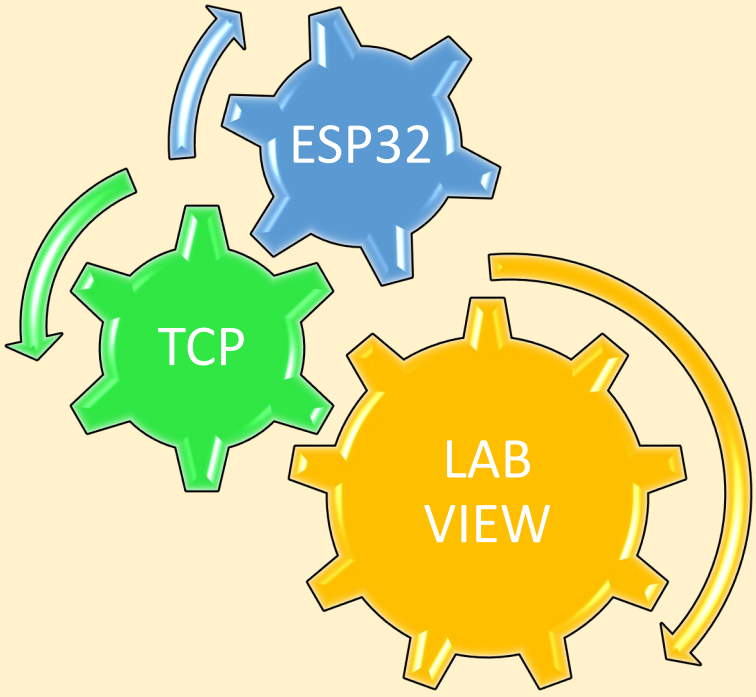
Project name: **INTRUSION DETECTION WITH IR SENSOR**

Task: Connecting ESP32 with Lab VIEW.

Material used:
ESP32, Breadboard, Connecting wires, IR sensor module, Servo motor, 10uF capacitor.

Software used:
Lab VIEW, Arduino IDE.

Theory:
In this project we have used the ESP32 as the brain of our project which controls all the sensor module and the servo motor attached to it. Furthermore, we have used the Arduino IDE as the programming tool to interface and communicate with the ESP32 to carry out needed operations. Then we used Lab VIEW to present controls from the user and connecting the user to the hardware-based IR sensor through a programming environment.

