Revision A
9/15/86

GMX™ Micro-20
68020 Single-board Computer

Hardware Technical Manual

The information in this document has been carefully checked and is believed to be entirely reliable. However, no responsibility is assumed for inacurracies. Furthermore, GMX Inc. reserves the right to make changes to any products described herein to improve reliability, function, or design. GMX Inc. does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights or the rights of others.

GMX Inc.
1337 West 37th Place
Chicago, IL 60609
(312) 927-5510 * TWX 910-221-4055

GMX™ Micro-20 68020 Single-board Computer
Hardware Technical Manual

Revision History

Revision	Changes	Date
A	First Complete Edition	9/15/86

GMX™ Micro-20 68020 Single-board Computer
Hardware Technical Manual

TABLE OF CONTENTS

(Con't)

Section 6: HALT INDICATORS, ON-BOARD (LED1) AND REMOTE

Section 7: POWER SUPPLY REQUIREMENTS AND CONNECTIONS

Section 8: SENSE SWITCH (SW1)

Section 9: RANDOM ACCESS MEMORY (RAM)

Section 10: READ-ONLY MEMORY (ROM) SOCKETS

Section 11: SERIAL I/O PORTS

Section 12: 68230 PARALLEL INTERFACE/TIMER (PI/T)

Section 13: General Purpose Parallel Port

(Con't)

Section 14: Time-of-Day Clock

Section 15: PERIODIC INTERRUPT GENERATOR

Section 16: FLOPPY DISK INTERFACE

Section 17: SASI/SCSI PERIPHERAL INTERFACE

Section 18: I/O EXPANSION PORT

(Con't)

APPENDICIES

LIST OF FIGURES

Page

Section 3: RESET SWITCH

Section 4: SINGLE-STEP/ABORT SWITCH

Section 5: REMOTE POWER INDICATOR

Section 6: HALT INDICATORS, ON-BOARD (LED1) AND REMOTE

Section 7: POWER SUPPLY REQUIREMENTS AND CONNECTIONS

Section 10: READ-ONLY MEMORY (ROM) SOCKETS

Section 11: SERIAL I/O PORTS, 68681 DUARTs

Section 13: General Purpose Parallel Port

Section 15: PERIODIC INTERRUPT GENERATOR

Section 16: FLOPPY DISK INTERFACE

(Con't)

Section 17: SASI/SCSI PERIPHERAL INTERFACE

Section 18: I/O EXPANSION PORT

GENERAL SPECIFICATIONS

Size: 8.8" x 5.75" x 0.68" (refer to Appendix E for mtg. hole locations)

DC Power Requirements: +5V and +12V DC Regulated (see Section 7 for current requirements)

Microprocessor: 12.5, 16.67, or 20 MHz 68020

Coprocessor: 12.5, 16.67, or 20 MHz 68881 FP Coprocessor (optional)

Random Access Memory: 2 Megabytes of high-speed (1 wait-state) DRAM, organized as 512K 32-bit words.

Read-only Memory: Supports up to 256K bytes of ROM, organized as 32-bit words. Accepts four 2764, 27128, 27256, or 27512 EPROMs or equiv.

Serial Ports: Two 68681 DUARTs provide four asyncronous serial ports. Supports 50 to 38.4K baud; 5 to 8 data bits; odd, even, no parity. TTL level I/O is converted to RS-232 etc. by a separate board.

Parallel Interface: 68230 PI/T provides 8-bit parallel input or output port, and controls TOD clock interface, periodic interrupt generator and other misc. functions.

Periodic Interrupt Generator: Generates periodic interrupts at a jumper selectable interval from 10 microseconds to 20 minutes.

Floppy Disk Interface: WD1772 FDC or equiv., handles one or two std. 5 1/4" floppy disk drives.

SASI/SCSI Interface: SASI and SCSI (subset) compatible interface for hard disk and tape controllers, etc.

Time-of-Day Clock: RTC58321 or equiv. TOD clock with full clock/calendar functions and battery backup.

I/O Expansion Port: 16-bit data bus, 1K to 4K address space, two interrupt levels, and control signals for I/O expansion.

Jumper Options Summary

Jumper	Description	Page(s)
JA-1A	ROM Size Select	10-2
JA-IB	Periodic Interrupt Gen. Rate Select	15-2
JA-2	Battery Disconnect	14-6
JA-3	Parallel Port Data Direction	13-2
JA-4	68020 Cache Disable	1-2
JA-5	Single-step/Abort Option	4-2
JA-6	ROM Wait-state Select	10-2
JA-7	Parallel Port Handshake Direction	13-2
JA-8	Floppy Disk Ready Option	16-2
JA-9	68881 Coprocessor Clock Option	2-1
W1	Spare IC (U3) VCC Select	h-1
W2	Spare IC (U3) GND Options	h-1
W3	TOD Clock TEST Disable	h-1

Connector Summary

Connector	Description	Page(s)
P1	I/O Expansion Connector, 44-pin dual-row socket for .025" sq. pins on .1" centers. Mates with: Samtec #SSQ-122-04-G-D, or eq.	18-5
P2	Serial I/O Connector, 50-pin dual-row header w/.025" Sq. pins on .1" centers. Mates with: T&B Ansley #609-5030, or eq.	11-4
P3	Parallel I/O Connector, 36-pin dual-row header w/.025" sq. pins on .1" centers. Mates with: T&B Ansley #609-3630, or eq.	13-1
P4	Floppy Disk Interface, 34-pin dual-row header w/.025" sq. pins on .1" centers. Mates with: T&B Ansley #609-3430, or eq.	16-8
P5	SASI/SCSI Interface, 50-pin dual-row header w/.025" sq. pins on .1" centers. Mates with: T&B Ansley #609-5030, or eq.	17-16
P6	Front panel indicators, 5-pin single-row header w/.025" sq. pins on .1" centers. Mates with: Molex #22-01-2051 using #08-50-0114 contacts, or eq.	5-1, 6-1
P7	Front panel switches, 6-pin single-row header w/.025" sq. pins on .1" centers. Mates with: Molex #22-01-2061 using #08-50-0114 contacts, or eq.	3-1, 4-1
P8	DC power connector, 4-pin 90 deg. header Mates with: AMP #1-480424-0 using #60617-1 contacts, or eq.	7-2

On-Board Status Indicators

Indicator	Description	Page(s)
LED1	Processor HALT Indicator, lights when the processor halts due to a double bus fault or when processor is halted by the Single-step option.	6-1
LED2	Software Diagnostic indicator, flashes coded fault messages if 020Bug power-up tests fail. Also indicates state of floppy disk side select line during normal operation. Refer to the 020Bug User's manual for more info.	16-8

Optional (user-supplied) Off-Board Status Indicators

Indicator	Description	Page(s)
HALT	An LED connected to P6 duplicates the function of LED1.	6-1
POWER	An LED connected to P6 lights when +5V is applied to the board.	5-1

INTRODUCTION

This manual contains detailed information on the hardware features of the GMX Micro-20 68020 Single-Board Computer. It is primarily intended for those who are designing custom hardware to interface to the board, and for those who must write programs that directly access the on-board hardware.

For those using the board with the GMX Micro-20 Support ROM firmware or one of the optional disk operating systems, information on basic hardware setup can be found in the 'GMX Micro-20 Hardware Setup Manual', and in the software documentation.

Since the GMX Micro-20 can be ordered in several configurations, some of the information in this manual may not apply to a particular board. For example, not all boards include the 68881 Floating-point Coprocessor, and the information in section 2 does not apply.

The Micro-20 can be orderded configured for one of several processor speeds. Generally, processor speed is not significant except in terms of overall performance. However, certain parameters are clock speed dependent and, where necessary, information for each available speed is given in this manual. The processor speed of a given board can be determined by dividing the frequency of the master clock oscillator (U37) in half. For example, a clock oscillator frequency of 25 MHz, indicates a 12.5 MHz processor speed.

Some of the information in this manual only applies to a particular revision or revisions of the GMX Micro-20 printed circuit board. The revision level of a particular board is indicated by a single letter following the part number etched on the solder side of the PC board. For example, part number #24-0085B indicates a "B" revision board. Revision letters are assigned in alphabetical order, so that a revision "B" board is "newer" or "later" that a revision "A" board and "older" or "earlier" than a revision "C" board.

Every effort is made to maintain compatibility between various versions of the GMX Micro-20. Generally, hardware or software designed to work on older versions of the board will work with newer versions; however, the reverse may not be true. If it is necessary to maintain compatibility with older versions (e.g. an installed system base), the specifications for the older boards should be used whenever possible.

If you have any questions or comments concerning the GMX Micro-20, the documentation, or any other GMX product, please let us know.

********** CAUTION **********

The GMX Micro-20 includes a battery which provides backup capability for the on-board Time-of-Day clock. Because of this, parts of the circuitry are powered at all times, even when the external power source is disconnected.

Use caution when handling the board, to prevent short circuits which can damage the board and components. NEVER place the GMX Micro-20 on a conductive surface, such as a metal table top, and NEVER wrap the board in metal foil for static protection. Use only high-resistance anti-static materials such as conductive plastic or cloth to protect the board from static discharge.

To minimize (but not eliminate) the possibility of damage from short circuits, remove battery jumper JA-2 when handling or servicing the board. Further information can be found in section 14 of this manual.

****

1-1: General Description

This section provides specific information on the GMX Micro-20 implementation of the 68020 microprocessor. For complete information on the characteristics of the processor, refer to the 'MC68020 32-Bit Microprocesor User's Manual', Motorola publication number MC68020UM/AD.

1-2: Power-up/Hardware Reset Conditions

When power is first applied to the GMX Micro-20, on-board logic keeps the processor's RESET line asserted long enough for the power supplies and on-board logic to stabilize. After this initial power-up delay, RESET is deasserted an the processor begins normal operation. The processor and on-board logic can also be reset manually by a reset switch or by an external device connected to the I/O Expansion Port (refer to sections 3 and 18 of this manual for further information). These three forms of reset are called 'hardware resets'.

All hardware resets have essentially the same effect. The processor, coprocessor, serial ports (68681 DUARTs), parallel port (68230 PI/T), floppy disk controller (1772 FDC), and SASI interface are all reset as described in the device documentation and/or the appropriate sections of this manual. Any devices connected to the reset outputs from the SASI interface and the I/O Expansion Port are also reset. The contents of the on-board RAM, which is undefined on power-up, is unaffected by other forms of reset. Reset also causes the on-board ROMS to be mapped into low memory, as described in section 1-4.

1-3: Using the RESET Instruction

A 'software reset' is a reset caused when the processor executes the RESET instruction. The effects of a software reset are identical to those of a hardware reset, except that the processor itself is not reset. Instead of fetching the restart vectors and beginning execution at the restart address, the processor simply continues execution at the next instruction.

The on-board logic can not tell the difference between a software and a hardware reset, so both cause the ROMs to be mapped into low memory as described in section 1-4. This restricts the use of the processors RESET instruction to programs located in ROM.

The RESET instruction can NOT be executed from RAM. If used, the RESET instruction must be located in the ROMs. Immediately following the RESET instruction, a read of address $00000004 must be performed. This read restores normal memory mapping by simulating the vector fetch that occurs after a hardware reset. The data read is the restart vector from ROM address $00800004. Once this read is performed, normal operation can resume.

If it is necessary for programs in RAM to reset the system, a subroutine to perform a reset can be placed in the ROMs. Any program that needs to reset the system can then do so by calling the ROM subroutine. The subroutine need only include the RESET instruction, followed by a read of address $00000004 and an RTS instruction.

1-4: Restart Vectors

After a hardware reset, the processor expects the first two entries in the vector table (vector number zero, address $00000000 and vector number one, address $00000004) to contain initial values for the Interrupt Stack Pointer (ISP) and program counter (PC). Since these addresses are normally occupied by RAM on the GMX Micro-20, a reset causes the ROMs to appear temporarily in both their own address space and in low memory, overlaying the RAM. This dual mapping allows the processor to read the initial ISP and PC from the first two locations in the ROMs ($00800000 and $00800004). Normal mapping is restored as soon as the processor finishes reading the second vector. Refer to section 10 for further information.

1-5: Cache Disable Jumper (JA-4)

The 68020 has an input that disables its internal instruction cache, regardless of the state of the cache enable bit in the Cache Control Register (CACR). For normal cache operation, Cache Disable Option Jumper JA-4 must be open (no jumper). If a jumper is connected between the pins of JA-4, cache operation is inhibited.

2-1: General Description

This section provides information on the GMX Micro-20 implementation of the 68881 Floating-point Coprocessor. For complete information on the characteristics of the coprocessor, refer to the 'MC68881 Floating-point Coprocessor User's Manual', Motorola publication number MC68881UM/AD.

2-2: Coprocessor Interface

The coprocessor interface on the GMX Micro-20 is a full implementation of the M68000 family coprocessor interface defined by Motorola. Coprocessor instructions are coded in-line with 68020 instructions and the coprocessor registers are accessed as though they were part of the 68020 register set. The coprocessor is configured for coprocessor identification code (Cp-ID) 001, which is the recommended ID for the 68881. Software provided by GMX uses Cp-ID 001 by default when generating coprocessor instructions.

2-3: Coprocessor-installed Status Bit

A bit in the SASI Status Register (SASR) allows programs to determine whether or not a coprocessor is installed on the board. This bit simply indicates that there is a coprocessor in the socket, it is not used during normal coprocessor operations.

If there is no coprocessor installed, bit 6 of the SASR (address $00FF800E) will be clear (0). If a coprocessor is installed, SASR bit 6 will be set (1). For further information on the SASR refer to section 17.

2-4: Adding a Coprocessor

In order to add a 68881 coprocessor to a board originally purchased without one, it may be necessary to change the Programmable Logic Device (PLD) that, among other functions, controls the coprocessor address decoding. Different versions of this PLD (U-25) are used, depending on whether or not the board has a coprocessor and on the memory configuration needed. For further information on adding a coprocessor, please contact the factory.

2-5: Clock Rate

Normally, the processor and coprocessor operate at the same basic clock rate. However, due to cost and/or parts availability, it may be necessary or desirable to operate the two at different clock rates.

Revision 'B' and later boards (the revision letter is appended to the part number etched on the solder side of the circuit board, e.g. 24-0085B) allow the use of an additional clock oscillator to provide a separate clock for the coprocessor.

Jumper area JA-9 is used to select one of two sources for the coprocessor's clock. If JA-9 pins 1 and 2 are jumpered, the processor and coprocessor operate from the same clock. The clock rate is one-half of the master clock oscillator (U-37) frequency (e.g. for a 12.5 MHz processor/coprocessor, the oscillator frequency is 25 MHz).

If JA-9 pins 2 and 3 are jumpered, a separate clock oscillator, installed at location U-3, provides the coprocessor clock. The coprocessor clock oscillator frequency must equal the desired coprocessor clock rate and its output must have a 50% duty cycle. (e.g. a 12.5 MHz oscillator for a 12.5 MHz coprocessor).

3-1: General Description

Connector P7 is used to connect a user-supplied reset switch to the. GMX Micro-20. The reset switch provides a manual means of resetting the processor and other hardware to a known state, without removing power from the board. When the switch is activated, the processors RESET input is asserted causing the processor to begin reset exception processing. It also resets the on-board logic and peripherals, devices connected to the SASI interface, and devices connected to the I/O Expansion Port RST line.

A monostable circuit on the board provides contact debouncing for the switch and insures that the duration of the reset signal is sufficient to be recognized by the processor.

There are three other ways in which the system can be reset. An external device connected to the I/O expansion port can drive the expansion port's RST line (refer to section 18), the processor can reset the system by executing the RESET instruction (refer to section 1), or the power can be turned off.

Almost any switch with one set of normally open contacts capable of handling logic-level signals can be used as a reset switch. RESET is asserted when the switch is closed, and deasserted approximately 500 milliseconds after the switch is opened.

3-2: Connections (P7)

Figure 3-1 shows the connections between P7 and the reset switch. Refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual for information on mating connectors.

Reset Switch Connections (P7)

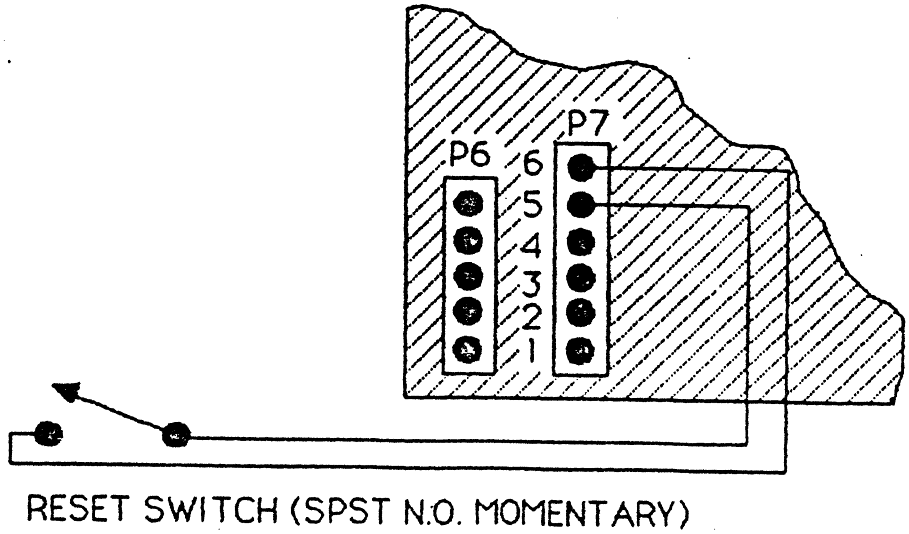


Figure 3-1

NOTES

4-1: General Description

A Single-step/Abort (SS/ABT) switch can be implemented by connecting a user-supplied momentary action switch to the appropriate terminals of connector P7. This switch serves one of two functions, depending on the configuration selected by jumper area JA-5.

When JA-5 is set for the abort mode, activation of the SS/ABT switch will cause a level 7 (non-maskable) autovectored interrupt to the 68020. This interrupt is commonly used by monitor and debugger software to stop program execution.

When JA-5 is set for the single-step mode, the processors HALT line is asserted, causing all processing to stop. Each time the SS/ABT switch is activated, the halt is released until the processor completes its next bus cycle. This mode can be used for low-level hardware debugging.

4-2: Connections (P7)

The Single-step/Abort switch must be a single-pole, double-throw switch capable of switching logic-level signals. Figure 4-1 shows the connections between the switch and connector P7. Refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual for information on mating connectors.

Single-step/Abort Switch Connections (P7)

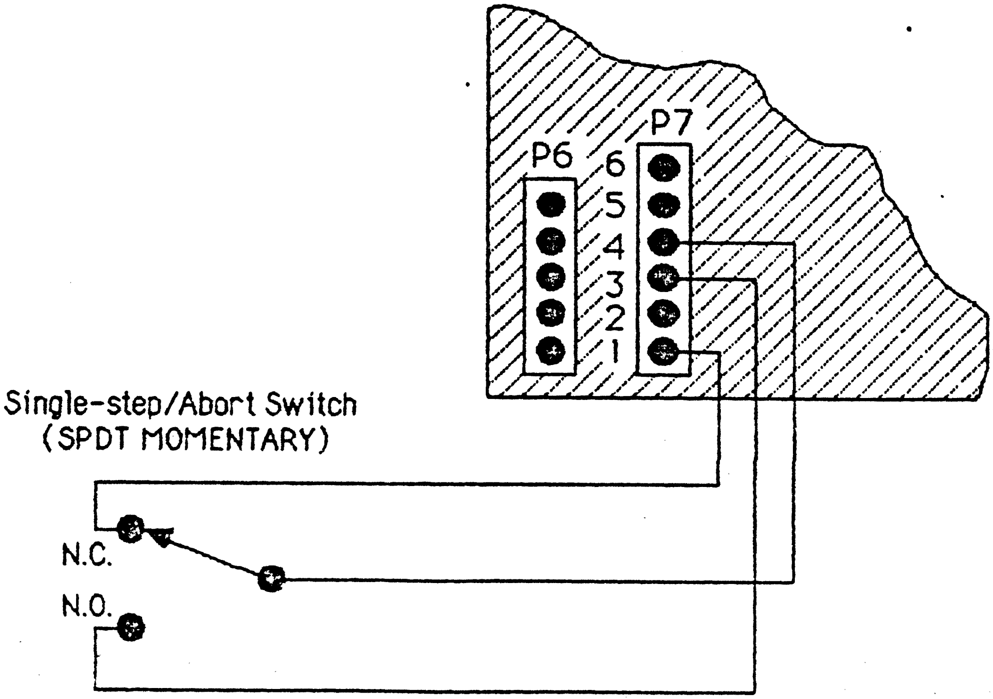


Figure 4-1

4-3: Single-step/Abort Option Jumper

Jumper area JA-5 is used to configure the SS/ABT switch for either the single-step or the abort mode. Figure 4-2 shows the configuration of JA-5 for both modes.

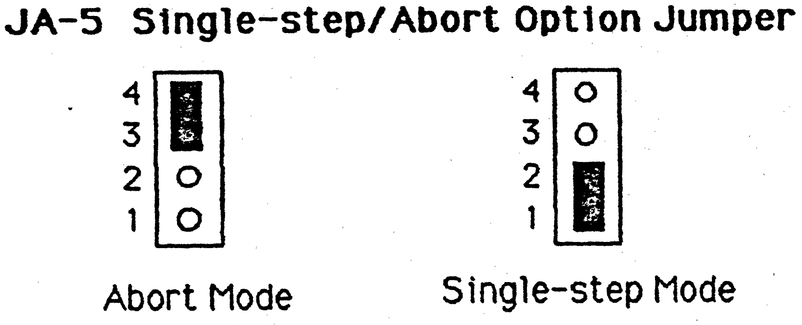


Figure 4-2

4-4: Abort Mode

In this mode, activation of the SS/ABT switch causes a level 7 (non-maskable) interrupt to the processor. The only other device on the GMX Micro-20 capable of generating a level 7 interrupt is the floppy disk controller.

When a level 7 interrupt occurs, the interrupt handler can determine its source from the current status of the DRQ bit and the last value written to the IRQEN bit in the Control/Status Register (CTSR) at address $00FF8004. These two bits share the same location (bit 7) in the register. Reading the CTSR returns the DRQ status and writing controls the IRQEN status. Since the state of the IRQEN bit can not be read directly by the processor, a copy of the last value written to the CTSR or a flag to indicate its current state must be available to the interrupt handler.

If DRQ interrupts are enabled (IRQEN = 1) and the DRQ bit is set (1), the interrupt was caused by the floppy disk controller. Otherwise, the interrupt was caused by the SS/ABT switch. For more information on floppy disk interrupts refer to the 'FLOPPY DISK CONTOLLER' section of this manual.

When an interrupt from the SS/ABT switch occurs, it is necessary to insure that only one interrupt is recognized for each activation of the switch. The interrupt condition is maintained as long as the SS/ABT switch is activated. When the processor services the interrupt it raises its interrupt mask level, precluding further interrupts.

However, if the switch is still activated when the mask level is lowered, another level 7 interrupt will be recognized by the 68020.

To insure that only one interrupt is recognized each time the switch is activated, the interrupt handler must wait for the switch to be released before taking any action that lowers the interrupt mask level (e.g. executing an RTE or MOVE to SR instruction).

The H3 handshake line of the 68230 PI/T indicates the state of the SS/ABT switch. H3 will be low (0) when the switch is in its normal inactive position, and high (1) when the switch is activated. The state of H3 can be read directly as bit 6 of the PI/T Port Status Register (PSR) at address $00FF90CD. Refer to section 12 for further information on the PI/T.

4-5: Single-step Mode

When the SS/ABT switch is configured for the single-step mode, the processor is normally halted (HALT line asserted) and no processing occurs. Each activation of the SS/ABT switch causes the processor to run to the end of the next bus cycle and then halt again.

In single-step mode, the processor halts at the completion of each bus cycle external to the processor, not necessarily at the completion of an instruction. Each 'step' may encompass more than one instruction, if the instructions do not require external bus cycles, or only a portion of an instruction if it causes multiple bus cycles. For example, a program in the processor's on-chip cache that does not generate external accesses can not be single-stepped using this method.

The single-step mode is useful for debugging very low-level hardware problems when even the simplest software routines fail to execute properly. By single-stepping while directly examining the processors address and control signals with an oscilloscope or logic analyzer, the actions of the processor can be traced.

When the single-step mode is selected, the halt indicators (LED1 and the optional remote halt LED) will light, indicating that HALT is asserted. Although HALT is released each time the SS/ABT switch is activated, the LEDs will appear to remain lit because HALT is only released for a short time.

NOTES

5-1: General Description

Connector P6 provides a way to connect a user-supplied, remotely mounted power-on indicator to the GMX Micro-20. The indicator can be almost any standard 1.2 Volt Light Emitting Diode (LED). The LED is connected across the +5V supply through a resistor on the GMX Micro-20, and simply indicates that power is being supplied to the board.

5-2: Connections (P6)

An on-board current limiting resistor (R19) between pin 5 of connector P6 and ground provides current limiting for the power indicator LED. The board is supplied with a 270 ohm resistor at R19, providing approximately 14 milliamps to the LED.

Figure 5-1 shows the connections between P6 and the power indicator LED. Refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual for information on mating connectors.

Remote Power Indicator Connections (P6)

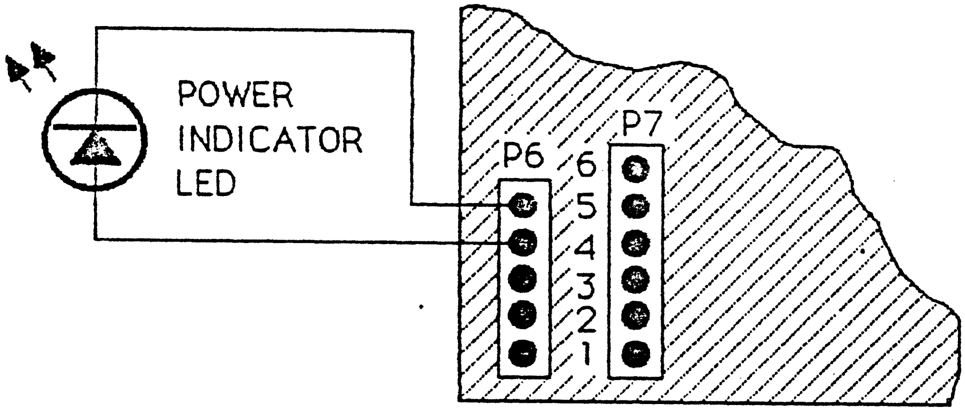


Figure 5-1

NOTES

6-1: General Description

LED1 on the GMX Micro-20 provides an indication that the processor has halted. A second LED (user-supplied), which duplicates the function of LED1, can be wired to connector P6 and mounted in a convenient, easily visible location. The remote indicator can be almost any standard 1.2 Volt LED.

The halt indicator(s) are driven by the processor's HALT line. HALT is a bi-directional signal that is driven low by the processor to indicate that a double bus fault has occurred. HALT can also be driven by the on-board logic, providing the hardware single-stepping capability described section 4.

When the Single-step/Abort Option Jumper is set for the Abort mode, the halt indicator(s) will light if the processor halts due to a double bus fault. When the jumper is set for the Single-step mode, the halt indicator(s) appear to light continuously, although they actually turn off briefly each time the Single-step/Abort Switch is activated.

6-2: Connections (P6)

Both the on-board (LED1) and remote halt indicators are driven by one section of an open-collector TTL buffer (U-7, 7407), with separate current limiting resistors for each LED. A 270 ohm resistor is normally supplied at R18, providing approximately 14 milliamps to the remote LED.

Figure 6-1 shows the connections between P6 and the remote halt indicator LED. Refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual for information on mating connectors.

Remote Halt Indicator Connections (P6)

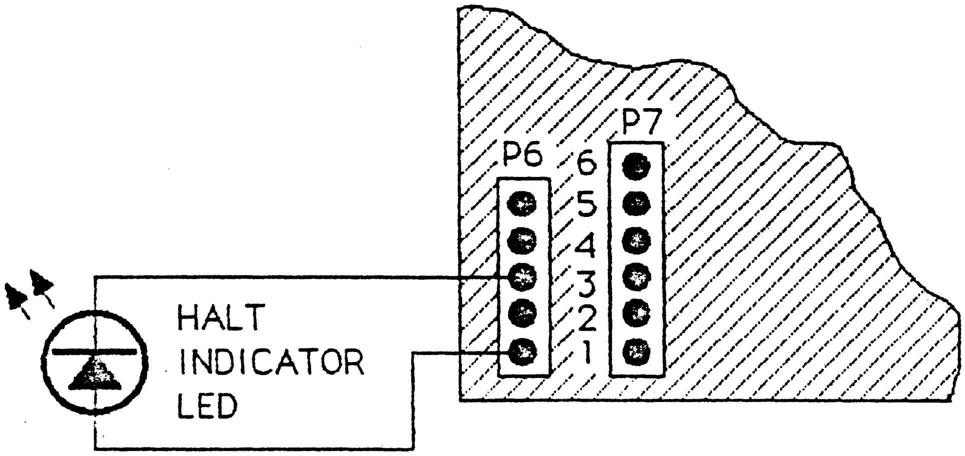


Figure 6-1

NOTES

7-1: General Description

Power for the GMX Micro-20, the Serial I/O Adapter board, and most I/O expansion boards is supplied through power supply connector P8. Two supply voltages are normally required: +5 Volts for the logic and +12 Volts. The GMX Micro-20 itself does not use the +12 Volt supply; however, +12 Volts is required by some serial adapter and I/O expansion boards that derive their power from the GMX Micro-20.

7-2: Voltage and Current Requirements

The GMX Micro-20 requires a well filtered and regulated +5 Volt (+/- 5%) D.C. supply (measured at connector P8). Refer to the serial adapter and I/O expansion board documentation for information on +12 Volt supply requirements.

The power supply current requirements vary, depending on the processor speed and whether or not a coprocessor is installed. Current requirements also depend on the serial adapter board used and on the number and type of I/O expansion boards installed. When calculating the total power supply current, the current requirements of the serial adapter board and any I/O expansion boards powered by the GMX Micro-20 must be added to the figures in the following table.

GMX Micro-20 Basic Power Supply Current

Processor	68881	+5 Volt Supply Current	
Speed	Coprocessor	Typical	Maximum
12.5 MHz	NO	3.4 A	4.1 A
12.5 MHz	YES	3.6 A	4.3 A
16.67 MHz	NO	3.6 A	4.3 A
16.67 MHz	YES	3.8 A	4.5 A
20 MHz	NO	3.8 A	4.5 A
20 MHz	YES	4.0 A	4.7 A

Note: The GMX Micro-20 does not directly use +12 Volts. The +12 Volt requirements depend entirely on the board(s) connected to the I/O expansion and serial I/O connectors (P1 and P2).

7-3: Connections (P8)

Connector P8 is physically and electrically the same as the power connector used on most standard 5 1/4" disk drives. Figure 7-1 shows the pinout of power supply connector P8. For information on mating connectors, refer to the 'SPECIFICATIONS' section at the beginning of this manual.

*********** CAUTION ***********

Be sure that the power supply connector is wired properly and that the voltages are correct before applying power to the board! Incorrect polarity and/or supply voltages will cause extensive damage to the board and components!

****

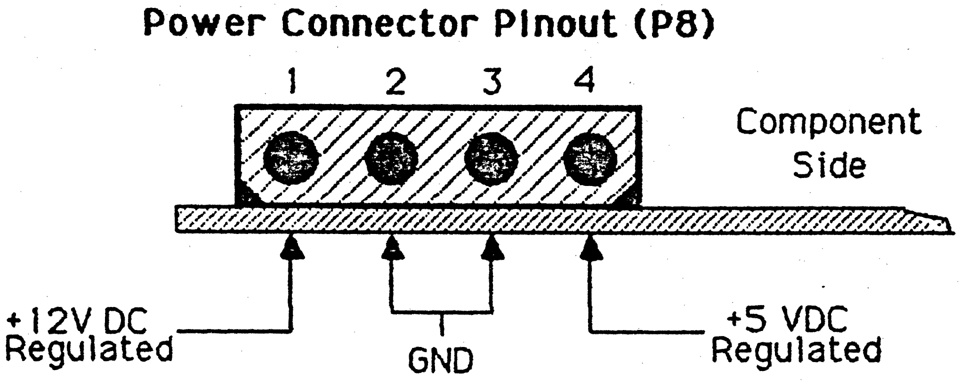


Figure 7-1

SECTION 8: SENSE SWITCH (SW1)

8-1: General Description

The Sense switch (SW1) is a 5-position DIP-switch that can be 'read' by the processor to determine the state of each of the 5 switches. Most of the software supplied by GMX uses the Sense switch at some point to control software options. For example, the 020Bug™ Debugging Package reads switch #1 (SW1-1) to determine whether to enter the debugger on power-up/reset or to jump directly to a location in ROM where an operating system bootstrap loader or user program is located.

Two of the switches, SW1-1 and SW1-2, are dedicated to the 020Bug package and their function can not be user defined if 020Bug is retained. (SW1-1 can, however, cause a user program in ROM to be executed on power-up/reset.)

The remaining three switches (SW1-3,4, and 5) are used by the optional disk operating systems to select operating modes and to configure them for specific hardware options. Their function can not be user defined when one of these operating systems is used. For information on specific switch functions, refer to the software documentation.

8-2: Accessing the Sense Switch

The Sense switch is accessed by reading the on-board Control/Status Register (CTSR) at address $00FF8004. The CTSR is a read/write register that is also used by the floppy disk interface.

The most-significant three bits (b5 - b7) of the CTSR are used by the floppy disk interface and should be ignored when reading the sense switch. The least-significant five bits in the CTSR represent the state of the five sense switches. These bits are set (1) when the corresponding switch is ON (CLOSED) and clear (0) when it is OFF (OPEN). The following table shows the relationship between the CTSR bits and the Sense switches.

CTSR Sense Switch Bits

SW1-X	CTSR bit	
1	b0	0 = ON (CLOSED)
2	b1	
3	b2	1 = OFF (OPEN)
4	b3	
5	b4	

NOTES

9-1: General Description

The on-board RAM consists of 2 Megabytes (2,097,152 bytes) of high-speed dynamic memory, organized as 512K (524,288) 32-bit long words. The processors ability to perform unaligned bus transfers allows the RAM to be accessed without regard to data alignment. For example, word and long word transfers can be made to any byte, word, or long word address. The only restriction on alignment is that instructions must be aligned on word address boundaries.

Although the processor supports unaligned data transfers, such transfers may require multiple bus cycles and can have an adverse effect on performance. For maximum performance, data alignment (i.e. word accesses on word boundaries and long words on long word boundaries) should be maintained whenever possible.

Only one wait-state is added to each bus cycle that accesses the RAM. The RAM is refreshed continuously in a manner that is essentially transparent the user. Two 'CAS-before-RAS' refresh cycles are performed every 30 microseconds, regardless of the processor clock rate.

On power-up reset, the contents of the RAM are undefined. However, once powered up, data in the RAM is not affected by reset or halt conditions. All data in the RAM is lost when power is removed from the board.

9-2: Access Restrictions

Depending on the board configuration ordered, access to the RAM may or may not be restricted with respect to the processor's supervisor and user states.

Most boards, including those supplied with the OS-9/68000™ Operating System and those supplied without an operating system, impose no restrictions on RAM accesses. The entire RAM address space is accessible, for both program and data accesses, in both the user and supervisor states.

A special RAM configuration is used on boards supplied with the non-MMU version of the UniFLEX™ Operating System. These boards restrict user state accesses to the lower 512K bytes of the RAM ($00000000-$0007FFFF). The entire RAM ($00000000-$001FFFFF) is accessible in the supervisor state. Access to all I/O devices (including the I/O Expansion Port) is also restricted to the supervisor state on these boards.

Access permissions for the RAM and I/O devices are controlled by a Programmable Logic Device (PLD), and can only be changed by replacing the PLD (U-25). For further information, contact the factory.

NOTES

10-1: General Description

The GMX Micro-20 has four 28-pin sockets that will accept up to 256K bytes of Read-Only Memory (ROM). The ROM space is organized as 32-bit wide memory. Hardware jumpers configure the board for one of four different ROM sizes (8K x 8, 16K x 8, 32K x 8, or 64K x 8), and for the number of wait-states generated during ROM accesses. Because of the way the ROM is organized, all four sockets must always be used and all four devices must be the same size.

The ROMs are normally mapped high in the processors address space, allowing the system RAM to begin at address zero. However, after power-up or reset the ROMs are temporarily mapped into low memory so the processor can read the restart vector and initial stack pointer from them.

10-2: ROM Types Supported

The GMX Micro-20 can be configured for four different industry standard EPROM types: 2764 (8K x 8), 27128 (16K x 8), 27256 (32K x 8) and 27512 (64K x 8). Note: All four devices must be the same size. Although UV Erasable ROMs (EPROMs) are normally used with the GMX Micro-20, other devices (except RAMs) that are pin compatible with the listed EPROMs could also be used.

The maximum access time of the ROMs must be less than or equal to the maximum times listed in the 'ROM Speed Option Jumper' sub-section. Since slower ROMs require more wait-states, which results in slower program execution, the speed (access time) of the ROMs used can be an important factor in some applications. If the ROMs are used mainly to bootstrap programs that execute from RAM, then it may be possible to use slower ROMs without seriously affecting overall performance of the system. However, if the software executes mainly from ROM, then faster ROMs must be used to obtain maximum system performance.

In time or speed critical applications, the number of wait-states required by the ROMs must also be considered when programs developed and debugged in RAM will eventually be executed from ROM. Since the RAM only requires one wait-state and the ROMs may require up to 5, program execution times may be considerably slower when the program is moved to ROM.

10-3: Size Option Jumper (JA-1A)

Jumper area JA-1A is used to configure the ROM sockets for the size of the ROMs being used. All devices must be the same size and, because the ROM space is organized as 32-bit memory, four devices must always be used. Figure 10-1 shows the configuration of JA-1A for each of the four ROM sizes supported.

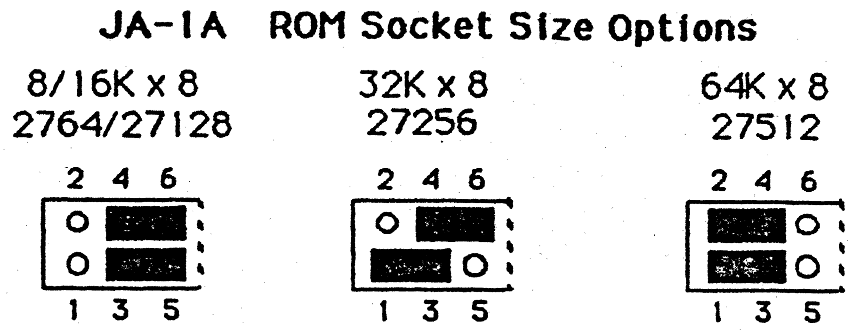
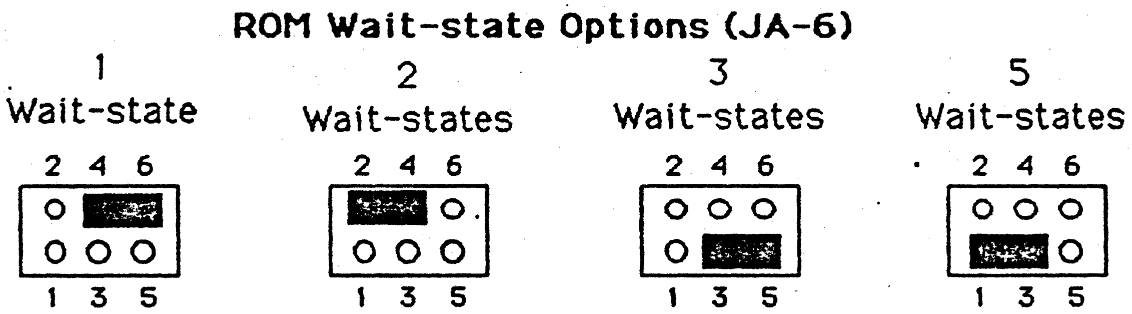


Figure 10-1

10-4: Wait-state Option Jumper (JA-6)

Jumper Area JA-6 is used to set the number of wait-states generated by ROM accesses. The jumper is set according to the access time of the ROMs and the 68020 clock speed.

Figure 10-2 shows the configurations of JA-6, the resulting number of wait-states and the maximum access times for different processor speeds. JA-6 must be set so that the access time obtained is greater than or equal to the maximum access time of the ROMs. For example, to use ROMs with a 300 nanosecond maximum access time, JA-6 must be set for either three wait-states (12.5 MHz 68020) or 5 wait-states (16.67 or 20 MHz 68020). Note: The processor clock speed is one-half of the master clock oscillator (U37) frequency.



Maximum ROM Access Time			Wait-states Required
12.5 MHz 68020	16.67 MHz 68020	20 MHz 68020	
205 ns	150 ns	120 ns	1
285 ns	210 ns	170 ns	2
365 ns	270 ns	220 ns	3
525 ns	390 ns	320 ns	5

Figure 10-2

10-5: Addressing

The ROM sockets occupy 256K bytes of the processor's address space, from address $00800000 through address $0083FFFF. The largest ROMs that the board supports are 64K x 8 devices, and four of them occupy the entire ROM address space. If smaller devices are used, they appear more than once in the ROM address space. For example, if 32K x 8 devices are used they appear from $00800000 through $0081FFFF and again from $00820000 through $0083FFFF.

10-6: Data Organization

The ROM sockets are organized as 32-bit memory, with each of the four sockets connected to 8 of the processors 32 data lines. The following table shows the relationship between the processor data bus and the ROM sockets.

ROM Socket	68020 Data Bus
U-13	D24-D31
U-10	D16-D23
U-8	D8-D15
U-6	D0-D7

This relationship is important since programs and data must be split into four parts and placed in the correct ROMs in order to be read properly by the processor. The following table shows how the long word values $01234567 and $89ABCDEF would be placed in the ROMs so they appear at addresses $80000000 and $80000004 (the first two locations in the ROM space).

ROM ADDRESS	U-13	U-10	U-8	U-6
0	$01	$23	$45	$67
1	$89	$AB	$CD	$EF

10-7: Processor Restart Vectors

The first two long words in the ROMs (addresses $00800000 and $00800004) must contain the initial stack pointer and program counter values to be used by the processor on power-up and reset. The processor expects to find these values at addresses $00000000 and $00000004, which are normally occupied by RAM on the GMX Micro-20. However, immediately after power-up or reset the ROMs are mapped into low memory so the data read from addresses $00000000 and $00000004 will be the data in the first two long words of the ROMs. Once these two long words have been read, normal memory mapping is restored and subsequent reads and writes of these addresses access the RAM.

NOTES

11-1: General Description

The GMX Micro-20 uses two 68681 Dual Asynchronous Receiver/Transmitters (DUARTs) to provide four asynchronous serial I/O ports. These ports support a variety of transmission formats, including: 5 to 8 data bits; odd parity, even parity, no parity, or force parity; one, one and one-half, or two stop bits; and eighteen software selectable baud rates from 50 to 38,400 baud.

Serial data and flow control (handshake) lines are accessed through the serial I/O connector (P2). Each port provides transmit and receive data lines and two handshake lines; one input and one output. One of the ports also has six additional handshake lines (eight total) that can be used in applications such as modem control.

The I/O lines from the DUARTs are unbuffered and connect directly to I/O connector P2, making it possible to adapt the serial ports to almost any serial interface standard (RS-232, RS-422, etc.) by using an appropriate external adapter. Standard adapter boards are available to provide RS-232 level translation.

11-2: Accessing the DUART Registers

Each DUART is controlled by sixteen internal 8-bit read/write registers. These registers are accessed by the processor at sixteen consecutive byte addresses, using standard byte-wide instructions (move.b, clr.b, etc.). The special 'move peripheral' (movep) instructions are not required and should not be used when accessing the DUARTs.

********** NOTE **********

Due to the speed of the 68020 processor, accessing the DUARTs using multiple byte instructions (move.w, clr.l, etc.) or accessing them with consecutive byte-wide instructions, will violate the timing constraints of the device. For this reason, only byte-wide instructions should be used to access the DUARTs, and a 'No Operation' instruction (NOP) must be inserted between any two consecutive instructions that access DUART registers.

****

The following table lists each of the DUART registers and its corresponding address. Port numbering follows the conventions used in the GMX serial adapter board documentation. Port 0 (U-12, side A) is used as the system console port by software supplied by GMX. For a complete functional description of the registers, refer to the manufacturer's data sheet for the 68681 DUART.

Register		DUART #1 (U12)	DUART #2 (U14)
Read	Write	Ports 0(A) & 1(B)	Ports 2(A) & 3(B)
MR1A/2A	MR1A/2A	$00FF8080	$00FF80A0
SRA	CSRA	$00FF8081	$00FF80A1
*	CRA	$00FF8082	$00FF80A2
RBA	TBA	$00FF8083	$00FF80A3
IPCR	ACR	$00FF8084	$00FF80A4
ISR	IMR	$00FF8085	$00FF80A5
CUR	CTUR	$00FF8086	$00FF80A6
CLR	CTLR	$00FF8087	$00FF80A7
MR1B/2B	MR1B/2B	$00FF8088	$00FF80A8
SRB	CSRB	$00FF8089	$00FF80A9
*	CRB	$00FF808A	$00FF80AA
RBB	TBB	$00FF808B	$00FF80AB
IVR	IVR	$00FF808C	$00FF80AC
Input Port	OPCR	$00FF808D	$00FF80AD
Start CTR	OPR Set	$00FF808E	$00FF80AE
Stop CTR	OPR Reset	$00FF808F	$00FF80AF

* These locations are used for factory testing and must not be read.

Note: Each DUART actually occupies 32 bytes of address space, with each of the registers repeated twice. (e.g. DUART #1 MR1A/2A also appears at $00FF8090)

11-3: Baud Rate Clock

Each DUART is provided with a 3.6864 MHz clock on its CLK input pin, allowing any of the 18 standard baud rates supported by the part to be selected under program control.

The standard baud rates are divided into two sets and, while the baud rate for each half of the DUART can be selected separately, both rates must be selected from the same set. Many baud rates are common to both sets; however, some are only available in one of the two. For this reason, it is not possible to select certain combinations of baud rates directly from the standard sets. For example, if set #2 is being used, 19.2K baud is available but 38.4K is not.

In order to overcome this potential problem, the DUARTs built-in counter/timer can be used to add an additional baud rate to either of the standard sets. This is accomplished by programming the timer to use the external clock (X1/CLK), programming it for the appropriate division ratio, and selecting it as the baud rate clock for the desired port.

Since the baud rate clock is normally sixteen times the actual baud rate, the correct division ratio can be found by dividing the input clock frequency (3.6864 MHz) by sixteen times the desired baud rate. For example, the correct divisor for 38.4K baud is: 3,686,400 / (16 * 38,400) = 6. Since the value placed in the counter/timer registers (CTUR and CTLR) must be one-half of the desired divisor, a zero (0) would be written to the upper byte (CTUR), and a three (3) would be written to the lower byte (CTLR) to obtain the correct clock frequency for 38.4K baud.

In addition to the standard 3.6864 MHz clock input, DUART #2 (U-14) also has a 2 MHz clock connected to its IP3 and IP4 input lines. This clock can be used to provide special clock rates for applications such as networking.

11-4: Connections (P2)

Connector P2 on the GMX Micro-20 provides access to the data and selected handshake signals from both DUARTs. Figure 11-1 lists the signals available, and the pinout of connector P2. Be sure to observe proper polarity (pin 1) when making connections to the board.

For information on mating connectors, refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual.

******** CAUTION ********

The serial I/O signals at connector P2 are TTL level signals which must normally be converted to appropriate levels (such as RS-232 levels) by an external adapter board. Damage to the GMX Micro-20 may occur if non-TTL level signals are connected directly to the board.

****

Either a standard GMX interface adapter board or a custom designed board can be used to convert the TTL level signals to the levels required by the interface standard being used. The GMX Micro-20 normally includes an adapter board that converts the signals to RS-232 levels and provides separate I/O connectors for each port. Refer to the interface adapter board documentation for more information.

The actual function of each I/O signal is determined by the by the software being used. For information on signal functions, refer to the software documentation.

11-5: Powering Serial Adapter Boards

As shown in Figure 11-1, both +5 Volts and +12 Volts DC are available at connector P2 on the GMX Micro-20. These supplies are used to power the serial adapter boards supplied by GMX and may also be used to power custom adapters, provided that the current requirements of the adapter do not exceed 1 Amp from the +5 Volt supply and 500 ma. from the +12 Volt supply.

Serial I/O Connector (P2)

Port	68681 Signal	Signal Name	Direction	Pin
Port 0 U12-A	TxDA	TX (0)	From Computer	1
	RxDA	RX (0)	To Computer	3
	OP0	DTR (0)	From Computer	5
	IP0	CTS (0)	To Computer	7
Port 1 U12-B	TxDB	TX (1)	From Computer	9
	RxDB	RX (1)	To Computer	11
	OP1	DTR (1)	From Computer	13
	IP1	CTS (1)	To Computer	15
Port 2 U14-A	TxDA	TX (2)	From Computer	17
	RxDA	RX (2)	To Computer	19
	OP0	DTR (2)	From Computer	21
	IP0	CTS (2)	To Computer	23
Port 3 U14-B	TxDB	TX (3)	From Computer	25
	RxDB	RX (3)	To Computer	27
	OP1	DTR (3)	From Computer	29
	IP1	CTS (3)	To Computer	31
	OP3	RTS (3)	From Computer	33
	OP5	OP5	From Computer	35
	OP7	OP7	From Computer	37
	IP2	DCD (3)	To Computer	39
	IP5	IP5	To Computer	41
	OP2	OP2	From Computer	43
All	Signal and Power Ground Returns			Even Numbered Pins 2 - 44
	+5 Volt DC Supply from Computer			45,47,48
	Not connected, spare			46
	+ 12 Volt DC Supply from Computer			49,50

Figure 11-1

Note: Pin 46 of connector P2 is not connected and may be used as an additional signal line in special applications.

11-6: Interrupts

The Interrupt Request (IRQ) outputs of both DUARTs is permanently connected to the interrupt logic on the GMX Micro-20. A level 3 autovectored interrupt will be generated when either device asserts its IRQ line. Since autovectored interrupts are used, there is no need to program the DUART Interrupt Vector Registers.

Level 3 autovectored interrupts can also be generated by external sources, connected to pin 30 (/681) of the GMX Micro-20 I/O Expansion Connector (P1). When a level 3 interrupt occurs, the interrupt handler must poll the DUARTs, and any devices connected to the expansion connector, to determine the source of the interrupt. The interrupt condition is cleared by taking whatever action is necessary to make the device causing the interrupt deassert its interrupt request output.

NOTES

12-1: General Description

The 68230 Parallel Interface/Timer (PI/T) is a multi-function peripheral device that performs four separate functions on the GMX Micro-20. It includes two 8-bit parallel I/O ports, a counter/timer, and miscellaneous handshake and control lines.

One 8-bit port is used as a general purpose parallel I/O port. This port has eight data lines and two handshake lines, and can be configured for either input or output. External devices such as parallel printers can be connected to this port through the parallel port connector P3.

The second 8-bit port and several of the control lines are used to access the built-in time-of-day clock/calendar on the GMX Micro-20. All of the functions needed to set and read the clock are performed through the PI/T.

The counter/timer and one of the control lines are used to control the GMX Micro-20's periodic interrupt (tick) generator. The control line is used to enable or disable CPU interrupts from the tick generator and the counter/timer can be used to count "ticks" as they occur.

One of the miscellaneous control lines is used to sense the status of the front panel Single-step/Abort switch, allowing software to differentiate between interrupts caused by the switch and those caused by the floppy disk controller.

The remainder of this section describes the basic methods used to access the PI/T and the signal lines used by each of the functions it performs. More detailed information on specific functions, such as accessing the time-of-day clock, is located in the corresponding sections of this manual. For a complete description of the 68230 refer to the manufacturer's data sheet for the device.

12-2: Accessing PI/T Registers

Most of the PI/T registers are byte values and are usually accessed using the appropriate byte-oriented instructions (move.b, clr.b, etc.). However, the timer count and preload registers return 24-bit values from three consecutive registers. Since both of these groups of registers are preceded by a null register, a long word read or write can be used to access the entire 24-bit value with a single instruction (e.g. move.l). Word or long word instructions can also be used to access any two or four consecutive registers in a single instruction.

Note: Since the PI/T registers occupy consecutive byte addresses, the special peripheral access instructions (i.e. MOVEP) are NOT used to access the PI/T.

12-3: Register Addresses

The PI/T has 32 internal 8-bit registers accessible to the user. They appear in 32 consecutive bytes of the processors address space from $00FF80C0 through $00FF80DF. Nine of these registers are unused, and are defined as "null registers". Null registers always return all zeros when read. Writing to a null register has no effect. The following table lists each of the registers and its address. Standard Motorola naming conventions are used. For complete functional descriptions of the registers, refer to the manufacturer's data sheet for the 68230 PI/T.

