For Intel® Target Processors

PowerPack® EA/SW In-Circuit Emulator

Hardware Reference

MICROTEK INTERNATIONAL
Development Systems
Doc. No. l49-001080
Part No. 14867-002
October 1996

Trademark Acknowledgments

PowerPack is a registered trademark and SLD is a trademark of Microtek International.

IBM, LAN, and OS/2 are trademarks of IBM.

Microsoft is a registered trademark and MS, MS-DOS, and Windows are trademarks of Microsoft Corporation.

Intel is a registered trademark and Intel386 and Intel486 are trademarks of Intel Corporation.

PC-NFS is a registered trademark of Sun Microsystems.

MICROTEK INTERNATIONAL

Development Systems Division
3300 N.W. 211th Terrace
Hillsboro, OR 97124-7136
USA
Tel: (503) 645-7333
Fax: (503) 629-8460
Email: info@microtekintl.com
Web: http://www.microtekintl.com

6, Industry East Road 3
Science-based Industry Park
Hsinchu 30077
Taiwan, ROC
Tel: +886 35 772155
Fax: +886 35 772598
Email: easupport@adara1.adara.com.tw

Contents

Product Summary

The term "PowerPack emulator" refers to any PowerPack® in-circuit emulator for embedded system development. The terms "PP", "SW", and "EA" refer to the PowerPack PP, SW, and EA emulators respectively. The terms "SLD™ software", "emulator interface", and "debugger software" refer to the SLD™ source-level debugger.

This chapter describes the emulator and debugger documentation, host system requirements, and how to contact Microtek International for information and technical support.

Documentation

The following describes the printed and online documentation resources for the PowerPack emulators. The manuals in your emulator package are the *SLD™ Source-Level Debugger User's Manual* (referred to as the *User's Manual*) and the *PowerPack® EA/SW In-Circuit Emulator Hardware Reference*, referred to as the *Hardware Reference* and formerly known as the *Up & Running*. Other, related publications described at the end of this list are not included in your emulator package.

Resource	Chapter	Contents
Hardware Reference	Product Summary	Parts, features, documentation, support
	Software Installation	Configuring your PC or workstation; installing the SLD software
	Hardware Installation	Installing the PowerPack hardware; running the confidence tests
	Tutorial	Practicing basic emulator tasks
	Target Hardware	SAST board schematics; signals
User's Manual	Getting Started	Host sytem requirements; contacting Microtek
How to...	Defining the Debug Environment	Creating a loadfile; starting and exiting the SLD software; configuring memory and registers; using an initialization file
	Debugging in Source	Viewing source code, disassembly, and stack; editing variables; controlling emulation
	Debugging in Registers and Memory	Accessing CPU and peripheral signals and numeric or disassembled memory contents

Debugging with Triggers and Trace	Emulation and trace control using triggers; numeric and symbolic address formats

Reference

powerpak.ini File	powerpak.ini file contents
Toolbar	Toolbar controls
Shell Window	Shell window contents, controls, commands
Source Window	Source window contents, controls
Variable Window	Variable window contents, controls
Breakpoint Window	Breakpoint window contents, controls
CPU Window	CPU window contents, controls
Stack Window	Stack window contents, controls
Memory Window	Memory window contents, controls
Peripheral Window	Peripheral window contents, controls
Trace Window	Trace window contents, controls
Event Window	Event window contents, controls
Trigger Window	Trigger window contents, controls

PowerPack
SLD Help

For help on using online help, choose How to Use Help from any SLD Help menu or press <F1> twice.

Whether or not the emulator is active, you can invoke the SLD online help from within Windows. Choose the SLD Help icon (shown at left). SLD online help conforms to the standard Windows help interface, as described in your Microsoft Windows documentation.

For help from within the SLD software, choose a Help menu item; or, press <F1> at any time. In most SLD dialog and message boxes, you can choose a Help button for context-sensitive help. In the Shell window, you can list Shell command syntax with a Help command.

Related Publications

Topic	Resource
Windows 3.1; Windows 95; Windows for Workgroups 3.11	Microsoft documentation
Your target processor	Your chip vendor documentation
Your toolchain and loadfile format	Your compiler, assembler, linker, and converter documentation
C++ symbols	The Annotated C++ Reference Manual, Margaret Ellis and Bjarne Stroustrup (Addison-Wesley, 1990)

How to Contact Microtek

To register for technical support and ongoing product information, complete and mail the registration card enclosed with the emulator.

Contact Microtek/DSD to purchase Gold Club membership. Gold Club provides firmware, software, and hardware updates and priority service, in addition to repairs.

As a Microtek customer, you can contact Microtek technical support for help with an emulator problem during your warranty period. The email and fax lines are operational 24 hours a day, 7 days a week.

Internet email	csupport@microtekintl.com (technical support) info@microtekintl.com (other information)
World Wide Web	http:\\www.microtekintl.com (product news)
Microtek/DSD, Western USA	(503) 645-7333 voice; (503) 629-8460 fax (voice contact available Monday through Friday, 8:00 am to 5:00 pm USA Pacific Time)
Microtek, Eastern USA	(610) 783-6366 voice; (610) 783-6360 fax (voice contact available Monday through Friday, 8:00 am to 5:00 pm USA Eastern Time)
Microtek, Hsinchu, Taiwan	+886-3-577-2155 voice; +886-3-577-2598 fax (voice contact available Monday through Friday, 8:00 am to 5:00 pm Taiwan Time)
Adara International, Taipei, Taiwan	+886-2-501-6699 voice; +886-2-505-0137 fax. (voice contact available Monday through Friday, 8:00 am to 5:00 pm Taiwan Time)

Before you call, please read the PowerPack® Emulator Problem Report Form in the SLD online help.

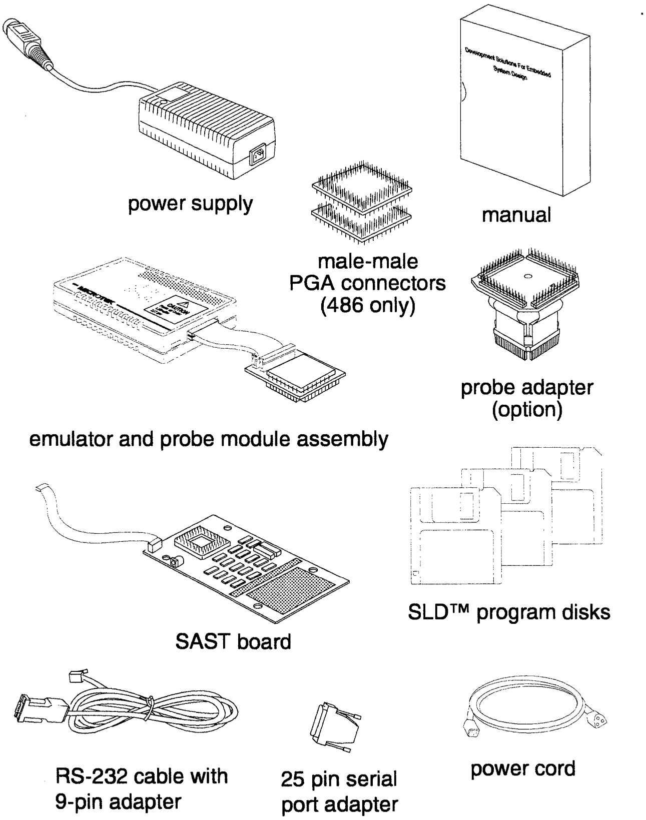
When you call, please be at your computer with the SLD software running and have the emulator documentation and filled-out problem report form (printable from the online help) nearby.

Emulator Parts

When you take the emulator out of its shipping package, check to be sure all the following are present (see the figure following this list).

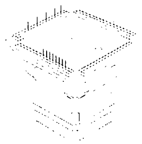
- the emulator and probe assembly
- optionally, an adapter for connecting the probe to the target board
- with the EA-486 emulator only, two male-to-male PGA connectors.

- a stand-alone self-test (SAST) board
- an RS-232C cable for connecting the emulator to the host system
- a 25-pin and a 9-pin serial adapter
- a power supply
- a power cord
- three SLD software program disks
- besides this manual, the *SLD User's Manual* and a slipcase binder



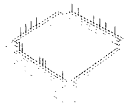
Target Adapters

When your target hardware is ready, you need an adapter to connect the emulator to the processor chip or socket your target board. You can order the adapter from Microtek with your emulator order or separately. The following target adapters are available:



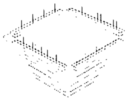
386 Clip-over
(PP-ET-132QFPCO)

Clips over and tri-states a surface-mounted 386 target CPU.



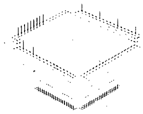
Amp
(PP-ET-132SM-AMP)

Plugs into an Amp 821949-5 socket on the 386 target board.



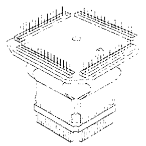
Textool
(PP-ET-132SM-3M)

Plugs into a 3M/Textool 2-0132-07244-000-018-007 socket on the 386 target board.



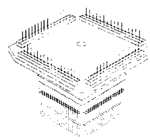
386 solder down
(PP-ET-132QFPSD)

Solders to the target board in place of the 386 target CPU.



486 Clip-overs
(168PGA-208CO or 168PGA-196CO)

Clips over and tri-states a surface-mounted 486 target CPU.



486 solder downs
(168PGA-208SD or 168PGA-196SD)

Solders to the target board in place of the 486 target CPU.

Emulator Features

Communication between the emulator and the host system is via RS-232C communications at 19.2K, 38.4K, 57.6K, or 115K baud.

The emulator automatically configures itself for 5V or 3V operation.

Source-Level Symbolic Debugging

The SLD™ (Source-Level Debugger) software runs as a Microsoft Windows 95, Windows 3.1, or Windows for Workgroups 3.11 application with context-sensitive online help. Besides using a mouse or Windows-style keyboard entry with menus and buttons, you can enter commands via the SLD Shell window command line.

You can open several SLD windows at once. For example, you can monitor variables and view the trace while debugging at the source level. You can view two sections of source code simultaneously in the Source window. You can have up to 20 different Memory windows open simultaneously with various numeric, ASCII, and disassembly views of memory.

You can debug from the vantage of your C and assembly language source:

- All symbol types are supported, including static variables, stack-based local variables, register-based variables, structures, arrays, and pointers.
- You can selectively load object code and symbolic information into target or overlay memory and into the symbol table, for OMF86 and OMF386 load formats .
- Source display formats include source and assembly language from your source files, disassembly from memory when the source files are unavailable, and disassembly from memory interleaved with the corresponding lines from your source files.

You can set breakpoints on a line of source or disassembly or on a symbolic or numeric address:

- 128 software breakpoints are available for the SW; 256 are available for the EA.
- Up to four hardware breakpoints are available, using the debug registers (DR[0..3]). You can reserve the debug registers for use by your program instead of as breakpoint registers.
- The emulator automatically chooses whether a breakpoint is set in hardware or in software; or you can access the debug registers to explicitly specify a hardware data or execution breakpoint.

Memory and Register Access

You can substitute 1M or 4M bytes of emulator-controlled overlay memory for your target RAM or ROM memory. You can configure the overlay memory with zero or more wait states.

You can monitor the stack, the CPU registers, the peripheral registers, and memory contents during emulation.

A single-line assembler is available for patching loaded code.

Real-Time Trace

Real-time, full-speed tracing is optionally available:

- You can collect 128K frames of address, data, and signal trace in the SW, or 256K frames in the EA.
- You can qualify trace collection by address, data, and signal criteria in the EA.
- You can display trace as instructions, as bus cycles, or (in the EA) as clock cycles.
- You can link the Trace and Source windows to scroll together, to view the disassembled trace synchronously with the corresponding source lines and disassembled memory.
- During emulation, you can start and stop trace collection without affecting emulation.
- In the EA, besides manually starting and stopping trace and emulation, you can define up to four sequential trigger conditions to conditionally control emulation and trace collection. Each trigger is a logical combination of up to eight events, with optional counter and timer dependencies. An event is defined as signal values and inclusive, exclusive, or masked address and data ranges.

Software Configuration

The terms "SLD software" and "emulator software" refer to the SLD™ source-level debugger for the PowerPack® emulator.

The SLD software runs under Windows 3.1, Windows for Workgroups 3.11, and Windows 95.

Host System Requirements and Recommendations

- An Intel486 or Pentium based PC or 100% compatible system
- Windows 95; or MS-DOS 5.0 or 6.x with Windows 3.1 or Windows for Workgroups 3.11 running in 386-enhanced mode
- At least 8M bytes of RAM
- At least 8M bytes of free memory after you have loaded your Windows interface and any other applications besides the SLD software.
- At least 5M bytes of available disk space
- A VGA or Super VGA graphics card and color monitor (a graphics accelerator card recommended to boost performance; a monitor capable of at least 800x600 operation recommended)
- A mouse
- A serial port for connection to the emulator (16550 UART recommended for operation at 57.6K baud and above)
- At least 4M bytes for a swap file (permanent swap file recommended, with a disk cache such as smartdrive for improved Windows performance)
- Config.sys entries of at least Files=30 and Buffers=30

SLD™ Software Installation

To install the SLD software on your host system:

1. Run setup.exe from Program Disk 1.
2. Follow the instructions presented in the window. The files and subdirectories are installed in a default directory named \powerpak unless you specify otherwise when prompted. (Examples throughout this manual use the powerpak directory name.)

3. At the end of the installation, you can view the readme.txt file for the latest release notes.
4. Exit Windows to install the firmware, as noted in the last screen of the installation. In Windows 95, and on some systems running Windows or Windows for Workgroups, you can install the firmware from a DOS window or from the Program Manager.
5. Insert Program Disk 3 into a disk drive. Either make Program Disk 3 the current drive or copy all files from Program Disk 3 to the current drive and directory.
6. Enter install at the DOS prompt; or, in the Program Manager File Run dialog box, run install.bat from Program Disk 3.
7. Follow the instructions presented on the DOS screen.

The installation creates a powerpak.ini file in your Windows directory. Any previously existing powerpak.ini is renamed powerpak.bak.

To uninstall the SLD software:

1. Delete the powerpak directory and its contents.
2. Delete powerpak.ini from your Windows directory.
3. Delete the emulator icons and group.

Toolchains

Because of OMF86 and OMF386 loadfile format standards, the output formats of most x86 development toolchains differ little.

When using the Metaware HC toolchain, compile with the switch Optimize_for_Space (-Os) OFF and the switch Align_Routines ON. This combination aligns the line number information for function entry points on the actual function execution addresses. This alignment is necessary for the SLD software to set source-line breakpoints on the start addresses of the function entries and to display local symbols.

When using the Borland C compiler, before loading your OMF386 loadfile, set the emulator's maximum bitfield size to 16 bits. On the SLD Shell command line enter:

```
maxBitFieldSize 16
```

When using PharLap LinkLoc 7.1, use its -regvars switch to include symbolic information for register variables. The emulator supports register variable extensions to the x86 symbol table.

Hardware Configuration

This chapter explains how to:

- *Connect the emulator to the host system and connect the parts of the emulator together.*
- *Connect the emulator to the stand-alone self-test (SAST) board.*
- *Power-up and power-down the emulator; run the hardware confidence tests*
- *Connect the emulator to a user target.*

Ensuring a Proper Physical Environment

The emulator or debugger requires the same physical environment as your host system:

- Avoid excessive heat and humidity. Microtek recommends an ambient temperature within 0 - 40° C (32 - 104° F) and an ambient humidity range within 85% maximum relative humidity, noncondensing.
- Leave a few inches around the main chassis for air circulation.
- Use good grounding practices against electrostatic discharge.
- Keep the emulator away from electromagnetic interference.

ESD CAUTION

The circuitry of the emulator probe can be damaged by excessive electrostatic discharge (ESD). Protect your probe from ESD:

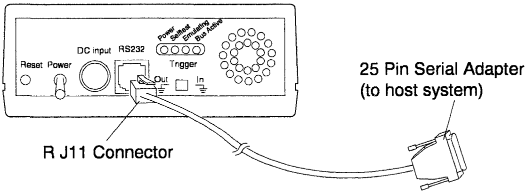
- *Ensure the emulator, host system, and workbench are properly grounded before applying power.*
- *Work in a static-free environment.*
- *Use a wrist-strap attached to ground while handling the probe.*
- *Avoid touching the exposed connector on the probe when you are improperly grounded.*

Connecting the Emulator to Your Host System

The supplied RS-232C cable resembles a telephone cable because both use RJ11 connectors. However, you cannot use a telephone cable in place of the supplied RS-232C cable. If you attach the emulator to your host system with a telephone cable, the emulator will not work.

Depending on your host system serial port, you may need to substitute the 9-pin serial adapter for the 25-pin serial adapter.

RS-232C cable plugging into the back of the emulator



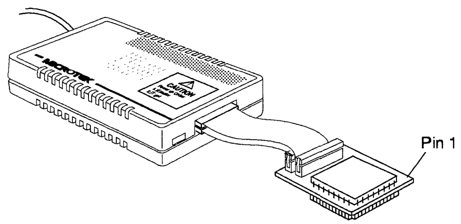
Connect the emulator to your host system using the RS-232C cable:

1. Firmly seat the RJ11 connector in the emulator's RS-232C port.
2. Firmly seat the 25-pin or 9-pin serial adapter in your host system's COM1, COM2, COM3, or COM4 serial port.

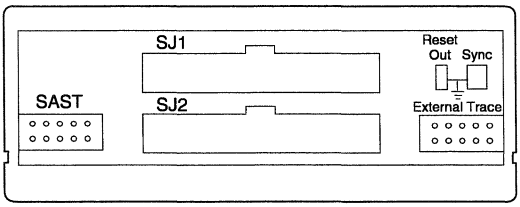
Connecting the Probe Cables to the Emulator

The emulator and probe are shipped as an assembled unit, connected by a pair of in-circuit emulation (ICE) cables, as shown in the following. If the ICE cables are disconnected, reconnect them.

Emulator and probe assembly



SJ1 and SJ2 plugs where the ICE cables connect to the front of the emulator



The orientation of the cable jacks in the plugs is significant. The jacks slide easily into the plugs when oriented correctly. To firmly seat a jack on a plug, hook the metal side-clips on the jack into the side-flanges on the plug.

One of the ICE cables is longer than the other. When connecting the cables, ensure the longer cable extends from the SJ1 plug on the emulator to the plug closest to the processor on the probe.

To connect the emulator and probe with the ICE cables:

1. Ensure the emulator's power switch is off.
2. Seat the short cable firmly in the bottom (SJ2) emulator plug.
3. Seat the bottom (SJ2) cable firmly in the probe plug closest to the short side of the probe, with the cable extending away from the probe processor.
4. Seat the long cable firmly in the top (SJ1) emulator plug.
5. Seat the top (SJ1) cable firmly in the probe plug closest to the probe processor, with the cable extending away from the probe processor.

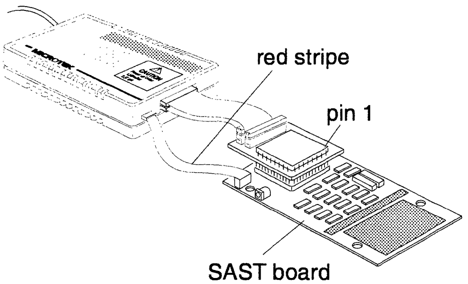
Connecting the SAST Board

The stand-alone self-test (SAST) board is used for:

- The confidence tests
- The tutorial
- Your target software when your target hardware is unavailable

The probe and SAST board are shipped from the factory as an assembled unit.

SAST board and probe assembly



Power CAUTION

With the emulator connected to your target board, apply and remove power to the emulator and target board in the correct sequence to avoid severely damaging both units:

1. *Apply power to the emulator.*
2. *Apply power to the target system.*
3. *Remove power from the target system.*
4. *Remove power from the emulator.*

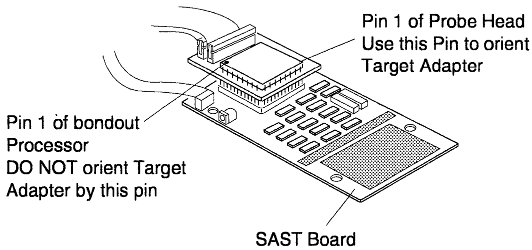
To connect the emulator to the SAST board:

1. Ensure the emulator's power switch is off (toggled down).
2. Plug the probe into the SAST board, matching pin 1 on the probe with pin 1 on the SAST board.

3. Connect the SAST power cable to the emulator and SAST board. The SAST power cable is a gray ribbon cable with one red edge and a ten-pin female jack on each end. When oriented correctly, the cable is not twisted. Push on the jacks to seat them firmly on the plugs. Powering-up the emulator powers-up the SAST board.

386EX CAUTION

To avoid permanently damaging the emulator and 386EX SAST board, be careful to connect pin 1 on the probe to pin 1 on the 386EX SAST board. Pin 1 of the probe is labeled as such. Pin 1 of the bondout (the 386EX processor on the probe) is 180 degrees opposite pin 1 of the probe and is marked by a white dot and a notched corner. The following shows the pin 1s.



Setting the 386 SAST Board Jumpers

When adding memory or circuitry to the SAST board, be sure to reconfigure any jumpers as needed. The jumpers control power, target (not overlay) memory, and reset options:

5V, 3.3V	selects 5 volt or 3.3 volt operation.
JACK, POD	selects whether power is obtained from the emulator via the ribbon cable (POD) or from a separate power supply plugged into the SAST board jack (JACK).
RESET	resets the SAST board when shorted. You can use this jumper to add a target reset input.
memory	is controlled by a pair of jumpers. The SAST board is shipped from the factory with 8K-byte memory chips and both jumpers set to 8K. When adding memory to the SAST board, adjust the jumpers accordingly.
X1, X2	on the CX/SX SAST board must be set on X1.

Setting the 486 Probe Jumpers

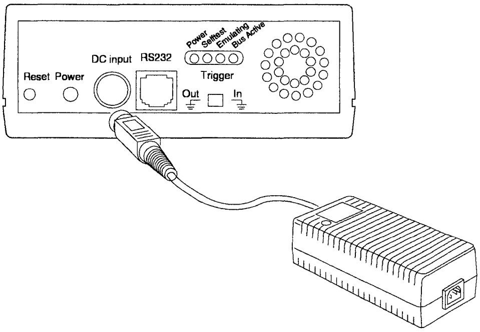
The FPU (floating-point unit) and SLE (SL-enhanced) jumpers on your 486 probe must correspond to the probe processor for correct emulator start-up and the 32B jumper must be installed. The SLE controls SRESET, SMI#, SMIACT#, and STPCLK#. The following lists the supported 486 processors and the corresponding jumper settings:

Processor	FPU	SLE
DX, DX2	ON	ON
SX, SX2	OFF	ON
DX, DX2 non-SLE	ON	OFF
SX, SX2 non-SLE	OFF	OFF

Applying Power to the Emulator

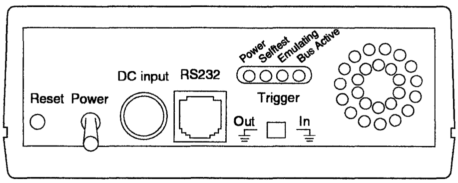
For emulator power, use the provided DC power supply. The emulator powers the probe and SAST board via the provided cables.

