EE CprE 491 – Fall 2019
MicroCART Senior Design Team
Weekly Report 6

Oct 28st - Nov. 4th
Faculty Advisors: Phillip Jones, Matt Cauwels, James Talbert

Team Members:

Evan Blough -- Technical Team Lead, Embedded Software Lead
Kynara Fernandes -- Ground Control Station Lead
Aaron Szeto -- Controls Lead
Joe Gamble -- Embedded Hardware Lead
Shubham Sharma -- Crazy Fly Implementation Lead, Website Manager
Jacob Brown -- Physical Hardware Lead

Summary for Progress this Week

This week we tried to make progress on software development for all related areas (Matlab, GCS, and Embedded Software/Hardware). Several of us ran into issues. We are going to continue to research solutions to these issues. We continued to develop hardware on the drone, but one connector broke.

Past Week Accomplishments

- Identified and fixed Data tool plotting issue, updated the Data tool guide to match current code - Aaron
- Crazyflie maintenance - Joe
- We looked into the Refactor Control issue. We want to finish James' development on this feature. We went to the structs_cleanup branch and tried to compile the code onto the Zybo to test - Evan, Joe
- Fixed the readme for running Vivado - Evan
- Investigated connection between the GUI and Backend and how to apply this to error display feature - Kynara, Evan
- Figured out the compilation issue with the Data visualization tool and fixed the code. - Shubham

Pending Issues

- Dx6 Controller still broke. I didn't have time to research how to fix it.
- Structs_cleanup embedded software showing a compilation error. Xtmrctr.h not found. Haven't researched this extensively yet.
- Trying to find a signal that is sent from the backend to a slot on the GUI. Naming conventions between the frontend and backend aren't the same, so it is difficult to find where this camera data is being updated.
- CF Adapter contains incomplete code and missing files. Will have to look into a different approach to use the CF.
-

Individual Contributions

Team Member	Contribution	Week 3 Hours	Total Hours
Evan Blough	Organized Drone assembly parts. Looked into the Refactor Control issue with Joe. Worked on Ground Station error calculation widget with Kynara. Worked on Design Doc.	9	62
Kynara Fernandes	Tried to figure out communication of components. Consulted Evan because I seemed to be getting nowhere. Design doc and lightning talk.	7	52.5
Joe Gamble	Soldered on some new buttons and thought up design ideas for crazyflies. Aided in the investigation of quad code issues	6	41
Jacob Brown	Got current reading from motors, calculated draws, and organized lab equipment.	8	34
Aaron Szeto	Fixed Data tool plotting issue and added comment warnings about the problem code line. Updated the Data tool guide to match the current code.	8	40
Shubham Sharma	Attempted to run the CF adapter and ran into multiple compilation issues. Working on a different approach to fly the CF. Fixed the bug with the data visualization tool.	3	46

Plans for Coming Week

- Finish Attaching shield board cables to second drone platform
- Finish Assembling Drone and test with master embedded build
- Finish implementing Error display on GUI
- Get the updated Data Tool code/guide on the github

Summary of Weekly Advisor Meeting:

During this meeting with Dr. Jones we discussed several issues we were having. We talked about how one of the RC controllers broke. We talked about some of our difficulties with the computers in the lab. We talked about some issues with the Data Analysis Tool. We talked about adding a flight mode indicator LED to the drone. We talked about saving all current drone parameters on the drone to a text file.

Appendix:

Edit to run Vivado / Xsdk readme:

```
44  order to launch the
45  program
    -     - In a terminal, enter `source /opt/Xilinx/2018.2/Vivado/2018.2/settings64.sh`
46  +     - In a terminal, enter `source /remote/Xilinx/2018.3/Vivado/2018.3/settings64.sh`
47        - In terminal type `xsdk &`
48  1. ISU Remote Linux Servers (linux-X, research-x.ece.iastate.edu)
```

CF Research Paper on flying multiple drones:
https://act.usc.edu/publications/Hoenig_Springer_ROS2017.pdf

Flying Multiple UAVs Using ROS

Wolfgang Hönig and Nora Ayanian

Department of Computer Science,
University of Southern California, Los Angeles, CA, USA
whoenig@usc.edu
ayanian@usc.edu
http://act.usc.edu

Abstract. This tutorial chapter will teach readers how to use ROS to fly a small quadcopter both individually and as a group. We will discuss the hardware platform, the Bitcraze Crazyflie 2.0, which is well suited for swarm robotics due to its small size and weight. After first introducing the `crazyflie_ros` stack and its use on an individual robot, we will extend scenarios of hovering and waypoint following from a single robot to the more complex multi-UAV case. Readers will gain insight into physical challenges, such as radio interference, and how to solve them in practice. Ultimately, this chapter will prepare readers not only to use the stack as-is, but also to extend it or to develop their own innovations on other robot platforms.

