ADSP-21061 EZ-KIT Lite™ Reference Manual

Part Number: ADDS-21061-EZLITE
Revision 2.0
May 2001

ANALOG DEVICES

Notice

Analog Devices, Inc. reserves the right to make changes to or to discontinue any product or service identified in this publication without notice.

Analog Devices assumes no liability for Analog Devices applications assistance, customer product design, customer software performance, or infringement of patents or services described herein. In addition, Analog Devices shall not be held liable for special, collateral, incidental or consequential damages in connection with or arising out of the furnishing, performance, or use of this product.

Analog Devices products are not intended for use in life-support applications, devices, or systems. Use of an Analog Devices product in such applications without the written consent of the Analog Devices officer is prohibited. Users are restricted from copying, modifying, distributing, reverse engineering, and reverse assembling or reverse compiling the ADSP-21061 EZ-KIT Lite operational software
(one copy may be made for back-up purposes only). No part of this document may be reproduced in any form without permission.

Trademarks and Service Mark Notice

The Analog Devices logo, ADSP-21061 SHARC, the ADSP-21061 SHARC logo, JTAG, and EZ-ICE are registered trademarks; and EZ-KIT Lite, VisualDSP++, the VisualDSP++ logo, TigerSHARC, the TigerSHARC logo, and the DSP Collaborative are trademarks of Analog Devices, Inc. Microsoft and Windows are registered trademarks and Windows NT is a trademark of Microsoft Corporation. Adobe and Acrobat are trademarks of Adobe Systems Incorporated. All other brands and product names are trademarks or service marks of their respective owners.

Limited Warranty

The ADSP-21061 EZ-KIT Lite hardware is warranted against defects in materials and workmanship for a period of one year from the date of purchase from Analog Devices or from an authorized dealer.

Revision 2.0, May 2001 ADDS-21061-EZLITE

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1. Introduction

Thank you for purchasing the ADSP-21061 EZ-KIT Lite™ evaluation kit. The evaluation board is designed to be used in conjunction with the VisualDSP++™ development environment and is based on the ADSP-21061 SHARC ® floating-point digital signal processor (DSP). The kit is shipped with an evaluation board and VisualDSP++ software. The VisualDSP++ that comes with the kit will only operate with the evaluation board. The complete version must be purchased separately. Using the EZ-KIT Lite with VisualDSP++, you can observe the ADSP-21061 DSP execute programs from on-chip RAM, interact with on-board devices, and communicate with other peripherals.

You can access the ADSP-21061 SHARC processor using a PC through a serial port or an optional JTAG® In-Circuit Emulator (ICE). The monitor program that runs on the EZ-KIT Lite gives you access to the ADSP-21061 SHARC processor's internal memory space through the serial port. By contrast, the JTAG emulator allows the PC to perform in-circuit emulation through the processor's JTAG interface. The board's features include:

- Analog Devices ADSP-21061 DSP running at 40 MHz
- Analog Devices AD1847 16-bit Stereo SoundPort® codec
- RS-232 interface
- Socketed EPROM
- User pushbuttons
- User programmable LEDs
- Power supply regulation
- Expansion connectors

The EZ-KIT Lite board is equipped with hardware that facilitates interactive demonstrations. The pushbutton switches and user programmable LEDs provide user control and board status. Additionally, the AD1847 SoundPort codec provides access to an audio input (selectable as line level or microphone) and an audio output (line level).

The board can run stand-alone or can connect to the RS-232 port of the PC. The EZ-KIT Lite includes a monitor program stored on the original boot EPROM. The monitor program lets you download, execute, and debug ADSP-21061 EZ-KIT Lite programs. By removing the socketed EPROM, and replacing it with an EPROM containing code, the board can run as a stand-alone unit.

You can also connect an optional JTAG emulator to the ADSP-21061 EZ-KIT Lite. The emulator allows you to load programs, start and stop program execution, observe and alter registers and memory, and perform other debugging operations.

JTAG emulators are available from Analog Devices and other third-party resellers.

1.2. EZ-KIT Lite System Architecture

Figure 1-1 is a block diagram of the ADSP-21061 EZ-KIT Lite system. You can access the ADSP-21061 SHARC processor from the PC through the RS-232 interface. The boot PROM provides when the ADSP-21061 EZ-KIT Lite is operating in stand-alone mode and loads a kernel that manages the RS-232 interface.

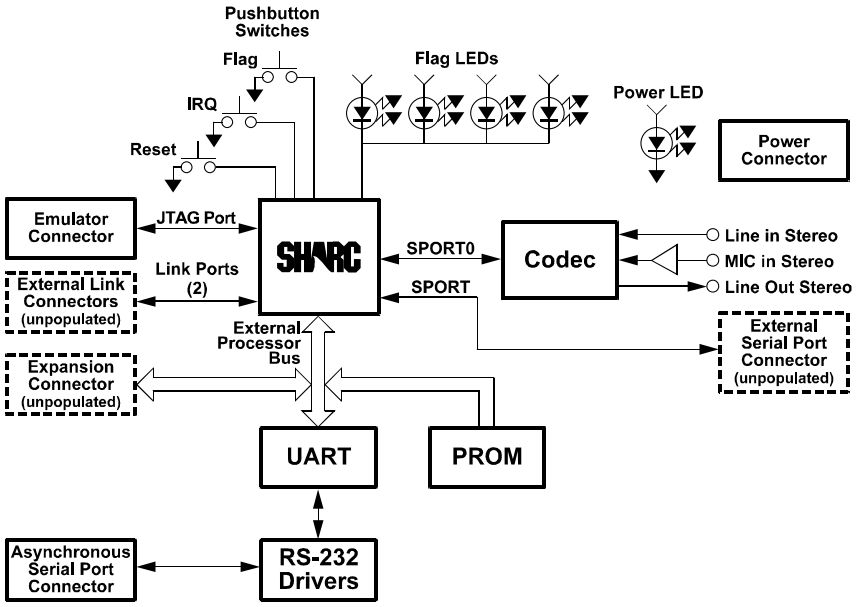


Figure 1-1 ADSP-21061 EZ-KIT Lite System Block Diagram

In-circuit emulation is achieved with a JTAG probe connected to the JTAG port. The AD1847 SoundPort Stereo codec is accessed through a serial port that connects directly to the ADSP-21061 SHARC processor. The ADSP-21061 EZ-KIT Lite board has several sites for connectors, allowing you to expand the board's capabilities. These sites are not populated with connectors when you receive the board from Analog Devices.

1.3. For More Information About Analog Devices, Inc. Products

Analog Devices is accessible on the Internet at www.analog.com. The DSP web page is directly accessible at www.analog.com/dsp. This page provides access to DSP-specific technical information and documentation, product overviews, and product announcements.

1.4. For Technical or Customer Support

You can reach our Customer Support group in the following ways:

- Fill in the online support request form at:
  http://www.analog.com/industry/dsp/tools/form_techsupport.html

- Email questions to dsptools.support@analog.com

1.5. Purpose of this Manual

This manual shows how to install the evaluation board and software on the PC. Also, the manual provides guidelines for running user code on the ADSP-21061 SHARC processor.

1.6. Intended Audience

DSP programmers who are familiar with Analog Devices' DSPs are the primary audience for this manual. This manual assumes that the audience has a working knowledge of Analog Devices DSP architecture and DSP instruction set.

DSP programmers who are unfamiliar with Analog Devices DSPs can use this manual, but should supplement this manual with the *ADSP-2106x User's Manual* and the VisualDSP++ tools manuals.

1.7. Manual Contents Description

This manual contains the following information:

- Chapter 1 – Introduction

  Provides manual overview and Analog Devices contact information.

- Chapter 2 – Getting Started

  Provides information needed to install the software and the ADSP-21061 EZ-KIT Lite evaluation board.

- Chapter 3 – Using EZ-KIT Lite Software

  Provides monitor level software information on how the EZ-KIT Lite board operates with the installed software.

- Chapter 4 – Demonstration Programs

  Describes loading and running the demonstration programs supplied with the ADSP-21061 EZ-KIT Lite board.

- Chapter 5 – Working with EZ-KIT Lite Hardware

  Describes the hardware characteristics of the ADSP-21061 EZ-KIT Lite board.

- Chapter 6 – Programming Reference

  Provides technical details needed when writing programs for the ADSP-21061 EZ-KIT Lite.

- Chapter 7 – Reference

  This is a reference for VisualDSP++ and provides information on all of the menu selections, commands, and dialog boxes when the target is the ADSP-21061 EZ-KIT Lite evaluation board.

- Chapter 8 – Schematics

  Provides design information used to build the ADSP-21061 EZ-KIT Lite board.

- APPENDIX – Restrictions

  Provides restrictions that apply to release 2.0 of the ADSP-21061 EZ-KIT Lite board.

1.8. Documentation and Related Products

For more information on the ADSP-21061 SHARC processor and the components of the ADSP-21061 EZ-KIT Lite system, see the following documents:

- *ADSP-21061 SHARC Processor Data Sheet*
- *ADSP-2106x SHARC User's Manual*
- *AD1847 Serial Port 16-Bit SoundPort Stereo Codec Data Sheet*
- *PC16550D Universal Asynchronous Receiver/Transmitter with FIFOs Data Sheet (National Semiconductor)*

The ADSP-21061 family of processors is supported by a complete set of evaluation tools. Software tools include the C/C++ compiler, assembler, run-time libraries, linker, loader, simulator, and PROM splitter. See the following documents:

- *VisualDSP++ Getting Started Guide*
- *VisualDSP++ User's Guide for the ADSP-21xxx Family DSPs*
- *Assembler Manual for the ADSP-21xxx Family DSPs*
- *C/C++ Compiler & Library Manual for the ADSP-21xxx Family DSPs*
- *Linker & Utilities for the ADSP-21xxx Family DSPs*
- *Product Bulletin for the VisualDSP++ and the ADSP-21xxx Family DSPs*

These documents are found on the Analog Devices Technical Documentation web site at: http://www.analog.com/industry/dsp/tech_doc/gen_purpose.html#32

If you plan to use the EZ-KIT Lite with the JTAG emulator, refer to the documentation that accompanies the emulator.

Your software installation kit includes online Help as part of the Windows interface. This Help file provides information about the ADSP-21061 EZ-KIT Lite evaluation board and accompanying tools.

2. GETTING STARTED

2.1. Overview

This chapter provides you with the information you need to install your software and the ADSP-21061 EZ-KIT Lite evaluation board. Install your software and hardware in the order presented for correct operation. This chapter also provides basic board information.

2.2. Contents of Your EZ-KIT Lite Package

Your ADSP-21061 EZ-KIT Lite should contain the following items. If any item is missing, contact the vendor from whom you purchased your EZ-KIT Lite.

The evaluation board contains ESD (electrostatic discharge) sensitive devices. Electrostatic charges readily accumulate on the human body and equipment and can discharge without detection. Permanent damage may occur on devices subjected to high-energy discharges. Proper ESD precautions are recommended to avoid performance degradation or loss of functionality. Unused boards should be stored in the protective shipping package.	

- ADSP-21061 EZ-KIT Lite board
- Power cable with 9V DC power supply
- 9-pin RS-232 serial port cable
- CD-ROM containing EZ-KIT Lite target .dll files, examples, Help file, and utilities
- CD-ROM with VisualDSP++
- CD-ROM (DSP Designer's Reference) containing DSP documentation

2.3. PC Configuration

For correct operation of the VisualDSP++ software and EZ-KIT Lite demos, your computer must have the minimum configuration shown below.

Table 2-1 PC Minimum Configuration

Windows 95, 98, 2000	Windows NT
Windows 95b, Windows 98, or Windows 2000	Windows NT, release 4.0, Service Pack 3 or later
Pentium processor 166 MHz or faster	Pentium processor 166 MHz or faster
VGA Monitor and color video card	VGA Monitor and color video card
2-button mouse	2-button mouse
100 MB available space	100 MB available space
32 MB RAM	32 MB RAM
CD-ROM	CD-ROM

2.4. VisualDSP++

ADSP-21061 EZ-KIT Lite is shipped with the VisualDSP++ Integrated Development Environment (IDE), debugger, and code generation tools. VisualDSP++ is limited in functionality by the EZ-KIT Lite serial number shipped with this product. The EZ-KIT Lite serial number restricts the VisualDSP++ debugger to connect only to the ADSP-21061 EZ-KIT Lite evaluation board running the debug monitor via the serial port (no emulation or simulation support). Additionally, the linker restricts you to only 25% (4k words) of the ADSP-21061 SHARC processor's on-chip program memory space. If you purchase the full VisualDSP++ software suite you will obtain a new serial number from Analog Devices to lift the restrictions mentioned above.

The basic components that are shipped with VisualDSP++ are:

Integrated Development Environment (IDE) — graphical interface for project management, allowing you to set project options and access the code generation tools.

Debugger — allows you to view the insides of the DSP and perform debug operations such as read/write memory, read/write registers, load programs, run, step, halt, and more.

SHARC Family Code Generation Tools — C/C++ compiler, assembler, run-time libraries and linker, loader, simulator, and PROM splitter.

Example Projects — Both VisualDSP++ and the ADSP-21061 EZ-KIT Lite are shipped with example projects and C/C++ and Assembly source code that demonstrate various features of the tools and the ADSP-21061 DSP.

2.5. Installation Procedures

The following procedures are provided for the safe and effective use of the ADSP-21061 EZ-KIT Lite evaluation board. Follow these instructions in the order presented to ensure correct operation of your software and hardware.

2.5.1. Installing the EZ-KIT Lite Hardware

The ADSP-21061 EZ-KIT Lite board is designed to run outside the PC as a stand-alone unit. There is no need to remove the chassis from your computer. To connect the EZ-KIT Lite board:

1. Remove the EZ-KIT Lite board from the package. Be careful when handling the board to avoid the discharge of static electricity, which may damage some components.
2. Connect the RS-232 cable to an available COM Port on the PC and the ADSP-21061 EZ-KIT Lite evaluation board.
3. Plug the provided 9V-power supply into a 120-Volt AC receptacle and plug the connector at the other end of the cable into P2 on the evaluation board.

The FLAG3 and FLAG2 LEDs will start to toggle. The POWER LED remains on.
If the LEDs do not light up, check the power connections. If you plug a pair of self-powered computer speakers into the Jack J22 (Output) on the board you will be able to hear a tune.

To configure your board to take advantage of the audio capabilities of the demos:

1. Plug a set of self-powered computer speakers into Jack J22 on the board. Turn on the speakers and set the volume to an adequate level.
2. Connect the line out of an electronic audio device to Jack J23 (Input) on the board. Set jumpers JP6 and JP8 to LINE.

This completes the hardware installation.

2.5.2. Installing VisualDSP++

The EZ-KIT Lite includes the latest evaluation version of VisualDSP++ for the ADSP-21061 SHARC DSP Family. You must install this software prior to installing the EZ-KIT Lite software.

Insert the VisualDSP++ CD-ROM into the CD-ROM drive. This will bring up the CD browser.

Click on the “Install VisualDSP++” option. This will launch the setup wizard.

Follow this wizard with the on-screen instructions.

2.5.3. Installing the EZ-KIT Lite License

Before the VisualDSP++ software can be used, you must install the license software. To install the EZ-KIT Lite license software follow these steps:

1. VisualDSP++ has been installed.

2. Insert the VisualDSP++ CD-ROM into the CD-ROM drive if it is not already in the drive.

3. Once the browser appears, select the "Install License" option.

4. Follow the setup wizard instructions.

NOTE: Ensure that you have the proper serial number, which is located on the back of the CD sleeve.

2.5.4. Installing the EZ-KIT Lite Software

The EZ-KIT Lite utility software is supplied on a separate CD-ROM.

To install the EZ-KIT Lite software:

1. Ensure VisualDSP++ has been installed.

2. Close VisualDSP++ and all Windows applications. The install will not work correctly if any VisualDSP++ applications are running.

3. Insert the EZ-KIT Lite CD into the CD-ROM drive. The setup will automatically start. Follow the installation wizard by choosing the appropriate options.

4. When the setup has completed, reboot the machine, if necessary.

2.5.4.1. Default Settings

After you have installed the board and utility software, your PC and EZ-KIT Lite have the default settings shown in Table 2-2. You can change these settings through the **Settings** menu in the debugger.

Table 2-2 User Configurable EZ-KIT Lite Settings

Selection	Default Setting
COM Port	COM 1
Baud Rate	115200

The VisualDSP++ debugger comes with a complete on-line help file and Adobe .pdf files of all manuals.

- You can use the context help button  to get help on any command or icon

Or

- Highlight a command and press F1.

For help on commands and dialog boxes click from the toolbar **Help -> Help Topics** to get to the *Debugger Help* file.

