Excellence in Compliance Testing

Certification Exhibit

FCC ID: HSW-DNT2400P
IC: 4492A-DNT2400P

FCC Rule Part: 15.247
IC Radio Standards Specification: RSS-247

ACS Project Number: 16-0346

Manufacturer: Murata Electronics North America
Models: DNT2400PC, DNT2400PP

Manual

2 of 2

the lower byte of the system address to a remote dynamically. When using fixed network addressing, any device that can connect to a parent that has any children with fixed network addresses must also have a unique fixed network address, *including child routers*.

HeartbeatIntrvl - this parameter sets the interval that heartbeat status messages are sent from each system radio. Status messages include the node's parent's *BaseModeNetID*, its own routing address, and miscellaneous performance data. The default value for this parameter is 20 seconds. If the heartbeat interval is set to 0x0000, remote heartbeats are disabled. Since router heartbeats are needed to maintain the system routing table, setting the *HeartbeatIntrvl* value on a router to 0x0000 will cause heartbeats to be sent at the default 20 second rate. Setting the *HeartbeatIntrvl* parameter to 0xFFFF will suppress heartbeats except during registration or when an error is detected in the routing table.

The base maintains a 2-bit counter for each router in the system. The counter is decremented at the base's heartbeat interval. If the base does not receive a heartbeat packet from a router for two to three heartbeat intervals, that router and all of its child routers (and their child routers, etc.) are deleted from the routing table. It is possible to set the base's heartbeat interval to a value greater than the heartbeat interval for all the other devices in the network, to prevent premature router timeouts due to network congestion causing heartbeats from routers to be delayed or lost.

The base treats router heartbeat packets differently than remote heartbeat packets. Heartbeat packets from remotes are not ACKed, while ACKs are sent to a routers originating a heartbeat packets to indicate reception. This prevents additional heartbeat transmissions by a router until the next heartbeat interval. If a router does not receive a reply to a heartbeat packet within the configured *P2PReplyTimeout* interval, it will persistently re-send the heartbeat packet until its is ACK'ed.

TreeRoutingSysID - this parameter holds the system ID for a tree-routing system.

EnableRtAcks - this parameter controls remote ACK replies for peer-to-peer data packets. The default configuration for this parameter is 0, which suppresses remote peer-to-peer ACKs. Setting this parameter to 1 enables peer-to-peer ACKs. This parameter applies to both point-to-multipoint and tree-routing peer-to-peer communications.

4.2.2 Bank 1 - System Settings

Bank	Loc'n	Name	R/W	Size in bytes	Range	Default; Options
0x01	0x00	FrequencyBand	R/W	1	0..FF	**0x00** to 0x08 = general purpose, 0x04, 0x05 & 0x07 = France, 0xFF = auto
0x01	0x01	AccessMode	R/W	1	0..4	**2 = TDMA Dynamic Slots**
0x01	0x02	BaseSlotSize	R/W	1	6..233	**50 bytes**
0x01	0x03	LeasePeriod	R/W	1	0..250	**5 s** (0 to disable)
0x01	0x04	ARQ_Mode	R/W	1	0..3	**1 = redundant broadcast to *ARQ_AttemptLimit*, *ARQAttemptLimit* pass to remotes enabled**
0x01	0x05	ARQ_AttemptLimit	R/W	1	0..63	**8 attempts**
0x01	0x06	MaxSlots	R/W	1	0..15	**4 slots**
0x01	0x07	CSMA_Predelaty	R/W	1	0..255	**0x03**
0x01	0x08	CSMA_Backoff	R/W	1	0..255	**0x0A**
0x01	0x09	MaxPropDelay	R/W	1	0..255	**0x45** (20 mi, 32.18 km)
0x01	0x0A	LinkDropThreshold	R/W	1	0..255	**0x0C**
0x01	0x0B	CSMA_RemtSlotSize	R/W	1	1..255	**64**
0x01	0x0C	CSMA_BusyThreshold	R/W	1	1..255	**20**
0x01	0x0D	RangingInterval	R/W	1	0..255	**0** = disable
0x01	0x0E	AuthMode	R/W	1	0..3	**0** = authentication disabled
0x01	0x0F	P2PReplyTimeout	R/W	1		**16** hops

Bank 1 holds configuration parameters to be input to the base only. The base passes these parameters to the remotes as needed. The exception is the *ARQ_AttemptLimit* parameter. If *ARQ_Mode* bit 1 is set to 1 at the base, the *ARQ_AttemptLimit* parameter can be set in the remotes and used.

FrequencyBand - this parameter selects the frequency hopping subband. Nine subbands are available, as shown below. Note that operation in France is limited to subbands 0x04, 0x05 and 0x07.

Subband	Channels	Frequency Range(s)	Notes
0x00	37	2409.3-2467.1 MHz	General purpose 37 channel subband
0x01	17	2441.1-2467.1 MHz	17 channel subband, avoids 802.11b/g channels 1-4
0x02	17	2409.3-2416.6 MHz, 2451.2-2467.1 MHz	17 channel subband, avoids 802.11b/g channels 5-6
0x03	17	2409.3-2426.7 MHz, 2461.3-2467.1 MHz	17 channel subband, avoids 802.11b/g channels 7-8
0x04	17	2409.3-2435.3 MHz	17 channel subband, avoids 802.11b/g channels 9-10, also for use in France
0x05	21	2409.3-2441.1 MHz	21 channel subband, avoids 802.11b/g channels 11-13, also for use in France
0x06	21	2428.1-2461.3 MHz	21 channel subband, avoids 802.11 b/g channel 1
0x07	15	2409.3-2432.4 MHz	15 channel subband, avoids 802.11b/g channels 8-13, also for use in France
0x08	15	2433.9-2455.6 MHz	15 channel subband, avoids 802.11b/g channels 1-2 and 13
0xFF	15 to 37	Auto	Autoscan for remote to match base

AccessMode - this sets the channel access mode that remotes will use to communicate with the base:

Access Mode	Description	Max # of Remotes	Remote Slot Size
0	Polling	unlimited	manual
1	CSMA	unlimited	manual
2 (default)	TDMA dynamic slots	up to 16	automatic
3	TDMA fixed slots	up to 16	automatic
4	TDMA with PTT	up to 16 + unlimited listen only	automatic

BaseSlotSize - This parameter set the maximum number of user data bytes that the base can send on a single hop. This value must be set by the user for all access modes. The default value is 50 bytes.

LeasePeriod - this sets the duration in seconds for leases that remotes receive from the base. If a period of zero is specified, then lease functions are disabled. The minimum valid lease period is two seconds. Remotes will attempt to renew their leases at an interval equal to half the lease period. For example, if the lease period is set to four seconds, remotes will renew their leases every two seconds.

ARQ_Mode - this sets the ARQ mode for delivery of application messages. In ARQ mode, an ACK is expected from the receiving radio for each message addressed and sent to it. If no ACK is received, up to *ARQ_AttemptLimit*, attempts to send the data will be made, after which the message is discarded. In redundant broadcast mode, each broadcast message is sent exactly *ARQ_AttemptLimit* times. No ACKS are sent or expected. The following bit options control this function:

bits 7..2	Not used
bit 1	If set to 0, the base can pass a new *ARQ_AttemptLimit* to the remotes If set to 1, the remotes use their own *ARQ_AttemptLimit in Bank 1*
bit 0	If set to 1, ARQ mode is enabled for Protocol mode; the base will send broadcast packets *ARQ_AttemptLimit* times instead of once. If set to 0, broadcast packets are sent once

ARQ_AttemptLimit - this sets the maximum number of attempts that will be made to send a data packet on the RF link. Setting this parameter to the maximum value of 63 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 nopackets can be lost. Note - if this mode is used in a multipoint network, one remote that has lost link will shut down the entire network if the base is trying to send it data.

MaxNumSlots - in TDMA access modes, this sets the number of slots that are allowed. In fixed slot mode, this allocates the number of slots directly. In dynamic slot mode, this sets the maximum number of slots that may be generated regardless of the number of remotes that attempt to link with the base. Any remotes requesting registration after this limit is reached will be denied registration by the base.

CSMA_Predelay - in CSMA mode 1, this parameter sets the maximum delay between when the base transmission has finished and when a remote checks for a clear channel. The value of each parameter count depends on the data rate as shown below. Refer to Section 2.10.2 for more information.

Data Rate	Parameter
38.4 kbps	1000 µs/count
115.2 kbps	800 µs/count
200 kbps	600 µs/count
500 kbps	400 µs/count

CSMA_Backoff - in a CSMA mode 1, this parameter sets the maximum length of time that a remote will back off after it detects a busy channel. The value of each parameter count depends on the data rate as shown in the *CSMA_Predelay* table shown above. Refer to Section 2.10.2 for more information.

MaxPropDelay - this is the maximum propagation delay that the base and the remotes will use in their slot timing calculations, in units of 3.1 µs. This is used to increase the amount of time dedicated to the registration slot. Increasing this value will subtract slightly from the overall slot time available to remotes for sending data. Note that the free-space round trip propagation delay for one mile is 10.72 µs. Each increment of *MaxPropDelay* thus corresponds to a maximum radius from the remote to the base of 0.29 mi (0.46 km). The default setting provides enough time to handle remotes up to 20 miles away. It is recommended to use the default setting unless a path greater than 20 miles is planned at start up. Once linked to the base, remotes will periodically update their timing based on ranging information from the base, except in Polling Mode. The frequency with which this value is updated is set by the *RangingInterval* parameter discussed below. The current range information is available in the *CurrPropDelay* parameter.

LinkDropThreshold - this is the number of consecutive beacons missed by a remote that causes the remote to restart a link acquisition search. Please contact RFM technical support before making changes to the parameter.

CSMA_RemtSlotSize - this sets the maximum size for a remote data transmission in polling or CSMA channel access modes. Setting this parameter to a large value allows a remote to send more data in a single hop, but can result in fewer remotes having time to send on a given hop. The default is 64 bytes.

CSMA_BusyThreshold - this sets the RSSI energy detection threshold that remotes use to determine whether the channel is occupied. The factory default should be sufficient for most applications and it is recommended that this value not be changed.

RangingInterval - this sets the interval in seconds/count when remotes will reassess their range to the base. Polling (mode 0) disables ranging, so remotes receive ranging information only once each time they join a network. The *RangingInterval* timer does not advance while a remote is sleeping.

AuthMode - this parameter is valid on the base only. It controls how remotes are permitted to join the network. Permitted values are:

0 = Any remote may join
1 = Authentication by base radio permission table
2 = Authentication by request to host application
3 = Lock authentication to permit only currently registered remotes

P2PReplyTimeout - This parameter sets the reply timeout for peer-to-peer packets sent from one node to another. Because each leg of the journey from one node to another and back may take multiple transmit attempts, the length of time to confirm receipt and issue a *TxDataReply* is subject to more variation than a transmission directly between a base and a remote. The *P2PReplyTimeout* parameter specifies the maximum number of hops or hop pairs that a remote will wait for a reply from its recipient. If a reply returns sooner than the timeout, the remote will send a *TxDataReply* indicating success to its host as soon as it is received, and cancels the timeout. If a reply does not come back before the timeout expires, the remote will send a *TxDataReply* to its host indicating failure. If a reply should come back after the timeout expires the remote will ignore it, as a *TxDataReply* has already been sent. The units of this parameter are in hops for non tree-routing operation and in hop pairs for tree-routing operation. The default is eight hops/eight hop pairs.

There is some coupling between the *HeartbeatIntrvl* parameter setting and the *P2PReplyTimeout* parameter setting. If the heartbeat interval in seconds is less than the *P2PReplyTimeout* in hop pairs, it is possible that a router will not repeat an un-ACKed heartbeat packet quickly enough to prevent the base from timing that router out (heartbeats are repeated only when an ACK is not received within the *P2PReplyTimeout* interval.) Thus, setting the *P2PReplyTimeout* to a very large value relative to the heartbeat interval could cause problems.

4.2.3 Bank 2 - Status Registers

Bank	*Loc'n*	*Name*	*R/W*	*Size in bytes*	*Range*	*Default*
0x02	0x00	MacAddress	*R*	3	0..0x0fffffff	fixed value
0x02	0x03	CurrNwkAddr	*R*	1	0..255	as set
0x02	0x04	CurrNwkID	*R*	1	0..255	as set
0x02	0x05	CurrRF_DataRate	*R*	1	0..3	as set
0x02	0x06	CurrFreqBand	*R*	1	0..1	as set
0x02	0x07	LinkStatus	*R*	1	0..4	current status
0x02	0x08	RemoteSlotSize	*R*	1	0..243	as set
0x02	0x09	TDMA_NumSlots	*R*	1	0..16	as set
0x02	0x0A	Reserved	*R*	1	0..255	reserved
0x02	0x0B	TDMA_CurrSlot	*R*	1	0..16	current slot
0x02	0x0C	HardwareVersion	*R*	1	0..255	0x00 = DNT2400 rev A
0x02	0x0D	FirmwareVersion	*R*	1	0..255	current firmware load
0x02	0x0E	FirmwareBuildNum	*R*	2	$0..2^{16}$	current firmware load
0x02	0x10	Reserved	*R*	1	0..255	reserved
0x02	0x11	SuperframeCount	*R*	1	0..255	current value
0x02	0x12	RSSI_Idle	*R*	1	0..255	as set
0x02	0x13	RSSI_Last	*R*	1	0..255	as set
0x02	0x14	CurrTxPower	*R*	1	0..255	as set
0x02	0x15	CurrAttemptLimit	*R*	1	0..255	as set
0x02	0x16	CurrRangeDelay	*R*	1	0..255	as set
0x02	0x17	FirmwareBuildDate	*R*	8	ASCII	as set
0x02	0x1F	FirmwareBuildTime	*R*	8	ASCII	as set
0x02	0x27	ModelNumber	*R*	1	0..255	0x01
0x02	0x28	CurrBaseModeNetID	*R*	1	0..63, 255	0xFF
0x02	0x29	AveRXPwrOvHopSeq	*R*	1	0..255	as received
0x02	0x2A	ParentACKQual	*R*	1	0..255	4*number of attempts to get ACK

MacAddress - returns the radio's unique 24-bit MAC address.

CurrNwkAddr - this returns the address of the radio in its parent's network.

CurrNwkID - this returns the ID of the network the radio is currently assigned to or connected to. A value of 0xFF means the radio is scanning for a network but has not yet joined one.

CurrRF_DataRate - this returns the RF data rate of the network that the radio is currently assigned to or connected to. If the radio is scanning for a network, this is the current data rate it is using in the scan.

CurrFreqBand - this returns the frequency band of the network that the radio is currently assigned to or connected to. A value of 0xFF means the radio is scanning for a network but has not yet joined one.

LinkStatus - this returns the radio's current connection status to the network. The following codes are defined:

LinkStatus	**Remote Status**	**Base Status**
0	initializing	initializing
1	unlinked, scanning for a network	not used
2	linked, acquiring network parameters	not used
3	linked, registering with the base	not used
4	linked and registered	ready for data transfer

RemoteSlotSize - returns the current remote slot size, defined as the maximum number of message bytes a remote can send on a single hop. When using protocol mode, the entire packet, including overhead bytes must be less than or equal to this value or the packet will be discarded. In the three TDMA modes the remote slot size is automatically computed, and this value is read-only. In polling and CSMA modes, the remote slot size must be set by the user. The parameter to set this is *CSMA_RemtSlotSize* in Bank 1.

TDMA_NumSlots - in TDMA access modes, this returns the number of slots currently allocated.

TDMA_CurrSlot - returns the current TDMA slot number assigned to the remote in modes where the slot position is automatically computed. In modes where this number is not applicable, it is read as 0xFF.

HardwareVersion - returns an identifier indicating the type of radio. A value of 0x41 is defined for the DNT2400 Rev A hardware.

FirmwareVersion - returns the firmware version of the radio in 2-digit BCD format.

FirmwareBuildNum - returns the firmware build number, in binary format.

SuperframeCount - returns the current superframe count. The count increments every 64 hops.

RSSI_Idle - returns the last measurement of RSSI made during a time when the RF channel was idle. Can be used to assess the noise floor or detect interferers.

RSSI_Last - returns the last measurement of RSSI made during the receipt of an RF packet with a valid CRC. Can be used for network commissioning and diagnostic purposes.

CurrTxPower - returns the current transmitter power setting of a remote, allowing the automatic power setting to be tracked. This parameter is the nominal output power setting in dBm, and is a 2's complement value. Note that the *CurrTxPower* parameter value returned from a base or repeater is not valid.

CurrAttemptLimit - this returns the value of *ARQ_AttemptLimit* currently in use (depending on the selected ARQ_Mode, it may not always match the local EEPROM value).

CurrRangeDelay - returns the current propagation delay for this remote as measured from the base (applies to remote nodes only).

FirmwareBuildDate - date of firmware build in MM/DD/YY format.

FirmwareBuildTime - time of firmware build in HH:MM:SS format.

ModelNumber - DNT model number parameter, 0x01 = DNT900, 0x02 = DNT2400.

CurrBaseModeNetID - returns the current base-mode network ID.

AveRXPwrOvHopSeq - returns the average beacon power received over the last tree-routing hop sequence.

ParentACKQual - returns the number of transmission sent before and ACK is received, multiplied by 4.

4.2.4 Bank 3 - Serial and SPI Settings

Bank	Loc'n	Name	R/W	Size in bytes	Range	Default
0x03	0x00	SerialRate	*R/W*	2	1..384	**0x0030** (9.6 kb/s)
0x03	0x02	SerialParams	*R/W*	1	0..7	**0x00** (8N1)
0x03	0x03	SerialControls	*R/W*	1	0..7	**0X07**
0x03	0x04	SPI_Mode	*R/W*	1	0..2	**0x00** (SPI disabled)
0x03	0x05	SPI_Divisor	*R/W*	1	$1..2^7$	**0x0A** (80.64 kb/s)
0x03	0x06	SPI_Options	*R/W*	1	0..3	**0x00** (standard SPI configuration)
0x03	0x07	SPI_MasterCmdLen	*R/W*	1	$0..2^5$	**0x00**
0x03	0x08	SPI_MasterCmdStr	R/W	32	ASCII	**all 0x00 bytes**

SerialRate - sets the serial rate divisor according to the following formula:

Serial rate in b/s = 460800/SerialRate

Serial rate division settings for commonly used baud rates are:

Setting	Serial rate
0x0001	460.8 kb/s
0x0002	230.4 kb/s
0x0004	115.2 kb/s
0x0006	76.8 kb/s
0x0008	57.6 kb/s
0x000C	38.4 kb/s
0x0010	28.8 kb/s
0x0018	19.2 kb/s
0x0030	9.6 kb/s (default)
0x0060	4.8 kb/s
0x00C0	2.4 kb/s
0x0180	1.2 kb/s

SerialParams - 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.

SerialControls - this parameter affects the way the radio responds to the various serial control lines. Enabling or disabling response to some serial control signals can facilitate communicating with devices that support only a reduced serial interface. The parameter is defined as a bitmask, with the following options:

bits 7..3 Reserved
bit 2 Base /DCD mode:
1 = The base will only assert /DCD when at least one remote is registered (default).
0 = The base always asserts /DCD, regardless of whether any remotes are attached.
bit 1 /HOST_RTS enable:
1 = Radio will respond to changes on the /HOST_RTS control line (default).
0 = Radio ignores the /HOST_RTS pin and assumes flow control is always asserted.
bit 0 SLEEP/DTR enable.
1 = Radio will respond to changes on the SLEEP Pin (default)
0 = Radio ignores the SLEEP Pin and is always in the awake state.

SPI_Mode - this register enables and configures SPI port operation. When SPI functions are enabled, the primary serial (UART) port operation is disabled in SPI Slave mode and restricted in SPI Master mode. The diagnostic serial port continues to operate normally. Note that only protocol formatted messages can be used when a DNT2400 is configured for SPI operation. *SPI_Mode* has the following settings:

Setting	*Mode*
0x00	SPI disabled - serial UART mode (default)
0x01	SPI Slave mode
0x02	SPI Master mode

When a DNT2400 is configured for SPI Slave mode operation, all messages are routed through the SPI port in lieu of the primary serial (UART) port. The /HOST_CTS signal provides the same flow control function for the MOSI input that it provides for the RADIO_RXD serial input. The Master (host) can clock transmit messages into the DNT2400 SPI Slave whenever /HOST_CTS is set to a logic low state. The Master can also complete clocking a protocol formatted transmit message into the DNT2400 if /HOST_CTS switches high part way through the message, but must then stop inputting transmit messages until the DNT2400 resets /HOST_CTS to a logic low state.

In order for the Master to receive data from a DNT2400 SPI Slave, it must clock bytes into the DNT2400. These bytes may be message bytes and/or 0x00 null bytes. The DNT2400 will return null bytes on the MISO output until the DNT2400 receives a packet. The received message will then be clocked out. GPIO4 can be alternately configured to provide an SPI RX data available flag, SPI_RX_AVL, to signal when the DNT2400 slave is holding a received message(s). See Section 2.13 for additional information. In SPI Slave mode, the maximum continuous SPI clock rate supported is 80.64 kb/s. The Master (host) clock rate should closely match the DNT2400 SPI clock rate setting for best data transfer efficiency. See the *SPI_Divisor* description below.

In SPI slave mode, de-asserting and then asserting the /SS line resets the DNT2400 SPI port on a byte boundary. The /SS line can be toggled this way between every byte to assure bit streams into and out of the SPI port remain byte framed. Less frequent /SS line toggling is also acceptable in most applications. It is recommended that /SS be toggled at the start and end of each transmit message, and after no more than 256 null bytes when clocking to output a received message. The /SS line should also be toggled at

the end of each received message. Figure 4.2.4.1 shows a typical relation ship between the /SS line (red trace) and the SCLK line (blue trace).

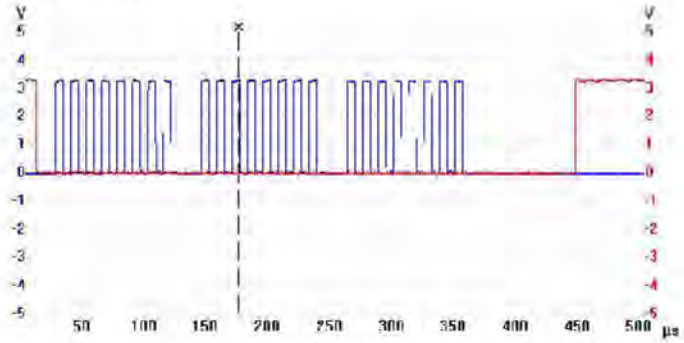


Figure 4.2.4.1

When periodic I/O reporting is enabled on a DNT2400 remote configured as an SPI Master, the remote will clock out a stored command string, *SPI_MasterCmdStr*, to collect data from a Slave peripheral each time the I/O report timer fires. The collected data is then transmitted to the base as a *TXData* message.

Alternatively, a host connected to the base can transmit an SPI command as a *TXData* message to the remote. The remote will clock the command into its Slave peripheral and transmit back the Slave's response. In either case, the command string and response string are limited to 32 bytes. Only data messages are routed through the DNT2400's SPI port in Master mode. Command packets and command replies are routed through the primary serial port.

When configured as an SPI Master, the DNT2400 sets /SS low one SPI bit period before the start of message clocking and sets /SS high after clocking the last message bit. When the DNT2400 is operating in SPI Slave mode, the Master (host) must set /SS low at least one SPI bit period before clocking data in/out of the DNT2400. See the *SPI_Divisor* description below.

Changes to the *SPI_Mode* setting must be saved and the DNT2400 reset to take effect. This avoids the possibility of setting SPI mode inadvertently and being unable to communicate with the DNT2400 to switch it back to serial mode. If the /CFG pin is grounded at power up, the *SPI_Mode* setting is overridden and the DNT2400 will start up in serial (UART) mode.

SPI_Divisor - this parameter sets the clock rate in SPI Master mode and clock rate related timing, such as /SS sampling, in Slave mode. The SPI rate is set according to the following formula:

SPI rate in b/s = 806400/SPI_Divisor

The valid range for *SPI_Divisor* is 1 to 127, providing SPI rates from 6.35 to 80.64 kb/s. For best data transfer efficiency in Slave mode, the Master (host) clock rate should closely match the DNT2400 SPI data rate setting.

SPI_Options - this parameter sets the clock options for the SPI modes:

Setting	Clock Option
0x00	Normal operation
0x01	Second clock edge
0x02	High idle clock polarity
0x03	Second clock edge with high idle clock polarity

SPI_MasterCmdLen - 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.

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

4.2.5 Bank 4 - Host Protocol Settings

Bank	Loc'n	Name	R/W	Size in bytes	Range	Default; Options
0x04	0x00	ProtocolMode	*R/W*	1	0..1	**0 = transparent**; 1 = protocol
0x04	0x01	ProtocolOptions	*R/W*	1	0..255	**0x05**
0x04	0x02	TxTimeout	*R/W*	1	0..255	**0x00 (no timeout)**
0x04	0x03	MinPacketLength	*R/W*	1	1..255	**1 byte**
0x04	0x04	AnnounceOptions	*R/W*	1	0..7	**0x07 all enabled**
0x04	0x05	TransLinkAnnEn	*R/W*	1	0..1	**0 = disabled**; 1 = <LINK> announce
0x04	0x06	ProtocolSequenceEn	*R/W*	1	0..2	0 = disabled; 1 = startup, **2 = anytime**
0x04	0x07	TransPtToPtMode	*R/W*	1	0..1	**0 = multipoint**, 1 = point-to-point
0x04	0x08	MaxPktsPerHop	*R/W*	1	0..3	**0x03**

ProtocolMode - this parameter selects the host protocol mode. The default is 0, 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 DNT2400'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 1 enables the DNT2400 host protocol, which is recommended for point-to-multipoint applications and is preferred for new designs. It is not necessary to define the same protocol mode for all radios in a network. For example, it is frequently useful to configure all the remotes for transparent mode and the base for protocol mode. Note that it is possible for the host to switch the radio from transparent mode to protocol mode and back if desired by transmitting an *EnterProtocolMode* command.

ProtocolOptions - this is a bitmask that selects various options for the protocol mode. The default is 0x05.

bits 7..3	Reserved
bit 2	Enable output of TxReply packets
bit 1	Reserved
bit 0	Enable output of Announce packets

AnnounceOptions - this is a bitmask that enables/disables different types of Announce packets:

bit 7..3	Reserved
bit 2	Enable bit for Announce types E0-EA (error notification)
bit 1	Enable bit for Announce types A1-A7 (<LINK> notifications)
bit 0	Enable bit for Announce types A0 (initialization)

TxTimeout - this parameter is the transmit timeout used for determining message boundaries in transparent data mode. 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.

MinPacketLength - sets the minimum message length used for determining packet boundaries in transparent data mode. 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 *TxTimout*.

TransLinkAnnEn - enables a link announcement function for transparent mode. Whenever link is acquired or dropped, the strings "<LINK>" or "<DROP>" are sent to the local host.

ProtocolSequenceEn - enables or disables the *EnterProtocolMode* ASCII command string to switch from transparent mode to protocol mode. Valid settings are 0 = disabled, 1 = one time at startup, 2 = enabled at any time. The default is enabled at anytime.

TransPtToPtMode- controls the behavior for addressing packets in transparent mode. When this setting is zero (default), in transparent mode the base will direct packets to the broadcast address. This is useful for point-to-multipoint where the base is sending data to multiple remotes, for instance in applications where a wireless link is replacing an RS-485 serial bus. When this setting is one, in transparent mode the base will direct packets to the last remote that registered with it. This is useful for point-to-point networks where there are only two endpoints, for instance in applications where a simple serial cable is being replaced.

MaxPktsPerHop - this parameter sets a limit on the maximum number of packets a radio can send on each frequency hop. The default value is 3, the range is 1 to 3.

4.2.6 Bank 5 - I/O Peripheral Registers

Bank	Loc'n	Name	R/W	Size in bytes	Range in bits	Default
0x05	0x00	GPIO0	R/W	1	1	0
0x05	0x01	GPIO1	R/W	1	1	0
0x05	0x02	GPIO2	R/W	1	1	0
0x05	0x03	GPIO3	R/W	1	1	0
0x05	0x04	GPIO4	R/W	1	1	0
0x05	0x05	GPIO5	R/W	1	1	0
0x05	0x06	ADC0	R	2	10	N/A
0x05	0x08	ADC1	R	2	10	N/A
0x05	0x0A	ADC2	R	2	10	N/A
0x05	0x0C	Event Flags	R	2	10	N/A
0x05	0x0E	PWM0	R/W	2	9	0
0x05	0x10	PWM1	R/W	2	9	0

GPIO0..5 - writing to these registers sets the corresponding driver for pins that are enabled outputs. Writing to pins that are enabled as inputs enables or disables the internal pull-up. Reading these registers returns the current level detected on the corresponding pins.

ADC0..2 - read-only, returns the current 10-bit ADC reading for the selected register. See the discussion of the *ADC_SampleIntvl* 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 message:

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 3	GPIO3 edge transition
bit 2	GPIO2 edge transition
bit 1	GPIO1 edge transition
bit 0	GPIO0 edge transition

PWM0..1 - sets the PWM (DAC) outputs. The DC voltage derived from the integrated low-pass filters on the PWM output provides an effective DAC resolution of 7 bits (8 bits achievable with external filtering). The range of this parameter is 0x0000 to 0x00FF.

4.2.7 Bank 6 - I/O Setup

Bank	Loc'n	Name	R/W	Size in bytes	Range in bits	Default; Options
0x06	0x00	GPIO_Dir	R/W	1	6	**0** (all inputs)
0x06	0x01	GPIO_Init	R/W	1	6	**0** (all zeros)
0x06	0x02	GPIO_Alt	R/W	1	6	**0x08** = use GPIO3 for RS485 enable
0x06	0x03	GPIO_Edge Trigger	R/W	1	8	**0x00**
0x06	0x04	GPIO_SleepMode	R/W	1	1	**0 = off**; 1 = use sleep I/O states
0x06	0x05	GPIO_SleepDir	R/W	1	6	**0** (all inputs)
0x06	0x06	GPIO_SleepState	R/W	1	6	**0** (all zeros)
0x06	0x07	PWM0_Init	R/W	2	10	**0x0000**
0x06	0x09	PWM1_Init	R/W	2	10	**0x0000**
0x06	0x0B	ADC_SampleIntvl	R/W	2	16	**0x0001** (10 ms)
0x06	0x0D	ADC0 ThresholdLo	R/W	2	10	**0x0000**
0x06	0x0F	ADC0_ThresholdHi	R/W	2	10	**0x03FF**
0x06	0x11	ADC1_ThresholdLo	R/W	2	10	**0x0000**
0x06	0x13	ADC1_ThresholdHi	R/W	2	10	**0x03FF**
0x06	0x15	ADC2_ThresholdLo	R/W	2	10	**0x0000**
0x06	0x17	AC2_ThresholdHi	R/W	2	10	**0x03FF**
0x06	0x19	IO_ReportTrigger	R/W	1	0..1	**0x01 (GPIO0)**
0x06	0x1A	IO_ReportInterval	R/W	4	32	**0x00000BB8** (every 30 seconds)
0x06	0x1E	IO_ReportPreDel	R/W	1	8	**0x00**
0x06	0x1F	IO_ReportRepeat	R/W	1	8	**0x01**

GPIO_Dir - this parameter is a bitmask that sets whether the GPIOs are inputs (0) or outputs (1). The default is all inputs.

GPIO_Init - 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.

GPIO_Alt - this parameter is a bitmask to select alternate functions for GPIO3, GPIO4 and GPIO5 as shown below:

Bit	Alternate function
3	GPIO3 functions as an RS-485 driver enable output
4	GPIO4 functions as an SPI_RX_AVL (SPI RX data available) flag
5	GPIO5 functions as an antenna diversity control output

GPIO_EdgeTrigger - when GPIO triggers are enabled for automatic I/O reporting, this function controls the trigger behavior:

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

GPIO_SleepMode - when set to 1, this parameter enables setting of GPIOs to the designated direction and state whenever a device is asleep.

GPIO_SleepDir - when *GPIO_SleepMode* is enabled, this parameter functions as a secondary *GPIO_Dir* 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. Set a *GPIO_SleepDir* bit to 1 to specify an output, or to 0 to specify an input.

GPIO_SleepState - when *GPIO_SleepMode* 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.

PWM0_Init - this parameter sets the initial value for PWM0 at startup.

PWM1_Init - this parameter sets the initial value for PWM1 at startup.

ADC_SampleIntvl - this parameter sets the interval between the beginning of one ADC read cycle and the next ADC read cycle. The three ADC inputs are read on each ADC read cycle. Each *ADC_SampleIntvl* count equals 10 ms. This interval will be the worst-case latency for ADC generated interrupts. This interval is independent of the *IO_ReportInterval* as the ADCs will be read again on that interval.

ADC0..2_ThresholdLo/Hi - these values define 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"; i.e., if the measurement remains there, additional reports are not triggered until the value crosses the threshold again. The thresholds are met whenever one of the following inequalities are satisfied:

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

IO_ReportTrigger - when a selected trigger source is enabled, a trigger event will cause the remote to send an EVENT message to its base containing the entire current values of the I/O Register Bank from GPIO0 up to and including the *EventFlags*, but not the PWM settings which are output-only.

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

I/O reporting is supported for remotes only, not the base.

IO_ReportInterval - when periodic I/O reporting is enabled, this parameter sets the interval between reports. Units are 10 ms increments, and the default report interval is every 30 seconds but can be set as long as 497 days.

IO_ReportPreDel - this parameter sets the delay in milliseconds between an event trigger and the time the I/O register bank is read and sent in an EVENT report.

IO_ReportRepeat - this parameter sets the number of times the I/O register bank is read and resent as an EVENT report following an event trigger. The default parameter value is 1, causing the EVENT report to be sent once.

4.2.8 Bank 7 - Authentication List

Bank	Loc'n	Name	R/W	Size in bytes
0x07	0x00	ApprovedAddr0	R/W	3
0x07	0x03	ApprovedAddr1	R/W	3
0x07	0x06	ApprovedAddr2	R/W	3
0x07	0x09	ApprovedAddr3	R/W	3
0x07	0x0C	ApprovedAddr4	R/W	3
0x07	0x0F	ApprovedAddr5	R/W	3
0x07	0x12	ApprovedAddr6	R/W	3
0x07	0x15	ApprovedAddr7	R/W	3
0x07	0x18	ApprovedAddr8	R/W	3
0x07	0x1B	ApprovedAddr9	R/W	3
0x07	0x1E	ApprovedAddr10	R/W	3
0x07	0x21	ApprovedAddr11	R/W	3
0x07	0x24	ApprovedAddr12	R/W	3
0x07	0x27	ApprovedAddr13	R/W	3
0x07	0x2A	ApprovedAddr14	R/W	3
0x07	0x2D	ApprovedAddr15	R/W	3

ApprovedAdd0..15 - the three-byte parameters in Bank 7 are the MAC addresses of the remotes authorized to join the network. The addresses are entered in little-endian format such that a radio with MAC address 012345 would be entered 0x452301.

4.2.9 Bank 8 - Tree-Routing Active Router ID Table

Bank	Loc'n	Name	R/W	Size in bytes
0x08	0x00	Base NetworkID (0x00)	R	1
0x08	0x01 to 0x3F	ParentNetworkID1 ParentNetworkID63	R	1

ParentNetID0..63 - this set of parameters contains the tree-routing active router ID table maintained by a base for its system. It describes the organization of all active routers in the system. This table is used by the base and the routers to determine which direction to send a packet. The base updates the information in the routing table from the heartbeat packets it receives from the routers in the system, and broadcasts the routing table periodically to inform all devices in the system of the current system configuration.

4.2.10 Bank 9 - Registered MAC Addresses

Bank	Loc'n	Name	R/W	Size in bytes
0x09	0x00 to 0x19	RegMACAddr0 RegMACAddr25	R	15

RegMACAddr0..25 - this bank holds the MAC addresses of all radios registered to a base or router. Up to 126 MAC addresses can be registered. Each bank parameter can hold up to five MAC addresses, with each MAC address containing three bytes are in little-Endian order. Three-byte segments in a parameter

not holding a MAC address with hold a null address: 0x00 0x00 0x00. Note that unlike parameters in other banks, the bank offset used in get commands is *by parameter rather than by byte*. Only one Bank 9 parameter can be retrieved at a time with a get command. In a remote, this bank will contain only null addresses.

4.2.11 Bank FF - Special Functions

This bank contains three user functions, *UcReset*, *SleepModeOverride* and *MemorySave*:

Bank	Loc'n	Name	R/W	Size in bytes	Range	Description
0xFF	0x00	UcReset	*W*	1	0..90	0x00 = reset, 0x5A = reset with factory defaults
0xFF	0x0C	SleepModeOverride	R/W	1	0..2	**0 = inactive**, 1 = stay awake, 2 = cancel stay awake
0xFF	0x1C	RoutingTableUpd	R/W	1	0..255	**0x14 (20 seconds)**
0xFF	0x20	DiagSerialRate	R/W	2	0..384	**0x000C (38.4 kb.s)**
0xFF	0xFF	MemorySave	*W*	1	0..2	0x00 = load factory defaults, 0x01 = save settings to EEPROM, 0x02 = save settings and reset

UcReset - writing a value of 0x00 to this location forces a software reset of the microcontroller. This will enable those changes which require a reset. If this is written to before 0x01 is written to the *MemorySave* parameter, the last parameter values saved before the reset will be in effect. A reply packet, either local or over-the-air, may not be received when writing a value to this register. Writing 0x5A to this location will reset the radio and load the factory default settings. This is equivalent to writing 0x00 to *UcReset* followed by writing 0x00 to *MemorySave*. Note that 0x01 must be written to *MemorySave* to save the retrieved factory defaults.

SleepModeOverride - when remotes are operating in sleep mode, writing 0x01 to the location will cause the remotes to stay awake. Writing 0x00 to this location causes the remotes to resume sleeping in 10 seconds. Writing 0x02 to this location causes the remotes to resume sleeping immediately (subject to their configuration).

RoutingTableUpd - this parameter is the interval in seconds for the base station to broadcast the tree-routing table to its system. The default interval is 20 seconds.

DiagSerialRate - sets the diagnostic port serial rate divisor according to the following formula:

Serial rate in b/s = 460800/DiagSerialRate

Serial rate division settings for commonly used baud rates are:

Setting	Serial rate
0x0000	460.8 kb/s
0x0001	460.8 kb/s
0x0002	230.4 kb/s
0x0004	115.2 kb/s
0x0006	76.8 kb/s
0x0008	57.6 kb/s
0x000C	38.4 kb/s (default)
0x0010	28.8 kb/s
0x0018	19.2 kb/s
0x0030	9.6 kb/s
0x0060	4.8 kb/s
0x00C0	2.4 kb/s
0x0180	1.2 kb/s

Note that if a value of 0x0000 is specified, the maximum data rate of 460.8 kb/s will be selected

MemorySave - writing 0x00 to this location clears all registers back to factory defaults. Writing a 0x01 to this location commits the current register settings to EEPROM. Writing 0x02 to this location saves the current setting to EEPROM and forces a software reset. When programming registers, all changes are considered temporary until an 0x01 or 0x02 command is executed.

4.2.12 Protocol Mode Configuration Example

In this example, the host configures the base to transmit 10 dBm (10 mW) of RF power using the *SetRegister* command, 0x04. The *TxPower* parameter is stored in bank 0x00, register 0x18. A one-byte parameter value of 0x01 selects the 10 dBm (10 mW) power level. The protocol formatting for the command is:

0xFB 0x05 0x04 **0x18 0x00 0x01 0x01**

Note the order of the bytes in the command argument: register, bank, span, parameter value. When the base receives the command it updates the parameter setting and return a *SetRegisterReply* message as follows:

0xFB 0x01 **0x14**

In order for this new RF power setting to persist through a base power down, *MemorySave* must be invoked. This is done by setting a one-byte parameter in register 0xFF of bank 0xFF to 0x01 with another *SetRegister* command:

0xFB 0x05 0x04 **0xFF 0xFF 0x01 0x01**

The base will write the current parameter values to EEPROM and return a *SetRegisterReply* message:

0xFB 0x01 **0x14**

4.2.13 Protocol Mode Sensor Message Example

In this example, the base host requests an ADC1 reading from a remote using the *GetRemoteRegister* command, 0x0A. The MAC address of the remote is 0x000102. The current ADC1 measurement is read from register 0x08 in bank 0x05. The ADC reading spans two bytes. The protocol formatting for this command is:

0xFB 0x07 0x0A **0x02 0x01 0x00** 0x08 0x05 0x02

Note the remote MAC address 0x000102 is entered in Little-Endian byte order, 0x02 0x01 0x00. The ADC reading is returned in a *GetRemoteRegisterReply* message:

0xFB 0x0B 0x1A **0x00** 0x02 0x01 0x00 **0xC4** 0x08 0x05 0x02 **0xFF 0x02**

Substantial information is returned in the message. The last two byes of the message give the ADC reading in Little-Endian format, 0xFF 0x02. The ADC reading is thus 0x02FF. The RSSI value is the byte following the address, 0xC4 (-60 dBm). The *TxStatus* byte to the right of the *GetRemoteRegisterReply* Packet Type is 0x00, showing the packet was acknowledged on the RF channel.

4.2.14 Protocol Mode Event Message Example

In this example, the *IO_ReportInterval* is set to 10 seconds and the *periodic report timer* bit in the *IO_ReportTrigger* parameter is set on the remote, with MAC address 0x123456. This causes event messages to be sent from this remote every 10 seconds. The *IO_ReportInterval* and the *IO_ReportTrigger* parameters are loaded using *SetRemoteRegister* commands. The command to set the *IO_ReportInterval* to 10 seconds is:

0xFB 0x0B 0x0B 0x56 0x34 0x12 0x1A 0x06 0x04 **0xE8 0x03 0x00 0x00**

The *IO_ReportInterval* parameter starts in location 0x1A of bank 0x06. The report interval is set in 10 ms units, so a 10 second report interval is 1000 units or 0x000003E8 (Little-Endian format E8 03 00 00). The *IO_ReportInterval* parameter is updated and *SetRemoteRegisterReply* is returned:

0xFB 0x06 0x1B 0x00 0x56 0x34 0x12 0xC4

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

0xFB 0x08 0x0B 0x56 0x34 0x12 0x19 0x06 0x01 **0x10**

The *periodic report timer* bit in *IO_ReportTrigger* is located in bit position four (00010000b) or 0x10. The *IO_ReportTrigger* parameter is updated and *SetRemoteRegisterReply* is returned:

0xFB 0x06 0x1B 0x00 0x56 0x34 0x12 0xC4

The remote will start sending event messages on 10 second intervals as shown in the log records below:

```
FB 16 28 56 34 12 CB 00 05 0E 01 00 00 00 01 01 F9 01 DF 01 C9 01 10 00
FB 16 28 56 34 12 B6 00 05 0E 01 00 00 00 01 01 F8 01 DF 01 CC 01 10 00
FB 16 28 56 34 12 B3 00 05 0E 01 00 00 00 01 01 F8 01 E0 01 CC 01 10 00
FB 16 28 56 34 12 B1 00 05 0E 01 00 00 00 01 01 F9 01 DF 01 C9 01 10 00
FB 16 28 56 34 12 AE 00 05 0E 01 00 00 00 01 01 F9 01 DF 01 C8 01 10 00
FB 16 28 56 34 12 AD 00 05 0E 01 00 00 00 01 01 F9 01 E1 01 CF 01 10 00
```

IO_ReportTrigger generates *RxEvent* messages (*PktType* 0x28). The message payload consists of the first 14 bytes in Bank 5, including the state of GPIO0 through GPIO5, the input voltages measured by ADC0 through ADC2, and the state of the event flags. Note the ADC readings and the event flags are presented in Little-Endian order.

5.0 DNT2400DK Developer's Kit

Figure 5.0.1 shows the main contents of a DNT2400DK Developer's kit:

Figure 5.0.1

5.1 DNT2400DK Kit Contents

- Two DNT2400P radios installed in DNT interface boards (labeled Base and Remote)
- Two installed U.FL coaxial jumper cables and two 2 dBi dipole antennas
- 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, one RJ-11/DB-9F cable assembly, two A/B USB cables
- One DNT2400DK documentation and software CD

5.2 Additional Items Needed

To operate the kit, the following additional items are needed:

- One PC with Microsoft Windows XP or Vista Operating System. The PC must be equipped with a USB port or a serial port capable of operation at 9.6 kb/s.

5.3 Developer's Kit Operational Notes:

DNT2400DK kits are preconfigured to run at a 500 kb/s RF data rate with 1 mW of RF transmitter power. Due to the high sensitivity of the DNT2400 radio module which provides its exceptional range, if the RF transmit power is increased from the default 1 mW, the DNT2400 nodes must be separated by a minimum of 6 feet for 10 mW or 25 feet at 63 mW in order to reliably link.

5.4 Developer's Kit Default Operating Configuration

The default operating configuration of the DNT2400DK developer's kit is TDMA Mode 2, point-to-point, with transparent serial data at 9.6 kb/s, 8N1. One DNT2400P 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 DNT Demo utility discussed in Section 5.5. The default RF power setting is 0 dBm (1 mW), which is suitable for operation at a spacing of about 2 m (6 ft). The RF power level should be set higher as needed for longer range operation. Note that setting the RF power to a high level when doing testing at 2 m can overload the DNT2400P receiver and cause erratic operation. See Section 5.3.

5.5 Developer's Kit Hardware Assembly

Observe ESD precautions when handling the kit circuit boards. The components that make up a development board are shown in Figure 5.5.1, and are shipped with the DNT2400P radios and U.FL coax jumper cables installed in the interface board. If a DNT2400P radio and/or the U.FL jumper cable has been unplugged after receipt, confirm the DNT2400P 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 5.5.2. 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, install the dipole antennas.

Figure 5.5.1

Figure 5.5.2

As shown in Figure 5.5.3, there is a jumper on pins J14. This jumper can be removed and a current meter connected across J14 to measure just the DNT2400's current consumption during operation.

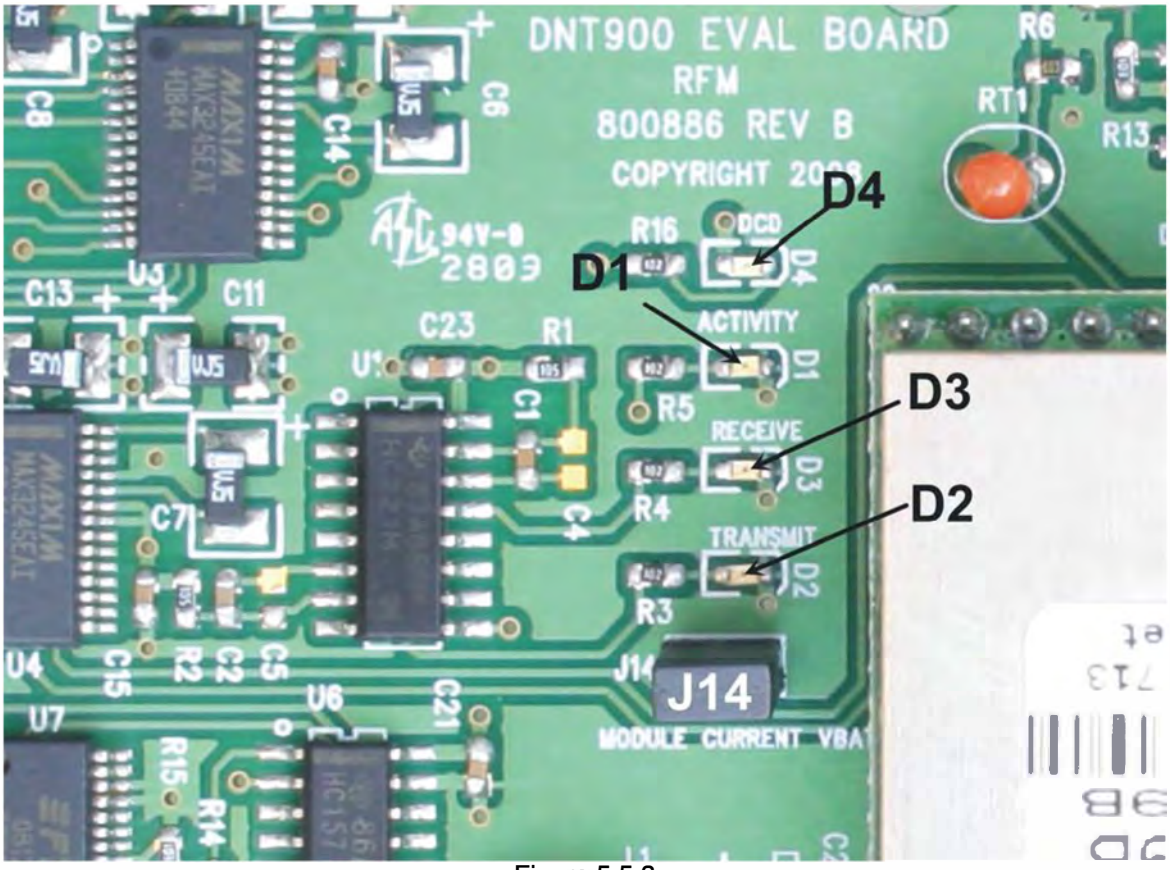


Figure 5.5.3

There are three serial connectors on the interface boards, as shown in Figure 5.5.4. The RJ-45 connector provides a high-speed RS232 interface to the DNT2400P's main serial port. The USB connector provides an optional interface to the radio's main serial port. The RJ-11 connector provides a high-speed RS232 interface to the radio's diagnostic port. The DNT Demo utility program runs on the radio's main port.

Figure 5.5.4

Many desktop PCs have a built-in serial port capable of operation at 9.6 kb/s. The kit can be run satisfactorily at the 9.6 kb/s 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.

5.6 DNT Demo Utility Program

The DNT Demo utility requires only one PC for initial kit operation and sensor applications (ADC, PWM and digital I/O). Two serial/USB ports are required for bidirectional serial communications. Section 5.6.1 below covers using the DNT Demo utility for initial kit operation and familiarization. Section 5.6.2 covers serial message communication and radio configuration.

5.6.1 Initial Kit Operation

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

The DNT Demo utility program runs on the radio's main port. The preferred PC interface is a serial port capable of operating at 9.6 kb/s or faster. As discussed above, the USB interface can also be used. Connect the Base to the PC and power up the Base and the Remote development boards using the wall-plug power supplies.

The DNT Demo utility program is located in the *PC Programs* folder. The DNT Demo requires no installation and can be simply copied to the PC and run. Start the DNT Demo on the PC. The start-up window is shown in Figure 5.6.1.1.

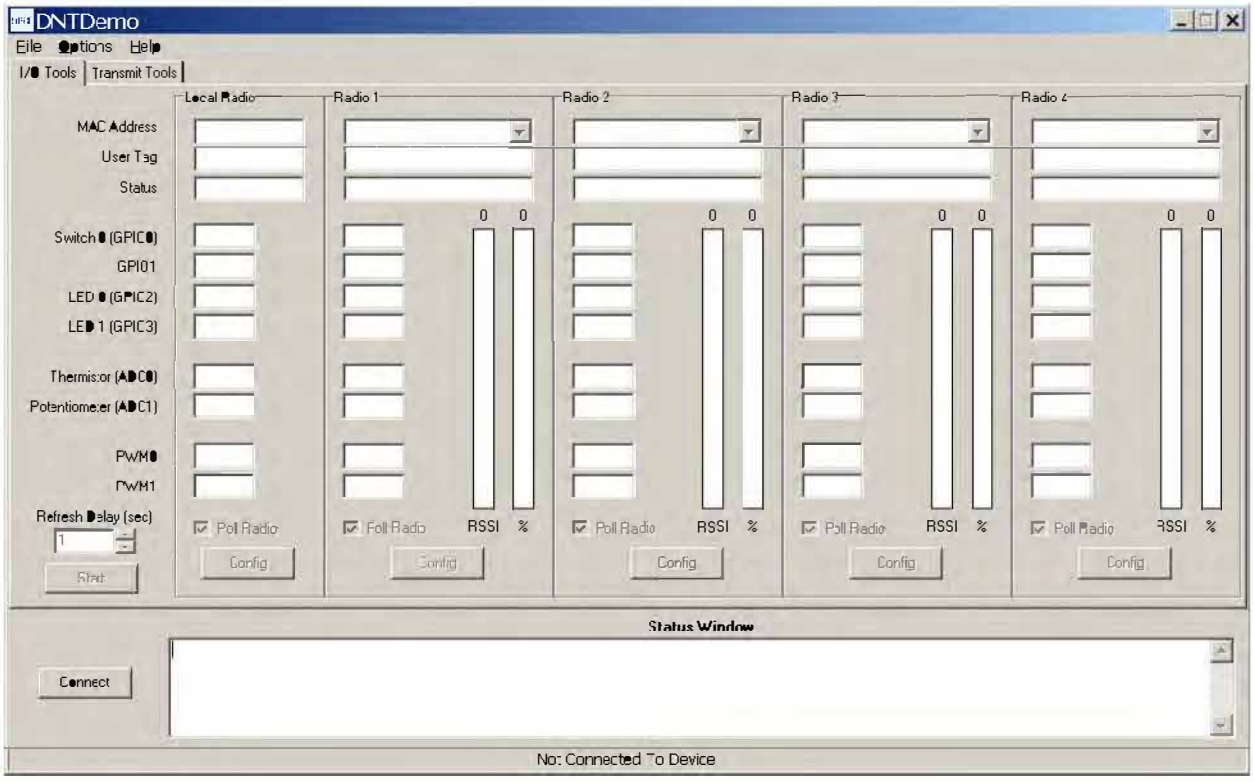


Figure 5.6.1.1

Click on *Connect* to open the *Select Comm Port Settings* dialog box, as shown in Figure 5.6.1.2. Set the baud rate to 9600 (9.6 kb/s). 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.

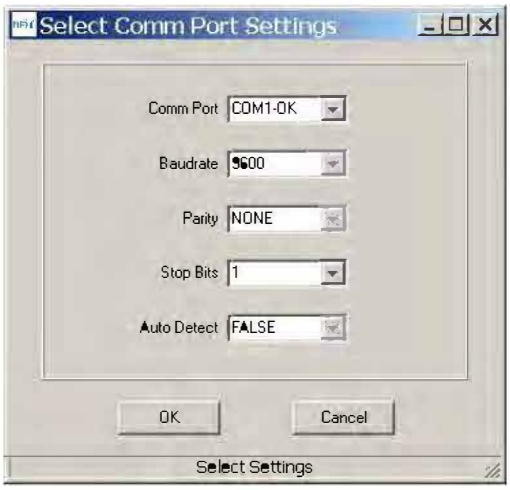


Figure 5.6.1.2

At this point the Demo will collect data from the Base, filling in data in the *Local Radio* column on the Demo window as shown in Figure 5.6.1.3. The *Status Window* should also show that the Remote has joined the Base. Click on the drop-down box at the top of the *Radio 1* column and select the *MAC Address* for the Remote. Next press the *Start* button using the default 1 second *Refresh Delay*.

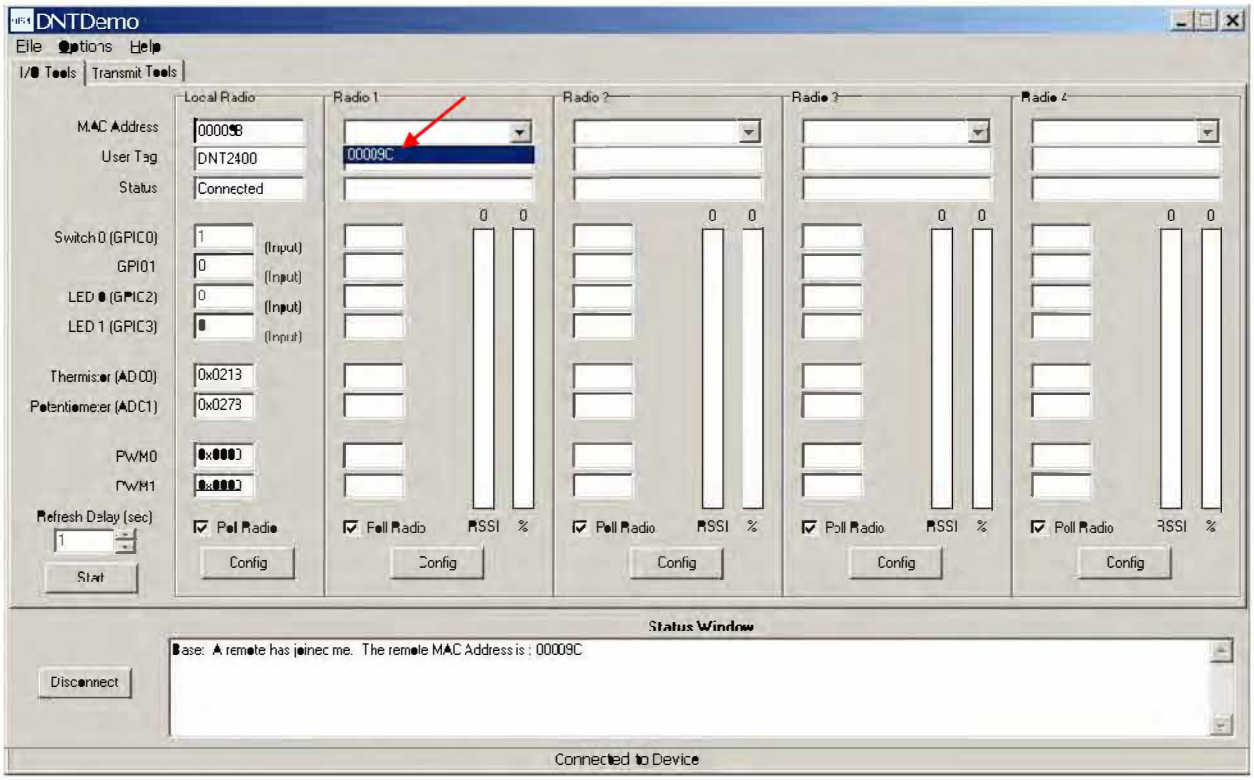


Figure 5.6.1.3

The Demo 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 5.6.1.4. Adjusting the large pot on the Remote can be observed on the *Potentiometer (ADC1)* row.

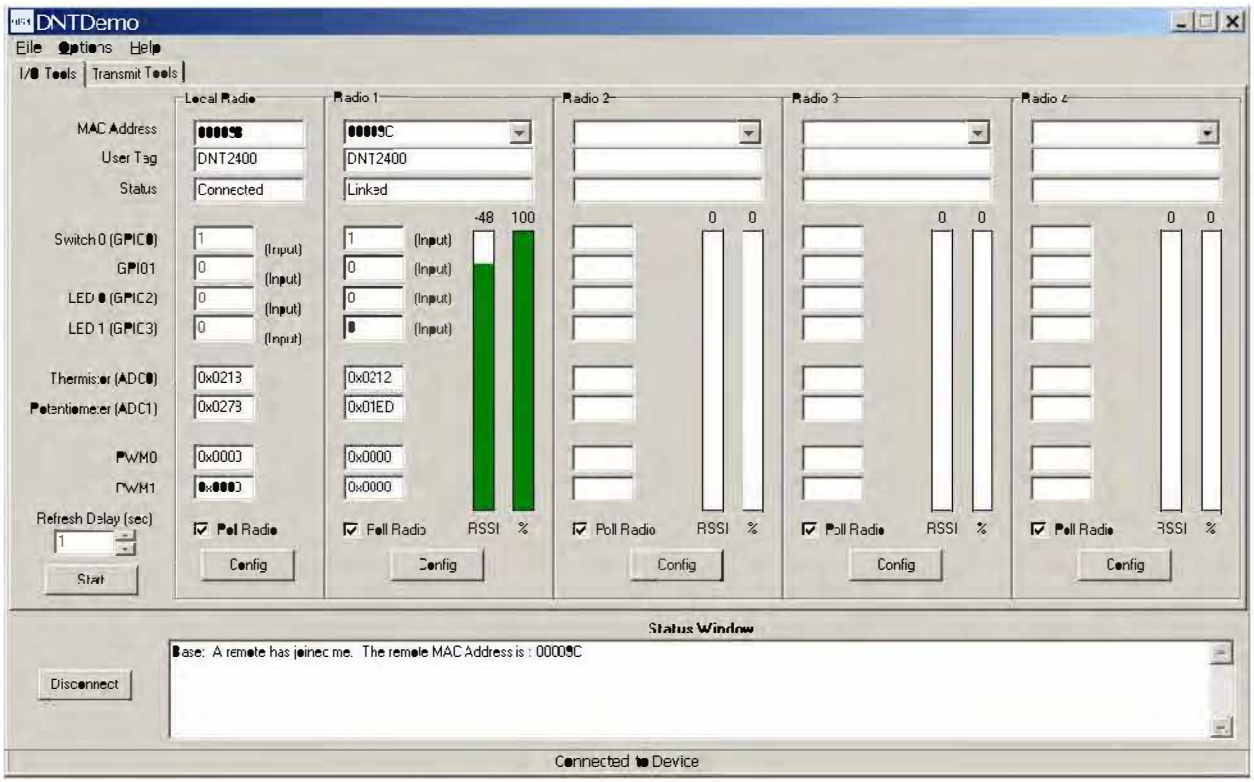


Figure 5.6.1.4

To perform serial data loop back testing with the kit, move the two jumpers on the Remote board labeled *Ext_TX* and *Ext_RX* to connect the center and right header pins. This disconnects the module's TX and RX pins from the USB and RS-232C circuits. Use a banana clip or other short jumper to connect together the two pins on the header labeled J11 (*Ext_MICRO)*. See Figure 5.6.1.5. Attempting loop back testing by connecting Pins 2 and 3 of the DB9 serial connector can cause erratic behavior due to noise coupling from the serial TX and RX lines into the weakly pulled up flow control lines on the board.

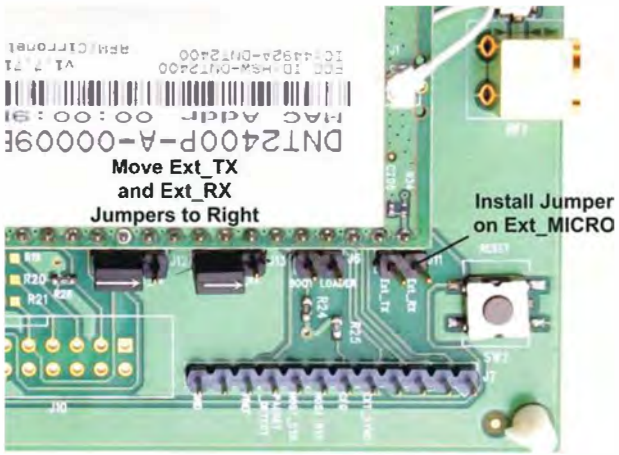


Figure 5.6.1.5

In order to turn off the DCD LED (D4) on the development board in sleep mode, the *GPIO_ SleepState* parameter in the DNT2400P has been set to 0xC0 rather than the factory default value of 0x00. If the DNT2400P is reset to its factory defaults, the DCD LED with remain on in sleep mode until the *GPIO_ SleepState* is set to 0xC0. See Section 4.2.7 for additional information on the *GPIO_SleepState* parameter.

If any difficulty is encountered in setting up the DNT2400DK development kit, contact RFM's module technical support group. The phone number is +1.678.684.2000. 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@rfm.com.

5.6.2 Serial Communication and Radio Configuration

Connect PCs to both the Base and the Remote for serial communication testing. 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 5.6.2.1.

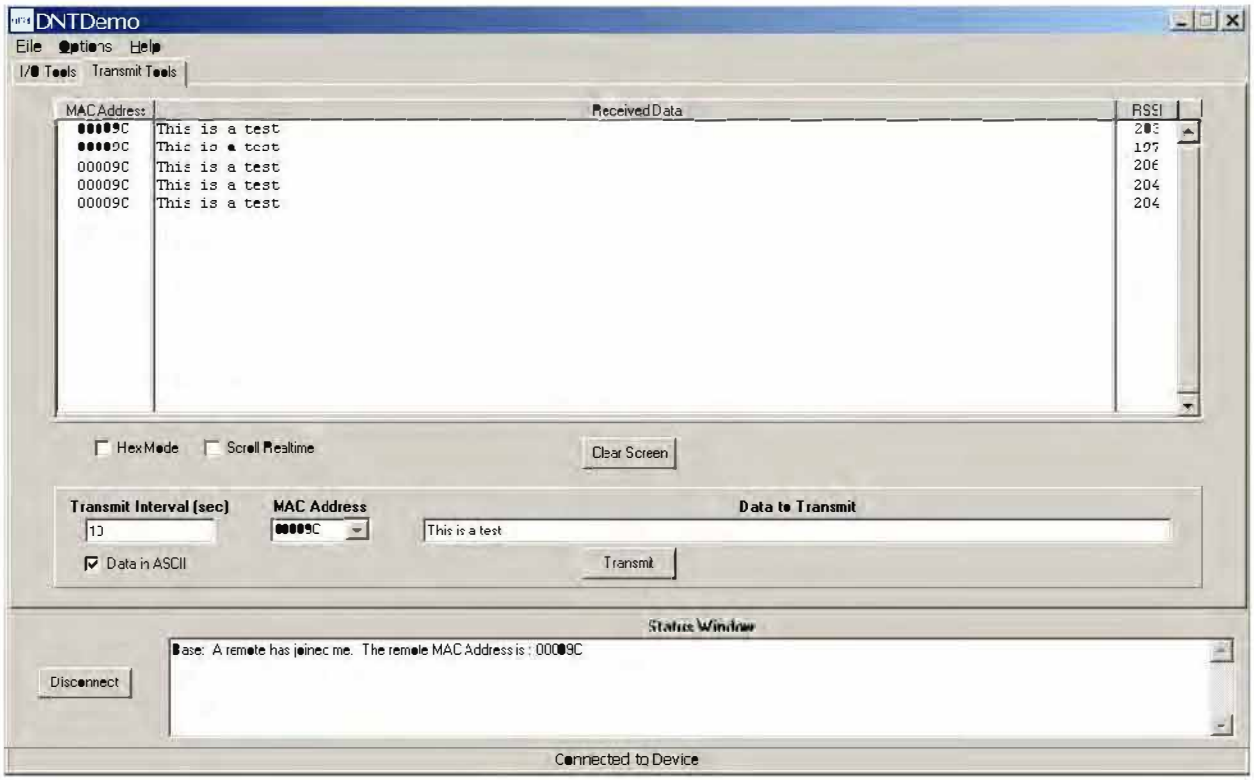


Figure 5.6.2.1

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 always 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, the data on the next tab corresponding to Bank 7, the data on the following two tabs corresponding to Bank 8, and the data on the last tab corresponding to Bank 9.

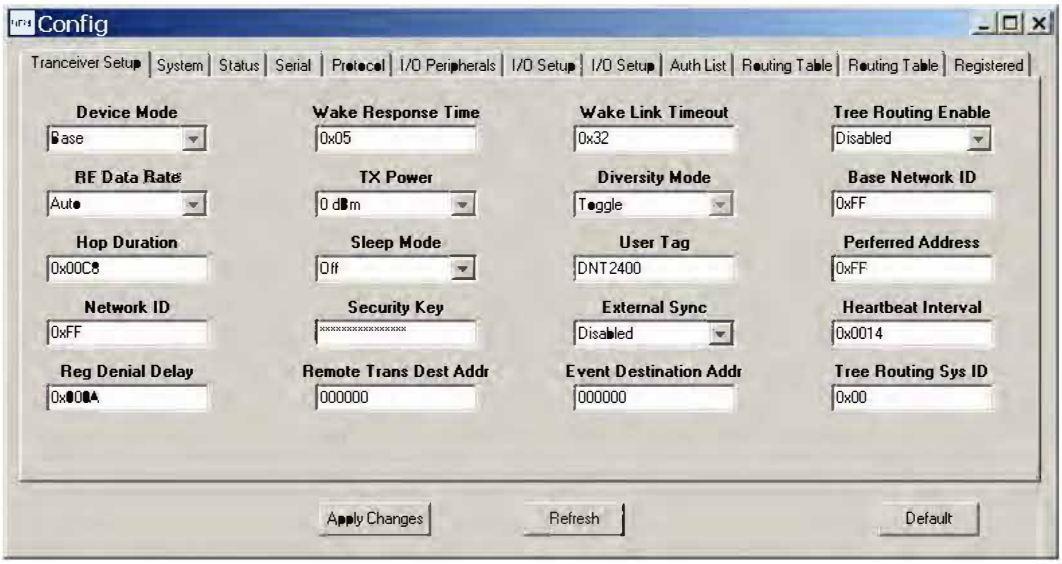


Figure 5.6.2.2

The *Transceiver Setup* Tab is shown in Figure 5.6.2.2 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 Demo automatically reorders multi-byte data to and from Little-Endian order when building or interpreting messages.

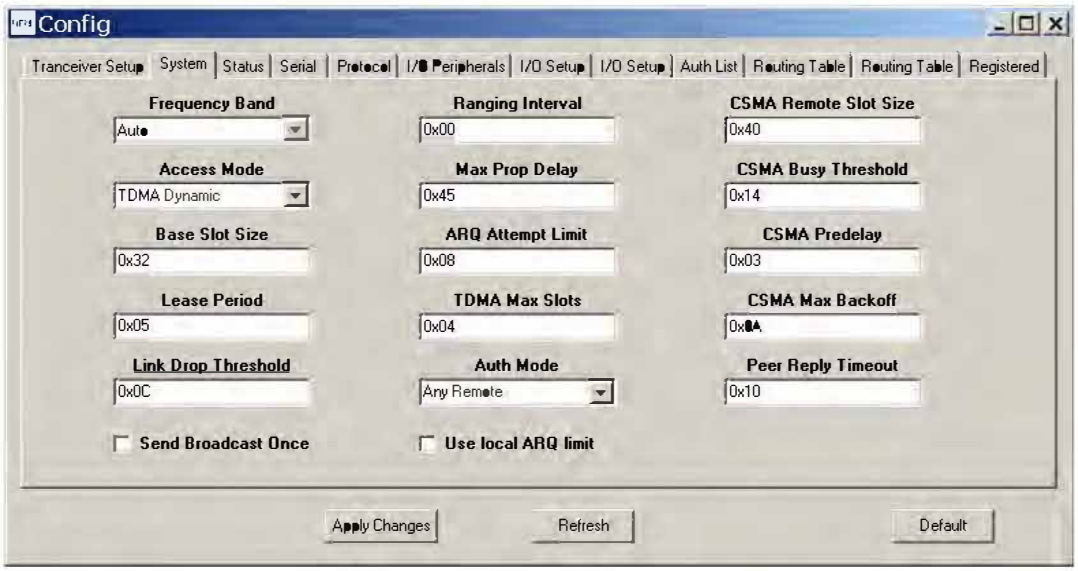


Figure 5.6.2.3

Figure 5.6.2.3 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 menus 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 *ARQ_Mode,* which applies to both the base and the remotes.

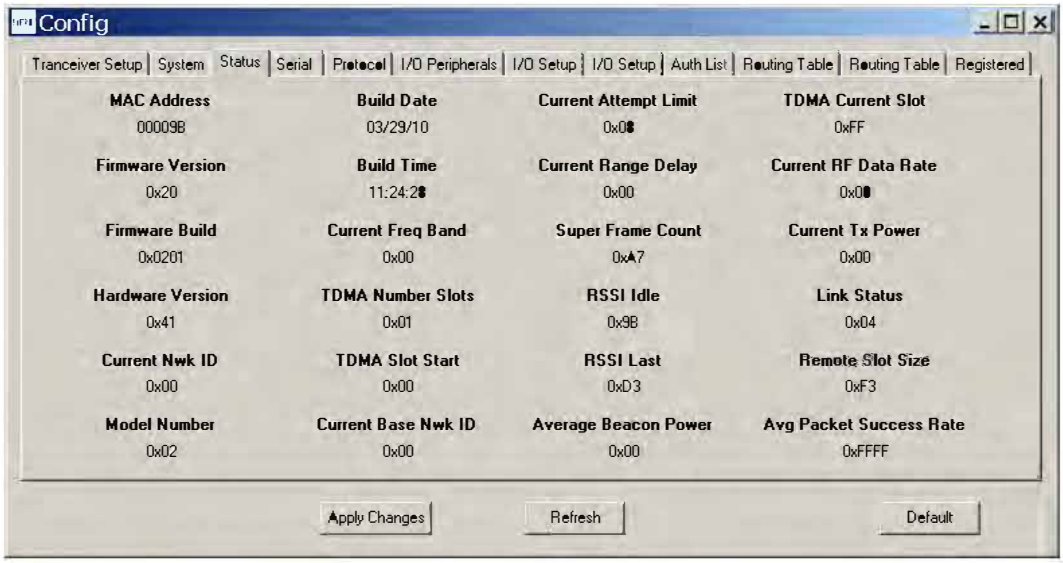


Figure 5.6.2.4

Figure 5.6.2.4 shows the *Status* tab contents, corresponding to Bank 2. Note the *Status* tab contains read-only parameters.

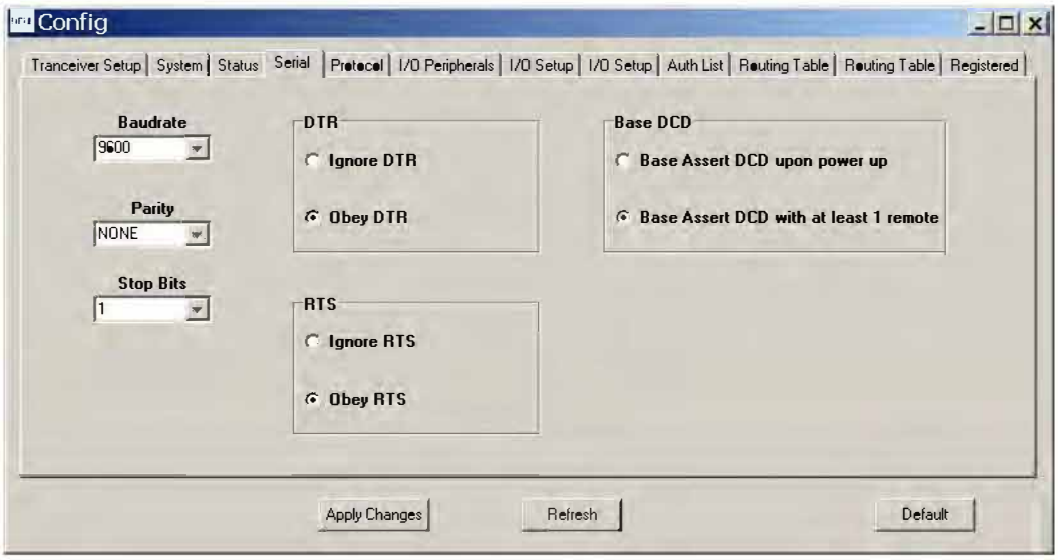


Figure 5.6.2.5

Figure 5.6.2.5 shows the *Serial* tab contents corresponding to Bank 3. The values shown are the defaults for serial port operation.

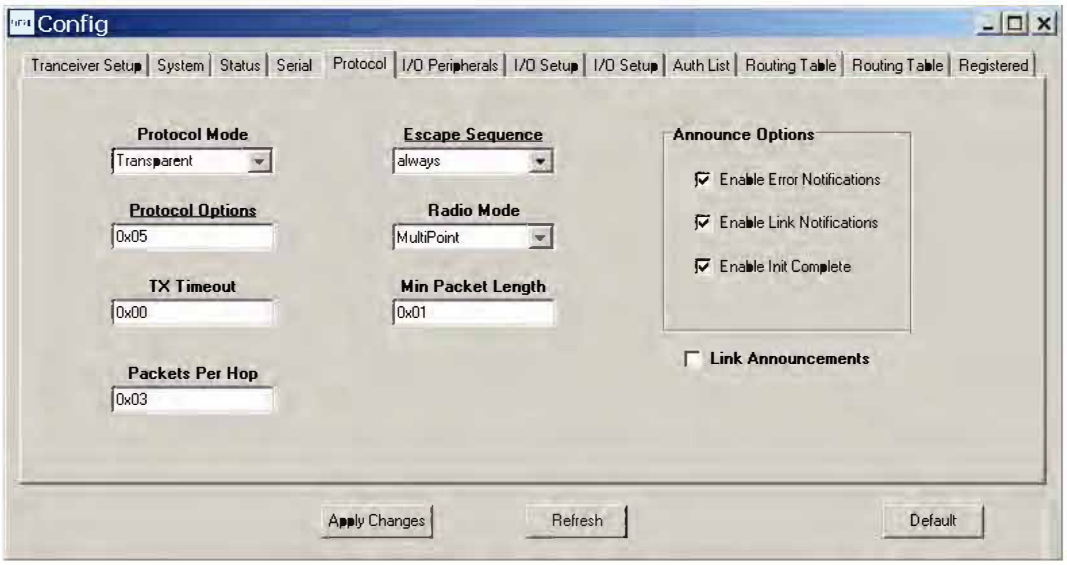


Figure 5.6.2.6

Figure 5.6.2.6 shows the *Protocol* tab contents, corresponding to Bank 4. Transparent data serial communication is currently chosen.

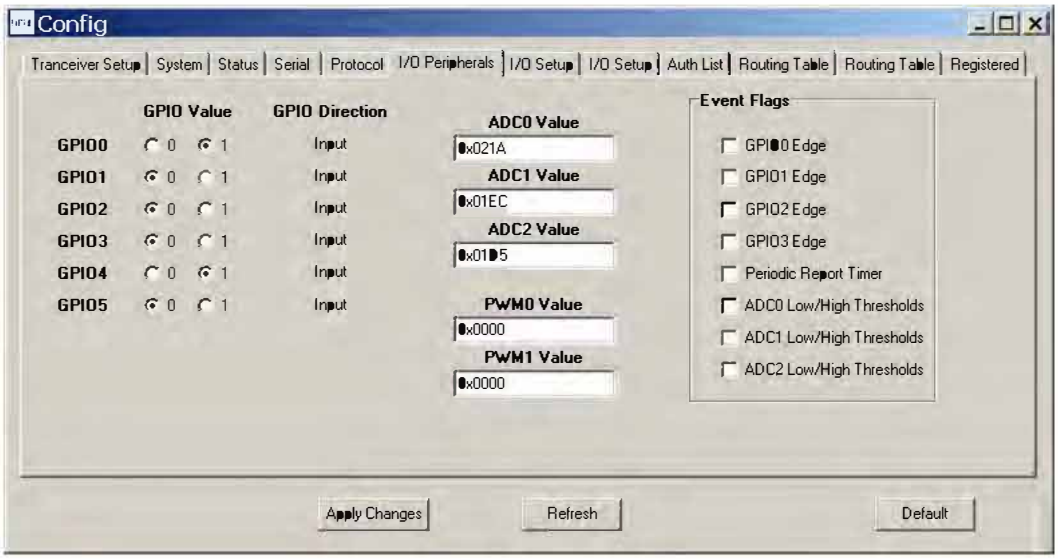


Figure 5.6.2.7

Figure 5.6.2.7 shows the *I/O Peripherals* tab contents, corresponding to Bank 5. GPIO ports 1 through 5 are logic low, GPIO port 0 is logic high. The 10-bit ADC input readings and PWM output settings are given in *Big-Endian* byte order. Event flags are presented on the right side of the window.

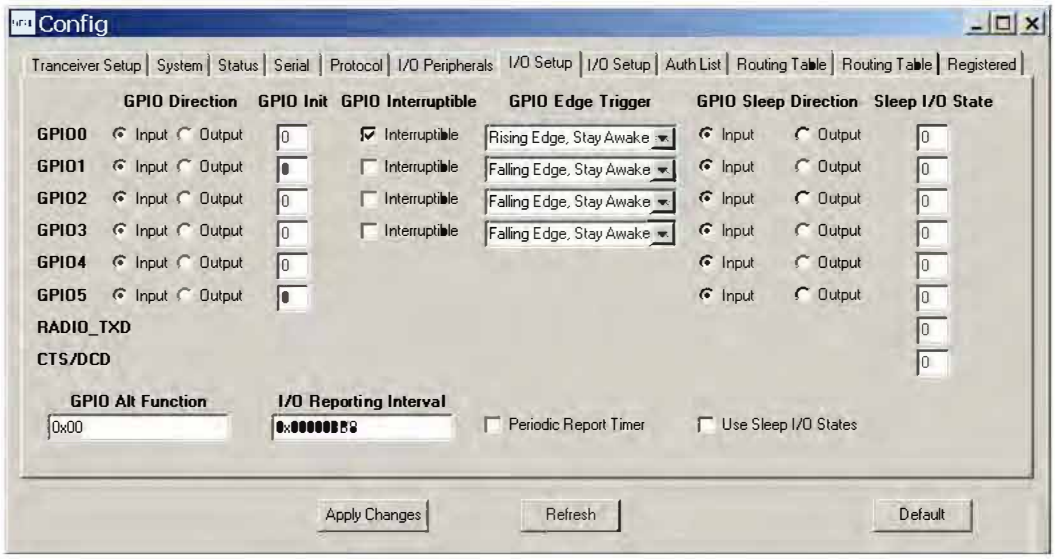


Figure 5.6.2.8

Figure 5.6.2.8 shows the first *I/O Setup* tab contents, corresponding to Bank 6 GPIO parameters. 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. GPIO alternate function, periodic I/O reporting, reporting interval and enable/disable sleep I/O states can also be specified under this tab.

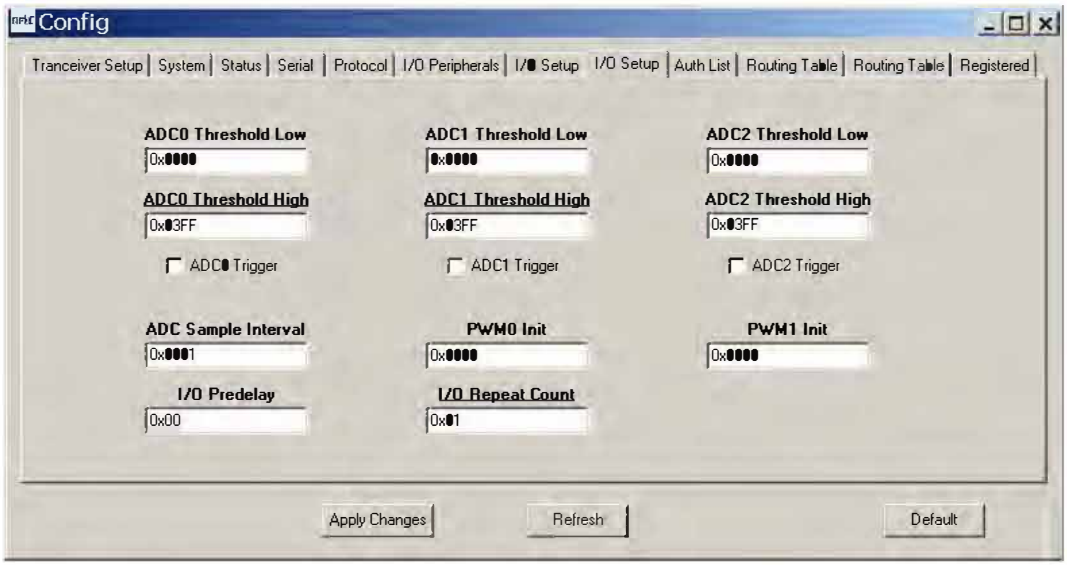


Figure 5.6.2.9

Figure 5.6.2.9 shows the second *I/O Setup* tab contents, corresponding to Bank 6 ADC input and PWM output parameters. The ADC sampling interval, high and low thresholds for event reporting and event reporting triggers on each ADC channel can be set, along with the start-up output values for each PWM (DAC) channel. The event reporting predelay and repeat count parameters can also be set from this tab.

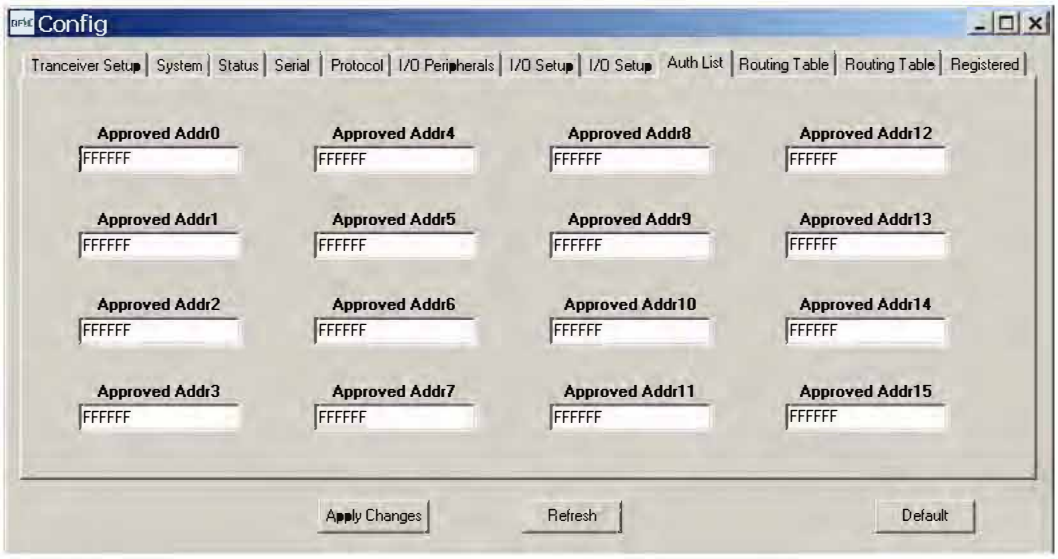


Figure 5.6.2.10

Figure 5.6.2.10 shows the *Auth List* tab, where the MAC addresses of the remotes authorized to join the network in *AuthMode* 1 are input into Bank 7.

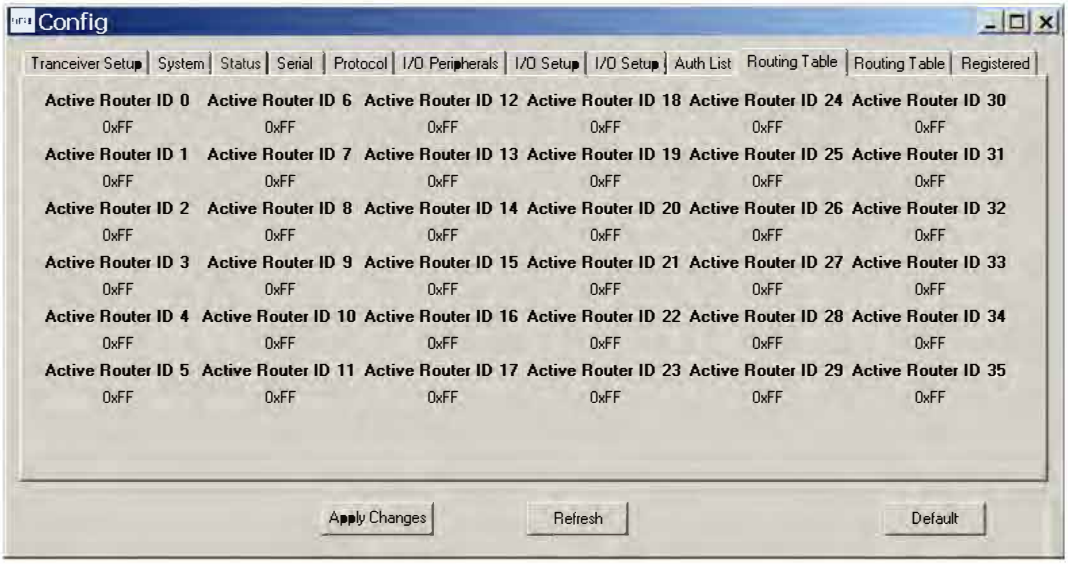


Figure 5.6.2.11

Figure 5.6.2.11 shows the first *Routing Table* tab, which displays part of the contents of Bank 8. This bank contains the tree-routing active router ID table, which is maintained by a base for its system. It describes the organization of all active routers in the system. This table is used by the base and the routers to determine which direction to send a packet.

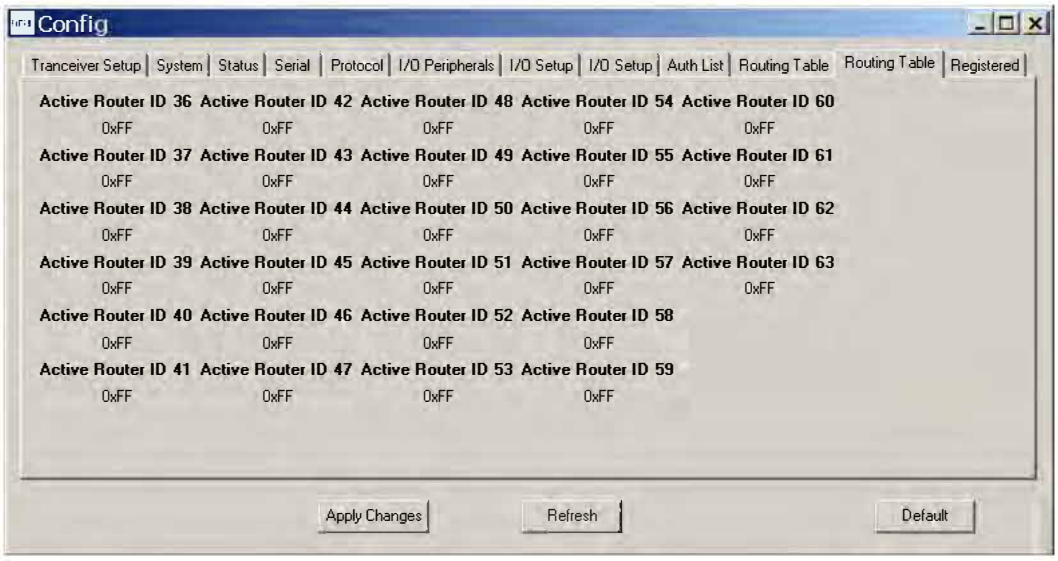


Figure 5.6.2.12

Figure 5.6.2.12 shows the second *Routing Table* tab, which displays the rest of the contents of Bank 8.

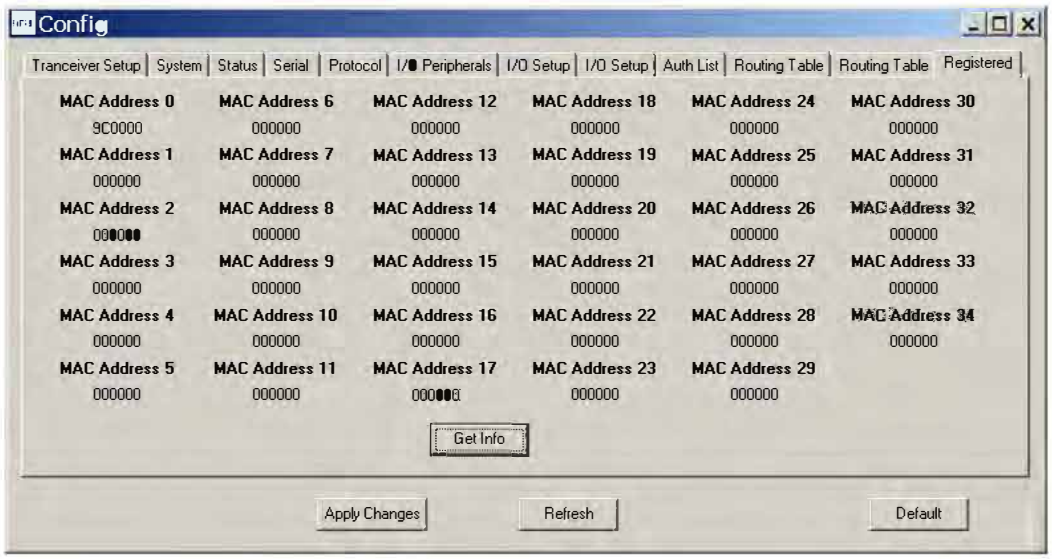


Figure 5.6.2.13

Figure 5.6.2.13 shows the *Registered* tab, which displays the first seven parameters in Bank 9. This bank holds the MAC addresses of all radios registered to a base or router. Up to 126 MAC addresses can be registered. Each bank parameter can hold up to five MAC addresses, with each MAC address containing three bytes. Three-byte segments in a parameter not holding a MAC address with hold a null address: 0x000000. In a remote, this bank will contain only null addresses.

5.7 DNT Wizard Utility Program

DNT Wizard is a complementary program to DNTDemo that emphasizes the details of serial communication between a DNT2400 and its host computer. The DNT Wizard start-up window is shown in Figure 5.7.1.

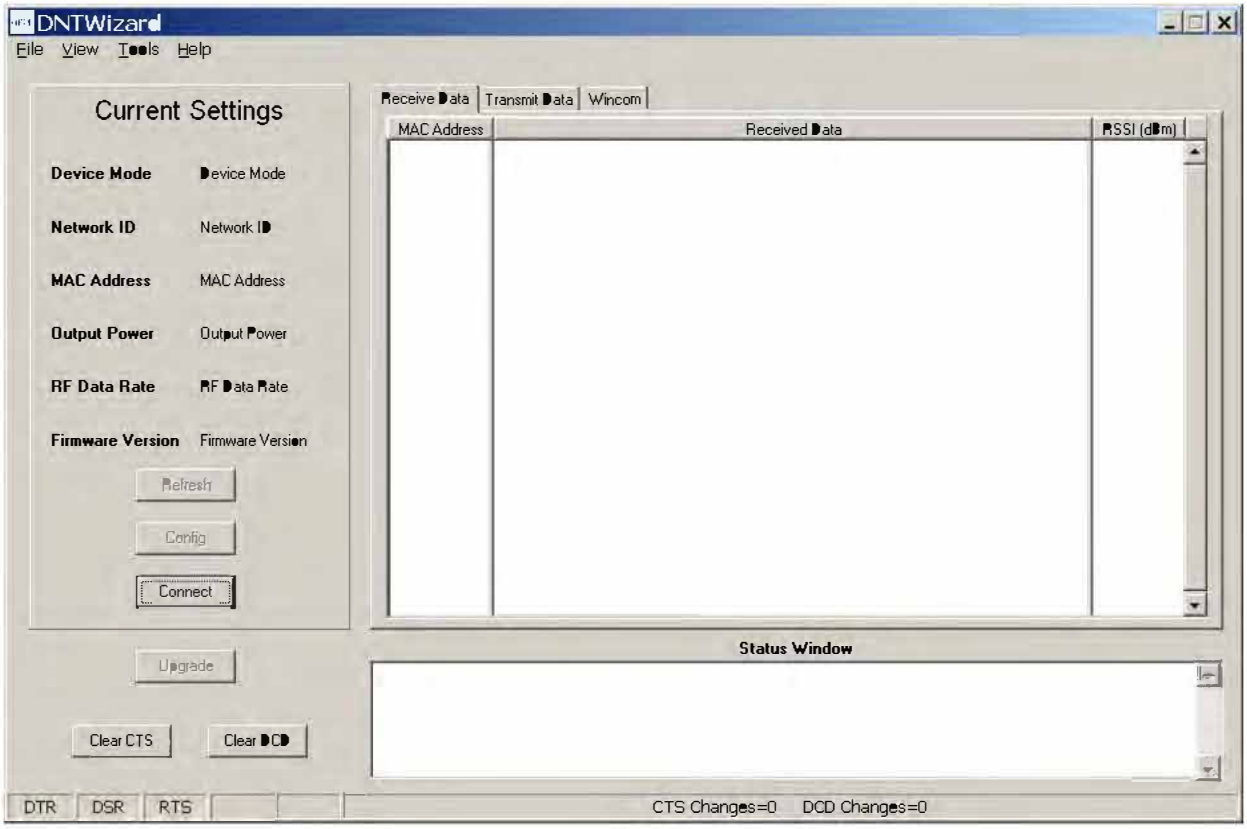


Figure 5.7.1

DNT Wizard uses three Function keys:

F1 toggles the SLEEP/DTR input to the DNT2400 on and off. This allows the DNT2400 to be reset.
F2 toggles the RTS input to the DNT2400 on and off, providing manual flow control.
F6 toggles test transmissions on and off. The test message is “This is a test”.

DNT Wizard also includes the ability to log messages to and from the DNT2400. This feature is especially useful in confirming the format of protocol command and reply messages. Control of the log file is under the *View* menu. Logging is enabled by default. The log file created is *logfile.dat*, and is in ASCII text format. An example log is shown at the end of this section.

Click on *Connect* to open the *Select Comm Port Settings* dialog box, as shown in Figure 5.7.2. Set the baud rate to 9600 (9.6 kb/s). 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.

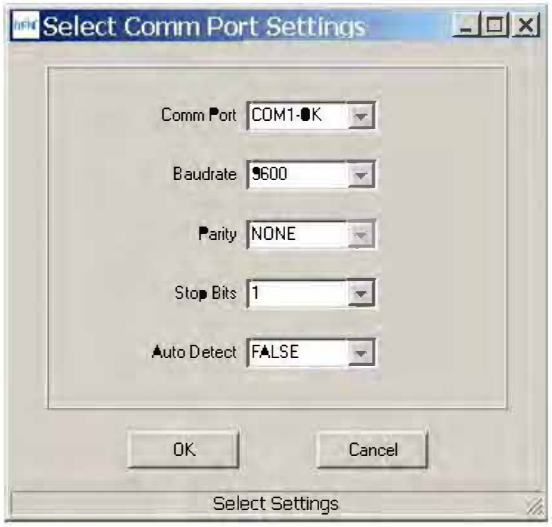


Figure 5.7.2

At this point the Wizard will collect data from the Base, filling in data under *Current Settings* as shown in Figure 5.7.3. The *Status Window* should also show that the Remote has joined the Base.

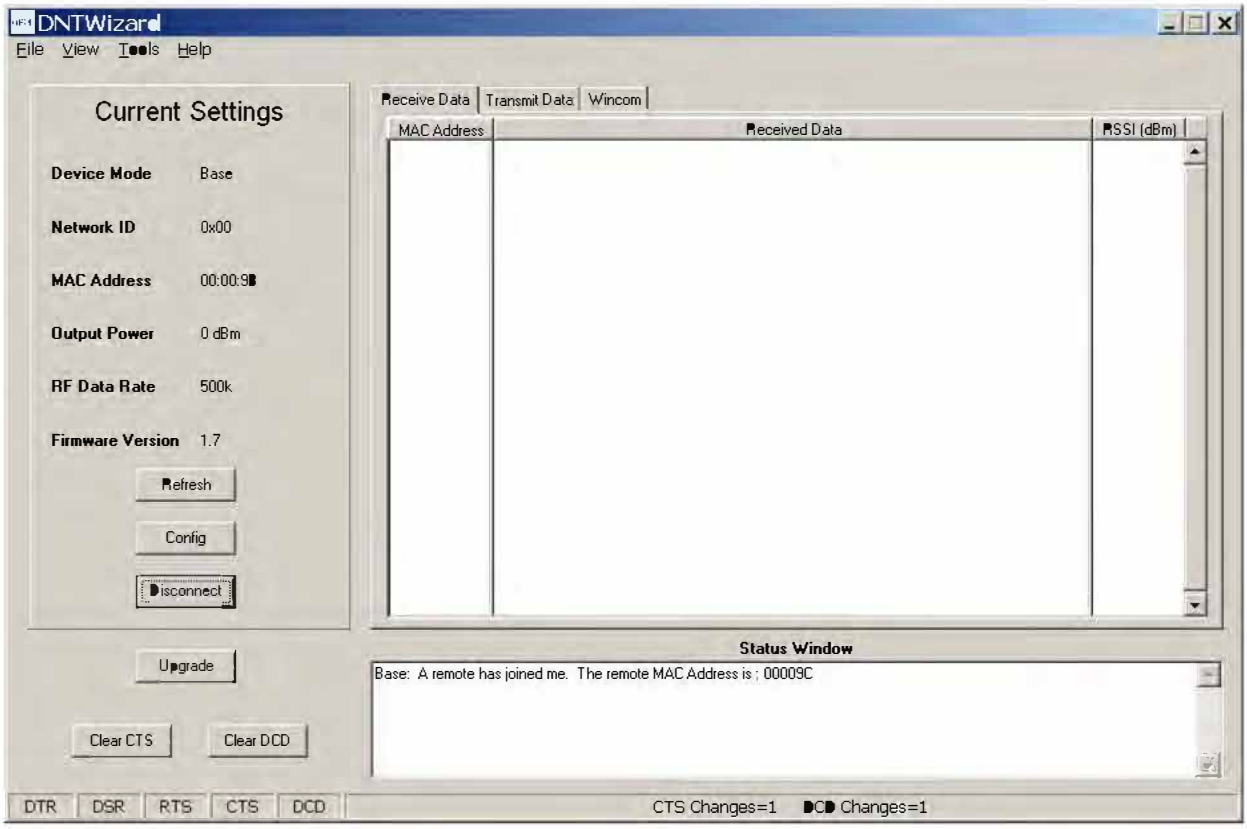


Figure 5.7.3

There are three tabs on the DNT Wizard main window: *Receive Data*, *Transmit Data* and *Wincom*.

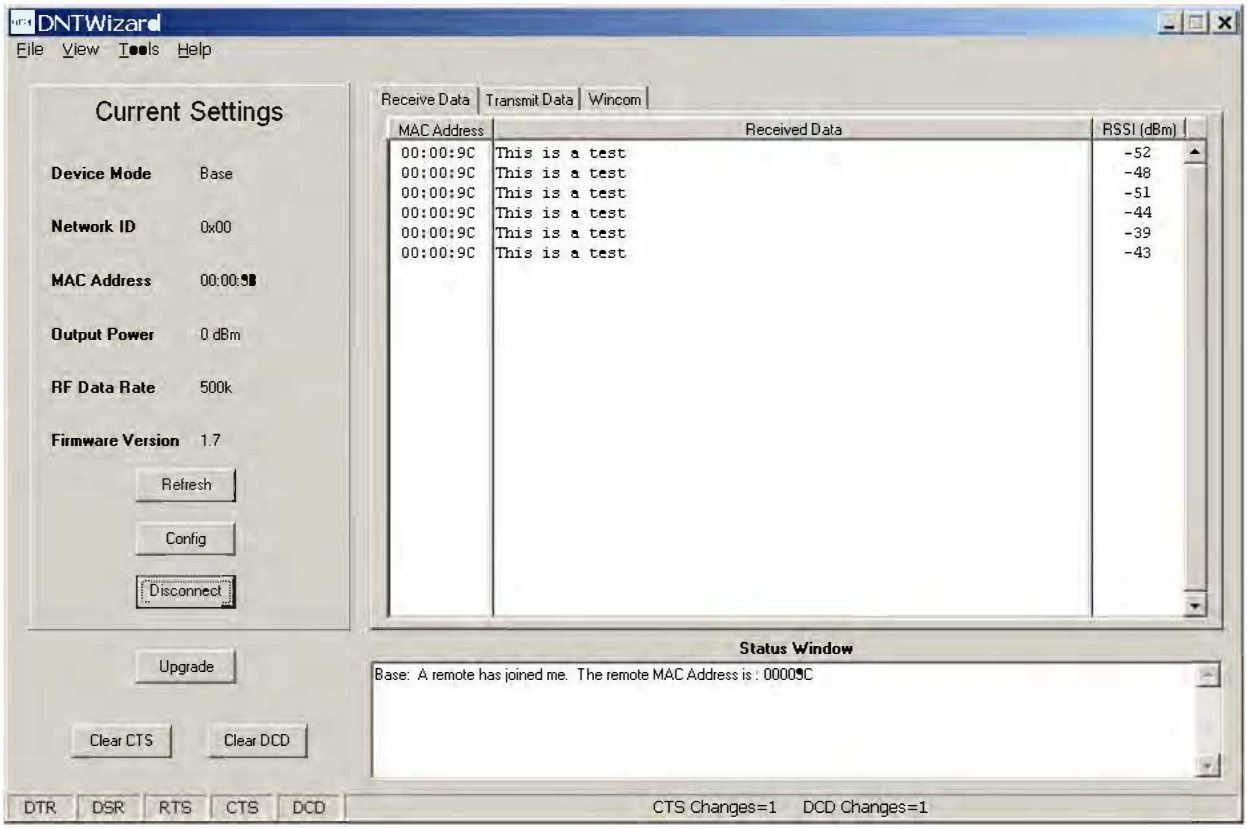


Figure 5.7.4

Received messages are displayed in the *Receive Data* tab, along with the MAC address of the sender and the RSSI (signal strength) of the received message in dBm. See Figure 5.7.4.

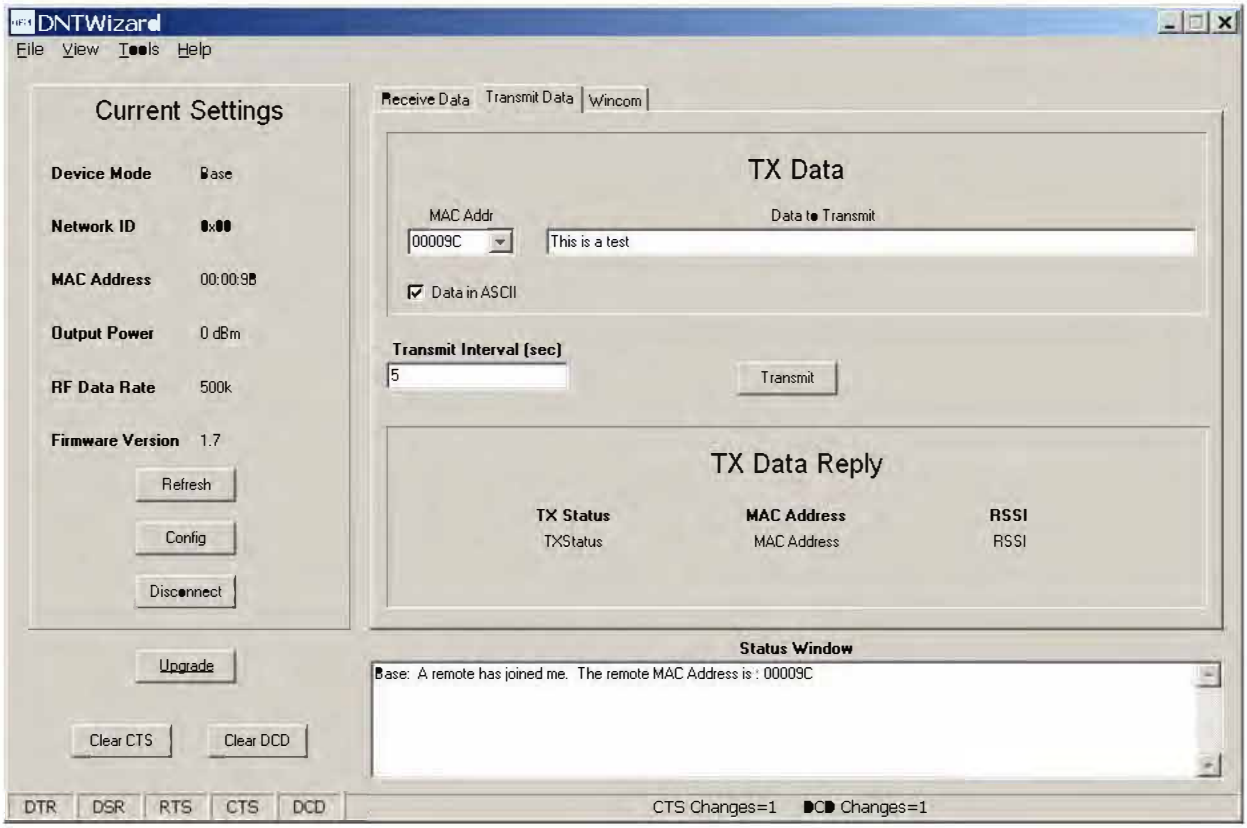


Figure 5.7.5

The *Transmit Data* tab is shown in Figure 5.7.5. The message to send is input in the *Data to Transmit* text box. The address to send the message can be chosen or input in the *MAC Addr* drop-down box. ***Note that the MAC address a remote uses for the base is always 0x000000***. If the *Transmit Interval* is set to 0, the message is sent once each time the *Transmit* button is clicked. When the *Transmit Interval* is set to an integer greater than zero, the message will sent at the beginning of each interval until the *Stop* button (was *Transmit* button) is clicked. The status of each transmission is shown below *TX Data Reply*. Figure 5.7.5 shows that the transmitted message was ACKed, with the received signal strength of the ACK -66 dBm.

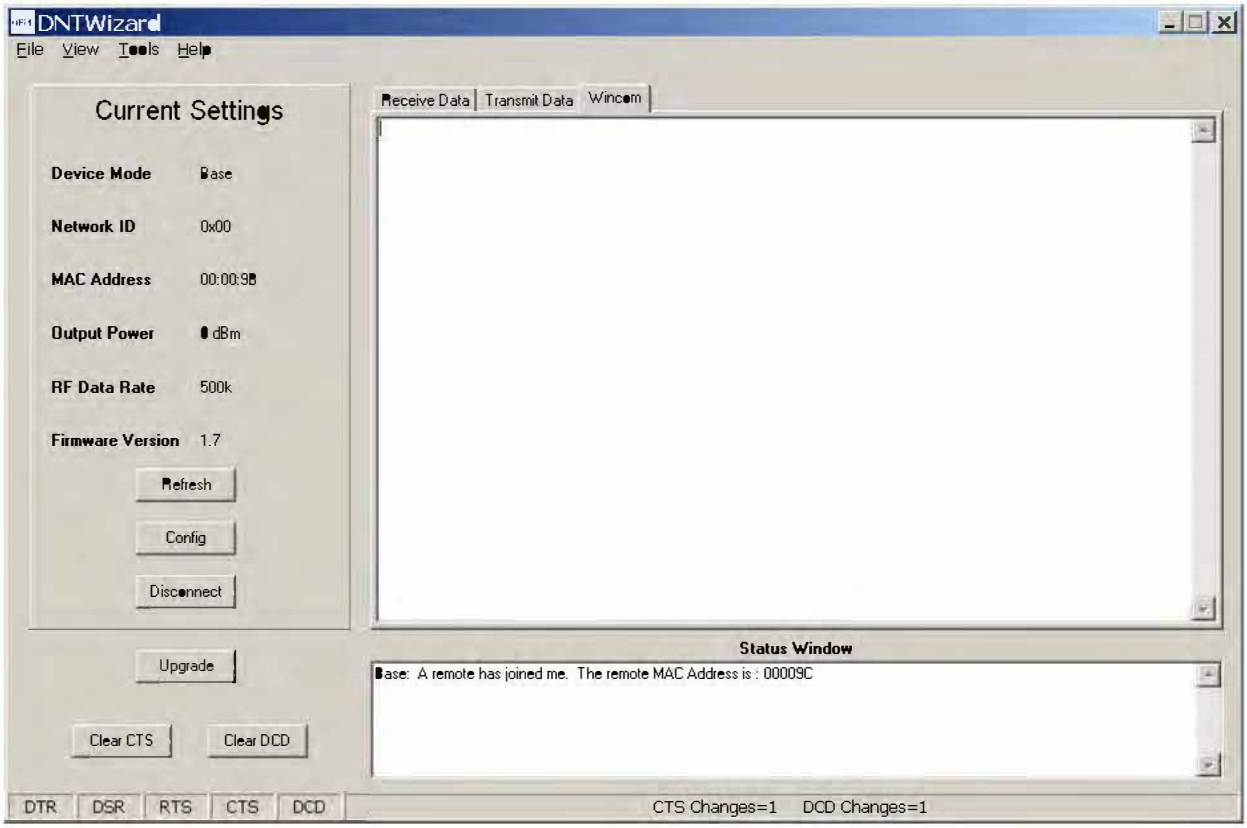


Figure 5.7.6

As shown in Figure 5.7.6, the *Wincom* tab provides the basic functionality of a serial terminal program. Messages typed in are sent, and messages received are appended to the bottom of the on-screen text.

The multi-tab *Configuration* window is 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, the data on the next tab corresponding to Bank 7, the data on the following two tabs corresponding to Bank 8, and the data on the last tab corresponding to Bank 9.

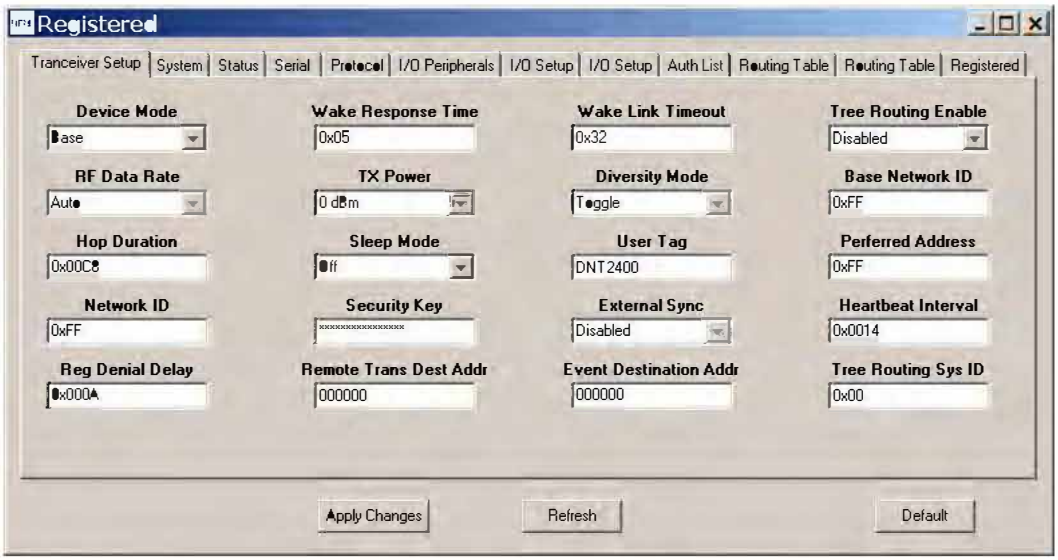


Figure 5.7.7

The *Configuration* window in the DNT Wizard is identical the *Configuration* window in the DNT Demo. See Figures 5.6.2.2 through 5.6.2.13 for *Configuration* window details.

Details of the *File*, *View* and *Tools* menus are shown in Figure 5.7.8.

Figure 5.7.8

The *Tools* menu contains two very useful items. *Packet Builder* opens the window shown in Figure 5.7.9. On the left, the *Packet Type* drop-down box provides a selection of all packet types used in the DNT2400 protocol. On the right, the reply packet types are presented.

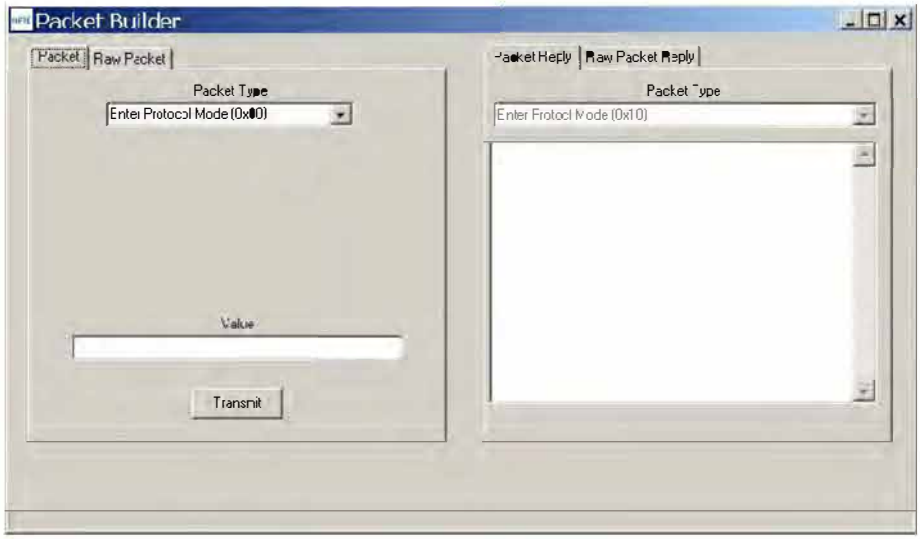


Figure 5.7.9

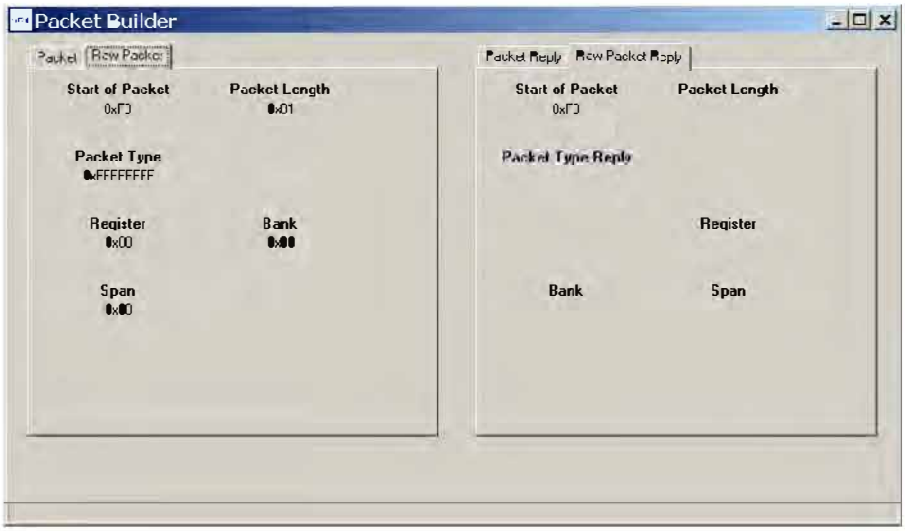


Figure 5.7.10

Figure 5.7.10 shows the *Raw Packet* and *Raw Packet Reply* tabs. Components of the protocol messages are shown in hexadecimal format.

Log Data controls the log file. Logging is enabled by default. The log file created is *logfile.dat*, and is in ASCII text format. An example log is shown below:

```
New Data----------------------------------------
8/12/2009  |  3:50:24 PM
DNTWizard Version 1.04
Sent Data : FB 07 00 44 4E 54 35 30 30 (Enter Protocol Mode)
Recv Data : FB 01 10 (Enter Protocol Mode Reply)
Sent Data : FB 04 03 00 00 31 (Get Register)
Recv Data : FB 35 13 00 00 31 01 FF CB 00 FF 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 00
05 32 00 00 02 02 44 4E 54 32 34 30 30 00 00 00 00 00 00 00 00 00 0A 00 00 00 00 (Get Register
Reply)
Recv Data : FB 06 15 01 FF FF FF 7F (TX Data Reply)
Sent Data : FB 04 03 00 01 10 (Get Register)
Recv Data : FB 14 13 00 01 10 FF 02 32 05 01 08 04 03 0A 45 0C 32 14 00 00 10 (Get Register
Reply)
Sent Data : FB 04 03 00 02 28 (Get Register)
Recv Data : FB 2C 13 00 02 28 9B 00 00 00 00 00 00 04 F3 00 00 FF 41 17 02 01 1B 01 98 00 00 08
00 30 38 2F 31 31 2F 30 39 31 37 3A 30 33 3A 30 33 02 (Get Register Reply)
Sent Data : FB 04 03 00 03 04 (Get Register)
Recv Data : FB 08 13 00 03 04 30 00 00 07 (Get Register Reply)
Sent Data : FB 04 03 00 04 08 (Get Register)
Recv Data : FB 0C 13 00 04 08 00 05 00 01 07 00 02 00 (Get Register Reply)
Sent Data : FB 04 03 00 05 12 (Get Register)
Recv Data : FB 16 13 00 05 12 01 00 00 00 00 00 19 02 78 02 1D 02 00 00 00 00 00 00 (Get Register
Reply)
Sent Data : FB 04 03 00 06 1E (Get Register)
Recv Data : FB 22 13 00 06 1E 00 00 00 01 00 00 C0 00 00 00 00 01 00 00 00 FF 03 00 00 FF 03 00
00 FF 03 01 B8 0B 00 00 (Get Register Reply)
Sent Data : FB 04 03 00 07 30 (Get Register)
Recv Data : FB 34 13 00 07 30 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF (Get Register
Reply)
Recv Data : FB 07 27 A2 9C 00 00 01 00 (Announce)
Recv Data : FB 13 26 9C 00 00 CC 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Recv Data : FB 13 26 9C 00 00 D0 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Recv Data : FB 13 26 9C 00 00 CD 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Recv Data : FB 13 26 9C 00 00 D4 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Recv Data : FB 13 26 9C 00 00 D9 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Recv Data : FB 13 26 9C 00 00 D5 54 68 69 73 20 69 73 20 61 20 74 65 73 74 (RX Data)
Sent Data : FB 01 01
(Exit Protocol Mode)
Recv Data : FB 01 11 (Exit Protocol Mode Reply)
```

The log file is especially useful in confirming the format of specific protocol commands and replies.

5.8 DNT2400 Interface Board Features

The location of LEDs D1 through D4 and jumper pin set J14 are shown in Figure 5.8.1.

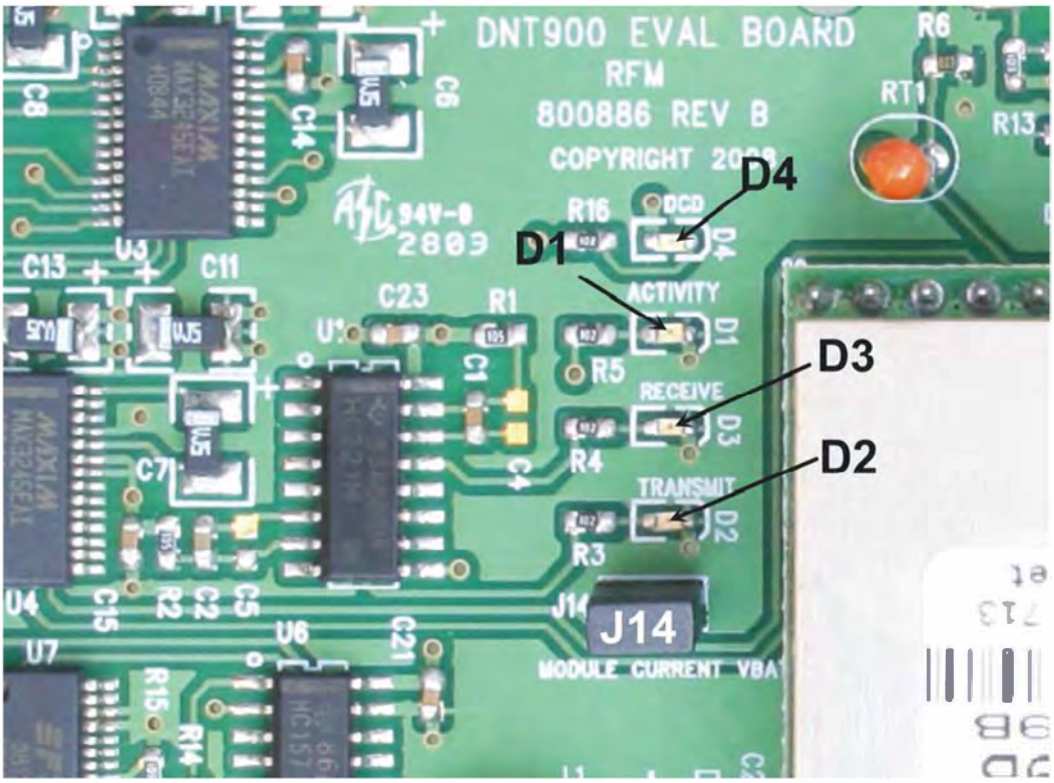


Figure 5.8.1

Amber DCD LED D4 illuminates on a remote to indicate it is registered with the base and can participate in RF communications. DCD LED D4 illuminates on the base when one or more remotes are registered to it, unless the base has been configured to assert DCD on power up. In this case it will be on as long as the dev board is powered. Green Activity LED D1 illuminates when transmitting or receiving RF data. Red Receive LED D3 illuminates when sending received data through the serial port to the PC. Green Transmit LED D2 illuminates when the PC sends data through the serial port to be transmitted. Jumper pin set J14 is provided to allow measurement of the DNT2400P current. For normal operation J14 has a shorting plug installed.

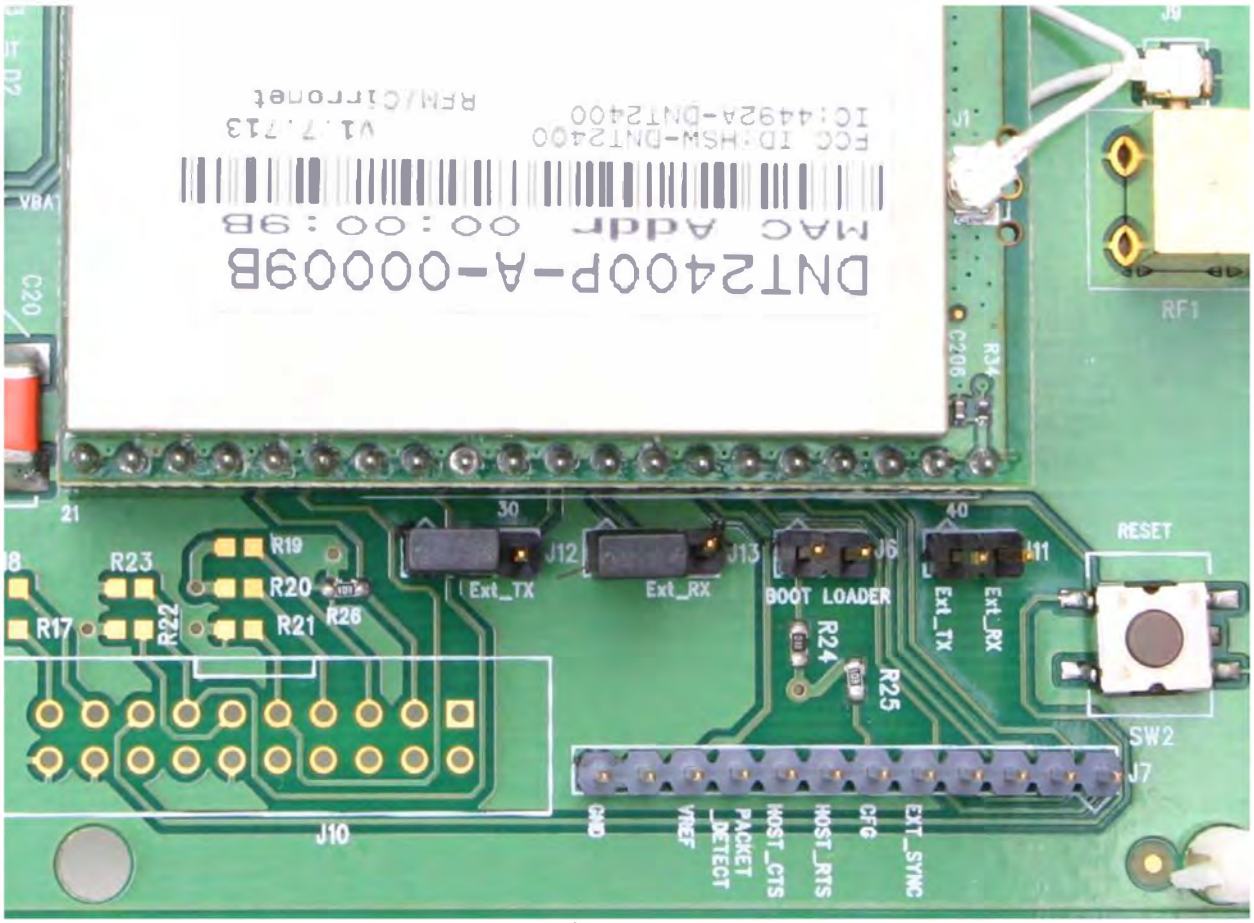


Figure 5.8.2

Figure 5.8.2 shows the connectors to the right of the DNT2400P mounting socket. Jumper pin sets J12 and J13 normally have shorting plugs installed as shown in Figure 5.8.2, which connects the DNT2400P RADIO_TXD and RADIO_RXD pins to the respective serial data lines on the evaluation board. It is possible to connect directly to RADIO_TXD and RADIO_RXD by moving the jumpers over. In this case, J11-1 is the input for transmitted data and J11-2 is the output for received data. *Note this a 3 V logic interface.* Placing a shorting plug on jumper pin set J6 allows the DNT2400P to be powered up in boot loader mode. This is used for factory code loads and functional testing. The DNT2400 has its own boot loader utility that allows the protocol firmware to be installed with a terminal program that supports YMODEM. Pin strip J7 provides access to various DNT2400 pins as shown on the silkscreen. Pressing switch SW2 will reset the DNT2400P.

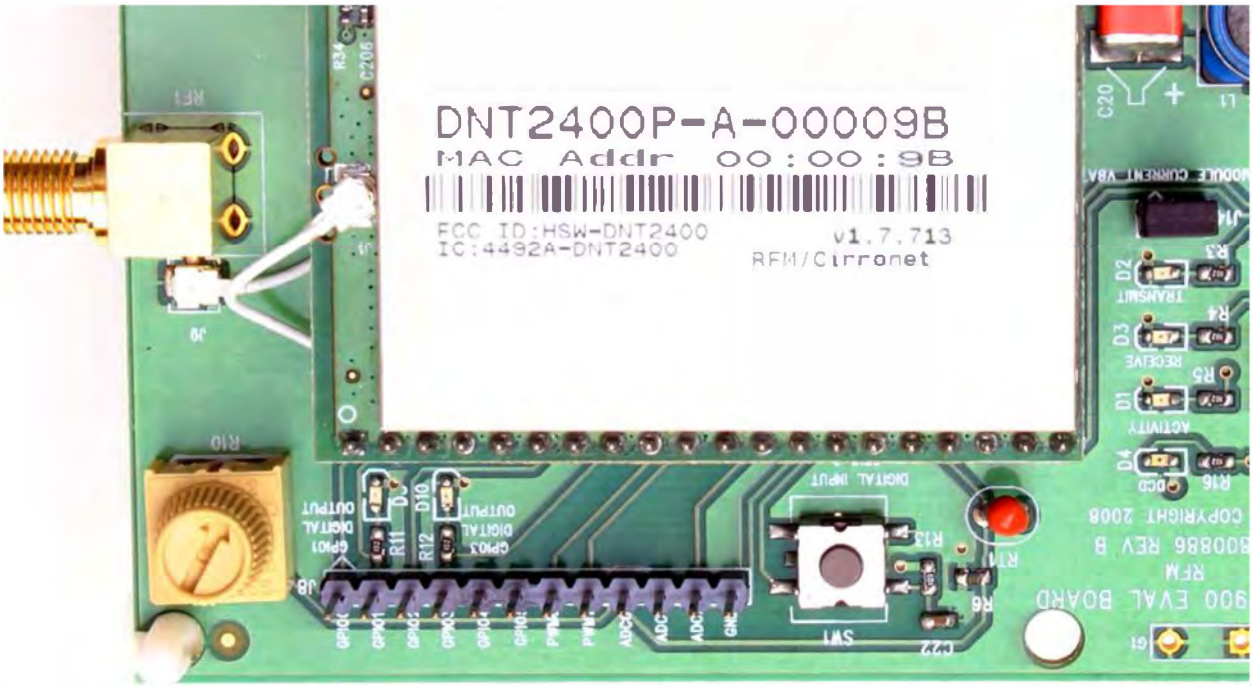


Figure 5.8.3

Figure 5.8.3 shows the connectors to the left of the DNT2400P mounting socket. Pressing switch SW1 switches GPIO0 from logic high to logic low. Pin strip J8 provides access to various DNT2400 pins as shown on the silkscreen. The wiper of pot R10 drives the input of ADC1. Clockwise rotation of the pot wiper increases the voltage. Thermistor RT1 is part of a voltage divider that drives ADC0. LED D5 illuminates when GPIO1 is set as a logic high output. LED D10 illuminates when GPIO3 is set as a logic high. The DNT2400P interface board includes a 5 V regulator to regulate the input from the 9 V wall-plug power supply. Note: do not attempt to use the 9 V wall-plug power supply to power the DNT2400P directly. The maximum allowed voltage input to the DNT2400P is 5.5 V.

6.0 Demonstration Procedure

The procedure below provides a quick demonstration of the DNT2400P using a DNT2400DK development kit:

1. Confirm that each DNT2400P transceiver is installed correctly in an interface board, and that the U.FL jumpers between the DNT2400P radios and the interface boards are installed. Also confirm that a dipole antenna is installed on each interface board, and that J14 has a jumper block installed on each interface board. See Figures 5.4.1, 5.4.2 and 5.4.3 above.

2. Attach the Base (see label on bottom of interface board) to a computer with the DNT Demo program installed.

3. Place the transceiver/interface boards at least 6 feet (two meters) apart.

4. Start the DNT Demo program.

5. Click the *Connect* button on the Demo window. This will open a serial port setup dialog box. Set the baud rate to 9600 (9.6 kb/s) and select the COM port where the Base is connected.

6. At this point the Demo will collect data from the Base, filling in data in the *Local Radio* column on the Demo window. The *Status Window* should also show that the Remote has joined the Base.

7. Click on the drop-down box at the top of the *Radio 1* column and select the *MAC Address* for the Remote. Next press the *Start* button using the default 1 s *Refresh Delay*.

8. The Demo 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. Adjusting the large pot on the Remote can be observed on the *Potentiometer (ADC1)* row.

7.0 Troubleshooting

DNT2400 not responding - make sure SLEEP/DTR is not asserted (logic low) to bring the radio out of sleep mode.

Can not enter protocol mode - make sure the host data rate is correct. The DNT2400 defaults to 9.6 kb/s. If using the *EnterProtocolMode* command, send the complete protocol format for this command.

A remote never detects carrier (DCD) - check that the base is running, and that the remote *InitialNwkID* parameter is the same as the base, or is set to 0xFF. If a remote is operating near the base (less than 6 ft or 2 m), be sure the transmitter power setting on both the base and remote is set to 1 mW. If the remote is operating far from the base, be sure the transmitter power setting on the base and remote is adequate for communication. Also make sure that the security keys are the same.

Carrier is detected, but no data appears to be received - make sure that RTS is asserted to enable receive character flow. Make sure the RF transmit power is not on a high settings if the nodes are close together.

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.

Transmitting terminal flashes (drops) CTS occasionally - this indicates that the transmit buffer is filling up. This most often is caused by using a higher serial port data rate than RF data rate. Adjusting the protocol parameters may increase the network efficiency.

7.1 Diagnostic Port Commands

FSTAT <option> - enables a frequency-ordered channel activity status log.

Available options are:

0 - off
1 - on

For either a base or remote, FSTAT shows the DataTx (data packet transmitted), AckRx (ACK received) and RegRx (registration or renewal request or reply received) activity status on each frequency with ASCII characters as follows:

‘.’ (0x2E) - no activity
‘1’ (0x31) - DataTx activity only
‘2’ (0x32) - AckRx activity only
‘3’ (0x33) - DataTx and AckRx activity
‘4’ (0x34) - RegRx activity only
‘5’ (0x35) - DataTx and RegRx activity
‘6’ (0x36) - AckRx and RegRx activity
‘7’ (0x37) - DataTx, AckRx and RegRx activity

A sample FSTAT output for 37 channel operation is shown below. The status data is order from the lowest operating frequency on the left to the highest operating frequency on the right. An ASCII CR-LF terminates each line. On most frequencies, DataTx and AckRx occur on the same frequency ('3'). Occasionally there is DataTx, AckRx and RegRx activity ('7'), DataTx only activity ('1'), or no activity ('.').

```
333333 33333333 33333373 33333333 3333333
333333 33333333 33333333 33333333 3333333
333333 31333333 33333333 33333333 3333333
23.333 333133.3 3.333323 31333.33 333.333
333333 23333331 33313333 33333333 23.3333
```

INSTR <option>

Available options are:

0 - off
1 - on

For either a base or a remote, INSTR streams instrumentation packets including a time stamp and seven data values. The time stamp is a string of four characters. The first three characters are ASCII representations of hexadecimal numbers. The fourth character is a '>' character. Each time stamp count is 100 µs. The time stamp rolls over to 0x000 after count 0xCE5. The content of the seven data values are detailed below. ASCII representations of the hexadecimal value of each byte are output, separated by space characters:

Byte 0: current channel, 0 to 36
Byte 1: same data as FSTAT 1, except no activity is 0x30
Byte 2: RSSI_Last, RSSI of last RF packet received in dBm, 8-bit signed value
Byte 3: RSSI_Idle, most recent RSSI value on a channel with no expected activity
Byte 4: Range delay estimate, remote only, 3.1 µs/tick
Byte 5: Status bits 1
- b7.. b4 - reserved
- b3 - inbound data RF flow control, 1 = device has disabled RF flow
- b2 - outbound data RF flow control, 1 = base has disabled RF flow
- b1 - /HOST_CTS, status of CTS line, 1 = hold
- b0 - /DCD, link LED on; on a remote indicates registration with base, on the base it indicates at least on remote is registered (default)

Byte 6: Status bits 0
- b7 - CSMA channel busy
- b6 - Parser/CRC error
- b5 - valid RF packet received (any type)
- b4 - DataRx, data packet received
- b3 - RegTx, registration or renewal request or reply transmitted
- b2 - RegRx, registration or renewal request or reply received
- b1 - AckRx, ACK received
- b0 - DataTx, data packet transmitted

A sample INSTR output is shown below.

```
896> 21 01 DA 98 00 01 01
940> 0A 03 DA 9C 00 01 23
9E2> 1D 03 DA 99 00 03 23
A8A> 1E 01 D6 9F 00 03 21
B36> 11 03 D6 99 00 03 23
```

Looking a the first line in detail:

```
896> 21 01 DA 98 00 01 01
```

The time stamp is 0x896>
Byte 0, 0x21, indicates the channel of operation is 33 (decimal)
Byte 1, 0x01, is the same status data as FSTAT data above. In this case, DataTx activity only.
Byte 2, 0xDA, is the last RSSI value, -38 dBm
Byte 3, 0x98, is the RSSI idle value, -100 dBm
Byte 4, 0x00, is the range delay estimate (remote only). The remote is very close to the base.
Byte 5, 0x01, provides the serial port status - /HOST_CTS is high
Byte 6, 0x01, provides the communication status - DataTx active only. Note that this byte provides additional status bytes compared to byte index 1.

8.0 Appendices

8.1 Ordering Information

DNT2400C: transceiver module for solder-pad mounting
DNT2400P: transceiver module for pin-socket mounting

8.2 Technical Support

For DNT2400 technical support call RFM at (678) 684-2000 between the hours of 8:30 AM and 5:30 PM Eastern Time.

8.3 DNT2400 Mechanical Specifications

DNT2400C Outline and Mounting Dimensions

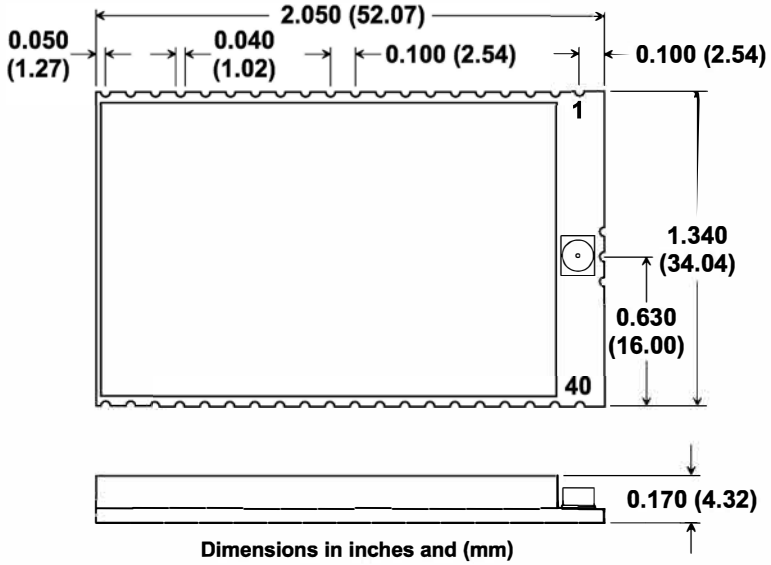


Figure 8.3.1

DNT2400C PCB Footprint

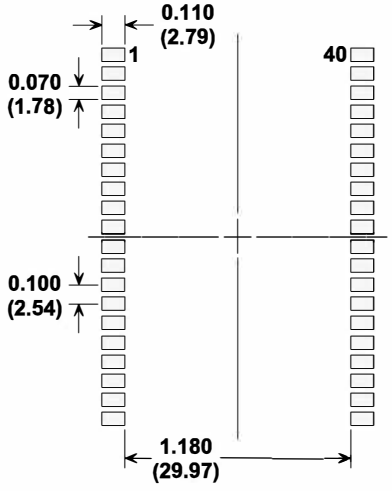
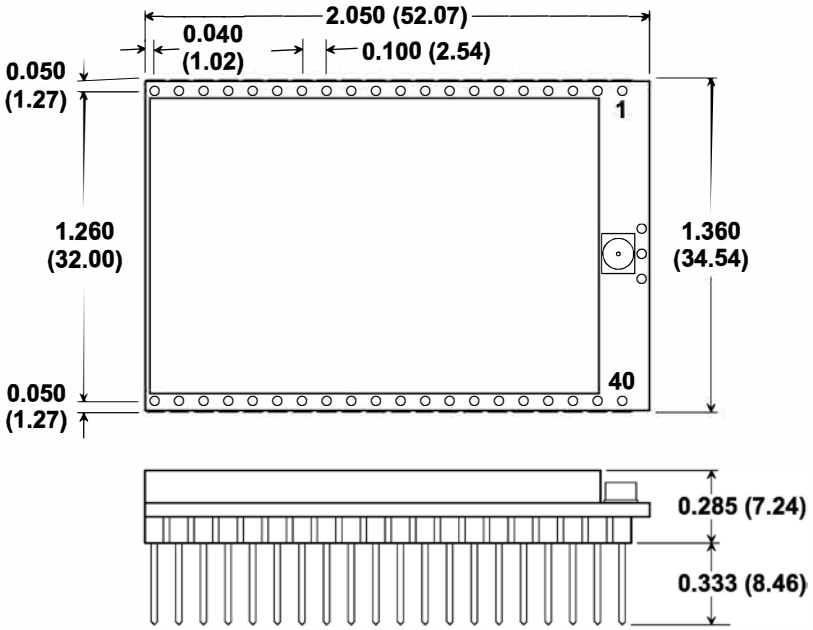


Figure 8.3.2

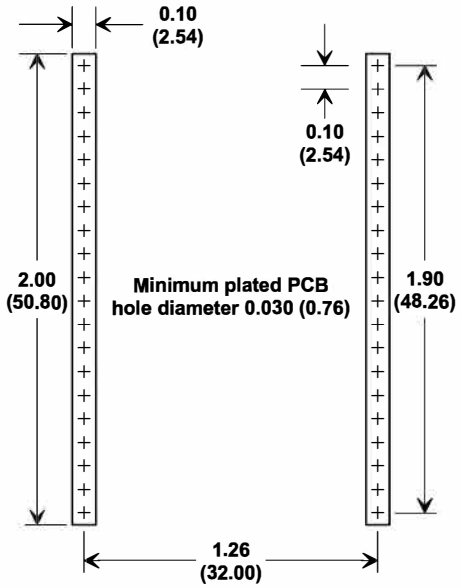
DNT2400P Outline and Mounting Dimensions



Dimensions in inches and (mm)

Figure 8.3.3

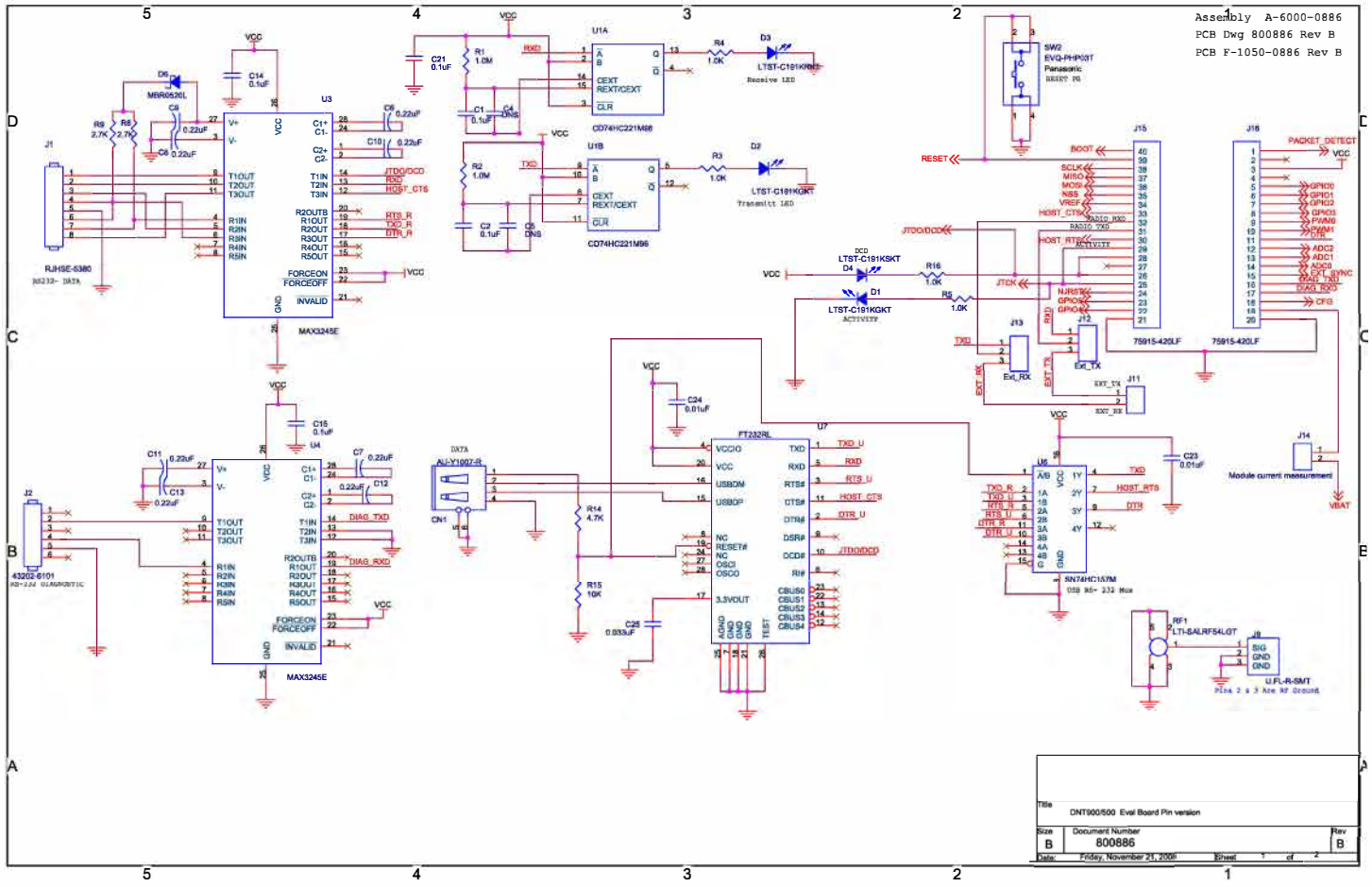
DNT2400P Interface Connector PCB Layout Detail

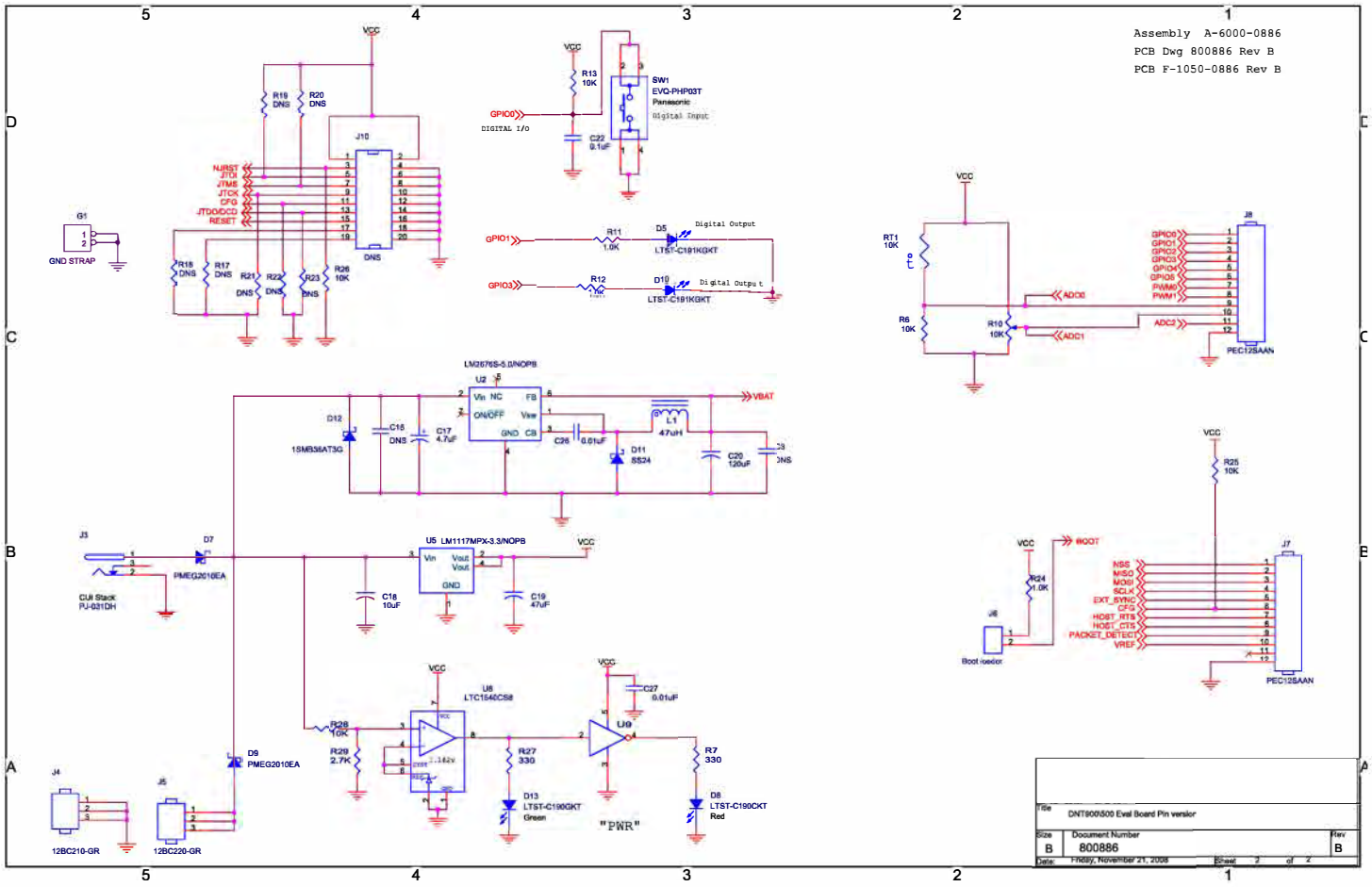


Connectors are FCI Electronics 75915-420LF or equivalent
Dimensions in inches and (mm)

Figure 8.3.4

8.4 DNT2400 Development Board Schematic





9.0 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.