PORTLAND STATE UNIVERSITY

ECE 412

CAPSTONE PROJECT PROPOSAL 2020

RISC-V Betaflight Autopilot

Bliss Brass - brass@pdx.edu
Eric Schulte - eschulte@pdx.edu
Nikolay Nikolov - nnikolov@pdx.edu
Ruben Maldonado - mruben@pdx.edu

Advisor:
Roy Kravitz
Sponsor:
Galois

February 29, 2020

Contents

1 Executive Summary

Our objective is to port Betaflight, a mini open-source OS for drones to a board running on the RISC-V architecture. We are going to achieve our aim by turning SiFive's HiFive1 Revision B RISC-V development board into the flight controller for a standard 4-propeller racing drone.

The successful completion of the project will act as a proof-of-concept for the newer RISC-V architecture, as compared to older RISC machines like ARM.

Our project sponsor, Galois Inc., has asked us to deliver a flying racing drone with the HiFive1 Rev. B board running Betaflight by the end of the spring term in June 2020. Our sponsor, Galois, is purchasing and providing a drone kit for us to replace the native flight controller with the HiFive1 Rev. B.

2 Project Design Specification

2.1 Concept of operations

The primary purpose of this project is to demonstrate RISC-V's capabilities in a fun and exciting way. The primary users will be Galois and possibly other drone hobbyists looking to adapt future RISC-V boards for use on racing drones. It may also be used by others as a reference or basepoint to launch their own RISC-V based projects.

2.2 Stakeholders

The main project stakeholder is our project sponsor, Galois Inc. We will be providing them with a functioning drone running on RISC-V architecture. The other stakeholder (though they may not know it yet) is the Betaflight development team. We will share our RISC-V port via Github pull request at the end of the project.

2.3 Background

2.3.1 Betaflight

Betaflight is an open-source flight control software for drones and fixed-wing aircraft. Written in the C language, Betaflight currently supports over 100 flight control board models. Users compile and build model-specific

versions of Betaflight through a Google App or using a command line interface. Betaflight currently only supports flight control boards with STM32 RISC microcontrollers.

2.3.2 RISC-V Architecture

RISC-V is an open-source Instruction Set Architecture (ISA) developed at UC Berkeley and introduced in 2010. RISC-V is a modular ISA centered around the philosophy of 'use only what you need,' meaning the ISA aims to minimize wasteful use of resources. RISC-V includes a static core, RV32I, with the option of adding one or more standard extensions that hardware can consist of or not based on the needs of an application. The modular concept enables very small and low energy implementations of RISC-V, which can be critical for embedded applications.

In recent years RISC-V has even drawn the interest of companies such as Apple, Facebook, Google, and Samsung. Interested in the modular philosophy of the ISA, these companies have begun investing resources into creating their own RISC-V based silicon. The industry currently requires more practical applications to demonstrate the capabilities of RISC-V. We aim to create a racing drone to serve as one of these demonstration platforms.

2.4 Market Analysis

All drones, racing or otherwise, are controlled using a flight control board. The flight control board is a PCB that acts as the 'brains' of the drone; receiving the user input from an RC transmitter on the ground, processing the data, and outputting control instructions to the motors. All flight control boards must include at least two components: a processing unit, and an inertial measurement unit (IMU). Additional features may consist of components such as cameras, compasses, barometers, and telemetry devices.

The popularity of drones has exploded over the last ten years. As a result, the flight controller market has grown to include a diverse set of products from hundreds of companies. It's now possible to find a quality flight controller for between $25.00 - $50.00.

By porting Betaflight to the HiFive1 Rev. B development board, our team is essentially creating a custom flight control board. Our board won't be the first RISC-V flight controller to market, but, as far as we can tell, it will be the first RISC-V board to support Betaflight.

