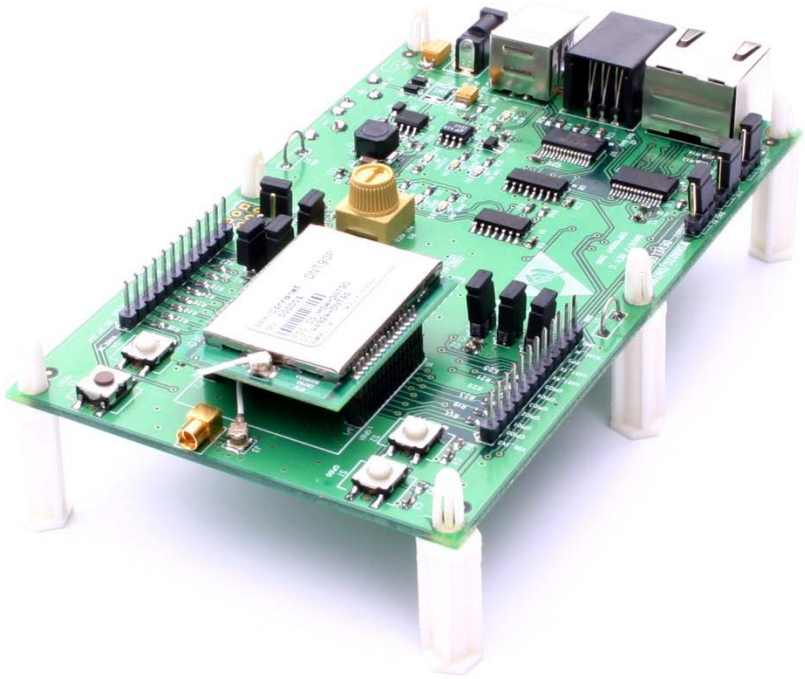
DNT90M Series 900 MHz Spread Spectrum Wireless Transceivers

Integration Guide



Important Regulatory Information

RFM Product FCC ID: HSW-DNT90
IC 4492A-DNT90

Note: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

1) Re-orientate or relocate the receiving antenna,
2) Increase the separation between the equipment and the radiator,
3) Connect the equipment into an outlet on a circuit different from that to which the receiver is connected,
4) Consult the dealer or an experienced radio/TV technician for help.

FCC Antenna Gain Restriction and MPE Statement:

The DNT90M has been designed to operate with any dipole antenna of up to 5.1 dBi of gain, or any Yagi of up to 6.1 dBi gain.

The antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.

Industry Canada Specific Statements:

The term "IC:" before the radio certification number only signifies that Industry Canada technical specifications were met.

*This Class **B** digital apparatus meets all requirements of the Canadian Interference Causing Equipment Regulations. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.*

Cet appareillage numérique de la classe B répond à toutes les exigences de l'interférence canadienne causant des règlements d'équipement. L'opération est sujette aux deux conditions suivantes: (1) ce dispositif peut ne pas causer l'interférence nocive, et (2) ce dispositif doit accepter n'importe quelle interférence reçue, y compris l'interférence qui peut causer l'opération peu désirée.

IC RSS-210 Detachable Antenna Gain Restriction:

This device has been designed to operate with the antennas listed below, and having a maximum gain of 6.1 dB. Antennas not included in this list or having a gain greater than 6.1 dB are strictly prohibited for use with this device. The required antenna impedance is 50 ohms:

Murata RWA092R Omnidirectional Dipole Antenna, 2 dBi
Murata OMNI095 Omnidirectional Dipole Antenna, 5 dBi
Murata YAGI099 Directional Antenna, 6.1 dBi

To reduce potential radio interference to other users, the antenna type and its gain should be so chosen that the equivalent isotropically radiated power (e.i.r.p.) is not more than that permitted for successful communication.

See Section 6.8 of this manual for regulatory notices and labeling requirements. Changes or modifications to a DNT90M not expressly approved by Murata may void the user's authority to operate the module.

Contents

1.0 DNT90M Introduction 7

1.1 Why Spread Spectrum? 7

1.2 Frequency Hopping versus Direct Sequence 8

2.0 DNT90M Network Overview 9

2.1 DNT90M Addressing 10

2.2 DN24M Transmission Error Control 10

2.3 Transparent and Protocol-formatted Serial Data 10

3.0 DNT90M Application Interfaces 11

3.1 Serial Ports 11

3.2 SPI Port 11

3.3 Digital I/O 14

3.4 Analog I/O 14

3.5 I/O Event Reporting and I/O Binding 15

4.0 DNT90M System Configuration 16

4.1 Configuration Parameters 16

4.2 DNT90M Common Network Configurations 16

4.3 DNT90M Default Network Configuration 17

4.4 Customized DNT90M Network Configurations 17

5.0 DNT90M Application Interface Configuration 18

5.1 Configuring the Serial Port 18

5.2 Configuring the SPI Port 18

5.3 Configuring Digital I/O 19

5.4 Configuring Analog I/O 19

5.5 Configuring I/O Event Reporting and I/O Binding 20

5.6 Configuring Sleep Mode 20

6.0 DNT90M Hardware 22

6.1 Specifications 23

6.2 Module Pin Out 24

6.3 Antenna Connector 25

6.4 Power Supply and Input Voltages 26

6.5 ESD and Transient Protection 26

6.6 Interfacing to 5 V Logic Systems 26
6.7 Mounting and Enclosures 26
6.8 Labeling and Notices 27
7.0 DNT90M Protocol-formatted Messages 28
7.1 Protocol Formats 28
7.2 Message Types 28
7.3 Message Format Details 29
7.4 Configuration Parameter Registers 36
7.4.1 Bank 0x00 - Transceiver Setup 36
7.4.2 Bank 0x01 - System Settings 37
7.4.3 Bank 0x02 - Status Parameters 38
7.4.4 Bank 0x03 - Serial and SPI Settings 39
7.4.5 Bank 0x04 - Host Protocol Settings 40
7.4.6 Bank 0x05 - I/O Parameters 41
7.4.7 Bank 0x06 - I/O Settings 42
7.4.8 Bank 0xFF - Special Functions 47
7.5 Protocol-formatted Message Examples 48
7.5.1 Data Message 48
7.5.2 Configuration Messages 49
7.5.3 Sensor Message 49
7.5.4 Event Message 50
8.0 DNT90MDK Developer's Kit 51
8.1 DNT90MDK Kit Contents 51
8.2 Additional Items Needed 51
8.3 Developer's Kit Default Operating Configuration 51
8.4 Developer's Kit Hardware Assembly 52
8.5 DNT90M Utility Program 53
8.6 Initial Kit Operation 54
8.6.1 Serial Communication and Radio Configuration 57
8.7 DNT90M Interface Board Features 63
9.0 Troubleshooting 65
9.1 Diagnostic Port Commands 65

10.0 Appendices 66
10.1 Ordering Information 66
10.2 Technical Support 66
10.3 DNT90M Mechanical Specifications 67
10.4 DNT90M Development Board Schematic 71
11.0 Warranty 74

1.0 DNT90M Introduction

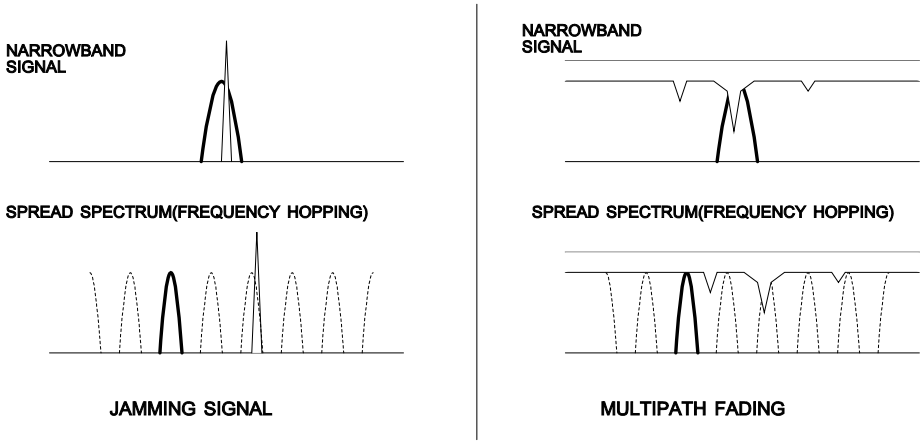
DNT90M transceivers provide low latency, highly reliable wireless connectivity for direct peer-to-peer network applications. Frequency hopping spread spectrum (FHSS) technology ensures maximum resistance to multipath fading and robustness in the presence of interfering signals, while operation in the 900 MHz ISM band allows license-free use in many regions of the world. The DNT90M supports serial data rates for host communications from 1.2 to 250.0 kbps, plus three SPI data rates from 125 to 500 kbps. On-board data buffering plus an error-correcting radio protocol provide smooth data flow and simplify the task of integration with existing applications. Key DNT90M features include:

- Multipath fading resistant frequency hopping technology with up to 26 frequency channels, 902.76 to 926.76 MHz
- Receiver protected by low-loss SAW filter, providing excellent receiver sensitivity and interference rejection important in outdoor applications
- Direct peer-to-peer radio communications provides very low message latency
- FCC 15.247 and IC RSS-210 certified for license-free operation
- Five mile plus range with omnidirectional antennas (antenna height dependent)
- Transparent ARQ protocol with data buffering ensures data integrity
- Analog and Digital I/O supports wireless sensing applications
- Simple interface handles both data and control at up to 250 kbps on the serial port or 500 kbps on the SPI port
- 64 selectable hopping patterns support multiple co-located systems
- Plug-in or solder reflow versions available with either U.FL RF connector or integral antenna
- AES encryption provides protection from eavesdropping
- Nonvolatile memory stores DNT90M configuration when powered off
- Selectable +16 dBm (40 mW) or +22 dBm (158 mW) transmit power levels
- Automatic I/O event reporting mode simplifies application development
- I/O binding mode provides wireless transmission of analog and digital values

1.1 Why Spread Spectrum?

A radio channel can be very hostile, corrupted by noise, path loss and interfering transmissions from other radios. Even in an interference-free environment, radio performance faces serious degradation from a phenomenon known as multipath fading. Multipath fading results when two or more reflected rays of the transmitted signal arrive at the receiving antenna with opposing phases, thereby partially or completely canceling the signal. This problem is particularly prevalent in indoor installations. In the frequency domain, a multipath fade can be described as a frequency-selective notch that shifts in location and intensity over time as reflections change due to motion of the radio or objects within its range. At any given time, multipath fades will typically occupy 1% - 2% of the band. From a probabilistic viewpoint, a conventional radio system faces a 1% - 2% chance of signal impairment at any given time due to multipath fading.

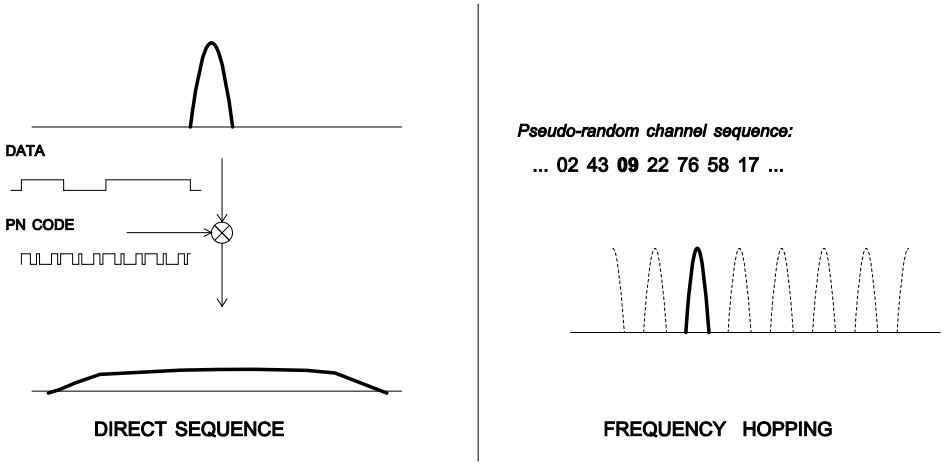
Spread spectrum reduces the vulnerability of a radio system to both multipath fading and jammers by distributing the transmitted signal over a larger region of the frequency band than would otherwise be necessary to send the information. This allows the signal to be reconstructed even though part of it may be lost or corrupted in transmission.



Narrow-band versus spread spectrum transmission
Figure 1.1.1

1.2 Frequency Hopping versus Direct Sequence

The two primary approaches to spread spectrum are direct sequence spread spectrum (DSSS) and frequency hopping spread spectrum (FHSS), either of which can generally be adapted to a given application. Direct sequence spread spectrum is produced by multiplying the transmitted data stream by a much faster, noise-like repeating pattern. The ratio by which this modulating pattern exceeds the bit rate of the base-band data is called the processing gain, and is equal to the amount of rejection the system affords against narrow-band interference from multipath and jammers. Transmitting the data signal as usual, but varying the carrier frequency rapidly according to a pseudo-random pattern over a broad range of channels produces a frequency hopping spectrum system.



Forms of spread spectrum - direct sequence and frequency hopping
Figure 1.1.2

One disadvantage of direct sequence systems is that due to design issues related to broadband transmitters and receivers, they generally employ only a minimal amount of spreading, often no more than the minimum required by the regulating agencies. For this reason, the ability of DSSS systems to overcome fading and in-band jammers is relatively weak. By contrast, FHSS systems are capable of hopping throughout the entire band, statistically reducing the chances that a transmission will be affected by fading or interference.

This means that a FHSS system will degrade gracefully as the band gets noisier, while a DSSS system may exhibit uneven coverage or work well until a certain point and then give out completely.

Because it offers greater immunity to interfering signals, FHSS is often the preferred choice for co-located systems. Since direct sequence signals are very wide, they can offer only a few non-overlapping channels, whereas multiple hoppers can interleave, minimizing interference. Frequency hopping systems do carry some disadvantages, in that they require an initial acquisition period during which the receiver must lock on to the moving carrier of the transmitter before any data can be sent. In summary, frequency hopping systems generally feature greater coverage and channel utilization than comparable direct sequence systems. Of course, other implementation factors such as size, cost, power consumption and ease of implementation must also be considered before a final radio design choice can be made.

2.0 DNT90M Network Overview

A DNT90M network is referred to as *direct peer-to-peer* network. A transmission from a DNT90M can be directed to any other peer in its network (unicast), or to all other peers in its network (broadcast). Unless a DNT90M is transmitting or is in sleep mode, it is constantly scanning all the channels in its frequency hopping sequence for a transmission from another peer. When a DNT90M has data to transmit, it transmits a beacon that allows the other peers in its network to rapidly synchronize with its phase in the hopping sequence. After sending the beacon, the transmitting radio immediately sends its data packet. If the data packet is addressed to a specific peer and is received without errors, the destination peer transmits an acknowledgement (ACK) to the originating peer, completing the error-free transmission. If the data packet is being broadcast to all other peers in the system, it is transmitted several times on different channels to mitigate the chances of a reception error due to poor propagation or interference on one channel. Most DN90M unicast packets are transmitted and acknowledged in less than 50 milliseconds. This very low transmission latency is achieved without compromising the robustness inherent in FHSS communications. Figure 2.0.1 depicts the communication paths available in a network consisting of four DNT90M peers.

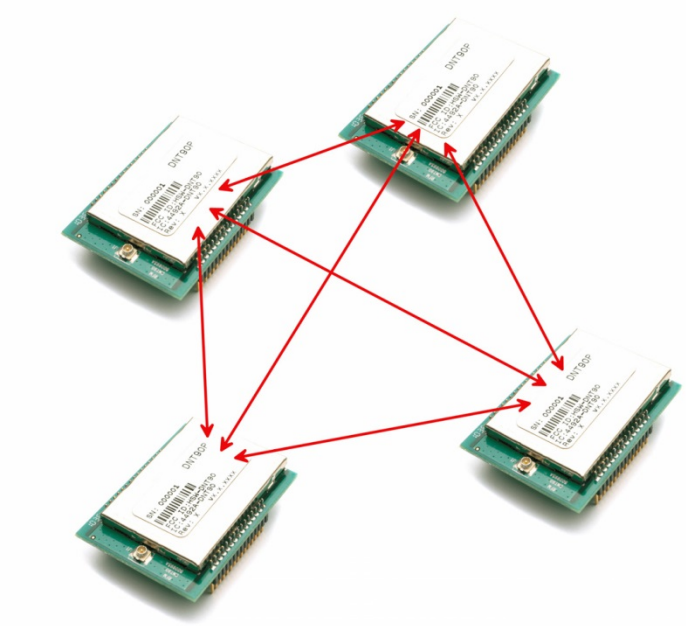


Figure 2.0.1

2.1 DNT90M Addressing

Each DNT90M has a unique three-byte *MAC address*, which is used to send unicast packets to it. A DNT90M can also send a packet to all other DNT90M's in its network by using the broadcast address 0xFFFFFF. The MAC address can be read or bar-code scanned from the label on top of each DNT90M radio, or retrieved through the radio's serial port or SPI interface. There are four sources of destination MAC addresses a DNT90M can use when sending a packet:

- the broadcast MAC address, 0xFFFFFF
- the MAC address in the transparent mode destination address *(RmtTransDestAddr)* parameter
- the MAC address included in a protocol-formatted message from the DNT90M's host
- the MAC address of the originator of the most recent packet received, when the *TransPtTo-PtMode* parameter is enabled.

Parameters related to MAC address selection are discussed in detail in Section 7.4 of this document.

All DNT90M radios hold a *system ID* parameter that can be used to distinguish between overlapping DNT90M networks. System IDs provide network filtering for broadcast packets. Also, a unique frequency hopping pattern is associated with each of the 64 available system IDs, minimizing the chances of transmission collisions between overlapping DNT90M networks.

2.2 DN24M Transmission Error Control

DNT90M packets include 24 error detection bits, allowing the integrity of a received packet to be tested with very high confidence. A received packet is discarded if an error is detected. Error-free unicast packets are acknowledged by an ACK message back to the originator from the destination radio. If the originator does not receive an ACK for a transmitted packet, it will resend it on the next channel in the frequency hop sequence, etc., up to a limit set by the *ArqAttemptLimit* parameter. The default value for this parameter is 10 attempts, which ensures message delivery in all but the most extreme conditions.

The number of times a broadcast packet is sent is controlled by the *BcstAttemptLimit* parameter. The radio moves to the next channel in the hopping sequence after each broadcast transmission to mitigate the chances of a reception error due to poor propagation or interference on one channel. The default value of the *BcstAttemptLimit* parameter is four.

To manage transmission collisions between multiple DNT90M radios trying to transmit at the same time, a progressive random delay is used to offset the time an unacknowledged packet is retransmitted as a radio progresses through the network's hopping sequence.

2.3 Transparent and Protocol-formatted Serial Data

A DNT90M can directly input and output data bytes and data strings on its serial port. This is referred to as *transparent* serial port operation. DNT90M radios also support *protocol-formatted messages*, which can also be used to carry data bytes and data strings, and must be used for:

- configuration commands and replies
- I/O event messages
- announcement messages including heartbeats

Protocol-formatted messages are discussed in detail in Section 7. Briefly, protocol-formatted messages include a start-of-messages character, message length and message type information, the destination MAC address for the message, and the message payload.

Transparent data is routed using the transparent mode destination address held in the *RmtTransDestAddr* parameter. The default value for this parameter is the broadcast MAC address, 0xFFFFFF. The *TransPtToPtMode* parameter is enabled by default, which causes a radio to unicast a packet to the peer from which it most recently received a message. These two parameter defaults make a network consisting of two DNT90M radios “plug-and-play”, as they automatically select each other’s MAC addresses for transmitting transparent data back and forth efficiently.

3.0 DNT90M Application Interfaces

A DNT90M module provides a variety of application interfaces including two serial ports, an SPI port, six digital I/O ports (logic state), three 12-bit ADC input ports, and two 12-bit DAC output ports. Each of these interfaces is discussed below.

3.1 Serial Ports

The DNT90M includes two serial ports, one for communication and an optional one for diagnostics. The communication port is a full-duplex UART interface with hardware flow control on two of the digital I/O pins an optional feature. One digital I/O pin can also be configured as an RS485 enable function. The serial communication port can be configured with baud rates from 1.2 to 250 kbps, with 9.6 kbps the default baud rate. The DNT90M communication port transmits/receives 8-bit data with a choice of even, odd or no parity and 1 or 2 stop bits. The default configuration is no parity and one stop bit. See Section 5.1 for recommendations on configuring the communication port, and Section 7.4.4 for detailed information on configuration parameters. The diagnostic port is enabled as an alternate function of the LINK and ACTIVITY pins, and can be configured with baud rates from 1.2 to 250 kbps, with 9.6 kbps the default baud rate. The diagnostic port transmits/receives 8-bit data with no parity and 1 stop bit. See Section 7.4.8 for diagnostic port configuration details.

3.2 SPI Port

The DNT90M serial peripheral interface (SPI) port can operate either as a master or a slave. The port includes the four standard SPI connections - MISO, MOSI, SCLK and /SS, plus three signals used to support SPI slave mode operation - /HOST_RTS, /HOST_CTS and DAV. The serial port and SPI master mode can run simultaneously. Serial port operation is disabled when the SPI port is configured for slave mode. Note that all SPI slave mode messages must be protocol formatted.

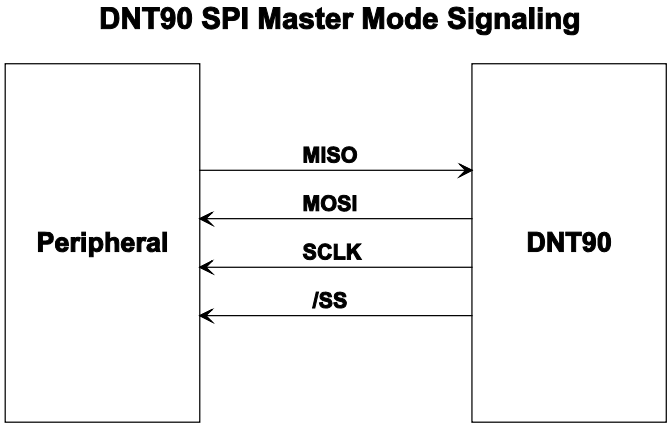


Figure 3.2.1

The DNT90M SPI port can run at three clock rates in master mode - 125, 250 or 500 kbps. There are two message sources available to a DNT90M SPI master, a protocol-formatted *RxData* message or a stored command. The DNT90M master will clock a message from either source into its slave and return the bytes clocked out as a protocol-formatted *TxData* message. The DNT90M event timer triggers sending the stored command to the DNT90M's slave. The stored command can be up to 16 bytes in length. Figure 3.2.1 shows the required SPI master mode-signal connections, and Figure 3.2.2 shows the SPI master-mode timing.

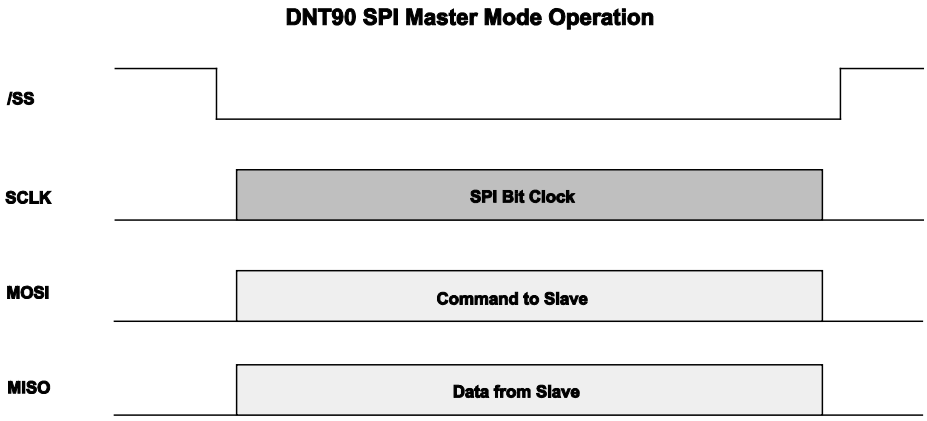


Figure 3.2.2

In SPI slave mode, the host can stream data into DNT90M at up to 250 kbps, provided the host suspends clocking within 10 bytes following a low-to-high transition on /HOST_CTS. The host can clock data into the DNT90M at up to 4 Mbps for data bursts of up to 50 bytes, provided the interval from the end of one burst to the start of the next burst is at least 2 ms, and the host suspends clocking on a low-to-high transition on /HOST_CTS. See Figure 3.2.4.

