohm pullup resistor on the Carrier board, unless they are continuously driven by the Carrier board with CMOS logic. This assists in assuring full compatibility between TTL and CMOS logic.

16.2. 32 MHz Loading

Timing at 32 MHz is considerably less forgiving than at 8 MHz.

Loading on the Module is defined by the following:

- All logic signals: 3.0 mA (logic low) maximum DC current in parallel with 20 pF

Loading on the Carrier is defined by the following:

- All logic signals: 3.0 mA (logic low) maximum DC current in parallel with 20 pF

The signal DMAEnd* has special requirements that apply equally to Carrier boards and Modules:

- DMAEnd*: 12 pF maximum capacitive loading in parallel with a 1.2 KΩ pullup resistor to +5 volts.
- DMAEnd*: Driver must be able to provide 24 mA of sink current at 0.4 volts. This signal is driven open-drain.
- DMAEnd*: Do not connect to this signal line unless you receive or drive this line.

All signals except DMAEnd* have the following drive requirement on both the Carrier board and Modules:

- Sink Current: 15 mA minimum at 0.4 volts maximum.
- Source Current: 5 mA at 2.4 volts minimum.

In addition to the drive requirements above, Carrier Boards and Modules must be able to meet the timing requirements under the loading and test conditions contained in this Specification.

All drivers and receivers are assumed to be CMOS, with "TTL" switching thresholds, unless otherwise stated.

A Carrier may drive a maximum of four IP Modules on one Logic Interface Bus.

The above loading specification means that the maximum load on any signal line shall be 100 pF (20 pF per module, plus 20 for the Carrier board), plus additional capacitive loading from trace wiring and connectors. The maximum sink current for a logic low shall be 15 mA.

All signals have a 10K ohm pullup resistor on the Carrier board (except DMAEnd*), unless they are continuously driven by the Carrier board with CMOS logic. This assures against any floating CMOS input lines.

OBSERVATION 16-34: A 20 pF load corresponds to two pins, typically. IC pins vary from 8 pF to 12 pF for many common ICs. The lower load is typical of bus buffer inputs. The higher load is typical of PLD pins that can be both inputs and outputs.

16.3. Timing Measurements

To assure a consistent platform for timing measurements, timing measurements are specified using the test conditions in Section 14.

16.4. Current Loading

The maximum current that an IP Module may draw from the Carrier Board is limited by the specifications of the connector, which is one amp per pin.

The Table below shows the maximum current limits permitted.

Note that regulatory compliance, over-current protection, power and heat management, available power from the Carrier Board, and I/O power limits are issues beyond the scope of this Specification.

Table 7 Current Loading Limits of IP Modules

Voltage	Limit
+5 V	two amp maximum
+12 V	one amp maximum
–12 V	one amp maximum
Ground Return	two amp maximum aggregate

17. Physical Dimensions

See Figure 23 for basic mechanical dimensions of a single-size IP Module. Figure 24 shows connector mounting and hole details. Figure 25 shows pin numbering on the connectors. Figure 26 shows double-size mechanical dimensions.

In all four figures pin 1 is indicated by a square.

The dimensions of a single-size IP Module are 1.800 inches by 3.900 inches. These dimensions require +.000 / –.020 inches tolerance.

The dimensions of a double-size IP Module are 3.600 inches by 3.900 inches. The form factor is identical to two single-size IP Modules one touching the other.

Each single-size IP Module has two AMP plug connectors, part no. 173279-3, 50 pins each, or an equivalent connector. One is for the Logic Interface, defined in this document. The other is for IP Module dependent I/O—typically routed to a 50 pin flat cable header on the Carrier board. The I/O connector may be missing from an IP Module only if an equivalent mounting mechanism is provided.

For more information about acceptable connectors and manufacturers, see Section 15, Pin Assignment.

Double-size IP Modules may have two, three or four connectors, as required by the application. If two logic connectors are used, they both must conform to this specification as if two separate single-size IP Modules were installed.

RECOMMENDATION 17-35: It is recommended that double IP Modules should use the "A" logic side if they are going to use only one of the two logic interfaces. Double IP Modules need have only a single ID PROM if desired, but if so, it must be on the "A" logic interface. If fewer than four connectors are used, the IP Module must provide for alternative equivalent mechanical mounting that is compatible with the IP Module mechanical specifications.

The interboard spacing between the Carrier board and the IP Module is 0.472 inches. This space is allocated 0.180 to the Carrier board and 0.290 to the IP Module.

OBSERVATION 17-36: The IP Module is mounted on the Carrier board with the components facing the Carrier board. Normally the pins of IP Module components are trimmed flush to the IP Module board, and a label is affixed providing information about the IP Module.

RECOMMENDATION 17-37: It is recommended that the top-most portions of components mounted on IP Modules, and the top-most portion of all components mounted on Carrier boards, not have a conductive surface. Examples of components that may need special consideration, such as a coating or a sleeve, are some ceramic PGA ICs, stand-up axial discretes (such as resistors and capacitors), and oscillators in metal cases. This recommendation is to prevent shorts between components on the Carrier board and components on the IP Modules. Such a short could occur if a board or component were out of specification, or under other adverse conditions.

This specification assumes that 0.062 inch thick printed circuit material is used for the IP Module. Thicker material is not permitted. If thinner material is used, it is the manufacturer's responsibility to clearly document these differences, which may include lessened capability to withstand vibration and shock, or may require special mounting hardware.

RECOMMENDATION 17-38: Because of the close proximity of components on the Carrier boards and the IP Modules the use of only low power components is strongly recommended. Standard power bipolar PLDs, most bipolar analog components, and TTL bus transceivers are specifically too hot for typical mounting on or under IP Modules.

OBSERVATION 17-39: Note that air flow on VMEbus boards is typically "vertical," or along the axis of the two connectors. Positioning components to permit maximum airflow is recommended. Using surface mount components or other techniques to reduce the height of components on both the Carrier boards and the IP Modules is encouraged.

OBSERVATION 17-40: Note that if components are mounted at the maximum height permitted on both the IP Module and the Carrier board that all air flow is blocked.

IP Module and Carrier board manufacturers must provide holes and clearance to permit M2 screw and nut mounting of Modules to the Carrier boards. The manufacturer of the Carrier board must provide a clear zone around the screw hole where there are no traces or components that interfere with or could be damaged by the nut. This clearance should be 5.0 mm diameter plus any manufacturing tolerances.

RECOMMENDATION 17-41: IP Modules are optionally secured to the Carrier boards with screws and nuts. This option is available to the end-user or system integrator. IP Module and Carrier board manufacturers must provide holes and clearance to permit this type of mounting. The screws and nuts are metric M2 thread (2 mm diameter, 0.4 mm pitch thread). Details of mounting options are left to the manufacturers. Typical mounting consists of a flat-head stainless steel screw inserted through the IP Module. The IP Module is countersunk so that the head of the screw is at or below the surface of the IP Module's PCB. The nut is also stainless steel, attached from the solder-side of the Carrier board. The nut may be secured to the screw with the aid of an adhesive, such as Lok-Tite®. Plastic screws or nuts are not recommended because they are not strong enough. Manufacturers may provide an alternative mounting option, however it must be compatible with the screw and nut system described here to permit IP Modules and Carrier boards from multiple vendors to be securely mounted to each other. IP Modules that are so secured to the Carrier board should meet the mechanical requirements of the host bus, if any.

IP Modules are identified as one of three types, Type I, Type II or Type III. Type I is the oldest type. If a Module is not explicitly identified as one of these three Types then it must be Type I.

Type I Modules have no components on the back of the Module. These modules have 0.534 inches (.472 + .062) as a maximum height above the Carrier board, meeting, for example the VME bus mechanical specifications.

Type II Modules have components on the back, subject to the following limitation: no component may exceed 0.072 inches in height above the back of the board, and the board must be no thicker than 0.062 inches. Type II Modules must be clearly identified as Type II Modules by the manufacturer.

RECOMMENDATION 17-42: It is recommended that manufacturers use the "Type II" designation everywhere they use the "IP Module" term. For example, "IP Module, Type II."

RECOMMENDATION 17-43: It is recommended that Type II Modules have no components on the back with exposed metal above 0.055 inches above the surface of the board. Manufacturers of Type II Modules may wish to consider providing means to protect the components on the back of the Module from handling or physical damage. Type II Modules, when installed on Carrier boards, may not meet all host bus mechanical specifications.

Type III Modules have components on the back, without subject to height limitation. Type III Modules must be clearly identified as Type III Modules by the manufacturer.

RECOMMENDATION 17-44: It is recommended that manufacturers use the "Type III" designation everywhere they use the "IP Module" term. For example, "IP Module, Type III." Type III Modules will generally use two ore more host bus slots. This Type is useful for providing capabilities beyond the mechanical capabilities of Type I and Type II Modules, such as optical fiber interfacing.

18. Allowable Combinations of Features

18.1. Feature Combinations for Carrier Boards

This section lists permissible feature implementation options and combinations for IP Module Carrier Boards.

18.1.1. Number of Slots

Carrier Boards that support 8 MHz only operation may implement from one to six single-size IP Module positions. Carrier Boards that support 32 MHz operation may implement from one to four single-size IP Module positions. These may be any suitable combination of single-size slots or double-size slots, where double-size slots count as two positions. For more than these stated maximum number of positions, a second bus must be used.

OBSERVATION 18-45: The purpose of limiting 32 MHz carrier boards to four slots is to limit the total capacitive bus loading.

18.1.2. 32-bit Support

16-bit support is required. 32-bit support is optional.

18.1.3. Select Spaces

ID space and I/O space is required. Memory space and Vector space support is strongly recommended. ID space, I/O space and Vector space are fixed size. Memory space is variable size up to 4 Mwords (8 Mbytes) per single-size IP Module. Supporting the full memory space is preferred, however smaller memory spaces may be supported. Selected space support does not have to be identical for all Module positions, although this is recommended.

18.1.4. Addressing

Connecting An (where n is 1 to 22) on the IP Module interface to the same An on the host is preferred, but not required. Connecting Dn (where n is 0 to 15) to the same Dn on the host is strongly recommended. Variations are most likely in some 32-bit hosts and some 32-bit Module implementations. The preferred Byte Strobe line implementation is based on data byte lanes used, not on addressing. For example, both big-endian and little-endian systems should look to byte lane coding, not A0..A2 for Byte Strobe decoding.

Address connections when driving 32-bit IP Modules may shift the addresses one bit (i.e. connect A1 on the IP Module interface to the host's A2) to provide linear 32-bit data support.

18.1.5. Clock Speed

8 MHz support is required. 32 MHz support is optional.

18.1.6. DMA

DMA support is optional. I/O DMA only may be supported or both I/O DMA and Memory DMA support. Support for Memory DMA only is not permitted. DMA support may be for one channel or two channels per Module. If only one channel is supported, it must be Channel 0. Not all Modules slots on the Carrier are required to have identical levels of DMA support. Carrier Boards must respond to DMAEnd* if driven by a Module, but do not have to drive DMAEnd*, although this is recommended,

18.1.7. **Voltages**

Carrier boards must provide all three power voltages, on the pins specified.

18.1.8. **Reserved Lines**

Carrier boards must not use the reserved lines, although they may provide a passive pull-up resistor.

18.1.9. **Wait States**

Carrier boards must respond to Wait States. They do not need to generate Hold states.

18.1.10. **Bus Timeout**

Carrier boards do not need to implement a bus timeout, although this is strongly recommended.

18.1.11. **Reset**

Carrier boards must drive the Reset* line as specified on power-up, and optionally (and recommended) on host system reset. Reset* may also optionally be driven under program control.

18.1.12. **Mechanical**

Carrier boards must implement the mechanical requirements in this Specification. In particular, Carrier boards must provide sufficient space with no interference under the Module.

18.1.13. **Signal Definition**

Carrier boards must implement the signals as defined in this Specification.

18.1.14. **Timing Requirements**

Carrier boards must follow the timing specifications in this document.

