```c
/*
 * FileName: main.c
 * Version: 1
 *
 * Created: 9/14/2022 1:51 PM
 * Author: Ethan Zeronik
 *
 * Operations: test the io config and the debugger for the lab
 *
 * Hardware:
 *   Atmega2560          micro controller
 *   PORTA.0-3           button inputs (pullup resistors on)
 *   PORTA.4-7           led output
 */

/* NOTE: Includes */
#include <avr/io.h>

#define F_CPU 16000000UL
#include <util/delay.h>

#include "Debugger.h"

/* NOTE: Custom Macros */
// TODO: None

/* NOTE: Global Variables */
// TODO: None

/* NOTE: Function prototypes */
// inits IO ports
void IO_init(void);

/* NOTE: Application implementation */
// the main loop of the function, provided to us
int main(void)
{
    initDebug();

    IO_init();

    while(1)
    {
        PORTA = (0x55 & 0xF0) | (PORTA & 0x0F);
    }
}

/* NOTE: Function implementations */
void IO_init(void)
{
    // the top nibble is the motor output while the bottom is button input
    DDRA  = 0xF0;
    // turn on pullup resisitors on the bottom nibble
    PORTA = 0x0F;
}
```

```c
/*
 * FileName: main.c
 * Version: 1
 *
 * Created: 9/14/2022 1:58 PM
 * Author: Ethan Zeronik
 *
 * Operations: rotates motor 2 revolutions in every mode of operation depending on input
 *
 * Hardware:
 *   Atmega2560          micro controller
 *   PORTA.0-3           button inputs (pullup resistors on)
 *   PORTA.4-7           led output
 */

/* NOTE: Includes */
#include <avr/io.h>

#define F_CPU 16000000UL
#include <util/delay.h>

#include "StepperMotor.h"
#include "Debugger.h"

/* NOTE: Custom Macros */
// TODO: None

/* NOTE: Global Variables */
// TODO: None

/* NOTE: Function prototypes */
// inits IO ports
void IO_init(void);

/* NOTE: Application implementation */
// the main loop of the function, provided to us
int main(void)
{
    initDebug();

    IO_init();

    SM_init(&DDRA, &PORTA);

    while(1)
    {
        switch(PINA & (0xf0))
        {
            case 0x10:
            {
                SM_move((StepperMotorRunMode_t)0, 2);
            }
            break;
            case 0x20:
            {
                SM_move((StepperMotorRunMode_t)1, 2);
            }
```

```
58              break;
59              case 0x40:
60              {
61                  SM_move((StepperMotorRunMode_t)2, 2);
62              }
63              break;
64              default:
65                  break;
66          }
67      }
68  }
69
70  /* NOTE: Function implementations */
71  void IO_init(void)
72  {
73      // the bottom nibble is the motor output while the top is button input
74      DDRA  = 0x00;
75      // turn on pullup resisitors on the top nibble
76      PORTA = 0xf0;
77  }
```

```c
/*
 * FileName: main.c
 * Version: 1
 *
 * Created: 9/14/2022 3:00 PM
 * Author: Ethan Zeronik
 *
 * Operations: moves the motor by the given amount in degrees
 *
 * Hardware:
 *   Atmega2560          micro controller
 *   PORTA.0-3           button inputs (pullup resistors on)
 *   PORTA.4-7           led output
 */

/* NOTE: Includes */
#include <avr/io.h>

#define F_CPU 16000000UL
#include <util/delay.h>

#include "StepperMotor.h"
#include "Debugger.h"

/* NOTE: Custom Macros */
#define Angle 180

/* NOTE: Global Variables */
// TODO: None

/* NOTE: Function prototypes */
// inits IO ports
void IO_init(void);

/* NOTE: Application implementation */
// the main loop of the function, provided to us
int main(void)
{
    initDebug();

    IO_init();

    SM_init(&DDRA, &PORTA);

    while(1)
    {
        switch(PINA & (0xf0))
        {
            case 0x10:
            {
                SM_movePosition((StepperMotorRunMode_t)0, Angle);
            }
            break;
            case 0x20:
            {
                SM_movePosition((StepperMotorRunMode_t)1, Angle);
            }
```

```
            break;
            case 0x40:
            {
                SM_movePosition((StepperMotorRunMode_t)2, Angle);
            }
            break;
            default:
                break;
        }
    }
}

/* NOTE: Function implementations */
void IO_init(void)
{
    // the bottom nibble is the motor output while the top is button input
    DDRA  = 0x00;
    // turn on pullup resisitors on the top nibble
    PORTA = 0xf0;
}
```

```c
/*
 * FileName: StepperMotor.h
 * Version: 1
 *
 * Created: 9/14/2022 2:00 PM
 * Author: Ethan Zeronik
 *
 * Operations: header for the stepper motor submobule
 */

#ifndef StepperMotor_h_INCLUDED
#define StepperMotor_h_INCLUDED

#if defined(__cplusplus)
extern "C" {
#endif

#include <stdbool.h>
#include <stdint.h>
#include <stdio.h>

/* NOTE: Custom Types */
// typing for the stepper motor enum
typedef enum StepperMotorRunMode_t
{
    // wave step mode
    Wave = 0,
    // wave step mode
    Full = 1,
    // wave step mode
    Half = 2,
} StepperMotorRunMode_t;

/* NOTE: Function prototypes */
// inits IO for the stepper motor
// takes a pointer to the port to use, assumes botom nibble
void SM_init(volatile uint8_t * pRegister, volatile uint8_t * pPort);

// moves the motor in the given mode to the given distance
// distance is in units of rotation
void SM_move(StepperMotorRunMode_t mode, double distance);

// moves the motor in the given mode to the given position
// distance is in units of degrees
void SM_movePosition(StepperMotorRunMode_t mode, uint16_t distance);

#if defined(__cplusplus)
} /* extern "C" */
#endif

#endif // StepperMotor_h_INCLUDED
```

```c
/*
 * FileName: StepperMotor.c
 * Version: 1
 *
 * Created: 9/14/2022 2:00 PM
 * Author: Ethan Zeronik
 *
 * Operations: run the stepper motor in one of three modes
 */

/* NOTE: Includes */
#include "StepperMotor.h"

// TODO: move this
#define F_CPU 16000000UL
#include <util/delay.h>

/* NOTE: Global Variables */
// implementation of the wave step map
static uint8_t sWaveStepMap[4] = {
    0x01,
    0x02,
    0x04,
    0x08,
};

// implementation of the full step map
static uint8_t sFullStepMap[4] = {
    0x03,
    0x06,
    0x0c,
    0x09,
};

// implementation of the wave step map
static uint8_t sHalfStepMap[8] = {
    0x09,
    0x01,
    0x03,
    0x02,
    0x06,
    0x04,
    0x0c,
    0x08,
};

// instance pointer to the motor port
static volatile uint8_t * sMotorPort;

/* NOTE: Function implementations */
void SM_init(volatile uint8_t * pRegister, volatile uint8_t * pPort)
{
    // configure port register
    *pRegister = (*pRegister & 0xf0) | 0x0f;

    // turn on pullup resisitors on the bottom nibble
    *pPort = 0x00;
```

```c

    // save the port pointer to the static var
    sMotorPort = pPort;
}

void SM_move(StepperMotorRunMode_t mode, double distance)
{
    uint8_t * pArray = NULL;
    uint8_t   size   = 0;
    uint32_t  steps  = 0;

    switch(mode)
    {
        case Wave:
        {
            pArray = sWaveStepMap;
            size   = sizeof(sWaveStepMap) / sizeof(uint8_t);
            steps  = (distance * 2048);
        }
        break;
        case Full:
        {
            pArray = sFullStepMap;
            size   = sizeof(sFullStepMap) / sizeof(uint8_t);
            steps  = (distance * 2048);
        }
        break;
        case Half:
        {
            pArray = sHalfStepMap;
            size   = sizeof(sHalfStepMap) / sizeof(uint8_t);
            steps  = (distance * 4096);
        }
        break;
        default:
            break;
    }

    for(uint32_t i = 0; i < steps; i++)
    {
        *sMotorPort = pArray[i % size];
        _delay_ms(3);
    }

    *sMotorPort = 0x00;
}

void SM_movePosition(StepperMotorRunMode_t mode, uint16_t distance)
{
    SM_move(mode, ((double)distance / 360));
}
```