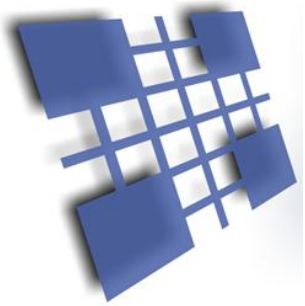
Sierra Radio Systems



DCN
Device Control Network

Version 1.0 Overview

Sierra Radio Systems
Device Control Network

INTRODUCTION

The Sierra Radio Systems Device Control Network (DCN) provides a way to allow multiple devices communicate on a wired or wireless network. The primary application for the DCN is to allow a master computer or device, to control and monitor a network of real-time control devices. The DCN specification defines four components, the physical and RF connections and electrical signals, data link layer, the application layer and network topology. The purpose of the physical layer is to define the standard mechanical connectors, pin assignments, signaling voltages, and RF frequencies . The next layer up, the data link layer, defines the format of data packets sent on the network. The third layer defines the format of the payload being sent from point A to point B and finally recommendations for the network topology.

SECTION 1 - PHYSICAL CONNECTIONS AND ELECTRICAL SIGNALING

The DCN is a “dual-band” system meaning that data may be transmitted over wired or RF communications paths or both.

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 DCN 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 12-14 VDC. This provides enough of headroom to power any DCN compatible device.

Wireless RF Data Network

The DCN can also use RF modules that operate on 2.4 GHz and 900 MHz. If mesh network radio modules are used, if 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 device's CPU.

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

SECTION 2 - DATA RATES AND PACKET FORMAT

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

The 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...

/A01:RY1,1:4D <13>

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

Start of Packet

A forward slash character / is reserved for the start of packet 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 direct, addressed and addressed with no error checking.

They start each packet with the characters //, /A, and /0 (zero) respectively.

Direct Packet type // (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 header // and the payload.

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.

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

These packets start with /A and contain the source and destination address, payload and error check data.

Addressed Packet, no error checking 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.

Device Addresses

You can assign any node an address using any printable character. However, for maximum functionality, we recommend using numbers as the addresses.

.

Pre-assigned default device addresses

0	System master. .
1-9	Devices 1-9
1	Station Controller
2	Remote RF coax relay
3	GPIO board
4	Not assigned
5	Not assigned
6	Not assigned
7	RadioRouter audio mixing and switching device
8	Not assigned
9	Not assigned
A-Z	Not assigned
*	Broadcast to all devices.

Any other characters are reserved and should not be used.

If you add a second device and the default device address is already in use, just pick another.

Payload

The payload is application dependent. See standard command summary.

Error Check Value

The error check value is an 8 bit LRC.. The LRC is applied to all characters in the packet except the initial start of packet character /. Of course the LRC is not applied to the LRC characters either.

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

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:

RY1,0

Where the command is “RY1” and argument 1 is “0”. In this example the command tells the target device to set relay 1 to a value of 0 (or off).

STANDARD 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.

Common System Commands

PING,2	Pings device #2
ROLLCALL	Every device will respond with its registration state (registered or not)
REBOOT	Restart the device
REGISTER	Tell the device the master knows it is there
RELEASE	Tell the device it is no longer registered to the master
SETADDR,5	Set device address to 5. Dip switch over rides this on reboot
ECHO,BLA	Send the string BLA back to the master
HELP	Display help information

System Controller specific commands

Relay commands

RY,10100	Set relays 1..5 to be on, off, on, off, off
RY3,1	Set relay 3 to be on (options: 1=on, 0=off, T=Toggle, P=Pulse)

Front panel commands

FP,BEEP,2	Generate 2 beeps
FP,CW,CQDX	Send CQDX in CW
FP,VOL,50	Set the tone generator to a volume of 50 in a range from 0 to 255
FP,LCD,2,5,HELLO	Tell the front panel processor to display the string "HELLO" on line 2 character position 5
FP,LCD,CLS	Clear LCD screen