18.1.15. **Labeling and Identification**

Manufacturers of Carrier boards must follow the labeling and identification requirements in this Specification. Only products that are fully compliant with this Specification may use the terminology, “IP Module.”

18.2. Feature Combinations for IP Modules.

This section lists permissible feature implementation options and combinations for IP Modules.

18.2.1. **Size**

IP Modules may be single-size or double-size. They may be Type I, Type II, or Type III. No other size or thickness is permitted.

18.2.2. **Select Spaces**

ID space is required. I/O space, Memory space and Vector space are optional, and may be implemented in any combination.

If only one interrupt request is used, it is recommended that it be IntReq0*.

18.2.3. **Data Width for Single-size Modules**

Single-size modules may be 8-bit or 16-bit. 8-bit Modules use D0..D7.

18.2.4. **Data Width for Double-size Modules**

Double-size Modules may be 8-bit, 16-bit or 32-bit. If they are 32-bit, then they use both the A and B side. If they are 8-bit or 16-bit, they may use the A side only, or both the A and B sides. If they use both the A and B sides for 8 and/or 16-bit operation, then the double-size module is performing identically to two independent single-size modules. A double-size module may implement different spaces as different widths.

If only a single ID space is implemented, it is always on the A side.

Interrupts are always implemented as if there were two independent single-size modules.

32-bit Modules must be explicitly clear in how address lines are interpreted.

18.2.5. **Byte Strobes**

Modules may ignore Byte Strobe lines for ID, I/O and Interrupts spaces. They may not ignore Byte Strobe lines for their Memory space.

18.2.6. **Address Decoding**

Modules are not required to fully decode memory lines, for any Space, beyond what is required to implement this Specification.

18.2.7. **Response**

Modules do not need to respond to cycles that are inappropriate.

Note that complex or unpredictable response can place a high burden on the end user or system integrator to assure reliable system operation under all situations.

Modules must be clearly documented with their longest response time, and an accurate and complete description of which spaces they respond to under what conditions.

18.2.8. **Clock Speed**

8 MHz support is required. The ID PROM is the only feature that is required to be operational at 8 MHz. 32 MHz support is optional. The manufacturer must specify, if 32 MHz is supported, what features and performance is available at 8 MHz.

18.2.9. **DMA**

DMA support is optional. I/O DMA and Memory DMA may be implemented independently, however if only DMA Memory is supported the Module may not claim to implement or support DMA.

If I/O DMA is implemented the Module must respond to DMAEnd*. Driving of DMAEnd* is optional.

One or two DMA channels may be implemented. If only one channel is implemented, it must be Channel 0.

18.2.10. **Hold States**

Modules operating at 8 MHz must respond to Hold States. They do not need to generate Wait states.

18.2.11. **Reset**

Modules must respond the Reset* line as specified, except in special cases such as system monitoring functions, or non-volatile memory, and in cases where it is appropriate that a local Reset **not** reset the Module. In cases where the Module does

not respond to the Reset* signal, an alternative method must be provided under program control to restore the state of the Module to a known, and/or idle state.

18.2.12. **Reserved Lines**

Modules must not use the reserved lines, although they may provide a passive pull-up resistor.

18.2.13. **Signal Definition**

Modules must implement the signals as defined in this Specification.

18.2.14. **Timing Requirements**

Modules must follow the timing specifications in this document.

18.2.15. **Mechanical**

Modules must implement the mechanical requirements in this Specification. In particular, for the defined sizes and types, the Module must not exceed the mechanical envelope.

18.2.16. **Labeling and Identification**

Manufacturers of IP Modules must follow the labeling and identification requirements in this Specification. Only products that are fully compliant with this Specification may use the terminology, “IP Module.”

19. Changes this Revision

19.1. Revision Change to 1.0.d

Changes from 1.0.c to 1.0.d include:

- Mechanical height limits of components on carrier boards and IP Modules changed from a recommendation to a requirement in Section 17.
- DMAck*, Section 2.11, timing clarified.
- DMA Operation, Section 7.1, unnecessary text deleted.
- Minor grammer corrections.

Changes are marked with a vertical bar to the left of changed paragraphs. The “Scope” section is not so marked, nor are the name changes, nor the State Diagrams.

19.2. Revision Change to 1.0.c

Changing from 1.0a or 1.0.b included:.

- Name change from “VITA-4 Module” to “IP Module”.
- Addition of “Scope” section at the beginning of the Standard.
- Section 16.4, Current Loading for IP Modules, was added.
- Measurement requirements were expanded to include 8 MHz operation as well as 32 MHz.
- State Diagrams were restructured to improve readability.
- Clarification of when Bus Turnaround States are needed.
- ID Select cycles were clarified to include both read and write.
- Unnecessary observations were removed from the 32-bit Access discussion and the 8 MHz Data Transfer Cycle discussion.
- Minor text editing.
- 1.0.c.1 corrected some typos and added Appendix B.

Changes are marked with a vertical bar to the left of changed paragraphs. The “Scope” section is not so marked, nor are the name changes, nor the State Diagrams.

19.3. Revision Change to 1.0.b

Changing from 1.0a to 1.0.b involved cleanup and input from reviewers. Changes are marked with a vertical bar to the left of changed paragraphs. All changes are summarized:

- 19.1.1 and following Observation rewritten to support 6 slots for 8 MHz and 4 slot limit only for 32 MHz operation.
- Minor editing in Abstract and Description.

19.4. Revision Change to 1.0.a

Changing from 0.7 to 1.0.a involved a great deal of rewrite and formatting changes for more formality. Because of this, it is not possible to mark paragraphs where changes were made. The only functional changes that were intended as a result of this major reformatting are listed below. Major changes are summarized:

- Sections and subsections were numbered and the order changed.
- Recommendations and Observations were broken out.
- Many more timing diagrams were added, including 32 MHz and DMA.
- A 32 MHz state diagram was added.
- Section 7 on DMA was rewritten for better clarity. Some timing changes were made on DMA operation.

- Section 9 on ID PROM data was expanded to support IEEE based 24-bit Company ID Numbers, using the new ID PROM format Type II.
- Section **Error! Reference source not found.** on 32 MHz cycle and timing details was added.
- Section 14 on test conditions was added.
- Types I, II, and III were formally recognized in the Mechanical Section 16.3.
- Section 18 on allowable feature combinations for Carriers and Modules was added.
- 32 MHz operation was changed, and timing specifications were added.
- All references to Modules now use the term, “IP Module,” or the shorter, “Module.”

Other, more minor changes to this revision include:

- Clarification of the use of the Strobe* line.
- Clarification of when specifications apply to single-size or both sizes of Modules.
- Comments about future revision to the Specification were removed.
- Removal of some unnecessary Observations.

19.5. Older Revision History

Revision 0.7 was created primarily to fix several relatively minor typographical errors in Revision 0.6. No major changes in IP Module functionality were made.

A summarized list of changes from Revision 0.6 to 0.7 are:

1. The heights given in Physical Dimension section of Revision 0.6 were not consistent. The maximum height of components on Carrier boards is 0.157 inches. The maximum height of components on IP Modules in 0.314 inches. The recommended maximum height for components on IP Modules is 0.290 inches. It is also recommended that the highest portion of components on both Carrier boards and IP Modules not be a conducting surface (metal).
2. Figure 3 did not show R/W* correctly. This is now correct.
3. Figure 4 did not show the address lines being driven valid as long as the Data. This is now shown correctly.
4. Figure 7 did not have the states labeled correctly. This is now corrected.
5. The two DMA Acknowledge lines are now a single DMA Acknowledge line. Selection of channel 0 or channel 1 acknowledge is indicated by address line A1, same as for interrupt acknowledge cycles. This change is shown in Table 1 and in the Signal Descriptions section. This change will not generally impact IP Modules that have only a single DMA channel, or no DMA.
6. The two Byte Strobe lines must be driven on all cycles. This change is in the Signal Descriptions section. This change will generally impact only Carrier boards that chose to not drive the BS0* and BS1* lines. The Special Functions that were also on the Byte Strobe lines have been eliminated.
7. +5STBY function has been removed. This change is in the Signal Descriptions section.
8. The ID Space has been shortened to 32 words from 64 words in order to support future serial EEPROM options. It is now possible to generate write as well as read cycles to the ID Space, although these are not currently recommended. This is discussed in the ID PROM section.

9. The changes from items 5 and 7 above cause two lines on the Logic Interface to become “reserved.” These lines must be driven to logic one by Carrier boards.

10. The “32-bit sequential” access mode, which was proposed but not defined in Revision 0.6, has been dropped. This change is reflected in the 32-bit Access section.

11. The definition of 32 MHz operation, although still not fully defined, has been enhanced. See section “32 MHz Operation.”

Added Rev 0.7.1

12. Details of screw mounting were added as the last paragraph in the Physical Dimensions section.

Figure 1 Two IP Modules Mounted on 3U VMEbus Carrier

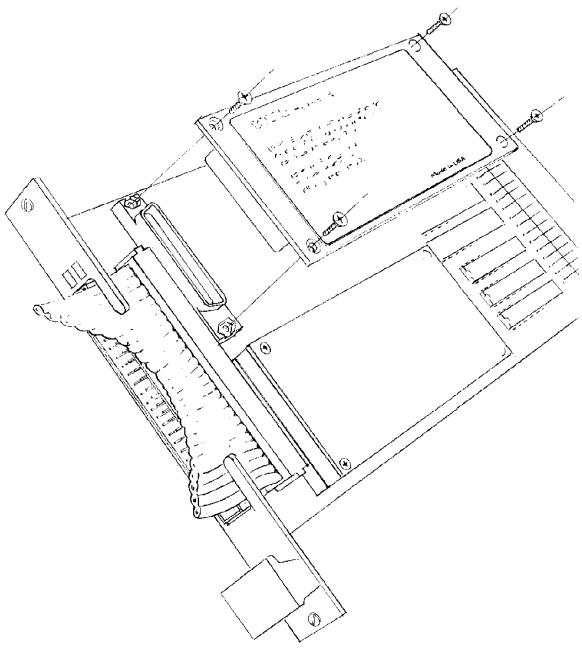
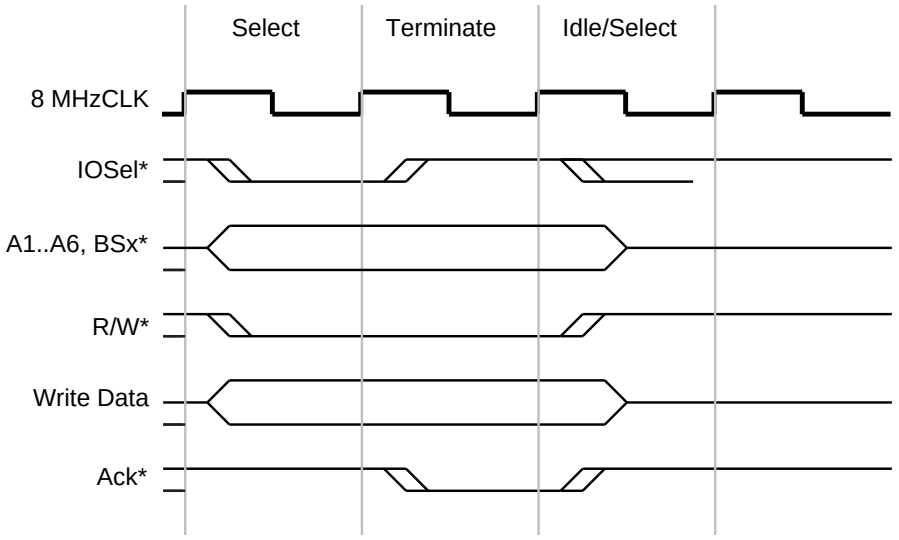


Figure 2 Fastest 8 MHz I/O Write Cycle



Note: For DMA timing, do not use this Figure.

Figure 3 Fastest 8 MHz I/O Read Cycle

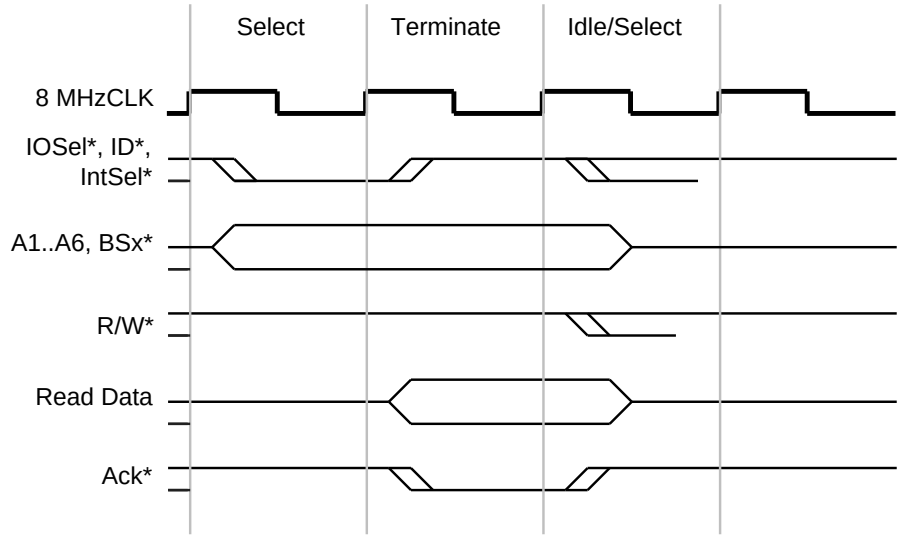


Figure 4 8 MHz Memory Read Cycle with Hold State

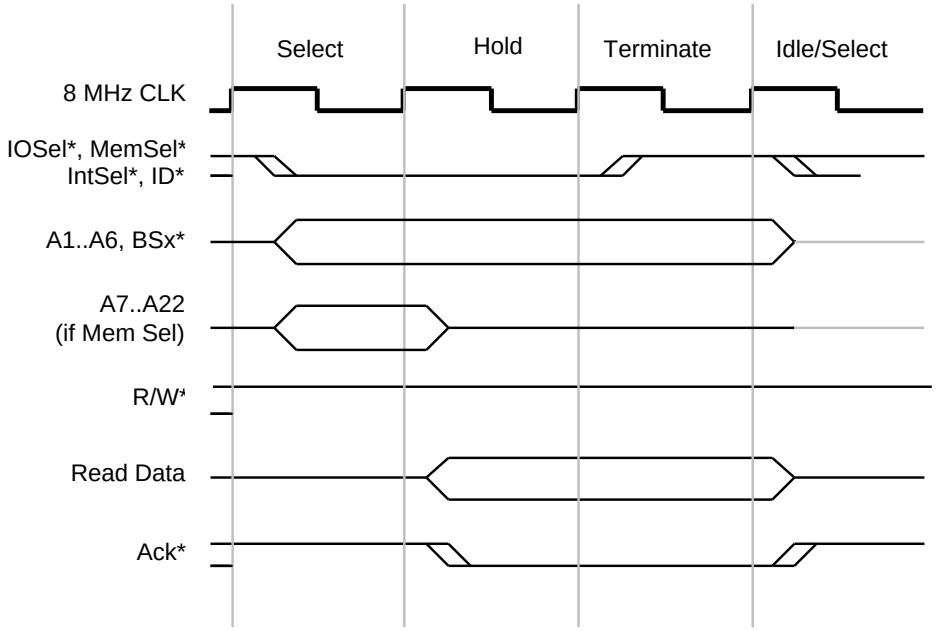


Figure 5 8 MHz Memory Read Cycle with Wait State

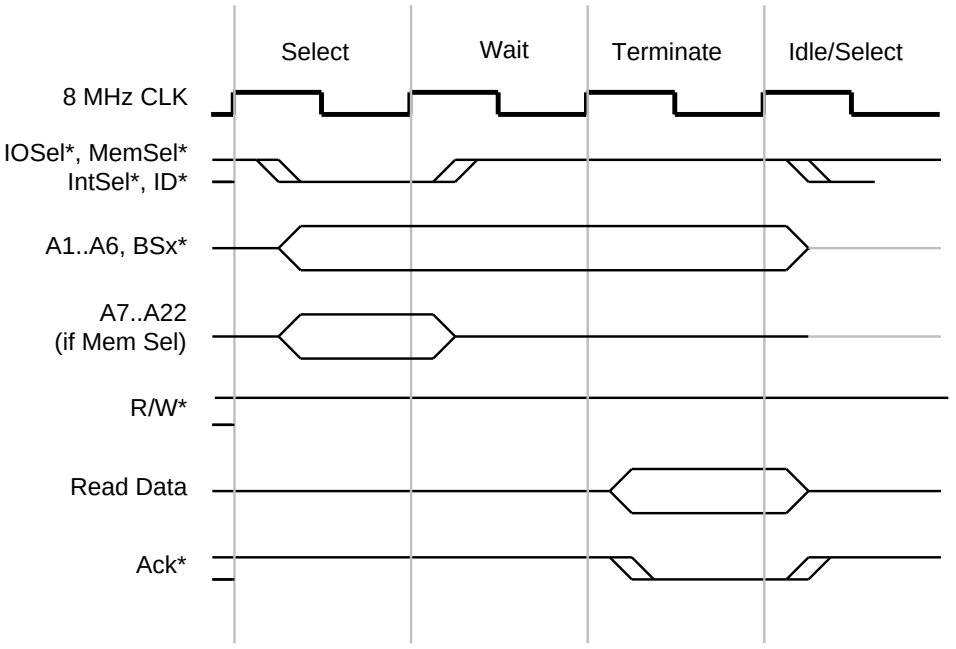
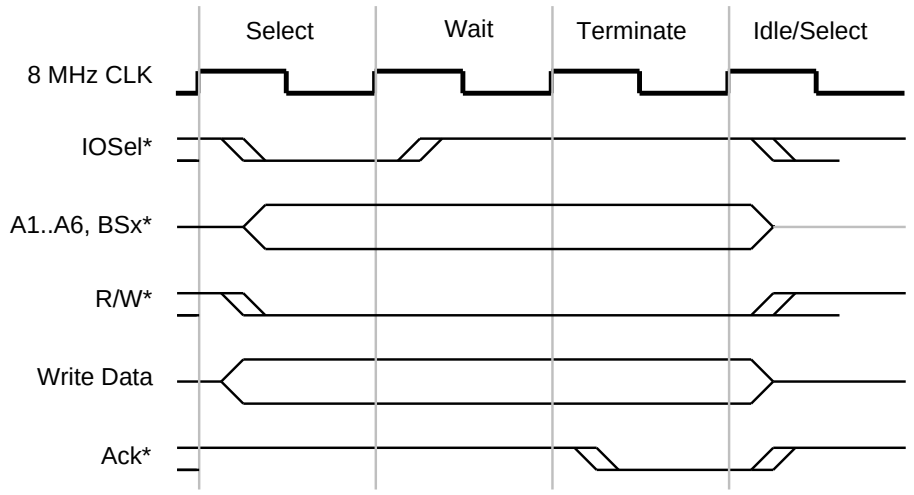


Figure 6 8 MHz I/O Write Cycle with Wait State



Note: For DMA timing, do not use this Figure.

Figure 7 8 MHz Memory Write Cycle with Wait and Hold State

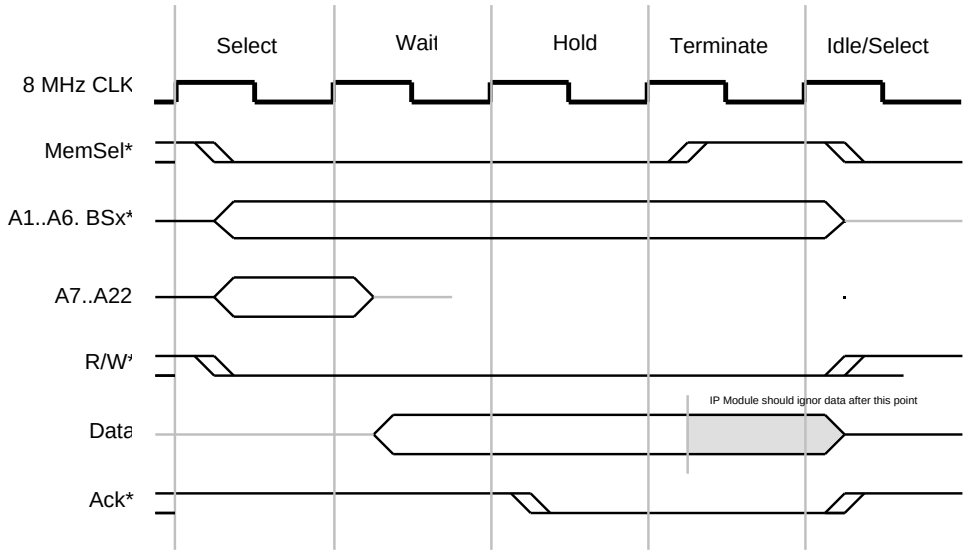
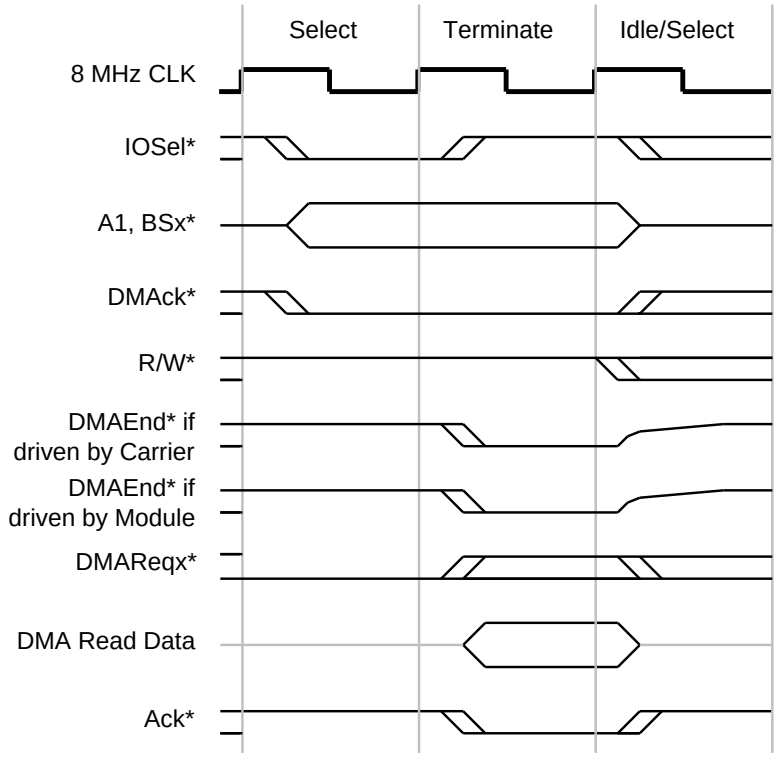
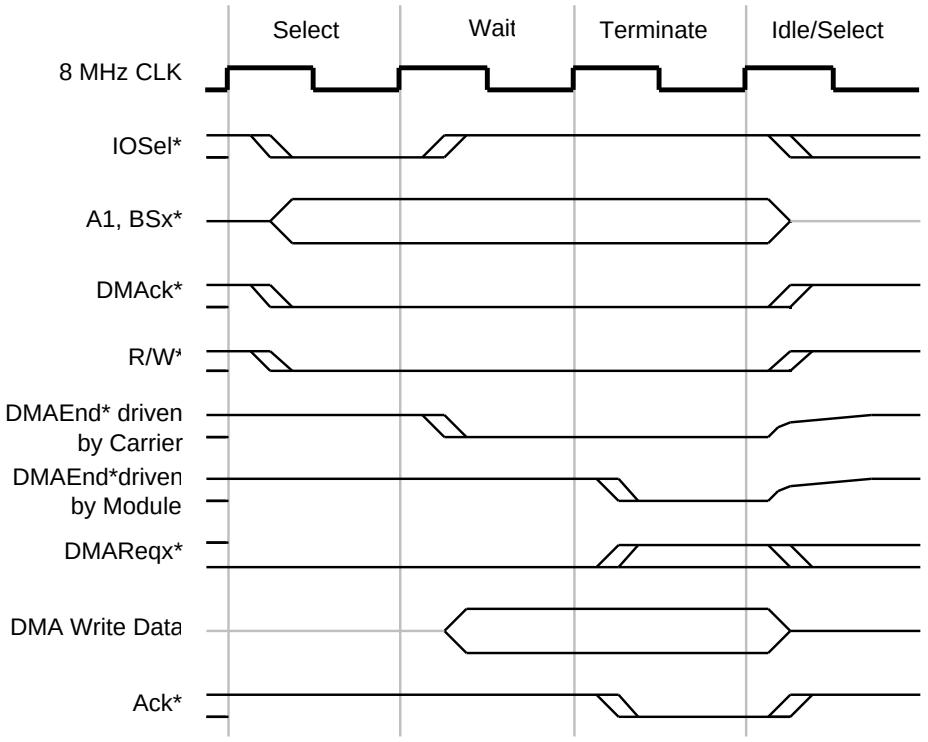


