Universidad del Valle de Guatemala
Natalia de León Bercián
Electrónica Digital 2
carne: 18193
Mayo 2021
sección: 20

Proyecto # 4

Control de registro para parqueos

Link video

https://youtu.be/c0t8WvCdW0w

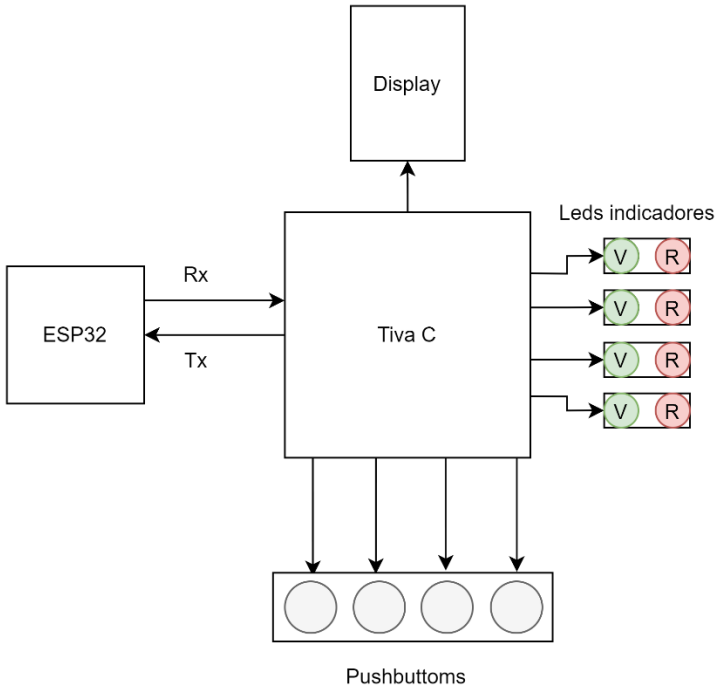
Link Github

https://github.com/nataliadlb/LABS_REPOSITORIO.git

Explicación de mi proyecto

Para simular los sensores se utilizó pushbuttoms, los cuales, si se presiona una vez, significa que está ocupado y si se vuelve a presionar se libera y así repetidamente, esto es posible observarlo en los leds indicadores pues van cambiando de color. En el display se despliega el número de parqueos disponibles. El webserver se actualiza cada 1 segundo y en este se muestran los 4 parqueos y su estado: Ocupado o libre.

Diagrama



Datos

Organización de los pines de leds indicadores y leds del 7 segmento

```
#define led_green1 GPIO_PIN_5 // PORT B
#define led_green2 GPIO_PIN_0 // PORT D
#define led_green3 GPIO_PIN_4 // PORT E
#define led_green4 GPIO_PIN_3 // PORT D
```

```
#define led_red1 GPIO_PIN_1 // PORT B
#define led_red2 GPIO_PIN_2 // PORT D
#define led_red3 GPIO_PIN_4 // PORT B
#define led_red4 GPIO_PIN_2 // PORT E
```

```
#define PUSH_1 GPIO_PIN_0 // PORT B
#define PUSH_2 GPIO_PIN_1 // PORT D
#define PUSH_3 GPIO_PIN_5 // PORT E
#define PUSH_4 GPIO_PIN_1 // PORT E

#define led_a GPIO_PIN_7 // PORT
#define led_b GPIO_PIN_4 // PORT A
#define led_c GPIO_PIN_7 // PORT C
#define led_d GPIO_PIN_6 // PORT C
#define led_e GPIO_PIN_3 // PORT A
#define led_f GPIO_PIN_6 // PORT D
#define led_g GPIO_PIN_3 // PORT B
#define led_dp GPIO_PIN_6 // PORT B
```

