```c
/*
 * main.c
 *
 *  Created on: Mar 21, 2017
 *      Author: mohamed
 */
#include <stdio.h>
#include <stdint.h>
#include <stdbool.h>
#include <avr/io.h>
#include <string.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include "comms.h"
#include "servo.h"
#include "stepper.h"
#include "dcMotor.h"
#include "timer2.h"
#include <string.h>

#ifndef F_CPU                   // if F_CPU was not defined in Project -> Properties
#define F_CPU 1000000UL         // define it now as 1 MHz unsigned long
#endif

//String contains the status of the rover
static char carStatus[10];

//Buffer stores the message sent from the computer to the microcontroller
//through serial communication
static char buffer[50];

//Integer contains the number of characters stored in the variable buffer
static int n;

//Declare the functions that controll the rover. See below for more details
void forwards(void);
void reverse(void);
void left(void);
void right(void);
void stop(void);

///////////////////////////////////////////////DC motors function

//Function to move the rover forwards
void forwards(void){
     //Move motor right forwards
     motorRfwd(0);
     //Move motor left forwards
     motorLfwd(0);
     //Update the variable carStatus
     sprintf(carStatus, "Forward");
}

//Function to move the rover backwards
void reverse(void){
    //Move motor right backwards
    motorRbwd(0);
    //Move motor left forwards
    motorLbwd(0);
    //Update the variable carStatus
    sprintf(carStatus, "Reverse");
}

//Function to turn the rover to the left
void left(void){
    //Move motor right forwards
    motorRfwd(0);
    //Move motor left backwards
    motorLbwd(0);
    //Update the variable carStatus
    sprintf(carStatus, "Right");
}

//Function to turn the rover to the right
void right(void){
    //Move motor right backwards
    motorRbwd(0);
    //Move motor left forwards
    motorLfwd(0);
    //Update the variable carStatus
    sprintf(carStatus, "Left");
}

//Function to stop the rover
void stop(void){
    //stop right motot
    motorL_stop();
    //stop left motot
    motorR_stop();
    //Update the variable carStatus
    sprintf(carStatus, "Stopped");
}



///////////////////////////////////////// Main function

int main(void) {
    //Initializations

    //Initialize timer2
    timer_init();

    //Initialize communication through bluetooth
    uart_init();

    //Initialize DC motors and make sure to stop them
    motor_init();
    stop();

    //Initialize servo
    //Declare the claw servo position, start position is 90 degrees
    int servo1Position = 90;
    //Declare the camera servo position, start postion is the midddle 85 degrees
    int servo2Position = 85;
    //Call servo_init() to initialize servo motors control
    servo_init();
    //Make sure that the servo motors move to the desired start postion
    move_servo1(servo1Position);
    move_servo2(servo2Position);

    // Initialise system status variables
    //String contains the status of the rover
    char roverState[50];
    //String contains the status of the stepper
    char stepperState[50];
    //Store the initial status of the rover "Stopped"
    sprintf(roverState, "Stopped");
    //Store the initial status of the rover "Relaxed"
    sprintf(stepperState, "Relaxed");
    //Notify the user that the system is ready to receive commands
    n = sprintf(buffer, "Ready! \n");
    send_str(buffer);


    //Stay forever inside this while loop
    while (1) {

        //get the control command
        char command = get_char();

        // switch case statement to execute the user command
        switch (command) {

            case 'x':
            //Call hold_stepper() which makes the stepper holding its postion
            hold_stepper();
            //Change the stepper status in stepperState
            sprintf(stepperState, "Holding");
            //Notify the user that the winch is holding
            n = sprintf(buffer, "Winch holding\n");
            send_str(buffer);
            break;

            case 'z':
            //Call relax_stepper() which makes the stepper relaxed
            relax_stepper();
            //Change the stepper status in stepperState
```

```
sprintf(stepperState, "Relaxed");
//Notify the user
n = sprintf(buffer, "Winch released\n");
send_str(buffer);
break;

case 'q':
//move the winch one step up
full_step_forward(25);
//Change the stepper status in stepperState
