Crazyflie Swarm Research

Nick Robbins - Project Lead

Cole Beaulieu - Communications Lead

Jacob Frazier-Florres - Web Design Lead

Grant Manley

Ben Nelson

Chengrui Yang - Ideas Lead

Tianxiang Shen

Contents

1 Introduction

1.1 Project Statement

The Quadcopter swarm project revolves around creating a swarm of Crazyflie quadcopters. As it stands currently, the Crazyflie open source platform enables the user to fly one or two crazyflies with one included radio. In order to fly more, we need to change how the radio and ground station process these crazyflies. Once we get more than two flying we are going to also implement crazyflie to crazyflie communication. We also have the benefit of a camera system to tell us exactly where these crazyflies are located.

1.2 Purpose

The implementations for this project are extremely open. With a swarm of quadcopters being controlled as a group or individually the user could achieve things such a carrying a heavy or strangely shaped load, choreographing a multi camera scene, as well as recreational uses like programming them to perform aerial acrobatics.

1.3 Goals

We have multiple goals that will build upon each other:

1. Implement a wifi adapter instead of using the Crazyflie Radio
2. Remove the acknowledgement (that exists in the radio systems) the ground station waits for.
3. Implement the BigQuad expansion deck to expand crazyflie size.
4. Enable Peer to Peer communication over the Crazyflie's
5. Be able to lift a load with multiple Crazyflie in sync
6. Control 5-10 Crazyflie at one time

2 Deliverables

Hardware:

- 5 – 10 functional quads, each consisting of a crazyflie controller, expansion board, enlarged frame and motors, and an ESP8266 Wifi chip
- A compatible Wi-Fi enabling device for the ground station

Software:

- New firmware for the ESP WiFi chips that is capable of point to point, broadcast, and mesh communication (receiving and transmitting).
- Firmware adaptation for the crazyflies that allow them to receive communication from the ESP Wi-Fi chip.

- New ground station code that enables communication over the new Wi-Fi device.

3 Design

The Design of the project will revolve around the ability to produce a large quad-copter while at the same time creating a swarm controlling system that can fly multiple large quads. The main focus areas will be around the construction of the quad, radio communications, and electrical design systems for the power systems.

In order to achieve the project goals, we have two options: make alterations to the current communication stack or scrap the current system and implement something new.

Alter Current System:

Benefits:

- Utilizes previous work done on the project, which could potentially save time
- Doesn't require the purchase of any new communication links.

Negatives:

- System was not designed for flying multiple quads at once and therefore is not easily translatable to our project.

To alter the current system, we would either need to be changing frequencies on the fly or implementing another form of multiple access (TDMA, CDMA). Changing frequencies on the current hardware proved to consume too much time to reliably fly the quads and the other forms of multiple access were either too complex or impossible to implement on the current radio. For these reasons we elected to scrap the current communication system and implement one chosen to specifically satisfy our needs.

Implement New System:

Benefits: System can be chosen and designed to meet our exact needs.
Negatives: More overhead to implement.

After researching multiple options, we decided to implement the system using ESP8266 Wifi modules because the lab has had success using them in the past, with their simplicity of implementation and ability to work with the current crazyflie radio's, they will greatly ease the conversion process they are simple to implement, and they work in a similar fashion to the current crazyflie radios, easing the conversion process.

3.1 System specifications

The specifications that we are given to adhere to in this project are:

- Create a large scale quadcopter from scratch using non-custom parts and hardware

- Redesign the radio communication systems to be able to talk to multiple quadcopters with a given radio dongle
- Implement code into firmware so that quad to quad communication can occur if necessary
- Use the camera systems provided to help log data with the tracking systems for giving 3D spacing requirements

3.1.1 Non-functional

- Fit up to 10 flying small diameter quads in the given lab environments
- Nearly zero complete failure (crashing) rate
- Low command failure rate (failing to follow a command without crashing)
- High data transfer rate for quad to quad and ground station to quad communications
- Easy to use user interface

3.1.2 Functional

- Fly 10 quads at once
- Give precise commands to each quad individually
- Broadcast messages to all quads
- Quad to quad communication
- Coordinate multiple quads to perform a simple task

3.2 PROPOSED DESIGN/METHOD

Our proposed design method is to initially use the provided the hardware and software that is given to us and initially figure out how it works. Then we will try to modify the code either my introducing new hardware and software with new units until we get the ability to fly multiple large quads with a single dongle at the same time. The Software team of the project is currently looking in streamlining the communication software to improve the ability of multiple communication points. While the Hardware team is building a new quad that will have faster and better hardware to enable the software team to better improve their systems.

3.3 DESIGN ANALYSIS

Our initial approach was to modify the current ground station software to improve the efficiency of the CrazyRadio Dongle. When we started, the radio was communicating to different quadcopters by switching to a unique channel for each quadcopter. Channel switching is a costly operation, so we thought that by keeping all of the quadcopters on the same channel and using separate addresses, we would be able to support more quadcopters. We also wanted to remove the acknowledge packet used in the current protocol to improve performance.

However, as we tried to implement these changes, we discovered several things:

1. The library we were using to interface with the CrazyRadio in C++ was not fully implemented, poorly documented, and not designed to handle modification well. This made all changes to the ground station software take significantly longer than expected.
2. The CrazyRadio Dongle is difficult to interface with.

3. The acknowledge packet is integral to the operation of a lower-level protocol used by the system, and its removal would be non-trivial.

These discoveries were not all bad news, however. This work gave us a much more thorough understanding of the system as a whole, including the fact that it would likely be possible to totally bypass the current radio setup. We decided to use small, inexpensive ESP8266 WiFi module, which we could connect to the ground station and to the STM32F405 microcontrollers on the quadcopters.

This would allow us to use mature, well-developed libraries for the communication between the ground station and the quadcopters (rather than make changes to the existing protocol). The WiFi modules should enable the ground station to support many more quadcopters than at present. To implement we would only need to create two adapters. First, we would write one in the ground station to convert CRTP packets to something the Wifi module can send. Second, we would write one in the STM32F405 firmware to convert from what the WiFi module receives to the CRTP packets the STM32F405 firmware currently expects.

This current plan should allow us to support the required number of quadcopters while also minimizing the amount of new code we need to design, implement, and test.

4 Testing/Development

4.1 HARDWARE/SOFTWARE

Indicate any hardware and/or software used in the testing phase. Provide brief, simple introductions for each to explain the usefulness of each.

The hardware that we are using for testing this project include:

Arduino Uno - Enables us to power the wifi module as well as pass data through a serial port

Breadboard - Enables us to set up the circuit needed to flash firmware onto the wifi module

Function Generator - Used to test the motors and esc that will be used on the Big Quad

FTDI Programmer - Used to connect the computer's usb port to the wifi module to view packets being sent to the module

The software that will be used in the development of this project include:

ESP8266 Mesh Library - A previous mesh code was used to test if our modules have the capability for connecting to multiple other modules

Current C Client Code - This was used as the basis for what we build everything off of.

4.2 PROCESS

Testing without ACK:

Removing the acknowledgment sent back to the ground station from the crazyflie after a packet has been received must be tested from two sides, the crazyflie and the ground station. To test that the ACK is turned off from the crazyflie side, we print out the header file on the packet being sent back to the ground station. Within this header file, we can see the bit that relates to the ACK.

Next, we need to check if our system can run without having the ACK turned off. To do this we compile the code, turn off the ACK with a command, and wait for the loop to fail. We realized that we could make it through the initialization stage without the ack, however, when we tried to send commands the loop would fail. We believe this problem will need to be solved in the crazyflie firmware instead of our PC Client.

Testing Broadcast and Individual Messaging:

Below (figure 4) you can see the setup we used to test if our code for sending broadcast packets (packets sent from one computer to multiple modules at one time) worked. We sent a message from our broadcast PC, then checked the serial monitors configured on both individual PC's to check if they both received it. We then wanted to check if we could send an individual message without the other module receiving it. To accomplish this, we sent an individual message from the broadcast station and checked if the specific module received it as well as if the other module did not. Both of these tests were confirmed. This was a large step in our project as we essentially eliminated the lag problems we had with the last radio.