Código Code Composer

```
//*************************************************
/**
 Proyecto # 4
 Natalia de León Bericán
 carne: 18193
 Digital 2
 Sección 20
 Mayo 2021
*/
//*************************************************

#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "inc/hw_memmap.h"
#include "driverlib/debug.h"
#include "inc/hw_types.h"
#include "inc/hw_ints.h"
#include "driverlib/sysctl.h"
#include "driverlib/systick.h"
#include "driverlib/interrupt.h"
#include "driverlib/gpio.h"
#include "driverlib/timer.h"
#include "driverlib/uart.h"
#include "driverlib/rom.h"
#include "driverlib/pin_map.h"
#include <string.h>
#include "grlib/grlib.h"

#include "driverlib/fpu.h"
#include "inc/hw_uart.h"
#include "driverlib/rom_map.h"

#define XTAL 16000000
#define led_green1 GPIO_PIN_5 // PORT B
#define led_green2 GPIO_PIN_0 // PORT D
#define led_green3 GPIO_PIN_4 // PORT E
#define led_green4 GPIO_PIN_3 // PORT D

#define led_red1 GPIO_PIN_1 // PORT B
#define led_red2 GPIO_PIN_2 // PORT D
#define led_red3 GPIO_PIN_4 // PORT B
#define led_red4 GPIO_PIN_2 // PORT E

#define PUSH_1 GPIO_PIN_0 // PORT B
#define PUSH_2 GPIO_PIN_1 // PORT D
#define PUSH_3 GPIO_PIN_5 // PORT E
#define PUSH_4 GPIO_PIN_1 // PORT E

#define led_a GPIO_PIN_7 // PORT B
#define led_b GPIO_PIN_4 // PORT A
#define led_c GPIO_PIN_7 // PORT C
#define led_d GPIO_PIN_6 // PORT C
#define led_e GPIO_PIN_3 // PORT A
#define led_f GPIO_PIN_6 // PORT D
#define led_g GPIO_PIN_3 // PORT B
#define led_dp GPIO_PIN_6 // PORT B


//*************************************************
//
// VARIABLES
//
//*************************************************

uint8_t ban_ocupado = 0; //valor de parqueos
ocupados
uint8_t debouncing = 0; //debouncing de todos los
botones
uint8_t Aumento1 = 1; //bandera para saber si ya
se presiono push1
uint8_t Aumento2 = 1; //bandera para saber si ya
se presiono push2
uint8_t Aumento3 = 1; //bandera para saber si ya
se presiono push3
uint8_t Aumento4 = 1; //bandera para saber si ya
se presiono push4

//*************************************************
//
// PROTOTIPOS DE FUNCIONES
//
//*************************************************
void InitUART(void);
void DISPLAY(uint8_t num_display);
void delayMs(uint32_t ui32Ms);
void LEDS_G_R(void);
void Mostrar_display(void);

//*************************************************
//
// The error routine that is called if the driver
library encounters an error.
//
//*************************************************
#ifdef DEBUG
void
__error__(char *pcFilename, uint32_t ui32Line)
{
    while(1);
}
#endif

//*************************************************
//
// CONFIGURACION
//
//*************************************************

int main(void){

    //Config de reloj a 40MHz
```

```
    SysCtlClockSet(SYSCTL_SYSDIV_5 |
SYSCTL_USE_PLL | SYSCTL_XTAL_16MHZ |
SYSCTL_OSC_MAIN);

    // Config reloj para habilitar puertos
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);

    // Verificar que se habilitaron los puertos

while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOA){
     }
while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOB){
     }
while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOC){
     }
while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOD){
     }
while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOE){
     }
while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOF){
