Joseph Daniel Moreno, U#60156204

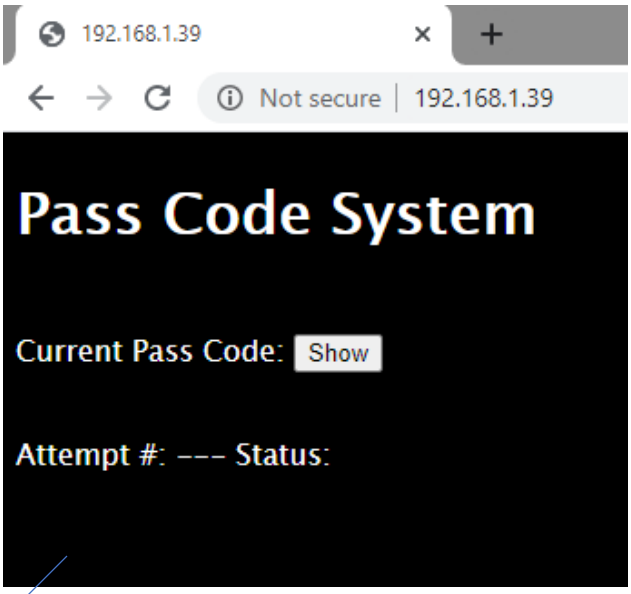
Embedded Systems – Final Project

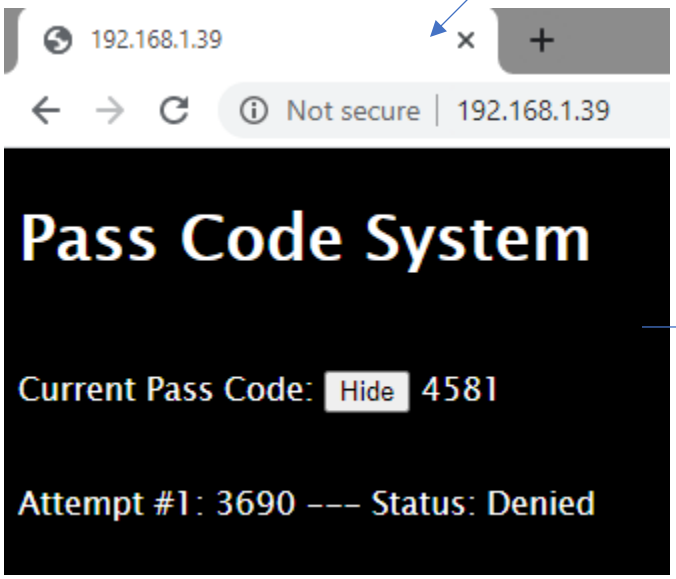
Problem Description:
Using the PIC18F45K50, create a system that uses two communication protocols (Wi-Fi being one of them) and create a user interface for the system.

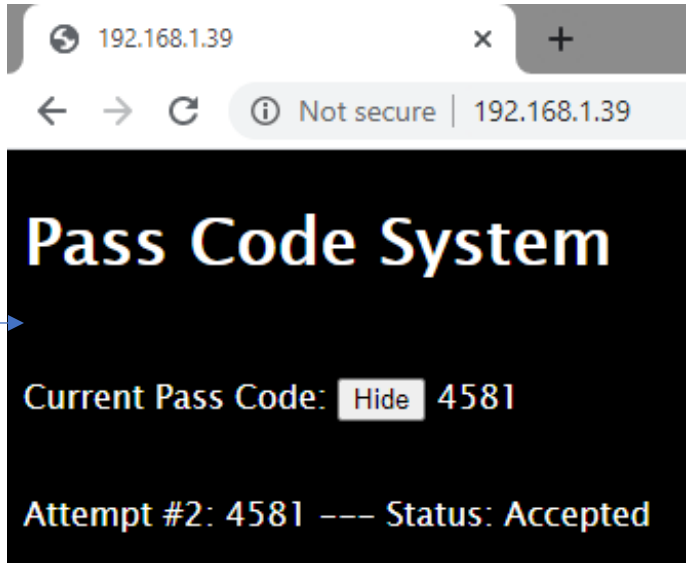
My System:
My system uses the ESP8266 ESP-01 Wi-Fi module, a 4x4 keypad, and the user interface is displayed as an HTML web page sent by the ESP8266. It is a pass code system that takes 4 numbers from the keypad, which are communicated serially to the ESP8266, and the program uploaded to the ESP8266 will determine whether the 4 entered numbers are the correct pass code. The ESP8266 then communicates this information using an HTML web page. My original plan to meet the problem description was to have I^2C as the second communication protocol and to use the DS1307 real-time clock to time stamp the pass code attempts, but my final design does not have it.

Demo Video Link:

https://www.youtube.com/watch?v=oDEwcmvcGWM







Joseph Daniel Moreno, U#60156204

PIC18F45K50 Code:
Code from “FinalProject.c”; included in .zip file.

```
#include "FinalProject_Header.h"

void main(void) {

    // Set a 16 MHz clock frequency.
    asm("MOVLW 0x78");
    asm("MOVWF OSCCON");

    // PORTA set up.
    // Column 1 (left-most column) is PORTA[0], ... , and column 3 is PORTA[2].
    asm("CLRF PORTA");
    asm("CLRF LATA");
    asm("CLRF TRISA");
    ANSELA = 0x00;

    // PORTB set up.
    // Row 1 (top row) is PORTB[7], row 2 is PORTB[6], ... , and row 4 is PORTB[4].
    asm("CLRF PORTB");
    asm("CLRF LATB");
    asm("CLRF TRISB");
    ANSELB = 0x00;
    TRISBbits.TRISB7 = 1;    // Row 1
    TRISBbits.TRISB6 = 1;    // Row 2
    TRISBbits.TRISB5 = 1;    // Row 3
    TRISBbits.TRISB4 = 1;    // Row 4

    // EUSART set up.
    SPBRGH1 = 0x00;
    SPBRG1 = 0x19;           // See 17.4, example 17-1 "Calculating Baud Rate Error"
    TXSTA1bits.BRGH = 0;
    BAUDCON1bits.BRG16 = 0; // Set BRGH and BRG16 to 0 for 9600 baud rate.
    TXSTA1bits.SYNC = 0;     // SYNC = 0 for asynchronous mode.
    RCSTA1bits.SPEN = 1;     // SPEN = 1 for enabling serial port.
    TXSTA1bits.TXEN = 1;     // TXEN = 1 for enabling transmission.
    asm("BSF TRISC, 6");     // TX pin
    asm("BSF TRISC, 7");     // RX pin (both TX and RX TRIS must be set to 1 for EUSART operation)
```

The clock frequency is set to 16 MHz, PORTA[2:0] is used as output (for the columns of the keypad), and PORTB[7:4] are used as input (for the rows of the keypad). The EUSART is set up with a baud rate of 9600.

```
    while(1) {
        // 1st column pulled high.
        PORTAbits.RA0 = 1;
        LATAbits.LATA0 = 1;

        if (PORTBbits.RB7) {
            TXREG1 = '1';
            __delay_ms(250);
        }

        if (PORTBbits.RB6) {
            TXREG1 = '4';
            __delay_ms(250);
        }

        if (PORTBbits.RB5) {
            TXREG1 = '7';
            __delay_ms(250);
        }

        // 2nd column pulled high.
        PORTAbits.RA0 = 0;
        LATAbits.LATA0 = 0;
        PORTAbits.RA1 = 1;
        LATAbits.LATA1 = 1;

        if (PORTBbits.RB7) {
            TXREG1 = '2';
            __delay_ms(250);
        }

        if (PORTBbits.RB6) {
            TXREG1 = '5';
            __delay_ms(250);
        }

        if (PORTBbits.RB5) {
            TXREG1 = '8';
            __delay_ms(250);
```

```
        if (PORTBbits.RB5) {
            TXREG1 = '8';
            __delay_ms(250);
        }

        if (PORTBbits.RB4) {
            TXREG1 = '0';
            __delay_ms(250);
        }

        // 3rd column pulled high.
        PORTAbits.RA1 = 0;
        LATAbits.LATA1 = 0;
        PORTAbits.RA2 = 1;
        LATAbits.LATA2 = 1;

        if (PORTBbits.RB7) {
            TXREG1 = '3';
            __delay_ms(250);
        }

        if (PORTBbits.RB6) {
            TXREG1 = '6';
            __delay_ms(250);
        }

        if (PORTBbits.RB5) {
            TXREG1 = '9';
            __delay_ms(250);
        }

        // Pull 3rd column low.
        PORTAbits.RA2 = 0;
        LATAbits.LATA2 = 0;
    }

    return;

}
```

The while loop sends a logic 1 through each of the columns (PORTA[2:0]) one at a time. The rows (PORTB[7:4]) are pulled to ground while the keypad isn't being pressed. 1 through 9 on the keypad send the corresponding ASCII value through the EUSART.