sprintf(stepperState, "Holding");
//Notify the user that winch is relaxed
n = sprintf(buffer, "Winch moved one step up\n");
send_str(buffer);

break;

case 'a':
//move the winch one step down
full_step_back(25);
//Change the stepper status in stepperState
sprintf(stepperState, "Holding");
//Notify the user that the winch moved up
n = sprintf(buffer, "Winch moved one step down\n");
send_str(buffer);
break;

case 'w':
//ckeck if servo postion is within the range
if (servo1Position < 150) {
    servo1Position += 5;
}
//move servo 1
move_servo1(servo1Position);
//Notify the user about the new postion of the claw
n = sprintf(buffer,"Claw servo moved to position %d*\n",
servo1Position);
send_str(buffer);
break;

case 's':
//ckeck if servo postion is within the range
if (servo1Position > 0) {
    //subtract 5 degrees
    servo1Position -= 5;
}
//move servo 1
move_servo1(servo1Position);
//Notify the user about the new postion of the claw
n = sprintf(buffer,"Claw servo moved to position %d*\n",
servo1Position);
send_str(buffer);
break;


case 'r':
//ckeck if servo postion is within the range
if (servo2Position < 150) {
    //Add 5 degrees
    servo2Position += 5;
}
//move servo 1
move_servo2(servo2Position);
//Notify the user about the new postion of the camera
n = sprintf(buffer,"Camera moved to position %d*\n",
servo2Position);
send_str(buffer);
break;

case 'e':
//ckeck if servo postion is within the range
if (servo2Position > 0) {
    //subtract 5 degrees
    servo2Position -= 5;
}
//move servo 2
move_servo2(servo2Position);
//Notify the user about the new postion of the camera
n = sprintf(buffer,"Camera moved to position %d*\n",
            servo2Position);
            send_str(buffer);
            break;

            case 'd':
            //'d' moves the camera to the start position
            servo2Position = 85;
            //move servo 2
            move_servo2(servo2Position);
            //Notify the user about the new camera postions
            n = sprintf(buffer,"Camera moved to position %d*\n",
            servo2Position);
            send_str(buffer);
            break;

            case 'i':
            //Call forwards() to move the rover forwards
            forwards();
            //change the rover status in roverState
            sprintf(roverState, "moving");
            //Notify the user that the rover is moving forwards
            n = sprintf(buffer,"Car is moving forwards\n");
            send_str(buffer);
            break;

            case 'm':
            //call revese() to move the rover backwards
            reverse();
            //change the rover status in roverState
            sprintf(roverState, "reversing");
            //Notify the user that the rover is moving forwards
            n = sprintf(buffer,"Car is moving backwards\n");
            send_str(buffer);
            break;

            case 'j':
            //call right() to move the rover to the right
            right();
            sprintf(roverState, "turning right");
            //Notify the user that rover is turning left
            n = sprintf(buffer,"Car is moving to the Left\n");
            send_str(buffer);
            break;

            case 'l':
            //call left() to move the rover to the left
            left();
            //change the rover status in roverState
            sprintf(roverState, "turning left");
            //Notify the user that rover is turning right
            n = sprintf(buffer,"Car is moving to the Right\n");
            send_str(buffer);
            break;

            case 'k':
            case ' ':
            //stop the rover
            stop();
            //change the rover status in roverState
            sprintf(roverState, "Stopped");
            //Notify the user that rover stopped
            n = sprintf(buffer,"Car stopped\n");
            send_str(buffer);
            break;

        }

        //Jump one line for better readability
        n = sprintf(buffer,"\n");
        send_str(buffer);

        //Display the status of the system
        n = sprintf(buffer, "Claw position=%d\r\nCamera position=%d\r\nRover:%s\r\nWinch:%s\n", se
rvo1Position, servo2Position, roverState, stepperState);
        send_str(buffer);

        //Jump one line for better readability
        n = sprintf(buffer,"\n");
        send_str(buffer);
```

```
    }
    return (0); // should never get here, this is to prevent a compiler warning
}
```