Reference Manual

Version 1.0

Contents

Hardware

- Xbee backpack board
- Xbee base station
- Xbee firmware configuration
- RS485 network power injector

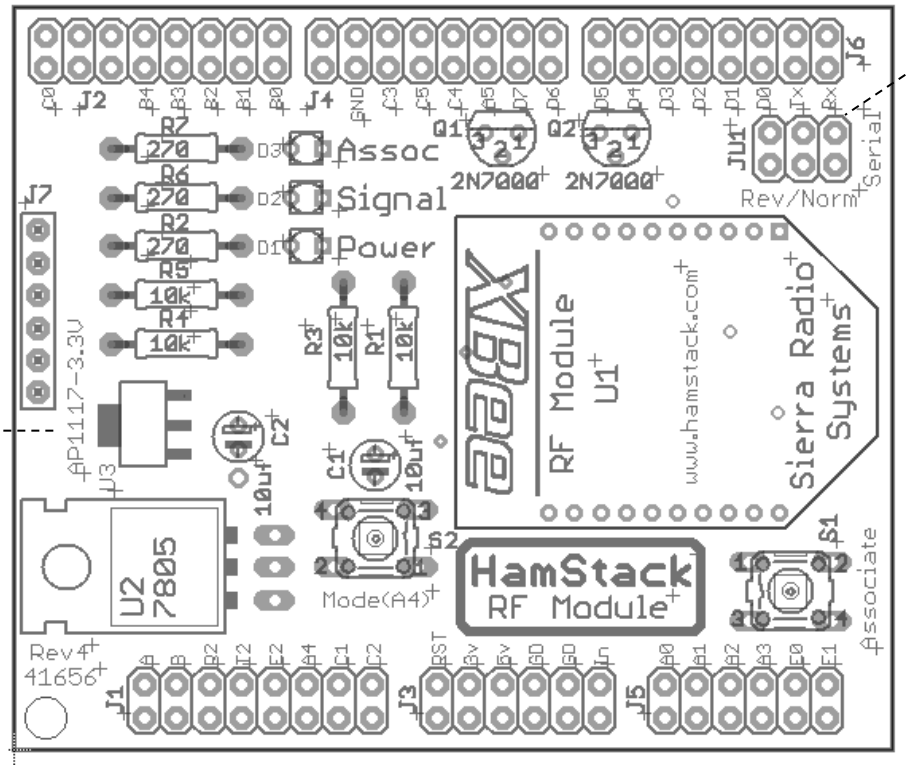
Protocol specification

- StationStack control protocol

For more information, go to the Sierra Radio Systems web site at www.sierraradio.net or www.hamstack.com

