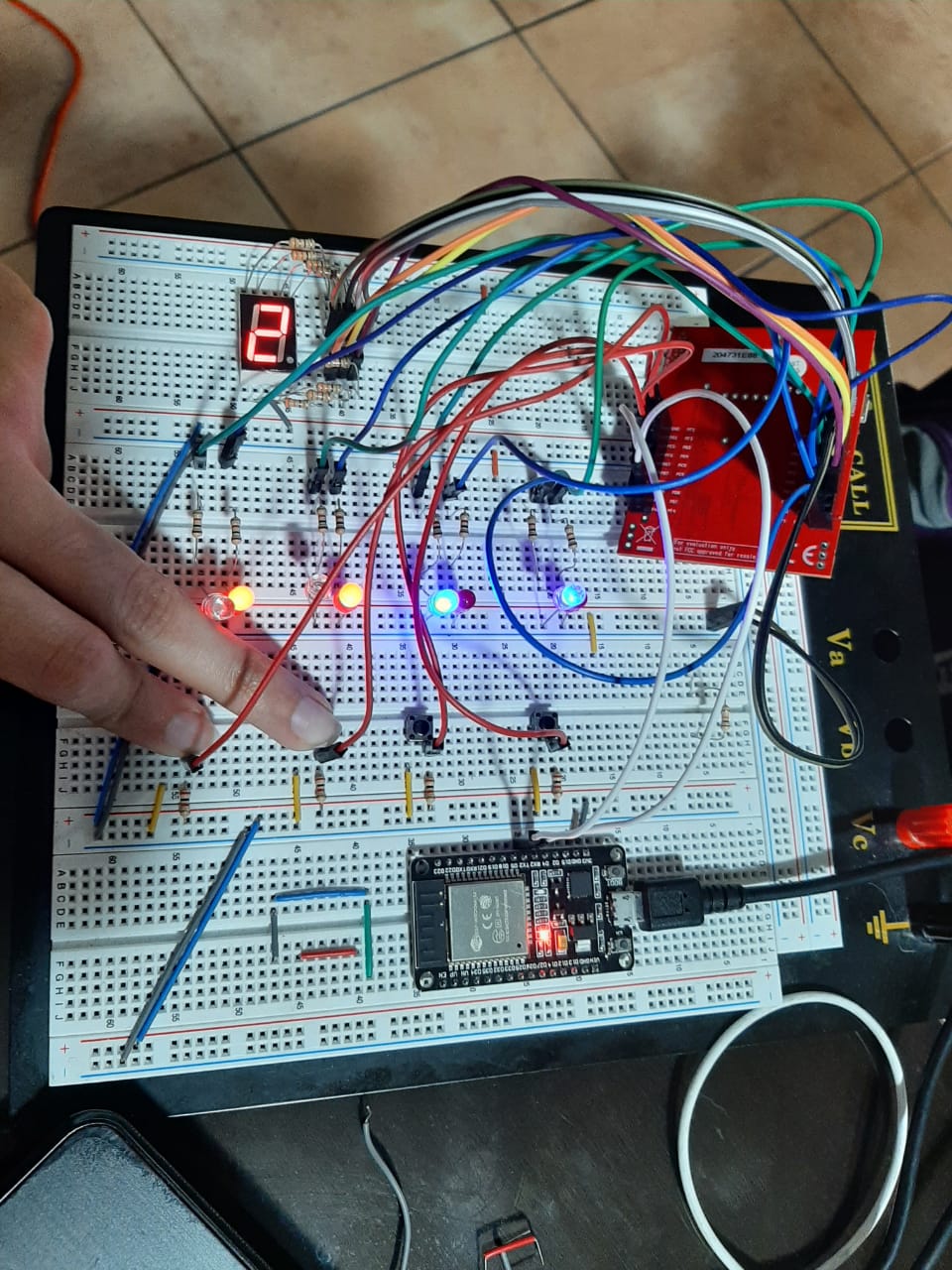
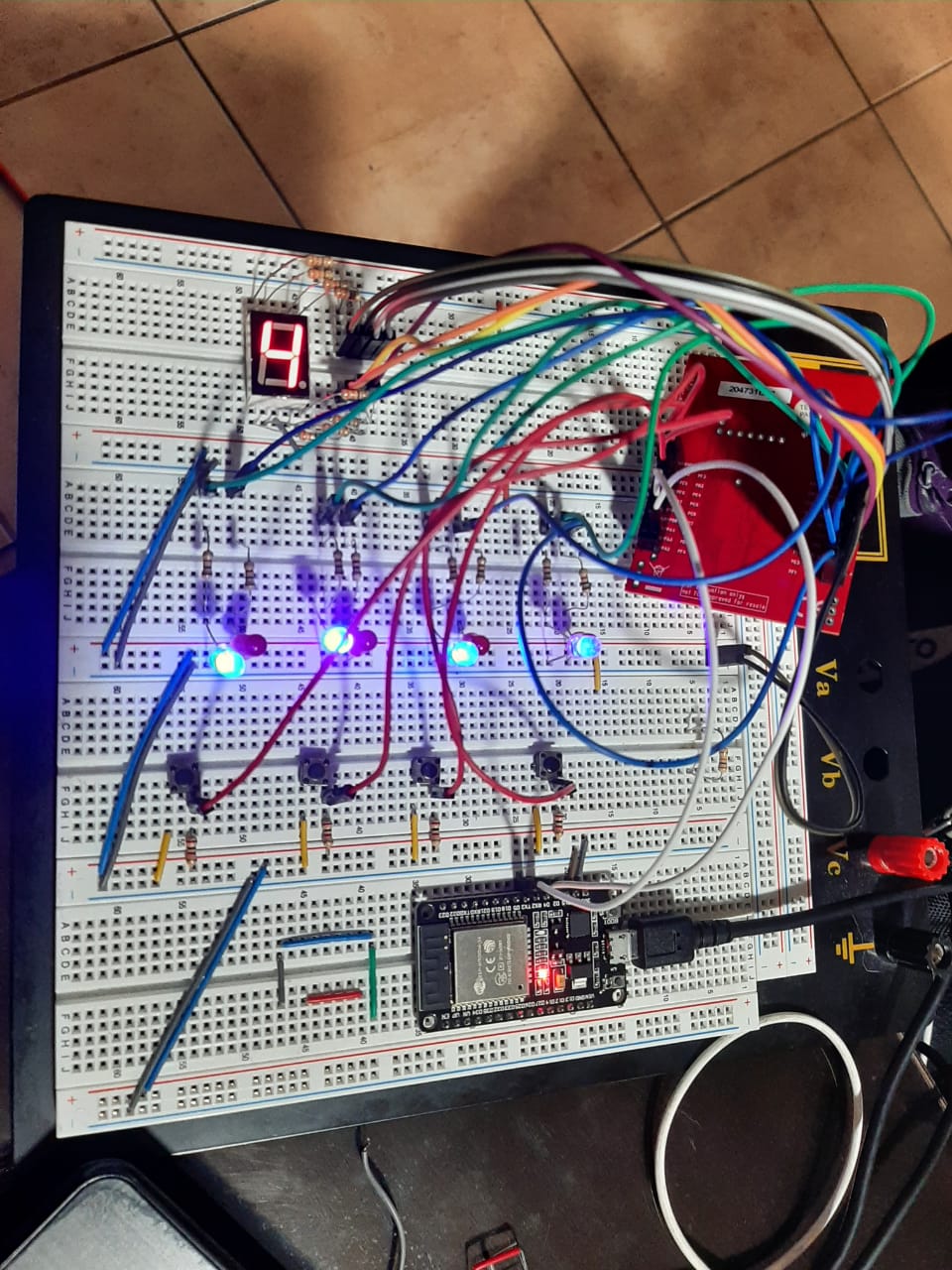
Camila Lemus Lone