We'll be modeling our board's functionality around that of the Seriously Pro Racing F3 Flight Controller (SPRacingF3). The SPRacingF3 includes an accelerometer and gyroscope for determining orientation and speed, a compass for determining direction, and a barometer for determining altitude. We plan on completely implementing the accelerometer and gyroscope, while the compass and barometer remain stretch goals.

2.5 Requirements

Most broadly, the drone must fly with the HiFive1 Rev. B running a ported version of Betaflight. More specifically, the drone must respond accurately and responsively to user input controlling speed, direction, and altitude.

Since Betaflight is an open-source software project, our team must also document the porting process thoroughly enough so that someone else could recreate the project. We will share the final documentation with the Betaflight development team via a Github merge request at the end of the project. As such, a Betaflight user should be able to use the Betaflight tool to flash their own RISC-V based flight control board.

2.6 Specifications

Our team must use Betaflight as the flight control software, and use the HiFive1 Rev. B board as the flight control board. These are the only specifications provided by our project sponsor.

2.7 Deliverables

- Project Proposal
  - Product Design Specification
  - Project Management Plan
- Weekly progress reports
- Test plan
- Test results report
- Final report
- ECE Capstone poster session poster

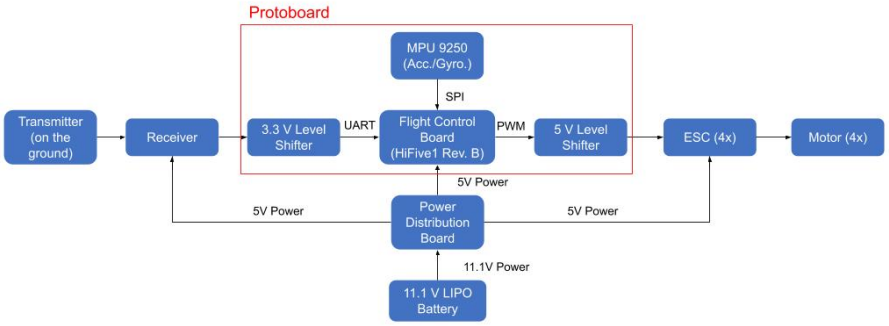
- Github merge request (to incorporate the RISC-V port into Betaflight's mainstream repository)

2.8 Initial Designs

2.8.1 Drone Kit Block Diagram

Below is the high-level block diagram for the assembled drone kit. The power distribution board (PDB) takes 11.1V power from the LIPO battery and converts it to 5V, powering the receiver, flight control board and the electronic speed controllers (ESCs). The receiver detects signals from the transmitter on the ground and relays data to the flight control board via a UART interface. The flight control board also receives input from the MPU 9250 accelerometer/gyroscope, via a SPI interface. The flight control board then outputs control data to the four ESCs in the form of PWM signals. Level shifters are required to step down input signal voltage from the receiver, and to step up output signal voltage to the ESCs. Lastly, the four propeller motors are connected to the four ESCs.

Figure 1: Drone kit block diagram.

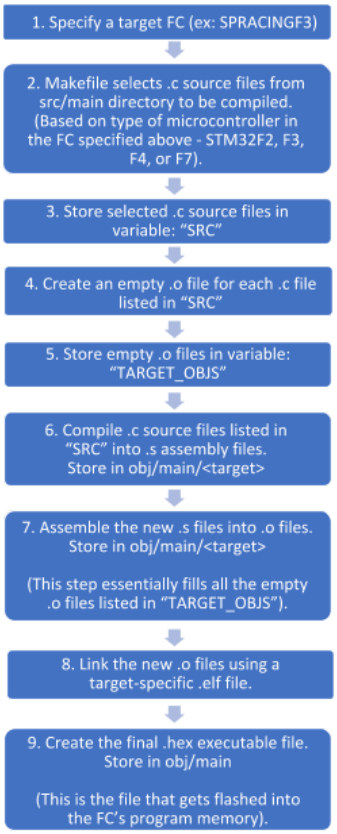


2.8.2 Betaflight's Makefile

Betaflight's Makefile provides instructions to the make utility on how to compile and build the final Betaflight executable file. This executable file then gets flashed to the target flight control board.