xBee Mesh Network Backpack Board

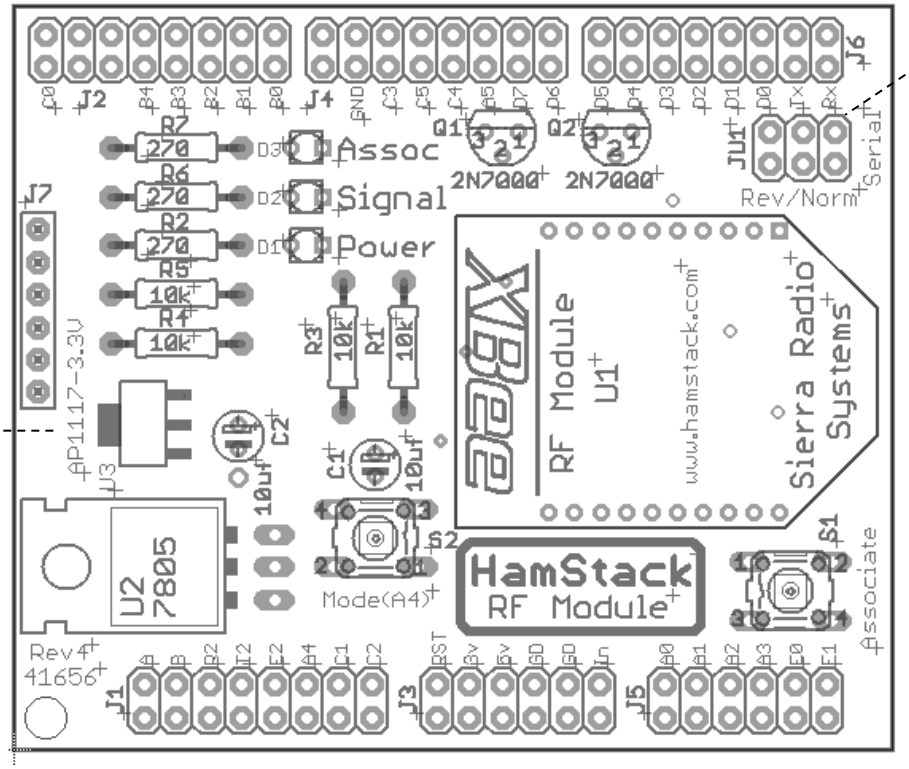
Parts Placement



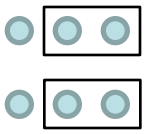
- ❑ C1, C2 10 uf electrolytic capacitor
- ❑ D1, D2, D3 LED
- ❑ J1 2x8 Stacking header or male header pointing down
- ❑ J2 2x8 Stacking header or male header pointing down
- ❑ J3 2x6 Stacking header or male header pointing down
- ❑ J4 2x8 Stacking header or male header pointing down
- ❑ J5 2x7 Stacking header or male header pointing down
- ❑ J6 2x8 Stacking header or male header pointing down
- ❑ J7 2x6 Stacking header
- ❑ JU1 2x3 pin header
- ❑ Q1, Q2 2N7000
- ❑ R1 10k
- ❑ R2 270
- ❑ R3 10k
- ❑ R4 10k
- ❑ R5 10k
- ❑ R6 270
- ❑ R7 270
- ❑ SOC1, SOC2 2mm 10 pin female SIP socket
- ❑ SW1 Push button
- ❑ SW2 Push button
- ❑ U1 RF module
- ❑ U2 7805
- ❑ U3 AP1117-3.3v 3.3v voltage regulator

xBee Mesh Network Backpack Board

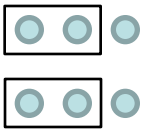
Connections



JU1 Rev/Norm serial jumpers
This jumper block selects the path of the serial data.



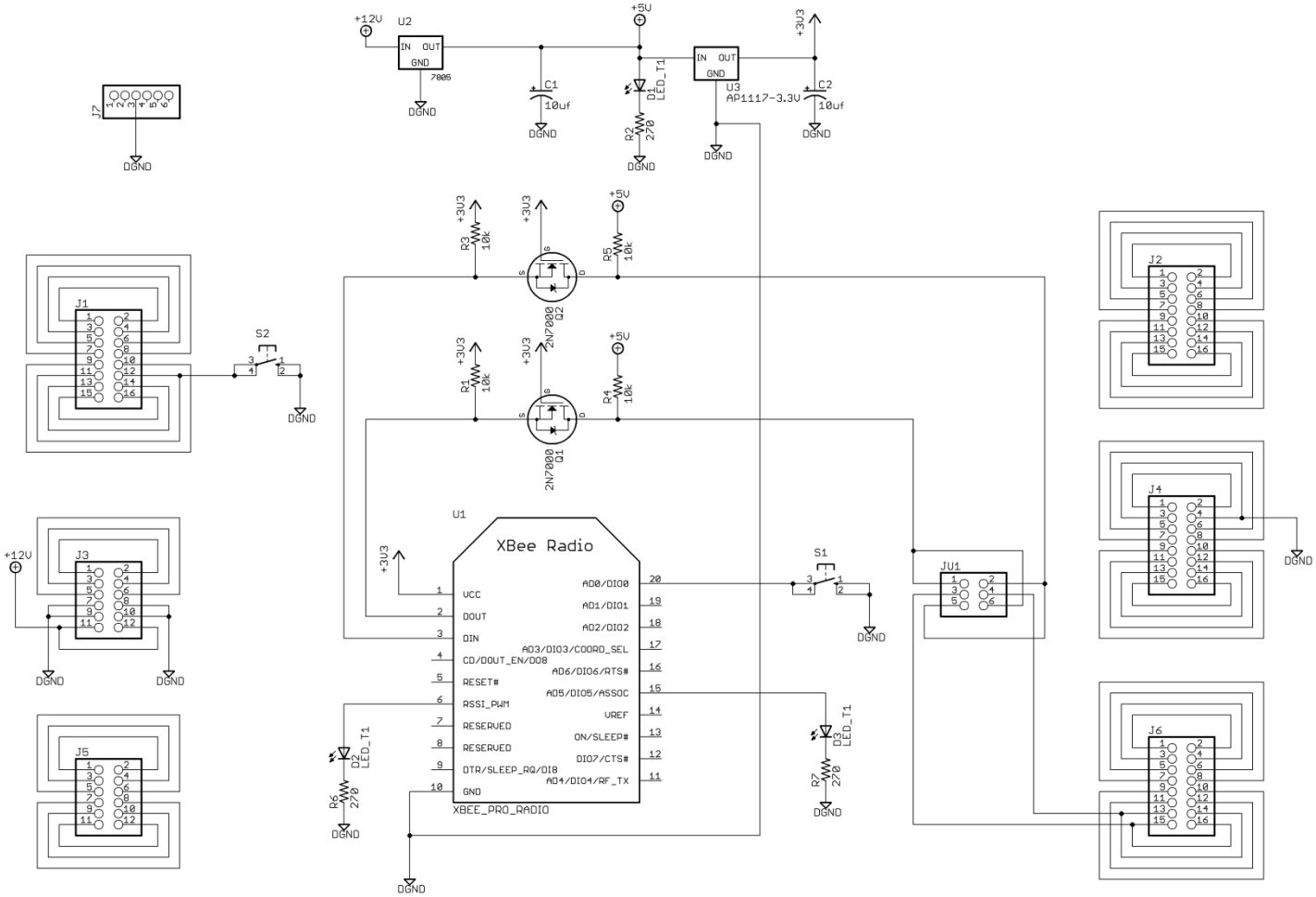
Normal position will connect the HamStack CPU UART pins to the RF module serial pins.