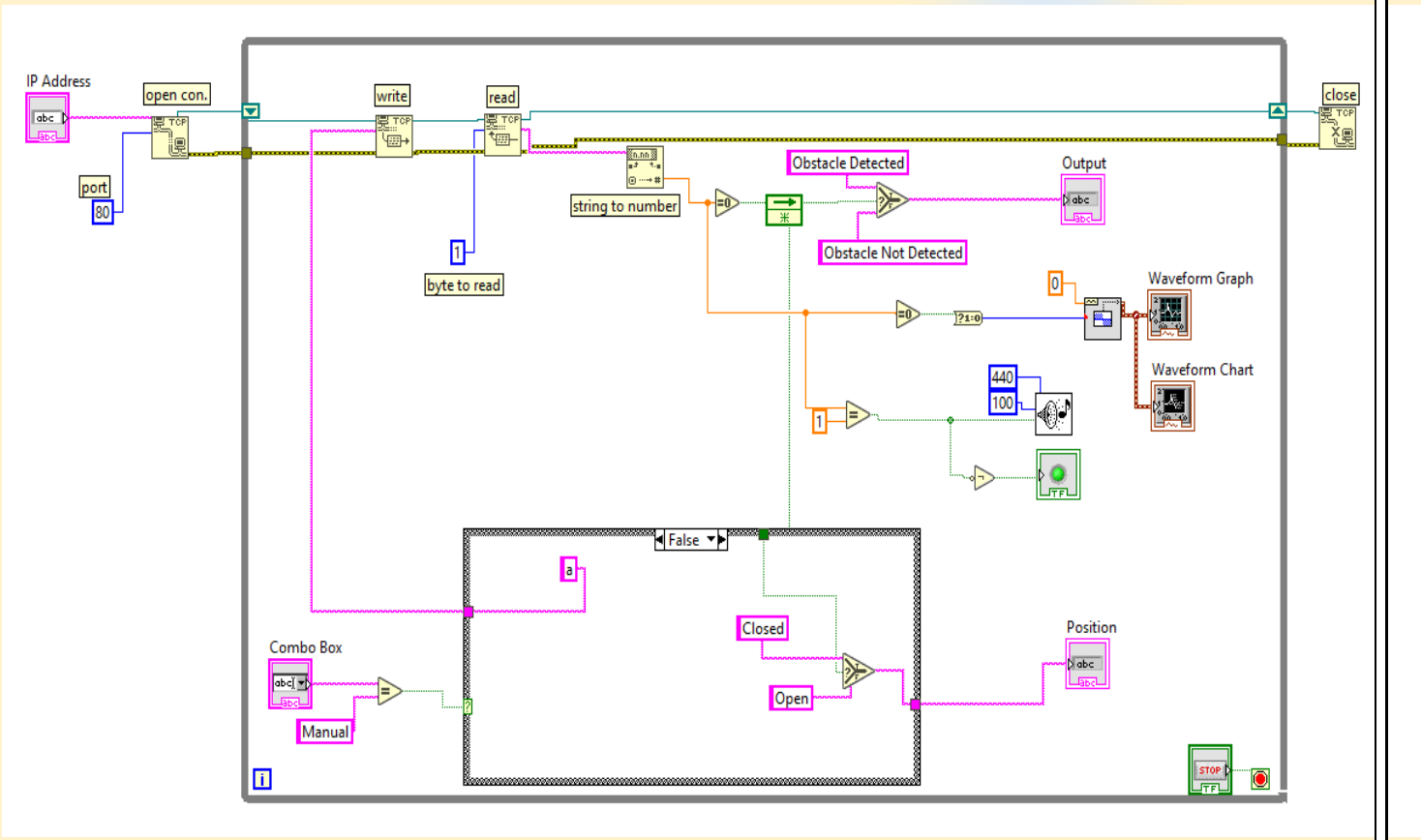


Procedure:

1. Formulated a code on Arduino IDE to make the IR sensor functional and interfacing IR sensor with the servo motor.
2. The code includes the libraries for Wi-Fi connection (Wifi.h) and servo-motor connection (Servo.h).
3. Defined the input and output pins for the sensors and various other connections on the PCB.
4. Established with Wi-Fi connection and setup the sensors using code.
5. To make the servo motor work we made two functions namely servo180 and servo0.
6. Made a function lab which controlled the servo motor according to the IR sensor readings.
7. Then established the connection with the client in the main code and manipulated the commands.
8. Based on the client inputs we used the switch case to formulate the code as per user's choice.
9. Uploaded the code on ESP32 using USB cable after properly making the circuit connections.
10. Now, we created the Block diagram on LabVIEW.
11. Used TCP function blocks from the connectivity panel.
12. Used open connection block to give IP address and port in order to connect ESP32 with lab view.
13. Used write function from TCP block to send data from lab VIEW to ESP32.
14. Used read function from TCP block to get data from the IR sensor attached to ESP32.
15. Used string to number converter to convert the string input from ESP32, if any, to numeric data.
16. Then checked the value if 0, showed "Obstacle detected" and if 1, showed "Obstacle not detected".
17. Used a waveform graph, waveform chart, Boolean LED and beep function to indicate the intrusion.

18. Made two pathways for the data entry i.e., automatically and manually.
19. Used combo box for the user to select the pathway.
20. If the selection is manual, using radio buttons the user can open or close the gate but if selection is automatic then it directly takes the reading from IR sensor.
21. Used the while loop to process all the functions for continuous working. At last closed the connection outside the loop once the process is completed.

Block diagram:

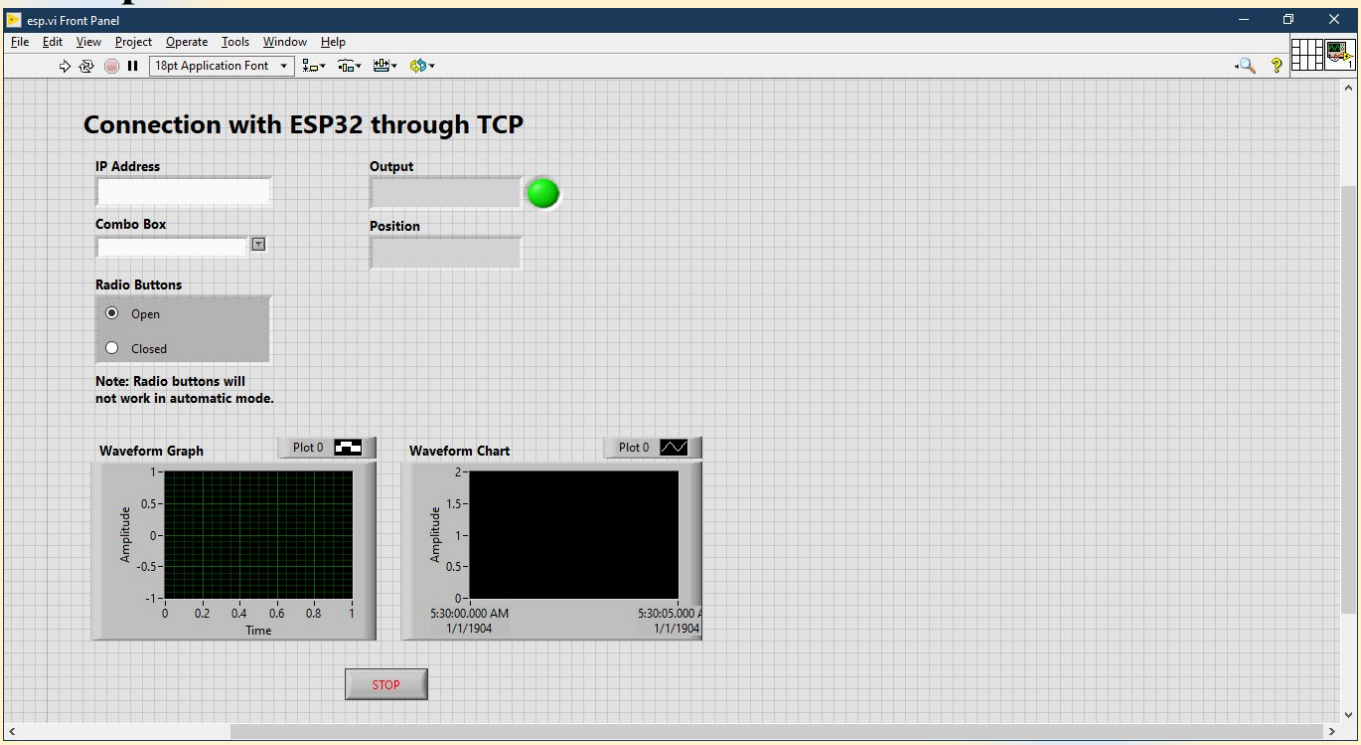


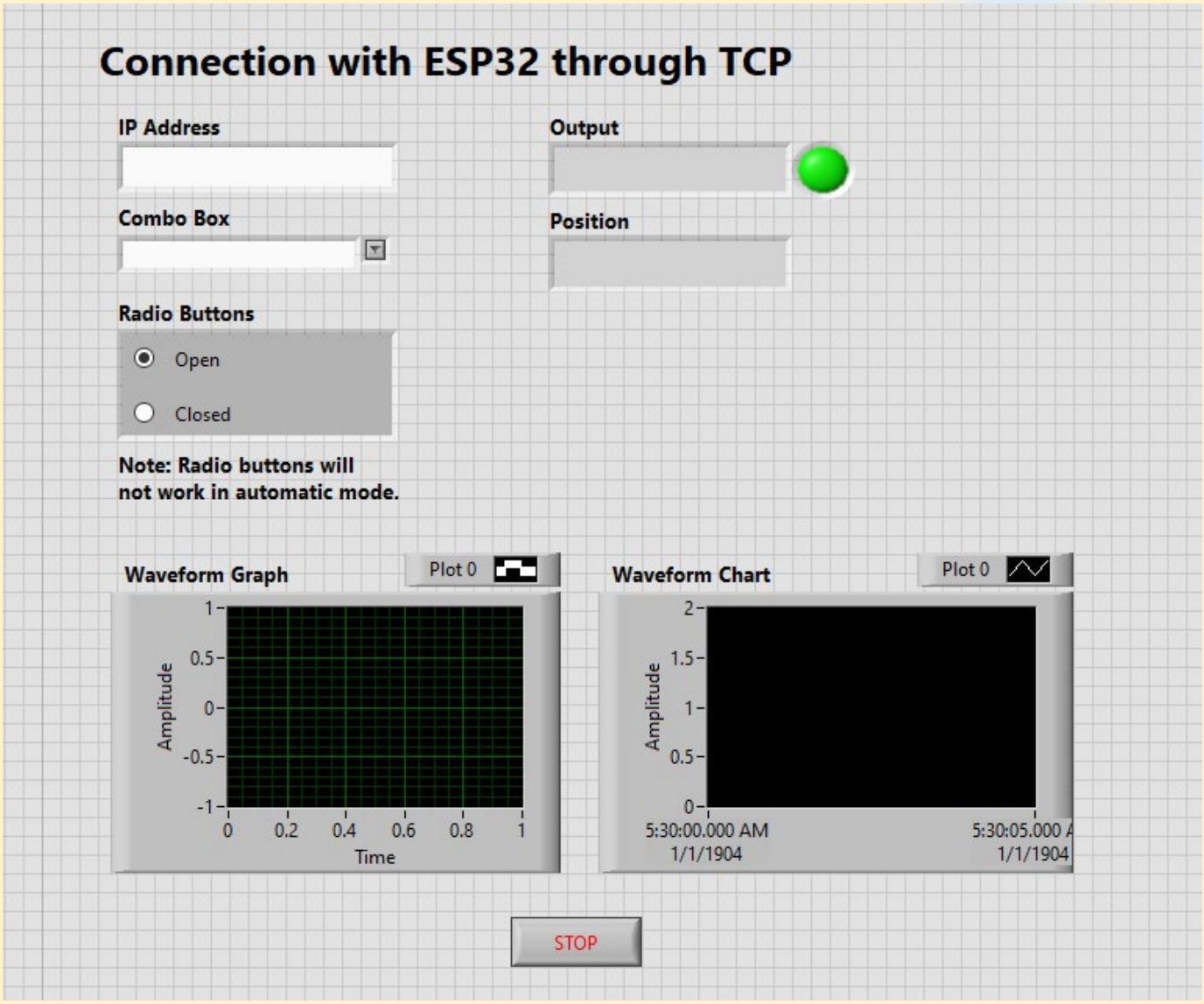
Code:

```
#include <WiFi.h>
#include <Servo.h>

#define ir 23
#define servo 22

#define id "TheCosmos"
#define pass "thedivinefun"

Servo myservo;
int pos = 0;
int inputVal=0;

WiFiServer server(80);

void setup() {
  myservo.attach(servo);
  pinMode(ir,INPUT);
  Serial.begin(9600);
  Serial.print("\nConnecting to ");
  Serial.println(id);

  WiFi.begin(id, pass);
  while (WiFi.status() != WL_CONNECTED) { // Waiting for successful connection
       delay(500);
       Serial.print(".");
  }
  Serial.print("WiFi connected. IP address: ");
  Serial.println(WiFi.localIP());

  server.begin();
}

void servo180(){ //to close
  for (pos = 0; pos <= 180; pos += 1) {
    myservo.write(pos);
  }
}

void servo0(){ //to open
  for (pos = 180; pos >= 0; pos -= 1) {
    myservo.write(pos);
  }
}

void lab(){
  if(digitalRead(ir)==1){
     if(pos==0){
       servo180();
     }
     else{
       pos=180;
     }
   }
   if (digitalRead(ir)==0){
     if(pos==180){
       servo0();
     }
     else{
       pos=0;
     }
   }
```

```
}

void loop() {
  WiFiClient client = server.available();

  if (client){
    if(client.connected()){
      Serial.println("Client Connected");
    }
    while(client.connected()){
      while(client.available()>0)
      {
        uint8  t data =client.read();
        Serial.write(data);

        switch(data){
          case 'a':
            lab();
            Serial.println(digitalRead(ir));
            client.print(digitalRead(ir));
            break;

          case 'c':
            Serial.println(digitalRead(ir));
            client.print(digitalRead(ir));
            if(pos==0){
              servo180();
            }
            else{
              pos=180;
            }
            break;

          case 'o':
            Serial.println(digitalRead(ir));
            client.print(digitalRead(ir));
            if(pos==180){
              servo0();
            }
            else{
              pos=0;
            }
            break;
        }
        //Serial.println(digitalRead(ir));
        //client.print(digitalRead(ir));
      }
    }
    client.stop();
    Serial.println("Client disconnected");
  }
  delay(100);
}
```

