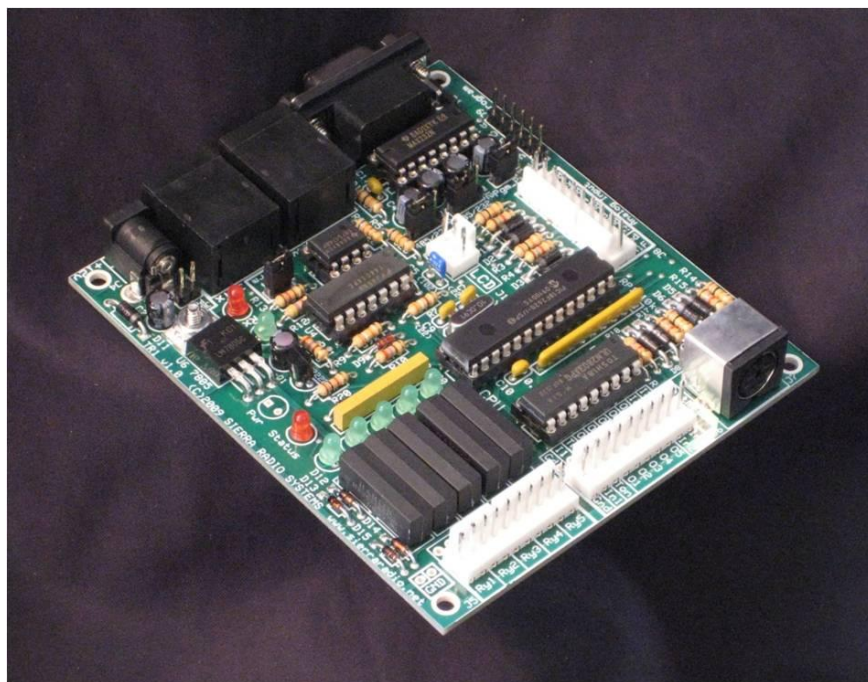


Sierra Radio Systems

TR1 Board

“Do-It-Yourself” Project Documentation



Version 1.0
October 2009



Do-It-Yourself Projects With the TR1

Introduction

The TR1 was originally designed to perform a specific function - to read the trigger input pins to determine if the transmit sequence should be initiated. When triggered, each output is turned on in order with a programmed delay between each output transition. The hardware for this application is typical of many control applications, read a set of inputs, process them to determine what, if any outputs should change. With a variety of inputs and outputs, the TR1 board can be used with the standard TR sequencer firmware or re-programmed for many applications.

Hardware Overview

The TR1 board is based on the Microchip PIC 18F2620 microcontroller CPU. This processor runs at 40 MHz and can store 64 KB of flash programmed firmware and 4 kb of working RAM. In addition to the CPU, the TR1 board provides all the necessary input / output signal conditioning and interfaces that would be convenient for many projects.

Hardware features

- Power regulation
- 3 buffered digital inputs
- 5 buffered digital outputs with slaved SPST reed relays with LEDs
- 4 analog to digital converter inputs with 10 bits of resolution
- A serial UART with either RS-232 or RS-485 interface
- 2 unassigned / unfiltered IO pins
- RS-485-based Sierra Bus multi-drop network support
- Second RS-232 output to drive a serial LCD display