18272

**INFORME PROYECTO #4**

**Sistema de parqueos – Tivaware y WebServer**

El proyecto consiste en un sistema de parqueos, el cual muestra la cantidad de parqueos disponibles en un display ánodo de 7 segmentos. Los leds rojos sirven para indicar que el parqueo está ocupado y los leds azules indican que el parqueo está disponible. Se utilizó push buttons para simular los sensores. A continuación, se muestra un ejemplo del funcionamiento de los sensores y el display.



La lógica en la que me basé es básicamente utiliza arrays para los parqueos y va chequeando las posiciones dependiendo del botón que se presiona. La lógica de mi programa es un poco compleja ya que, para chequear los 4 botones, utiliza 4 loops (1 por botón). Esto es, ya que mi loop llama a la función Parqueos() únicamente cuando el botón cambia de estado. En esta función, se enciende o se apaga el led correspondiente según lo que lee el sensor y se manda un carácter mediante comunicación UART, que indica que parqueo fue el que cambió. Para el display, lo único que se hizo fue una función en la cual mediante un for de 0 a 3 (que son las posiciones del array) chequea el estado del botón y va sumando los valores del array que chequeaba el estado del botón. Luego mediante un case switch se muestra el número que corresponde a la suma de los valores del arreglo y se muestra el numero en el display.

**Código Comentado**

**TIVAWARE**

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Camila Lemus Lone

// 18272

// Universidad del Valle de Guatemala

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**#include**<stdint.h>

**#include**<stdbool.h>

**#include**"inc/hw\_memmap.h"

**#include**"inc/hw\_types.h"

**#include**"inc/tm4c123gh6pm.h"

**#include**"driverlib/sysctl.h"

**#include**"driverlib/gpio.h"

**#include**"driverlib/timer.h"

**#include**"driverlib/interrupt.h"

**#include** "driverlib/uart.h"

**#include** "driverlib/pin\_map.h"

**#include** "inc/hw\_gpio.h"

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*VARIABLES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**bool** flag = **false**;

**int** led;

**char** letra;

**int** parqueo;

**int** contador;

**int** suma;

//int cont[] = {0,0,0,0};

**int** pushP[] = { GPIO\_PORTF\_BASE, GPIO\_PORTF\_BASE, GPIO\_PORTB\_BASE,

GPIO\_PORTC\_BASE };

**int** pushV[] = { GPIO\_PIN\_2, GPIO\_PIN\_3, GPIO\_PIN\_3, GPIO\_PIN\_4 };

**int** state[] = { 0, 0, 0, 0 };

**int** stateantes[] = { 1, 1, 1, 1 };

//GPIOPinRead(GPIO\_PORTF\_BASE, GPIO\_PIN\_2); //Lee push 1

//GPIOPinRead(GPIO\_PORTF\_BASE, GPIO\_PIN\_3); //Lee push 2

//GPIOPinRead(GPIO\_PORTB\_BASE, GPIO\_PIN\_3); //Lee push 3

//GPIOPinRead(GPIO\_PORTC\_BASE, GPIO\_PIN\_4); //Lee push 4

**#define** RED GPIO\_PIN\_1

**#define** GREEN GPIO\_PIN\_3

**#define** BLUE GPIO\_PIN\_2

**#define** OFF 0

// Display

//-----------

// PB5 - g

// PB0 - f

// PB1 - a

// PE4 - b

// PE5 - e

// PB4 - d

// PA5 - c

// Leds azules

//------------

// PD0

// PD1

// PD2

// PD3

// Leds rojos

//------------

// PE1

// PE2

// PE3

// PF1

//Pushes

//------------

// PF2

// PF3

// PB3

// PC4

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*INSTANCIAR FUNCIONES\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void** **UART\_Config**(**void**);