Joseph Daniel Moreno, U#60156204

ESP8266 ESP-01 Code:
Code from “FinalProject.ino”; included in .zip file.

```
/*
 * Rui Santos's code used as a template.
 * Complete project details at https://randomnerdtutorials.com
 */

// Load Wi-Fi library
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <WiFiClient.h>

// Replace with your network credentials
const char* ssid     = "MY_SSID";
const char* password = "MY_PASSWORD";

// Set web server port number to 80
WiFiServer server(80);

// Variable to store the HTTP request
String header;

// Array for receiving the 4-number code.
unsigned int code[4];
unsigned int i, attempt;
String output, pass_code, attempt_str, attempt_status;

// Current time
unsigned long currentTime = millis();
// Previous time
unsigned long previousTime = 0;
// Define timeout time in milliseconds (example: 2000ms = 2s)
const long timeoutTime = 20000;

// Soft-AP config
IPAddress ip(192,168,1,39);
IPAddress gateway(192,168,1,1);
IPAddress subnet(255,255,255,0);

void setup() {
  Serial.begin(9600); // *** Baud rate setting. ***
```

The 4 numbers entered by the user will be stored in code[4]. The current attempt number will be stored in attempt and attempt_str will be used to display that information on the HTML web page. The numbers entered from the latest attempt will be stored in output. The correct pass code is hard-coded into pass_code (for the demo video, it’s 4581). The attempt_status will say “Accepted” or “Denied” depending on whether the latest attempt matches pass_code. The baud rate of the ESP8266 is set to 9600.

```
  // Initialize the output string, the pass code, i, the attempt #, attempt string, and attempt status.
  output = "";
  pass_code = "4581";
  i = 0;
  attempt = 0;
  attempt_str = "";
  attempt_status = "";

  // Connect to Wi-Fi network with SSID and password
  /*ESP.eraseConfig();
  WiFi.disconnect();
  WiFi.setAutoConnect(false);
  WiFi.setAutoReconnect(false);*/
  //WiFi.hostname("ESP-5A2DA2");
  //WiFi.disconnect();
  //WiFi.config(ip, gateway, gateway, subnet); // Before WiFi connection...?
  WiFi.mode(WIFI_STA);
  //Serial.print("Connecting to ");
  //Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  // Print local IP address and start web server
  //Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  server.begin();
}
```

The variables are initialized and the ESP8266 connects to the Wi-Fi.

```
            // Display the HTML web page
            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");
            client.println("<link rel=\"icon\" href=\"data:,\">");
            client.println("<style>html { font-family: \"Lucida Sans Unicode\", \"Lucida Grande\", sans-serif; color: white; background-color: black; display: inline-bl
ock; margin: 0px auto;}");
            client.println("h1 { text-align: left } </style></head>");

            // Web Page Heading
            client.println("<body><h1>Pass Code System</h1>");

            // Web page body.
            client.println("<p style=\"display: inline-block\">Current Pass Code:</p>");
            client.println("<button id=\"ShowHideBut0\" onclick=\"ShowHidePassCode()\">Show</button>");
            client.println("<div id=\"PassCodeDiv\" style=\"display: none\">" + pass_code + "</div>");
            client.println("<br>");
            client.println("<p style=\"display: inline-block\">Attempt #" + attempt_str + ": " + output + " --- Status: " + attempt_status + "</p>");
```

```
// HTML script.
client.println("<script>");
client.println("function ShowHidePassCode() {");
client.println("var div0 = document.getElementById(\"PassCodeDiv\");");
client.println("var but0 = document.getElementById(\"ShowHideBut0\");");
client.println("if (div0.style.display === \"none\") {");
client.println("  div0.style.display = \"inline-block\";");
client.println("}else {");
client.println("  div0.style.display = \"none\";");
client.println("}");
client.println("if (but0.innerHTML === \"Show\") {");
client.println("  but0.innerHTML = \"Hide\";");
client.println("}else {");
client.println("  but0.innerHTML = \"Show\";");
client.println("}");
client.println("}");
client.println("</script>");

// Ending body tag.
client.println("</body></html>");
```

This is the HTML web page that is sent to connected clients. There's a "Show/Hide" button for the user to see pass_code, and the information of the latest attempt is shown in the body (attempt_str, output, and attempt_status).

