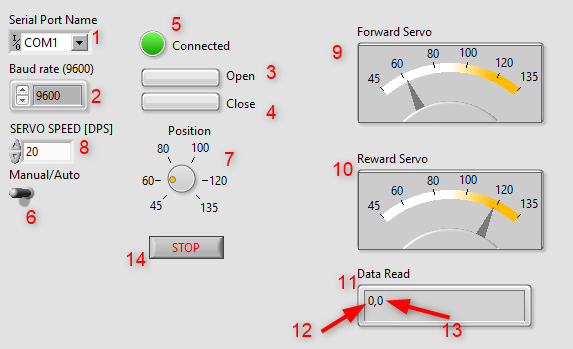
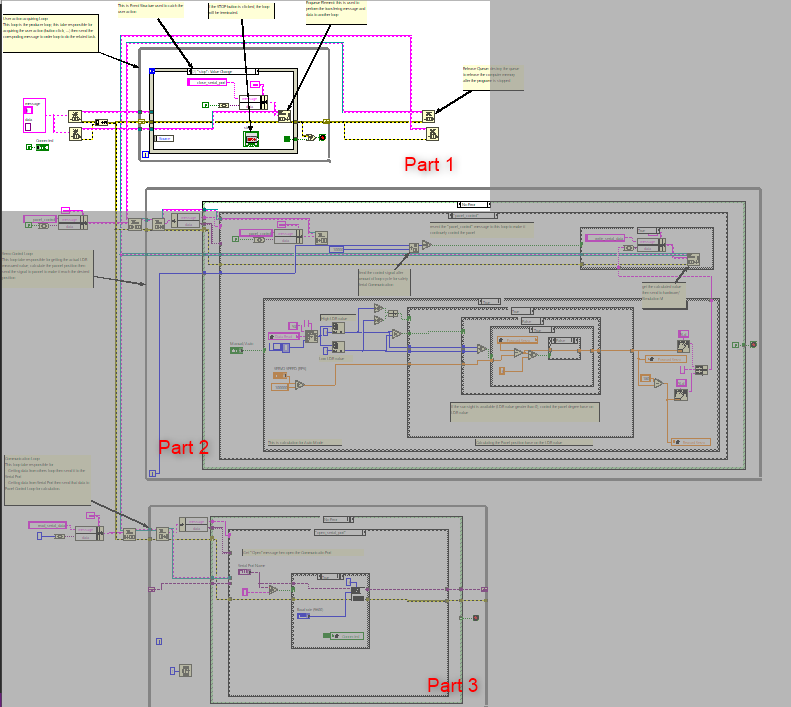
# User Interface for controlling the motor system



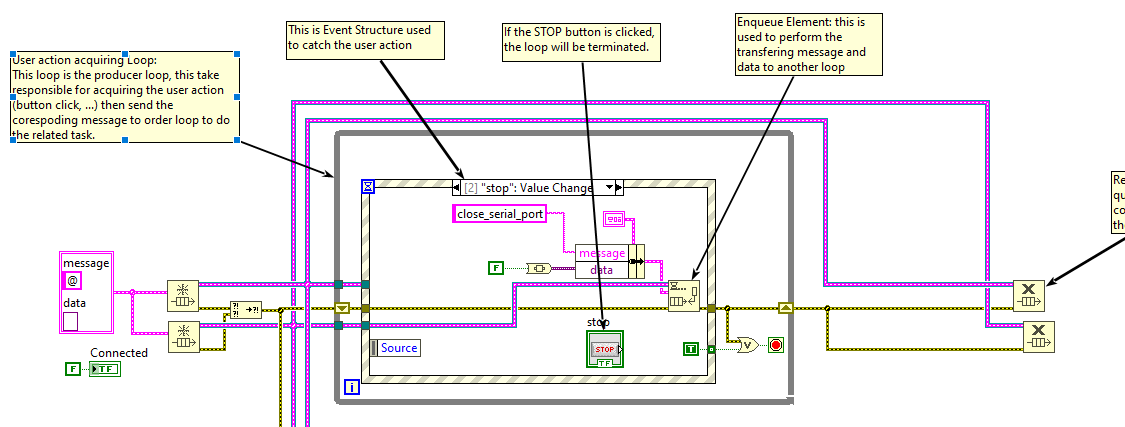
1. Serial Port Name:  
   The name of the Serial Port available for communication (to communicate to the hardware which control the motor or the software which simulate the motor).
2. Baud rate:  
   The baud rate of the Serial port, 9600 is mostly use due to its stable behavior.
3. Open button:  
   Click to open the port and start the communication.
4. Close button :  
   Click to close the port and stop the communication.
5. Connected:  
   The LED to indicate that the communication is ready to be used (send/receive data).
6. Manual/Auto Switch:  
   User can choose operating mode by toggle this switch.  
   In Manual mode, the servo position is get from the Position Knob value.  
   In Auto mode, the servo position is calculated based on the High LDR value and Low LDR value.
7. Position Knob:  
   Control the Servo position in Manual mode.
8. SERVO SPEED:  
   Indicating how fast the servo will turn.
9. Forward Servo:  
   Indicate the forward servo position.
10. Reward Servo:  
    Indicate the reward servo position.
11. Data Read:  
    The data read from Serial Port.
12. The value of High LDR read from Serial Port.
13. The value of Low LDR read from Serial Port.
14. Stop button:  
    Click this button to stop the program.

**Diagram:**



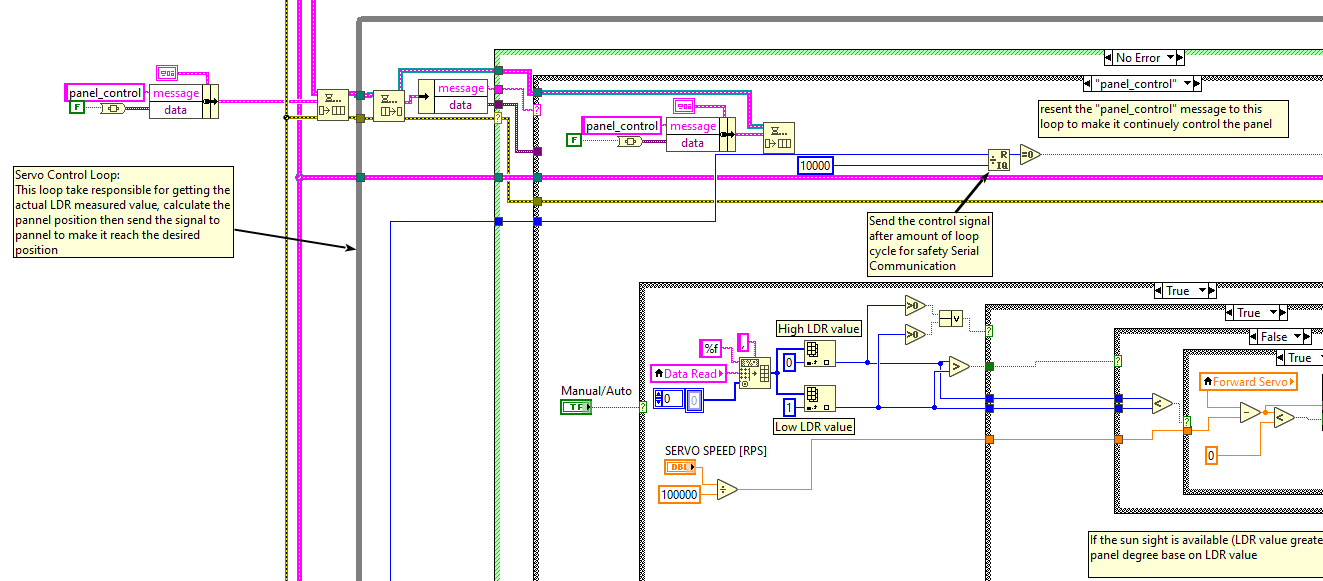
1. **Part 1:**

User action acquiring Loop:  
This loop is the producer loop, this take responsible for acquiring the user action (button click, ...) then send the corresponding message to order loop to do the related task.



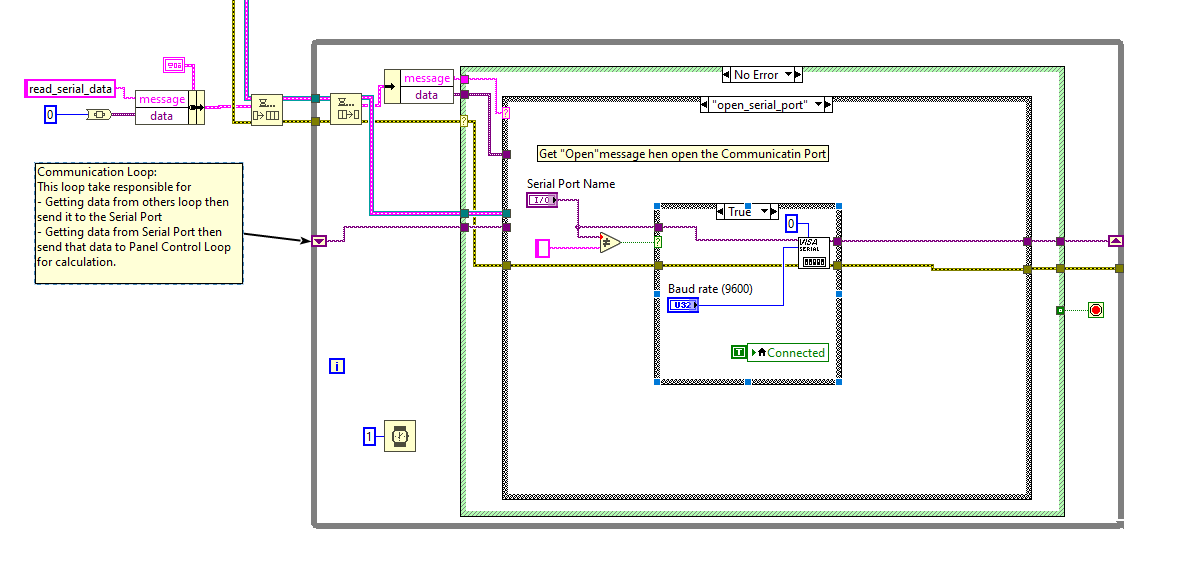
1. **Part 2:**

Servo Control Loop:  
This loop take responsible for getting the actual LDR measured value, calculate the pannel position then send the signal to pannel to make it reach the desired position

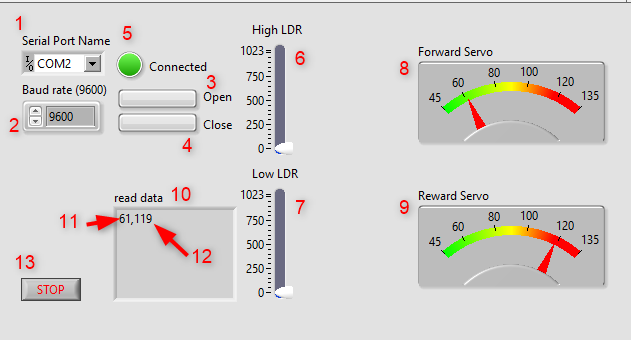


1. **Part 3:**

Communication Loop:  
This loop take responsible for   
- Getting data from others loop then send it to the Serial Port  
- Getting data from Serial Port then send that data to Panel Control Loop for calculation.

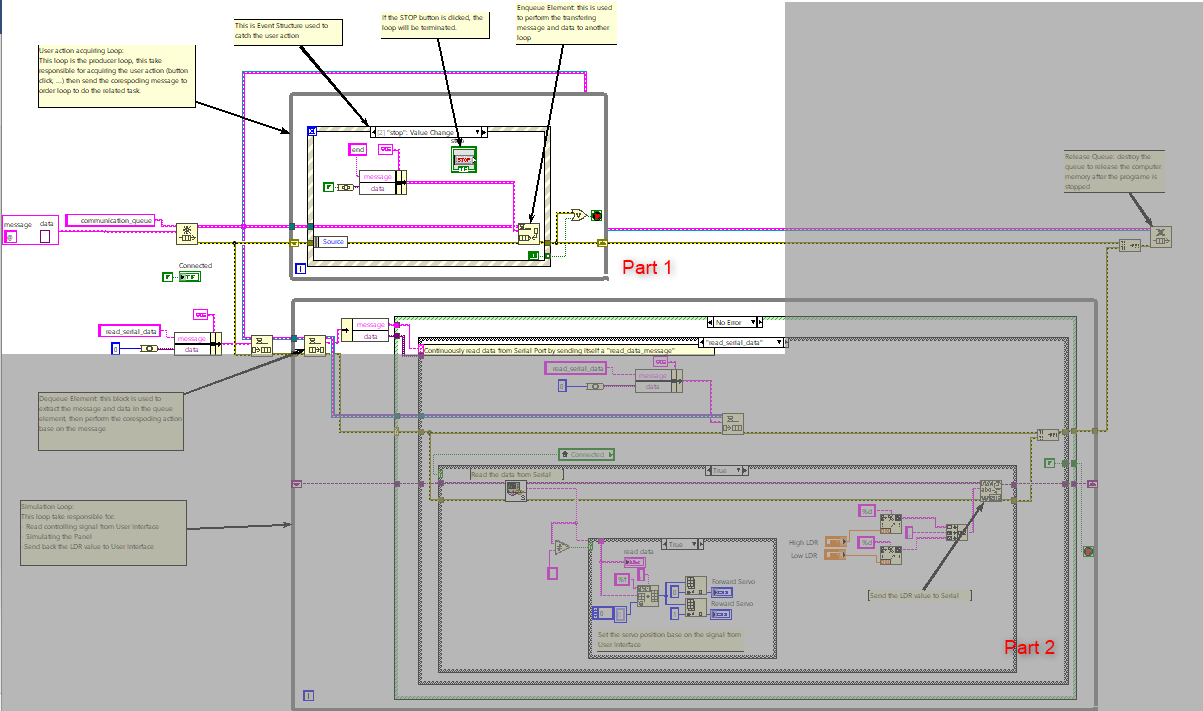


# Panel Simulation:



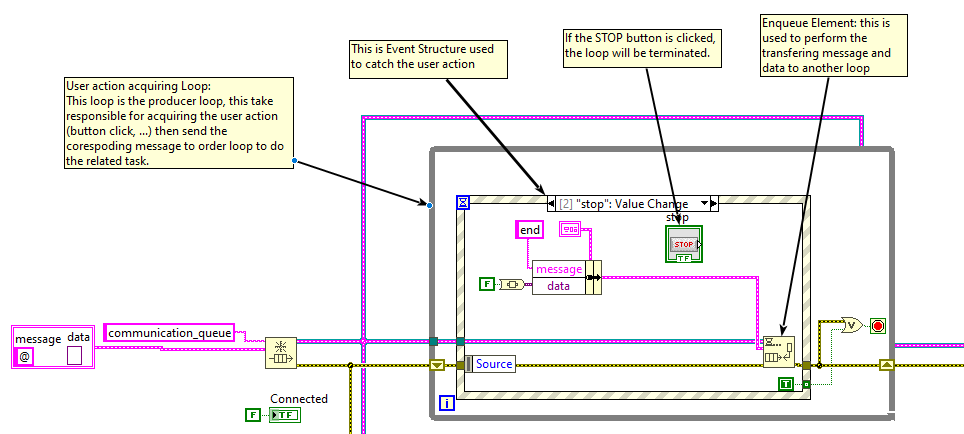
1. Serial Port Name:  
   The name of the Serial Port available for communication (to communicate to the hardware which control the motor or the software which simulate the motor).
2. Baud rate:  
   The baud rate of the Serial port, 9600 is mostly use due to its stable behavior.
3. Open button:  
   Click to open the port and start the communication.
4. Close button :  
   Click to close the port and stop the communication.
5. Connected:  
   The LED to indicate that the communication is ready to be used (send/receive data).
6. High LDR:  
   Simulated High LDR value.
7. Low LDR:  
   Simulated Low LDR value.
8. Forward Servo:  
   Indicate the forward servo position.
9. Reward Servo:  
   Indicate the reward servo position.
10. Read data:  
    Data read from Serial Port.
11. Forward Servo Position .
12. Reward Servo Position .
13. Stop button:  
    Click this button to stop the program.

**Diagram:**



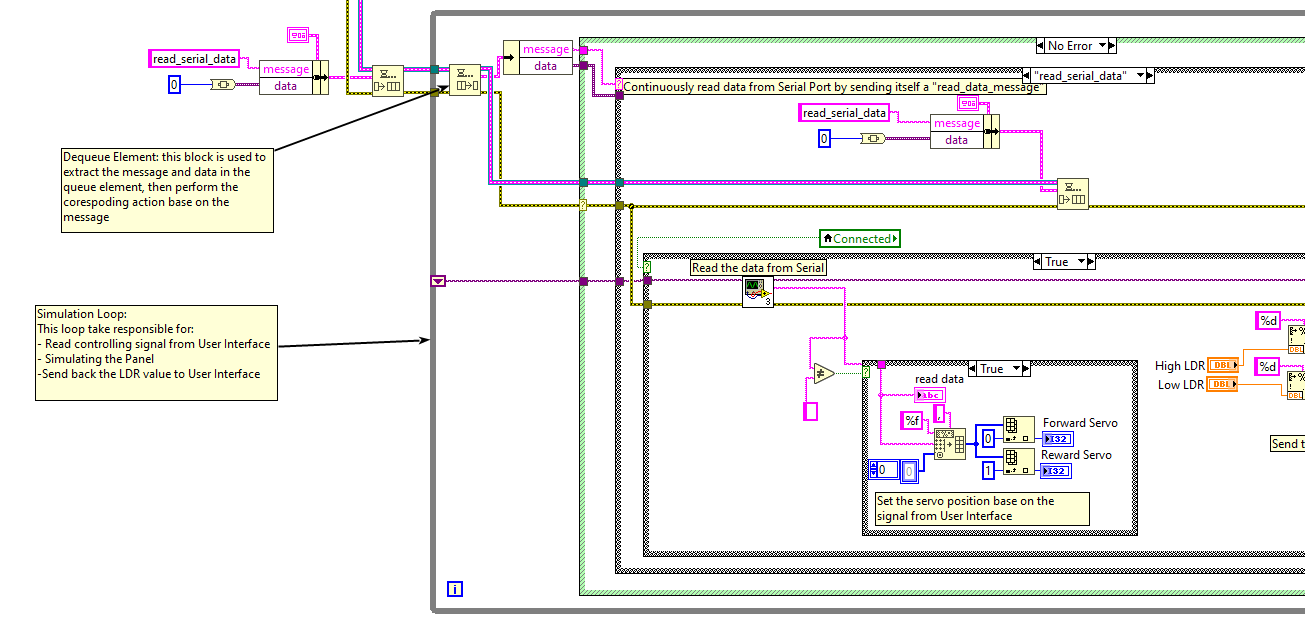
1. **Part 1:**

User action acquiring Loop:  
This loop is the producer loop, this take responsible for acquiring the user action (button click, ...) then send the corespoding message to order loop to do the related task.



1. **Part 2:**

Simulation Loop:  
This loop take responsible for:   
- Read controlling signal from User Interface  
- Simulating the Panel  
-Send back the LDR value to User Interface



**Overview** **Diagram:**

**Prj2\_Main.vi**

Port

Servo Control Loop

Desired Servo Position

**System\_2.vi (Panel simulation)**

Port

Simulation Loop

Panel Simulation

Serial port control

User interface acquiring loop

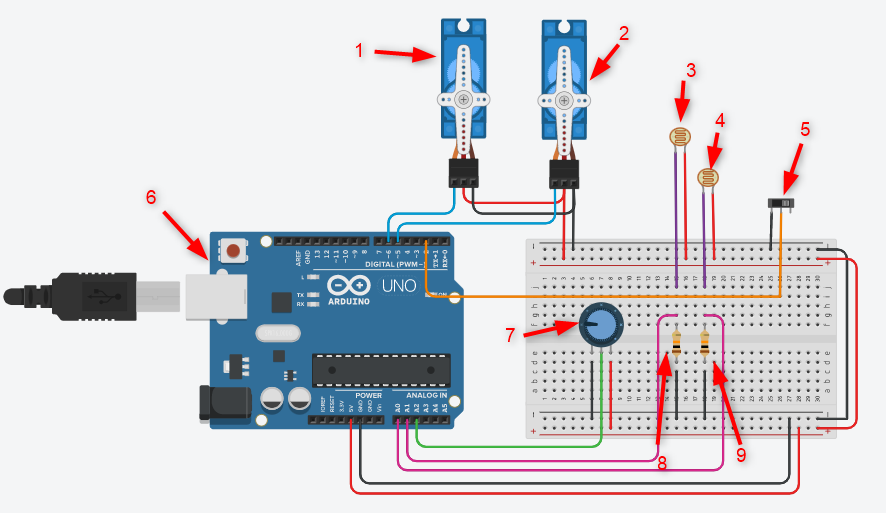
Communication Loop

User interface acquiring loop

Serial Bus

Real servo position

# Tinker CAD Diagram



1. Reward Servo
2. Forward Servo
3. Low LDR
4. High LDR
5. Mode switch to select Auto or Manual mode
6. Arduino Board
7. Potential Meter for control the Servo in Manual mode
8. Resistor support the Low LDR
9. Resistor support the High LDR

# Arduino code

1. **Assign alias to Arduino pin**

// Macro for the Arduiono pin connect to high LDE pin

#define HIGH\_LDR\_PIN A0

// Macro for the Arduiono pin connect to low LDE pin

#define LOW\_LDR\_PIN A1

//  Macro for the the Arduiono pin connect to the potentialmeter which con trol the servo in manual mode

#define POT A2

// Macro for the pin connect to mode change switch

#define AUTO\_SWITCH 2

// Macro for the pin connect to the forward servo signal pin

#define FORWARD\_SERVO\_PIN 5

// Macro for the pin connect to the forward servo signal pin

#define REWARD\_SERVO\_PIN 6

1. **Assign alias to constants**

// Macro define the TRUE value cause tinkerCad does not have TRUE value

#define TRUE HIGH

// Macro define the FALSE value cause tinkerCad does not have FALSE value

#define FALSE LOW

// Macro define servo speed

#define SERVO\_SPEED 5

// Macro define the min position of servo

#define SERVO\_MIN\_POSITION 45

// Macro define the max position of servo

#define SERVO\_MAX\_POSITION 135

// Macro define the position of servo when the sun does not appear

#define SERVO\_NO\_SUN\_POSITION 90

// Macro define the delay time of each program loop

#define RUNNING\_CYCLE 1

// Macro define the value of auto mode

#define AUTO\_MODE TRUE

// Macro define the value of manual mode

#define MANUAL\_MODE FALSE

// Macro define the min value of ldr

#define LDR\_MIN\_VALUE 54

1. **Declare necessary variable**

//  Create servo object

Servo forward\_servo;

Servo reward\_servo;

// Create variable to store the servo osition

long forward\_servo\_position = 90;

long reward\_servo\_position = 90;

1. **Setup function for setting up Arduino pin for read/write signal**

/\*

void setup() funtion

The Arduino void setup and void loop functions are mandatory.

When you run a “standard” C/C++ program, you have to write a “main” function.

This main function will be called first, and from there,

you will call other functions and execute the functionalities of your program.

\*/

void setup(){

  // Attach the Forward servo object to arduino pin

  forward\_servo.attach(FORWARD\_SERVO\_PIN);

  // Attach the Reward servo object to arduino pin

  reward\_servo.attach(REWARD\_SERVO\_PIN);

  // Turn the Forward servo to the position stored in forward\_servo\_position

  forward\_servo.write(forward\_servo\_position);

  // Turn the Forward servo to the position stored in forward\_servo\_position

  reward\_servo.write(reward\_servo\_position);

  // Set the pin connect to the mode control switch as input

  pinMode(AUTO\_SWITCH, INPUT\_PULLUP);

  // Enable the serial communication of the Arduino

  Serial.begin(9600);

}

1. **Loop function for executing the Panel control and print out the system information (Control mode, servo position) to Serial Monitor.**

/\*

In Arduino, there is no main function. This is replaced by setup and loop

The code inside the void loop will be executed again and again (hence the name “loop”), until you:

  - Power off the Arduino board.

  -or Restart the Arduino program by

    pressing the reset button /

    uploading a new sketch /

    re-opening the Serial Monitor on some Arduino boards.

\*/

void loop() {

  // If user select Auto mode

  if (digitalRead(AUTO\_SWITCH) == AUTO\_MODE)

  {

    // read the analog value of High LDR and Low LDR

    int high\_ldr\_value = analogRead(HIGH\_LDR\_PIN);

    int low\_ldr\_value = analogRead(LOW\_LDR\_PIN);

    // When the sunsight is availalbe

    if ((high\_ldr\_value > LDR\_MIN\_VALUE) || (low\_ldr\_value > LDR\_MIN\_VALUE))

    {

      // if High LDR value greater than Low LDR, increase the forward servo degree

      if (high\_ldr\_value > low\_ldr\_value)

      {

        // if the position of the forward servo in movable position, let it move

        if (forward\_servo\_position < SERVO\_MAX\_POSITION)

        {

          // let sero move by increasing position value

          forward\_servo\_position = forward\_servo\_position + SERVO\_SPEED;

        }

      }

      // if High LDR value less than Low LDR, decrease the forward servo degree

      else if (high\_ldr\_value < low\_ldr\_value)

      {

        // if the position of the forward servo in movable position, let it move

        if (forward\_servo\_position > SERVO\_MIN\_POSITION)

        {

          // let sero move by decresing position value

          forward\_servo\_position = forward\_servo\_position - SERVO\_SPEED;

        }

      }

    }

    // When there are no sunsight, both servo return to 90 degree

    else

    {

      // if the servo position is less than center position (SERVO\_NO\_SUN\_POSITION)

      if (forward\_servo\_position < SERVO\_NO\_SUN\_POSITION - SERVO\_SPEED)

      {

        // increasing the servo degree until it reach the SERVO\_NO\_SUN\_POSITION degree

        forward\_servo\_position = forward\_servo\_position + SERVO\_SPEED;

      }

      // if the servo position is greater than center position (SERVO\_NO\_SUN\_POSITION)

      else if (forward\_servo\_position > SERVO\_NO\_SUN\_POSITION + SERVO\_SPEED)

      {

        // decreasing the servo degree until it reach the SERVO\_NO\_SUN\_POSITION degree

        forward\_servo\_position = forward\_servo\_position - SERVO\_SPEED;

      }

    }

    // the reward servo position will in opposite site against the forard servo

    reward\_servo\_position = 180 - forward\_servo\_position;

  }

  // If user select Manual mode

  else if (digitalRead(AUTO\_SWITCH) == MANUAL\_MODE)

  {

    // read the potential meter value then calculate the servo

    float pot\_value = float(analogRead(POT));

    // calculate the servo position base on the value connect to potentialmeter

    // 135 is the max position of the servo

    // 45 is the min position of the servo

    // 1023 is the max input value on the pin connect to the potential meter

    // calculation below mapping the potentialmeter value to the servo position value

    // Ex: if value on potentialmeter is 0, servo position is 45

    // Ex: if value on potentialmeter is 1023, servo position is 135

    forward\_servo\_position = pot\_value\*(135 - 45)/1023 + 45;

    // the reward servo position will in opposite site against the forard servo

    reward\_servo\_position = 180 - forward\_servo\_position;

  }

  // Turn the Forward servo to the position stored in forward\_servo\_position

  forward\_servo.write(forward\_servo\_position);

  // Turn the Reward servo to the position stored in reward\_servo\_position

  reward\_servo.write(reward\_servo\_position);

  // Print out the system information (control mode, servo position)

  if (digitalRead(AUTO\_SWITCH) == AUTO\_MODE)

  {

    // Print to serial moitor to indicate that the program is in AUTO mode

    Serial.print("AUTO");

  }

  else

  {

    // Print to serial moitor to indicate that the program is in AUTO mode

    Serial.print("MANUAL");

  }

  // print the position of the servo out the Serial monitor

  Serial.print('\t');

  Serial.print("forward\_servo: " + String(forward\_servo\_position));

  Serial.print('\t');

  Serial.print("reward\_servo: " + String(reward\_servo\_position));

  Serial.print("\t\r\n");

  // delay a little bit for safety functionality

  delay(RUNNING\_CYCLE);

}