Figure 8 8 MHz DMA Read Cycle



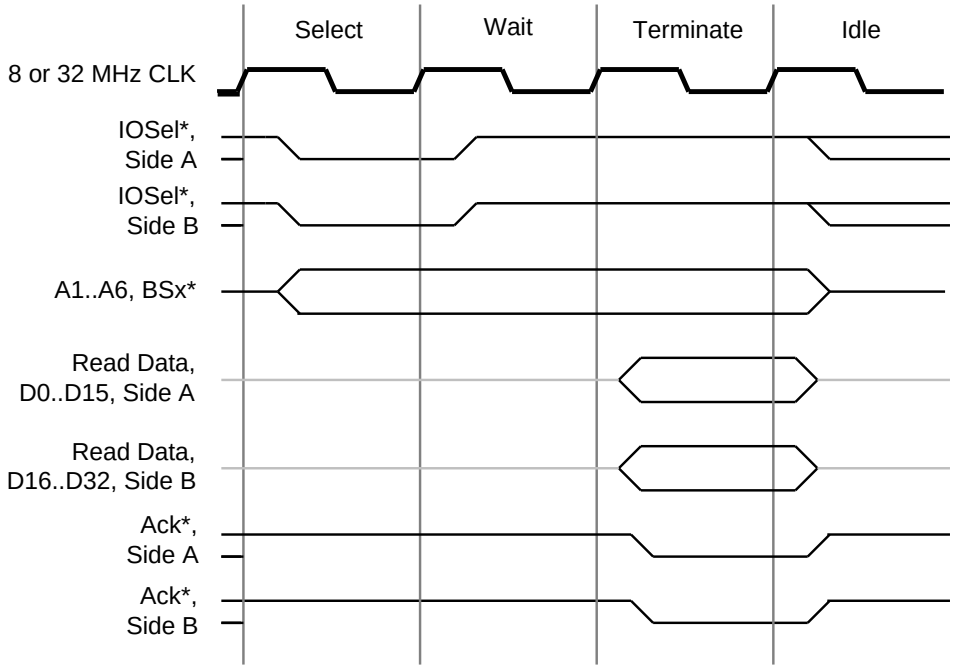
Notes: (1) DMAck* is driven true at the same time as A1, BSx*.
(2) DMAEnd*, when driven by the Carrier, is asserted the clock cycle following Select.
(3) DMAEnd*, when driven by the Module, is asserted during Ack* true.
(4) DMAEnd* signal is driven open drain, with a 1.2 KΩ resistor on the Carrier and each Module that uses DMA.
(5) DMAReqx* is removed or re-asserted during Ack* true.
(6) Read data is driven on the cycle during Ack* true.

Figure 9 8 MHz DMA Write Cycle with Wait State



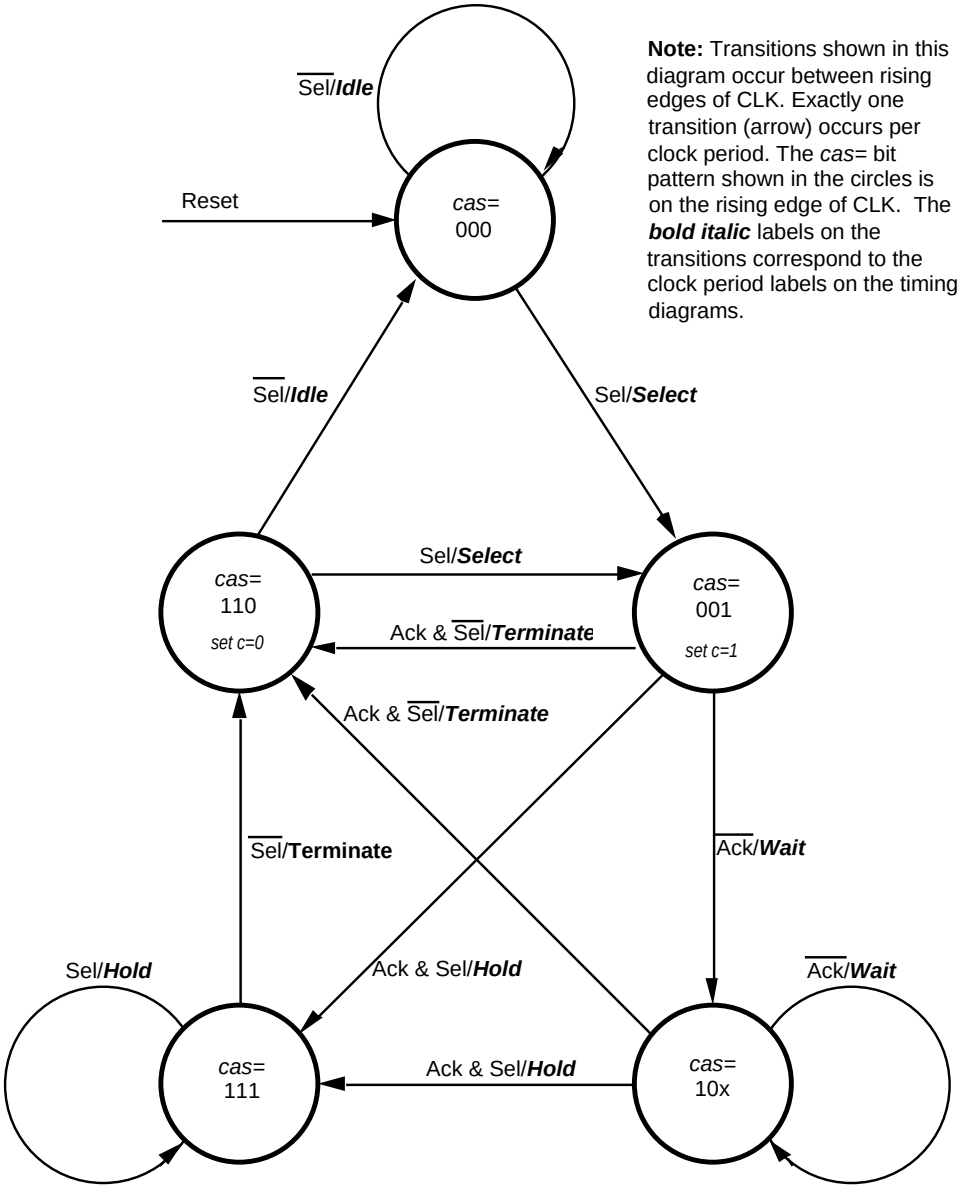
Notes: (1) DMAck* is driven true at the same time as A1, BSx*.
(2) DMAEnd*, when driven by the Carrier, is asserted the clock cycle following Select.
(3) DMAEnd*, when driven by the Module, is asserted during Ack* true.
(4) DMAEnd* signal is driven open drain, with a 1.2 KΩ resistor on the Carrier and each Module that uses DMA.
(5) DMAReqx* is removed or re-asserted during Ack* true.
(6) Write data is driven on the cycle following Select.

Figure 10 32-bit Double-size Operation



Notes: (1) Select lines driven simultaneously for A and B side.
(2) Ack* is asserted simultaneously for A and B side.
(3) Low order bytes on A side; high order bytes on B side.
(4) All four byte strobes must be driven to match byte lanes.
(5) I/O Addresses are driven bussed to both A and B sides.
(6) See text for interpretation of memory addresses.
(7) See Figure 26 for Mechanical Outline.

Figure 11 8 MHz State Diagram

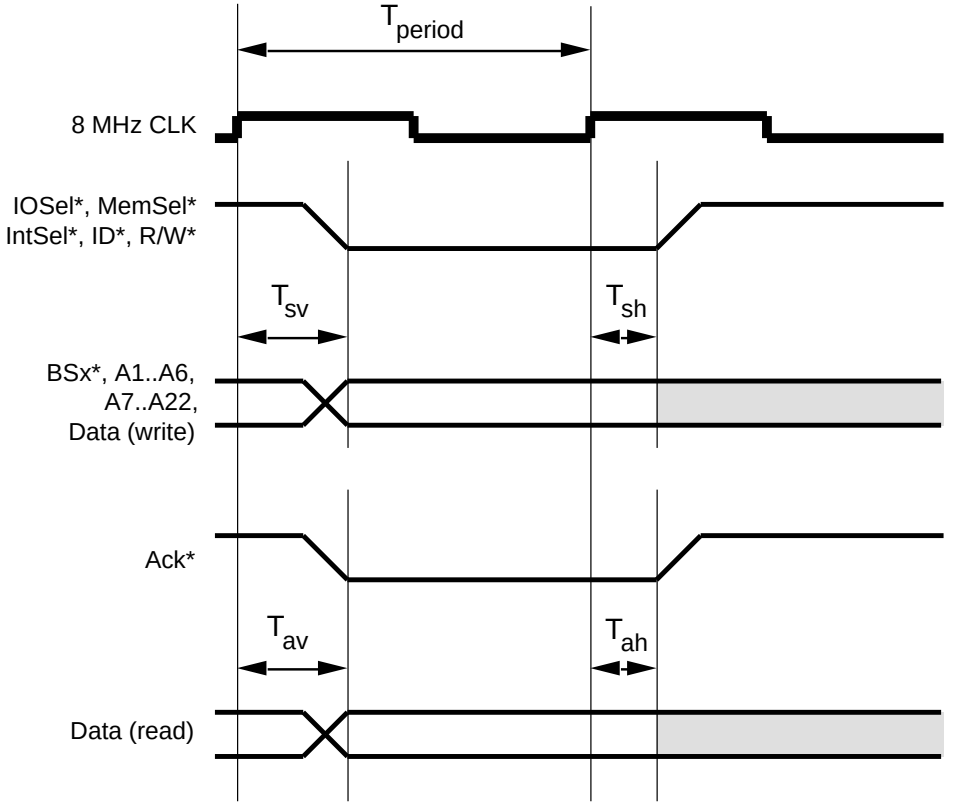


IP Module Logic Interface State Diagram

cas = 3 bit state number shown above, where,
c = cycle (local one-bit state variable),
a = Ack* signal from interface,
s = select signal from interface (IOSel*, MemSel* IntSel* or IDSel*),
1 = true, 0 = false.

Note: Signals from interface are sampled on the rising clock edge prior to the state.

