Šolski center Celje, Srednja šola za kemijo, elektrotehniko in računalništvo

Pametna garažna vrata

Raziskovalna naloga

Avtor
Boštjan Planko, R-4.A

Mentor
Borut Slemenšek

Celje, February 15, 2019

Povzetek

Ključne besede

Kazalo

Kazalo tabel

Kazalo slik

1 Uvod

1.1 Opis problema

Problem današnjih garažnih vrat je, da jih je običajni možno odpreti samo na dva načina. S priloženim daljincem ali s tipko, običajno nameščeno na notranji strani garažnih vrat. Če želimo torej garažna vrata odpretni moramo bitjt ali v garaži ali pa moramo imeti pri sebi daljinec. To pa je v vsakdanjem življenju nepraktično, sploh v primeru, ko pri hiši živi veliko ljudi, vsi pa rabijo dostop do garaže.

V tej nalogi bom predstavil svojo idejo, pametna garažna vrata, kot sem si jih zamislil in poskušal realizirati.

1.2 Cilji

Za raziskovalno nalogo, sem si postavil naslednje cilje:

- Garažna vrata bo možno upravljati preko telefona
- Garažna vrata bo možno upravljati preko spletne strani
- Raspberry Pi bo spremljal ali je avto v garaži ali ne in glede na to samodejno zapiral garažna vrata
- Raspberry Pi bo spremljal temperaturo v garaži in jih samodejno zaprl v primeru prenizke ali previsoke temperature
- Raspberry Pi bo samodejno zaprl garažna vrata, če ostanejo odprta po določeni uri
- Raspberry Pi nas bo preko potisnih obvestil obveščal o spremembi stanja garažnih vrat
- Raspberry Pi bo beležil kdo in kdaj je aktiviral garažna vrata
- uporabnik bo imel možnost preklicati samodejno zapiranje vrat

2 Izbira komponent

Ker ne potrebujem veliko precesorske moči, hkrati pa želim, da je moj projekt kar se da kompakten kot krmilnik izberem Raspberry Pi Zero W. To je najmanjša verzija Raspberry Pi-ja, z že vgrajenim WiFi-jem in Bluetoothom. Slednja bosta pri projektu najverjetneje potrebna.

Za upravljanje garažnih vrat bom uporabil 1-kanalni rele. Le tega bom sprogramiral tako, da se bo obnašal kot tipka tj. zaprl se bo za kratek časovni interval prib. 0.5s, nato pa se znova odprl. Nameščen bo v bližini že obstoječe tipke, ki se uporablja za upravljanje garažnih vrat. Z le to bo vzporedno vezan.

Za spremljanje stanja garažnih vrat bom uporabil reed stikala. In sicer dve stikali ter in magnet. Stikali bosta nameščeni na ogrodje vrat, medtem ko bo magnet nameščen neposredno na garažna vrata.

Za spremljanje temperature v garaži uporabim 1-Wire digitalni element.... (nevem imena zle). Le ta bo nameščen nekje v garaži, po možnosti meter od tal, na najmanj prepišnem mestu v garaži.

Ultrazvočni senzor, s katerim bom preverjal ali je avto v garaži ali ne, bo nameščen ali na stropbu garaže, najverjetneje pa kar na motorju garažnih vrat.

Ker želim, da bo mogoče v garaži preveriti ternutno temperaturo ter čas, bom uporabil tudi 16x2 LCD zaslon.

Poleg že naštetih komponent bo uporabil še dve LED diodi in dve tipki. Le te bodo paroma uporabljene kot indikator stanja avta oziroma temperature v garaži. Če bo naprimer garaža odprta in bo vanjo pripeljal avto, se bo pognal program, ki bo po določenem času samodejno zaprl vrata. Istočasno, bo začela utripati ustrezna LED dioda, uporabnik pa bo imel s pritiskom tipke možnost da prekliče samodejno zapiranje garaže. Pri temperaturi je namen LED diode in tipke enak, le da spremljamo temperaturo v garaži.

3 Priprava Raspberry Pi-ja

Da bom lahko uporabljal Raspberry Pi, moram najprej naložiti usterzen opreacijski sistem na Raspberry Pi. Ker za svoj projekt ne potrebujem grafičnega vmesnika, na Raspberry Pi namestim Raspbian Lite. To storim tako, da iz [uradne strani](https://www.raspberrypi.org/downloads/raspbian/) Raspberry Pi prenesem Raspbian Stretch Lite. Nato sledim [navodilom](https://www.raspberrypi.org/documentation/installation/installing-images/README.md) za namestitev operacijskega sistema na microSd kartico, ki jo nato vstavim v Raspberry Pi.

Ker do Raspberry Pi-ja že od samega začetka nimam dostopa preko tipkovnice, moram pred zagonom omogočiti še SSH ter vnesti podatke, ki jih Raspberry Pi potrebuje za povezavo na WiFi dostopno točko. Da omogočim SSH, na boot particijo microSD katrice dodam datoteko ssh. Da pa se bo Raspberry Pi lahko povezal na WiFi dostopno točko, moram na boot particiji ustvariti datoteko wpa_supplicant.conf, v katero vnesem naslednje:

```
1 country=SI
2 ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
3 update_config=1
4 network={
5         ssid="imeDostopneTpočke"
6         psk="geslo"
7         key_mgmt=WPA-PSK
8 }
```

Nato microSD kartico vstavim v Raspberry Pi in priključim napajanje.

Nato se iz terminala na svojem računalniku preko SSH povežem na Raspberry Pi:

```
ssh pi@IP_RaspberryPi #uporabnik pi, privzeto geslo pa je raspberry
```

Ko je povezava vzpostavljena uporabim ukaz *passwd pi*, da spremenim geslo uporabnika pi. Nato zaženem ukaz *sudo raspi-config*

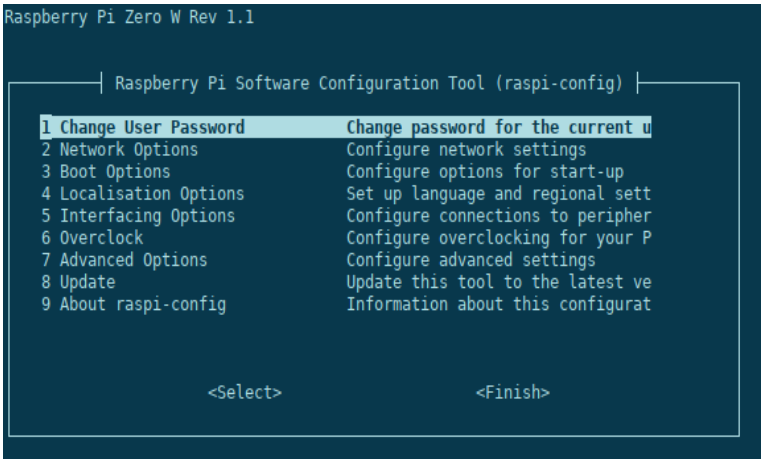

Figure 1: raspi-config: Glavni meni.

Pojavi se meni po katerem se premikam s pomočjo tipkovnice.
Tukaj nastavim vse potrebno:

- hostname spremenim iz ”raspberrypi” v ”projectPi”
- časovno območje nastavim na Europe/Ljubljana
- omogočim SPI, I2C in 1-Wire (vse to bom potreboval za priklop senzorjev)
- razširim datotečni sistem

Preden namestim kakršnekoli nove pakete, je potrebno sistem posodobiti:

```
sudo apt update && sudo apt upgrade -y
```

Ker bom program za upravljanje garažnih vrat napisal v Pythonu, le tega namestim na Raspberry Pi:

```
sudo apt install python python-pip
```

Ker pa bom uporabil tudi LCD, moram namestiti še RPILCD knjižnico:

```
sudo pip install RPLCD
```

S tem je priprava Raspberry Pi-ja zaključena.

4 Priključitev komponent

4.1 Rele

Sam priklop releja je zelo enostaven. Modul z relejem ima tri priključke: VCC, GND in IN1. VCC priključek povežem na 5V pin. GND priključim na GND pin. Za IN1 izberem BMC pin 16.

4.2 Reed stikali

Priključitev reed stikal je v osnovi zelo enostavna. Paziti je treba, da imamo NO (Normally Open) in ne NC (Normally Closed) stikala. Nato en del stikala povežemo na 3.3V, drugi del pa na enega izmed GPIO pinov. Zaj sem za eno stikalo (garaža odprta) priklopil na BCM pin 5, drugega (garaža zaprta) pa na BCM pin 6.

Po priklopu, je treba paziti, da v programu ne pozabimo nastaviti pull-down uporov, oba pina pa morata biti nastavljena kot vhodna. Tako ima pin vrednost 1, če je stikalo zaprto in vrednost 0, če je stikalo odprto.

4.3 Ultrazvočni senzor razdalje HC-SR04

Ker je bilo to prvič, da sem uporabljal ultrazvočni senzor razdalje na Raspberry Pi, sem si pomagal z vodičem na ModMyPi[1].

Senzor ima štiri pine: VCC, GND, Trig, Echo. VCC povežem na 5V. GND, povežem na GND pin, Trig na BCM pin 11, Echo pa najprej preko 1 kΩ upora na BCM pin 20, nato pa še preko 2 kΩ na GND.

Da pridobim razdaljo, uporabim naslednjo metodo:

```
1  def checkCar():
2      GPIO.output(GPIO_VARS_DICT['TRIG'], False)
3      time.sleep(0.001)
4
5      GPIO.output(GPIO_VARS_DICT['TRIG'], True)
6      time.sleep(0.00001)
7      GPIO.output(GPIO_VARS_DICT['TRIG'], False)
8
9      while GPIO.input(GPIO_VARS_DICT['ECHO'])==0:
10       pulse_start = time.time()
11
12     while GPIO.input(GPIO_VARS_DICT['ECHO'])==1:
13       pulse_end = time.time()
14
15     pulse_duration = pulse_end - pulse_start
16     distance = pulse_duration * 17150
17
18     return round(distance, 2)
```

4.4 DS18B20 temperaturni senzor

Ker je bilo to prvič, da sem v projektu uporabil DS18B20 senzor temperature, sem si pomagal z vodičem na [PiMyLifeUp](https://pimylifeup.com/raspberry-pi-temperature-sensor/).

4.5 LCD zaslon

4.6 Tipki in LED diodi

5 Programske rešitve

5.1 Premikanje garažnih vrat

```
1  def toggleGarage():
2      GPIO.output(GPIO_VARS_DICT['RELAY'], 0)
```

```
    time.sleep(.5)
    GPIO.output(GPIO_VARS_DICT['RELAY'], 1)
```

5.2 Preveranje stanja garažnih vrat

```
def checkDoor():
    if GPIO.input(GPIO_VARS_DICT['REED_OPEN']) == True:
        return "odprta"
    elif GPIO.input(GPIO_VARS_DICT['REED_CLOSED']) == True:
        return "zaprta"
    else:
        return "priprta"
```

5.3 Spremljanje temperature

5.3.1 Izpisovanje temperature na LCD

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
from datetime import datetime
import ConfigParser
from RPLCD.gpio import CharLCD
import RPi.GPIO as GPIO
import argparse
import os
import time
import glob
import sys

if __name__=="__main__":
    pid = str(os.getpid())
    pidfile = "/tmp/LCD_temp.pid"

    if os.path.isfile(pidfile):
        print "%s already exists, exiting" % pidfile
        sys.exit()
    file(pidfile, 'w').write(pid)
    try:
        init()
        lcd.clear()
        # Do some actual work here
        while True:
                lcd_write(datetime.now().strftime("%d.%m.%y  %H:%M"), "Temp: "
                ↪  + str(read_temp())+unichr(223)+"C")
                time.sleep(1)
    except:
        os.unlink(pidfile)
        #os.delete(pidfile)
        destroy()
```

5.3.2 Samodejno zapiranje ob neprimerni temperaturi

```
def monitorTemp():
    time.sleep(TIMEOUTS_VARS_DICT['BEGIN_TEMP_WATCH'])

    ↪ GPIO.add_event_detect(GPIO_VARS_DICT['OVERRIDE_TEMP'],GPIO.RISING,bouncetime=300)
    count = 0
    while 1:
        blink(GPIO_VARS_DICT['LED_MONITOR_TEMP'])
        if GPIO.event_detected(GPIO_VARS_DICT['OVERRIDE_TEMP']):
            break
        elif checkDoor() == 'zaprta':
            break
        temp = read_temp()
        if temp < TEMP_VARS_DICT['MIN_TEMP']:
            toggleGarage()
            pushover.send_message("Temperatura v garaži prenizka! Zapiram
            ↪ garažo!", title="Garaža prehladna!")
            while checkDoor() != "zaprta":
                time.sleep(1)
            pushover.send_message("Garaža zaprta zaradi prenizke temperature!",
            ↪ title="Garaža zarta!")
            time.sleep(2)
            break;
        elif temp > TEMP_VARS_DICT['MAX_TEMP']:
            toggleGarage()
            pushover.send_message("Temperatura v garaži previsoka! Zapiram
            ↪ garažo!", title="Garaža pretopla!")
            while checkDoor() != "zaprta":
                time.sleep(1)
            pushover.send_message("Garaža zaprta zaradi previsoke
            ↪ temperature!", title="Garaža zarta!")
            time.sleep(2)
            break;
        count += 1
```

5.4 Samodejno zapiranje glede na avto

```
def monitorCar():

    ↪ GPIO.add_event_detect(GPIO_VARS_DICT['OVERRIDE_CAR'],GPIO.RISING,bouncetime=300)
    distance = checkCar()
    for x in range(0,TIMEOUTS_VARS_DICT['CAR_STATUS_TIMEOUT']):
        if GPIO.event_detected(GPIO_VARS_DICT['OVERRIDE_CAR']):
            break
        elif checkDoor() == 'zaprta':
            break
        blink(GPIO_VARS_DICT['LED_MONITOR_CAR'])
        if (distance >=25 and checkCar() <= 20) or (distance <=20 and
        ↪ checkCar() >= 25):
```

```
            for i in range(0,5):
                blink(GPIO_VARS_DICT['LED_MONITOR_CAR'])
            if GPIO.event_detected(GPIO_VARS_DICT['OVERRIDE_CAR']):
                break
            elif checkDoor() != 'odprta':
                break
            if distance >=25 and checkCar() <= 20:
                toggleGarage()
                while checkDoor() != "zaprta":
                    time.sleep(5)
                time.sleep(2)
                break
            elif distance <=20 and checkCar() >= 25:
                toggleGarage()
                pushover.send_message("Avto odpeljal! Zapiram garažo!",
                 ↪  title="Garaža")
                while checkDoor() != "zaprta":
                    time.sleep(5)
                    pushover.send_message("Avto odpeljal! Garaža zaprta!",
                     ↪  title="Garaža")
                time.sleep(2)
                break
```

5.5 Če garažnih vrat ni možno zapreti

```
def doorAjar():
    for attempts in range(0, TIMEOUTS_VARS_DICT['AJAR_CLOSE_ATTEMPTS']):
        toggleGarage()
        for x in range(0, TIMEOUTS_VARS_DICT['AJAR_TIMEOUT']):
            if checkDoor() != 'priprta':
                break
            time.sleep(1)
        if checkDoor() == 'zaprta':
            break
        elif checkDoor() == 'odprta':
            closeDoor()
```

5.6 Konfiguracijska datoteka

5.6.1 Datoteka

```
[gpio]
TRIG = 11
ECHO = 20
RELAY = 16
OVERRIDE_CAR = 26
OVERRIDE_TEMP = 12
LED_MONITOR_CAR = 13
LED_MONITOR_TEMP = 21
REED_OPEN = 5
```

```
10 REED_CLOSED = 6
```

5.6.2 Branje iz datoteke

```
1 def readConf(section, vars, val_dict):
2     configParser = ConfigParser.RawConfigParser(allow_no_value=True)
3     configParser.read(os.environ['HOME']+'/.garage/garage.conf')
4     for value in vars:
5         val_dict[value] = int(configParser.get(section, value))
```

6 Zaključek

Acronyms

SSH Secure Shell. *Glossary:*

Bibliography

[1] ModMyPi LTD. HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi. `https://www.modmypi.com/blog/hc-sr04-ultrasonic-range-sensor-on-the-raspberry-pi`. Zadnji dostop: 15. 2. 2019.