```
/*
 * Lab 4-5: Washing Machine
 * stepper_motor.c
 *
 * Created: 2/7/2021
 * Author : Alec Derwent
 *
 * VERSION 1: Lab 3 - Stepper motor (2/7/2021)
 * This is a program designed to drive a stepper motor,
 * receiving instructions from the main.c file and
 * values from the stepper_motor.h file. It contains
 * three functions; Stepper_init() initializes the desired
 * port to be used for output, Stepper_Drive() uses paramenters
 * mode and revolutions to drive the stepper in wave, full, or half mode,
 * and Stepper_Position() which drives the motor using Wave mode to
 * an angle specified in stepper_motor.h.
 *
 * Version 2: Lab 4&5 - Washing Machine (2/14/2021)
 * Added a new function, Washing_Machine(). This function is used to
 * drive the stepper motor as desired for lab 4&5, taking two parameters
 * to set the mode (agitate or spin) and the time measured in seconds.
 * Agitate mode uses the Half step mode, utilizing arrays Half[] and HalfR[]
 * to slowly rotate the motor CW and CCW for one second each. The spin mode
 * uses Wave step mode, utilizing array Wave[], to quickly spin CW for the
 * specified amount of time. The output port used was also changed to PORTC
 * from PORTA, a change detailed in stepper_motor.h. For the sake of testing
 * purposes, the function Stepper_Drive was also updated. This allowed for
 * greater customization of rotation type, but required the creation of
 * two more arrays, WaveR[] and FullR[], to drive those modes in CCW mode.
 *
 * Hardware
 *        Stepper Motor            PORTC.0-3
 *
 */

#include "stepper_motor.h"

// Define previously initialized arrays with correct stepper values.
// Arrays with R are designed to drive the motor CCW.
uint8_t Wave[4] = {0x01, 0x02, 0x04, 0x08};
uint8_t WaveR[4] = {0x08, 0x04, 0x02, 0x01};
uint8_t Full[4] = {0x03, 0x06, 0x0C, 0x09};
uint8_t FullR[4] = {0x09, 0x0C, 0x06, 0x03};
uint8_t Half[8] = {0x09, 0x01, 0x03, 0x02, 0x06, 0x04, 0x0C, 0x08};
uint8_t HalfR[8] = {0x08, 0x0C, 0x04, 0x06, 0x02, 0x03, 0x01, 0x09};

void Stepper_init()
{
    Stepper_Register = 0xFF; //set Stepper_Register (DDRA) to output
    Stepper_Output = 0x00;     //initialize Stepper_Output (PORTA) as off
}
```

```
void Stepper_Drive(char mode, uint8_t revolutions, uint8_t direction,
                                uint16_t duration_ms, uint16_t delay)
{
    uint16_t steps; // Initialize step count variable

    switch (mode) // begin switch statement
    {
        case 'W': // if "mode" is indicating WAVE mode
        // given that there are 2048 steps per revolution, multiply by
        // desired number of revolutions to find total steps
        if (duration_ms == 0)
        {
            steps = 2048 * revolutions;
        }
        else
        {
            steps = (unsigned long) duration_ms/delay;
        }


        // begin for loop for steps
        for (uint16_t i = 0; i < steps; i++)
        {
            // there are four values input per step; begin new for loop
            // inside step loop
            if (direction == CW)
            {
                for (uint16_t j = 0; j < 4; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = Wave[j];
                    // delay by delay
                    _delay_ms(delay);
                }
            }
            else if (direction == CCW)
            {
                for (uint16_t j = 0; j < 4; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = WaveR[j];
                    // delay by delay
                    _delay_ms(delay);
                }
            }
        }

        break;

        case 'F': // if "mode" is indicating FULL mode
        // given that there are 2048 steps per revolution, multiply by
        // desired number of revolutions to find total steps
```

```
        if (duration_ms == 0)
        {
            steps = 2048 * revolutions;
        }
        else
        {
            steps = (unsigned long) duration_ms/delay;
        }


        for (uint16_t i = 0; i < steps; i++)
        {

            // there are four values input per step; begin new for loop
            // inside step loop
            if (direction == CW)
            {
                for (uint16_t j = 0; j < 4; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = Full[j];
                    // delay by 600 ms
                    _delay_ms(delay);
                }
            }
            else if (direction == CCW)
