Universal Ground Station Receiver

For Hubsan X4 – CDR

Embedded Systems Laboratory EN.525.743.91

Kyle Mercer – 2/26/2018

Table of Contents

[System Description 3](#_Toc507326072)

[Capabilities 3](#_Toc507326073)

[Limitations 3](#_Toc507326074)

[Functional Description 4](#_Toc507326075)

[Overall System 4](#_Toc507326076)

[Ground Station System 4](#_Toc507326077)

[MCU Software Operation 5](#_Toc507326078)

[MCU Software Startup 5](#_Toc507326079)

[Quad over Bluetooth Universal Protocol (QoBUP) Developer ICD 6](#_Toc507326080)

[Overview 6](#_Toc507326081)

[Commands (from Transmitter) 6](#_Toc507326082)

[Overall Message Structure 6](#_Toc507326083)

[Command Blocks Structure 7](#_Toc507326084)

[Command Block Definitions 7](#_Toc507326085)

[Status (to Transmitter) 8](#_Toc507326086)

[Status Flag Definition 8](#_Toc507326087)

[Timing Considerations 8](#_Toc507326088)

[Materials & Resources 8](#_Toc507326089)

[Parts List 8](#_Toc507326090)

[Hardware 8](#_Toc507326091)

[Software 9](#_Toc507326092)

[Test Equipment 9](#_Toc507326093)

[Vendors 9](#_Toc507326094)

[Group Project Logistics 10](#_Toc507326095)

[Schedule 10](#_Toc507326096)

[Risk & Mitigation Plan 11](#_Toc507326097)

[References 11](#_Toc507326098)

# System Description

The Universal Ground Station Receiver is a headless embedded system which will bind with compatible Bluetooth devices, listen to incoming Bluetooth control packets, and re-broadcast control packets in a format that the Hubsan X4 (H107L) Quadcopter can understand. The ultimate goal is for the system to be able to accept virtually any Bluetooth slave device as a transmitter, enabling a broad range of standard compliant devices to potentially become a quadcopter controller with minimal software setup time. Examples of devices which could communicate with the ground station include (but are not limited to):

* A Bluetooth enabled personal computer
* A smartphone
* A Bluetooth handheld classic controller (Corrie’s project)
* A glove controller with Bluetooth adapter (Corrie’s project)

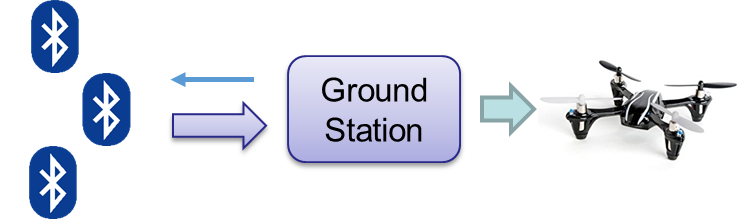


Figure : Top level control flow of system (Hubsan X4 pictured on right)

## Capabilities

The Ground Station Receiver will support the following capabilities:

* Perform command translation and relay communication for Quadcopter controls
* Any Bluetooth enabled device may pair
* Support a Quadcopter over Bluetooth Universal Protocol (QoBUP) communication scheme.
  + QoBUP has an accompanying developer ICD which will be developed as a part of this project
* Support a “training mode” for new user to become accustom to new interfaces and mode of control

## Limitations

In order to constrain the scope of the project, the Ground Station Receiver will have the following limitations:

* Only Bluetooth protocol will be support between transmitter and receivers
* Single Quadcopter brand supported (Hubsan X4 H107L)
* Only one quadcopter may be commanded by one transmitter
* Only the A7105 RF chip will be supported for the Quadcopter RF communication

# Functional Description

## Overall System

This project is only one half of the entire functional system. In order to fully realize the capability of the Universal Ground Station Receiver two different types of controllers will be implemented by another member of the class, Corrie Russell. She is the project partner and her half of the system is required for the full system implementation. See the CDR documents pertaining to Corrie’s controllers for specific implementation details on the transmitter side. Each controller designed will adhere to the QoBUP standard. Each transmitter will leverage the QoBUP to send controls to the ground station.

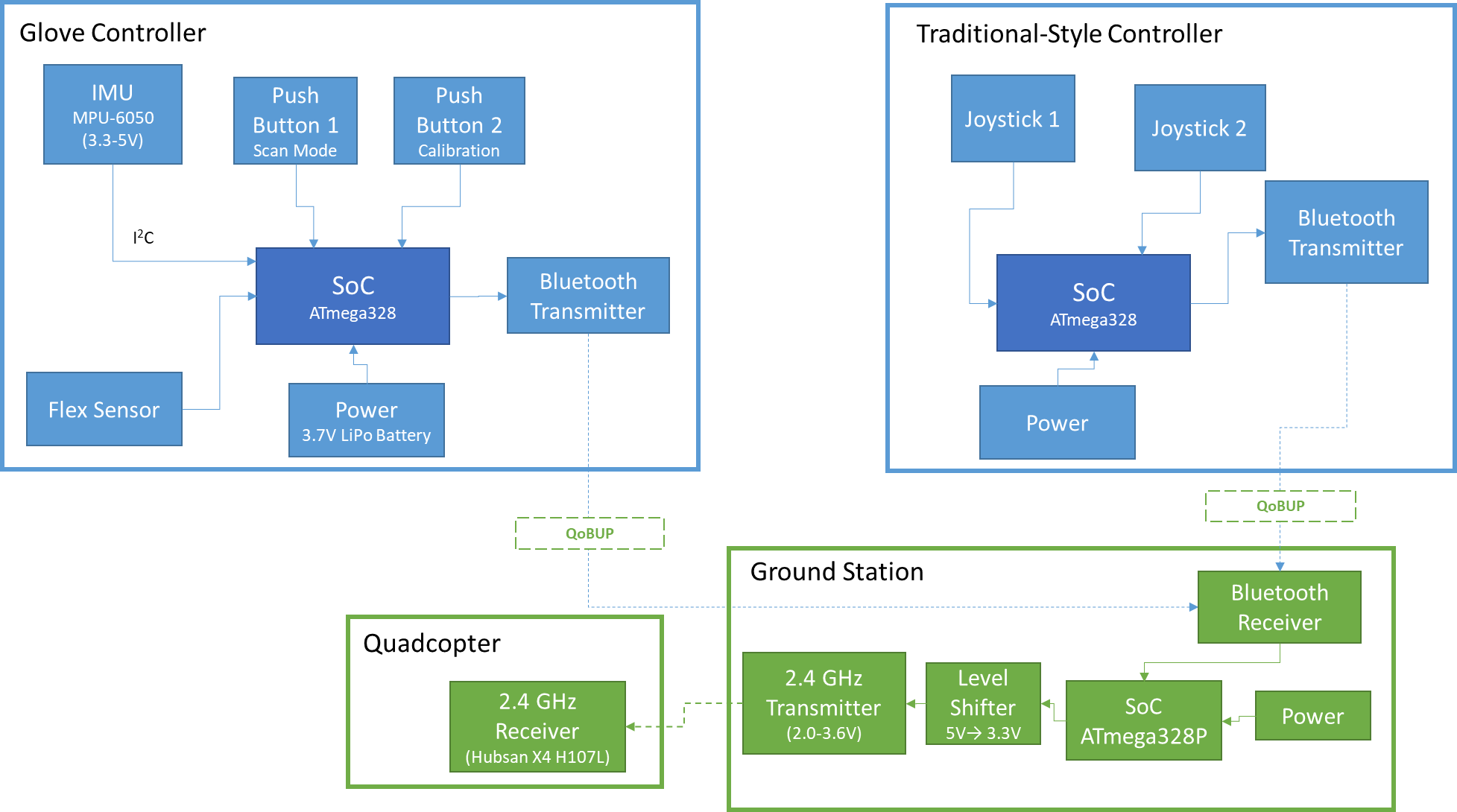


Figure : Overall system block diagram.

## Ground Station System

Upon reception of controls from the Bluetooth enabled transmitter the Microcontroller Unit (MCU) will read the contents of the incoming message over a standard UART interface which connects to the Bluetooth module receiver. As per the QoBUP ICD, a status message will also be transmitted back to the transmitter for each message received. Refer to the Quad over Bluetooth Universal Protocol (QoBUP) Developer ICD for specific details on about the QoBUP.

### MCU Software Operation

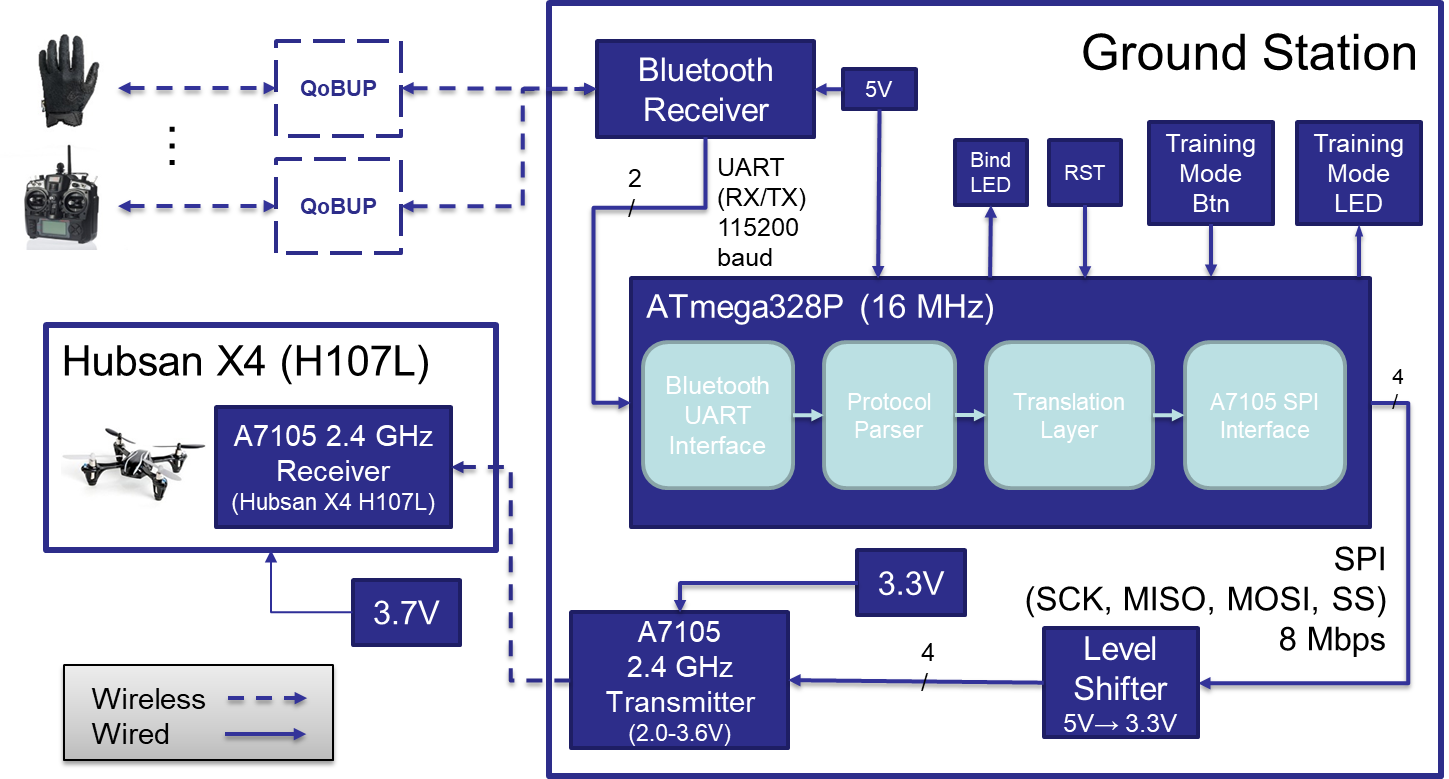
The MCU will read in control messages through the Bluetooth UART Interface into a buffer where it will get picked up by the Protocol Parsing Interface. At this stage, validation is done on the message to ensure the message adheres to the QoBUP ICD. Validation is complete once all sub-blocks are parsed out into common software structures and command processing is initiated via the Translation Layer. During this stage of software processing, the allocated structures are picked up by a translation stack which is written specifically for the target quadcopter, in this case, the Hubsan X4. The structures are then translated into a set of primitive register read/write commands which are then queued up into the A7105 SPI Interface. Controls are then dispatched to the A7105 over the SPI bus at a controlled rate dictated by the Hubsan X4 communication spec. The flight control update rate for the Hubsan X4 is 10ms (Hung, 2015). Finally, the SPI bus will need to be level shifted in order to accommodate the A7105 transmitter’s 2V-3.6V tolerance. The overall ground station hardware/software block diagram can be seen in Figure 3.

Figure : Overall ground station hardware/software block diagram.

### MCU Software Startup

During system initialization, the ground station will attempt to automatically bind to the Hubsan X4. The handshaking involved to complete the binding process is quite involved; however, the entire procedure has been decoded into a step-by-step process outlined on the internet (Hung, Reverse Engineering a Hubsan X4 Quadcopter, 2015).

An additional aspect of system startup will include configuration and initialization of the Bluetooth transceiver module. The Bluetooth module connected to the ground station will act as a Bluetooth master whereas all transmitters that wish to pair will act as the slave device. All communication (both configuration and controls) to and from the Bluetooth module will occur over UART connection set to an 115200 baud rate.

# Quad over Bluetooth Universal Protocol (QoBUP) Developer ICD

## Overview

This section describes the communication protocol between a potential Bluetooth transmitter and the Universal Ground Station Receiver. The ICD is designed to be extensible to accommodate other dimensions of control for quadcopters which may be more sophisticated than the Hubsan X4. At the time of writing this ICD several assumptions are being made about the transmitter/receiver pair. There are highlighted as follows:

* The protocol only provides functionality for one-way control communication, that is, transmitter to receiver
* Only status responses will be provided back to the transmitter for diagnostics/error checking
* All timing is based off of a serial (RX/TX) transmission rate of 115200 N8 between the Bluetooth module and the ground station processor
* Messages are made up of 8-bit words
* Command queuing is currently not implemented meaning if duplicate command block IDs are present in a single message then only the last command block will take effect
* A message’s contents will only take effect if no errors were detected in the message (ie. the returned status flags are all 0)

## Commands (from Transmitter)

### Overall Message Structure

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Message Header | Identifies the type of message |
| Message word count | Number of 8-bit words in entire message, including the header and footer |
| Message unique ID | A unique 8-bit value (used for status reporting) |
| Command block 1 |  |
| Command block 2 |  |
| Command block 3 |  |
| … |  |
| Message footer | Always 0xA5 |
| Footer length | Always 0x02 |

### Command Blocks Structure

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | Identifies the type of block |
| Message word count | Number of 8-bit words in entire message, including the header and footer |
| User defined block data |  |
| … |  |
| … |  |

### Command Block Definitions

#### Throttle Update

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | 0x00 |
| Message word count | 0x03 |
| Throttle value | 0x00 (no throttle) to 0xFF (max throttle) |

#### Yaw Update

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | 0x01 |
| Message word count | 0x03 |
| Yaw value | [0x00 (max left yaw), 0xFF (max right yaw)] (0x80 = no yaw) |

#### Pitch Update

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | 0x02 |
| Message word count | 0x03 |
| Pitch value | [0x00 (max pitch forward), 0xFF (max pitch backwards)] (0x80 = no pitch) |

#### Roll Update

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | 0x03 |
| Message word count | 0x03 |
| Roll value | [0x00 (max roll right), 0xFF (max roll left)] (0x80 = no roll) |

#### Flight Control Update

This block updates all flight controls in a single block command

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Block ID | 0x04 |
| Message word count | 0x06 |
| Throttle value | 0x00 (no throttle) to 0xFF (max throttle) |
| Yaw value | [0x00 (max pitch forward), 0xFF (max pitch backwards)] (0x80 = no pitch) |
| Pitch value | [0x00 (max pitch forward), 0xFF (max pitch backwards)] (0x80 = no pitch) |
| Roll value | [0x00 (max roll right), 0xFF (max roll left)] (0x80 = no roll) |