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

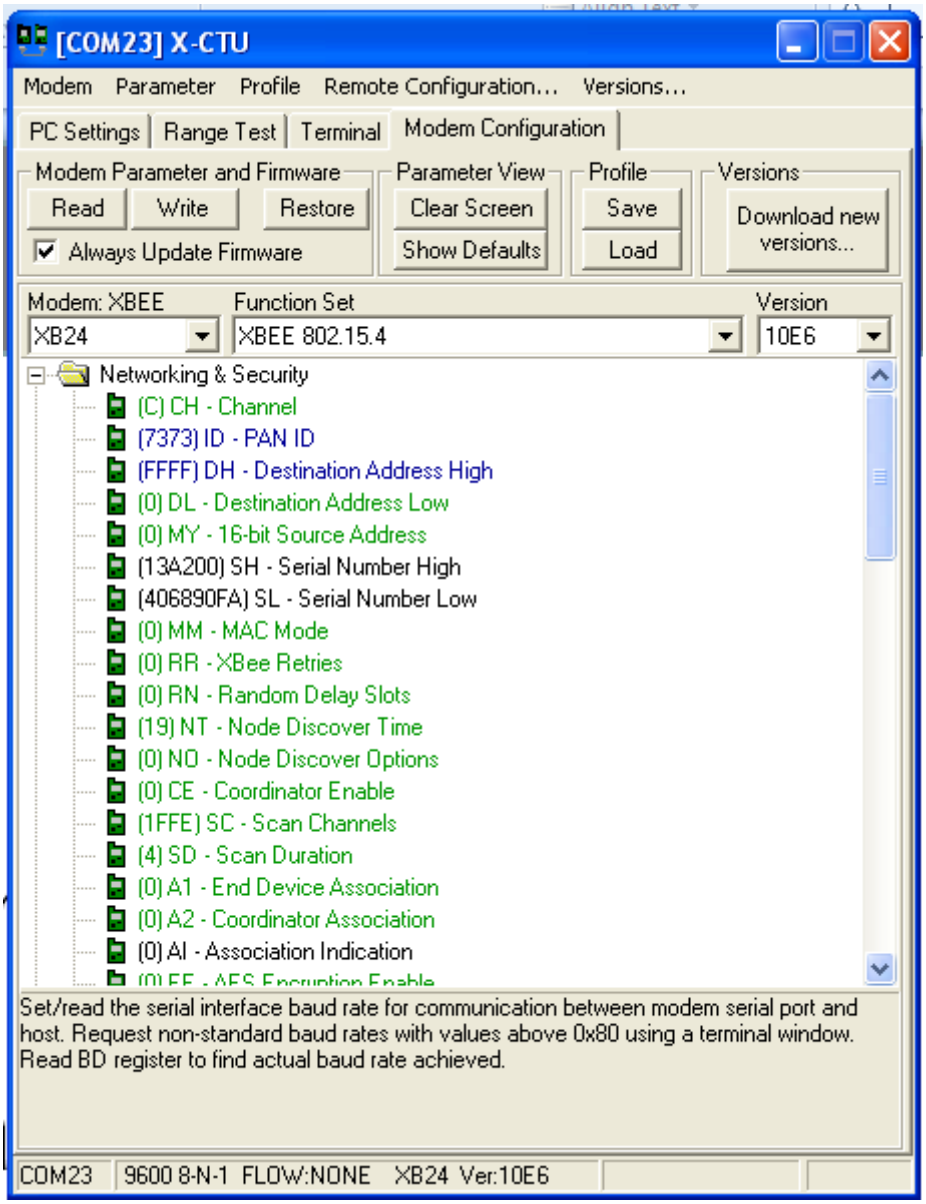
❑ Step 6 – Set serial port configuration

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

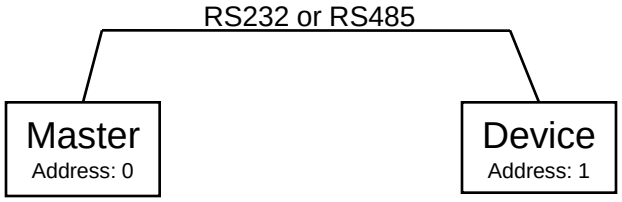


Network Topology

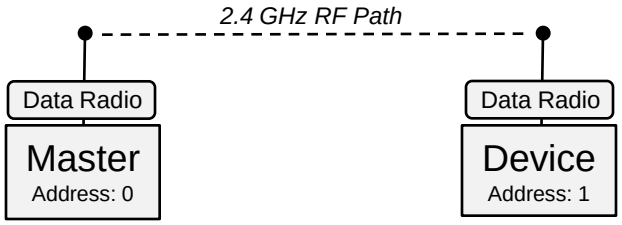
Overview

The Sierra Radio Device Control Network can be build with wired or wireless devices or a combination of both. The network protocol is completely independent from the physical communications channels. In the DCN, all devices are listening all the time to the network. One device, typically a computer, is the master and all other devices are slaves. All slaves remain quiet until the master communicates with them. Every device, including the master has unique address. Typically the master is address 0 (zero) and devices are numbered 1, 2, 3 and so on. Think of the network as one big “party line” where everyone is connected all the time. When anyone transmits, all devices can hear it. There are three commonly used physical connections used in a DCN. They are RS232 for point to point applications with one master and one slave device. RS485 which is a serial interface but very different from RS232. RS485 uses differential signaling on a pair of wires (called the A and B wires) and operate half duplex. The advantage to RS485 is the ability to hang multiple devices on the A/B pair and the long physical distances that can be used. The third common connection type is an RF data channel. The DCN data radios take serial data into their UART and package the data up in a packet and transmits them to the other data radios in the network where the same data comes out the UART on the other end and into the local devices microcontroller.

Lets look at some typical network configurations.



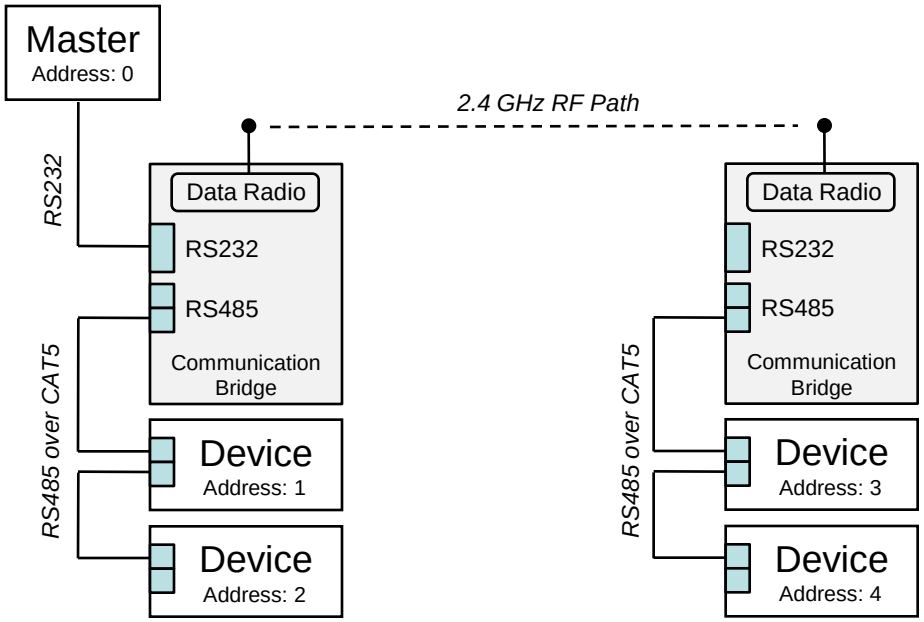
In the most simple example, the master can talk to a slave over a wired connection. Some Sierra Radio products have RS232 ports but all devices have the RS485 DCN connector which is a RJ45 (ethernet) modular connector.



When using a RF data radio module instead of a wired connection, the logical behavior is exactly the same. Data that comes out of the master’s UART will arrive at the input to the UART on the slave.

Network Topology

Wired and wireless connections can be combined. One example is where the local devices are connected with cable and the remote devices connected with a wireless data connection. For example…



In this example, all devices are listing at the same time to the DCN regardless of the medium of communications.

DigiMesh RF Data Module Reference

Channel*	SC	Frequency (MHz)
B	1	2.405
C	2	2.410
D	4	2.415
E	8	2.420
F	10	2.425
10	20	2.430
11	40	2.435
12	80	2.440
13	100	2.445
14	200	2.450
15	400	2.455
16	800	2.460
17	1000	2.465
18	2000	2.470
19	4000	2.475
1A	8000	2.480