Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Žiga Korelc

Razvoj varnostnega sistema za zaščito doma z mikrokrmilnikom Arduino

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: prof. dr. Branko Šter

Ljubljana, 2018

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Zasnujte in izdelajte cenovno dostopen varnostni sistem, ki vsebuje vse osnovne funkcije za zaščito doma pred vlomilci. Uporabite mikrokrmilnik in ustrezne senzorje, kot so IR senzor plamena, IR senzorji gibanja, I2C LCD zaslon, pasivni piskači, senzor vlage in temperature ter senzor nagiba. Za programiranje senzorjev lahko uporabite že obstoječe knjižnice. Preverite delovanje sistema na maketi hiše, na koncu pa ocenite tudi ceno izdelka.

Zahvala gre mentorju, prof. dr. Branku Šteru, za nasvete in podporo pri izdelavi diplomske naloge. Posebna zahvala pa gre še moji družini za podporo pri študiju.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
I^2C	Inter-Integrated Circuit	Tip serijskega vodila
LCD	Liquid Cristal Display	Tekočekristalni zaslon
GSM	Global System for Mobile communications	Standard mobilnih komunikacij
IR	Infrared radiation	Infrardeče sevanje
IDE	Integrated development environment	Integrirano razvojno okolje
GPRS	General Packet Radio Service	Mobilna podatkovna storitev v okviru standarda GSM
SIM	Subscriber Identity Module	Kartica ponudnika mobilnih storitev
UART	Universal Asynchronous Receiver-Transmitter	Vezje za serijski prenos podatkov

Povzetek

Naslov: Razvoj varnostnega sistema za zaščito doma z mikrokrmilnikom Arduino

Avtor: Žiga Korelc

Varnost je za ljudi vedno bolj pomembna, kar je privedlo do razvoja varnostnih sistemov. Sprva so bili varnostni sistemi nedostopni večini populacije, vendar se je z razvojem tehnologije in cenovno bolj dostopnimi komponentami to začelo spreminjati. Sam sem se lotil problema z vidika cenovne dostopnosti varnostnega sistema, ki vsebuje vse osnovne funkcije za zaščito doma pred vlomilci. Cilj diplomske naloge je bil zato izdelava varnostnega sistema za zaščito doma z uporabo mikrokrmilnika Arduino in senzorjev, namenjenih mikrokrmilniku. V ta namen sem uporabil IR senzor plamena, IR senzorje gibanja, I^2C LCD zaslon, pasivne piskače, senzor vlage in temperature in senzor nagiba. Ker je podpora za uporabo in razvoj programske opreme za mikrokrmilnik Arduino obsežna, mi je to omogočilo uporabo številnih že obstoječih knjižnic za programiranje senzorjev. Ko je bila programska koda za senzorje napisana, pa jih je bilo potrebno še testirati. Ko so senzorji v prototipu delovali pa so bili nameščeni na maketo hiše, ki služi tako za demonstracijo delovanja varnostnega sistema, kot tudi lažjo uporabo in testiranje samega sistema.

Ključne besede: Arduino, varnostni sistem, zaščita, senzor, C.

Abstract

Title: Development of security system for house protection based on Arduino microcontroller

Author: Žiga Korelc

To people, the sense of security is becoming more important each day, which has led to development of security systems. Initially, security systems were inaccessible to the majority of the population, but with the development of technology and more affordable components, this began to change. I have dealt with the problem in terms of affordability of the security system, which contains all the basic functions to protect your home from burglaries. The goal of my diploma was therefore the production of a home security system based on Arduino microcontroller and sensors intended for the microcontroller. For this purpose I used and IR flame sensor, IR motion sensors, an I^2C LCD screen, passive buzzers, humidity and temperature sensor and a tilt sensor. Because the support for use and development of Arduino microcontroller software is extensive, it has enabled me to use a number of existing libraries for programming of the sensors. When the program code for sensors was finished, they had to be tested. When the prototype was functional and working as it is supposed to, the prototype has been installed in a model house, which serves for demonstrating the functioning security system. But this also serves for easier use and testing of the system itself.

Keywords: Arduino, security system, protection, sensor, C.

Poglavje 1

Uvod

Varnost je bila za človeka vedno pomembna, vendar je zaradi porasta kriminala in vlomov v zadnjem času postala še pomembnejša. V novih gradnjah je vgradnja varnostnega sistema že skoraj nekaj samoumevnega in nujnega. Zato je v zadnjem času cena varnostnih sistemov postala dostopnejša večji populaciji, vendar pa še zmeraj predstavlja kar visok strošek, še posebno če se ga odločimo vgraditi naknadno, saj je potrebno hišo ali stanovanje pripraviti na vgradnjo takega sistema.

Ravno zaradi stroška, ki ga varnostni sistem predstavlja, sem za namen diplomskega dela izbral izdelavo cenovno ugodnejšega varnostnega sistema, ki temelji na mikrokrmilniku Arduino in senzorjih, ki so izdelani za mikrokrmilnik.

Preden pa se začne tak sistem načrtovati in izdelovati, je potrebno poznati potrebe in specifikacije objekta, kamor se bo sistem vgrajeval. V pomoč pri načrtovanju varnostnega sistema nam bo tudi pregled aktualne ponudbe pri obstoječih ponudnikih varnostnih sistemov.

Programiranje mikrokrmilnika Arduino je potekalo v namenskem razvojnem okolju Arduino IDE. Končni izdelek pa bo vgrajen v maketo hiše, saj to omogoča tudi lažje testiranje delovanja vseh senzorjev.

V nadaljevanju je najprej predstavljena vsa strojna oprema, uporabljena pri razvoju varnostnega sistema, in nato postopek vezave vse te strojne

opreme v celoten varnostni sistem. Sledi opis postopka programiranja v Arduino IDE in testiranje delovanja sistema pred vgradnjo. Za konec bom predstavil še izgradnjo makete in vgradnjo sistema.

1.1 Pregled področja varnostnih sistemov

S to temo so se do sedaj na FRI ukvarjali predvsem v številnih diplomskih nalogah. En primer uporabe mikrokrmilnika Arduino je pri diplomski nalogi, kjer se avtor ukvarja z varnostnim vidikom pri implementaciji pametne hiše in tudi izdela centralno enoto [34]. Drug primer diplomske naloge je bila digitalizacija in avtomatizacija vrta z uporabo mikrokrmilnika Arduino, kjer ga avtor uporablja za avtomatizirano skrb za vrt preko aplikacije [33]. Še ena zanimiva uporaba mikrokrmilnika Arduino pa je pri razvoju sistema za nadzor doma, saj je potrebno premisliti, katere komponente izbrati in iz njih razviti sistem, kot je to storil avtor diplomske naloge [35]. V Sloveniji obstaja podjetje MOBICOM d.o.o. [23], ki se ukvarja s prodajo protivlomnih sistemov, video-nadzornih sistemov in protipožarnih sistemov. Omogočajo informativne izračune za varnostne sisteme, ki jih konfiguriramo povsem sami, torej sami izberemo vse komponente. Seveda pa tudi pri informativnih izračunih pišejo navodila za izbiro sistema, ki bo deloval, ne pa, da kupimo alarmno centralo, ki podpira 4 senzorje, nato pa kupimo še 8 senzorjev, kar bi pomenilo, da na centralo ne bomo mogli priključiti vseh senzorjev, ki smo jih kupili.

Poglavje 2

Opis uporabljenih programskih orodij, naprav in tehnologij

V tem poglavju bodo opisana programska orodja, naprave in tehnologije, ki sem jih potreboval in uporabil pri izdelavi diplomske naloge.

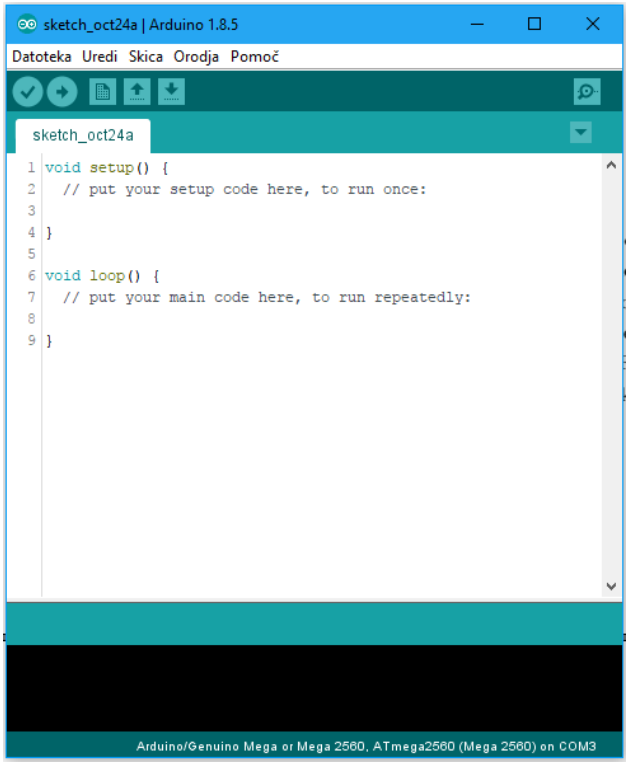
2.1 Arduino IDE

Arduino IDE [4] je odprtokodno programsko orodje, namenjeno programiranju mikrokrmilnika Arduino. Trenutno je najnovejša različica 1.8.5 in jo lahko poganjamo na operacijskem sistemu Windows, Mac OS X in Linux. To programsko orodje nam omogoča programiranje vseh različic mikrokrmilnika Arduino ali pa cenejše alternative kot so Funduino, Genuino in podobni. Mikrokrmilnike programiramo v programskem jeziku osnovanemu na C/C++ [26].

Samo programsko orodje je dokaj enostavno, kar je vidno na Sliki 2.1. Kot osnovo za delovanje programov ima mikrokrmilnik Arduino dve glavni funkciji:

- Funkcija `setup()` se izvede ob zagonu programa in se jo uporabi za inicializacijo spremenljivk, uporabo knjižnic in določitev načina delovanja posameznih vhodov in izhodov

- Funkcija `loop()` pa se nepretrgoma izvaja v zanki



Slika 2.1: Programsko orodje Arduino IDE

2.2 Arduino

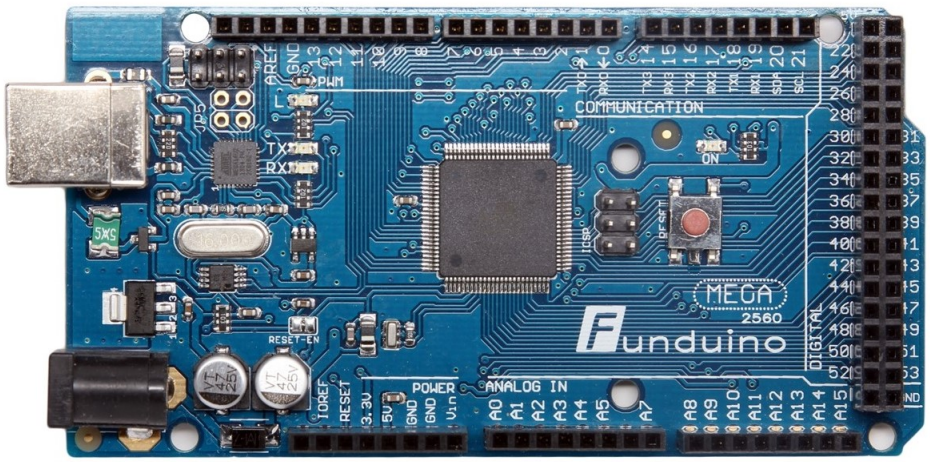
Arduino je odprtokodna platforma [3], ki temelji na uporabi strojne in programske opreme. Ponudba različnih mikrokrmilnikov je obsežna, vsem pa je skupno to, da vsebujejo mikroprocesor, digitalne vhode in izhode, analogne vhode in izhode, ter nekaj pomnilnika za programsko kodo in spremenljivke. Mikrokrmilnik se programira v programskem orodju Arduino IDE, nato pa se napisano programsko kodo preko USB kabla prenese na mikrokrmilnik, kjer se začne izvajati.

Programiranje mikrokrmilnika je dobro podprto na številnih forumih in pa s številnimi že obstoječimi knjižnicami, ki jih lahko vključimo v naš projekt. Prav tako obstaja veliko že narejenih projektov in primerov [7]. V

Tabeli 2.1 je narejena primerjava najbolj pogostih mikrokrmilnikov Arduino in njihovih najpomembnejših lastnosti [25].

Tabela 2.1: Primerjava mikrokrmilnikov Arduino

Mikrokrmilnik	**Procesor**	**Analogni V/I**	**Digitalni VI/PWM**	**EEPROM[kB]**	**SRAM[kB]**	**FLASH[kB]**
101	Intel Curie	6/0	14/4	-	24	196
Uno	ATmega328P	6/0	14/6	1	2	32
Pro	ATmega168	6/0	14/6	0,512	1	16
	ATmega328P			1	2	32
Nano	ATmega168	8/0	14/6	0,512	1	16
	ATmega328P			1	2	32
Micro	ATmega32U4	12/0	20/7	1	2,5	32
LilyPad	ATmega168V	6/0	14/6	0,512	1	16
	ATmega328P					
Leonardo	ATmega32U4	12/0	20/7	1	2,5	32
Mega 2560	ATmega2560	16/0	54/15	4	8	256



Slika 2.2: Mikrokrmilnik Funduino Mega 2560

Zaradi obsežnosti programske kode sem že v začetku kot temelj diplomske naloge izbral Funduino Mega 2560, ki ga vidimo na Sliki 2.2. Za Funduino namesto Arduina sem se odločil zaradi cenovne dostopnosti, saj je zmogljivost identična mikrokrmilniku Arduino Mega 2560. Funduino Mega 2560 stane približno 18 € ali celo 11 € v primeru znižanja [9] na spletni strani kot je

DealExtreme, medtem ko Arduino Mega 2560 na uradni strani stane 35 € [5], če pa ga želimo kupiti v Sloveniji, pa stane skoraj 58 € v trgovini Conrad [6].

Sam mikrokrmilnik ponuja 8KB pomnilnika SRAM in 256KB bliskovnega pomnilnika (FLASH), kar je več kot dovolj za celotno programsko kodo. Prav tako pa ponuja mikrokrmilnik 4 pare serijskih priključkov in s tem pripomore k lažji vezavi večih modulov, ki za delovanje potrebujejo serijsko komunikacijo. V moji diplomski nalogi je preko serijskega priključka vezan le GSM modul, vendar pa z dodatnimi serijskimi priključki puščam možnost dodajanja novih modulov. Prav tako pa nam zaradi večjega števila serijskih priključkov ni potrebno v programski kodi uporabljati programskih serijskih priključkov, kar omogoča prihranek bliskovnega pomnilnika zaradi krajše programske kode.