## Status (to Transmitter)

|  |  |
| --- | --- |
| **Word (8-bits)** | **Description** |
| Status header ID | Echoed from command’s “Message unique ID” |
| Status flags | Error flags from returned message processing |

### Status Flag Definition

|  |  |
| --- | --- |
| Bits | Description |
| 0 | Indicates an unknown message ID was received |
| 1 | Indicates an unknown block ID was received in message |
| 2 | Indicates a bad message size |
| 3-7 | Reserved |

## Timing Considerations

As stated in the assumptions list above, the interface between the Bluetooth module and the ground station processor is a 115200 N8 (RX/TX) UART port. This makes the theoretical max transfer rate of controls roughly 14,400 bytes per second. The max message size for this protocol is 256 bytes long including any headers and footer blocks. Thus, the maximum transfer bandwidth from the transmitter to the ground station can be:

These timing measurement should be kept in mind when designing controls for the Quadcopter you are targeting. For the Hubsan X4, the calculated numbers above are conservative as this Quadcopter requires flight control packets be sent every 10ms regardless of control values. The Ground Station will abstract these timing requirements away from the transmitter developer as it will do the necessary padding/combining of controls if the rate of controls being received is not the same as the timing requirements for control transmission to the Quadcopter. However, the developer should keep in mind the effects of sending controls at a faster or slower rate than their quadcopter calls for.

# Materials & Resources

## Parts List

### Hardware

The list of necessary hardware for the completed ground station is scoped out as follows:

* ATmega328P AVR MCU
* COTS Hubsan X4 (H107L) Quadcopter
* 9V LiPo batteries
* 5V regulator
* Capacitors/Resistors
* LEDs
* Prototyping board
* Insulated prototyping wires
* A7105 Wireless RF 2.4GHz Transceiver Module
* Bluesmirf Bluetooth modem (RN-42)
* 5V to 3.3V level shifter
* 2mm to 2.54mm pitch header adapter

The ground station will operate standalone with a 9V battery powering the entire system, including the MCU and peripherals.

### Software

The following list of development and operational software will be leveraged for the project:

* Linux (development environment)
* Arduino CLI toolchain
  + AVR compiler
  + SPI
  + UART
* Collaboration tools (to work with lab partner)
  + Git
  + Google cloud

All software, with the exception of the AVR support libraries, will be developed as a part of this project. Configuration management will be aided by the use of git as a means for collaboration between the ground station development and the transmitters being developed which is required for the overall system.

### Test Equipment

The following list of test equipment will be used for the project:

* Arduino Mega
* Volt meter
* Oscilloscope
* Breadboard
* Prototyping wires
* LiPo battery charger

## Vendors

The major vendors used for obtaining the above listed parts comprehensively include Amazon and E-Bay. All parts with the exception of the A7105 RF module are available on Amazon at their 2-day Prime shipping rate. The A7105 RF module has already been obtained, but had a lead time of approximately 15 days. Redundant parts (double) were ordered for the entire parts list to ensure zero downtime for loss of hardware due to part failure. In the event additional parts must be ordered (so long as it is not the A7105 part) the lead time will be, in theory, only two days.

## Group Project Logistics

To complement the ground station system two transmitters will be developed by Corrie Russell, the project partner. She will be developing two transmitters which implement the QoBUP, namely a glove controller and a traditional double joystick controller. Expected deliverables from this project for the controller implementations include the QoBUP ICD as well as an operational ground station for full system testing.

Conversely, the ground station will require both controllers from Corrie’s project to satisfy full system testing. Again, both controllers are required to implement the QoBUP.

