Oregon State University

ECE 406: Project

Project Proposal

Travis Fredrickson

fredritr@oregonstate.edu

Date: 2024-11-13

# Table of Contents

[Table of Contents 2](#_Toc182650539)

[Ideas 3](#_Toc182650540)

[General Ideas 3](#_Toc182650541)

[Machine and Environment Monitor and Control System 3](#_Toc182650542)

[Pen Plotter 3](#_Toc182650543)

[Microcontroller Options 4](#_Toc182650544)

[ESP32-C6 4](#_Toc182650545)

[Links 4](#_Toc182650546)

[3D Systems Senior Embedded Engineer Job Description Skills 5](#_Toc182650547)

[Tier 1 Skills 5](#_Toc182650548)

[Tier 2 Skills 5](#_Toc182650549)

[Tier 3 Skills 5](#_Toc182650550)

[Don Advising Appointment 6](#_Toc182650551)

[Form to Submit 6](#_Toc182650552)

[Title of Project 6](#_Toc182650553)

[Four or More Sentence Overview 6](#_Toc182650554)

[At Least Four Engineering Requirements 7](#_Toc182650555)

[Engineering Requirement Evaluation 7](#_Toc182650556)

[Notes 8](#_Toc182650557)

[RTOS 8](#_Toc182650558)

[Rust 8](#_Toc182650559)

# Ideas

## General Ideas

* Avoid uncommon, non-reuseable, and large parts.
  + Use an STM32 or ESP32 development board.
  + Use the DE10-Lite FPGA board.
* Target a company’s job description. For example, the company 3D Systems.

## Machine and Environment Monitor and Control System

* Imagine how an industrial manufacturing environment would have many machines and require a monitoring and control system.
* There could be two main parts: a GUI and a microcontroller unit (MCU). They would communicate with each other. Could add an SQL database that stores monitoring logs from the MCU and stores “tests” or “commands” that can be sent to the MCU.
* The MCU could be a “machine” in an industrial manufacturing environment, with sensors and such.
* The GUI would display information received from the MCU. It could also transmit commands to the MCU. It could be scaled to include many MCUs, aka “machines”. Perhaps two or more MCUs should be purchased for this.
* This project could extend another project with an actual machine, such as a pen plotter.

## Pen Plotter

* Could create by either:
  + Modifying a 3D printer.
  + Building from scratch, such as using aluminum extrusions.
* Could use image analyzation or generation algorithms to create input.
* Could create custom firmware to process inputs into pen plotter motor instructions.

# Microcontroller Options

## ESP32-C6

After some research, the “ESP32-C6-DevKitC-1 v1.2” seems to be the ideal microcontroller and development board. It has many peripherals (e.g., Bluetooth, Wi-Fi) and fits on a standard 5-hole column breadboard with 1 open pin on either side. Some other common development boards do not fit on a standard breadboard.

However, it seems to only have one core, which would prevent multithreading.

ESP32-C6-DevKitC-1 v1.2 description: “ESP32-C6 general-purpose development board, based on ESP32-C6-WROOM-1 or ESP32-C6-WROOM-1U. It has all the ESP32-C6 pins exposed and is easy to connect and use. Most of the I/O pins are broken out to the pin headers on both sides for easy interfacing. Developers can either connect peripherals with jumper wires or mount ESP32-C6-DevKitC-1 on a breadboard.”

Note: ESP32-C6-WROOM-1 vs ESP32-C6-WROOM-1U: U has no internal antenna, but an external connector.

### Links

<https://en.wikipedia.org/wiki/ESP32>

<https://www.espressif.com/en/products/devkits>

<https://docs.espressif.com/projects/esp-dev-kits/en/latest/esp32c6/esp32-c6-devkitc-1/user_guide.html#hardware-reference>

<https://www.youtube.com/watch?v=u5unB24lhC4>

<https://www.youtube.com/watch?v=zqwZGWxJs-0>

# 3D Systems Senior Embedded Engineer Job Description Skills

This is a collection of skills found in the job description for a Senior Embedded Engineer at the company 3D Systems in Wilsonville, Oregon. The skills are organized in tiers, with lower number tiers being more important to target.

## Tier 1 Skills

* Mastery of:
  + Embedded C/C++.
* Proficiency of:
  + **Device drivers**.
  + Embedded systems architecture/design principles from concept to production.
  + Hardware/software protocols like PCIe, **SPI, I2C, and RS-232**.
  + **Real-time, preemptive**, or multi-threaded software.
* Familiarity of:
  + Python, **Rust**, OOP, **GUI**, Linux, Windows, and **networking protocols**.