2.3 GSM modul

V diplomski nalogi sem uporabil GSM modul SIM900, ki ga vidimo na Sliki 2.3. Modul nam omogoča številne GPRS funkcije, kot je pošiljanje SMS ali povezovanje na GPRS omrežje. V diplomski nalogi sem uporabljal modul za pošiljanje SMS sporočil glede na sproženi alarm v primeru vdora v hišo. Za delovanje je zaradi tega potrebna še SIM kartica, saj brez nje ne moremo pošiljati SMS sporočil.

Slika 2.3: GSM modul SIM900

Poglavje 3

Razvoj strojne opreme varnostnega sistema

V tem poglavju je podrobnejši opis vezave mikrokrmilnika Arduino, vseh senzorjev, števílčnice, LCD zaslona in GSM modula SIM900.

3.1 GSM modul SIM900

Ker mikrokrmilnik Arduino ni zmožen pošiljati SMS sporočil, smo se odločili za uporabo temu namenjenega GSM modula SIM900. SIM900 uporablja serijsko komunikacijo (preko RS-232) za komuniciranje z mikrokrmilnikom Arduino. Vezava modula na mikrokrmilnik je vidna v Tabeli 3.1.

Tabela 3.1: Vezava GSM modula SIM900

Mikrokrmilnik	**GSM modul SIM900**	**Angleški opis**	**Slovenski opis**
GND	GND	Ground	Ozemljitev
TXD	RX3	Receiving	Sprejemanje
RXD	TX3	Transmitting	Oddajanje

Pred začetkom uporabe je potrebno GSM modul SIM900 konfigurirati. Konfiguracija poteka preko AT ukazov, ki so posredovani preko RS-232 povezave med mikrokrmilnikom in modulom. Ime AT ukazov izhaja iz besede

ATtention citeBasicAT. Za vsak modul, ki uporablja AT ukaze, se ukazi razlikujejo, kar pomeni, da posamezni AT ukazi nosijo poseben pomen za napravo, ki jo nastavljamo [8], v našem primeru GSM modul SIM900. Modul je za uporabo potrebno spraviti v način delovanja AT, kar dosežemo s pritiskom na tipko na modulu. Ko je modul v načinu AT, začne poleg LED za napajanje utripati še dodatna LED lučka, ki nam pokaže, da je GSM modul v načinu AT in prijavljen na omrežje ponudnika mobilnih storitev.

Zaporedje ukazov, s katerimi smo izvedli konfiguracijo GSM modula SIM900:

1. AT - Če dobimo odgovor modula OK, pomeni, da modul deluje v načinu AT.

2. AT+CMGF=1 - S tem izberemo tip SMS sporočila. 1 pomeni, da bo modul deloval v načinu teksta.

3. AT+CMGS="<mednarodna klicna koda><telefonska številka>" - S tem določimo telefonsko številko, na katero želimo poslati SMS. Istočasno pošlje ta ukaz GSM modul v čakanje na besedilo, ki ga želimo poslati.

4. Vpis besedila, ki ga želimo poslati.

5. \x1A - V programski kodi je to zamenjava za kombinacijo tipk <CTRL>+<Z>. Ta ukaz pove GSM modulu SIM900, da naj pošlje vpisano besedilo. Prav tako se vrne GSM modul v stanje čakanja novih AT ukazov.

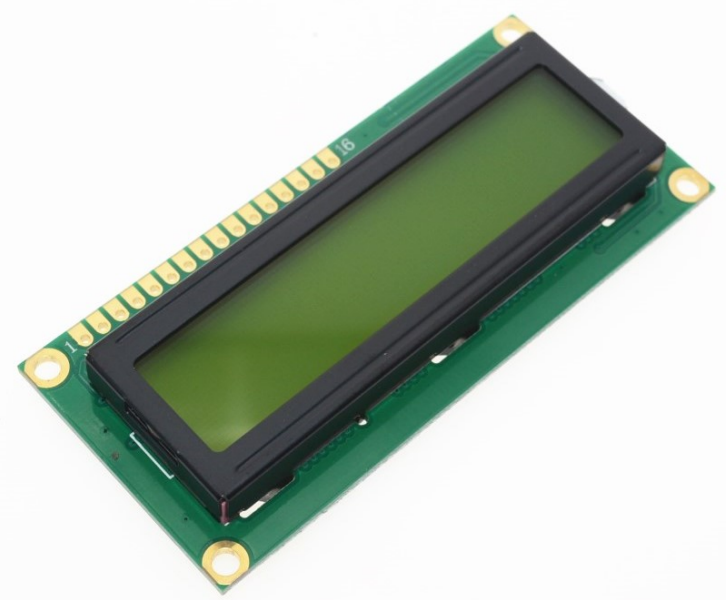
3.2 LCD zaslon

Za izpisovanje trenutnega stanja varnostnega sistema sem uporabil LCD zaslon za mikrokrmilnikom Arduino velikosti 16*2, ki ga vidimo na Sliki 3.1. Zaslon ima 2 vrstici, vsaka vrstica pa lahko prikazuje 16 znakov. LCD zaslon ima modro osvetlitev in 16 priključkov, s katerimi ga lahko povežemo na mikrokrmilnik. V svojem primeru sem uporabil LCD zaslon z integriranim vezjem (Slika 3.2), ki po standardu I^2C poenostavi vezavo na mikrokrmilnik, saj ga povežemo z uporabo le 4 žic, prav tako pa poenostavi nadzorovanje

LCD zaslona. Na vezju imamo tudi nadzor nad kontrastom zaslona. Vezava LCD zaslona na mikrokrmilnik je vidna v Tabeli 3.2.

Tabela 3.2: Vezava LCD zaslona

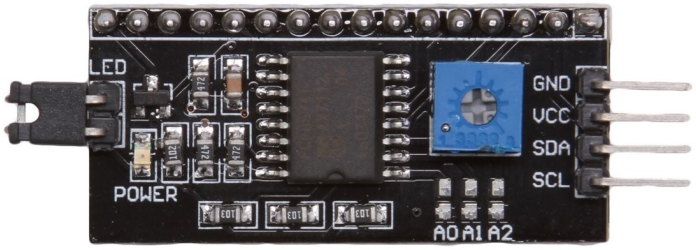
Mikrokrmilnik	**LCD zaslon**	**Angleški opis**	**Slovenski opis**
GND	GND	Ground	Ozemljitev
5V	VCC	Power	Napajanje
SDA	SDA	Data Line	Podatki
SCL	SCL	Clock Line	Ura



Slika 3.1: LCD zaslon velikosti 16*2

3.2.1 Protokol I^{2}C

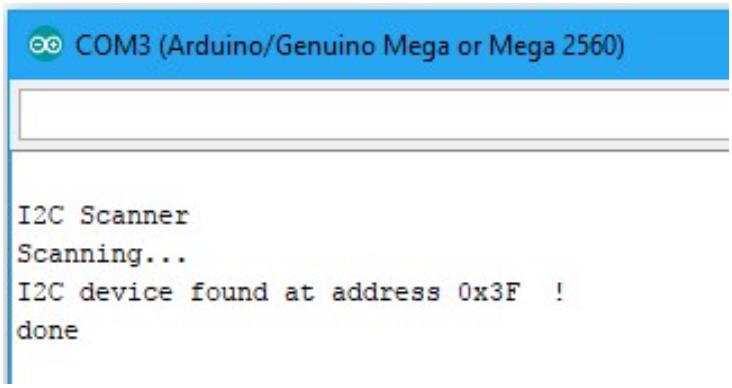
Protokol I^{2}C je poseben tip vodila, ki ga je razvilo podjetje Philips Semiconductor leta 1982. Tipična uporaba je priključitev vgrajenih vezij, ki za delovanje potrebujejo nižje hitrosti, na procesorje in mikrokrmilnike. Vodilo sestavljata dve žici. SDA za prenos podatkov in SCL za prenos urinega



Slika 3.2: I²C vezje

signala. Obe žici sta dvosmerni, saj je potreben prenos podatkov med gospodarjem in sužnjem. Vsaka I²C naprava ima svoj unikatni naslov, ki je vnaprej določen, lahko pa se ga tudi spreminja. Komunikacija poteka med gospodarjem, ki generira urin signal, ki ga skupaj z ukazi pošlje sužnju, in pa sužnjem, ki gospodarju odgovarja takrat, ko gospodar želi odgovor [11].

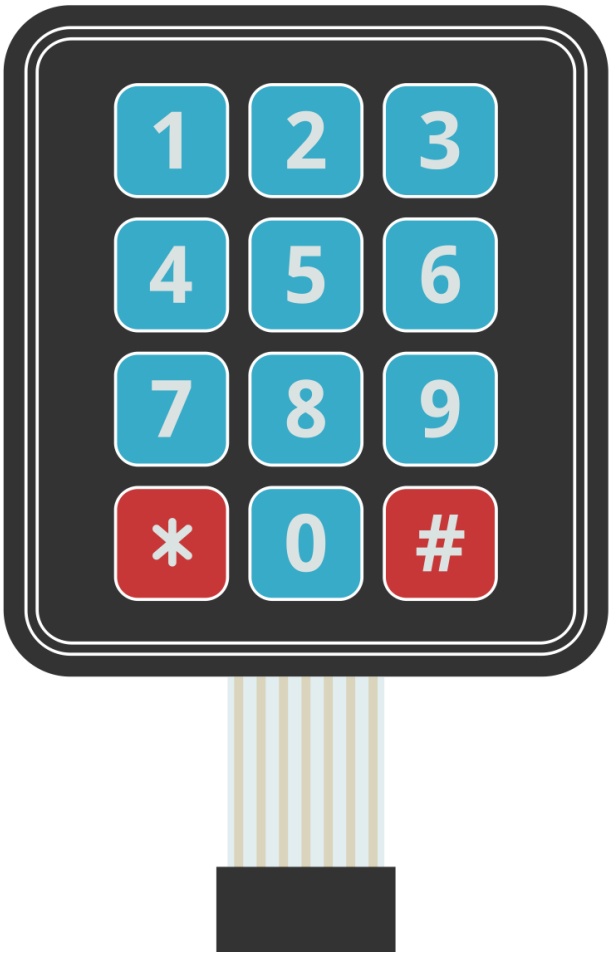
Da bi lahko LCD zaslon priključili na mikrokrmilnik Arduino preko I²C vezja, je najprej treba poznati unikaten naslov LCD zaslona. Ker tega podatka ni bilo navedenega ob nakupu LCD zaslona, sem uporabil že napisano Arduino programsko kodo [12]. Ta programska koda zazna vse I²C naprave, ki so priključene na mikrokrmilnik Arduino, na katerem se izvaja, in nam izpiše vse njihove unikatne naslove, kot je vidno na Sliki 3.3. Kot je razvidno s Slike 3.3, ima moj I²C LCD zaslon naslov `0x3F`, katerega potrebujem pri pisanju programske kode.



Slika 3.3: Izpis programske kode I²C Scanner

3.3 Številska tipkovnica

Da lahko sistem aktiviramo in deaktiviramo, je potrebna številska tipkovnica. Za svoj varnostni sistem sem uporabil številsko tipkovnico za mikrokrmilnik Arduino velikosti 4*3, vidno na Sliki 3.4. Tipke tipkovnice so povezane z žicami po vrsticah in stolpcih, kar nam omogoči, da za povezavo tipkovnice na mikrokrmilnik Arduino potrebujemo le 7 žic, 3 za stolpce in 4 za vrstice. To nam prav tako omogoči prepoznavo katera tipka je bila pritisnjena. Vezava tipkovnice z mikrokrmilnik je vidna v Tabeli 3.3.



Slika 3.4: Številska tipkovnica

Tabela 3.3: Vezava številske tipkovnice

Mikrokrmilnik	Številska tipkovnica	Angleški opis	Slovenski opis
pin 8	pin 1	Column 3	Stolpec 3
pin 7	pin 2	Column 2	Stolpec 2
pin 6	pin 3	Column 1	Stolpec 1
pin 5	pin 4	Row 4	Vrstica 4
pin 4	pin 5	Row 3	Vrstica 3
pin 3	pin 6	Row 2	Vrstica 2
pin 2	pin 7	Row 1	Vrstica 1

3.4 Senzorji

Glavni del varnostnega sistema so poleg mikrokrmilnika Arduino, ki sistem upravlja, še senzorji, ki mikrokrmilnik opozorijo na dogodek. S tem mikrokrmilnik ve, da se je zgodilo nekaj, kar je potrebno preprečiti s sprožitvijo alarma in opozorilom lastnika hiše.

3.4.1 IR senzor HC-SR505

V svojem varnostnem sistemu sem uporabil 3 IR senzorje HC-SR505 za zaznavanje gibanja, vidne na sliki 3.5. Prednost izbranega senzorja je visoka občutljivost, kar privede do hitrega zaznavanja gibanja, 100° kot zaznavanja, 3m zaznavne razdalje in pa nizka električna poraba [15]. Senzorji gibanja so lahko uporabljeni po celotni stavbi, vendar so najbolj učinkoviti pri zaznavanju vloma skozi okno, saj en senzor s strani brez težav pokrije celotno višino okna. V Tabeli 3.4 je vidna vezava 3 IR senzorjev HC-SR505 z mikrokrmilnikom Arduino.

3.4.2 Pasivni piskač

Za zvočni alarm sem v varnostnem sistemu uporabil 3 pasivne piskače, ki jih vidimo na Sliki 3.6. Piskač deluje glede na zapis signala na njegov vhod. Ob

Slika 3.5: IR senzor za zaznavanje gibanja

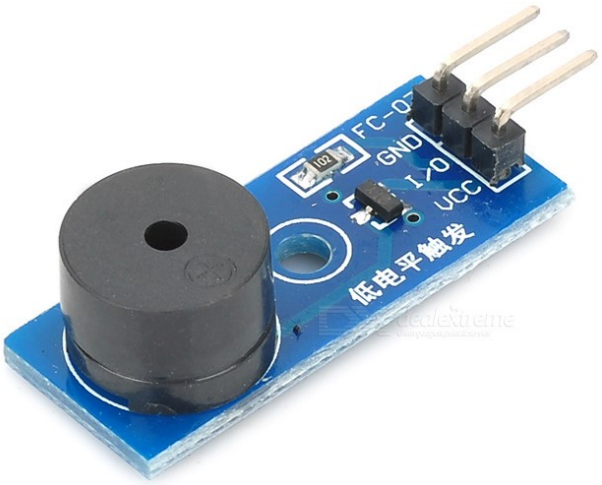
Tabela 3.4: Vezava IR senzorja HC-SR505

Mikrokrmilnik	**IR senzor HC-SR505**	**Angleški opis**	**Slovenski opis**
5V	VCC	Power	Napajanje
GND	GND	Ground	Ozemljitev
pin 22 pin 24 pin 26	OUT	Output	Izhod

visokem pulzu se piskač oglasi, nizek pulz pa je potreben za ponovno aktivacijo piskača. Mikrokrmilnik Arduino sproži piskača, če se sproži katerikoli od senzorjev v varnostnem sistemu, ko je le-ta aktiviran. Prav tako se sproži vsako sekundo po aktivaciji sistema in s tem uporabniku odšteva sekunde do aktivacije sistema. Vezava piskačev na mikrokrmilnik Arduino je vidna v Tabeli 3.5.

3.4.3 IR senzor plamena

V svojem varnostnem sistemu sem predvidel tudi splošno varnost hiše, katero želimo zavarovati, zato sem vključil IR senzor plamena, ki ga vidimo na Sliki 3.7. Senzor je občutljiv in deluje na IR valovni dolžini med 760nm in



Slika 3.6: Pasivni piskač

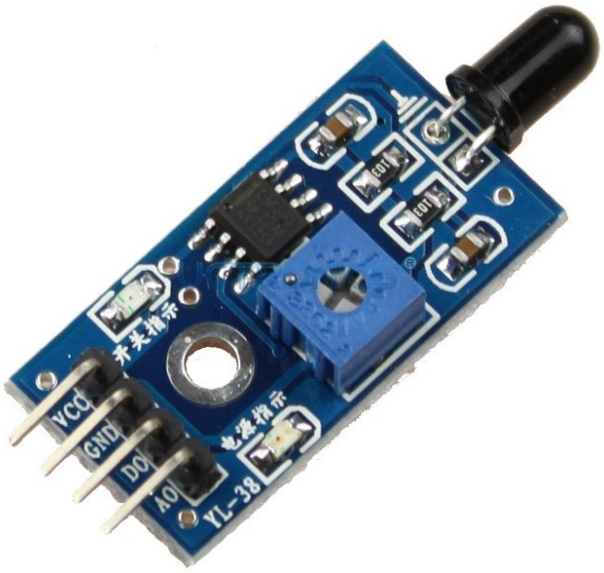
Tabela 3.5: Vezava pasivnega piskača

Mikrokrmilnik	**Pasivni piskač**	**Angleški opis**	**Slovenski opis**
5V	VCC	Power	Napajanje
GND	GND	Ground	Ozemljitev
pin 23 pin 25 pin 27	I/O	Input/Output	Vhod/Izhod

1100nm, kar nato z analognim izhodom prikaže kot število v območju med 0 in 1024, programska koda pa nato določi, ali je zaznan plamen ali pa ne [30]. S senzorjem plamena poskrbimo, da je hiša zaščitena v primeru požara, kar je v krajih, kjer je večja verjetnost požarov, dobrodošla funkcija. Vezava IR senzorja plamena pa je prikazana v Tabeli 3.6.

3.4.4 Senzor nagiba SW-520D

Za vključitev senzorja nagiba SW-520D v varnostni sistem sem se odločil, ker ga lahko uporabimo za zaznavanje, ali je bila kljuka na vratih prema-

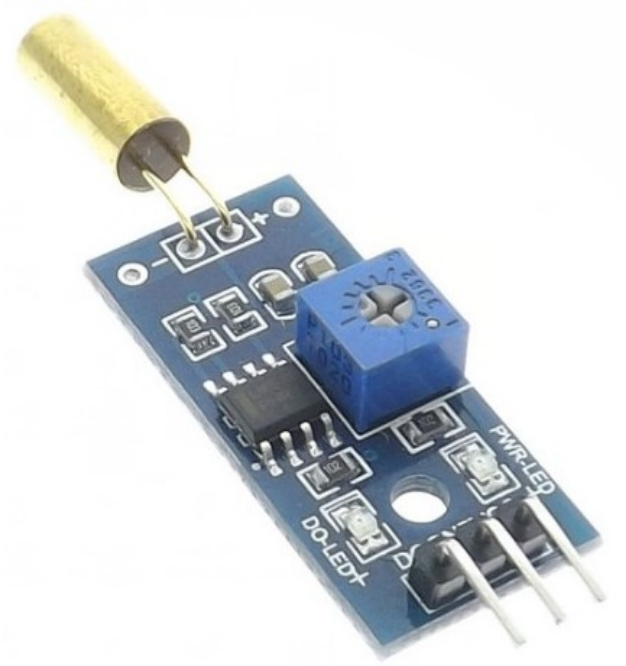


Slika 3.7: IR senzor plamena

Tabela 3.6: Vezava IR senzorja plamena

Mikrokrmilnik	**IR senzor plamena**	**Angleški opis**	**Slovenski opis**
5V	VCC	Power	Napajanje
GND	GND	Ground	Ozemljitev
A0	A0	Analog0	Analogni priključek

knjena. Kljuka se lahko premakne ob našem vstopu v hišo ali pa ob vlomu skozi vrata. V obeh primerih imamo nekaj časa za deaktivacijo varnostnega sistema, vendar bo varnostni sistem v primeru vloma še vedno aktiven, saj vlomilci ne poznajo lokacije in varnostne kode, kar bo privedlo do aktivacije alarma. Uporabljen je bil senzor nagiba SW-520D, ki ga vidimo na Sliki 3.8. Deluje tako, da ima v cevki 2 kroglici, ki se prosto premikata po cevki. Ko je nagib večji od 45°, se bosta kroglici dotaknili žic na koncu cevke, kar sklene tokokrog [28]. Vezava senzorja nagiba SW-520D je prikazana v Tabeli 3.7.



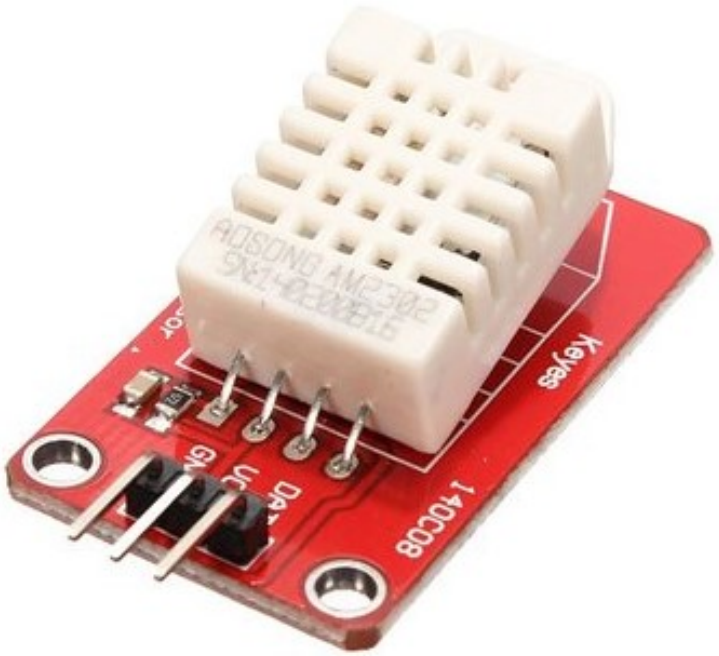
Slika 3.8: Senzor nagiba SW-520D

Tabela 3.7: Vezava senzorja nagiba SW-520D

Mikrokrmilnik	**Senzor nagiba SW-520D**	**Angleški opis**	**Slovenski opis**
3,3V	VCC	Power	Napajanje
GND	GND	Ground	Ozemljitev
pin 30	D0	Data0	Podatki

3.4.5 Senzor vlage in temperature DHT22

Kot dodatek za kakovost življenja pa sem v varnostni sistem dodal še senzor vlage in temperature DHT22, ki ga vidimo na Sliki 3.9. Senzor zaznava vlago in temperaturo prostorov ali okolice, kar lahko ob namestitvi v problematične prostore izkoristimo za informacijo, ali je prostor primerno prezračen in ogrevan, ali pa je potrebno karkoli spremeniti. Najbolj uporabljana DHT senzorja sta DHT11 in pa DHT22 [24]. Primerjava glavnih lastnosti med senzorjema je prikazana v Tabeli 3.8. Vezava senzorja DHT22 pa je prikazana v Tabeli 3.9.



Slika 3.9: Senzor vlage in temperature DHT22

Tabela 3.8: Primerjava senzorja DHT11 in DHT22

Senzor	**Območje vlage[%]**	**Natančnost[%]**	**Območje temperature[°C]**	**Hitrost zajemanja[Hz]**
DHT11	20 do 80	5	0 do 50	1
DHT22	0 do 100	2 do 5	-40 do 125	0,5

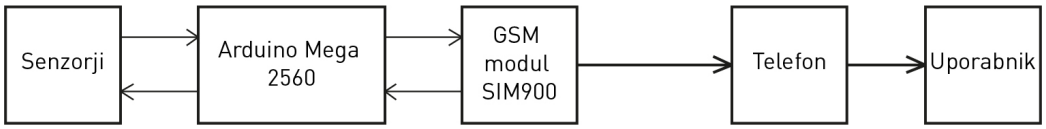
Tabela 3.9: Vezava senzorja vlage in temperature DHT22

Mikrokrmilnik	**Senzor DHT22**	**Angleški opis**	**Slovenski opis**
5V	VCC	Power	Napajanje
GND	GND	Ground	Ozemljitev
pin 40	DATA	Data	Podatki

Poglavje 4

Razvoj programske opreme

V tem poglavju je predstavljeno programiranje varnostnega sistema v razvojnem okolju Arduino IDE, programska koda za posamezne komponente varnostnega sistema, delovanje posameznih odsekov programske kode kot celote in pa uporabljene knjižnice za določene komponente. Komunikacija med varnostnim sistemom in uporabnikom preko GSM modula SIM900 je vidna na Sliki 4.1.



Slika 4.1: Komunikacija varnostnega sistema z uporabnikom

4.1 Uporabljene knjižnice

V programski kodi so bile z namenom hitrejšega programiranja v projekt uvožene in uporabljene knjižnice s sledečimi zaglavnimi (header) datotekami:

- `Keypad.h` [19], za delo s številčno tipkovnico.
- `DHT.h` [18], za branje podatkov o temperaturi in vlagi iz modula DHT22.

- `String.h` [21], za delo z nizi in ne samo s posameznimi znaki.
- `Wire.h` [22], za delo z I^{2}C LCD zaslonom.
- `LiquidCrystal_I2C.h` [20], prav tako za delo z I^{2}C LCD zaslonom.

4.2 IR senzor HC-SR505

IR senzor HC-SR505 je v našem varnostnem sistemu zadolžen za zaznavanje premikanja. V programski kodi zaznava gibanje le, ko je sistem aktiviran, saj ko je sistem deaktiviran, po tem ni potrebe.

Za delovanje IR senzorja so potrebne naslednje spremenljivke v programski kodi (vidne na Sliki 4.2):

- `pinPIR1`, `pinPIR2` in `pinPIR3`, s katerimi določimo izbrane pine na mikrokrmilniku Arduino. V našem varnostnem sistemu so to pin22, pin24 in pin26.
- `calibrationTime`, s katerim določimo čas kalibracije IR senzorja.
- `lowIn` je spremenljivka, ki ima shranjen čas, ko IR senzor naredi prehod iz visokega v nizek pulz.
- `pause` je spremenljivka, ki določa čas v milisekundah, ko senzor ne zazna več premikanja. Po tem času lahko predvidevamo, da se je premikanje nehalo.
- `lockLow` je spremenljivka za omejitev izvajanja dela programske kode, ko senzor prvič zazna gibanje.
- `takeLowTime` je spremenljivka, ki prav tako kot `lockLow` omeji izvajanje dela programske kode, le da tokrat tistega dela, ki shrani čas, ko senzor ne zaznava več gibanja.

Kot je razvidno s Slike 4.3, je potrebno v funkciji `setup()` nastaviti pine izbrane preko `pinPIR` spremenljivk kot `INPUT`, saj bodo IR senzorji

```
int ledPin = 13;
int pinPIR1 = 22;
int pinPIR2 = 24;
int pinPIR3 = 26;
int calibrationTime = 30;

long unsigned int lowIn;

long unsigned int pause = 50;

boolean lockLow = true;
boolean takeLowTime;
```

Slika 4.2: Spremenljivke potrebne za delovanje IR senzorja HC-SR505

pošiljalji podatke mikrokrmilniku Arduino. Nato pa se izvede še kalibracija IR senzorjev, ki se v mojem varnostnem sistemu izvaja 30 sekund.

```
void setup() {
  pinMode(ledPin, OUTPUT);
  pinMode(pinPIR1, INPUT);
  pinMode(pinPIR2, INPUT);
  pinMode(pinPIR3, INPUT);

  Serial.begin(9600);

  Serial.print("calibrating sensor ");
  for(int i = 0; i < calibrationTime; i++){
    Serial.print(".");
    delay(1000);
    }
  Serial.println(" done");
  Serial.println("SENSOR ACTIVE");
  delay(50);
}
```

Slika 4.3: Inicializacija IR senzorja HC-SR505

Za testiranje delovanja sem potreboval serijsko komunikacijo, saj sem tako lahko v Arduino IDE spremljal, ali senzor zaznava gibanje. Slika 4.4 prikazuje delovanje enega senzorja, v tem primeru IR senzorja prikljopljenega na pin22 mikrokrmilnika Arduino.

```
void loop() {
  if(digitalRead(pinPIR1) == HIGH){
       digitalWrite(ledPin, HIGH);
       if(lockLow){
         lockLow = false;
         Serial.println("---");
         Serial.print("motion detected at ");
         Serial.print(millis()/1000);
         Serial.println(" sec");
         delay(50);
         }
         takeLowTime = true;
       }

      if(digitalRead(pinPIR1) == LOW){
        digitalWrite(ledPin, LOW);
        if(takeLowTime){
         lowIn = millis();
         takeLowTime = false;
         }

        if(!lockLow && millis() - lowIn > pause){
            lockLow = true;
            Serial.print("motion ended at ");
            Serial.print((millis() - pause)/1000);
            Serial.println(" sec");
            delay(50);
            }
        }
}
```

Slika 4.4: Delovanje IR senzorja HC-SR505

4.3 I²C LCD zaslon in številska tipkovnica

I²C LCD zaslon in številsko tipkovnico bom strnil v skupno poglavje, saj je bila njuna programska koda programirana istočasno, saj sem tako najlažje poskrbel za delovanje obeh, prav tako pa sem lahko sprogramiral logiko za aktivacijo in deaktivacijo varnostnega sistema.

Za delovanje I²C LCD zaslona in pa številske tipkovnice sem sprva potreboval zaglavja knjižnic `Keypad.h`, `Wire.h` in `LiquidCrystal_I2C.h`. Prav tako sem definiral stalno dolžino niza varnostne kode v spremenljivko

`Pass_Len`. Ker je bilo potrebno upoštevati še znak `NULL`, je za varnostni niz dolžine 4 potrebna definicija niza dolžine 5.

Preostale spremenljivke, potrebne za delovanje I^2C LCD zaslona in številske tipkovnice, vidne na Sliki 4.5, so sledeče:

- `numRows` in `numCols` sta spremenljivki, s katerimi definiramo število vrstic in stolpcev na naši številski tipkovnici. Ker sem v svojem projektu izbral tipkovnico velikosti 4*3, potrebujem definicijo 4 vrstic in 3 stolpcev.

- `keymap` je dvodimenzionalna tabela znakov, v kateri definiramo vrednosti, ki jih pritisk določenega gumba na številski tipkovnici pošlje mikrokrmilniku Arduino. To pomeni, da bi lahko bila tipkovnica tudi črkovna, vendar, ker so znaki na naši tipkovnici številke, sem se odločil za definicijo identične tipkovnice v programski kodi.

- `rowPins` in `colPins` sta tabeli, ki imata definirane pine na mikrokrmilniku Arduino, kamor se bo priključila številska tipkovnica. Ker imamo 4 vrstice, je velikost tabele `rowPins` enaka številu vrstic, tj. 4, velikost tabele `colPins` pa je zaradi 3 stolpcev na tipkovnici velika 3.

- `myKeypad` je objekt tipkovnice, kar nam omogoča knjižnica Keypad. Potrebno je definirati našo tipkovnico, kar storimo z `makeKeymap(keymap)`, nato sledi definicija pinov za vrstice in pinov za stolpce. Na koncu pa je potrebno še definirati število vrstic in stolpcev.

- Za delovanje LCD zaslona je v tem trenutku potrebna le uporaba knjižnice LiquidCrystal_I2C. Z njo definiramo spremenljivko `lcd`, v kateri uporabimo unikatni naslov I^2C vodila, kar je pri našem zaslonu `0x3F` (glej Sliko 3.3). Sledi definicija pinov, ki jih uporablja vodilo I^2C, in pa kako deluje osvetlitev zaslona.

- `inputPass` in `masterPass` tabeli sta si zelo podobni, saj sta obe enako veliki. Vendar je razlika ta, da je v tabeli `masterPass` niz, ki

sistem deaktivira in aktivira, v tabeli `inputPass` pa je vnešeno geslo, ki ga vnesemo z uporabo številske tipkovnice.

- `inputCount` je spremenljivka, ki jo potrebujemo za štetje vnešenih znakov, saj lahko le tako vemo, kdaj je vnešeno enako znakov kot je definirana dolžina varnostnega niza.

- `lcdDisplay` je tabela znakov, ki jih prikazujemo na LCD zaslonu, v našem primeru so to *, saj varnostnega niza ne želimo prikazovati javno, vendar želimo uporabniku vseeno prikazati število vnešenih znakov.

- Spremenljivka `unlocked` pove, ali je varnostni sistem deaktiviran ali aktiviran. Zaradi testiranja našega sistema bo ob zagonu sistem deaktiviran, vendar želimo zaradi varnostnih razlogov, da se v primeru ponovnega zagona varnostnega sistema le-ta zažene kot aktiviran. S tem preprečimo poizkus vlomilcev, da bi sistem deaktivirali, če mu začasno odstranijo vir elektrike.

- `command` je spremenljivka, ki ima shranjen ukaz * ali #, ki ju potrebujemo za aktivacijo in deaktivacijo varnostnega sistema.

- Spremenljivka `workingDisplay` je uporabljana zaradi lažje spremembe programske kode. Tako lahko prilagodimo znak, s katerim zamenjamo vnešeno številko varnostnega niza s katerimkoli znakom želimo. Zaradi standardnega prikaza sem sam izbral znak *.

- Spremenljivka `poskusi` ima shranjeno število poskusov, ki so nam še ostali, da sistem uspešno deaktiviramo. V primeru prevelikega števila napačnih vnosov varnostnega niza sistem sproži alarm.

Kot je razvidno iz Slike 4.6, je potrebno v funkciji `setup()` nastaviti velikost LCD zaslona, ki ga uporabljamo. Ker je naš zaslon velik 16*2 znakov, kličemo funkcijo `begin(16,2)`, kar pomeni, da imamo 16 znakov v vrstici in 2 vrstici. Nato s klicem `setCursor()` postavimo kazalec na mesto 0,0, kar je levi zgornji kot LCD zaslona. Za prehod na začetek druge vrstice bi

```
#include <Keypad.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

#define Pass_Len 5

const byte numRows = 4;
const byte numCols = 3;
int ledPin = 13;

char keymap[numRows][numCols] =
{
  {'1', '2', '3'},
  {'4','5', '6'},
  {'7','8', '9'},
  {'*','0', '#'}
};

byte rowPins[numRows] = {2, 3, 4, 5};
byte colPins[numCols] = {6, 7, 8};

Keypad myKeypad = Keypad(makeKeymap(keymap), rowPins, colPins, numRows, numCols);
LiquidCrystal_I2C lcd(0x3F, 2, 1, 0, 4, 5, 6, 7, 3, POSITIVE);


char inputPass[Pass_Len];
char masterPass[Pass_Len] = "1231";
int inputCount = 0;

char lcdDisplay[Pass_Len];

bool unlocked = false;
char command = '0';
char workingDisplay = '*';
int poskusi = 4;
```

Slika 4.5: Spremenljivke, potrebne za delovanje LCD zaslona in številske tipkovnice

uporabili koordinati 0,1. Same številske tipkovnice ni potrebno inicializirati v funkciji `setup()`.

Kot je razvidno s Slike 4.7, zajamemo vrednost pritisnjenega gumba na številski tipkovnici v spremenljivko `keypressed` s klicem funkcije `getKey()`. Funkcija je del objekta `myKeypad`, ki predstavlja našo številsko tipkovnico. Če smo pritisnili gumb, preidemo v sekvenco preverjanja, kateri

```
void setup() {

  pinMode(ledPin, OUTPUT);

  Serial.begin(9600);

  lcd.begin(16,2);
  lcd.setCursor(0,0);
  lcd.print("Initializing...");
  delay(1000);
  lcd.clear();
  lcd.setCursor(0,0);
  if(!unlocked)
    lcd.print("Zavarovano!");
  else if(unlocked)
    lcd.print("Nezavarovano!");
}
```

Slika 4.6: Inicializacija LCD zaslona

gumb je bil pritisnjen, od česar je odvisno nadaljnje izvajanje programske kode. V našem varnostnem sistemu je znak # ukaz za začetek deaktivacije varnostenga sistema, znak * pa ukaz za začetek aktivacije varnostnega sistema.

```
char keypressed = myKeypad.getKey();
if(keypressed != NO_KEY) {
  if(keypressed == '#') {
```

Slika 4.7: Zajem znaka s številko tipkovnico in prehoda v deaktivacijo varnostnega sistema

Na Sliki 4.8 je prikazana funkcija `clearData()`, katera se izvede, ko je vnešen niz, enak dolžini nastavljenega varnostnega niza. Če si niza nista enaka, se izpiše preostalo število poskusov, nakar sledi izpraznitev niza, v katerega smo vnesli varnostni niz.

Vnos varnostnega niza se omogoči šele s pravilno izbiro ukaza izmed # in *, saj vnos varnostnega niza v primeru deaktiviranega sistema z ukazom # za ponovno deaktivacijo nima smisla, prav tako v primeru že aktiviranega varnostnega sistema. V primeru vnosa napačnega niza zmanjšamo število preostalih poskusov in uporabnika pozovemo, da poskusi ponovno.

```
void clearData() {
  if(strcmp(inputPass, masterPass) != 0) {
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Poskusite znova");
    lcd.setCursor(0,1);
    lcd.print("Se ");
    lcd.print(poskusi);
    lcd.print(" poskusov");
  }
  while(inputCount != -1) {
    inputPass[inputCount] = 0;
    lcdDisplay[inputCount] = 0;
    inputCount--;
    Serial.println(inputCount);
  }
  inputCount = 0;
  command = '0';
  Serial.println("It was cleared");
  return;
}
```

Slika 4.8: Izbris vnešenega varnostnega niza

4.4 Pasivni piskač

Pasivni piskač je potreben za simulacijo alarmnega zvoka v našem varnostnem sistemu. Potreboval sem le 3 spremenljivke, `pinBuzzer1`, `pinBuzzer2` in `pinBuzzer3`, kar so pini na mikrokrmilniku Arduino, kamor bodo piskači povezani (vidno na Sliki 4.9).

```
int pinBuzzer1 = 23;
int pinBuzzer2 = 25;
int pinBuzzer3 = 27;
```

Slika 4.9: Spremenljivke, potrebne za delovanje pasivnih piskačev

V funkciji `setup()` je bilo potrebno le definirati prej izbrane pine na mikrokrmilniku Arduino kot izhodne (Slika 4.10), saj pasivnemu piskaču posredujemo podatke in jih od njega ne prejemamo.

Ker se pasivne piskače aktivira na večih mestih programske kode var-

```
pinMode(pinBuzzer1, OUTPUT);
pinMode(pinBuzzer2, OUTPUT);
pinMode(pinBuzzer3, OUTPUT);
```

Slika 4.10: Inicializacija pasivnih piskačev

nostnega sistema, sem za njihovo aktivacijo in deaktivacijo spisal ločeno funkcijo, poimenovano `piskanje()`, prikazano na Sliki 4.11. Funkciji je potrebno podati vrednost `dT`, ki v funkciji določa, s kakšnim zamikom bomo piskače aktivirali in deaktivirali. Ko piskaču pošljemo logično 1, kar mu predstavlja visok pulz, ga aktiviramo, nato imamo `dT` milisekund zamika, nakar mu pošljemo logično 0 oziroma nizek pulz, da ga deaktiviramo. Temu sledi še en zamik `dT` milisekund. S temi zamiki preprečimo, da bi piskač prejel preveč sprememb pulza in bi se pokvaril.

```
void piskanje(int dT) {
  digitalWrite(pinBuzzer1, 1);
  digitalWrite(pinBuzzer2, 1);
  digitalWrite(pinBuzzer3, 1);
  delay(dT);
  digitalWrite(pinBuzzer1, 0);
  digitalWrite(pinBuzzer2, 0);
  digitalWrite(pinBuzzer3, 0);
  delay(dT);
}
```

Slika 4.11: Aktivacija pasivnih piskačev

4.5 Senzor vlage in temperature DHT22

Senzor vlage in temperature DHT22 sem v svoj varnostni sistem vključil zaradi dodatnega informiranja uporabnika glede stanja v določeni sobi v njegovi hiši ali njegovem stanovanju. S tem lahko uporabnik izboljša kvaliteto življenja v tistem prostoru, saj vidi, kakšen je odstotek vlažnosti v prostoru in kakšna je temperatura prostora. Kot je vidno na Sliki 4.12, je za delo-

vanje senzorja potrebna knjižnica DHT in njena zaglavna datoteka `DHT.h`. Potrebno je bilo definirati pin, na katerega bomo priključili senzor DHT22, kar smo storili z `DHTPIN`. Senzor za pravilno delovanje potrebuje še definiran tip senzorja, saj imamo senzor DHT11 in DHT22, katerega sem uporabil v svojem varnostnem sistemu. To se definira v spremenljivki `DHTTYPE`. Za uporabo senzorja moramo narediti objekt tipa DHT, kateri je poimenovan `dht`, vključuje pa podatek o pinu, na katerega je priključen, in pa tip senzorja.

Uporabljene so bile še naslednje spremenljivke:

- Spremenljivka `hum` tipa float za shranjevanje podatka o vlažnosti v prostoru.
- Spremenljivka `temp` prav tako tipa float za shranjevanje podatka o temperaturi v prostoru.

```
#include <DHT.h>;

#define DHTPIN 40
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
float hum;
float temp;
```

Slika 4.12: Spremenljivke, potrebne za delovanje senzorja DHT22

Za inicializacijo senzorja DHT22 moramo v funkciji `setup()` samo klicati funkcijo `begin()` na objektu `dht`, ki predstavlja naš senzor vlage in temperature, kar je razvidno s Slike 4.13.

```
void setup() {
  Serial.begin(9600);
  dht.begin();
}
```

Slika 4.13: Inicializacija senzorja vlage in temperature DHT22

V funkciji `loop()` najprej v spremenljivki `hum` in `temp` zajamemo podatke o vlagi in temperaturi. To storimo s klicem funkcije `readHumidity()` in `readTemperature()` iz knjižnice DHT. Nato te podatke lahko izpišemo na serijski izhod ali pa na LCD zaslon. Ko podatke zajamemo, je potrebno narediti časovni zamik 2 sekund, saj lahko DHT22 senzor zajema podatke na vsaki 2 sekundi in hitrejše zajemanje nima smisla. Odsek programske kode za uporabo senzorja je prikazan na Sliki 4.14.

```
void loop(){
    hum = dht.readHumidity();
    temp= dht.readTemperature();

    Serial.print("Vlaga: ");
    Serial.print(hum);
    Serial.print(" %, Temp: ");
    Serial.print(temp);
    Serial.println(" Celsius");
    delay(2000);
}
```

Slika 4.14: Delovanje senzorja vlage in temperature DHT22

4.6 IR senzor plamena

IR senzor plamena sem se odločil v varnostni sistem vključiti zaradi splošne varnosti hiše uporabnika v primeru, da izbruhne požar, ko lastnika ni doma. V primeru večje požarne nevarnosti v okolici hiše je uporabnik lahko bolj miren, saj bi senzor zaznal ogenj in bi uporabnika na to opozoril. Za delovanje IR senzorja plamena sta potrebni le 2 spremenljivki, prikazani na Sliki 4.15 in sicer `sensorMin` in `sensorMax`. `SensorMin` pove, kolikšna je najmanjša možna zajeta vrednost, `sensorMax` pa kolikšna je najvišja možna zajeta vrednost.

Nato v funkciji `loop()`, vidni na Sliki 4.16, zajamemo analogno vrednost, ki jo zazna naš senzor plamena, ter jo shranimo v spremenljivko `sensor-`

```
const int sensorMin = 0;
const int sensorMax = 1024;
```

Slika 4.15: Spremenljivke, potrebne za delovanje IR senzorja plamena

`Reading`. Nato z uporabo funkcije `map()` vrednost spremenljivke `sensorReading` in spremenljivk `sensorMin` ter `sensorMax` prilagodimo na vrednost med 0 in 3. S tem ugotovimo, ali je plamen oddaljen, ali je blizu, ali pa ni senzor zaznal še nobenega plamena.

```
void loop() {

  int sensorReading = analogRead(A0);

  int range = map(sensorReading, sensorMin, sensorMax, 0, 3);

  switch (range) {
  case 0:
    Serial.println("Bliznji ogenj");
    break;
  case 1:
    Serial.println("Oddaljen ogenj");
    break;
  case 2:
    Serial.println("Ni ognja");
    break;
  }
  delay(10);
}
```

Slika 4.16: Delovanje IR senzorja plamena

4.7 Senzor nagiba SW-520D

Senzor nagiba SW-520D igra v našem varnostnem sistemu pomembno vlogo, saj zaznava nagib kljuke na vratih. V primeru, da se odločijo vlomiti preko vrat, bo senzor sprožil 60 sekundno odštevanje, preden se sproži alarm. To je potrebno, saj moramo ob vstopu skozi vrata deaktivirati varnostni sistem.

Potrebne so naslednje spremenljivke, vidne na Sliki 4.17:

- Spremenljivka `inPin`, ki določa pin na mikrokrmilniku Arduino, na katerega bo priključen senzor nagiba SW-520D.
- Spremenljivka `reading`, v kateri je shranjeno trenutno stanje, ki ga je mikrokrmilnik Arduino pridobil od senzorja nagiba.
- Spremenljivka `previous`, v kateri je shranjeno prejšnje stanje, ki ga je mikrokrmilnik Arduino pridobil. S tem lahko primerjamo, ali se je zgodila sprememba v nagibu.
- Spremenljivka `time`, v katero shranjujemo čas zadnje zaznane spremembe.
- Spremenljivka `debounce`, s katero preprečimo kratke dogodke, kot je v primeru potresa ali stiska senzorja.

```
int inPin = 30;
int outPin = 13;

int reading;
int previous = LOW;

long time = 0;
long debounce = 50;
```

Slika 4.17: Spremenljivke, potrebne za delovanje senzorja nagiba SW-520D

V funkciji `setup()`, vidni na Sliki 4.18, je potrebno `inPin` nastaviti na vhodni pin, saj pošilja senzor nagiba podatke mikrokrmilniku Arduino. Pomembno je še, da na `inPin` zapišemo stanje `HIGH`, saj s tem vključimo vgrajeni dvižni (pull-up) upor.

V funkciji `loop()` je potrebno pri senzorju nagiba upoštevati več pogojev, ki so prikazani na Sliki 4.19, in katere bom sedaj predstavil. Najprej je potrebno preveriti, ali se je zgodila sprememba stanja senzorja, in če se je ta

```
void setup()
{
  Serial.begin(9600);
  pinMode(inPin, INPUT);
  digitalWrite(inPin, HIGH);
  pinMode(outPin, OUTPUT);
}
```

Slika 4.18: Inicializacija senzorja nagiba SW-520D

zgodila, v spremenljiko `time` shranimo čas spremembe stanja senzorja nagiba. Če je ta sprememba aktivna dlje časa kot ga določa spremenljivka `debounce`, zapišemo v spremenljivko `switchstate` trenutno stanje senzorja nagiba, kar uporabimo za aktivacijo alarma, če je senzor nagnjen. Nazadnje pa še zapišemo trenutno stanje senzorja nagiba v spremenljivko `previous`, saj s tem pomnimo zadnje prebrano stanje senzorja.

```
void loop()
{
  int switchstate;

  reading = digitalRead(inPin);
  Serial.println(reading);

  if (reading != previous) {
    time = millis();
  }

  if ((millis() - time) > debounce) {
     switchstate = reading;
    if (switchstate == HIGH)
      LEDstate = LOW;
    else
      LEDstate = HIGH;
  }
  digitalWrite(outPin, LEDstate);


  previous = reading;
}
```

Slika 4.19: Delovanje senzorja nagiba SW-520D

Poglavje 5

Testiranje varnostnega sistema

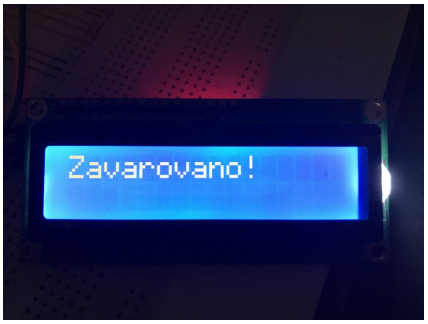
V tem poglavju bom predstavil potek testiranja varnostnega sistema kot celote pred namestitvijo na maketo hiše. Testiranje je s stališča delovanja varnostnega sistema pomembno, saj s tem izločimo možne napake v delovanju sistema. Testiranje je potekalo na dva načina in sicer preko izpisa opozoril na I^2C LCD zaslonu ter z uporabo GSM modula SIM900 za pošiljanje SMS opozoril na telefon.

5.1 I^2C LCD zaslon in številska tipkovnica

Sprva je bilo pri testiranju pomembno, da je sistem zaznaval številsko tipkovnico, saj ga tako aktiviramo in deaktiviramo. To sem lahko testiral preko serijskega vmesnika v razvojnem okolju Arduino IDE, vendar je potrebno uporabniku prikazati napredek vnosa varnostne kode preko vmesnika. Zato sem uporabil še I^2C LCD zaslon, na katerem se izpiše trenutno stanje varnostnega sistema, kar je vidno na Sliki 5.1, ki prikazuje izpis neaktivnega varnostnega sistema, ter Sliki 5.2, ki prikazuje izpis aktivnega varnostnega sistema. Ko je sistem aktiven, delujejo vsi senzorji, saj pri neaktivnem sistemu po tem ni potrebe.

Ko uporabnik želi sistem aktivirati (Slika 5.3), na številčni tipkovnici najprej pritisne gumb z znakom *, nato pa začne vnašati varnostno kodo. Zaradi

Slika 5.1: Izpis na I²C LCD zaslonu, ko je varnostni sistem deaktiviran



Slika 5.2: Izpis na I²C LCD zaslonu, ko je varnostni sistem aktiviran

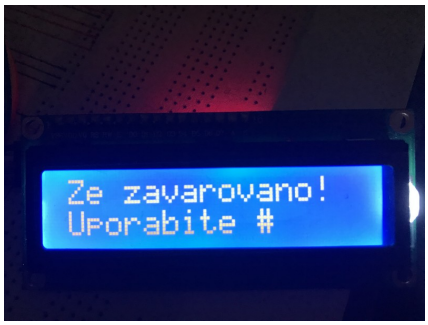
varnosti podatkov se na I²C LCD zaslonu ne prikazujejo pritisnjena števila, temveč jih zamenja znak *. Za deaktivacijo varnostnega sistema (Slika 5.4) pa uporabnik na številčni tipkovnici najprej pritisne gumb z znakom #, nato pa vnese varnostno kodo. Kot pri deaktivaciji varnostnega sistema se tudi pri aktivaciji vnešena števila zamenjajo z znakom *. Ko se sistem aktivira, pa ima uporabnik še 60 sekund časa, da zapusti stavbo, saj se šele po izteku tega časa sistem aktivira in je pripravljen sprožiti alarm. Tudi odštevanje do aktivacije sistema je vidno na I²C LCD zaslonu in je prikazano na Sliki 5.13.

Slika 5.3: Izpis na I²C LCD zaslonu med aktiviranjem varnostnega sistema

Slika 5.4: Izpis na I²C LCD zaslonu med deaktivacijo varnostnega sistema

Ker bi bilo nesmiselno varnostni sistem aktivirati, če je le-ta že aktiven,

sem v programski kodi to preprečil, za uporabnika pa pripravil izpis na I^{2}C LCD zaslon, ki mu sporoči, da je sistem že aktiven in stavba že zavarovana, če pa želi sistem deaktivirati, naj uporabi gumb #, kar je razvidno s Slike 5.5. Isti princip velja pri deaktivaciji že deaktiviranega sistema, saj tudi to ni smiselno dejanje. V tem primeru pa izpis na I^{2}C LCD zaslonu uporabniku sporoči, da je gumb za sprožitev vnosa varnostnega gesla za aktivacijo sistema * in naj uporabi le-tega, če želi varnostni sistem aktivirati, kar je razvidno s Slike 5.6.



Slika 5.5: Izpis na I^{2}C LCD zaslonu, če želimo sistem aktivirati, ko je že aktiven

Slika 5.6: Izpis na I^{2}C LCD zaslonu, če želimo sistem deaktivirati, ko je že neaktiven

Pri tem segmentu testiranja nisem naletel na nobeno težavo, saj je samo testiranje delovanja številčnice in aktivacije ter deaktivacije varnostnega sistema enostavno.

5.2 IR senzor HC-SR505

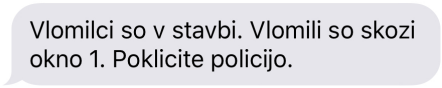
Pri testiranju delovanja IR senzorja HC-SR505 se je že v začetku pokazala težava, saj senzor pokrije široko območje, prav tako pa je pokrita razdalja kar velika. To je pomenilo, da senzorja nisem mogel testirati, ne da bi mu omejil vidno polje. To sem za potrebe testiranja storil s papirnato cevko, v katero sem dal IR senzor HC-SR505. Ko je bilo poskrbljeno za to, pa je bilo potrebno testirati, ali se senzor odziva na gibanje in pa na prenehanje

gibanja. Kot je razvidno s Slike 5.7, je senzor zaznal gibanje. Izpis na LCD zaslonu se razlikuje glede na lokacijo senzorja. V prikazanem primeru je bilo zaznano gibanje na oknu 1. S Slike 5.8 pa je razvidna uspešna realizacija konca zaznave gibanja.

Slika 5.7: Izpis na I²C LCD zaslonu, ko je zaznano gibanje

Slika 5.8: Izpis na I²C LCD zaslonu, ko ni več zaznanega gibanja

Potrebno je bilo še testiranje IR senzorja HC-SR505 v kombinaciji z GSM modulom SIM900, saj želimo uporabniku v primeru vloma skozi okno to tudi sporočiti. Na Sliki 5.9 vidimo, da je bilo pošiljanje SMS sporočila z uporabo GSM modula SIM900 uspešno, kar pomeni, da bi bil uporabnik mojega varnostnega sistema opozorjen na vlomilce.

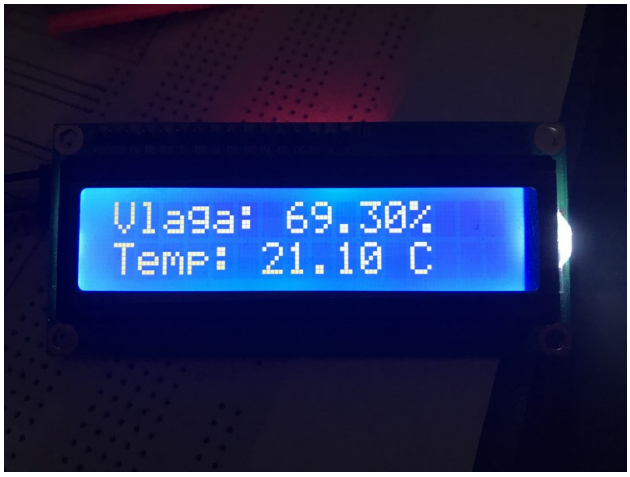


Slika 5.9: Opozorilni SMS ob vlomu skozi okno

5.3 Senzor vlage in temperature DHT22

Senzor vlage in temperature DHT22 je bilo potrebno testirati, če vsaj približno pravilno pridobiva podatke o temperaturi in vlagi v okolici, kar je bilo najlažje testirati s približevanjem vroče roke in odmikanjem le-te. Ko je bila roka približana, je izpisana temperatura narasla kot pričakovano. Kot

je razvidno s Slike 5.10, je zajemanje podatkov uspešno, saj se tudi na LCD zaslonu izpiše trenutno stanje v prostoru. Ker pa želimo izpisovati trenutno stanje na LCD zaslonu periodično in ne vsaki 2 sekundi, je bilo potrebno še testiranje časovne omejitve, kdaj želimo to izpisovati. To je se je v programski kodi preverilo vsako iteracijo funkcije `loop()` in sicer se je le preverilo, kdaj smo nazadnje izpisali podatek o temperaturi in vlagi, ter če je minilo dovolj časa (npr. 1 minuta) za ponoven izpis trenutne temperature in vlage na LCD zaslon, kar je bilo tudi uspešno izvedeno.

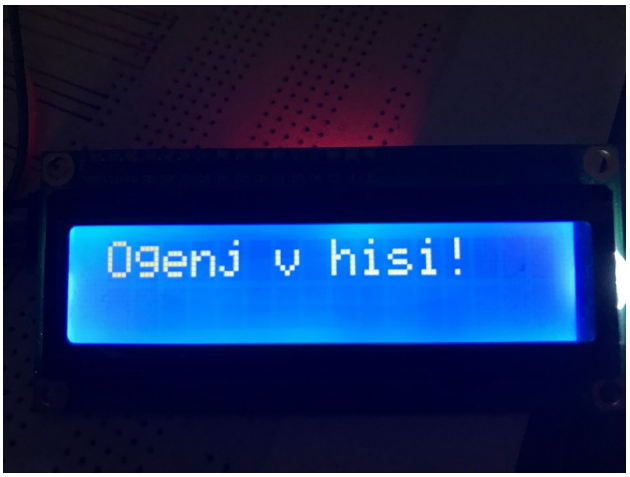


Slika 5.10: Izpis temperature in vlage v prostoru s senzorjem DHT22

5.4 IR senzor plamena

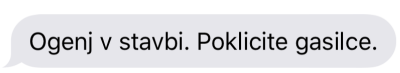
Testiranje IR senzorja plamena je prav tako kot testiranje IR senzorja HC-SR505 imelo svoje izzive. Prvi tak izziv je bilo testiranje pravilnega mapiranja vrednosti, ki jih senzor vrne mikrokrmilniku Arduino, saj sem te vrednosti razdelil na 3 dele, ki predstavljajo zaznavanje oddaljenega plamena, zaznavanje bližnjega plamena in brez zaznave plamena. To sem testiral preko serijskega vmesnika v razvojnem okolju Arduino IDE z uporabo vžigalnika. Le-tega sem premikal bližje senzorju ter ga odmikal. Ko je to delovalo, je bilo potrebno testirati, ali deluje izpis na I^2C LCD zaslon, ko senzor zazna plamen, ne glede na to, ali je oddaljen ali je blizu. Kot je razvidno s Slike 5.11,

je bilo testiranje uspešno in je delovanje senzorja pravilno.



Slika 5.11: Zaznava ognja v hiši z IR senzorjem

Sledilo je še testiranje senzorja in varnostnega sistema skupaj z GSM modulom SIM900, saj je ob zaznanem plamenu potrebno na to opozoriti uporabnika, da lahko le-ta pokliče gasilce in prepreči požar. Kot je vidno na Sliki 5.12, je mikrokrmilnik Arduino uspešno zahteval od GSM modula SIM900 pošiljanje SMS sporočila, GSM modul pa ga je uspešno poslal.



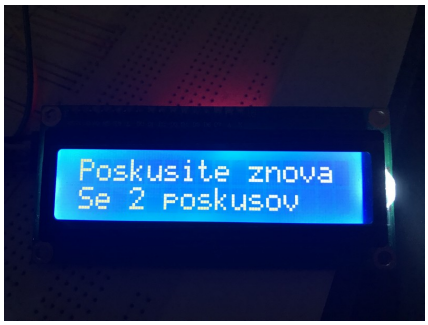
Slika 5.12: Opozorilni SMS ob zaznavi plamena

Še ena zanimiva situacija, ki se je primerila pri testiranju IR senzorja plamena, pa je delovanje senzorja na sončni svetlobi. Senzor je sprva, ko je bil testiran v notranjih prostorih, deloval kot je bilo potrebno, ko pa je bil testiran zunaj na sončni svetlobi, je senzor vseskozi zaznaval plamen. Ko sem malo raziskal situacijo, sem ugotovil, da senzor zaznava vsako svetlobo in ne samo plamena. Senzor sem zato za trenutne potrebe testiranja in izdelave varnostnega sistema zaščitil s slamico, ki je nameščena na senzor. S tem preprečim učinke sončne ali ambientne svetlobe, kar vodi v pravilno delovanje IR senzorja.

5.5 Senzor nagiba SW-520D

Zadnje testiranje je bilo testiranje senzorja nagiba SW-520D. Prav tako je testiranje imelo svoje izzive, saj je bilo potrebno ugotoviti, kakšen bo pravi začetni nagib in kateri nagib bo sprožil alarm. Zato je bila programska koda spisana tako, da je začetna pozicija senzorja vodoravna, senzor pa je obrnjen z žicami v levo smer in s kroglicami v cevki v desno. Tako se senzor aktivira ob nagibu kljuke. Ko se zgodi nagib, ki ga senzor zazna, le-ta sporoči to mikrokrmilniku Arduino, ki nato požene sekvenco odštevanja časa, v katerem lahko uporabnik deaktivira varnostni sistem. Izpis na I^2C LCD zaslonu se posodablja vsako sekundo (Slika 5.13) in uporabniku odšteva čas, po izteku katerega se bo sprožil alarm. Če uporabnik napačno vnese varnostno kodo, se mu prikaže izpis o neuspešnem vnosu in preostalem številu poskusov, prikazan na Sliki 5.14.

Slika 5.13: Sprožitev odštevalnika časa ob zaznavi nagiba kljuke s senzorjem nagiba SW-520D

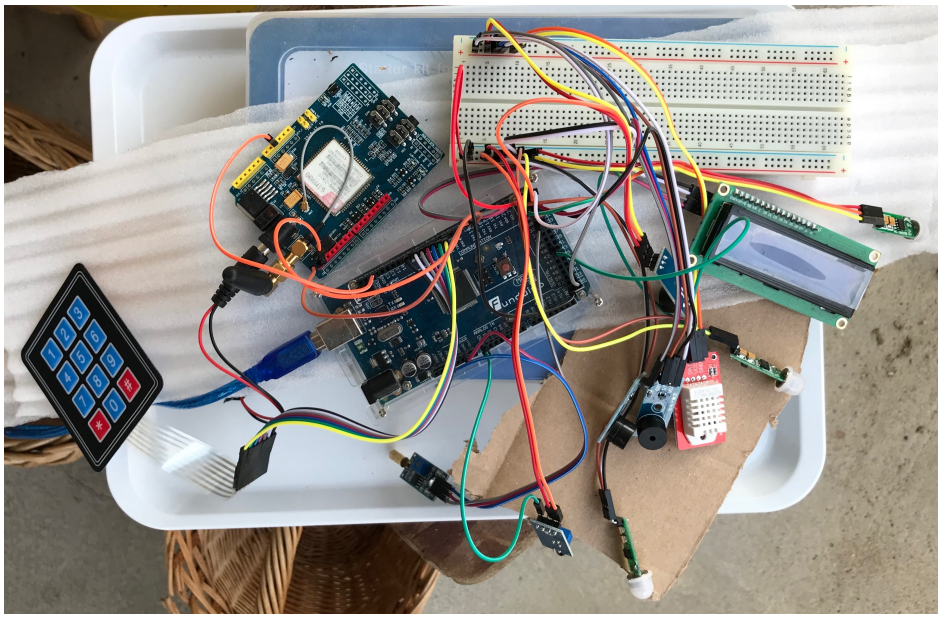


Slika 5.14: Spodletel poskus deaktivacije varnostnega sistema

5.6 Testiranje celotnega sistema

Ko so bila opravljena testiranja posameznih senzorjev, je bilo vse to potrebno še stestirati kot celoto. Prva pomembna stvar je bilo delovanje senzorjev le, ko je bil varnostni sistem aktiviran. Edini senzor, ki je deloval tudi pri

deaktiviranem varnostnem sistemu, je senzor vlage in temperature DHT22. Celoten varnostni sistem je deloval kot je potrebno, prav tako je sprožil alarm, ko se je sprožil katerikoli izmed senzorjev. Prav tako so bili skupaj s celotnim varnostnim sistemom testirani pasivni piskači za zvočni alarm, saj delujejo v svoji lastni funkciji znotraj programske kode, kar je zahtevalo le posamezen klic te funkcije, ko je katerikoli senzor sprožil alarm. Vendar pa za popolno testiranje sam prototip varnostnega sistema, ki ga vidimo na Sliki 5.15, ni bil dovolj merodajen, ne da bi bil nameščen v podobno okolje kot je stanovanje ali hiša. Zato sem se odločil narediti maketo hiše, v kateri bo varnostni sistem napeljan in ga bo možno natančneje testirati in tudi prikazati njegovo delovanje.



Slika 5.15: Prototip varnostnega sistema

Izdelava makete in napeljava varnostnega sistema bo prikazana in opisana v naslednjem poglavju.

Poglavje 6

Izgradnja makete in namestitev varnostnega sistema

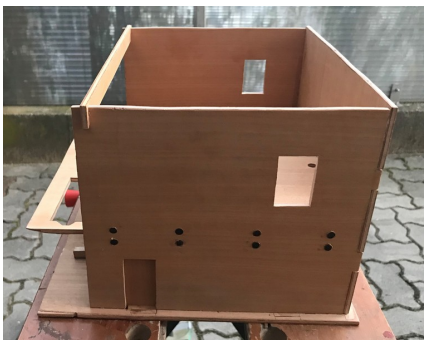
V tem poglavju bom predstavil potek izgradnje makete hiše in pa namestitve varnostnega sistema na maketo. Za izgradnjo makete sem se odločil zaradi boljše preglednosti varnostnega sistema kot celote, prav tako pa mi maketa omogoča boljšo urejenost kablov in zmanjša verjetnost medsebojnega motenja senzorjev.

6.1 Izgradnja makete hiše

Za začetek sem potreboval okvirne mere za velikost makete hiše, saj sem želel, da ima vsak senzor dovolj prostora za pravilno delovanje. Tako sem za širino in globino makete odmeril 26cm veliko spodnjo ploskev. Odločil sem se za odprt koncept hiše, saj sem postavil le 3 stene, s tem pa omogočil enostavnejšo namestitev varnostnega sistema in boljši dostop do samega mikrokrmilnika Arduino. Stene so visoke 17cm, od tega je 7cm višina spodnjega nadstropja, kjer bo mikrokrmilnik Arduino in pa vsa električna napeljava, v zgornjem nadstropju pa bodo le senzorji. Za uporabo IR senzorjev HC-SR505 za zaznavanje gibanja skozi okna pa sem v stene izžagal okna. Maketa hiše je prikazana na Sliki 6.1, kjer je razvidna postavitev nadstropij, prav tako

pa je razviden način pritrditve mikrokrmilnika Arduino s pritrditvijo lesenih palčk, ki so pritrjene v enakem razmaku kot je širina mikrokrmilnika, ter na Sliki 6.2, kjer je prikazano okno. Prav tako je na Sliki 6.1 vidna palica, ki deluje kot dodatno ojačanje trdnosti sten, saj povezuje levo in desno steno med seboj.

Slika 6.1: Pogled na maketo hiše od spredaj

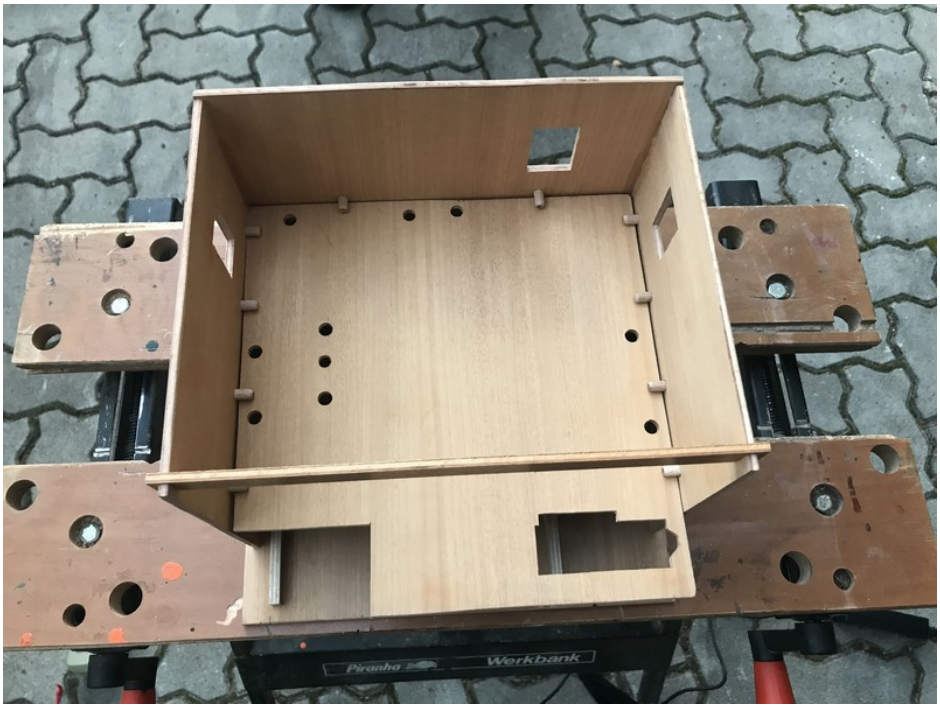


Slika 6.2: Pogled na maketo hiše s strani

Ker sem želel lažji dostop do spodnjega nadstropja, sem z uporabo moznikov (Slika 6.3) pritrdil tla v zgornjem nadtropju, istočasno pa jim omogočil, da jih lahko potegnem ven. S tem sem si omogočil lažji dostop do spodnjega nadstropja, kjer bo mikrokrmilnik Arduino.

Slika 6.3: Leseni moznik

Sledilo je izbiranje mesta senzorjev, saj sem nato moral v tla zgornjega nadstropja izvrtati luknje, skozi katere se bo pri namestitvi varnostnega sistema napeljevalo kable, potrebne za delovanje posameznega senzorja. Stanje makete z izvrtanimi luknjami za napeljavo kablov je vidno na Sliki 6.4.



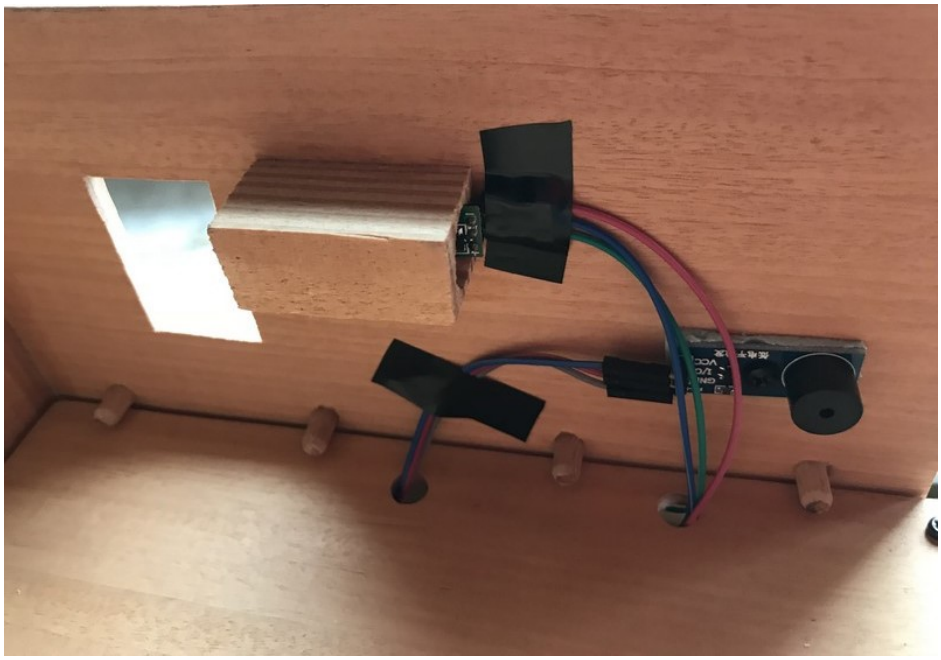
Slika 6.4: Maketa z luknjami za napeljavo kablov

6.2 Namestitev varnostnega sistema

Najprej je bilo potrebno nalepiti nosilce za IR senzorje gibanja HC-SR505. Ko se je lepilo za les strdilo, sem namestil IR senzorje ter jih pritrdil z uporabo črnega izolirnega traku. Pod IR senzorje pa sem namestil piskača za zvočni alarm, kateremu sem žice prav tako pritrdil z uporabo črnega izolirnega traku. To je vidno na Sliki 6.5.

Ker pa samega piskača ne moremo priviti neposredno na maketo hiše, saj bi s tem poškodovali senzor, sem med senzor in maketo položil ploščico stirodura (Slika 6.6) v katerega se je senzor ugreznil. To mi je omogočilo pritrditev piskača na maketo. Stirodur sem uporabil še pri namestitvi IR senzorja plamena ter senzorja nagiba SW-520D.

Kot je razvidno s Slike 6.7, sem poleg IR senzorja gibanja ter piskača namestil še senzor za temperaturo in vlago DHT22. Tega sem najprej z vijakom pritrdil na leseno ploščico, nato pa to ploščico z dvema vijakoma pritrdil na



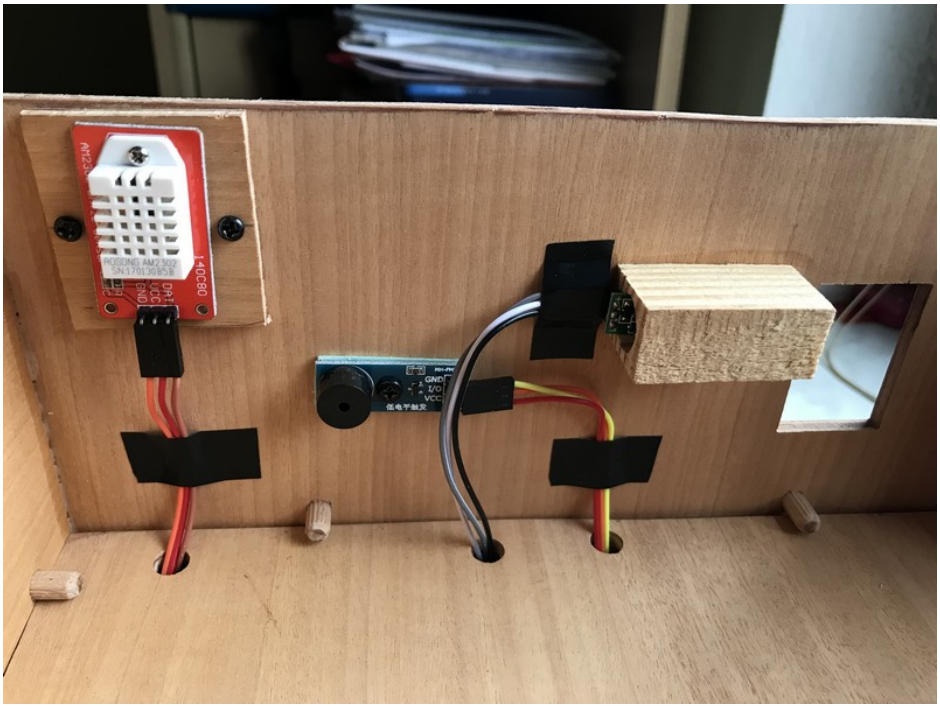
Slika 6.5: IR senzor gibanja in piskač

Slika 6.6: Stirodur

maketo hiše. S tem sem si omogočil enostavno prestavljanje senzorja. Prav tako pa sem pri vseh senzorjih uporabil črni izolirni trak za pritrditev kablov.

Sledi prikaz mesta namestitve IR senzorja plamena. Nameščen je na rob tal zgornjega nadstropja (vidno na Sliki 6.8), saj sem si tako omogočil lažje testiranje delovanja in zmanjšal verjetnost vnetja makete z ognjem v primeru demonstracije. Ker je IR senzor nameščen na odstranljiva tla nadstropja, sem žice z izolirnim trakom pritrdil na spodnji strani in s tem preprečil, da bi se žice odklopile s senzorja v primeru premikanja tal nadstropja.

Namestitve senzorja nagiba SW-520D sem se lotil z izdelavo kljuke, katero



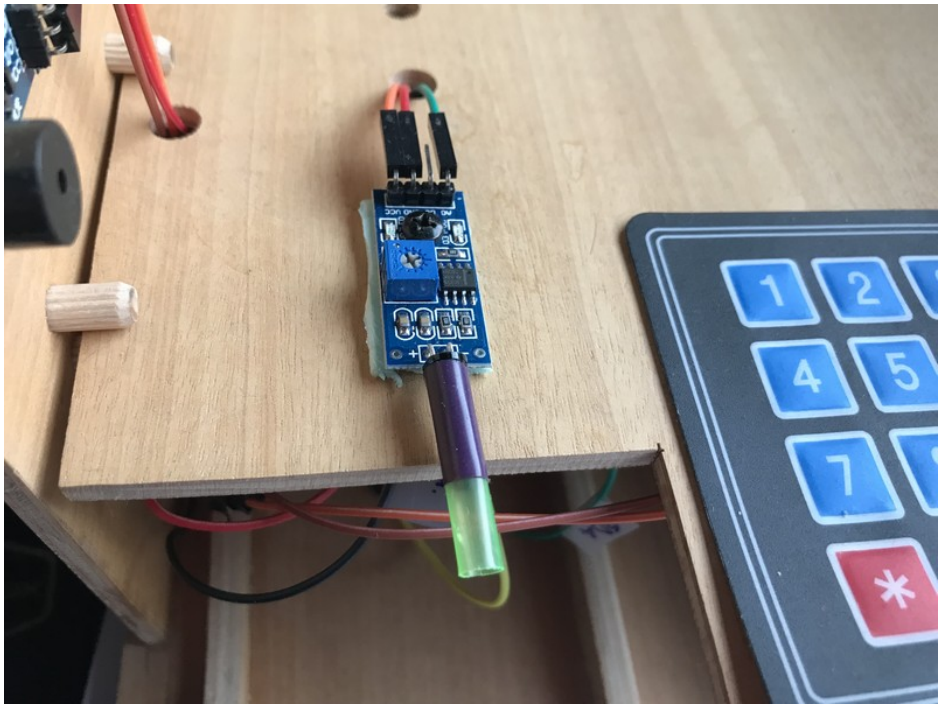
Slika 6.7: IR senzor gibanja, piskač ter senzor vlage in temperature DHT22

sem izdelal iz železne palice. To sem nato pritrdil na kovinsko os, sajmi to omogoča nagibanje kljuke. Na kljuko sem nato privil senzor nagiba SW-520D. Kljuka je prikazana na Sliki 6.9. Ta kljuka je bila nato nameščena ob stran hiše za demonstracijo delovanja senzorja nagiba SW-520D.

Potrebno je bilo napeljati in pritrditi še številsko tipkovnico in pa I^2C LCD zaslon. Tipkovnico sem na tla v zgornjem nadstropju pritrdil z dvostranskim lepilom, žice pa na spodnjo stran tal nadstropja z izolirmin trakom. I^2C LCD zaslon sem pritrdil s štirimi vijaki in maticami, saj je nameščen v zanj namensko izžagano luknjo v tleh zgornjega nadstropja makete hiše. Namestitev številske tipkovnice in I^2C LCD zaslona je vidna s Slike 6.10, kjer je vidna namestitev tudi preostalih senzorjev znotraj makete hiše.

GSM modul SIM900 sem namestil v prazen prostor na sredini zgornjega nadstropja makete hiše, saj je tako neovirana antena modula, prav tako pa ne moti delovanja drugih senzorjev.

Kot je bilo že omenjeno, je v spodnjem nadstropju makete nameščen



Slika 6.8: IR senzor plamena

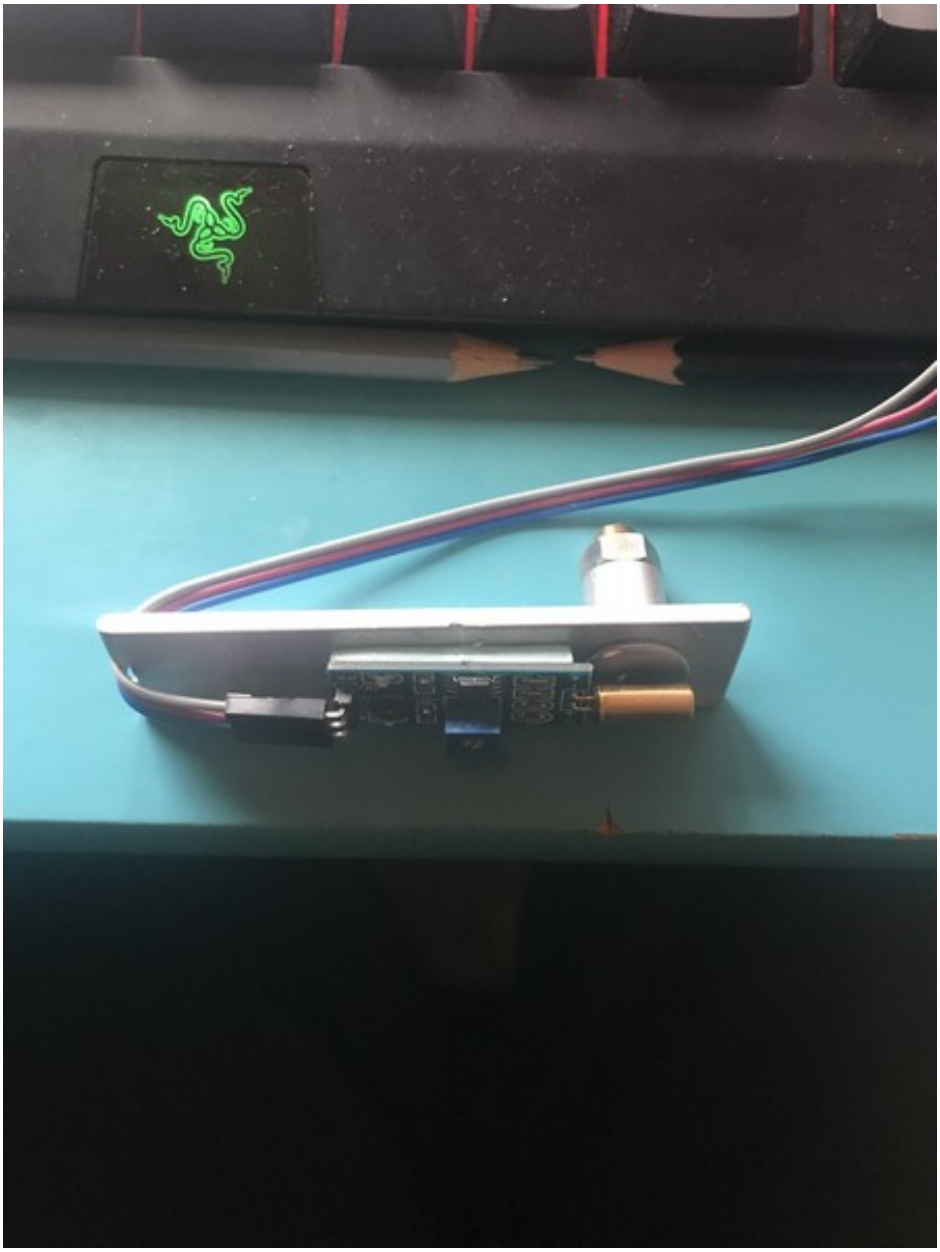
mikrokrmilnik Arduino in pa ploščica za priključitev napajalnih kablov. S tem omogočim karseda čist izgled zgornjega nadstropja, kjer so vsi senzorji in se izognem nepotrebnim kablom. Spodnje nadstropje s kabli je vidno na Sliki 6.11.

6.3 Končni izdelek

Ker je bil cilj diplomske naloge izdelati cenovno ugoden varnostni sistem, je čas za primerjavo posameznih delov mojega varnostnega sistema in pa varnostnega sistema v podobni konfiguraciji kot je moj prototip varnostnega sistema. Pri primerjavi cen sem si pomagal z informativnim izračunom podjetja MOBICOM d.o.o. [14].

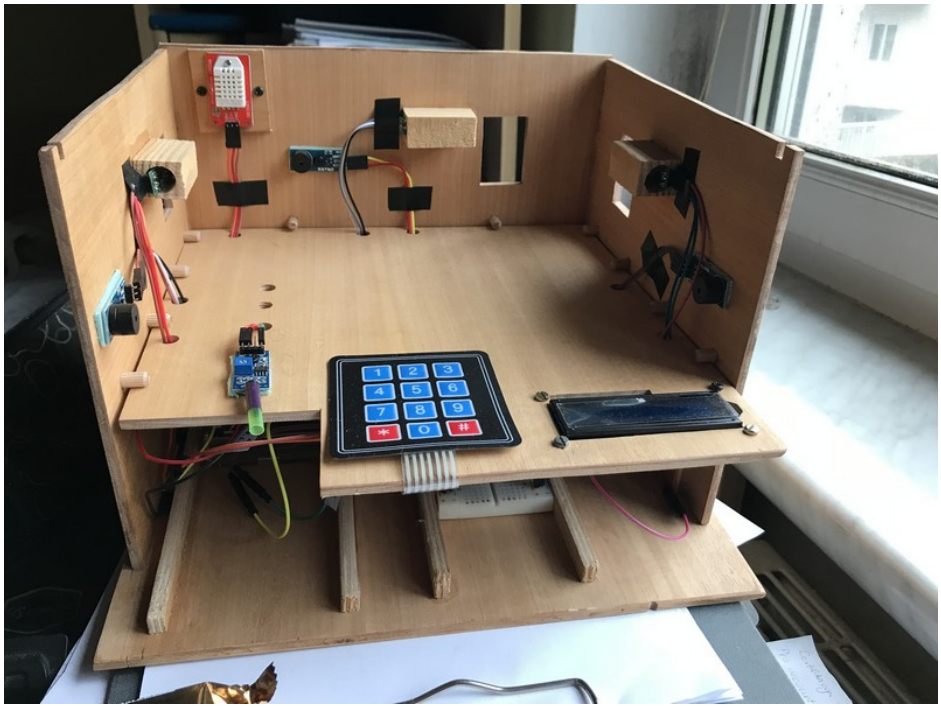
Prvi je izračun cene mojega prototipa varnostnega sistema:

- Mikrokrmilnik Funduino Mega 10,79€ [9]



Slika 6.9: Senzor nagiba SW-520D na kljuki

- 120 kosov 20cm žic za vezavo senzorjev 4,76 € [13]
- SIM900 GSM modul 19,00 € [31]
- Senzor za temperaturo in vlago DHT22 3,44 € [27]



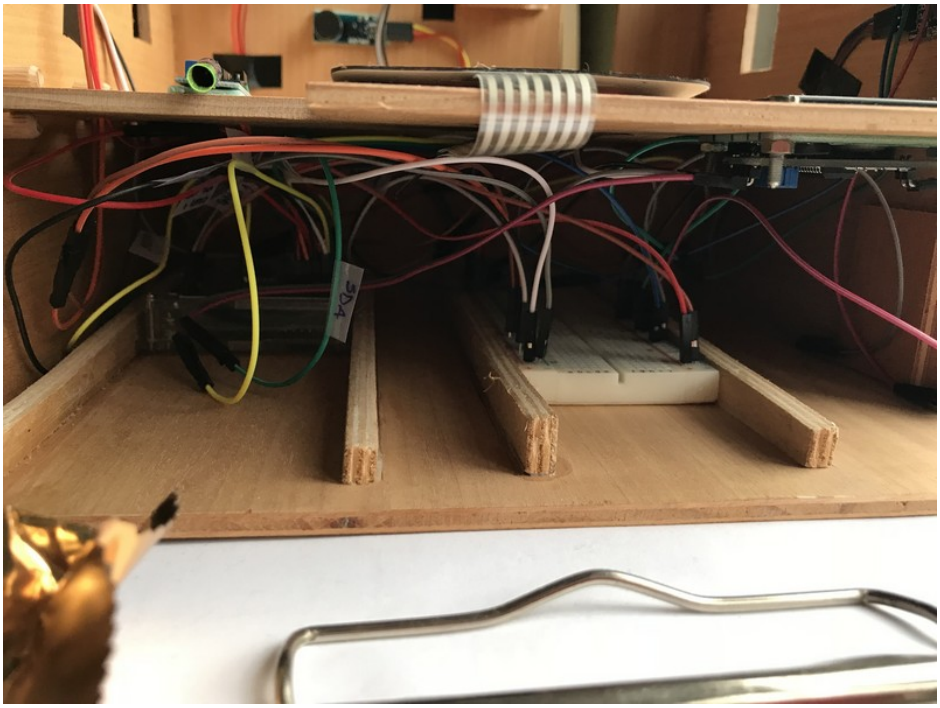
Slika 6.10: Notranjost makete hiše z vsemi senzorji

- I^2C LCD zaslon 16*2 2,10 € [10]
- 3 IR senzorji HC-SR505 1,10 €/kos [16]
- IR senzor plamena 1,00 € [17]
- Senzor nagiba SW-520D 0,50 €/kos [29]
- Številska tipkovnica 0,51 € [32]
- 9V priključek za napajanje SIM900 GSM modula 0,33 € [2]
- 9V alkalna baterija za napajanje SIM900 GSM modula 3,90 € [1]

Končna cena varnostnega sistema je: 49,63 €

Edina pomanjkljivost pa je naročanje večine komponent s Kitajske, kar povzroči daljše čakalne čase.

Sledi izračun cene varnostnega sistema pdojetja MOBICOM d.o.o.(cena je brez vštetega DDV):



Slika 6.11: Spodnje nadstropje makete hiše

- Žični protivlomni sistem PC1616+PK5501(številčnica z LCD zaslonom) 138 € + DDV
- Dodatek za centralo - Transformator 14,00 € + DDV
- Dodatek za centralo - Akumulator 7.0Ah 21,00 € + DDV
- Razširitveni modul PC5208 52,00 € + DDV
- 3 notranji senzorji LC-100PI 13,00 €/kos + DDV
- Protipožarna zaščita in zaščita pred CO2 - protipožarni senzor ADSF-D-A35BS 27,00 € + DDV
- 3 notranje sirene MOS1 14,00 €/kos + DDV
- GSM/GPRS komunikator GS3105 132,00 € + DDV

Skupna cena komponent varnostnega sistema podjetja MOBICOM d.o.o. je: 465,00 € + DDV

Vendar je potrebno k tej ceni prišteti še strošek vgranje sistema, ki v primeru že pripravljene inštalacije znaša 183,00 € + DDV, ter strošek programiranja in zagona sistema, ki znaša 150,00 € + DDV. Tako znese končna cena napeljanega in delujočega varnostnega sistema podjetja MOBICOM d.o.o. 798,00 € + DDV.

Iz te primerjave je razvidno, da današnji varnostni sistemi niso več cenovno nedostopni, vendar pa so za nekoga, ki živi na minimalni plači, cene še vedno previsoke. Tako se mi zdi, da bi moj cenovno ugodnejši varnostni sistem (49,63 €) tudi takim ljudem omogočil nekaj osnovne varnosti njihovih domov.

Poglavje 7

Sklepne ugotovitve

Namen diplomske naloge je bil pregled področja varnostnih sistemov in pa razvoj prototipa cenovno ugodnega varnostnega sistema, kjer je osnova mikrokrmilnik Arduino, kar nam je skozi diplomsko nalogo tudi uspelo. Izdelati smo želeli varnostni sistem, ki bi bil čimbolj enostaven za uporabo in ki bi zagotovil čim boljšo zaščito doma uporabnika.

Sam izdelek je nameščen na izdelano maketo hiše, na kateri tudi deluje kot je potrebno. Občasen problem povzroča le IR senzor plamena, ki se sproži ob prvem zagonu programske opreme. Ko je varnostni sistem v fazi delovanja, pa IR senzor plamena ne povzroča težav.

Sam varnostni sistem ponuja obilico možnih izboljšav, kot je dodajanje novih vrst senzorjev, shranjevanje dogodkov na spominsko kartico, dodatek kamere v varnostni sistem in snemanje dogodkov, brezžična povezljivost s pametnim telefonom in upravljanje z varnostnim sistemom preko telefona. Vendar pa zadnja izboljšava varnostnega sistema prinaša tudi svoje slabosti, saj je potem potrebno poskrbeti še za zadostno brezžično varnost, saj se lahko vsiljivec, če to zna storiti, poveže med nas in varnostni sistem, ter tako pridobi varnostno kodo za deaktivacijo sistema. Posledično s tem onemogoči varnostni sistem in se mu odpre možnost vloma v hišo.

Tema diplomske naloge je s stališča varnosti in cenovne dostopnosti trenutnih varnostnih sistemov aktualna, saj so varnostni sistemi, ki so trenutno

na trgu, občutno dražji kot pa prototip našega varnostnega sistema.

Literatura

[1] 9V alkalna baterija. Dosegljivo: https://www.mimovrste.com/navadne-baterije/varta-alkalna-baterija-9v-6lr61-high-energy. [Dostopano: 11. 12. 2017].

[2] 9V napajalni priključek. Dosegljivo: https://www.aliexpress.com/item/free-shipping-90-DC-9V-Battery-button-power-plug-for-Arduino-Mega-2560-1280-UNO-R3/32712837070.html?spm=a2g0s.9042311.0.0.Be2MXl. [Dostopano: 11. 12. 2017].

[3] Arduino. Dosegljivo: https://en.wikipedia.org/wiki/Arduino. [Dostopano: 24. 10. 2017].

[4] Arduino IDE. Dosegljivo: https://www.arduino.cc/en/Main/Software. [Dostopano: 24. 10. 2017].

[5] Arduino Mega 2560 Arduino Store ponudba. Dosegljivo: https://store.arduino.cc/arduino-mega-2560-rev3. [Dostopano: 25. 10. 2017].

[6] Arduino Mega 2560 Conrad ponudba. Dosegljivo: https://www.conrad.si/ARDUINO-MEGA-2560-BOARD-A000067.htm?websale8=conrad-slowenien&pi=191790. [Dostopano: 25. 10. 2017].

[7] Arduino primeri. Dosegljivo: http://playground.arduino.cc/. [Dostopano: 24. 10. 2017].

[8] AT ukazi za GSM modul SIM900. Dosegljivo: https://www.espruino.com/datasheets/SIM900_AT.pdf. [Dostopano: 25. 10. 2017].

[9] Funduino Mega 2560 DealExtreme ponudba. Dosegljivo: http://www.dx.com/p/improved-funduino-mega-2560-r3-module-compatible-w-official-arduino-mega-2560-r3-blue-black-256335#.WfBroWh-paQ. [Dostopano: 25. 10. 2017].

[10] I2C LCD zaslon 16*2. Dosegljivo: https://www.ebay.de/itm/New-Blue-IIC-I2C-TWI-1602-16x2-Serielles-LCD-Modul-Display-fur-Arduino-GW/272935056799?hash=item3f8c32759f:g:VY8AAOSw2GlXINYW. [Dostopano: 11. 12. 2017].

[11] I2C protokol. Dosegljivo: https://en.wikipedia.org/wiki/I%C2%B2C. [Dostopano: 25. 10. 2017].

[12] I2C Scanner. Dosegljivo: https://playground.arduino.cc/Main/I2cScanner. [Dostopano: 25. 10. 2017].

[13] Žice za vezavo senzorjev. Dosegljivo: https://www.ebay.de/itm/Dupont-Draht-Kabel-Linie-Jumper-Wire-Pi-Breadboard-F-F-M-M-F-M-fur-Arduino/191991448647?hash=item2cb3950447:m:msgOyuudMdQaMCc4pQVrwPw. [Dostopano: 11. 12. 2017].

[14] Informativna ponudba podjetja MOBICOM. Dosegljivo: http://www.mobicom.si/informativna-ponudba. [Dostopano: 11. 12. 2017].

[15] IR senzor HC-SR505. Dosegljivo: https://www.elecrow.com/wiki/index.php?title=HC-SR505_Mini_PIR_Motion_Sensor. [Dostopano: 26. 10. 2017].

[16] IR senzor HC-SR505. Dosegljivo: https://www.ebay.de/itm/HC-SR505-Mini-Body-Sensor-Schalter-Mini-Sensing-Modul-Body-Sensing-Modul/322893271691?hash=item4b2df0528b:g:i9MAAOSwFyhaCv7a. [Dostopano: 11. 12. 2017].

[17] IR senzor plamena. Dosegljivo: https://www.ebay.de/itm/IR-Infrared-Flame-Detection-Sensor-Module-for-Arduino-d/231463466396?hash=item35e44c359c:g:i4IAAOSwIyNZ3tSE. [Dostopano: 11. 12. 2017].

[18] Knjižnica DHT. Dosegljivo: https://github.com/adafruit/DHT-sensor-library. [Dostopano: 27. 10. 2017].

[19] Knjižnica Keypad. Dosegljivo: https://github.com/Chris--A/Keypad. [Dostopano: 27. 10. 2017].

[20] Knjižnica LiquidCrystal_I2C. Dosegljivo: https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library. [Dostopano: 27. 10. 2017].

[21] Knjižnica String. Dosegljivo: https://www.arduino.cc/en/Reference/StringObject. [Dostopano: 27. 10. 2017].

[22] Knjižnica Wire. Dosegljivo: https://github.com/esp8266/Arduino/tree/master/libraries/Wire. [Dostopano: 27. 10. 2017].

[23] MOBICOM. Dosegljivo: http://www.mobicom.si/protivlomni-sistem. [Dostopano 23. 5. 2017].

[24] Primerjava DHT11 in DHT22. Dosegljivo: https://learn.adafruit.com/dht/overview. [Dostopano: 26. 10. 2017].

[25] Primerjava vseh mikrokontrolerjev Arduino. Dosegljivo: https://www.arduino.cc/en/Products/Compare. [Dostopano: 24. 10. 2017].

[26] Programski jezik za programiranje Arduino mikrokrmilnika. Dosegljivo: https://www.arduino.cc/en/Reference/HomePage. [Dostopano: 24. 10. 2017].

[27] Senzor DHT22. Dosegljivo: http://www.ebay.de/itm/AM2302-Digital-Feuchtigkeit-Temperatur-DHT22-Board-Modul-Sensor-

fuer-Arduin-D1T3/262957137460?_trkparms=aid%3D555019%26algo%3DPL.BANDIT%26ao%3D1%26asc%3D20161121134654%26meid%3D9e0c265f45c24783b094c0137d48f296%26pid%3D100506%26rk%3D1%26rkt%3D1%26&_trksid=p2045573.c100506.m3226. [Dostopano: 11. 12. 2017].

[28] Senzor nagiba SW-520D. Dosegljivo: http://funduino.de/DL/SW-520D.pdf. [Dostopano: 26. 10. 2017].

[29] Senzor nagiba SW-520D. Dosegljivo: https://www.ebay.de/itm/2PCS-SW-520D-3-3V5V-Angle-Sensor-Module-Ball-Schalter-Tilt-Sensor-Module-Arduino/152685283698?hash=item238cc08972:g:jPYAAOSwkklZm5fy. [Dostopano: 11. 12. 2017].

[30] Senzor plamena. Dosegljivo: http://www.instructables.com/id/Arduino-Modules-Flame-Sensor/. [Dostopano: 26. 10. 2017].

[31] SIM900 GSM modul. Dosegljivo: https://www.ebay.de/itm/SIM900-GSM-GPRS-Modul-module-Shield-Platte-IComSat-Kit-kompatibel-Arduino/141925505298?hash=item210b6b5912:g:Ob8AAOSwk5FUsyJd. [Dostopano: 11. 12. 2017].

[32] Številska tipkovnica 4*3. Dosegljivo: https://www.aliexpress.com/item/4x3-Matrix-Array-12-Key-Membrane-Switch-Keypad-Keyboard-3-4-Control-Panel-Microprocessor-Keyboard-for/1718997674.html?spm=a2g0s.9042311.0.0.Be2MXl. [Dostopano: 11. 12. 2017].

[33] Luka Colarič. Digitalizacija in avtomatizacija vrta. Diplomska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2016.

[34] Dejan Kljajić. Varnostni vidik pri implementaciji pametne hiše. Diplomska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2016.

[35] Gorazd Štamcar. Razvoj sistema za nadzor doma. Diplomska naloga, Fakulteta za računalništvo in informatiko, Univerza v Ljubljani, 2016.