Keywords: ROS, UAV, Multi-Robot-System, Crazyflie, Swarm

Design Document:

MicroCART

Design Document

Team Number: 50
Client: Dr. Phillip Jones
Advisers: Matt Cauwels, James Talbert
Team Members:

- Evan Blough -- Technical Team Lead, Embedded Software Lead
- Kynara Fernandes -- Ground Control Station Lead
- Aaron Szeto -- Controls Lead
- Joe Gamble -- Embedded Hardware Lead
- Shubham Sharma -- Crazy Fly Implementation Lead, Website Manager
- Jacob Brown -- Physical Hardware Lead

Team Email: sdmay20-50@iastate.edu
Team Website: http://sdmay20-50.sd.ece.iastate.edu/
Revised: 09/04/20

Development Standards & Practices Used

Code documentation

Follow the Principles of Programming

IEEE Code of Ethics

VHDL Coding Standards

IEEE Floating Point Standard

Wireless Networking – "WiFi"

Coding Standards for High-Confidence Embedded Systems

Summary of Requirements

List all requirements as bullet points in brief.

1) Integration of a **secondary** quadcopter into the high-speed camera system,

2) Extend modular design to enable testing advanced controllers and signal processing,

3) Design and develop capabilities to support the implementation of the real world application,

4) Design and implement a mock real-world demo for visitors.

Applicable Courses from Iowa State University Curriculum

List of Iowa State University courses which were applicable towards the MircoCART project:

- CPRE 288:
  - This course:
    - Goes over elementary embedded design flow/methodology
    - It gives students a basic understanding into micro-controllers.
    - Applications laboratory exercises with embedded devices.
  - We work with microcontrollers a lot with the project as evident from the project title. Hence, this course is a great foundation to assist us in the project.
- EE 230:
  - This course:
    - Is an overview of circuitry design and analysis
    - Gives students experience working with laboratory instrumentation and measurements
  - This is an applicable course as the project consists of circuit design knowledge from 230.
- CPRE 381:
  - This course:
    - Is an Introduction to computer organization, evaluating the performance of computer systems
    - Datapath and control, scalar pipelines, introduction to memory and I/O systems
  - The project consists of ARM-processors we would need to work with. And since this course is centered heavily around processors it is helpful towards the project.
- EE 330:
  - This course:
    - Goes over semiconductor technology for integrated circuits.
    - Analysis and design of analog building blocks.
    - Laboratory exercises and design projects with CAD tools and standard cells.
  - The project would require an understanding of circuit issues such as unmatched impedance and circuit devices such as current mirrors. Which is why this course is applicable for the project.
- EE 224:
  - This course:

        - Introduction to using Matlab for EE work
        - Analysis of Signals and various signal systems like LTI
    - This class teaches how to use Matlab and quadcopter controls are entirely based in Matlab hence it is helpful for the project.
- EE 321:
    - This course:
        - Continuation of EE 224
        - Focuses more on modulation and data transmission
    - Focuses on data transmission which applies to how the quadcopter communicates with the controls and ground station, which is heavily used in the project.
- CPRE 488:
    - This course is about:
        - Embedded microprocessors and embedded memory
        - Component interfaces communicating to embedded software
        - Platform-based FPGA technology w/ hardware synthesis
        - And Real-time operating system concepts
    - This project requires knowledge with VHDL and C development on Xilinx platforms, multidisciplinary projects, embedded system design, and control systems on drones. Hence, why the course is applicable to the project.

New Skills/Knowledge acquired that was not taught in courses

Qt software libraries

RC communication

PID Controls

Table of Contents

1 Introduction 4

1.1Acknowledgement 4

1.2 Problem and Project Statement 4

1.3 Operational Environment 4

1.4 Requirements 4

1.5 Intended Users and Uses 4

1.6 Assumptions and Limitations 5

1.7 Expected End Product and Deliverables 5

2. Specifications and Analysis 5

2.1 Proposed Design 5

2.2 Design Analysis 6

2.3 Development Process 6

2.4 Design Plan 6

3. Statement of Work 6

3.1 Previous Work And Literature 6

3.2 Technology Considerations 7

3.3 Task Decomposition 7

3.4 Possible Risks And Risk Management 7

3.5 Project Proposed Milestones and Evaluation Criteria 7

3.6 Project Tracking Procedures 7

3.7 Expected Results and Validation 7

4. Project Timeline, Estimated Resources, and Challenges 8

4.1 Project Timeline 8

4.2 Feasibility Assessment 8

4.3 Personnel Effort Requirements 8

4.4 Other Resource Requirements 8

4.5 Financial Requirements 9

5. Testing and Implementation 9

5.1 Interface Specifications 9

5.2 Hardware and software 9

5.3 Functional Testing 9

5.4 Non-Functional Testing 9
5.5 Process 10
5.6 Results 10
6. Closing Material 10
6.1 Conclusion 10
6.2 References 10
6.3 Appendices 10

List of figures/tables/symbols/definitions (This should be similar to the project plan)

Term	**Definition**
CLI	Command-line interface
Continuous Integration	An automated process of running tests on every commit to the repository
Demo	Short for demonstration; this is one of the deliverables of the project: a demonstration of the quad's capabilities, for example, doing a backflip with the quad, finding an object and following it, communicating with a second quad to perform flight patterns

FPGA	Field Programmable Gate Array; this is a board that can be programmed to simulate electrical hardware components. Used often in reconfigurable computing
GCS	Ground Control Station, the application that runs on a host computer that communicates with the quad via a Wi-Fi connection and sends its coordinates to the quad
GUI	Graphical user interface
IR	Infrared wavelengths of light longer than visible light; used in the VRPN system to determine the position of the quad
LIDAR	Light Detection and Ranging; this is a system for determining the altitude (z) of the quad using the onboard sensor
Optical Flow	A system using pattern of motion of objects, surfaces, and edges caused by the relative motion between the and the scene to determine position; used by the quad to calculate the position (x, y) when not in the lab using the VRPN system
PID	Proportional-integral-derivative control system; standard control algorithm used on the quad
Quad	Short for quadcopter; this is the hardware platform we use in this project
Setpoint	In a control system, the target value for an essential variable
VRPN	Virtual-Reality Peripheral Network; this is the system used to determine the position (x, y, z) and orientation (φ,θ,ψ) of the quad in the lab using a set of 12 stationary cameras and an IR transmitter on the quad

1 Introduction

1.1 Acknowledgement

The development of this project has been aided by ECPE Faculty and graduate students at Iowa State. They offer valuable technical advice and insight. We would like to acknowledge their contributions to this project below.

- Matthew Cauwels
- Dr. Phillip Jones
- James Talbert
- May 2020 MicroCart Team

1.2 Problem and Project Statement

1.2.1 Problem Statement

MicroCART is a drone platform that will be used by graduate students to perform research on embedded systems and controls topics. The platform will also need to demonstrate its operation to professionals or future students to showcase the utility of skills obtained through the course of a EE/CPRE/SE degree. This platform will also need to be maintainable and modifiable by future senior design teams. Currently, the MicroCART platform is functional, and it will be extended to a copy of the current platform to implement swarm flight. In order to achieve these objectives, we will need to copy, refine, and modify the current platform to have more intuitive interface and more accurate flight performance.

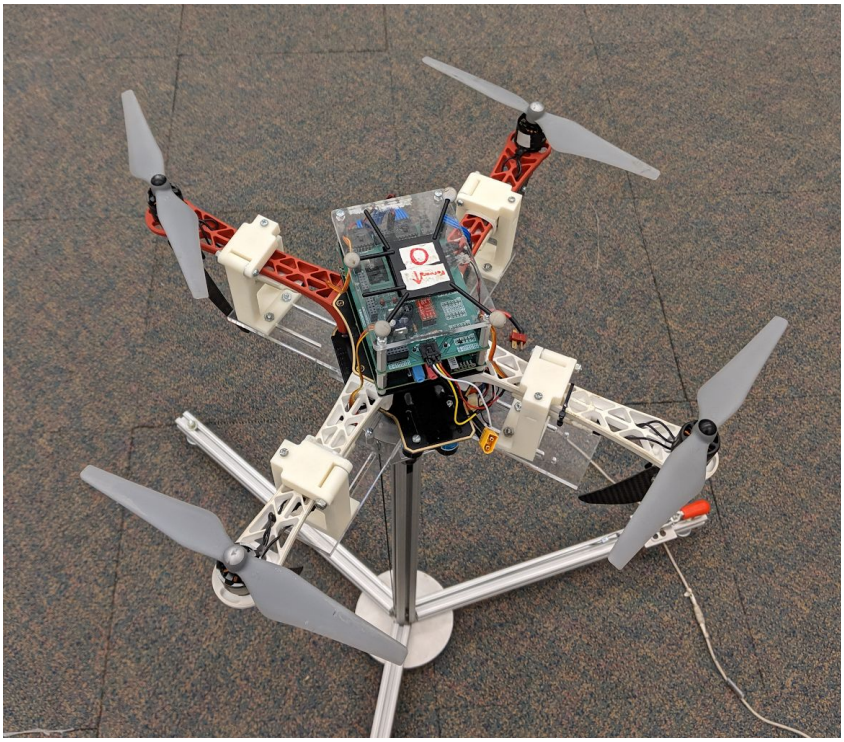


Figure 1.2.1: Current drone implementation

1.2.2 Solution Approach

We plan to further our knowledge of the previous development done on the project. We will need to read current documentation of the drone's subsystems and infer the functionality of portions that aren't documented. We also elected focus areas for each team member to specialize in areas of drone development. Once each of us are familiar with our respective subsystems, we can start building the second platform.

There is a smaller drone platform that we can use to familiarize ourselves with navigation. We hope to implement swarm flight with these smaller drones, and once we have experience with the smaller drones we will be able to move on to navigating the larger drone. By the end of the first semester, we hope to have both large drones operating with swarm flight.

The second semester would then consist of us adding new features to the drone to fulfill the purposes outlined above. The controls and embedded software leads will work on making more intuitive source for swapping controls algorithms on the drone. The GUI leads will be working on creating an easy to use interface for the drone. The embedded hardware will generate fast feedback loops to make the control systems more efficient and regulate memory use for each control algorithm. The hardware lead will be… As a team we might work to add Linux to the second core to promote development on our platform. This second Linux core could have high level C++ libraries like OpenCV that could be used for soft real time object tracking.

