```
/*
 * Project name: WASH_MC_SYS_DEV.c
 * File name: control_stg_1.c
 * Created: 9/24/2023 8:14:03 PM
 * Author: Caiden Moreno
 * Overview: This program operates as a watching machine with 4 input switches 3 that
choose one out of
 three temperatures (hot, cold, or warm)and display it on the output LEDs.

 Hardware:
 Micro controller  Atmega2560

 Inputs:
 (Temperature selection switches):
 1. Hot-  SW0  = PINA.0
 2. Warm- SW1  = PINA.1
 3. Cold- SW2  = PINA.2

 Door open Switch   -SW3 = PINA.3
 Start push-button       = PINB.0

 Outputs:
(Motor Control):
IN1 = PINL.0
IN2 = PINL.1
IN3 = PINL.2
IN4 = PINL.3

(LED outputs):
Drain Valve        = PINC.0
Hot Water Valve    = PINC.1
Cold Water Valve   = PINC.2
Wash done LED      = PINC.3
Agitate LED        = PINC.4
SPIN LED           = PINC.5
 ********************************************************************************/
#include <avr/io.h>

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

#include "Debugger.h"
#include "stepper_motor.h"

////////////////////////////// checkoff 1 ////////////////////////////////////////////

void io_init(void);

int main(void)
{
   io_init(); //call initialized io ports function
   initDebug(); //call debug function
   while (1)
   {
   }

}
```

```
void io_init(void) //initialize io ports
{
	//inputs
	DDRA=(0x00);
	PORTA=(0xFF);

	//start push-button
	DDRL=0x00;
	PORTL=0xFF;

	//LED Outputs
	DDRC = 0xFF;
	PORTC = 0x00;
}


////////////////////////////////// checkoff 2 ////////////////////////////////////////

//Define start button to check state of the PB
#define startButton (PINL & 0x01)

//Define switch inputs to check the state of the switches
#define hotSwitch  (PINA & 0x01)  //PINA.0
#define warmSwitch (PINA & 0x02)  //PINA.1
#define coldSwitch (PINA & 0x04)  //PINA.2
#define dooropenSwitch (PINA & 0x08) //PINA.3

//define output port
#define outputPort (PORTC)

//define led outputs to control LEDS
#define done_LED    (PORTC |= 0x01) //PINC.0
#define agitate_LED (PORTC |= 0x02) //PINC.1
#define spin_LED   (PORTC |= 0x04)  //PINC.2

//define led outputs to control valves
#define drainValve_led (PORTC |= 0x10) //PINC.4
#define hotValve_led   (PORTC |= 0x20) //PINC.5
#define coldValve_led  (PORTC |= 0x40) //PINC.6


void io_init(void);

void Temp_select_input(int hotStatus, int warmStatus, int coldStatus);

int main(void)
{
   io_init(); //call initialized io ports function
   initDebug(); //call debug function

   while (1)
   {
	//while the door is open the washing machine will not start
	while(!((startButton == 0x01) && (dooropenSwitch == 0x00)))
   {
	//do nothing
   }
   int hotStatus = hotSwitch;
```

```
  int warmStatus = warmSwitch;
  int coldStatus = coldSwitch;
  //fill cycle
  if (hotStatus || warmStatus || coldStatus)
  {
        Temp_select_input(hotStatus, warmStatus, coldStatus);
  }
 _delay_ms(4000);


  //Wash cycle
  PORTC |= agitate_LED;//turn on agitate LED

 _delay_ms(4000); //motor in agitate mode for 4seconds

_delay_ms(4000);//motor in agitate mode for 4seconds

  PORTC &= ~agitate_LED;//turn off agitate LED

 //Drain cycle
PORTC |= drainValve_led;//turn on drain valve LED

  _delay_ms(4000);

PORTC &= ~drainValve_led;//turn off drain valve LED

  //Fill cycle again
   if (hotStatus || warmStatus || coldStatus)
  {
        Temp_select_input(hotStatus, warmStatus, coldStatus);
  }


 _delay_ms(4000);


  //Rinse cycle
 PORTC |= agitate_LED; //turn on agitate LED

  _delay_ms(12000); //motor in agitate mode for 12 seconds

 PORTC &= ~agitate_LED; //turn off agitate LED

  //Rinse cycle again
 PORTC |= drainValve_led;//turn on drain valve LED
_delay_ms(1000);  //1s delay

_delay_ms(9000); //motor in spin mode for 9 seconds

  PORTC &= ~drainValve_led;//Turn off drain valve LED

  //Done LED on
  PORTC |= done_LED;//turn on done LED
  //Read closed door switch and check if the door is open or not
  while(dooropenSwitch != 0x08)
  {
        //do nothing
  }
```

```
   PORTC &= ~done_LED;

   }

}

void io_init(void) //initialize io ports
{
       //inputs
       DDRA =0x00; // Set the lower four bits to 0 for input
       PORTA = 0xFF;   // Set the lower four bits to 1 to enable pull-up resistors

       //start push-button
       DDRL=0x00;
       PORTL=0xFF; //enable PB pull-up resistor

       //LED Outputs
       DDRC = 0xFF;//Turn LEDS off at initialization
       PORTC = 0x00;
}

void Temp_select_input(int hotStatus, int warmStatus, int coldStatus)
{
       // Determine the temperature selection based on switch states
       int temperatureSelection = 0; // Initialize to cold by default

       if (hotStatus == 1) // Hot switch is active
       {
              temperatureSelection = 2; // Set to hot
       }
       else if (warmStatus == 0x02) // Warm switch is active
       {
