R-515JC
USER'S GUIDE

Version 2.01
July 2000

RIGEL CORPORATION
PO BOX 90040
GAINESVILLE, FLORIDA 32607
(352) 373-4629
FAX (352) 373-1786
www.rigelcorp.com
tech@rigelcorp.com

Warranty

RIGEL CORPORATION - CUSTOMER AGREEMENT

Return Policy. This policy applies only when product purchased directly from Rigel Corporation. If you are not satisfied with the items purchased, prior to usage, you may return them to Rigel Corporation within thirty (30) days of your receipt of same and receive a full refund from Rigel Corporation. You will be responsible for shipping costs. Please call (352) 373-4629 prior to shipping. A refund will not be given if the READS package has been opened.

READS and RROS License. The READS and RROS being purchased is hereby licensed to you on a non-exclusive basis for use in only one computer system and shall remain the property of Rigel Corporation for purposes of utilization and resale. You acknowledge you may not duplicate the READS or RROS for use in additional computers, nor may you modify, disassemble, translate, sub-license, rent or transfer electronically READS or RROS from one computer to another, or make it available through a timesharing service or network of computers. Rigel Corporation maintains all proprietary rights in and to READS and RROS for purposes of sale and resale or license and re-license. BY BREAKING THE SEAL AND OTHERWISE OPENING THE READS PACKAGE, YOU INDICATE YOUR ACCEPTANCE OF THIS LICENSE AGREEMENT, AS WELL AS ALL OTHER PROVISIONS CONTAINED HEREIN.

Limited Warranty. Rigel Corporation warrants, for a period of sixty (60) days from your receipt, that READS disk(s), RROS, hardware assembled boards and hardware unassembled components shall be free of substantial errors or defects in material and workmanship which will materially interfere with the proper operation of the items purchased. If you believe such an error or defect exists, please call Rigel Corporation at (352) 373-4629 to see whether such error or defect may be corrected, prior to returning items to Rigel Corporation. Rigel Corporation will repair or replace, at its sole discretion, any defective items, at no cost to you, and the foregoing shall constitute your sole and exclusive remedy in the event of any defects in material or workmanship.

THE LIMITED WARRANTIES SET FORTH HEREIN ARE IN LIEU OF ALL OTHER WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

YOU ASSUME ALL RISKS AND LIABILITY FROM OPERATION OF ITEMS PURCHASED AND RIGEL CORPORATION SHALL IN NO EVENT BE LIABLE FOR DAMAGES CAUSED BY USE OR PERFORMANCE, FOR LOSS PROFITS, PERSONAL INJURY OR FOR ANY OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES. RIGEL CORPORATION'S LIABILITY SHALL NOT EXCEED THE COST OF REPAIR OR REPLACEMENT OF DEFECTIVE ITEMS.

IF THE FOREGOING LIMITATIONS ON LIABILITY ARE UNACCEPTABLE TO YOU, YOU SHOULD RETURN ALL ITEMS PURCHASED TO RIGEL CORPORATION.

Board Kit. If you are purchasing a board kit, you are assumed to have the skill and knowledge necessary to properly assemble same. Please inspect all components and review accompanying instructions. If instructions are unclear, please return the kit unassembled for a full refund or, if you prefer, Rigel Corporation will assemble the kit for a fee of $30.00. You shall be responsible for shipping costs. The foregoing shall apply only where the kit is unassembled. In the event the kit is partially assembled, a refund will not be available, however, Rigel Corporation can, upon request, complete assembly for a fee based on an hourly rate of $50.00. Although Rigel Corporation will replace any defective parts, it shall not be responsible for malfunctions due to errors in assembly. If you encounter problems with assembly, please call Rigel Corporation at (352) 373-4629 for advice and instruction. In the event a problem cannot be resolved by telephone, Rigel Corporation will perform repair work, upon request, at the foregoing rate of $50.00 per hour.

Governing Law. This agreement and all rights of the respective parties shall be governed by the laws of the State of Florida.

Table of Contents

1 INTRODUCTION

1.1 Hardware Overview

The R-515JC is a six-layer board with separate VCC and Ground planes intended for demanding industrial applications. Ideal for data logging or mixed mode communications this board is the newest in Rigel's expanding line of 8051 controllers. It's low cost and flexibility also make it ideal as a training platform. The R-515JC has four serial ports, one a dedicated RS-232 port, one a synchronous serial port, and two which can be configured for RS-232 or RS-422 / RS-485. The R-515JC also has CAN, Controller Area Network with the physical layer on-board. The board comes standard with 32K RAM and 32K EPROM, but will accept up-to 512K RAM and 128K EPROM. There is a Y2K Real Time Clock / Calendar (RTCC) which may be populated as well as a battery back-up system for both the RAM and RTCC. The processor supports 10-bit A/D, as well as having 48 I/O lines, 3 16-bit timers and a watchdog timer. For more details on the processor and board, see the listings below.

1.1.1 C515C Processor In The P-MQFP-80 Package Includes:

- 64k byte on-chip ROM (external program execution is possible)
- 256 byte on-chip RAM
- 2K byte of on-chip XRAM
- Up to 64K byte external data memory
- Superset of the 8051 architecture with 8 datapointers
- Up to 10 MHz external operating frequency (1 µs instruction cycle time at 6 MHz external clock)
- Eight ports : 48+1 digital I/O lines, 8 analog inputs
  Quasi-bi-directional port structure (8051 compatible)
  Port 5 selectable for bi-directional port structure (CMOS voltage levels)
- On-chip Full-CAN (Controller Area Network) controller system
  256 register/data bytes are located in external data memory area
  Upto 1 Mbit/sec CAN communications at 8-10 MHz operating frequency
- Three 16-bit timer/counters
  Timer 2 can be used for compare/capture functions
- 10-bit A/D converter with multiplexed inputs and built-in self calibration
- Full duplex serial interface with programmable baudrate generator (USART)
- SSC synchronous serial interface (SPI compatible)
  Master and slave capable
  Programmable clock polarity / clock-edge to data phase relation
  LSB/MSB first selectable
  1.25 Mbit/sec transfer rate at 10 MHz operating frequency
- Seventeen interrupt vectors, at four priority levels selectable
- Extended watchdog facilities
  15-bit programmable watchdog timer
  Oscillator watchdog
- Power saving modes
  Slow-down mode
  Idle mode (can be combined with slow-down mode)
  Software power-down mode with wake-up capability through INT0 pin
  Hardware power-down mode
- CPU running condition output pin
- ALE can be switched off
- Multiple separate VCC/VSS pin pairs

1.1.2 The R-515JC Board:

- 32K of SRAM / optional 128K - 512K SRAM
- 32K of monitor EPROM / optional 128K EPROM
- Optional Y2K Real-Time Clock / Calendar (RTCC) (uses DS1685)
- Optional battery back-up for RAM and Real-Time Clock / Calendar
- 3 full-duplex serial ports on board
  CPU UART port with a RS-232 driver
  Optional Dual UART 88C92 for two additional serial ports

Each of these ports may be used with either an RS-232 driver or an RS-422/RS-485 driver
2 Serial ports terminate at DB-9 connectors
Optional header style connectors on board for all serial ports

- CAN (Controller Area Network) physical layer (80C250 or 80C251) built into the board
- CAN port terminates on screw-type terminal blocks
- 12 general purpose digital input/output bits on screw type terminal blocks
- Two-way reset feature gives access to all interrupt vectors
- Demultiplexed processor Address and Data lines
- All system signals are available on a 96-pin header
- Power supplied to the board by way of a 2 position terminal block
- Power on LED
- Board operates on +5 volts
- Operating temperature 0 to 70C
- Machine screw sockets under all through-hole IC's
- 4 layer 4" x 5.5" board
- Mounting holes in corners

1.2 Software Overview

READS51, version 4.00, is Rigel Corporation's Integrated Development Environment for the 8051 family of processors. READS51 constitutes a complete system for developing embedded control applications when used with Rigel Corporation's 8051 boards. Efficient software development and rapid hardware prototyping are combined in a single integrated development environment. READS51 includes an editor, a host-to-board communications system, and an assembler, linker locator, and C complier. READS51 is completely rewritten in native 32-bit code to run on Windows95, 98, and NT. READS51 includes a sophisticated project management system to simplify code reusability and version control. READS51 supports a full debugger for assembly. The debugger allows you to step through your code with breakpoints and variable watches as the compiled code runs on the target board, similar to the operation of an in-circuit emulator.