Figure 12 8 MHz Detailed Timing Diagram



Symbol	Name	Time	Reference
Tperiod	Clock Period	125±2 ns	IP Carrier
Tsv	Signal Valid Delay	40 ns maximum	IP Carrier
Tsh	Signal Hold	0 ns minimum	IP Carrier
Tav	Acknowledge Valid Delay	40 ns maximum	IP Module
Tah	Acknowledge Hold	0 ns minimum	IP Module

Figure 13 32 MHz Two I/O Write Cycles Back to Back

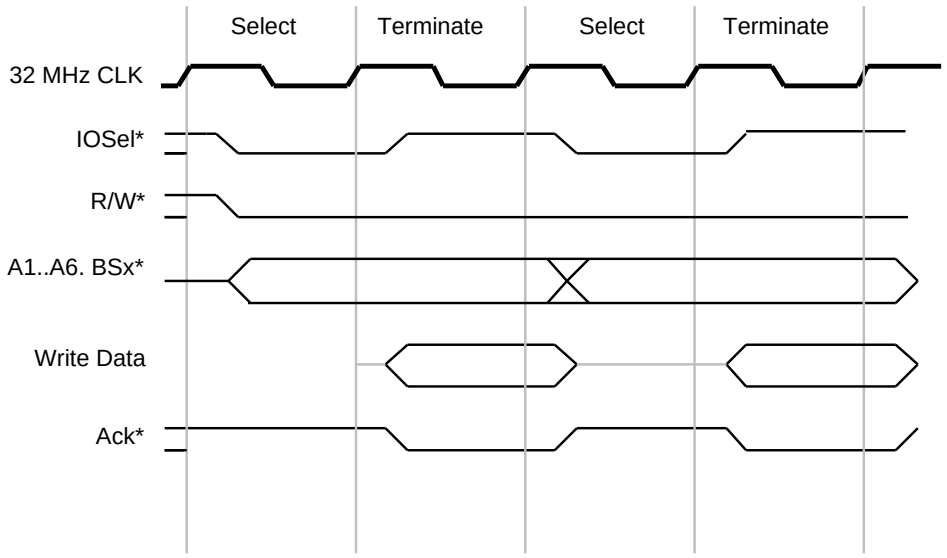


Figure 14 32 MHz Two I/O Read Cycles Back to Back

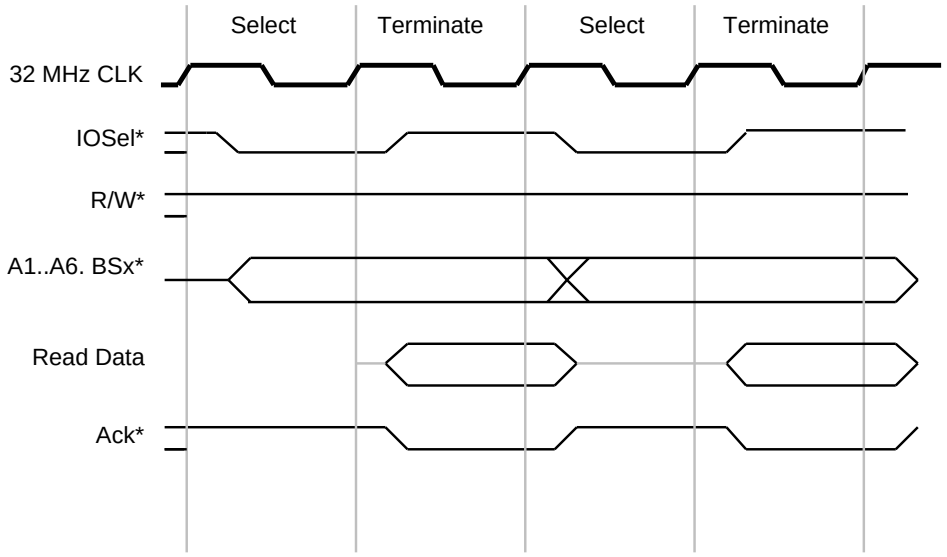


Figure 15 32 MHz Fastest Memory Write Cycle

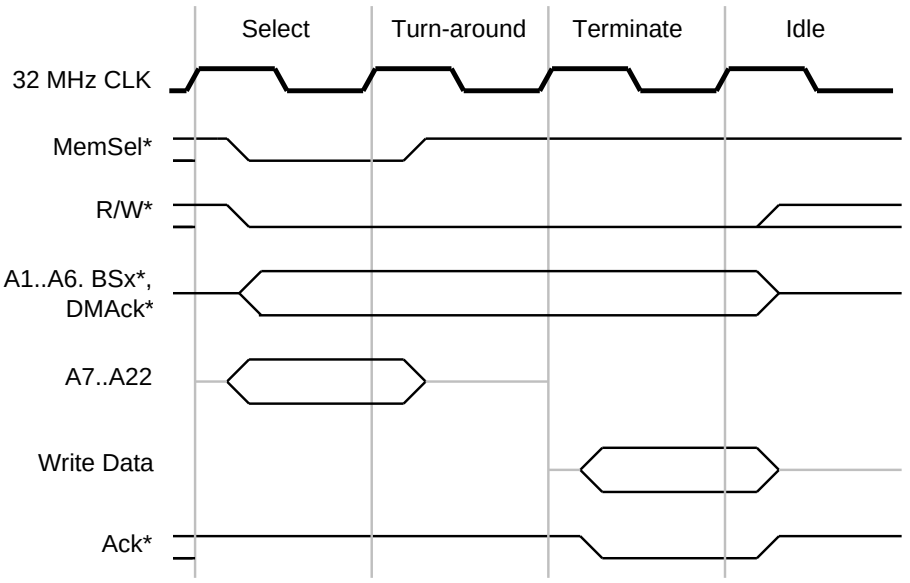


Figure 16 32 MHz Fastest Memory Read Cycle

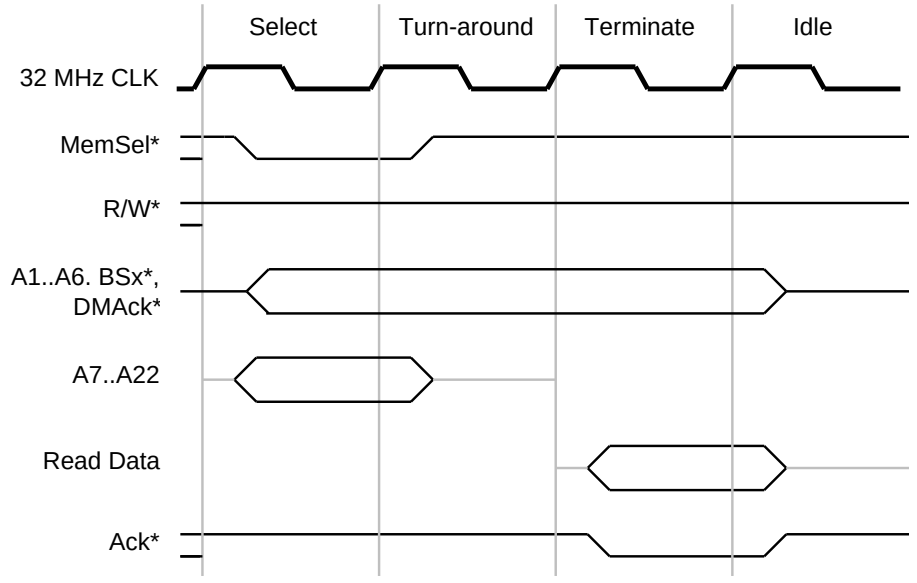


Figure 17 32 MHz Memory Read Cycle with Wait State

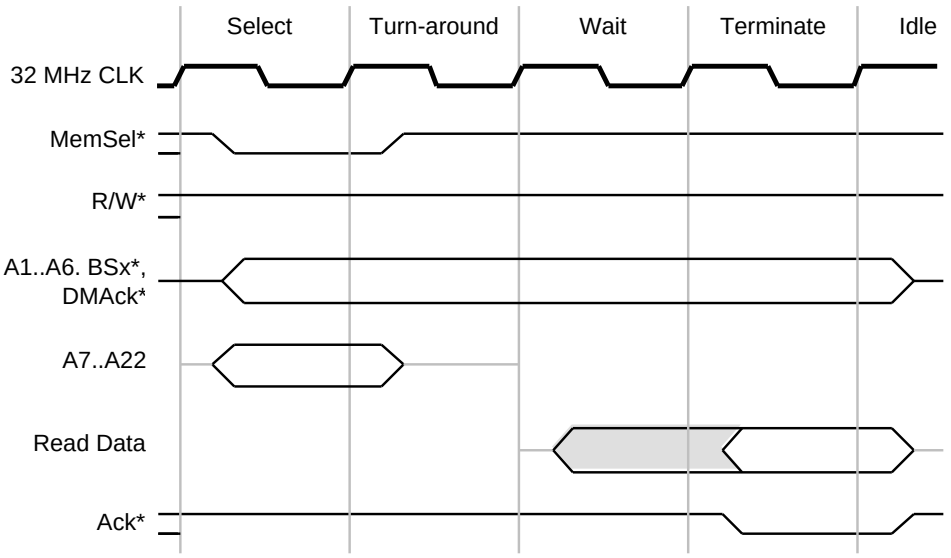


Figure 18 32 MHz Two Memory Cycles Back to Back

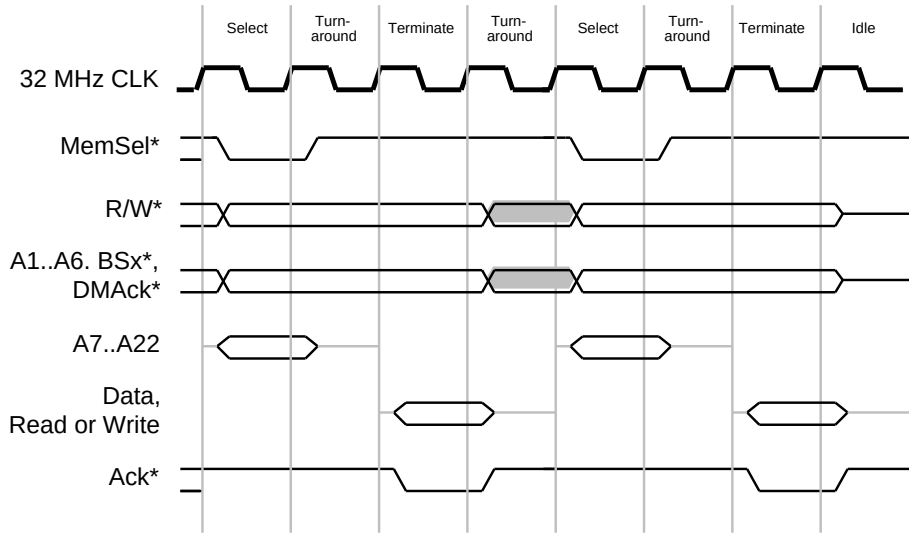
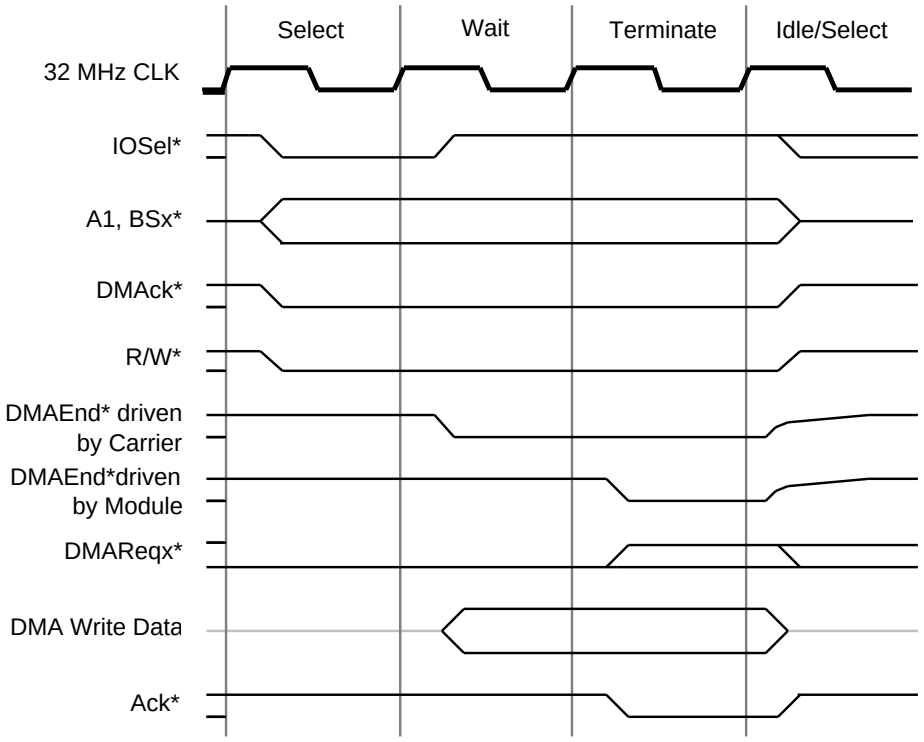
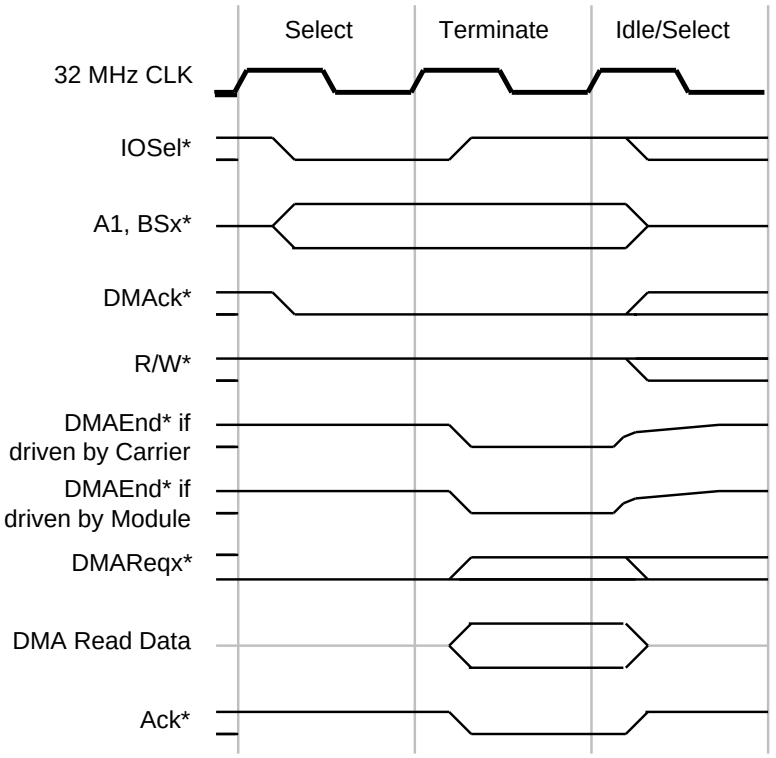


