Lab 5: Stepper Motor Control

Micah Galos - SID: 862082339
Joseph Ayala - SID: 862080244

Due: October 17th, 2020 - 11:59PM

Abstract [JDA]

The purpose of this lab is to implement the microcontroller an L298N Motor controller driver to operate a stepper motor in several different modes of operation. These modes of operation are spinning the motor in a clockwise rotation (CW), counter clockwise rotation (CCW), and at different speeds. The L298N motor driver is a H bridge high current motor driver that is able to drive the stepper motor using transistor to transistor logic that is receiving signals from the MK64F microcontroller. This signal is then amplified by the transistors and switches on the respective channel turning on the motor in a particular direction when two channels are selected. From the stepper motor a sequence of four inputs are available for us to connect to the L298N motor driver which will turn on the motor accordingly, as well the enable pins onboard the L298N connected to the respective enabling signal pins on the MK64F Microcontroller.

Onboard the MK64F microcontroller the input signal that will be received will be incoming from the 4 Channel DIP switch that which of the switches are connected to the GPIO pins. This will be influencing the speed and direction of the stepper motor, and as well demonstrate the utility of learning the use of inputting signals from these switches.

Flowchart [MG]

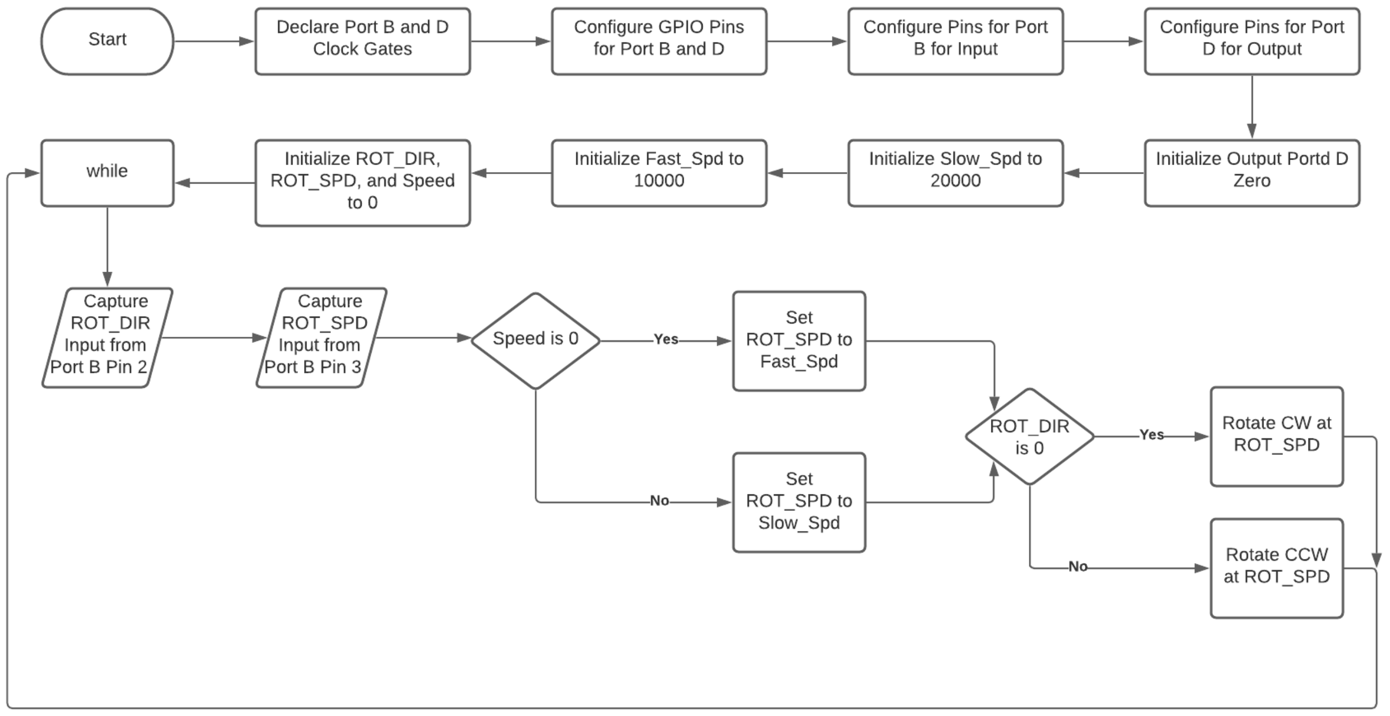
Start
Declare Port B and D Clock Gates
Configure GPIO Pins for Port B and D
Configure Pins for Port B for Input
Configure Pins for Port D for Output
Initialize Output Portd D Zero
Initialize Slow_Spd to 20000
Initialize Fast_Spd to 10000
Initialize ROT_DIR, ROT_SPD, and Speed to 0
while
Capture ROT_DIR Input from Port B Pin 2
Capture ROT_SPD Input from Port B Pin 3
Speed is 0
Yes
No
Set ROT_SPD to Fast_Spd
Set ROT_SPD to Slow_Spd
ROT_DIR is 0
Yes
No
Rotate CW at ROT_SPD
Rotate CCW at ROT_SPD