Power supply plugging into the emulator



Once the DC power supply is plugged into the emulator and into an AC power source, the emulator power is controlled by the power switch on the back of the emulator.

Emulator back panel with power switch toggled off (down)



To connect the emulator to the power supply:

1. Ensure the emulator power switch is off.
2. Plug the power supply into the emulator DC input jack.

3. Connect a power cord between the DC power supply and an AC power source.

To power-up the emulator:

1. Check the seating of the cables between the host system, emulator, and probe, and between the AC power source, DC power supply, and emulator. Check the seating of the probe in the SAST board and the power cable from the emulator to the SAST board.
2. Turn on the power switch on the back of the emulator chassis. A power-on reset occurs and the system processor tests and initializes the system. The Power and Selftest LEDs on the emulator glow on reset. When the emulator and target have powered-up successfully, the Selftest LED dims and the Bus Active LED glows.

Power CAUTION

Turn on the emulator before turning on the target system. Power must be applied and removed in the correct sequence. Failure to follow this sequence will severely damage your target system and the emulator:

1. *Apply power to the emulator.*
2. *Apply power to the target system.*
3. *Remove power from the target system.*
4. *Remove power from the emulator.*

Reading the Status LEDs

The emulator power-on and processing status is reported by LEDs on the back panel of the emulator.

LED	Meaning
Power	Green: The emulator is powered-up.
Selftest	Red: either the system processor is executing the power-on self tests or a failure has occurred.
Emulating	Green: the probe processor is emulating.
Bus Active	Green: The emulator has established communication with the target.

Press the Reset button on the back of the emulator. When the Selftest LED dims and the Bus Active LED glows, start the SLD software.

Running the Hardware Confidence Tests

The confidence tests report whether the emulator hardware is functioning properly. For these tests, the emulator must be connected to the SAST board. On the EA-486 probe, the FPU, SLE, and 32B jumpers must be installed.

To run the tests, enter a test command in the Shell window Command Entry pane. The test command syntax is:

Test [Loop] [Repeat | Continue] [Brief | Verbose] [Except] [<name> | <number>]

With no arguments, test runs all the confidence tests, displaying the final result of each test.

Argument	Action
Loop	Repeat the low-level operations in the specified test. As the operations repeat, you can observe them on an oscilloscope. To stop, press <Esc>.
Repeat	Repeat the specified tests. To stop, press <Esc>.
Continue	Continue through all tests, even if one fails.
Brief	Display only the test sequence. The default is to show the final result of each test in sequence.
Verbose	Display progress reports during each test, in addition to the default information (the final result of each test).
Except	Exclude the specified tests and run all others.
<name>	Identify one or more tests by name, as listed below.
<num>	Identify one or more tests by number, as listed below.

The following lists the test numbers and names and describes what each test demonstrates and some actions you can take when some tests fail. Not all tests apply to all system configurations. If a test fails despite any recommended actions, contact Microtek.

Number **Name**

3 traceMemory

Tests the emulator trace memory.

7 xilinxProgram

Tests the probe firmware programming. If this test fails, try reseating the ICE cables between the emulator chassis and probe.

8 sastConnect

Tests the control connections between the emulator and SAST board.

If this test fails, try reseating the probe and power connections on the SAST board.

9 targetConnect

Tests the connections between the probe and SAST board. If this test fails, try reseating the probe and power connections on the SAST board.

10 procConnect

Tests the connections between the probe processor and both the chassis and the SAST board. If this test fails, try reseating the probe in the SAST board; try reseating the ICE cables between the emulator chassis and probe.

11 iceMemory

Tests the ICE memory.

12 procBasic

Tests the probe processor functionality.

13 procInternal

Runs the built-in self-test of the probe processor. If this test fails, try reseating the probe on the SAST board.

14 mapMemory

Tests the overlay memory mapping logic.

15 overlayMemory

Tests the overlay memory. If this test fails, contact Microtek.

16 procGrHalt

Tests the ResetAndGo and Halt operations. If this test fails, try reseating the probe on the SAST board.

17 procGoHalt

Tests the Go...Halt operations. If this test fails, try reseating the probe on the SAST board.

18 procReset

Tests processor RESET behavior.

19 procStep

Tests single-stepping.

20 procSwBkpt

Tests software breakpoints.

21 procHwBkpt

Tests hardware beakpoints.

22 procTrigger

Tests the trigger breakpoint logic.

24 sastMemory

Tests the SAST memory.

25 mapBkpt

Tests breaking on memory access violations.

26 mapRom

Tests intercepting overlay memory write violations.

27 signalGating

Tests enabling and disabling signals from the target. The configurable signals for your processor are listed in the "Target Hardware" section of this manual.

29 catTraceData

Tests the trace system

30 targetConnectToggle

Tests the target connection.

31 targetConnectAndIn

Tests the target connection.

Confidence test examples

>test	/* Run all tests and display all results. */
>test brief	/* Run all tests and display only the final result. */
>test verbose	/* Run all tests; display all */ /* results and progress reports. */
>test 9 3 4 5	/* Run tests 9, 3, 4, and 5 */
>test verbose OverlayRam	/* Run OverlayRam test. */
>test loop 13	/* Run scope loop 13 until <Esc> is pressed. */ /* The processor is reset as often as possible. */ /* No results are displayed. You can watch the */ /* resets with an oscilloscope as they occur. */
>test repeat 13	/* Run scope loop 13 until <Esc> is */ /* pressed; display results after each iteration. */
>test continue	/* Run all tests, continuing /* /* even if one or more tests fail. */

Connecting to a Target System

Power CAUTION

Turn off both the target and the emulator before connecting the emulator probe to a target. Leaving either part powered-on can damage the target and emulator. When the probe is attached to a target, turn on the emulator before turning on the target. Failure to follow this sequence will severely damage both target and emulator:

1. *Apply power to the emulator.*
2. *Apply power to the target system.*

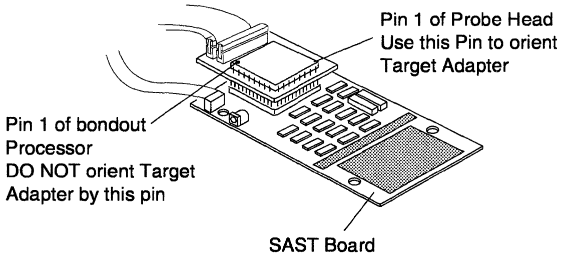
After removing the SAST board from the emulator, connect the probe to your target using an appropriate adapter. The available adapters are described in the "Getting Started" chapter of this manual.

To connect the probe, adapter, and target:

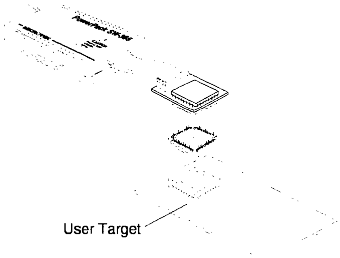
1. Ensure the power is off to the emulator and target.
2. Disconnect the target power cable.
3. Remove the target processor chip, unless you are using a clip-over adapter.
4. Plug the adapter into your target or clip the adapter over the target processor, matching the pin 1s.
5. Plug the probe head into the adapter, matching the pin 1s.
6. Apply power first to the emulator, then to the target system.

386EX CAUTION

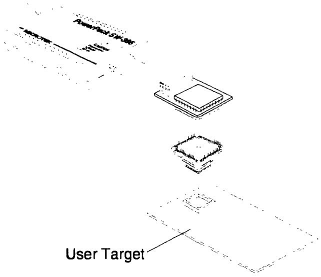
To avoid permanently damaging the target and emulator, be careful to connect pin 1 on the 386EX probe to pin 1 on the adapter and target. Pin 1 is labeled on the probe. Pin 1 of the 386EX processor in the probe is 180 degrees opposite pin 1 of the probe and is marked by a white dot and a notched corner. The following shows the pin 1 orientations, using the SAST board as an example target board.



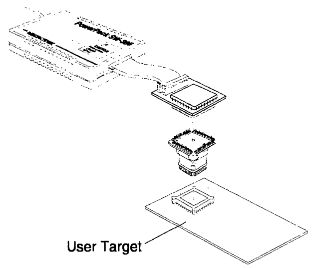
Amp adapter for 386 emulators



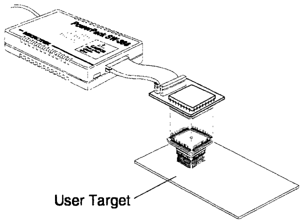
3M/Textool adapter for 386 emulators



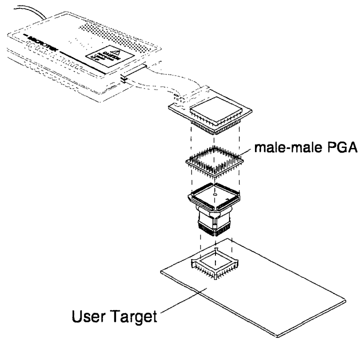
Clip-over adapter for 386 emulators



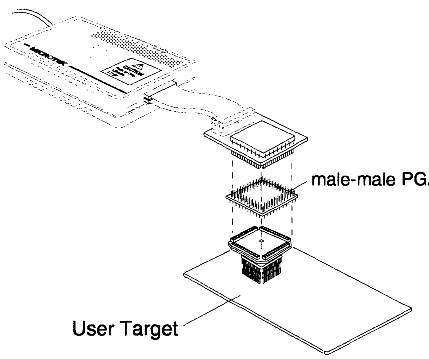
Solder-down adapter for 386 emulators



Clip-over adapter for 486 emulators



Solder-down adapter for 486 emulators



To disconnect the probe from the target processor or socket, ensure the power is off to both the emulator and the target; then, gently pull the adapter out of the socket or off of the processor.

Tutorial

This tutorial provides a guided tour of some of the most commonly used debugging commands, with the opportunity to practice the following:

- *Invoke the SLD software, configure memory, and load a sample program.*
- *Navigate the source, memory, stack, and trace displays.*
- *Start and stop emulation in various ways.*
- *Collect and examine trace information.*

The illustrations in this chapter show this tutorial running on various PowerPack PP, EA, and SW emulators. You can use this tutorial with any PowerPack x86 emulator with overlay memory. Some displays differ between emulators and between processors.

This tutorial assumes you have learned to use the Microsoft Windows environment. If you are unfamiliar with Windows, study your Windows tutorial before starting the SLD software.

Before starting this tutorial, connect the emulator to the SAST board.

The \powerpak\samp386\demo.omf sample loadfile, used by this tutorial, was compiled and linked using the Microtek Research Inc. (MRI) PC toolchain. The source and command files used to generate demo.omf are also provided. Examine these files for information helpful in recreating the loadfile with your own toolchain.

Configure the SLD Software for the Tutorial

Read the "Defining the Debug Environment" chapter in the *User's Manual*, following the instructions for:

- starting an emulator session, noting which (if any) buttons on your Toolbar are grayed-out to indicate unavailable operations
- selecting the COM port and baud rate for communication between your emulator and host system
- co-ordinating Intel386 emulator and target CPUs (if you are using a 386 emulator)
- mapping memory, accepting the default values in the Add dialog box
- loading code and symbols from \powerpak\samp386\demo.omf.

Arrange Your Desktop

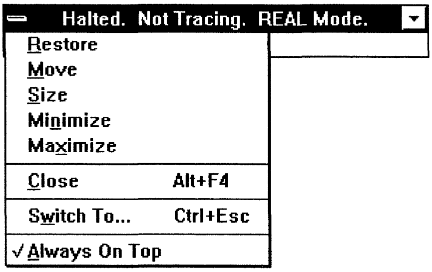
Simplify your desktop by minimizing or closing the Shell window. The Shell window icon is shown here in the left margin.

The first time you start the SLD software, the Status window also appears. This completes the initial default layout: Toolbar, Shell window, and Status window open and all other windows closed. The following shows the Status window before you have done any emulation or modified any memory or registers. The emulation processor is initially in real mode.

Minimize the Status window to simplify your desktop. The Status window icon is shown here in the left margin. Note the status message in the label below the icon.

According to the initial default layout, the Status window (whether open or iconized) remains visible regardless of any other SLD window position. To change this positioning, open the Status window Control menu and disable (toggle-off the checkmark) Always on Top.

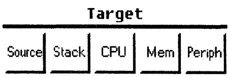
Status window with open Control menu



Examine the Loaded Code and Symbols

In this part of the tutorial, you will practice displaying different parts of the loaded code and symbolic information in the Source, Variable, and Memory windows. You will be changing only the window display and cursor position, without doing any emulation or changing the CS:EIP.

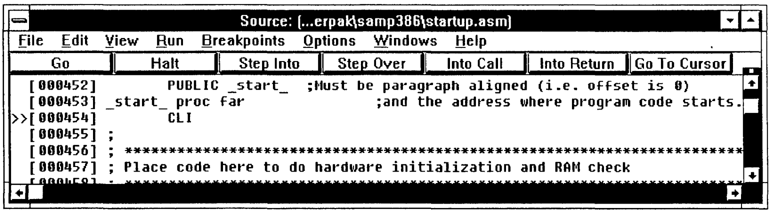
Display Source



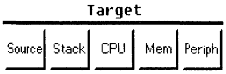
From the Target section of the Toolbar, choose the Source button. You are viewing the startup module in the Source window. The associated source filename, startup.asm, appears in the title bar. The current

CS:EIP (program counter) is marked by >>.

Source window showing the startup module

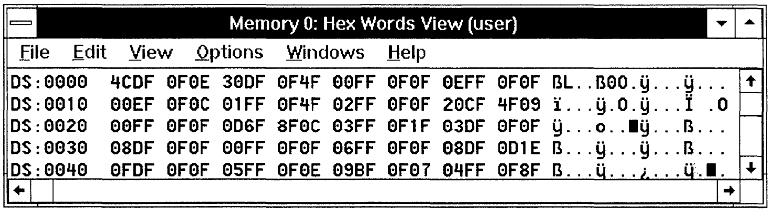


Display Memory



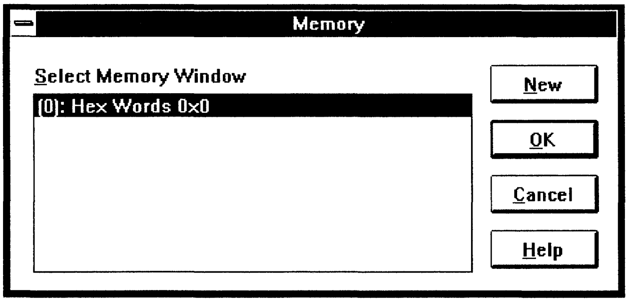
From the Target section of the Toolbar, choose the Mem button to display a Memory window. You are viewing the beginning of the data segment as hexadecimal word values.

Memory window showing the data segment



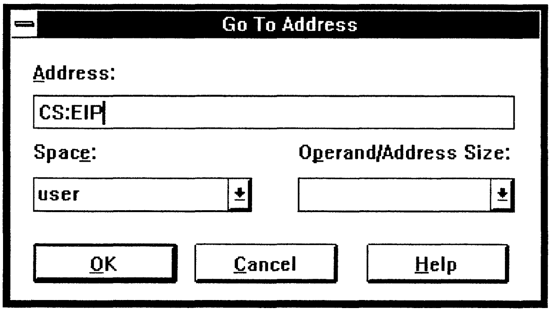
You can have up to 20 Memory windows open simultaneously. The first Memory window is labelled, in its title bar, Memory 0. Choose the Toolbar Mem button again to display the Memory dialog box for selecting or opening a Memory window. The following shows the Memory dialog box, with Memory window 0 available.

Memory dialog box for selecting an open or new Memory window

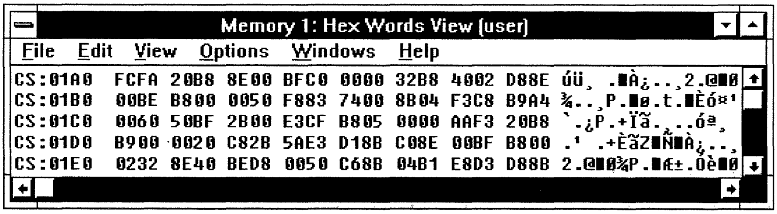


Choose the New button in the Memory dialog box. Another Memory window appears, labelled Memory 1. In this Memory window, open the Edit menu, choose Go To Address, and enter CS:EIP in the dialog box. The memory display changes to the program counter.

Go To Address dialog box for changing the Memory window display

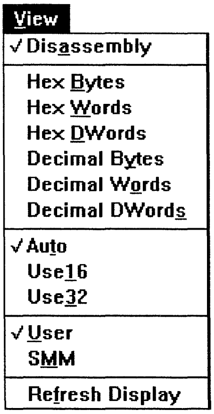


Memory window showing the code loaded at CS:EIP

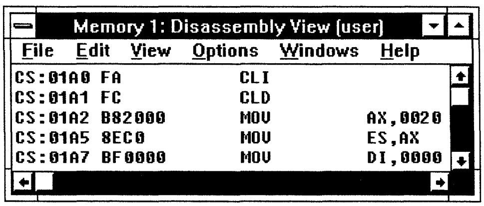


You can use multiple Memory windows to display different sections of memory or the same memory in different formats. Initially, both Memory window 0 and Memory window 1 display hexadecimal words (noted in their title bars). In Memory window 1, open the View menu and choose Disassembly.

Memory window View menu specifying disassembly

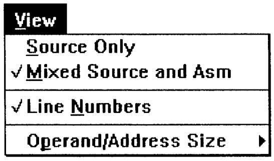


Memory window 1 showing memory contents as disassembly

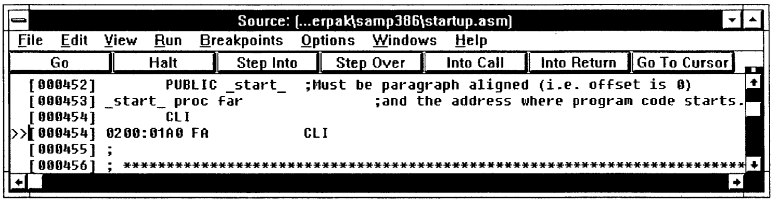


You can see the disassembly in the Source window corresponding to the disassembly in the Memory window. In the Source window, open the View menu and choose Mixed Source And Asm. The following shows the Source window View menu and the consequent display.

Source window View menu specifying mixed source and disassembly display

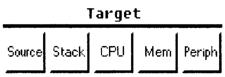


Source window showing disassembly interleaved with source



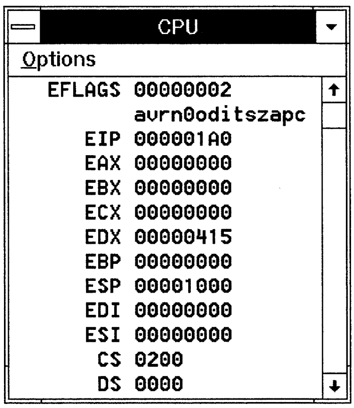
Examine the line of disassembly at the program counter, indicated by >> in the left margin. Compare the address and instruction to the Memory window 1 line showing address CS:01A0.

Display Registers



From the Target section of the Toolbar, choose the CPU button to display the CPU registers. The program counter consists of the CS and EIP registers. CS contains 0200 and EIP contains 01A0, also shown by the line with the >> program counter marker in the Source window in mixed view.

CPU window (EA-486 emulator)

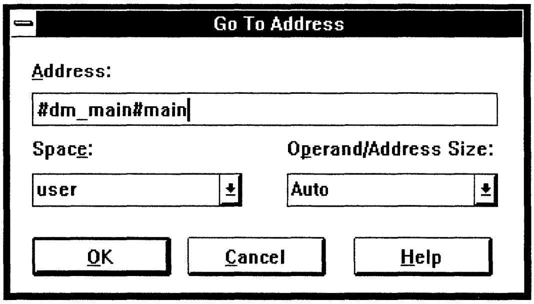


Find Modules and Functions by Symbolic Address

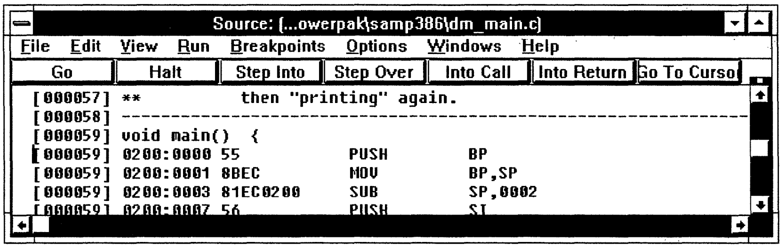
The section of code displayed when you first open the Source window is at the CS:EIP. Since no emulation has been done yet, the display is in the startup module. The title bar at the top of the Source window shows the source file for the displayed module.

Change the Source window display to the main function in the dm_main module, using the module and function symbols. In the Source window, open the Edit menu; choose Go To Address. Enter the fully-qualified symbol #dm_main#main in the Go To Address dialog box.

Go To Address dialog box to display the main function in the Source window



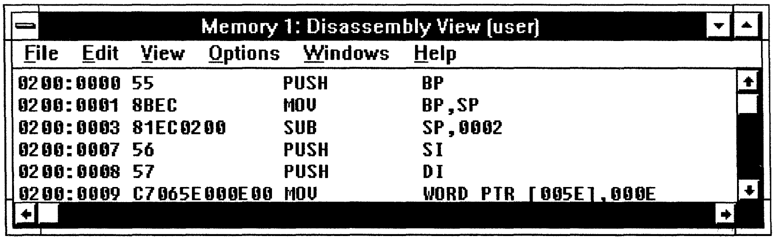
Source window showing main function in mixed source and assembly view



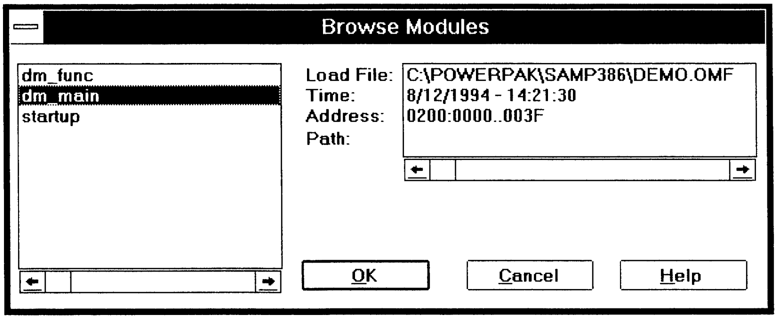
The emulator supports physical, linear, virtual, and symbolic addresses, interpreting numeric addresses as virtual unless you specify the L (linear) or P (physical) suffix.