To create an executable for the RISC-V environment, our team must integrate RISC-V support into the Makefile. The first step in that process is exploring how the Makefile tells the compiler to build an executable. Below is a high-level block diagram of the Betaflight Makefile's functionality:

Figure 2: Makefile flowchart.



2.8.3 The Minimum Port

Before porting any specific Betaflight drivers, we want to complete a 'dry run' of the porting process. This will involve editing Betaflight's makefile to include RISC-V development tools and flags, and getting the makefile to produce an executable for the HiFive1 Rev. B. The idea is to learn how to build RISC-V executables using a version of Betaflight's makefile, prior to attempting to port any specific software. We refer to this initial dry run as the Minimum Port, and its steps are listed below:

Figure 3: Minimum port flowchart.

1. Start with the Betaflight makefile.
2. Download and include a path to the prebuilt RISC-V toolchain which includes GCC, G++, GDB, etc.
3. The toolchain comes with Freedom Studio, or it can be found on SiFive's website. The path can then be hardcoded in the makefile, or the tool.mk file can be updated to find the toolchain.
4. Add paths to .s and .S files. These files can be found after compiling a sample app with HiFive1 Rev. B listed as the target.
5. Add Cflags, Ldflags, etc. These files can be found after compiling a sample app with HiFive1 Rev. B listed as the target.
6. Add metal libs/includes files. These files can be found after compiling a sample app with HiFive1 Rev. B listed as the target.
7. Build a simple example program (i.e. flashing LEDs)
8. Exit makefile

2.9 Verification Plan

After completing the minimum port, we plan on splitting the porting work into modules.

The first module to be ported is the receiver firmware. Once that is complete, we will unit-test the firmware with the FS-iA6B Receiver to ensure standalone functionality. After the port of the receiver module, the team will split into two-person subteams, each focusing on one module.

One team will be responsible for porting the accelerometer firmware, and the other responsible for porting the ESC/motor output firmware. Similar to the receiver module testing, each team will unit-test their module to check for standalone functionality.

Once we confirm each module is working on its own, we will remove the stock flight control board from the drone and install the HiFive1 Rev. B as the flight controller. The final step will be performing integration testing to ensure all modules function correctly together as a single package.

These are just preliminary verification plans, with more specifics to come. But as of right now, this is the general outline of our verification plan.

2.10 Risks

Our team's biggest risk is our mutual lack of porting experience. We all have experience with software development, but no one has actually ported software before. This unfamiliarity may introduce confusion and uncertainty throughout the project. Therefore, it's important we develop an overall strategy before diving into any specific code (i.e. first creating a "minimum port" discussed above).

Another risk comes from the capabilities of the HiFive1 Rev. B itself. For example: are there enough GPIO pins and SPI ports to support Betaflight? Will the program memory of the HiFive1 Rev. B be large enough for the final executable? These are the types of issues that could arise later in the porting process.

Lastly, is the issue of compatibility between the stock drone kit and the HiFive1 Rev. B. Out of the box, the kit is designed to support an ARM-based flight controller. While the HiFive1 Rev. B should provide all the features of an ARM microcontroller, it's possible there may be a few compatibility issues.

3 Project Management Plan

3.1 Timeline

Figure 4: Project schedule for winter and spring terms.