2.6. Hardware Connections

Figure 2-1 highlights the external hardware connections to the ADSP-21061 EZ-KIT Lite. The following sections describe each of the connections.

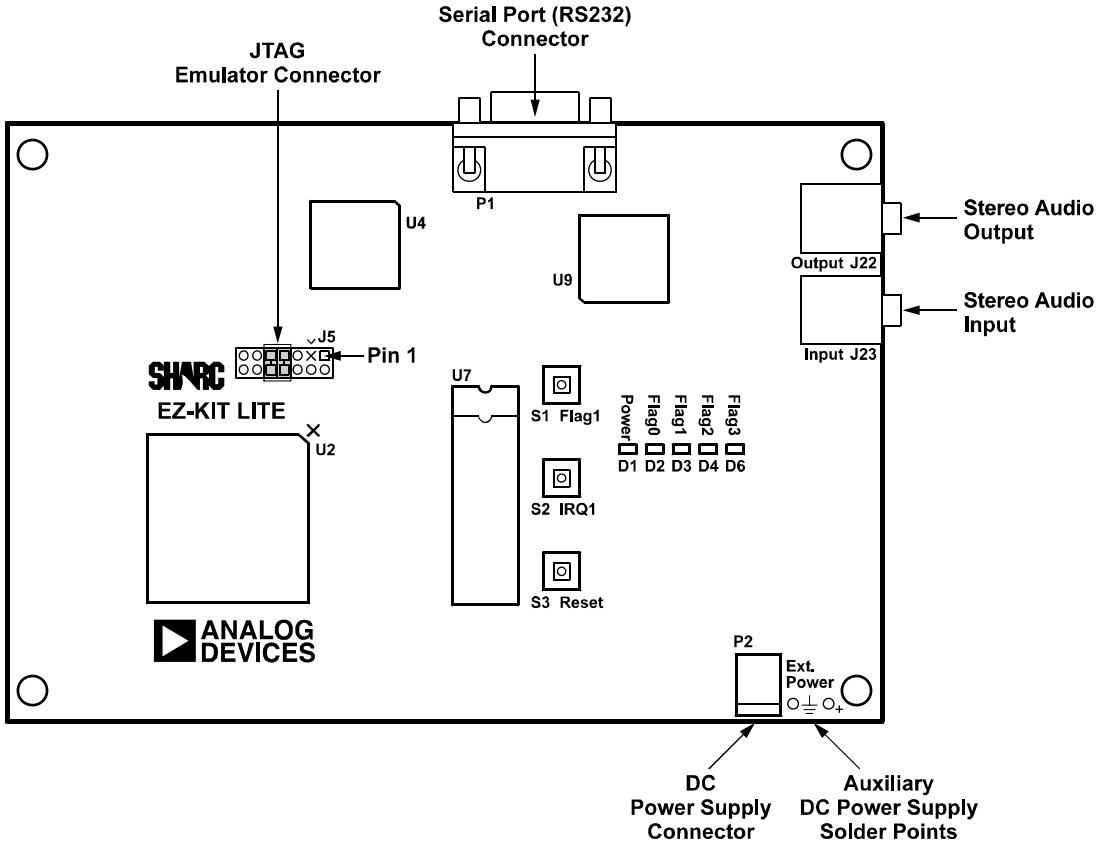


Figure 2-1 Hardware Connections to the ADSP-21061 EZ-KIT Lite

2.6.1. Serial Port (RS-232) Connector

P1 is female 9-pin D-Sub connector used to communicate with a host computer using RS-232 signal levels and asynchronous serial protocols. The supplied cable provides a straight-through connection from the DCE port on the EZ-KIT Lite to the DTE port on your PC. The DCD, DTR, and DSR signals are connected on the EZ-KIT Lite board.

Table 2-3 Serial Port Pin Descriptions

Pin No.	Signal Name
1	DCD
2	Transmit Data (output)
3	Receive Data (input)
4	DTR
5	Signal Ground
6	DSR
7	Request to Send (input)
8	Clear to Send (output)
9	Not Connected

2.6.2. Stereo Audio Output

The Stereo Audio Output Jack connects to the left (L) and right (R) LINE OUTPUT pins of the AD1847 codec. Use standard audio cables with 1/8 inch (3.5mm) stereo plugs to connect these signals to a set of amplified speakers.

2.6.3. Stereo Audio Input

The Stereo Audio Input jack connects directly to the left (L) and right (R) LINE 1 INPUT pins of the AD1847 codec. Use standard audio cables with 1/8 inch (3.5mm) stereo plugs to supply these inputs with line-level signals. You can also connect a microphone-level signal by changing the Input Source Selector Jumpers. (See section 5.3.1)

2.6.4. DC Power Supply Connector

The power supply connector is used to supply DC voltages to the ADSP-21061 EZ-KIT Lite board. The DC power supply included with your board mates directly to this connector.

3. USING THE EZ-KIT LITE SOFTWARE

3.1. Overview

The combination of the EZ-KIT Lite board and the monitor software operate as a target for the VisualDSP++ debugger. The debugger allows you to view the processor registers and memory and perform several debugging activities, such as setting breakpoints, stepping through code, and plotting a range of memory.

If VisualDSP++ is not installed, install it from the VisualDSP++ CD-ROM that came with this product. For more information, refer to Chapter 2, section "VisualDSP++".

This chapter provides monitor level software information on how the EZ-KIT Lite board operates with the installed software. This chapter also provides information to help you run your own programs on the ADSP-21061 EZ-KIT Lite board. This information appears in the following sections:

- "Standard Operation" – Describes the operation of the EZ-KIT Lite board
- "Running You Own Programs" – Provides information about writing and running your own DSP executables that link with the monitor program to run on the EZ-KIT Lite board

3.2. Standard Operation

This section covers the standard operation of the EZ-KIT Lite board. It describes the I/O capabilities of the on-board components, board power-up, and the on-board monitor program.

3.2.1. I/O Devices

This section describes the different I/O devices on the ADSP-21061 EZ-KIT Lite. These are flags, external interrupts, and serial ports.

3.2.1.1. Flags

The ADSP-21061 SHARC processor has four I/O flags that you can program as inputs or outputs. Bits in the MODE2 register control the direction. At reset, all of the flags are configured as inputs. Bits in the ASTAT register contain the value of each of the flag pins.

FLAG0 controls the pin on the AD1847 codec. Clearing FLAG0 (=0) holds the AD1847 in reset. Setting FLAG0 (=1) releases RESET and restarts the AD1847. The programmer should configure FLAG0 as an output and set high during program initialization. FLAG0 is also connected to LED D2 on the ADSP-21061 EZ-KIT Lite. LED D2 is lit when FLAG0 is cleared.

FLAG1 is connected to the pushbutton switch labeled “FLAG1” (S1) and to LED D3. The programmer should configure FLAG1 as an input during program initialization. The DSP’s FLAG1 input value is “0” (and LED D3 will light) when the pushbutton is depressed and the value is “1” when the pushbutton is released.

FLAG2 is connected to LED D4. The programmer should configure FLAG2 as an output during program initialization. The LED D4 lights while FLAG2 is zero.

FLAG3 is connected to LED D6. The programmer should configure FLAG3 as an output during program initialization. The LED D6 lights while FLAG3 is zero.

Table 3-1 Flag Summary

FLAG	USE
FLAG1	Pushbutton Input
FLAG0, 2, and 3	LED Feedback

3.2.1.2. External Interrupts

The ADSP-21061 SHARC DSP has three external interrupt pins. They are prioritized, individually maskable (IMASK register), and can be configured to be edge sensitive or level sensitive (bits in the MODE2 register). At reset, all external interrupts are level-sensitive and masked. Other relevant ADSP-21061 SHARC DSP registers are MODE1 (NESTM and IRPTEN bits), IRPTL, and IMASKP.

Two of the ADSP-21061 SHARC DSP’s three external interrupt inputs are allocated on the ADSP-21061 EZ-KIT Lite.

IRQ0 is not connected to any devices on the ADSP-21061 EZ-KIT Lite; however, it is connected to one of the expansion connectors.

IRQ1 is connected to a pushbutton switch on the ADSP-21061 EZ-KIT Lite board. Pressing the pushbutton generates the interrupt.

IRQ2 is connected to the 16550 UART, which can be programmed to generate an interrupt when it requires attention.

3.2.1.3. Serial Ports

The ADSP-21061 SHARC processor has two high-speed, synchronous serial ports (SPORTs). They can operate in point-to-point connection in full-duplex mode with independent transmit and receive data lines and clocks, or they can be wired together with multiple SPORTs to operate in a time division multiplexed (TDM) mode.

SPORT0 is connected to the serial port on the AD1847 SoundPort codec. This port is configured for multi-channel TDM operation.

SPORT1 is not connected to any devices on the ADSP-21061 EZ-KIT Lite; however, it is connected to one of the expansion connectors.

3.2.2. POST Routines

POST (Power On Self-Test) routines are a series of standard tests and initializations that the EZ-KIT Lite performs on a power-on reset.

3.2.2.1. UART Check/Initialization

The UART check is performed in three stages. Two of these stages are implemented in the POST. The third is controlled by the host (PC), when it attempts to connect to the EZ-KIT Lite. These stages are:

• Register Write

This test confirms that the ADSP-21061 SHARC processor is capable of writing to (and reading from) a register in the UART. Three patterns are written to and then read from a register in the UART, and tested. All three patterns must be read back correctly to pass this test.

• Internal Loop Back

In this test, 256 bytes are sent to and read from the UART. This test checks the functionality of the UART connections from the ADSP-21061 SHARC processor, up to and through the UART chip.

• Transmitted Loop Back

The last UART test is performed by the host after the POST is complete. In this test, the host sends the UART test protocol. This protocol specifies the number of bytes that are transmitted to the EZ-KIT Lite board, and instructs the board to echo the byte stream back to the host. This test determines whether the EZ-KIT Lite board is set to the correct baud rate and verifies the external connections between the board and the host.

On power up, the EZ-KIT Lite board defaults to a baud rate of 115200 with 8 data bits, 1 stop bit, and no parity. If you want to change this rate, change it after the POST is complete by using the **Settings -> Baud Rate** command from the menu bar. Note that setting the baud rate to a lower number can significantly slow the board's response to all debug activities.

3.2.3. Monitor Program Operation

The Monitor runs on the EZ-KIT Lite board as part of the DSP executable, and provides the ability to download, debug, and run user programs. The Monitor uses VisualDSP++ as the interface. Using the EZ-KIT Lite as a target allows you to operate the board remotely.

On initial power-up of the EZ-KIT Lite board, the original monitor is booted from the EPROM to the target board. Once you start the EZ-KIT Lite debug session using VisualDSP++, this monitor downloads a new monitor from your PC onto the target board. The new monitor communicates to the target dll through the serial port.

There are three main components of the new monitor program:

- Halt loop
- UART ISR
- Command Processing Kernel

The monitor program **idles in the Halt loop** when it is not running user code. While there, you can read/write memory, read/write registers, download programs, set breakpoints, change the UART's baud rate, and single-step through code.

To enter the halt loop from your code, you must halt user code—either with a breakpoint or a halt instruction. At this point, the halt loop polls the UART. With every character received from the UART, the command-processing kernel verifies whether a full command has been received. If a command has been received, the kernel processes the command; otherwise control is returned to the halt loop to wait for more characters. The only method of executing your code once the halt loop has been entered is to send a Run or Single Step command in the debugger.

The **UART ISR** is entered when user code is running, but the host is still interacting with the board. As the host sends bytes, the UART ISR takes the data stream from the UART and builds the command. Similar to the halt loop, each character received is passed to the command-processing kernel. Unlike the halt loop, the monitor returns to the user code immediately after the interrupt is serviced.

Be aware of the following restrictions to ensure correct board operation.

- If the user program disables the UART interrupt or changes the UART interrupt vector the host loses contact with the monitor while the user program is running.

- When nesting is turned off the host loses contact with the monitor while the program is running and in an ISR.

- The host loses contact with the monitor while the program is running and in the timer ISR, provided the highest priority timer vector is used.

- The host cannot halt with the debugger's Debug, Halt command if global IRQ enable is disabled (IRPTEN bit); however, breakpoints work.

Command processing, initiated from either the UART ISR or the Halt Loop, is done in the command-processing kernel. This kernel parses the commands and executes the instructions. If the instruction requires that data be sent back to the host, the kernel initiates the response.

3.3. Running Your Own Programs

This section provides basic information needed to run your own programs on the ADSP-21061 EZ-KIT Lite board. Build these programs using the ADSP-21061 SHARC processor tools. This information includes rules for using processor memory.

Although there are many ways to develop programs in VisualDSP++, all program evaluation within the environment include the following steps:

1. Create a new project file
2. Set the target processor under "Project Options"
3. Add and edit project source files
4. Customize project build options
5. Build a debug version of the project
6. Debug the project
7. Build a release version of the project

By following these steps, DSP projects build consistently and accurately with minimal project management. The *ADSP-2106x SHARC User's Manual* provides detailed information on programming the processor, and the VisualDSP++ manuals provide information on code development with the ADSP-21061 SHARC processor tools.

- Do not run more than one ADSP-21061 EZ-KIT Lite session at any one time. You may run an EZ-KIT Lite debug session and a simulator, or you can run a JTAG ICE session and a simulator session at the same time.

- Before making any changes to the source code in VisualDSP++, clear all breakpoints and close all Editor windows. Then make the changes, rebuild the program, and reload it in VisualDSP++.

- If you are creating a C/C++ program use the 060_hdr.asm file supplied with the demo programs. This file reserves the IRQ2 interrupt vector table for the UART.

3.3.1. ADSP-21061 SHARC Processor Memory Map

The ADSP-21061 SHARC processor has 1M of internal SRAM that can be used for program or data storage. The configuration of on-chip SRAM is detailed in the *ADSP-2106x SHARC User's Manual*. Table 3-2 shows the memory map of the ADSP-21061 EZ-KIT Lite.

The IMDW0 bit in the SYSCON register must be set to 1 to keep communication with the host. This bit determines whether data accesses made to internal memory block 0 are 40-bit, three-column accesses (set = 1) or 32-bit, two-column accesses (cleared = 0). The monitor program requires three-column data accesses to memory block 0.

On reset, restart, and halt, the debug monitor kernel forces IMDW0 to 1 and IMDW1 to 0, but user code should also set these bits to ensure that it operates in the same way on both the simulator and the EZ-KIT Lite board. These settings affect data accesses only, not instruction fetches. Block 0 resides in three-column memory. If you are storing data in block 0, it must be in three-column format.

Table 3-2 Memory Map

	End Address	**Content**
0x0000 0000	0x0000 0FF	Registers
0x0000 0100	0x0001 ffff	(reserved)
0x0002 0000	0x0002 3FFF	Block 0 Normal Word (32/48) Addresses
0x0002 0000	0x0002 1fff	Block 0 48-bit word Addressing
0x0002 0000	0x0002 3fff	Block 0 32-bit word Addressing
0x0002 4000	0x0002 7fff	Block 1 Normal Word (32/48) Addresses
0x0002 4000	0x0002 5fff	Block 1 48-bit word Addressing
0x0002 4000	0x0002 7fff	Block 1 32-bit word Addressing
0x0004 0000	0x0004 7fff	Block 0 Short word (16-bit) Addressing
0x0004 8000	0x0004 ffff	Block 1 Short word (16-bit) Addressing

NOTE: Use caution when accessing the boot EPROM. The EPROM chip select (BMS) has the same limitations as MS0. EPROM's larger than 128K x 8 have restricted access to their data below address 0x020000, and their data aliases to other memory locations. The user program can access this data from these other locations.

Table 3-3 shows currently used memory locations on the EZ-KIT Lite board by the monitor. You may not use these locations in programs.

Table 3-3 Restricted Memory Space

Segment Name	Description	Memory Block	Width	Start Address	End Address
seg_newrth	Routine that will overwrite old monitor's run-time header (rth) with one from the new Monitor	0	48	0x21974	0x21992
seg_rsvd_rth	Run-time header for new Monitor	0	48	0x21993	0x21a12
seg_post	POST routine of the Monitor	0	48	0x21a13	0x21a52
seg_jmp	Jump Routines	0	48	0x21a53	0x21a59
seg_rsvd_pmco	Kernel Code	0	48	0x21a5a	0x21da0
seg_rsvd_pmda	PM Data of the Monitor	0	48	0x21da1	0x21fff
seg_rsvd_dmda	DM Data of the Monitor	1	32	0x27ffd	0x27fff

NOTE: In order for your program to work properly, your program must not overwrite any memory location utilized by the monitor.

4. DEMONSTRATION PROGRAMS

4.1. Overview

This chapter describes how to load and run the demonstration programs supplied with the ADSP-21061 EZ-KIT Lite board. The demos are designed to run on the VisualDSP++ debugger supplied on a CD-ROM that shipped with this product. For detailed information on the debugger features and operation, see the *VisualDSP++ Debugger Guide & Reference.*

4.2. Starting the VisualDSP++ Debugger

After the VisualDSP++ software and license have been installed, click the Windows Start menu.

Select **Programs -> VisualDSP -> Debugger** from the **Start** menu.

From the **Session** menu, choose **New Session**. The **New Session** dialog box appears.

Configure the debug session as shown in Figure 4-1 and click **OK**.

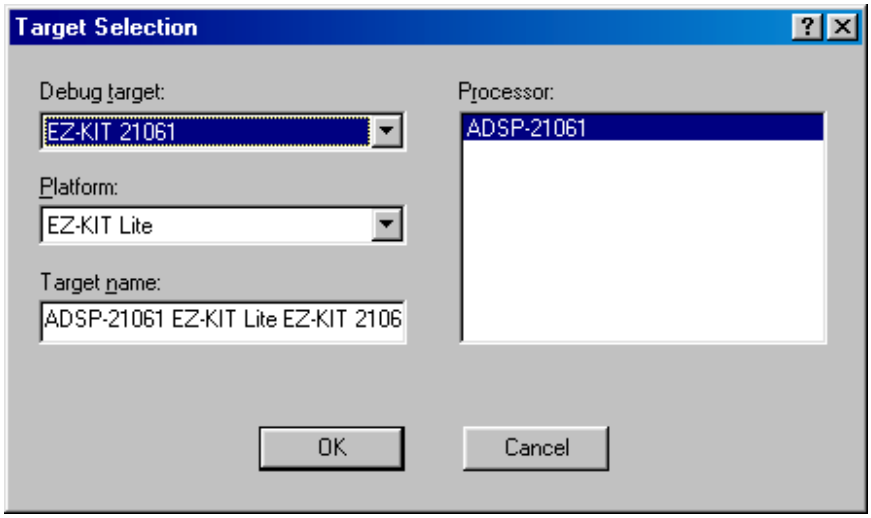


Figure 4-1 Target Selection Dialog

A Target Message dialog box appears.

Figure 4-2 Target Message

Press the **Reset** button on the EZ-KIT Lite evaluation board.

After 5 seconds all the LEDs light up and then all of them turn off except the POWER LED. Ensure that all LEDs turn off (except for the POWER LED) before you click **OK**.

During this delay, the POST test verifies operation of the UART. After the LEDs go dark, a message box opens with the message shown in Figure 4-3.

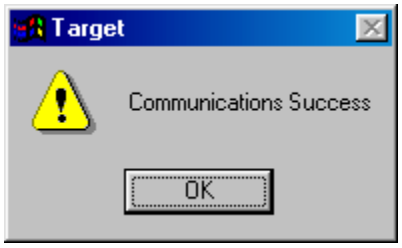


Figure 4-3 Target Communications Status Message Box

Click **OK**.

The initialization completes and the disassembly window opens. The code in the disassembly window is the EZ-KIT Lite monitor program.

4.3. Debugger Operation with the ADSP-21061 EZ-KIT Lite

The *VisualDSP++ Debugger Guide & Reference* contains most of the information you need to operate the VisualDSP++ debugger with your EZ-KIT Lite evaluation board.

4.3.1. Loading Programs

Because you are loading programs to a hardware target through the serial port, the load process takes more time than loading using the simulator. Wait for the **Load Complete** message in the **Output** window before you attempt any debug activities.

To load a program:

From the **File** menu, choose **Load**. The **Open a Processor Program** dialog box appears.

Navigate to the folder in which the DSP executable file resides. The demos that are supplied with the EZ-KIT Lite are located in C:\Program Files\Analog Device\VisualDSP++\21k\EZ-KITS\ADSP-21061\Demos

NOTE: All file directories assume a default installation.

Select the .dxe file and click **Open**. The file loads and the message **Load Complete** appears in the Output window when the load process has completed.

4.3.2. Registers and Memory

To see current values in registers, use the **F12** key or **the Window -> Refresh** command.

- Values may not be changed while a user program is running.
- The current version of the VisualDSP++ debugger does not let you view hardware stack information.

4.3.3. Setting Breakpoints and Stepping

- Breakpoints set in the last three instructions of a DO-loop are allowed, but cause improper debugger operation.
- Breakpoints set after a delayed branch instruction and before the branch occurs cause improper debugger operation.
- The single-step command steps through a delayed branch instruction and the last three instructions of a DO-loop.
- VisualDSP++ automatically inserts breakpoints at the function **`Main()`** and at the **`_exit`** instruction when the **`Settings->Run To Main`** command is selected.

4.3.4. Resetting the EZ-KIT Lite Board

You can reset the EZ-KIT Lite board with the pushbutton switch on the board or with the **Debug -> Reset** command in VisualDSP++. Both methods clear and reset the chip's memory and debug information, so you will need to reload any programs that were running. The **Debug -> Restart** command resets the processor; however, the processor retains all debug information and memory contents.

- Do not use the **reset** pushbutton while the debugger is open unless the debugger requests you to press it.

NOTE: If a message in the VisualDSP++ debugger Output window requests you to reset board, you must do the following to ensure proper download of the monitor from the PC:

1. **From Session menu, choose Select Session.**
2. **Select your Monitor Debug session.**
3. **Follow on-screen dialog boxes.**

4.4. Demonstration Program

As described in the previous sections, you can start the included EZ-KIT Lite demonstration programs from the File menu or from toolbar buttons. Each of the following sections describes what the demonstration programs do and how to run them.

4.4.1. Bandpass Filter Demo

This program demonstrates the effect of four bandpass filters against no filter on a codec input source or an internally-generated noise source. This demonstration starts with a talk-through program, with the Input Source set to Codec and the Filter range set to None. The AD1847 codec digitizes the analog input signal and transmits the data to the ADSP-21061 SHARC processor's serial port. The ADSP-21061 SHARC processor reads data from the serial port and retransmits the data back to the codec. The codec converts the data to an analog signal that drives the output device. The sample rate for the digital data is 8 kHz. You may change the filter range and source without changing the code:

To change the filter range:

1. From the Memory menu, choose **Two Column**. A two-column window appears.
2. Right click inside the Memory window and click **Go To…**
3. From the Go To Address dialog box, click **Browse.**
4. From the Browse Symbol dialog box double-click on the symbol **filter**. In the Memory window you will now be at the address for filter.

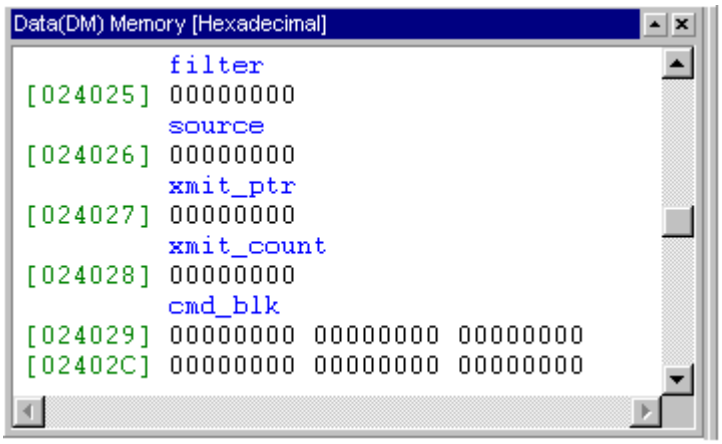


Figure 4-4 Memory Window for Symbol 'Filter'

The address that **filter** represents is right under the symbol name. In this case it is address 0x24025.

5. Change the value at this location to:

Table 4-1 Bandpass Filter Demo – Filter Ranges

Value @ Filter Address	Filter Range
0x00000000	None
0x00000001	Pass 328-448 Hz
0x00000002	Pass 521-710 Hz
0x00000003	Pass 825-1125 Hz
0x00000004	Pass 1308-1783 Hz

6. Hit **run**. (Note: the filter will change.)

To Change the source:

1. From the Memory menu, choose Two Column. A two-column window appears.
2. Right click inside the Memory window and click **Go To…**
3. From the Go To Address dialog box click on **Browse.**
4. From the Browse Symbol dialog box double click on the symbol **source**. In the Memory window you will now be at the address for the filter.

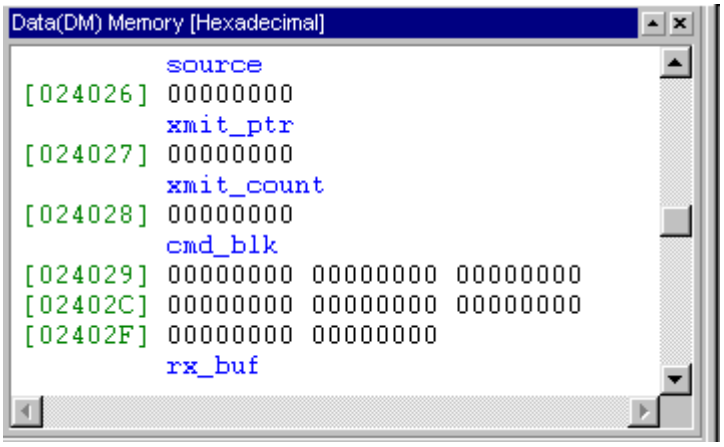


Figure 4-5 Memory Window for Symbol 'Source'

5. The address that **source** represents is right under the symbol name. In this case it is address 0x24026.

6. Change the value at this location to:

Table 4-2 Bandpass Filter Demo – Source Inputs

Value @ Source Address	Source
0x00000000	codec
0x00000001	Noise

7. When you change the source to **Noise** the ADSP-21061 SHARC processor supplies a fabricated noise source.

8. Hit **run**. (NOTE: Changes will occur.)

The C source code for this program is located in the Program Files\Analog Devices\VisualDSP\21k\EZ-KITs\ADSP-21061\Demos\Bp directory.

NOTE: All directory paths are based on default installation.

4.4.2. Primes Demo

This program calculates the first twenty prime numbers. When the calculation is completed, you can view the results in the **Expressions window**:

Open an Expressions window from View -> Debug Window -> Expressions

Click inside the **Expressions window** so it is active and type the word *primes* inside the window. Then hit **enter**.

Now expand the *primes* by clicking on the plus sign next to the word *primes*.

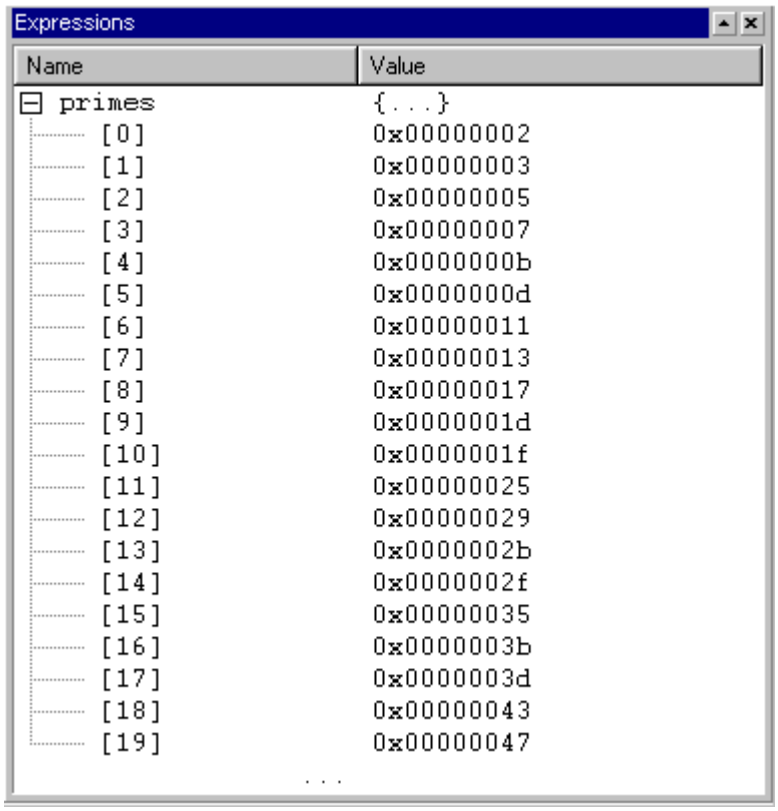


Figure 4-6 Expressions Window for Primes Demo

The C source code for this program is located in the Program Files\Analog Devices\VisualDSP\21k\EZ-KITs\ADSP-21061\Demos\primes directory.

4.4.3. Peter Gunn Demo

This program demonstrates the Karplus-Strong algorithm for simulating the sound of a "plucked" string. The ADSP-21061 SHARC processor uses the algorithm to generate digital audio samples according to data that specifies the notes to be played. The audio samples are transmitted to the AD1847 codec over a serial port. The codec converts the data to an analog signal that drives the output device.

The demo is also contained in the boot PROM and executes when the power is first applied or when you press and release the RESET button.

The assembly source code for this program is located in the Program Files\Analog Devices\VisualDSP\21k\EZ-KITs\ADSP-21061\Demos\gunn directory.

4.4.4. Blink Demo

This simple program demonstrates how to use the ADSP-21061 SHARC processor's built-in timer to toggle two LEDs on the EZ-KIT Lite board.

The C source code for this program is located in the Program Files\Analog Devices\VisualDSP\21k\EZ-KITs\ADSP-21061\Demos\blink directory.

5. WORKING WITH EZ-KIT LITE HARDWARE

5.1. Overview

This chapter describes the hardware characteristics of the ADSP-21061 EZ-KIT Lite board. It includes a description of the board's major features and section that describes the user configurable items.

5.2. Board Layout

Figure 5-1 shows the layout of the ADSP-21061 EZ-KIT Lite board, which consists of a printed circuit board measuring 4.5 inches by 6.5 inches. Figure 5-1 highlights the locations of the major components described in the following sections.

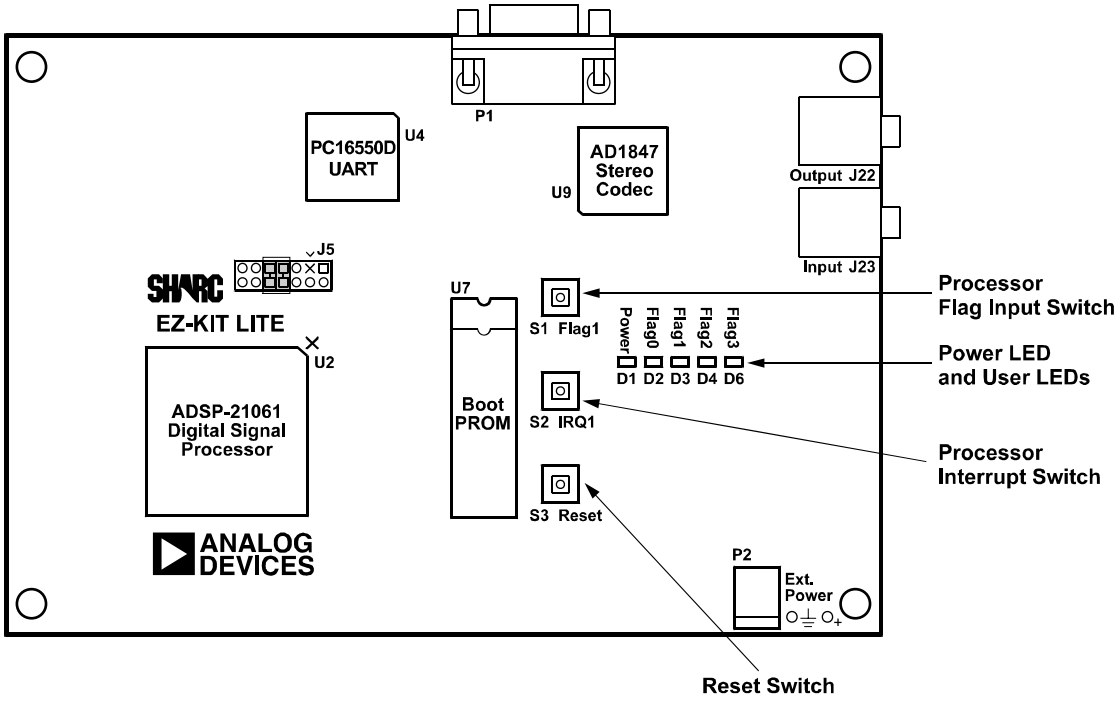


Figure 5-1 Major Components of the ADSP-21061 EZ-KIT Lite Rev. 2 Board

5.2.1. ADSP-21061 SHARC Processor

This is the ADSP-21061 SHARC processor, which operates at 40 MHz. The Pin 1 Index is located in the upper-right corner.

5.2.2. Boot PROM

The boot PROM (U7) provides 8-bit program storage that can be loaded by the ADSP-21061 SHARC processor at start-up. The socket mounted on this board is designed to accept EPROMs from 256K bits up to 8M bits. Jumpers JP1 through JP4 provide the necessary adjustments required to accommodate the different sizes of EPROM. When the ADSP-21061 SHARC processor is configured for PROM booting, the first 256 instructions (1536 bytes) are automatically loaded by the ADSP-21061 SHARC processor when reset is released.

The remaining program image must be loaded by the program that is installed in those first 256 instructions. The ldr21k utility can do this for you. Refer to the *ADSP-2106x SHARC User's Manual* for more information on program booting.

5.2.3. User Pushbutton Switches

For user input/control, there are eight pushbutton switches on the ADSP-21061 EZ-KIT Lite board: RESET, FLAG 0-3, and IRQ 0-2.

• The RESET switch initiates a power-on reset to the DSP. There are no restrictions to when the switch can be used, so do not press the switch unless you want a complete DSP reset.

• The FLAG1 switch toggles the status of a flag pin (FLAG1) to the DSP. This lets you manually trigger the flag, providing an "event" while executing software.

• The IRQ1 switch sends an interrupt (IRQ1) to the DSP. This lets you manually cause this interrupt when executing a program. IRQ2 is shared with the UART, and IRQ0 is brought out to the expansion connector.

See "Flags" section in Chapter 3, for more information on interfacing to the pushbutton switches from DSP programs.

5.2.4. User LEDs

There are four LEDs on the ADSP-21061 EZ-KIT Lite board. Your DSP program can control them to indicate certain conditions in the software or to provide feedback. The LEDs are controlled by processor FLAG outputs of the DSP and are labeled according to the flag that enables them.

5.2.4.1. Power LED

The Power LED, when on indicates that +5 VDC used by the DSP and digital circuitry is present.

5.2.5. Expansion Port Connectors

There are seven expansion connector sites that provide the signals for adding optional custom hardware. The interface contains the ADSP-21061 SHARC processor bus as well as six link ports, a synchronous serial port, interrupts, flags, and various control signals.

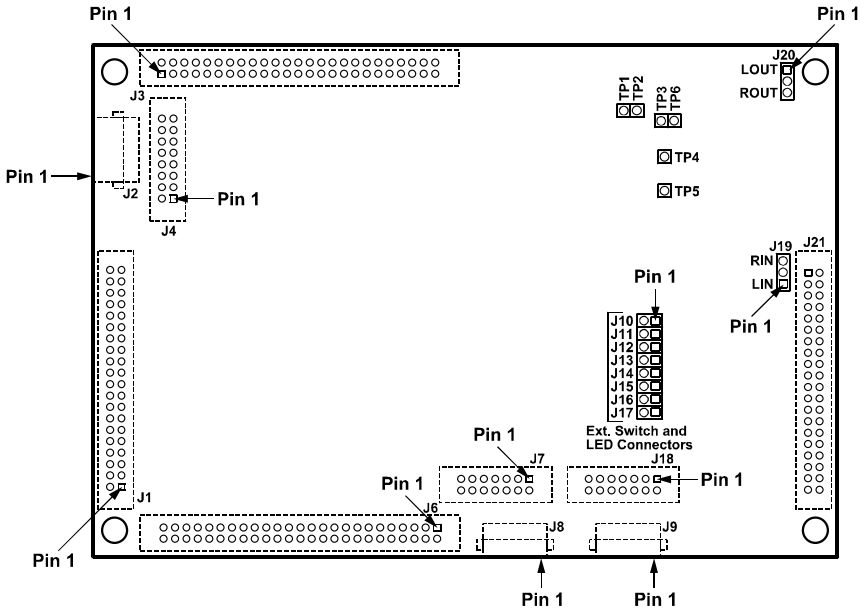


Figure 5-2 Expansion Port Connectors

5.2.6. JTAG Connector (Emulator Port)

The JTAG header (Figure 5-2) is the connecting point for the JTAG emulator probe. Note that one pin is missing (Pin 3) to provide keying. The Pin 3 socket in the mating connector has a plug inserted at that location.

The ADSP-21061 EZ-KIT Lite board is shipped with two jumpers installed across Pins 7 & 8 and 9 & 10. Remove these jumpers before installing the JTAG probe. When the JTAG probe is removed, replace these jumpers to ensure that the ADSP-21061 SHARC processor initializes correctly on power-up.

The proper power up sequence is:

1. JTAG Emulator

2. ADSP-21061 EZ-KIT Lite board

To remove power, the sequence is:

1. ADSP-21061 EZ-KIT Lite board

2. JTAG Emulator

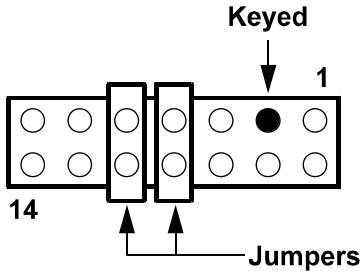


Figure 5-3 JTAG Connector with Jumpers Installed

5.3. Jumpers

Figure 5-4 shows the locations of configuration jumpers on the ADSP-21061 EZ-KIT Lite board. These jumpers should be checked before using the board to ensure proper operation. Each of the jumper selection blocks is described in the following sections.

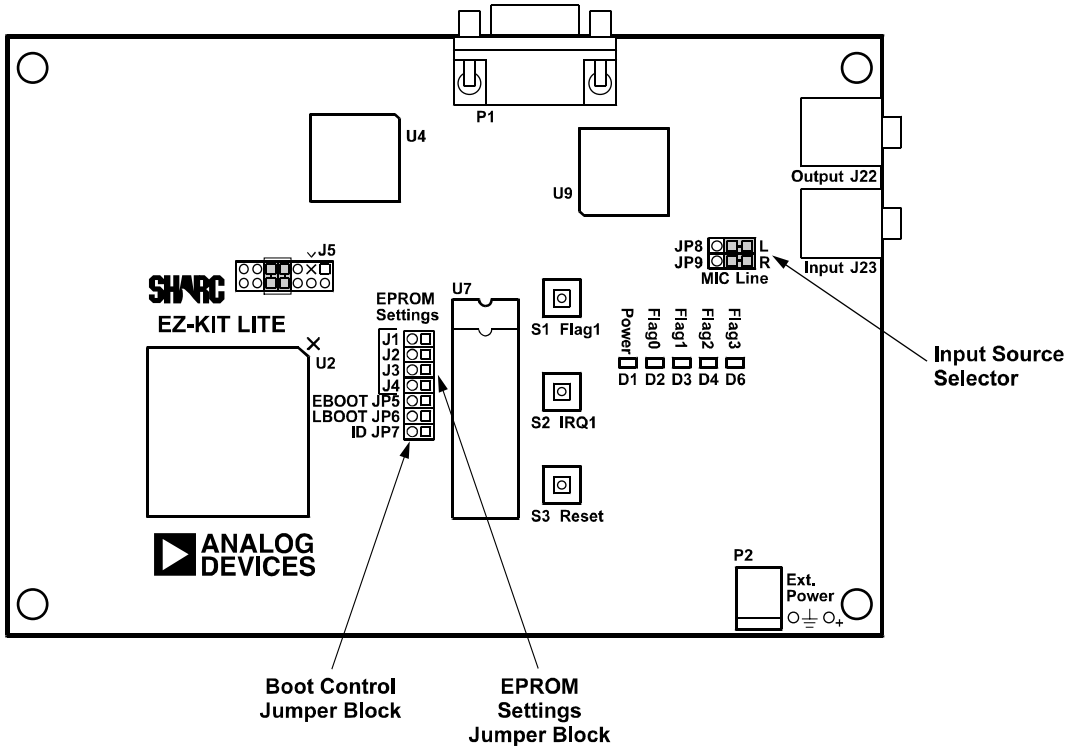


Figure 5-4 Location of Configuration Jumpers on the ADSP-21061 EZ-KIT Lite Board Rev. 2.0

5.3.1. Input Source Selector

This pair of jumpers selects whether your input signal is standard line level or microphone level. If you select microphone level, the input signal is amplified before it is sampled by the AD1847 codec.

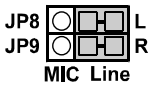


Figure 5-5 Line-Level Input Audio (Factory Default)

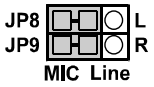


Figure 5-6 Microphone-Level Input Audio

5.3.2. Boot Control Jumper Block

These jumpers control three of the ADSP-21061 SHARC processor's pins: EBOOT (JP5), LBOOT (JP6), and ID0 (JP7). The first two define the processor's boot mode according to the following table:

Table 5-1 Boot Mode Selection

Mode	JP5	JP6
EPROM	OUT	IN
Host Port	IN	IN
Link Port	IN	OUT

The ADSP-21061 EZ-KIT Lite is shipped with EPROM boot mode selected by default. The pins at location JP6 are shorted on the circuit board.

The ID0 pin determines the ADSP-21061 SHARC processor's multiprocessor ID. If JP7 is shorted, the multiprocessor ID number is 000. If JP7 is left open (the factory-shipped default), the multiprocessor ID number is 001.

5.3.3. EPROM Settings Jumper Block

These four jumpers (JP1-JP4) define the size of EPROM that you have installed in the boot PROM socket. The ADSP-21061 EZ-KIT Lite is shipped with 27C010 selected by default. The pins at locations JP2 and JP3 are shorted on the circuit board.

Table 5-2 EPROM Jumper Selection Chart

U7	JP1	JP2	JP3	JP4
27C256 (32K x 8)	OUT	IN	OUT	IN
27C512 (64K x 8)	OUT	IN	IN	OUT
27C010 (128K x 8)	OUT	IN	IN	OUT
27C020 (256K x 8)	IN	OUT	IN	OUT
27C040 (512K x 8)	IN	OUT	IN	OUT
27C080 (1M x 8)	IN	OUT	IN	OUT

5.4. UART

The UART (U4) and the line driver (U5) provide the RS-232 interface used to communicate with the PC. The PC16550D is similar to devices used in most PCs. It has a programmable bit rate and has transmit and receive FIFO registers.

The UART is attached to the ADSP-21061 SHARC processor's external memory bus and is selected by MS1 (external memory bank 1). The UART can generate an interrupt to the ADSP-21061 SHARC processor on IRQ2.

5.5. AD1847

The AD1847 codec (U9) provides the stereo audio input (A/D) and output (D/A) interface. It is connected to the ADSP-21061 SHARC processor via SPORT0. This high-speed synchronous serial port carries all of the data, control, and status information between the DSP and the codec.

6. PROGRAMMING REFERENCE

6.1. Overview

This chapter provides the technical details you need to write programs for the ADSP-21061 EZ-KIT Lite. One section focuses on memory models, addresses, and other resources for programs that run on the ADSP-21061 SHARC processor. The other section focuses on the serial host interface so that you can write programs on the PC that talk directly to the monitor program running on the EZ-KIT Lite board.

6.2. DSP Programs

This section describes the model for EZ-KIT Lite DSP programs by focusing on the resources that are available to the ADSP-21061 SHARC processor. These resources include the memory map, the flags, the interrupts, and the serial ports. Table 6-1 summarizes these resources as they are implemented on the EZ-KIT Lite board.

Table 6-1 Summary of EZ-KIT Lite ADSP-21061 SHARC Processor Resources

	MS (Memory Select)	**FLAG**	**IRQ**	**Serial Port**
0	Expansion Connector	LED D2 / 1847 codec RESET	Expansion Connector	1847 codec
1	16550 UART	From Pushbutton / LED D3	From Pushbutton	Expansion Connector
2	Expansion Connector	LED D4	From 16550 UART	
3	Expansion Connector	LED D6		

6.2.1. Memory Map

The ADSP-21061 SHARC processor memory model defines three main memory spaces. Internal memory space addresses an ADSP-21061 SHARC processor's on-chip dual-ported SRAM. Multiprocessor memory space addresses the on-chip SRAM of other ADSP-21061 SHARC processors in the same cluster (i.e., ADSP-21061 SHARC processors that share a common processor bus). External memory space addresses other devices on the shared bus such as SRAM or DRAM.

6.2.1.1. Internal Memory Space

Since the ADSP-21061 EZ-KIT Lite has no external memory, you must store all code instructions and data in the built-in SRAM. The ADSP-21061 SHARC processor has one megabit of internal dual-ported SRAM divided into two 512-kilobit blocks. The blocks are designed so that you can configure regions of memory to be either 32 or 48 bits wide. The *ADSP-2106X SHARC User's Manual* contains detailed information about the configuration and limitations of this on-chip SRAM.

If you are writing programs to be loaded by the built-in kernel, you should be aware of how it uses memory. The .ldf file for the EZ-KIT Lite defines memory segments that are compatible with programs you write for the C/C++ compiler and the assembler.

6.2.1.2. Multiprocessor Memory Space

The multiprocessor memory space (MMS) is consumed by ADSP-21061 SHARC processors connected to the external processor bus (up to six as defined by the maximum cluster size). ADSP-21061 SHARC processors appear in specific portions of the MMS according to their multiprocessor ID. The default multiprocessor ID is one (001) for the ADSP-21061 SHARC processor. You can change the ID to zero (000) with a jumper setting (see section 5.3.2).

Since the ADSP-21061 EZ-KIT Lite is a single-processor board, the only way to view other ADSP-21061 SHARC processors through the MMS is through the expansion connectors to your custom hardware.

6.2.2. Stereo Audio Codec Programming

The stereo audio interface built into the ADSP-21061 EZ-KIT Lite is based on the AD1847 Stereo SoundPort codec. You can find detailed programming information in the AD1847 data sheet.

The ADSP-21061 EZ-KIT Lite uses SPORT0 to communicate with the AD1847's control and data interface. The ADSP-21061 SHARC processor's FLAG0 port controls the AD1847's RESET and BM pins. You can disable the codec by clearing FLAG0 to zero.

6.2.3. Kernel Compatibility

When you write programs that run on the EZ-KIT Lite board's ADSP-21061 SHARC processor, there are certain programming restrictions you should consider to ensure that the on-board kernel program continues to operate. If you violate any of these restrictions, the kernel may become disabled or unstable.

- **Avoid using kernel memory regions.** The kernel uses two regions of the ADSP-21061 SHARC processor's internal SRAM. These regions are defined as separate segments in the EZ-KIT Lite's Linker Description File (.ldf).
- **Avoid the UART.** Since the kernel uses the RS-232 interface to communicate with the host computer, it must manage the UART chip. The UART is accessed in external memory space within external memory bank 1 (see Table 6-1).
- Your program can freely change the MSIZE setting in the SYSCON register (potentially changing the base address of external memory bank 1) since the kernel recalculates the UART base address every time it needs to access the chip.
- **Do not interfere with the UART interrupt.** The kernel depends on the hardware interrupt signal IRQ2 to communicate with the UART. There are several ways that your program can affect the kernel's ability to receive interrupts from the UART.
- **Do not disable interrupts globally or disable interrupt nesting.** Bit 12 (IRPTEN) and Bit 11 (NESTM) in the MODE1 register must remain set.
- **Do not mask IRQ2.** The IMASK register allows your program to enable or disable individual interrupts. Bit 6 (IRQ2I) in the IMASK register must remain set.
- **Do not change IRQ2's sensitivity.** The UART uses this interrupt signal to indicate multiple conditions. If the kernel satisfies one condition, there may be other conditions requiring attention; however, the IRQ2 signal will not transition to the inactive state and then back to the active state. Therefore, the ADSP-21061 SHARC processor's interrupt controller must treat IRQ2 as a level-sensitive signal. Bit 2 (IRQ2E) in the MODE2 register must remain cleared.

7. REFERENCE

7.1. OVERVIEW

This chapter is a reference for VisualDSP++. Because the debugger is dynamic, menu selections, commands, and dialog boxes change, depending on the target being used. This chapter provides information on all of the menu selections, commands, and dialog boxes when the target is the ADSP-21061 EZ-KIT Lite evaluation board. For all other commands, see the *VisualDSP++ Guide & Reference Manual*. Note that grayed out commands are unavailable with this target.

7.2. SETTINGS MENU COMMANDS

All of the commands that pertain to the EZ-KIT Lite board are contained in the Settings and Demo menus. The Settings menu provides access to the following commands:

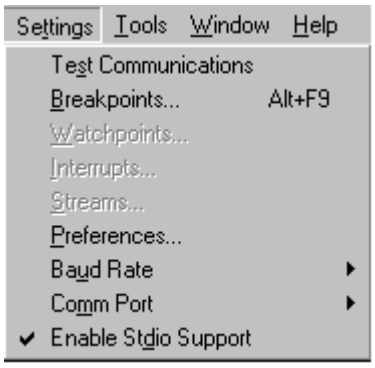


Figure 7-1 Settings Menu Commands

7.2.1. Test Communications

Tests the PC to ADSP-21061 EZ-KIT Lite communications. Responses are Communications Success or various error messages sent to the Output window. In most cases, resetting the ADSP-21061 EZ-KIT Lite re-establishes communication.

7.2.2. Baud Rate

Sets up the baud rate of the current COM port and UART. Choices are 9600, 19200, 38400, 57600, and 115200. The default rate is 115200. Once change is made, resetting is not needed for subsequent debug sessions. (NOTE: Using a baud rate of 9600 causes the ADSP-21061 EZ-KIT Lite to operate very slowly and can also cause it to hang.)

7.2.3. COM Port

Selects a PC communications port for the ADSP-21061 EZ-KIT Lite board. Choices are COM (1-4). When changing the COM port it takes a second for the window to become active again.

8. SCHEMATICS

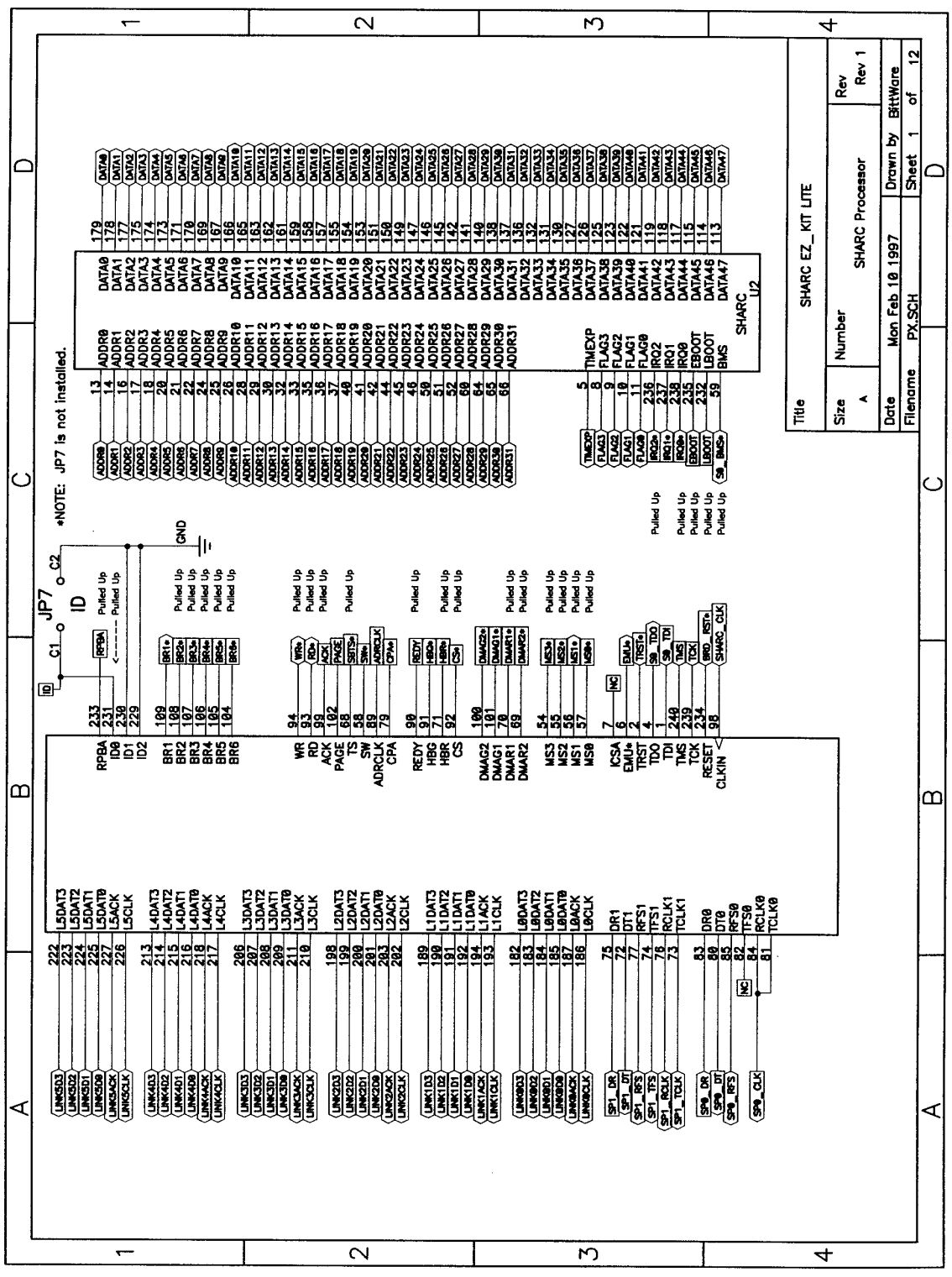
8.1. Overview

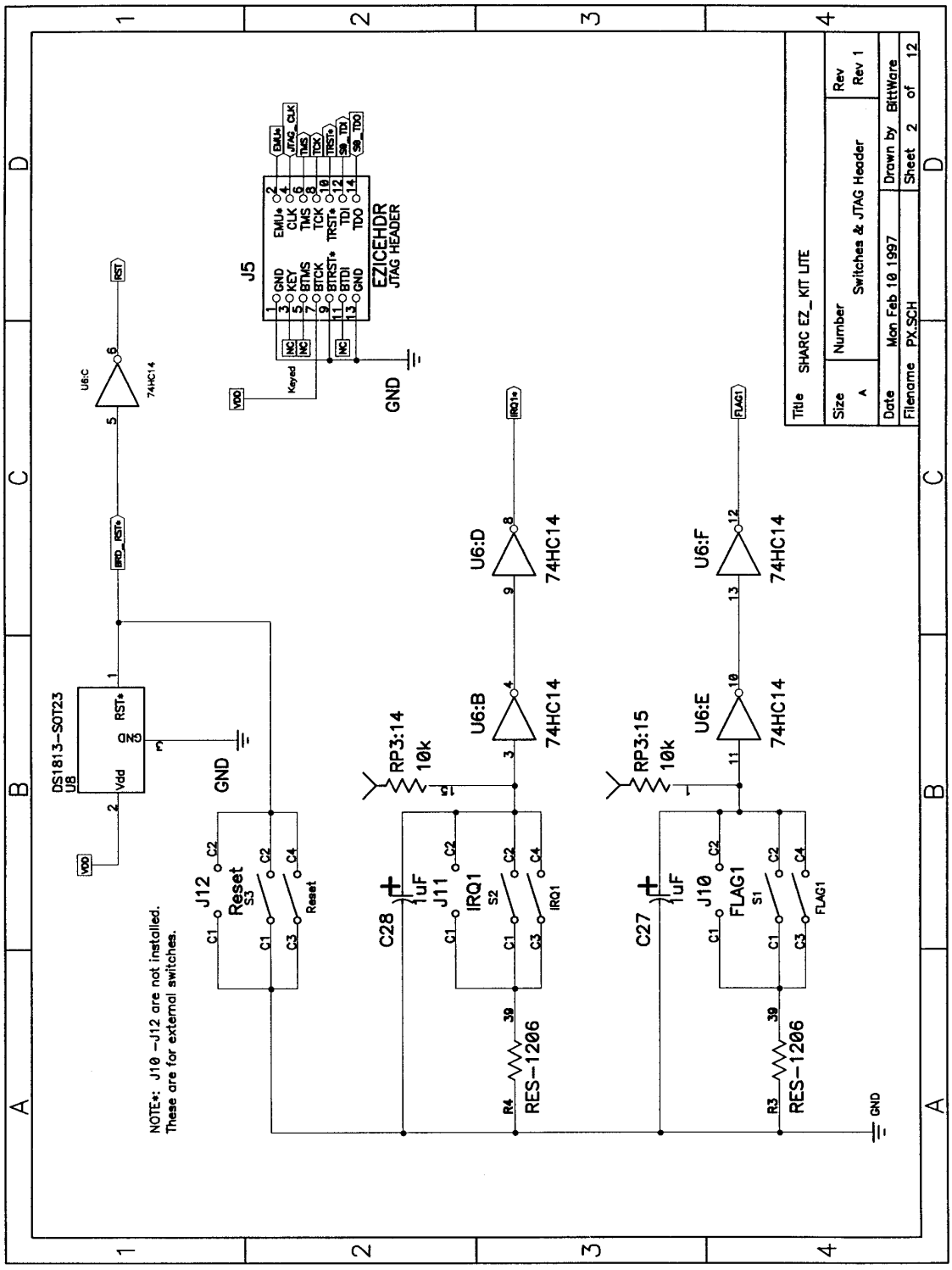
This chapter provides the design information used to build the ADSP-21061 EZ-KIT Lite board. The schematics are in section 8.2. The logic equations and state diagram for the 16V8 programmable logic device (U3) are listed in section 8.3.

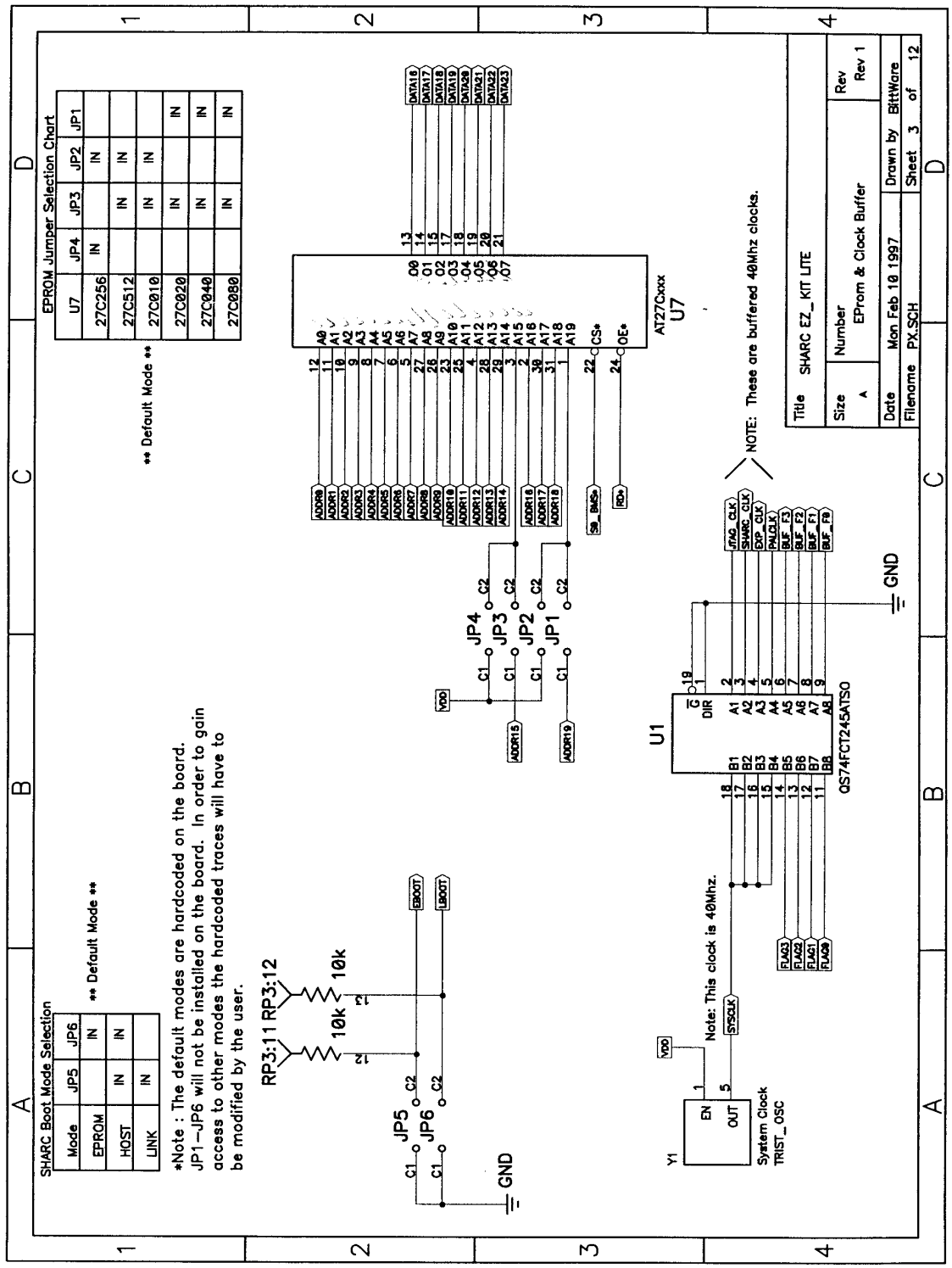
8.2. Board Schematics

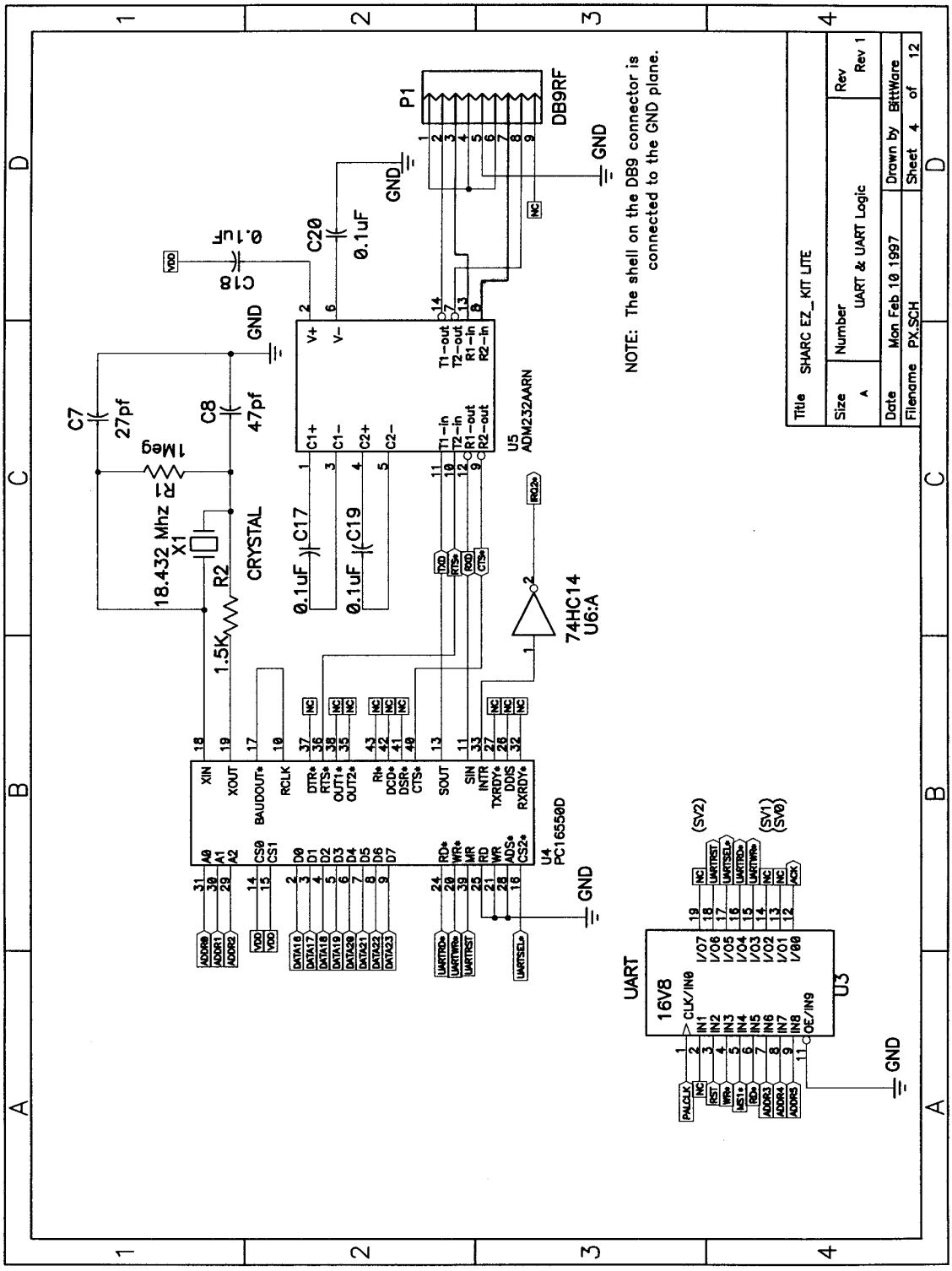
The following pages provide complete electrical schematics for the ADSP-21061 EZ-KIT Lite Rev. 2.0 board.

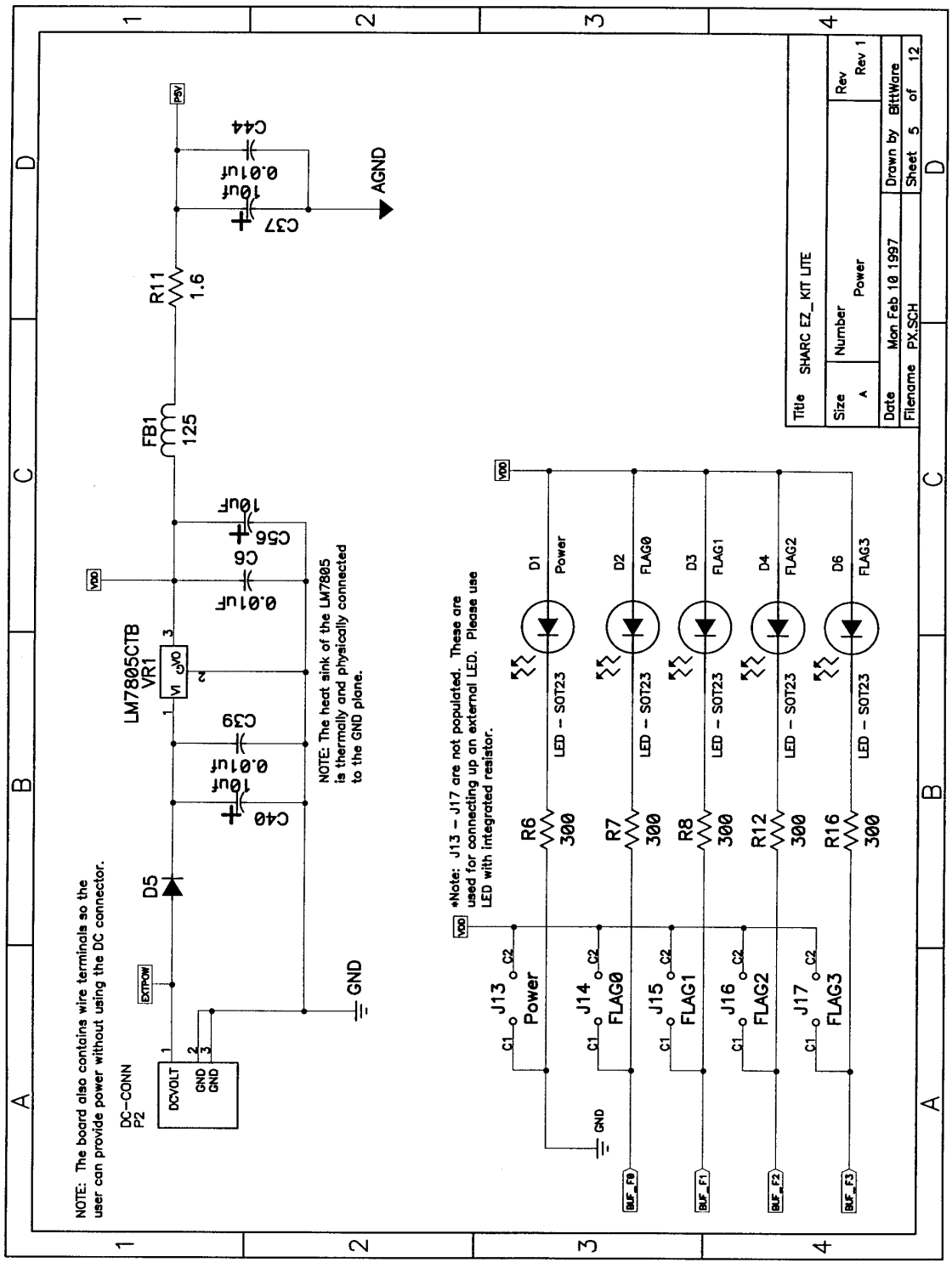
Table 8-1 ADSP-21061 EZ-KIT Lite Schematic Contents

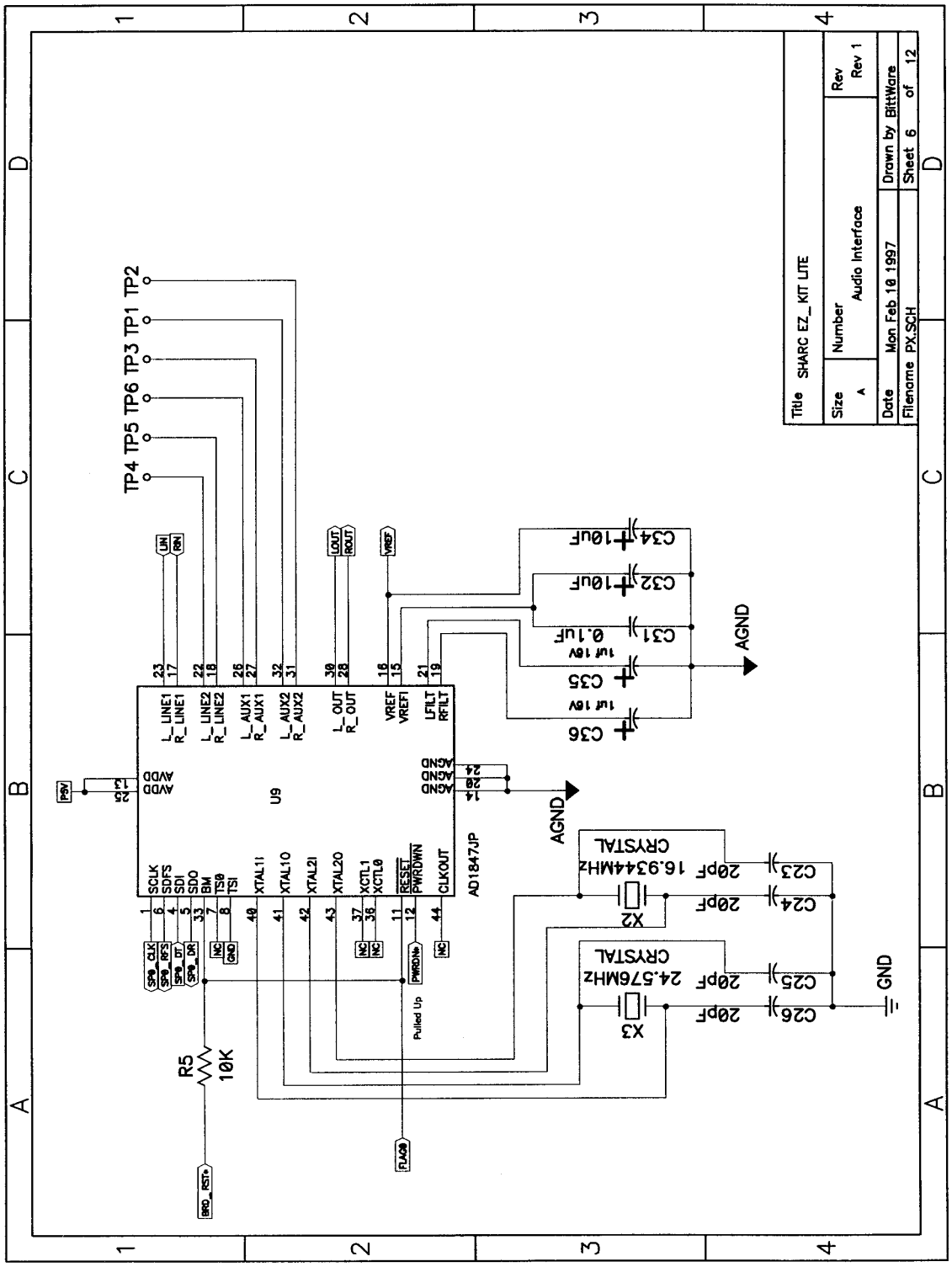

SHARC EZ_KIT LITE
SHARC Processor
Rev 1
Mon Feb 10 1997
BitWare
PX.SCH
Sheet 1 of 12
*NOTE: JP7 is not installed.
JP7
ID
C1
C2
GND
Pulled Up
SHARC U2

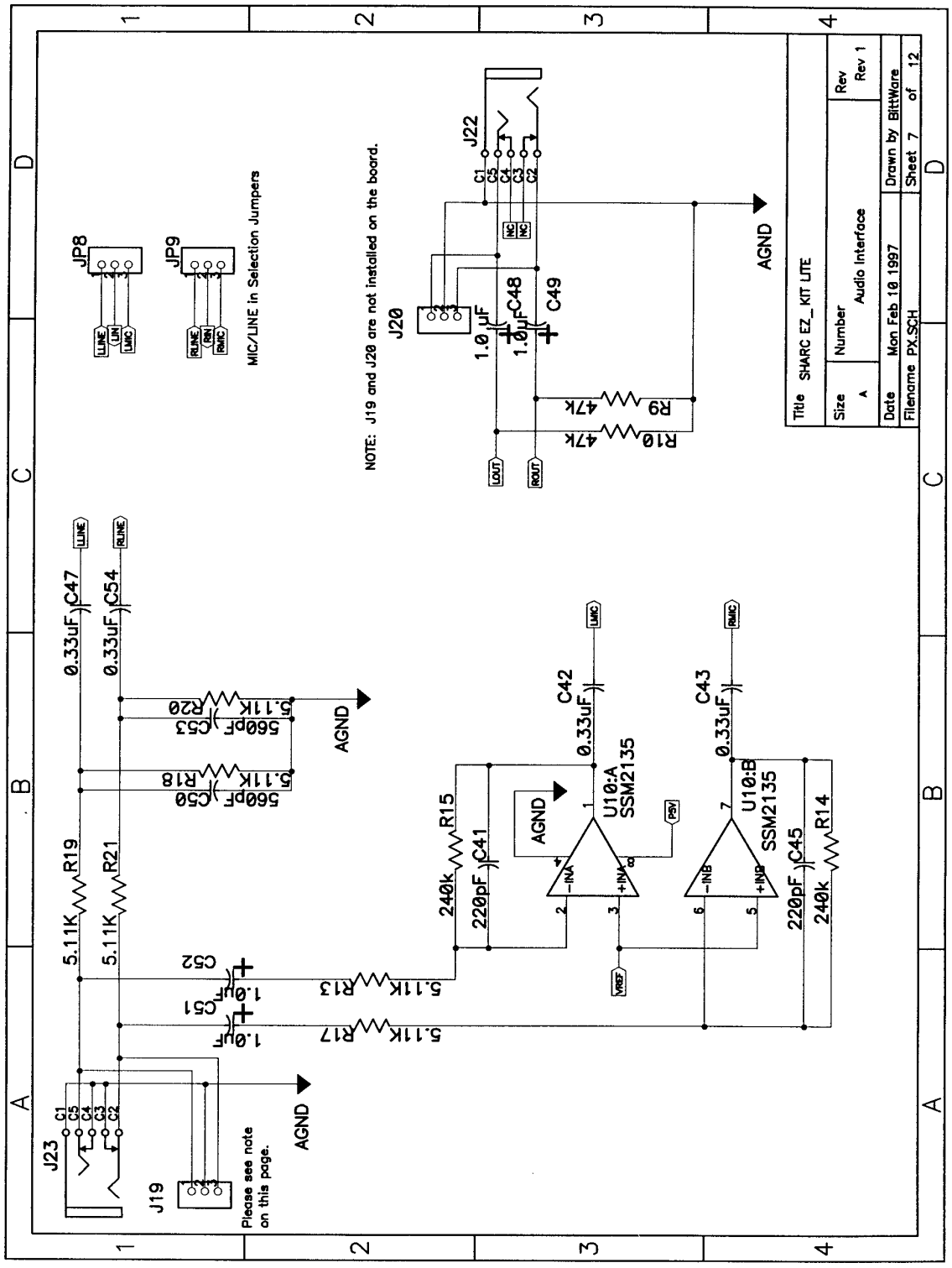

Title SHARC EZ_KIT LITE
Size A
Number Switches & JTAG Header
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 2 of 12
J5
EZICEHDR
JTAG HEADER
GND
VDD
Keyed
U6:C
74HC14
RST
IRD_RST*
DS1813-SOT23
U8
RST*
Vdd
J12
Reset
S3
C28
.1uF
J11
IRQ1
S2
C27
.1uF
J10
FLAG1
S1
RP3:14
10k
RP3:15
10k
U6:B
U6:D
U6:E
U6:F
IRQ1*
FLAG1*
R4
R3
RES-1206
NOTE*: J10 -J12 are not installed.
These are for external switches.

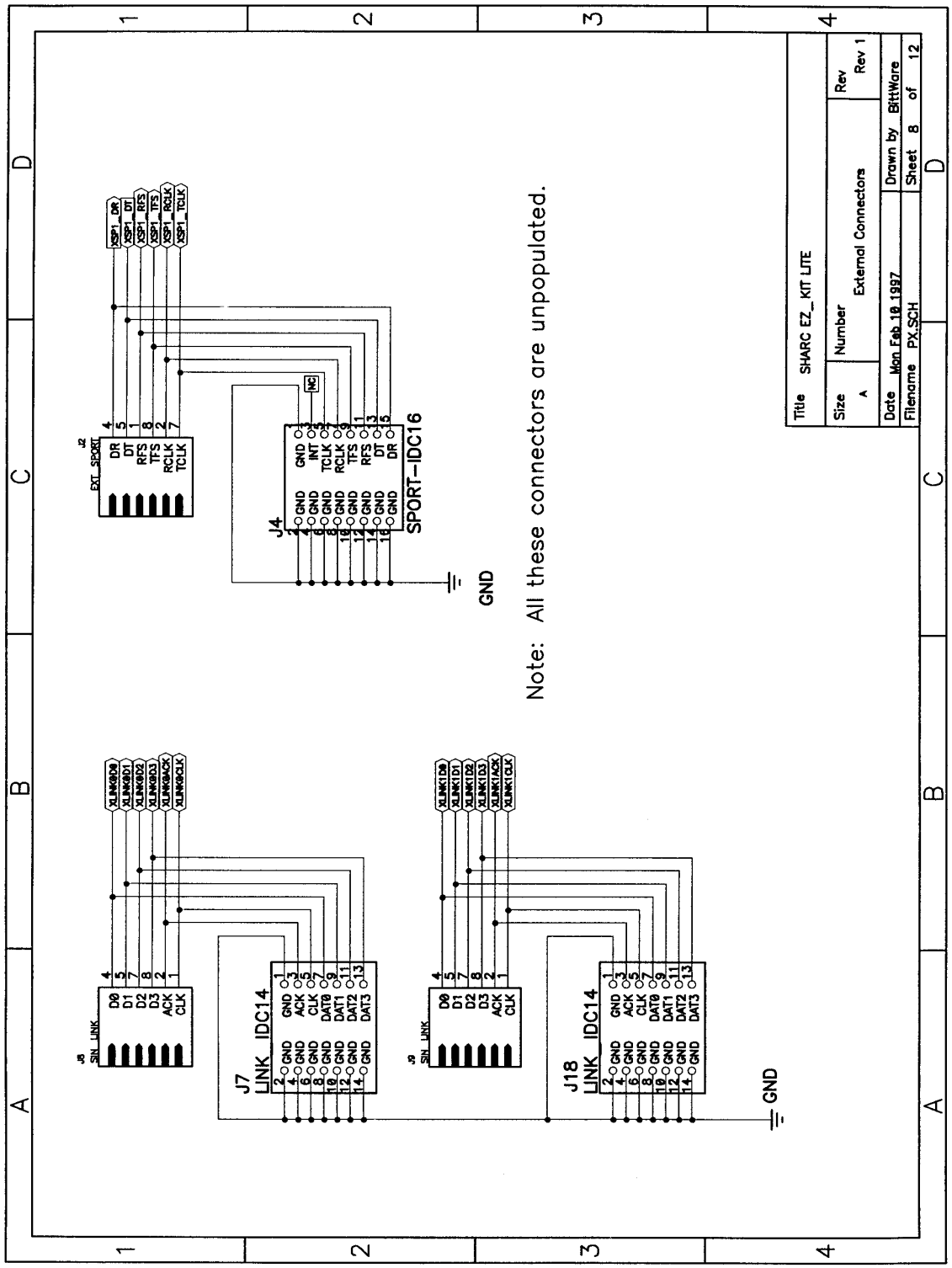

EPROM Jumper Selection Chart
U7 JP4 JP3 JP2 JP1
27C256 IN IN
27C512 IN IN
27C010 IN IN
27C020 IN IN
27C040 IN IN
27C080 IN IN
** Default Mode **
SHARC Boot Mode Selection
Mode JP5 JP6
EPROM IN
HOST IN IN
LINK IN
** Default Mode **
*Note : The default modes are hardcoded on the board.
JP1–JP6 will not be installed on the board. In order to gain
access to other modes the hardcoded traces will have to
be modified by the user.
RP3:11 RP3:12
10k 10k
EBOOT
LBOOT
JP5 JP6 JP4 JP3 JP2 JP1
GND
AT27Cxxx
U7
U1
QS74FCT245ATSO
NOTE: These are buffered 40Mhz clocks.
Note: This clock is 40Mhz.
Y1
System Clock
TRIST_OSC
Title SHARC EZ_ KIT LITE
Size A
Number EProm & Clock Buffer
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 3 of 12

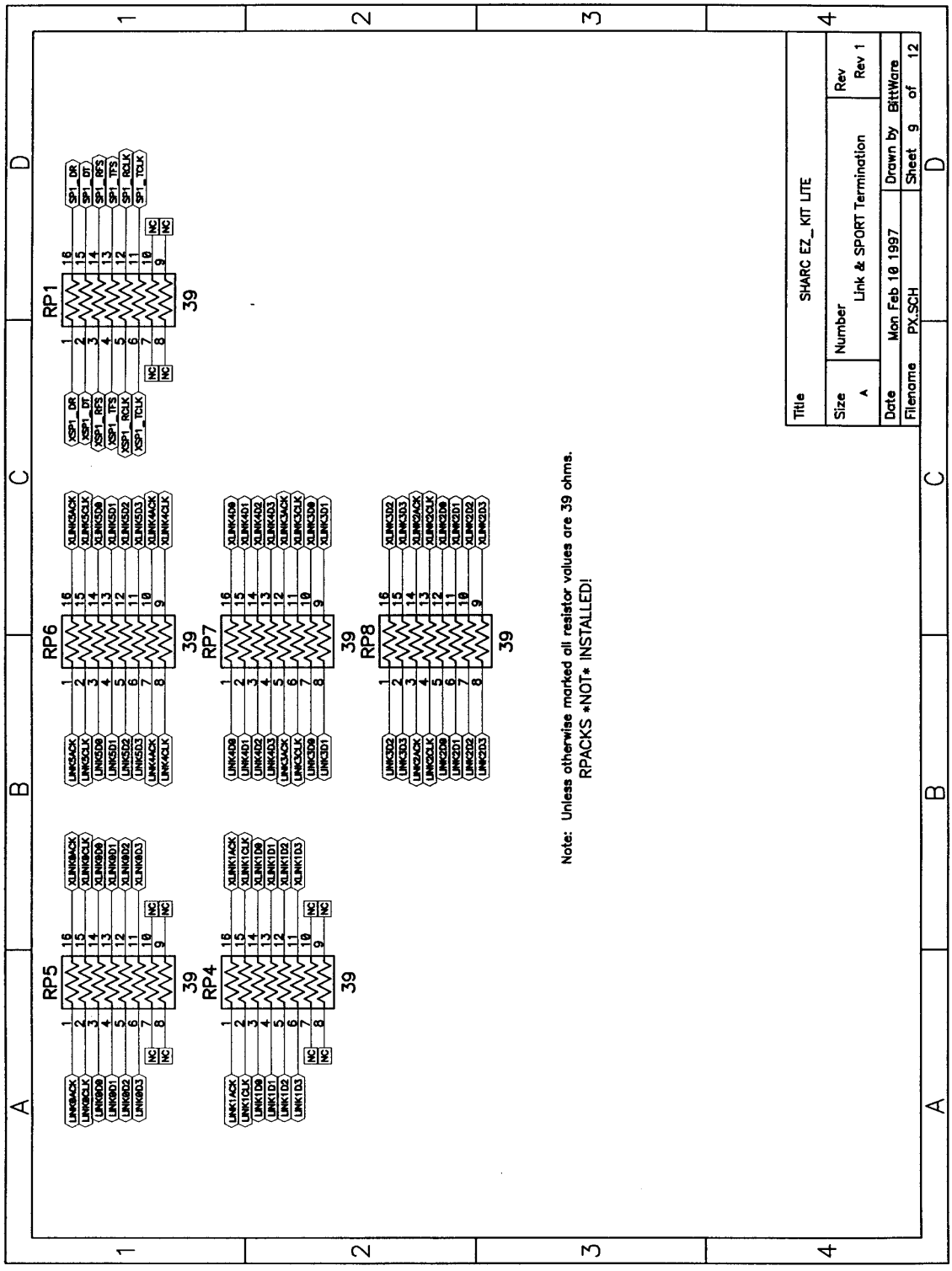

P1
DB9RF
GND
C20
0.1uF
C18
0.1uF
VDD
C7
27pf
C8
47pf
1Meg
R1
18.432 Mhz
X1
R2
1.5K
CRYSTAL
C17
0.1uF
C19
0.1uF
U5
ADM232AARN
U6:A
74HC14
U4
PC16550D
UART
16V8
U3
NOTE: The shell on the DB9 connector is connected to the GND plane.
Title SHARC EZ_KIT LITE
Size A
Number UART & UART Logic
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 4 of 12

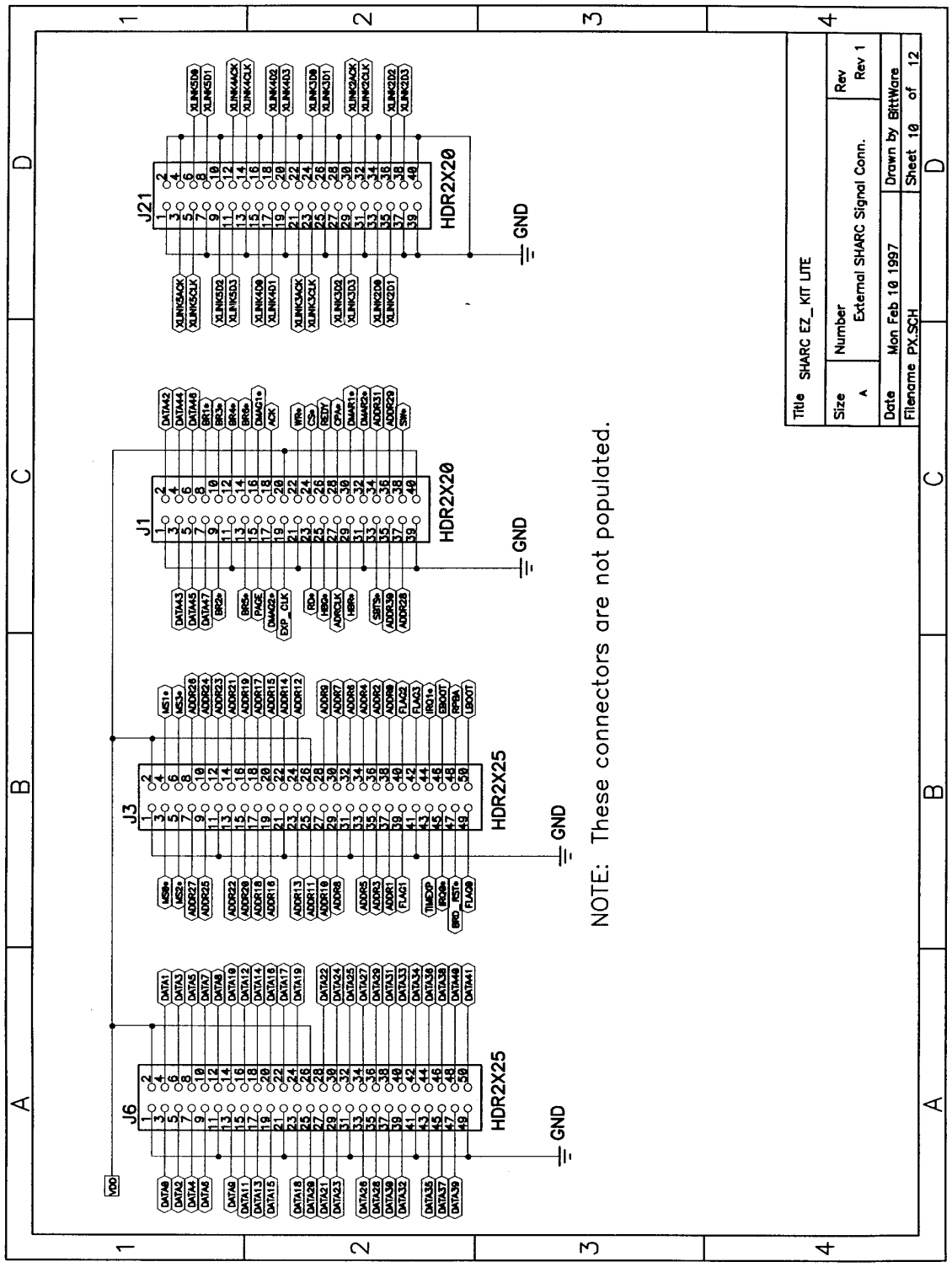

NOTE: The board also contains wire terminals so the user can provide power without using the DC connector.
DC-CONN P2
DCVOLT
GND
GND
EXTPOW
D5
LM7805CTB VR1
VI
GND
VO
3
2
1
C40 10uf
C39 0.01uf
C6 0.01uF
C56 10uF
NOTE: The heat sink of the LM7805 is thermally and physically connected to the GND plane.
GND
FB1 125
R11 1.6
C37 10uf
C44 0.01uf
AGND
VDD
*Note: J13 - J17 are not populated. These are used for connecting up an external LED. Please use LED with integrated resistor.
J13 Power
J14 FLAG0
J15 FLAG1
J16 FLAG2
J17 FLAG3
C1
C2
R6 300
R7 300
R8 300
R12 300
R16 300
D1 Power
D2 FLAG0
D3 FLAG1
D4 FLAG2
D6 FLAG3
LED - SOT23
BUF_F0
BUF_F1
BUF_F2
BUF_F3
Title SHARC EZ_KIT LITE
Size A
Number
Power
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 5 of 12

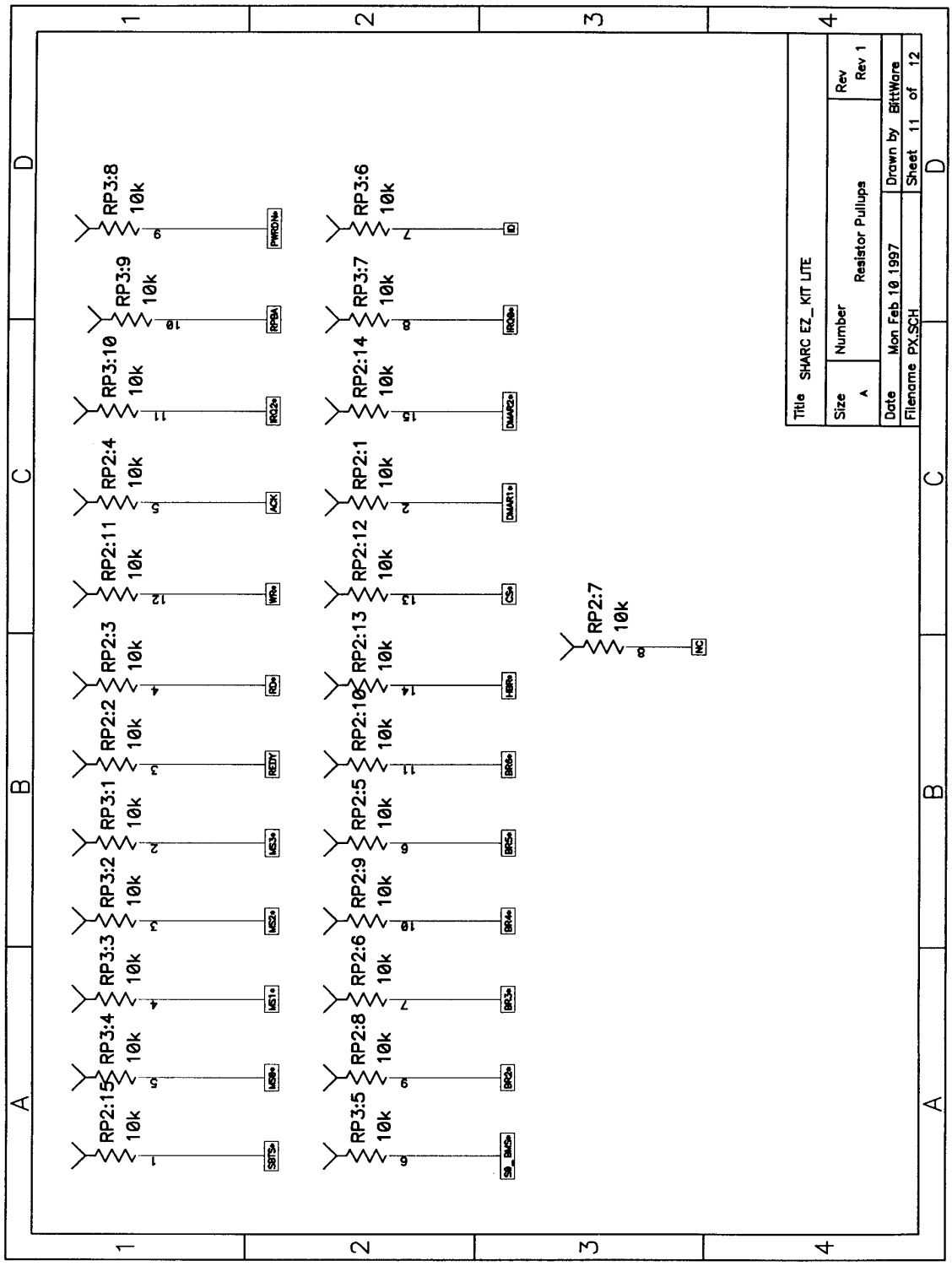

TP4 TP5 TP6 TP3 TP1 TP2
U9
AD1847JP
SCLK
SDFS
SDI
SDO
BM
TSO
TSI
XTAL1I
XTAL1O
XTAL2I
XTAL2O
XCTL1
XCTL0
RESET
PWRDWN
CLKOUT
L_LINE1
R_LINE1
L_LINE2
R_LINE2
L_AUX1
R_AUX1
L_AUX2
R_AUX2
L_OUT
R_OUT
VREF
VREFI
LFILT
RFILT
AVDD
AGND
LIN
RIN
LOUT
ROUT
VREF
C34 10uF
C32 10uF
C31 0.1uF
C35 1uF 16V
C36 1uF 16V
AGND
X2 16.9344MHz CRYSTAL
X3 24.576MHz CRYSTAL
C23 20pF
C24 20pF
C25 20pF
C26 20pF
GND
R5 10K
Pulled Up
SPRO_CLK
SPRO_RFS
SPRO_DT
SPRO_DR
SPRO_RST*
FLAG0
PWRDN*
Title SHARC EZ_KIT LITE
Size A
Number Audio Interface
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 6 of 12

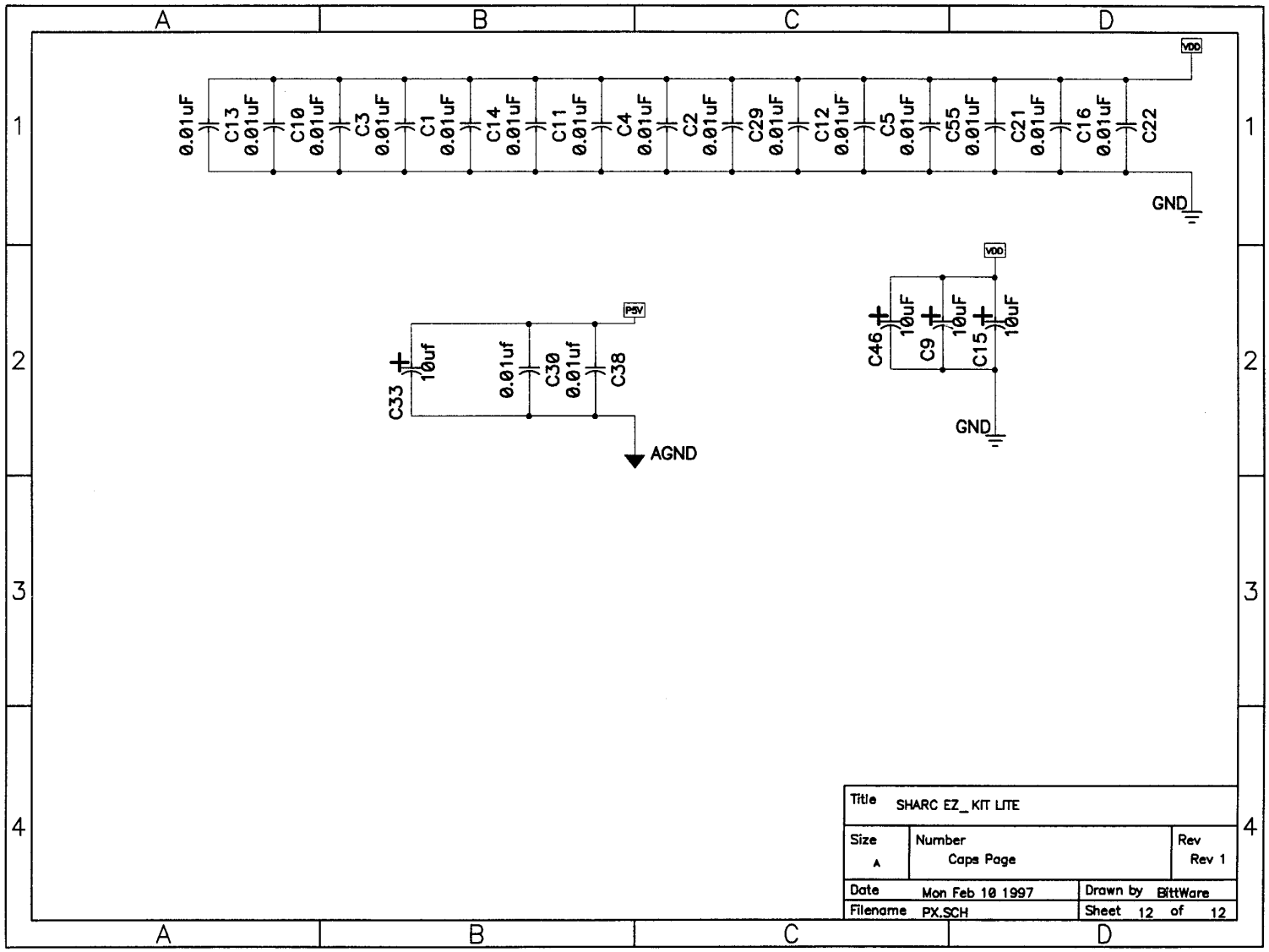

Title SHARC EZ_KIT LITE
Size A
Number Audio Interface
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 7 of 12
NOTE: J19 and J20 are not installed on the board.
MIC/LINE in Selection Jumpers
Please see note on this page.
AGND
U10:A SSM2135
U10:B SSM2135
J19
J20
J22
J23
JP8
JP9
C41
C42
C43
C45
C47
C48
C49
C50
C51
C52
C53
C54
R9
R10
R13
R14
R15
R17
R18
R19
R20
R21
LINE
RLINE
LMIC
RMIC
LOUT
ROUT


Note: All these connectors are unpopulated.
J2
EXT_SPORT
DR
DT
RFS
TFS
RCLK
TCLK
J4
SPORT-IDC16
GND
J5
SIN_LINK
D0
D1
D2
D3
ACK
CLK
J7
LINK_IDC14
J9
J18
Title SHARC EZ_KIT LITE
Size A
Number
External Connectors
Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 8 of 12


RP1
RP4
RP5
RP6
RP7
RP8
39
Note: Unless otherwise marked all resistor values are 39 ohms.
RPACKS *NOT* INSTALLED!
Title
SHARC EZ_ KIT LITE
Size
A
Number
Link & SPORT Termination
Rev
Rev 1
Date
Mon Feb 10 1997
Drawn by
BittWare
Filename
PX.SCH
Sheet
9
of
12


J6
HDR2X25
J3
HDR2X25
J1
HDR2X20
J21
HDR2X20
GND
NOTE: These connectors are not populated.
Title SHARC EZ_ KIT LITE
Size A
Number External SHARC Signal Conn.
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 10 of 12


RP3:8 10k
RP3:9 10k
RP3:10 10k
RP2:4 10k
RP2:11 10k
RP2:3 10k
RP2:2 10k
RP3:1 10k
RP3:2 10k
RP3:3 10k
RP3:4 10k
RP2:15 10k
RP3:6 10k
RP3:7 10k
RP2:14 10k
RP2:1 10k
RP2:12 10k
RP2:13 10k
RP2:10 10k
RP2:5 10k
RP2:9 10k
RP2:6 10k
RP2:8 10k
RP3:5 10k
RP2:7 10k
PWRON
RP3:9
RP3:10
ACK
WR
RD
RDY
MS3
MS2
MS1
MS0
SBTS
DMAR2
DMAR1
CS
HBR
BR6
BR5
BR4
BR3
BR2
SA_BM
NC
Title SHARC EZ_KIT LITE
Size A
Number
Resistor Pullups
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 11 of 12


A
B
C
D
1
2
3
4
VDD
0.01uF
C13 0.01uF
C10 0.01uF
C3 0.01uF
C1 0.01uF
C14 0.01uF
C11 0.01uF
C4 0.01uF
C2 0.01uF
C29 0.01uF
C12 0.01uF
C5 0.01uF
C55 0.01uF
C21 0.01uF
C16 0.01uF
C22
GND
PSV
C33 10uf
0.01uf
C30 0.01uF
C38
AGND
VDD
C46 10uF
C9 10uF
C15 10uF
GND
Title SHARC EZ_KIT LITE
Size A
Number Caps Page
Rev Rev 1
Date Mon Feb 10 1997
Drawn by BittWare
Filename PX.SCH
Sheet 12 of 12

8.3. PAL Equations

Equations used to program the 16v8 PAL device U3.

```
"  PSTATE.ABL
"  ABEL code created by Visual Software Solution's StateCAD Version 3.00
"  Thu Feb 06 09:23:40 1997

MODULE PSTATE

DECLARATIONS
"clock name
   CLK PIN 1;

"Input variables
   RD PIN 6;
   RESET PIN 3;
   SEL PIN 5;
   WR PIN 4;
   ROE PIN 11;

"Output variables
   ACK PIN 12 ISTYPE 'com';

"Logic variables
   UARTRST PIN 18 ISTYPE 'com';

"State variables
   SV0  PIN 13 ISTYPE 'reg';
   SV1  PIN 14 ISTYPE 'reg';
   SV2  PIN 19 ISTYPE 'reg';
   URD  PIN 16 ISTYPE 'reg';
   USEL  PIN 17 ISTYPE 'reg';
   UWR  PIN 15 ISTYPE 'reg';

"Vectors
DECLARATIONS
   SV=[
      USEL,
      URD,
      UWR,
      SV2,
      SV1,
      SV0
   ];
```

```
"State Register assignment
DECLARATIONS
    sreg=[ SV0,SV1,SV2,URD,USEL,UWR];

EQUATIONS
    sreg.clk=CLK;
    !sreg.oe = ROE;
    ACK.oe = !SEL;


DECLARATIONS
    STATE0=[1, 1, 1, 1, 1, 1];
    STATE1=[0, 0, 0, 1, 0, 1];
    STATE2=[1, 0, 0, 1, 0, 1];
    STATE3=[0, 0, 0, 0, 0, 1];
    STATE4=[1, 0, 0, 0, 0, 1];
    STATE5=[0, 0, 0, 1, 0, 0];
    STATE6=[1, 0, 0, 1, 0, 0];
    STATE7=[0, 1, 0, 1, 0, 0];
    STATE8=[0, 1, 0, 0, 0, 1];
    STATE9=[1, 1, 0, 0, 0, 1];
    STATE10=[0, 0, 1, 0, 0, 1];
    STATE11=[1, 0, 1, 0, 0, 1];
    STATE12=[0, 1, 1, 0, 0, 1];
    STATE13=[1, 1, 1, 0, 0, 1];
    STATE14=[1, 1, 0, 1, 0, 0];
    STATE15=[0, 0, 1, 1, 0, 0];
    STATE16=[1, 0, 1, 1, 0, 0];
    STATE17=[1, 1, 0, 1, 0, 1];
    STATE18=[0, 0, 1, 1, 0, 1];
    STATE19=[0, 1, 0, 1, 0, 1];

EQUATIONS
    WHEN (!(
          (sreg.FB==STATE0)
       # (sreg.FB==STATE1)
       # (sreg.FB==STATE2)
       # (sreg.FB==STATE3)
       # (sreg.FB==STATE4)
       # (sreg.FB==STATE5)
       # (sreg.FB==STATE6)
       # (sreg.FB==STATE7)
       # (sreg.FB==STATE8)
       # (sreg.FB==STATE9)
       # (sreg.FB==STATE10)
       # (sreg.FB==STATE11)
```

```
       # (sreg.FB==STATE12)
       # (sreg.FB==STATE13)
       # (sreg.FB==STATE14)
       # (sreg.FB==STATE15)
       # (sreg.FB==STATE16)
       # (sreg.FB==STATE17)
       # (sreg.FB==STATE18)
       # (sreg.FB==STATE19)
   ) &
       ( RESET )) THEN {
       [sreg] := [STATE0];
   }

   WHEN (!(
         (sreg.FB==STATE0)
       # (sreg.FB==STATE1)
       # (sreg.FB==STATE2)
       # (sreg.FB==STATE3)
       # (sreg.FB==STATE4)
       # (sreg.FB==STATE5)
       # (sreg.FB==STATE6)
       # (sreg.FB==STATE7)
       # (sreg.FB==STATE8)
       # (sreg.FB==STATE9)
       # (sreg.FB==STATE10)
       # (sreg.FB==STATE11)
       # (sreg.FB==STATE12)
       # (sreg.FB==STATE13)
       # (sreg.FB==STATE14)
       # (sreg.FB==STATE15)
       # (sreg.FB==STATE16)
       # (sreg.FB==STATE17)
       # (sreg.FB==STATE18)
       # (sreg.FB==STATE19)
   ) &
       !( RESET )) THEN {
       [sreg] := [STATE0];
   }

state_diagram sreg;

state STATE0:
   ACK=1;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !SEL ) THEN STATE1;
```

```
       IF ( SEL ) THEN STATE0;
state STATE1:
   ACK=1;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE2;
state STATE2:
   ACK=1;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !SEL & RD & WR  #  !SEL & !WR & !RD ) THEN STATE2;
      IF ( SEL ) THEN STATE0;
      IF ( !SEL & !RD & WR ) THEN STATE3;
      IF ( !SEL & !WR & RD ) THEN STATE5;
state STATE3:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE4;

state STATE4:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE8;
state STATE5:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE6;
state STATE6:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE7;
state STATE7:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE14;
state STATE8:
   ACK=0;
   IF ( RESET ) THEN STATE0;
   ELSE
      IF ( !0) THEN STATE9;
state STATE9:
```

```
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE10;
state STATE10:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE11;
state STATE11:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE12;
state STATE12:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE13;
state STATE13:
    ACK=1;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE19;

state STATE14:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE15;
state STATE15:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE16;
state STATE16:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE17;
state STATE17:
    ACK=0;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE18;
state STATE18:
```

```
    ACK=1;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE19;
state STATE19:
    ACK=1;
    IF ( RESET ) THEN STATE0;
    ELSE
        IF ( !0) THEN STATE0;

"Logic Equations
EQUATIONS
UARTRST =  RESET ;
END PSTATE
```

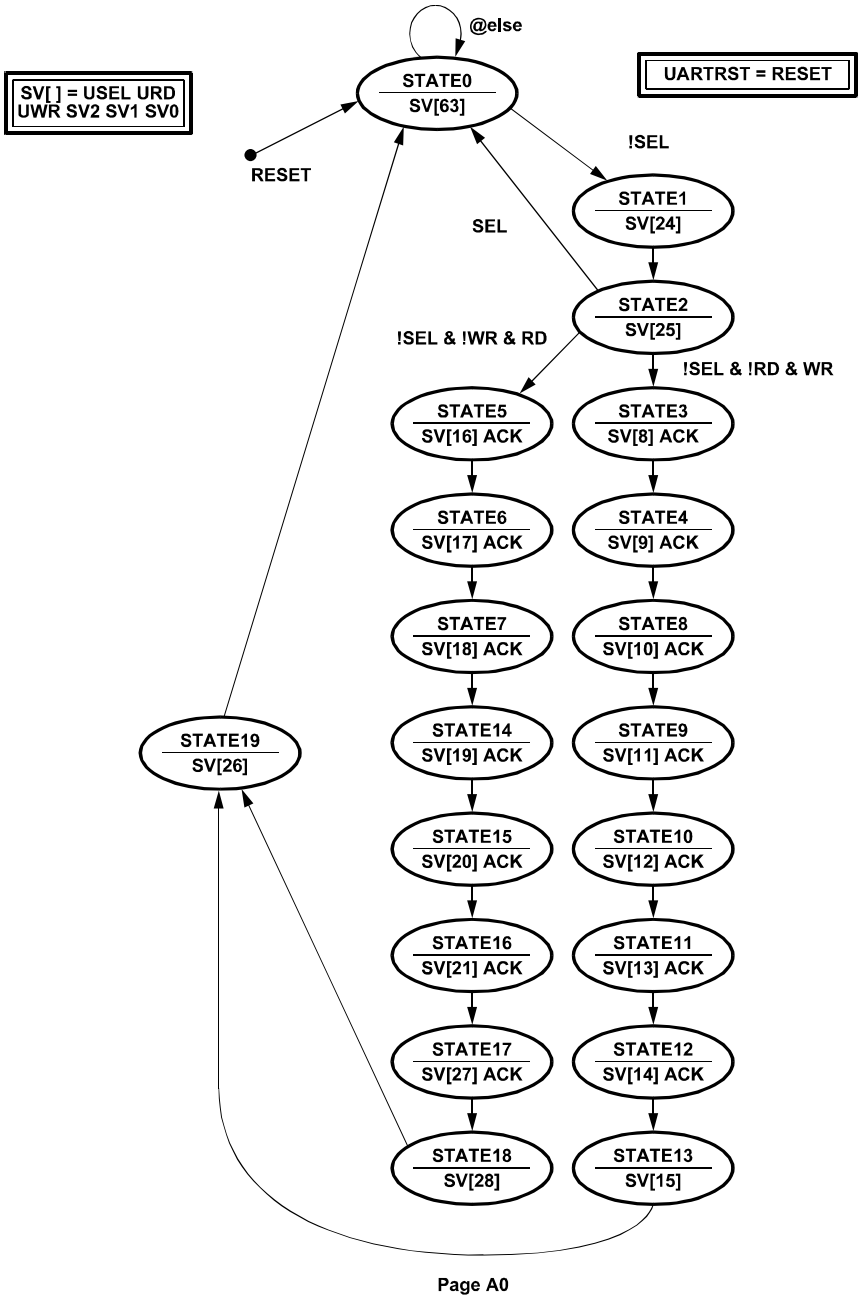


Figure 8-1 State Diagram for PAL U3

APPENDIX

RESTRICTIONS

The following restrictions apply to release 2.0 of the ADSP-21061 EZ-KIT Lite board. For information on any ADSP-21061 silicon anomalies, see the anomaly sheet that accompanied this product.

- Breakpoints set in the last three instructions of a DO-loop are allowed, but cause your code to run incorrectly.
- Breakpoints set after a delayed branch instruction and before the branch occurs causes your code to run incorrectly.
- Using the single stepping function  steps through a delayed branch instruction and the last three instructions of a DO-loop.
- The host loses contact with the monitor while the user program is running if the user program disables the UART interrupt or changes the UART interrupt vector.
- The host loses contact with the monitor while the program is running and in an ISR when nesting is turned off.
- The host loses contact with the monitor while the program is running and in the timer ISR, provided the highest priority timer vector is used.
- This current version of the EZ-KIT Lite software does not let you view hardware stack information.
- Do not use the reset button while the debugger is open unless the debugger requests you to. This will cause the debugger to crash.
- The IMDW0 bit in the SYSCON register must be set to 1 to keep communication with the host. The IMDW0 bit determines if data accesses made to block 0 are 48-bit, three-column accesses (1) or 32-bit, two-column accesses (0). The monitor program requires three-column data accesses to memory block 0.

If The IMDW0 bit is set to 0, the monitor accesses incorrect memory locations within block 0. (See *ADSP-2106x User's Manual* for further discussion of IMDW0).

- The setting of IMDW0 will have no effect on C programming as long as RND32 is not set for 40-bit, floating-point precision.
- Do not run more than one ADSP-21061 EZ-KIT Lite session in the debugger at any one time. You may run an EZ-KIT Lite session and a simulator or ICE session at the same time, or you can open two debugger interfaces to run more than one EZ-KIT Lite session.
- User should not remove files 61_kernelformatDM32.hex and 61_kernelformatDM32.hex from their current location, otherwise the new monitor will not be download to the target. If this occurs, there will no communication between the target board and the host.

INDEX