Figure 1
Flowchart

Experiment System Specification

Schematic Diagrams[JDA]

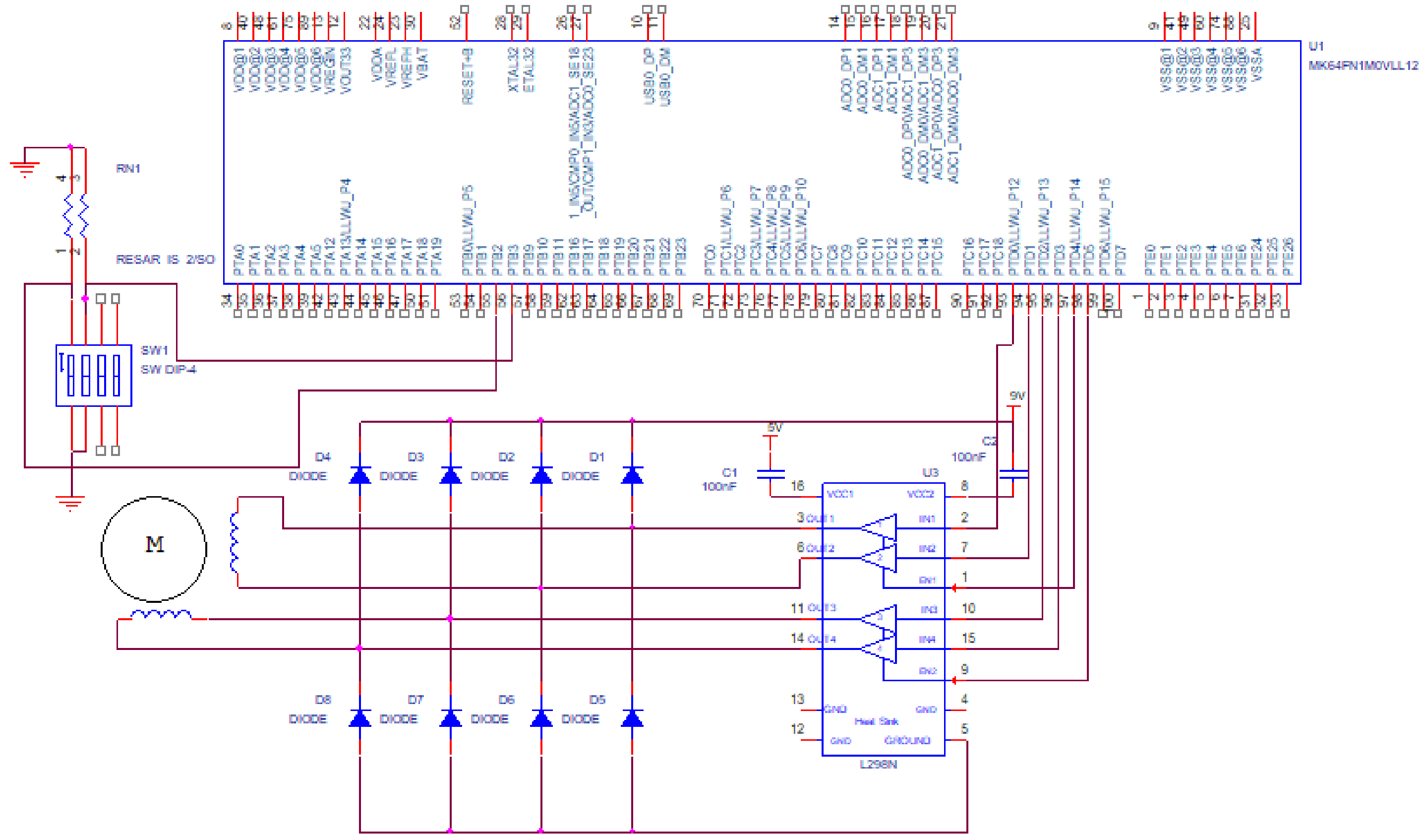

Figure 2
Schematic

Board & Circuit [JDA]

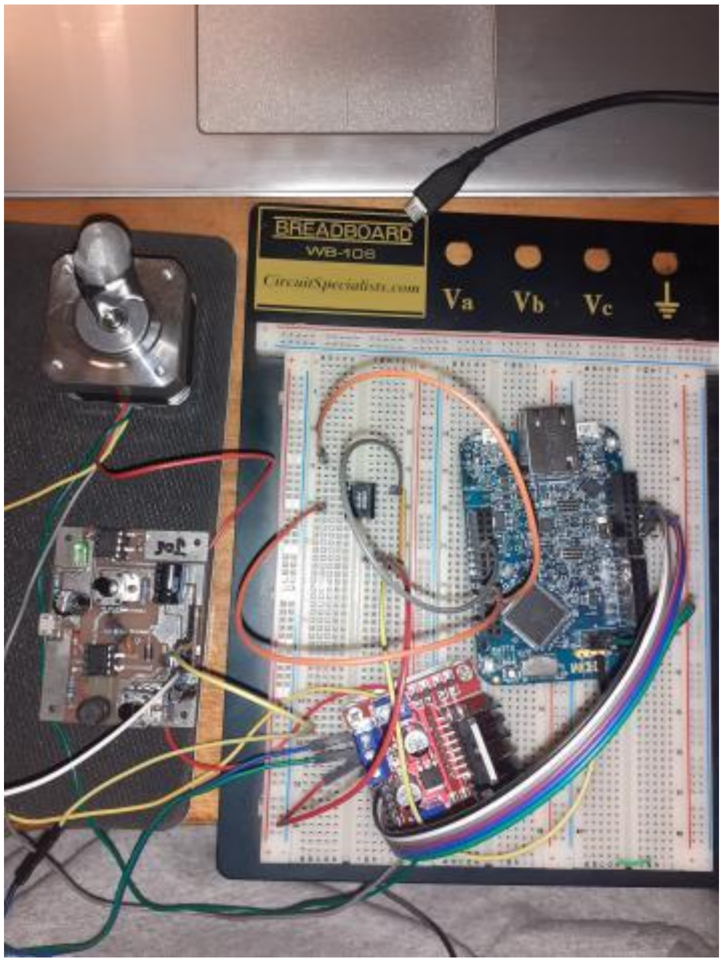

Figure 3
Implementation of Circuit

Software Design [MG]

```
#include "fsl_device_registers.h"

void software_delay(unsigned long delay)
{
    for(int i = 0; i < delay; i++);
    return;
}

int main(void)
{
    // Configure Clock Gating for Port B and D
    SIM_SCGC5 |= SIM_SCGC5_PORTB_MASK;
    SIM_SCGC5 |= SIM_SCGC5_PORTD_MASK;

    // Configure Port D Pins 0-7 for GPIO;
    PORTD_GPCLR = 0x00FF0100;
```

```
// Configure Port B Pin 2 & 3 for GPIO;
PORTB_GPCLR = 0x000C0100;

// Configure Port B Pin 2, 3 for Input;
GPIOB_PDDR |= (0 << 2);
GPIOB_PDDR |= (0 << 3);

// Configure Port D Pins 0-7 for Output;
GPIOD_PDDR |= 0x000000FF;

// Initialize Port D to 0 */
GPIOD_PDOR |= 0x00;

unsigned long Fast_Spd = 10000, Slow_Spd = 20000;
unsigned long Speed = 0x00;
unsigned long ROT_DIR = 0x00;
unsigned long ROT_SPD = 0x00;

while(1) {
    ROT_DIR = (GPIOB_PDIR) & 0x08;
    Speed = (GPIOB_PDIR) & 0x04;

    if(Speed == 0x00) ROT_SPD = Fast_Spd;
    else{
      ROT_SPD = Slow_Spd;
    }

    if(ROT_DIR == 0x00){
        GPIOD_PDOR = 0x36;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x35;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x39;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x3A;
        software_delay(ROT_SPD);
    }
    else{
        GPIOD_PDOR = 0x36;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x3A;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x39;
        software_delay(ROT_SPD);
        GPIOD_PDOR = 0x35;
```

```
                software_delay(ROT_SPD);
            }

        }
        return 0;
}
```

