EMUL16/300-PC™

User Guide

Product Notes

European CE Requirements

We have 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.

Warnings

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.

When using the pod with a target, disable all pod resources that are duplicated on the target. Failure to disable the pod's resources may damage the pod or the target or both. This includes the MCU, the serial port, RAM, crystal, and, particularly, the power supply. If using the clip to attach to the target, remove the MCU from the pod.

When installing a controller into a pod, never press on the chip body. Press only on the carrier or cover. Pressing on the chip may bend pins and cause short circuits.

Minimum 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 at support@icetech.com.

The following are **minimum system requirements**:

- Pentium 200 (Pentium II or faster is recommended)
- Single-Processor System
- Monitor resolution of 800 x 600 or better for best graphics display
- Windows 95, 98, NT, or 2000
- Random Access Memory (RAM)
  - For Windows 95/98: 32 MB
  - For Windows NT/2000: 64 MB
- Two ISA slots in your PC if the optional trace board is purchased, otherwise purchase the HSP box, part # EMUL-PC/BOX-HSP.

You can custom configure the hardware to your requirements with various jumpers.

About This Guide

This *EMUL16/300–PC User Guide* is intended for both novice and advanced users. It is important to understand that the EMUL16–PC is different from the EMUL300–PC. When seeking technical support from Nohau you need to specify which family you seeking support for (16 or 300).

The EMUL16/300–PC is a personal computer based in-circuit emulator for Motorola's 68HC16 16-bit and 683xx 32-bit microprocessors. This guide helps you get started with the basics of setting up, configuring, and running the Seehau software and the emulator. For detailed hardware documentation, please refer to Chapter 5 of this guide. If you have any questions contact Nohau Technical Support at support@icetech.com or refer to the "Sales Offices, Representatives and Distributors" list at the end of this guide.

> **Note**
>
> Customers also need the Motorola *Technical Summary Users Manual* for your MCU.

Online context sensitive Help is also available from the Seehau software by pressing the F1 or the Help keys, depending on the type of keyboard you have.

The *EMUL16/300–PC Getting Started Guide* introduces the following tasks:

- Installing and Configuring the Seehau Software
- Installing and Configuring the Emulator
- Installing and Configuring Trace Boards
- Installing and Configuring Pods
- Starting the Emulator and Seehau Software
- Time Program Examples
- Trace Memory Example
- Macro Example
- Shutting Down Seehau Software
- POD–(9)16Y1
- POD–(9)16Y3
- POD–(9)16X1
- Troubleshooting Tips
- Hex Pin Addressing

Downloading EMUL16/300–PC Product Documentation

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

1. Open Nohau's home page at www.icetech.com.
2. Click **Publications**.
3. Click **Manuals**.
4. Scroll down to EMUL16/300–PC. Then select **EMUL16/300–PC** to download a PDF version of the guide.

Overview of the EMUL16/300–PC Emulator System

The basic hardware for the EMUL16/300–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 two short ribbon cables.
- Five-foot twisted-pair ribbon cable—connects the emulator and pod.
- Target adapter—allows you to connect the pod board to your target system.

To connect to your target system, the pod board usually requires an adapter. To determine the adapter board that your pod requires check the price list, your representative or Nohau Technical Support.

The EMUL16/300–PC/BDM system requires a working target board. It can either be the user's working target board or one provided by Nohau. (See Chapter 3, "Installing and Configuring the Emulator Board" and Chapter 5, "Installing the Pod Boards" in this guide.) An optional trace board (Chapter 4, "Installing and Configuring the Trace Board" in this guide) can be added to full emulator systems for tracing capabilities.

Four system configurations are available. Two are full emulator systems and two are BDM systems. The full emulator systems have trace capability while the BDM systems do not have trace capability. The four systems available are:

- PC Plug-In. See the following "ISA Card Emulator (PC Plug-In)" section. A full emulator system.
- High-Speed Parallel Box (HSP) connects to the parallel printer port. See the following "High-Speed Parallel Box (HSP)" section. A full emulator system.
- Enhanced Parallel Cable (EPC). See the following "EPC" section. A BDM system with no trace capability.
- Low Cost-Industry Standard Architecture (LC-ISA). See the following "LC-ISA" section. A BDM system with no trace capability.

ISA Card Emulator (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 trace board can also be plugged into the PC and connects to the emulator board through two ribbon cables.

> **Note**
>
> If the optional trace board were purchased for PC installation, you would need to ensure that your computer motherboard has at least two open ISA slots or you will need to purchase the HSP box.

High-Speed Parallel Box (HSP)

The HSP box lets you use the in-circuit emulator and optional trace board where no ISA slots are available. Nohau company personnel mount the emulator board, HSP card, and optional trace board in the HSP box chassis. The trace board connects to the emulator board through two ribbon cables. The pod board connects to the emulator board in the HSP with a five-foot ribbon cable. The HSP card connects to the PC's parallel printer port. This is the second most portable method of connection when used with a laptop computer and gives you full trace capability.

Figure 1. HSP Box Connected to a Pod Board and Laptop Computer

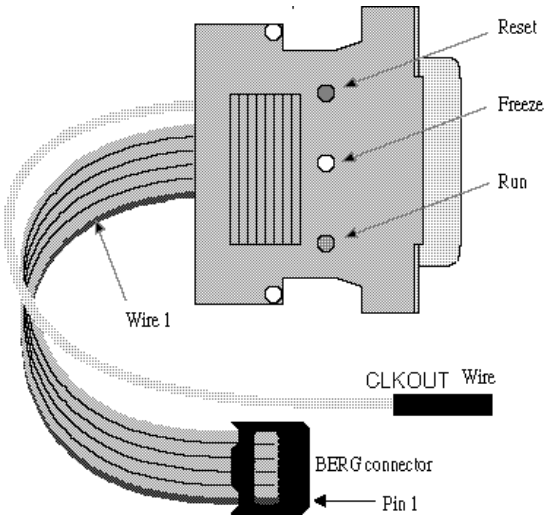


Figure 2. BDM Pod With BERG Connector

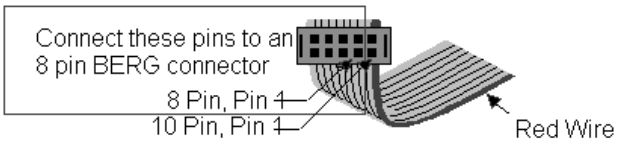


Figure 3. BERG Connector

Emulator Parallel Cable (EPC)

The EPC is an interface device plugged into the parallel port of your PC on one end and a Background Debug Mode (BDM) pod on the other end. The BDM is a low-cost development tool for all CPU16/32/32+ microcontrollers and includes status indicator LEDs. The LEDs indicate Reset, Freeze and Run. The pod is connected to a cable terminated in a 10-pin BERG connector, which in turn is connected to the target system. The BERG connector is not keyed. Wire one on the ribbon cable is red. If the header on the target is an older design and has eight pins, insert the target pin one into the BERG plug three leaving BERG plugs one and two empty. The EPC provides the most portable method of connection and is perfect for laptop computers. The trace board is not an option with this setup.

Low Cost-Industry Standard Architecture (LC-ISA)

The EMUL/LC–ISA board is an 8-bit PC card that fits into any ISA slot in your PC. This board must be connected to a pod to operate. Low cost emulators do not have Shadow RAM, or provision for a real-time trace (or the ability to add a trace board). The maximum frequency is set by the frequency limit on the pod.

You can configure the emulator hardware to your requirements with various jumpers. For details on configuring your emulator board, refer to Chapter 3, “Installing and Configuring the Emulator Board” in this guide. For details about the optional trace board, refer to Chapter 4, “Installing and Configuring the Trace Board” in this guide, or refer to Seehau Help.

User Interface

The emulator is configured and operated by the Seehau user interface. Seehau is a high-level language user interface that allows you to perform the following tasks:

- 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 software breakpoints (there are no hardware breakpoints in the 16/300 family).
- Analyze code with Program Performance Analysis (PPA)

Quick Start for Installing Your Emulator System

The following illustration shows the major steps for installing and configuring the EMUL16/300–PC. For details, refer to the chapter referenced in each step.

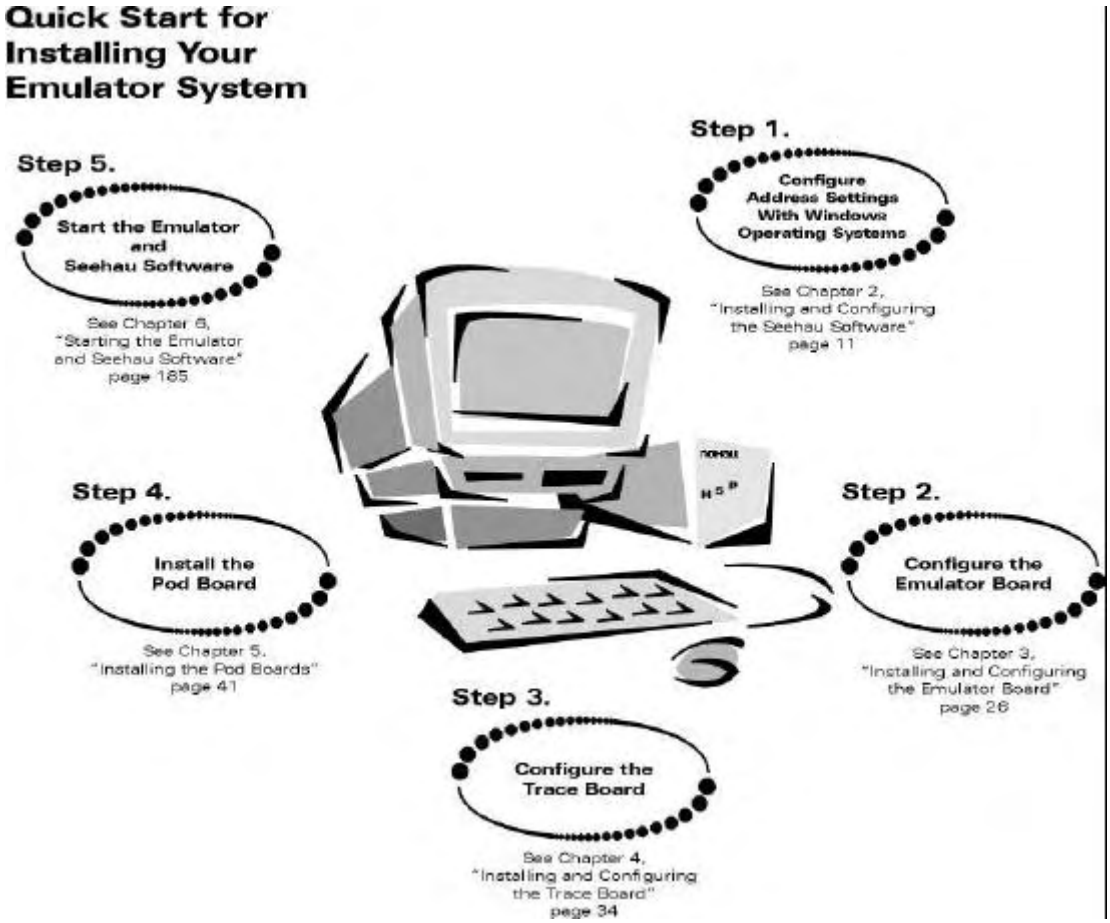


Figure 4. Steps for Installing and Configuring the EMUL16/300–PC and Seehau Software

Quick Start for Installing the Hardware

The following illustration shows the major steps for installing the EMUL16/300–PC hardware.

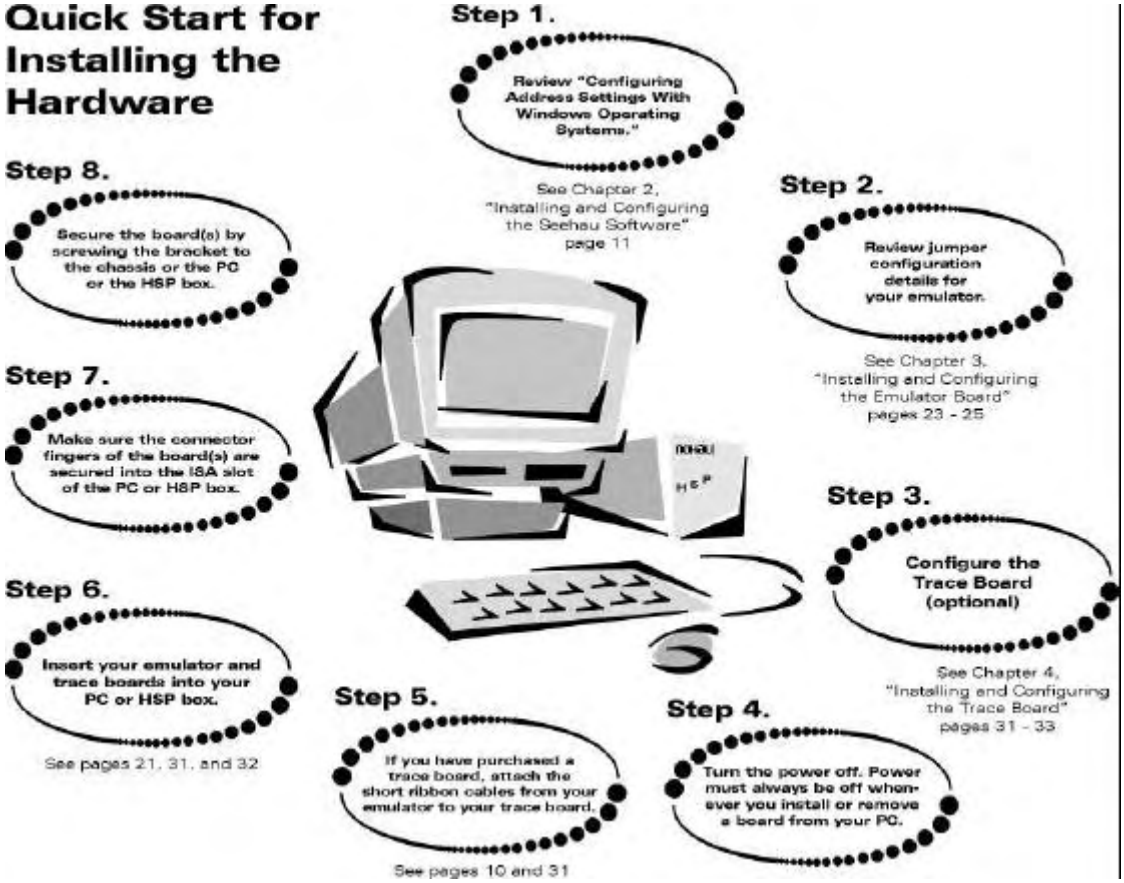


Figure 5. Steps for Installing the EMUL16/300–PC Hardware

2 Installing and Configuring the Seehau Software

Installing the Software

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.

> **Note**
>
> If the installation does not start automatically, your Windows **Autorun** feature is probably disabled. You will then need to select **Start/Run/Browse** and navigate to the CD root directory. Double-click **Autorun.exe**.

2. Follow the instructions that appear on your screen.

3. If you wish to add an icon to your desktop, use Windows Explorer to find the file Seehau.exe located in the C:/Nohau/Seehau16 subdirectory.

4. Right-click on the file, click the **Create Shortcut** option. Drag the shortcut to your desktop. Rename the icon as desired.

Configuring Seehau Software

When first started, Seehau loads a configuration file called Startup.bas. This file is created by the Seehau Configuration Program and stores Startup.bas in the following default subdirectory: C:\Nohau\Seehau16\Macro.

The Seehau software automatically starts Seehau Config if it does not find the startup file.

You do not need to have the emulator connected to the PC to run the Seehau Config Program. However, for the Seehau regular executable to operate properly, you do need the emulator connected with the jumpers properly set.

Get familiar with the emulator in stand-alone mode (not connected to a target system) or the demo mode before connecting to a target hardware system. The demo mode can be selected by doing the following: Select **Start/Programs/Seehau16/Demo**.

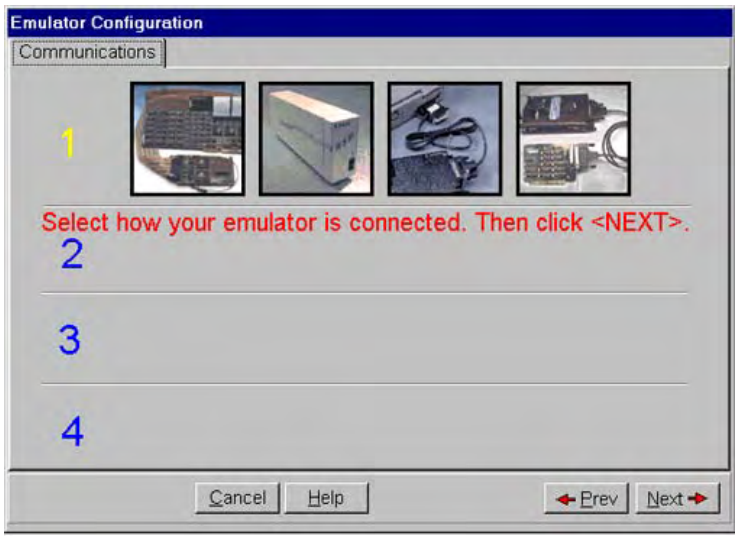


Figure 6. Emulator Configuration (Communications) Menu

WARNING

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

Purchasers of Emulator and Trace Boards

If you are purchasing the emulator board and the trace board, you might want to consider the following points:

- You will need a PC with at least two ISA (Industry Standard Architecture) slots. These slots should be close enough to allow you to connect the short ribbon cables that connect the boards or consider purchasing the HSP box.
- It will be easier to connect the short ribbon cables before installation. Waiting until the boards are already installed may result in scraped and/or bloody knuckles due to the restricted work area.
- If you purchase the trace board after the emulator board, you should consider removing the emulator board, making the ribbon connections, and then installing the boards together.

Configuring Address Settings With Windows Operating Systems

The following applies to all Windows operating systems:

- Default Address Ranges:
  - Emulator Board: 200H
  - Trace Board: 208H
- 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.

Information about Windows NT Installation

When installing under Windows NT you will be changing the registry and installing our kernel mode driver. You must do this from an account with Administrator privileges.
One of the causes of the message **Incorrect Parameter** either in the system log or from the Devices application is that there may be a device already installed with the address given for the emulator.

Known Device Driver Conflicts

We are 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 Technical 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 200H, scroll down to look for an unused address range:
   - Recommended for emulator boards are addresses 200H, 210H, and 240H.
   - Recommended for trace boards are addresses 208H, 218H, and 248H.
   - The trace board address must always be at least 8H above the emulator board (i.e., 200/208, 210/218, 240/248).
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 E, “Emulator / Trace Address Examples” to reconfigure 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.

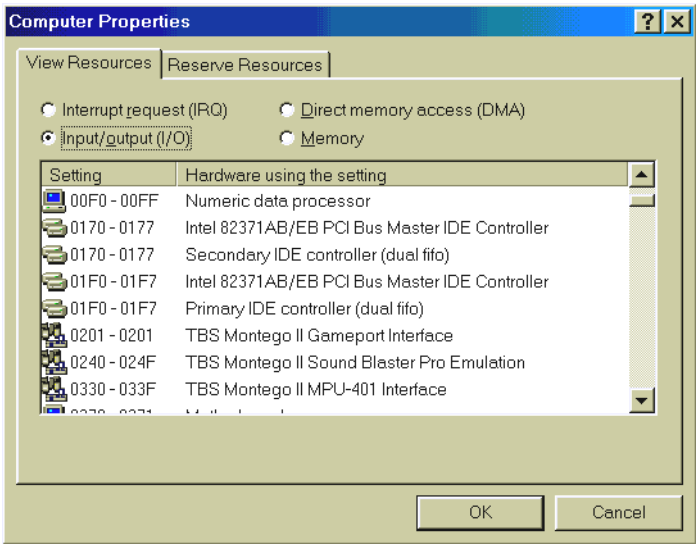


Figure 7. System I/O Resources

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 8).
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 9).

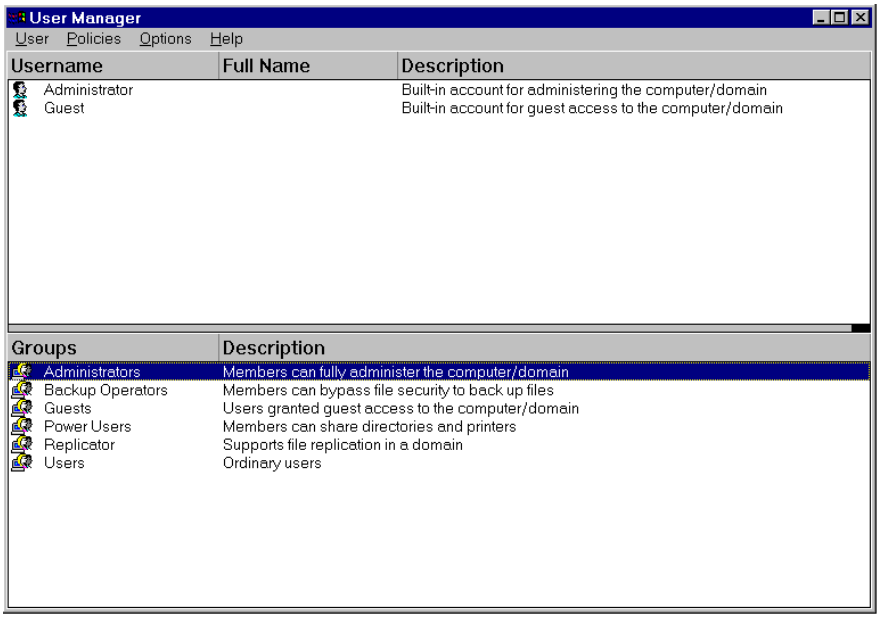


Figure 8. User Manager Dialog Box for Windows NT

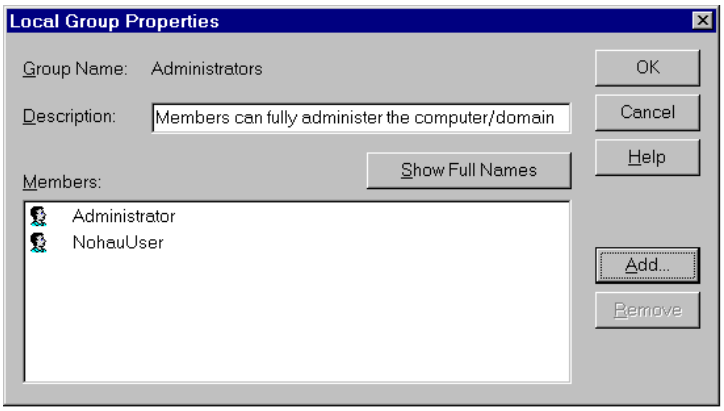


Figure 9. Local Group Properties Dialog Box for Windows NT

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 10).
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 200H, scroll down to look for an unused address range:
   - 200H, 210H, or 240H for the emulator board
   - 208H, 218H, or 248H for the trace board

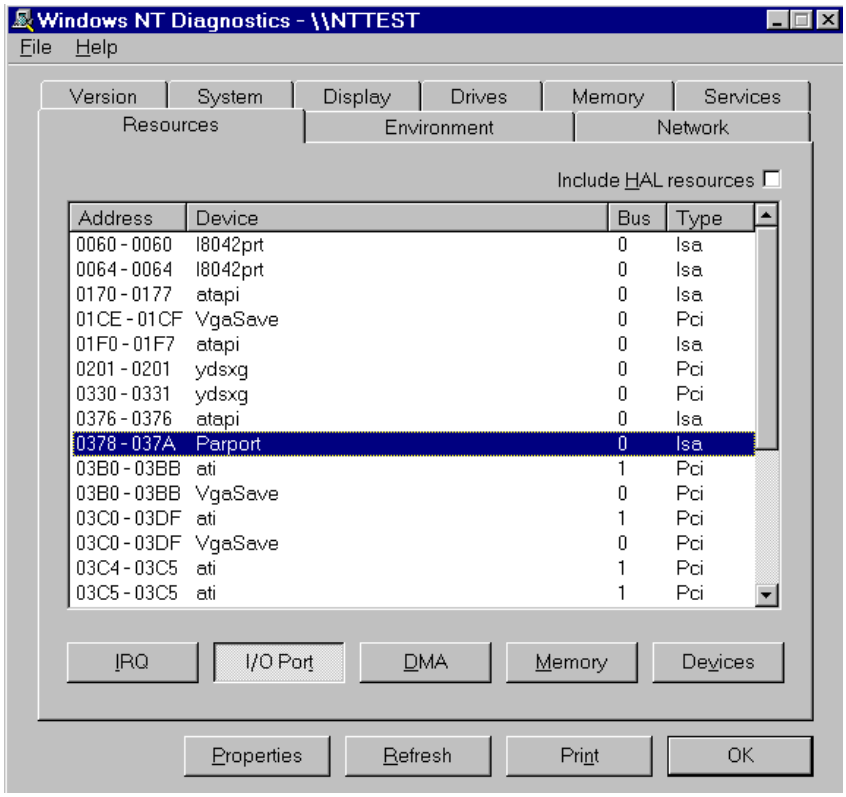


Figure 10. NT Diagnostics Window

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 Appendix D, “Troubleshooting Tips.”

- 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.

Nohau16/300 Device Driver

After installation, Windows NT Diagnostics will show the Nohau16/300 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 11), you will see three columns: Device, Status, and Startup.

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

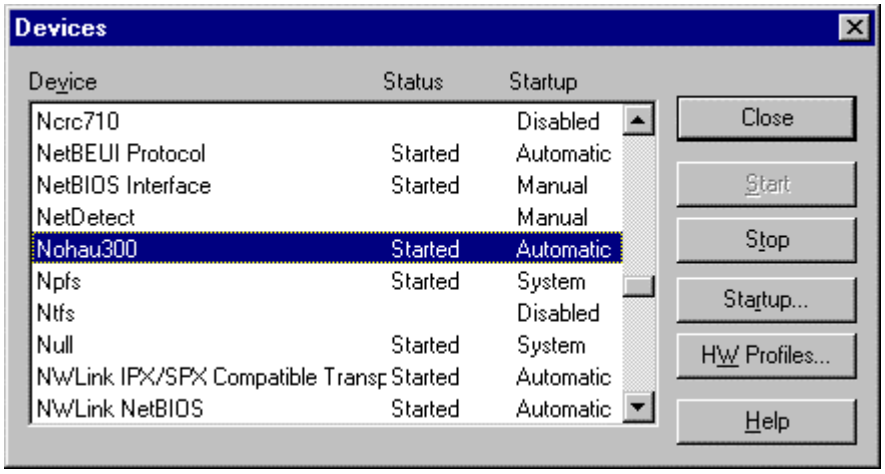


Figure 11. Control Panel Devices Window

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 12).

3. Click the **Advanced** tab. Now click the **Advanced** button. The Local Users and Groups window opens (Figure 13).

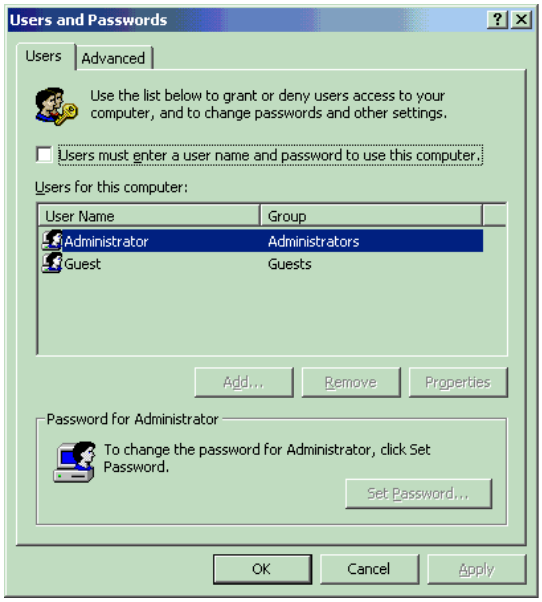


Figure 12. Users and Passwords Window

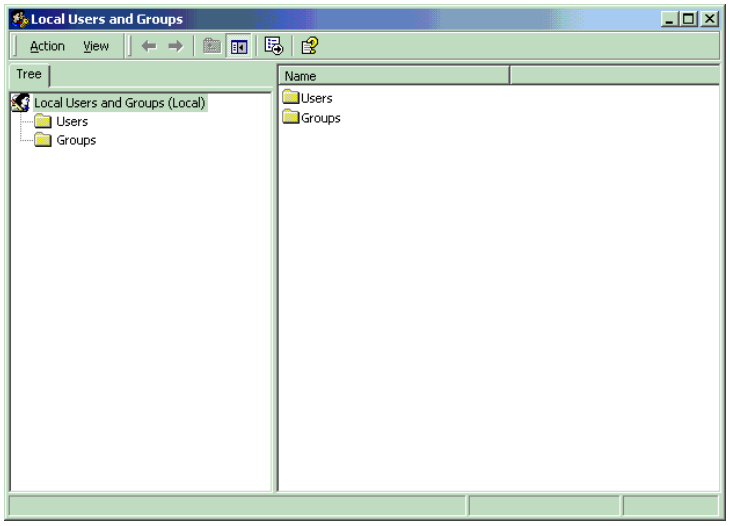


Figure 13. Local Users and Groups Window

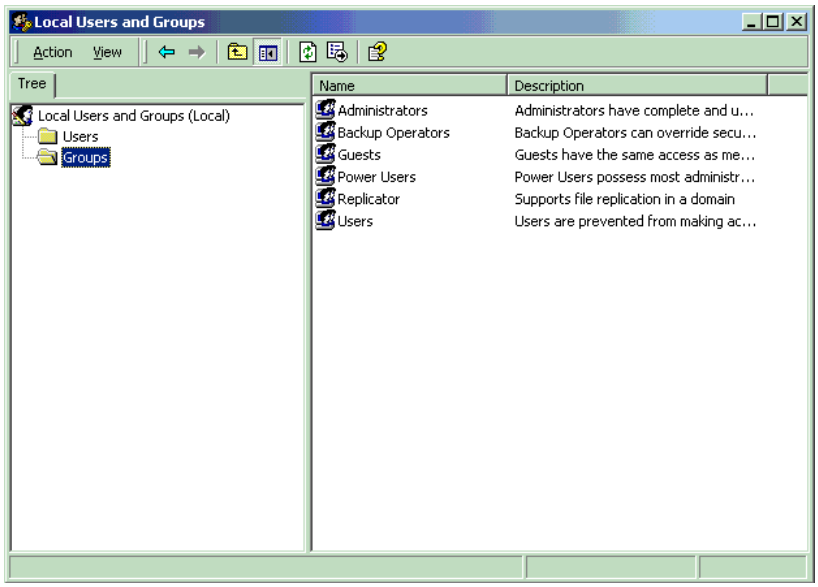


Figure 14. Local Users and Groups Window with Groups Folder

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 14).

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.

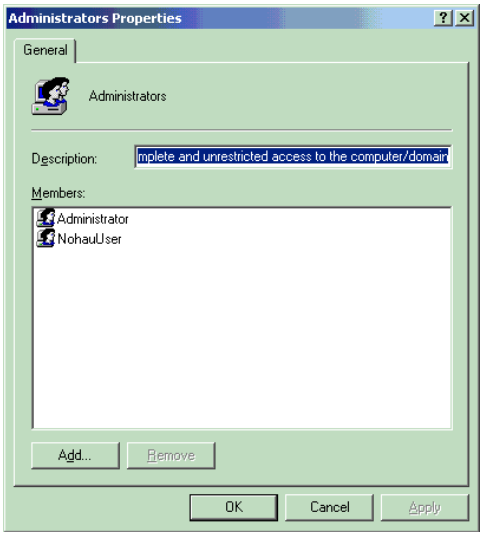


Figure 15. Administrator Dialog Box

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 16).

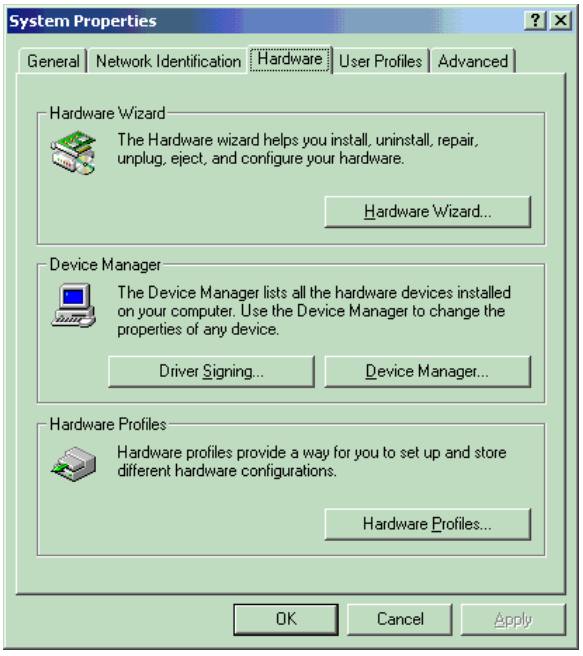


Figure 16. System Properties Window

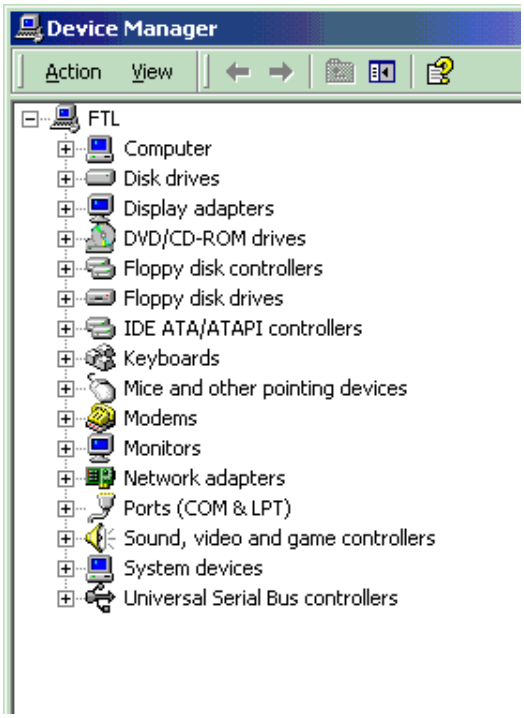


Figure 17. Device Manager Window

2. Click the **Hardware** tab. Then click **Device Manager**. The Device Manager window opens (Figure 17).

3. In the Device Manager window, select the **View** menu. Then click **Resources by Type**. A window opens that shows the system resources (Figure 18).

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.

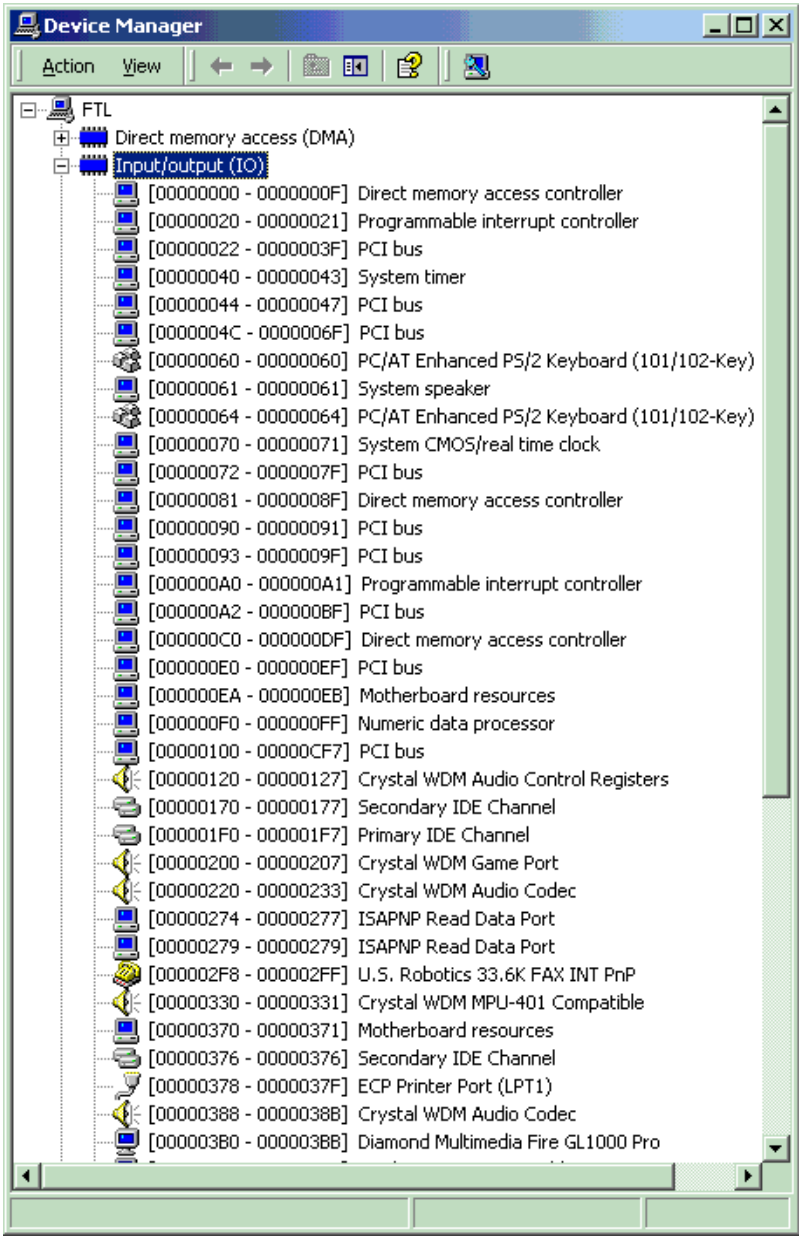


Figure 18. System Resources

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 200H, scroll down to look for an unused address range:
   - 200H, 210H, or 240H for any emulator board
   - 208H, 218H, or 248H for the 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 reconfigure the base address of your board.

Driver Troubleshooting

For details, see Appendix D, “Troubleshooting Tips.”

- 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 recheck that your port address has no conflicts.

Nohau16/300 Device Driver With Windows 2000

To verify that the Nohau16 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. Refer to the **Name** column and scroll down to **nohau16**.
5. Verify the driver is running. In the **State** column, you should see the word **Running**. In the **Status** column, you should see **OK**.

3 Installing and Configuring the Emulator Board

If you are using the ISA card inside the PC, you will need to ensure the jumpers on the board are set for the default address of 200H. If you are using the optional trace board the default address is 208H. (See Chapters 3 and 4 of this guide.)

> **Note**
>
> On some computer systems, the game port is configured for 200H. If your system is configured this way, you need to change the emulator board address to any open address (Figure 20). If the emulator address is changed, the trace board address (208H) also needs to be changed (Figure 20 and Figure 34).

If you order an HSP box, the emulator, and HSP card, (and optional trace board if purchased) are factory installed unless you order the box only. If you are using the HSP box, connect the parallel cable to the parallel port of the PC or laptop. Connect the 5-volt power supply to the HSP (there are two power supply units available, only use the short tail power supply for this). It is important that you plug the power supply into the HSP box before plugging it into an outlet.

Installing the Emulator Board

The emulator board address jumpers have been factory preset to 200H for a typical system. After you have inspected the boards for any damage and with the address jumpers in place, you can install the emulator board in your PC by performing the following steps:

1. Turn the PC power off. Power must always be off when you plug in any board.
2. Plug the emulator board edge connector into the ISA slot of your PC.
3. Insert the screw into the PC chassis to secure the board.
4. Connect the long ribbon cable to the connector on the emulator board located at the back of the PC or HSP.
5. Connect the other end of the ribbon cable from the emulator board to the pod. (Refer to Chapter 5, “Installing the Pod Boards” in this guide for specific pod setups or Appendix A though C of this guide for specific pod exceptions.)
6. Close the locks on the pod board connector over the cable end.

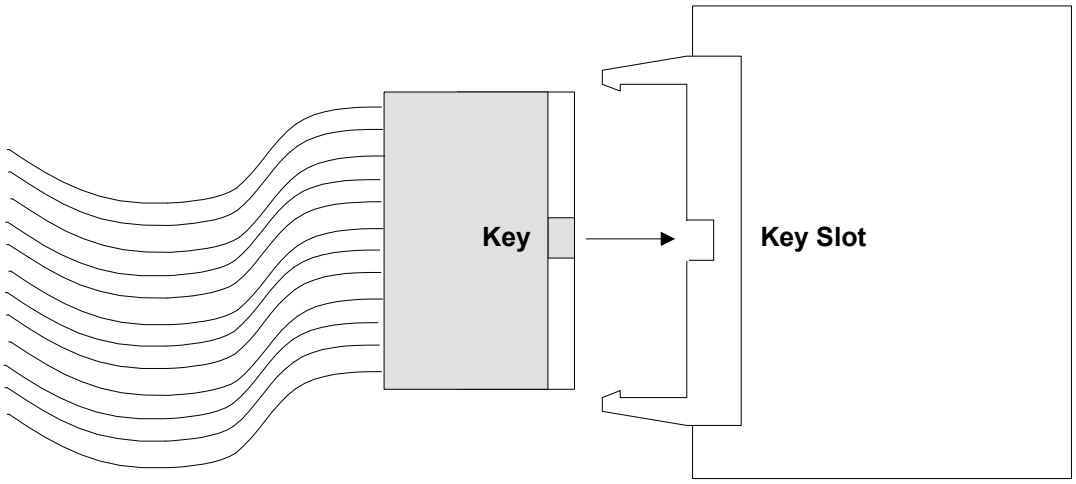


Figure 19. Connecting the Emulator to Your Pod Board With the Ribbon Cable

Note

The connectors of the ribbon cable are identical so it does not matter which end is connected to the pod or the emulator board. Although the ribbon cable connecting the emulator to the pod board is keyed, it is possible to force the key on the connector the wrong way. Caution should be used when making this connection to ensure that the key and slot line up properly.

The emulator board contains the logic and connectors necessary to communicate with the pod and to support a trace board.

I/O Address

Setting the I/O address jumpers is accomplished by moving jumpers around on the J2 header. Each pair of pins in the address header E5 represents one bit in the 10-bit address. Address bits 0, 1, and 2 represent addresses within the eight consecutive addresses, and they do not have pin pairs to represent them. This leaves six address bits (pin pairs) to set with jumpers: A3 through A9. Shorting two pins represents a 0 in the address. A pair of pins with no jumper represents a 1.

The following table shows how a typical system uses its address locations. If your system is presently using location 200H, you must find an alternate address location and make appropriate changes to the jumpers and software. If your emulator board is in an HSP box, you can use the default address regardless of the I/O address being used in the computer.

Typical PC I/O Addresses

HEX Location	Typical Use
000 - 0FF	DMA Controller
020 – 021	Interrupt Controller 1
040 – 043	Timer
060 – 064	Keyboard Controller
070 –071	CMOS RAM & NMI Mask Registers
080 – 09F	DMA Page Registers
0A0 – 0BF	Interrupt Controller 2
0C0 – 0CF	Reserved
0C0 – 0DF	DMA Controller 2
0E0 – 0EF	Reserved
1F0 – 1F8	Fixed Disk
200 – 207	Game I/O Adapter
220 –24F	Reserved
278 – 27F	Parallel Printer 2
2F0 – 2F7	Reserved
2F8 - 2FF	Serial Port 2
300 - 31F	Prototype Card
320 - 323	Fixed Disk

If the current emulator board address conflicts with any other hardware, change the address to 210H. The emulator board requires eight consecutive addresses. If you change the address and/or memory jumpers, the software address settings must also be changed to 210H.

Factory and Alternate Settings

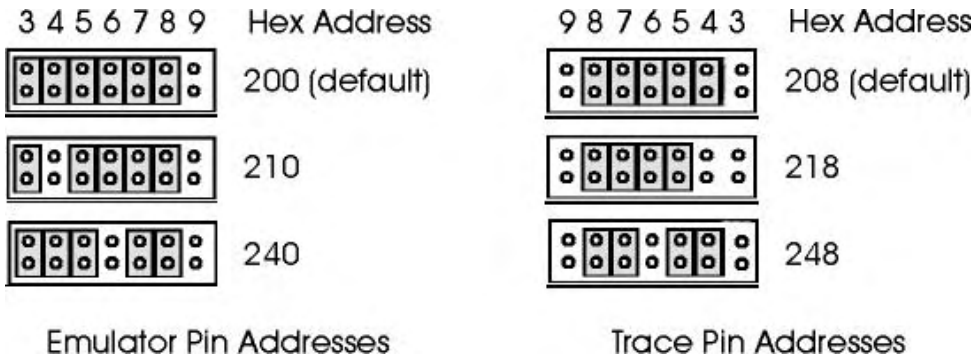


Figure 20. Emulator and Trace Pin Addressing

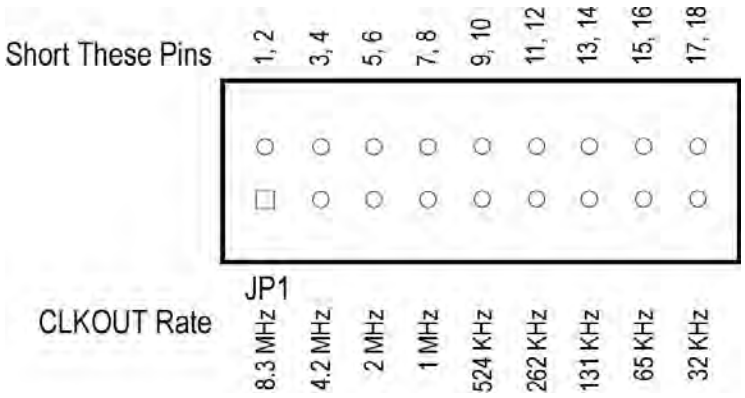


Figure 21. Header JP1

These are the only address settings that we recommend for setting the emulator and trace board. If you change the emulator hardware and software settings, you must also change the trace hardware and software settings.

Setting Target Communication Rate Using BDM

When connected directly to the target and the CLKOUT signal is available, EMUL16/300–PC can use a BDM synchronous communication rate equal to the ¼ the CLKOUT frequency rate. Changing the CPU clock rate also changes the communication rate, keeping it at the maximum allowed by the microcontroller.

If a BDM connector connects the pod to the target, the CLKOUT signal is not available and it is not possible for the emulator to follow a changing BDM communication rate. Under these circumstances, the application software must be configured to use the BDM connector and the emulator must use a fixed communication rate equal to ¼ the slowest clock rate used on the target.

To set the synchronous communication rate, look up the slowest processor clock rate set by the application. To set the appropriate speed you will need to short a pair of jumpers. The pins are not labeled but they are two rows parallel to each other to the left of header J2. The JP1 pins are numbered left to right. The top row is even numbered 2 through 18 and the bottom row 1 through 17. The set speed for each pair of pins is shown in Figure 21.

Note

Header JP1 is only used when the software is configured to communicate via BERG connector. When configured for direct communications JP1 is ignored.

Factory Settings

Board jumpers have been preset at the factory prior to shipment.

Emulator Board

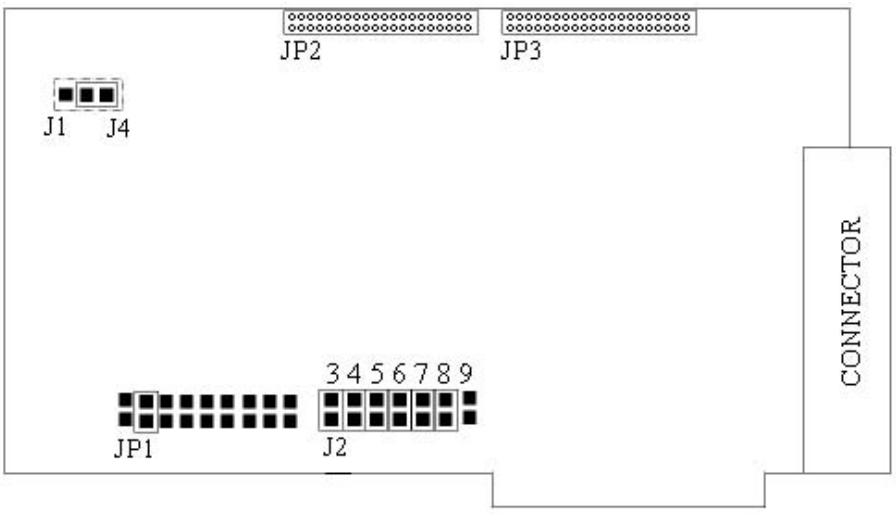


Figure 22. Emulator Board

Emulator Board Description

Jumper Designation	Description
J1	Applies to emulator boards with 1 megabyte of Shadow RAM. With less than 1 megabyte of RAM, the jumper must be, set on pins 2 & 3. • If pin A19/CS6 is an address pin short pins 1 & 2.
J2 (board I/O address	• Each pair of pins represents 1 bit in 10-bit address. • Shorting pins represents a zero, no jumper on pins represents a 1. • Least significant bit on the left.
J4	Applies to emulator boards with 1 megabyte of Shadow RAM. With less than 1 megabyte of RAM the jumper must be, set on pins 2 & 3. • If pin A19/CS6 carries chip-select signal and < 1meg of Shadow RAM short pins 2 & 3.
JP1	Default value is 4.2 MHz. Allows the CPU clock rate to be changed. • Can be adjusted from 32 kHz to 8.3 MHz. • When CLKOUT signal available allows BDM sync communication rate = to ¼CLKOUT frequency rate. • Changing the CPU clock rate changes communication rate, keeping it at maximum allowed by microcontroller.
JP2 (short ribbon cable connection)	• Connects to the J2 on the trace board and allows tracing while development takes place.
JP3 (short ribbon cable connection)	• Connects to JP5 on the trace board and allows tracing while development takes place.

Configuring the Emulator

1. Click the Seehau icon on your desktop. You do not need the emulator connected at this time.

2. Enter the correct settings as shown in the **Communications Configuration** dialog box.

   Figure 23 shows the settings if you are using the ISA card inside the PC. The **Emulator Board Address** box contains the address of the internal communication link from your computer. For the ISA (PC-Plug-In) card, the most common address is 200, which may cause a conflict with some game ports. If you find that there is a conflict with this address uncheck the default box under the **Emulator Board Address** and change it to 210. If an optional trace board is purchased the Step 4 **What is your Trace Type?** address will be 208 (218 if the emulator was changed to 210). If no trace board is used then Step 4 will be **None**.

   Figure 24 shows the settings used for the HSP box (the ISA card and the HSP configuration setup are almost identical). A double-headed DB-25 male/female plugs into your PC parallel port. A DB-15 male connector on the other end connects to the HSP card installed in the HSP box. The DB-25 female portion of the plug end allows you to maintain your printer connection to your parallel port. If no trace board is used then Step 4 will be **None**.

   Figure 25 shows the settings if you are using the EPC cable attached to a laptop computer (the configuration is almost identical to the LC-ISA). The **Select Processor** box will contain the BDM type rather than a processor type. Since there is no trace capability for the BDM, Step 4 will default to **None**.

   Figure 26 shows the settings if you are using the LC-ISA board inside the PC. The **Select Processor** box will contain the BDM type rather than a processor type. For the LC-ISA card, the most common address is 200. If you find that there is a conflict with this address clear the default check box under the **Emulator Board Address** and change it to 210. Since there is no trace capability for the BDM, Step 4 will default to **None**.

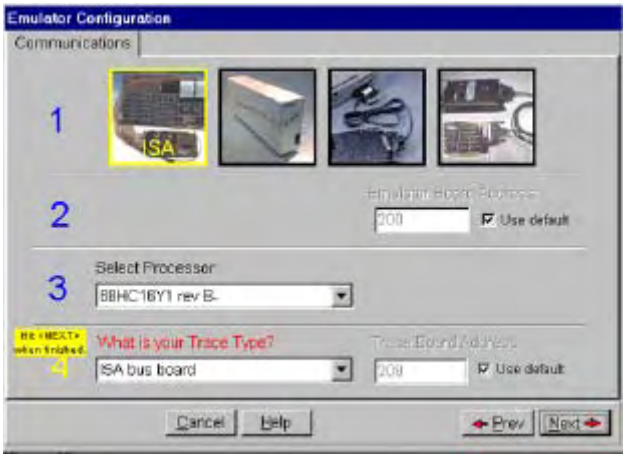


Figure 23. ISA Emulator Configuration

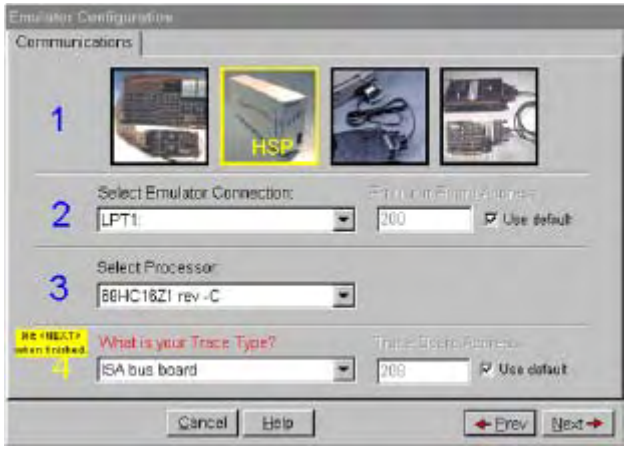


Figure 24. HSP Emulator Configuration

Figure 25. EPC Emulator Configuration (BDM)

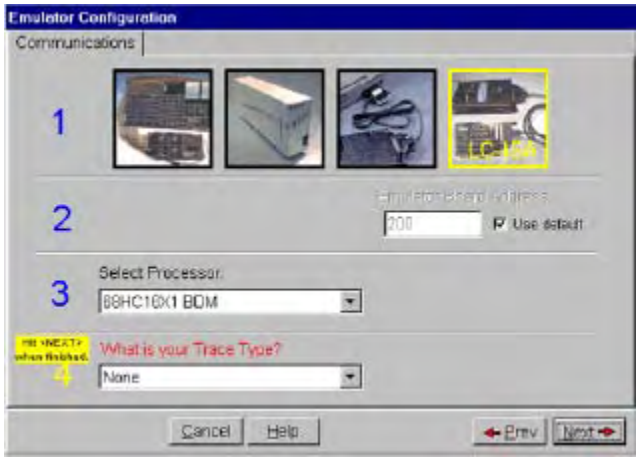


Figure 26. LC-ISA Emulator Configuration (BDM)

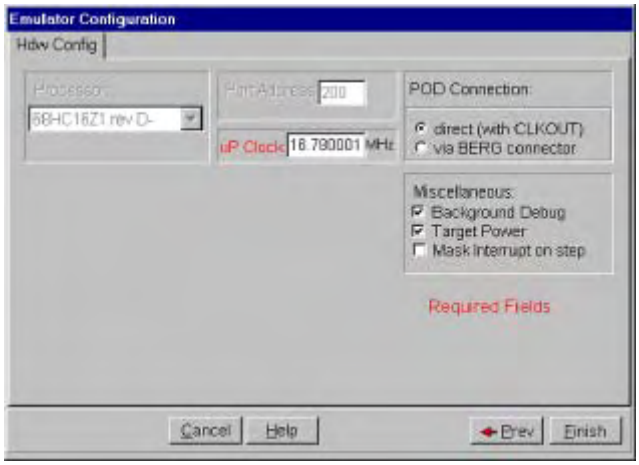


Figure 27. Full Emulator Hardware Configuration

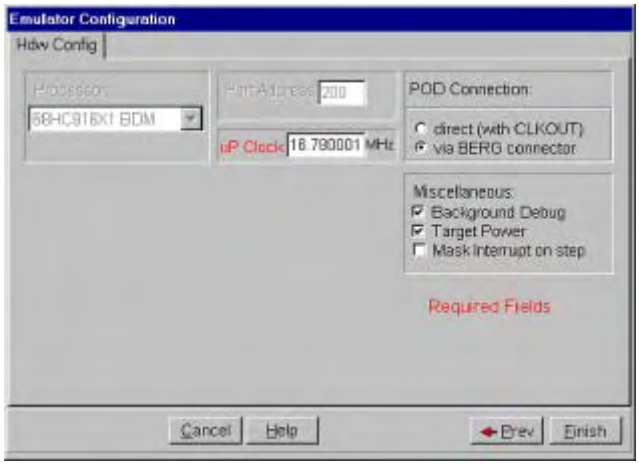


Figure 28. BDM Emulator Hardware Configuration

In the **Select Processor** list, click the down arrow and select the microprocessor/BDM that you are going to emulate. Make a selection that matches your pod/BDM board type:

- For 68HC16Y1/(9)16Y1, see Appendix A in this guide.
- For 68HC16Y3/(9)16Y3, see Appendix B in this guide.
- For 68HC16X1/(9)16X1, see Appendix C in this guide.

If you are uncertain about which microprocessor to select, refer to Chapter 5, “Installing Pod Boards” in this guide or click **Help** in the dialog box.

3. In the **Trace Type** list, click the down arrow. Two options are available for the ISA plug-in and HSP box, select **ISA bus board** or **None** depending upon whether you purchased an optional trace board. The EPC and LC-ISA default to **None** as there is no trace capability.

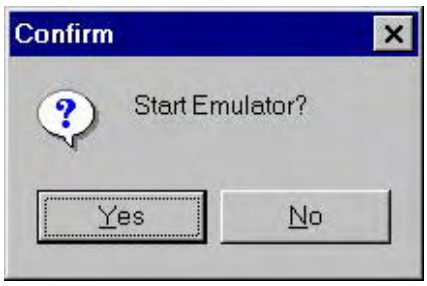


Figure 29. Confirm Dialog Box

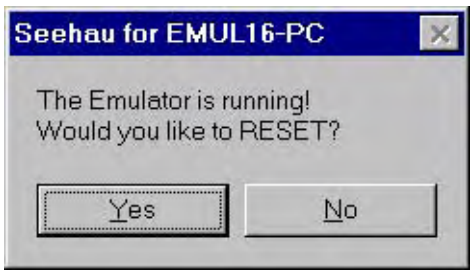


Figure 30. Reset Emulator Dialog Box

4. Click **Next** to open the dialog box shown in Figure 27 and Figure 28. These two figures show the hardware configuration difference between a full emulator (Figure 27) and a BDM emulator (Figure 28). The only difference (other than the processor) is the **POD Connection**, which is direct (with CLKOUT) with a full emulator and via BERG connector with a BDM emulator.

5. The **uP Clock** is the internal CPU clock. This setting is used only for the calculation of the trace timestamp. It has no effect on the operating speed of the emulation controller. The time entered here should be the internal processor speed (not necessarily the crystal speed).

6. After making your choices select **Finish**. The **Emulator Configuration** dialog box will close.

7. The Seehau configuration program creates the Startup.bas file. Seehau is now configured to run your emulator. The **Confirm** dialog box will open and ask whether you would like to **Start Emulator?** (Figure 29).

8. Reply **Yes** if you would like to start the Seehau software. (Make sure that your emulator equipment is powered on at this time or you will receive an error message and will need to restart your Seehau software.)

9. If you have replied **Yes** you may receive another dialog box (Figure 30) asking **The Emulator is running! Would you like to RESET?** Reply **Yes** if you wish to start your emulator.

If you have completed these steps with no errors, you are ready to run the Seehau User Interface and the Seehau software will start.

At this point you can expand the initial menu to a full screen mode, rearrange and expand the interior windows and proceed to configure the Seehau software.

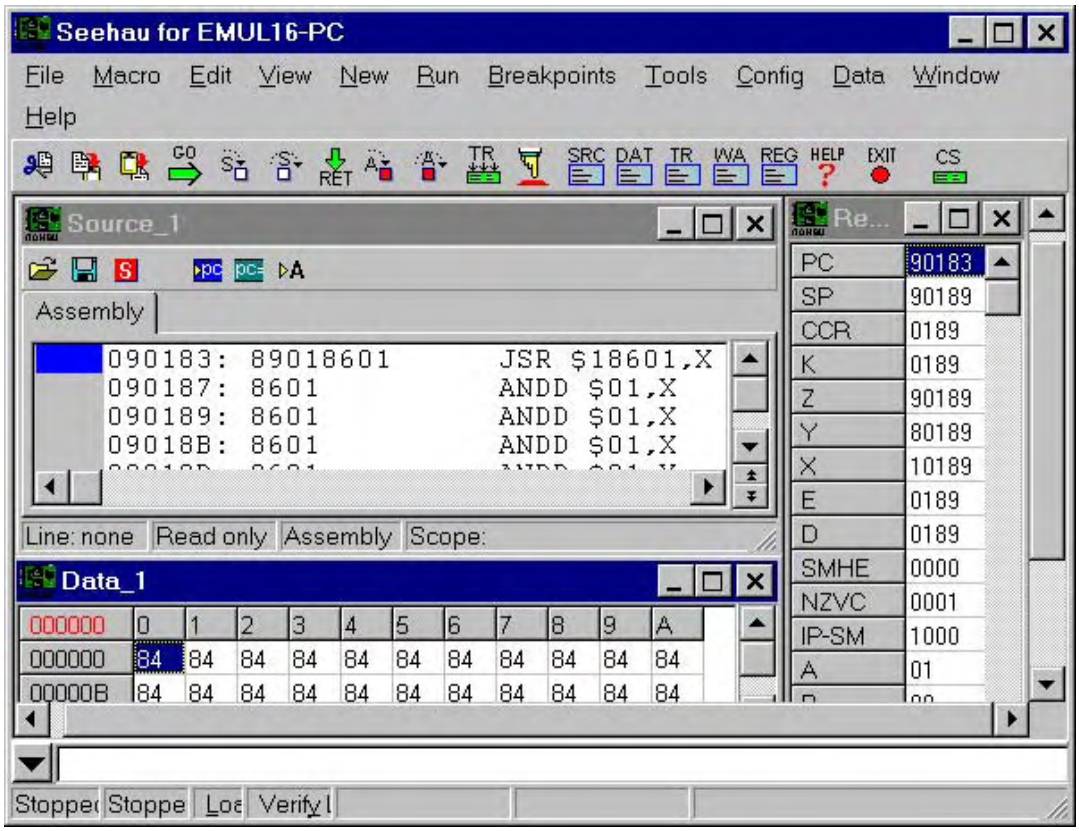


Figure 31. Seehau Software Menu

Quick-Save Hardware Settings

Due to the instability of PCs and operating systems, it is important to take precautions after setting up your hardware and software. Rather than wait until you have finished doing your tests on the target system you may want to save the emulator settings to avoid unnecessary repetition in case of system failure. The quick and easy way to avoid this problem is to follow the procedure below:

1. To save the emulator configuration, click the **Config** option and select **Environment**.

2. From the **Environment Configuration** menu (Figure 32), check the **Use Start up Dialog?** under the **Preferences** tab. This check box will be found under the **Miscellaneous** section.

3. Select **Apply** or **OK**. The **Environment Configuration** dialog box will close.

4. Exit from the Seehau software.

5. The **Save Settings** dialog box opens where you can choose the filename for the newly created macro. Enter a filename of your choosing and click **Save**.

The macro is ready to use and will accurately recreate your emulator configuration settings.

WARNING

The target power must never be on when the pod is powered off. To avoid damage, power the pod and target on and off in the following sequence. To power up: (1) Power on the pod, then (2) Power on the target. To power down: (1) Power off the target, then (2) power off the pod.

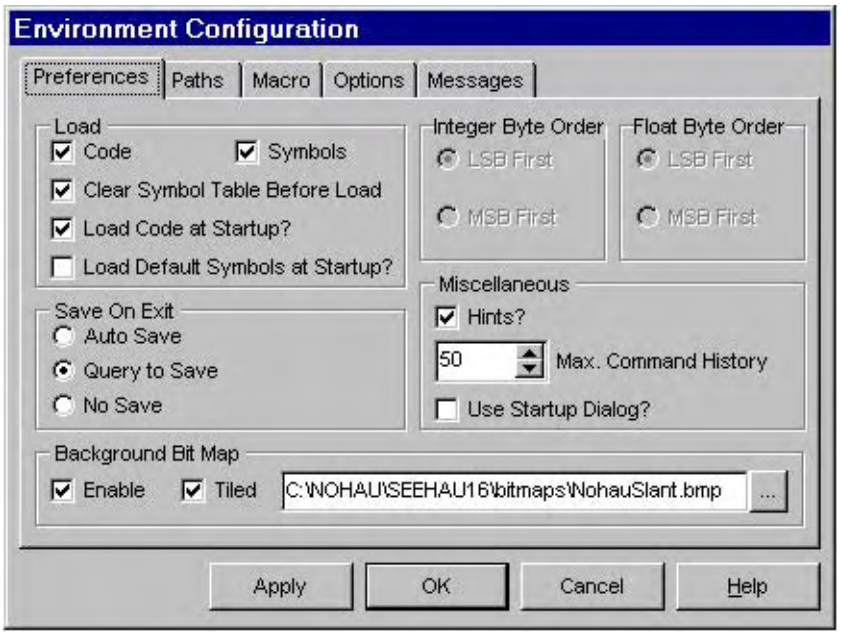


Figure 32. Environment Configuration Menu

4 Installing and Configuring the Trace Board

Overview

The EMUL16/300–PC needs RAM to record a history of the data used and instructions executed. The trace board contains this RAM. 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 board to the trace board. The 104-bit trace records include address, data, timestamp, processor status and user defined external data. This trace board can trigger on or filter fetch, read or write addresses, (not data values.) The standard trace boards for the 68HC16/916 and 683XX families have a 32K deep trace buffer at 16.78 MHz. Only this frequency is available.

Installing the Trace Board

The trace board address jumpers have been factory preset to 208H. If the trace board and emulator boards are purchased for installation in a PC, it might be easier to attach the cables before installation. If purchased later you might consider removing the emulator board, attaching the short ribbon cables and then reinstalling both boards. A bit of caution should be used when installing these boards as they may result in scraped and bleeding knuckles caused by projecting pins.

Note to PC-ISA Users

If you found it necessary to change the emulator board address from 200H to any other address (210H and 240H recommended) then it will also be necessary to change the trace board address (218H and 248H recommended). The trace board is preset to 208H and will require that you change the hex address accordingly (plus 8H to your emulator address). Always change the trace board according to this pattern, i.e. 200H/208H, 210H/218H, 240H/248H.

After you have inspected the boards for any damage, with the address jumpers in the correct place you can install the trace board in your PC or HSP following these steps:

1. Turn the power off. Power must always be off when you plug in any board.

2. Connect the short ribbon cables.

3. If you are installing the trace board in a PC, insert the trace board into an ISA slot next to the emulator board (also in an ISA slot.) You will need to ensure that the ribbon cables will reach from one board to the other.

4. Check to ensure the edge connector of the board is secured into the expansion slot of the PC or HSP motherboard (only necessary in the HSP if it was purchased separately from the emulator and trace boards.)

5. If you have trouble seating the boards, ensure that the metal tab end of the board is positioned correctly and not hanging up on the PC chassis or motherboard. Once seated, screw the board down to the chassis for a secure connection.

I/O Address

> **Note**
>
> If you purchased the HSP box, skip this chapter and refer to Chapter 9, "Trace Memory Example" in this guide.

Each pair of pins in the address represents one bit in the 10-bit address. Address bits 0, 1, and 2 represent addresses within the 16 consecutive addresses, and they do not have pin pairs to represent them. This leaves seven address bits (pin pairs) to set with jumpers: A3 through A9. (A3 is on the right and A9 is on the left, which is just the opposite of the emulator board.) Shorting two pins represents a 0 in the address. A pair of pins with no jumper represents a 1.

The trace board address jumpers have been factory preset to 208H. If your system is presently using location 208H, you must find an alternate address location and make appropriate changes to the jumpers and software. If you changed the address of your emulator board, you must also change the address for the trace board. If your trace board is in an external HSP box, you can use the default address regardless of I/O address being used in the computer.

The trace board requires 16 (or 10H) consecutive addresses. If you change the address and/or memory jumpers (hardware), the software settings must also be changed.

Installing the Trace Board With an HSP Box

If you purchased the HSP box, you can use the default address (208H) regardless of the I/O addresses used by the computer. Skip to the "Steps for Installing the Trace Board" section.

Installing the Trace Board With a PC

The trace board address jumpers have been factory preset to 208H. If your system currently uses location 208H, you must find an alternate address location and make appropriate changes to the jumpers and software. See previous sections about "Alternative Addressing" for the different Microsoft operating systems.

Factory Settings

On the trace board, the locations of jumper pins (header J1) are designated on the board artwork as A9 through A3 (left to right) which is just the opposite of the emulator board (A3 through A9). These pins are located to the left of the BNC connectors (connectors facing to the right). The recommended settings are as follows:

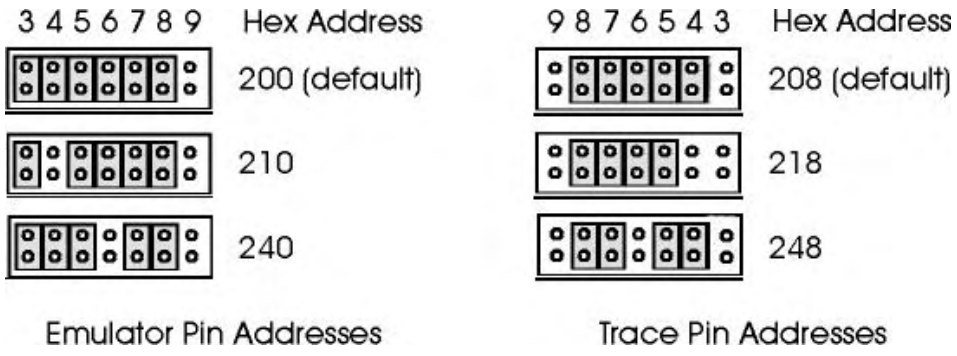


Figure 33. Emulator and Trace Pin Addressing

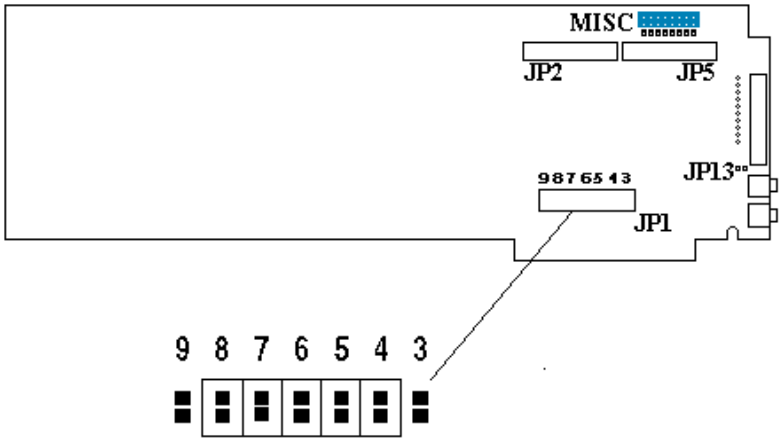


Figure 34. Trace Board With Address 208H

These are the only address settings that we recommend for setting the emulator and trace board. If you change the emulator hardware and software settings, you must also change the trace hardware and software settings.

Bank Switch Jumpers

Jumper Designation	Description
JP1	• Allows the trace board address to be changed by moving jumpers from one set of pins to another. • Least significant bit on right. Pins labeled A9 to A3 left to right. • Recommended settings are 208H, 218H, and 248H.
JP2 / JP5	• Used to connect the short ribbon cables from the emulator board to the trace board. JP2 to JP2 and JP3 to JP5.
JP13	• Allows you to change the clock speed while using the BDM emulators.
MISC	• Three rows of 8 pins (7 through 0, left to right, A, B, C top to bottom). Default pins jumpered are columns 7 through 0, rows A and B.

Configuring the Trace Board

The trace board records what the microprocessor is doing. However, if everything is recorded, you can end up with data of no interest to you. To prevent this, the Seehau software lets you set up the trace board to record only the information you want. By default, all frames are collected and no frames cause a trigger.

The trace board records eight external digital inputs with every bus cycle. These signals are input through a DB-25 connector (Figure 35) on the back of the trace board. Trace boards have two additional input/outputs on the back called BNC connectors for **TRIGGER_IN** and **TRIGGER_OUT** recording.

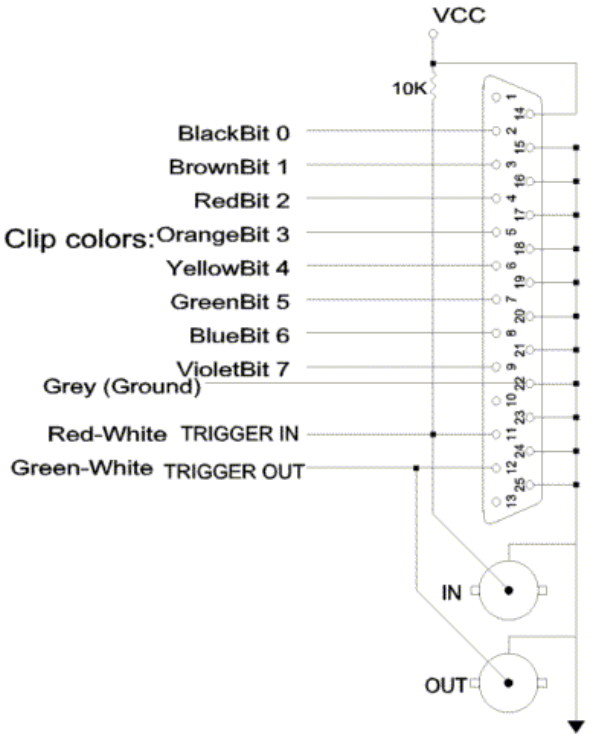


Figure 35. DB-25 Connector Pin Definitions

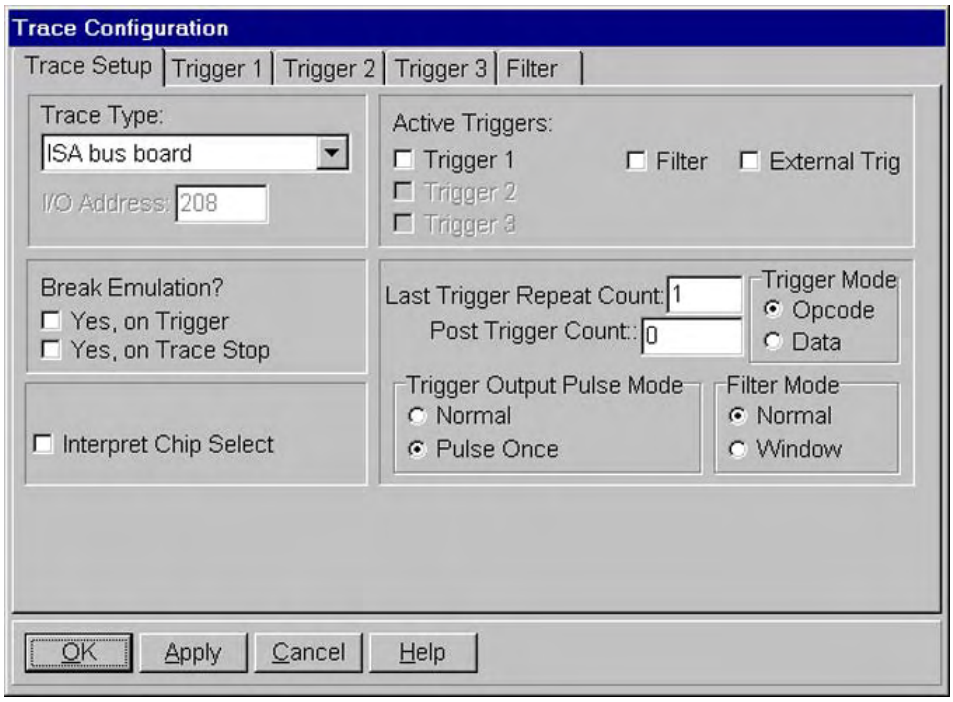


Figure 36. Trace Configuration Dialog Box

From the **Config** menu, select **Trace**. The **Trace Configuration** dialog box opens (Figure 36). The **Trace Configuration** dialog box has a **Trace Setup** tab, three **Trigger** tabs and a **Filter** tab.

The **Trace Setup** tab allows you to configure the different functions of the Trigger and Filter.

In the **Trigger** tabs selection (Figure 37), you will find the **Address Cycle Type** and **Data Trigger Type** beginning and ending values. These are lists of values to look for in each frame as it is collected. For each machine cycle executed, the trace board compares the current frame with the conditions in **Address Cycle Type** list and the **Data Trigger Type** list. These comparisons produce Boolean results: either a *true* or a *false*.

In the **Filter** tab selection (Figure 38), you will find the **Address Cycle Type** and **Data Trigger Type** beginning and ending values. These are lists of values to look for in each frame as it is collected. For each machine cycle executed, the trace board compares the current frame with the conditions in **Address Cycle Type** list and the **Data Trigger Type** list. These comparisons produce Boolean results: either a *true* or a *false*.

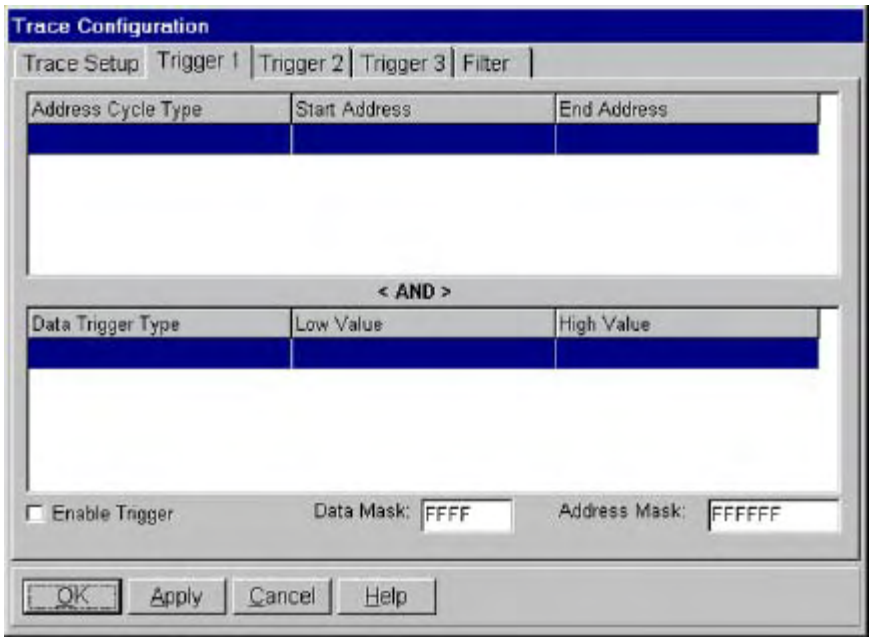


Figure 37. Trace Trigger 1-3 Dialog Box

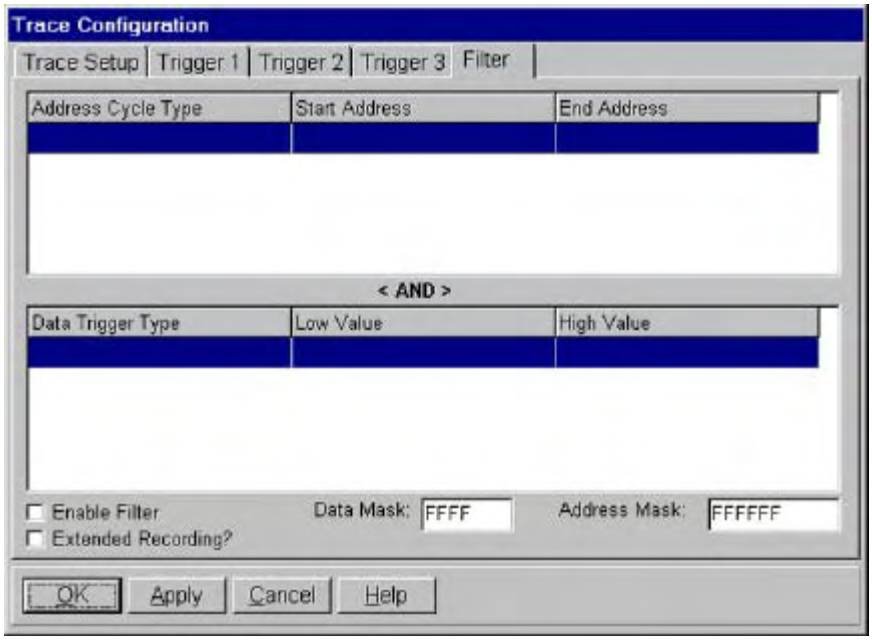


Figure 38. Trace Filter Configuration Dialog Box

Trace Setup Fields

Trace Type

- Select **ISA Bus Board** if you have a trace board.
- Select **None** if you have no trace board installed and skip the rest of this chapter.

Break Emulation

Yes, On Trigger—Stops trace when certain conditions happen.

Yes, On Trace Stop—Stops trace conditionally.

Active Triggers

Trigger1,2,3—Selectively chooses your active triggers to set your trigger conditions.

Filter—Selectively chooses after which triggered pass to start your trace capture.

External Trig—Enables a trigger inhibit input. If the external line is low then the trigger will be inactive. Once released the trigger is left enabled.

Trigger Mode

Opcode—Enables the operation for selective triggering mode: OF, R/W, Include All and Exclude All.

Data—Modes are Include All, Data Read, Write and Exclude All.

Trigger Output Pulse Mode

Normal—Output on Trigger Out is pulsed every time the trigger is qualified.

Pulse Once—Output is pulsed only on an initial trigger being met.

Filter Mode

Normal—More control is given to triggers. A trigger in Normal mode either stops recording, starts recording, or starts the countdown until recording will be stopped.

Window—You can selectively record program threads in a way that record filtering cannot. Once recording has started, the conditions in the filter decide whether to record a bus cycle or not. More control is given to controlling which bus cycles are recorded.

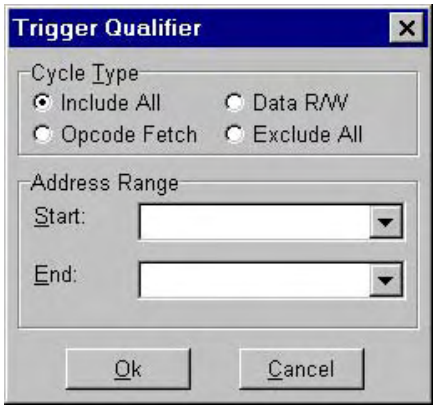


Figure 39. Edit Trigger Qualifier Window

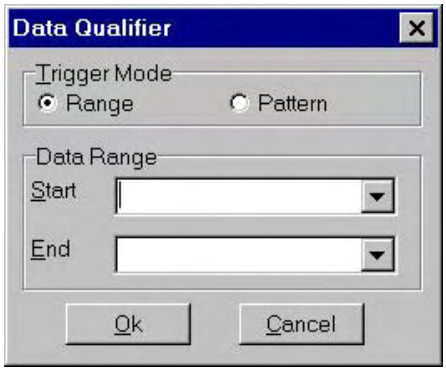


Figure 40. Edit Data Qualifier Window

Miscellaneous Parameters

Interpret Chip Select—Available address signals with base addresses are combined to calculate an address that matches symbol norms and power values.

Last Trigger Repeat Count—Number of times the highest numbered trigger conditions must be met before the trigger itself occurs.

Post Trigger Count—The number of frames recorded after the trigger occurs.

Trigger Conditions

Adjusts the amount of PRE/POST trigger you want to see. The default is set to the halfway mark.

Delay—Shows the number of post trigger samples.

Condition—Allows you to select the function you want to trigger the trace capture.

The **Trace Configuration** dialog box (Figure 38) refers to the conditional setup summary for triggers and/or filters. To edit a condition, click on the appropriate tab and then double-click an appropriate condition or right-click that condition and enter the appropriate data. The Edit Trigger Qualifier window appears (Figure 39) for **Address Cycle Type** and the Edit Data Qualifier window (Figure 40) appears for **Data Trigger Type**.

Trigger Qualifier

Cycle Type

Include All—Include all cycles.

Opcode Fetch—Include the opcode fetch of an instruction.

Data R/W—Include specific Data Read or Write.

Exclude All—Include none.

Address Range

Use the following syntax for the **Address** field:

Start Address—Enter the starting address in hexadecimal.

End Address—Enter the ending address in hexadecimal.

Data Qualifier

Target Mode

Range—Use this range of data.

Pattern—A singularly defines address rather than a range of addresses.

Data Range

Start Address—Starting value of data desired.

End Address—Ending value of data desired.

5 Installing the Pod Boards

Overview

The EMUL16/300-PC pod boards contain an MCU with a 4.192-MHz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod boards also include a Motorola standard Background Debug connector (BERG) connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Pod Types

68HC(9) 16x Pods	
• 16S2	• 16Y3
• 16V1	• (9)16Y3
• 16X1	• CM16Z1
• 16Y1	• 16Z1/Z2
6873XX Pods	
• 331	• 336
• 332	• 338
• F333	• 340
• 334	• 341
• 335	• 376

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.

Connecting the Pod to the Target Board

There are a number of ways to connect the pod to the target system. Each method has advantages and disadvantages. The pod section will explain in detail how these connections are made and the devices that are used to make these connections.

How this Chapter is Organized

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

- **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.

- **Configuration Options**: Describes the pin setting in stand-alone mode and as the pod is shipped from the factory.

- **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.

Factory Configuration of Pod Boards

When shipped from the factory, all headers have jumpers located for stand-alone operation (without target). When you connect a pod board to a target be sure to examine all jumpers and make sure that they are all correctly placed. Use the description for your pod as a guide to jumper placement.

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.

Do not discard the conducting foam if you plan to store it later. If you remove the pod from your target socket or target adapter, you will need to reinstall the conducting foam to protect the exposed pins.

Features Common to All Pods

Background Debug Mode (BDM)

All the pod boards supported by EMUL16/300–PC include one very important design feature that aids emulation. This is the Background Debug (BDM) Mode. BDM is an alternate operating mode that suspends normal instruction execution and uses special micro-code to communicate with the emulator. Once in BDM, EMUL16/300–PC communicates with the MCU through a dedicated synchronous serial line that runs at speeds up to 4 Megabits per second. This serial communication module (separate from any application serial ports) sends and receives debugging commands such as read data, write registers, read program data, etc. All this communication is completely transparent to the user.

In BDM, the MCU suspends all normal activity and asserts the FREEZE signal. Most modules, including the General-Purpose Timer module, can choose whether to respond to the FREEZE signal.

Background Debug Connector (BERG)

The 10-pin connector near the corner opposite the ribbon connector is the Background Debug (BERG) connector. This standard connector defined by Motorola allows the EMUL16/300–PC to debug and control target boards (with a similar connector) with no physical room for the pod board. However, when using just the BERG connector, no emulation RAM is available to the target and EMUL16/300–PC cannot control the target as completely as when the pod is directly connected the target. Without a direct connection between the pod and target, shadowing RAM and maintaining the trace buffer are both impossible. Also, without a direct connection, the communication rate between EMUL16/300–PC and the target must be set to 1/4 of the slowest clock rate. With a direct connection, Seehau sets the communication rate automatically, even when the application changes clock rate.

Use the following table to confirm that the pin assignment of the BERG connector on the target agrees with the pin assignments on the EMUL16/300–PC pod board.

Signal	Pin #	Pin #	Signal
DS	1	2	BERR
GND	3	4	BKPT
GND	5	6	FREEZE
RESET	7	8	IPIPE1 / DSI
V_{CC}	9	10	IPIPE1 / DSO

Using Just the BERG Connector

Just as you must remove any duplicated resources with the pod connected to the target, when using the BERG connector, you must remove the duplicated resources. This really means the PWR jumper and the controller in the pod. Removing the controller disables all the other resources (clock, memory, serial port, etc.) on the pod.

When using a BERG connector, be sure to configure Seehau correctly. Use the **Emulator board** item in the **Config** menu to open the **Hardware Configuration** dialog box. In the **POD connection** box, click on the **via BERG connector** option. Also, be sure to set the **Target Communication** rate with the emulator board header JP1.

Indicator Lights

Most pod boards contain four lights except for a few exceptions. The EPC pods contain only three lights. The pods with four lights are labeled HALT, RESET, FREEZE, and RUN. The HALT, RESET, and FREEZE lights display the condition of their respective MCU pins.

The RUN light shines when the MCU asserts either AS or DS. This occurs when the MCU is executing an external bus cycle or when the MCU is executing an internal bus cycle and either of the SHEN bits is set in the Module Configuration Register.

Note

If the MCU is executing internal bus cycles and the SHEN bits are not set, the RUN light will not be lit.

The RESET light is lit when any device pulls the RESET pin low. This may be the emulator, the MCU's watchdog timer, or an external reset circuit. If the RESET light stays lit, that usually means that the MCU is having trouble synchronizing to the clock circuit.

The FREEZE light means that the MCU has reached BDM and is waiting for the emulator to give it instructions.

Disabling Resources

To Disconnect:

- MCU: Remove it from pod.
- RAM: Remove the RAM header jumper.
- Crystal: Remove all three jumpers next to the crystal.
- Serial Port: Remove the jumper in the SER header.
- Power Supply: Remove the jumper from the PWR header.

Removing and Installing the Controller

The best way to remove the controller from the pod is to use the AMP extraction tool (part number 821-958-2) and to follow the instructions that come with it.

If an extraction tool is not available, use a small flat head screwdriver (not a Phillips). Place the blade in one corner between the socket and the carrier. Using tweezers or needle nose pliers, gently squeeze the socket flaps at the corner with the screwdriver. If the carrier does not rise enough the keep the flaps from moving back into place, gently twist the screwdriver to pry the carrier apart from the socket. Do the same for the other three corners. If a large twisting force is required, then do not continue. Instead, try again on another corner.

At this point, the carrier should be loose and you should be able to lift the carrier off the socket with your fingers. If not, repeat the above steps until the carrier is loose.

Before insertion, make sure that none of the pins are bent and that they all sit precisely in their respective channels in the carrier. After insertion, check that all four of the socket’s ears have locked the carrier to the socket equally well. Use a magnifying glass, if necessary.

The black wire with the micro-clip is a ground wire, which is helpful for ensuring that the pod and target grounds are at the same potential before connecting the pod to the target.

Hint

ISO-160 can be used to isolate any signal on the target from the pod. This may be simpler than modifying the target board.

WARNING

When using the pod with a target, disable all pod resources that are duplicated on the target. Failure to disable the pod's resources may damage the pod or the target or both. This includes the MCU, the serial port, RAM, crystal, and, particularly, the power supply. If using the clip to attach to the target, remove the MCU from the pod.

When installing a controller into a pod, never press on the chip body. Press only on the carrier or cover. Pressing on the chip may bend pins and cause short circuits.

POD–16S2

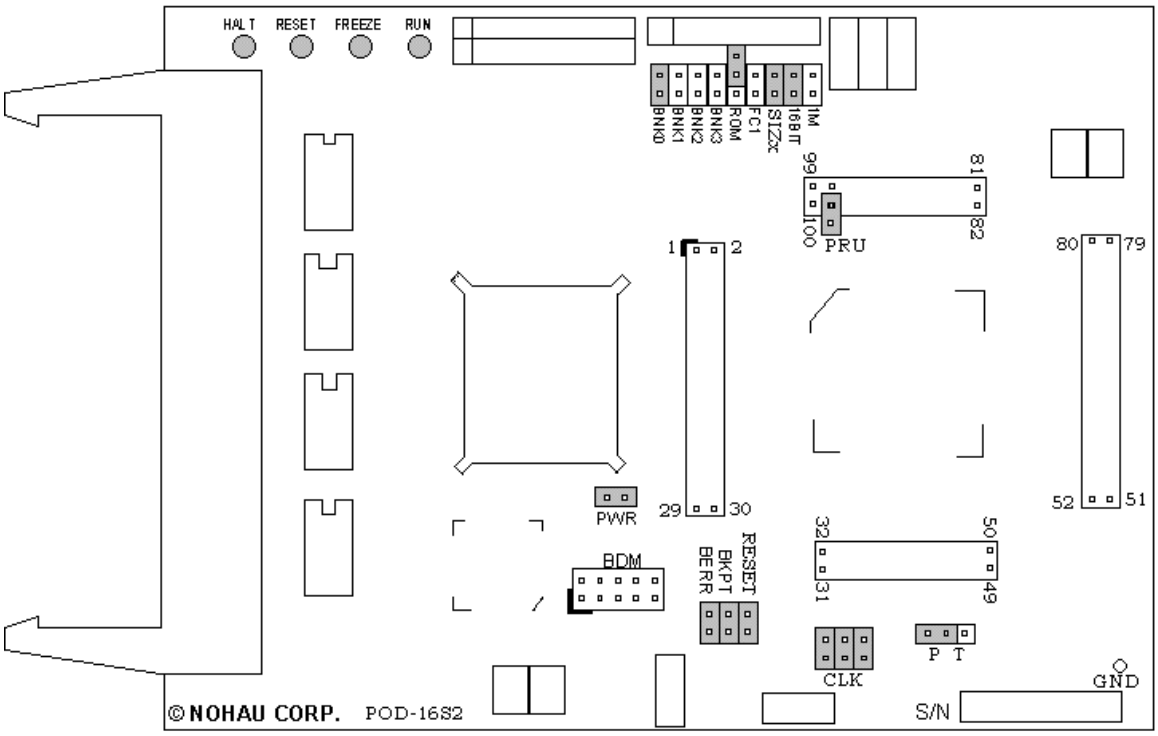


Figure 41. POD–16S2

Overview

Use this pod if you are running the 68HC16S2 series MCU. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

POD–16S2 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–16S2

POD–16S2 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected To a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls the pod memory width
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
RESET	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
P T	P	P or T	Enables either the pod or the target controller

Header and Jumper Details

1M

This jumper must not be removed unless you are using POD–16S2/4M4B. The max address range supported by any one chip select signal is 512K. RAM on the pod can come with 1 MB in a single bank. If your pod has that much RAM, half of it will be wasted unless the Function code signal FC1 is used to effectively bank switch between the two halves of memory. If you have a POD–16S2/1M1B pod, and the FC1 jumper is in place, the FC1 signal from the controller will be

POD–16S2

directed to the high order address bit on the RAM chip, which will create two banks of RAM. One bank will be used for program space. The other bank will be used for data space bus cycles. The same chip select will control both banks, and so will have the same starting address and same size.

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

SIZx

Supports using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is only useful if the PRU jumper is removed AND your application has programmed the port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, contact your local distributor ("Sales Offices, Representatives and Distributors" list at the end of this guide) or Nohau Technical Support at support@icetech.com for assistance.

FC1

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have at least 1 MB of RAM, one bank of RAM will be used for data space bus cycles and the other bank will be used for program space. Both banks will be controlled by the same chip select, and will have the same starting address and same size. The FC1 pin connects to the pod RAM.

POD–16S2

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will NOT effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They should be OFF for targets that use the chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. EMUL16/300–PC can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

POD–16S2

BERR / BKPT / RESET

If a target has a reset button that grounds the /RESET pin when pressed, this will not interfere with emulation so the /RESET jumper should be ON. If the target ties BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RESET pin low. During debugging, that may be inconvenient. The signal from the target /RESET pin passes though the /RESET header. Removing the /RESET jumper will prevent the external device from setting the pod controller.

Note

When emulating the single chip mode, remove the /BERR jumper.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Note

Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

P T

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

POD–16S2

Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where CSE is not available.

The controller on this pod does not need to be removed when there is also a controller on the target. Logic on the pod prevents both controllers from running at the same time. This logic requires that the TSC pin on the target be pulled high through its own 10K resistor, and not be directly tied to V_{CC}. If the target MCU is not installed, and if the TSTME/TSC pin is connected to V_{CC}, you may remove the P T jumper to reduce the power consumed.

Using the Port Replacement Unit

EMUL16/300–PC uses some of the Port E (/AS, /DS, SIZ0, and SIZ1) signals for supporting Shadow RAM and the trace board. This pod includes a Port Replacement Unit (PRU) that completely duplicates the Port E configuration registers and pins. Some target applications use the Port E pins for I/O. For these targets, use the PRU to provide the I/O signals to the target while the pod MCU provides the bus control signals to the emulator.

Applications using the PRU to replace Port E must let the MCU's port E pins carry the bus control signals.

If your target application configures the Port E pins to carry I/O and if you can use the MCU on the pod, follow the instructions below for setting up and using the PRU.

Note

You do not need to use the PRU If your application does not use the Port E pins for I/O. Remove the PRU jumper to disable the PRU.

To use the PRU to emulate Port E, the target application must be recompiled to write to the PRU's Port E configuration registers, not the MCU's Port E configuration registers. The PRU's Port E registers will be at a different address than the MCU's Port E registers.

Chip Select /CS5

The PRU pin is placed next to the header pin for chip select 5 (/CS5), which is MCU pin number 115. You can place a jumper between those two pins if /CS5 is not being used for any other purpose.

If /CS5 is being used for some other purpose, you must choose some other chip select signal to activate the PRU and run a wire between that signal source and the PRU pin. The select signal can be either one generated by the MCU or one generated by address decoding logic on your target.

POD–16S2

Whether you choose /CS5, another MCU chip select signal, or a signal from your own address decoding logic, the signal must be active for at least 16 addresses and not used by any other device on the bus. Further discussion will assume that you are using /CS5 and have a jumper between MCU pin 115 and the PRU pin. Adjust the instruction to suit your particular design.

Set Up Conditional Compilation

There are only two software changes will be needed to support the PRU and they are both typically in the start-up instructions. If you set up a compile-time switch, you can easily switch between using the real Port E (without the emulator) and using the replacement.

Locate the Replacement Registers

To map the replacement Port E registers using /CS5 you must configure /CS5 base address and option registers. We suggest that you map the replacement registers to the same page as the other configuration registers. The smallest block size supported by /CS5 is 2048 bytes, and must start on a 2048 byte page boundary. That leads to a /CSBAR5 value of $FFF8. This value will map those 16 registers to address $FF800. They will repeat every 16 bytes through address $FFFFF.

The option register for /CS5 needs to support reading and writing, both high and low bytes of a 16-bit wide bus, Fast Termination cycles, supervisor or user space, and no autovector support. This leads to a /CSOR5 register value of $7BB0.

Use Replacement Registers

With the PRU jumper in place and with these two values in these two registers, the replacement registers will appear on the bus every 16 bytes, repeatedly from $7FF800 to $80000. They will not hide the internal configuration registers because the replacement registers are external. Of the 16 bytes, 12 are reserved and 4 are used to configure the replacement Port E.

$0	Reserved	Port E Data Register (PORTE)	$1
$2	Reserved	Port E Data Register (PORTE)	$3
$4	Reserved	Port E Data Direction Reg. (DDRE)	$5
$6	Reserved	Port E Pin Assignment Register	$7
$8	Reserved	Reserved	$9
$A	Reserved	Reserved	$B
$C	Reserved	Reserved	$D
$E	Reserved	Reserved	$F

POD–16S2

Connecting the Pod to the Target Board

There are three ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ET/EPP-100-QF49W is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- A Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–16X1

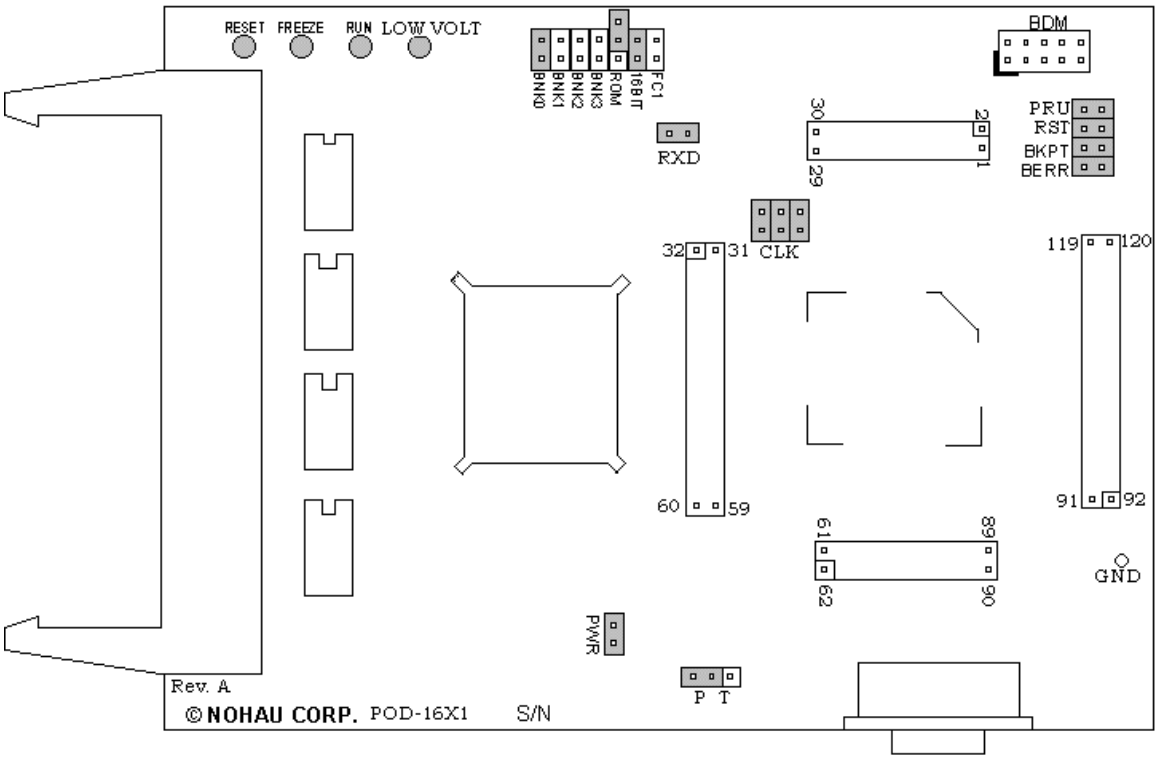


Figure 42. POD–16X1

Overview

POD–16X1 contains a 68HC(9)16X1 chip, a crystal, between 256K and 4 MB of static RAM for instructions and data, a DB-9 connector for the chips serial port, and a PRU used for emulating single-chip applications. They can be used to debut, target applications that use the 68HC(9)16X1 with internal ROM instead of flash ROM. The chip is not socketed.

> **Note**
>
> The 16X1 is significantly different from the 916X1. The pod should be ordered with the appropriate CPU.

This pod board has a standard 68HC(9)16X1 MCU that you can replace if it fails or if you want to use a different processor from the same family. In addition, POD–16X1 includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have BDM connectors and do not have room to connect the pod directly to the target processor.

POD–16X1 LED Indicators

Name	Function	Description
D1	**RESET**	A red LED that lights when the reset to the processor is asserted.
D2	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D3	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**LOW VOLT**	An amber LED indicates that the pod is running on low voltage (+3V) rather than the standard +5 volts.

POD–16X1 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected To a Target	Description
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CS0 to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CSM to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS3 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS5 to pod RAM bank3
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
RXD			Drives the serial port input pin on the controller
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
P T	P	P or T	Enables either the pod or the target controller
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5

POD–16X1

Header and Jumper Details

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will NOT effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

There are up to four banks of RAM on the pod. These jumpers only effect pods with more than one bank of RAM. They should be OFF for targets that use the chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

BERR / BKPT / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the BERR jumper.

FC1

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The CPU16 controllers can address up to 1 MB of RAM, but the largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have a POD–16X1/1M1B, you may use the FC1 signal to allocate one half of the 1-MB emulation memory bank on the pod for program bus cycles (instruction fetch) and the other half for data bus cycles. If your POD–16X1 has 1 MB in bank0 and you have the FC1 header shorted by a jumper, data and code bus cycles will go to separate 512K banks with overlapping addresses. The FC1 pin connects to the pod RAM.

RXD

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on this header.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. EMUL16/300–PC can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

POD–16X1

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Note

Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

P T

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where /CSE is not available.

The controller on this pod does not need to be removed when there is also a controller on the target. Logic on the pod prevents both controllers from running at the same time. This logic requires that the TSC pin on the target be pulled high through its own 10K resistor, and not be directly tied to Vcc. If the target MCU is not installed, and if the TSTME/TSC pin is connected to Vcc, you may remove the P T jumper to reduce the power consumed.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

POD–16X1

Note

The following section only applies to pods with more than one bank of emulation RAM.

RAM BNK1 and /CSM

If your pod is equipped with a 68HC(9)16X1 and not with 68HC16X1, the MCU will not support the /CSM signal. If you wish to use BNK1, you will need to run a wire between some other /CS and the outer of the two pins marked BNK1.

MCU Operating Modes

The 68HC(9)16X1 can operate in one of three operating modes: Fully Expanded (16-bit external bus), Partially Expanded (8-bit external bus), and Single Chip node (no external bus). To find out more about these modes you will need to read the Motorola manuals on these chips.

Before you proceed, make sure that you know which mode is used by the target design, and which pins are held low during the reset cycle.

Is Emulation Possible

This section describes the ways that target designs may interfere with or even prevent emulation. This may be a useful section to review before you attempt to connect your target to the emulator or if you are having trouble emulating.

/CSM and Chip Select Function

If the chip is forced into emulation mode, /CSE will become the chip select for the PRU. /CSM will become chip select whenever the MCU assesses on-chip ROM. In other words, the ROM module must be brought into emulation mode for proper functioning.

For /CSM to function, data bits 11 – 15 must be manipulated appropriately during reset. If the ROM module is mask enabled to respond to reset vectors, pulling bit 14 low will disable the ROM completely. It will enter low power mode and will not respond to reset vectors.

For each of the different modes, certain bits cannot be modified. This is because that when the MCU is in one of the different modes not every combination of bits are possible. If for example the MCU is in single chip mode, no address signals will be brought out to the pins so it makes no sense to configure those bits to carry the address signals.

POD–16X1

Transparent Mode

Some target systems are designed to run in a configuration where bus control needs to be granted to a DMA or other controller. The same pins that carry /CSM and /CSE also carry the signals used to arbitrate control of the bus. If your target design shares the bus between the 68HC16X1 and some other controller, choose the transparent pod configuration.

In transparent mode, all configuration bits may be controlled. You must know precisely how the chip must be configured for successful emulation. It is important that you create a configuration that allows for a successful emulation.

Note

Use transport mode only when /CSE is not available and/or when another bus master shares the pin. Choosing transport mode and setting the same bits as single chip mode is not the same as single chip mode.

Pull TSC Low

Seehau uses the TSC pin to disable either the pod MCU or the target MCU, depending upon the position of the P T jumper. The pod uses a PRU to provide important Port E signals (at all times except for applications with multiple bus masters.) Ideally, you want your target to pull the TSC pin low through a 10K Ohm resistor. This will allow the emulator to disable the target MCU. The pod will also pull this pin up or down as necessary so that you may leave it unconnected (or cut a trace) while the pod is attached to the target.

It is not unreasonable for a target design to tie the TSC pin directly to ground, doing so will prevent the emulator from disabling the target MCU. If it is tied to ground, you must do one of two things: You may disable the target MCU yourself (by removing it from the target, for example). Or, you must place the P T jumper in the T position and understand that the PRU will not be available to support emulation in single-chip or partially expanded mode.

Pull /BKPT and /BERR High

These two pins might be tied directly to +5V on your target. If they are, your target will operate as you expect without an emulator, but will not emulate correctly. Seehau drives these two pins for some very basic features. By removing the /BKPT and /BERR jumpers, you can isolate the target circuitry from the pod circuitry, but this will only be effective if the target controller is disabled. If the P T header in the T position, they must each be pulled up on the target through a separate resistor with a resistance of at least 10K Ohms. The pod will drive these pins up or down as necessary so you may leave them unconnected or cut a trace while the pod is attached.

POD–16X1

Healthy CLKOUT Signal

- No matter which oscillator you use, the pod oscillator or the target oscillator, the signal coming out of the CLKOUT pin (on the active MCU) must be a normal, clear signal.

- If you see the red /RESET light on the pod stay lit after reset should have ended, most likely a clock problem is to blame.

- A target that in some way interferes with the CLKOUT signal coming out of the processor will also interfere with emulation.

If the preceding conditions are met (and you have followed the design constraints of the MCU itself) you will be able to reset the controller, get it into Background Debug Mode, and communicate with the emulator. You may or may not be able to read the contents of memory, but you will be able to read the contents of internal registers, including the MCU registers like the program counter and the stack pointer. If this is not the case, examine the quality of the connections between the adapter and the target, and verify that the pod and the target each operate as expected when not connected.

Other Bus Masters

Some targets are designed with external DMA controllers or include a memory-sharing scheme that requires that the 68HC16X1 give up control of the address and data bus to another device. The handshake that grants control of the bus to another device uses some of the same pins required by the PRU (/BR/CS0, /BG/CSM, and /BGACK/CSE). If the target design holds /BERR high and data pin 2 low during the reset cycle, those pins will be used for bus grant handshaking and cannot be used for the PRU.

Designs that grant bus control to other devices can be emulated, but there are a few restrictions:

- PRU jumper must be off.

- If the Port E signals /AS, /DS, SIZ0, and SIZ1 are available to the emulator, the Shadow RAM and trace board will operate as if the PRU were functioning. The trace board will even capture the bus cycles of the other bus master (and display them as data cycles).

If the SIZ0 and SIZ1 pins are assigned to carry I/O the Shadow RAM data will not be accurate and the trace buffer display may not correctly interpret the data collected. However, the raw data collected (the address and data bus values) will still be correct; it will be exactly what was on the data and address busses. Without AS and DS Shadow RAM the trace board will not operate.

Multiple Bus Masters and Partially Expanded Mode

In addition, if the target has multiple bus masters and runs the MCU in partially expanded mode, there are further limitations:

POD–16X1

- Internal bus cycles will not be completely available to the emulator (or trace board). Only the upper half of internal bus cycles will be shown on the eight available bus pins.

- Emulation RAM on the pod can be used to replace the internal ROM memory, but the external bus cycles will be 8 bits wide. The application timing will be very different than if the application were running out of internal (16-bit) ROM or FLASH memory.

Connecting the Pod to the Target Board

There are four ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ET/CLIP-160-QF07-B is a clip that connects the pins of a surface mounted 68HC16X1 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, the target should be designed so that the processor can manipulate the signal called TSC. 10K pull down in the target system is recommended. If this pin is driven high, the target processor will tri-state all outputs and use the pod MCU for emulation.

- Part number ET/EPP-160-QF07-W is an adapter that solders onto the target board in place of the MCU. This adapter mates with the pod board pins. This is also does not affect the board design, but it does require some special prototype board assembly.

- Finally, a Background Debug connector (BERG) designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. These two emulator features will not operate when the BERG connector is used.

POD–16Y1

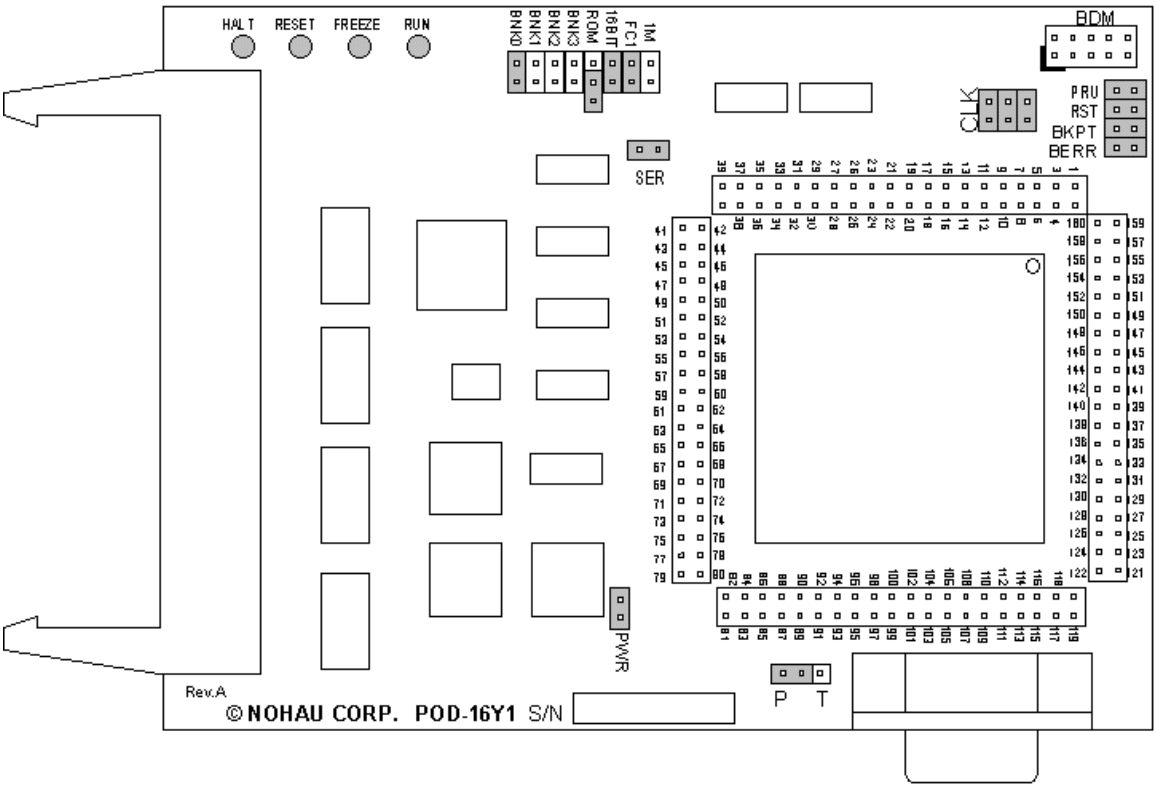


Figure 43. POD–16Y1

Overview

Use this pod if you are running a 68HC(9)16Y1 processor. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

> **Note**
>
> The 16Y1 is significantly different from the (9)16Y1. The pod should be ordered with the appropriate CPU.

POD–16Y1 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–16Y1 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected To a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
16BIT	ON or OFF	ON (OFF if emulating 8-bit memory)	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	On or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	On or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
P T	P	P or T	Enables either the pod or the target controller

POD–16Y1

Header and Jumper Details

1M

This jumper must not be removed. The max address range supported by any one chip select signal is 512K. RAM on the pod can come with 1 MB in a single bank. If your pod has that much RAM, half of it will be wasted unless the Function code signal FC1 is used to effectively bank switch between the two halves of memory. If the FC1 jumper is in place, the FC1 signal from the controller will be directed to the high order address bit on the RAM chip, which will create two banks of RAM. One bank will be used for Program space. The other bank will be used for Data space bus cycles. The same chip select will control both banks, and so will have the same starting address and same size.

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

FC1

16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The CPU16 controllers can address up to 1 MB of RAM, but the largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have a POD–16Y1/1M1B, you may use the function code signal FC1 to allocate one half of the 1-MB emulation memory bank on the pod for program bus cycles (instruction fetch) and the other half for data bus cycles. If your POD–16Y1 has 1 MB in bank0 and you have the FC1 header shorted by a jumper, data and code bus cycles will go to separate 512K banks with overlapping addresses. The FC1 pin connects to the pod RAM.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits in /CSPAR0, which controls whether the /CSBOOT generates bus cycles and address for 8-bit or 16-bit memory. Removing the 16BIT jumper disables half of the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure chip select registers with the software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and it tries to read a 16-bit wide device (or vice versa) the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

POD–16Y1

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will NOT effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They should be OFF for targets that use the chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

Remove the jumper if you have a serial device on the serial port.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

BERR / BKPT / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set to OFF when all the following the conditions are met:

POD–16Y1

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

P T

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where /CSE is not available.

The controller on this pod does not need to be removed when there is also a controller on the target. Logic on the pod prevents both controllers from running at the same time. This logic requires that the TSC pin on the target be pulled high through its own 10K resistor, and not be directly tied to Vcc. If the target MCU is not installed, and if the TSTME/TSC pin is connected to Vcc, you may remove the P T jumper to reduce the power consumed.

POD–16Y1

Notes On the 16Y1 Pod

Jumpers BNK0 through BNK3 control use of emulation memory bank 0 through bank 3. They allow use of an arbitrary MCU's chip select to control an emulation memory bank. By default, they connect the following signals:

Jumper	MCUs / CS	Emulation Memory Bank
BNK0	/CSBOOT	0
BNK1	/CS0	1
BNK2	/CSM	2
BNK3	/CS3	3

If any or the entire above chip selects can not be used for emulation memory control, because they are used by a present target resource, any other set of chip selects may be used. When this is necessary, simply remove a jumper to disconnect an MCUs /CSx from the respective emulation memory bank and wire-wrap desired /CS to the pin marked BNKx.

If the target does not use MCUs /CS to control it's memory (specifically the memory which you want to replace by emulation memory), then output of the target address decoder may be used to control an emulation memory bank. This can be accomplished by connecting a wire between the target's address decoder output and the pin on the pod marked BNKx.

The /CSM spans only address range equal to the size of the on-chip ROM (48K on 68HC16Y1). To test BNK2, connect /CS5 (pin 128) to the pin marked BNK2 as per instructions above.

Connecting the Pod to the Target Board

There are five ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ET/EPP-160-QF07-W is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Part number ET/CLIP-160-QF07-B clip-over for 160-pin plastic QFP SMD. Not recommended for ceramic packages.

POD–16Y1

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–16Y3

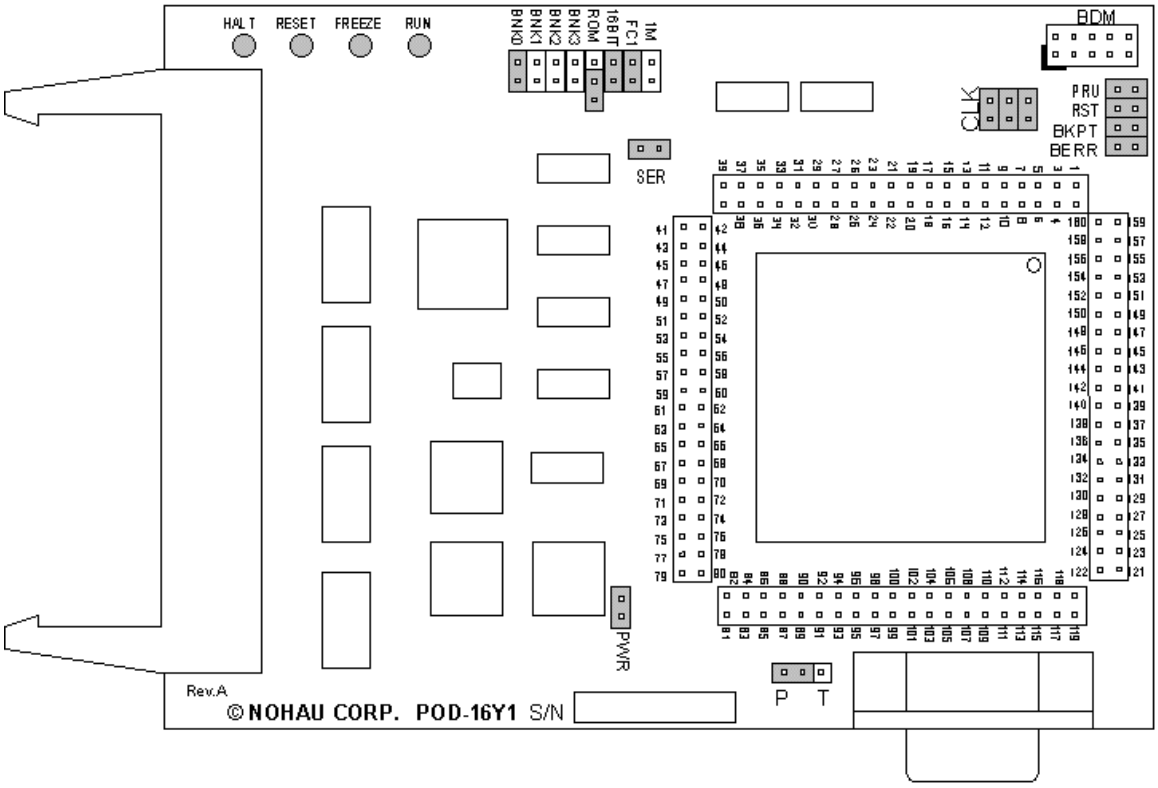


Figure 44. POD–16Y3

Overview

Use this pod if you are running the 68HC(9)16Y3 processor in expanded multiplexed mode or special test mode (external memory). This pod board has a standard 68HC(9)16Y3 that you can replace if it fails or if you want to use a different processor from the same family. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

> **Note**
>
> The (9)16Y3 is significantly different from the 16Y1. The pod should be ordered with the appropriate CPU.

POD–16Y3 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–16Y3 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected To a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
PWR	ON or OFF	OFF (if there is a target power supply)	Provides +5V power to the target
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
16BIT	ON or OFF	ON (OFF if emulating 8-bit memory	Controls memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
P T	P	P or T	Enables either the pod or the target controller
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
BKPT	ON or OFF	ON (OFF if target BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU

POD–16Y3

Header and Jumper Details

1M

The max address range supported by any one chip select signal is 512K. RAM on the pod can come with 1 MB in a single bank. If your pod has that much RAM, half of it will be wasted unless the Function code signal FC1 is used to effectively bank switch between the two halves of memory. If you have the FC1 jumper is in place, the FC1 signal from the controller will be directed to the high order address bit on the RAM chip, which will create two banks of RAM. One bank will be used for Program space. The other bank will be used for Data space bus cycles. The same chip select will control both banks, and so will have the same starting address and same size.

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

FC1

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The CPU16 controllers can address up to 1 MB of RAM, but the largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have a POD–16Y3/1M1B, you may use the FC1 signal to allocate one half of the 1-MB emulation memory bank on the pod for program bus cycles (instruction fetch) and the other half for data bus cycles. If your POD–16Y3 has 1 MB in bank0 and you have the FC1 header shorted by a jumper, data and code bus cycles will go to separate 512K banks with overlapping addresses. The FC1 pin connects to the pod RAM.

POD–16Y3

16BIT

During /RESET, pin 0 of the data bus configures the two least significant bits in /CSPAR0, which controls whether the /CSBOOT generates bus cycles and address for 8-bit or 16-bit memory. Removing the 16BIT jumper disables half of the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure chip select registers with the software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and it tries to read a 16-bit wide device (or vice versa) the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0 (makes RAM bank0 read-only). The ROM jumper will not effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be off.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. It should be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

Jumpers BNK0 to BNK3 control use of emulation memory bank0 through bank3. They allow use of an arbitrary MCU’s chip select to control an emulation memory bank. For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header’s chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

If any or all of the chip selects can not be used for emulation memory control, because they are used by a present target resource, any other set of chip selects may be used. When this is necessary, simply remove a jumper to disconnect an MCU’s /CSx from the respective emulation memory bank and wire-wrap desired /CS to the pin marked BNKx.

POD–16Y3

If the target does not use MCU's /CS to control it's memory (specifically the memory which you want to replace by emulation memory), then output of the target address decoder may be used to control an emulation memory bank. This can be accomplished by connecting a wire between the target's address decoder output and the pin on the pod marked BNKx.

The /CSM spans only address range equal to the size of the on-chip ROM (48K for the 68HC16Y3). To test bank2, connect /CS (128) to the pin marked BNK2.

> **Note**
>
> The 68HC16Y3 does not support the signal /CSM at all (there is no ROM module on the chip and flash module does not support the /CSM). It is therefore necessary to control the bank2 by some other chip select.

P T

Setting determines whether the power is set for the pod or the target MCU. For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where /CSE is not available.

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

SER

Remove the jumper if you have a serial device on the serial port.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

POD–16Y3

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal)
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating single-chip mode, remove the /BERR jumper.

Special Considerations for the 68HC(9)16Y3 MCU

When you receive the POD–16Y3 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.

Chip Select Logic

The 68HC(9)16 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 68HC(9)16 chips and 68HC3xx series chips. You can use the same procedure with 68HC(9)16 chips, except that you expand the address range of the memory transfer to (BF00 - BFFF).

POD–16Y3

Connecting the Pod Board to the Target Board

There are five ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ET/EPP-160-QF07-W is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Part number ET/CLIP-160-QF07-B clip-over for 160-pin plastic QFP SMD. Not recommended for ceramic packages.

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–16Z1

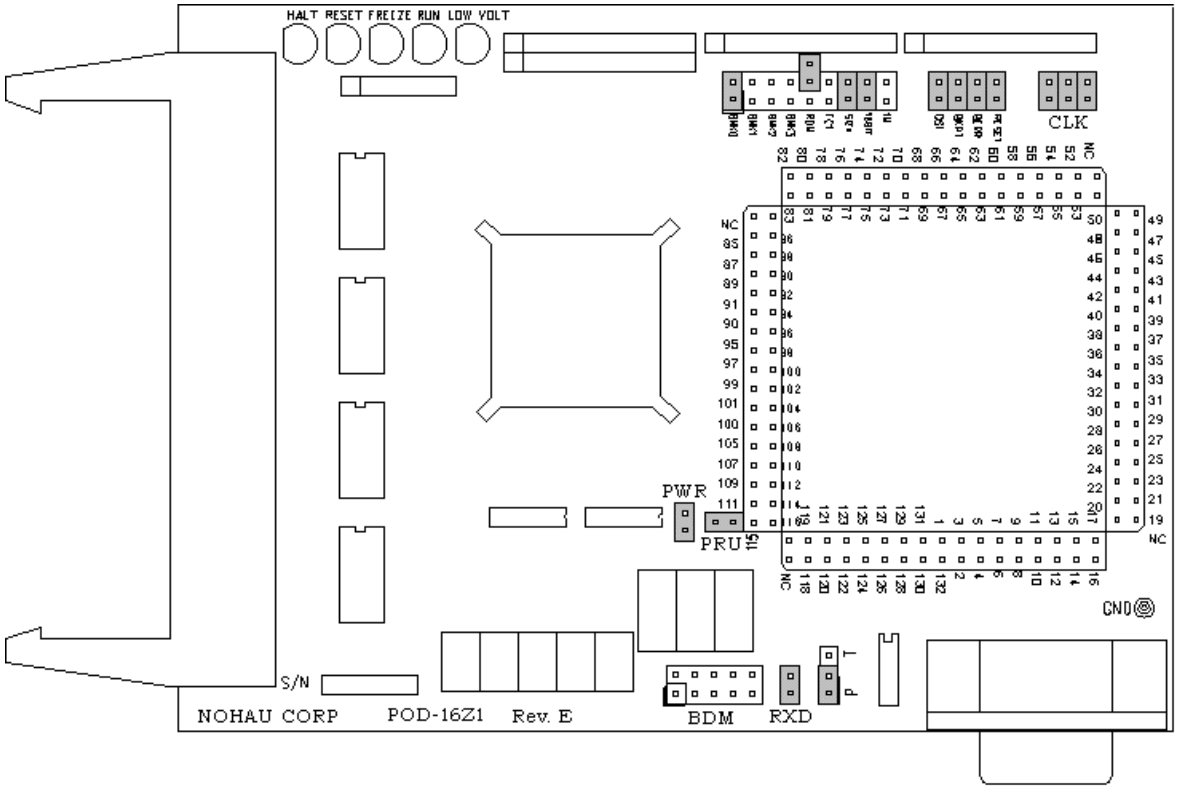


Figure 45. POD–16Z1

Overview

Use this pod if you are running the 68HC16Z1/Z2. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

POD–16Z1 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.
D5	**LOW Volt**	An amber LED that indicates that the pod is running on low voltage (+3.3V) from the target rather than the standard +5V. **Note:** Applicable to Rev. D or later, all previous models had only four LED indicators.

POD–16Z1 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected To a Target	Description
P T	P	P or T	Enables either the pod or the target controller
RXD			Drives the serial port input pin on the controller
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
1M	ON or OFF	ON or OFF	Set at the factory, do not change
16BIT	ON or OFF	ON (OFF if emulating 8-bit memory	Controls memory width
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
RESET	ON	ON (OFF if external watchdog resets MCU)	Connects target RESET signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
DSI	ON or OFF	ON (OFF if target /DSI is tied to +5V)	Connects target /DSI signal to pod MCU

POD–16Z1

Header and Jumper Details

Note

The pod board for the 68HC16Z2 is similar to the pod board for the 68HC16Z1 except for the following features:

- A 4-MHz crystal
- A diode between RESET (pin 69) and D14 (pin 92). This diode disables the internal ROM so that instructions can be executed out of emulation RAM through a reset.
- To support the 68HC16Z2, use the 68HCZ1 with the EMUL16/300-PC/QFP132 CLIP clip-over adapter.

P T

The controller on Rev. D or later does not need to be removed even when there is also a controller on the target. Logic on the pod prevents both controllers from running at the same time. This logic requires that the TSC pin on the target be pulled high through its own 10K resistor, and not be directly tied to Vcc. If the target MCU is not installed, and if the TSTME/TSC pin is connected to Vcc, you may remove the P T jumper to reduce the power consumed.

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where CSE is not available.

RXD

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on this header.

POD–16Z1

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled by using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

1M

If you do not have a POD–16Z1/Z2/1M1B or a POD–16Z1/Z2/4M, do not remove this jumper. The max address range supported by any one chip select signal is 512K. RAM on the pod can come with 1 MB in a single bank. If your pod has that much RAM, half is wasted unless the function code signal FC1 is used to effectively bank switch between the two halves of memory. If you have a POD–16Z1/Z2/1M1B and the FC1 jumper is in place, the FC1 signal from the controller is directed to the high order address bit on the RAM chip, which creates two banks of RAM. One bank is used for program space. The other bank is used for data space bus cycles. Both banks are controlled by the same chip select, and have the same starting address and same size.

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

POD–16Z1

16BIT

During /RESET, pin 0 of the data bus configures the two least significant bits in /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit or 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select with the software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device the vectors are fetched will be incorrect and the MCU will attempt to execute from a wrong address.

SIZx

Useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call your customer representative or Nohau Technical Support for assistance.

FC1

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have at least 1 MB of RAM, one bank of RAM will be used for data space bus cycles and the other bank will be used for program space. Both banks will be controlled by the same chip select, and will have the same starting address and same size. The FC1 pin connects to the pod RAM.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

The jumpers for BNK1, BNK2 and BNK3 only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

POD–16Z1

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank 3. You must remove the BNK3 jumper and use some other chip select signal.

BERR / BKPT / DSI / RESET

If a target has a reset button that grounds the /RESET pin when pressed, this will not interfere with emulation so the /RESET jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR/DSI is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal)
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RESET pin low. During debugging, that may be inconvenient. The signal from the target /RESET pin passes though the /RESET header. Removing the /RESET jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

Port Replacement Unit (PRU)

Seehau uses some of the Port E signals (/AS, /DS, SIZ0, and SIZ1) for supporting Shadow RAM and the trace board. This pod includes a Port Replacement Unit (PRU) that duplicates the Port E configuration registers and pins. Some target applications use the Port E pins for I/O. For these targets, use the PRU to provide the I/O signals to the target while the pod MCU provides the bus control signals to the emulator.

If your target application configures the Port E pins to carry I/O and if you can use the MCU on the pod, follow the instructions below for setting up and using the PRU.

> **Note**
>
> If your application does not use the Port E pins for I/O, you do not need to use the PRU. Remove the PRU jumper to disable the PRU.

POD–16Z1

To use the PRU to emulate Port E, the target application must be recompiled to write to the PRU Port E configuration registers, not the MCU Port E configuration registers. The PRU Port E registers are located at a different address than the MCU Port E registers.

Chip Select /CS5

The PRU pin is placed next to the header pin for chip select 5 (MCU pin number 115). If /CS5 is not being used for any other purpose, you can place a jumper between those two pins very conveniently.

If /CS5 is being used for some other purpose, you must choose some other chip select signal to activate the port replacement unit and run a wire between that signal source and the PRU pin. The select signal can be either one generated by the MCU or one generated by address decoding logic on your target.

Whether you choose /CS5, another MCU chip select signal, or a signal from your own address decoding logic, the signal must be active for at least 16 addresses that are not used by any other device on the bus. Further discussion following assumes you are using /CS5 and have a jumper between MPU pin 115 and the PRU pin. Adjust the instructions to suit your particular design.

Conditional Compilation

Only two software changes support the PRU and they are both in the start-up instructions. If you set up a compile-time switch, you can easily switch between using the real Port E (without the emulator) and using the replacement Port E.

Locate the Replacement Registers

To map the replacement port E registers using /CS5, you must configure /CS5 base address and option registers. We suggest you map the replacement registers to the same page as the other configuration registers. The smallest block size supported by /CS5 is 2048 bytes, and must start on a 2048 byte page boundary. That leads to a /CSBAR5 value of $FFF8. This value maps those 16 registers to address $FF800. They repeat every 16 bytes through address $FFFFF.

The option register for /CS5 needs to support reading and writing, both high and low bytes of a 16-bit wide bus, Fast Termination cycles, supervisor or user space, and no autovector support. This leads to a /CSOR5 register value of $7BB0.

Use the Replacement Registers

With the PRU jumper in place and with these two values in these two registers, the replacement registers appear on the bus every 16 bytes, repeatedly from $FF F800 to $FF FFFF. They do not hide the internal configuration registers because the replacement registers are external.

Of the 16 bytes, 12 are reserved and 4 are used to configure the replacement Port E.

POD–16Z1

Port E Configurations

$0	Reserved	Port E Data Register (PORTE)	$1
$2	Reserved	Port E Data Register (PORTE)	$3
$4	Reserved	Port E Data Direction Reg. (DDRE)	$5
$6	Reserved	Port E Pin Assignment Register	$7
$8	Reserved	Reserved	$9
$A	Reserved	Reserved	$B
$C	Reserved	Reserved	$D
$E	Reserved	Reserved	$F

Notes on Rev. A / B

The 68HC16Z1/Z2 MCUs are very configurable. However, some configurations interfere with operation of the EMUL16/300–PC. Nearly every pin that carries a control signal can be configured to carry a general-purpose input/output signal instead. However, the Seehau software needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10K Ohm pull-up resistor keeps the CPU from accidentally entering BDM and still allows the Seehau to pull the line low when it needs to.

Hint

ISO–160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This might be simpler than modifying the target board. For more information about ISO–160, see Appendix F, “ISO–160.”

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM does not correctly reflect the state of emulation or target RAM. The trace board needs /AS, /DS, and CLKOUT. Target designs that use these port E pins for other signals can interfere with Shadow RAM and trace board operation.

Notes on Rev. C

Unlike the revisions A or B, POD–16Z1/Z2, Rev. C has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable.

Note

To emulate through a reset, bank0 must contain your reset vectors and startup code.

POD–16Z1

The pod was also reconfigured with the following options added or deleted:

Added	Deleted
Z2, /DS, /RST, BKP, BNK0-3, SIZx, ROM	8-bit RAM

Notes on Rev. D or Later

Rev. D, E, and F are identical in functionality and jumper layouts.

Stand-Alone Operation—This revision uses CMOS logic, which has higher input impedance than the logic on previous versions of the pod. As a result, on a stand-alone pod, electrical noise may cause the trace board to capture frames before the MCU comes out of reset. If that happens, the Trace window will show trace frames with unusual looking data.

These pod revisions include a PRU that is active when the P T pod jumper is in the P position. In the P position 8 VDC is applied to the TSC pin of the target board processor (pulled low through a pull down resistor). Raising this pin to 8 VDC will disable all target processor output signals except CLKOUT. The result will be CLKOUT jitter.

The following applies only when using the pod processor (P T jumper in P position) and the target processor is soldered down or socketed. If you have physically removed the 16Z1 from your target board, disregard the following information:

Obtain four SAMTEC headers from Nohau, remove the pin to the XFC signal with pliers (easily pulled from the header), and remove the two pod CLK jumpers closest to the Z1 pod crystal. This will leave one CLK jumper in place (closest to the pod cable). The XFC is pin 60 as numbered on the top of the pod (numbering corresponding to 132-pin package pin assignment). Place a scope on CLKOUT pod-pin 63 and it will now show a clean square wave with no jitter.

For Rev. D the following options were added or changed on the pod:

Added	Deleted
P T, LOW VOLT LED, 1M, PRU, RXD, BERR, FC1	RST to RESET, BKP to BKPT

Connecting the Pod to the Target Board

There are six ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

POD–16Z1

- Part number ET/EP5-132-QF03T is an adapter that plugs into the 132-pin TESTOOL socket.

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Part number ADP-332PGA connects to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–CM16Z1

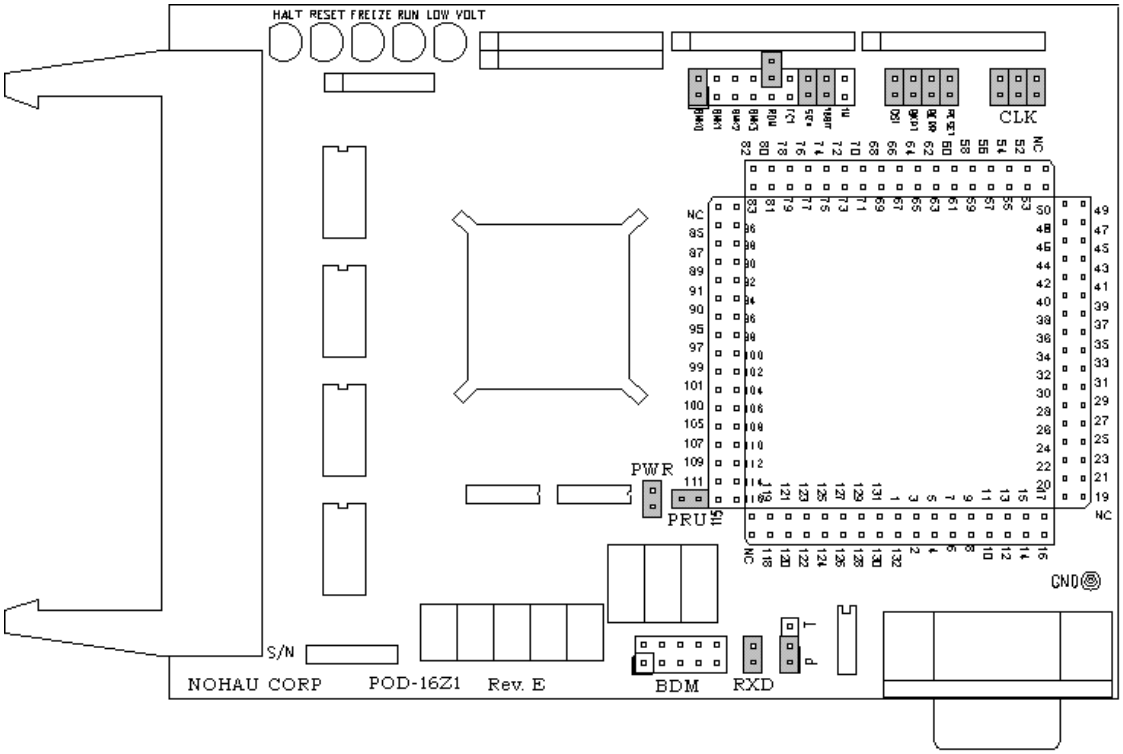


Figure 46. POD–CM16Z1

Overview

Use this pod if you are running the 68HCM16Z1/68HCK16Z1 series MCU.

> **Note**
>
> The CM16Z1 is different from the CK16Z1. The pod should be ordered with the appropriate CPU.

POD–CM16Z1 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.
D5	**LOW Volt**	An amber LED that indicates that the pod is running on low voltage (+3.3V) rather than the standard +5V.

POD–CM16Z1 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
PRU	ON or OFF	ON or OFF	Enables Port Replacement Unit using /CS5
RXD			Drives the serial port input pin on the controller
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RESET	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
DSI	ON or OFF	ON (OFF if target /DSI is tied directly to +5V)	Connects target /DSI signal to pod MCU
FC1	ON or OFF		Connects FC1 pin to pod RAM
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
16BIT	ON	ON (OFF if emulating 8-bit memory)	Controls pod memory width
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
1M	ON or OFF	ON or OFF	Set at the factory, do not change
CLK	ON	OFF	Connects the pod crystal circuit to the pod MCU

POD–CM16Z1

Header and Jumper Details

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

RXD

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin on J1 to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on this header.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled by using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

FC1

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have at least 1 MB of RAM, one bank of RAM will be used for data space bus cycles and the other bank will be used for program space. Both banks will be controlled by the same chip select, and will have the same starting address and same size. The FC1 pin connects to the pod RAM.

POD–CM16Z1

BERR / BKPT / RESET / DSI

If a target has a reset button that grounds the /RESET pin when pressed, this will not interfere with emulation so the /RESET jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR/DSI is tied to +5V this should be set to OFF when the following conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal.
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RESET pin low. During debugging, that may be inconvenient. The signal from the target /RESET pin passes though the /RESET header. Removing the /RESET jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating single chip mode, the /BERR jumper must be removed.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. It should be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

POD–CM16Z1

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will NOT effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits in /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit or 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select with the software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device the vectors are fetched will be incorrect and the MCU will attempt to execute from a wrong address.

SIZx

Supports using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is only useful if the PRU jumper is removed AND your application has programmed the port E pins SIZ signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

POD–CM16Z1

Note

The following section only applies to pods with more than one bank of emulation RAM.

RAM BNK1 and /CSM

If your pod is equipped with 68CM16Z1 or 68CK16Z1 the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are four ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

- Part number ADP-16Z1-SQFP is an adapter that solders onto the target board in place of the 144-pin TQFP (thin QFP) SMD. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–331

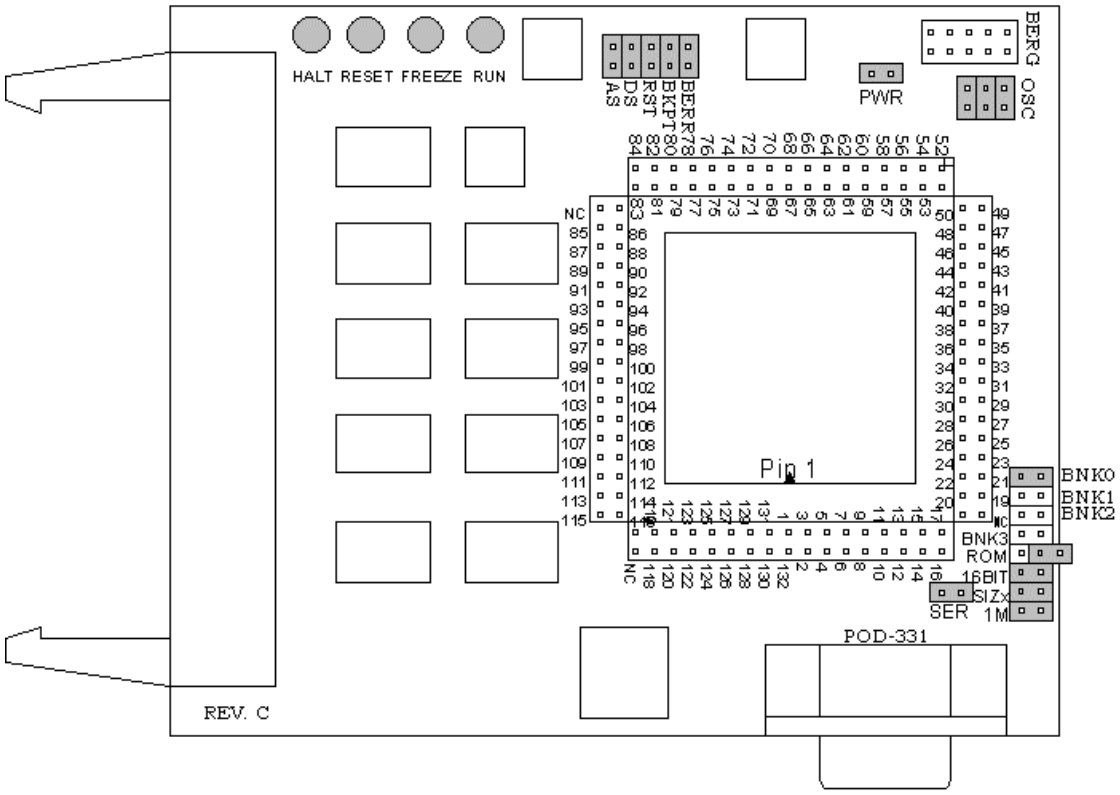


Figure 47. POD–331

Overview

This pod board contains a 16.78-MHz 68331 chip, a 32768-Hz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod board also includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board itself is 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–331 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–331 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
16BIT	ON	ON (OFF if emulating 8-bit memory)	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
OSC	ON	OFF	Connects pod crystal circuit to the pod MCU
AS	ON or OFF		Connects target Address Strobe signal to pod MCU
DS	ON or OFF		Connects target Data Strobe signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU

POD–331

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

SIZx

Support using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

POD–331

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

Remove the jumper if you have a serial device on the serial port.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

OSC

The jumper in position 1 should remain ON if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin. Connects the pod crystal circuit to the pod MCU.

AS

This is a timing signal that indicates the validity of an address on the address bus and of many control signals. It is asserted one half clock after the beginning of a bus cycle.

DS

This is a timing signal. For a read cycle, the MCU asserts the /DS to signal an external device to place data on the bus. This is asserted at the same time as /AS during a read cycle. For a write cycle, the /DS signals and external device that the data on the bus is valid. The MCU asserts /DS one full clock cycle after the assertion of /AS during a write cycle.

POD–331

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be OFF so that the emulator can control the /BKPT pin, as it needs to. When target /BERR/BKPT is tied to +5V these should be set to OFF. For each of these jumpers, remove the jumper only if all three of these conditions are met:

1. The target has circuitry that drives this pin.

2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).

3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

Note

When emulating the single chip mode, remove the BERR jumper.

RAM BNK1 and /CSM

(This section only applies to pods with more than one bank of emulation RAM.)

If your pod is equipped with 68331, 68332, or 68334, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are six ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

POD–331

- Part number QFP132CLIP is a clip that connects the pins of a surface-mounted 68331 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, any production target board connects directly to POD–331. The disadvantage of this adaptation approach is that the BKPT signal must not be tied to +5V directly. The BKPT signal must be pulled up through a resistor (about 10 k Ohms) otherwise the emulator will not be able to assert that signal (by driving it low).

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- A board design that includes a 132 through-hole AMP chip socket can use part number ET/EP5-132-QF03A to connect the pod to the target. The advantage of this approach is that once the target board includes a chip socket, the adapter connects POD–331 to the target. No other changes to the board design or manufacture are needed to use the emulator. The disadvantages are that the socket footprint is slightly larger than the chip without the socket and the cost of the socket itself.

- Part number ADP-332PGA connects a POD–331 to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

68331 Configuration Requirements

The 68331 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

Note

ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

POD–331

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1 MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT. /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively. If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–331 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

> **Note**
>
> To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 megabyte of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned above, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

POD–331

Note

To emulate through a reset, bank0 must contain your reset vectors and startup code.

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of RAM, the best approach is to use the "user defined (below)" chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers /CSBARx so that the size and starting addresses avoid overlap.

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

POD–331

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68332 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68331 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68331 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory. Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68331 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68331 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a standalone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

Other Startup Code Suggestions

Resetting the 68331 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68332 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68331 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are

POD–331

set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68331UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68331 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68331 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiplies the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–331) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the Periodic Interrupt Timer timeout period.

POD–332

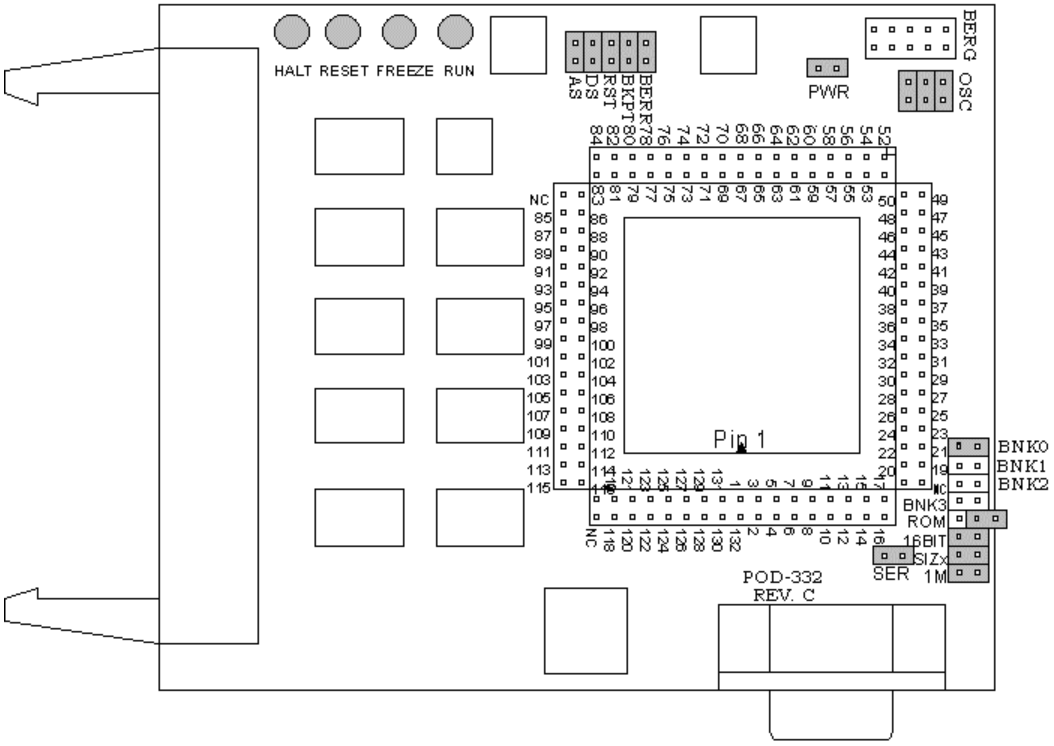


Figure 48. POD–332

Overview

This pod board contains a 16.78-MHz 68332 chip, a 32768 Hz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod board also includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board itself is 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–332 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–332 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
16BIT	ON	ON (OFF if emulating 8-bit memory)	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK3	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
OSC	ON	OFF	Connects pod crystal circuit to the pod MCU
AS	ON or OFF		Connects target Address Strobe signal to pod MCU
DS	ON or OFF		Connects target Data Strobe signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON or OFF	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU

POD–332

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

SIZx

Support using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM.

POD–332

Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

Remove the jumper if you have a serial device on the serial port.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

OSC

The jumper in position 1 should remain ON if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin. Connects the pod crystal circuit to the pod MCU.

AS

This jumper connects target Address Strobe signal to pod MCU.

DS

This jumper connects target Data Strobe signal to pod MCU.

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

POD–332

1. The target has circuitry that drives this pin.

2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).

3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

Note

When emulating the single chip mode, remove the /BERR jumper.

RAM Bank1 and /CSM

(This section only applies to pods with more than one bank of emulation RAM.)

If your pod is equipped with a 68332, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are six ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number QFP132CLIP is a clip that connects the pins of a surface-mounted 68332 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, any production target board connects directly to POD–332. The disadvantage of this adaptation approach is that the BKPT signal must not be tied to +5V directly. The BKPT signal must be pulled up through a resistor (about 10 k Ohms) otherwise the emulator will not be able to assert that signal (by driving it low).

POD–332

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- A board design that includes a 132 through-hole AMP chip socket can use part number ET/EP5-132-QF03A to connect the pod to the target. The advantage of this approach is that once the target board includes a chip socket, the adapter connects POD–332 to the target. No other changes to the board design or manufacture are needed to use the emulator. The disadvantages are that the socket footprint is slightly larger than the chip without the socket and the cost of the socket itself.

- Part number ADP-332PGA connects a POD–332 to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

- Finally, a Background Debug (BERG) connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

68332 Configuration Requirements

The 68332 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

> **Note**
>
> ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

POD–332

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1 MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT. /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively (the 256K pod has only one bank0 and so only uses /CSBOOT). If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–332 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

Note

To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 MB of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned above, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

Note

To emulate through a reset, bank0 must contain your reset vectors and startup code.

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of

POD–332

RAM, the best approach is to use the user defined (below) chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers /CSBARx so that the size and starting addresses avoid overlap.

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68332 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68332 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68332 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory.

POD–332

Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68332 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68332 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a stand-alone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

Startup Code Suggestions

Resetting the 68332 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68332 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68332 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68332UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68332 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

POD–332

Watchdog Timer

When the 68332 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiplies the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–332) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the periodic interrupt timer timeout period.

POD–333

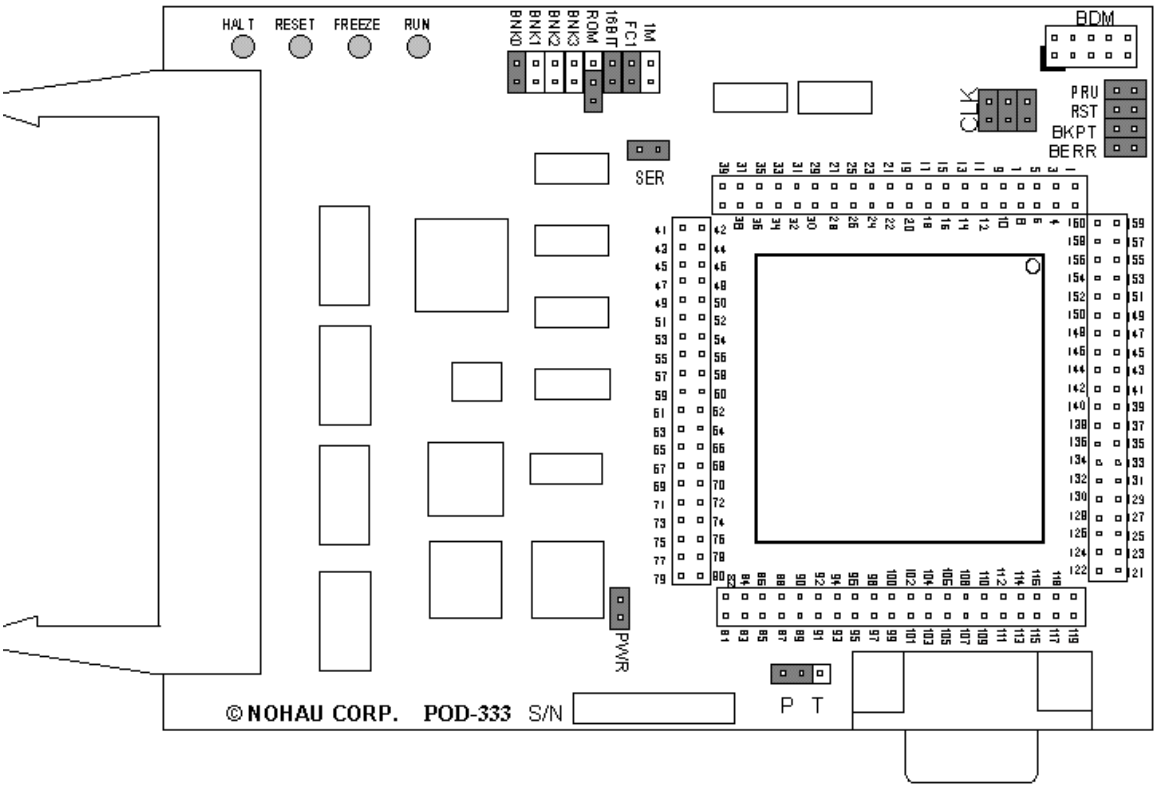


Figure 49. POD–333

Overview

POD–333 contains a 68F333 chip, a crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port.

The pod board also includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board itself is 3.9 inches by 5.1 inches (99.08 mm. by 129.7 mm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target, depending upon which adapter is being used to connect the pod to the target.

POD–333 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–333 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON if 1 meg. banks are used and A19 id available	Connects A19 to emulation RAM
FC1	ON or OFF	ON (when pod RAM > 512K)	Connects FC1 pin to pod RAM
16BIT	ON	ON (OFF if emulating 8-bit memory)	Controls memory width
ROM	On or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	OFF (if there is target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CSM to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS3 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets the MCU)	Connects target /RESET signal to pod MCU
P T	P	P or T	Enables either the pod or the target controller
PRU	ON	ON	Enables the Port Replacement Unit

POD–333

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

FCI

EMUL16/300 pods, including those for the CPU16 controllers, are available with 1 MB of memory bank0. The largest bank size for a single CPU16 chip select signal is 512K. This would normally make one half of each 1-MB emulation memory bank unusable. The FC1 header allows you to use the entire 1-MB bank as two 512K banks.

If you have at least 1 MB of RAM, one bank of RAM will be used for data space bus cycles and the other bank will be used for program space. Both banks will be controlled by the same chip select, and will have the same starting address and same size. The FC1 pin connects to the pod RAM.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

POD–333

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

This jumper connects the MCU serial port to the DB-9 connector on the pod.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Note

Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

POD–333

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.

2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).

3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

Note

When emulating the single chip mode, remove the /BERR jumper.

P T

The controller on Rev. D or later does not need to be removed even when there is also a controller on the target. Logic on the pod prevents both controllers from running at the same time. This logic requires that the TSC pin on the target be pulled high through its own 10K resistor, and not be directly tied to Vcc. If the target MCU is not installed, and if the TSTME/TSC pin is connected to Vcc, you may remove the P T jumper to reduce the power consumed.

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Because the pod has the PRU emulating, nearly all single-chip and partially expanded applications will be easier using the pod MCU.

Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where CSE is not available.

POD–333

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

Note

The following section only applies to pods with more than one bank of emulation RAM.

RAM Bank2 and /CSM

If your pod is equipped with a 68F333, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Chip Description

POD–333 is identical to POD–332 in many ways. The main difference is that POD–333 is capable of running multiple modes such as single chip, partially expanded, and fully expanded. For this reason we have added a new system integration module called Single Chip Integration Module (SCIM), previously referred to as System Integration Module (SIM).

Additional ports A, B, and H are shared with the address bus and data bus respectively. To run in single chip mode it is essential to have an on-chip bug storage device (nonvolatile). For the F333, this is Flash ROM.

With regard to port C and chip select logic, /CSM, /CSE and FC1 replace /CS1, /CS2 and /CS4. If the chip is forced into emulation mode, /CSE will become the chip select for the PRU. /CSM will become the chip select whenever the CPU accesses on-chip ROM. The ROM module must be brought into emulation mode for proper functioning. As the F333 does not have ROM, /CSM is not supported and has no functionality. /CS4 is no longer functional, and you can no longer program this pin to be chip select.

Is Emulation Possible?

This section describes the ways that target designs may interfere with or even prevent emulation. This may be a useful section to review before you attempt to connect your target to the emulator or if you are having trouble emulating.

POD–333

Pull TSC Low

EMUL16/300–PC uses the TSC pin to disable either the pod MCU or the target MCU, depending upon the position of the P T jumper. The pod uses a PRU to provide important Port E signals (at all times except for applications with multiple bus masters.) Ideally, you want your target to pull the TSC pin low through a 10-k Ohm resistor. This will allow the emulator to disable the target MCU. The pod will also pull this pin up or down as necessary so that you may leave it unconnected (or cut a trace) while the pod is attached to the target.

It is not unreasonable for a target design to tie the TSC pin directly to ground, doing so will prevent the emulator from disabling the target MCU. If it is tied to ground, you must do one of two things:

- You may disable the target MCU yourself (by removing it from the target, for example).

OR

- You must place the P T jumper in the T position and understand that the PRU will not be available to support emulation in single-chip or partially expanded mode.

Pull /BKPT and /BERR High

These two pins might be tied directly to +5V on your target. If they are, your target will operate as you expect without an emulator, but will not emulate correctly. EMUL16/300–PC drives these two pins for some very basic features. By removing the /BKPT and /BERR jumpers, you can isolate the target circuitry from the pod circuitry, but this will only be effective if the target controller is disabled. If the P T header in the T position, they must each be pulled up on the target through a separate resistor with a resistance of at least 10 k Ohms. The pod will drive these pins up or down as necessary so you may leave them unconnected or cut a trace while the pod is attached.

Healthy CLKOUT Signal

No matter which oscillator you use, the pod oscillator or the target oscillator, the signal coming out of the CLKOUT pin (on the active MCU) must be normal, clear signal. If you see the red /RESET light on the pod stay lit after reset should have ended, most likely a clock problem is to blame. A target that in some way interferes with the CLKOUT signal coming out of the processor will also interfere with emulation.

If all three of the above conditions are met (and you have followed the design constraints of the MCU itself) you will be able to reset the controller, get it into Background Debug Mode, and communicate with the emulator. You may or may not be able to read the contents of memory, but you will be able to read the contents of internal registers, including the MCU registers like the program counter and the stack pointer. If this is not the case, examine the quality of the connections between the adapter and the target, and verify that the pod and the target each operate as expected when not connected.

POD–333

Other Bus Masters

Some targets are designed with external DMA controllers or include a memory-sharing scheme that requires that the 68HC16X1 give up control of the address and data bus to another device. The handshake that grants control of the bus to another device uses some of the same pins required by the PRU (BR/CS0, BG/CSM, and /BGACK/CSE). If the target design holds BERR high and data pin 2 low during the reset cycle, those pins will be used for bus grant handshaking and cannot be used for the PRU.

Designs that grant bus control to other devices can be emulated, but there are a few restrictions:

- PRU jumper must be off.

- If the Port E signals /AS, /DS, SIZ0, and SIZ1 are available to the emulator, the Shadow RAM and trace board will operate as if the PRU were functioning. The trace board will even capture the bus cycles of the other bus master (and display them as data cycles).

- If the SIZ0 and SIZ1 pins are assigned to carry I/O the Shadow RAM data will not be accurate and the trace buffer display may not correctly interpret the data collected. However, the raw data collected (the address and data bus values) will still be correct; it will be exactly what was on the data and address busses. Without /AS and /DS Shadow RAM and the trace board will not operate at all.

Multiple Bus Masters and Partially Expanded Mode

In addition, if the target has multiple bus masters and runs the MCU in partially expanded mode, there are further limitations:

- Internal bus cycles will not be completely available to the emulator (or trace board). Only the upper half of internal bus cycles will be shown on the eight available bus pins.

- Emulation RAM on the pod can be used to replace the internal ROM memory, but the external bus cycles will be 8 bits wide. The application timing will be very different than if the application were running out of internal (16-bit) ROM or FLASH memory.

Connecting the Pod to the Target Board

There are four ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignment of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

POD–333

- Part number ET/CLIP-160-QF07-B is a clip that connects the pins of a surface-mounted 68F333 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, the target board must be designed so the processor can manipulate the signal called TSC. 10K pull down in the target system is recommended. If this pin is driven high, the target processor will tri-state all outputs and use the pod MCU for emulation.

- Part number ET/EPP-160-QF07 is an adapter that solders onto the target board in place of the MCU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

MCU Operating Modes

The 68F333 and the 68333 can operate in one of three operating modes: fully expanded (16-bit external bus), partially expanded (8-bit external bus) and single chip mode (no external bus). These three modes are defined by Motorola and are described in detail in the Motorola Users Manual for each chip.

EMUL16/300–PC software also defines a mode we call transparent mode. In this fourth mode, any combination of 16 pins can be held low or high during the reset cycle. Some target systems are designed to run in a configuration where bus controls needs to be granted to a DMA or other controller. The same pins that carry /CSM and /CSE also carry the signals used to arbitrate control of the bus. If your target design shares the bus between the 68F333 and some other controller, choose the transparent pod configuration.

In transparent mode, any and/or all configuration bits may be controlled. You must know precisely how the chip must be configured for successful emulation. You must create a configuration that follows the guidelines.

Before you proceed, make sure you know which mode is used by the target design, and which pins are held low during the reset cycle.

BDM

It is possible to use BDM with POD–333 operating in the single chip mode with the limitation that bus error will be pulled low permanently. If the processor accidentally executes an access to an unpopulated location, it will run into problems. In this case, enable the internal bus monitor thereby asserting the bus error into the emulator.

POD–333

Startup Code Suggestions

Resetting the 68F333 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or "shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68332 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68F333 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68331UM/AD User’s Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68F333 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68331 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiples the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–333) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up

POD–333

to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the Periodic Interrupt Timer timeout period.

Port Configuration

There are three modes: single-chip, partially expanded and fully expanded.

To force the chip to operate in single-chip mode, pull the bus error (/BERR) line low during reset and keep it low during normal operation. In single-chip mode, the address and data bus are not visible externally, and are replaced by ports A, B, G and H.

In partially expanded mode, you havehalf the data bus (8 bits) available, so you can use 8-bit external devices. In this mode, it is impossible to program any of the chip selects to run in 16-bit mode. For instance, if address lines A0 to A18 inclusive are enabled, port A and port B are no longer available.

In fully expanded mode, you get one additional port, which will show cycles and try to execute from on-chip memory. Internally all access will be 16-bit and transfers will not be seen externally on other addresses. It is impossible to multiplex or slow down since two bus transfers are required instead of one, therefore, you will have to show both.

If the bus error line is not kept low during reset, either fully expanded or partially expanded mode will be enabled. To enable 16-bit fully expanded mode, pull data bit 1 low. Partially expanded mode is enabled if bit 1 is not pulled low. For any other configuration option, use data bus bits.

POD–334

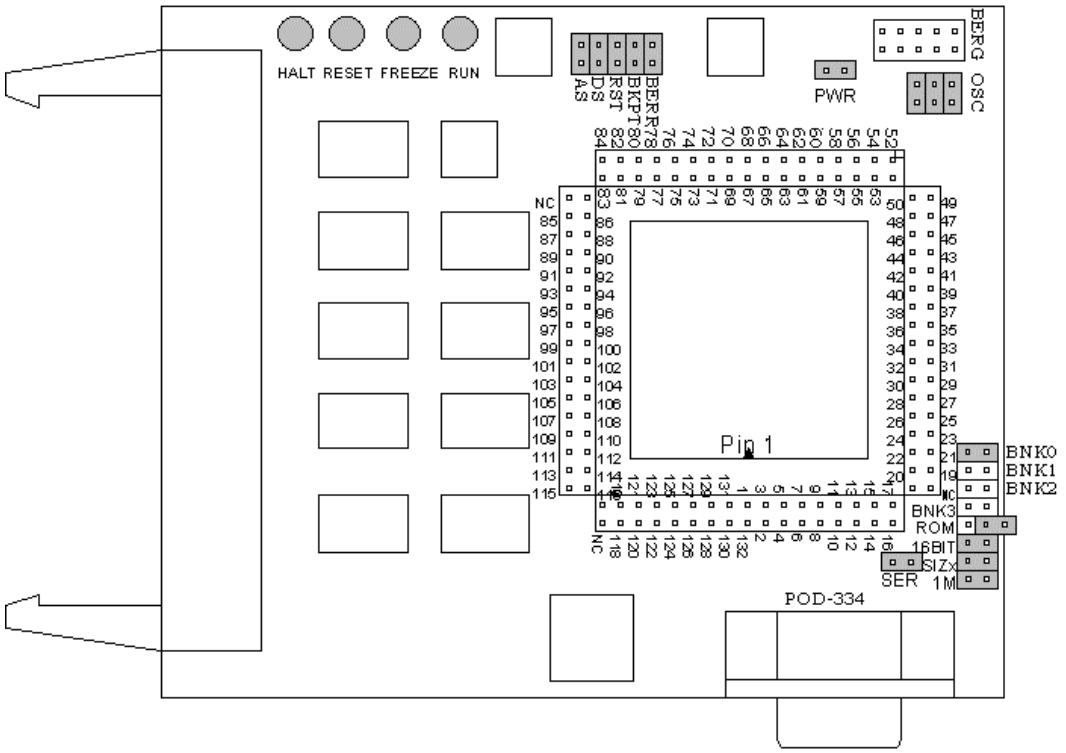


Figure 50. POD–334

Overview

This pod board contains a 16.78-MHz 68334 chip, a 32768-Hz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the pod's serial port. The pod board also includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board itself is 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–334 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–334 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
16BIT	ON	ON (OFF if emulating 8-bit memory)	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
AS	ON or OFF		Connects target Address Strobe signal to pod MCU
DS	ON or OFF		Connects target Data Strobe signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
OSC	ON	OFF	Connects pod crystal circuit to the pod MCU

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

POD–334

SIZx

Support using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Makes RAM bank0 read-only. Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

This jumper connects the MCU serial port to the DB-9 connector on the pod.

POD–334

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

AS

This jumper connects target Address Strobe signal to pod MCU.

DS

This jumper connects target Data Strobe signal to pod MCU.

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

POD–334

OSC

The jumper in position 1 should remain ON if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin. Connects the pod crystal circuit to the pod MCU.

> **Note**
>
> The following section only applies to pods with more than one bank of emulation RAM.

RAM Bank1 and /CSM

If your pod is equipped with a 68334, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are five ways to connect the pod to the target system. Each method has advantages and disadvantages.

- Part number QFP132CLIP is a clip that connects the pins of a surface-mounted 68334 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, any production target board connects directly to POD–334. The disadvantage of this adaptation approach is that the BKPT signal must not be tied to +5V directly. The BKPT signal must be pulled up through a resistor (about 10 Kilo ohms) otherwise the emulator will not be able to assert that signal (by driving it low).

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- A board design that includes a 132 through-hole AMP chip socket can use part number ET/EP5-132-QF03A to connect the pod to the target. The advantage of this approach is that once the target board includes a chip socket, the adapter connects POD–334 to the target. No other changes to the board design or manufacture are needed to use the emulator. The disadvantages are that the socket footprint is slightly larger than the chip without the socket and the cost of the socket itself.

- Part number ADP-332PGA connects a POD–334 to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

POD–334

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

68334 Configuration Requirements

The 68334 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

> **Note**
>
> ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1 MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT. /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively. If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–334 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

POD–334

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

> **Note**
>
> To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 MB of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned earlier, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

> **Note**
>
> To emulate through a reset, bank0 must contain your reset vectors and startup code.

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of RAM, the best approach is to use the user defined (below) chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers /CSBARx so that the size and starting addresses avoid overlap.

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

POD–334

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68334 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68334 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68334 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory. Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68334 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68334 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a standalone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

POD–334

Other Startup Code Suggestions

Resetting the 68334 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68334 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68334 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68334UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68334 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68334 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

POD–334

Processor Clock Rate

At reset, the processor registers are set so that the processor multiples the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–334) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the periodic interrupt timer timeout period.

POD–335

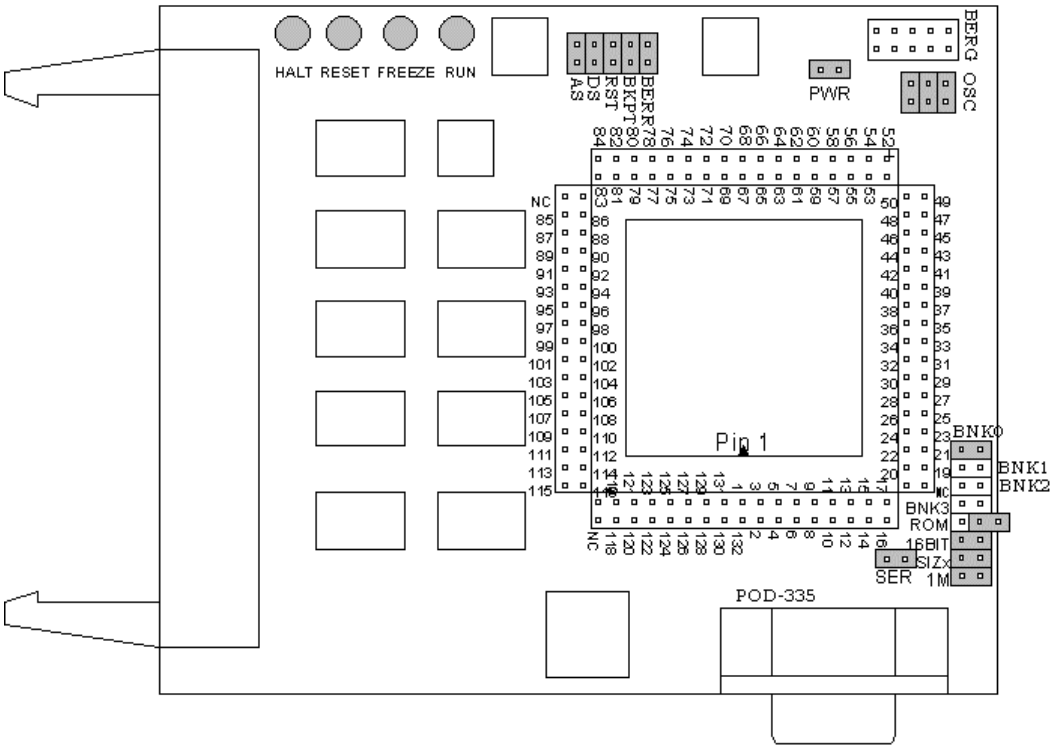


Figure 51. POD–335

Overview

This pod board contains a 16.78-MHz 68335 chip, a 32768-Hz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod board also includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board itself is 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–335 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–335 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	On or OFF	ON or OFF	Makes RAM bank0 read-only
OSC	ON	OFF	Connects pod crystal circuit to the pod MCU
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
AS	ON or OFF		Connects target Address Strobe signal to pod MCU
DS	ON or OFF		Connects target Data Strobe signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU

Header and Jumper Details

OSC

The jumper in position 1 should remain ON if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin. Connects the pod crystal circuit to the pod MCU.

POD–335

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

SIZx

Support using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

POD–335

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

This jumper connects the MCU serial port to the DB-9 connector on the pod.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

AS

This jumper connects target Address Strobe signal to pod MCU.

DS

This jumper connects target Data Strobe signal to pod MCU.

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

POD–335

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

RAM Bank1 and /CSM

(This section only applies to pods with more than one bank of emulation RAM.)

If your pod is equipped with a 68335, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are five ways to connect the pod to the target system. Each method has advantages and disadvantages.

- Part number QFP132CLIP is a clip that connects the pins of a surface-mounted 68334 to the pins on the pod. This connection is temporary and has the advantage that it does not affect the board design or manufacture at all. With this clip, any production target board connects directly to POD–334. The disadvantage of this adaptation approach is that the BKPT signal must not be tied to +5V directly. The BKPT signal must be pulled up through a resistor (about 10 k Ohms) otherwise the emulator will not be able to assert that signal (by driving it low).

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- A board design that includes a 132 through-hole AMP chip socket can use part number ET/EP5-132-QF03A to connect the pod to the target. The advantage of this approach is that once the target board includes a chip socket, the adapter connects POD–334 to the target. No other changes to the board design or manufacture are needed to use the emulator. The disadvantages are that the socket footprint is slightly larger than the chip without the socket and the cost of the socket itself.

- Part number ADP-332PGA connects a POD–334 to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

POD–335

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

68335 Configuration Requirements

The 68335 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

> **Note**
>
> ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1 MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT. /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively. If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–335 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

POD–335

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

> **Note**
>
> To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 MB of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned above, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

> **Note**
>
> To emulate through a reset, bank0 must contain your reset vectors and startup code.

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of RAM, the best approach is to use the user defined (below) chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers /CSBARx so that the size and starting addresses avoid overlap.

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

POD–335

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68335 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68335 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68335 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory. Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68335 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68335 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a stand-alone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

POD–335

Other Startup Code Suggestions

Resetting the 68335 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68332 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68334 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68334UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68335 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68335 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiples the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–335) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up

to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the Periodic Interrupt Timer timeout period.

Connecting the Pod to the Target Board

There are five ways to connect the pod to the target system. Each method has advantages and disadvantages.

- Part number ET/EP5-132-QF03T is an adapter that plugs into the 132-pin TESTOOL socket.

- Part number ET/EPP-132-QF03-LG is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Part number ADP-332PGA connects a POD–335 to a 132-pin PGA socket. Or, if the target board assembly solders the controller directly to the board, the PGA socket included with ADP-332PGA can be soldered onto the target board in place of the controller.

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–336

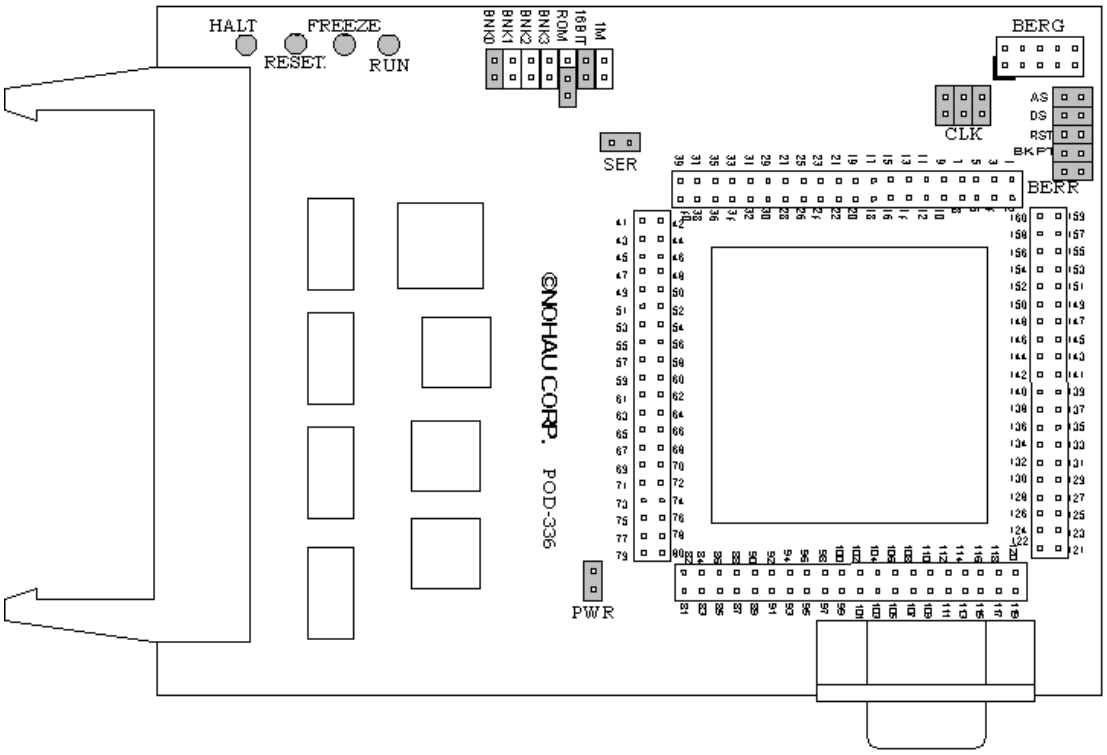


Figure 52. POD–336

Overview

The EMUL16/300–PC pod board contains a 68336 MCU, a 4.192-MHz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board measures 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–336 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–336 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
AS	ON or OFF		Connects target Address Strobe signal to pod MCU
DS	ON or OFF		Connects target Data Strobe signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

POD–336

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

This jumper connects the MCU serial port to the DB-9 connector on the pod.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

POD–336

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Note

Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

AS

This jumper connects target Address Strobe signal to pod MCU.

DS

This jumper connects target Data Strobe signal to pod MCU.

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

POD–336

Note

When emulating the single chip mode, remove the /BERR jumper.

RAM Bank1 and /CSM

(This section only applies to pods with more than one bank of emulation RAM.)

If your pod is equipped with a 68336, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

68336 Configuration Requirements

Each MCU is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do the job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

Note

To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Connecting the Pod to the Target Board

There are four ways to connect the pod to the target system. Each method has advantages and disadvantages.

- Part number ET/CLIP-160-QF07-B is a clip-over for 160-pin plastic QFP SMD. Not recommended for ceramic packages.

- Part number ET/EPP-160-QF07-W is an adapter that solders onto the target board in place of the CPU. This adapter mates with the pod board pins. This also does not affect the board design, but it does require some special board assembly.

- Part number SAMTEC/SSQ-117-03-GD is a 34-pin extender strip to raise the height of the pod. Sold individually, you will need four strips per layer. Each layer of four strips elevates the pod approximately ½inch.

POD–336

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–338

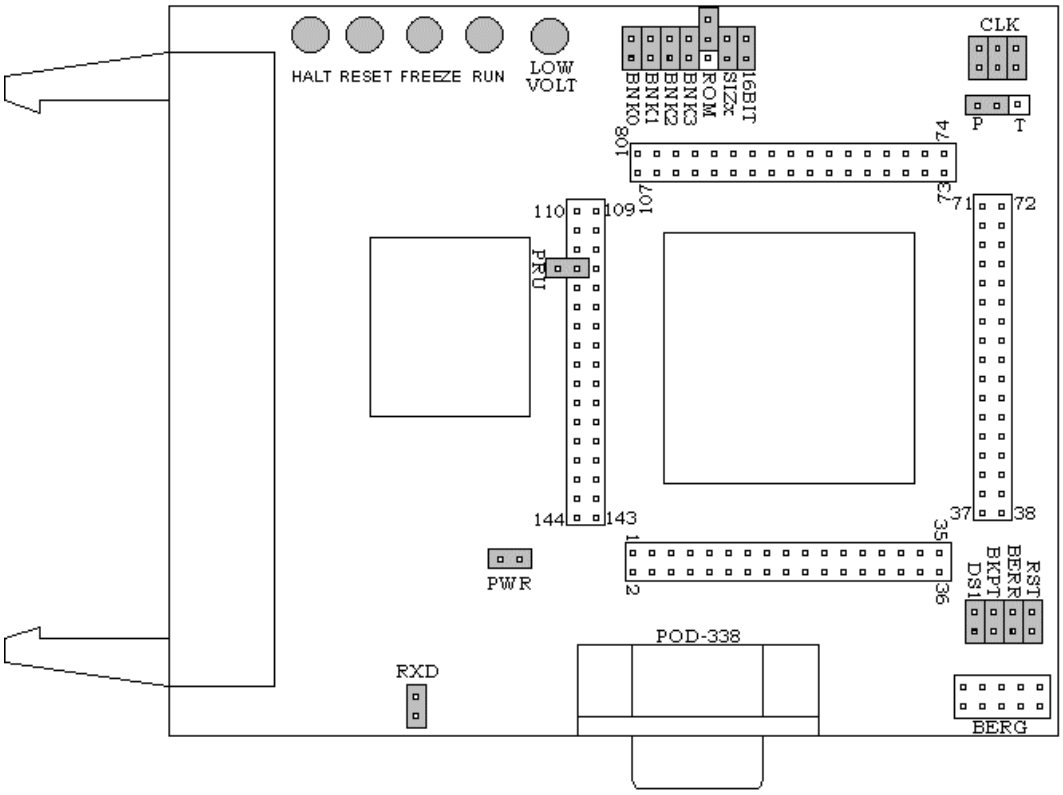


Figure 53. POD–338

Overview

This pod board is used to emulate the Motorola microcontrollers 68300 series in single-chip mode. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

POD–338 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–338 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
SIZx	ON	ON	Supports using emulation RAM without Port E SIZ0 and SIZ1 signals
ROM	On or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
PRU	ON	ON	Enables Port Replacement Unit using /CS5
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
RXD			Drives the serial port input pin on the controller
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU
DSI	ON or OFF	ON (OFF if target /DSI is tied directly to +5V)	Connects target data stream input signal to pod MCU
P T	P	P or T	Enables either the pod or the target controller
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU

Header and Jumper Details

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

POD–338

SIZx

Support using emulation RAM without Port E SIZ0 and SIZ1 signals. The SIZx header is useful only if the PRU jumper is removed and your application has programmed the Port E pins SIZx signals to carry I/O signals. If you cannot use the PRU and the Port E SIZx pins are used to carry I/O, call Nohau Technical Support for assistance.

ROM

Makes RAM bank0 read-only. Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

PRU

The Port Replacement Unit (PRU) makes it possible to emulate single-chip and partially expanded applications. It replaces the MCU Port E pins in all modes, which assures that Shadow RAM and the trace board will have the bus control signals they need at all times. Do not remove this jumper unless /CSE is not available. The PRU is enabled using /CS5. The PRU supports full-featured emulation even if Port E is used for I/O.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

POD–338

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Note

Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

RXD

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin on J1 to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on this header.

BKPT / BERR / DSI / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR/DSI is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.
2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).
3. The P T jumper is in the P position.

POD–338

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

P T

For most circumstances, leave the jumper in the default or P position. When the jumper is in the P position, the target processor is tri-stated and the pod processor controls the bus and peripherals. Putting the jumper in the T position will disable the pod processor and allow the target processor to run. This is required for applications that include multiple bus masters, and other designs where /CSE is not available.

> **Note**
>
> The following section only applies to pods with more than one bank of emulation RAM.

RAM Bank1 and /CSM

If your pod is equipped with a 68338, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CSx signal and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are three ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ADP-144-050-SM adapter to solder to 144 pin TQFP Surface Device (SMD) pads.

POD–338

- Finally, a Background Debug connector designed into the target connects the pod to the target with a 10-wire ribbon cable. The small ribbon cable can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the above adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. Those two emulator features will not operate when the BERG connector is used.

POD–340

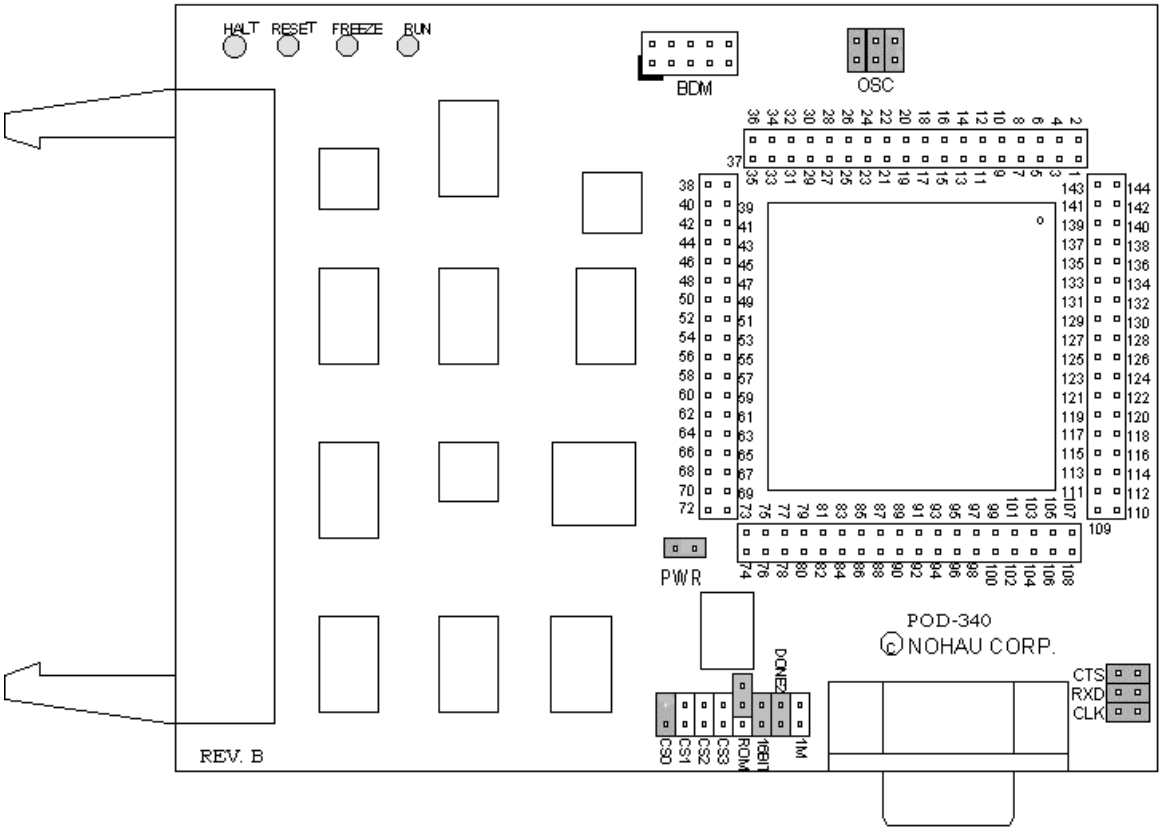


Figure 54. POD–340

Overview

This pod board contains a 16.78-MHz 68340 chip, a 32768 crystal, 256K, 1 MB, 4 MB of emulation RAM for instructions and data, and a DB-9 connector for connecting to a serial port. The pod board also includes a BERG connector that allows debugging target boards that have BERG connectors and do not have room to connect the pod directly to the target processor.

POD–340 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–340 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
OSC	ON	OFF	Connects pod crystal circuit to the pod MCU
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
1M	ON or OFF	ON or OFF	Set at the factory, do not change
DONE-2			DMA controller signal allows the trace board to monitor the DONE2 signal
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
CS0			Chip select line used to control emulation RAM
CS1			Chip select line used to control emulation RAM
CS2			Chip select line used to control emulation RAM
CS3			Chip select line used to control emulation RAM
CTS			Connects the CTS signal from channel A to the DB-9 port
RXD			Drives the serial port input pin on the controller
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU

Header and Jumper Details

OSC

The jumper in position 1 should remain ON if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin. Connects the pod crystal circuit to the pod MCU.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

POD–340

Note

This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

DONE-2

If a trace board is not present, this pin pair does nothing. If there is a trace board, shorting this pin pair with a jumper will allow the trace board to monitor the DONE2 signal. Without this jumper, the trace board will monitor SHOWCYCLES (generated by pod board) instead.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0 (makes RAM bank0 read-only). The ROM jumper will not effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper must be off.

CSO-3

These headers are used to control emulation RAM and are used by the CPU right after reset. Emulation RAM is often used in place of boot RAM.

POD–340

CTS

This jumper connects the CTS signal to DB-9 from serial channel A.

RXD

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin on J1 to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on the RXD header.

CLK

This jumper connects serial channel A clock to DB-9.

Connecting the Pod to the Target Board

There are six ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ADP-144-065-SM is an adapter that solders to 144 pin PQFP SMD pads.

- Part number ADP-340PGA plugs into a pin grad array socket and has pins for plugging into POD–340. IF the target design has a PGA socket, this adapter does not require any special assembly or design. If the target board assembly solders the controller directly to the target board, the PGA socket included with the ADP-340PGA can be soldered onto the target board in place of the controller.

- Part number ADP340-050-SM adapter to solder to 144-pin TQFP.

- Part number ET/CLIP-144-QF10-BC clip-over for 144-pin QFPSMD.

POD–340

- If designed into the board a 10-wire ribbon cable (BERG connector) designed to allow the pod board to connect to the target board. The cable and connector can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the previously mentioned adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. These two emulator features will not operate when the BERG connector is used.

> **Note**
>
> ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The 68340 was designed with only four chip select lines, /CS0, /CS1, /CS2, and /CS3. One of them must be used to control emulation RAM. Choose one chip select line to dedicate to emulation RAM. The best choice is /CS0. Not only is this the one used by the CPU right after a reset, but since emulation RAM is often used in place of the boot ROM, /CS0 is likely to be free to use for emulation RAM. While it is possible to use /CS1, /CS2, or /CS3 to select emulation RAM, it will be difficult unless /CS0 selects a ROM with the reset vectors and startup code. After every reset, you will have to manually set the chip select registers, the stack pointer, and the program counter. On the /CSx header, put a jumper on the pair of pins that correspond to the chosen chip select.

68340 Configuration Requirements

The 68340 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

> **Note**
>
> ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

POD–340

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1-MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT. /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively. If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–340 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

Note

To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 MB of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned above, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

Note

To emulate through a reset, bank0 must contain your reset vectors and startup code.

POD–340

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of RAM, the best approach is to use the user defined (below) chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers CSBARx so that the size and starting addresses avoid overlap.

String WRITE	'(ROM * CSx * RW	:Any write if not ROM emulator
	+ROM * FRZ *CSx * RW)'	:Write only while in BDM
A17A0	= A0 * 16BIT * 1MB	:A0 in 8-bit mode
	+ A17 * 16BIT * 1MB	:A17 in 16-bit mode
	+ A17 * 1MB	:A17 with 512K chips
A19A0	= A0 * 16BIT	:A0 in 8-bit mode
	+ A19 * 16BIT	:A19 in 16-bit mode
A18VCC	= * 1MB	:VCC with 128K chips
	+ A18 * 1MB	:A18 with 512K chips
UWE	= WRITE* 16BIT	:Any write in 8-bit mode
	+ WRITE* 16BIT * A0	:Any even write in 16-bit mode
LWE	= WRITE* A0	:Any odd write
	+ WRITE* SIZ1 * SIZ0* A0	:Any word write
	+ WRITE* SIZ1 * SIZ0* A0	:Even 3-byte write
	+ WRITE* SIZ1 * SIZ0* A0	:Even long write
	+16BIT	:Keep the RAM tri-stated in 8-bit mode

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

POD–340

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68340 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68340 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68340 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory. Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68340 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68340 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a standalone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

Other Startup Code Suggestions

Resetting the 68340 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or "shown" to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68340 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show

POD–340

internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68340 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68334UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68340 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68340 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiples the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–340) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the Periodic Interrupt Timer timeout period.

POD–341

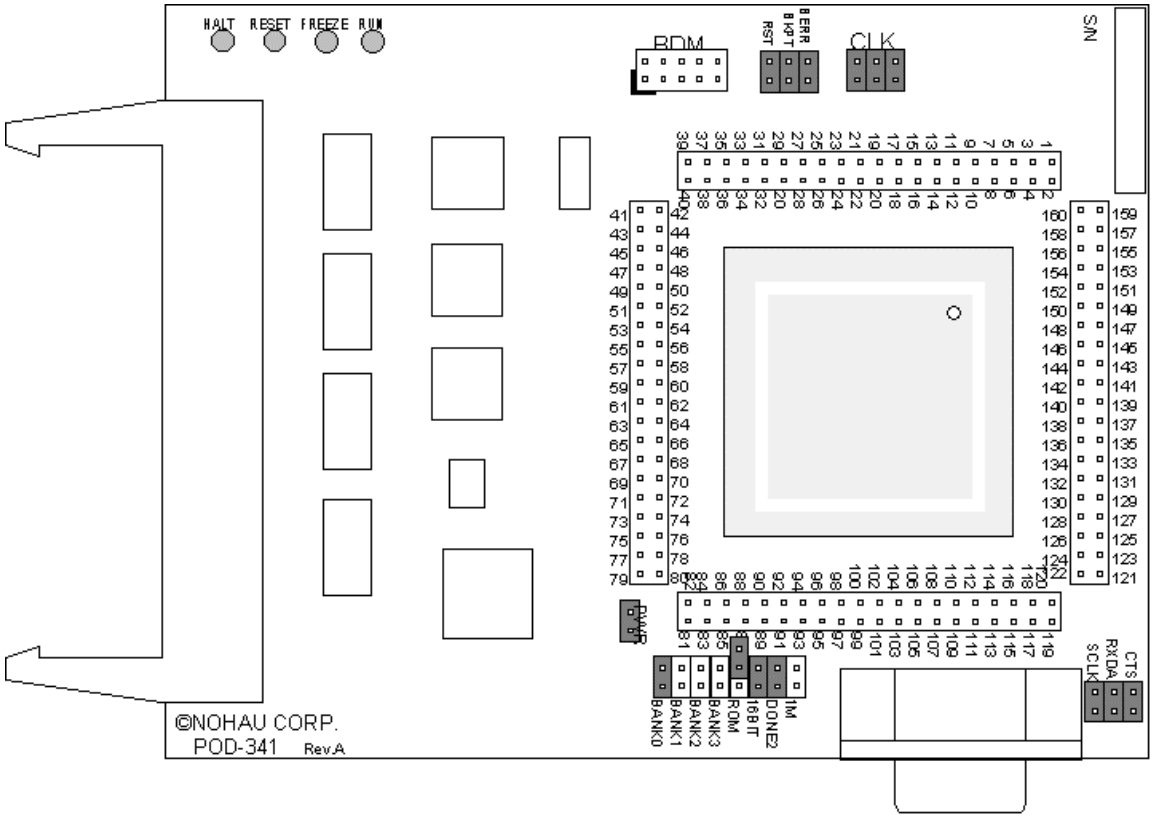


Figure 55. POD–341

Overview

This pod board is used to emulate the Motorola microcontrollers 68341. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

The pod board has a standard 68341 series MCU. You can replace the part in case of failure or if you want to use another part in the same family.

POD–341 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D2	**RESET**	A red LED that lights when the reset to the processor is asserted.
D3	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D4	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.

POD–341 Configuration Options

The "Header and Jumper Details" section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
1M	ON or OFF	ON or OFF	Set at the factory, do not change
DONE2			DMA controller signal allows the trace board to monitor the DONE2 signal
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
BERR	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BERR signal to pod MCU
BKPT	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BKPT signal to pod MCU
RST	On	ON (OFF if external watchdog resets MCU)	Connects target /RST signal to pod MCU
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
CTS			Connects the CTS signal to DB-9 from channel A
RXDA			Drives the serial port input pin on the controller
SCLK	ON		Connects the serial CLK channel A to DB-9

Header and Jumper Details

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

POD–341

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

DONE-2

If a trace board is not present, this pin pair does nothing. If there is a trace board, shorting this pin pair with a jumper will allow the trace board to monitor the DONE2 signal. Without this jumper, the trace board will monitor SHOWCYCLES (generated by pod board) instead.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0 (makes RAM bank0 read-only). The ROM jumper will not effect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper must be off.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

POD–341

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.

2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).

3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

> **Note**
>
> When emulating the single chip mode, remove the /BERR jumper.

CTS

This jumper connects the CTS signal from serial channel A to the DB-9 port.

RXDA

If your target outputs debugging information on the serial port, you might want to connect an RS232 device alike a terminal or a PC to header J1. This pod includes a MAX 232 chip to convert the signal levels from RS232 to TTL levels. Whether or not you connect the RXD pin on J1 to an RS232 device, the MAX232 chip will drive the serial port input pin on the controller. To keep the MAX232 chip from driving the serial input pin on the controller, remove the jumper on the RXD header. To allow the MAX232 chip to drive the serial port input pin, place a jumper on this header.

SCLK

This jumper connects the serial CLK channel A to the DB-9 port.

POD–341

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

Connecting the Pod to the Target Board

There are six ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ADP-144-065-SM is an adapter that solders to 144 pin PQFP SMD pads.

- Part number ADP-340PGA plugs into a pin grad array socket and has pins for plugging into POD–340. IF the target design has a PGA socket, this adapter does not require any special assembly or design. If the target board assembly solders the controller directly to the target board, the PGA socket included with the ADP-340PGA can be soldered onto the target board in place of the controller.

- Part number ADP340-050-SM adapter to solder to 144-pin TQFP.

- Part number ET/CLIP-144-QF10-BC clip-over for 144-pin QFPSMD.

- If designed into the board a 10-wire ribbon cable (BERG connector) designed to allow the pod board to connect to the target board. The cable and connector can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the previously mentioned adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. These two emulator features will not operate when the BERG connector is used.

Note

ISO-160 can be used to isolate the BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The 68341 was designed with only four chip select lines, /CS0, /CS1, /CS2, and /CS3. One of them must be used to control emulation RAM. Choose one chip select line to dedicate to emulation RAM. The best choice is /CS0. Not only is this the one used by the CPU right after a reset,

POD–341

but since emulation RAM is often used in place of the boot ROM, /CS0 is likely to be free to use for emulation RAM. While it is possible to use /CS1, /CS2, or /CS3 to select emulation RAM, it will be difficult unless /CS0 selects a ROM with the reset vectors and startup code. After every reset, you will have to manually set the chip select registers, the stack pointer, and the program counter. On the /CS header, put a jumper on the pair of pins that correspond to the chosen chip select.

68341 Configuration Requirements

The 68341 is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do its job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

Note

ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

The circuitry that maintains Shadow RAM needs /AS, /DS, CLKOUT, SIZ0, and SIZ1. All of these signals must appear at their respective pins or Shadow RAM will not correctly reflect the state of emulation or target RAM. The trace board also needs /AS, /DS, and CLKOUT. Target designs that use these Port E pins for other signals may interfere with Shadow RAM and trace board operation.

Using Emulation RAM

The pod contains either 256K, 1 MB, or 4 MB of RAM that can be used to emulate target RAM or ROM. The first part of the discussion below assumes that the SIZ0 and SIZ1 signals are available to the emulator and have not been configured to carry I/O signals.

The pod with 256K of RAM has 1 bank of RAM, 256K in size. The 1-MB pod board has 4 banks, each bank is 256K in size. The 4-MB pod board has 4 banks, each with 1 MB of RAM. Each bank, regardless of the size, requires one chip select signal. Bank0 is controlled by /CSBOOT, /CS0, /CS1, and /CS2 control banks 1, 2, and 3 respectively. If /CSBOOT, /CS0, /CS1, or /CS2 have to be used by the target, then remove the respective jumper(s) and disable the associated bank of RAM. Chip select signals used to control an emulation memory bank must be configured for reading and writing. If the RAM is emulating 16-bit target RAM or ROM, the signal must also select for both high and low bytes. For emulating 8-bit devices, the signal should only select the high byte.

POD–341 Rev. B has a PAL to control emulation RAM. This PAL uses a variety of controller signals and the jumpers on the pod to determine which RAM chips to enable, and which to write-enable. (Refer to Appendix G, “PAL Equations for RAM.”)

POD–341

All memory banks must be either 8 bits or 16 bits wide. Any chip select can control 1 bank of emulation memory on the pod if a wire is run (wire wrap is best) to the header for that bank (the pin close to the edge). This also works for 16-bit memory without size signals.

Note

To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

Bank0 is special because it is controlled by /CSBOOT. Resetting the controller activates /CSBOOT for reading and writing 1 megabyte of memory starting at address 0. Bank0 is also special, because inserting the ROM jumper will make it impossible for the application to write into bank0 while still allowing the emulator to write breakpoints and load code. As mentioned above, even /CSBOOT must make RAM writeable. Your startup code may make bank0 read-only, especially if /CSBOOT is used to control a ROM. Configuring /CSBOOT to be active during read and write cycles and keeping the ROM jumper in place will allow the emulator to load code and write breakpoints into that bank, but prevent the application from writing to that bank.

Note

To emulate through a reset, bank0 must contain your reset vectors and startup code.

Configured this way, every time the controller is reset, RAM banks 1, 2, and 3 will be disabled. Data windows scrolled to show this RAM will only show asterisks instead. In this state, the emulator will not be able to load code or data into these other banks. Of course, as soon as the application starts, the RAM will be reactivated by the startup instructions. To use the other banks of RAM, the best approach is to use the user defined (below) chip select option. For each chip select register, fill in the values compatible with your application. Then, every time the emulator resets the controller, it will also write the chip select registers, reactivating the other banks of RAM.

When using more than one bank of emulation RAM, the chip select signals must map the bank used to different addresses. If bank0 is mapped from 0 to $3FFFF, no other bank can be mapped to those addresses. The startup code must set the chip select registers /CSBARx so that the size and starting addresses avoid overlap.

POD–341

String WRITE	'(ROM * CSx * RW	:Any write if not ROM emulator
	+ROM * FRZ *CSx * RW)'	:Write only while in BDM
A17A0	= A0 * 16BIT * 1MB	:A0 in 8-bit mode
	+ A17 * 16BIT * 1MB	:A17 in 16-bit mode
	+ A17 * 1MB	:A17 with 512K chips
A19A0	= A0 * 16BIT	:A0 in 8-bit mode
	+ A19 * 16BIT	:A19 in 16-bit mode
A18VCC	= * 1MB	:VCC with 128K chips
	+ A18 * 1MB	:A18 with 512K chips
UWE	= WRITE* 16BIT	:Any write in 8-bit mode
	+ WRITE* 16BIT * A0	:Any even write in 16-bit mode
LWE	= WRITE* A0	:Any odd write
	+ WRITE* SIZ1 * SIZ0* A0	:Any word write
	+ WRITE* SIZ1 * SIZ0* A0	:Even 3-byte write
	+ WRITE* SIZ1 * SIZ0* A0	:Even long write
	+16BIT	:Keep the RAM tri-stated in 8-bit mode

Without SIZx Signals

If the SIZ0 and SIZ1 signals are not available; if their pins have been assigned to carry I/O signals, many emulator features will be affected. Although Shadow RAM and the trace board will not work perfectly, much of emulation RAM will be available, with a little care.

Using 8-bit emulation RAM without the SIZx signals is easy. After pulling both the 16BIT jumper and the SIZx jumper, emulation RAM will still require one chip select per bank, and it must be configured for 8-bit reads and writes, but otherwise, it will work just as if the size signals were available and the SIZx jumper were still in place.

To use 16-bit wide emulation RAM without the SIZx signals, first you must pull the SIZx jumper. This will disable RAM bank1. It cannot be used with the 16-bit jumper in place. Second, configure /CSOR0 to assert /CS0 for all odd address write cycles and only the write cycles. For example, for 0 wait states, the value in /CSOR0 should be 3030. Third, set /CSBAR0 to generate LWE (odd byte writes only) across all of emulation RAM, to cover all emulation RAM banks in use. This means that under these special circumstances, all emulation RAM (banks 0, 2, and/or 3) must be mapped contiguously. It also means that a 4-MB pod will be limited to 1 MB, the maximum address range of one chip select, even though RAM banks 0, 2, and 3, together would provide 3 MB of RAM.

POD–341

Stand-alone Pod With a 16-Bit vs. 8-Bit Emulation Memory

A full discussion of the 68341 configurability and how it affects the address decoding logic design is beyond the scope of this manual. However, some configuration information is necessary at this point. A 68341 may be designed into a target that uses 8-bit or 16-bit memory or I/O devices. In either case, the board design and the chip select register values in the SIM module must agree. Because there is also an emulator in the circuit, the emulator pod jumpers and software configuration must also be consistent with the design.

When reset, the 68341 polls several pins and sets (or clears) corresponding bits in certain registers. Normally, these pins float up to logic 1 during reset but if they are pulled low by external logic, the corresponding bits are cleared. DB-0, which corresponds to bit 0 in /CSPAR0, controls whether the External Bus Interface generates bus cycles and addresses for 8-bit or 16-bit memory. Although all the chip-select registers can be reprogrammed by the initialization software after the reset is complete, bit 0 of /CSPAR0 must be set (or cleared) before the 68341 reads the reset vectors at the end of the reset period. If it is set incorrectly, the 68341 will read the reset vectors incorrectly and begin executing at the wrong address.

Although the pod does support emulating 8-bit ROMs, it does not have the logic to hold DB-0 low during a reset. This is why a stand-alone pod cannot emulate through /RESET while emulating an 8-bit ROM. It is possible to place a diode between pins 68 (the /RESET pin) and 111 (Data Bus pin 0) so that the /RESET signal itself holds DB-0 low during the reset cycle but so that data bus signals do not pull the /RESET line low. This will allow a standalone pod to emulate through a /RESET with 8-bit bus cycles and with the 16BIT jumper removed. If a target is directly connected to the pod and has, the reset logic required to hold DB-0 low, then the diode will not be necessary.

Other Startup Code Suggestions

Resetting the 68341 clears the SHEN (SHow cycles ENable) bits. The pod does not require that either of these bits be set, but they do affect the emulator user interface and how much information the emulator gets, so some explanation is in order. These two bits control bus arbitration. They also control whether or not internal bus cycles are brought to the pins or shown to external devices. If an internal bus cycle occurs, such as a WRITE to the SIMMCR register for example, the 68341 does not need to use the external bus at all. If the internal bus activity remains internal, the emulator will have no way of knowing what happened. Setting either of these bits will show internal bus cycle activity on the external bus pins. These bits are important to the emulator because if the SHEN bits are both cleared, then WRITEs to registers will not be duplicated in Shadow RAM and the trace board will not be able to monitor those bus cycles. When either of these bits are set, the registers that control the peripheral modules will be shadowed, just as with all other RAM.

The 68341 has relatively little RAM and no ROM so relatively few internal bus cycles will be generated. As Motorola produces other chips that have internal ROM and greater internal RAM, the potential for confusion will increase if both of these bits are cleared. If either of these bits are set, internal bus cycles will be shown to the emulator, which will allow the emulator to show this

POD–341

activity to the user. The design of the target will determine which bit should be set (or if both should be set). For more information about bus arbitration and the SHEN bits, refer to a MC68341UM/AD User's Manual.

Timers and the FREEZE Signal

An EMUL16/300–PC generated break suspends the CPU by placing it in BDM. While in BDM, no instructions are executed and the FREEZE signal is asserted. However, while the CPU is suspended, the other modules like the watchdog and periodic interrupt timers can continue to run, which means that an interrupt or a reset can occur while in BDM. The FREEZE bits in most modules are cleared when the 68341 is reset, which allows those modules to run while instruction execution is suspended. Setting the FREEZE bits in all of the module configuration registers (e.g. SIMMCR, and QMCR) in the startup code may simplify debugging by preventing unexpected behavior.

Watchdog Timer

When the 68341 is reset, most internal devices such as the Queued Serial Module are disabled. However, the most significant bit in the SYPCR register, the SWE bit is set. This enables the watchdog timer with a very short timeout period. Unless the target application software was designed to service the watchdog timer, the target software startup code should clear the SWE bit to disable the watchdog timer.

Processor Clock Rate

At reset, the processor registers are set so that the processor multiples the crystal rate by 256. A crystal frequency of 32768 Hz (the frequency of the crystal on POD–341) will generate a CPU clock frequency (CLKOUT signal) of 8.39 MHz. The simplest way to double this frequency up to the maximum clock rate of 16.78 MHz is to set the X bit in the SYNCR register. Keep in mind that changing this clock rate will change some rates, such as the serial port baud rate, and will not change others, such as the period of the Periodic Interrupt Timer timeout period.

POD–376

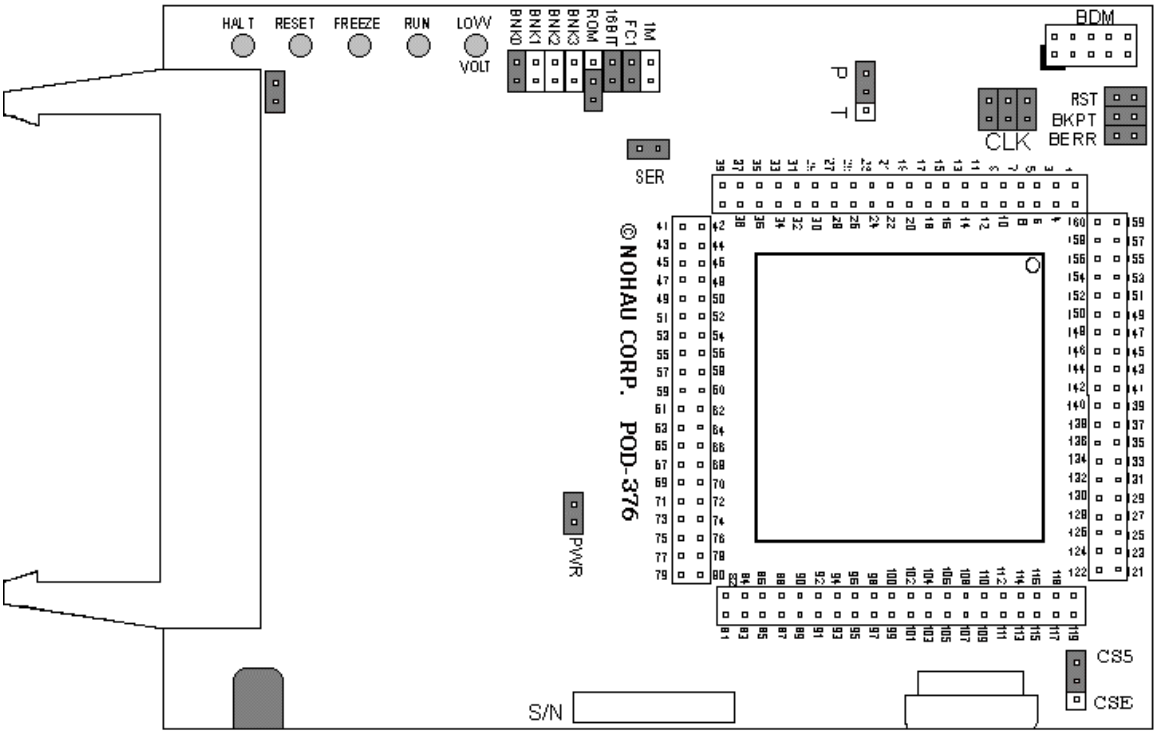


Figure 56. POD–376

Overview

The EMUL16/300–PC pod board contains a 68376 MCU, a 4.192-MHz crystal, between 256K and 4 MB of static RAM for instructions and data, and a DB-9 connector for the chip's serial port. The pod board includes a Motorola standard Background Debug connector, also called a BERG connector, that allows debugging target boards that have similar connectors and do not have room to connect the pod directly to the target processor.

Physical Dimensions

The pod board measures 4.5 inches by 3.75 inches (11.4 cm. by 9.5 cm). The pod requires between one and two inches (2.5 cm to 5 cm) of space above the target.

POD–376 LED Indicators

Name	Function	Description
D1	**HALT**	An amber LED that indicates that the CPU is in a HALT state.
D3	**RESET**	A red LED that lights when the reset to the processor is asserted.
D4	**FREEZE**	A yellow LED that indicates the chip is in BDM mode or has received an error or command to stop. If the yellow LED and the green LED light up at the same time, the pod is probably trying to access nonexistent memory.
D5	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. If the green LED and the yellow LED light up at the same time, the pod is probably trying to access nonexistent memory.
D12	**LOW VOLT**	An amber LED indicates that the pod is running on low voltage (+3V) rather than the standard +5 volts.

POD–376 Configuration Options

The “Header and Jumper Details” section following this table gives a more detailed description of the following options available on the pod.

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
1M	ON or OFF	ON or OFF	Set at the factory, do not change
16BIT	ON	ON (OFF if emulating 8-bit memory	Controls pod memory width
ROM	ON or OFF	ON or OFF	Makes RAM bank0 read-only
BNK0	ON	ON (OFF if running from target ROM)	Connects /CSBOOT to pod RAM bank0
BNK1	ON or OFF	ON or OFF	Connects /CS0 to pod RAM bank1
BNK2	ON or OFF	ON or OFF	Connects /CS1 to pod RAM bank2
BNK3	ON or OFF	ON or OFF	Connects /CS2 to pod RAM bank3
SER	ON or OFF	ON or OFF	Connects the MCU serial port to the DB-9 connector on the pod
PWR	ON	OFF (if there is a target power supply)	Provides +5V power to the target
CLK	ON	OFF	Connects pod crystal circuit to the pod MCU
AS	ON or OFF		Connects target Address Strobe signal to pod MCU.
DS	ON or OFF		Connects target Data Strobe signal to pod MCU.
BKPT	ON or OFF	ON (OFF if target /BKPT is tied directly to +5V)	Connects target /BKPT signal to pod MCU
BERR	ON or OFF	ON (OFF if target /BERR is tied directly to +5V)	Connects target /BERR signal to pod MCU

POD–376 Configuration Options (continued)

Jumper Designation	Stand-Alone Position	Position Connected to a Target	Description
RST	ON	ON (OFF if external watchdog resets MCU)	Connects target /RESET signal to pod MCU
PLL	ON		Jumper is used for enabling/disabling PLL operation (option on Rev. C only)
EXT PWR	OFF	ON (pod will have a power port on it)	Jumper is inserted if the pod is powered from a PC (option on Rev. C only)
CS5/CSE	ON or OFF		Controls which chip select signal is used for selecting the on-board PRU (option on Rev. C only)

Header and Jumper Details

1M

WARNING

This jumper is set at the factory according to the amount of RAM your pod is equipped with, do not change.

16BIT

During reset, pin 0 of the data bus configures the two least significant bits of /CSPAR0, which controls whether the /CSBOOT generates bus cycles and addresses for 8-bit and 16-bit memory. Removing the 16BIT jumper disables half the memory in memory bank0 and configures the memory control PAL for 8-bit memory read and write cycles.

For other banks of emulation RAM, you can correctly configure the chip select registers with the Seehau software, but the MCU tries to read the reset vectors before it is completely out of the reset cycle. If the vectors are stored in an 8-bit wide device and the hardware is configured for a 16-bit wide device or likewise, the vectors fetched will be incorrect and the MCU will attempt to execute from a wrong address.

ROM

Shorting the pins on the ROM header will prevent the application from writing to the RAM on the pod in bank0. The ROM jumper will not affect loading code, editing instructions, or software breakpoints. If your application needs to write to the RAM in bank0 for any reason, including stack or variables, the ROM jumper should be OFF.

BNK0-3

These jumpers only effect pods with more than one bank of RAM. They need to be OFF for targets that use this chip select for anything.

POD–376

For greatest flexibility, remove the jumpers and run a wire wrap from the outside (or even numbered pin) of the BNKx header, to any chip select signal you want to activate emulation RAM. Often this signal will be one of the other chip select pins in the two row headers around the controller on the pod. External address decoding logic may also be used.

To prevent bus collisions, the jumper on any BNKx header must be removed if that header's chip select signal is used to activate a device on the target. For example, if /CS2 is used to control a memory device on the target, you may not use /CS2 to control emulation RAM bank3. You must remove the BNK3 jumper and use some other chip select signal.

SER

This jumper connects the MCU serial port to the DB-9 connector on the pod.

PWR

If this jumper is in place, the emulator safely provides up to 0.5 amps of +5V power from the PC power supply to the target board. Remove it if the target has its own source of power. Seehau can emulate +3V target designs when the PWR jumper is removed, but it cannot provide +3V of power. All +3V target designs must have an external power supply.

> **Note**
>
> This source of power is not fused and may damage the pod and/or emulator if more than 0.5 amps of power is used by the target.

CLK

The jumper in position 1 should remain on if the pod board has a PRU. The position 1 pin can be identified by a white mark on the corner closest to the pin.

> **Note**
>
> Some MCUs are designed to use a 32-kHz input and others are designed to use a 4-MHz crystal. If there is a conflict between the target crystal and the pod MCU, replace the MCU on the pod with one from your inventory that can use the target crystal and remove the three CLK jumpers.

AS

This jumper connects target Address Strobe signal to pod MCU.

DS

This jumper connects target Data Strobe signal to pod MCU.

POD–376

BKPT / BERR / RST

If a target has a reset button that grounds the /RST pin when pressed, this will not interfere with emulation so the /RST jumper should be ON. If the target ties /BKPT pin to +5V and you are using the controller on the pod, the /BKPT jumper should be off so that the emulator can control the /BKPT pin as it needs to. When target /BKPT/BERR is tied to +5V this should be set of OFF when all the following the conditions are met:

1. The target has circuitry that drives this pin.

2. The target circuitry is interfering with emulation (like an external watchdog that does not respond to the FREEZE signal).

3. The P T jumper is in the P position.

Occasionally, a target might contain an external device designed to reset the controller by pulling the /RST pin low. During debugging, that may be inconvenient. The signal from the target /RST pin passes though the /RST header. Removing the /RST jumper will prevent the external device from setting the pod controller.

Note

When emulating the single chip mode, remove the /BERR jumper.

PLL

This jumper is used for enabling/disabling PLL operation. In normal operation, the jumper should be in position closer to capacitor C19 (i.e. between letters P and L on the PLL label). When an external system clock signal is applied, PLL jumper must be in position further from C19 (i.e. between letters L and L on the PLL jumper).

EXT PWR

The external power jumper is inserted if the pod is powered from a PC. It is also possible to power the pod and target from an external power supply connected to the power jack on the pod. The central pin on the power jack is positive. The external power supply must provide +5V DC (+/- 5%). In this case, the external power jumper must be removed. This jumper is located close to the 50-pin connector for the emulator cable and the LEDs.

CS5 / CSE

The /CS5/CSE jumper is used for controlling which chip select signal is used for selecting the on-board Port Replacement Unit (PRU).

POD–376

If the pod is used with a 68331/3332/334 SCIM (Single Chip Integration Module) chip, the jumper must be in /CSE position. The SCIM has a special chip select signal /CSE that permits seamless operation of the MCU and PRU.

If the pod is used with a 68336 SIM (System Integration Module), the jumper must be in the /CS5 position. This causes the /CS5 signal to select the PRU. Alternately, by wire wrapping between the central pin of the jumper and the desired /CSx signal on the 160-pin another /CSx signal may be used.

Note

The following section only applies to pods with more than one bank of emulation RAM.

RAM Bank1 and /CSM

If your pod is equipped with 68376, the MCU will not support /CSM signal. If you wish to use bank1, you will need to run a wire between some other /CS and the outer of the two pins that are now marked BNK1.

Connecting the Pod to the Target Board

There are four ways to connect the pod to the target system. Each method has advantages and disadvantages.

- The most secure and most reliable way to connect the pod to the target is to design a set of headers onto your target board that match the dimensions and pin assignments of the header sockets on the bottom of the pod. With this kind of connection, you may connect and remove the pod from the target many, many times without damaging either the pod or the processor on your target. While the two are connected, the physical and electrical connections will be most reliable possible, and will eliminate the pod connection as a source of emulation problems.

- Part number ET/CLIP-160-QF07-W connects the pins of a surface-mounted MCU to the pins on the pod. This connection is temporary and has the advantage that is does not affect the board design or manufacture at all. With this clip, any production target board connects directly to the pod. The disadvantage of this adaptation approach is that the BKPT pin must not be tied to +5V directly. The BKPT pin must be pulled up by a resistor (10 k Ohms) otherwise the emulator will not be able to assert that signal.

- An adapter plate, part number ET/EPP-160-QF07-B, that solders onto the target board and mates with the pod board pins in place of the MCU. This does not affect the board design, but requires some special assembly.

POD–376

- A 10-wire ribbon cable (BERG connector) designed to allow the pod board to connect to the target board. The cable and connector can reach into small places and can connect the emulator to the target when geometry would prohibit using any of the previously mentioned adapters. The disadvantage of using the BERG connector is that it does not pass any of the signals needed by Shadow RAM and the trace board. These two emulator features will not operate when the BERG connector is used.

68376 Configuration Requirements

Each MCU is very configurable. Nearly every pin that carries a control signal can be configured to carry general-purpose input/output signals instead. However, the emulator needs certain control signals to do the job. The /BKPT/DSCLK line must not be forced either high or low. Instead, a 10-k Ohm pull-up resistor will keep the CPU from accidentally entering BDM.

ISO-160 can be used to isolate the /BKPT (or any other) signal on the target from the pod. This may be simpler than modifying the target board.

Note

To isolate the controller from the target, ISO-160 must come between the target hardware and the controller. This makes the ISO-160 less useful when used with the QFP132CLIP adapter.

EPC BDM Pods

Overview

The ECP version of the emulator consists of a printer port adapter plug with cable, a power supply adapter, and pod with a short ribbon cable ending in a BERG (BDM) connector. The printer port connection is via a pass-through connector, permitting normal printer operation in conjunction with the emulator. The connection is transparent to the printer (the emulator and printer do not interfere with each other during normal operation, and both may remain connected at all times). The power adapter is required to supply power to the pod.

The BERG (BDM) connector is compatible with all of these configurations. When attaching a BERG to an 8-pin BERG connector, insert header pins 1 through 8 into the plug pin sockets 3 through 10, leaving pin sockets 1 and 2 unused. These unused pins are only needed if there is RAM controlled by /CSBOOT, and then only if the reset vector values fetched at the end of the reset cycle point to nonexistent memory.

8-Pin and 10-Pin BERG Connector Definitions

Function	Pin #	Pin #	Function
DS	1	2	BERR
GND	3	4	BKPT/DSCLK
GND	5	6	FREEZE
RESET	7	8	IFETCH/DSI
VDD	9	10	IPIPE/DSO

CPU32 BERG Connector

Function	Pin #	Pin #	Function
GND	1	2	BKPT/DSCLK
GND	3	4	FREEZE
RESET	5	6	IFETCH/DSI
VDD	7	8	IPIPE0/DSO

Early CPU32 BERG Connector

Function	Pin #	Pin #	Function
DS	1	2	BERR
GND	3	4	BKPT/DSCLK
GND	5	6	FREEZE
RESET	7	8	IPIPE1/DSI
VDD	9	10	IPIPE0/DSO

CPU16 BERG Connector

16R1/R3 EPC

There must be only one power source to the system. If the target has its own power supply, external power supply should not be connected to the power adapter, and vice versa. External power supply, if needed, must provide 5V +/- 5%(200mA + current required by the target). If a low voltage target is used, it must have its own power supply. The center of the optional power supply jack is the positive pole and the outer part is the negative pole.

The EPC system consists of an Emulator Parallel Cable (EPC), an EPC pod, a +5V power supply, software and the EMUL16/300 Users Manual. The PC communicates with the pod using the EPC.

The EPC cable is connected to the PC parallel port (usually the printer port). The other end of the cable with the female DB-25 plug is connected to the EPC pod. The other end of the pod is connected to the BERG connector on the target.

It is possible to use the EPC for low voltage battery powered targets, with extremely low power consumption without consuming any power from the battery. In that case, a special version of the EPC pod is required. For details, contact you local representative ("Sales Offices, Representatives and Distributors" list at the end of this guide) or contact Nohau Technical Support at support@icetech.com.

The EPC pod does not support a trace buffer (execution history), data write Shadow RAM, emulation memory or dynamic data exchange.

EPC BDM LED Indicators

Name	Function	Description
D1	**RESET**	A red LED that lights when the reset to the processor is asserted.
D2	**FREEZE**	A yellow LED that indicates output from the chip is not running or in BDM mode.
D3	**RUN**	A green LED that indicates the processor is running and memory cycle is in progress. This LED will stay on as this type of pod is always in BDM mode when running.

Installing the EPC BDM

1. Turn off the power to the computer.

2. Disconnect the printer cable from the parallel port connector if present.

3. Connect the EPC adapter (double ended DB-25) connector to the computer parallel port (DB-25 male end).

4. Reconnect the printer cable removed in Step 2 to the EPC adapter (DB-25 female end of the connector in Step 3).

5. Connect the EPC pod to the other end of the EPC cable (single DB-25 female connector).

6. Connect the power supply to the EPC pod.

7. Connect the power supply to the power strip. We recommend that the power supply be plugged into the same power strip as the computer so that they power on and off at the same time.

8. Follow the instructions below for connecting the pod to your target.

Connecting the Pod to the Target Board

The pod consists of a DB-25 connector with a short ribbon cable terminated in a BERG connector. The BERG connector is not keyed, use caution when connecting to your target. Wire 1 on the ribbon cable is the red wire. The socket for pin 1 on the plug is on the same side of the connector as wire 1 (the red wire) on the ribbon cable.

If the header on the target is an older design and has 8 pins, insert the header pins 1 through 8 into plug sockets 3 through 10 (leave pins 1 and 2 uncovered). Pins 1 and 2 are only needed if there is RAM controlled by /CSBOOT, and then only if the reset vector values fetched at the end of the reset cycle point to nonexistent memory.

If the target header has 10 pins, align the emulator plug so all pins have a socket that they are inserted into. Make sure that pin 1 is inserted into socket 1 (pin 1 on the board usually has a white area on the corner next to it). The easiest way to do this is to attach the BERG connector to the board with the red wire as close as possible to the white corner of the BERG pins.

The yellow wire from the pod is for connection to the CLKOUT signal from your target if available.

Target Boards

In support of the pods, we make small, simple target boards for most of the supported controllers. These boards are inexpensive but have all the basic components to make useful for application prototyping or software development. All target boards have the controller, 32K of RAM, an oscillator, a BERG connector, and a prototyping or wire wrap area. All can accommodate larger RAM if desired.

The 3-inch by 5-inch wire wrap area is for experiments and prototyping. The wire wrap area has a 5-volt (Vcc) bus along one side and a ground (GND) bus along the other. All boards have a Vcc plane and a ground plane (even in the wire wrap area) which provides excellent noise suppression.

The pads for the RAM chip are designed to accommodate higher capacity chips (you must supply your own chips). The pads are not wired for 512K-by-8 chips, but 128K-by-8 chips can be soldered directly into place. If you wish to use the 512K-by-8 chips, please call Nohau Technical Support for assistance in adding a wire to the target board.

6 Starting the Emulator and Seehau Software

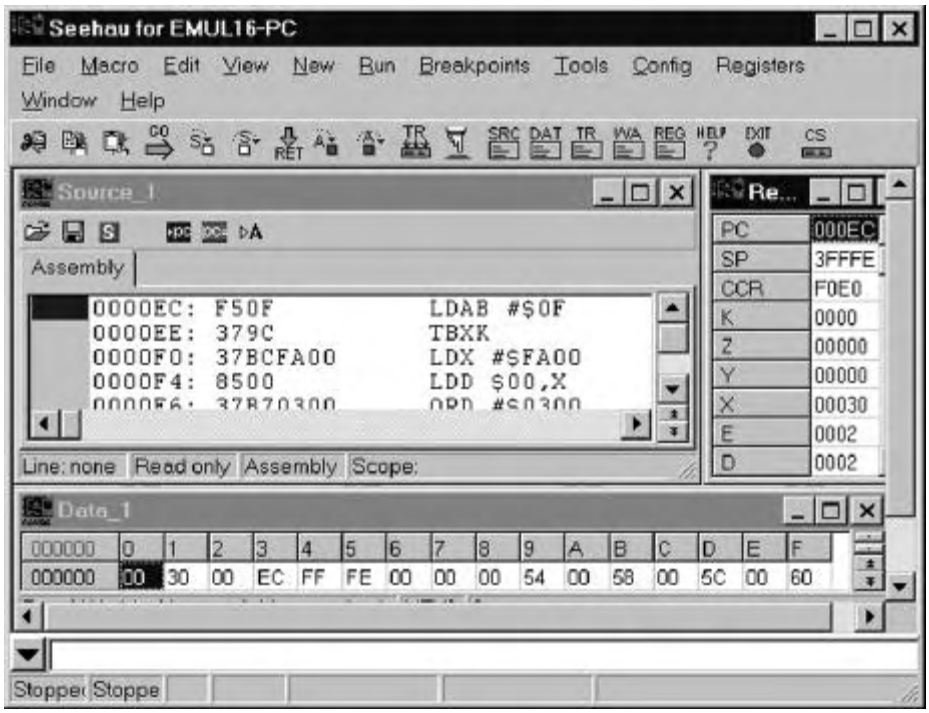


Figure 57. Seehau for EMUL16/300–PC

Verify the Full Function pod is configured for stand-alone mode (not connected to the target board.) Make sure the power jumper is inserted and the crystal jumpers are set for internal crystal.

Starting Seehau

To start the Seehau software, do the following:

1. Double-click the Seehau16 icon on your desktop. (See Step 3 in the "Installing the Software" section in Chapter 2, "Installing and Configuring the Seehau Software" in this guide.) Seehau will load its configuration from the Startup.bas file. Notice that **Macro** is displayed in red at the bottom of the main window while Startup.bas is running.

2. When the software startup is complete, you can position and resize the main window to your preference. Figure 57 shows how the Seehau software will appear when first started up.

3. To open new windows, go to the **New** menu and click a window type.

For further information, refer to Appendix D, "Troubleshooting Tips" in this guide, contact Nohau Technical Support as support@icetech.com, or refer to the "Sales Offices, Representatives and Distributors" list at the end of this guide.

7 Time Program Examples

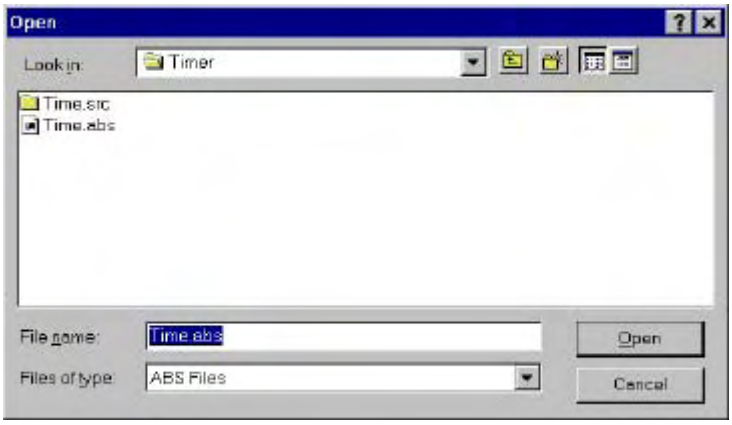


Figure 58. Loading Code

We provide a small example program called Time.abs that is found in the C:\Nohau\Seehau16\Examples\Examples\Timer default subdirectory.

Start the emulator following the instructions in Chapter 6, “Starting the Emulator and Seehau Software” in this guide.

1. Resize the windows on your screen, but do not add the Trace or Watch windows.

2. From the **Seehau File** menu, click **Load Code**. The **Open** dialog box appears.

3. Four folders will appear. Double click the Examples folder. You will have a folder and a file appear (Figure 58).

4. Highlight the Time.abs file for your pod and click **Open**. You can also double-click the file name and it will load into the emulator.

5. Click the Source Step Into button on the toolbar and the program will run to the start of MAIN.

   Notice that the Time.c tab appears on the Source window. You can easily switch between assembly and source language by clicking these tabs.

6. Right-click in the Source window with the Time.c tab selected. Select **Mixed Mode** and select **At PC Line**. You will see the assembly code mixed in with the appropriate source lines (Figure 59). Notice the program counter indicated by the blue blocks at the start of MAIN. You can see the program counter value in the Register window.

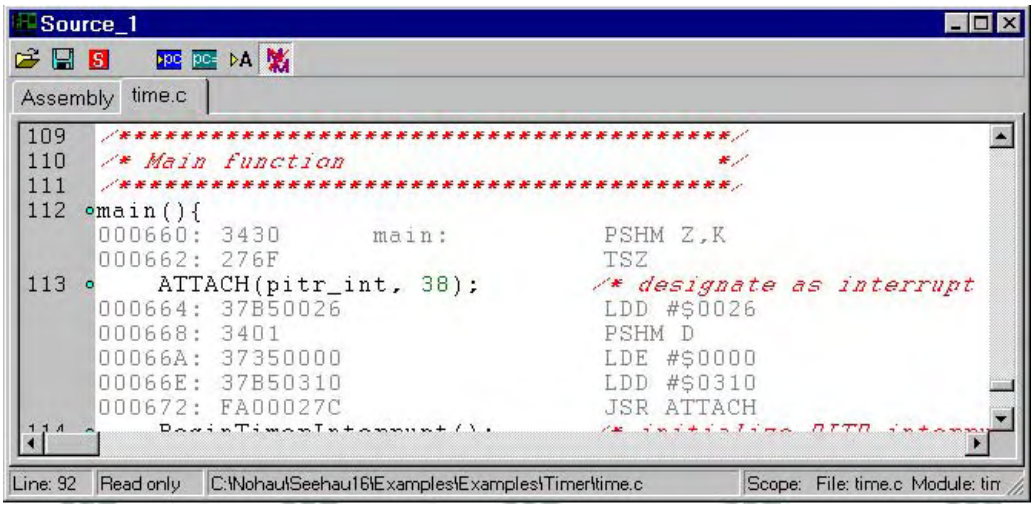


Figure 59. Time Program

7. Remove Mixed Mode from the Source window so only the C source code remains. To remove Mixed Mode, right-click in the Source window and select **Mixed Mode**.

8. Click the Source Step Into button repeatedly and the program counter will advance through the CPU initialization code. Where there is assembly code only, the steps are done at source level.

9. Click the Assembly Step Into button repeatedly and the program counter will advance through the assembly level code.

> **Note**
>
> You might see several program counter indicators (shown as blue rectangles) in the left column of the Source window. The program counter indicators provide line number information supplied by compiler manufacturers. The indicators show that several source lines have the same address. To verify this, go to the **View** menu and click **Symbol Browser**.

Watching Data in Real-Time with Shadow RAM

The EMUL16/300–PC emulator board contains either 64K, 256K, or 1 MB of Shadow RAM used to duplicate the contents of the target RAM. Every time the CPU generates a WRITE bus cycle, the pod captures the address/data pair and the emulator board writes that data to the same address in Shadow RAM. The Seehau application can simultaneously read Shadow RAM. This allows the software to display values written by the application without interrupting emulation.

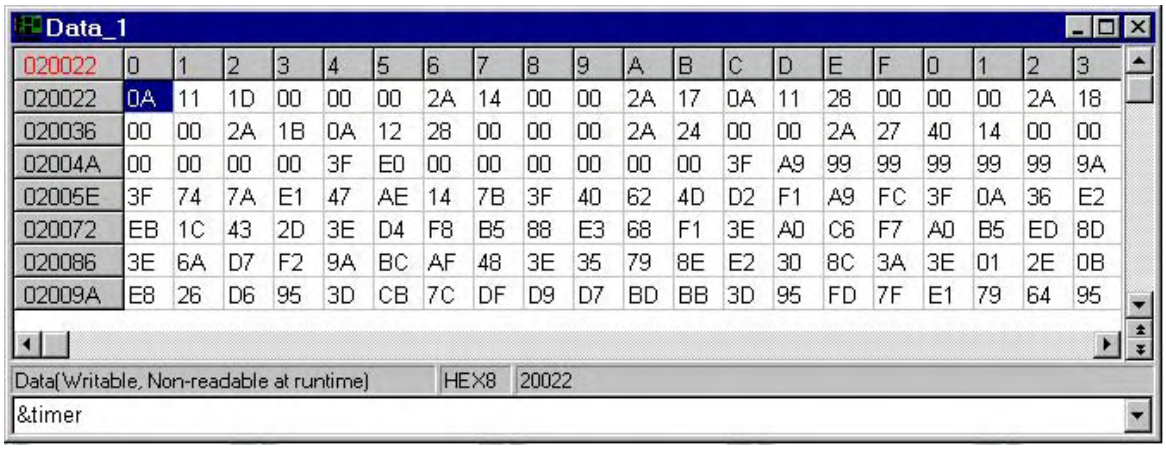


Figure 60. Data Window

The Shadow RAM feature allows you to view memory contents in real-time without stealing cycles from the emulation CPU. The following example assumes you have completed all the steps so far in this guide and that Time.abs is still loaded in your emulator.

To open a Data window, click the Data button or from the **New** menu, click **Data**. The Data window appears (Figure 60).

1. The data will be in hexadecimal as shown. Resize the window as needed.

2. In the address box at the bottom, highlight the existing address and type:
   `&timer`
   Then press ENTER.

> **Note**
>
> From the Data window, the number in red in the top left corner indicates the address of the currently selected location in this window.

3. Right-click the Data window. The **Data Window** menu appears (Figure 61).

4. To change the Data Display mode, right-click the Data window and select **Display As**. The **Format** dialog box appears (Figure 62).

5. Select an appropriate mode (for example, **ASCII**). Select **OK**.

6. Click the GO button or press the F9 key. The program Time.c will run. The time will be updated in real-time. No CPU cycles are stolen to accomplish this.

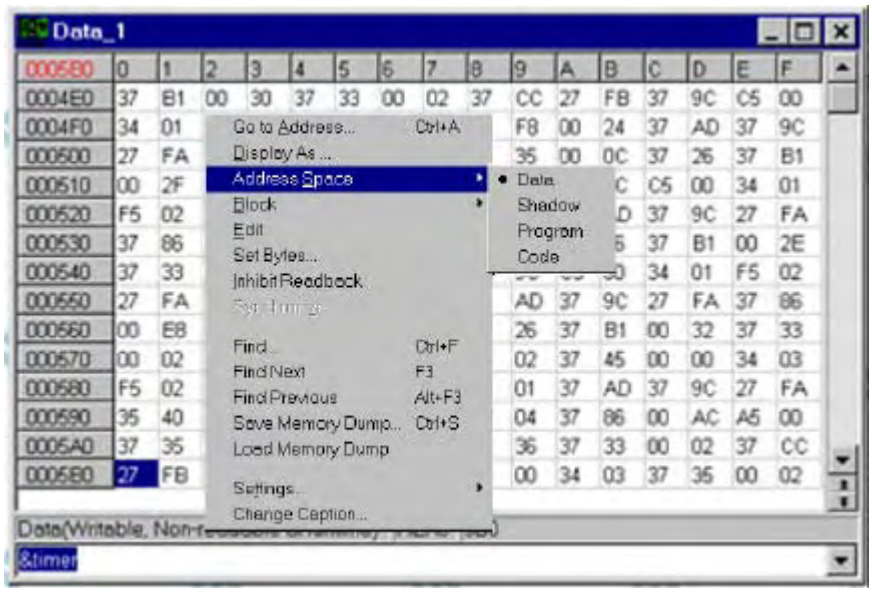


Figure 61. Data Window Menu

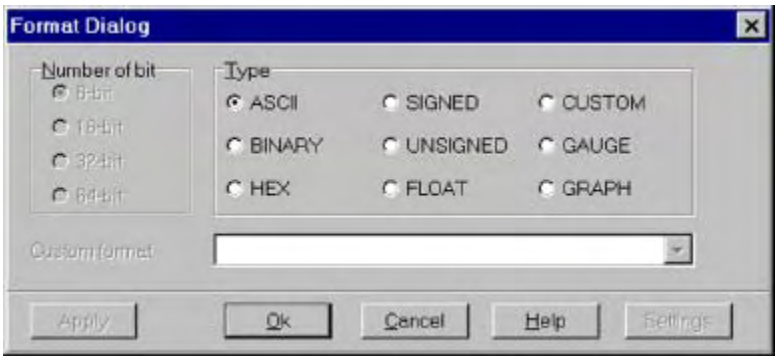


Figure 62. Format Dialog Box

Saving the Hardware Configuration

1. To save the emulator configuration, click the **Configurations** menu and select **Emulator**.
2. From the **Marco** menu, click **Start Recording**. The Configuration window will disappear.
3. Reopen the Configuration window from the appropriate menu item. Click **Apply**. The settings associated with this window will be saved.
4. From the **Macro** menu, select **Stop Recording**.
5. The **Save Settings** dialog box opens where you can choose the filename for the newly created macro. Enter a filename of your choosing and click **Save**.

The macro is ready to use and will be accurately recreate your emulator configuration settings. Configuration settings are also saved when general Seehau settings are saved.

8 Shutting Down Seehau Software

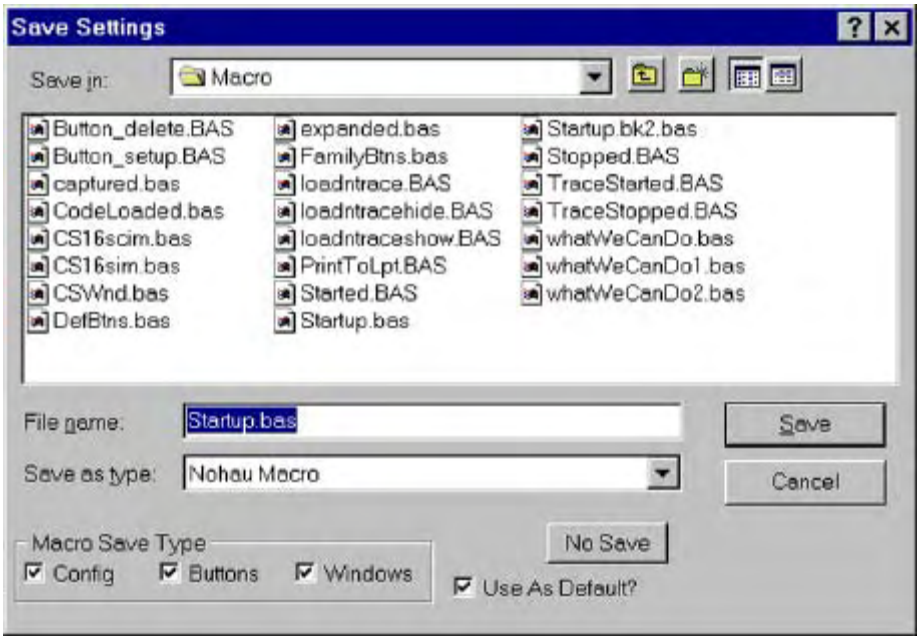


Figure 63. Save Settings Dialog Box

To shut down Seehau, do the following:

1. Click the X (Close button) at the far upper right corner of the title bar or from the **File** menu, select **Exit**. The **Save Settings** dialog box opens (Figure 63).

2. To save your settings, in the **File name** text box enter Startup.bas or another macro file name.

3. Select the **Use as Default** option in the lower right of the dialog box. When Seehau starts, it will use Startup.bas or the macro file you entered in the **File name** text box.

4. Under **Macro Save Type** in the lower portion of the dialog box, select **Config**, **Buttons**, or **Windows**. The items you select will be saved in the macro indicated by the file name.

5. Select **Save** and exit from Seehau. (If you do not want to save your settings, select **No Save**.)

Important Software and Hardware Notes

Do not simply delete the Seehau files or folders. Always use Uninstall to unload any existing version of Seehau from your computer. Do not install Seehau on top of an existing copy! Save your personal macros and source files into another directory. Make sure you delete the old copy of Startup.bas and other remaining files by verifying that the Seehau software was deleted.

To use **Uninstall**, do the following:

1. In My Computer, double-click **Control Panel**.
2. Double-click the Add/Remove Programs icon.
3. The Install/Uninstall window will open.
4. Scroll through the list of programs and highlight **Seehau16**.
5. Click **Add/Remove** to start the Uninstall Shield for removing the software.

> **Note**
>
> You may get a message asking if you want to delete certain files. Some files can be deleted without affecting other software. If you are uncertain whether deleting these files will affect other software reply **No!**

9 Introduction to Tracing

A trace history is a time ordered recording of bus cycles. Events that do not affect the CPU external bus, such as testing a CPU internal data register, are not recorded. Events that do affect the bus will get recorded if the recorder is turned on and set up to record those types of events.

Tracing emulators record bus events and not actual instruction execution. Seehau's trace board for EMUL16/300–PC includes pipeline decoding and marks opcode fetches that are not executed. Consequently, the display software can show the trace records as though the pipeline did not exist. The software can also display the uncorrected bus cycles just as they were recorded.

Starting Trace

Tracing starts automatically every time emulation starts. Executing a single-step turns on the trace recording only during that step.

To stop recording, click the red-colored Stop Trace button.

Until emulation starts, there will be nothing to record. Once trace recording has started, the Start Trace button changes to a red-colored Stop Trace button and allows you to stop recording when clicked. The trace buffer continues to collect records until recording stops.

Recording is stopped by one of the following methods:

- A trigger
- By stopping emulation
- By clicking the Stop Trace button.

One of the trace controls is the filter found in the **Filter** field of the **Trace Setup** dialog box. The filter allows you to set up conditions for recording. When bus cycles are being recorded, every cycle is examined to see it meets the conditions in the **Filter** field. Bus cycles that are not the correct type or that fall outside the address range(s) specified in the Filter field will not be added to the buffer.

The trace buffer is a ring buffer that continues to collect new records and replace old records until recording is stopped. When tracing starts, the buffer is cleared. After recording a single step, the trace buffer contains only the records for that one instruction or source line. As long as trace recording continues, records are added to the buffer.

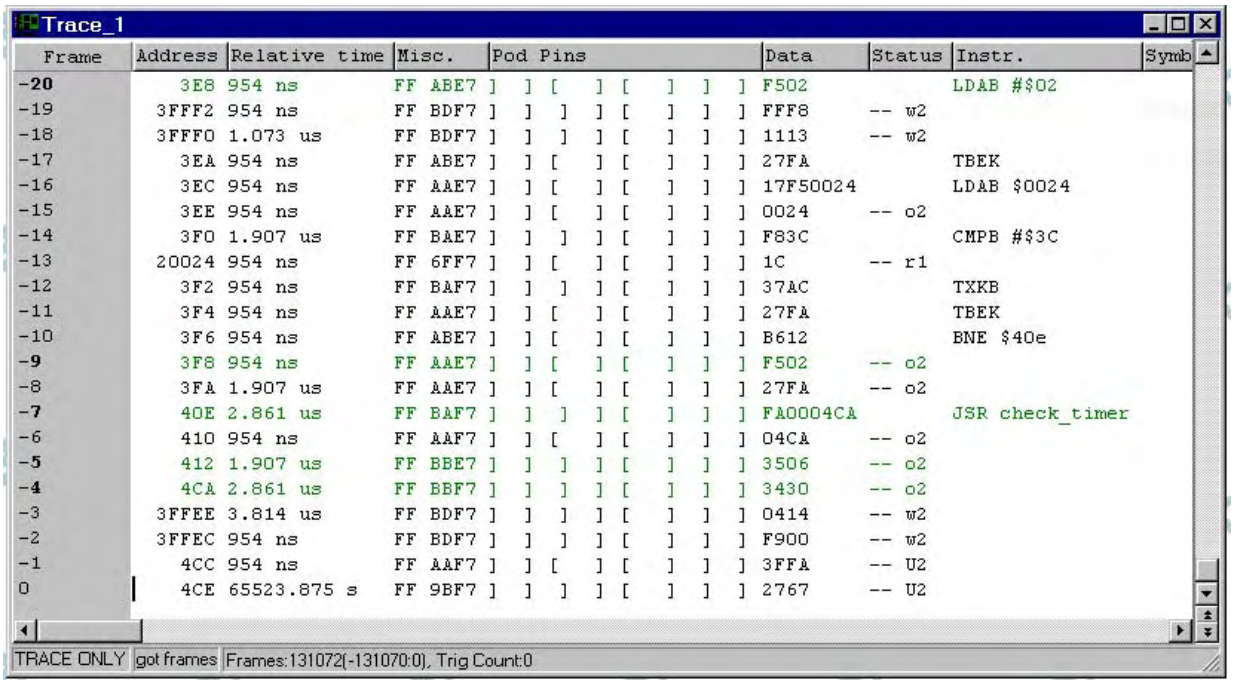


Figure 64. Trace Window (Non-compressed)

> **Note**
>
> All buffers are limited in size. If left recording too long the ring or circular buffer will eventually fill. Once the buffer is full, the newest records begin to overwrite the oldest records.

Triggers

The trace board use triggers to stop trace buffer recording and can be configured to stop emulation. A trigger occurs when certain conditions on the bus are met. For example, you might set up to trigger when the instruction at 4FEH has been fetched for the fourth time. These are useful if you are executing out of ROM. For additional information about how to create triggers, refer to Chapter 10, “Trace Memory Example” in this guide.

Trace Window

The contents of the trace buffer are displayed in the Trace window (Figure 64). To open a Trace window, use the TR button on the toolbar.

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

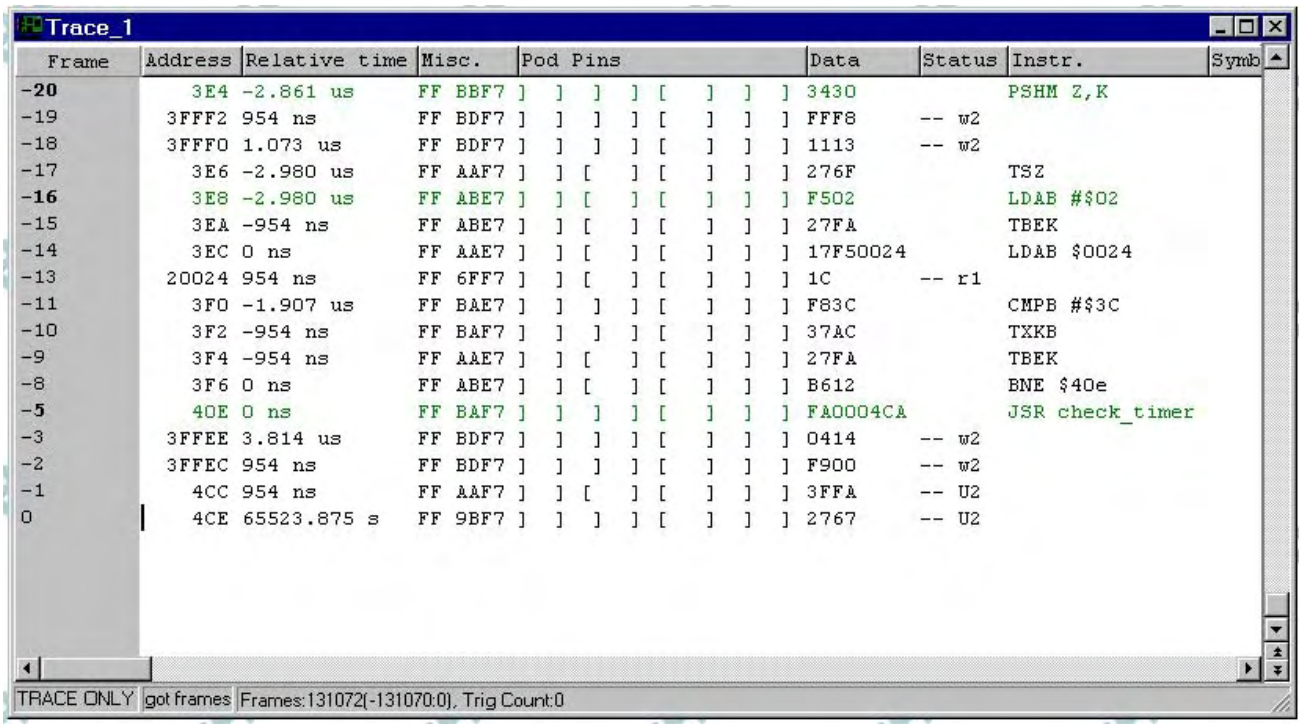


Figure 65. Trace Window (Compressed)

Description of Trace Window Columns

Frame 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.
- **Relative Time** displays the amount of time it takes for one bus cycle.
- **Misc.** is all the other information that is gathered that may prove to be useful.
- **Pod Pins** shows the locations and positions of the pins on the board.
- **Data** displays bytes of data from the displayed bus cycle.
- **Status** displays the status of the W1 and W2 trace board jumpers.
- **Instr.** displays the instruction disassembly.
- **Symbol** displays any symbolic label that refers to a specific address.

Accessing the Emulator and Trace Setup Windows

To access the **Emulator Configuration** dialog box, go to the **Config** menu and select **Emulator**. The **Emulator Configuration** dialog box appears (Figure 27).

To setup the trace configuration, go to the **Config** menu and select **Trace**. The **Trace Configuration** dialog box appears (Figure 36).

Trace Triggering and Trace Filtering

A trigger is an event that can be set up to occur each time that the trace recording is started. This allows you to set conditions on the controller bus. When this condition is met, a trigger will occur. Each machine cycle of the 68HC16xx/(9)16xx microprocessors presents a different address, data, and port information to the trace board. The information from one such cycle is called a frame. Triggers can start and stop trace buffer recording. For information on how to create triggers, see Chapter 9, “Introduction to Tracing.”

Technically, there are three different triggering types:

- Electrical—Input and output of oscilloscopes.
- An event recognized by the trace logic as contrasted to filtering.
- An event that is recognized by the trace logic that is part of a chain of events that can ultimately stop the trace recording. This is the most widely used aspect of triggering that we use with the Seehau software.

There are two ways to set up triggers and filtering: Normal mode and Window mode.

In Normal mode, more control is given to triggers. A trigger in Normal mode either stops recording, starts recording, or starts the countdown until recording will be stopped. Frame 0 is always the frame where the trigger occurs.

In Window mode you get a different kind of control over what cycles are recorded and can selectively record program threads in a way that record filtering cannot. Once recording has started, the conditions in the filter decide whether to record a bus cycle or not. More control is given to controlling which bus cycles are recorded.

Trace filtering is a set of conditions that determine which frames are allowed into the trace buffer.

10 Trace Memory Example

This chapter describes the trace memory including how to set up a trigger to start and stop the trace memory and how to stop program execution. However, do not change any settings in the software. You will need your present settings to continue.

> **Note**
>
> You must have the optional trace board to complete this chapter.

The Time Program example in Chapter 7, “Time Program Examples” showed the Shadow RAM providing memory contents in real-time. Recall there is no cycle stealing from the emulation controller. While the Time.c program has been running, the trace memory has been operating in the background in real-time.

Make sure the emulator is running the Time.c program. The two boxes in the bottom left-hand corner of the main window will contain **Running**. The Go and Trace buttons on the toolbar are red in color.

1. To open a Trace window, click the **New Trace** button, or from the **New** menu, click **Trace**.

2. Position the windows so you have the Trace and Data windows visible. The Trace window can have some data recorded or be empty. This depends on previous emulation runs.

The Trace window is empty as the trace buffer is being filled. It is not possible to view the trace contents at this time. The status bar at the bottom of the Trace window shows that the trace memory is already full and how many trigger events have occurred. The trigger event count is 0.

Stop the trace by selecting the Stop Trace button. The Trace window now contains recorded controller cycles.

You can add some columns by right clicking the Trace window and selecting them or by selecting **All Options** (Figure 66).

When you select **All Options**, the **Trace Options Summary** menu appears (Figure 67). You can select or clear any or all options and make a single update.

Registers and addressing modes are all displayed. The Trace window has the ability to display C source code with the resulting assembly code.

Start the trace memory recording by selecting the Start Trace button. The time displayed in the Data window does not stop or slow. The trace memory is a circular buffer and is being continuously overwritten with new values. This will continue until the recording is stopped manually or by a trigger event.

Figure 66. Local Trace Menu

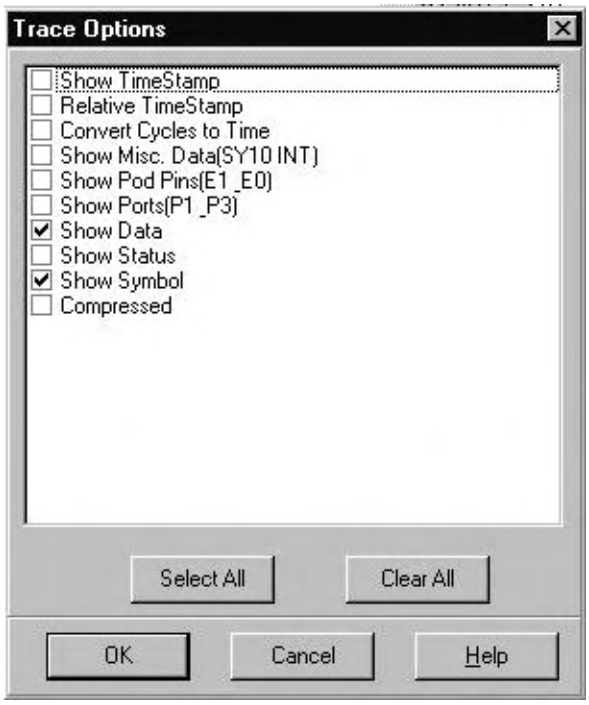


Figure 67. Trace Options Summary Menu

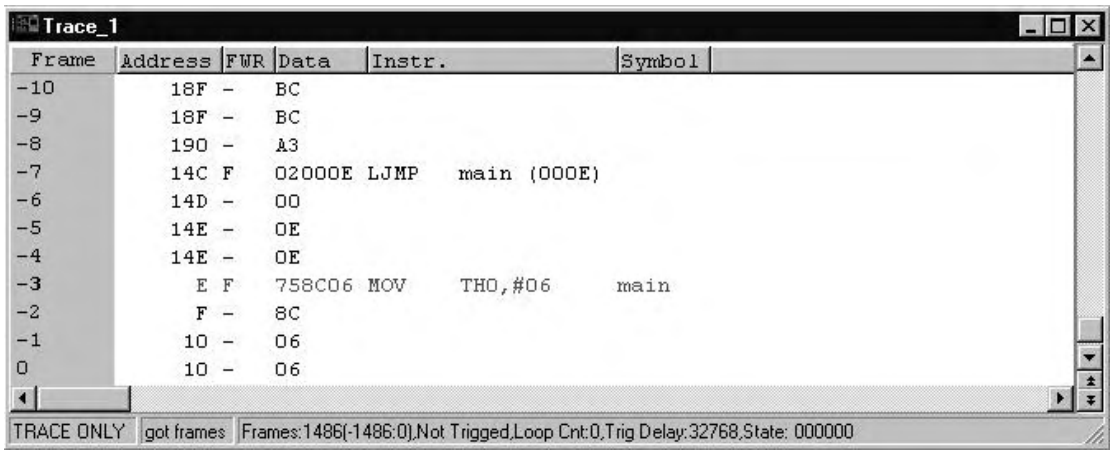


Figure 68. Trace Display Window

The trace can record various bus cycles (Figure 68). Right-click the Trace window or from the **Config** menu, click **Trace**. Refer to Chapter 4, “Installing and Configuring the Trace Board” in this guide.

11 Macro Examples

Macro commands control all interactions with the emulator. The graphical user interface uses the same commands as those that you type in a macro file. This ensures you can control all emulator functions from your script or from the command line.

Seehau Commands

Seehau commands allow you to control the Seehau user interface in two ways:

- The command line, or
- By typing in a sequence of commands in a macro file.

You can run the macro file by clicking the **Macro** menu and selecting **Run**.

When Seehau starts up, it will look for the file Startup.bas. This file is a script file containing Seehau macro commands. When this macro executes, the commands in the file will be processed in sequential order. The sequence of commands sets up the emulator hardware and configures the graphical user interface.

For an introduction to what a macro file looks like, from the **Macro** menu, click **Open**. The **Open** dialog box appears. Now click the **Captured.bas** file. You can now see what a typical macro looks like.

The Seehau commands are an extension to the Seehau Basic scripting language. This Seehau language is a largely compatible subset of Microsoft's Visual Basic language. If you are familiar with the Visual Basic language, you should be able to put a script together using the same logic. If you have trouble with the Seehau macro language, you can select **Help** in the **Macro** dialog box for additional information. The scripting language allows you to write basic code using IF, WHILE, and FOR statements.

Description of the Macro Window

The Macro window gives you full visibility of the macrocode. As shown in Figure 69, the Macro window provides you with a full development and debug environment. In the second row of the toolbar you can see the Step Into and Step Over buttons. As you single-step through your code, selecting the **Debug** menu and clicking **Quick Watch** can inspect variables.

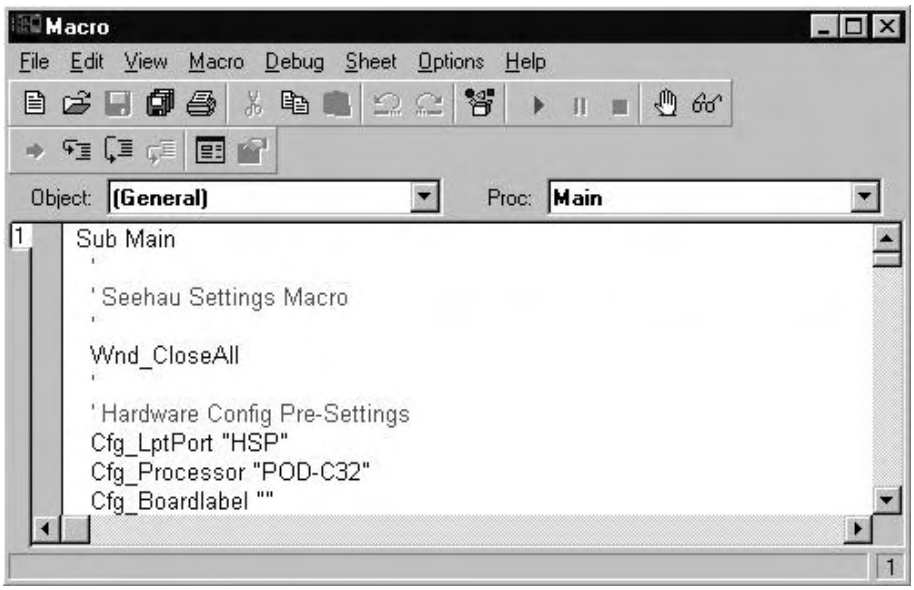


Figure 69. Seehau Macro Editor

Macro Construction

To create a macro, click the **Macro** menu and select **Start Recording**. You can now perform tasks in the user interface, for example, clicking a Single Step button. This action is now recorded in the macro as follows:

```
Run StepInto
```

Select the **Macro** menu and click **Stop Recording**. The **File Save** dialog box appears and prompts you for the macro file name of the macro you just recorded.

You can type in the macro commands with any editor in a file ending with an extension .bas. Any such macro can then be executed by clicking the **Macro** menu and selecting **Run**. You can also type the macro name in the command line text box and press ENTER.

Macro Execution

When Seehau starts up, it executes the Startup.bas file. You can specify other macros that will be executed at emulator events such as starting emulation, stopping emulation, starting trace, or stopping trace. Specify these macro names from the **Config** menu. Click **Environment**, and the **Environment Configuration** dialog box appears. Then click the **Macro** tab (Figure 70).

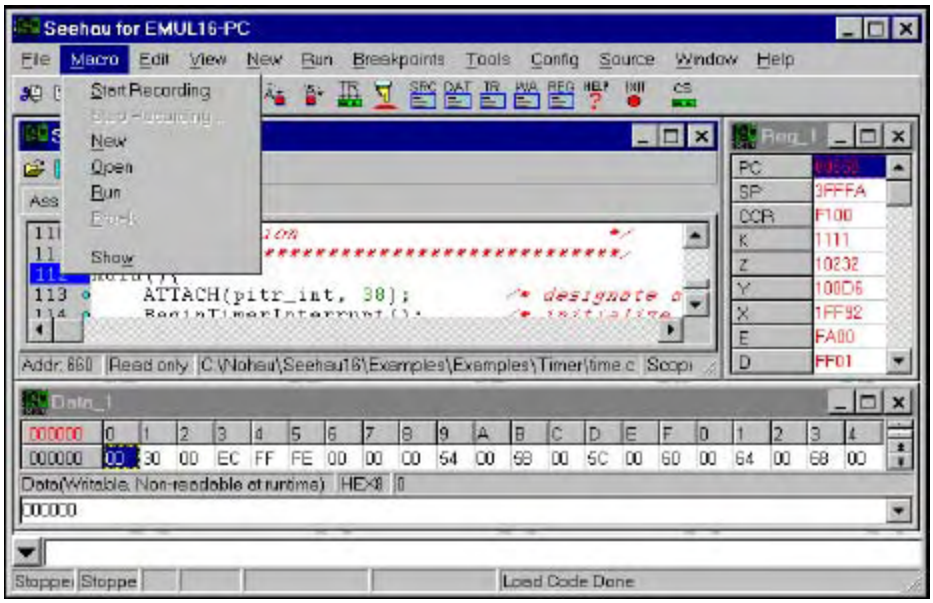


Figure 70. Macro Menu

Individual commands execute as you enter them on the Seehau command line and press ENTER. Figure 70 illustrates the command line Edit field at the bottom of the main window. Click the down arrow by the Edit field to see a command line history. By double-clicking a command in the history list, the command appears on the command line. You can view the result of the individual command that executed in the status bar located at the bottom of the main window (Figure 70).

Command Format

Commands are of the form *groupname_commandname* fields. The commands are in ASCII text. Commands are written in upper and lower case letters for easy readability, but Seehau is case insensitive. An example of a command is: Cfg_ResetDelay “100.” This command tells the emulator to use a 100 ms delay after reset.

Command Examples

The following is an example of directly entering a command:

In the dialog entry box type: `Wnd_Height “300”` and press RETURN. The open window or the one most recently selected by a macro command will have its height adjusted to 300 pixels. To modify the main window, you might have to type in this command first: `Wnd_Select`.

This example uses the EMUL16/300 emulator for the MCS 16/300 family. Enter the appropriate fields for your system as indicated in the caption on the main window.

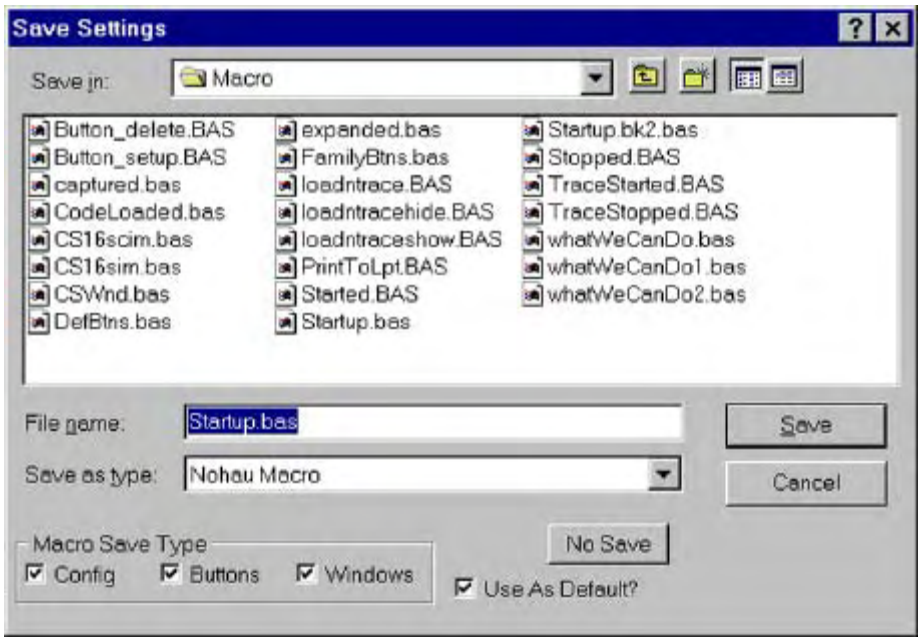


Figure 71. Save Settings Dialog Box

If you enter these commands in the file Startup.bas, they will be executed every time you start up Seehau. You can also save your own macro with the Seehau configuration setup. You are given this opportunity when you exit Seehau. Figure 71 shows the options you can select:

- **Config**: Saves hardware configuration items such as breakpoints and triggers.
- **Buttons**: Saves the position and type of buttons displayed on the main window.
- **Windows**: Determines the position and size of the windows displayed.
- **Use as Default**: The specified file will be the one used by Seehau on startup. The file Startup.bas is the initial default. If Startup.bas is not present, Seehau will create one from a composite of existing .bas files. You can specify any macro file name as the default Startup.bas.

Command Groups

Macro commands are divided into 12 groups. Each group represents one basic function. The group is specified in the first part of the command name as in **Brkpt**_ClrBrkPoints. Seehau Help files contain detailed information of all the macro commands. Consult Seehau Help for details on the fields associated with the commands and for more examples.

BrkPT: These commands are associated with breakpoints. For example, Brkpt_ClrBrkPoints clears all the breakpoints.

Cfg: General emulator configuration settings. For example, Cfg_Map maps emulator memory to the target.

Data: Operations on data such as move, search and set. For example, Data_Move moves data from a source to a destination address.

File: Operations on files such as symbol table and user code loads into the emulator or target memory. For example, File_LdCode filename loads the specified user object code into the emulator.

Hlp: Help commands. Displays Help on various subjects from the Seehau Help file.

Src: Program commands. These refer to the user code in the Source window. For example, Src_SelectModule selects the source module to be displayed in the Source window.

Reg: CPU register commands are used to inform Seehau of the name, size, and address of the CPU registers. These commands are used in operating Seehau and to display registers in the Register window. For example, Reg_Size “16” informs Seehau the previously selected register is 16 bits wide.

Run: Commands used to start the emulation CPU. Commands include Run_Break, Run_Stepover, and Run_GoForever.

Src: Used to access commands regarding the source code.

Trace: Commands that control the hardware trace functions. For example, Trace_TraceBegin is used to start the trace.

View: Commands that control the Inspect, Evaluate and Watch windows. For example, View_Evaluate passes the expression to Seehau it is to evaluate.

Wnd: Windows commands used to control aspects of the windows including position and size, button definition and show/hide windows.

Quick Saving of the Hardware Configuration

Due to the instability of PCs and operating systems, it is important to take precautions after setting up your hardware and software. Rather than wait until you have finished doing your tests on the target system you may want to save the emulator settings to avoid unnecessary repetition in case of system failure. The quick and easy way to avoid this problem is to follow the procedure below:

1. To save the emulator configuration, click the **Config** option and select **Environment**.

2. From the **Environment** menu (Figure 72), check the **Use Startup Dialog** under the **Preferences** tab. This check box will be found under the **Miscellaneous** section.

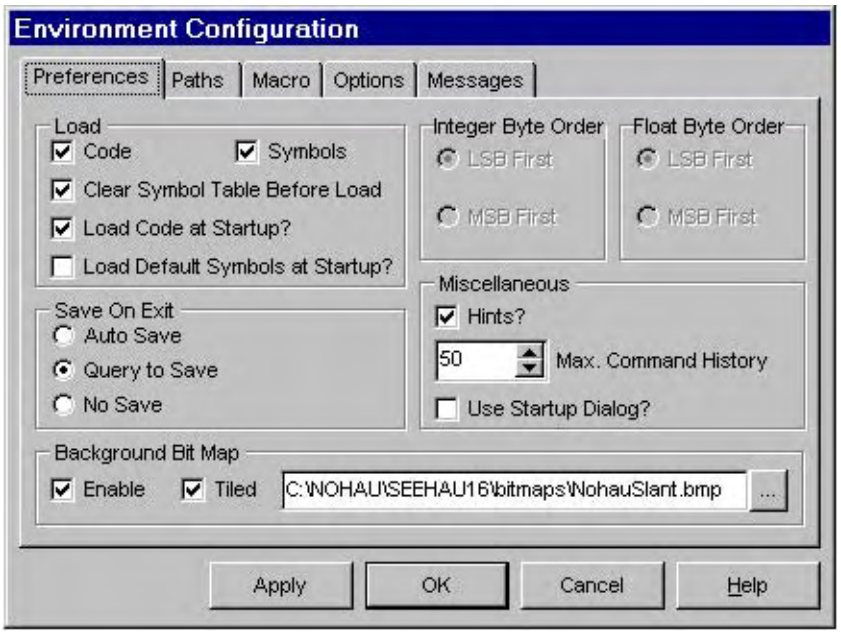


Figure 72. Environment Configuration Menu

3. Select **Apply** or **OK**. The **Environment Configuration** dialog box will close.

4. Exit from the Seehau software. The **Save Settings** dialog box opens where you can choose the filename for the newly created macro (Figure 71). Enter a filename of your choosing and click **Save**.

The macro is ready to use and will accurately recreate your emulator configuration settings. Configuration settings are also saved when general Seehau settings are saved.

Creating New Buttons

Buttons can be created and deleted on the icon bar in Seehau's main window. You can create a button that is attached to a macro. The icons are Windows bitmaps (*.bmp) and created with virtually any graphics program.

To create a new button, do the following:

1. Open the **Customize Buttons** dialog box (Figure 73) from Seehau's main window. Right-click the toolbar, then click **Buttons**. Or from the **Config** menu, select **Buttons**.

2. From the **Customize Buttons** dialog box, open the bitmap browser by selecting the button to the right of the **Bitmap** text field. The **Select Glyph** dialog box appears, displaying the contents of the bitmap directory (Figure 74).

3. Highlight the file Button W.bmp. Select **Open** and the image appears on the right of the dialog box (Figure 74).

4. In the **Hint** text field, type in a sentence as shown in the example (Figure 75). Then position the cursor on the W icon. The sentence appears in a yellow box.

5. Under **Command Type** (Figure 75), select **Macro**. The **Command Type** options specify what action will be associated with the new button.

6. In the **Cmnd** text field, select the file Startup.bas. Use the down arrow to select the Macro directory or use the browse button to navigate to the file.

7. Drag the W icon to the Seehau main window right of the Exit icon. Confirm that if you place your cursor on the W icon, the hint you typed appears.

8. Click the W icon in the Seehau main window and the Startup.bas sequence will execute.

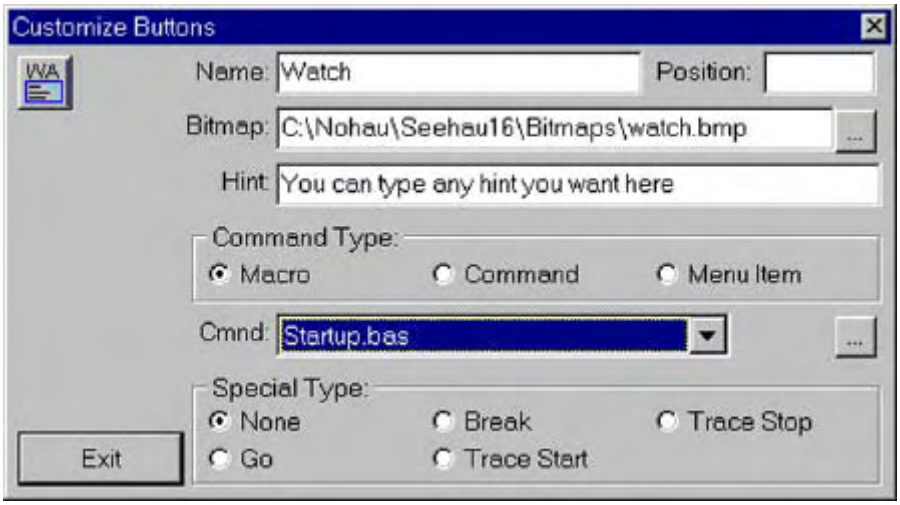


Figure 73. Customize Buttons Dialog Box

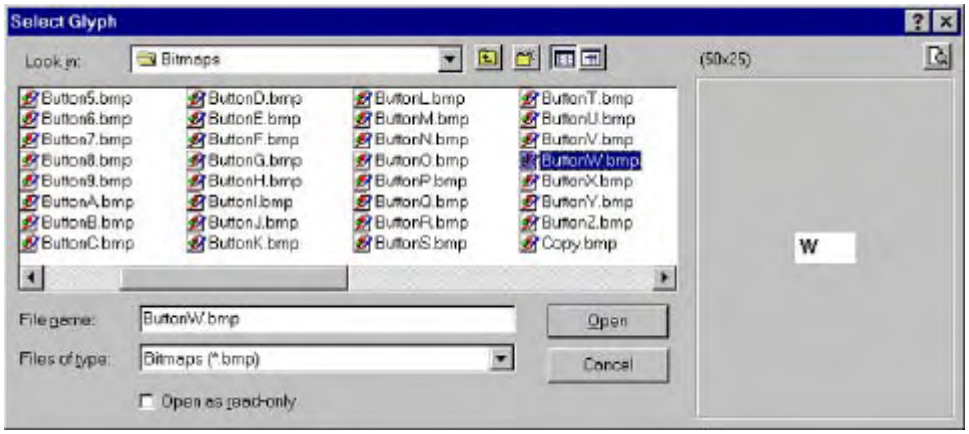


Figure 74. Select Glyph Dialog Box

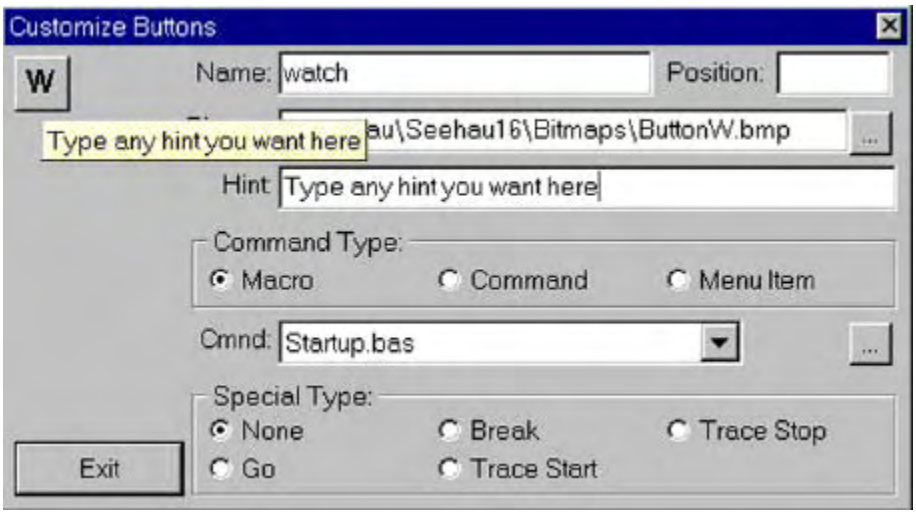


Figure 75. Button Highlight

Note

To remove a button, drag the button back to the Customize Buttons window.

Appendix A. POD–16Y1

The setup to the Nohau POD–16Y1 differs from most Nohau 68HC16 pods in several ways. This section contains information on these differences and a step-by-step procedure for running the timer example.

You should have the Motorola *MC68HC916Y1 Technical Summary* for clarity in case you have problems running these pods. In this guide the "POD–16Y1" section in Chapter 5, "Installing the Pod Boards" applies to POD–16Y1 and POD–(9)16Y1 except as noted.

These pods are identified as POD–16Y1 in black printing on a small white label on the largest connector, and by POD–16Y1 printed in white at the edge of the printed circuit board.

> **Note**
>
> The hardware details of the 68HC16Y1 and the 68HC(9)16Y1 MCUs are not identical! Read the Motorola documentation carefully!

System Clock

A 68HC(9)16Y1 chip with a 4.194-MHz crystal and a reset value for SYNCR (address $YFFA04) of 3000 hex produces an 8.3-MHz system clock frequency. A reset value of 7000 hex produces a 16.7-MHz system clock frequency. The behavior of the W, X, and Y fields in SYNCR may not be reconcilable with some versions of the Motorola *68HC16Y1/916Y1 User's Manual*. The Clock Control Multipliers generally do not agree with the Motorola User's Manual. The manual also indicates that the FASTREF pin must be pulled up (default) to Vcc for a 4.194-MHz crystal and grounded for a 32.768-kHz crystal.

Timer Example

These procedures should work with all the variations of the 16Y family as indicated above.

If you have a full-function emulator at address 200H, you should be able to select that project, load Time.abs, make the patches, select GO and run the timer example.

1. Ensure that there is a jumper inserted connecting BNK0 (pin 2, close to outer edge of the board) to /CS0 (pin one (1), away from the outer edge of the board opposite BNK0). Make sure that the 16BIT jumper 5 position is also inserted. (This step is checking that /CSBOOT will select pod memory.)

2. From the main menu click **Config**, **Emulator**. The **Emulator Configuration** menu should appear with the **Hdw Config** tab already selected (Figure 76).

3. Click the **Misc Setup** tab at the top of the dialog box (Figure 77).

4. Check the **Program Counter** dialog box and enter EC for a value.

5. Check the **Stack Pointer** dialog box and enter 6000 for a value.

6. If the **Reset chip after load file** is not already checked do so at this time.

7. Click the **MCU** tab at the top of the dialog box. The **MCU** dialog box will appear. (Figure 78).

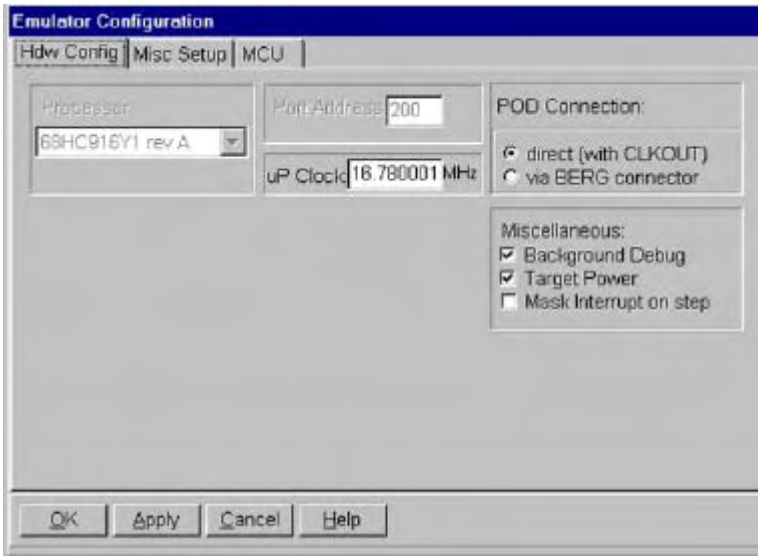


Figure 76. Emulator Hardware Configuration

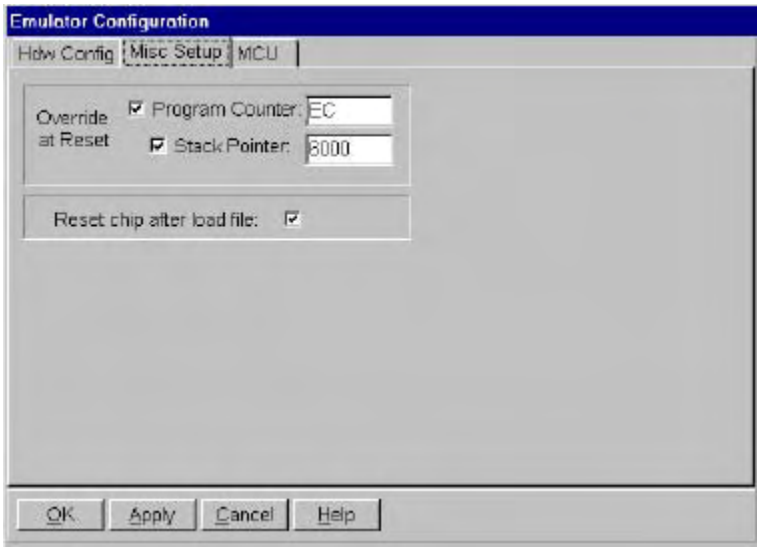


Figure 77. Miscellaneous Emulator Configuration Dialog Box

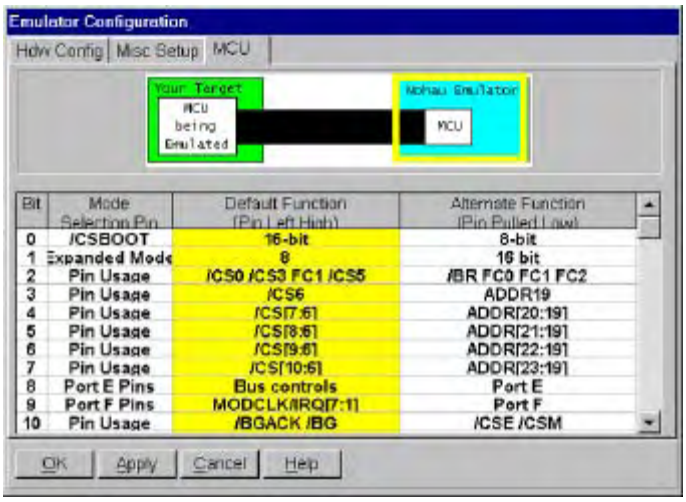


Figure 78. Emulator MCU Configuration

If the flash memories are disabled, they will still set the Stack Pointer and Program Counter on reset if they are set to do so. We need to force the values needed by the timer program. This program puts the values it expects in locations 0 through 7. Other 68HC16 family members would use locations as reset vectors. We should be able to just transcribe the proper values into the **Miscellaneous Setup** dialog box, however, the stack wraps around the bottom of memory, so change the stack to 6000. This is in the middle of emulator memory and unlikely to cause problems. (This is a cheat that you should not plan to use in your final system.) We are using the program memory, which should eventually be ROM of some kind, as writeable data memory.

8. Click the **MCU** tab at the top of the dialog box (Figure 78).

9. The **MCU** dialog box has two distinct areas: The top area has a graphic that has what appear to be two boards connected by a black ribbon. Move your cursor over the left part labeled **Your Target – MCU being Emulated** and select it. A colored box will move to the left and a secondary dialog box appears with four option buttons (Figure 79). This area represents the chip that the emulator is pretending to be.

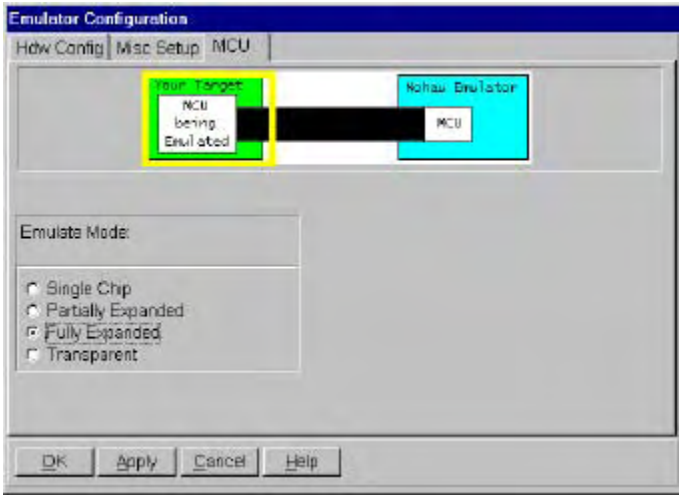


Figure 79. Emulator Target Configuration

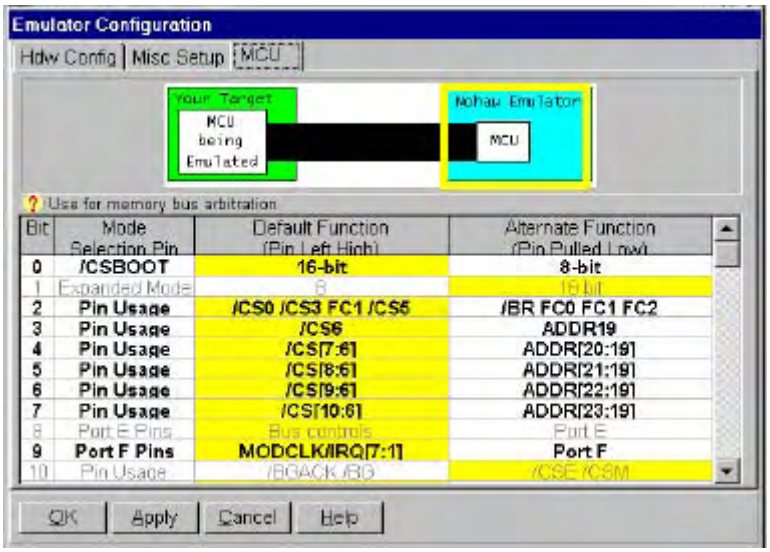


Figure 80. Emulator MCU Values

10. Select **Fully Expanded** to set up the pod configuration.

11. Move your cursor over the graphic at the top labeled **Nohau Emulator – MCU** and select it. The colored box moves back to the right and the initial dialog box reappears (Figure 80 and Figure 81). Some of the text in this box is bold and some text is faded. The faded text is not selectable. The column labeled **Default Function (Pin Left High)** is initially colored green. Move your cursor over the last column **Alternate Function (Pin Pulled Low)** and select the following boxes if not already selected (boxes already selected will be green in color).

Get the data lines set to the proper values on reset. Because the pod has a port replacement unit on it, we want to have the full 16-bit memory bus enabled to use pod memory and allow tracing. The flash memories need to be disabled to prevent them from intruding into the emulator memory address space. There is no guarantee what values they will be programmed with.

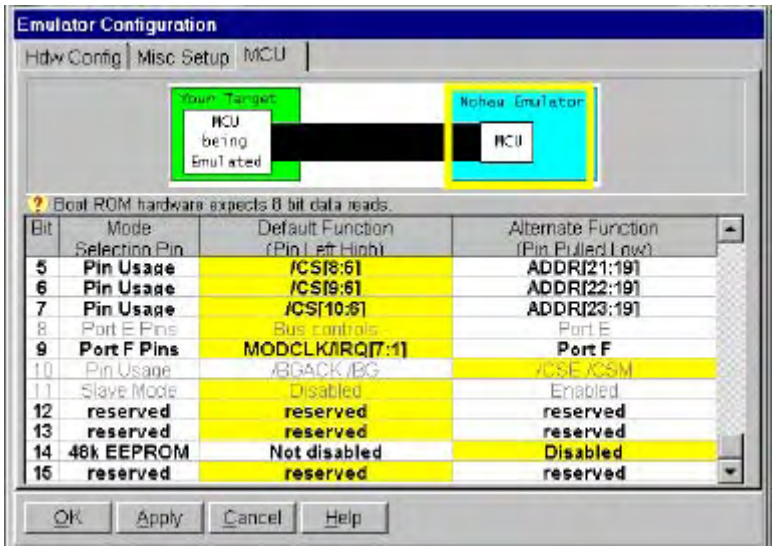


Figure 81. Emulator MCU Values

The following are alternate function settings for the MCU values:

- Bit 1 – Expanded Mode - 16 Bit
- Bit 10 – Pin Usage - /CSE /CSM.
- Bit 14 – ROM – Disabled (For 68HC16Y1 only)
- Bit 14 – 48K EEPROM – Disabled (For 68HC(9)16Y1 only)

12. Move your cursor to the **OK** button and select it. The emulator and trace boards will load.

The next three steps are part of the timer program.

13. Open a Trace window if you have a trace board.

14. Select **File**, **Load Code** after the Seehau software is configured and running.

15. Select the file Time.abs (C://Nohau/Seehau/Examples/Examples/Timer/Time.abs). Either open the file or double-click the file to run it.

Patch the program initialization code to set the clock synthesizer control register (SYNCR) to a value that works for this chip.

16. Open a program window (found at the top of the Source window as a red A (go to address)), set the address to 100. When the line is found:

a. Right-click the Source_1 window.

b. Select **Edit line**.

c. Change **LDD #$7F01** to **LDD #$7001**.

Patch the program initialization code to put the TPU RAM into low power stop so that it does not interfere with the demo.

17. Using the down arrow key move down to address (line) 120 and change **LDD #$0000** to **LDD #$8000**.

Patch the program initialization code to avoid putting the TPU RAM into the middle of program memory space. This is not required under normal circumstances, but if there is any confusion about memory operation, the TPU RAM sitting in the middle of memory confuses the debugging process.

18. Using the down arrow key again move down to address (line) 10A and change **LDX #$FB00** to **JMP 126**.

The rest of the steps are the normal running of the timer demo program.

19. Open a Shadow Memory window (the data entry area at the bottom of the Data window), input **Status** and press ENTER. Right-click the Data window and select **Display as**. Select **ASCII**.

20. Select **GO**.

Appendix B. POD–16Y3

The setup to the Nohau POD–16Y3 differs from most Nohau 68HC16 pods in several ways. This section contains information on these differences and a step-by-step procedure for running the timer example.

You should have the Motorola *MC68HC916Y3 Technical Summary* for clarity in case you have problems running this pod. In this guide the "POD–16Y3" section in Chapter 5, "Installing the Pod Boards" applies to POD–16Y3 and POD–(9)16Y3 except as noted.

> **Note**
>
> The hardware details of the 68HC16Y3 and the 68HC(9)16Y3 **are not identical!** Read the Motorola documentation carefully.

System Clock

A 68HC(9)16Y3 chip with a 4.194-MHz crystal and a reset value for SYNCR (address $YFFA04) of 3000 hex produces an 8.3-MHz system clock frequency. A reset value of 7000 hex produces a 16.7-MHz system clock frequency. The behavior of the W, X, and Y fields in SYNCR may not be reconcilable with some versions of the Motorola User's Manual. The manual also says that the FASTREF pin must be pulled up (default) to Vcc for a 4.194-MHz crystal and grounded for a 32.768-kHz crystal.

Timer Example

If you have a full-function emulator at address 200H, you should be able to select 68HC(9)16Y3/16Y3, load Time.abs, make the patches, click **GO** and run the timer example.

1. Ensure that there is a jumper inserted connecting BNK0 (pin 2, close to outer edge of the board) to /CS0 (pin one (1), away from the outer edge of the board opposite BNK0). Make sure that the 16BIT jumper 5 position is also in. (This step is checking that /CSBOOT will select pod memory.)

2. From the main menu click **Config**, **Emulator**. The **Emulator Configuration** menu should appear with the **Hdw Config** tab already selected (Figure 82).

3. Click the **Misc Setup** tab at the top of the dialog box (Figure 83).

4. Check the **Program Counter** dialog box and enter the value **EC**.

5. Check the **Stack Pointer** dialog box and enter the value **6000**.

6. If the **Reset chip after load file** is not already checked do so at this time.

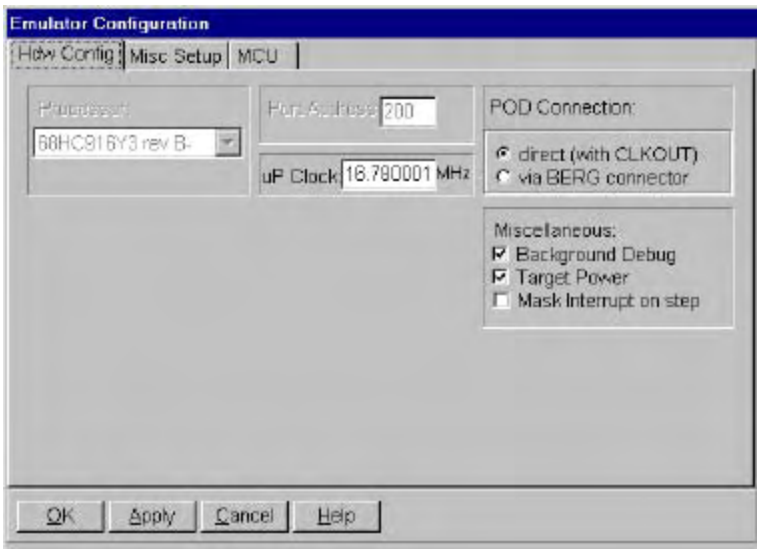


Figure 82. Emulator Hardware Configuration

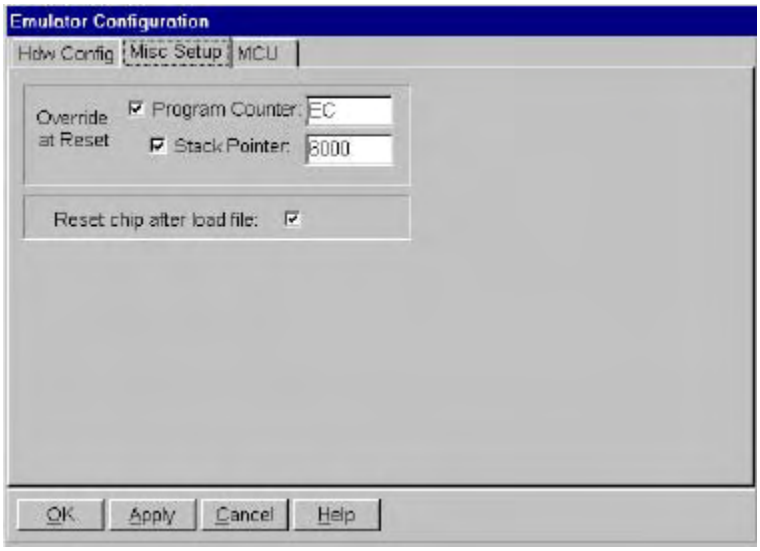


Figure 83. Miscellaneous Emulator Configuration Dialog Box

If the flash memories are disabled, they will still set the Stack Pointer and Program Counter on reset if they are set to do so. We need to force the values needed by the timer program. This program puts the values it expects in locations 0 through 7. Other 68HC16 family members would use locations as reset vectors. We should be able to just transcribe the proper values into the **Miscellaneous Setup** dialog box, however, the stack wraps around the bottom of memory, so change the stack to 6000. This is in the middle of emulator memory and unlikely to cause problems. (This is a cheat that you should not plan to use in your final system.) We are using the program memory, which should eventually be ROM of some kind, as writeable data memory.

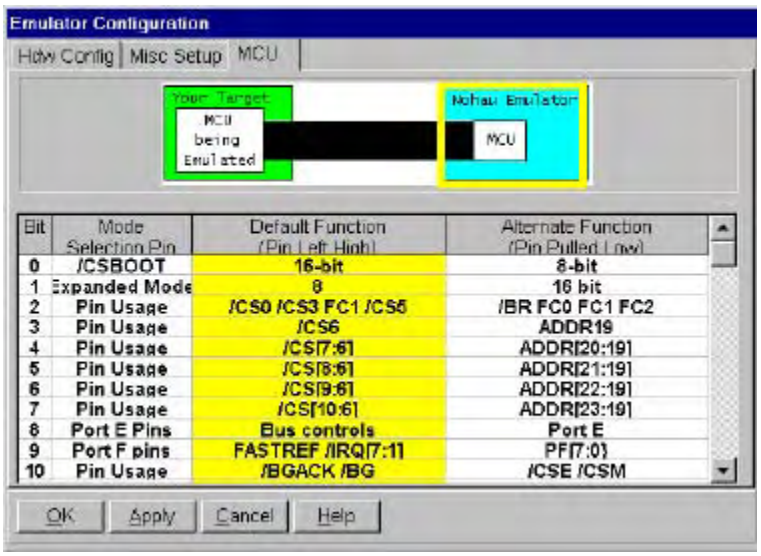


Figure 84. Emulator MCU Configuration

7. Click the **MCU** tab at the top of the dialog box (Figure 84).

8. The **MCU** dialog box has two distinct areas: The top area has a graphic that has what appear to be two boards connected by a black ribbon. Move your cursor over the left part labeled **Your Target – MCU being Emulated** and click it. A colored box will move to the left and a secondary dialog box appears with four option buttons (Figure 85). This area represents the chip that the emulator is pretending to be.

9. Select **Fully Expanded** to set up the pod configuration.

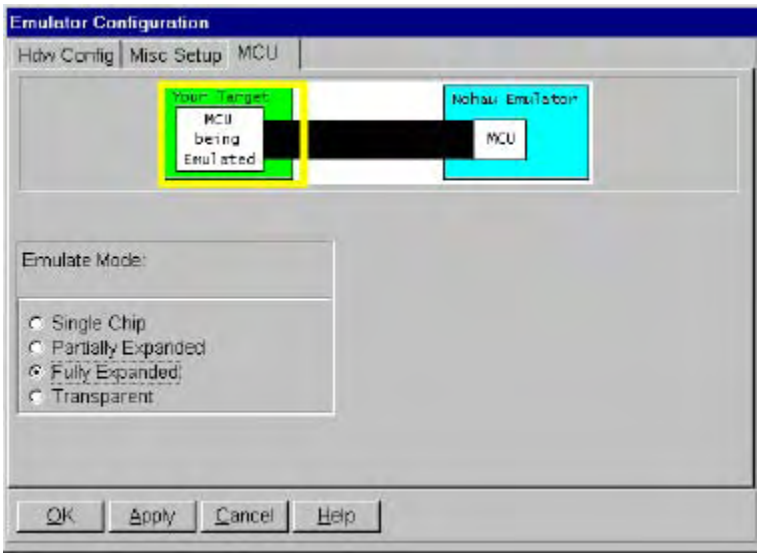


Figure 85. Emulator Target Configuration

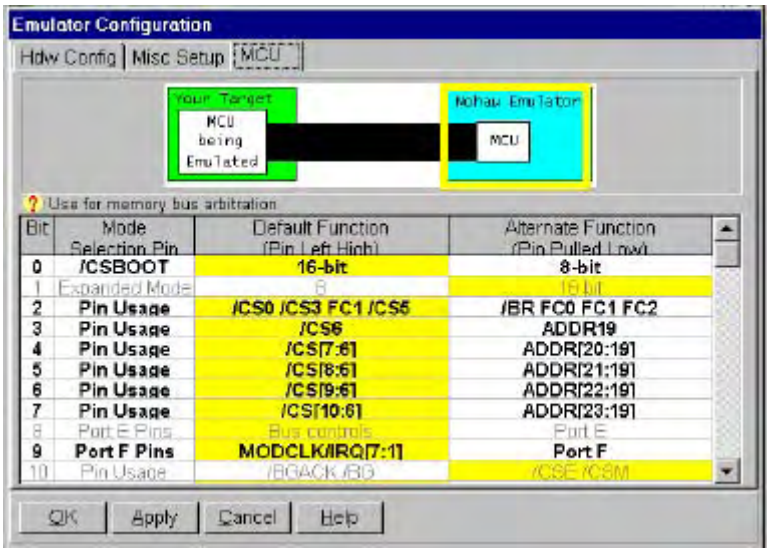


Figure 86. Emulator MCU Values

10. Move your cursor over the graphic at the top labeled **Nohau Emulator – MCU** and click it. The colored box moves back to the right and the initial dialog box reappears (Figure 86 and Figure 87). Some of the text in this box is bold and some faded. The faded text is not selectable. The column labeled **Default Function (Pin Left High)** is initially colored green. Move your cursor over the last column **Alternate Function (Pin Pulled Low)** and select the following boxes if not already selected (boxes already selected will be green in color).

Get the data lines set to the proper values on reset. Because the pod has a port replacement unit on it, we want to have the full 16-bit memory bus enabled to use pod memory and allow tracing. The flash memories need to be disabled to prevent them from intruding into the emulator memory address space. There is no guarantee what values they will be programmed with.

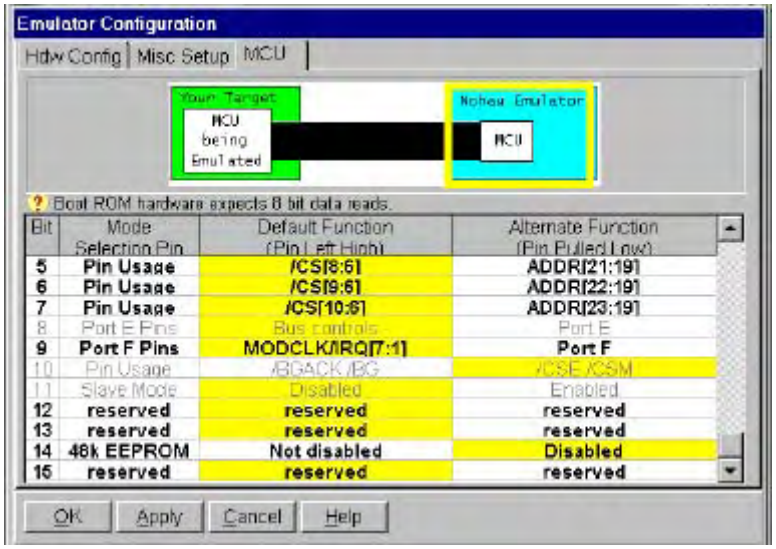


Figure 87. Emulator MCU Values

The following are alternate function settings for the MCU values:

- Bit 1 – Expanded Mode - 16 Bit
- Bit 10 – Pin Usage - /CSE /CSM
- Bit 12 – TPUFLASH – Disabled (68HC(9)16Y3 only)
- Bit 12 – reserved – reserved (68HC16Y3 only)
- Bit 14 – FLASH – Disabled (68HC(9)16Y3 only)
- Bit 14 – ROM – Disabled (68HC16Y3 only)

11. Move your cursor to the **OK** button and click it. The emulator and trace boards will load.

12. Open a Trace window if you have a trace board.

13. Select **File**, **Load Code** after the Seehau software is configured and running.

14. Select the file Time.abs (C://Nohau/Seehau/Examples/Examples/Timer/Time.abs). Either open the file or double-click the file to run it.

Patch the program initialization code to set the clock synthesizer control register (SYNCR) to a value that works for this chip.

15. Open a program window (found at the top of the Source window as a red A (go to address)), set the address to 100. When the line is found:

a. Right-click the Source_1 window.

b. Select **Edit line**.

c. Change **LDD #$7F01** to **LDD #$7001**.

Patch the program initialization code to put the TPU RAM into low power stop so that it does not interfere with the demo.

16. Using the down arrow key move down to address (line) 120 and change **LDD #$0000** to **LDD#$8000** in the same manner as line 100.

The rest of the steps are the normal running of the timer demo program.

17. Open a Shadow Memory window (the data entry area at the bottom of the Data window), input **Status** and press ENTER. Right-click the Data window and select **Display as**. Select **ASCII**.

18. Select **GO**.

Appendix C. POD–16X1

The setup to the Nohau POD–16X1 differs from most Nohau 68HC16 pods in several ways. This section contains information on these differences and a step-by-step procedure for running the timer example.

You should have the Motorola *MC68HC916X1 Technical Summary* for clarity in case you have problems running this pod. In this guide the "POD–16X1" section in Chapter 5, "Installing the Pod Boards" applies to POD–16X1 and POD–(9)16X1 except as noted.

> **Note**
>
> The hardware details of the 68HC16X1 and the 68HC(9)16X1 **are not identical!** Read the Motorola documentation carefully!

Timer Example

If you have a full-function emulator at address 200H, you should be able to select that project, load Time.abs, make the patches, click **GO** and run the timer example.

1. Ensure that there is a jumper inserted connecting BNK0 (pin 2, close to outer edge of the board) to /CS0 (pin one (1), away from the outer edge of the board opposite BNK0). Make sure that the 16BIT jumper 5 position is also inserted.

   /CSBOOT is not present on the 68HC(9)16X1 so we need to use another chip select for the emulator memory. Here we use /CS3, but any other available chip select could be used. The chip selects given in this example are for a 16-bit emulator memory. We also need to disable /CSBOOT and enable /CS3 or an alternative chip select signal, if selected.

2. From the main menu click **Config**, **Emulator**. The **Emulator Configuration** menu should appear with the **Hdw Config** tab already selected (Figure 88).

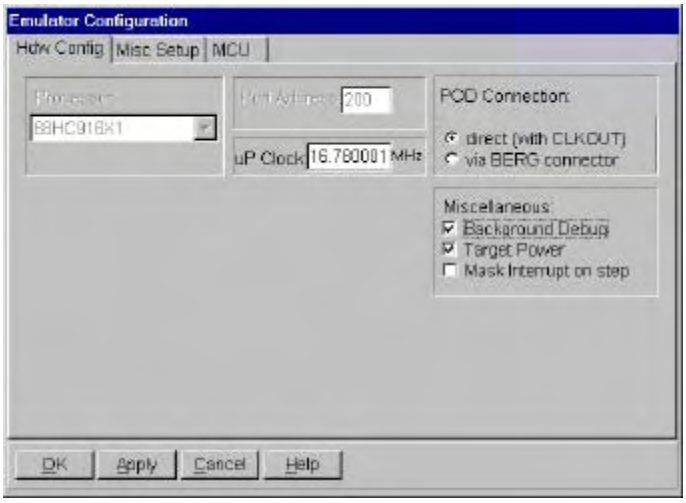


Figure 88. Emulator Hardware Configuration

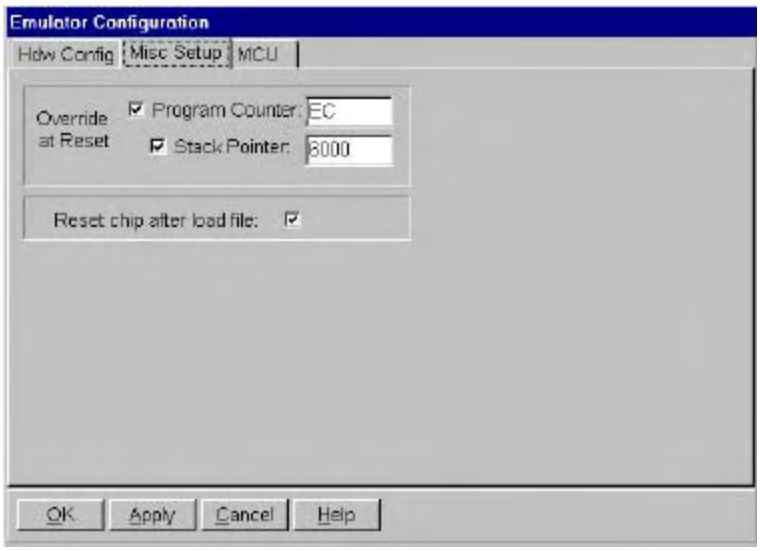


Figure 89. Miscellaneous Emulator Configuration Dialog Box

3. Click the **Misc Setup** tab at the top of the dialog box (Figure 89).

4. Check the **Program Counter** dialog box and enter **EC** for a value.

5. Check the **Stack Pointer** dialog box and enter **6000** for a value.

6. If the **Reset chip after load file** is not already checked do so at this time.

   If the flash memories are disabled, they will still set the Stack Pointer and Program Counter on reset if they are set to do so. We need to force the values needed by the timer program. This program puts the values it expects in locations 0 through 7 where other 68HC16 family members would use them as reset vectors. We should be able to transcribe the proper values into the **Miscellaneous Setup** dialog box, however, the stack wraps around the bottom of memory, so change the stack to 6000 which is in the middle of emulator memory and unlikely to cause problems. (This is a cheat that you should not plan to use in your final system.) We are using the program memory, which should eventually be ROM of some kind, as writeable data memory.

7. Click the **MCU** tab at the top of the dialog box (Figure 90).

8. The **MCU** dialog box has two distinct areas: The top area has a graphic that has what appear to be two boards connected by a black ribbon. Move your cursor over the left part labeled Your **Target – MCU being Emulated** and click it. A colored box will move to the left and a secondary dialog box appears with four option buttons (Figure 91). This area represents the chip that the emulator is mimicking.

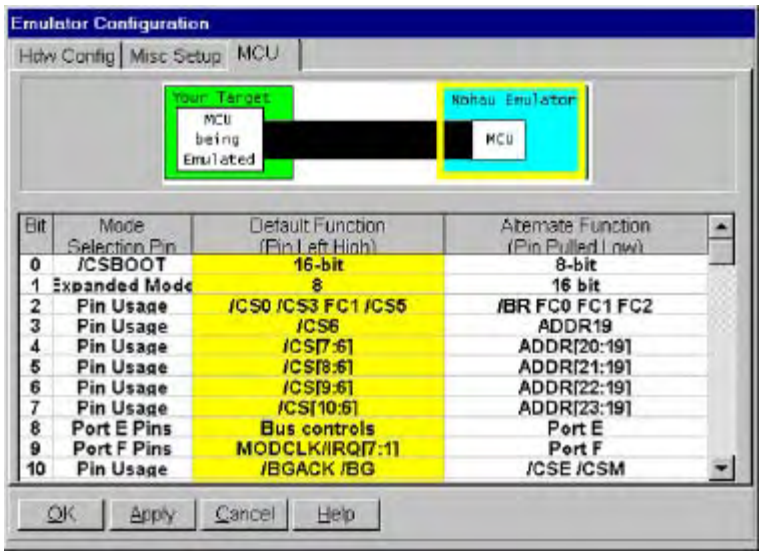


Figure 90. Emulator MCU Configuration

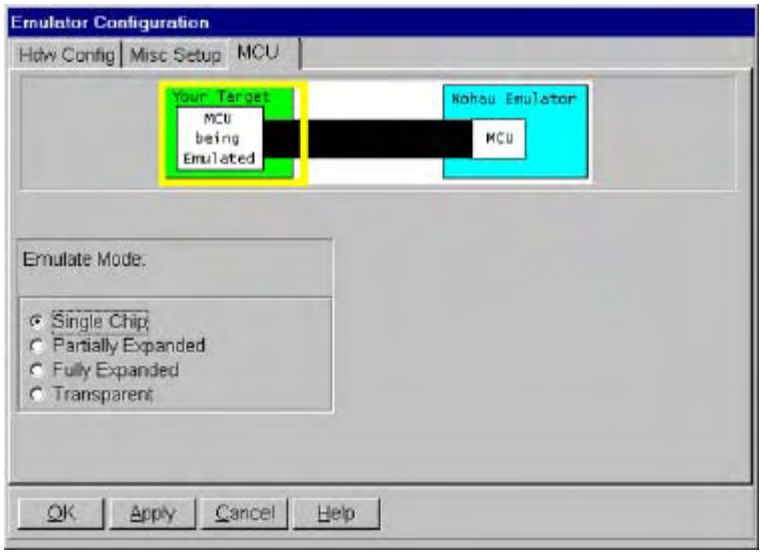


Figure 91. Emulator Target Configuration

9. Select **Single Chip** to set up pod configuration.

10. Move your cursor over the graphic at the top labeled **Nohau Emulator – MCU** and click it. The colored box moves back to the right and the initial dialog box reappears (Figure 90 and Figure 91). Some of the text in this box is bold and some faded. The faded text is not selectable. The column labeled **Default Function (Pin Left High)** is initially colored green. Move your cursor over the last column **Alternate Function (Pin Pulled Low)** and select the following boxes if not already selected (boxes already selected will be green in color):

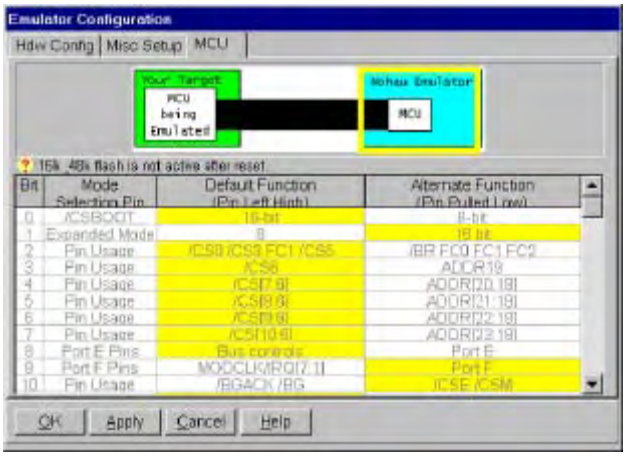


Figure 92. Emulator MCU Values

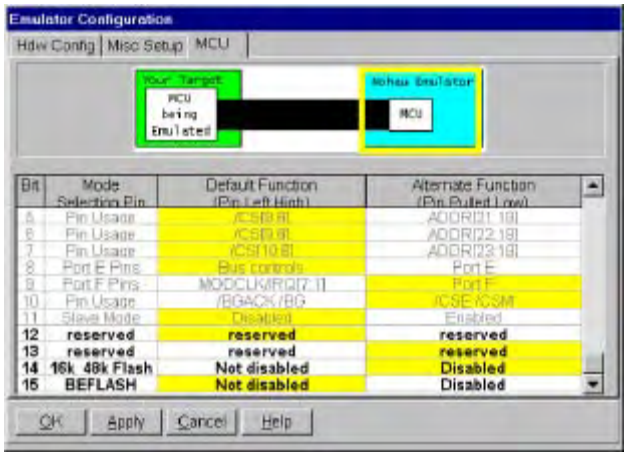


Figure 93. Emulator MCU Values

The following are alternate function settings for the MCU values:

- Bit 1 – Expanded Mode - 16 Bit
- Bit 8 – Port E Pins - Port E
- Bit 10 – /BGACK /BG - Enabled
- Bit 13 – /CSM – Enabled (68HC16X1 only)
- Bit 13 – Reserved – Reserved (68HC(9)16X1 only)
- Bit 14 – ROM – Disabled (68HC16X1 only)
- Bit 14 – 16k 48k Flash – Disabled (68HC(9)16X1 only)

11. Move your cursor to the **OK** button and click it. The emulator and trace boards will load.

12. Open a Trace window if you have a trace board.

The rest of the steps are the normal running of the timer demo program.

13. Select **File**, **Load Code** after the Seehau software is configured and running.

14. Select the file Time.abs (C://Nohau/Seehau/Examples/Examples/Timer/Time.abs). Either open the file or double-click the file to run it.

15. Open a Shadow Memory window (the data entry area at the bottom of the Data window), input **Status** and press ENTER. Right-click the Data window and select **Display as**. Select **ASCII**.

16. Select **GO**.

Appendix D. Troubleshooting Tips

If you encounter a problem when starting or running the emulator and/or Seehau, try the following troubleshooting tips. For detailed troubleshooting instructions, refer to Appendix A, "Troubleshooting" in the *EMUL16/300–PC Windows User Guide*, or contact Nohau Technical Support at support@icetech.com

WARNING

Always turn the power off before you plug in or unplug boards, ribbon cables, or the pod board!

- Verify the proper pod type is selected, and jumper configurations match the default configuration. Refer to Chapter 4, "Pod Boards" in the EMUL16/300 User Guide for your specific pod type.

- Determine if the emulator and pod operate together when not connected to the target system. Remove the pod from the target and attempt to start the system in stand-alone mode. The emulator does not require a target. To troubleshoot in stand-alone mode, make sure the power jumper is selected for internal power and the crystal (clock) jumpers are in the pod position (Refer to Chapter 4, "Pod Boards" of the EMUL16/300 User Guide).

- Try another PC.

- Reload Seehau. To reload, use the Windows Add/Remove Programs option. This ensures all files and registry entries are properly deleted.

- To access the **Add/Remove Programs Properties** dialog box, go to the **Start** menu and point to **Settings**. Click **Control Panel**. Double-click **Add/Remove Programs**.

- Verify there is no address conflict with the PC. If you are using the default emulator board address (200H), make sure there is no other device using 200H in your PC. If there is a conflict, refer to Chapter 3, "Installing and Configuring the Emulator Board," and Chapter 4, "Installing and Configuring the Trace Board" in this guide.

- If you have a trace board and you are using the default trace board address (208H), make sure that there is no other device using 208H. If there is a conflict, refer to Chapter 4, "Installing and Configuring the Trace Board" in this guide.

- If you have trouble printing while the printer is connected to the HSP, make sure that the HSP is powered on for the printer to receive printer port signals.

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.

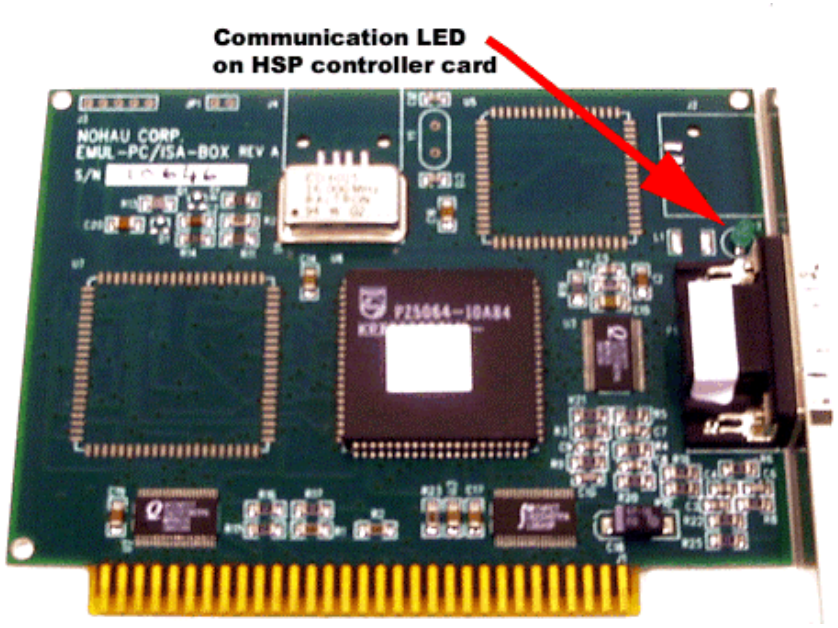


Figure 94. HSP Card LED

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 **SeehauHC16**, 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 **SeehauHC16** 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 5, “Installing the Pod Boards.”

- **No**. Refer to Chapter 2, “Installing and Configuring the Seehau Software.” 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 Nohau16 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 95).

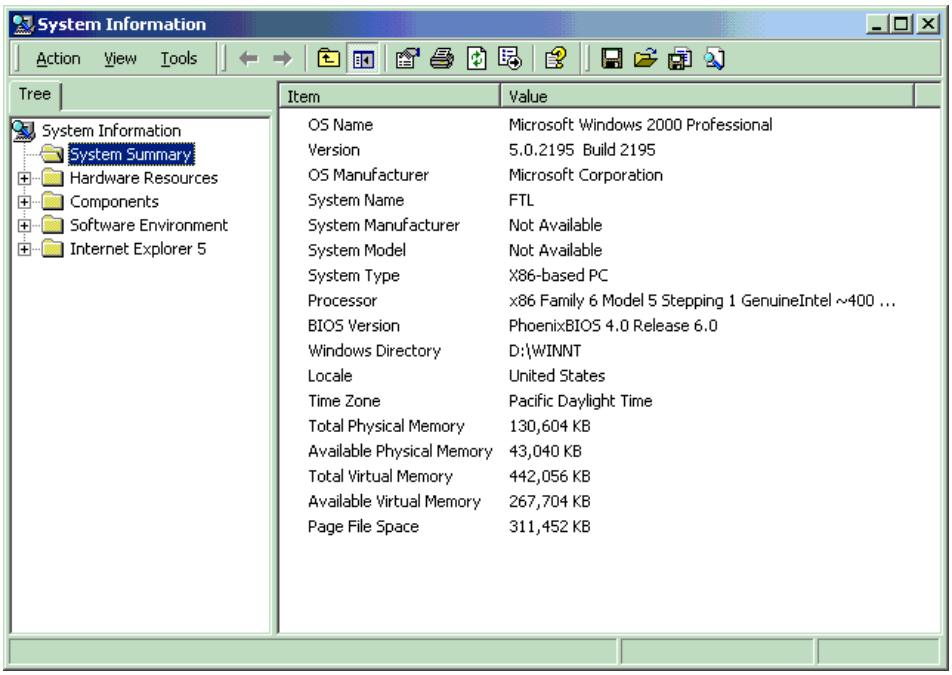


Figure 95. System Information Window

3. Click **Software Environment**.

4. Click **Drivers** to display a list of active drivers. Refer to the **Name** column and scroll down to Nohau16 (Figure 96).

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

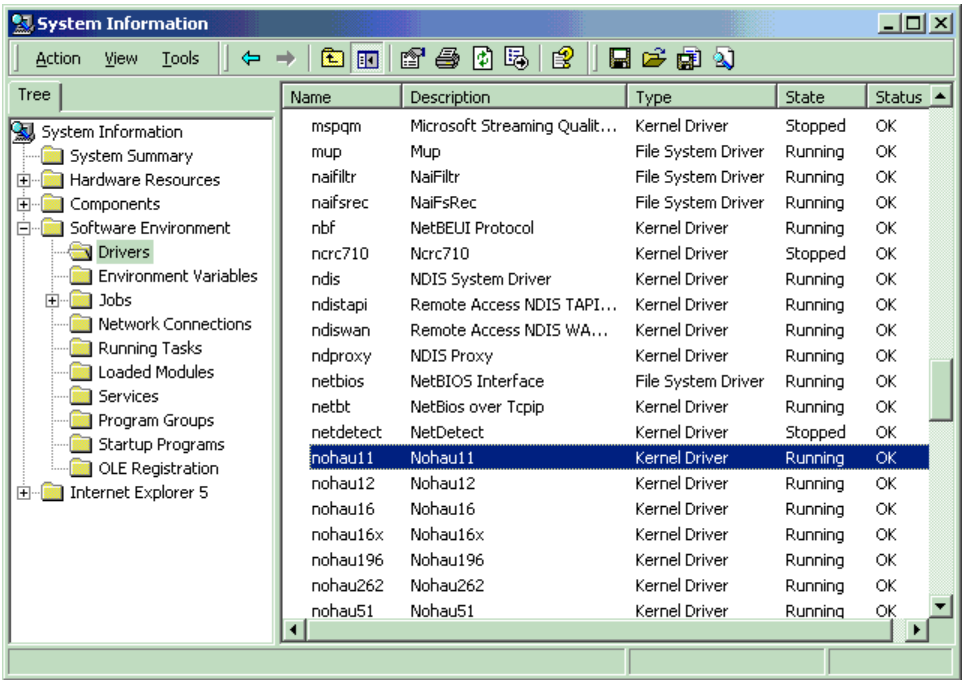


Figure 96. List of Active Drivers

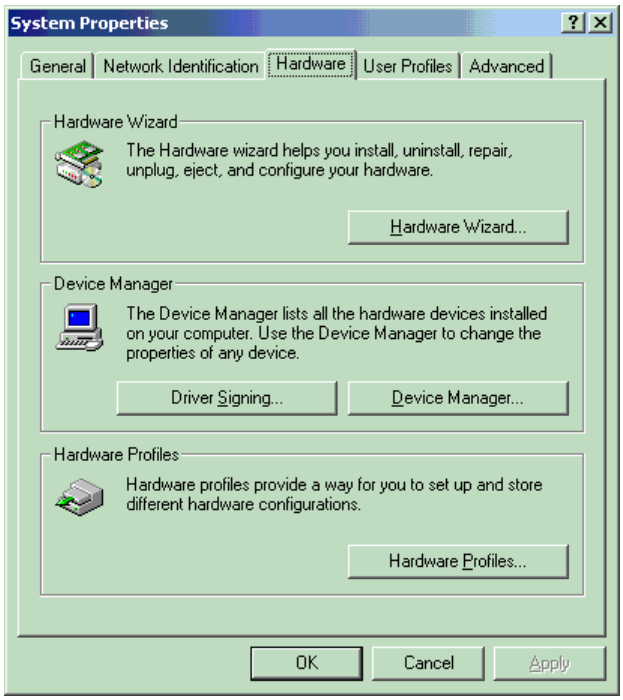


Figure 97. System Properties Window

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 97).

2. Click the **Hardware** tab. Then click **Device Manager**. The Device Manager window opens (Figure 98).

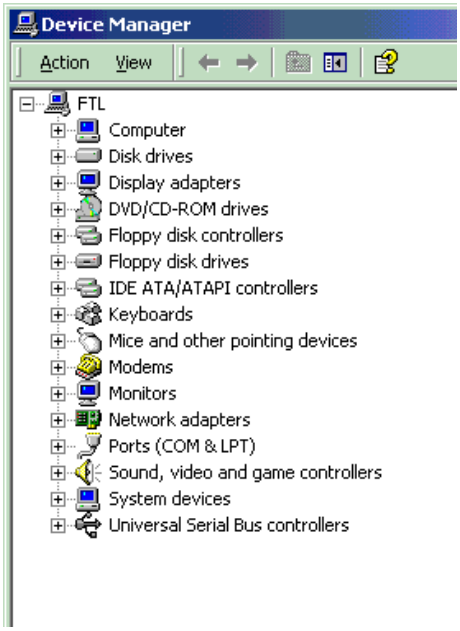


Figure 98. Device Manager Window

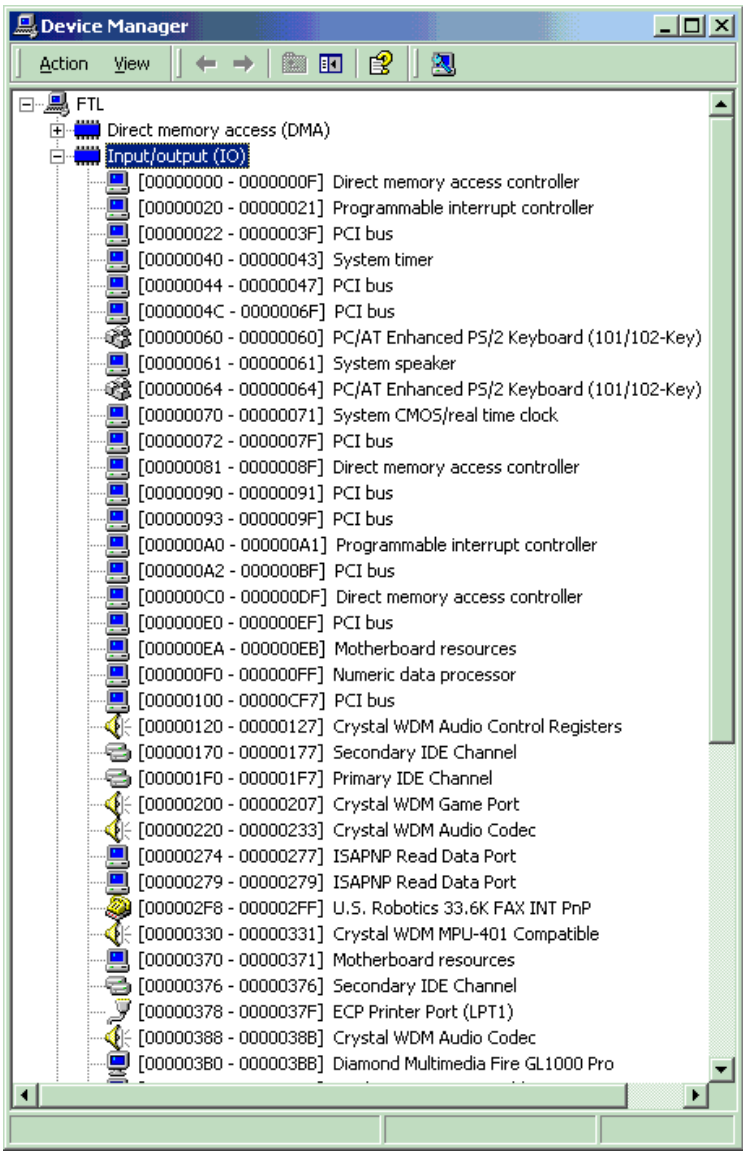


Figure 99. Device Manager Window Displaying the System Resources

3. In the Device Manager window, select the **View** menu. Then click **Resources by Type**. A window appears that shows system resources (Figure 99).

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, verify that 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 **SeehauHC16**, 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 **SeehauHC16** 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 5, "Installing the Pod Boards."

- **No**. Refer to Chapter 2, “Installing and Configuring the Seehau Software.” 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

We are 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 Technical 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.
- 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.

- 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

- Check the target for any device that is enabled by address qualification only and does not use 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 2, "Installing and Configuring the Seehau Software" or the “Starting Seehau” section in Chapter 6, “Starting the Emulator and Seehau Software.” • Check R/W signals on the target. • Make sure you have an expanded mode pod.

Appendix E. PAL Equations for RAM

```
                    •
RWN         =       •   /RW
WRITE       =       •   FRZ * /CSB0 * /RW ; ROM emulation
            +       •   /CSB1 * /RW
            +       •   /CSB2 * /RW
            +       •   /CSB3 * /RW
A17A0       =       •   A0*   /16BIT * /1MB; A0 in 8-bit mode
            +       •   A17*  /16BIT * /1MB; A17 in 16-bit mode
            +       •   A17*  1MB           ; A17 with 512K chip
```

Appendix F. ISO–160

ISO-160 is a set of four parts that, when used together, may be useful with targets that have an external watchdog timer, or other externally generated signals that interfere with emulation. The ISO-160 can be used with any pod and with any kind of adapter. Inserting four of these isolators between the pod and the adapter (between the controller and the target) inserts 160 DIP switches, each dedicated to one signal, so any single signal or any combination of signals from the target board can be interrupted before they can reach the controller.

Each isolator is designed with enough switches to support as many as 40 signals. A set of four can disable any combination of 160 signals. This means that there are more pins and sockets on the isolator than on some pod boards and adapters. In this case, install the isolator so that the excess pins all extend to the right of the header and pins on the adapter and the pod. In other words, the pins near the ISO-160 label should be the unused pins.

This adapter is good for isolating chip select lines needed for emulation RAM from the target board. Flip the switch for the offending chip select signal, clip or solder a pull up resistor to the target side of the switch, and the target device will be isolated from the controller with no trace cutting or pad lifting or target hardware modifications.

Note

By interrupting an input signal to the controller, an open switch can create a floating input signal to the controller. If no pull-up or pull down resistor is used to give the input a definite state, the controller may behave unexpectedly.

Appendix G. Compilers

Intermetrics/Whitesmiths

Check your distribution disk labels to make sure that you have version 3.31 Mod 9 or later. If not, contact Intermetrics to obtain an update.

Assembler Notes

The assembler does not produce the symbols for assembler source lines. This means that assembler source files will not be shown in the Source window. Global and local variables will be added to the object file if the –n switch is used.

Compiler Notes

EMUL16/300/BDM does not support symbolic debugging for programs compiled in the COMPACT memory model with the Mod 9 compiler. Such programs can still be converted to S records, loaded into the emulator, and then debugged without symbols. In later Mods of the compiler, the COMPACT memory model may be supported. Call Nohau Technical Support for the latest information.

The Whitesmiths/Cosmic C compiler is a DOS standard mode application. It does not use extended or expanded memory. You will have no difficulty running it in a DOS window under Windows.

Compiler Switches

The Whitesmiths/Cosmic C compiler comes with a powerful compiler driver, C.EXE. It reads a prototype file, usually named C.PRO, which contains an elaborate script, and processes your command line by the rules in the prototype file. Most of the intelligence of the driver lies in this prototype file. Once the prototype file is set up with the desired switches, using the Whitesmiths/Cosmic C Compiler is quite easy, but getting it set up can be difficult.

Of the switches shown only –dxdebug is required to support source level debugging. All others are optional. The other switches cause the tool chain to produce other, helpful information, such as link map files.

-dxdebug	Generate the symbol and line number information
-v	Display on the screen the command used to invoke each phase of the compiler
-dmod_	You must specify any memory model other than Compact
-dsavlnk	Save the linker command file created
-dmap	Generate a link map file
-dxref	Generate a cross-reference as a part of the listing file
-dlistcs	Generate a listing file

INTERMET/C682X (for 6833x or 68340)

Assembler Notes

The –d switch will put both symbols and line number information into the object file. Modules assembled with the –d switch will appear in the Source window, just the C source files.

No other switches affect the emulator, but the following switches can provide information helpful when debugging:

-l	Generate a listing file
-x	Generate cross-reference listing
-b	Generate symbol table listing
-v	Verbose mode: report the status of assembly

Compiler Switches

At one time, the Itools C682X compiler was sold with a memory manager that conflicted with the one used by Windows. It could not be run in a DOS window under Windows. If they are still shipping that version, a telephone call to Intermetrics will get you an update that will run in a DOS window.

The –d is the only switch that controls the debugging information in the object file. If this switch is included in the compiler command line, the object file produced will contain the information necessary for source level and high level debugging.

Under most circumstances, the –d switch will be the only switch necessary. However, circumstances can arise where actions of the optimizer can cause problems for the debugger. If the debugger displays blatantly incorrect information about variable locations and source code locations, you may wish to add the –do switch to the compiler command line. This switch will turn off the kinds of optimizations likely to cause problems for the debugger.

Other helpful switches are:

-l	Generate a listing file
-l	Generate interleaved source and assembly listing
-x	Generate cross-reference listing
-v	Report the status of the compile

The output from the linker is in a text based format that is unsupported by the EMUL16/300/BDM debugger. To generate a file that the emulator can load, use the formatting utility (called form.exe) with the –f c switch. This will generate a Common Object File Format (COFF) file, which is a standard format supported by EMUL16/300/BDM.

Introl

INTROL/C16–MSDOS–5HD-1 (for 68HC16)

Assembler Notes

The –l (lower case L) switch will create debugging symbols for all assembler symbols and line numbers. This may not be in the Introl documentation. Use this switch to see assembler source in the Source window.

Startup Code Notes

The file cpu16.s contains macros that, when evaluated, create name macros for most of the 68HC16 configuration registers. The code in start.s sets the switch to just create the name macros that accept 32-bit arguments and load the K register along with the X register, for example. It has not been changed.

The file CONFIG.LIB contains a switch use_nohau that supports EMUL16/300/BDM (stand-alone), which seems to work fine.

Compiler Notes

One compiler switch –gg is required for source level debugging. All other switches are optional. The compiler will leave a trail or history of what it has been doing if you use the other switches shown in the following table to produce listing and other files.

-gg	Turns on debug symbol generation. Required for source debugging
-k	Displays tool name and version number during each compile
-l	Generate compiler listing file

INTROL/C62–MSDOS–5HD–1 (for 6833x or 68340)

Assembler Notes

The –l (lower case L) switch will create debugging symbols for all assembler symbols and line numbers. This may not be in the documentation. Use this switch to see assembler source in the Source window.

Compiler Notes

One compiler switch –gg is required for source level debugging. All other switches are optional. The compiler will leave a trail of what it has been doing if you use the other switches shown in the following table to product listing and other files.

-gg	Turns on debug symbol generation. Required for source debugging
-k	Displays tool name and version number during each compile
-l	Generate compiler listing file

Microtec Research MCC68K (for 6833x or 68340)

Assembler Notes

The Microtec Research assembler does not support source level debugging. There are no switches that, when used, will tell the assembler to include the line number information necessary to synchronize the assembler source file with the loaded object file. The Program window will display the disassembled instructions, just as it does with any kind of object file. If you include the –f d switch, the assembler will include assembler symbols (variable and function names) in the object file for use by the emulator software.

Compiler Notes

The file MCC68K.EXE is a driver that looks at the command line and very intelligently figures out what to invoke based upon the switches and file name suffixes. All of the switches described here should be passed to MCC68K.EXE, not any of the individual compiler passes, if invoked separately.

The compiler driver and all compiler passes in our tests have run successfully in a DOS window under Windows. Other, more complex programs may not compile without the –B switch, which uses extended memory, which may conflict with your installation of Windows. If you have trouble running these tools under Windows, contact Microtec Research for assistance.

The MRI compiler driver has possibly the longest list of compiler switches of any compiler running on any host. There are 90 described in the summary section of the manual.

Of those 90 switches, the –g switch turns on debugging information, and the –Kf switch forces the compiler to create a stack frame for every function. These two switches are both necessary for either the Source level or the High level debugger. In addition to the –g and –Kf switches, you may also use –Gf which will add full paths to the source file names, but the emulator will strip off the full path names before it uses the file names. Other switches that turn on debugging information, -Gl, -Gm, and –Gr, all modify or restrict the kind of debugging information produced, and should not be used with EMUL16/300/BDM.

Other helpful switches that are not strictly required by EMUL16/300/BDM include:

-l	Produces a listing file
-pcpu32	Produces instructions and alignments needed by all CPU32 controllers

It is possible for the optimizer to hoist code and reassign variables to registers in a way that can confuse the emulator. If, while debugging optimized code, you see values that are especially confusing or just plain impossible, try turning off optimizations (use the –nO switch or remove the –O switch) and see if the display then makes better sense.

Sierra Systems C Compiler (for 6833x or 68340)

Assembler Notes

By default, the assembler includes global symbols in the object file. No switch is necessary to get variables and code labels in the object file. (The switch –Sx will exclude these symbols.) Unfortunately, there is no switch to tell the assembler to include line number information in the object file. Assembly source files cannot be displayed in the Source window without this information. The object code generated from assembly source will be disassembled in the Program window as is normally done.

Compiler Notes

The Sierra C compiler comes with several compiler drivers which intelligently examine the command line, sort the input files by suffix, and then call the various tools (compiler, assembler, linker, etc.) as needed to create a linked executable. When developing 68332 applications, the driver to use is called c332. The compiler driver accepts certain switches and translates them into comparable switches needed for each tool.

Of the many switches available, only the switches –g –Of1 are required for Source and High level debugging. Using the –g switch will tell the compiler (and assembler) to include the symbol information and line numbers that EMUL16/300/BDM needs to display source files and inspect variables. The –Of1 switch forces the compiler to produce a stack frame for every function, even those simple enough to not need one.

There are three other switches that turn on debugging, ‘g1, -q, and –q1. Of these switches, -g1 will produce correctly formatted debugging information, but it allows multiple variables to share a single register, which may confuse the emulator. The other two switches produce debugging information formatted for a different debugger.

As with most sophisticated compilers, certain optimizations are very difficult to display correctly. In general, using some or all of these switches: -Oc0 –Ox0 –Op0 –O10 will simplify the object file and allow the emulator to display the code and local variables as correctly as possible.

At this time, the compiler produces constructs that are very difficult or impossible for the EMUL16/300/BDM to correctly display no matter what switches are used. These include unprototyped stack frames, 64-bit floating point numbers, 32-bit fast Motorola floating point numbers, and constants embedded within the code. Call Nohau Technical Support for the latest information about support for any of these features.

Examples

MAKEFILE (make utility)

```
CFLAGS=-V 32 –f –J –M disk=yes
demo.map: demo.obj makefile
        sym –P "%n-30s %s-8s %v4X %r-15s" –o demo.map demo.obj

demo.obj: funcs.o demo.o link.spc d:\sds\lib68332\start.o makefile
        linker –f link.spc demo.o funcs.o d:\sds\lib68332\start.o –o demo.obj

func.o: funcs.c funcs.h makefile
      d:\sds\cmd\cc68000 $(CFLAGS) funcs.c

demo.o: demo.c funcs.h makefile
        d:\sds\cms\cc68000 $(CFLAGS) demo.c
```

MAKER.BAT (example using spec file and make utility)

```
d:\sds\cmd\cc68000 -V 32 -f -J -M disk=yes funcs.c -og
d:\sds\cmd\cc68000 -V 32 –f -J -M disk=yes demo.c
d:\sds\cmd\linker -f link.spc demo.o funcs.o d:\sds\lib68332\start.o -o demo.obj
sym -P "%n-30s %s-8s %v4X %r-15s" -o demo.map demo.obj
```

LINK.SPC (specification file)

Example specification file for the 68000 family. This example assumes ROM is at address 0x0000 and RAM is at address 0xC000. Region "data" (which contains initialized RAM variables) must be linked into RAM to give the variables their correct addresses, but will be downloaded into ROM at location DATA using downloader option "-m data, DATA". Startup code will copy it from location DATA back into RAM.

```
partition { overlay {
    region {} reset [addr=0];          /* reset vector */
    region {} vects [addr=8];          /* other vectors */
    region {} code;                    /* executable code */
    region {} const;                   /* constant strings */
    region {} string;                  /* where to download region data
    DATA = $;                          */
} o1; } ROM;
Partition { overlay {
     region {} data[roundsize=4];      /* RAM to be initialized on reset */
     region {} ram[roundsize=4];       /* RAM to be zeroed on reset */
     region {} mal-                    /* RAM available to malloc() */
loc[size=0x1000];                      /* stack of at least 0x1000 bytes */
     region {}                         /* stack pointer reset value */
stack[size=0x1000];
     STKTOP = $;
} o2; } RAM[addr=0x4000;
```

Appendix H. Target Boards

TRG–16Y1

This target board includes a 68HC16Y1, a 32K by 8-RAM chip, a 32-kHz oscillator, a BERG connector, and two diodes.

The two diodes configure the controller during the /RESET cycle. Diode D3 connects the /RESET line and DB0 so that the controller will come out of /RESET expecting an 8-bit memory device. If you connect another RAM chip in the wire wrap area to /DB0 through /DB7, be sure to remove the diode shown in D3 below. The other diode connects the /RESET line and DV14 to disable the internal ROM on the 68HC16Y1. This second diode (labeled D2 on the board) runs through header JP2 so that when the jumper is in place, on-chip ROM is disabled. If you wish to use the ROM on the controller, remove the jumper from header JP2.

Although the target uses 8-bit memory, selecting the 8-bit emulation memory option in the **Hardware Configuration** dialog box will not correctly configure the chip select registers at this time. Using the **User defined** option, the CSBARBT value should be set to either 0002 or 0003. A value of 0002 supports a 16384-byte device at address 0, which will make 16384 bytes of RAM unusable. A value of 0003 will make the entire chip usable, but because it is active for all addresses below 65535, address wrapping could occur between 32767 and 65535. It is up to you to choose either 2 or 3 and to put that value in the **CSBART** field, or to put some other appropriate value in that field (and possibly other fields) if you have changed RAM chips.

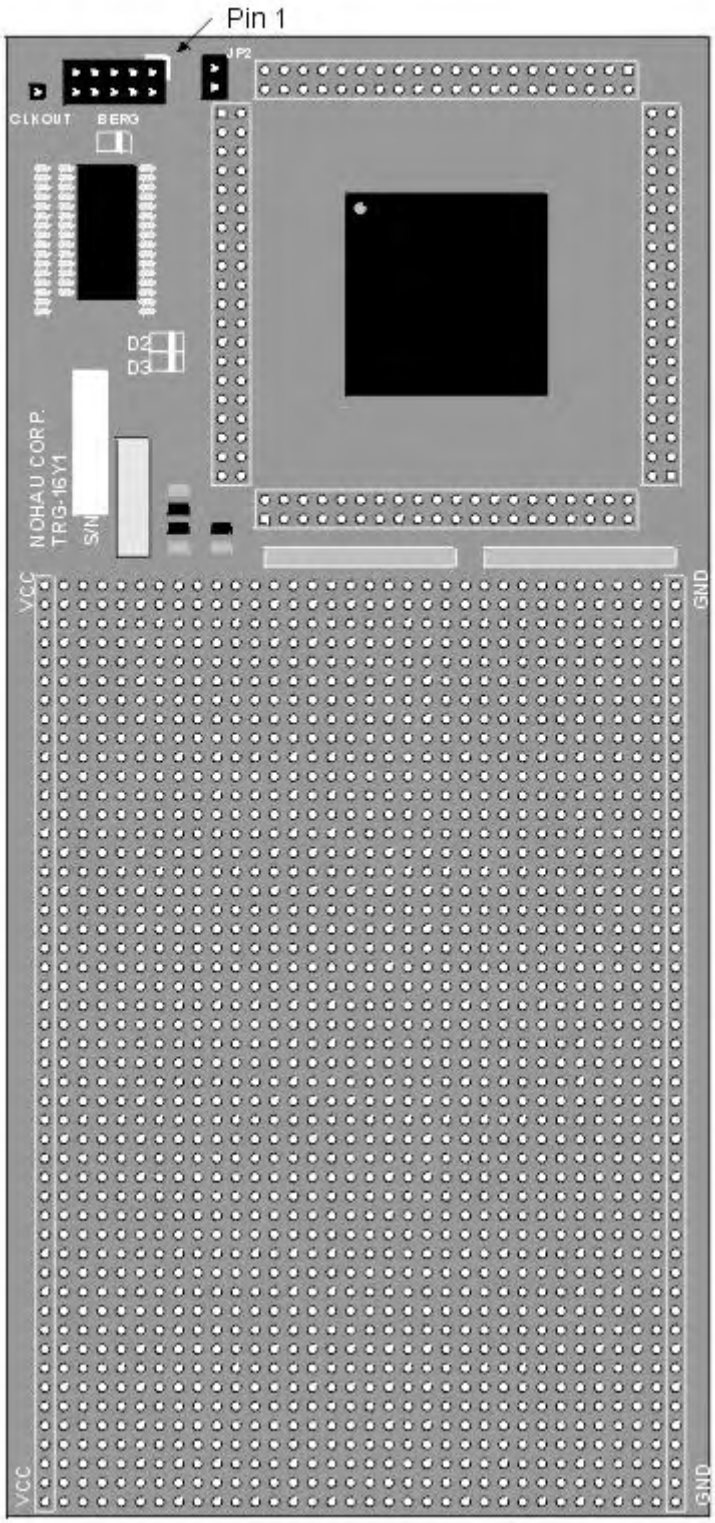


Figure 100. Target Board TRG–16Y1

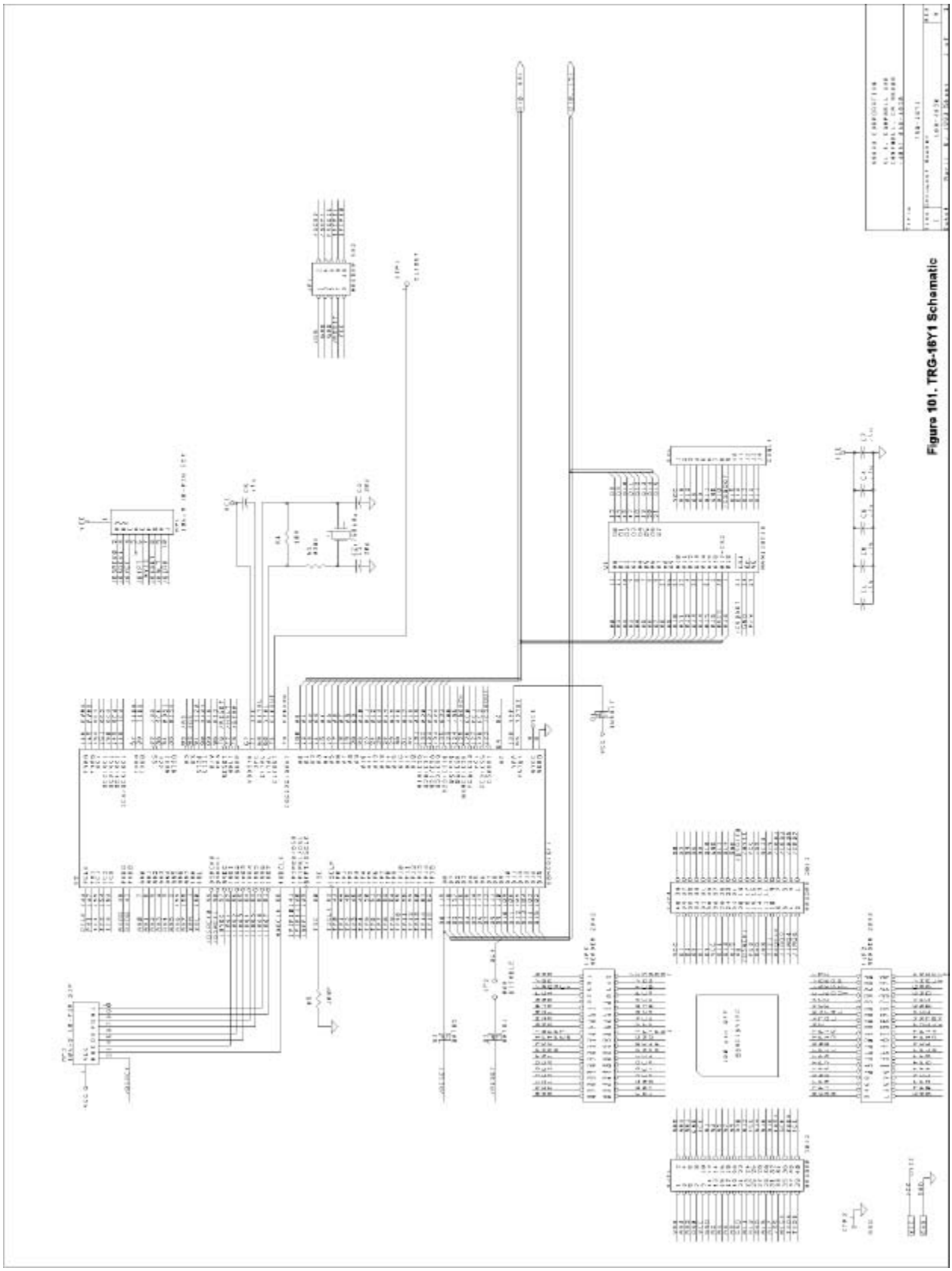


Figure 101. TRG–16Y1 Schematic

TRG–16Z1

This target board includes a 68HC16Z1, a 32K x 8-RAM chip, a 32-kHz oscillator, and a BERG connector. In addition, there is a diode between the /RESET line and /DB0 so that the controller will come out of /RESET expecting an 8-bit device.

Even though the target uses 8-bit memory, selecting the 8-bit emulation memory option in the **Hardware Configuration** dialog box will not correctly configure the chip select registers at this time. Using the **User defined** option, the CSBARBT value should be set to either 0002 or 0003. A value of 0002 supports a 16384-byte device at address 0, which will make 16384 bytes of RAM unusable. A value of 0003 will make the entire chip usable, but because it is active for all addresses below 65535, address wrapping could occur between 32767 and 65535. It is up to you to choose either 2 or 3 and to put that value in the **CSBART** field, or to put some other appropriate value in that field (and possibly other fields) if you have changed RAM chips.

If you connect another RAM chip in the wire wrap area to /DB0 through /DB7, be sure to remove the diode shown in D1 below. This diode is used to pull /DB0 low during a /RESET, configuring the controller for 8-bit memory. Removing this diode will allow /DB0 to float, configuring the controller for 16-bit memory. Removing this diode will ensure that the reset vectors are read correctly, and will help ensure that the target will operate correctly when emulating through a /RESET.

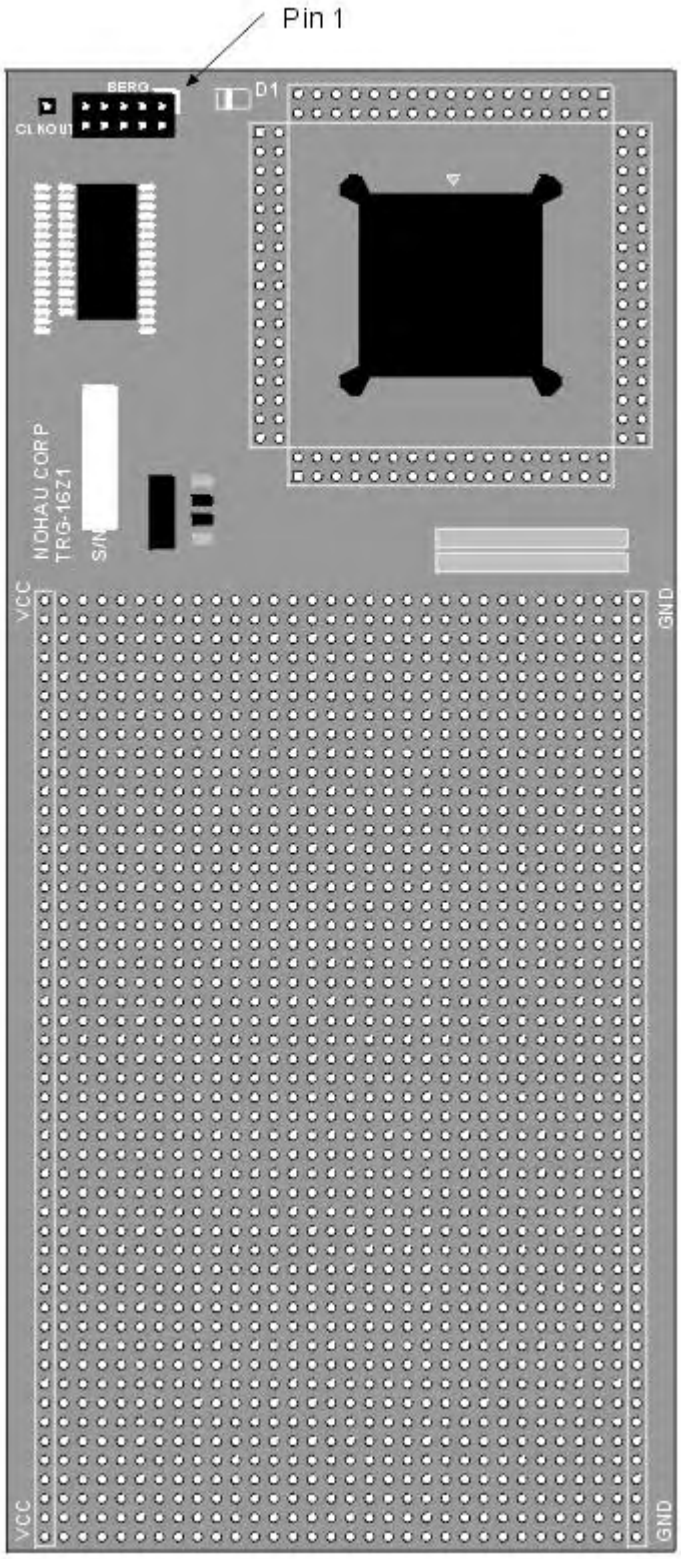


Figure 102. Target Board TRG–16Z1

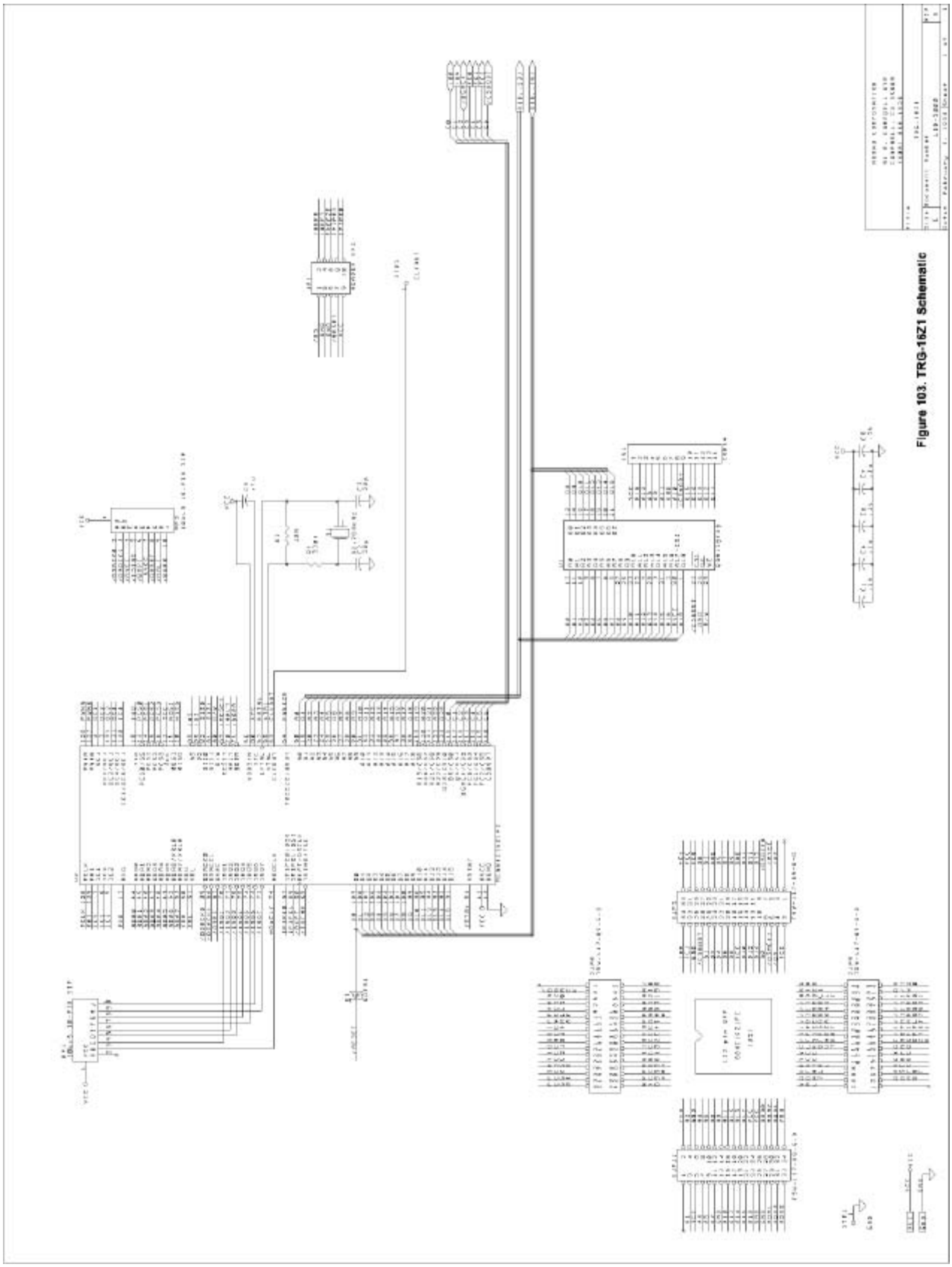


Figure 103. TRG–16Z1 Schematic

TRG–331 or TRG–332

This target board includes either a 68331 or a 68332, a 32K by 8-RAM chip, a 32-kHz oscillator, and a BERG connector. It also has a diode between the /RESET line and /DB0 so that the controller will come out of /RESET expecting an 8-bit device. The only difference between the TRG-331 and the TRG-332 is the controller.

Although the target uses 8-bit memory, selecting the 8-bit emulation memory option in the **Hardware Configuration** dialog box will not correctly configure the chip select registers at this time. Using the **User defined** option, the CSBARBT value should be set to either 0002 or 0003. A value of 0002 supports a 16384-byte device at address 0, which will make 16384 bytes of RAM unusable. A value of 0003 will make the entire chip usable, but because it is active for all addresses below 65535, address wrapping could occur between 32767 and 65535. It is up to you to choose either 2 or 3 and to put that value in the **CSBART** field, or to put some other appropriate value in that field (and possibly other fields) if you have changed RAM chips.

If you connect another RAM chip in the wire wrap area to /DB0 through /DB7, be sure to remove the diode shown in D1 below. This diode is used to pull /DB0 low during a /RESET, configuring the controller for 8-bit memory. Removing this diode will allow /DB0 to float, configuring the controller for 16-bit memory. Removing this diode will ensure that the reset vectors are read correctly, and will help ensure that the target will operate correctly when emulating through a /RESET

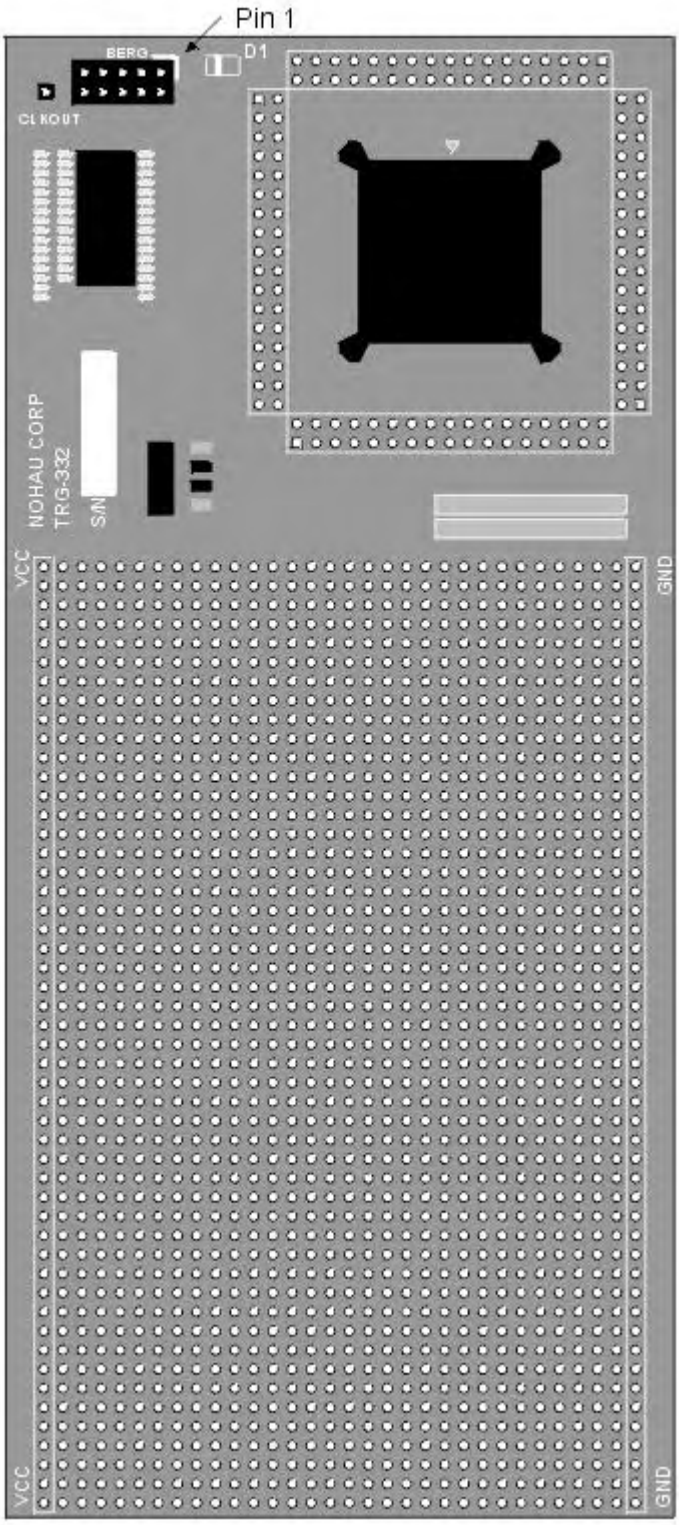


Figure 104. Target Board TRG–331/TRG–332

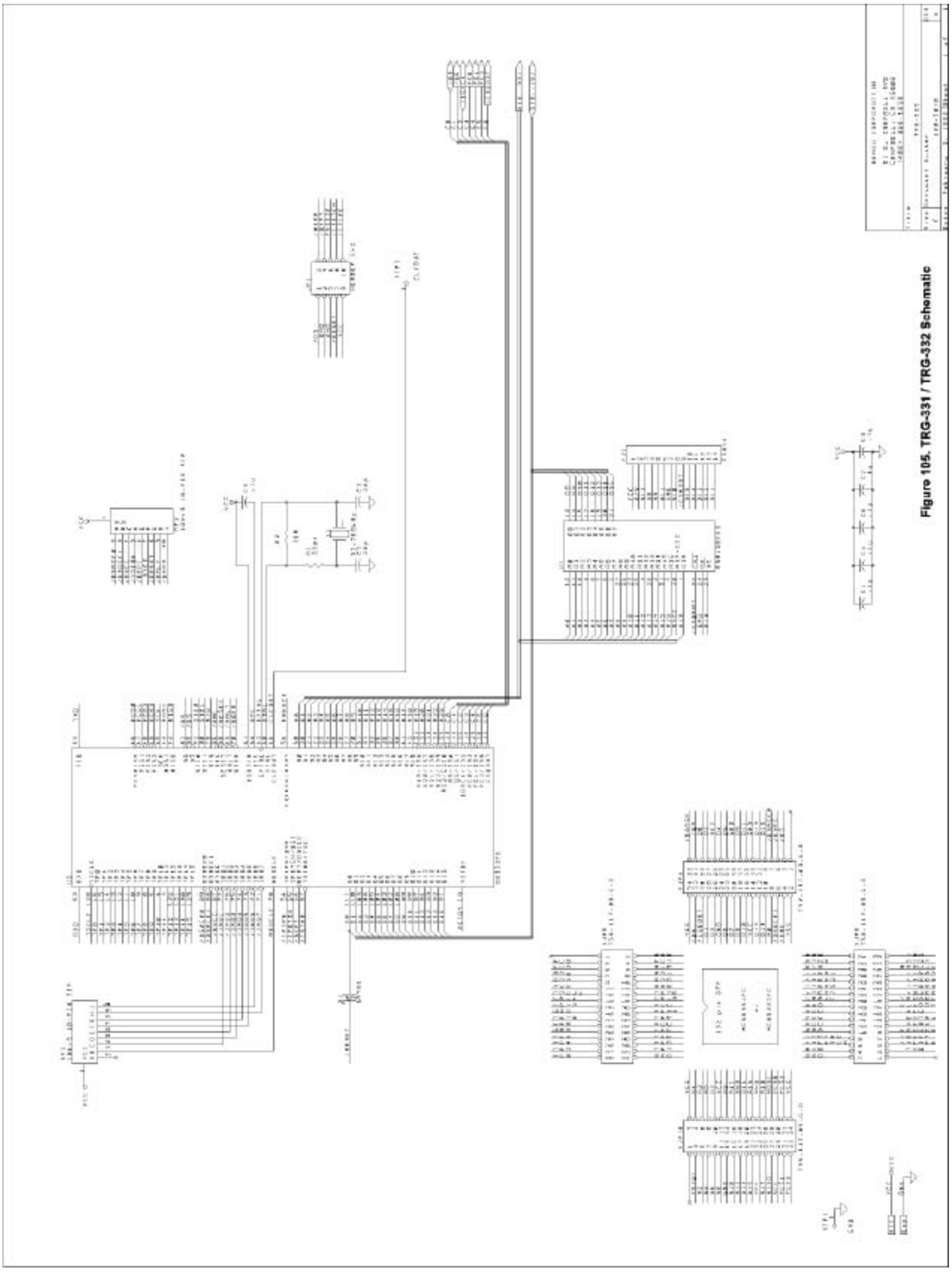


Figure 105. TRG-331/TRG-332 Schematic

TRG–333

This target board includes a 68F333, a 32K by 8-RAM chip, a 32-kHz oscillator, a BERG connector, and 5 diodes.

Three diodes configure the controller during the /RESET cycle. Diode D5 connects the /RESET line and DB0 so that the controller will come out of /RESET expecting an 8-bit memory device. If you connect another RAM chip in the wire wrap area to /DB0 through /DB7, be sure to remove the diode shown in D3 below.

Two other diodes, D3 and D4, connect the /RESET line to either DB14 or DB15 respectively to disable the internal EEPROMs on the 68F333. Each of these diodes runs through header JP2 so that when the jumper is in place, that on-chip ROM is disabled. With TRG-333 oriented as shown in the following figure, the top jumper controls the 16K EEPROM module and the lower one controls the 48K EEPROM module. If you wish to use EEPROM on the controller, remove one or both of the jumpers from header JP2.

Although the target uses 8-bit memory, selecting the 8-bit emulation memory option in the **Hardware Configuration** dialog box will not correctly configure the chip select registers at this time. Using the **User defined** option, the CSBARBT value should be set to either 0002 or 0003. A value of 0002 supports a 16384-byte device at address 0, which will make 16384 bytes of RAM unusable. A value of 0003 will make the entire chip usable, but because it is active for all addresses below 65535, address wrapping could occur between 32767 and 65535. It is up to you to choose either 2 or 3 and to put that value in the **CSBART** field, or to put some other appropriate value in that field (and possibly other fields) if you have changed RAM chips.

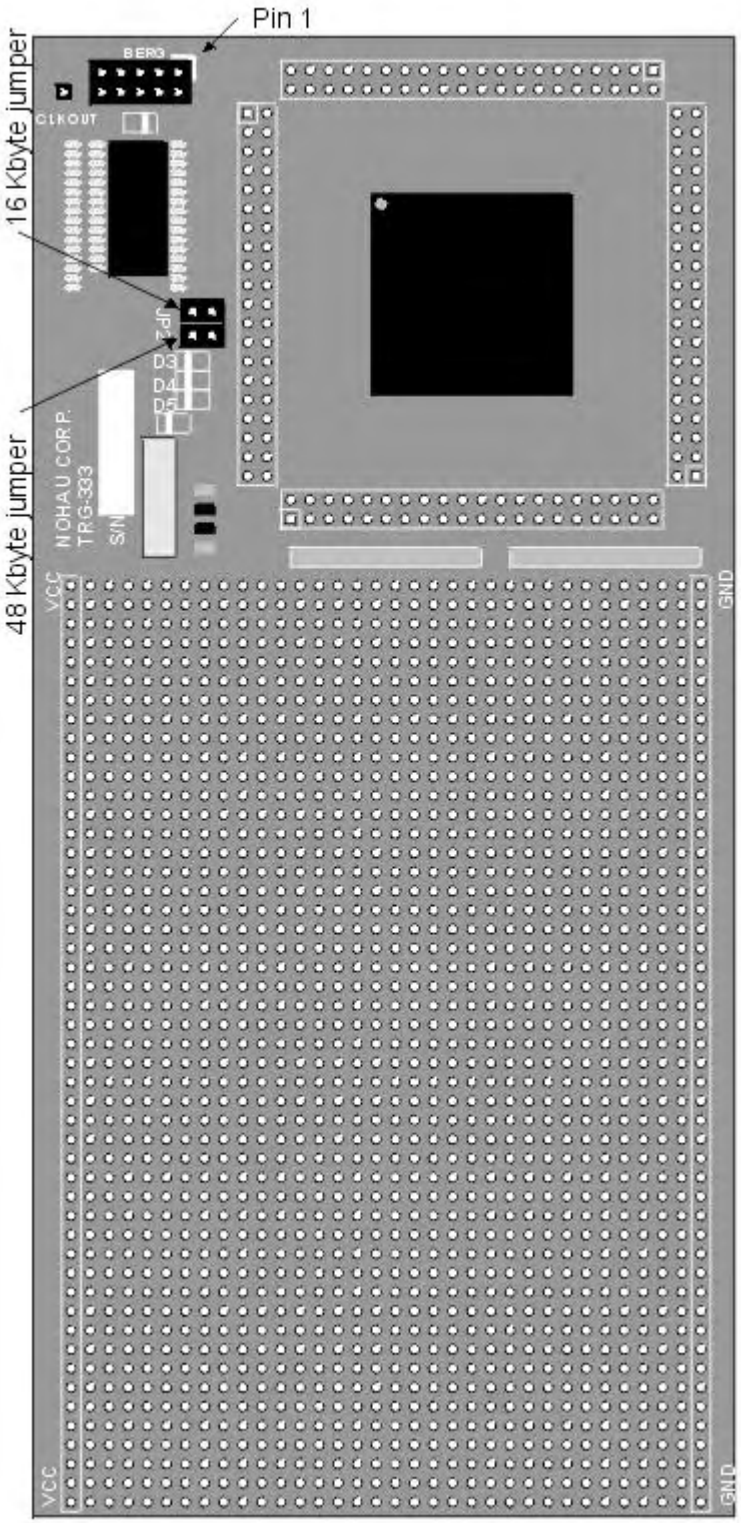


Figure 106. Target Board TRG–333

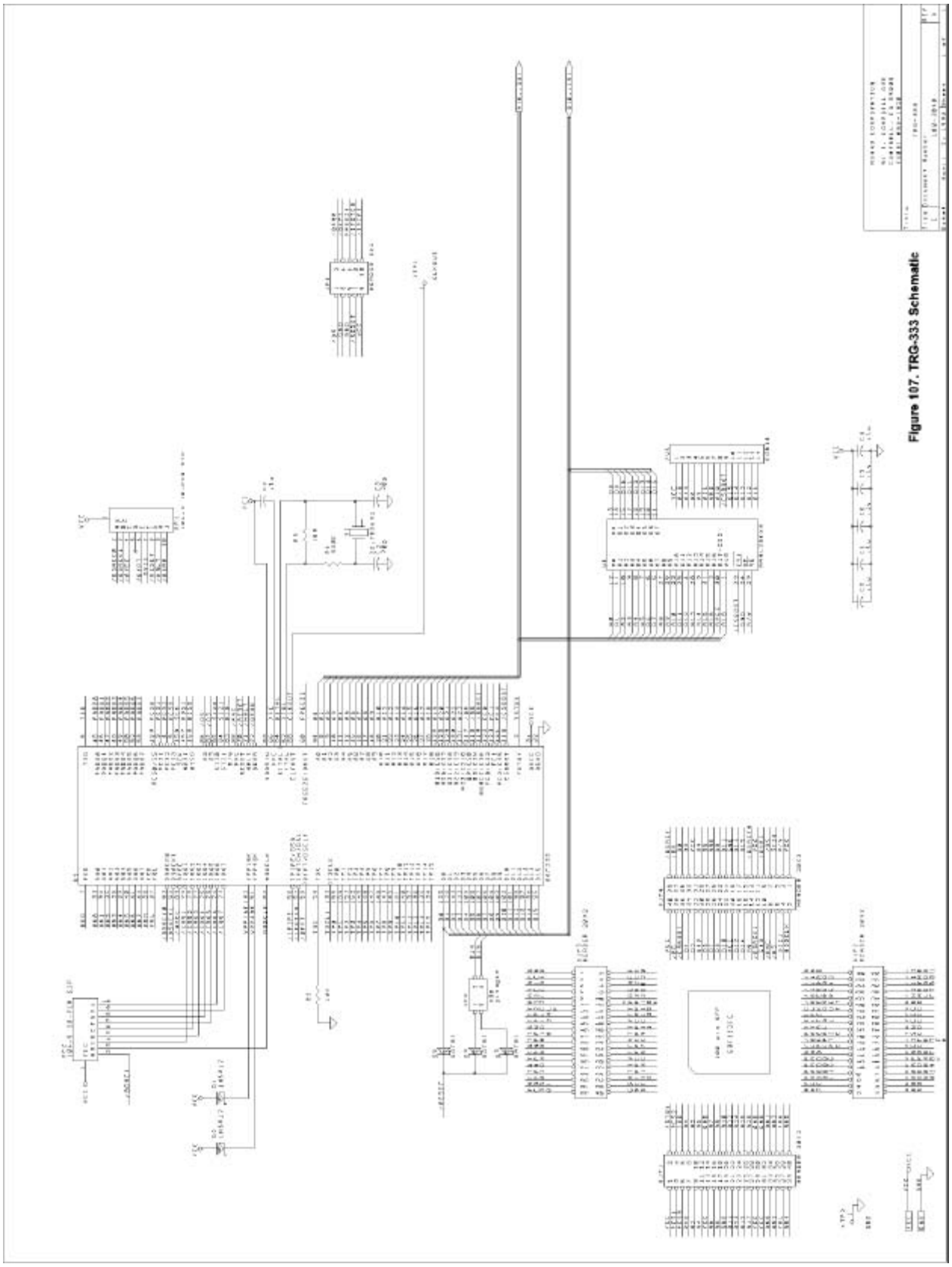


Figure 107. TRG–333 Schematic

TRG–340

This target board includes a 68340 controller, a 32K by 8-RAM chip, a 32-kHz oscillator, a BERG connector, and a header that controls RAM size.

Header JP2 connects the DSACK0 pin with the /CS0/AVEC pin. Thus wired, the controller will be configured for an 8-bit boot memory device (RAM or ROM) controlled by /CS0 every time the controller is reset (and before the EMUL300 software has a chance to update the chip select registers). If you connect another RAM chip in the wire wrap area to DB0 through DB7, be sure to remove the jumper on header JP2. Removing this jumper will allow DSACK0 to float, configuring the controller for 16-bit memory when the controller is /RESET. Removing this jumper will ensure that the reset vectors are read correctly, and will help ensure that the target will operate correctly when emulating through a /RESET.

With the jumper on header JP2, /CS0 will be active for all addresses and all address spaces after every /RESET and until the valid bit is set in the /CS0 base address register. To let the emulator configure /CS0 for the correct RAM size on TRG-340, click on the **User defined** option. Next click on the box next to the **MBAR** field, and set that field to the base address of any free 2000 byte page (ORd with the valid bit). The value FFFFF001 is often used. Next, place a check mark next to the row for /CS0. For the **Base address**, use 000000F1. For the **Address mask**, use the value 00007FF2. These values will configure the controller to assert /CS0 only for addresses below 8000 hex, and for all memory spaces. If you add RAM chips to the wire wrap area or replace the 32K chip for a larger one, be sure to change the chip select registers accordingly.

Note

The algorithm that automatically detects the kind of processor and the size of RAM may report that the CPU not recognized or not supported by this software version. If this happens, open the **Config Emulator Hardware** dialog box and manually set the pod type field to **POD-340**.

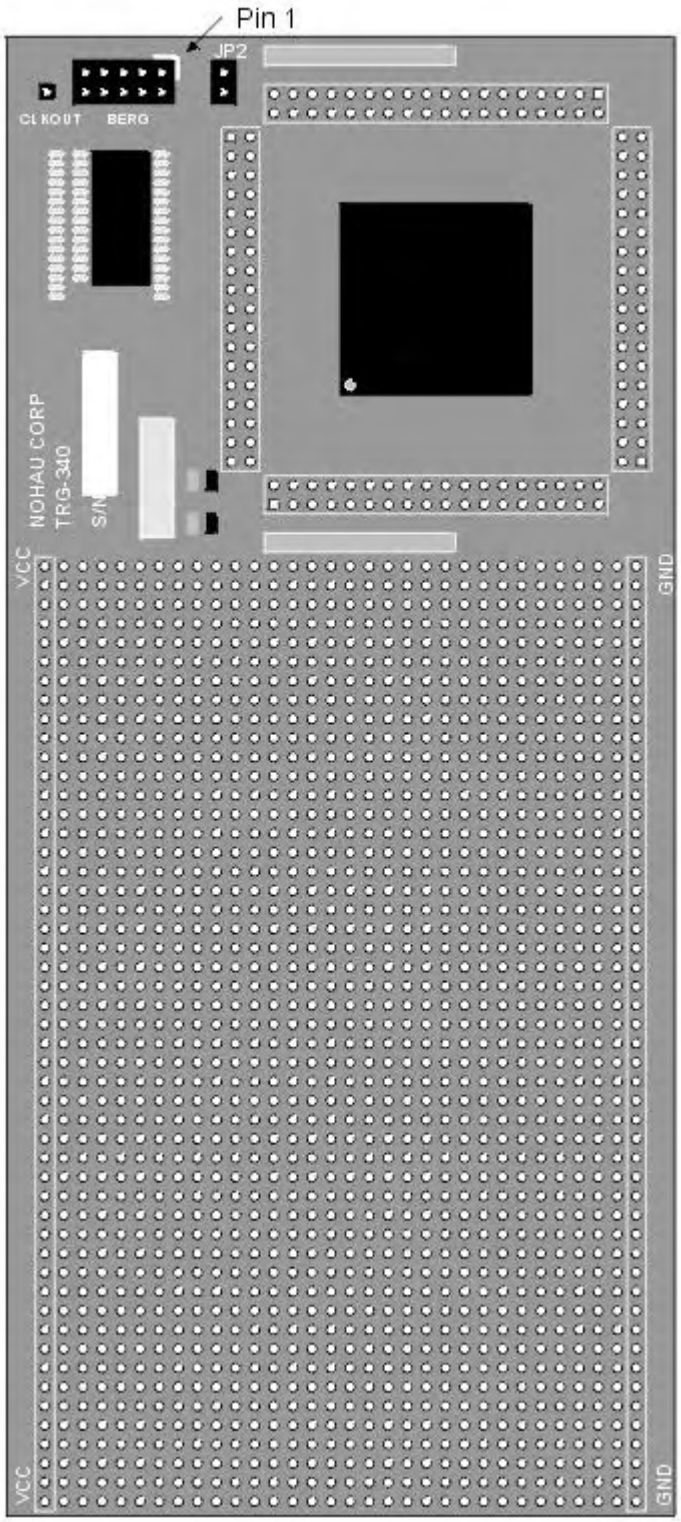


Figure 108. Target Board TRG–340

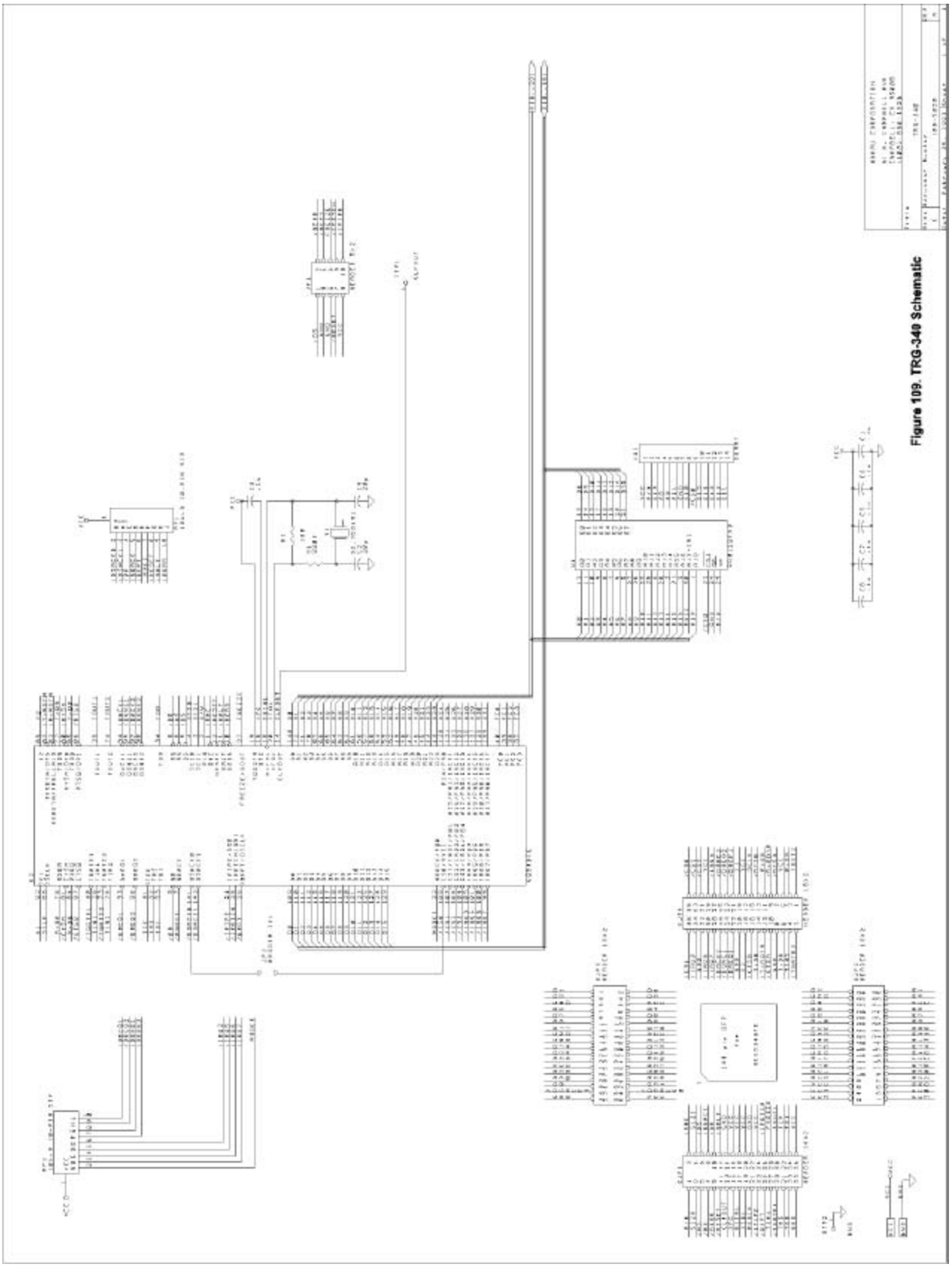


Figure 109. TRG–340 Schematic

Appendix I. Emulator / Trace Address Examples

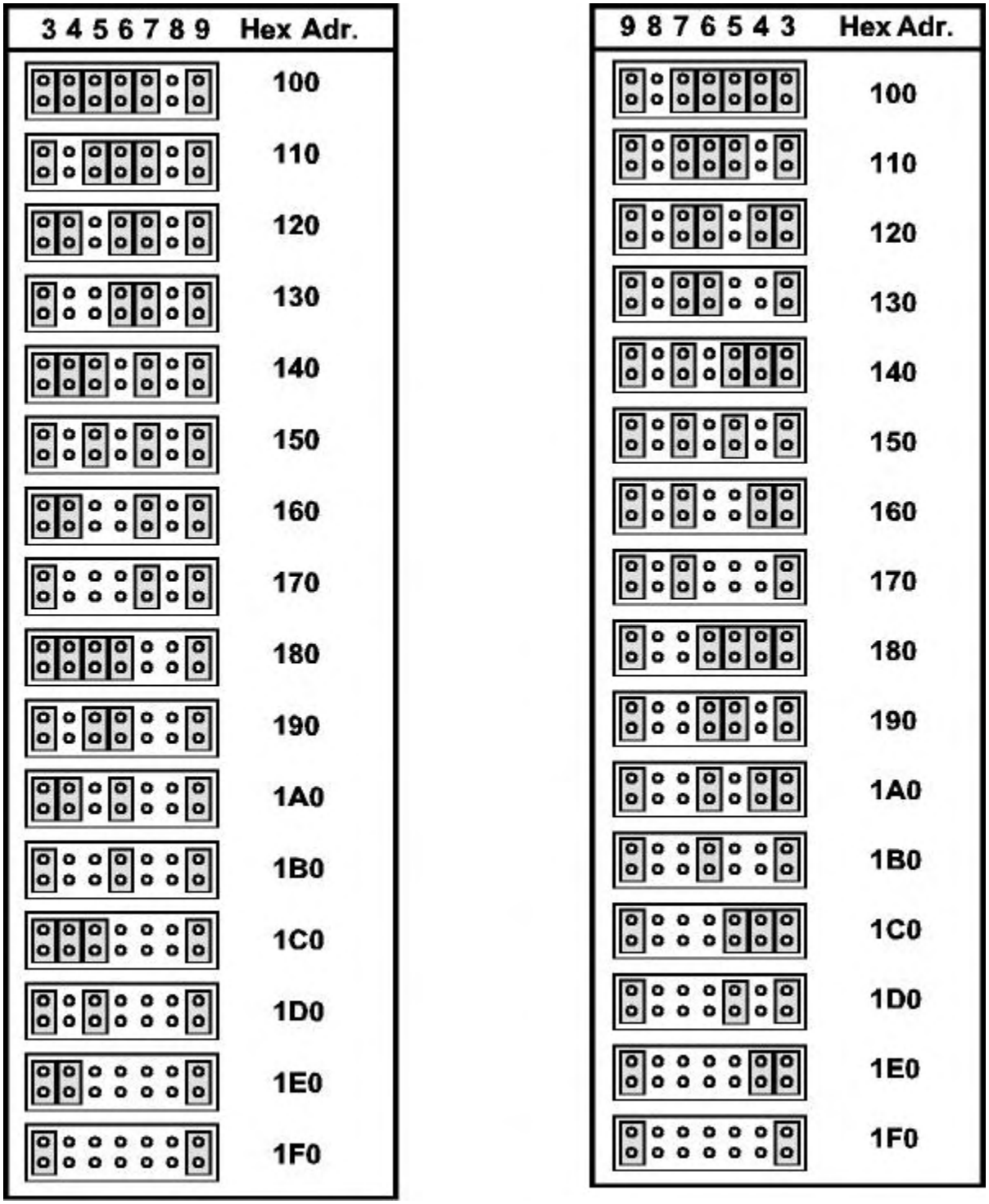


Figure 110.Pin Addressing 100 Hex Range

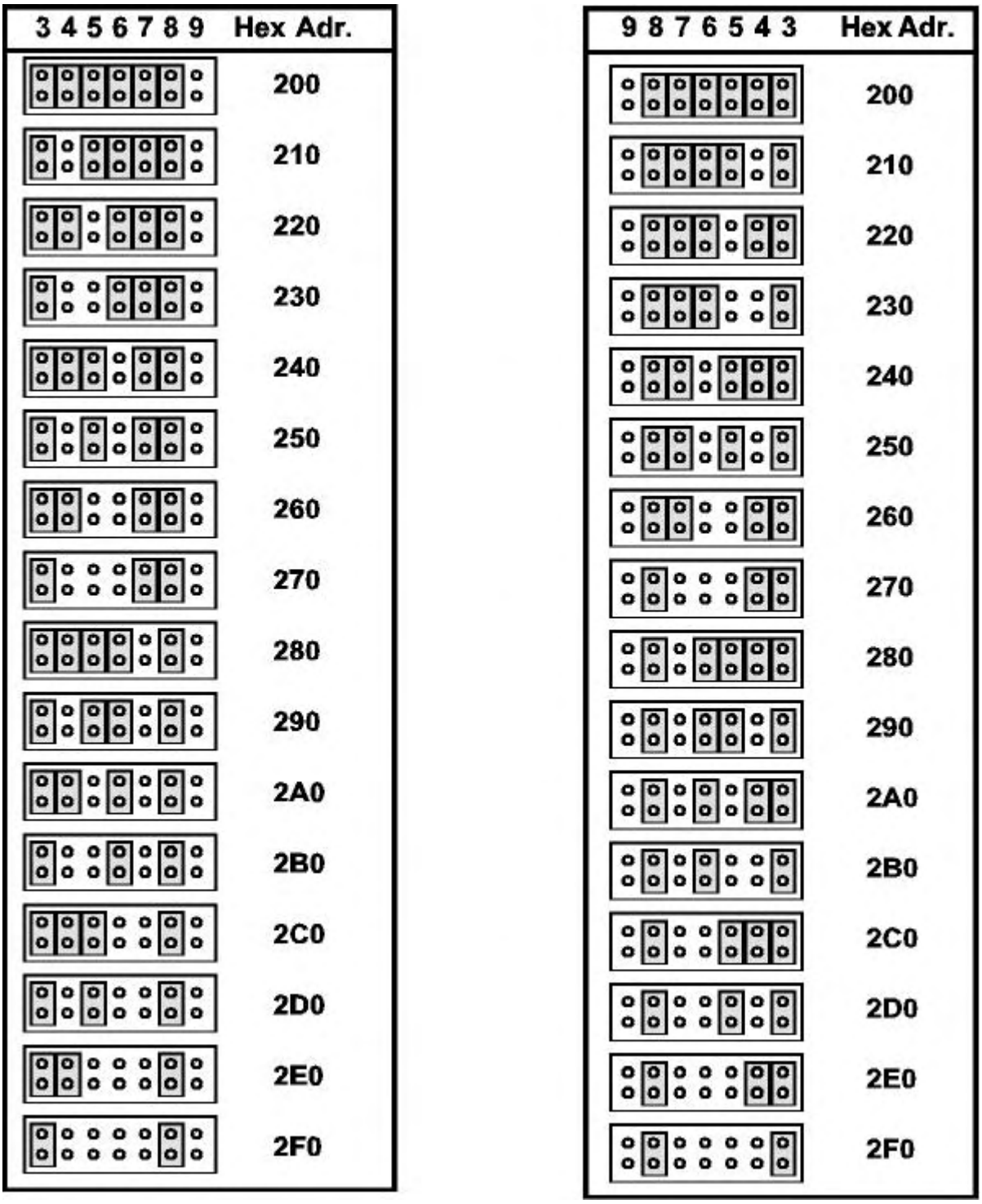
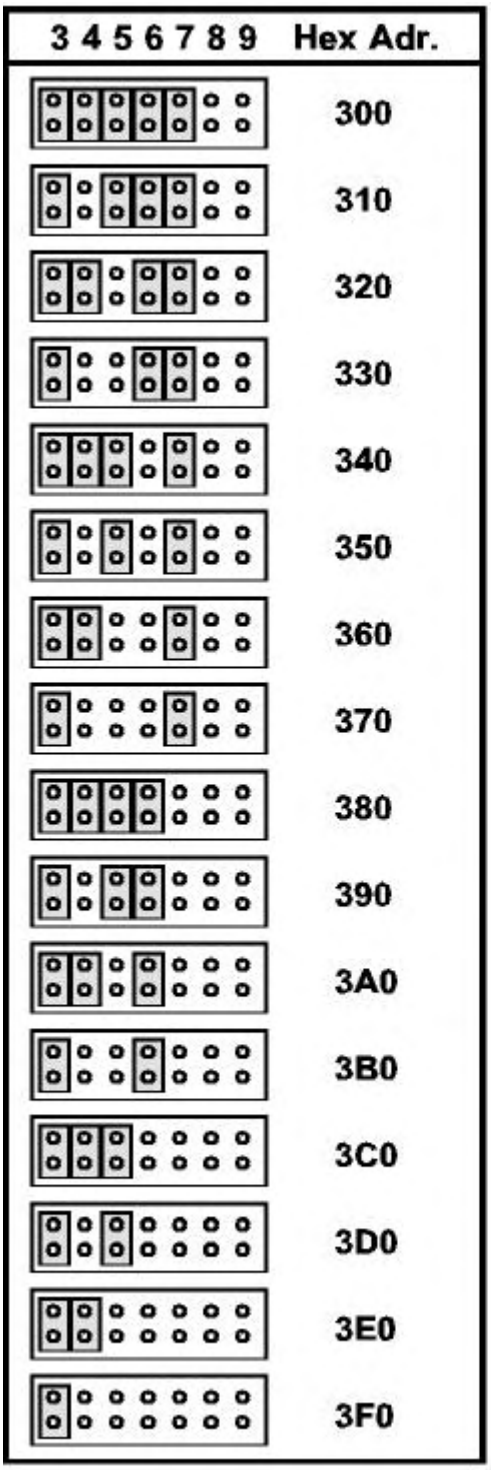


Figure 111. Pin Addressing 200 Hex Range



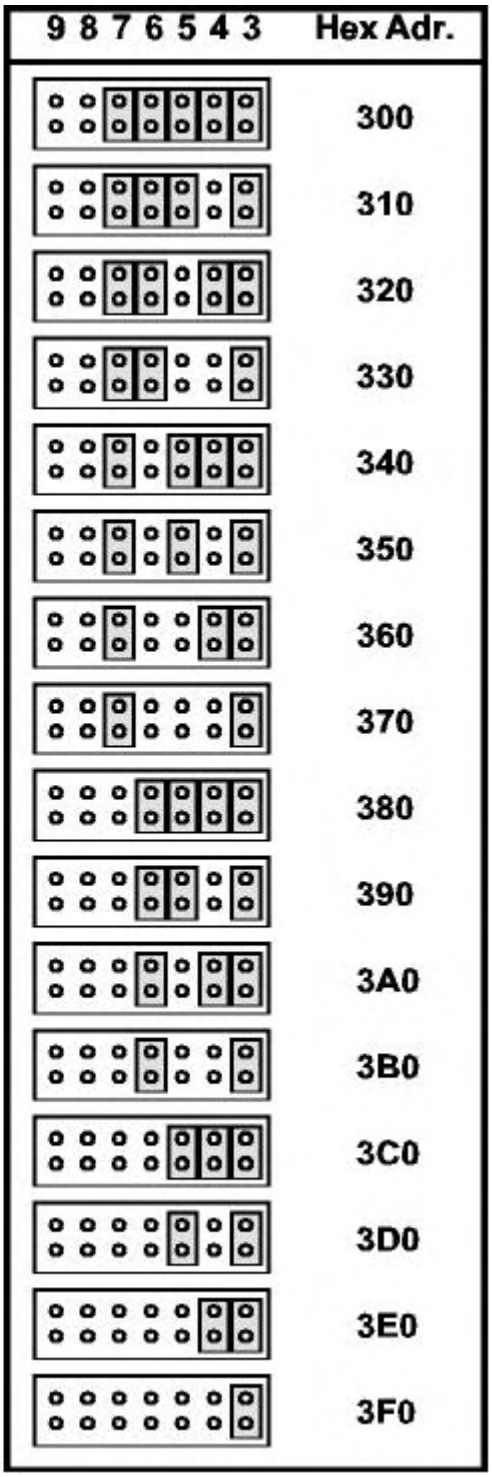


Figure 112. Pin Addressing 300 Hex Range

Glossary

A

ABS file

Many compiler assembler linker systems put information about the source code, such as symbol definitions and line numbers in the ABS file. Absolute load file generated by a linker operation after compiling/assembling the application code. A superset of the 8051 OMF. A file containing absolute (i.e. fixed location) data to load into a computer.

Adapter

The device that serves as an interface between the system unit and the devices attached to it. A device used for making an electrical connection between different package types, such as DIP40 to PLCC44. A connector or cable adapter, which changes one type of connector to another.

Address

Refers to where a particular piece of data or other information is found. Also can refer to the location of a set of instructions. Normally, but not always the words are 8-bit quantities. Some systems might have a program memory with 12-bit words and a data memory with 8-bit words.

Address Header

The range of memory locations that are addressable by changing jumpers on the emulator and trace boards.

Address Space

In Nohau emulators, there are often several address spaces. Each address space has a defined way of reading and usually writing to a memory. Several address spaces might address the same physical memory in different ways that are convenient for different usage. Some address spaces might address spaces that cannot be accessed by any other address space. Some address spaces, such as Shadow might exist only in the Nohau emulator and not in the target system.

ASCII

An acronym for American Standard Code for Information Interchange. A coding scheme using seven or eight bits that assigns numeric values up to 256 characters.

Assembler

An assembler is a program that converts a symbolic representation of computer instructions into the representation that the computer requires for operation. In Nohau emulators, the assembler window displays the contents of program memory as a numeric quantity, and in a symbolic form that is acceptable to an assembler for the processor being emulated. Information is lost in translation when an assembler processes a program. The symbolic form that the Nohau emulator disassembles might not be exactly the original assembler input that produced the numeric quantity found in memory.

B

Bank Number Translation

In bank switching design, cross-reference tables between bank number and control byte pattern. This is necessary if the bit sequence of bank switching control signals is scrambled and not in order with the bit sequence of control byte. This is not recommended.

Bank Switching

A method of expanding the Code Memory Space beyond that of microcontroller address bus limitation by creating additional high order address buses from a microcontroller I/O port or a memory mapped latch. The details vary widely. In general, one or more registers select one of relatively large continuous banks of memory to be accessed by a range of addresses.

*See also **Memory window**, **Page**, **Paging**, **Page register**, and **Window**.*

Banking

*See also **Bank Switching**, **Memory window**, **Page**, **Paging**, **Page register**, and **Window**.*

Base Register

In the Motorola HCl2, HCl6, and 683xx families a Special Register that sets that starting location of the block of Special Registers that control the processor and the on-chip input/output devices.

Basic CPU Register

The registers that Seehau initially shows in the Register window. Some of these registers, such as the PC and SP might not be accessible in any other way. Others can be Special Registers with a memory address that can be viewed and modified in several different ways in Seehau.

BDM

An acronym for Background Debug Mode. This is a debugging mode available on some families of processors supported by Nohau emulators. The production processors dedicate a small number of pins to the BDM functions. This allows a small cable to connect an emulator and a production system that includes the processor. The emulator can then control the processor for debugging in the production system. BDM emulator features are usually limited by the small size of the cable.

BERG connector

A 10-pin connector used to connect a BDM pod to a target system.

Big Endian

Having the bytes of a multi-byte number ordered with the most significant (biggest) byte first. Motorola usually uses this order.

*See also **Endian**, and **Little Endian**.*

Binary Data

Refers to the computer numbering system that consists of two numerals 0 and 1 Also called base-2.

BNC connector

An acronym for Bayonet Neil-Councelman, British Naval Connector, Baby N-connector, or Bayonet-Nut-Coupler. A connector for coaxial cables that locks when one connector is inserted onto another and rotated 90 degrees. Trace boards have two additional input/output connectors on the back called BNC connectors for TRIGGER-IN and TRIGGER-OUT recording.

Bondout

Short for bondout chip. A special variation of a processor that brings out (bonds out to additional pins) many extra internal signals. These additional internal signals allow the emulator to display and control internal states and functions that cannot be accessed in the mass production version of the processor.

Bondout Chip

*See also **Bondout**, **Bondout Emulation**, and **Bondout Pod**.*

Bondout Emulation

The emulation processor on the pod is a special bondout processor, which usually features more pins than a standard production processor. Also they are more expensive due to low production volume.

*See also **Bondout**, **Bondout Chip**, and **Bondout Pod**.*

Bondout Pod

A pod that has a bondout emulation processor.

*See also **Bondout**, **Bondout Chip**, and **Bondout Emulation**.*

Boot

To load a program into the computer. The term comes from the phrase pulling a boot on by the bootstrap.

Bootstrap

A technique or device designed to bring itself into a desired state by means of its own action. The term is used to describe the process by which a device such as a PC goes from its initial power-on condition to a running condition without human intervention.

*See also **Boot**.*

Break

To stop the execution of a processor in a way that allows the processor to resume execution as if nothing had happened.

Breakpoint

A debugging feature that breaks the processor at a particular location in a program or when a particular data item is accessed. This may be a software breakpoint or a hardware breakpoint.

*See also **Breakpoint Replacement**.*

Breakpoint Replacement

If enabled, the emulator will stop right before the breakpoint, otherwise it will stop right after the breakpoint. This applies only to hardware breakpoints.

*See also **Breakpoint**.*

BSW

*See **Bank Switching**.*

Buffer

A block of memory used as a holding tank to store data temporarily. A region of memory to hold data that is waiting to be transferred between two locations as between an application's data and an I/O device.

Byte

A collection of bits that makes up a character or other designation. Generally, a byte is 8 data bits. When referring to system RAM, an additional parity bit is also stored, making the total 9 bits.

Byte Order

The order of bytes in a word. Some processors (for example Motorola) store the most significant byte first and others (for example Intel) store the least significant byte first. It appears that there is no decisive advantage to either scheme. In machines without hardware support for words (for example the 8051) some compilers use one order and some use the other. One compiler even uses one order for integers and the other for floating point.

*See **Big Endian** and **Little Endian**.*

C

CCR

Acronym for (Emulator) Configuration Control Register. These registers are used to control the configuration of the emulator as contrasted with the registers in the system being emulated.

Chip

An integrated circuit. Used to refer to processor or memory integrated circuits.

Chip Select

A type of signal generated by some processors that is suitable for connecting to a chip select (CS) input of a memory chip. This allows the connection between the processor and the memory to be about as simple as connecting the address lines, the data lines, and the chip select.

Code coverage

A feature of some Nohau emulators that records the fetching (in the 196, 16, and 300) or the execution of instructions. Useful for evaluating the effectiveness of a test suite. If the test has executed an instruction once, this gives assurance that the instruction can be executed without fault in at least one case. If an instruction has not been executed at all, there is a danger that it cannot be executed without causing an error.

COFF

An acronym for Common Object File Format, a format for load files derived from the UNIX culture.

Command Line

The line on the display screen where a command is expected. Generally, the command line is the line that contains the most recently displayed command prompt.

Command Set

A named set of commands. A set of commands used to perform specific operations / tasks with the emulator. A list of instructions recognized by a microcontroller.

Compiler

A program that converts a symbolic description of a computer program into a form the computer can execute. Compilers are distinguished from assemblers in that they accept input that is not directly related to the actual machine instructions. The output from a compiler is called an object program. Most Nohau emulators support C and C++ compilers by reading extra information provided by the compiler. This allows the emulators to display the compiler source code that corresponds to a program location and to display the values of variables in a style appropriate to the way they were defined in the source program.

Core Command

Hardware specific operation command issued by the user interface to perform a specific operation. A Seehau command that is processed by the core the part of Seehau that deals with the target hardware, file loading, and keeping the symbol table. Contrast a window or GUI (Graphical User Interface) command that only affects the appearance of a window.

CPU symbol

Default symbolic reference to the microcontrollers Standard CPU registers. Symbols used in the microcontroller CPU architecture definition, especially for Special Function Registers (SFRs), usually related with a SFR physical address.

CPU

An acronym for Central Processing Unit. The computational and control unit of a computer; the brains. This device interprets and executes instructions.

Crystal

A frequency determining element. A resonating crystal used in a clock circuit for the microcontroller. A piece of silicon that oscillates at a predetermined frequency. An oscillator of some kind drives all microcontrollers. The device on the pod or the target that provides time base for the clock generation circuitry in microcontroller. Usually required to be connected to two crystal pins on the microcontroller. It determines the operation speed for the microcontroller. There is usually a frequency multiplier or divider involved, i.e. 5-MHz crystal *4 (multiplier) = 20-MHz System Clock.

*See also **Internal Crystal** and **External Crystal**.*

Cycle Type

A named sequence of events. A category of instruction action in a machine cycle usually related with a chip-select or strobe signal issued by the microcontroller. For example, in the 8051 family, there are Opcode Fetch Cycles (/PSEN), XData Write Cycles (IWR), and XData Read Cycles (/RD). Cycle type refers to the default symbol table that has been defined by Nohau for the microcontrollers internal registers.

D

D connector

*See **DB-25 connector**.*

Data Bus Width Override

Determines how much data can be transmitted at one time. For example, a 16-bit bus can transmit 16 bits of data, whereas a 32-bit bus can transmit 32 bits of data.

Data Window

1. A window in Seehau user interface that displays data stored in memory.

2. 68HCl 2 MCU chip feature that allows a section of the 16-bit address space to address pages of data space. The DPAGE register controls the page visible in the window.

DB-25 connector

A 25-pin D-shell connector primarily used for PC parallel ports. The mechanical interface of a 25-wire cable with a male (M) and female (F) DB-25 pin connector attached to either end. A plug with 25 pins or receptacles, each of which is attached to a single wire with a specific function. Originally called an RS-232 (now EIA-232), the latest which defines not only the type of connectors to be used but also the specific cable and plugs and the functionally of each pin. In Nohau culture also referred to as a D connector.

Debug File

A file generated by C Compilers / Assemblers, which contains both code and symbol cross-reference in the source file. The file usually has a special filename extension such as OMF, ABS, or no extension at all. To get symbol information, the debug file should be loaded into the Seehau software. Hex files will not provide symbol information.

Delay

The number of trace frames collected after a trigger event occurs.

Dialog box

A special window displayed by the system or application to solicit a response from the user.

Double

A C floating point type usually represented in 64 bits.

DPRAM

An acronym for Dual-ported RAM. DPRAM is a dual-ported random access memory with separated ports of different bus width for READ and WRITE function. The bus width is 1 bit for READ and 16 bits for WRITE. The READ and WRITE access can be performed simultaneously or at independent clock rates at frequencies up to 50 MHz.

DRAM

An acronym for Dynamic Random Access Memory. A form of semiconductor random access memory. Temporary storage that must be refreshed over time.

DWARF

An acronym for Debug With Arbitrary Record Format. The debugging information format associated with the ELF load file format. Designed for the better support of C++.

*See also **ELF**.*

E

Edge connector

The part of a circuit board containing a series of printed contact that is inserted into an expansion slot or connector. The part of the expansion board that is inserted into the motherboard.

*See also **Expansion Board**.*

EEPROM

An acronym for Electrically Erasable Programmable Read Only Memory. A type of nonvolatile memory chip used to store semipermanent information. An EEPROM can be erased and reprogrammed directly in the host system without special equipment. Read Only Memory (ROM) that can be electrically erased and programmed too repeatedly. Also called flash ROM.

*See also **Flash Memory**.*

ELF

An acronym for Executable and Linkable Format. A file format for program and data to be loaded into a processor. Usually associated with the DWARF debugging information format.

*See also **DWARF**.*

Emulator

A piece of test apparatus that emulates or imitates the function of a particular chip. This device runs the code in the same way and at the same speed as the real microcontroller. The emulator facilitates debugging by providing more information about internal operations of the microcontroller. The emulator gives better control of operations and faster more flexible loading of programs.

Emulator Board

An ISA card, which is plugged into the PC or an HSP box with emulation memory on board. It is connected to a pod through a special flat cable. In some emulators the functionality of the emulator board is integrated into the pod so, it might not be available separately.

Emulator Memory

Memory internal to the emulator. Some emulator memory may be used for the internal purposes of the emulator that will not normally be visible to the emulator user. Memory provided in the emulator that makes code debugging possible without a target. Overlay memory for the emulator to simulate the memory for On or Off chip memory.

Emulation Mode

The emulation processor on the pod running customer application code to enable it to simulate a real target processor.

Emulation Processor

The processor provided on the pod, which is used to emulate a target processor. It might or might not be the same as the target processor.

Endian

The bytes, which are most significant in a multi-byte word.

*See **Little Endian** and **Big Endian**.*

Environment settings

User interface setups for the debug environment such as load path, source file path(s), file load options, etc.

EPAGE

A register in some Motorola 68HCl 2 family members that controls what page of Extra memory it mapped into the Extra page window. See the Motorola User's Guide for the particular family member for further information.

EPC

An acronym for Emulator Parallel Cable. A Nohau product that attaches an emulator to a laptop computer parallel (printer) port.

EPROM

An acronym for Erasable Programmable Read Only Memory. A type of ROM that can be erased by exposure to ultraviolet light and programming repeatedly.

*See also **EEPROM**.*

Evaluate/Modify window

A Seehau user interface window that allows evaluation and modification of existing expressions or C and C++ variables using the symbols of the currently loaded program.

Expanded Mode

*See **External Mode**.*

Expansion Board

Any board that plugs into one of the computer's expansion slot. Expansion boards include controller boards, trace boards, and emulator boards.

Expansion Card

*See **Expansion Board**.*

External Crystal

The crystal on the target is used for the crystal source of emulation processor.

External Mode

The code and or data memory is external to the microcontroller. Some ports on the microcontroller are functioning as data bus and/or address bus.

See also ***Single-Chip Mode****.*

External Mode Pod

A pod whose emulation processor works in External mode. The same as a standard production processor.

External Power

The emulation processor on the pod gets power from the target. Recommended for Bondout pods and Low Voltage Emulation. The external power can be used only when the target is connected to the pod.

Extra Window

A 68HC12 MCU chip feature that allows a small section of the 16-bit address space to address pages of extra space. The EPAGE register controls the page visible in the window.

F

Fast Break Write

A feature of some Nohau emulators that allows writing to target memory while the target is running. It causes minimal interruption to the target execution. It does stop (break) the target processor for a short time.

Filter

A set of conditions that determine which frames are allowed into the trace buffer. Filtering selects the type of information in an address range, and the type of data that is recorded in the trace memory. This is a distinct action separate from Filter mode.

Filter Mode

This mode is different from filter.

See ***Window Filter*** *and* ***Normal Filter****.*

Flash ROM

Flash ROM, also known as flash. A type of EEPROM that has been optimized for faster (flash) erasing and programming. See also EPROM and EEPROM.

float

A C type for floating point numbers. Often 32 bits in length.

Floating Point

The computer version of Scientific Notation for numbers. A fraction and an exponent represent the number.

Frame

A unit of information in a trace buffer. Usually a record of a target memory reference, but can record other events such as a low power state or the passage of time.

Frame Number

An arbitrary number assigned by the emulator to each frame in a trace buffer. Positive frame numbers occurred after the trigger and negative frame numbers occurred before the trigger. If a trigger did not stop tracing, the last recorded frame number is -1.

Frequency Multiplication

Multiplying external clock input signal's frequency by a factor and feed it to the CPU inside a microcontroller. Often done by a Phase Lock Loop (PLL) circuit inside the microcontroller. Electromagnetic Interference (EMI) is reduced this way. A factor used by a microcontroller to multiply the crystal (oscillator) frequency, i.e. 5-MHz crystal *4(multiplier) = 20-MHz System Clock.

Frequency limit

The maximum or minimum frequency at which an emulator is designed to operate properly.

Full Emulator

An emulator with the ability to add trace capabilities. Full emulators often use the BDM pins on the processor to do some of the emulator functions in addition to providing tracing, shadow memory, and more breakpoints.

G

GPT

An acronym for the General Purpose Timer, a hardware feature found on some Motorola embedded processors.

GUI Command

An acronym for Graphical User Interface. A program interface that allows users to choose commands and functions by pointing to a graphical icon using either a keyboard or pointing device such as a mouse, trackball or touch pad. Seehau is divided into a GUI part that handles the display and control for the user. This part is common to all Nohau emulator families. The Seehau GUI

sends commands to the core which is specific to a particular family of processors. These commands can be recorded in macros. To allow macros to affect the display, the GUI also sends commands to itself. These are called GUI commands.

H

Hardware Breakpoint

A breakpoint function implemented in hardware, either in a processor chip or in an emulator. The distinguishing feature is the hardware, which does not require any software modification to place the breakpoint. This allows breakpoints to be effective in ROM of all kinds.

*See also **Breakpoint**, and **Software Breakpoint**.*

Hex

Hexadecimal, a number encoded as base-16 instead of base-10. Widely used for display of memory addresses and hardware registers because humans and computers more easily translate it into bits than standard decimal notation.

*See also **Hexadecimal**.*

Hex File

An ASCII file consisting of a number of hex records, which represent machine language code and/or constant data with hexadecimal numbers. A load file or absolute file with the data in hexadecimal numbering (base-16). It is usually used to transfer the program and data that would be stored in a ROM or EPROM. No symbol/debug information is included in the hex file. Nohau supports a standard hex file format for each family, Motorola S-Record or S19 for Motorola families, and Intel Hex for Intel families. Other manufacturers adopt one or the other as their hex file format standard.

Hexadecimal

A numbering system used in computers. 16 characters: 0 through 9, and A through F (upper or lower case) to represent the numbers 0 through 15. One hexadecimal digit is equal to 4 bits, and one byte is two hexadecimal digits.

*See also **Hex**.*

High Speed Parallel Box

*See **HSP**.*

Hooks Emulation Mode

The emulation processor on the pod is a standard production microcontroller, but is put into a special Hooks mode, which makes the emulation possible.

Hooks Mode

A special emulation mode that is built into specific microcontrollers. A special operation mode, in which the microcontroller provides extra internal signals through unused bus cycles, emulator circuitry also controls the microcontroller through these cycles. It is an extra feature built into the microcontroller. An emulation mode peculiar to some 8051 derivatives.

Hooks Mode Pod

A pod whose emulation processor works in Hooks Emulation mode.

HSP

A box with its own power supply, which can hold an emulator board and, or an optional trace board. Manufactured by ICE Technology, the HSP box contains a motherboard with three ISA slots. The HSP box allows the use of the in-circuit emulator and optional trace board when no ISA slots are available in your PC. The HSP connects to the PC printer port, so it can be used with a laptop computer or a standard PC. It is valued for power-up, and power-down emulator convenience.

See also ***High Speed Parallel box****.*

HSP card

An ISA board in the HSP that connects to a PC through a cable from the serial interface on the card to the parallel port on the PC or laptop computer.

I

I/O

Input/Output. A circuit path that enables independent communications between the processor and external devices.

I/O Port

Input/Output port. Used to communicate to and from devices, such as a printer or disk.

IC

An acronym for Integrated Circuit. A complete electronic circuit contained on a single chip.

See also ***Chip****.*

IEEE-695

A widely supported load file format specified by IEEE (Institute of Electrical and Electronics Engineers) standard 695.

Inspect Window

A Seehau user interface window that displays current value of the selected item. The window is updated each time emulation breaks. It can be used for evaluation and modification of some expressions as well as of C and C++ variables using the symbols of the currently loaded program.

int

A C type for integer

Integer

A whole, or ordinal number. A number without a fractional part, for example 1.

Intel absolute object format

Absolute file format defined by Intel. (INTEL OMF) A debug file format defined by Intel Corporation in 1982. A superset of the 8051 OMF.

*See also **ABS**.*

Intel Byte Order

Having numbers represented by a sequence of bytes with the first byte holding the least significant 8 bits of the number.

*See also **Little Endian**.*

Intel Hex

A load file format. This format is a text file with the addresses and the data to be loaded in hex. This format makes no provision for communicating symbol values to the emulator software.

Internal Crystal

The factory default crystal on the pod is used for the crystal source of emulation processor.

Internal Power

The emulation processor on the pod gets power from the emulator board. Recommended in all times except for bondout pods and low voltage emulation.

Internal Symbol Table

The default, built in symbols.

*See also **Symbol Table**.*

Interrupt Vector Table

A table, which keeps a cross-reference between interrupt source and the starting address of the corresponding interrupt service subroutine.

*See also **Vector Table**.*

ISA

An acronym for Industry Standard Architecture. The original IBM PC-AT plug-in card format and bus structure. Some Nohau emulators plug into this bus.

ISA slot

A connection socket for a peripheral designed according to the ISA standard that applies to the bus developed for use in the 80286 motherboard.

Isolator

An adapter that has many small switches in the middle so you can connect or disconnect, signals selectively. Used in target connection trouble shooting.

J

Jargon File

A dictionary widely available on the Internet that defines many obscure informal computer terms, such as Big Endian and Little Endian.

Jumper

A small, plastic-covered metal clip that slips over two pins protruding from a circuit board. Sometimes also called a shunt. When in place, the jumper connects the pins electrically and closes the circuit. By doing so, it connects the two terminals of a switch, turning it on. A group of jumpers are referred to as a jumper block.

L

LED

An acronym for Light Emitting Diode. A semiconductor diode that emits light when a current is passed through it.

Linker

A program that takes relocatable object files produced by compilers and assemblers and combines them with precompiled library programs into a load file that can be loaded into a target processor.

The linker usually puts the symbols defined in the source programs into the load file in a way the emulator loader can decode.

Little Endian

Having the bytes of a multi-byte number ordered with the least significant littlest byte first. Intel usually uses this order.

*See also **Big Endian**.*

Load file

A file that contains a program in binary form to be loaded into the target. Nohau emulators support many formats of load files. Most formats include the final translated values of the symbols used in the original source files.

Locator

A term used for a linker that takes directives for setting the locations in the target processor of the various arts of the program.

long

A C type for integer that is at least as big as int. Usually 32 bits or 64 bits.

Low Voltage Emulation

The target works at a voltage lower than 5V DC, such as 3V DC. External power is required.

LPT Port

Line Printer port, a common system abbreviation for a parallel printer port.

LSB First

An acronym for Least Significant Bit first. Having numbers represented by sequence of bytes with the first byte holding the least significant 8 bits of the number.

*See also **Little Endian**.*

M

Macro

A set of keystrokes and instructions recorded and saved under a short key or macro name. Used to save time by replacing an often-used, sometimes lengthy, series of strokes with a shorter version.

Maximum frequency

The frequency limit on the pod.

MCU

An acronym for Micro Controller Unit. Industry jargon for a single chip computer of some kind.

Memory

The storage parts of a computer. Usually organized into addresses, which pick one of many locations that each hold, the same number of bits. Often the contents are bytes of 8 bits.

Memory Dump

The hexadecimal representation of an area of memory. The copying of raw data from one place to another with little or no formatting for readability. Usually, dump refers to copying data from main memory to a display screen or a printer. Dumps are useful for diagnosing bugs.

Memory Image

A copy of one area of memory in another area of memory.

*See also **Shadow RAM**.*

Memory Mapping

Controls the operation of selection between emulation memory and the users target memory. Any memory address may be either mapped to emulator memory or target memory. If it is mapped to emulator memory, it will be mapped to RAM on the emulator itself and the memory space on the target will be ignored. If it is mapped to target, whatever device on the target (RAM, ROM, I/O) will be used and the corresponding emulator memory (RAM) will be ignored.

(memory) Page

A section of a memory with a larger address range that is accessed in a smaller window (Q.V.) in an address space with a smaller address range. The page (number) or page register supplies the most significant bits of the larger address, and the address in the smaller window supplies the least significant bits of the address.

Individual families of processors, and individual models within families have unique variations in the details of paged addressing schemes. If you have to understand a particular scheme, read the manufacturer's documentation carefully.

See also banking, bank switching, memory window, page, paging, page register, and window.

Memory Space

A range of memory that is accessible to a microcontroller. Used for different purposes, but can occupy the same addresses in memory. The number of address lines used usually determines the size of this space. The property of physically or logically separate memory blocks, which are accessed by different type of instructions such as code, external data, and internal data.

Menu

A list of options from which a user can select in order to perform a desired action.

MHz

An abbreviation for megahertz, a unit of measurement indicating the frequency of one million cycles per second.

Micro-clip

A series of wires connected to the DB-25 connector on one end and small clips attached to the wires on the other end. Rather than a ribbon cable there are individual wires emanating from the connector that can be used for input and output data. The ends with the small clips can be attached to the target system.

Mixed mode

A display format for Seehau source and trace windows where the source lines are displayed with the disassembled instructions that were compiled from the source line.

Monitor Mode

The emulator actions are being observed in this mode. The emulation processor is running Emulator Monitor Code. Microcontroller specific code that runs when the emulator is not running the user application code. When there is no program execution going on, but we still have to access memory, registers and set up windows, etc. Monitor mode runs everything *except* the target program execution.

Motorola absolute object format

Absolute file format defined by Motorola. (MOTOROLA COFF)

Motorola Byte Order

Having numbers represented by a sequence of bytes with the first byte holding the most significant 8 bits of the number.

See also ***Big Endian***.

Motorola Families

Groups of microcontrollers that are closely related in characteristics that are manufactured by Motorola.

MSB First

An acronym for Most Significant Bit first. Having numbers represented by a sequence of bytes with the first byte holding the most significant 8 bits of the number.

See also ***Big Endian***.

N

ncore.log

Log file produced by the operation of emulator, contains information of the data passed between the CORE level program and the users interface. The **Log to file** check box in the **Environment Configuration, Options** tab controls the writing of this log file. The file is written to the directory where Seehau.exe is installed.

New Hacker's Dictionary

A very useful dictionary of computer jargon published by the MIT press.

*See also **Jargon File**.*

Normal Filter Mode

In this mode the last enabled trigger can be assigned a repeat counter that causes the trace to look for this last trigger a number of times before a trigger is recognized.

O

OLE automation

An acronym for Object Linking and Embedding. A distributed object system. The ability for an external program, that is or contains a programming language, to control another program.

OMF file

A debug file format, abbreviation of Object Module Format. Object Meta File-industry base standard. A load file format.

Oscillator

A self-contained device which generates a clock signal of a specified frequency without the assistance of external feedback circuitry. Its output usually can be fed directly to the clock-input pin of a microcontroller.

P

P & E

A manufacturer of software. A symbol file format used with Motorola MCU's is named for them.

Page

A fixed-size block of memory whose physical address can be changed through mapping hardware.

Paging

A method of expanding a computer's memory beyond the limits of an address size. A technique for implementing virtual memory. The virtual address. The virtual address is divided into a number of fixed-sized blocks called pages, each of which can be mapped onto any of the physical addresses available on the system. One or more registers select one of relatively large continuous pages of memory to be accessed by a range of addresses. Sometimes used as a synonym for bank switching.

Paged Addressing

See also ***Bank Switching****.*

PC

Common industry acronym for Program Counter, but also used to mean Personnel Computer.

See also ***Program Counter****.*

Pipelined Architecture

A computer architecture that speeds up the instruction execution rate by executing each instruction in stages, and executing different stages of several instructions at the same time. A common set of stages is Instruction fetch, data fetch and data store. In a computer with this type of pipelined architecture a single instruction would progress through these stages in sequence.

At the same time the computer might be doing:

- fetch of instruction 3
- data fetch for instruction 2
- data store of the results of instruction 1
- At the next cycle the computer would be doing
- fetch of instruction 4
- data fetch for instruction 3
- data store of the results of instruction 2

A confusing side effect of this architecture is that the memory references for instructions and data, may not be in the order one would expect from what instructions the computer is executing. (Notice that in the preceeding example, instruction 3 is read from memory before the results of instruction 2 are stored.)

The trace feature of Nohau emulators for processors with pipelined architecture make an effort to clarify this confusing situation by giving either a hardware view showing the memory references in

the actual order on the memory bus, or a software view showing the memory references as they are logically executed by the computer.

For historical reasons, the terms for the hardware and software views are not uniform among the different computer families supported by Nohau.

PLCC

An acronym for Plastic Leaded-Chip Carrier. A popular chip-carrier package with J-Ieads around the perimeter of the package.

Plug-and-sleeve connector

A connector type that has 9 or more different sizes that look almost the same. It is necessary to get an exact size match to get reliable operation.

Pod

A small module of electronics or a circuit board, which contains an emulation processor and some accessory circuitry that, connects the emulator to the target through an adapter. This can be a small plastic container with a circuit board inside or an open circuit board with exposed pins that connect to the target system.

See also ***Bondout Pod****,* ***Hooks Mode Pod****, and* ***External Mode Pod****.*

Power Selection

To determine whether to use internal power or external power for the emulation processor on the pod. Usually controlled by a jumper on the pod.

Power supply (short and long tail)

An electrical/electronic circuit that supplies all operating voltage and current to the system. There are two power supply units for powering the emulators. Both of these units are 5-volt, 6-amp units, but with one difference; one has a short tail (the length of the power cord from the converter to the plug) and the other a long tail. The long tail is used for powering the BDM pods only, not the HSP box. The short tail power supply unit may be used for powering the BDM pod or the HSP box.

PPA (Program Performance Analyzer)

A statistical tool used to collect information from a currently executing program. Displays the number of clock cycle functions required for accessing data.

PPAGE

A register in some Motorola 68HCl 2 family members that controls what page of program memory it mapped into the Program Page Window. See the Motorola Users' Guide for the particular family member for further information.

Program counter

A contraction of the more descriptive Program Location Counter. Often abbreviated to PC. The program counter indicators provide line number information supplied by compiler manufacturers. The register in a computer that holds the address of the current instruction. Normally it is incremented by the size of each instruction as the instruction is fetched from memory to be executed. (Jump, Branch and Call instructions can change the PC to a new, out of sequence location.) Interrupts also change the PC.

Program Step

General term for one of four emulator features that allow the user to see the results of program execution in a step by step fashion. They are: Source Step Over, Source Step Into, Assembler Step Over and Assembler Step Into.

Program Window

A 68HC12 MCU chip feature that allows a section of the 16-bit address space to address pages of program space, usually on chip flash memory. The PPAGE register controls the page visible in the window.

PWM

An acronym for Pulse Width Modulator or Pulse Width Modulation. Embedded computers often have a PWM output unit. This unit generates logic outputs with variable widths and selectable rates. These pulses are frequently averaged to give a variable voltage between logic high and logic low.

R

RAM

An acronym for Random Access Memory. All memory accessible at any instant (randomly) by a microprocessor. Used to refer to the read/write memory of an embedded computer system.

Register

Storage area in memory having a specified storage capacity, such as a bit, a byte, or a computer word, and intended for a special purpose. Something that holds a value. A set of high-speed memory within a microprocessor or other electronic device. A collection of electronic circuits that holds a number. Used in reference to memory locations. In Seehau documentation, we refer to many kinds of registers. Among them are Special Function Registers (SFR), Configuration Control Register (Emulator (CCR)), Special Register, Base Register, Basic CPU Register and User-Defined Register.

Reset and Go

A feature of Nohau emulators that applies a reset signal to the target system and then starts it immediately. This simulates the real-world effects of a short power failure. Useful for testing

initialize code, especially on systems that limit the writing of some registers to a small fixed number of cycles after a reset.

Ribbon cables

A flat cable containing up to 1 00 parallel wires for data and control lines. Nohau uses these cables to connect the pod boards to the PC or HSP and the trace board to the emulator board.

ROM

An acronym for Read Only Memory. A type of memory that has values permanently or semipermanently burned in. Often used in imbedded systems for program storage. Older versions might require a special process at the semiconductor facility to program.

*See also **EPROM**, **EEPROM**, and **Flash**.*

Rotational Cable

A cable adapter that allows connection of a pod to a target with a thin cable that can go out over the target system in the direction of any side of the target processor. Very useful for connecting an emulator to a cased target system

S

S19

A load file format. This format is a text file with the addresses and the data to be loaded in hex. This format makes no provision for communicating symbol values to the emulator software.

SAX

A subset of Microsoft's Visual Basic. SAX is Nohau’s version of a micro code language that is somewhat compatible with Visual Basic.

Seehau

A high-level language user interface that allows you to perform many useful tasks including the following: Load, run, single-step and stop programs based on C or Assembly code. Set trace triggers and view trace. Modify and view memory contents including Registers. Set breakpoints. Analyze code with Program Performance Analysis.

Set Breakpoints

A directive to actually place hardware or software breakpoints into the target system.

SFR

An acronym for Special Function Register. For some microprocessor families, this has a precise meaning spelled out in the microprocessor documentation. For other families, it is used very imprecisely, but always to refer to registers in the system being emulated.

See also ***Registers*** *and* ***Special Registers****.*

SFR branch

Some Special Function Registers are BIT addressable, so the base register can be branched to its BIT level definitions.

SFR symbol

Default symbolic reference to the microcontrollers Special Function Registers.

Shadow RAM

A real time mapping of microcontroller data memory. Updated in full speed emulation. Shadow RAM is used to duplicate the contents of the target RAM. Every time the CPU generates a WRITE bus cycle, the pod captures the address/data pair and the emulator board writes that data to the same address in Shadow RAM. The Shadow RAM feature allows you to view memory contents in real-time without stealing cycles from the emulation CPU.

short

A C integer type which may be any size from character to integer.

Single Chip Mode

Code and/or data memory is inside the microcontroller. No address/data bus available externally on microcontroller pins. This mode can be emulated only by bondout pods or hooks mode pods. This mode cannot be emulated with external mode pods.

Software Breakpoint

A breakpoint function implemented by replacing an instruction with another instruction that causes the target system to stop in an orderly fashion (break). The distinguishing feature is that no special purpose hardware is required for placing the breakpoint. This allows a large number of breakpoints. A software breakpoint can not function in ROM.

See also ***Breakpoint*** *and* ***Hardware Breakpoint****.*

Solder Down

A socket that is soldered down to a PCB in place of a microcontroller that allows an emulator adapter to be plugged in to the target circuit. Some solder down sockets allow the microcontroller or the emulator adapter to be plugged. This usually consists of a detachable top half and a solder-down base, so that the pod can be easily removed. In contrast with a socketed connection.

See also ***Solder Down Adapter*** *and* ***Solder Down Base****.*

Solder Down Adapter

The adapter is soldered down directly on the target surface mount footprint, which replaces the socket.

*See also **Solder Down** and **Solder Down Base**.*

Solder Down Base

The lower half of a solder down adapter assembly. Usually included in a solder down adapter assembly but can be ordered separately.

Source

A window in Seehau user interface that displays the source program.

*See also **Source Program**.*

Source Program

An informal name for the program description that the software engineers or programmers write for input to the compiler or assembler. The source of the intermediate files and the final program that can be executed by the computer.

SP

An acronym for Stack Pointer.

*See **Stack Pointer**.*

Special Register

In Motorola families, the memory locations in the system being emulated that have side-effects such as controlling input/output devices and processor configuration; as contrasted with ordinary memory (RAM and ROM) locations that just hold data.

Seehau allows you to add registers (Add Register) to define a new special register, (Add Special Registers) to display a special register defined in a file and File - Load Default CPU Symbols to make the special register definitions available for disassembly and in-line assembly.

SRAM

An acronym for Static Random Access Memory. A type of RAM that will hold its contents without any electronic activity as long as power is applied. See also DRAM (Dynamic Random Access Memory) which requires periodic electronic refresh cycles to keep its contents.

*See also **DRAM**.*

Stack Overflow

A situation where the Stack Pointer exceeds the maximum allowed value or falls short of minimum possible value, and is pointing to some address which, is not a stack. A condition where the size of a stack has exceeded or attempted to exceed the memory allocated to the stack. Usually this

happens when there is too many nested function calls. Not always recognized automatically, and generally prevents continued correct operation of programs using the stack.

Stack Pointer

A register that contains the current location of the program stack.

startup.bas

A macro file that is used by Seehau to input stored values for starting a previously setup project.

Symbolic Data

Included in many file load formats to give the user the ability to debug their code with specifics to symbol type, module, functions, etc. Symbolic data can be used to refer to files that represent computer code with symbols, that is either assembler or compiler source files. It can also refer to compiler, assembler or linker output files that have preserved some of the original symbolic information for debugging.

Symbolic Format

A format using identifiers rather than numbers. Accessing a component by its symbolic name.

System Clock

The main CPU clock. The operating frequency of the microcontroller.

T

Target

A general name for the embedded system being developed with the help of an emulator. A customer application circuit board, the microcontroller on which is to be replaced by a pod during emulation.

Time.bas

A macro file used to run the timer program to test pods.

Time Stamp

A feature that displays the number of machine cycles that have elapsed since the beginning of program execution.

Trace

A comprehensive tool used to analyze the microprocessor environment. An emulator feature that records detailed information about target memory accesses while the target system is in operation. Triggering features allow the trace to be stopped on conditions of interest so that the user can look at the trace information and save it to disk without disturbing the operation of the target.

Trigger

An event that stops trace buffer recording.

Tristated

An output, which has a third state of high impedance in addition to the regular high and low state. When tristated, there is a high impedance seen by the rest of the circuit, there is no current sourcing or sinking.

U

uP clock

The uP Clock is the internal CPU clock of the microprocessor. This setting is used only for the calculation of the trace timestamp.

User-Defined Register

Registers added to the register window by a user using the Registers - Add Special Registers (SFR) menu item. They are saved when the configuration settings are saved. Also used to refer to some of the symbols defined in a load file.

V

Vcc

In electronic designs the supply for transistor collectors originally, now usually the commonly used positive power supply. The power supply in Bipolar Integrated Circuits (IC) usually wired to the transistor collector in the IC. Normally positive 5 volts.

Vector Table

A table of addresses to jump to when certain actions occur. There is usually a start vector where program execution starts, and an error vector (hardware trap) where the controller jumps too if a problem occurs, and many other vectors.

W

Watch window

A window in Seehau user interface that allows display of C and C++ expressions using the symbols of the currently loaded program. The Watch window is not as flexible as the Inspect window, but offers the advantage of compactly displaying the value of many items.

Window

A portion of the screen that can contain its own document or message.

1. A rectangular area of display on you monitor.

2. Section of target memory that is handled especially for HC12 (Program window, Data window and Extra window).

Window Filter Mode

Restricts the triggering logic, but allows recording only of references to program or data areas of interest. More useful information can be collected in this mode before old information is over-written in the trace memory.

Windows NT/2000/95/98

Operating Systems (OS) produced by Microsoft Corporation. Windows 98 replaced Windows 95 and Windows 2000 replaced Windows NT 4. These are the dominant opeerating systems for PCs today.

X

XRAM

Provides access to 2K of on-chip RAM. No external bus cycles are executed for these accesses.

Index

A

C

D

E

F

G

H

I

M

N

O

P

Q

R

S

T

W