Access control system

Demo: Control gate and door with ESP8266 via web interface | PM2021 @ ACS, UPB

Autor

Dragus Alexandru
Linkedin
Grupa: 334CB

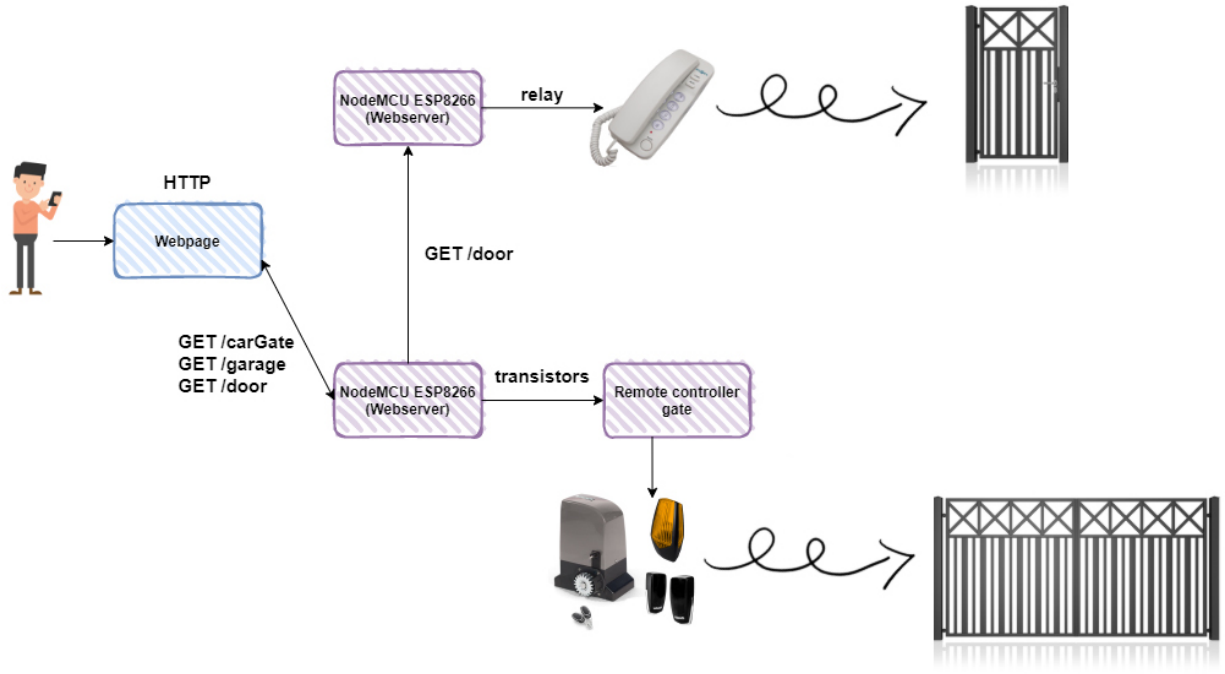
Introducere

- Prezentul proiect este realizat in cadrul materiei de **PM 2021**, la Facultatea de Automatica si Calculatoare, Universitatea Polithenica Bucuresti
- Proiectul presupune realizarea unui **sistem wireless de acces**, prin transmiterea de semnale unor device-uri third-party din viata reala: un interfon care deschide o usa si un controller care deschide automat (cu motor) o poarta de acces auto.
- **Scopul**: dupa implementarea proiectului, controlul sistemelor mentionate se va face de pe un telefon sau un calculator din reteaua locala, fara a mai fi nevoie de telecomanda fizica cu infrarosu (pentru poarta auto → reducere costuri) sau de deplasarea pana la interfon (pentru usa → accesibilitate)

Descriere generala

- Sistemul presupune controlul unor componente (`releu → intrerupator → interfon`) sau a unor device-uri (`tranzistor → remote controller poarta auto/ garaj`) de la distanta
- **Scalabilitatea** este un factor important, pentru ca sistemului i se pot adauga alte functionalitati (`smarthome`)
- Pentru device-urile de controlat din apropierea celor deja implementate, se pot trasa doar fire, costul fiind foarte mic. Pentru device-uri la distanta (cum este poarta exterioara), costul este acela al elementelor hardware (**NodeMCU ESP8266**)
- Se foloseste NodeMCU ESP8266 pentru server-ul web si pentru controlul releelor sau a semnalelor.
- Server-ul web este in **reteaua locala** a locuintei, practic securizarea fiind in grija access point-ului wireless.

Schema bloc



Device-ul din partea de sus, conectat la interfon, va fi referit in continuare prin **`Device interfon`**, si reprezinta un server care asculta pe /door si activeaza un releu.

Device-ul de jos este **`Device hub`** si reprezinta webserver-ul principal, cel care ofera interfata utilizatorului. Totodata, hub-ul transmite si semnale remote controller-ului pentru poarta.

Hardware Design

Piese necesare

Toate piesele au denumiri comune si se pot gasi printr-o cautare scurta pe internet.

Nr.crt	Piesa	Nr. bucati
1	NodeMCU ESP8266	2
2	Tranzistor NPN 2N3904	3
3	Rezistente 1k	3
4	Headere pini mama	60
5	Conector dublu cu surub	6
6	Regulator tensiune 5V	1
7	Regulator tensiune 3.3V	2
8	Modul releu 5V	1
9	Headere pini tata	10
10	Cap jumper	4
11	Fire	60
12	Placa prototipare PCB	2

13	Fludor	1
14	Conector DC Jack	1
15	Buton on/ off	2
16	Incarcator 220V → 7-12V	2

Echipamente recomandate

Acestea sunt echipamentele folosite in dezvoltarea prezentului proiect. Sistemul final se poate realiza si pe breadboard, fara a fi nevoie de lipituri.

Nr.crt	Echipament
1	Pistol de lipit
2	Multimetru
3	Pistol cu silicon
4	Surubelnita
5	Cleste fire

Sistemele real-life cu care este conectat produsul rezultat

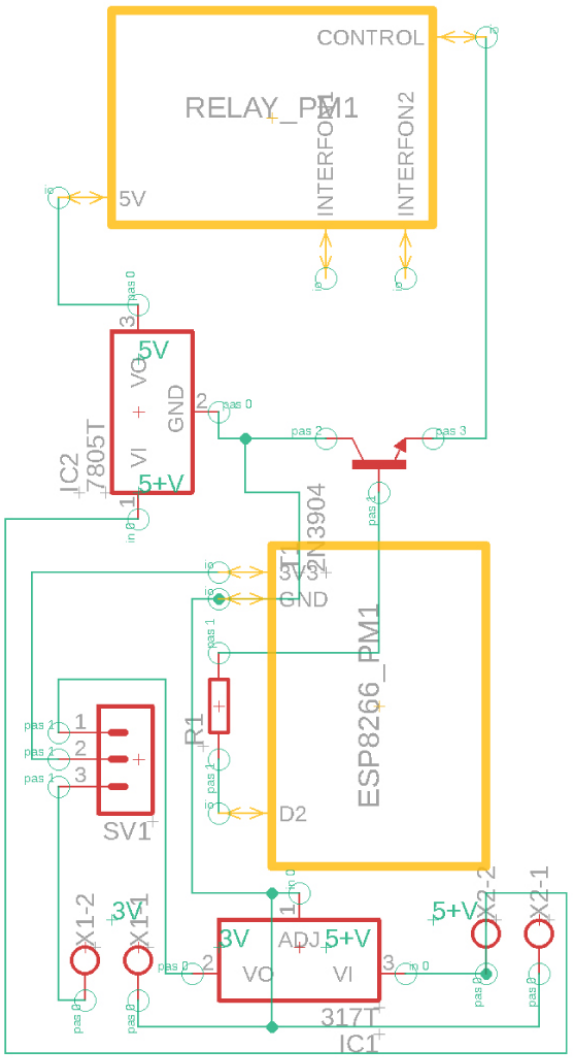
Scurta descriere a sistemelor reale cu care este conectat produsul finit:

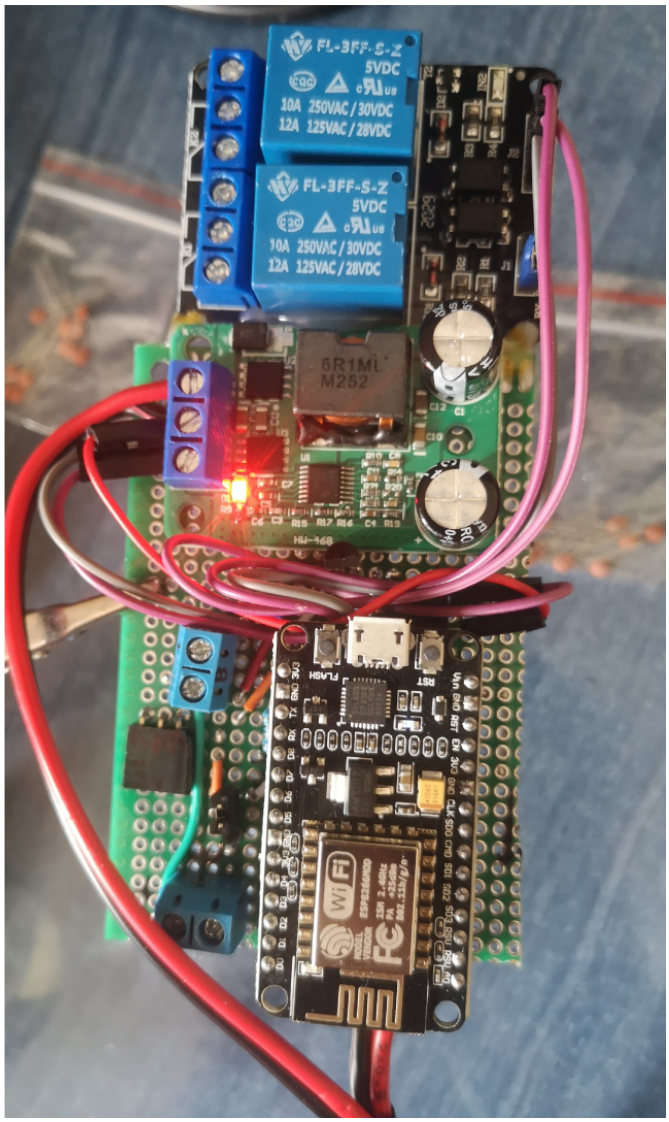
Nr.crt	Sistem	Explicatie
1	Interfon care deschide poarta prin contact mecanic	NodeMCU ESP8266 face contactul printr-un releu
2	Sistem automat deschidere poarta auto Nice	Actionat de telecomanda
3	Sistem automat deschidere garaj Nice	Actionat de telecomanda
4	Telecomanda Nice	Actionate butoanele de NodeMCU ESP8266 prin tranzistoare

Scheme electrice si placi fizice

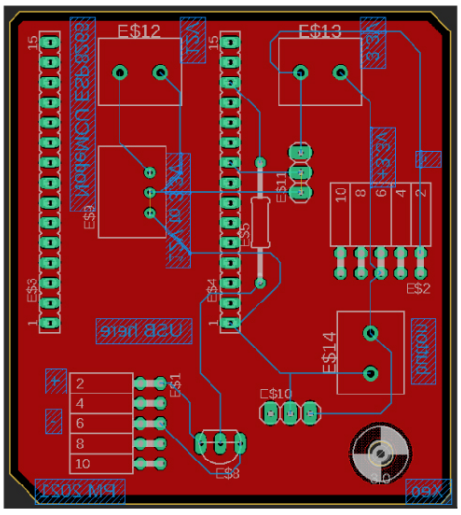
Device interfon

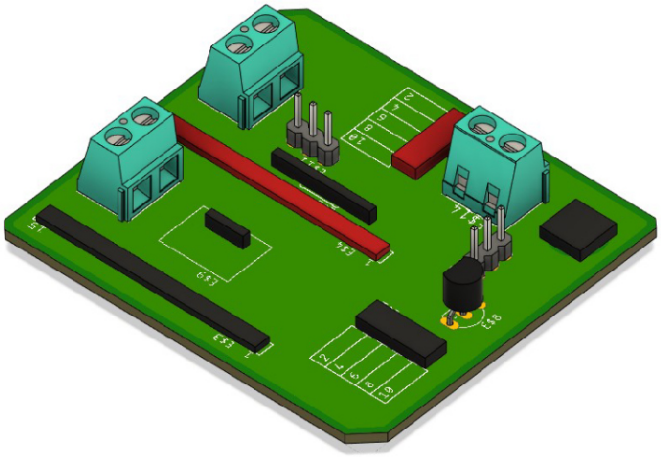
Device-ul interfon se poate alimenta direct la 3.3V sau la o tensiune intre 7 si 24V. Acesta are atasat un modul de relee prin care poate controla dispozitive cu tensiuni mari. In cazul de fata, releul face contactul la butonul pentru deschiderea portii, semnalul venind printr-un tranzistor, pentru a putea transmite 5V.





Device-urile se pot produce si pe placi PCB proiectate in software, dupa cum se vede in exemplul de mai jos, realizat in Eagle. Totodata, Eagle ofera suport pentru proiectarea 3D a placutei, cu ajutorul Fusion360.

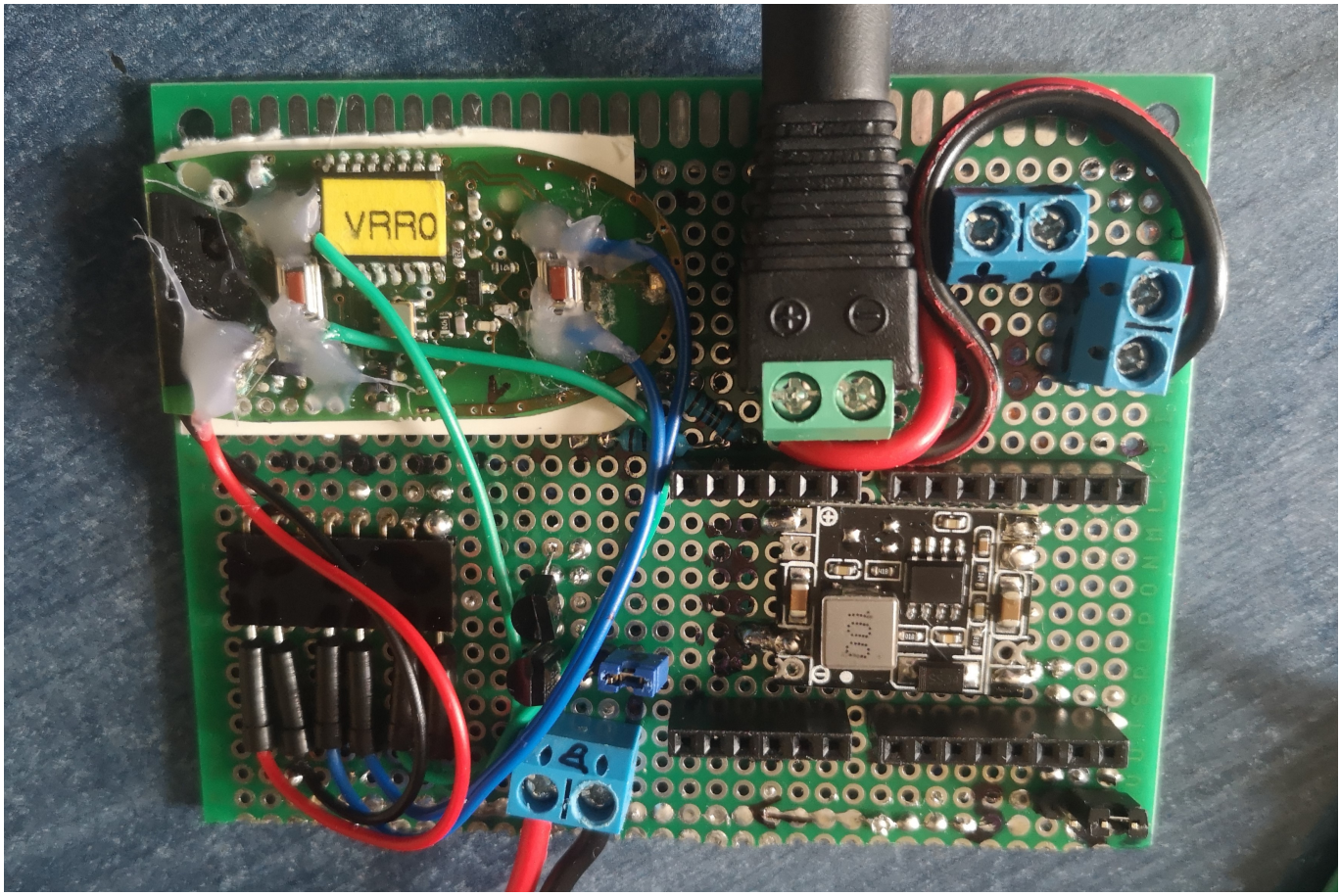




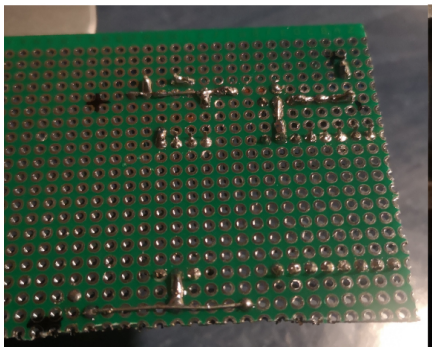
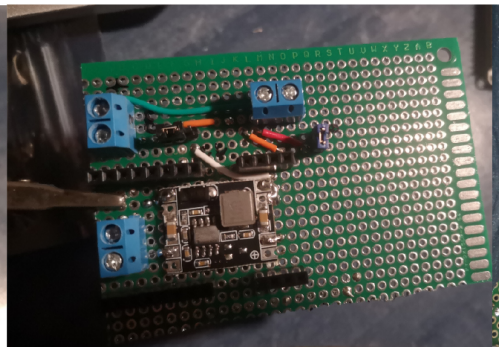
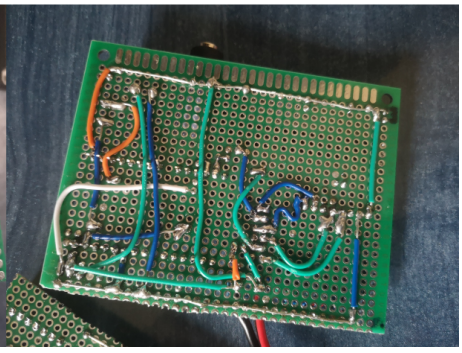
Imaginile de proiectare PCB/ 3D sunt exemplificative, placa finala avand atasate modulele de releu si regulator 5V in plus.

Device hub

Device-ul hub este alimentat de la 8V prin incarcatorul unui telefon fix, lipit cu un conector Jack. Acesta are atasata o telecomanda pentru poarta care are butoanele conectate la colectorul si emitorul tranzistorilor, astfel facandu-se contact atunci cand se transmite semnal in baza.



Imagini din timpul procesului de lipire:

Software Design

- Mediu de dezvoltare: **Arduino IDE**
- Placute de dezvoltare: **NodeMCU ESP8266**

- Biblioteci folosite: **ESP8266WiFi, ESPAsyncWebServer, ESP8266HTTPClient** (link la resurse)

Aplicatia trebuie sa fie cat mai usor de folosit de catre un utilizator obisnuit, astfel ca am ales o interfata web simpla, cu 3 butoane, cate unul pentru fiecare 'usa' de deschis. Programarea se realizeaza Arduino-like pe placutele NodeMCU ESP8266.

Device interfon

Codul are urmatoarea logica:

1. Se conecteaza dispozitivul la reteaua de wifi setata
2. Se initializeaza un server HTTP, pe portul 80
3. Se asteapta un request `GET /door`
4. La request, se transmite un semnal HIGH pe pinul D2, care va activa tranzistorul si, in consecinta, releul. Astfel, butonul interfonului va face contact si se va deschide poarta.
5. Se returneaza o pagina HTML care redirecteaza utilizatorul la adresa IP statica a `Device Hub`

GET /door handler:

```
server.on("/door", HTTP_GET, [](AsyncWebServerRequest *request) {
    // Send webpage
    request->send(200, "text/html", doorWebpage());
    Serial.println("Door pressed");
    timePressDoor = millis();
    doorState = "on";

    // Send signal for door (relay)
    digitalWrite(doorPin, HIGH);
});
```

Device hub

Codul are urmatoarea logica:

1. Se conecteaza dispozitivul la reteaua de wifi setata
2. Se initializeaza un server HTTP, pe portul 80
3. Se asteapta request-uri: `GET /garage`, `GET /carGate`, `GET /door`
4. Pentru fiecare request, exista un handler care se apeleaza
5. Pentru request-urile garage si carGate, se returneaza o pagina care confirma actiunea, redirecteaza utilizatorul la home si seteaza pin-ul corespunzator pe HIGH
6. Pentru request-ul `GET /door`, se redirecteaza utilizatorul spre adresa `Device interfon` pentru a face GET acolo
7. Dupa un delay prestabilit, se reseteaza pe LOW toti pinii, dupa ce a fost apasat un buton

Exemplu handler GET request:

```
server.on("/cargate", HTTP_GET, [](AsyncWebServerRequest *request) {
```

```
    // Send webpage
    request->send(200, "text/html", buttonWebpage("cargate"));
    Serial.println("Car gate pressed");
    timePressCarGate = millis();
    carGateState = "on";

    // Send signal for Car gate
    digitalWrite(carGatePin, HIGH);
});
```

Functie care returneaza o pagina web simpla:

```
String buttonWebpage(String which) {
  String webpage = "";
  webpage = webpage + ("<!DOCTYPE html><html>");
  webpage = webpage + ("<head><meta name=\"viewport\"
content=\"width=device-width, initial-scale=1\">");
  webpage = webpage + ("");
  webpage = webpage + ("<style>html { font-family: Helvetica; display:
inline-block; margin: 0px auto; text-align: center;}");
  webpage = webpage + ("</style>");
  webpage = webpage + ("<meta http-equiv=\"refresh\"
content=\"1;url=http://192.168.0.103\" />");
  webpage = webpage + ("</head>");
  webpage = webpage + ("<body><h1>Access system</h1>");
  webpage = webpage + ("<h2>Se deschide ");
  if (which == "garage")
    webpage = webpage + "garajul";
  else if (which == "cargate")
    webpage = webpage + "poarta auto";
  else
    webpage = webpage + "poarta mica";
  webpage = webpage + ("</h2>");
  webpage = webpage + ("<br></br><br></br><br></br>");
  webpage = webpage + ("<h3>Proiect realizat in cadrul materiei<br></br>");
  webpage = webpage + ("Proiectarea pe Microprocesoare<br></br>");
  webpage = webpage + ("</body></html>");
  return webpage;
}
```

Snippet cod care reseteaza butonul la starea 'neapasat':

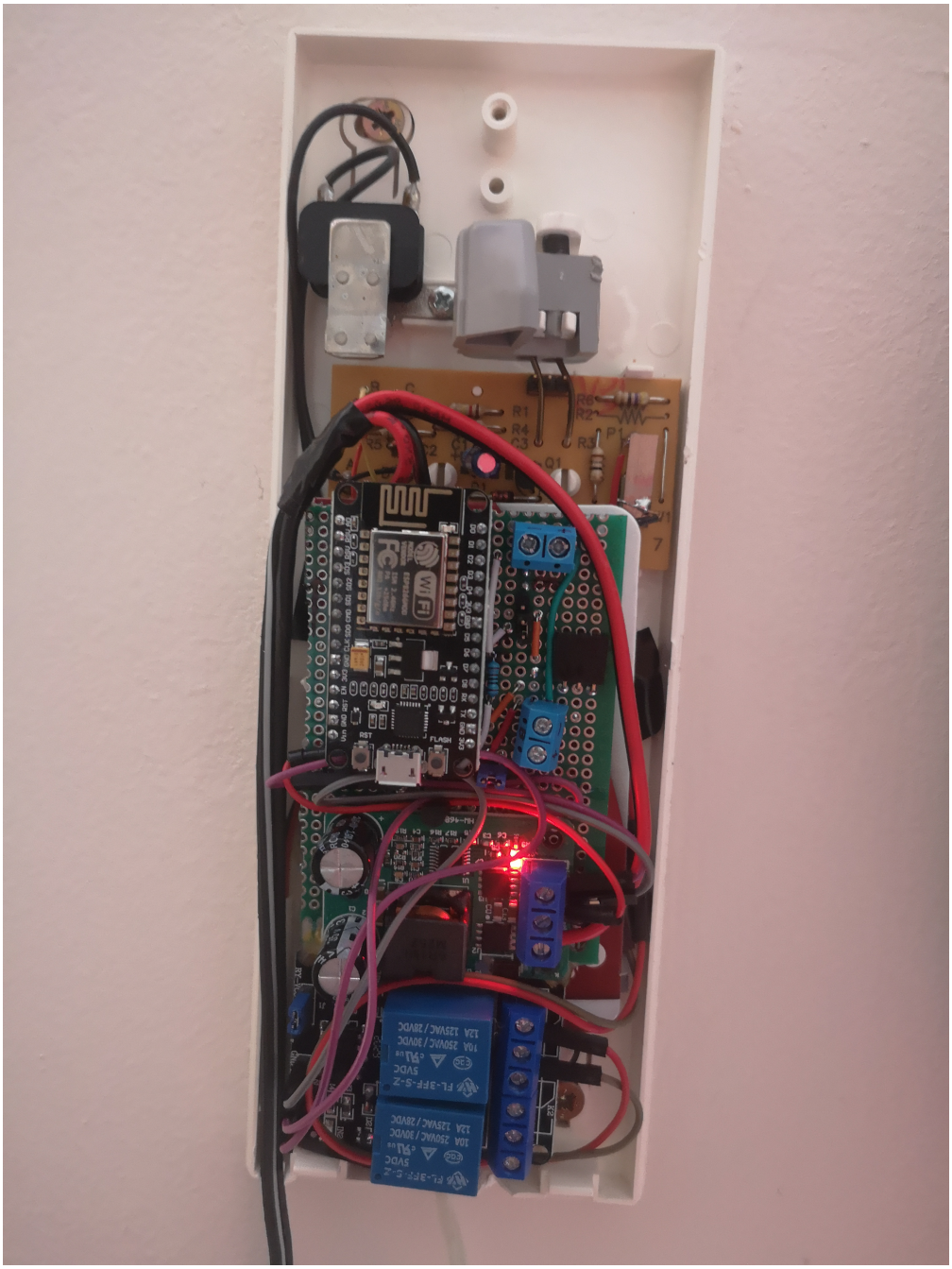
```
  currentTime = millis();
  // Release button if delay time passed
  if (garageState == "on" && currentTime - timePressGarage > timeoutButtons)
{
    garageState = "off";
```

```
    digitalWrite(garagePin, LOW);
    Serial.println("Garage unpressed");
  }
```

Rezultate Obtinute

Demo: Control gate and door with ESP8266 via web interface | PM2021 @ ACS, UPB

Proiectul a fost finalizat cu succes si este implementat intr-o situatie din viata reala. Mai jos este o imagine cu `Device-ul interfon` montat intr-un interfon real.



Concluzii

Consider ca proiectul la PM a fost o activitate placuta, care mi-a adus mai multa experienta hands-on in domeniul embedded.

Download

Code and schematics (GitHub)

Jurnal

- **31 mai**: Finalizare documentatie
- **30 mai**: Video demo
- **30 mai**: Conectare device-uri in configuratia finala la sistemele real-life
- **29 mai**: Cod final si conectare intre device-uri
- **26 mai**: Realizare scheme placute
- **25 mai**: Placa prototipare device Interfon
- **23 mai**: Cod test
- **21 mai**: Comanda #2 componente (inca un NodeMCU ESP8266)
- **16 mai**: Placa prototipare device Hub
- **15 mai**: Prototip breadboard
- **10 mai**: Lipire fire telecomanda si interfon
- **9 mai**: Comanda #1 componente
- **7 mai**: Research componente
- **4 mai**: Research fezabilitate aplicare real-life
- **25 aprilie**: Pagina wiki
- **20 aprilie**: Alegere tema proiect

Bibliografie/ Resurse

- **Demo**: Control gate and door with ESP8266 via web interface | PM2021 @ ACS, UPB
- Access control system
- Code and schematics (GitHub)
- NodeMCU ESP8266 Client and Server tutorial
- Biblioteca ESPAsyncWebServer
- Biblioteca ESPAsyncTCP
- Schema conectare tranzistor
- Pinout NodeMCU ESP8266
- Pagina wiki Etapa 1

Export to PDF