Technical Problems

A problem we came across during the designing process was that the switch would not change the speed or direction of rotating after the first time toggle. The speed and direction would change after the first toggle, but the next input would not register at all.

Questions

1. Can a stepper motor change its speed from zero to a high value instantly? Also, can a stepper motor switch its direction while running at high speed? Answer with a brief explanation.

No, should not be possible because the motors need time for the energy to transfer in order for the motor to start spinning. The process in powering up to start the motor is instant, but the time it requires to generate the energy due to its mechanical parts. A similar idea for the direction switching, the current transfer is instant, but not for the mechanical parts to cause the motor to switch directions.

2. Suppose that there is a 4-phase stepper motor, as shown in the right figure. The rotor magnet is assumed to have 2 poles

a) Write a table of clockwise stepper control steps in "onephase on, full step" mode.

Step	A1	A2	B1	B2	C1	C2	D1	D2
1	**+**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
2	**-**	**-**	**-**	**-**	**-**	**-**	**+**	**-**
3	**-**	**-**	**+**	**-**	**-**	**-**	**-**	**-**
4	**-**	**-**	**-**	**-**	**+**	**-**	**-**	**-**
5	**-**	**+**	**-**	**-**	**-**	**-**	**-**	**-**
6	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**+**
7	**-**	**-**	**-**	**+**	**-**	**-**	**-**	**-**
8	**-**	**-**	**-**	**-**	**+**	**-**	**-**	**-**

b) Write a table of clockwise stepper control steps in "twophase on, full step" mode.

Step	A1	A2	B1	B2	C1	C2	D1	D2
1	+	-	-	-	-	-	+	-
2	-	-	+	-	-	-	+	-
3	-	-	+	-	+	-	-	-
4	-	+	-	-	+	-	-	-
5	-	+	-	-	-	-	-	+
6	-	-	-	+	-	-	-	+
7	-	-	-	+	-	+	-	-
8	+	-	-	-	-	+	-	-

c) Write a table of clockwise stepper control steps in "half-stepping" mode.

Step	A1	A2	B1	B2	C1	C2	D1	D2
1	+	-	-	-	-	-	-	-
2	+	-	-	-	-	-	+	-
3	-	-	-	-	-	-	+	-
4	-	-	+	-	-	-	+	-
5	-	-	+	-	-	-	-	-
6	-	-	+	-	+	-	-	-
7	-	-	-	-	+	-	-	-
8	-	+	-	-	+	-	-	-
9	-	+	-	-	-	-	-	-
10	-	+	-	-	-	-	-	+
11	-	-	-	-	-	-	-	+
12	-	-	-	+	-	-	-	+
13	-	-	-	+	-	-	-	-
14	-	-	-	+	-	+	-	-
15	-	-	-	-	-	+	-	-

16	**+**	**-**	**-**	**-**	**-**	**+**	**-**	**-**

Conclusion [MG]

In conclusion, the lab was a success. The main issue that occurred to us was not in the hardware nor the implementation itself reading the inputs. The problem became of the initial speed values I set for slow and fast mode. At first I set slow and fast to 3000 and 6000, respectively. This was the incorrect approach because the gap is too small for the motor to read the inputs for the mechanical parts to change directions and speed. So, I changed the values for the slow and fast to 10000 and 20000, respectively. After reprogramming the board and launching the software, the implementation is working as intended.