              temperatureSelection = 1; // Set to warm (hot and cold valves both active)
       }

       switch (temperatureSelection)
       {
              case 0:
              // Cold temperature selected, you can perform actions accordingly
              PORTC |= coldValve_led;
              _delay_ms(4000);
              PORTC &= ~coldValve_led;
              break;

              case 1:
              // Warm temperature selected (both hot and cold), perform actions
accordingly
              PORTC |= hotValve_led;
              PORTC |= coldValve_led;
              _delay_ms(4000);
              PORTC &= ~coldValve_led;
              PORTC &= ~hotValve_led;
              break;

              case 2:
              // Hot temperature selected, perform actions accordingly
              PORTC |= hotValve_led;
              _delay_ms(4000);
```

```
                PORTC &= ~coldValve_led;
                _delay_ms(4000);
                break;

                default:
                break;
        }
}


////////////////////////////////// checkoff 3 ////////////////////////////////////////

//Define start button to check state of the PB
#define startButton (PINB & 0x01) //PINB.0

//Define switch inputs to check the state of the switches
#define hotSwitch  (PINA & 0x01)  //PINA.0
#define warmSwitch (PINA & 0x02)  //PINA.1
#define coldSwitch (PINA & 0x04)  //PINA.2
#define dooropenSwitch (PINA & 0x08) //PINA.3

//define output port
#define outputPort (PORTC)

//define led outputs to control LEDS
#define done_LED    (PORTC |= 0x01) //PINC.0
#define agitate_LED (PORTC |= 0x02) //PINC.1
#define spin_LED   (PORTC |= 0x04)  //PINC.2

//define led outputs to control valves
#define drainValve_led (PORTC |= 0x10) //PINC.4
#define hotValve_led   (PORTC |= 0x20) //PINC.5
#define coldValve_led  (PORTC |= 0x40) //PINC.6


void io_init(void);

void Temp_select_input(int hotStatus, int warmStatus, int coldStatus); //temperature
selection prototype

int main(void)
{
        io_init(); //call initialized io ports function
        initDebug(); //call debug function

        while (1)
        {
                //while the door is open the washing machine will not start
                while(!((startButton == 0x01) && (dooropenSwitch == 0x00)))
                {
                        //do nothing
                }

                //create variables to hold the value for the switches
                int hotStatus = hotSwitch;
                int warmStatus = warmSwitch;
                int coldStatus = coldSwitch;
```

```
		//Fill cycle
		if (hotStatus || warmStatus || coldStatus)//check if hot warm or cold is
selected
		{
			Temp_select_input(hotStatus, warmStatus, coldStatus); //send value
to function
		}
		_delay_ms(4000); //4s delay
		//Wash cycle
		PORTC |= agitate_LED;//turn on agitate LED
	_delay_ms(1000);
			stepper_movement('A', 4);   // Motor in agitate mode for 4 seconds

		    stepper_movement('A', 4); // Motor in agitate mode for 4 seconds

		PORTC &= ~agitate_LED;//turn off agitate LED

		//Drain cycle
		stepper_movement('O', 0); //turn off stepper motor
		PORTC |= drainValve_led;//turn on drain valve LED

		_delay_ms(4000); //4s delay

		PORTC &= ~drainValve_led;//turn off drain valve LED

		//Fill cycle again
		if (hotStatus || warmStatus || coldStatus) //check if hot, warm, or cold
status was selected
		{
			Temp_select_input(hotStatus, warmStatus, coldStatus); //send value
to function to display LED
		}
			_delay_ms(4000);//4 ms delay

		//Rinse cycle
		PORTC |= agitate_LED; //turn on agitate LED
		_delay_ms(1000); //1 ms delay to observe the LED being lit
		stepper_movement('A', 12); //motor in agitate mode for 12 seconds

		PORTC &= ~agitate_LED; //turn off agitate LED
		stepper_movement('O', 0);//turn off motor function

		//Rinse cycle again
		PORTC |= drainValve_led;//turn on drain valve LED
		_delay_ms(1000);  //1s delay

		stepper_movement('S', 1); //motor in spin mode for 9 seconds

		PORTC &= ~drainValve_led;//Turn off drain valve LED
		stepper_movement('O', 0);//turn off motor function

		//Done LED on
		PORTC |= done_LED;//turn on done LED
		//Read closed door switch and check if the door is open or not
		while(dooropenSwitch != 0x08)
		{
			//do nothing
		}
```

```
                PORTC &= ~done_LED; //turn off LED

        }

}

void io_init(void) //initialize io ports
{
        //inputs
         DDRA &= 0xF0; // Lower four bits as input, upper four bits unchanged
         PORTA |= 0x0F;  // Enable pull-up resistors for PINA.0-3

        // initialize motor controls (IN1-IN4)
        DDRL = 0xFF;
        PORTL = 0x00;


        //start push-button
        DDRB=0x00;
        PORTB=0xFF; //enable PB pull-up resistor

        //LED Outputs
        DDRC = 0xFF;//Set PORTC as LED output
        PORTC = 0x00; //Turn LEDS off at initialization
