5221 CENTRAL AVENUE
RICHMOND, CA 94804
(415) 524-2101

MORROW DESIGNS

DECISION I

MICROCOMPUTER

PRELIMINARY

Product Reference Manual

and Performance Specifications

DECISION I MICROCOMPUTER SYSTEM

VERSATILE AND COST EFFECTIVE.- THE DECISION I WAS DESIGNED TO BE THE MOST VERSATILE AND COST EFFECTIVE MULTI-USER, MULTI-TASKING MICROCOMPUTER AVAILABLE TODAY. THE DECISION I IS IDEAL FOR BOTH SMALLER BUSINESS AND WORD PROCESSING APPLICATIONS. THE DECISION I'S "UNIVERSAL" CHARACTER PROVIDES THE SYSTEMS DEV-ELOPER WITH A SINGLE HARDWARE PACKAGE THAT CAN ACCOMMODATE THE WIDEST VARIETY OF NEEDS. FROM SINGLE USER, DEDICATED APPLICATIONS TO MULTI-USER, MULTI-APPLICATION ENVIRONMENTS. PLUS, MODULAR EXPANDABILITY WHICH PROVIDES MAXIMUM UPWARD COMPATABILITY.

BIG SYSTEM FEATURES IN MICROCOMPUTER.- THE DECISION I CPU FEATURES A 4-TO-6 MEGAHERTZ Z8ØA, SOPHISTICATED MEMORY MANAGEMENT HARDWARE AND A FLOATING POINT PROCESSOR. THE SOFTWARE FEATURES uNIX-FUNCTIONALLY IDENTICAL TO BELL LAB'S UNIX.* PLUS CP/M,** A HOST OF LANGUAGES (INCLUDING C), BUSINESS APPLI-CATIONS AND WORD PROCESSING PACKAGES.

A SYSTEMS APPROACH.- MEMORY MANAGEMENT HARDWARE INCLUDES A MEMORY MAP THAT SUPPORTS UP TO 16 TASKS WITHOUT SWAPPING (MORE WITH SWAPPING). EACH TASK ENJOYS COMPLETE MEMORY PROTECTION AND AUTOMATIC MEMORY ALLOCATION. ONE OF THE TASKS MAY BE DELEGATED AS THE SUPERVISOR TO HANDLE PRIVILEGED SYSTEM FUNCTIONS THAT HAVE BEEN FORBIDDEN TO ORDINARY TASKS. SUCH FUNCTIONS (I/O CALLS, UNAUTHORIZED MEMORY ACCESS, ETC.), WILL TRAP TOO THE SUPER-VISOR FOR PROPER HANDLING. A SWITCH ARRAY ON THE CPU BOARD ALLOWS TURNKEY CONFIGURATION OF THE SYSTEM IF THE SOFTWARE DOES NOT REQUIRE THE FULL RANGE OF SUPERVISOR FUNCTIONS.

AN UNMATCHED SOFTWARE BASE.- THE DECISION I'S uNIX OPERATING SYSTEM SUPPORTS UNIX SYSTEM CALLS IN A MANNER THAT IS SOURCE COMPATIBLE WITH UNIX. THUS, UNIX PROGRAMS WILL COMPILE DIRECTLY AND UNIX DOCUMENTATION IS ALMOST TOTALLY

*

APPLICABLE. OUR CP/M HAS ALSO BEEN CONFIGURED TO RUN UNDER uNIX AND TO COMMUNICATE WITH BOTH CP/M AND UNIX-STANDARD MEDIA FOR MAXIMUM PORTABILITY. MORROW DESIGNS FEELS THAT EXISTING CP/M AND UNIX SOFTWARE, TOGETHER WITH THE GROWING REALIZATION OF THE POWER OF UNIX, GIVES THE DECISION I A SOFTWARE BASE UNMATCHED IN THE DECISION I'S PRICE/PERFORMANCE ARENA.