     }

    // Configurar OUTPUTS/INPUTS
    GPIOPinTypeGPIOOutput(GPIO_PORTA_BASE,
GPIO_PIN_3 | GPIO_PIN_4);
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE,
GPIO_PIN_1 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5
| GPIO_PIN_6 | GPIO_PIN_7);
    GPIOPinTypeGPIOOutput(GPIO_PORTC_BASE,
GPIO_PIN_6 | GPIO_PIN_7);
    GPIOPinTypeGPIOOutput(GPIO_PORTD_BASE,
GPIO_PIN_0 | GPIO_PIN_2 | GPIO_PIN_3 |
GPIO_PIN_6);
    GPIOPinTypeGPIOOutput(GPIO_PORTE_BASE,
GPIO_PIN_2 | GPIO_PIN_4);
    //GPIOPinTypeGPIOOutput(GPIO_PORTF_BASE,
GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3);

    GPIOPinTypeGPIOInput(GPIO_PORTB_BASE,
GPIO_PIN_0); //PUSH COMO ENTRADA
    GPIOPinTypeGPIOInput(GPIO_PORTD_BASE,
GPIO_PIN_1);
    GPIOPinTypeGPIOInput(GPIO_PORTE_BASE,
GPIO_PIN_1 | GPIO_PIN_5);

    GPIOPadConfigSet(GPIO_PORTB_BASE, GPIO_PIN_0,
GPIO_STRENGTH_2MA, GPIO_PIN_TYPE_STD_WPD); //weak
pull DOWN
    GPIOPadConfigSet(GPIO_PORTD_BASE, GPIO_PIN_1,
GPIO_STRENGTH_2MA, GPIO_PIN_TYPE_STD_WPD); //weak
pull DOWN
    GPIOPadConfigSet(GPIO_PORTE_BASE, GPIO_PIN_1,
GPIO_STRENGTH_2MA, GPIO_PIN_TYPE_STD_WPD); //weak
pull DOWN
    GPIOPadConfigSet(GPIO_PORTE_BASE, GPIO_PIN_5,
GPIO_STRENGTH_2MA, GPIO_PIN_TYPE_STD_WPD); //weak
pull DOWN

    /**
        CONFIGURACION UART
    */
    // Se inicializa la comunicación UART
    InitUART();

    //TODOS LOS PARQUEOS DISPONIBLES
    GPIOPinWrite(GPIO_PORTB_BASE, led_green1 ,
led_green1);
    GPIOPinWrite(GPIO_PORTD_BASE, led_green2 ,
led_green2);
    GPIOPinWrite(GPIO_PORTE_BASE, led_green3 ,
led_green3);
    GPIOPinWrite(GPIO_PORTD_BASE, led_green4 ,
led_green4);
    DISPLAY(4);
    ban_ocupado = 0;

//*************************************************
    // Loop forever.

//*************************************************

    while(1){
        LEDS_G_R(); // En donde lee si se
presionan los push
        Mostrar_display(); //Muestra en el display
el numero de parqueos libres
    }
}

//*************************************************
//
// FUNCIONES
//
//*************************************************

//*************************************************
// FUNCION PARA ENCENDER LEDS DEPENDIENDO SI ESTA
VACIO U OCUPADO
//*************************************************
void LEDS_G_R(void){
    if((GPIOPinRead(GPIO_PORTB_BASE, PUSH_1))){
        while(GPIOPinRead(GPIO_PORTB_BASE,
PUSH_1)){}              //debouncing push 1
        if (Aumento1 == 1){
            GPIOPinWrite(GPIO_PORTB_BASE,
led_green1 , 0);         //se apaga led verde
            GPIOPinWrite(GPIO_PORTB_BASE, led_red1
, led_red1);    //se enciende led rojo
            ban_ocupado++;
//aumenta valor de parqueos ocupados
            debouncing = 1;
//bandera de debouncing
            Aumento1 = 0;
//control de que si ya se apacho una vez el boton
            UARTCharPut(UART1_BASE, '0');
//se manda el caracter por UART
        }
        else if (Aumento1 == 0){
            GPIOPinWrite(GPIO_PORTB_BASE,
led_green1 , led_green1); //se enciende led verde
            GPIOPinWrite(GPIO_PORTB_BASE, led_red1
, 0);            //se apaga led rojo
            ban_ocupado--;
//se disminuye variable de parqueos ocupados
            debouncing = 1;
//bandera debouncing
            Aumento1 = 1;
//bandera que significa que se presiono por
segunda vez el boton
            UARTCharPut(UART1_BASE, '1');
//se manda valor a UART
```

```
        }

    }
