**MVME197BUG/D1A1**
**August 1994**

# Supplement to

# MVME197BUG

# 197Bug™ Debugging Package

# User's Manual

# (MVME197BUG/D1)

The attached pages are replacements and/or additions to the user's manual. They correct minor errors and update some features.

Please replace the pages according to the following table and place this page behind the title page of the user's manual as a record of this change:

| Replace Old | With New |
|---|---|
| v/vi, | v/vi, |
| xvii/xviii, | xvii/xviii, |
| 1-3/1-4, | 1-3/1-4, |
| 1-7/1-8, | 1-7/1-8, |
| 1-11 through 1-14, | 1-11 through 1-14, |
| 3-53 through 3-56, | 3-53 through 3-56, |
| 3-63/3-64, | 3-63/3-64, |
| 5-3/5-4, | 5-3/5-4, |
| 5-73/5-74, | 5-73/5-74, |
| A-1 through A-4, | A-1 through A-4, |
| IN-5/IN-4, | IN-5/IN-4, |
| IN-9/IN-10, | IN-9/IN-10, |
| IN-13/IN-14 | IN-13/IN-14 |

- ❏ A vertical bar ( | ) in the margin of a replacement page indicates a text change or addition.
- ❏ The supplement number is shown at the bottom of each replacement page.

## Notice

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.



## Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282-9602

The following conventions are used in this document:

**bold**

is used for user input that you type just as it appears. Bold is also used for commands, options and arguments to commands, and names of programs, directories, and files.

*italic*

is used for names of variables to which you assign values. Italic is also used for comments in screen displays and examples.

`courier`

is used for system output (e.g., screen displays, reports), examples, and system prompts.

**<RETURN>**

represents the carriage return key.

**CTRL**

represents the Control key. Execute control characters by pressing the **CTRL** key and the letter simultaneously, e.g., **CTRL-d**.

## Related Documentation

The following publications are applicable to the MVME197 module series and may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your Motorola sales office.

| Document Title | Motorola Publication Number |
|---|---|
| MVME197LE Single Board Computer User's Manual | MVME197LE |
| MVME197LE Single Board Computer Support Information | SIMVME197LE |
| MVME197DP and MVME197SP Single Board Computer User's Manual | MVME197 |
| MVME197DP and MVME197SP Single Board Computer Support Information | SIMVME197 |
| MVME197LE, MVME197DP, and MVME197SP Single Board Computers Programmer's Reference Guide | MVME197PG |
| MVME197BUG 197Bug Diagnostic Firmware User's Manual | MVME197DIAG |
| MVME712M Transition Module and P2 Adapter Board User's Manual | MVME712M |
| MVME712-12, MVME712-13, MVME712A, MVME712AM, and MVME712B Transition Module and LCP2 Adapter Board User's Manual | MVME712A |
| MC68040/MC68EC040/MC68CL040 Microprocessor User's Manual | M68040UM/AD |
| MC88110 Second Generation RISC Microprocessor User's Manual | MC88110UM/AD |
| MC88410 Secondary Cache Controller User's Manual | MC88410UM/AD |

**Notes**

1. **The support information manuals (SIMVME197LE and SIMVME197) contain: the connector interconnect signal information, parts lists, and the schematics for the specific board indicated.**

# List of Figures

## FIGURES

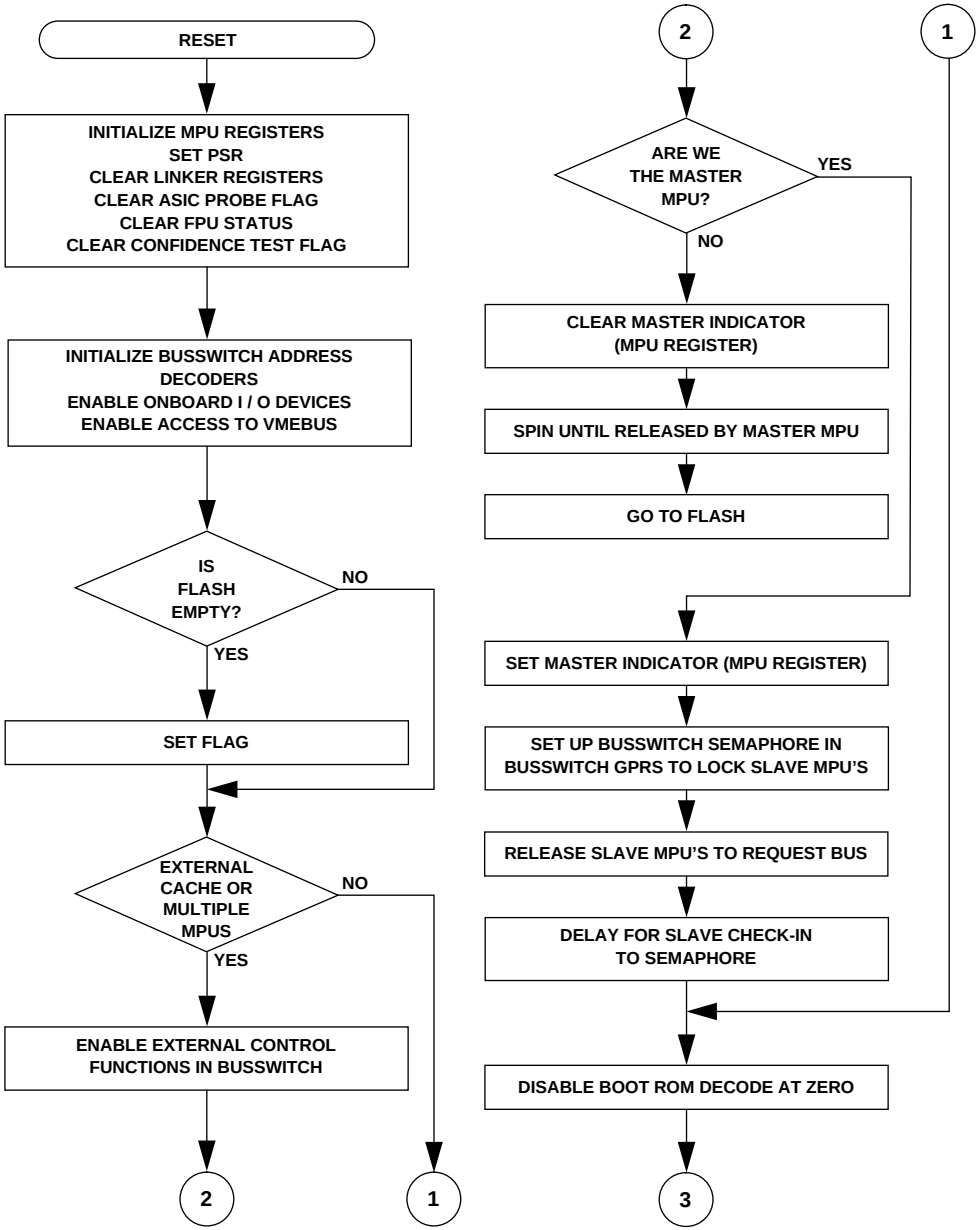


**Figure 1-1. Flow Diagram of 197Bug Start-Up for BOOT ROM (Sheet 1 of 4)**

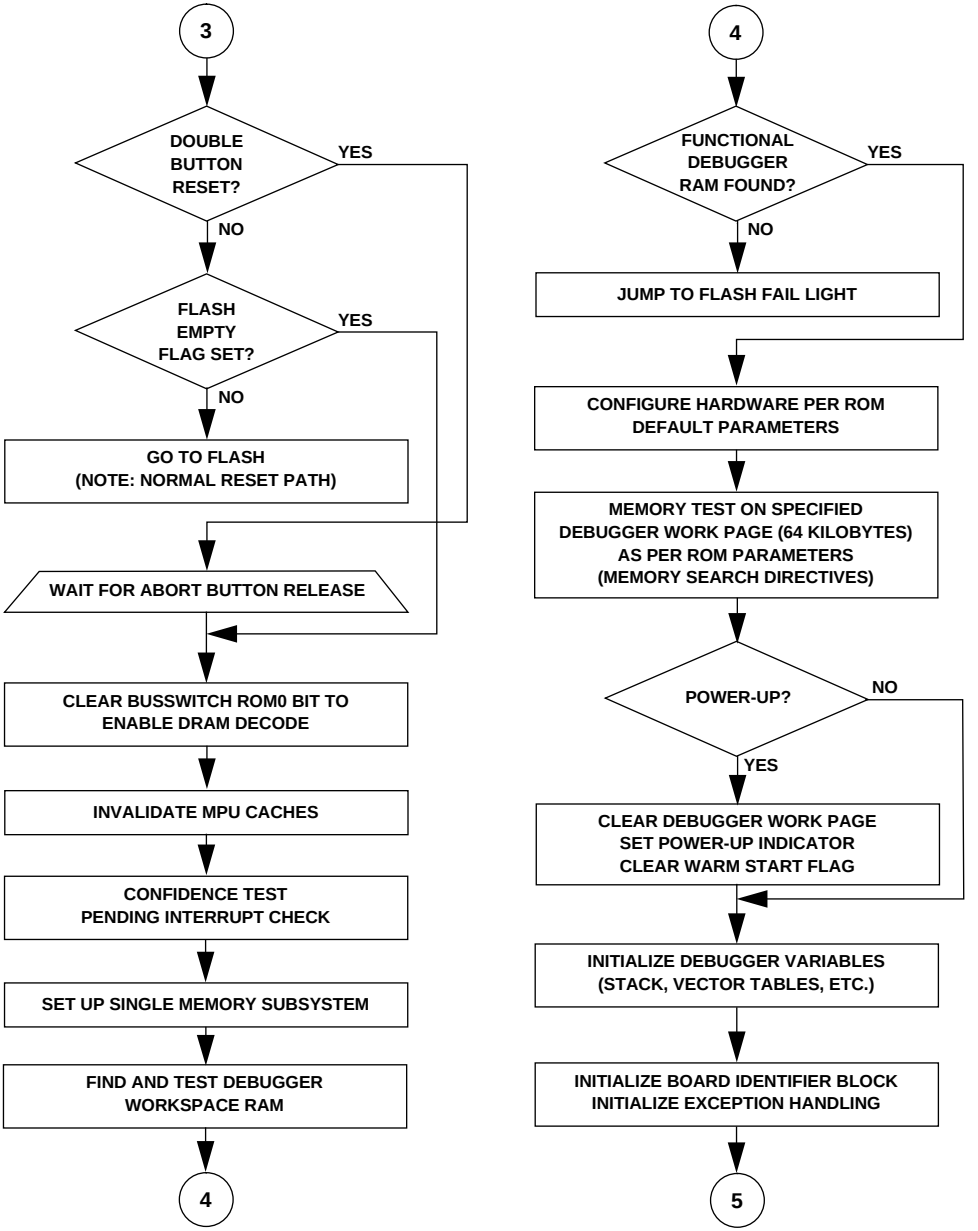


**Figure 1-1. Flow Diagram of 197Bug Start-Up for BOOT ROM (Sheet 2 of 4)**

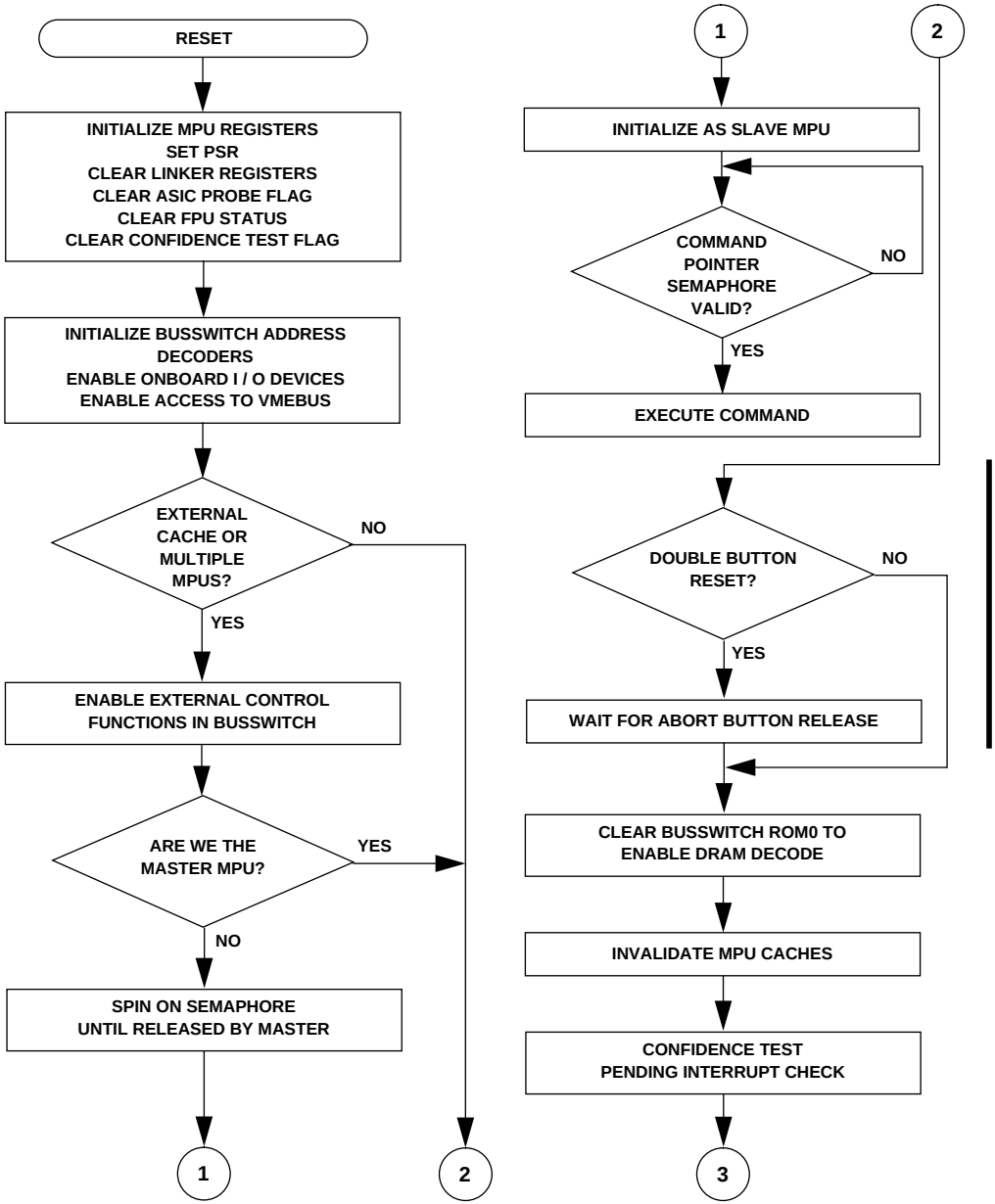


**Figure 1-2. Flow Diagram of 197Bug Start-Up for FLASH-based Debugger (Sheet 1 of 6)**

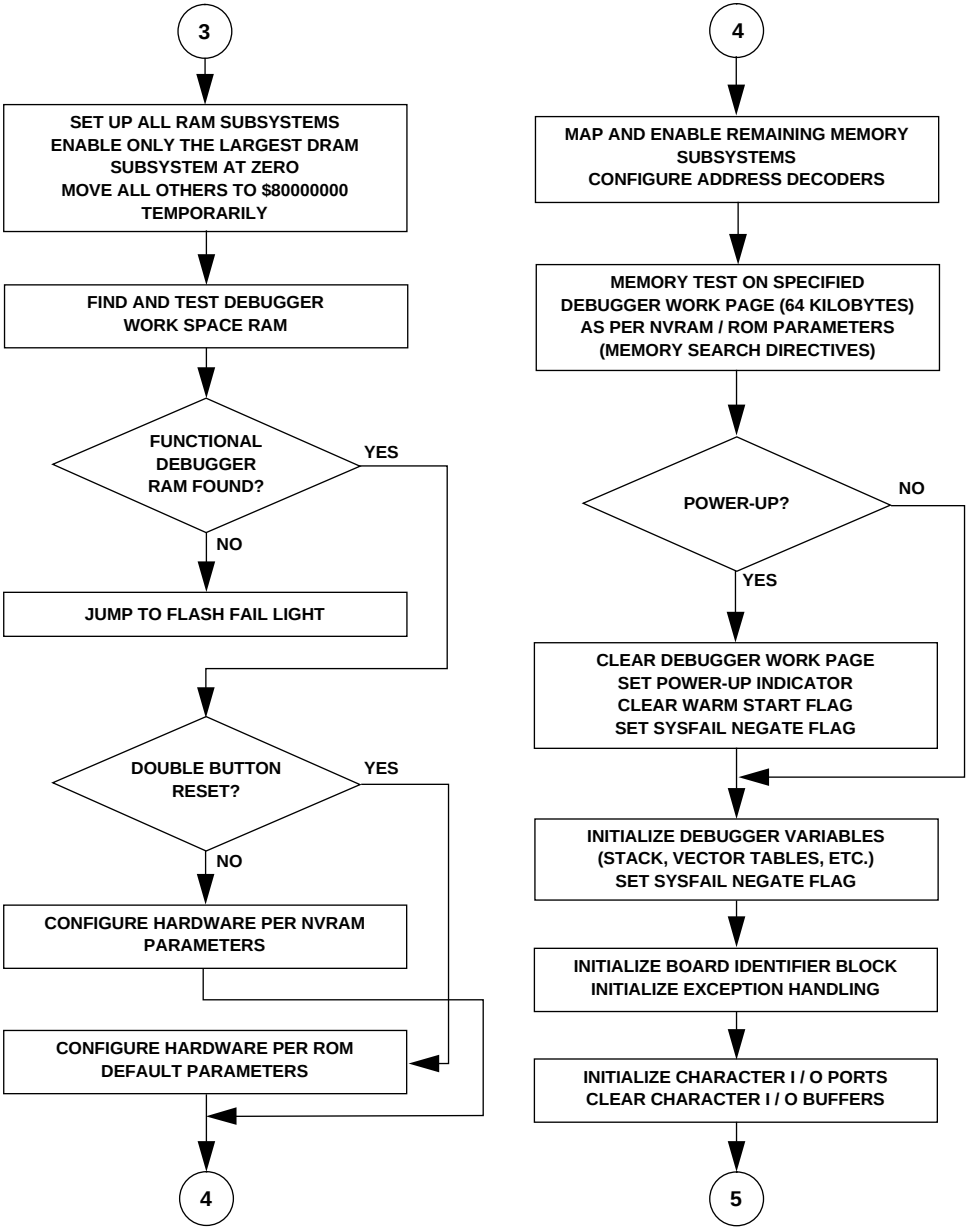


**Figure 1-2. Flow Diagram of 197Bug Start-Up for FLASH-based Debugger (Sheet 2 of 6)**

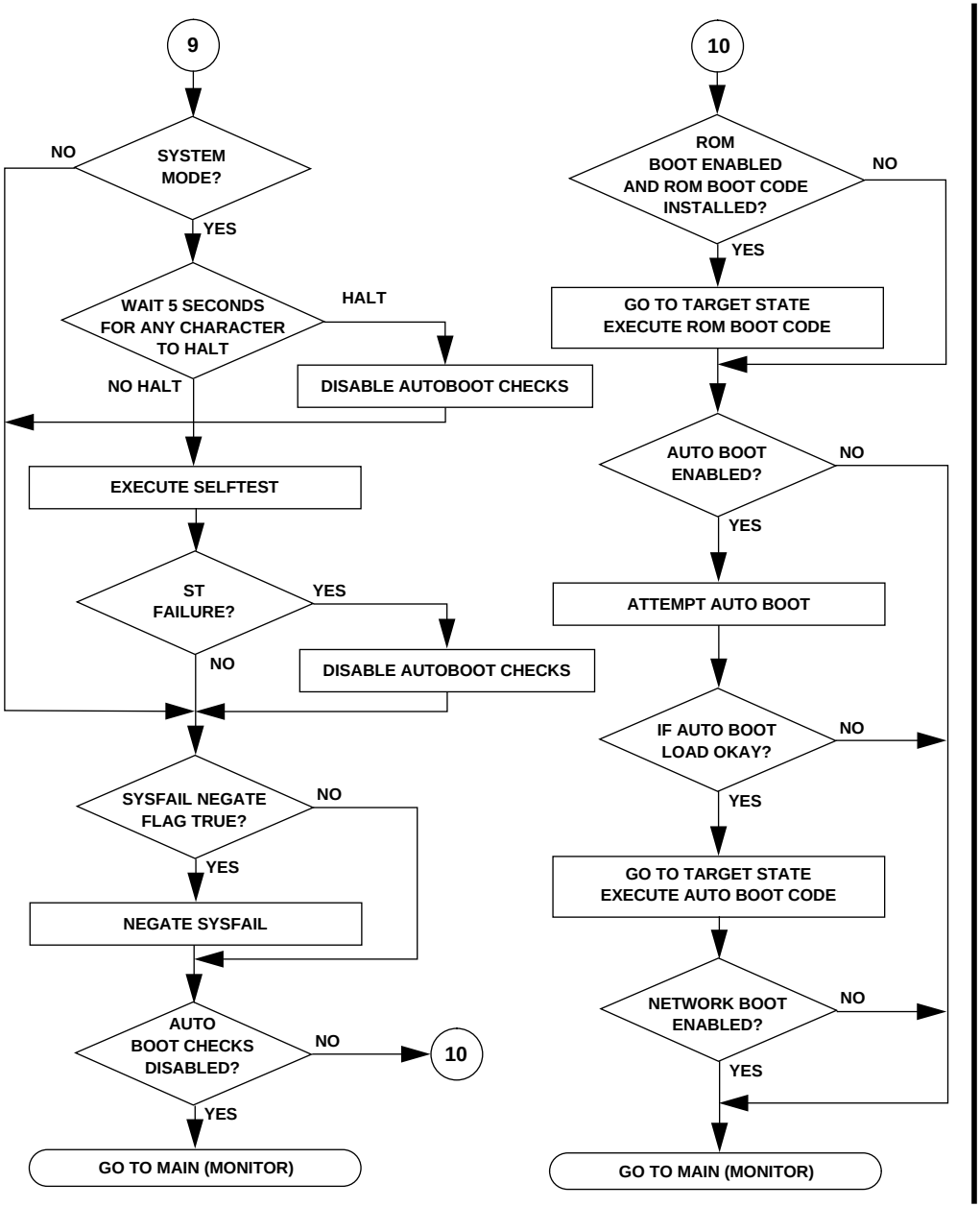


**Figure 1-2. Flow Diagram of 197Bug Start-Up for FLASH-based Debugger (Sheet 5 of 6)**

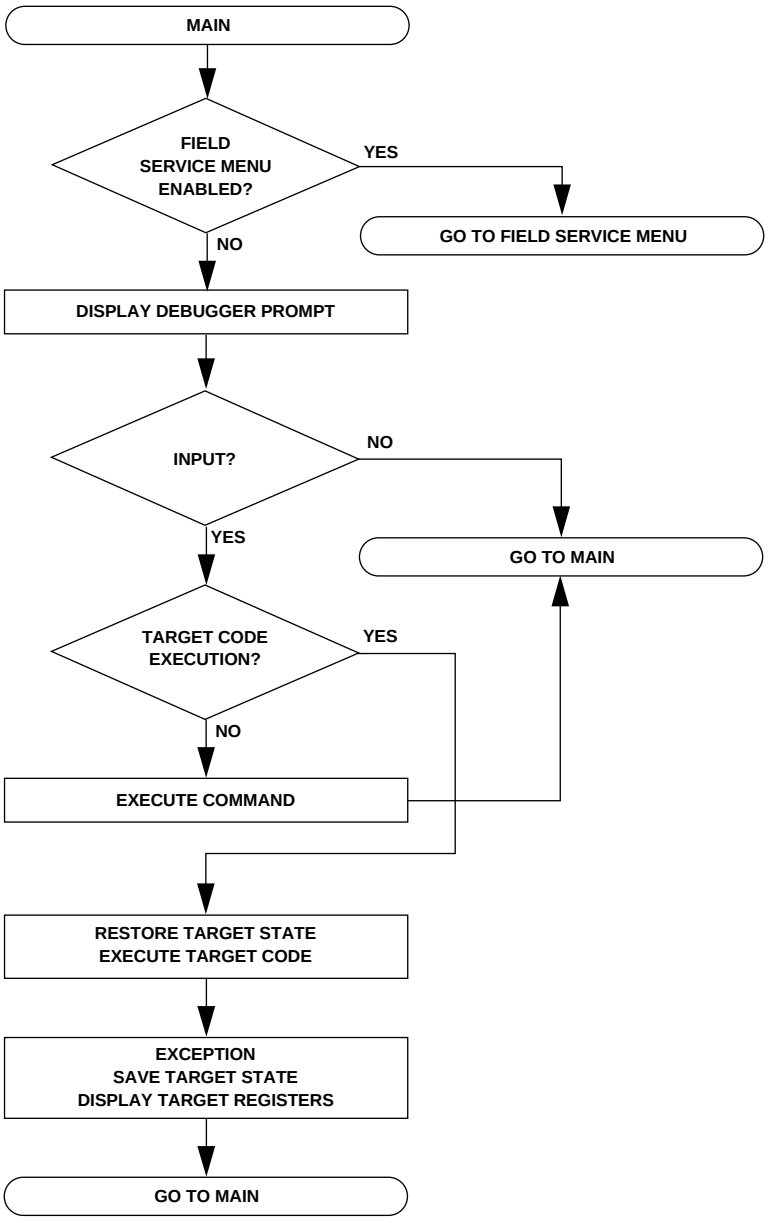


**Figure 1-2. Flow Diagram of 197Bug Start-Up for FLASH-based Debugger (Sheet 6 of 6)**

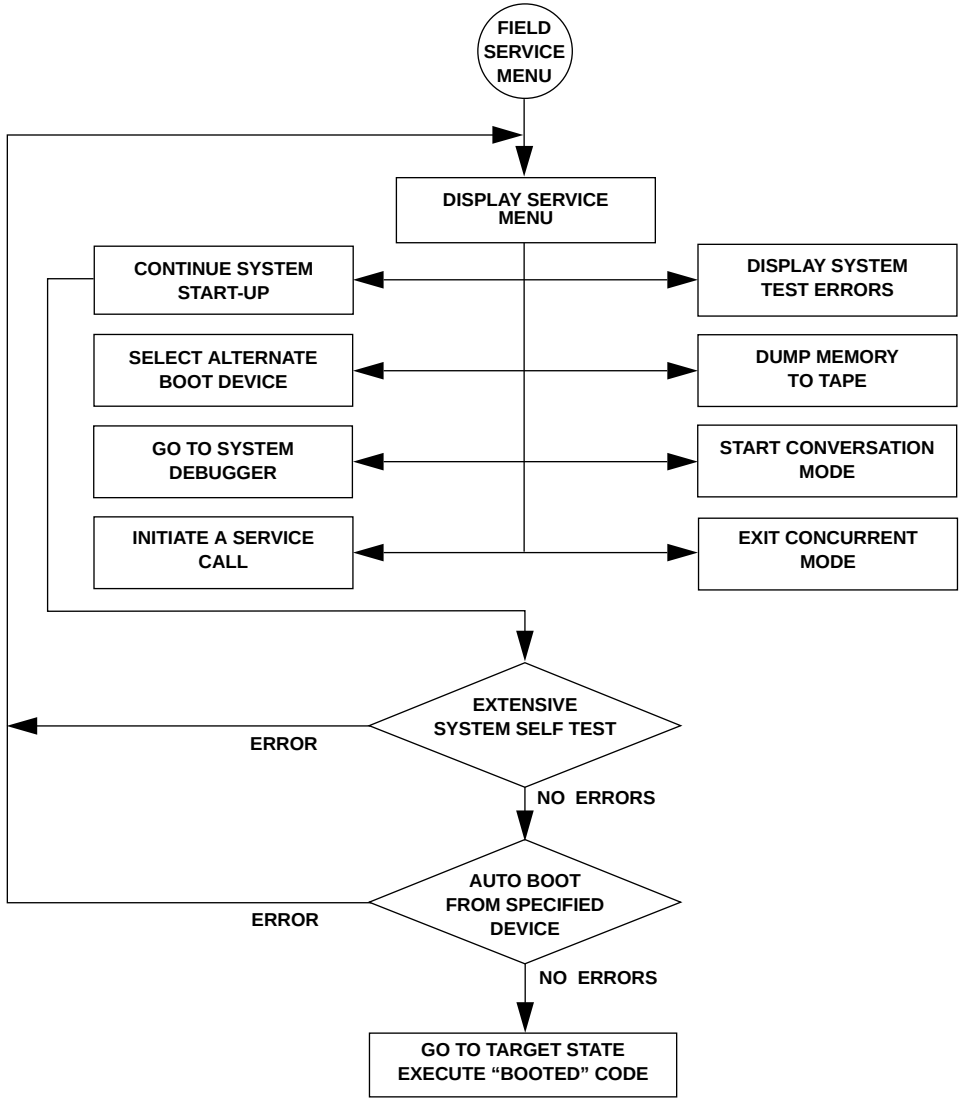


**Figure 1-3. Flow Diagram of 197Bug Field Service Menu Operational Mode**

## Comparison with M68000-Based Firmware

Those users who have used one or more of Motorola's other debugging packages will find 197Bug very similar, after making due allowances for the architectural differences between the M68000 and M88000 CPU architectures. These are primarily reflected in the instruction mnemonics and addressing modes of the assembler/disassembler, and in the use of registers instead of the stack for the passing of arguments to or from the TRAP #496 handler. Some effort has also been made to make the interactive commands more consistent. For example, delimiters between commands and arguments may now be commas or spaces interchangeably.

## 197Bug Implementation

### FLASH-Based Debugger

197Bug is contained in the FLASH memory devices located on the MVME197 board. The FLASH devices are electrically re-writable and may be reprogrammed without removing the physical devices from the MVME197 board. This allows the user to incorporate updated versions of the 197Bug as they become available by simply loading the newer version into the FLASH memory and overwriting the older version.

The **PFLASH** command (refer to the *197Bug Debugger Command Set* chapter) describes how to reprogram the FLASH memory contents. The executable code is checksummed at every power-on or reset firmware entry. Users are cautioned against reprogramming of the FLASH memory contents unless rechecksum precautions are taken. Refer to the **CS** command description in the *197Bug Debugger Command Set* chapter for checksum information.


**WARNING**

**Reprogramming any portion of FLASH memory will erase everything currently contained in FLASH, including the debugger. A valid version of 197Bug must be transferred from RAM into the FLASH during FLASH reprogramming in order for the debugger to operate.**

**The *197Bug Debugger Command Set* chapter describes the command set of the FLASH-based debugger.**

### BOOT ROM

A subset of 197Bug is also programmed into the BOOT ROM, which is an EPROM or One-Time Programmable ROM on the MVME197 module. This scaled-down 197Bug is referred to as the “BootBug”, or “197BBug”.

# Set Environment to Bug/Operating System

**ENV**

**ENV [;[D]]**



The **ENV** command allows the user to interactively view/configure all Bug operational parameters that are kept in Battery Backup RAM (BBRAM), also known as Non-Volatile RAM (NVRAM). The operational parameters are saved in NVRAM and used whenever power is lost.

Any time the Bug uses a parameter from NVRAM, the NVRAM contents are first tested by checksum to ensure the integrity of the NVRAM contents. In the instance of BBRAM checksum failure, certain default values are assumed as stated in the examples below.

The bug operational parameters (which are kept in NVRAM) are not initialized automatically on power up/warm reset. It is up to the Bug user to invoke the **ENV** command. Once the **ENV** command is invoked and executed without error, Bug default and/or user parameters are loaded into NVRAM along with checksum data. If any of the operational parameters have been modified, these new parameters will not be in effect until a reset/powerup condition.

If the **ENV** command is invoked with no options on the command line, the user is prompted to configure all operational parameters. If the **ENV** command is invoked with the option **D**, ROM defaults will be loaded into NVRAM.

## Programming the VMEbus to Local Peripheral Bus Map Decoders

The VMEbus slave map decoders allow a VMEbus master to view a block of the local peripheral bus (usually memory) through a VMEbus window. The following procedure can be used with the **ENV** command to configure the VMEbus to Local Peripheral Bus (slave) map decoders. This is not the only procedure that can be used to program the map decoders.

1. Determine the local base address (for onboard DRAM memory this is the Base Address of Local Memory) and size of the memory block to be viewed through the VMEbus window. The following restrictions must be considered when defining the local peripheral bus address of the block and the block size.

   The map decoder logic performs address translation by replacing a portion of the VMEbus address with an address from the address translation register. Therefore, translation is performed in increments of the block size and the block size must be a power of 2 and located on a power of 2 boundary. For example, a 32MB block cannot be addressed on

a 4MB boundary. However, any 4MB block of the 32MB memory can be addressed on any 4MB boundary.

Also note that if the block size is not a power of 2, then rounding up to a power of 2 boundary is necessary. For example, a 12MB block must be accessed at 0, 16MB, 32MB, etc.

2. Set the Slave Address Translation Address Register parameter with the LOCAL base address of the block.
3. Set the Slave Address Translation Select Register parameter with the 2's compliment of the block size.
4. Set the Slave Starting Address Register parameter with the starting address of the VMEbus window.
5. Set the Slave Ending Address Register parameter with the ending address of the VMEbus window.

**Note** **The VMEbus window size may be any number of 64KB blocks up to the block size.**

6. If the VMEbus window is entirely below the 16MB boundary, enable A24 and/or A32 addressing. If the VMEbus window is entirely above the 16MB boundary, enable only A32 addressing. If the VMEbus window spans the 16MB boundary, enable A32 addressing. If access is required to the portion below the 16MB boundary using A24 addressing, the second map decoder should be programmed to provide A24 access to the portion of the VMEbus window below the 16MB boundary.

   Set the Slave Control parameter to $01EF to enable A32 addressing, $01DF to enable A24 addressing, and $01FF to enable A32 and A24 addressing.

## Configuring ENV Parameters

The parameters to be configured are:

```
Bug or System environment [B/S] = S?
```

B — Disables automatic execution of extended confidence testing (**ST**) on any reset and selects the 197-Bug> prompt in the debugger command line mode. The **SD** command must be issued to allow diagnostic execution.

S — Enables automatic execution of extended confidence testing (**ST**) on any reset. Also, selects the 197-Diag> prompt, enabling diagnostic test execution from the debugger command line.

```
Field Service Menu Enable [Y/N] = Y?
```

Y Field service menu mode is selected. The menu operation is explained in Appendix A. (Default)

N Debugger command line mode is selected. The 197-Bug or 197-Diag prompt will be displayed based on the Bug or System environment parameter.

```
Remote Start Method Switch [G/M/B/N] = B?
```

The Remote Start Method Switch is used when the MVME197 is cross-loaded from another VME-based CPU, to start execution of the cross-loaded program.

G Use the Global Control and Status Register (GCSR) in the VMEchip2 to pass and start execution of cross-loaded program.

M Use the Multiprocessor Control Register (MPCR) in shared RAM to pass and start execution of cross-loaded program.

B Use both the GCSR and the MPCR methods to pass and start execution of cross-loaded program. (Default)

N Do not use any Remote Start Method.

```
Probe System for Supported Disk/Tape Controllers [Y/N] = Y?
```

Y Accesses will be made to the VMEbus to determine the presence of supported controllers. (Default)

N Accesses will not be made to the VMEbus to determine the presence of supported controllers.

```
Negate VMEbus SYSFAIL* Always [Y/N] = N?
```

Y Negate VMEbus SYSFAIL during board initialization.

N Negate VMEbus SYSFAIL after successful completion or entrance into the bug command monitor. (Default)

```
Local SCSI Bus Reset on Debugger Setup [Y/N] = Y?
```

Y Local SCSI bus is reset on debugger setup.

N Local SCSI bus is not reset on debugger setup.

```
Local SCSI Bus Negotiations Type [A/S/N] = A?
```

A Use Asynchronous negotiations on the Local SCSI bus.

S Use Synchronous negotiations on the Local SCSI bus.

N (None). Do not precede the SCSI data transfer with a type negotiation. Do all data transfers in Asynchronous mode.

**ENV**



```
Ignore CFGA Block on a Hard Disk Boot [Y/N] = Y?
```

Y Enable the ignorance of the Configuration Area (CFGA) Block (hard disk only).

N Do not enable the ignorance of the Configuration Area (CFGA) Block.

```
Auto Boot Enable [Y/N] = N?
```

Y The auto boot function is enabled.

N The auto boot function is disabled. (Default)

```
Auto Boot at power-up only [Y/N] = Y?
```

Y Auto Boot is attempted at power up reset only. (Default)

N Auto Boot is attempted at any reset.

```
Auto Boot Controller LUN = 00?
```

Refer to Appendix E for a listing of disk/tape controller modules currently supported by the Bug. The default for this parameter is $0.

```
Auto Boot Device LUN = 00?
```

Refer to Appendix E for a listing of disk/tape devices currently supported by the Bug. The default for this parameter is $0.

```
Auto Boot Abort Delay = 15?
```

Time in seconds that the Auto Boot sequence will delay before starting the boot. The purpose is to allow the user the option of stopping the boot by use of the Break key. The time value is from 0 through 255 seconds.

```
Auto Boot Default String [NULL for an empty string] = <none>
```

The user may specify a string (filename) which is passed on to the code being booted. The maximum length of this string is 16 characters. The default for this parameter is the null string.

```
ROM Boot Enable [Y/N] = N?
```

Y The ROMboot function is enabled.

N The ROMboot function is disabled. (Default)

```
ROM Boot at power-up only [Y/N] = Y?
```

Y ROMboot is attempted at power up only. (Default)

N ROMboot is attempted at any reset.

```
Master Address Translation Select #4 = 00000000?
```

This register defines which bits of the address are significant. A logical one "1" indicates significant address bits, logical zero "0" is non-significant. (Default is 0).

```
Master Control #4 = 00?
```

This defines the access characteristics for the address space defined with this master address decoder. The default is $00.

```
Short I/O (VMEbus A16) Enable [Y/N] = Y?
```

Y Yes, Enable the Short I/O Address Decoder. (Default)

N Do not enable the Master Address Decoder.

```
Short I/O (VMEbus A16) Control = 01?
```

This defines the access characteristics for the address space defined with the Short I/O address decoder. The default is $01.

```
F-Page (VMEbus A24) Enable [Y/N] = Y?
```

Y Yes, Enable the F-Page Address Decoder. (Default)

N Do not enable the F-Page Address Decoder.

```
F-Page (VMEbus A24) Control = 02?
```

This defines the access characteristics for the address space defined with the F-Page address decoder. The default is $02.

```
ROM Speed Bank A Code = 03?
ROM Speed Bank B Code = 03?
```

These parameters are used to set up the ROM speed, which is dependent on the MPU clock speed.

| MPUCLK | ROM Speed A and B (Default values) |
|---|---|
| 50 | 3 |
| 40 | 2 |
| 33 | 2 |
| any other | 7 |

```
PCC2 Vector Base = 05?
VMEC2 Vector Base #1 = 06?
VMEC2 Vector Base #2 = 07?
```

These parameters are the base interrupt vector for the component specified. (Default: PCCchip2 = $05, VMEchip2 Vector 1 = $06, VMEchip2 Vector 2 = $07).

```
VMEC2 GCSR Group Base Address = D0?
```

This parameter specifies the group address ($FFFFXX00) in Short I/O for this board. (Default = $D0).

```
VMEC2 GCSR Board Base Address = 00?
```

This parameter specifies the base address ($FFFFCEXX) in Short I/O for this board. (Default = $00).

```
VMEbus Global Time Out Code = 01?
```

This controls the VMEbus timeout when systems controller. (Default $01 = 64 microseconds).

```
Local Peripheral Bus Time Out Code = 01?
```

This controls the local peripheral bus timeout. (Default $00 =64 microseconds).

```
VMEbus Access Time Out Code = 02?
```

This controls the local peripheral bus to VMEbus access timeout. (Default $02 = 32 milliseconds).

**Table 5-1. 197Bug System Call Routines (Continued)**

| Code | Function | Description |
|---|---|---|
| $0024 | .WRITELN | Output line (pointer/count format) |
| $0025 | .WRITDLN | Output line with data (pointer/count format) |
| $0026 | .PCRLF | Output carriage return and line feed |
| $0027 | .ERASLN | Erase line |
| $0028 | .WRITD | Output string with data (pointer/count format) |
| $0029 | .SNDBRK | Send break |
| $0043 | .DELAY | Timer delay function |
| $0050 | .RTC_TM | Time initialization for RTC |
| $0051 | .RTC_DT | Date initialization for RTC |
| $0052 | .RTC_DSP | Display RTC time and date |
| $0053 | .RTC_RD | Read the RTC Registers |
| $0060 | .REDIR | Redirect I/O of a TRAP #496 function |
| $0061 | .REDIR_I | Redirect input |
| $0062 | .REDIR_O | Redirect output |
| $0063 | .RETURN | Return to 197Bug |
| $0064 | .BINDEC | Convert binary to Binary Coded Decimal (BCD) |
| $0067 | .CHANGEV | Parse value |
| $0068 | .STRCMP | Compare two strings (pointer/count format) |
| $0069 | .MULU32 | Multiply two 32-bit unsigned integers |
| $006A | .DIVU32 | Divide two 32-bit unsigned integers |
| $006B | .CHK_SUM | Generate checksum |
| $0070 | .BRD_ID | Return pointer to board ID packet |
| $0071 | .ENVIRON | Access boot environment parameters |
| $0073 | .PFLASH | Program FLASH Memory |
| $0090 | .SIOPEPS | Retrieve SCSI pointers (NOTE 3) |
| $0100 | .FORKMPU | Fork MPU (Multiple MPU Configuration) (NOTE 2) |
| $0101 | .FORKMPUR | Fork Idle MPU with Register Set (NOTE 2) |

**Table 5-1. 197Bug System Call Routines (Continued)**

| Code | Function | Description |
|---|---|---|
| $0110 | .IDLEMPU | Idle MPU (Multiple MPU Configuration) (NOTE 2) |
| $0120 | .IOINQ | Port Inquiry |
| $0124 | .IOINFORM | Port Inform |
| $0128 | .IOCONFIG | Port Configure |
| $012C | .IODELETE | Port Delete |
| $0130 | .SYMBOLTA | Attach Symbol Table |
| $0131 | .SYMBOLTD | Detach Symbol Table |
| $0140 | .ACFSTAT | ACFAIL Status Inquiry |



**Notes**

1. **In most examples of commands and displays given in this manual, 197Bug is used. However, the commands, displays, and system calls apply to all 88K RISC debugging packages, unless otherwise noted.**
2. **This utility is only available on multi-processor modules such as the MVME197 series and the MVME188 series.**
3. **This function applies only to modules with SCSI; does not apply to MVME188 or MVME188A series modules.**

## .PCRLF Function



TRAP FUNCTION: .PCRLF - Print **<CR><LF>** sequence

CODE: $0026

DESCRIPTION: .PCRLF sends a **<CR><LF>** sequence to the default output port.

ENTRY CONDITIONS:

No arguments required.

EXIT CONDITIONS DIFFERENT FROM ENTRY:

None

## .PFLASH Function

**.PFLASH**

TRAP FUNCTION: .PFLASH - Program FLASH memory

CODE: $0073

DESCRIPTION: The purpose of this TRAP is to program FLASH memory under program control. The address of the packet is passed as an argument to the function. The address of the packet is passed in the longword memory location pointed to by the current stack pointer. The packet contains the necessary arguments/data to program the FLASH memory.

5

ENTRY CONDITIONS:

SP ==> Address: Starting address of control packet *longword*

EXIT CONDITIONS DIFFERENT FROM ENTRY:

None

FORMAT OF FLASH MEMORY CONTROL PACKET:

The FLASH Memory Control Packet must be longword/word (32-bit) aligned.

| | **31** **24** | **23** **16** | **15** **5** | **8** | **7** **0** |
|---|---|---|---|---|---|
| **$00** | Status Word | | Control Word | | |
| **$04** | Source Starting Address | | | | |
| **$08** | Number of Bytes to Program | | | | |
| **$0C** | Destination Starting Address | | | | |
| **$10** | Instruction Execution Address | | | | |

Field descriptions:

Control/Status Word

Specifies control and status of the various phases of the FLASH memory programming. This parameter has two 16-bit parts: bits #31 to #16 specify status and bits #15 to #0 specify control.

# MVME197BUG SYSTEM MODE OPERATION

A

## General Description

To provide compatibility with the Motorola Delta Series systems, the MVME197Bug has a special mode of operation for system environments.

If system mode is selected in the **ENV** parameters (default), extended confidence tests are run automatically at power-up or at any reset of the MVME197. A delay precedes this testing and the tests can be skipped by typing "S" on the console keyboard. During this delay, control will be passed to the debugger prompt or menu (see below) if errors are detected.

If the field service menu is enabled, a menu will be displayed instead of the debugger prompt, if none of the alternate actions (**AB**, **RB**, **NBO**, etc.) has been enabled via the **ENV** parameters. This allows several system start-up features to be selected, such as:

1. Continue System Start-Up
2. Select Alternate Boot Device
3. Go to System Debugger
4. Initiate Service Call
5. Display System Test Errors
6. Dump Memory to Tape

The flow of menu mode operation is shown in Figure A-1. Upon either power up or system reset, the MVME197 first executes a limited confidence test suite. This is the same test suite that the Bug normally executes on power up when not in the system mode. Upon successful completion of the limited confidence tests, a five second period is allowed to interrupt the start-up sequence. By typing any character the user can cause the module to deviate from the normal start-up sequence and display one of the debugger prompts or display the field service menu, permitting the selection of an alternate boot device, entry to the debugger, etc., as described above. Upon selection of "continue start-up" the module conducts a more extensive confidence test (**ST**). Successful completion of the extended confidence test initiates the autoboot sequence, with boot taking place either from the default device (refer to the *197Bug Debugger Command Set* chapter for information on entering/changing the default boot device) or from the selected boot device if an alternate device has been selected.

If the limited confidence test fails to complete correctly, the FAIL LED may blink to indicate a failure and processing is suspended.

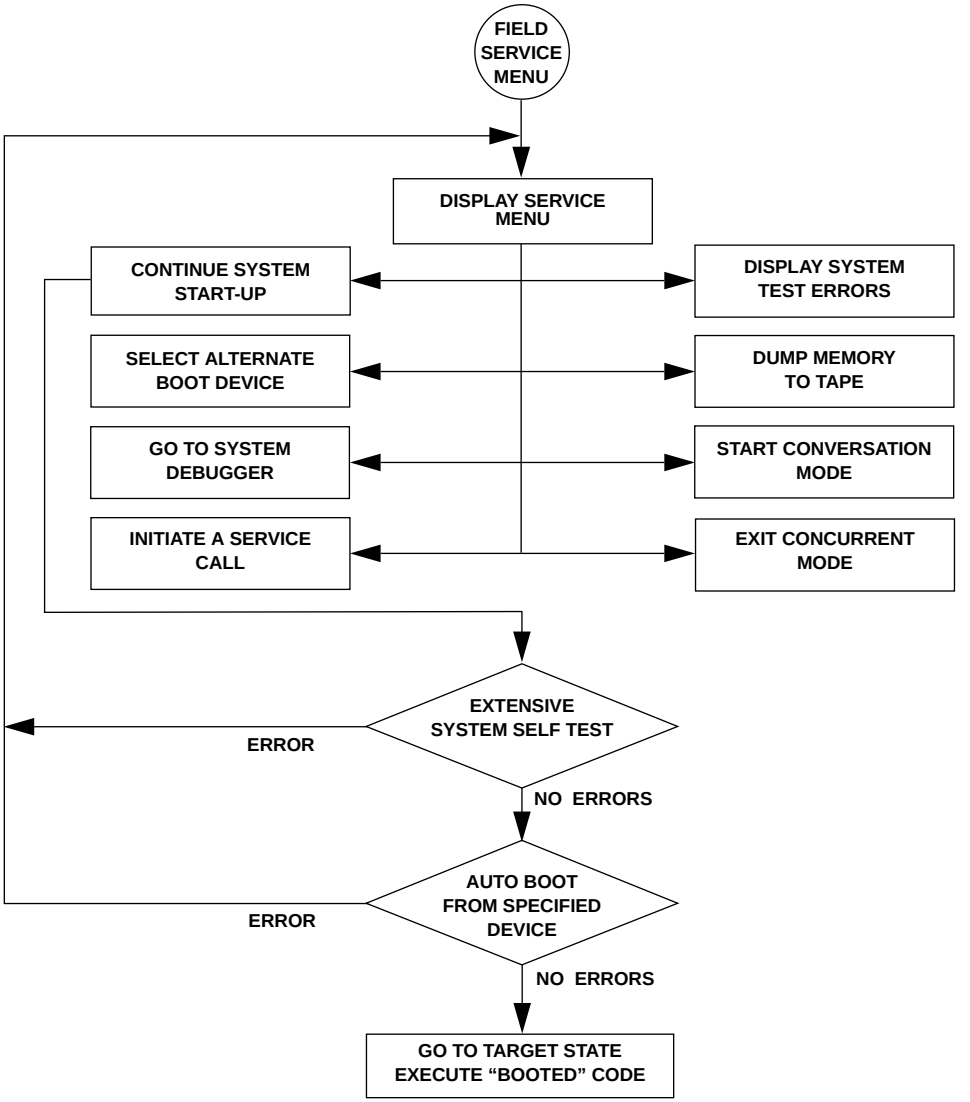


**Figure A-1. Flow Diagram of 197Bug Field Service Menu Operational Mode**

Error message explanations for the extended confidence test are given in the heading for the failed test.

# Menu Details

Following are more detailed descriptions of the menu selections.

## Continue System Start Up

The only action required by the user is to enter a **1** followed by a carriage return. The system then continues the start-up process by performing extended confidence testing (**ST**) followed by a system boot from the device specified by the **ENV** parameters active.

## Select Alternate Boot Device

After entering a **2**, the user is prompted with:

```
Enter Alternate Boot Device:
Controller:
Drive     :
File      :.
```

The selection of devices supported by the 197Bug is listed in Appendix E. Entry of a selected device followed by a carriage return redisplays the menu for another selection, normally "continue system start-up" at this point.

## Go to System Debugger

When **3** is selected, this entry places the user in 197Bug diagnostic mode, if system mode is selected, indicated by the prompt 197-Diag>. If desired, return to the menu can be accomplished by typing "**menu**" when the Bug prompt appears. If not in system mode, the 197-Bug> prompt will be displayed and the user must type "**SD**" to enable diagnostic execution. When in 197-Diag mode, operation is defined by the *MVME197BUG 197Bug Diagnostic Firmware User's Guide*.

## Initiate Service Call

The initiate service call function (enter **4**) is described in the following sections.

### General Flow

Initiated by typing a **4 <CR>** in response to the menu prompt, this function is normally used to complete a connection to a service organization which can then use the "dual console" mode of operation to assist a customer with a problem. Interaction with the service call function proceeds as follows:

First, the system asks:

```
Modem Type:
0) Terminal
1) Manual
2) UDS-2122662
3) UDS-2122980 (Hayes)
4) UDS-2123382 (Hayes)
Your Selection ( )?.
```

Explanation:

UDS means that the modem is compatible with the UDS modem protocol as used in internal Delta Series modems. The model number of this modem is UDS 2122662.

Hayes means that the modem is compatible with a minimal subset of the Hayes modem protocol. This minimum subset is chosen to address the broadest spectrum of Hayes compatible modem products. Note that the modem itself is not tested when Hayes protocol is chosen, while the modem is tested with the UDS protocol choice.

Manual mode connects directly to the modem in an ASCII terminal mode, allowing any nonstandard protocol modem to be used.

Terminal mode is used to connect any ASCII terminal in place of a modem, via a null modem, or equivalent cable. It is useful in certain trouble-shooting applications for providing a slave terminal without the necessity of dialing through a modem.

When a selection of one of the above options is made (option 0 in this case), the system asks:

```
Do you want to change the baud rate from 1200 (Y/N)?
```

Note that any question requiring a **Y** or **N** answer defaults to the response listed furthest to the right in the line (i.e., a question with Y/N defaults to NO if only a carriage return is entered. If the user answers **Y** to the baud rate question, the system prompts:

```
Baud rate [300, 1200, 2400, 4800, 9600] 1200?
```

## E

## F

## M



INDEX

## N

## S

## T