ME218C Communications Protocol

2014-2015 Communications Committee

Table of Contents

Section 0: Definitions 3
Section 1: Communications Overview 4
Section 2: Hardware Requirements 5
Section 3: Pairing 6
3.0 Pairing Overview *6*
3.1 Pairing Procedure *6*
3.2 Pairing Rejection *6*
3.3 Unpairing *6*
3.3.1 Unpair Event *7*
3.3.2 Communication Failure *7*
3.3.3 Balloon Popped Event *7*
Section 4: Paired Communication 8
4.0 Paired Communication Overview *8*
4.1 Timers *8*
Section 5: State Machine 9
5.1 Event Definitions *9*
5.2 State Diagram *10*
Section 6: Data Structure 11
6.1 XBee Protocol Definition *11*
6.2 ME218C Data Byte Definition *13*
Section 7: Validation Testing 16
7.1 Pairing *16*
7.2 Communication testing *16*
7.3 Unpairing *16*

Section 0: Definitions

This section will define terms that will be used throughout this document. They are placed here for ease of access, and further explanations, where necessary, will be provided at the relevant sections.

Term	Definition
SPECTRE	A hovercraft that is capable of communication with any MI6.
MI6	A controller that is capable of communication with any SPECTRE.
Free	Status of a SPECTRE with an inflated balloon.
Mind controlled	Status of a SPECTRE with a popped balloon.
Paired	The state where an MI6 and a SPECTRE has successfully exchanged a pairing request and an acknowledgement packet, and has switched to point-to-point communication. The paired MI6 will no longer broadcast pairing requests, and the paired SPECTRE will ignore any pairing request.
Pairing Input	Electromechanical part(s) of an MI6 that the user operates to select which SPECTRE to send a pairing request to.
Pairing Action	A sequence of actions that an MI6 user must perform in order for the MI6 to initiate a pairing request with a SPECTRE.
Unpairing Action	Action(s) that an MI6 user must perform in order for the MI6 to disconnect from the point-to-point communication it has established with a SPECTRE.
Pulse	A pair of information packets, consisting of a CTRL packet sent by the MI6 to the SPECTRE, and a STATUS packet response from the SPECTRE to the MI6.
Packet Interpreter	A software service * on the SPECTRE and the MI6, which handles reception and parsing of individual packets received by the XBee module, and posts packet specific events to the receive state machine.

* Packet interpreter specifics are left up to ME218 teams' discretion.

Section 1: Communications Overview

This document describes a class radio communication protocol used by the ME218C Spring 2015 class for a term project. The project involves a radio-controlled hovercraft (hereby referred to as SPECTRE) and the radio-controller (hereby referred to as MI6). The project culminates in a game, where players will be split into two teams, Free team and Mind Controlled team (defined in Section 0). Each player will control a SPECTRE through an MI6. The SPECTREs controlled by players from the Free team will each start with an inflated balloon on board. A Free team's SPECTRE whose balloon has been popped will join the Mind Controlled team.

Radio communication between the SPECTRE and the MI6 will use an XBee radio module (using Zigbee standards). This two-way communication will occur at a rate of 5Hz. The user selects on the MI6 a SPECTRE to control (based on a SPECTRE number displayed on the SPECTRE). Once the user selects a SPECTRE and performs the MI6-specific pairing action (defined in Section 0), the user's MI6 broadcasts a radio message (pair request packet) to all the SPECTREs. The MI6 and the SPECTRE will go through a pairing procedure as defined in Section 3.

Once pairing is established between an MI6 and a SPECTRE, communication proceeds with the periodic exchange of CTRL and STATUS packets between the MI6 and the SPECTRE. The MI6 sends out CTRL packets containing commands (these commands include thrust, orientation, balloon popping, braking and extra functionalities, as defined in Section 6). In response to the CTRL packet, the SPECTRE will send a STATUS packet to the MI6, and report on the balloon status (refer to Section 4).

Section 2: Hardware Requirements

In order to implement the radio communication described in this document, the following minimum hardware requirements are necessary:

- All teams will use XBee radio modules (Zigbee) to implement radio communication between MI6 and SPECTRE. Both the SPECTRE and MI6 will carry one of these modules on board.
- Each radio module will have an address associated with it.
- Each radio will implement communications at a rate of 9600 baud.
- The MI6 will have hardware mechanism(s) to implement unpair functionality, select a SPECTRE (pairing input), and initiate pairing (pairing action). (3.3.1).
- Each SPECTRE will have a pairing indicator of whether it is paired to the MI6 or not.
- Each MI6 will have a pairing indicator of whether it is paired to the SPECTRE or not.

Section 3: Pairing

3.0 Pairing Overview

A pairing sequence will link an MI6 to a specific SPECTRE through a two-packet exchange (see 3.1). Both devices will initialize into an Unpaired state, and will remain in this state until successful pairing. While in the Unpaired state, the SPECTRE will only respond to "Pairing" packets, and the MI6 will only transmit "Pairing" packets. After a successful pairing sequence, the two will be considered "paired," meaning that they will disregard any packet that was not issued by their partner device. The two devices will remain paired until an un-pairing condition is met (see 3.3)..

Throughout this section, comments about specific data packets will be made. Section 6 (Data Structure) will go into further details about these packets.

3.1 Pairing Procedure

The MI6 user is responsible for initiating pairing by selecting the SPECTRE number on the pairing input, and performing the MI6-specific pairing actions. The MI6 will then attempt to make contact with the SPECTRE selected, by broadcasting a packet that contains two data bytes: first the "Request Pairing" header (0x01) and then the device number of the SPECTRE that it wishes to pair with (between 0x01 and 0x0C). For the pairing to be successful, the designated SPECTRE must respond with a packet that is specifically addressed to the MI6 that initiated the pairing. This response packet contains two data bytes: first the "Pairing Acknowledged" header (0x03) and then a byte in which the SUC bit is set (0x01). A SPECTRE will always accept a pairing request from an MI6 in this fashion, unless one of the cases of Pairing Rejection (3.2) applies. After a SPECTRE accepts the pairing request in this fashion, it will be considered paired and its paired indicator will turn on until unpairing (see 3.3). If the MI6 receives the response packet, it will be considered paired, its paired indicator will turn on, and any subsequent pairing actions will be ignored until after unpairing (see 3.3).

3.2 Pairing Rejection

There are specific instances in which an MI6 pairing to a specific SPECTRE is not allowed, and the MI6's pairing request is therefore rejected by the SPECTRE. If in the MI6's pairing request, the MI6 designates a SPECTRE number that is already paired, the paired SPECTRE will ignore the pairing request. Similarly, if in the MI6's pairing request, the MI6 designates a SPECTRE number which it was paired immediately prior to that SPECTRE undergoing a Balloon Popped Event (3.3.2), and that Balloon Popped Event occurred within 10 seconds of the pairing request, the SPECTRE will ignore the pairing request.

3.3 Unpairing

An MI6 – SPECTRE pair can be unpaired by three methods: an Unpair Event (3.3.1), a Communication Failure (3.3.2), or a Balloon Popped Event (3.3.3). Upon unpairing, both the SPECTRE and the MI6 will turn off their paired indicators and return to their Unpaired states.

3.3.1 Unpair Event

Each MI6 will have an Unpair functionality which, upon activation, will set the UNPAIR bit in the ACTION byte of the CTRL packet . The transmission of a packet in which this bit is activated will unpair the MI6 and the SPECTRE with which it is paired (if the MI6 is currently paired to a SPECTRE). This input will have no effect if the MI6 is unpaired.

3.3.2 Communication Failure

If either device in an MI6-SPECTRE pair does not receive any packets for 1 second, the device will unpair (4.1).

3.3.3 Balloon Popped Event

In the event that the balloon on a paired SPECTRE is popped, that SPECTRE will immediately transmit a two-byte STATUS packet: a STATUS header (0x04) and a DATA byte with the MIND bit set (0x01). After this packet is transmitted, regardless if it is successfully received or not, the SPECTRE will unpair. When the MI6 receives the STATUS packet with the MIND bit set, it will also unpair. If the STATUS packet is not successfully received, the MI6 will still unpair because it will stop receiving STATUS packets from the SPECTRE. Balloon Popped Event is the only unpairing in which the two devices cannot again immediately pair (see 3.2).

Section 4: Paired Communication

4.0 Paired Communication Overview

Following a successful pairing sequence, the MI6 and the SPECTRE will perform point-to-point communication, exchanging CTRL bytes (sent from the MI6 and received by SPECTRE) and STATUS bytes (sent from the SPECTRE and received by MI6). The exchange occurs every 200 milliseconds (at a 5 Hz frequency) and the two devices will unpair should an exchange fail to occur for over 1 second, or if the unpair functionality (3.3.1) is activated.

