2820 Wilderness Place, Unit C, Boulder, CO 80301

WIKA Module User Manual

Rev 1.1

Author: David Kramer
11/18/2022

Regular FCC/ISED Part 15C/RSS-Gen warning statement

For FCC

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.
(2) This device must accept any interference received, including interference that may cause undesired operation.

For ISED

This device contains licence-exempt transmitter(s)/receiver(s) that comply with Innovation, Science and Economic Development Canada's licence-exempt RSS(s). Operation is subject to the following two conditions:

(1) This device may not cause interference.
(2) This device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'ISDE Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes :
(1) l'appareil ne doit pas produire de brouillage, et
(2) l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

Regular FCC/ISED RF exposure considerations

For FCC

This modular complies with FCC RF radiation exposure limits set forth for an uncontrolled environment. This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
RF Exposure - This device is only authorized for use in a mobile application. At least 20 cm of separation distance between the module and the user's body must be maintained at all times.

For ISED

This device meets the IC requirements for RF exposure in public or uncontrolled environments. This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
RF Exposure - This device is only authorized for use in a mobile application. At least 20 cm of separation distance between the module and the user's body must be maintained at all times.

Instruction to module Integrator (for module integration)

Labelling
A label must be affixed to the outside of final commercial product with the following statements:

This device contains FCC ID: N4T-WMC915R1
Contains IC: 3196A-WMC915R1

Antenna

The following antennas are recommended for use with the WIKA Radio Module:

Nearson vertical whip P/N S1551AH-915S (Peak Gain = 2.0 dBi)

Pulse Larsen “mag mount” antenna P/N NMO5T900B (Peak Gain = 2.75dBi)

To comply with FCC regulations, any antenna used must either employ a “non-standard” connector, or be permanently affixed to the equipment containing the RF Module. The Nearson antenna is supplied with an “Reverse Polarity SMA connector (RP-SMA)”, which meets this requirement. The Pulse Larsen antenna must be used with a magnetic mount containing a non-standard connector such as the Laird P/N: GB195RPSMAI, which also uses an RP-SMA connector.

Alternative antennas may be used, provided peak gain is equal to or lower than antennas used for FCC compliance testing. Any alternative antenna used must employ a “non-standard” antenna or be permanently affixed to the housing containing the radio module.

Peak gain for alternative antennas are as follows:

Vertical Whip: peak gain =< 2.7 dBi

Mag Mount Antenna: peak gain <= 2.75dBi

FCC RF exposure considerations

Consistent with §2.909(a), the following text must be included within the user’s manual or operator instruction guide for the final commercial product:

This modular complies with FCC RF radiation exposure limits set forth for an uncontrolled environment. This transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.
This device is only authorized for use in a mobile application. At least 20 cm of separation distance between the module and the user's body must be maintained at all times.

Additional testing, Part 15 Subpart B disclaimer

The final host / module combination may also need to be evaluated against the FCC Part 15B criteria for unintentional radiators in order to be properly authorized for operation as a Part 15 digital device. The FCC Part 15 Statement shall be included in the user manual of final commercial product if applicable.

Caution Statement for Modifications:
CAUTION: Any changes or modifications not expressly approved could void the user’s authority to operate the equipment.

1 Overview

The WIKA Radio Module is a radio transceiver designed to provide medium range digital communication between two nodes. Typically, these nodes will be a battery powered remote sensor and a gateway, both utilizing the transceiver module. It operates in the 902 – 928 MHz frequency band which allows for unlicensed operation in North America.

The module is represented pictorially below:

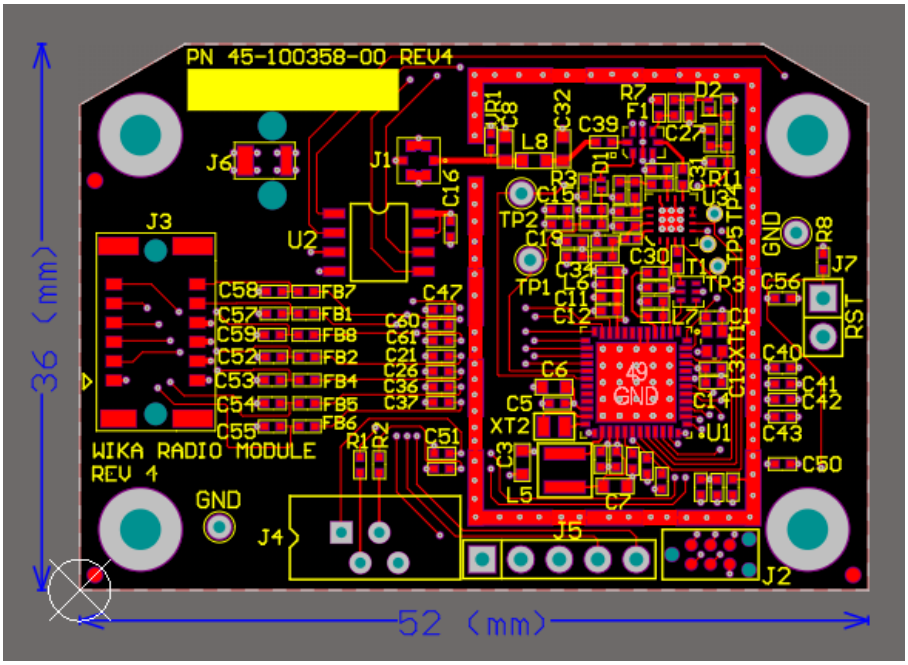


Figure 1 -RF Module Design

As is shown, the dimensions are 36mm x 52mm. There are four mounting holes for attachment into the housing. The connector at J3 mates to the digital control board, which provides digital I/O and +3.3VDC regulated power. J4 attaches to an FFC (Flexible Flat Cable) that attaches to a user accessible indicator/switch panel. J2 is the “tag connect swd” programming port.

The antenna cable attaches at J1.

2 Specifications

2.1 I/O

2.1.1 Digital I/O (J3)

The main interface and input/output connector is an ERNI#234206, which establishes the following signals:

Table 1 - J3 Pin Descriptions

Pin	Signal Name	Description	Levels
A1	VDD	3.3V Regulated DC from Digital Board	
A2	TXD	USART TXD to interface Board (OUTPUT)	0/3.3V
A3	RXD	USART RXD from interface Board (INPUT)	0/3.3V
A4	CAN_CTS	Handshake signal from interface board (INPUT)	0/3.3V
A5	RADIO_CTS	Handshake signal to interface board (OUTPUT)	0/3.3V
A6	CLI_TXD	Command line usart tx (OUTPUT)	0/3.3V
B1	VDD	3.3V Regulated DC from Digital Board	
B2	GND	System Ground	
B3	GND	System Ground	
B4	GND	System Ground	
B5	CLI_RXD	Command line usart rx (INTPUT)	0/3.3V
B6	CLI_ACTIVE	When high, activates command line interface CLI_RXD and CLI_TXD (INPUT)	0/3.3V

2.1.2 FFC Connector (LED & Switch)

Table 2- J4 Pin Descriptions

Pin	Signal Name	Description
1	VDD	3.3V Regulated DC from Digital Board
2	N_LED_2_FFC	LOW ENABLES LED2
3	N_LED_1_FFC	LOW ENABLES LED1
4	SWITCH	EXT SWITCH LINE. PUSH SHORTS PIN 1, PIN4

Note: Connector is Molex#39-53-2045

2.1.3 Antenna Connector

The antenna connector is an IPEX# 20579-001E which is compatible with IPEX MHF 4L series "locking" RF coaxial cables.

2.2 Radio Specification

2.2.1 General

OTA Data Rate:	200kbps
Modulation:	OQPSK DSSS
Spreading Factor:	8
Chipping Code Length:	32

2.2.2 Radio Transmitter

Frequency of Operation:	902 – 928 MHz
Number Frequency Channels:	19
Min RF Frequency (CH 0)	906 MHz
Max RF Frequency (CH 18)	924 MHz
Max Transmit Output Power (FEM enabled):	26 dBm nominal
DC Current Max Output Power (FEM enabled):	350 mA

Max Transmit Output Power (w/pass-through):	19 dBm nominal
DC Current Max Output Power (w/pass-through):	102 mA
Min Transmit Power (w/pass-through):	-23dbm nominal
DC Current at Min Transmit Power (w/pass-through):	9 mA

2.2.3 Radio Receiver

Receive Sensitivity:	-102dBm nom
Receiver Noise Figure:	<2dB
DC Current in Receive Mode:	19mA nom

3 Theory of Operation

Please refer to the block diagram (Figure 2- System Block Diagram) below relating to the following points.

The heart of the design is the Silicon Labs EFR32MG12P433F1024GM48-C MCU/Radio IC. This part combines a 902-928 MHz radio transceiver with ARM Cortex M4 MCU. RF output power of up to +20dBm is possible, however for this design, we will incorporate the internal step-down regulator, which will lower both DC current consumed and maximum RF power.

To increase the available output power and improve receiver sensitivity a Skyworks SKY66423-11 Front End Module (FEM) is incorporated. This IC includes a power amplifier (PA), low noise amplifier (LNA), and appropriate switching depending on radio mode (transmit or receive). It also features a “pass through” mode, which permits the module to use only the PA within the EFR32MG12 part, saving the current consumption associated with FEM, when full power is not required. Full output power is +26dBm (with FEM), and minimum power is -23dBm (without FEM).

The main interface connector (J3) contains a digital serial “command line interface” (CLI) used for diagnostics, and digital I/O lines which provide command/control communication to the customer main board. The main board provides regulated 3.3V which is distributed throughout the radio module. Additional regulation for the radio is provided within the EFR32MG transceiver IC. There is an additional I/O connector (J4) for the front panel indicators and controls. This is a four-line flat flexible connector (FFC).

The EFR32MG transceiver contains two crystal oscillators, using external crystals. One at 32.768KHz is used for real time clock tracking, and one at 38.4MHz provides the reference for the radio frequency synthesizer(s) and main MCU clock.

The EFR32MG transceiver provides separate outputs for the TX and RX signals in the (lower frequency) band used in this design. These signals are routed to the FEM input and output respectively. The FEM LNA provides about 18dB of gain, with a NF of 1.5dB typical. The PA provides a gain of about 28dB and output power of 26dBm. This FEM provides a “bypass” mode, which will lower module current consumption in close range situations, where high output power is not required.

The FEM interfaces to a directional coupler/filter combination ceramic LTCC. The directional coupler, with the addition of two detector diodes allows for testing of the antenna return loss and transmitted power. Detector voltage is measured using ADCs within the EFR32MG12. This is useful for module production testing, and field performance diagnostics. The filter provides a minimum of 27dB of loss at all relevant harmonics.

The FEM RF input/output connects to an IPEX MHF4L (J1) series connector. This connector locks the antenna cable to the connector, reducing the likelihood that the cable will pull off with vibration or shock in the field.

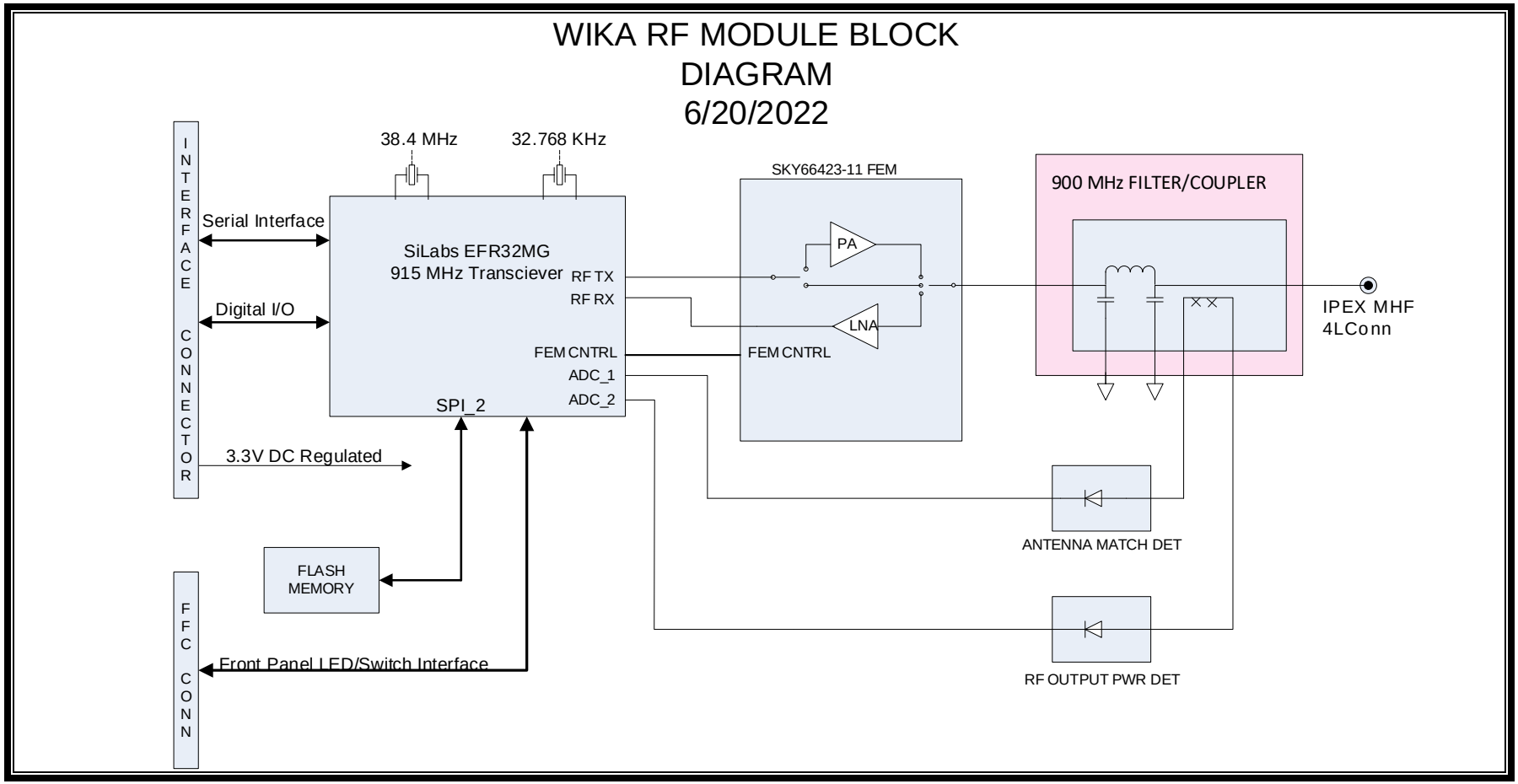


Figure 2- System Block Diagram

4 Module Mechanical

The drawing below details the PCB dimensions, and location of mounting holes.

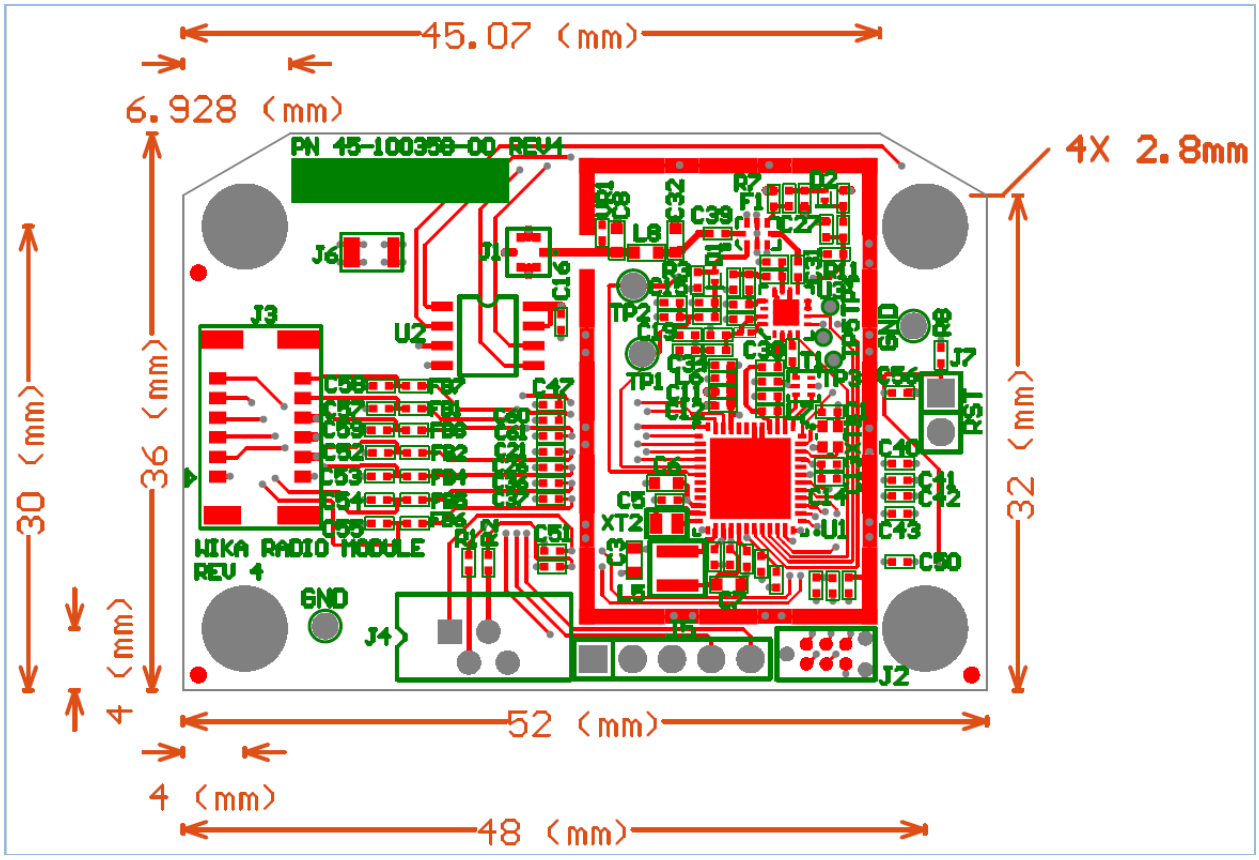


Figure 3 - PCB Dimentions and layout

If additional detail is required, a 3D step model is available from Phase IV engineering.

5 Command and Control

5.1 UART Communications

5.1.1 UART Communications

The radio modules shall service a UART communication channel with the UART-to-CAN (H2 or H3) interface board. This communications channel allows for both Command/Response from the UART-to-CAN, but also asynchronous outgoing event messages to the UART-to-CAN board.

5.1.1.1 Hardware Communication Protocol

Baud Rate: 1Mbps
Data Bits: 8
Parity Bit: No
Stop Bits: 1
No hardware or software flow control is supported; however, a 'handshake' is implemented that in some ways acts similar to RTS/CTS.

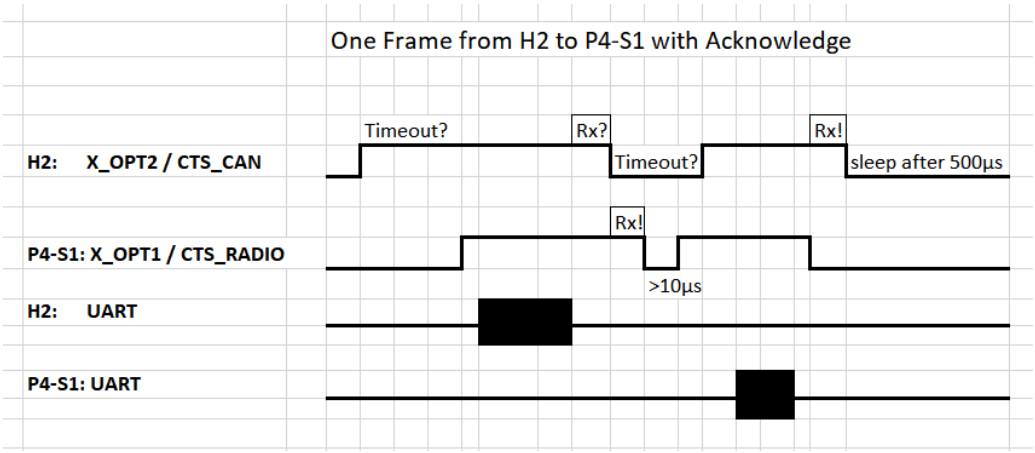
5.1.1.2 Hardware Handshaking

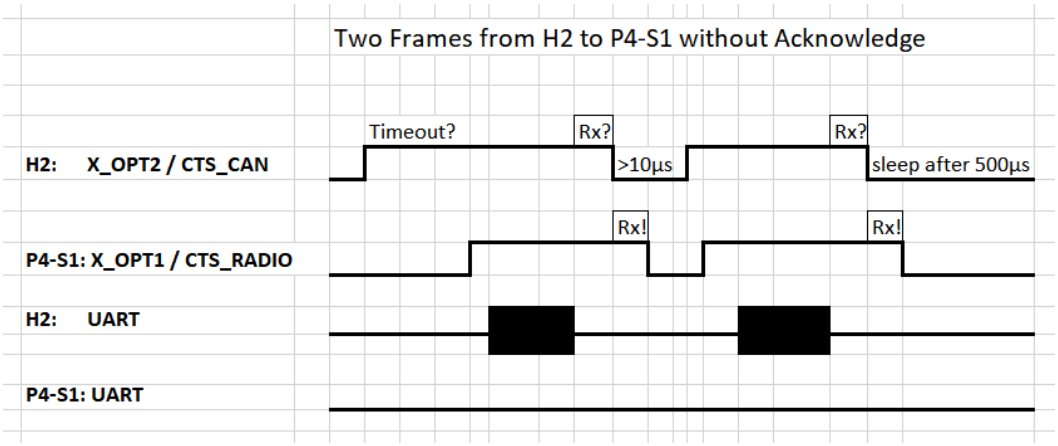
Because this is a low power system, to save power each piece of hardware in the system must go into 'sleep' mode when not actively processing data. The USART hardware within each microcontroller cannot receive data when in sleep mode (as the clock driving the peripheral is stopped). For the boards to communicate, they must use a handshaking method to notify the opposite board that is has data to send, and alternately must let the other board know it is ready to receive data.

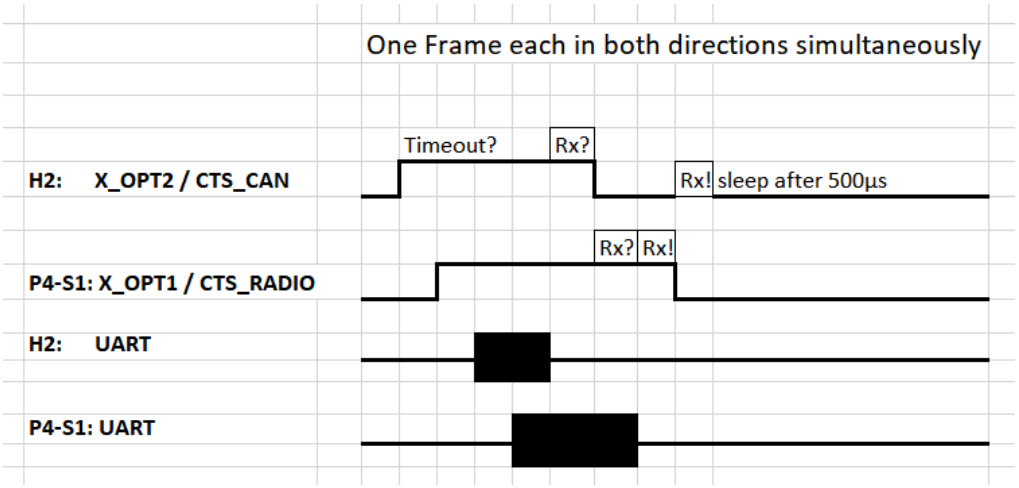
CTS_CAN and CTS_RADIO is used both to request wake-up and to signal readiness.
UART transmission is only allowed if both CTS_CAN and CTS_RADIO is high.
The transmitter must be able to receive data at the same time.
The transmitter sets its CTS low after one packet is transmitted but keeps the receiver active if it was receiving something at that point of time.
If one device was only receiving it must check CTS of the transmitter to detect the end of the frame, then it sets its own CTS low.
If one device is already receiving, and at least 2 bytes are received, it is not allowed to start simultaneous transmission, and any transmission must be delayed until the active receive packet is finished by CTS becoming low.

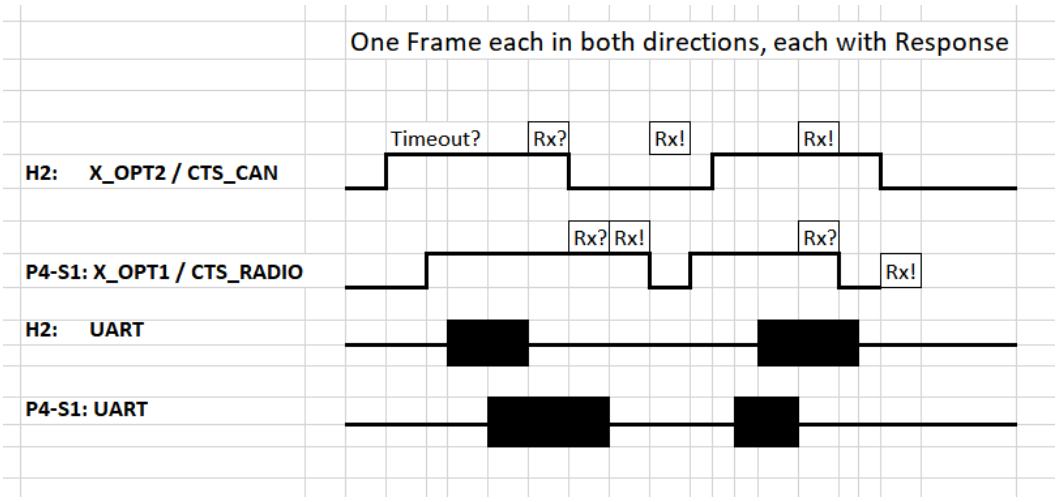
Meaning of CTS_CAN Low->High:
P4-S1 must wake up (if not already awake)
P4-S1 must enable receiving UART
P4-S1 must set CTS_RADIO high (if not already done)
H2 is awake
H2 is ready to receive
H2 might start transmitting if CTS_RADIO is high, or not transmit
Meaning of CTS_CAN High->Low:
Transmission of H2 is finished (if there was transmission)
H2 might go sleeping after 500 µs if P4-S1 has set CTS_RADIO to low
P4-S1 must check / evaluate UART Rx buffer
P4-S1 must set CTS_RADIO low for at least 10 µs to enable next UART transmission

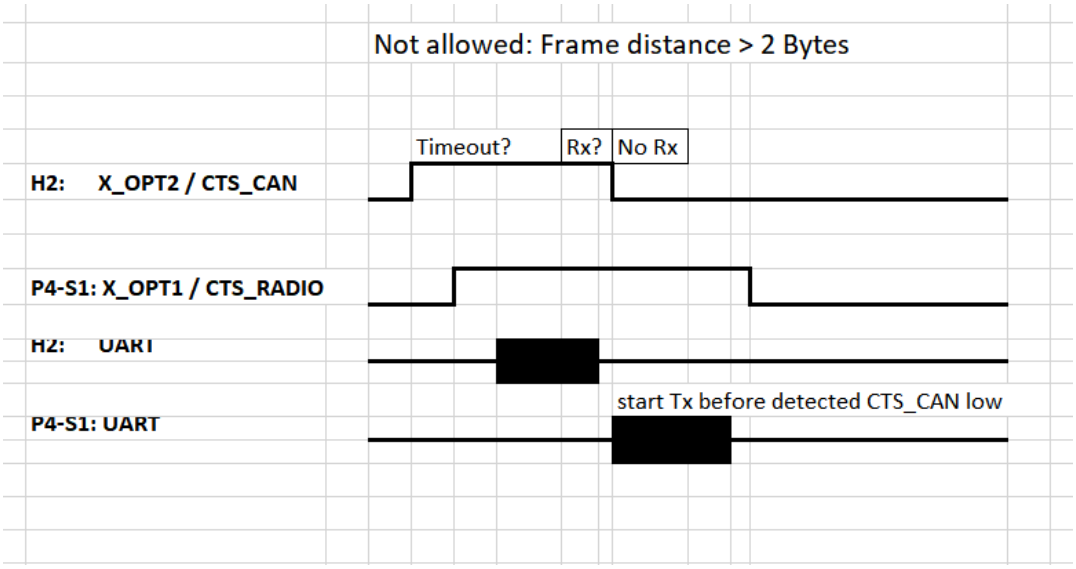
P4-S1 is allowed to sleep if P4-S1 has no own actions to do











5.1.1.3 Message Framing

Offset	Size	Name	Type	Description
0	1 Bytes	length	uint8_t	The length of the entire packet, including the 'length' field.
1	1 Byte	msgId	Commands Enum(see section 5.1.1.5)	The enumeration that defines what is found in the messageData.
2	Length – offset Bytes	messageData	uint8_t[]	Application data bytes. Length will depend on the command specified by 'msgId'.

5.1.1.4 Enumerations

The following enumerations are used in the communication protocols. It is assumed that they follow standard C enumeration rules (i.e., the first value is 0 and each value after is incremented by 1). After the first version of firmware is released, new values should always be appended to the end to maintain backwards compatibility.

5.1.1.5 Commands

Each command in this table will have a corresponding command described in the sections below. To focus on core functionality, the "Phase 1 Implementation" column denotes commands that will be implemented in the first phase of the project. The remaining commands can be implemented in further refinements and expansion of the system.

Name	Enum Value	Interface Direction
SetSystemTime	0	P4-H1 -> H2 P4-H2 <- H3
SetSystemTimeResponse	1	P4-H1 <- H2 P4-H2 -> H3
SendCANDataPacket	2	P4-H1 <- H2 P4-H2 <- H3
SendCANDataPacketResponse	3	P4-H1 -> H2 P4-H2 -> H3
CANDataPacketEvent	4	P4-H2 -> H3 P4-H1-> H2
NetworkStateChangedEvent	5	P4-H1 -> H2 P4-H2 -> H3
NetworkStateChangedEventResponse	6	P4-H1 <- H2 P4-H2 <- H3
GetSystemTime	7	P4-H1 <- H2 P4-H2 <- H3
GetSystemTimeResponse	8	P4-H1 -> H2 P4-H2 -> H3
GetNetworkInfo	9	P4-H1 <- H2 P4-H2 <- H3
GetNetworkInfoResponse	10	P4-H1 -> H2 P4-H2 -> H3
SetChannel	11	P4-H1 <- H2 P4-H2 <- H3
SetChannelResponse	12	P4-H1 -> H2 P4-H2 -> H3
EnableJoining	13	P4-H1 <- H2 P4-H2 <- H3
EnableJoiningResponse	14	P4-H1 -> H2 P4-H2 -> H3
SetTxPower	15	P4-H1 <- H2 P4-H2 <- H3

SetTxPowerResponse	16	P4-H1 -> H2 P4-H2 -> H3
RemovePairing	17	P4-H2 <- H3 P4-H1 <- H2
RemovePairingResponse	18	P4-H2 -> H3 P4-H1 -> H2
GetDevices	19	P4-H2 <- H3
GetDevicesResponse	20	P4-H2 -> H3
GetDeviceInfo	21	P4-H2 <- H3
GetDeviceInfoResponse	22	P4-H2 -> H3
StartEnergyScan	23	P4-H2 <- H3
StartEnergyScanResponse	24	P4-H2 -> H3
EnergyScanCompleteEvent	25	P4-H2 -> H3
RemoteSetNetworkNodeType	26	P4-H2 <- H3
RemoteSetNetworkNodeTypeResponse	27	P4-H2 -> H3
RemoteJoinNetwork	28	P4-H2 <- H3
RemoteJoinNetworkResponse	29	P4-H2 -> H3
GetAllDevicesInfo	30	P4-H2 <- H3
GetAllDevicesInfoResponse	31	P4-H2 -> H3

Table 3 - Commands Enumeration

5.1.1.5.1 CommandStatus

Name	**Enum Value**	Description
CommandSuccess	0	The command was successfully executed.
CommandParameterError	1	A value of a passed parameter was invalid.
CommandBufferFull	2	A radio message could not be sent because the transmit buffer is full for the destination. This can occur when trying to send a lot of data to sleepy end devices which have not 'polled' for data recently.
CommandNetworkError	3	An undefined network error occurred.
CommandCCAFail	4	The transmit attempt failed because all CCA attempts indicated that the channel was busy.
CommandUnknownDestination	5	Transmission failed: the destination node does not appear in the neighbor or child tables.
CommandFailBusy	6	The command could not be executed because a previously running process was running and cannot be interrupted.

5.1.1.5.2 NetworkState

The current state of the network from the viewpoint of the radio module.

Name	**Enum Value**	**Description**
NetworkDown	0	The node is not associated with a network in any way.
NetworkPairing	1	The node is currently attempting to join or pair to a network.
NetworkConnected	2	The connection to the network is functional.
NetworkDisconnected	3	The connection to the gateways is currently not functional, but the device is 'paired'.

NetworkUnknown	4	When device resets and the state is unknown. The only time this should ever happen is on reset.

Table 4 - NetworkState Enumeration

5.1.1.5.3 NetworkNodeType Enum

Defines the capabilities of a node on the network. Note: Not all these types are to be used for all networks.

Name	Enum Value	Description
Unknown	0	The device is not in network
StarCoordinator	1	Extended star mode device: Will relay messages and can act as a parent to range extender and end device nodes.
StarRangeExtender	2	Will relay messages and can act as a parent to end device nodes. Joins to a coordinator.
StarEndDevice	3	Communicates only with its parent and will not relay messages.
StarSleepyEndDevice	4	An end device whose radio is turned off when not communicating to save power. Must poll its parent to receive messages.
DirectDevice	5	A device able to send and receive messages from other devices in range on the same PAN ID, with no star topology restrictions. Such device does not relay messages.
MacModeDevice	6	A device able to send and receive MAC-level messages.
MacModeSleepyDevice	7	A sleepy device able to send and receive MAC-level messages. The radio on the device is turned off when not communicating.

Table 5 - NetworkNodeType Enumeration

5.1.1.5.4 Application Data Communications Protocol

The following Command/Response packets define the protocol

5.1.1.5.4.1 SetSystemTime

Sets the RTC of the system clock on the P4H2 (gateway) or P4H1 board. If commanded to the gateway, this will cause the gateway to trigger the time synchronization of all end devices.

Offset	Size	Name	Type	Description
0	8 Bytes	timeMs	uint64_t	The number of milliseconds that have expired since Jan 1, 1970.

Table 6 - SetSystemTime Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command

Table 7 - SetSystemTimeResponse

5.1.1.5.4.2 SendCANDataPacket

Sends a CAN data buffer with payload of up to 20 bytes to the specified address. Note: If the destination end device is a sleepy end device, the message will remain in the transmit buffer until the end device polls for data. If the destination is the gateway, the message will be sent immediately. When an end device (sensor side) is sending the CAN message to the gateway, the 'shortAddress' parameter should be 0x0000. A SendCANDataPacketResponse is sent

immediately when the data is validated and enqueued for transmission and not when the message is transmitted. Radio packets are automatically retried if the first transmission is not successful.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the device to send the message to. Use 0xFFFF to send to all paired devices.
2	1 Byte	length	uint8_t	The number of bytes in the following CAN data message. The maximum length is 20.
3	"length" Bytes	payload	uint8_t[length]	The binary array to send to the end device.

Table 8 - SendCANDataPacket Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The local response to the command (before radio tries to transmit)

*Table **9** – SendCANDataPacketResponse*

5.1.1.5.4.3 CANDataPacketEvent

Triggered when the radio interface receives a CAN message with up to 20 bytes from another radio interface.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short address of the device the CAN packet arrived from.
2	1 byte	Rssi	int8_t	The received signal strength this message was received at.
3	2 bytes	Tx Power	int16_t	The transmit power at which this packet was transmitted with.
5	1 Byte	length	uint8_t	The number of bytes to follow containing the CAN packet buffer. Maximum of 20.
6	'length' bytes	data	uint8_t[length]	The CAN data buffer.

Table 10 – CANDataPacketEvent

5.1.1.5.4.4 NetworkStateChangedEvent

This message will be sent when the network state changes. Changes are triggered on the Heartbeat message interval.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The local short address of the device that changed

				state.
2	1 Byte	newState	NetworkState	The new network state of the device.

Table 11 – NetworkStateChangedEvent

This message is sent by H2 in response to a NetworkStateChangedEvent (above) from P4-H1.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The local short address of the device that changed state.
2	1 Byte	newState	NetworkState	The network state of the device to acknowledge correct receipt.

Table 12 - NetworkStateChangedEventResponse

5.1.1.5.4.5 GetSystemTime

Queries the RTC of the system clock on the gateway (or end device).

Offset	Size	Name	Type	Description
N/A	N/A	N/A	N/A	N/A

Table 13 - GetSystemTime Command

Offset	Size	Name	Type	Description
0	8 Bytes	timeMs	uint64_t	The number of milliseconds that have expired since Jan 1, 1970.
0	1 Byte	status	CommandStatus	The response to the command

Table 14 - GetSystemTimeResponseGetSystemTimeResponse

5.1.1.5.4.6 GetNetworkInfo

Queries the radio module for its current network state and information.

This message has no arguments.

Offset	Size	Name	Type	Description
N/A	N/A	N/A	N/A	N/A

Table 15 - GetNetworkInfo Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The result of the command
1	8 Bytes	mcuId	uint8_t[]	The EUI64 of the radio
9	1 Byte	state	NetworkState	The current state of the network
10	2 Bytes	panId	uint16_t	The id of the network that the gateway is managing.
12	1 Byte	channel	uint8_t	The logical radio channel the network is operating on.

13	2 Bytes	txPower	int16	The transmit power of the radio in centi dBm.
15	1 Byte	allowingJoin	bool	Whether the gateway is currently accepting Join requests from end devices. (only valid on gateway, on end device will always be false)
16	1 byte	parentRssi	int8_t	
17	1 byte	rssi	int8_t	

Table 16 - GetNetworkInfoResponse

5.1.1.5.4.7 SetChannel

Sets the radio channel the network device should operate on. For a gateway, this will permanently set the new channel until another SetChannel command is received. For end devices, this will temporarily set the channel until the frequency agility algorithm causes a change in channels.

Offset	Size	Name	Type	Description
0	1 Byte	channel	uint8_t	The channel the device should change to.

Table 17 - SetChannel Command

After processing the command, the network device will send a response message.

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command. If 'channel' is outside the defined set of channels for the enabled PHY, **CommandParameterError** is returned as the CommandStatus.

Table 18 – SetChannelResponse

5.1.1.5.4.8 EnableJoining

Starts the joining mode on a gateway or range extender.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short id of the network device that should allow joining. If the receiving device is a gateway and the shortAddress ==0, the gateway will enable pairing. If shortAddress!= 0, a radio message will be sent to the child node to enable joining (only if the child node is a range extender).
2	1 byte	timeoutSec	uint8_t	The time in seconds to stay in joining mode. Joining mode is exited after the specified timeout or after the first successful join with an end device. A value of '0' will disable any current joining session. A value of 0xFF will enable joining indefinitely.

3	1 Byte	payloadLen	uint8_t	The length of the following joining data. To disable the joinPayload, set payloadLen = 0. Maximum payloadLen = 20
4	'payloadLen' bytes	joinPayload	uint8_t[payloadLen]	Causes the gateway or range extender to only accept devices attempting to join that include a matching payload in the join request. If this field is null (payloadLen=0), all join requests will be granted.

Table 19 - EnableJoining Command

After pairing mode has been started, the gateway will respond back with a response message.

Offset	Size	Name	Type	Description
0	2 Byte	shortAddress	uint16_t	The short network address of the network device that command was sent to.
2	1 Byte	status	CommandStatus	The result of the command. If a device with 'shortAddress' cannot be found, will return CommandUnknownDestination. If an end device not in range extender mode, will be `CommandParameterError`

Table 20 - EnableJoiningResponse

5.1.1.5.4.9 SetTxPower

Sets the radio transmit power that the module will use. Note that not every value is achievable. The module will use the closest available value to the requested value. For an end device, if 'automatic' power control is desired, the 'txPower' value should be set to INT16_MAX (32767). Gateways do not implement automatic power control.

Offset	Size	Name	Type	Description
0	2 Bytes	txPower	int16_t	The desired transmit power of the radio in centi dBm.

Table 21 – SetTxPower Command

In response to the command, a response is sent:

Offset	Size	Name	Type	Description
0	1 Byte	actualTxPower	int16_t	The power setting the radio was able to achieve.

Table 22 - SetTxPowerResponse

5.1.1.5.4.10 RemovePairing

Erases the pairing information for an end device (or all devices) on the gateway. Note that communication with this end device will stop functioning and the short address will be reused on another end device pairing.

If executed on an end device, shortAddress must be the end devices short address or 0xFFFF.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the end device to remove from the device list. Use 0xFFFF to remove all pairings.

Table 23 - RemovePairing Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command

Table 24 – RemovePairingResponse

5.1.1.5.4.11 GetDevices

Queries the gateway for a list of devices in the network. The list that is returned may be used to enumerate each device further using the GetDeviceInfo (see 5.1.1.5.4.12) command.
This message has no arguments.

Offset	Size	Name	Type	Description
N/A	N/A	N/A	N/A	N/A

Table 25 - GetDevices Command

Response Message

Offset	Size	Name	Type	Description
0	1 Byte	numDevices	uint8_t	The number of paired devices.
1	2*numDevices Bytes	shortAddresses	uint16_t[numDevices]	An array of the short network addresses for all paired devices.

Table 26 - GetDevicesResponse

5.1.1.5.4.12 GetDeviceInfo

Requests the gateway to send the information for a given end device. The table this information is coming from is stored in RAM of the gateway and thus values will be cleared on a power cycle (except for items that are stored in NVM as defined in **Error! Reference source not found.**).

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the end device to get info.

Table 27 - GetDeviceInfo Command

Offset	**Size**	**Name**	**Type**	**Description**
0	2 Byte	shortAddress	uint16_t	The short network address of the end device.
2	8 Bytes	longAddress	uint8_t[8]	The EUI64 of the end device.
10	1 Byte	state	NetworkState	The current communication state of the end device.
11	1 Byte	nodeType	NetworkNodeType	Defines the role the device is currently configured as.
12	1 Byte	deviceType	uint8_t	Identifies the sensor type that this radio module is connect to. These values are defined by WMC and not tracked by the radio modules.
13	4 Bytes	lastReportedMs	uint32_t	The number of elapsed milliseconds since the last data message was received from the end device.

17	1 byte	rssi	int8_t	The RSSI of the last message received from the end device.
18	1 byte	parentRssi	int8_t	
19	2 Bytes	parentShortAddress	uint16_t	The short address of the parent of this device. A gateway always has parent 0x0.
21	2 bytes	txPower	int16_t	The transmit power the devices is using in centi-dB.

Table 28 - DeviceInfo_t structure

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command, if a device cannot be found that matches the shortAddress, this will be CommandUnknownDestination and all fields below will be 0.
1	Sizeof(DeviceInfo_t)	DeviceInfo	DeviceInfo_t	Structure containing the device info. (See Table 28)

Table 29 - GetDeviceInfoResponse

5.1.1.5.4.13 StartEnergyScan

Initiates an 'energy scan' procedure on the gateway to determine relative noise floor levels for each channel. The RSSI for each channel in the current radio profile is measured 'numSamples' times. Statistics are generated for each channel and will be reported in the "EnergyScanComplete" event. When the gateway is performing an energy scan, normal radio communication is disrupted from all end devices. Communication will resume when the scan is complete automatically.

Offset	Size	Name	Type	Description
0	1 Byte	numSamples	uint8_t	The number of times the RSSI should be sampled for each channel.
1	**1 Byte**	**listSize**	**uint8_t**	**The number of channels to report back in the EnergyScanComplete message. The top 'listSize' channels will be sorted according to the best 'mean' RSSI values. Must not be greater than the maximum number of channels supported by the current PHY.**

Table 30 - StartEnergyScan Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command

Table 31 – StartEnergyScanResponse

Offse	Size	Name	Type	Description

t				
0	1 Byte	listSize	uint8_t	The number of EnergyScanResult_t objects to follow
1	sizeof(EnergyScanResult_t)*listSize	energyScanResults	EnergyScanResult+t[listSize]	The list of EnergyScanComplete objects sorted according to the best 'mean' RSSI values.

Table - EnergyScanCompleteEvent

```
typedef struct
{
    uint8_t channel;
    uint16_t frequency; //In MHz
    int8_t meanRssi;     //In dBm
    int8_t minRssi; //In dBm
    int8_t maxRssi; //In dBm
    uint16_t varianceRssi;
}EnergyScanResult_t;
```

5.1.1.5.4.14 RemoteSetNetworkNodeType

Changes the NetworkNodeType of an end device to the specified type. Note: Gateways cannot change NetworkNodeTypes, so this command will only affect end devices.

This message is intended to be sent to a gateway as a 'pass through' command, which will forward a radio message to the end device specified by 'shortAddress'. If the desired end device is already configured as 'nodeType', the command will not return an error message and will remain the desired type.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the end device to change type.
2	1 Byte	nodeType	NetworkNodeType (see 5.1.1.5.3)	The desired network operating mode to switch to.

Table 32 - SetNetworkNodeType Command

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the end device to change type.
2	1 Byte	status	CommandStatus	The response to the command

Table 33 – SetNetworkNodeTypeResponse

5.1.1.5.4.15 RemoteJoinNetwork

Commands an end device to join a new network.

Offset	Size	Name	Type	Description
0	2 Bytes	shortAddress	uint16_t	The short network address of the end device to send the message to.
2	**2 bytes**	**panId**	**uint16_t**	**The Pan ID of the new network to join.**
4	**1 byte**	**channel**	**uint8_t**	**The radio channel to look for the new network on.**

Table 34 - RemoteJoinNetwork Command

Offset	Size	Name	Type	Description
0	2 Bytes	oldShortAddress	uint16_t	The short address that the node used to be on the network and that was used as the 'shortAddress' field in the outgoing command. The joining process may assign a new address, which is captured in the 'shortAddress' field below.
2	**2 Bytes**	**newShortAddress**	**uint16_t**	**The short network address of the end device the message is from.**
4	**1 Byte**	**status**	**CommandStatus**	**The response to the command from the end device.**

Table 35 – RemoteJoinNetworkResponse

5.1.1.5.4.16 GetAllDevicesInfo

Returns the device info for all paired end devices in one message. Can only be run on gateway.

Offset	Size	Name	Type	Description
N/A	N/A	N/A	N/A	N/A

Table 36 - GetAllDevicesInfo Command

Offset	Size	Name	Type	Description
0	1 Byte	status	CommandStatus	The response to the command from the end device. If not CommandSuccess, no data to follow.
1	1 byte	Num Devices	uint8_t	The number of DeviceInfo_t structures to follow.
2	NumDevices * sizeof(DeviceInfo_t)	Devices	DeviceInfo_t[NumDevices]	Array of data structures holding device information.

Table 37 – GetAllDevicesInfoResponse