Lab 1 – Feasibility Model Phase 1

ECE 298 – S2021

|  |  |  |  |
| --- | --- | --- | --- |
| **Team Members** | Team Member 1 Barry Cimring  Team Member 2 Anweshi Anavadya | **Team Number** | 90 |

# Part 1 – Project Design Requirements

1. **Functional Requirements**
   1. The speed and direction of both wheels must be controlled by two (2) user inputs in the form of potentiometers acting as speed and steering inputs. The result of the potentiometer movement must correspond to max/min speeds and max/min turning directions.
   2. The motorized wheelchair must be able to rotate its wheel at an approximate maximum speed of 1.7 m/s and minimum speed of -1.7 m/s, which is approximately the speed of walking human adults.
   3. The controller must have two modes of operation: a mode in which the user can control the movement of the wheelchair (Run mode) and a mode in which there is no motion of the wheelchair (Locked mode).
2. **Non-functional Requirements**
   1. The ramp-up of the motor that controls the speed of the wheelchair must be pleasing to the user of the wheelchair.
   2. The bulkiness of the hardware components of the wheelchair controller must be small enough such that the user would not believe that it looks out of place on the wheelchair.
   3. When purchased, the ease of setup must be such that the user is required to perform as minimal operations as possible (i.e., it is fool proof).
3. **Constraint Requirements**
   1. The total cost of the wheelchair control system must not add more than $100 to the total cost of the wheelchair.
   2. The wheelchair control system must not be more than 0.5 kg. This is to keep the weight of the wheelchair, for a human’s perspective, unaltered, since the typical weight of the wheelchair is much heavier than this.

# Part 2 – Project Considerations for I/O

## **Project Sensors and User Inputs**

**Battery-Level Sensor**

**Battery-Level Sensor Description**

A circuit will be connected to an external battery powering the controller to measure its voltage level over time. It is assumed that the maximum possible battery that this project requires is 12 V. The battery-level sensor range must be able to transform a 12 V voltage into a voltage on the range of 0-3.3V as input to the MCU’s ADC.

**Batter-Level Sensor Connection**

The circuit that will sense the battery voltage level, whose maximum possible value is assumed to be 12 V, will be connected to the ADC peripheral device of the MCU. The voltage of the battery will be sent through an op-amp buffer (to isolate it from the ADC and minimize the ADC injection current), then the output of the voltage buffer will be connected to a voltage divider whose output range (for an input range of 0-12 V) is 0-3.3 V, the maximum voltage of the ADC. This can be achieved with a 220 k and 100 k resistor. For testing, the parts used are the ECE298\_GEN\_OPAMP for the voltage buffer, and ECE298\_GEN\_RES for the resistors in the voltage divider.

**DC Motor Encoder**

**DC Motor Encoder Description**

The motor encoder that will be used as a sensor to the motor in this project is a built-in component of the DC motor. The motor encoder outputs 3 square waves, whose frequency is dependant on the speed of rotation. The output labelled ‘IDX’ sends a pulse whenever the zero value of the wheel is rotated around (i.e., a revolution has occurred). The two outputs labelled ‘Q1’ and ‘Q2’ will determine the direction of the angular speed (which can be forwards or backwards, depending on which signal’s edge is high when the other is low) and the absolute angular position. The requirements for rotation speed are a maximum and minimum of 360 RPM, which is justified in the DC Motor description.

**DC Motor Encoder Connection**

The outputs of the motor encoder (IDX, Q1, Q2) will be sent as inputs to three of the GPIO pins of the MCU. Since the outputs of the encoder are at 0-5 V (corresponding to ‘0’ and ‘1’), a circuit to scale the voltages to a 0-3.3 V range is necessary for the MCU’s GPIO pins. Additionally, the external battery will power the VCC rail of the motor. A counter will be used to time the rate at which IDX pin transmits a pulse (to determine rotational speed) and an input GPIO pin will be used to determine which of Q1 or Q2 is high while the other is low to determine the direction of rotation via polling of the voltage levels. The ECE298\_FAST\_DC\_MOTOR\_ENCODER will be used for this component for testing.

**Potentiometer**

**Potentiometer Description**

The potentiometer will act as a user input to the control system that determines the speed and direction that the left and right motors will turn. The range of the potentiometers should be from 1 k to 100 k which will be connected to an analog input pin of the MCU in a voltage divider circuit (more described below in ‘Potentiometer Connection’).

**Potentiometer Connection**

Both potentiometers will be connected as the first resistor in a voltage divider circuit, which is powered by the battery. At the node between the potentiometer and the second resistor, one of the ADC input pins will be connected. This second resistor will be 300 k to ensure that the minimum sensed voltage of the ADC is 12 V = 0.03987 V and the maximum voltage is 12 V \* 3 V. This will correspond respectfully to maximum/minimum speed and maximum left/maximum right turning for speed and steering. To identify whether a change in the potentiometer has been made, the potentiometer’s input will trigger an interrupt that the MCU will deal with my changing the speed and steering direction of the wheelchair. The potentiometer used for testing is the ECE298\_GEN\_POTENTIOMETER.

**Switch**

**Switch Description**

A switch that controls the voltage of a particular wire will be used as a user input for two specific inputs. One switch will be used as an On/Off button for the system, and the other will be used to switch the mode of operation between Locked Mode and Run Mode.

**Switch Connection**

The switch will be connected in series with a resistor that is sourced by a GPIO pin set to output 3.3 V. The output of the circuit will be sent to another GPIO input pin that will sense whether the voltage level is high or low. If the switch is closed, the GPIO input pin will sense just under 3.3V or ‘1’ (due to the voltage divider circuit). If the switch is closed, the GPIO pin will be connected to a pull-down resistor to ground, where it will sense a ‘0’. This signal will be processed to determine which mode the controller is in. This will be achieved via an interrupt. The switch used for testing is the ECE298\_GEN\_SWITCH.

## Project Actuators and User Outputs

**DC Motor**

**DC Motor**

An actuator required for the project is a DC motor that will control the speed of rotation of the wheelchair’s wheels. It is assumed that the ratio of the number of revolutions of the motor to the number of revolutions of the wheelchair’s wheels is 6:1. Thus, as per the functional requirement (max/min wheelchair speeds of , the maximum/minimum rotation speed of the motor must be (if we assume that the approximate wheel is of 0.3 m radius): . Therefore, the *estimated* max/min rpm of the motor required is . To allow for error in the wheel radius estimate, motor controller must be able to reach a motor speed of 360 RPM.

**DC Motor**

Because the DC motor requires a negative voltage to spin in reverse, the motor will be connected to two timer pins (TIMx) connected to each side (polarity) of the motor. The input to the motor will then be controlled by transistors acting as switches. The TIMx pins are channels that are capable of outputting a PWM signal, the duty cycle of which will control to speed of rotation of the motor. Additionally, an amplifier circuit to amplify the PWM signal is necessary to reach the prescribed speeds required. In the amplifier circuit, the op-amp will scale the voltage from 0-3.3 V from the MCU pin to 0-12 V for the motor input. This achieves the desired rotation speed, as per the functional requirements. When testing the motor in part 3 we see that the range required to achieve the maximum/minimum speeds desired. The motor used for testing is the ECE298\_FAST\_DCMOTOR\_ENCODER.

**Multicoloured LEDs**

**Multicoloured LEDs Description**

Several LEDs will be used as user outputs to convey information regarding the controller mode and battery. One green LED will be shining when the wheelchair is on. Another set of LEDs are present to encode information of the battery level of the controller. If the battery is greater than 90% a green LED will be switched on. If the battery charge is anywhere between 80%-90% then a yellow LED is switched on. If the battery level is anywhere between 60%-80% then an orange LED is turned on and if the battery level is below 60% then a red LED is set to be flashing.

**Multicoloured LED Connection**

The LEDs to will be controlled via an MCU General-purpose I/Os (GPIO) pin. The GPIO pin will control the gate voltage of a transistor that is connected in series to the LED. This transistor will act as a switch to turn on the led, which will be powered by the battery in series with a resistor to control the current sent through the LED. The LEDs used for testing are LED-GREEN, LED-YELLOW, LED-ORANGE, and LED-RED.

**LCD – Liquid Crystal Display (LM016L)**

**LCD – Liquid Crystal Display Description**

The liquid crystal display (LCD) will be used to display the mode that the wheelchair is in and the RPM of both wheels on the wheelchair. The display will show 16x2 characters. The top row will display the mode that the wheelchair will be in (Locked or Run mode), and the bottom row will display the RPM of the left and right wheels.

**LCD – Liquid Crystal Display Connection**

The LCD has 11 digital inputs, power (VDD) and ground (VSS). Only 8 of the digital inputs will used as the LCD will be used in 4-bit mode to minimize wiring connections. The digital inputs on the LCD are the RS, RW (read/write), E (enable) and D[4..7] pins. The only digital output pin used will be the D0 pin, which indicates if the LCD is busy with BF (busy flag).