68230 PI/T Register Addresses

Register Name	Mnemonic	Address
Port General Control Register	PGCR	$00FF80C0
Port Service Request Register	PSRR	$00FF80C1
Port A Data Direction Register	PADDR	$00FF80C2
Port B Data Direction Register	PBDDR	$00FF80C3
Port C Data Direction Register	PCDDR	$00FF80C4
Port Interrupt Vector Register	PIVR	$00FF80C5
Port A Control Register	PACR	$00FF80C6
Port B Control Register	PBCR	$00FF80C7
Port A Data Register	PADR	$00FF80C8
Port B Data Register	PBDR	$00FF80C9
Port A Alternate Register	PAAR	$00FF80CA
Port B Alternate Register	PBAR	$00FF80CB
Port C Data Register	PCDR	$00FF80CC
Port Status Register	PSR	$00FF80CD
Null Register	NULL	$00FF80CE
Null Register	NULL	$00FF80CF
Timer Control Register	TCR	$00FF80D0
Timer Interrupt Vector Register	TIVR	$00FF80D1
Null Register	NULL	$00FF80D2
Counter Preload Register High	CPRH	$00FF80D3
Counter Preload Register Middle	CPRM	$00FF80D4
Counter Preload Register Low	CPRL	$00FF80D5
Null Register	NULL	$00FF80D6
Count Register High	CNTRH	$00FF80D7
Count Register Middle	CNTRM	$00FF80D8
Count Register Low	CNTRL	$00FF80D9
Timer Status Register	TSR	$00FF80DA
Null Register	NULL	$00FF80DB
Null Register	NULL	$00FF80DC
Null Register	NULL	$00FF80DD
Null Register	NULL	$00FF80DE
Null Register	NULL	$00FF80DF

12-4: Interrupts

The PI/T can be programmed to directly cause two different autovectored interrupts to the CPU. If the timer interrupt output (PC3/TOUT) is asserted (active low), a level 4 autovector interrupt is generated. If the port interrupt output (PC5/PIRQ) is asserted (active low), a level 2 autovector interrupt is generated.

The PI/T can also indirectly cause a third level of autovectored interrupt by enabling the periodic interrupt (tick) generator. When the PI/T H4 handshake output is asserted (active low), the tick generator is enabled and periodic level 6 autovector interrupts are generated. Refer to the section 15 of this manual for more information.

12-5: I/O Pin Functions

The following table lists each of the PI/T I/O pins and the function it performs on the GMX Micro-20. The table uses standard Motorola mnemonics for the pin names.

68230 PI/T I/O Pin Functions

Pin No.	Name	Function	Direction	Active Level
4	PA0	Parallel Port Data Line D0	I/O †	x
5	PA1	Parallel Port Data Line D1	I/O †	x
6	PA2	Parallel Port Data Line D2	I/O †	x
7	PA3	Parallel Port Data Line D3	I/O †	x
8	PA4	Parallel Port Data Line D4	I/O †	x
9	PA5	Parallel Port Data Line D5	I/O †	x
10	PA6	Parallel Port Data Line D6	I/O †	x
11	PA7	Parallel Port Data Line D7	I/O †	x
13	H1	Parallel Port Handshake	I	x
14	H2	Parallel Port Handshake	I/O †	x
17	PB0	TOD Clock Data/Address 0	I/O	x
18	PB1	TOD Clock Data/Address 1	I/O	x
19	PB2	TOD Clock Data/Address 2	I/O	x
20	PB3	TOD Clock Data/Address 3	I/O	x
24	PB7	TOD Clock BUSY	I	L
30	PC0	TOD Clock STOP	O	H
31	PC1	TOD Clock WRITE	O	H
34	PC4	TOD Clock READ	O	H
36	PC6	TOD Clock WRITE ADDRESS	O	H
37	PC7	TOD Clock TEST	O	H
32	PC2/TIN	Periodic Interrupt Clock In	I	x
16	H4	Periodic Interrupt Enable	O	L
15	H3	ABORT Switch sense	I	H
33	PC3/TOUT	Level 4 Autovector Interrupt	O	L
35	PC5/PIRQ	Level 2 Autovector Interrupt	O	L
21	PB4	Unused, No Connection	-	-
22	PB5	Unused, No Connection	-	-
23	PB6	Unused, No Connection	-	-

† Jumper programmable for input or output

12-6: Programming Considerations

Since the PI/T is shared by several different functions, programmers must avoid inadvertent changes to registers or parts of registers that control other functions. For example, a printer driver for the general purpose parallel port must not change the state of any of the lines controlling the TOD clock or the tick generator. This can usually be accomplished by reading the contents of the register to determine its current state and then changing only the necessary bits, or by using the appropriate bit-set and bit-clear instructions which automatically perform a similar function and only affect specific bits.

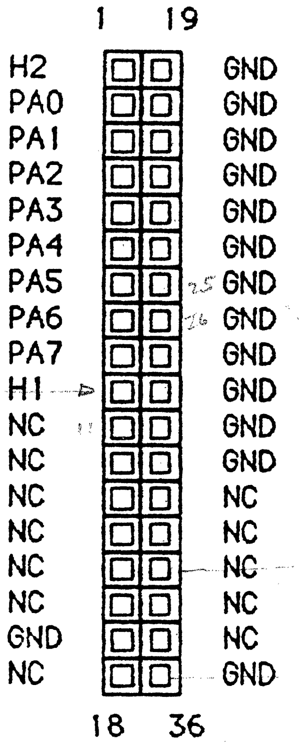
13-1: General Description

The general purpose parallel port provides 8-bit parallel data input or output capability, with provisions for bi-directional handshaking. All of the signals are buffered, with hardware selectable buffer direction for the eight data lines and one handshake line. The other handshake line is permanently configured as an input. The parallel port uses data port A and the H1 and H2 handshake lines of the 68230 PI/T.

13-2: Connections (P3)

Figure 13-1 shows the pinout of connector P3. The pinout of this connector matches the 'Centronics' parallel interface pinout, and a ribbon cable with the appropriate connectors can be used to connect most parallel printers directly to the port. For information on mating connectors, refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual.

Parallel Port Connector (P3)



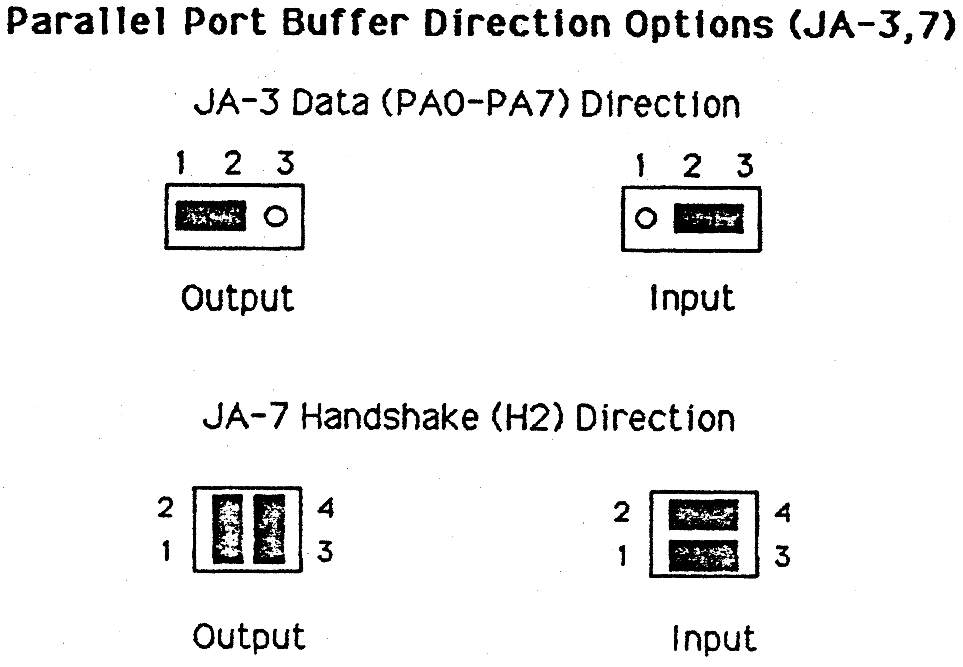
NC = No Connection

Figure 13-1

13-3: Buffer Direction Options

Jumper area JA-3 configures the parallel port for either 8-bit input or 8-bit output. Jumper area JA-7 configures the programmable handshake line (PI/T signal H2) for either input or output. Figure 13-2 details the configuration of JA-3 and JA-7.

Note: For use as an output port for a standard 'Centronics' parallel interface, both jumpers should be set for output.



Parallel Port Buffer Direction Options (JA-3,7)

JA-3 Data (PA0-PA7) Direction

JA-7 Handshake (H2) Direction

Figure 13-2

13-4: Interrupts

The parallel port can generate level 2 autovectored interrupts by asserting the PC5/PIRQ output of the PI/T. The conditions under which interrupts are generated are determined by the programming of the PI/T. Refer to section 12 of this manual and the manufacturer's data sheet for the 68230 PI/T for more information on interrupts.

13-5: Electrical Characteristics

All of the signal lines at parallel port connector P3 are standard TTL level signals (logic 0 = 0.7 volt maximum, logic 1 = 2 volts minimum). As outputs, they are capable of sinking 24 milliamps and sourcing -12 milliamps maximum. As inputs, they require drivers capable of sourcing 20 microamps and sinking -200 microamps maximum.

13-6: Software

The most common application for the parallel port is driving a printer or other output device having a standard 'Centronics' type parallel interface. If one of the optional disk operating systems available from GMX is used, printer drivers are included with the operating system. Refer to the operating system documentation for information on this software.

For other applications, user written software may be needed. If user written drivers are used in conjunction with a standard operating system, be sure that the software does not inadvertently affect any of the other functions that use the PI/T.

NOTES

14-1: General Description

The TOD Clock on the GMX Micro-20 provides complete clock and calendar functions, including automatic leap-year correction. A battery backup system, powered by an on-board nickel-cadmium battery, maintains timekeeping when the system power is off.

The TOD clock uses a 58321 real time clock/calendar chip, available from several manufacturers. The device uses an internal precision time base, and provides data as 4-bit Binary Coded Decimal (BCD) digits. The device is interfaced to the processor through the 68230 PI/T.

14-2: Clock Registers

The TOD clock chip has sixteen internal 4-bit registers that are used to set or read the clock. Individual registers are accessed through the 68230 PI/T in a two step process described in the sub-section on register access. The following table lists the function of each of the registers.

Register Address	Function	Possible Values	Additional Functions
$0	LSD Seconds	0-9	† Bit 3 of register 5
$1	MSD Seconds	0-5	sets 12/24 hour mode
$2	LSD Minutes	0-9	(0 = 12 hr 1 = 24 hr)
$3	MSD Minutes	0-5	† Bit 2 of register 5
$4	LSD Hours	0-9	sets AM/PM (12 hr mode)
$5	MSD Hours †	0-1	(0 = AM 1 = PM)
$6	Day-of-Week	0-6	
$7	LSD Day-of-Month	0-9	§ Bits 2 & 3 of register
$8	MSD Day-of-Month §	0-3	8 select leap year mode
$9	LSD Month	0-9	
$A	MSD Month	0-1	b3 b2 Mode
$B	LSD Year	0-9	----------------------
$C	MSD Year	0-9	0 0 Western
$D	Divider Reset		0 1 Japanese
$E	Ref. Read		
$F	Ref. Read		

LSD = Least Significant Digit
MSD = Most Significant Digit

14-3: Data and Control Signals

The TOD clock chip has four data lines and six control lines that are accessed through ports B and C of the 68230 PI/T. All of the sequencing and timing needed to access the clock must be generated by the software. The following table lists each of the clock signals, along with its function and the corresponding signal on the 68230

PI/T. For further information refer to the manufacturer's data sheet for the 58321 TOD Clock Chip.

Clock Signal	Function	PI/T Line
D0	Data and Address (lsb)	PB0
D1	" " "	PB1
D2	" " "	PB2
D3	Data and Address (msb)	PB3
BUSY*	Low during counter rollover	PB7
STOP	High to stop the clock	PC0
WRITE	High to write data to clock	PC1
READ	High to read data from clock	PC4
WRITE ADDRESS	High to write register address	PC6
TEST	LOW for normal operation	PC7

14-4: Clock Register Access

Register access is a two step process, using the Data/Address and control lines to first select a register and then to read or write the data. In order to access the clock, the PI/T must first be programmed for the proper operating mode. The following code fragment shows the necessary initialization.

NOTE: It is important to remember that the PI/T also controls other functions on the GMX Micro-20. Any programs that manipulates the PI/T must preserve the current state of any PI/T registers or register bits that affect other functions.

```
******************************************************************
********************* PI/T INITIALIZATION ************************
******************** for TOD Clock Access ************************

* First setup PI/T port B mode without affecting other functions
* (H3 and H4) controlled by the Port B Control Register (PBCR)

       ori.b   #%10000000,PBCR  set port B mode 0, submode 1x

* Clear random garbage in ports B and C data registers to prevent
* accidental writes to the clock.

       clr.b   PBDR             clear port B data register
       clr.b   PCDR             clear port C data register

* Now set the data direction for ports B and C to output so we
* can control the clock and select the register we want.

       move.b  #%11010011,PCDDR set PC7,6,4,1,0 to output
       move.b  #%00001111,PBDDR set PB0-3 to output

****************************************************************
```

After the PI/T is initialized, the next step is to write the address of the desired register to the clock chip. Both address and data information use the same four data lines (PB0-PB3) of the PI/T. The following code illustrates the address selection sequence.

```
*****************************************************************
****************** Select TOD Clock Register ******************

* First place the desired register address in the lower 4 bits
* of the port B data register (PBDR). This example selects the
* Day-of-Week register (#6)

       move.b  #6,PBDR         put reg. #6 address on PB0-3

* Next assert the WRITE ADDRESS line (PC6) to strobe the address
* into the clock chip, delay a while, and then remove the strobe
* After releasing the strobe, a short delay (2 NOPs) is added
* to allow for address hold time.

       move.b  #%01000000,PCDR assert PC6 (ADDRESS WRITE)
       bsr     Delay2          delay for at least 2 us.
       move.b  #%00000000,PCDR deassert PC6 (ADDRESS WRITE)
       nop                     allow for address hold time
       nop                       "    "     "      "   "

*****************************************************************
```

Once the desired register address is latched into the clock chip, the next step depends on whether the register operation is to be a read or a write. The following code illustrates a write to a previously selected register. It assumes that the port B data direction register is still set to output after the register select sequence.

```
*****************************************************************
************* Write Data To TOD Clock Register ****************

* First place the desired data in the lower 4 bits of the port
* B data register. d0.b contains the data we want to write.

       move.b  d0,PBDR         put data in port B data reg.

* Next assert the WRITE (data) line (PC1) to strobe the data
* into the clock chip, delay a while, and release the strobe

       move.b  #%00000010,PCDR assert PC1 (WRITE)
       bsr     Delay2          delay at least 2 us
       move.b  #%00000000,PCDR deassert PC1 (WRITE)

*****************************************************************
```

If the operation to be performed on the clock register is a read, then an additional step is needed to reverse the direction of the port B data lines. The following code illustrates the procedure for reading a previously selected clock register.

```
****************************************************************
*************** Read Data From TOD Clock Register ***************

* First set the port B data direction register (PBDDR) for input

        clr.b   PBDDR            set port B for input

* Next assert the READ (data) line (PC4) to place the data on the
* the PI/T inputs, delay a while, read the data, and release the
* strobe.

        move.b  #%00010000,PCDR assert PC4 (READ)
        bsr     Delay6           delay at least 6 us
        move.b  PBDR,d0          read the data into D0
        move.b  #%00000000,PCDR deassert PC4 (READ)

 *   The most significant bit (b7) of the data read from the
 *   PBDR reflects the status of the clock's BUSY bit and may
 *   be either a 0 or a 1. Bits 6, 5, and 4 are undefined and
 *   may also be either a 0 or a 1. These bits must be masked
 *   off or ignored when the data in d0 is used.

 ***************************************************************
```

The preceding code fragments only illustrate the basic requirements for accessing the TOD clock on the GMX Micro-20. They do not take into consideration factors such as clock rollover which are discussed elsewhere.

Where delays are indicated, they are necessary to meet the timing requirements of the 58321 clock chip. The delay times shown are minimum times, longer delays may be used to insure sufficient delay under all conditions. Delay subroutines must be coded to provided the minimum delay times under all conditions and at the maximum intended CPU clock speed.

14-5: STOP function

The TOD clock chip has a STOP function that stops the operation of the clock's internal counters. This function is useful for insuring that the counters do not rollover while the clock is being set, and for starting the clock a specific time.

The STOP function is controlled by the PC0 output of the PI/T. When this bit is asserted (high), internal clock counting is stopped. However, reading and writing of the clock's internal registers is still possible. When the STOP bit is deasserted (low), normal counting is enabled.

A typical use for the stop function is in routines that set the clock. The clock is stopped at the beginning of the routine, remains stopped until all of the registers are set, and is then started when the actual time corresponds to the register values. The following code illustrates the use of the STOP function.

```
*************************************************************
***************** Select TOD Clock Register *****************
************** Using STOP to hold clock counters **************

          move.b  #%01000001,PCDR assert Address WRITE & STOP
          bsr     Delay2          delay for at least 2 us.
          move.b  #%00000001,PCDR deassert Address Write only
          nop                     allow for address hold time
          nop                       "     "     "      "   "

*************************************************************
```

Each subsequent write to the PCDR would have bit 0 set to keep STOP asserted until all of the registers are set. Then the clock is started by clearing bit 0 in the PCDR.

14-6: Clock Rollover

Since the clock's internal counters are normally running while the time is being read, there is a chance that one or more will change (roll over) while the clock is being read, making the data read invalid.

There are several ways to insure that the counters didn't change during a read, invalidating the data. The clock's BUSY output can be used for this function; however, this method requires precise timing and can waste excessive amounts of CPU time. The STOP function can be used to stop the clock's counters during reads; however, this method can reduce time keeping accuracy if the clock is stopped for too long.

The easiest way to insure against rollover is to simply read the data from the clock twice in succession and then compare the two sets of data. If they are the same, the clock didn't roll over and the data is correct. If they are not the same, the process is repeated until the data compares.

14-7: Preventing Unintentional Clock Accesses

Since the levels on the inputs and outputs of the 68230 PI/T are undefined during the transition between powered and unpowered states, occasional glitches may occur on the clock chip inputs when the system is powered up or down.

At the end of any clock access routine, the clock and PI/T should be left in the state illustrated by the following code. This will minimize the chances that a glitch will affect the contents of the clock. It also provides protection against inadvertent modification of the clock's contents by an errant program.

```
******************************************************************
*************** Terminate TOD Clock Operations ****************

* First select clock register $0F (Ref. Read). An accidental
* access of this register won't cause any problems.

       move.b  #%00001111,PBDDR set port B for output
       move.b  #$0F,PBDR        select register $0F
       move.b  #%01000000,PCDR  assert PC6 (ADDRESS WRITE)
       bsr     Delay2           delay for at least 2 us.
       move.b  #%00000000,PCDR  deassert PC6
       nop                      allow for address hold time
       nop                        "     "    "       "    "

* Then leave the port B and C data direction registers set to
* input so writes to PI/T data regs. don't affect the clock.

       clr.b   PBDDR           set port B for input
       clr.b   PCDDR           set port C for input

******************************************************************
```

14-8: TEST signal

The clock's TEST input, driven by PI/T output PC7, enables a test mode used by the manufacturer and must be low for normal operation. On revision 'B' and later PC boards, the trace at W3 can be cut to disconnect TEST from the PI/T and allow PC7 to be user-defined.

14-9: Battery Jumper (JA-2)

Jumper area JA-2 controls TOD clock battery backup. If a jumper is installed at JA-2, the battery is charged when system power is on and timekeeping is maintained when system power is off. If the jumper at JA-2 is removed the battery is disconnected, preventing the battery from being charged and disabling battery backup of the clock.

The battery jumper should be removed before long periods of storage and when the board is being worked on or transported. This helps prolong the life of the battery and minimizes the chances of a short circuit that could damage the battery or the GMX Micro-20.

******** CAUTION ********

Always be careful to avoid short circuits when handling the GMX Micro-20 board. Power is always present on parts of the board, even when the battery jumper is removed. Short circuits can severely damage the battery and other components.

NEVER place the board on a conductive surface, such as a metal table, or wrap the board in metal foil. For static protection, use a high resistance material such as anti-static plastic.

15-1: General Description

The periodic interrupt (tick) generator provides a means of generating interrupts to the CPU at a constant, fixed rate. Periodic interrupts are generally required in a multi-user/multi-tasking environment to tell the operating system when it is time to switch to the next task (time slicing). The tick generator causes level 6 autovectored interrupts at any of 57 jumper selectable intervals ranging from once every 10 microseconds to once every 20 minutes. It uses an internal high-precision crystal oscillator to provide accurate timing of the interrupts.

Tick interrupts are enabled and disabled under program control by the H4 handshake output of the 68230 PI/T. A free running clock at the selected tick frequency is also applied to the timer input (PC2/TIN) of the PI/T. This clock is present at the PI/T input regardless of whether or not tick interrupts are enabled.

Tick interrupts require no special response by the software to clear the interrupt condition once it has occurred. The interrupt circuitry is reset automatically by the processor as it fetches the level 6 interrupt vector.

15-2: Enabling Periodic Interrupts

Level 6 autovector interrupts at the selected tick rate are enabled and disabled under program control by the H4 handshake output of the 68230 PI/T. When the H4 output is in the high state, tick interrupts are disabled. When H4 is in the low state, tick interrupts are enabled. On power-up or after a system reset, the H4 line is high, disabling tick interrupts.

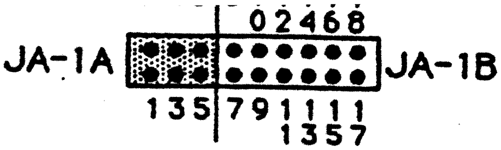
Tick interrupts are totally asynchronous with respect to the system clock and to the tick interrupt enable signal. After tick interrupts are enabled, an interrupt may be generated immediately, or it may take up to one tick period for the first interrupt to occur. Subsequent interrupts will occur at the selected interval.

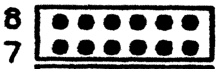
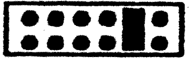
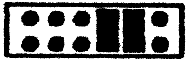
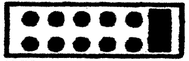
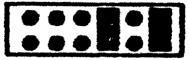
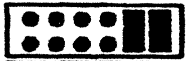
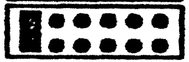
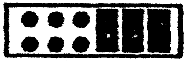
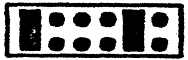
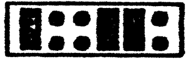
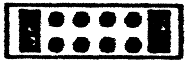
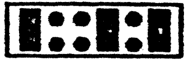
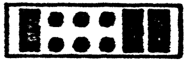
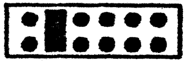
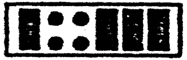
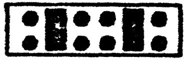
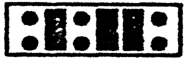
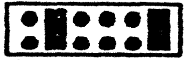
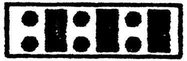
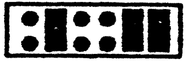
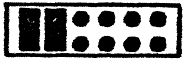
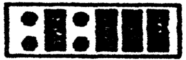
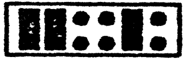
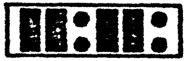
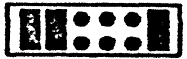
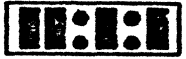
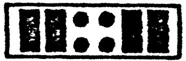
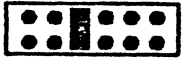
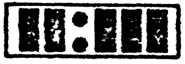
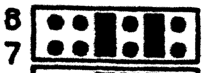
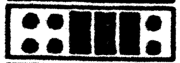
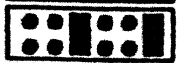
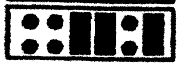
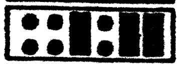
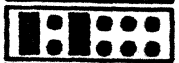
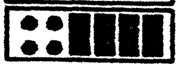
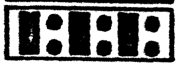
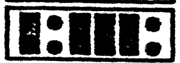
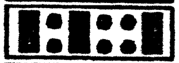
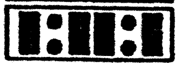
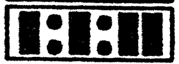
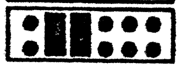
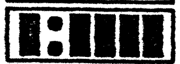
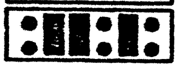
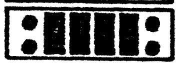
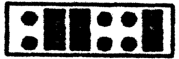
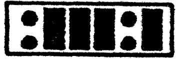
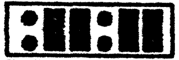
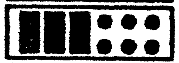
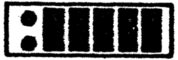
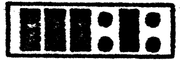
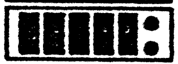
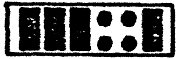
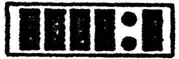
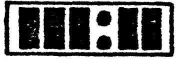
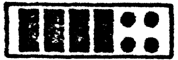
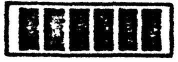
Once tick interrupts have been enabled, they will occur at the selected rate until the system is reset or until the PI/T is reprogrammed to force the H4 output high.

15-3: Interrupt Rate Selection (JA-1B)

The interrupt rate of the tick generator is determined by the positions of the programming jumpers at jumper area JA-1B. There are 6 rate select jumpers at JA-1B, giving 64 possible combinations, of which 57 yield unique tick rates. Figure 15-1 shows each combination with the resulting tick frequency and period.