In the Memory window, open the Edit menu; choose Go To Address. Enter #dm_main#main in the Go To Address dialog box, the same as you did for the Source window.

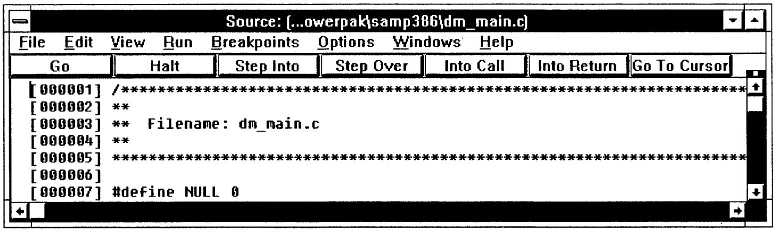
Memory window showing disassembly from the #dm_main#main address



Display the dm_func module in the Source window. Open the File menu and choose Browse Modules. Select dm_func in the Browse Modules dialog box. Choose OK.

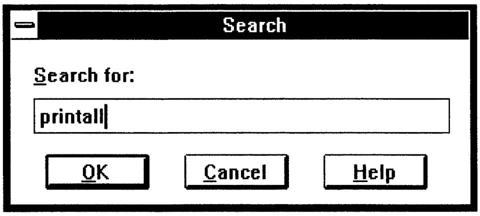


Browse Modules dialog box for displaying the dm_main module in the Source window

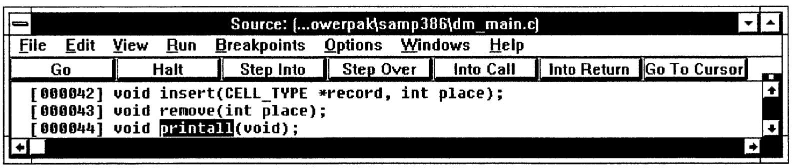


Source window displaying the beginning of the dm_main module in mixed source and assembly view.

Find a reference to the printall function. Open the Edit menu and choose Search. Enter printall in the Search dialog box. The SLD software finds the first occurrance of printall.



Search dialog box for finding the first occurrance of the printall string in the Source window

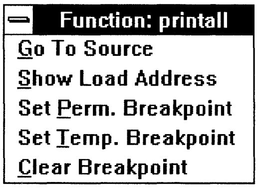


Source window after a successful search for printall

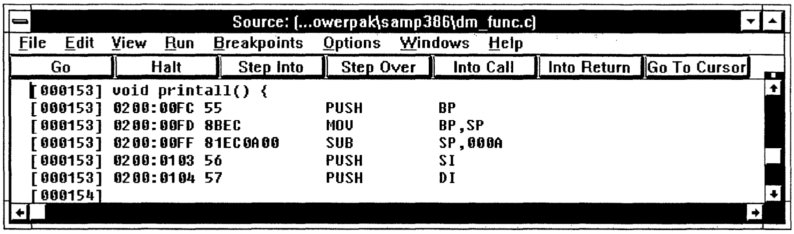
Find a Function From Any Reference

You can move the Source window cursor to the entry point of a function from any occurrance of the function name.

The last search positioned the cursor at the first occurrance of the printall function name in the dm_func module. Double-click on the function name. The Function pop-up menu appears Choose Go To Source. The Source window displays the first line of the function.



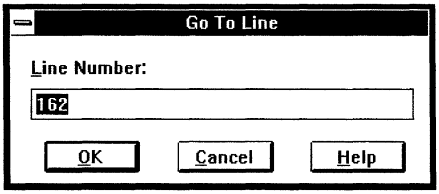
Function pop-up menu accessed by double-clicking on a printall string in the Source window



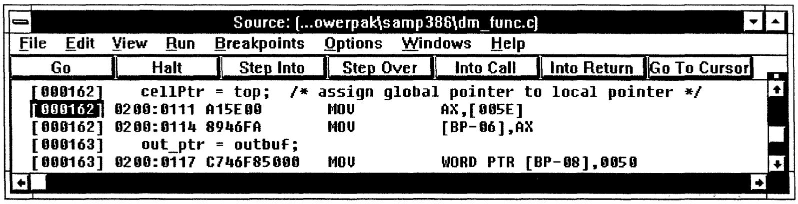
Source window display after choosing Go To Source in the Function: printall pop-up menu

View a Specific Line

You can find any line in the currently displayed module by line number. Open the Edit menu and choose Go To Line. Enter 162 in the Go To Line dialog box.



Go To Line dialog box for finding line 162 in the Source window

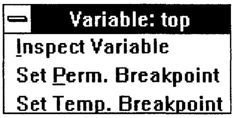


Source window displaying line 162

Inspect a Local Variable

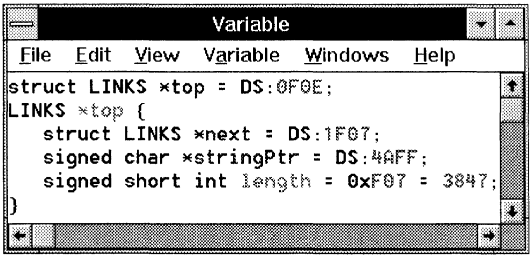
You can examine the definition and value of any variable displayed in the Source window. On line 162, double-click on top. In the Variable pop-up menu, choose Inspect Variable.

Variable pop-up menu accessed by double-clicking on a top string in the Source window



In the Variable window, you can change the values of variables (red) and dereference pointers (blue). Double-click on top, displayed in blue. top points to a structure of two pointers, next and stringPtr, and a short integer, length. Variables out of scope have unknown values.

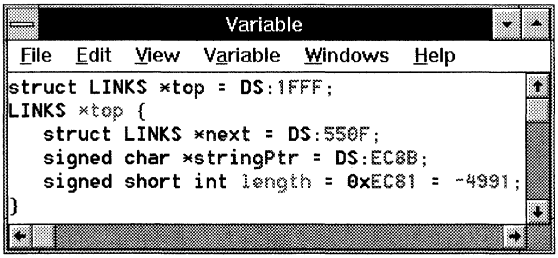
Variable window displayed by choosing Inspect Variable in the Variable: top pop-up menu, with the top pointer dereferenced



Remove the dereferenced structure from the display. Click on the LINKS *top { line; open the Variable menu and choose Delete.

Make top point elsewhere. Double-click on the red 0F0E value of top. In the edit field box appears, enter 1FFF. Since top points to a different location, the values shown for next, stringPtr, and length change.

Result of changing the address in top



Double-click on the 1FFF to open the edit field; enter 0F0E to reset the initially loaded value of top. The displayed values revert.

View the Program Counter Address

Reposition the view and cursor back to the current CS:EIP (program counter). In the Source window, open the Edit menu and choose Go To CS:EIP. The display returns to the startup module.

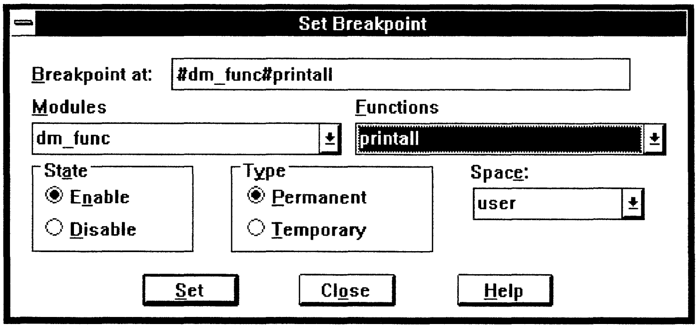
Control Emulation With Breakpoints

In this part of the tutorial, you will learn different ways to run and halt emulation, including the use of breakpoints, the source cursor, and immediate emulation controls.

Set a Breakpoint With the Source Window Breakpoints Menu

In the Source window, open the Breakpoints menu and choose Set Breakpoint. In the Set Breakpoint dialog box, select the dm_func module and the printall function. Choose Set to set the breakpoint.

Set Breakpoint dialog box, setting a permanent, enabled breakpoint at the beginning of the printall function in the dm_func module



The breakpoint is marked with a red highlight on the first assembly line at the specified address. Close the Set Breakpoint dialog box.

Set a Breakpoint With the Source Window Mouse Cursor

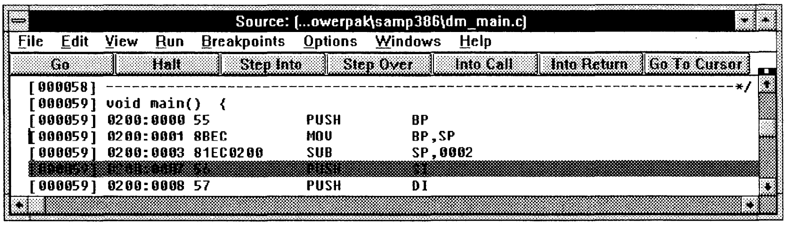
You can use the mouse to set a breakpoint on a specific line or instruction in the Source window. When you point the mouse to the left of a source line, the SLD software displays a cross-hair cursor (shown here in the left margin).

Use the mouse to set a breakpoint at the beginning of main:

1. Browse to the dm_main module. In the Source window, open the File menu; choose Browse Modules; in the Browse Modules dialog box, select dm_main from the list box.
2. Scroll the display to line 59. Open the Edit menu; choose Go To Line; in the Go To Line dialog box, enter 59. The source display in mixed view shows the source line 59 followed by the associated instructions disassembled from memory.
3. Position the mouse pointer in the left margin of the Source window beside the assembly line [000059] 0200:0007 56 PUSH SI.

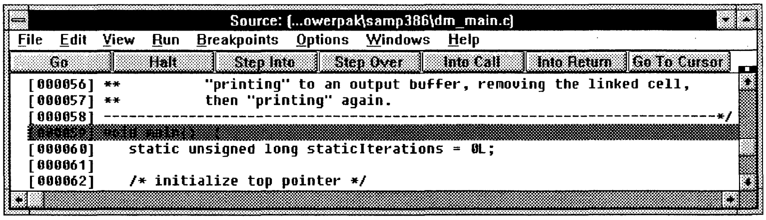
When the mouse pointer becomes a cross-hair cursor, click a mouse button to set the breakpoint. The primary button sets a permanent breakpoint. The secondary button sets a temporary breakpoint. The breakpoint line is highlighted in red.

Source window showing breakpoint line in mixed source and assembly view



Mixed view highlights the assembly line with the breakpoint. Source-only view highlights the entire source line, regardless of which instruction or statement has the breakpoint. Open the View menu and choose Source Only to display source without interleaved disassembly.

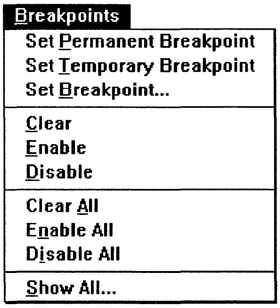
Source window showing breakpoint line in source-only view



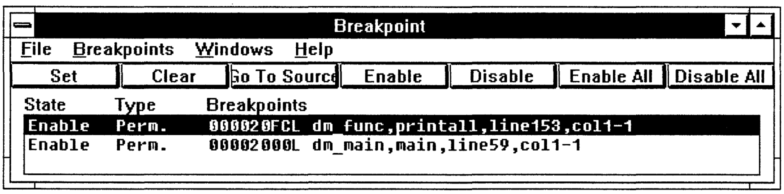
View the Currently Set Breakpoints

Open the Source window Breakpoints menu and choose Show All. The Breakpoint window appears, listing all currently set breakpoints.

Source window Breakpoints menu

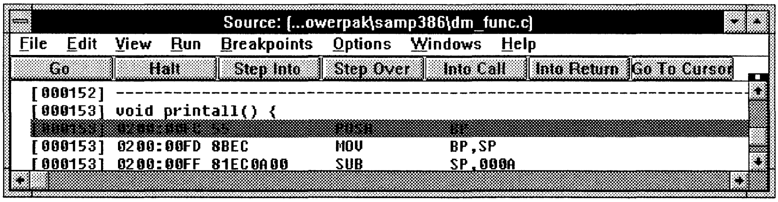


Breakpoints window opened by the Source window Breakpoints menu Show All item

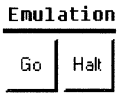


With the highlight on the printall breakpoint, choose the Go To Source button. The Source window displays the breakpoint line, as follows.

Source window display (in mixed source and assembly view) corresponding to a line in the Breakpoint window



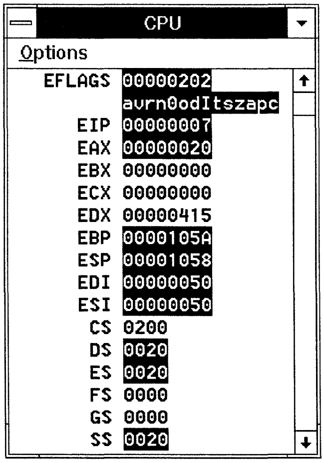
Emulate To a Breakpoint

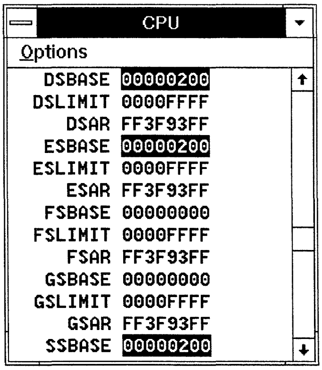


On the Toolbar, choose the Go button. Emulation starts, then breaks on the first breakpoint. The new CS:EIP address is marked by >> (the program counter) as well as by the red highlight (a breakpoint).

Look at the CPU window. During emulation, some CPU register values changed and are highlighted.

Registers in CPU window modified by program execution



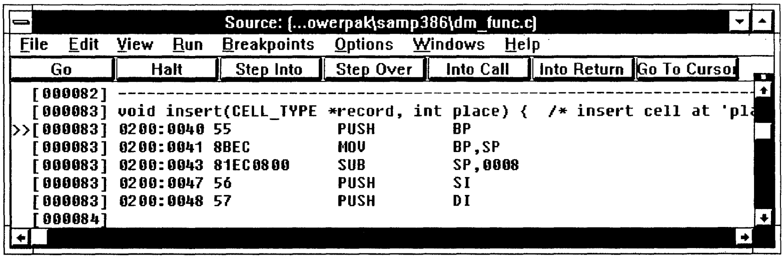


Control Emulation With Buttons and Menus

Emulate By Stepping

In the Source window, open the Run menu and choose Step Into to step into an executable statement. The CS:EIP moves to the next executable statement. In mixed view, stepping is one assembly line at a time; in source-only view, stepping is one source line at a time. Ensure the display is in mixed view before continuing this tutorial.

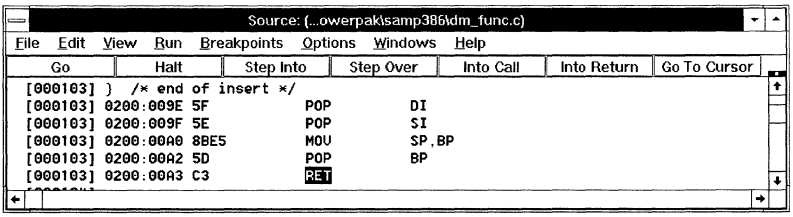
Continue choosing Step Into until the CS:EIP is on the disassembly line [000067] 0200:0017 E82600 CALL dm_func#83 (insert). The next Step Into will emulate into the insert function call, changing the source display to show the CS:EIP indicator on the first executable line of the insert function. Choose Step Into again.



Source window display after stepping into the insert function

Emulate to the Cursor

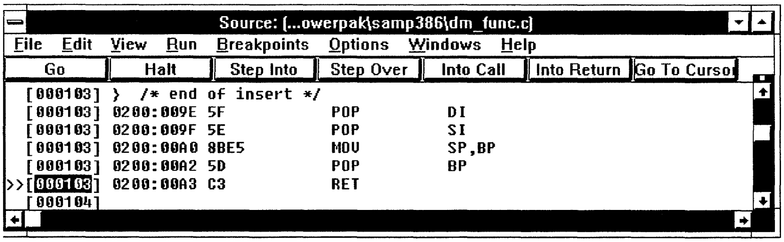
You can progress emulation to a specific line without setting a breakpoint. Open the Source window Edit menu; choose Go To Line and enter 103. With the mouse, click somewhere on the line [000103] 0200:00A3 C3 RET (not in the left margin) to position the source cursor on the return instruction.



Source window cursor on RET instruction

Open the Run menu and choose Goto Cursor. The CS:EIP moves to the location you selected with the cursor, as shown in the following.

Source window displaying current execution point (CS:EIP), indicated by >>

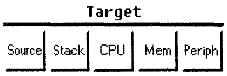


Go To commands in the Edit menu move the cursor without without changing the CS:EIP. Go To commands in the Run menu change both the CS:EIP and the source cursor.

View the Call Stack

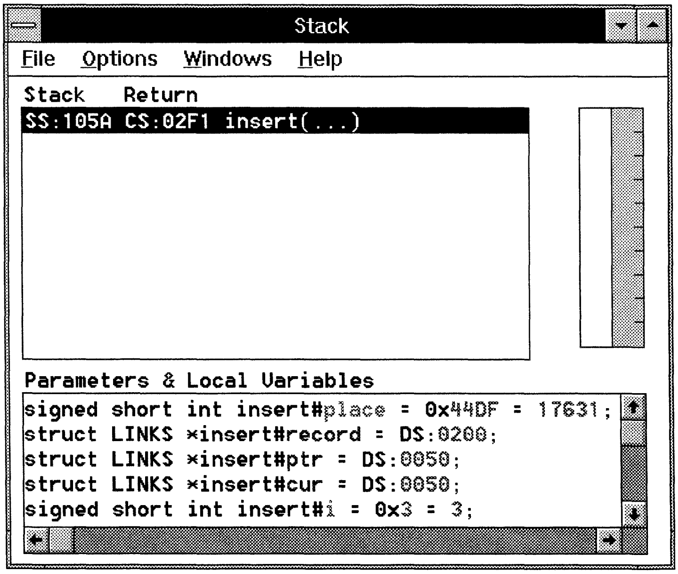
Once you load and execute a program through several function calls, the stack list contains stack frames. In this part of the tutorial, you will learn how to configure and read the Stack window.

Open the Stack Window



On the Toolbar, choose Stack.

Stack window showing the stack and return addresses, parameters, and local variables of the function where emulation is halted



The Stack window contains two panes and a stack meter:

Stack list — (top pane) displays the nested call sequence leading up to the current context. Each call is shown as a single line, called a frame. Click on a frame in the stack list to highlight (select) it. Stack and code addresses appear, depending on the Options menu settings, in each frame:

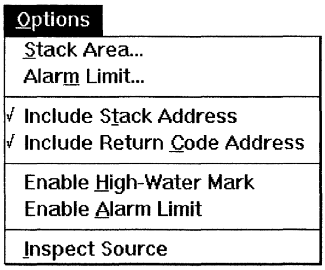
- Stack address — is the address of the frame in the stack. Dashes mean the compiler generated no stack frame for the function.
- Code address — is the address in the calling function to which the program counter will return.

Variables list — (bottom pane) displays the local variables and parameters of the selected frame in the stack list. An unknown value for a variable means the variable is not in scope and is not saved on the stack. The colors in the Stack window variables list have the same meanings as in the Variable window.

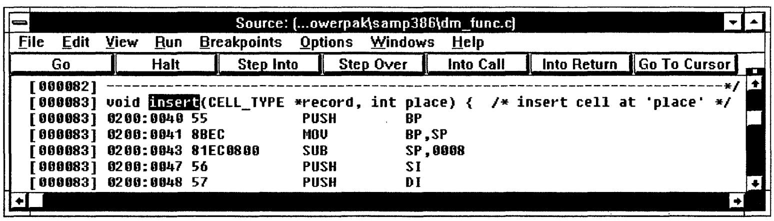
Stack meter — (to the right of the top pane) shows what percentage of the stack area is currently used. Blue shows current stack usage; yellow shows stack underflow; purple shows stack overflow. You can configure the stack meter for statistical information about stack area usage.

You can view any listed function's source. With the highlight on the insert frame in the stack list, open the Options menu and choose Inspect Source.

Stack window Options menu



Source window showing source and disassembly for the insert function on the stack

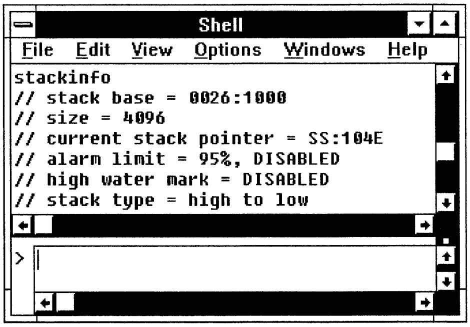


Track Stack Usage Statistically

In this section, you will configure the stack meter to show stack usage statistics. For the stack meter to be active, the stack base and stack size known to the emulator must logically match the current SS and accomodate the current ESP used by the program.

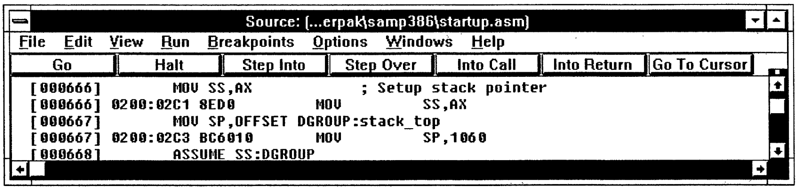
The initial stack base was reported in the Load Complete information box immediately after you loaded demo.omf. This value is unchanged because you have entered no Shell or menu command to change it. To retrieve the stack base, enter stackinfo in the Shell window.

Shell window StackInfo command

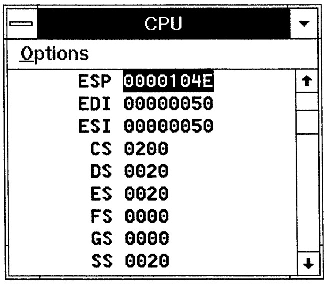


The initial stack base was put into the loadfile during linking. The startup code changes SS:ESP.

Source of startup code initializing SS:ESP



CPU window showing the current values of SS and ESP, with ESP highlighted to indicate change during emulation

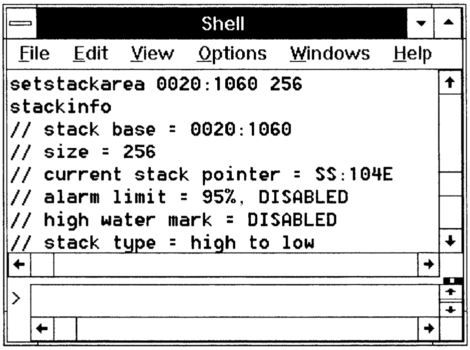


The stack meter in the Stack window monitors stack activity within an area defined by a stack base and size that you can set with Shell commands and Stack window menu items. When the SS:ESP is inside this monitored stack area, the stack meter is active and the statistical options (stack alarm and high-water mark) can be enabled.

