STM32 MOTOR CONTROL

Author: Yukun Lou

Date: Feb. 8th, 2021

Outline

1. Hardware Configuration
2. Current logic
3. Debug
4. MODBUS Protocol

Hardware Configuration

1. Hardware Requirement

1. STM32F103C8 * 1
2. USB-TTL * 1
3. Microstep Driver * 1
4. Stepper Motor * 1
5. Sensor * 2

2. Software Library

1. STM32CUBEMX
2. Keil uVision5
3. HAL Library

3. Port Configuration

Port	Usage	Connection
PA8	TIM1_CH1	DC-
PB13	GPIO_OUTPUT	DIR-
PB14	GPIO_OUTPUT	ENA-
PA5	GPIO_INPUT	BOTTOM SENSOR
PA6	GPIO_INPUT	TOP SENSOR
PA9	USART1_TX	USB-TTL RX
PA10	USART1_RX	USB-TTL TX
3.3V	VCC	SENSOR COM
5V	VCC	DC+, DIR+, ENA+

4. Default Value

BaudRate: 115200

MODBUS Address: 0x01

Current Logic

States:

- INITSTOP (Initial stop position)
- ONWAY (On moving track)
- ATTOP (At the top)
- ATBOTTOM (At the bottom)

Requirements:

- INITIAL (Initial the state)
- MOVEBOTTOM (Move to the bottom)
- MOVETOP (Move to the top)

Peripherals:

- Top Sensor
- Bottom Sensor

Logic

```
if (State == INITSTOP) {
    if (Require == INITIAL || Require == MOVEBOTTOM) {
        Move bottom;
        State = ONWAY;
    } else if (Require == MOVETOP) {
        State = ONWAY;
    }
} else if (State == ONWAY) {
    Keep current move;
    if (top sensor == Detected) {
        State = ATTOP;
    } else if (bottom sensor == Detected) {
        State = ATBOTTOM;
    }
} else if (State == ATBOTTOM) {
    stop;
    direction = UP;
    if (Require == MOVEBOTTOM || Require == INITIAL) {
        send "At bottom";
    } else if (Require == MOVETOP) {
        Move top;
        State = ONWAY;
    }
} else if (State == ATTOP) {
    stop;
    direction = DOWN;
    if (Require == MOVEBOTTOM || Require == INITIAL) {
        Move bottom;
        State = ONWAY;
    } else if (Require == MOVETOP) {
        send "At top";
    }
}
```

Debug

Speed

- bsp_StepMotor.h
    1. comment and uncomment `#define STEPMOTOR_TIM_PRESCALER`
    2. Set `#define MICRO_STEP`
- main.c
    1. Set the parameter of `Spin(int speed)`

Moving time

- Set the parameter of `void delay_with_input(int ms)`

Direction

- If `currdir = DOWN` , then `STEPMOTOR_DIR_REVERSAL();`
- If `currdir = UP`, then `STEPMOTOR_DIR_FORWARD();`

Stop

- Use `STOP_SOON()`
- Use `STEPMOTOR_OUTPUT_DISABLE()`
- Use `HAL_GPIO_WritePin(STEPMOTOR_ENA_PORT,STEPMOTOR_ENA_PIN,GPIO_PIN_RESET)`

Sensor

- Use as GPIO_INPUT and logic in `void delay_with_input(int ms)` in `main.c`

UART

- Use modbus protocol to receive requirement

MODBUS Protocol

1. General Layout

Address	Function	regH	regL	DataH	DataL	CRC1	CRC2
0x01(Default)	0x06(Default)	high bits of reg	low bits of regL	high bits of data	low bits of data	high bits of CRC	low bits of CRC

2. Set BaudRate

Address	Function	regH	regL	DataH	DataL	CRC1	CRC2
0x01	0x06	0x01	0x01	0x00	Option	high bits of CRC	low bits of CRC

Option	BaudRate
0x00	2400
0x01	4800
0x02	9600
0x03	19200
0x04	38400
0x05	57600
0x06	115200 (Default)
0x07	230400
0x08	460800
0x09	921600

3. Set Address

Address	Function	regH	regL	DataH	DataL	CRC1	CRC2
0x01	0x06	0x02	0x02	0x00	new Address	high bits of CRC	low bits of CRC

Address = new Address

4. Set Requirement

Address	Function	regH	regL	DataH	DataL	CRC1	CRC2
0x01	0x06	0x00	0x00	0x00	Option	high bits of CRC	low bits of CRC

Option	Effect
0x00	INITIAL
0x01	MOVEBOTTOM
0x02	MOVETOP