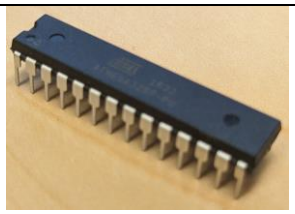
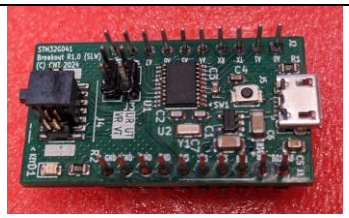
# CMPE2750 Prototyping Progress (Breadboard)

This document covers developments since the project proposal, to help you determine where you should be in the process. You want to be moving into the breadboard phase, and setting up the toolchain for the development process ASAP.

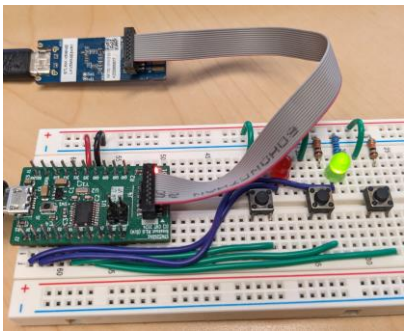
In my proposal I discussed the 'fake' development board I wanted to 'purchase' for breadboard development. A development board is actually not available for the STM32G041F8 variant I want to use, so I designed one and sent it out for manufacture. Students would typically use a chip like the ATmega328P that can be directly installed into a breadboard, or simply purchase a development board that provides access to chip pins. So I now have a development board to work with.

You will need to determine which micro you are using, and get some hardware in a breadboard. If you are going the route of the ATmega328P/PB, this process is well documented in the '*Using the ATmega328P with Microchip Studio from Scratch R1_8_6*' document found in the course LMS.

| | | |
|---|---|---|
|  |  |  |
| ATmega328P DIP chip, available from Tech Services in J105 | My development board, would have been purchased, if available. | Commercially available development board (comes with built-in debugger) |

Because I am using a different micro in the sample project, I will cover some of the toolchain elements here.

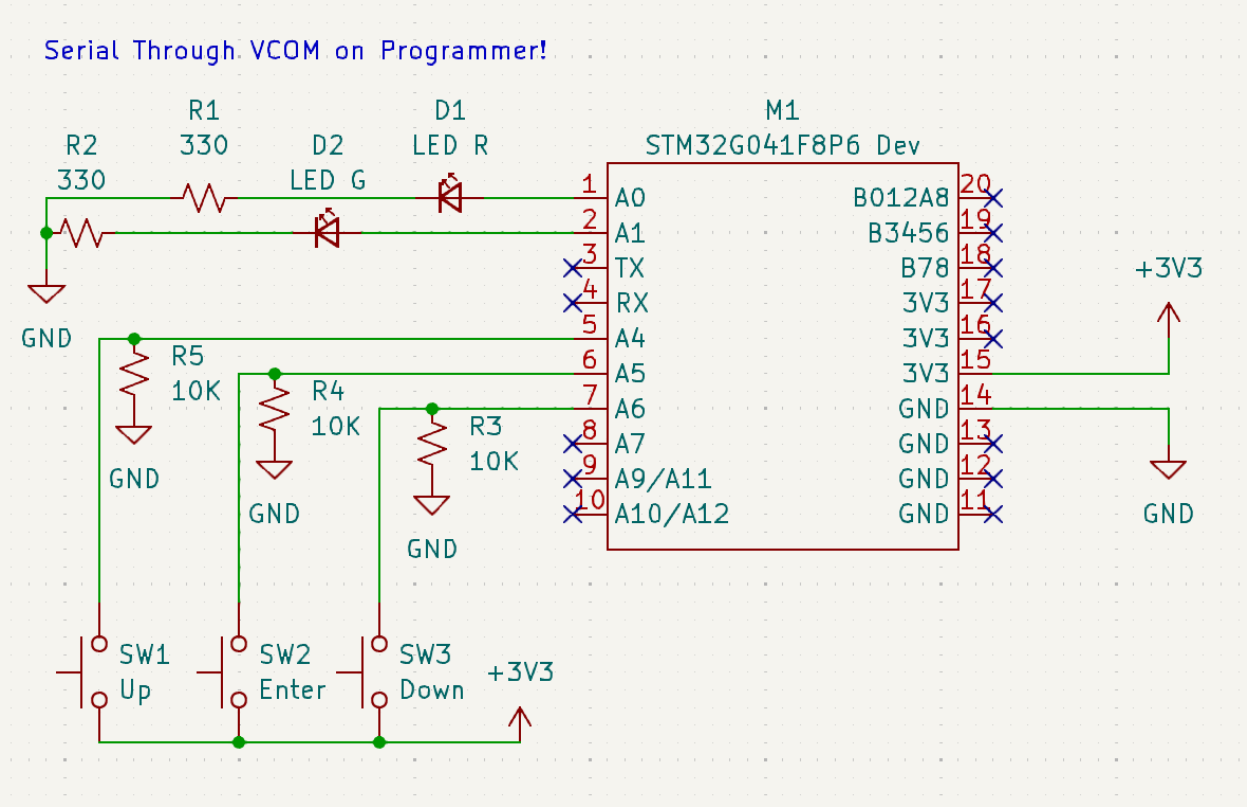
My development board principally serves to power and carry the STM32G041F8P6 chip variant, while breaking out the pins to make them accessible. I included a reset switch, external 32kHz crystal, and programming interface. The ST-LINK-V3-MINIE programmer supports a VCOM port, so I mapped the USART pins from the micro to the VCOM interface on the programmer through a jumper header. Here is the development board in the breadboard, with the programmer connected. I've also added some LEDs and some switches which may be included in the final design (the number of switches and LEDs is not yet finalized). I also have not included the OLED display and accelerometer, but these items will be connected from another external development board module.



You should be at the point where you are accumulating hardware that you intend to use in your breadboard prototype, including any sensors or devices you need to operate in your project. If the items can't be connected to a breadboard directly, you will need to purchase modules that expose the pins. Usually ADAFRUIT, Amazon, and other web offerings will supply such modules. Do shop around – different vendors offer radically different prices and supporting documentation on their sites.

If you are interested in the schematic or PCB layout of the development board, the dev package is available on the course LMS as 'STM32G041F8P6_dev' in the *Design Notes* of the *Paperwork* module.

My initial schematic for the breadboard only contains LEDs and Switches, and uses the development module as a single module symbol:




In order to get the toolchain up and running, I wanted to use Segger as a development environment, with the STLINK V3 MINIE as the programming interface. NOTE: if you use a canned NUCLEO board as your development board, you do not need an external programmer.

The default tools for working with ST chips are ST's own offering of the CUBE IDE and optionally CUBE MX, which allows visual configuration of the device. Segger is a very simple and lightweight development environment. It would actually make sense to install all packages, as the CUBE tools can help validate your design choices, and can actually generate code that you can steal to make your project work, although this is a steeper learning curve.

You will be provided with some starter libraries (eventually), so you don't need to research every aspect of programming your micro. Links for the tools will be posted in the Teams channel.