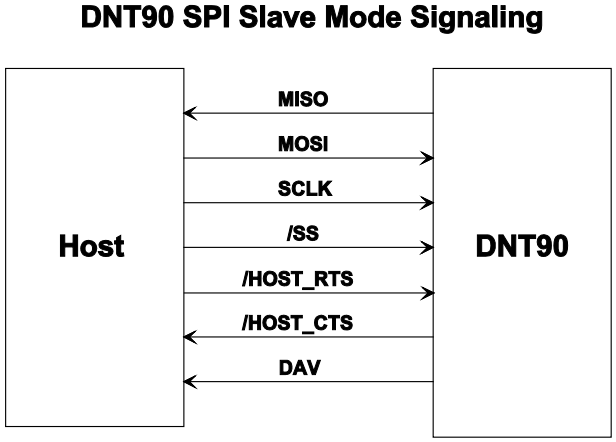


Figure 3.2.3

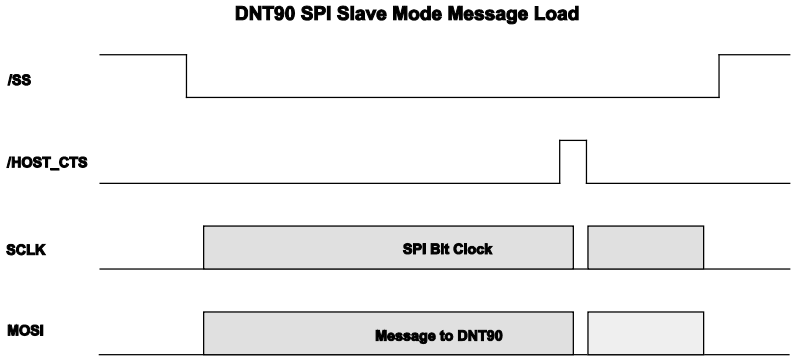


Figure 3.2.4

The host should use the following steps to fetch data from a DNT90M SPI slave, as show in Figure 3.2.5:

1. The host sets the /HOST_RTS signal high to allow the DNT90M to signal data available.
2. The DNT90M sets the data available (DAV) high to signal the host it has data.
3. The host set the /SS signal low to enable SPI operation.
4. The host clocks in one dummy byte (ignore the output byte) and then sets /HOST_RTS low.
5. The host begins to clock out the data, which can include several messages.
6. The host continues to clock out data until a 0x00 byte occurs in the byte stream where a 0xFB start-of-message would be expected.
7. The host has now clocked out all messages and the 0x00 is discarded.
8. The host sets /HOST_RTS and /SS high to allow the DNT90M to signal DAV the next time it has data.

Note that the DAV signal can go low before the last message is clocked out. It is not a reliable indication that the last byte of the message(s) has been clocked out. See Section 5.2 for recommendations on configuring the SPI port, and Section 7.4.4 for detailed information on SPI port configuration parameters.

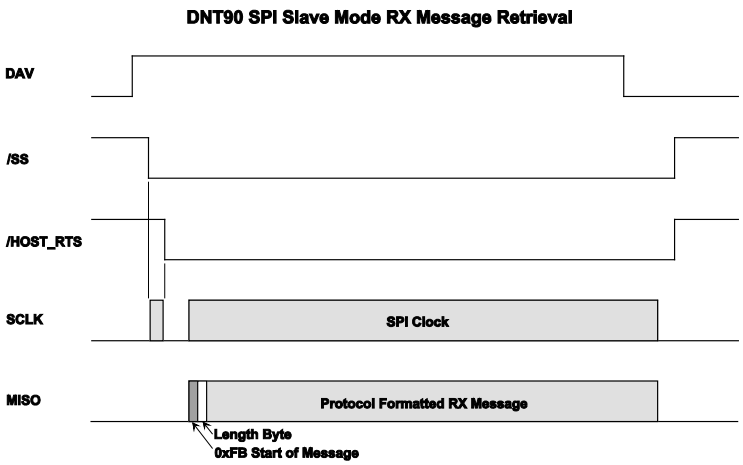


Figure 3.2.5

3.3 Digital I/O

The DNT90M's six digital (state) I/O ports are labeled GPIO0 through GPIO5. GPIO5 has an alternate function of /HOST_RTS and GPIO4 of /HOST_CTS, providing hardware handshaking for the serial port and SPI slave mode operation. If serial port hardware handshaking is not required and SPI slave mode is not enabled, GPIO4 and GPIO5 can be used for other digital I/O functions. When SPI slave mode is enabled, GPIO5 and GPIO4 *must* be used for /HOST_RTS and /HOST_CTS respectively, and GPIO3 *must* be used to provide the DAV signal (SPI slave mode overrides any other configuration for these ports). Except in SPI slave mode, GPIO0 through GPIO5 are available for customer-defined functions:

- The direction of each GPIO pin can be set for both active and sleep modes.
- The initial state (power on) of all GPIO pins configured as outputs can be set.
- The state of all GPIO pins configured as outputs in sleep mode can be set.
- GPIO triggering of I/O event reporting can be configured.
- GPIO level control of sleep hold-off can be configured.

See Section 5.3 for recommendations on configuring the digital I/O, and Sections 7.4.6 and 7.4.7 for detailed information on GPIO parameters.

3.4 Analog I/O

The DNT90M's three ADC input channels are labeled ADC0 through ADC2. The ADC can be disabled if unused to reduce current consumption. The ADC can be operated in either single-ended mode or differential mode. In single-ended mode, up to three sensor inputs can be measured. The negative sensor inputs are connected to ground and the positive sensor inputs are connected to ADC0, ADC1 and ADC2 respectively. Single-ended measurements are unsigned 11-bit values. In differential mode, one or two sensor inputs can be measured as 12-bit signed values. The first differential measurement is the difference between the voltage on ADC1 and the voltage on ADC0, and is referred to as the ADC0 differential measurement. The second differential measurement is the difference between ADC2 and ADC0, and is referred to as the ADC1 differential measurement. Operating the ADC in differential mode takes advantage of common mode rejection to provide the best measurement stability. Differential mode also incorporates a programmable gain preamplifier function, with gains settings from 1 to 64 available.

There are two options for the ADC full-scale reference:

1. The DNT90M regulated supply voltage divided by 1.6, or about 2.06 V
2. A low impedance voltage source applied to the DNT90M's ADC_EXT_REF input pin, 2.7 V maximum. If no connection is made to this pin, a voltage equal to about 2.7 V will be present.

Note that when differential ADC mode is used, the maximum output voltage available from the preamplifier at any gain setting is 2.4 V, so the maximum ADC reading that can be made using a 2.7 V ADC reference will be about 88.9% of full scale. The ADC channels are read each ADC sample interval, which is configurable. High and low measurement thresholds can be set for each ADC channel to trigger I/O event reporting messages.

The DNT90M's two DAC outputs are labeled DAC0 and DAC1. The DACs can be disabled if unused to reduce current consumption. The DAC settings have 12-bit resolution. There are two options for the DAC full-scale reference:

1. The DNT90M regulated supply voltage, about 3.3 V
2. A low impedance voltage source applied to the DNT90M's ADC_EXT_REF input pin, 2.7 V maximum. If no connection is made to this pin, a voltage equal to about 2.7 V will be present.

See Section 5.4 for recommendations on configuring the analog I/O, and Sections 7.4.6 and 7.4.7 for detailed information on analog I/O parameters.

3.5 I/O Event Reporting and I/O Binding

The DNT90M's I/O event reporting function can generate a protocol-formatted *RxEvent* message when triggered by one of the following I/O events:

- A specific state change of GPIO0, GPIO1, GPIO2 or GPIO3.
- Firing of the periodic event report timer.
- A high or low threshold exceeded on a measurement by ADC0, ADC1 or ADC2.

An I/O report message includes:

- The states of GPIO0 through GPIO5.
- The latest measurements made by ADC0 through ADC2.
- A set of flags indicating which event(s) triggered the I/O report.
- The settings of DAC0 and DAC1.

The I/O binding function works in conjunction with I/O event reporting. When I/O binding is enabled on a DNT90M, data received in an I/O event report it is mapped as follows:

- GPIO2 will output the state of GPIO0 in the last received event report.
- GPIO3 will output the state of GPIO1 in the last received event report.
- DAC0 will output the voltage read by ADC0 in the last received event report.
- DAC1 will output the voltage read by ADC1 in the last received event report.

I/O binding is used to transmit switch positions or analog signals from one location to another. Note that I/O binding cannot be used in a DNT90M when SPI slave mode is enabled or differential ADC mode is used. See Section 5.4 for recommendations on configuring I/O event reporting and binding, and Sections 7.4.6 and 7.4.7 for detailed information on I/O reporting and binding parameters.

4.0 DNT90M System Configuration

DNT90M radios feature an extensive set of configuration options that allows them to be adapted to a wide range of applications. Configuration defaults have been carefully selected to minimize the configuration effort for most applications, while providing the ability to individually adjust the configuration of each radio to achieve highly optimized system operation.

4.1 Configuration Parameters

The configuration of a DNT90M is controlled by a set of *parameters* (registers). Parameters that address a particular aspect of operation are grouped into a *bank*. All parameters can be accessed through a module's serial port and over the radio link. Most parameters are read/write. Read-only parameters include fixed values such a MAC addresses, firmware version numbers and parameters that are dynamically adjusted during system operation. Write-only parameters include security keys and certain action triggers such as reset. Incorrectly configuring certain parameters can disable a module's radio link, but the configuration can always be corrected through the serial port. The organization of the parameter register banks and the details of each parameter are covered in Section 7.4 of this guide. Sections 4.3 through 5.6 discuss which parameters apply to various aspects of configuring a DNT90M network or application interface.

4.2 DNT90M Common Network Configurations

DNT90M's direct peer-to-peer operation is readily adapted to two other common network configurations: point-to-point networks and point-to-multipoint networks.

A point-to-point network is shown in Figure 4.2.1. Point-to-point systems are often used to replace wired serial connections. Point-to-point networks are also used to transmit switch positions and/or analog signals from one location to another.

Figure 4.2.1

Figure 4.2.2 shows the topology of a point-to-multipoint (star) network, where one DNT90M acts as a "base" unit connected to the computer running the network application, with the rest of the radios in the network communicating primarily with the base. Point-to-multipoint networks are typically used for data, sensor and alarm systems.

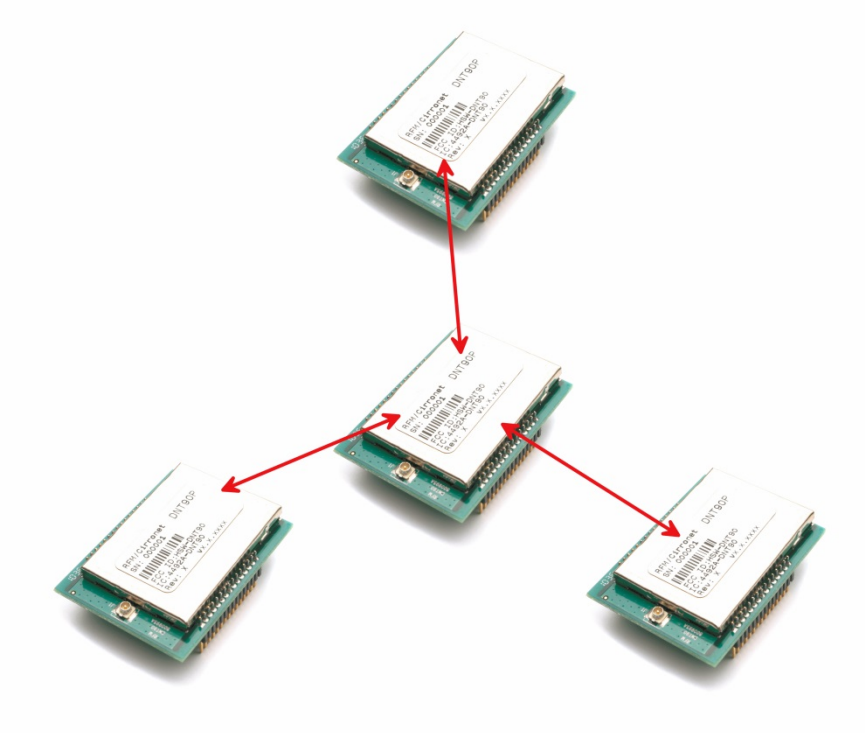


Figure 4.2.2

4.3 DNT90M Default Network Configuration

The default parameter values in a DNT90M provide “plug-and-play” operation for a point-to-point network. By default, DM90M radios are configured for transparent serial data communication at 9.6 kbps, 8 bit data, one stop bit, no parity. Data is routed using the transparent mode destination address held in the *RmtTransDestAddr* parameter. The default value for this parameter is the broadcast MAC address, 0xFFFFFF. However, the *TransPtToPtMode* parameter is enabled by default, which causes a radio to unicast packets to the peer from which it most recently received a message. These two parameter defaults make a network consisting of two DNT90M radios “plug-and-play”, as they automatically select each other’s MAC addresses for transmitting transparent data back and forth efficiently.

4.4 Customized DNT90M Network Configurations

The DNT90M includes many configuration parameters that allow extensive network customization. Most applications will require only a few of these parameters be changed from their default values. But for those applications that need them, Murata recommends the following configuration sequence. Skip the configuration steps where the default parameter value is satisfactory.

1. For any network configuration other than a point-to-point system, set the *TransPtToPtMode* parameter to 0x00 in each radio to disable the “reply to last received” feature.
2. For any DNT90M network communicating sensitive data, set the AES security key in all network radios by loading your selected 16-byte string into the *SecurityKey* parameter in Bank 0 (the default is 16 bytes of 0x00).
3. Configure the system ID in all radios by setting the *SystemID* parameter in Bank 0 (the default is OK if there is no chance of overlapping systems).
4. Set the transmitter power level as needed in all radios by setting the *TxPower* parameter in Bank 0 (the default is 158 mW).

5. For a peer-to-peer network (more than 2 radios), set the *ProtocolMode* parameter in Bank 4 of each radio to 0x01. Each radio's host application will require a database containing the MAC addresses of the other radios in the network, as discussed in Section 2.1 above.

6. For a point-to-multipoint network, set the *ProtocolMode* parameter in Bank 4 of the DNT90M acting as the network *base* to 0x01. The base radio will use protocol-formatted messages containing MAC addresses to communicate with the other radios in the network. Radios other than the base can set the *RmtTransDestAddr* parameter to the MAC address of the base unit and use transparent mode for sending serial data provided they only communicate with the base. If radios other than the base are sending I/O data, they must be in protocol mode and include the base MAC address in their protocol formatted messages. If SPI slave mode will be used, protocol mode must be enabled in all network radios.

7. If using transparent serial mode in the network:

    a. Set the timeout for transmission of transparent data as needed. The parameter that controls the timeout is the *TxTimeout* in Bank 4 (the default is to send as soon as possible).

    b. Set the minimum message length for transmission of transparent data in the remotes as needed. The parameter that controls the length is the *MinPacketLength* in Bank 4 (the default is one byte).

8. Load an optional "friendly description" in each system radio in the *UserTag* parameter, Bank 0.

5.0 DNT90M Application Interface Configuration

DNT90M modules include a comprehensive set of application interfaces and related options that support a wide range of applications including wireless RS232/485 cable replacements, wireless sensor networks, wireless alarm systems and industrial remote control applications. Recommended configuration steps for each application interface are discussed in Sections 5.1 through 5.6 below.

5.1 Configuring the Serial Port

The default serial port configuration is 9.6 kbps, 8-bit data, no parity and 1 stop bit.

1. Configure the serial data rate as required from 1.2 to 250 kbps by setting the *SerialRate* parameter in Bank 3.

2. Configure the parity and number of stop bits by setting the *SerialParams* parameter in Bank 3.

3. Enable/disable serial port hardware flow control as required by setting the *GpioAlt* parameter in Bank 6. Hardware flow control is disabled by default, but is recommended when operating at higher baud rates and/or sending large blocks of data.

5.2 Configuring the SPI Port

1. Enable either SPI *master mode* or SPI *slave mode* by setting the *SpiMode* parameter in Bank 3. The serial port remains operational in SPI master mode but is disabled in SPI slave mode.

2. If using SPI master mode:

    a. Select the SPI clock rate by setting the *SpiRateSel* parameter in Bank 3 (default is 125 kbps)

b. Set the SPI master command string and string length by setting the *SpiMasterCmdStr* and *SpiMasterCmdLen* parameters respectively in Bank 3.

3. Configure the edge trigger direction, bit-sampling edge and bit-order options by setting the *SpiOptions* parameter in Bank 3.

5.3 Configuring Digital I/O

1. GPIO2 through GPIO 5 have configurable alternate functions as discussed in Section 7.4.7. Select either digital (state) functionality or alternate functionality for each of these pins by setting the *GpioAlt* parameter in Bank 6. Note that selecting SPI slave mode overrides the *GpioAlt* parameter setting for GPIO3 though GPIO5.

2. Configure the direction of each GPIO pin as needed by setting the *GpioDir* parameter in Bank 6 (the default is all inputs).

3. Configure the direction of each GPIO pin for sleep mode as needed by setting the *GpioSleepDir* parameter in Bank 6 (the default is all inputs).

4. Set the initial state (power on) of all GPIO pins configured as outputs by setting the *GpioInit* parameter in Bank 6 (the default is all logic low).

5. Set the state of all GPIO pins configured as outputs in sleep mode by setting the *GpioSleepState* parameter in Bank 6 (the default is all logic low).

6. GPIO0 through GPIO3 can trigger I/O event reporting when functioning as digital inputs. Enable event report triggering and optional sleep hold-off for these pins by setting the *GpioEdgeTrigger* parameter in Bank 6.

5.4 Configuring Analog I/O

1. Select the ADC full-scale reference by setting the *AdcReference* parameter in Bank 6. This setting applies to all ADC channels. The default is the ADC_EXT_REF input. If ADC operation is not needed, setting this parameter to 0x03 disables ADC operation, reducing current consumption.

2. Select the ADC mode, either single-ended or differential by setting the *AdcDiffMode* parameter in Bank 6. The default is single-ended ADC operation.

3. If differential ADC mode is selected, set the desired ADC preamplifier gain for each ADC channel with the *AdcGainCh0* and *AdcGainCh1* parameters in Bank 6. The default gain is 1. Note that the full scale output voltage from the preamplifier is 2.4 V.

4. Reconfigure the ADC measurement interval as needed by setting the *AdcSampleIntvl* parameter. The default is 100 ms, and applies to all ADC channels.

5. Set the *AdcAveSelect* parameter to the number of ADC readings to be averaged to produce a measurement. The larger the *AdcAveSelect* parameter is set, the greater the noise filtering effect, but the longer it takes to produce a measurement. Setting this parameter to 8 or more when the ADC is operating in single-ended mode is especially helpful in stabilizing ADC measurements.

6. Measurements on each ADC input can be compared to high/low threshold values, triggering an I/O event report if the measurements go above/below the respective thresholds. The thresholds for each ADC channel are set by loading the *AdcXThresholdLo* and *AdcXThresholdHi*, where *X* refers to the ADC channel designator, 0 through 2.

When the ADC is operating in differential mode, the ADC1 to ADC0 differential measurement is compared to the "0" high and low thresholds, and the ADC2 to ADC0 differential measurements is compared to the "1" high and low thresholds. In this case the "2" threshold values are not used.

7. Set the *IoPreDelay* parameter as needed in Bank 6 to allow signals to stabilize following a module wakeup event.

8. Set the *AdcSkipCount* parameter in Bank 6 as needed to allow *internal* transients in the ADC sample-and-hold circuit to settle out. This parameter must be set to at least 3 when *AdcDiffMode* is selected. Note that the *Io-PreDelay* parameter discussed above provides a delay to allow signals *external* to the DNT90M to settle following an event, while *AdcSkipCount* skips measurements that may be distorted because the *internal* voltage on the ADC sample-and-hold has not settled.

9. Select the DAC full scale reference by setting *DacReference* in Bank 6. This setting applies to both DAC channels. The default is the ADC_EXT_REF input. If DAC operation is not needed, setting this parameter to 0x03 will disable DAC operation, reducing current consumption.

10. Configure the initial (power on) output level for DAC0 and DAC1 by loading the initial settings in the *Dac0Init* and *Dac1Init* parameters respectively.

The ADC and DAC channels are factory calibrated. It may be desirable to fine tune these calibrations after the DNT90M has been integrated with the customer's hardware in some applications. For analog calibration support, contact Murata technical support.

5.5 Configuring I/O Event Reporting and I/O Binding

1. Select the analog, digital and timing events that will trigger an I/O event report by setting the respective bits in the *IoReportTrigger* parameter in Bank 6. The default is no triggers set.

2. Configure the trigger behavior bits in the *GpioEdgeTrigger* parameter, Bank 6, for each GPIO input selected to generate an I/O event report.

3. For each ADC channel selected to generate an I/O event, set the high and low measurement threshold values. The *AdcThreshold* parameters are in Bank 6. When the ADC is operating in differential mode, the ADC1 to ADC0 differential measurement is compared to the "0" high and low thresholds, and the ADC2 to ADC0 differential measurements is compared to the "1" high and low thresholds. In this case the "2" threshold values are not used.

4. If the periodic timer has been selected to generate an event report, load the required timer report interval into the *IoReportInterval* parameter in Bank 6. The default timer interval is 30 seconds.

5. *Set the MaxQueuedEvents* parameter in Bank 6 as needed to limit the number of Event Reports that can be queued at one time by a DNT90M.

6. If I/O binding operation is desired, set the *IoBindingEnable* parameter in Bank 6 to 0x01. I/O binding is disabled by default, and cannot be used when the ADC is operating in differential mode.

5.6 Configuring Sleep Mode

Sleep mode can be used in conjunction with I/O reporting to greatly extend battery life on DNT90M remotes. At least one I/O report trigger must be enabled to allow sleep mode to be used.

1. Enable sleep mode as desired in each remote by setting the *SleepModeEn* parameter in Bank 0 to 1.

2. Configure the maximum time a remote in sleep mode will remain awake after receiving an ACK, processing a message addressed to it, or receiving a serial or SPI message by setting the *Wake-ResponseTime* parameter. The default response time is 500 ms. Note that the setting of this parameter is overridden by some *GpioEdgeTrigger* parameter settings.

6.0 DNT90M Hardware

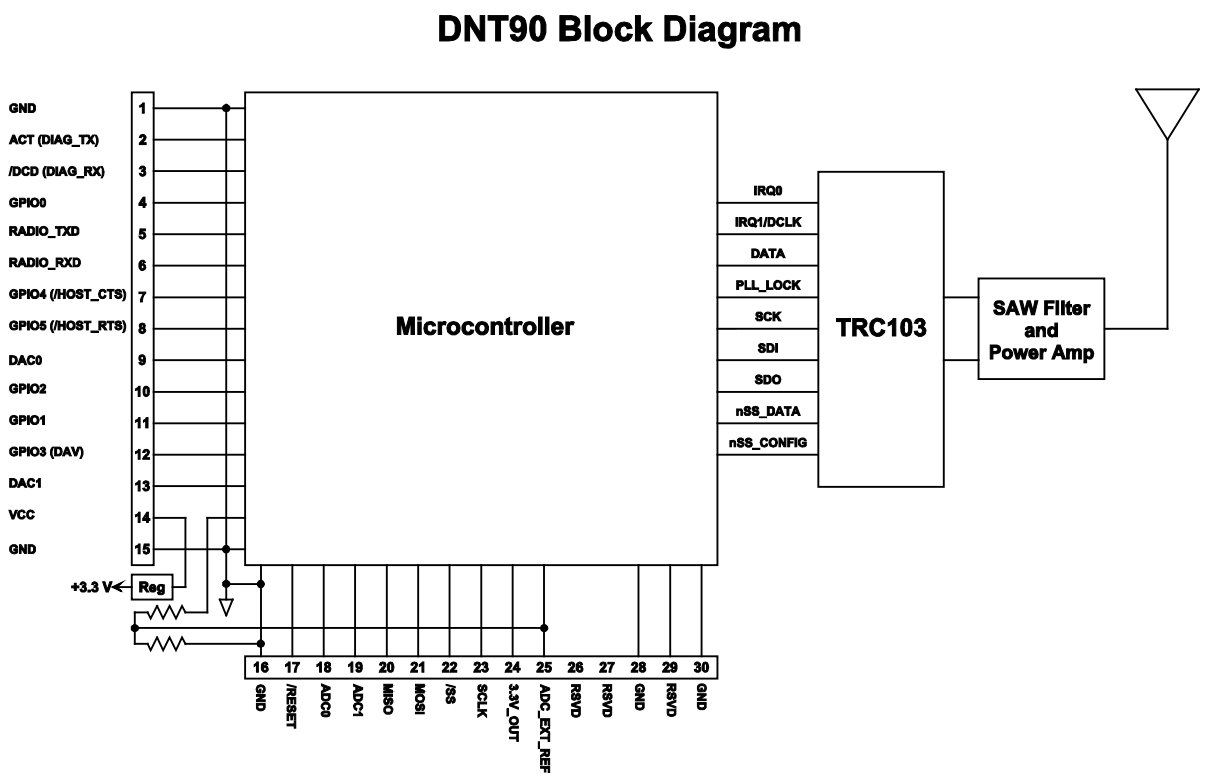


Figure 6.0.1

The major components of the DNT90M series modules include a 900 MHz FHSS transceiver and a low current 8-bit microcontroller. There are 64 selectable hopping patterns. DNT90M modules also have two selectable RF output power levels: 40 and 158 mW.