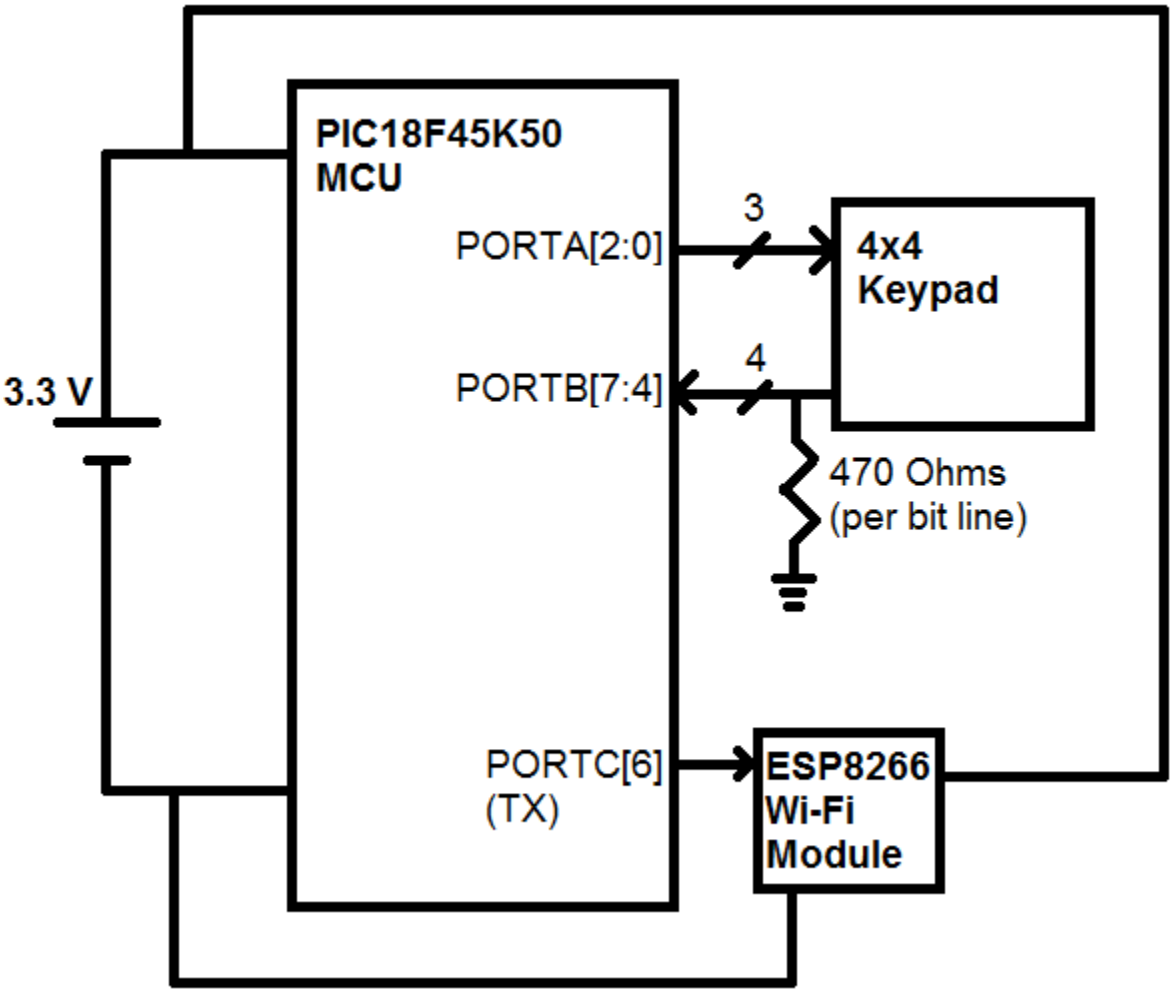
```
unsigned int x;
while(Serial.available() > 0) {
  serial_in = Serial.read();
  Serial.println(serial_in);

  if (serial_in >= '0' && serial_in <= '9') {
    code[i] = (serial_in - 48);

    if (i == 3) {
      output = "";
      ++attempt;

      attempt_str = String(attempt);

      for(x = 0; x < 4; ++x) {
        switch(code[x]) {
          case 0: output = output + "0";
                  break;
          case 1: output = output + "1";
                  break;
          case 2: output = output + "2";
                  break;
          case 3: output = output + "3";
                  break;
          case 4: output = output + "4";
                  break;
          case 5: output = output + "5";
                  break;
          case 6: output = output + "6";
                  break;
          case 7: output = output + "7";
                  break;
          case 8: output = output + "8";
                  break;
          case 9: output = output + "9";
                  break;
        }
      }

      if (output == pass_code) {
        attempt_status = "Accepted";
      }else {
        attempt_status = "Denied";
      }

      i = 0;
    }else ++i;
  }
 }
}
```

If a number from the keypad is pressed, the ESP8266 reads it in. To ensure it is a number, the if-statement tests if the character sent is >= ‘0’ and <= ’9’; if it is, then put it in code[4]. The unsigned int i is used as a counter, so it will increment with each received number. Once i == 3, the numbers stored in code[4] are converted to strings and concatenated with output.

The output is compared to the pass_code, attempt_status is set accordingly, and i is set back to 0 for the next attempt.

Joseph Daniel Moreno, U#60156204

Wiring Diagram:



System's Picture: