COLEX

COLEX DEBUG 68000

This monitor contains routines to modify memory, I/O ports, or any 68000 register. Also included are a disk saver/loader and a hexadecimal loader for downloading programs through the serial I/O channel. A breakpoint may be placed in ROM or RAM to stop program execution and either ROM or RAM may be single stepped. The monitor program is fully relocatable and may be moved by a simple command to any even boundry address.

FEATURES:

*   User callable traps for I/O

*   13 commands as follows:

B   Breakpoint in ROM or RAM

C   Copy monitor to other address and execute

D   Display memory in tabular form

E   Execute a memory address

L   Load an Intel hexadecimal format program

M   Modify a memory word

O   Offset the modify and display commands

P   Ports may be modified using byte addressing

R   Registers may be modified

T   Trace a program in single step mode

X   Examine all registers

>   Put a program to floppy disc

<   Get a program from floppy disc

INITIALIZATION:

After a reset the EPROM instructs the processor to copy its contents into ram at location 0000 to 07FF. The EPROM is then deactivated by an I/O read from the Z80 8-bit port 00 (68000 physical address E00000). Next the program copies itself to location 001000 and transfers control to the copied program. The bottom 1024 bytes of RAM from 000000 to 0003FF are filled with an error trap address. Selected TRAP instruction vectors are loaded with addresses of I/O routines, and the TRACE TRAP vector is initialized. After initialization the PROGRAM announces its name as COLEX DEBUG 68000 V1.0 at address 001000' and the $ prompt is displayed.

*** COMMANDS ***

B BREAKPOINT

A breakpoint is a command to stop execution of a program at a specified address. The breakpoint command turns on the trace command but the trace command does not display any registers until the breakpoint address is reached. Breakpoints may be in RAM or ROM/EPROM.If no breakpoint is set 'Breakpoint is not set ' will be displayed.

EXAMPLE -

```
B 1234 (Return)
```

Program trace will be turned on, no registers will be displayed until the address 001234 is executed. To return to the '$' prompt type a period '.'. Tracing may be continued with the 'T' command or registers may be examined/modified.

***

C COPY

The copy command allows relocation of the Monitor. After typing 'C', the user should enter the desired address for the program to reside at. The Monitor will move to that address and will execute at that address, TRAP vectors will be updated, the program will sign on and the new load address will be displayed.The next '$' prompt will come from the copied and executed program. The only restriction is that the new address must be at least 2K Bytes away from the old address. Two copies may be required to get the monitor to the desired address.

EXAMPLE -

```
C 008000 (Return)
```

Will relocate the monitor to location 008000 and execute it.

***

D DISPLAY

The display command requires two numbers as input, a starting address, a comma or space, and an ending address. This block of memory will then be displayed in tabular form. If a logical offset is activated (see OFFSET command), then the physical and logical addresses will be displayed. The logical address will be followed by a single (') mark. A period '.' may be entered to stop the display.

EXAMPLE -

D 001000,0017FF (Return)

Will display the addresses 1000 to 17FF (the monitor).

***

E EXECUTE

The execute command allows the programmer to transfer control to a program. After entering the desired address, control will be transferred to the instruction at the entered address. If no address is entered, then the saved program counter will be used as a starting address. To execute a program in user mode , clear the supervisor mode bit in the SR register with the R command (see R command).

EXAMPLE -

E 400 (Return)

The Processor will restore the registers and execute the program found at location 000400. Normally the program will be executed in SUPERVISOR mode unless the bit is cleared in the status register.

***

L LOAD

The L command allows simple serial I/O loading of programs from a host computer. The format is INTEL hexadecimal format. Upto 65K Bytes of RAM may be down-loaded. The format is ASCII and allows error checking of the transfer through a simple check-sum technique.The syntax for the load command is as follows:

: A colon should be the first character on a line. Any other characters upto the ':' will be ignored.

LENGTH The length of the line (in hex) expressed in two bytes.

ADDRESS The desired load address for the load is next. It is a four digit hexadecimal number.