**void** **Parqueos**(**void**);

**void** **Display**(**void**);

/\*\*

\* main.c

\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**int** **main**(**void**)

{

**SysCtlClockSet**(

SYSCTL\_SYSDIV\_5 | SYSCTL\_USE\_PLL | SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN); // Clk a 40MHz

UART\_Config();

//Peripheral Enable

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOA);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOB);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOC);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOD);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOE);

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOF);

//PUSH BUTTONS

**GPIOPinConfigure**(GPIO\_PF2\_T1CCP0);

**GPIOPinConfigure**(GPIO\_PF3\_T1CCP1); //2

**GPIOPinConfigure**(GPIO\_PB3\_T3CCP1); //3

**GPIOPinConfigure**(GPIO\_PC4\_WT0CCP0); //4

//LEDS

**GPIOPinConfigure**(GPIO\_PD0\_WT2CCP0); //Parque 1 verde D0

// GPIOPinConfigure(GPIO\_PE1\_U7TX); //Parque 1 roja E1

**GPIOPinConfigure**(GPIO\_PD1\_WT2CCP1); //Parque 2 verde D1

// GPIOPinConfigure(GPIO\_PD0\_WT2CCP0); //Parque 2 roja E2

**GPIOPinConfigure**(GPIO\_PD2\_WT3CCP0); //Parque 3 verde D2

// GPIOPinConfigure(GPIO\_PD0\_WT2CCP0); //Parque 3 roja E3

**GPIOPinConfigure**(GPIO\_PD3\_WT3CCP1); //Parque 4 verde D3

**GPIOPinConfigure**(GPIO\_PF1\_TRD1); //Parque 4 roja F1

//Salidas y entradas

**GPIOPinTypeGPIOOutput**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5); //A

**GPIOPinTypeGPIOOutput**(

GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_4 | GPIO\_PIN\_5); //B

**GPIOPinTypeGPIOOutput**(GPIO\_PORTD\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3); //D

**GPIOPinTypeGPIOOutput**(

GPIO\_PORTE\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3 | GPIO\_PIN\_4

| GPIO\_PIN\_5); //E

**GPIOPinTypeGPIOOutput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1); //F

**GPIOPinTypeGPIOInput**(GPIO\_PORTB\_BASE, GPIO\_PIN\_3); //B

**GPIOPinTypeGPIOInput**(GPIO\_PORTC\_BASE, GPIO\_PIN\_4); //C

**GPIOPinTypeGPIOInput**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2 | GPIO\_PIN\_3); //F

//Push Button - Pullup

**GPIOPadConfigSet**(GPIO\_PORTF\_BASE, GPIO\_PIN\_2, GPIO\_STRENGTH\_2MA,

GPIO\_PIN\_TYPE\_STD\_WPU);

**GPIOPadConfigSet**(GPIO\_PORTF\_BASE, GPIO\_PIN\_3, GPIO\_STRENGTH\_2MA,

GPIO\_PIN\_TYPE\_STD\_WPU);

**GPIOPadConfigSet**(GPIO\_PORTC\_BASE, GPIO\_PIN\_4, GPIO\_STRENGTH\_2MA,

GPIO\_PIN\_TYPE\_STD\_WPU);

**GPIOPadConfigSet**(GPIO\_PORTB\_BASE, GPIO\_PIN\_3, GPIO\_STRENGTH\_2MA,

GPIO\_PIN\_TYPE\_STD\_WPU);

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_0, GPIO\_PIN\_0); //Led verde parqueo 1 encendida

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_1, GPIO\_PIN\_1); //Led verde parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_2, GPIO\_PIN\_2); //Led verde parqueo 3 encendida

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_3, GPIO\_PIN\_3); //Led verde parqueo 4 encendida

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_1, 0); //Led roja parqueo 1 OFF

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_2, 0); //Led roja parqueo 2 OFF

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_3, 0); //Led roja parqueo 3 OFF

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 0); //Led roja parqueo 4 OFF

UART\_Config();

**int** n = 0;

**while** (1)

{

// contador = parqueo1 + parqueo2 + parqueo3 + parqueo4;

state[n] = **GPIOPinRead**(pushP[n], pushV[n]); //Lee el push que apachó

**if** (state[n] != stateantes[n]) //Si cambia de estado

{

stateantes[n] = state[n];

parqueo = n + 1;

Parqueos();

}

**else**

{

}

**if** (n < 4) //Sirve para que el n se reinicie cuando pase el tamaño del array

{

n++;

}

**else**

{

n = 0;

}

Display();

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* UART \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void** **UART\_Config**(**void**)

{

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_UART2); //Peripheral del UART enable

**SysCtlPeripheralEnable**(SYSCTL\_PERIPH\_GPIOD); //Peripheral del puerto C enable

**GPIOPinConfigure**(GPIO\_PD6\_U2RX);

**GPIOPinConfigure**(GPIO\_PD7\_U2TX);

**GPIOPinTypeUART**(GPIO\_PORTD\_BASE, GPIO\_PIN\_6 | GPIO\_PIN\_7); //Los pines TX y RX se ponen en modo UART

//Inicializar el modulo UART con: 115200, 8 data bits, 1 stop bit, None parity.

**UARTConfigSetExpClk**(

UART2\_BASE, **SysCtlClockGet**(), 115200,

UART\_CONFIG\_WLEN\_8 | UART\_CONFIG\_STOP\_ONE | UART\_CONFIG\_PAR\_NONE);

**UARTEnable**(UART2\_BASE);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* LEDS PARQUEO \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void** **Parqueos**(**void**)

{

**switch** (parqueo)

{

**case** 1:

**if** (state[0] == 0)

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_0, 0); //Led verde parqueo 1 OFF

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_1, GPIO\_PIN\_1); //Led roja parqueo 1 ON

**UARTCharPut**(UART2\_BASE, 'A');

}

**else**

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_0, GPIO\_PIN\_0); //Led verde parqueo 1 ON

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_1, 0); //Led roja parqueo 1 OFF

**UARTCharPut**(UART2\_BASE, 'B');

}

**break**;

**case** 2:

**if** (state[1] == 0)

{

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_2, GPIO\_PIN\_2); //Led verde parqueo 2

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_1, 0); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'C');

}

**else**

{

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_2, 0); //Led ROJO parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_1, GPIO\_PIN\_1); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'D');

}

**break**;

**case** 3:

**if** (state[2] == 0)

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_2, 0); //Led ROJO parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_3, GPIO\_PIN\_3); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'E');

}

**else**

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_2, GPIO\_PIN\_2); //Led ROJO parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_3, 0); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'F');

}

**break**;

**case** 4:

**if** (state[3] == 0)

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_3, 0); //Led ROJO parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, GPIO\_PIN\_1); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'G');

}

**else**

{

**GPIOPinWrite**(GPIO\_PORTD\_BASE, GPIO\_PIN\_3, GPIO\_PIN\_3); //Led ROJO parqueo 2 encendida

**GPIOPinWrite**(GPIO\_PORTF\_BASE, GPIO\_PIN\_1, 0); //Led ROJO parqueo 2 encendida

**UARTCharPut**(UART2\_BASE, 'H');

}

**break**;

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* DISPLAY \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**void** **Display**(**void**) //Enciende el numero correspondiente en el display

{

suma = 4;

**int** i;

**for**(i=0; i<4; i++){ //For que chequea cuantos parqueos hay disponibles

**if** (state[i] == 0){

suma--;

}

}

**switch** (suma)

{

**case** 0:

//CERO

**GPIOPinWrite**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5, 0); //A

**GPIOPinWrite**(GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5,

GPIO\_PIN\_5); //B

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_4 | GPIO\_PIN\_5, 0); //E

**break**;

**case** 1:

//UNO

**GPIOPinWrite**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5, OFF); //A

**GPIOPinWrite**(GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5); //B

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_4 | GPIO\_PIN\_5, GPIO\_PIN\_5); //E