PROJECT TITLE	RISC-V Cleanflight Autopilot	TEAM NAME	Capstone Team 8
PROJECT SPONSOR	Galois Inc.	DATE	2/21/2020

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	Research, PDS docs, Minimum Port					
1.1	Project Proposal (PDS and schedule)	All	1/27/20	2/7/20	10	100%
1.2	Startup file research	Bliss	1/27/20	1/31/20	4	100%
1.3	SPI Readme	Ruben	1/27/20	1/31/20	4	100%
1.4	SDK Readme	Ruben	1/27/20	1/31/20	4	100%
1.5	Minimum Port (see PDS for details)	All	2/3/20	2/22/20	18	100%
2a	Receiver Port					
2.1a	ID relevant Cleanflight source files	All	2/24/20	2/28/20	4	20%
2.2a	Incorp. Freedom Metal drivers, interface code	All	3/2/20	3/13/20	11	0%
2.3a	Edit Makefile and flash to HiFive board	All	3/16/20	3/20/20	4	0%
T-1	Test Phase 1					
T-1.1	Receiver Module unit testing	All	3/23/20	3/27/20	4	0%
2b	Initial Drone Assembly (using SPRacingF3)					
2.1b	Body, electrical H/W assembly	Niko, Bliss	2/24/20	2/28/20	4	0%
2.2b	Build Cleanflight, Flash, Fly	Niko, Bliss	2/24/20	2/28/20	4	0%

WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
4	ESCs/Motor Output Port					
4.1	ID relevant Cleanflight source files	Team 1 (TBD)	3/30/20	4/3/20	3	0%
4.2	Incorp. Freedom Metal drivers, interface code	Team 1 (TBD)	4/6/20	4/17/20	11	0%
4.3	Edit Makefile and flash to HiFive board	Team 1 (TBD)	4/20/20	4/24/20	4	0%
T-2	Test Phase 2					
T-2.1	ESC Output Module unit testing	Team 1 (TBD)	4/27/20	5/1/20	4	0%
5	Accelerometer/Gyroscope Port					
5.1	ID relevant Cleanflight source files	Team 2 (TBD)	3/30/20	4/3/20	3	0%
5.2	Incorp. Freedom Metal drivers, interface code	Team 2 (TBD)	4/6/20	4/17/20	11	0%
5.3	Edit Makefile and flash to HiFive board	Team 2 (TBD)	4/20/20	4/24/20	4	0%
T-2	Test Phase 2					
T-2.1	Accelerometer/Gyroscope Module unit testing	Team 2 (TBD)	4/27/20	5/1/20	4	0%
T-3	Test Phase 3					
T-3.1	Integration testing: Receiver/ESC modules	All	5/4/20	5/8/20	4	0%
T-3.2	Integration testing: Receiver+ESC/IMU modules	All	5/11/20	5/22/20	11	0%
7	Final Documentation					
7.1	Presentation Poster	All	5/25/20	6/5/20	10	0%
7.2	Final Report	All	6/1/20	6/5/20	4	0%

3.2 Milestones

1. Complete research phase
2. Complete Minimum Port (see PDS, Initial Designs section for more details)
3. Project Proposal approved by sponsor
   (a) PDS
   (b) Project Management Plan
4. Assemble drone kit and install stock flight control board
   (a) Flash Betaflight to stock board
   (b) Test flight capabilities
5. Receiver module tested and ported to HiFive1 Rev. B
6. Accelerometer/Gyroscope module tested and ported to HiFive1 Rev. B
7. ESC/motor output module tested and ported to HiFive1 Rev. B
8. Install HiFive1 Rev. B as flight control board
9. Complete final integration testing
10. Create final documentation:
    (a) RISC-V Github pull request
    (b) Final report
    (c) Presentation poster

3.3 Project Budget and Resources

3.3.1 BOM

Item	Description	Qty.	Price	Total
LHI 280 Race Quad ARF	Drone kit (including SPRacingF3 FC)	1	$105.99	$105.99
Flysky FS-i6X 10CH 2.4GHz AFHDS RC Transmitter w/ FS-iA6B Receiver	RC transmitter and receiver	1	$52.99	$52.99
Ovonic 11.1V 2200mAh Lipo Battery	Battery	1	$17.99	$17.99
Sparkfun MPU-9250 Breakout board	Accelerometer/gyroscope	1	$14.95	$14.95
HiFive1 Rev. B Development Board	RISC-V development board	2	$59.00	$118 .00
Adafruit Proto-screwshield	RISC-V Protoboard	1	$14.95	$14.95
Total				$324.87