RECORD TYPE This is either 00 or 01. The 00 record type is for data. The 01 record type is END-OF-FILE. If END-OF-FILE is read , the address field will be examined and if zero control will be returned to the monitor. If non-zero, then control will be returned to the load address.

DATA This is a field of hex characters of data to be loaded.

CHECKSUM The checksum is a number, which when added to the preceeding bytes except the ':' results in zero. If a wrong checksum is encountered, then the monitor will type 'Load error @ XXXXXX' where the Xs represent the address of the improperly loaded data block.

EXAMPLE -

L (Return)

:020500001122C6 (Return)

Will load location 500 with 11, and 501 with 22

***

M MODIFY

The modify command will allow the user to examine and modify any memory address. The user should enter the 24-Bit address of the memory location desired upto six hexadecimal digits. The processor will respond with the logical address if the logical offset is activated, followed by the physical memory address and the data from that address expressed as four hex digits. If no modification of the data is required press RETURN to examine the next address, or '.' to abort the command and return to the '$' prompt. If the address is to be modified, enter the new data as one to four hex digits and press SPACE. If the data required is 0000 then just press SPACE. An ' ' may be entered instead of data to back up to the previous address.

EXAMPLE -

M 400 (Return)

The processor will respond with:
000400 XXXX Where XXXX is data found in location 400.

***

O OFFSET

The offset command is used as an aid in debugging users programs that are relocatable. If the programs load address is unknown or it does not agree with the listing, then debugging would be difficult. The offset command allows the programmer to enter the starting address of a routine and the Display, Trace and Modify commands will show both the logical and physical address of the program. This way the listing will match the program counter.

EXAMPLE -

O 400 (Return)

Update the offset program counter for a program loaded at address 400.

P PORT

The port command will allow the user to examine any I/O port. The user should enter the 8-bit logical address of the port desired as two hexadecimal digits. The processor will respond with the physical port address (E000XX) and the port data expressed as two hex digits If no modification of the ports is desired, press RETURN to examine the next port, or '.' to abort the command and return to the '$' prompt. If the port is to be modified, enter the one or two hex digits of the desired output data and press SPACE. If the new data is to be 00 then just press SPACE. An '²' may be entered instead of data to back up to the previous port.

EXAMPLE -

P 40 (Return)

The processor will respond with:
40 XX Where XX is the port data found in port 40

***

R REGISTER

The R coomand requires a two-digit Register name as an operand. Any 68000 register name may be used. If the register is to be modified enter the new data after the register name and data are typed. If no modification is desired, type a '.' If a program is to be executed in SUPERVISOR mode, the SR register should equal 2XXX indicating the SUPERVISOR bit is set. If the 2XXX bit is cleared, programs may be executed in USER mode.

EXAMPLE -

R PC
PC=00030000 at this point a new program counter value may be entered

R US
US=00007F00 at this point a new stack pointer may be entered.

R D0
D0=12345678 ABCDEF(Return) at this point the user has changed D0 to ABCDEF.

R SR
SR=2700 0 (Return) the user wishes to execute a program in USER mode.

***

T TRACE

The trace command allows single step operation of any program. If the T command is followed by an address, then the program execution will begin at that address. If the address is not entered or is zero then the current saved program counter will be used. The trace operation may be in RAM or ROM/EPROM. After the registers have been displayed a '.' will return the user to the '$' prompt, or a (Return) will display one more instruction.If a breakpoint is set, tracing will not display the registers until the breakpoint is reached, or CONTROL-D is typed on the console. Tracing may be in USER or SUPERVISOR mode.

EXAMPLE-

```
T 4000 (Return)
```

The 68000 Registers will be restored, program execution will begin at 004000, the instruction there will be executed and the resulting 68000 register set will be displayed on the console device.

***

X EXAMINE

The x command will display all registers. These registers are actual 68000 that are saved into memory locations by the Trace trap and are unsaved by the Trace and Execute commands. The user status register and the user offset program counter will be displayed on the first line for easy readability.

