LAB: Stepper Motor

Date : 2023.11.08

Author/Partner: Duwon Yang

Github : LINK

Demo Video : LINK

Introduction

In this lab, We create a simple program that uses a FSM to design the algorithm for stepper motor control.We drive a stepper motor with digital output of GIPOs of MCU.

Requirement

Hardware

- MCU
  - NUCLEO-F411RE
- Actuator/Sensor/Others:
  - 3Stepper Motor 28BYJ-48
  - Motor Driver ULN2003
  - breadboard

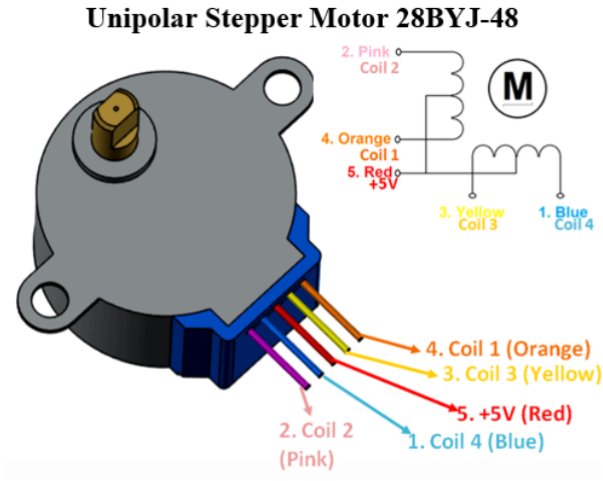
Software

- Keli uVision, CMSIS, EC_HAL library

Problem 1: Stepper Motor

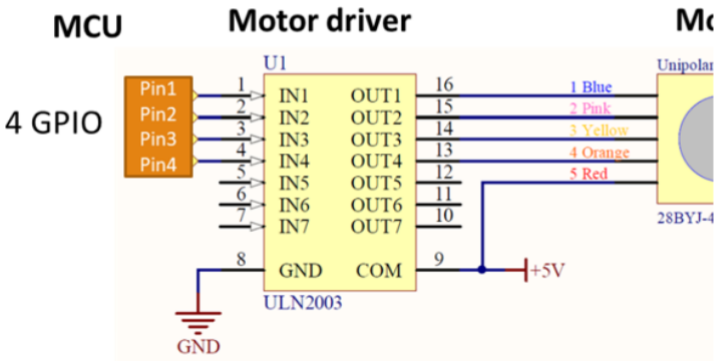
Hardware Connection

There is a specification sheet of the motor and the motor driver for wiring and min/max input voltage/current.



- Rated Voltage: 5V DC
- Number of Phases: 4
- Stride Angle: 5.625°/64
- Gear ratio: 1/32
- Pull in torque: 300 gf.cm
- Coil: Unipolar 5 lead coil

MCU connection wiring

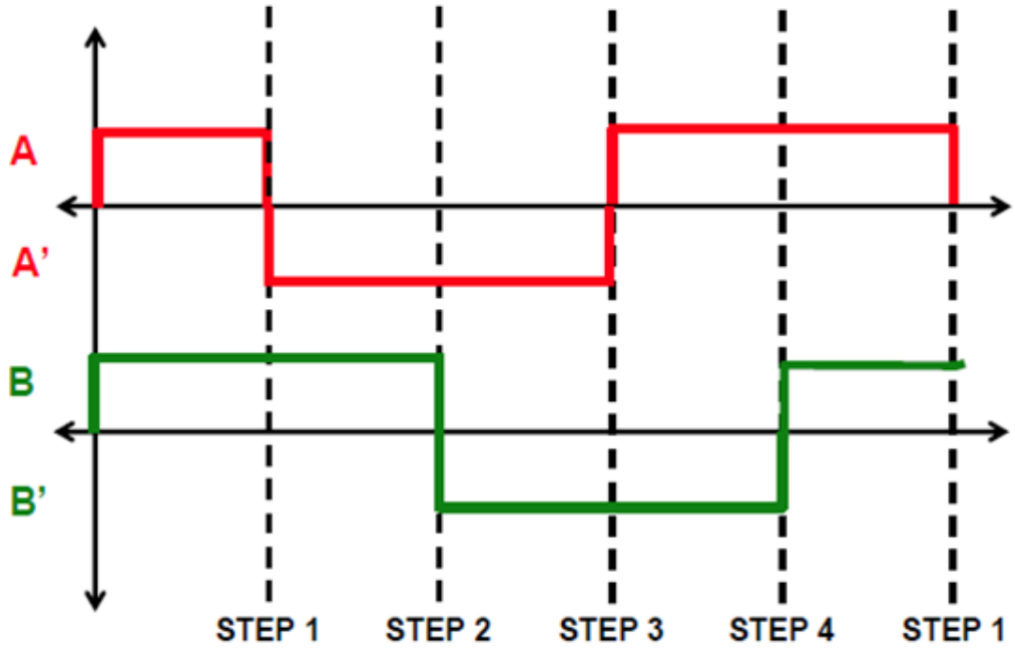


Motor driver (ULN2003)

Stepper Motor Sequence

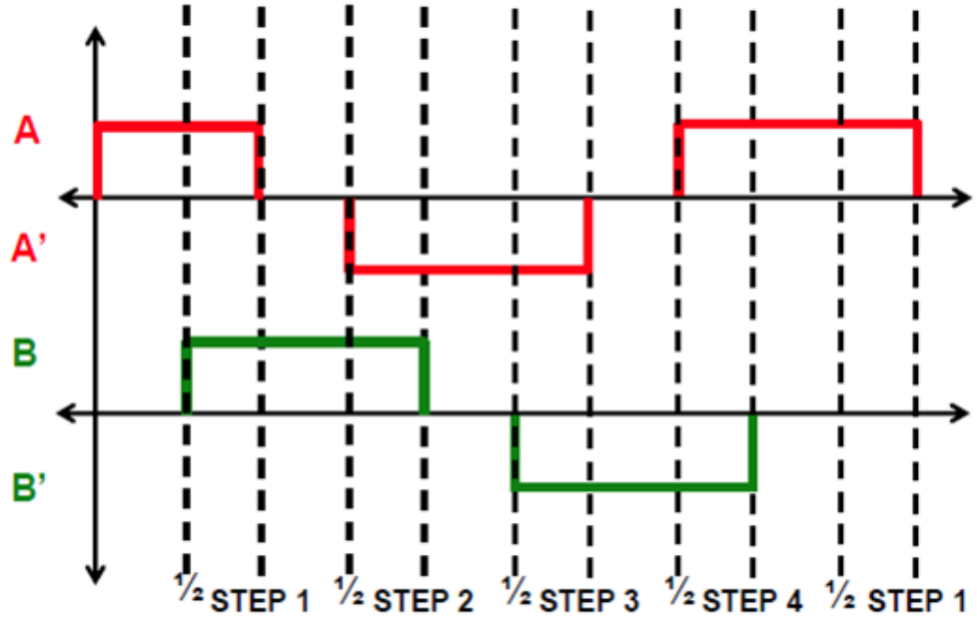
We will use unipolar stepper motor for this lab

Full-stepping sequence



Phase	Port_Pin	Sequence			
		1	2	3	4
A	PB_10	H	L	L	H
B	PB_4	H	H	L	L
A'	PB_5	L	H	H	L
B'	PB_3	L	L	H	H

Half-stepping sequence



Phase	Port_Pin	Sequence							
		1	2	3	4	5	6	7	8
A	PB_10	H	H	L	L	L	L	L	H
B	PB_4	L	H	H	H	L	L	L	L
A'	PB_5	L	L	L	H	H	H	L	L
B'	PB_3	L	L	L	L	L	H	H	H

Finite State Machine

Draw a State Table for Full-Step Sequence.

- Full-Stepping Sequence

State	Next State		Output
	DIR = 0	DIR = 1	(A B A' B')
S0	S1	S3	(H H L L)
S1	S2	S0	(L H H L)
S2	S3	S1	(L L H H)
S3	S0	S2	(H L L H)

- Half-Stepping Sequence

State	Next State		Output
	DIR = 0	DIR = 1	(A B A' B')
S0	S1	S7	(H L L L)
S1	S2	S0	(H H L L)
S2	S3	S1	(L H L L)
S3	S4	S2	(L H H L)
S4	S5	S3	(L L H L)
S5	S6	S4	(L L H H)
S6	S7	S5	(L L L H)
S7	S0	S6	(H L L H)

Problem2: Firmware Programming

Create HAL library

Declare and define the following functions in your library

- **ecStepper.h, ecStepper.c**

You must update your header files located in the directory `EC \lib\`

ecStepper.h

```
// Initialize with 4 pins
// ( A, B,  AN,  BN)
void Stepper_init(GPIO_TypeDef* port1, int pin1, GPIO_TypeDef* port2, int pin2,
GPIO_TypeDef* port3, int pin3, GPIO_TypeDef* port4, int pin4);

//or   using ecPinNames.h
void Stepper_init2(PinName_t A, PinName_t B,  PinName_t AN, PinName_t BN);


// whatSpeed [rev/min]
void Stepper_setSpeed(long whatSpeed);

// Run for n Steps
void Stepper_step(uint32_t steps, uint32_t direction, uint32_t mode);

// Immediate Stop.
void Stepper_stop(void);
```

Procedure

1. Create a new project under the directory `\repos\EC\LAB\LAB_Stepper_Motor`

- The project name is **"LAB_Stepper_Motor".**
- Create a new source file named as **"LAB_Stepper_Motor.c"**

