

```
1  /*
2   * FileName: main.c
3   * Version: 1
4   *
5   * Created: 9/14/2022 1:51 PM
6   * Author: Ethan Zeronik
7   *
8   * Operations: test the io config and the debugger for the lab
9   *
10  * Hardware:
11  *   Atmega2560          micro controller
12  *   PORTA.0-3          button inputs (pullup resistors on)
13  *   PORTA.4-7          led output
14  */
15
16 /* NOTE: Includes */
17 #include <avr/io.h>
18
19 #define F_CPU 16000000UL
20 #include <util/delay.h>
21
22 #include "Debugger.h"
23
24 /* NOTE: Custom Macros */
25 // TODO: None
26
27 /* NOTE: Global Variables */
28 // TODO: None
29
30 /* NOTE: Function prototypes */
31 // inits IO ports
32 void IO_init(void);
33
34 /* NOTE: Application implementation */
35 // the main loop of the function, provided to us
36 int main(void)
37 {
38     initDebug();
39
40     IO_init();
41
42     while(1)
43     {
44         PORTA = (0x55 & 0xF0) | (PORTA & 0x0F);
45     }
46 }
47
48 /* NOTE: Function implementations */
49 void IO_init(void)
50 {
51     // the top nibble is the motor output while the bottom is button input
52     DDRA = 0xF0;
53     // turn on pullup resistors on the bottom nibble
54     PORTA = 0x0F;
55 }
```

```
1  /*
2   * FileName: main.c
3   * Version: 1
4   *
5   * Created: 9/14/2022 1:58 PM
6   * Author: Ethan Zeronik
7   *
8   * Operations: rotates motor 2 revolutions in every mode of operation depending on input
9   *
10  * Hardware:
11   *   Atmega2560          micro controller
12   *   PORTA.0-3          button inputs (pullup resistors on)
13   *   PORTA.4-7          led output
14  */
15
16  /* NOTE: Includes */
17  #include <avr/io.h>
18
19  #define F_CPU 16000000UL
20  #include <util/delay.h>
21
22  #include "StepperMotor.h"
23  #include "Debugger.h"
24
25  /* NOTE: Custom Macros */
26  // TODO: None
27
28  /* NOTE: Global Variables */
29  // TODO: None
30
31  /* NOTE: Function prototypes */
32  // inits IO ports
33  void IO_init(void);
34
35  /* NOTE: Application implementation */
36  // the main loop of the function, provided to us
37  int main(void)
38  {
39      initDebug();
40
41      IO_init();
42
43      SM_init(&DDRA, &PORTA);
44
45      while(1)
46      {
47          switch(PINA & (0xf0))
48          {
49              case 0x10:
50              {
51                  SM_move((StepperMotorRunMode_t)0, 2);
52              }
53              break;
54              case 0x20:
55              {
56                  SM_move((StepperMotorRunMode_t)1, 2);
57              }
58          }
59      }
60  }
```

```
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 }
```

```
1  /*
2   * FileName: main.c
3   * Version: 1
4   *
5   * Created: 9/14/2022 3:00 PM
6   * Author: Ethan Zeronik
7   *
8   * Operations: moves the motor by the given amount in degrees
9   *
10  * Hardware:
11   *   Atmega2560          micro controller
12   *   PORTA.0-3          button inputs (pullup resistors on)
13   *   PORTA.4-7          led output
14  */
15
16  /* NOTE: Includes */
17  #include <avr/io.h>
18
19  #define F_CPU 16000000UL
20  #include <util/delay.h>
21
22  #include "StepperMotor.h"
23  #include "Debugger.h"
24
25  /* NOTE: Custom Macros */
26  #define Angle 180
27
28  /* NOTE: Global Variables */
29  // TODO: None
30
31  /* NOTE: Function prototypes */
32  // inits IO ports
33  void IO_init(void);
34
35  /* NOTE: Application implementation */
36  // the main loop of the function, provided to us
37  int main(void)
38  {
39      initDebug();
40
41      IO_init();
42
43      SM_init(&DDRA, &PORTA);
44
45      while(1)
46      {
47          switch(PINA & (0xf0))
48          {
49              case 0x10:
50              {
51                  SM_movePosition((StepperMotorRunMode_t)0, Angle);
52              }
53              break;
54              case 0x20:
55              {
56                  SM_movePosition((StepperMotorRunMode_t)1, Angle);
57              }
58          }
59      }
60  }
```

```
58         break;