## Tier 2 Skills

* Debugging skills.
* Full-stack development.
* Lab instrumentation (oscilloscopes, logic analyzers, etc.).
* Linux and Windows environments.
* Low-cost design/reuse approaches.
* PC-hosted and cloud-based applications.
* Web technologies (bonus).

## Tier 3 Skills

* Communication/documentation skills.
* Jira and Confluence.
* Modern development workflows (configuration management, defect tracking).
* Print processes and imaging pipelines (bonus).
* Technical specifications/documentation.

# Don Advising Appointment

Don advised that Modbus and RS-45 may be common in industrial manufacturing environments. Big machines probably are not moved often, thus routing wires is a worthy investment. However, using wireless network communication protocols is still a valid option that demonstrates skills.

Do advised that voltages for wired communication could be an issue. Development board may only be able to supply certain voltages. This could require external circuitry.

Do advised that for wireless networks, network reliability, number of nodes, and signal distance are factors to consider demonstrating. Some questions to answer are: What if a node drops out, how is it handled? How are bad acters handled?

# Form to Submit

Form link:

<https://docs.google.com/forms/d/e/1FAIpQLSf3ZudK3yJMP9BU3osduIdm15W9gow4SFm0ibh8oIbQM-wAPg/viewform>

## Title of Project

Machine Network Monitor and Control System.

## Four or More Sentence Overview

Imagine how an industrial manufacturing environment would have many machines and require a monitor and control system. There would be two main parts: a graphical user interface (GUI) and microcontroller units (MCUs); the MCUs would be “machines”. The GUI would display information received from the MCUs and/or would transmit commands to the MCUs. It could be designed to be scalable to include many MCUs that communicate over a network. The inspiration for this project and its engineering requirements are based on Senior Embedded Engineer job description’s skills and qualifications at the company 3D Systems in Wilsonville, Oregon.

## At Least Four Engineering Requirements

Reminder: Engineering requirements (ERs) should be:

* **Traceable:** It is clear why the ER exists.
* **Abstract:** Refers to how an ER should describe what a system can do, not how it does it.
* **Unambiguous:** Can be shown to be completed without any stakeholders being able to disagree.
* **Verifiable:** Can be reasonably shown to be completed.

ERs:

1. The system shall have a custom GUI that communicates with at least one MCU.
2. The system shall have wired (e.g., SPI, I2C, or RS-232) or wireless (e.g., Wi-Fi, Bluetooth, Zigbee, or Thread) communication between multiple MCUs.
3. The system shall have real-time, preemptive, or multi-threaded software.
4. The system shall have some Rust code of arbitrary functionality (e.g., as a custom RS-232 driver).

## Engineering Requirement Evaluation

1. There will be visually observable effects resulting from the communication between the GUI and MCU. E.g., a button press on the GUI lights up an LED connected to an MCU GPIO pin, or a button press connected to an MCU GPIO pin sends a debug message to the GUI.
2. There will be visually observable effects resulting from the communication between the MCUs. E.g., a button press connected to an MCU will enable a task that transmits to other MCUs that then light up an LED connected to their GPIO pins. If using a router type network, it could be demonstrated that the transmitted message is first sent from one end node MCU to a router MCU, then from the router MCU to another end node MCU.
3. There will be multiple software tasks of different priority levels to demonstrate preemptive task scheduling. E.g., there could be multiple buttons and LEDs connected to different GPIO pins on an MCU that represent different tasks. Based on the button pressed, a corresponding task, represented visually by an LED, would be enabled. Depending on the priority levels, different button press combinations should demonstrate that the LEDs either change or remain constant. It will be demonstrated that this is a result of different priority levels, not some cheap if/else statements.
4. There will be code that is demonstrated to be of Rust syntax and its arbitrary functionality will be demonstrated.

# Notes

## RTOS

“Introduction to RTOS Part 1 - What is a Real-Time Operating System (RTOS)? | Digi-Key Electronics”

<https://www.youtube.com/watch?v=F321087yYy4>

Very good guide for FreeRTOS on an ESP32 and related topics.

“Reasons for Using an RTOS, Real Time Operating System, with an MCU”

<https://www.youtube.com/watch?v=bOFMXuoc5kE>

“A realtime developer's checklist”

<https://lwn.net/Articles/837019/>

“Real Time Operating System (RTOS) Multitasking Scheduling”

<https://www.youtube.com/watch?v=6L5DjHHHi-U>

Okay example of real-time preemptive vs. non-preemptive software.

“#22 RTOS Part-1: What is a Real-Time Operating System?”

<https://www.youtube.com/watch?v=TEq3-p0GWGI>

## Rust

<https://www.google.com/search?client=firefox-b-1-d&q=mixing+c+and+rust>

“A little Rust with your C”

<https://docs.rust-embedded.org/book/interoperability/rust-with-c.html>