68000

Debugger

Hawthorne Technology

68000

DEBUGGER

Hawthorne Technology
8836 S. E. Stark
Portland, OR 97216

(503) 254-2005

* * * PRODUCT DISCLAIMER * * *

This software and manual are sold 'as is' and without warranties as to performance or merchantablility. These programs are sold without any express or implied warranties. No warranty of fitness for a particular purpose is offered. The user must assume the entire risk of using the programs and is advised to test the programs thoroughly before relying on them.

Any liability of seller or manufacturer will be limited exclusively to product replacement or refund of the purchase price.

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

68000 DEBUGGER

TABLE OF CONTENTS

DEBUG USER GUIDE

The purpose of the HTDBUG program is to make it easier to create and debug programs running under K-OS ONE. There are commands that allow the user to examine and replace bytes, words, or long words, in memory. A single line assembler is included to make it possible to patch programs. A disassembler is included to make it possible to look at sections of code in memory.

The user can also examine and change individual registers. A program can be run and traced. Break points can be set. This provides a level of debugging normally limited to hardware in-circuit emulators.

All of the commands that work with a disk file or a peripheral device may be directed to or from any device in the system.

Each command is discussed in detail in a separate section of this manual. The commands are grouped together because the functions and syntax are similar for all the commands in a group.

COMMAND SUMMARY

CODE	SYNTAX	DESCRIPTION
MX	MX <ADR>,<ADR>	MEMORY DISPLAY HEX
ML	ML <ADR>	MEMORY EXAMINE/CHANGE LONG
MW	MW <ADR>	MEMORY EXAMINE/CHANGE WORD
MB	MB <ADR>	MEMORY EXAMINE/CHANGE BYTE
MO	MO <ADR>	MEMORY EXAM/CHG ODD BYTES
ME	ME <ADR>	MEMORY EXAM/CHG EVEN BYTES
MA	MA <ADR>	MEMORY EXAM/CHG ASSEMBLY
MD	MD <ADR>,<COUNT>	MEMORY DISASSEMBLY
BF	BF<ADR1>,<ADR2>,<WORD>	BLOCK FILL
BM	BM<ADR1>,<ADR2>,<ADR3>	BLOCK MOVE
BT	BT<ADR1>,<ADR2>	BLOCK MEMORY TEST
BC	BC<ADR1>,<ADR2>,<ADR3>	BLOCK COMPARE
A0-A7	<REG>,<EXPRESS>	DISPLAY/SET ADDRESS REG
D0-D7	<REG>,<EXPRESS>	DISPLAY/SET DATA REG
PC	PC,<EXPRESS>	DISPLAY/SET PC
SR	SR,<EXPRESS>	DISPLAY/SET SR
US	US,<EXPRESS>	DISPLAY/SET USP
DR	DR	DISPLAY ALL REGISTERS
BR	BR<ADR>,<ADR>...	SET BREAK POINT
GO	GO<ADR>	EXECUTE PROGRAM
TR	TR<ADR>	TRACE PROGRAM
DS	DS<PATH>,<ADR>,<ADR>,<ADR>	DUMP MEMORY-S REC.
DH	DH<PATH>,<ADR>,<ADR>,<ADR>	DUMP MEMORY-INTEL H
DB	DB<PATH>,<ADR>,<ADR>	DUMP MEMORY-BINARY
LS	LS<PATH>,<OFFSET>	LOAD S RECORDS
LH	LH<PATH>,<OFFSET>	LOAD INTEL HEX
LB	LB<PATH>,<OFFSET>	LOAD BINARY
VS	VS<PATH>,<OFFSET>	VERIFY S RECORDS
VH	VH<PATH>,<OFFSET>	VERIFY HEX
VB	VB<PATH>,<OFFSET>	VERIFY BINARY
QT	QT	QUIT DEBUGGER

DEBUG USER GUIDE ---- MEMORY EXAMINE AND CHANGE

These commands are provided for examining and changing the memory of the system while a program is being debugged. Different commands are used because there are different ways to look at memory and different reasons why you would want to.