The SS:ESP is 0020:104E, outside of the stack area defined by the Stack window stack base (0026:1000) and size (4096). However, this SS:ESP is consistent with the SS:ESP set in startup (no subsequent code has relocated the stack).

Change the Stack window stack base to 0020:1060 to match the startup SS:ESP. To make small stack usages visible on the stack meter, reduce the Stack window stack size to 256. Use setstackarea for both changes, then stackinfo to display the new monitored area.

Shell window SetStackArea and StackInfo commands

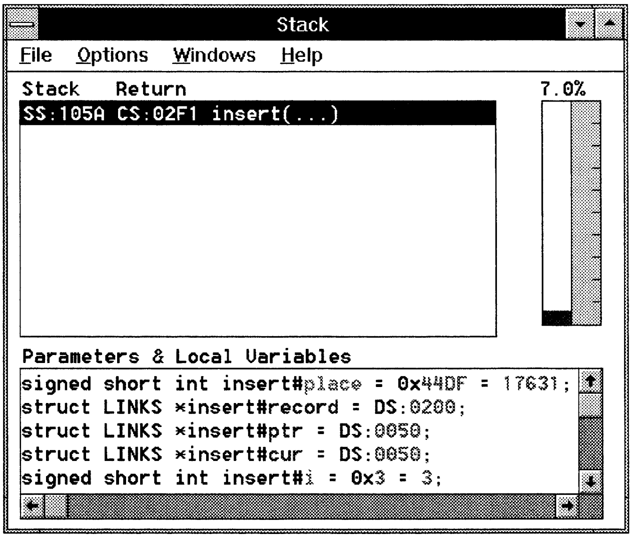


Stack CAUTION

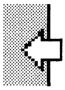
Avoid changing the SS and ESP shown in the CPU window. Changing these registers would affect the stack used by your program.

The stack meter in the Stack window becomes active.

Stack window with active Stack meter, showing 7% (about 18 bytes) of the stack area (256 bytes at 0020:1060) in use

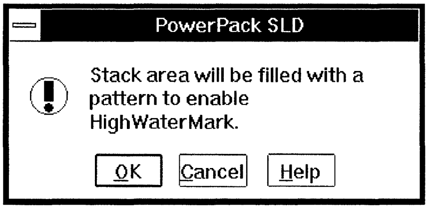


Changing the stack size recognized by the emulator does not affect the amount of memory available to the program for stack activity. Changing the stack base recognized by the emulator does not affect the SS or ESP. The stack base and size are used only by the emulator to maintain the stack usage statistics.



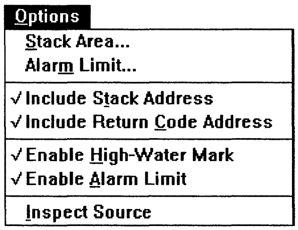
Display the greatest stack usage since stack initialization. Open the Options menu and choose Enable High Water Mark. The high-water mark appears as an arrow on the stack meter (see figure at left). Initially, the high-water mark is set at the current level of stack usage, and any unused part of the monitored stack area is filled with a pattern.

Warning message that a pattern will be written to the unused part of the monitored stack area

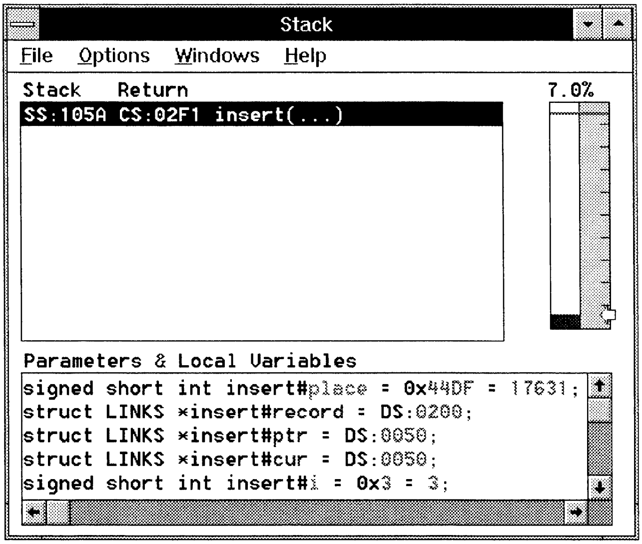


Set a stack alarm to warn when stack usage exceeds a specified percentage of the stack area. Open the Options menu and choose Enable Alarm Limit. The alarm limit appears as a red line at 95% on the stack meter. Each time emulation stops, if the alarm limit is currently exceeded a warning message appears. The following shows the Options menu with the high-water mark and alarm enabled.

Stack window Options menu with high-water mark and alarm limit enabled



Stack meter with high-water mark and alarm limit enabled



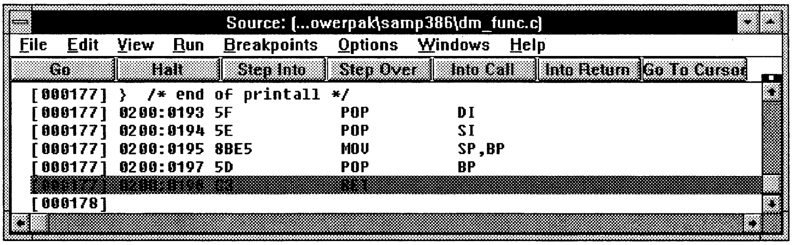
The Stack window is updated only when emulation stops. To monitor the stack during execution, emulate with Step Into/Over Continuously from the Source window Run menu. Each step updates the Stack window. Choose Halt to stop stepping.

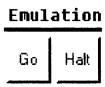
Collect and Examine Trace Information

In this part of the tutorial, you will collect and view trace information.

Ensure no breakpoints are set. In the Source window, open the Breakpoints menu and choose Clear All. In mixed view, set a breakpoint on the RET instruction at the end of printall.

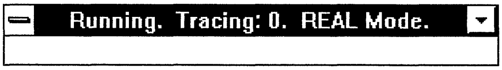
Source window showing breakpoint on printall RET instruction





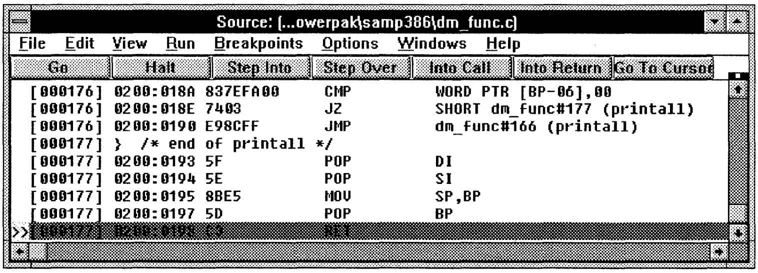
On the Toolbar, choose the Go button. Tracing starts when emulation starts and stops when emulation stops. The Status window title lists the emulation and tracing status.

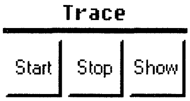
Status icon and window during emulation and tracing



Emulation and tracing stop at the breakpoint.

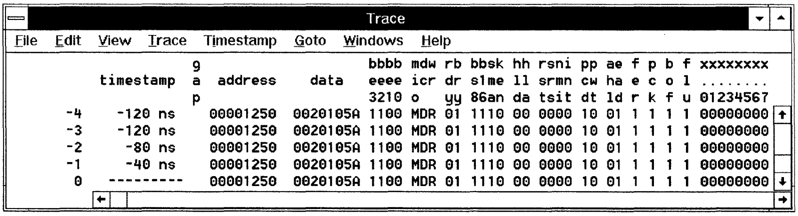
Source window showing emulation halted at the breakpoint





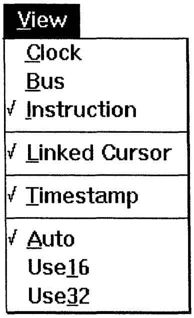
On the Toolbar, choose the Show button to display the Trace window. The trace information appears as bus cycles. Scroll back to address 000218A, corresponding to address 0200:018A in the Source window. You can see the opcodes on the data bus corresponding to the opcodes in the Source window disassembly. (In a Memory window showing the hexadecimal values, you can see the same opcodes there.) Besides the opcodes at code addresses, other values appear on the data bus associated with addresses not among the loaded code. These are memory reads and writes resulting from the opcode executions.

EA-486 trace display

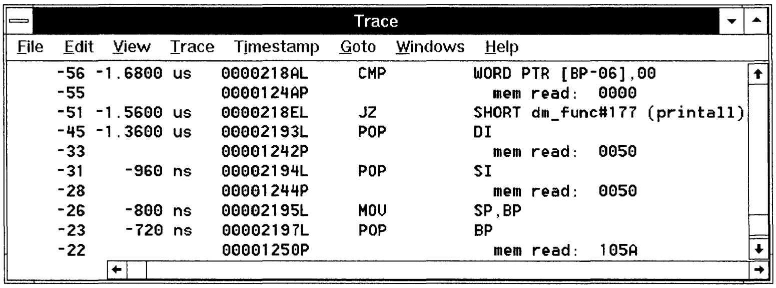


In the Trace window, open the View menu and choose Instruction. The trace information is disassembled.

Trace window View menu specifying trace be displayed as disassembly

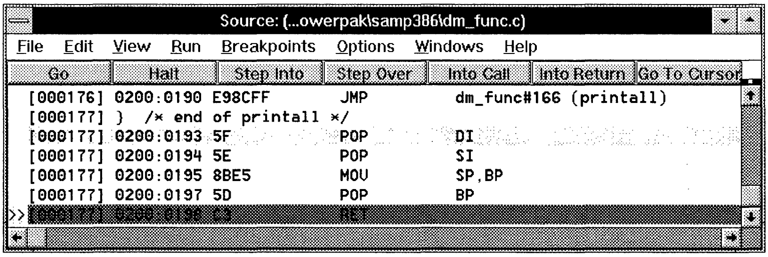


Disassembled trace in Trace window



With the Trace window in instruction view, open the View menu again and choose Linked Cursor. Yellow highlights appear on corresponding lines in the Source and Trace windows. Scroll the Trace window and observe how the Source window scrolls synchronously.

Source window display corresponding to Trace window display (linear address 02193, appearing at frame -45 in the above Trace window)



This is the end of the tutorial for the SW emulator.

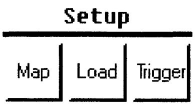
Control Emulation and Tracing With Triggers

The remainder of this tutorial requires an EA emulator.

In this part of the tutorial, you will:

- Define events and triggers for collecting trace information.
- View the collected trace information and find the triggering event.

View And Configure The Trigger Window



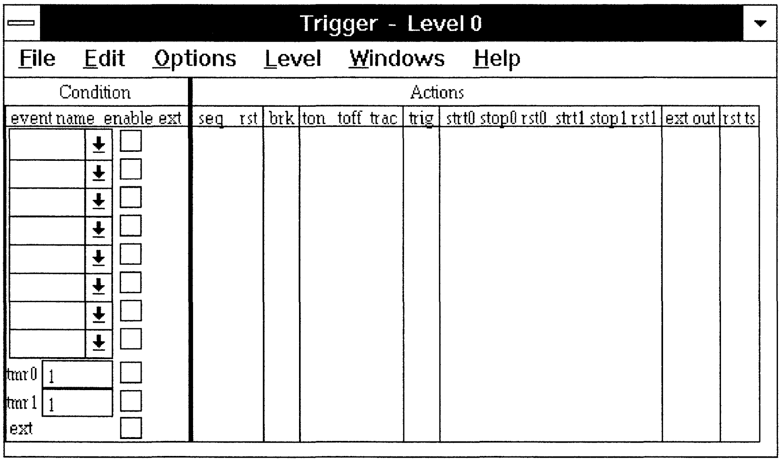
On the Toolbar, choose the Trigger button. The Trigger window has two panes:

Condition (on the left) causes the trigger. Conditions can be events, counter or timer values, and external active-low signals.

Actions (on the right) respond to each condition. Actions can control trace collection, emulation, and trigger conditions.

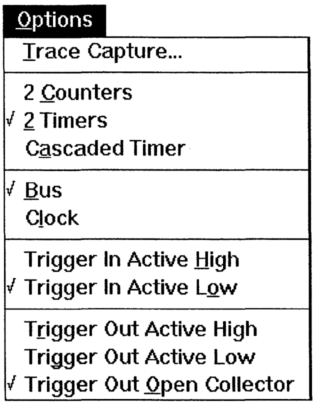
If the labels on the conditions and actions are displayed incorrectly, a screen font manager such as Adobe Type Manager may be overriding the SLD software default font. In Windows, turn off the font manager

Trigger window



If the Trigger window on your screen shows a pair of counters or timers, open the Options menu and choose Cascaded Timer.

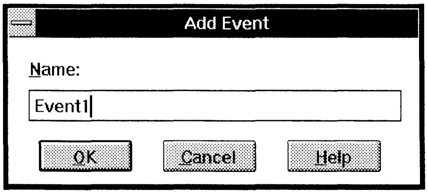
Trigger window
Options menu with 2
Timers enabled



Define an Event to Trigger Trace Collection

In the Trigger window, click on an event area (or open the Edit menu and choose Events). The Event window appears. Since no events are yet defined, the Add Event dialog box also appears. Enter the name Event1.

Add Event dialog box, defining the new Event1 event.

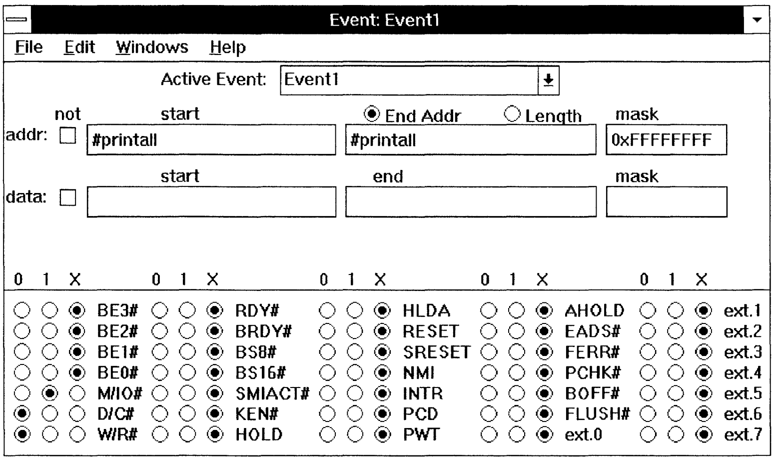


Editing the Event window differs from filling-in a dialog box. Pressing <Enter> has no effect on the field you are editing. Pressing <Delete> can cause an error. To ensure a field accepts an entry:

1. Position the cursor in the field.
2. Type the value.
3. Move the cursor to a different field.

Edit the Event window as shown in the following figure (the signals in the Event window may be different for your processor), defining Event1 as the execution of an instruction from any of the first 10 memory locations where printall is loaded. After filling-in all necessary fields, move the cursor to a different field and close the Event window.

Event window defining Event1 as a memory code read (instruction fetch) at the beginning of the printall function



Specify Trace Capture Options

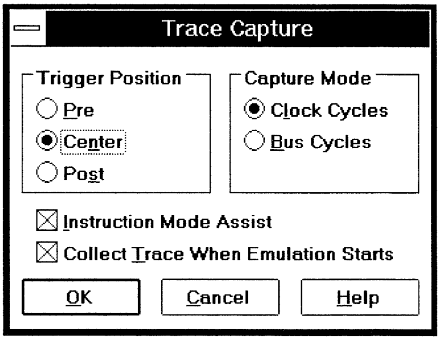
Open the Trigger window Options menu Trace Capture dialog box and select the options as follows:

- Trigger Position Center, to stop trace 125000 cycles after a trig action
- Capture Mode Clock Cycles, to capture trace as clock cycles that can be disassembled

- Instruction Mode Assist, to capture the branch messages needed for disassembly
- Collect Trace When Emulation Starts, to start tracing when you start emulating

Trace Capture dialog box specifying that:

- Trace capture starts when emulation starts and stops 125000 clock cycles after a trig action.
- Clock cycles and branch messages are captured.



Define the Action to Take When an Event Occurs

In the Trigger window, select the top Event box to specify Event1. Select the enable box to enable the trigger. Select the trig check box, so when Event1 occurs the emulator will:

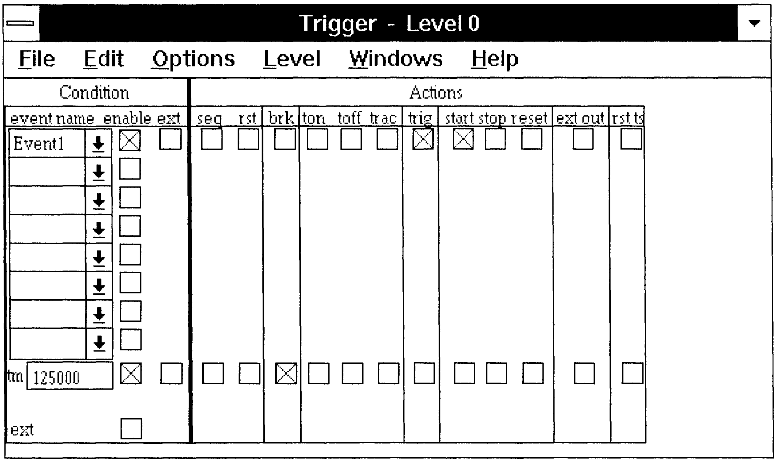
1. Trigger.
2. Fill the trace buffer, positioning Event1 according to the Trace Capture specifications.
3. Turn off tracing without halting emulation.

To halt emulation at the same time as halting trace:

1. For Event1, select the start check box, to start the timer.
2. The trigger position in the Trace Capture dialog box is Center, so trace is captured for 125000 clock cycles after Event1. Set the timer to halt emulation after 125000 clock cycles.

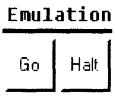
Trigger window (EA), showing the following enabled for the Level 0 trigger:

- the Event1 event condition to perform a trig action and start the timer
- the Cascaded Timer condition to break emulation after 125000 clock cycles



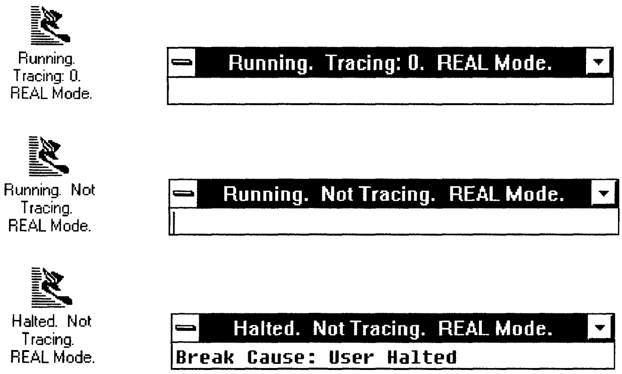
Collect, View, and Save the Trace Information

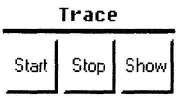
Ensure no breakpoints are set. In the Source window, open the Breakpoints menu and choose Clear All.



From the Emulation section of the Toolbar, choose the Go button. Tracing starts automatically with emulation. When the Event1 trigger occurs, tracing stops and emulation continues. When the Cascaded Timer trigger occurs, emulation halts

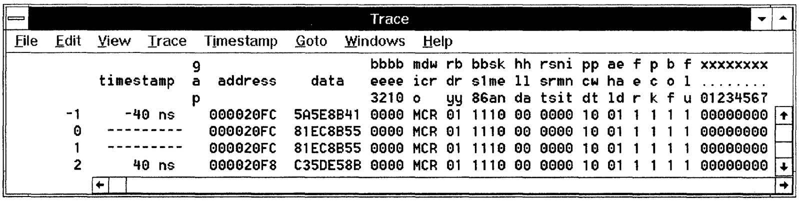
Status window progression during emulation with trace, during emulation without trace, and after emulation halts





From the Trace section of the Toolbar, choose the Show button. The following figure shows trace buffer 0 displayed in a PowerPack Trace window. Read the signal mnemonic labels vertically. The trace frame number appears in the leftmost column.

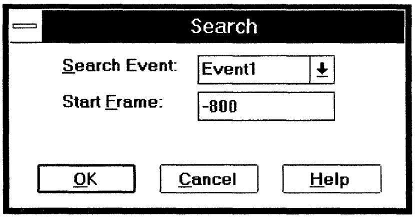
Trace window, displaying trace as clock cycles, with the trigger frame in view



Using the View menu, display the trace as bus cycles and instructions. Observe the frame numbers. Because bus cycles and instructions can span multiple clock cycles, frame numbers in those views are discontinuous.

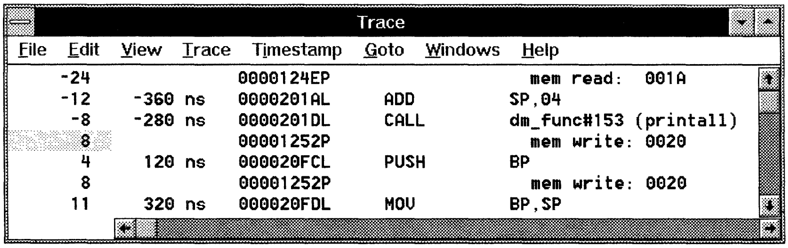
Scroll the section of trace around the trigger frame (frame 0) off screen and find it again with the Edit menu Search dialog box. The trigger frame is the clock cycle matching the condition that caused the trig action: Event1. In the 125000 clock cycles of trace collected after the trigger, the condition may have occured again, so the first occurrance of Event1 is the trigger frame. In the Search dialog box, specify Event1 and a negative frame number.

Search dialog box to find the trigger frame by its event definition



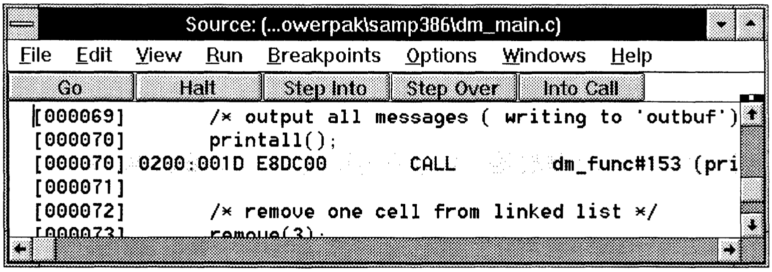
View the trace information as disassembly

Trace window showing trace as disassembly near the trigger frame



Ensure the Source window is open in mixed view. In the Trace window, open the View menu and enable Linked Cursor. The Source window displays the disassembly and source lines corresponding to the Trace window display. Scrolling the Trace window scrolls the Source window synchronously.

Source window showing the in mixed source and assembly corresponding to the Trace window display



Target Hardware

This chapter provides schematics and signal information specific to the EA-486 and SW-386 emulators and SAST boards.

Trace and Event Window Signals

The Trace (SW and EA emulators) and Event window (EA emulators) display signal mnemonics corresponding to the processor mnemonics. Different signals are supported for different emulators.

Address and data bus traces are supported for all emulators. Address trace is 24, 26, or 32 bits wide. Data trace is 16 or 32 bits.