Software Development Tools

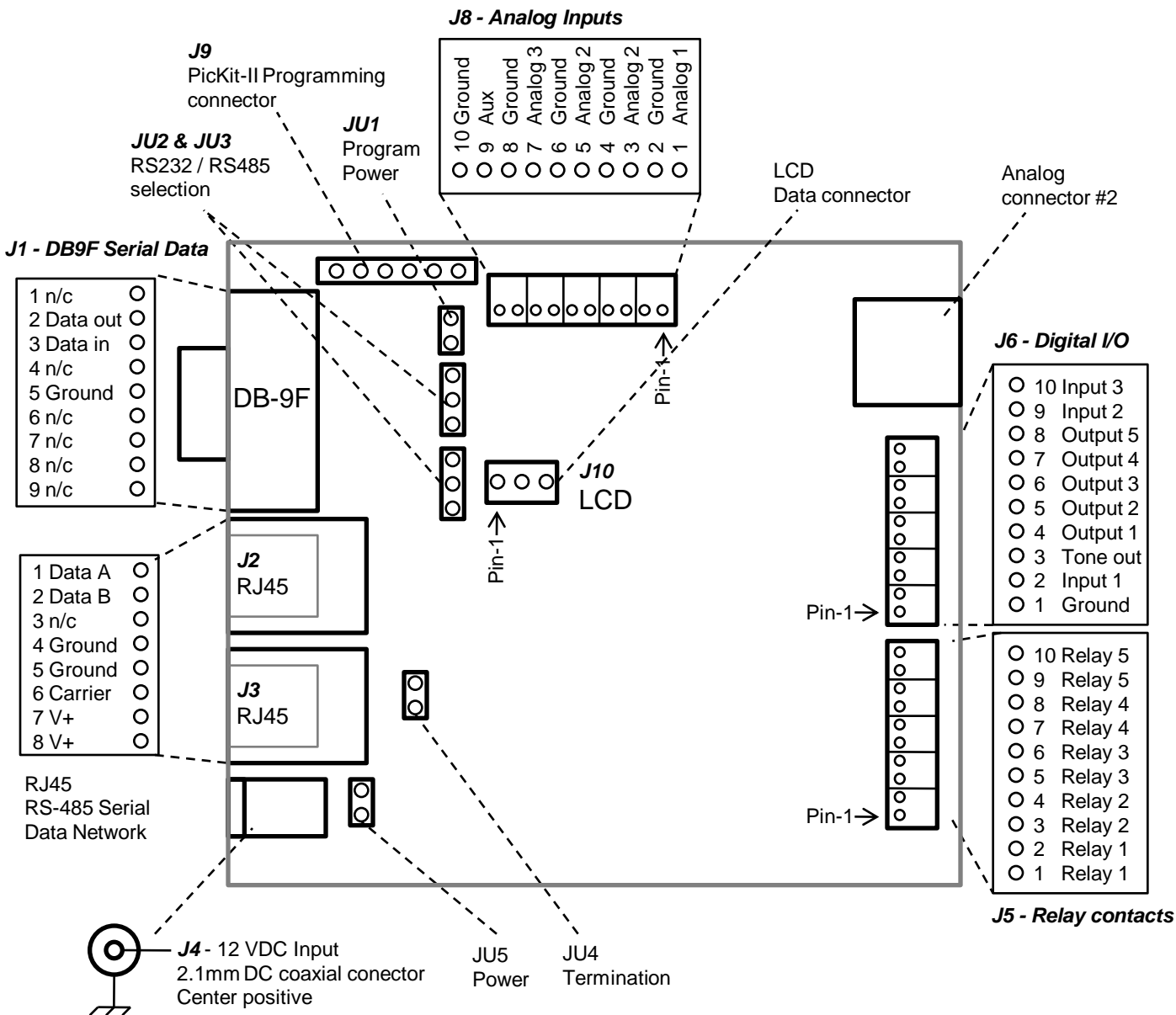
Firmware can be developed using a variety of compilers and assemblers. Some very good compilers include Microchip C18 compiler, Proton Basic, from Crownhill Associates, and PicBasic Pro from ME Labs. Most commercial development tools are free to a few hundred dollars. In addition to the compiler, an in-circuit programmer is required to download the firmware into the board. The TR1 board is designed to use the inexpensive Microchip PicKit2 programmer.

The process of firmware development is very simple.

1. Write the source code for the program using any text editor
2. Compile the program into a .hex file
3. Erase the contents of the program flash memory of the CPU using the in-circuit programming software
4. Use the in-circuit programmer to transfer the .hex file into the TR1 board's CPU

Your project is now up and running.

TR1 Hardware Details - I/O Connections



Board & Processor Pin Assignments

Digital IO Connector Pins (J6)

<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
	1	Ground	
7 RA5	2	Digital input 1 "In1"	Pulled up to +5v
13 RC2/CCP2	3	Tone out "Ton"	User programmable
15 RC4	4	Digital output 1 "O1"	Open collector
16 RC5	5	Digital output 2 "O2"	Open collector
21 RB0	6	Digital output 3 "O3"	Open collector
22 RB1	7	Digital output 4 "O4"	Open collector
23 RB2	8	Digital output 5 "O5"	Open collector
24 RB3	9	Digital input 2 "In2"	Pulled up to +5v
25 RB4	10	Digital input 3 "In3"	Pulled up to +5v

Relay IO Pins - Slaved to outputs 1...5 (J5)

<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
15 RC4	1 & 2	SPST reed relay 1	Dry contacts
16 RC5	3 & 4	SPST reed relay 2	Dry contacts
21 RB0	5 & 6	SPST reed relay 3	Dry contacts
22 RB1	7 & 8	SPST reed relay 4	Dry contacts
23 RB2	9 & 10	SPST reed relay 5	Dry contacts

Analog Input Pins (J8)

<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
2 RA0	1	Analog input 1 "A1"	0-25v DC scaled to 0-5v
	2	Ground	
3 RA1	3	Analog input 2 "A2"	0-25v DC scaled to 0-5v
	4	Ground	
4 RA2	5	Analog input 3 "A3"	0-25v DC scaled to 0-5v
	6	Ground	
5 RA3	7	Analog input 4 "A4"	0-25v DC scaled to 0-5v
	8	Ground	
14 RC3	9	Aux input	RS485 network carrier detect
	10	Ground	

RS-232 Serial Data IO Pins (J1)

<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
17 RC6	DB9-2	RS-232 data out	RS-232
18 RC7	DB9-3	RS-232 data in	RS-232

LCD Out Pins (J10)

<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
6 RA4	1	RS-232 data out	RS-232
	2	+5v DC to power serial LCD display	
	3	Ground	

CPU Pins with no PCB IO connection

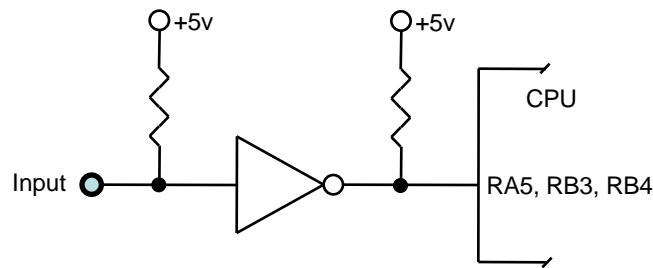
<u>PIC pin</u>	<u>PCB pin</u>	<u>Function</u>	<u>Board IO type</u>
11 RC0	None	RS485 network carrier assert (RJ45-6)	
12 RC1	None	Status LED	0=Lights LED

Digital Inputs, Outputs, Relays

Digital Inputs

The digital inputs are buffered using an inverting CMOS buffer chip to help protect the input to the CPU. When the input is unconnected or pulled up to +5VDC by an external device, the CPU will see this as a logic 0 (because of the inverting buffer). When the input pin is pulled to ground, the inverting buffer will put a logic 1 at the input pin of the CPU. The digital input pin at the IO connector is pulled up to +5v, or logic "1" through a 10k resistor which results in a logic 0 at the CPU after the inverting buffer.

Equivalent circuit:



Digital Outputs and Relay Contacts

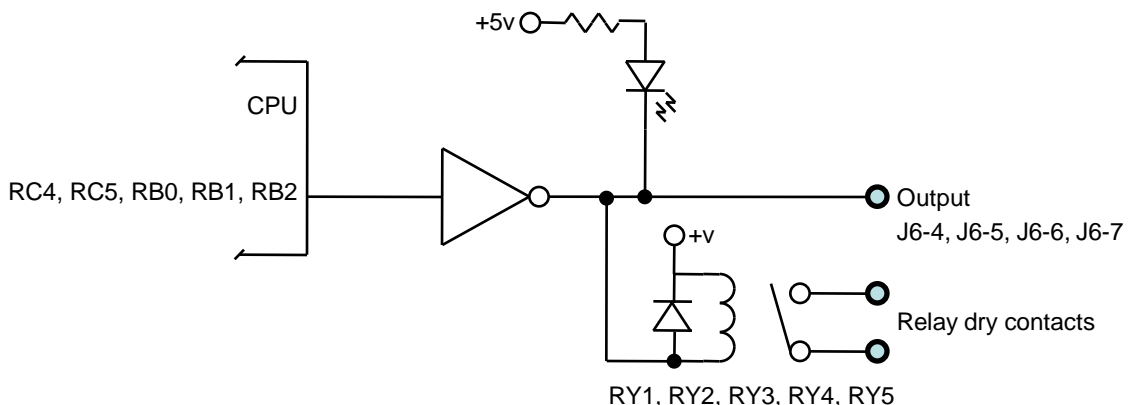
The digital outputs use an open collector driver transistor capable of sinking up to 500ma. When the CPU presents a logic 1 at the output pin, the inverting buffer will pull the open collector output to ground. When the CPU presents a logic 0, the output will float in the open collector mode.

The digital outputs also drive green LEDs to indicate when the output is active (ie: pulled to ground).

Each digital output pin is also connected to a SPST reed relay capable of passing up to 500ma. When the output on the CPU is set to a logic 1, the open collector output pin is pulled to ground, activating the reed relay and closing the SPST dry contacts.

When the output on the CPU is set to a logic 0, the output pin is in the open collector mode and the relay contacts are in the open position.

Equivalent circuit:

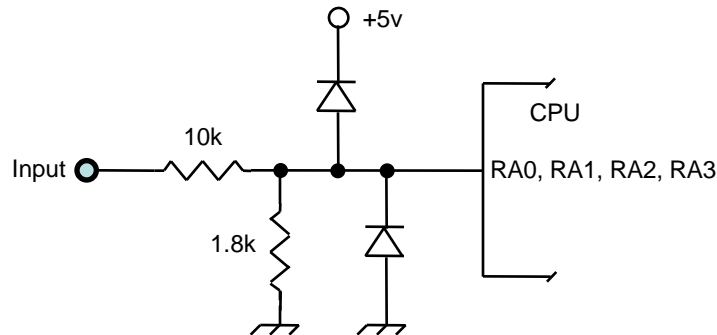


Analog Inputs

Analog Inputs

The analog input can measure “slow moving” DC analog voltages. The A to D converters built into the CPU can accept a voltage range from 0-5 V DC. The board’s input circuitry uses diode protection to prevent overvoltage conditions from damaging the CPU and a voltage divider to scale the input to allow measurement of voltages from 0 to 28 VDC.

Equivalent circuit:



The voltage divider will scale the input voltage down to an acceptable range of 0-5 VDC. This means that the maximum input voltage can be 28V at which point you hit the 5v upper limit. Care must be taken not to drive the input over 28 volts. Keep in mind that this is the absolute maximum input voltage in the standard configuration.

The A to D converters inside the PIC CPU can be configured to provide either 8 bit or 10 bit resolution. In 8 bit mode, the input voltage range of 0-28 volts is divided into 256 voltage steps resulting in a resolution of just over 0.1 v. When using the 10 bit resolution mode, you can achieve a resolution of approximately 0.025 v.