Periodic Interrupt Generator Rate Options (JA-1B)



JA-1B	Frequency	Period
	100 KHz	10 µs
	50 KHz	20 µs
	33.3 KHz	30 µs
	25 KHz	40 µs
	20 KHz	50 µs
	16.6 KHz	60 µs
	10 KHz	100 µs
	8.3 KHz	120 µs
	5 KHz	200 µs
	3.3 KHz	300 µs
	2.5 KHz	400 µs
	2 KHz	500 µs
	1.6 KHz	600 µs
	1 KHz	1 ms
	833.3 Hz	1.2 ms
	500 Hz	2 ms
	333.3 Hz	3 ms
	250 Hz	4 ms
	200 Hz	5 ms
	166.6 Hz	6 ms
	100 Hz	10 ms ①
	83.3 Hz	12 ms
	50 Hz	20 ms
	33.3 Hz	30 ms
	25 Hz	40 ms
	20 Hz	50 ms
	16.6 Hz	60 ms
	10 Hz	100 ms
	8.3 Hz	120 ms
	5 Hz	200 ms
	3.33 Hz	300 ms
	2.5 Hz	400 ms
	2 Hz	500 ms
	1.6 Hz	625 ms
	1 Hz	1 s
	0.83 Hz	1.2s
	1/2 Hz	2 s
	1/3 Hz	3 s
	1/4 Hz	4 s
	1/5 Hz	5 s
	1/6 Hz	6 s
	1/10 Hz	10 s
	1/12 Hz	12 s
	1/20 Hz	20 s
	1/30 Hz	30 s
	1/40 Hz	40 s
	1/50 Hz	50 s
	1/60 Hz	60 s
	1/100 Hz	100 s
	1/120 Hz	120 s
	1/200 Hz	200 s
	1/300 Hz	300 s
	1/400 Hz	400 s
	1/500 Hz	500 s
	1/600 Hz	600 s
	1/1000 Hz	1000 s
	1/1200 Hz	1200 s

① Standard configuration for OS-9/68000 and UniFLEX

Figure 15-1

15-4: Driving the PI/T Counter/Timer

In addition to generating periodic tick interrupts, the output of the periodic interrupt generator can be used to drive the counter/timer portion of the 68230 PI/T. This function is completely independent of the tick interrupt function, which uses separate interrupt generation circuitry. A free running clock at the selected tick generator frequency is applied to the PC2/TIN input of the 68230 PI/T, regardless of whether or not tick interrupts are enabled (H4 asserted).

The optional OS-9/68000™ and UniFLEX™ operating systems use the PI/T counter/timer, making it unavailable to the user. However, it can be used in applications that do not depend on one of these operating systems.

The operating systems have internal software clocks that use interrupts (ticks) from the periodic interrupt generator as their time-base. The TOD clock is only read once, during system initialization, to set the software clock. All further timekeeping is maintained by the software.

Because a software clock can normally only count interrupts that are actually processed, timekeeping accuracy is lost if interrupts are masked for longer than one tick interval. To maintain software clock accuracy, operating systems for the GMX Micro-20 use the PI/T counter/timer to count ticks that occur while interrupts are masked. The PI/T counter is read each time a tick interrupt is processed, allowing the software to compensate for missed ticks and maintain accurate timekeeping.

NOTES

SECTION 16: FLOPPY DISK INTERFACE

16-1: General Description

The floppy disk interface provides an interface between the GMX Micro-20 and one or two 5 1/4" floppy disk drives. It supports both single density (FM) and double density (MFM) recording at data rates of 125KBits/Sec and 250 KBits/Sec respectively. The interface supports both single and double-sided operation on either 40 Track (48 TPI) or 80 Track (96 TPI) drives.

The interface uses the industry standard 34-pin 5 1/4" floppy disk pinout, and is compatible with most 5 1/4" floppy disk drives. 8" drives and 5 1/4" drives that emulate them by using 8" data rates (500 KBits/Sec) are NOT compatible with the GMX Micro-20.

The basis of the floppy disk interface is a Western Digital WD1772 Floppy Disk Controller/Formatter (FDC), or its equivalent. This integrated circuit provides most of the necessary control functions and has a built-in digital data separator. Additional control functions are provided by several bits in the GMX Micro-20 Control/Status register (CTSR).

16-2: Accessing the 1772 FDC

The 1772 FDC contains four 8-bit registers which are accessible at four consecutive byte addresses beginning at address $00FF8000. Standard byte oriented instructions (move.b, clr.b, etc.) are used to access these registers. The following table lists the address of each of the FDC registers. For a functional description of these registers, refer to the 1772 FDC data sheet.

1772 FDC Register Addresses

Register Function		Address
Read	Write	
Status Register	Command Register	$00FF8000
Track Register	Track Register	$00FF8001
Sector Register	Sector Register	$00FF8002
Data Register	Data Register	$00FF8003

16-3: Control/Status Register Functions

Several bits in the GMX Micro-20 Control/Status Register (CTSR), located at address $00FF8004, are used by the floppy disk interface. The CTSR is an 8-bit read/write register; however, it performs different functions during reads and writes, and values written there can not be read back. Figure 16-1 shows the function of each bit in the CTSR for both reads and writes.

"GMX STATUS"

see pg D-65 for W1772 status

Control/Status Register (CTSR)
Address = $FF8004

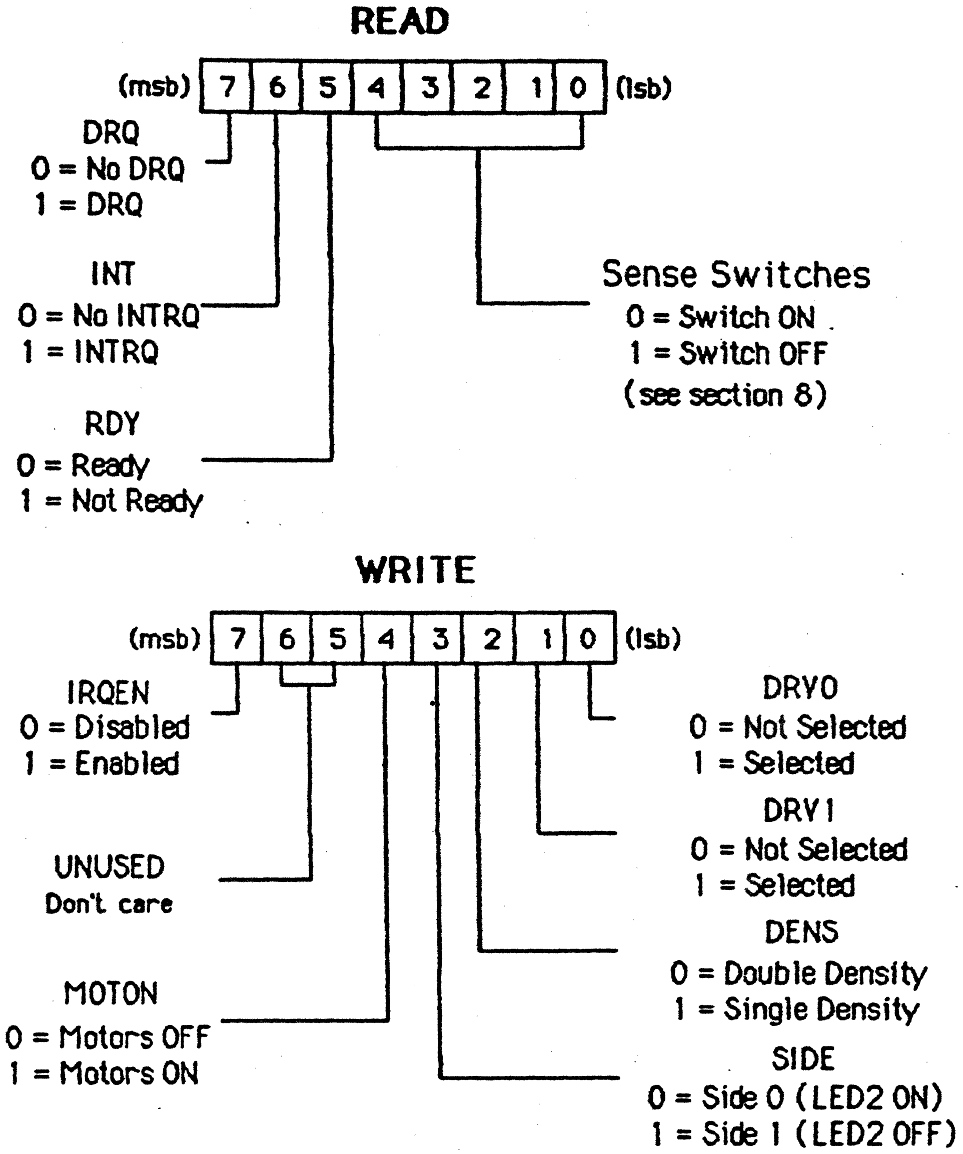


Figure 16-1

All of the write-only bits in the CTSR are cleared (0) by both hardware and software generated system resets.

The following paragraphs describe the functions of each bit in the CTSR that pertains to the floppy disk interface. The five sense switch bits, which are not used by the floppy disk interface, are described in section 8 of this manual.

DRQ (b7 - read only)

This bit reflects the state of the Data Request (DRQ) output of the 1772 FDC. It is set (1) when the floppy controller requests a data transfer during a disk read or write operation, and cleared (0) by reading or writing the FDC data register. Data Request also generates a level 7 autovectored interrupt if DRQ interrupts have been enabled by setting IRQEN (b7) in the CTSR.

INT (b6 - read only)

This bit reflects the state of the Interrupt Request (INTRQ) output of the 1772 FDC. It is set (1) at the completion of any command issued to the FDC, and cleared (0) when the FDC Status Register is read or a new FDC command is issued. A level 5 autovectored interrupt is always generated when the FDC asserts INTRQ.

Note: After power-up or reset the 1772 FDC may assert INTRQ immediately, causing an interrupt that must be cleared during system initialization. Refer to the section 16-4 for more information.

RDY (b5 - read only)

This bit indicates the 'ready' status of the floppy disk drive(s), and is cleared (0) if a selected drive is ready and set(1) if it is not. Since some drives do not provide a ready signal, RDY can be forced to the ready state (0) by setting jumper area JA-8 for drives without ready (see section 16-5).

IRQEN (b7 - write only)

This bit is enables and disables level 7 autovectored interrupts caused by the FDC Data Request (DRQ) output. When set (1) DRQ interrupts are enabled and when cleared (0) interrupts are disabled. It does not affect the DRQ status bit in the CTSR, which always reflects the current state of the DRQ output of the FDC.

MOTON (b4 - write only)

This bit is used to manually control the floppy disk drive motors. When set (1) the drive motors are turned on, regardless of the state of the 1772 FDC Motor On (MO) output. MOTON is logically ORed with the 1772 FDC MO output so that either can start the drive motors, but both must be off to stop them (see section 16-6).

SIDE (b3 - write only)

This bit controls the side select signal, and determines which side of a double-sided drive can be accessed. SIDE is cleared (0) to access side 0 and set (1) to access side 1. SIDE also controls Status Indicator LED2, which is on whenever SIDE is clear (0) and off when it is set (1) (see section 16-11).

DENS (b2 - write only)

DENS drives the 1772 FDC Double-Density Enable (DDEN) input and is is used to select single or double density operation. It must be cleared (0) for double-density and set (1), for single-density operation.

DRV1 DRV0 (b1, b0 - write only)

Each bit selects one of the two drives that can be connected to the interface. Setting (1) DRV0 selects the drive jumpered as drive 0 and setting DRV1 selects drive 1 (see section 16-7). Only one of these bits should be set at a time.

16-4: Floppy Disk Interrupts

The floppy disk interface can generate two different levels of autovectored interrupt. Level 5 interrupt are always generated when the 1772 FDC asserts its INTRQ output at the completion of a command. The interrupt is cleared by reading the 1772 FDC Status Register or by issuing a new command. The INT bit (b6) in the CTSR is set (1) when INTRQ is asserted, and clear (0) when it is not.

******** NOTE ********

The 1772 FDC may assert INTRQ immediately after power-up or reset, causing an interrupt as soon as the processors interrupt mask level is lowered below level 5. Since this interrupt can not be masked by the hardware, it must be cleared during system initialization by reading the 1772 FDC status register.

The 020Bug Debugging Package clears the interrupt during its power-up selftest sequence and no other action is required in systems that have 020Bug installed. Systems not using 020Bug must clear the interrupt or install an interrupt handler before the processors interrupt mask level is lowered and it recognizes the interrupt.

****

The second interrupt that can be generated by the floppy disk interface is a level 7 autovectored interrupt. It indicates that the 1772 FDC has data ready during disk reads or that it is ready for more data during disk writes. If the IRQEN bit (b7) in the CTSR is set (1), a level 7 autovectored interrupt is generated when the 1772 FDC asserts its Data Request (DRQ) output. The interrupt is cleared when the 1772 FDC data register is accessed.

Both the floppy disk interface and the Single-step/Abort switch (in Abort mode) can generate level 7 interrupts, making it necessary to determine which of them caused a particular interrupt. The DRQ bit in the CTSR can be read to determine the current state of the 1772 FDC Data Request (DRQ) output; however, an interrupt will only be generated if DRQ is asserted AND interrupts are enabled (IRQEN set). The interrupt enable bit (IRQEN) in the CTSR is write-only, making it necessary to maintain a copy of the last value written to the CTSR in order to determine the current state of IRQEN.

16-5: Drive Ready Options (JA-8)

Since the 1772 FDC has no provisions for monitoring the 'ready' signal available from many disk drives, a separate ready circuit is included on the GMX Micro-20. The status of the ready signal from the drives is read as bit 5 (RDY) in the CTSR. RDY will be clear (0) if a drive is selected and it is ready, and set (1) if no drive is selected or a selected drive is not ready.

Some drives don't have a ready output and some have the ready output on a non-standard interface pin. The ready circuit only supports drives with a ready signal on pin 34 of the interface cable. Refer to the drive manufacturer's documentation for information on the availability of a drive ready signal.

If a ready signal is not provided by the drive(s) used, the ready circuit can be disabled by setting JA-8 to the 'Drives Without Ready' position. This forces the RDY bit to read 0 (ready), and allows software that checks it to be used with drives that don't have ready outputs. Figure 16-2 shows the configurations of jumper area JA-8 for drives with and without a ready signal on pin 34.

Floppy Disk Ready Options (JA-8)

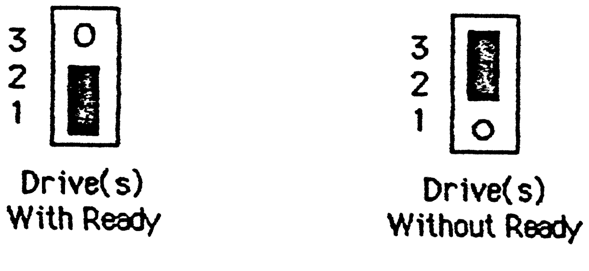


Figure 16-2

The ready signal indicates whether or not a drive is ready for read or write operations. The conditions that cause a ready indication vary between drive models, but generally indicate that there is a disk in the drive, the door is closed, and the motor is up to speed.

In order to test for ready, a drive must be selected and its motors must be turned on. The MOTON bit in the CTSR allows the software to start the drive motors without issuing a command to the 1772 FDC (see section 16-6).

To test for ready, select a drive and start the motors by setting on of the drive select bits (DRV0 or DRV1) and MOTON in the CTSR. Then poll the RDY bit until it is clear (0), indicating that the drive is ready. A time-out is usually included so an error can be reported if the drive isn't ready within a specific length of time. The length of the time-out depends on the application and the drives being used.

16-6: Motor Control

The MOTON bit (b4) in the CTSR is used to start the drive motors without issuing a command to the 1772 FDC. This is necessary in order to check for a drive ready condition, as described in section 16-5.

The signal from the MOTON bit is logically ORed with the Motor-On (MO) output of the 1772 FDC so that the drive motors will be on if either signal asserted.

The 1772 FDC asserts MO when it executes a disk access command, and deasserts it if, after the command is finished, ten disk revolutions occur and another command has not been received.

The MOTON bit in the CTSR is used to start the motors for the ready check and to keep them running until the 1772 FDC takes control by asserting its MO output. MOTON should be cleared at the completion of a disk operation so that the motors will stop when the 1772 FDC times out and deasserts MO.

The 1772 FDC counts index pulses from the selected drive in order to control motor time-out, and the drive must remain selected until time-out occurs or the motors will not turn off. Therefore, at the completion of a disk operation, MOTON should be cleared to allow the motors to time out, but the drive select bit (DRV0 or DRV1) should remain set.

16-7: Drive Select

The DRV0 and DRV1 bits (b0 and b1) in the CTSR are used to select the disk drive to be accessed. In order for a drive to be selected, it must be programmed to respond to the appropriate drive select signal on the interface cable.

If only one drive is used, it is normally programmed to respond to drive select line 0 (pin 10 on the interface cable). This drive will be selected when DRV0 in the CTSR is set (1). If a second drive is used, it should be programmed to respond to drive select 1 (pin 12 on the interface cable). This drive is selected when DRV1 in the CTSR is set (1). Refer to the drive manufacturer's documentation for information on drive programming.

The drive select outputs from the floppy disk interface are qualified by the motor-on signal (pin 16 on the interface cable) so that the select signal to the drive is only asserted when the both the drive select and motor-on signals are asserted.

16-8: Stepping Rates

The 1772 FDC floppy disk controller supports four different disk drive stepping rates. The slowest available stepping rate is 12 milliseconds. The drives used must be capable of stepping at 12 milliseconds or less in order to be compatible with the GMX Micro-20.

The step rate tables in some versions of the Western Digital data sheet for the 1772 FDC are incorrect. The following table lists the correct step rate values for the 1772 FDC. r1 and r0 are the step rate select bits in the command byte. Refer to the 1772 FDC data sheet for more information.

1772 FDC Stepping Rates

r1	r0	Step Rate
0	0	6 ms
0	1	12 ms
1	0	2 ms
1	1	3 ms

Note: If it is necessary to use a GMX Micro-20 with disk drives that are not capable of stepping at 12 ms or faster, a WD1770 floppy disk controller can be substituted for the 1772 FDC normally supplied. The WD1770 is functionally identical to the 1772 FDC, except that it supports step rates as slow as 30 milliseconds.

16-9: Connections (P4)

Disk drives connect to the GMX Micro-20 through connector P4, which uses the industry standard 5 1/4" floppy disk drive pinout. The drives are connected in a daisy-chain arrangement, using a 34-conductor ribbon cable. Be sure to observe proper orientation of pin 1 when connecting the cable to P4.

Figure 16-3 shows the pinout of connector P4. For information on mating connectors, refer to the connector summary in the 'SPECIFICATIONS' section in the beginning of this manual.

Floppy Interface Connector Pinout (P4)

Pin #	Signal	Pin #	Signal
2	No Connection	20	Step
4	No Connection	22	Write Data
6	No Connection	24	Write Gate
8	INDEX	26	Track 0
10	DRV SEL 0	28	Write Prot
12	DRV SEL 1	30	Read Data
14	No Connection	32	Side Select
16	Motor On	34	READY
18	Direction		

All odd numbered pins are grounded

Figure 16-3

16-10: Drive Termination

Floppy disk drives are provided with resistors to properly terminate the interface cable. When a single drive is used the terminating resistors must be installed in the drive. If two drives are used, the terminating resistors must be removed from one of the drives. In two drive systems, only the drive at the end of the interface cable should have terminating resistors installed.

16-11: Status Indicator LED2

Status Indicator LED2 on the GMX Micro-20 is driven by the floppy disk interface's side select signal. This signal is controlled by the SIDE bit (bit 3) in the CTSR and is used to select the desired side of double-sided drives.

LED2 simply indicates the state of the side select signal. When the SIDE bit is set (1), selecting side 1, LED 2 will be off. When it is clear (0), selecting side 0, the LED 2 will be on.

Generally, LED2 serves no useful purpose in connection with the floppy disk interface, although it can be helpful during software debugging. Its primary function is as a low-level fault indicator for self-test and diagnostic software. The 020Bug monitor and diagnostics normally supplied with the GMX Micro-20 use LED2 to signal hardware faults in situations where the normal serial I/O channel might be inoperative.

17-1: General Description

The design of the GMX Micro-20 SASI interface has been optimized for performance, as well as reliability and ease of use. To simplify programming, much of the necessary timing and handshake decoding is handled by the interface logic. Although the interface has an 8-bit (byte wide) data path, up to four bytes can be read or written with a single 'move' instruction, with the interface logic providing the necessary synchronization. To prevent unresponsive or non-existant devices from 'hanging' the entire system, a timeout mechanism generates a Bus Error Exception if the device fails to respond within a specific period.

The GMX Micro-20 can be connected to up to eight intelligent peripheral devices that have SASI (Shugart Associates Standard Interface) or SCSI (Small Computer System Interface) interfaces.

The interface implemented on the GMX Micro-20 is a subset of the interface defined by the ANSI (American National Standards Institute) SCSI interface standard. This subset, commonly called a SASI interface, is the original standard from which the SCSI interface was derived.

The interface on the GMX Micro-20 does not support some of the features of the full SCSI standard. However, if these features are not needed, most SCSI devices can be used in a mode that is compatible with the GMX Micro-20. In fact, many devices described as having a SCSI interface do not support these features either, and are not full SCSI devices.

There are two types of devices associated with a SASI or SCSI interface. A 'host' or 'initiator' is a device that initiates an operation by 'selecting' another device and issuing commands to it. A 'controller' or 'target' is a device that accepts commands from the host and acts on them. Throughout this manual, 'host' will be used to refer to the interface on the GMX Micro-20, and 'controller' will be used to refer to any external device connected to the interface.

The following paragraphs describe the differences between the interface on the GMX Micro-20 and the SCSI standard.

The GMX Micro-20 interface does not support bus arbitration, multiple hosts, or 'disconnect' and 'reconnect'. Only one host (the GMX Micro-20) and up to eight controllers are supported. Once a controller is 'connected' to the bus, it must remain connected until the entire operation has been completed.

The GMX Micro-20 interface does not support the optional ninth (parity) bit on the data bus and parity generation and detection must be disabled on all controllers.

The GMX Micro-20 interface does not support synchronous data transfers. All data transfers are handled asynchronously, controlled by handshake signals from the controller.

The GMX Micro-20 interface does not support the attention signal (/ATN), and therefore does not support the transmission of 'messages' from the host to the controller. However, messages from controller to host are supported.

17-2: Controller Compatibility

While the SASI Interface on the GMX Micro-20 was designed to be as compatible as possible, certain requirements may make it incompatible with a particular controller or application. Since it is impractical to test all of the available controllers, those who wish to use a controller not supported by GMX will have to compare the specifications in this manual with those of the controller and application to determine any incompatibilities. Incompatibilities fall into two categories: functional and timing.

Functional incompatibilities are generally functions of the full SCSI interface that are not supported by the GMX Micro-20 (Refer to section 17-1). In itself, these incompatibilities do not prevent a controller from being used, as long as the unsupported functions are not needed for the intended application.

Timing incompatibilities (especially bus error timing) are the most likely problem area. All of the GMX Micro-20 timing specifications, including select, bus error, and read and write cycle timing, must be compared with the controller specifications to determine whether or not the controller is compatible and will operate reliably.

For example, many of the OMTI 5000 series controllers are incompatible because they do not respond to the select signal (/SEL) fast enough. The delay between the assertion of /SEL by the host and the assertion of /BSY by the controller exceeds the bus error timeout limit.

Appendix E lists controllers that are supported by GMX software or are otherwise known to be compatible with the GMX Micro-20 interface. Additional information on compatible controllers may be found in the disk operating system manuals or may be obtained by contacting the factory.

Note: This does not imply that the controllers listed are the only ones compatible with the GMX Micro-20 or that future revisions by their manufacturers will not make these controllers incompatible.

17-3: Status Register (SASR)

The SASR (SASI Status Register) is a read-only 8-bit register used to obtain the current status of the SASI interface. The SASR is located at address $00FF800E. A read (byte) of this location returns status information; a write causes a bus error.

Only five SASR bits are used for the SASI interface. Of the remaining three bits, bit 6 is used to indicate the presence of an MC68881 floating-point coprocessor. Bits 4 and 5 of the SASR are undefined and may read back as either a 1 or a 0. All three bits must be ignored when determining the SASI interface status. Figure 17-1 shows the arrangement of the bits in the SASR.

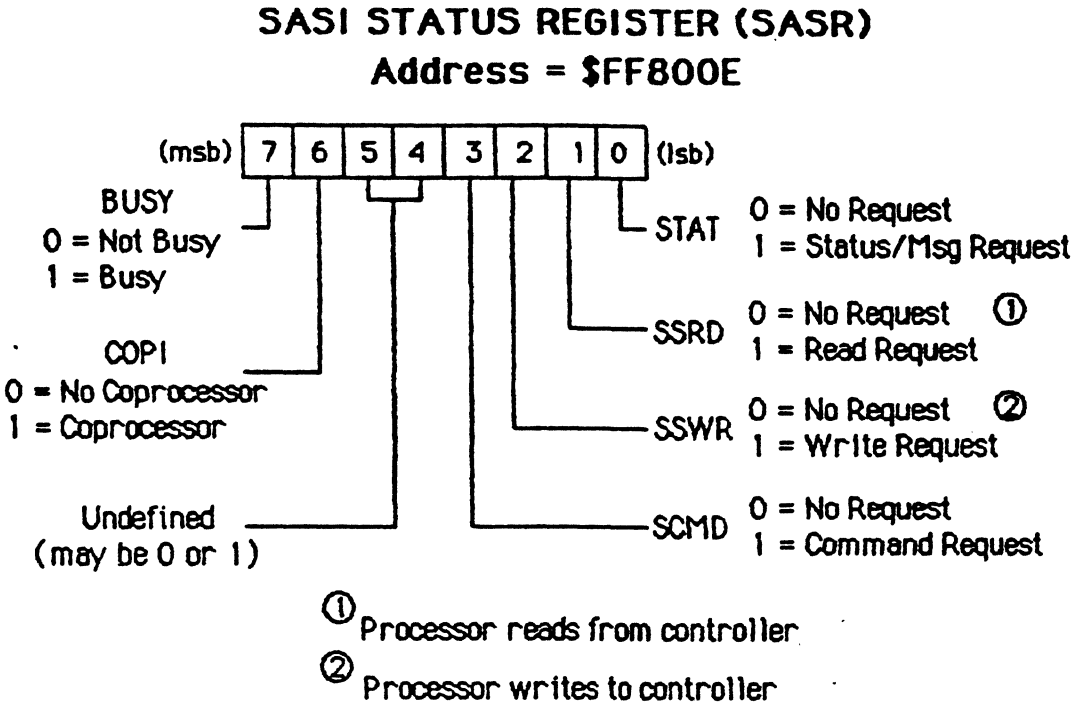


Figure 17-1

The interface logic on the GMX Micro-20 decodes the five handshake signals (/REQ, /CD, /IO, /MSG, and /BSY) from the controller to provide four status bits (SCMD, SSWR, SSRD, and STAT), each indicating a particular phase of the interface protocol. Since the handshake signals are decoded by the interface, the software need only determine which (if any) of the four bits are set (1) to determine if a transfer is being requested by the controller and what type it is.

