MPZ80 CPU
Technical Manual
Revision 1

April 1982

MORROW DESIGNS

SCOPE: This manual is intended for users with at least a moderate technical knowledge of microcomputers and Central Processing Units. It is a technical reference for operating, interfacing, troubleshooting and maintaining Morrow Designs' MPZ8Ø CPU board.

* * * * * * * * * * * * * * * * *

References to CP/M refer to the operating system designed by Digital Research. CP/M is a registered trademark of Digital Research.

Micronix is a UNIX operating system designed by Morrow Designs for their microcomputers.

* * * * * * * * * * * * * * * * *

A source listing for the MPZ8Ø CPU software is available for a nominal cost from Morrow Designs.

MPZ8Ø CPU Technical Manual

Revision 1

Table of Contents

Table of Contents, Cont.

Appendices:

List of Figures

List of Tables

MPZ8Ø CPU

Technical Reference Manual

Revision 1

1. INTRODUCTION

Morrow Designs' Decision CPU circuit board is an S-1ØØ bus single board computer conforming to the proposed IEEE-696 standards for interface. The CPU features:

* 4 MHz Z8Ø microprocessor chip.
* Optional 9512 floating point processor.
* 24 bits of address space allowing access to over 16M bytes of physical memory.
* On-board memory consisting of 2K bytes of EPROM and 1K byte of RAM memory.
* Sophisticated hardware memory management circuitry which is dynamically alterable by the operating system.
* Sophisticated hardware trap mechanisms that allows the operating system control over user access and operation.
* Stop trap option which may be activated to aid in debugging software.

The CPU board only requires power from the +8 volt and +16 volt supplied sources for operation. The Decision CPU is made up of the following system blocks:

1. **Bus Control Logic.** All logic necessary to make the Z8Ø processor conform to the S-1ØØ standards for interface. This circuitry provides the addresses, data and control signals required by external memory and I/O devices.

2. **Memory Mapping Logic.** All logic required to provide the 24 bits of address space (the Z8Ø is hardware limited to 64K bytes at any one time). The circuitry allocates memory by tasks (64K maximum per task) and 4K segments. Mapping is implemented by use of very high speed memory. This memory provides the addresses as well as the protection attributes of these addresses.

3. **Trap Logic.** The Decision CPU may be set to trap on an occurrence of any of the following conditions*:

   - **Halt:** User has attempted to execute a halt instruction.

   - **Interrupt:** Interrupt has occurred during users' program.

   - **Reset:** The RESET button has been pushed.

   - **Illegal Memory:** The user has tried to read or write memory which has yet to be allocated to him.

   - **I/O:** User has attempted to execute an I/O operation.

   - **Stop:** Front panel stop has been executed.

   - **Auxiliary:** User defined trap. On systems with 9512 option this can be connected to the 9512 status line. The board may also be strapped with this line tied to the S-100 NMI, PWRFAIL or ERROR lines.

   - **R10:** User has attempted to access a memory segment with the R10 protection bit set.

4. **Local Logic.** Provides all the on-board clocks and strobes for devices on the CPU card itself.

A system block diagram is included at the end of this manual along with the component layout and schematics; the user is referred to these illustrations for a further understanding of the features within the MPZ80 CPU.

1.1. Bus Control Logic

The Decision CPU is configured as an S-100 bus master, and as such, supplies all of the signals to control bus slaves. The states of the Z80 chip are decoded using a bipolar PROM (chip 15A). The Z80 signals MEMRQ, IORQ, RDSTB and M1 are decoded by the PROM and then sent to the bus control logic for conditioning. All the outputs of the PROM except START are sent to the status latch (chip 14A) along with the Z80 signal HALT ACK where they are latched by STB ENBL. In addition, the S-100 status line, SXTRQ, is held high by the CPU since it is an 8-bit device. Besides the Z80 signals, the PROM also decodes the signal NULL BUS, which, when active, prevents the status signals from reaching the S-100 bus and creates a NULL BUS cycle. NULL BUS is active only when a hardware trap has occurred (see Section 1.3.).

*Note that upon trapping on any one of the above conditions, it is up to the operating system to decide what to do. In some instances the program may be allowed to continue. In others, the program may be interrupted from executions. The traps can be set in any configuration of the above, or in the case where the user is running only one task (e.g. CP/M), disabled completely.

The PROM output START indicates the Z8Ø is beginning a bus cycle. This signal is qualified with MCLK (4 MHz CPU clock) by chip 1ØA to form the signal SYNC ENBL. This signal performs three functions: 1) It is latched, buffered and sent out to the bus as PSYNC. 2) The SYNC ENBL signal provides the signal STB ENBL (chip 3C) which latches the outputs of the decoding PROM into chip 14A for driving the S-1ØØ bus status lines. 3) After qualification with MCLK and STB ENBL (chip 1C), provides the S-1ØØ signal PSTVAL (indicating that the CPU status is now present on the S-1ØØ status lines).

The CPU board provides two clocks to the S-1ØØ bus: 1) o 2, which is the 4 MHz master clock and 2) CLOCK which is a divide-by-two version of o 2 (done by chip 1C). These signals are provided for bus synchronization.

The on-board signal, M1, (indicating the Z8Ø is performing a fetch operation) is extended by one MCLK (chip 1C) to create the signal M1A, which when qualified by STADR (chip 4A), drives the S-1ØØ bus PRDY line. This generates one wait state for every instruction fetch cycle (due to the Z8Ø short M1 cycle). The XRDY and PRDY S-1ØØ lines are ORd together (chip 1D) and are sensed by the Z8Ø WAIT pin (chip 17A, pin-24).

The signals RDSTB, WRSTB, INTA, STB ENBL and NULL BUS are decoded by chip 2A, 5A and 3A to provide the signals DBIN and WRITE. DBIN is sent out to the S-1ØØ bus as PDBIN and WRITE is sent out to the bus as PWR by chip 13A. The Z8Ø signal, BUS ACK, which indicates a DMA device has been granted access to the bus) is sent out to the bus as PHLDA also by chip 13A.

The CPU data lines are buffered and sent to DOØ - DO7 (the S-1ØØ data output lines) by chip 12D. DI lines Ø - 7 (S-1ØØ bus lines) are qualified by the on-board signal DBIN and TRAP HALT by chip 14D and reach the Z8Ø CPU data lines during a bus read cycle. The TRAP HALT signal prohibits the Z8Ø halt opcode from reaching the CPU if this trap has been set.

The CPU address lines, AØ - A11, are latched and buffered by 1ØD and 1ØC and go onto the AØ - A11 S-1ØØ address lines. The CPU address lines, A12 - A15, are conditioned by the memory mapping logic and sent onto the bus as A12 - A19 address lines. The on-board task register drives the upper four extended address lines, A2Ø - A23. Chip 1ØD latches and buffers A12 - A15 and chip 7C latches and buffers A16 - A23.

During an I/O operation, the port address must be presented on the lower address lines of the S-1ØØ bus as well. The S-1ØØ bus standard specifies the port address be presented only on the lower address lines, AØ - A7. However, due to the lateness of the standard, and the fact that the 8Ø8Ø CPU put the port address on both the upper (A8 - A15) and lower S-1ØØ address lines, the Decision CPU offers the port address on both the upper (A8 - A15) and lower (AØ - A7) lines of the bus. When the on-board signal ZIO MODE is set high, the port addresses are sent out on

lines AØ - A15 by chip 11C. If the signal is low, the Z8Ø address lines, A8 - A15, drive the S-1ØØ address lines buffered by chip 11D. (Refer to the Z8Ø Technical Manual for the contents of the upper address lines during a Z8Ø I/O operation.)

Chips 3C and 2A decode the state of the Z8Ø (Memory or I/O) to determine whether to turn on the memory address or port address drivers for the upper address lines, A8 - A15.

The S-1ØØ signal PHOLD (indicating that a temporary bus master has requested access to the bus for a DMA operation) is terminated and then buffered by chip 16B and sent to pin-25 of the Z8Ø (BUSRQ).

The S-1ØØ line PRESET is terminated through an RC network to provide a power up PRESET strobe lasting approximately 3Ø msec after the first application of power. This signal is also tied to the front panel RESET switch. Either event or the occurrence of a low going-strobe on the S-1ØØ POC line resets the Decision CPU board.

The S-1ØØ interrupt lines, PINT and NMI, are decoded and qualified by chips 4C and 6A and sent to the Z8Ø INT line (pin-16) and the Z8Ø NMI line (pin-17). These lines reach the CPU only if the TRAP MASK has been set to allow interrupts to occur regardless of whether a Z8Ø EI opcode has been executed. To allow the interrupt lines to reach the CPU during task 1, task 15, the TINT mask bit, must be set low.

To allow an interrupt to be recognized in task Ø (supervisor), the SINT mask bit must be low. The NMI line is latched by chip 4C and may be an edge-triggered event which can disappear before being acknowledged by the CPU. However, the PINT line should always remain asserted by the interrupting device until the CPU issues a SINTA signal to the bus signifying it has recognized the interrupt. The interrupt, as in any S-1ØØ system, is lost unless this condition is met. The interrupt lines from the S-1ØØ bus are not sensed during the first 16 instructions following the occurrence of a trap or the eight instructions required when switching to a new task.

The S-1ØØ bus vectored interrupt lines, VIØ through VI7, are brought onto the Decision CPU board to termination pads where they may be connected to the output of the 9512 floating point processor chip. This allows the 9512 to generate a vectored interrupt upon completion of a command or an error. These pads are in area 2E to 5E of the PC board (directly above the S-1ØØ edge connector).

The Decision CPU allows temporary bus masters to drive the bus during a DMA or similar operation. When a temporary bus master wishes to access the bus, it must generate the signals required to turn the Decision CPU bus drivers off. The signals DODSBL (turn off the data output drivers), ADDRDSBL (turn off the address drivers), CCDSBL (disable the control signal drivers) and

STDSBL (disable the status drivers) from the S-1ØØ bus perform this task.

The proposed IEEE-696 standard for the S-1ØØ bus specifies the protocol for switching bus control and the Decision CPU supports this.

1.2. Decision Memory Management Hardware

In this section, all memory size references are listed as decimal unless otherwise stated.

The Decision CPU can access a full 16 megabytes of address space. This memory is divided into sections referred to as tasks and segments.

Segments are the smallest increments a memory may be divided into and in this case are 4K bytes. Tasks may consist of a minimum of 1 segment to a maximum of 16 segments. Therefore, the maximum addressable memory by any one task is 64K bytes (this limitation is imposed by the address space of the Z8Ø processor).

The total memory space is 16M bytes, but 16 tasks of 64K each only maps 1M byte of this space. The other four bits of the address space (A2Ø - A23) are selected by the operating system as a bank select scheme. Therefore, any of the 16 tasks may be situated anywhere in the 16M byte range of the CPU card.

The Z8Ø can only execute the 16 tasks available one at a time. The tasks actually time-share the CPU chip. It is the responsibility of the operating system to allocate the CPU's time adequately. The operating system (referred to as the supervisor) runs in a special task, task Ø, which allows it access to the special features available on the CPU. These special features include access to the Trap Registers, map RAMs, floating point processor and the on-board EPROM and RAM memory. These devices are permanently memory-mapped into task Ø's memory space starting at Ø (4K bytes total) and cannot be accessed by any of the other tasks.

The program running in task Ø is referred to as the "supervisor" and all other tasks are referred to as "users". In its role as the system supervisor, task Ø is responsible for allocating or de-allocating the memory space of all 16 tasks within the system.

In addition to this, the supervisor assigns access privileges to the memory it has allocated to a particular task and determines what types of operations a user may request the Z8Ø chip to perform are allowed.

The supervisor allocates memory to a task by writing into the mapping RAMs residing at 600 (hex) to 7FF (hex) in task 0. The size of each memory segment is 4K bytes and each segment has its own set of protection attributes. A segment may be configured as:

Table 1-1: Memory Segment Configuration

No Access	-	User is not allowed any access to the memory segment.
Execute-Only	-	User is allowed to only execute the code contained within the segment.
Read-Only	-	The user is allowed to both execute and read the memory segment.
Full Access	-	The user is allowed unlimited access to the memory segment.

If any of the above conditions are violated by a task (e.g. a user has attempted to write into a read-only segment), the operation is aborted and task execution is stopped. At this point, the supervisor begins executing. This is referred to as a "trap" and will be described in more detail later.

The allocation of the memory segments is done dynamically, i.e., the supervisor can allocate memory to a task at will. In the instance where a task requires more memory (perhaps it has outgrown its allocated stack area) the operating system may or may not allow the task to have more memory.

Another feature of the Decision CPU memory management hardware is that any task may share up to 16 memory segments with any other tasks. If the operating system has data to swap with a task, it need only write a duplicate map into the mapping RAMs' image for both tasks for that segment. This also allows one program to be shared by up to 16 different tasks but reside in only one memory area. The supervisor alone has access to the mapping RAMs, which in turn allow it to access any of the memory space of any of the other tasks.

There is an additional protection bit in the Decision CPU which causes a trap to occur and allows the operating system to monitor user memory references. In the instance where a task has outgrown its memory allocation, the operating system can examine this bit and determine whether to allow the task's space to grow. This is a hardware trap protection and it is written into the protection attribute RAM along with the other protection attributes.

This allows the supervisor to allocate a full 64K bytes to a task of which perhaps 16K bytes was designated as No Access. If the task outgrows this area, and the extra protection bit is high, a trap occurs, allowing the supervisor to update the task's memory

map to access more. In this mode, the write or read occurred before the trap. If R1Ø was not set, the read or write would have aborted and the task would likely be refused access to more memory. The supervisor would then send an error message to that user. This bit is referred to as R1Ø throughout the documentation.

The versatility of the Decision CPU lies in its ability to allocate memory dynamically over the 16 Mbyte range and to trap on the occurrence of an illegal event occurring within a particular task, making it more sophisticated than the typical S-1ØØ CPU board. But this does not prohibit the use of the CPU board as a standard S-1ØØ bus master with a 64K memory space. The on-board EPROM is responsible for configuring the CPU for the user's application.

If a user wished the board to run only one task, (e.g. the CP/M operating system in one task alone), the EPROM would contain code which would essentially disable all the trapping hardware on the CPU and allocate a full 64K bytes of unprotected memory to task 1. (Remember that task Ø has its lower 4K bytes taken up in Decision CPU features and therefore would not be able to run CP/M). The switches on the CPU board determine the power-on-jump location for the operating system or bootstrap loader. The CPU may jump to any location on a 2K boundary. In this mode, the Decision CPU can emulate any other standard S-1ØØ CPU card without these features. By simply replacing the EPROM, the board may again be configured as a multi-tasking, multi-user board with large system trapping and protection features. (For details on booting up the system, see Section 2.)

1.3. Decision CPU Trap Hardware

The power of the Decision CPU lies in its ability to trap on the occurrence of an undesired event within a program. This feature allows the operating system to control the flow of execution within all the tasks in the system and prevent undesired conditions from arising.

There are essentially seven levels of trapping available with the Decision CPU. Their configuration allows the operating system to restrict the execution of certain operations within a users program and may or may not be used.

Table 1-2: Low-Level Traps

Trap Reset	-	Indicates the RESET button has been pushed or the CPU has just powered up.
Trap Stop	-	The front panel stop switch has been activated.
Trap Aux	-	User defined trap which may be connected to the floating point processor.

The other four traps are considered high level traps because they abort the occurrence of the attempted operation:

Table 1-3: High-Level Traps

Trap Halt	-	A task has attempted to execute a Z8Ø halt operation and halts are not allowed in that task.
Trap Int	-	An interrupt was generated during the execution of a task which was not allowed to honor interrupt requests.
Trap Void	-	A task attempted to execute an I/O instruction or an illegal memory access has occurred.
R1Ø	-	The user has attempted to access a memory segment which had protection attributes with the R1Ø bit set.

Trap Void is actually a compilation of several types of trap conditions. By reading the Trap Status Port (4Ø3H) in task Ø, segment Ø, the operating system can determine precisely what type of trap occurred. These traps may be further broken down into the following conditions:

Table 1-4: Trap Conditions

a) In Trap	-	A task has attempted to execute an input instruction when I/O was not permitted.
b) Out Trap	-	A task has attempted to execute an output instruction when I/O was not permitted.
c) Read Trap	-	A task has attempted to perform a memory read operation on memory which had either execute-only or No Access protection attributes.
d) Write Trap	-	A task has attempted to write into a memory location which had Read-Only, Execute-Only or No Access protection attributes.
e) Exec Trap	-	A task has attempted to execute code in an unallocated memory segment.

When any of the above trap conditions occur, the S-1ØØ bus control and status signals (sINP, sOUT, sMEMR, sWO, sM1, pWR, pDBIN) are inhibited and do not reach the bus. This effectively aborts the execution of the instruction. When this trap occurs, task Ø begins execution and decides whether to allow the trapped event

to occur. It is the responsibility of the firmware on the CPU to save the trapped task's registers to allow proper return to that task.

