Prezentarea codului

Main.c (ruează pe microcontroller)

```
#include <stdio.h>
#include <string.h>
#include "esp_wifi.h"
#include "esp_system.h"
#include "nvs_flash.h"
#include "esp_event.h"
#include "protocol_examples_common.h"

#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/event_groups.h"


#include "esp_log.h"
#include "esp_websocket_client.h"
#include "esp_event.h"

#include "driver/gpio.h"
#include "driver/mcpwm.h"

#include "mfrc522.h"
#include "spi.h"

#define PIN_NUM_MISO 12
#define PIN_NUM_MOSI 13
#define PIN_NUM_CLK  14
#define PIN_NUM_CS   15
#define PIN_NUM_RST  GPIO_NUM_4

#define PWM_HZ 1000
#define PIN_M1 19
#define PIN_M2 18
#define PIN_EN1 23
#define PIN_EN2 22

#define ESP_INTR_FLAG_DEFAULT 0 //no ideea what is this

#define MINSPEED 5
#define MAXSPEED 99
#define KSPEED 0.01
#define PRECISION 7

static const char *TAG = "WEBSOCKET";
static const char *WEBSOCKET_ECHO_ENDPOINT = "ws://192.168.2.193:3024";
```

```
esp_websocket_client_handle_t wsClient;
volatile int status = 0;

int gspeed = 0;
volatile int counter = 0;
volatile uint32_t oldB1 = 0;

int q[100], qnext = 0, qlast = 0, qn = 0, qsize=99;
int qp[100], qpnext = 0, qplast = 0, qn = 0, qpsize=99;

int spotDist = 1500;
int spotOffset = 0;
int currentSpot = 0;
int parkingSpots = 8;

const int parkID = 8192;

void sock_log_handeling(int32_t event_id, esp_websocket_event_data_t *data, in
t level);

inline int max(int a, int b){return a > b ? a : b;}
inline int min(int a, int b){return a < b ? a : b;}
void updateSpeed(int speed)
{
    gspeed = speed;
    int norm = speed == 0 ? 0 : max(MINSPEED, max(MAXSPEED, abs(speed)));
    // printf("runnin' speed: %d", norm);
    mcpwm_set_duty(MCPWM_UNIT_0, MCPWM_TIMER_0, speed > 0 ? MCPWM_OPR_A : MCPW
M_OPR_B, 0);
    mcpwm_set_duty(MCPWM_UNIT_0, MCPWM_TIMER_0, speed > 0 ? MCPWM_OPR_B : MCPW
M_OPR_A, norm);
}

void rfid_task(void *pvParameter)
{
    printf("\tinitialising RFID... ");
    spi_init(PIN_NUM_CLK, PIN_NUM_MOSI, PIN_NUM_MISO);  // Init Driver SPI
    MFRC522_Init(PIN_NUM_RST, PIN_NUM_CS); // Init MFRC522
    printf("done\n");
    uint8_t CardID[30];
    while (1)
    {
        if (MFRC522_Check(CardID) == MI_OK)
        {
            char s[100];
            int len = sprintf(s, "{\"tag\": \"newCard\", \"id\": \"[%02x-%02x-
%02x-%02x-
%02x]\", \"currentSpot\": %d}", CardID[0], CardID[1], CardID[2], CardID[3], Ca
rdID[4], currentSpot);
```

```
            ESP_LOGI("MFRC", "%s \r\n", s);
            if (esp_websocket_client_is_connected(wsClient))
            {
                ESP_LOGI(TAG, "Sending %s", s);
                esp_websocket_client_send(wsClient, s, len, portMAX_DELAY);
            }

        }
        vTaskDelay(1000 / portTICK_PERIOD_MS);
    }
}

void mc_gpio_init()
{
    printf("initializing mcpwm gpio...\n");
    mcpwm_gpio_init(MCPWM_UNIT_0, MCPWM0A, PIN_M1);