    else if((GPIOPinRead(GPIO_PORTD_BASE,
PUSH_2))){
        while(GPIOPinRead(GPIO_PORTD_BASE,
PUSH_2)){}               //debouncing push2
        if (Aumento2 == 1){
            GPIOPinWrite(GPIO_PORTD_BASE,
led_green2 , 0);          //OFF green
            GPIOPinWrite(GPIO_PORTD_BASE, led_red2
, led_red2);     //ON red
            ban_ocupado++;
            debouncing = 1;
            Aumento2 = 0;
            UARTCharPut(UART1_BASE, '2');
//se manda valor a UART
        }
        else if (Aumento2 == 0){
            GPIOPinWrite(GPIO_PORTD_BASE,
led_green2 , led_green2); //ON green
            GPIOPinWrite(GPIO_PORTD_BASE, led_red2
, 0);           //OFF red
            ban_ocupado--;
            debouncing = 1;
            Aumento2 = 1;
            UARTCharPut(UART1_BASE, '3');
//se manda valor a UART
        }

    }
    else if((GPIOPinRead(GPIO_PORTE_BASE,
PUSH_3))){
        while(GPIOPinRead(GPIO_PORTE_BASE,
PUSH_3)){}               //debouncing push 3

        if (Aumento3 == 1){
            GPIOPinWrite(GPIO_PORTE_BASE,
led_green3 , 0);
            GPIOPinWrite(GPIO_PORTB_BASE, led_red3
, led_red3);
            ban_ocupado++;
            debouncing = 1;
            Aumento3 = 0;
            UARTCharPut(UART1_BASE, '4');
//se manda valor a UART

        }
        else if (Aumento3 == 0){
            GPIOPinWrite(GPIO_PORTE_BASE,
led_green3 , led_green3);
            GPIOPinWrite(GPIO_PORTB_BASE, led_red3
, 0);
            ban_ocupado--;
            debouncing = 1;
            Aumento3 = 1;
            UARTCharPut(UART1_BASE, '5');
//se manda valor a UART
        }


    }
    else if((GPIOPinRead(GPIO_PORTE_BASE,
PUSH_4))){
        while(GPIOPinRead(GPIO_PORTE_BASE,
PUSH_4)){}               //debouncing push 4
        if (Aumento4 == 1){
            GPIOPinWrite(GPIO_PORTD_BASE,
led_green4 , 0);
            GPIOPinWrite(GPIO_PORTE_BASE, led_red4
, led_red4);
            ban_ocupado++;
            debouncing = 1;
            Aumento4 = 0;
            UARTCharPut(UART1_BASE, '6');
//se manda valor a UART
        }
        else if (Aumento4 == 0){
            GPIOPinWrite(GPIO_PORTD_BASE,
led_green4 , led_green4);
            GPIOPinWrite(GPIO_PORTE_BASE, led_red4
, 0);
            ban_ocupado--;
            debouncing = 1;
            Aumento4 = 1;
            UARTCharPut(UART1_BASE, '7');
//se manda valor a UART
        }
    }

}

//*********************************************
// FUNCION PARA ENCENDER LEDS DEPENDIENDO SI ESTA
VACIO U OCUPADO
//*********************************************
void Mostrar_display(void){

    if (debouncing == 1 &&
(GPIOPinRead(GPIO_PORTE_BASE, PUSH_4) == 0) &&
(GPIOPinRead(GPIO_PORTE_BASE, PUSH_3) == 0) &&
(GPIOPinRead(GPIO_PORTD_BASE, PUSH_2) == 0) &&
(GPIOPinRead(GPIO_PORTB_BASE, PUSH_1) == 0)){
        if (ban_ocupado == 0){
            DISPLAY(5); //Se apagan todos
            DISPLAY(4);
            debouncing = 0;
        }
        else if (ban_ocupado == 1){ // Si hay un
parqueo ocupado
            DISPLAY(5);
            DISPLAY(3);
            debouncing = 0;
        }
        else if (ban_ocupado == 2){ // Si hay dos
parqueo ocupado
            DISPLAY(5);
            DISPLAY(2);
            debouncing = 0;
        }
        else if (ban_ocupado == 3){ // Si hay tres
parqueo ocupado
            DISPLAY(5);
            DISPLAY(1);
            debouncing = 0;
        }
        else if (ban_ocupado == 4){ // Si hay