COMMAND SUMMARY

CODE	SYNTAX	DESCRIPTION
MX	MX <ADR>,<ADR>	MEMORY DISPLAY HEX
ML	ML <ADR>	MEMORY EXAMINE/CHANGE LONG
MW	MW <ADR>	MEMORY EXAMINE/CHANGE WORD
MB	MB <ADR>	MEMORY EXAMINE/CHANGE BYTE
MO	MO <ADR>	MEMORY EXAM/CHG ODD BYTES
ME	ME <ADR>	MEMORY EXAM/CHG EVEN BYTES
MA	MA <ADR>	MEMORY EXAM/CHG ASSEMBLY
MD	MD <ADR>,<COUNT>	MEMORY DISASSEMBLY

MX MX <ADR>,<ADR> MEMORY DISPLAY HEX

This displays the contents of a block in hex and in ASCII characters where a printable character exists. This is the classic memory dump.

ML ML <ADR> MEMORY EXAMINE/CHANGE LONG

This examines and allows a user to change emeroy in long words. The entire long word (32 bits) is changed at one time making it suitable for changing interrupt vectors.

MW MW <ADR> MEMORY EXAMINE/CHANGE WORD

This allows memory to be examined and changed one word at a time.

MB MB <ADR> MEMORY EXAMINE/CHANGE BYTE

This allows all the bytes to be examined and changed.

MO MO <ADR> MEMORY EXAMINE/CHANGE ODD BYTES

This is like the MB command but only works on bytes on odd addresses. This is useful for working with I/O devices that are memory mapped.

ME ME <ADR> MEMORY EXAMINE/CHANGE EVEN BYTES

This is like the MO command but works only on bytes that are on even addresses.

MA MA <ADR> MEMORY EXAMINE/CHANGE ASSEMBLY

This invokes the single line assembler. The contents of the memory location are disassembled and displayed. A single line of assembly code can be entered to replace the line that was displayed. Once the single line assembler is invoked, the following commands should be used:

```
     .      (period) to Quit
     ;      (semi colon) to Skip Word
  <cr>      (carriage return) to Skip Instruction
  TEXT<cr>  to Assemble Text at Current Location
```

MD MD <ADR>,<COUNT> MEMORY DISASSEMBLY

This command is used to disassemble and display a block of memory. Normally 16 lines will be displayed at a time.

COMMAND SUMMARY

CODE	SYNTAX	DESCRIPTION
BF	BF<ADR1>,<ADR2>,<WORD>	BLOCK FILL
BM	BM<ADR1>,<ADR2>,<ADR3>	BLOCK MOVE
BT	BT<ADR1>,<ADR2>	BLOCK MEMORY TEST
BC	BC<ADR1>,<ADR2>,<ADR3>	BLOCK COMPARE

BF BF<ADR1>,<ADR2>,<WORD> BLOCK FILL

This fills the memory between the two addresses specified with a word value. Because the 68000 loads and stores words on even boundaries this command will drop the last bit of the address making it the next lower word.

BM BM<ADR1>,<ADR2>,<ADR3> BLOCK MOVE

This command is used to move the contents of one memory block to another. This is useful for making a copy of the memory to be compared later.

BT BT<ADR1>,<ADR2> BLOCK MEMORY TEST

This command is used to test the memory hardware. The memory tested may have its contents changed by the test. The test will detect many kinds of memory errors and if used on a system that has no known design bugs is very good at detecting bad memory chips or defects in some support circuits. On a new design there are errors that are possible in addressing that this test will not detect.

BC BC<ADR1>,<ADR2>,<ADR3> BLOCK COMPARE

This command is used to compare two blocks of memory. If both blocks are the same then a good message will be given. If the two blocks are not the same then the addresses of the words that are different will be given.

COMMAND SUMMARY

CODE	SYNTAX	DESCRIPTION
A0 - A7	<REG>,<EXPRESS>	DISPLAY/SET ADDRESS REG
D0 - D7	<REG>,<EXPRESS>	DISPLAY/SET DATA REG
PC	PC,<EXPRESS>	DISPLAY/SET PC
SR	SR,<EXPRESS>	DISPLAY/SET SR
SS	SS,<EXPRESS>	DISPLAY/SET SS
US	US,<EXPRESS>	DISPLAY/SET US
DR	DR	DISPLAY ALL REGISTERS

A0 - A7 <REG>,<EXPRESS> DISPLAY/SET ADDRESS REG

These commands are used to examine the value in an address register or set the register to a given value.

D0 - D7 <REG>,<EXPRESS> DISPLAY/SET DATA REG

These commands are used to examine the value in a data register or set the register to a given value.