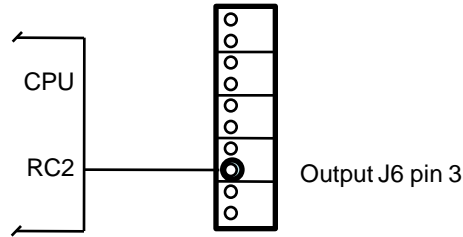
If you need higher resolution at lower maximum voltages or higher DC voltage support, you can change the values of the voltage divider resistor. The standard value is 1.8k Ohms. These resistors designated as R2, R4, R15, and R17 can be removed from the board and other resistors of higher value used instead. For example, replacing R2, the voltage divider on analog input #1 (going to AN0 CPU pin 2) with a value of 4.7k will read from 0-10.6 volts with a resolution of ~0.01 volts per division when using the ADC in 10 bit mode.

The diodes are used to clamp the incoming signal to a maximum of 5 volts and a minimum of 0 volts. This protects the input to the CPU from an over or under voltage condition.

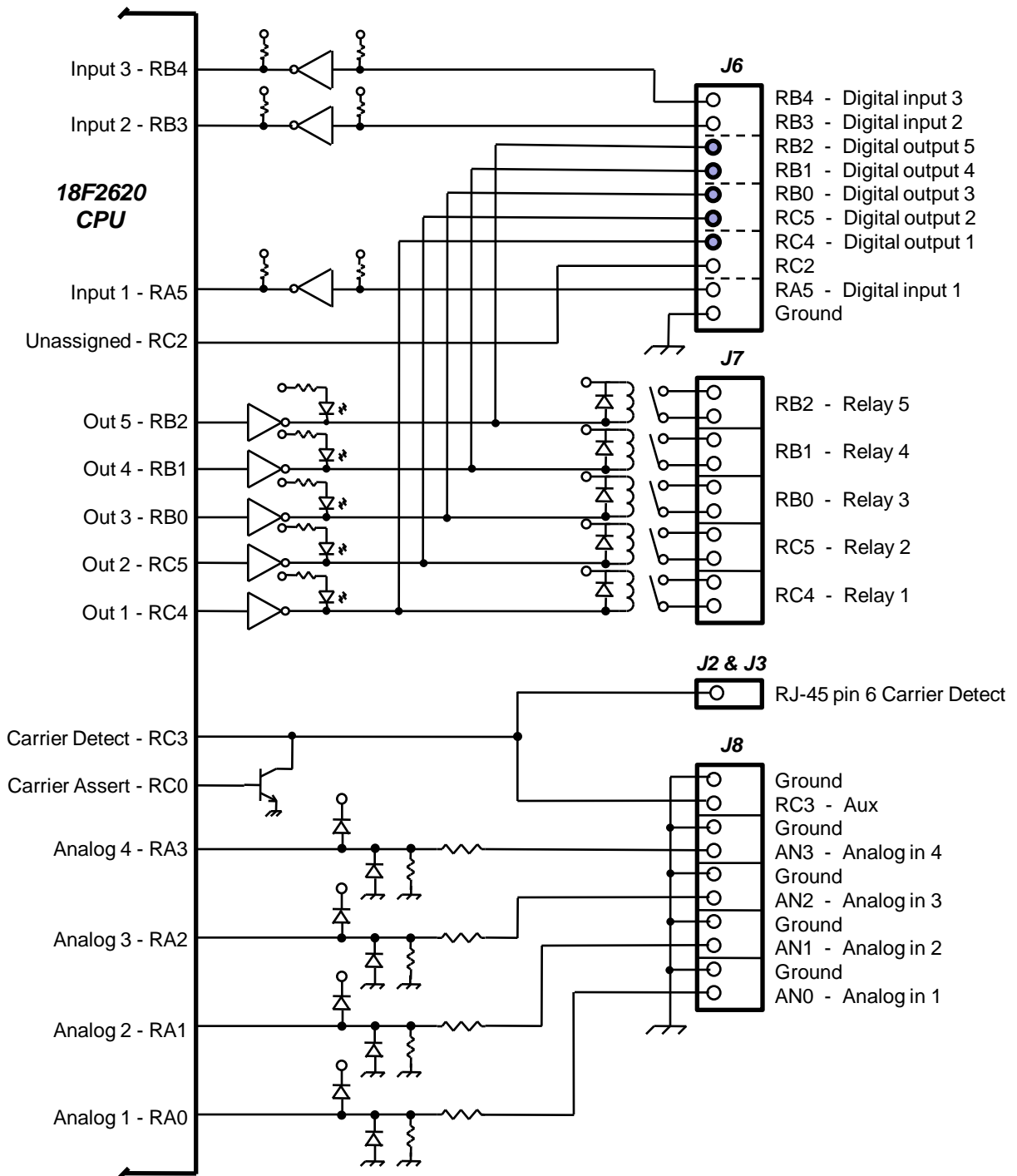
Unassigned / User Programmable Pin

The unassigned CPU pin 13, RC2, may be used as a general purpose IO pin or with the built in CCP2 function. As a general purpose IO pin any pullups, pulldowns, or buffers will have to be supplied external from the board. Using the pin in the CCP mode, the built in PWM output can be used to generate audio tones with a proper signal smoothing filter. The pin may also be used to drive a power transistor to control a motor.