Whenever a trap of any kind occurs, execution of the current task is suspended and code in the EPROM on the CPU (location ØBFØ hex) begins execution. This EPROM contains all the code necessary to save all the primary and alternate Z8Ø registers in the on-board CPU RAM. Each task has an allocated area in this memory for its registers and its Trap Mask. The Trap Mask is a one byte description of the types of operation allowed within a task and is written into the Mask Register whenever the CPU switches to that task. The Mask Register is at location 4Ø3 hex in task Ø, segment Ø memory space.

The Trap Halt condition indicates a halt has been attempted but the task was not allowed to perform it. In this case, the trap logic must prevent the halt instruction from reaching the Z8Ø chip. To accomplish this, the data input driver to the CPU data lines is turned off whenever a halt "opcode" (operation code - 76h) is detected during an M1 cycle. This condition also enables the EPROM on the CPU board to gate a "nop" (no operation instruction - ØØh) into the Z8Ø chip. Therefore, location ØBFOh in the EPROM must also contain a nop since this is the location executed whenever a trap occurs.

To reiterate, the trap conditions for a particular task are set just before the task begins execution by writing a value into the Mask Register on the CPU board. This register may be set to allow or not allow a task to execute I/O, halts or to acknowledge interrupts. In addition, this register enables and disables the front panel stop switch and the auxiliary enable bit. It also determines if the port addresses during the execution of I/O are in long or short mode. Long mode puts the addresses on A8 - A15 and AØ - A7, while short mode puts them on the lower eight address lines only.

Upon the occurrence of a trap, since the EPROM code has successfully stored all the registers of the trapped task, the supervisor can examine the conditions which led to the trap and determine a course of action. Since the program counter for the task has been saved and contains the location of the next instruction which would have been executed had the trap not occurred, the supervisor may handle the trap and then return to that task.

For instance, in the event a task runs out of memory space, the supervisor may decide to grant the task more memory and then to return to the very instruction which caused the trap to occur. Only this time, the execution continues without any traps. Another example: A task is interrupted by external interrupt logic when a disk file had been successfully loaded, but the task was not allowed to acknowledge interrupts. In this case the task might have been waiting for the contents of the file for

further execution. The condition causes a trap to the supervisor which handles the interrupt, transfers the data to the task and then returns to that task.

As in the case of the memory management hardware, the power of the CPU lies in its ability to trap as described. There are cases when the CPU may want to be configured as a standard S-100-type CPU board without any of these features. In this case, the on-board EPROM contains code to inhibit all the trapping mechanisms from activating which are executed before the CPU jumps to the desired task. This would be typical in a single type environment, such as CP/M, which does not support the trap features. By simply writing a 2Bh into the mask register, all traps are inhibited and the CPU board emulates other S-100 Z80 CPU bus masters.

1.4. Local Logic

The following section describes the Decision CPU local logic required to use the memory mapped functions on the CPU board itself. This circuitry is not affected by operations on the S-100 bus and is only activated when in task 0 (the supervisor).

The Z80 data and address lines are buffered for internal use by chips 13D and 16A. The data lines are connected to the on-board registers, RAM, EPROM and floating point processor. These devices are all memory mapped into the first 4K of memory space in task 0 (see memory map chart for overview). The devices on the board are activated only when the signal LOCAL STB is true. This signal becomes active when 1) a memory request for the first 4K segment of task 0 occurs, 2) whenever a trap has occurred or 3) whenever the CPU is switching to a new task (condition decoded by chips 3A, 4A 10A or 11A). LOCAL STB is the enable for chip 6C which decodes address lines 10 and 11 to form the 1K boundaries within the first 4K where devices reside. The decoding is as follows:

Table 1-5: Decoding - Address Lines 10 and 11

A11	A10	DEVICE	ADDRESS
0	0	Local RAM	0000 - 03FFh
0	1	Local I/O (Registers, Maps)	0400 - 07FFh
1	0	EPROM	0800 - 0BFFh
1	1	Floating Point Processor	0C00 - 0FFFh

The signal LOCALIO is further decoded with WR STB, AØ and A1 (by chip 12A) to form the chip selects for the various registers. The registers are mapped as follows:

Table 1-6: Map of the Decoding Registers

A1	AØ	WR STB	DEVICE	ADDRESS
Ø	Ø	Active	Front panel segment	Ø4ØØh
Ø	1	Active	Front panel column	Ø4Ø1h
1	Ø	Active	Task Register	Ø4Ø2h
1	1	Active	Mask Register	Ø4Ø3h
Ø	Ø	Inactive	Trap Address Register	Ø4ØØh
Ø	1	Inactive	Front panel keyboard	Ø4Ø1h
1	Ø	Inactive	Switch register	Ø4Ø2h
1	1	Inactive	Trap Status Register	Ø4Ø3h

The signal LOCALIO is also decoded with AØ and A9 (chip 6C) to form the chip selects for the map RAMs (chips 6B, 7B and 9B). Chips 7B and 9B are the map address RAMs and 6B is the protection attributes RAM. They are activated as follows:

Table 1-7: RAM Activation

A9	AØ	DEVICE	ADDRESS
Ø	Ø	inactive	----------
Ø	1	inactive	----------
1	Ø	Mapping RAM	Ø6ØØh - Ø7FEh (even)
1	1	Attribute RAM	Ø6Ø1h - Ø7FFh (odd)

Each segment in the map RAM has a corresponding protection attribute in the next memory location. When writing to the Map and Attribute RAM chips, CPU address lines A1 - A8 are used to select the desired memory location within the map. When the Map and Attribute RAMs are not being accessed, their address lines become the lower four bits of the task register and address lines A12 - A15. The multiplexing is performed by chips 1ØB and 11B.

Since the decoding circuit does not use the RD STB or the WR STB to decode the Map and Attribute RAM WE line, at no time should an attempt be made to read these memories from task Ø. This causes their contents to be altered. Remember, this is write-only memory from task Ø and a copy of its contents should be kept in memory, which is read/write (see Section 2).

The mask register (chip 8C) allows a number of trap conditions to be set up. These conditions are detailed in the section on

Decision CPU Trap Hardware. The trap logic is enabled (set) whenever the task register is written (location 402h in task Ø is written whenever a new task is swapped). When one of the mask conditions have been violated, a CPU trap occurs. The events are as follows:

1. Chip 1B, pin-6, goes high (signal called PRETRAP). This latches into chip 14B the condition which caused the trap. At this time, the address at which the trap occurred is shifted into chip 12B (only A12 - A15 are saved).

2. On the next M1 cycle, the signal called TRAP (pin-9 of chip 1B) goes high. This latches the address at the time of trap and the next address after the trap into chip 13B, the Trap Address Register.

3. TRAP also enables the on-board EPROM and forces its address lines to ØBFØh (accomplished by chips 15B, 15C and 16C). This forces the CPU to begin executing the code in the EPROM which saves all the registers of the trapped task. After the 15th instruction the EPROM jumps to Ø8ØØh where the EPROM memory space begins. After the 15th instruction, the CPU is allowed to execute at its normal PC address and the EPROM is no longer forced active.

The reset operation is exactly as above since it is actually a trap condition. During TRAP RESET however, the highest address of the 2716 EPROM is set low. Since only a 1K segment is available for EPROM, only half of the EPROM is visible at any one time. Code executes out of the lower half of the EPROM until the Task Register is written the first time. (The lower half of the EPROM is no longer accessible until the RESET switch is pressed). Latch 4C generates the signal TRAP RESET and is set by the signal TASK.

The signal TRAP HALT is generated by decoding DIØ - DI7 (the halt opcode is 76h) during an M1 cycle. The decoding is performed by chip 8D, 9D and 5C. Chip 5C (16R4 PAL) is used to generate a modified M1 signal for use by the trap logic. The Z8Ø chip has extended opcodes from the 8Ø8Ø instruction set. In order to execute these instructions, the Z8Ø chip generates an additional M1 cycle. Whenever the PAL detects one of these opcodes (DI lines contain CBh, DDh, EDh or an FDh), it inhibits the second M1 signal from reaching the trap circuits. This assures that the traps always occur on actual instruction boundaries and not in the middle of an instruction.

Chips 2B and 4A, in conjunction with the STOP ENBL and RUN ENBL outputs of the Mask Register, allow the CPU to be trapped in the middle of execution of a task. If pin-4 of 2B is grounded, and the SOP ENBL line of the Mask Register is low, a trap occurs. If the RUN ENBL line is held high, the CPU is allowed to execute a task until some other trap occurs. However, if this line is low,

the task is allowed to execute only one instruction and then another trap occurs. By using this function, the CPU may be single-stepped through code in a particular task.

Chips 1A, 3A, 3B, 7D and 8D make up the circuitry which generate the signal, TRAP VOID. This trap occurs whenever I/O was attempted in a task not allowed I/O operations (the I/OENBL bit of the Mask Register is high) or if a memory segment has been violated in some way.

The protection attributes from the Attribute RAM are input to chip 1A. If pin-9 (SEL MEM) is low, an I/O operation is occurring and is trapped (one of the DØ through D4 lines appears on the Y output, pin-5). If SEL MEM is high, the contents of R8 and R9 (the Attribute RAM outputs) determine which of the D4 through D7 input lines of chip 1A appear on the Y output.

The Y output of chip 1A is latched by chip 7D to create the signal TRAP VOID. The signal TRAP VOID generates the signal NULL BUS, which aborts the attempted operation (chip 8D generates a NULL BUS on either TRAP VOID or on the signal LOCAL). Latch 7D is used to provide the signal ACCESS. This signal indicates that the task had access to the memory during the last M1 cycle and should have access during the next cycles (used in execute-only memory segments).

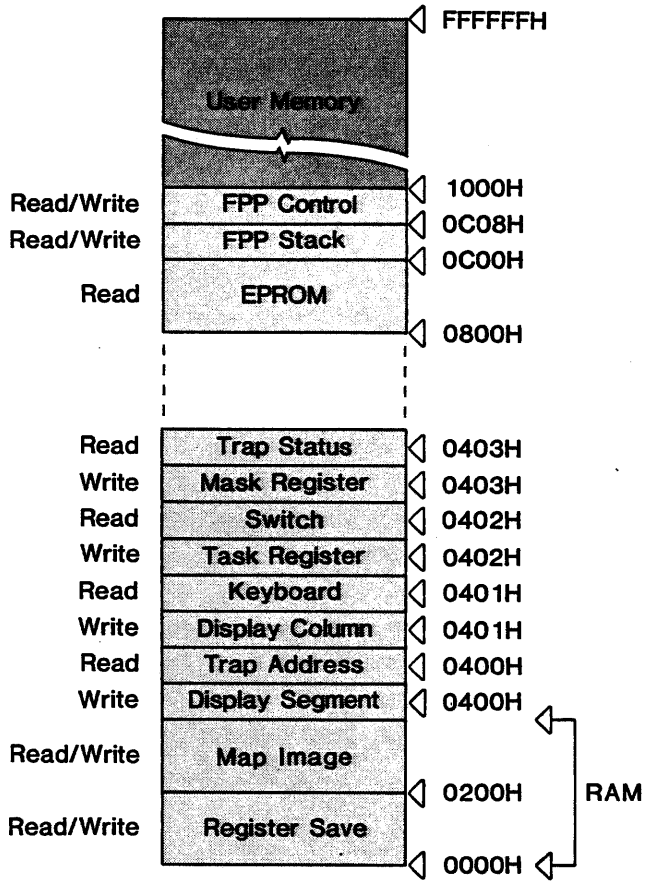


Fig. 1-1: CPU Memory Map Diagram

2. MPZ8Ø MONITOR PROGRAM

The following monitor description applies to MPZ8Ø EPROMs at revision level 3.7 and above (EPROMs marked MON 3.7* or greater). This description is slightly different for Micronix (versions marked MON 4.4*).

The MPZ8Ø processor board contains a monitor (location 16C) which allows the user to read and write memory or I/O locations. This monitor contains a memory test routine, a routine to establish the highest addressable memory location, a hex math routine and a fill command which fills a block with a constant. One routine BOOT (or b) also allows the user to jump to an address and execute code stored there. The following table depicts common switch settings for Morrow Design products.

Table 2-1: Switch 16C - Boot Addresses

S1	S2	S3	S4	S5	ADDRESS
OFF	OFF	OFF	OFF	OFF	F8ØØh
OFF	OFF	OFF	OFF	ON	FØØØh
OFF	OFF	OFF	ON	OFF	E8ØØh
OFF	OFF	OFF	ON	ON	EØØØh
OFF	OFF	ON	OFF	OFF	D8ØØh
OFF	OFF	ON	OFF	ON	DØØØh
OFF	OFF	ON	ON	OFF	C8ØØh
OFF	OFF	ON	ON	ON	CØØØh
ON	ON	ON	OFF	ON	BOOT DJDMA
ON	ON	ON	ON	OFF	BOOTMW
ON	ON	ON	ON	ON	BOOTHD

Besides the address switches, two other switches, S6 and S8 should be set as follows:

S6 - ON - Forces the CPU to power up in the Monitor program.

OFF - Forces the CPU to perform a power-on-jump to the location specified by the CPU switches, or to boot the appropriate disk device (see below.)

S7 - Not implemented.

S8 - ON - Enables the decoded S-1ØØ signal MWRITE to go out on the S-1ØØ bus. The DJ2B requires this signal for correct operation.

OFF - Disables MWRITE from coming out on the S-1ØØ bus.

Switches 1 through 5 may also be set to accommodate requirements of other vendors. Below is a complete list of available jump addresses:

Table 2-2: Switch 16C - Power-On-Jump Addresses

S1	S2	S3	S4	S5	JUMP LOCATION
OFF	OFF	OFF	OFF	OFF	F8ØØH
OFF	OFF	OFF	OFF	ON	FØØØH
OFF	OFF	OFF	ON	OFF	E8ØØH
OFF	OFF	OFF	ON	ON	EØØØH
OFF	OFF	ON	OFF	OFF	D8ØØH
OFF	OFF	ON	OFF	ON	DØØØH
OFF	OFF	ON	ON	OFF	C8ØØH
OFF	OFF	ON	ON	ON	CØØØH
OFF	ON	OFF	OFF	OFF	B8ØØH
OFF	ON	OFF	OFF	ON	BØØØH
OFF	ON	OFF	ON	OFF	A8ØØH
OFF	ON	OFF	ON	ON	AØØØH
OFF	ON	ON	OFF	OFF	98ØØH
OFF	ON	ON	OFF	ON	9ØØØH
OFF	ON	ON	ON	OFF	88ØØH
OFF	ON	ON	ON	ON	8ØØØH
ON	OFF	OFF	OFF	OFF	78ØØH
ON	OFF	OFF	OFF	ON	7ØØØH
ON	OFF	OFF	ON	OFF	68ØØH
ON	OFF	OFF	ON	ON	6ØØØH
ON	OFF	ON	OFF	OFF	58ØØH
ON	OFF	ON	OFF	ON	5ØØØH

Table 2-2, Cont.

ON	OFF	ON	ON	OFF	4800H
ON	OFF	ON	ON	ON	4000H
ON	ON	OFF	OFF	OFF	3800H
ON	ON	OFF	OFF	ON	3000H
ON	ON	OFF	ON	OFF	2800H
ON	ON	OFF	ON	ON	2000H
ON	ON	ON	OFF	OFF	1800H
ON	ON	ON	OFF	ON	Boot DJ/DMA
ON	ON	ON	ON	OFF	Boot HD/DMA
ON	ON	ON	ON	ON	Boot HDCA

Switch 6 determines if the board will power up allowing access to the monitor program. If the switch is ON, the board comes up in the monitor. If the HDCA, HD/DMA or DJ/DMA devices have been selected however, the first sector of the appropriate disk drive will be loaded into memory before the monitor program is entered. If this controller is not responding, a time-out (approximately one minute) will occur and the monitor will be entered regardless of the condition of switch 6.

The map RAM is written to give task Ø the first 64K of available memory; task 1 has the same memory block. Tasks Ø and 1 have unlimited access to this memory. The monitor allows reading, writing and executing any memory location. If switch 6 is OFF, the CPU jumps to the location specified by the switches.

Switch 7 is currently a spare bit; switch 8 generates the S-1ØØ signal MWRITE (decoded from sOUT and PWR) for devices which require this signal. In most configurations the switch remains in the ON position.

Morrow Designs currently offers several disk products which cannot be booted by merely jumping to a memory address. For this reason, the MPZ8Ø monitor has several routines built-in to allow the user to boot from these devices. The MPZ8Ø switch settings for these devices are depicted in the following table:

Table 2-3: Switch 16C - MPZ8Ø Switch Settings for Morrow Disk Products

S1	S2	S3	S4	S5	DEVICE
ON	ON	ON	ON	ON	HDCA Controller
ON	ON	ON	ON	OFF	HD/DMA Controller
ON	ON	ON	OFF	ON	DJ/DMA Controller

For these controllers, special routines have been incorporated so that when these switch settings are active, the monitor will restore the appropriate disk drive to track Ø and load-in the first sector. At this point it examines switch 6 of the MPZ8Ø (monitor switch) to see if the user wishes to enter the monitor program or directly boot up the system without entering the monitor. It is important to note here that the EPROM attempts to load in the sector in both cases. If there is any error while attempting to load-in the sector, the monitor will be invoked and an appropriate error status will be sent to the terminal. For decoding these error reports, see Section 3.