PC PC,<EXPRESS> DISPLAY/SET PC

This command is used to see the last instruction executed. If no starting address is given when the user program is started this is where execution will begin.

SR SR,<EXPRESS> DISPLAY/SET SR

This is used to set the initial value of the status register for a program to start with.

SS SS,<EXPRESS> DISPLAY/SET SS

This is used to examine or set the value of the system stack pointer.

US US,<EXPRESS> DISPLAY/SET US

This command is used to display or set the user stack pointer. This is the value the stack pointer will have in it when the program is started.

DR DR DISPLAY ALL REGISTERS

This command is used to display all the processor registers at the same time. This is like a snap shot of the state of the machine.

COMMAND SUMMARY

CODE	SYNTAX	DESCRIPTION
BR	BR<ADR>,<ADR>...	SET BREAK POINT
GO	GO<ADR>	EXECUTE PROGRAM
TR	TR<ADR>	TRACE PROGRAM

BR BR<ADR>,<ADR>... SET BREAK POINT

This command is used to set a break point in the code being worked on. The illegal instruction is used to set a static break point. Examples:

```
BR <cr>              Displays Breakpoints
BR <adr> [,<adr>]    Sets Breakpoints,
   (or Clears Breakpoints if previously set.)
```

GO GO<ADR> EXECUTE PROGRAM

This command is used to turn control of the machine over to the user program. The values that are in the user registers or that have been set by the programmer will be loaded into the registers when the program starts. If no start address is given then the value that is in the PC register will be used.

TR TR<ADR> TRACE PROGRAM

This command is like the GO command but it sets the trace bit in the program status first. This causes an exception after each opcode is executed. The debugger will display each code as it is executed and may be stopped at any time to examine the contents of any of the registers.

```
.       (period) to Quit
<cr>    (carriage return) to Trace Next
Any other char. to Dump Registers and Trace Next
```


COMMAND SUMMARY

CODE/SYNTAX	DESCRIPTION
DS<PATH>,<ADR>,<ADR>,<ADR>	DUMP MEMORY-S RECORD
DH<PATH>,<ADR>,<ADR>,<ADR>	DUMP MEMORY-INTEL HEX
DB<PATH>,<ADR>,<ADR>	DUMP MEMORY-BINARY
LS<PATH>,<OFFSET>	LOAD S RECORDS
LH<PATH>,<OFFSET>	LOAD INTEL HEX
LB<PATH>,<OFFSET>	LOAD BINARY
VS<PATH>,<OFFSET>	VERIFY S RECORDS
VH<PATH>,<OFFSET>	VERIFY HEX
VB<PATH>,<OFFSET>	VERIFY BINARY

DS DS<PATH>,<ADR>,<ADR>,<ADR> DUMP MEMORY-S RECORDS

This command dumps the contents of the range of memory specified in standard Motorola S record format. You give it the PATH, beginning memory address, ending memory address, and an offset that is to be added to the record address field.

DH DH<PATH>,<ADR>,<ADR>,<ADR> DUMP MEMORY-INTEL HEX

This command dumps the contents of the range of memory specified in standard Intel Hex format. You give it the PATH, beginning memory address, ending memory address, and an offset that is to be added to the record address field.

DB DB<PATH>,<ADR>,<ADR> DUMP MEMORY IN BINARY

This command dumps the contents of the range of memory specified in pure binary or memory image form. This is the form that should be used if you intend to create a .BIN file that will later be executed under the K-OS operating system. If a binary image is sent to a device then it may contain code sequences that will wreck havok on the device.

LS LS<PATH>,<OFFSET> LOAD S RECORDS

This command is used to load a program in Motorola S record format from the path specified.

LH LH<PATH>,<OFFSET> LOAD INTEL HEX

This command is used to load a progam in Intel hex format from the path specified.

LB LB<PATH>,<OFFSET> LOAD BINARY

This command is used to load a binary file from the device specified. This is the normal command to use to load a .BIN file that was prepared to run under the K-OS operating system.

VS VS<PATH>,<OFFSET> VERIFY S RECORDS

This command works like the load S record command but instead of actually loading something into memory it mearly notes any differences from the veryify file and the contents of memory.

VH VH<PATH>,<OFFSET> VERIFY HEX

This is used to verify an Intel hex file.

VB VB<PATH>,<OFFSET> VERIFY BINARY

This command is used to verify the loading or changes in a binary file.