The fifth status bit (BUSY) directly reflects the current state of the busy (/BSY) signal from the controller. This bit indicates whether the interface is in-use or idle, and must be checked for the idle state before attempting to select a controller.

The following is a description of each of the interface status bits in the SASR.

BUSY (b7)

This bit is an inverted copy of the SASI signal /BSY. If this bit is clear (0), the interface is not in use and is waiting for the host to select a controller. If this bit is set (1), the interface is in use.

SCMD (b3)

This bit will be set (1) when a controller is selected and waiting for a command input from the host. The SCMD bit is derived from the SASI handshake signals as follows:

SCMD = /REQ * MSG * /BSY * /CD * IO

SSWR (b2)

This bit is set (1) when a controller is selected and waiting for data input (other than a command) from the host. The SSWR bit is derived from the SASI handshake signals as follows:

SSWR = /REQ * MSG * /BSY * CD * IO

SSRD (b1)

This bit is set (1) when a controller is selected and waiting to output data (other than status information) to the host. The SSRD bit is derived from the SASI handshake signals as follows:

SSRD = /REQ * MSG * /BSY * CD * /IO

STAT (b0)

This bit is set (1) when a controller is selected and waiting to output status information to the host. The STAT bit is derived from the SASI handshake signals as follows:

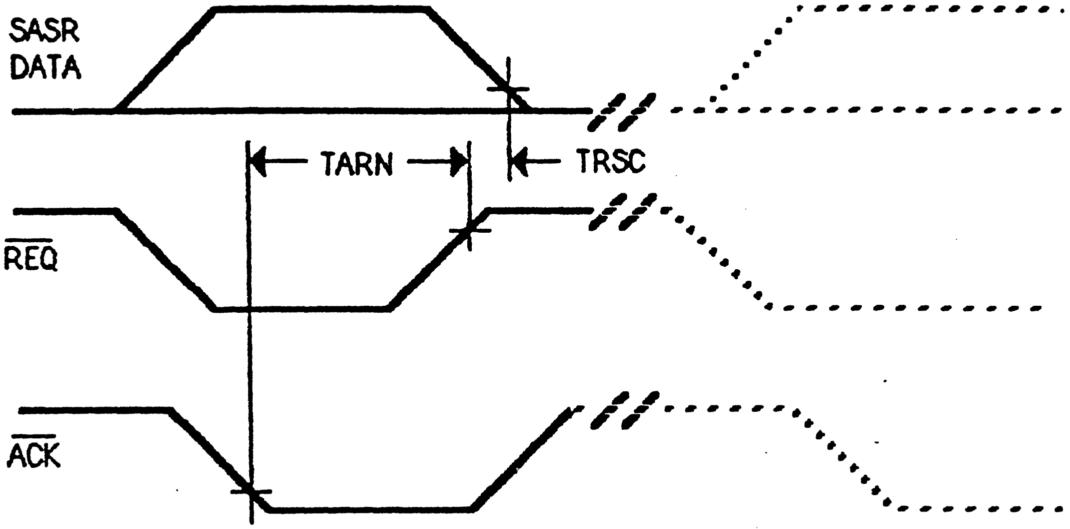
STAT = /REQ * /BSY * /CD * /IO

Note: The /MSG handshake signal is not included in this equation; therefore, STAT will be set for both 'status' and 'message' transfers.

********** NOTE **********

The SASR can be read at any time to determine the interface status. However, due to the speed of the 68020 and interface design, the SASR can be read before the status of a previous operation has cleared, causing a false status indication. To insure that status information is current, delays may be required (in software) to allow the old status to clear before the SASR is read. Delays are only necessary when the SASR is to be read immediately upon completion of an operation on the SASI data bus (e.g., immediately after reading or writing data).

Figure 17-2 shows the timing relationship between the data in the SASR and the interface handshake signals. The amount of delay required is primarily a function of the time required by the controller to release the request signal (REQ) after it has received the acknowledge signal (ACK) from the host. This time, 'TARN', must be be determined from the controller documentation. 'TRSC' represents the delay from the time REQ is negated until the SASR is cleared. This time must be added to the controllers 'TARN' (maximum) to determine the amount of delay, if any, required.



SASI Status Register (SASR) Timing		
TARN	$\overline{ACK}$ to $\overline{REQ}$ negated	Depends on Controller
TRSC	$\overline{REQ}$ negated to Status cleared	83 ns. Maximum

Figure 17-2

17-4: SASI Controller Select Register (SCSR)

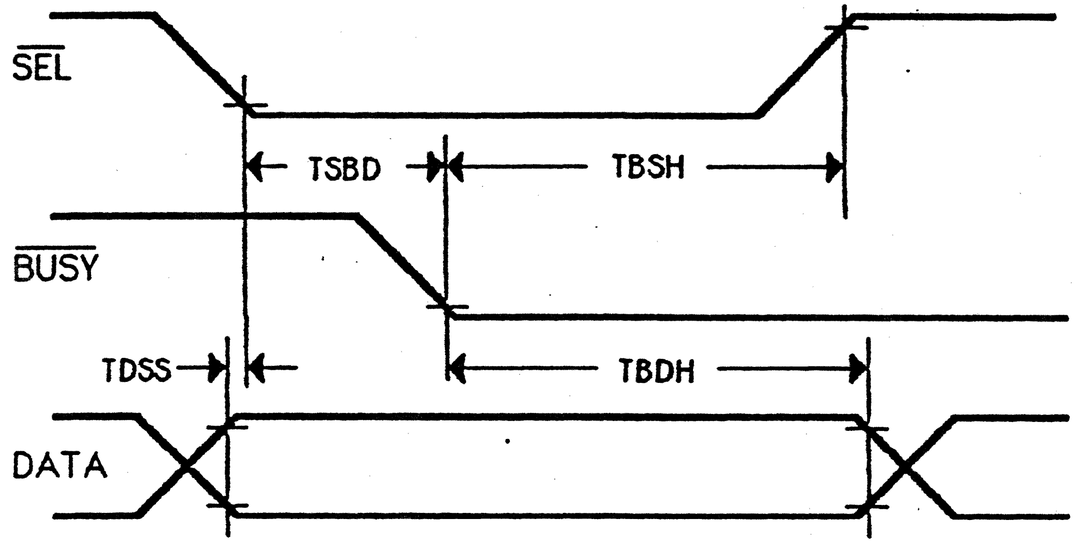
The SCSR is an 8-bit, write-only register used to select one of eight possible controllers connected to the interface. Each controller must be set to respond to one of the eight possible addresses (0-7). For information on controller addressing, refer to the controller documentation. Software provided by GMX requires controllers to be located at specific addresses. Refer to the software documentation for specific addresses.

The SASR is located at address $00FF800D. A write (byte) to this location is used to select a controller, any read of this location will cause a bus error.

The GMX Micro-20 SASI interface logic provides the timing and sequencing necessary to perform the SASI select sequence. The software need only write the appropriate value to the SCSR to indicate the controller to be selected. To select a controller, a one (1) is written to the bit in the SCSR that corresponds to the address of the desired controller. For example, to select the first controller (controller #0) the (binary) value 00000001 must be written to the SCSR, to select the second controller (controller #1) the value 00000010 must be written, etc.

*********** IMPORTANT ***********

The GMX Micro-20 bus error time-out mechanism is used to prevent attempts to select non-existant or defective controllers from 'hanging' the interface, and possibly the rest of the system. When an attempt is made to select a controller (by writing to the SASR) the controller must respond by asserting the /BSY signal within a specified time or a bus error will occur. Figure 17-3 shows the timing generated by the GMX Micro-20 during a select sequence. The controller MUST respond to the assertion of /SEL within 'TSBD', or a bus error will occur.



SASI SELECT TIMING		12.5 Mhz		16.67 Mhz		20 Mhz	
		Min.	Max.	Min.	Max.	Min.	Max.
TSBD	$\overline{\text{SEL}}$ to $\overline{\text{BUSY}}$ Delay		*		*		*
TBSH	$\overline{\text{SEL}}$ hold after $\overline{\text{BUSY}}$		55 ns.		55 ns.		55 ns.
TDSS	DATA setup before $\overline{\text{SEL}}$	205 ns		145 ns		115 ns.	
TBDH	DATA hold after $\overline{\text{BUSY}}$	120 ns		90 ns		75 ns.	

* 61 µs. for Revision A and older boards
125 µs. for Revision B and later boards

Figure 17-3

Figure 17-3 also shows the data setup and hold times provided during the select sequence. When choosing a controller for use with the GMX Micro-20, the controller's timing requirements must be compared with the timing shown to determine whether or not the controller is compatible. Refer to the section 17-2 for more information.

17-5: SASI Interrupt Enable Register (SIER)

The SIER is a write-only location in memory, used to enable interrupts to the processor from the SASI interface. The SIER is located at address $00FF800C. Any write (byte) to this location enables interrupts from the interface , any read of this location will cause a bus error.

No specific data is associated with the SIER. Any write to this location causes interrupts to be enabled. They remain enabled until an interrupt is generated and the processor services it (i.e., fetches the corresponding interrupt vector). Interrupts can only be disabled by the processor, when it services the interrupt. Refer to section 17-9 for more information.

17-6: SASI Data Register (SADR)

The SADR (SASI Data Register) is a read/write memory location used to transfer data between the host and controller over the interface. The SADR is located at addresses $00FF8008-$00FF800B.

Although the interface bus is only 8-bits wide, the design of the interface on the GMX Micro-20 allows 8, 16, or 32-bits to be transferred with a single instruction. The appropriate 'MOVE' instructions (MOVE.B, MOVE.W or MOVE.L) can be used to transfer bytes, words, or long-words to or from the controller. The interface performs all of the necessary timing and sequencing so that, for example, a long-word read of the SADR will sequentially read four bytes from the controller.

In order to use word or long-word transfers, the number of bytes to be transferred must be evenly divisible by the transfer size. For example, a controller might require a ten byte block of parameters as part of its initialization sequence. Since ten is not evenly divisible by four, long-word transfers alone could not be used to transfer the parameters. In this case, two long-word writes and one word write could be combined to move the necessary ten bytes to the controller. Five word transfers could be also be used in this situation. Refer to section 17-11 for more information.

********** NOTE **********

The bus error time-out mechanism on the GMX Micro-20 is used to prevent a defective controller or an improper access sequence from 'hanging' the system. Once an access of the SADR has been initiated by the processor, the controller must respond within a specified period of time or a bus error will occur. For multiple byte transfers (i.e., word or long-word reads and writes), the time-out applies to each successive byte transferred.

****

Figure 17-4 shows the bus error timing for the SADR. SASI PORT SELECT is an internal GMX Micro-20 signal generated by a read or write of the SADR. Once it has been asserted (by the processor executing a read or write of the SADR), the controller MUST respond fast enough so that 'TBER' is not violated. Figure 17-4 also show the requirements for subsequent cycles of multiple byte (word or long-word) transfers. Note that read and write cycles are referenced to opposite edges of the signals. Refer to the controller documentation and section 17-2 for more information.

The data transfer mechanism is interlocked with the SASI bus direction control signal /IO. Any attempt to perform transfers in the wrong direction (e.g., reading when a write is requested) will cause a bus error.

17-7: Controller Reset

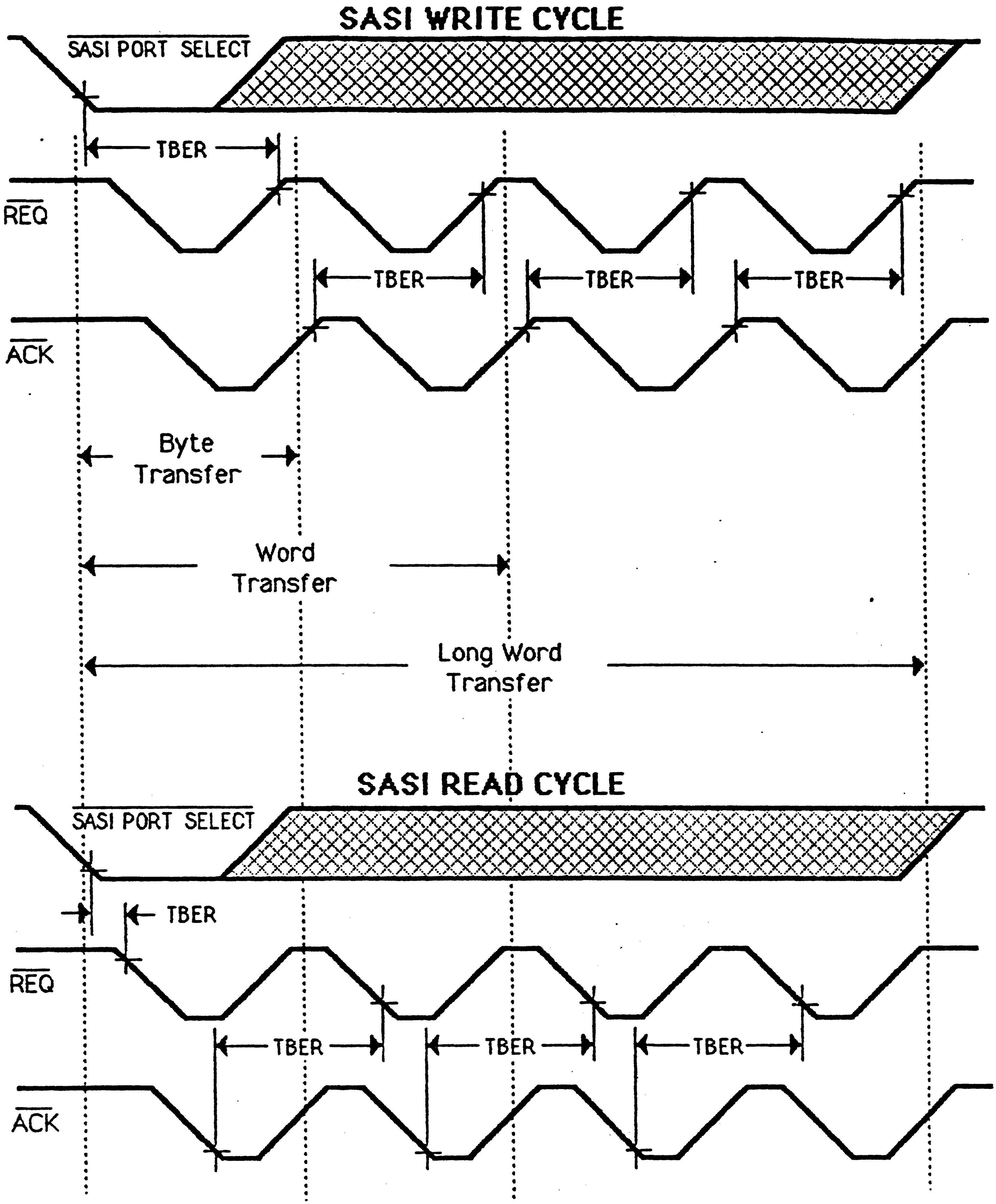
The SASI reset line (/RST) is driven by the internal reset line on the GMX Micro-20. /RST will be asserted whenever a hardware or software reset occurs. There are no provisions for asserting /RST independently. Assertion of the /RST signal causes controllers connected to the interface to be reset, terminating any operation in progress. Refer to the controller documentation for information on the effect of reset on the controller.

17-8: Interface Timing

Figures 17-5 and 17-6 show the general timing relationships between the SASI interface on the GMX Micro-20 and the controller(s). Requirements for both reads (host reads from the controller) and writes (host writes to the controller) are shown. In some cases timing depends on the processor clock frequency, which can be determined by dividing the master clock oscillator (U37) frequency in half.

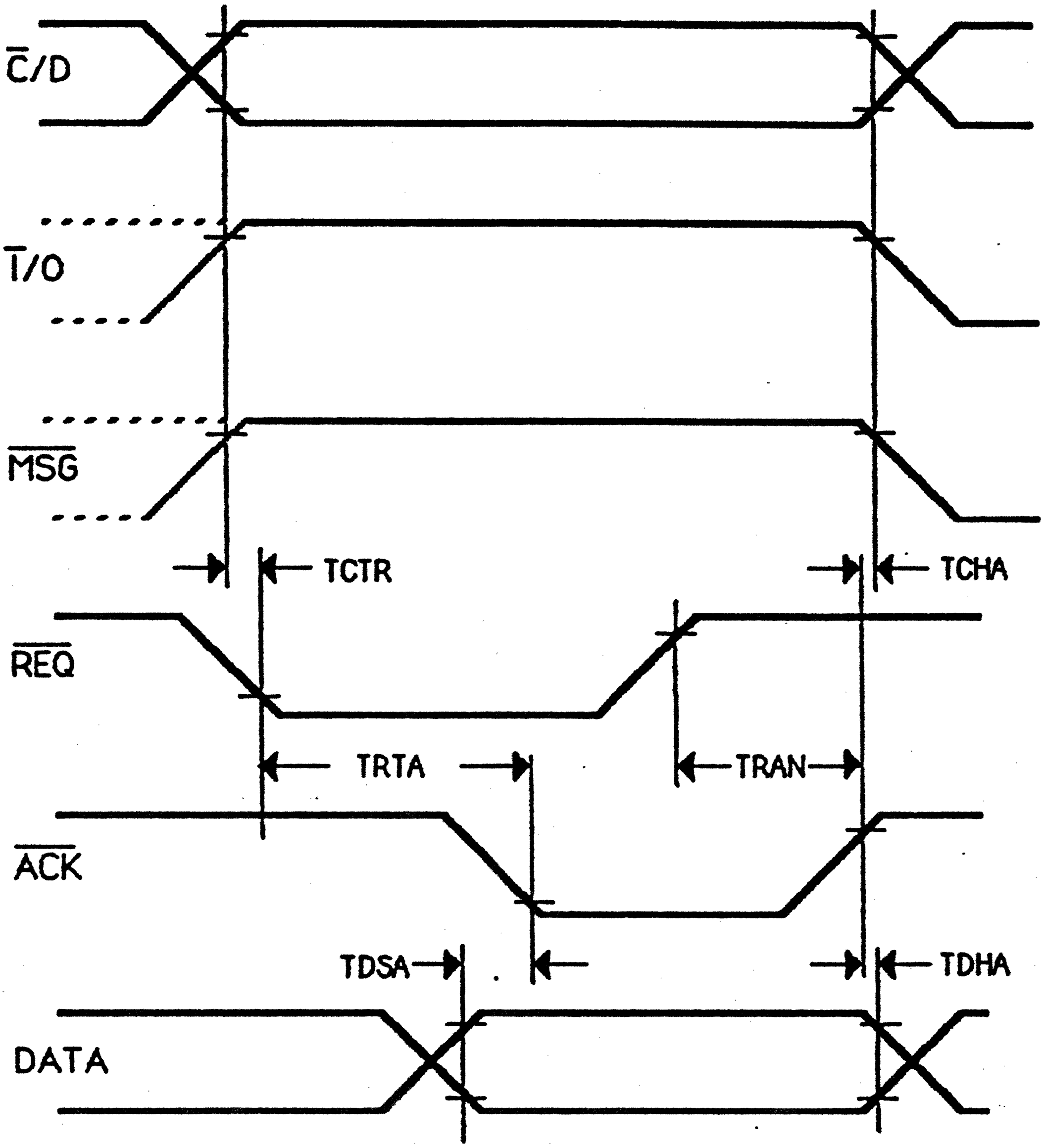
When choosing a controller, the interface's timing requirements must be compared with the controller's specifications to determine whether or not the controller is compatible. For additional information refer to sections 17-2, 17-4, and 17-6.

BUS ERROR TIMING



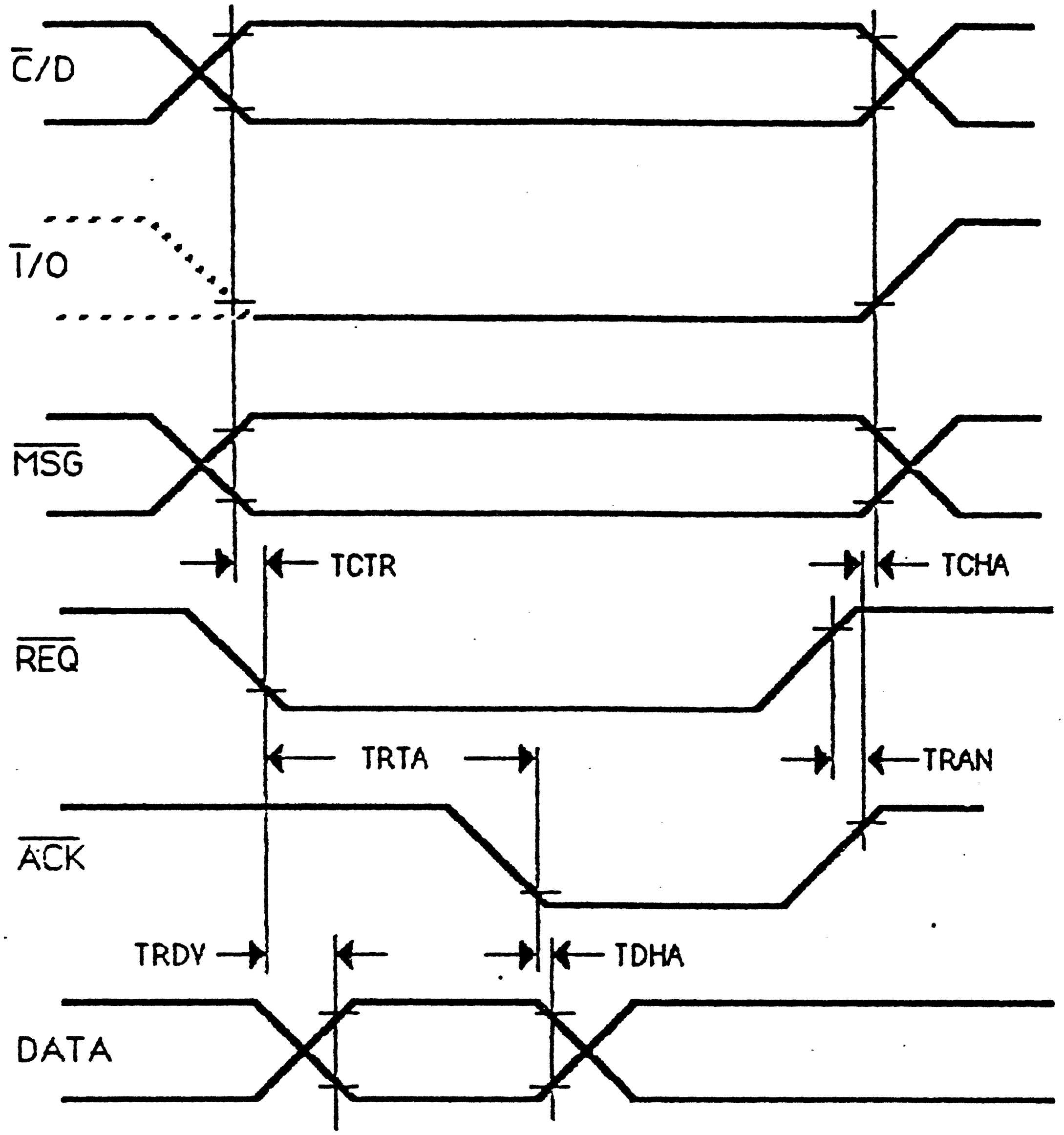
TBER = 61 µs. Maximum for Revision A and older boards
125 µs. Maximum for Revision B and later boards

Figure 17-4



SASI WRITE TIMING		12.5 Mhz		16.67 Mhz		20 Mhz	
		Min.	Max.	Min.	Max.	Min.	Max.
TCTR	Control setup before $\overline{REQ}$	0		0		0	
TCHA	Control hold after $\overline{ACK}$ negated	0		0		0	
TRTA	$\overline{REQ}$ to $\overline{ACK}$ asserted	40 ns.		40 ns.		40 ns.	
TRAN	$\overline{REQ}$ negated to $\overline{ACK}$ negated	130 ns.		105 ns.		95 ns.	
TDSA	DATA setup before $\overline{ACK}$	10 ns.		10 ns.		10 ns.	
TDHA	DATA hold after $\overline{ACK}$	.0		0		0	

Figure 17-5



SASI READ TIMING		12.5 Mhz		16.67 Mhz		20 Mhz	
		Min.	Max.	Min.	Max.	Min.	Max.
TCTR	Control setup before $\overline{\text{REQ}}$	0		0		0	
TCHA	Control hold after $\overline{\text{ACK}}$ negated	0		0		0	
TRTA	$\overline{\text{REQ}}$ to $\overline{\text{ACK}}$ asserted	110 ns.		85 ns.		75 ns.	
TRAN	$\overline{\text{REQ}}$ negated to $\overline{\text{ACK}}$ negated		45 ns.		45 ns.		45 ns.
TRDV	$\overline{\text{REQ}}$ to DATA valid		45 ns.		35 ns.		35 ns.
TDHA	DATA hold after $\overline{\text{ACK}}$	0		0		0	

Figure 17-6

17-9: Interrupts

The SASI interface is capable of interrupting the processor when a data or status transfer is requested by a controller. Command transfers can not generate interrupts. If interrupts are enabled a level 1 Autovectored interrupt is generated when the controller requests a data transfer to or from the host, or when it requests a status transfer to the host.

Interrupts from the SASI interface are enabled by writing any value to the SIER (refer to section 17-5). Once enabled, an interrupt will be generated when one of the interrupt conditions described previously occurs. If an interrupt condition is already present (e.g. a data transfer is being requested by the controller) when the SIER is written to, an interrupt will be generated immediately.

SASI interrupts are disabled automatically as soon as the processor services the interrupt. No further interrupts are generated until the SIER is written to again. If the processor does not service the interrupt (i.e., the processors interrupt mask level is set at 1 or above) and the interrupting condition is subsequently cleared, no interrupt will be generated and interrupts will remain enabled.

17-10: Controller Addressing

Controllers are set to respond to a particular address (controller number) by jumpers or switches on the controller. Addresses must be set to match the software drivers being used and if multiple controllers are used each must have a unique address.

Most software, including that supplied by GMX, requires that the controller(s) be set for a particular address or addresses. Refer to the software documentation for information on specific addresses.

The addresses used by GMX are assigned sequentially, starting with controller address zero. In order to minimize the potential for conflicts with current or future software releases, users who write their own drivers should avoid addresses already assigned by GMX. We recommended that users assign addresses in reverse order, starting with address seven.

17-11: Programming Considerations

The design of the GMX Micro-20 SASI interface greatly simplifies the task of writing software drivers. Since the handshake signals are decoded in hardware the programmer need not deal with the individual signals. Each phase of bus operation is indicated by a single bit in the SASR (plus the BUSY bit which is always set (1) while the bus is in use). Optimum performance is achieved through the use of multiple-byte data transfers and loop structures to move data to or from the controller. The following paragraphs list some recommended procedures for using the interface.

The BUSY bit (bit 7) in the SASR is used to arbitrate interface use between multiple programs or tasks. The BUSY bit is set (1) from the time a controller is selected until the entire command sequence is completed. All driver software must test the status of the BUSY bit before attempting to use the interface. If the interface is in use (BUSY = 1) the program must wait until the BUSY bit is clear (0) before attempting to select a controller.

In an interrupt driven situation, interrupts should be masked during the BUSY test and controller selection to insure that multiple processes do not attempt to access the interface simultaneously. The flow chart in Figure 17-7 illustrates the recommended sequence for checking BUSY and selecting a controller.

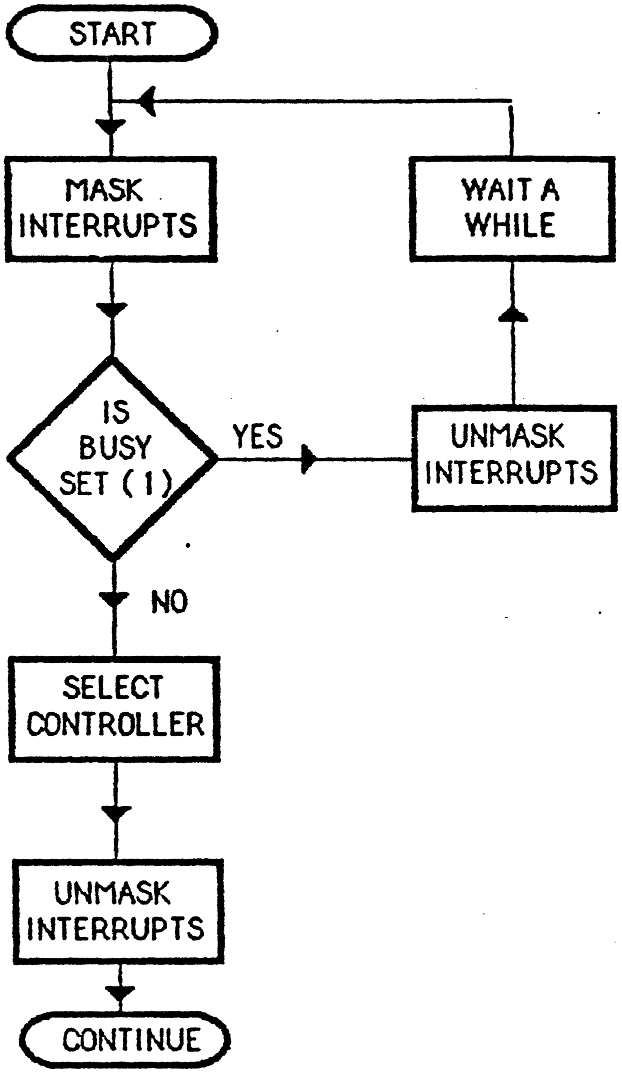


Figure 17-7

There are two basic approaches to transferring data on the interface. In the first approach, the status register (SASR) is tested before each byte is transferred to determine the type (command, data, or status/message) and direction of a transfer request. This method is relatively slow and inefficient, since it does not use the multiple-byte transfer capabilities of the interface and requires a read of the status register for each byte transferred. Its only advantage is that it does not require prior knowledge of the total number of bytes to be transferred. Normally, this approach should only be used when the transfer size can not be determined in advance. The following code fragment demonstrates this approach:

```
* This code checks the SASR to determine what type
* of transfer the controller is requesting. It loops
* until a request is made and then processes it.
* It assumes that a command has already been passed
* to the controller and does not check the SCMD bit.

TestStat
        btst.b  #2,SASR         check for SSWR bit set
        bne     WritData        Yes, process write
        btst.b  #1,SASR         check for SSRD bit set
        bne     ReadData        Yes, process read
        btst.b  #0,SASR         check for STAT bit set
        bne     ReadStat        Yes, process status
        bra     TestStat        else, try again

* The following code writes one byte of data to the
* interface from the buffer pointed to by (a0),
* then loops back to wait for the next request
* from the controller.

WriteData
        move.b  (a0)+,SADR      move one byte

* Note: A delay may be required here to allow the
*       SASR to clear before it is read again.
*       The amount of delay (if any) depends on the
*       controller being used.
*       See section 17-3 for more information.


        bsr     Delay           wait for SASR to clear
        bra     TestStat        look for next request

* Additional routines here to process read data and
* status/message requests.
```

In most cases the number of bytes to be transferred is known, and the second approach uses a loop-and-counter method to transfer data. Using this approach, the status register only needs to be tested at the beginning of each phase of the transfer. Once the type of transfer is determined from the SASR, the program enters a loop to transfer the data. The loop need only read or write data until the correct number of bytes have been transferred. The status register need not be read again until the transfer is completed. Since the interface will automatically synchronize the transfers with requests from the controller, byte, word, or long-word transfers may be used as appropriate.

The following code fragment replaces the 'WritData' routine in the previous example, and illustrates the recommended method of transferring data using the loop-and-counter approach. By reversing the 'direction' of the move instruction, the same code would be used to read data from the controller.

```
* This code assumes a transfer size of 256 bytes,
* a typical 'sector' size, and transfers the data
* using long-word moves for maximum efficiency.

WritData
        move.l  (256/4)-1,d0    set loop counter (63)

WritLoop
        move.l  (a0)+,SADR       move 4 bytes
        dbra    d0,WritLoop      loop till done

* See comment in the previous example regarding delays

        bsr     Delay            wait for SASR to clear
        bra     TestStat         look for next request
```

Since the delay between the time a command is issued and the time the controller is ready to accept or return data is often long, interrupts are usually used to signal the driver when the data can be transferred. While waiting for the interrupt to occur, other processing can take place.

The SASI interface can be programmed to generate a level 1 autovectored interrupt on a request for data (read or write) or status/message transfers. Requests for command transfers can not generate interrupts.

Since interrupt processing is highly system dependent, only general information concerning the use of interrupts is included in this manual. Refer to sections 17-5 and 17-9 for additional information.

Assuming the driver has access to the interface and has selected a controller, it first waits for the controller to request a command transfer, and then issues the desired command. After the command has been issued, the driver should (if necessary) set any flags needed to indicate to the system which driver and controller are using the interface. (This must be done before enabling interrupts, to prevent the interrupt from occurring and being processed before the flags are set.) Once these preparations are complete, the driver can then enable interrupts by writing to the SIER. Once interrupts have been enabled the driver can deactivate itself to allow other processing while it waits for the interrupt.

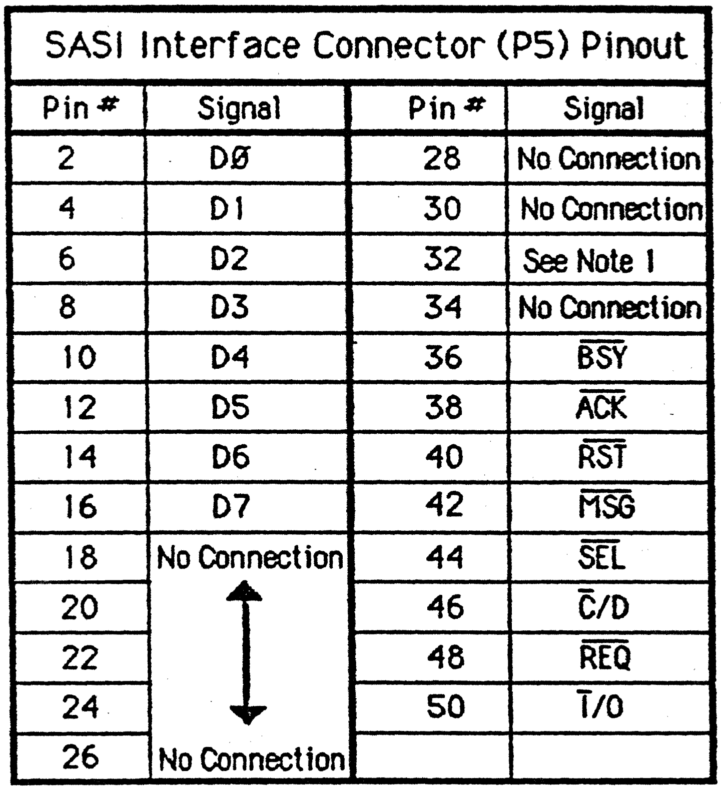
In most applications the interrupt handler does not perform the actual data transfer. It simply notes that an interrupt has occurred and reactivates the driver, which then finishes processing the command and releases the interface. If necessary, the interrupt handler can use flags set by the driver(s) to determine which driver and/or controller is responsible for the interrupt.

No action by the interrupt handler or the driver is required to disable interrupts from the SASI interface. They are automatically disabled when the processor services the interrupt and only one interrupt can be generated each time they are enabled.

The SASI interface is the only source of level 1 interrupts, so no means is provided for determining their source. If a level 1 interrupt is received, it can be assumed to have come from the SASI interface.

17-12: Connections (P5)

Figure 17-8 shows the pinout of the SASI interface connector (P5). For information on mating connectors, refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual. Be sure to observe proper polarity (pin 1) when connecting the interface cable and controller(s).



SASI Interface Connector (P5) Pinout

Pin #	Signal	Pin #	Signal
2	DØ	28	No Connection
4	D1	30	No Connection
6	D2	32	See Note 1
8	D3	34	No Connection
10	D4	36	$\overline{BSY}$
12	D5	38	$\overline{ACK}$
14	D6	40	$\overline{RST}$
16	D7	42	$\overline{MSG}$
18	No Connection	44	$\overline{SEL}$
20	↑	46	$\overline{C}$/D
22		48	$\overline{REQ}$
24	↓	50	$\overline{I}$/O
26	No Connection		

Notes:

1: Terminated through 150 Ω to +5V

2: Odd numbered pins, except 25, are grounded

3: Pin 25 is not connected

Figure 17-8

With the exception of the data bus parity (DBP) and attention (/ATN) signals, which are not supported by the GMX Micro-20, P5 matches the pinout of the standard SASI/SCSI interface.

The interconnecting cable (not supplied) should be 50-conductor flat or twisted-pair ribbon cable and must be limited to a maximum length of 20 feet (6 meters). If multiple controllers are used, they are all connected to P5 in a daisy-chain arrangement.

17-13: Electrical Characteristics

All of the input signals (/REQ, /IO, /CD, /MSG, /BSY, and D0-D7) are TTL level inputs terminated with 220 ohms to VCC (+5V) and 330 ohms to ground. The the controller outputs driving these lines must be capable of sinking 24 ma. to insure proper signal levels.

The three control signal outputs (/ACK, /SEL, and /RST) are open-collector TTL level outputs, capable of sinking 40 ma. per output. No internal termination of these lines is provided, they must be terminated by the controller for proper operation.

The SCSI attention signal (/ATN) is not used by the interface. However, /ATN is terminated with 150 ohms to +5V on the GMX Micro-20 to minimize noise pickup on the unused line.

The output drivers for the eight data lines (D0-D7) are TTL level drivers, capable of sinking 48 ma. per output. However, since these signals are terminated on the GMX Micro-20, approximately 24 ma. of output drive is available for external devices.

17-14: Cable Termination

The SASI bus must be terminated at both ends for proper operation. The GMX Micro-20 provides the necessary termination, and should normally be located at one end of the cable. The controller installed at the opposite end of the cable must also provide termination. If only one controller is used, it must have terminators installed. If multiple controllers are used, the last controller on the cable must be the only one with terminators installed. All other controllers must have their terminating resistors removed or otherwise disconnected from the bus. Refer to the controller manufacturer's documentation for information on termination options.

Note: The GMX Micro-20 does not support the SCSI terminator-power option on connector P5. Power for external terminators must be provided by the terminating device.

NOTES

18-1: General Description

The I/O Expansion Port provides a way to connect off-the-shelf or custom designed peripheral devices to the GMX Micro-20. The I/O Expansion Connector (P1) provides access to the signals needed to support both 8 and 16-bit devices. The expansion connector also provides both +5 and +12 volts DC which may be used to power the expansion board(s).

Although 4K (4096) bytes of the processors address space are allocated to the expansion port, the standard expansion connector only provides ten address lines. This allows a maximum of 1K (1024) bytes of address space to be decoded. If additional space is required, two undefined pins on the expansion connector can be used to provide additional address lines. Refer to the memory map in appendix C for the location of the expansion port in the processor's address space.

18-2: Interface Protocol

The expansion port uses an asynchronous interface protocol. All timing relationships are based on the assertion of the Select signal (/SEL) by the host and the subsequent assertion of one of the Data Transfer and Size Acknowledge (DSACK) signals (/DSK0 or /DSK1) by the peripheral device.

/SEL is asserted by the host in response to a read or write that falls within the expansion port's address space. After /SEL is asserted the peripheral must either accept or provide data, based on the state of the Read/Write line and then assert one of the DSACK lines to terminate the bus cycle. The size of the transfer is indicated by the DSACK line used. If a DSACK line is not asserted within the bus error timeout period a the bus cycle is aborted with a bus error.

18-3: Read Cycle Timing

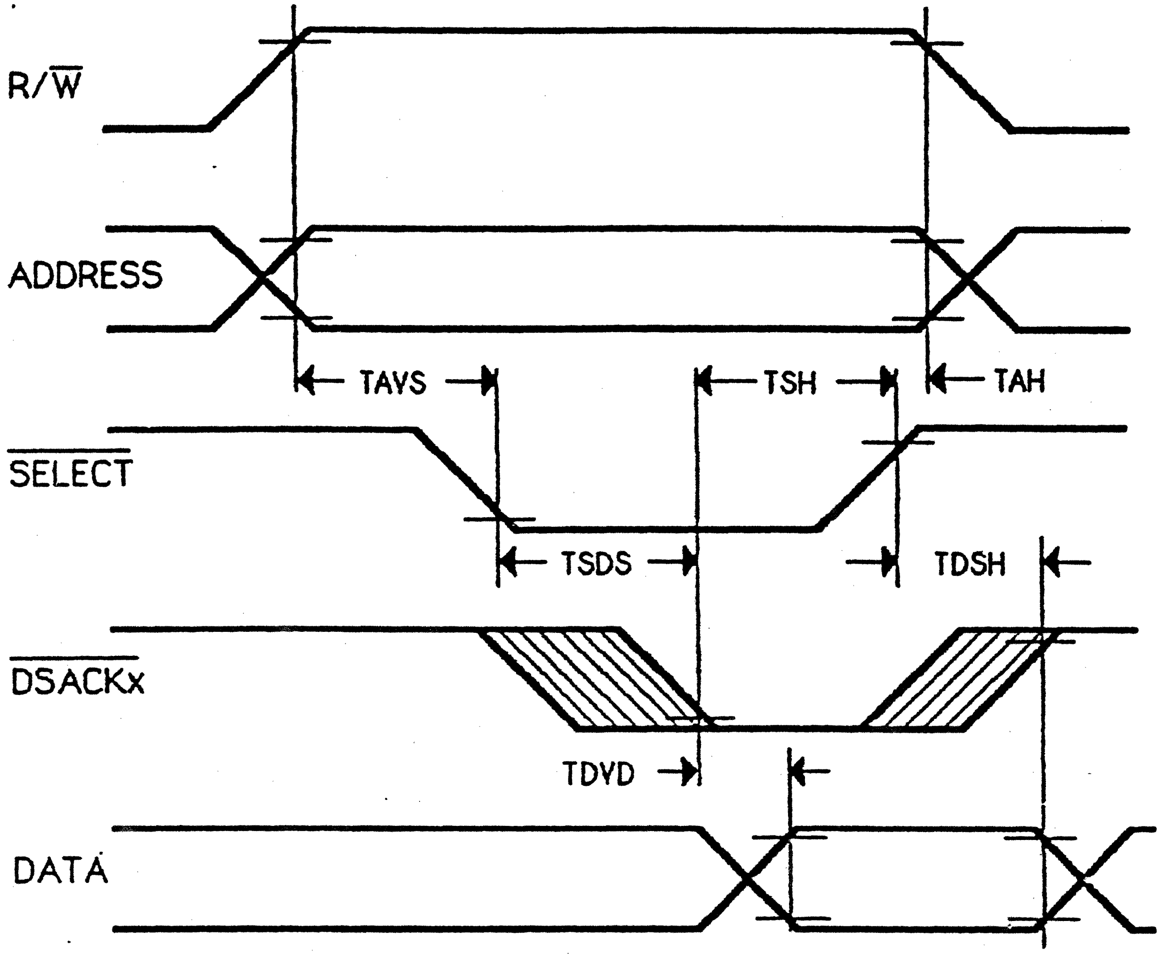
Figure 18-1 shows the timing requirements of a read cycle (processor reads data from the peripheral) at all processor clock rates.

A read cycle begins when /SEL is asserted with Read/Write (R/W) high. 'TAVS' indicates the setup time for the R/W and address lines.

Within 'TSDS' of the assertion of /SEL, the peripheral must decode the address lines, drive the data bus, and assert the proper DSACK line to indicate the size of the transfer (8 or 16 bits). The data must be valid within 'TDVD' of the assertion of DSACK. If DSACK is not asserted within 'TSDS' the cycle will be aborted with a bus error.

The DSACK line must remain asserted and the data must remain valid until /SEL has been negated. After the DSACK line is asserted, the data is latched by the processor and /SEL is negated, after 'TSH', to complete the cycle. The R/W and Address lines remain valid for 'TAH' after /SEL is negated. The DSACK line must be negated within 'TDSH' of the rising edge of /SEL.

Read Cycle Timing

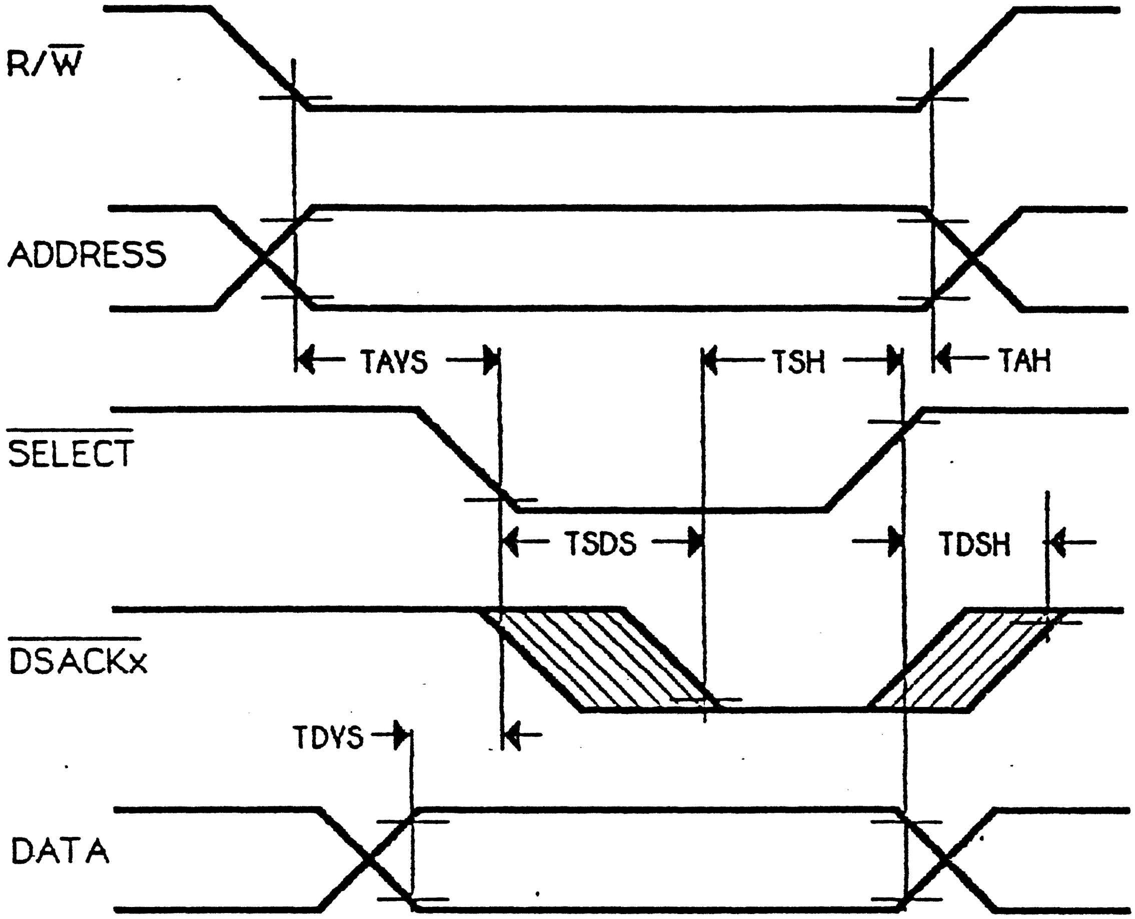


Read Cycle Timing		12.5 Mhz		16.67 Mhz		20 Mhz	
		Min.	Max.	Min.	Max.	Min.	Max.
TAVS	Address Valid before Select	100 ns		80 ns		70 ns.	
TSDS	Select to Dsack delay	0	*	0	*	0	*
TSH	Select Hold after Dsack	110 ns		85 ns		75 ns.	
TAH	Address Hold after Select	20 ns		10 ns		10 ns.	
TDSH	Dsack Hold after Select	0	90 ns	0	60 ns.	0	45 ns
TDVD	Data Valid after Dsack		45 ns.		35 ns.		28 ns.

* 61 µs. for Revision A and older boards
125 µs. for Revision B and later boards

Figure 18-1

Write Cycle Timing



Write Cycle Timing		12.5 Mhz		16.67 Mhz		20 Mhz	
		Min.	Max.	Min.	Max.	Min.	Max.
TAVS	Address Valid before Select	100 ns		80 ns		70 ns.	
TSDS	Select to Dsack delay	0	*	0	*	0	*
TSH	Select Hold after Dsack	110 ns		85 ns		75 ns.	
TAH	Address Hold after Select	20 ns		10 ns		10 ns.	
TDSH	Dsack Hold after Select	0	90 ns	0	60 ns.	0	45 ns.
TDVS	Data Valid before Select	20 ns		10 ns		5 ns	

* 61 µs. for Revision A and older boards
125 µs. for Revision B and later boards

Figure 18-2

18-4: Write Cycle Timing

Figure 18-2 shows the timing requirements of a write cycle (processor writes data to the peripheral) at all processor clock rates.

The timing for a write cycle is similar to that of a read cycle, with the exception of the data timing. A write cycle begins when /SEL is asserted with the R/W line low. Valid data is available on the bus for 'TDVS' before /SEL is asserted.

After /SEL is asserted, the peripheral must latch the data and assert the proper DSACK line to indicate the size of the transfer (8 or 16 bits). As in the case of a read cycle, DSACK must be asserted within 'TSDS' or a bus error will occur.

/SEL remains asserted and the data is valid for 'TSH' after the DSACK line is asserted. The DSACK line must be negated within 'TDSH' of the rising edge of /SEL.

18-5: Data Bus Sizing and Alignment

The processor determines the data bus width of a peripheral device by the DSACK line it uses. 8-bit peripherals must use /DSK0 to acknowledge a data transfer and 16-bit peripherals must use /DSK1.

Peripheral devices must be connected to the appropriate data lines. The most-significant bit of the peripheral's data bus is always connected to data line D31. 8-bit devices use D24-D31 and 16-bit devices use D16-D31.

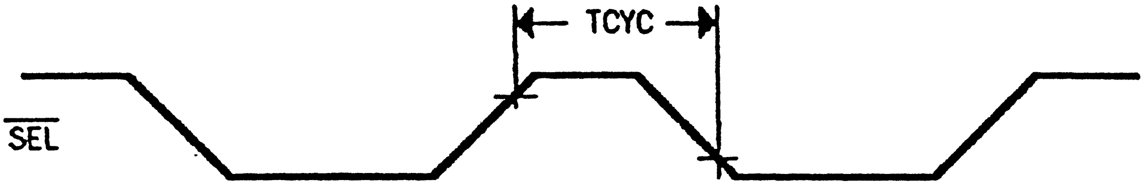
Since the processors 'size' signals (SIZ0 and SIZ1) are not present, MISALIGNED DATA TRANSFERS ARE NOT PERMITTED on the expansion port. All data transfers must occur on the appropriate address boundaries (i.e. byte to byte, word to word, long word to long word boundaries).

The lack of the size signals also prohibits data transfers smaller than the width of the peripheral device, so byte transfers are not permitted to word devices. However, transfers larger than the width of the device (e.g. long word transfers to a byte or word device) are permitted, as long as the transfer is properly aligned and the peripheral can respond fast enough to the sequential accesses generated by this type of transfer.

18-6: Peripheral Response Time

Figure 18-3 shows the minimum time (TCYC) that can elapse between sequential accesses to a peripheral device (i.e., from the time /SEL is negated to the beginning (/SEL asserted) of the next bus cycle). The peripheral device must be capable of responding to sequential accesses within the time shown, or else single instructions, that cause multiple accesses, must not be used. For example, a word transfer on a byte-wide device can produce two bus cycles, separated by 'TCYC'. If the peripheral can not respond fast enough to recognize the second bus cycle, such transfers must be avoided.

Under certain conditions, even separate instructions (e.g., two sequential 'move' instructions) can produce similar timing. In this case, instructions sequences must be chosen, or delays introduced, to prevent violation of the peripheral's timing requirements.

