UMassAmherst
The Commonwealth's Flagship Campus

SDP '21 - Comprehensive Design Review - Active Windows

Team 6

Jonathan Clifford, Frank Cremonini, Griffin Manns, Connor Moore
Advisor: Professor Arman Pouraghily
Manhattan2 CTO & Contact: Mr. Glenn Weinreb

Overview

- Problem Statement
- Introduction
- System Specifications
- Our System & Block Diagrams
- CDR Deliverables
- Current Prototype & Demonstration
- Project Management
- FPR Plan

Problem Statement

“With climate change becoming ever more relevant, engineers from every discipline are scrambling to combat the issue. Through methods such as green energy technologies and reducing reliance on fossil fuels, our society has made significant progress. However, there is still much work to be done. The average home presents opportunities for countless improvements in energy conservation. In particular, heating and cooling of homes present the opportunity for many inefficiencies to be addressed, including the lack of automation of residential climate control. For instance, to account for temperature variances, an individual may choose to run power-hungry central air systems as opposed to using manually adjusted power-saving alternatives. However, a sufficiently smart home automation system could replace human negligence and laziness and utilize alternative energy saving options that take advantage of the existing features of homes.”

Project Overview

- Window Operation
- We are focused on contributing to window automation in order to try and combat climate change.
- To do so, our team is focused on developing the end to end communications that this proposed system will utilize to communicate internally and externally.
- This is done via CAN internally and Amazon Web Services (AWS) externally, both to forward system status as well as User commands to control and monitor the window.

https://www.pellabranch.com/webres/Image/misc/2018-window-glass-option-header.jpg

Our Solution

Windows are “Dumb”, we will attempt to make them more “Active” by allowing automated remote operation of the window.

To achieve this we have the following components:

- **Android App** for the user to interface with the system and operate the window.
- **Cloud Connection** → Monitor and Control from anywhere.
- If Cloud is down, locally we have:
  - A manual override switch.
  - Direct connection between the Android App & the system.
- **Network Master Controller** to control the local network and interface with Cloud services
- **Motor Module** to operate window.

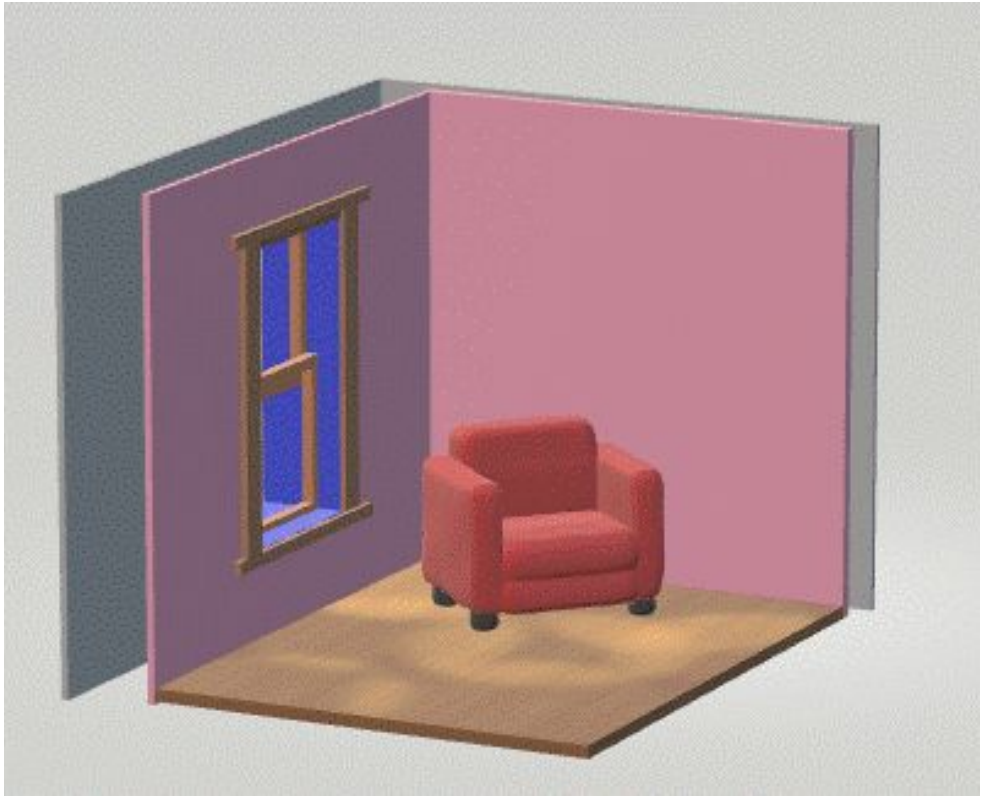
https://backtest-rookies.com/2018/10/26/tradingview-opening-a-window/

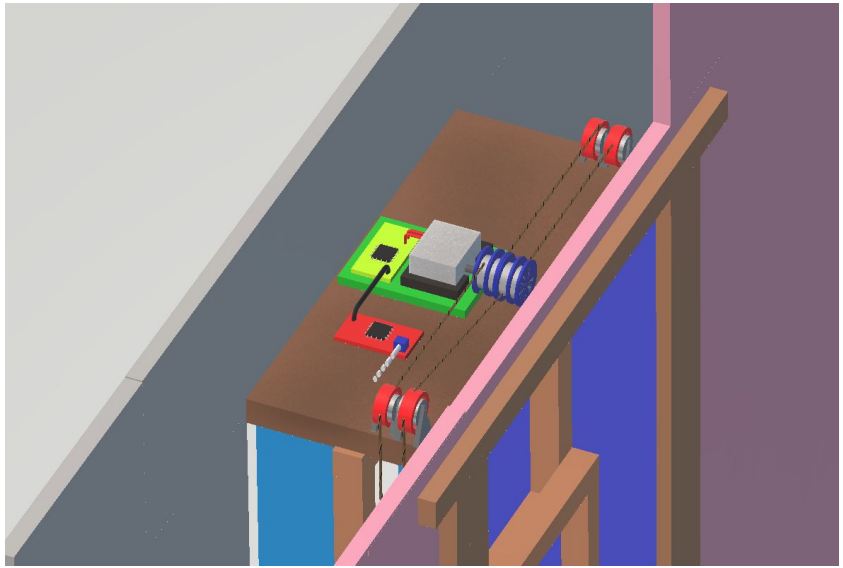
System Specifications

The Active Window Subsystems consisting of the Network Master Controller, IoT/Phone Application, and the Motor Module will meet or exceed the following specifications:

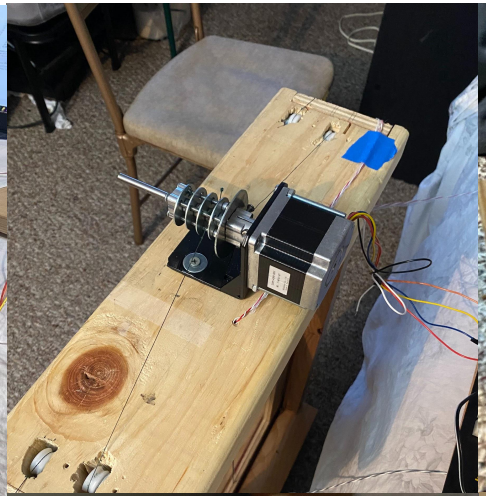
1) Capable of receiving signals from Amazon Web Services (AWS) for direct communication between the user and the rest of the system.
2) Capable of transmitting commands from the Android Application through AWS to the Network Master Controller that drives a stepper motor driver, which in turn adjusts a window.
3) User interaction via an Android Phone Application that transmits data via AWS to the controllers. This Application will include a user interface to control and monitor window status.
4) The Motor Module**'s** size is at least 14cm x 8cm in order to maximize installation in a home.
5) DC Stepper Motor and Driver, will utilize a 24V power supply and use a pulley system solution that can adjust the window via commands. This pulley system will physically lower and raise the test window once it receives the proper command.
6) In addition, there will also be a manual override switch for the test window that is capable of bypassing the Android Application and will raise or lower the window when it is flipped.
7) In the event of external network failure, the Android Application is capable of sending commands to the Network Master Controller if they are on the same network.

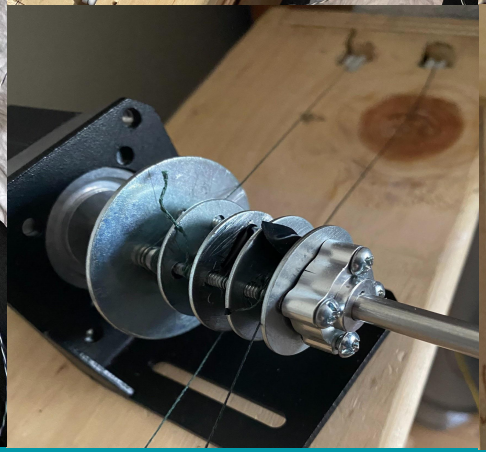
Our Solution in The Field





Model Window:

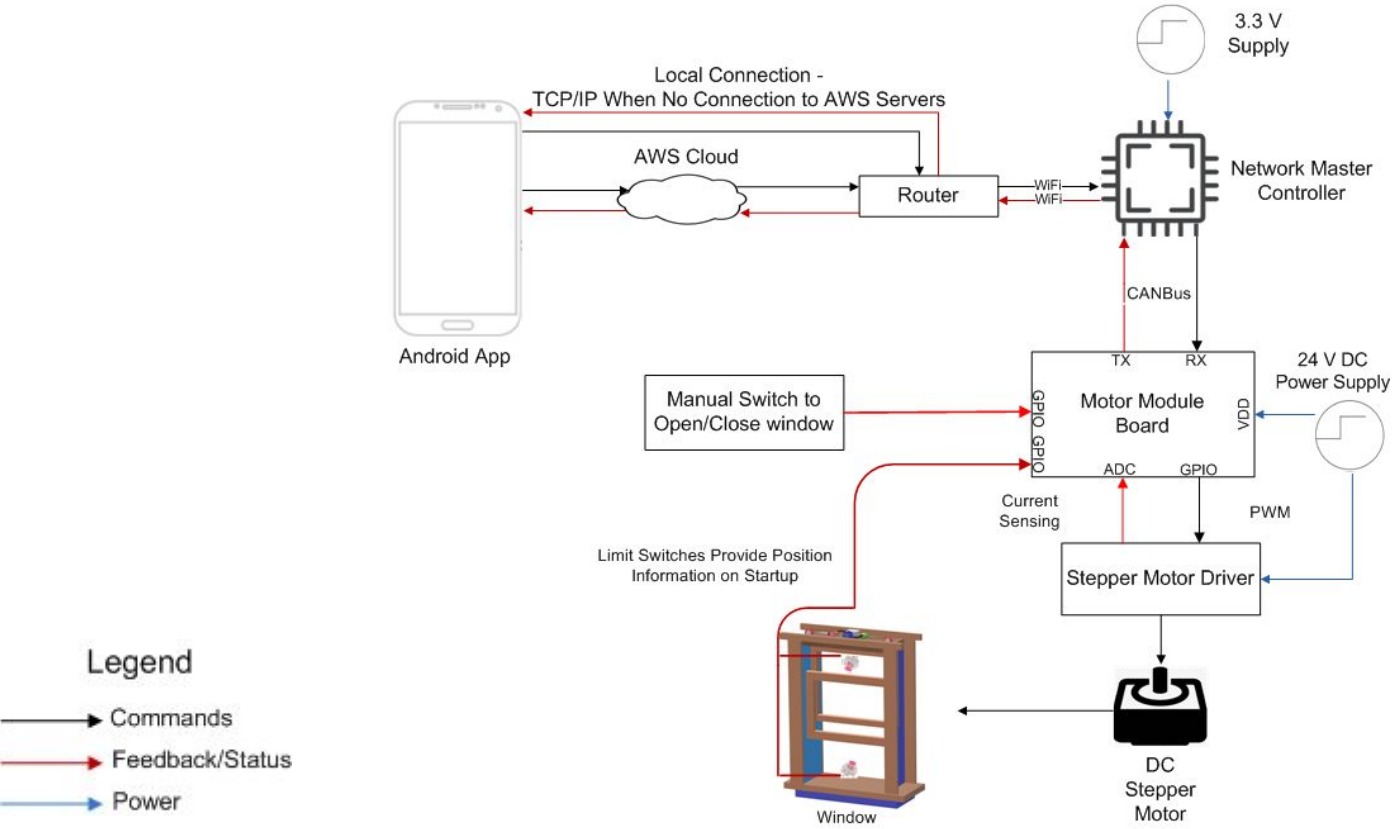




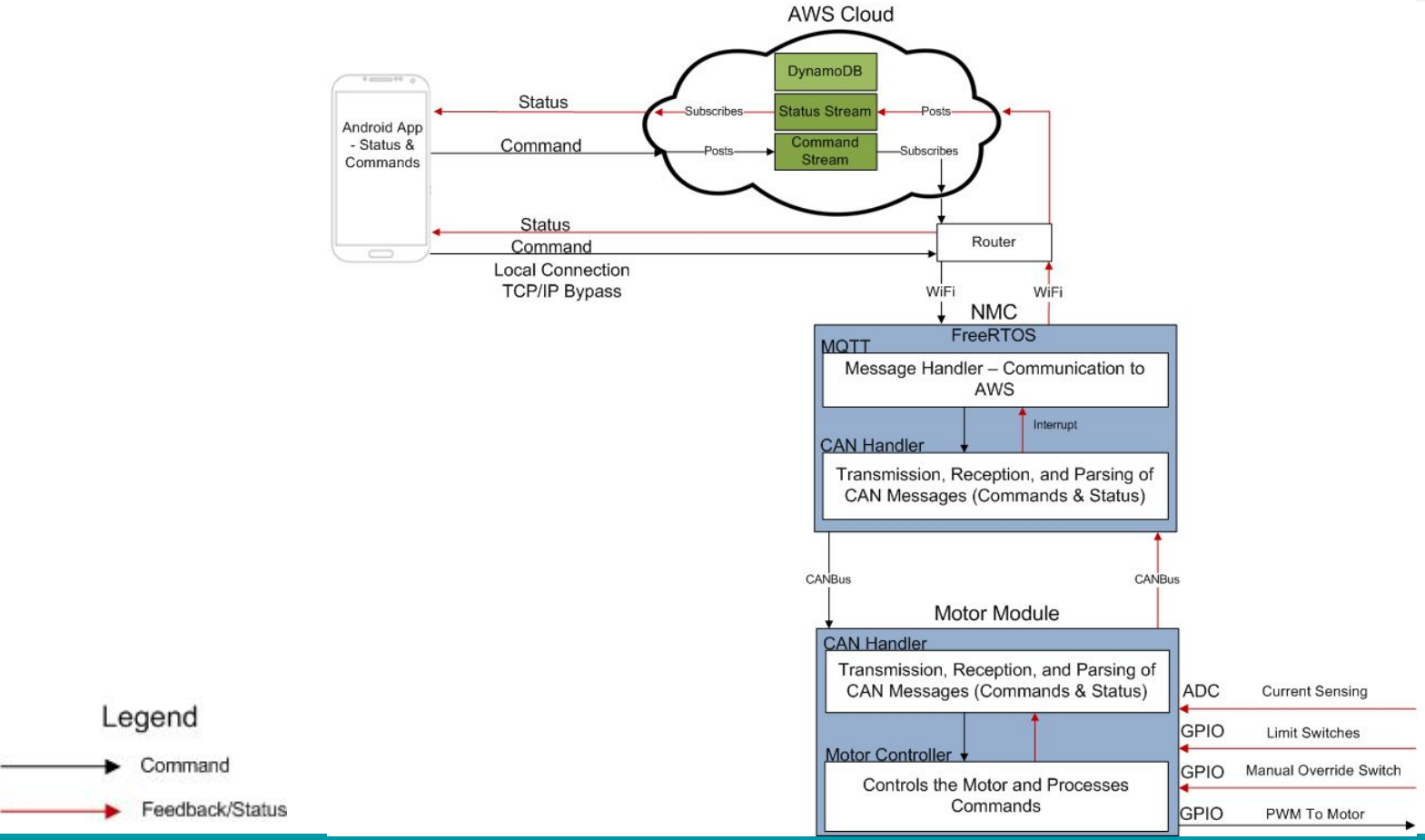
Updated Hardware Block Diagram

UMassAmherst

Team 6 – Updated Block Diagram of Active Window System



Updated Software Block Diagram



CDR Deliverables

- Full end to end communication.
  - Send a command via the Android phone application ✓
  - AWS receives this command and forwards it to the NMC ✓
  - NMC receives the command, sent to Motor Module using CANBus ✓
  - Motor Module acutates based on command, sends status to NMC ✓
  - NMC Forwards to AWS, displayed in Android Application ✓
- Local Connection
  - NMC and Android Application communicate over local (same) network —
- Manual Override Switch
  - Local switch to adjust window ✓

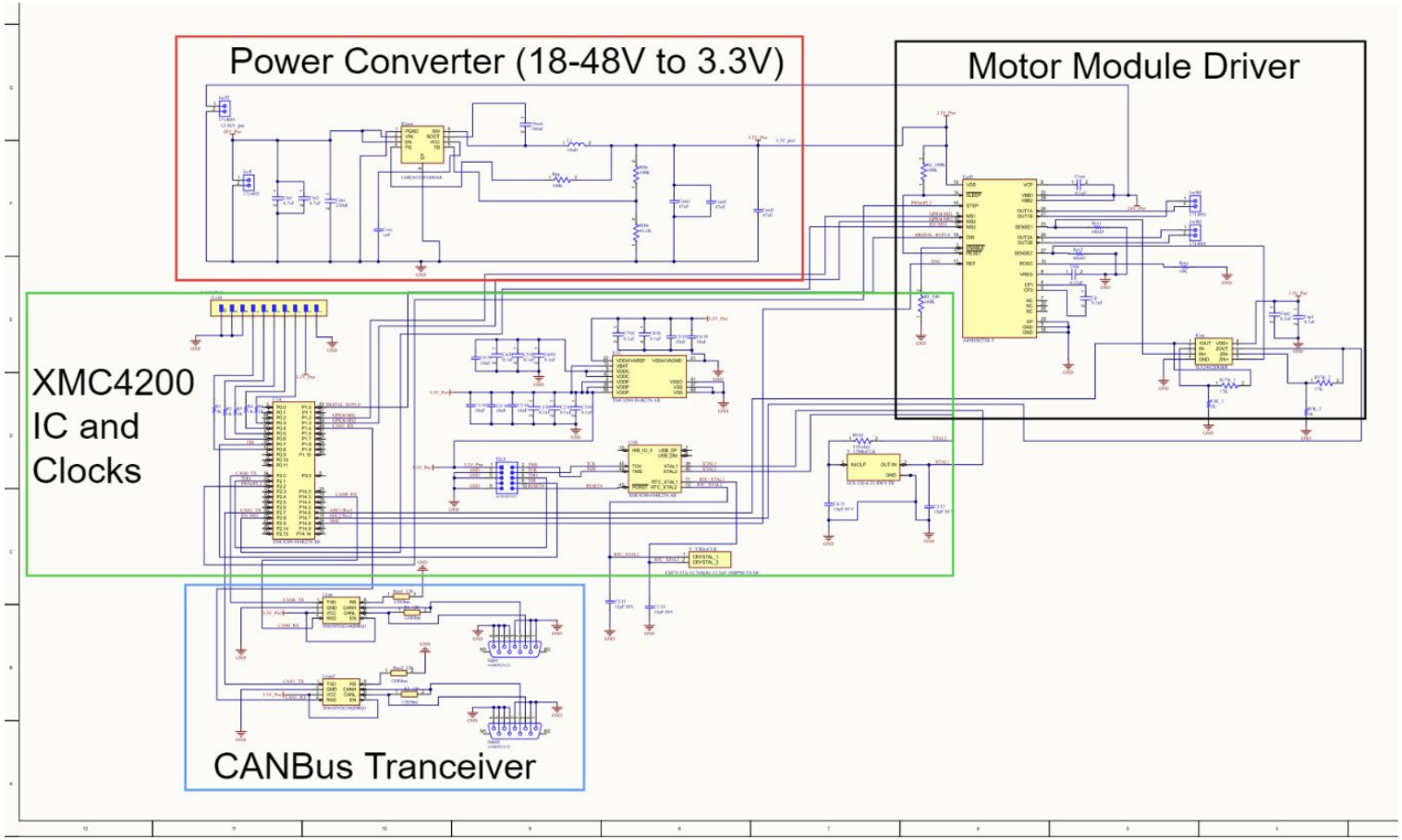
CDR Demo - Documentation

- System fully works and is integrated
  - Local connection working between two Android Phones
- User boots up Android Phone Application
  - Boot up the AWS Protocol Task.
- NMC Responds with an acknowledgement.
- Send a command to open the window to a specified percentage, in intervals of 10% (0% - 100%)
- Manual override switch to adjust the window manually, in intervals of 10% (0% - 100%)

CDR Demo

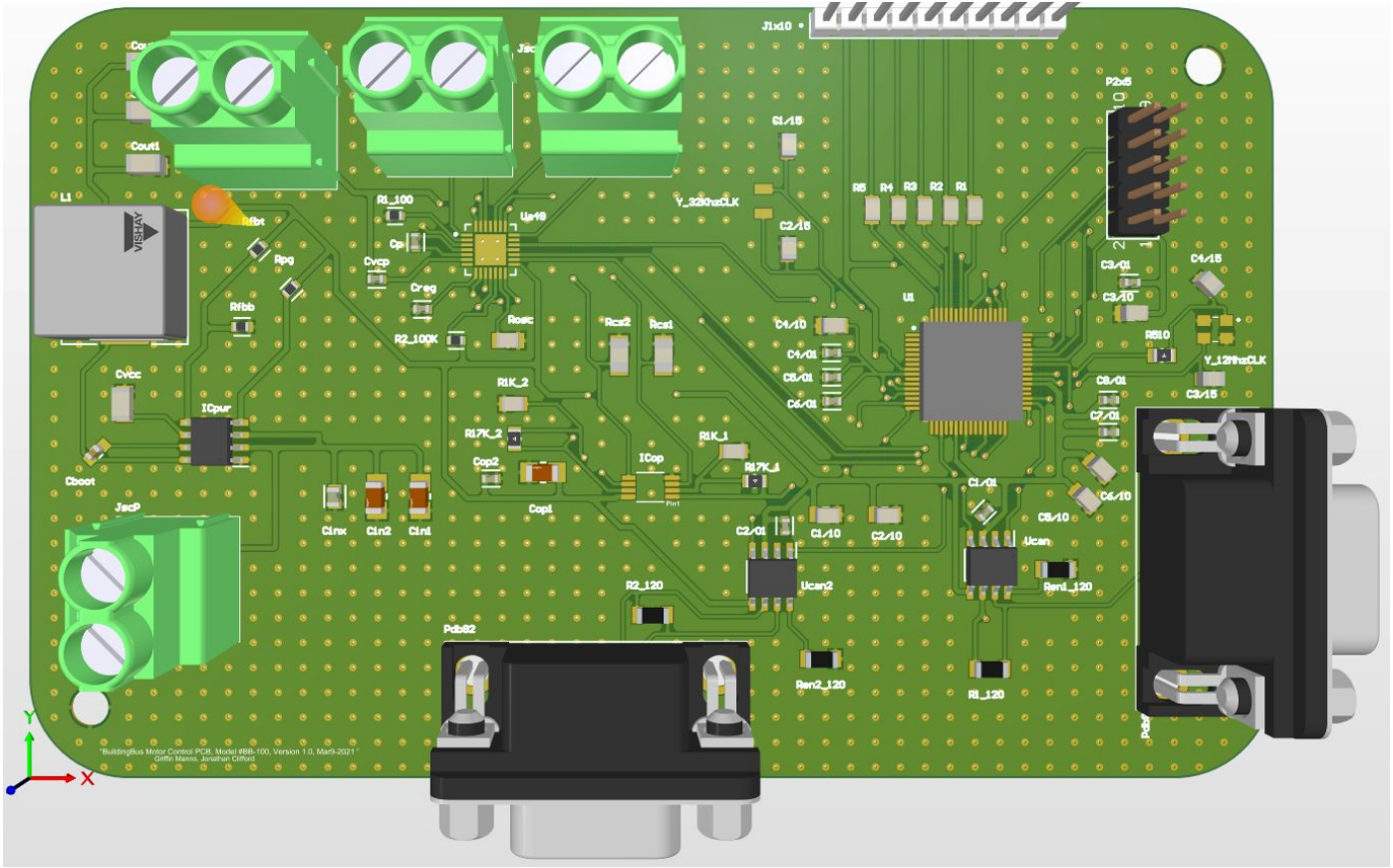
PCB Status & Overview

UMassAmherst



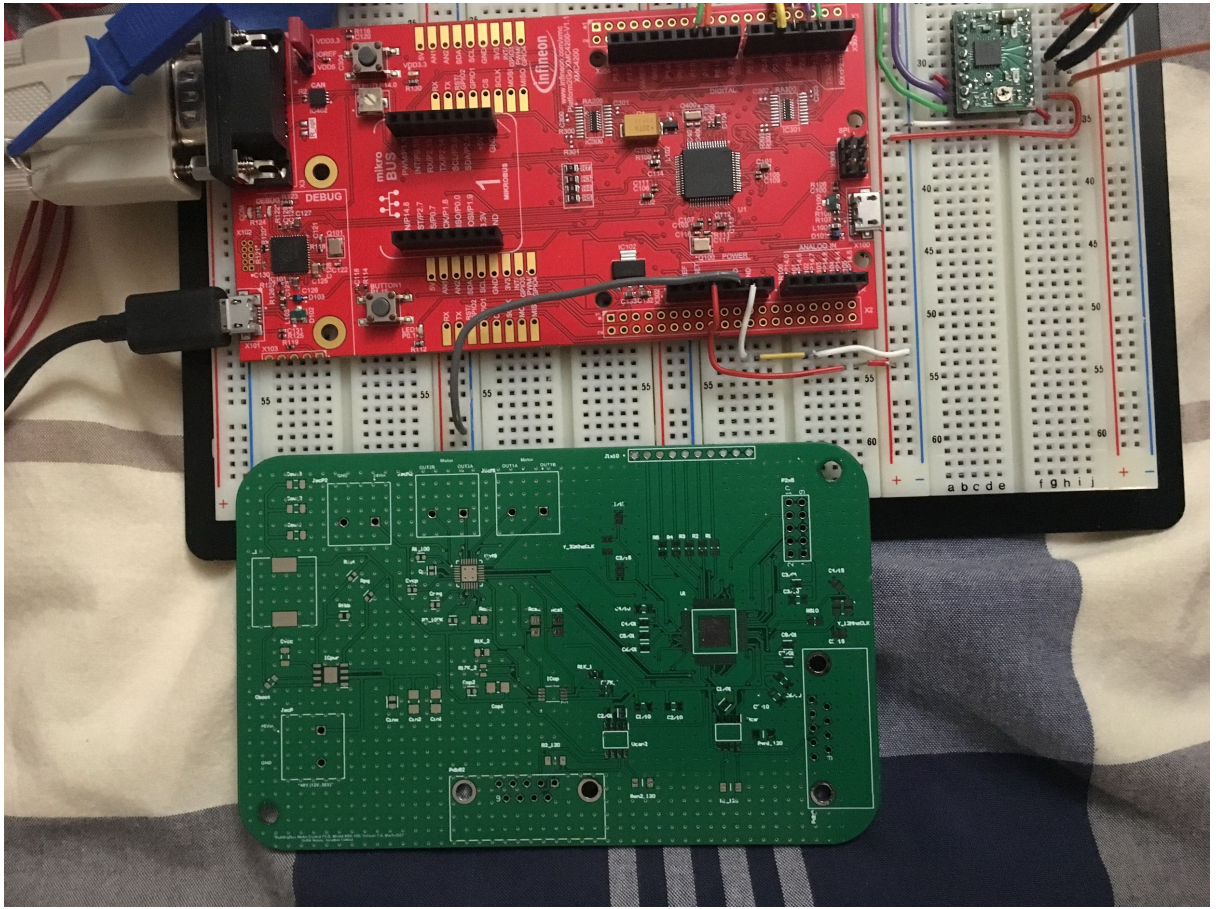
PCB Status & Overview

UMassAmherst



PCB Status & Overview

UMassAmherst



FPR Plan

- Full end to end connectivity
- Fully assembled PCB
  - If the PCB is not working properly, then full documentation of the failure points and what can be done as an improvement will be provided.
- Local connection backup software functionality
- Quality of Life Features on the app
  - Includes images as opposed to current text output

Hardware Plan For FPR

- Fully assembled PCB for the Motor Module with all 70 Components
- The Window will be fully functional
- The Network Master Controller (NMC) board will remain as-is

Hardware & Software Used

UMassAmherst

Software	Hardware

• DAVE IDE
  ◦ DAVE APPs
• Android Studio
  ◦ AWS MQTT Library
• Amazon Web Services (AWS)
  ◦ IoT Core (Message Streams)
  ◦ Cognito
• Altium
• Texas Instruments WEBENCH
	• XMC-4800 Microcontroller
	• XMC-4200 Microcontroller
	• A4988 Motor Driver
	• 24V DC Stepper Motor
	• Power Supply
	• CANBus Wire
	• Window Hardware (wood, pulleys, switches, etc.)

Project Management

Jonathan Clifford
Team Coordinator & AWS,
Android Application Developer

Griffin Manns
Stepper Motor Module
& Altium Lead

Professor Arman Pouraghily
Advisor

Frank Cremonini
CAN Bus
Communication

Connor Moore
Network Master Controller
& Budget

Mr. Glenn Weinreb
Manhattan2 CTO & Contact

Gantt Chart - Spring 2021 - FPR

UMassAmherst

Task/Week Of	5-Apr	12-Apr	19-Apr
Local Connection - Finishing Up			
PCB Soldering & Debugging			
Final Testing			
FPR			

Jonathan	Frank	Connor	Griff	All	Free	CDR/FPR

Project Expenditures

UMassAmherst

Current Expenses:

Part	Price
XMC4800 Eval Board	95.93
XMC4200 Eval Board	55.47
Stepper Motor	50.00
24v Shield (motor driver)	25.97
Motor Driver	8.00
XMC4800 Processor	23.65
XMC4200 Processor	14.62
XMC 4000 debugger	95.00
PCB components	22.89
Custom board version 1	58.15
Custom board version 2	35.11
Mouser order for crystal oscillator	14.79
DigiKey order (remaining pcb parts)	49.37
TOTAL	$548.95

Estimated Future Expenses:

Part	**Cost ($)**
None	0

Overall Expenses (Current + Future)	$548.95

Questions & Answers

CDR Demo - Backup Video

UMassAmherst