1.3 Operational Environment

The operational environment for the quad is the inside of Coover 3050, and this room has a VRPN camera system that can be used for drone navigation. For indoors operation there will not be any hazardous environmental factors like extreme heat or cold. The environment in the lab will have people in it, and the drone could collide with expensive equipment in the lab, so failsafes for flight faults are important considerations for our operating environment. Due to the fact that the operation environment is identical to its use environment, we expect little design difficulty caused by the operational environment

1.4 Requirements

The functional requirements for this project are building our quadcopter, get it flying at the same time as another quadcopter, and modifying the controls system to be compatible with other control systems. Some economic/market requirements include last years quadcopter, quadcopter replacement parts, and spare Lithium Ion batteries. The environmental requirement is that the quadcopter should be able to fly in an outdoor environment. UI requirements include making the control systems more user friendly.

1.5 Intended Users and Uses

Platform will be used to test different control systems and their varying efficiencies for students. Student will require quick, simple interfaces to maximize learning and reduce time use for lab. The system will also be utilized in future senior design projects, requiring our team to use good documentation and written communication techniques. Finally, the project will be utilized in college demonstrations as a means of showing off to our employers what our college can do as well as an attempt to recruit future students.

1.6 Assumptions and Limitations

1.6.1 Assumptions

- We won't have more than two large quadcopters in our operating environment
    - Further maybe hazardous/ too complicated
- The operating environment indoors doesn't contribute any significant hazards
    - Humans acting in a hazardous manner might complicate this
- We are assuming the shield board added to the drone works properly
    - Not tested by the previous team

1.6.2 Limitations

- Drone flying environment limited to the area of the camera sensors within Coover 3050
    - The flight is dependent upon cameras tracking drone
- The computational power of the drone
    - Limited in processing time
    - Limited in memory access speed

- Feedback delays in control systems by the camera systems
    - Delay in ms expected
    - From Wi-fi communication
- Battery to payload ratio imposes limitations on overall runtime
    - Battery technology and weight characteristics of drone limit flight to appr. 5 minutes
    - Frequent replacement of battery systems necessary

1.7 Expected End Product and Deliverables

Our product is academic in nature so we won't have commercialized descriptions. We will need to provide detailed documentation through gitlab documentation and documentation generators like doxygen.

1.7.1 Implementation of Second Drone

We are going to make an exact copy of the current drone platform from the previous year. Most of the work done on this portion will be in the form of research on the existing system. In order to make another platform we will need to research the existing documentation. We are making this deliverable, because we need to have one stable platform to demo our solution to people and one for a senior design team to do development on. This deliverable should be completed by October 25th.

1.7.2 Swarm Flight of Crazy Flies

After we have a second drone, we will hopefully have access to several crazy flies. We can use these platforms to test out extending our GCS software to multiple platforms. This is an intermediate deliverable. We will use this deliverable to get experience with programming swarm flight with inexpensive platforms and move on to our permanent solution for swarm flight. We should have this portion completed by November 1st.

1.7.3 Swarm Flight of large drones

After we have experience with swarm flight on the crazy flies and have the second drone built, we will work on having two drones operate simultaneously. This deliverable relates to our client need of demoing to students and faculty. Showing multiple vehicles operating simultaneously is more impressive than just a single vehicle. This showcases the usefulness of skills obtained with an ECPE degree. Controls students also will want to test control algorithms with multiple platforms interacting. This should be done by December 2nd.

1.7.4 Addition of Linux to Second Core

After we have both drones fully implemented and functioning properly, we will need to add new features to our board to showcase skills that we have learned from our degree. One of the features we plan on adding is Linux to the second core. The Zybo board has two ARM cores. We will install Linux on the second core and this will give researchers and future project teams greater flexibility to control our platform. We can also use it to view the value of a degree. We can install an image detection library like OpenCV and a camera on Linux and use image detection to give control commands to the drone. This will be done at some point during the second semester of this course.

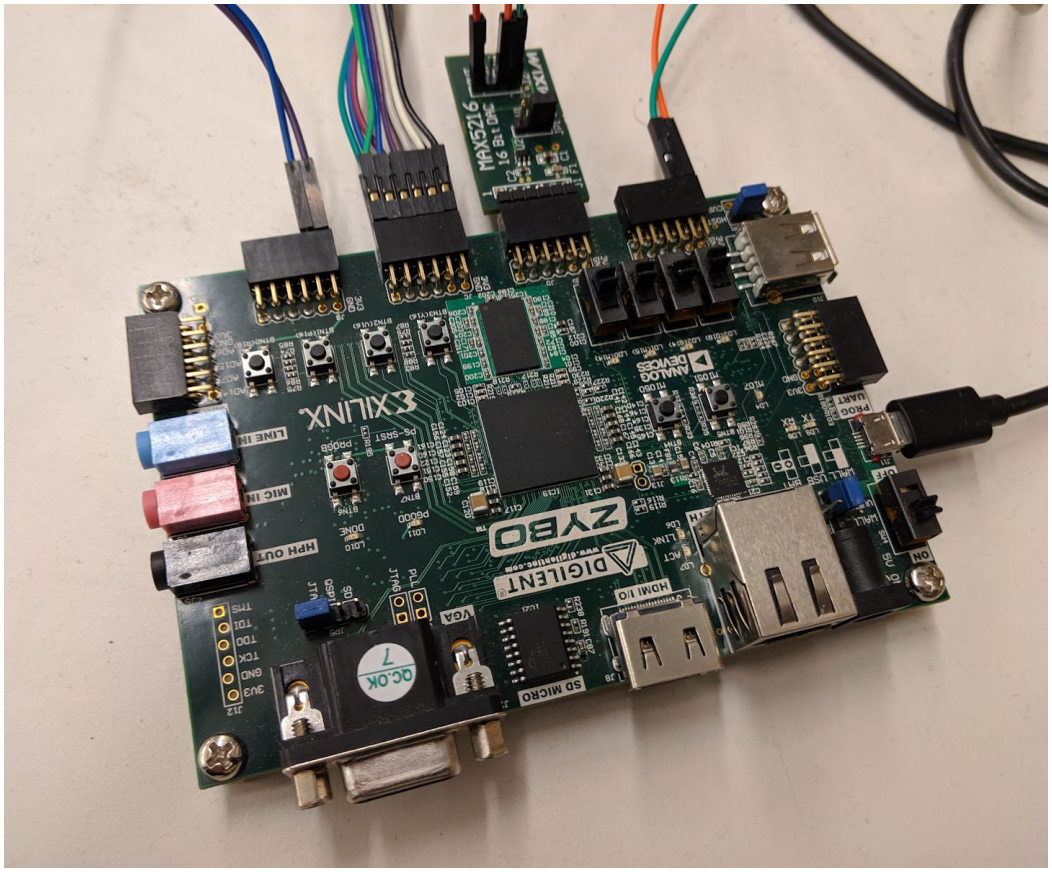


Figure 1.7.4: Zybo platform for drone development

1.7.5 Modular Control Algorithm Swapping on the Drone.

Another feature we plan on implementing is the modularity on control algorithms. Currently, the drone control program runs a PID based position correction algorithm. It measures the differences between the set points and the actual position of the drone that is captured.

1.7.6 More Interactive and Comprehensive UI

When talking with the current users of our project, we found that it would be helpful to create a more interactive element for the data from the quadcopter, camera system and ground station. We will do this by adding new CTEs, sliders and graphs to help visualize the input, output and calculated data.

2. Specifications and Analysis

2.1 Proposed Design

So far we've started looking at the documentation and getting familiar with previous years work. We've also talked with our client about what's expected of us and to help us plan out what we want to do with the quadcopter. We're planning to get the soldering done this week and start working on the crazyflies as a prototype for our work on the quadcopter. To approach solving these problems we are using the Agile

method to coordinate our coding work and the discord app to coordinate in-person meetings to work on the hardware aspects of the quadcopter. When working on our project we will follow IEEE safety standards.

2.2 Design Analysis

During the beginning design portion of the project we have worked to understand the current implemented system. Each component of the system has been built over many years and passed through many teams. It takes time to understand each portion of the system to an extent at which it can be utilized; therefore, the majority of the time has been spent reviewing documents.

Our ability to process the current documents has progressed well. We have begun to run the drone independently and some initial work. Each person on the team has demoed the drone and has read documentation. We have begun to complete initial work including soldering, control system analysis, gui interfaces, and initial design document creation.

Strengths

- Team comradery and communication
- Good mix of disciplines w/ large knowledge base
- Contacts with previous MicroCART users to answer questions
- Large group to distribute work

Weakness

- Large number of systems with intricacies
- No control theory knowledge base
- Poor documentation
- Time to complete is only a year

2.3 Development Process

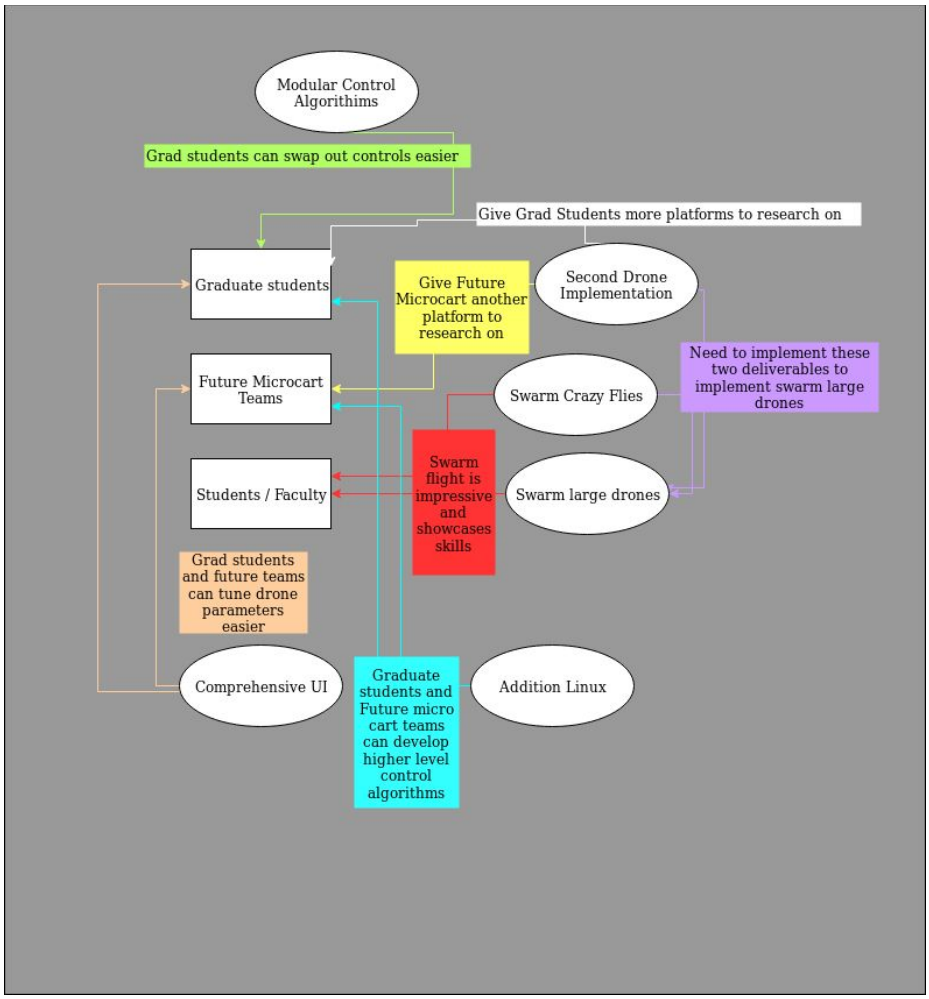
After being assigned our project, we met with our customer and advisor to discuss the requirements for our project. Since we're working on an embedded systems project, our system has multiple components that would have different requirements that would change as we move through the school year.

We are currently in the design phase of our project. We are researching materials like control theory etc. to help us understand our customer/users better and in turn, produce a better tailored product.

With these two phases, we are currently in the waterfall process. But since our requirements will change during the phase of the year, we are transitioning into an agile development process. The plan is as such:

- **Tickets:** We are currently using Trello to handle our tasks and keep track of the tasks that need to be done, the tasks that are in progress and our accomplished tasks.
- **Meeting with stakeholders:** We meet with our customers, users, and advisor once a week on Mondays to discuss our accomplished tasks and the concerns or goals they have for the coming sprint or sprints.
- **Sprints:** We have 1-week sprints in which we will go through one iteration of standup, grooming, retrospectives and stakeholder meetings.
- **Grooming:** After our stakeholder meeting, we regroup, and create and prioritize tickets on Trello for the new sprint.
- **Stand-up and Retrospective:** We start our weekly team meetings with scrum, where we talk about what we did, what we plan on doing and if we had any blockers for that sprint. Since we are full-time students, we compromised on doing this weekly instead of daily.

2.4 Design Plan



3. Statement of Work

3.1 Previous Work And Literature

The Microcart Quadcopter is an ongoing project at ISU that has been worked on by generations of students. Previous teams have already built a quadcopters drone and we will use their drone as starting point for how we want to build and modify our own version of the quadcopter. An advantage of having the previous years drone and code available is that it gives us a great idea of what we're trying to do and how to go about achieving it. A disadvantage is that we can't follow previous years example too closely because

some of their code doesn't work. Our team has already encountered several examples of broken code that previous teams have uploaded to the github.

3.2 Technology Considerations

Highlight the strengths, weakness, and trade-offs made in technology available.

Discuss possible solutions and design alternatives

Strengths

- High performance processing
    - Complex control integration
    - Visual data computation
    - General quickening in data processing
- New batteries with improved battery life and energy densities
- High performance motors
- PCB and CAD designing tools
- Large software libraries
- Open source material
- IMU sensors with high accuracy positional data
- Camera system with high accuracy positional data

Weaknesses

- Cost of new material and parts
- Weight added to quadcopter resulting in lower battery life and risk of loss of flight
- Increased dependency on a large volume of different technologies results in a convoluted mashing of technologies that is not digestible by future teams/students

Trade-offs (Summary)

- Cost and complexity for more advanced solutions with return on drone abilities
- Usage of new technologies tends to increase the abilities, but increases weight, power, and generally reduces performance of the drone

3.3 Task Decomposition

3.3.1 Drone Construction

Leveraging of current drone hardware buildup and extra parts to create a new, fully functioning drone. The drone buildup will require the use of knowledge in electrical hardware skills and some mechanical understanding. The information on how to build-up the new drone will come from observations of the previous drone.

3.3.2 Controller Swapping

Current drone software performs calculations to determine motor output using a PID scheme. The desired is to make the scheme swappable. Effort needs to be put forth into understanding how the current

implementation works and adding commands that are usable across all control schemes. An understanding of how to construct these control methods must also be done.