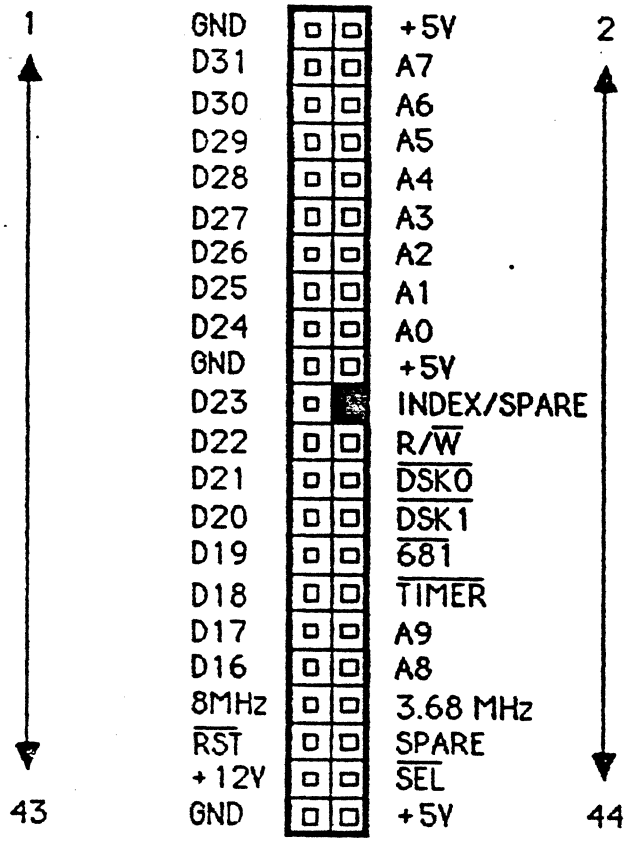


Expansion Port Sequential Access Timing		12.5 MHz	16.67 MHz	20 MHz
TCYC	CYCLE TIME (Min.)	140 ns.	110 ns.	95 ns.

Figure 18-3

18-7: Signal Descriptions

Figure 18-4 shows the signals available and the pinout of the I/O expansion connector (P1).



GND	+5V
D31	A7
D30	A6
D29	A5
D28	A4
D27	A3
D26	A2
D25	A1
D24	A0
GND	+5V
D23	INDEX/SPARE
D22	R/$\overline{W}$
D21	$\overline{DSK0}$
D20	$\overline{DSK1}$
D19	$\overline{681}$
D18	$\overline{TIMER}$
D17	A9
D16	A8
8MHz	3.68 MHz
$\overline{RST}$	SPARE
+12V	$\overline{SEL}$
GND	+5V

Expansion Connector (P1) Pinout

As viewed from the component side of the GMX Micro-20.

Figure 18-4

Address Bus (A0-A9)

These lines are connected directly to the processors address bus. Although 4K (4096) bytes of address space are allocated to the I/O expansion port, only the lower 10 address lines are normally available at connector P1. This allows a maximum of 1K (1024 bytes) to be decoded. If a 2K or 4K address space is needed, one or both of the undefined pins on the expansion connector (spare and index/spare) can be used as additional address lines. In order to provide these signals, jumpers connecting the unused pins to the appropriate points on the GMX Micro-20 board must be added as described in section 18-10.

Data Bus (D16-D31)

These lines are connected directly to the processors bi-directional data bus. 8-bit (byte wide) devices must be connected to the eight most-significant data lines (D24-D31); 16-bit (word wide) devices use all sixteen lines. The most-significant bit of the peripheral device's data bus is always connected to D31; the least-significant bit connects to D24 (8-bit devices) or D16 (16-bit devices).

Read/Write (R/W)

Read/Write is an output, connected directly to the processors R/W line, that indicates data transfer direction. R/W is high (1) during reads from the expansion port and low (0) during writes to the expansion port.

Select (/SEL)

Select is the primary timing reference for data transfers on the expansion port. It is a combination of the high-order address lines, which define the address range occupied by the expansion port, and the control signals necessary to provide proper timing of port accesses. /SEL is an active low output from the GMX Micro-20.

DSACK0 and DSACK1 (/DSK0 & /DSK1)

The asynchronous nature of the processor interface requires a handshake signal from the peripheral device to the processor to complete a data transfer (bus cycle). The DSACK lines provide the necessary handshake and are also used to inform the processor of the size (bus width) of the peripheral.The DSACK lines are active low inputs to the GMX Micro-20. 8-Bit devices must assert /DSK0; 16-bit devices must assert /DSK1.

Reset (/RST)

Reset is a bi-directional signal connected directly to the processors reset line. /RST is an active low signal, used to reset peripheras to a known state. /RST is asserted by both hardware ans software resets. Peripheral devices can also reset the processor (and everything else connected to the processors reset line) by driving /RST low. For additional information, refer to sections 1 and 3.

If driven by the expansion board, /RST must be asserted for a minimum of 520 processor clock periods (approximately 42 us. at a 12.5 Mhz clock rate), in order to guarantee recognition by the processor.

Interrupt Requests (/681 & /TIMER)

/681 and /TIMER are active low inputs to the GMX Micro-20 that generate autovectored interrupts to the processor. /681 generates a level 3 autovector interrupt, which can also be generated by the 68681 DUARTs on the GMX Micro 20. /TIMER generates a level 4 autovector interrupt. A level 4 interrupt can also be generated by the 'TOUT' output of the 68230 PI/T (refer to section 12). However, level 4 interrupts are not normally used by the standard software supplied with the board.

Clocks (8 MHz & 3.68 MHz)

These two free-running clock outputs may be used for a variety of purposes. Both clocks are totally asynchronous with respect to each other and to the processor clock. They CAN NOT be used directly to establish processor related timing.

The 8Mhz clock is an 8 megahertz square-wave, with a 50% duty-cycle. It can be used directly or divided down to produce slower clock rates if necessary.

The 3.68 MHz clock output (actually 3.6864 MHz) is a square-wave with approximately a 60/40 duty-cycle. This clock is used by 68681 DUARTs to provide standard baud rate frequencies.

Power and Ground (+5V, +12V, and GND)

The +5V and +12V pins can be used to power peripheral devices. The ground pins provide both logic ground and power supply return.

The power supply pins are capable of providing approximately 1 Amp at 12 Vdc, and 2 to 3 Amps at +5 Vdc. If the peripheral requires higher supply currents, separate supply connections should be made directly to the peripheral board. The power supply lines should be decoupled by capacitors distributed around the peripheral board. Bulk decoupling capacitors, located as close as possible to the connector, are also recommended.

Regardless of the method used to power the peripheral board, logic ground on the peripheral must be connected to the ground pins on the expansion connector. All three ground pins should be used.

Undefined Pins (SPARE and SPARE/INDEX)

These pins are currently undefined and are not connected on the

GMX Micro-20. They may be user defined for special applications (e.g., additional address lines). The pin labeled 'SPARE/INDEX' normally has an indexing key installed, and is used by GMX expansion boards to help insure proper alignment of the expansion connectors.

******** WARNING ********

If the 'SPARE/INDEX' pin is used as a signal line, eliminating the index key, EXTREME CAUTION must be exercised to prevent misalignment of the connectors. If power is applied with the connectors misaligned, the +12V supply can be fed back through the signal lines; DESTROYING the processor, coprocessor, and other devices on the GMX Micro-20. To prevent damage to the board(s), an indexing key should be used (and the corresponding pin removed from the expansion board) if at all possible.

****

18-8: Electrical Characteristics

To insure reliable operation, devices connected to the expansion port must chosen to meet the output loading and/or input drive specifications of the signal lines. If necessary, buffers must be used to limit loading and provide the necessary drive capabilities.

The following paragraphs describe drive requirements and/or capabilities of all expansion port signals.

The address and clock lines (A0-A9, 8MHz, and 3.68MHz) are TTL level outputs capable of driving the equivalent of one standard TTL load (1.6 ma. sink, 40 ua. source).

The Read/Write line (R/W) is a TTL level output capable of driving two standard TTL level loads (3.2 ma. sink, 80 ua. source).

The Select line (/SEL) is a TTL level output, capable of driving ten standard TTL level loads (16 ma. sink, 400 ua. source).

The Interrupt lines (/681 and /TIMER) are TTL level inputs that require open-collector or open-drain drivers capable of sinking at least 2 ma. Pullup resistors for these lines are provided on the GMX Micro-20; external pullups are not required.

The DSACK lines (/DSK0 & /DSK1) are TTL level inputs that may be driven by either open-collector (open-drain) or three-state drivers, capable of sinking at least 2 ma. Many standard M68000 family peripheral devices use a three-state output to drive their Dtack output (equivalent to the 68020 DSACK). When three-state drivers are used, the output is driven low to assert DSACK (DTACK), driven high to negate it, and then placed in the high-impedance state. By driving the output high first, the rise-time of the signal is controlled. Regardless of driver type chosen, the DSACK line(s) must not driven beyond the maximum time shown in the timing diagrams in sections 18-3 and 18-4. Pullup resistors for these lines are provided on the GMX Micro-20, so external pullups are not required.

The Data lines (D16-D31) are bi-directional TTL level signals.

As outputs (during write cycles) they are capable of driving one standard TTL load (1.6 ma. sink, 40 ua. source). As inputs (during read cycles) they require drivers capable of driving 1 TTL load. When not driving the busduring a read cycle, the drivers must present a high-impedance to the data lines from the processor.

The Reset line (/RST) is an active low, bi-directional TTL level signal. As an output (from the processor) it is capable of driving one standard TTl load (1.6 ma. sink, 40 ua. source). Reset is driven low during both hardware and software resets (refer sections 1 and 3). /RST can also be driven by the expansion board, to cause the entire system (including the processor) to be reset. In order to drive the /RST line, an open-collector (open-drain) driver capable of sinking 8.5 ma. (plus any load presented by the expansion board itself) must be used. A pullup resistor is provided on the GMX Micro-20, so an external pullup is not required.

18-9: Interface Circuit Example

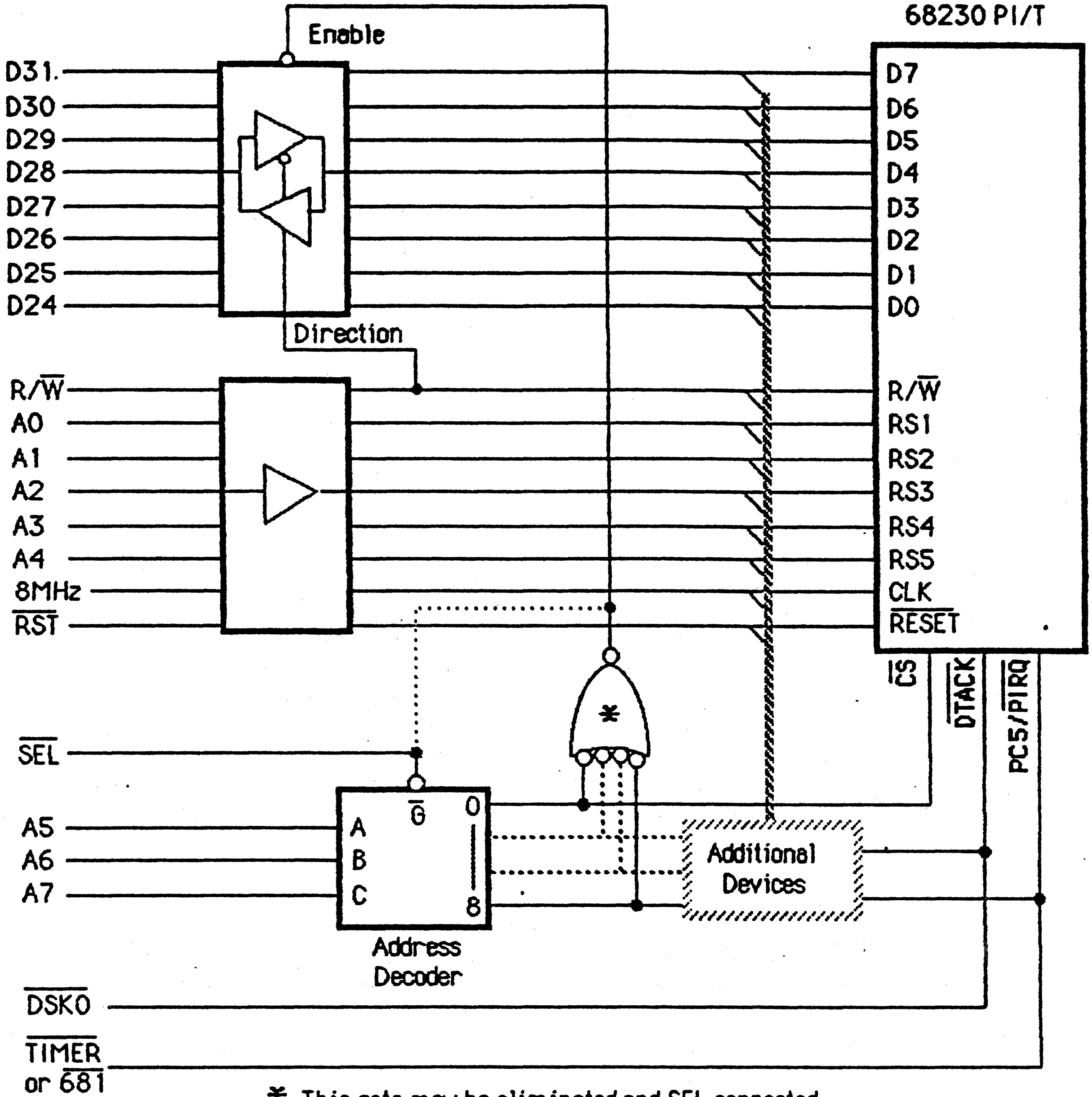
Figure 18-5 shows a typical circuit for connecting M68000 family peripheral devices to the expansion interface. Although an MC68230 PI/T is shown, the basic circuit is applicable to many other devices. In many cases, only the address decoding needs to be changed to match the address space requirements of the device(s) used. For 16-bit devices an additional data buffer, with its direction and enable inputs connected in parallel with the existing 8-bit buffer, is also required.

The sample circuit supports up to eight devices, each occupying a 32-byte block of addresses. Since only the lower eight address lines (A0-A7) are decoded, the 256-byte block of address occupied by the eight devices would repeat every 256 bytes in the expansion port address space. To provide more complete decoding and allow space for other boards or devices, additional address lines must be included in the address decoding logic.

The circuit includes buffers for data, address, and control signals. Buffers must be used, where necessary, to meet the drive and loading specifications of the expansion interface lines.

Two methods of enabling the data buffer are shown in the example. If the board is designed so that two boards can be connected simultaneously, the buffer must be enabled by 'OR'ing together the individual outputs from the address decoder. In this way, the buffer is only enabled when one of the devices associated with it is selected. Similar logic must be used on each board.

If only one board is to be used, the design can be simplified by eliminating the 'OR' logic and connecting the Select (/SEL) signal from the expansion interface directly to the data buffer enable (as shown by a dotted line). When connected in this way, the data buffer is enabled whenever the expansion address space is accessed, and only one board can be connected at a time.



* This gate may be eliminated and SEL connected directly to the data buffer enab~~le~~(as shown by the dotted line) if only one board is to be used (see text).

Figure 18-5

18-10: Adding Address Lines

In order to expand the I/O Expansion Port's address space from the standard 1K space to 2K or 4K, one or two additional address lines must be wired to connector P1. If a 2K space is needed, address line A10 should be connected to P1, pin 40 (SPARE). If a 4K space is needed, address line A11 must also be connected to P1. A11 should be connected to pin 22 (SPARE/INDEX).

********** WARNING **********

The use of the SPARE/INDEX pin (P1, pin 22) as an additional signal line necessitates the removal of the the indexing key from connector P1, eliminating protection from imcorrect connector insertion. If the indexing key is eliminated, use EXTREME caution when installing I/O expansion boards to prevent improper mating of the connectors and possible destruction of the processor and other components. Refer to the warning in section 18-7 for more information.

****

The board does not have jumpers to make the necessary connections between P1 and address lines A10 and A11. Wire jumpers must be installed on the solder side of the board. The connections should be made using light gauge wire, such as 30 guage wire-wrapping wire, a low-wattage soldering iron, and rosin-core solder appropriate for electronics work. The connections should be as short and direct as possible.

The appropriate pin(s) on connector P1 can be wired to any of several points on the address bus. The connection to address line A10 can be made at pin B12 of the 68020, pin 25 of ROM socket U-13, or at the RAM address multiplexers. Connections to A11 can be made at pin A12 of the 68020, pin 24 of ROM socket U-13, or at the RAM address multiplexers.

Note: Since different address multiplexing schemes are used on some revisions of the circuit board, no pin number references are listed above for these parts. Refer to the logic diagram in appendix G to locate A10 and A11 on the address multiplexers (U-20, U-21, or U-22).

18-11: Connector P1 Mechanical Details

Figure 18-7 shows the location of expansion connector P1 with respect to the board's four mounting holes. These holes can be used to attach expansion boards to the GMX Micro-20, using 6-32 hardware and spacers. Insulated spacers or washers must be used to prevent shorts to the components and traces on the circuit boards.

The expansion connector is a 44-pin Double Row receptacle (AMP Part # 1-86418-2, or equivalent), and accepts standard .025" square pins on .100" X .100" centers. For information on mating connectors, refer to the connector summary in the 'SPECIFICATIONS' section at the beginning of this manual.

Note: Due to the effects of cable capacitance, a cable should not be used to connect peripheral devices to the expansion connector. Connections between the expansion port and external devices must be as short as possible, and expansion boards should plug directly into the connector on the GMX Micro-20.

The length of the pins used on an expansion board, and the separation between the board and the GMX Micro-20, must be chosen to allow proper mating with the expansion connector. Figure 18-6 shows the minimum and maximum insertion depth allowed by the connector. If the pins do not project far enough into the connector, erratic contact will result. If the pins project too far into the connector, the circuit boards may be distorted, and possibly damaged, when they are fastened together. Note: The expansion connector is an open-bottom design, allowing the mating pins to rest on the top surface of the GMX Micro-20 board at the maximum insertion depth. This should not cause any problems, as long as excessive pressure is not applied to the pins. However, excessive pressure could cause the pins to puncture the circuit board and short to the board's inner layers.

When choosing connector and spacer lengths, proper air flow and cooling of the GMX Micro-20 must be considered. The expansion board must be spaced far enough above the main board to permit air to circulate between the boards. The minimum recommended spacing is one-half inch, from the top of the GMX Micro-20 to the bottom of the expansion board. Greater spacing may be required if the expansion board extends over areas on the main board where the I/O connectors are located.

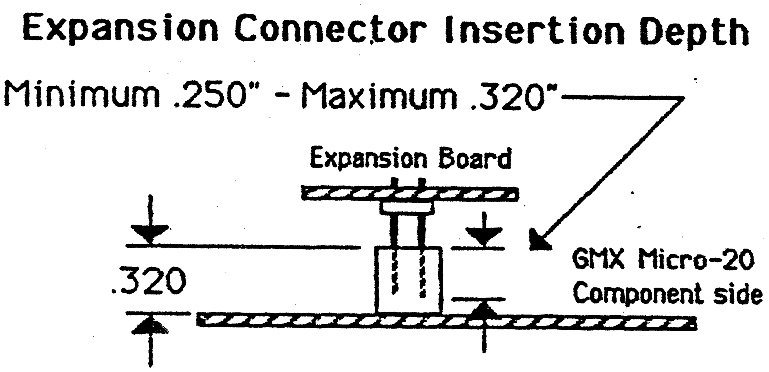

Expansion Connector Insertion Depth
Minimum .250" - Maximum .320"
Expansion Board
.320
GMX Micro-20
Component side

Figure 18-6

I/O Expansion Connector and Mounting Hole Locations

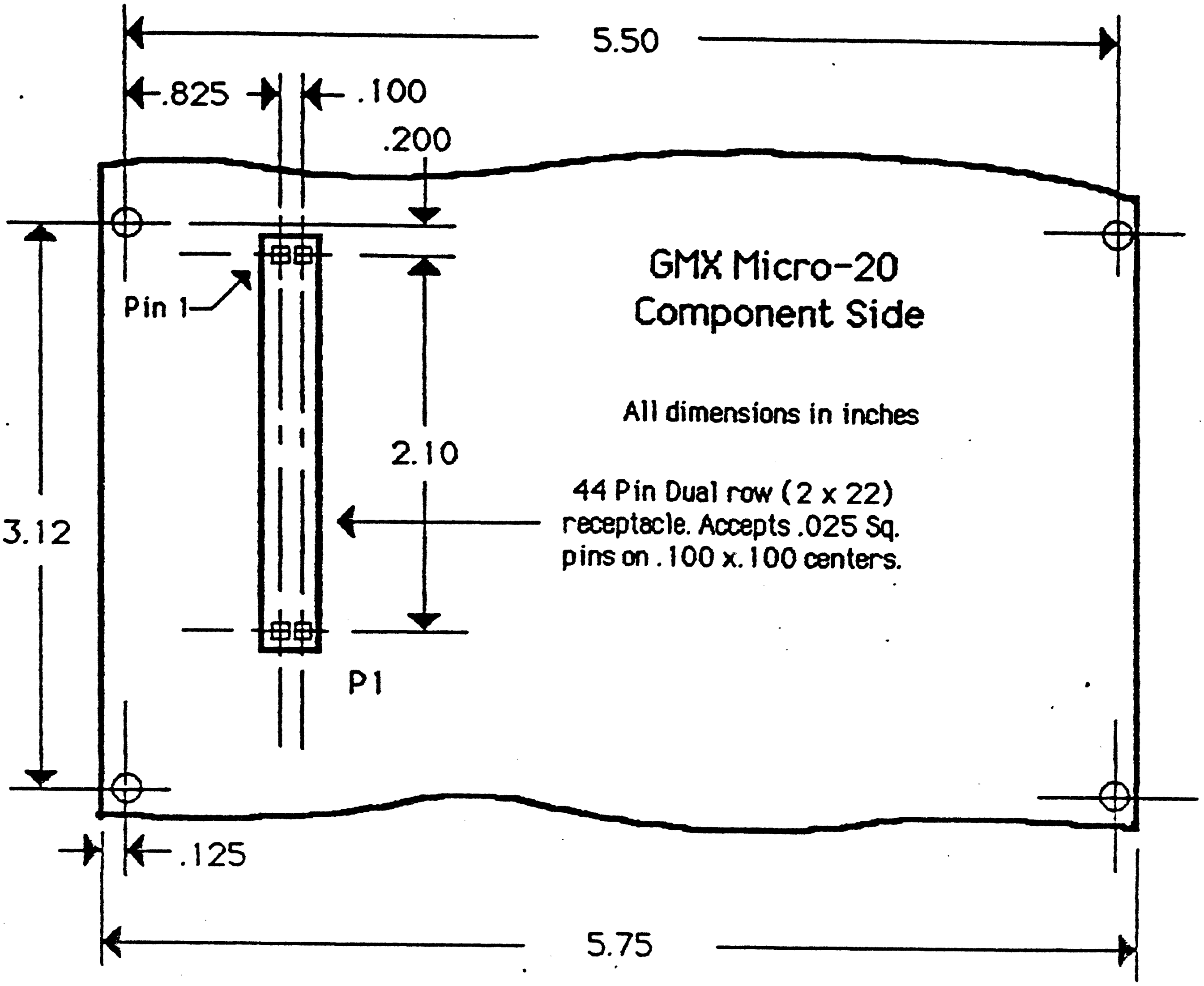


Figure 18-7

NOTES

Control Register Functions

SASI Status Register (SASR) - Read Only

Address: $00FF800E

Bit	Name		Function
b7	BUSY	0 = Not Busy 1 = Busy	Indicates the busy status of an attached SASI controller.
b6	COPI	0 = No 1 = Yes	Indicates whether or not a 68881 floating-point coprocessor is installed.
b5 b4		Undefined Undefined	These bits are not used and may read as either a 1 or a 0.
b3	SCMD	0 = False 1 = True	Set (1) when the SASI controller is waiting for a command input.
b2	SSWR	0 = False 1 = True	Set (1) when the SASI controller is waiting for data input.
b1	SSRD	0 = False 1 = True	Set (1) when the SASI controller is waiting to output data.
b0	STAT	0 = False 1 = True	Set (1) when the SASI controller is waiting to output status info.

SASI Controller Select Register (SCSR) - Write Only

Address: $00FF800D

Bit	Name	Function
b7 . . . b0	SSEL	Writing a one (1) to any one of the 8 data bits selects one of eight possible controllers connected to the interface. (e.g. writing $01, selects controller 0) Only one bit should be set at a time.

SASI Interrupt Enable Register (SIER) - Write Only

Address: $00FF800C

Bit	Name	Function
b7 . . . b0	SIER	Any write (data = don't care) to this location enables the interrupt output from the SASI interface. The interrupt is automatically cleared when it is serviced by the processor.

