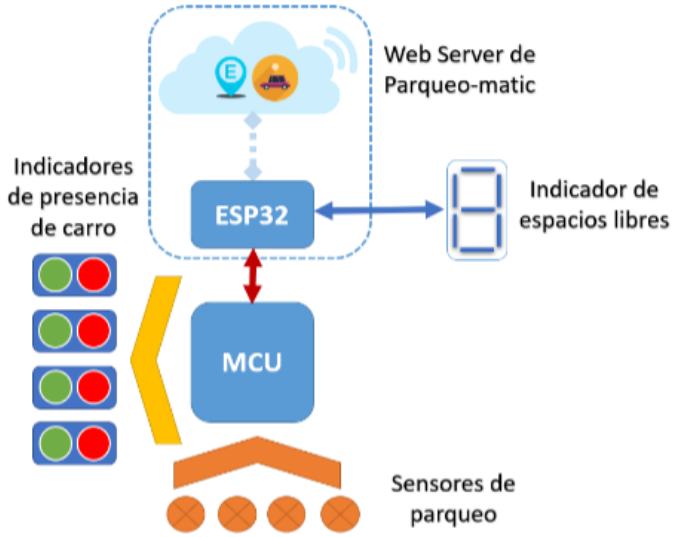
Electronica Digital II
Sección: 20
Mario Estrada - 18123

Proyecto 4

- Circuito Utilizado:



- Explicacion:

La Tiva C fue utilizada como MCU en este circuito. Para implementar los sensores en este caso se usaron botones en configuración Pull Down, que se encontraban conectados a la Tiva. Si el botón se encontraba sin presionar se tomó como si el parqueo estuviera vacío. Se utilizaron leds como indicadores de parque, rojo para indicar que el parqueo estaba ocupado y verde para indicar que estaba disponible. Al presionar un botón, la señal ingresa a la Tiva, la cual envía una señal indicando el apagado del indicador de disponibilidad y encender el indicador de ocupado.

El ESP32 es el encargado de crear el WebServer y dar señal al indicador de espacios disponibles. El ESP32 estaba comunicado a la Tiva C por medio UART, la Tiva envía los estados de los botones y el ESP32 envía los datos al servidor. Tomando en cuenta los botones que no se están presionando,

estos se convertían en un valor hexadecimal el cual es enviado al puerto donde está conectado e imprime el valor en el display 7 segmentos.

- Codigo CodeComposer:

```
//Proyecto 4
//Mario Estrada


#include <stdint.h>
#include <stdbool.h>
#include "inc/tm4c123gh6pm.h"
#include "inc/hw_memmap.h"
#include "inc/hw_types.h"
#include "inc/hw_ints.h"
#include "inc/hw_gpio.h"
#include "driverlib/sysctl.h"
#include "driverlib/interrupt.h"
#include "driverlib/gpio.h"
#include "driverlib/timer.h"
#include "driverlib/uart.h"
#include "driverlib/pin_map.h"

#define XTAL 16000000

// Variables
uint32_t i = 0;
uint32_t j = 0;
uint32_t Total = 0;
uint32_t P1 = 0;
uint32_t P2 = 0;
uint32_t P3 = 0;
uint32_t P4 = 0;
uint32_t Ocupado = 0;
uint8_t Parqueos = 0;

void disp(uint32_t numero);

//Main
int main(void)
 {
    // Seteado del reloj
    SysCtlClockSet(SYSCTL_SYSDIV_1 | SYSCTL_USE_OSC | SYSCTL_OSC_MAIN|
SYSCTL_XTAL_16MHZ);

SysCtlClockSet(SYSCTL_SYSDIV_5|SYSCTL_USE_PLL|SYSCTL_XTAL_16MHZ|SYSCTL_
OSC_MAIN);

    // Seteado de perifericos
```

```
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);


    // Se configuran los puertos como entradas y salidas
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0);

    GPIOPinTypeGPIOInput(GPIO_PORTC_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7);
    GPIOPadConfigSet(GPIO_PORTC_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7, GPIO_STRENGTH_8MA,
GPIO_PIN_TYPE_STD_WPD);

    GPIOPinTypeGPIOOutput(GPIO_PORTD_BASE,
GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0);
    HWREG(GPIO_PORTD_BASE + GPIO_O_LOCK) = GPIO_LOCK_KEY;
    HWREG(GPIO_PORTD_BASE + GPIO_O_CR) |= GPIO_PIN_7;
    GPIOPinConfigure(GPIO_PD7_U2TX);

    GPIOPinTypeGPIOOutput(GPIO_PORTE_BASE,
GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0);


    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART2);
    GPIOPinTypeUART(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_7);
    UARTConfigSetExpClk(UART2_BASE, SysCtlClockGet(), 115200,
(UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE | UART_CONFIG_PAR_NONE));
    UARTIntClear(UART2_BASE, UART_INT_RX | UART_INT_RT | UART_INT_TX |
UART_INT_FE | UART_INT_PE | UART_INT_BE | UART_INT_OE | UART_INT_RI |
UART_INT_CTS | UART_INT_DCD | UART_INT_DSR);


//Loop

    while (1)
    {
        P1 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_4) && GPIO_PIN_4);
        P2 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_5) && GPIO_PIN_5);
        P3 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_6) && GPIO_PIN_6);
        P4 = (GPIOPinRead(GPIO_PORTC_BASE, GPIO_PIN_7) && GPIO_PIN_7);

        Parqueos = GPIOPinRead(GPIO_PORTC_BASE,
GPIO_PIN_4|GPIO_PIN_5|GPIO_PIN_6|GPIO_PIN_7) >> 4;
        Ocupado = P4+P3+P2+P1;
        Total = (4 - Ocupado) ;
        disp(Total);

```

```
        GPIOPinWrite(GPIO_PORTD_BASE,
GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, Parqueos);
        GPIOPinWrite(GPIO_PORTE_BASE,
GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_PIN_0, ~Parqueos);


        UARTCharPut(UART2_BASE, P1);
        UARTCharPut(UART2_BASE, P2);
        UARTCharPut(UART2_BASE, P3);
        UARTCharPut(UART2_BASE, P4);
        UARTCharPut(UART2_BASE, 10);
    }

}


//Funcion 7 segmentos
void disp(uint32_t numero)
{
    //Imprime el numero en el puerto B dependiendo del caso
    switch(numero)
    {
    case 0: GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0, 0x3F); break;
    case 1: GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0, 0x06); break;
    case 2: GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0, 0x5B); break;
    case 3: GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0, 0x4F); break;
    case 4: GPIOPinWrite(GPIO_PORTB_BASE,
GPIO_PIN_6|GPIO_PIN_5|GPIO_PIN_4|GPIO_PIN_3|GPIO_PIN_2|GPIO_PIN_1|GPIO_
PIN_0, 0x66); break;
    default: break;
    }
}
```

- Codigo Arduino:

```
// Proyecto 4
// Mario Estrada - 18123
/**********
  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 = "TURBONETT_4BF538";  // Enter your SSID here
const char* password = "859B594874";  //Enter your Password here
WebServer server(80);  // Object of WebServer(HTTP port, 80 is defult)
uint8_t Push[4];
uint8_t Dispon = 0;
uint8_t LED1pin = 2;
bool LED1status = LOW;


void setup() {
  Serial.begin(115200);
  Serial2.begin(115200);
  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);
  server.on("/datos", handle_Data); // handler para enviar datos
  server.onNotFound(handle_NotFound);
  server.begin();
  Serial.println("HTTP server started");
  delay(100);
}
```

```
void loop() {
  server.handleClient();
  if (Serial2.available())
    {
      digitalWrite(2, 1);
        for (int i = 0; i<=3; i++)
        {
          Push[i] = Serial2.read();
        }
        Dispon = (((4 - Push[0])- Push[1])- Push[2])- Push[3];
    }
    digitalWrite(2, 0);
}


void handle_OnConnect() {
  server.send(200, "text/html", SendHTML());
}


void handle_Data() {
  server.send(200, "application/json", data_json());
}


String SendHTML(void) {
  String ptr = "<!DOCTYPE html>\n";
  ptr = "<html>\n";
  ptr += "<body>\n";
 // Formato del servidor
  ptr +="<body>\n";
  ptr +="<body style=\"background-color:#F0FFF0;\">\n";
  ptr +="</body>\n";

  ptr +="<style>\n";
  ptr +="h1 {\n";
  ptr +="  text-align: center;\n";
  ptr +="  font-size: 300%;\n";
  ptr +="}\n";
  ptr +="h3 {\n";
  ptr +="  text-align: center;\n";
  ptr +="  font-size: 100%;\n";
  ptr +="}\n";
  ptr += "table {\n";
  ptr += "  width: 50%;\n";
  ptr += "  height: 200px;\n";
```

```
  ptr += "}\n";
  ptr +="</style>\n";
  ptr += "<h1>Control de estacionamientos</h1>\n";

//Tabla de control de parqueos

  ptr += "<table border=\"1\" style=\"margin: 0 auto\"; \n";
  ptr += "  <tr>\n";
  ptr += "    <th># de parqueo</th>\n";
  ptr += "    <th>Parqueo 1</th>\n";
  ptr += "    <th>Parqueo 2</th>\n";
  ptr += "    <th>Parqueo 3</th>\n";
  ptr += "    <th>Parqueo 4</th>\n";
  ptr += "  </tr>\n";
  ptr += "  <tr>\n";
  ptr += "    <td>Disponibilidad</td>\n";
  ptr  +=  "        <td><canvas  id=\"Parqueo  1\"  width=\"200\"  height=\"100\"  style=\"border:0px  solid
#000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "</td>\n";
  ptr  +=  "        <td><canvas  id=\"Parqueo  2\"  width=\"200\"  height=\"100\"  style=\"border:0px  solid
#000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "</td>\n";
  ptr  +=  "        <td><canvas  id=\"Parqueo  3\"  width=\"200\"  height=\"100\"  style=\"border:0px  solid
#000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "</td>\n";
  ptr  +=  "        <td><canvas  id=\"Parqueo  4\"  width=\"200\"  height=\"100\"  style=\"border:0px  solid
#000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "</td>\n";
  ptr += "  </tr>\n";
  ptr += "</table>";
  ptr += "<h3>Verde = Disponible</h3>\n";
  ptr += "<h3>Rojo = Ocupado</h3>\n";
  ptr += "<canvas id=\"Cantidad\" width=\"900\" height=\"60\" style=\"border:0px solid #000000;\">\n";
  ptr += "</canvas>\n";
  ptr += "<script>\n";

  //Funcion de control de botones
  ptr += "function boton(n_parqueo, valor){\n";
  ptr += "var canvas = document.getElementById(n_parqueo);\n";
  ptr += "var ctx = canvas.getContext(\"2d\");\n";
  ptr += "if (valor == 0){\n";
  ptr += "ctx.fillStyle = \"#7CFC00\";\n";
  ptr += "};\n";
```

```
  ptr += "if (valor == 1){\n";
  ptr += "ctx.fillStyle = \"#fc0303\";\n";
  ptr += "};\n";
  ptr += "ctx.fillRect(0,0,200,100);\n";
  ptr += "ctx.fillStyle = \"#000000\";\n";
  ptr += "};\n";

 //Funcion de lugares disponibles
  ptr += "function Estacionamiento(cantidad){\n";
  ptr += "var canvas = document.getElementById(\"Cantidad\");\n";
  ptr += "var ctx = canvas.getContext(\"2d\");\n";
  ptr += "ctx.fillStyle = \"#F0FFF0\";\n";
  ptr += "ctx.fillRect(0,0,825,40);\n";
  ptr += "ctx.fillStyle = \"#000000\";\n";
  ptr += "};\n";

//Funcion de envio al WebServer
  ptr += "Estacionamiento(4);\n";
  ptr += "boton(\"Parqueo 1\", 0);\n";
  ptr += "boton(\"Parqueo 2\", 0);\n";
  ptr += "boton(\"Parqueo 3\", 0);\n";
  ptr += "boton(\"Parqueo 4\", 0);\n";
  ptr += "var sendHttpRequest = function(){\n";
  ptr += "var xhr = new XMLHttpRequest();\n";
  ptr += "xhr.open(\"GET\", \"http://192.168.1.2/datos\");\n";
  ptr += "xhr.responseType = \'json\';\n";

//Asignacion de botones a cuadros de parqueo
  ptr += "xhr.onload = function() {\n";
  ptr += "  console.log(xhr.response);\n";
  ptr += "Estacionamiento(xhr.response.Espacios);\n";
  ptr += "boton(\"Parqueo 1\", xhr.response.lugar1);\n";
  ptr += "boton(\"Parqueo 2\", xhr.response.lugar2);\n";
  ptr += "boton(\"Parqueo 3\", xhr.response.lugar3);\n";
  ptr += "boton(\"Parqueo 4\", xhr.response.lugar4);\n";
  ptr += "};\n";
  ptr += "xhr.send();\n";
  ptr += "return xhr.response;\n";
  ptr += "};\n";
  ptr += "setInterval(function(){\n";
  ptr += "sendHttpRequest();\n";
  ptr += "},1);\n";
  ptr += "</script>\n";
  ptr += "</body>\n";
  ptr += "</html>\n";
  return ptr;
}
```

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