3.3.3 Image Data Processing

A second linux core is being added to the device to process image data from cameras not currently on the system. We then want the libraries on the linux core to support a C++ image processing library like OpenCV. We will have to find an attachment point for the camera and develop and electrical interface, which could be I2C or USB.

3.3.4 Multi-Drone Flight

The new drone built from the drone assembly process will be flown in tandem with the current working drone. The crazyflies in development are also going to be flown with the large drones. This will all require code for the drones to recognize each others position and ensure no collisions occur.

3.3.5 Organizing UI and Git

Much of the work that has been done over the past few years has been somewhat organized by the many hands the work has passed through. We will work to organize the hardware components, software code, git, and help-documents.

3.4 Possible Risks And Risk Management

3.4.1 Equipment Risks

There are several things that may impede our progress on this solution. The equipment that we were given to support software development on our platform is very dated. We were supplied one 15 year old computer that did not support c++ development, which is what our ground station was built with. This has already impeded development progress and may continue to slow our implementation time in the future. The other working development computer had several networking issues that were introduced by integration with the VPRN camera system. Some of the components that are supported by our previous teams are very old. The age of the sensors can introduce development problems.

3.4.2 Knowledge Area Risks

There are several development areas that could present risks because we have a lack of knowledge in these areas. The swarm flight requirements introduce some significant variation to the networking scheme of the current system. The current system has the ground station machine connect to a wifi network created by a small wifi microprocessor chip on our flight control board. This introduces a risk to our team because in the new configuration this configuration is not viable. We also have limited knowledge of methods for implementing a new networking scheme with the existing hardware. This will likely impede development progress at some point.

The implementing Linux on the second core feature has some knowledge area risks. We don't have knowledge of how trying to implement this feature will affect our latency in communications with the control station. We will have to thoroughly research and test to ensure this feature implements itself without compromising the safety-critical feedback systems.

3.4.3 Costs Risks

We have several dependent hardware components that we need to buy to support our drone applications. Buying these components impedes our progress because we have to go through a purchasing procedure and justify our need for these components. Some components may break or malfunction, so we will have to buy new ones. We have to buy custom circuit boards, RC controllers, Lithium Polymer batteries, etc. We have to wait for these things to be purchased in order to develop.

3.5 Project Proposed Milestones and Evaluation Criteria

Some key milestones for our project are

1. Having multiple autonomous Crazy Flies controlled with our GCS software
2. Build our own Quadcopter drone
3. Swarm flight with the Quadcopter
4. Implementing controls algorithm swapping
5. Implement Linux to second core
6. Improve UI and Github organization

For testing our first and second milestones we need to test how well they fly and then working on improving our controls system. The other milestones we will need to test using code.

3.6 Project Tracking Procedures

Our team has multiple ways of tracking our project's progress. We use Trello to communicate what needs to be done, their deadlines, and by whom. Our client also requires us to write weekly reports to track our one week sprint's progress and a meeting to discuss the sprint. We have a scrum-style stand-up every week to ensure that our team members are making significant progress, that no progress is being blocked and to plan for collaboration.

3.7 Expected Results and Validation

This project will be considered a success if the features we outlined earlier meet the needs of our clients. We have three client bases, future Microcart teams, Controls systems graduate students, and demonstration viewers. We need to support future Microcart teams by having detailed and organized documentation. We need to have maintainable software development practices. Meeting these requirements will give us a desired outcome for this client base.

We also need our application to support graduate student research development. To ensure this feature we need a comprehensive GUI tool that they can use to interface with the drone. The tool will need to interact with the quadcopter system to support real-time very precise indoor navigation. We will interact with our client frequently to ensure the tool implements his desired features. An example of one of these features is a slider bar to set PID gains on the control algorithm on the drone platform. One requirement that supports this outcome is the installation of Linux onto the second core. We will generate a successful outcome if this client finds our work satisfactory.

Finally, we will need to develop software and hardware features to showcase the value of an ECPE/SE degree to alumni, future students, anyone who may be interested, etc. Having features like swarm flight supports this outcome. These features showcase the value of skills gained because it is an incredibly difficult feature to implement.

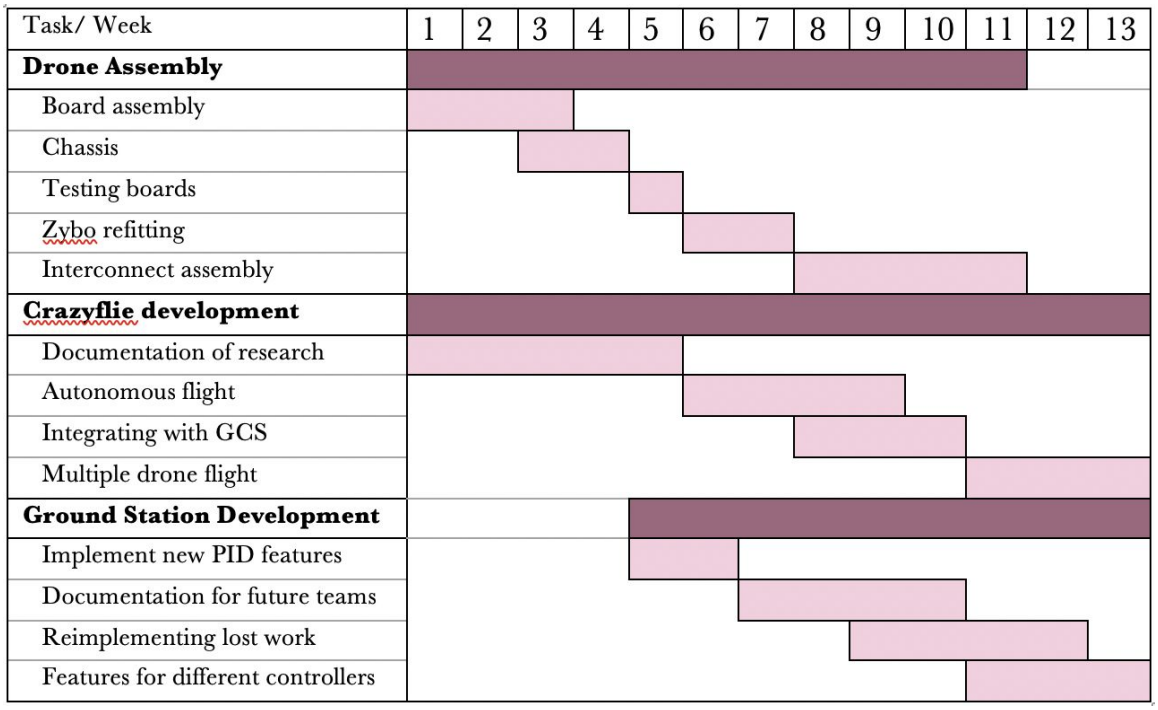
We will confirm that all the desired outcomes are met at a high level by continuously asking for client feedback during and after the development process. We will generate various test cases based on our understanding of the users needs, and we will make sure that these tests pass.

4. Project Timeline, Estimated Resources, and Challenges

4.1 Project Timeline

How would you plan for the project to be completed in two semesters? Represent with appropriate charts and tables or other means.

Make sure to include at least a couple paragraphs discussing the timeline and why it is being proposed. Include details that distinguish between design details for present project version and later stages of project.



4.2 Feasibility Assessment

Realistically we should be able to complete most of our set milestones. We've already achieved two of them, getting the Crazyflies flying and getting a second drone built. Now we're working on the more coding intensive work and we've already run into some issues with previous years code not working. We're currently debugging them and are struggling due to inexperience with the new programs. However we should still be able to complete most of our goals since we are making progress and the deadline is about half a year from now.

4.3 Personnel Effort Requirements

Include a detailed estimate in the form of a table accompanied by a textual reference and explanation. This estimate shall be done on a task-by-task basis and should be based on the projected effort required to perform the task correctly and not just "X" hours per week for the number of weeks that the task is active

Evan	11
Joe	9
Jacob	10
Kynara	9
Aaron	9
Shubham	10

4.4 Other Resource Requirements

We have identified the parts and materials required for this project in this document. In addition to that we have identified and ordered switch parts that we need for the Crazyfly and Drone. We also need access to the Coover 3050 room since it holds the quadcopters and the programs needed to work on the drone.

4.5 Financial Requirements

We shouldn't need any financial resources for our project. For anything we need to buy we need to get approval from Dr. Jones. If he agrees he'll order them for us.

5. Testing and Implementation

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or a software library

Although the tooling is usually significantly different, the testing process is typically quite similar regardless of CprE, EE, or SE themed project:

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study for functional and non-functional requirements)
2. Define the individual items to be tested
3. Define, design, and develop the actual test cases
4. Determine the anticipated test results for each test case 5. Perform the actual tests
6. Evaluate the actual test results
7. Make the necessary changes to the product being tested 8. Perform any necessary retesting
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you've determined.

5.1 Interface Specifications

– Discuss any hardware/software interfacing that you are working on for testing your project

5.2 Hardware and software

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

5.3 Functional Testing

Examples include unit, integration, system, acceptance testing

5.4 Non-Functional Testing

Testing for performance, security, usability, compatibility

5.5 Process

- Explain how each method indicated in Section 2 was tested
- Flow diagram of the process if applicable (should be for most projects)

5.6 Results

– List and explain any and all results obtained so far during the testing phase

- – Include failures and successes
- – Explain what you learned and how you are planning to change it as you progress with your project
- – If you are including figures, please include captions and cite it in the text

• This part will likely need to be refined in your 492 semester where the majority of the implementation and testing work will take place

-**Modeling and Simulation**: This could be logic analyzation, waveform outputs, block testing. 3D model renders, modeling graphs.

-List the **implementation Issues and Challenges**.

6. Closing Material

6.1 Conclusion

Summarize the work you have done so far. Briefly re-iterate your goals. Then, re-iterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

6.2 References

This will likely be different than in project plan, since these will be technical references versus related work / market survey references. Do professional citation style(ex. IEEE).

6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar that does not directly pertain to the problem but helps support it, include that here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc. PCB testing issues etc. Software bugs etc.