cuatro parqueo ocupado
            DISPLAY(5);
            DISPLAY(0);
            debouncing = 0;
        }
    }
}
//*********************************************
```

```
// FUNCION DE NUMEROS PARA DISPLAY
//*****************************************
void DISPLAY(uint8_t num_display){
    switch(num_display){
        case 0:
            GPIOPinWrite(GPIO_PORTB_BASE, led_a ,
led_a);
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
led_b);
            GPIOPinWrite(GPIO_PORTC_BASE, led_c ,
led_c);
            GPIOPinWrite(GPIO_PORTC_BASE, led_d ,
led_d);
            GPIOPinWrite(GPIO_PORTA_BASE, led_e ,
led_e);
            GPIOPinWrite(GPIO_PORTD_BASE, led_f ,
led_f);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
led_dp);
            break;
        case 1:
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
led_b);
            GPIOPinWrite(GPIO_PORTC_BASE, led_c ,
led_c);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
led_dp);
            break;
        case 2:
            GPIOPinWrite(GPIO_PORTB_BASE, led_a ,
led_a);
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
led_b);
            GPIOPinWrite(GPIO_PORTC_BASE, led_d ,
led_d);
            GPIOPinWrite(GPIO_PORTA_BASE, led_e ,
led_e);
            GPIOPinWrite(GPIO_PORTB_BASE, led_g ,
led_g);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
led_dp);
            break;
        case 3:
            GPIOPinWrite(GPIO_PORTB_BASE, led_a ,
led_a);
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
led_b);
            GPIOPinWrite(GPIO_PORTC_BASE, led_c ,
led_c);
            GPIOPinWrite(GPIO_PORTC_BASE, led_d ,
led_d);
            GPIOPinWrite(GPIO_PORTB_BASE, led_g ,
led_g);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
led_dp);
            break;
        case 4:
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
led_b);
            GPIOPinWrite(GPIO_PORTC_BASE, led_c ,
led_c);
            GPIOPinWrite(GPIO_PORTD_BASE, led_f ,
led_f);
            GPIOPinWrite(GPIO_PORTB_BASE, led_g ,
led_g);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
led_dp);
            break;
        case 5:
            GPIOPinWrite(GPIO_PORTB_BASE, led_a,
0);
            GPIOPinWrite(GPIO_PORTA_BASE, led_b ,
0);
            GPIOPinWrite(GPIO_PORTC_BASE, led_c ,
0);
            GPIOPinWrite(GPIO_PORTC_BASE, led_d ,
0);
            GPIOPinWrite(GPIO_PORTA_BASE, led_e ,
0);
            GPIOPinWrite(GPIO_PORTD_BASE, led_f ,
0);
            GPIOPinWrite(GPIO_PORTB_BASE, led_g ,
0);
            GPIOPinWrite(GPIO_PORTB_BASE, led_dp ,
0);
            break;
    }
}
//*****************************************
// Inicialización de UART - Tomado del ejemplo que
subio Pablo Mazariegos
//*****************************************
void InitUART(void){

    /*Enable the peripheral UART Module 2*/
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART1);


while(!SysCtlPeripheralReady(SYSCTL_PERIPH_UART1))
{
    }
    /*Enable the GPIO Port d*/
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);

    GPIOPinConfigure(GPIO_PC4_U1RX);
    GPIOPinConfigure(GPIO_PC5_U1TX);

    // Se habilitan las interrupciones Globales
    IntMasterEnable();

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

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

    IntEnable (INT_UART1);

    UARTIntEnable(UART1_BASE, UART_INT_RX |
UART_INT_RT);
    UARTEnable (UART1_BASE);
}
```