EXAMPLE-

```
X (Return)
SR=    2700 PC'=000410'
PC=00000400 D0=12345678 D1=ABCDEF01 D2=AAAAAAAA D3= etc. etc.
D5=22222222 D6=33333333 D7=44444444 A0=12345678 etc.etc
A3=etc.etc
```

***

> WRITE

This command will allow any track and sector of the floppy disc to be written from memory. The routine will respond with 'ENTER TRACK,SECTOR,LENGTH'. The user should enter the track number (in hex), the sector number (in hex), length in sectors (in hex). When the end of the current track is reached a seek to the next higher track will be performed. The write command will then write to the selected track and sector from memory starting at location 000400.

EXAMPLE-

```
$> (Return)
ENTER TRACK,SECTOR,LENGTH  4,3,3 (Return)
```

Will write memory locations from 000400 onto track 4, sectors 3, 5 and 7.

***

< READ

This command allows any track and sector of the floppy disc to be written into memory. The routine will respond with 'ENTER TRACK,SECTOR,LENGTH'. The user should enter the track number (in hex), sector number (in hex), length in sectors. When the end of the current track is reached a seek to the next higher track will be performed. The command will then read the selected track and sector into memory starting at location 000400. If the disc is not ready, then the '$' prompt is returned. A 2X interleave is used.

EXAMPLE-

```
< (Return)
ENTER TRACK,SECTOR,LENGTH 0,1,7 (Return)
```

Will read track 0, sectors 1;3,5,7,9,11,13 into location 000400 onwards.

***

TRAP Instruction Vectors

By executing a TRAP #X Instruction, any of 16 possible SUPERVISOR-mode routines in the monitor can be executed. These routines are available to USER or SUPERVISOR programs. The monitor uses 12 of these TRAP vectors for internal use. The addresses of the routines are found in memory locations 80 through BF hexadecimal.

TRAP processing is the only way for programs in USER mode to access SUPERVISOR mode routines to access the Z80 I/O ports. If any attempt is made in USER mode to acess the addresses from E00000 to FFFFFF, then bus error exception processing will start, SUPERVISOR mode will be entered, the program counter and status register will be placed on the SUPERVISOR stack (A7), the value found in the BUS ERROR vector at location 8 will be loaded into the program counter, and the monitor will announce an error message. If the memory hardware is activated by the bus signal MEMEX, the trap vectors will be found in the SUPERVISOR address space.

The 12 TRAPS used by the monitor are:

TRAP #0	RE-BOOT	TRAP #6	OUTPUT WORD
TRAP #1	RE-START	TRAP #7	OUTPUT BYTE
TRAP #2	GET NUMBER	TRAP #8	READ CHARACTER
TRAP #3	PRINT DATA	TRAP #9	WRITE CHARACTER
TRAP #4	OUTPUT LONG WORD	TRAP #A	PRINT CRLF
TRAP #5	OUTPUT ADDRESS	TRAP #B	PRINT SPACE
TRAP #C	Not Used	TRAP #D	Not Used
TRAP #E	Not Used	TRAP #F	Not Used

Some routines require certain registers to be set up before the TRAP instruction is executed. For example, if a character is to be written to the console device, the character will be sent to the TRAP routine in data register D0. Some routines return with values in registers, and other routines do not return to the caller at all. Consult the individual listing for each vectors usage of registers.

NOTE: The monitor is relocatable to any even address in memory via the 'C' command. If the 'C' command is used to relocate the monitor the TRAP instruction vectors will be updated to the new monitor load address. If a user program requires the use of already used trap vectors, the user program should save the trap vector and restore it after use.

TRAP VECTOR DEFINITIONS

TRAP #0 Re-Boot. This trap will not return to the caller. The monitor will restart as if RESET was activated, The screen may be cleared, the program should announce the name, Revision level, date of origin, and load address of the monitor.

