

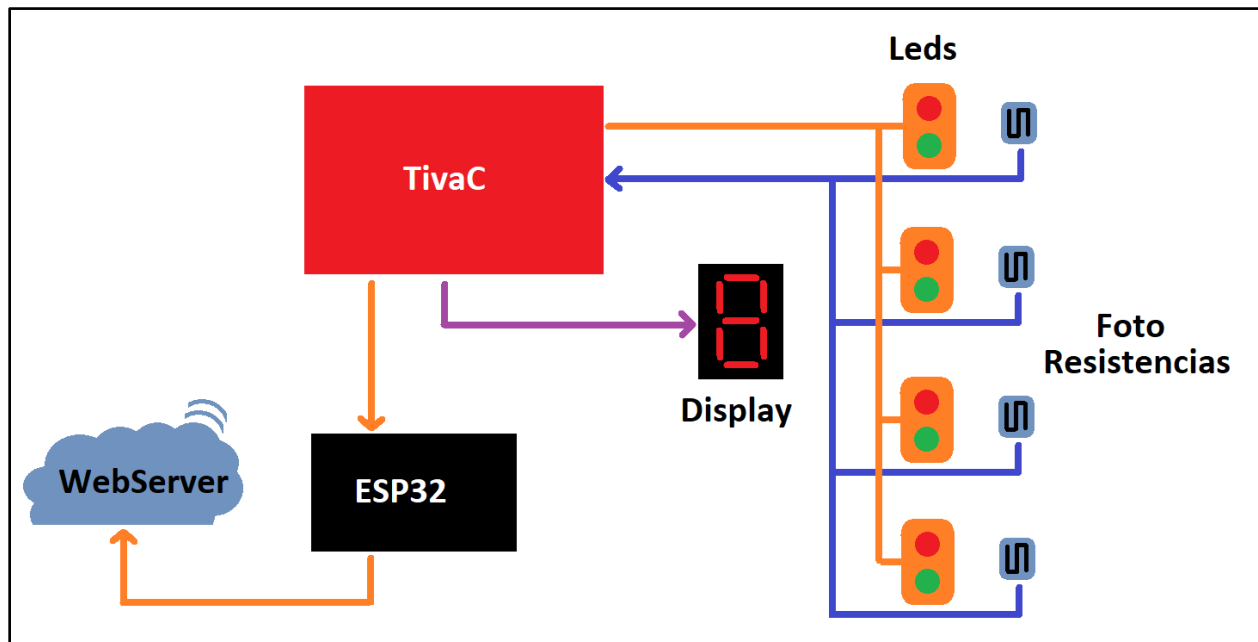
## PROYECTO No. 4 - PARQUEO-SPACE

Este proyecto consiste en el desarrollo de un circuito de un parqueo para cuatro vehículos, los cuales deben de ser leídos por sensores que determinen si hay disponibilidad de parqueo. Para el desarrollo de este proyecto se tiene como objetivo poner en practica los conocimientos obtenidos en el curso de Digital, los cuales consisten en el uso del microcontrolador TivaC usando código en C++ y el programa CodeComposer de la compañía Texas Instruments y conectarlo al microcontrolador ESP32 que es usado como un Web-Server para mandar los datos obtenidos a la nube.

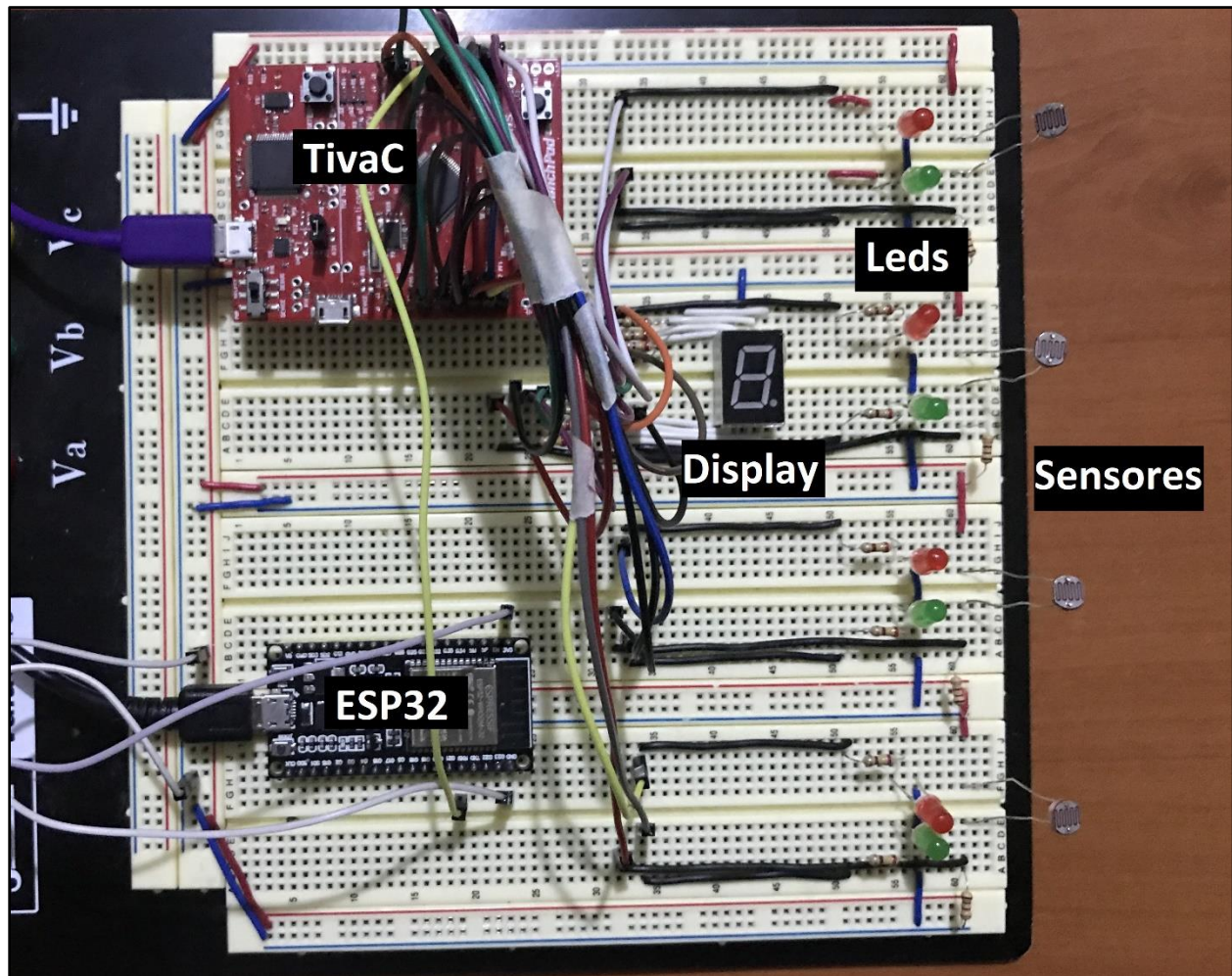
### 1. Circuito TIVA y ESP32

Para mostrar la estructura del proyecto, se presenta a continuación la estructura del diagrama, en el cual se conecta a la Tiva-C los leds como salida para indicar al usuario si el parqueo esta disponible o no. Asimismo, se conecta a la salida de la Tiva-C el display de 7 segmentos para mostrar la cantidad de parqueos total disponible. Como Entrada para la Tiva-C se conectan los foto-resistores, los cuales indican al presencia de luz o no.

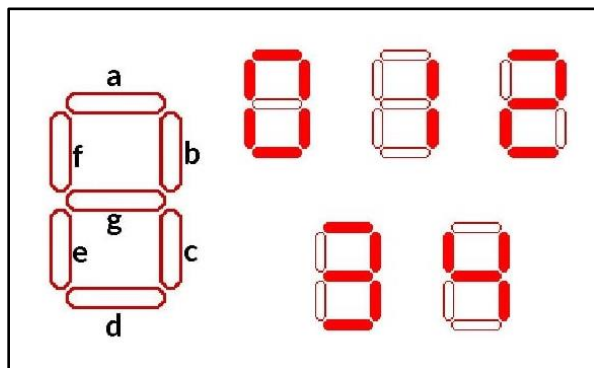
De parte del ESP32, se conecta un cable del **Tx** de la Tiva-C al **Rx** de este microcontrolador para recibir la comunicación UART a un Baud-rate de 115,200. Luego este con su modulo Wi-Fi sube los datos obtenidos al Web-Server indicando los parqueos disponibles.



El circuito físico se muestra a continuación, donde se usaron varios Jumpers y resistencias respectivas para suministrar la corriente necesaria al circuito. El voltaje se obtuvo del ESP32, y las tierras de ambos microprocesadores se *puntearon* para que existiera una sincronización para los leds, display y foto-resistencias.



## 2. Display



Para el 7 Segmentos se encendieron y apagaron los respectivos leds de la configuración para mostrar la numeración que se deseaba. Esto se realizó con la programación en la TivaWare y dependiendo del valor obtenido del total de parqueos disponibles, se coloca el numero correspondiente. Como son leds, al igual que los que muestran la disponibilidad de parqueos, se les colocó una resistencia de  $230\Omega$  a cada pin de salida de la Tiva-C.

### 3. Circuito Foto-Resistor

Para la foto-resistencia se procedió a usar el manual de la Tiva-C, la Tabla 24-6. En esta se determinó que el voltaje correspondiente de salida del sensor sería un 1 lógico o 0 lógico dependiendo del valor de entrada del pin:

- Para el 0 lógico es el  $V_{in} \cdot 0.35$ , es decir 1.15V.
- Para el 1 lógico es el  $V_{in} \cdot 0.65$ , es decir 2.27V.

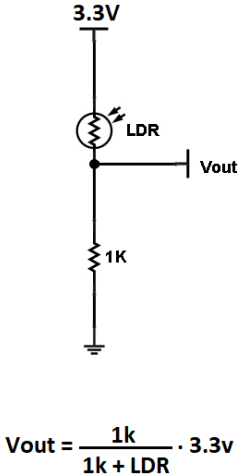
Por esta razón es que se usó una resistencia de 1K para una fotoresistencia de 100K, la cual al recibir luz permite el paso de la electricidad, y por lo tanto se convierte en un cable, por lo que según la ecuación mostrada a continuación el voltaje de salida queda:

- Si hay luz, la resistencia LDR es casi cero, por lo que el voltaje  $V_{out} = 3.5V$ .
- Si no hay luz, la resistencia LDR es grande, por lo que el voltaje  $V_{out} = 0V$ .

**Table 24-6. Recommended GPIO Pad Operating Conditions**

Parameter	Parameter Name	Min	Nom	Max	Unit
$V_{IH}$	GPIO high-level input voltage	$0.65 \cdot V_{DD}$	-	5.5	V
$V_{IL}$	GPIO low-level input voltage	0	-	$0.35 \cdot V_{DD}$	V
$V_{HYS}$	GPIO input hysteresis	0.2	-	-	V
$V_{OH}$	GPIO high-level output voltage	2.4	-	-	V
$V_{OL}$	GPIO low-level output voltage	-	-	0.4	V
$I_{OH}$	High-level source current, $V_{OH}=2.4 V^a$				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
$I_{OL}$	Low-level sink current, $V_{OL}=0.4 V^a$				
	2-mA Drive	2.0	-	-	mA
	4-mA Drive	4.0	-	-	mA
	8-mA Drive	8.0	-	-	mA
	8-mA Drive, $V_{OL}=1.2 V$	18.0	-	-	mA

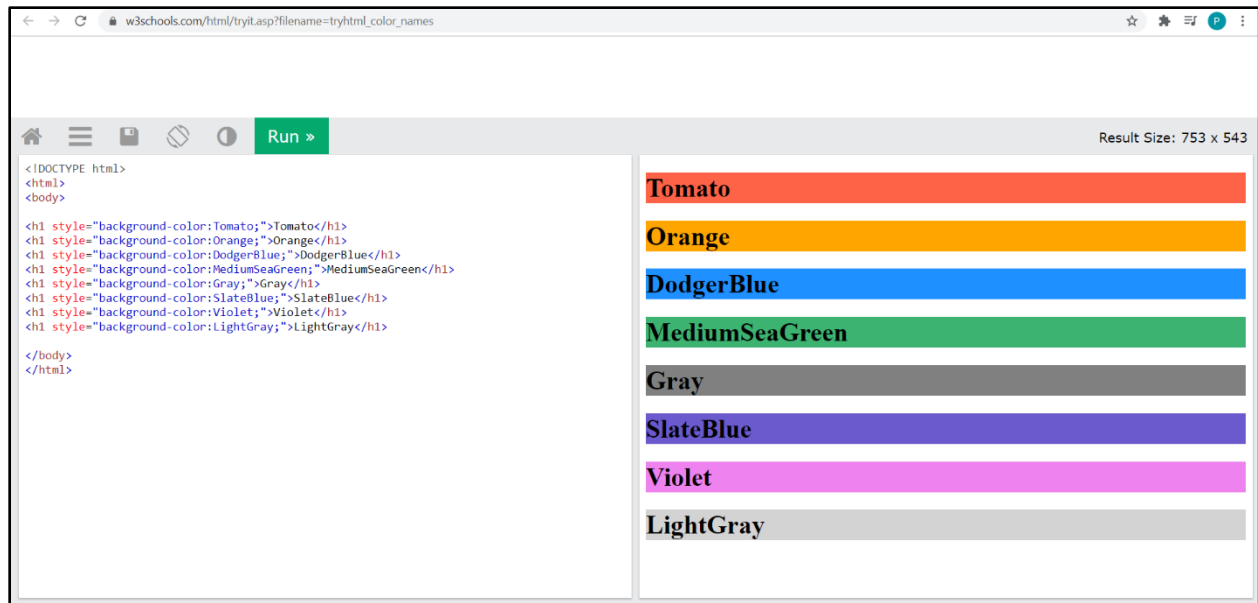
a.  $I_O$  specifications reflect the maximum current where the corresponding output voltage meets the  $V_{OH}/V_{OL}$  thresholds.  $I_O$  current can exceed these limits (subject to absolute maximum ratings).



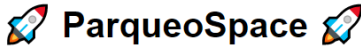
$$V_{out} = \frac{1k}{1k + LDR} \cdot 3.3v$$

### 4. Web-Server

Para nuestra conexión a internet y la pagina desarrollada se usó el código HTML. Este lenguaje se baso en los ejemplos encontrados en la página [www.w3school.com](http://www.w3school.com), la cual contiene ejemplos de como editar texto y de cómo dibujar tablas, insertar gráficos y refrescar la pagina cada tiempo determinado. Para este proyecto se graficó específicamente una tabla donde se mostraba la disponibilidad o no del parqueo con el color rojo y el verde, y al final se mostraba en fondo amarillo la cantidad de parqueos disponibles. En el código HTML al final se determino que cada medio segundo seria tiempo suficiente para actualizar la página, por lo que se programo de esa manera en el ESP32, el cual contiene el módulo Wifi que se conecto a la red.



192.168.1.5



Parqueo1	<input checked="" type="checkbox"/>
Parqueo2	<input type="checkbox"/>
Parqueo3	<input checked="" type="checkbox"/>
Parqueo4	<input type="checkbox"/>
TOTAL	2

## 5. Modulo UART

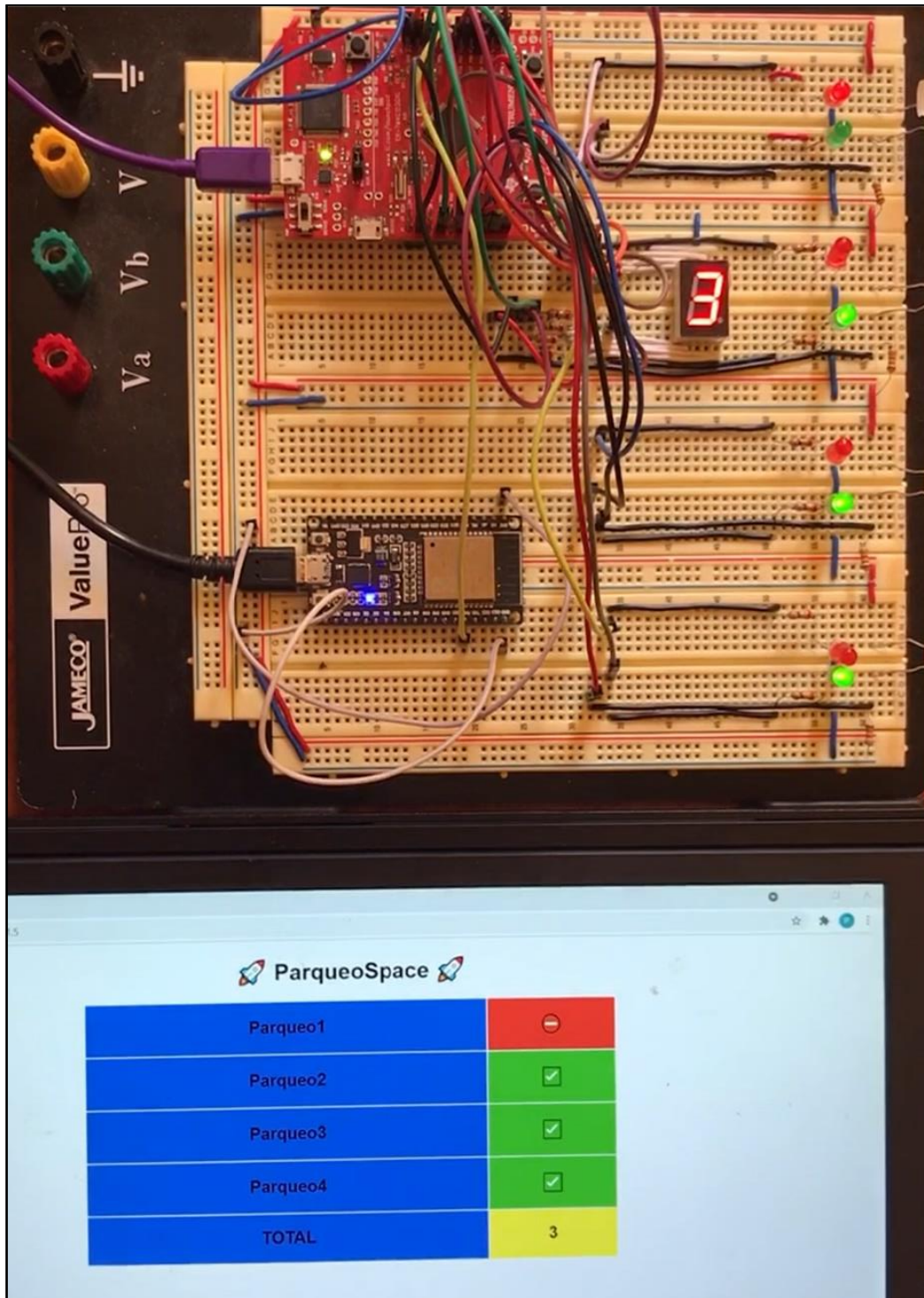
Para el modulo UART se uso el puerto C de la Tiva-C, exactamente el pin 6 que transmite **Tx**, y para el pin que recibe la información se uso el pin **Rx** del ESP32. Los datos enviados fueron separados con coma para separarlos en la recepción. Las características de la comunicación son las siguientes:

- BaudRate de 115,200
- Reloj del Sistema de 40 MHz
- 8 bits de datos tipo *unsigned char* en el envio
- Variables de ancho definido de 8 bits asimismo en la recepción
- Cadena enviada: **"0,0,0,0,"**



## 6. Funcionamiento

El funcionamiento final del proyecto se ve a continuación, donde dependiendo del valor obtenido en la lectura de datos, se muestra en el Display y en el WebServer la disponibilidad de parqueos.



## 7. Código TIVAC

```
//*****

#include <stdint.h>           // Variables de Ancho Definido
#include <stdbool.h>          // True or False
#include "inc/hw_memmap.h"    // Mapa de Memoria
#include "driverlib/debug.h"
#include "driverlib/gpio.h"   // Puertos
#include "driverlib/sysctl.h"
#include "inc/hw_types.h"
#include "inc/tm4c123gh6pm.h" // Dirverlib
#include "driverlib/timer.h"  // Timer
#include "driverlib/systick.h"
#include "driverlib/uart.h"   // UART
#include "driverlib/pin_map.h" // Pines
#include "driverlib/interrupt.h" // Interrupciones
#include "driverlib/fpu.h"

#define XTAL 16000000         // Frecuencia de Osilacion

uint8_t luz1;                // Sensores
uint8_t luz2;
uint8_t luz3;
uint8_t luz4;

uint8_t cont1 = 0;           // Contadores Tiva
uint8_t cont2 = 0;
uint8_t cont3 = 0;
uint8_t cont4 = 0;
uint8_t total = 0;

unsigned char char1 = '0';   // UART para ESP32
unsigned char char2 = '0';
unsigned char char3 = '0';
unsigned char char4 = '0';

//*****
//
// Rutina de error por si hay error en la libreria
//
//*****
#ifdef DEBUG
void
__error__(char *pcFilename, uint32_t ui32Line)
{
    while(1);
}
#endif

//*****
//
// Configuracion UART
//
//*****

void UART1config(void){

    // Enable reloj del UART1
    SysCtlPeripheralEnable(SYSCTL_PERIPH_UART1);

    // Colocar GPIO C4 - C5 como pines para UART
    GPIOPinConfigure(GPIO_PC4_U1RX);
    GPIOPinConfigure(GPIO_PC5_U1TX);

    // Enable los pines para ser perifericos
    GPIOPinTypeUART(GPIO_PORTC_BASE, GPIO_PIN_4 | GPIO_PIN_5);

    // Inicio Modulo UART: 115200, 8 bits de datos, 1 stop bit, No Paridad
    UARTConfigSetExpClk(UART1_BASE, SysCtlClockGet(), 115200, (UART_CONFIG_WLEN_8 | UART_CONFIG_STOP_ONE |
    UART_CONFIG_PAR_NONE));

    // Enable del UART1
    UARTEnable(UART1_BASE);
}

//*****
```

```

//
// Main
//
//*****
int main(void)
{
    // Reloj del Sistema: Principal | 16 MHz | (400 MHz/2) = 200 MHz | (200MHz/5) = 40 MHz
    SysCtlClockSet(SYSCTL_OSC_MAIN | SYSCTL_XTAL_16MHZ | SYSCTL_USE_PLL | SYSCTL_SYSDIV_5 );

    // Enable del reloj Puerto A
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOA);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOA))
    {}

    // Enable del reloj Puerto E
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOE);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOE))
    {}

    // Enable del reloj Puerto F
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOF);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOF))
    {}

    // Enable del reloj Puerto D
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOD);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOD))
    {}

    // Enable del reloj Puerto B
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOB);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOB))
    {}

    // Enable del reloj Puerto C
    SysCtlPeripheralEnable(SYSCTL_PERIPH_GPIOC);
    // Verifica si hay acceso a perifericos
    while(!SysCtlPeripheralReady(SYSCTL_PERIPH_GPIOC))
    {}

    // Enable GPIO para el LED. Pines de salida y digitales
    GPIOPinTypeGPIOOutput(GPIO_PORTA_BASE, GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_6 | GPIO_PIN_7);
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE, GPIO_PIN_1 | GPIO_PIN_3 );
    GPIOPinTypeGPIOOutput(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_3 );

    // Enable GPIO para el DISPLAY. Pines de salida y digitales
    GPIOPinTypeGPIOOutput(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
    GPIO_PIN_6 | GPIO_PIN_7);

    // Enable GPIO. Pines de entrada
    GPIOPinTypeGPIOInput(GPIO_PORTA_BASE, GPIO_PIN_4 | GPIO_PIN_5);
    GPIOPinTypeGPIOInput(GPIO_PORTC_BASE, GPIO_PIN_2 | GPIO_PIN_5);

    // Configuracion: Puerto, Pines, Corriente, Pull Up de la Tiva
    GPIOPadConfigSet(GPIO_PORTA_BASE, GPIO_PIN_4, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_STD_WPU);
    GPIOPadConfigSet(GPIO_PORTC_BASE, GPIO_PIN_2, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_STD_WPU);
    GPIOPadConfigSet(GPIO_PORTC_BASE, GPIO_PIN_5, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_STD_WPU);
    GPIOPadConfigSet(GPIO_PORTA_BASE, GPIO_PIN_5, GPIO_STRENGTH_8MA, GPIO_PIN_TYPE_STD_WPU);

    // Funcion Cofiguracion UART
    UART1config();

    while(1)
    {
        //*****

        luz1 = GPIOPinRead(GPIO_PORTA_BASE, GPIO_PIN_4); // Sensor uno recibe luz?
        if (luz1 == 0)
        {
            GPIOPinWrite(GPIO_PORTA_BASE, GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_3);
            cont1 = 0;
            char1 = '1'; // Led Rojo y contador en 0
        }
    }
}

```

```

    }
    else
    {
        GPIOWrite(GPIO_PORTA_BASE, GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_2);
        cont1 = 1;
        char1 = '0';
    }
    // Led Verde y contador en 1

//*****

luz2 = GPIOPinRead(GPIO_PORTA_BASE, GPIO_PIN_2);    // Sensor uno recibe luz?
if (luz2 == 0)
{
    GPIOWrite(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_3, GPIO_PIN_3);
    cont2 = 0;
    char2 = '1';
}
// Led Rojo y contador en 0
else
{
    GPIOWrite(GPIO_PORTD_BASE, GPIO_PIN_6 | GPIO_PIN_3, GPIO_PIN_6);
    cont2 = 1;
    char2 = '0';
}
// Led Verde y contador en 1

//*****

luz3 = GPIOPinRead(GPIO_PORTA_BASE, GPIO_PIN_5);    // Sensor uno recibe luz?
if (luz3 == 0)
{
    GPIOWrite(GPIO_PORTA_BASE, GPIO_PIN_1 | GPIO_PIN_3, GPIO_PIN_3);
    cont3 = 0;
    char3 = '1';
}
// Led Rojo y contador en 0
else
{
    GPIOWrite(GPIO_PORTA_BASE, GPIO_PIN_1 | GPIO_PIN_3, GPIO_PIN_1);
    cont3 = 1;
    char3 = '0';
}
// Led Verde y contador en 1

//*****

luz4 = GPIOPinRead(GPIO_PORTA_BASE, GPIO_PIN_5);    // Sensor uno recibe luz?
if (luz4 == 0)
{
    GPIOWrite(GPIO_PORTA_BASE, GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_6);
    cont4 = 0;
    char4 = '1';
}
// Led Rojo y contador en 0
else
{
    GPIOWrite(GPIO_PORTA_BASE, GPIO_PIN_2|GPIO_PIN_3|GPIO_PIN_6|GPIO_PIN_7, GPIO_PIN_7);
    cont4 = 1;
    char4 = '0';
}
// Led Verde y contador en 1

//*****

total = cont1 + cont2 + cont3 + cont4;
if (total == 0)
{
    GPIOWrite(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
GPIO_PIN_6 | GPIO_PIN_7, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 | GPIO_PIN_6);
}
else if (total == 1)
{
    GPIOWrite(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
GPIO_PIN_6 | GPIO_PIN_7,GPIO_PIN_2 | GPIO_PIN_3);
}
else if (total == 2)
{
    GPIOWrite(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
GPIO_PIN_6 | GPIO_PIN_7, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_7 | GPIO_PIN_5 | GPIO_PIN_4);
}
else if (total == 3)

```



```

    {
        GPIOWrite(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
        GPIO_PIN_6 | GPIO_PIN_7, GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_7);
    }
    else if (total == 4)
    {
        GPIOWrite(GPIO_PORTB_BASE, GPIO_PIN_0 | GPIO_PIN_1 | GPIO_PIN_2 | GPIO_PIN_3 | GPIO_PIN_4 | GPIO_PIN_5 |
        GPIO_PIN_6 | GPIO_PIN_7, GPIO_PIN_6 | GPIO_PIN_7 | GPIO_PIN_2 | GPIO_PIN_3);
    }
}

//*****

UARTCharPut(UART1_BASE, char1); // 0 0 1 // La cadena enviada es 0,0,0,0,
UARTCharPut(UART1_BASE, ','); // ,
UARTCharPut(UART1_BASE, char2); // 0 0 1
UARTCharPut(UART1_BASE, ','); // ,
UARTCharPut(UART1_BASE, char3); // 0 0 1
UARTCharPut(UART1_BASE, ','); // ,
UARTCharPut(UART1_BASE, char4); // 0 0 1
UARTCharPut(UART1_BASE, ','); // ,
}
}

```

## 8. Código ESP32

```

//-----
// Librerías
//-----

#include <WiFi.h> // Wi-Fi
#include <WebServer.h> // WebServer

//-----
// Variables globales
//-----

const char* ssid = "TURBONETT_21E83C"; // SSID
const char* password = "1D36E8B5D6"; // Password

WebServer server(80); // Object of WebServer(puerto HTTP POR DEFAULT)

bool Parqueo1 = LOW; // pagina web
bool Parqueo2 = LOW;
bool Parqueo3 = LOW;
bool Parqueo4 = LOW;

uint8_t conteo1 = 0; // suma de total
uint8_t conteo2 = 0;
uint8_t conteo3 = 0;
uint8_t conteo4 = 0;
uint8_t total = 0;

String uart1 = "0"; // UART
String uart2 = "0";
String uart3 = "0";
String uart4 = "0";

//-----
// Configuración
//-----

void setup() {
    Serial.begin(115200); // BaudRate comunicacion serial
    Serial.println("Try Connecting to ");
    Serial.println(ssid);

    WiFi.begin(ssid, password); // Conectarse al WIFI
    while (WiFi.status() != WL_CONNECTED) { // Se chequea si Wi-Fi esta conectado a la red
        delay(1000);
        Serial.print(".");
    }
    Serial.println("");
    Serial.println("WiFi connected successfully");
    Serial.print("Got IP: ");
}

```

```

Serial.println(WiFi.localIP());           // Mostrar la direccion IP del ESP32

server.on("/", handle_OnConnect);        // Conectarse deirectamente a una funcion desde la IP

server.onNotFound(handle_NotFound);      // Si no encuentra la IP se manda a la funcion error

server.begin();                          // Inicia el servidor HTTP
Serial.println("HTTP server started");
delay(100);
}

//-----
// loop principal
//-----

void loop() {
    server.handleClient();                // Servidor conecta a los clientes
    total = conteo1 + conteo2 + conteo3 + conteo4;

    if (Serial.available()) {             // si hay caracter en el Rx

        uart1 = Serial.readStringUntil(','); // lee valor leído hasta que haya una coma
        uart2 = Serial.readStringUntil(','); // lee valor leído hasta que haya una coma
        uart3 = Serial.readStringUntil(','); // lee valor leído hasta que haya una coma
        uart4 = Serial.readStringUntil(','); // lee valor leído hasta que haya una coma
    }

    if (uart1.toInt() == 0){               // Si es cero, es verde y el conteo 1
        Parqueo1 = LOW;                   // Si es uno, es rojo y el conteo 0
        conteo1 = 1;}
    else{
        Parqueo1 = HIGH;
        conteo1 = 0;}

    if (uart2.toInt() == 0){               // Si es cero, es verde y el conteo 1
        Parqueo2 = LOW;                   // Si es uno, es rojo y el conteo 0
        conteo2 = 1;}
    else{
        Parqueo2 = HIGH;
        conteo2 = 0;}

    if (uart3.toInt() == 0){               // Si es cero, es verde y el conteo 1
        Parqueo3 = LOW;                   // Si es uno, es rojo y el conteo 0
        conteo3 = 1;}
    else{
        Parqueo3 = HIGH;
        conteo3 = 0;}

    if (uart4.toInt() == 0){               // Si es cero, es verde y el conteo 1
        Parqueo4 = LOW;                   // Si es uno, es rojo y el conteo 0
        conteo4 = 1;}
    else{
        Parqueo4 = HIGH;
        conteo4 = 0;}

    Serial.print(uart1);                   // Se imprimen en la consola
    Serial.print(uart2);                   // los valores de los pines
    Serial.print(uart3);
    Serial.println(uart4);
}

//-----
// Handler de Inicio página
//-----

void handle_OnConnect() {
    Serial.println("Funcion exitosa");
    server.send(200, "text/html", SendHTML(Parqueo1, Parqueo2, Parqueo3, Parqueo4));
}

//-----
// Procesador de HTML
//-----

String SendHTML(uint8_t parqueo1, uint8_t parqueo2 ,uint8_t parqueo3 ,uint8_t parqueo4) {

    String ptr = "<!DOCTYPE html> <html>\n";           // Pagina Principal, Titulo, Tabla y Total
    ptr += "<head>\n";
    ptr += "<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align:center;}\n";

```

```

ptr += "table { width:50%;}}\n";
ptr += "th, td { padding: 5px; text-align: center;}\n";
ptr += "#t01 td:nth-child(odd) { background-color: #0E48DA;}\n";
ptr += "</style>\n";
ptr += "</head>\n";
ptr += "<body>\n";
ptr += "  <h1>&#128640 ParqueoSpace &#128640</h1>\n";
ptr += "  <table id=\"t01\" align=\"center\">\n";
ptr += "    <tr>\n";

//-----

ptr += "\t <td> <h2> Parqueo1 </h2></td>\n"; // Parqueo 1
if (parqueo1) // Si es uno, se coloca rojo
{ // Si es cero, se coloca verde
  ptr += "\t <td style=\"background-color: #F71919;\"><h2> &#9940</h2></td>\t\n";
}
else
{
  ptr += "\t <td style=\"background-color: #1BC50C;\"> <h2> &#9989</h2> </td>\t\n";
}
ptr += "\t</tr>\n";
ptr += "\t<tr>\n";

//-----

ptr += "\t <td><h2> Parqueo2 </h2></td>\n"; // Parqueo 2
if (parqueo2) // Si es uno, se coloca rojo
{ // Si es cero, se coloca verde
  ptr += "\t <td style=\"background-color: #F71919;\"><h2> &#9940</h2></td>\t\n";
}
else
{
  ptr += "\t <td style=\"background-color: #1BC50C;\"> <h2> &#9989</h2> </td>\t\n";
}
ptr += "\t</tr>\n";
ptr += "\t<tr>\n";

//-----

ptr += "\t <td><h2> Parqueo3 </h2></td>\n"; // Parqueo 3
if (parqueo3) // Si es uno, se coloca rojo
{ // Si es cero, se coloca verde
  ptr += "\t <td style=\"background-color: #F71919;\"><h2> &#9940</h2></td>\t\n";
}
else
{
  ptr += "\t <td style=\"background-color: #1BC50C;\"> <h2> &#9989</h2> </td>\t\n";
}
ptr += "\t</tr>\n";
ptr += "\t<tr>\n";

//-----

ptr += "\t <td><h2> Parqueo4 </h2></td>\n"; // Parqueo 4
if (parqueo4) // Si es uno, se coloca rojo
{ // Si es cero, se coloca verde
  ptr += "\t <td style=\"background-color: #F71919;\"><h2> &#9940</h2></td>\t\n";
}
else
{
  ptr += "\t <td style=\"background-color: #1BC50C;\"> <h2> &#9989</h2> </td>\t\n";
}
ptr += "\t</tr>\n";
ptr += "  <tr>\n";
ptr += "    <td><h2> TOTAL </h2></td>\n";

//-----

if (total == 0){ // Valor del total
  ptr += "\t <td style=\"background-color: #FCFF00;\"><h2>0</h2></td>\t\n";
}
else if (total == 1){
  ptr += "\t <td style=\"background-color: #FCFF00;\"><h2>1</h2></td>\t\n";
}
else if (total == 2){
  ptr += "\t <td style=\"background-color: #FCFF00;\"><h2>2</h2></td>\t\n";
}
else if (total == 3){

```

```

    ptr += "\t <td style=\"background-color: #FCFF00;\"><h2>3</h2></td>\t\n";
}
else if (total == 4){
    ptr += "\t <td style=\"background-color: #FCFF00;\"><h2>4</h2></td>\t\n";
}
ptr += "\t</tr>\n";

//-----

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

//-----
// Handler no encontrado
//-----

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

```