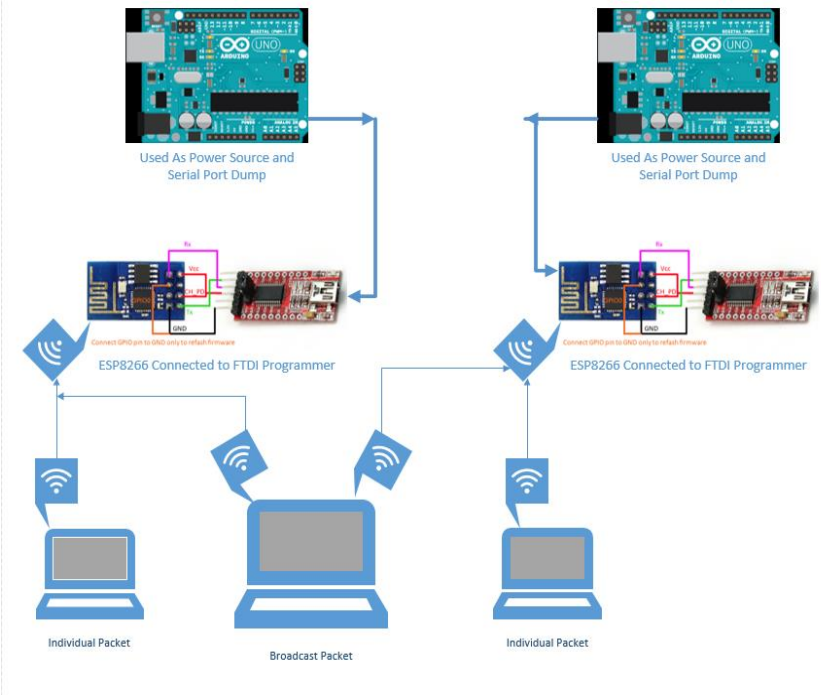

Figure 1

Testing Multi Module Mesh Networking:

Below is a flowchart showing how we tested our mesh network. You can see that we configured our setup to send a packet from our ground station computer to one of the modules. This module then sends the same packet on to the next module. This module then sends back an acknowledgement that it received this packet and that acknowledge is sent back to the ground station computer