}

void Temp_select_input(int hotStatus, int warmStatus, int coldStatus)
{
        // Determine the temperature selection based on switch states
        int temperatureSelection = 0; // neither warm or hot is selected

        if (hotStatus == 1) // Hot switch is active
        {
                temperatureSelection = 2; // Set to hot
        }
        else if (warmStatus == 0x02) // Warm switch is active
        {
                temperatureSelection = 1; // Set to warm (hot and cold valves both active)
        }

        switch (temperatureSelection)
        {
                case 0:
                // Cold temperature selected
                PORTC |= coldValve_led; //cold valve LED on
                _delay_ms(4000);
                PORTC &= ~coldValve_led; //cold valve LED off
                break;

                case 1:
                // Warm temperature selected (hot and cold)
                PORTC |= hotValve_led; //turn on hot valve LED
                PORTC |= coldValve_led; //turn on cold valve LED
                _delay_ms(4000);
                PORTC &= ~coldValve_led;//turn off hot valve LED
                PORTC &= ~hotValve_led; //turn off cold valve LED
                break;
```

```
            case 2:
            // Hot temperature selected
            PORTC |= hotValve_led; //turn on hotvalve LED
            _delay_ms(4000);
            PORTC &= ~coldValve_led;//turn off hot valve LED
            break;

            default:
            break;
      }
}


/*****************************************************************************//
 * stepper_motor.c
 *
 * Created: 10/4/2023 3:44:50 PM
 *  Author: Caiden Moreno
 *****************************************************************************//
#include "stepper_motor.h" //includes stepper_motor.h file

void stepper_movement(char mode, uint8_t seconds) {
   uint16_t steps;
   uint16_t half[8] = {9,1,3,2,6,4,0X0C,8}; //clockwise sequence
   uint16_t halfcounter[8] = {8,0x0C,4,6,2,3,1,9}; //counter clockwise sequence
   uint8_t Full[4] = {3,6,0x0C,9};//full step sequence


    switch (mode) {
        case 'A': // Agitate mode
        seconds = seconds/4;//time divided by 4 to create desired time frame to run
agitate mode
            steps=45;//calculated the steps needed for each revolution.
        for(uint16_t i=0; i<seconds; i++){
              for(uint16_t i=0; i<steps; i++)
              {
                    for(uint16_t j=0; j<8; j++)//loop through the array based on
                    // the calculated steps needed for each revolution
                    {
                          stepper_output = half[j]; //uses the half step array
(clockwise)
                          _delay_ms(6); //6ms delay
                    }
              }
              for(uint16_t i=0; i<steps; i++)
              {
                    for(uint16_t j=0; j<8; j++)//loop through the array based on
                    // the calculated steps needed for each revolution
                    {
                          stepper_output = halfcounter[j]; //uses half counter array
counterclockwise
                          _delay_ms(6); //6ms delay
                    }
              }
        }
        break; //break from case
```

```
        case 'S': // Spin mode
            steps = 770*seconds; //calculated steps needed for each revolution
            for(uint16_t i=0; i< steps; i++)
            {
                   for(uint16_t j=0; j<4; j++) //loop through the array based on
                  // the calculated steps needed for each revolution
                   {
                          stepper_output = Full[j]; //outputs full step array to stepper
motor
                          _delay_ms(3); //3 ms delay
                   }
            }
            break; //break from case

        default: // Default mode (OFF)
            stepper_output = 0x00; // turns motor off
            break;
    }
}


/*************************************************************************************//
 * stepper_motor.h
 *
 * Created: 10/4/2023 3:46:18 PM
 *  Author: Caiden moreno
 *************************************************************************************//

//checkoff3

#ifndef STEPPER_MOTOR_H_
#define STEPPER_MOTOR_H_

#define F_CPU 16000000UL //clock frequency (16MHz)

//include files
#include <avr/io.h>
#include <util/delay.h>


//Define function Prototypes
void stepper_movement(char mode, uint8_t seconds); // Function to set the stepper motor's
movement

//define modes
#define stepper_output PORTL
//Global variables

extern uint8_t Full[4]; // Full-step drive clockwise sequence
extern uint8_t half[8]; //half step drive clockwise sequence
extern uint8_t halfcounter[8]; //half counter clockwise sequence


#endif /* STEPPER_MOTOR_H_ */
```