The READS51 software has the following distinctive features:
The 8051 boards are designed to communicate with a PC (IBM PC or compatible) acting as a host. The host-to-board communications are carried out through a serial port (COM1 - COM4). The host-based development system READS is a menu-driven environment with an editor, assembler, debugger, and PC-to-board communications software.

The monitor program (RROS) includes a monitor system and user-accessible system calls for control and communication support. The RROS monitor may be used to communicate with an ASCII terminal when the PC host is unavailable. The source code of the user-accessible systems calls is provided.

The development boards use the 8051 family of microcontrollers. The instruction set of these microcontrollers is the MCS-51 instruction set. The boards use external RAM during the development cycle. Once an application program is developed, it may be permanently placed in EPROM. With an application-specific program installed, the board may be used as an embedded controller.

1.2.1 Example Software

Tutorial source code is provided to experiment with the capabilities of the R-515JC board and READS. Examples are designed to illustrate the features of the 8051 family of microcontrollers, specifically digital and serial input/output, timers and counters, and interrupt logic. The example software may be found on the READS CD-ROM. Please refer to the comments embedded in the programs for further information.

Users are encouraged to modify the circuit diagrams and example software in developing their own specific applications. The source code of the user-accessible systems calls, as well as all examples on the distribution disk may be used or incorporated into applications by the registered buyer without any royalties, fees, or limitations. Rigel Corporation is not responsible for the suitability or correctness of the example software. Refer to warranty for additional information.

1.3 Package List

Your R-515JC / READS package includes the following:

- R-515JC populated with 32K RAM, 32K EPROM
- R-515JC User's Guide, paper copy
- READS software on CD with on-line help and User's Guide
- Source code for user-accessible system calls on CD
- Example software on CD

A serial modem cable with a male DB9 connector and a regulated 5 volt 500mA power source are to be supplied by the user.

2 SOFTWARE SETUP

2.1 System Requirements

Reads51 Version 4.x is designed to work with an IBM PC or compatible, 486 or better, running Windows 95, 98, or Windows NT. The newest version of the software is always available to download off our web site, www.rigelcorp.com. We encourage you to check our web site often to keep up-to-date.

2.2 Software Installation, Reads51

If you receive a CD from Rigel, follow these steps:

1. Place the CD-ROM in your drive.
2. Go to the Rigel Products **|** 8051 Software **|** Reads51 **|** Win95-nt | and click on the SetupReads400.exe file. The program will then install in your system.
3. Follow the standard install directions answering the questions with the appropriate answers

If you download the software from the web, (www.rigelcorp.com)

1. Click on the SetupReads400.exe file. The program will then install in your system.
2. Follow the standard install directions answering the questions with the appropriate answers

2.3 Quick Start

2.3.1 R-515JC Start Up

1. Check to make sure jumpers are the EA# and PE# positions on the header.
2. Check to make sure the jumper above U6 is in the <256K position.
3. Check to make sure the slide switch is in the MON position.
4. Connect the R-515JC to the PC host via a modem cable in the P1 DB9 connector.
5. Connect the R-515JC to a well-regulated 5-Volt supply. The red LED should light up when power is connected.
6. Run the READS51 host driver by selecting **Start | Programs | READS51**. You may also start READS51 by double clicking on the READS51 short cut icon if installed.
7. Specify the serial port (COMM Port) that is connected to the board by opening the **Options | TTY Options** window.
8. Select the Toolchain and Target platform by selecting **Options | Toolchain/Target** and selecting Reads51 Toolchain v4 and the target RROS.
9. Open the TTY window using the menu command **View | TTY Window**.
10. Press RESET on the embedded controller board and observe the prompt in the TTY window.

2.3.2 Verifying that the Monitor is Loaded

Make sure the TTY window is active, clicking the mouse inside the TTY window to activate it if necessary. Then type the letter ‘***H***’ (case insensitive) to verify that the monitor program is responding. The ‘H’ command displays the available single-letter commands the monitor will recognize.

The READS monitors use single-letter commands to execute basic functions. Port configurations and data, as well as memory inspection and modifications may be accomplished by the monitor. Most of the single-letter commands are followed by 4 hexadecimal digit addresses or 2 hexadecimal digit data bytes.

The list of monitor commands is displayed with the **H** command while the monitor program is in effect. The **H** command displays the following table.

```
B xxxx          sets Break point at address xxxx
C xxxx-xxxx     displays Code memory
D xx-xx         displays internal Data ram
D xx=nn         modifies internal Data ram
D xx-xx=nn      fills a block of internal Data ram
G xxxx          Go - starts executing at address xxxx
H               Help - displays monitor commands
K               Kills (removes) break point
L               down Loads Intel hex file into memory
P x             displays data on Port x
```

P x=nn	modifies data on Port x to nn
R	displays the contents of the Registers
S	displays Special function register addresses
S xx-xx	displays Special function registers
S xx=nn	modifies Special function registers
S xx-xx=nn	fills Special function registers
X xxxx-xxxx	displays eXternal memory
X xxxx=nn	modifies eXternal memory
X xxxx-xxxx=nn	fills eXternal memory

A single-letter command may be followed by up to 3 parameters. The parameters must be entered as hexadecimal numbers. Each ’x’ above represents a hexadecimal digit (characters 0..9, A..F). Intermediate spaces are ignored. Alphabetic characters are converted to upper case. The length of the command string must be 16 characters or less. The command syntax is:

Letter [address][-address][=data]<CR>.

2.3.2 Downloading and Running an Assembly Program

1. Use the use the **Project | Open Project** command to open the project AbsoluteAssembly01.
2. Assemble the program using the **Compile | Build** command.
3. Use the **Compile | Toggle Mode** command to switch to the Run/Debug Mode.
4. Click on the **Compile | Run** command and specify the starting address 8000 (hex).
5. Again the **Compile | Toggle** Mode command to revert back to the Build Mode.

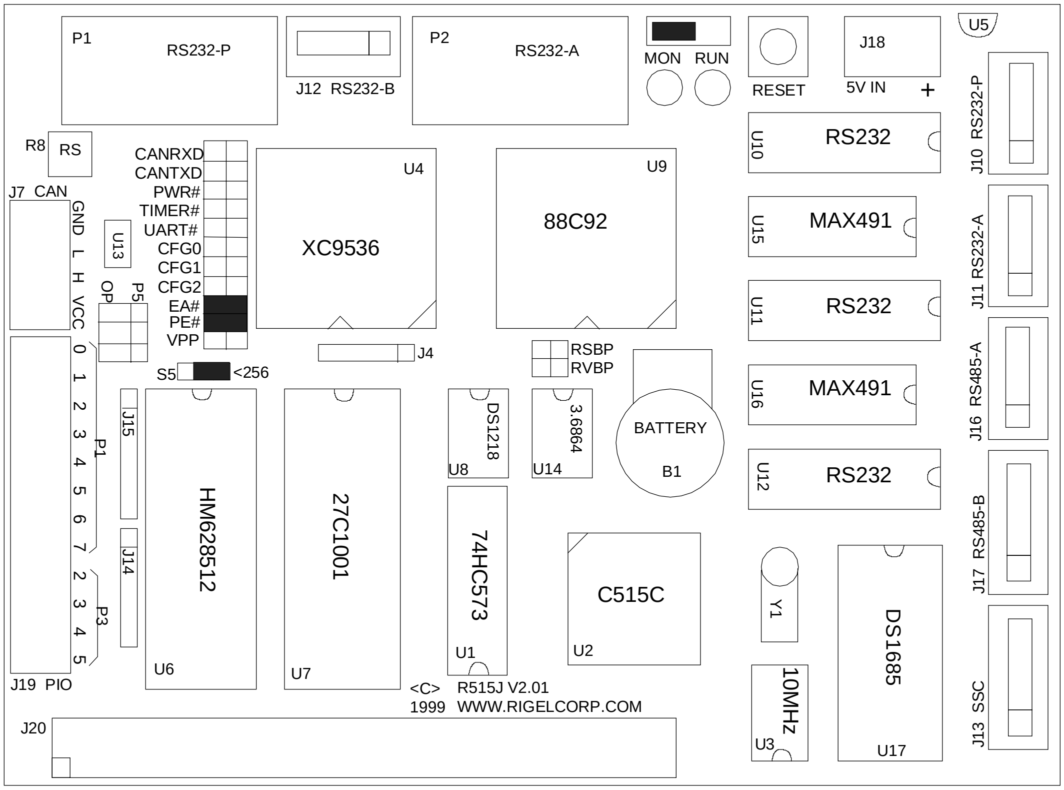
2.3.3 Downloading and Running a C Program

