EMUL68-PC™

User Guide

© ICE Technology

Contents

Product Notes

European CE Requirements

Nohau has included the following information in order to comply with European CE requirements.

User Responsibility

The in-circuit debugger application, as well as all other unprotected circuits need special mitigation to ensure Electromagnetic Compatibility (EMC).

The user has the responsibility to take required measures in the environment to prevent other activities from disturbances from the debugger application according to the user and installation manual.

If the debugger is used in an environment other than the intended (for example, field service applications), it is the user's responsibility to control that other activities cannot be disturbed in such a way that there may be risk for personal hazard/injuries.

Special Measures for Electromagnetic Emission Requirements

To reduce the disturbances to meet conducted emission requirements it is necessary to place a ground plane on the table under the pod cable and the connected processor board. The ground plane shall have a low impedance ground connection to the host computer frame. The insulation sheet between the ground plane and circuit boards shall not exceed 1mm of thickness.

System Requirements

CAUTION

Like all Windows applications, the Seehau software requires a minimum amount of free operating system resources. The recommended amount is at least 40%. (This is only a guideline. This percentage might vary depending on your PC.) If your resources are dangerously low, Seehau might become slow, unresponsive or even unstable. If you encounter any of these conditions, check your free resources. If they are below 40%, reboot and limit the number of concurrently running applications. If you are unable to free more than 40% operating system resources, contact your system administrator or Nohau Technical Support.

The following are minimum system requirements:

- Pentium 200 (Pentium II or faster is recommended)
- Single-Processor System
- Windows 95, 98, NT, or 2000
- Random Access Memory (RAM)
  - For Windows 95/98: 32 MB
  - For Windows NT/2000: 64 MB

Warnings

If E-Clock goes out to the target system in monitor mode, data might accidentally be written to a memory or peripheral in the target system.

To avoid damage to the pod or to your target, do not connect the pod to your target when the pod or target power is on

When powering up, always power up the emulator first followed by the target system. When powering down, power down the target system first followed by the emulator. Failing to do so can cause damage to your target and/or emulator.

Do not apply power to your system unless you are absolutely sure the target adapter is correctly oriented. Failing to do so can cause damage to your target and/or emulator.

About This Guide

The *EMUL68–PC User Guide* describes how to use the EMUL68–PC emulation system with the Seehau graphical user interface. If you are using the Windows or DOS versions of the EMUL68–PC, however, this guide also provides help with those specific products.

This *EMUL68–PC User Guide* is intended for both novice and advanced users.

Downloading EMUL68–PC Product Documentation

To download an electronic version of this guide, do the following:

1. Open the Nohau home page at www.icetech.com.
2. Click **USA**.
3. Click **Documents**.
4. Click **Nohau Manuals**.
5. Scroll down to EMUL68–PC. Then select **EMUL68–PC User Guide** to download a PDF version of the guide.

1 Overview of the EMUL68–PC Emulator System

The basic hardware for the EMUL68–PC emulator system includes the following:

- Emulator board—plugs into a an ISA slot inside the PC or High Speed Parallel (HSP) Box.
- Pod board—device-dependent board that replaces the microcontroller chip on the target system.
- Trace board (optional)—plugs into an ISA slot inside the PC or HSP and connects to the emulator board through a short ribbon cable.
- Five foot ribbon cable—connects the emulator and pod (see Figure 1).
- Target adapter—allows you to connect the pod board to your target system.

Two system configurations are available:

- High-Speed Parallel Box (HSP) connects to the parallel printer port. See the following “High-Speed Parallel Box (HSP)” section.
- PC Plug-In. See the following “PC Plug-In” section.

For information about configuring and installing your hardware, refer to Chapter 2, “Installing the Hardware” in this guide.

High-Speed Parallel Box (HSP)

In the HSP chassis, the emulator board and optional trace board are installed in an external box that connects to the PC’s parallel printer port. The HSP option makes the emulator system portable, allowing you to move the system from one PC to another. You can also use this option with a portable laptop computer. The HSP eliminates the need for the two ISA bus slots.

Figure 1. HSP Box Connected to the Emulator, Trace, and Pod Board

PC Plug-In

The emulator ISA board is plugged into an ISA slot in your PC and is connected with a five-foot cable to a device-dependent pod board. The optional ISA trace board is plugged into the PC and connects to the emulator board through a ribbon cable.

You can custom configure the emulator hardware to your requirements with various jumpers. See Chapter 2, “Installing the Hardware,” for details about jumper configuration.

User Interface

Seehau is a high-level language user interface that allows you to perform many useful tasks, for example:

- Load, run, single-step and stop programs based on C or Assembly code.
- Set triggers and view trace (with optional trace board).
- Modify and view memory contents including Special Function Registers (SFRs).
- Set hardware breakpoints.
- Analyze code with Program Performance Analysis (PPA)

The emulator is configured and operated by the Seehau user interface.

Important Product Notes

Special Emulation Behavior When Using the Reset and Go Command

When your power-on reset routine writes to any of the registers that must be written in the first 64 cycles after reset, (such as the INIT and Option registers), you must start your program with the Reset and Go command rather than two separate commands (Reset Chip and then Go). This does not apply in test mode since the 64-cycle limit is disabled in this mode.

You can also activate the Reset and Go command by pressing Ctrl G. This command causes the following actions to occur.

- Emulation mode becomes active, trace starts, and reset is driven low by the emulator on the same bus cycle.
- Reset is held active (low) for approximately 1/5 second and then released.
- When the HC11 is reset, the MCU fetches the reset vector and branches to the beginning of the user code.

Note

Bus cycles that occur while reset is active are recorded in the trace buffer and counted by the execution cycle counter. Therefore, the first time you stop execution after Reset and Go, the cycle counter will be meaningless.

Quick Start for Installing Your Emulator System

The following illustration shows the major steps for installing and configuring the EMUL68–PC. For details, refer to the chapter referenced in each step.

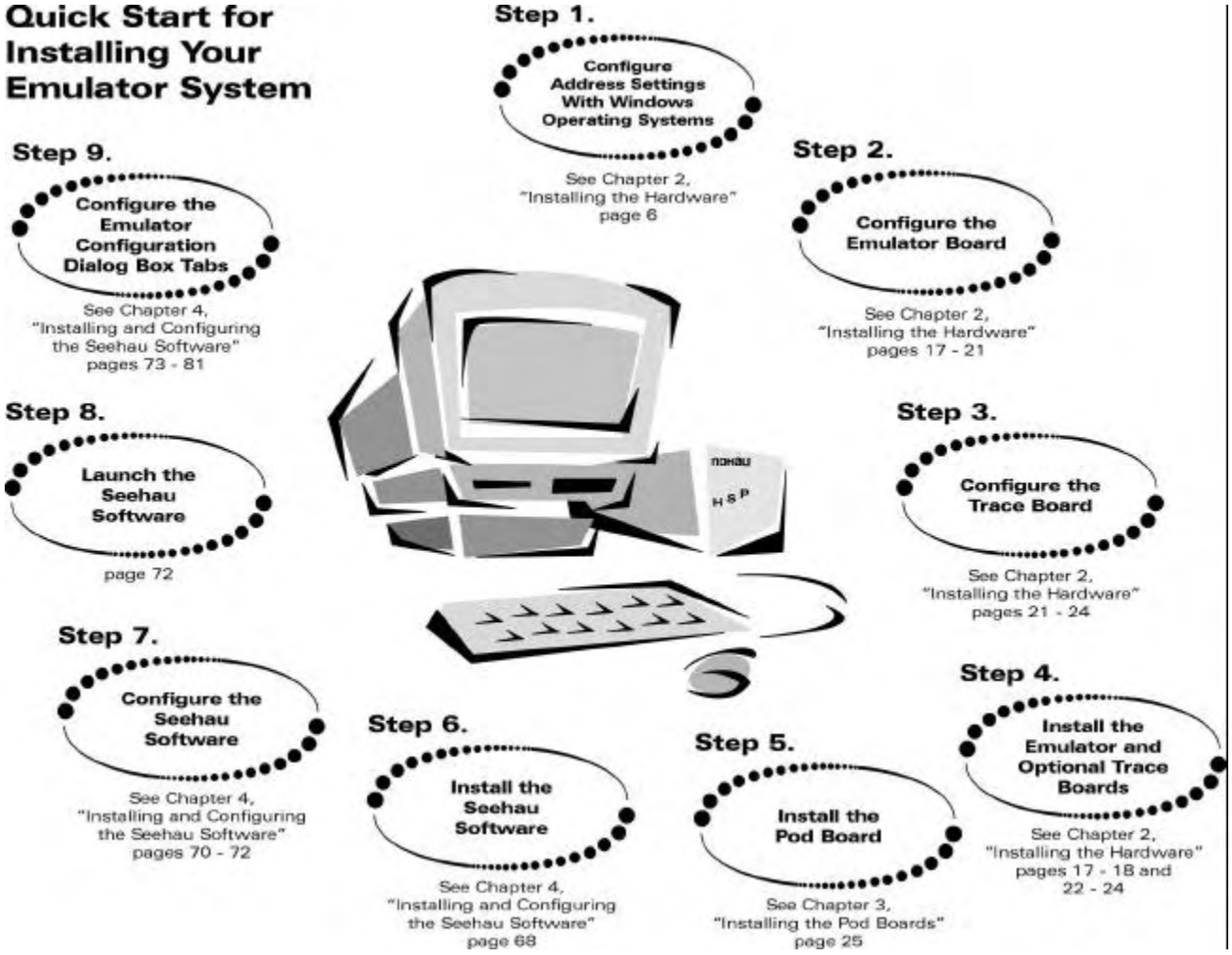

Figure 2. Steps for Installing and Configuring the EMUL68–PC and Seehau Software

2 Installing the Hardware

Quick Start for Installing the Hardware

Figure 3 shows the major steps for installing the EMUL68–PC hardware.

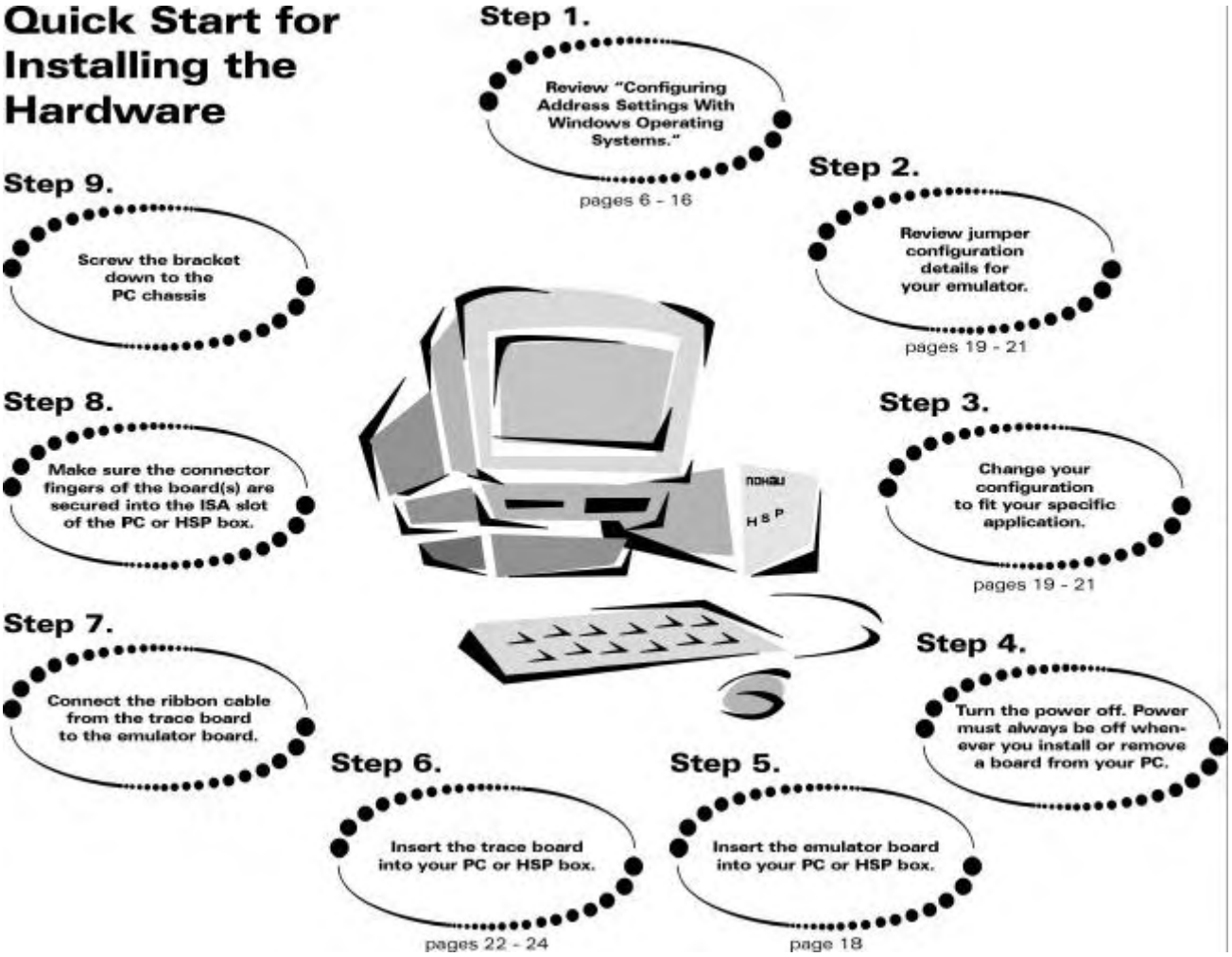

Figure 3. Steps for Installing the EMUL68–PC Hardware

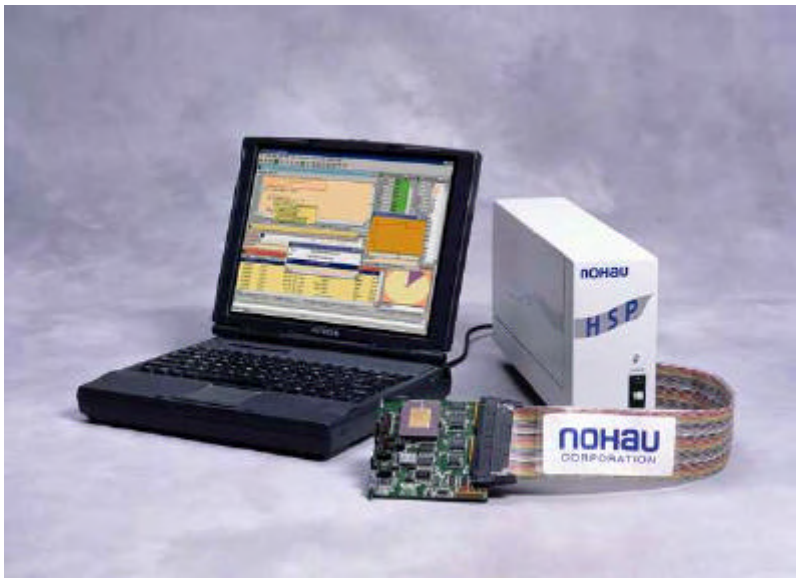

Figure 4. HSP Box Connected to the Emulator, Trace, and Pod Boards

Configuring Address Settings With Windows Operating Systems

The following applies to all Windows operating systems:

- Default Address Ranges:
  - Emulator Board: 120H to 12FH
  - Standard Trace Board: 130H to 13FH
  - Enhanced Trace Board: 100H to 11FH
- Default Address Settings for the HSP Box:

  No address conflict is possible when installing the HSP box with any Windows operating system. Use the default address ranges (listed above).

  Skip to “Installing Emulator Boards” later in this chapter.

Configuring Address Settings for the Emulator and Optional Trace Board

The following sections provide details about configuring address settings for the emulator and optional trace board for each Windows operating system. Refer to the section that covers your specific operating system.

Known Device Driver Conflicts

Nohau is aware of potential device driver conflicts with certain network cards running on Novell/Netware networks. Problems have been reported with both 3COM ISA network cards and some Novell network cards. Most of these problems have been experienced when running Windows NT or Windows 2000 operating systems.

Possible Symptoms

- When starting Seehau, communication with the network stops. (You will be unable to access resources on the network.)
- Seehau will not start.

A possible solution might be to change your network card. Nohau Support has not tested all network cards, although some customers have reported that the following network cards have resolved this conflict:

- Intel Ether Express Pro 10/100 ISA
- 3COM Etherlink III (905B or later) 10/100 PCI
- Bay Networks NetGear FA310TX 10/100 PCI

Configuring Address Settings With Windows 95/98

Checking Your PC for Default Address Conflicts

1. Click the **Start** menu, and select **Settings**.
2. Click **Control Panel**.
3. Double-click **System**. The **Systems Properties** dialog box opens.
4. Click the **Device Manager** tab.
5. Click **Properties**.
6. Click **Input/Output**. Scroll the contents of the window to verify that no device is listed within that range.

Alternative Addressing

If you see a device present in the default address range for your emulator or trace board, do the following:

1. Beginning at the address 100H, scroll down to look for an unused address range:
   - 10H for any emulator board
   - 10H for the Standard Trace Board
   - 20H for the Enhanced Trace Board.
2. When you locate an unused address range, make a note of the base address of the range for use when configuring Seehau.
3. Refer to Appendix G, “Address Examples” to re-configure the base address of your board.

   The base address must be an even multiple of 10 (such as 200 or 210). If you have to change the address of the emulator or trace board, be sure to change both the board jumpers and the jumper settings in the software.

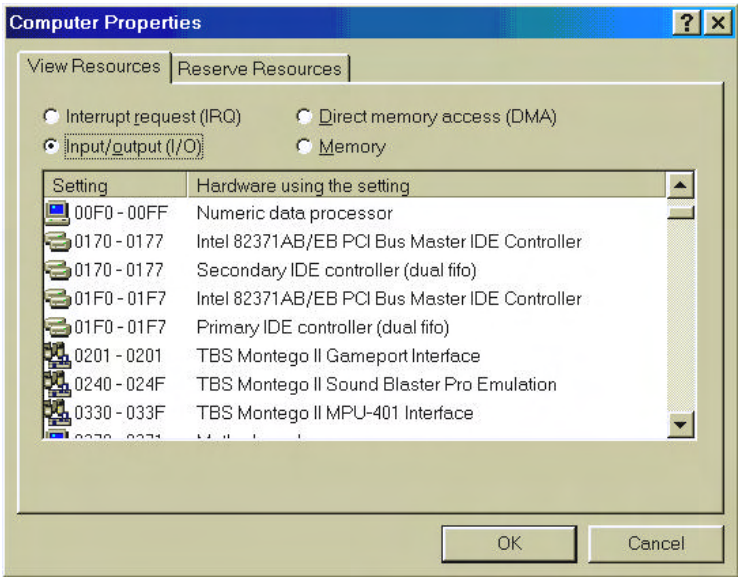

Figure 5. System I/O Resources

Configuring Address Settings With Windows NT

- First check your administrative privileges.
- Then check your PC for default address conflicts.

Checking Administrative Privileges

1. Click the **Start** menu, and select **Programs**.
2. Select **Administrative Tools**, and click **User Manager**. The **User Manager** dialog box opens (Figure 6).
3. In the bottom half of the dialog box, double-click **Administrators**. The **Local Group Properties** dialog box opens displaying a list of login names (Figure 7).

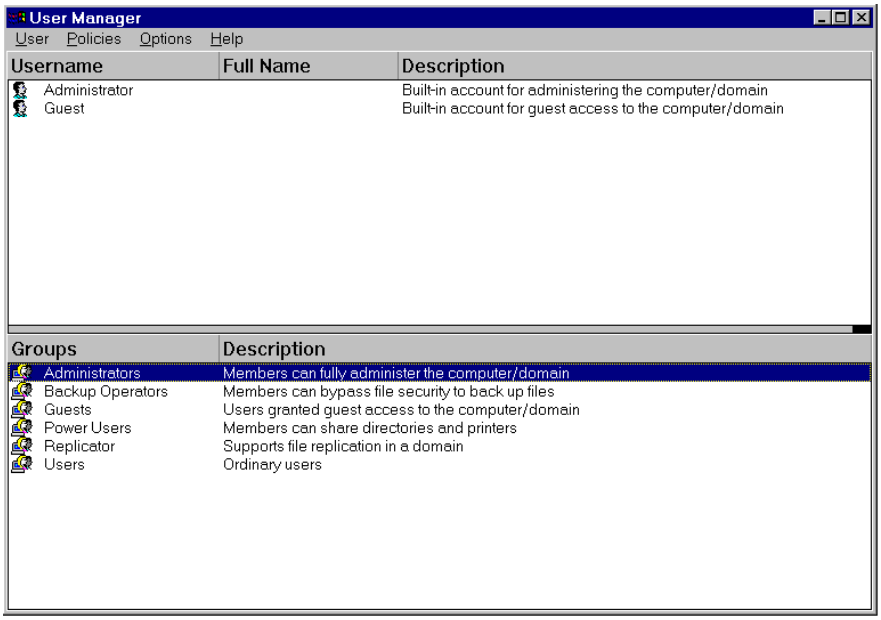

Figure 6. User Manager Dialog Box for Windows NT

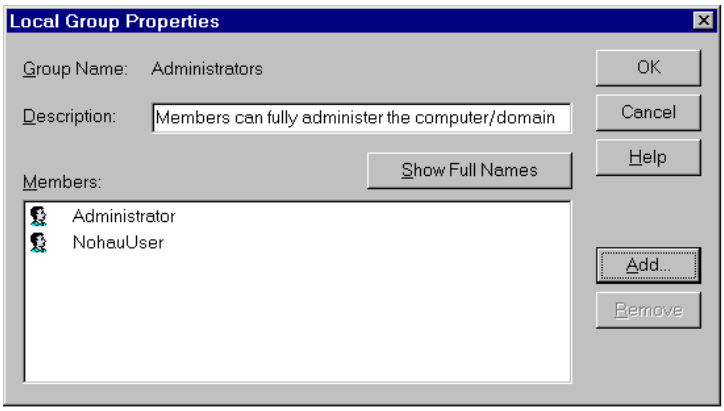

Figure 7. Local Group Properties Dialog Box for Windows NT

4. Look for your login name in the list of names. If your login name is not present, you are not set up with administrative privileges. Contact your System Administrator to update your privileges or give you the administrator's password.

Checking Your PC for Default Address Conflicts

1. Click the **Start** menu, and select **Programs**.
2. Select **Administrative Tools**, and click **Windows NT Diagnostics**. The Windows NT Diagnostics window opens (Figure 8).
3. Click the **Resources** tab.
4. Click **I/O Port**.
5. Check the I/O resources listed to verify that no device appears in a default address range.

Alternative Addressing

If you see a device present in the default range for your emulator or trace board, do the following:

1. Beginning at the address 100H, scroll down to look for an unused address range:
   - 10H for any emulator board
   - 10H for the Standard Trace Board
   - 20H for the Enhanced Trace Board.

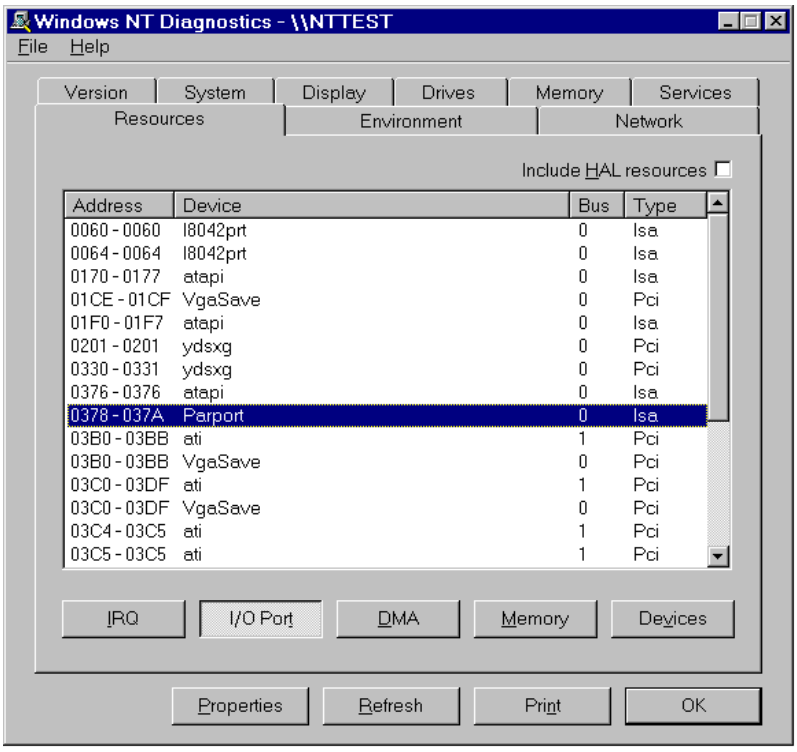

Figure 8. NT Diagnostics Window

2. When you locate an unused address range, make a note of the base address of the range for use when configuring Seehau.

3. Refer to Appendix G, “Address Examples” to re-configure the base address of your board.

Driver Troubleshooting

For details, see Chapter 8, “Troubleshooting.”

- If you get a **Service or driver failed** error message when rebooting, you probably have a resource conflict.

- If you get a **create file failed** error message upon execution, the device driver did not properly start.

NohauHC11 Device Driver With Windows NT

After installation, Windows NT Diagnostics will show the Nohau11 device driver present in the upper I/O range (FFxx). After launching Seehau, the driver is reassigned to the actual address ranges. In the Control Panel Devices window (Figure 9), you will see three columns: Device, Status, and Startup.

- Device: lists the Nohau device driver
- Status: displays **Started**
- Startup: displays **Automatic**

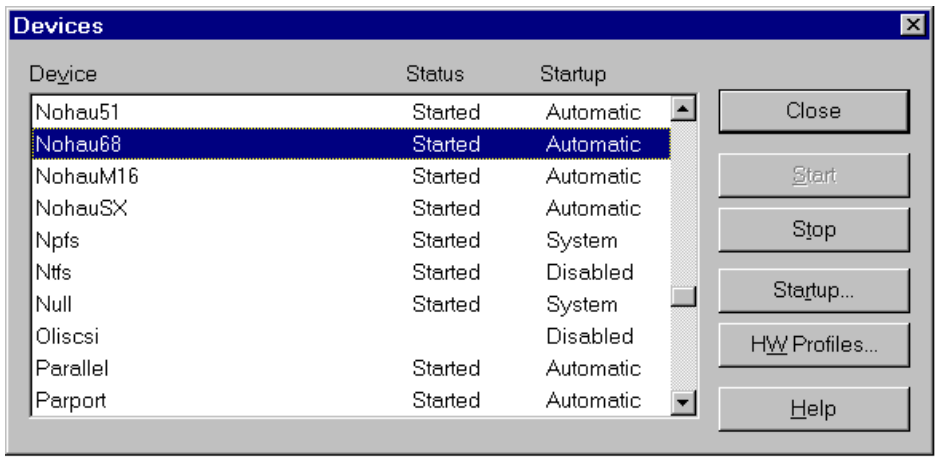

Figure 9. Control Panel Devices Window

Configuring Address Settings With Windows 2000

- First check your administrative privileges.
- Then check your PC for default address conflicts.

Checking Administrative Privileges

1. Click the **Start** menu, and select **Settings**. Click **Control Panel**.

2. From the **Control Panel**, double-click **Users and Passwords**. The Users and Passwords window opens (Figure 10).

3. Click the **Advanced** tab. Now click the **Advanced** button. The Local Users and Groups window opens (Figure 11).

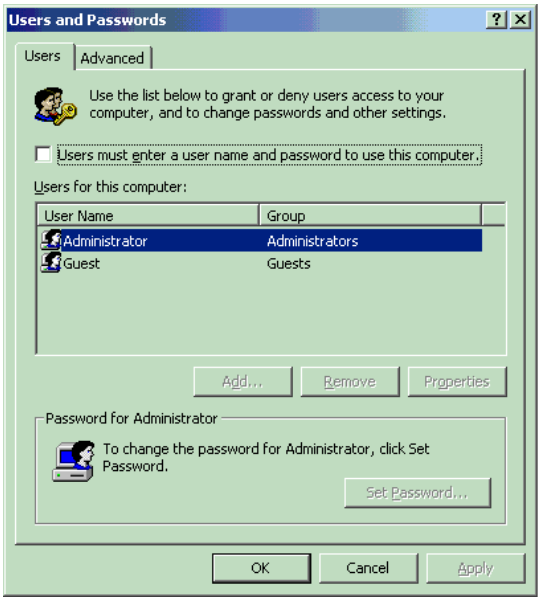

Figure 10. Users and Passwords Window

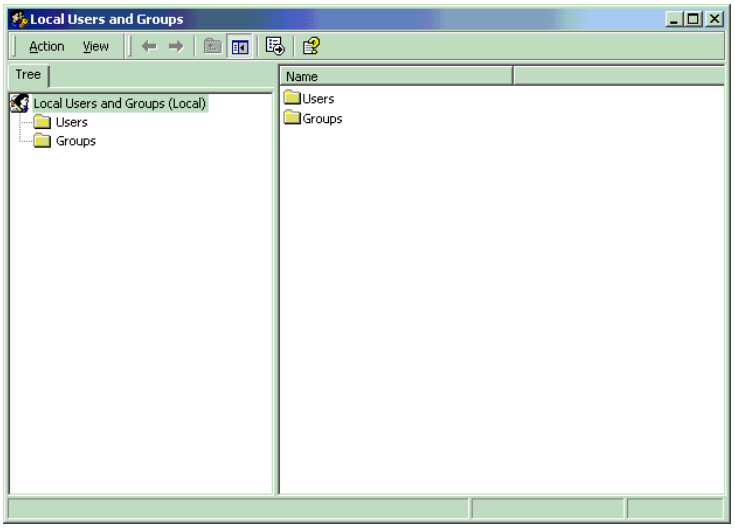

Figure 11. Local Users and Groups Window

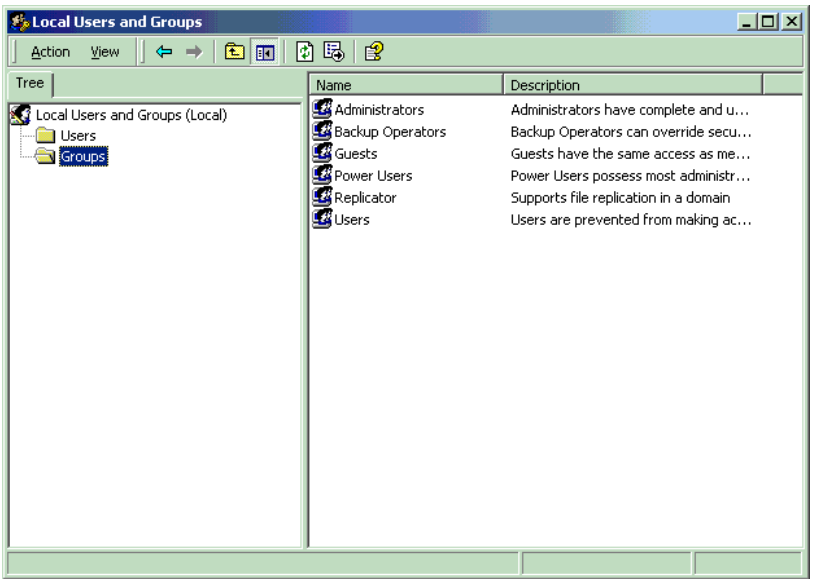

Figure 12. Local Users and Groups Window with Groups Folder

4. Click the Groups folder located in the left region of the window beneath Local Users and Groups.

5. Double-click the Groups folder. A list of groups appears in the right region of the window (Figure 12).

6. Double-click **Administrators**. Your user name should be listed.

> **Note**
>
> If you are not an administrator, ask your System Administrator to add you to this list.

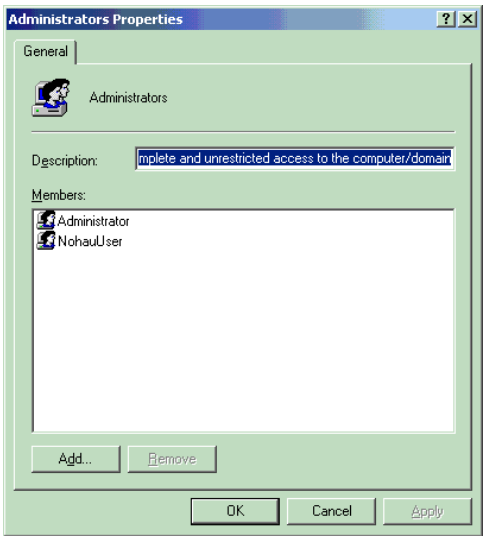

Figure 13. Administrator Dialog Box

Checking Your PC for Default Address Conflicts

1. Right-click the My Computer icon on your desktop, and select **Properties**. The System Properties window opens (Figure 14).

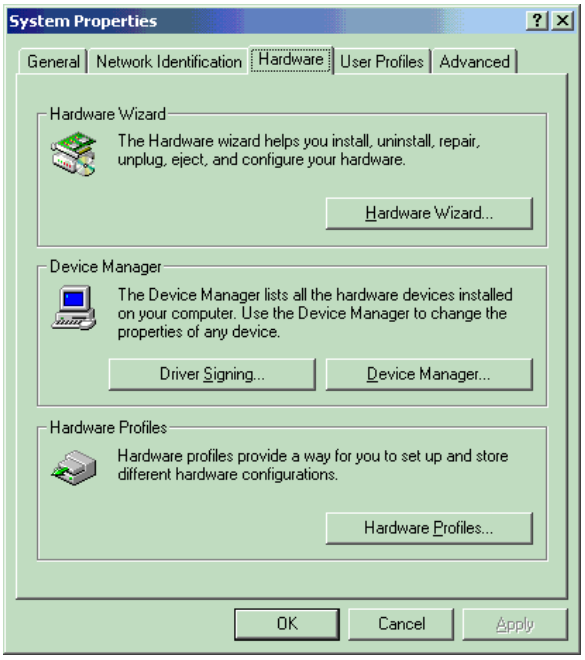

Figure 14. System Properties Window

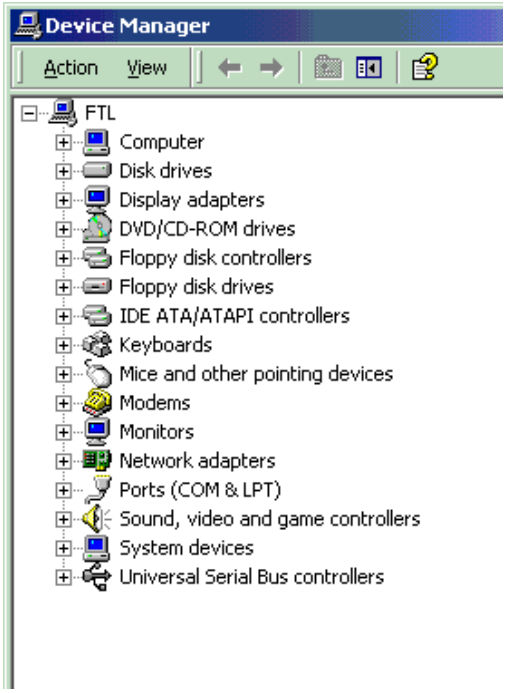

Figure 15. Device Manager Window

2. Click the **Hardware** tab. Then click **Device Manager**. The Device Manager window opens (Figure 15).

3. In the Device Manager window, select the **View** menu. Then click **Resources by Type**. A window opens that shows the system resources (Figure 16).

4. Double-click **Input/Output (I/O)**.

5. Check the I/O resources listed to verify that no device appears in the default address ranges.

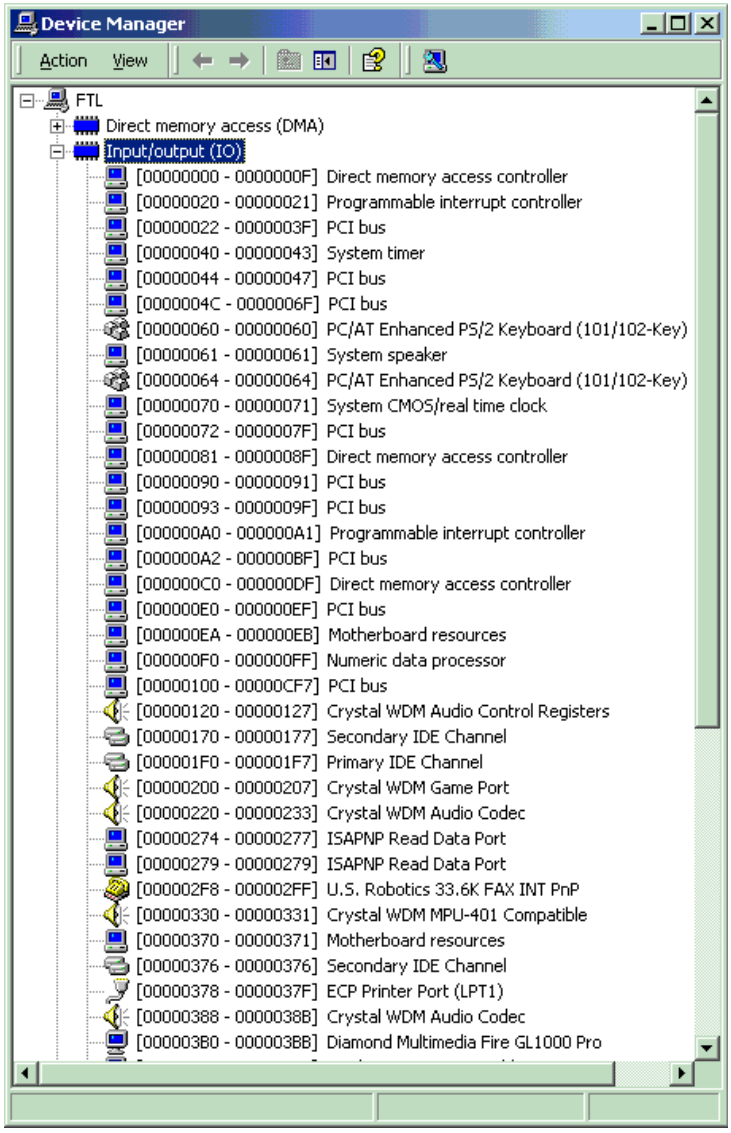

Figure 16. System Resources

Alternative Addressing

If you see a device present in the default address range for your emulator or trace board, do the following:

1. Beginning at the address 100H, scroll down to look for an unused address range:
   - 10H for any emulator board
   - 10H for the Standard Trace Board
   - 20H for the Enhanced Trace Board.
2. When you locate an unused address range, make a note of the base address of the range for use when configuring Seehau.
3. Refer to Appendix G, "Address Examples" to re-configure the base address of your board.

Driver Troubleshooting

For details, see Chapter 8, "Troubleshooting."

- If you get a **Service or driver failed** error message when rebooting, you probably have a resource conflict.
- If you get a **create file failed** error message upon execution, the device driver did not properly start. Review the steps in this section again. You can use Windows 2000 System Properties to re-check that your port address has no conflicts.

Nohau11 Device Driver With Windows 2000

To verify that the Nohau11 device driver is properly installed, do the following:

1. From the **Start** menu, select **Programs**. Select **Accessories**, then **System Tools**.
2. Click **System Information**. The System Information window opens.
3. Double-click the Software Environment folder.
4. Double-click the Drivers folder. A list of active drivers appears (Figure 17). Refer to the **Name** column and scroll down to **nohau11**.
5. Verify the driver is running. In the **State** column, you should see the word **Running**. In the **Status** column, you should see **OK**.

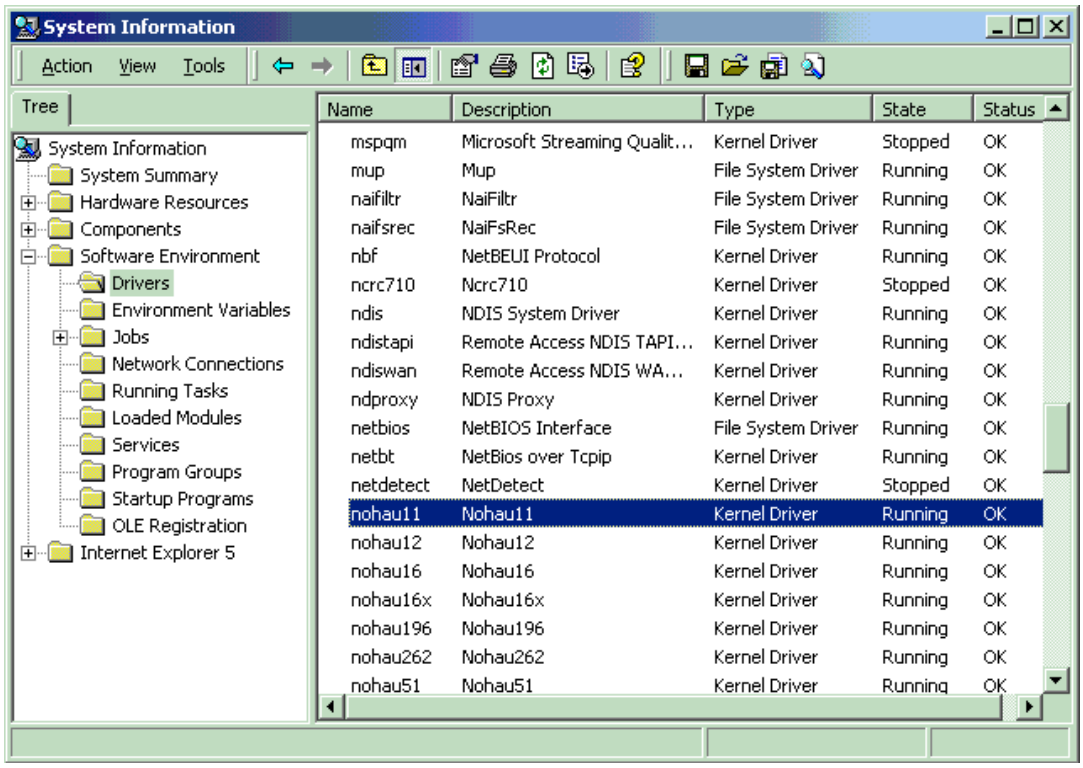

Figure 17. List of Active Drivers

Installing Emulator Boards

This section provides installation details for the following emulator boards: standard, 256/512 bank switch, and the EB1M bank switch.

Note

Depending on which MCU your pod uses, the Special Function Register (SFR) block can be located at either address 0 or at address 1000. The emulator must be configured for the correct address in order to shadow the INIT register which tracks movement of the SFR block and internal RAM. For more information, refer to your emulator board's SFR base address jumper designation.

If the SFR base address jumper(s) are incorrectly configured, you will get the following error message:
INIT shadow does not match INIT register.

After you have inspected the boards for any damage, you can install the emulator board in your PC or HSP by doing the following:

1. Configure your emulator board jumpers. In this chapter, refer to the appropriate jumper description for your board.

2. Turn the power off. (Power must always be off when you plug in any PC board.)

3. Plug the emulator board into the ISA slot in your PC or HSP box (if not already installed).

4. Connect the long ribbon cable to the connector on the emulator board located at the back of the PC or HSP box. The connector has a notch on it, so you cannot insert it the wrong way. (The connectors of the ribbon cable are identical so it does not matter which end is connected to the pod or the emulator board.)

5. Connect the other end of the ribbon cable from the emulator board to the pod. Close the locks on the connector over the cable.

6. Screw the bracket down to the PC chassis.

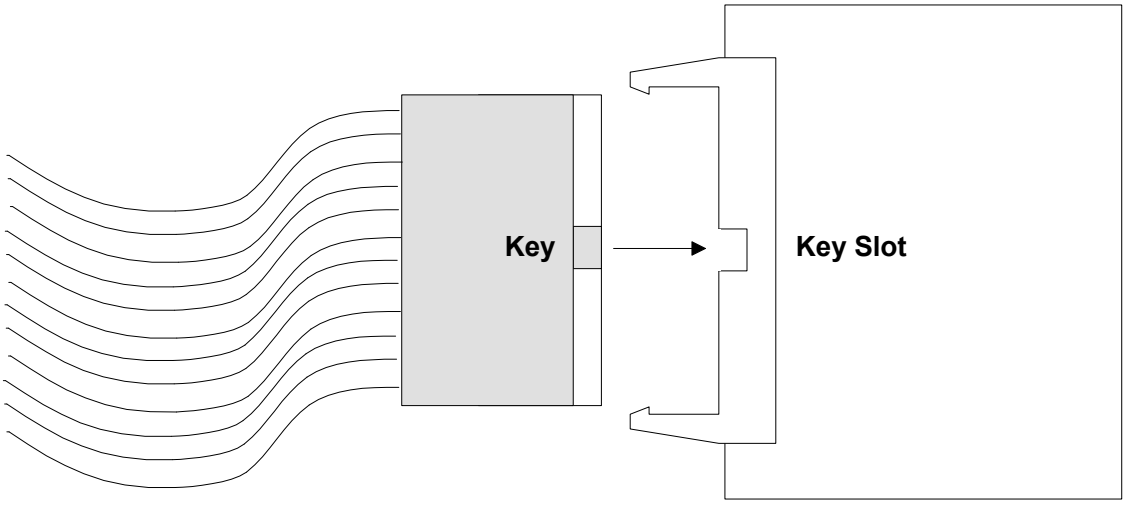

Figure 18. Connecting the Emulator to Your Pod Board

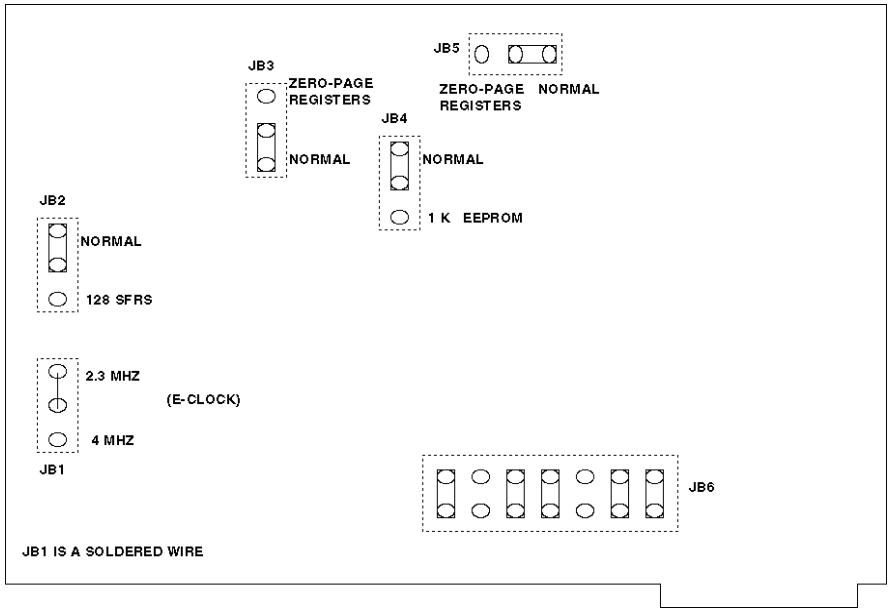

Figure 19. Standard Emulator Board

Standard Emulator Boards

> **Note**
>
> For emulator board addressing examples, refer to Appendix G. Address Examples.

Standard Emulator Board Description

Standard Emulator Board Jumpers

Jumper Designation	Description
JB1 (E-Clock)	Cannot be modified by the user. • Factory wired in the 2.3-MHz position for 2.1-MHz and 3.3-MHz emulators. • Factory wired in the 4.0-MHz position for 4.0-MHz emulators.
JB2 (SFR block size)	• Set to normal position for up to 128 byte SFR block. • Set to 128 position for greater than 128 byte SFR block.
JB3/JB5 (SFR base address)	Default value is set to normal. • Set to normal position for MCUs with SFR block starting at 1000H. • Set to zero-page position for MCUs with SFR block starting at 0H. The SFR block on the following pods starts at 0H: POD–11Dx, POD–11Kx, POD–11Px, POD–11FL.
JB4 (EEPROM)	• Set to normal position for up to 1K of EEPROM. • Set to 1K position for greater than 1K of EEPROM.
JB6 (board address)	Used to set the address of the emulator board.

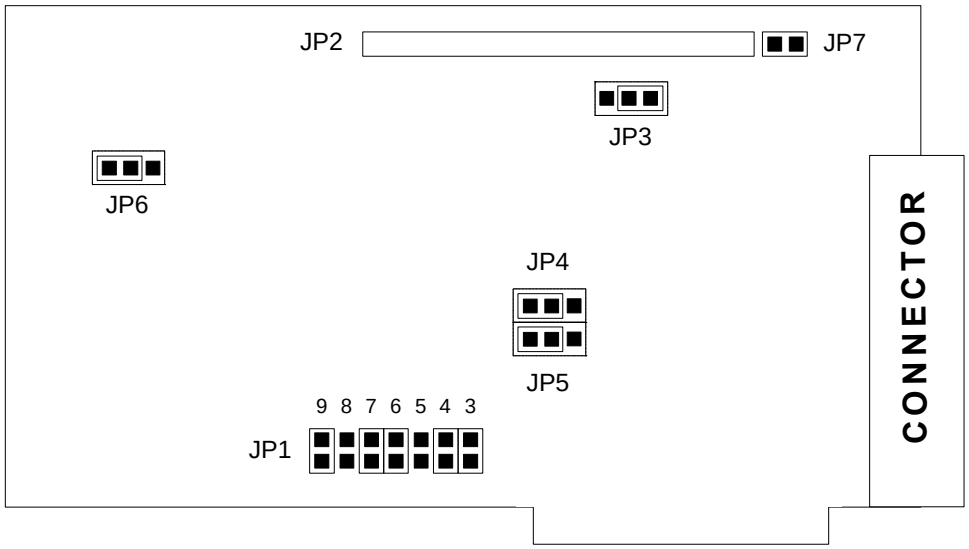

Figure 20. EB 256/512 Bank Switch Emulator Board

Installing the EB 256/512 Bank Switch Emulator Board Jumpers

Bank Switch Jumpers

Jumper Designation	Description
JP1 (board address)	Used to set the address of the emulator board.
JP3/JP4 (SFR base address)	• Set in the position shown in Figure 20 (normal) for MCUs with the SFR block starting at 1000H. • Set in the opposite position as shown in Figure 20 for MCUs with the SFR block starting at address zero. The SFR block on the following pods starts at 0H: POD–11Dx, POD–11Kx, POD–11Px, POD–11FL.
JP5 (EEPROM)	EEPROM size: • Set in the position shown in Figure 20 for processors with EEPROM of 2K or smaller. • Set in the opposite position if bigger than 2K.
JB6 (emulation memory size)	Emulation memory size is pre-set at the factory: • The jumper should be factory set in the position shown in Figure 20 if you have 256K of emulation memory. • The jumper should be factory set in the opposite position if you have 512K of emulation memory.
JP7	• In if you do not have a trace board. • Out if you have a trace board.

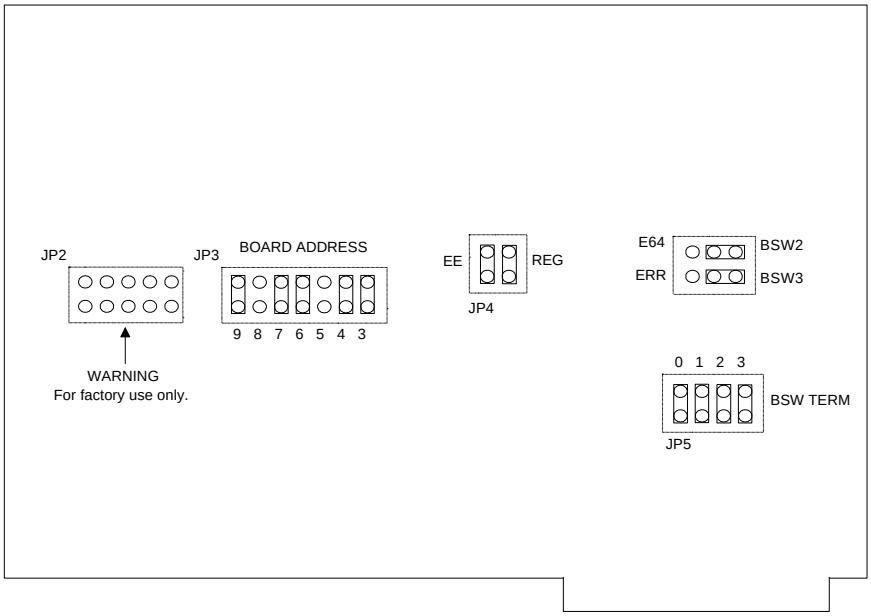

Figure 21. EB1M Emulator Board

Installing the EB1M Emulator Board

Jumper Designation	Description
JP3 (board address)	Used to set the address of the emulator board.
JP4 (REG=SFR base address; EE=EEPROM size)	Install the Reg jumper for the following pods: POD–11Dx, POD–11Kx, POD–11Px, POD–11FL • EE out for 4K EEPROM • EE in for all others
E64/BSW0	• In BSW0 position for pods with BSW0. (POD–11KE, POD–11FL0) • In E64 position for pods without BSW0. (For all other pods)
ERR/BSW1	• In BSW1 position for pods with BSW1. (POD–11KE, POD–11FL0) • In ERR position for pods without BSW1. (For all other pods)
BSW TERM	• In for pods with BSW signals buffered. (POD–11KE, POD–11FL0) • Out to connect external BSW signals direct. (For all other pods)

Installing and Configuring Standard Trace Boards

Trace is an optional part of an emulator system that allows you to perform more advanced debugging. The standard trace board is a full-length card that can occupy any ISA slot as long as the ribbon cable can reach from the emulator card to the trace card.

The standard trace board includes 48 bits of RAM for each trace record and can hold either 4096 or 16384 records.

I/O Address

Installing the Standard Trace Board with an HSP Box

If you purchased the HSP box, you can use the default address (130H) regardless of the I/O addresses used by the computer. Skip to the “Steps for Installing the Standard Trace Board” section.

Installing the Standard Trace Board with a PC

The trace board address jumpers have been factory preset to 130H. If your system currently uses location 130H, you must find an alternate address location and make appropriate changes to the jumpers and software.

Alternate Addressing Examples

Figure 22 shows several examples of how to set the jumpers on header J2. The jumper columns below are arranged the same as board jumpers. For a complete list of addressing examples, refer to Appendix G, “Address Examples.”

Steps for Installing the Standard Trace Board

Be sure to inspect the boards for any damage. Then with the address jumpers in place, you can install the trace board in your PC or HSP by doing the following:

1. Turn the power off. Power must always be off when you plug in any PC board.
2. Insert the trace board with the short ribbon cable connected to it.
3. Connect the ribbon cable to the emulator board. Make sure that both rows of the trace cable are connected to and properly aligned with the trace header on the emulator board.

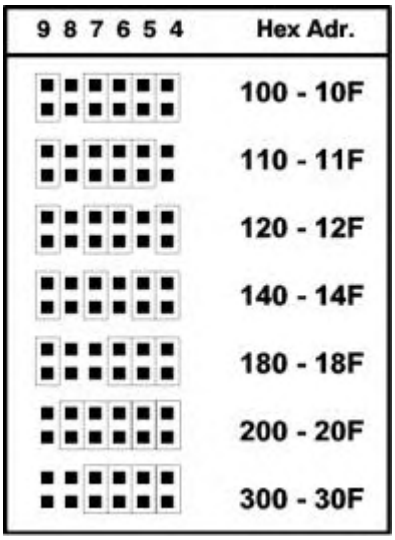

Figure 22. Alternate Standard Trace Addressing Examples

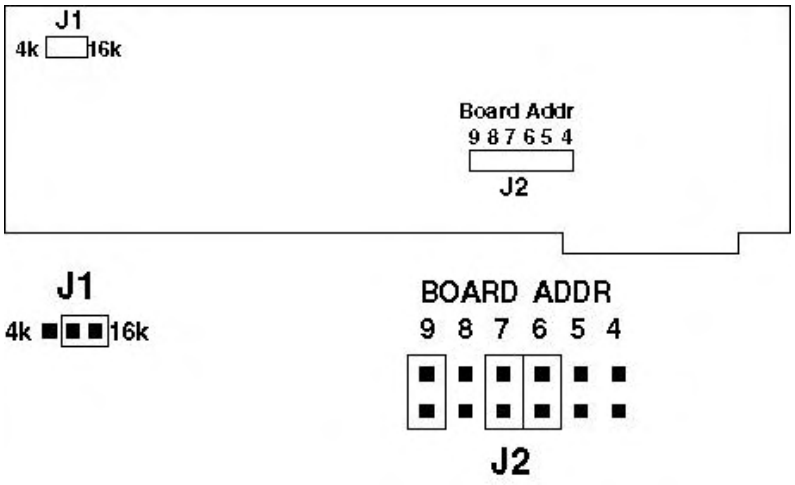

Figure 23. Standard Trace Board with 16K Installed, Address 130H

4. Check to ensure the connector fingers of the board(s) are secured into the female connector of the PC's motherboard.

5. Screw the bracket down to the PC chassis.

Installing and Configuring Enhanced Trace Boards

The enhanced trace board includes 64 bits of RAM for each trace record and can hold either 64K or 256K records.

Addressing Examples

Figure 24 shows examples of how to set the jumpers on header JP1. Jumper columns are arranged in the same order in which the jumpers are physically located on the board. For a complete list of addressing examples, refer to Appendix G, "Address Examples."

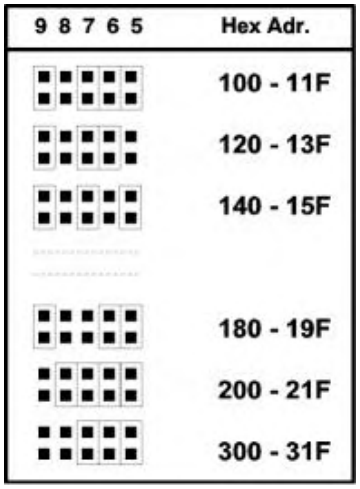

Figure 24. Enhanced Trace Addressing Example

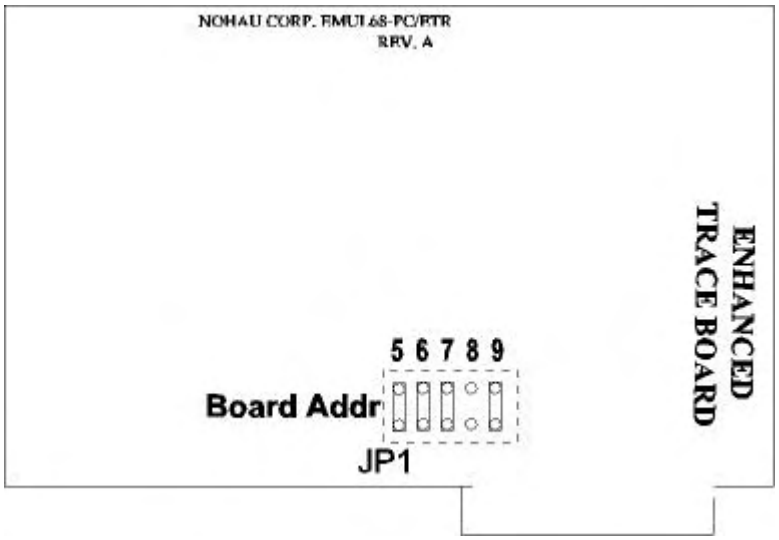

Figure 25. 64K/256K Enhanced Trace Board Jumpers and Headers (not to scale)

Factory Settings

Board jumpers have been preset at the factory to 100H prior to shipment. Compare the jumper configurations on the boards you receive against the configurations described in Figure 25. If a jumper is installed other than as shown, refer to these figures before changing the jumper's position.

Steps for Installing the Enhanced Trace Board

Be sure to inspect the boards for any damage. Then with the address jumpers in place, you can install the trace board in your PC or HSP by doing the following:

1. Turn the power off. Power must always be off when you plug in any PC board.
2. Insert the trace board with the short ribbon cable connected to it.
3. Connect the ribbon cable to the emulator board. Make sure that both rows of the trace cable are connected to and properly aligned with the trace header on the emulator board.
4. Check to ensure the connector fingers of the board(s) are secured into the female connector of the PC's motherboard.
5. Screw the bracket down to the PC chassis.

3 Installing the Pod Boards

This chapter provides detailed information on 10 pod board types, including:

Pod Types

- 11DE
- 11DS
- 11E
- 11FE
- 11FL0
- 11KE
- 11KS
- 11PE
- 11PS
- 11S

After selecting a pod type, you will need to set up the various pod board jumpers. Refer to the section in this chapter that provides details for your pod board type including: board layout illustrations, diagrams of jumper locations, and tables describing jumper configuration options.

How this Chapter is Organized

Pod types are listed in alphabetical order. Each pod section presents information in the following format:

- **Configuration Options**: Describes configuration options.
- **Illustrations**: Shows various configurations for switches and jumpers, target power and internal crystal. The illustrations throughout this chapter are representative of the pod board layout. The notations used in the illustrations might not match the silk screens on the boards.
- **Special Considerations**: Provides specifics about the pod's features and functions.

Important Notes About Pod Boards

Read the following notes first before installing your pod board.

Remove Black Conducting Foam Before Using Your Pod

When using your pod in stand-alone mode, be sure to remove any black conducting foam. This foam is usually inserted at the factory to protect pins that mate with a target adapter or a socket on your target board. The foam covers pins which protrude from the bottom of the pod or from an adapter attached to the pod. The pod will not work with the conducting foam attached and might cause damage. If you are using POD–11E–PLCC or POD–11S–PLCC, you will not have this problem. These pods do not have any pins protruding from the built-in adapter after the adapter is attached.

If you remove the pod from your target socket or target adapter and plan to store it, you will need to re-install the conducting foam to protect the exposed pins.

Memory Mapping Requirements for Single-Chip Mode Pods

Note

In general, map all memory to target except the address space where your code resides.

If you use a single-chip mode pod with the Windows interface software, be sure to change the memory map as described in this section. The memory mapping requirement described here should not present a problem for any user who does not move the SFRs by writing to the INIT register. The HC11s currently in production do not have enough internal program memory to cover the SFRs. However, the default memory map with the current Windows software maps all memory to the emulator regardless of pod type. This is not an issue for DOS software. When you select a single-chip mode pod, a default memory map always sets the SFRs to target mapping.

For single-chip mode pods, if Ports B, and C, (and Port F for non-multiplexed bus parts), are being used for parallel I/O pins, you must map the first 64 bytes of the SFR block to the target. These requirements also apply to the port containing the pin used for the R/W bar signal.

For A, E, F, and similar parts, you must map the range 1000 to 103F to target. For D, K, and P parts, you must map 0 to 3F to target. This requirement assumes you do not move the SFRs by writing to the INIT register.

Avoid Setting the E/GE Pod Jumper to the E Position

Most EMUL68 pods have a strip of three jumper pins which are labeled “E” (non-gated) on one side of the strip and “GE” (gated) on the other. Unless you have a very good reason, do not set this jumper to the E position. Retain the factory default setting of GE. If you set it to the E position, expect that certain locations on your target will be written to while the emulator is in monitor mode.

If you place a jumper from the center pin to the outer pin labeled “E,” E-Clock will be driven to the target at all times. This occurs whether or not the emulator is in monitor mode or emulation mode. The emulator is designed in such a way that this causes certain locations in your target space to be written when the emulator is in monitor mode (for example, if the program is not running, and the emulator’s status shows “Stopped.”) The commands entered in monitor mode determine which locations are destroyed.

If you place a jumper from the center pin to the outer pin labeled “GE,” this gates the E-Clock signal. As a result, the E-Clock is not driven to the target in monitor mode unless an access to target memory is required to refresh or modify data mapped to the target. This position also prevents accidental writing to the target in monitor mode.

POD–11DE

Overview

Use this pod if you are running the 68HC11D series MCU in expanded multiplexed mode or special test mode (external memory). This pod board has a standard 68HC711D3 or 68HC11D0 that you can replace if you want to use a different processor from the same D family.

Figure 26 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board. On the solder side of the board, the P1 connector is a 44-pin male PGA connector. This connector usually requires an adapter for target systems that have a DIP or PLCC socket. (See the EMUL68–PC Price List for further information.)

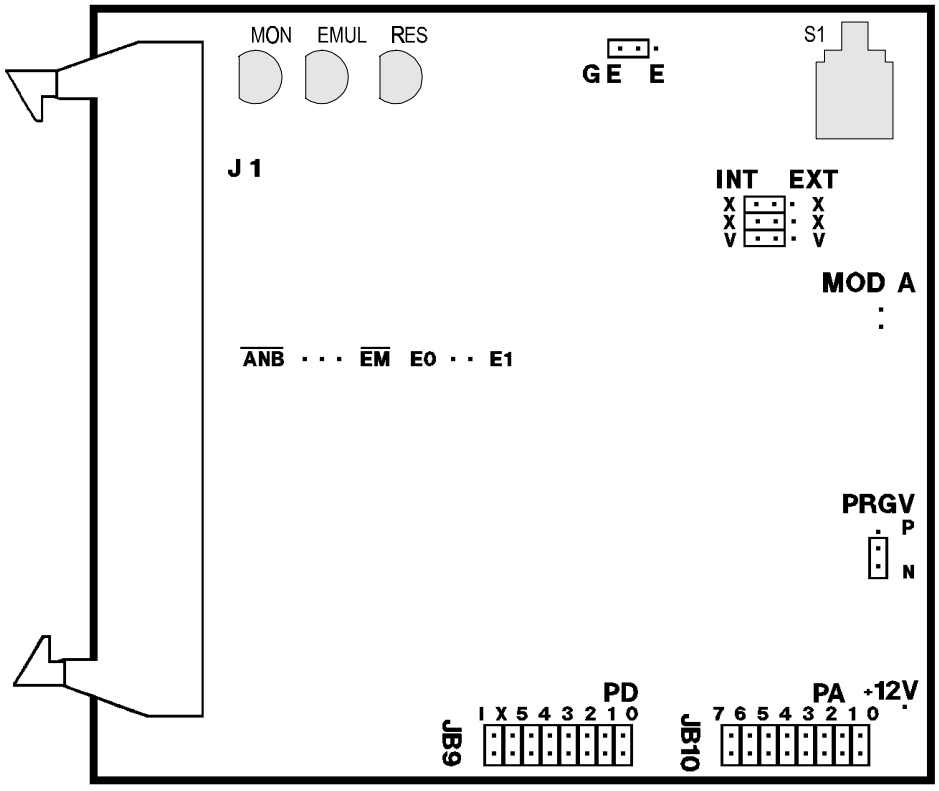

Figure 26. Location of Switches and Jumpers on the POD–11DE Board

POD–11DE LED Indicators and Reset Switch

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.
S1	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.

POD–11DE Configuration Options

Jumper Designation	Function	Description
X, X	**Crystal Selection**	A pair of three-pin, two-position jumper blocks used to select the source of the crystal or clock. Move jumpers as a pair: • Jumpers installed on pins closest to the INT side connect the pod clock to the MCU. • Jumpers installed on the pins closest to the EXT side connect the target clock to the MCU.
V	**MCU Power Selection**	A three-pin, two-position jumper block used to select the source of power for the MCU on the pod. • A jumper installed on the pins closest to the INT side connects pod power to the MCU. • A jumper installed on the pins closest to the EXT side connects target power to the MCU.
GE, E	**E-Clock Gating**	A three-pin, two-position jumper block used to select the E-Clock gating mode: • If a jumper is installed on the pins closest to the E side, the E-Clock signal always goes to the target system (independently of emulator mode). **Warning:** If the E-Clock goes out to the target system in monitor mode, data might accidentally be written to a memory or peripheral in the target system. • If a jumper is installed on pins closest to the GE side, the E-Clock only goes out to the target when the system is in emulation mode and the current address is mapped to the target. (If E is gated, there will be one shorter E pulse at the very end of the emulation mode, when a breakpoint is reached.)

POD–11DE Configuration Options (continued)

Jumper Designation	Function	Description
MOD A	**Isolate MOD A**	A two-pin jumper header: • If no jumper is installed, the target system is isolated from the processor signal MOD A. • If a jumper is installed, the target system will see the processor MOD A (/LIR). **CAUTION:** The target system must not drive this pin. The /LIR signal is essential for emulator function.
E0, E1	**External Signals**	A two-pin header. E0 and E1 connect the external signals to the emulator and are used as trace qualifiers. E0 can also be used in the breakpoint logic.
ANB, EM	**ANB, FLF, EM**	A three-pin header that carries signals from the emulator and trace boards: • EM is high in monitor mode and low in emulation mode. • FLF goes low when the A condition goes *true* on the trace board. FLF goes high when the B condition is *true*. B is dominant; if both A and B conditions are *true*, FLF will be high. If B goes *false* and A goes *true*, FLF will go low again. • ANB is similar to FLF except that ANB stays high after one cycle of high-low-high, independently of A and B.
JB9	**Port D**	An 8 by 2 jumper pin header: • The 8-pin header closest to the center of the board carries the MCU Port D pins 0 through 5. Pins 0 through 5 designate PD0 through PD5; X designates XIRQ, I designates IRQ. • The 8-pin header closest to the edge of the board is connected to the trace PROBE0. External signals can be connected to these pins if jumpers to the Port D pins are removed. The input load is one ALS input (74ALS258).
JB10	**Port A**	An 8 by 2 jumper pin header: • The 8-pin header closest to the center of the board carries the MCU Port A pins 0 through 7. Pins 0 through 7 designate PA0 through PA7. • The 8-pin header closest to the edge of the board is connected to the trace PROBE1. External signals can be connected to these pins if jumpers to the Port A pins are removed. The input load is one ALS input (74ALS258).
P, N	**PRGV**	A three-pin, two-position header: • A jumper installed between the middle pin and the P pin connects the MCU pin XIRQ to the externally-supplied programming voltage. • A jumper installed between the middle pin and the N pin is the normal position and connects the XIRQ signal to the XIRQ pin.
+12V	**Prog. Voltage**	A single pin. Connect an externally-supplied programming voltage to this pin. Used in conjunction with the PRGV header previously described. (For the voltage requirement, consult the latest documentation from the manufacturer.)

POD–11DS

Use this pod if you are running the 68HC11D series MCU in single-chip mode or special bootstrap mode (internal memory). The single-chip mode is emulated using a 68HC11D MCU running in expanded multiplexed mode with Port B, Port C, PD6 and PD7 re-created using a XILINX Logical cell array. The re-created ports have a 100-Ohm series resistor to emulate the output characteristics of the 68HC11D series. Because of limitations in the 68HC11D series MCU, PD6 and PD7 can only be used as outputs. As the 68HC11D MCU is running in expanded multiplexed mode, the MDA bit in the register HIPRIO will be one instead of zero and must not be changed.

This pod board has a standard 68HC711D or 68HC11D MCU that you can replace if it fails or if you want to use a different processor from the same D family. You might also want to change the XILINX part to an equivalent part in case of failure.

Figure 27 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board.. On the solder side of the board, the P1 mates with a PGA connector on the target system. You will need to add an adapter (available from Nohau) if the target system has a DIP socket or PLCC socket.

For more information, refer to Chapter 2, “Installing the Hardware,” and review details about your specific emulator board jumper configuration.

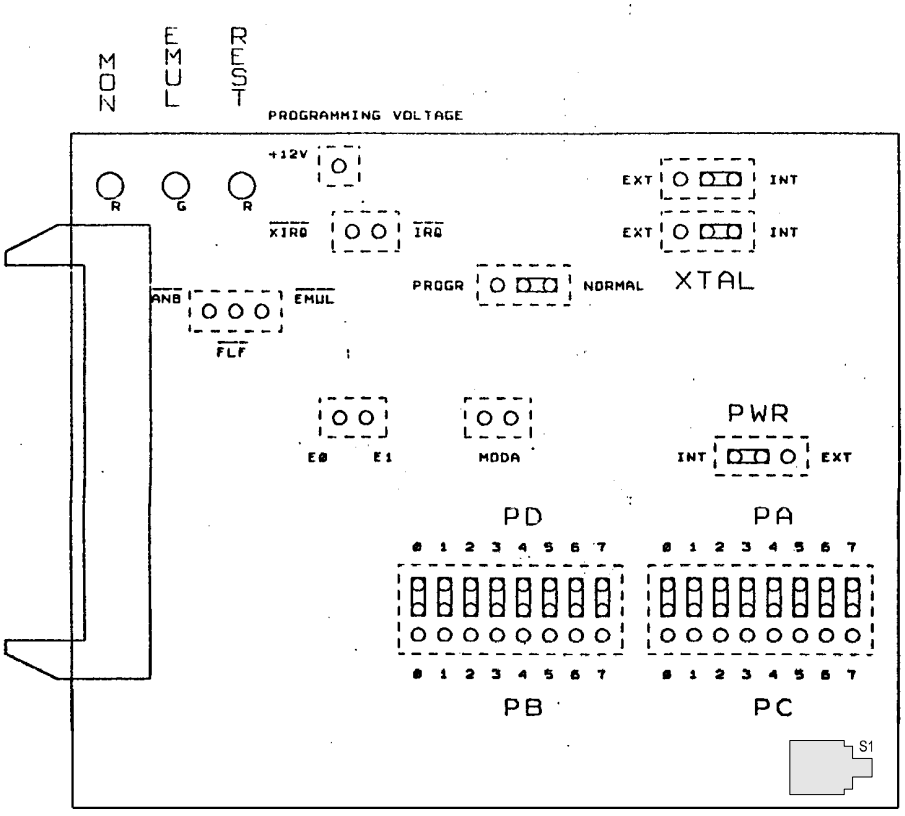

Figure 27. POD–11DS Board Configured for Target Power and Internal Crystal

POD–11DS LED Indicators and Reset Switch

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.
S1	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.

POD–11DS Configuration Options

Jumper Designation	Function	Description
XTAL	**Crystal Selection**	A pair of three-pin, two-position jumper blocks used to select the source of the crystal or clock. Move jumpers as a pair: • Jumpers installed on pins closest to the INT side connect the pod clock to the MCU. • Jumpers installed on the pins closest to the EXT side connect the target clock to the MCU.
PWR	**MCU Power Selection**	A three-pin, two-position jumper block used to select the source of power for the MCU on the pod. • A jumper installed on the pins closest to the INT side connects pod power to the MCU. • A jumper installed on the pins closest to the EXT side connects target power to the MCU. The XILINX part is always powered from the emulator.
MOD A	**Isolate MOD A**	A two-pin jumper header: • If no jumper is installed, the target system is isolated from the processor signal MOD A. • If a jumper is installed, the target system will see the processor MOD A (/LIR). **CAUTION:** The target system must not drive this pin. The /LIR signal is essential for emulator function.
E0, E1	**External Signals**	A two-pin header. E0 and E1 connect the external signals to the emulator and are used as trace qualifiers. E0 can also be used in the breakpoint logic.

POD–11DS Configuration Options (continued)

Jumper Designation	Function	Description
ANB, EM	**ANB, FLF, EM**	A three-pin header that carries signals from the emulator and trace boards: • EM is high in monitor mode and low in emulation mode. • FLF goes low when the A condition goes *true* on the trace board. FLF goes high when the B condition is *true*. B is dominant; if both A and B conditions are *true*, FLF will be high. If B goes *false* and A goes *true*, FLF will go low again. • ANB is similar to FLF except that ANB stays high after one cycle of high-low-high, independently of A and B.
PD, PB	**Port D, Port B**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the center of the board carries the 68HC11D Port D pins (PD6 and PD7 re-created) and the lower row of pins carries the re-created Port B pins. • The middle pins are reflected on the trace setup and display as PROBE0. External signals can be connected to these pins; the input load is one ALS input (74ALS258).
PA, PC	**Port A, Port C**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the center of the board carries the 68HC11D Port A pins: 0 through 7 designate PA0 through PA7. • The lower row of pins carries the re-created Port C pins. • The middle row is reflected on the trace setup and display as PROBE1. External signals can be connected to these pins; the input load is one ALS input (74ALS258).
JP1	**XIRQ, IRQ**	A strip of two pins. Jumper J1 carries the processor signals XIRQ and IRQ. XIRQ is on the left pin, and IRQ is on the right. XIRQ is gated, so that it is always high in monitor mode.
P, N	**PRGV**	A three-pin, two-position header: • A jumper installed between the middle pin and the P pin connects the MCU pin XIRQ to the externally-supplied programming voltage. • A jumper installed between the middle pin and the N pin is the normal position and connects the XIRQ signal to the XIRQ pin.
+12V	**Prog. Voltage**	A single pin. Connect an externally-supplied programming voltage to this pin. Used in conjunction with the PRGV header previously described. (For the voltage requirement, consult the latest documentation from the manufacturer.)

Special Considerations for the D Family

The special considerations described here refer to differences between the D family and the 68HC11A series, in particular the enabling and disabling of the Computer Operating Properly (COP), and the size of the BOOT ROM.

COP

The COP is enabled when the processor is started in expanded multiplexed mode. The emulator automatically disables the COP when the emulator is invoked, and if a RESET CHIP is issued in monitor mode (when the emulator is not emulating). The COP will be enabled again if the chip is reset during emulation.

Emulation ends when a breakpoint occurs or when emulation is forced to break, but the COP continues to run. Because monitor mode doesn't restart the COP, eventually the COP will be activated and start resetting the chip, while still in monitor mode. This will create problems for the monitor program. It is therefore important that the user code includes instructions to disable the COP at reset. The COP must be disabled within the first 64 cycles, and that the Config can only be written to once.

BOOT ROM

The size of the BOOT ROM is another difference between 68HC11 D-series family chips and 68HC11 A-series chips. The procedure given later in Appendix F, “Restrictions and Differences” for copying the BOOT ROM into emulator memory refers to 68HC11 A-series chips. You can use the same procedure with 68HC11D family chips also, except that you expand the address range of the memory transfer to (BF00 – BFFF).

INIT

The addresses of the re-created ports PB, PC, PD6 and PD7 on the POD–11DS are fixed at the default values and cannot be changed by writing to the INIT register. This only refers to the POD–11DS.

POD–11E

Use this pod if you are running a 68HC11 processor in expanded multiplexed mode or special test mode (external memory). The following descriptions apply to the A-series and the E-series MCUs: 68HC11A0, 68HC11A1, 68HC11E0, 68HC11E1, and 68HC811E2.

For the E-series MCUs, also refer to the “Special Considerations for 68HC11 E-series” section later in this chapter.

Figure 28 shows how switches and jumpers are set when the board is shipped from the factory.

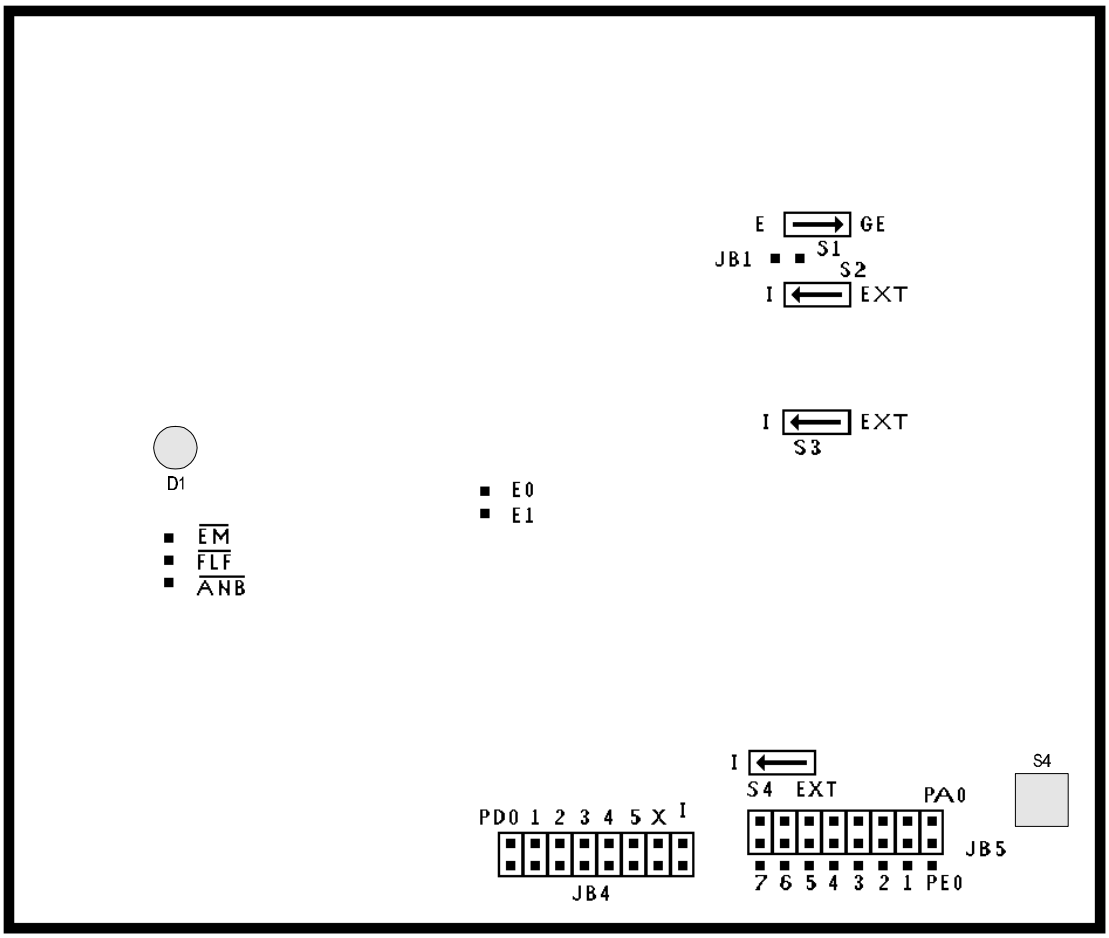

Figure 28. POD–11E Board Configured for Target Power and Internal Crystal

POD–11E Configuration Options

Jumper Designation	Function	Description
S1	**E-Clock**	A two-position slide switch: • If set in E position, the E-Clock signal always goes to target system, (independently of emulator mode). • If set in GE position, clock E only goes to target in emulation mode. (If E goes to target in monitor mode, something can accidentally be written to a memory or peripheral in the target. • If E is gated, there will be one shorter E pulse at very end of the emulation mode, when a breakpoint is reached.)
S2, S3	**Crystal Selection**	Two 2-position slide switches. • If both are set in the position marked INT, the on-board pod crystal is connected to the oscillator pins. • If both are set in the EXT position, the crystal on the target system or target clock is connected to the oscillator pins.
S4	**MCU Power Selection**	A 2-position slide switch. • If set in INT position, the 68HC11 MCU receives power from the PC. • If set in EXT position, the MCU receives power from the target system.
JB1	**Isolate MOD A**	A two-pin jumper header: • If no jumper is installed, the target system is isolated from the processor signal MOD A. • If a jumper is installed, the target system will see the processor MOD A (/LIR). **CAUTION:** The target system must not drive this signal, because the /LIR signal is essential for emulator function.
JB2	**E0, E1**	A two-pin strip. • E0 and E1 are available for connecting external signals to the emulator and are used for the trace function. • E0 can also be used in the breakpoint logic.
JB3	**EM, FLF, ANB**	A three-row strip of pins that carries signals from the emulator and trace boards: • EM is high in monitor mode and low in emulation mode. • FLF goes low when the A condition goes *true* on the trace board and goes high when the B condition is *true*. B is dominant; if both A and B conditions are *true*, FLF will be high. If B goes *false* and A goes *true*, FLF will go low again. • ANB is similar to FLF except that ANB stays high after one cycle of high-low-high, independently of A and B.

POD–11E Configuration Options (continued)

Jumper Designation	Function	Description
JB4	**Port D**	An 8-pin by 2-pin array of jumper pins. • In the upper row the first five pins, PD0 - PD5, carry the 68HC11 Port D pins. The pin on the right carries IRQ. The second pin from the right carries XIRQ. (XIRQ is gated, so that it is always high in monitor mode.) • Levels connected to the lower row of pins are reflected on the trace display. The input load is one ALS input (74ALS258). When shipped from the factory, jumpers are set between pins in the middle and lower rows, so that normally PD0 – PD5, IRQ and XIRQ will be traced. To trace external signals, remove jumpers and then connect external signals to the lower row of pins. JB4 is called PROBE0 in the **Trace Setup** menu and the Qualifier Register command.
JB5	**Port A, Port E**	An 8-pin by 3-pin array of jumper pins. • Upper row of pins, PA0 - PA7, carries port A signals from 68HC11. • Middle row of pins goes to trace inputs. • Lower row of pins, PE0 - PE7, carries port E signals from 68HC11. When shipped from the factory, jumpers are set between pins in the upper and middle rows, so that Port A is normally traced. Users can move jumpers to the lower row, or connect external signals to the middle row. JB5 is called PROBE1 in the **Trace Setup** menu and the Qualifier Register command.
S5	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.
D1	**Status**	A red LED that indicates the MCU is receiving a reset signal.

Note

Connectors J2 and J3 mate with the adapter. The white line on the adapter should be aligned with the white line on the pod board.

The J1 connector mates with the ribbon cable from the emulator board. The J2 and J3 connectors mate with a socket adapter that plugs into the solder side of the pod board. Socket adapters are available for mating with a DIP or PLCC socket on the target system.

EPROM Programming

> **Note**
>
> Currently Seehau is not capable of programming EPROMs. Refer to the documentation included with the EMUL68 for DOS software.

The POD–11E and POD–11S do not provide any connection for the programming voltage. Use the special adapter (Part #EMUL68–PC/PLCC52ZIF–PLCC52). This adapter has a zero insertion force socket with a connector.

POD–11FE

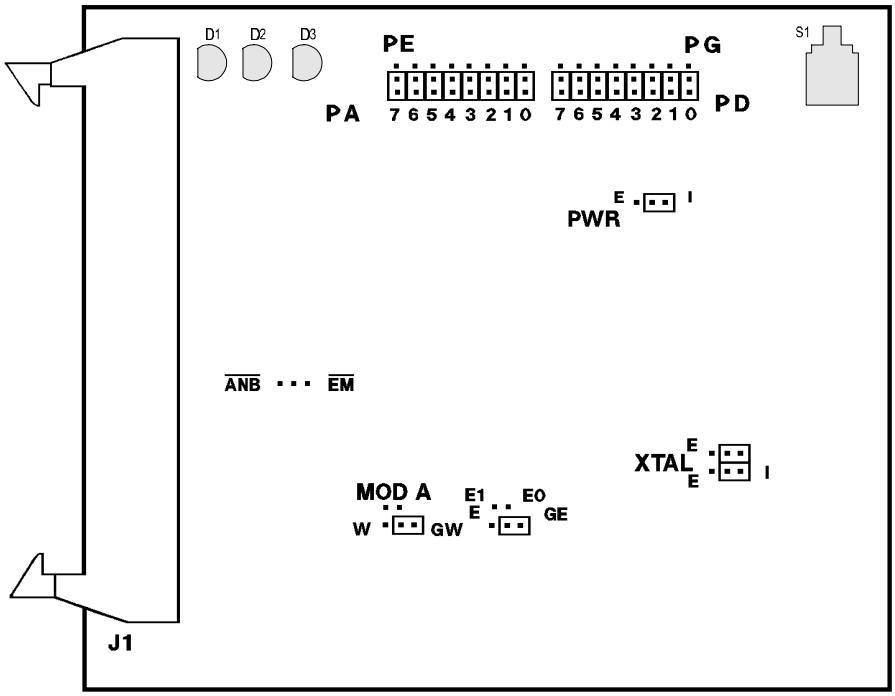

Figure 29. POD–11FE Board Configured for Target Power and Internal Crystal

Use this pod if you are running the 68HC11F1 processor in expanded multiplexed mode or special test mode (external memory). This pod board has a standard 68HC11F1 that you can replace if it fails or if you want to use a different processor from the same F family.

Figure 29 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board. On the solder side of the board, the P1 connector mates with a PGA connector on the target system. You will need to add an adapter (available from Nohau) if the target system has a PLCC socket.

POD–11FE LED Indicators and Reset Switch

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.
S1	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.

POD–11FE Configuration Options

Jumper Designation	Function	Description
XTAL	**Crystal Selection**	Two strips of three jumper pins for selecting the source of the crystal or clock. • Jumpers installed on the pins closest to the E side means the source is in the target system. • Jumpers installed on pins closest to I side means the crystal on the pod is to be used.
PWR	**MCU Power Selection**	A strip of three jumper pins for selecting source of power for the MCU on the pod. • A jumper installed on the pins closest to the I side means power comes from the emulator. • A jumper on the pins closest to the E side means power comes from the target system.
E, GE	**E-Clock**	A strip of three jumper pins. • If a jumper is installed on the pins closest to the E side, the E-Clock signal always goes to the target system directly form the F1 chip. • If a jumper is installed on the pins closest to the GE side, the E-Clock goes through a gate that holds it low in monitor mode or if the memory is mapped to the emulator board. • If the E-Clock goes out to the target system in monitor mode, something might accidentally be written to a memory or peripheral in the target system. If E is gated, there will be one shorter E pulse at the very end of the emulation mode, when a breakpoint is reached.
W, GW	**R/W Line**	A strip of three pins that select the function of the R/W line. • A jumper installed on the pins closest to W means the R/W signal to the target system will be connected directly to the R/W line on the F1 chip. • A jumper installed on the pins closest to GW means R/W will go through a gate that holds it high in monitor mode or if memory is mapped to the emulator board.
MOD A	**Isolate MOD A**	A strip of two pins. • If no jumper is installed, the target system will be isolated from the processor signal MOD A. • If a jumper is installed, the target system will see the processor MOD A (/LIR) (in which case the target system must not drive this signal because the /LIR signal is essential for the emulator function).
E0, E1	**External Signals**	A two-pin strip. E0 and E1 are available for connecting external signals to the emulator and are used for the trace function. E0 can also be used in the breakpoint logic.

POD–11FE Configuration Options (continued)

Jumper Designation	Function	Description
ANB, EM	**ANB, FLF, EM**	A strip of three pins that carries signals from the emulator and trace board. • The pin closest to EM goes high in monitor mode and low in emulation mode. • The middle pin, designated FLF, goes low when the A condition in the trace board goes *true*, and goes high when the B condition goes *true*. (B is dominant, so the FLF pin will be high if the A and B conditions are both *true*. If B goes *false*, and then A goes *true*, the FLF pin will go low again. The signal at the pin closest to ANB behaves basically the same as FLF, but stays high after one cycle of high-low-high, independently of A and B.)
PE, PA	**Port E, Port A**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11F1 Port E pins: 0 through 7 designate PE0 through PE7. • The row of pins toward the center of the board carries the 68HC11F1 Port A pins: 0 through 7 designate PA0 through PA7. • The middle row of pins is reflected on the trace setup and display as PROBE1. External signals can be connected to these pins if the jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that port A is traced.
PG, PD	**Port G, Port D**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11F1 Port G pins: 0 through 7 designate PG0 through PG7. • The row of pins toward the center of the board carries the 68HC11F1 Port D pins: 0 through 5 designate PD0 through PD5. The silk screen artwork has two errors here: the pin designated PD6 should instead be X (for XIRQ), and the pin designated PD7 should instead be I (for IRQ). XIRQ is gated, so that it is always high in monitor mode. • The middle row of pins is reflected on the trace setup and display as PROBE0. External signals can be connected to these pins if jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that PD0 – PD5, XIRQ and IRQ will be traced.

Special Considerations for the 68HC11F1 MCU

When you receive the POD–11FE changes have been made to the configuration of the MCU to allow its use with the emulator. Specifically, the EEPROM has been moved to address BE00 - BFFF (because the default addresses conflict with the standard reset vector addresses and the EEPROM is also disabled), and the COP has been disabled.

BPROT

Note that all 68HC11 F-series MCUs discussed here have a Bank Protection Register (BPROT). If you want to program the EEPROM, zeros must be written to this register. When emulating, the BPROT register should be initialized correctly by the user software. When not emulating, you can change the BPROT register by entering a value in the BPROT override field found in the **Hardware Configuration** dialog box. Then click **OK** or **Apply**. Anytime you reset the emulator, the BPROT register is reloaded with your override value.

Chip Select Logic

The 68HC11F1 chip select logic feature will not be switched off when emulation breaks. This means that chip select signals to the target can be generated by the chip select logic even if the emulator is in monitor mode (not emulating).

BOOT ROM

The size of the BOOT ROM is another difference between 68HC11F1 chips and 68HC11 A-series chips. The procedure described in Appendix F, “Restrictions and Differences,” refers to copying the BOOT ROM into emulator memory for 68HC11 A-series chips. You can use the same procedure with 68HC11F1 chips, except that you expand the address range of the memory transfer to (BF00 - BFFF).

POD–11FL0

Overview

This pod has a standard 68HC11FL0. Use this pod if you are running the 68HC11FL0 in expanded multiplexed mode or special test mode (external memory).

Figure 30 shows how switches and jumpers are set when the board is shipped from the factory. J1 is a connector that mates with the ribbon cable from the emulator board. The P1 connector is on the solder side of the board and mates with a four-sided dual row header that connects to an adapter. The adapter then connects to the target system.

For further information, refer to Chapter 2, “Installing the Hardware. Go to the section “Configuring the Emulator Board for Chips with Zero-Page Registers.”

POD–11FL0 Default Positions for Jumpers

Jumpers are configured for stand-alone operation rather than target operation. Figure 30 shows the default positions of the jumpers for the 68HC11FL0 emulation pod board.

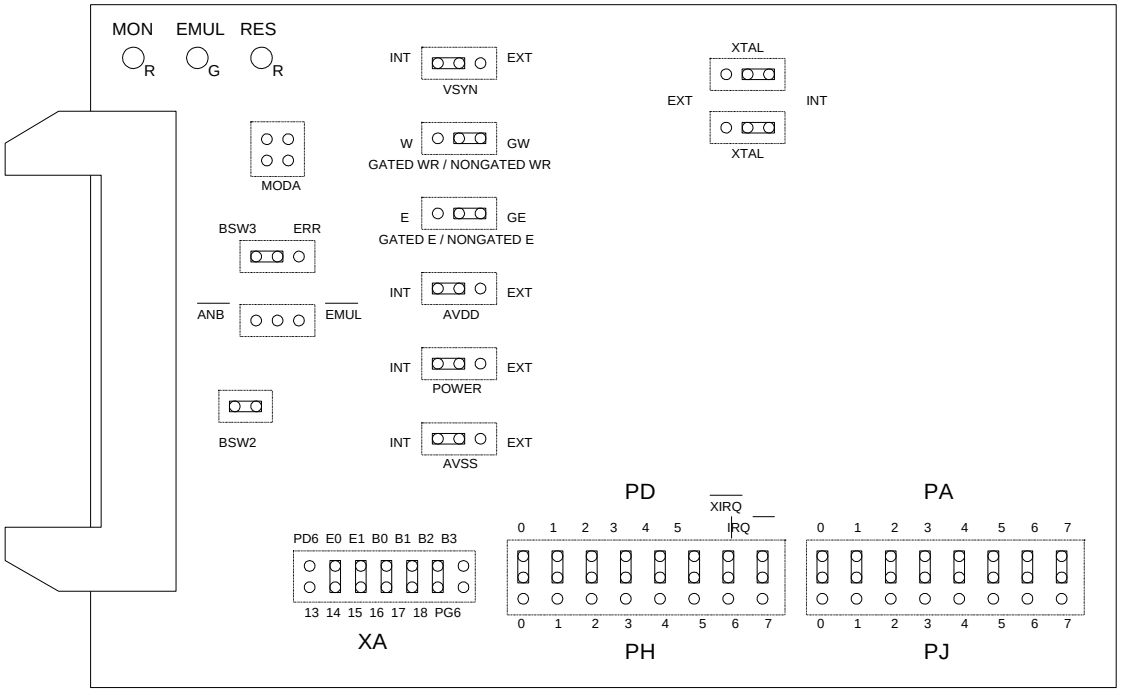

Figure 30. POD–11FL0 Jumper Positions

POD–11FL0 Configuration Options

Jumper Designation	Function	Description
MON		A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
EMUL		A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
RESET		A red LED that indicates the MCU is receiving a reset signal.
AVSS, AVDD, VSYN		• When running in stand-alone mode, these jumpers must be in the INT (Internal) position. The pod is shipped with these jumpers in the INT position. • When connected to a target system, these jumpers can be placed in the EXT (external) position to receive the corresponding grounds and power from the target system.
XTAL	**Crystal Selection**	Two strips of three jumper pins for selecting the source of the crystal or clock. • Jumpers installed on the pins closest to the E side means the source is in the target system. • Jumpers installed on pins closest to I side means the crystal on the pod is to be used.
POWER	**MCU Power Selection**	A strip of three jumper pins for selecting source of power for the MCU on the pod. • A jumper installed on the pins closest to the I side means power comes from the emulator. • A jumper on the pins closest to the E side means power comes from the target system.
BSW0		Installed if the pod is used with EMUL68–PC/EB1M.
BSW1	**ERR**	Set in BSW1 if the pod is used with EMUL68–PC/EB1M.
W, GW	**R/W Line**	A strip of three pins that selects the function of the R/W line: • A jumper installed on the pins closest to W means the R/W signal to the target system will be connected directly to the R/W line on the FL0 chip. • A jumper installed on the pins closest to GW means R/W will go through a gate that holds it high in monitor mode or if memory is mapped to the emulator board.
E, GE	**E-Clock**	A strip of three jumper pins: • If a jumper is installed on the pins closest to the E side, the E-Clock signal always goes to the target system directly form the FL0 chip. • If a jumper is installed on the pins closest to the GE side, the E-Clock goes through a gate that holds it low in monitor mode or if the memory is mapped to the emulator board. • If the E-Clock goes out to the target system in monitor mode, something might accidentally be written to a memory or peripheral in the target system. If E is gated, there will be one shorter E pulse at the very end of the emulation mode, when a breakpoint is reached.

POD–11FL0 Configuration Options (continued)

Jumper Designation	Function	Description
ANB, FLF, EMUL	**ANB, FLF, EM**	A strip of three pins that carries signals from the emulator and trace board. • The pin closest to EM goes high in monitor mode and low in emulation mode. • The middle pin, designated FLF, goes low when the A condition in the trace board goes *true*, and goes high when the B condition goes *true*. (B is dominant, so the FLF pin will be high if the A and B conditions are both *true*. If B goes *false*, and then A goes *true*, the FLF pin will go low again. The signal at the pin closest to ANB behaves basically the same as FLF, but stays high after one cycle of high-low-high, independently of A and B.)
XA		These jumpers must be positioned correctly when you use the HC11FL0's internal bank switching. • The jumpers are set for 32 banks and 16K bank size. For example, note that XA14 is connected to E0 and XA15 is connected to E1. • If you require 64 banks, you must connect PG6 to BS3 (put in the jumper). Port G bit 6 must be designated to switch between the first 32 banks and the second set of 32 banks. (FL0's bank switching scheme is only set up for a maximum of 32 banks). • This jumper connects to the emulator only. To get the bank switch bit 6 to the trace, connect PG6 (use pin 5 on the FL0) to PROBE1–3 middle pin after removing the jumper. (PROBE1 is marked PD/PH.) • If you want to use 8K banks, you must connect XA13 to E0, and XA14 to E1, etc. In order to do this, you must wire wrap the pins. There is no reason to connect PD6. This pin is connected to PD6 on the FL0 (pin 3).
MOD A Pins	**Isolate MOD A**	Leave these pins disconnected.
PROBE0	**PA/PJ**	The default jumper setting will make PA (Port A) appear under PROBE0 in the trace display. You can change jumpers to connect to PJ (Port J) or connect the middle pin to any other signal you want to trace.
PROBE1	**PD/PH**	Default jumper setting will make PD (Port D and IRQ) appear under PROBE0 in the trace display. You can change jumpers to connect to PJ (Port J) or connect the middle pin to any other signal you want to trace.

POD–11KE

Overview

Use this pod if you are running the 68HC11K series in expanded multiplexed mode or special test mode (external memory). This pod board has a standard 68HC11K series MCU or 68HC711K series MCU that you can replace if it fails or if you want to use a different processor from the same K family.

> **Note**
>
> If you use K, N, or P family parts, SeehauHC11 will not work if the PAREN bit in CONFIG is set to one. Be sure to set the PAREN bit to zero. PAREN activates the PPAR pull-up assignment register.

Figure 31 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board. On the solder side of the board, the P1 connector mates with a PGA connector on the target system. You will need to add an adapter if the target system has a PLCC connector (available from Nohau).

Refer to Chapter 2, “Installing the Hardware.” Go to the section that covers your specific emulator board. Then refer to the jumper label SFR base address.

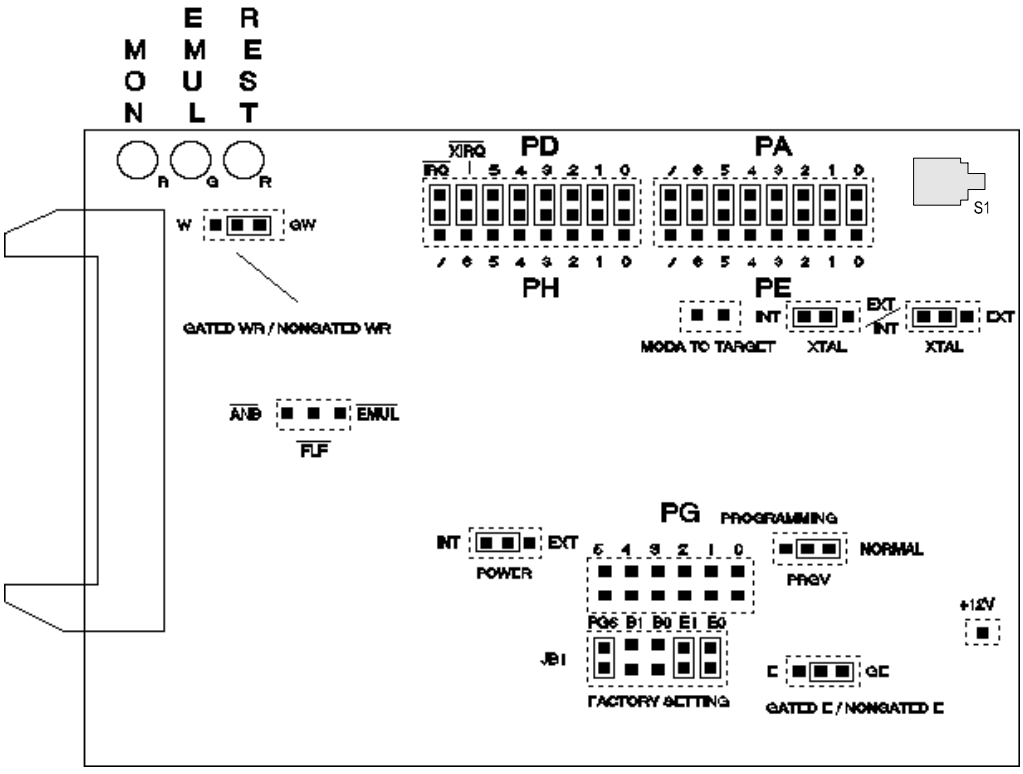

Figure 31. POD–11KE Board Configured for Target Power and Internal Crystal

POD–11KE LED Indicators and Reset Switch

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.
S1	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.

POD–11KE Configuration Options

Jumper Designation	Function	Description
XTAL	**Crystal Selection**	Two strips of three jumper pins for selecting the source of the crystal or clock. • Jumpers installed on the pins closest to the E side means the source is in the target system. • Jumpers installed on pins closest to I side means crystal on the pod is to be used.
PWR	**MCU Power Selection**	A strip of three jumper pins for selecting source of power for the MCU on the pod. • A jumper installed on the pins closest to the INT side means power comes from the emulator. • A jumper on the pins closest to the EXT side means power comes from the target system.
E, GE	**E Clock**	A strip of three jumper pins. • If a jumper is installed on the pins closed to the E side, the E-Clock signal always goes to the target system directly from the K4 chip. • If a jumper is installed on the pins closest to the GE side, the E-Clock goes through a gate that holds it low in monitor mode or if the memory accessed is mapped to the emulator board. (If E-Clock goes out to the target system in monitor mode, something might accidentally be written to a memory or peripheral device in the target system.) • If E is gated, there will be one shorter E pulse at the very end of the emulation mode, when a breakpoint is reached.
W, GW	**R/W Line**	A strip of three pins that selects the function of the R/W line. • A jumper installed on the pins closest to W means the R/W signal to the target system is connected directly to the R/W line on the J6 chip. • A jumper installed on the pins closest to GW means R/W will go through a gate that holds it high in monitor mode or if memory is mapped to the emulator board.

POD–11KE Configuration Options (continued)

Jumper Designation	Function	Description
MOD A	**Isolate MOD A**	A strip of two pins. • If no jumper is installed, the target system will be isolated from the processor signal MOD A. • If a jumper is installed, the target system will see the processor MOD A (/LIR) (in which case the target system must not drive this signal because the /LIR signal is essential for the emulation function).
PG	**Port G**	A part of a six-pin by two-pin array of jumper pins. • The upper row and the leftmost pin in the lower row carries the 68HC11K Port G pins. • The upper row contains port pins as designated by the labels above it and the leftmost pin in the lower row carries bit 6 of Port G.
E1, E0	**External Signals**	A part of a six-pin by two-pin array of jumper pins. These are the two pins in the lower row designated E0 and E1. E0 and E1 are available for connecting external signals to the emulator and are used for trace functions. E0 can also be used in the breakpoint logic. When the bank switch emulator is used E0 and E1 also duplicates as the E0 and E1 inputs.
B0, B1	**Bank Switch Select Input**	A part of a six-pin by two-pin array. These are the two pins in the lower row designated B0 and B1. They are used with the bank switch emulator as BSW0 and BSW1 inputs.
ANB, EM	**ANB, FLF, EM**	A strip of three pins that carries signals from the emulator and trace board. • The pin closest to EM goes high in monitor mode and low in emulation mode. • The middle pin, designated FLF, goes low when the A condition in the trace board goes *true*, and goes high when the B condition goes *true*. (B is dominant, so the FLF pin will be high if the A and B conditions are both *true*. If B goes *false*, and then A goes *true*, the FLF pin will go low again.) • The signal at the pin closest to ANB behaves basically the same as FLF, but stays high after one cycle of high-low-high, independently of A and B.
PE, PA	**Port E, Port A**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11K Port A pins: 0 through 7 designate PA0 through PA7. • The row of pins toward the center of the board carries the 68HC11K Port E pins: 0 through 7 designate PE0 through PE7. • The middle row of pins is reflected on the trace setup and display as PROBE1. External signals can be connected to these pins if the jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that Port A is traced.

POD–11KE Configuration Options (continued)

Jumper Designation	Function	Description
PH, PD	**Port H, Port D**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11K Port D pins: 0 through 5 designate PD0 through PD5. The pin following PD5 is the XIRQ pin and the next pin is the IRQ pin from, the MCU. XIRQ is gated and will always be high when in monitor mode. • The row of pins toward the center of the board carries the 68HC11K Port H pins: 0 through 7 designated PH0 through PH7. • The middle row of pins is reflected on the trace setup and display as PROBE0. External signals can be connected to these pins if jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that PD0 – PD5, XIRQ and IRQ will be traced.
PRGV	**Select XIRQ Input**	A strip of three pins that is used to connect the programming voltage to the XIRQ pin of the 68HC711K4 chip. • In the P position the +12V pin (see Figure 31) will be connected to XIRQ and you can program the EPROM of the MCU. • In the N position the XIRQ of the MCU will be connected to the XIRQ from the target system.
+12V	**Vpp**	A single pin. Apply the externally supplied programming voltage here. For the voltage requirement, consult the latest documentation from Motorola.
JB1		This five-pin by two-pin array of jumper pins is used to control some timing parameters of the POD–11KE and should never be changed except on explicit instruction from Nohau.

POD–11KS

This pod board is used to emulate the Motorola microcontrollers 68HC11K series in single-chip mode.

Single chip mode is emulated using a 68HC11K MCU running in expanded multiplexed mode with Port B, Port C, Port F, and PG7 re-created using a XILINX Logical cell array. The re-created ports have a 100-ohm series resistor to emulate the output characteristics of the 68HC11K series.

Refer to Chapter 2, “Installing the Hardware.” Go to the section that covers your specific emulator board. Then refer to the jumper label SFR base address.

Figure 32 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board. On the solder side of the board, the P1 connector mates with a PGA connector on the target system. You will need to add an adapter if the target system has a PLCC connector (available from Nohau).

The pod board has a standard 68HC11K series MCU. You can replace the part in case of failure, or if you want to use another part in the same K family. You can also change the XILINX part to an equivalent part in case of failure.

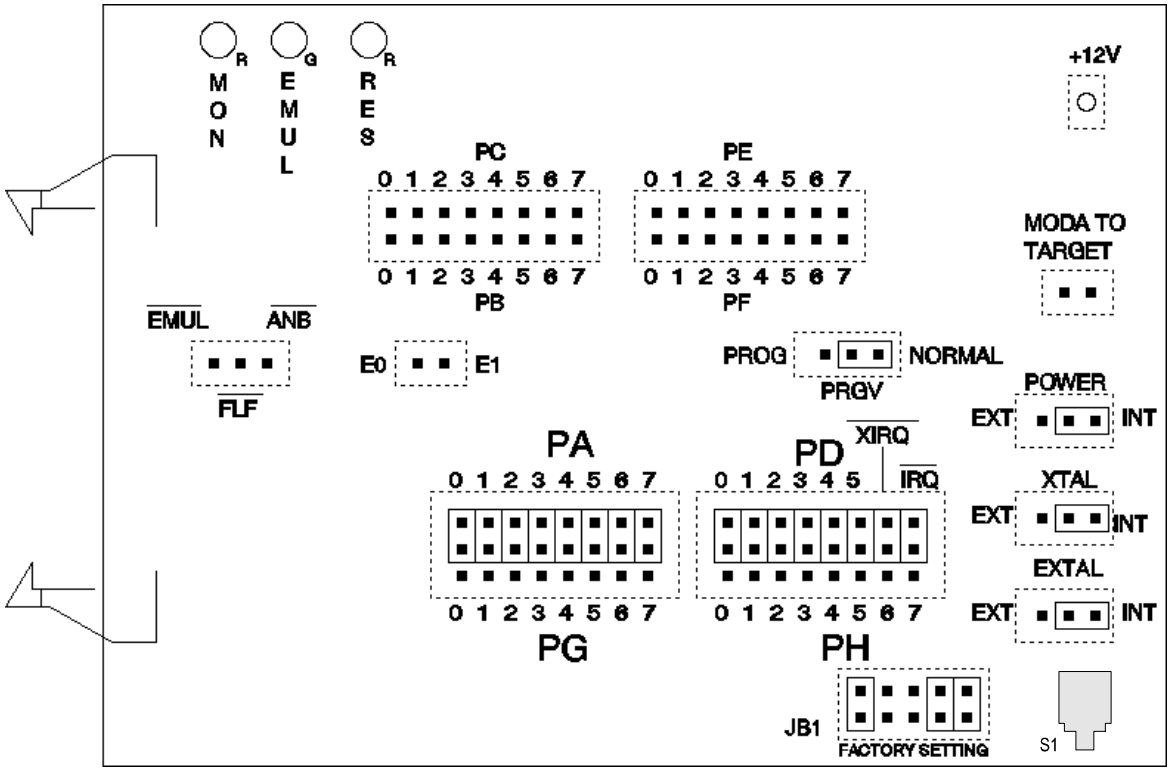

Figure 32. POD–11KS Board Configured for Target Power and Internal Crystal

POD–11KS LED Indicators

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.

POD–11KS Configuration Options

The POD–11KS has three jumpers for power and crystal connection to the 68HC11K series chip on the board.

Jumper Designation	Function	Description
EXTAL, XTAL	**Crystal Selection**	Jumpers marked EXTAL and XTAL determine if the crystal or clock is in the target system, or the crystal on the pod is used. The jumpers should be in the right position for the crystal on the pod board (I), and in the left position for target supplied crystal or clock (E).
PWR	**MCU Power Selection**	The jumper marked Pwr is used for power to the on board MCU chip. • If the power is to be supplied from the emulator, the jumper should be in the right position (INT). • If power is to come from the target system, the jumper should be in the left position (EXT). • The XILINX part that re-creates PB, PC, PF is always powered from the emulator.
MOD A	**Isolate MOD A**	The jumper MOD A is used to isolate the target system from the processor signal MOD A. If the jumper is in, the target system will see the processor signal MOD A (/LIR). The target system must not drive this signal, since the /LIR signal is essential for the emulator function. This jumper should normally be out.
E1, E0	**External Signals**	The jumper pins marked E0 – E1 are used to connect external signals to the emulator. These are used for the trace function. The left pin, E0, can also be used in the breakpoint logic.
EM	**ANB, FLF, EM**	The jumper marked EM carries signals from the emulator and trace board. • The left pin EM is high in monitor mode, and low in emulation mode. • The middle pin FLF, goes low, when A condition goes *true* in the trace board, and goes high when the B condition is *true*. B is dominant, so if both A and B conditions are *true*, the signal will be high. If B goes *false*, and then A goes *true*, the pin will go low again. • The right pin ANB does basically the same as FLF, but stays high after one cycle of high-low-high independent of A and B.

POD–11KS Configuration Options (continued)

Jumper Designation	Function	Description
PA, PG	**Port A, Port G**	• The upper row of pins on jumper block PA/PG carries the 68HC11K chip Port A pins. • The lower row of pins carries the Port G pins. PA0 and PG0 are on the left, and PA7 and PG7 are on the right. • The middle pins are reflected on the trace display. The board is delivered with jumpers, so that PA0-PA7 will be traced. External signals can be connected to the middle pins if the jumpers to Port A are removed. The input load is one ALS input (74ALS258).
PD, PH	**Port D, Port H**	• The upper row of pins on jumper block PD/PH carries the 68HC11K chip Port D pins plus XIRQ/ and IRQ/. • The lower row of pins carries the Port H pins. PD0 and PH0 are on the left, and IRQ/ and PH7 are on the right. • The middle pins are reflected on the trace display. The board is delivered with jumpers, so that PD0-PD5, XIRQ/ and IRQ/ will be traced. External signals can be connected to the middle pins if the jumpers to Port D are removed. The input load is one ALS input (74ALS258).
PRGV	**Select XIRQ Input**	The jumper PRGV is used to select if the MCU pin XIRQ will be connected to programming voltage (left P position), or in normal position (right N position). The pin +12V is used for the externally supplied programming voltage. Refer to the latest documentation from Motorola for the voltage requirement.
PC/PB, PE/PF	**Ports C, B and Ports E, F**	Jumper blocks PC/PB and PE/PF carry the corresponding port pins with pin 0 on the left and pin 7 on the right.
JB1		Jumper block JB1 is factory set and should not be changed.
S1	**Reset**	Resets the MCU. The S1 switch can be used instead of the target reset.

Port G Bit 7 Must be Used As an Output

If you use POD–11KS or POD–11PS, Port G bit 7 must be used as an output, and not as an input. This is a limitation in the design of the emulator. For expanded mode pods, this limitation does not apply.

Special Considerations for 68HC711K4 and 68HC11K1

When you receive the POD–11KE or POD–11KS, certain changes have been made to the configuration of the MCU to allow its use with the emulator. Specifically, the EPROM has been disabled. If you decide to configure a new 68HC711K4 or 68HC11K1 chip for use by EMUL68, refer to the “Changing the Pod MCU,” section found later in this chapter.

N and P Family Users: Set PAREN Bit in the Config Register to Zero

If you use K, N or P family parts, SeehauHC11 will not work if the PAREN bit in CONFIG is set to 1. Be sure to set it to zero. PAREN activates the PPAR pull-up assignment register.

BPROT Register

All 68HC11K series MCUs discussed here have a Bank Protection Register (BPROT). If you want to program the EEPROM, zeros must be written to this register. When emulating, the BPROT register should be initialized correctly by the user software. When not emulating, you can change the BPROT register by entering a value in the BPROT override field found in the **Hardware Configuration** dialog box. Then click **OK** or **Apply**. Anytime you reset the emulator, the BPROT register is reloaded with your override value.

INIT2 Command

INIT2 is a register at address 0037H in the register area. INIT2 is used to move the EEPROM to an address other than 0D80H. You can change the address to xD80H, where x ranges from 0 to F. In this case, you need to communicate to the emulator that EEPROM is at the new address. If not, SeehauHC11 will fail if it tries to stop emulation when the processor is executing code within the EEPROM. Use the INIT2 command to set this up correctly. It is not possible to change this command every time something is written to INIT2. Alternatively, the emulator should be set to the default value for INIT2.

POD–11PE

Use this pod if you are running the 68HC11P or PH series in expanded multiplexed mode or special test mode (external memory). This pod board has a standard 68HC11P or PH that you can replace if it fails or if you want to use a different processor from the same P family.

Figure 33 shows how switches and jumpers are set when the board is shipped from the factory. The J1 connector mates with the ribbon cable from the emulator board. On the solder side of the board, the P1 connector mates with a PGA connector on the target system. You will need to add an adapter (available from Nohau) if the target system has a PLCC connector.

> **Note**
>
> The POD–11PE is a modified POD–11KE. Therefore the silk screen on the pod is labeled as POD–11KE.

Refer to Chapter 2, “Installing the Hardware.” Go to the section that covers your specific emulator board. Then refer to the jumper label SFR base address.

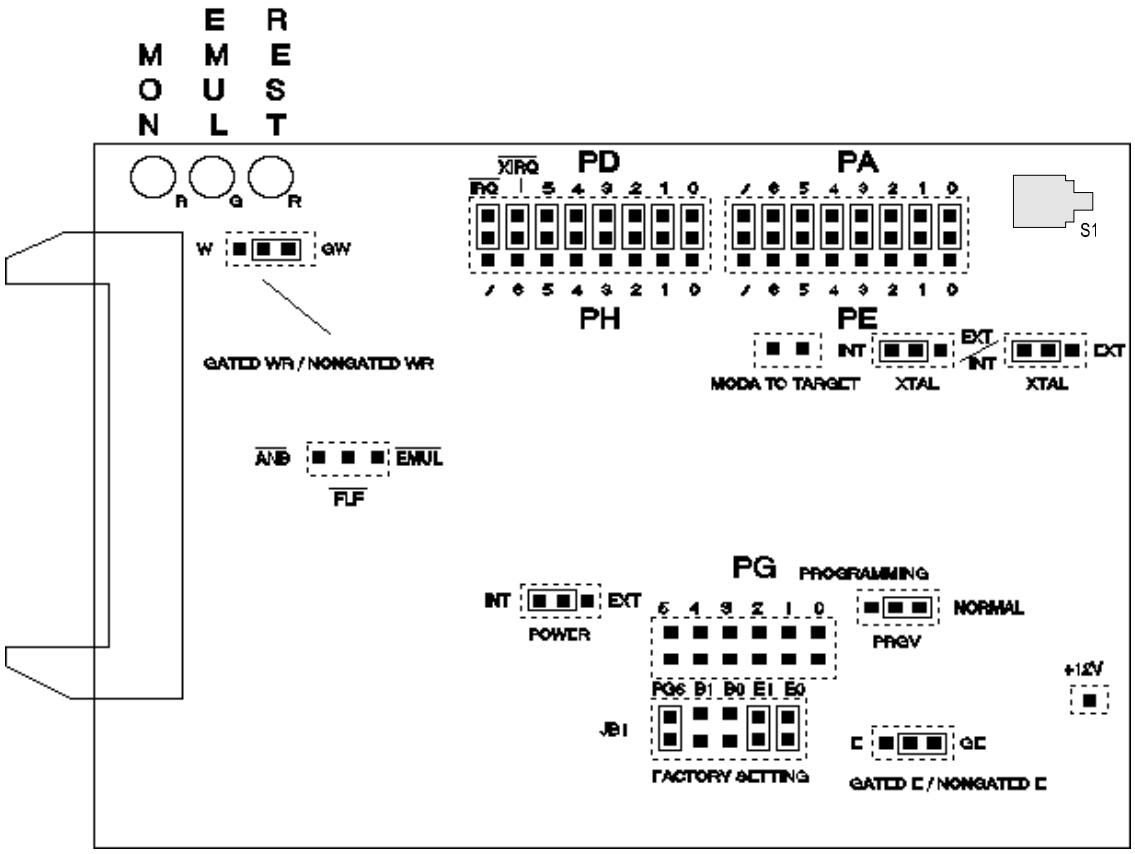

Figure 33. POD–11PE Board Configured for Target Power and Internal Crystal

POD–11PE LED Indicators and Reset Switch

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.
S1	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.

POD–11PE Configuration Options

Jumper Designation	Function	Description
XTAL	**Crystal Selection**	Two strips of three jumper pins for selecting the source of the crystal or clock: • Jumpers installed on the pins closest to the E side means the source is in the target system. • Jumpers installed on pins closest to I side means crystal on the pod is to be used.
PWR	**MCU Power Selection**	A strip of three jumper pins for selecting source of power for the MCU on the pod: • A jumper installed on the pins closest to the INT side means power comes from the emulator. • A jumper on the pins closest to the EXT side means power comes from the target system.
E, GE	**E-Clock**	A strip of three jumper pins: • If a jumper is installed on the pins closed to the E side, the E-Clock signal always goes to the target system directly from the MCU • If a jumper is installed on the pins closest to the GE side, the E-Clock goes through a gate that holds it low in monitor mode or if the memory accessed is mapped to the emulator board. (If E-Clock goes out to the target system in monitor mode, something might accidentally be written to a memory or peripheral device in the target system.) • If E is gated, there will be one shorter E pulse at the very end of the emulation mode, when a breakpoint is reached.

POD–11PE Configuration Options (continued)

Jumper Designation	Function	Description
W, GW	**R/W Line**	A strip of three pins that selects the function of the R/W line. • A jumper installed on the pins closest to W means the R/W signal to the target system will be connected directly to the R/W line on the J6 chip. • A jumper installed on the pins closest to GW means R/W will go through a gate that holds it high in monitor mode or if memory is mapped to the emulator board.
MOD A	**Isolate MOD A**	A strip of two pins. If no jumper is installed, the target system will be isolated from the processor signal MOD A. If a jumper is installed, the target system will see the processor MOD A (/LIR) (in which case the target system must not drive this signal because the /LIR signal is essential for the emulation function).
PG	**Port G**	A part of a six-pin by two-pin array of jumper pins. The upper row and the leftmost pin in the lower row carries the 68HC11P or PH series Port G pins. The upper row contains port pins as designated by the labels above it and the leftmost pin in the lower row carries bit 6 of Port G.
E1, E0	**External Signals**	A part of a six-pin by two-pin array of jumper pins. These are the two pins in the lower row designated E0 and E1. • E0 and E1 are available for connecting external signals to the emulator and are used for trace functions. • E0 can also be used in the breakpoint logic. When the bank switch emulator is used E0 and E1 also duplicates as the E0 and E1 inputs.
B0, B1	**Bank Switch Select Input**	A part of a six-pin by two-pin array. These are the two pins in the lower row designated B0 and B1. They are used with the bank switch emulator as BSW0 and BSW1 inputs.
ANB, EM	**ANB, FLF, EM**	A strip of three pins that carries signals from the emulator and trace board. • The pin closest to EM goes high in monitor mode and low in emulation mode. • The middle pin, designated FLF, goes low when the A condition in the trace board goes *true*, and goes high when the B condition goes *true*. (B is dominant, so the FLF pin will be high if the A and B conditions are both *true*. If B goes *false*, and then A goes *true*, the FLF pin will go low again.) • The signal at the pin closest to ANB behaves basically the same as FLF, but stays high after one cycle of high-low-high, independently of A and B.

POD–11PE Configuration Options (continued)

Jumper Designation	Function	Description
PE, PA	**Port E, Port A**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11P series Port A pins: 0 through 7 designate PA0 through PA7. • The row of pins toward the center of the board carries the 68HC11P series Port E pins: 0 through 7 designate PE0 through PE7. • The middle row of pins is reflected on the trace setup and display as PROBE1. External signals can be connected to these pins if the jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that Port A is traced.
PH, PD	**Port H, Port D**	An 8-pin by 3-pin array of jumper pins. • The row of pins toward the edge of the board carries the 68HC11P series Port D pins: 0 through 5 designate PD0 through PD5. The pin following PD5 is the XIRQ pin and the next pin is the IRQ pin from, the MCU. XIRQ is gated and will always be high when in monitor mode. • The row of pins toward the center of the board carries the 68HC11P series Port H pins: 0 through 7 designated PH0 through PH7. • The middle row of pins is reflected on the trace setup and display as PROBE0. External signals can be connected to these pins if jumpers are removed; the input load is one ALS input (74ALS258). When the board leaves the factory, jumpers are installed so that PD0 – PD5, XIRQ and IRQ will be traced.
PRGV	**Select XIRQ Input**	A strip of three pins that is used to connect the programming voltage to the XIRQ pin of the 68HC711KP series MCU. • In the P position the +12V pin (see Figure 33) will be connected to XIRQ and you can program the EPROM of the MCU. • In the N position the XIRQ of the MCU will be connected to the XIRQ from the target system.
+12V	**Vpp**	A single pin. Apply the externally supplied programming voltage here. For the voltage requirement, consult the latest documentation from Motorola.
JB1		This five-pin by two-pin array of jumper pins is used to control some timing parameters of the POD–11PE and should never be changed except on explicit instruction from Nohau.

Support Added for the 68HC11PH8 on Release 5.2B—Requires EPROM Version 2.4

The 68HC(7)11PH8 is now supported. EPROM version 2.4 is required. If your emulator does not have Version 2.4 you can contact Nohau or program an EPROM using a file provided by Nohau.

The Motorola S1 record file called COM24.S1 should be used with a 27C64-70 EPROM. The program's memory range is A000 to BFFF.

CAUTION

The emulator will not work if you relocate the 2K internal RAM on the 68HC11PH8 to 3000, 4000, 5000, 6000, or 7000, or if you relocate the internal EEPROM to 4D00 or 5D00.

POD–11PS

Overview

This pod board is used to emulate the Motorola microcontrollers 68HC11P and 68HC711PH series in single-chip mode.

> **Note**
>
> If you use K, N, or P family parts, SeehauHC11 will not work if the PAREN bit in CONFIG is set to 1. Be sure to set the PAREN bit to zero. PAREN activates the PPAR pull-up assignment register.

Single-chip mode is emulated using a 68HC11P or PH series MCU running in expanded multiplexed mode with Port B, Port C, Port F, and PG7 re-created using a XILINX Logical cell array. The re-created ports have a 100-Ohm series resistor to emulate the output characteristics of the 68HC11P or PH.

> **Note**
>
> The POD–11PS is a modified POD–11KS. Therefore the silk screen on the pod is labeled as POD–11KS.

Refer to Chapter 2, “Installing the Hardware.” Go to the section that covers your specific emulator board. Then refer to the jumper label SFR base address.

The J1 connector on the left side of the board is used to connect the pod with a ribbon cable to the emulator board.

The P1 connector on the solder side of the board is used to connect the pod to a target system with a PGA connector. If the target system has a PLCC socket, an adapter has to be used (available from Nohau).

The pod board has a standard 68HC11P or PH series MCU. You can replace the part in case of failure, or if you want to use another part in the same P or PH family. You can also change the XILINX part to an equivalent part in case of failure.

POD–11PS LED Indicators

Name	Function	Description
D1	**MON**	A red LED that indicates when the system is in monitor mode. MON turns off during emulation mode (Run or Go) and turns back on when a break occurs and the system returns to monitor mode.
D2	**EMUL**	A green LED that indicates when the system is in emulation mode. (It performs the inverse function of MON.)
D3	**RES**	A red LED that indicates the MCU is receiving a reset signal.

POD–11PS Configuration Options

Jumper Designation	Function	Description
EXTAL, XTAL	**Crystal Selection**	Jumpers marked EXTAL and XTAL determine if the crystal or clock is in the target system, or the crystal on the pod is used: • The jumpers should be in the right position for the crystal on the pod board (I). • The jumpers should be in the left position for target supplied crystal or clock (E).
PWR	**MCU Power Selection**	The jumper marked PWR is used for power to the on board MCU: • If the power will be supplied from the emulator, the jumper should be in the right position (INT). • If power will be supplied from the target system, the jumper should be in the left position (EXT). • The XILINX part that re-creates PB, PC, PF is always powered from the emulator.
MOD A	**Isolate MOD A**	The jumper MOD A is used to isolate the target system from the processor signal MOD A. If the jumper is in, the target system will see the processor signal MOD A (/LIR). The target system must not drive this signal, since the /LIR signal is essential for the emulator function. This jumper should normally be out.
E1, E0	**External Signals**	The jumper pins marked E0 – E1 are used to connect external signals to the emulator. These are used for the trace function. The left pin, E0, can also be used in the breakpoint logic.
EM	**ANB, FLF, EM**	The jumper marked EM carries signals from the emulator and trace board. • The left pin EM is high in monitor mode, and low in emulation mode. • The middle pin FLF, goes low, when A condition goes *true* in the trace board, and goes high when the B condition is *true*. B is dominant, so if both A and B conditions are *true*, the signal will be high. If B goes *false*, and then A goes *true*, the pin will go low again. • The right pin ANB does basically the same as FLF, but stays high after one cycle of high-low-high independent of A and B.
PA, PG	**Port A, Port G**	• The upper row of pins on jumper block PA/PG carries the 68HC11P series MCU Port A pins, and the lower row of pins carries the Port G pins. PA0 and PG0 are on the left, and PA7 and PG7 are on the right. • The middle pins are reflected on the trace display. The board is delivered with jumpers, so that PA0-PA7 will be traced. External signals can be connected to the middle pins if the jumpers to Port A are removed. The input load is one ALS input (74ALS258).

POD–11PS Configuration Options (continued)

Jumper Designation	Function	Description
PD, PH	**Port D, Port H**	• The upper row of pins on jumper block PD/PH carries the 68HC11P series MCU Port D pins plus XIRQ/ and IRQ/. • The lower row of pins carries the Port H pins. PD0 and PH0 are on the left, and IRQ/ and PH7 are on the right. • The middle pins are reflected on the trace display. The board is delivered with jumpers, so that PD0-PD5, XIRQ/ and IRQ/ will be traced. External signals can be connected to the middle pins if the jumpers to Port D are removed. The input load is one ALS input (74ALS258).
PRGV	**Select XIRQ Input**	The jumper PRGV is used to select if the MCU pin XIRQ will be connected to programming voltage (left P position), or in normal position (right N position). The pin +12V is used for the externally supplied programming voltage. Consult the latest documentation from Motorola for the voltage requirement.
PC/PB, PE/PF	**Ports C, B and Ports E, F**	Jumper blocks PC/PB and PE/PF carry the corresponding port pins with pin 0 on the left and pin 7 on the right.
JB1		Jumper block JB1 is factory set and should not be changed.
S1	**Reset**	Resets the MCU. The S1 switch can be used instead of the target reset.

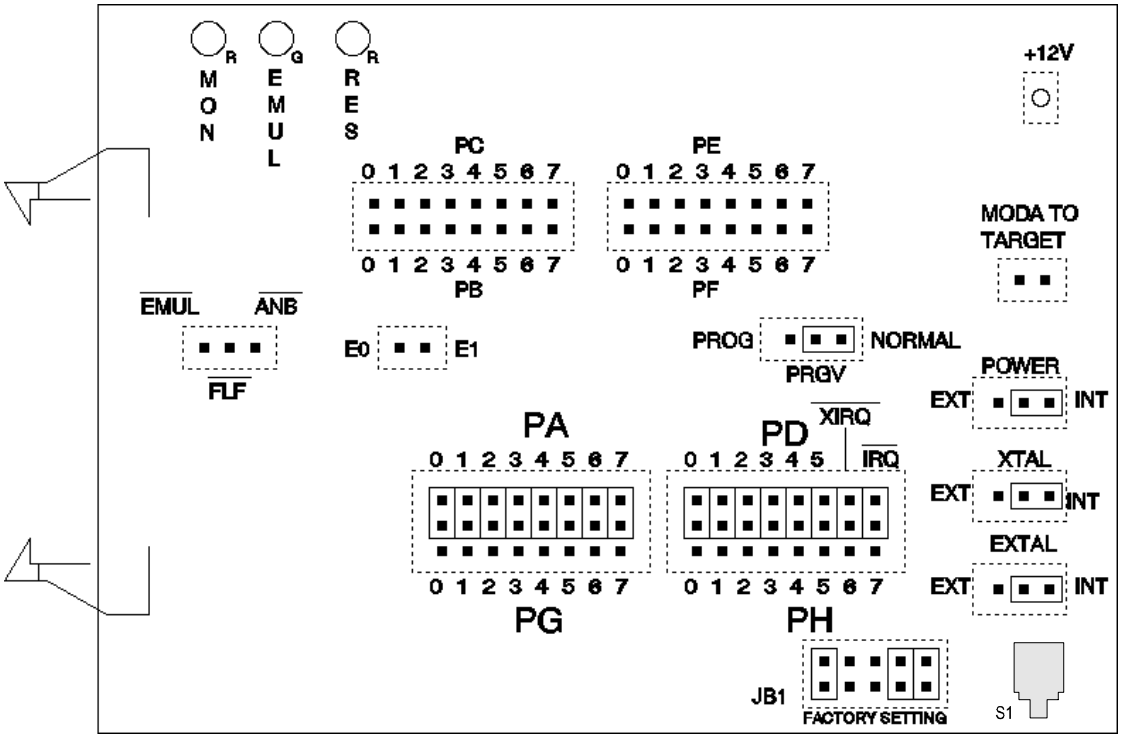

Figure 34. POD–11PS Board Configured for Target Power and Internal Crystal

Port G Bit 7 Must be Used as an Output

If you use a POD–11PS, Port G bit 7 must be used as an output, and not as an input. This is a limitation in the design of the emulator. For expanded mode pods, this limitation does not apply.

Support Added for the 68HC11PH8 MCU on Release 5.2B— Requires EPROM Version 2.4

The 68HC(7)11PH8 is now supported. EPROM version 2.4 is required. If your emulator does not have Version 2.4 you can contact Nohau or program an EPROM using a file provided by Nohau.

The Motorola S1 record file called COM24.S1 should be used with a 27C64-70 EPROM. The program's memory range is A000 to BFFF.

CAUTION

The emulator will not work if you relocate the 2K internal RAM on the PH8 to 3000, 4000, 5000, 6000, or 7000, or if you relocate the internal EEPROM to 4D00 or 5D00.

POD–11S

Overview

Use this pod if you are running a 68HC11 processor in single-chip mode or in special bootstrap mode (internal memory). The following descriptions apply to A-series and E-series MCUs.

For the E9 and 811E2, also refer to the section “Special Considerations for 68HC11 E-series” later in this chapter.

Figure 35 shows how switches and jumpers are set when the board is shipped from the factory.

Changing the Pod MCU

Pod boards contain the 68HC11A1 PLCC chip. To change the pod PLCC chip to the 68HC11E0, 68HC11E1, 68HC811E2, or 68HC11E9 PLCC part, the QILEXT-1 extractor tool is recommended.

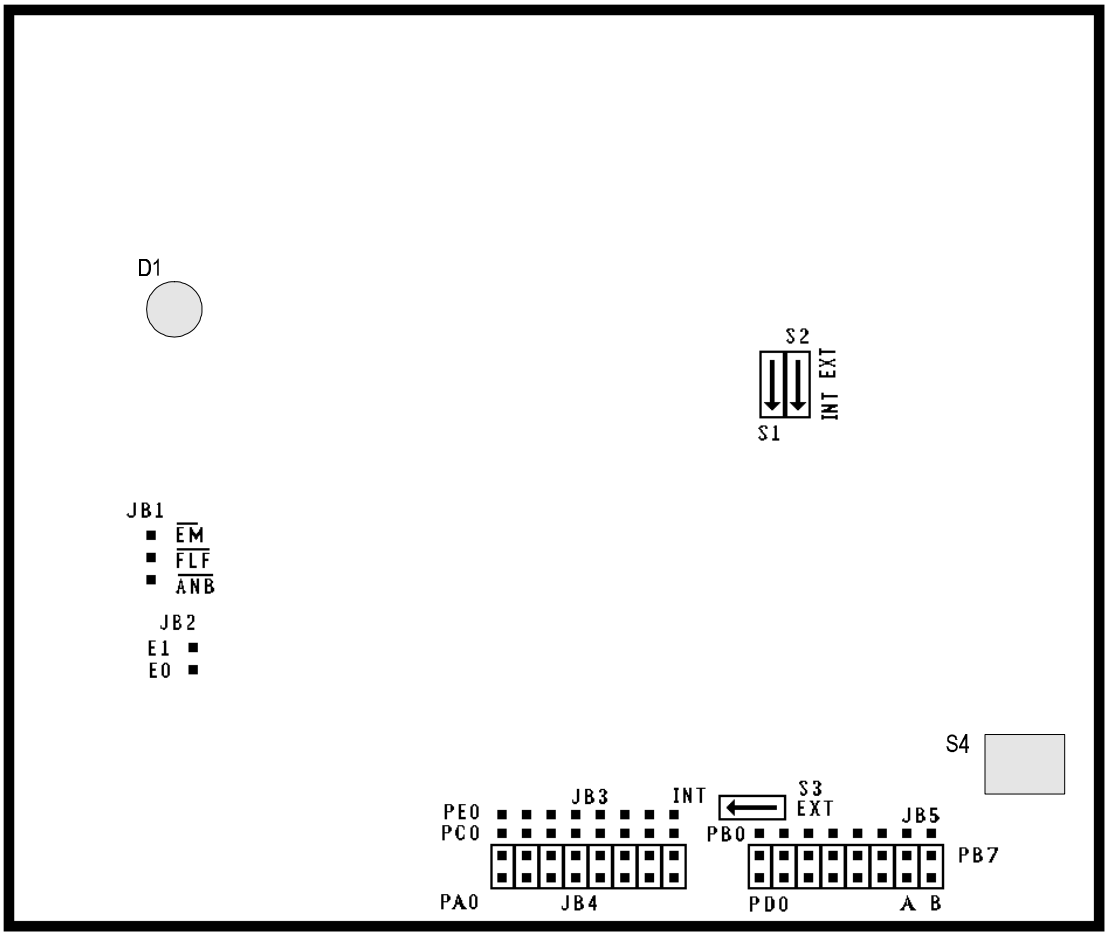

Figure 35. POD–11S Board Configured for Target Power and Internal Crystal

POD–11S Configuration Options

Jumper Designation	Function	Description
S1, S2	**Crystal Selection**	Two 2-position slide switches: • If both are set in the position marked INT, the on-board pod crystal is connected to the oscillator pins. • If both are set in the EXT position, the crystal on the target system or clock is connected to the oscillator pins.
S3	**MCU Power Selection**	A two-position slide switch: • If set in INT position, the 68HC11 CPU receives power from the PC. • If set in EXT position, the CPU receives power from the target system.
JB1	**EM, FLF, ANB**	A three-row strip of pins that carries signals from the emulator and trace boards. • EM is high in monitor mode and low in emulation mode. **Note:** On some boards this pin is incorrectly labeled EN. • FLF goes low when the A condition is *true* on the trace board and goes high when the B condition is *true*. B is dominant; so if both A and B conditions are *true*, FLF will be high. If B goes *false* and A is *true*, FLF will go low again. • ANB is similar to FLF, except that ANB stays high after one cycle of high-low-high, independently of A and B.
JB2	**E0, E1**	A two-pin strip. E0 and E1 are available for connecting external signals to the emulator and are used for the trace function. E0 can also be used in the breakpoint logic.
JB4, JB3	**Port A, Port C**	JB4 is an 8-pin by 3-pin array of jumper pins: • The lower row, PA0 – PA7, carries the 68HC11 Port A pins. • The upper row, PC0 – PC7, carries the Port C signals coming from the 68HC24. The 68HC24 is used to emulate Port B, Port C, STRA and STRB. The level connected to the middle row of pins is reflected on the trace display. The input load is one ALS input (74ALS258). • When shipped from the factory, jumpers are set up between pins in the middle and lower rows so that normally Port A will be traced. To trace Port C instead of Port A, move the jumpers, and connect the pins in the middle and upper rows. To trace external signals, connect the pins in the middle row in place of the jumpers. • JB4 is called PROBE0 in the **Trace Setup** menu and the Qualifier Register command. • JB3 is a single row of eight pins that carries Port E signals. To trace them, connect external wires between the PE0 – PE7 pins and the middle row of pins on JB4.

POD–11S Configuration Options (continued)

Jumper Designation	Function	Description
JB5	**Port D, Port B**	An 8-pin by 3-pin array of jumper pins. • In the upper row of pins, PB0 – PB7 carry the Port B signals from the 68HC24. • The middle row of pins goes to the trace inputs. In the lower row of pins, PD0 – PD5 come from the 68HC11. • STRA and STRB come from the 68HC24. • When shipped from the factory, jumpers are set up between pins in the middle and lower rows. You can move the jumpers to the upper row, or connect external signals to the middle row. JB5 is called PROBE1 in the **Trace Setup** menu and the Qualifier Register command.
S4	**RESET Switch**	Resets the MCU. The S1 switch can be used instead of the target reset.
D1	**Status**	An LED that lights when the emulator is running. After power is supplied from the emulator, the LED stays off until emulation is started. The LED turns off when a breakpoint is reached, and the system returns to monitor mode.

Note

Connectors J2 and J3 mate with the adapter. The white line on the adapter should be aligned with the white line on the pod board.

The J1 connector mates with the ribbon cable from the emulator board. The J2 and J3 connectors mate with a socket adapter that plugs into the solder side of the pod board. Socket adapters are available for mating with a DIP or PLCC socket on the target system.

Using EMUL68 with an HC(7)11E20 MCU

The EMUL68 emulator will not function if an E20 MCU is inserted into the CPU socket on the pod and the ROMON bit in the chip's CONFIG register is set to 1. This is caused by the very large on-chip ROM on the HC11E20 (EPROM on the HC711E20).

There is currently no way to change the ROMON bit for an E20 chip from 1 to 0 when using a Nohau emulator. The emulator is not functional with ROMON set to 1.

If you want to use an actual E20 chip in the Nohau pod, you must set the ROMON bit to zero before inserting the chip into the pod. This can be done with a Motorola EVS for the E series, and some universal programmers. Most Motorola distributors should be able to perform this service for you.

Alternatively, you can use an HC11E0, E1 or E9 in the pod. These parts have 512 bytes of on-chip RAM as opposed to 768 for the E20; this is the only known difference. If you use the extra 256 bytes of RAM from 200 to 2FF, you might be able to map the memory space to emulation memory and let the emulator's RAM emulate the extra 256 bytes. Since the standard emulator board's mapper resolution is 4K, you will need to map the entire area from 0 to 0FFF to emulation memory to do this. This strategy will not work if you have a memory map device in the 0 to 0FFF range.

The EMUL68 emulator cannot currently program the EPROM on an HC711E20; the EPROM must be enabled in order to program it. Until a strategy to program the E20 with an EMUL68 emulator can be found, use a Motorola EVS or a suitable universal programmer instead.

Special Considerations for 68HC11 E-Series

The following describes some special considerations for the -811E2 and -E9, -E0, -E1 versions in POD–11S and POD–11E.

BPROT Register

Note that all 68HC11 E-series MCUs discussed here have a Bank Protection Register (BPROT). If you want to program the EEPROM, zeros must be written to this register. When emulating, the BPROT register should be initialized correctly by the user software. When not emulating, you can change the BPROT register by entering a value in the BPROT override field found in the **Hardware Configuration** dialog box. Then click **OK** or **Apply**. Anytime you reset the emulator, the BPROT register is reloaded with your override value.

EEPROM Programming

The EEPROM is automatically programmed when data is written within the EEPROM area by the emulator. The only requirement is that the BPROT register enables programming and EEPROM is enabled. If you frequently need to program new devices, Nohau provides a ZIF socket adapter to prevent the socket on the pod from wearing out prematurely.

EPROM Programming

> **Note**
>
> Currently Seehau is not capable of programming EPROMs. Refer to the documentation included with the EMUL68 for DOS software.

The POD–11E and POD–11S do not provide any connection for the programming voltage. Use the special adapter (Part #EMUL68–PC/PLCC52ZIF–PLCC52). This adapter has a zero insertion force socket with a connector.

Using the Watchdog COP Reset

Due to the extra capacitance that is on the reset line of the pod boards, the COP reset will not operate correctly when the pod is working in stand-alone mode. To correct this limitation, you can attach a 10-kohm pull-up resistor to the reset line. When the COP resets the processor, it will output a low level on the reset line for four E cycles and then release the reset line. Two E cycles later, the MCU monitors the reset line again. If the reset line is still low, it is considered to be an external reset. If it is high, it is considered to be a COP reset. If the pull-up is not used, the capacitance will keep the reset line low during these two E cycles, and the processor will consider it an external reset.

Refer to the following table for adding the 10K-Ohm pull-up resistor to your pod board.

Pull-Up Chart

Pod Type	IC #	Pin #	to Pin #
POD–11DE	U10	2	14
POD–11DS	U9	2	14
POD–11E	U6	9	14
POD–11FE	U9	2	14
POD–FL0	U8	7	20
POD–11KE	U9	2	14
POD–11KS	U2	2	14
POD–11PE	U9	2	14
POD–11PS	U2	2	14
POD–11S	U3	2	14

Changing the Pod MCU

Overview

There are several reasons you might want to install a new MCU in your pod board:

- For more accurate emulation, you might want to install an MCU that is identical to the one in your target.
- The pod's MCU might be damaged.
- You might want to program the MCU for use on your target without the emulator.

Installing and Configuring the New MCU

Note

If you have a POD–11DE or 11DS, do **Step 1** only.

For all other chips, do **Step 1**. Then try starting the emulator. If you do not receive any error messages, you can skip **Step 2** through **Step 9**.

If you do receive error messages, do the following steps.

If your new MCU contains EPROM, you will need to disable the EPROM before the chip can be used in your emulator. Do the following steps:

1. Remove the existing MCU. Then install the new MCU. To remove the pod MCU in the PLCC chip package, use the QILEXT-1 extractor tool. (Refer to the Nohau Price List for further details about this tool.)

2. From the **Start** menu, select **Programs**. Then select the Seehau HC11 program group, and click **Reconfig**. The **Emulator Configuration Communications** dialog box opens.

3. Go to Region 3, **Processor Type**, and verify that the processor type matches your new MCU. Then click **Next** until the **Hdw Config** dialog box appears.

4. In the **Hdw Config** dialog box, select the **Test Mode after Reset** option. Then click **Finish**. Now start Seehau HC11.

5. From the Register window, right-click to open the Local menu. Click **Add Special Register**. The SFR window opens.

6. Double-click **Special**, and a list of registers appears. If listed, click the BPROT register. In the lower right region of the SFR window, the current BPROT value appears. Enter zero in the **Value** field. Then click **Change**.

7. From the register list on the left side of the SFR window, scroll down and double-click **CONFIG**.

   Review the following configuration bits and set appropriate values as follows:

   - **EEON bit**: If the MCU EEPROM is located in the reset vector region, you must set the EEON bit to zero.
   - **ROMON bit**: Set the ROMON bit to zero to diable the EPROM.
   - **NOCOP bit**: Normally, set this bit to one. If you are debugging the COP, you can set this bit to zero, but you must select the **COP Kicking Enabled** option in the **Hdw Config** dialog box.

8. From the Seehau Speedbar, click the Reset button. 

9. From the **Hdw Config** dialog box, clear the **Test Mode after Reset** option. Then click **OK**.

4 Installing and Configuring the Seehau Software

Installing Seehau Software from the CD

To install the Seehau software, do the following:

1. Locate your Seehau CD and insert the CD into your CD ROM drive. The installation process will start automatically.
2. Follow the instructions that appear on your screen.

> **Note**
>
> If the installation does not start automatically, you probably have your Windows Autorun feature disabled. You will then need to use Windows Explorer and navigate to the CD root directory. Double-click **Install**.

Downloading and Installing Seehau from the Internet

1. Go to the Nohau web site (http://www.icetech.com/). Click **USA**.
2. Click **Downloads**. The Nohau Software Downloads page opens.
3. Click **Current Software Versions**. The Current Software Versions page opens displaying a table with listings of current software versions.
4. Locate the EMUL68–PC product listing. In the Information and Software Download column, click **Seehau**.
5. Review the “Known Issues” section.
6. Click **Yes I Want to Download**. A Customer Information Form page opens. Complete this form, then click **Proceed**.
7. Click **Go to Download**. Click either option for a download site. The Nohau Software Updates page opens.
8. Select the **Download Site** option, then click **EMUL68–PC**.
9. Click **S6811.exe**. The application will start downloading.
10. After downloading the application, click **S6811.exe** and follow the installation instructions.

Note

After installing the Seehau software, the **Setup Complete** dialog box opens where you can review the Readme.txt file and/or launch the Seehau HC11 configuration.

If you are installing Seehau for the first time, you must launch the Seehau HC11 configuration before running the Seehau software.

Selecting to Automatically Start the Seehau Configuration Program

After installing Seehau, it is recommended that you automatically start the Seehau Configuration program. Do the following steps before starting Seehau:

1. From the **Setup Complete** dialog box, select **Launch Seehau68 Configuration**.
2. Click **Finish**.

If you do not select to automatically start the Seehau Configuration Program, do the following:

1. From the **Start** menu, select **Programs**.
2. Select **SeehauHC11**. Then click **Config** to open the **Emulator Configuration Communications** dialog box (Figure 36).

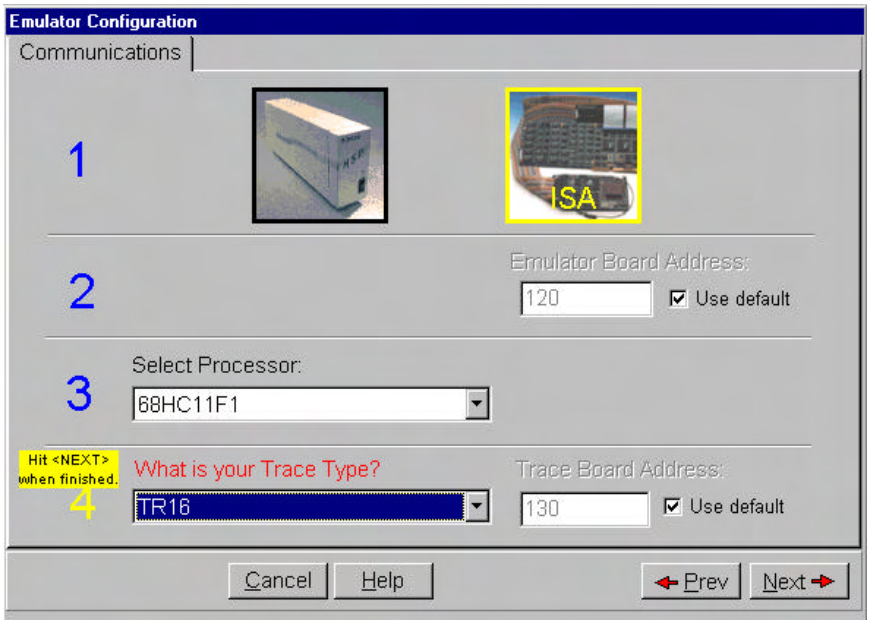

Figure 36. Emulator Configuration Communications Dialog Box

Note

You do not need the hardware connected at this time.

Configuring Seehau

First select one of the following communication interfaces:

- **ISA Board**—Communicates with the emulator system through the PC's ISA bus.
- **High Speed Parallel Box (HSP)**—The emulator is placed in an external box and communicates with the PC through a standard PC parallel port (LPTx).

Note

The serial expansion box called BOX–S is now obsolete and is not supported by Seehau. If you have a BOX–S, you can purchase a conversion kit to upgrade to an HSP box.

To configure your software, do the following steps:

1. From the **Start** menu, select **Programs**.
2. Select **SeehauHC11**, and click **Config**. The **Emulator Configuration Communications** dialog box opens.

As an example of setting up your configuration, the ISA communications interface is shown in Figure 36. The HSP communications interface is very similar, and the steps you do when using the HSP communications interface are almost identical.

The graphical user interface for this dialog box is divided into four regions. Do the following in each region:

1. Region 1—**Communications Interface**:
   Select either the ISA or HSP communications interface.
2. Region 2—**Emulator Board Address**:
   Contains the address of the internal communication link from your computer. For the ISA card, the default address is 120. To disable this default, clear the **Default** option and insert the appropriate address for the emulator board.
3. Region 3—**Select Processor**:
   Click the down arrow and select the processor type you are using.
4. Region 4—**Trace Board**:
   Click the down arrow and select your trace type. If you do not have a trace board, select **None**.
5. Click **Next**. The **Emulator Configuration, Hdw Config** tab opens.

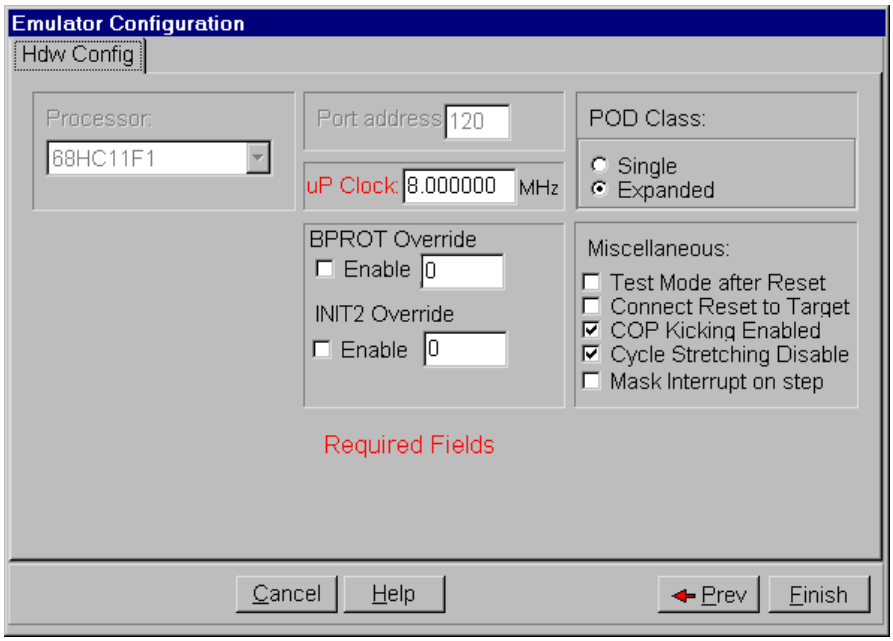

Figure 37. Emulator Configuration, Hdw Config Tab

Complete the following fields in the **Hdw Config** tab:

- **Processor**: Shown for reference only. If you need to change the processor type, click **Prev**.
- **Port address**: Shown for reference only. If you need to change the port address, click **Prev**.
- **uP Clock**: Set this frequency to E-clock * 4. For most pods, the stand-alone crystal frequency is 8 MHz. If you select the target clock jumpers, you need to enter your target frequency in the field.
- **BPROT Override**: If this option is enabled, the emulator automatically presets the BPROT SFR to the specified value when an emulator reset occurs. Valid only for processors with the BPROT register.
- **INIT2 Override**: If this option is enabled, the emulator automatically presets the INIT2 SFR to the specified value when an emulator reset occurs. Valid only for processors with the INIT2 register.
- **POD Class**: Verify the pod class matches your MCU mode. This must also match the pod type.
- **Miscellaneous**:
  - **Test Mode after Reset**: Select this option if you want to enable test mode after reset. This option might be necessary when changing the pod MCU. (Refer to the "Changing the Pod MCU" section in Chapter 3, "Installing the Pod Boards.") This option might be helpful for troubleshooting when the emulator will not start. Allows access to certain SFRs. (Refer to Motorola's MCU documentation for further details.)

- **Connect Reset to Target**: By default, the reset signal from the target is not connected to the emulator. Select this option if you want to connect the target reset to the emulator.
- **COP Kicking Enabled**: If you enable the 68HC11 watchdog feature called COP, you must enable the **COP Kicking** option. This allows the monitor code to feed the watchdog while in monitor mode. Select this option when using a D or FL0 family processor or whenever the NOCOP bit in the processor's Config register is set to zero.
- **Cycle Stretching Disable**: For MCUs that support cycle stretching, you can automatically disable cycle stretching upon reset with this option. For single-chip applications cycle stretching is normally automatically disabled on chip. Because Nohau single-chip pods actually use the MCU in expanded mode with port replacement, it is necessary to disable cycle stretching using this feature to maintain proper bus timing.
- **Mask Interrupt on step**: This function allows single-stepping through your non-interrupt source code without servicing frequent or time-based interrupts. (This does not effect the Go commands.)

- **Finish**: Click to save the configuration and exit the dialog box. A window appears asking whether you want to start the emulator. Select **Yes** to launch Seehau. Select **No** to exit Seehau Configuration.

WARNING

- To avoid damage to the pod or to your target, do not connect the pod to your target when the pod or target power is on
- Make sure you always power up the emulator first followed by the target system. When powering down, power down the target system first followed by the emulator. Failing to do so can cause damage to your target and/or emulator.

Starting Seehau

Configuring the Emulator Options From Within Seehau

From Seehau open the Emulator Configuration window. Select the **Config** menu and click **Emulator**. The **Emulator Configuration** dialog box opens (Figure 37).

There are six tabs across the top of the main **Emulator Configuration** window. When selected, each tab allows you to access the following dialog boxes:

Hdw Config:	Set up emulator hardware options.
Misc. Config:	Select reset options and tab size.
Mem Map Config:	Map address ranges to the emulator or target with or without write protection.
Banking:	Configure the bank switching logic.
Breakpoint:	Set up special breakpoint options.
Eprom:	Program the on-chip EPROM. (Not implemented yet.)

Hardware Configuration Tab

- **Processor**: Shown for reference only. If you need to change the processor type, first exit Seehau. Then do the following: From the **Start** menu, select **Programs**. Then select **Seehau HC11**, and click **Config** to delete current settings or **Reconfig** to retain your initial hardware configuration.

- **Port address**: Shown for reference only. If you need to change the port address, do the following: From the **Start** menu, select **Programs**. Select **Seehau HC11**, then click **Config** to delete current settings or **Reconfig** to retain your initial hardware configuration.

- **uP Clock**: Set this frequency to E-clock * 4. For most pods, the stand-alone crystal frequency is 8 MHz. If you select the target clock jumpers, you need to enter your target frequency in this field.

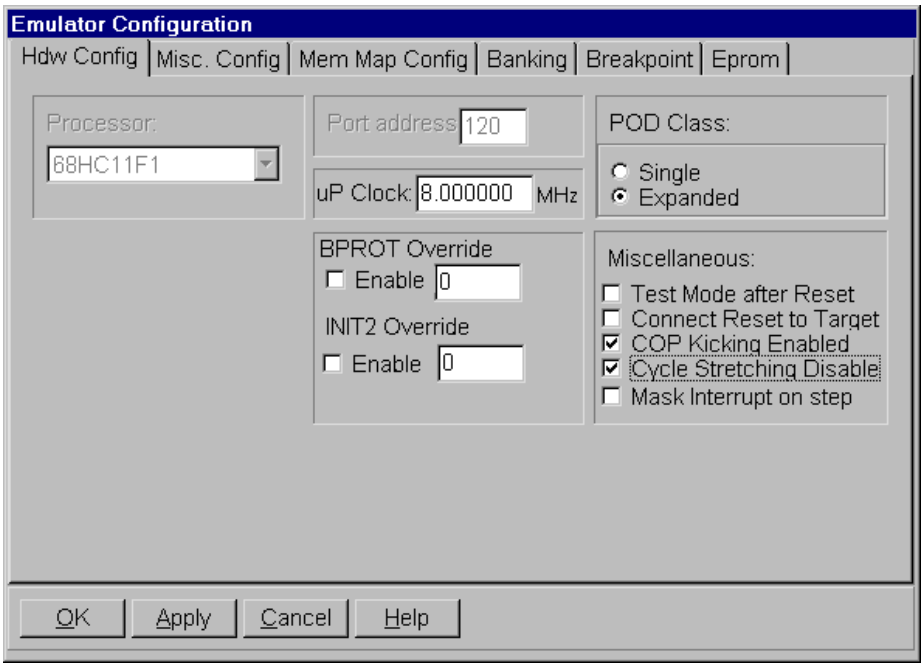

Figure 38. Hardware Configuration Tab

- **BPROT Override**: If this option is enabled, the emulator automatically presets the BPROT SFR to the specified value when an emulator reset occurs. Valid only for processors with the BPROT register.

- **INIT2 Override**: If this option is enabled, the emulator automatically presets the INIT2 SFR to the specified value when an emulator reset occurs. Valid only for processors with the INIT2 register.

- **POD Class**: Verify the pod class matches your MCU mode. This must also match the pod type.

- **Miscellaneous**:

  - **Test Mode after Reset**: Select this option if you want to enable test mode after reset. Allows access to certain SFRs. (Refer to Motorola's MCU documentation for further details.)

  - **Connect Reset to Target**: By default, the reset signal from the target is not connected to the emulator. Select this option if you want to connect the target reset to the emulator.

  - **COP Kicking Enabled**: If you enable the 68HC11 watchdog feature called COP, you must enable the **COP Kicking** option. This allows the monitor code to feed the watchdog while in monitor mode. Select this option when using a D family processor or whenever the NOCOP bit in the processor's CONFIG register is set to zero.

  - **Cycle Stretching Disable**: For MCUs that support cycle stretching, you can automatically disable cycle stretching upon reset with this option. For single-chip applications cycle stretching is normally automatically disabled on chip. Because Nohau single-chip pods actually use the MCU in expanded mode with port replacement, it is necessary to disable cycle stretching using this feature to maintain proper bus timing.

  - **Mask Interrupt on step**: This function allows single-stepping through your non-interrupt source code without servicing frequent or time-based interrupts. (This does not effect the Go commands.)

Miscellaneous Configuration Tab

- **Reset chip after load file**: Select this option if you want to have the emulator automatically reset the MCU after loading a code file.

- **Tab Size**: Enter a number to set tab spaces for source code display. The default is set at eight. If your source code uses many layers of nesting, use a smaller number than the default. This allows easier viewing of source code.

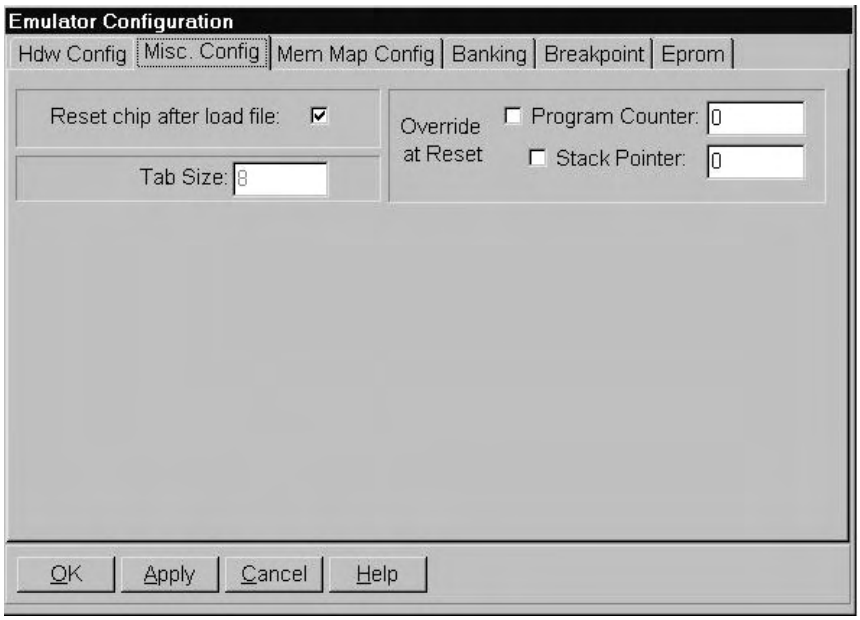

Figure 39. Miscellaneous Configuration Tab

- **Override at Reset**:
  - **Program Counter**: Select this option if you want to pre-set the program counter to a specified value upon emulator reset.
  - **Stack Pointer**: Select this option if you want to pre-set the stack pointer to a specified value upon emulator reset.

Memory Map Configuration Tab

By default, all memory is mapped to the emulator. To access memory mapped target resources, you must map memory ranges to the target. This section describes how to set up memory mapping for the standard and bank switched emulator boards.

Memory Mapping Requirements for Single-Chip Mode Pods

For single-chip mode pods, the memory range covered by the first 64 bytes of the special function registers must be mapped to target and not to the emulator if Ports B and C (and also Port F for non-multiplexed bus parts) are being used for parallel I/O pins. This also covers the port containing the pin used for the R/W bar signal.

Thus, on A, E, F and similar parts, the range 1000 to 103F (1000 to 1FFF for standard emulator boards) must be mapped to target. For D, K, FL0, and P parts, the range 0 to 3F (0 to FFF for standard emulator boards) must be mapped to target (assuming you do not move the SFRs by writing to the INIT register).

Ports B and C (and for non-multiplexed bus parts also Port F) and the port containing the pin used for R/W bar in expanded mode will not work correctly for parallel I/O if the corresponding addresses by which they are accessed are mapped to emulator. This is a design characteristic of the emulator and pod.

The default memory map in the Seehau software is set for all memory to be mapped to emulator regardless of pod type. If you use a single-chip mode pod, be sure to change the memory map as previously described. This is not an issue in the DOS software; selecting a single-chip mode pod results in a default memory map that always sets the SFRs to default to target mapping.

Standard Emulator Board

If you have a standard emulator board, do the following:

1. From the **Emulator Configuration** dialog box, click the **Mem Map Config** tab (Figure 40).

2. From the **Mem Map Config** tab, click **Add** to add a memory range. The **EditAddMapHC11** dialog box opens (Figure 41).

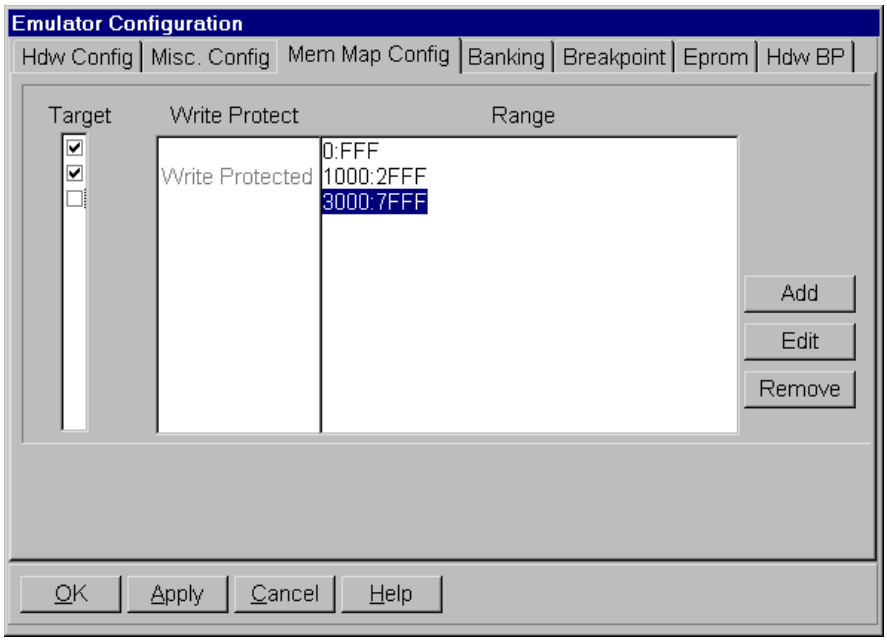

Figure 40. Memory Map Configuration Tab for the Standard Board

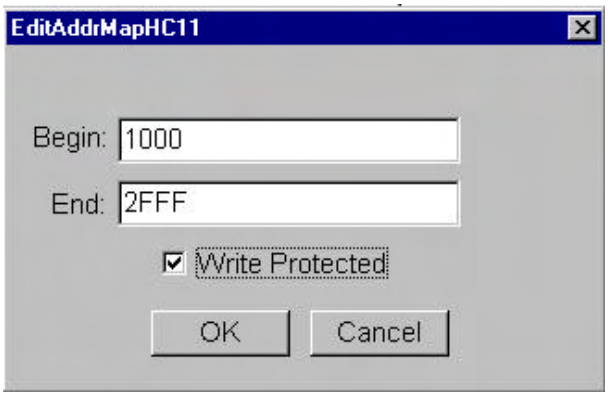

Figure 41. Add Dialog Box

3. In the **EditAddMapHC11** dialog box, enter the **Begin** and **End** range.

   For the standard emulator board, you must map memory on even 4K (1000H) byte boundaries. Use the following formula:

   Valid Begin address: x000
   Valid End address: xFFF

   (where x=0-F)

4. Select **Write Protected** to cause a break to occur if a write is detected in this address range (currently not supported for POD–11FL0).
5. Click **OK**. The Memory Mapping window is updated automatically with your new range.

   The map-to-target option is selected by default and indicates the range is mapped to the target. To map to the emulator, clear this option.

Bank-Switched Emulator Board

(includes 256/512/1M boards)

> **Tip**
>
> For most applications, course resolution mapping is recommended, such as in the example for the standard emulator board. Maximum mapping resolution is possible down to 64-byte block resolution when appropriate.

If you have a bank-switched emulator board, do the following:

1. From the **Emulator Configuration** dialog box, click the **Mem Map Config** tab. The **Mem Map Config** view appears (Figure 42).

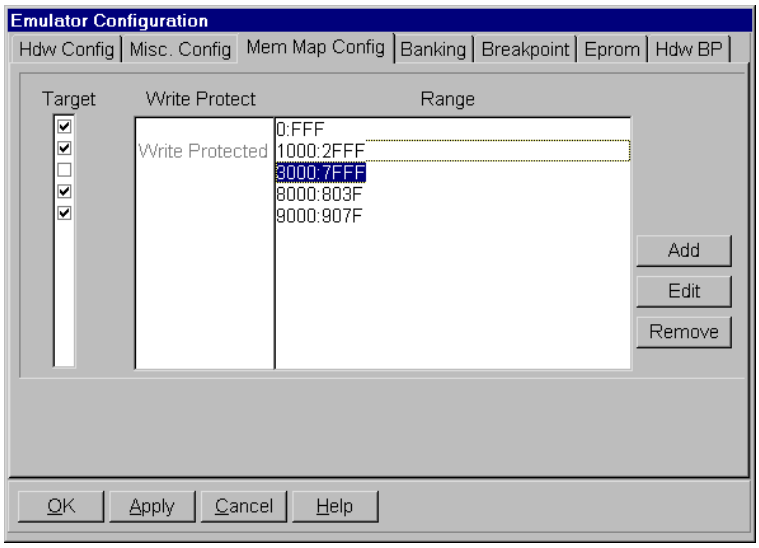

Figure 42. Memory Mapping Tab for the Bank-Switched Board

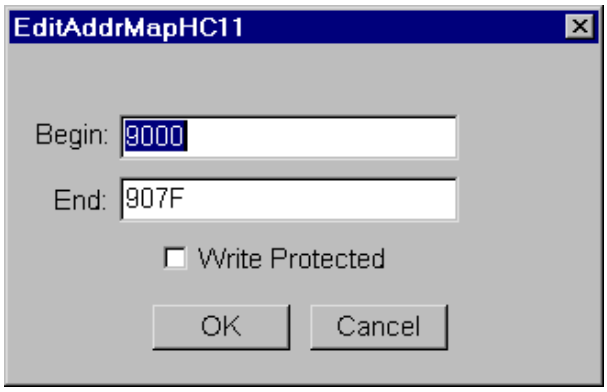

Figure 43. Add Dialog Box for the Bank-Switched Board

2. In the Memory Mapping window, click **Add** to add a memory range. The **EditAddMapHC11** dialog box opens (Figure 43).

3. In the **EditAddMapHC11** dialog box, enter the **Begin** and **End** range. Seehau automatically rounds to the nearest 40H boundary.

4. Select **Write Protected** to cause a break to occur if a write is detected in this address range.

   If the map-to-target option is selected or cleared, once a range is write protected, it will remain so until edited or removed.

5. Click **OK**.

   The Memory Mapping window is updated automatically with your new range.

Emulator Cannot Load Code Into Write Protected Memory Space

A minor limitation in the Seehau software does not allow instructions or data to be written to write protected memory using the Load command. Therefore, if you want to write protect part or all of your memory space during emulation, you must disable write protection during the process of loading your program. You can do this with a Seehau macro by doing the following steps:

1. Disable write protection for your code space.
2. Load your application code.
3. Enable write protection for your code space.

Banking Tab

- **Bank Area**:
  - **Enable Banking**: Select this option to enable bank switching (required).
  - **Range**: Enter hex values for the begin and end address of your bank-switched area (required).

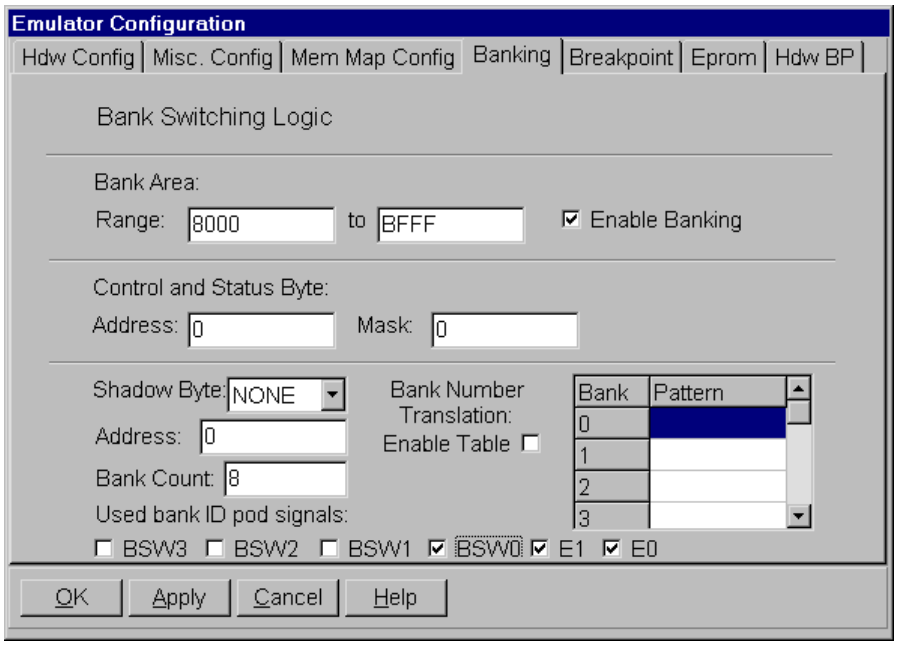

Figure 44. Banking Tab

> **Note**
>
> If you are using a pod that supports internal bank switching (POD–11KE, POD–11FL0), ignore the following fields:
>
> - **Control and Status Byte**
> - **Shadow Byte**
> - **Address**
> - **Bank Number Translation**
> - **Enable Table**

- **Control and Status Byte**: (currently not implemented)
  - **Address**: If you are using a memory location to control bank switching such as a port SFR, enter the hex value.
  - **Mask**: Enter the hex value bit mask used by the bank-switched logic. For example, if you are using the three least significant bits of Port A, the address=1000H and the mask=7H.
- **Shadow Byte**: (currently not implemented) If the control byte is write only, specify a shadow byte location that contains the same value as the control byte. The emulator uses this byte to identify which bank is currently active when entering monitor mode. Click the down arrow and select either SFR or DATA space as the location of your shadow byte.
- **Address**: Enter the address of the shadow byte.
- **Bank Count**: Enter the number of banks your application uses (required).

- **Used bank ID pod signals**: Use the following table to determine which pod signals to select (required):

	Signals Used					
# of Banks	BSW3	BSW2	BSW1	BSW0	(E1)	(E0)
2						✓
3 to 4					✓	✓
5 to 8				✓	✓	✓
9 to 16			✓	✓	✓	✓
17 to 32		✓	✓	✓	✓	✓
33 to 64	✓	✓	✓	✓	✓	✓

- **Bank Number Translation**: Usually the bank number directly corresponds to the bit pattern of the control byte (for example, bank0=000; bank1=001; bank2=010). If the bank number does not correspond to the bit pattern, you need to complete the bank number translation table (for example, bank0=111; bank1=110; bank2=101).

Breakpoint Tab

The EMUL68 family supports hardware breakpoint ranges that allow you to specify bus cycle type. Opcode fetches can also be logically ANDed with external inputs E0 and E1.

The **Breakpoint** tab contains three columns (Figure 45):

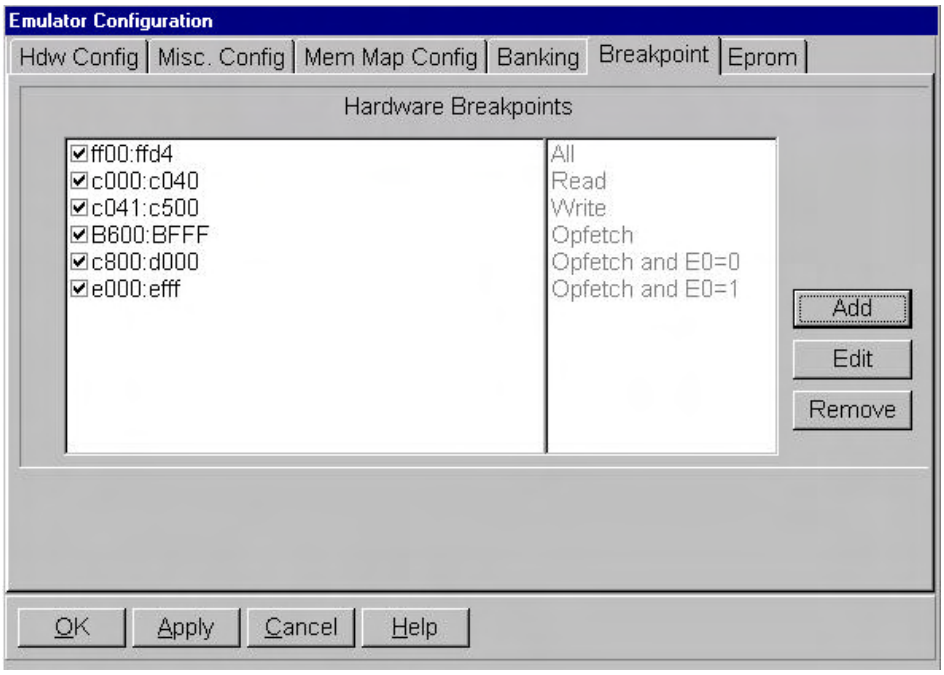

Figure 45. Breakpoint Tab

- **Check Box**: By default, the breakpoint range is selected. To disable the range, clear the breakpoint range.

- **Address Range**: Previously entered ranges are displayed here for reference only. To modify, click to select the entry. Then click **Edit**. The **EditAddrHC11** dialog box opens.

- **Cycle Type**: Previously entered cycle types are displayed here for reference only. To modify, click to select the entry. Then click **Edit**. The **EditAddrHC11** dialog box opens.

To enter a new breakpoint range, click **Add**. The **EditAddrHC11** dialog box opens (Figure 46).

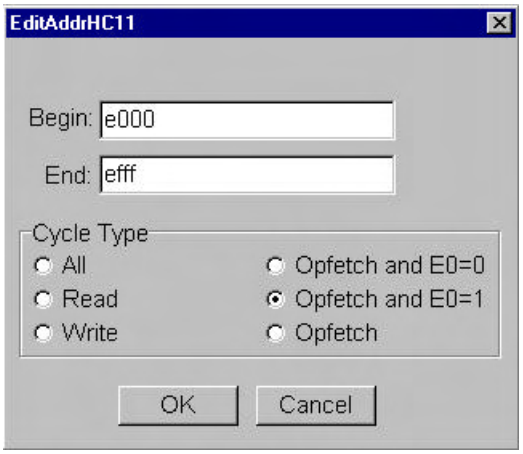

Figure 46. EditAddrHC11 Dialog Box

- **Begin**: Enter the Begin address in hex.

- **End**: Enter the End address in hex.

- **Cycle Type**: Click appropriate cycle type.
  - **All**: Select **All** to break on a read/write or opfetch in the specified range.
  - **Read**: Select **Read** to break on a read in the specified range.
  - **Write**: Select **Write** to break on a read in the specified range.
  - **Opfetch and E0=0**: Select **Opfetch and E0=0** to break on an opfetch in the specified range only when pod input E0=0.
  - **Opfetch and E0=1**: Select **Opfetch and E0=1** to break on an opfetch in the specified range only when pod input E0=1.
  - **Opfetch**: Select **Opfetch** to break on a code fetch in the specified range.

EPROM Tab

(Currently Not Implemented)

5 Connecting the Emulator to Your Target Board

WARNING

Make sure you always do the following: power up the emulator first followed by the target system. When powering down, power down the target system first followed by the emulator. Failing to do so can cause damage to your target and/or emulator.

Plugging the Emulator into Your Target Board

Support for 3.3 Volts

Nohau pods can support the 68L11 series 3.3V chips in expanded mode only. The customer must supply the MCU. Special adapters are available using current limiting resistors. This allows both 5-volt and 3.3-volt support by changing the MCU. Contact Nohau Technical Support for further information.

Installing PLCC Socket Adapters

Do the following steps:

1. Make sure the power to the target system and the emulator host PC or HSP is turned off.
2. Connect the pod ground clip to the target ground.
3. Make sure the adapter is correctly oriented (pin 1 of the target processor to pin 1 of the pod adapter). For details about correct adapter orientation, see the following “Verifying Adapter Orientation” section.
4. Plug the adapter into the target socket.

Installing PGA Socket Adapters

Most Nohau pods for 68HC11 chips in the PLCC package require a PGA to PLCC adapter to connect to the target. Since most targets use PLCC sockets that are soldered into a PGA footprint, you can install a PGA socket that allows PGA pods to mate directly to the target. A PLCC socket plugs directly into the PGA socket for running the target with the MCU instead of the emulator.

Installing Surface Mount Adapters

First make sure you have a qualified technician solder the surface mount portion of the adapter to your target. Make sure you check the soldering for opens and shorts under a microscope, as these conditions are hard to detect otherwise.

Then do the following steps:

1. Make sure the power to the target system and the emulator host PC or HSP is turned off.
2. Connect the pod ground clip to the target ground.
3. Verify that pin 1 of the target processor is connected to pin 1 of the pod adapter. For details about correct adapter orientation, see the following "Verifying Adapter Orientation" section.
4. Plug the adapter into the surface mount portion of the target side adapter.

Verifying Adapter Orientation

WARNING

Do not apply power to your system unless you are absolutely sure the target adapter is correctly oriented. Failure to do so can cause damage to your target and/or emulator.

Usually pin 1 is clearly marked on the pod, target adapter, and target. If not, you need to manually verify that the pod adapter and target are correctly oriented. To verify proper connection, you will need an Ohm meter to check the status of one or two signals.

To verify proper connection, do the following steps:

1. Make sure all power is off.
2. Connect the adapter to the pod. Pin 1 can be marked in various ways. For example, you might see a beveled corner, the number one, or a V-shaped notch (Figure 47).

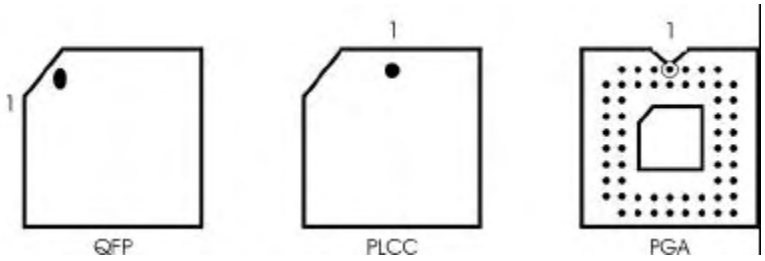

Figure 47. Pin Variations

3. Connect the adapter to the target.

4. Choose an easily accessible signal such as the XTAL signal. Then connect one side of the Ohm meter to the target signal and one side to the equivalent pin on the pod target header.

Tips for Avoiding Common Adapter Problems

- Select the correct socket. Select a quality socket that can withstand repeated use. Some inexpensive sockets tend to wear out quickly.

- Check the condition of the adapter and socket.
  - Make sure the adapter and socket are free of oxidation and corrosion.
  - Make sure the connector pins are not bent out of place. They should appear parallel to one another (Figure 48).

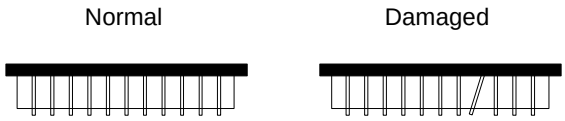

Figure 48. Connector Pins

  - If you are using a PLCC socket, make sure the pins have not flattened out. This tends to occur after repeated adapter insertions (Figure 49).

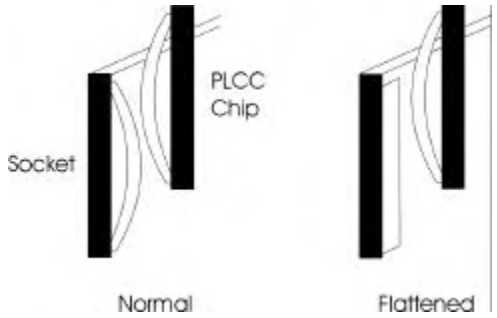

Figure 49. PLCC Socket Pins

> **Note**
>
> Usually surface mount PLCC sockets tend to wear out more quickly. Therefore, you need to inspect these sockets more frequently.

- Use the minimal amount of space required to connect to the target system. For example, avoid stacking sockets to meet height requirements. When necessary, use one piece extender adapters or short, flexible ribbon cables. (Refer to the *EMUL68–PC Price List* for further information about these items.)

Setting Up Proper Pod Support

In most cases, the pod will require additional support on the ribbon cable side to prevent the adapter from being pulled out of the socket.

Configuring Jumpers for Target Operation

Clock Jumpers

Each pod contains two clock jumpers or switches. These should be set to the external or target side position. Setting the jumpers or switches to this position ensures the following:

- The emulator will run at the target clock speed.
- The baud rate generator will have the same crystal reference.
- The periodic time-based interrupt will occur at the correct intervals.

Power Jumper

Set this jumper to the external position if you are connecting to a 3-volt target.

Control Signal Jumpers

Most pods have gated Read/Write and E-Clock control signals. These gates prevent the signals from going out to the target unless an access to target memory occurs. In some cases, target logic might require one or more of these signals to be present at all times. When this occurs, move the appropriate jumpers to the non-gated position.

CAUTION

When a control signal is placed in a non-gated condition, that signal will always be sent to the target and might cause bus collisions or unintentional writes to target data memory. To avoid these problems, make sure you review your control logic carefully.

Configuring Seehau for Target Operation

Emulator Configuration Tabs

After you connect the emulator to your target, review the following **Emulator Configuration** tabs and enter changes as appropriate. From the Seehau main menu, click the **Config** menu. Then click **Emulator**. The **Emulator Configuration** dialog box opens.

Hdw Config

- Enter your target clock frequency in the **uP Clock** field.

- Select the **Connect Reset to Target** option. The default is **target reset disconnected from emulation MCU**. (optional)

Misc Config

- Select the **Program Counter** option if you want to override the reset vector. Enter the value that you want to preload into the program counter. (optional)

- Select the **Stack Pointer** option if you want to preload the stack pointer with a designated value. (optional)

Mem Map Config

By default, all memory is mapped to the emulator. Click **Add** and enter memory ranges to access target resources such as ROM, RAM, or memory mapped I/O.

Banking

If your target uses bank switching, make sure you configure this tab prior to emulation. (Refer to Chapter 4, “Installing and Configuring the Seehau Software” in this guide for details about the **Banking** tab.)

Executing Code from Target ROM

If you are executing from target ROM, you can perform source level debug by loading symbols only. Do the following:

1. From the Seehau main menu bar, click the **Config** menu.

2. Click **Environment**.

3. From the **Preferences** tab, clear the **Code** option. Then click **OK**.

4. From the Seehau main menu bar, click the **File** menu. Then click **Load Code**. Seehau will now load only the debug symbol information from the selected file.

6 Introduction to Tracing

Overview of Tracing

Trace is a comprehensive tool used to analyze the microprocessor environment. While using the trace feature, you can passively analyze user code in real time as the code continues to execute. Without the trace tool, the emulator only provides a snapshot of the processor environment.

The trace feature provides the same basic capabilities as a logic analyzer. Each trace frame is time stamped and records the following fields: address and data values, cycle type, and up to 18 external inputs. Based on various combinations of these fields, you can set up trigger and/or filter conditions to control the trace recording.

By default, trace automatically starts recording when you begin user code execution. When you stop code execution, the trace history is displayed automatically.

Tracing allows you to perform many tasks, including:

- Detecting an error condition
- Analyzing a history of the sequence of events leading to an error
- Characterizing code behavior using tools such as Program Performance Analysis (PPA) and Code Coverage (available only with Enhanced Trace)
- Sampling time measurements
- Analyzing peripheral I/O

The Seehau software provides a symbolic interface for both the trace setup and display.

Basic Features of Tracing

- **Trigger**: An event that stops trace buffer recording.
- **Delay**: The number of trace frames collected after a trigger event occurs.
- **Filter**: A set of conditions that determine which frames are allowed into the trace buffer.
- **Time Stamp**: A feature that displays the number of machine cycles that have elapsed since the beginning of program execution.

Trace Window

The contents of the trace buffer are displayed in the Trace window (Figure 51). To open a Trace window, use the TR button on the toolbar, or select the **New** menu and click **Trace**.

The **Trace** menu controls most of the Trace window features.

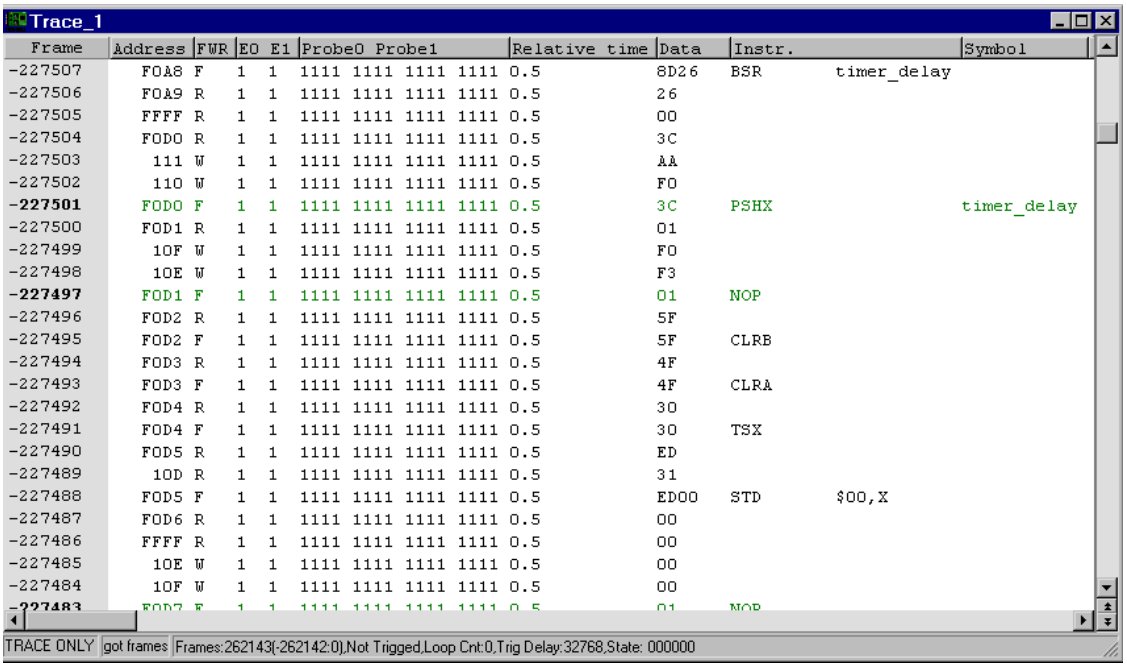

Figure 50. Trace Window (non-compressed)

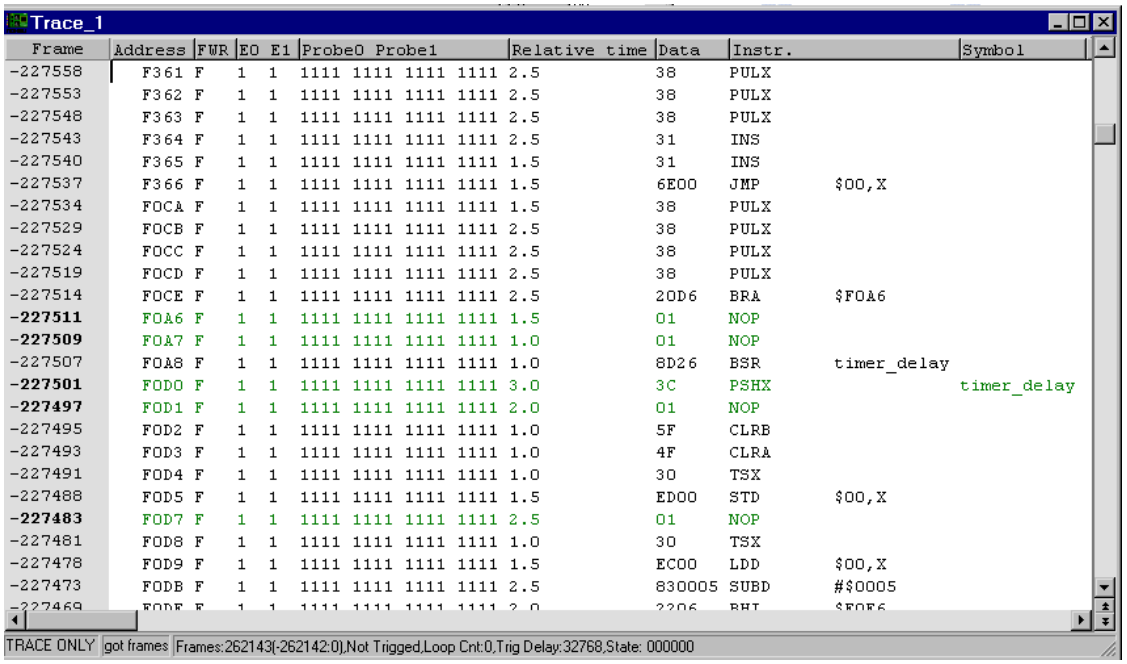

Figure 51. Trace Window (compressed)

Trace Window Columns

- **Frame Number** is on the far left of the window. Frame 0 always represents the trigger frame. If there is no trigger, frame 0 is the last frame in the buffer.
- **Address** displays the address of the bus cycle in hexadecimal notation.
- **FWR** shows the cycle type: first byte of Fetch, Write, Read, or other frames (shown with a dash).
- **E0 E1** displays status of the E1 and E0 inputs from the pod board.
- **Probe0 Probe1** displays the user-selected ports in binary format.
- **Time Stamp** displays one of the following options based on what you choose in the **Trace Display** options local menu:
  - **Relative Time**: amount of elapsed time for the current instruction.
  - **Absolute Time**: amount of elapsed time since the beginning of execution.
  - **Relative Cycle**: number of CPU cycles for the current instruction.
  - **Absolute Cycle**: total number of CPU cycles since the beginning of execution.
- **Data** displays bytes of data from the displayed bus cycle.
- **Instr.** shows the instruction disassembly.
- **Symbol** shows any symbolic label that refers to a specific address.

Local Trace Menu

To access the **Local Trace** menu, right click the Trace window or select the **Trace** menu which appears in the main menu bar only when the Trace window is active (Figure 52).

- **Go to Frame number**: Opens a dialog box where you can enter a specific frame number for display.
- **Find Trigger Point**: Displays the trigger point (frame zero).
- **Zero Time at Cursor**: Changes the time stamp at the selected frame to zero and makes all other time stamps relative to the selected frame.
- **Synchronize Source Window**: Automatically aligns the display of code in the Source window as you scroll through the Assembly code in the trace buffer. You must use the up/down arrow keys in conjunction with this feature.

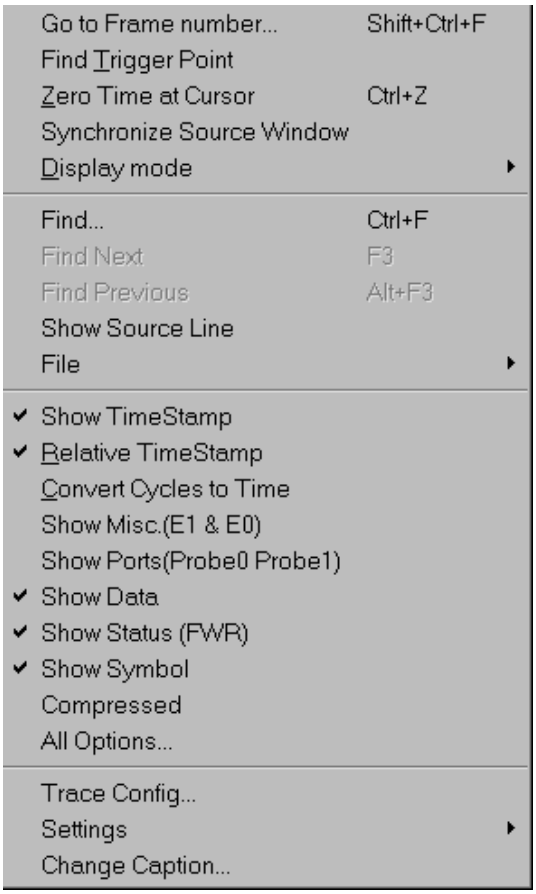

Figure 52. Local Trace Menu

- **Display mode**: Opens a submenu that allows you to select which code to display:
  - **Source**: Displays C source code only in the trace buffer.
  - **Assembly**: Displays Assembly code only in the trace buffer.
  - **Both**: Displays both C and Assembly code in the trace buffer.

- **Find**: Opens the **Find Address** dialog box where you can search the trace buffer for an address or a range of addresses and a cycle type.

- **Find Next**: Click to find the next frame match.

- **Find Previous**: Click to find the previous frame match.

- **Show Source Line**: Displays the associated source code for the current frame. The frame address must match a source line address.

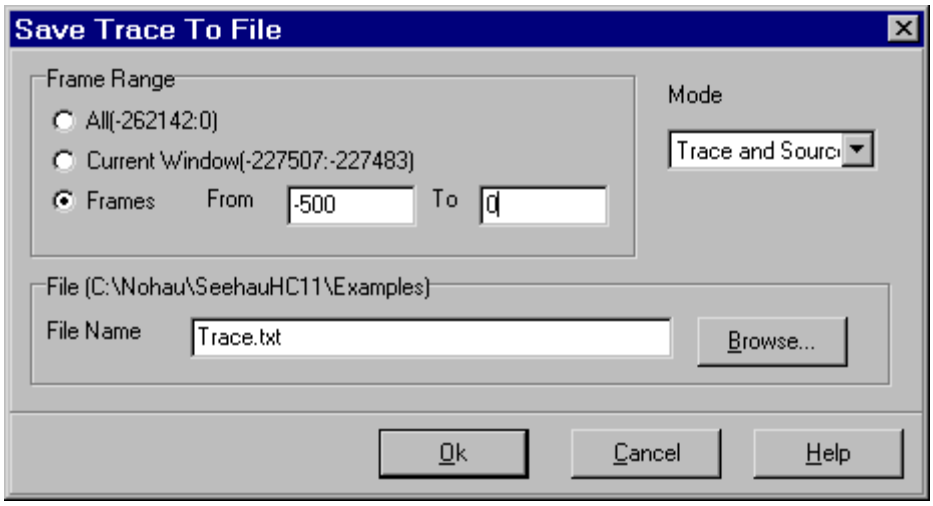

Figure 53. Save Trace to File Dialog Box

- **File**:
  - **Save to File**: Opens the **Save Trace To File** dialog box where you can save the contents of the trace buffer as text to a file (Figure 53).
    - **Frame Range**: Select one of the following:
      All: Saves the entire trace buffer
      Current Window: Saves only the contents of the current Trace window
      Frames: Allows you to specify a range of frames
  - **Print**: Allows you to send the trace buffer to a printer.

> **Note**
>
> The following menu items can be toggled individually or can be configured all at once. Click **All Options** to open the **Display Options** dialog box.

- **Show TimeStamp**: Displays the time stamp which represents the number of machine cycles that have elapsed since the beginning of program execution.
- **Relative TimeStamp**: Displays the time stamp as the number of machine cycles that have elapsed since the execution of the previous instruction.
- **Convert Cycles to Time**: Converts the time stamp from machine cycles to actual time based on the microprocessor clock (uP clock).
- **Show Misc.(E1 & E0)**: Displays status of the E1 and E0 inputs from the pod board.
- **Show Ports (Probe0 Probe1)**: Displays the user-selected ports in binary format.
- **Show Data**: Displays the data field.

- **Show Status (FWR)**: Displays bus cycle type (Fetch, Write, or Read).
- **Show Symbol**: Displays symbolic labels associated with the address field.
- **Compressed**: Displays Assembly code only. (**Note**: RD/WR cycles are not shown.)
- **All Options**: Opens the Display Options window where you can select or clear trace options in a single update.
- **Trace Config**: Opens the **Trace Configuration** dialog box. Refer to Chapter 7, “Configuring Standard and Enhanced Trace Boards” in this guide.
- **Settings**: Opens a submenu that allows you to set up Trace window attributes.
- **Change Caption**: Allows you to change the Trace window caption in the title bar.

7 Configuring Trace Boards

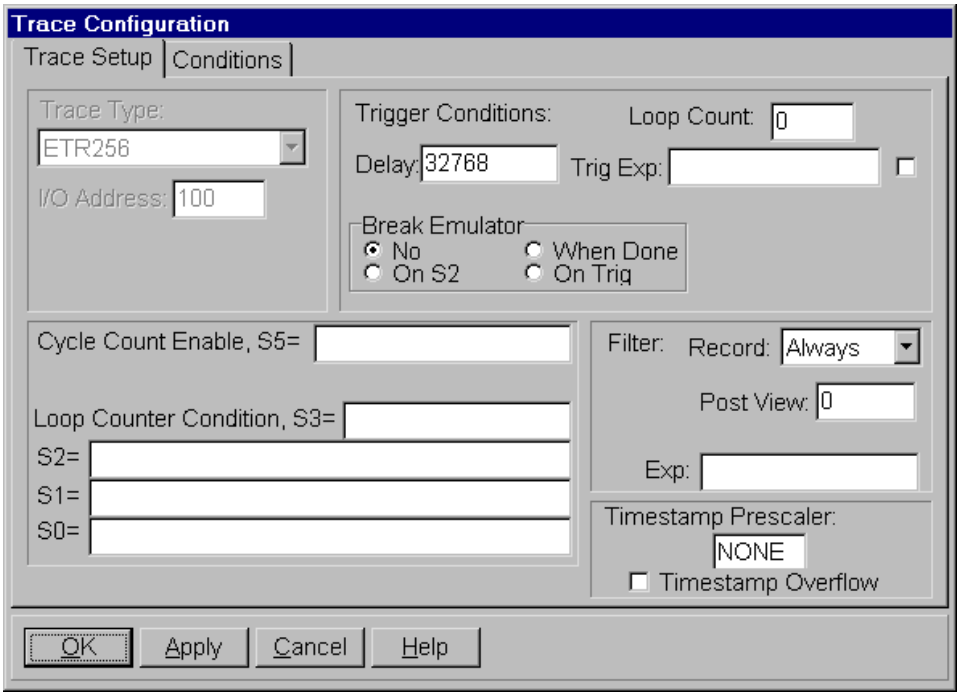

Figure 54. Standard Trace Configuration Dialog Box (Trace Setup)

To program the trace board for specific tasks, use the **Trace Configuration** dialog box (Figure 54). To open the dialog box, do the one of the following:

- From the **Trace** menu, click **Trace Config**
- From the **Config** menu, click **Trace**

Configuring the Standard Trace Board

After the **Trace Configuration** dialog box opens, click **Trace Setup**. Enter or select appropriate data for the following **Trace Setup** fields:

Trace Setup Tab

- **Trace Type**: Displayed for reference only. Exit Seehau and run Seehau Reconfig to change this setting.
- **I/O Address**: Displayed for reference only. Exit Seehau and run Seehau Reconfig to change this setting.

- **Trigger Conditions**
  - **Loop Count**: The number of times the trigger condition must be met before the trigger action is taken.
  - **Delay**: The number of frames recorded after the trigger condition is met.
  - **Condition**:
    - **No Trig**: Disables triggering. Trace collects data continuously in a FIFO buffer.
    - **Trig On A**: Triggers when the A condition is met.
    - **Trig On B**: Triggers when the B condition is met.
    - **A then B**: A sequential trigger where A condition must be met first, followed by the B condition.
    - **A Loop**: Triggers after A condition is met ***n*** times, where ***n*** is the loop count field.
    - **B Loop**: Triggers after B condition is met ***n*** times, where ***n*** is the loop count field.
    - **A Loop then B**: A sequential trigger, where A condition must be met ***n*** times, followed by the B condition.
    - **(A then B) loop**: A sequential trigger, where A condition must be met first, followed by the B condition ***n*** times.

- **Break Emulator**
  - **No**: Emulation does not stop as a result of trace trigger conditions.
  - **On Trig**: Emulation stops when the trigger condition is met.
  - **On B**: Emulation stops when the B condition is met.

- **Filter**: The **Filter** field controls which frames are recorded by the trace board.
  - **Record**: Click the down arrow and select one of the following options:
    - **All**: All frames are captured. There is no filtering.
    - **A**: Only frames which match the A condition are recorded.
    - **B**: Only frames which match the B condition are recorded.
    - **A & B**: Frames which meet both A and the B conditions are captured.
    - **A On, B Off**: When the A condition is met, the trace is turned on and all frames are recorded. When the B condition is met, the trace is turned off.

  - **Post View**: The number of frames collected after each match of the filter condition. This feature is most useful when filtering data/port reads and writes. Allows you to determine where in your code the read or write occurred.

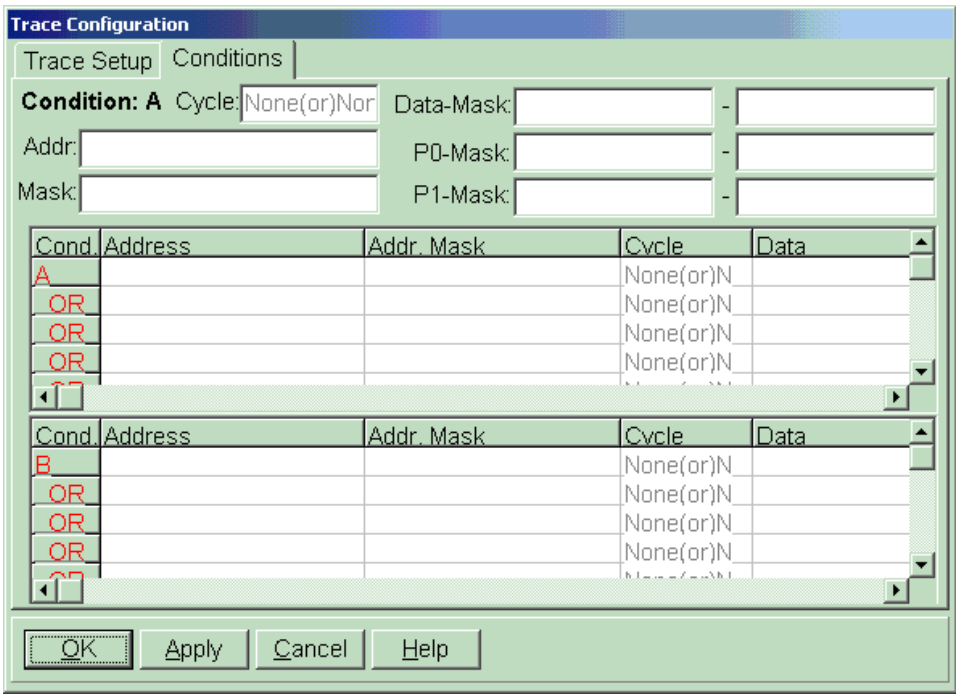

Figure 55. Standard Trace Configuration Dialog Box (Conditions Tab)

Conditions Tab

Right-click in the Conditions window to access the **Add**, **Edit**, or **Remove** menu. Select **Edit** from this menu, and the **Edit Data Conditions** dialog box opens (Figure 56).

Edit Data Conditions Dialog Box

> **Note**
>
> If left blank, the following fields are evaluated as Don't Cares.

- **Enabled**: Select to enable condition.
- **Address**: Enter an address or range of addresses. (See Appendix A for correct syntax.)
- **Address Mask**: Enter an address mask. The mask is used as an alternate method to create address ranges. The address mask value is logically ANDed with the address value. A zero in the result corresponds to a Don't Care in the address value. For example:
  Address=ABCDH
  Mask=FF0FH
  Resulting range=ABxDH
- **PROBE0**: Enter an 8-bit value or range corresponding to the port that is compared to the TTL level data being traced by PROBE0. On most pods PROBE0 is connected to Port D by jumpers. By removing the jumpers and using wires, any eight external signals can be traced as PROBE0. Purchase optional E-Z hooks from Nohau (Part #EMUL-PC/E-Z).

- **PROBE0 Mask**: Enter a PROBE0 mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the PROBE0 value. A zero in the result corresponds to a Don't Care in the value. For example:
  PROBE0=A5H
  Mask=F0H
  Resulting range=AxH

- **Data**: Enter an 8-bit (data bus) value or range.

- **Data Mask**: Enter a data mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the data value. A zero in the result corresponds to a Don't Care in the value. For example:
  Data=A5H
  Mask=F0H
  Resulting range=AxH

- **PROBE1**: Enter an 8-bit value or range corresponding to the port that is compared to the TTL level data being traced by PROBE1. On most pods PROBE1 is connected to Port A by jumpers. By using wires, any eight external signals can be traced as PROBE1.

- **Probe1 Mask**: Enter a PROBE1 mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the PROBE1 value. A zero in the result corresponds to a Don't Care in the value. For example:
  PROBE1=A5H
  Mask=F0H
  Resulting range=AxH

- **Miscellaneous**

  - **Type**: Select one of the following bus cycle types:
    - **None**: Don't Care
    - **Read**: A data read
    - **Write**: A data write
    - **R/W**: A data read or write
    - **Fetch**: A code fetch.

  - **E0**: The E0 external input pin on the pod. Select one of the following levels:
    - **High**
    - **Low**
    - **None**: Don't Care

  - **E1**: The E1 external input pin on the pod. Select one of the following levels:
    - **High**
    - **Low**
    - **None**: Don't Care

Configuring a Trigger Condition

The following example will trigger when the value of 0FH is written to the symbol **timer.min**. The trace will then force a break when the trigger occurs.

1. Open the Trace Configuration window by doing one of the following: from the **Config** menu, click **Trace**. Or from the **Trace** menu, click **Trace Config**.

2. In the **Trig Exp** text box, enter A for the condition. Then select the check box to enable the trigger expression.

3. In the Break Emulator area, select **On Trig**. (**On Trig** breaks immediately. Alternatively, you can choose **When Done** which breaks after the number of delay cycles are collected.)

4. From the Condition A line, right click and select **Edit**. The **EditDataCondHC11** dialog box opens (Figure 56).

5. Complete the following fields:

   - **Enabled check box**: Automatically selected when you enter an address in the **Address** text box.
   - **Address**: Enter &timer.min. (The **&** denotes the address of the symbol rather than the value of the symbol.)
   - **Mask**: Enter FFFF. (A one in any bit position denotes that the value specified in the address field must match the address field in the trace frame. Whereas, a zero denotes a Don't Care bit.)
   - **Data**: Enter FH.
   - **Mask**: Enter FF. (A one in any bit position denotes that the value specified in the address field must match the address field in the trace frame. Whereas, a zero denotes a Don't Care bit.)
   - **Miscellaneous Type**: Select **Write** from the pull-down menu.

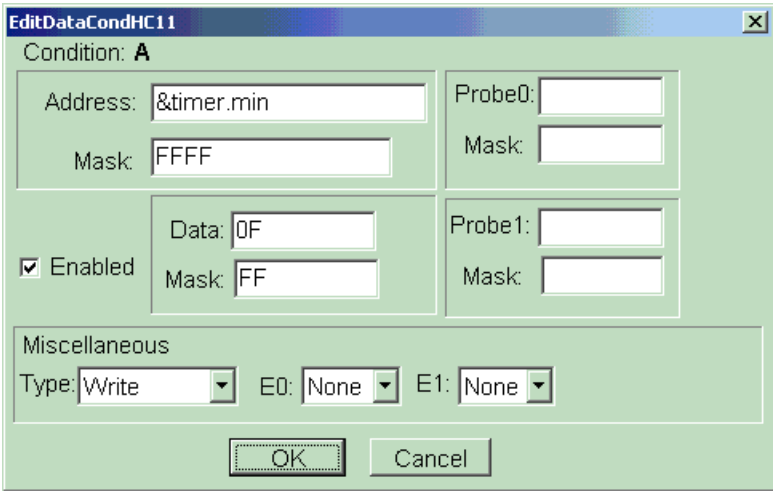

Figure 56. EditDataCondHC11 Dialog Box

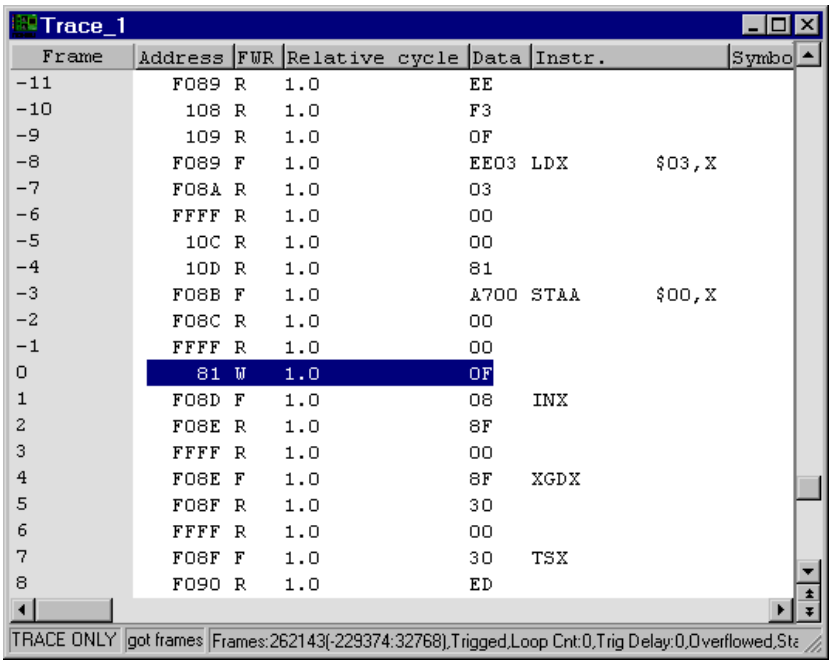

Figure 57. Trace Display Window

6. Click **OK**.

7. Click **OK** in the Trace Configuration window (Figure 55).

8. Click the Reset and Go  button on the Speedbar. The Trace Display window will now update.

(For addition trace examples, refer to the “Trace Configuration Examples” section.)

Configuring the Enhanced Trace Board

After the **Trace Configuration** dialog box opens, click **Trace Setup**. Enter or select appropriate data for the following fields:

Trace Setup Tab

- **Trace Type**: Displayed for reference only. Exit Seehau and run Seehau Reconfig to change this setting.

- **I/O Address**: Displayed for reference only. Exit Seehau and run Seehau Reconfig to change this setting.

- **Trigger Conditions**

  - **Loop Count**: The number of times the trigger condition must be met before the trigger action is taken.
  - **Delay**: The number of frames recorded after the trigger condition is met.

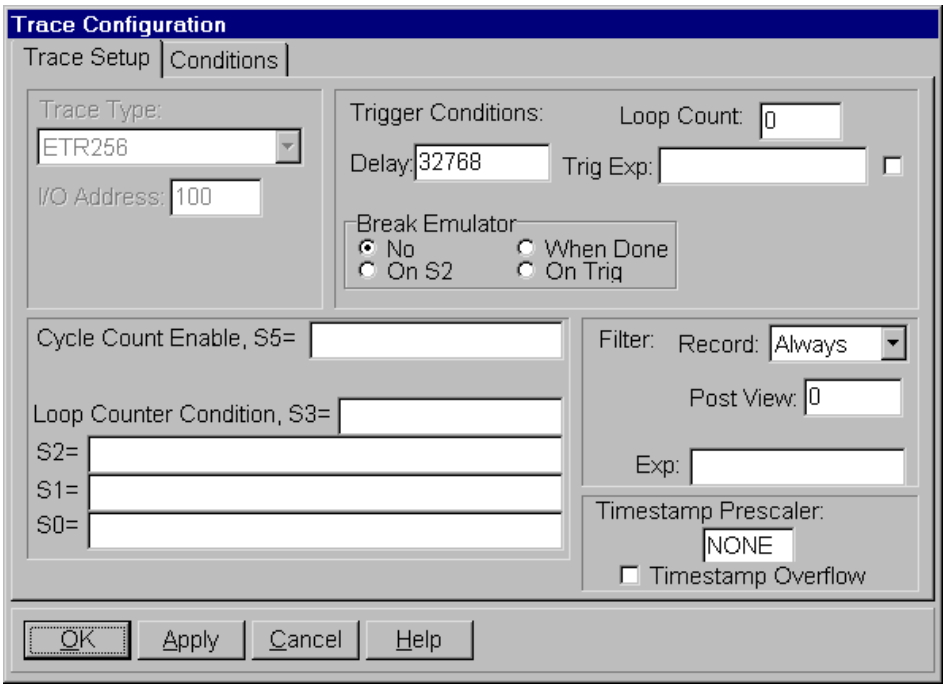

Figure 58. Enhanced Trace Configuration Dialog Box (Trace Setup)

- **Trig Exp**: Select this option to enable the trigger. Clear this option to disable the trigger.

  In the **Trig Exp** text box, enter a Boolean expression that uses the qualifying registers A – H, S0 – S5 and the output of the loop counter CO. The Boolean express can also contain the THEN operator.

  If the THEN operator is used the state bits S0, S1 and S2 can not be used. These bits are being used to generate the THEN condition.

  Examples

  A
  NOT A
  A OR B
  A AND B
  A THEN B
  A AND NOT B
  A THEN B THEN C THEN D THEN E THEN F THEN G THEN H
  A OR B THEN C AND NOT D OR E
  A OR S3
  A AND CO
  A THEN B OR S3
  A THEN B THEN CO AND C

- Break Emulator:
  - **No**: Emulation does not stop as a result of trace conditions.
  - **On S2**: Emulation stops when S2 is *true*.
  - **When Done**: Emulation stops after the trigger condition is met and the delay counter reaches zero.
  - **On Trig**: Emulation stops when the trigger condition is met.

- **Cycle Count Enable, S5=**: Consists of a Boolean expression of events, S0 to S5 and C0. When the expression is *true*, a 32-bit counter is enabled to count cycles. This is used to measure cycles or time without stopping emulation. These numbers are displayed on the toolbar of the Seehau user interface. On pods that have the pin labeled FLF, the S5 bit is logically output here. This signal can also be found on pin 22 of the 50-pin ribbon cable connector.

- **Loop Counter Condition, S3=**: Consists of a Boolean expression of events, S0 to S5 and/or qualifiers A – H that are used to describe how the loop counter will operate. When the condition goes from *false* to *true*, the loop counter will shut down.

  S0, S1, S2: Used only if the THEN operator is not being used in the **Trig** or **Record** fields. Otherwise, these fields can be used freely for Boolean equations.

- **Filter**: The **Filter** field controls which frames are recorded by the trace board.
  - **Record**: Click the down arrow and select one of the following options:
    - **Always**: If **Always** is selected, there is no filtering of the trace capture; therefore, all frames will be recorded.
    - **IF**: If a trace frame matches **Exp**, then the frame is recorded.
  - **Post View**: The number of frames collected after each match of the filter condition. This feature is most useful when filtering data/port reads and writes. Allows you to determine where in your code the read or write occurred.
  - **Exp**: (See Trig Exp)
  - **Timestamp Prescaler**: Consists of one 32-bit hardware counter with a 16-bit prescaler. Only the 32-bit counter is recorded in the trace buffer. If the prescaler is none, the result is full resolution in which each frame represents a full bus cycle. Therefore, 4,294,967,294 cycles can be counted before the counter rolls over to 0. With the prescaler set to 2, the time is doubled. By programming the prescaler to a high enough number, the time of overflow time can be in minutes, hours, days, etc.
  - **Timestamp Overflow**: By checking the box to the left of **Timestamp Overflow**, the roll-over of the counter can be forced in the trace buffer.

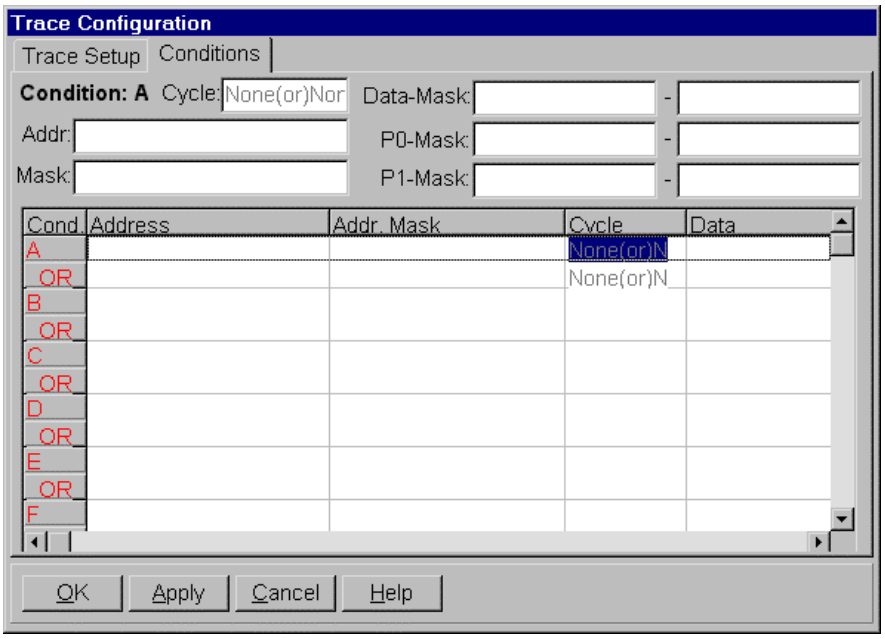

Figure 59. Enhanced Trace Configuration Dialog Box (Trace Setup)

Conditions Tab

Right click in the Conditions window to access the **Edit** or **Remove** menu. Select **Edit** from this menu, and the **Edit Data Conditions** dialog box opens.

> **Note**
>
> If left blank, the following fields are evaluated as Don't Cares.

- **Enabled**: Select to enable condition.
- **Address**: Enter an address or range of addresses. (See Appendix A for correct syntax.)
- **Address Mask**: Enter an address mask. The mask is used as an alternate method to create address ranges. The address mask value is logically ANDed with the address value. A zero in the result corresponds to a Don't Care in the address value. For example:
  Address=ABCDH
  Mask=FF0FH
  Resulting range=AbxDH
- **PROBE0 Mask**: Enter a PROBE0 mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the PROBE0 value. A zero in the result corresponds to a Don't Care in the value. For example:
  Probe0=A5H
  Mask=F0H
  Resulting range=AxH

- **Data**: Enter an 8-bit (data bus) value or range.

- **Data Mask**: Enter a data mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the data value. A zero in the result corresponds to a Don't Care in the value. For example:
  Data=A5H
  Mask=F0H
  Resulting range=AxH

- **PROBE1**: Enter an 8-bit value or range corresponding to the port that is compared to the TTL level data being traced by PROBE1. On most pods PROBE1 is connected to Port A by jumpers. By using wires, any eight external signals can be traced as PROBE1.

- **PROBE1 Mask**: Enter a PROBE1 mask. The mask is used as an alternate method to create ranges. The mask value is logically ANDed with the PROBE1 value. A zero in the result corresponds to a Don't Care in the value. For example:
  Probe1=A5H
  Mask=F0H
  Resulting range=AxH

- **Miscellaneous**

  - **Type**: Select one of the following bus cycle types:
    - **None**: Don't Care.
    - **Read**: A data read.
    - **Write**: A data write.
    - **R/W**: A data read or write.
    - **Fetch**: A code fetch.

  - **E0**: The E0 external input pin on the pod. Select one of the following levels:
    - **High**
    - **Low**
    - **None**: Don't Care.

  - **E1**: The E1 external input pin on the pod. Select one of the following levels:
    - **High**
    - **Low**
    - **None**: Don't Care.

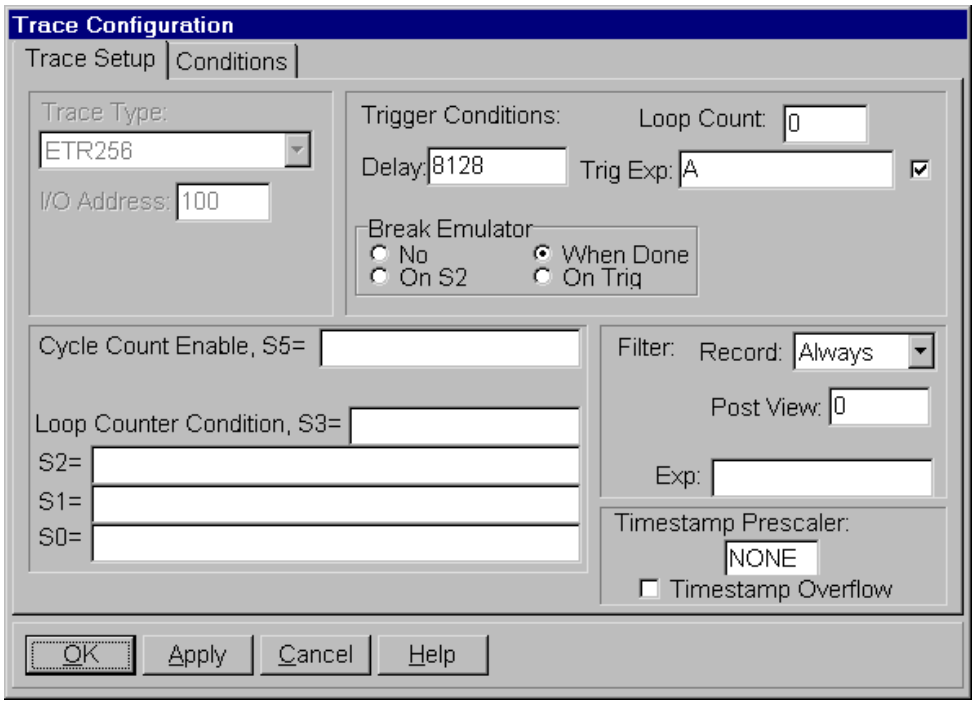

Figure 60. Trace Trigger Setup

Trace Configuration Examples

Trigger Example

This section describes how to set up the emulator trace to trigger on a specific condition with the following parameters:

- Set the trace to trigger on the function Main.
- Collect 8K bus cycles after the trigger.
- Allow the emulator to continue running after the trigger.

From the **File** menu, select **Load Code**. Double-click the file Timer.d07. Then do the following steps:

1. From the **Config** menu, open the **Trace Configuration** dialog box.
2. Set the **Delay** field to 8,192 frames. This setting will allow the trace to collect 8K bus cycles after the trigger condition is met.
3. Set the **Trigger Condition** field to A.
4. In the **Break Emulator** area of the dialog box, click **No** to allow the emulator to continue running after the trigger.

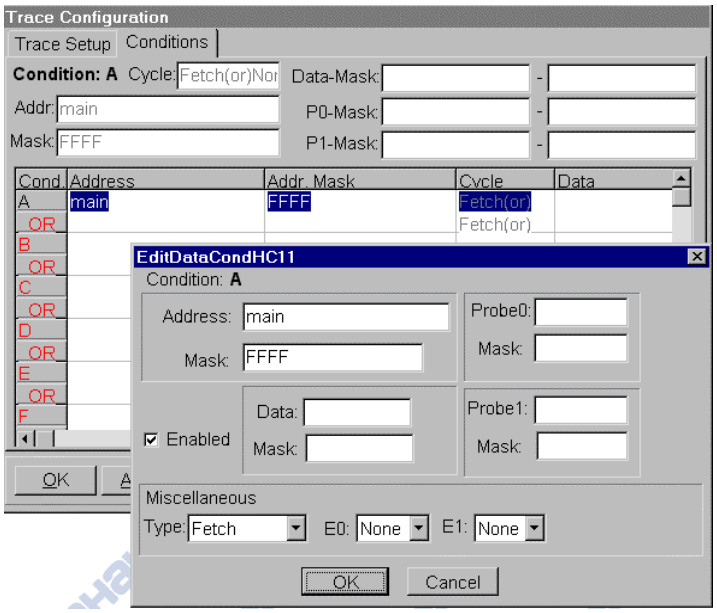

Figure 61. Trace Conditions Setup

5. Click the **Conditions** tab. This will allow you to configure condition A for the entry to function Main.

6. Right-click on the A condition line, and select **Edit**. This opens the **Condition A** dialog box.

7. In the **Address** field, type `Main`

8. From the **Type** pull down list, select **Fetch**.

9. Click **OK** to complete the trace setup.

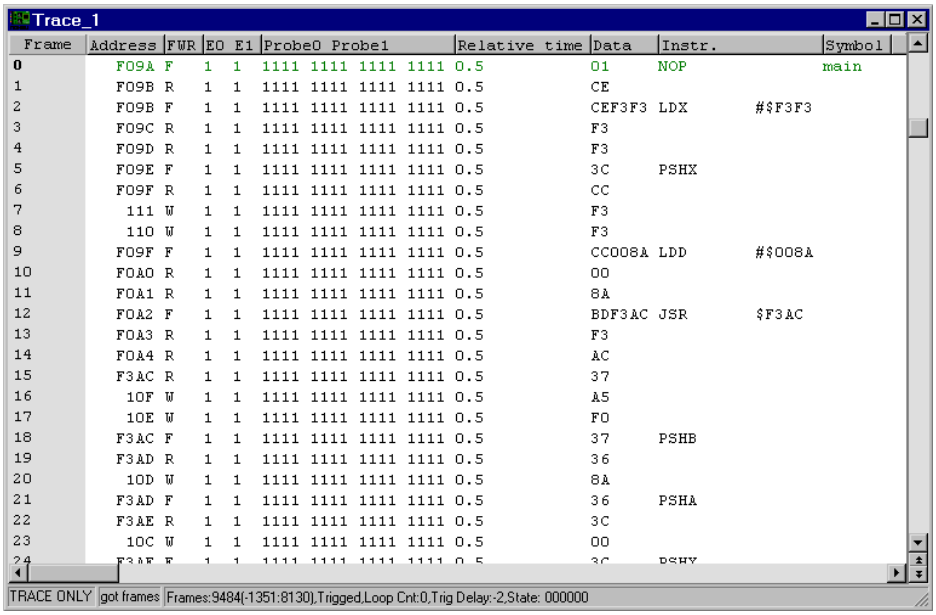

Figure 62. Trace Window with Trigger Example

To capture the trace, on the Seehau Speedbar click the Reset button  . Then click the Go button 

.

As soon as the trigger condition is met, the Trace Display window automatically updates. (See Figure 62.) Frame 0 is the trigger point which is the entry to function Main.

Filter Example

This section describes how to set up the emulator trace to filter data writes with the following parameters:

- Capture only data writes to a structure field called Timer.sec.
- Stop the trace after collecting 5,000 frames.

From the **File** menu, select **Load Code**. Double-click the file Timer.d07.
Then do the following steps:

1. From the **Config** menu, open the **Trace Configuration** dialog box.
2. Set the **Delay** field to 5,000 frames. This setting will allow the trace to collect 5,000 bus cycles after the trigger condition is met.
3. Set the **Trigger Condition** field to A.
4. Set the **Record** field to A.

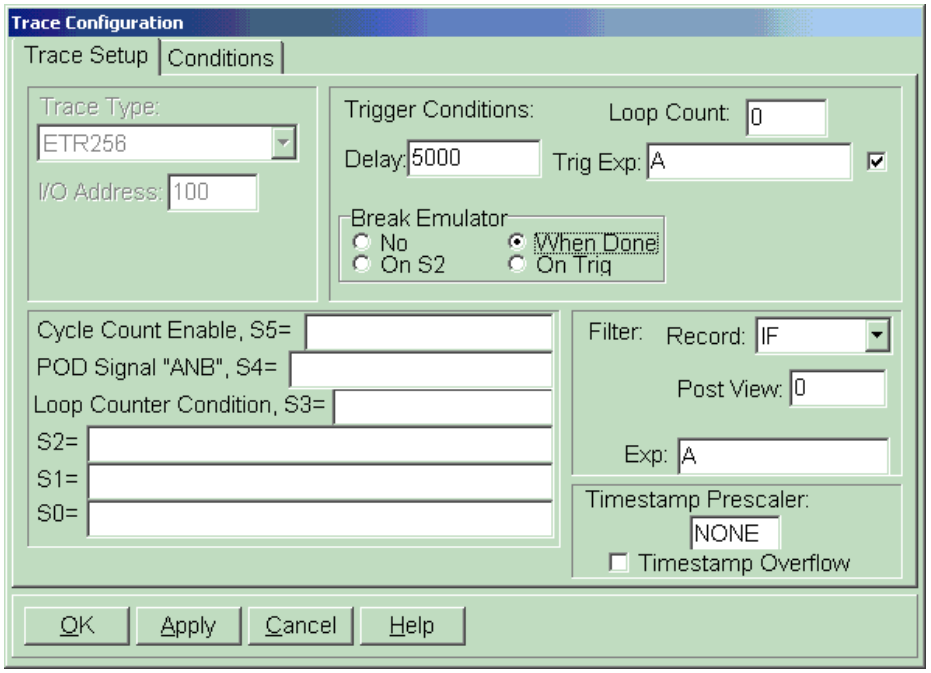

Figure 63. Trace Setup Tab

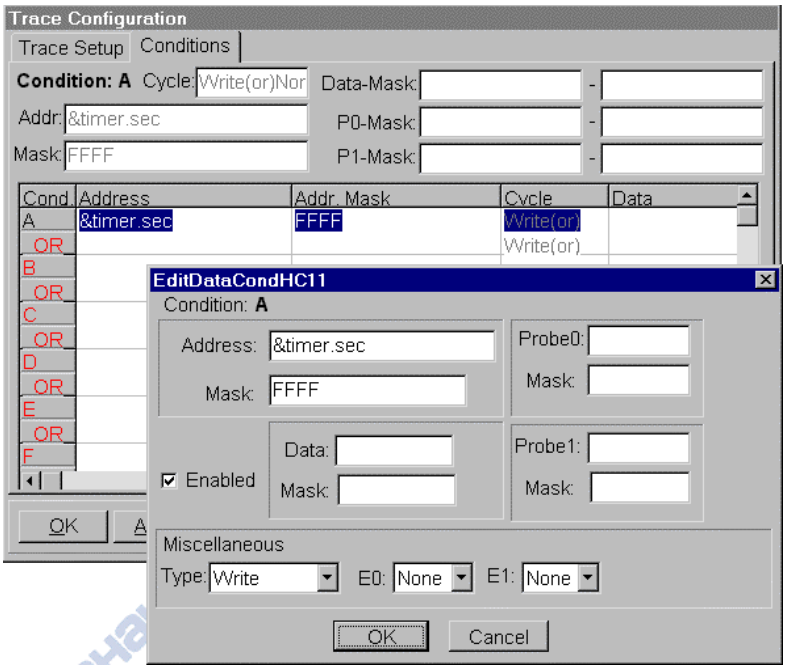

Figure 64. Trace Filter Setup

5. In the **Break Emulator** area of the dialog box, click **No** to allow the emulator to continue running after the trigger.

6. Click the **Conditions** tab. This will allow you to configure condition A for writes to the Timer.sec structure field.

7. Right-click on the A condition line, and select **Edit**. This opens the **Condition A** dialog box.

8. In the **Address** field, type `&timer.sec`.

9. From the **Type** pull down list, select **Write**. Then click **OK** to complete the trace setup.

To capture the trace, on the Seehau Speedbar click the Reset button  .Then click the Go button  .

As soon as the trigger condition is met, the Trace window automatically updates. (See Figure 65.) Cycle 0 is the trigger point which is the first write to Timer.sec.

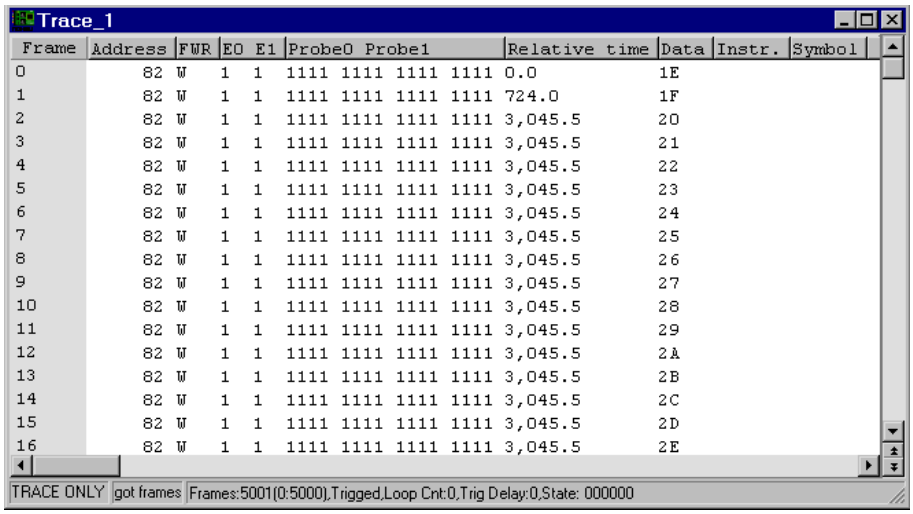

Figure 65. Trace Window with Filter Example

Combination of Conditions

When setting up conditions, it is possible to enter multiple qualifiers under a single condition. A qualifier is displayed on a single line under the Trace Condition Display. For example, under Condition A, a qualifier preceded by the word or indicates that condition has multiple qualifiers.

When a trace condition has multiple qualifiers, the fields in the qualifier are logically ORed together. For example, if you define Condition A as a Write to address 81H “OR” a Read from address 82H, the logically combined condition results in a Read “OR” Write to or from address 81H “OR” 82H.

Alternatively, setting condition B to a Write to address 81 and setting condition C to a Read from 82, then triggering or filtering on “B or C” results in the entire condition being ORed rather than the individual fields. The resulting condition is a Write to 81H OR a Read from 82H.

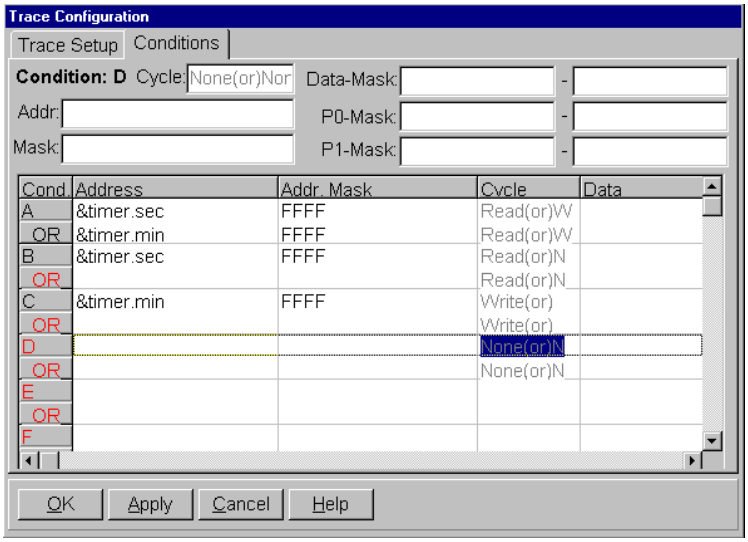

Figure 66. Combination of Conditions

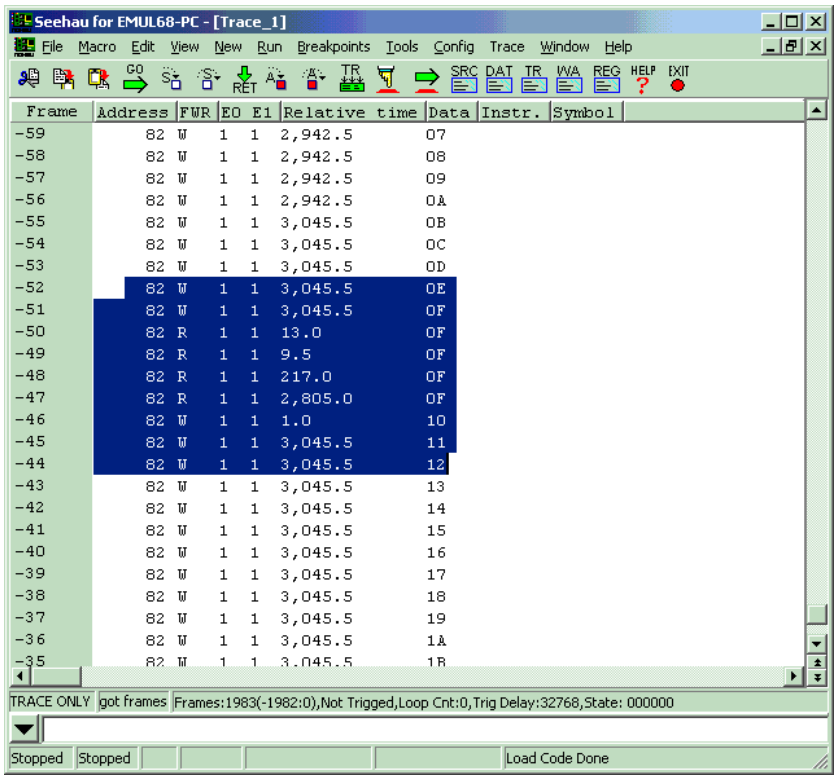

Figure 67. Trace Display Filter A

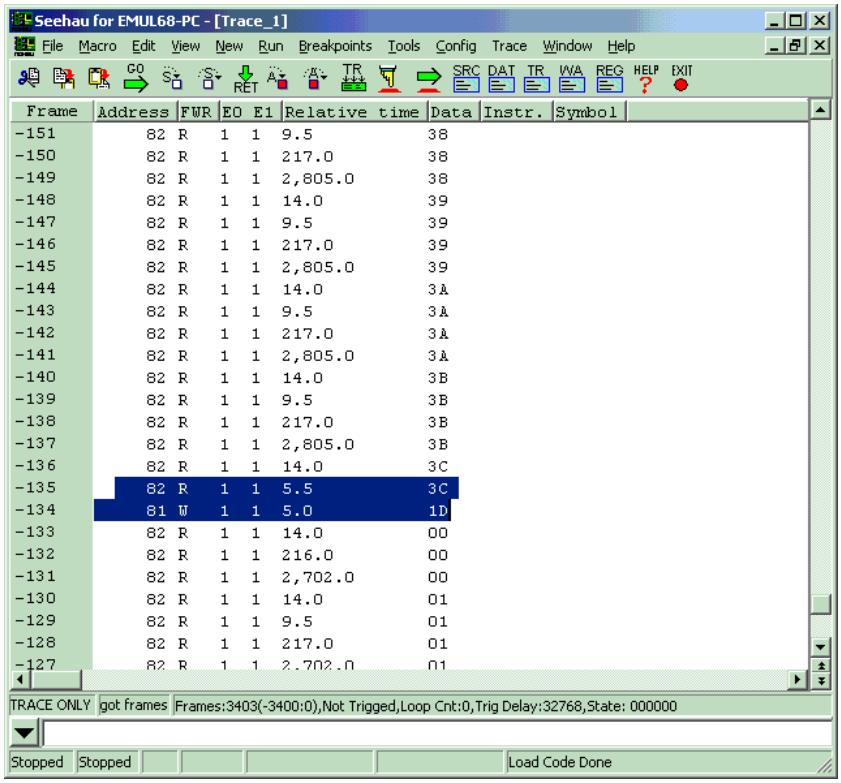

Figure 68. Trace Display Filter B or C

Building a State Machine Using S(x) Functions

The following provides an example using S2 – S0 as a 3-bit counter (eight states) for each time the A event equals the First Opcode fetch at address F0CA hex. In this case, the result of S0 – S2 triggered the trace when S2, S1and S0 are all *true*. This example can be expanded by using more events (A-H) in the equations.

Using the program Timer.d07 the address F0CA hex is the address of the timer_delay function. All trace fields should be at their default values except for the following:

- Trig = (S0 and S1 and S2)
- S2 = S2 AND /(S0 and S1 and A) or /S2 and (S0 and S1 and A)
- S1 = S1 and /(S0 and A) or /S1 and (S0 and A)
- S0 = (S0 and /A) or (/S0 and A)

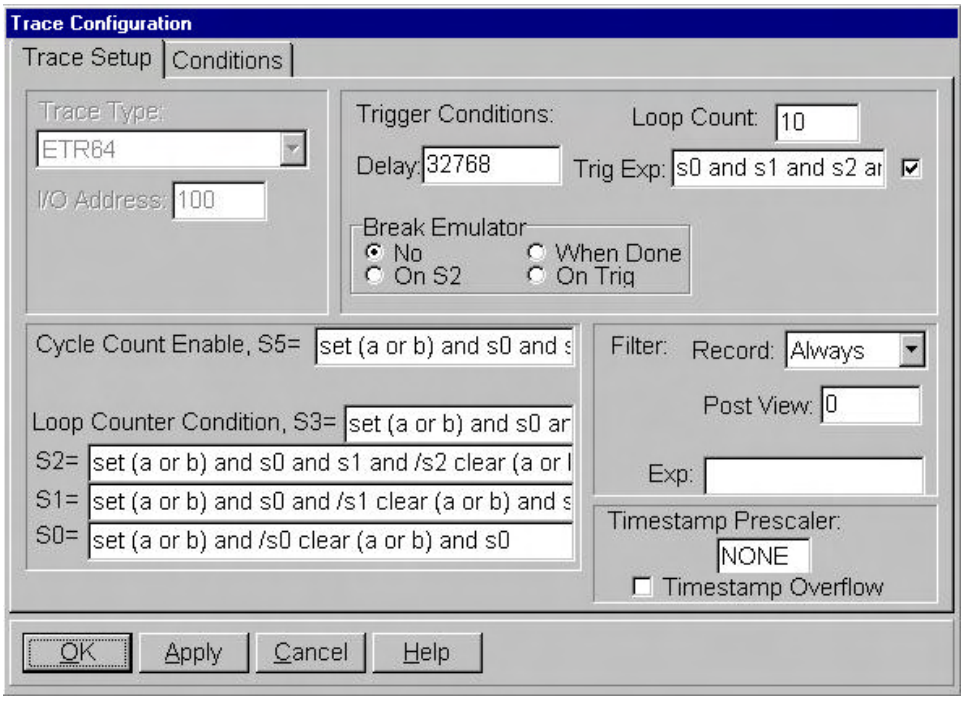

Figure 69. 6-Bit State Machine

The expression in S0, S1 and S2 could have been written using the Boolean XOR; however, the XOR is not implemented. The equivalent expressions are:

- S2 = S2 xor (S0 and S1 and A)
- S1 = S1 xor(S0 and A)
- S0 = S0 xor A

Example of How to Use the Set-Clear with the Sx Functions

S0 = set A clear B

S0 is set to *true* when A becomes *true*. It remains *true* until B becomes *true* (even if A is *false* before B is *true*). A and B in the previous example can be Boolean expressions. This can be used in a number of ways. The following example shows how to create a window for recording.

Record Expression = IF
Exp=S0 or A
S0 = set A clear B

The A must be in the RECORD expression in order for the result of the SET A to appear in S0 delayed by one frame. Otherwise, the address F0E2 hex will be missed.

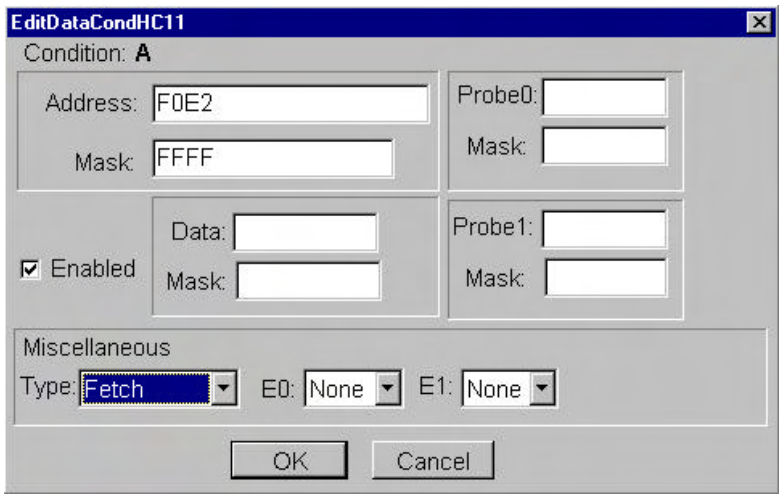

Figure 70. Event Field (A)

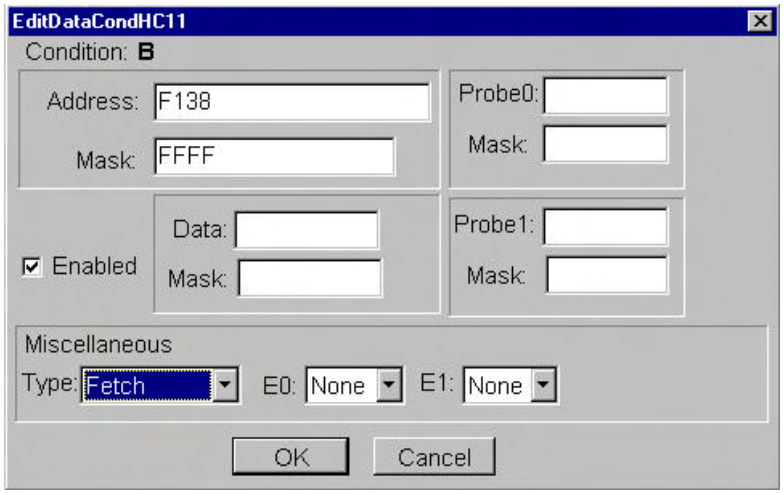

Figure 71. Event Field (B)

All activities taking place between the execution of the two specified addresses are recorded into the trace buffer. Any calls to subroutines that would be outside of the address range between F0E2 and F138 are included.

Figure 69 is an example of a 6-bit state machine using the Set-Clear and demonstrating how the loop counter and cycle counter work.

The program Timer.d07 located in the Examples directory will write to the symbolic address of Timer.sec (address C0h) at an approximate rate of one second. First click the Reset button on the Speedbar, then click the Go button to see the 6-bit state machine bits being updated in the Trace window status bar.

With this example the loop counter, state machine bit 3, is set for 10. The counter will decrement every time the loop counter condition changes from *false* to *true* (0 to 1). The status of the state machine bits, value of cycle count, and the triggered or not triggered status are all displayed on the status bar in the Trace window.

Note

The MSB of the state machine is on the left and the LSB in on the right of the binary display.

Tracing During Reset and Go

If your power-on reset routine writes to any of the registers that must be written in the first 64 cycles after reset, such as the INIT and OPTION registers, then you must always start your program with a Reset and Go command, and not two separate commands (Reset Chip and then Go). This does not apply in test mode, since the 64-cycle limit is disabled in test mode.

The Reset and Go command can also be activated by the **Reset and Go** selection on the **Run** menu. It causes the following actions to occur:

- Emulation mode is entered, trace is started and reset is driven low by the emulator on the same bus cycle.
- Reset is held active (low) for approximately 1/5 second and then released.
- The HC11 comes out of reset, fetches the address of the user's power-on reset routine, and branches to it.

Because of the way Reset and Go is implemented, bus cycles that occur while reset is active are recorded in the trace buffer and counted by the execution cycle counter. Therefore, the first time you stop execution after Reset and Go, the cycle counter will be meaningless.

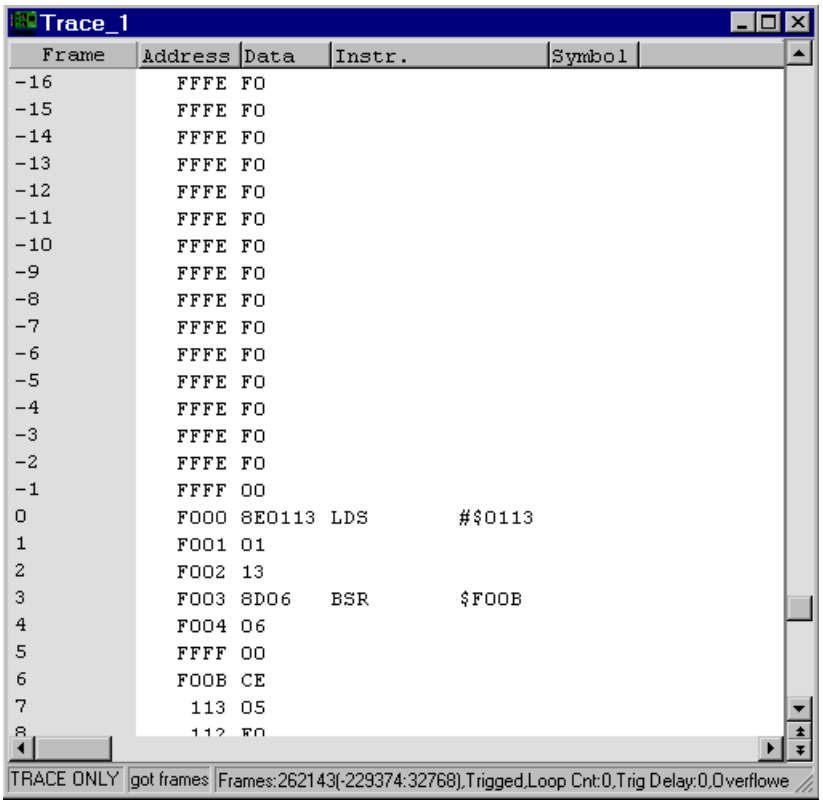

Figure 72. Trace Capture of Reset and Go

8 Troubleshooting

Before you start troubleshooting, first check the following items:

- Are the cables connected properly?
- Is the pod connected to the emulator board?
- If you are using an HSP box, is the HSP power turned on?
- Did you remove any foam that might be present on the bottom pins of the pod?
- Did you configure Seehau correctly for your MCU and pod?
- If the pod is not connected to your target, are the power and crystal jumpers/switches in the INT position?
- If the pod is connected to your target, is the target power turned on?

HSP Box

Step 1. When you start Seehau, does the HSP card LED flash?

- **Yes**. Go to **Step 2**.
- **No**. Make sure the power is on. Make sure the following are connected:
  - HSP box is connected to computer.
  - Power supply is connected to HSP.
  - Pod is connected to emulator board

If the HSP card LED is still not working, refer to the “Debugging the Parallel Port” section.

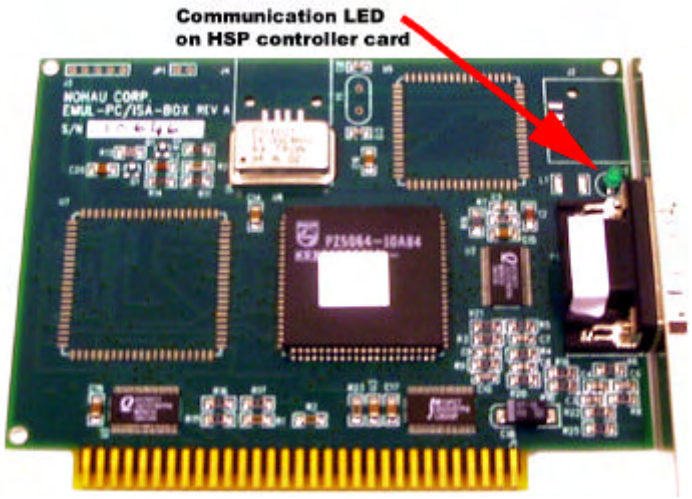

Figure 73. HSP Card LED

Step 2. If your pod has a reset LED, does it flash when you start Seehau?

- **Yes**. Go to **Step 4**.
- **No**. Go to **Step 3**.

Step 3. Do board I/O addresses match the values in the Seehau configuration?

If your reset LED does not flash or your pod is not equipped with a reset LED, verify that the board I/O addresses (for emulator and trace boards) match the values in the Seehau Configuration:

- **Yes**. The I/O addresses match the values:

  1. From the **Start** menu, select **Programs**.
  2. Select **SeehauHC11**, then click **Reconfig**. If the board I/O addresses match the values in the Seehau configuration, go to the "Configuring Address Settings with Windows Operating Systems" section in Chapter 2. Pay specific attention to alternate addressing.

     If you still encounter problems, contact Nohau Technical Support.

- **No**. The I/O addresses do not match the values:

  1. From the **Start** menu, select **Programs**.
  2. Select **SeehauHC11** and click **Reconfig**.
  3. Enter the appropriate values.

Now does the reset LED flash?

- **Yes**. The reset LED flashes.

Does Seehau start?

  - Yes. Troubleshooting is complete!
  - No. Seehau does not start. Go to **Step 4**.
  - No. The reset LED does not flash. Contact Nohau Technical Support.

Step 4. Will Seehau start if you configure for test mode after reset?

- **Yes**. Refer to Chapter 3, “Installing the Pod Boards.” Review the “Installing and Configuring the New MCU” section.

- **No**. Refer to Chapter 2, “Installing the Hardware.” Review the “Configuring Address Settings With Windows Operating Systems” section.

Is the problem solved?

  - Yes. Troubleshooting is complete!
  - No. Contact Nohau Technical Support.

Debugging the Parallel Port

Step 1. Disconnect other devices that might be sharing this parallel port (such as printers, zip, or jazz drives, parallel CD ROM drives, or software dongle keys).

Now is it working?

- **Yes**. You're done. You might opt to purchase an additional parallel port card.
- **No**. Do the following:

NT Users

Check the Nohauxx driver status by doing the following:

- To check the status, go to the **Start** menu. Select **Control Panel**. Then double-click **Devices**.
  - If the status shows **Started**, go to **Step 2**.
  - If the status shows **Stopped**, check the ParPort driver for **Started** status.
  - If the ParPort driver shows **Stopped** click **Start**.

- Now re-check the driver status.
  - If the driver shows **Started**, try restarting Seehau.
  - If the ParPort driver still shows **Stopped**, go to NT Diagnostics:
    1. From the **Start** menu, select **Programs**.
    2. Then select **Administrative Tools**, and click **Windows NT Diagnostics**. The Windows NT Diagnostics window opens.
    3. Click the **Resources** tab.
    4. Click **I/O Port**. Scroll down to address 378 (LPT1) and look for a device at this address.
    5. From the Control Panel, double-click **Devices**. Disable the device located at 378.
    6. Attempt to restart Seehau. If this fails, go to **Step 2**.

Windows 9x Users

Check the parallel port mode. Go to **Step 2**.

Windows 2000 Users

Verify that the Nohau11 device driver is properly installed. Do the following:

1. From the **Start** menu, select **Programs**. Select **Accessories**, then click **System Tools**.
2. Double-click **System Information**. The System Information window opens (Figure 74).

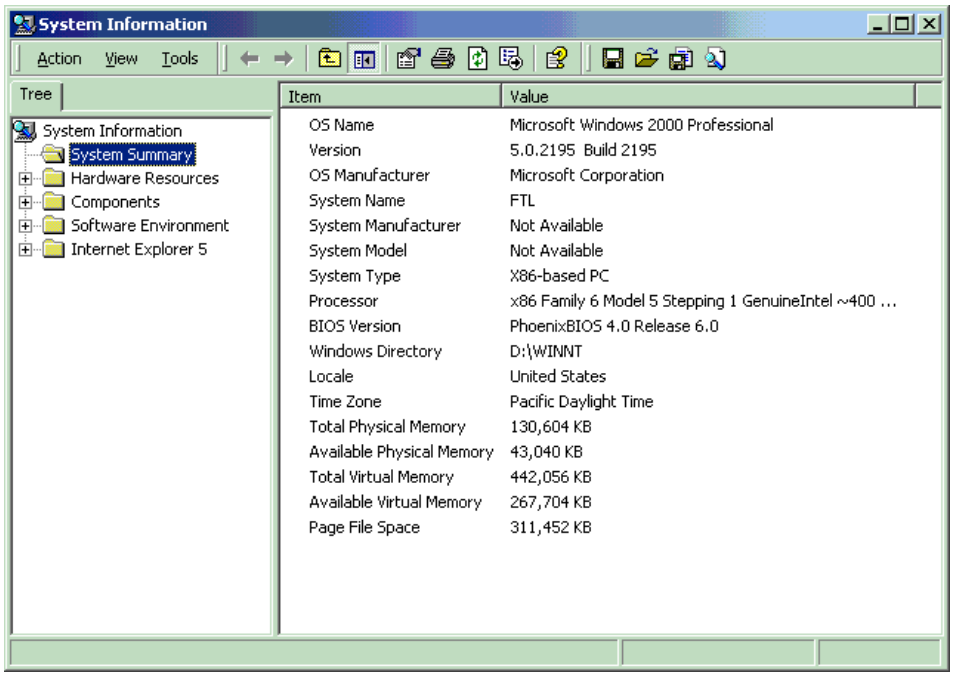

Figure 74. System Information Window

3. Click **Software Environment**.

4. Click **Drivers** to display a list of active drivers. Refer to the **Name** column and scroll down to Nohau11 (Figure 75).

5. In the **State** column, verify the driver is running. In the **Status** column, you should see OK.

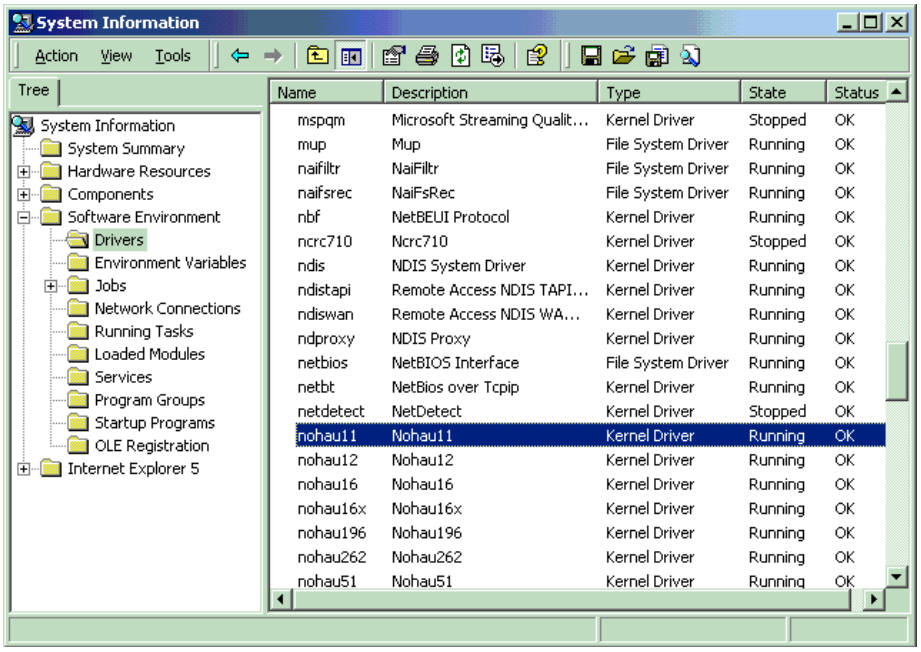

Figure 75. List of Active Drivers

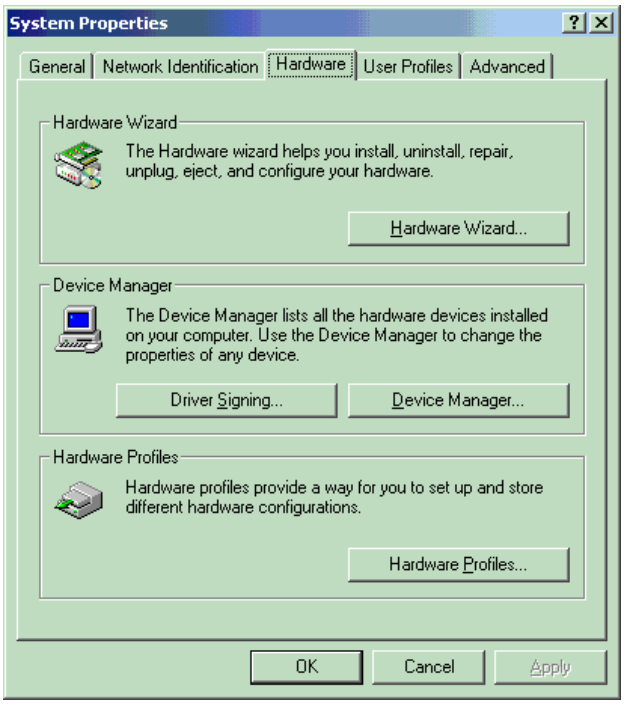

Figure 76. System Properties Window

If the ParPort driver still shows "Stopped," do the following:

1. Right-click the My Computer icon on your desktop, and select **Properties**. The System Properties window opens (Figure 76).

2. Click the **Hardware** tab. Then click **Device Manager**. The Device Manager window opens (Figure 77).

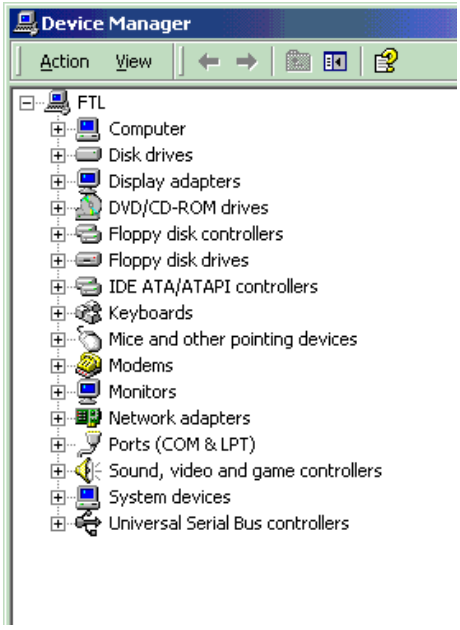

Figure 77. Device Manager Window

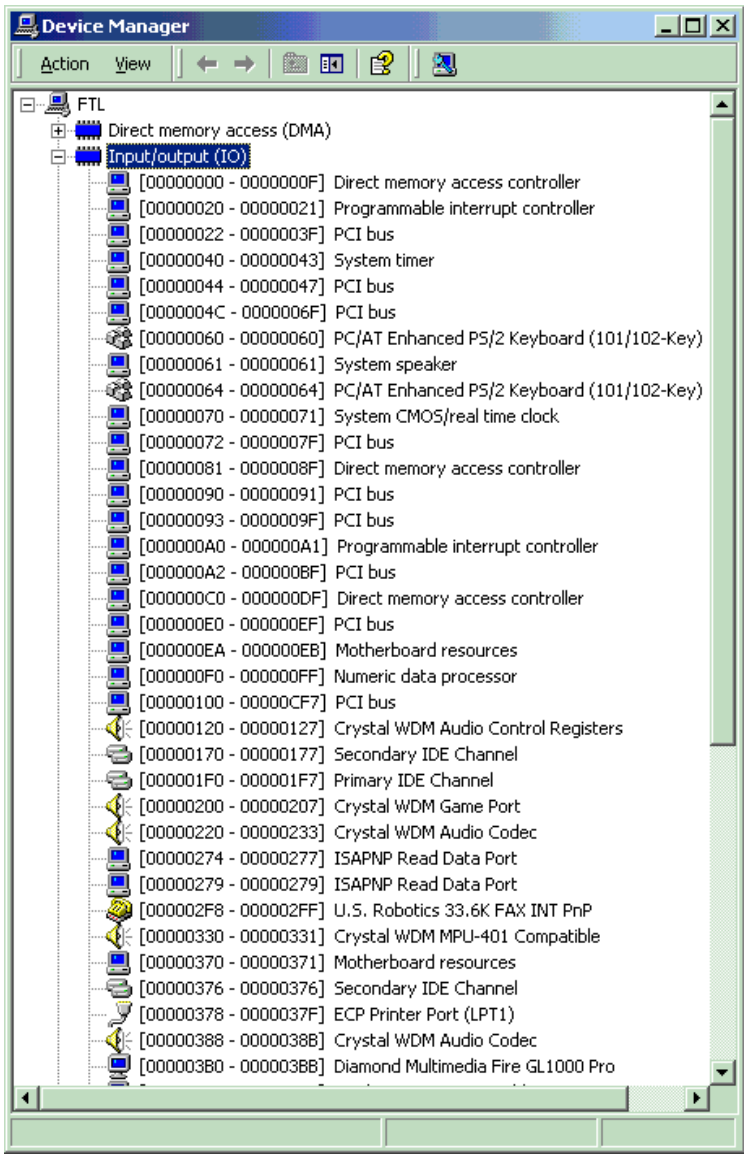

Figure 78. Device Manager Window Displaying the System Resources

3. In the Device Manager window, select the **View** menu. Then click **Resources by Type**. A window appears that shows system resources (Figure 78).

4. Double-click **Input/Output (I/O)**.

5. Scroll down to address 378 (LPT1) and look for a device at this address. Go back to the Control Panel and double-click **Devices**. Disable the device located at address 378. Attempt to restart Seehau. If this fails, proceed to **Step 2**.

Step 2. Check the parallel port mode.

1. Reboot and enter BIOS setup. From BIOS setup, check for one of the following parallel port modes:
   - Normal
   - Standard
   - Compatible
   - Output only
   - Bi-directional
2. Ensure that one of these modes is selected.
3. Then try selecting another mode.
4. Save your settings and reboot.

> **Note**
>
> The following modes have been known to cause problems: ECP, EPP, or ECP + EPP.

ISA

Step 1. If your pod has a reset LED, does it flash when you start Seehau?

- **Yes**. Go to **Step 3**.
- **No**. Go to **Step 2**.

Step 2. Do board I/O addresses match the values in the Seehau configuration?

If your reset LED does not flash or your pod is not equipped with a reset LED, verify that the board I/O addresses (for emulator and trace boards) match the values in the Seehau Configuration:

- **Yes**. **The I/O addresses match the values:**

  1. From the **Start** menu, select **Programs**.
  2. Select **SeehauHC11**, then click **Reconfig**. If the board I/O addresses match the values in the Seehau configuration, go to the "Configuring Address Settings with Windows Operating Systems" section in Chapter 2. Pay specific attention to alternate addressing.

If you still encounter problems, contact Nohau Technical Support.

- **No. The I/O addresses do not match the values:**

  1. From the **Start** menu, select **Programs**.
  2. Select **SeehauHC11** and click **Reconfig**.
  3. Enter the appropriate values.

Now does the reset LED flash?

- **Yes. The reset LED flashes.**

Does Seehau start?

  - Yes. Troubleshooting is complete!
  - No. Seehau does not start. Go to **Step 3**.
  - No. The reset LED does not flash. Contact Nohau Technical Support.

Step 3. Will Seehau start if you configure for test mode after reset?

- **Yes**. Refer to Chapter 3, "Installing the Pod Boards." Review the "Installing and Configuring the New MCU" section.

- **No**. Refer to Chapter 2, “Installing the Hardware.” Review the “Configuring Address Settings With Windows Operating Systems” section.

  Is the problem solved?

  – Yes. Troubleshooting is complete!
  – No. Contact Nohau Technical Support.

Known Device Driver Conflicts

Nohau is aware of potential device driver conflicts with certain network cards running on Novell/ Netware networks. Problems have been reported with both 3COM ISA network cards and some Novell network cards. Most of these problems have been experienced when running Windows NT or Windows 2000 operating systems.

Possible Symptoms

- When starting Seehau, communication with the network stops. (You will be unable to access resources on the network.)
- Seehau will not start.

A possible solution might be to change you network card. Nohau Support has not tested all network cards, although some customers have reported that the following network cards have resolved this conflict:

- Intel Ether Express Pro 10/100 ISA
- 3COM Etherlink III (905B or later) 10/100 PCI
- Bay Networks NetGear FA310TX 10/100 PCI

If the Emulator Does Not Start When Connected to the Target System

- Make sure power is applied to the target system.
- If the target has an external watchdog timer, make sure you do not select **Connect Reset to Target** in the **Emulator Configuration** dialog box.
- Check the E-Clock for a clean signal.
- Try switching the crystal jumpers/switches to the INT position.
- Disconnect the target. Make sure you change the crystal and power jumpers/switches to the INT position. Then try starting Seehau.

- Check the orientation of the target adapter. Confirm that the adapter is inserted properly. For more information, refer to Chapter 5, “Connecting the Emulator to Your Target Board.”

- Check for grounding problems. The emulator and target should have a solid common ground. Targets that are improperly grounded or designed with a *floating* ground might experience improper operation. A closer examination of control signals might reveal excessive over / undershoot or ground noise.

- If you are able to start the emulator, the problem is with one or more of the following critical target signals:
  - address and data bus
  - R/W signal
  - E-Clock

- Check the target for any device that is enabled by address qualification only and does not use E-Clock and/or R/W signals.

- Check the address/data bus loading. If your target design approaches maximum fanout for CPU drive capability, the emulator might not function correctly. This is caused by additional loads (approximately two TTL loads) from the pod board.

Target Does Not Operate Correctly

Problem:	**Serial port is not operating correctly.**
Cause/ Solution:	• Check jumpers/switches on pod board for external crystal selection. • Check signal and ground connections between pod and target.

Problem:	**Cannot access target memory or memory mapped I/O.**
Cause/ Solution:	• Check the **Memory Map Configuration** tab to ensure that the memory address range is mapped to target. (For details, refer to Chapter 4, "Installing and Configuring the Seehau Software." Go to the “Starting Seehau” section. • Check R/W and E-Clock signals on the target. • Make sure you have an expanded mode pod.

Problem:	**Cannot access SFRs that are write restricted to the first 64 E-Clock cycles after reset.**
Cause/ Solution:	• Use **Reset and Go** from the **Run** menu or the Reset and Go button on the Speedbar. • Check to make sure the SFR base address jumpers are set correctly.

Problem:	**Cannot access Ports B, C, and F (on non-multiplexed MCUs) on single-chip mode pods.**
Cause/ Solution:	• Check the **Memory Map** tab to ensure that the SFR memory address range is mapped to target. • Check to make sure the SFR base address jumpers are set correctly. Refer to Chapter 2, "Installing the Hardware," and review jumper details for your specific board. **Note**: If you are re-mapping the SFRs, you need both the original and destination addresses mapped to the target.

Problem:	• **Target memory or memory-mapped I/O gets unwanted data written in monitor mode (data corruption).** • **Target is not using E-Clock and/or Write to qualify writes to target memory.** • **E-Clock or WR gated/non-gated option is selected for non-gated.**
Cause/ Solution:	If your pod has E\GE and/or W\GW jumpers, set the jumpers to the GE and/or GW position. When in monitor mode, Nohau gates the E-Clock and Write strobe signals to prevent accidental writes to the target. If the target control logic does not qualify writes with these signals, target data might accidentally be corrupted. If your pod has E\GE and/or W\GW jumpers, setting these jumpers to the E or W position might cause unwanted writes to occur. If the target must have a continuous E-Clock (or WR) and you are experiencing target data corruption, contact Nohau Technical Support for assistance.

Avoid Setting the E/GE Pod Jumper to the E Position

Most EMUL68 pods have a strip of three jumper pins labeled E on one side of the strip and GE on the other side. Placing a jumper from the center pin to the outer pin labeled E will cause ECLK to be driven to the target for addresses that are mapped to the target. This occurs regardless of whether the emulator is in monitor mode or emulation mode. Due to the emulator's design, certain locations in your target space are written when the emulator is in monitor mode. The commands that are entered in monitor mode determine which locations are destroyed.

By placing a jumper from the center pin to the outer pin labeled GE, you will gate the ECLK signal so that ECLK is not driven to the target in monitor mode. This prevents writing to the target in monitor mode and is the default setting from the factory. This is the preferred selection.

Unless you have a very good reason, do not set this jumper to the E position. Leave the jumper at the factory default setting of GE. If you set it to the E position, then expect that certain locations on your target will be written to while the emulator is in monitor mode.

Special Emulation Behavior When Using the Reset and Go Command

Start your program with a Reset and Go command rather than two separate commands (Reset Chip and then Go) for the following scenario: if your power-on reset routine writes to any of the registers that must be written in the first 64 cycles after reset, such as the INIT and Option registers. This does not apply in test mode, since the 64-cycle limit is disabled in test mode.

The Reset and Go command can also be activated by pressing Ctrl G. This command causes the following actions to occur.

- Emulation mode becomes active, trace starts, and reset is driven low by the emulator on the same bus cycle.
- Reset is held active (low) for approximately 1/5 second and then released.
- The HC11 comes out of reset, fetches the address of the user's power-on reset routine, and branches to it.

Bus cycles that occur while reset is active are recorded in the trace buffer and counted by the execution cycle counter. Therefore, the first time you stop execution after Reset and Go, the cycle counter will be meaningless.

Appendix A. Bank Switching

Required Pod Board Connections for Tracing Banked Applications

Read this section if you have a trace board and are working with an application that consists of banking application code.

Bank Switch Control Signals

Signal Input	Pod Connection Point	Bank Bit #
E0	Pin labeled E0	0
E1	Pin labeled E1	1
E0	Pin 38 of the pod cable connector (K and P series pods there is a pin labeled B0 which is already connect to pin 38.)	2
E1	Pin 29 of the pod cable connector (K and P series pods there is a pin labeled B1 which is already connect to pin 29.)	3

The trace board will automatically trace the signals on E0 and E1 but will not automatically trace the signal connected to pins 38 and 29. This means to trace this signals you must also connect these to the PROBE1 bit 0 and 1 locations. Refer to Appendix E for more information and location on this pins. This will allow the trace to synchronize the appropriate source code for the respective bank that was executed and captured.

Normally, the PROBE1 inputs are connected to one of the microcontrollers ports. In order to trace the bank switch signal the shunting header must be removed from the jumper header pin so that you can connect the signal to the correct input on the PROBE1 pin header. Refer to Appendix E for identification of the of the PROBE1 pins.

The PROBE1 pin header can have two or three rows of pins. If it has three the PROBE input pin is located on the center pin or the header for each bit position. If it has two rows of pins, typically the lower row of pins is connected to the PROBE inputs. Refer to Appendix E for identification of the PROBE1 pins.

In the trace display the fifth digit of the address will represent the bank number following by the normal 64K address. For example, bank 5 address 8000h will be displayed as 58000h in the trace buffer.

Pod Setup for Bank Switching With K and P Series Pods

The POD–11KE has four pins where the bank switch signals can be attached: E0, E1, B0 and B1. The two high order bits B0 (E0 and B1 (E1) if used should also be connected to the PROBE1 bits 0 and 1 respectively. These are located to the left of the E0 and E1 pins on the 2 by 6 header near the bottom of the pod, if you hold the pod with the ribbon connector in your left hand. The connections from the pins labeled B0 and B1 to the pins directly below these two are for PROBE1 bits 0 and 1.

Support for the K and FL Families when Using Internal Banking Logic

Pod setup requires connecting the active XA signals on Port G to the following pins on the pod: E0, E1, B0, and B1 (listed from lowest to highest order). On POD–11KE, you do not have to solder bits 2 and 3 of the bank select signals (called E0 and E1 in the manual) to the cable connector as described, since POD–11KE has two pins labeled B0 and B1 to provide the same function.

Holding the pod with the cable connector on the left, you will notice there is a dual-row header with two rows of six pins about 1 inch above the bottom edge, just below resistor pack RP3. The top row of this header is connected to Port G pins from 0 to 5, defined from right to left beginning with Port G0 positioned to the far right on the top row. Port G bit 6 is positioned to the far left of the bottom row. Port G carries the XA signals. Some of these signals will need to be connected to the posts labeled E0, E1, B0 and B1 on the bottom row of the same header, starting with E0 as the second pin from the right on the bottom row, and going from right to left.

For example, if your application has 16 8K banks then connect the following pins on the header as described previously:

K or FL Pod Pin Configurations Using 16x8K Banks

XA Signal	Port G Pin	Connect To Pin
XA13	PG0	E0
XA14	PG1	E1
XA15	PG2	B0
XA16	PG3	B1

FL Pod Pin Configurations Using 32x16K Banks

XA Signal	Port G Pin	Connect To Pin
XA14	PG0	E0
XA15	PG1	E1
XA16	PG2	B0
XA17	PG3	B1
XA18	PG4	B2

On page 4–21 of the ‘K4 manual (Motorola document MC68HC11K4/D), Table 4–8 shows which XA signals are active depending on bank area size, and in certain cases, bank area location. The relevant information in that table is summarized below modified to assume that there are no more than 16 banks (the maximum allowed by SeehauHC11), and to ignore XA14 for 32K banks with bank area starting at 4000.

Sample MCU Configuration Showing Bank Size vs. Active XA Signals (K or FL Families)

Bank Size	Bank Area Start Location	Active XA Signals	Port G Pins to Connect	Max # of Banks
8K	Any 8K bndry	XA13–XA16	PG0–PG3	16
16K	Any 16K bndry	XA14–XA17	PG1–PG4	16
32K	0/4000/8000	XA15–XA18	PG2–PG5	15

Sample MCU Configuration Showing Bank Size vs. Active XA Signals (FL Family)

Bank Size	Bank Area Start Location	Active XA Signals	Port G Pins to Connect	Max # of Banks
8K	Any 8K bndry	XA13–XA17	PG0–PG4	32
16K	Any 16K bndry	XA14–XA18	PG1–PG5	32
32K	0/4000/8000	XA15–XA18	PG2–PG6	31

Note

If you use 64 banks, connect PG6 to BS3 using the XA jumper block. Configure Port G bit 6 to switch between the first 32 banks and the second 32 banks. The pod XA jumper B3/PG6, connects to the emulator only. To get the bankswitch bit 6 to the trace, connect PG6 to PROBE1-3 middle pin after removing the jumper. (PROBE1 is marked PD/PH.)

It is only necessary to connect active XA signals to pod posts E0, E1, B0 and B1 (B2 and B3 for POD–11FL0). For example, if you have only four banks connect only two XA signals.

If you use 32K banks and your bank area starts at 4000, do not connect XA14 to E0 even though XA14 is used in the addressing process. The signal XA14 is not actually part of the bank number. Connect XA15 to E0, XA16 to E1, XA17 to B0, and XA18 to B1.

Note

The 512K emulator board can support only fifteen 32K banks if you map the non-banked area to emulation memory. You must reserve 32K of emulation memory for the non-banked area. Similarly, if you map the 48K non-banked area to emulation memory, the 256K board can only support thirteen 16K banks.

In order to see source code when using more than four banks, connect the active high-order bank select signals B0 through B3 to PROBE1, pins 0 through 3. The low order bank signals E0 and E1 are already connected to the trace board. Refer to Appendix E, “Trace Probe Headers” for the location of PROBE1.

Appendix B. Disabling Cycle Stretching

Disable cycle stretching on the K0/K1/K3/K4 if you are emulating single-chip mode applications. Clock stretching is not needed in expanded mode, or emulating a KA4 with factory-supplied K4 MCU on the pod.

If you are emulating any K family processor and if one of the following is true, you should dis-able the default cycle stretching on the K4's program chip-select pin in expanded mode by writing zeroes to bits 0 and 1 of the Chip Select Clock Stretch Register, (CSCSTR):

- You are emulating a single-chip mode application using POD–11KS.
- You are emulating an expanded-mode application using POD–11KE, you use external memory selected by the program chip-select pin to store your program, and that device (EPROM, flash or RAM) is fast enough for the MCU to run without cycle stretching on the program chip select pin.
- You are emulating a KA4 in either single-chip or expanded mode using a POD–11KE or POD–11KS with the factory-supplied K4 MCU in the socket of the pod.

This should be done as one of the first instructions of your power-on reset interrupt service routine and can be completed with one instruction, CLR CSCSTR. In order to reduce program load time, you can also disable cycle stretching on K family parts before loading your program by using the cycle stretching disable check box option.

With the Seehau interface, the same can be accomplished by selecting the **Cycle Stretching Disable** option in the **Emulator Hardware Configuration** tab from the **Config** menu.

> **Note**
>
> This check box has no effect on chips that do not support cycle stretching.

The default stretch on the program chip select pin is one cycle in expanded mode and zero cycles in test, single-chip and bootstrap modes. Disabling the default stretch on CSPROG will have the following benefits:

- Your program will run nearly twice as fast.
- Default stretching on the program chip select pin reduces the 64-cycle limit after reset for setting certain registers to an effective value of approximately 32 cycles; disabling cycle stretching as soon as possible after reset will prevent most of this reduction.
- The emulator will respond more quickly to commands entered that require communication with the emulator after you have disabled stretching.

For single-chip mode applications, disabling cycle stretching is essential for accurate emulation because the HC11K4 in the pod is actually operating in expanded mode, and so stretching is enabled out of reset. If you do not disable stretching, your program will appear to run at approximately half the speed that it will run when your code is burned into the chip and installed directly into your target.

If you want to avoid the requirement of clearing the E-Clock stretch register for KA4 applications, you can purchase an adapter from Nohau which allows you to replace the factory-supplied K4 in the pod with a user-supplied KA4 MCU (Part # EDI/68PL/84PL-ZL-KA4/K4). Contact Nohau Customer Support for details.

Adjusting Trace Time Stamp for Cycle Stretching

If your pod has a K1/K2/K4 part, and you do not disable cycle stretching in expanded mode as described in the last two sections of this document, then you should set crystal frequency in Seehau's **Emulator Configuration** tab to half of the actual frequency of your crystal. Default cycle stretching on the K and C family parts doubles the length of all cycles that access program memory, which is probably most of them. Cycles that access internal memory and special function registers are not doubled. Dividing crystal frequency by two if stretching is enabled is necessary so that Seehau can correctly calculate timeout values to perform certain operations that require communication with the emulator.

Appendix C. Considerations for Placement of Internal RAM, Special Function Registers, and EEPROM

Most versions of the HC11 allow the application program to relocate internal RAM and special function registers (SFRs) to the start of any 4K boundary by writing to the INIT register. In some cases, the on-chip EEPROM's location can be selected by reprogramming the CONFIG register or, on the K family and its derivatives, by writing to the INIT2 register.

Due to the internal architecture of EMUL68, there are a few restrictions on where internal memory can be relocated:

- If the internal EEPROM occupies an area up to and including a location xFFF where "x" represents any hex digit from 0 to F, then the EEPROM cannot be located from Bxxx to BFFF in test mode, or at Fxxx to FFFF in expanded mode. Parts that currently fall into this category include the HC811E2, the HC11F1, and the K, N and P families.

  Note

  If you are emulating a part that runs in single-chip mode on your target system, the chip in the pod actually runs in expanded mode.

- On the 68HC811E2 where the EEPROM is 2K in size, in addition to the restriction noted above it cannot be moved to reside at 4800 – 4FFF, 5800 – 5FFF, or 6800 – 6FFF.

- The internal RAM on the F1, K4, or P2 cannot be relocated to 5000 or 6000.

- The internal RAM on the PH8 cannot be relocated to 3000, 4000, 5000, 6000 or 7000.

Appendix D. Configuring Seehau to Read the On-Chip EPROM

> **Note**
>
> The following steps should be adaptable to other processors in the HC11 family, although they have not yet been tested with other processors.

Do the following steps to enable the on-chip EPROM on the HC11E9 processor:

1. Start the emulator. From the Seehau **Config** menu, select **Emulator**. The **Hardware Config** tab opens.
2. In the Miscellaneous area, select **Test Mode after Reset**. Then click **OK**.
3. Right-click on the Register window, and select **Special Register**. A window opens where you can add SFRs.
4. In the SFR window, double-click **Special**.
5. Scroll down and click **Config**. Then click the **>** to add the Config register.
6. Click **Add to**. A list menu appears. Select **New Window**. A new Register window opens with the Config register displayed. Now close the SFR window.
7. From the New Register window, right-click the Config register. Select **Edit bits**. The Bit Editor (CONFIG) window is displayed.
8. From this window select **bit 1 ROMON**. Click **OK**.
9. Click the Reset the chip icon  on the Seehau Speedbar.

Do the following steps to copy the contents of the EPROM to emulation RAM:

1. Verify that all memory is mapped to the emulator.
2. Right-click in the Data window. Select **Block**, then click **Block Move**. The **Block Move** dialog box opens. Enter data as shown in Figure 79:

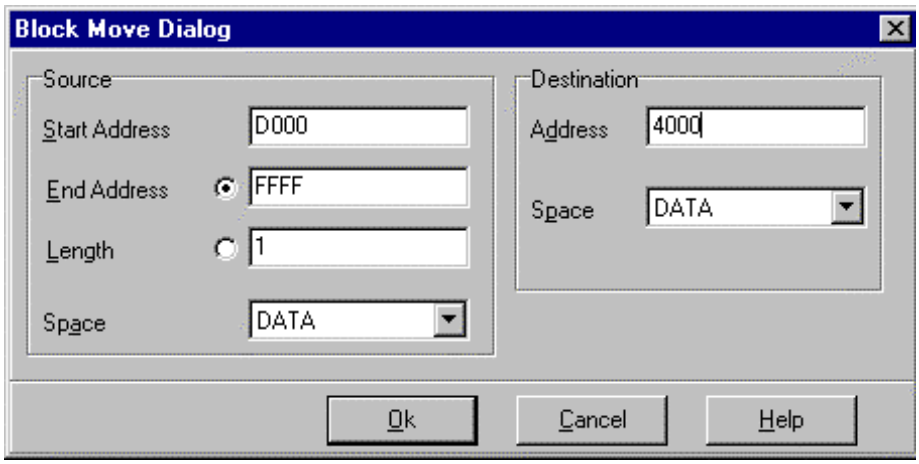

Figure 79. Block Move Dialog Box 1

3. From the New Register window, right-click the Config register. Select **Edit bits**. The Bit Editor (CONFIG) window opens.

4. From this window clear **bit 1 ROMON**. Click **OK**.

5. Click the Reset the chip icon  on the Seehau Speedbar.

6. Right-click in the Data window. Select **Block**, then click **Block Move**. The **Block Move** dialog box opens. Enter data as shown in Figure 80:

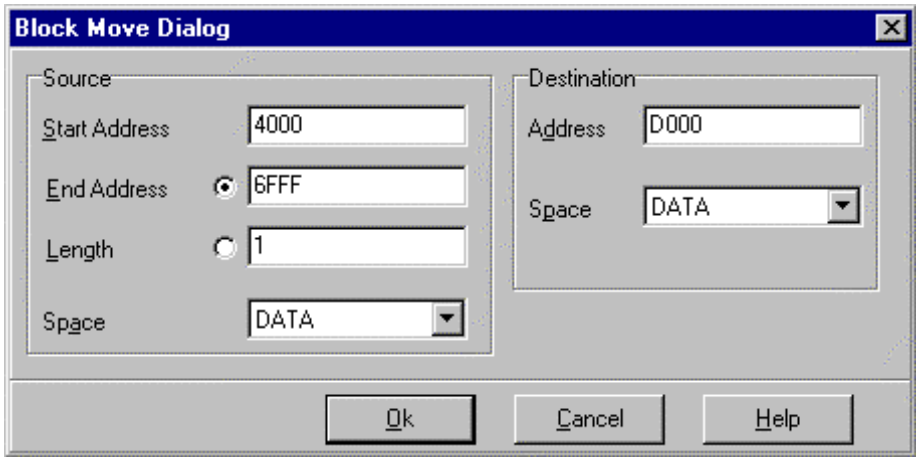

Figure 80. Block Move Dialog Box 2

7. Right-click in the Data window. Click **Save Memory Dump**. The **Save Memory to File (DATA)** dialog box opens.

8. Click **OK**.

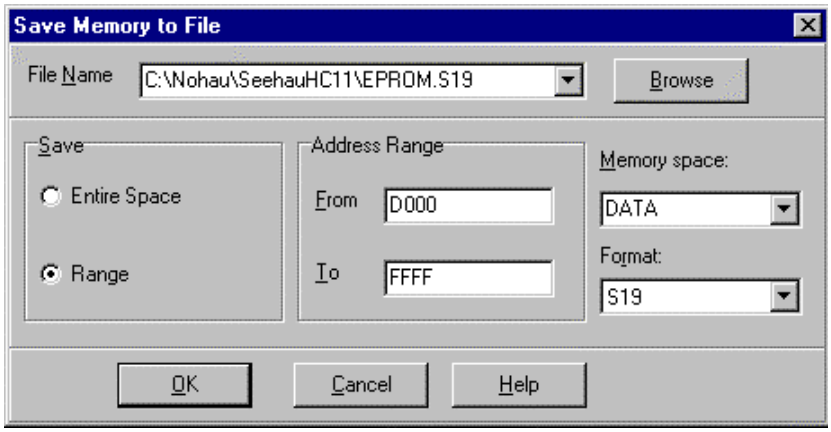

Figure 81. Save Memory Dump Dialog Box

Appendix E. Trace Probe Headers

Each pod has two clusters of 100 mil header pins, usually in two or three rows of eight pins per row. One of the rows of each cluster is called PROBE0 or PROBE1. The TTL signals on these probes are captured in the trace buffer. When there are three rows, the middle row is always the probe signals and the outside rows connect to the processor's I/O ports. Specific I/O ports are usually labeled on the pod silkscreen as Px, such as PA for Port A, PD for Port D, and so on. If there are only two rows, then one row is the probe pins and the other row carries the port I/O signals. Jumpers can be used to connect the I/O port pins to the probe pins, and pods usually come from the factory with these jumpers installed.

To locate a particular set of probe pins (0 or 1), a good general rule is that except for POD–11S, PROBE0 is associated with Port D. On POD–11S, PROBE1 is associated with Port D.

The following table helps you identify which of the two header clusters is associated with a particular probe:

		PROBE0		PROBE1	
Pod Type	**Header Location**	**Location**	**Ports**	**Location**	**Ports**
POD–11E	LR	JB4/Left	**D**	JB5/Right	**A**, E
POD–11S	LR	JB4/Left	E, C, **A**	JB5/Right	B, **D**
POD–-11DE	LR	JB9/Left	**D**	JB10/Right	**A**
POD–11DS	LR	Left	**D**, B	Right	**A**, C
POD–11FE	UR	Right	**G**, D	Left	**E**, A
POD–11KS	LR	Right	**D**, H	Left	**A**, G
POD–FL0	LR	Left	**D**, H	Right	**A**, J
POD–11KE	UR	Left	**D**, H	Right	**A**, E
POD–11Px is the same as POD–11Kx.					

Note

The use of bold text in this table designates a default connection.

For CE, Gx, Jx, Lx, and Nx, contact Nohau Technical Support.

The terms left, right, top, bottom, upper and lower used in this table apply when the pod is held so that the ribbon cable connector is on the left.

Header Location—Location on the pod of probe header clusters.

LR—Lower right corner of pod, when held with ribbon cable connector to the left.

UR—Upper right corner of pod, when held with ribbon cable connector to the left.

PROBE0/1 Location—Location on the pod of this probe's header cluster with respect to the other probe's header.

JBx—Silkscreen label for header cluster, if there is one.

Left, Right, Upper, Lower—Describes the location of the header cluster containing pins for this probe with respect to the other probe.

PROBE0/1 Inputs—I/O ports available in this header cluster. If two letters appear in this column, the header cluster is three rows of eight pins each, with probe pins as the middle row. The first letter shown is the port on the top row, the second letter shown is the port on the bottom row.

If only one letter appears, you have a POD–11E or POD–11DE and the probe header cluster contains only two rows. In that case, the letter shown in this column represents the port on the top row, and the bottom row (closest to the edge of the pod) carries the probe pins.

If you have a POD–11S, the header cluster associated with PROBE0 is four rows, which contain from top to bottom as follows: Port E, Port C, PROBE0, Port A.

Appendix F. Restrictions and Differences

The following describes some restrictions that apply when the EMUL68–PC emulates the different modes of the 68HC11. Differences are noted between operation of the EMUL68–PC and the 68HC11.

Executing Code from the HC11's Internal Memory

Code can be executed for the microcontroller's internal RAM or EEPROM memories. However, it is not possible to break emulation if the code is executing from either of these two memory areas. If a breakpoint is encountered in these areas, the emulator will enter a pending break state. As soon as the emulator exits the regions of the internal memory, emulation will stop.

If you try to manually stop emulation, two Seehau error message boxes open. The first error message displays **Failed to break emulation** and the second message displays **trying to stop emulation**. When the program exits the internal memory access and moves to external memory, the emulator can stop execution.

If you are single-stepping through code and the code jumps to internal memory, two Seehau error message boxes open. The first message displays **Failed to break emulation** and the second message displays **waiting for emulation to stop > step**. When the program exits the internal memory access and moves to external memory, the emulator can stop execution.

If the code does not exit the internal memory and you want to stop emulation, press the reset button on the pod. If you try using the Reset button in the Seehau interface, you will receive a few Seehau error messages, and the software will exit without saving settings.

When code is executed in internal memory, data from data read cycles will not appear on the port pins unless you are in Test mode and the IRV bit in HPRIO is set. The trace cannot record the correct data from these cycles. Therefore, the trace disassembler cannot interpret the data and will display erroneous mnemonics or will display the recorded data in Frame mode. If this causes a problem, use this workaround: Re-map the internal memory to an unused area. The emulator memory can then be used in place of the internal memory. For example:

LDAA #21	For K, P, and D pods use the values of 20 (hex).
STAA $103D	Address of the INIT register. Use address 3D (hex) for K, P, and D pods.
NOP	Sets a breakpoint on this instruction.

From a Data window, modify the values at address FFFE to C0 and FFFF to 00. Modify the values from the **Run** menu. Then click **Reset and Go** to execute the code. This command moves the internal RAM from the normal reset condition to address 2000h. Code should now execute from emulation memory so you can debug your application.

Note

INIT can only be written to during the first 64 cycles, therefore the Reset and Go operation must be used.

For the EEPROM memory, do the following:

1. From the **Config** menu, click **Emulator**. The **HDW Config** tab opens.
2. From the **HDW Config** tab, select the **Test Mode after Reset** option.
3. Click **OK**.
4. From the Data window, go to the address of the CONFIG register (103Fh or 3Fh depending on your processor type) and enter a new value to turn off the EEON bit.
5. Now click the Reset button on the Speedbar, and the change should take effect.
6. Bring up the **HDW Config** tab again and clear the **Test Mode after Reset** option.
7. Click **OK**.

Stack Considerations

If the stack is within the register bank or in EEPROM when a break occurs, the break will be executed (if on an external instruction). However, you will get an error message. This situation most likely occurs from a bug in the user program, and emulation should not be continued. Instead, go to the **Run** menu, and click **Full Reset** so that emulation can resume.

The User stack of the 68HC11 is normally not used by the emulator. This difference has a noticeable effect only if the stack is in internal RAM when emulation is started: the monitor program performs a PSHA and a PULA which destroys the next unused byte on the stack. The PSHA and PULA allows you to load the A-register without changing any of the condition code flags. A problem would only occur if the next unused byte on the stack was in fact used for data storage, for example. In other words, the stack was at its lowest allowable point. No problem occurs if the stack is in external memory because the emulator's monitor mode overlay memory is used.

TCNT

The free-running Timer Counter Register (TCNT) is not stopped when emulation breaks. There is actually no way to stop it, but everything connected to it, (such as Output Compare, Input Capture, and so on), will be disabled. The Timer Counter is read when the emulation breaks and the value in the TCNT Special Function Register shows the value of the Timer Counter after emulation was stopped.

EEPROM Writes

EEPROM is taken care of by the emulator when writing. As long as the EEPROM is writable, SeehauHC11 will do the special procedure required to get that data written. If the write is to an EEPROM, the EEPROM must be enabled (bit 0 in the CONFIG register). If the CPU used has the BPROT register, that register has to be set properly so as not to protect the memory. From the **Config** menu, click **Emulator**. From the **HDW Config** tab, select the BPROT override for those chips that have the BPROT register. Enter a value in and select this option to enable or disable protection to areas of the EEPROM memory.

Special Function Registers

The 68HC11 has many Special Function Registers (SFR). Three of these SFRs deserve comment: INIT, HPRIO, and CONFIG.

INIT—The SeehauHC11 maintains a shadow register of INIT. The emulator is therefore able to keep track of where the registers and RAM are mapped. The 68HC11 can only write to INIT during the first 64 cycles after a hardware reset to the chip, (in either of the two protected modes, expanded and single-chip mode). This is emulated by the shadow register. However, it only works after using the Reset and Go operation, (or a hardware reset). The Reset and Go operation is not the same as using the Reset button followed by the Go button.

In the two unprotected modes, Bootstrap and Test, the 64-cycle restriction does not apply. If a switch is made from either of the two unprotected modes to either of the protected modes, the INIT register cannot be written to. This is not emulated, however. So, if a write to INIT takes place after a switch to protected mode, the shadow register of the emulator changes, but not the INIT itself. This must be avoided because the emulator will lose track of the mapping of the registers and internal RAM.

HPRIO—The 68HC11 in the EMUL68-PC must always run in either Expanded mode or in Test mode. The Single-chip mode and the Bootstrap mode are emulated using the Expanded mode and the Test mode. (See detailed description of 68HC11 modes later in this chapter.) This means that the MDA bit in HPRIO must always be left high. (For details, refer to the following “Single-Chip Pod, POD–11S” section.) If the MDA bit is cleared, the emulator will fail because the chip is then put into Single-Chip or Bootstrap mode. Therefore, the chip is unable to communicate with the rest of the hardware through the expanded ports. Use a Reset Chip command to clear the problem.

CONFIG—The CONFIG register can only be changed in Test or Bootstrap mode. The EEPROM must be erased before new data can be written into the register. Note that any changes to CONFIG will not take effect until after the next reset of the chip. This also means that the changes written into CONFIG cannot be read back until after the next Reset. (See 68HC11 data book for details.)

68HC11 Modes

The 68HC11 has four different modes in which the processor can work. It is possible to switch between modes. The EMUL68-PC uses two different pod boards to emulate the different modes. This places some restrictions on how switching between different modes can be emulated.

Single-Chip Pod, POD–11S

This pod is designed to emulate the 68HC11 in Single-Chip mode and in Bootstrap mode. POD-11S includes the 68HC24 port replacement unit, which replaces the address and data ports so that the 68HC11 can still be run in Expanded mode. The 68HC24 makes it possible for the pod to read instructions from the emulator memory and to communicate with the rest of the hardware on the emulator board.

All memory except the upper 8K (ROM in the single-chip) should be mapped to TARGET. The register bank ***must*** be mapped to TARGET; otherwise the 68HC24 will not be reached by the address and data bus.

Because an Expanded mode 68HC11 together with the 68HC24 port expansion unit are used to emulate Single-Chip mode, the 68HC11 must stay in expanded mode. The hardware pin MOD A is always held high on the pod so that at Reset, the 68HC11 will be forced into either Expanded mode or Test mode. In Test mode, the MDA bit in HPRIO can be cleared. This forces the 68HC11 into Bootstrap mode. If this happens, the 68HC11 will lose contact with the rest of the emulator and nothing will work. Therefore, be sure to keep MDA high at all times.

The IRQ is connected both to the target system and to the 68HC24. These two signals are ANDed before they are connected to the 68HC11. This means that if interrupts are generated from the 68HC24, only level sensitive interrupts can be used. In other words, you cannot use edge triggered interrupts if the parallel interface interrupt is used.

Expanded-Chip Pod, POD–11E

POD–11E is designed to emulate the 68HC11 in Expanded mode and in Test mode. As with the POD-11S, it is important to keep the MDA bit in HPRIO high at all times.

Although the Bootstrap mode is a single chip mode, you can still emulate the Bootstrap mode with the POD-11E. To do this, perform the following procedure:

1. From the **Config** menu, click **Emulator**. In the **HDW Config** tab, select the **Test Mode after Reset** option.

2. Map segment B000 to EMUL. This is the 4K bank where the Bootstrap program in EEPROM resides.

3. Set the RBOOT bit of the HPRIO register.

4. From the Data window, go to address 103C and enter a value that is based on the current value logically ORed with 80h.

5. From the Data window, right-click any of the data cells and select the option **Block**. Then select **Block Move**.

6. For **Starting Address**, enter `BF40`. For the **Ending Address**, enter `BFFF`. In the destination address field, enter `BF40`.

7. Select **OK**.

   This procedure copies the Bootstrap program from the EEPROM to the emulator RAM. (When the byte read from the EEPROM is written back to the same address, it is also written to the emulator.)

8. From the Data window, go to address 103C. Enter a value that is based on the current value logically ANDed with 7Fh.

Now the Boot program is moved from the internal boot ROM to the emulation memory. The program will be started whenever the processor is reset and started in Test mode. The switch from Boot to Expanded mode is really a switch from Test to Expanded mode, so both MDA and SMOD should be changed.

In the emulator, the switch happens by clearing SMOD. But when the emulator is replaced by a CPU and an EPROM, the switch does not happen that way. A change of SMOD will switch from Boot to Single-chip mode.

Different chips might contain different Boot programs, depending on security option, size of internal RAM, and so on. If you decide to save the Boot program to a file, use that file only for the specific processor it has been read from.

ROMON Bit Set Problem

You should never set the ROMON bit of the CONFIG register. This enables the ROM internal to the processor and thus prevents the emulator from starting after reset.

To get out of this situation, do the following:

1. Start Seehau with the **Seehau Reconfig** option.

2. Walk through the options and on the second **HDW Config** dialog box, select the option for **Test Mode after Reset**. Then finish the reconfiguration and start the emulator hardware.

3. Select a Data window that is set for Code or Data space and go to the address of the CONFIG register for the chip you are using. Now enter a new value in this memory cell that will not have the ROMON bit set.

4. Click the Reset button on the Speedbar to set the value.

5. From the **Config** menu, click **Emulator**.

6. Clear the **Test Mode after Reset** option, and click **OK**.

> **Note**
>
> Due to memory layout, this procedure will not work for the 68HC711E20 microcontroller. Either you can use an external chip programmer to effect this operation, or you can order the parts from Motorola with this bit already disabled.

Appendix G. Address Examples

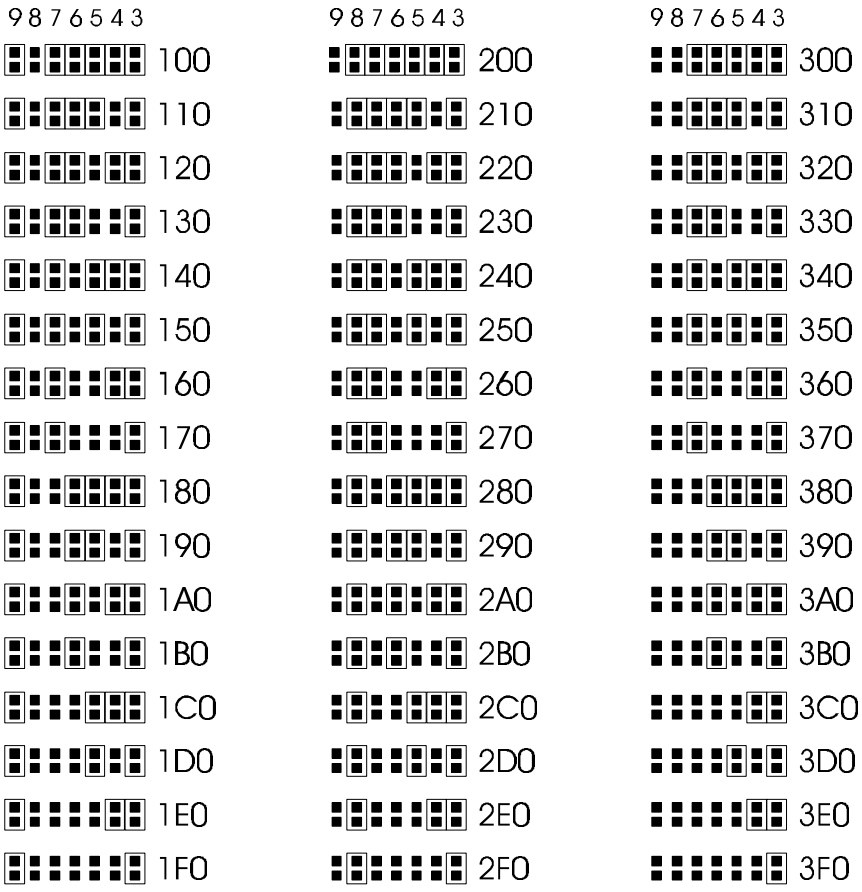

Figure 82. Emulator Addressing Example

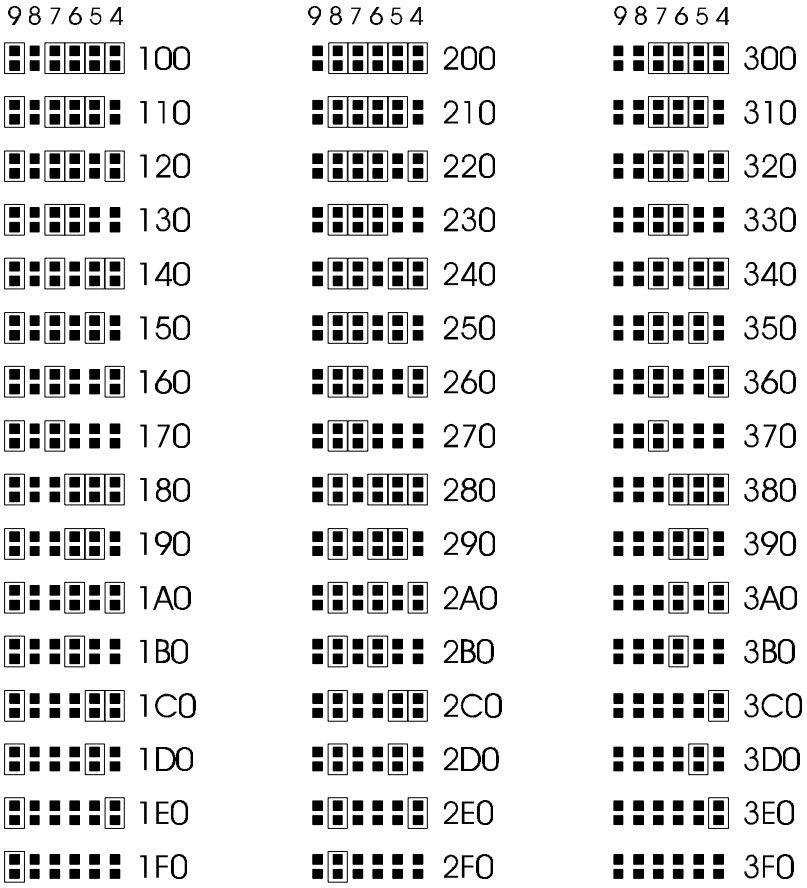

Figure 83. Standard Trace Addressing Example

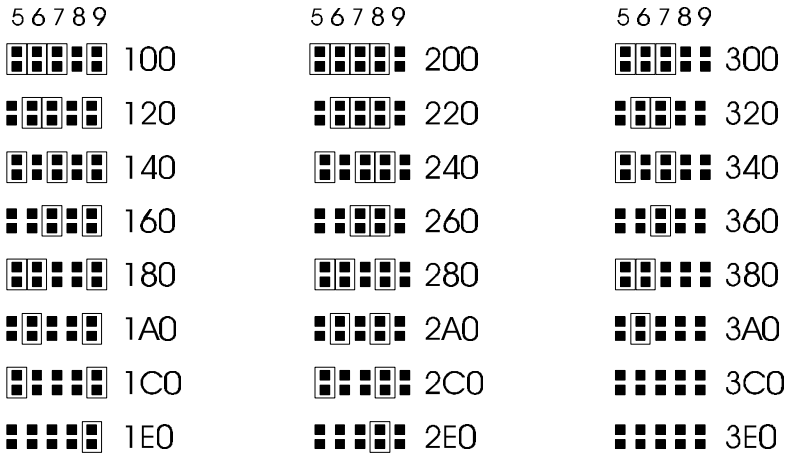

Figure 84. Enhanced Trace Addressing Example

Index

C

D

E

F

G

H

I

J

K

L

M

Q

R

S

T

U

V

W

Z