The DNT90M modules provide a variety of hardware interfaces. There are two serial ports plus one SPI port. Either the primary serial port or the SPI port can be selected for data communications. The second serial port is dedicated to diagnostics. The primary and diagnostic serial ports support most standard baud rates up to 250.0 kbps. The SPI port supports data rates up to 500 kbps. Also included are three ADC inputs, two DAC outputs and six general-purpose digital I/O ports. Four of the digital I/O ports support an optional interrupt-from-sleep mode when configured as inputs.

There are four module configurations in the DNT90M Series:

- The DNT90MC is designed for use with an external antenna and for solder reflow mounting.
- The DNT90MP is designed for use with an external antenna and for plug-in connector mounting.
- The DNT90MCA has a built-in chip antenna and is designed for solder reflow mounting.
- The DNT90MPA has a built-in chip antenna and is designed for plug-in connector mounting.

6.1 Specifications

Absolute Maximum Rating	Value	Units
Power Supply Input	-0.5 to +6.5	V
All Input/Output Pins	-0.5 to +3.3	V
Input Power to RFIO Port	0	dBm
Non-operating Ambient Temperature Range	-40 to +85	^{o}C

Table 6.1.1

Operating Characteristic	Sym	Minimum	Typical	Maximum	Units
RF Communication Topology		Direct Peer-to-Peer			
Spread Spectrum Mode		Frequency Hopping			
Operating Frequency Range		902.76		926.76	MHz
Number of RF Channels		26			
Number of Hopping Patterns		64			
Hop Duration				400	ms
Modulation		FSK			
RF Data Transmission Rate		100			kbps
Receiver Sensitivity, 10^{-5} BER			-100		dBm
Transmitter RF Output Power		40 or 158			mW
Antenna Impedance, DNT90MC and DNT90MP			50		Ω
RF Connection, DNT90MC and DNT90MP		U.FL Connector			
ADC Input Range		0		2.7	V
ADC Input Resolution				12	bits
ADC Sample Rate			100		Hz
Signal Source Impedance for ADC Reading				10	KΩ
ADC External Reference Voltage Range		1.0		2.7	V
DAC Output Range		0		3.3	V
DAC Output Resolution				12	bits
Primary and Diagnostic Serial Port Baud Rates		1.2, 2.4, 4.8, 9.6, 19.2, 14.4 28.8, 38.4, 57.6, 115.2, 230.4, 250			kbps
Master Serial Peripheral Interface Data Rate		125	250	500	kbps
Slave Serial Peripheral Interface Data Rate				4000	kbps
Digital I/O:					
Logic Low Input Level		-0.5		0.8	V
Logic High Input Level		2.45		3.3	V
Logic Input Internal Pull-up Resistor			20		KΩ
Power Supply Voltage Range	V_{CC}	+3.3		+5.5	Vdc
Power Supply Voltage Ripple				10	mV_{P-P}
Peak Transmit Current, 158 mW Output				170	mA
Receive Current			40		mA
Sleep Current			3	6	µA
Operating Temperature Range		-40		85	^{o}C
Operating Relative Humidity Range, Non-condensing		10		90	%

Table 6.1.2

6.2 Module Pin Out

Electrical connections to the DNT90MC are made through the I/O pads and through the I/O pins on the DNT90MP. The hardware I/O functions are detailed in the table below:

Pin	Name	I/O	Description
1	GND	-	Power supply and signal ground. Connect to the host circuit board ground.
2	ACT (DIAG_TX)	O (O)	This pin's default configuration is transmitter activity (ACT) output. The ACT signal is asserted whenever any data packet other than just an ACK is transmitted. The alternate function for this pin is the diagnostic serial port output.
3	/DCD (DIAG_RX)	O (I)	The /DCD signal is asserted when a DNT90M receives a valid packet. If a radio is transmitting data and receiving ACKs, both the ACT and /DCD signals will be asserted. If a radio is receiving packets only, the /DCD signal will be asserted. The alternate function for this pin is the diagnostic serial port input.
4	GPIO0	I/O	Configurable digital I/O port 0. When configured as an input, an internal pull-up resistor can be selected and direct interrupt from sleep can be invoked. When configured as an output, the power-on state is configurable. In sleep mode the pin direction, input pull-up selection or output state are also separately configurable.
5	RADIO_TXD	O	Serial data output from the radio.
6	RADIO_RXD	I	Serial data input to the radio.
7	GPOI4 (/HOST_CTS)	I/O (O)	GPIO4 with the same configuration options as GPIO2. Alternate pin function is UART/SPI flow control output. The module sets this line low when it is ready to accept data from the host on the RADIO_RXD or MOSI input. When the line goes high, the host must stop sending data.
8	GPIO5 (/HOST_RTS)	I/O (I)	GPIO5 with the same configuration options as GPIO2. Alternate pin function is UART/SPI flow control input. The host sets this line low to allow data to flow from the module on the RADIO_TXD pin. When the host sets this line high, the module will stop sending data to the host.
9	DAC0	O	12-bit DAC 0 output. Full scale can be referenced to the voltage at pin 25, or the 3.3 V regulated module bus voltage.
10	GPIO2	I/O	Configurable digital I/O port 2. Same configuration options as GPIO0.
11	GPIO1	I/O	Configurable digital I/O port 1. Same configuration options as GPIO0.
12	GPIO3 (DAV)	I/O (O)	Default pin function is GPIO3 with the same configuration options as GPIO0. When SPI slave mode operation is enabled, a logic high on this pin indicates when data is available to be clocked out by the SPI master.
13	DAC1	O	12-bit DAC 1 output. Same specifications and configuration options as DAC0.
14	VCC	I	Power supply input, +3.3 to +5.5 Vdc.
15	GND	-	Power supply and signal ground. Connect to the host circuit board ground.
16	GND	-	Power supply and signal ground. Connect to the host circuit board ground.
17	/RESET	I	Active low module hardware reset.
18	ADC0	I	ADC input 0. This pin is a direct ADC input when the ADC is operating in single-ended mode, or the differential negative input for positive inputs applied to ADC1 or ADC2 when the ADC is operating in differential mode. Full-scale reading can be referenced to Pin 25 for ratiometric measurements. For absolute measurements, the ADC can use the regulated supply voltage divided by 1.6 (about 2.06 V), or an external voltage applied to Pin 25. In single-ended mode, ADC measurements are 11-bit unsigned values with full scale nominally 2.7 V when referenced to a 2.7 V input on Pin 27. In differential mode, ADC measurements are 12-bit signed values.
19	ADC1	I	ADC input 1. Direct input when the ADC is operating in single-ended mode, positive differential input relative to ADC0 when the ADC is operating in differential mode.
20	MISO	I/O	This pin is the SPI master mode input or slave mode output.
21	MOSI	I/O	This pin is the SPI master mode output or slave mode input.
22	/SS	I/O	SPI active low slave select. This pin is an output when the module is operating as a master, and an input when it is operating as a slave.
23	SCLK	I/O	SPI clock signal. This pin is an output when operating as a master, and an input when operating as a slave.

Pin	Name	I/O	Description (continued)
24	ADC2	I	ADC input 2. Direct input when the ADC is operating in single-ended mode, positive differential input relative to ADC0 when the ADC is operating in differential mode.
25	ADC_EXT_REF	I/O	ADC external reference voltage pin. The voltage at this pin can be used by the ADCs as a reference for ratiometric measurements. With no external voltage or load applied, this pin presents a nominal 2.7 V output through a 2.126 K source resistance. A low impedance external reference voltage in the range of 1.0 to 2.7 V may be applied to this pin as an option.
26	RSVD	-	Reserved pin. Leave unconnected.
27	RSVD	-	Reserved pin. Leave unconnected.
28	GND	-	Connect to the host circuit board ground plane.
29	RSVD	-	Reserved pin. Leave unconnected.
30	GND	-	Connect to the host circuit board ground plane.

Table 6.2.1

6.3 Antenna Connector

A U.FL miniature coaxial connector is provided on both DNT90MC and DNT90MP for connection to the RFIO port. A short U.FL coaxial cable can be used to connect the RFIO port directly to an antenna. In this case the antenna should be mounted firmly to avoid stressing the U.FL coaxial cable due to antenna mounting flexure. Alternately, a U.FL coaxial jumper cable can be used to connect the DNT90M module to a U.FL connector on the host circuit board. The connection between the host circuit board U.FL connector and the antenna or antenna connector on the host circuit board should be implemented as a

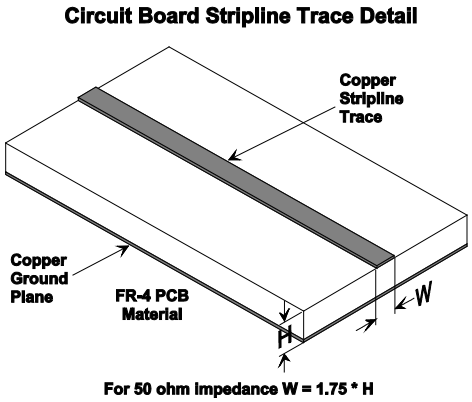


Figure 6.3.1

Trace Separation from 50 ohm Microstrip	Length of Trace Run Parallel to Microstrip
100 mil	125 mill
150 mil	200 mil
200 mil	290 mil
250 mil	450 mil
300 mil	650 mil

Table 6.3.2

50 ohm stripline. Referring to Figure 6.3.1, the width of this stripline depends on the thickness of the circuit board between the stripline and the groundplane. For FR-4 type circuit board materials (dielectric constant of 4.7), the width of the stripline is equal to 1.75 times the thickness of the circuit board.

Note that other circuit board traces should be spaced away from the stripline to prevent signal coupling, as shown in Table 6.3.2. The stripline trace should be kept short to minimize its insertion loss.

6.4 Power Supply and Input Voltages

DNT90M radio modules can operate from an unregulated DC input (Pad 19) in the range of 3.3 to 5.5 V with a maximum ripple of 5% over the temperature range of -40 to 85 °C. *Applying AC, reverse DC, or a DC voltage outside the range given above can cause damage and/or create a fire and safety hazard. Further, care must be taken so logic inputs applied to the radio stay within the voltage range of 0 to 3.3 V. Signals applied to the analog inputs must be in the range of 0 to ADC_EXT_REF (Pad/Pin 25). Applying a voltage to a logic or analog input outside of its operating range can damage the DNT90M module.*

6.5 ESD and Transient Protection

All DNT90M radio modules are electrostatic discharge (ESD) sensitive. ESD precautions must be observed when handling and installing these components. Installations must be protected from electrical transients on the power supply and I/O lines. This is especially important in outdoor installations, and/or where connections are made to sensors with long leads. *Inadequate transient protection can result in damage and/or create a fire and safety hazard.*

6.6 Interfacing to 5 V Logic Systems

All logic signals including the serial ports on the DNT90M are 3.3 V signals. To interface to 5 V signals, the resistor divider network shown in Figure 3.7.1 below must be placed between the 5 V signal outputs and the DNT90M signal inputs. The output voltage swing of the DNT90M 3.3 V signals is sufficient to drive 5 V logic inputs.

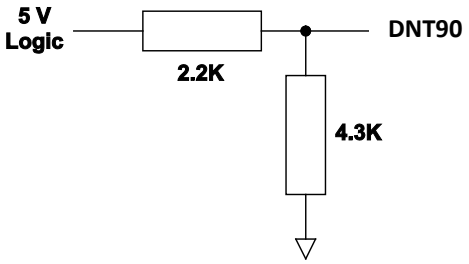


Figure 6.6.1

6.7 Mounting and Enclosures

DNT90MC and DNT90MCA radio modules are mounted by reflow soldering them to a host circuit board. DNT90MP and DNT90MPA modules are mounted by plugging their pins into a set of mating connectors on the host circuit board. Refer to Section 10.3 for connector details.

DNT90M radio enclosures must be made of plastics or other materials with low RF attenuation to avoid compromising antenna performance where antennas are internal to the enclosure. Metal enclosures are not suitable for use with internal antennas as they will block antenna radiation and reception. Outdoor enclosures must be water tight, such as a NEMA 4X enclosure.

6.8 Labeling and Notices

DNT90M FCC Certification - The DNT90M hardware has been certified for operation under FCC Part 15 Rules, Section 15.247. *The antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter.*

DNT90M FCC Notices and Labels - *This device complies with Part 15 of the FCC rules. Operation is subject to the following two conditions: (1) this device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.*

A clearly visible label is required on the outside of the user's (OEM) enclosure stating the following text:

Contains FCC ID: HSW-DNT90
Contains IC: 4492A-DNT90

Murata (Insert Model Designation DNT90MC DNT90MCA DNT90MP or DNT90MPA depending on the model used): This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

WARNING: This device operates under Part 15 of the FCC rules. Any modification to this device, not expressly authorized by Murata, may void the user's authority to operate this device.

This apparatus complies with Health Canada's Safety Code 6 / IC RSS 210.

IC RSS-210 Notice - *Operation is subject to the following two conditions: (1) this device may not cause interference, and (2) this device must accept any interference, including interference that may cause undesired operation of the device.*

ICES-003

This digital apparatus does not exceed the Class B limits for radio noise emissions from digital apparatus as set out in the radio interference regulations of Industry Canada.

Le present appareil numerique n'emet pas de bruits radioelectriques depassant les limites applicables aux appareils numeriques de Classe B prescrites dans le reglement sur le brouillage radioelectrique edicte par Industrie Canada.

7.0 DNT90M Protocol-formatted Messages

7.1 Protocol Formats

DNT90M modules can work in one of two serial data modes - transparent or protocol. Transparent mode requires no data formatting, but is limited to sending data to either a single destination or broadcasting data to all destinations. A node that needs to send messages to individual destinations must use protocol formatting unless the data being sent includes addressing information. Protocol formatting is also required for configuration commands and replies, and sensor I/O commands, replies and events. All protocol-formatted messages have a common header as shown in Figure 7.1.1:

0	1	2	3 …
SOP	Length	PktType	variable number of arguments …

Figure 7.1.1

The scale above is in bytes.

The *Start-of-Packet* (SOP) character, 0xFB, is used to mark the beginning of a protocol-formatted message and to assure synchronization in the event of a glitch on the serial port at startup.

The *Length* byte is defined as the length of the remainder of the message following the length byte itself, or the length of the entire message - 2.

The *Packet Type* (PktType) byte specifies the type of message. It is a bitfield-oriented specifier, decoded as follows:

Bits 7..6	Reserved for future use
Bit 5	Event - this bit is set to indicate an event message
Bit 4	Reply - this bit is set to indicate a message is a reply
Bits 3..0	Type - these bits indicate the message type

As indicated, the lower four bits (3..0) specify a message type. Bit 4 indicates that the message is a reply. A reply message has the original command type in bits 3..0, with Bit 4 set to one. Bit 5 indicates an event message. Arguments vary in size and number depending on the type of message and whether it is a message sent from the host, or is a reply or event message from the radio. See Section 7.3 below.

7.2 Message Types

Messages generated on the serial interface by the user are referred to as *host* messages. Messages generated on the serial interface by the radio are referred to as *reply* or *event* messages. Host messages carry commands. For most commands, there is a corresponding reply message. For example, when the host sends a *TxData* command message, the radio can return a *TxDataReply* message to indicate the status of the transmission - whether it succeeded or failed. To assist in interpreting the command-reply data flow, the direction is indicated by the high nibble in the message type. For example, an *EnterProtocolMode* command from the host is a message type 0x00, and the *EnterProtocolModeReply* from the radio is a message type 0x10.

Event messages from a DNT90M, such as received data or status announcements make up a third category of messages. Event messages, including *RxData*, *RxEvent* and *Announce* packets are indicated by 0x20 in the high nibble of the type byte**.** If multiple arguments are to be provided, they are to be concatenated in the order shown in Section 7.3 below. Little-Endian byte order is used for all multi-byte
arguments except text strings. Little-Endian byte order places the lowest order byte in the left-most byte of the argument and the highest order byte in the right-most byte of the argument.

7.3 Message Format Details

Table 7.3.1 below summarizes the DNT90M protocol-formatted messages:

Command	Reply	Event	Type	Direction
0x00	-	-	*EnterProtocolMode*	from Host
-	0x10	-	*EnterProtocolModeReply*	from Radio
0x01	-	-	*ExitProtocolMode*	from Host
0x02	-	-	*DeviceReset*	from Host
-	0x12	-	*DeviceResetReply*	from Radio
0x03	-	-	*GetRegister*	from Host
-	0x13	-	*GetRegisterReply*	from Radio
0x04	-	-	*SetRegister*	from Host
-	0x14	-	*SetRegisterReply*	from Radio
0x05	-	-	*TxData*	from Host
-	0x15	-	*TxDataReply*	from Radio
0x06	-	-	*GetRemoteRegister*	from Host
-	0x16	-	*GetRemoteRegisterReply*	from Radio
0x07	-	-	*SetRemoteRegister*	from Host
-	0x17	-	*SetRemoteRegisterReply*	from Radio
-	-	0x26	*RxData*	from Radio
-	-	0x27	*Announce/Error*	from Radio
-	-	0x28	*RxEvent*	from Radio

Table 7.3.1

EnterProtocolMode command and reply format details are presented in Tables 7.3.2 and 7.3.3:

Enter Protocol Mode Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x07 = Number of bytes in message following this byte
0x02	Packet Type	0x00 = EnterProtocolMode
0x03 - 0x08	Payload	String = “DNTCFG” or 0x44 0x4E 0x54 0x43 0x46 0x47

Table 7.3.2

Enter Protocol Mode Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x01 = Number of bytes in message following this byte
0x02	Packet Type	0x10 = EnterProtocolModeReply

Table 7.3.3

ExitProtocolMode command format details are shown in Table 7.3.4:

Exit Protocol Mode Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x01 = Number of bytes in message following this byte
0x02	Packet Type	0x01 = ExitProtocolMode

Table 7.3.4

DeviceReset command and reply format details are shown in Tables 7.3.5 and 7.3.6:

Device Reset Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x02 = Number of bytes in message following this byte
0x02	Packet Type	0x02 = DeviceReset
0x03	Reset Type	0x00 = Normal Device Reset 0x01 = Reset to Serial Bootloader 0x02 = Reset to Over-the-Air Bootloader

Table 7.3.5

Device Reset Reply		
Byte Offset	**Field**	**Description**
0x00	Start-Of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x01 = Number of bytes in message following this byte
0x02	Packet Type	0x12 = DeviceResetReply

Table 7.3.6

GetRegister command and reply format details are shown in Tables 7.3.7 and 7.3.8:

Get Register Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x04 = Number of bytes in message following this byte
0x02	Packet Type	0x03 = GetRegister
0x03	Register Offset	Register offset in its bank
0x04	Register Bank	Register bank number
0x05	Register Size	Register size in bytes, only one parameter at a time (wrong register size will produce an error response)

Table 7.3.7

Get Register Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x05 to 0x14 = Number of bytes in message following this byte
0x02	Packet Type	0x13 = GetRegisterReply
0x03	Register Offset	Register offset in its bank
0x04	Register Bank	Register bank number
0x05	Register Size	Register size in bytes
0x06 - 0x15	Register Value	Register value, all bytes in the register (only one parameter at a time)

Note: an *Error* message will be returned instead of a *GetRegisterReply* in case of a format error.

Table 7.3.8

SetRegister command and reply format details are shown in Tables 7.3.9 and 7.3.10:

Set Register Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x05 to 0x14 = Number of bytes in message following this byte
0x02	Packet Type	0x04 = SetRegister
0x03	Register Offset	Register offset in its bank
0x04	Register Bank	Register bank number
0x05	Register Size	Register size in bytes
0x06 - 0x15	Register Value	Register value, all bytes in the register (only one parameter at a time)

Table 7.3.9

Set Register Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x01 = Number of bytes in message following this byte
0x02	Packet Type	0x14 = SetRegisterReply

Note: an *Error* message will be returned instead of a *SetRegisterReply* in case of a format error.

Table 7.3.10

TXData command and reply format details are shown in Tables 7.3.11 and 7.3.12:

TX Data Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x04 to 0x6B = Number of bytes in message following this byte
0x02	Packet Type	0x05 = TxData
0x03 - 0x05	Destination MAC Address	Destination MAC address, in Little Endian byte order
0x06 - 0x72	Tx Data	Up to 103 bytes

Table 7.3.11

TX Data Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x07 = Number of bytes in message following this byte
0x02	Packet Type	0x15 = TxDataReply
0x03 - 0x05	Destination MAC Address	Destination MAC address, in Little Endian byte order
0x06	Status	0x00 = ACK received from destination 0x01 = no ACK received from destination (NAK)
0x07	RSSI	Packet RX power in dBm, -128 to 126 or 127 if invalid

Note: *TxDataReply* messages are only returned to the host when the *EndToEndAckEnable* parameter is set to 0x01.

Table 7.3.12

GetRemoteRegister command and reply details are shown it Tables 7.3.13 and 7.3.14:

Get Remote Register Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x07 = Number of bytes in message following this byte
0x02	Packet Type	0x06 = GetRemoteRegister
0x03 - 0x05	Destination MAC Address	Destination MAC address, in Little Endian byte order
0x06	Register Offset	Register offset in its bank
0x07	Register Bank	Register bank number
0x08	Register Size	Register size in bytes, only one parameter at a time (wrong register size will produce an error response)

Table 7.3.13

Get Remote Register Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x0A to 0x19 = Number of bytes in message following this byte
0x02	Packet Type	0x16 = GetRemoteRegisterReply
0x03	Status	Error status (0x00 = No Error, 0xE1 = Invalid Argument)
0x04 - 0x06	Originator MAC Address	Originator's MAC address, in Little Endian byte order
0x07	RSSI	(-128 to 126 or 127 if invalid)
0x08	Register Offset*	Register offset in its bank
0x09	Register Bank*	Register bank number
0x0A	Register Size*	Register size in bytes
0x0B - 0x1A	Register Value*	Register value, all bytes in the register (only one parameter at a time)

*Bytes eight through the end of the message will not be returned in case of an error

Table 7.3.14

SetRemoteRegister command and reply format details are shown in Tables 7.3.15 and 7.3.16:

Set Remote Register Command		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	Number of bytes in message following this byte
0x02	Packet Type	0x07 = SetRemoteRegister
0x03 - 0x05	Destination MAC Address	Destination MAC address, in Little Endian byte order
0x06	Register Offset	Register offset in its bank
0x07	Register Bank	Register bank number
0x08	Register Size	Register size in bytes
0x09 - 0x18	Register Value	Register contents

Table 7.3.15

Set Remote Register Reply		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x06 = Number of bytes in message following this byte
0x02	Packet Type	0x17 = SetRemoteRegisterReply
0x03	Status	Error status: 0x00 = no error, 0xE1 = invalid argument
0x04 - 0x06	Originator MAC Address	Originator's MAC address, in Little Endian byte order
0x07	RSSI	Packet RX power in dBm, -128 to 126, or 127 if invalid

Table 7.3.16

RxData event packet format details are shown in Figure Table 7.3.17:

RX Data Packet		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x05 to 0x6C = Number of bytes in message following this byte
0x02	Packet Type	0x26 = RxData event message
0x03 - 0x05	Originator MAC Address	Originator's MAC address, in Little Endian byte order
0x06	RSSI	Packet RX power in dBm, -128 to 126, or 127 if invalid
0x07 - 0x73	Rx Data	Up to 103 data bytes

Table 7.3.17

Announce/Error message format details are shown in Tables 7.3.18 through 7.3.19:

Startup Announcement or Error Code		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x02 = Number of bytes in message following this byte
0x02	Packet Type	0x27 = Indicates this is an Announce/Error message
0x03	Announce Status	0xA0 = Startup initialization complete 0xE1 = Invalid argument 0xE4 = Register read only error 0xEC = Brownout reset 0xED = Watchdog reset 0xEE = Hardware Error (Crystal or Radio Error)

Table 7.3.18

Heartbeat Announcement		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x07 = number of bytes in message following this byte
0x02	Packet Type	0x27 = Indicates this is an Announce/Error message
0x03	Announce Status	0xA8 = Heartbeat message
0x04 - 0x06	Originator MAC Address	MAC address of originator, in Little Endian byte order
0x07	RSSI	Received power in dBm, -128 to 126, 127 if invalid
0x08	Beacon RX Power	Average beacon RX power in dBm, uses 0.0625 “alpha” averaging filter, -128 to 126 or 127 if invalid

Table 7.3.19

RxEvent message format details are shown in Table 7.3.20:

RX Event Packet		
Byte Offset	**Field**	**Description**
0x00	Start-of-Packet	0xFB = Indicates start of protocol formatted message
0x01	Length	0x12 = number of bytes in message following this byte
0x02	Packet Type	0x28 = RxEvent
0x03 - 0x05	Originator MAC Address	Originator's MAC address, in Little Endian byte order
0x06	RSSI	Packet RX power in dBm (-128 to 127)
0x07	GPIO Readings	Bit Field (GPIO0..GPIO5) indicating GPIO readings
0x08 - 0x09	ADC0 Reading	ADC0 Reading, 0x0000 - 0x0FFF, in Little Endian byte order
0x0A - 0x0B	ADC1 Reading	ADC1 Reading, 0x0000 - 0x0FFF, in Little Endian byte order
0x0C - 0x0D	ADC2 Reading	ADC2 Reading, 0x0000 - 0x0FFF, in Little Endian byte order
0x0E - 0x0F	Event Flags	Bit Field Indicating which events have occurred: Bit 0: GPIO0 Triggered Bit 1: GPIO1 Triggered Bit 2: GPIO2 Triggered Bit 3: GPIO3 Triggered Bit 4: Periodic Report Interval Bit 5: ADC0 Threshold Triggered Bit 6: ADC1 Threshold Triggered Bit 7: ADC2 Threshold Triggered Bits 8-15: Unused (0)
0x10 - 0x11	DAC0 Setting	DAC0 setting, 0x0000 - 0x0FFF, in Little Endian byte order
0x12 - 0x13	DAC1 Setting	DAC1 setting, 0x0000 - 0x0FFF, in Little Endian byte order

Table 7.3.20

7.4 Configuration Parameter Registers

The configuration parameters in a DNT90M module are stored in a set of variable length *registers*. Most registers are read-write, with a few read-only or write-only. Changes made to the register settings are temporary until a *MemorySave* command is executed. Resetting or power-cycling the module will clear any changes that have not been saved to permanent memory using the *MemorySave* command. DNT90M modules can be configured to start in protocol mode at power-up, in which case the *EnterProtocolMode* command is not required.

7.4.1 Bank 0x00 - Transceiver Setup

Bank	Location	Name	R/W	Size	Range	Default
0x00	0x00	Reserved	-	1	-	-
0x00	0x01	Reserved	-	1	-	-
0x00	0x02	Reserved	-	1	-	-
0x00	0x03	SecurityKey	*R/W*	16	0..2^128-1	0
0x00	0x13	SleepModeEn	*R/W*	1	0..2	0 (off)
0x00	0x14	WakeResponseTime	*R/W*	2	0..30000	500 (500 ms)
0x00	0x16	Reserved	-	1	-	-
0x00	0x17	Reserved	-	1	-	-
0x00	0x18	TxPower	*R/W*	1	0..1	1 (+22 dBm)
0x00	0x19	UserTag	*R/W*	16	string	“DNT90M”
0x00	0x29	RmtTransDestAddr	*R/W*	3	0..0xFFFFFF	0xFFFFFF
0x00	0x2C	Reserved	-	1	-	-
0x00	0x2D	Reserved	-	1	-	-
0x00	0x2E	HeartbeatIntrvl	*R/W*	2	0..0xFFFF	0xFFFF (disabled)
0x00	0x30	SystemId	*R/W*	1	0..63	0
0x00	0x31	AckEnable	*R/W*	1	0..1	0 (disabled)
0x00	0x32	Reserved	-	2	-	-
0x00	0x34	Reserved	-	2	-	-
0x00	0x36	Reserved	-	1	-	-
0x00	0x37	MaxUnicastFrames	*R/W*	1	1..16	5
0x00	0x38	MaxBcstFrames	*R/W*	1	1..16	5

Table 7.4.1.1

SecurityKey - this 16-byte parameter sets the 128-bit AES encryption key. To protect the key, it is a write-only parameter for the user. It always reads back as 0x2A.

SleepModeEn - this parameter enables/disables sleep mode. Sleep mode is used in conjunction with the automatic I/O reporting feature to wake up on specific triggers. The default value for this parameter is 0 (off). Setting this parameter to 1 invokes sleep mode immediately. Setting this parameter to 2 invokes sleep mode following reset, allowing this and other parameter updates to be stored before sleep mode is invoked.

WakeResponseTime - this parameter set how long sleep is deferred in a DNT90M remote configured for sleep mode after:

receiving an ACK
receiving a packet that requires processing by the device
receiving a protocol packet from the device’s local host

TxPower - this parameter sets the transmit power level (default is 0x01):

0x00 = +16 dBm or 40 mW
0x01 = +22 dBm or 158 mW

UserTag - this parameter is a user definable field intended for use as a location description or other identifying tag such as a “friendly name”.

RmtTransDestAddr - this parameter holds the default destination for transparent mode data packets and event packets. This parameter defaults to the broadcast address, 0xFFFFFF.

HeartbeatInterval - when set to 0x0000, all heartbeats are disabled. When set to 0xFFFF (default), periodic heartbeats are disabled but the initial heartbeat at power up is enabled. The periodic heartbeat interval is scaled 1 second/count, and applies to DNT90Ms where sleep mode is disabled. DM90M radios with sleep mode enabled must have periodic reports and/or ADC sampling enabled for heartbeats to be generated.

SystemId - this parameter holds the ID for a DTN90 system. DNT90M systems that may physically overlap must have different system IDs.

AckEnable - when this parameter is set to 1 and the DNT90M is in protocol mode, the originator will indicate in its transmitted packet that an ACK message is expected from the packet’s destination node, which is passed on to the originator’s host.

MaxUnicastFrames - this parameter sets the maximum number of packets addressed to a specific node that will be transmitted on a channel before switching to the next channel in the frequency hopping sequence.

MaxBcstFrames - this parameter sets the maximum number of different broadcast packets that will be transmitted on a channel before switching to the next channel in the frequency hopping sequence.

7.4.2 Bank 0x01 - System Settings

Bank	Location	Name	R/W	Size	Range	Default
0x01	0x00	Reserved	-	1	-	-
0x01	0x01	Reserved	-	1	-	-
0x01	0x02	Reserved	-	1	-	-
0x01	0x03	Reserved	-	1	-	-
0x01	0x04	BcstAttemptLimit	*R/W*	1	0..254	1
0x01	0x05	ArqAttemptLimit	*R/W*	1	1..255	10
0x01	0x06	Reserved	-	1	-	-
0x01	0x07	P2PReplyTimeout	*R/W*	1	0..255	10 (1 second)
0x01	0x08	RegistryTimeout	*R/W*	1	0..255	5 (seconds)
0x01	0x09	Reserved	-	1	-	-

Table 7.4.2.1

BcstAttemptLimit - setting this parameter to 0 enables automatic broadcast message repeats based on the *ArqAttemptLimit* parameter value. Setting this parameter to a value between 1 and 254 specifies the number of broadcast message repeats independent of the *ArqAttemptLimit*. This parameter should not be set to 0 if *ArqAttemptLimit* is set to 255.

ArqAttemptLimit - this sets the maximum number of attempts that will be made to send a message on the RF link. Setting this parameter to the maximum value of 255 is a flag value indicating that there should be no limit to the number of attempts to send each packet (infinite number of attempts). This mode is intended for point-to-point networks in serial data cable replacement applications where absolutely no packets can be lost.

P2PReplyTimeout - this parameter sets the reply timeout for messages sent from one node to another.

RegistryTimeout - this parameter sets the number of seconds a DNT90M will hold the Transaction ID (TID) history for another DNT90M before discarding the history. Any communication between these radios before the timeout occurs restarts the timeout counter. The TID is used to filter out duplicate packets.

7.4.3 Bank 0x02 - Status Parameters

Bank	Location	Name	R/W	Size	Range	Default
0x02	0x00	MacAddress	*R*	3	0..0xFFFFFF	Fixed value
0x02	0x03	Reserved	-	-	-	-
0x02	0x04	Reserved	-	-	-	-
0x02	0x05	Reserved	-	-	-	-
0x02	0x06	Reserved	-	-	-	-
0x02	0x07	Reserved	-	-	-	-
0x02	0x08	HardwareVersion	*R*	1	0x41..0x5A	Current HW
0x02	0x09	FirmwareVersion	*R*	1	0x00..0xFF	Current FW load
0x02	0x0A	FirmwareBuildNum	*R*	2	0..0xFFFF	Current FW load
0x02	0x0C	FirmwareBuildDate	*R*	3	BCD (“YYMMDD”)	Current FW load
0x02	0x0F	FirmwareBuildTime	*R*	3	BCD (“HHMMSS”)	Current FW load
0x02	0x12	RssiIdle	*R*	1	-128..127	Current Value
0x02	0x13	RssiLast	*R*	1	-128..127	Current Value
0x02	0x14	AvgBeaconPower	*R*	1	-128..127	Current Value
0x02	0x15	Reserved	-	-	-	-
0x02	0x18	ModelNumber	*R*	1	0x90	indicates DNT90M
0x02	0x19	OtaTxQueueBusy	*R*	1	0..1	Current Value

Table 7.4.3.1

MacAddress - this parameter holds the radio's unique 24-bit MAC address.

HarwareVersion - this parameter holds an identifier indicating the hardware revision (ASCII character).

FirmwareVersion - this parameter holds the firmware version of the radio in 2-digit BCD format.

FirmwareBuildNum - this parameter holds the firmware build number, in binary format.

FirmwareBuildDate - this parameter holds the date of firmware build in MM/DD/YY format.

FirmwareBuildTime - this parameter holds the time of the firmware build in HH:MM:SS format.

RssiIdle - this 2’s compliment parameter holds the last RSSI measurement in dBm made during a time when the RF channel was idle. This parameter is useful for detecting interferers.

RssiLast - this 2’s compliment parameter holds the last RSSI measurement in dBm made during the receipt of an RF packet with a valid CRC. This parameter is useful for network commissioning/diagnostics.

AvgBeaconPower - this 2’s compliment parameter holds the alpha-filtered beacon power (dBm) received from a device’s parent, where alpha = 0.0625.

ModelNumber - this parameter specifies the DNT model, in this case a DNT90M.

OtaTxQueueBusy - this parameter indicates if the transmitter queue is currently holding bytes to transmit. If bytes are present in the queue the value is 1. If the queue is empty the value is 0.

7.4.4 Bank 0x03 - Serial and SPI Settings

Bank	Location	Name	R/W	Size	Range	Default
0x03	0x00	SerialRate	*R/W*	1	0..10	3 (9600 baud)
0x03	0x01	SerialParams	*R/W*	1	0..7	0 (8-N-1)
0x03	0x02	SpiMode	*R/W*	1	0..2	0 (SPI disabled)
0x03	0x03	SpiRateSel	*R/W*	1	0..2	0 (125 kHz)
0x03	0x04	SpiOptions	*R/W*	1	0..7	0
0x03	0x05	SpiMasterCmdLen	*R/W*	1	0..16	0
0x03	0x06	SpiMasterCmdStr	*R/W*	16	0..16 byte string	All 0x00 bytes

Table 7.4.4.1

SerialRate - this parameter sets the serial data rate as shown below:

Setting	***Serial rate***
0x00	1.2 kbps
0x01	2.4 kbps
0x02	4.8 kbps
0x03	9.6 kbps
0x04	14.4 kbps
0x05	19.2 kbps
0x06	28.8 kbps
0x07	38.4 kbps
0x08	57.6 kbps
0x09	115.2 kbps
0x0A	230.4 kbps
0x0B	250.0 kbps

SerialParams - this parameter sets the serial mode options for parity and stop bits:

Setting	***Mode***
0x00	No parity, 8 data bits, 1 stop bit (default)
0x01	No parity, 8 data bits, 2 stop bits
0x02	Reserved
0x03	Reserved
0x04	Even parity, 8 data bits, 1 stop bit
0x05	Even parity, 8 data bits, 2 stop bits
0x06	Odd parity, 8 data bits, 1 stop bit
0x07	Odd parity, 8 data bits, 2 stop bits

Note that 8-bit data with no parity is capable of carrying 7-bit data with parity for compatibility without loss of generality for legacy applications that may require it.

SpiMode - this parameter sets the SPI operating mode:

Setting	***Mode***
0x00	SPI disabled - serial UART mode (default)

0x01	SPI Slave mode
0x02	SPI Master mode

SpiRateSel - this parameter sets the SPI master mode clock rate:

Setting	***Mode***
0x00	125 kbps
0x01	250 kbps
0x02	500 kbps

SpiOptions - this parameter allows the SPI to be configured with the following options:

Setting	***Option***
0x00	Leading edge rising, sample leading edge, MSBs sent first
0x01	Leading edge rising, sample falling edge, MSBs sent first
0x02	Leading edge falling, sample leading edge, MSBs sent first
0x03	Leading edge falling, sample falling edge, MSBs sent first
0x04	Leading edge rising, sample leading edge, LSBs sent first
0x05	Leading edge rising, sample falling edge, LSBs sent first
0x06	Leading edge falling, sample leading edge, LSBs sent first
0x07	Leading edge falling, sample falling edge, LSBs sent first

SpiMasterCmdLen - this parameter sets the length for the SPI master command string that will be used to interrogate the slave peripheral, when SPI master mode is selected with periodic I/O reporting enabled.

SpiMasterCmdStr - this parameter holds the SPI master command string that is used to interrogate the slave peripheral when SPI master mode is selected and periodic I/O reporting is enabled.

7.4.5 Bank 0x04 - Host Protocol Settings

Bank	Location	Name	R/W	Size	Range	Default
0x04	0x00	ProtocolMode	*R/W*	1	0..1	0 (Transparent)
0x04	0x01	TxTimeout	*R/W*	1	0..255	0 (No timeout)
0x04	0x02	MinPacketLength	*R/W*	1	0..255	1 (byte)
0x04	0x03	TransPtToPtMode	*R/W*	1	0..1	1 (Last RX)
0x04	0x04	MsgsPerHop	*R/W*	1	1..8	8

Table 7.4.5.1

ProtocolMode - this parameter selects the host protocol mode. The default is 0x00, which is transparent mode, meaning the radio conveys whatever characters that are sent to it *transparently*, without requiring the host to understand or conform to the DNT90M's built-in protocol. This setting is recommended for point-to-point applications for legacy applications such as wire replacements where another serial protocol may already exist. Setting this parameter to 0x01 enables the DNT90M protocol formatting. It is not necessary to define the same protocol mode for all radios in a network. It is frequently useful to configure all radios in a point-to-multipoint network for transparent mode except the base, which should be in protocol mode. Note that it is possible for the host to switch the radio from transparent mode to protocol mod by transmitting an *EnterProtocolMode* command, and switching back to transparent mode by transmitting an *ExitProtocolMode* command.

TxTimeout - this parameter is used to group transparent data to be sent in a single transmission rather than being split over two hops. Messages sent over two hops can have gaps in the received data stream that can cause problems for the receiving application - for example, Modbus RTU. This parameter is the amount of time the DNT90M will wait without receiving a byte through the serial port before transmitting the data. Parameter units are in milliseconds. A message boundary is determined whenever a gap between consecutive characters is equal to or greater than the *TxTimeout* value, or the number of bytes reaches the *MinPacketLength*. Either condition will trigger a transmission. The default *TxTimeout* value is 0 ms, which means data will be sent as soon as the *MinPacketLength* requirement is met.

MinPacketLength - this parameter is similar to *TxTimeout* except it uses the number of bytes received instead of the amount of time without receiving a byte. The default is one byte. A transmission is triggered when either the number of bytes reaches *MinPacketLength* or a gap is detected between consecutive characters greater than *TxTimeout*.

TransPtToPtMode - when this parameter is set to 0, the destination address of transparent mode packets will be the configured *RemoteDestAddr*. When set to 1, the destination address will first be the *RemoteDestAddr,* but then will update to the originator of the most recent RX packet processed.

MsgsPerOtaPacket - this parameter sets the maximum number of messages a DNT90M can send in one packet. The default value is 8 messages, which is suitable for most applications.

7.4.6 Bank 0x05 - I/O Parameters

Bank	Location	Name	R/W	Size	Range In Bits	Default
0x05	0x00	All-IO	*R/W*	13	104	N/A
0x05	0x0D	Gpio0	*R/W*	1	1	0
0x05	0x0E	Gpio1	*R/W*	1	1	0
0x05	0x0F	Gpio2	*R/W*	1	1	0
0x05	0x10	Gpio3	*R/W*	1	1	0
0x05	0x11	Gpio4	*R/W*	1	1	0
0x05	0x12	Gpio5	*R/W*	1	1	0
0x05	0x13	Adc0	*R*	2	12	N/A
0x05	0x15	Adc1	*R*	2	12	N/A
0x05	0x17	Adc2	*R*	2	12	N/A
0x05	0x19	EventFlags	*R/W*	2	16	N/A
0x05	0x1B	Dac0	*R/W*	2	12	0
0x05	0x1D	Dac1	*R/W*	2	12	0

Table 7.4.6.1

All-IO - this 13-byte parameter packs all the following parameters into a single value. Note that the information in parameters GPIO0 through GPIO5 is compressed into a single byte to save space in the All-IO parameter. When the ADC is operating in differential mode, the ADC1 to ADC0 differential reading is stored in the ADC0 position, and the ADC2 to ADC0 differential reading is stored in the ADC1 position. The ADC2 reading is not used in ADC differential mode and this position is set to 0.

Gpio0 through Gpio5 - if a pin is configured as an output, writing to its corresponding parameter to sets the pin's logic state. If a pin is configured as an input, writing to its corresponding parameter enables or disables the pin's internal pull-up. Reading these registers returns the current level detected on the corresponding pins.
Adc0 through Adc2 - read-only parameters that return the current reading for the selected ADC channel (Little-Endian byte order). When the ADC is operating in differential mode, the ADC1 to ADC0 differential reading is stored in the ADC0

position, and the ADC2 to ADC0 differential reading is stored in the ADC1 position. The ADC2 reading is not used in ADC differential mode and this position is set to 0. Also, see the discussion of the *AdcSampleIntvl* parameter below.

EventFlags - used with the automatic I/O reporting feature, this parameter indicates which I/O events have been triggered since the last report (write 0x0000 to reset):

bits 15..8	Reserved
bit 7	ADC2 high/low threshold excursion
bit 6	ADC1 high/low threshold excursion
bit 5	ADC0 high/low threshold excursion
bit 4	Periodic timer report
bit 2	GPIO2 edge transition
bit 1	GPIO1 edge transition
bit 0	GPIO0 edge transition

Dac0 through Dac1 - sets the DAC outputs. The range of this parameter is 0x0000 to 0x0FFF.

7.4.7 Bank 0x06 - I/O Settings

Bank	Location	Name	R/W	Size	Range In Bits	Default
0x06	0x00	GpioDir	*R/W*	1	6	0x00 (All inputs)
0x06	0x01	GpioInit	*R/W*	1	6	0x00 (All zeros)
0x06	0x02	GpioAlt	*R/W*	1	6	0x00
0x06	0x03	GpioEdgeTrigger	*R/W*	1	8	0x01
0x06	0x04	GpioSleepMode	*R/W*	1	6	0x00 (Off)
0x06	0x05	GpioSleepDir	*R/W*	1	6	0x00 (All inputs)
0x06	0x06	GpioSleepState	*R/W*	1	6	0x00 (All zero)
0x06	0x07	Dac0Init	*R/W*	2	12	0x0000
0x06	0x09	Dac1Init	*R/W*	2	12	0x0000
0x06	0x0B	AdcSampleIntvl	*R/W*	4	32	0x0A (100 ms)
0x06	0x0F	Adc0ThresholdLo	*R/W*	2	12	0xF800
0x06	0x11	Adc0ThresholdHi	*R/W*	2	12	0x07FF
0x06	0x13	Adc1ThresholdLo	*R/W*	2	12	0xF800
0x06	0x15	Adc1ThresholdHi	*R/W*	2	12	0x07FF
0x06	0x17	Adc2ThresholdLo	*R/W*	2	12	0xF800
0x06	0x19	Adc2ThresholdHi	*R/W*	2	12	0x07FF
0x06	0x1B	IoReportTrigger	*R/W*	1	8	0x01 (GPIO0)
0x06	0x1C	IoReportInterval	*R/W*	4	32	30000 (ms)
0x06	0x20	IoPreDelay	*R/W*	1	8	8 (ms)
0x06	0x21	IoBindingEnable	*R/W*	1	1	0 (Disabled)
0x06	0x22	DacReference	*R/W*	1	2	0 (ADC_EXT_REF)
0x06	0x23	AdcReference	*R/W*	1	2	0 (ADC_EXT_REF)
0x06	0x24	AdcAveSelect	*R/W*	1	8	0x01
0x06	0x25	ExtAdcScaleFactor	*R/W*	2	16	0x8000
0x06	0x27	ExtAdcOffset	*R/W*	2	16	0x0000
0x06	0x29	ExtDacScaleFactor	*R/W*	2	16	0x8000
0x06	0x2B	ExtDacOffset	*R/W*	2	16	0x0000
0x06	0x2D	VccAdcScaleFactor	*R/W*	2	16	0x8000
0x06	0x2F	VccAdcOffset	*R/W*	2	16	0x0000
Bank	**Location**	**Name**	**R/W**	**Size**	**Range In Bits**	**Default**
0x06	0x31	VccDacScaleFactor	*R/W*	2	16	0x8000
0x06	0x33	VccDacOffset	*R/W*	2	16	0x0000

0x06	0x35	1VAdcScaleFactor	*R/W*	2	16	0x8000
0x06	0x37	1VAdcOffset	*R/W*	2	16	0x0000
0x06	0x39	1VDacScaleFactor	*R/W*	2	16	0x8000
0x06	0x3B	1VDacOffset	*R/W*	2	16	0x0000
0x06	0x3D	AdcDiffMode	*R/W*	1	8	0 (single-ended)
0x06	0x3E	AdcGainCh0	*R/W*	1	8	0 (gain = 1)
0x06	0x3F	AdcGainCh1	*R/W*	1	8	0 (gain = 1)
0x06	0x40	AdcDiffScaleFactorCh0	*R/W*	2	16	0x8000
0x06	0x42	AdcDiffOffsetCh0	*R/W*	2	16	0x0000
0x06	0x44	AdcDiffScaleFactorCh1	*R/W*	2	16	0x8000
0x06	0x46	AdcDiffOffsetCh1	*R/W*	2	16	0x0000
0x06	0x47	FastAdcPrescaler	*R/W*	1	1 byte, range 1..7	5 (÷128)
0x06	0x48	SlowAdcPresccaler	*R/W*	1	1 byte, range 0..7	2 (÷16)
0x06	0x49	MaxQueuedEvents	*R/W*	1	1 byte, range 0..20	8 (reports)
0x06	0x4A	AdcSkipCount	*R/W*	1	1 byte	0 (samples)

Table 7.4.7.1

GpioDir - this parameter is a bitmask that sets whether each GPIO is an input (0) or outputs (1). The default is all inputs.

GpioInit - this parameter is a bitmask that sets the initial value for any GPIOs which are enabled as outputs. For GPIOs enabled as inputs, this sets the initial pull-up setting.

GpioAlt - Specifies which GPIO pins will have their alternate functions enabled: Bit 2 - diversity toggle enable, Bit 3 - RS485 enable, Bit 4 - /HOST_CTS enable, Bit5 - /HOST_RTS enable.

Bit	Alternate Function	Default	Bit Mask
0	(none)	0	0x01
1	(none)	0	0x02
2	Diversity Toggle	0	0x04
3	RS485 (N/A in SPI Slave mode)	0	0x08
4	/Host_CTS (N/A in SPI Slave mode)	1	0x10
5	/HOST_RTS (N/A in SPI Slave mode)	1	0x20

Table 7.4.7.2

GpioEdgeTrigger -This parameter consists of a set of four 2-bit fields that define when GPIO triggers are enabled for I/O event reporting:

bits 7..6 GPIO3 edge function
bits 5..4 GPIO2 edge function
bits 3..2 GPIO1 edge function
bits 1..0 GPIO0 edge function

The bit values for each GPIO map to the following settings:

Value	GPIO edge behavior
11	Rising edge trigger, neither level keeps remote awake
10	Bidirectional edge trigger, neither level keeps remote awake
01	Rising edge trigger, holding high keeps remote awake
00	Falling edge trigger, holding low keeps remote awake

Table 7.4.7.3

GpioSleepMode - this parameter is a bitmask that enables configuring the I/O direction and state of GPIO0..GPIO5 when the module is sleeping. Bits 0..5 correspond to GPIO0..GPIO5. Setting a *Gpio-SleepMode* bit to 1 enables sleep mode configuration of the corresponding GPIO. Setting a *GpioSleepMode* bit to 0 causes the corresponding GPIO to remain configured as in active mode. Note that when the *GpioAlt* bit is set for GPIO4, the corresponding *GpioSleepMode* bit is ignored and GPIO4 is controlled directly by the *GpioSleepState* parameter bit 7.

GpioSleepDir - when *GpioSleepMode* is enabled, this parameter functions to set the direction of the GPIOs during a device's sleep period. This enables the user to provide alternate configurations during sleep that will help minimize current consumption. Bits 0..5 correspond to GPIO0..GPIO5. Setting a *GpioSleepDir* bit to 1 to specifies an output; 0 specifies an input.

GpioSleepState - when *GpioSleepMode* is enabled, this parameter functions as a bitmask to control the states of the GPIOs, the RADIO_TXD output, and the /HOST_CTS and /DCD outputs during a device's sleep period. This allows the user to set alternate configurations during sleep to minimize current consumption. Bits 0..5 correspond to GPIO0..GPIO5 respectively. Bit 6 sets the state of RADIO_TXD, and bit 7 sets the states of /HOST_CTS and /DCD. A sleep state bit is set to 1 to specify a high output or an internal pull-up on an input, or to 0 to specify a low output or no internal pull-up on an input. Bit 6 must be set low in order to achieve minimum sleep current (high impedance load assumed), and the other bits may need to be set low or high depending on their external loads. When bit 6 is set low, expect a serial "break" condition to occur as the module wakes from sleep. The serial break condition can be eliminated by setting bit 6 high, but sleep current will be increased.

Dac0Init - this parameter sets the initial value for DAC0 at startup.

Dac1Init - this parameter sets the initial value for DAC1 at startup.

AdcSampleIntvl - this parameter sets the frequency (sample interval) of ADC measurements used to determine if a threshold has been exceeded or in calculating an average measurement value. The ADC channels are read on each ADC cycle, along with the states of GPIO2 and GPIO3. Each *AdcSampleIntvl* count equals 10 ms. The default is 100 ms. This interval will be the worst-case latency for ADC generated interrupts. Note that *AdcSampleIntvl* is independent of *IoReportInterval* as the ADCs are read on both intervals.

Adc0..2ThresholdLo/Hi - these parameters set the thresholds to trigger an I/O report based on ADC measurements. If I/O reporting is enabled, a single event report containing the contents of the I/O bank is generated when a threshold is crossed. Reporting is edge-triggered with respect to threshold boundaries, not level-triggered. Additional reports are not triggered unless the ADC measurement first returns inside the threshold boundary and then crosses the threshold again. Triggers occur whenever one of the following inequalities is satisfied:

ADC*x* < ADC*x*_ThresholdLo
ADC*x* > ADC*x*_ThresholdHi

IoReportTrigger - a trigger event on any enabled trigger source will cause a DNT90M router or remote to send an event message to the base containing the entire current values of the Bank 5.

bit 7	ADC2 high/low thresholds
bit 6	ADC1 high/low thresholds
bit 5	ADC0 high/low thresholds
bit 4	Periodic timer
bit 3	GPIO3 edge
bit 2	GPIO2 edge
bit 1	GPIO1 edge
bit 0	GPIO0 edge

IoReportInterval - when periodic timer I/O reporting is enabled, this parameter sets the interval between reports. The parameter scaling is 10 ms/count, and the default report interval is every 30 seconds.

IoPreDelay - this parameter sets the time in milliseconds to delay collection of ADC readings after a module wakeup event occurs, to allow settling of ADC input voltages.

IoBindingEnable - this parameter enables I/O binding. Setting this parameter to 0x00 disables I/O binding (I/O mirroring) from a remote device. Setting this parameter 0x01 enables I/O mirroring. When enabled, the data from any received event report is used to drive the device's own outputs. GPIO2 will be set to the event report's GPIO0 reading, GPIO3 will be set to the event report's GPIO1 reading, and DAC0 and DAC1 will be set with the ADC0 and ADC1 readings respectively. Note that if the *AdcDiffMode* parameter is set to 1, I/O binding cannot be used.

DacReference - this parameter selects the reference voltage for the DACs:

Setting	***Reference***
0x00	ADC_EXT_REF
0x01	AVVC (Analog Vcc)
0x02	Reserved
0x03	Disable DAC operation

AdcReference - this parameter selects the reference voltage for the ADCs:

Setting	***Reference***
0x00	ADC_EXT_REF
0x01	Internal Vcc divided by 1.6
0x02	Reserved
0x03	Disable ADC operation

AdcAveSelect - this parameter selects the number of ADC measurements to average to produce each ADC reading, from 1 to 255 samples. Averaging over a larger number of measurements increases noise filtering but also increases the time it takes to generate a set of readings:

ADC Mode	Module Awake	Module Sleeping
Single-ended, reading all three channels	216 µs	381 µs
Differential, reading both channels	160 µs	273 µs

Table 7.4.7.4

ExtAdcScaleFactor - this parameter is the scale factor applied to an ADC measurement when the ADC reference is an external voltage. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

ExtAdcOffset - this parameter is the 2's complement offset added to the scaled ADC measurement when the ADC reference is an external voltage.

ExtDacScaleFactor - this parameter is the scale factor applied to a DAC measurement when the DAC reference is an external voltage. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

ExtDacOffset - this parameter is 2's complement the offset added to the scaled DAC measurement when the DAC reference is an external voltage.

VccAdcScaleFactor - this parameter is the scale factor applied to an ADC measurement when the ADC reference is Vcc/1.6. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

VccAdcOffset - this parameter is the 2's complement offset added to the scaled ADC measurement when the ADC reference is derived from Vcc/1.6.

VccDacScaleFactor - this parameter is the scale factor applied to a DAC measurement when the DAC reference is Vcc. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

VccDacOffset - this parameter is the 2's complement offset added to the scaled DAC measurement when the DAC reference is Vcc.

1VAdcScaleFactor - this parameter is the scale factor applied to an ADC measurement when the ADC reference is the 1 V internal reference. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

1VAdcOffset - this parameter is the 2's complement offset added to the scaled ADC measurement when the ADC reference is the internal 1 V reference.

1VDacScaleFactor - this parameter is the scale factor applied to a DAC measurement when the DAC reference is the internal 1 V reference. The scale factor parameter is multiplied by 32768. For example, the parameter value for a scale factor of 1.12 = 1.12 * 32768 = 36700.16 or 0x8F5C.

1VDacOffset - this parameter is the 2's complement offset added to the scaled DAC measurement when the DAC reference is the internal 1 V reference.

AdcDiffMode - a parameter value of 0 selects single-ended ADC mode. In this mode, negative sensor inputs are connected to ground and positive sensor inputs to ADC0, ADC1 and ADC2 respectively. Three ADC measurements are made in this mode with a range of 0x0000 to 0x07FF. A parameter value of 1 selects signed differential mode with gain. In this mode, the negative sensor inputs are connected to ADC0 and the positive inputs are connected to ADC1 and ADC2. Two ADC measurements are made in this mode, ADC1 to ADC0 and ADC2 to ADC0, with a range (signed) from 0xF800 to 0x07FF.

In differential mode, the *AdcGainCh0* and *AdcGainCh1* parameters can change the selected gain for the two ADC readings, and the *AdcDiff* scale factors and offsets, both supplied by the customer, are used.

AdcGainCh0 - this parameter sets the preamplifier gain applied when making a differential measurement of ADC1 relative to ADC0. Setting this parameter to 0x00 sets the gain to 1, 0x01 sets the gain to 2, 0x02 sets the gain to 4 and so on, up to 0x06 which sets the gain to 64. Note that the preamplifier output voltage saturates at 2.4 V regardless of the gain setting.

AdcGainCh1 - this parameter sets the gain applied when making a differential measurement of ADC2 relative to ADC0. Setting this parameter to 0x00 sets the gain to 1, 0x01 sets the gain to 2, 0x02 sets the gain to 4 and so on, up to 0x06 which sets the gain to 64. Note that the preamplifier output voltage saturates at 2.4 V regardless of the gain setting.

AdcDiffScaleFactorCh0/1 and AdcDiffOffsetCh0/1 - these parameters are applied to the raw ADC readings in differential mode. These values are not factory calibrated, but can be calibrated by the user.

FastAdcPrescaler - this parameter is the system clock divisor used to generate the ADC clock when the system is being clocked at 16 MHz. Default value is 0x05 (system clock ÷128). Higher values correspond to slower ADC clock rates. For example, 0x07 = ÷512, and 0x00 = ÷4. Note that larger prescalers will increase the amount of time it takes to collect all readings. DIV4 is not valid when running at 16 MHz because the maximum ADC clock rate is 2 MHz, so DIV8 is the lowest allowed.

SlowAdcPrescaler - System clock divisor used to generate the ADC clock when the system is being clocked at 2 MHz, when exiting sleep mode. Default value is 0x02 (system clock ÷16). Higher values correspond to slower ADC clock rates. For example, 0x07 = DIV512, and 0x00 = DIV4.

MaxQueuedEvents - this parameter sets the maximum number of Event Reports that can be queued at one time by a DNT90M.

AdcSkipCount - this parameter sets the number of measurements to skip (discard) when switching to a new ADC channel. The skipped measurements allow transients in the ADC sample-and-hold circuit to settle out. This parameter must be set to at least 0x03 when *AdcDiffMode* is selected. Note that the *IoPreDelay* parameter discussed above provides a delay to allow signals *external* to the DNT90M to settle following a wake up event, while *AdcSkipCount* skips measurements that may be distorted because the *internal* voltage on the ADC sample-and-hold has not settled.

7.4.8 Bank 0xFF - Special Functions

Bank	Location	Name	R/W	Size	Range	Default
0xFF	0x00	UcReset	*W*	1	0..2	N/A
0xFF	0x01	MemorySave	*W*	1	0xD0..0xD2	N/A
0xFF	0x04	DiagSerialRate	*R/W*	1	0..11	7 (38400 kbps)
0xFF	0x0C	ForceDiscover	*W*	3	0..255	0.1 s/count
0xFF	0x0E	DiagPortEn	*R/W*	1	0..1	0 (disabled)

Table 7.4.8.1

UcReset - writing a 0 to this parameter initiates a full reset, writing 1 to initiates a reset to the serial bootloader, or writing a 2 to initiates a reset to the OTA bootloader client.

MemorySave - writing 0xD0 to this parameter reloads default parameter values, writing 0xD1 saves the current parameter values to EEPROM, or writing 0xD2 saves current parameter values to the EEPROM and resets the module, and 0xD3 saves the default values to EEPROM and resets the module.

DiagSerialRate - this parameter sets the diagnostic port serial data rate as shown below:

Setting	***Serial rate***
0x00	1.2 kbps
0x01	2.4 kbps
0x02	4.8 kbps
0x03	9.6 kbps
0x04	14.4 kbps
0x05	19.2 kbps
0x06	28.8 kbps
0x07	38.4 kbps (default)
0x08	57.6 kbps
0x09	115.2 kbps
0x0A	230.4 kbps
0x0B	250.0 kbps

ForceDiscover - writing to this register forces a heartbeat reply. The timing of the reply is a random interval from 0 to the time implied by the value in the *ForceDiscover* register (0.1 second/count). A *ForceDiscover* register write is usually sent as a broadcast packet and is used to discover the DNT90M radios in range of the sender.

DiagPortEn - setting this parameter to 0x01 enables diagnostic port operation.

7.5 Protocol-formatted Message Examples

7.5.1 Data Message

In this example, the ASCII text *"Hello"* is sent from the base to a remote using the *TxData* command.
The MAC address of the remote is 0x123456. The protocol formatting for the host message is:

SOP	Length	PktType	Lo MAC	MAC	Hi MAC	"H"	"e"	"l"	"l"	"o"
0xFB	0x09	0x05	0x56	0x34	0x12	0x48	0x65	0x6C	0x6C	0x6F

There are 9 bytes following the length byte, so the length byte is set to 0x09. Note that the 0x123456 network address is entered in Little-Endian byte order, 56 34 12. When an ACK to this message is received from the remote, the base outputs a *TxDataReply* message to its host:

SOP	Length	PktType	Lo MAC	MAC	Hi MAC	Status	RSSI
0xFB	0x07	0x15	0x56	0x34	0x12	0x00	0xB0

The 0x00 *TxStatus* byte value indicates the ACK reception from the remote. The *RSSI* value of the received ACK is 0xB0, indicating a received signal strength of approximately -80 dBm.

The ASCII *"Hello"* message is output at the remote as a 0x26 *RxData* event. The address field contains the originator's address, 0x00 0x00 0x00, which is the base. The *RSSI* value of the received message is 0xB4, indicating a received signal strength of approximately -76 dBm. The data following the RSSI value is the *"Hello"* text.

SOP	Length	PktType	Lo MAC	MAC	Hi MAC	RSSI	"H"	"e"	"l"	"l"	"o"
0xFB	0x0A	0x26	0x00	0x00	0x00	0x35	0x48	0x65	0x6C	0x6C	0x6F

Note that if the remote was in transparent mode, only the *"Hello"* text would be output.

7.5.2 Configuration Messages

In this example, the remote with MAC address 0x123456 is configured by the base (MAC address 0x000000) to generate *RxEvent* messages every 10 seconds. To do this, the *IoReportInterval* in the remote is set to 10 seconds and the *periodic report timer* bit in the *IoReportTrigger* parameter is set ON. The *IoReportInterval* and the *IoReportTrigger* parameters are loaded using *SetRemoteRegister* commands. The command to set the *IoReportInterval* to 10 seconds is:

SOP	Length	PktType	Lo MAC	MAC	Hi MAC	Reg	Bank	Size	Lo Val	Val	Val	Hi Val
0xFB	0x0B	0x07	0x56	0x34	0x12	0x1C	0x06	0x04	0x10	0x27	0x00	0x00

The *IoReportInterval* parameter starts in location 0x1C of Bank 6. The report interval scaling is 1 ms/count, so a 10 second report interval is 10,000 units or 0x00002710 (Little-Endian format 10 27 00 00). The *IoReportInterval* parameter is updated and *SetRemoteRegisterReply* is returned:

SOP	Length	PktType	Status	Lo MAC	MAC	Hi MAC	RSSI
0xFB	0x06	0x17	0x00	0x00	0x00	0x00	0xB2

The command to set the *periodic report timer* bit in *IoReportTrigger* is:

SOP	Length	PktType	Lo Mac	MAC	Hi MAC	Reg	Bank	Size	Val
0xFB	0x08	0x07	0x56	0x34	0x12	0x1B	0x06	0x01	0x10

The *IoReportTrigger* parameter is in location 0x1B of Bank 6. The *periodic report timer* bit in *IoReportTrigger* is located in bit position four (00010000b) or 0x10. The *IoReportTrigger* parameter is updated and *SetRemoteRegisterReply* is returned:

SOP	Length	PktType	Status	Lo MAC	MAC	Hi MAC	RSSI
0xFB	0x06	0x17	0x00	0x00	0x00	0x00	0xB4

7.5.3 Sensor Message

In this example, the base host requests an ADC1 reading from a remote using the *GetRemoteRegister* command, type 0x06. The MAC address of the remote is 0x123456. The current ADC1 measurement parameter is read starting at register location 0x15 and Bank 5. The ADC reading spans two bytes. The protocol formatting for this command is:

SOP	Length	PktType	Lo Mac	MAC	Hi MAC	Reg	Bank	Size
0xFB	0x07	0x06	0x56	0x34	0x12	0x15	0x05	0x02

Note the remote MAC address is entered in Little-Endian byte order, 56 34 12.

The ADC reading is returned in a *GetRemoteRegisterReply* message:

SOP	Length	PktType	Status	Lo MAC	MAC	Hi MAC	RSSI	Reg	Bank	Size	Lo Val	Hi Val
0xFB	0x0B	0x16	0x00	0x00	0x00	0x00	0xB7	0x1C	0x06	0x02	0x7B	0x08

Substantial information is returned in the message. The last two byes of the message give the ADC reading in Little-Endian format, 7B 08. The ADC reading is thus 0x087B (2171). The RSSI value is the byte following the address, 0xB7 (-73 dBm). The *TxStatus* byte to the right of the *GetRemoteRegisterReply* Packet Type is 0x00, showing the packet was acknowledged on the RF channel.

7.5.4 Event Message

The configuration example shown in Section 7.5.2 above causes the remote with MAC address 0x123456 to start sending event messages every 10 seconds as shown in the log below:

```
FB 12 28 56 34 12 B8 00 7A 01 36 01 FF 01 10 00 20 01 40 01
FB 12 28 56 34 12 B0 00 79 01 35 01 C0 01 10 00 20 01 40 01
FB 12 28 56 34 12 A9 00 72 01 35 01 D3 01 10 00 20 01 40 01
FB 12 28 56 34 12 AC 00 75 01 36 01 E7 01 10 00 20 01 40 01
```

The first received message in the above log is constructed as follows:

SOP	Length	PktType	Addr	Addr	Addr	RSSI	Data
0xFB	0x12	0x28	0x56	0x34	0x12	B8	

GPIO	ADC0		ADC1		ADC2		Event Flags		DAC0		DAC1	
0x00	0x7A	0x01	0x36	0x01	FF	0x01	0x10	0x00	0x20	0x01	0x40	0x01

RxEvent messages are *PktType* 0x28. The message payload consists of the states of GPIO0 through GPIO5, the input voltages measured by ADC0 through ADC2, the event trigger(s), and the DAC output settings. Note the ADC readings, event flags and DAC settings are presented in Little-Endian order. The remote is assumed to be always ON in this example. If the remote is placed in periodic sleep mode (*SleepMode* = 1), a suitable value of the *WakeResponseTime* parameter should be set to allow the base application to analyze the I/O report and send back a command to the remote as needed.

8.0 DNT90MDK Developer's Kit

Figure 8.0.1 shows the main contents of a DNT90MDK Developer's kit:

Figure 8.0.1

8.1 DNT90MDK Kit Contents

- Two DNT90MP radios installed in DNT90M interface boards (labeled Base and Remote)
- Two 9 V wall-plug power suppliers, 120/240 VAC, plus two 9 V batteries (not show above)
- Two RJ-45/DB-9F cable assemblies and two A/B USB cables
- Two installed U.FL coaxial jumper cables
- Two 2 dBi dipole antennas with two MMCX-to-RSMA coaxial adaptor cables
- One DNT90MDK documentation and software CD

8.2 Additional Items Needed

To operate the kit, the following additional item is needed:

- One PC with Microsoft Windows XP, Vista or Windows 7 operating system. The PC must be equipped with a USB port or a serial port capable of operation at 9600 bps.

8.3 Developer's Kit Default Operating Configuration

The default operating configuration of the DNT90MDK developer's kit is point-to-point with transparent serial data at 9600 bps, 8N1. One DNT90MP is preconfigured as a base and the other as a remote. Labels on the bottom of the interface boards specify Base or Remote. The defaults can be overridden to test other operating configurations using the DNT90M Demo utility discussed in Section 8.5.

8.4 Developer's Kit Hardware Assembly

Observe ESD precautions when handling the kit circuit boards. When shipped, the DNT90MP radios and U.FL coax jumper cables are installed in the interface boards, as shown in Figure 8.4.1. If a DNT90MP radio and/or the U.FL jumper cable have been unplugged after receipt, confirm the DNT90MP is correctly plugged into its interface board with the radio oriented so that its U.FL connector is next to the U.FL connector on the interface board, as shown in Figure 8.4.2. Also check the radio's alignment in the socket on the interface board. No pins should be hanging out over the ends of the connector. Next, screw each dipole antenna into and adaptor cable and "snap" the other end of the adaptor cable into the MMCX RF connector on the development board as shown in Figure 8.4.2.

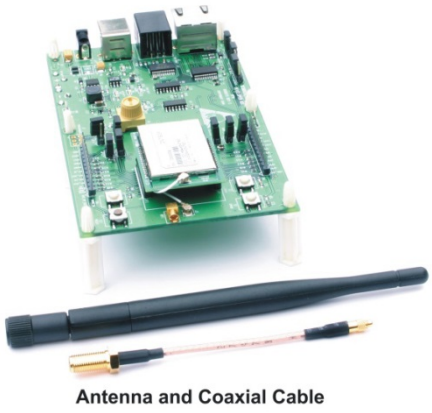


Figure 8.4.1

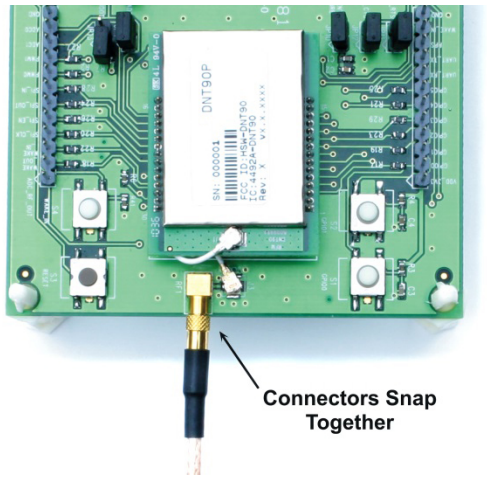


Figure 8.4.2

As shown in Figure 8.4.3, confirm there is a jumper on J10 (this jumper can be removed later and a current meter connected across J10 to measure just the DNT90M's current consumption during operation).

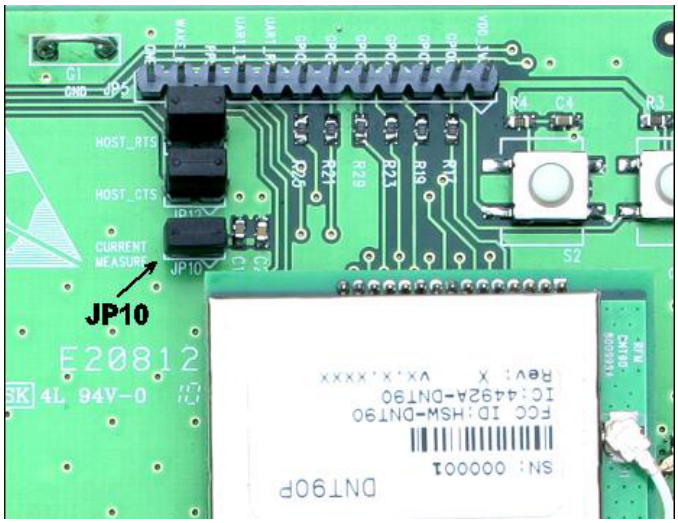


Figure 8.4.3

There are three serial connectors and a power connector on the end of each interface board, as shown in Figure 8.4.4. The RJ-45 connector provides an RS232 interface to the DNT90MP's main serial port. The USB connector provides an optional interface to the radio's main serial port. The RJ-11 connector provides an RS232 interface to the radio's optional diagnostic port. The DNT90M Demo utility program runs on the radio's main port.

Figure 8.4.4

Many desktop PCs have a built-in serial port capable of operation at 9600 bps. The kit can be run satisfactorily at the 9600 bps data rate, but not at its fastest throughput. Use the RJ-45 to DB-9F cable assemblies for serial port operation.

Optionally, the kit development boards can be run from USB ports. Plugging in the USB cable automatically switches operation from the RJ-45 connector. The USB interface is based on an FT232RL serial-to-USB converter IC manufactured by FTDI. The FT232RL driver files are located in the i386 and AMD64 folders on the kit CD, and the latest version of the drivers can downloaded from the FTDI website, www.ftdichip.com. The drivers create a virtual COM port on the PC. Power the Base using one of the supplied wall-plug power supplies. Next connect the Base to the PC with a USB cable. The PC will find the new USB hardware and open a driver installation dialog box. Enter the letter of the drive holding the kit CD and click *Continue*. The installation dialog will run *twice* to complete the FT232R driver installation.

8.5 DNT90M Utility Program

The DNT90M utility program requires only one PC for initial kit operation and sensor applications (ADC, DAC and digital I/O). Two serial/USB ports are required for bidirectional serial communications. Section 8.6 below covers using the DNT90M Demo utility program for initial kit operation and familiarization. Section 8.6.1 covers serial message communication and radio configuration.

8.6 Initial Kit Operation

Create a file folder on the PC and copy the contents of the kit CD into the folder.

Connect the Base to the PC and power up the Base using a wall-plug power supply.

The DNT90M Demo utility program is located in the *PC Programs* folder. The DNT90M Demo utility program requires no installation and can be simply copied to the PC and run. Start the utility program on the PC. The start-up window is shown in Figure 8.6.1.

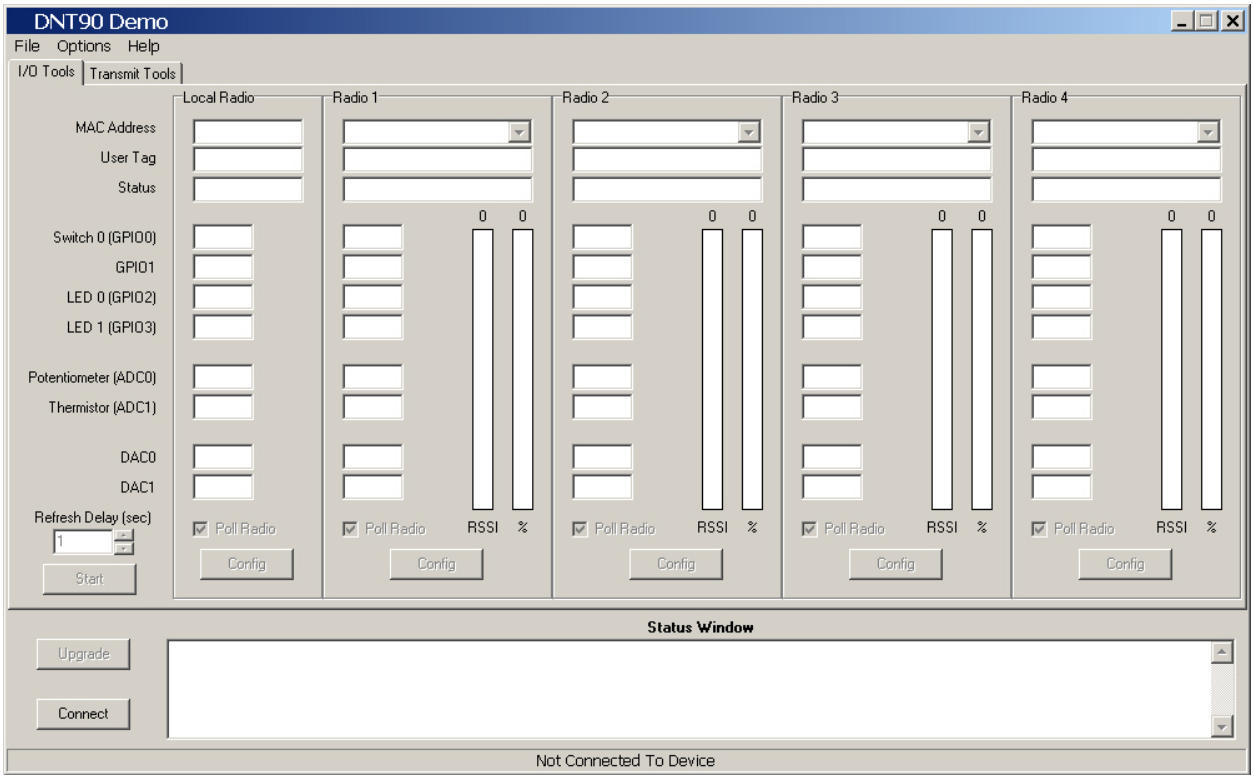


Figure 8.6.1

Click on *Connect* to open the *Select Comm Port Settings* dialog box, as shown in Figure 8.6.2. If necessary, set the baud rate to 9600 bps. Set the *CommPort* to match the serial port connected to the Base, either the hardware port or the USB virtual serial port. Then click *OK* to activate the serial connection.

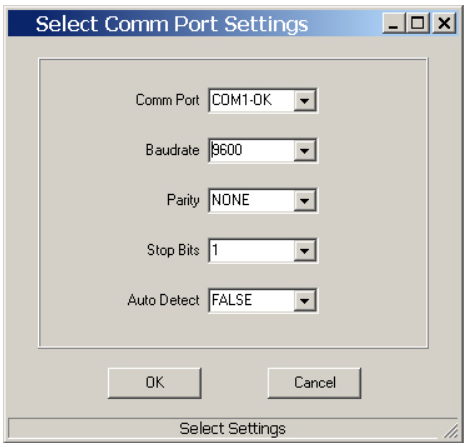


Figure 8.6.2

At this point the utility program will collect data from the Base, filling in the *Local Radio* column as shown in Figure 8.6.3. Next, power-up the Remote using a wall-plug power supply. The Remote will transmit a "heartbeat" message on power up as shown in the *Status Window*. Click on the drop-down box at the top of the *Radio 1* column and load the *MAC Address* for the Remote from the heartbeat message. Next press the *Start* button using the default 1 second *Refresh Delay*.

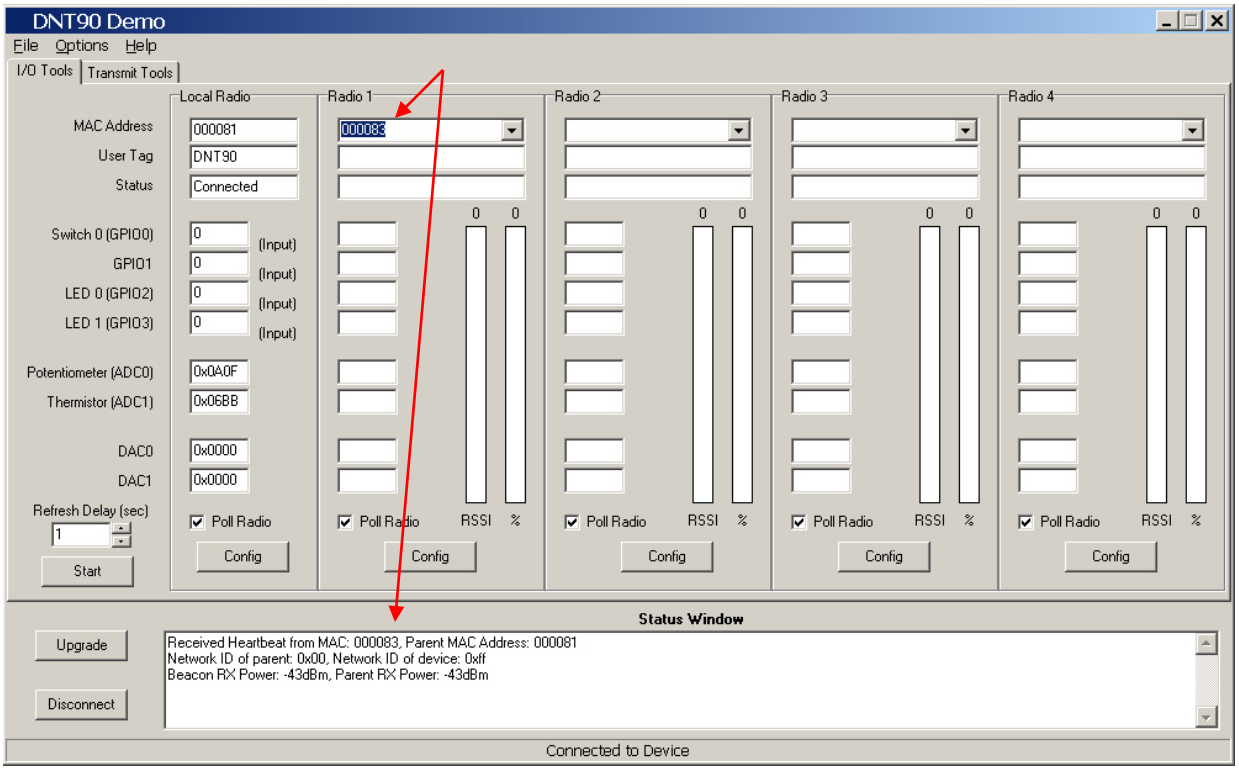


Figure 8.6.3

The utility program will display updated data on the Remote in the *Radio 1* column, including bar graphs of *RSSI* signal strength in dBm and percent packet success rate, as shown in Figure 8.6.4. Adjusting the large pot on the Remote can be observed on the *Potentiometer (ADC0)* row.

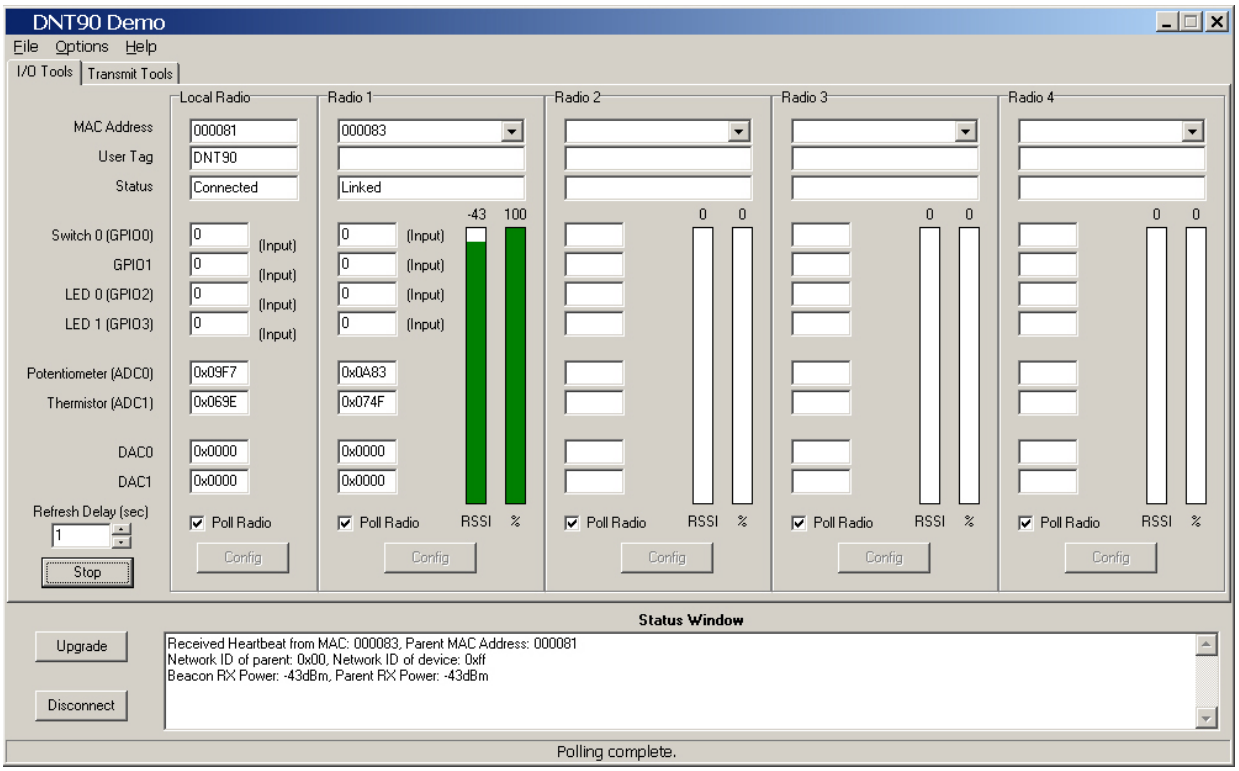


Figure 8.6.4

Note: If the Remote is powered up before the DNT90M Demo program is running and connected to the Base, the initial Remote heartbeat will be missed and it will be necessary to manually enter the Remote's MAC address in the MAC Address field under Radio 1 and then press the Enter key to display the Remote information.

If any difficulty is encountered in setting up the DNT90MDK development kit, contact Murata's module technical support group. The phone number is +1.678.684.2004. Phone support is available from 8:30 AM to 5:30 PM US Eastern Time Zone, Monday through Friday. The E-mail address is tech_sup@murata.com.

8.6.1 Serial Communication and Radio Configuration

Connect PCs to both the Base and the Remote for serial communication testing (alternately one PC can be used with two serial ports and two instances of the DNT90M Demo program running). Click the *Stop* button under the *Refresh Delay* label on the *I/O Tools* tab and move to the *Transmit Tools* tab, as shown in Figure 8.6.1.1.

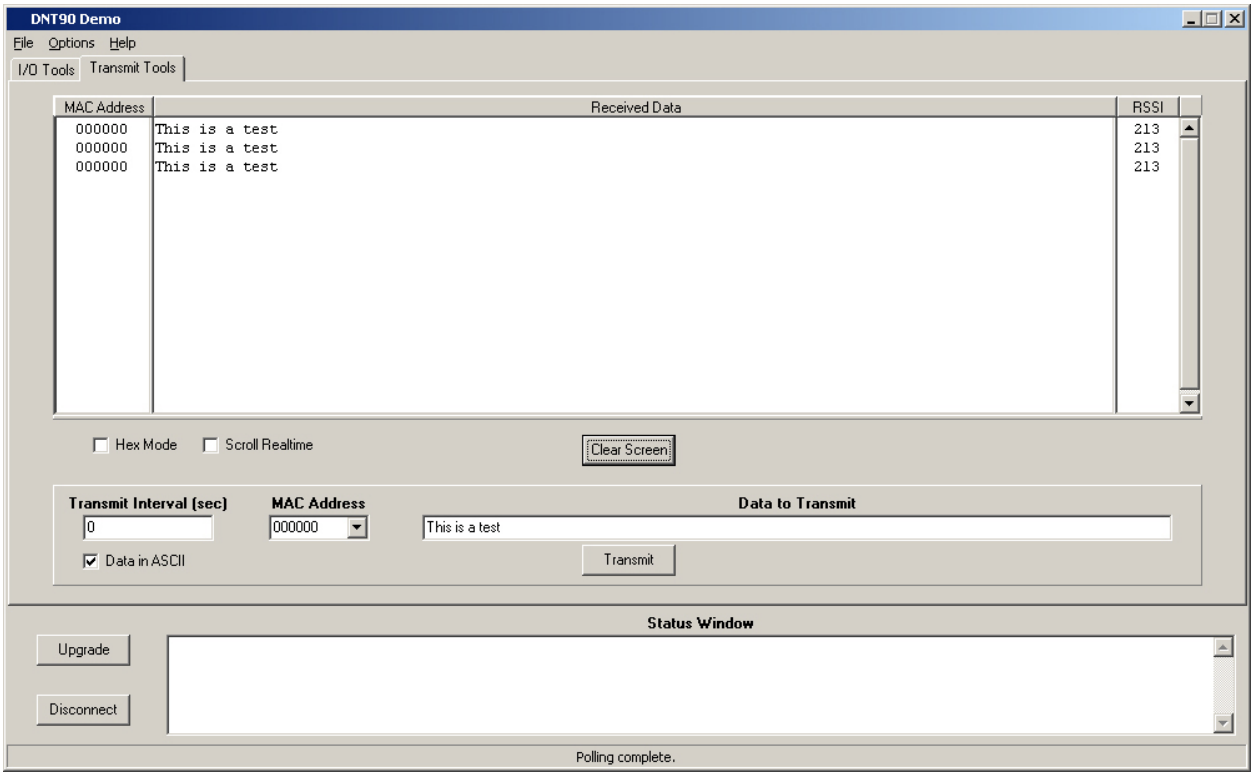


Figure 8.6.1.1

Pressing the *Transmit* button on this screen sends the message in the *Data to Transmit* text box to the selected *MAC Address*. Note that the *MAC address* a remote uses for the base is 0x000000. Data sent to the local radio is displayed in the *Received Data* text box. Received data can be displayed as ASCII (default) or in Hexadecimal format by checking the *Hex Mode* check box. When the *Transmit Interval* is set to zero, *Data to Transmit* is sent once when the *Transmit* button is clicked. When the *Transmit Interval* is set to a positive number, Pressing the *Transmit* button once will cause a transmission each transmit interval (seconds) until the button is pressed again.

Returning to the *I/O Tools* tab, the multi-tab *Configuration* window for each radio can be accessed by clicking on its *Config* button. The data presented on the first six tabs corresponds to configuration register Banks 0 through 5 as discussed in Section 4.2 above, with the data on the next two tabs corresponding to configuration register Bank 6.

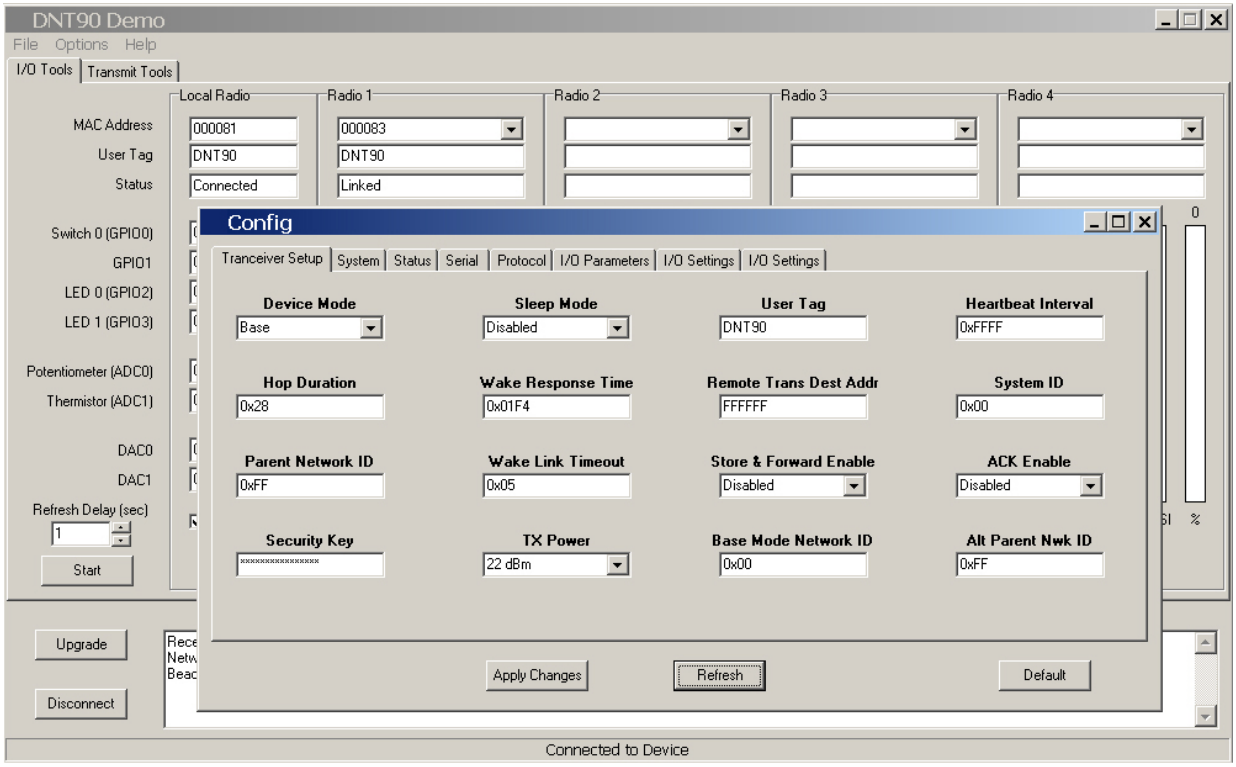


Figure 8.6.1.2

The *Transceiver Setup* Tab is shown in Figure 8.6.1.2 and corresponds to Bank 0. The current values of each Bank 0 parameter are displayed and can be updated by selecting from the drop-down menus or entering data from the keyboard, and then pressing the *Apply Changes* button. Note that data is *displayed and entered in Big-Endian order*. The utility program automatically reorders multi-byte data to and from Little-Endian order when building or interpreting messages.

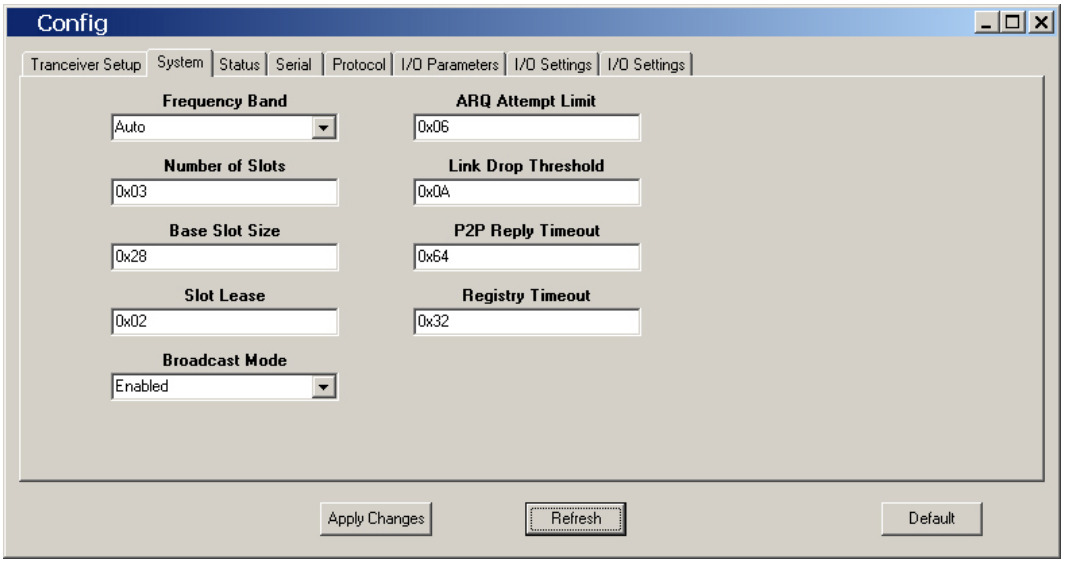


Figure 8.6.1.3

Figure 8.6.1.3 shows the *System* tab contents, corresponding to Bank 1. The current values of each parameter are displayed and can be updated by selecting from the drop-down menu or entering data from the keyboard, and then pressing the *Apply Changes* button. Note that Bank 1 holds configuration parameters for the base only except for *Broadcast Mode,* which applies to both the base and the remotes.

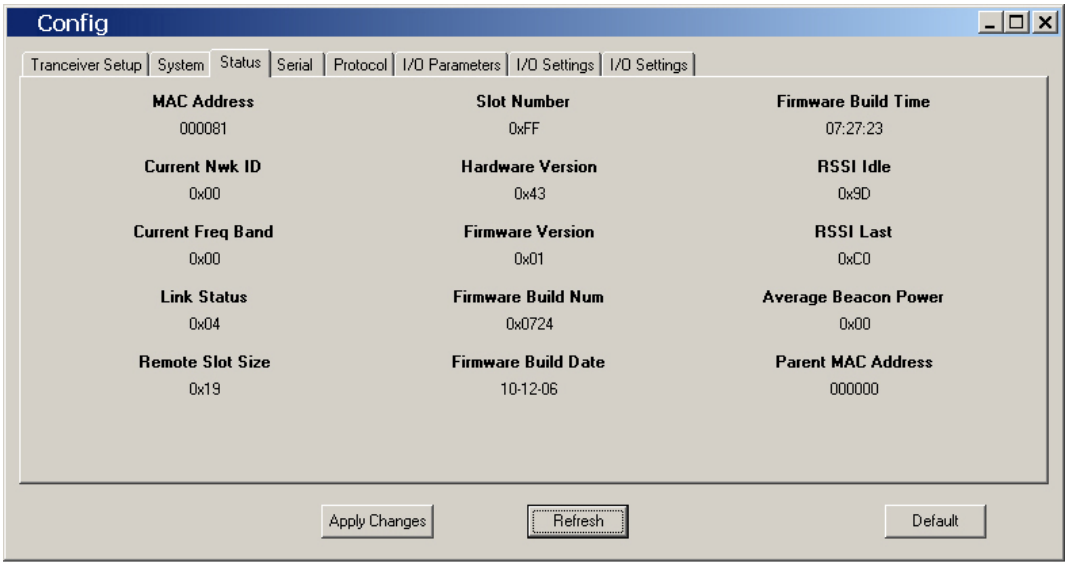


Figure 8.6.1.4

Figure 8.6.1.5 shows the *Status* tab contents, corresponding to Bank 2. Note the *Status* tab contains read-only parameters.

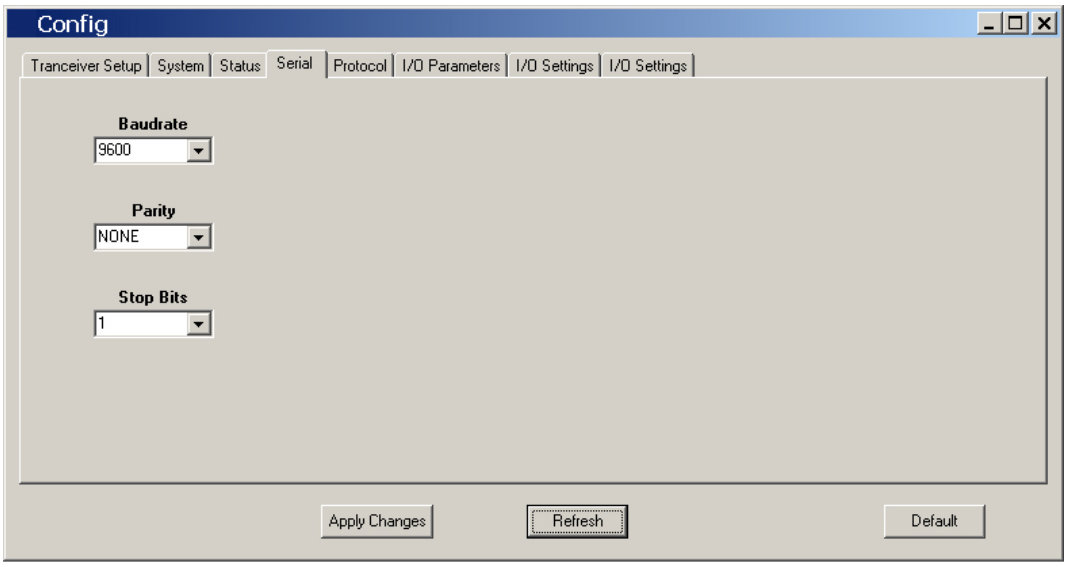


Figure 8.6.1.5

Figure 8.6.1.5 shows the *Serial* tab contents corresponding to the serial parameters in Bank 3. The values shown are the defaults for serial port operation.

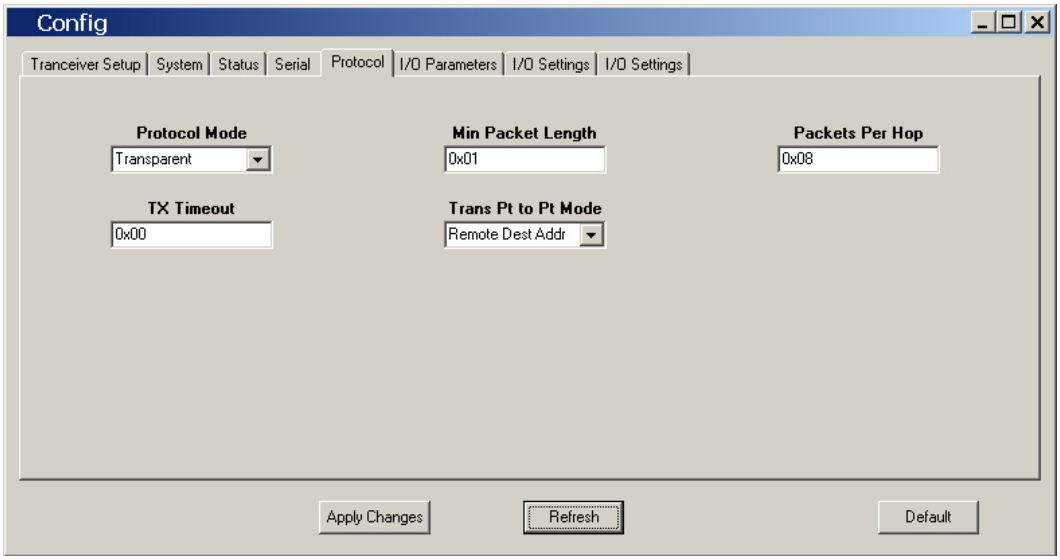


Figure 8.6.1.6

Figure 8.6.1.6 shows the *Protocol* tab contents, corresponding to Bank 4. Transparent serial data communication is currently chosen.

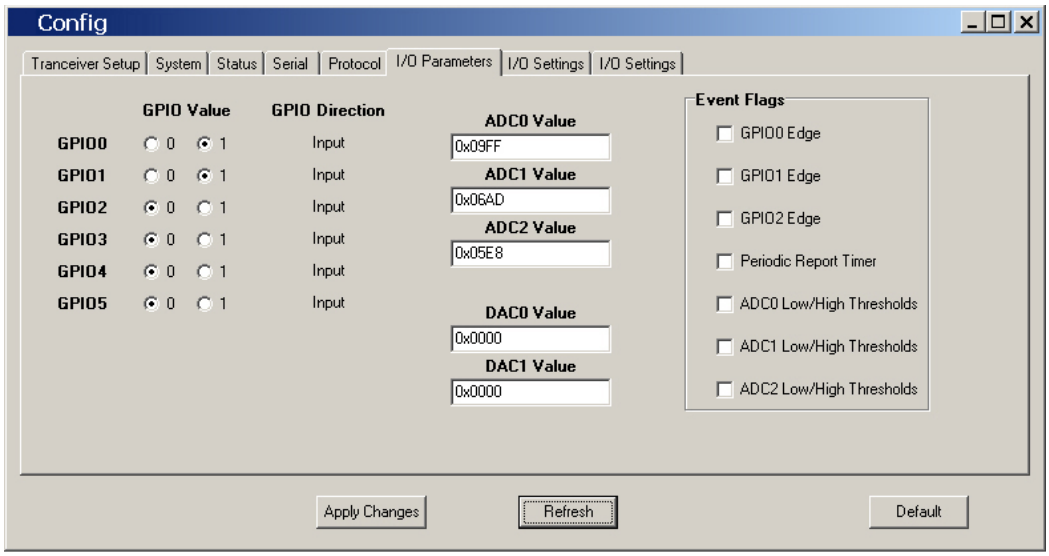


Figure 8.6.1.7

Figure 8.6.1.7 shows the *I/O Parameters* tab contents, corresponding to Bank 5. All GPIO ports are configured as inputs. The 12-bit ADC input readings and DAC output settings are given in *Big-Endian* byte order. Event flags are presented on the right side of the window.

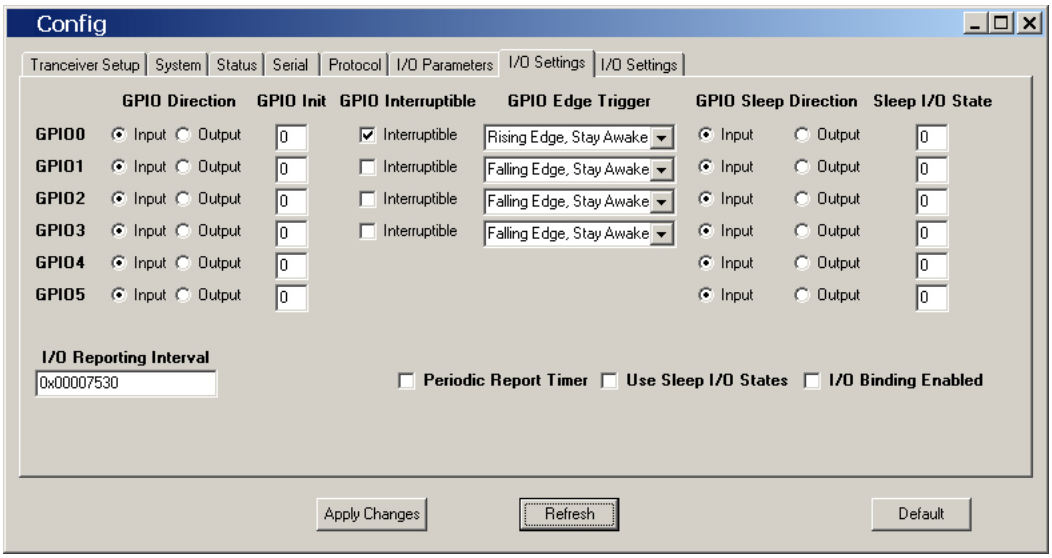


Figure 8.6.1.8

Figure 8.6.1.8 shows the first *I/O Settings* tab contents, corresponding to Bank 6 GPIO configurations other than alternate GPIO functions. This tab allows the direction of the GPIO ports to be set both for active and sleep modes, and in the case of GPIO outputs, the initial power up states and sleep mode states to be set. When GPIO ports 0 - 3 are configured as inputs, event interrupts can be set for them with check boxes. The type of interrupt trigger is selected from the drop-down boxes to the right of the check boxes. Periodic I/O reporting, reporting interval and enable/disable sleep I/O states and I/O binding can also be configured under this tab.

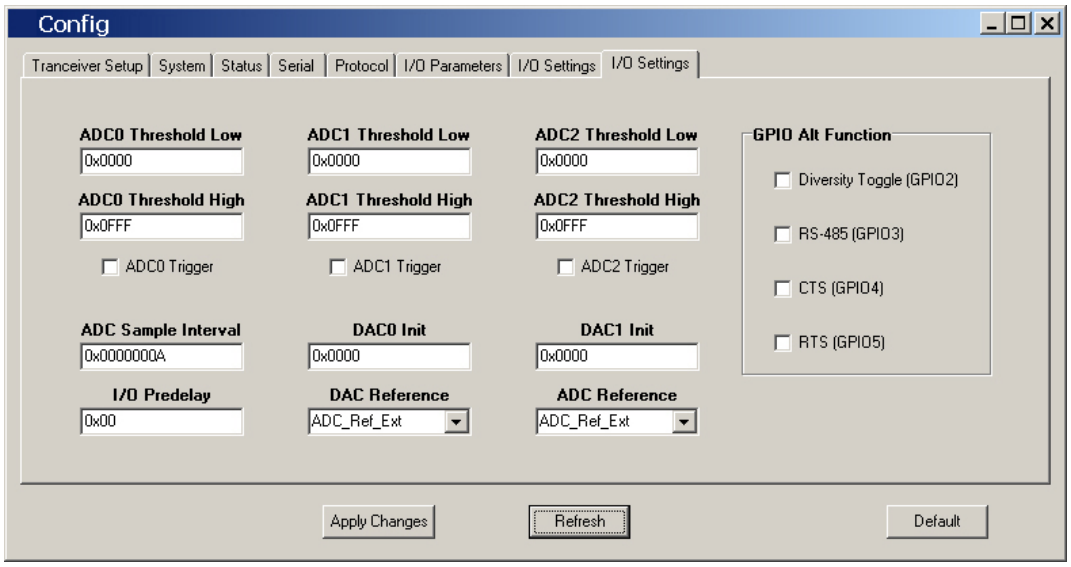


Figure 8.6.1.9

Figure 8.6.1.9 shows the second *I/O Setup* tab contents, corresponding to Bank 6 ADC input and DAC output parameters. The ADC and DAC reference voltages, the ADC sampling interval, the high and low ADC thresholds for event reporting and event reporting triggers on each ADC channel can be set, along with the initial output values for each DAC channel. The event reporting I/O predelay and alternate GPIO functions can also be set from this tab.

The DNT90M Demo Utility *File*, *Options* and *Help* menus are shown in Figure 8.7.8.

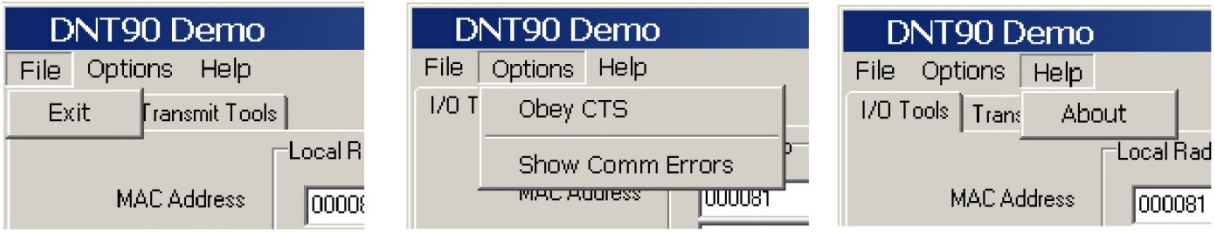


Figure 8.7.8

8.7 DNT90M Interface Board Features

The locations of the LEDs on the interface board that are used by the DNT90M are shown in Figure 8.8.1.

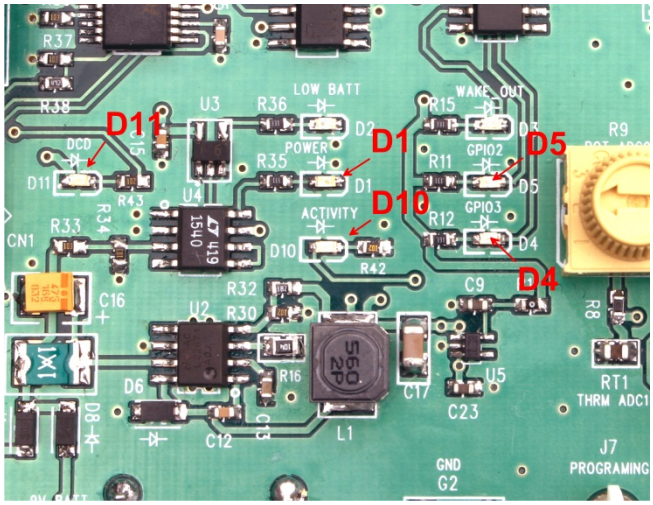


Figure 8.8.1

DCD LED, D11, illuminates when a DNT90M receives a valid packet. Activity LED, D10, illuminates when transmitting RF data. Power LED, D1, illuminates when the DNT90M and its interface board are powered. GPIO2 LED, D5, and GPIO3 LED, D4, can be controlled by configuring GPIO2 and GPIO3 as outputs on the DNT90M. These LEDs are illuminated with a logic high signal.

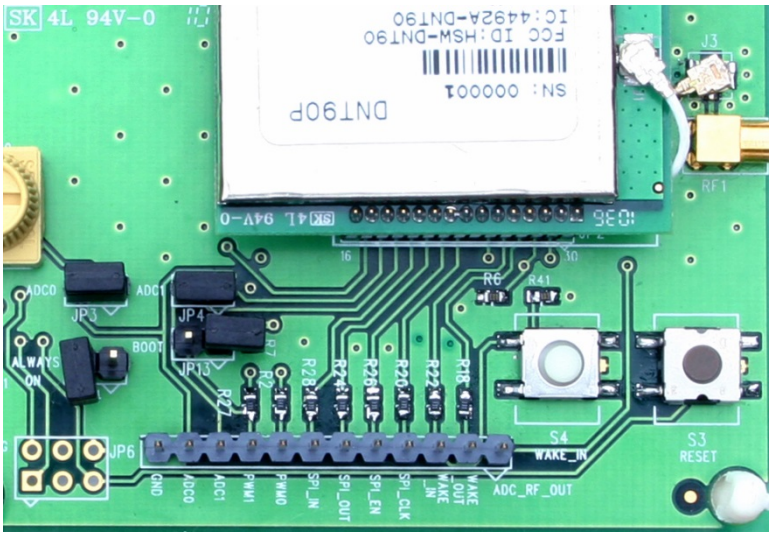


Figure 8.8.2

Figure 8.8.2 shows the connectors and switches to the right of the DNT90MP mounting socket. JP3 and JP4 normally have shorting plugs installed as shown in Figure 8.8.2. JP3 connects ADC0 to the yellow potentiometer. Clockwise rotation of the potentiometer increases the voltage. JP4 connects ADC1 to a thermistor temperature sensor. The DNT90M has its own boot loader utility that allows the protocol firmware to be installed with a terminal program that supports YMODEM. The boot loader is activated with a shorting plug on JP13. J6 provides access to various DNT90M pins as shown on the silkscreen. Pressing switch SW3 will reset the DNT90MP. Switch S4 is not used with the DNT90M.

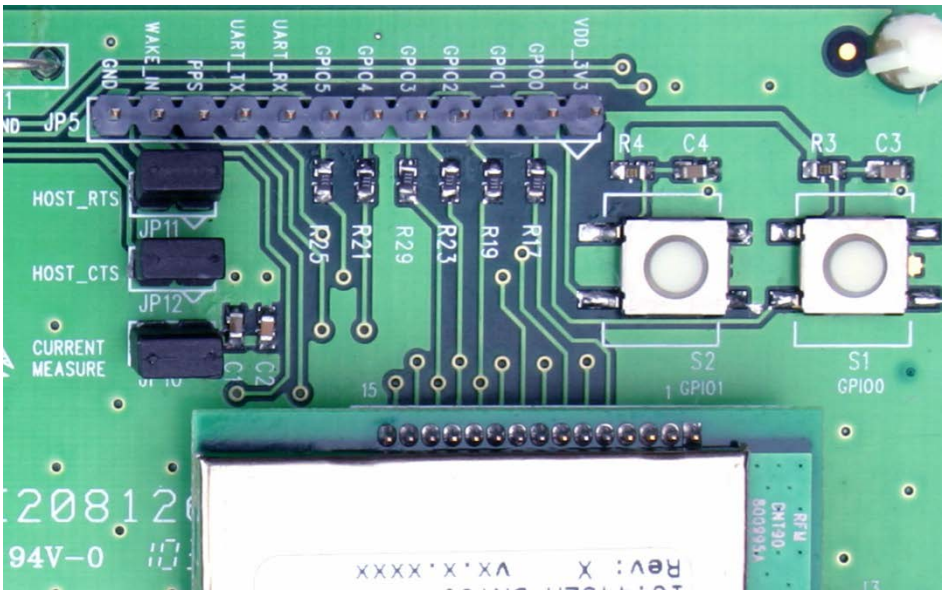


Figure 8.8.3

Figure 8.8.3 shows the connectors to the left of the DNT90MP mounting socket. Pressing switch SW1 switches GPIO0 from logic high to low, and pressing SW2 switches GPIO1 from logic high to low. The DNT90MP interface board includes a 5 V regulator to regulate the input from the 9 V wall-plug power supply. Do not attempt to use the 9 V wall-plug power supply to power the DNT90MP directly. The maximum allowed voltage input to the DNT90MP is 5.5 V.

9.0 Troubleshooting

DNT90M not responding - make sure /RESET is not asserted (logic low). Make sure the host serial port settings match the DNT90M serial port settings.

Can not enter protocol mode - make sure the host data rate is correct. The DNT90M defaults to 9.6 kbps. If using the *EnterProtocolMode* command, send the complete protocol format for this command.

Carrier is detected, but no data appears to be received - if /HOST_RTS is enabled, make sure it is asserted (logic low) to enable character flow from the DNT90M.

Range is extremely limited - this is usually a sign of a poor antenna connection or the wrong antenna. Check that the antenna is firmly connected. If possible, remove any obstructions near the antenna.

9.1 Diagnostic Port Commands

The diagnostic port shares its RX and TX signal lines with the Activity and DCD indications, respectively. Consequently, the debug port feature must be enabled before being used (Bank 0xFF). The change must be saved and the module then needs to be reset for this to take effect. The diagnostic port is defaulted to 38.4 kbps, 8N1.

The diagnostic port supports the following user commands:

**rbr <bank> <reg> ** - read a parameter register's value from the module.

rbw <bank> <reg> <value> [<value> <value>] - write a parameter register's value with a span of up to 3 bytes

stat <option> - option = 0 is off, option = 1 displays DataTx/AckRx for a hop sequence in time order, and option = 2 displays any packet RX or packet error for a hop sequence in frequency order.

10.0 Appendices

10.1 Ordering Information

DNT90MC: Direct peer-to-peer transceiver for solder-pad mounting, for use with external antenna
DNT90MP: Direct peer-to-peer transceiver for pin-socket mounting, for use with external antenna
DNT90MCA: Direct peer-to-peer transceiver for solder-pad mounting, includes on-board chip antenna
DNT90MPA: Direct peer-to-peer transceiver for pin-socket mounting, includes on-board chip antenna

10.2 Technical Support

For DNT90M technical support call Murata at 678.684.2004 between the hours of 8:30 AM and 5:30 PM Eastern Time

10.3 DNT90M Mechanical Specifications

DNT90C Outline and Mounting Dimensions

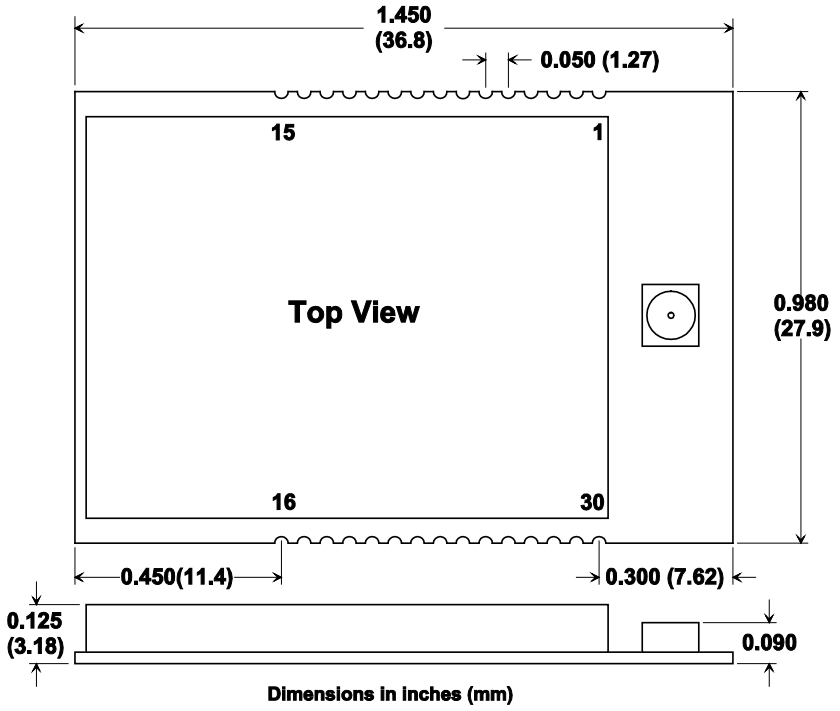


Figure 10.3.1

DNT90C Solder Pad Dimensions

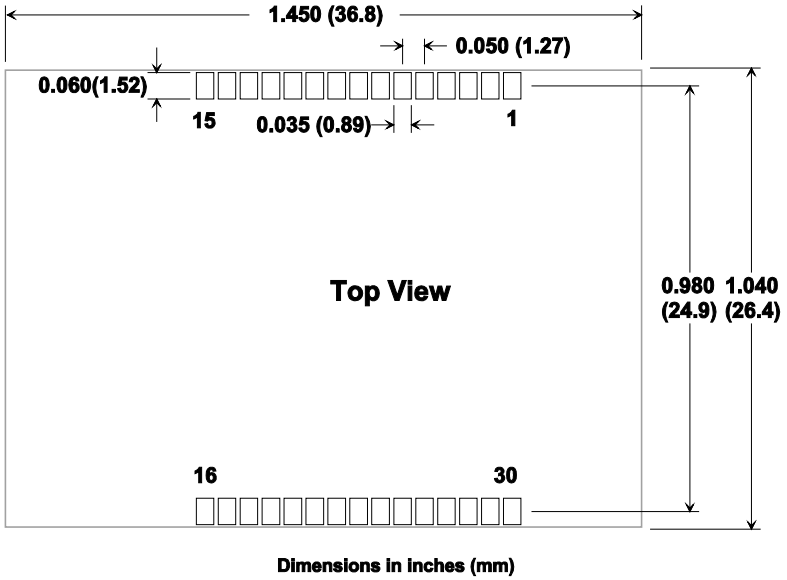


Figure 10.3.2

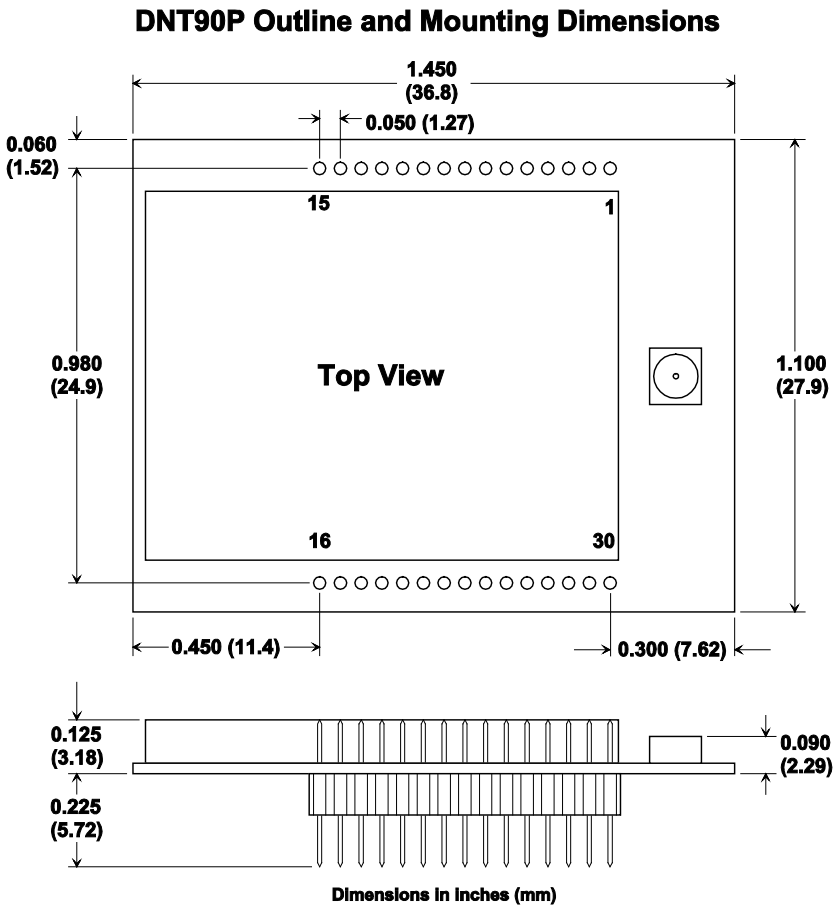


Figure 10.3.3

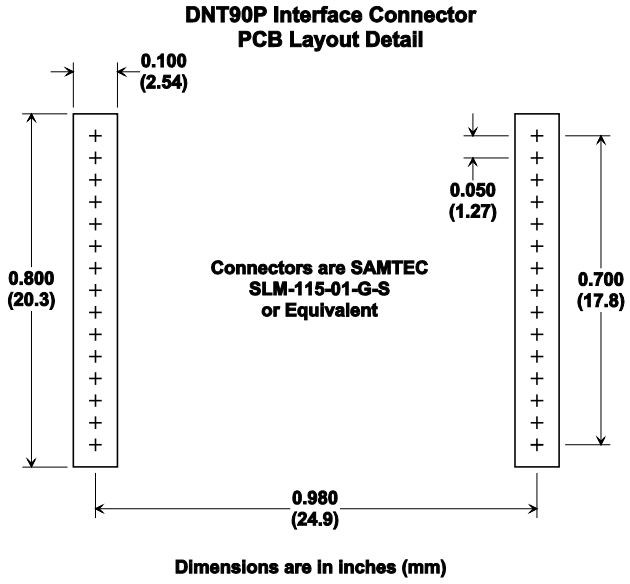


Figure 10.3.4

DNT90CA Outline and Mounting Dimensions

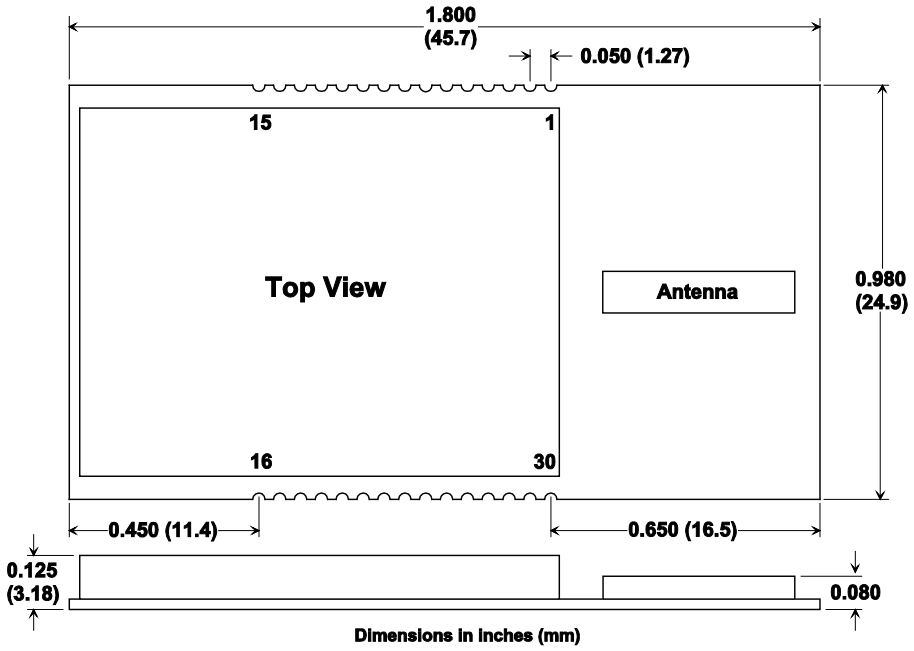


Figure 10.3.5

DNT90CA Solder Pad Dimensions

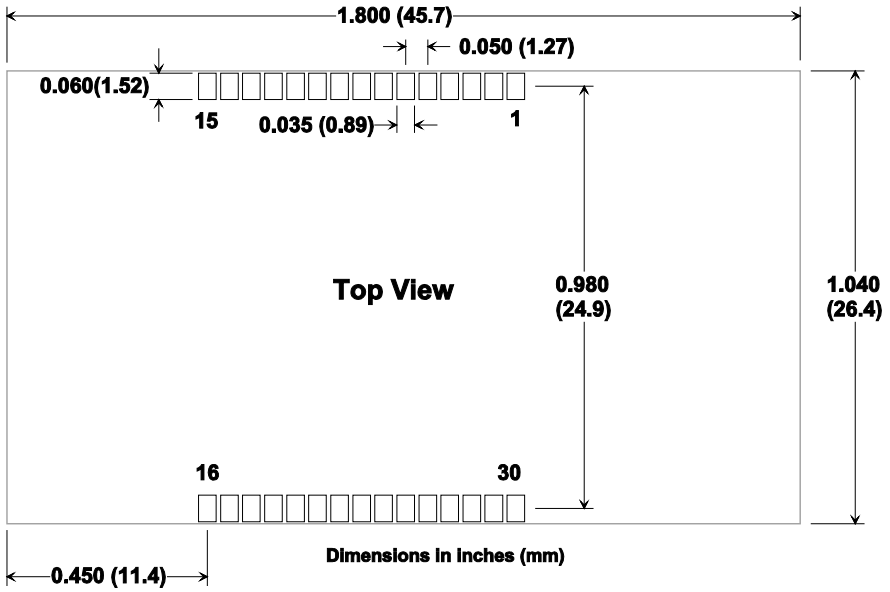


Figure 10.3.6

DNT90PA Outline and Mounting Dimensions

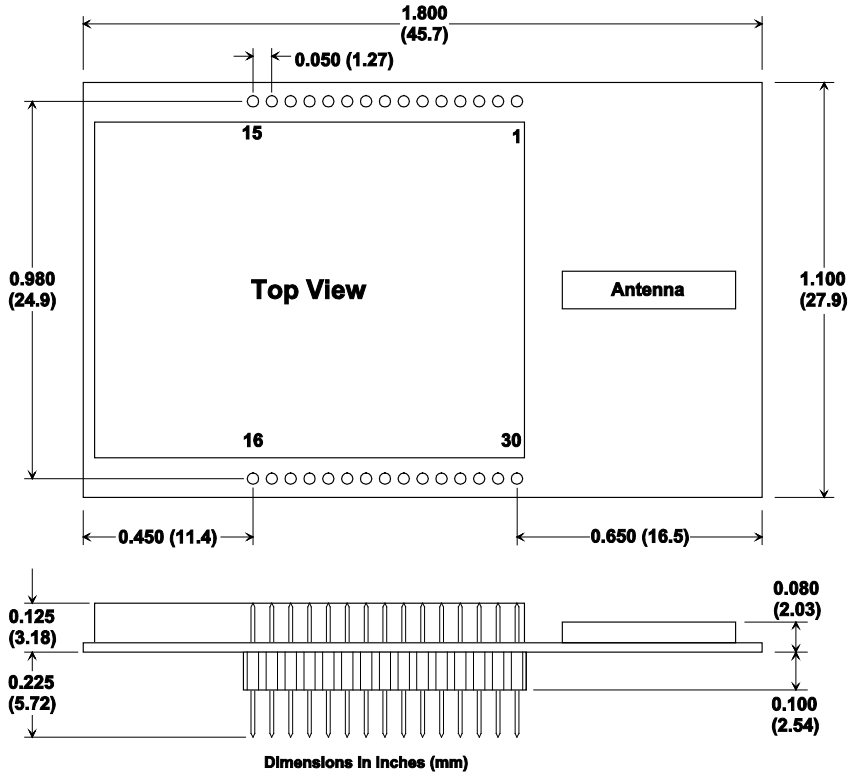


Figure 10.3.7

DNT90PA Interface Connector
PCB Layout Detail

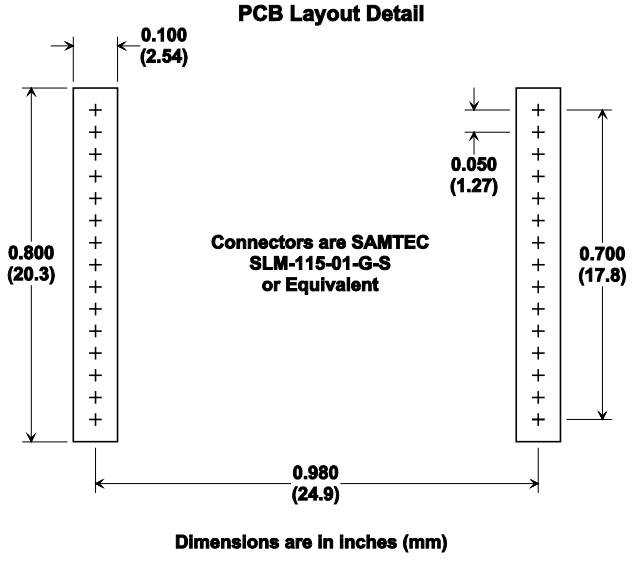
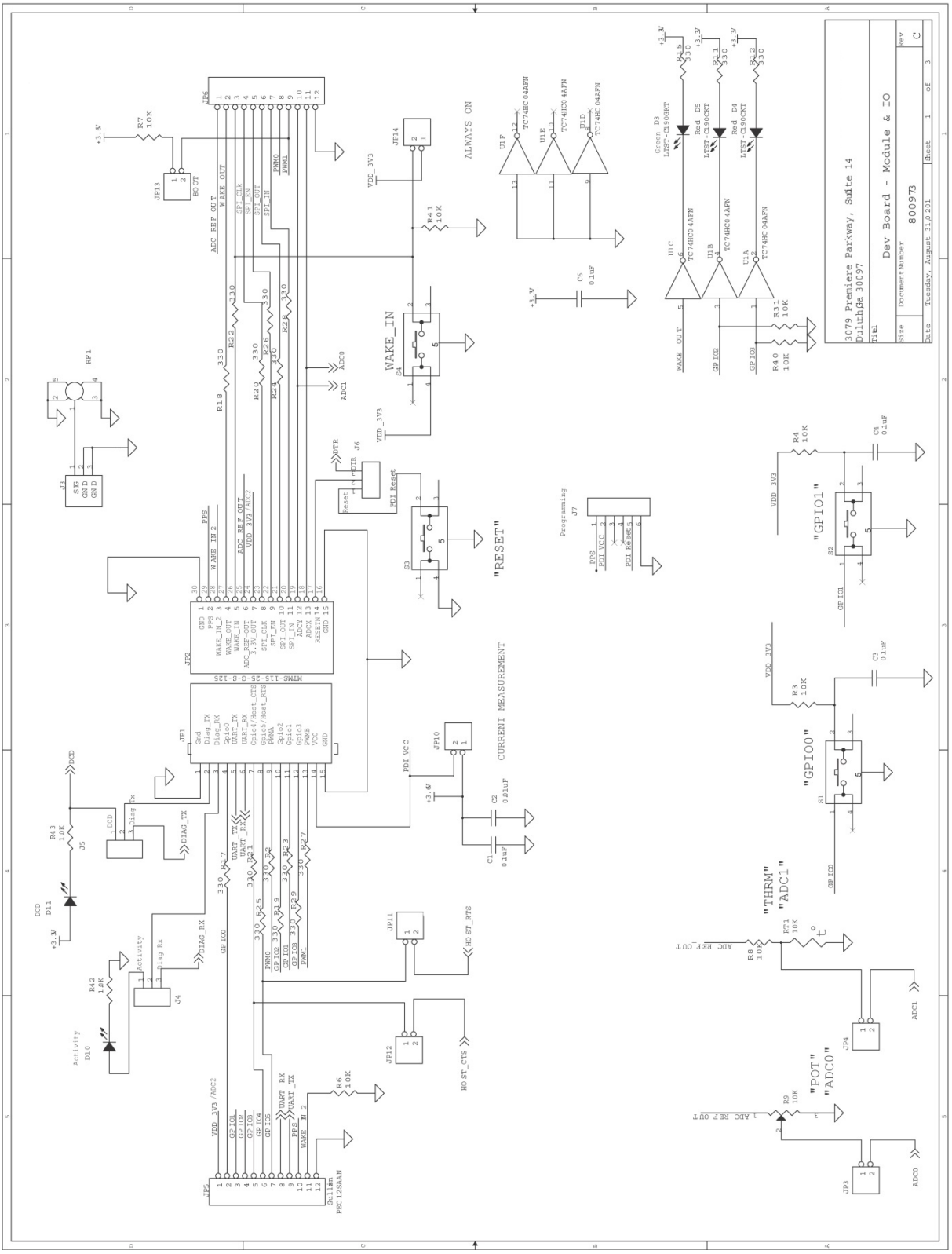
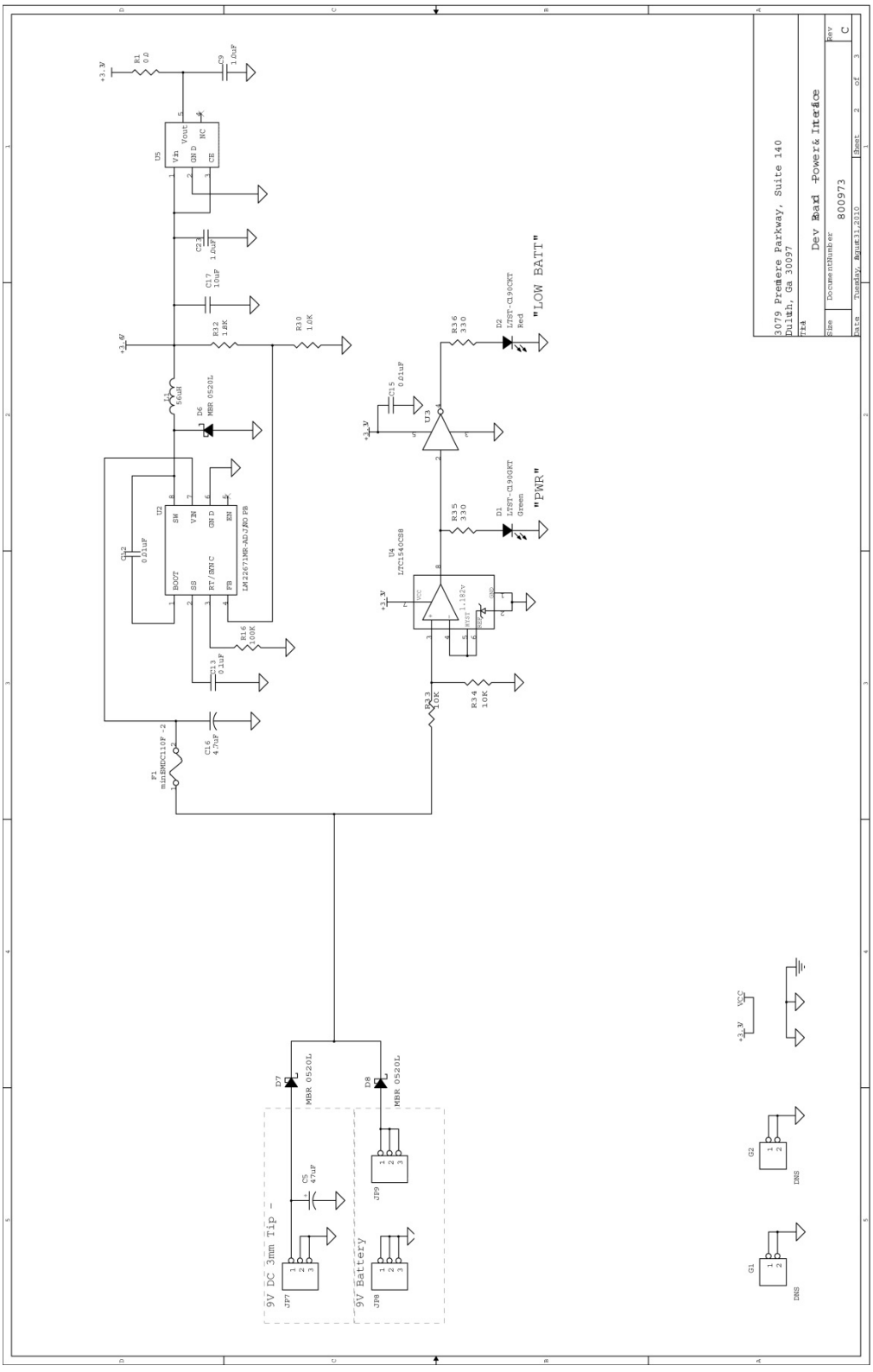
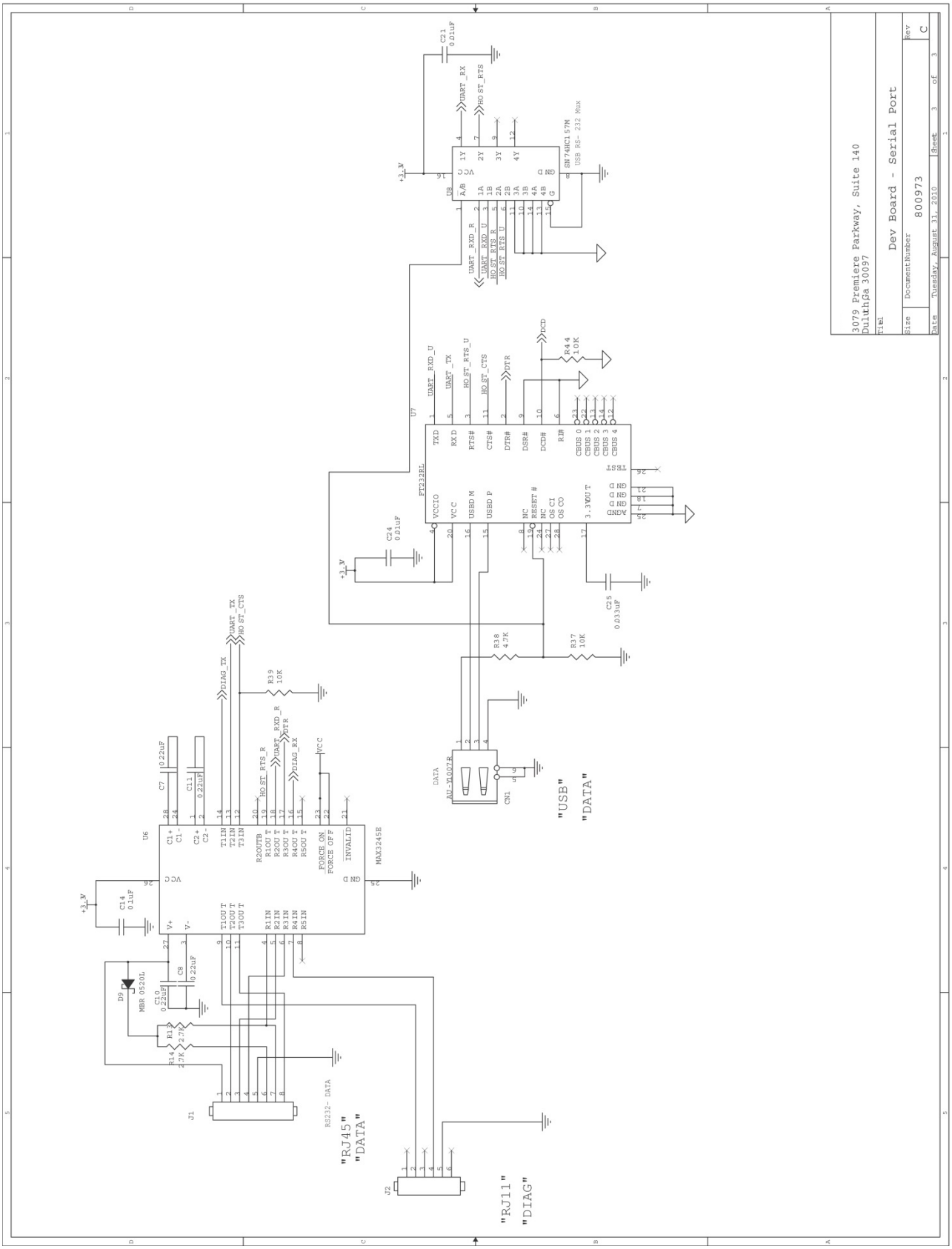


Figure 10.3.8

10.4 DNT90M Development Board Schematic




3079 Premiere Parkway, Suite 140
Duluth, Ga 30097
Dev Bad +Power& Interface
800973
"LOW BATT"
"PWR"
9v DC 3mm Tip -
9v Battery
+3.3V
VCC



11.0 Warranty

Seller warrants solely to Buyer that the goods delivered hereunder shall be free from defects in materials and workmanship, when given normal, proper and intended usage, for twelve (12) months from the date of delivery to Buyer. Seller agrees to repair or replace at its option and without cost to Buyer all defective goods sold hereunder, provided that Buyer has given Seller written notice of such warranty claim within such warranty period. All goods returned to Seller for repair or replacement must be sent freight prepaid to Seller's plant, provided that Buyer first obtain from Seller a Return Goods Authorization before any such return. Seller shall have no obligation to make repairs or replacements which are required by normal wear and tear, or which result, in whole or in part, from catastrophe, fault or negligence of Buyer, or from improper or unauthorized use of the goods, or use of the goods in a manner for which they are not designed, or by causes external to the goods such as, but not limited to, power failure. No suit or action shall be brought against Seller more than twelve (12) months after the related cause of action has occurred. Buyer has not relied and shall not rely on any oral representation regarding the goods sold hereunder, and any oral representation shall not bind Seller and shall not be a part of any warranty.

THE PROVISIONS OF THE FOREGOING WARRANTY ARE IN LIEU OF ANY OTHER WARRANTY, WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL (INCLUDING ANY WARRANTY OR MERCHANT ABILITY OR FITNESS FOR A PARTICULAR PURPOSE). SELLER'S LIABILITY ARISING OUT OF THE MANUFACTURE, SALE OR SUPPLYING OF THE GOODS OR THEIR USE OR DISPOSITION, WHETHER BASED UPON WARRANTY, CONTRACT, TORT OR OTHERWISE, SHALL NOT EXCEED THE ACTUAL PURCHASE PRICE PAID BY BUYER FOR THE GOODS. IN NO EVENT SHALL SELLER BE LIABLE TO BUYER OR ANY OTHER PERSON OR ENTITY FOR SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES, INCLUDING, BUT NOT LIMITED TO, LOSS OF PROFITS, LOSS OF DATA OR LOSS OF USE DAMAGES ARISING OUT OF THE MANUFACTURE, SALE OR SUPPLYING OF THE GOODS. THE FOREGOING WARRANTY EXTENDS TO BUYER ONLY AND SHALL NOT BE APPLICABLE TO ANY OTHER PERSON OR ENTITY INCLUDING, WITHOUT LIMITATION, CUSTOMERS OF BUYERS.