FEATURES:

1. uNIX (UNIX IDENTICAL)
2. CP/M
3. 32K TO 1 MEGABYTE OF HARDWARE MANAGED MEMORY
4. FLOATING POINT PROCESSOR (OPTIONAL)
5. SUPERVISOR CONTROL IN HARDWARE AND SOFTWARE
6. 64K ADDRESSES SPACE FOR EACH PROCESS
7. Z80A MICROPROCESSOR
8. 200K TO 104 MEGABYTES OF DISK MEMORY
9. MODULAR EXPANDABLITY
10. 3 SERIAL PORTS AND 2 PARALLEL PORTS MINIMUM

*UNIX is a trademark of Bell Labs.
**CP/M is a trademark of Digital Research .
*

Technical Manual

Decision CPU

Introduction

The Morrow Designs Decision CPU circuit board is an S-100 bus single board computer conforming to the proposed IEEE 696 standards for interface. The CPU features:

* 4 Mhz Z80 microprocessor chip (6 Mhz optional)
* Hardware floating point (9512 floating point processor)
* Full 23 bits of address space allowing access to over 8 megabytes of physical memory
* On board memory consisting of 2K bytes of EPROM and 1 K bytes of RAM memory
* Sophisticated hardware memory management circuitry which is dynamically alterable by the operating system.
* Sophisticated hardware trap mechanisms allowing the operating system to control user operations and access

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

1. Bus Control Logic - all logic necessary to make the Z80 processor conform to the S-100 standards for interface. This circuitry provides the addresses, data and control signals required by external memory and I/O devices.

*

2. Memory Mapping Logic - this block consists of all the logic required to provide the 23 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 segments (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 occurance of any of the following conditions:

a) Halt	- user has attemped to execute a halt instruction
b) Interrupt	- interrupt has occured during a users program.
c) Reset	- the reset button has been pushed.
d) Illegal Memory	- the user has tried to read or write memory which has yet to be allocated to him.
e) I/O	- user has attempted to execute an I/O operation
d) Stop	- front panel stop has been executed
f) Auxiliary	- user defined trap. On systems with 9512 option this can be connected to the 9512 status line ERROR. The board may also be strapped with this line tied to the S-1ØØ NMI, PWRFAIL or ERROR lines.

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 interupted 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. CPM), disabled completely.

4. Local Logic - logic which provides all the onboard clocks and strobes for devices on the CPU card itself.

*

Decision Memory Management Hardware

For the purposes of this section, all memory size references will be listed as decimal unless otherwise stated.

The Decision CPU can access a full 8 megabytes of address space. This memory is devided into sections referred to as tasks and segments. Segments are the smallest increments a memory may be devided into and are 4 K 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 64 K bytes (this limitation is imposed by the address space of the Z8Ø processor).

The total memory space is 8 M bytes but 16 tasks of 64K each only maps 512 K bytes of this space. The other 4 bits of the address space (A2Ø - A23) are selectable by the operating system as a bank select scheme. Therefore any of the 16 tasks may be situated anywhere in the 8 M byte range of the CPU card.

Of the 16 tasks available, the Z8Ø can execute only one at a time and therefore 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 (from now on this will be 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 (4 K 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 5ØØ (hex) to 7FF (hex) in task Ø. The
*

size of each memory segment is 4 K bytes and each segment has its own set of protection attributes. A segment may be configured as:

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 will be aborted and execution of the task running will stop with the supervisor now executing. This is referred to as a 'Trap' but more on them 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 added feature of the Decision CPU memory management hardware is that any task may share any number (up to 16) of memory segments with any other tasks. In the event the operating system has data to swap with a task, all that need be done is write into the mapping rams a duplicate map 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 allows 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 does not cause a trap to occur but does allow the operating system to monitor user memory references. In the instance where a task has outgrown its memory allocation and requires more memory, the operating system can examine this bit and determine whether or not to allow the tasks space to grow. This is strictly a software protection but is written into the protection attribute ram along with the other protection attributes. The advantage of this attribute is that the supervisor can allocate a full 64 K bytes to a task of which perhaps 16 K bytes was designated as 'No Access' protection. If
*

the task outgrows this area, and the extra protection bit is high, the supervisor will update the task's memory map to allow it access to more memory. If this bit were low, the task would be refused access to more memory and the supervisor would send an error message to that user. This bit will be referred to as the 'Grow' bit.

