Bounty Engineering Pod

PT3 Ready Flight Companion PC (FCPC)

Jul 6, 2024

Background

The FCPC is an microprocessor enabled computation unit which is an addition to a bare bones Flight control stack, and is primarily responsible for handling user experience and logging and transceiving systems data. The intent is to remove excess computational load from the low-level microcontroller based feedback flight controller (FC), as well as prevent lock-in to any single off the shelf flight control stack. This is achieved by relegating the FC to only essential low-level functionality, while creating a stepping off point toward developing a fully functional and open flight stack in the future.

Objectives

Develop a system that can parse and log avionics network data from all relevant system modules onboard the craft, act as an on the ground receiver of transmitted data.

Develop a system to host an Inflight UI showing key flight data for both Pilots on the craft and technicians at the ground control station.

Develop a system that can transmit that information reliably over long distances and update the ground control station.

Requirements

The system should be able to:

Parse and Interpret Cyphal CAN Data Packets going to and from 6 AMPX 300A(12-24S) HV Drone ESC. Along the form described here .

Parse and Interpret VESC CAN Data Packets from 6 ENNOID-BMS XLITE-V4. Along the form described here for the ID structure, here for the data body structure, and here for the transmission structure .

Parse and Interpret Cyphal CAN Data Packets from 2 HJLG3-M - JHL Joysticks. Along the form described here .

Parse and Interpret UART Data Packets from an Embention Veronte X1, along the following form :

datatag	data_description
altitude_AGL	Altitude Above Ground Level
altitude_AGL_set	Altitude Above Ground Level Demanded
altitude_ABS	Pressure Altitude
heading	Heading
compass	Compass
attitude_pitch	Pitch Attitude
attitude_roll	Roll Attitude
vertical_speed_KTS	Aircraft Vertical Speed
airspeed_KTS	Airspeed
OAT	Ambient Temperature
latitude	GPS Latitude
longitude	GPS Longitude

{

"altitude_AGL": 3294.45, "altitude_AGL_set": 9480.29, "altitude_ABS": 623.55, "heading": 175.62, "compass": 45.98, "attitude_pitch": 16.25, "attitude_roll": 103.14, "vertical_speed_KTS": 22.31, "airspeed_KTS": 340.75, "OAT": 12.56, "latitude": ‘45.789456 N’, "longitude": ‘-73.987654 W’

}

Log all the above information in a reasonable and easily interpretable file format, which should be distinguishable by timestamp .

The system should have:

A User interface consisting of 2 system windows displayed across 2 10.1 inch monitors and with elements visible an arms length away from the pilot. These windows should have the following capabilities :

Display 1: System State Window (SSW)

The SSW displays dynamic state information, such as aircraft control inputs states, altitude, attitude and position. It should consist of graphical dials, gauges and windows, any warnings or errors and system events flags, as well as text, to enable intuitive and at a glance understanding of the craft's motion .

Display 2: System Information Window (SIW)

The SIW should display more static flight information, chiefly flight time, and the battery and motor data .

This UI should be replicable on the ground control station so data can be viewed by ground operators as they test the overall system .

The information on the type, number and structure of the to be displayed and transmitted can be found here:

Cockpit Display Parameters

The system should be reasonably electrically encapsulated within an easily deployable format, from both a software, where installation is quick and simple, and hardware, where it is compatible with the layout and requirements of the other system modules as shown here, perspective.

Process and Timeline

Weekly Progress Update Meetings

Attend a 15 - 30 minute weekly meetings to update the progress and estimated remaining time to completion. Day & Time to be agreed upon

Track progress via dework task board, generating and detailing system elements and stages, as well as producing a record of work document detailing significant development events and junctures, challenges and decisions.

Testing and Integration

Be available for hardware testing and integration assistance as required

Bounty Period

Taking into account work already completed:

~ 60 Hrs of work to completion, including documentation and integration

~ 3 - 4 weeks @ 50% time commitment (allowing for interruptions for other supplemental tasks)

Submission Guidelines

Bill of Material (BOM)

Produce a BOM document for all hardware and software components used during development and production

Program Repository

All software elements and hardware design files should be returned in a format encapsulate-able within a single Github repository

Documentation

Produce section by section documentation detailing the:

- Functionality and Structure
- Rationale and intent behind each element of the design
- Testing and integration plan/manual.

This should be in a form easily transposable to a markdown layout

Future Work Recommendations

Within the documentation, include a section detailing any design limitations and potential resolutions and extensibility

Evaluation Criteria

System section quantity and degree of completion and robustness

Documentation quality as judged upon review by:

- Readability & clarity
- Detail
- Reproducibility

Reward

~ 60 hrs

@$30/hr @25% Arrow (pre market $0.3/Arrow)

Total = $1350 USDC + $1500 Arrow

** Accounting began towards the end of the project

Contact Information

For any inquiries or further details, please reach out to A. Alperen Gündoğan Alexander Dada Thomas Garrison or @engineeringpod in the Arrow Air Discord Server.