4.1 Timers

The MI6 will ensure that communication occurs at a rate of 5Hz. The MI6 will internally keep a Control Timer(CONTROL_TMR), set at 200 milliseconds, and at every timeout, it will send a new CTRL packet to the paired SPECTRE and reset the Control Timer. The SPECTRE will respond to the CTRL packet by sending a STATUS packet.

In addition to the Control Timer, the MI6 will monitor the paired communication with a 1-second Communication Timer (COMM_TMR). The Communication Timer is reset every time the MI6 receives a STATUS packet from the SPECTRE with which it is paired. The SPECTRE monitors the connection with a similar 1 second Communication Timer that is reset every time the SPECTRE receives a CTRL packet from the MI6 with which it is paired. If either Communication Timer times out, the corresponding device will immediately unpair. It will thereby stop communicating with the paired device, which will cause the paired device's Communication Timer to also timeout. Such a failure will therefore cause both the MI6 and the SPECTRE to unpair (3.3).

Section 5: State Machine

5.1 Event Definitions

The following tables define the events that each SPECTRE and MI6 must be able to recognize.

SPECTRE Events:

Event	Description
EV_PAIRED	Generated by packet interpreter; pairing broadcast from MI6 received, destination address matches self-address of SPECTRE.
EV_UNPAIR	Generated by packet interpreter; UNPAIR notification received from currently paired MI6.
EV_NEWCOMMAND	Generated by packet interpreter; new point to point packet received from currently connected MI6 containing command data.
EV_BALLOONPOPPED	Notification of balloon popping received internally from balloon monitor. (Note: specifics of generating this event are left to the discretion of each team.)
ES_TIMEOUT[COMM_TMR]	COMM_TMR specifies the maximum time the SPECTRE will wait for the next packet before terminating the connection with the currently connected MI6. (COMM_TMR set to 1 sec.)
ES_TIMEOUT[RECONNECT_TMR]	When a EV_BALLOONPOPPED event is received, the SPECTRE and the MI6 that were paired at the time will disconnect and may not immediately reconnect, for a duration of time specified by RECONNECT_TMR. (RECONNECT_TMR set to 10 sec.)

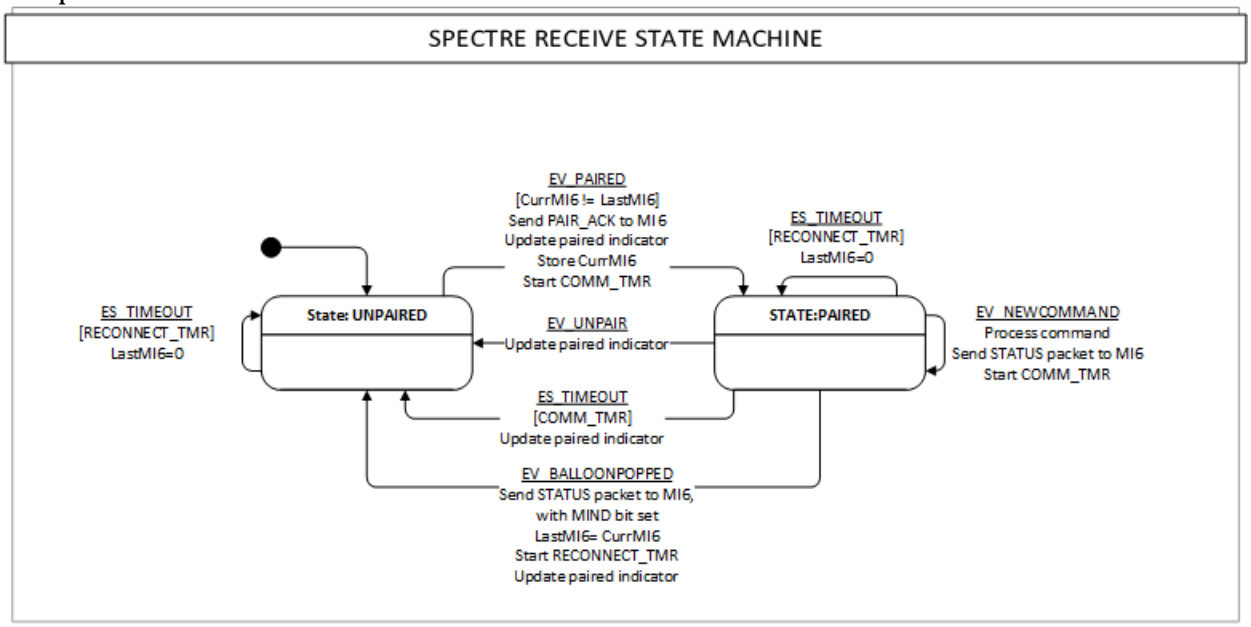
MI6 Events:

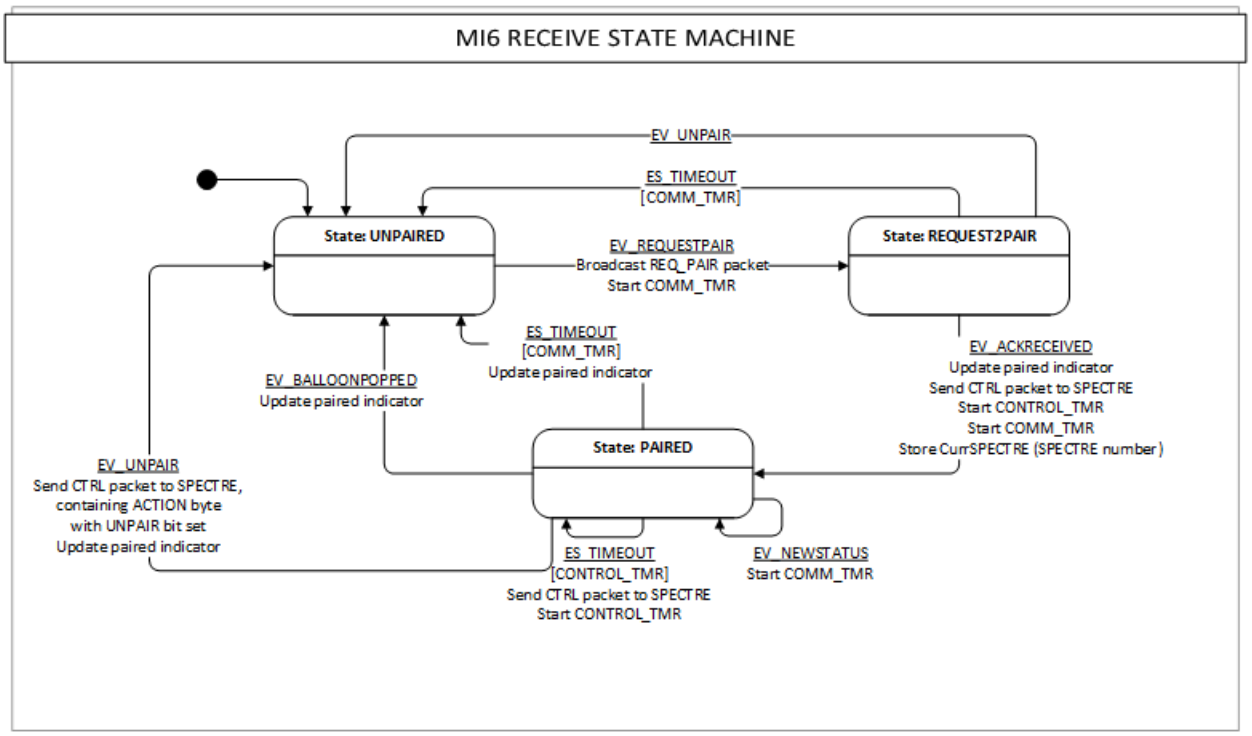
Event	Description
EV_REQUESTPAIR	Generated in response to pair action (initiated by MI6 user).
EV_ACKRECEIVED	Generated by packet interpreter; PAIR_ACK byte received from target SPECTRE.
EV_UNPAIR	Generated in response to unpair action (initiated by MI6 user).
EV_NEWSTATUS	Generated by packet interpreter; STATUS packet received from SPECTRE.
EV_BALLOONPOPPED	Generated by packet interpreter; STATUS packet received from SPECTRE with MIND bit set.
ES_TIMEOUT[COMM_TMR]	COMM_TMR specifies the maximum time the MI6 will wait for the next packet before terminating the connection with the currently connected SPECTRE. (COMM_TMR set to 1 sec.)
ES_TIMEOUT[CONTROL_TMR]	CONTROL_TMR specifies frequency at which new control packets will be sent from the MI6 to the SPECTRE. (CONTROL_TMR set to 200ms)

5.2 State Diagram

The following diagrams are the pairing state machines for the receive side of SPECTRE and the MI6 as outlined in previous sections.