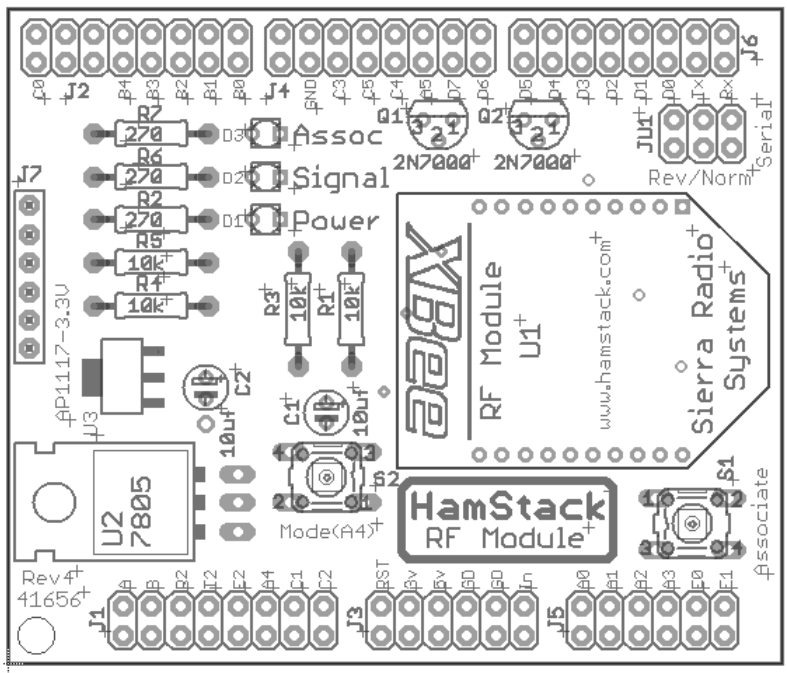


Reverse position will cross over the serial pins. This mode is used when cross connecting two communications boards with out a CPU board to form a communications bridge. For example between an ethernet board and the RF mesh network board.

