```
/*
 * File:   main.c
 * Author: Pravin Gurjar
 *
 * Created on April 20, 2023, 7:44 PM
 */

/*
 General Algorithm:
 * Start Conveyor
 * Check for Object
 * If object detected, Stop Conveyor
 *  Sense color
 *  Check Switch State
 *  If object is light (switch is not pressed)
 *      Adjust Flaps
 *      Start Conveyor
 *      Reset Flaps after object goes into bucket
 *      Stop Conveyor
 *  Else, if object is heavy
 *      Start Conveyor
 *      If IR 2 (near jaw) detects object
 *          Rotate servo 120 degrees
 *      Stop Conveyor
 */

#include <xc.h>
#include <pic18f4550.h>

#define object_detector_ir PORTCbits.RC0    // IR Sensors
#define jaw_adjust_ir PORTCbits.RC1
#define s0 PORTDbits.RD0            // Color Sensor
```

```
#define s1 PORTDbits.RD1
#define s2 PORTDbits.RD2
#define s3 PORTDbits.RD3
#define out PORTDbits.RD4
#define limit_switch PORTDbits.RD5
#define conv_pin_1 PORTAbits.RA0    // DC 1
#define conv_pin_2 PORTAbits.RA1
#define red_flap_1 PORTAbits.RA2    // Stepper 1
#define red_flap_2 PORTAbits.RA3
#define red_flap_3 PORTAbits.RA4
#define red_flap_4 PORTAbits.RA5
#define blue_flap_1 PORTBbits.RB0   // Stepper 2
#define blue_flap_2 PORTBbits.RB1
#define blue_flap_3 PORTBbits.RB2
#define blue_flap_4 PORTBbits.RB3
#define jaw_1 PORTBbits.RB4         // Stepper 3
#define jaw_2 PORTBbits.RB5
#define jaw_3 PORTBbits.RB6
#define jaw_4 PORTBbits.RB7
#define led_pin PORTDbits.RD7       // Test LED
#define red_led PORTEbits.RE0       // Red LED
#define blue_led PORTEbits.RE1      // Blue LED
#define _XTAL_FREQ 16000000         // 16 MHz
unsigned int col;

void init(void) {
//    Setting input/output pins
    TRISCbits.TRISC0 = 1;   // IR Sensor as input
    TRISCbits.TRISC1 = 1;   // IR Sensor as input
    TRISDbits.TRISD0 = 0;   // S0 of color sensor as output
    TRISDbits.TRISD1 = 0;   // S1 of color sensor as output
    TRISDbits.TRISD2 = 0;   // S2 of color sensor as output
```

```
    TRISDbits.TRISD3 = 0;   // S3 of color sensor as output
    TRISDbits.TRISD4 = 1;   // OUT of color sensor as input
    TRISDbits.TRISD5 = 1;   // Limit Switch as input
    TRISAbits.TRISA0 = 0;   // Conveyor Motor pin 1 as output
    TRISAbits.TRISA1 = 0;   // Conveyor Motor pin 2 as output
    TRISAbits.TRISA2 = 0;   // Light Red Stepper pin 1 as output
    TRISAbits.TRISA3 = 0;   // Light Red Stepper pin 2 as output
    TRISAbits.TRISA4 = 0;   // Light Red Stepper pin 3 as output
    TRISAbits.TRISA5 = 0;   // Light Red Stepper pin 4 as output
    TRISBbits.TRISB0 = 0;   // Light Blue Stepper pin 1 as output
    TRISBbits.TRISB1 = 0;   // Light Blue Stepper pin 2 as output
    TRISBbits.TRISB2 = 0;   // Light Blue Stepper pin 3 as output
    TRISBbits.TRISB3 = 0;   // Light Blue Stepper pin 4 as output
    TRISBbits.TRISB4 = 0;   // Jaw Stepper pin 1 as output
    TRISBbits.TRISB5 = 0;   // Jaw Stepper pin 2 as output
    TRISBbits.TRISB6 = 0;   // Jaw Stepper pin 3 as output
    TRISBbits.TRISB7 = 0;   // Jaw Stepper pin 4 as output
    TRISDbits.TRISD7 = 0;   // Test LED Pin as output
    TRISEbits.TRISE0 = 0;   // Red LED Pin as output
    TRISEbits.TRISE1 = 0;   // Blue LED Pin as output
//    Set Frequency scaling of TCS 3200 to 20%
    s0 = 1;
    s1 = 0;
//    Checking if setup is done.
    led_pin = 1;
    __delay_ms(3000);
    led_pin = 0;
   }


void toggle_conveyor(unsigned int in_1, unsigned int in_2) {
    conv_pin_1 = (unsigned char)in_1;       // Type casted because
of "Implicit warnings" :(
    conv_pin_2 = (unsigned char)in_2;
```

```
}
unsigned int measure_color(unsigned int s2_val, unsigned int s3_val)
{
//    Set filters to measure specified color
    s2 = (unsigned char)s2_val;
    s3 = (unsigned char)s3_val;
    __delay_ms(500);    // wait for stable output
    col = out;        // Read OUT pin value and store in the global
variable above
    return col;
}
unsigned int get_color() {
    unsigned int red_val, blue_val;
    red_val = measure_color(0,0);   // For red filter s2 = 0, s3 = 0
    __delay_ms(500);
    blue_val = measure_color(0,1);  // For blue filter s2 = 0, s3 =
1
    if(red_val > blue_val){          // If more red value then color
is red
        red_led = 1;                 // Show that red color is
detected
        __delay_ms(2000);
        red_led = 0;
        return 1;
    }
    else if(blue_val > red_val){     // If more blue value then
color is blue
        blue_led = 1;                // Show that blue color is
detected
        __delay_ms(2000);
        blue_led = 0;
        return 2;
    }
    return 0;
}
```

```
unsigned int get_weight() {
    if(limit_switch == 0)           // Means switch is pressed => heavy
        return 1;
    else if(limit_switch == 1)      // Means switch is not pressed => light
        return 2;
    return 0;
}
void adjust_red_flap(unsigned int degrees, unsigned int direction) {
    // step_angle = 5.625 degrees;
    unsigned int no_of_steps = (unsigned int)(degrees / 5.625);
    if(direction == 0) {    // CLOCKWISE
        for(unsigned int i=0; i<no_of_steps; i++) {
            red_flap_1 = 1;
            red_flap_2 = 0;
            red_flap_3 = 0;
            red_flap_4 = 0;
            __delay_ms(2);
            red_flap_1 = 0;
            red_flap_2 = 1;
            red_flap_3 = 0;
            red_flap_4 = 0;
            __delay_ms(2);
            red_flap_1 = 0;
            red_flap_2 = 0;
            red_flap_3 = 1;
            red_flap_4 = 0;
            __delay_ms(2);
            red_flap_1 = 0;
            red_flap_2 = 0;
            red_flap_3 = 0;
            red_flap_4 = 1;
```

```
            __delay_ms(2);
        }
    }
    else if(direction == 1) {
        for(unsigned int i=0; i<no_of_steps; i++) {
            red_flap_1 = 0;
            red_flap_2 = 0;
            red_flap_3 = 0;
            red_flap_4 = 1;
            __delay_ms(2);
            red_flap_1 = 0;
            red_flap_2 = 0;
            red_flap_3 = 1;
            red_flap_4 = 0;
            __delay_ms(2);
            red_flap_1 = 0;
            red_flap_2 = 1;
            red_flap_3 = 0;
            red_flap_4 = 0;
            __delay_ms(2);
            red_flap_1 = 1;
            red_flap_2 = 0;
            red_flap_3 = 0;
            red_flap_4 = 0;
            __delay_ms(2);
        }
    }
}
void adjust_blue_flap(unsigned int degrees, unsigned int direction)
{
    // step_angle = 5.625 degrees
    unsigned int no_of_steps = (unsigned int)(degrees / 5.625);
    if(direction == 0) {    // CLOCKWISE
```

```
    for(unsigned int i=0; i<no_of_steps; i++) {
        blue_flap_1 = 1;
        blue_flap_2 = 0;
        blue_flap_3 = 0;
        blue_flap_4 = 0;
        __delay_ms(2);
        blue_flap_1 = 0;
        blue_flap_2 = 1;
        blue_flap_3 = 0;
        blue_flap_4 = 0;
        __delay_ms(2);
        blue_flap_1 = 0;
        blue_flap_2 = 0;
        blue_flap_3 = 1;
        blue_flap_4 = 0;
        __delay_ms(2);
        blue_flap_1 = 0;
        blue_flap_2 = 0;
        blue_flap_3 = 0;
        blue_flap_4 = 1;
        __delay_ms(2);
    }
}
else if(direction == 1) {
    for(unsigned int i=0; i<no_of_steps; i++) {
        blue_flap_1 = 0;
        blue_flap_2 = 0;
        blue_flap_3 = 0;
        blue_flap_4 = 1;
        __delay_ms(2);
        blue_flap_1 = 0;
        blue_flap_2 = 0;
```

```
            blue_flap_3 = 1;
            blue_flap_4 = 0;
            __delay_ms(2);
            blue_flap_1 = 0;
            blue_flap_2 = 1;
            blue_flap_3 = 0;
            blue_flap_4 = 0;
            __delay_ms(2);
            blue_flap_1 = 1;
            blue_flap_2 = 0;
            blue_flap_3 = 0;
            blue_flap_4 = 0;
            __delay_ms(2);
        }
    }
}
void adjust_jaw(unsigned int degrees, unsigned int direction) {
    // step_angle = 5.625 degrees
    unsigned int no_of_steps = (unsigned int)(degrees / 5.625);
    if(direction == 0) {
        for(unsigned int i=0; i<no_of_steps; i++) {
            jaw_1 = 1;
            jaw_2 = 0;
            jaw_3 = 0;
            jaw_4 = 0;
            __delay_ms(2);
            jaw_1 = 0;
            jaw_2 = 1;
            jaw_3 = 0;
            jaw_4 = 0;
            __delay_ms(2);
            jaw_1 = 0;
```

```
        jaw_2 = 0;
        jaw_3 = 1;
        jaw_4 = 0;
        __delay_ms(2);
        jaw_1 = 0;
        jaw_2 = 0;
        jaw_3 = 0;
        jaw_4 = 1;
        __delay_ms(2);
    }
}
else if(direction == 1) {
    for(unsigned int i=0; i<no_of_steps; i++) {
        jaw_1 = 0;
        jaw_2 = 0;
        jaw_3 = 0;
        jaw_4 = 1;
        __delay_ms(2);
        jaw_1 = 0;
        jaw_2 = 0;
        jaw_3 = 1;
        jaw_4 = 0;
        __delay_ms(2);
        jaw_1 = 0;
        jaw_2 = 1;
        jaw_3 = 0;
        jaw_4 = 0;
        __delay_ms(2);
        jaw_1 = 1;
        jaw_2 = 0;
        jaw_3 = 0;
        jaw_4 = 0;
```

```
            __delay_ms(2);
        }
    }
}
void sort_light_obj(unsigned int col) {
    if(col == 1) {
        adjust_red_flap(45,0);     // Set red flap
        __delay_ms(500);
        toggle_conveyor(1,0);      // Start Conveyor (c.w)
        __delay_ms(3000);           //  Wait till Object gets sorted
        toggle_conveyor(0,0);      // Stop Conveyor
        adjust_red_flap(45,1);     // Reset red flap
        __delay_ms(500);
    }
    else if(col == 2) {
        adjust_blue_flap(45,0);     // Set blue flap
        __delay_ms(500);
        toggle_conveyor(1, 0);      // Start Conveyor (c.w)
        __delay_ms(3000);           //  Wait till Object gets sorted
        toggle_conveyor(0, 0);      // Stop Conveyor
        adjust_blue_flap(45,1);     // Reset blue flap
        __delay_ms(500);
    }
}
void sort_heavy_obj(unsigned int col) {
    toggle_conveyor(1,0);
    if(col == 1) {
        while(jaw_adjust_ir == 0);  // Wait till the object is
detected by 2nd IR sensor
        adjust_jaw(120,0);
        __delay_ms(500);
    }
    else if(col == 2) {
```

```
        while(jaw_adjust_ir == 0);  // Wait till the object is
detected by 2nd IR sensor
        adjust_jaw(120,1);
        __delay_ms(500);
    }
}
void take_action(unsigned int c, unsigned int w) {
    if(c == 1 && w == 1)
        sort_light_obj(c);
    else if(c == 1 && w == 2)
        sort_heavy_obj(c);
    else if(c == 2 && w == 1)
        sort_light_obj(c);
    else if(c == 2 && w == 2)
        sort_heavy_obj(c);
}
void main(void) {
    init();
    unsigned int color, weight;
    while(1) {
        while(object_detector_ir == 0) {
            toggle_conveyor(1, 0);        // Start Conveyor (c.w.)
        }
        led_pin = 1;
        __delay_ms(500);
        led_pin = 0;
        toggle_conveyor(0, 0);      // Stop Conveyor
        color = get_color();      // Read Color
        weight = get_weight();    // Read Weight
        take_action(color, weight); // Sort the object accordingly
    }
    return;
}
```