The University of Arizona
Department of Aerospace and Mechanical Engineering
Mechatronics Laboratory
Instructor: Professor. Eniko T. Enikov, enikov@email.arizona.edu

Laboratory Task Sheet 14

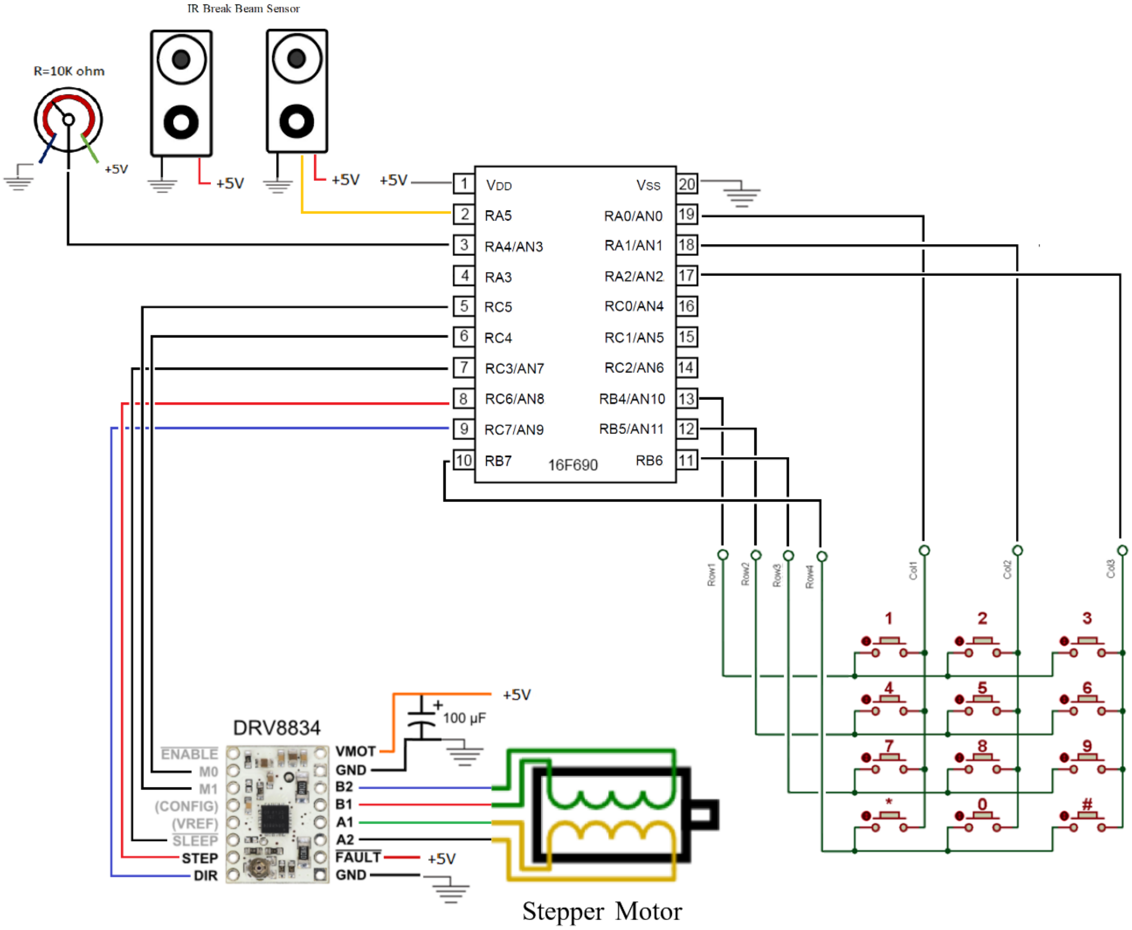
Title: Stepper Motor Position Control

Registers to be learned:

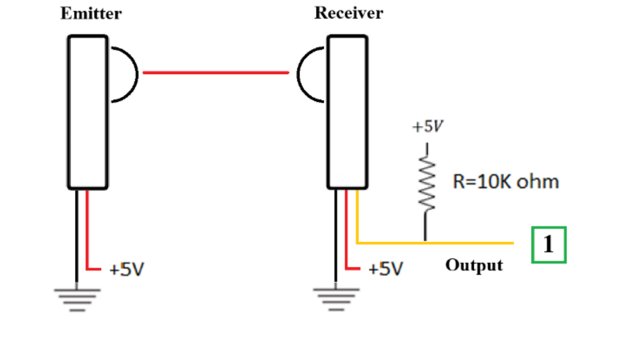
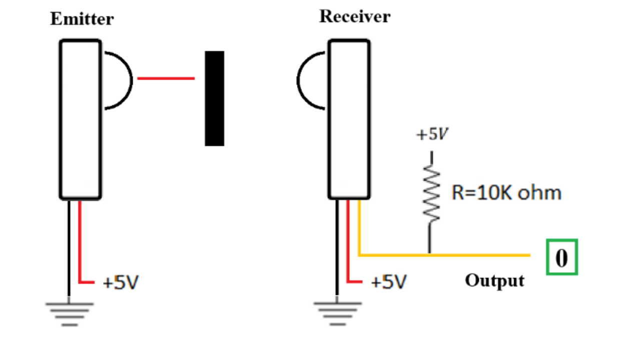
Objective: Program the microcontroller such that for numbers between zero to nine pressed on the Keypad the Clock Hand rotates CW to reach the number on the Clock. Pressing ***** button must reset the Clock, rotating CCW until the Clock Hand reaches the Light Beam Sensor. The potentiometer must be used to control the speed of the rotation.

Tasks

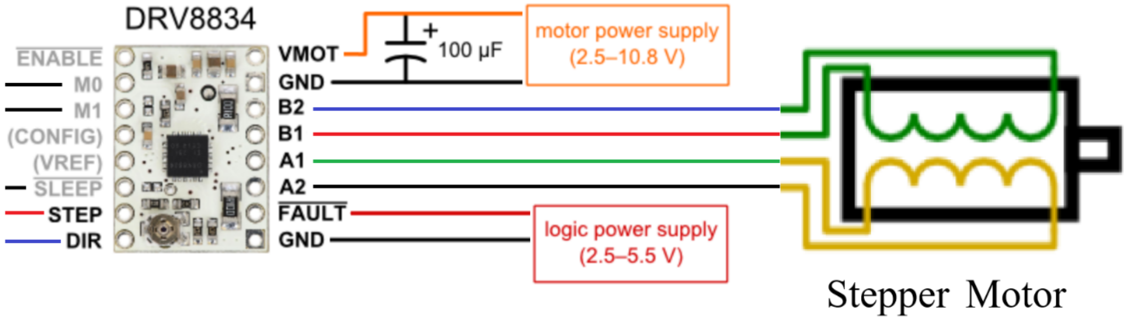
1. Create the circuit below using a Stepper Motor, a Beam Light Sensor, a Low Current Driver, a keypad and a potentiometer.



Infrared (IR) Break Beam Sensors: This sensor has an emitter which sends a beam of human-invisible IR light and a receiver across the way sensitive to this light. The output cord of the receiver must be connected to Pull Up,

IR Beam is not broken	IR Beam is broken
 Emitter Receiver +5V R=10K ohm +5V +5V Output 1	 Emitter Receiver +5V R=10K ohm +5V +5V Output 0

Driver: Make us able to control the stepper motor's power supply, direction of rotation, resolution and number of steps.



Pin	Description	Option	
M0	Micro Step Resolution	-	
M1			
SLEEP	Enable or Disable the Driver	0	Disable Driver
		1	Enable Driver
STEP	Start	0	Do not Take Step
		1	Take One Step
DIR	Direction of Rotation	0	Clock Wise
		1	Counter Clock Wise

M0	M1	Micro Step Resolution
0	0	Full Step
1	0	Half
-	0	Quarter
0	1	1/8
1	1	1/16
-	1	1/32

Matching the Driver and the Stepper Motor: Before connecting the driver to the stepper motor and running it, you need to make sure the voltage and the current that the Driver delivers matches the voltage and the current that the Motor can take. Usually to get the full torque from the motor you need to choose a Deriver with the continuous current ratting higher than the Motor's current and then set the Driver's current to match the motor's current. The information related to the Driver and the Stepper Motor is provided below.

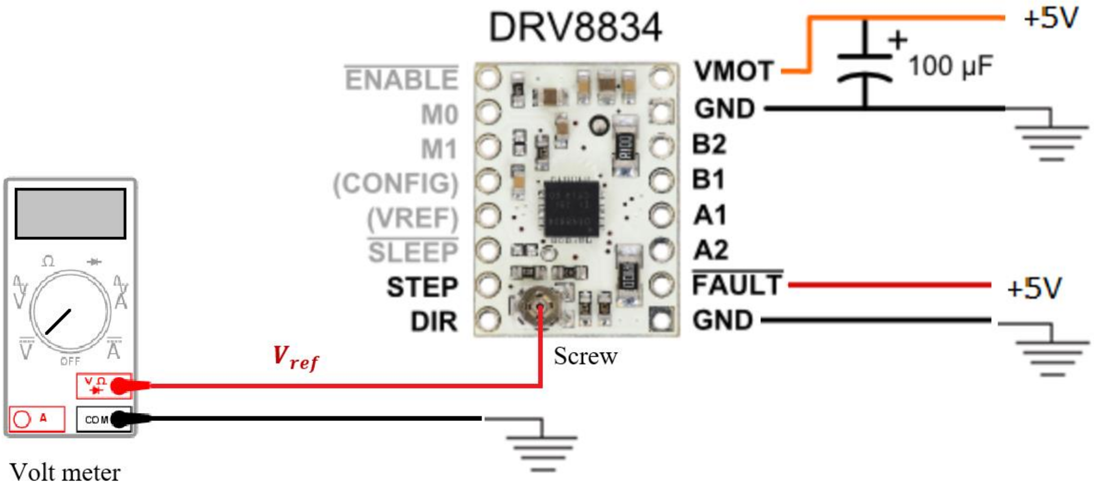
	Driver – DRV8834	**Stepper Motor – SY28STH32-0674A**
Voltage	2.5 – 10.8 V	3.8 V
Current	Approximately Maximum current of 1.5 A	Maximum current of 0.670 A

Matching the Motor's Voltage: To match the Motor's voltage ...

