Universal Motor Controller Development

Week 1 (June 21-28)

Summary: The first week was a lot of setup and research. Development with the tm4c requires several tools to do effectively, and stepper motors take some amount of knowledge to drive. For the first week, there was little documentation created, and the details of what the controller was actually going to do was murky— essentially, in this week, the motor was made to spin and stop.

Research: I referred to section 10.5 of Embedded Systems - Shape The World by Jonathan Valvano for information on how to interface the tm4c with a stepper motor. As well as the data sheet for the tm4c, and the datasheet for the 28byj-48 stepper motor found easily online.

Setup: The setup for week 1 of development with the tm4c used the following:

Hardware

- The tm4c board
- A micro-usb cable
- 28byj-48 stepper motor
- ULN2003 stepper motor driver board
- An arduino power supply module
- Jumper wires
- A 9v DC wall adapter

Software

- An IDE (VsCode)
- A cross-compiler (arm-none-eabi-gcc)
- A flashing tool (lm4flash)
- A build tool (make)
- A debugger (arm-none-eabi-gdb)
- An on-chip debugger (openOCD)
- A startup file and linker script for the tm4c
- A makefile

For the hardware setup I connected the devices as shown in the below image:

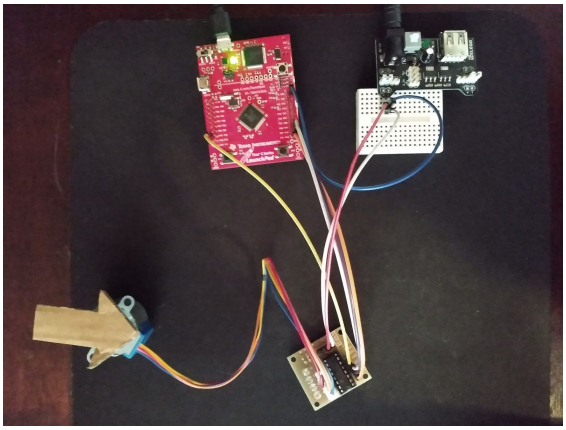

Figure 1: The hardware setup. The Tm4C connects to the driver board (bottom center) through 4 digital output pins. The power supply unit (top right) supplies 5V DC to the driver board, and has a common ground with the Tm4C. The output of the driver board connects to the stepper motor (bottom left).

Design: See the design document. This was done retroactively at the end of the week, after the idea had become more clear.

Software: Each of the four functional modes described in the design document was developed and works in software for the 28byj-48 stepper.

Work to be done:

- The inputs to the controller are currently hard-coded into the software, as such, physical inputs need to be set up for interfacing.
- There are some algorithms in the code which can most definitely be improved.
- Use is currently confined to the 28byj-48 stepper, and abstracting is needed to facilitate use with the nema-17.