Universidad del Valle de Guatemala
Electrónica Digital 2
Prof. Pablo Mazariegos
Prof. Kurt Kellner

Diego Mencos
Carné 18300

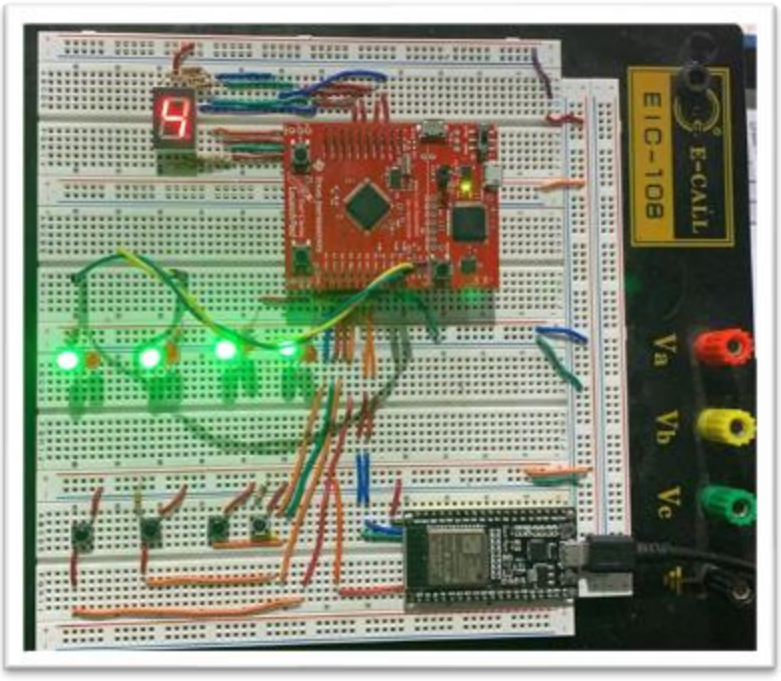
Proyecto #4

Circuito

Para este proyecto, se utilizaron:

- Tiva C
- ESP32
- 4 PushButtoms
- 8 LED (4 rojos y 4 verdes)
- 1 Display de 7 segmentos
- 12 resistencias 220 ohms
- 4 resistencias de 10 K

A continuación, se muestra una fotografía del circuito ya armado:



Explicación

La implementación de este proyecto fue relativamente sencilla, debido a que ya teníamos los conocimientos previos de todo el curso. En la tiva C se controlaron los 8 LEDs, pushbuttoms y el display de 7 segmentos. Para manipular los 8 LEDs solo se usaron 4 salidas de la tiva, debido a que se colocaron cada par de LEDs inversamente. De esta manera cada vez que se encendiera uno, el otro automáticamente iba a estar apagado, y viceversa. Esto ayudo a que no se usasen

tantos pines de la Tiva C. El ESP32 únicamente se utilizó para poder controlar el servidor y poderlo manipular mediante la información que recibía de la Tiva mediante UART. Para poder mostrar si un parqueo esta libre u ocupado, se utilizaron 2 emojis, los cuales se pueden ver en el video adjunto a este archivo.

Código

Código Code Composer Tiva C:

```
//Bibliotecas
#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 a usar en el programa
int status1=1;
int status2=1;
int status3=1;
int status4=1;

int status1prev=1;
int status2prev=1;
int status3prev=1;
int status4prev=1;

int contador=0;
int parqueo1=0;
int parqueo2=0;
int parqueo3=0;
```

```
int parqueo4=0;


//DISPLAY CODIGO
//a- PB5
//b- PB0
//c- PA5
//d- PA7
//e- PA6
//f- PE4
//g- PE5
//.- PB4

//Prototipo de funciones
void UARTIntHandler(void);  //funcion interrupcion cuando se recibe un dato
por UART
void InitUART(void); //funcion de configuracion UART 0
void InitUART1(void); //funcion para configurar UART1 que es el que
estarea utilizando
void InitUART2(void);
void LEDsparqueo(void);
void delayMs(uint32_t  ui32Ms);


//Main Loop
int main(void) {
   //Establecer reloj del microcontrolador a 40 MGHZ
   SysCtlClockSet(SYSCTL_SYSDIV_5|  SYSCTL_USE_PLL|
SYSCTL_XTAL_16MHZ);

   //Funcion de configuracion de UART1
   InitUART1();

   //Habilitar periféricos que estare utilizando
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);
```

```
    //Habilitar pines de salida y entrada

    GPIOPinConfigure(GPIO_PD6_WT5CCP0);  //Entrada Parqueo 3
    GPIOPinConfigure(GPIO_PC7_WT1CCP1);  //Entrada Parqueo 2
    GPIOPinConfigure(GPIO_PC6_WT1CCP0);  //Entrada Parqueo 1
    GPIOPinConfigure(GPIO_PB3_T3CCP1);  //Salida Parqueo 4
    GPIOPinConfigure(GPIO_PF2_T1CCP0);  // Salida parqueo 2
    GPIOPinConfigure(GPIO_PF3_TRCLK);  //Salida parqueo 3
    GPIOPinConfigure(GPIO_PA3_SSI0FSS);  //Entrada Parqueo 4
    GPIOPinConfigure(GPIO_PB2_T3CCP0);  //Salida parqueo 1




    GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE,
GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3);
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE,
GPIO_PIN_2|GPIO_PIN_3);

//SALIDAS DEL DISPLAY
    GPIOPinTypeGPIOOutput(GPIO_PORTA_BASE,
GPIO_PIN_7|GPIO_PIN_6|GPIO_PIN_5);
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_1|  GPIO_PIN_0|GPIO_PIN_5);
    GPIOPinTypeGPIOOutput(GPIO_PORTE_BASE,
GPIO_PIN_4|GPIO_PIN_5);

    GPIOPinTypeGPIOInput(GPIO_PORTF_BASE,  GPIO_PIN_4);
    //Configura los pushbuttoms  con weak pull-up

   //  GPIOPinTypeGPIOInput(GPIO_PORTD_BASE,  GPIO_PIN_7);
    GPIOPinTypeGPIOInput(GPIO_PORTD_BASE,  GPIO_PIN_6);
    GPIOPinTypeGPIOInput(GPIO_PORTC_BASE,  GPIO_PIN_6);
    GPIOPinTypeGPIOInput(GPIO_PORTC_BASE,  GPIO_PIN_7);
    GPIOPinTypeGPIOInput(GPIO_PORTA_BASE,  GPIO_PIN_3);
    //Configura los pushbuttoms  con weak pull-up

    GPIOPadConfigSet(GPIO_PORTF_BASE,  GPIO_PIN_4,
GPIO_STRENGTH_2MA,  GPIO_PIN_TYPE_STD_WPU);
```

```
   GPIOPadConfigSet(GPIO_PORTC_BASE,  GPIO_PIN_6,
GPIO_STRENGTH_2MA,  GPIO_PIN_TYPE_STD_WPU);
   GPIOPadConfigSet(GPIO_PORTC_BASE,  GPIO_PIN_7,
GPIO_STRENGTH_2MA,  GPIO_PIN_TYPE_STD_WPU);
   GPIOPadConfigSet(GPIO_PORTD_BASE,  GPIO_PIN_6,
GPIO_STRENGTH_2MA,  GPIO_PIN_TYPE_STD_WPU);
   GPIOPadConfigSet(GPIO_PORTA_BASE,  GPIO_PIN_3,
GPIO_STRENGTH_2MA,  GPIO_PIN_TYPE_STD_WPU);


   //loop infinito
   while (1){
      //Leelo todos los parqueos
      status3 = GPIOPinRead(GPIO_PORTD_BASE,  GPIO_PIN_6);
      status1 = GPIOPinRead(GPIO_PORTC_BASE,  GPIO_PIN_6);
      status2 = GPIOPinRead(GPIO_PORTC_BASE,  GPIO_PIN_7);
      status4 = GPIOPinRead(GPIO_PORTA_BASE,  GPIO_PIN_3);

      //Funcion para encender los LEDs de rojo/verde y los del DISPLAY
      LEDsparqueo();


      //Verifico si hay algun cambio
      if (status1prev!=status1){
         //Si si lo hay, actualizo mi variable previa
         status1prev=status1;
         //Si esta ocupado
         if (status1==0){
                  parqueo1=1;
                  UARTCharPut(UART1_BASE,  'a');
               }
         //Si esta libre
                  else {
                     parqueo1=0;
                     UARTCharPut(UART1_BASE,  'b');

                  }
      }
```

```
        if (status2prev!=status2){
            status2prev=status2;
            if (status2==0){
                parqueo2=1;
            //    GPIOPinWrite(GPIO_PORTF_BASE,
GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,  6); //Rojo
                UARTCharPut(UART1_BASE,  'c');

      }
      else {
          parqueo2=0;
          UARTCharPut(UART1_BASE,  'd');
      }
        }
        if (status3prev!=status3){
            status3prev=status3;
            if (status3==0){
                parqueo3=1;
                UARTCharPut(UART1_BASE,  'e');
           }
            else{
                parqueo3=0;
                UARTCharPut(UART1_BASE,  'f');
            }
        }

        if (status4prev!=status4){
                status4prev=status4;
                if (status4==0){
                    parqueo4=1;
                    UARTCharPut(UART1_BASE,  'g');

               }
                else{
                    parqueo4=0;
                    UARTCharPut(UART1_BASE,  'h');

                }
            }
```

```
        else{
          //  GPIOPinWrite(GPIO_PORTF_BASE,
GPIO_PIN_1|GPIO_PIN_2|GPIO_PIN_3,  0); //Rojo
        }
   }
}


void InitUART(void){
   /*Enable the GPIO Port A*/
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);

   /*Enable the peripheral UART Module 0*/
   SysCtlPeripheralEnable(SYSCTL_PERIPH_UART0);

   /* Make the UART pins be peripheral controlled. */
   GPIOPinTypeUART(GPIO_PORTA_BASE,  GPIO_PIN_0 | GPIO_PIN_1);

   /* Sets the configuration of a UART. */
   UARTConfigSetExpClk(UART0_BASE,  SysCtlClockGet(),
115200,(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE |
UART_CONFIG_PAR_NONE));

   //Habilitamos las interrupciones de UART
  //  UARTIntEnable(UART0_BASE,  UART_INT_RX | UART_INT_RT);

   //LLamamos  a la funcion de interrupcion
  //  UARTIntRegister(UART0_BASE  ,UARTIntHandler);
}

void InitUART1(void){
   /*Enable the GPIO Port A*/
   SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);

   /*Enable the peripheral UART Module 0*/
   SysCtlPeripheralEnable(SYSCTL_PERIPH_UART1);

   //Establecemos los pines RX/C4 y TX/C5
   GPIOPinConfigure(GPIO_PC5_U1TX);
```

```
    GPIOPinConfigure(GPIO_PC4_U1RX);

    /* Make the UART pins be peripheral controlled. */
    GPIOPinTypeUART(GPIO_PORTC_BASE, GPIO_PIN_4 | GPIO_PIN_5);


    /* Sets the configuration of a UART. */
    UARTConfigSetExpClk(UART1_BASE, SysCtlClockGet(),
115200,(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE |
UART_CONFIG_PAR_NONE));

}


void InitUART2(void){
    /*Enable the GPIO Port A*/
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);

    /*Enable the peripheral UART Module 0*/
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART2);

    /* Make the UART pins be peripheral controlled. */
    GPIOPinTypeUART(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_7);

    /* Sets the configuration of a UART. */
    UARTConfigSetExpClk(UART2_BASE, SysCtlClockGet(),
9600,(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE |
UART_CONFIG_PAR_NONE));


}

//Interrupcion cuando se recibe un dato, guardamos en una variable el valor
y cambiamos el color
void UARTIntHandler(){
    UARTIntClear (UART1_BASE, UART_INT_RX|UART_INT_TX);
    //UARTCharPut(UART0_BASE, 'r');

}
```

```
void LEDsparqueo(void){
   //Reseteo mi contador a 0
   contador=0;
   //Si esta ocupado, apago verde y automaticamente se enciende rojo
   if(parqueo3== 1 ){
      GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_3, 0);
   }
   //Si esta libre, enciende verde y automaticamente se apaga rojo
   else if (parqueo3==0){
      GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_3, GPIO_PIN_3);

   }
   if(parqueo4==1){
      GPIOPinWrite(GPIO_PORTB_BASE,GPIO_PIN_3, 0);

   }
   else if (parqueo4==0){
      GPIOPinWrite(GPIO_PORTB_BASE,GPIO_PIN_3, GPIO_PIN_3);

   }
   if(parqueo2==1){
      GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, 0);
   }
   else if (parqueo2==0){
      GPIOPinWrite(GPIO_PORTF_BASE, GPIO_PIN_2, GPIO_PIN_2);

   }
   if(parqueo1==1){
      GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_2, 0);
   }
   else if (parqueo1==0){
      GPIOPinWrite(GPIO_PORTB_BASE, GPIO_PIN_2, GPIO_PIN_2);

   }

   //Verifico cuantos parqueos estan ocupados
   contador=parqueo1+parqueo2+parqueo3+parqueo4;
   //Si estan todos libres, pongo 4
   if (contador==0){
```

```
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7,GPIO_PIN_6|GPIO_PIN_7);
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_0,GPIO_PIN_4|GPIO_PIN_5 );
        GPIOPinWrite(GPIO_PORTE_BASE,  GPIO_PIN_5|GPIO_PIN_4,0 );

    }
    //Pongo 3
    else if (contador==1){
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7,GPIO_PIN_6);
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_0,GPIO_PIN_4);
        GPIOPinWrite(GPIO_PORTE_BASE,
GPIO_PIN_5|GPIO_PIN_4,GPIO_PIN_4 );
    }
    //Pongo 2
    else if (contador==2){
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_7|GPIO_PIN_6|GPIO_PIN_5,GPIO_PIN_5);
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_1|GPIO_PIN_0,0 );
        GPIOPinWrite(GPIO_PORTE_BASE,
GPIO_PIN_5|GPIO_PIN_4,GPIO_PIN_4 );
    }
    //Pongo 1
    else if (contador==3){
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7,GPIO_PIN_6|GPIO_PIN_7);
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_0,GPIO_PIN_4|GPIO_PIN_5 );
        GPIOPinWrite(GPIO_PORTE_BASE,  GPIO_PIN_5|GPIO_PIN_4,0xFF
);
    }

    //Si estan todos ocupados, pongo 0
    else if (contador==4){
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7,0);
```

```
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_1|GPIO_PIN_0,0  );
        GPIOPinWrite(GPIO_PORTE_BASE,
GPIO_PIN_5|GPIO_PIN_4,GPIO_PIN_5  );
    }
    else {
        GPIOPinWrite(GPIO_PORTA_BASE,
GPIO_PIN_7|GPIO_PIN_6|GPIO_PIN_5,  0xFF);
        GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_4|GPIO_PIN_1|GPIO_PIN_0,0xFF  );
        GPIOPinWrite(GPIO_PORTE_BASE,  GPIO_PIN_5|GPIO_PIN_4,0xFF
);

    }
}
//Este delay sirve para poder  utilizar el delay que traen las librerias pero con
un
//valor exacto en ms
void delayMs(uint32_t  ui32Ms) {

    // 1 clock cycle = 1 / SysCtlClockGet() second
    // 1 SysCtlDelay = 3 clock cycle = 3 / SysCtlClockGet() second
    // 1 second = SysCtlClockGet() / 3
    // 0.001 second = 1 ms = SysCtlClockGet() / 3 / 1000

    SysCtlDelay(ui32Ms  * (SysCtlClockGet() / 3 / 1000));
}
```

Código Arduino ESP-32

```
//**************************************************************************************
***********
// Librerías
//**************************************************************************************
***********
#include <WiFi.h>
#include <WebServer.h>
#include <SPIFFS.h>
```

```
//*************************************************************************************
***********
// Variables globales
//*************************************************************************************
***********
// SSID & Password
const char* ssid = "TURBONETT_523";   // Enter your SSID here
const char* password = "a4c564b0c0";   //Enter your Password here


#define RXD2 16
#define TXD2 17

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

int dato = 0;
//Variables para indicar si un parqueo esta libre o ocupado
bool parqueo1 = HIGH;
bool parqueo2 = HIGH;
bool parqueo3 = HIGH;
bool parqueo4 = HIGH;

uint8_t LED1pin = 2;
bool LED1status = LOW;
int contador = 0;

//*************************************************************************************
***********
// Configuración
//*************************************************************************************
***********
void setup() {
  //Configuro mis seriales
  Serial.begin(115200);
  Serial2.begin(115200, SERIAL_8N1, RXD2, TXD2);


  while (!Serial);
  while (!Serial2);
```

```
  Serial.println("Try Connecting to ");
  Serial.println(ssid);

  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() {

  //Serial.println(contador);
  server.handleClient();
```

```
  ///Leo y guardo lo que ingresa por mi UART2
  if (Serial2.available()  > 0) {
    dato = Serial2.read();
    // Serial.println(Serial2.read());
    Serial.println(dato);
  }

//Para Parqueo 1
  if (dato == 97) { //Esta ocupado
    parqueo1  = LOW;
  }
  else if (dato == 98) {
    parqueo1  = HIGH; //Esta libre
  }

  //Para Parqueo 2
  if (dato == 99) {
    parqueo2  = LOW;
  }
  else if (dato == 100) {
    parqueo2  = HIGH;
  }

  //Para Parqueo 3
  if (dato == 101) {
    parqueo3  = LOW;
  }
  else if (dato == 102) {
    parqueo3  = HIGH;
  }

  //Para Parqueo 4
  if (dato == 103) {
    parqueo4  = LOW;
  }
  else if (dato == 104) {
    parqueo4  = HIGH;
  }
```

```
  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>Proyecto 4</title>\n";
  ptr += "<body bgcolor=#101010>";
  ptr += "<style>html { font-family: Helvetica; display: inline-block; margin:
0px auto; text-align: center;}\n";
  ptr += "body {margin-top: 50px;} h1 {color: #FFFFFF;margin: 50px auto
30px;}  h4 {color: #FFFFFF;margin:  15px;} h3 {color: #FFFFFF;margin-
bottom: 15px;}\n";
  ptr += "table {";
  ptr += "  font-family: Helvetica, sans-serif;";
  ptr += "  border-collapse: collapse;";
  ptr += "  width: 100%;";
  ptr += "  margin: 50px auto 30px;";
  ptr += "  text-align: center;";
  ptr += "}";
  ptr += "td, th {";
  ptr += " border: 1px solid #101010;";
  ptr += " text-align: center;";
  ptr += " padding: 8px;  ";
  ptr += "  background-color: #008080;";
  ptr += "}";
  ptr += "tr:nth-child(even) {";
  ptr += "  background-color: #FFFFFF;";
  ptr += "}  ";
  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 += "</style>\n";
  //Codigo para que la pagina se refresque cada 0.5 segundos
ptr += "<script>\n";
ptr += "<!--\n";
ptr += "function timedRefresh(timeoutPeriod)  {\n";
ptr += "\tsetTimeout(\"location.reload(true);\",timeoutPeriod);\n";
ptr += "}\n";
ptr += "\n";
ptr += "window.onload  = timedRefresh(500);\n";
ptr += "\n";
ptr += "//   -->\n";
ptr += "</script>";
  ptr += "</head>\n";
  ptr += "<body>\n";
  ptr += "<h1>ParqueoMatic-Prototipo &#128664</h1>\n";
  ptr += "<h3>Diego Mencos - 18300 </h3>\n";
  ptr += "<h4>Proyecto 4 </h1>\n";
  ptr += "<h2> </h2>";
  ptr += "<table>";
  ptr += " <table style= margin: 0 auto;>";
  ptr += " <tr>";
  ptr += "   <th><span style='font-size:30px;'>Parqueo 1</th>";
  ptr += "   <th><span style='font-size:30px;'>Parqueo 2 </th>";
  ptr += "   <th><span style='font-size:30px;'>Parqueo 3 </th>";
  ptr += "   <th><span style='font-size:30px;'>Parqueo 4</th>";
  ptr += " </tr>";
  ptr += "  <tr>";

  //ptr +=  "<td> <span style='font-size:60px;'> &#9989;</span></td>";
  //ptr += "<td> <span style='font-size:60px;'>&#10060;</span></td>";
  //ptr += "<td> <span style='font-size:60px;'> &#9989;</span></td>";
  //ptr +=" <td> <span style='font-size:60px;'>&#10060;</span></td>";

//Para Parqueo 1
  if (parqueo1 == HIGH) {
    ptr +=  "<td> <span style='font-size:60px;'> &#9989;</span></td>";
```

```
  }
  else if (parqueo1 == LOW) {
    ptr += "<td> <span style='font-size:60px;'>&#10060;</span></td>";
  }

//Para Parqueo 2
  if (parqueo2 == HIGH) {
   ptr += "<td> <span style='font-size:60px;'> &#9989;</span></td>";
  }
  else if (parqueo2 == LOW) {
    ptr += "<td> <span style='font-size:60px;'>&#10060;</span></td>";
  }

//Para Parqueo 3
  if (parqueo3 == HIGH) {
    ptr +=  "<td> <span style='font-size:60px;'>  &#9989;</span></td>";
  }
  else if (parqueo3 == LOW) {
    ptr += "<td> <span style='font-size:60px;'>&#10060;</span></td>";
  }

//Para Parqueo 4
  if (parqueo4 == HIGH) {
    ptr += "<td> <span style='font-size:60px;'> &#9989;</span></td>";
  }
  else if (parqueo4 == LOW) {
    ptr += "<td> <span style='font-size:60px;'>&#10060;</span></td>";
  }



  ptr += " </tr>";
  ptr += "</table>";



  if (led1stat)
  {
```

```
    ptr += "<p>Presione el boton para refrescar la pagina.</p><a
class=\"button button-off\" href=\"/led1off\">ACT</a>\n";
  }
  else
  {
    ptr += "<p>Presione el boton para refrescar la pagina.</p><a
class=\"button button-on\" href=\"/led1on\">ACT</a>\n";
  }

  ptr += "</body>\n";
  ptr += "</html>\n";
  return ptr;
}

//*******************************************************************************
***********
// Handler de not found
//*******************************************************************************
***********
void handle_NotFound() {
  server.send(404, "text/plain", "Not found");
}
```