Bluetooth Communications Protocol – A15 (Protocol Rev 1)

A communications library will be provided that enables Bluetooth communication between the reactor control system (i.e., the playing field) and the robot on the field. The library will provide standardized interfaces for sending and receiving the various messages (documented below).

Use of the Bluetooth communications library is optional. The project can be completed without using Bluetooth functionality (although this may entail a loss of overall robot performance and fewer maximum points scored).

The library operates by implementing a simple packet-based communications protocol over Bluetooth. There are seven standard messages:

- Storage tube availability
- Supply tube availability
- Stop movement
- Resume movement
- Radiation alert
- Robot status
- Heartbeat

Of these messages, the reactor control system generates the storage and supply tube availability messages as well as the stop and resume movement messages. The software you write to control the robot should be designed to receive these messages and act upon them as appropriate.

The only message the robot must generate is the heartbeat message.[1] The robot may optionally generate radiation alert messages (see the game rules). Robot status messages may also optionally be generated by the robot. The status messages may be useful in debugging the robot control code but generally will serve no purpose as far as actual game play is concerned.

Bluetooth Client Library Function Calls

The function calls for using the Bluetooth library are detailed below.

Receive Data

bool receive(byte *data);

This function receives a single byte of data at a time over the Bluetooth communications channel. The argument is a pointer to the byte which is to receive the data. The function returns True or False depending on whether a data byte was received.

Send Data

void send(byte data);

The argument is a data byte to be sent over the Bluetooth communications channel.

[1] This is assuming the team chooses to use the Bluetooth communications protocol.

The receive and send function calls depend on a buffer area having been set aside where the data packet is either constructed (prior to being sent) or received. Care should be taken to ensure that the buffer areas are large enough to accommodate the largest packet that can be sent or received.

Bluetooth Library Message Packet Format

Each message packet has a standardized format as follows:

Table 1: General Message Structure

Field	***Offset***	***Length***	***Description***
Start delimiter	0	1	Always 0x5F
Length (in bytes)	1	1	Number of bytes from the length through the checksum inclusive
Message type	2	1	See Table 2 for details
Source address	3	1	0x00 = reactor control system (i.e., the playing field); > 0x00 = unique robot address (team number)
Destination address	4	1	0x00 = broadcast (all robots); > 0x00 = unique robot address (team number)
Data	5	Varies	Depends on message type (see below)
Checksum	Varies	1	0xFF minus the 8-bit sum of bytes from offset 1 up to but not including this byte (i.e., data bytes are included if they are present)

All messages have the structure shown in Table 1. All received messages should be checked to ensure they are correct (begin with 0x5F, have the appropriate length given the message type, and have the correct checksum). Transmitted messages should also be properly constructed (begin with 0x5F, have the appropriate length given the message type, and have the correct checksum).

Robots should check the destination address field of any received message to ensure that it applies to them. Broadcast messages apply to all robots; messages may also be directed to individual robots. If a message is received that does not apply because it is directed to a different robot, then the message may simply be discarded. Note that the reactor control system (i.e., the field control computer) has address 0x00 – which means that it is always included in any broadcast messages.

Source addresses are FYI for receiving robots; the source address may be used if needed. A correct source address (your team number) is required when a robot transmits a message.

Different messages are distinguished by their Type ID. See Table 2 for a list of all defined message types.

Table 2: Message Types

Type ID	***Description***	***Notes***
0x00, 0x08 – 0x0F	Reserved	
0x01	Storage tube availability	See data structure in Table 3
0x02	Supply tube availability	See data structure in Table 3
0x03	Radiation alert	See data structure in Table 4
0x04	Stop movement	No data associated with this message
0x05	Resume movement	No data associated with this message
0x06	Robot status	See data structure in Table 5

Type ID	*Description*	*Notes*
0x07	Heartbeat	No data associated with this message
0x10 and higher	User defined	See below

Storage & Supply Tube Availability Messages

The storage and supply tube availability messages have exactly the same structure (see Table 3) but have different message types to distinguish them.

Table 3: Storage & Supply Tube Availability Messages

Field	*Length*	*Description*
Availability mask	1	Bitmask of available tubes. Bit 0 = tube 1; Bit 1 = tube 2, … If tube bit = 0, tube is empty; if tube bit = 1, tube is occupied

The reactor control system will broadcast storage and supply tube availability messages periodically (at least once every 5 seconds). These messages are guaranteed to be generated as appropriate within 500 milliseconds of a change in the status of any tube.

Radiation Alert Message

Radiation alert messages must be generated by the robot while carrying a rod. Messages should begin being broadcast when the robot begins to extract a fuel rod from a tube and should continue until the rod has been inserted into another tube. The message should be broadcast no more often than once per second (i.e., no 'spamming' of the reactor control system is allowed), and no less often than once every 5 seconds.

Table 4: Radiation Alert Message

Field	*Length*	*Description*
Radiation level	1	0x2C = Carrying spent fuel rod 0xFF = Carrying new fuel rod

Robot Movement Messages

Robot movement messages have no data associated with them. The Type ID of the message carries all of the necessary command information.