Matching the Motor's Current: It is recommended to use a motor with current less than its maximum current to avoid problems that comes from running a system at its limit. In this work we try to run the motor with 0.3 A. The formula below is provided by the maker to calculate the Deriver's V_{ref} based on Motor's desired current.

$$V_{ref} = 0.5 * (Motor's \text{ current}) = 0.5 * 0.3 \ = 0.15 \text{ V}$$

Therefore, by turning the screw on the driver and tuning the Drive's V_{ref} equal to 0.15 V, we can adjust the Driver' current equal to 0.3 A.



- The DRV8834 Driver **must not** be connected to the microcontroller during the current matching.

2. Make a copy of the P16f690_Template file and name it TASK14Group00. Open the file in MPLAB Software and use the table below to construct the code.

Suggested Code Structure

```
Define Save, Digit, SaveDigit, Speed, and all the necessary Memory Bytes

Start

Call Initialization
Go to Main



Main
Call Keypad
Call ReleaseKeypad
Call GetSpeed
Subtract decimal 10 from Digit and put the result in work register
Test if the result is zero
If it is not zero, skip the next line
If it is zero, go to ResetClock
Subtract decimal 11 from Digit and put the result in work register
Test if the result is zero
If it is zero, skip the next line
If it is not zero, add decimal 255 to Digit
Subtract decimal 12 from Digit and put the result in work register
Test if the result is zero
If it is not zero, skip the next line
If it is zero, go to Main
Subtract SaveDigit from Digit and put the result in work register
Test if the result is zero
If it is not zero, skip the next line
If it is zero, go to Main
Test if the result is negative
If it is not negative, skip the next line
If it is negative, add decimal 10 to work register
 (Find number of steps that stepper motor must take to sweep one segment on the clock??)
Multiply Digit by this number and put the result in Run
Call RunMotor
Decrement Run and test if the result is zero
If it is zero, skip next line
If it is not zero go back for three lines
Move Digit to SaveDigit
Go to Main

GetSpeed
Use ADCON0 register to initiate the conversion
Wait until the conversion is done
Use ADRESH Register to get the result of the conversion
Divide this number by decimal 10 and save the result in Speed
Increment Speed
Increment Speed
```

```
Return

ResetClock
Test PORTA5
If it is clear, go to Main
If it is set, go ahead
Set PORTC7
Move decimal 10 to SaveDigit
Test PORTA5
If it is zero, skip the next line
If it is not zero, call RunMotor and go back for two lines
Call RunMotor
Call RunMotor
Clear PORTC7
Go to Main

RunMotor
Set PORTC6
Call Delay
Clear PORTC6
Call Delay
Return

Keypad
Like TASK11

ReleaseKeypad
Like TASK12

Delay
Move Speed to ByteA
Move Decimal 255 to ByteB
Decrement ByteB until ByteB is zero
Decrement ByteA and go back for two lines until ByteA is zero
Return

Subtraction
Like TASK01C

Multiplication
Like TASK01B

Initialization
Bank2
Use ANSEL and ANSELH Registers to define all the ports as digital
Use ANSEL and ANSELH Registers to define PORTA4 as analog
Use WPUB Register to turn on Weak Pull Up for PORTB4-7
Bank1
Use OSCCON Register to set the oscillator on 8 MHz
Use ADCON1 to set the ADC (Analog to Digital Convertor) clock on FOSC/16
Use TRISA Register to define PORTA0-2 as output
Use TRISA Register to define PORTA4-5 as Input
Use TRISB Register to define PORTB4-7 as Input
Use TRISC Register to define PORTC3-7 as output
Use OPTION_REG to turn on the Weak Pull Up main switch
Use WPUA Register to turn on Weak Pull Up for PORTA5
Bank0
Use ADCON0 register to enable ADC
Use ADCON0 register to set PORTA4 as input channel of the convertor
Initialize PORTA0-2
Initialize PORTC3-5 according to the Driver table to Enable the Driver and set it on Full Steps
Clear PORTC7 to make the direction of the rotation CW
Clear Digit
Clear SaveDigit
Return

end
```

3. Program the microcontroller and test it on the circuit.
4. Demonstrate the result to the instructor.
5. Upload the code on D2L and save it for yourself.