Rocketry@VT

PROPULSION SUBTEAM

HYBRID MOTOR CONTROLLER INTERFACE CONTROL DOCUMENT

Version 1.0.0

**0. Update History**

Updates to the ICD are recorded here. These include the addition of new commands and data output types.

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author | Description |
| 1.0.0 | ND | MM | Release Version |

**1.1 Introduction**

The Arduino utilizes a basic user-interface command structure that requires communication via serial ports. This manual describes the Motor Controller commands and logs that the controller is capable of generating. Detail is provided to allow users to understand the purpose, syntax, and structure of the available commands and logs.

**1.2 Code Authors**

Matt Marti, [mmatth3@vt.edu](mailto:mmatth3@vt.edu)

- MATLAB Graphical Interface

- MATLAB serial communication

- Arduino serial communication interface

- XBee serial configuration

- Arduino to XBee wiring

- Sensors and electronics configuration

- Host computer Unit Tests

**2.1 Binary Data Packet Structure**

The majority of commands and logs are transmitted in a strictly machine readable format. This is done because the rate at which data is transmitted is relatively high compared to the amount of data transmitted per packet. The basic format of the binary packets are as follows:

Header 2 Synchronization bytes, followed by 1 byte representing the

number of bytes in Data length.

Data Variable number of bytes, see descriptions in following section.

Checksum 2 Bytes for error checking

Every binary data packet, either received by or transmitted by the Arduino, is preceded by a header except for one commnad. The header is three bytes long and begins with two header bytes, followed by one byte that represents the number of bytes contained in the packet.

The Data portion of the packet is a variable length array of bytes that represent the data to obtained from the Motor Controller, or data transmitted to the Motor Controller when sending commands. All data begins with one byte that represents the type of data transmitted in the packet.

The Checksum consists of two bytes transmitted at the end of the data packet. These are the result of an error checking algorithm that generates a unique number based off the data previously transmitted. This number varies greatly with differences in single bits. When computed on the receiving device, it can be compared to the received checksum. Differences between the two indicate errors in the transmitted data.

The only packet that will not have a header or checksum is the RESET command. This command is described in detail in *Section 2.3.1 RESET*.

**2.1.1 Header Structure**

The 2 Synchronization Bytes will always be the same. They will be followed by the number of bytes transmitted in the Data section of the transmitted packet. These are shown in the following table:

|  |  |  |
| --- | --- | --- |
| Byte | Hex | Decimal |
| First | 0xAA | 170 |
| Second | 0x14 | 20 |
| Third | Varies | Varies |

**2.2 Motor Controller Transmitted Packets**

The following data packets are transmitted directly from the Motor Controller during operation. Each subsection contains a brief description of the contents of the packet, and a table used for decoding the bits obtained.

To obtain the length of the transmitted packet, which is contained in Byte 3 of the Header, add the Bytes and Byte offset values in the last field of the packet description, and subtract header length H.

Output floating point values are IEEE 754 standard 32 bit floating point numbers.

**2.2.1 PACKET 0x00**

**Initialization Message**

On startup, or after the Reset command is received by the Arduino, the Arduino is configured to output to the serial port the current firmware version in ASCII. This is a string that indicates what firmware version is currently loaded on the Arduino. The firmware string is terminated by a newline character.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 1 | Firmware Version | ASCII | C | 0 |
| 2 | Newline character (10) | hex | 1 | C+1 |

**2.2.2 PACKET 0x10**

**Mode Type**

This packet is transmitted on the reception of all Mode change commands sent to the Arduino. The purpose of the command is to confirm that the Arduino has been set to the desired runmode

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 1 | Packet Header (L = 0x01) | --- | 3 | 0 |
| 2 | Packet Type, Value: 0x10 | uint8 | 1 | 3 |
| 3 | Mode Value (See below table) | uint8 | 1 | 4 |

The available modes that can the Motor Controller can be set to are described in the following table.

|  |  |
| --- | --- |
| Mode | Description |
| 0 | Default |
| 1 | Arm |
| 2 | Firing |
| 3 | Post Firing |
| 4 | N/A |
| 5 | Simulation Mode |
| 6 | Post Simulation |