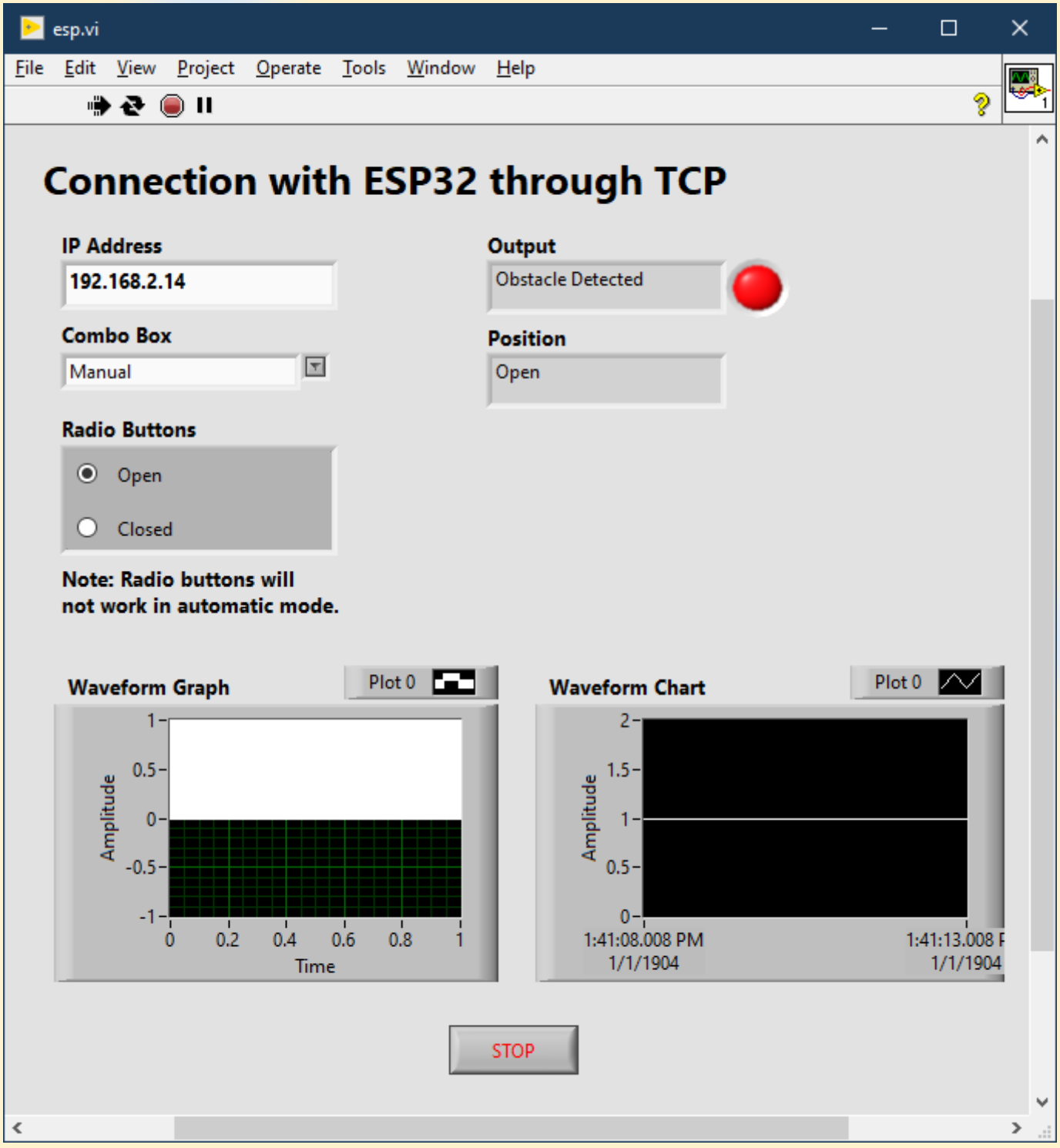
Front panel:


esp.vi Front Panel
File Edit View Project Operate Tools Window Help
18pt Application Font
Connection with ESP32 through TCP
IP Address
Output
Combo Box
Position
Radio Buttons
Open
Closed
Note: Radio buttons will not work in automatic mode.
Waveform Graph
Plot 0
Amplitude
Time
Waveform Chart
Plot 0
Amplitude
5:30:00.000 AM
1/1/1904
5:30:05.000
1/1/1904
STOP

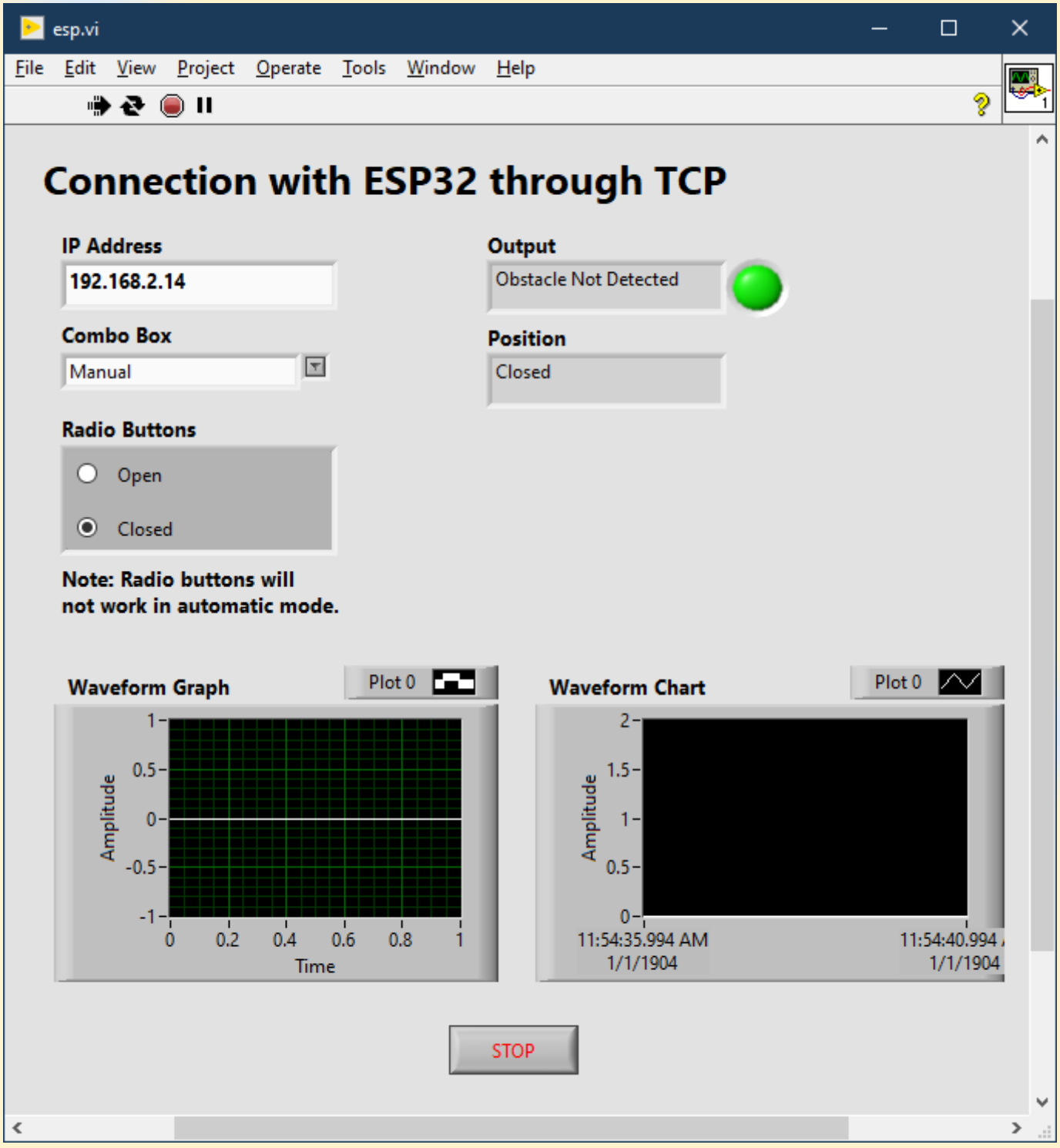

Connection with ESP32 through TCP
IP Address
Output
Combo Box
Position
Radio Buttons
Open
Closed
Note: Radio buttons will not work in automatic mode.
Waveform Graph
Plot 0
Amplitude
Time
Waveform Chart
Plot 0
Amplitude
5:30:00.000 AM
1/1/1904
5:30:05.000
1/1/1904
STOP

Result:

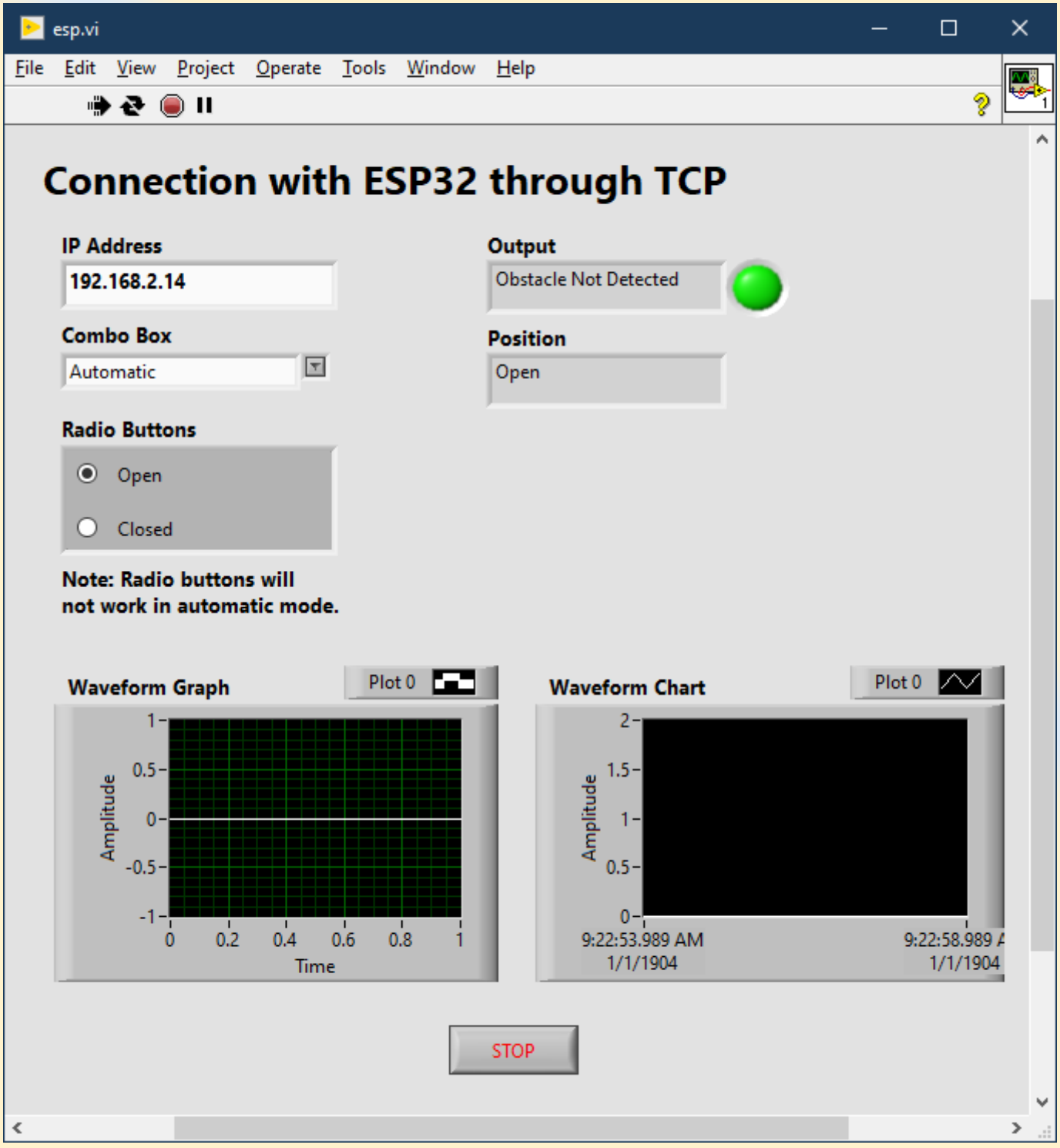
MANUAL CONTROL - OPEN POSITION



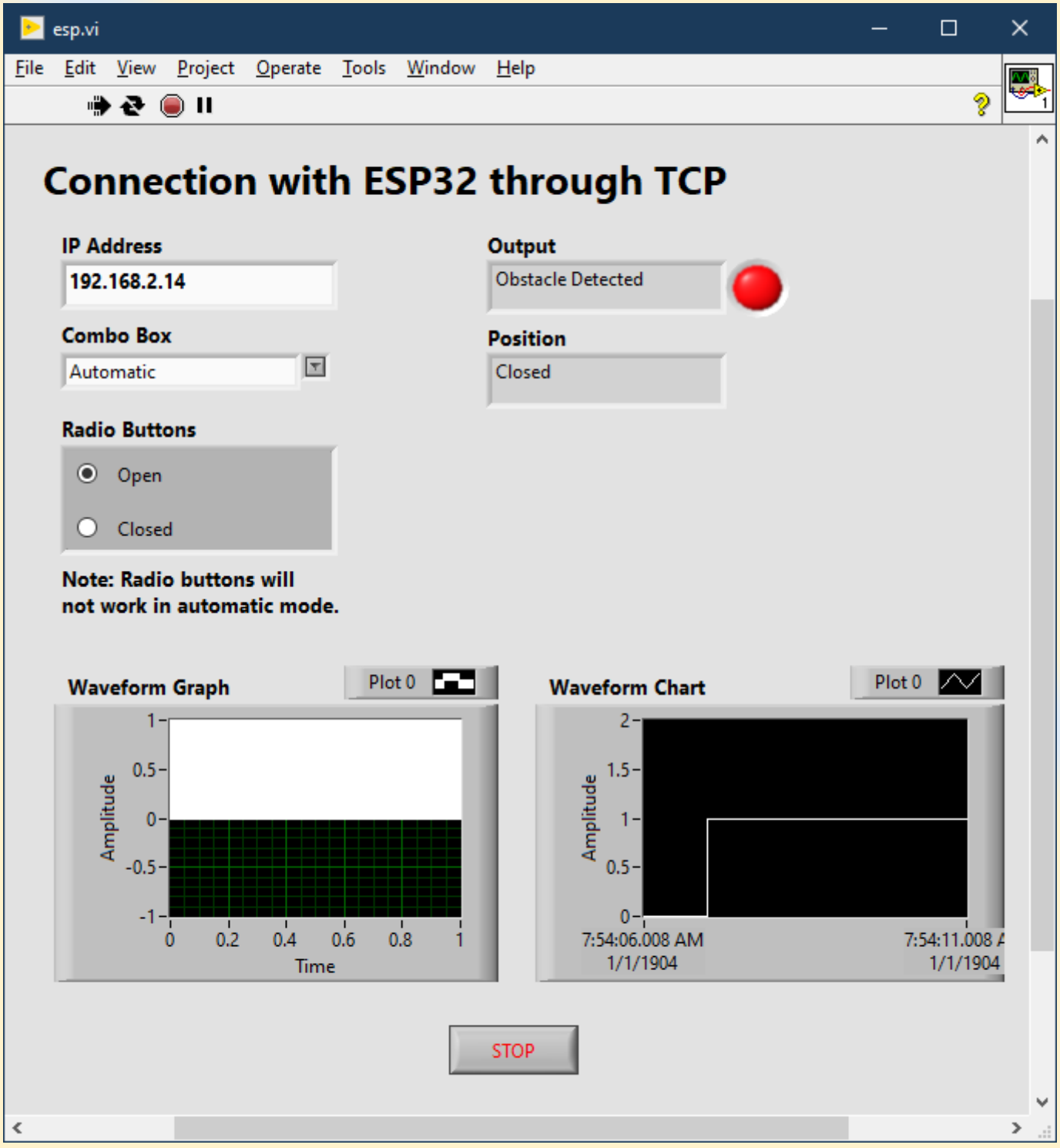
MANUAL CONDITION – CLOSED CONDITION



AUTOMATIC CONTROL- OPEN POSITION



AUTOMATIC CONTROL- CLOSED POSITION



File attached:

ir_servo_controller.ino

esp.vi