*Note: Devices should ignore packets where the Checksum does not match their own computed Checksum.





Section 6: Data Structure

This section defines the data structures to be used for communications between the MI6s and SPECTREs. There are four different types of data packets: CTRL, STATUS, REQ_PAIR (request for pairing), and PAIR_ACK (pairing acknowledgement).

6.1 XBee Protocol Definition

ME218C will be using the radio devices in the non-beacon API mode of operation. The following API data frames will be used in the communication protocol specified in this document.

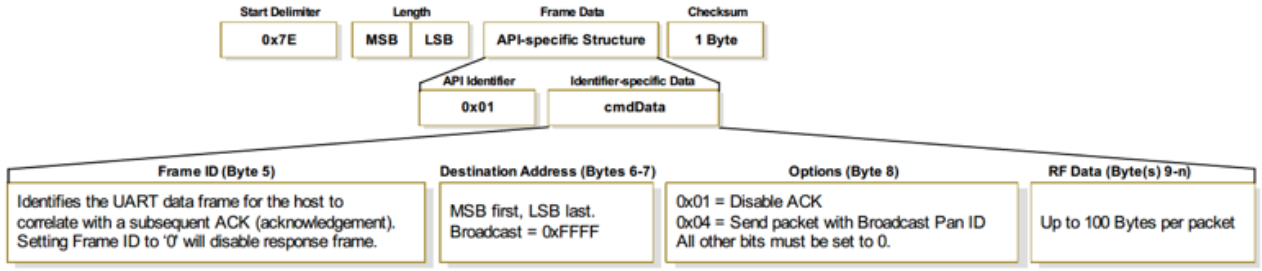
Common UART data frame structure in API mode:

The data frame for all communication between the microprocessor and the XBee has a common basic structure. The length is the length of the frame data. The checksum is defined as (0xFF - ∑ bytes in frame data). Recipient devices should ignore packets where the Checksum of the received packet does not match their own computed Checksum.

The frame data contains a Frame ID byte. This can be set independently for each frame of data. However, in the 218C remote-control application, the response for a lost CTRL or STATUS packet will be to send the next CTRL or STATUS packet, so the Frame ID is not specified by this protocol, and is available for teams to use for troubleshooting. **However, that Frame ID SHALL NOT be set to 0x00, as this disables acknowledgement packets from the XBee back to the microcontroller.**

6.1.1 TX Request Packet

This is the request-to-send frame sent from the microcontroller to the XBee radio.

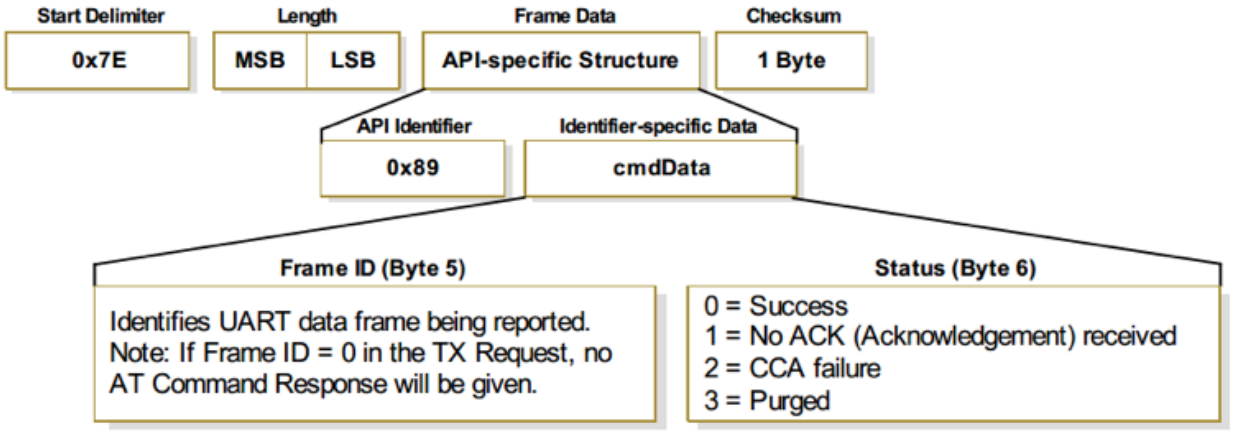


The Frame ID will not be used and must be set to anything except 0x00. The destination address is the XBee address of the desired MI6/SPECTRE to send a packet to. If it is a

broadcast packet, the destination address is 0xFFFF. The options byte should be set to 0x00 so that ACK is enabled if it is not a broadcast packet and 0x04 if it is a broadcast packet.

6.1.2 TX Status Packet

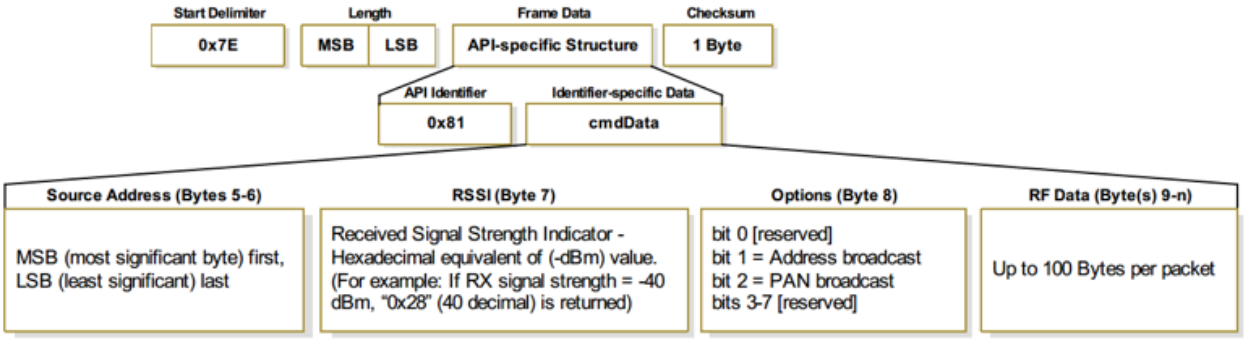
This is the frame sent from the XBee to the microcontroller on attempt to send a TX request packet (6.1.1).



When a TX request packet from the microcontroller is processed by the transmitting XBee radio, the transmitting XBee radio sends back a TX status packet to the microcontroller. The TX status packet lets the microcontroller know whether the packet was successfully delivered to the destination XBee radio. Teams should be prepared to receive TX status packets but do not need to respond to them. If a packet is not successfully sent, teams should not resend the same packet. The primary check for acknowledgement will be the CTRL and STATUS pulse packets.

6.1.3 RX Packet

This is the frame sent from the XBee to the microcontroller on receipt of a data frame from a different radio.



Bytes 5-6 contain the source address. An MI6/SPECTRE should ignore the packet if the source address does not match the address they stored as being the paired device.

6.2 ME218C Data Byte Definition

The ME218C data is contained in the RF data section of each of the above Zigbee API data frames. The RF data will consist of an initial packet header byte specifying the type of the packet, followed by anywhere from 1 to 6 additional bytes of data.

The ME218C communication protocol defines the following types of packets to be placed in RF data:

Name	DIR	HDR	Data [MSB to LSB]
REQ_PAIR	M → S	0x01	PAIRDATA:[---- ---- ---- ---- ADR3 ADR2 ADR1 ADR0]
PAIR_ACK	S → M	0x02	ACKDATA:[---- ---- ---- ---- ---- ---- ---- SUC]
CTRL	M → S	0x03	THRUST: [THR7 THR6 THR5 THR4 THR3 THR2 THR1 THR0] ORIENT: [ORI7 ORI6 ORI5 ORI4 ORI3 ORI2 ORI1 ORI0] ACTION: [UNPAIR ---- ---- ---- ---- ---- BRK POP] EXA: [EXA7 EXA6 EXA5 EXA4 EXA3 EXA2 EXA1 EXA0] EXB: [EXB7 EXB6 EXB5 EXB4 EXB3 EXB2 EXB1 EXB0] EXC: [EXC7 EXC6 EXC5 EXC4 EXC3 EXC2 EXC1 EXC0]
STATUS	S → M	0x04	SDATA: [---- ---- ---- ---- ---- ---- ---- MIND]

Notes: DIR is the direction of communication (MI6 (M) to SPECTRE (S), or vice versa). HDR is the first byte of the RF data packet, indicating the type of packet being sent.

REQ_PAIR

Direction: MI6 to SPECTRE
Length: 2 bytes

The REQ_PAIR packet is sent from an MI6 looking to pair to an unpaired SPECTRE

Byte 1: HEADER
HDR = 0x01

Byte 2: PAIRDATA
Bits <7:4> are not used and should be set to 0.
Bits <3:0>: ADR3-ADR0 is the SPECTRE number.

PAIR_ACK

Direction: SPECTRE to MI6
Length: 2 bytes

The PAIR_ACK packet is sent by the SPECTRE in response to the PAIR_REQ packet from an MI6.

Byte 1: HEADER
HDR = 0x02

Byte 2: ACKDATA
Bits <7:1> are not used and should be set to 0.
Bit <0>: SUC indicates if the pairing is successful or not.
SUC = 1: Pairing successful
SUC = 0: Pairing unsuccessful

CTRL

Direction: MI6 to SPECTRE
Length: 7 bytes

The CTRL packet is sent by the MI6 to its paired SPECTRE. It contains data about the navigation, actions to unpair, brake, and pop the balloon, and (2) analog and (1) digital bytes for extra functionalities that are optional for teams to use.

Byte 1: HEADER
HDR = 0x03

Byte 2: THRUST
Bits <7:0>: THR indicates the thrust of the SPECTRE. It is a value ranging from -128 to 127.
THR = 127: Full speed forward
THR = 0: Coast (No thrust)
THR = -128: Full speed reverse

Byte 3: ORIENT
Bits <7:0>: ORI indicates the orientation of the SPECTRE. It is a value ranging from -128 to 127.
ORI = 127: Maximum right turn
ORI = 0: Straight
ORI = -128: Maximum left turn

Byte 4: ACTION
Bit <7>: UNPAIR is the MI6 telling the SPECTRE to end communications.
UNPAIR = 0: No action
UNPAIR = 1: Sever communication, revert to UNPAIRED state
Bits <6:2> are not used and should be set to 0.
Bit <1>: BRK is the digital command for braking.
BRK = 0: No brake
BRK = 1: Brake
Bit <0>: POP is the command for popping the balloon. NOTE: SPECTREs need to have a 2 second timer when a balloon pop is executed. It may only try to pop once every 2 seconds.

It is the SPECTRE's responsibility to ignore a POP command if it is transmitted by the MI6 too soon.
POP = 0: No action
POP = 1: Deploy the popping mechanism

Byte 5: EXA
Bits<7:0>: EXA is an analog input for extra functionalities (0-255). TBD by teams.

Byte 6: EXB
Bits <7:0>: EXB is an analog input for extra functionalities (0-255). TBD by teams.

Byte 7: EXC
Bits <7:0>: EXC is a digital input for extra functionalities. Each bit serves as a digital input. TBD by teams.

STATUS

Direction: SPECTRE to MI6
Length: 2 bytes

The STATUS packet will be sent from a SPECTRE to a paired MI6 in response to a CTRL packet. This packet will be used to send status information to the MI6, and will also function as half of the pulse to monitor the communications link from SPECTRE to MI6. The status packet will notify the MI6 whether the SPECTRE is Free or Mind Controlled.

Byte 1: HEADER
HDR = 0x04

Byte 2: SDATA
Bits <7:1> are not used and should be set to 0.
Bit <0>: MIND is the status of the SPECTRE.
MIND = 0: Free
MIND = 1: Mind Controlled

Section 7: Validation Testing

This section will describe the testing procedure to ensure that all MI6s are capable of communicating with any of the SPECTREs.

7.1 Pairing

- Power up the SPECTRE, the paired indicator should signal Unpaired state
- Power up the MI6, the paired indicator should signal Unpaired state
- Select the SPECTRE number on the MI6
- Connect to the SPECTRE by activating the pairing action on the MI6
- The SPECTRE and the MI6 should activate their paired indicators.

7.2 Communication testing

- Pair the MI6 with the SPECTRE using the pairing procedure (7.1)

- Direction control

    - Adjust input on MI6 for thrust and orientation and the SPECTRE should move as expected.

- Braking
    - Activate brake on MI6 and the SPECTRE should brake.

- Balloon Popping

    - Perform the action for popping the balloon on the MI6 and the balloon popping mechanism on the SPECTRE should be activated.

7.3 Unpairing

- Pair the MI6 with a SPECTRE following the pairing procedure (7.1).
- Test each of the following independently and the MI6 and the SPECTRE should unpair:
    - Simulate a popped balloon
    - Turn off the MI6
    - Turn off the SPECTRE
    - Walk out of range
    - Activate the Unpair functionality on MI6