2.1. Booting the HDCA Controller

Morrow Designs' HDCA controller is a port-mapped device which controls and drives either Shugart SA4ØØØ, Fujitsu M23Ø1B or Winchester M23Ø2B hard disks. Morrow Designs' BOOTHD program has been implemented in the Decision I firmware. The program has been modified slightly to time-out after one minute of attempting to load a sector. If a drive is not ready, or an error occurs, the boot routine will abort and the monitor will be entered. The monitor will then print an error message. Any attempt to execute the boot command will re-invoke the monitor program. In the normal CPM for the HDC disks, the sector gets loaded in beginning at address Ø1ØØh. While in the monitor, the user may type:

```
D11ØØ,14ff      (CR)
```

to display the data (sector contents) which were actually loaded into the system memory. The monitor, due to its memory mapping, loads the data into the first segment of task 1's space. This is the reason for dumping location 11ØØ rather than 1ØØ in the step above. This should be helpful for debugging in cases where the disk will not boot.

2.2. Booting the HD/DMA Controller

Morrow Designs' HD/DMA controller, being a channel-driven device, requires system memory to be able to execute commands. This controller was designed for the Seagate ST-5Ø6 and Shugart SA1ØØØ type of interface. The command channel address (set by the MPZ8Ø EPROM) is at 8Øh. To view the commands, the user simply types from the monitor:

D1Ø8Ø,1Ø8f (CR)

This will display the 16 bytes of commands used by the HD/DMA. (Note: these commands are written into channel memory only if the switch settings on the MPZ8Ø are set for this device). If a drive is not ready, or an error occurs, the boot routine will abort after approximately one minute and the monitor will be entered. The monitor will then print an error message. Any attempt to execute the boot command will simply re-invoke the monitor program. In the normal CPM for the HD/DMA disks, the DMA address is set for 1ØØh. Provided everything is functioning, there should be 1Ø24 bytes of data loaded-in (provided the sectors are formatted for 1Ø24 bytes, as defined by Morrow Designs software). While in the monitor, the user may type:

D11ØØ,14ff (CR)

to display the data (sector contents) which were actually loaded into the system memory. The monitor, due to its memory mapping, loads the data into the first segment of task 1's space. This is the reason for dumping location 11ØØ rather than 1ØØ in the step above. This is also true for viewing the command channel and should be helpful for debugging in cases where the disk will not boot.

2.3. Booting the DJ/DMA Controller

The MPZ8Ø EPROM code will allow the DJ/DMA to load-in a sector, provided the switches on the MPZ8Ø have been appropriately set and the DJ/DMA has been set up to power on boot in "hog" mode. This sector is 128 bytes long and is loaded-in at address 8Øh. Thus, by typing

D1Ø8Ø,1Øff (CR)

from the monitor, the user may examine the sector data loaded into memory by the DJ/DMA. If the DJ/DMA does not respond correctly, or reports an error, the boot routine will time-out after approximately one minute and the monitor will be entered reporting an error status. Any attempt to execute the boot command will cause the monitor to be re-invoked.

2.4. I/O and PIC Initialization

The monitor is set up to perform its I/O either through the Mult I/O or the Wunderbuss I/O. The monitor initializes the three UARTs with an 8-bit word length, 2 stop bits and parity inhibited. The baud rate is switch selected using switch 1ØA on the Wunderbuss I/O motherboard. The following table indicates the switch settings for the baud rates available in the standard version.

Table 2-4: Switch 1ØA - Standard Baud Rate Settings

S1	S2	S3	Baud Rate
ON	ON	ON	96ØØ*
ON	ON	OFF	192ØØ
ON	OFF	ON	96ØØ
ON	OFF	OFF	48ØØ
OFF	ON	ON	24ØØ
OFF	ON	OFF	12ØØ
OFF	OFF	ON	3ØØ
OFF	OFF	OFF	11Ø

Since the Mult I/O does not have any readable switches, the monitor uses the default* baud rate of 96ØØ baud. The standard base address of the WB I/O or the Mult I/O is assumed to be 48H. As already mentioned, the monitor initializes the UARTs on the Mult I/O board (Rack Mount Decision I) and the Wunderbuss I/O (Table Top Decision I). In addition to initializing these UARTs, the monitor checks to see if they can transmit and receive characters correctly by forcing them into a test mode. If any UART should fail, an error message will be printed on the terminal (assuming that serial port 1 is active). The user will be prevented from booting the system until the problem is remedied.

The monitor initializes the 8259 Programmable Interrupt Controller as a single master PIC in 8Ø8Ø mode. The interrupt mask has been initialized in the OFF position to disable interrupt requests from the various devices. The "Call Vectors" have been set to point to the eight 8Ø8Ø restart locations; the PIC has been set in level-triggered mode.

The monitor also examines the bus to see if there is any memory in the system. If there is RAM in the system, the monitor will print out the highest available address (the program checks memory from the top down). If there is no RAM, the monitor prints an error message and will not allow the user to boot the system until there is some RAM in the system.

3. MONITOR ERROR MESSAGE INTERPRETATION

The following sections interpret the error messages displayed when booting the disk controllers, initializing the UARTs, or when no addressable memory is present in the system.

3.1. HDCA Controller Errors

Whenever an error is encountered attempting to boot this controller, the message:

Dxxyy

will be sent to the terminal. The D indicates the error was caused by the HDC I/O controller. The xx value is a hex value returned from the HDCA secondary status port. The yy value represents the byte returned from the HDCA primary status port.

3.2. HD/DMA Controller Errors

Whenever an error is encountered trying to boot the HD/DMA controller, the message:

Hxxyy

is sent to the terminal before the monitor is invoked. The H character indicates the error was caused by the HD/DMA. The xx represents the command issued to the HD/DMA that caused the error and the yy represents the status returned by the controller. The commands and status may be broken down as follows:

Table 3-1: HDCA Error Code Summary

Commands			Status		
ØØ	-	Read	ØØ	-	Busy
Ø1	-	Write	Ø1	-	Not ready
Ø2	-	Read header	Ø2	-	Wrong cylinder
Ø3	-	Format	Ø3	-	Wrong head
Ø4	-	Load constants	Ø4	-	Header not found
Ø5	-	Return status	Ø5	-	Data not found
Ø6	-	Seek	Ø6	-	Data overrun
			Ø7	-	Data CRC error
			Ø8	-	Write fault
			Ø9	-	Header CRC error
			FF	-	O.K.

3.3. DJ/DMA Controller Errors

Whenever an error is encountered attempting to boot the DJ/DMA controller, an error:

Fxxyy

is sent to the terminal. The **F** indicates the error was caused by the DJ/DMA. The **xx** is the low byte of the address the MPZ8Ø was instructed to jump to by the DJ/DMA. The **yy** is the byte of status returned by the DJ/DMA. The status is interpreted as follows:

Table 3-2: DJ/DMA Error Code Summary

ØØ	-	Controller not responding
4Ø	-	O.K.
8Ø	-	Improper command
81	-	Illegal disk drive
82	-	Drive not ready
83	-	Illegal track
84	-	Media not readable
85	-	Improper sector header/missing sync byte
86	-	Header CRC error
87	-	Seek error
88 - 8D	-	Compare error in sector header scan
8E	-	Data CRC error
8F	-	Illegal sector value for current media
9Ø	-	Media is write protected
91	-	Lost data - DMA channel did not respond
92	-	Lost command - channel did not respond

The **xx** byte of the status (low byte of the jump address listed above) will, in most cases, be an 8Øh, since the first sector normally loads-in at 8Øh (128 bytes long).

3.4. UART Errors

Whenever a UART error is encountered during the monitor initialization sequence, the message:

Uxxyy

will be printed at the terminal before the monitor program is invoked. The U indicates a UART caused the error. The **xx** indicates which UART caused the error (1, 2 or 3) and the **yy** represents the data byte returned by the UART.

Note that the character sent to the UARTs is a byte of ØØ. The UARTs should return this value. In the case of an error, the user should compare the faulty bits returned as **yy** above. Also note that the monitor performs its I/O through serial port 1; if this port is dead, nothing will appear at the terminal.

3.5. Memory Errors

The monitor checks only for the presence of memory in the system. If it detects there is no addressable memory in the first 64K of memory address, the error

MØBAD

is sent to the terminal. Normally the monitor will print the address of the highest byte of usable memory in the first 64K of user space (for most systems this will be FFFF).

The following is a summary of the monitor commands which are available upon a system reset:

Table 3-3: Monitor Commands Summary
(page 1 of 3)

CMD	Parameters	Description
B	(none)	Jump to an address selected by the switches on the CPU card. The CPU will begin executing in the Task specified by the CTASK memory location (ØØØ6h). This location is initialized for task 1. The memory map is set for the first 64K of user memory. CPU trap mechanisms have been set to trap on the pressing of the front panel stop switch or the execution of a halt instruction.
C	(none)	Continue the task which just trapped with CPU registers and memory map all restored to the condition before the trap and the users mask register is returned to its original state. Execution begins at the instruction which was in the Program Counter before the trap occurred.
D	xxxx,yyyy (CR)	Display memory contents beginning with memory address xxxx and ending with memory address yyyy.
F	xxxx,yyyy,zz (CR)	Fill memory from xxxx to yyyy with constant zz. Constant is a one byte hexadecimal number.

Table 3-3, Cont.
(Page 2 of 3)

G	xxxx (CR)	Go to memory location specified by xxxx and begin execution.
H	xxxx,yyyy (CR)	Hex math performs the following: xxxx + yyyy and then xxxx - yyyy The numbers are represented as 2 byte hexadecimal integers and the results are printed on the following line with the indicated format.
I	xx (CR)	Input data from one of the 256 available input ports and display contents in binary form. The port is specified as a 1 byte hex number.
M	xxxx,yyyy,zzzz	Move the memory contents beginning with location xxxx and ending with yyyy to new location beginning with zzzz.
O	xx,yy (CR)	Output data specified by yy to one of the 256 output ports specified by the one byte hex number xx.
S	xxxx (~~CR~~) SP	Substitute the memory location specified by xxxx with a value. The command first prints the contents of the location. If the location is to be modified, type in new value yy. If the location is not to be modified, hit (CR) to return back to monitor or (SP) to proceed to the next memory location for modification. Continuing to hit the space will increment one location allowing another memory modification.
T	xxxx,yyyy (CR)	Test a memory location beginning with location xxxx and ending with location yyyy.

Table 3-3, Cont.
(Page 3 of 3)

U	(none)	Continue the task which just trapped, execute the next instruction after the trap and then return to the monitor. The users registers are all restored to their state before the task trapped. This is a single step mode.
V	xxxx,yyyy,zzzz (CR)	Verify the memory location beginning at address xxxx with the location yyyy byte count zzzz. The zzzz indicates the number of bytes compared.

In the monitor, the user has access to the extended functions of the MPZ8Ø CPU. The user can alter the memory map and memory protection attributes, change the trap mask and access the floating point processor and on-board RAM and registers.

4. CPU TRAPPING

The MON3.7* versions of the MPZ8Ø EPROM initialize the supervisor (located at ØØØØh in task Ø) to jump to the on-board monitor whenever a CPU trap occurred. This memory location (ØØØØh) may be changed by the user to point to his operating system, but in the meantime all traps will fall into the monitor program. (MON4.4*, the Micronix EPROM, initializes the supervisor to point to the operating system whenever a trap occurs. The only trap which will cause the monitor program to be invoked by Micronix is the front panel stop switch trap.)

Whenever the monitor is invoked subsequent to a CPU trap, the message:

xxyy

is sent to the console (serial port 1). The value **xx** represents the memory segment the CPU was executing in at the time of the trap (the contents of the Trap Address Register). The value **yy** represents the event which caused the trap condition (the contents of the Trap Status Register). The trap condition may be interpreted from this value by referring to Table 4-4 of this manual.

In addition to this information, the contents of the CPU registers at the time of the trap are also sent to the console. The monitor actually calculates where in memory the registers have been saved and dumps those memory addresses to the console. This allows the user indirect access to the CPU registers. By

changing any of the memory locations where these registers have been saved, the user will change the associated task's register and the next time that task executes, the registers in memory will be restored to the Z8Ø CPU registers.

By examining task Ø memory location ØØØ6h the user may determine which task was last executing. Examining task Ø location ØØØ7h, the user may determine the mask (privilege levels) for the task which just trapped (see Table 4-5 for interpreting this byte).

The task register save area for the MON3.7* EPROMs utilizes the MPZ8Ø on-board RAM and begins at Task Ø memory location ØØØ8h (NOTE: Micronix has a dynamic task register save area which is pointed to by task Ø memory locations ØØ14h - ØØ15h). This task save memory is configured for non-Micronix environments according to the following format:

Table 4-1a: Task Save Area Registers(non-Micronix)

ØØØ8h	Interrupt Register	1 byte
ØØØ9h	IX Register	2 bytes
ØØØBh	IY Register	2 bytes
ØØØDh	Alternate BC Registers	2 bytes
ØØØFh	Alternate DE Registers	2 bytes
ØØ11	Alternate HL Registers	2 bytes
ØØ13	Alternate AF Registers	2 bytes
ØØ15	Program Counter	2 bytes
ØØ17	BC Registers	2 bytes
ØØ19	DE Registers	2 bytes
ØØ1B	HL Registers	2 bytes
ØØ1D	AF Registers	2 bytes
ØØ1F	Stack Pointer	2 bytes
ØØ21	User's Mask	1 byte

The contents of locations ØØ14H and ØØ15H (REGSAV) point to the beginning of the Micronix register save area, which is configured as:

Table 4-1b: Task Save Areas Registers (Micronix)

Pointer	Register	Bytes
*(REGSAV)	CTASK	1
(REGSAV+1)	(cMASK)	1
(REGSAV+2)	Program Counter	2
(REGSAV+4)	Stack Pointer	2
(REGSAV+6)	AF Registers	2
(REGSAV+8)	BC Registers	2
(REGSAV+1Ø)	DE Registers	2
(REGSAV+12)	HL Registers	2
(REGSAV+14)	INTRPT./Flag Regs.	2
(REGSAV+16)	IX Registers	2
(REGSAV+18)	IY Registers	2
(REGSAV+2Ø)	AF' Registers	2
(REGSAV+22)	BC' Registers	2
(REGSAV+24)	DE' Registers	2
(REGSAV+26)	HL' Registers	2

There are 15 task save areas associated with each of the 15 tasks which can run with the MPZ8Ø. The 15 task save areas are formatted identically to the layout indicated for task 1. The beginning addresses of each of the task save areas are:

Table 4-2: Task Save Areas - Addresses

T1	begins at location ØØØ8
T2	begins at location ØØ22
T3	begins at location ØØ3C
T4	begins at location ØØ56
T5	begins at location ØØ7Ø
T6	begins at location ØØ8A
T7	begins at location ØØA4
T8	begins at location ØØBE
T9	begins at location ØØD8
T1Ø	begins at location ØØF2
T11	begins at location Ø1ØC
T12	begins at location Ø126
T13	begins at location Ø14Ø
T14	begins at location Ø15A
T15	begins at location Ø174

A short cut to the beginning address of a particular task save area is to multiply the task number (from location ØØØ6 in task Ø) in hex by the hex constant 1A (the size of each task save area) and add the result to the hex number 8 (beginning of the task save areas).

* Parentheses indicate a pointer to the registers noted.

Besides the task's registers, the user may also examine the following registers in task Ø:

Table 4-3: MPZ8Ø Read-Only Registers

Memory Location	Name	Description
4ØØh	Trap Address	This register contains two nibbles describing which of the 16 pages the program was executing in at the time of the trap. The lower nibble contains the address immediately before the trap and the upper nibble contains the actual address of the trap. This information is particularly useful in determining when a task has just overgrown a page boundary and the trap occurred at the crossing of one of these boundaries.
4Ø1h	Keyboard	Implemented on front panel versions only.
4Ø2h	Switch	The upper six bits are the CPU switch locations S1 - S6. Bit-1 is the Interrupt Pending bit and bit-Ø is the Trap Reset bit. Trap Reset is low active while Interrupt Pending is high active.
4Ø3h	Status	This location details which type of event has caused the most recent trap to occur. The bit definitions follow.

Bit	Name	Description
Ø	VOID	Low during an illegal memory or I/O reference.
1	IORQ	Low during an illegal I/O reference.
2	HALT	Low if a halt instruction was attempted and that task was not allowed to execute a halt.
3	INT	Low if an interrupt occurred within a task and that task was not allowed to honor interrupts.

Table 4-3, Cont.

4	STOP	Low when front panel stop is activated.
5	AUX	Low when auxiliary trap is activated.
6	R1Ø	High when a trap was caused by memory reference to a segment with protection bit R1Ø set.
7	RD STB	Low if a read operation was occurring at the time of the trap.

The following is a list of valid trap conditions. All numbers not listed are either invalid or not common and may be decoded from the information listed above:

Table 4-4: Valid Trap Conditions

Register 4Ø3h Contents in hex	Type of Trap
1F	Auxiliary trap during a read operation.
2F	Stop switched pressed during a read operation.
37	Interrupt occurred in a task which was not allowed to handle interrupts.
3C	Trap occurred when a task attempted to execute an IN instruction and the mask was set to prohibit I/O instruction within that task.
3E	Trap occurred when a task tried to read a memory area which was permitted either No Access or Execute-Only permission.
5F	An auxiliary trap occurred during a read operation (R1Ø set).
6F	Stop switched pressed during a read operation (R1Ø set).
77	An interrupt occurred in a task which was not allowed to service interrupts (R1Ø set).

Table 4-4, Cont.

7C	Task attempted to execute an IN instruction and was not allowed to do I/O (R1Ø set).
7E	An attempt was made to read a memory segment which was Execute-Only or No Access (R1Ø set).
9F	Auxiliary trap during a write operation.
AF	The stop switch pressed during a write operation.
B7	Interrupt occurred in a task not allowed to handle interrupts.
BB	The current task attempted to execute a halt instruction and halts were not allowed in that task.
BC	A task has tried to execute an OUT instruction and I/O was not allowed to occur in that task.
BE	A task tried to write into a Read-Only, Execute-Only or No Access memory segment.
DF	An auxiliary trap occurred during a write operation (R1Ø set).
EF	Stop switch pressed during a write operation. (R1Ø set)
F7	An interrupt occurred in a task when a task was not allowed to handle interrupts.
FB	A halt instruction was attempted in a task not allowed to execute halts. (R1Ø set)
FC	A task was trying to execute an OUT instruction and the task was not allowed to perform I/O. (R1Ø set)
FE	A task attempted to write into a memory segment that was Read-Only, execute-Only or No Access. (R1Ø set)

The registers above are 'read-only;' when written to, the meaning of each location changes to the following:

Table 4-5: Write-Only Registers

400h	-	Display segment used on front panel option only.
401h	-	Keyboard option used on front panel only.
402h	-	Task Register- this register, when written, causes the CPU to switch to the new task written into it. The upper four bits map directly to A20 - A23 on the S-100 address bus for selecting 1M byte banks. The lower four bits determine which of 16 tasks are executed. After writing into the task register, there is a hardware delay of seven instruction fetches (M1 cycles) before execution begins in the desired task's memory space. The 7th instruction fetch after writing into the task register should therefore be a jump to the desired user routine. The other six instructions may be either 'nops' or any other desired instruction (remember that the Z80 has some instructions which require two M1 cycles to complete) and execute in task 0.
403h	-	Mask register contains the mask which user is allowed to execute. The masks are defined as:

Bit	Description
0	Stop Enable- when low allows the stop switch to cause a trap of current task to task 0. When high, disables the stop switch from affecting a task.
1	Aux Enbl - spare enable output
2	Tint Enable - when low this enables interrupts to be handled within a task. If this bit is high, any interrupt occurring within a task causes a trap-to-task 0.

Table 4-5, Cont.

3	Run Enable - When high allows a task to run without stopping. If this line is low, the task is allowed to execute one instruction and then trap back to task Ø.
4	Halt Enable - When low this bit allows a task to execute a CPU halt instruction. If this bit is high, any attempt to execute a halt instruction in a task causes a trap to task Ø.
5	Sint Enable - when low disables interrupts from occurring in task Ø (supervisor). When high this bit allows interrupts to occur in task Ø.
6	I/O Enable - When low, this bit allows a task to execute CPU I/O instructions. If this bit is high, any attempt by a user to execute an I/O instruction causes a trap to task Ø.
7	ZIO Mode - When this bit is low, the port address during an I/O instruction comes out on the low eight address lines. When high, the port address is echoed onto the upper address lines as well (emulating an 8Ø8Ø CPU). The upper address lines in the ZIO mode contains whatever is on the Z8Ø's upper address line.

4.1. The Stop Trap Feature

The MPZ8Ø CPU board provides a trap-on-stop condition which can be very helpful when debugging software. The "Stop" trap is activated by grounding pin-12 of the "Header socket" located at 12C on the MPZ8Ø board. The CPU will then complete the current instruction executing and enter the monitor program.

For more permanent installations, the user is advised to purchase a 14 pin header to plug into the socket at 12C and an spst type switch (spring loaded to the 'off' position) and wire the switch across pin-12 and pin-7 (ground) of the header. Then by simply depressing the switch, the user will cause a "Stop" trap to occur.

4.2. The Halt Trap Feature

There are two ways to enter task Ø from another task to alter the registers or maps. The first way is to press the "Stop" switch. This causes the users' task to trap and the monitor to begin executing in task Ø. The other way allows the user to enter task Ø, alter whatever registers or memory desired, then return to the original task. The procedure for accomplishing this follows. (Note: This does not apply to Micronix EPROM code.)

1. User program has two consecutive halt instructions imbedded into its code. These halt instructions must reside in the memory area above Ø1ØØØh as the lower 4K of task Ø cannot be masked out.

2. The MPZ8Ø monitor copies the map of the task attempting to execute the two halt instructions into the map of task Ø so they share the same memory areas (from 1ØØØh-FFFFh). The code which was running in the trapping task continues executing in task Ø at the very next location following the two halt instructions. Remember that this code must run above 1ØØØh in task Ø because of the memory mapped facilities in task Ø.

3. After the code task Ø has finished, the user returns to the trapped task for continuation of execution there. To do this, alter the program counter value stored in the trapped task's user save area so it points to the next location to be executed when you swap back to that task (see section on the task save areas). Then simply jump to the routine in the MPZ8Ø EPROM called RESTORE which restores the maps to their previous state, then returns to the trapped task to begin executing at the set PC value.

An example of an altered program is listed on the following page.

Sample Program:

```
                        nmap      equ     0806h
                        restore   equ     0818h
                        tskbse    equ     0803h
                        pcofst    equ     0dh

1000  76                        halt              ;Code executing
1001  76                        halt              ;in task 1 causes
                                                    ;trap-to-task 0.

1002  0E  03                    ld    c,03h       ;This code is execu-
1004  3E  20                    ld    a,20h       ;ting in task 0 and
1006  47               fmap:    ld    b,a         ;writes new map for
1007  CD 06 08                  call  nmap        ;tasks 2 through 15.

100A  3C                        inc   a
100B  FE 00                     cp    00h
100D  C2 06 10                  jp    nz,fmap
1010  3A 06 00                  ld    a,(ctask)   ;get the task #
1013  CD 03 08                  call  tskbse      ;get beginning ad-
                                                    ;dress of task save
1016  11 0D 00                  ld    de,pcofst   ;area- offset to
1019  19                        add   hl,de       ;task's pc-
101A  11 23 10                  ld    de,return   ;modify the task's
101D  73                        ld    (hl),e      ;pc to continue exe-
101E  23                        inc   hl          ;cution after code.
101F  72                        ld    (hl),d
1020  C3 18 08                  jp    restore

                                return  equ $
1023   C3 00 00                 jp  0000          ;CPM warm boot
                                                    ;this code is
                                                    ;now executing
                                                    ;in task 1...
```

Fig. 4-1: MPZ80 Maps Using Halt Trap Feature

4.3. Altering the Memory Maps

In task 0 of the monitor, the user may examine and alter the contents of the memory map. This map contains the memory allocation vectors for 16 tasks and the protection attributes associated with each 1K hex memory segment.

The map that controls the management resides at 00600h - 007FFh and is alterable from task 0 only. This map may only be written to. Never attempt to read this map, it could alter its contents. For this reason, all Morrow Designs firmware keeps a duplicate of this map at task 0 memory locations 00200 - 003FF which is read/write memory. To examine the contents of the map, one must

examine the associated memory locations between ØØ2ØØh - Ø3FFh (referred to as the Image Map. The map between ØØ6ØØh - ØØ7FFh is referred to as the Actual Map). It is advisable to update the Image Map whenever the Actual Map is modified. This is the only way to know exactly what is in the Actual Map.

When in the monitor, examining the locations between ØØ2ØØh - ØØ3FFh reveals some details about the map using the monitor 'D' command, type D2ØØ,3FF (CR)]. The allocations for the various tasks within this map are as follows:

Table 4-6: Task Map Locations

Task #	Image Map Location	Actual Map Location
TASK Ø	ØØ2ØØH - ØØ21FH	ØØ6ØØH - ØØ61FH
TASK 1	ØØ22ØH - ØØ23FH	ØØ62ØH - ØØ63FH
TASK 2	ØØ24ØH - ØØ25FH	ØØ64ØH - ØØ65FH
TASK 3	ØØ26ØH - ØØ27FH	ØØ66ØH - ØØ67FH
TASK 4	ØØ28ØH - ØØ29FH	ØØ68ØH - ØØ69FH
TASK 5	ØØ2AØH - ØØ2BFH	ØØ6AØH - ØØ6BFH
TASK 6	ØØ2CØG - ØØ2DFH	ØØ6CØH - ØØ6DFH
TASK 7	ØØ2EØH - ØØ2FFH	ØØ6EØH - ØØ6FFH
TASK 8	ØØ3ØØH - ØØ31FH	ØØ7ØØH - ØØ71FH
TASK 9	ØØ32ØH - ØØ33FH	ØØ72ØH - ØØ73FH
TASK 1Ø	ØØ34ØH - ØØ35FH	ØØ74ØH - ØØ75FH
TASK 11	ØØ36ØH - ØØ37FH	ØØ76ØH - ØØ77FH
TASK 12	ØØ38ØH - ØØ39FH	ØØ78ØH - ØØ79FH
TASK 13	ØØ3AØH - ØØ3BFH	ØØ7AØH - ØØ7BFH
TASK 14	ØØ3CØH - ØØ3DFH	ØØ7CØH - ØØ7DFH
TASK 15	ØØ3EØH - ØØ3FFH	ØØ7EØH - ØØ7FFH

The monitor currently initializes the map for task Ø and task 1 to share the same 64K of memory space and have unlimited access to this memory. Tasks 2 through 15 have been initialized for 64K and unlimited access beginning with bank 3 (extended address 2ØØØØh - FFFFFh) assigned sequentially. Bank 1 has not been allocated during initialization and is reserved for the future use of task 1. By examining Image Map locations ØØ2ØØh - ØØ23Fh,

the format for writing the map locations may be more closely examined. Typing from the monitor:

```
D200,23F (CR)
```

should display the following contents:

```
200  00 03 01 03 02 03 03 03 04 03 05 03 06 03 07 03
210  08 03 09 03 0A 03 0B 03 0C 03 0D 03 0E 03 0F 03
220  00 03 01 03 02 03 03 03 04 03 05 03 06 03 07 03
230  08 03 09 03 0A 03 0B 03 0C 03 0D 03 0E 03 0F 03
```

Using the previous table, we can see that the task 1 map area begins at location 220h and the first two locations are:

```
220  00 03
```

The first location following the number 220 contains a zero which corresponds to the 16 banks of 64K we are assigning this segment. In a typical environment, each task is assigned its own 64K bank of addressable memory. For instance, task 0's memory would start at 00000 and go to 0FFFF, task 1's memory would start at 10000 and go to 1FFFF (etc for all tasks). Since the monitor initializes task 0 and task 1 with the same map (the first 64K of memory) their image map areas look identical. The contents of this high nibble are defined as:

Table 4-7: High Nibble Contents

Nibble Contents	Bank	Memory Address Range in hex
0	0	00000 - 0FFFF
1	1	10000 - 1FFFF
2	2	20000 - 2FFFF
3	3	30000 - 3FFFF
4	4	40000 - 4FFFF
5	5	50000 - 5FFFF
6	6	60000 - 6FFFF
7	7	70000 - 7FFFF
8	8	80000 - 8FFFF
9	9	90000 - 9FFFF
A	10	A0000 - AFFFF
B	11	B0000 - BFFFF
C	12	C0000 - CFFFF
D	13	D0000 - DFFFF
E	14	E0000 - EFFFF
F	15	F0000 - FFFFF

The zero immediately following the first represents the 4K (decimal) segment within that bank we are assigning. Any 4K segment can be mapped to any other 4K segment but typically, each segment is mapped to one segment of the task's 64K space giving the task 64K of contiguous memory space. By mapping all segments

within a task to the same 4K segment, the task would take up a maximum of 4K of memory space (this is the minimum size of an active task). An example where two users might want to have the same segments assigned to them would be a data storage area which needed to be co-resident in each task. Each task would then have a portion of their memory space shared or common with the other tasks. In this case, only one user might be allowed to update and all the other users were granted read-only access. This greatly enhances system versatility and throughput. The segmentation (lower nibble) of each bank conforms to the following:

Table 4-8: Low Nibble Contents

Low Nibble Contents	Segment	Memory Address Range in hex
Ø	Ø	ØØØØ - ØFFF
1	1	1ØØØ - 1FFF
2	2	2ØØØ - 2FFF
3	3	3ØØØ - 3FFF
4	4	4ØØØ - 4FFF
5	5	5ØØØ - 5FFF
6	6	6ØØØ - 6FFF
7	7	7ØØØ - 7FFF
8	8	8ØØØ - 8FFF
9	9	9ØØØ - 9FFF
A	1Ø	AØØØ - AFFF
B	11	BØØØ - BFFF
C	12	CØØØ - CFFF
D	13	DØØØ - DFFF
E	14	EØØØ - EFFF
F	15	FØØØ - FFFF

The next byte, Ø3, represents the memory protection attributes associated with that allocation vector. The high nibble contents are ignored and the lower nibble contents are arranged as follows:

Table 4-9: Protection Attributes

Byte Contents	Protection Attributes
ØØ	No Access
Ø1	Read-Only
Ø2	Execute-Only
Ø3	Full Access

These protection attributes pertain to the bank/segment selected in the previous byte in the map and are always the odd memory locations within the Image and Actual Maps. Any violation of the above attributes causes a trap-to-task Ø to occur. Trying to

cause a trap to occur (remember that a trap simply means suspending the current task which is being executed and beginning execution of a monitor or supervisor in task Ø).

There is an additional protection bit referred to as RlØ. When RlØ is set, the above access attributes take on a slightly different meaning. The hex values written into the Image and Actual protection RAM are listed below.

Table 4-lØ: Protection Attributes - RlØ Set

Byte Contents	Protection Attributes
Ø4	No Access
Ø5	Read-Only
Ø6	Execute-Only
Ø7	Full Access

Now the difference is that any reference to a memory segment with RlØ set causes a trap-to-task Ø to occur. The attempted access of the segment completes as if it were a 'write to a segment' with RlØ set. If the task which trapped was allowed to grow in size, the access attributes would be rewritten, but this time with RlØ reset (allowing access to the segment). If RlØ is not set and a segment is violated, the read or write attempted would not be completed. Let's look at a typical example where this might be employed.

A user is granted full access to 48K of memory space (attributes written as Ø3 for the first 12 segments). He has 16K of memory space available however these four segments have attributes with RlØ set and Ø7 at Full Access. To conserve memory perhaps each one of these segments was mapped to the same 4K segment. The user is running a program and suddenly his stack overgrows the 48K allotted to him. Perhaps he's just executed a PUSH H (Push HL in Z8Ø mnemonics) and crossed into a segment which had the RlØ bit set. The write operation would occur and then the task would trap. At this point the supervisor, which is running in task Ø, sees that the task was outgrowing its stack and let the user have another 4K of memory.

If RlØ had not been set, however, the supervisor would likely have advised the current task that it had exceeded its memory space and not grant it another segment.

The monitor contains routines to write the Image and Actual Maps which may be used. Remember, these are only available while in task Ø (Monitor). The routine is referred to as NMAP. The routines expect:

'A' register contains the Task number in the upper nibble and the segment number in the lower nibble of the segment you wish to modify.

'B' register contains the new allocation vector (the new contents you wish to write).

'C' register contains the new access attributes you wish the segment to have.

Calling the routine writes the appropriate map and attributes into both the Actual and the Image Map areas. Another routine does the exact reverse ('A' contains the task and segment number) returning the old contents of the map in 'B' and the old access attributes in 'C'. It is called 'GETMAP'.

CAUTION: The monitor allows the user to alter the registers and memory contents in task Ø's first memory segment. Task Ø always has the first 4K assigned to the on-board facilities. If a user inadvertently types the fill command (e.g. FØØØØ,ØFFF,FF) with parameters set for the first segment, the maps would be overwritten and the monitor would cease to function until the RESET switch was pressed (re-initializing the maps). Extreme care must be exercised when accessing the special functions of task Ø's first segment not to inadvertently overwrite these sensitive areas. It is especially important not to write into the Task Register (location Ø4Ø2H) unless you wish to switch to another task. The 7th instruction after writing into the Task Register will be the task just written to. If there is no code there, or if that task's map has not been initialized, results are not be predictable.

5. MPZ8Ø DIAGNOSTICS

The MPZ8Ø monitor contains built-in diagnostic routines to aid in troubleshooting the CPU in the unlikely event of a component failure within the module. These routines are switch selected and generate strobes and patterns at specified points within the module. It is recommended that only service personnel familiar with microprocessor troubleshooting techniques attempt to service the board. Tools necessary are IC DIP clips for ease in probing and a 2 channel, 5Ø MHz bandwidth or greater oscilloscope (external trigger is imperative). The scope should also have a delayed time base for accurate measurements (scopes such as the Tektronix 465 or the Phillips 3214 are more than adequate).

The MPZ8Ø has two distinct modes of operation on power up. The first mode (already discussed in the section on the monitor) is "monitor mode" and is the normal mode of operation. The mode of concern in this section, "diagnostic mode", can be entered only if pin-13 of the header at location 12C is grounded. (NOTE: In MPZ8Ø EPROMS earlier than MON3.7, the diagnostic mode may be entered only after lifting pin-2 of 15D.) The diagnostic mode

will generate troubleshooting patterns and strobes on the board depending upon the settings of the on-board DIP switches. The tests are selected as follows:

Table 5-1: Monitor Switch Settings - Diagnostics

S2	S3	S4	S6	FUNCTION
ON	ON	ON	OFF	Read Registers
ON	ON	OFF	OFF	Write Registers
ON	OFF	ON	OFF	Write Map RAMs
ON	OFF	OFF	OFF	Write R/W RAMs
OFF	ON	ON	OFF	R/W FPP
OFF	ON	OFF	OFF	R/W S-1ØØ Bus
OFF	OFF	ON	OFF	R/W S-1ØØ Bus
OFF	OFF	OFF	OFF	Read Switches

The states of switches 1, 5 and 7 are insignificant in the diagnostic mode. However, if switch 6 is turned on at any point during the test, unless pin-9 of chip 4B or pin-6 of chip 4C have been lifted as in the Bus Read/Write test, the normal MPZ8Ø Monitor program is entered.

Switch S8 should always be on as this switch enables the S-1ØØ signal MWRITE onto the bus.

If pin-13 of header 12C is not grounded, the diagnostic routines will not be available and only the monitor mode is available.

The monitor mode functions are fully described in Section 2.

The procedure then is to ground pin-13 of header 12C, set the switches for the appropriate test routine, set switch 6 (monitor override) to the OFF position and then reset the computer.

Unless the problem has been identified already, the normal approach to troubleshooting the CPU is to sequentially step through the tests checking each point for correct signals. This section assumes that the technician performs these tests in order.

5.1. Preliminary Tests

Before using the tests within the EPROM monitor, the technician should first check for correct voltage levels within the module. Check the outputs of the four, 5 volt on-board regulators. They should be at 5 volts with a maximum of 1ØØ mv of ripple (to accurately measure this noise, be sure scope is isolated from AC ground and is AC coupled). Check pin-16 of chip 17D for a 12 volt level.

Set the CPU switches to run the Read Register tests. Also check chip 17A, pin-6 for the presence of Ø2 clock signal. It should be a 5Ø% duty cycle 4 MHz square wave (25Ø ns period). While the processor is running, check the CPU outputs M1, MRQ, IORQ, WR, RD, RFSH, BUSAK, and HALT. All lines except BUSAK and HALT should be active if you are in the Read Register test. Refer to the Z8Ø Technical Manual (ZILOG, 1977) pages 11 - 17 and page 71 to interpret the relationships of these signals.

Press the RESET switch and using a scope or meter, determine that pin-26 (RESET) on the Z8Ø chip changes state (from 1 to a Ø and back to a 1). Again using a scope, check the WAIT line of the Z8Ø (pin-24) it should be low only during M1 cycles.

If the above portions of the MPZ8Ø are functioning, proceed with the tests. If for some reason the CPU doesn't execute the EPROM code, power up the CPU with pin-3 of the S-1ØØ bus (XRDY) at ground potential. This forces the CPU to wait at location Ø until the XRDY line is brought high. During this state, check the following:

1. Chip 5A pin-6 should be high.
2. Chip 17C pins 1-4 and 22 and 23 should all be high.
3. Chip 17C pins 5-8 should be low.
4. Chip 17C pins 18 and 2Ø should be low.
5. Chip 17C pins 9-17 should all be low.
6. Pins 11-15 of chip 15B should all be low.
7. Chip 4A pin-6 (/XCHG) should be low. (If it is high, check 4A, pin-4, it should be low. If not, check the counter outputs of 4D, pins 8-11. Pin-8 should be high. Toggle the reset line until pin-8 goes high and resume).
8. Chip 1ØA pin-8 should be low.
9. Chip 1ØA pin-12 should be high.
10. Chip 3A pin-8 should be high.
11. Chip 15A pins 1Ø, 11, 12 and 14 should be low and 13 high.

5.2. Read Register Test

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

Place the external trigger probe on pin-2 of chip 14A (S-100 signal SOUT) and adjust trigger level. At the beginning of each pass through the test, the program executes an output instruction and then proceeds with the test. If no output signal is present, either the EPROM code is not running correctly or the CPU/ decoding PROM are not functioning correctly. Do not proceed further until you successfully trigger on SOUT.

Use probe 'A' to examine the outputs of chip 12A, pins 11, 10, 9 and 7. There should be negative-going strobes on these pins of approximately 500 ns in duration. Strobes should sequence as follows: pin-11 first, 10 second, 9 third and 7 last and should have a 3.5 us gap between pulses. The destination of these strobes should also be checked. Check chip 13B, pin-1, 13A, pin-15, chip 15D, pins 1 and 19 and chip 14B, pin-1. Each of these chips enables its outputs onto the internal data bus when its strobe is active. Check to be sure the data outputs of these chips are actually reaching the CPU data lines when the strobe is present.

The test may be summarized as follows:

Read Register Test

1. Read location 400h
2. Read Location 401h
3. Read Location 402h
4. Read location 403h

5.3. Write Register Test

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

This tests checks all the chip select strobes for the on-board registers. Register 402, the task register is not checked with this test but is checked during the S-100 bus R/W test.

With the trigger exactly the same as the last test, use channel 'A' to examine chip 12A pins 15, 14, and 12. They should have approximately 5ØØ ns strobes on them spaced 3.5 us apart and should occur in the listed order. The test first writes zeroes to these registers and then repeats writing 'FFh' to the registers. Check each register to determine if the strobes are reaching their inputs and that they latch the appropriate data onto their outputs. Check P1 (location 12C) pin-1 and 2 for the first 2 strobes (these are used only on the front panel option). Check pin-11 of chip 8C for strobe and pins 2, 5, 6, 9, 12, 15, 16 and 19 for correctly latched outputs.

The test may be summarized as follows:

1. Write a ØØ to location 4ØØh
2. Write a ØØ to location 4Ø1h
3. Write a ØØ to location 4Ø3h
4. Write an FFh to location 4ØØh
5. Write an FFh to location 4Ø1h
6. Write and FFh to location 4Ø3h
7. Repeat until switch settings change

5.4. Write Map RAM Test

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

Again trigger on SOUT but this time to check to see that the Map RAM and Protection RAM can be written. Using probe 'A', check pin-15 of chip 6C to see 9 strobes each 25Ø ns long and spaced at 5 us intervals (approximately). The first eight strobes are the ones of concern in this test. Probe 'A' should then be placed on pin-9 of chip 6C when checking the Protection RAM and pin-1Ø of 6C when checking the Map RAMs.

Channel 'B' is then used to examine the RAM address and data lines during the period when the strobes are active. The test first writes a zero to the first two map locations (6ØØ - Map and 6Ø1 - for protection attributes). Then the test writes an 'FFh' to location 6Ø1h and then 6ØØh. At this point check the data input lines to chips 7B and 9B for the Map, then 6B for the Protection RAM during the times when the enable (as viewed on channel 'A') of the appropriate chip is low. The map RAM address lines should all be low when the three RAM chips are enabled for these four write operations.

The test then writes 'FFh' to locations 7FEh and 7FFh (the last two locations of these RAMs) and then ØØh to 7FFh and 7FEh, in that order. During these writes, all the RAM address lines should be high (the opposite of the first part of this test). If

at any point the address or data lines to the chips are incorrect, check back to the address multiplexers (1ØB and 11B) and make sure that pin-1 of each chip is high and that the address lines are correct.

During this test there should be a total of eight write strobes, four writes to the Map RAMs and four writes to the protection RAM.

The test may be summarized as follows:

1. Write ØØ to location 6ØØh
2. Write ØØ to location 6Ø1h
3. Write FF to location 6Ø1h
4. Write FF to location 6ØØh
5. Write FF to location 6FEh
6. Write FF to location 6FFh
7. Write ØØ to location 6FFh
8. Write ØØ to location 6FEh
9. Repeat until switch settings change

5.5. R/W RAM Test

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

Trigger on SOUT as in the above tests and place channel 'A' probe on pin-8 of either chip 13C or 14C (be sure to check for the strobe at both locations). If there is no strobe, trace back to chip 6C pin-4. The strobes should be approximately 25Ø ns at 5 us intervals. There should be eight chip selects in all, four for the writes and four for the reads.

During the first four chip selects, the address lines should all be low. During the last four chip selects, they should all be high. The WE line (pin-1Ø of the RAMs) should be low as the data

lines are driven by the RAM chips and again, should all be Øs. The next two chip selects (the address lines should all still be low) repeats the first pattern, but the data lines are all high during the strobes. The last four chip selects repeat the pattern of the first four, but with all the address lines high.

The test can be summarized as follows:

1. Write a 00h to location 00h
2. Read from location 00h
3. Write an FFh to location 00h
4. Read from location 00h
5. Write a 00h to location 3FFh
6. Read from location 3FFh
7. Write an FFh to location 3FFh
8. Read from location 3FFh
9. Repeat until switch settings change

5.6. Floating Point Processor Test

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

This test does not actually test the floating point processor but it does check the strobes necessary for correct operation of the FPP chip.

Sync on SOUT and place probe 'A' on pin-18 of chip 17D. There should be two negative-going strobes lasting about 250ns at 3.5 us intervals. A third strobe is present 3.5 us after the second and should last 500ns. If no strobes are present, trace back to chip 6C, pin-7, the origin of this strobe.

During the first strobe, pin-21 of chip 17D is low as are all data lines of 17D (pins 8-15). Pin-19 should be low as well. During the second strobe, pin-21 of 17D should be high and again pin-19 is low as are all the data lines.

During the third strobe, pins 20 and 21 should be low while pin-19 should be high. If there is a 9512 Math Processor in the socket, it should drive the data lines with data from its stack at the occurrence of the last strobe. If no chip is present, the data lines float to a 2 volt level.

The test may be summarized as follows:

1. Write a 00h to location C00h
2. Write a 00h to location C08h
3. Read data from location C00h
4. Repeat until switch settings change

5.7. S-1ØØ Bus R/W Test (high/low)

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

This test requires chip 4C pin-6 and chip 4B, pin-9 be lifted to prevent the MPZ8Ø from attempting to switch to another task. (Remember that anytime the task register is written, the traps are set and a new task is swapped in. In order to test the task register, we must prevent this from occurring by lifting these two pins.) The task register's upper four bits must be written in order to set all the S-1ØØ addresses high.

The purpose of this test is to check all the MPZ8Ø address and data lines during the time it is doing an S-1ØØ bus access. Sync on SOUT, as in all the above steps, and put probe 'A' on the S-1ØØ signal line PWR (chip 13A pin-9, should be a low going strobe). Using channel 'B', check all the address lines during the very first PWR pulse. They should all be high. Next check the addresses during the second PWR strobe. They should all be low.

During the above two PWR strobes, the S-1ØØ data lines should all be low.

Now move the channel 'A' probe to chip 13A, pin-7, PDBIN. During the first PDBIN, all address lines are high. During the second PDBIN, all the addresses should be low.

If any of the address lines are not at their appropriate states during the above test, check the upper four bits of the task register (chip 9C, pins 2, 5, 16 and 19) as they control the S-1ØØ address lines, A2Ø - A23. If address lines A12 - A19 are incorrect, suspect a problem with the Map RAMs and repeat that portion of the diagnostic tests. If there is a problem with AØ - A11, suspect a problem with chips 1ØC or 1ØD, the address drivers.

The test may be summarized as follows:

1. Write a ØØ to location FFFFFFh
2. Write a ØØ to location ØØØØØØh
3. Read location FFFFFFh
4. Read location ØØØØØØh
5. Repeat until switch settings change

5.8. S-1ØØ Bus R/W Test (alternating pattern)

Channel A/B:	2 volts/div
Sweep Mode:	Alternate
Coupling:	DC
Trigger:	External, negative edge
Timebase:	5us/div
Sweep Delay:	Normal

This test requires chip 4C, pin-6, and chip 4B, pin-9 be lifted to prevent the MPZ8Ø from attempting to switch to another task. (Remember that anytime the task register is written, the traps are set and a new task is swapped in. In order to test the task register, we must prevent this from occurring by lifting these two pins.) The task register upper four bits must be written in order to set all the S-1ØØ addresses appropriately.

The purpose of this test is to check all the MPZ8Ø address and data lines during the time it is doing an S-1ØØ bus access. Sync on SOUT, as in all the above steps, and put probe 'A' on the S-1ØØ signal line, PWR (chip 13A, pin-9 should be a low going strobe). Using channel 'B', check all the address lines during the very first PWR pulse. They should all be alternating 1s and Øs, forming the address AAAAAAh.

Next check the addresses during the second PWR strobe. They should also be alternating 1s and Øs, but this time forming the address 555555H. During the above two PWR strobes, the S-1ØØ data lines should first be AAh and then be 55h.

Now move the channel 'A' probe to chip 13A, pin-7, PDBIN. During the first PDBIN, all address lines are AAAAAAh. During the second PDBIN, all the addresses should be 555555h. The data byte will contain the contents of the addressed memory location (most likely FFh, unless there is memory at these addresses).

If any of the address lines are not at their appropriate states during the above test, check the upper four bits of the task register (chip 9C, pins 2, 5, 16 and 19) as they control the S-1ØØ address lines, A2Ø - A23. If the address lines A12 - A19 are incorrect, suspect a problem with the Map RAMs and repeat that portion of the diagnostic tests. If there is a problem with AØ - A11, suspect a problem with chips 1ØC or 1ØD, the address drivers.

The test may be summarized as follows:

1. Write a AAh to location AAAAAAh
2. Write a 55h to location 555555h
3. Read location AAAAAAh
4. Read location 555555h
5. Repeat until switch settings change

6. ENGINEERING SPECIFICATIONS

This section describes the characteristics and specifications for the Morrow Designs Decision I CPU board.

6.1. General Description

The Decision CPU card is a Z8Ø based S-1ØØ computer and is designed to operate as an S-1ØØ bus master. The Decision CPU contains sophisticated trap logic allowing the operating system to constantly monitor the operations of the systems user. Besides the trap logic, the board also contains memory management logic which controls the segmentation of the users' memory from the operating system. The CPU generates a full 24 bits of address space allowing it to access over 16M bytes of physical memory on the S-1ØØ bus.

The CPU also contains 2K bytes of EPROM for firmware routines supporting the memory management and trap logic as well as 1K byte of RAM. Also included on the CPU card is a floating point processor (AMD 9512) and RAM memory associated with the memory mapping and protection logic.

The CPU has a standard 4 MHz system clock and supplies a 2 MHz version of this clock on the S-1ØØ bus for use by bus slaves. Besides the clock, the module provides all signals required to comply with the IEEE-696 proposed standards for the S-1ØØ bus. These include control, status, data and address lines.

The CPU is configured to either process or hardware trap on an interrupt generated by a bus slave. There are two levels of interrupt - those occurring during the execution of a user program and those occurring during the execution of the operating system. Each level is independently controllable with hardware allowing for maximum flexibility with interrupt handling. The CPU handles both PINT and NMI from the S-1ØØ bus.

The Decision I CPU also supports the operation of DMA devices and relinquishes the bus to another device which issues the S-1ØØ signal pHOLD. Upon the reception of a hold request from another device, the CPU completes its current instruction and issues a pHOLDA, thus granting the bus to the requesting device. It is the responsibility of the requesting device to disable the control, status, data and address lines of the Decision CPU by issuing the appropriate S-1ØØ signals (ADSB, DOSB, CDSB and SDSB).

The board may be configured to generate a trap on the occurrence of a power failure (PWRFAIL on the S-1ØØ bus) or an error on the bus (ERROR) allowing maximum flexibility for the operating system.

The Decision CPU has been designed to achieve large system sophistication with high reliability and low cost to provide a the user with years of trouble free service.

6.2. Performance Characteristics

The Decision CPU has been designed to operate in an S-1ØØ environment with adequate ventilation. As a part of the Decision I Computer system, the CPU requires only +8 volts (filtered) and +16 volts (filtered) for correct operation.

The module dissipates approximately 17 watts while active and requires adequate ventilation to remain within the specified, ambient operating environment.

The CPU uses a Z8ØA processor capable of executing at a full 4 MHz clock rate. An optional 6 MHz CPU is also available to increase system throughput. The CPU card allows full use of the Z8Ø instruction set and, through hardware trapping, prevents a halt from being executed inadvertently by a user.