**2.2.3 PACKET 0x20**

**Unit Test Response**

This data output transmits a string from the device an ASCII string, which is the messages from the Unit Tests. This is the response to the command 0x05 (*Section 2.3.6*). Message lines are separated by newline characters ‘\n’.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 4 | Unit Test Messages | ASCII | C | H+2 |

**2.2.4 PACKET 0x51**

**Data Packet #2**

This packet was developed to output data from the Arduino during testing. Outputs sensor measurements as floating point numbers.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 1 | Packet Header (L = 31) | --- | H | 0 |
| 2 | Message Type: 0x51 | uchar | 1 | H |
| 3 | Controller Time | uint32 | 4 | H+1 |
| 4 | Operating Mode | uint8 | 1 | H+5 |
| 5 | Error Status | uint16 | 2 | H+6 |
| 6 | PRESSURE\_OXIDIZER | float | 4 | H+8 |
| 7 | PRESSURE\_COMBUSTION | float | 4 | H+12 |
| 8 | TEMPERATURE\_PRECOMB | float | 4 | H+16 |
| 9 | TEMPERATURE\_COMBUSTION | float | 4 | H+20 |
| 10 | THRUST | float | 4 | H+24 |
| 11 | New Data flag | uint8 | 1 | H+28 |
| 12 | 16 bit checksum | hex | 2 | H+29 |

**2.2.5 PACKET 0x52**

**Data Packet #3**

This packet was developed to output data from the Arduino during testing. This packet features a third temperature output

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 1 | Packet Header (L = 35) | --- | H | 0 |
| 2 | Message Type: 0x52 | uchar | 1 | H |
| 3 | Controller Time | uint32 | 4 | H+1 |
| 4 | Operating Mode | uint8 | 1 | H+5 |
| 5 | Error Status | uint16 | 2 | H+6 |
| 6 | PRESSURE\_OXIDIZER | float | 4 | H+8 |
| 7 | PRESSURE\_COMBUSTION | float | 4 | H+12 |
| 8 | TEMPERATURE\_PRECOMB | float | 4 | H+16 |
| 9 | TEMPERATURE\_COMBUSTION | float | 4 | H+20 |
| 10 | TEMPERATURE\_POSTCOMB | float | 4 | H+24 |
| 11 | THRUST | float | 4 | H+28 |
| 12 | New Data flag | uint8 | 1 | H+32 |
| 13 | 16 bit checksum | hex | 2 | H+33 |

**2.3 Motor Controller Received Packets**

The following data packets are received by the Motor Controller during operation. Each subsection contains a brief description of the contents of the packet or command, and a table used for decoding the bits obtained. These are numerous, and may contain unsupported operations. Unsupported operations are noted at the bottom of the page in which they are documented, and will be updated with each release of the Motor Controller firmware. However, the developers are only human. If an unsupported operation is encountered, contact the appropriate developer.

Received floating point values are IEEE 754 standard 32 bit floating point numbers.

**2.3.1 RESET**

**Software Defined Reset of the Motor Controller**

This command is used to reset the operation of the Motor Controller. This command is unique because it is not preceded by a header and is not proceeded by a checksum.

When the Motor Controller is reset, the mode of the controller is set to 0. In future releases of the firmware, the various devices connected to the Motor Controller will be reset to bring the motor into a state which is safe to approach.

Additionally, this command causes the Motor Controller to print the version of the software to the XBee. The version is printed as “Motor Control X.Y.Z”, where X represents the version corresponding to major changes in the hardware, Y corresponds to major changes in the interface, and Z corresponds to changes in function implementations and bug fixes.