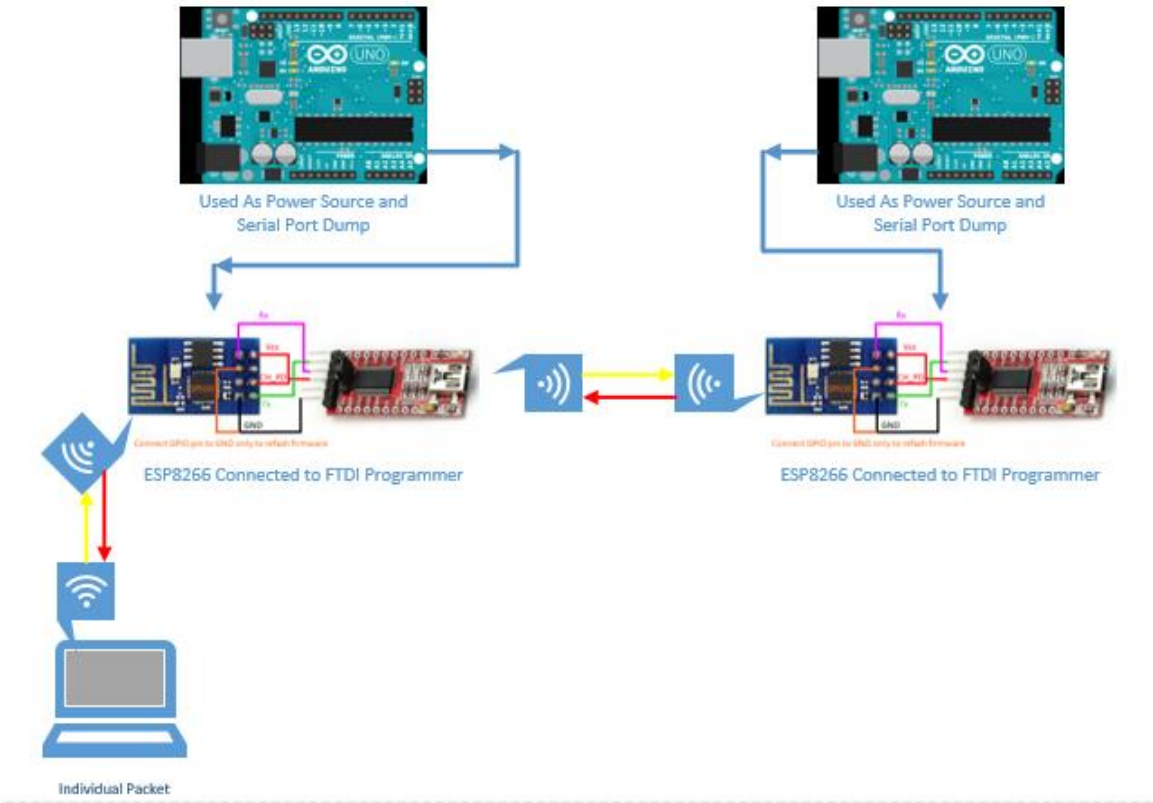

Figure 2

Testing BigQuad ESC and Motors:

To test if the ESC and Motors we are going to implement on our big quad were functioning, we connected them to a batter to serve as the power source as well as a function generator to mimic the function giving off by the controller we are using. We used the below settings for our function generator.

Amp: 3.3V

Freq: 400 Hz

Duty Cycle: 40%

Offset: 1.6V

We were able to see each of the 4 motors spin along with varying speed by adjusting the duty cycle.

5 Results

Hardware:

Testing Big Quad: We used a previous obsolescent quadcopter as our prototype model at first. At first, we tested the motor and ESC on it and all the four motor and ESC systems worked well. And the thrust seems big enough for the use of our ideal lifting loads in the future test.

Implement the expansion board: We got our expansion board and implemented it on our big quad. By now, the motors and ESCs on the big quad can work via the expansion board and can be controlled by the radio functionally, but the stabilization and balancing of this big quad are not yet ensured. We will continue to integrate and improve these characteristics

Software:

Radio communication: After going deeply through The CrazyRadio Dongle, we found it is difficult to interface and accomplish communication precisely and functionally with our requirements. Thus we are replacing the radio with the ESP8266 WiFi module.

We realized that we can initialize the system when the ACK is turned off, however we meet some problems in sending the commands, and this problem will need to be solved in the crazyflie firmware.

Wifi module:

Confirmed that we can send an individual message without other modules receiving it. Finished the testing of mesh network. To be specific, we successfully send a packet from ground station to one module and this packet can be forwarded to the next module. Also a feedback of acknowledgement will be sent back to ground station.

Website:

Just getting started on work with the website and we'll work on it for the next few weeks.

6 Conclusions

So far we have tested and proven the feasibility of almost all aspects of our proposed design including: utilizing the ESP8266 Wifi module to build a local network, evaluating the crazyflie firmware ability to adapt to the new communication link, and building an enlarged quad prototype to show the new hardwork working in unison. These test have proven that we will be able to use our proposed system to build a communication network capable of flying upwards of 8 quads at once and use them to perform uniformed tasks. It has also shown that quad to quad communication is a feasible goal for future iterations of the project. Furthermore, our proposed solution of building a new communication stack is the best solution for the following reasons: the current communication stack is incapable of reliably achieving our goals, the Wifi modules are specifically designed to build local networks of this type therefore making the implementation of such easier on our software team, and the Wifi modules are similar enough to our current links to ease the adaptation process in our firmware and ground station.

7 References

ESP8266 Library: https://github.com/esp8266/Arduino/blob/master/doc/esp8266wifi/readme.md

Bitcraze Documentation: https://www.bitcraze.io/

Crazyflie Wiki: https://wikis.ece.iastate.edu/distributed-autonomous-and-networked-control-lab/index.php/Crazyflie_Swarm