2. Include updated library in `\repos\EC\lib\` to your project.

- **ecGPIO.h, ecGPIO.c**
- **ecRCC.h, ecRCC.c**
- **ecEXTI.h, ecEXTI.c**
- **ecSysTick.h, ecSysTick.c**
- **ecStepper.h, ecStepper.h**

3. Connect the MCU to the motor driver and the stepper motor.

4. Find out the number of steps required to ratate 1 revolution using Full-stepping

$$\frac{steps}{rev} = 64 \times 32 = 2048[step]$$

5. Then, rotate the stepper motor 10 revolutions with 2 rpm. Measure if the motor rotates one revoultion per second.

6. Repeat the above process in the opposite direction.

The code has been modified by changing the line from "while(1){Stepper_step(2048, 1, FULL);}" to "while(1){Stepper_step(2048, 0, FULL);}", thereby setting the direction in the opposite direction.

7. Increase and decrease the speed of the motor as fast as it can rotate to find the maximum and minimum speed of the motor.

When the speed of the stepping motor was increased by 1 unit at a time, it operated smoothly up to a maximum speed of 14 rpm. However, it did not rotate when the speed was set to 15 rpm or higher. On the other hand, when the speed was decreased by 0.1 rpm increments, it operated smoothly down to a minimum speed of 1 rpm.

8. Apply the half-stepping and repeat the above.

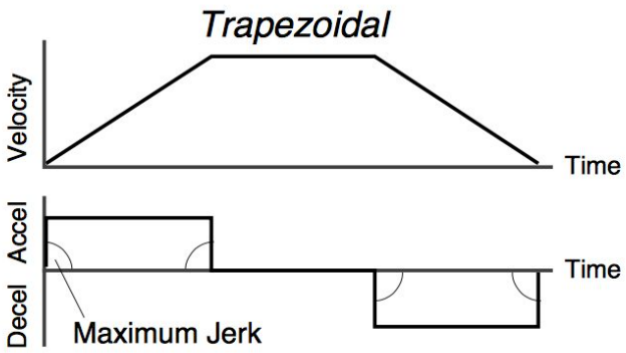
The code has been modified by changing the line from "while(1){Stepper_step(2048, 1, FULL);}" to "while(1){Stepper_step(2048, 1, HALF);}", thereby setting half-stepping mode.

Configuration

Digital Out	SysTick
PB10, PB4, PB5, PB3 NO Pull-up Pull-down Push-Pull Fast	delay()

Discussion

1. Find out the trapezoid-shape velocity profile for a stepper motor. When is this profile necessary?



A stepper motor rotates by a fixed step angle, and in order to smoothly control its movement and achieve precise positioning, it is necessary to implement a gradual acceleration and deceleration to reach a given speed. This is why a smooth acceleration and deceleration are required for a stepper motor.

2. How would you change the code more efficiently for micro-stepping control? You don't have to code this but need to explain your strategy.

The micro step size determines the resolution of the motor. Choosing a smaller micro step wsize can provide more detailed movements, but may reduce motor torque and reduce accuracy. Therefore, a balcne must be struck between movement precision and motor performance by choosing an appropriate micro step size.

Code

The code below controls the stepping motor at a speed of 2rpm using the full-stepping method. A funciton is included to stop the step motor using the External Interrupt function as the button is pressed.

```
/*
******************************************************************************
* @author  SSSLAB
* @Mod       2023-11-08 by DuwonYang
* @brief   Embedded Controller:  LAB - Timer Input Capture
*                                        - with Ultrasonic Distance Sensor
*
******************************************************************************
*/

#include "stm32f411xe.h"
#include "math.h"
#include "ecGPIO.h"
#include "ecRCC.h"
#include "ecTIM.h"
#include "ecPWM.h"
#include "ecEXTI.h"
#include "ecUART_simple.h"
#include "ecSysTick.h"
#include "ecStepper.h"

int RPM = 2;

void EXTI15_10_IRQHandler(void);
```

```
void setup(void);

int main(void){

    setup();

    while(1){Stepper_step(2048, 1, FULL);}
}

void setup(void){

    RCC_PLL_init();
    SysTick_init();

    EXTI_init(GPIOC, BUTTON_PIN, FALL,0);
    GPIO_init(PC_13, INPUT);

    Stepper_init2(PB_10, PB_4, PB_5, PB_3);
    Stepper_setSpeed(RPM);
}

void EXTI15_10_IRQHandler(void) {
    if(is_pending_EXTI(BUTTON_PIN)) {
        Stepper_stop();
        clear_pending_EXTI(BUTTON_PIN);
    }
}
```

Results

In this lab, We create a simple program that operates a stepper motor at the desired RPM speed and in the desired direction using two different modes: Full-stepping mode and Half-stepping mode.

demo video : LINK

Reference

https://ykkim.gitbook.io/ec/ec-course/lab/lab-report-template

https://ykkim.gitbook.io/ec/ec-course/lab/lab-stepper-motor

https://www.motioncontroltips.com/what-is-a-motion-profile/