The LCD will be initialized and controlled via commands as described in the LM106L datasheet (examples provided in section 3). To achieve this, the pin of the LCD’s I/O pins will be connected to 8 of the 16 GPIO pins on the STM32F401RE MCU. The MCU will send initialization commands and read/write commands in 4-bit format via the GPIO channels. The LCD used in testing is the LM016L.

## Project MCU Internal Resources

**ADC – Analog to Digital Converter**

The STM32F401RE has a built-in ADC peripheral. The ADC will be used to measure the voltage level of the battery as well as the voltage level from the variable resistor voltage divider that indicates the speed and direction that the user desires.

The ADC, which has 16 possible inputs, can be read via channels that the user can set in code. The ADC can be triggered in code, sending its measured value to a memory address, then read at 12-bit accuracy on a 0-3.3 V scale (0xFFF indicates 3.3 V, 0x000 indicates 0 V).

The time it takes for the ADC to read from an MCU channel and send to memory is approximately 9 us. Given this application’s requirements of reading from a human-input and battery voltage level, this will be fast enough so that it is not required to use the direct memory transfer that the DMA can facilitate.

**Internal MCU timer**

The STM32F401RE consists of one advanced-control timer, seven general-purpose timers, and two watchdog timers. All timer counters have the ability to be frozen in debug mode. The advanced-control timer can be seen as three-phase PWM generators multiplexed on 4 independent channels. The seven general-purpose timers can be synchronized with each other to keep steady time.

The DC motor encoder will be connected to two of the MCU’s timers that will output a PWM signal. As mentioned above in the DC Motor actuator description, the DC motor’s + and - pin will receive a PWM signal that is generated by two of the timer channels of the MCU. These signals are between 0-3.3V, so a scaler circuit will be in place to scale the voltages to 0-12 V.

**GPIO**

The STM32F401RE has 16 General-purpose I/O (GPIO) pins that accept input or send output. Each of the STM32F401RE’s GPIO pins can be configured by software to act as input, output, or as peripheral alternate function.

The wheelchair controller project uses the GPIOs for processing various inputs and outputs. Eight of the sixteen GPIOs pins are connected to the LCD screen. Three of the GPIO pins are also used as an input pins for the DC motor encoder that is connected. The LEDs that are required to display information to the user are also connected as output to the GPIO pin set as an output channel. Finally, the switches that will be used for turning the controller on/off and changing the operation mode are sent as input to the GPIO pins. The GPIO functionality is very important for interfacing with digital input/output that are not easily controlled otherwise.

Since each of the GPIO pins are limited to a 3.3 V input (typically, and 5 V with several provisos) and 20 mA injection current, the hardware design must meet these requirements.

**Interrupts**

External interrupts will be necessary for the MCU to process several user/sensory inputs efficiently. The internal interrupts will be used to indicate whether the voltage level of the battery has dipped below a certain threshold and if the variable resistor has been changed by the user of the wheelchair. This prevents the MCU from polling each of the two input signals periodically to receive input.

Additionally, interrupts will monitor the input pulses from the motor encoder and measure how many pulses the MCU receives per 0.5 seconds, as well as which direction the wheels are rotating. An internal timer interrupt will then compute the instantaneous RPM (averaged over 0.5 s) and output the value and direction of rotation onto the LCD.

**Software Parameters**

The software parameters that must be kept track of are:

1. The voltage level of the battery.
2. The maximum and minimum possible battery input.
3. The angular speed of the two motors attached to the wheels.
4. The angular direction of the two motors attached to the wheels.
5. The maximum and minimum angular speeds/directions of the motors.
6. The desired user input (after processed from the potentiometer input) of the speed of the wheelchair.
7. The maximum and minimum possible user input of the potentiometer.
8. The mode that the control system is in.
9. The initialization sequence of the LCD.
10. A look-up table used for writing to the LCD.
11. The busy flag of the LCD.

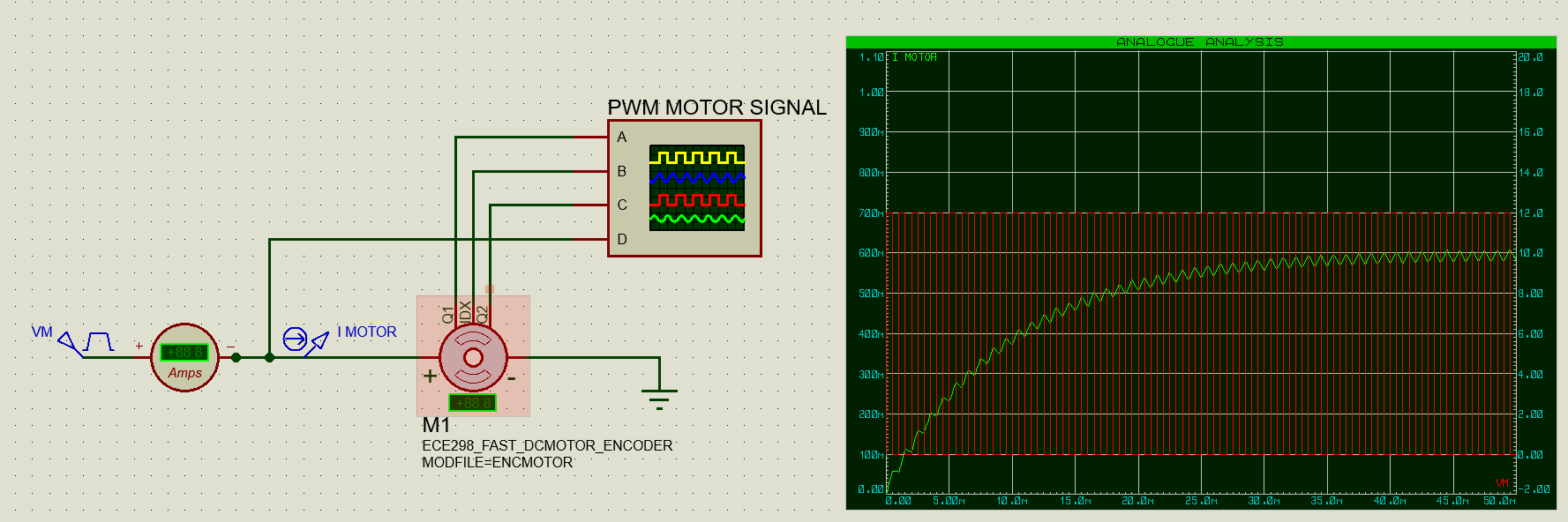
# Part 3 – Device Testing Methodology

## Device 1 – ECE298\_FAST\_DCMOTOR\_ENCODER

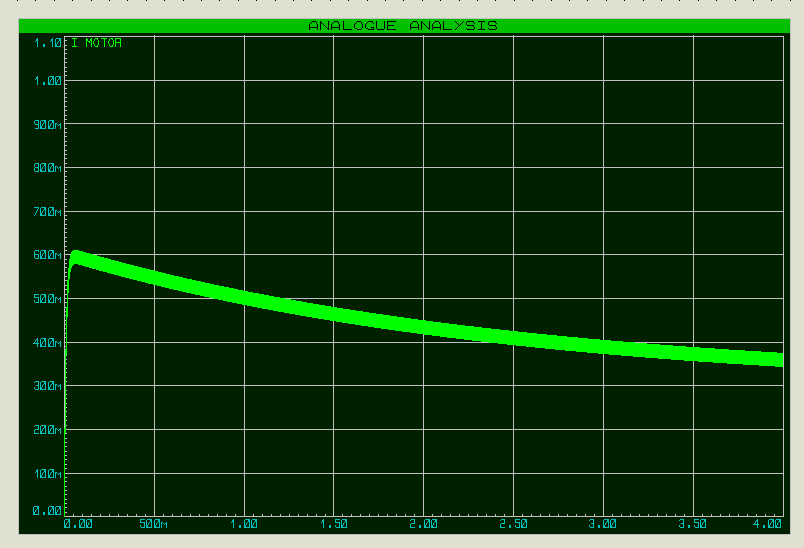
The ECE298\_DCMOTOR\_ENCODER is the DC motor that will be used to control the speed of revolution of the wheels on the wheelchair. For this application, the motor’s rotation speed is controlled by a PWM voltage, supplying an average current, and the output is sensed by an oscilloscope. The current that the motor draws from a power source is proportional to the rotation speed of the motor.

To control the device, a voltage source will be connected to the + pin and the – pin of the motor will be connected to ground. For testing this is shown as a voltage source. In reality, this will be a pin that will output a PWM signal from the MCU timer.