# Schedule

The overall project schedule is strategically broken down by task in order to mitigate the highest impact risks associated with the completion of the project by the end of the 10 week period. See the Risk & Mitigation Plan sections for details on enumerated risk items.

|  |  |
| --- | --- |
| Task # | Objective |
| 1 | Design and finalize QoBUP ICD for handoff to Corrie |
| 2 | Implement Bluetooth receiver interface |
| 3 | Implement common protocol processing interface |
| 4 | Unit test Bluetooth communication (with implement transmitter, if available) and QoBUP processing |
| 5 | Create A7105 SPI wrapper interface for Hubsan communication |
| 6 | Establish communication with Hubsan Quadcopter (binding phase) |
| 7 | Unit test basic A7105 RF interface to ensure proper quadcopter fight control communication |
| 8 | Implement translation layer |
| 9 | Verify full ground station via unit testing (with Corrie’s traditional style transmitter) |
| 10 | Implement “training mode” feature |
| 11 | Test full system with Glove/traditional controller driving control data |
| 12 | Move prototype to final ruggedized hardware assembly |
| 13 | Final full system test |

# Risk & Mitigation Plan

There are several risks identified for the development and testing of the ground station receiver.

* Development of the QoBUP protocol
  + **Low probability & low impact**

The QoBUP ICD is a required dependency of both the ground station’s Bluetooth receiver interface as well as the transmitter’s Bluetooth interface. As part of risk reduction efforts, the task for developing the ICD was placed first in the schedule. However, if this risk becomes realized more time will be allocated to this task. This risk has low probability of occurring with low impact to project schedule. An initial design of the [QoBUP](#_Quad_over_Bluetooth) is included as part of this CDR.

* Binding to the COTS Hubsan Quadcopter with the A7105 RD module
  + **Low probability & medium/high impact**

Establishing the initial binding to the Hubsan quadcopter is required before flight control may be sent to the device. Luckily, the binding process has been completely reversed engineered for the original transmitter by an online group who have published the steps necessary to setup the A7105 RF module and establish proper RF communication with the quadcopter. See the References section for details on the protocol. If the task of binding the purchased A7105 RF module with the Hubsan proves to be unsuccessful then the original controller may be used as a surrogate for getting past the binding process, though this may add scope to the project and prolong the schedule. Based upon the research conducted on the published protocol, it appears to have a high success rate among those who have used it. Thus, this risk has low probability of being realized with medium-to-high impact on the project.

* Program flash of the ATmega328P insufficient
  + **Medium probability & low impact**

The full software binary may be too large to fit into the ATmega328P’s 32KB flash memory. If the final pieces of software written end up generating a binary larger than 32KB then the mitigation strategy will be to upgrade to the ATmega2560 MCU (256KB). The compiler toolchain being used fully supports the fallback MCU with next to no software changes required. If an MCU upgrade is required then hardware changes should only be limited to rewiring (eg. no power requirement changes). This risk has medium probability of occurring with low impact to the project.

# References

The following references are critical to the success of ground station receiver.

* **Hubsan X4 reversed protocol specification:** This text document gives a breakdown of the necessary setup and binding steps required to connect to the Hubsan X4 with an A7105 RF chip. This will be used as a guide when writing the A7105 SPI interface software.
  + <http://www.jimhung.co.uk/wp-content/uploads/2014/11/HubsanX4_ProtocolSpec_v1.txt>
  + <http://www.jimhung.co.uk/?p=1349>
* **Deviation10 firmware (targets 32-bit ARM):** The Deviation10 framework is an open source, aftermarket firmware written specifically for the DEVO brand controllers (ARM-based). The controller must be hardware modded and loaded with the firmware prior to use. If loaded, the firmware essentially provides controls for a wide variety of quadcopters and small UAV planes. Since it provides support for the Hubsan quadcopters, this could be a helpful resource for reference on establishing communication with the Hubsan X4, despite the difference in target MCU architectures.
  + <https://bitbucket.org/PhracturedBlue/>
* **Bluetooth modem RN-42 AT command set:** This source provides documentation for interfacing with the Bluetooth module.
  + <https://cdn.sparkfun.com/datasheets/Wireless/Bluetooth/bluetooth_cr_UG-v1.0r.pdf>
* **Atmel’s AVR documentation:** This reference provides the necessary documentation about the MCU being using for the ground station.
  + <http://www.microchip.com/wwwproducts/en/ATmega328p>
* **ATmega328P standalone assembly:** This reference provides information on how to implement the ATmega328P MCU on a breadboard for prototyping purposes.
  + <https://www.arduino.cc/en/Main/Standalone>