TRAP #1 Re-Start. This will transfer control to the monitor The Supervisor stack will be reset, and the monitor will be reset, and the monitor will prompt with a carriage return, line feed, and a '$'.This is the normal entry point for user routines back to the monitor This trap will not return to the caller. By replacing the RTS (Return From Subroutine) instruction at location 17FE and 17FF with any legal instruction, and examining register D0, additional commands may be added. The monitor will always BSR (Branch to Subroutine) to this location before command string processing is terminated. If a user wishes to expand upon the commands available in the monitor this trap may be used to end command processing. If a legal user added command is in register D0, the user should process the command and return to the monitor with TRAP #1. If an illegal command was entered in register D0, the user command should execute an RTS instruction. A '?' will then be printed.

TRAP #2 Get Number. The TRAP #2 may be used to read a hexadecimal number from 00000000 to FFFFFFFF. Any non-hexadecimal character entered will force a return to the caller. The last character entered will be in register D0. The entered number will be returned in register D5. A blank line (RETURN only) will exit with a RETURN in D0, and 00000000 in D5.

TRAP #3 Print Data. The TRAP#3 routine requires that register A1 point to a text string of ASCII CHARACTERS. THE END OF THE STRING MUST BE MARKED BY A CONTROL-D OR 04. The string will be printed to the terminal using TRAP#9. Upon exit, register A1 will point to the byte following the 04 character. Register D0 is destroyed.

TRAP #4 Output Long Word. This routine prints using TRAP #9 the register D0 as a string of 8 ASCII hexadecimal digits. Register D0 is destroyed.

TRAP #5 Output Address. This routine prints using TRAP #9 the register D0 as a string of 6 ASCII hexadecimal digits. Register D0 is destroyed.

TRAP #6 Output Word. This routine prints using TRAP #9 the register D0 as a string of 4 ASCII hexadecimal digits. Register D0 is destroyed.

TRAP #7 Output Byte. This routine prints using TRAP #9 the register D0 as a string of 2 ASCII hexadecimal digits. Register D0 is destroyed.

TRAP #8 Read Character. This routine uses Z80 ports 03 for data and 01 for control. A Z80 SIO-type chip is expected at those addresses. The SIO will be setup when the monitor is activated. No modem control lines are needed. Parity is ignored. There are two modes of returning control to the calling routine in Read Character, called Normal Return and Immediate Return.

Normal Return: This routine requires that bit 0 of register D4 be cleared upon entry. When a character is entered from the Keyboard, the routine will return with the character in register D0. Bit 0 of D4 will remain cleared.

Immediate Return: If bit 0 of register D4 is set upon entry, the keyboard will be tested for data. If no data is ready, the routine will return to the caller with bit 0 of register D4 still set. If a character is available, then bit 0 in D4 will be cleared, and register D0 will contain the character.

TRAP #9 Write Character. This routine will output the ASCII byte found in register D0. No immediate return mode is available. No registers are affected.

TRAP #A CRLF. This routine uses TRAP #9 to print a carriage return and line feed. Register D0 is destroyed.

TRAP #B SPACE. This routine will use TRAP #9 to print a space. Register D0 is destroyed.

SAMPLE DEBUG SESSION

Strapping information for CPU, DRAM, PSIO Boards

```
CPU 68000      J2 3-4
DRAM           J1 All closed
               J2 Open
               J3 1-2, 3-4, closed
               J4 3-4, closed
               J5 open
             N.B. Remove wire link from J6 (long one)

PSIO           J2 all closed
               J3 3-4, 21-22 closed
               J4 1-2, 4-7, 5-9
             N.B. Ensure IOXP is grounded.
```

Enter the following code with the 'M' command:

```
M 4000 (Return)
004000 XXXX 4280 (Press SPACE)      (4280 = clear D0)
004002 XXXX 5280 (Press SPACE)      (5280 = add 1 to D0)
004004 XXXX 61FC (Press SPACE)      (61FC = branch back 4 bytes)
004006 XXXX      (Press '.')
```

Now use 'T' command

```
T 4000 (Press Return)
SR=A704
PC=4002 D0=00000000 D1=XXXXXXXX etc. etc.          (press RETURN)

SR=A700
PC=4004 D0=00000001 D1=XXXXXXXX etc. etc.          (press RETURN)

SR=A700
PC=4002 D0=00000002 D1=XXXXXXXX etc. etc.   (Press RETURN or '.')
```

***