            {
                for (uint16_t j = 0; j < 4; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = FullR[j];
                    // delay by 600 ms
                    _delay_ms(delay);
                }
            }
        }

        break;

        case 'H': // if "mode" is indicating HALF mode
        // given that there are 4096 steps per revolution, multiply by
        // desired number of revolutions to find total steps
        if (duration_ms == 0)
        {
            steps = 4096 * revolutions;
        }
        else
        {
            steps = (unsigned long) duration_ms/delay;
        }

        for (uint16_t i = 0; i < steps; i++)
```

```
        {
            // there are eight values input per step; begin new for loop
            // inside step loop
            if (direction == CW)
            {
                for (uint16_t j = 0; j < 8; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = Half[j];
                    // delay by delay
                    _delay_ms(delay);
                }
            }
            else if (direction == CCW)
            {
                for (uint16_t j = 0; j < 8; j++)
                {
                    // output each part of the Wave[] array for every step
                    Stepper_Output = HalfR[j];
                    // delay by delay
                    _delay_ms(delay);
                }
            }
        }

        break;

    }
}

void Stepper_Position(void)
{
    // initialize steps variable
    uint16_t steps;

    // calculate the number of steps by converting the angle into step
    // units. multiply the angle by # of steps per revolution, then divide
    // by 360
    steps = (Angle * 2048UL) / 360;

    // begin for loop using newly calculated steps variable
    for (uint16_t i = 0; i < steps; i++)
    {
        // there are four values input per step; begin new for loop inside
        // step loop
        for (uint16_t j = 0; j < 4; j++)
        {
            // output each part of the Wave[] array for every step
            Stepper_Output = Wave[j];
            // delay by 600 ms
            //_delay_ms(600);
        }
```

```
    }
}

void Washing_Machine(char mode, uint8_t cycleTime_s)
{
    switch (mode)
    {
        // if agitate mode is selected
        case 'A':
        {
            // start for loop, cycleTime_s/2 as there is 1 second for each CW/CCW
            for (uint16_t i = cycleTime_s/2; i > 0; i--)
            {
                // for 21 total steps (21 steps * 8 array values * 6 ms = 1 s)
                for (uint16_t k = 0; k < 21; k++)
                {
                    // cycle through Half[] array
                    for (uint16_t j = 0; j < 8; j++)
                    {
                        // output each part of the Half[] array for every step
                        Stepper_Output = Half[j];
                        // delay by 6 ms
                        _delay_ms(6);
                    }
                }

                // for 21 total steps (21 steps * 8 array values * 6 ms = 1 s)
                for (uint16_t k = 0; k < 21; k++)
                {
                    // cycle through HalfR[] array
                    for (uint16_t j = 0; j < 8; j++)
                    {
                        // output each part of the HalfR[] array for every step
                        Stepper_Output = HalfR[j];
                        // delay by 6 ms
                        _delay_ms(6);
                    }
                }

            }
            break;
        }

        // if spin mode is selected
        case 'S':
        {
            // for how ever many seconds is desired
            for (int i = cycleTime_s; i > 0; i--)
            {
                // 84 * 4 array values * 3 ms = 1 second,
                for (uint16_t k = 0; k < 84; k++)
                {
```

```
                    // there are four values input per step; begin new for loop inside
                    // step loop
                    for (uint16_t j = 0; j < 4; j++)
                    {
                        // output each part of the Wave[] array for every step
                        Stepper_Output = Wave[j];
                        // delay by 3 ms
                        _delay_ms(3);
                    }
                }
            }
            break;
        }
    }
}
```