Figure 19 32 MHz DMA I/O Write Cycle with Wait State



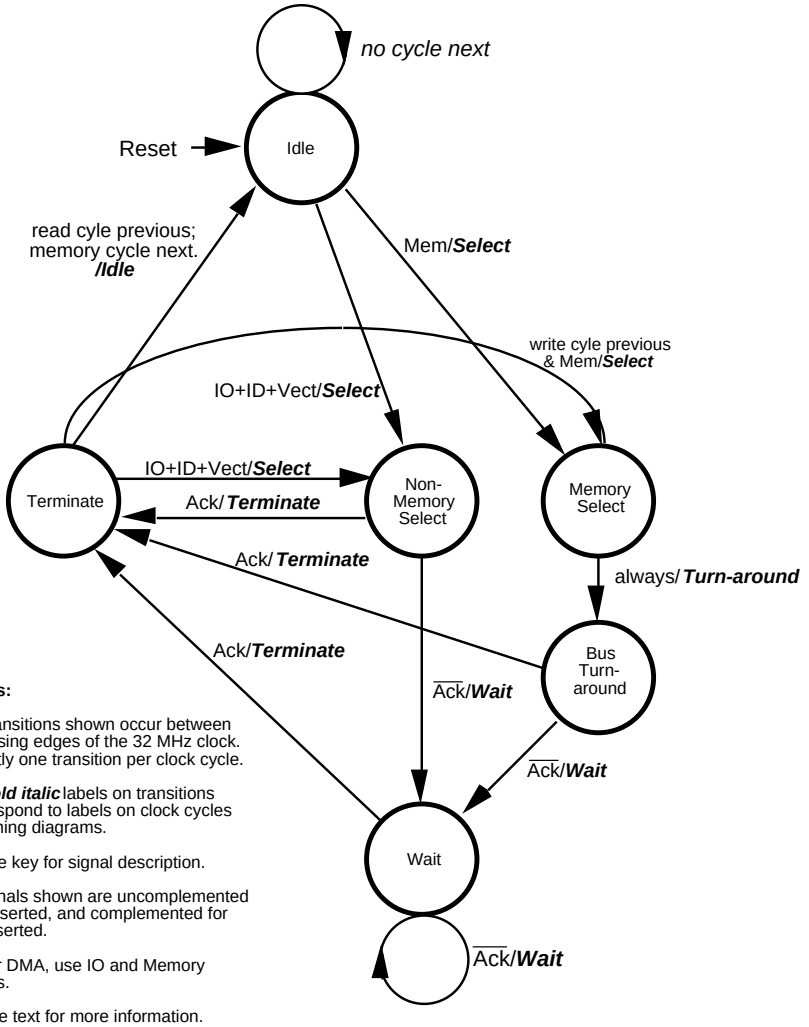
Notes: (1) DMAck* is driven true at the same time as A1, Bsx*.
(2) DMAEnd*, when driven by the Carrier, is asserted the clock cycle following Select.
(3) DMAEnd*, when driven by the Module, is asserted during Ack* true.
(4) DMAEnd* signal is driven open drain, with a 1.2 KΩ resistor on the Carrier and each Module that uses DMA.
(5) DMAReqx* is removed or re-asserted during Ack* true.
(6) Write data is driven on the cycle following Select.

Figure 20 32 MHz DMA I/O Read Cycle



Notes: (1) DMAck* is driven true at the same time as A1, Bsx*.
(2) DMAEnd*, when driven by the Carrier, is asserted the clock cycle following Select.
(3) DMAEnd*, when driven by the Module, is asserted during Ack* true.
(4) DMAEnd* signal is driven open drain, with a 1.2 KΩ resistor on the Carrier and each Module that uses DMA.
(5) DMAReqx* is removed or re-asserted during Ack* true.
(6) Read data is driven on the cycle during Ack* true.

Figure 21 32 MHz State Diagram



Notes:

1. Transitions shown occur between the rising edges of the 32 MHz clock. Exactly one transition per clock cycle.

2. ***Bold italic*** labels on transitions correspond to labels on clock cycles on timing diagrams.

3. See key for signal description.

4.Signals shown are uncomplemented for asserted, and complemented for unasserted.

5. For DMA, use IO and Memory cycles.

6. See text for more information.

Key:

Ack — Acknowledge signal
IO — IO cycle
ID — ID cycle
Mem — Memory cycle
Rd — Read cycle
Vect — Interrupt Vector cycle
Reset — Reset signal
\+ — logic OR
& — logic AND

Figure 22 32 MHz Detailed Timing

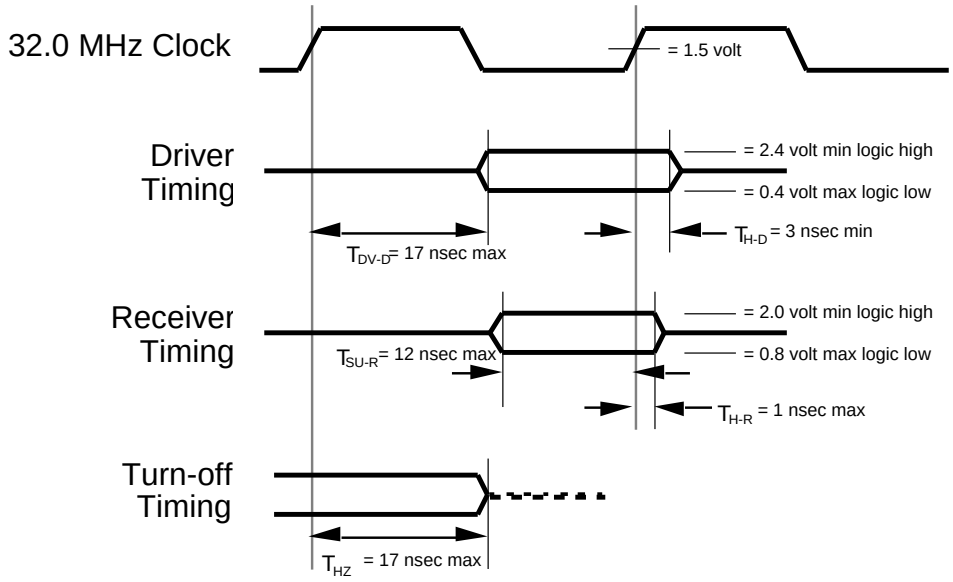


Figure 23 Single-size IP Module Mechanical Outline

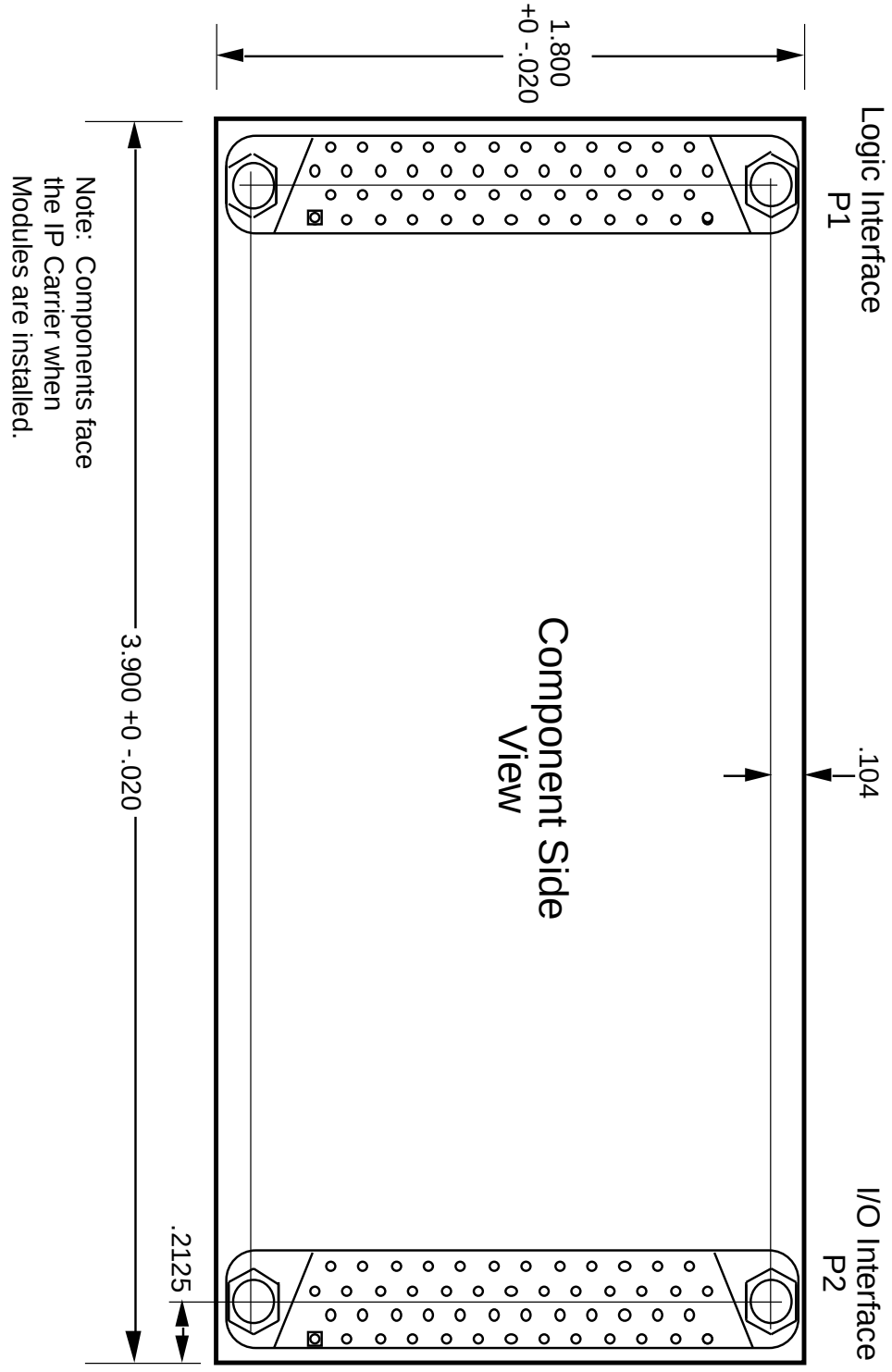

1.800
+0 -.020
Logic Interface
P1
.104
Component Side
View
I/O Interface
P2
.2125
3.900 +0 -.020
Note: Components face
the IP Carrier when
Modules are installed.