59     case 0x40:
60     {
61         SM_movePosition((StepperMotorRunMode_t)2, Angle);
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 }
```

```
1  /*
2   * FileName: StepperMotor.h
3   * Version: 1
4   *
5   * Created: 9/14/2022 2:00 PM
6   * Author: Ethan Zeronik
7   *
8   * Operations: header for the stepper motor submobule
9   */
10
11 #ifndef StepperMotor_h_INCLUDED
12 #define StepperMotor_h_INCLUDED
13
14 #if defined(__cplusplus)
15 extern "C" {
16 #endif
17
18 #include <stdbool.h>
19 #include <stdint.h>
20 #include <stdio.h>
21
22 /* NOTE: Custom Types */
23 // typing for the stepper motor enum
24 typedef enum StepperMotorRunMode_t
25 {
26     // wave step mode
27     Wave = 0,
28     // wave step mode
29     Full = 1,
30     // wave step mode
31     Half = 2,
32 } StepperMotorRunMode_t;
33
34 /* NOTE: Function prototypes */
35 // inits IO for the stepper motor
36 // takes a pointer to the port to use, assumes botom nibble
37 void SM_init(volatile uint8_t * pRegister, volatile uint8_t * pPort);
38
39 // moves the motor in the given mode to the given distance
40 // distance is in units of rotation
41 void SM_move(StepperMotorRunMode_t mode, double distance);
42
43 // moves the motor in the given mode to the given position
44 // distance is in units of degrees
45 void SM_movePosition(StepperMotorRunMode_t mode, uint16_t distance);
46
47 #if defined(__cplusplus)
48 } /* extern "C" */
49 #endif
50
51 #endif // StepperMotor_h_INCLUDED
```

```
1  /*
2  * FileName: StepperMotor.c
3  * Version: 1
4  *
5  * Created: 9/14/2022 2:00 PM
6  * Author: Ethan Zeronik
7  *
8  * Operations: run the stepper motor in one of three modes
9  */
10
11 /* NOTE: Includes */
12 #include "StepperMotor.h"
13
14 // TODO: move this
15 #define F_CPU 16000000UL
16 #include <util/delay.h>
17
18 /* NOTE: Global Variables */
19 // implementation of the wave step map
20 static uint8_t sWaveStepMap[4] = {
21     0x01,
22     0x02,
23     0x04,
24     0x08,
25 };
26
27 // implementation of the full step map
28 static uint8_t sFullStepMap[4] = {
29     0x03,
30     0x06,
31     0x0c,
32     0x09,
33 };
34
35 // implementation of the wave step map
36 static uint8_t sHalfStepMap[8] = {
37     0x09,
38     0x01,
39     0x03,
40     0x02,
41     0x06,
42     0x04,
43     0x0c,
44     0x08,
45 };
46
47 // instance pointer to the motor port
48 static volatile uint8_t * sMotorPort;
49
50 /* NOTE: Function implementations */
51 void SM_init(volatile uint8_t * pRegister, volatile uint8_t * pPort)
52 {
53     // configure port register
54     *pRegister = (*pRegister & 0xf0) | 0x0f;
55
56     // turn on pullup resistors on the bottom nibble
57     *pPort = 0x00;
```

```
58
59 // save the port pointer to the static var
60 sMotorPort = pPort;
61 }
62
63 void SM_move(StepperMotorRunMode_t mode, double distance)
64 {
65     uint8_t * pArray = NULL;
66     uint8_t size = 0;
67     uint32_t steps = 0;
68
69     switch(mode)
70     {
71         case Wave:
72         {
73             pArray = sWaveStepMap;
74             size = sizeof(sWaveStepMap) / sizeof(uint8_t);
75             steps = (distance * 2048);
76         }
77         break;
78         case Full:
79         {
80             pArray = sFullStepMap;
81             size = sizeof(sFullStepMap) / sizeof(uint8_t);
82             steps = (distance * 2048);
83         }
84         break;
85         case Half:
86         {
87             pArray = sHalfStepMap;
88             size = sizeof(sHalfStepMap) / sizeof(uint8_t);
89             steps = (distance * 4096);
90         }
91         break;
92         default:
93             break;
94     }
95
96     for(uint32_t i = 0; i < steps; i++)
97     {
98         *sMotorPort = pArray[i % size];
99         _delay_ms(3);
100     }
101
102     *sMotorPort = 0x00;
103 }
104
105 void SM_movePosition(StepperMotorRunMode_t mode, uint16_t distance)
106 {
107     SM_move(mode, ((double)distance / 360));
108 }
```