RIX 51

USER'S MANUAL

January 2000

RIGEL CORPORATION
PO Box 90040
Gainesville, Florida 32607
(352) 373-4629
FAX (352) 373-1786
www.rigelcorp.com

Table of Contents

1. INTRODUCTION 1

1.1 RIX51 ARCHITECTURAL OVERVIEW 1
1.1.1 Rix51 - A Smart Evaluation Board 1
1.1.2 The Two Phases of Operation 1
1.1.3 Saving and Switching Between Stored Programs 2
1.1.4 Modifying the Configuration Bytes to Evaluate the Performance of the Microcontroller 2
1.2 HARDWARE OVERVIEW 2
1.3 SOFTWARE OVERVIEW 3
1.3.1 The Rix51 Utility Software 3
1.3.2 The Rix51 Utility Software 3
1.3.3 Third Party Software 3
1.3.4 A*p*BUILDER 3
1.4 PACKAGE CONTENTS 4

2. SYSTEM REQUIREMENTS AND SOFTWARE INSTALLATION 5

2.1 SYSTEM REQUIREMENTS 5
2.2 RIX51 UTILITY SOFTWARE AND READS51 INSTALLATION 5
2.3 THIRD PARTY SOFTWARE 5
2.4 A*P*BUILDER INSTALLATION 5

3. QUICK START TUTORIAL 6

3.1 BOARD SETUP 6
3.2 RIX51 UTILITY SETUP 6
3.3 START A SESSION 6
3.4 DOWNLOADING AND RUNNING A FILE 6
3.5 CHANGING THE CONFIGURATION BYTES 6
3.6 PROGRAMMING FLASH 6
3.7 MANUALLY LOADING A FLASH FILE TO RAM 6
3.8 AUTOMATICALLY LOADING AND RUNNING A FLASH FILE 7
3.9 AUTOMATICALLY RUNNING THE CURRENT PROGRAM IN RAM UPON RESET 7

4. THE RIX51 BOARD 8

4.1 OVERVIEW 8
4.2 POWER 8
4.2.1 J 7, 9VAC Power Jack 8
4.2.2 J6, 9VAC Terminal Block 9
4.2.3 J8, 5 V Terminal Block 9
4.2.4 Connecting the Power to the Board 9
4.2.5 Troubleshooting the Power Connection 9
4.3 SERIAL PORTS 9
4.3.1 P1, Internal UART 10
4.3.2 P2, Internal UART for the 251TB 10
4.3.3 P3, External UART 10
4.3.3 Baud Rates 10
4.3.4 Connecting the Serial Port to the Board 10
4.3.5 Troubleshooting the Serial Port Connections 10
4.4 LEDS 11
4.4.1 Power LED 11
4.4.2 Status LEDs 11
4.4.3 Port 1 LEDs 11
4.5 RESET BUTTON 11
4.6 CHANGING MICROCONTROLLERS ON THE EVALUATION BOARD 11

5. JUMPER SELECTIONS 12

5.1 DEFAULT JUMPER SETTINGS 12
5.2 J2, EA# 12

5.3 J12, VPP 12
5.4 J5, WP# 12
5.5 J2, CFG1, CFG0, AND EDIS 12
5.6 S6, A17 12
5.7 J4, A15 12
5.8 S1, UART INT 12

6. HEADERS 13

6.1 J10, SYSTEM HEADER 13
6.2 J13, RIO HEADER 14
6.3 J9, EXPANSION HEADER 14
6.4 J11 15

7. CONFIGURATION OPTIONS 16

7.1 MEMORY MAP 16
7.2 BUS MODE 16
7.3 SOURCE VERSUS BINARY MODE 16
7.4 INTERRUPT MODE 16
7.5 UCON AND EMAP# 16
7.6 WAIT STATES 16
7.7 EXTENDED ALE 16
7.8 WSB / EDF# 17

APPENDIX A: OPERATING NOTES FOR THE RIX51 18

A.1 THE RIX51 BOARD RESET PROCESS AND OPTIONS 18
A.2 THE SETUP AND RUN PHASES 18
A.3 BUS CONFIGURATION DURING THE RUN PHASE 19
A.4 MEMORY MAP DURING THE RUN PHASE 19
A.5 RIX51 UTILITY SERIAL PORT OPTIONS 19
A.6 INDICATORS 20

APPENDIX B: A*P*BUILDER 21

B.1 PRODUCT SELECTION TOOL 21
B.2 ON-LINE TECHNICAL REFERENCE 21
B.3 PROGRAMMING AIDE 21
B.4 QUESTIONS & ANSWERS 21
B.5 HILITES 21
B.6 PERIPHERAL EDITOR 21
B.7 REGISTER EDITOR 22
B.8 INSTRUCTION EDITOR 22
B.9 EDITOR 22

APPENDIX C: PARTS LISTS 23

RIX51 BILL OF MATERIALS 23

APPENDIX D: TOP OVERLAY AND SCHEMATICS 25

1. INTRODUCTION

We recommend that you read section 1.1 in its entirety before going through the quick start tutorial given in section 3.

1.1 Rix51 Architectural Overview

1.1.1 Rix51 - A Smart Evaluation Board

The Rix51 evaluation board was designed with the following key criteria:

1. Capable of placing the microcontroller in all its possible configurations without an excessive number of jumpers or switches. These include the page and non-page bus modes, the four different memory maps, and various wait states and control signal timings.
2. Support many different software vendors' monitor programs as well as user programs. Several monitor programs and user programs co-exist on the board, and are easily updateable. The board may be set (by a simple slide switch) to load one of the stored programs and run upon reset.
3. Capable of being customized for user applications or third-party vendors requirements.