Performance Summary:

Clock Speed:	4 MHz 6 MHz (optional)
System Bus:	S-1ØØ (conforms to the IEEE-696 proposed interface standards for the S-1ØØ bus)
Bus Configuration:	Bus Master
Physical Characteristics:	1Ø.Ø in X 5.425 in. X .Ø62 in. Ø.6ØØ lbs (S-1ØØ standard size)
Layout:	Multilayer (4 layers)

DC Power:

Nominal Voltage	Tolerance	Peak	Average
+8 Volts	Unregulated: filtered must be: > +7.Ø volts - < +25.Ø volts	2.8 Amps	2.Ø Amps
+16 Volts	Unregulated: filtered must be: > +14.5 volts - < +35.Ø volts	Ø.13 Amps	Ø.1Ø Amps

Power Dissipation: 24 watts (absolute maximum)
17 watts (typical)

Environmental Considerations:

Operating:	
Temperature	1Ø C to 4Ø C
Relative Humidity	1Ø% to 9Ø%
Elevation	Sea Level to 12,ØØØ Ft.
Air	Filtered
Non-Operating	
Temperature	-4Ø C to + 6Ø C
Relative Humidity	1Ø% to 9Ø%
Elevation	Sea Level to 12,ØØØ Ft.
Air	Unfiltered

Electrical Interface:

Power Input

+8 volts - pins 1 and 51
Ø volts - pins 5Ø and 1ØØ
+16 volts - pin-2

6.3. Power Requirements

The power requirements of the Decision I CPU are DC voltages of +8 volts unregulated and +16 volts unregulated. Both these supplies must be filtered and at no time should they drop to less than 85% of nominal value. In addition, they should never exceed 2ØØ% of nominal. Both supplies share a common return to ground. When operating modules at voltages greater than 15Ø% of nominal, both power and heat dissipation must be accounted for and ventilation should be increased accordingly.

6.4. Signal Requirements

Signal interface to and from the module is accomplished through the 1ØØ pin edge connector, P1. These signals are defined in Appendix B according to the proposed IEEE-696 Standards for interfacing the S-1ØØ bus.

6.5. Drivers

All drivers on the S-1ØØ module have the following specifications:

High level output voltage:	2.4 volts minimum
	3.4 volts typical
	2.6 mA typical
Low level output voltage:	Ø.5 volts maximum
	Ø.4 volts typical
	24.Ø mA typical
High impedance state:	2Ø.Ø uAmps leakage current

6.6. Receivers

All receivers on the S-1ØØ module have the following characteristics:

High level input voltage:	2.Ø volts minimum
	2Ø.Ø uAmps
Low level input voltage:	Ø.7 volts maximum
	2.Ø mA

6.7. System Reliability

The Mean Time Between Failure (MBTF) is estimated to be 2Ø,ØØØ hours. Failures such as power supply, cooling, bus problems or operator error are not considered failures of the module.

The Mean Time To Repair (MTTR) is estimated to be Ø.5 man hours per incident. MTTR is defined as the average time for trained service personnel to correctly diagnose and repair a failure within the module.

6.8. Preventive Maintenance

The Decision CPU requires no preventive maintenance.

6.9. Service Life

The Decision CPU provides an estimated useful life of 5 years or over 40,000 hours. Repair is permitted within the useful life of the product.

6.10. Grounding

The Decision CPU, since it is operating at 4 - 6 MHz, requires adequate grounding for correct noise immunity and signal generation. The S-100 bus specifications (IEEE proposed standard 696) provide adequate ground connection for correct operation. Pins 20, 50, 53, 70 and 100 provide these ground paths and should be common within the system. In addition, capacitive coupling of signals on the S-100 backplane help to reduce overall performance and reliability. It is therefore advised that this module be operated in an adequately terminated motherboard.

6.11. Installation

The Decision CPU may be mounted in either the horizontal or vertical planes. In either plane adequate circulation of air is required for correct operation.

APPENDIX A

MPZ8Ø CPU SIGNAL DEFINITIONS

The following are the definitions of signals internal to the MPZ8Ø module and are included as an aid in troubleshooting the on-board logic. These signals are listed alphabetically.

Signals are defined as:

Logic 'Low' - Ø.Ø volts to Ø.8 volts

Logic 'High' - 2.Ø volts to 5.Ø volts

Signal	Active	Definition
ACCESS	Low	Indicates if current segment executing had access during last M1 cycle; it has access to memory during the next M1 cycle. This is used for Execute-Only code protection attributes.
AUX ENBL	Low	Spare enable bit of Mask Register
BUS ACK	Low	Indicates Z8Ø is yielding the bus to a DMA device. This signal leaves the board as the S-1ØØ signal pHLDA.
DBIN	Low	Z8Ø is inputting data either for an opcode or data fetch and has tri-stated its data drivers. This signal occurs when either a RDSTB or an INTA is present. It goes out on the S-1ØØ bus as pDBIN.
DELAY	Low	Delay is low whenever the Task Register is written into and remains low for seven M1 (instruction fetches) cycles at which point it returns high. This signal keeps the local devices enabled for the seven fetches and also sets the trap logic to catch next trap. All traps and interrupts are inhibited by hardware when delay islow.
DISP COL	Low	Write strobe generated by writing to address 4Ø1h in task Ø only. Used for front panel display option only.

DISP SEG	Low	Write strobe generated by writing to address 400h in task 0 only. Used for front panel display only.
ENBL MADDR	Low	Enables the map addresses onto the bus (S-100) as A8 - A15. This signal turns on driver 10D unless and I/O operation is in progress.
ENADR	Low	Enables A0 - A7 address lines onto the S-100 bus.
ENDATA	Low	When low, enables data coming from Z80 onto the S-100 bus as D00 - D07. This signal is always low unless a DMA device is controlling the bus.
END	High	Output from the 9512 FPP indicating it has completed the current operation. Reset or a read of the 9512 status register resets this signal.
ERROR	High	Output from the 9512 FPP indicates an error condition in the execution of the last command. RESET or a read of the status register resets this signal.
FPP	Low	Floating point processor enable line, provides chip select to the 9512 chip (17D). Strobe occurs whenever a read or write to C00h - FFFh in task 0 is performed.
HALT ACK	Low	Indicates Z80 has executed a halt instruction. Signal goes onto the S-100 bus as sHLTA.
HALT ENBL	Low	When low allows a task to execute a Z80 halt instruction. If the line is high, a trap occurs whenever a task attempts to execute a halt instruction.
INIT	Low	Upon power up or system reset, this line holds low for one R/C constant and 8 MCL periods following. It is used to clear the Mask, Task and Trap Address Registers and the Trap Reset latch. Init also clears the Interrupt Detect latch.
INP	High	This signal goes high whenever the Z80 is executing an input instruction. It goes out onto the S-100 bus as sINP.

INST	High	Indicates the Z8Ø is in an instruction fetch (M1) cycle. Signal goes onto the S-1ØØ bus as sM1.
INTA	High	The CPU is acknowledging a request to honor an interrupt. It is decoded from I/OREQ and M1 being active and goes on the S-1ØØ bus as sINTA.
INT PEND	High	When active indicates that an interrupt is requested and has yet to be serviced (either PINT or NMI went low).
I/O ENBL	Low	When low, allows a task to execute an input or output instruction. If this signal is high an a task attempts an I/O instruction, a trap occurs.
KEYBOARD	Low	Read strobe generated whenever a read from location 4Ø1h in task Ø is performed. It is used to read the key board in the front panel option.
LOCAL	High	Indicates that task Ø is accessing segment Ø (i.e. RAM, ROM, I/O or FPP) on the CPU card. This signal also allows the Z8Ø address lines A1 - A8 to reach the map RAMs for modifying their their contents if needed. When in local mode, a NULL BUS is generated.
LOCAL STB	Low	Indicates a read or write to segment Ø from task Ø is occurring and XCHG is inactive (low). It is used along with Z8Ø address lines A1Ø and A11 to supply the local device selects for FFP, EPROM, RAM and memory mapped Local I/O.
LOCAL I/O	Low	Used to generate read/write strobes for devices memory mapped at location 4ØØh-4Ø3h in segment Ø of task Ø.
LONG I/O	Low	Enables chip 11C which supplies the port address (from the lower eight address lines of the Z8Ø) during an I/O operation onto the S-1ØØ high address lines A8 - A15. This signal is low whenever I/O occurs and ZI/OMODE is high.
LOW	Low	When low, this signal indicates task Ø is performing a memory operation (i.e. MEMREQ is low) in segment Ø. It is used in the generation of strobes on CPU card.

MØ - M7	-	Used as the address lines for the map RAMs. When LOCAL is active, the low address lines from the CPU reach the Map RAMs. When LOCAL is inactive, the addresses become Z8Ø addresses A12 - A15 (for MØ - M3) and the 4 least significant bits of the Task Register (for M4 - M7).
M1	Low	Z8Ø signal indicating the Z8Ø is beginning an opcode fetch or an interrupt acknowledged cycle.
M1A	High	An inverted version of M1 and delayed by 1 MCLK cycle.
MASK	Low	Strobe generated by writing to location 4Ø3h in task Ø segment Ø. It is used to write the desired trap conditions in the Mask Register.
MCLK	-	Main system clock - 4 MHz (5Ø% duty).
MEMR	High	Decoded from Z8Ø MEMREQ and RDSTB; signal indicates a memory read cycle is occurring. Signal comes out on S-1ØØ bus as sMEMR.
NULL BUS	Low	Low whenever trap occurs as a result of illegal memory access or I/O. This signal is also active when the CPU is in local mode (see LOCAL). NULL BUS prevents the control and status signals from going out onto the S-1ØØ bus, thus voiding the trapped operation from occurring. NULL BUS remains low until DELAY returns to its inactive state.
OUT	High	Indicates Z8Ø performing an output to a port and is decoded from IOREQ; no RD STB present. Signal goes onto the bus as sOUT.
pRDY	Low	S-1ØØ signal causes the Z8Ø to enter a wait state; primarily used for slow memory or peripherals. The Decision CPU generates one wait state on every M1 due to the Z8Ø's fast M1 cycle.

PRETRAP	High	Indicates trap condition detected by the CPU trap detect logic. This signal latches trap conditions into the Trap Status Register (chip 14B); also forces the interrupt multiplexer (chip 6A) into the supervisor interrupt mode. PRETRAP is generated at the beginning of M1.
RØ - R7	-	Outputs of the map RAMs which form the S-1ØØ address lines A12 - A19.
R8 - R11	-	Access RAM outputs which contain the protection/access attributes for any particular segment. These protection attributes are used in the memory trap logic.
RAM	Low	Enables local read/write memory in task Ø, segment Ø area of ØØØØh - Ø3ffh. Provides chip enables for chips 13C and 14C, 1K x 4 RAM chips.
RESET	High	Generated from S-1ØØ PRESET or POC, this signal remains high for one R/C time constant (33Ø ms) and then returns low. It is used to reset the INIT counter (chip 4D) and the 9512 (chip 17D).
ROM	Low	Indicates memory access to segment Ø from task Ø in space from Ø8ØØh - ØBFFh (CPU ROM space).
ROM SEL	Low	Provides the Output Enable (OE) signal for the 2716 ROM (chip 17C) on the CPU card. This signal goes low if any of the following conditions are met: - TRAP VOID - illegal memory access - TRAP HALT - halt instruction trapped - XCHG - processor switching tasks - Task Ø is accessing ROM memory area
RUN ENBL	Low	If this line is low and STOP ENBL is low, the CPU traps (TRAP STOP). This line should be high if the front panel stop switch is to be used. RUN ENBL is bit-3 of the Mask Register.
SEL MEM	High	Indicates current Read/Write operation is a memory reference.

SHORT I/O	Low	Strobes Z8Ø port address onto the low eight address lines of the S-1ØØ bus. This strobe occurs during any I/O instruction execution when ZIOMODE is low.
SINT ENBL	Low	When low, allows interrupts during task Ø to reach the Z8Ø chip. A NMI on the S-1ØØ bus is latched even though the SINT ENBL may be high and remains latched until a INT ACK resets it.
STADR	Low	Latches addresses from the Z8Ø into the address latches (chips 7C, 1ØD and 1ØC) when they are valid. These addresses become the S-1ØØ address lines.
START	High	Indicates the start of a new bus cycle. Signal comes out on the S-1ØØ bus as pSYNC.
STATUS	Low	Read strobe generated from reading location 4Ø3h in task Ø segment Ø. It is used to determine the type of trap last encountered.
STOP ENBL	Low	When low, if RUN ENBL is high, allows the front panel stop switch to generate a TRAP STOP. If high, the front panel stop switch is inhibited. Signal is bit-Ø of the Mask Register.
STVAL	High	When high, indicates that status information on the S-1ØØ bus is valid. It is derived from SYNC ENABLE and MCLK and goes out on the S-1ØØ bus as pSTVAL.
SVRQST	High	Becomes active at the completion of a command by the 9512 if the SVRQST bit in the command word has been set. Signal is cleared by RESET or by executing an instruction with the SVRQST bit low.
SWITCH	Low	Read strobe generated by reading memory location 4Ø2h in task Ø, segment Ø. The upper six bits are the actual switch on the CPU. Bit-1 is the INT PEND signal and bit-Ø is the TRAP RESET signal.
SYNC ENBL	Low	Signals beginning of a Z8Ø non-refresh bus cycle. Signal is used to supply pSYNC on the S-1ØØ bus.

TØ - T7	-	Task Register outputs (chip 9C) of which bits TØ - T3 are used for generating the upper four bits of the map RAM address lines (lower four bits come from A12 - A15). The bits T4 - T7 are used to supply the S-100 address lines A20 - A23.
TASK	Low	Write strobe generated by writing to location 402h in segment Ø from task Ø. It is used to write into the Task Register the desired task to execute. Task also starts the DELAY counter to begin counting M1 cycles (see DELAY).
TINT ENBL	Low	When low allows a task (other than Ø) to use interrupts (NMI or PINT). If this line is high and a task attempts to execute an interrupt, a TRAP INT occurs. It is bit-2 of the Mask Register.
TRAP	High	When active indicates a trap has occurred. This is a delayed and latched version of PRETRAP and produces a NULL BUS as well as latching the address of the trapped instruction into Trap Address Latch (chip 13B). Trap is reset by the signal DELAY when a new task is about to begin.
TRAP ADDR	Low	Read strobe generated by reading address 400h in task Ø, segment Ø. The data from the Trap Address Register is set up as: Bits 4-7 - contain the address of the location which caused the trap to occur. Bits Ø-3 - contain the location of the address before the trap occurred.
TRAP HLT	Low	Indicates a user has attempted to execute a halt instruction and halts had been disabled. This signal disables the data input drivers preventing the CPU from fetching the instruction. Instead, fetch comes from the ROM on the CPU card.

TRAP RESET	Low	Indicates a RESET (hardware) has occurred. The signal enables the lower half of the 2716 EPROM (chip 17C) and remains low until the Task Register is written into, at which point it goes high enabling the upper EPROM. Signal does not go low again unless a RESET is generated.
TRAP VOID	Low	When active indicates an invalid memory (restricted access) operation was attempted or an I/O instruction was attempted and I/O was not activated. In either case a trap occurs.
WMØ	Low	Strobes Map RAM for writing into the 2 RAM chips (9B = low nibble/low address, 7B = high nibble/high address) which control memory segmentation (bits A12-A15)
WM1	Low	Strobe Map RAM for writing into RAM chip 6B, Access privilege RAM. Only the lower nibble is used.
WO	Low	Indicates the Z8Ø is executing a Write operation either to I/O or to memory. Signal is also used in the generation of the S-1ØØ signal sWO. It is decoded from (MEMREQ or I/OREQ) and READ STB.
WRITE	Low	Indicates the Z8Ø is performing a write operation and that the data lines contain valid data to be written to either memory or I/O devices. The signal goes onto the S-1ØØ bus as pWR.
XCHG	Low	When active, indicates either a trap or a RESET has occurred. It is used to enable the EPROM and forces the EPROM address lines to BFØh. XCHG returns high after 15 RDSTBs have occurred.
ZIOMODE	Low	When low allows port addresses during an I/O operation to come out on only the eight lowest address lines (AØ-A15). When high, the port address comes out on both the lowest and upper address lines (A8 - A15). This is bit-7 of the Mask Register.

APPENDIX B

S-100 SIGNAL DESCRIPTIONS

The following signals are supplied or acknowledged by the Decision I CPU for control of S-100 bus slaves:

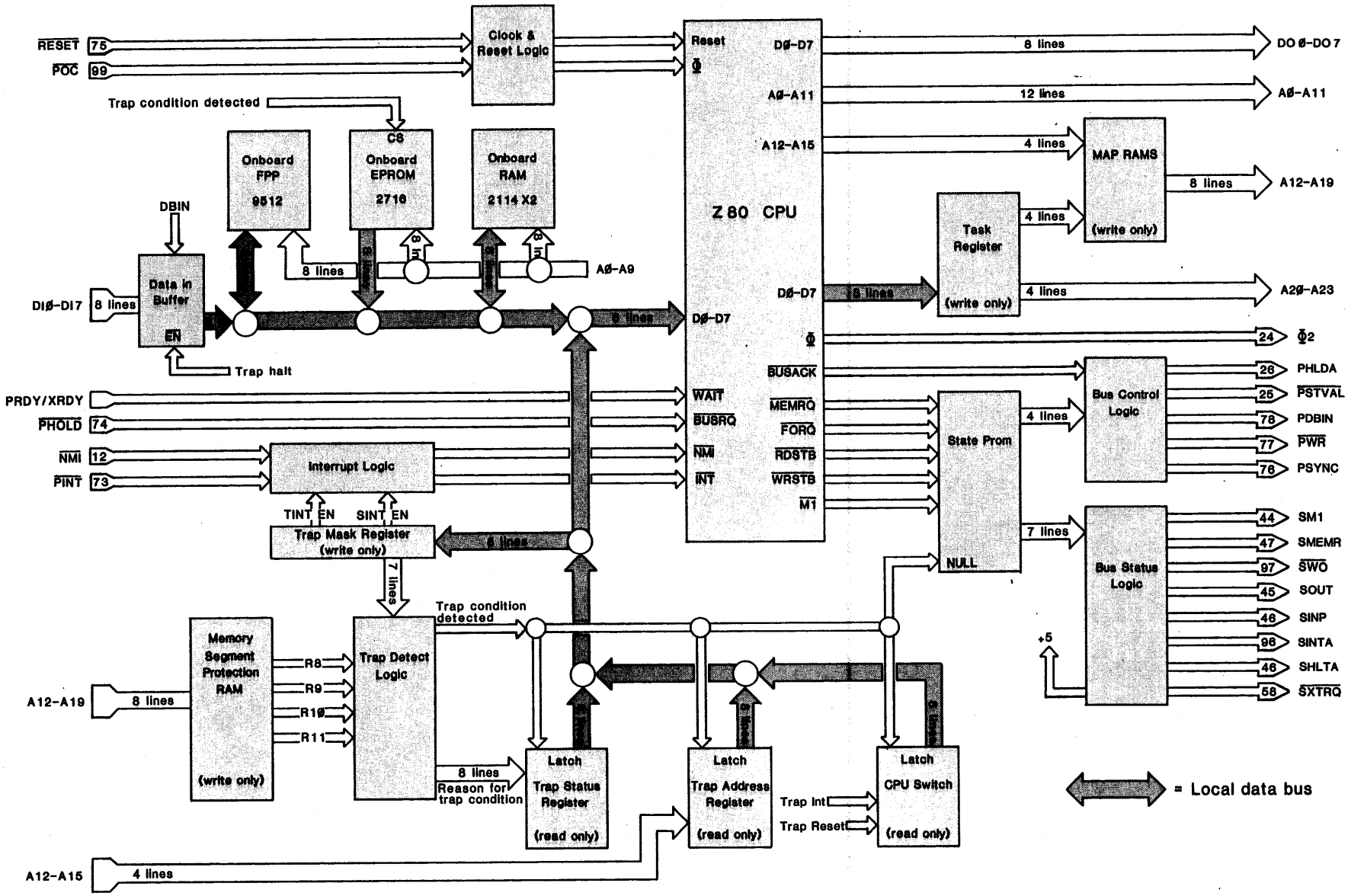
PIN	SIGNAL NAME	DESCRIPTION
1	+8 volts	Power line.
2	+16 volts	Power line.
3	XRDY	Input to CPU from external device indicating it requires a CPU wait operation.
12	NMI	Non maskable interrupt, goes to Z80 NMI line on CPU if Interrupt Trap is not activated.
13	PWRFAIL	Input to CPU indicating power failure.
15	A18	CPU output address line 18.
16	A16	CPU output address line 16.
17	A17	CPU output address line 17.
18	SDSB	Control signal to disable the eight status signals of the CPU board.
19	CDSB	Control signal used to disable the five control signal drivers on the CPU board.
20	GND	Common ground return.
2	ADSB	The control signal used to disable the 24 address line drivers on the CPU.
23	DODSB	The control signal used to disable the CPU data output drivers.
24	Ø	The 4 MHz master timing signal from the CPU board.
25	pSTVAL	Status valid strobe from CPU board.

26	pHLDA	A control signal indicating the CPU is yielding to a DMA device.
29	A5	CPU address line 5.
3Ø	A4	CPU address line 4.
31	A3	CPU address line 3.
32	A15	CPU address line 15.
33	A12	CPU address line 12.
34	A9	CPU address line 9.
35	DO1	CPU data output bit-1.
36	DOØ	CPU data output bit-Ø.
37	A1Ø	CPU address line 1Ø.
38	DO4	CPU data output bit-4.
39	DO5	CPU data output bit-5.
4Ø	DO6	CPU data output bit-6.
41	DI2	Data input bit-2 (to CPU).
42	DI3	Data input bit-3 (to CPU).
43	DI7	Data input bit-7 (to CPU).
44	sM1	Status output from CPU indicating the current bus cycle is an opcode fetch.
45	sOUT	Status output from CPU indicating the current bus cycle is an output.
46	sINP	Status output from CPU indicating the current cycle is an input.
47	sMEMR	Status output from CPU indicating the current cycle is transferring data from a slave to the CPU.
48	sHLTA	Status output from CPU indicating a halt instruction has executed.
49	CLOCK	2 MHz output from CPU, synchronous to the 4 MHz main clock (Ø).
5Ø	GND	Common return ground.

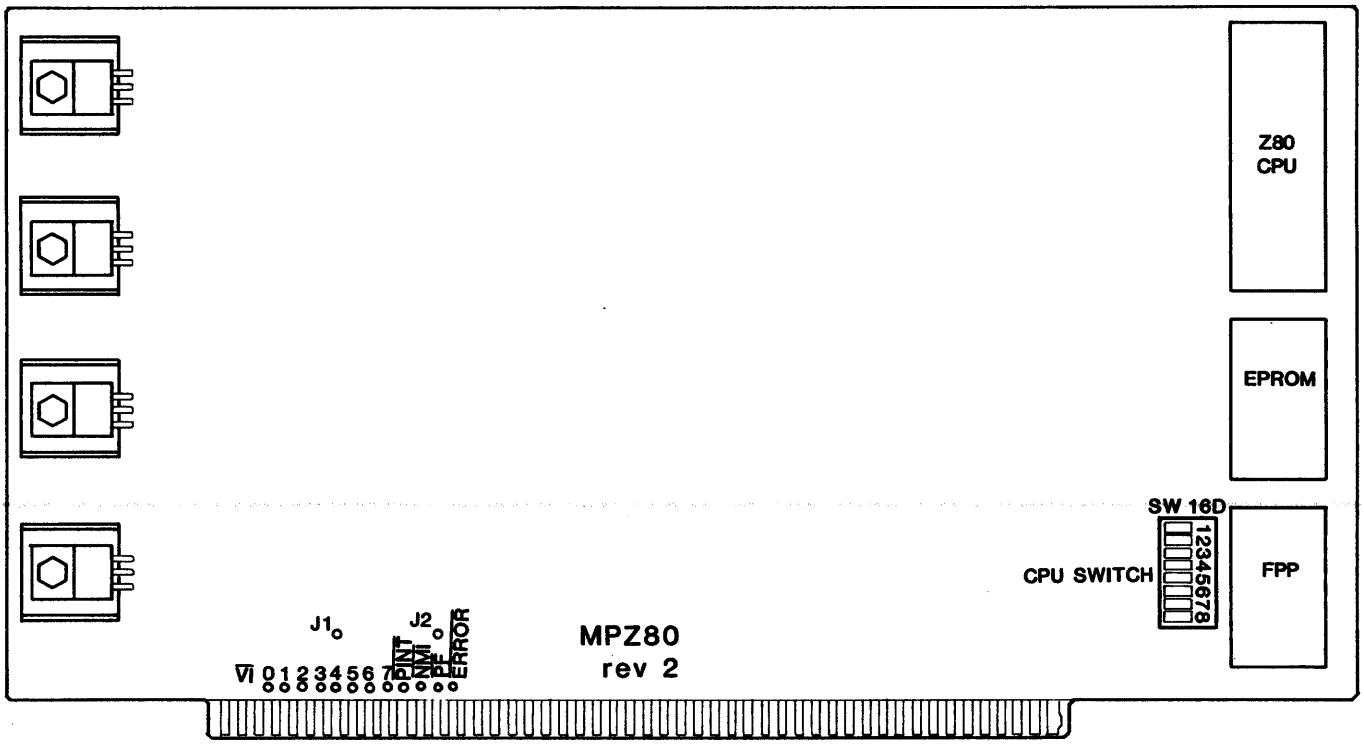
51	+8 volts	Power line.
52	-16 volts	Power line.
53	GND	Common return ground.
58	sXTRQ	CPU status output line held high to signify an eight-bit master controlling the bus.
59	A19	CPU address line 19.
61	A2Ø	CPU address line 2Ø.
62	A21	CPU address line 21.
63	A22	CPU address line 22.
64	A23	CPU address line 23.
68	MWRT	Indicates the CPU is writing data to a bus slave. This is supplied when S8 on the CPU card is ON.
7Ø	GND	Common return ground.
72	pRDY	Indicates slave (or the CPU itself) is requesting the Z8Ø to enter a wait state in the current cycle.
73	pINT	Indicates a device wishes to interrupt the CPU execution.
74	pHOLD	Input to the CPU requesting it to relinquish the bus to another bus master device (DMA).
75	RESET	Signal to reset the CPU.
76	pSYNC	Control output from the CPU indicating the Z8Ø is beginning a new bus cycle.
77	pWR	The CPU control output indicating there is valid data on the DO lines for a bus slave.
78	pDBIN	The CPU control output.
79	AØ	CPU address line Ø.
8Ø	A1	CPU address line 1.
81	A2	CPU address line 2.

82	A6	CPU address line 6.
83	A7	CPU address line 7.
84	A8	CPU address line 8.
85	A13	CPU address line 13.
86	A14	CPU address line 14.
87	A11	CPU address line 11.
88	DO2	CPU data output line 2
89	DO3	CPU data output line 3.
90	DO7	CPU data output line 7.
91	DI4	CPU data input line 4.
92	DI5	CPU data input line 5.
93	DI6	CPU data input line 6.
94	DI1	CPU data input line 1.
95	DIØ	CPU data input line Ø.
96	sINTA	Indicates the bus master is acknowledging a request from an interrupting device.
97	sWO	When low indicates a bus cycle which is transferring data to a bus slave from the bus master.
98	ERROR	May be jumpered to the Decision Trap Aux line - indicates the present bus cycle has generated an error condition.
99	POC	Power-on-clear signal for all bus devices. It remains low for 33 ms after power up.
100	GND	Signal ground.

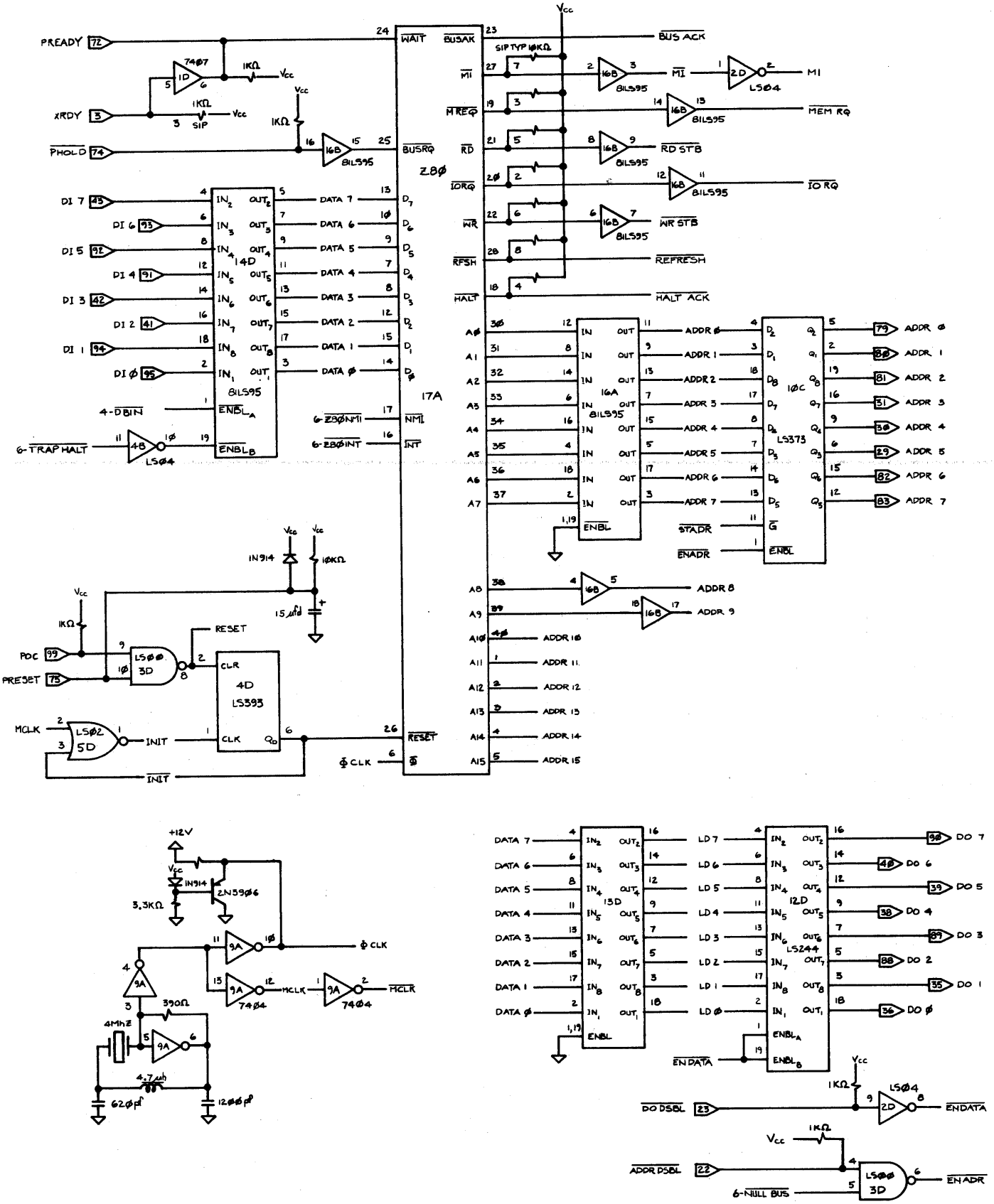
DIAGRAMS/SCHEMATICS



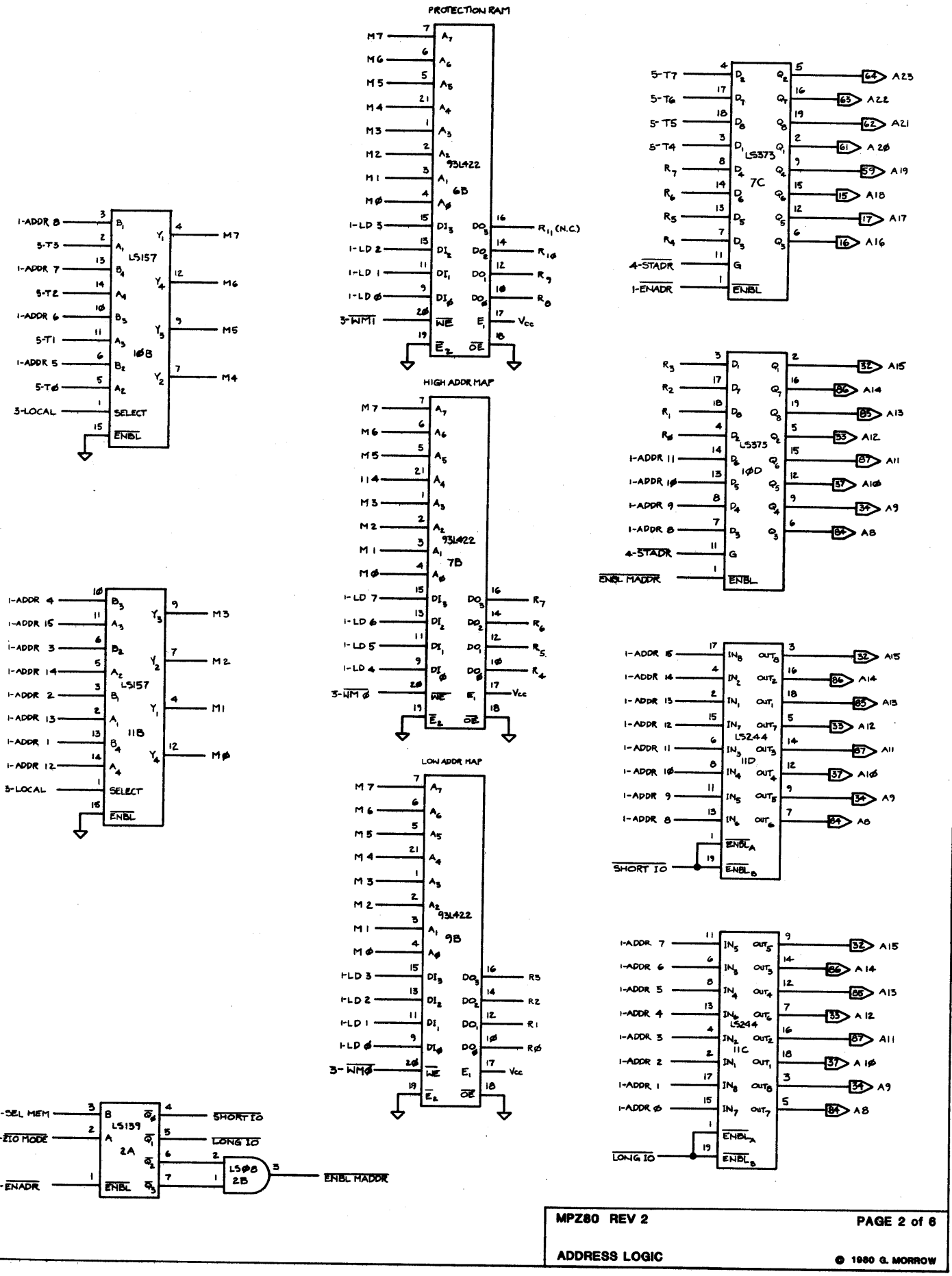
MPZ80 Block Diagram, rev 2



MPZ80 Board Component Layout

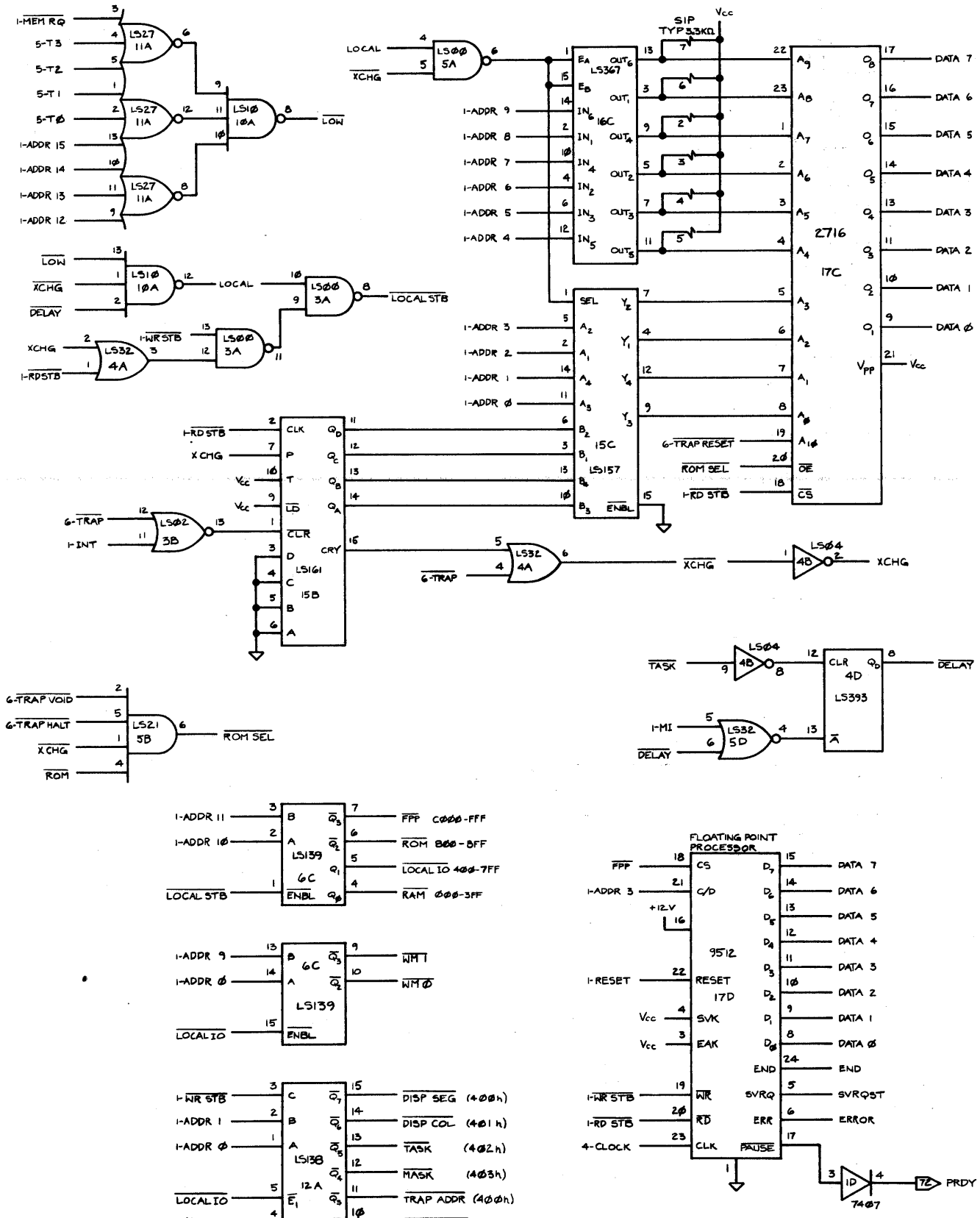


MPZ80 REV 2	PAGE 1 of 6
CPU, BUFFERS & CLOCK	© 1980 G. MORROW

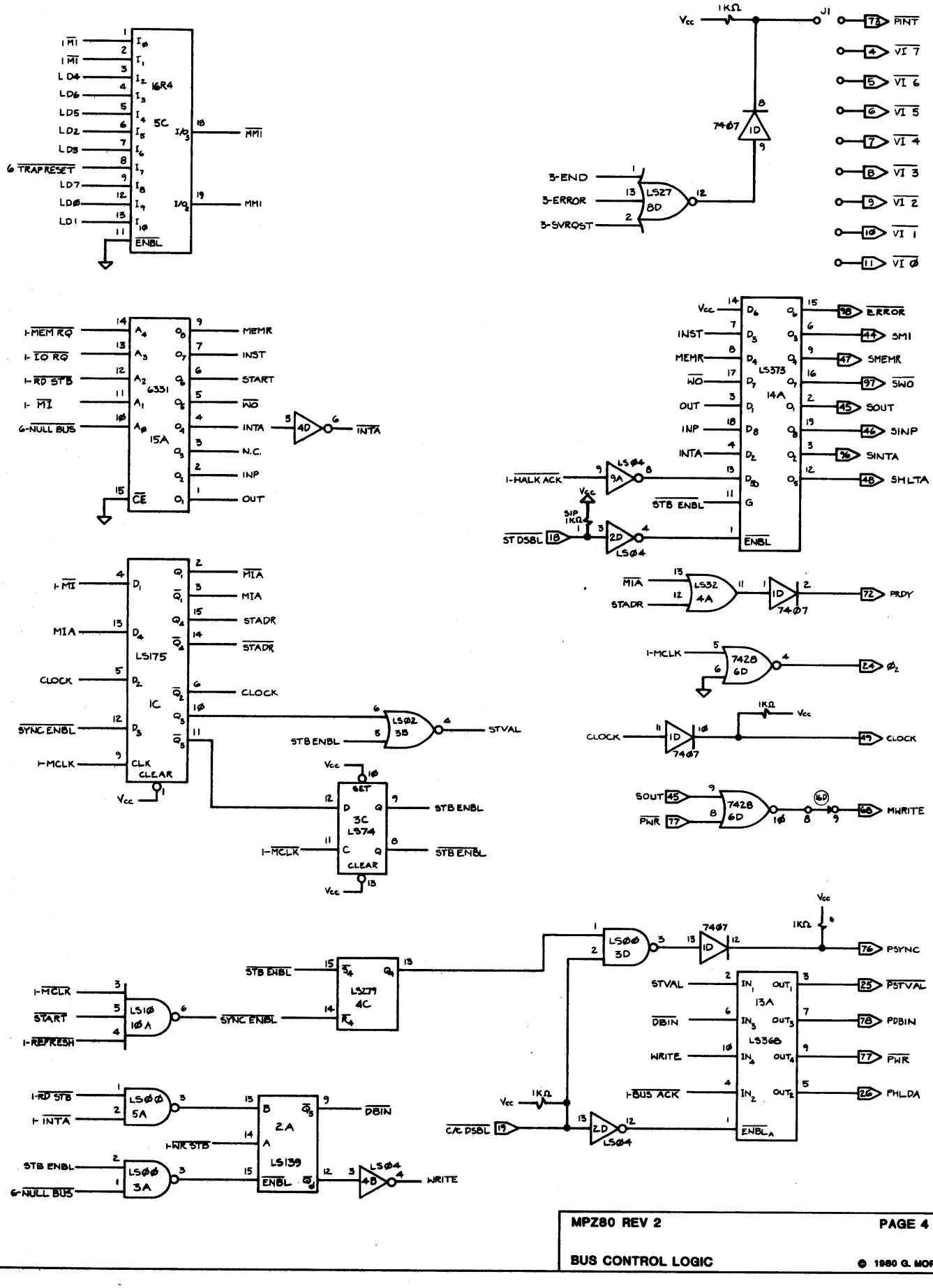


MPZ80 REV 2 **PAGE 2 of 6**

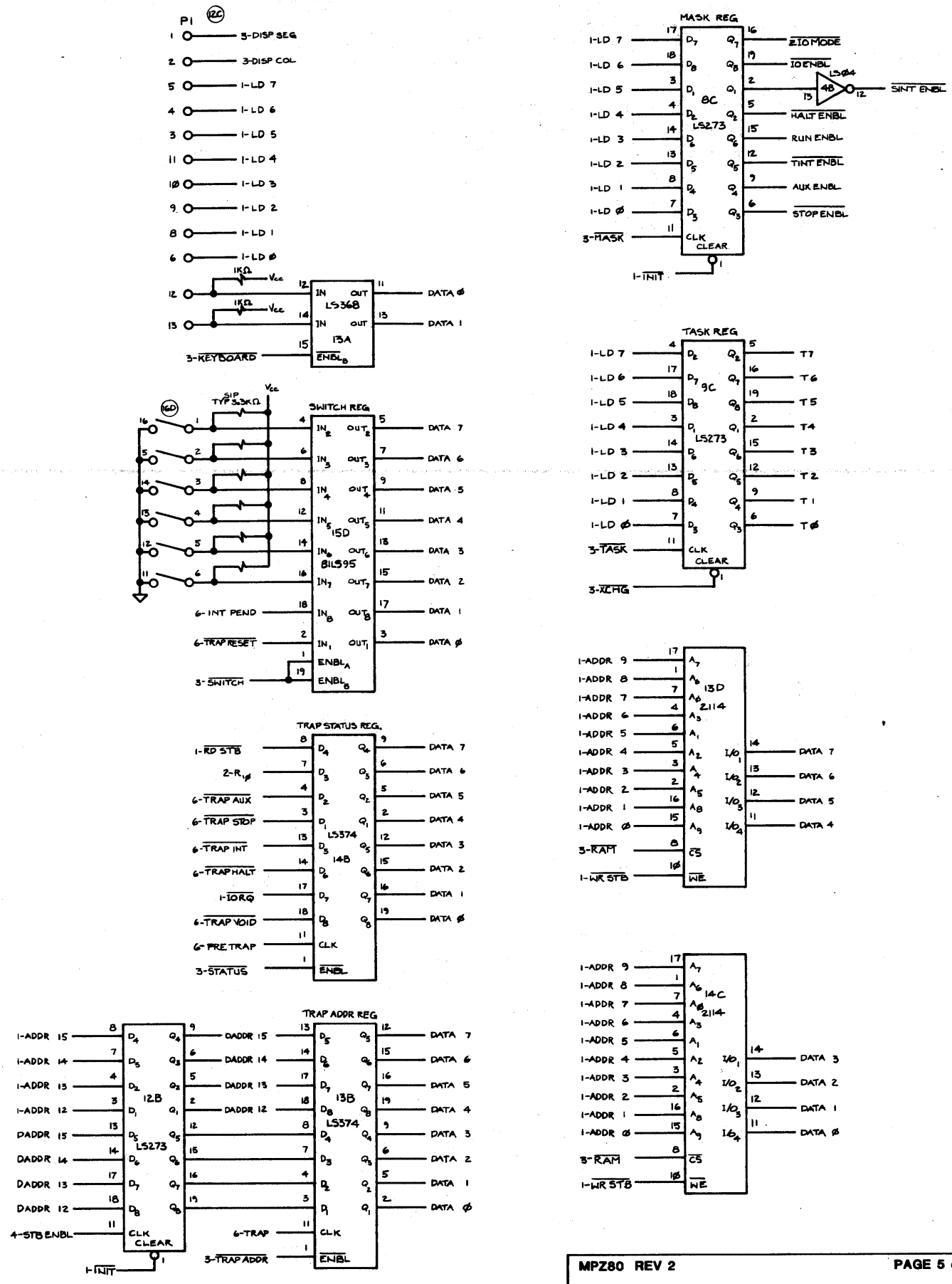
ADDRESS LOGIC



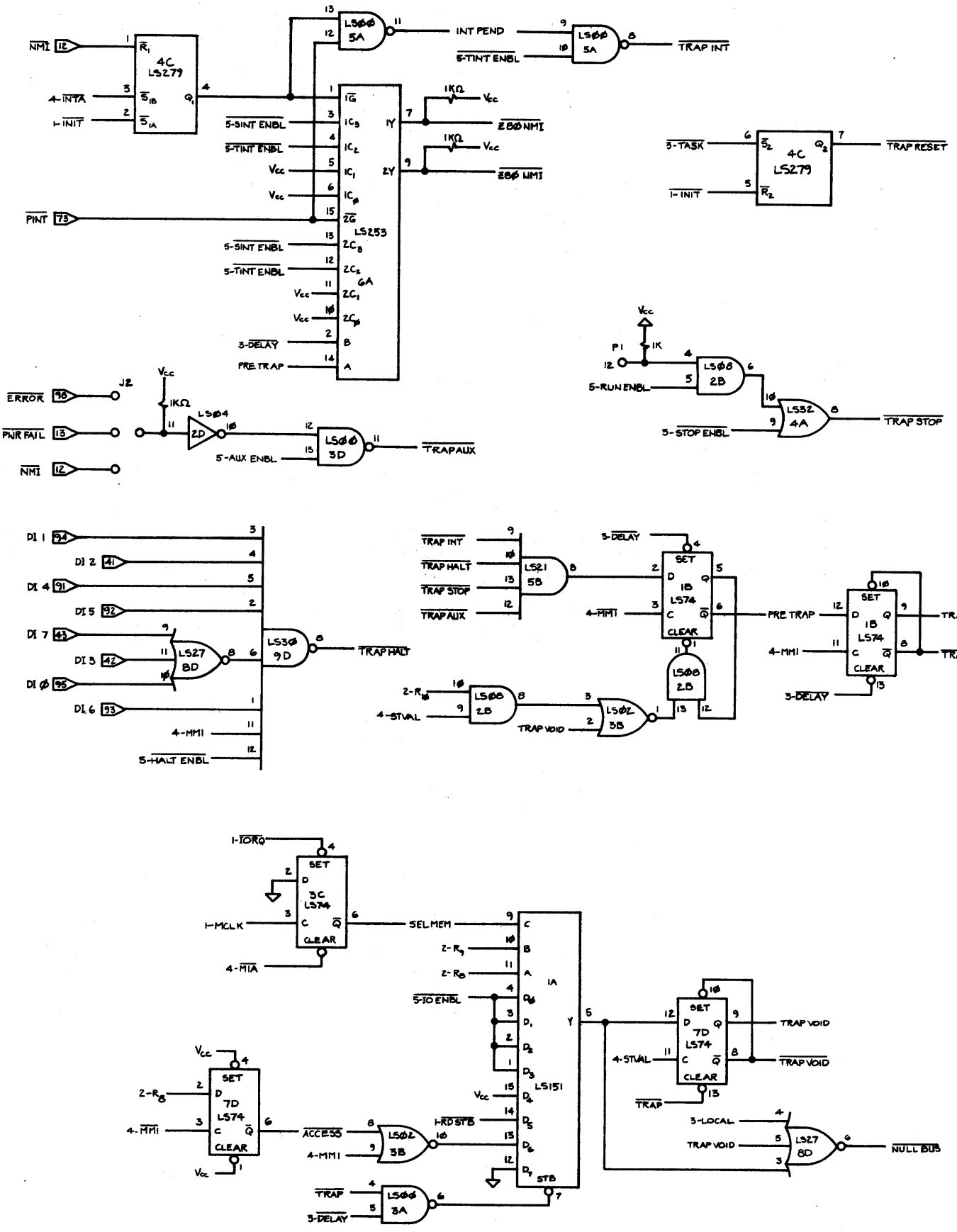
LOCAL STROBES, ROM & FPP



BUS CONTROL LOGIC



LOCAL REGISTERS & RAM



MPZ80 REV 2	PAGE 6 of 6
TRAP LOGIC	