1. Use the **Project | Open Project** command to open the project Hello.
2. Compile the program and download it to the board using the **Compile | Build and Download** command. The project will be compiled and the resultant HEX code will be downloaded to the target board.
3. Press and hold the RESET button on the board. While the RESET button is pressed, flip the MON / RUN switch to the RUN position. This swaps the memory map on the board so that RAM occupies low memory. The HEX code downloaded to RAM executes when you release the RESET button.

3 OPERATING NOTES

3.1 Overview

R-515JC uses the Siemens C515C processor, an 8051 compatible microcontroller in the 80 pin MQFP package. These microcontrollers use the MCS-51 instruction set. The R-515JC uses external RAM during the development cycle. Once an application program is developed, it may be permanently placed in EPROM, or battery backed RAM. With an application-specific program installed, the R-515JC may be used as an embedded controller.



The R-515JC has 12 terminal blocks connected to Port 1 and 4 bits of Port 3. Each port may be used as either an input or an output port. The C515C ports may be operated in the 8051 mode. That is, the ports contain open-drain transistors with pull-up resistors. In this mode, when used as output ports, it is recommended that the ports sink current. Similarly, when used as input ports, first write 1's to the ports and have the external signal drive the port low. Some of the C515C ports may be configured as true push-pull ports. Refer to the C515 data book for further information. The top overlay above is not drawn to scale. It is a drawing to show position of the various components and of the default jumper and switch settings.

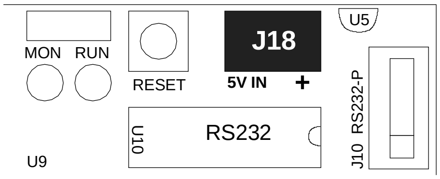


Figure 3.2 Power Connector

3.2 Power, J18

Power is brought to the R-515JC board by a two-position screw-type terminal block, J18. A well-regulated (+/- 5%) 5V DC source is required. The (+) terminal is marked on the board. Note that a diode is placed across the input in reverse. If the power is applied to the R-515JC board in reverse polarity, the diode will short the power supply attempting to prevent damage to the board. Note that, the board may optionally be powered by the CAN connector (J7) or the 96-pin I/O header (J20).

3.3 Communication

There are 4 serial ports and one CAN port on the R-515JC. The serial ports consist of 3 asynchronous serial ports and 1 synchronous serial port. Two of the asynchronous serial ports can be accessed by the DB9 connectors located at the top of the board or headers mounted on the right side of the board. The third serial port and the synchronous serial port are available only on the corresponding headers mounted on the right side of the board. The CAN port is available on terminal blocks located on the left side of the board.

3.3.1 Asynchronous Serial Ports

There are three asynchronous serial ports available on the R-515JC board. The DB-9 connectors P1 and P2 are only available as RS-232 serial ports and may be used to communicate with the PC host during programming and debugging the board. The serial port P1 is always populated and used as an RS-232 serial port. The other two serial ports are optional and may be used as RS-232 or RS-422/RS-485. P2 is connected to channel A of the dual UART. Note that if only one serial port is needed, the dual UART may be left unpopulated.

3.3.1.1 Serial Port P1, J10

The CPU serial port on the R-515JC is accessed through an RS-232 level converter in U10. The microcontroller supports the CPU transmit and receive signals. A minimal serial port is constructed with just 3 lines: transmit, receive, and ground, disregarding all hardware handshake signals. P1 of the R-515JC is a DB-9 female connector used to connect the board to an IBM compatible PC. A straight-through modem cable may be used. That is a cable connecting pin 2 of the R-515JC to pin 2 of the host, and similarly pin 3 to pin 3, and pin 5 to pin 5. P1 is also available on the header J10 labeled RS-232-P. Furthermore, P1 RS-232-level receive (SR0) and transmit (ST0) signals are available on the 96-pin I/O Header J20. The corresponding TTL-level signals from the microprocessor are also available on this header.

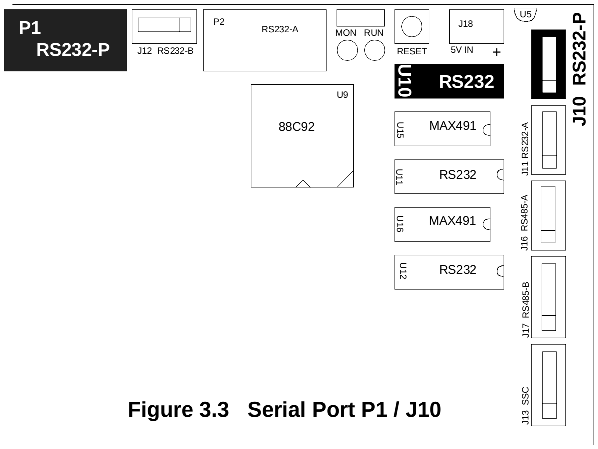


Figure 3.3 Serial Port P1 / J10

3.3.1.2 Using the Optional Serial Ports

There are two additional serial ports available on the R-515JC. These ports are optional. These are made available with the use of a dual UART the 88C92 which will only be populated if additional serial ports were requested at the time the board was purchased. The additional serial ports may be used as either RS-232 or RS-422 / RS-485 ports. When using the optional serial ports please note that although there are sockets on the boards for 2 RS-232 IC's (U11, and U12) and 2 sockets for the RS-485 IC's (U15 and U16) care should be taken to populate only one driver for each port. For channel A of the dual UART, either an RS-232 driver (U11) or an RS-422/ RS-48 driver (U15) may be used. Do not populate both U11 and U15. Note that the receive outputs of the to drivers terminate at the same UART pin. Since both drivers will not be receiving the same data, populating both drivers would result in a conflict at this pin. Similarly for dual UART channel B, do not populate both U12 and U16. Refer to circuit diagrams for further connectivity information.

3.3.1.2.1 Serial Port, P2, RS-232-A

The DB-9 connector P2 may be used as a second RS-232 serial port for debugging when U11 is populated with an RS-232 IC. The dual UART supports the CTS and RTS hardware handshake signals. The handshake signals are not active in the default configuration. The automatic generation and use of these handshake signals are selected by programming the 88C92 control registers, usually in the initialization code. Refer to the 88C92 data book for further information on the available operating modes.