Equivalent circuit:



I/O Schematic



RS-232 & RS-485 Serial Network IO

Introduction

The CPU communicates to the outside world through a serial UART in the CPU chip. The TR1 board supports two physical interfaces: traditional RS-232 for direct “device to computer” communications and a multidrop serial network that uses RS-485 differential signaling which allows multiple devices to talk to a master PC or each other. In either case, the TR1 board must be jumper configured for one mode or the other and does not support both at the same time on a single board.

Direct RS-232 Communications

The RS-232 interface provides standard EIA RS-232 voltage signaling at J1, a DB-9 female connector. DB9 pin 2 is RS-232 data out from the board, DB9 pin 3 is RS-232 data into the board, and DB9 pin 5 is ground. Hardware handshaking is not supported. The baudrate and other communications parameters are set by the program in the PIC CPU. The most common, reliable mode is 9600 baud, N81 signaling with no hardware handshaking.

RS-485 Multi-drop Serial Network

This alternative communications channel replaces the RS-232 interface which references logic 1 and 0 using positive and negative voltages referenced to ground with a differential signaling pair of wires called signals A and B. Differential signaling allows the physical serial cable to be many hundreds or even thousands of feet long. This two-wire system runs in a half-duplex mode which means that data is either being transmitted or received by a device at any given time but can not do both at the same time. This allows multiple devices to be connected or “bussed” on the same pair of “AB” wires eliminating the need to directly connect each receiver to each sender with a separate set of wires. This greatly simplifies the physical installation. RS-485 networks can be built in various topologies including backbone, hub or daisy-chain configurations.

The Sierra Bus Implementation of the RS-485 Network

As useful as RS-485 can be, there is no established physical connection or hardware flow control standard. The Sierra Bus extends the basic RS-485 electrical signaling technique by adding these capabilities.

The physical connection uses commonly available Ethernet cable and RJ-45 connectors. Ethernet cable comes prebuilt in various lengths, provides good strain relief, and more than enough wires. Ethernet cable provides 4 twisted pairs of wires. The RS-485 data only requires 2 of those wires. Good engineering practice calls for adding a ground wire to keep down the noise. We also add a carrier detection signal and power.

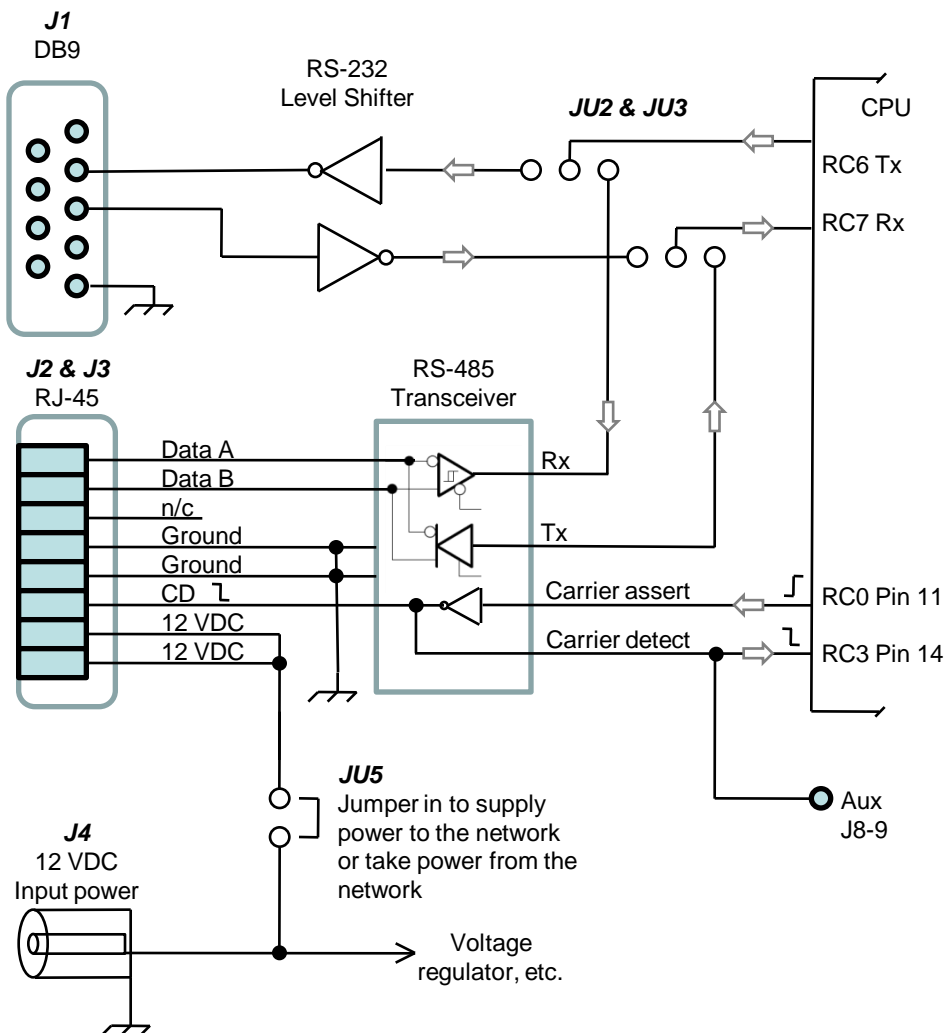
The carrier detection signal is asserted when a device wants to transmit on the network. Each device listens to that signal and when it is active, the device knows that another device or computer is transmitting and that it should be listening for traffic that may be sent to it. It also should prevent any listening device from transmitting and causing a data collision on the network.

RS-232 & RS-485 Serial Network IO

Power can also be supplied on the Sierra Bus' Ethernet cable. This power is +12 VDC and can be either supplied by the board itself or an external device. The power mode is jumper selectable on the TR1 board.

The combination of RS-485 signaling, carrier detection and power over the cable allows you to create a network of devices deployed over a large physical area and located in locations where there is no local power such as radio towers, outbuildings, etc.

Equivalent circuit:



Configuring The TR1 As A RS-232 to RS-485 Converter

The TR1 board can be built or modified to be used as a signal converter between a RS-232 port on a computer to a RS-485 Sierra Bus network.

Install the following components...

Connectors

- DB9 connector
- RJ-45 connector (1 or 2)
- DC power jack

Integrated Circuits

- 7805 regulator
- MAX232 IC and socket
- 76175 RS-485 and socket
- 74AC14 and socket

Capacitors

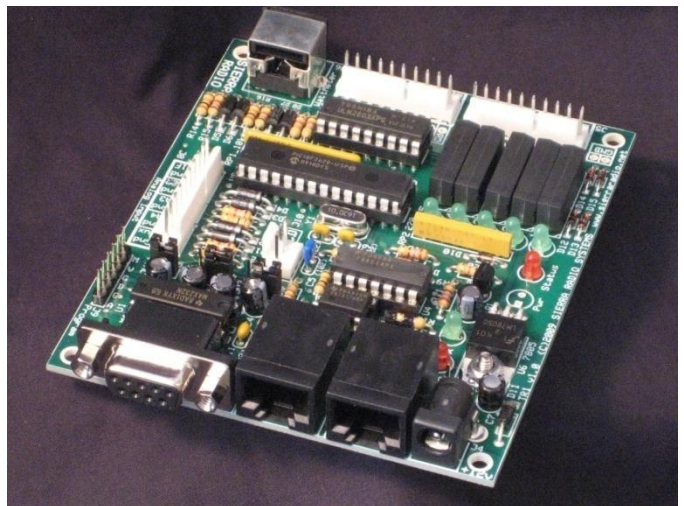
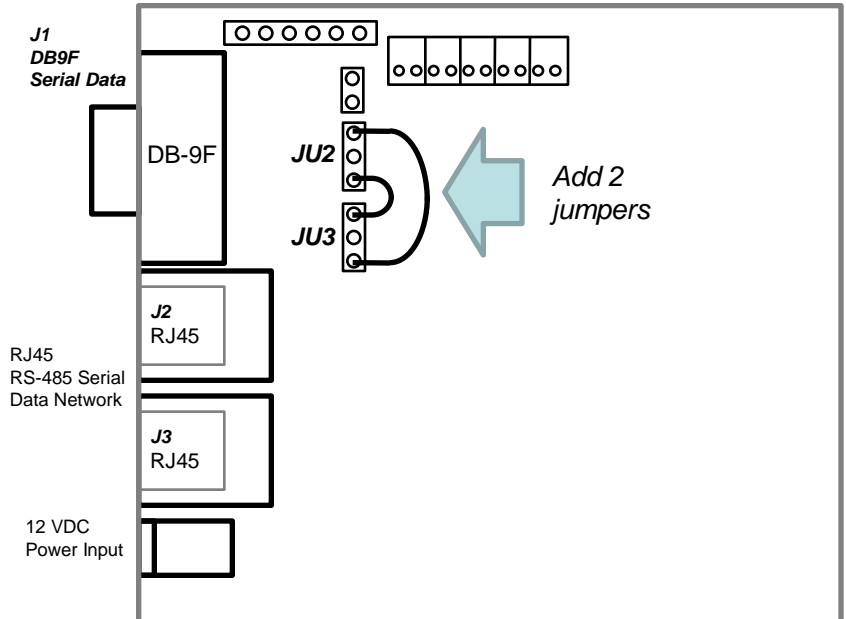
- C1, C2, C3, C4 - 1uf caps
- C11 0.1
- C6 10uf
- C7 10uf

Resistors

- R9 - 39k
- R10 - 270
- R11 - 270
- R20 - 1k
- R21 10k

Diodes and Transistors

- Q1 - 2N2222
- D9 - 1N4448
- D11 - 1N4004 or similar
- Rx, Tx LED's



Built-In TR1 Commands

One option when building your own projects with the TR1 board is to use the standard TR1 board firmware and write your own PC program to control the TR1 board. The following commands are available in the standard TR1 firmware.

Software Configuration Basics

When using the TR1 in the normal configuration, connect a serial interface cable between the TR1 board and the computer. All configuration parameters can be set by using a dumb terminal or through the TR1's configuration program.

When the TR1 board is powered up, it will display the current operating parameters on the config program or send them to the dumb terminal. The default on/off delays are 100 ms between activating or deactivating the outputs. If you need different delays, simply enter the values in the config program or type the command followed by the delay in ms.

For example, to set the delay from initial trigger input pulling to ground and the first output going low after 50 ms., enter:

```
//ONDELAY01 50
```

The command is composed of “//” followed by “ONDELAY” followed by a zero “0” followed by the output number “1” to “5”, followed by a space then the value of the delay in milliseconds. The zero in the middle is used to maintain command compatibility with devices that support more than 9 outputs.

The corresponding off delay is a similar format, for example an off delay of 200 from the 4th output to the 3rd output would be:

```
//OFFDELAY03 200
```

The initial characters “//” are used to tell the TR sequencer that a new command follows. Note that if the command is not entered exactly right, nothing will be executed and you will not get any indication from the TR sequencer that you entered a bad command. To check the current configuration you can enter the “status” command

```
//status
```

This will list all current operating parameters.

Once you have set the operation parameters you want to keep, enter the “save” command and all current operating parameters will be written to the on-board flash memory and will be reloaded automatically the next time the TR sequencer is powered up.

```
//save
```

Advanced User Commands

Advanced sequencing and I/O control commands

The TR1 has many commands that can be used to set various operating parameters beyond the basic setup commands. These commands follow the same syntax as the basic commands.

<code>//enable x</code>	Enable output x (1-5)	Default is all outputs enabled
<code>//disable x</code>	Disable output x (1-5)	

Enables or disables output x in the sequence. If an output is disabled, when it comes time to transition the output to pull to ground, that output will be skipped and left in a floating state as if it did not exist.

<code>//normalon</code>	Turn on sequence order set to 1, 2, 3, 4, 5	Default
<code>//normaloff</code>	Turn off sequence order set to 5, 4, 3, 2, 1	Default
<code>//reverseon</code>	Turn on sequence order set to 5, 4, 3, 2, 1	(opposite of <code>//reverseoff</code>)
<code>//reverseoff</code>	Turn off sequence order set to 1, 2, 3, 4, 5	(opposite of <code>//reverseon</code>)

The normal/reverse on/off commands let you change the order of outputs turning on and off with a single command. This is used in very limited cases and the default should be used for typical applications.

<code>//out0x on</code>	Turns on output x (1-5)	Example: " <code>//out04 on</code> " turns out output 4
<code>//out0x off</code>	Turns off output x (1-5)	

The `//out` commands allow you to manually force an output to the on or off state. This is helpful for testing individual output signals.

Build in CW Keyer

The TR1 has a built in CW keyer that will take a command string and send it in CW. This is not used in normal TR sequencing, it can be handy for some applications. When the TR1 is set to "keyer" mode, output 5 will send a string of characters in CW.

<code>//keyer on</code>	Turns on keyer mode (default)
<code>//keyer off</code>	Turns off keyer mode, back to TR sequencer mode

<code>//cw xxxxx</code>	Sends the string xxxxx in CW to putput #5
-------------------------	---

<code>//wpm xx</code>	Sets the CW speed to approximately xx words per minute
-----------------------	--

For example:

`//cw cq cq cq de k6abc` will send the characters "CQ CQ CQ DE K6ABC" to output \$5

Advanced User Commands

Analog Inputs

The TR1 has 4 analog inputs which are constantly being monitored. Each input can measure a slow moving DC voltage between 0 and 30 VDC in approximately 0.1 volt increments. In normal operation the analog inputs are not used, however they can be used determine if the TR sequencing should be allowed to continue or be inhibited. The behavior of the TR sequencer is set by the following commands.

//rad Read and display all A/D input values.

Interactive command to read the current values of the 4 A/D converters.

//ad1min x.xxx Set minimum acceptable voltage to x.xxx volts (0-30 VDC)
//ad2min x.xxx Set minimum acceptable voltage to x.xxx volts (0-30 VDC)
//ad3min x.xxx Set minimum acceptable voltage to x.xxx volts (0-30 VDC)
//ad4min x.xxx Set minimum acceptable voltage to x.xxx volts (0-30 VDC)
//ad1max x.xxx Set maximum acceptable voltage to x.xxx volts (0-30 VDC)
//ad2max x.xxx Set maximum acceptable voltage to x.xxx volts (0-30 VDC)
//ad3max x.xxx Set maximum acceptable voltage to x.xxx volts (0-30 VDC)
//ad4max x.xxx Set maximum acceptable voltage to x.xxx volts (0-30 VDC)

The adxmin and adxmax commands are used to set the min and max threshold voltage that is an acceptable operating range. If the sampled voltage drops below the minimum or goes above the maximum voltage, the TR1 will be put in an alarm condition. The default alarm sends a text message to the serial port. The TR1 configuration program constantly monitors the messages sent from the TR1 board. When the alarm condition is triggered, the configuration program will alert the operator.

The TR sequencer can also be set to inhibit sequencing based on an alarm condition.

//adcal1 x.xxx Sets the voltage calibration offset for AD input #1 to x.xxx volts
//adcal2 x.xxx Sets the voltage calibration offset for AD input #1 to x.xxx volts
//adcal3 x.xxx Sets the voltage calibration offset for AD input #1 to x.xxx volts
//adcal4 x.xxx Sets the voltage calibration offset for AD input #1 to x.xxx volts

Due to variations in component values on the input voltage divider, each AD input may report a slightly different voltage value. These variations are typically up to +/- 10%. To offset these variations, the user can optionally add an offset to each AD converter input. The offset value should be between -30.0 and +30.0 volts but are typically less than one volt.