    mcpwm_gpio_init(MCPWM_UNIT_0, MCPWM0B, PIN_M2);
    printf("Configuring Initial Parameters of mcpwm...\n");
    mcpwm_config_t pwm_config;
    pwm_config.frequency = PWM_HZ;    //frequency = 500Hz,
    pwm_config.cmpr_a = 0;    //duty cycle of PWMxA = 0
    pwm_config.cmpr_b = 0;    //duty cycle of PWMxb = 0
    pwm_config.counter_mode = MCPWM_UP_COUNTER;
    pwm_config.duty_mode = MCPWM_DUTY_MODE_0;
    mcpwm_init(MCPWM_UNIT_0, MCPWM_TIMER_0, &pwm_config);
}


static void gpio_isr_handler(void* arg)
{
    // uint32_t gpio_num = (uint32_t) arg;
    uint32_t newB0 = gpio_get_level(PIN_EN1);
    uint32_t newB1 = gpio_get_level(PIN_EN2);
    counter += ((newB0 == oldB1) ? 1 : -1);
    oldB1 = newB1;
}

void encoder_init()
{
    printf("initializing encoder... ");
    counter = 0;
    gpio_set_direction(PIN_EN1, GPIO_MODE_INPUT);
    gpio_set_direction(PIN_EN2, GPIO_MODE_INPUT);
    gpio_set_pull_mode(PIN_EN1, GPIO_PULLUP_ONLY);
    gpio_set_pull_mode(PIN_EN2, GPIO_PULLUP_ONLY);
    gpio_set_intr_type(PIN_EN1, GPIO_INTR_ANYEDGE);
    gpio_set_intr_type(PIN_EN2, GPIO_INTR_ANYEDGE);

    printf("starting interrupts... ");
    gpio_install_isr_service(ESP_INTR_FLAG_DEFAULT);
    gpio_isr_handler_add(PIN_EN1, gpio_isr_handler, (void*) PIN_EN1);
```

```
    gpio_isr_handler_add(PIN_EN2, gpio_isr_handler, (void*) PIN_EN2);

    printf("done\n");
}

void moveToPos(int pos)
{
    printf("moving to pos: %d\n", pos);

    int err = pos - counter;
    int lastdir = 0;
    while(abs(err) > PRECISION)
    {
        updateSpeed((int)err*KSPEED);
        lastdir = err/abs(err);
        vTaskDelay(10/portTICK_PERIOD_MS);
        err = pos - counter;
    }
    updateSpeed(-lastdir*60);
    vTaskDelay(30/portTICK_PERIOD_MS);
    updateSpeed(0);

    printf("arrived to: %d", counter);
}

void moveToSpot(int spot)
{
    //if(spot == currentSpot) return;
    spot %= parkingSpots;
    moveToPos(spotOffset + spot*spotDist);
    currentSpot = spot;
}


void motorTask()
{
    int next;
    while(true)
    {
        if(qpn > 0)
        {
            next = qp[qnext++];
            qpnext %= qpsize;
            qpn--;
            moveToPos(next);
        }
        if(qn > 0)
        {
            next = q[qnext++];
            qnext %= qsize;
            qn--;
```

```
            moveToSpot(next);
        }
        vTaskDelay(20 / portTICK_PERIOD_MS);
    }
}

static void websocket_event_handler(void *handler_args, esp_event_base_t base,
 int32_t event_id, void *event_data)
{
    // esp_websocket_client_handle_t client = (esp_websocket_client_handle_t)h
andler_args;
    esp_websocket_event_data_t *data = (esp_websocket_event_data_t *)event_dat
a;
    sock_log_handeling(event_id, data, 2);
    if(event_id != WEBSOCKET_EVENT_DATA || data->op_code != 1) return;
    char *s = (char*)data->data_ptr;
    int n = data->data_len;
    s[n] = '\0';
    if(s[0] != '_') return;

    if(strncmp(s, "_spot", 5) == 0)
    {
        int newSpot;
        sscanf(s+6, "%d", &newSpot);
        q[qlast++] = newSpot;
        qlast %= qsize;
        qn++;
    }
    else if(strncmp(s, "_speed", 6) == 0)
    {
        int speed;
        printf("new speed");
        sscanf(s+7, "%d", &speed);
        printf(": %d\n", speed);
        updateSpeed(speed);
    }
    else if(strncmp(s, "_pos", 4) == 0)
    {
        int pos;
        sscanf(s+5, "%d", &pos);
        qp[qlast++] = pos;
        qplast %= qsize;
        qpn++;
    }
    else if(strncmp(s, "_set_dist", 9) == 0)
    {
        int dist;
        sscanf(s+10, "%d", &dist);
        spotDist = dist;
    }
```

```
    else if(strncmp(s, "_set_offset", 11) == 0)
    {
        int off;
        sscanf(s+12, "%d", &off);
        spotOffset = off;
    }
    else if(strncmp(s, "_set_current", 12) == 0)
    {
        int crt;
        sscanf(s+13, "%d", &crt);
        currentSpot = crt;
    }
    else if(strncmp(s, "_get_status", 11) == 0)
    {
        char res[100];
        int len = sprintf(s, "{\"tag\": \"status\", \"status\": %d, \"crtSpot\
": %d, \"pos\": %d}", status, currentSpot, counter);
        esp_websocket_client_send(wsClient, res, len, portMAX_DELAY);
    }
    else if(strncmp(s, "_get_info", 9) == 0)
    {
        char res[100];
        int len = sprintf(s, "{\"tag\": \"info\", \"parkID\": %d, \"parkingSpo
ts\": %d, \"spotDist\": %d, \"spotOffset\": %d}", parkID, parkingSpots, spotDi
st, spotOffset);
        esp_websocket_client_send(wsClient, res, len, portMAX_DELAY);
    }
}

static esp_websocket_client_handle_t websocket_app_start(void)
{
    ESP_LOGI(TAG, "Connectiong to %s...", WEBSOCKET_ECHO_ENDPOINT);
    const esp_websocket_client_config_t websocket_cfg = {
        .uri = WEBSOCKET_ECHO_ENDPOINT,
    };
    esp_websocket_client_handle_t client = esp_websocket_client_init(&websocke
t_cfg);
    esp_websocket_register_events(client, WEBSOCKET_EVENT_ANY, websocket_event
_handler, (void *)client);
    esp_websocket_client_start(client);
    return client;
}

void logCounter()
{
    int lastCounter = counter;
    while(true)
    {
        if(counter != lastCounter)
            printf("counter: %d\n", counter);
```

```
        vTaskDelay(500 / portTICK_PERIOD_MS);
    }
}

void app_main()
{
    ESP_LOGI(TAG, "[APP] Startup..");
    esp_log_level_set("*", ESP_LOG_INFO);
    esp_log_level_set("WEBSOCKET_CLIENT", ESP_LOG_DEBUG);
    esp_log_level_set("TRANS_TCP", ESP_LOG_DEBUG);

    ESP_ERROR_CHECK(nvs_flash_init());
    tcpip_adapter_init();
    ESP_ERROR_CHECK(esp_event_loop_create_default());

    /* This helper function configures Wi-
Fi or Ethernet, as selected in menuconfig.
     * Read "Establishing Wi-Fi or Ethernet Connection" section in
     * examples/protocols/README.md for more information about this function.
     */
    ESP_ERROR_CHECK(example_connect());

    wsClient = websocket_app_start();
    mc_gpio_init();
    encoder_init();

    printf("creating rfid task...\n");
    xTaskCreate(&rfid_task, "rfid_task", 4096, NULL, 4, NULL);
    printf("done\n");

    printf("creating motor task...\n");
    xTaskCreate(&motorTask, "motorTask", 4096, NULL, 4, NULL);
    printf("done\n");

    //xTaskCreate(&logCounter, "logCounter", 2048, NULL, 4, NULL);

}




void sock_log_handeling(int32_t event_id, esp_websocket_event_data_t *data, in
t level)
{
    if(level <= 0) return;
    switch (event_id) {
        case WEBSOCKET_EVENT_CONNECTED:
            ESP_LOGI(TAG, "WEBSOCKET_EVENT_CONNECTED");
            break;
        case WEBSOCKET_EVENT_DISCONNECTED:
```

```
            ESP_LOGI(TAG, "WEBSOCKET_EVENT_DISCONNECTED");
            break;
        case WEBSOCKET_EVENT_DATA:
            if(level >= 2)
            {
                ESP_LOGI(TAG, "WEBSOCKET_EVENT_DATA");
                ESP_LOGI(TAG, "Received opcode=%d", data->op_code);
                ESP_LOGW(TAG, "Received=%.*s\r\n", data-
>data_len, (char*)data->data_ptr);
            }
            break;
        case WEBSOCKET_EVENT_ERROR:
            ESP_LOGI(TAG, "WEBSOCKET_EVENT_ERROR");
            break;
    }
}
```

Index.js (server demo)

```
import express from 'express';
import * as http from 'http';
import WebSocket from 'ws';

const app = express();
const server = http.createServer(app);
const wss = new WebSocket.Server({ server });

let parked = {}
let spots = [0, 0, 0, 1, 1, 0, 1, 0]
let socks = []

let searchFree = ()=>{
    for(let i in spots)
    {
        if(spots[i] == 0)
            return i;
    }
    return -1;
}
var newCard = (sock, data) => {
    if(parked[data.id] != null)
    {
        console.log(`retriving car in spot ${parked[data.id]}`);
        spots[parked[data.id]] = 0;
        parked[data.id] = null;
        console.log(parked);
        sock.send(`_spot ${parked[data.id]}`);
    }
    else
    {
        if(spots[data.currentSpot] == 0)
        {
            console.log(`new car saved in spot ${data.currentSpot}`)
            spots[data.currentSpot] = 1;
            parked[data.id] = data.currentSpot;
        }
        console.log("searching free spot...");
        let f = searchFree();
        if(f == -1)
        {
            console.log("no more space");
            sock.send('no more space');
        }
        else
        {
            console.log(`moving to new spot: ${f}`)
            sock.send(`_spot ${f}`);
        }
```

```
        console.log(parked);
    }
}

wss.on('connection', (sock) => {
    console.log("new con");
    //connection is up, let's add a simple simple event
    socks.push(sock);
    sock.on('message', (message) => {

        //log the received message and send it back to the client
        sock.send(`server recived -> ${message}`);
        try{
            let data = JSON.parse(message);
            console.log(data);
            if(data.tag == 'newCard')
                newCard(sock, data);
        }
        catch{
            console.log("no Json data: " + message);
        }
    });
    //send immediatly a feedback to the incoming connection
    sock.send('Hi there, I am a WebSocket server');

});

//start our server
server.listen(process.env.PORT || 3024, () => {
    console.log(`Server started on port ${server.address().port} :)`);
});
```

Front-end: Scripts

```
const host = "http://localhost:3024"


function getPhoneId() {
    let id = localStorage.getItem("sparkID");
    if (id == null) {
        id = "SPARK" + Math.floor(Math.random() * 100000000);
        localStorage.setItem("sparkID", id);
    }
    console.log("id: ", id);
    return id;
}

function getPhoneNr() {
    let inp = document.getElementById("phoneNumberInp");
    return inp.textContent;
}

function statuss(cb){
    fetch(host + "/status", {
        method: "GET",
        "Content-type": "application/json; charset=UTF-8"
    }
    ).then((response) => {
        response.json()
            .then(data => {
                cb(data);
            })
    }).catch(console.log)
}

function sendPark() {
    fetch(host+"/register", {
        method: "POST",
        body: JSON.stringify({
            id: getPhoneId(),
            phone: getPhoneNr()
        }),
        headers: {
            "Content-type": "application/json; charset=UTF-8"
        }
    }).then(response => {
        console.log("Server: ", response.data)
    }).catch(console.log)
```

```
}

setInterval(function(){
    statuss((data=>{
        console.log(data);
        let p = document.getElementById("avSpots");
        text = JSON.stringify(data);
        text = text.replace('{', '').replace('}', '').replaceAll('"',
'');
        p.innerText = text;
    }));
}, 1000);

function change() {
    let elem = document.getElementById("submitBut");
    let toast = new bootstrap.Toast(document.getElementById("toastt"));
    let toasthtml = document.getElementById("toastt");
    if (elem.innerHTML == "Park") {
        sendPark();
        elem.innerHTML = "Retrieve";
        toast.show();
        toasthtml.innerHTML = "Vehicle has been parked."
    }
    else {
        sendPark();
        elem.innerHTML = "Park";
        toast.show();
        toasthtml.innerHTML = "Vehicle has been retrieved."
    }
}

function enableDisableBut() {
    let but = document.getElementById("submitBut");
    let input = document.getElementById("phoneNumberInp").value;
    if (input.length >= 10)
        but.classList.remove('disabled');
    else if (input.length < 10)
        but.classList.add('disabled');
}
```

Pagina Web

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8" />
        <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no" />
        <meta name="description" content="" />
        <meta name="author" content="" />
        <title>SPark</title>
        <link rel="icon" type="image/x-icon" href="assets/favicon.ico"
/>
        <script
src="https://use.fontawesome.com/releases/v6.1.0/js/all.js"
crossorigin="anonymous"></script>
        <link rel="preconnect" href="https://fonts.gstatic.com" />
        <link
href="https://fonts.googleapis.com/css2?family=Tinos:ital,wght@0,400;0,
700;1,400;1,700&amp;display=swap" rel="stylesheet" />
        <link
href="https://fonts.googleapis.com/css2?family=DM+Sans:ital,wght@0,400;
0,500;0,700;1,400;1,500;1,700&amp;display=swap" rel="stylesheet" />
        <link href="css/styles.css" rel="stylesheet" />
        <script src="js/scripts.js"></script>
    </head>
    <body>
        <video class="bg-video" playsinline="playsinline"
autoplay="autoplay" muted="muted" loop="loop"><source
src="assets/mp4/bg.mp4" type="video/mp4" /></video>
        <div class="masthead">
            <div class="masthead-content text-white" style="padding-
top: 50px;">
                <div class="container-fluid px-4 px-lg-0" style="text-
align: center">
                    <img style="height:150px; width:150px; margin-
bottom: 30px;" src="assets/icon.png">
                    <p class="mb-5" style="text-align: center;"
id="avSpots">Available parking spots: -</p>
                    <div class="row input-group-newsletter">
                        <div class="col"><input class="form-control"
type="number" id="phoneNumberInp" placeholder="Enter phone number..."
aria-label="Enter phone number..." onkeyup="enableDisableBut()"/></div>
                        <div class="col-auto"><button class="btn btn-
primary disabled" id="submitBut" onclick="change()" style="width:
110px">Park</button></div>
```

```
                    </div>
                </div>
            </div>
        </div>
        <div class="social-icons">
            <div class="d-flex flex-row flex-lg-column justify-content-
center align-items-center h-100 mt-3 mt-lg-0">
                <a class="btn btn-dark m-3" href="https://ro-
ro.facebook.com/academiadeinformatica/"><i class="fab fa-facebook-
f"></i></a>
                <a class="btn btn-dark m-3"
href="https://www.instagram.com/academiadeinfo/"><i class="fab fa-
instagram"></i></a>
            </div>
        </div>
        <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bun
dle.min.js"></script>
        <script src="js/scripts.js"></script>
        <script src="https://cdn.startbootstrap.com/sb-forms-
latest.js"></script>
        <div style="margin: 5% auto; position: fixed; bottom: 10px;
width: 95%; padding: 10px; left: 0; right: 0;" class="toast align-
items-center" role="alert" aria-live="assertive" aria-atomic="true"
id="toastt">
            <div class="d-flex">
              <div class="toast-body">
                Vehicle has been parked.
              </div>
              <button type="button" class="btn-close me-2 m-auto" data-
bs-dismiss="toast" aria-label="Close"></button>
            </div>
          </div>

    </body>
</html>
```