486 Emulators

The following describes the supported signals for the EA-486 Trace and Event windows. Active-low signals are indicated by an octothorp (#). Check marks (✓) indicate which signals are supported for which processors. The 486 DX and SX entries also apply to DX2 and SX2, respectively.

486 Processor Type				**Mnemonic Labels in Window**	
SLE		**non-SLE**			
DX	**SX**	**DX**	**SX**	**Trace**	**Event**
✓	✓	✓	✓	AHL	AHOLD
✓	✓	✓	✓	BE0	BE0#
✓	✓	✓		BE1	BE1#
✓	✓	✓	✓	BE2	BE2#
✓	✓	✓	✓	BE3	BE3#
✓	✓	✓	✓	BOF	BOFF#
✓	✓	✓	✓	BRY	BRDY#
✓	✓	✓	✓	B16	BS16#
✓	✓	✓	✓	BS8	BS8#
✓	✓	✓	✓	DC	D/C#
✓	✓	✓	✓	EAD	EADS#
✓		✓		FER	FERR#

486 Emulator Trace and Event Window Signals (continued)

486 Processor Type				Mnemonic Labels in Window	
SLE		non-SLE			
DX	SX	DX	SX	Trace	Event
✓	✓	✓	✓	FLU	FLUSH#
✓	✓	✓	✓	HLA	HLDA
✓	✓	✓	✓	HLD	HOLD
✓	✓	✓	✓	INT	INTR
✓	✓	✓	✓	KEN	KEN#
✓	✓	✓	✓	MIO	M/IO#
✓	✓	✓	✓	NMI	NMI
✓	✓	✓	✓	PCD	PCD
✓	✓	✓	✓	PCK	PCHK#
✓	✓	✓	✓	PWT	PWT
✓	✓	✓	✓	RDY	RDY#
✓	✓			SMI	SMI#
✓	✓			SMA	SMIACT#
✓	✓			SRS	SRESET
✓	✓	✓	✓	WR	W/R#

386 Emulators

The following describes the supported signals for the SW-386 and EA-386 Trace and Event windows. Active-low signals are indicated by octothorps (#). Check marks (✓) indicate which signals are supported for which processors. Singly marked signals are supported by the EA only. Doubly marked (✓✓)signals are supported by both the SW and the EA.

386 Processor Type			Mnemonic Labels in Window	
CX	SX	EX	Trace	Event
✓			A20	A20M#
✓	✓	✓	ADS	ADS#
✓✓	✓✓	✓✓	BHE	BHE#
		✓✓	BS8	BS8#
✓	✓	✓	BSY	BUSY#
		✓	CS6	CS6#

386 Emulator Trace and Event Window Signals (continued)

386 Processor Type			Mnemonic Labels in Window	
CX	**SX**	**EX**	**Trace**	**Event**
✓✓	✓✓	✓✓	DC	D/C#
		✓	DQ0	DRQ0
		✓	DQ1	DRQ1
		✓	DK0	DACK0#
		✓	DK1	DACK1#
		✓	EOP	EOP#
✓	✓	✓	ERR	ERROR#
✓	✓	✓	GAP	unavailable
✓	✓		HLA	HLDA
✓	✓		HLD	HOLD
		✓	IN4	INT4
		✓	IN5	INT5
		✓	IN6	INT6
		✓	IN7	INT7
✓	✓		INT	INTR
		✓	LBA	LBA#
✓	✓		LCK	LOCK#
✓✓	✓✓	✓✓	MIO	M/IO#
✓	✓	✓	NA	NA#
✓	✓	✓	NMI	NMI
		✓	P10	P1.0
		✓	P11	P1.1
		✓	P12	P1.2
		✓	P13	P1.3
		✓	P14	P1.4
		✓	P15	P1.5
		✓	P16	P1.6
		✓	P17	P1.7
		✓	P20 - P24	P2.0 - P2.4
		✓	P25	P2.5

386 Emulator Trace and Event Window Signals (continued)

386 Processor Type			Mnemonic Labels in Window	
CX	**SX**	**EX**	**Trace**	**Event**
		✓	P26	P2.6
		✓	P27	P2.7
		✓	P30 - P31	P3.0 - P3.1
		✓	P32 - P35	P3.2 - P3.5
		✓	P36	P3.6
		✓	P37	P3.7
✓	✓	✓	PER	PEREQ
✓	✓	✓	RDY	READY#
✓	✓	✓	RST	RESET
✓		✓	SMI	SMI#
✓✓		✓✓	SMA	SMIACT#
		✓	UCS	UCS#
		✓	WDT	WDTOUT
✓✓	✓✓	✓✓	WR	W/R#

Chip Select Registers Saved and Restored for the 386 EX

CS0ADL	CS3MSKL	P1CFG	UCSADL	DMACFG
CS0ADH	CS3MSKH	P2CFG	UCSADH	INTCFG
CS0MSKL	CS4ADL	P3CFG	UCSMSKL	TMRCFG
CS0MSKH	CS4ADH	PINCFG	UCSMSKH	SIOCFG
CS1ADL	CS4MSKL	P1LTC		
CS1ADH	CS4MSKH	P1DIR		
CS1MSKL	CS5ADL	P2LTC		
CS1MSKH	CS5ADH	P2DIR		
CS2ADL	CS5MSKL	P3LTC		
CS2ADH	CS5MSKH	P3DIR		
CS2MSKL	CS6ADL			
CS2MSKH	CS6ADH			
CS3ADL	CS6MSKL			
CS3ADH	CS6MSKH			

Configurable Signals

With the CPU window Options menu Signals item or the Shell window Signals command, configure the following signals to be driven by the emulator or from a different source:

386			486 DX or SX		Signal
CX	SX	EX	SLE	non SLE	
✓	✓	✓			READY#
✓	✓	✓	✓	✓	RESET
✓	✓	✓	✓	✓	HOLD
✓	✓	✓	✓	✓	NMI
✓			✓	✓	INTR
✓	✓	✓			NA#
✓	✓	✓			Coprocess
✓	✓	✓			ERROR#
✓	✓	✓			BUSY#
✓	✓	✓			PEREQ
✓			✓	✓	A20M#
✓		✓			SMI#
		✓			INT0_3
		✓			INT4_7
			✓	✓	RDY#
			✓	✓	FLUSH#
			✓	✓	KEN#
			✓		SLE
			✓		SRESET
			✓		SMI#
			✓		STPCLK#

Signal Loading

The following lists the typical AC loads in picoFarads (pF) and the maximum DC loads in microAmps (uA), besides the CPU load, on various 386 EX and 486 signals for low-level (IiL) and high-level (IiH) input currents. The actual AC loads measurable with the emulator in your target system can differ from these calculated loads.

Signal	386 EX Loads			486 Loads		
	IiL uA	IiH uA	AC pF	IiL uA	IiH uA	AC pF
A1	11	11	12			
A2-3	11	11	12	10	10	6
A4-16	11	11	12	11	11	16
A17	11	11	12	13	13	26
A18-25	11	11	12	12	12	20
A20M#				1063	0	8
A26-31				12	12	20
ADS#	100	10	5	10	10	6
AHOLD				10	10	10
BE0-3#				21	21	22
BHE#	11	11	12			
BLAST#				10	10	6
BLE#	11	11	12			
BOFF#				20	20	16
BRDY#				20	20	16
BREQ				no loads; no CPU load		
BS8#	10	10	6	20	20	24
BS16#				20	20	24
BUSY#	0	0	7			
CLK				2410	510	14
CLK2	100	10	5			
CS6#	10	10	6			
D0-31				10	10	26
D/C#	11	11	12	11	11	16
DACK0#	10	10	6			

Signal Loading (continued)

Signal	386 EX Loads			486 Loads		
	IiL uA	IiH uA	AC pF	IiL uA	IiH uA	AC pF
DACK1#	10	10	6			
DP0-3				0	0	8
DRQ0	10	10	6			
DRQ1	10	10	6			
EADS#				10	10	10
EOP#	10	10	6			
ERROR#	0	0	7			
FERR#				10	10	10
FLUSH#				20	20	36
HLDA	101	11	11	30	30	42
HOLD	1	1	20	20	20	16
IGNNE#				0	0	8
INT0	10	10	13			
INT1	10	10	13			
INT2	10	10	13			
INT3	10	10	13			
INT4	1	1	20			
INT5	1	1	20			
INT6	0	0	7			
INT7	0	0	7			
INTR				20	20	36
KEN#				30	30	42
LBA#	10	10	0			
LOCK#	1	1	6	10	10	6
M/IO#	11	11	12	21	21	22
NA#	1	1	20			
NMI	1	1	20	20	20	36
P1.0-P1.4	10	10	6			
P2.0-P2.7	0	0	0			
P3.0-P3.7	0	0	0			

Signal Loading (continued)

Signal	386 EX Loads			486 Loads		
	IiL uA	IiH uA	AC pF	IiL uA	IiH uA	AC pF
PCD				20	20	16
PCHK#				10	10	26
PEREQ	0	0	7			
PLOCK#				no loads; no CPU load		
PWT				10	10	10
RD#	0	0	0			
RDY#				30	30	42
READY	100	10	12			
RESET	101	11	25	30	30	42
SMI#	1	1	20	1063	0	8
SMIACT#	11	11	19	20	20	28
SRESET				30	30	42
SRXCLK	0	0	0			
SSIORX	0	0	0			
SSIOTX	0	0	0			
STPCLK#				1063	0	8
STXCLK	0	0	0			
TCK	no loads; no CPU load			no loads; no CPU load		
TDI	no loads; no CPU load			no loads; no CPU load		
TDO	no loads; no CPU load			no loads; no CPU load		
TMS	no loads; no CPU load			no loads; no CPU load		
TRST#	no loads, no CPU load					
UCS#	10	10	6			
WDTOUT	10	10	6			
W/R#	0	0	0	10	10	6

Managing the 386 EX Signals

The following are suggestions for configuring various 386 EX signals:

RESET is active high synchronized to CLK2. You can pull this signal high or low but it must remain stable for initialization. This signal can be disabled in the CPU window to be driven by the emulator.

READY# must be synchronized to CLK2 with the proper setup time according to Intel specifications for any cycles for which the 386EX is not programmed. After the chip select unit is programmed, such signals include:

- any unmapped memory or I/O space
- any disabled on-chip expanded I/O space
- halt or shutdown cycles

The power-up condition for chip select and ready generation allows upper-chip-select memory accesses to the entire 64M byte address range. READY# must be tri-stated when the 386EX provides the ready due to LBA# cycles. READY# should have a resistor pull-up to VCC or be pulled low with resistor of 600 to 820 ohms for full-time zero wait states.

NA# must be synchronized to CLK2 and driven according to the need for pipelining. Do not float this signal. NA# can be disabled in the CPU window to be driven by the emulator.

BS8# must be synchronized to CLK2 and driven according to the actual bus size. Do not float the signal.

NMI must be driven as needed. When NMI is floated it must be disabled in the CPU window to be driven by the emulator.

SMI# must be driven as needed. When SMI# is floated it must be disabled in the CPU window to be driven by the emulator.

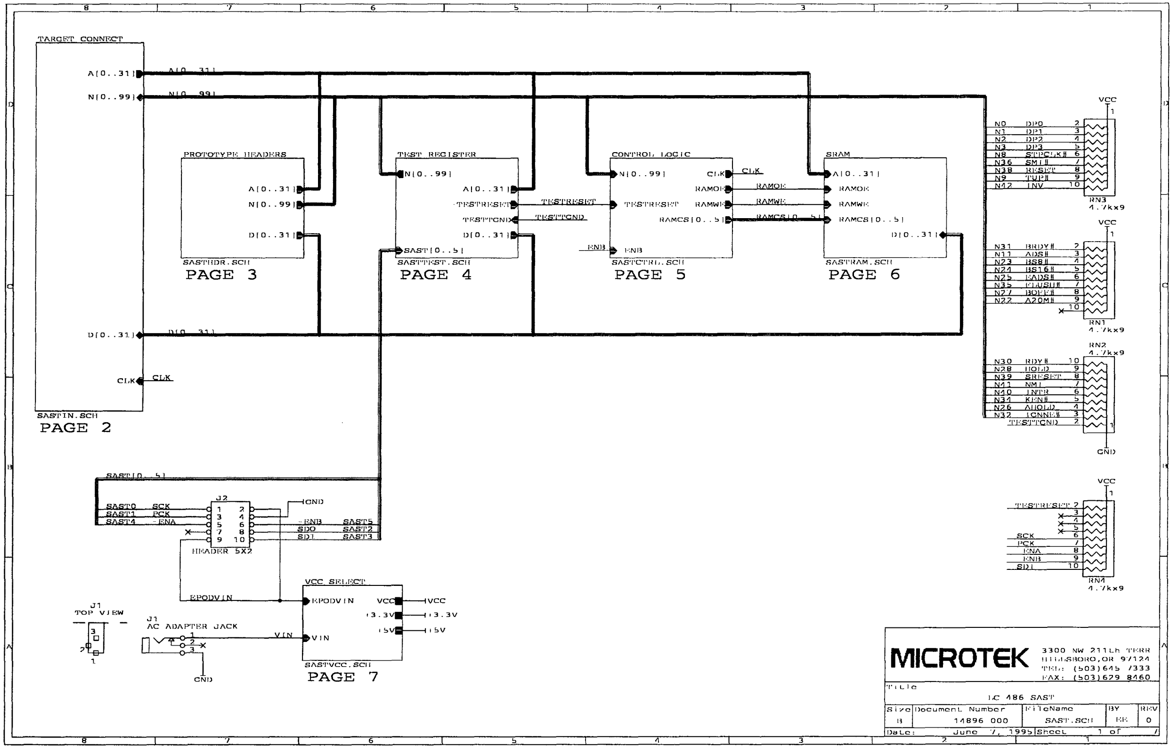
FLT# must have a resistor pull-up to VCC or be floated when the emulator is attached.

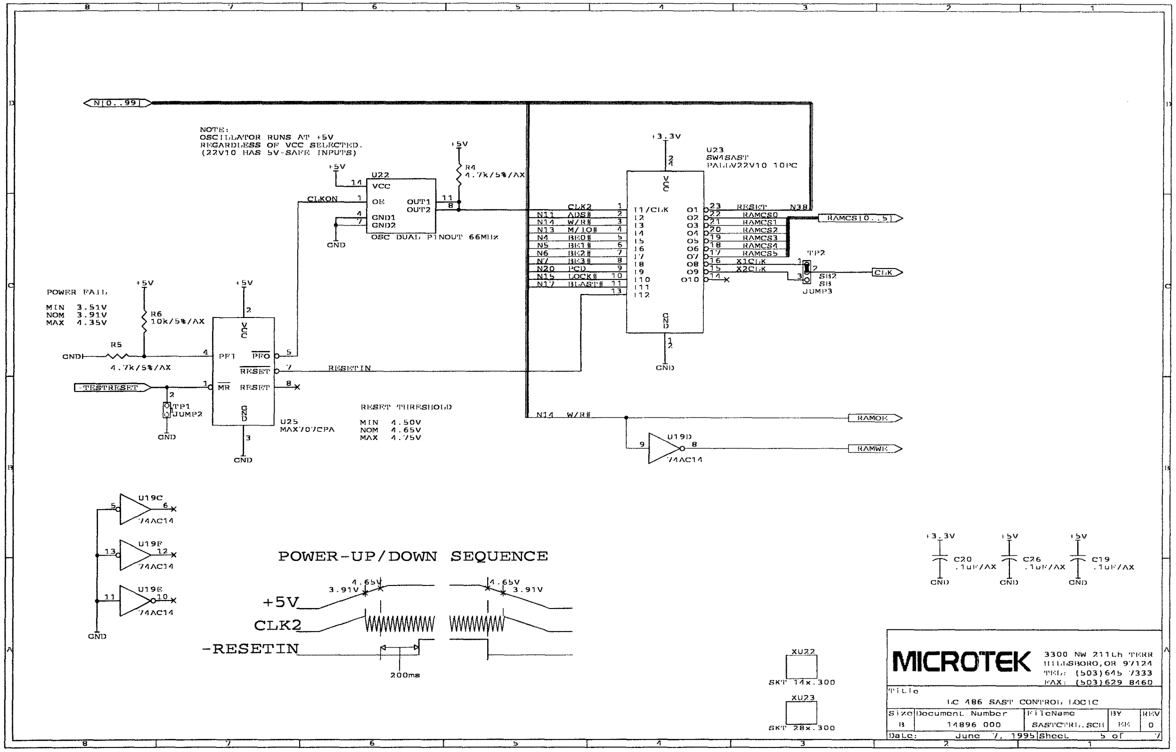
HLDA when this signal is configured as an output port, enter Config IgnoreHLDA On in the Shell window Command Entry pane, to inform the SLD software that the CPU has not granted the bus to another master.

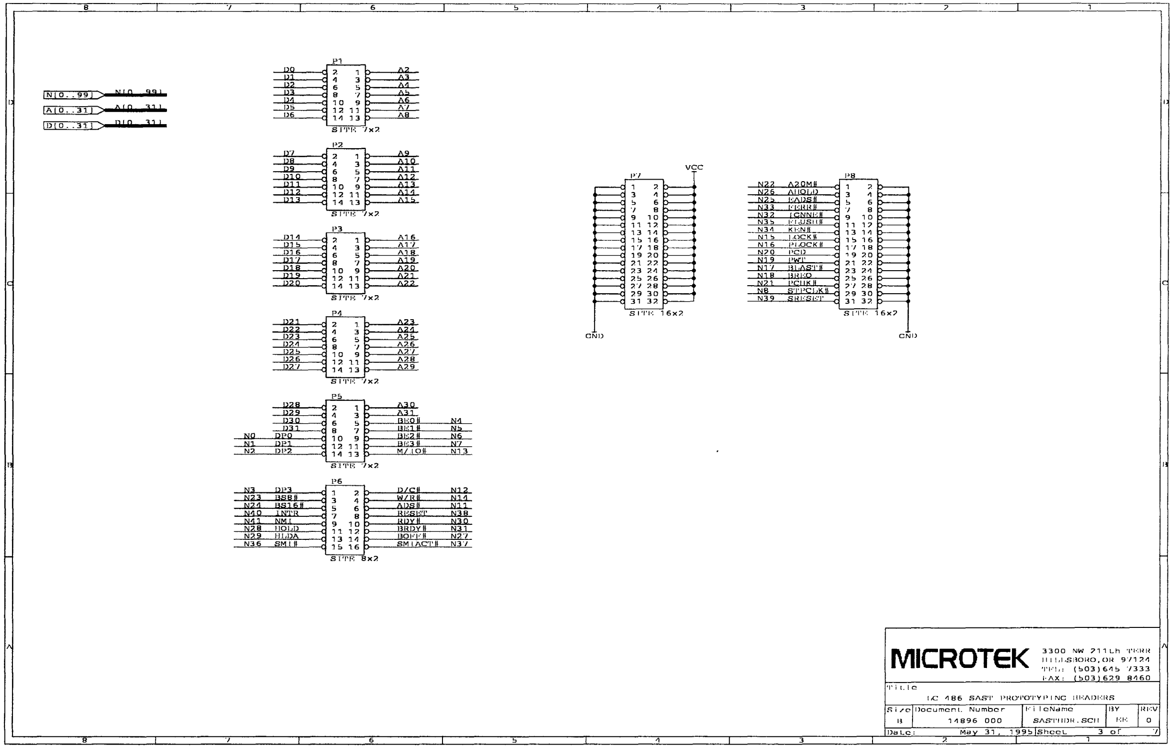
SAST Schematics

486 SAST Board

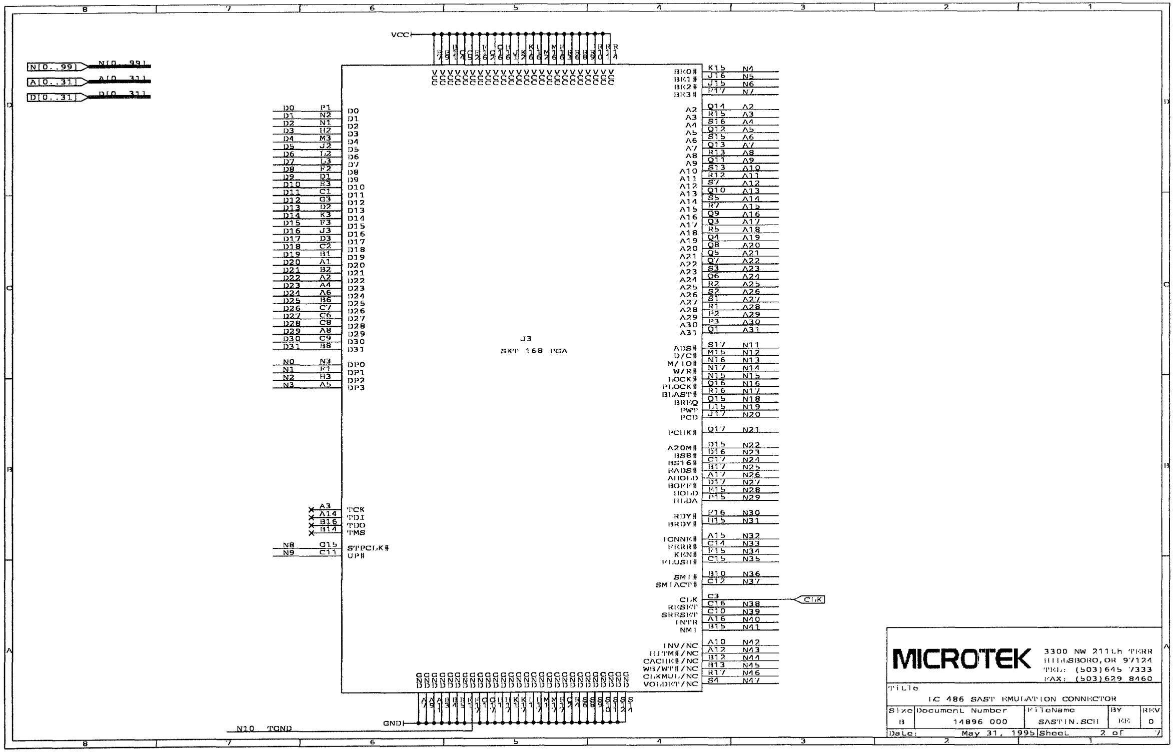
The following pages contain the EA-486 SAST board schematics.

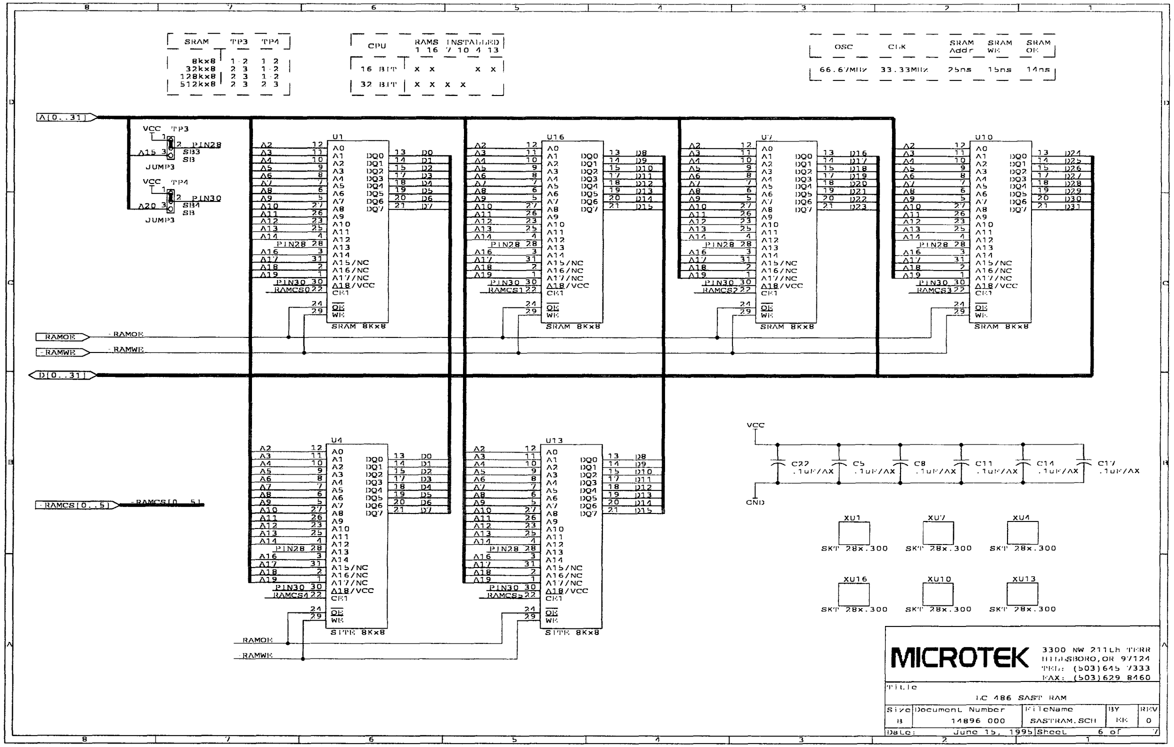

TARGET CONNECT
A[0..31]
N[0..99]
D[0..31]
CLK
SASTIN.SCH
PAGE 2
PROTOTYPE HEADERS
SASTHDR.SCH
PAGE 3
TEST REGISTER
TESTRESET
TESTGND
SAST[0..5]
SASTTEST.SCH
PAGE 4
CONTROL LOGIC
RAMOE
RAMWE
RAMCS[0..5]
ENB
SASTCTRL.SCH
PAGE 5
SRAM
SASTRAM.SCH
PAGE 6
VCC
GND
RN3 4.7kx9
RN1 4.7kx9
RN2 4.7kx9
RN4 4.7kx9
J2
HEADER 5X2
SAST0 SCK
SAST1 PCK
SAST4 ENA
ENB SAST5
SDO SAST2
SDI SAST3
EPODVIN
VCC SELECT
VIN
+3.3V
+5V
SASTVCC.SCH
PAGE 7
J1 TOP VIEW
J1 AC ADAPTER JACK
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST
Size B
Document Number 14896 000
FileName SAST.SCH
BY EE
REV 0
Date: June 7, 1995
Sheet 1 of 7

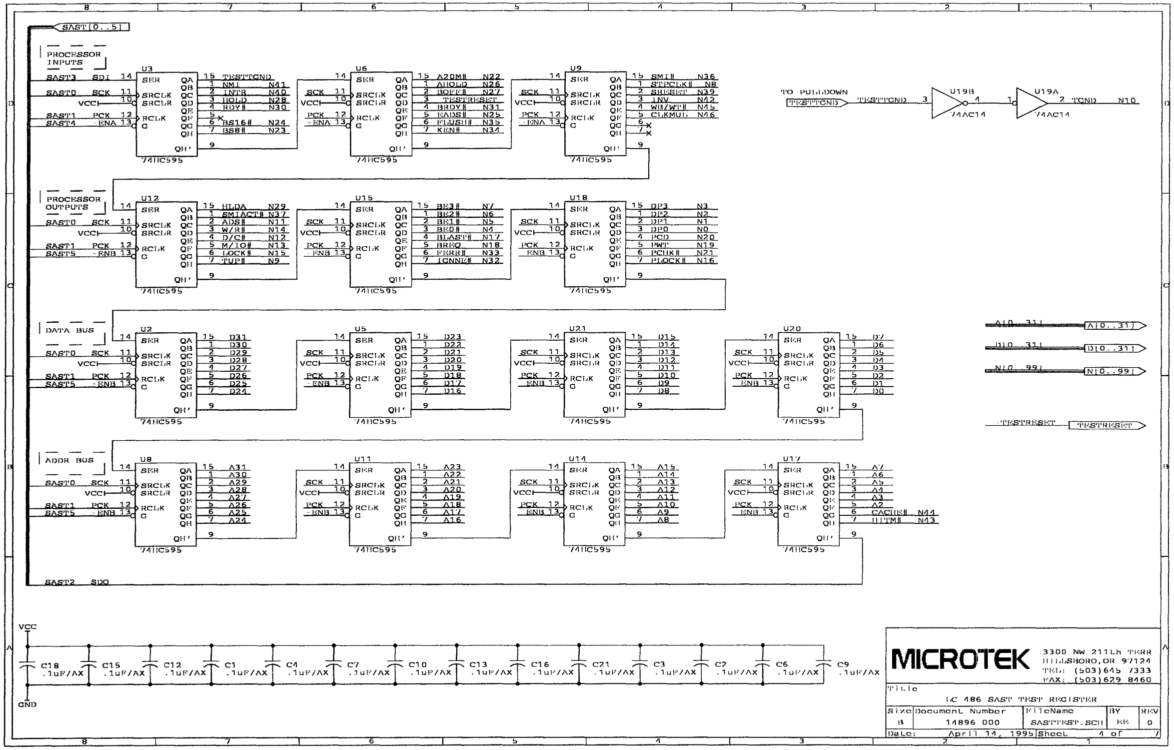

N[0..99]
NOTE:
OSCILLATOR RUNS AT +5V
REGARDLESS OF VCC SELECTED.
(22V10 HAS 5V-SAFE INPUTS)
U22
OSC DUAL PINOUT 66MHz
CLKON
R4
4.7k/5%/AX
U23
SW4SAST
PALLV22V10 10PC
RESET
RAMCS0
RAMCS1
RAMCS2
RAMCS3
RAMCS4
RAMCS5
X1CLK
X2CLK
RAMCS[0..5]
CLK
TP2
SB2
SB
JUMP3
POWER FAIL
MIN 3.51V
NOM 3.91V
MAX 4.35V
R6
10k/5%/AX
R5
4.7k/5%/AX
-TESTRESET
TP1
JUMP2
U25
MAX707CPA
RESETIN
RESET THRESHOLD
MIN 4.50V
NOM 4.65V
MAX 4.75V
N14 W/R#
RAMOE
RAMWE
U19D
74AC14
U19C
U19F
U19E
POWER-UP/DOWN SEQUENCE
+5V
CLK2
-RESETIN
200ms
C20
C26
C19
.1uF/AX
XU22
SKT 14x.300
XU23
SKT 28x.300
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST CONTROL LOGIC
Size B
Document Number 14896 000
FileName SASTCTRL.SCH
BY EE
REV 0
Date: June 7, 1995
Sheet 5 of 7

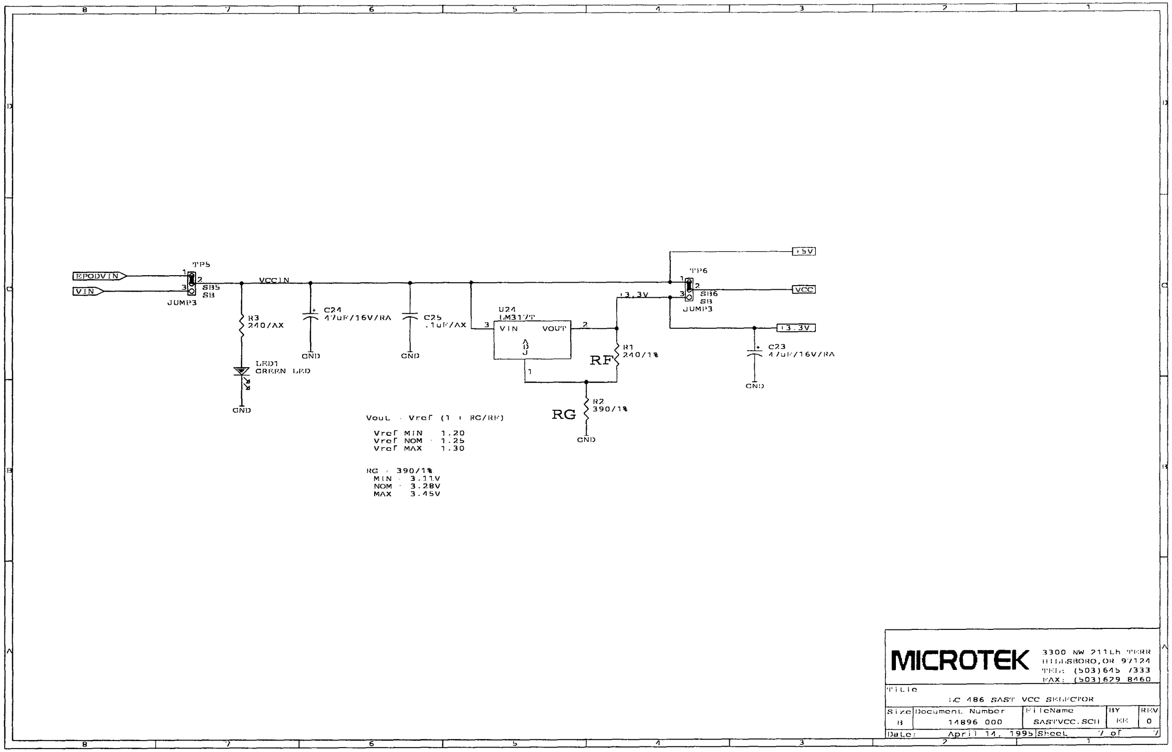


P1
SITE 7x2
P2
SITE 7x2
P3
SITE 7x2
P4
SITE 7x2
P5
SITE 7x2
P6
SITE 8x2
P7
SITE 16x2
P8
SITE 16x2
VCC
GND
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST PROTOTYPING HEADERS
Size B
Document Number 14896 000
FileName SASTHDR.SCH
BY KE
REV 0
Date: May 31, 1995
Sheet 3 of 7


VCC
GND
N[0..99]
A[0..31]
D[0..31]
J3
SKT 168 PGA
CLK
N10 TGND
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
EC 486 SAST EMULATION CONNECTOR
Size B
Document Number 14896 000
FileName SASTIN.SCH
BY EE
REV 0
Date: May 31, 1995
Sheet 2 of 7

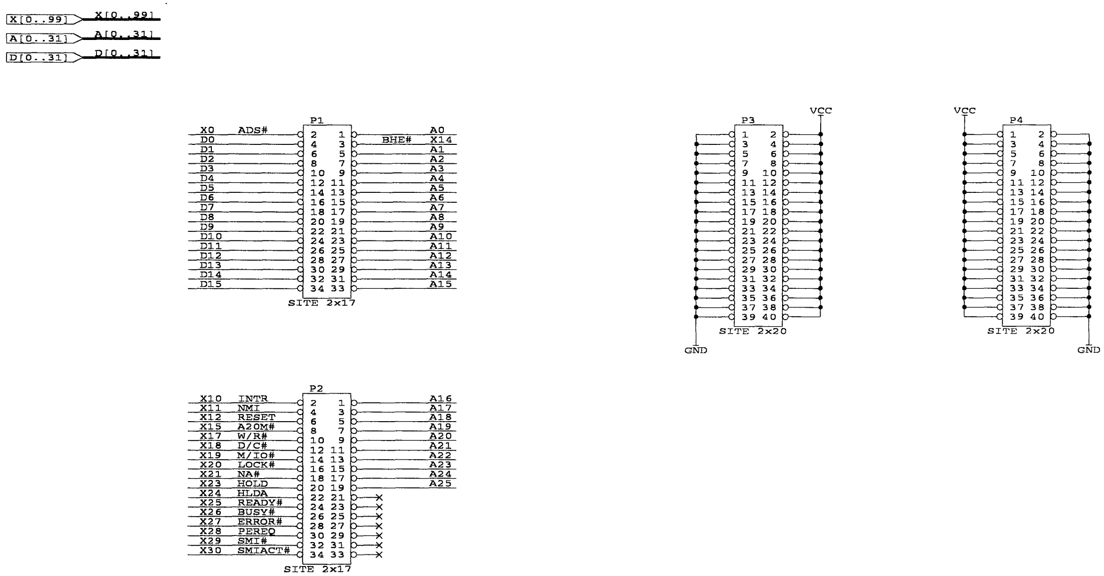

SRAM TP3 TP4
8kx8 1 2 1 2
32kx8 2 3 1 2
128kx8 2 3 1 2
512kx8 2 3 2 3
CPU RAMS INSTALLED 1 16 7 10 4 13
16 BIT X X X X
32 BIT X X X X
OSC CLK SRAM Addr SRAM WE SRAM OE
66.67MHz 33.33MHz 25ns 15ns 14ns
A[0..31]
D[0..31]
-RAMOE
-RAMWE
-RAMCS[0..5]
JUMP3
U1
U16
U7
U10
U4
U13
SRAM 8Kx8
VCC
GND
C22 .1uF/AX
C5 .1uF/AX
C8 .1uF/AX
C11 .1uF/AX
C14 .1uF/AX
C17 .1uF/AX
XU1 SKT 28x.300
XU7 SKT 28x.300
XU4 SKT 28x.300
XU16 SKT 28x.300
XU10 SKT 28x.300
XU13 SKT 28x.300
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST RAM
Size B
Document Number 14896 000
FileName SASTRAM.SCH
BY KK
REV 0
Date: June 15, 1995
Sheet 6 of 7


PROCESSOR INPUTS
PROCESSOR OUTPUTS
DATA BUS
ADDR BUS
TO PULLDOWN
74HC595
74AC14
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST TEST REGISTER
Size B
Document Number 14896 000
FileName SASTTEST.SCH
BY KK
REV 0
Date: April 14, 1995
Sheet 4 of 7


EPODVIN
VIN
TP5
SB5
SB
JUMP3
VCCIN
R3
240/AX
C24
47uF/16V/RA
C25
.1uF/AX
LED1
GREEN LED
GND
U24
LM317T
VIN
VOUT
ADJ
R1
240/1%
RF
R2
390/1%
RG
+3.3V
TP6
SB6
SB
JUMP3
+5V
VCC
C23
47uF/16V/RA
Vout = Vref (1 + RG/RF)
Vref MIN = 1.20
Vref NOM = 1.25
Vref MAX = 1.30
RG = 390/1%
MIN = 3.11V
NOM = 3.28V
MAX = 3.45V
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645 7333
FAX: (503)629 8460
Title
LC 486 SAST VCC SELECTOR
Size B
Document Number 14896 000
FileName SASTVCC.SCH
REV 0
Date: April 14, 1995
Sheet 7 of 7

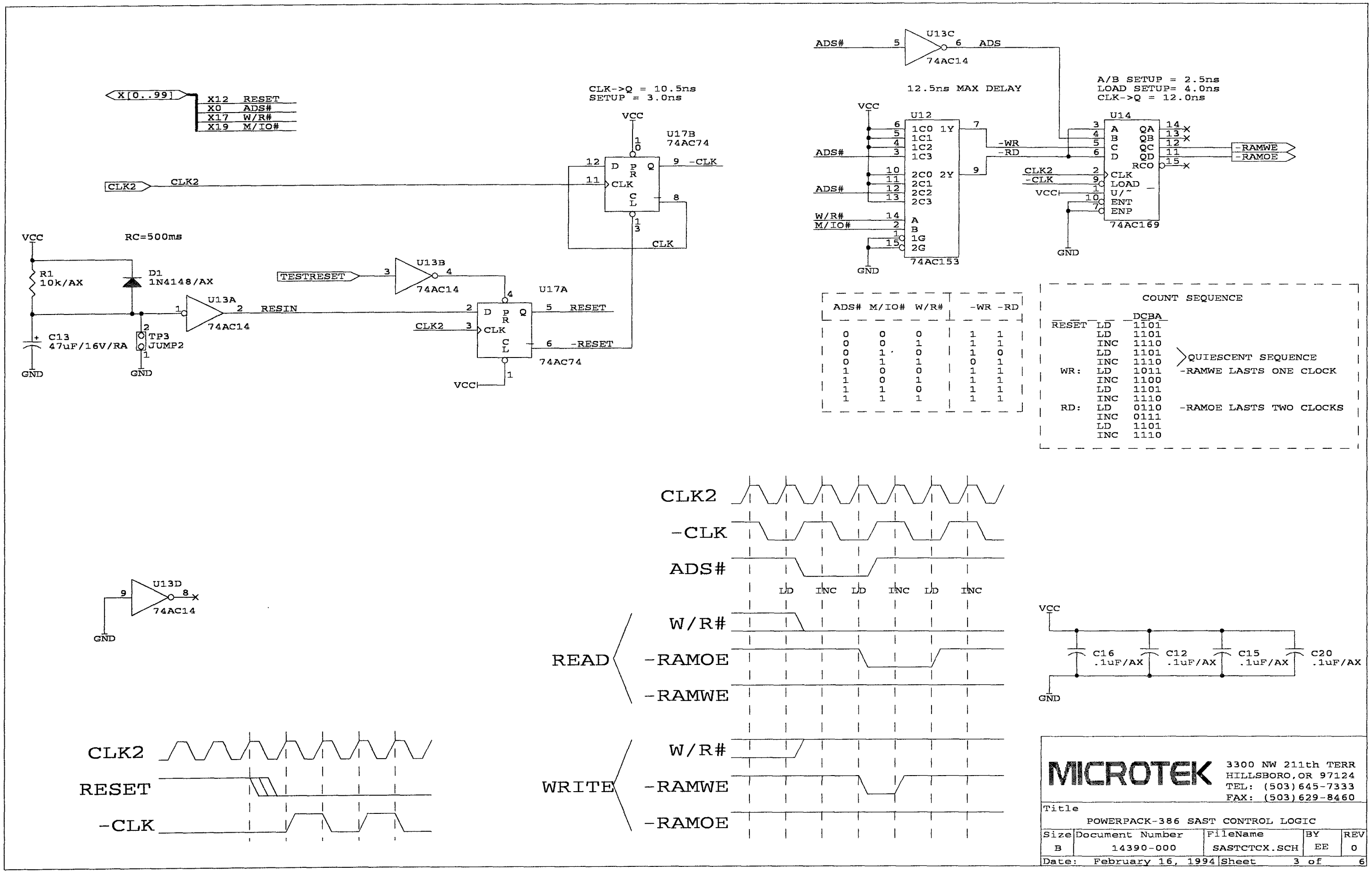
386 CX/SX SAST Board

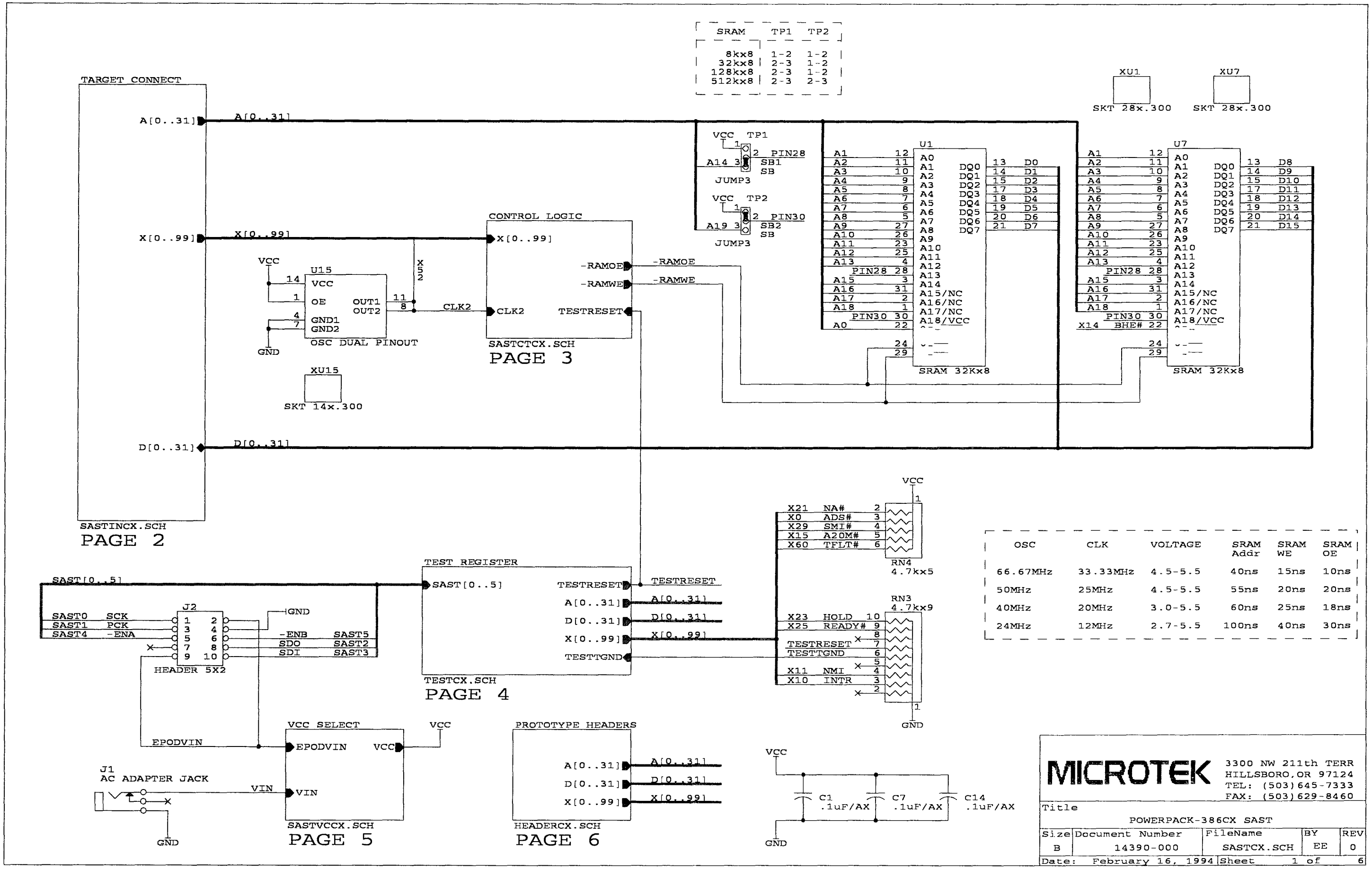
The following pages contain the 386 CX/SX SAST board schematics.



MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460

Title				
POWERPACK-386CX SAST PROTOTYPING HEADERS				
Size	Document Number	FileName	BY	REV
B	14390-000	HEADERCX.SCH	EE	0
Date: February 14, 1994		Sheet . 6 of 6		


X[0..99]
X12 RESET
X0 ADS#
X17 W/R#
X19 M/IO#
CLK->Q = 10.5ns
SETUP = 3.0ns
12.5ns MAX DELAY
A/B SETUP = 2.5ns
LOAD SETUP= 4.0ns
CLK->Q = 12.0ns
U13C 74AC14
U12 74AC153
U14 74AC169
U17B 74AC74
U17A 74AC74
U13A 74AC14
U13B 74AC14
U13D 74AC14
RC=500ms
R1 10k/AX
D1 1N4148/AX
C13 47uF/16V/RA
TP3 JUMP2
TESTRESET
RESIN
RESET
-RESET
CLK2
-CLK
-WR
-RD
-RAMWE
-RAMOE
VCC
GND
ADS# M/IO# W/R# -WR -RD
0 0 0 1 1
0 0 1 1 1
0 1 0 1 0
0 1 1 0 1
1 0 0 1 1
1 0 1 1 1
1 1 0 1 1
1 1 1 1 1
COUNT SEQUENCE
DCBA
RESET LD 1101
LD 1101
INC 1110
LD 1101
INC 1110
QUIESCENT SEQUENCE
WR: LD 1011
INC 1100
LD 1101
INC 1110
-RAMWE LASTS ONE CLOCK
RD: LD 0110
INC 0111
LD 1101
INC 1110
-RAMOE LASTS TWO CLOCKS
CLK2
-CLK
ADS#
LD INC LD INC LD INC
READ
W/R#
-RAMOE
-RAMWE
WRITE
W/R#
-RAMWE
-RAMOE
CLK2
RESET
-CLK
C16 .1uF/AX
C12 .1uF/AX
C15 .1uF/AX
C20 .1uF/AX
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386 SAST CONTROL LOGIC
Size B
Document Number 14390-000
FileName SASTCTCX.SCH
BY EE
REV 0
Date: February 16, 1994
Sheet 3 of 6

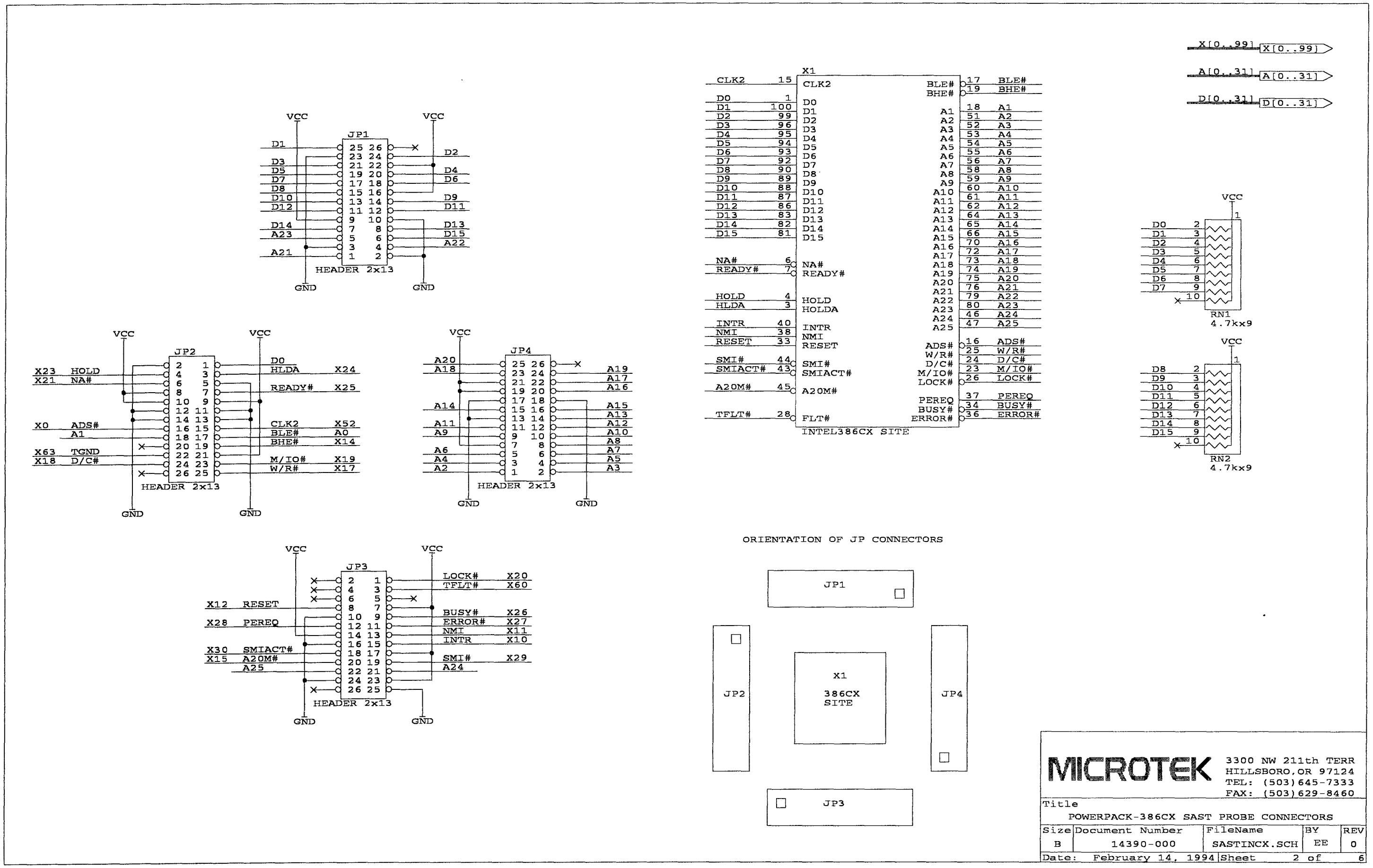


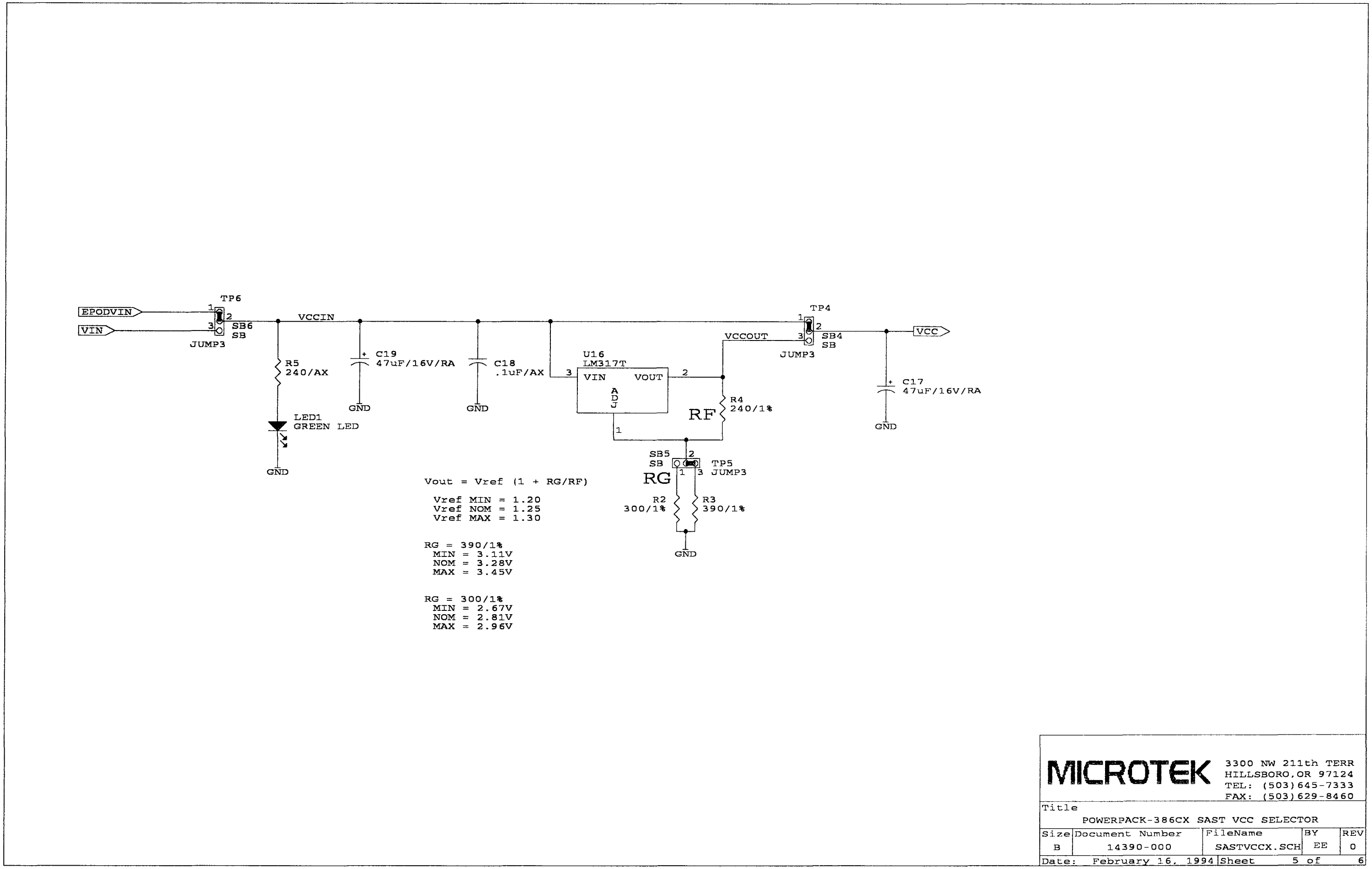
SRAM	TP1	TP2
8kx8	1-2	1-2
32kx8	2-3	1-2
128kx8	2-3	1-2
512kx8	2-3	2-3

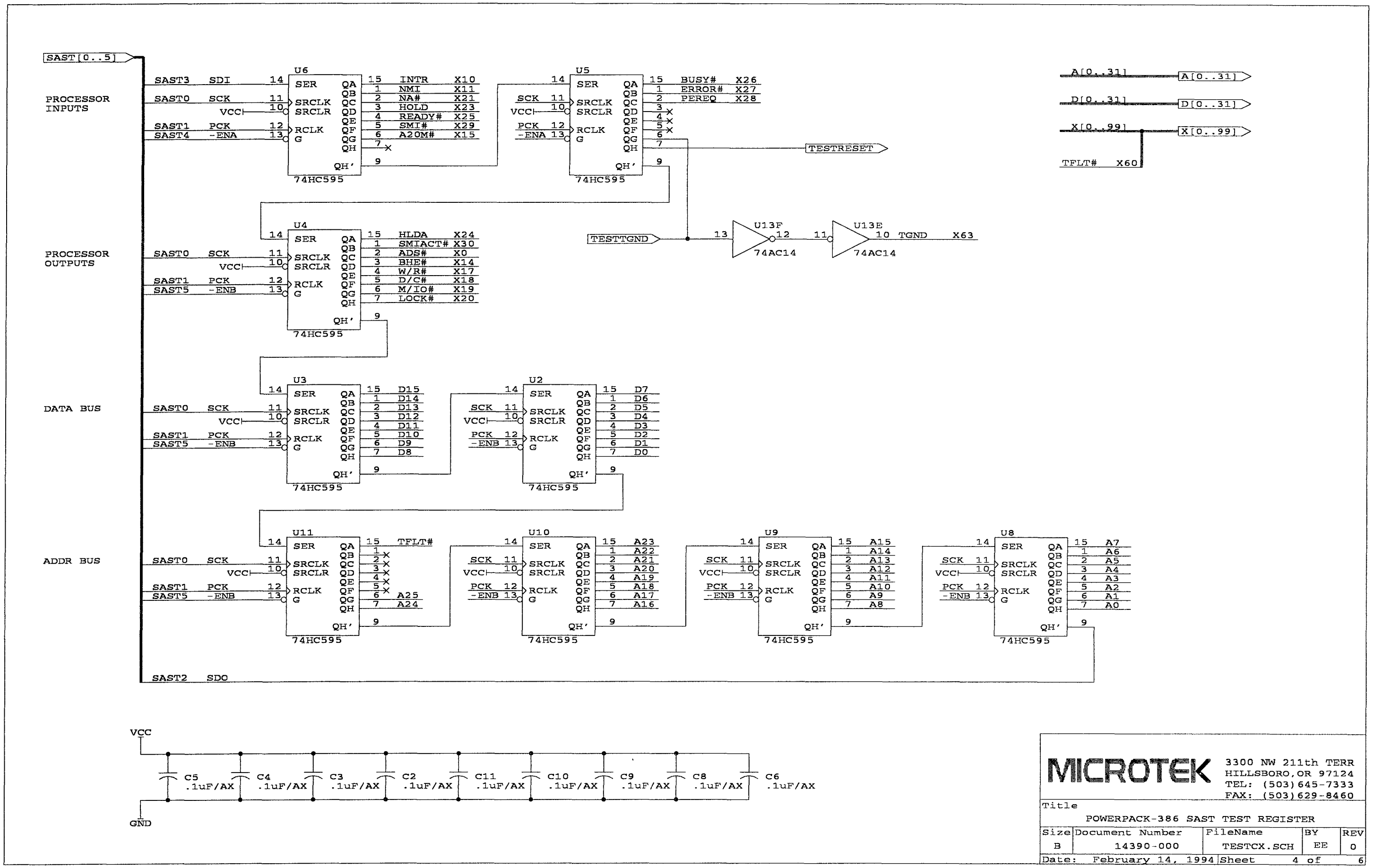
OSC	CLK	VOLTAGE	SRAM Addr	SRAM WE	SRAM OE
66.67MHz	33.33MHz	4.5-5.5	40ns	15ns	10ns
50MHz	25MHz	4.5-5.5	55ns	20ns	20ns
40MHz	20MHz	3.0-5.5	60ns	25ns	18ns
24MHz	12MHz	2.7-5.5	100ns	40ns	30ns

MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645-7333
FAX: (503)629-8460

Title				
POWERPACK-386CX SAST				
Size: B	Document Number: 14390-000	FileName: SASTCX.SCH	BY: EE	REV: 0
Date: February 16, 1994		Sheet 1 of 6		

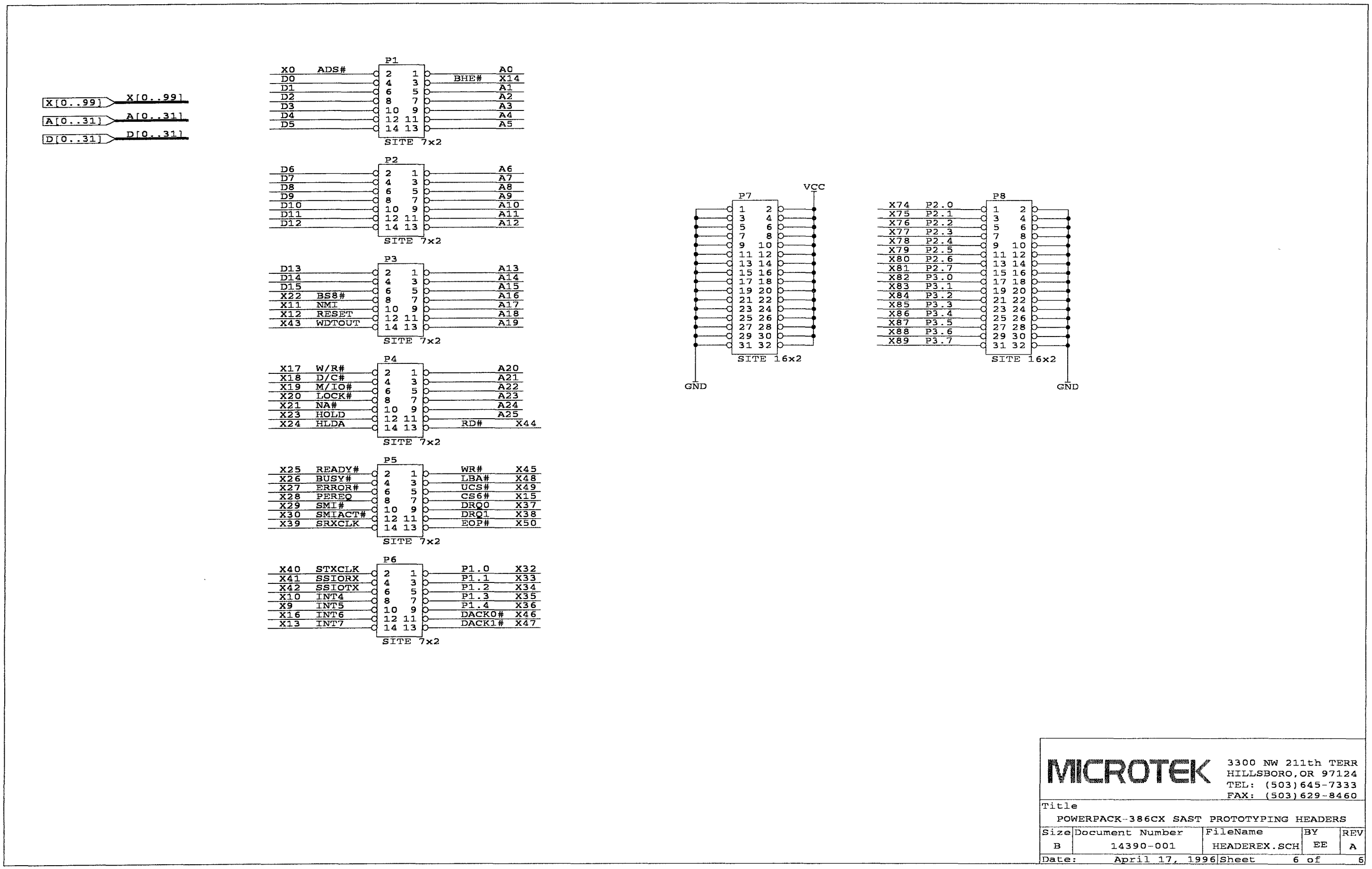

JP1
HEADER 2x13
JP2
HEADER 2x13
JP3
HEADER 2x13
JP4
HEADER 2x13
X1
INTEL386CX SITE
RN1
4.7kx9
RN2
4.7kx9
X[0..99]
A[0..31]
D[0..31]
ORIENTATION OF JP CONNECTORS
386CX SITE
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386CX SAST PROBE CONNECTORS
Size B
Document Number 14390-000
FileName SASTINCX.SCH
BY EE
REV 0
Date: February 14, 1994
Sheet 2 of 6

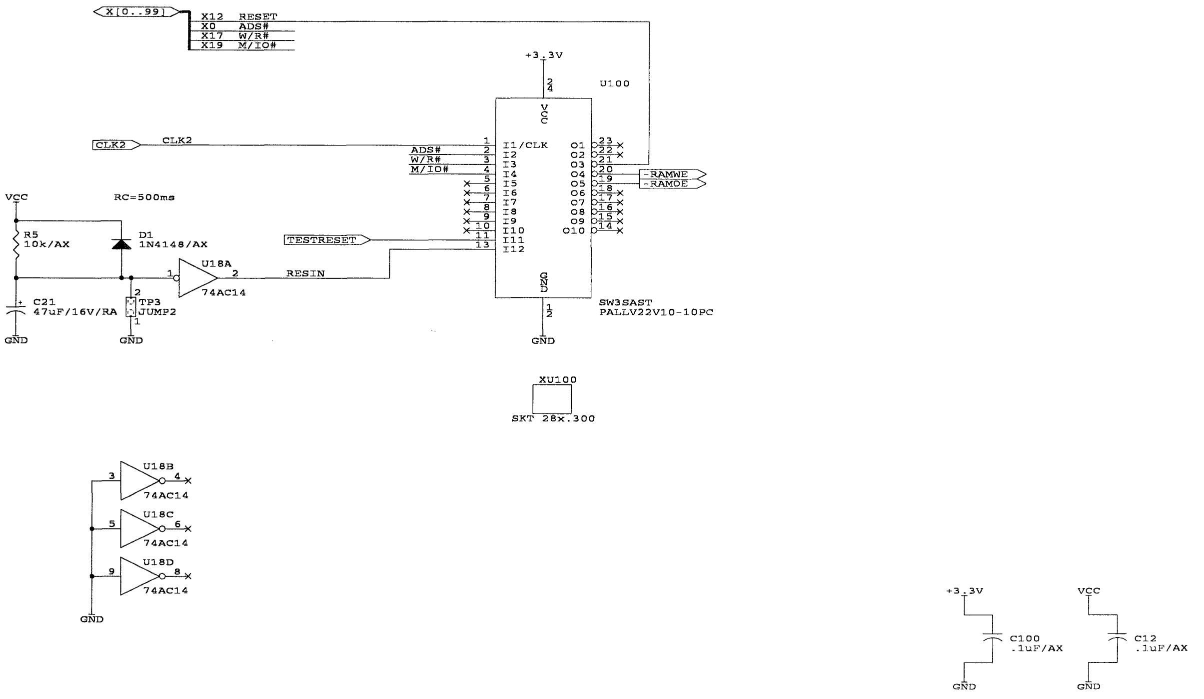

EPODVIN
VIN
TP6
JUMP3
SB6
SB
VCCIN
R5
240/AX
LED1
GREEN LED
GND
C19
47uF/16V/RA
C18
.1uF/AX
U16
LM317T
VIN
VOUT
ADJ
VCCOUT
TP4
SB4
SB
JUMP3
VCC
C17
47uF/16V/RA
R4
240/1%
RF
SB5
SB
TP5
JUMP3
RG
R2
300/1%
R3
390/1%
Vout = Vref (1 + RG/RF)
Vref MIN = 1.20
Vref NOM = 1.25
Vref MAX = 1.30
RG = 390/1%
MIN = 3.11V
NOM = 3.28V
MAX = 3.45V
RG = 300/1%
MIN = 2.67V
NOM = 2.81V
MAX = 2.96V
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386CX SAST VCC SELECTOR
Size B
Document Number 14390-000
FileName SASTVCCX.SCH
BY EE
REV 0
Date: February 16, 1994
Sheet 5 of 6


SAST[0..5]
PROCESSOR INPUTS
PROCESSOR OUTPUTS
DATA BUS
ADDR BUS
SAST3 SDI
SAST0 SCK
SAST1 PCK
SAST4 -ENA
SAST5 -ENB
SAST2 SDO
U6 74HC595
INTR X10
NMI X11
NA# X21
HOLD X23
READY# X25
SMI# X29
A20M# X15
U5 74HC595
BUSY# X26
ERROR# X27
PEREQ X28
TESTRESET
TESTTGND
U13F 74AC14
U13E 74AC14
TGND X63
U4 74HC595
HLDA X24
SMIACT# X30
ADS# X0
BHE# X14
W/R# X17
D/C# X18
M/IO# X19
LOCK# X20
U3 74HC595
U2 74HC595
U11 74HC595
TFLT#
A25
A24
U10 74HC595
U9 74HC595
U8 74HC595
A[0..31]
D[0..31]
X[0..99]
TFLT# X60
VCC
GND
C5 C4 C3 C2 C11 C10 C9 C8 C6
.1uF/AX
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386 SAST TEST REGISTER
Size B
Document Number 14390-000
FileName TESTCX.SCH
BY EE
REV 0
Date: February 14, 1994
Sheet 4 of 6

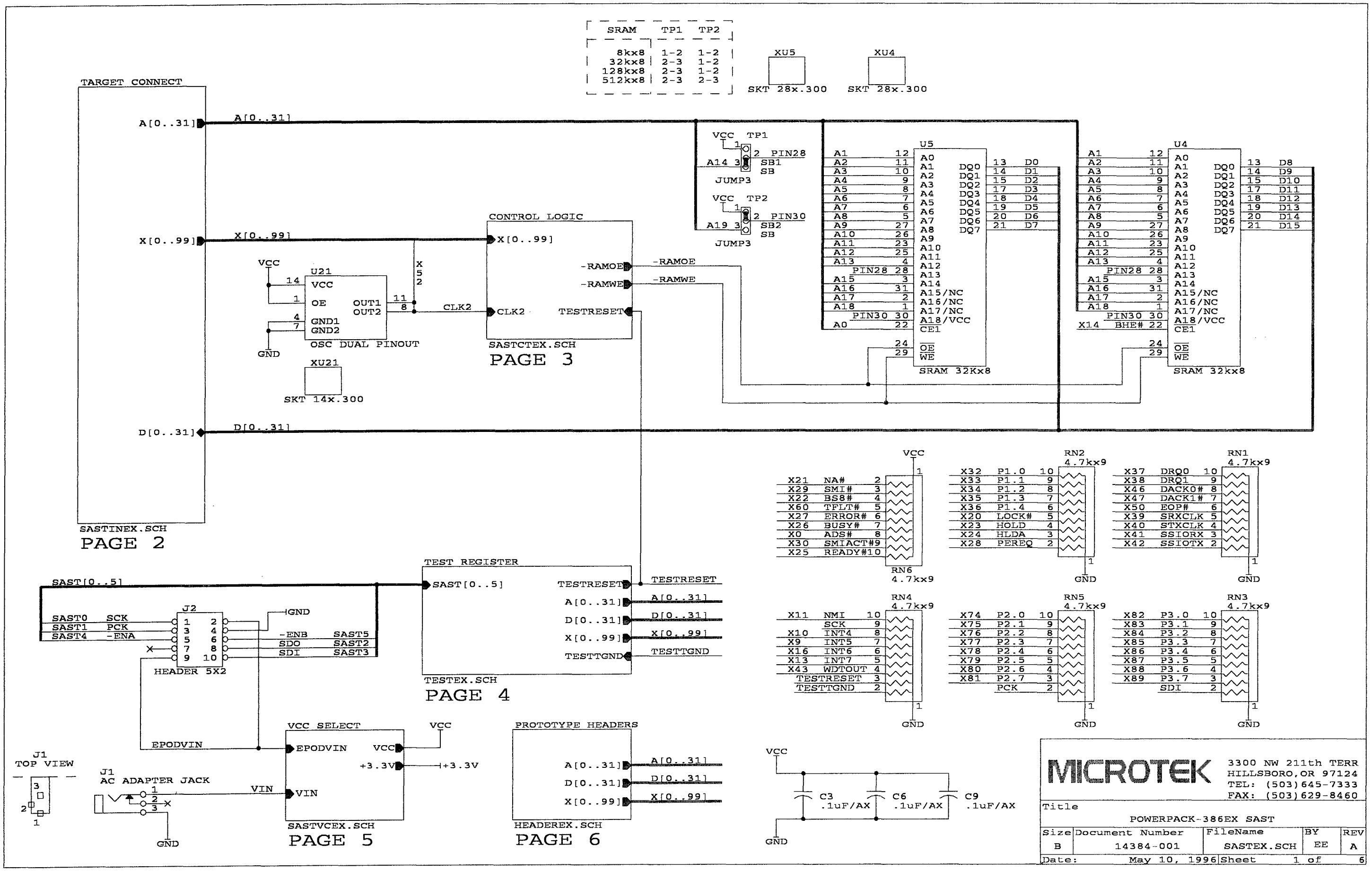
386 EX SAST Board

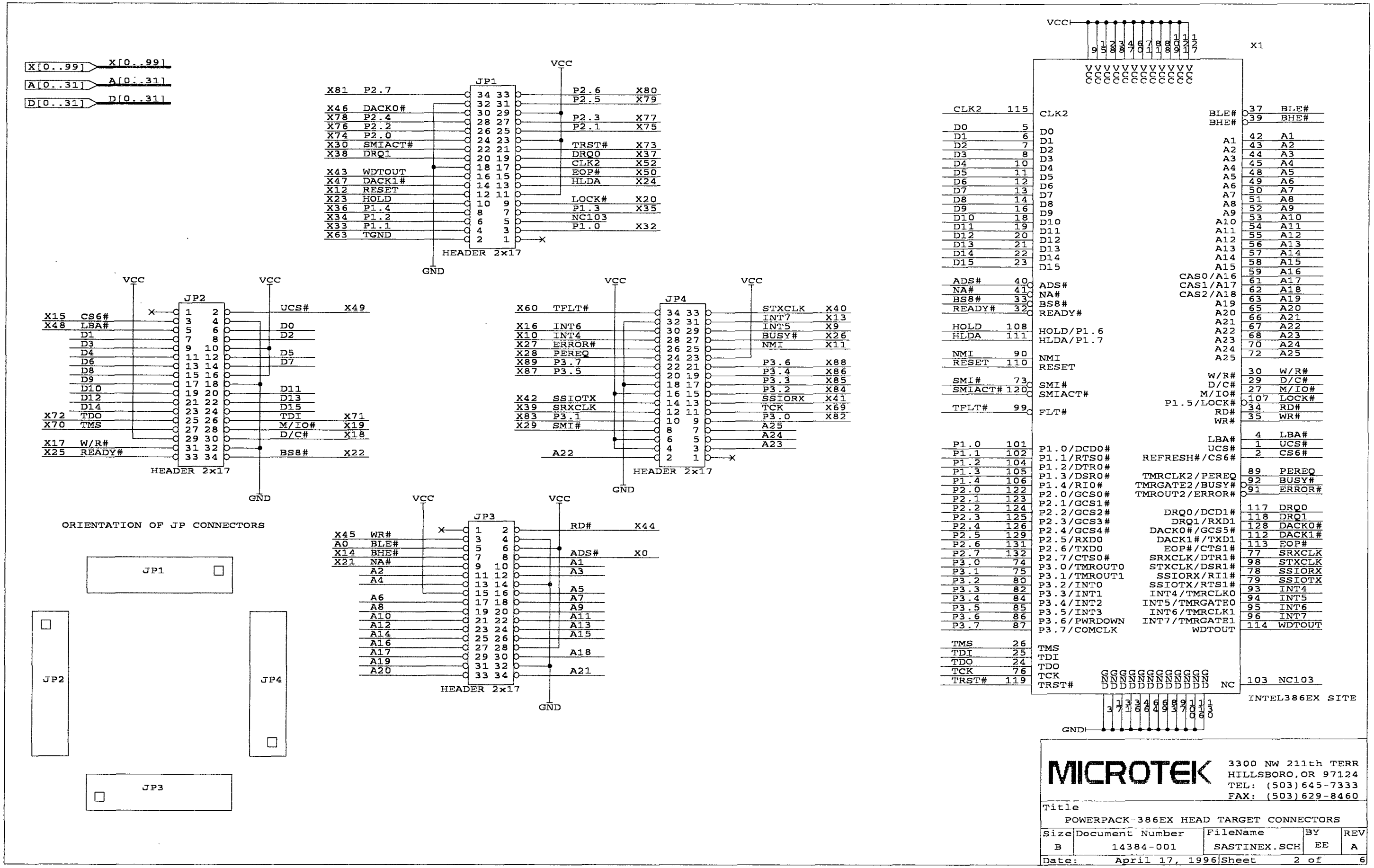
The following pages contain the 386 EX SAST board schematics.

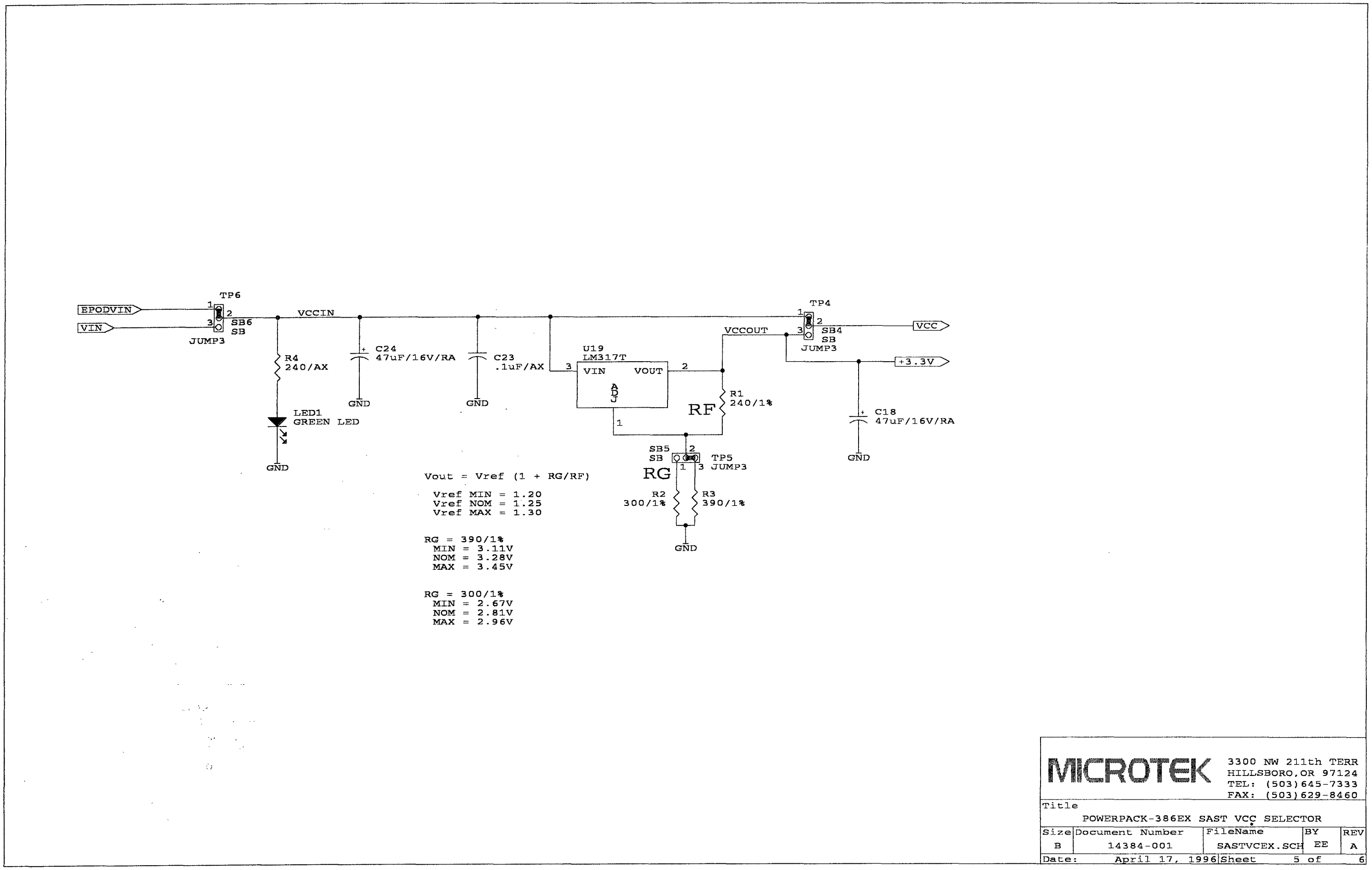

X[0..99]
A[0..31]
D[0..31]
P1
X0 ADS#
D0
D1
D2
D3
D4
D5
A0
BHE# X14
A1
A2
A3
A4
A5
SITE 7x2
P2
D6
D7
D8
D9
D10
D11
D12
A6
A7
A8
A9
A10
A11
A12
SITE 7x2
P3
D13
D14
D15
X22 BS8#
X11 NMI
X12 RESET
X43 WDTOUT
A13
A14
A15
A16
A17
A18
A19
SITE 7x2
P4
X17 W/R#
X18 D/C#
X19 M/IO#
X20 LOCK#
X21 NA#
X23 HOLD
X24 HLDA
A20
A21
A22
A23
A24
A25
RD# X44
SITE 7x2
P5
X25 READY#
X26 BUSY#
X27 ERROR#
X28 PEREQ
X29 SMI#
X30 SMIACT#
X39 SRXCLK
WR# X45
LBA# X48
UCS# X49
CS6# X15
DRQ0 X37
DRQ1 X38
EOP# X50
SITE 7x2
P6
X40 STXCLK
X41 SSIORX
X42 SSIOTX
X10 INT4
X9 INT5
X16 INT6
X13 INT7
P1.0 X32
P1.1 X33
P1.2 X34
P1.3 X35
P1.4 X36
DACK0# X46
DACK1# X47
SITE 7x2
P7
VCC
SITE 16x2
GND
P8
X74 P2.0
X75 P2.1
X76 P2.2
X77 P2.3
X78 P2.4
X79 P2.5
X80 P2.6
X81 P2.7
X82 P3.0
X83 P3.1
X84 P3.2
X85 P3.3
X86 P3.4
X87 P3.5
X88 P3.6
X89 P3.7
SITE 16x2
GND
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386CX SAST PROTOTYPING HEADERS
Size B
Document Number 14390-001
FileName HEADEREX.SCH
BY EE
REV A
Date: April 17, 1996
Sheet 6 of 6

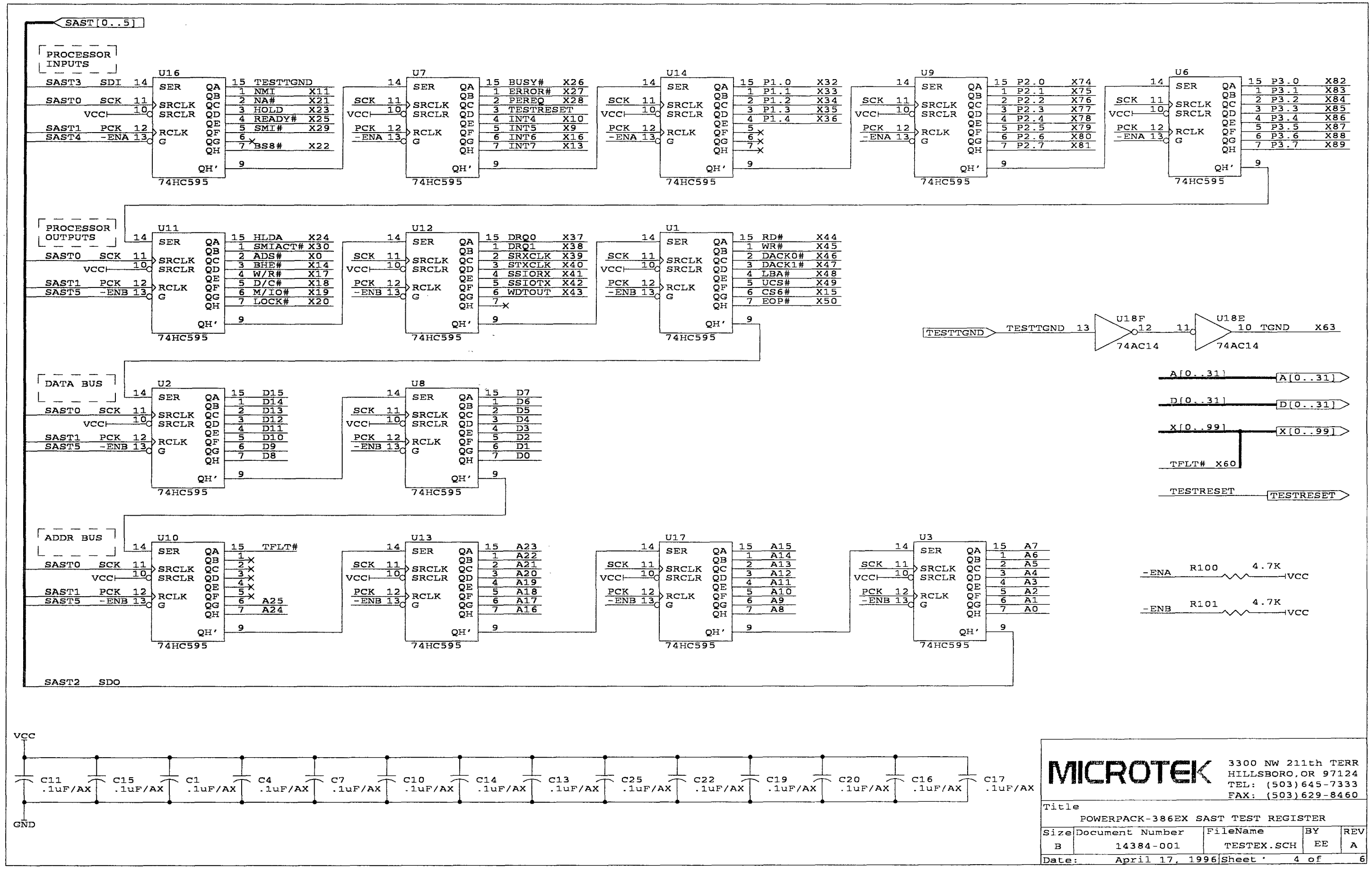


MICROTEK	3300 NW 211th TERR HILLSBORO, OR 97124 TEL: (503)645-7333 FAX: (503)629-8460			
Title POWERPACK-386EX SAST CONTROL LOGIC				
Size B	Document Number 14384-001	FileName SASTCTEX.SCH	BY EE	REV A
Date: April 17, 1996		Sheet 3 of 6		


SRAM TP1 TP2
8kx8 1-2 1-2
32kx8 2-3 1-2
128kx8 2-3 1-2
512kx8 2-3 2-3
XU5 SKT 28x.300
XU4 SKT 28x.300
TARGET CONNECT
A[0..31]
X[0..99]
D[0..31]
SASTINEX.SCH
PAGE 2
CONTROL LOGIC
-RAMOE
-RAMWE
TESTRESET
CLK2
SASTCTEX.SCH
PAGE 3
U21
OSC DUAL PINOUT
XU21 SKT 14x.300
JUMP3
U5 SRAM 32Kx8
U4 SRAM 32kx8
TEST REGISTER
SAST[0..5]
TESTTGND
TESTEX.SCH
PAGE 4
J2 HEADER 5X2
VCC SELECT
EPODVIN
VIN
+3.3V
SASTVCEX.SCH
PAGE 5
PROTOTYPE HEADERS
HEADEREX.SCH
PAGE 6
J1 TOP VIEW
J1 AC ADAPTER JACK
C3 .1uF/AX
C6 .1uF/AX
C9 .1uF/AX
RN1 4.7kx9
RN2 4.7kx9
RN3 4.7kx9
RN4 4.7kx9
RN5 4.7kx9
RN6 4.7kx9
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386EX SAST
Size B
Document Number 14384-001
FileName SASTEX.SCH
BY EE
REV A
Date: May 10, 1996
Sheet 1 of 6


JP1
HEADER 2x17
JP2
JP3
JP4
VCC
GND
ORIENTATION OF JP CONNECTORS
X1
INTEL386EX SITE
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386EX HEAD TARGET CONNECTORS
Size B
Document Number 14384-001
FileName SASTINEX.SCH
BY EE
REV A
Date: April 17, 1996
Sheet 2 of 6


EPODVIN
VIN
TP6
SB6
SB
JUMP3
VCCIN
R4
240/AX
LED1
GREEN LED
GND
C24
47uF/16V/RA
C23
.1uF/AX
U19
LM317T
VIN
VOUT
ADJ
VCCOUT
R1
240/1%
RF
SB5
SB
TP5
JUMP3
RG
R2
300/1%
R3
390/1%
TP4
SB4
SB
JUMP3
VCC
+3.3V
C18
47uF/16V/RA
Vout = Vref (1 + RG/RF)
Vref MIN = 1.20
Vref NOM = 1.25
Vref MAX = 1.30
RG = 390/1%
MIN = 3.11V
NOM = 3.28V
MAX = 3.45V
RG = 300/1%
MIN = 2.67V
NOM = 2.81V
MAX = 2.96V
MICROTEK
3300 NW 211th TERR
HILLSBORO,OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386EX SAST VCC SELECTOR
Size B
Document Number 14384-001
FileName SASTVCEX.SCH
BY EE
REV A
Date: April 17, 1996
Sheet 5 of 6


SAST[0..5]
PROCESSOR INPUTS
PROCESSOR OUTPUTS
DATA BUS
ADDR BUS
U16
U7
U14
U9
U6
U11
U12
U1
U2
U8
U10
U13
U17
U3
74HC595
SER
SRCLK
SRCLR
RCLK
G
QA
QB
QC
QD
QE
QF
QG
QH
QH'
SAST3 SDI
SAST0 SCK
SAST1 PCK
SAST4 -ENA
SAST5 -ENB
SAST2 SDO
TESTTGND
U18F
U18E
74AC14
10 TGND X63
A[0..31]
D[0..31]
X[0..99]
TFLT# X60
TESTRESET
-ENA R100 4.7K VCC
-ENB R101 4.7K VCC
VCC
GND
C11 .1uF/AX
C15 .1uF/AX
C1 .1uF/AX
C4 .1uF/AX
C7 .1uF/AX
C10 .1uF/AX
C14 .1uF/AX
C13 .1uF/AX
C25 .1uF/AX
C22 .1uF/AX
C19 .1uF/AX
C20 .1uF/AX
C16 .1uF/AX
C17 .1uF/AX
MICROTEK
3300 NW 211th TERR
HILLSBORO, OR 97124
TEL: (503)645-7333
FAX: (503)629-8460
Title
POWERPACK-386EX SAST TEST REGISTER
Size B
Document Number 14384-001
FileName TESTEX.SCH
BY EE
REV A
Date: April 17, 1996
Sheet 4 of 6

Index

U

V

W

Y

MICROTEK INTERNATIONAL
Development Systems Division
3300 N.W. 211th Terrace
Hillsboro, OR 97124-7136
Phone: (503) 645-7333
Fax: (503) 629-8460

PowerPack® EA/SW In-Circuit Emulator
Hardware Reference
(for Intel Target Processors)
Part Number 14867-002