Advanced User Commands

Using a Wavenode directional coupler

Wavenode makes a very nice high speed digital wattmeter. They interface their wattmeter to the feed line with a directional coupler that is used to sample the forward and reflected voltages on the feed line. The 6 pin mini-DIN connector is plug compatible with the Wavenode directional couplers. This means that you can monitor the forward and reflected voltages on the feedline while operating the TR sequencer. While the TR1 is not designed to be a highly accurate wattmeter, it can be used to detect a disconnected antenna or other major antenna system failure. All that is required is the coupler. Analog input #1 is connected to the Wavenode forward power to the antenna and analog input #2 is connected to the reflected power of the coupler. When using the mini DIN connector for the coupler, do NOT connect any other inputs the 10 pin analog input header connector pins 1 or 3. These are the normal pins used for analog input 1 and 2 respectively.

```
//wm off           Turn off monitoring the coupler. Regular analog input monitoring
//wm on           Turn on monitoring the coupler.
//wm custom       Turn on monitoring the coupler and use the custom trigger voltage.
//wmlimit x.xxx   Set the custom trigger voltage to x.xxx
```

When the “//wm on” condition is set the TR1 reads the forward and reflected feedline voltage. When the reflected voltage is equal to or greater than the forward voltage, the TR sequence will be inhibited.

While this is not a very accurate mode, it is a quick way to check to make sure there is no major antenna system failure like a broken feedline, missing antenna or loose connector.

To calibrate the watt meter checking mode, set the limit voltage with the “//wmlimit x.xxx” command and set the mode to customer with the “//wm custom” command. This will interrupt the TR sequence when the reverse voltage is above x.xxx volts. To determine the proper value, you should put a digital voltmeter on pin 3 of the 10 pin analog input connector. Transmit into the coupler to the antenna. Reproduce the effect you want to use when you want to inhibit the TR sequence. Note the voltage and enter that value with the “//wmlimit x.xxx” command.

Configuration save, restore and status commands

```
//save           Take the current operating parameters and saves them to flash memory.
//load           Take the operating parameters stored in flash memory and loads them into the working
                variables.
//reset          Reset all operating parameters to the default state.

//reboot         Soft reset. Restarts the TR sequencer after loading parameters from flash memory.

//status         Lists all operating parameters in human readable form.
//ping           Checks to see if the TR sequencer is working. Returns the health, and network address.
//state          Returns the current values of the outputs, analog inputs, and other operating state.
                Uses the networked addressed format.
```

Advanced User Commands

Unofficial Commands

These commands are used for system testing and software development. These commands may come and go with various versions of firmware.

<code>//eyecandy</code>	Force outputs to turn on and off in sequence causing the LEDs to chase back and forth.
<code>//srdebugon</code>	Generates a lot of messages used to debug the software
<code>//srdebugoff</code>	Turns off debug messages
<code>//netdebugon</code>	Turns on network traffic debugging
<code>//netdebugoff</code>	Turns off network traffic debugging
<code>//forcefail</code>	Forces the TR sequencer into an inhibited state and will not accept commands until a hardware reset.
<code>//echo xxxxx</code>	Returns the payload xxxxx with full addressing
<code>//msgdirect</code>	Sets messaging mode to direct
<code>//msgnet</code>	Sets messaging mode to network

Advanced Applications

Multi-Unit Network Configuration for Contest Stations Using the SierraBus

The TR1 supports multiple units on a twisted-pair local area network called the SierraBus. The SierraBus network uses the RS485 half duplex signaling interface. This allows several units to be multi-dropped off the same pair of wires. The SierraBus uses common CAT5 ethernet cable and RJ45 connectors for the physical network cable. In addition to the two wires used for data, the SierraBus also provides a hardware carrier detect signal, ground and power. The SierraBus allows you remotely locate TR1 board anywhere along the bus cable for a really distributed network. Power must be supplied at one node and all TR1 board maybe powered from the network or locally with its own power adapter.

In an SierraBus networked configuration, the serial commands to configure and control the TR1 boards are sent in an addressed packet format. Every node on the network has a unique address and error detection is built into the protocol.

Device address assignments

00	Master PC
01 – 15	TR sequencers #1 to #15
16 – 98	Reserved
99	Broadcast to all devices

Network commands

<code>//net</code>	Puts the TR sequencer in network address mode
<code>//local</code>	Puts the TR sequencer in local (direct) address mode (default)

<code>//setaddr xx</code>	Set TR sequencer board network address to xx
---------------------------	--

All TR1 boards come pre-programmed with an address of 1. The user can select any address between 1 and 15. Legal values are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15. No leading 0 is required for addresses from 1 to 9.

<code>//getaddr</code>	Return the current address set in the TR1 board.
------------------------	--

<code>//silent on</code>	Turns off messages
<code>//silent off</code>	Turns on messages

When various commands are executed, the TR1 board will reply with various text messages. Putting the TR1 board in silent mode will suppress the generation of the messages. Commands will still be executed.

<code>//boot silent</code>	Turn off boot message.
<code>//boot verbose</code>	Turns on boot message.

When the TR1 board is powered up or rebooted, it will send a long message to the serial port listing all operating parameters. Setting the boot mode to silent will turn off the boot up message generation. Recommended for network installations to minimize network traffic.