When a robot receives a Stop Movement message directed to it, it should cease movement on the field until such time as it receives a Resume Movement message. Movement in this context is defined to be driving or related motion. For example, the robot may continue other motions (such as moving an arm mechanism) after receiving a Stop Movement message as long as there is no change in position of the robot on the field. A 'change in position' includes turning the robot or having it rotate about its own center, as well as moving horizontally.

Robot Status Message

Robot status messages are useful for informing the reactor control system and/or other robots as to what is happening. A robot status message has three data bytes, each corresponding to a different message field as described in Table 5. Robot status messages might also be used for debugging purposes. The three status fields can be updated as appropriate during the operation of the robot and then the message may be transmitted. The data bytes of a robot status message are

displayed as two hexadecimal digits each on the lower line of the field control computer LCD display from left to right in the order: Movement, Gripper, and Operation status.

Table 5: Robot Status Message

Field	***Length***	***Description***
Movement status (data byte offset 0)	1	0x00 = Reserved 0x01 = Stopped 0x02 = Moving (teleoperation) 0x03 = Moving (autonomous)
Gripper status (data byte offset 1)	1	0x00 = Reserved 0x01 = No rod in gripper 0x02 = Rod in gripper
Operation status (data byte offset 2)	1	0x00 = Reserved 0x01 = Grip attempt in progress 0x02 = Grip release in progress 0x03 = Driving to reactor rod 0x04 = Driving to storage area 0x05 = Driving to supply area 0x06 = Idle (no operation in progress)

Robot status messages are optional. If robot status messages are being used, a message could be generated when there is a change in any of the Movement, Gripper, or Operation status fields. Status messages may be repeated periodically if there is no change in status, but repeated messages should be generated no more often than once every five seconds.

Heartbeat Messages

Heartbeat messages have no data associated with them. They are transmitted by the robot so the reactor control system knows that the robot is 'alive' and well.

Heartbeat messages should be transmitted by the robot no more than once every second and at least once every two seconds. If more than three seconds passes without the reactor control system receiving a heartbeat message from the robot, it will assume the Bluetooth connection to the robot has been lost. It will then automatically attempt to re-establish a Bluetooth connection to the robot. If the robot is in fact well but simply not sending heartbeat messages (at all or at least on time) then the reactor control system will end up dropping its end of the Bluetooth connection. The robot won't know this and it will likely lead to a failure to reconnect properly. It may become necessary in this case to manually power cycle the robot's Bluetooth module in order to then attempt to re-establish the connection.

User Defined Messages:

Users are free to implement their own messages. All robot control software should be designed to fully implement (i.e., correctly process) the standard messages (message Type ID values of 0x01 – 0x07). Messages with Type IDs of value 0x08 and higher should be ignored unless the particular software control system is prepared to deal with them.

Note that the 'createPkt()' method in ReactorProtocol.cpp would need to be updated if you wish to use that to create custom messages. Similarly, the 'getData()' method would need to be updated if you wished to use that to read custom messages.

Examples:

What follows are some examples of message packets. The first (Table 6) is a heartbeat message as generated by the Team 10 robot and sent to the reactor control system.

The second example (Table 7) is that of a storage tube availability message. This message is being broadcast by the reactor control system to all robots.

The final example (Table 8) is of a Stop Movement message. It too is generated by the reactor control system but in this case is being sent to Team 5.

Table 6: Heartbeat Message from Team 10

Field	***Offset***	***Content***	***Description***
Start delimiter	0	0x5F	Indicates the start of a new message
Length (in bytes)	1	0x05	Number of bytes from the length through the checksum
Message type	2	0x07	Identifies this as a heartbeat message
Source address	3	0x0A	The message is from team (robot) 10
Destination address	4	0x00	Send to all robots (including the field control system)
Checksum	5	0xE9	0xFF minus the 8-bit sum of bytes from offset 1 up to but not including this byte

Table 7: Storage Tube Availability Message (broadcast to all robots)

Field	***Offset***	***Content***	***Description***
Start delimiter	0	0x5F	Indicates the start of a new message
Length (in bytes)	1	0x06	Number of bytes from the length through the checksum
Message type	2	0x01	Identifies this as a storage tube availability message
Source address	3	0x00	The message is from the reactor control system
Destination address	4	0x00	Send to all robots
Data	5	0x05	Bitmask of occupied and available tubes (tubes 1 and 3 are occupied; tubes 2 and 4 are available)
Checksum	6	0xF3	0xFF minus the 8-bit sum of bytes from offset 1 up to but not including this byte

Table 8: Stop Movement Message to Team 5

Field	***Offset***	***Content***	***Description***
Start delimiter	0	0x5F	Indicates the start of a new message
Length (in bytes)	1	0x05	Number of bytes from the length through the checksum
Message type	2	0x04	Identifies this as a stop movement message
Source address	3	0x00	The message is from the reactor control system
Destination address	4	0x05	The message is addressed to team (robot) 5
Checksum	5	0xF1	0xFF minus the 8-bit sum of bytes from offset 1 up to but not including this byte