The schematics used to test the DC motor is below:

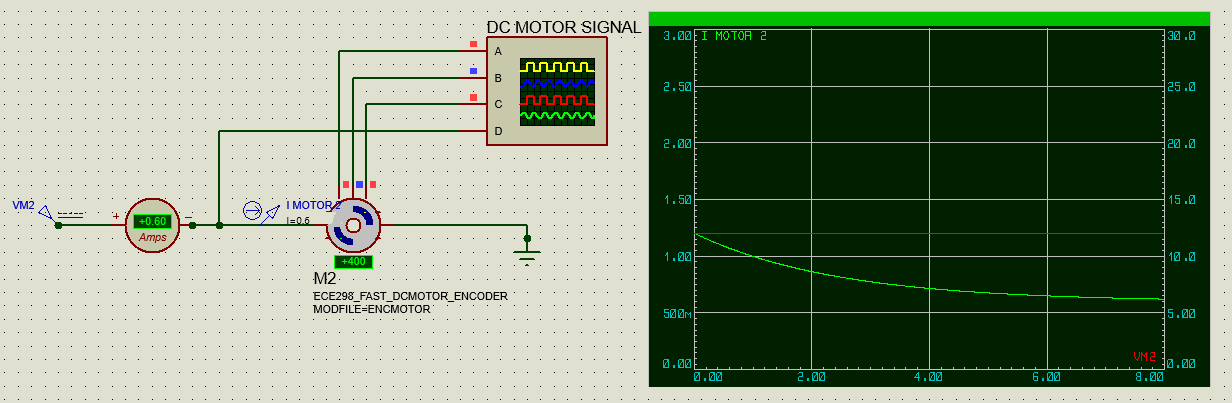


The DC motor’s positive pin is connected to a PWM signal which is at 1 kHz, V\_high = 12 V and V\_low = 0 V, drawing a current. In the figure above it is shown that when sent a PWM signal, there is a rise-time of the motor’s current, and therefore rotation speed. The initial rise time is estimated to be 30 ms which rises to double the average asymptotic current. Below is a graph of the current of the motor for a longer duration for the same startup:



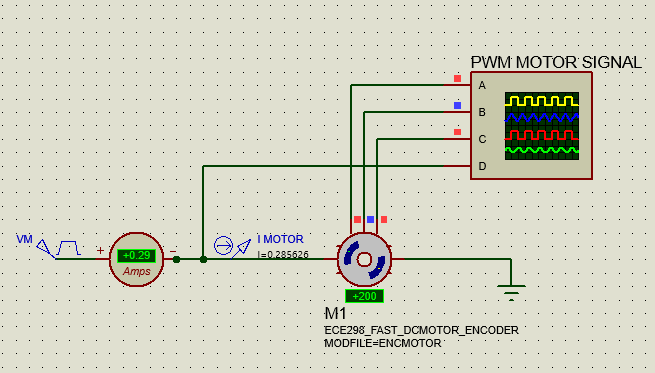
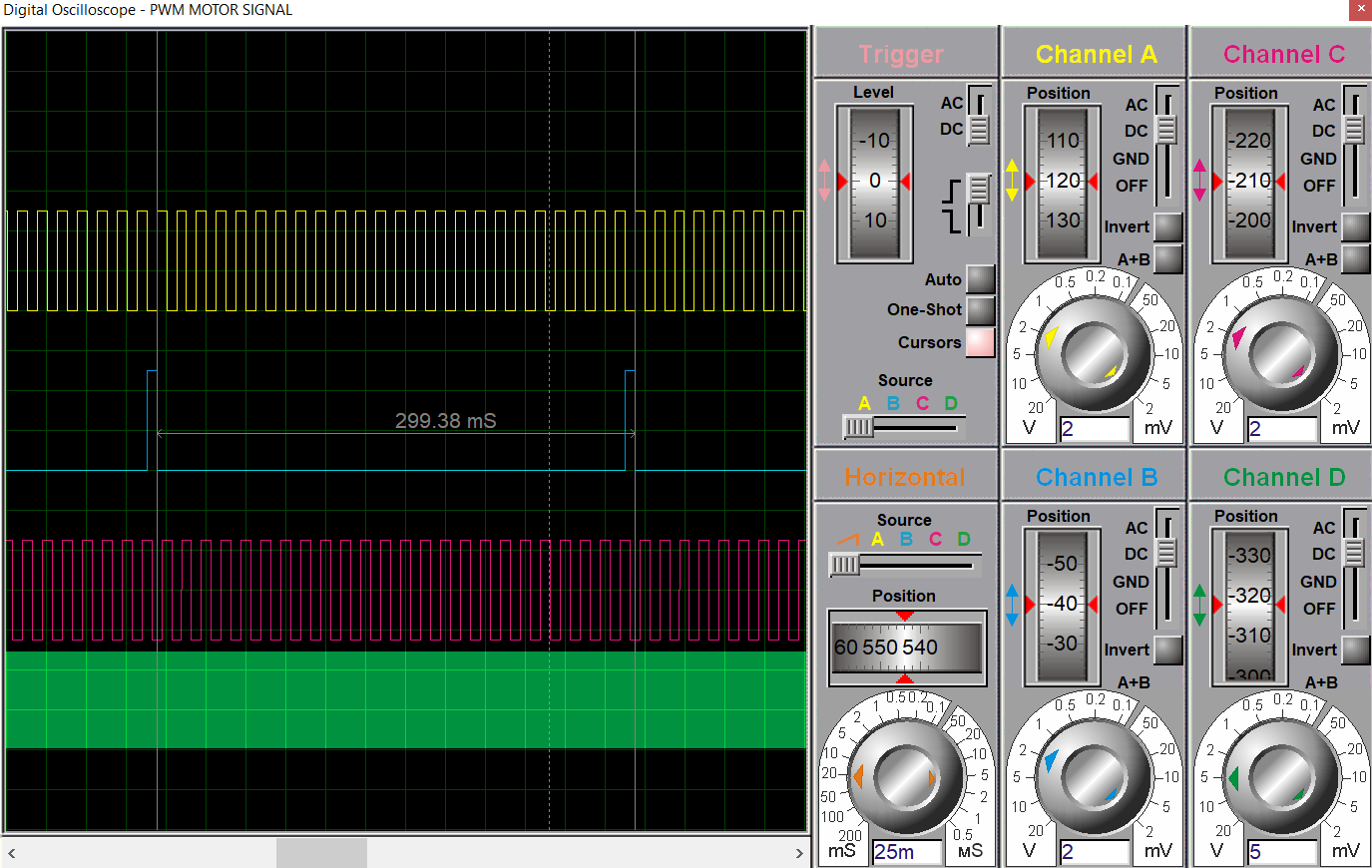
The motor current rises to more than double its asymptotic value of 0.29 A, and decays in approximately 2 s.