Control/Status Register (CTSR) - Read

Address: $00FF8004

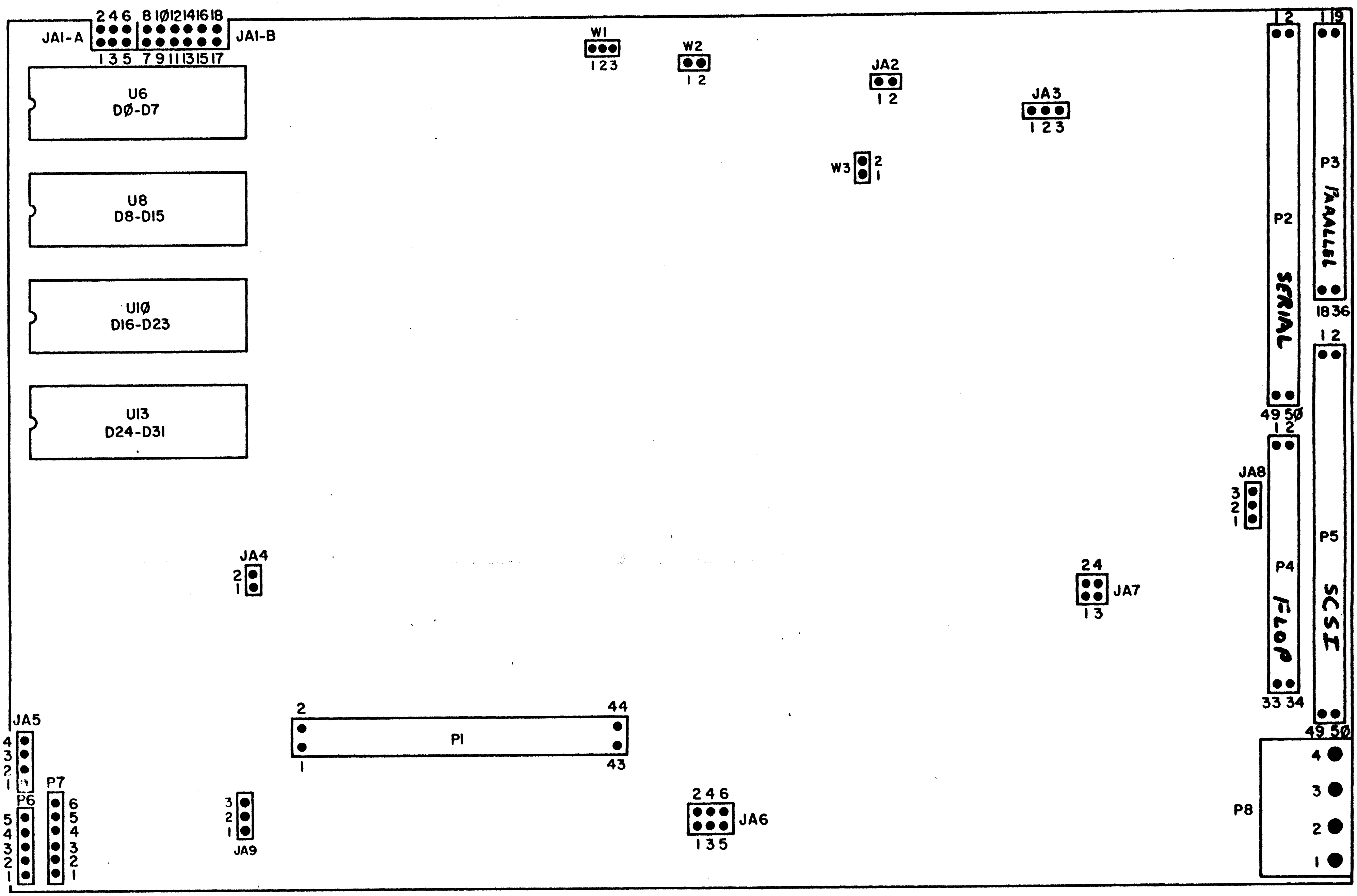
Bit	Name		Function
b7	DRQ	0 = False 1 = True	Set (1) if the floppy disk controller "DRQ" interrupt output is active (high).
b6	INT	0 = False 1 = True	Set (1) if the floppy disk controller "INTRQ" interrupt output is active (high).
b5	RDY	0 = Ready 1 = Not Ready	Indicates the state of the floppy disk "ready" line.
b4	SEN5	0 = ON 1 = OFF	Set (1) if sense switch S1-5 is OFF
b3	SEN4	0 = ON 1 = OFF	Set (1) if sense switch S1-4 is OFF
b2	SEN3	0 = ON 1 = OFF	Set (1) if sense switch S1-3 is OFF
b1	SEN2	0 = ON 1 = OFF	Set (1) if sense switch S1-2 is OFF
b0	SEN1	0 = ON 1 = OFF	Set (1) if sense switch S1-1 is OFF

Control/Status Register (CTSR) - Write

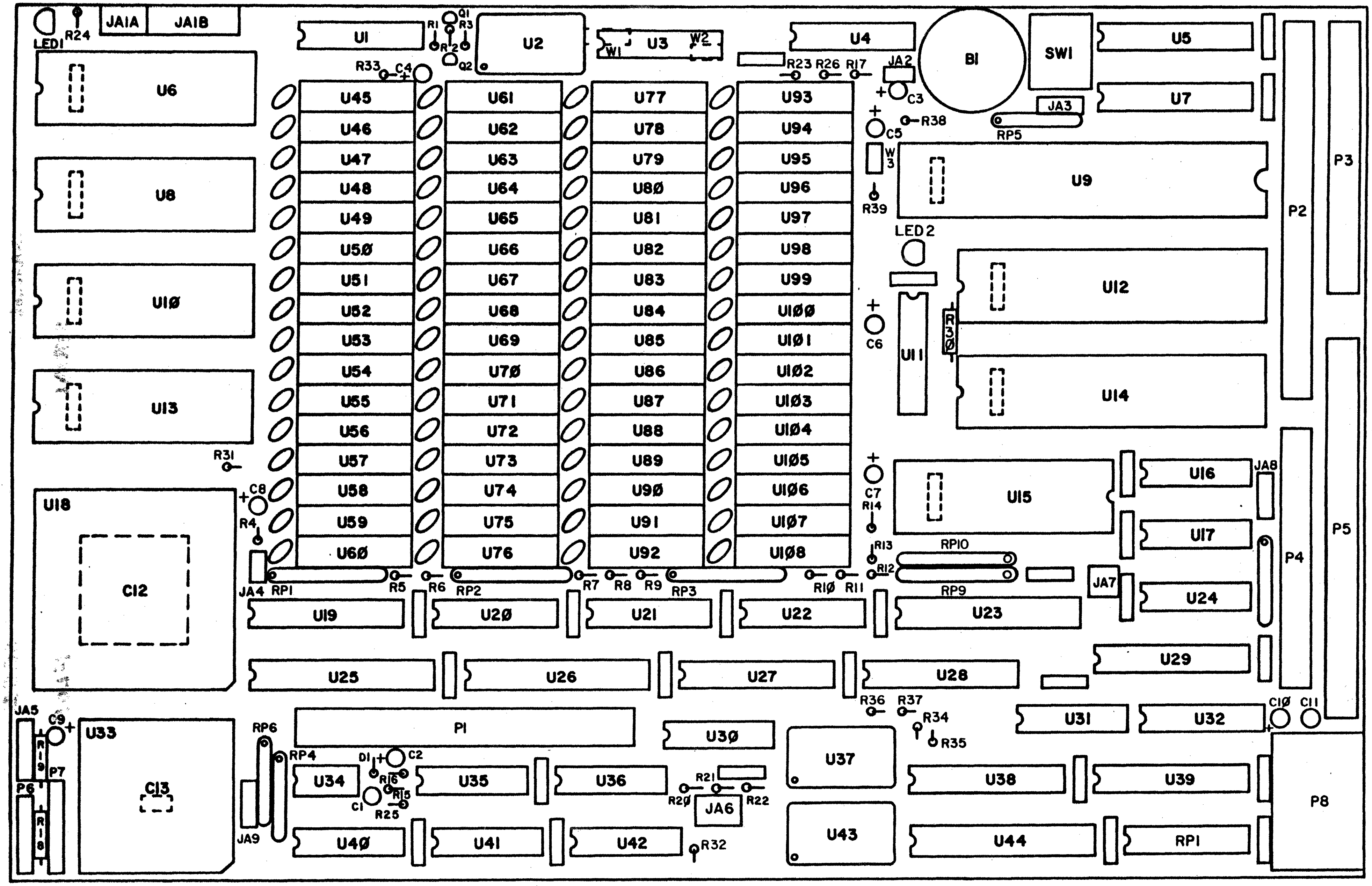
Address: $00FF8004

Bit	Name		Function
b7	IRQEN	0 = Disable 1 = Enable	When set (1), enables DRQ interrupts from the floppy disk controller to the processor.
b6		Unused	
b5			
b4	MOTON	0 = Off 1 = On	When set (1), starts the floppy disk motors.
b3	SIDE	0 = Side 0 1 = Side 1	Selects side 0 or 1 on double-sided floppy disk drives.
b2	DENS	0 = Double 1 = Single	Selects single or double density operation of the floppy disk controller.
b1	DRV1	0 = Not Selected 1 = Selected	Bits 0 & 1 select one of two floppy disk drives. Only one bit should be set (1) at a time.
b0	DRV0	0 = Not Selected 1 = Selected	

NOTES



GMX MICRO-2Ø JUMPER AND CONNECTOR LOCATIONS



▭ = CX

⬭ = CY

COMPONENT LAYOUT

GMX™ Micro-20
68020 Single-board Computer

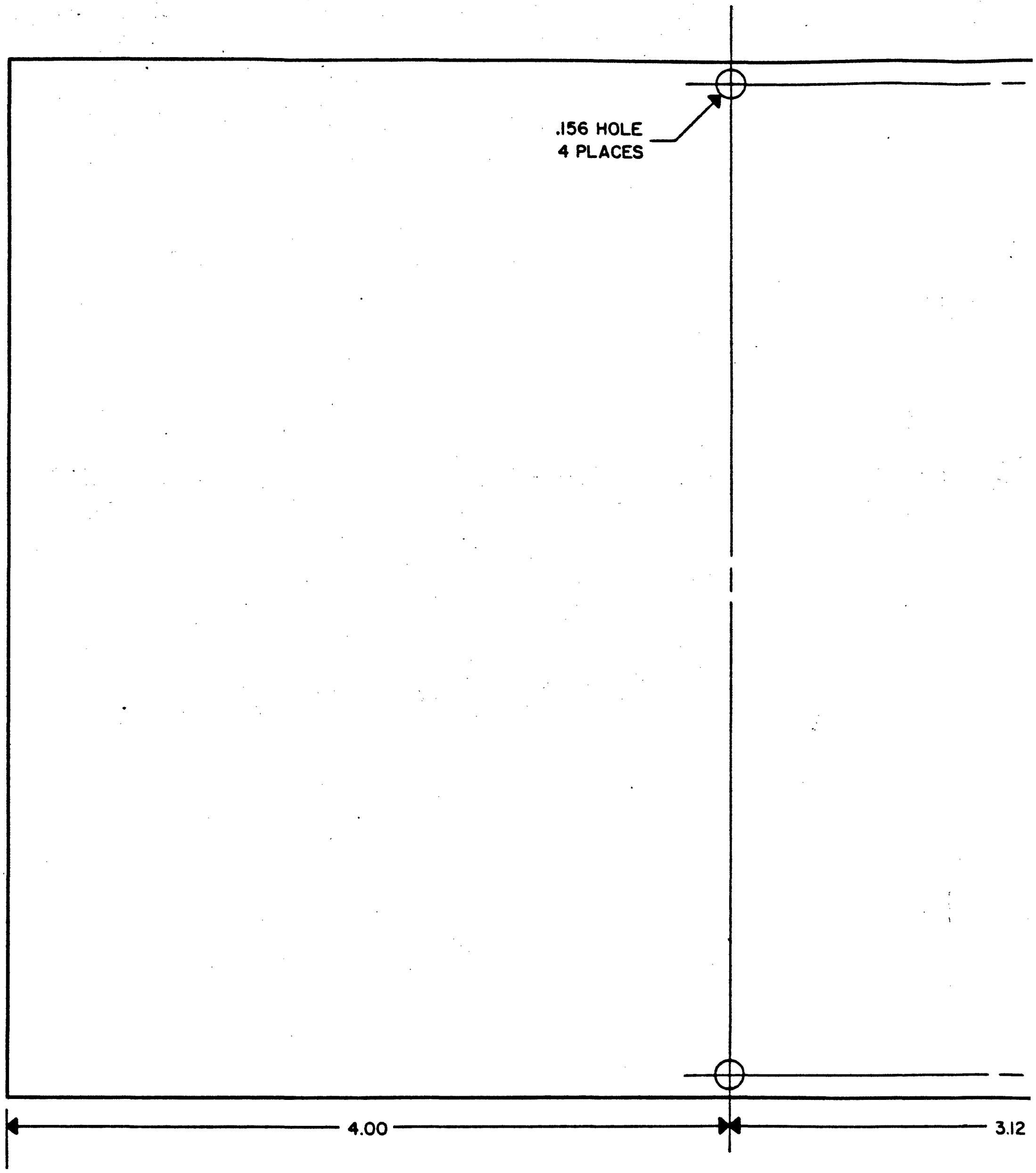
Memory Map

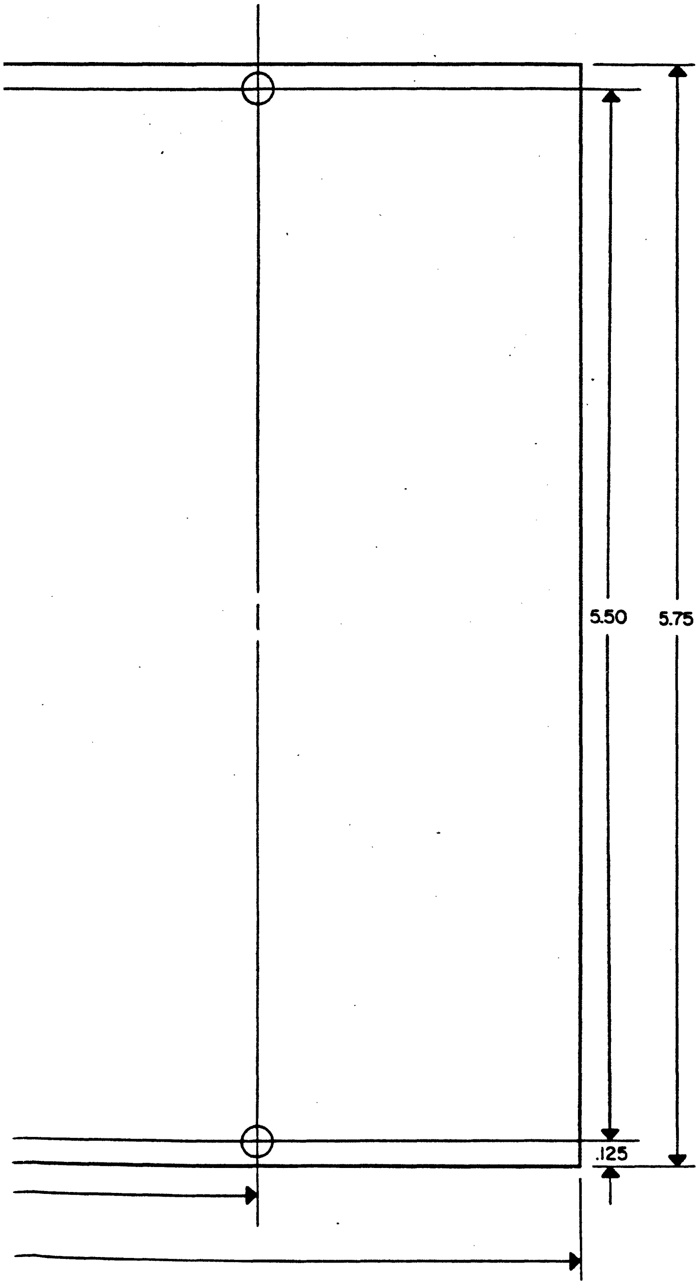
Address	Mode	Function
$FFFFFFFF / $01FFFFFF	---	$00000000 - $00FFFFFF repeats 255 times
$00FFFFFF / $00FFA000	---	$00FF8000 - $00FF9FFF repeats 3 times
$00FF9FFF / $00FF9000	R/W	I/O Expansion Connector
$00FF8FFF / $00FF80E0	---	Unused *
$00FF80DF / $00FF80C0	R/W	68230 PI/T
$00FF80BF / $00FF80B0	---	DUART #2 repeats
$00FF80AF / $00FF80A0	R/W	68681 DUART #2
$00FF809F / $00FF8090	---	DUART #1 repeats
$00FF808F / $00FF8080	R/W	68681 DUART #1
$00FF807F / $00FF8010	---	$00FF8000 - $00FF800F repeats 7 times

Memory Map (con't)

Address	Mode	Function
$00FF800F	---	Unused *
$00FF800E	Read Only	SASI Status Register (SASR)
$00FF800D	Write Only	SASI Controller Select (SCSR)
$00FF800C	Write Only	SASI Interrupt Enable (SIER)
$00FF800B / $00FF8008	R/W	SASI Data Register (SADR)
$00FF8007 / $00FF8005	---	Unused *
$00FF8004	R/W	Control/Status Register (CTSR)
$00FF8003 / $00FF8000	R/W	Floppy Disk Controller
$00FF7FFF / $00840000	---	Unused *
$0083FFFF / $00800000	Read Only	256K Byte EPROM space
$007FFFFF / $00200000	---	Unused *
$001FFFFF / $00000000	R/W	2 Megabyte RAM space

* Any access to unused areas, and accesses in the wrong mode (e.g. reading a write-only location), will cause a bus error exception.


.156 HOLE
4 PLACES
4.00
3.12



GMX INC.

1337 W. 37th PL. CHICAGO, IL. 60609

8-25-86	DRAWN BY: CLARK PETROUSEK

GMX MICRO-20
MOUNTING DIMENSIONS

M24-0085

GMX Micro-20 68020 Single-board Computer

Interrupt Source Summary

The GMX Micro-20 uses the autovectored interrupt capabilities of the 68020 to provide seven levels of prioritized hardware interrupts. Each of the seven possible interrupt levels is dedicated to a particular function or functions. The following table lists each interrupt level, the source or sources for that interrupt, and the the section(s) of this manual where more information can be found.

Autovector Number	Priority	Source(s)	Section
7	Highest	Single-step/Abort Switch 1772 FDC (DRQ)	4 16
6	-	Periodic Interrupt Generator	15
5	-	1772 FDC (INTRQ)	16
4	-	68230 PI/T (PC3/TOUT) I/O Expansion Port (/TIMER)	12 18
3	-	68681 DUARTs (INTRN) I/O Expansion Port (/681)	11 18
2	-	68230 PI/T (PC5/PIRQ)	12
1	Lowest	SASI/SCSI Peripheral Interface	17

NOTES

GMX Micro-20 Compatible SASI/SCSI Controllers

The following SASI/SCSI controllers were known to be compatible with the GMX Micro-20 SASI interface at the time this manual was published. As additional controllers are verified as compatible, they will be added to this list. For up to date information on compatible controllers please contact the factory.

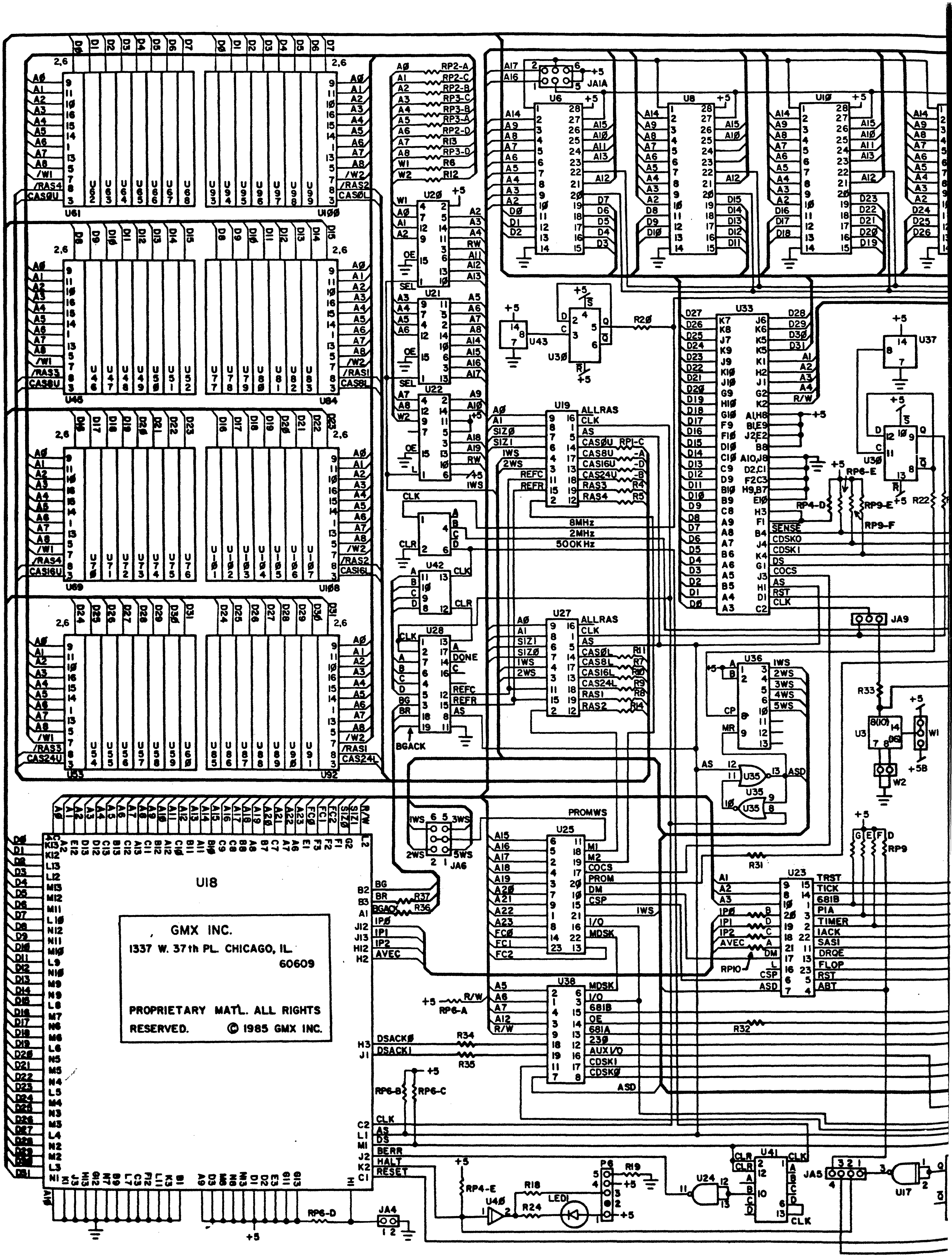
For more information on controller compatibility, refer to section 17 of this manual.

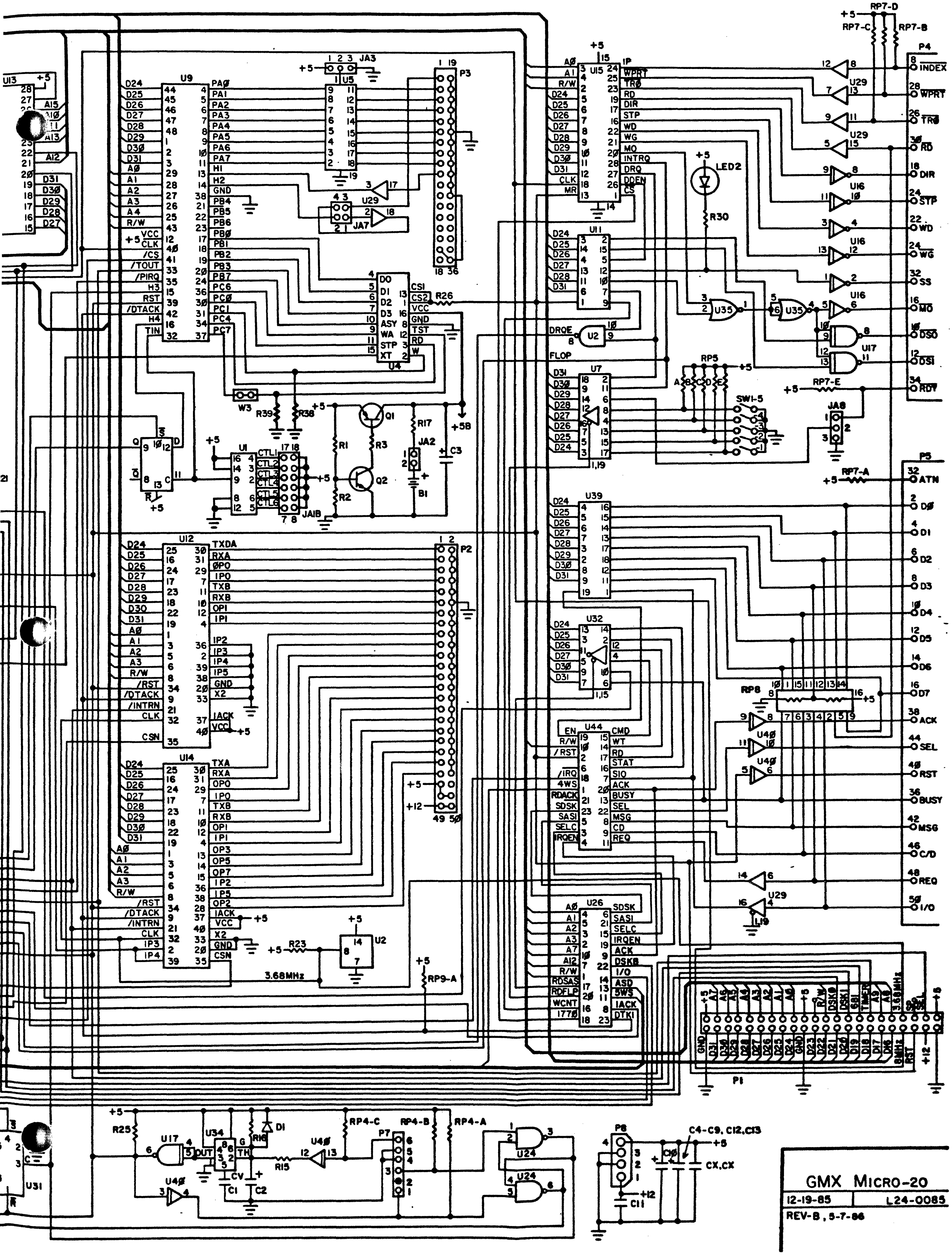
Controller Model	Type	Manufacturer
20C-1		OMTI
	Hard Disk	557 Salmar Avenue
5110 *		Campbell, CA 95008
1410		XEBEC Systems Inc.
	Hard Disk	432 Lakeside Drive
1410A		Sunnyvale, CA 94086
		EMULEX
MT02	Streaming Tape	3545 Harbor Blvd.
		Costa Mesa, CA 92626

* The OMTI 5110 is functionally equivalent to, and can be used with software designed for, the XEBEC 1410 controller. Other members of the OMTI 5000 family are not compatible with the GMX Micro-20.

Note: This list does not imply an endorsement of these products or their manufacturers; nor does it imply that these are the only controllers compatible with the GMX Micro-20. The versions of these controllers available at the time of this writing were tested by GMX and found to be compatible with the GMX Micro-20; however, GMX Inc. is not responsible for changes by their maunfacturers which make them incompatible.

NOTES

GMX INC.
1337 W. 37th PL. CHICAGO, IL.
60609
PROPRIETARY MAT'L. ALL RIGHTS
RESERVED. © 1985 GMX INC.
U18
U61
U100
U46
U84
U69
U108
U53
U92
U6
U8
U10
U19
U20
U21
U22
U23
U24
U25
U27
U28
U30
U33
U35
U36
U37
U38
U40
U41
U42
U43
U17
U3
JA1A
JA4
JA5
JA6
JA9
W1
W2
P6
LED1
R18
R19
R20
R22
R24
R31
R32
R33
R34
R35
R36
R37
RP2-A
RP2-B
RP2-C
RP2-D
RP3-A
RP3-B
RP3-C
RP3-D
RP4-D
RP4-E
RP6-A
RP6-B
RP6-C
RP6-D
RP6-E
RP9
RP9-E
RP9-F
RP10
8MHz
2MHz
500KHz
ALLRAS
PROMWS
BGACK
DSACK0
DSACK1
CLK
AS
DS
BERR
HALT
RESET
AVEC
IP0
IP1
IP2
BG
BR
TRST
TICK
681B
PIA
TIMER
IACK
SASI
DRQE
FLOP
RST
ABT
SENSE
CDSK0
CDSK1
COCS
ASD
+5
+5B



GMX MICRO-20

12-19-85 L24-0085

REV-B, 5-7-86