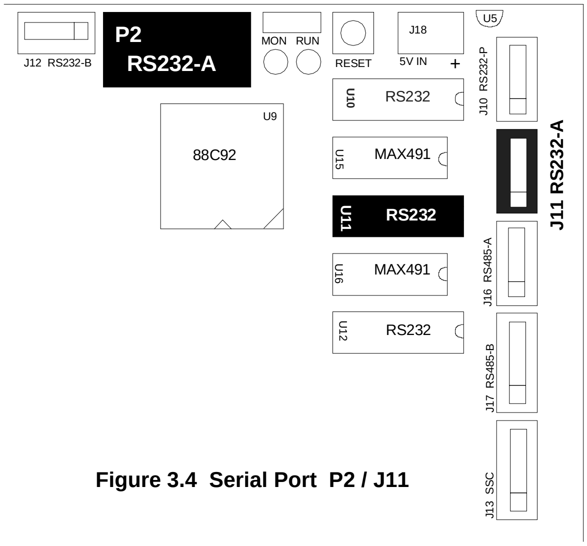


Figure 3.4 Serial Port P2 / J11

P2 includes the 5 lines: transmit, receive, CTS (Clear To Send), RTS (Request To Send) and ground. A straight-through modem cable may be used when connecting P2 to an IBM-PC serial port. That is a cable connecting pin 2 of the R-515JC to pin 2 of the host, and similarly pin 3 to pin 3, pin 5 to pin 5, pin 7 to pin 7, and pin 8 to pin 8. P2 is also available on the header J11 labeled RS-232-A. The 'A' indicates the association with channel A of the dual UART.

Pin	P2 of the R515JC	PC Serial Port
2	TXD	RXD
3	RXD	TXD
7	CTS	RTS
8	RTS	CTS
5	Ground	Ground

Note that CTS is an input to the transmitter. It is generated by the receiver at the other end of the serial link. When active, it signifies that the transmitter is clear to send data. RTS is an output from the receiver. It is sent to the transmitter at the other end of the serial link. When active, it signifies that the transmitter is requested to send data. RTS and CTS may be viewed as the two ends of the same handshake signal wire.

3.3.1.2.2 Serial Port, J12, RS-232-B

The TXD, RXD, CTS, RTS, and Ground signals from the dual UART Channel B RS-232 driver terminate at J12. The header J12 is located between the two DB-9s. The function and structure of J12 is similar to J11, except that J12 is associated with Channel B, whereas J11 is associated with Channel A of the dual UART.

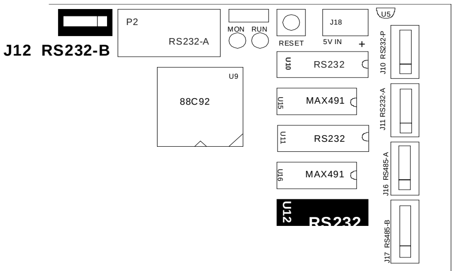


Figure 3.5 Serial Port RS-232-B

3.3.1.2.3 Serial Port, J16, RS485-A

J16 is used for the dual UART Channel A, when Channel A is used as an RS-422 / RS-485 port. J16 is connected to the IC in socket U15. When populated this socket contains a MAX491, an RS-422 / RS-485 driver chip. Note that only one chip should be populated in the sockets used for Channel A. Do not populate both sockets U11 and U15 at the same time. Please see the Maxim Data book or web site for details on this chip.

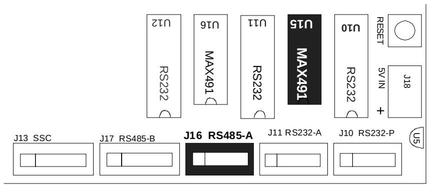


Figure 3.6 Serial Port RS485-A

3.3.1.2.4 Serial Port, J17, RS485-B

J17 is used for the dual UART Channel B when Channel B is used as an RS-422 / RS-485 port. J17 is connected to the IC in socket U16. When populated this socket contains a MAX491, an RS-422 / RS-485 driver chip. Note that only one chip should be populated in the sockets used for Channel B. Do not populate both sockets U12 and U16 at the same time. Please see the Maxim Data book or web site for details on this chip.

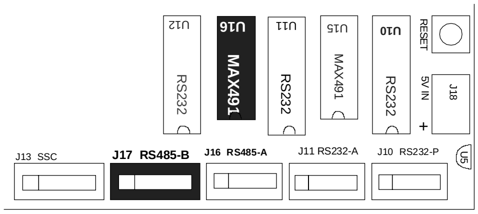


Figure 3.7 Serial Port RS485-B

3.3.2 Synchronous Serial Port

J13 is the header used for the configurable synchronous serial port on the board. Please see the Siemens data book for details on using the SSC.

3.3.3 CAN Terminal Block, J7

The CAN LO and CAN HI lines are brought to the terminal block marked J7. A two-wire twisted pair may be used to connect the R-515JC board to other CAN devices. The terminal block J7 also carries the VCC and GND lines for reference or to provide power. Please note that the CAN signals CANRS, CANLO, and CANHI are also available on the 96-pin I/O Header J20.

3.3.3.1 CAN Physical Layer Considerations

The R-515JC uses the 82C250 chip to provide a physical layer interface to a twisted pair. The outputs of the 82C250 are brought directly to J7 as CAN-LO and CAN-HI without any pull-up or termination resistors. If your physical layer contains no resistors, for example, if you connect two R-515JC boards together, pull-up (and pull-down) resistors will improve the line characteristics. The optimum resistor values depend on the transmission speed and the properties of the twisted pair. Note that the resistance and the capacitance of the twisted pair increase with the length of the connection. Also, the capacitance of the line becomes more important as the transmission speeds increase. Typically, two external resistors (1K to 10K) are sufficient, one pulling CAN-LO to VCC and the other pulling CAN-HI to GND. That is, use a pull-up resistor on CAN-LO and a pull-down resistor on CAN-HI. These resistors are in locations R9 and R10 and are not populated on the board.

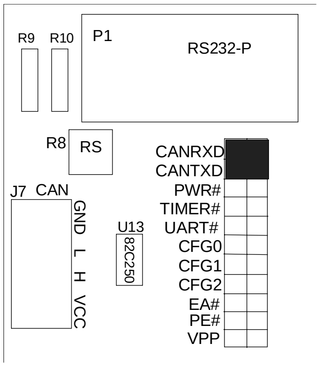


Figure 3.8 CAN Interface

3.4 Potentiometer R8

The potentiometer R8 is used by the slope control feature of the CAN interface chip, the 82C250. Turning R8 clockwise increases its value. Turned all the way counterclockwise, R8 connects the slope control pin of the 82C250 to ground. Slope control refers to desensitizing the CAN interface chip to fast acting transients. External noise sources may induce interference on the twisted pair CAN physical layer, which appear as transients to the CAN interface chip. When R8 is about zero, (grounding the slope control pin of the 82C250), the CAN interface chip processes the signals without measuring their slopes. This gives the fastest possible operation of the CAN interface chip. In this case, the physical layer should be coaxial or well insulated from external noise sources. As the value of R8 increases, the CAN interface chip enters its slope control mode. The higher the value of R8 (more counterclockwise rotation), the less sensitive the chip is to fast acting signals. This is more desirable in a noisy industrial environment when unprotected twisted pair physical media are used.

3.5 Switches

The R-515JC has one reset button and one slide switch. The pushbutton resets the board. The slide switch changes the memory configuration. The precise effect of the slide switch depends on the memory configuration (referred to as the memory mode, or simply as the mode) selected by the jumpers. Please refer to the Section 5 for detailed information about the memory modes.

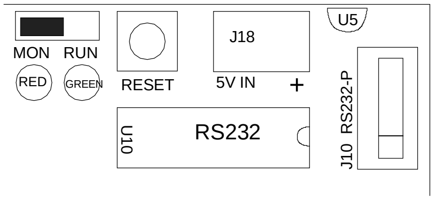


Figure 3.9 Switches and LEDs

3.6 LEDs

There are two LEDs on the R-515JC. The LEDs indicate the current operating mode of the board. They also provide visual confirmation that power is applied.

The LEDs light up depending on the operating mode which in turn is determined by the position of the slide switch. The MON LED (red) is lit when the R-515JC is run with the monitor program in the lower 32K of memory. The RUN LED (green) is lit when the board is operated with the RAM in the lower 32K of memory.

4 JUMPER SELECTION

4.1 EA#, (J1)

EA# jumper (External Enable) J1 is located in the row of headers just left of IC U4, the XC9536. Inserting a jumper in J1 grounds the EA# signal, removing the jumper connects the EA# signal to VCC through a pull-up resistor. The EA# signal 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. Since the microprocessors do not have valid code in factory-masked internal ROM, the EA# jumper is always installed in the standard configuration.

4.2 PE# (J2)

The PE# jumper (J2) is located in the row of headers just left of IC U4, the XC9536, right below the EA# jumper. The PE# jumper is inserted to ground the PE# signal. A low level (grounding) on this pin allows the software to invoke the CPU power down, idle and slow down modes. Removing the jumper at PE# holds the signal high which blocks the use of the software controlled power saving modes. If PE# is at a logic low level during reset, the watchdog timer is turned off. A high logic level during reset starts the watchdog timer. When left unconnected this pin is pulled high by an internal pull-up resistor.

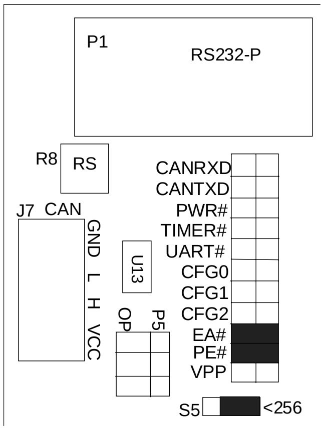


Figure 4.1 Default Jumper Settings

4.3 <256K (S5)

Jumper S5 is a three-pin header. A jumper must be inserted, connecting the center post to one of the outside posts. The size of the RAM placed in the socket for U6 determines the jumper position in the header. For 32K or 128K RAM devices, place the jumper in the position marked “<256K.” For 512K devices, place the jumper opposite to the “<256K” mark.

4.4 PWR#/ TIMER#/UART# (J3)

J3 is actually a collection of three jumpers. One side of the jumpers are connected to P7.0 of the C515C. This port functions as an interrupt input (INT7#) of the C515. The real-time clock / calendar has two interrupt output, namely, PWR# and TIMER# (IRQ#). Similarly, the dual UART has an interrupt output UART# (INT#). All three interrupt sources are generated by open-drain-type outputs. Simply connecting the interrupt sources is equivalent to a logical OR operation. From top to bottom, the J3 jumpers connect the Timer interrupts PWR# and IRQ#, and the UART interrupt INT# to P7.0. Refer to the circuit diagrams for further information.

4.5 CFG0/CFG1/CFG2 (J5)

J5 is actually a collection of three jumpers. These jumpers are connected to the CPLD to select among the available memory modes. Refer to Section 5 for further information on the memory maps and memory modes.

4.6 CANRxD, J8 and CANTxD (J9)

The two jumpers J8 and J9 connect the CAN receive and transmit signals to the microcontroller pins P4.7 and P4.6, respectively. Remove these jumpers if P4.6 and P4.7 are to be used as general-purpose input/output port bits. Note that removing J8 and J9 prevents the microcontroller from accessing the CAN physical layer driver U13.

4.7 RVBP (J21) and RSBP (J22)

RVBP and RSBP are bypass jumpers. Populate J21 and J22 if the battery backup is not used. In this case, not only the battery but the battery backup controller chip U8 (DS1218) is removed. In battery-backed operation, the controller chip, U8, monitors the supply voltage and the memory select signal. Memory select is disallowed if the supply voltage (VCC) falls below a limit. In this case, additionally, the battery supplies power to the memory device. When VCC rises above the limit, the memory chip select signals are allowed, and the memory device is powered by VCC.

Jumper J21 is the RAM Voltage By-Pass jumper. It bypasses the controller chip to directly supply the RAM with VCC. Similarly, Jumper J22 is the RAM Select By-Pass jumper. It bypasses the controller chip to directly provide the RAM chip select signal.

4.8 Bank Address Source Selections (S2, S3, S4)

There are two ways to generate the bank addresses A16, A17, and A18. Either Port 5 bits P5.0, P5.1, and P5.2 or the dual UART general-purpose output bits OP4, OP5, and OP6 may be used. The source of each bank address is determined by the jumpers S2, S3, and S4 respectively. Note that the port bits are at logic level high after a reset. The CPLD inverts the source signal so that the bank addresses are all at logic level low (0) after a reset. This assures that code is fetched from address 0 immediately following a reset.

4.9 Default Jumper and Switch Settings

The following are the default settings for the R-515JC as sent from the factory.

- A jumper in the EA# header.
- A jumper in the PE# header.
- A jumper in the ’<256’ position of S5.
- The slide switch in the ’MON’ position.

The top overlay on page 7 shows the default jumper settings and switch position.

5 MEMORY CONFIGURATION

The R-515JC may take up-to 512K RAM and 128K EPROM. The board is routinely populated with 64K of memory, of which 32K is EPROM and 32K is static RAM. Program memory and external data memory are decoded to overlap. In this configuration, programs may be downloaded from the PC host, placed in RAM as data, and subsequently executed as program instructions.

5.1 Memory Addressing

The 8031 family of microcontrollers address 64K of program. The microcontroller may read from both external data memory and external code memory using the `movx` and `movc` instructions. The microcontroller may write to external data memory but may not write to external code memory.

The microcontroller pin Program Segment Enable (PSEN#) is activated (made logic 0) when a byte is to be read from external program memory, and pin Read (RD#), when a byte is to be read from external data memory. By combining these signals by an AND gate (PSEN# AND RD#), the same physical 64K memory block is made to appear as both external code and data memory. The default R-515JC configuration overlaps external data and code memory blocks by combining PSEN# and RD# in this manner. This allows downloading and running programs on the R-515JC. The pound sign (#), when used as a suffix signifies that the signal is active when low. For example, external memory is enabled when the EA# line is held at logic low (close to ground voltage).

5.2 Memory Options

5.2.1 EPROM (U7)

R-515JC has a 32-pin socket for U7. This socket holds 32K, 64K, or128K of memory sing 27C256, 27C512, or 27C1001 EPROM devices.

5.2.2 RAM (U6)

R-515JC has a 32-pin socket for U6. This socket accepts 32K, 128K, or 512K of memory using 62C256, 68C1000, or 68C4000 RAM devices.

5.3 Memory Modes

The R-515JC accommodates up to 128K of EPROM and 512K of RAM. Memory is decoded by the CPLD device, the XC9536. Memory modes refer to the way memory is decoded. More specifically, the way the physical addresses are derived from logical addresses. The physical addresses are those seen by the memory devices, and the logical addresses are those generated by the microcontroller. The default CPLD program supports three major memory modes, selected by the configuration jumpers CFG0, CFG1, and CFG2, and by the MON/RUN slide switch. The CPLD program is given in Section 7. Note that the CPLD may be reprogrammed to support a very wide range of custom memory modes, for example to relocate memory-mapped input/outputs.

Memory Mode	**CFG2**	**CFG1**	**CFG0**
RROS	No	No	No
NVRAM	No	See Text	Yes
Banking	Yes	X	X

The ’X’s stand for "don’t care," meaning the corresponding mode is active irrespective of a jumper being present at this location.

5.3.1 Memory-Mapped Input/Output

There are three memory-mapped input/output ranges corresponding to the user I/O select (XIOSEL#), the external dual UART, and the Real-Time Clock Calendar (RTCC). The default CPLD equations (As of May 2000, boards shipped before MAY 2000 use the old base address) fix the address of memory-mapped input/outputs to the following base addresses:

Peripheral	Old Base Address	New Base Address
Real-Time Clock Calendar	0xFC00	0xF400
UART	0xFD00	0xF500
XIO	0xFE00	0xF600

Note that the memory-mapped input/output ranges are effective in all memory modes. That is, no memory select is active in these ranges, irrespective of the memory mode.

The RAM and CAN are enabled by clearing bit 0 of the SYSCON register.
The C515C CAN unit uses external data memory range [0xF700..0xF7FF].
The C515C also contains 2K on-chip RAM mapped to external data memory range [0xF800..0xFFFF].

5.3.1.1 The RROS Memory Mode

Jumpers CFG0, CFG1, and CFG2 are removed for the RROS mode. This is the default memory mode used with the READS51 environment. It is compatible with the earlier Rigel boards. It uses 32K of ROM (EPROM) and 32K of RAM. ROM and RAM are overlapped, so that user programs may be downloaded as data and executed as code. When the MON/RUN switch is in the "MON" or monitor position, ROM occupies the lower half of the 64K address range. In the "RUN" position, RAM occupies the lower half of the 64K address range.

The slide switch is used when downloaded programs need direct access to the interrupt vectors located in low memory. Such a program, with its origin at 0, is first downloaded into a RAM device placed in U6. While holding the RESET button down, S1 is moved to its RUN position. Now the program in U6 is in the low memory block, starting at address 0. Releasing the RESET button executes the user program in U6. The user program may then have direct access to all interrupt vectors.

5.3.1.2 The NVRAM Memory Mode

This mode is used for storing the user program in battery-backed non-volatile RAM. It assumes non-overlapped 64K of code memory and 64K of data memory. Code is placed into RAM when the MON/RUN switch is in the "MON" or monitor position. In the "RUN" position, the lowest 64K block of RAM is accessed as code memory and the next 64K block of RAM as data memory.

Slide Switch	Code Memory	Data Memory
MON	EPROM [0-FFFF]	RAM [0-FFFF]
RUN	RAM [0-FFFF]	RAM [10000-1FFFF]

The CFG1 jumper is used to toggle the A15 line so that the READS51 environment may be used to download upto 32K of user code into non-volatile RAM. Remove CFG1 if you are using READS51 to download user code.

5.3.1.3 The Banking Memory Mode

This mode supports code and data memory blocks larger than the 64K addressed by the microcontroller. Since the EPROM may be upto 128K, one additional address bit is needed (A16). Similarly, since the RAM may be upto 512K, three additional address bits are needed (A16, A17, and A18). These bits are not automatically generated by the processor, but must be asserted by other means, such as processor ports. Changing the high address bits not generated by the processor is often referred to as bank switching. Once the proper bank is switched into, the processor generates the address lines automatically.

There are two ways to generate the bank addresses A16, A17, and A18. Either Port 5 bits P5.0, P5.1, and P5.2 or the dual UART general-purpose output bits OP4, OP5, and OP6 may be used. The source of each bank address is determined by the jumpers S2, S3, and S4 respectively. Note that the port bits are at logic level high after a reset. The CPLD inverts the source signal so that the bank addresses are all at logic level low (0) after a reset. This assures that code is fetched from address 0 immediately following a reset.

6 HEADERS

The R-515JC board has ten headers: the I/O (input/output) header J20, 6 serial port headers J10, J11, J12, J13, J16, and J17, 2 headers for the UART input and output options, J14 and J15, and J4 the header for the JTAG programming of the CPLD.

6.1 J20 (I/O Header)

The I/O header contains Ports 1, 3, 4, 5, and 6 along with the HWPD#, Reset pins and CAN transmit and receive signals. The I/O header is a three-row 96-pin header. The first 25 pins of the first two rows are compatible with our I/O boards to allow for easy access to prototyping. Individual signals of these jumpers are listed below. Pin 1 may be identified as the post with the square pad on the printed circuit board.

Pin	Signal	**Pin**	Signal	**Pin**	Signal
1	GND	**2**	VCC	**3**	GND
4	GND	**5**	VCC	**6**	D0
7	P6.0	**8**	P6.1	**9**	D1
10	P6.2	**11**	P6.3	**12**	D2
13	P6.4	**14**	P6.5	**15**	D3
16	HWPD#	**17**	P6.7	**18**	D4
19		**20**	CANRS	**21**	D5
22	VAGND *	**23**	CANHI	**24**	D6
25	VAREF *	**26**	CANLO	**27**	D7
28	P4.0	**29**	P1.O	**30**	GND
31	P4.1	**32**	P1.1	**33**	A0
34	P4.2	**35**	P1.2	**36**	A1
37	P4.3	**38**	P1.3	**39**	A2
40	P4.4	**42**	P1.4	**42**	A3
43	P4.5	**44**	P1.5	**45**	A4
46	P4.6	**47**	P1.6	**48**	A5
49	P4.7	**50**	P1.7	**51**	A6
52	P5.0	**53**	P3.0	**54**	A7
55	P5.1	**56**	P3.1	**57**	A8
58	P5.2	**59**	P3.2	**60**	A9
61	P5.3	**62**	P3.3	**63**	A10
64	P5.4	**65**	P3.4	**66**	A11
67	P5.5	**68**	P3.5	**69**	A12
70	P5.6	**71**	P3.6	**72**	A13
73	P5.7	**74**	P3.7	**75**	A14
76	RSTIN#	**77**	P7.0	**78**	A15
79	RSTOUT#	**80**	SQW	**81**	A16
82	RSTOUT	**83**	CPUR#	**84**	A17
85	GND	**86**	XIOSEL#	**87**	A18
88	SR0	**89**	ALE	**90**	P3.6
91	ST0	**92**	PSEN#	**93**	P3.7
94	VCC	**95**	GND	**96**	GND

* The schematics use the designation P68 for VAGND and P69 for VAREF

6.2 Serial Port Headers

There are four serial ports on the board all of which have headers for access. Of the 4 available serial ports three are asynchronous serial ports. One of the asynchronous ports is a dedicated RS-232 port, the other two can be configured as RS-232 or RS-422 / RS-485 ports. The fourth serial port is a synchronous serial port. There are six headers to support the various serial port options.

6.2.1 J10, RS-232-P

J10 is the secondary connector for the primary serial port on the board. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header.

6.2.2 J11, RS-232-A

J11 is the secondary connector for the UART Channel A when used as a RS-232 port. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header.

6.2.3 J12, RS-232-B

J12 is the connector for the dual UART Channel B when used as a RS-232 port. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header

6.2.4 J16, RS-485-A

J16 is the connector for the dual UART Channel A when RS-422 / RS-485. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header

6.2.5 J17, RS-485-B

J17 is the connector for the dual UART Channel B when using the RS-422 / RS-485 port. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header

6.2.6 J13, SSC

J13 is the connector for the synchronous serial channel available on the board. Pin one is designated on the printed circuit board with a square pad. See the chart below for the signals found on this header.

PIN #	**HEADER**					
	J1O	**J11**	**J12**	**J13**	**J16**	**J17**
1	RXD	RXD	RXD	VCC	VCC	VCC
2	TXD	TXD	TXD	SR1	A	A
3	GND	GND	GND	ST0	B	B
4	XRST	RTS	RTS	SCLK	Z	Z
5	NC	CTS	CTS	SLS#	Y	Y
6	--	--	--	GND	GND	GND

6.3 J14, J15, UART Auxiliary I/O

The dual UART has 6 auxiliary inputs and 7 auxiliary outputs. Some of these are used for hardware handshaking (RTS and CTS) if so programmed. Also, three of the outputs may be used to generate the high address bits for bank switching. These inputs and outputs are also brought to two headers J14 and J15 for general-purpose use or testing. Please refer to the dual UART data book for details on programming the inputs and outputs. The pinouts for the headers are in the chart below. The square pad on the board designates pin one.

PIN #	**HEADER**	
	J14, U-IN	**J15, U-OUT**
1	VCC	VCC
2	IP2	OP2
3	IP3	OP3
4	IP4	OP4
5	IP5	OP5
6	IP6	OP6
7	GND	OP7
8	---	GND

6.4 J4, JTAG

The R-515JC board uses the Xilinx XC95C36 CPLD to decode memory maps. The header marked J4 on the board is the JTAG header used for programming the CPLD.

7 CPLD EQUATIONS

```
module r515jc
Title ’r515jc’

r515jc device;

Declarations
// --- inputs ---
A8        pin  8;
A9        pin  9;
A10       pin 11;
A11       pin 12;
A12       pin 13;
A13       pin 14;
A14       pin 18;
PA15      pin 19;
BA16X     pin  2;
BA17X     pin  3;
BA18X     pin  4;
PSENX     pin  7;
RDX       pin  5;
WRX       pin  6;
RSTINX    pin 34;
MON       pin 27;
AUX0      pin 20;
AUX1      pin 22;
AUX2      pin 24;

// --- outputs ---
ROMSELX   pin  1 istype ’com, pos’;
ROMRDX    pin 44 istype ’com, pos’;
RAMSELX   pin 43 istype ’com, pos’;
RAMRDX    pin 42 istype ’com, pos’;
XIOSELX   pin 26;
UARTSELX  pin 40;
TIMERSELX pin 39;
MA15      pin 38;
MA16      pin 37;
MA17      pin 36;
MA18      pin 35;
RSTOUT    pin 33;
RSTOUTX   pin 25;
LEDMONX   pin 28 istype ’com, pos’;
LEDRUNX   pin 29 istype ’com, pos’;

// --- nodes ---
ARMMODE  NODE istype ’reg, buffer’;
MONMODE  NODE istype ’reg, buffer’;

Equations
 RSTOUTX   = RSTINX;
 RSTOUT    = !RSTINX;
 XIOSELX   = !(PA15 & A14 & A13 & A12 & A11 & A10 &  A9 & !A8); // 0xFExx
 UARTSELX  = !(PA15 & A14 & A13 & A12 & A11 & A10 & !A9 &  A8); // 0xFDxx
 TIMERSELX = !(PA15 & A14 & A13 & A12 & A11 & A10 & !A9 & !A8); // 0xFCxx
// XSELX     = XIOSELX & UARTSELX & TIMERSELX;

 when(AUX2 & !AUX0) then
  {
"   ARMMODE.AR   = 1;              // flip-flop asynchronous reset
"   MONMODE.AR   = 1;              // flip-flop asynchronous reset
```

```
  // --- NVRAM mode ---
   when(!MON) then // MON mode
    {
     LEDMONX =  0;
     LEDRUNX =  1;
     when(!PSENX & XIOSELX & UARTSELX & TIMERSELX) then // EPROM access
      {
       MA15    =  PA15;
       MA16    =  0;
       ROMSELX =  0;
       ROMRDX  =  PSENX;
       RAMSELX =  1;
       RAMRDX  =  1;
       }
      else when( (!RDX # !WRX) & XIOSELX & UARTSELX & TIMERSELX) then // RAM access
       {
//        MA15    = !PA15;   // RROS inverts A15
        when(AUX1) then MA15=!PA15; else MA15=PA15;
        MA16    =  0;
        ROMSELX =  1;
        ROMRDX  =  1;
        RAMSELX =  0;
        RAMRDX  =  RDX;
        }
       else // indeterminant
        {
         MA15    =  0;
         MA16    =  0;
         ROMSELX =  1;
         ROMRDX  =  1;
         RAMSELX =  1;
         RAMRDX  =  1;
         }
     }
    else  // RUN mode
     {
      LEDMONX =  1;
      LEDRUNX =  0;
      MA15    =  PA15;
      ROMSELX =  1;
      ROMRDX  =  1;
      when(!PSENX & XIOSELX & UARTSELX & TIMERSELX) then // low RAM access
       {
        MA16    =  0;
        RAMSELX =  0;
        RAMRDX  =  PSENX;
        }
       else when( (!RDX # !WRX) & XIOSELX & UARTSELX & TIMERSELX) then // RAM access
        {
         MA16    =  1;
         RAMSELX =  0;
         RAMRDX  =  RDX;
         }
        else  // indeterminant
         {
          MA16    =  0;
          RAMSELX =  1;
          RAMRDX  =  1;
          }
      }
  // --- end NVRAM mode ---
  }
 else when(!AUX2) then
```

```
  {
"   ARMMODE.AR   = 1;               // flip-flop asynchronous reset
"   MONMODE.AR   = 1;               // flip-flop asynchronous reset
   // --- Bank Switching mode ---
   LEDMONX =  0;
   LEDRUNX =  0;
   MA15    =  PA15;
   MA16    =  !BA16X;
   MA17    =  !BA17X;
   MA18    =  !BA18X;
   when(!XIOSELX # !UARTSELX # !TIMERSELX) then // memory mapped IO select
    {
     ROMSELX =  1;
     ROMRDX  =  1;
     RAMSELX =  1;
     RAMRDX  =  1;
     }
    else
     {
      ROMSELX =  PSENX;
      ROMRDX  =  PSENX;
      RAMSELX =  RDX & WRX;
      RAMRDX  =  RDX;
      }
   // --- end Bank Switching mode ---
   }
 else when(AUX2 & !AUX1) then
  {
  // --- XEVA mode ---
  // MONMODE=1 : code memory is in EPROM, data memory is RAM
  // RUNMODE=1 : overlapped code and data memory is 64K of RAM
  // reset                   -> MONMODE=1, ARMMODE=0
  // (RD#=PSEN#=0 and A14=1) -> ARMMODE
  // ARMMODE and PSEN#=A14=0 -> RUNMODE

  MA16   =  !BA16X;
  MA17   =  !BA17X;
  MA18   =  !BA18X;

  ARMMODE.D    =  RSTINX;               // flip-flop data input
  ARMMODE.AR   = !RSTINX;               // flip-flop asynchronous reset
  ARMMODE.CLK  = !RDX & !PSENX & A14;  // flip-flop clock

  MONMODE.D    = !ARMMODE;              // flip-flop data input
  MONMODE.AP   = !RSTINX;               // flip-flop asynchronous preset
  MONMODE.CLK  = !PSENX & !A14;         // flip-flop clock

  LEDRUNX  = !ARMMODE.Q;                // ARMMODE.Q is the flip-flop output
  LEDMONX  = !MONMODE.Q;                // MONMODE.Q is the flip-flop output

  when(MONMODE.Q) then  // XEVA MON mode
//  when(!MON) then
   {
    LEDRUNX  = 1;
    LEDMONX  = 0;
    when(!PSENX) then // EPROM access
     {
      MA15    =  1; //PA15; Using 27C512’s -- upper 32K is Tele51
      ROMSELX =  !XIOSELX # !UARTSELX # !TIMERSELX;
      ROMRDX  =  0;
      RAMSELX =  1;
      RAMRDX  =  1;
      }
     else // when(!RDX # !WRX) then // RAM access
      {
       MA15    =  PA15;
       ROMSELX =  1;
       ROMRDX  =  1;
       RAMSELX =  (RDX & WRX) # !XIOSELX # !UARTSELX # !TIMERSELX;
       RAMRDX  =  RDX;
       }
    }
   else  // (MONMODE.Q==0) ... XEVA RUN mode
    {
     LEDRUNX  = 0;
     LEDMONX  = 1;
     MA15    =  PA15;
     ROMSELX =  1;
     ROMRDX  =  1;
     RAMSELX =  (PSENX & RDX & WRX) # !XIOSELX # !UARTSELX # !TIMERSELX;
     RAMRDX  =  PSENX & RDX;
     }
  // --- end XEVA mode ---
  }
 else
  {
  // --- RROS mode ---
"   ARMMODE.AR   = 1;               // flip-flop asynchronous reset
"   MONMODE.AR   = 1;               // flip-flop asynchronous reset

   MA15      = 0;
   MA16      = 0;
   MA17      = 0;
   MA18      = 0;

   when(!MON) then
    {
     LEDMONX  = 0;
     LEDRUNX  = 1;

     ROMSELX  =  PA15 # !(XIOSELX & UARTSELX & TIMERSELX);
     ROMRDX   =  RDX & PSENX;
     RAMSELX  = !PA15 # !(XIOSELX & UARTSELX & TIMERSELX);
     RAMRDX   =  RDX & PSENX;
     }
    else
     {
      LEDMONX  = 1;
      LEDRUNX  = 0;

      ROMSELX  = !PA15 # !(XIOSELX & UARTSELX & TIMERSELX);
      ROMRDX   =  RDX & PSENX;
      RAMSELX  =  PA15 # !(XIOSELX & UARTSELX & TIMERSELX);
      RAMRDX   =  RDX & PSENX;
      }
  // --- end RROS mode ---
  }

end r515jc
```

8 PARTS LISTS

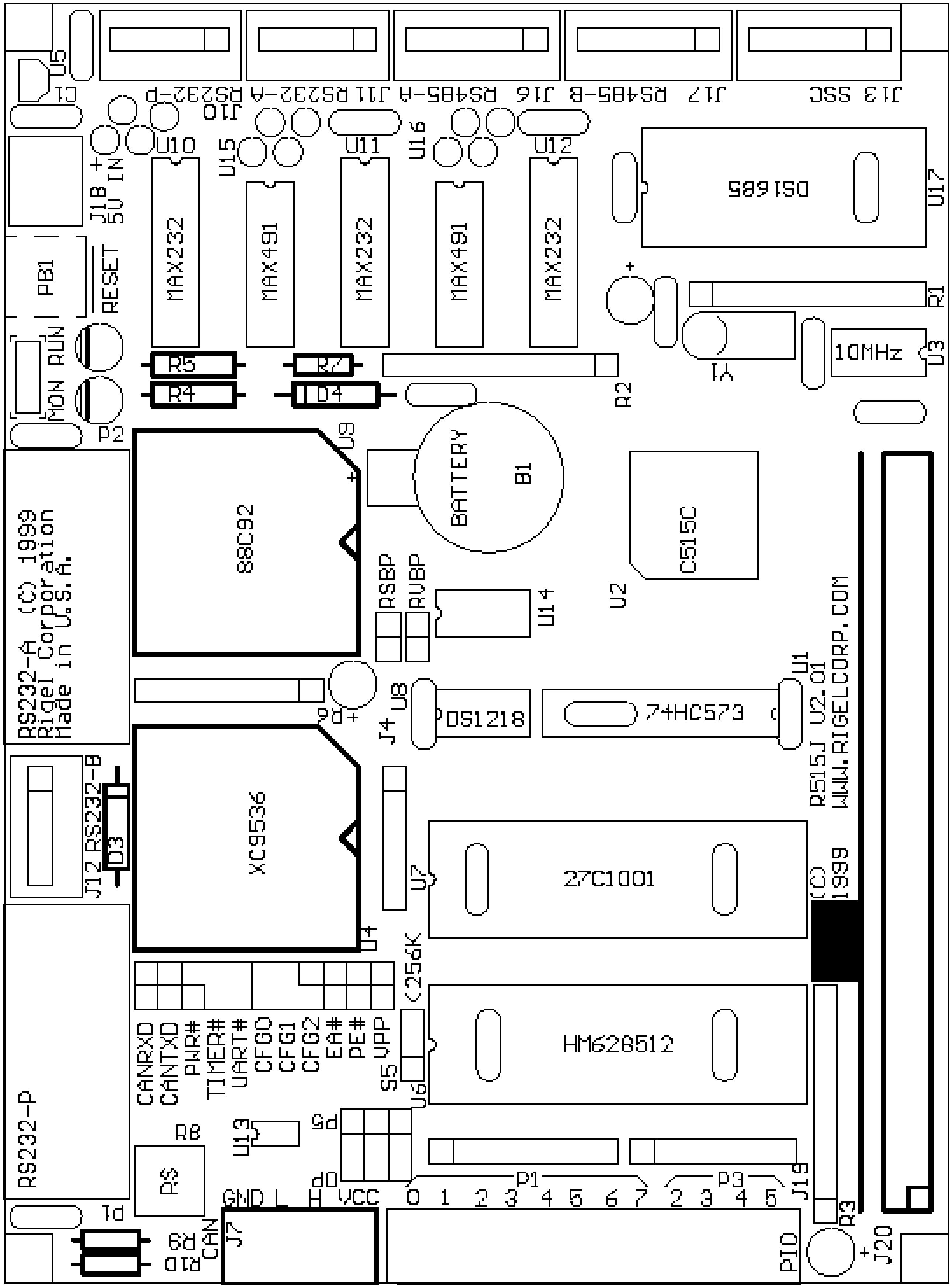
8.1 R-515JC Bill of Materials

The bill of materials given below lists all components by their reference as they appear on the board top overlay.
Revised: MAY1999
List Of Materials

Item	Quantity	Part	Reference	Description
	CAPACATORS			
1	19	10nF AXIAL	C1, C14-C31	Axial capacitor
2	12	1uF sm	C2-C13	Electrolytic capacitor
3	2	47uF 16V	C33, C34	Electrolytic capacitor
4	1	100uF 16V	C32	Electrolytic capacitor
	RESISTORS			
5	1	10K 8 Gang	R6	Gang Resistor
6	3	10K 10Gang	R1-R3	Gang Resistor
7	2	330 OHM 1/2W 5%	R4, R5	½ Watt 5% Carbon Resistor
8	1	1K 1/4W 5%	R7	¼ Watt 5% Carbon Resistor
9	2	10K 1/4W 5%	R9,R10	¼ Watt 5% Carbon Resistor
10	1	20K FLAT POT	R8	Flat 6mm Potentiometer
	DIODES			
11	1	LG Red LED	D1	LED T1 ¾ (red)
12	1	LG Green LED	D2	LED T1 ¾ (green)
13	2	1N4001	D3, D4	Diode
	CONNECTORS			
14	2	DB9 FEMALE	P1, P2	DB9 Connectors
15	1	2X 32	J20	.100 2X 32 Header
16	1	1X 32	J20	.100 1X 32 Header
17	3	2 Position Small TB	J18, J7	Terminal blocks (14)
18	4	3 Position Small TB	J19	Terminal block (2)
19	1	Battery Holder	B1	Battery Holder
20	1	Battery		Battery
21	1	Push Button	PB1	6 mm Pushbutton
22	1	Slide Switch	S1	Slide Switch
23	3	1X 5 Shrouded	J10, J11, J12	1X 5 Shrouded Header
24	3	1X 6 Shrouded	J13, J16, J17	1X 6 Shrouded Header
25	3	1X 3 headers	S4, S5	.100 1X 3 Headers
26	2	1X 2 Header	J21, J22	.100 1X 2 Headers
27	1	1X 6 Header	J4	.100 1X 2 Headers
28	1	1X 7	J14	.100 1X 7 Headers
29	1	1X 8	J15	.100 1X 8 Headers
30	1	2 X 11	J1/J2/J3/J5/J6/J8/J9	.100 2X 11 Headers
31	1	2X 3	S2/S3	.100 2X 3 Headers
	SOCKETS			
32	3	8 DIP	U3, U8, U14,	8 Pin Dip Socket
33	2	14 DIP	U15, U16	14 Pin Dip Socket
34	3	16 DIP	U10, U11, U12	16 Pin Dip Socket
35	1	20 DIP	U1	20 Pin Dip Socket
36	1	24 DIP (WIDE)	U17	24 Pin Dip Socket
37	2	32 DIP	U6, U7	32 Pin Dip Socket
38	2	PLCC44	U4, U9	44 Pin PLCC Socket
	ICS			
39	1	XC9536-15PC44C	U4	PLD
40	1	74HC573	U1	Octal latch

41	1	88C92	U9	Dual UART chip
42	1	27C256 / 512	U7	32K EPROM
43	1	62256 / 68512	U6	32K Static RAM
44	1	C515C	U2	Microcontroller
45	1	PCA 82C250	U13	CAN Interface IC
46	3	MAX232	U10,U11,U12	RS-232 IC
47	1	DS1233-10	U5	Reset Chip
48	1	DS1218	U8	Memory Battery Back-up IC
49	1	DS1685-5	U17	Real Time Clock
50	2	MAX491CPD	U15, U16	RS422/485 IC
51	1	32.768kHz	Y1	Crystal
52	1	10MHz Clock	U14	Microcontroller Clock
53	1	3.6864MHz Clock	U3	UART Clock

9 TOP OVERLAY



1O CIRCUIT DIAGRAMS