Código ESP32

```
/*************************************************
*************************************************

/**

 Proyecto # 4

 Natalia de León Bericán

 carne: 18193

 Digital 2

 Sección 20

 Mayo 2021

*/

//************************************************

// Librerías

//************************************************

#include <WiFi.h>

#include <WebServer.h>

#include <stdint.h>

#include <stdio.h>

#include "HardwareSerial.h"

//************************************************

// Variables globales

//************************************************

// SSID & Password

const char* ssid = "CASA WIFI 2";  // Enter your SSID here

const char* password = "casa2020";  //Enter your Password here

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

int Val = 57; // for incoming serial data

int p1 = 0;

int p2 = 0;

int p3 = 0;

int p4 = 0;

//************************************************
*************************************************

// Configuración

//************************************************
*************************************************

void setup() {

 Serial.begin(115200);

 Serial2.begin(115200, SERIAL_8N1, 16, 17);

 Serial.println("Try Connecting to ");

 Serial.println(ssid);

 // 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.onNotFound(handle_NotFound);

 server.begin();

 Serial.println("HTTP server started");

 delay(100);

}

//************************************************

// loop principal

//************************************************

void loop() {

 server.handleClient();

 // reply only when you receive data:
```

```
  if (Serial2.available() > 0) {
    // read the incoming byte:
    Val = Serial2.read();
  }
}
//*************************************************
// Handler de Inicio página
//*************************************************
void handle_OnConnect() {
  //Serial.println("INICIO");
  server.send(200, "text/html", SendInicioHTML(Val));
}
//*************************************************
// Procesador de HTML
//*************************************************
String SendInicioHTML(uint8_t Valor) {
  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>Parqueos</title>\n";
  ptr += "<style>html { font-family: Helvetica; display: inline-block; margin: 50px auto; text-align: center;}\n";
  ptr += ".column {float: left ; width: 25%; pading: 10px}\n";
  ptr += "row:after {content:""; display: table; clear: both}\n";
  ptr += "</style>\n";
  ptr += "</head>\n";

  ptr += "<body>\n";
  ptr += "<h1 style=\"font-size:48px\">Control de parqueos &#128663</h1>\n";
  ptr += "<div class=\"row\">\n"; // crear 4 columnas para los titulos de los parqueos

    ptr += "<div class=\"column\">\n";
    ptr += "<h2 style=\"font-size:32px\"> PARQUEO 1 </h2>\n";
    if (Valor == 48){ //si se ingresa el valor 0, quiere decir que esta ocupado P1
      p1 = 1;
      }
    if (Valor == 49){
      p1 = 0;
      }
    if (p1 == 0){
      p1 = 0;
      ptr += "<p style=\"font-size:25px\">LIBRE</p>\n";
      ptr += "<p style=\"font-size:25px\">&#9989</p>\n";
      }
    if (p1 == 1){
      p1 = 1;
      ptr += "<p style=\"font-size:25px\">OCUPADO</p>\n";
      ptr += "<p style=\"font-size:25px\">&#10060</p>\n";
      }
   ptr += "</div>";
   ptr += "<div class=\"column\">\n";
    ptr += "<h2 style=\"font-size:32px\"> PARQUEO 2 </h2>\n";
    if (Valor == 50 ){
      p2 = 1;
      }
    if (Valor == 51){
      p2 = 0;
      }
    if (p2 == 0){
      p2 = 0;
      ptr += "<p style=\"font-size:25px\">LIBRE</p>\n";
      ptr += "<p style=\"font-size:25px\">&#9989</p>\n";
      }
    if (p2 == 1){
```

```
    p2 = 1;
    ptr += "<p style=\"font-size:25px\">OCUPADO</p>\n";
    ptr += "<p style=\"font-size:25px\">&#10060</p>\n";
    }
  ptr += "</div>";
  ptr += "<div class=\"column\">\n";
   ptr += "<h2 style=\"font-size:32px\"> PARQUEO 3 </h2>\n";
    if (Valor == 52){
     p3 = 1;
     }
    if (Valor == 53){
     p3 = 0;
     }
    if (p3 == 0){
     p3 = 0;
     ptr += "<p style=\"font-size:25px\">LIBRE</p>\n";
     ptr += "<p style=\"font-size:25px\">&#9989</p>\n";
     }
    if (p3 == 1){
     p3 = 1;
     ptr += "<p style=\"font-size:25px\">OCUPADO</p>\n";
     ptr += "<p style=\"font-size:25px\">&#10060</p>\n";
     }
  ptr += "</div>";

  ptr += "<div class=\"column\">\n";
   ptr += "<h2 style=\"font-size:32px\"> PARQUEO 4 </h2>\n";
    if (Valor == 54){
     p4 = 1;
     }
    if (Valor == 55){
     p4 = 0;
     }
    if (p4 == 0){
     p4 = 0;
     ptr += "<p style=\"font-size:25px\">LIBRE</p>\n";
     ptr += "<p style=\"font-size:25px\">&#9989</p>\n";
     }
    if (p4 == 1){
     p4 = 1;
     ptr += "<p style=\"font-size:25px\">OCUPADO</p>\n";
     ptr += "<p style=\"font-size:25px\">&#10060</p>\n";
     }
   ptr += "</div>";
  ptr += "</div>";
  // refresh
  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(1000);\n";
  ptr += "\n";
  ptr += "//  -->\n";
  ptr += "</script>\n";
  ptr += "</body>\n";
  ptr += "</html>\n";
  return ptr;
}
//**********************************************
// Handler de not found
//**********************************************
void handle_NotFound() {
  server.send(404, "text/plain", "Not found");
}
```