The Rix51 fulfills these criteria by a two-phase process. The first phase initializes the RAM. The second phase runs the downloaded or selected FLASH program now in RAM. During the second phase, the entire memory of the microcontroller (except memory-mapped inputs and outputs) is mapped into RAM. This facilitates the inspection and modification of the configuration bytes (UCONFIGx). Following the setup, a ***Self-Induced Reset*** initializes the microcontroller to run the code in RAM. The Rix51 reads the same configuration during the self-induced reset and configures the bus mode (page versus nonpage) and the memory map (64k, 128K, or 256K) accordingly. The capability to inspect the configuration bytes and modify the bus and memory map accordingly makes the Rix51 a smart board.

The Rix51 hardware is built around the following two key components to satisfy the design criteria.

1. A CPLD (Complex Programmable Logic Device) to replace arrays of jumpers to implement different bus architectures and allow customization for specific applications.
2. A FLASH to store the various monitor and user programs.

A more detailed discussion of the operating phases is given next.

1.1.2 The Two Phases of Operation

The Rix51 enters an initial ***Setup Phase*** when reset. The Setup Phase determines the desired operating mode, the **Interactive** mode or the **Autoload** mode, by inspecting the state of the slide switch. In the autoload mode (slide switch in "AUTO" position), the board loads the jumper specified program into RAM and enters the ***Run Phase***.

In the interactive mode (slide switch in "MAN" position), the board expects to communicate with the Rix51 Utility software. The user may interactively download programs, inspect and modify the configuration bytes (UCONFIGx), and manage the FLASH. The interactive mode is terminated either by a reset while the slide switch is in the "AUTO" position, or by a "run" command from the Rix51 Utility.

In the Run Phase, the entire memory of the 8xC251, including the configuration bytes, is mapped to RAM. Before the Run Phase commences, the Rix51 inspects the configuration bytes (UCONFIGx) and sets the bus mode (page versus nonpage) and the memory map (256K, 128K, or 64K) accordingly. The Run Phase is initiated by a ***Self-Induced Reset***. This forces the 8xC251 to read the configuration bytes and start executing code from address FF:0000, now all in RAM.

1.1.3 Saving and Switching Between Stored Programs

Up to four user programs or monitors may be placed in FLASH. These programs are referred to as ***FLASH Files***, since they are placed in nonvolatile memory and may be selected to automatically be loaded and run with a self-induced reset. Each FLASH file is 64K. It is loaded into the FF: region of the 8xC251 microcontroller. The FLASH file to be automatically loaded and run is determined by the jumpers CFG1 and CFG0 as shown in Appendix A. The user may place four different monitor programs in FLASH. Then, the selected one is run upon the self-induced reset. Thus, the board appears to be configured for the software vendors development environment. The Rix51 Utility allows the user to manage the FLASH files. This utility allows the user to place programs in FLASH. This feature facilitates monitor updates from the software vendors (e.g. released on their web site) to conveniently be implemented in the field.

1.1.4 Modifying the Configuration Bytes to Evaluate the Performance of the Microcontroller

During the Run Phase, the entire memory is mapped to RAM. The program downloaded into RAM or copied into RAM from a FLASH file will typically include its own configuration bytes. The board may run this program as is, or the user may inspect the configuration bytes and modify them using the Rix51 Utility software. (The protocol for the Rix51 Utility software is available for software vendors who would like to implement this functionality into their environments. Please contact Rigel Corporation for technical information). For example, the bus mode may be switched to the page or nonpage mode. Similarly, various wait states and control signal timings may be changed. Then the performance of the microcontroller may be evaluated under these conditions. For example, it may be observed that the page mode runs faster than the nonpage mode, and that reducing the wait states results in faster program execution. Refer to the Quick Start Tutorial for an example.

1.2 Hardware Overview

The Rix51 evaluation board was designed for the 8XC51, and 8XC251TB processors, but can accept a variety of MCS® 51 based processors from Intel.   Any processor used with the board must be fully pin-to-pin compatible with the MCS 51 44 pin PLCC package.

The Rix51 evaluation board contains the following major components:

- 51FA/251 processor running at 24MHz
- An on-board switching 5VDC power regulator
- Three serial ports connected to female DB9s
- A 1.8432Mhz TTL clock for the external UART
- A CPLD for address and memory decoding
- 256K 5 volt FLASH
- Two 128K SRAMs
- Status, Power, and Page LEDs for immediate visual feedback on the board

- A Reset switch
- A 50-pin header carrying all system signals
- A 46-pin header for connecting to an external I/O board
- A 42-pin header for connecting to Rigel's RIO-MP or RIO-JR prototyping I/O boards
- Mounting holes in the corners of the board

1.3 Software Overview

The Rix51 evaluation board comes with four types of software. Rigel's Rix51 Utility software and the READS51 IDE, compilers/assemblers from third party vendors, and Intel's A*p*BUILDER software.

1.3.1 The Rix51 Utility Software

The Rix51 Utility software from Rigel Corporation is a Windows 95/98/NT program. It may be used in an interactive manner to download and run programs, to inspect and modify the configuration bytes. It is also the primary means to manage the on-board FLASH. Please visit Rigel's web site www.rigelcorp.com for any future updates to this software.

1.3.2 READS51

The READS51 is Rigel's Integrated Development System for the 80C51 microcontroller. Since the 8xC251 microcontroller executes 80C51 code as is, it may be used to develop software for the 8xC251 as well. READS51 v4.0 contains an absolute assembler, a relative assembler and OMF-51 linker, a library utility, a SmallC-compatible C compiler, and many hardware-oriented demonstration programs. The software is not limited in any way. It is free of charge to customers who purchase an evaluation board. Please visit Rigel's web site www.rigelcorp.com for any future updates to this software.

1.3.3 Third Party Software

The Rix51 evaluation board package also comes with software by third party vendors. Please refer to the vendors documentation for product overview, and installation instructions. Note that the Rix51 allows you to place software vendor's monitors in FLASH and run these upon reset. We suggest that you visit their web sites to download any new software or updates to existing software. These updates may be downloaded into the Rix51 FLASH using the Utility software to bring your system up to date.

1.3.4 A*p*BUILDER

A*p*BUILDER is a powerful tool for programming Intel's embedded controllers. It was developed by Intel's Application Engineers to reduce the learning curve associated with new products, and shorten total design time. With A*p*BUILDER you can develop peripheral initialization code, without prior knowledge of the architecture, with just a click of a mouse. A*p*BUILDER also documents the instruction set and accesses on-line hypertext reference material. The A*p*BUILDER software package includes: an on-line A*p*BUILDER introduction, Quick start, and help system. For more information about A*p*BUILDER see the appendices or the A*p*BUILDER software.

1.4 Package Contents

The Rix51 Prerelease Evaluation Package includes only part of the following list.
(This list is intended as an overview of the production package contents.)

- Rix51 Evaluation Board
- Rix51 User's Manual
- Rix51 Utility Software
- READS400 Integrated Development Environment
- A*p*BUILDER Software
- Hypertext Data Books/Sheets on the 8xC51FA, 8xC251Sx and 8xC251Tx.
- Data Sheets for IC's used on the board
- Adobe Reader 3.0
- Application Notes
- Third Party Software
- 9 VAC Transformer.
- Serial Cable

2. SYSTEM REQUIREMENTS AND SOFTWARE INSTALLATION

2.1 System Requirements

The software included in this package requires Microsoft Windows 95, 98 or NT, running on a IBM PC or compatible with a 486 or faster processor, 4MB RAM, a VGA display, CD ROM drive, and 2 MB of hard drive space.

2.2 Rix51 Utility Software and READS51 Installation

The Rix51 Utility Software is located on the CD-ROM included with the Rix51 User's Manual. This utility is needed to manage the FLASH. Once you place the software vendors monitor program in FLASH, you need not run the Rix51 utility. Alternatively, you may use the Rix51 utility to interactively download and run programs, or to inspect and modify the configuration bytes. The utility also includes a diagnostics routine to quickly test the Rix51 board.

Use the following steps to install the Rix51 Utility and READS51:

1. Insert the Rigel CD-ROM
2. Select **Rigel | Rix 51** and execute the program "SetupRix51.exe" and then follow the on-screen instructions.
3. Select **Rigel | Rigels51** and execute the program "SetupReads400.exe" and then follow the on-screen instructions.

2.3 Third Party Software

Please refer to the documentation from the third party vendors for system requirements, installation instructions, and user's instructions.

2.4 A*p*BUILDER Installation

Use the following steps to install the A*p*BUIlDER software:

1. Insert the Rigel CD-ROM
2. Look in the Intel directory and copy the zipped file, APBLD221.ZIP on the CD into a new directory.
3. Unzip the file.
4. Run setup.exe from Windows

Quick Tutor provides an efficient introduction to the A*p*BUILDER.

Note

To install the A*p*BUILDER software, you should have the Windows* operating system on your local PC drive. Problems may result from network versions of Windows that have read-only restrictions. Several software applications write .ini files to the Windows directory and will not install on the read-only network version.

3. QUICK START TUTORIAL

3.1 Board Setup

Before you apply the power:

1. Connect the serial cable to either the internal UART port (marked "P1 Internal (config.)" on the bottom of the board) or the external UART port (marked "P3 external" on the bottom of the board).
2. Connect the other end of the serial cable to an available PC port.
3. Slide switch to MAN
4. Check that the EA# jumper is present and the S6 jumper is in the left position.
5. Power the board. (Either 9VAC or regulated 5VDC).

3.2 Rix51 Utility Setup

1. Install and run the Rix51 Utility
2. Select the PC port. Note that the dialog controls are disabled unless you select a valid and available port.
3. Specify whether the internal or external UART port of the board is connected to the serial cable. The baud rate is automatically selected for the internal or external port.

3.3 Start a Session

1. Press RESET on the board.
2. Click on the Diagnostics button in the dialog and observe the message.

3.4 Downloading and Running a File

1. Click the "Download" button.
2. Select the file "\Demos\Bin.hex"
3. The status bar reports the progress and result of the downloaded.
4. Click the "Run" button. The program increments P1, as observable by the LEDs.

3.5 Changing the Configuration Bytes

1. If not already done, perform all the steps in Section 3.4.
2. Press Reset.
3. Click the "UCONFIG" button.
4. Note that the "Non-Page" option is selected and that UCONFIG0=FE.
5. Select the "Page" option and observe that UCONFIG0=FC.
6. Click OK to close the configuration bytes dialog.
7. Click the "Run" button. Note that the status LED S1 is now lit. (S1 is on in Page mode, and off in Non-page mode.)

3.6 Programming FLASH

1. Press Reset.
2. Click the "Program" button.
3. Select FLASH File 0.
4. Click OK to close the "Select FLASH File" dialog.
5. Select the file "Demos\Bin.hex"
6. Make sure the VPP jumper is present.
7. Click OK to start programming. The result of the operation is reported.

3.7 Manually Loading a FLASH File to RAM

1. Press Reset on the board.
2. Click the "Copy to RAM" button.

3. Select FLASH file 0.
4. The file is copied to RAM. You may click the "Run" button to run the program.

3.8 Automatically Loading and Running a FLASH File

1. Remove jumpers CFG1 and CFG0 to select FLASH file 0. You may specify other FLASH files by inserting or removing the CFG1:CFG0 jumpers. (See Section 5.5)
2. Remove jumper EDIS to enable FLASH to RAM transfer.
3. Move the slide switch to the "AUTO" position.
4. Press Reset to run the program.

3.9 Automatically Running the Current Program in RAM Upon Reset

1. Download the program to RAM as in Section 3.4 or copy a FLASH file to RAM as in Section 3.7.
2. Insert the EDIS jumper to disable automatic FLASH to RAM transfers.
3. Move the slide switch to the "AUTO" position
4. Press Reset to start the program.
5. You may press RESET again to restart the program.

4. THE Rix51 BOARD

4.1 Overview

Rix51 is designed for the Intel 8xC51FA and the 8xC251TB microcontrollers the 44-pin DIP package. The following diagram of the Rix51 board shows placement of the LEDs, jumpers, headers, switch, and major ICs.

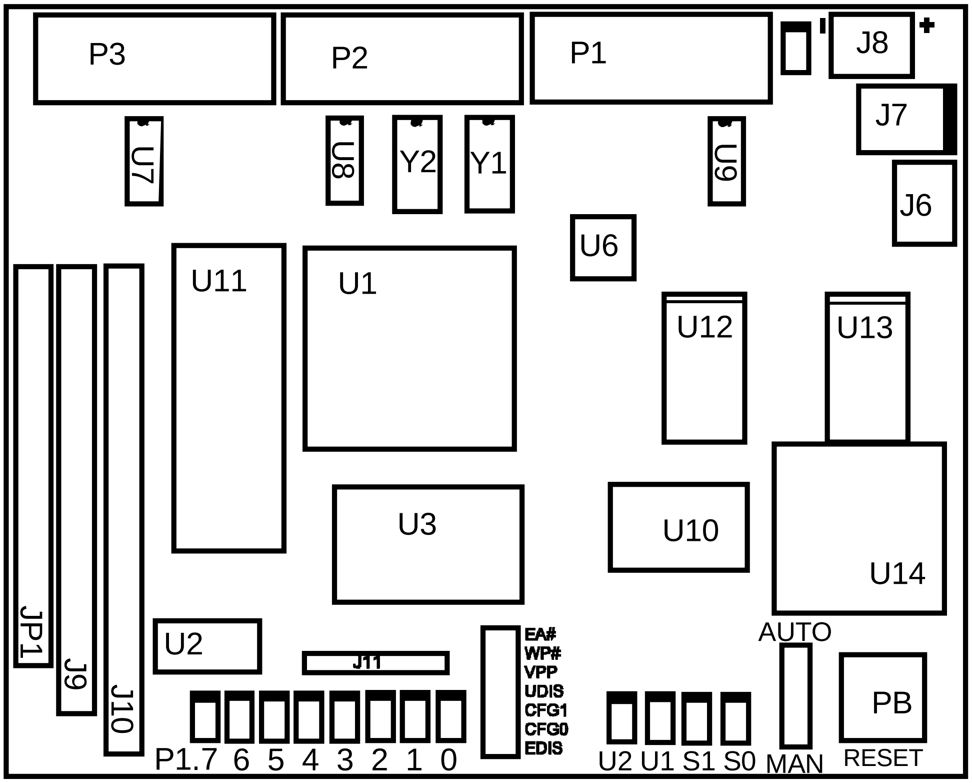


Figure 4.1 Top Board Overlay

The 51/251 evaluation board needs two connections: power supply and to the serial port of a host via a straight through serial cable.

4.2 Power

Power may be brought to the RIX-51 board in three different ways. Either a power jack (J7) or a two-position screw-type terminal block (J6) may be used to connect the board to 9VAC. This uses the on-board power regulator. Alternatively, the board may be connected to an external 5VDC regulated power supply through a two-position screw-type terminal block (J8).

4.2.1 J 7, 9VAC Power Jack

The 9VAC power supply (9VAC 500 mA transformer) provided in the kit should be used with the 2.5 mm power jack. The board has a built-in regulator which will provide the circuitry with the +5 volts needed. J7 is the default power connector.

4.2.2 J6, 9VAC Terminal Block

The terminal block J8 is in parallel with the power jack, providing an alternative connection for the 9VAC power supply. Use J8 if your power supply does not have the correct connector jack on it.

4.2.3 J8, 5 V Terminal Block

J8 may be used for power if a well-regulated +5V DC (+/- 5%) power source is available. Using J8 for power bypasses the on-board power regulator. The (+) terminal of J8 is marked on the board.

Note

When using J8 for power, a +5 volt regulated power supply must be used. Lower voltage will not operate the board. Higher voltage will irreversibly damage the active components on the board. An unregulated power supply may cause unpredictable failure conditions. Always check that the power supply is plugged into the board correctly. A diode is placed across the input in reverse. Thus if the power is applied to the Rix51 board in reverse polarity, the diode will short the power supply attempting to prevent damage to the board.

4.2.4 Connecting the Power to the Board

1. It is best to use the transformer that came with the kit. Alternatively, you may connect a 9VAC/9VDC supply, or have a well-regulated +5 volt power supply.
2. Plug the serial cable into the board.
3. Move the slide switch to the “MAN” position.
4. Plug the transformer into the wall socket.
5. Plug the transformer into the power jack J7 on the board. Alternatively, you may connect a 9VAC/9VDC supply to J6, or connect a well-regulated 5VDC supply to J8. In the latter case, observe the polarity marked on the board.
6. The power LED, located next to the power jack, should light up.

4.2.5 Troubleshooting the Power Connection

If the power LED does not operate as described in the power-up procedure, follow these steps.

1. Be sure the serial cable is connected to the board.
2. Check that the transformer is plugged into an active wall socket.
3. When using the terminal blocks instead of the power jack, make sure the power is plugged in correctly.
4. Double check that the slide switch is in the “MAN” position.
5. After checking these items, press the reset button on the board. If the board still does not respond, call the Intel Customer Support Number in your area.

4.3 Serial Ports

There are three serial ports on the board, which use the RS-232 standard for serial communications. One of the serial ports P1, uses the internal UART of the 80C51, and 80C251 microcontrollers. The second serial port on the board P2, may be used with second internal UART on the 251TB processor. The third serial port P3, uses an external UART, the 16C550.

The internal serial ports of the processor would typically be accessed by a user program according to the specific application. The external UART serial port provides a nonintrusive path to debugging the system while the user program can still communicate through the processor's serial ports.

4.3.1 P1, Internal UART

The DB9 connector labeled "P1 internal (config.)" on the bottom of the board is used for the internal UART of the microcontroller.

4.3.2 P2, Internal UART for the 251TB

P2 is the DB9 connector for the second UART on the 251TB processor. It is marked as "P2 internal" on the bottom of the board.

4.3.3 P3, External UART

The DB9 connector labeled "P3 external" on the bottom of the board is used for the external UART. The external UART is supplied with a 1.8432 MHz TTL clock to generate standard PC Baud rates.

4.3.3 Baud Rates

Rix51 Utility communicates with the board through a PC serial port. The board is in the Setup Phase during its interaction with the Rix51 Utility, running the code in the EPROM. This code listens to the internal UART (first serial port) and the external UART. Whichever port has activity is assumed to be connected to the PC running the Rix51 Utility.

The serial cable may be connected to either the internal UART of the microcontroller or to the external UART. The internal UART is run at 9600 Baud with a 24MHz oscillator. The external UART is run at 57600 Baud. Since the external UART has its own clock oscillator, its Baud rate is independent of the CPU clock frequency. If you change the CPU clock frequency (by inserting a different clock oscillator into the socket), you should use the external UART. The Rix51 utility has a selection box to pick which UART will be used. This simply sets the Baud rate to 9600 or 57600. Please note that slower computers may not be able to keep up with 57600-Baud communications. In this case, use the internal UART at 9600 Baud and the standard 24MHz oscillator.

4.3.4 Connecting the Serial Port to the Board

1. Turn off the board's power supply.
2. Connect a serial cable from P1 on the board to one of the PC serial ports. Be sure you are connecting to an available serial port on the PC.
3. Turn on the power supply for the board.

4.3.5 Troubleshooting the Serial Port Connections

If after applying power to the board you are unable to communicate with the board.

1. Make sure that the PC serial port is enabled.
2. Make sure the cables between the board, the PC, and the power supply are making good connections.
3. Make sure you are using a "straight-through" serial cable. Do not use a NULL MODEM CABLE.
4. Make sure the third party software you are using to communicate with the board is installed properly. It is recommended that you use the Rix51 Utility software to check the connections before you operate the board with a third-party software vendor's environment.

5. See the appropriate section in the manual to check that all the jumpers are in the correct positions for the processor on your board.
6. After checking these items press the reset button on the evaluation board and try to run the software again.
7. If none of the above works, call the Intel Customer Support Number in your area.

4.4 LEDs

There are 13 LEDs on the board, which provide various kinds of information about the functioning of the board.

4.4.1 Power LED

The RED LED, located next to the 9 VAC power connector lights up when power is supplied to the board.

4.4.2 Status LEDs

The four LEDs next to the slide switch and the reset button are used as status indicators. S0 is on while the board is in the Setup Phase. It is turned off when the board enters the Run Phase. The indicator S1 is on if the board is in the page mode, and off in the non-page mode. The other two indicators, labeled U1 and U2 are connected to the two general-purpose outputs of the external UART. They are left for user-specific implementations, under software control.

In addition, P1 (port 1) bits are buffered and displayed by a row of eight LEDs. The LEDs are on when the corresponding bits are 1.

4.4.3 Port 1 LEDs

P1 (port 1) bits are buffered and displayed by a row of eight LEDs. The LEDs are on when the corresponding bits are 1.

4.5 Reset Button

The Rix51 has one reset button. The push button resets the board. The Rix51 proceeds with the Setup Phase, and depending on the state of the slide switch, either proceeds to the Run Phase, or expects to communicate with the Rix51 Utility software. Refer to Appendix A for a more detailed discussion on the reset process.

4.6 Changing Microcontrollers on the Evaluation Board

You may replace the 8xC251 microcontroller by an 8xC51. Power to the board must be turned OFF when changing processors on the board. The processors are CMOS devices and should be handled with care in a static electricity free environment. The processors must be inserted with pin 1 in the correct position in the socket. The 44-pin PLCC socket has a marked corner, which should match the marked corner on the processor. Although the 8xC51-type processors do not use configuration bytes, the Rix51 board reads the configuration bytes to set the bus mode. The configuration bytes FF should be placed in FF:FFF8 and FF:FFF9 to instruct the Rix51 to start in the non-page mode. These are addresses FFF8 and FFF9 of the MCS-51 external code memory.

5. JUMPER SELECTIONS

5.1 Default Jumper Settings

The following are the default settings for the RIX-51. The EA# jumper inserted and the S6 jumper in the left position.

5.2 J2, EA#

EA# (J2) is the 2 post header located next to the evaluation processor. EA# selects between external memory and the on-board ROM of a processor, if present. The jumper, when installed, causes all instructions to be fetched from external memory. For most processors the jumper will need to be installed. When the jumper is removed, the on-board ROM is selected. This allows you to use the on-board ROM of microcontrollers like the 87C52 from Intel.

5.3 J12, VPP

Insert J12 to apply a strong 5V supply to the FLASH while programming. Remove J12 to prevent the unintended programming of the FLASH.

5.4 J5, WP#

Inserting J5 grounds the FLASH signal WP#. It prevents programming the boot sector of the FLASH. J5 should be removed in normal operation.

5.5 J2, CFG1, CFG0, and EDIS

The jumpers CFG1:CFG0 selects the FLASH file to be downloaded and run provided that EDIS is removed and the slide switch is in the “AUTO” position.

CFG1	CFG0	FLASH File
Absent	Absent	0
Absent	Present	1
Present	Absent	2
Present	Present	3

Insert EDIS to disable copying the selected FLASH file to RAM. In this case, if the slide switch is in the “AUTO” position, the program currently in RAM is executed upon reset.

5.6 S6, A17

The jumper S6 is a three-post header. A jumper must be placed in the A17 position for EPROMs of size 512K to be used in socket U11. By default, S6 should be in the left position.

5.7 J4, A15

The jumper S4 should be inserted if the EPROMs in socket U11 is of size 64K or larger. The default EPROM is a 32K device, so J4 should be removed.

5.8 S1, UART INT

S1 is a three-post header, selecting which interrupt input the external UART interrupts are to be connected. In the default configuration, S1 is left unpopulated, since the Rix51 Utility uses the external UART in a polled fashion. The Setup Phase uses P3.3, if the external UART interrupts are to be used, so it is recommended that the S1 be in the P3.2 position. If the user program uses the external UART interrupts and if the slide switch is in the “AUTO” mode, then S1 may be used in the P3.3 position.

6. HEADERS

6.1 J10, System Header

All system signals are available on the 50-pin header marked J10. The system signals on the board connect directly to the microcontroller lines. To use the header for connecting to external devices you may either sink or source current. Please check the data book for the current handling capabilities of the processor you are using. The pin assignments of J10 are given below. Refer to the circuit diagram for additional information.

Pin	Signal	Pin	Signal
1	VCC	2	VCC
3	P2.0	4	P0.0
5	P2.1	6	P0.1
7	P2.2	8	P0.2
9	P2.3	10	P0.3
11	P2.4	12	P0.4
13	P2.5	14	P0.5
15	P2,6	16	P0.6
17	P2.7	18	P0.7
19	GND	20	GND
21	GND	22	GND
23	P1.0	24	ALE
25	P1.1	26	PSEN#
27	P1.2	28	P3.6
29	P1.3	30	P3.7
31	P1.4	32	MEMDIS#
33	P1.5	34	
35	P1.6	36	RESET
37	P1.7	38	
39	P5V	40	P5V
41		42	
43	P3.4	44	P3.0
45	P3.5	46	P3.1
47	P3.6	48	P3.2
49	P3.7	50	P3.3

MDIS* can be used by external circuits to disable the on-board memory, for example, if a decoder is used to create chip selects for external devices, the chip select can be connected to MDIS* to map out the on-board memory.

6.2 J13, RIO Header

J13 connects the Rix51 to Rigel's prototyping boards RIO and RIO-JR. It carries the available I/O bits. For more information contact Rigel Corporation or visit the web site www.rigelcorp.com.

Pin	Signal	Pin	Signal
1	GND	2	VCC
3	GND	4	VCC
5		6	
7		8	
9		10	
11		12	
13		14	
15		16	
17		18	
19	P1.0	20	
21	P1.1	22	
23	P1.2	24	
25	P1.3	26	
27	P1.4	28	
29	P1.5	30	
31	P1.6	32	
33	P1.7	34	
35	P3.2	36	
37	P3.3	38	
39	P3.4	40	
41	P3.5	40	

6.3 J9, Expansion Header

J9 carries the demultiplexed address and data lines as well as the reset, read (RD#), write (WR#) ALE, XIOSEL#, and MEMDIS# lines. It is intended for interfacing with external memory-mapped input/output circuitry. J9 also includes the I/O ports P1 and P3.

Pin	Signal	Pin	Signal
1	VCC	2	VCC
3	RESET	4	ALE
5	P1.7	6	XIOSEL#
7	P1.6	8	MEMDIS#
9	P1.5	10	RD# (P3.7)
11	P1.4	12	WR# (P3.6)
13	P1.3	14	A15
15	P1.2	16	A14
17	P1.1	18	A13
19	P1.0	20	A12
21	P3.5	22	A11
23	P3.4	24	A10
25	P3.3	26	A9
27	P3.2	28	A8
29	D7	30	A7
31	D6	32	A6
33	D5	34	A5
35	D4	36	A4
37	D3	38	A3
39	D2	40	A2
41	D1	42	A1
43	D0	44	A0
45	GND	46	GND

6.4 J11

J11 is the JTAG header used to program the CPLD on the Rix51. Please contact Rigel Corporation if you would like more information on the CPLD or on implementing alternative CPLD logic.

7. CONFIGURATION OPTIONS

The Rix51 uses the same configuration bytes as the 8xC251 microcontroller to determine the bus and memory options. A brief review of the configuration bytes and their fields is given below. Refer to the data books for more information.

UCONFIG0

UCON	WSA1#	WSA0#	XALE#	RD1	RD0	PAGE#	SRC

UCONFIG1

-	-	-	INTR	WSB / EDF#	WSB1#	WSB0#	EMAP#

7.1 Memory Map

The bits RD0 and RD1 of UCONFIG0 determine the memory map.

7.2 Bus Mode

The bus mode is determined by bit PAGE# of UCONFIG0. The Non-Page mode is the bus configuration used by the MCS-51. Namely, the data and low address lines are multiplexed. In the Page mode, data and high address lines are multiplexed. Since the high address lines often do not change as consecutive instructions are executed, the Page mode yields a higher performance.

7.3 Source versus Binary Mode

Clear bit SRC if you wish to execute MCS-51 HEX code as is. When SRC is set, the microcontroller interprets the instructions to be 8xC251 instructions. In the latter case, you must recompile existing MCS-51 source code to obtain a new HEX file to run on the 8xC251.

7.4 Interrupt Mode

Set bit INTR if you want the 8xC251 to push 4 bytes onto stack upon an interrupt. When cleared, the microcontroller only pushes the two-byte address, as the MCS-51 architecture.

7.5 UCON and EMAP#

These flags are meaningful only for 8xC251 microcontrollers with on-chip ROM. Keep these set for the ROM-less members of the 8xC251 family.

7.6 Wait States

The 8xC251 allows external bus timings to be extended to accommodate slower memory devices. The wait states for region 01; is treated separately from other regions. For fastest execution times select 0 wait states. WSB1:WSB0 determine the wait states for region 01:. Similarly, WSA1:WSA0 determine the wait states for all other regions.

7.7 Extended ALE

Set XALE# for regular ALE pulses. Clearing this bit extends the ALE pulse. For fastest execution times set bit XALE#.

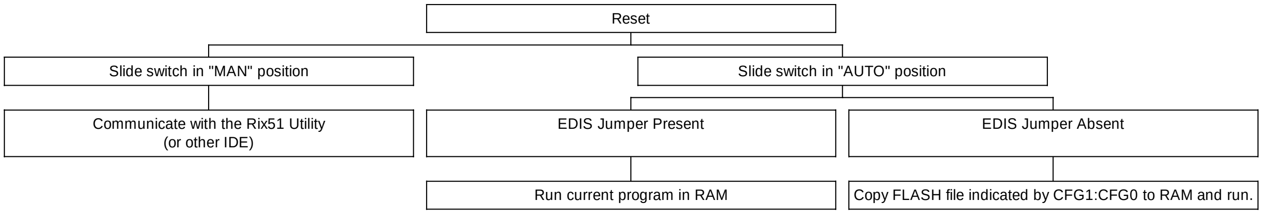
7.8 WSB / EDF#

WSB is used with the 8xC251Sx microcontrollers. Clearing WSB adds a wait state to region 01:. The 8xC251Tx microcontroller uses this bit to implement the Extended Data Float timing. Set this bit to achieve the highest execution speed.

APPENDIX A: OPERATING NOTES FOR THE Rix51

A.1 The Rix51 Board Reset Process and Options

The various reset options depend on the slide switch and jumpers as shown in Figure 1.


Reset
Slide switch in "MAN" position
Slide switch in "AUTO" position
Communicate with the Rix51 Utility (or other IDE)
EDIS Jumper Present
EDIS Jumper Absent
Run current program in RAM
Copy FLASH file indicated by CFG1:CFG0 to RAM and run.

If the slide switch is in the “MAN” position, the board resets and expects to communicate with the Rix51 Utility or a similar host PC program. The Rix51 Utility may be used to interactively download and run programs, inspect and modify the 8xC251 configuration bytes (UCONFIGx), and manage the on-board FLASH.

If the slide switch is in the “AUTO” position, the board resets and runs an existing program. The program that is executed depends on the jumper EDIS and the jumpers CFG1 and CFG0. If EDIS is present, the current program in RAM is executed. Otherwise, the FLASH file indicated by CFG1:CFG0 is executed. CFG1 and CFG0 form a two-bit word, CFG1 being the most significant bit. A jumper present is interpreted to be a logic 1, as listed below.

CFG1	CFG0	FLASH File
Absent	Absent	0
Absent	Present	1
Present	Absent	2
Present	Present	3

FLASH files may be user programs or monitors from third party vendors.

A.2 The Setup and Run Phases

In either case, the board starts in the so-called ***Setup Phase*** (SP). The status indicator S0 is on while the board is in the SP. The Rix51 Utility runs in the SP. The SP terminates by a ***Self-Induced Reset*** (SIR). After the SIR, the status indicator S0 is turned off and the program in RAM is executed. This is called the ***Run Phase*** (RP). Note that during the RP, the entire memory of the 8xC251 is mapped to RAM, excluding any memory-mapped I/O. That is, neither the EPROM nor the FLASH is accessible during the RP. (This is the default behavior of the Rix51 evaluation board. Other memory maps are possible by changing the CPLD equations for specific applications.)

The reset signal is conditioned by a supervisory circuit. It is kept active for 100-300 milliseconds after the pushbutton is released. If the slide switch is in the “AUTO” position, the SIR starts almost immediately after the hardware reset. (Transferring FLASH code into RAM takes only a few milliseconds.) So the user program is begun 200-600 milliseconds after the reset button is released.

A.3 Bus Configuration During the Run Phase

The configuration during the run phase is determined by the 8xC251 configuration bytes. Specifically, the 8xC251 Page and Non-Page modes require different external bus setups. The Rix51 board, just as the 8xC251 microcontroller, configures itself at the beginning of the Run Phase by reading the configuration bytes UCONFIGx. As long as your HEX file contains the configuration bytes, no further jumper or dialog setting is required.

A.4 Memory Map During the Run Phase

All of the external memory is mapped to the RAM devices. Depending on the UCONFIGx bytes, 64K, 128K, or 256K of RAM is used. Again, the Rix51 board reads the configuration bytes, specifically the two-bit field RD1:RD0 to determine the memory configuration. Some memory is reserved for memory-mapped input and outputs. The CPLD generates a select signal (XIOSEL#) for a range of external memory. All other memory devices are disable while XIOSEL# is active (low). This signal, along with the read, write, and demultiplexed address and data lines may be used to interface the Rix51 to external circuitry. Moreover, the MEMSEL# input is provided for external circuitry with its own decoder to disable the memory devices on the Rix51 board.

The memory-mapped I/O addresses depend on the memory model selected by the 8xC251 configuration bytes. Essentially, for the MCS-51 mode, the memory-mapped I/O is in external data memory. In all other modes, it is mapped to region FF:.

The MCS-51 compatible mode (RD1=1, RD0=1)

00:FC00 to 00:FCFF	external UART
00:FD00 to 00:FDFF	XIOSEL#
00:FE00 to 00:FEFF	reserved

Other modes

FF:FC00 to FF:FCFF	external UART
FF:FD00 to FF:FDFF	XIOSEL#
FF:FE00 to FF:FEFF	reserved

The simple program extuart.asm demonstrates how a user program may interact with memory-mapped I/O, in this case, the external UART.

A.5 Rix51 Utility Serial Port Options

Rix51 Utility communicates with the board through a PC serial port. The board is in the Setup Phase during its interaction with the Rix51 Utility, running the code in the EPROM. This code listens to the internal UART (first serial port) and the external UART. Whichever port has activity is assumed to be connected to the PC running the Rix51 Utility.

The serial cable may be connected to either the internal UART of the microcontroller or to the external UART. The DB9 connector associated with the internal UART is labeled “P1 internal (config.)” under the board. The DB9 connector associated with the external UART is labeled “P3 external” under the board. The internal UART is run at 9600 Baud with a 24MHz oscillator. The external UART is run at 57600 Baud. Since the external UART has its own clock oscillator, its Baud rate is independent of the CPU clock frequency. If you change the CPU clock frequency (by inserting a different clock oscillator into the socket), you should use the external UART. The Rix51 utility has a selection box for the board UART to be used. This simply sets the Baud rate to 9600 or 57600. Please note that slower computers may not be able to keep up with 57600 Baud communications. In this case, use the internal UART at 9600 Baud and the standard 24MHz oscillator.

A.6 Indicators

The four LEDs next to the slide switch and the reset button are used as status indicators. S0 is on while the board is in the Setup Phase. It is turned off when the board enters the Run Phase. The indicator S1 is on if the board is in the page mode, and off in the non-page mode. The other two indicators, labeled U1 and U2 are connected to the two general-purpose outputs of the external UART. They are left for user-specific implementations, under software control.

In addition, P1 (port 1) bits are buffered and displayed by a row of eight LEDs. The LEDs are on when the corresponding bits are 1.

APPENDIX B: A*p*BUILDER

A*p*BUILDER serves as a product selection tool, an on-line technical reference and a programming aide.

B.1 Product Selection Tool

A*p*BUILDER provides information to help you evaluate the architectural differences between devices from the Intel families of embedded microprocessors and microcontrollers.   You can use A*p*BUILDER's Device Diagrams, Data sheets, Hypertext Manuals, HiLites, and Peripheral Editor to help you evaluate and choose an Intel device for your microcontroller design. The fact sheets also give you information regarding development tools for Intel microcontrollers and telephone numbers for ordering tools literature.

B.2 On-line Technical Reference

A*p*BUlLDER's hypertext manual, on-line data sheets, and Questions & Answers can answer most of your design and development questions. You can use the Peripheral Editor and Register Editor to look at several configurations before deciding on the best configuration for your application

B.3 Programming Aide

After you research configuration options you can use A*p*BUILDER s editors (Peripheral, Register, and Instruction) to help generate the code to implement your configuration decisions. After selecting a configuration with the mouse, simply press the "show code" button in generate fully commented code and paste the code into your program file.

B.4 Questions & Answers

The Q&A icon on the A*p*BUILDER Tool Bar accesses a section called Questions and Answers. This section contains answers to the most commonly asked development questions. A*p*BUILDER responds to a click on the Q&A icon by jumping to the hypertext table of contents which lists a Q&A section for each peripheral of the selected device. If you select a peripheral on A*p*BUILDER's main screen before you click on the Q&A icon A*p*BUILDER jumps directly to the Q&A section for that peripheral.

B.5 HiLites

The HiLites icon on the A*p*BUILDER Tool Bar accesses an architectural overview of a selected block or peripheral. You cannot access HiLites before selecting a block or peripheral. Use this as a quick product selection tool for other Intel microcontrollers.

B.6 Peripheral Editor

The Peripheral icon on the A*p*BUILDER Tool Bar accesses the Peripheral Editor which has configuration options for the selected peripheral. Click on the peripheral you want to edit before you click on the Peripheral Editor button. Use the Peripheral Editor to study basic architecture and peripheral configurations; then generate initialization code to implement your configuration choices.

Note

The Peripheral Editor and Register Editor access and modify the same set of special function registers (SFRs); therefore you can view the results of those modifications in either editor.

B.7 Register Editor

The Register icon on the A*p*BUILDER Tool Bar accesses the Register Editor which offers bit-level register descriptions and manipulations. Click on the peripheral you want to edit before you click on the Register Editor button. Use the Register Editor to study a register and to set/clear register bits. A*p*BUILDER translates your decisions into assembly or C source code.

B.8 Instruction Editor

The Instruction icon on the A*p*BUILDER Tool Bar accesses the Instruction Editor. Use the Instruction Editor to write code, check syntax, and check instruction timings.

B.9 Editor

The Editor icon on the A*p*BUILDER Tool Bar accesses a Windows Notepad Editor file. You should use this for your source code file; then name and save it so you can access it again. You can also highlight the source code and cut and paste it directly into your editor.

APPENDIX C: PARTS LISTS

Rix51 Bill of Materials

The bill of materials given below lists all components by their reference as they appear on the board overlay.

QUAN	PART	DESIGNATOR	
	CAPACITORS		
27	10nF SMT cap	C1, C2, C21-38, C40-46	C2 Not Populated
7	10nF SMT cap	C39, C47-C52	Bottom of Board, Not Populated
2	100nF SMT cap	C15, C16	
12	1 uF cap SMT cap	C3-C14	
3	22uF SMT cap	C18, C19, C20	
1	470uF 35V	C17	
	RESISTORS		
3	1K 1/4W SMT	R4, R5, R6	
2	10K GANG SMT	R1, R2	
1	470 GANG (10)	R7	
1	470 GANG (6)	R3	
	DIODES		
3	1N4001 SMT	D1, D14, D16	
9	RED SMT LED	D6-D13	
2	Green SMT LED	D3, D4 (S0, U1)	
2	Yellow SMT LED	D2, D5 (U2, S1)	
1	BRIDGE SMT	DB1	
	CONNECTORS		
3	DB 9	P0, P1, P2	
1	PUSH BUTTON	PB1	
2	T BLOCK (2)	J6, J8	
1	DC JACK	J7	
1	SLIDE SWITCH	S2	
1	1X2 HEADER	J4	
5	1X3 HEADER	S1, S3, S4, S5, S6	S3, S4, S5 NOT POPULATE
1	1x6 HEADER	J11	
1	2X7 HEADER	(J1, J2, J3, J5, J12)	
1	2X21 HEADER	J1	
1	2X23 HEADER	J9	
1	2x25 HEADER	J10	
2	8 PIN SOCKET	Y1, Y3	
1	32 PLCC SOCKET	U3	
1	44 PLCC SOCKET	U1	
	IC'S		
3	RS232	U7, U8, U9	
1	XC9572-15TQ100C	U3	CPLD
2	UPD431000AGW-70L	U12, U13	RAM
1	E28F200B5T-80	U10	FLASH Memory
1	TL16C550	U6	External UART

2	DS1233D-10 SMT	U4, U5	Reset Chips
1	74HC240D	U2	Buffer
1	PT5101A-ND	U14	Power Regulator
1	8051//251	U1	
1	27C256	U3	EPROM
1	24MHz OSC	Y2	Processor Clock
1	1.8432 OSC	Y1	External UART Clock

APPENDIX D: TOP OVERLAY AND SCHEMATICS