xBee Mesh Network Backpack Board

Schematic

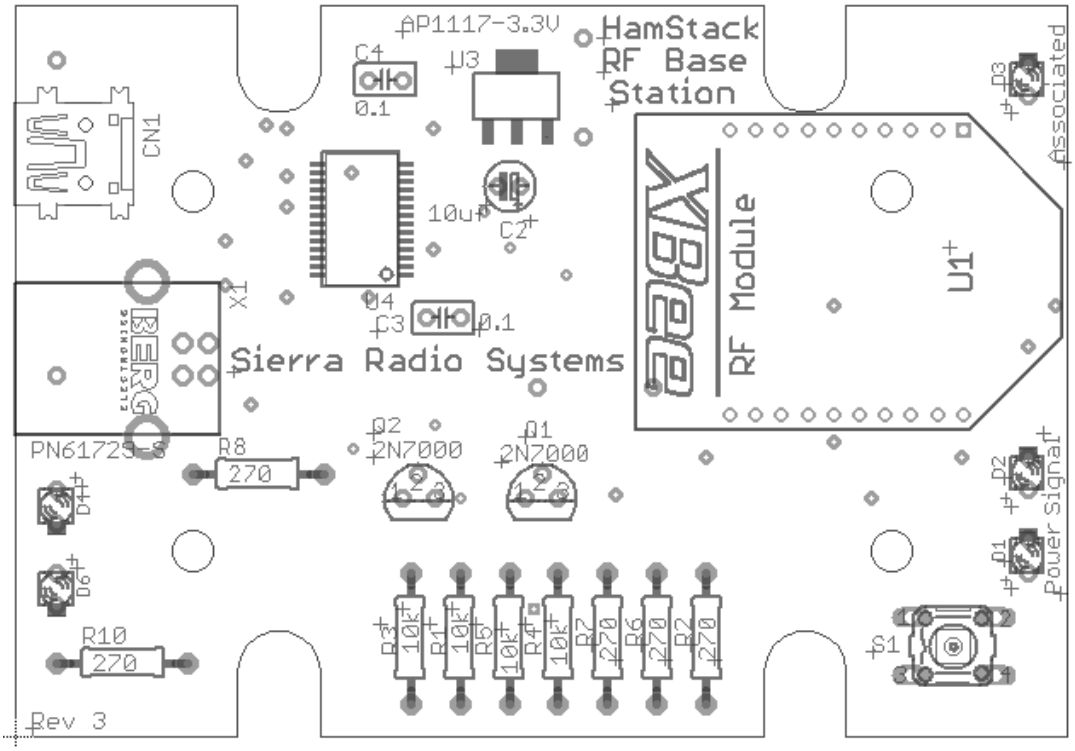




xBee USB Base Station Board

Parts Placement

USB Jack

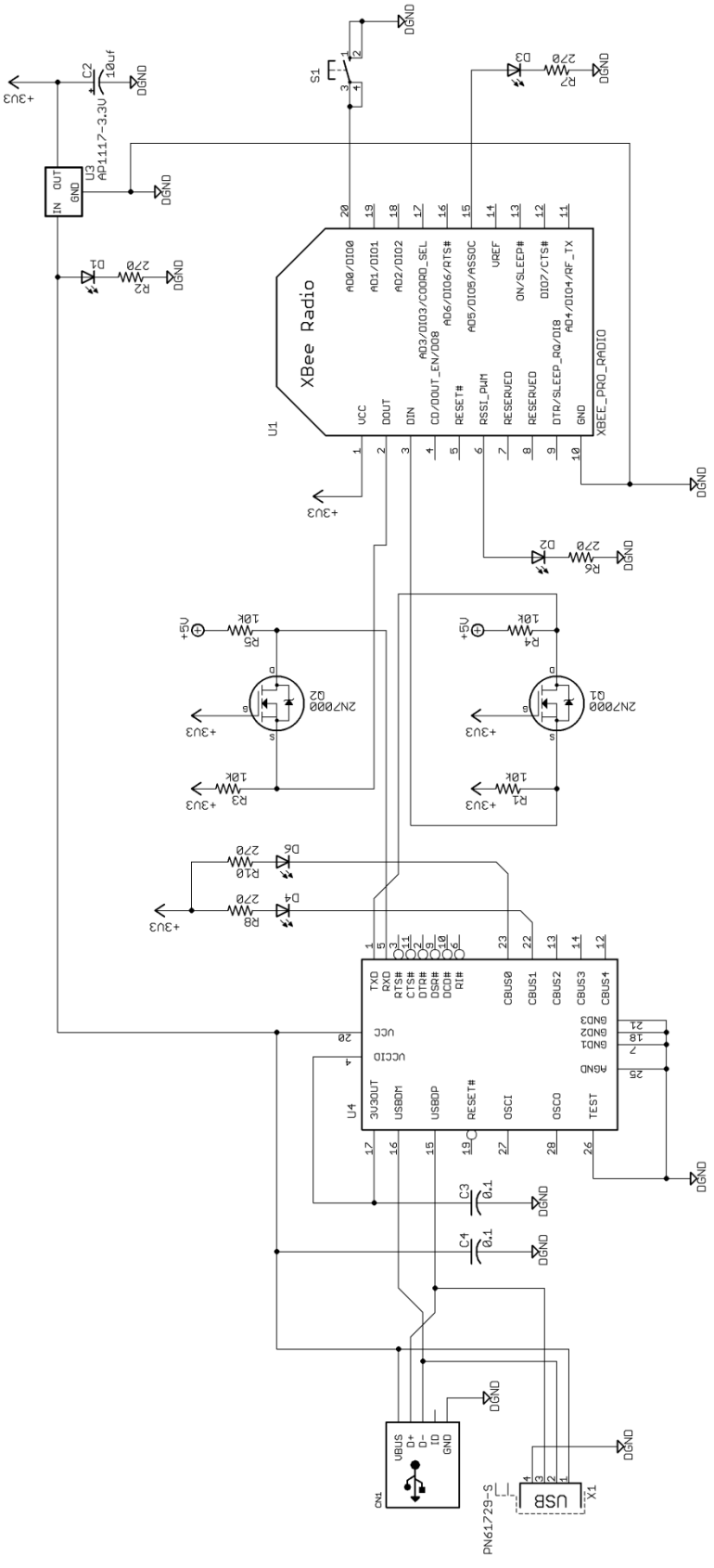


Associate Button

- ❑ C1, C2 10 uf electrolytic capacitor
- ❑ C3 0.1 uf capacitor
- ❑ D1, D2, D3, D4, D6 LED
- ❑ D5 None
- ❑ Q1, Q2 2N7000
- ❑ R1 10K
- ❑ R2 270
- ❑ R3 10K
- ❑ R4 10K
- ❑ R5 10K
- ❑ R6 270
- ❑ R7 270
- ❑ R8 270
- ❑ R9 None
- ❑ R10 270
- ❑ SOC1 2mm 10 pin female SIP socket under U1
- ❑ SOC2 2mm 10 pin female SIP socket under U1
- ❑ SW1 Push button
- ❑ U1 RF module
- ❑ U2 None
- ❑ U3 AP1117-3.3v 3.3v voltage regulator
- ❑ U4 FT232RL
- ❑ X1 USB Connector

xBee USB Base Station Board

Schematic



xBee Firmware Configuration Guide

Introduction

The xBee line of data radio modules from Digi-International are very flexible and can be configured in a variety of ways. These devices support two different network protocols – ZigBee and Digi-Mesh. Both protocol stacks are built on the industry standard IEEE 802.15.4 network layers. ZigBee and Digi-Mesh build functionality on top of that. Each protocol offers many configuration parameters to tailor the device's behavior to meet the needs of a wide range of applications. For a complete description of configuration options refer to the Digi web site for more details. It is far beyond the scope of this document to present all protocol, and configuration options. We have selected a specific configuration that works very well for the vast majority of applications. If you need different functionality, consult the Digi web site and on-line support groups. To install firmware and set configuration parameters in each radio module, you will use a program called X-CTU available free from Digi-International.

Firmware Stack

We recommend the Digi-Mesh protocol stack for our HamStack projects. Digi-Mesh has most of the advantages of the ZigBee mesh network protocol but is simpler to configure and deploy. The only disadvantage to Digi-Mesh is that many vendors support ZigBee while Digi-Mesh is unique to Digi-International products. One great feature of the Digi products is the ability to re-flash the firmware in each module with different protocols. If you start with Digi-Mesh and want to experiment with ZibBee, you just need to load the ZigBee firmware and you are ready to go.

❑ Step 1 – Install X-CTU PC software

The X-CTU software can be downloaded from the Digi-International web site.

❑ Step 2 - Install the Digi-Mesh firmware

For the 2.4 GHz xBee (1mw) module, install modem firmware type: **XB24-DM**
For the 2.4 GHz xBee PRO module, install modem firmware type: **XBP24-DM**
For the 900 MHz xBee PRO module, install modem firmware type: **XPB09-DM**
The field "Function Set" should be set to XBEE PRO DIGIMESH 2.4

There are only a few configuration parameters required to get your data radio module up and running on the network.

❑ Step 3 - Set network ID

This is the "name" of the RF network that all units will join.
This is an arbitrary 4 digit number. You can pick any number you want. You can run multiple independent networks on the same RF channel by setting some modules to one address and other modules in another network to another address and they will ignore each other.

HamStack default recommendation:

Set Networking ID – Modem VID to 7373

xBee Firmware Configuration Guide

❑ Step 4 – Set operating channel

This is one of the 12 available RF carrier channels that the network will operate on. Channels are assigned values from 0C to 17 (C, D, E, F, 10, 11, 12,13,14,15, 16, 17, 18) represented as a hex value. The RF carrier channels overlap with other devices in the 2.4 GHz band including WiFi and Bluetooth networks. Generally speaking you can pick any channel and it will work fine even with these other networks in operation. If you are in an RF dense environment and want to take all steps possible to avoid interference, pick a channel that does not overlap with your local WiFi networks. See the frequency table in the appendix. We recommend using channel C, this channel overlaps with WiFi channels 1, 2 and 3.

HamStack default recommendation:

Set Network CH – Operating Channel to C

❑ Step 5 – Set device address

Setting the high order destination address to 0 and the low order destination address to FFFF will put the radio in broadcast mode. All data packets received will be sent to the serial port. In a HamStack environment running the StationStack Network Control Protocol on the HamStack CPU, the device address decoding is done by the HamStack CPU. This makes the network operate as a simple mesh of all devices where any data going into one data radio's serial port will appear at the output of all data radios. The HamStack CPU will then process the payload of the packets.

HamStack default recommendation:

Set Addressing DL – Destination Address High to FFFF
Set Addressing DH – Destination Address High to 0