The versatility of the Decision CPU lies in its ability to allocate memory dynamically over the 8 M byte range and to trap on the occurance of an illegal event occuring within a particular task -- a good deal more sophisticated than the typical S-1ØØ CPU board. This does not however, prohibit the use of the CPU board as a standard S-1ØØ bus master with a 64K memory space. The onboard EPROM is responsible for configuring the CPU for the user's application.

In the event a user wished the board to run a total of only one task, (e.g. the CPM 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 64 K bytes of unprotected memory to task 1 (remember that Task Ø has the low 4 K bytes taken up as the Decision CPU features and therefore would not be able to run CPM). The switches on the CPU board will determine the power on jump location for the operating system or bootstrap loader. The CPU may jump to any location on a 2K boundry. 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.*

Decision CPU Trap Hardware

The power of the Decision CPU lies in its ability to trap on the occurance 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 6 levels of trapping available with the Decision CPU. They are configured in such a way as to allow the operating system to restrict the execution of certain operations within a users program and may or may not be utilized. The low level traps consist of:

Trap Reset	-	Indicates the reset button has been pushed or the CPU has just powerd 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 three traps are considered to be high level traps in that they actually abort the occurrance of the attempted operation. These consist of:

Trap Halt	-	A task has attempted to execute a Z8Ø Halt operation and halts are not allowed in that task.
Trap Int	-	An interupt was generated during the execution of a task which was not allowed to honor interupt requests.
Trap Void	-	A task attempted to execute an I/O instruction or an illegal memory access has occurred.

The last trap, Trap Void is actually a compilation of several types of trap condition. By reading the Trap Status Port (4Ø3H) in task Ø, segment Ø the operating system can determine precisely what type of trap occurred. Thus, Trap Void type traps may be further broken down into the following 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 will begin execution and will decide whether or not 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 code 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 a halt. In this case, the trap logic must prevent the halt instruction from reaching the Z8Ø chip. To accomplish this, the data intput driver to the CPU data lines is turned off whenever a halt op code (76h) is detected during an M1 cycle. This condition also enables the EPROM on the CPU board to gate a nop (ØØh) into the Z8Ø chip. Therefore location ØBFOh in the EPROM must contain a NOP since this is the location executed whenever a trap occurs.
*

To re-iterate, 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 interupts. In addition, this register enables the front panel stop (this can be deactivated as well) switch, the auxiallary enable bit and will determine if the port addresses during the execution of I/O will be in long or short mode (long mode putes the addresses on A8 - A15 and AØ - A7, while short mode puts them on the low 8 address lines only).

Upon the occurance of a trap, since the EPROM code has successfuly 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 instuction which caused the trap to occur. Only this time, the execution will continue without any traps. Another example might be when a task is interupted by external interupt logic (for instance when a disk file has been successfuly loaded) but this tasks was not allowed to acknowledge interupts. In this case the task might have been waiting for the contents of the file for further execution. The condition will cause a trap to the supervisor which will handle the interupt, transfer the data to the task and then return 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-1ØØ type CPU board without any of these features. In this case, the onboard EPROM will contain code to inhibit all the trapping mechanisms from activating which will be executed before the CPU jumps to the desired task. This would be typical in a single type enviorment such as CPM which does not support the trap features. By simply writing a 2Bh into the Mask register, all traps are inhibited and the CPU board will emulate other S-1ØØ Z8Ø CPU bus masters.
*