

A **debugger** is a program that enables a programmer to monitor the execution of a program. It can typically

- examine the contents of symbolic memory locations and registers;
- set breakpoints;
- set watchpoints;
- execute a program one instruction at a time;
- evaluate expressions and perform arithmetic by using the debugger as a calculator.

A **breakpoint** is the address of an instruction where execution of a program is suspended just before executing that instruction. At the breakpoint, the debugger gains control and the contents of memory locations and registers can be examined.

A **watchpoint** is the address of a location that is specified to determine when an instruction modifies that location. The program stops if an instruction modifies a watchpoint. The debugger gains control immediately after that instruction is executed. The programmer can then determine if the location has been correctly modified.

A **tracepoint**, when reached during execution while under the control of the debugger, causes the debugger to print a message indicating that the point has been reached before continuing execution.

To prepare for a debugging session, the Macro program must be assembled with
\$ MACRO/LIST/ENABLE=DEBUG filename1

If a Pascal module is used, it must be compiled with
\$ PASCAL/DEBUG/NOOPTIMIZE/LIST filename2

The two are linked with
\$ LINK/DEBUG filename1 + filename2

At runtime, you type

\$ RUN filename1

and the debugger automatically gains control, displays messages identifying itself, and issues the debugger prompt
DBG>

Before using the debugger, you should know that the VAX-11 debugger will, by default, do the following:

1. display its output values in hexadecimal;
2. interpret values in input commands as hexadecimal values; and
3. display data values as longword integers.

When you see the debugger prompt, you may issue any debugging command.

GO Command

Format: **GO**

Function: Start or continue execution of the program. The program runs until one of the following occurs:
1. a breakpoint or watchpoint or runtime error is encountered and the debugger takes control and prompts for a command;
2. an input statement to read from the terminal is encountered and the program waits for your input; or
3. the program is completed and the debugger takes control.

EXIT Command

Format: **EXIT**

Function: Terminate debugging session and return to DCL level.

<CTRL> Commands

Format: **<CTRL> Z**

Function: Terminate debugging session and return to DCL level.

Format: **<CTRL> C** or **<CTRL> Y**

Function: Interrupt execution of the program and return to DCL level. If one of the DCL commands **DEBUG** or **CONTINUE** is keyed, control is returned to the debugger at the point where the program was interrupted.

HELP Commands

Format: **HELP**

Function: Display the debugger commands.

Format: **HELP command**

Function: Display information about the specified command.

STEP Commands

Format: **STEP**

Function: Execute the next instruction.

Format: **STEP n**

Function: Execute the next n instructions. (n is a decimal integer)

SET Commands

Format: **SET RADIX DECIMAL**

Function: Change the mode to decimal until it is canceled or set to another mode.

Format: **SET RADIX HEX**

Function: Change the mode to hexadecimal until it is canceled or set to another mode.

Format: **SET RADIX OCTAL**

Function: Change the mode to octal until it is canceled or set to another mode.

Format: **SET TYPE ASCII**

Function: Change the type to ASCII until it is canceled or set to another type.

Format: **SET TYPE BYTE**

Function: Change the type to BYTE until it is canceled or set to another type.

Format: **SET TYPE LONG**

Function: Change the type to LONG until it is canceled or set to another type.

Format: **SET TYPE WORD**

Function: Change the type to WORD until it is canceled or set to another type.

EXAMINE Commands

Format: **EXAMINE address**

Function: Display the contents of the location at the specified address.

Format: **EXAMINE address1:address2**

Function: Display the contents of contiguous locations between address1 and address2, inclusive.

Format: **EXAMINE/DECIMAL address**

Function: Display, in decimal, the contents of the location at the specified address.

Format: **EXAMINE/BYTE address**

Function: Display the contents of the byte at the specified address.

Format: **EXAMINE/HEX/WORD address**

Function: Display, in hexadecimal, the contents of the word at the specified address.

Format: **EXAMINE/ASCII address**

Function: Display, in ASCII, the contents of the location at the specified address.

Format: **EXAMINE/INSTRUCTION address**

Function: Display the instruction at the specified address.

Format: **EXAMINE**

Function: Display the contents of the location that is immediately after the last location displayed.

Format: **EXAMINE register**

Function: Display the contents of a register, where the register is specified by R0, R1, ..., R12, AP, FP, SP, or PC.

Format: **EXAMINE register1:register2**

Function: Display the contents of contiguous register locations, where register1 and register2 are specified by R0, R1, ..., R12, AP, FP, SP, or PC.

SET BREAK Commands

Format: **SET BREAK address**

Function: Set breakpoint at the address specified. The program stops execution just before the instruction at the breakpoint is executed and the debugger takes control. At this point the contents of specific locations can be examined or altered.

Format: **SET BREAK /AFTER:n address**

Function: Set breakpoint to take effect when the program reaches that address the nth time during execution and every time after that. The first (n-1) times are ignored.

Format: **SET BREAK address DO command**

Function: Execute one or more commands when the breakpoint is reached.

Format: **SET BREAK /AFTER:n address DO command**

Function: Set breakpoint to take effect when the program reaches that address the nth time during execution and every time after that and execute the command(s) when that happens.
Format: **SET EXCEPTION BREAK**
Function: Treat execution errors as breakpoints and give control to the user rather than abort the program.

SET WATCH Command

Format: **SET WATCH address**
Function: Set watchpoint at the address specified. The program stops execution whenever an instruction writes to a watchpoint location. The system displays the address of the instruction that modified the watchpoint and the debugger takes control.

SET TRACE Commands

Format: **SET TRACE address**
Function: Set tracepoint at the address specified.
Format: **SET TRACE/BRANCH**
Function: Set tracepoints at all branch instructions.
(Note: Setting a tracepoint where a breakpoint has already been set cancels the breakpoint and retains the tracepoint.)

CANCEL Commands

Format: **CANCEL BREAK address**
Function: Cancel breakpoint at specified address.
Format: **CANCEL BREAK/ALL**
Function: Cancel all breakpoints.
Format: **CANCEL WATCH address**
Function: Cancel watchpoint at specified address.
Format: **CANCEL WATCH/ALL**
Function: Cancel all watchpoints.
Format: **CANCEL ALL**
Function: Cancel all breakpoints and watchpoints, restoring any user-set modes and types to their defaults.
Format: **CANCEL EXCEPTION BREAK**
Function: Cancel the breakpoint that stops the execution when an error occurs.
Format: **CANCEL TRACE/ALL**
Function: Cancel all tracepoints.
Format: **CANCEL MODE**
Function: Restore all modes and types to their default values.

SHOW Commands

Format: **SHOW BREAK**
Function: Display current breakpoints.
Format: **SHOW WATCH**
Function: Display current watchpoints.
Format: **SHOW TRACE**
Function: Display all tracepoints.
Format: **SHOW MODE**
Function: Display current mode.
Format: **SHOW TYPE**
Function: Display current type.

DEPOSIT Command

Format: **DEPOSIT address=value**
Function: Store value at address specified. The DEPOSIT command may have qualifiers similar to those in the EXAMINE command.

Evaluate Command

Format: **EVALUATE expression**
Function: Display the value of the expression where the EVALUATE command can use the qualifiers /DECIMAL, /HEX, or /OCTAL.