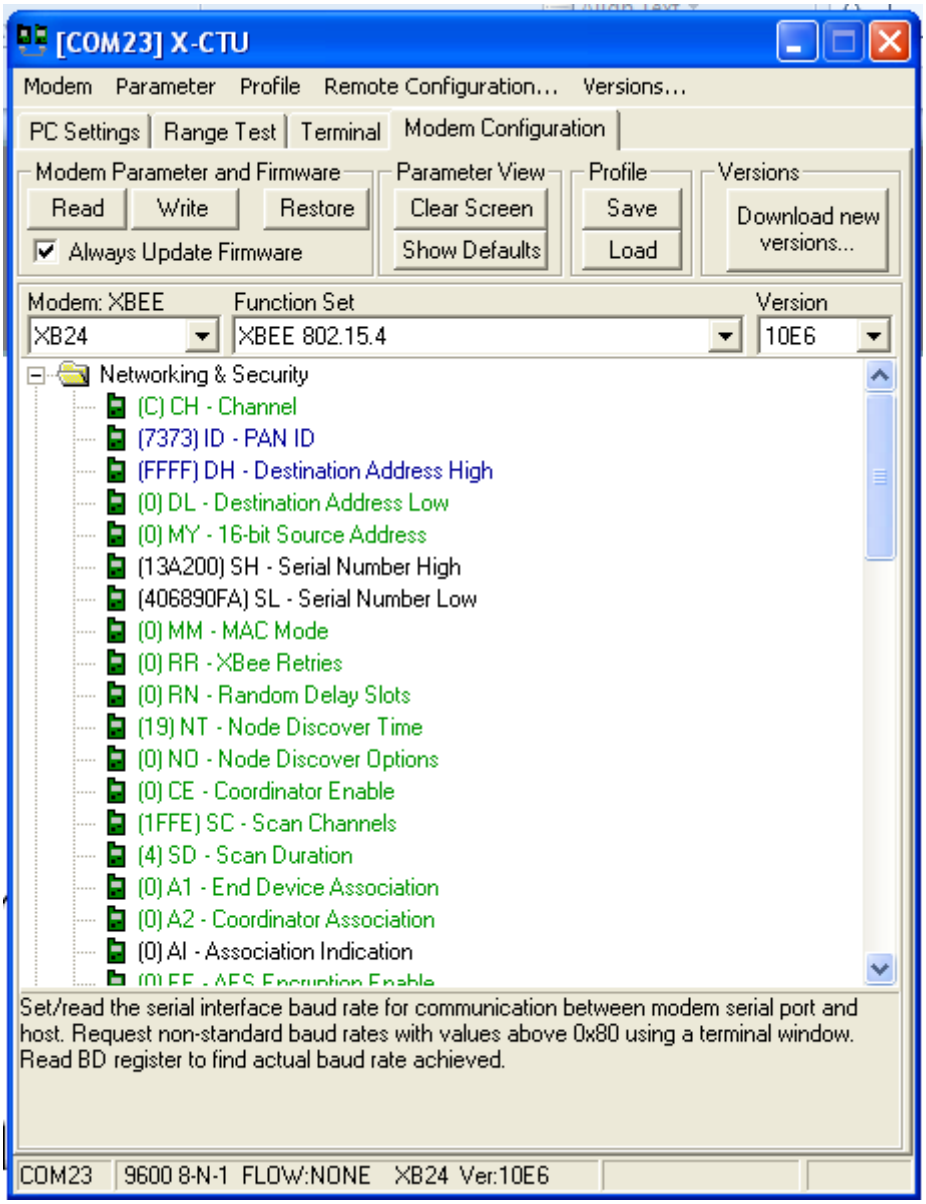
❑ Step 6 – Set serial port configuration

The data radio module's serial port can be configured to one of eight standard baud rates including 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, and 230400 baud.

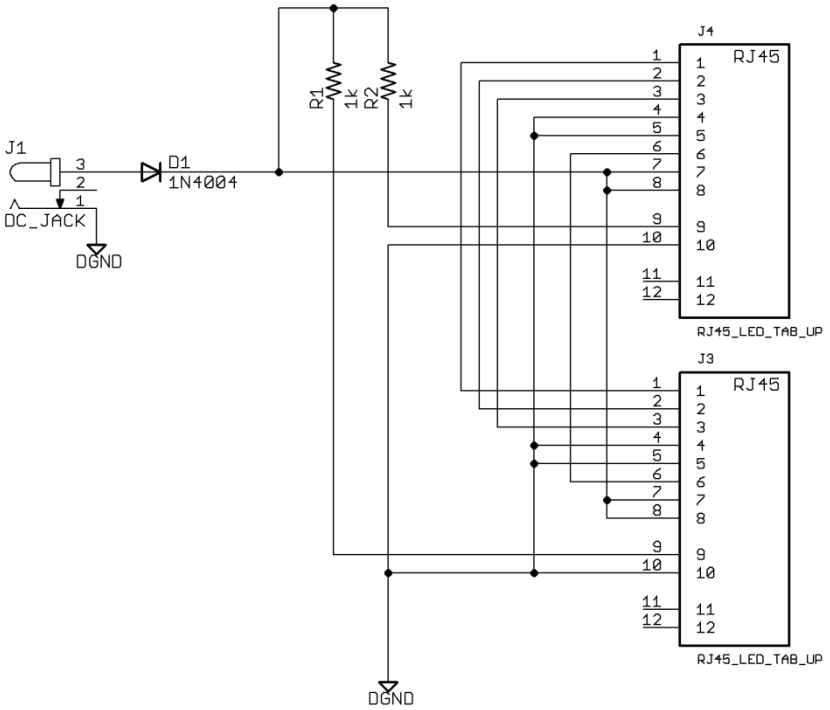
HamStack default recommendation:

Set Serial Interfacing BD – Baud Rate to 9600

Digi International X-CTU firmware configuration software

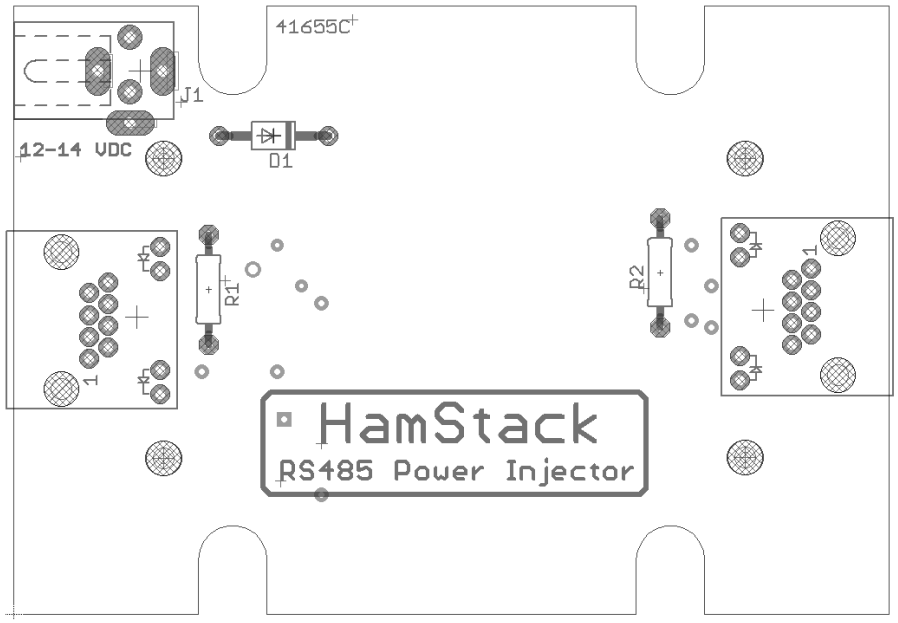


RS485 Power Injector



RJ45 Network Connector

Pin	Signal
1	RS485-A
2	RS485-B
3	Reserved
4	Ground
5	Ground
6	Reserved
7	+12 VDC
8	+12 VDC



D1	1N4004
J1	2.1mm DC coaxial connector, positive center pin
J2, J3	RJ45
R1, R2	1k resistor

StationStack Control Protocol

INTRODUCTION

The Sierra Radio Systems StationStack Control Prodocol provides a way to allow multiple devices and computers to communicate on a simple wired or wireless network. The primary application for the StationStack Control Protocol is to allow a master computer or device, to control and monitor a network of real-time control devices. The protocol defines three components, the physical connections and electrical signals, data link layer and the application layer. The purpose of the physical layer is to define the standard mechanical connector, pin assignments, and signaling voltages. The next layer up, the data link layer, defines the format of data packets sent on the network. The third and final layer defines the format of the payload being sent from point A to point B. The StationStack protocol application layer defines a set of commands that are used to control or monitor devices on the network. The application layer is carried as the payload in the data link layer. An implementation of the StationStack protocol may elect to follow only the network layer and use a set of commands unique to the application. Other implementations may use only legal commands defined in the StationStack protocol Command structure.

PHYSICAL CONNECTIONS AND ELECTRICAL SIGNALING

Physical Connections

The control protocol can be transmitted over any type of communications medium. The most common connections are wired through an RS232 or RS485 connection, a wireless connection, typically using an RF mesh data network or over ethernet.

Wired RS-485

The RS485 wired implementation uses commonly available Ethernet CAT5 cable and RJ-45 connectors. While the StationStack protocol has nothing at all to do with Ethernet except that we take advantage of the wide availability of premade cables. The CAT5 cable provides 8 wires which carry the network traffic and power. Many devices have two RJ-45 connectors wired in parallel. This allows for easy daisy-chaining of multiple devices using CAT5 cable. This makes it easy to add more devices to the network without the need for any kind of hub or switch.

RJ-45 Cable Assignments

1 – Network data signal A	5 - Ground
2 – Network data signal B	6 - Reserved
3 – Reserved	7 - +12 VDC
4 – Ground	8 - +12 VDC

Wired RS-485 Electrical Signaling

The electrical signaling used is based on RS-485. This signaling technique is a half-duplex, differential pair that allows multiple devices to be connected to a single pair of wires. RS-485 also has the advantage of allowing devices to be spread over 1000's of feet of cable without the need for signal conditioning or repeaters.

Power can be supplied by any device and delivered to all devices on the network. If a device can supply power to the network, there must be a way to disconnect the power, usually through a jumper block. Only 1 device is allowed to supply power to the network at a time. Network voltage should be between 10-14 VDC. This provides enough of headroom to power any StationStack protocol device.

RF Mesh Data Network

There are many short-range RF data network devices that can be used. The StationStack products are designed to use RF modules the Xbee data radios that support the IEEE 802.15.4 standard plus the DigiMesh firmware stack from Digi-International. These RF modules operate on 2.4 GHz and 900 MHz. They form an ad-hoc RF mesh data network. This means that as you add new nodes to the RF network, the automatically become part of the network. This is particularly convenient when extending the range between devices. Each node can be thought of as a serial port that taps into an invisible network of other devices. When data is sent into the serial port of the data radio, the packet will be delivered to every data radio in the network and the packet will be transmitted out of the rf module's serial port into the local HamStack CPU.

A network can consist of a mixture of wired RS-485 and RF data network enabled nodes.

DATA LINK LAYER - DATA RATES AND PACKET FORMAT

The StationStack protocol sends ASCII data at 9600 baud, non-inverted, 8 bits, no parity.

The StationStack protocol defines the format of packets of data transmitted on the network. The simple way to think of a packet is a string of ASCII characters that contains the payload to be transmitted from point A to point B and the additional characters necessary to provide synchronization, packet type identification, addressing, and error checking.

A typical packet looks like this...

/A0011222RESET<13>

Start of Packet	Packet Type	From Address	To Address	:	Payload	:	CRC Value	End of Packet

Start of Packet

A forward slash character / is reserved for the start of packet framing indication. When a slave device sees the slash, it knows there is a new packet.

Packet Type

The packet type character defines the format of the packet and instructions for how the packet is to be interpreted.

Packet types include /, A, R and 0 types.

Packet type A - Addressed packet. (Example: /A01:reset:39)

This is the standard network protocol format as described in the protocol section. Addressed packets include “from” and “to” addresses, error checking and the payload.

Address Assignments

0 System master. Typically a PC.

1-9 Devices 1-9

A-Z User devices

* Broadcast to all devices.

Any other characters are reserved and should not be used.

Packet type / - Direct packet. (Example: //reset)

This is a very simple format that is intended only for use in a system with a single node. The format for a direct packet is simply the payload.

For example: //reset

As you can see, there is no address or error checking. If multiple nodes are on the network and a direct command is issued, all nodes will decode and execute the command.

This can be very convenient to send master commands to all nodes but there is no error checking.

Packet type 0 (Example: /001:reset:34)

This is an Addressed packet that ignores the error checking field. This is used for manual entry of network addressed packets without the need to calculate the error checking value.

The error check field must, however, be included in the packet and can be filled with dummy data.

Error Check Value

The error check value is a 16 bit CRC. The CRC is applied to all characters in the packet except the initial start of packet character /. Of course the CRC is not applied to the CRC characters either.

Example: with the packet /A01:reset:39 the CRC is applied to A01:reset:

Payload

The payload is application dependent. See StationStack protocol command summary.

End of Packet

The end of packet character is a carriage return, ASCII byte value 013 (decimal). When an end of packet character is encountered, the input buffer is evaluated.

The evaluation process identifies a packet by finding the start of packet synchronizing character / and extracts the buffer contents up to the end of packet character.

The command parser then extracts the packet type, addresses, error check value and payload.

The error check value is calculated and compared to the packets error check value. If the values do not match, the buffer is flushed.

If the packet is good, then the to address is examined. If the to address is the same value as the devices address, the packet analysis will continue, if not, the packet is ignored and flushed from the buffer.

PAYLOAD FORMAT

The payload may contain any printable characters (0-9, A-Z , a-z, and punctuation except / and ,)

The payload may contain from zero to 9 fields delimited by commas. The comma delimiter must be placed between fields and not at the beginning of the payload.

The payload field assignment is typically a command field followed by zero to 8 argument fields.

For example:

RY,1,0

Where the command is “RY” and argument 1 is “1” and argument 2 is “0”. In this example the command tells the target device to set relay 1 to a value of 0 (or off).

COMMON COMMANDS

Every device may support a different set of commands. Refer to the device's reference manual for specific commands supported. These examples are presented to provide examples of typical commands.

Relay output on / off

Command RY,<relay_number>,<state>

Example //RY,1,0

Argument definition

<relay_number> is a value between 1 and n, where n is the number of relays on the target board.

<state> 1 = on, 0 = off

Digital output on / off

Command DO,<output_number>,<state>

Example //DO,1,0

Argument definition

<output_number> is a value between 1 and n, where n is the number of outputs on the target board.

<state> 1 = on, 0 = off

Read A/D converter input values

Command RAD

Example //RAD

Argument definition: none.

Reads the A/D converter input values and returns a packet in the format of

RAD, <value>,<value>,<value>,<value>

<value> = the A/D converters numeric value. For a 10 bit A/D converter input, the returned value will be a number between 0 and 1024.

Set digital input on / off

Command INnn <sp> ON|OFF <cr>

Example //IN01 <sp> ON<cr>

Definition Forces input nn on or off. The value may be temporary as the value may be over-written by the firmware itself.

Pulse output

Command PULSE <sp> n <cr>

Example //PULSE <sp> 1 <cr>

Definition Output n will be active for 250 ms then reset. This will pull the output pin to ground and close

the output relay for 250 ms. then reset to the normally off or open condition.