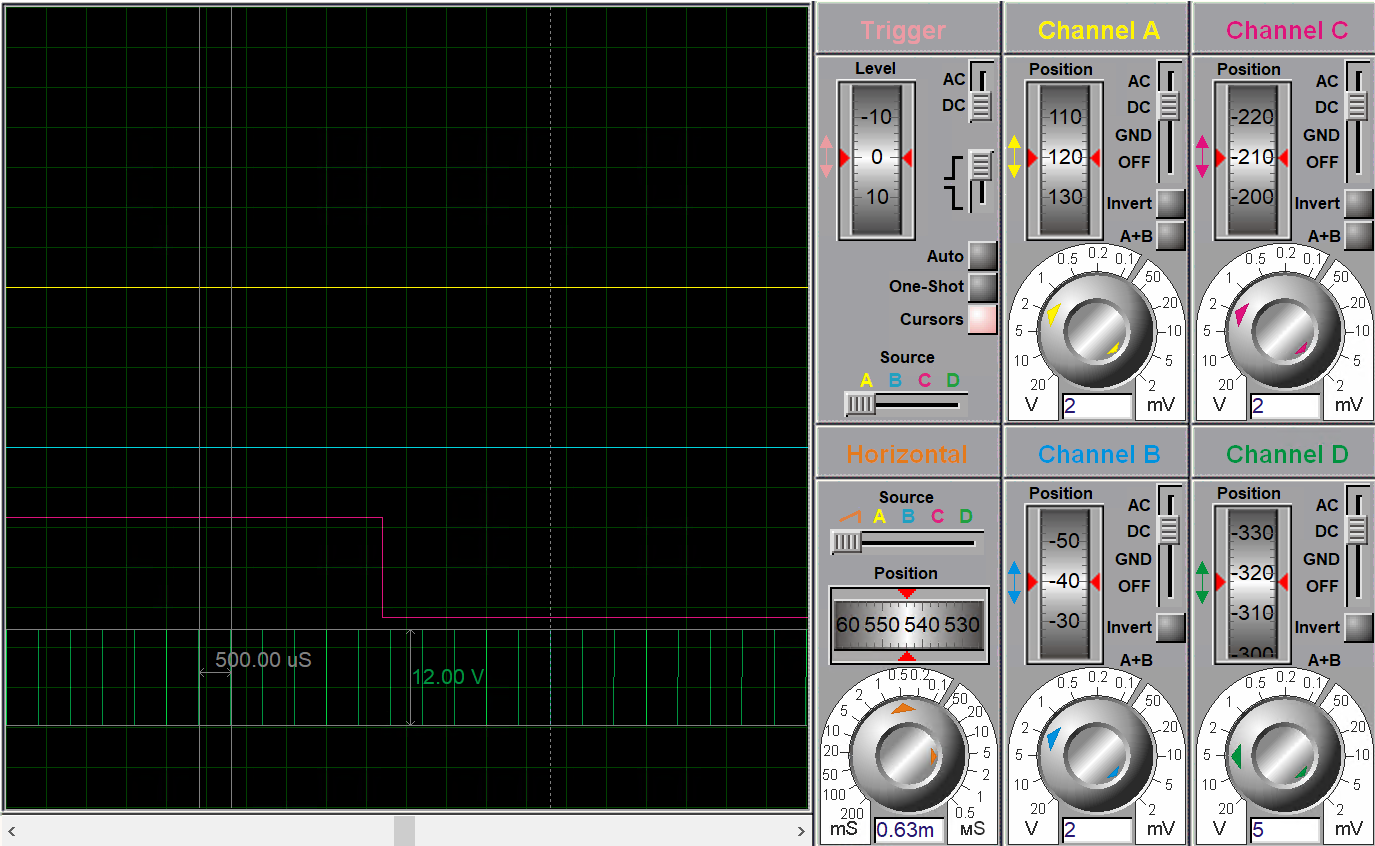
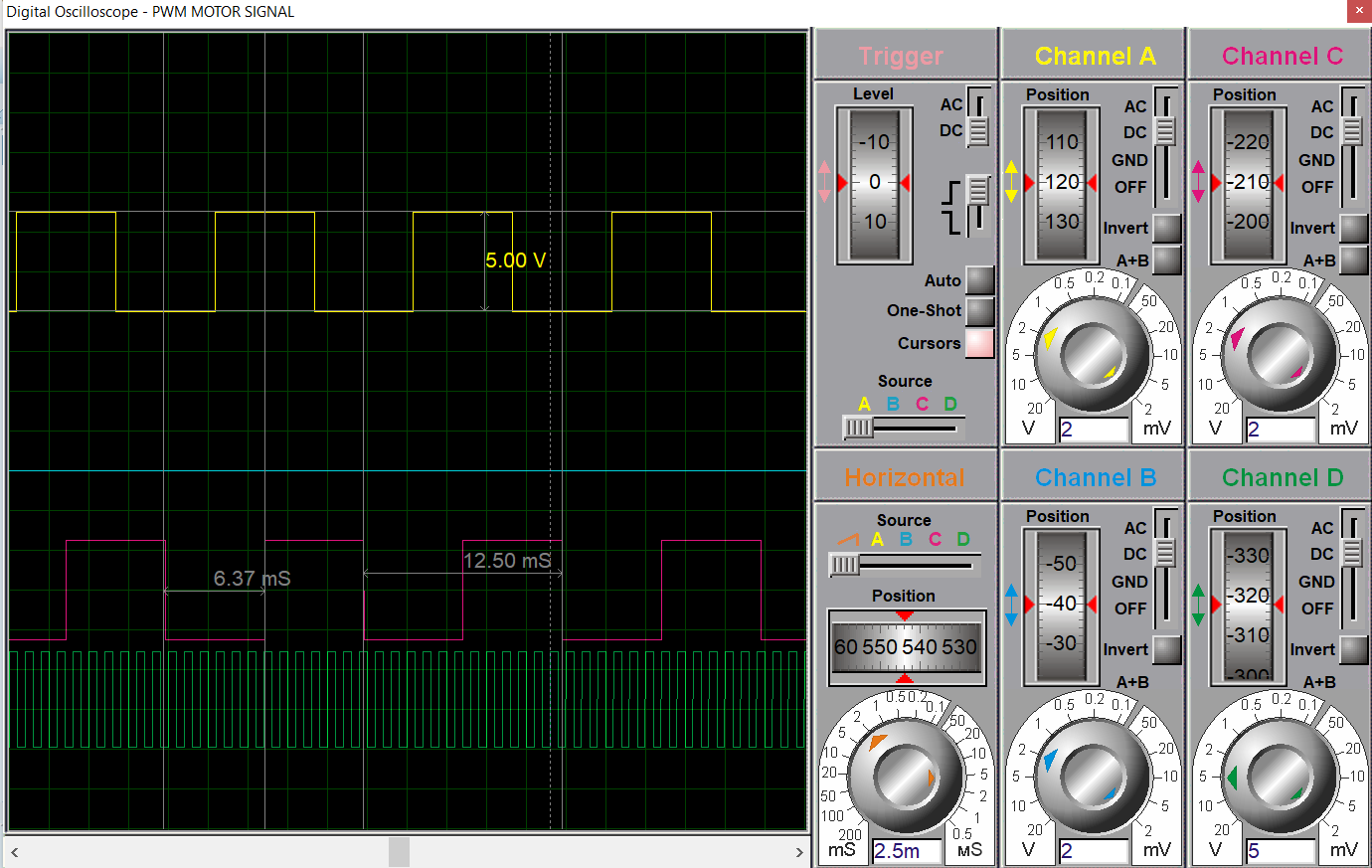
The following is a time-domain analysis of start-up of the motor. When sent a DC signal of 24 V is applied to the motor:



The current initially starts at almost 1.2 A and decays to its average value of 0.61 A. This is a lot of power required to achieve the desired rotation speed. The decay is expected behaviour as the motor can be modelled as an inductor in series with a resistor.

The output Q1, Q2, IDX pins, as well as the motor voltage are connected to an oscilloscope to monitor the output of the motor. It is seen that the rotation speed (in RPM) of the motor is directly proportional to the voltage *and* the duty cycle of the PWM signal supplied, and hence the current. When a PWM signal of 0-12 V at a 50% duty cycle at 1 kHz is sent, the motor draws an average current of 0.29 A, and motor’s rotation speed is 200 RPM:

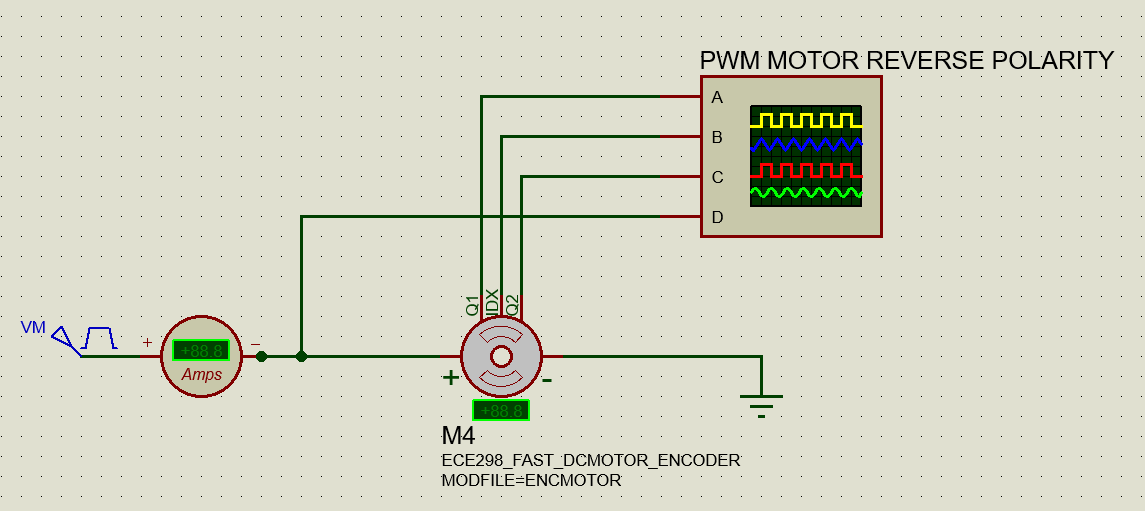
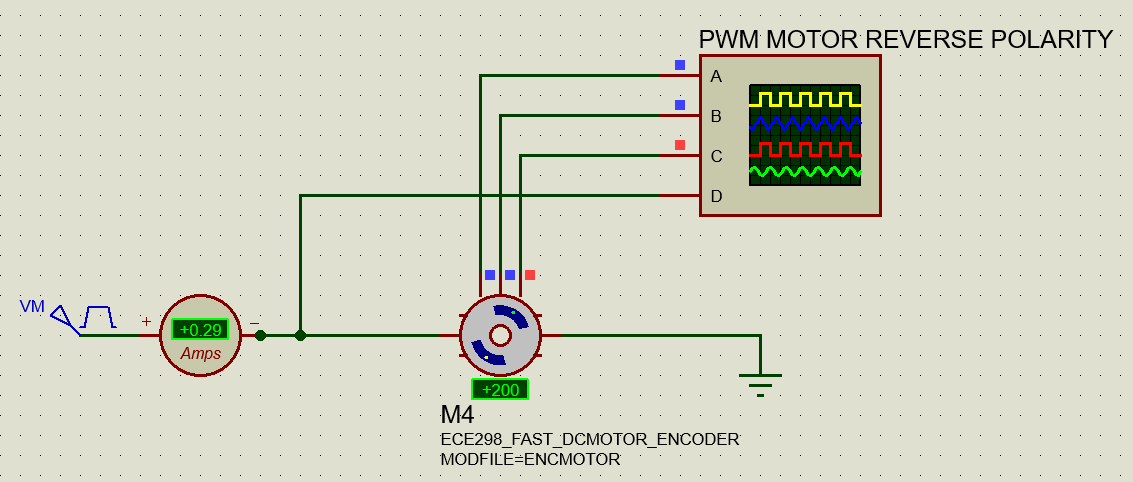
 



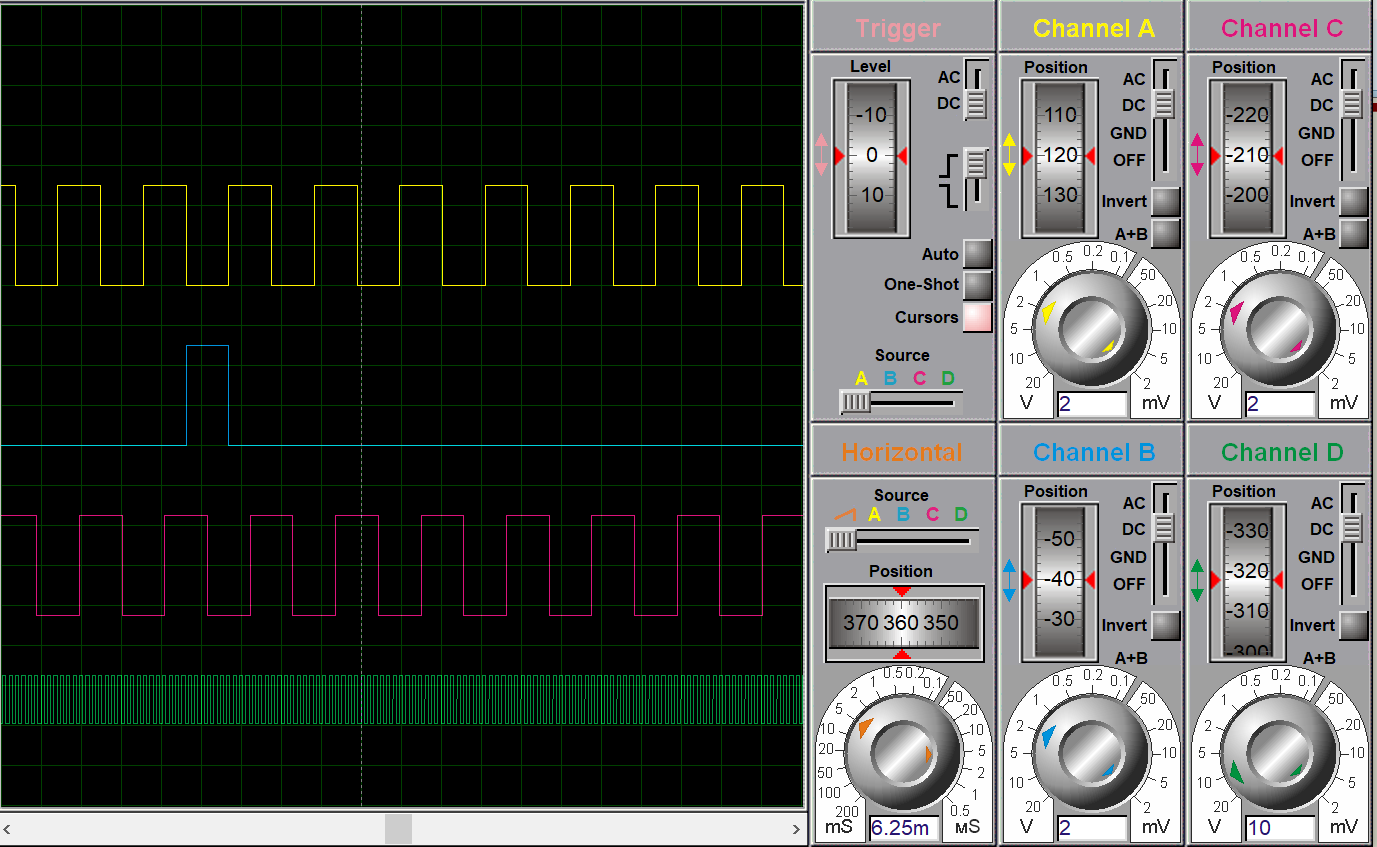
As shown, for 0. 29 A input (on average), where the PWM signal is a 50% duty cycle wave with 1 us rise and fall time, the output of Q1, Q2 is a square wave from 0-5 V with a period of 12.50 ms. This period is inversely proportional to the rotation speed. The duty cycle of the square wave (of Q1, Q2) is 50%. When the motor is rotating in the forward direction Q1 turns from low to high when Q2 is low. The period of the pulse given from IDX indicating that a revolution has occurred is 0.299 s, as expected for a 200 RPM . The square PWM signal is also shown, with its expected value of 12 V maximum and 1 ms period.

Since the 12 V DC input is equivalent to an 100% duty cycle 12 V PWM input, it is seen that the speed of the motor is proportional to the duty cycle of the PWM signal. It is seen that the current increases by 0.6/0.29 = 2.068, and so does the RPM of the motor 400/200 = 2. The periods of outputs IDX, Q1, Q2 are also reduced by the same factor.

It is seen that, in the same way, the motor will rotate in the opposite direction with the same magnitude RPM if a voltage is applied to the opposite terminals: (a PWM signal of V\_high = 12V, V\_low = 0 V and 50% duty cycle is applied)

In the figure above the pins of the DC motor are switched.



The same angular frequency occurs in the negative direction when the negative current is applied, and the rising edge of Q1 occurs when Q2 is high.

This motor has the output built-in to it so the MCU can receive input directly from the same component. Both the motor (actuator) and the motor encoder (sensor) has been tested in this section.

To test these circuits, the oscilloscope labelled ‘PWM MOTOR SIGNAL’ can be used to test the response of the motor to a PWM signal, the oscilloscope labelled ‘DC MOTOR SIGNAL’ can be used to test the response of the motor to a DC signal, and the oscilloscope labelled ‘PWM MOTOR REVERSE POLARITY’ can be used to test the response of the motor to a PWM signal with the input to the motor reversed in polarity.

These schematics verify that the DC motor behaves as an inductor and resistor (due to the start-up decay time), that the rotation speed is proportional to the current and duty cycle of the input PWM signal, that the IDX output is pulsed once per cycle, and that the motor will spin in the opposite direction when the reverse polarized signal is sent to it.

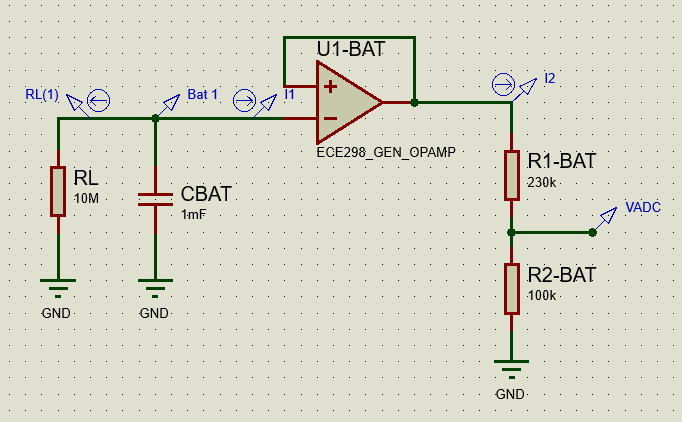
These tests also validate that the maximum required signal sent to the DC motor must be 360/400 \*0.6 A = 0.54 A on average to achieve a rotation speed of maximum 360 RPM, as per the design requirements.

## Device 2 – Battery-Level Sensor

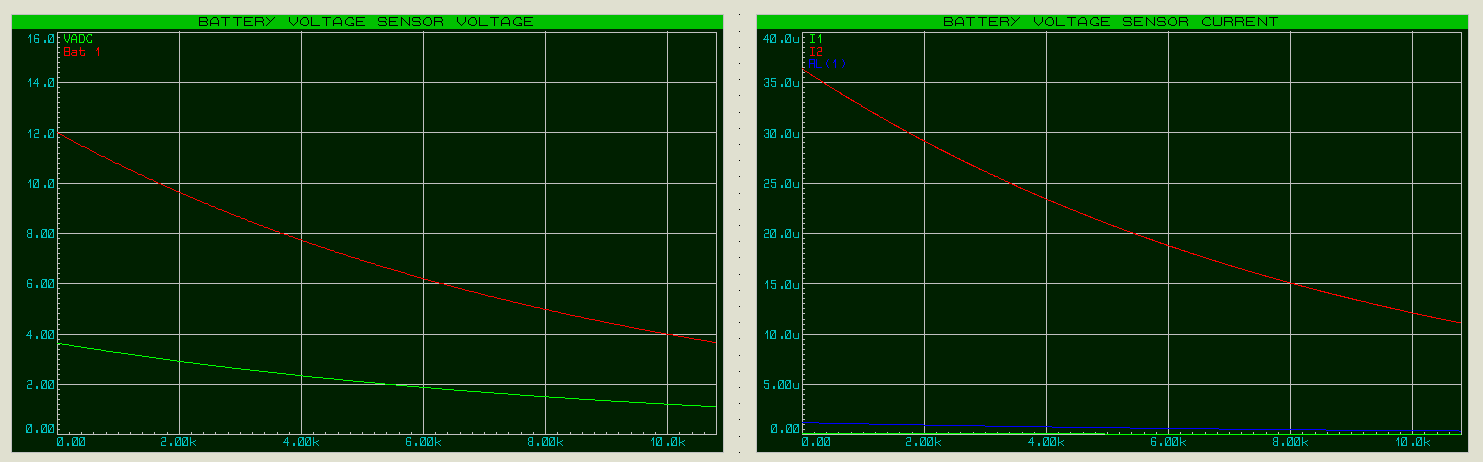
The battery-level sensor circuit is designed to output a voltage between 0-3.3 V given an input range of 0-12 V, which is the assumed maximum voltage of an external battery for this application. In the circuit below, the battery level sensor’s function is shown on graph of the voltage of the battery and battery sensor over time.

The output of this circuit will feed into the ADC peripheral of the STM32F401RE which will be converted to a digital signal that the firmware of the MCU can read. Depending on the battery level, the MCU will determine which LEDs (indicating battery percentage) will be turned on.

A schematic for the battery-level sensor is shown below:



The battery is modelled by a large capacitor – CBAT – whose initial voltage is set to 12 V. The battery voltage is sent through a buffer to isolate the signal from the rest of the circuit (RL, representing the circuit load), which is the input of a voltage divider providing the correct maximum output rated for the MCU’s ADC. VADC will feed into one of the 12 input pins of the ADC of the MCU in the realistic implementation.



Due to the modelling of the external load as a large resistor, the battery’s voltage will drop exponentially over time (since the current drawn by the op-amp buffer is negligible). The lifetime of this battery is . This is shown below in the time-domain voltage and current of VADC, Bat 1, I1, I2:

As shown, the battery, in this configuration, retains at least 50% of its charge for 6.1 ks = 1.7 h, which exponentially decreases with the lifetime as stated above. The current drawn from the battery is negligible, and the current drawn in the voltage divider circuit is of the order of uA, which is satisfactory for the injection current into the ADC. In this example the initial voltage read by the ADC would be 3.3 V – indicating to the MCU the battery is at 12 V charge.

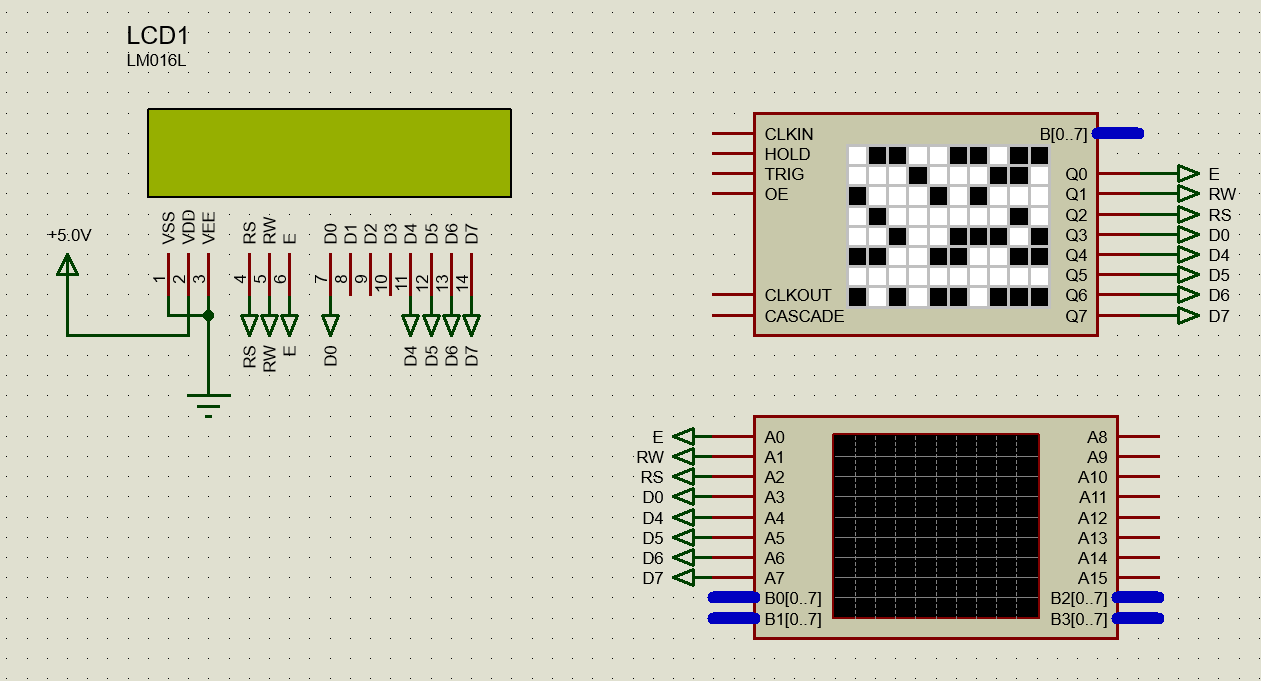
To test this circuit, the ‘PRECHARGE’ setting may be changed from 12 V in CBAT, and the resistance of RL may be altered to change the lifetime of the decay. The graphs of battery and sensed voltage/current may also be run to display the decay of the battery voltage.

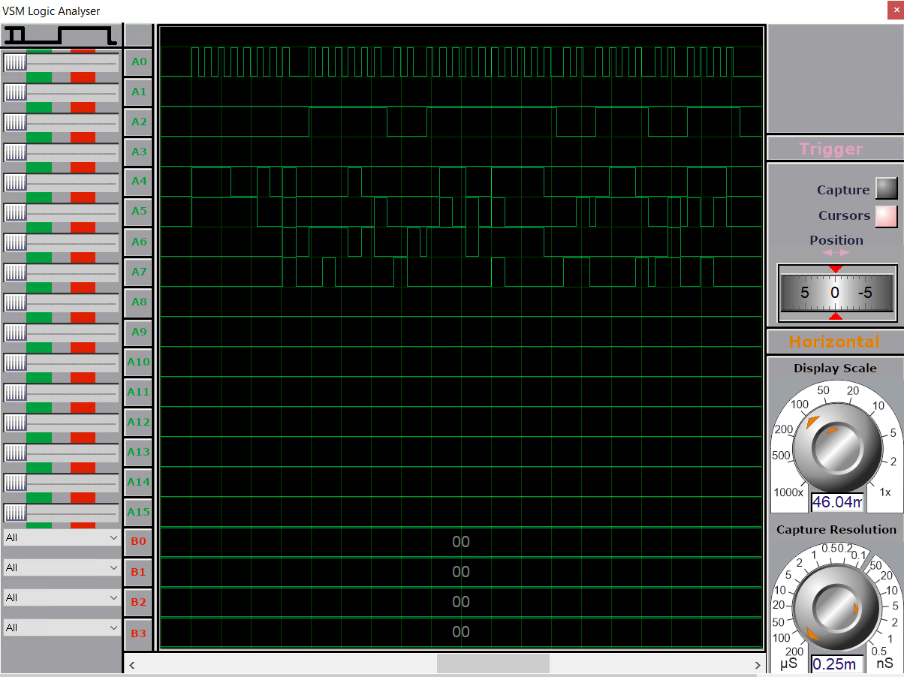
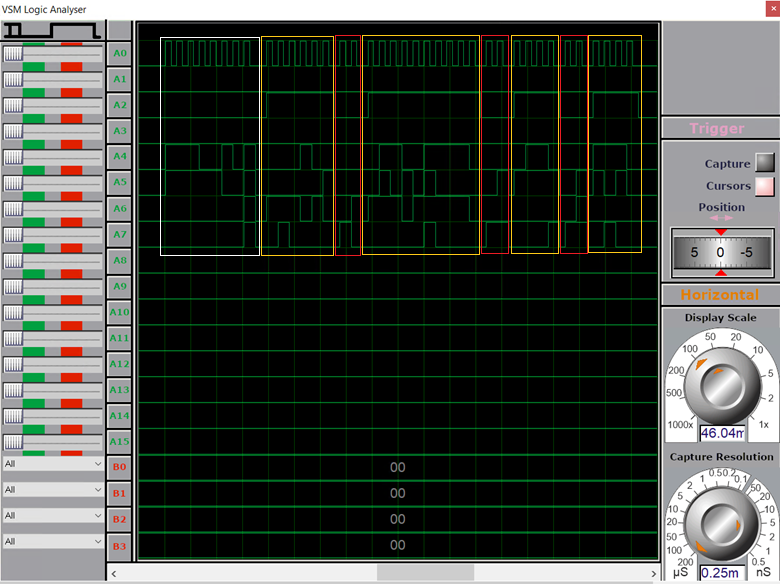
This verifies that the battery sensor will isolate and scale the voltage of a 0-12 V battery down to a voltage from 0-3.3 V that the ADC is able to read, and that the injection current into the MCU is negligible in the circuit.

These tests validate that the voltage level of the battery can be sensed via a voltage buffer and voltage divider, provided that the battery be no higher than 12 V and the output of the voltage divider is input into the ADC.

## Device 3– LM016L

The LM016L is the LCD that will be used to display the RPM and mode that the wheelchair controller is in. This will be achieved by 8 GPIO connections to the MCU that will send data and commands to the LCD. Below is the testing circuit designed to ensure functionality of the LM016L:



The VSS and VEE pins of the LCD are tied to ground, while the VDD pin is tied to +5.0 V. The digital input pins RS, RW, E, D0, D[4..7] are connected to a digital pattern generator to simulate the GPIO pins of the MCU. Below is a screenshot of the initialization process sent from the digital pattern generator as measured by the digital logic analyzer:

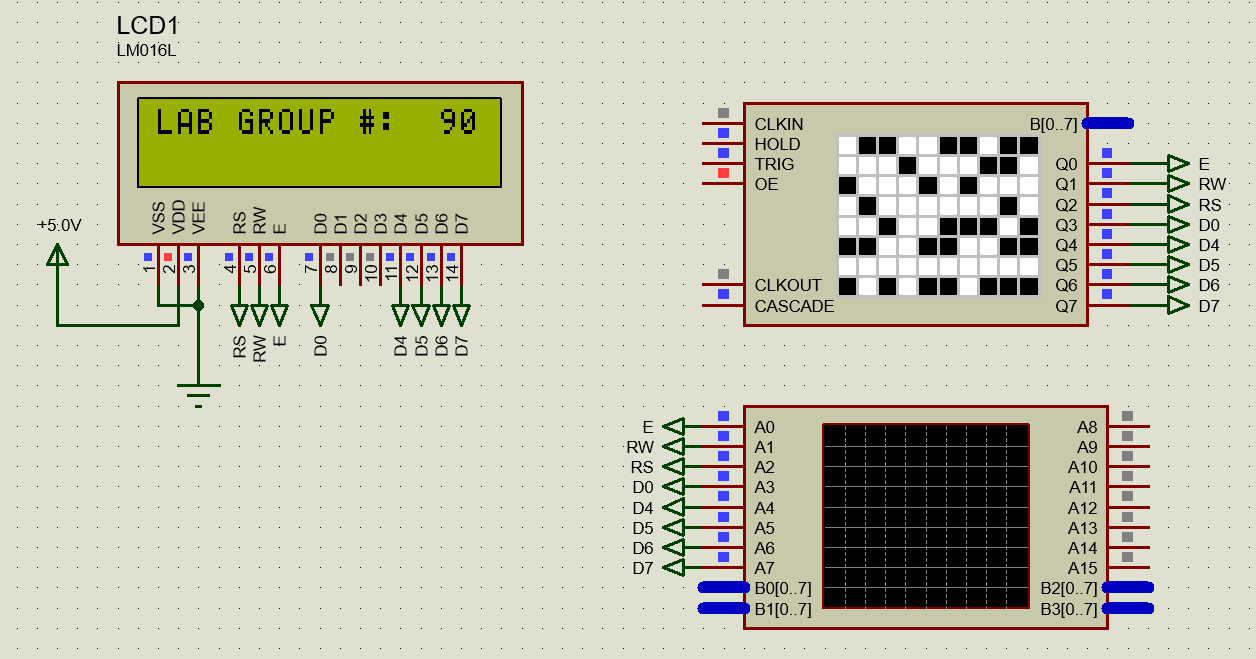
The pattern boxed in white represents the initialization procedure, the pattern in red represents changing the memory address of the LCD’s DDRAM where character data is written to, and the orange boxes show character data that is written to the LCD.

For each command, the input pins are latched to their desired configuration and the enable (E) signal is pulsed. This latches the internal circuitry of the LCD to whatever is connected to the input pins. The initialization procedure above sets the font, the cursor to blink at DDRAM position 0 and turns on the LCD display. All of the commands in the initialization set RS, RW as low.

The characters L-A-B are then written to the LCD. Each letter is written via 2-4 bit write operations, where RW is set to 1 (for writing).

The address of 0x05 is then sent to the address counter in 2-4 bit operations where D7 is high and RS, RW are low, indicating a DDRAM address is being sent to the LCD. In the 4-bit write operations the first 4 bits are D[4..7] and represent the MSB and the second 4-bits are D[0..3] and represent the LSB. The address is 7 bits long, and is in the order of D6-D0 (D[3..0] are the 4 bits sent in the second command).

The same write operation occurs to write G,R,O,U,P, and shift to address 0x0A, then write #, : and shift to address 0x0E to write the numbers 9, 0. The result of the above is shown below, where the desired data is displayed.

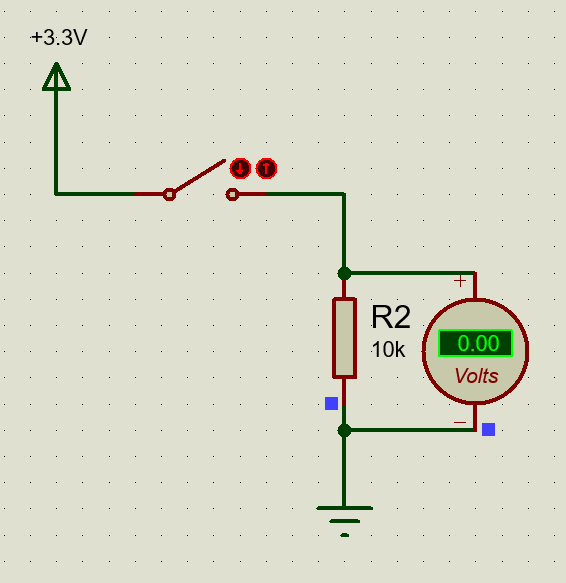


To test these simulations, simply run the schematic in animation mode to see the characters appear. The commands sent to the LCD may be changed using the logic analyzer.

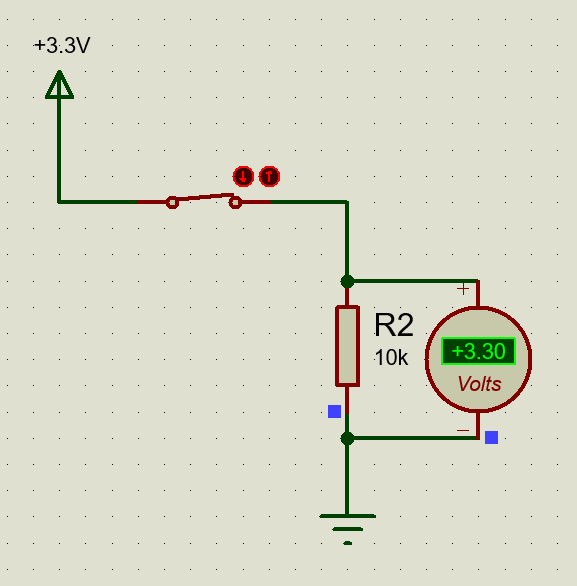
This verifies that the LCD displays 16 characters per row, that the LCD can be written to at a 10 kHz clock speed, and that the initialization procedure and mothed to write to the LCD is correct as per its documentation.

This also validates that an LCD can be used to display mode and speed information to the user in real-time to the user of the wheelchair.

## Device 4– Switch

The switches will be used to monitor whether the user has set the controller to be in ‘Run’ or ‘Locked’ mode, and whether the controller is set to on or off. The testing circuit shows a resistor connected to a 3.3 V signal that is grounded on the other end of the resistor, and a voltmeter measures the voltage across this resister to validate that function of the switch.

The circuit to the right shows us the switch in the open position, hence, the resistor is grounded with no current flowing across it, and the voltage across it is 0 V.



The circuit to the right shows the switch in the closed position. This allows current to flow through the resistor, whose voltage is shown to be 3.3 V.

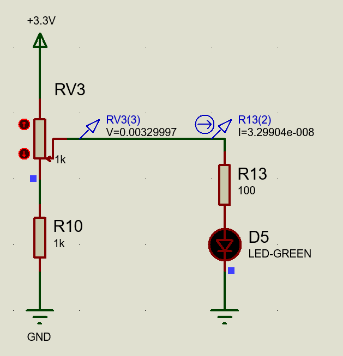
The voltmeter in these simulations models what a GPIO input pin of the MCU would measure from the switch.

To test this circuit, the switch may be toggled between open and closed and the resulting voltage is shown on the digital voltmeter.

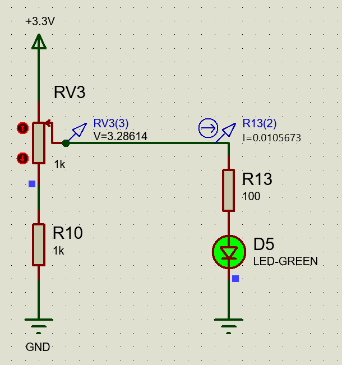
These tests confirm and verifies that that the toggling of the switch toggles the current flowing through the circuit, which may be sensed by a GPIO pin on the MCU.

It is validated that the GPIO pin will sense a 0 V or 3.3 V signal (‘0’ or ‘1’) when a pull-down resistor is used in the switch circuit. This satisfies the requirements that the mode of the controller can be determined by a user input switch, whose output is sensed by an MCU GPIO pin.

## Device 5– Potentiometer

Potentiometer is a variable resistor which allows for the user to control the flow of current through a circuit. The potentiometer will act as a user input to the control system that determines the speed and direction that the wheelchair must be set to. The circuit used to test the functionality of the potentiometer measures the current and voltage through a resistor and LED that is connected to the potentiometer. This confirms its behaviour by measuring the current and voltage through the non-variable resistor when the wiper resistance of the potentiometer is changed.

In the circuit to the right, the potentiometer is set to its highest resistance, and it is seen that all the current drawn from the source flows through R10 and there is no current flowing through the LED. The resistance shown is: (3.3V – 0.0032997V)/3.29904e-8 A = 99.93 M

In the circuit to the right, where the potentiometer is set to its lowest setting, much larger current is allowed to flow through the LED. It is seen that the LED is now on since the current flowing through it is around 10 mA. Additionally, the voltage across the led and R13 is almost the entire voltage supplied by the 3.3V source. (3.3V – 3.28614V)/0.0105673 A = 1.311

In the final implementation a variable resistor that models this behaviour with a resistance of 1 k -100 k will be used.

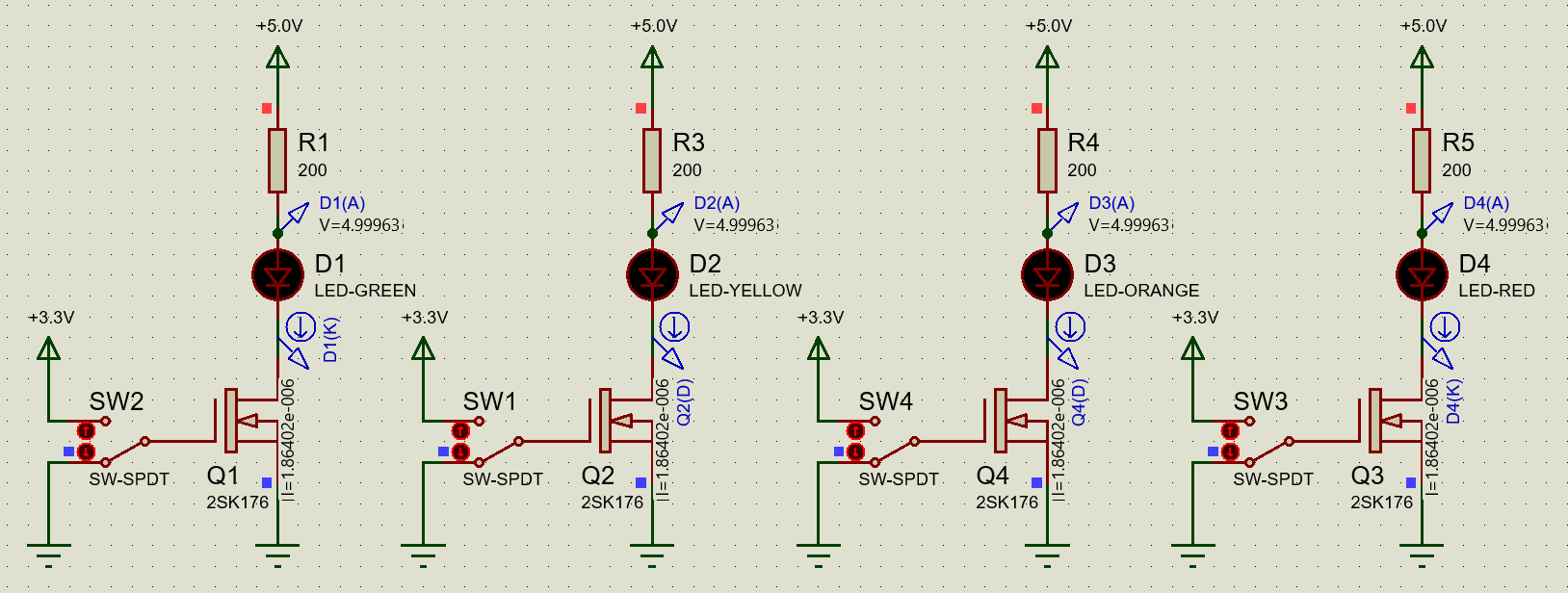
To test this circuit, simply run the schematic in animation mode and control the potentiometer’s resistance via the wiper arrows on the component. It will be shown that the brightness of the LED changes as the amount of current flowing through it changes.

This verifies that the variable resistor changes in resistance when the wiper position is changed, which controls the flow of current through the circuit.

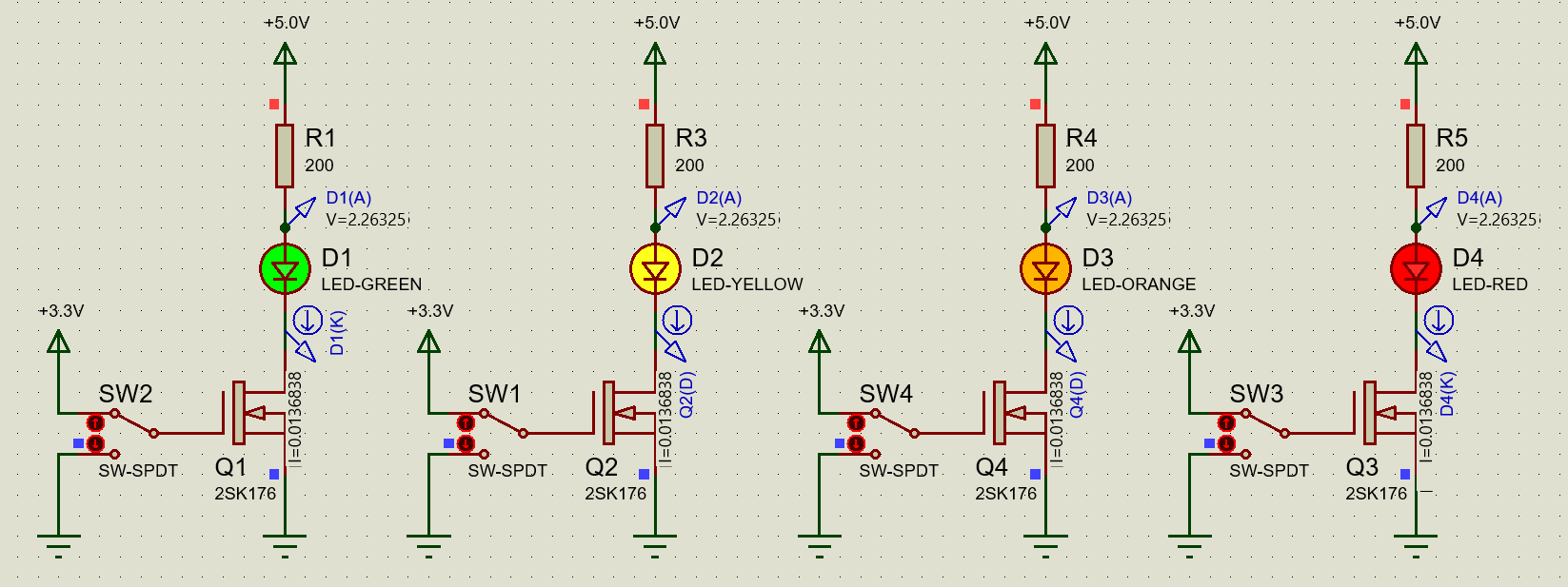
This also validates that a variable resistor can be used to scale an input of 3.3 V to approximately 0-3.3 V which can be sensed by the ADC to sense the speed and direction that the user desires for the motors of the wheelchair.

## Device 6– LED

The LEDs are used as user outputs to convey information about the state of the wheelchair controller and charge of the battery. There is one LED that is either on or off depending on the mode of the wheelchair. It is on (green) when the wheelchair is in the ‘run’ mode. There are also a set of LEDs to convey battery level and they can be green, yellow, orange or flashing red all is decreasing order of battery level left in the controller. The circuit below shows a green, yellow, orange, and red LED connected via a 200 resistor to a 5 V power source, where a transistor is used as a switch controlling whether current flows through it (i.e., it is observed to shine).



The switches connected to the transistor are set to ground, causing the transistor to act like a switch in the open position, not allowing current to flow. If all of the switches are now set to the 3.3 V source:



It is seen that current now flows through each of the LEDs, and they are shining their respective colours. The current through each LED is the same 13.68 mA. This is satisfactory since, for the Proteus models, the forward current of the diodes is 10 mA.

To test these circuit, one may change the state of the two-line switches when the schematic is run in animation mode.

This verifies that the LEDs may be turned on via a gate voltage signalled from the MCU. Additionally, this validates that a gate voltage of 0-3.3 V is enough to bring the current flowing through the LED from 1.86 uA to 13.68 mA, which will control which LEDs are on and off to signal information to the user.

# Part 4 – System-Level Design