Echos with just strings

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | 0xFF | Reset Message Byte | Byte | 1 | 0 |
| 2 | 0xFF | Reset Message Byte | Byte | 1 | 1 |
| 3 | 0xFF | Reset Message Byte | Byte | 1 | 2 |
| 4 | 0xFF | Reset Message Byte | Byte | 1 | 3 |

**2.3.2 COMMAND 0x00**

**Echo Firmware Version**

Command to tell the Arduino to output the current firmware version. The Arduino will echo this command with the PACKET 0x00 (Section 2.2.1).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x00 | Echo Firmware Version | uint8 | 1 | 3 |

**2.3.3 COMMAND 0x01**

**Turns an LED on on Pin 7**

This packet was developed to test the operation of the Motor Controller serial interface with the XBee. This command turns on an LED on Pin 7 of the Arduino.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x01 | Command: Turn on LED | uint8 | 1 | 3 |

**2.3.4 COMMAND 0x02**

**Turns an LED off on Pin 7**

This packet was developed to test the operation of the Motor Controller serial interface with the XBee. This command turns off an LED on Pin 7 of the Arduino.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x02 | Command: Turn off LED | uint8 | 1 | 3 |

**2.3.5 COMMAND 0x03**

**Clear Input Buffer**

This command clears the input buffer on the Arduino. This command is used to debug the serial I/O of the Arduino. The input buffer is a byte array that stores incoming bytes in the working memory of the Arduino.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x03 | Clear Input Buffer | uint8 | 1 | 3 |

**2.3.6 COMMAND 0x04**

**Set Mode**

This command sets the mode of the Motor Controller and is used to arm, fire, stop, and simulate the controller. Essentially, this command controls operations during tests.

The Arduino echos this command with the PACKET 0x10 (Section 2.2.2), which is the packet that describes the current mode of the device.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x04 | Command Set Mode | uint8 | 1 | 3 |
| 3 | Varies | Mode Type (see below table) | uint8 | 1 | 4 |

The available modes that can the Motor Controller can be set to are described in the following table.

|  |  |
| --- | --- |
| Mode | Description |
| 0 | Default |
| 1 | Arm |
| 2 | Firing |
| 3 | Post Firing |
| 4 | N/A |
| 5 | Simulation Mode |
| 6 | Post Simulation |

**2.3.7 PACKET 0x05**

**Run Unit Tests**

Command to make the device begin the suite of Unit tests and to output the results to the serial port. Will cause the device to respond with the data packet 0x20 (*Section 2.2.2*).

Echos with just the strings

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Description | Format | Bytes | Byte offset |
| 1 | Packet Header (L = 35) | --- | H | 0 |
| 2 | Message Type: 0x20 | uchar | 1 | H |
| 3 | Number of characters transmitted | uint16 | 2 | H+1 |
| 4 | Unit Test Messages | ASCII | C | H+2 |
| 13 | 16 bit checksum | hex | 2 | H+C+2 |

**2.3.8 COMMAND 0x10**

**Open Stepper Motor**

Command to open the stepper motor that controls the flow of Nitrous Oxide to the combustion chamber. Opens the stepper motor at a fixed rate.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x10 | Open Stepper Motor | uint8 | 1 | 3 |

**2.3.9 COMMAND 0x11**

**Close Stepper Motor**

Command to close the stepper motor that controls the flow of Nitrous Oxide to the combustion chamber. Closes the stepper motor at a fixed rate.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x11 | Close Stepper Motor | uint8 | 1 | 3 |

**2.3.10 COMMAND 0x20**

**Set Parameters**

This command sets parameters for the operation of the Motor Controller. This should be called before running a test.

Parameters that can be changed include:

Data Transmission Rate - The wait time, in milliseconds, between the transmission of two data packets.

Max Fire Time - Total time that the Motor will be in Fire mode, after which it shuts off automatically.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Value | Description | Format | Bytes | Byte offset |
| 1 | --- | Header | --- | 3 | 0 |
| 2 | 0x20 | Set Parameters | uint8 | 1 | 3 |
| 3 | Varies | Data Transmission Rate | uint16 | 2 | 4 |
| 4 | Varies | Max Fire Time | uint8 | 1 | 6 |

**2.4 Checksum Algorithm**

TBD

3 Motor States

- Sequences

- Idle

- Idle, full

- Armed

- Control Loop

- Ignition Sequence

- Shutdown Sequence

- Nitrous bleed-off post test

- Nitrous filling sequence

- Nitrous bleed-off emergency