Figure 24 IP Module Mechanical Detail

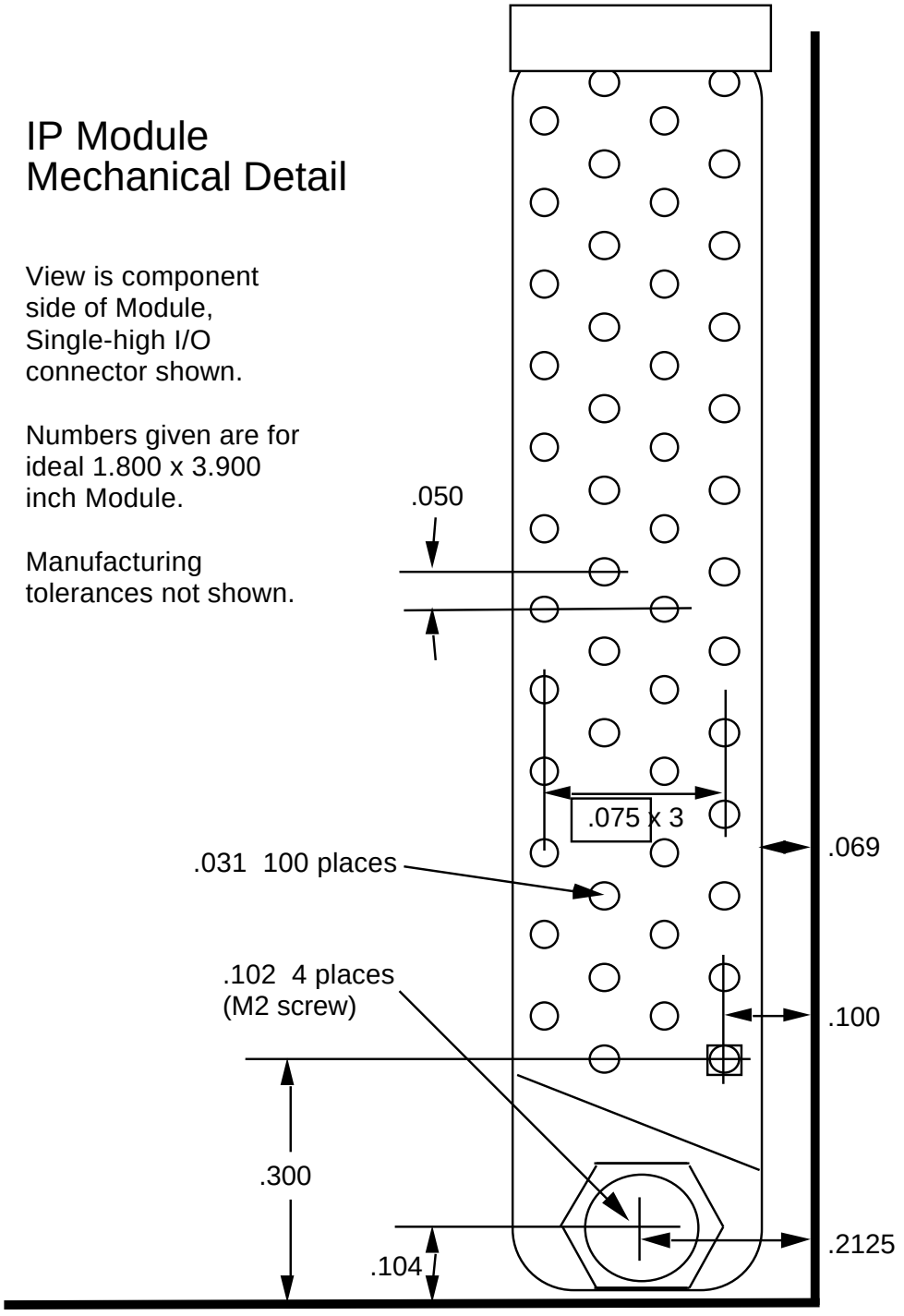


Figure 25 IP Module Pin Numbering

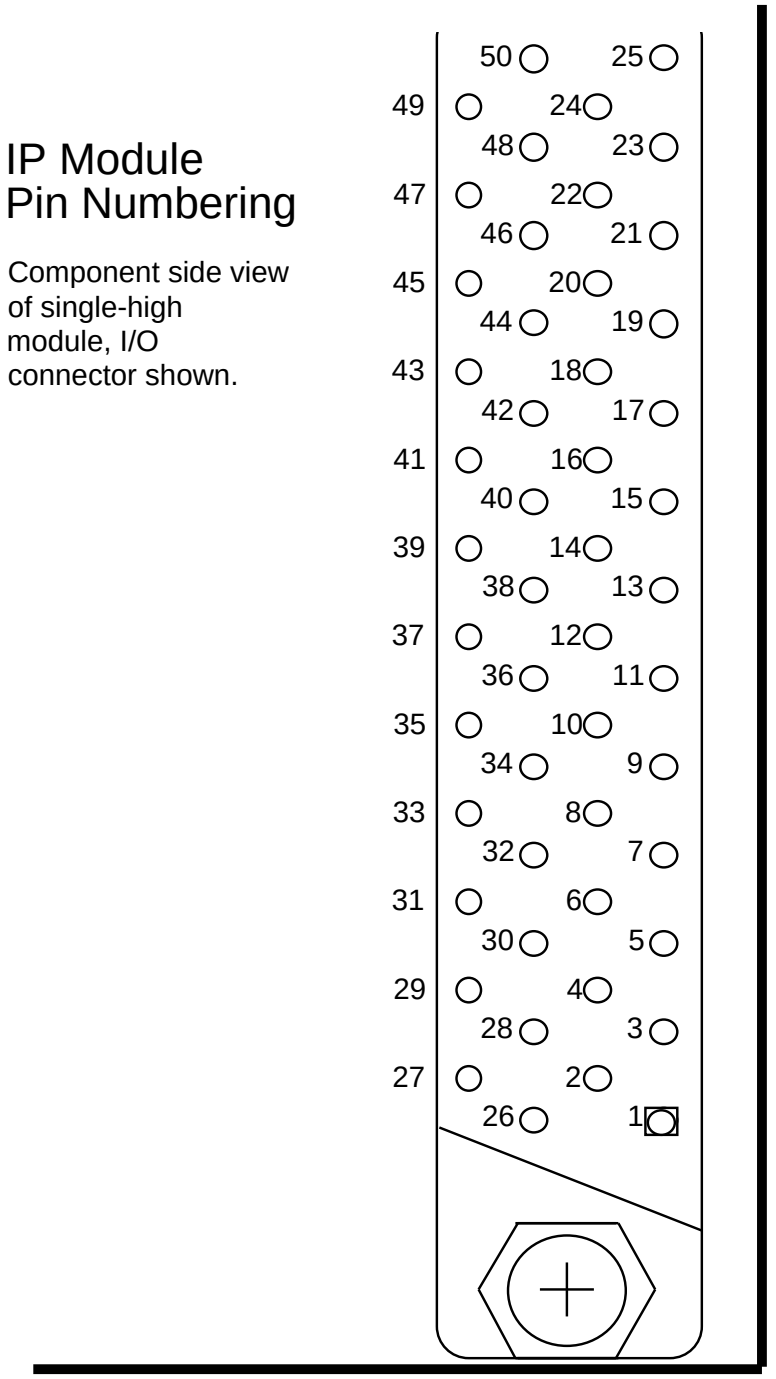
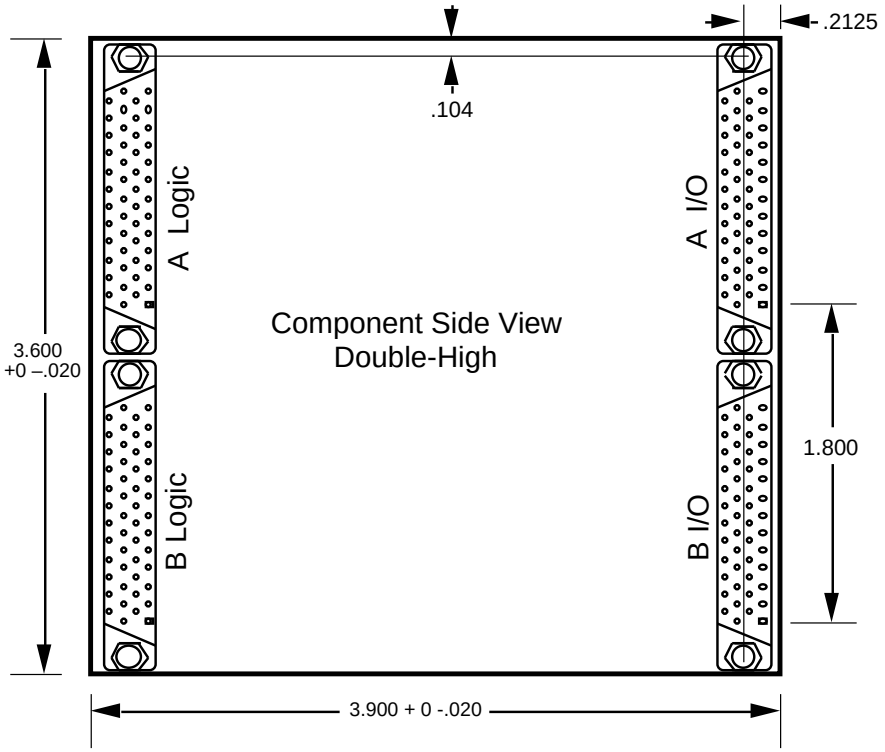


Figure 26 Double-size Module Mechanical



Note 1: Component side of Module shown. Components face Carrier when installed

Note 2: For 32-bit operation, logic A side carries LSB, logic B side carries MSB.

20. Appendix A

The following four program listings are provided as unsupported examples of code that can generated and test the checksum in the IP Module ID PROM.

This software has not been validated in any formal way, although it has been distributed widely in this form since 1992. The formal definition of the checksum is given in the body of this Standard.

The first listing is in 68K assembly language. The second listing is in MS-BASIC, assumed to run under DOS. The third and fourth listings are in C.

This is a 68000 Assembly Language Listing of a subroutine that checks CRC (checksums) in IP ID PROMs.

It was originally written by Rick Miley for GreenSpring Computers.

```
; CheckCRC performs a sum check on the 32 IP ID PROM bytes
; located in the private storage buffer pointed to by
; register A0.
;
; INPUT     A0.L        Pointer to buffer which holds IP PROM
Contents

CheckCrc    CMPI.L      #'IPAC',(A0)            ; if first four
                                                   bytes not
                                                   IPAC
            BNE.S       @err                    ; then not valid
                                                   pointer
            CMP.B       #$20,sizeOffset(A0)     ; check length
            BGT.S       @err
            MOVE.B      crcOffset(a0),D4        ; read current
                                                   CRC
            CLR.B       crcOffset(a0)           ; zero the CRC
                                                   out

            MOVEQ.l     #-1,D3                  ; index
            MOVE.L      #$FFFF,D1               ; remainder

@0          ADDQ.L      #1,D3                   ; increment
                                                   index
            CMP.B       sizeOffset(A0),D3       ; if D3=actual
                                                   size then
                                                   done
            BEQ.S       @done
            MOVE.L      #$80,D2                 ; mask
@1          MOVE.B      (A0,D3.W),D0            ; pick up data
            AND.B       D2,D0                   ; mask for one
                                                   bit
            BEQ.S       @2
            EORI.W      #$8000,D1
@2          ADD.L       D1,D1                   ; shift
                                                   remainder
            CMPI.L      #$FFFF,D1               ; note only WORD
                                                   compare
            BLE.S       @3
            ANDI.L      #$FFFF,D1
            EORI.W      #$1021,D1
@3          LSR.L       #1,D2                   ; rotate mask
            BNE.S       @1
            BRA.S       @0

@done       MOVE.B      D4,crcOffset(A0)        ; put the CRC
                                                   back in place
            NOT.W       D1
            CMP.B       D4,D1
            BNE.S       @err

@good       MOVEQ.L     #-1,D0
            RTS
```

```
@err        MOVEQ.L     #0,D0
            RTS
```

```
195 REM   n$[] holds the characters of each byte
196 REM   p[] holds the value of each byte
200 DIM N$[32],P[32]
210 INPUT "Enter the hex file name without the '.hex' :",A$
220 OPEN "R",#1,A$+".hex",128
240 FIELD#1,10 AS J$,3 AS N$[1],3 AS N$[2],3 AS N$[3],3 AS
N$[4],3 AS N$[5],
3 AS N$[6],3 AS N$[7],3 AS N$[8],3 AS N$[9],3 AS N$[10],3 AS
N$[11],3 AS N$[12],3 AS N$[13],3 AS N$[14],3 AS N$[15],3 AS
N$[16]
260 FIELD #1,67 AS J1$,3 AS N$[17],3 AS N$[18],3 AS N$[19],3 AS
N$[20],3 AS N$[21],3 AS N$[22],3 AS N$[23], 3 AS N$[24],3 AS
N$[25],3 AS N$[26],3 AS N$[27],3 AS N$[28],3 AS N$[29],3 AS
N$[30],3 AS N$[31],3 AS N$[32],7 AS J2$,4 AS SUM$
275 REM   Fetch the whole hex file
280 GET 1
300 SUM%=0
315 REM   This loop converts the strings to numbers
320 FOR X=1 TO 32
340 C$ = N$[X]:GOSUB 1000:P[X]=NN
360 SUM = SUM + NN
375 REM   To see your file, remove "rem" from next line
380 REM   PRINT P[X];
400 NEXT X
415 REM   If you want to see the current checksum, remove "rem"
:
420 REM   PRINT:PRINT "sum = ";SUM$
435 REM   The number of bytes the manufacturer uses in the ID
PROM
436 REM   is contained in the 11th byte of the PROM.
440 SIZ = P[11]
455 REM   The CRC byte must be set to zero to calculate the CRC
460 P[12]=0
480 GOSUB 2000
500 P[12]=INT(RM#)
520 RM$=HEX$(RM#)
535 REM   Make a two character field from the CRC (in T$)
540 T$="00":TT$=N$[12]
560 MID$(T$,3-LEN(RM$),LEN(RM$))=RM$
575 REM   Keep the old separator character (in TT$, now)
580 MID$(TT$,1,2)=T$
595 REM   Put the new CRC string back into the record
600 RSET N$[12]=TT$
620 PRINT "CRC (in hex) = ";LEFT$(N$[12],2)
640 P[12]=INT(RM#)
655 REM   Calculate the new download checksum
660 SUM=0
680 FOR X=1 TO 32
700 SUM=SUM+P[X]
720 NEXT X
740 REM   PRINT "new download checksum =";SUM
755 REM   Convert the checksum to a four character string
760 T$="0000"
780 SM$=HEX$(SUM)
800 MID$(T$,5-LEN(SM$),LEN(SM$))=SM$
815 REM   Put the checksum string into the record
820 RSET SUM$=T$
835 REM   Write the revised record back to the file
840 PUT 1,1
```

```
860 CLOSE 1
880 PRINT:PRINT"Finished adding CRC and new download checksum
into file ";A$+".hex"
895 REM   To stay in BASIC, put "REM" in front of next line.
900 SYSTEM
920 END
1000 REM This subroutine converts a hex string to an integer.
1020 REM The input string is in c$, the output number is nn.
1040 TL=ASC(MID$(C$,1,1))
1060 TR=ASC(MID$(C$,2,1))
1080 IF TL<=57 THEN NN=16*(TL-48) ELSE NN=16*(TL-55)
1100 IF TR<=57 THEN NN=NN+TR-48 ELSE NN=NN+TR-55
1120 RETURN
2000 REM  This subroutine generates the CRC.
2020 REM  It uses two nested loops: the outer loops is executed
once for
2040 REM  for each byte used by the manufacturer.  The inner
loop is
2060 REM  executed eight times, once per bit of the byte.
2075 REM   Set RM to &hffff
2080 RM# = 65535#
2100 FOR X= 1 TO SIZ
2120 MASK = &H80
2140 FOR XX= 1 TO 8
2160 IF (P[X] AND MASK) = 0 GOTO 2200
2175 REM   Invert the MSB of RM
2180 IF RM#>32767# THEN RM#=RM#-32768# ELSE RM#=RM#+32768#
2195 REM   Shift left
2200 RM# = RM# * 2
2215 REM   Test overflow (bit 17)
2220 IF RM#<=65535# GOTO 2310
2235 REM   Clear overflow (bit 17)
2240 RM# = RM# - 65536#
2255 REM   The next two lines do, "RM=RM XOR &h1021" only BASIC
2256 REM   can only do 15 bit logicals, so some extra work is
required.
2260 IF RM#<=32767# THEN RM#=RM# XOR &H1021:GOTO 2300
2280 RM#=RM#-32768#:RM#=RM# XOR &H1021:RM#=RM#+32768#
2300 REM  PRINT "2300   rm# =";RM#
2305 REM   Shift mask right.  Use integer division only.
2310 MASK = MASK\2
2320 NEXT XX
2340 NEXT X
2355 REM   Throw away sign bit to prevent overflow.
2360 IF RM#>32767# THEN RM#=RM#-32768#
2376 REM   Return low byte, ONE's complemented.
2380 RM#=(RM# AND &HFF) XOR &HFF
2400 RETURN
```

```
/*
 * ip _check_crc() -
 *
 * DESCRIPTION:
 *
 * Compute Ip Module ID PROM checksum
 *
 * ARGUMENTS:
 *
 * ipid - pointer to ID string
 * crcn - index to crc byte
 *
 * RETURN:
 *
 * 1 - found valid identifier string
 * 0 - non-valid crc
 *
 * This listing is by Hemsceidt Industrie Elektronik
 */

#define CRC_OFFSET 11               /* offset to crc byte */
#define SIZE_OFFSET 10              /* offset to 'number of bytes
used' */

int ip_check_crc (char *ipid)
{

 register int i, size;
 register u_int rem = 0xffff;
 char crc, accu, mask = 0x80;

 if (ipid && *ipid) {
   if (strncmp ("IPAC", ipid, 4 ) != 0)        /* check initials */
      return 0;
   crc = ipid[CRC_OFFSET];                     /* save crc out */
   ipid[CRC_OFFSET] = 0x0;                     /* zero crc */
   size = (int) ipid[SIZE_OFFSET];
   if (size > 32 ) return 0;                   /* check length */

   for (i=0, i<size, i++) {
      mask = 0x80;
      do {
         accu = ipid[i];                       /* pick up data */
         accu &= mask;                         /* mask out one bit */
         if (accu) {                           /* if true... */
            rem ^= 0x8000;                     /* ...xor remainder */
         }
      rem <<=1;                                /* shift remainder */
      if (rem > 0xfff) {                       /* if greater... */
         rem &= 0xffff;                        /* ...mask lower 16
bits */
         rem ^=0x1021;
         }
         mask >>=1;                            /* shift mask right */
      }
      while (mask);
   }
   ipid[CRC_OFFSET] = crc;                     /* restore crc value
*/
```

```
    if ((char) -rem == crc) return 1;
  }
  return 0;
}
```

```
#define  REGISTER register
#define  HIDDEN   static

typedef  int            int32;
typedef  short          int16;
typedef  char           int8;
typedef  unsigned int   uint32;
typedef  unsigned short uint16;
typedef  unsigned char  uint8;
typedef  float          float32;
typedef  double         float64;


/* ------------------------------------------------------------------
--- */
/* This program was written by GreenSpring Computers for IP ID CRCs.
*/
/* ------------------------------------------------------------------
--- */

/* Definitions */

typedef struct {
   uint8       gap0[0x80];          /* Offset from base to PROM */
   uint8       gap1;
   char        sync1;               /* 'I' */
   uint8       gap2;
   char        sync2;               /* 'P' */
   uint8       gap3;
   char        sync3;               /* 'A' */
   uint8       gap4;
   char        sync4;               /* 'C' */
   uint8       gap5;
   uint8       manid,               /* Manufacturer ID */
               gap6,
               model,               /* Model number */
               gap7,
               rev,                 /* Revision */
               gap8,
               reserved,            /* Reserved */
               gap9,
               driverlow,           /* Driver ID low */
               gap10,
               driverhigh,          /* Driver ID high */
               gap11,
               size,                /* Size of PROM data (#entries) */
               gap12,
               crc,                 /* CRC check */
               gap13;
} IPProm;


/* ------------------------------------------------------------------
--- */

/* Return 1 byte CRC code */

/* Limited to PROMs with 32 entries */
```

```
#define MAXENTRIES 32

HIDDEN uint32
prom_crc(ip)
REGISTER IPProm *ip;
{
   uint8 a[MAXENTRIES];
   REGISTER uint8 *p,*q;
   REGISTER int32 i=ip->size;
   REGISTER uint32 rem,mask;

   if(i>MAXENTRIES)                    /* Ensure no overflow */
      return(~0);
   for(p=(uint8 *)(&ip->sync1),q=a;i--;p+=2)   /* Copy prom contents
to RAM */
      *q++=*p;
   a[11]=0;                            /* Zap the CRC */

   rem=0xffff;
   for(i=ip->size,p=a;i--;p++) {       /* For each entry */
      mask=0x80;
      do {                             /* For each bit */
         if(mask&*p)
            rem^=0x8000;
         rem+=rem;
         if(rem>0xffff) {
            rem&=0xffff;
            rem^=0x1021;
         }
      } while((mask>>=1)!=0);
   }
   rem=~rem;
   rem&=0xff;
   return(rem);
}
```

21. Appendix B

This Appendix lists metric equivalents for units in inches used in this Specification.

From Section 2.

inches	metric mm
0.062	1.575
.007	0.1778
.008	0.2033

From Section 15.

inches	metric mm
.100	2.5407
.050	1.2703
.075	1.9055
.012	0.3049

From Section 17.

inches	metric mm
1.800	45.7317
3.900	99.0854
0.020	0.5801
3.600	91.4634
.472	11.9919
.157	3.9973
.314	7.9946
.290	7.3679
.250	6.3516
.062	1.5752
.534	13.5671
.072	1.8293
.055	1.3974

From Figure 23.

inches	metric mm
1.800	45.7317
3.900	99.0854
.020	0.5801
.104	2.6423
.2125	5.3989

From Figure 24.

inches	metric mm
.050	1.2703
.075	1.9055
.100	2.5407
.069	1.7530
.031	0.7876
.102	0.7876
.300	7.6220
.104	2.6423
.2125	5.3989

From Figure 26.

inches	metric mm
3.600	91.4634
3.900	99.0854
1.800	45.7317
.020	0.5801
.104	2.6423
.2125	5.3989