**break**;

**case** 2:

//DOS

**GPIOPinWrite**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5, GPIO\_PIN\_5); //A

**GPIOPinWrite**(GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5,

GPIO\_PIN\_0); //B

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_4 | GPIO\_PIN\_5, OFF); //E

**break**;

**case** 3:

//TRES

**GPIOPinWrite**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5, OFF); //A

**GPIOPinWrite**(GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5,

GPIO\_PIN\_0); //B

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_4 | GPIO\_PIN\_5, GPIO\_PIN\_5); //E

**break**;

**case** 4:

//CUATRO

**GPIOPinWrite**(GPIO\_PORTA\_BASE, GPIO\_PIN\_5, OFF); //A

**GPIOPinWrite**(GPIO\_PORTB\_BASE,

GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_4 | GPIO\_PIN\_5,

GPIO\_PIN\_1 | GPIO\_PIN\_4); //B

**GPIOPinWrite**(GPIO\_PORTE\_BASE, GPIO\_PIN\_4 | GPIO\_PIN\_5, GPIO\_PIN\_5); //E

**break**;

}

}

**Código Comentado**

**WEBSERVER**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ESP32 Web Server

Ejemplo de creación de Web server

Basándose en los ejemplos de:

https://lastminuteengineers.com/creating-esp32-web-server-arduino-ide/

https://electropeak.com/learn

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Librerías

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#include <WiFi.h>

#include <WebServer.h>

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Variables globales

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// SSID & Password

const char\* ssid = "CLARO1\_1CFE9F"; // Enter your SSID here

const char\* password = "485f7wvKFq"; //Enter your Password here

WebServer server(80); // Object of WebServer(HTTP port, 80 is defult)

bool parqueo1 = HIGH;

bool parqueo2 = HIGH;

bool parqueo3 = HIGH;

bool parqueo4 = HIGH;

uint8\_t LED1pin = 2;

bool LED1status = LOW;

int cont = 0;

int entrada = 0;

#define RXD2 16

#define TXD2 17

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Configuración

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void setup() {

Serial.begin(115200); //baudrate

Serial2.begin(115200, SERIAL\_8N1, RXD2, TXD2);

pinMode(LED1pin, OUTPUT);

// Connect to your wi-fi modem

WiFi.begin(ssid, password);

// Check wi-fi is connected to wi-fi network

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected successfully");

Serial.print("Got IP: ");

Serial.println(WiFi.localIP()); //Show ESP32 IP on serial

server.on("/", handle\_OnConnect); // Directamente desde e.g. 192.168.0.8

server.on("/led1on", handle\_led1on);

server.on("/led1off", handle\_led1off);

server.onNotFound(handle\_NotFound);

server.begin();

Serial.println("HTTP server started");

delay(100);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// loop principal

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void loop() {

server.handleClient();

// if (Serial2.read()> -1){

entrada = Serial2.read();

Serial.println(entrada);

//}

if (LED1status)

{

digitalWrite(LED1pin, HIGH);

}

else

{

digitalWrite(LED1pin, LOW);

}

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Handler de Inicio página

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void handle\_OnConnect() {

LED1status = LOW;

Serial.println("GPIO2 Status: OFF");

server.send(200, "text/html", SendHTML(LED1status));

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Handler de led1on

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void handle\_led1on() {

LED1status = HIGH;

Serial.println("GPIO2 Status: ON");

server.send(200, "text/html", SendHTML(LED1status));

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Handler de led1off

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void handle\_led1off() {

LED1status = LOW;

Serial.println("GPIO2 Status: OFF");

server.send(200, "text/html", SendHTML(LED1status));

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Procesador de HTML

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

String SendHTML(uint8\_t led1stat) {

String ptr = "<!DOCTYPE html> <html>\n";

ptr += "<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1.0, user-scalable=no\">\n";

ptr += "<title>LED Control</title>\n";

ptr += "<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}\n";

ptr += "body{margin-top: 50px;} h1 {color: #444444;margin: 50px auto 30px;} h3 {color: #444444;margin-bottom: 50px;}\n";

ptr += ".button {display: block;width: 80px;background-color: #3498db;border: none;color: white;padding: 13px 30px;text-decoration: none;font-size: 25px;margin: 0px auto 35px;cursor: pointer;border-radius: 4px;}\n";

ptr += ".button-on {background-color: #3498db;}\n";

ptr += ".button-on:active {background-color: #2980b9;}\n";

ptr += ".button-off {background-color: #34495e;}\n";

ptr += ".button-off:active {background-color: #2c3e50;}\n";

ptr += "p {font-size: 14px;color: #888;margin-bottom: 10px;}\n";

ptr += "table {";

ptr += " font-family: arial, sans-serif;";

ptr += " border-collapse: collapse;";

ptr += " width: 100%;";

ptr += "}";

ptr += "td, th {";

ptr += " border: 1px solid #dddddd;";

ptr += " text-align: left;";

ptr += " padding: 8px;";

ptr += "}";

ptr += "tr:nth-child(even) {";

ptr += " background-color: #dddddd;";

ptr += "}";

ptr += "</style>\n";

ptr += "</head>\n";

ptr += "<body>\n";

ptr += "<h1>ESP32 Web Server &#128664</h1>\n";

ptr += "<h3>Ejemplo de Web Server</h3>\n";

if (led1stat)

{

ptr += "<p>LED1 Status: ON</p><a class=\"button button-off\" href=\"/led1off\">OFF</a>\n";

}

else

{

ptr += "<p>LED1 Status: OFF</p><a class=\"button button-on\" href=\"/led1on\">ON</a>\n";

}

ptr += "<h2>HTML Table</h2>";

ptr += "<table>";

ptr += " <tr>";

ptr += " <th>Nombre del parqueo</th>";

ptr += " <th>Estado</th>";

ptr += " </tr>";

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

if (entrada == 65){ //Parqueo 1 Ocupado

ptr += " <tr>";

ptr += " <td>Parqueo 1</td>";

ptr += " <td>Ocupado</td>";

ptr += " </tr>";

}

else if (entrada == 66){

ptr += " <tr>";

ptr += " <td>Parqueo 1</td>";

ptr += " <td>Disponible</td>";

ptr += " </tr>";

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

else if (entrada == 67){//Parqueo 2 Ocupado

ptr += " <tr>";

ptr += " <td>Parqueo 2</td>";

ptr += " <td>Ocupado</td>";

ptr += " </tr>";

}

else if (entrada == 68){

ptr += " <tr>";

ptr += " <td>Parqueo 2</td>";

ptr += " <td>Disponible</td>";

ptr += " </tr>";

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

else if(entrada == 69){ //Parqueo 3 ocupado

ptr += " <tr>";

ptr += " <td>Parqueo 3</td>";

ptr += " <td>Ocupado</td>";

ptr += " </tr>";

}

else if (entrada == 70){

ptr += " <tr>";

ptr += " <td>Parqueo 3</td>";

ptr += " <td>Disponible</td>";

ptr += " </tr>";

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

else if(entrada == 71){ //Parqueo 4 ocupado

ptr += " <tr>";

ptr += " <td>Parqueo 4</td>";

ptr += " <td>Ocupado</td>";

ptr += " </tr>";

}

else if (entrada == 72){

ptr += " <tr>";

ptr += " <td>Parqueo 4</td>";

ptr += " <td>Disponible</td>";

ptr += " </tr>";

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ptr += "</table>";

ptr += "</body>\n";

ptr += "</html>\n";

return ptr;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Handler de not found

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void handle\_NotFound() {

server.send(404, "text/plain", "Not found");

}