Our total project budget comes out to roughly $325.00. This budget includes:

1. (1) Racing drone kit
2. (1) RC transmitter and receiver
3. (1) LIPO battery
4. (1) MPU-9250 accelerometer breakout board
5. (2) HiFive1 Rev. B Development Boards
6. (1) Protoboard

We want to first assemble the stock drone kit (including the stock flight control board), and flash a standard build of Betaflight to get a better idea of the build process for a specific target.

This will hopefully give us a better understanding of how to do the same process with the HiFive1 Rev. B, the only difference being the ported software.

We will use the Capstone Lab and the EPL during the hardware assembly and testing phases.

3.4 Team members and relevant skills

Bliss Brass, CE :

- C,C++ scripting,LaTeX,Markdown
- Firmware and Linux device drivers
- Microprocessor Architecture

Ruben Maldonado, CE :

- C,C++,Python, Shell scripting
- Firmware and Linux device drivers
- Microprocessor Architecture

Nikolay Nikolov, CE :

- C,C++,Python, Shell scripting,LaTeX,Markdown
- Firmware and Linux device drivers
- Microprocessor Architecture

Eric Schulte, EE :

- Microprocessor system design (ECE 371)
- Hardware prototyping and testing
- Some object-oriented coding (C/C++)

3.5 Development Process

Note: These plans are preliminary and subject to change, but this is the overall idea of how we expect the project to proceed.

Ruben has volunteered to act as liaison between the team and our industry sponsor and faculty advisor. We are using Slack for daily communication, and Github's Project feature for progress/task tracking.

The plan is to have everybody working on tasks in series for most of the project. Since we're all new to software porting we think this is the best strategy to minimize setbacks. The first thing we're going to 'build' is what we call the 'minimum port'.

The minimum port entails creating a basic makefile, based on Betaflight's makefile, to build an executable of a simple demo program (i.e. flash some LEDs), and then flash it to the HiFive1 Rev. B using SiFive's Freedom Studio. This simple makefile will serve as our team's entry-point into the Betaflight code. Once we are able to compile and build a simple demo program we can add more complexity and start building advanced executables like ported hardware drivers.

With our custom makefile complete, our team will begin porting the necessary firmware for the FS-iA6B Receiver. This device relays signals from the transmitter on the ground to the drone's flight control board. Since the receiver firmware will be our team's first port, we've made this phase the longest of the project. We anticipate this phase will be quite challenging; as such, our team plans to work in series.

Once the receiver port is complete, we'll begin focusing on porting firmware for the accelerometer and the electric speed controllers (ESC). ESCs act as an intermediary device between the flight control board and motors controlling the propellers. Hopefully by this point the porting process will be better understood, which is why we plan on splitting into teams of two to maximize efficiency. One team will work on porting accelerometer firmware, while the other ports firmware for the ESCs.

We plan to incorporate three phases of testing throughout the project: One phase of unit-testing for the receiver module, another for the accelerometer and ESC modules, and a third phase for final integration testing. As part of the integration testing, we will swap the kit's stock flight control board (SPRacingF3) with the HiFive1 Rev. B running the ported Betaflight firmware.

We plan to integrate modules cumulatively, meaning that as soon as we get one up and running we will immediately try to integrate it with the others. We hope this iterative process will result in less debugging effort

needed during the final month of the project.

After final integration testing is complete, we will focus on compiling our project's final documentation. This will include: assembling a Github merge request to submit our RISC-V port to the mainstream of Betaflight, writing the final report, and creating the presentation poster.

Note: We should be completing necessary documentation (i.e., wikis, manuals) for the Github merge request along the way, not all at once at the end.

Signature:

Date: