Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Elis Purkov

Hišni alarmni sistem

diplomsko delo

visokošolski strokovni študijski program prve stopnje računalništvo in informatika

Mentor: prof. dr. Patricio Bulić

Ljubljana, 2014

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Načrtujte hišni alarmni sistem. Sistem načrtujte na vgrajenem sistemu STM32F407 z mikrokrmilnikom ARM Cortex-M4. Na vgrajenem sistemu uporabiti operacijski sistem za delo v realnem času FreeRTOS. Uporabite naslednje naprave: LCD zaslon na dotik, čitalec RFID za radiofrekvenčno identifikacijo, infrardeči senzor za zaznavanje gibanja, stikala Reed in senzor plinov. Sistemu dodajte modula Ethernet in GSM. Prvi naj služi povezavi vgrajenega sistema z osebnim računalnikom, na katerem teče nadzorni program. Drugi naj služi oddaljenemu upravljanju alarmnega sistema.

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Elis Purkov, z vpisno številko **63100137**, sem avtor diplomskega dela z naslovom:

Hišni alarmni sistem

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Patricia Bulića,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- soglašam z javno objavo elektronske oblike diplomskega dela na svetovnem spletu preko univerzitetnega spletnega arhiva.

V Ljubljani, dne 16. oktober 2014 Podpis avtorja:

Rad bi se zahvalil prof. dr. Patriciu Buliću za mentorstvo in pomoč pri izdelavi diplomske naloge.
Zahvaljujem se družini, predvsem staršem, za podporo in razumevanje v času študija.

Kazalo

Seznam uporabljenih kratic

Kratica	Angleško	Slovensko
RFID	Radio Frequency Identification	Radiofrekvenčna identifikacija
PIN	Personal Identification Number	Osebna identifikacijska številka
LCD	Liquid Crystal Display	Zaslon s tekočimi kristali
AES	Advanced Encryption Standard	
DMA	Direct Memory Access	Neposredni dostop do pomnilnika
V	Volt	Volt
GND	Ground	Ozemljitev
NRZ	Non-Return to Zero	
FSK	Frequency Shift Keying	
PSK	Phase Shift Keying	
SMS	Short Message Service	Sistem kratkih sporočil
GSM	Global System for Mobile	Globalni sistem mobilnih komunikacij
MHz	Megahertz	Megaherc
ARM	Advanced RISC Machines	
A/D	Analog to Digital converter	Analogno-digitalni pretvornik
D/A	Digital to Analog converter	Digitalno-analogni pretvornik
I/O	Input/Output	Vhod/Izhod
SWD	Serial Wire Debug	
JTAG	Join Test Action Group	
I2C	Inter-Integrated circut	
SPI	Serial Peripheral Interface	Serijski vmesnik za komunikacijo
UART	Universal Asynchronous Receiver/Transmitter	
CAN	Controller Area Network	
SDIO	Secure Digital Input Output	
CRC	Cyclic Redundancy Check	Ciklično preverjanje redundance
LED	Light-Emitting Diode	Svetleča dioda
TFT	Thin-Film Transistor	
GDDRAM	Graphic Display Data RAM	
CR	Carriage return	
AT	ATtention	
UDP	User Datagram Protocol	
TCP	Transmission Control Protocol	
APB	Advanced Peripheral Bus	
AHB	Advanced High-performance Bus	

AXI	Advanced eXtensible Interface	
ACE	AXI Coherency Extensions	
AMBA	Advanced Microcontroller Bus Architecture	
EXTI	External Interrupt	
GPIO	General-Purpose Input/Output	Splošnonamenski vhod/izhod
NVIC	Nested Vectored Interrupt Controller	
ARM	Advanced RISC Machines	
USB	Universal Serial Bus	Univerzalno serijsko vodilo
FIFO	First In First Out	Prvi noter, prvi ven
XOR	Exclusive OR	Ekskluzivni OR
FSMC	Flexible Static Memory Controller	
PET	Polyethylene Terephthalate	Polietilen tereftalat

Povzetek

V današnjem času vse pogostejših tatvin se pojavlja tudi potreba po učinkovitejšem varovanju objektov, kar lahko zagotovimo z uporabo alarmnega sistema.

V okviru diplomskega dela smo izdelali alarmni sistem, ki prek različnih senzorjev spremlja dogajanje v varovanih prostorih objekta. Alarmni sistem uporablja tehnologijo RFID in dvostopenjsko avtentikacijo uporabnika, poleg tega omogoča interakcijo prek LCD zaslona in pošiljanje SMS sporočila v primeru sprožitve alarma. Vključuje tudi ethernet modul, prek katerega je povezan z nadzornim programom.

V diplomskem delu smo predstavili tehnologijo RFID in delovanje rezistivnih zaslonov na dotik. Temu sledi predstavitev delovanja posameznih komponent alarmnega sistema. V programskem delu smo opisali postopek nastavitve posameznih komponent, postopek kalibracije zaslona na dotik ter delovanje nadzornega programa in operacijskega sistema FreeRTOS.

Ključne besede: alarmni sistem, alarm, senzorji, nadzorni program, LCD zaslon, RFID, FreeRTOS, kalibracija zaslona.

Abstract

Living in a time where thievery is becoming rampant means an increase in the need to protect facilities. This may be achieved via an alarm system.

For the thesis paper, we have constructed an alarm system which uses various sensors to monitor events in the facility areas under guard. The alarm system uses RFID technology and a two-stage user identification method as well as enabling communication via an LCD display and SMS notification if the alarm gets triggered. It also includes an Ethernet module which connects it to the control program.

In the thesis paper, the RFID technology and the operation of resistive touch screens are presented. This is followed by a presentation of how individual alarm system components work. In the program part, the configuration procedure of individual components is presented, as well as the calibration of the touch screen, the operation of the control program and the FreeRTOS operating system.

Keywords: alarm system, alarm, sensors, monitoring software, LCD screen, RFID, FreeRTOS, screen calibration.

Poglavje 1

Uvod

Živimo v času, ko so tatvine vedno bolj pogoste. Zato čedalje več ljudi posega po različnih zunanjih in notranjih zaščitah za svoje objekte. Najpogostejše so notranje zaščite, s katerimi preprečimo gibanje po notranjih prostorih v času, ko v objektu ni nikogar. Med notranje zaščite spadajo protivlomna vrata, protivlomna okna, notranji video nadzor, alarmni sistemi itd. Večina ljudi se odloči za nakup protivlomnih vrat, saj ta navadno predstavljajo najcenejšo rešitev, ki pa ni nujno tudi dovolj učinkovita. Nekatera podjetja in posamezniki v kombinaciji s protivlomnimi vrati uporabljajo alarmne sisteme, ki pa so lahko zelo dragi. Ena izmed rešitev za običajnega uporabnika bi bila poleg uporabe protivlomnih vrat tudi uporaba alarmnega sistema, ki je hkrati učinkovit in cenovno ugoden.

Alarmni sistem je naprava, sestavljena iz nadzorne enote in različnih senzorjev, ki med delovanjem naprave zaznavajo neobičajne dogodke.

Nadzorna enota je sestavljena iz strojne in programske opreme, njena naloga pa je spremljanje vseh senzorjev, ki so priključeni nanjo. Če pride do neobičajnega dogodka, mora nadzorna enota izvesti primerno zaporedje akcij.

Poznamo različne alarmne sisteme, ki so namenjeni različnim situacijam. Poznamo na primer alarmne sisteme, namenjene izključno nadziranju dogodkov, kot so izlitje vode ali puščanje različnih plinov, pa tudi sisteme, ki

nadzirajo dogodke, kot so gibanje, lom okenskega stekla, puščanje kuhinjskega plina in podobno.

Cilj diplomskega dela je izdelati alarmni sistem, ki je zmožen varovati objekt pred vdorom. Alarmni sistem mora omogočati poljubno število uporabnikov, ki ga upravljajo s pomočjo RFID (ang. *Radio-frequency identification*) kartice in PIN (ang. *Personal identification number*) številke. Omogočena morata biti tudi dodajanje in upravljanje obstoječih uporabnikov sistema. Alarmni sistem mora objekt nadzirati s pomočjo senzorjev gibanja, senzorjev loma okenskega stekla, senzorjev dima in magnetnih stikal za zaznavanje vdora skozi vhodna vrata in okna. Omogočati mora tudi pošiljanje SMS sporočila v primeru vdora v objekt. Alarmni sistem mora biti povezan z računalnikom, ki prikazuje stanje sistema.

V diplomskem delu si bomo najprej ogledali delovanje LCD (ang. *Liquid-crystal display*) zaslonov na dotik in podrobneje opisali delovanje rezistivnih LCD zaslonov. Posvetili se bomo tudi delovanju tehnologije RFID in področju njene uporabe.

V drugem delu si bomo ogledali, kako je alarmni sistem sestavljen in izdelan. Predstavili bomo delovanje strojne opreme, pri čemer bomo opisali posamezne dele sistema. Sledil bo programski del, v katerem bo predstavljena nastavitev posameznih komponent alarmnega sistema. Ogledali si bomo tudi kalibracijo in branje koordinat z LCD zaslona, delovanje omrežnega modula, delovanje kriptirnega algoritma AES (ang. *Advanced Encryption Standard*), delovanje operacijskega sistema FreeRTOS in uporabo DMA (ang. *Direct memory access*) krmilnika. Za konec si bomo ogledali tudi delovanje sistema, ki ga alarmni sistem uporablja za nadzor in dvostopenjsko avtentikacijo uporabnikov.

Poglavje 2

LCD zasloni na dotik

Zaslon na dotik (ang. *Touchscreen*) je naprava s površino, namenjeno prikazovanju podatkov, zaznava pa tudi dotik ostrega predmeta (npr. pisala) ali uporabnikovega prsta. Za razliko od drugih vhodnih enot, kot je na primer miška, zaslon na dotik uporabniku omogoča neposredno interakcijo z grafičnimi elementi [2].

V današnjem času skoraj ni naprave, ki ne bi vsebovala zaslona na dotik. Uporabljajo jih predvsem uporabniki pametnih mobilnih telefonov, igralnih konzol in tabličnih računalnikov, najdemo pa jih tudi v industrijskem okolju in na javnih mestih (bankomati, informacijske točke) [3].

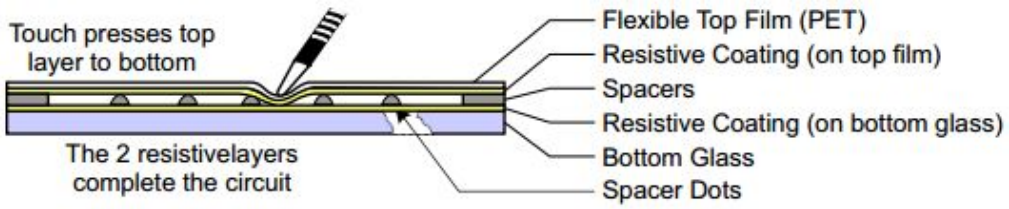
V praksi poznamo več vrst zaslonov na dotik, ki se razlikujejo glede na način delovanja [3]:

- kapacitivni zasloni,
- rezistivni zasloni,
- zasloni z infrardečo mrežo,
- zasloni s projicirano kapacitivnostjo,
- zasloni s površinskimi zvočnimi valovi.

Med najbolj razširjene vrste zaslonov spadajo kapacitivni in rezistivni zasloni, sledijo jim zasloni z infrardečo mrežo in zasloni s površinskimi zvočnimi valovi [4].

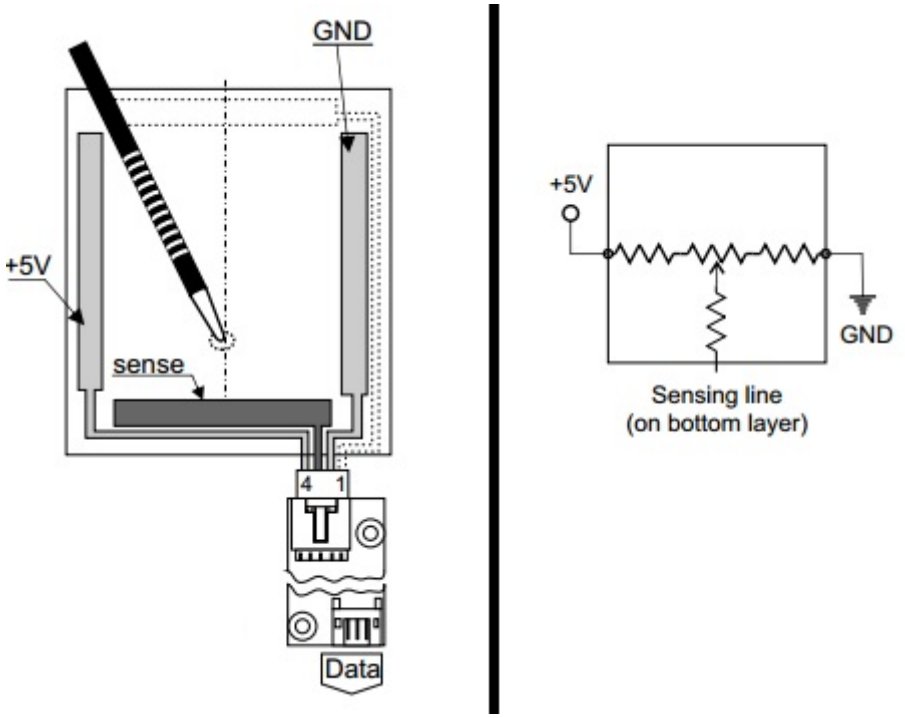
2.1 Rezistivni zasloni na dotik

Rezistivni zaslon na dotik je sestavljen iz dveh plasti. Na vrhu (Slika 2.1) je prozorna plastična PET (ang. *polyethylene terephthalate*) folija, na dnu pa steklena podlaga, ki ločuje LCD zaslon od folije, občutljive na dotik. Plastična folija in steklena podlaga sta premazani s prevodno oziroma rezistivno snovjo iz indij-kositrovega oksida (ang. *Indium tin oxide*) in ločeni s posebnimi distančniki, poleg tega vsaka vsebuje tudi dve elektrodi, namenjeni branju koordinat X in Y [5].



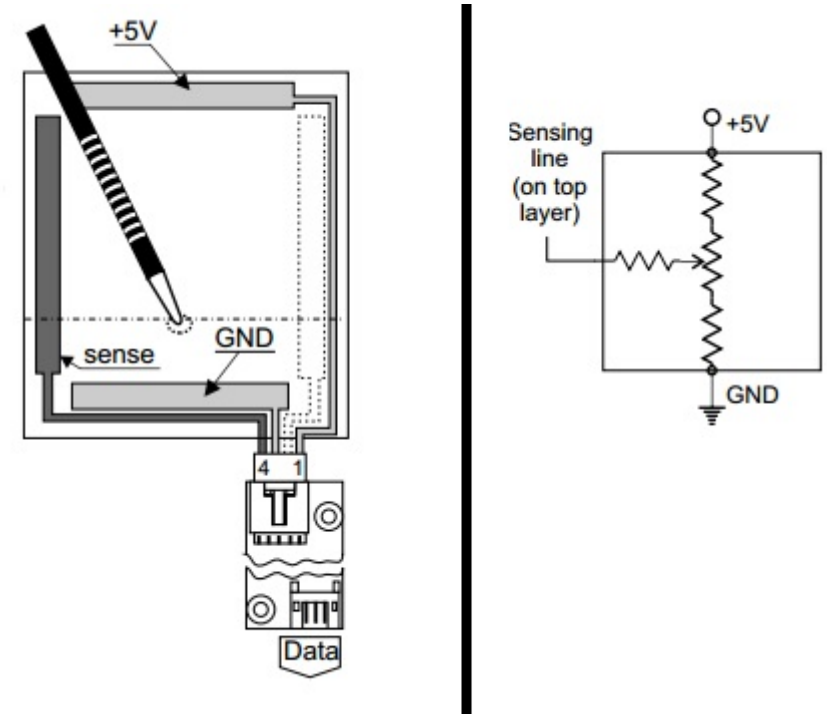
Slika 2.1: Sestava rezistivnega zaslona, vir: [5]

Ko uporabnik pritisne na zaslon, se rezistivni plasti stakneta in ustvarita stik. Krmilnik nato prebere koordinati X in Y na točki, kjer se rezistivni plasti stikata. Če želi krmilnik prebrati koordinato X, mora najprej levo elektrodo vrhnje rezistivne plasti priklopiti na +5 V (Slika 2.2), desno elektrodo pa na GND (ang. *Ground*). Na elektrodi spodnje rezistivne plasti pa prebere višino napetosti in jo s pomočjo analogno-digitalnega pretvornika spremeni v digitalno obliko [5].



Slika 2.2: Branje koordinate X, vir: [5]

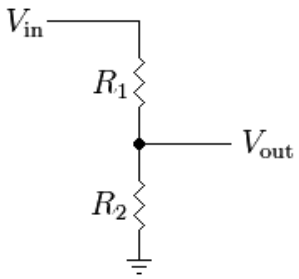
Branje koordinate Y poteka enako kot branje koordinate X, samo vloge elektrod so zamenjane. Višino napetosti krmilnik prebere z elektrode na zgornji plasti, elektrodi na spodnji plasti pa sta priklopljeni na +5 V in GND (Slika 2.3) [5].



Slika 2.3: Branje koordinate Y, vir: [5]

Sliki 2.2 in 2.3 na desni strani prikazujeta tudi skico vezja, ki se ustvari, ko pritisnemo na zaslon. Takšno vrsto vezja imenujemo napetostni delilnik. Ta je navadno sestavljen iz dveh ali več zaporedno vezanih uporov R1, R2, ..., Rn. Najosnovnejši napetostni delilnik je sestavljen iz dveh uporov R1 in R2 ter izhodnega pina Vout (Slika 2.4). Izhodna napetost Vout je odvisna od razmerja uporov R1 in R2 in višine vhodne napetosti Vin [6].

Recimo, da želimo izmeriti koordinati X in Y na točki, kjer je uporabnik pritisnil na zaslon. Predpostavimo, da zaslon z resolucijo 320 x 240 pikslov napajamo z napetostjo 5 V, rezistivni plasti pa se obnašata kot potenciometra z maksimalno upornostjo 1000 omov. Recimo, da je uporabnik pritisnil na sredino zaslona, mi pa želimo najprej izračunati koordinato X. Ker je uporabnik pritisnil na sredino zaslona, je to enako, kot da bi potenciometer nastavil na sredino, kar pomeni, da je R1 = 500 omov in R2 = 500 omov.



Slika 2.4: Napetostni delilnik, vir: [6]

Napetost na izhodnem pinu Voutx je enaka:

$$Voutx = \frac{R2x}{R1x + R2x} * Vin = \frac{500\Omega}{500\Omega + 500\Omega} * 5V = 2.5V \tag{2.1}$$

Iz tega sledi, da je koordinata X enaka:

$$X = \frac{Voutx}{Vin} * 320 = \frac{2.5V}{5V} * 320 = 160 \tag{2.2}$$

Zdaj izračunamo še napetost na izhodnem pinu Vouty. R1 = 500 omov in R2 = 500 omov, ker je uporabnik pritisnil na sredino zaslona:

$$Vouty = \frac{R2y}{R1y + R2y} * Vin = \frac{500\Omega}{500\Omega + 500\Omega} * 5V = 2.5V \tag{2.3}$$

Iz tega sledi, da je koordinata Y enaka:

$$Y = \frac{Vouty}{Vin} * 240 = \frac{2.5V}{5V} * 240 = 120 \tag{2.4}$$

Poglavje 3

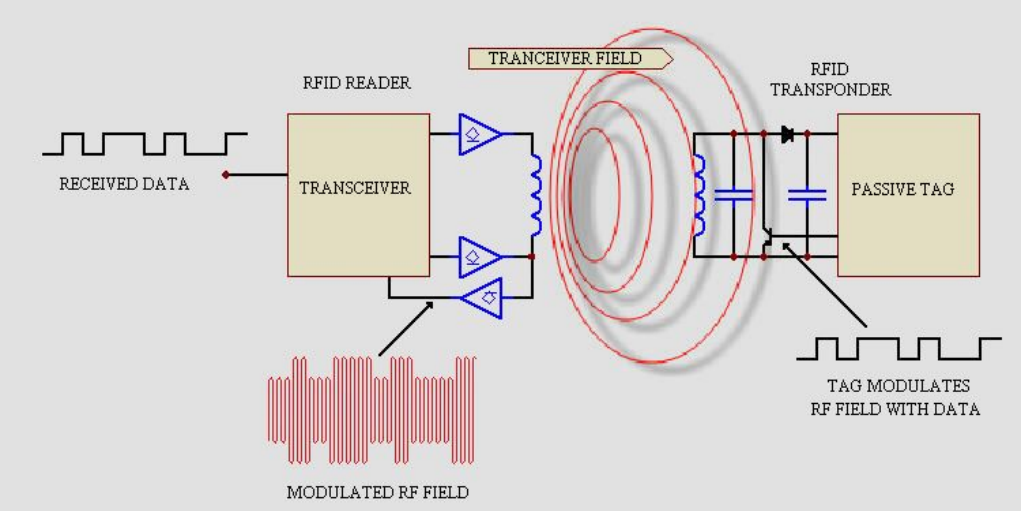
Radiofrekvenčna identifikacija

Radiofrekvenčna identifikacija (ang. *Radio Frequency Identification* ali RFID) je tehnologija, ki omogoča prenos podatkov med čitalcem in elektronsko značko. Elektronska značka je sestavljena iz antene, ki omogoča sprejemanje in oddajanje signalov, in iz integriranega vezja, ki obdeluje podatke ter izvaja modulacijo in demodulacijo signalov. Čitalec sprejema signale, ki jih oddajajo elektronske značke, kar omogoča identifikacijo predmetov, na katere je elektronska značka pritrjena [7].

3.1 Sestava in delovanje RFID sistema

RFID sistem sestavljajo RFID značke, RFID čitalniki in RFID antene (Slika 3.1). Kadar se RFID značka nahaja v območju dosega RFID čitalnika, se s pomočjo anten in radijskih valov podatki iz RFID značke prenesejo v RFID čitalnik, ki nato prebrane podatke posreduje centralnemu sistemu v obdelavo [8].

V praksi poznamo aktivne in pasivne RFID značke. Aktivne značke imajo svoje lastno napajanje in daljši doseg delovanja, medtem ko pasivne značke za delovanje uporabljajo energijo, ki se ustvari s pomočjo indukcije, zato je njihov doseg krajši [8].



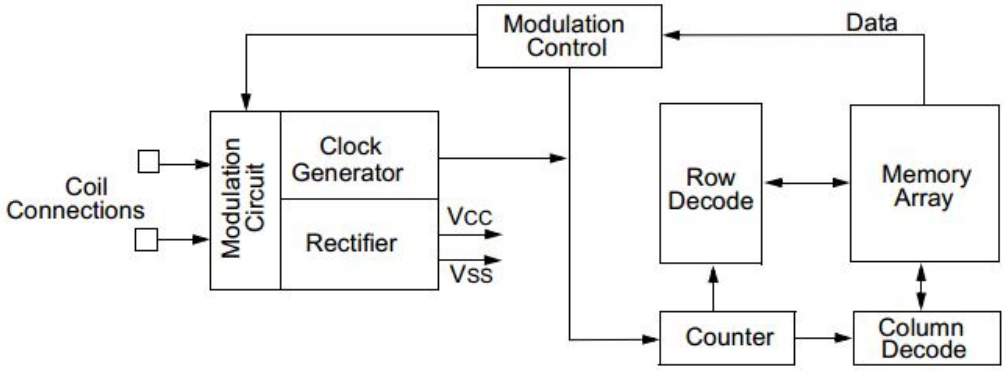
Slika 3.1: Komponente RFID sistema, vir: [9]

Radijske frekvence so pri RFID sistemih različne. V praksi uporabljamo frekvence med 100 KHz in vse do nekaj GHz. Ko izbiramo RFID komponente za svoj sistem, moramo biti pazljivi, da vse izbrane komponente delujejo na isti frekvenci (recimo 125 KHz), sicer sistem ne bo deloval [8].

3.2 Branje pasivne RFID značke

Branje RFID značke poteka tako, da čitalec oddaja neprekinjen sinusni radijski signal (nosilni signal), hkrati pa čaka na pojavitev moduliranega signala. Ko pride značka v doseg radijskega signala in dobi s pomočjo indukcije dovolj energije za pravilno delovanje, začne generirati urni signal, ki ima frekvenco za faktor nižjo od nosilnega radijskega signala. S pomočjo generiranega urnega signala značka pomika bite iz pomnilnika v modulacijsko enoto, ki vsebuje tranzistor, povezan z anteno. S pomočjo tranzistorja značka odvaja tok na anteni, s tem pa ustvari nihanja amplitude nosilnega signala glede na izbrani način modulacije. Čitalec zazna nenadne spremembe v amplitudi nosilnega signala in podatke interpretira glede na izbrani način kodiranja in modulacije [10].

V praksi se uporabljajo kodiranja NRZ (ang. *Non-Return to zero*), Differential Biphase in Biphase_L. Za modulacijo signala se uporabljata metodi FSK (ang. *Frequency Shift Keying*) in PSK (ang. *Phase Shift Keying*) [10].

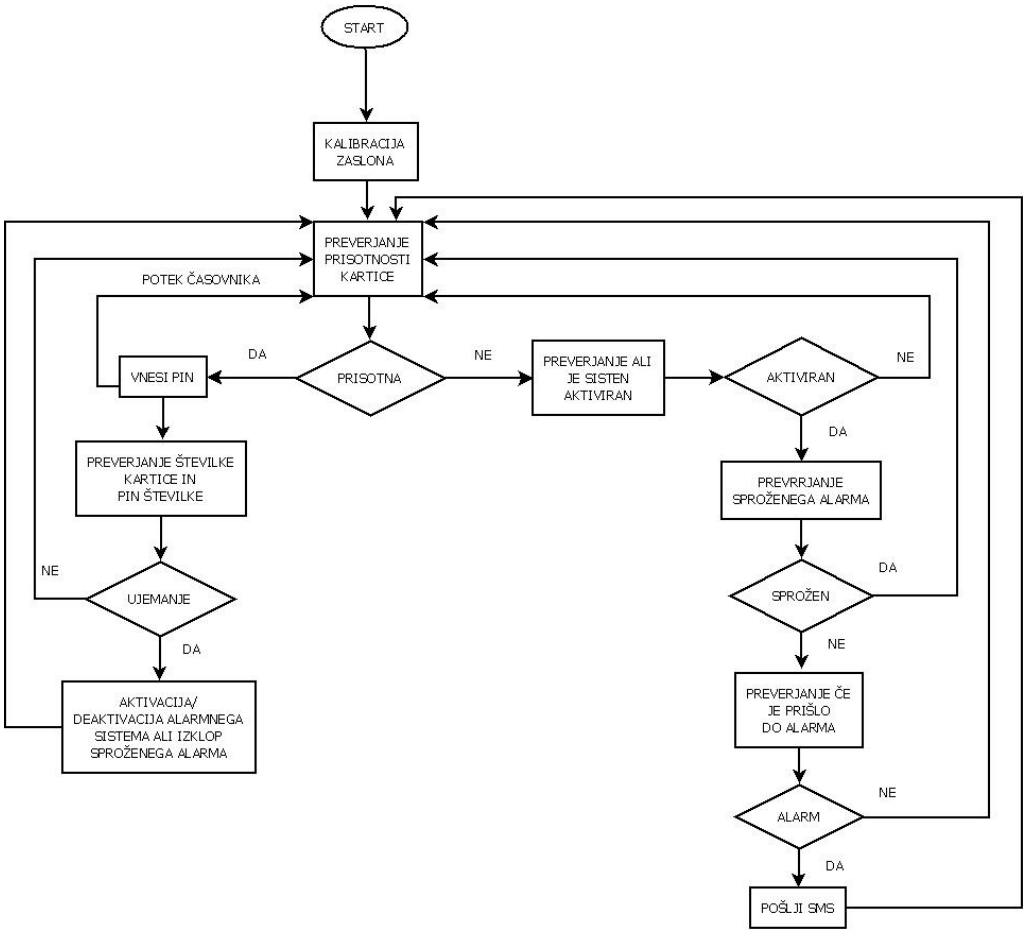


Slika 3.2: Shema vezja pasivne RFID značke, vir: [10]

Poglavje 4

Alarmni sistem

Ko uporabnik alarmni sistem priklopi na napajanje, mora najprej kalibrirati LCD zaslon. Kalibracija zaslona je potrebna zato, ker koordinate folije, občutljive na dotik, niso poravnane s koordinatami LCD zaslona. Alarmni sistem je ob vklopu v neaktiviranem stanju, kar pomeni, da se alarm ne more sprožiti, saj sistem ne spremlja stanja senzorjev. Oseba se v tem času lahko prosto giblje po varovanih prostorih objekta. Uporabnik lahko alarmni sistem aktivira s pomočjo RFID kartice in PIN številke. Ko je alarmni sistem v aktivnem stanju, se spremlja stanje vseh senzorjev, priklopljenih na sistem. Uporabnik lahko takrat, ko je alarmni sistem aktiviran, tega tudi deaktivira. Za deaktivacijo alarmnega sistema mora uporabiti RFID kartico in svojo PIN številko. Če je alarmni sistem aktiviran, senzorji pa zaznajo vdor, ima uporabnik na voljo nekaj sekund za deaktivacijo alarmnega sistema, preden se sproži alarm. Vsi dogodki, povezani z alarmnim sistemom, se posredujejo centralnemu nadzornemu sistemu. V primeru sprožitve alarma uporabniki varovanega objekta dobijo tudi SMS sporočilo, ki ga pošlje alarmni sistem. Slika 4.1 prikazuje delovanje alarmnega sistema z vidika uporabnika.



Slika 4.1: Shema delovanja vgrajenega sistema z vidika uporabnika

4.1 Strojna oprema

Alarmni sistem je sestavljen iz več komponent. Glavna komponenta sistema je razvojna plošča STM32F4-Discovery, na katero smo priklopili RFID čitalec, ethernet modul, reed stikali, senzorja gibanja, senzor plinov, senzorja loma stekla, LCD zaslon in GSM (ang. *Global System for Mobile*) modul. V nadaljevanju bodo predstavljene posamezne komponente, ki sestavljajo vgrajeni sistem (Slika 4.2).

Slika 4.2: Komponente alarmnega sistema

4.1.1 Razvojna plošča STM43F4-Discovery

Pri izdelavi alarmnega sistema smo uporabili razvojno ploščo STM32F4-Discovery podjetja STMicroelectronics. Razvojna plošča vsebuje mikrokrmilnik STM32F407VGT6, ki temelji na 32-bitnem jedru ARM (ang. *Advanced RISC Machines*) Cortex-M4, deluje pa z maksimalno frekvenco 168 MHz [11].

Značilnosti mikrokrmilnika STM32F407VG [11]:

- 1 MB bliskovnega pomnilnika,
- 192 KB RAM pomnilnika,
- 3 12-bitni A/D (ang. *Analog to Digital converter*) pretvorniki,

- 2 12-bitna D/A (ang. *Digital to Analog converter*) pretvornika,
- DMA krmilnik,
- 12 16-bitnih časovnikov in dva 32-bitna časovnika,
- 140 I/O (ang. *Input/Output*) vrat s podporo prekinitvam,
- enoti SWD (ang. *Serial Wire Debug*) in JTAG (ang. *Join Test Action Group*) za razhroščevanje,
- 3 I2C (ang. *Inter-Integrated Circut*) vmesniki,
- 4 UART (ang. *Universal Asynchronous Receiver/Transmitter*) vmesniki,
- 3 SPI (ang. *Serial Peripheral Interface*) vmesniki,
- 2 CAN (ang. *Controller Area Network*) vmesnika,
- SDIO (ang. *Secure Digital Input Output*) vmesnik,
- CRC (ang. *Cyclic Redundancy Check*) enota,
- generator naključnih števil,
- paralelni vmesnik za LCD,
- paralelni vmesnik za kamero.

Razvojna plošča STM32F4-Discovery ima naslednje značilnosti [12]:

- STM32F407VGT6 mikrokrmilnik,
- napajanje preko USB (ang. *Universal Serial Bus*) vodila ali zunanjega vira (5V)
- USB vmesnik za priklop na računalnik,
- pini s 3V in 5V napajanjem,

- 2 gumba,
- 8 LED (ang. *Light-Emitting Diode*) diod,
- audio D/A pretvornik z integriranim ojačevalcem,
- mikrofon,
- pospeškometer.

Prekinitveni krmilnik NVIC

Prekinitveni krmilnik NVIC (ang. *Nested Vectored Interrupt Controller*) upravlja s prekinitvenimi zahtevami, ki jih prožijo naprave, da bi prišle do storitev procesorja. Naslovi prvega ukaza prekinitveno servisnih programov so zapisani v vektorski tabeli, ki se nahaja na točno določenih naslovih v pomnilniku. NVIC omogoča do 240 različnih prekinitev, za vsako prekinitev pa tudi nastavljanje do 256 prioritetnih nivojev. Poleg navadnih prekinitev NVIC omogoča tudi zunanje prekinitve preko GPIO (ang. *General-Purpose Input/Output*) pinov s pomočjo EXTI (ang. *External Interrupt*) krmilnika. [13].

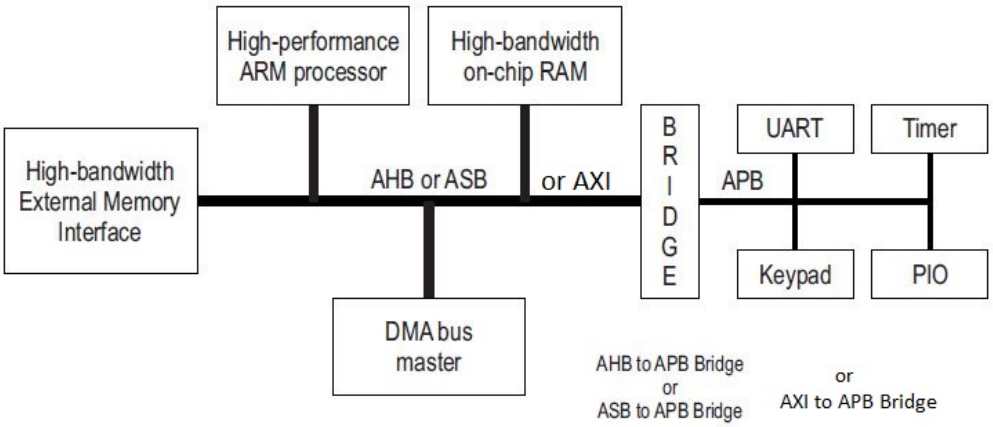
Mikrokrmilnik, ki smo ga uporabili za izdelavo alarmnega sistema, vsebuje 81 prekinitvenih kanalov [14].

Vodili APB in AHB

AMBA (ang. *The Advanced Microcontroller Bus Architecture*) je standard, ki določa vodila in protokole za komunikacijo med različnimi sistemskimi napravami in perifernimi enotami. Standard vključuje vodila, kot so APB (ang. *Advanced Peripheral Bus*), AHB (ang. *Advanced High-performance Bus*), AXI (ang. *Advanced eXtensible Interface*) in ACE (ang. *AXI Coherency Extensions*) [15].

APB vodilo je del AMBA hirearhije vodil in je narejeno za preproste periferne naprave z nizko porabo energije, ki ne potrebujejo hitrih prenosov. APB vodilo ima eno glavno in eno ali več podrejenih naprav. Glavna naprava

APB vodila je hkrati tudi most med APB vodilom in višjenivojskimi vodili, kot sta na primer AHB in AXI. S stališča AHB vodila je most med AHB in APB ena izmed podrejenih naprav AHB vodila. Prenosi med glavno in podrejeno napravo pri APB vodilu trajajo najmanj dve urini periodi [16].



Slika 4.3: Vodili AHB in APB ter most med njima, vir: [15]

AHB vodilo je namenjeno visokozmogljivim napravam, procesorjem, pomnilnikom, DMA krmilnikom itd. Na AHB vodilu je lahko največ 16 glavnih in 16 podrejenih naprav, v nekem trenutku pa lahko ena glavna naprava komunicira z eno podrejeno. Obstaja tudi verzija AHB vodila, ki je večplastno in omogoča več sočasnih prenosov med glavnimi in podrejenimi napravami [17].

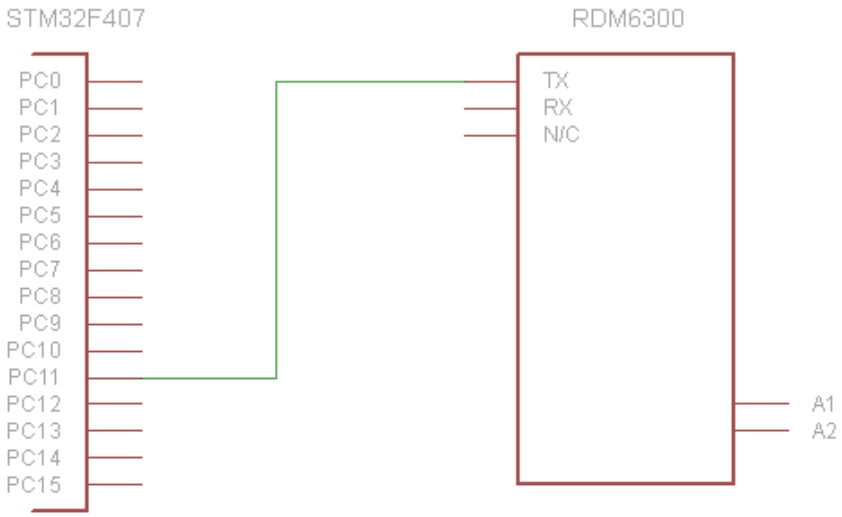
Na vodilu imamo tudi arbitra, ki določa, katera glavna naprava bo v določenem trenutku imela lastništvo nad vodilom. Eden izmed načinov izbire glavne naprave, ki jih uporablja arbiter, je izbira glavne naprave z najvišjo prioriteto. Poleg arbitra mora imeti AHB vodilo tudi dekodirno enoto, katere funkcija je, da na podlagi naslova aktivira izbirni (ang. *select*) signal za izbiro podrejene naprave, s katero želi glavna naprava komunicirati [17].

Za razliko od APB vodila AHB vodilo omogoča tudi eksplozijske prenose, ki so sestavljeni iz ene naslovne faze in več podatkovnih faz [17].

4.1.2 RFID čitalec

Pri izdelavi alarmnega sistema smo uporabili RFID čitalec z oznako RDM6300 za avtentikacijo uporabnika med njegovo interakcijo z alarmnim sistemom. Gre za 125KHz RFID modul (Slika 4.5), ki podpira branje značk tipa EM4100. Na mikrokrmilnik je priklopljen preko asinhronskega UART vmesnika in omogoča hitrost prenosa 9600 bps (ang. *baud per second*) [18].

RFID čitalec je priklopljen na 5V napajanje preko napajalnih pinov razvojne plošče. Pina A1 in A2 sta namenjena priklopu zunanje RFID antene, RFID čitalec pa je na mikrokrmilnik priklopljen preko pina PC11. Shema povezave RFID čitalca z razvojno ploščo je razvidna s Slike 4.4.

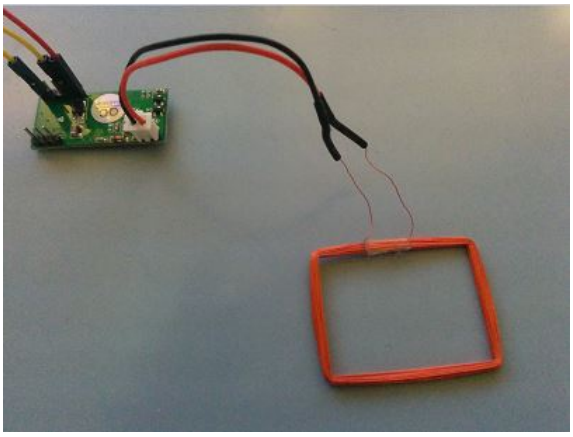


Slika 4.4: Shema povezave RFID čitalca z mikrokrmilnikom

Modul RDM6300, ki je bil uporabljen pri izdelavi alarmnega sistema, pošilja preko asinhronskega UART vmesnika sporočila v naslednjem formatu [18]:

- Start of text (0x02 HEX),
- ID značke,
- Checksum,

- End of text (0x03 HEX).



Slika 4.5: RFID čitalec RDM6300

4.1.3 Zaslon na dotik

Za prikaz informacij in interakcije s sistemom smo pri alarmnem sistemu uporabili rezistivni TFT (ang. *Thin-Film Transistor*) LCD zaslon na dotik (Slika 4.6), ki smo ga kupili preko spletne trgovine Ebay. LCD zaslon je proizvod neznanega proizvajalca, vsebuje pa LCD krmilnik SSD1289 in krmilnik za zaznavanje dotikov ADS7843.

Slika 4.6: TFT LCD zaslon

Lastnosti LCD zaslona:

- TFT LCD tip zaslona,
- LCD krmilnik SSD1289,
- krmilnik za zaznavanje dotikov ADS7843,
- velikost zaslona 320 x 240 pikslov,
- 65.536 različnih barv.

LCD krmilnik SSD1289

SSD1289 je krmilnik za TFT LCD zaslone, ki imajo resolucijo 320 x 240 pikslov in za prikaz slik uporabljajo do 262.144 barv. SSD1289 vsebuje grafični pomnilnik GDDRAM (ang. *Graphic Display Data RAM*) velikosti 172.800 bajtov, namenjen shranjevanju vzorcev, ki se prikažejo na zaslonu. Vsak

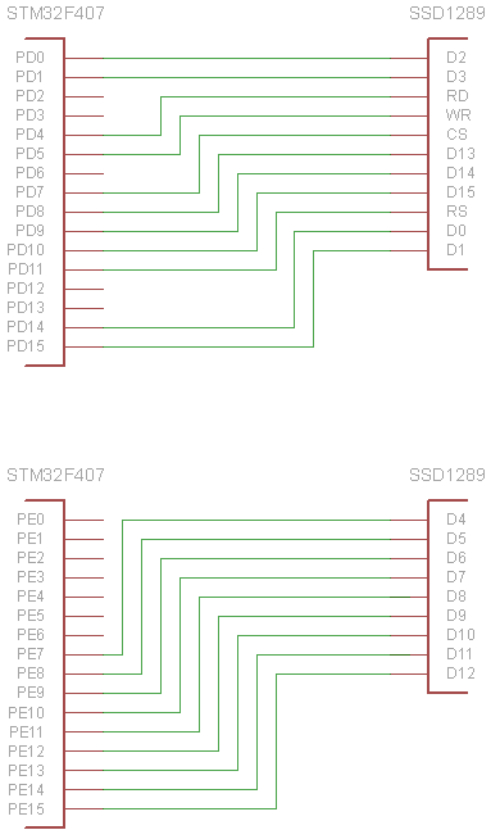
piksel je v pomnilniku GDDRAM predstavljen z največ 18 biti, kar pomeni šest bitov za rdečo, šest bitov za zeleno in šest bitov za modro barvo, to pa pomeni 262.144 različnih barv. V našem primeru uporabljamo LCD zaslon, ki prikazuje največ 65.536 različnih barv, z mikrokrmilnikom pa je povezan preko 16-bitnega vmesnika. Za prikaz 65.536 različnih barv potrebujemo pet bitov za rdečo, šest bitov za zeleno in pet bitov za modro barvo (Slika 4.7) [19].

			Hardware pins																	
Interface	Color mode	Cycle	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
18 bits	262k		R5	R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B5	B4	B3	B2	B1	B0
16 bits	262k	1st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
		2nd	B5	B4	B3	B2	B1	B0	x	x		R5	R4	R3	R2	R1	R0	x	x	
		3rd	G5	G4	G3	G2	G1	G0	x	x		B5	B4	B3	B2	B1	B0	x	x	
		1st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
		2nd	x	x	x	x	x	x	x	x		B5	B4	B3	B2	B1	B0	x	x	
		1st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
		2nd	B5	B4	B3	B2	B1	B0	x	x		x	x	x	x	x	x	x	x	
	65k		R4	R3	R2	R1	R0	G5	G4	G3		G2	G1	G0	B4	B3	B2	B1	B0	
9 bits	262k	1st										R5	R4	R3	R2	R1	R0	G5	G4	G3
		2nd										G2	G1	G0	B5	B4	B3	B2	B1	B0
8 bits	262k	1st										R5	R4	R3	R2	R1	R0	x	x	
		2nd										G5	G4	G3	G2	G1	G0	x	x	
		3rd										B5	B4	B3	B2	B1	B0	x	x	
	65k	1st										R4	R3	R2	R1	R0	G5	G4	G3	
		2nd										G2	G1	G0	B4	B3	B2	B1	B0	

Remark : x Don't care bits
Not connected pins

Slika 4.7: Predstavitev piksla pri uporabi različnih vmesnikov in barvnih načinov, vir: [19]

Mikrokrmilnik lahko komunicira s krmilnikom SSD1289 preko 8-/9-/16-/18-bitnih 6800/8080 paralelnih vmesnikov ali pa s pomočjo serijskega SPI vmesnika. Način komunikacije je določen s pomočjo pinov PS0-PS3 [19].



Slika 4.8: Povezava mikrokrmilnika s krmilnikom SSD1289

Za priklop LCD modula na mikrokrmilnik (Slika 4.7) smo morali na mikrokrmilniku rezervirati 20 pinov. Od tega je 16 pinov (D0–D15) namenjenih prenosu podatkov med LCD modulom in mikrokrmilnikom, štirje pini (RS, CS, WR in RD) pa so namenjeni kontroli.

Pini D0–D15 so namenjeni dvosmernemu prenosu podatkov med mikrokrmilnikom in LCD modulom [19].

Pin RS je namenjen izbiri lokacije, v katero krmilnik piše ali iz nje bere. Izbiramo lahko med registrom ali prikazovalnim pomnilnikom [19].

Pin WR se uporablja pri pisalnih ciklih. Z njim ukažemo krmilniku, naj prebere podatke iz podatkovnega vodila [19].

Pin RD se uporablja pri bralnih ciklih. Z njim ukažemo krmilniku, naj postavi podatke na podatkovno vodilo [19].

Pin CS je namenjen izbiri naprave. Naprava je izbrana, ko je pin v nizkem logičnem stanju [19].

Krmilnik ADS7843 za zaznavanje dotikov

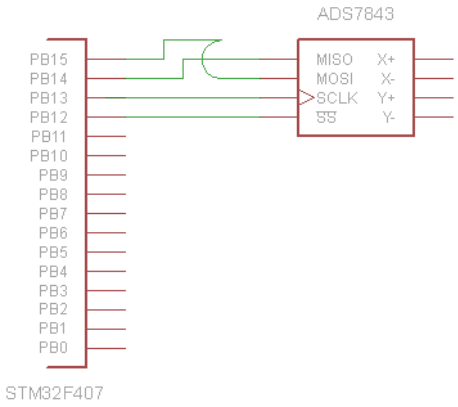
ADS7843 je krmilnik za rezistivne zaslone na dotik. Ima naslednje lastnosti [20]:

- napajalna napetost 2,7–5 V,
- SPI vmesnik za komunikacijo,
- 8- ali 12-bitna natančnost A/D pretvornika.

Zaslon je na krmilnik priklopljen preko štirih pinov, imenovanih X+, X–, Y+ in Y–. Analogni vhod je izbran preko štirikanalnega multiplekserja, posebna kombinacija stikal z nizko upornostjo pa omogoča uporabo prostih pinov za napajanje in ozemljitev priključkov rezistivnega zaslona pri branju koordinat [20].

Krmilnik deluje kot A/D pretvornik, ki koordinate zajema s pomočjo pinov X+ in Y+. Ko krmilnik zajema koordinato Y, je pin X+ vhod v A/D pretvornik, pina Y+ in Y– sta povezana na napajanje in ozemljitev. Podobno velja tudi za branje koordinate X, kjer je razlika samo v tem, da je pin Y+ vhod v A/D pretvornik, pina X+ in X– pa sta povezana na napajanje in ozemljitev [20].

Branje koordinat poteka tako, da mikrokrmilnik preko SPI vmesnika (Slika 4.9) pošlje posebej ukaz za branje koordinate X in posebej ukaz za branje koordinate Y. Za branje koordinat v 12-bitni natančnosti je treba posebej poslati ukaz 0xD0h za koordinato X in posebej ukaz 0x90h za koordinato Y [20].



Slika 4.9: Povezava mikrokrmilnika s krmilnikom ADS7843

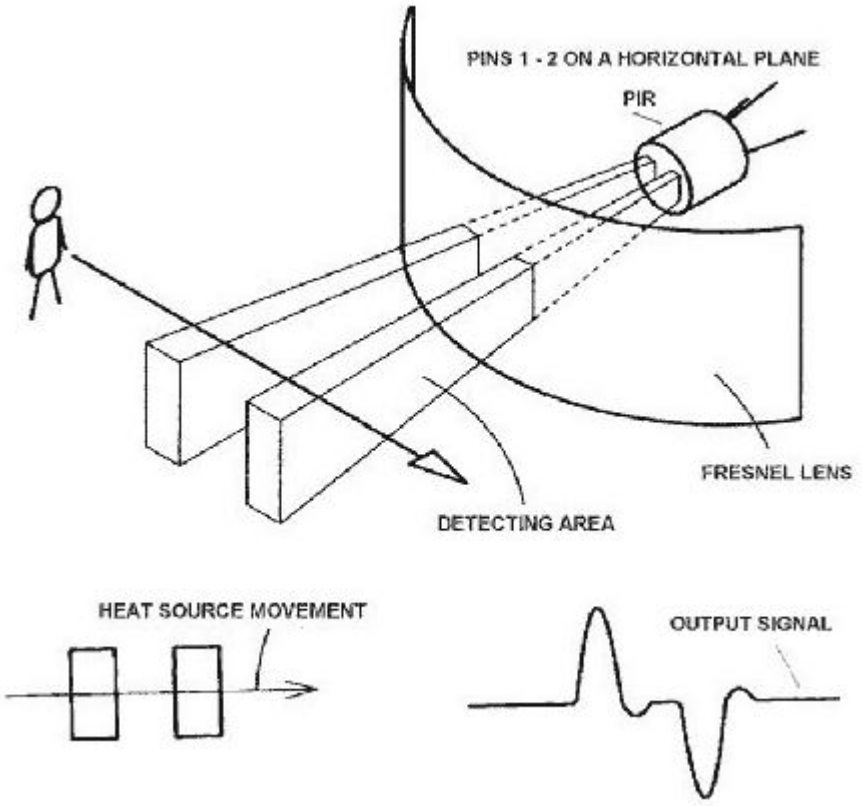
4.1.4 Infrardeči senzor za zaznavanje gibanja

Pri izdelavi alarmnega sistema smo uporabili dva infrardeča senzorja za zaznavanje gibanja, ki smo ju kupili preko spletne trgovine Ebay (Slika 4.10). Namen senzorjev je zaznavanje gibanja v prostorih, ko je alarmni sistem aktiven.

Slika 4.10: Infrardeči senzor za zaznavanje gibanja

Senzor za zaznavanje gibanja ima dve reži (Slika 4.11), ki vsebujeta material, občutljiv na infrardečo svetlobo. Ko senzor ne zaznava gibanja, obe

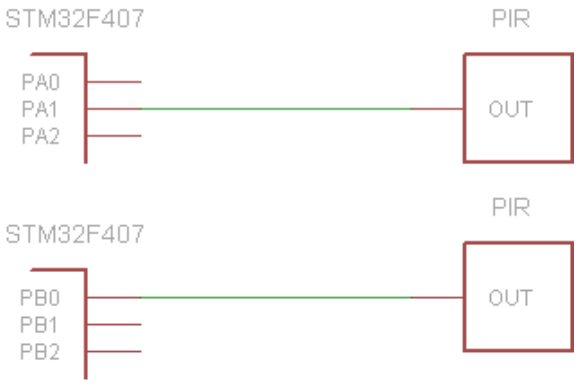
reži zaznavata enako količino infrardeče svetlobe, ki jo oddaja okolica. Ko bitje, ki oddaja infrardečo svetlobo (človek, živali itd.), prečka območje, ki ga pokrivata reži, najprej prečka območje prve reže, kar povzroči nekoliko bolj pozitiven signal na izhodu. Ko bitje zapušča območje, ki ga nadzira senzor, prečka tudi območje druge reže, kar povzroči nekoliko bolj negativen signal na izhodu. Spremembe v signalu pomenijo zaznavanje gibanja [21].



Slika 4.11: Delovanje infrardečega senzorja za zaznavanje gibanja, vir: [21]

Na Sliki 4.10 sta samo dva pasova za zaznavanje gibanja. Če želimo več pasov za zaznavanje gibanja v kotu med 0 in 180 stopinjami, moramo uporabiti Fresnelove leče, ki ustrezno fokusirajo infrardečo svetlobo [21].

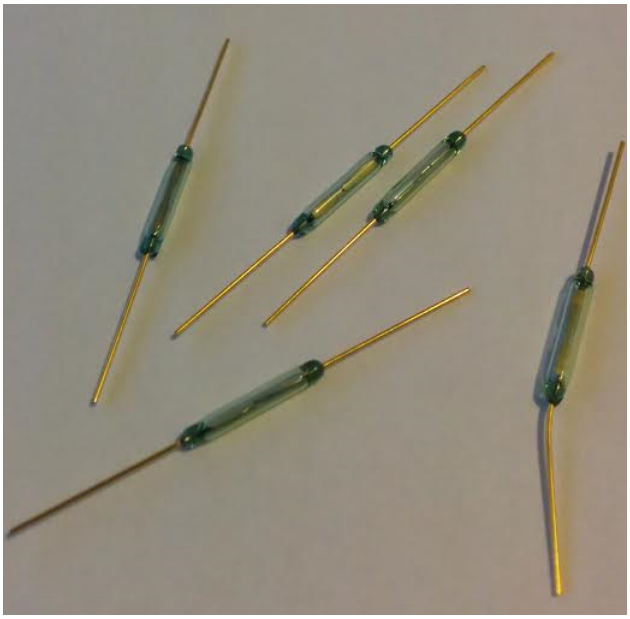
Senzorja sta na mikrokrmilnik priklopljena preko pinov PB0 in PA1 (Slika 4.12).



Slika 4.12: Povezava mikrokrmilnika s senzorji za zaznavanje gibanja

4.1.5 Reed stikalo

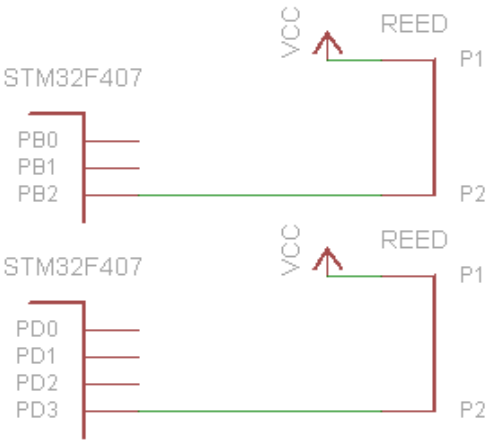
Pri izdelavi alarmnega sistema smo uporabili tudi reed stikala (Slika 4.13), ki smo jih, tako kot druge elemente, kupili preko spletne trgovine Ebay. Namen reed stikal je varovanje objekta pred vdori skozi okna, medtem ko je alarmni sistem aktiven. Ko nepridiprav na silo odpre okno, se reed stikalo sklene, kar sproži alarm.



Slika 4.13: Magnetna stikala

Reed stikalo je sestavljeno iz dveh feromagnetnih palčk, zaprtih v steklenem ohišju. Ko reed stikalu približamo magnet, se feromagnetni palčki v reed stikalu zaradi magnetnega polja stakneta in tako dobimo sklenjen tokokrog. Ko magnet umaknemo stran od stikala, se feromagnetni palčki razkleneta in tokokrog se prekine [22].

Reed stikali, ki smo ju uporabili pri alarmnem sistemu, sta na mikrokrmilnik priklopljeni preko pinov PB2 in PD3 (Slika 4.14).



Slika 4.14: Povezava mikrokrmilnika z reed stikali

4.1.6 Senzor loma stekla

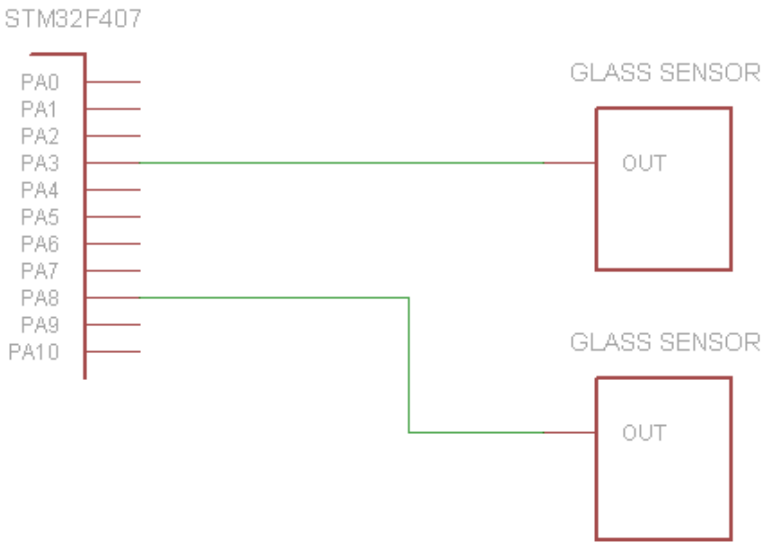
Pri alarmnem sistemu smo poleg zaščite oken z reed stikali uporabili tudi senzorje za zaznavo loma stekla (Slika 4.15), ki nudijo zaščito pred vdorom z razbitjem okenskega stekla. Senzorji so bili kupljeni preko spletne trgovine Ebay in so proizvod neznanega proizvajalca.

Slika 4.15: Senzor loma stekla

Senzor ima vgrajen mikrofon, namenjen analizi zvočnih frekvenc, ki prihajajo iz stekla, na katerega je pritrjen. Običajno je v senzor vgrajen mikrofon, ki zazna zvočne frekvence ob razbitju stekla. Če zvok preseže prag, ki ga je določil uporabnik, senzor sproži alarm [23].

Kompleksnejši senzorji izvajajo Fourierjevo analizo, s pomočjo katere primerjajo zvok s shranjenimi vzorci [23].

Senzorja, ki smo ju uporabili pri alarmnem sistemu, sta na mikrokrmilnik priklopljena preko pinov PA3 in PA8 (Slika 4.16).



Slika 4.16: Povezava mikrokrmilnika s senzorji loma stekla

4.1.7 Senzor plinov

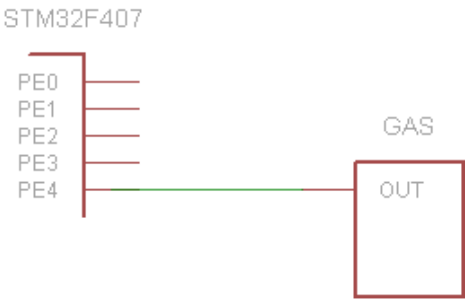
Poleg že omenjene zaščite je pomembna tudi zaščita pred uhajanjem različnih strupenih plinov v objektu, zato smo alarmnemu sistemu dodali senzor plinov (Slika 4.17). Senzor ima možnost zaznave dima, metana in butana.

Senzor plinov je naprava, ki zazna prisotnost različnih strupenih plinov v okolju, kjer se zadržujejo ljudje in živali. Ko senzor zazna povečane koncentracije strupenih plinov, se sproži alarm, ki ljudi opozori na nevarnost [24].

Slika 4.17: Senzor za zaznavanje strupenih plinov

V praksi poznamo več različnih tipov senzorjev, ki se razlikujejo med seboj glede na način delovanja. Tako poznamo elektrokemične, ultrazvočne, holografske in druge vrste senzorjev [24].

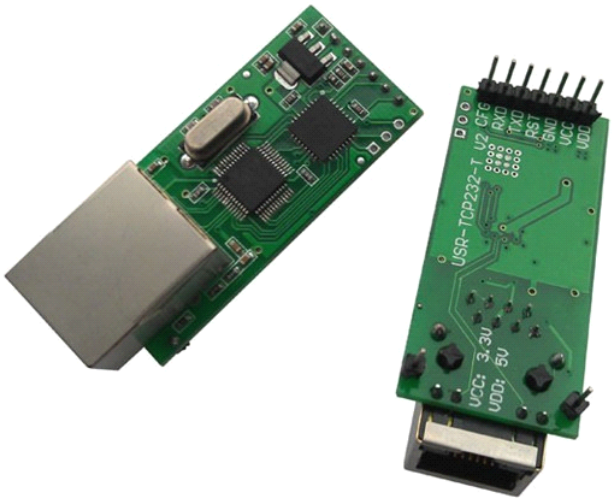
Senzor za zaznavanje plinov je na mikrokrmilnik priklopljen preko pina PE4 (Slika 4.18).



Slika 4.18: Povezava mikrokrmilnika s senzorjem plinov

4.1.8 Ethernet modul

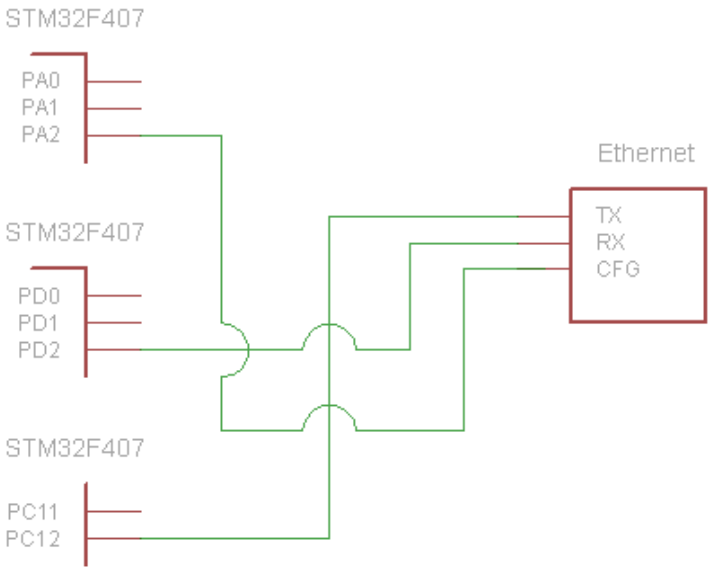
Za povezavo alarmnega sistema z nadzornim strežnikom smo uporabili ethernet modul (Slika 4.19), ki je na mikrokrmilnik priklopljen preko asinhronega UART vmesnika. Ethernet modul ima že realiziran TCP/IP sklad, deluje pa lahko kot TCP odjemalec, TCP strežnik, UDP odjemalec ali UDP strežnik [25].



Slika 4.19: Ethernet modul, vir: [25]

Branje in pošiljanje podatkov poteka preko UART vmesnika. Ko ethernet modul sprejme UDP ali TCP pakete, njihovo vsebino preko UART vmesnika posreduje mikrokrmilniku. Ko ethernet modul preko UART vmesnika dobi vsebino, ki jo mikrokrmilnik želi poslati, jo zapakira v TCP ali UDP pakete in pošlje v omrežje [25].

Ethernet modul je na mikrokrmilnik priklopljen preko pinov PC12, PD2 in PA2. Pina TX in RX sta namenjena pošiljanju in sprejemanju podatkov preko UART vmesnika, pin CFG pa je namenjen postavitvi ethernet modula v nastavitveni način (Slika 4.20).



Slika 4.20: Povezava mikrokrmilnika z ethernet modulom

4.1.9 GSM modul

Ko eden izmed prej opisanih senzorjev sproži alarm, je treba o tem takoj obvestiti tudi uporabnike objekta. V ta namen smo uporabili GSM modul (Slika 4.21), ki s pomočjo SMS sporočila uporabnike objekta obvesti, da je prišlo do morebitnega vdora v objekt. Modul smo tako kot ostale elemente kupili preko spletne trgovine Ebay.

Slika 4.21: GSM modul TC35

GSM modul TC35 deluje v frekvenčnih območjih 900 MHz in 1800 MHz, omogoča pa pošiljanje SMS sporočil in telefonske klice. Dostop do GSM modula je mogoč preko UART vmesnika, s pomočjo katerega pošiljamo AT (ang. *Attention*) ukaze, ki omogočajo nastavljanje GSM modula, telefonske pogovore in pošiljanje SMS sporočil [26].

AT ukaze uporabljamo za kontrolo modula. Z nizom AT najavimo začetek ukaza, nato sledi ukaz, ki ga končamo z znakom CR (ang. *carriage return*) [27].

GSM modul TC35 je na mikrokrmilnik priklopljen preko pina PC6. Pin TX je namenjen sprejemanju AT ukazov preko UART vmesnika (Slika 4.22).

Slika 4.22: Povezava mikrokrmilnika z GSM modulom

4.2 Programska rešitev

Za programiranje mikrokrmilnika STM32F407VGT6 smo uporabili orodje IAR Embedded Workbench, ki je namenjeno programiranju 8-/16-/32-bitnih mikrokrmilnikov [28]. Nalaganje programa v bliskovni pomnilnik mikrokrmilnika poteka preko na razvojni plošči vgrajenega ST-LINK/V2 programatorja/razhroščevalnika, ki s pomočjo JTAG/SWD vmesnika komunicira z mikrokrmilnikom. Za lažje programiranje mikrokrmilnika smo uporabili knjižnice, ki jih je mogoče dobiti na spletni strani proizvajalca STMicroelectronics [29].

4.2.1 Programiranje mikrokrmilnika

Pri programiranju smo si pomagali z osnovnimi knjižnicami, ki smo jih dobili na spletni strani proizvajalca STMicroelectronics. Preden smo začeli programirati, smo v programu IAR Embedded Workbench naredili nov projekt z vsemi nastavitvami, ki so potrebne za pravilno delovanje mikrokrmilnika (nastavitev frekvence sistemske ure, množilnikov in delilnikov ure, bliskovnega pomnilnika ...).

V glavnem programu smo inicializirali vsa opravila, ki se izvajajo. Poleg opravil smo inicializirali tudi strukture za komunikacijo med opravili, omogočili pa smo tudi vse potrebne ure in inicializirali vse naprave, ki so priključene na mikrokrmilnik. Generirali smo tudi vse podključe, ki so po-

trebni za pravilno delovanje kriptirnega algoritma AES. Na koncu glavnega programa smo poklicali funkcijo, ki zažene razporejevalnik opravil.

4.2.2 Nastavitev LCD krmilnika SSD1289

Ker imajo vse naprave ob vklopu onemogočene ure, je bilo najprej treba omogočiti uro za splošnonamenska vrata GPIOD in GPIOE, ki so povezana na vodilo AHB1. Uro smo omogočili s funkcijo RCC_AHB1PeriphClockCmd (Slika 4.23). Poleg tega je bilo treba vklopiti uro za FSMC (ang. *Flexible Static Memory Controller*) vmesnik, kar smo storili z ukazom RCC_AHB3PeriphClockCmd (Slika 4.24).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE); // Touch, Ethernet, LCD
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE, ENABLE); // Lcd
```

Slika 4.23: Vklop ure za splošnonamenska vrata GPIOD in GPIOE

```
RCC_AHB3PeriphClockCmd(RCC_AHB3Periph_FSMC, ENABLE); // Lcd
```

Slika 4.24: Vklop ure za FSMC krmilnik

Nato smo nastavili pine, ki smo jih uporabili za komunikacijo z LCD krmilnikom. Pine smo povezali s FSMC vmesnikom (Slika 4.25).

```
GPIO_InitTypeDef GPIO_InitStructure;


GPIO_PinAFConfig(GPIOD, GPIO_PinSource0, GPIO_AF_FSMC);              // D2
GPIO_PinAFConfig(GPIOD, GPIO_PinSource1, GPIO_AF_FSMC);              // D3
GPIO_PinAFConfig(GPIOD, GPIO_PinSource4, GPIO_AF_FSMC);              // NOE -> RD
GPIO_PinAFConfig(GPIOD, GPIO_PinSource5, GPIO_AF_FSMC);              // NWE -> WR
GPIO_PinAFConfig(GPIOD, GPIO_PinSource7, GPIO_AF_FSMC);              // NE1 -> CS
GPIO_PinAFConfig(GPIOD, GPIO_PinSource8, GPIO_AF_FSMC);              // D13
GPIO_PinAFConfig(GPIOD, GPIO_PinSource9, GPIO_AF_FSMC);              // D14
GPIO_PinAFConfig(GPIOD, GPIO_PinSource10, GPIO_AF_FSMC);             // D15
GPIO_PinAFConfig(GPIOD, GPIO_PinSource11, GPIO_AF_FSMC);             // A16 -> RS
GPIO_PinAFConfig(GPIOD, GPIO_PinSource14, GPIO_AF_FSMC);             // D0
GPIO_PinAFConfig(GPIOD, GPIO_PinSource15, GPIO_AF_FSMC);             // D1

GPIO_PinAFConfig(GPIOE, GPIO_PinSource7, GPIO_AF_FSMC);              // D4
GPIO_PinAFConfig(GPIOE, GPIO_PinSource8, GPIO_AF_FSMC);              // D5
GPIO_PinAFConfig(GPIOE, GPIO_PinSource9, GPIO_AF_FSMC);              // D6
GPIO_PinAFConfig(GPIOE, GPIO_PinSource10, GPIO_AF_FSMC);             // D7
GPIO_PinAFConfig(GPIOE, GPIO_PinSource11, GPIO_AF_FSMC);             // D8
GPIO_PinAFConfig(GPIOE, GPIO_PinSource12, GPIO_AF_FSMC);             // D9
GPIO_PinAFConfig(GPIOE, GPIO_PinSource13, GPIO_AF_FSMC);             // D10
GPIO_PinAFConfig(GPIOE, GPIO_PinSource14, GPIO_AF_FSMC);             // D11
GPIO_PinAFConfig(GPIOE, GPIO_PinSource15, GPIO_AF_FSMC);             // D12

GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0 | GPIO_Pin_1 | GPIO_Pin_4 | GPIO_Pin_5 |
                              GPIO_Pin_7 | GPIO_Pin_8 | GPIO_Pin_9 | GPIO_Pin_10 |
                              GPIO_Pin_11 | GPIO_Pin_14 | GPIO_Pin_15;

GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd  = GPIO_PuPd_NOPULL;
GPIO_Init(GPIOD, &GPIO_InitStructure);

GPIO_InitStructure.GPIO_Pin = GPIO_Pin_7 | GPIO_Pin_8 | GPIO_Pin_9 | GPIO_Pin_10 |
                              GPIO_Pin_11 | GPIO_Pin_12 | GPIO_Pin_13 | GPIO_Pin_14 |
                              GPIO_Pin_15;

GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_100MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd  = GPIO_PuPd_NOPULL;

GPIO_Init(GPIOE, &GPIO_InitStructure);
```

Slika 4.25: Nastavitev vseh potrebnih pinov

Poleg pinov smo morali nastaviti tudi FSMC vmesnik. Nastavitve FSMC vmesnika prikazuje Slika 4.26.

```
FSMC_NORSRAMInitTypeDef  FSMC_NORSRAMInitStructure;
FSMC_NORSRAMTimingInitTypeDef FSMC_NORSRAMTimingInitStructure;
FSMC_NORSRAMTimingInitStructure.FSMC_AddressSetupTime = 0;  //0
FSMC_NORSRAMTimingInitStructure.FSMC_AddressHoldTime = 0;   //0
FSMC_NORSRAMTimingInitStructure.FSMC_DataSetupTime = 2;     //3
FSMC_NORSRAMTimingInitStructure.FSMC_BusTurnAroundDuration = 0;
FSMC_NORSRAMTimingInitStructure.FSMC_CLKDivision = 1;//1
FSMC_NORSRAMTimingInitStructure.FSMC_DataLatency = 0;
FSMC_NORSRAMTimingInitStructure.FSMC_AccessMode = FSMC_AccessMode_A;

FSMC_NORSRAMInitStructure.FSMC_Bank = FSMC_Bank1_NORSRAM1;
FSMC_NORSRAMInitStructure.FSMC_DataAddressMux = FSMC_DataAddressMux_Disable;
FSMC_NORSRAMInitStructure.FSMC_MemoryType = FSMC_MemoryType_SRAM;
FSMC_NORSRAMInitStructure.FSMC_MemoryDataWidth = FSMC_MemoryDataWidth_16b;
FSMC_NORSRAMInitStructure.FSMC_BurstAccessMode = FSMC_BurstAccessMode_Disable;
FSMC_NORSRAMInitStructure.FSMC_WaitSignalPolarity = FSMC_WaitSignalPolarity_Low;
FSMC_NORSRAMInitStructure.FSMC_WrapMode = FSMC_WrapMode_Disable;
FSMC_NORSRAMInitStructure.FSMC_WaitSignalActive = FSMC_WaitSignalActive_BeforeWaitState;
FSMC_NORSRAMInitStructure.FSMC_WriteOperation = FSMC_WriteOperation_Enable;
FSMC_NORSRAMInitStructure.FSMC_WaitSignal = FSMC_WaitSignal_Disable;
FSMC_NORSRAMInitStructure.FSMC_AsynchronousWait = FSMC_AsynchronousWait_Disable;
FSMC_NORSRAMInitStructure.FSMC_ExtendedMode = FSMC_ExtendedMode_Disable;
FSMC_NORSRAMInitStructure.FSMC_WriteBurst = FSMC_WriteBurst_Enable;//disable
FSMC_NORSRAMInitStructure.FSMC_ReadWriteTimingStruct = &FSMC_NORSRAMTimingInitStructure;

FSMC_NORSRAMInit(&FSMC_NORSRAMInitStructure);
FSMC_NORSRAMTimingInitStructure.FSMC_AddressSetupTime = 0;    //0
FSMC_NORSRAMTimingInitStructure.FSMC_AddressHoldTime = 0;     //0
FSMC_NORSRAMTimingInitStructure.FSMC_DataSetupTime = 4;       //3
FSMC_NORSRAMTimingInitStructure.FSMC_BusTurnAroundDuration = 0;
FSMC_NORSRAMTimingInitStructure.FSMC_CLKDivision = 1;//1
FSMC_NORSRAMTimingInitStructure.FSMC_DataLatency = 0;
FSMC_NORSRAMTimingInitStructure.FSMC_AccessMode = FSMC_AccessMode_A;
FSMC_NORSRAMInitStructure.FSMC_WriteTimingStruct = &FSMC_NORSRAMTimingInitStructure;
FSMC_NORSRAMInit(&FSMC_NORSRAMInitStructure);

FSMC_NORSRAMCmd(FSMC_Bank1_NORSRAM1, ENABLE);
```

Slika 4.26: Nastavitev FSMC vmesnika

Inicializacijo LCD krmilnika smo zaključili z nizom ukazov, ki nastavijo registre LCD krmilnika na točno določene vrednosti, da zaslon lahko pravilno deluje. Slika 4.27 prikazuje del inicializacijske sekvence.

```
LCD_WriteReg(0x0007,0x0021);    Delay(50);
LCD_WriteReg(0x0000,0x0001);    Delay(50);
LCD_WriteReg(0x0007,0x0023);    Delay(50);
LCD_WriteReg(0x0010,0x0000);    Delay(90);
LCD_WriteReg(0x0007,0x0033);    Delay(50);
LCD_WriteReg(0x0011,0x6830);    Delay(50);
LCD_WriteReg(0x0002,0x0600);    Delay(50);
LCD_WriteReg(0x0012,0x6CEB);    Delay(50);
LCD_WriteReg(0x0003,0xA8A4);    Delay(50);
```

Slika 4.27: Del inicializacijske sekvence LCD krmilnika

Za zapis piksla na zaslon uporabimo funkciji PutPixel ali Pixel (Slika 4.28). Funkciji najprej postavita kazalec na natančno določeno lokacijo v prikazovalnem pomnilniku GDDRAM, ki ustreza točki T (X, Y) na zaslonu. To storita tako, da v registra 4Eh in 4Fh vpišeta koordinati X in Y. Funkciji nato na izbrano lokacijo v prikazovalnem pomnilniku zapišeta barvo piksla. To storita tako, da v register 22h vpišeta izbrano barvo v 16-bitnem formatu.

```
void PutPixel(int16_t x, int16_t y)
{
        if (( x > 239 )||( y > 319 )) return;
        LCD_SetCursor(x, y);
        LCD_WriteRAM_Prepare();
        LCD_WriteRAM(TextColor);
}

void Pixel(int16_t x, int16_t y,int16_t c)
{
        if (( x > 239 )||( y > 319 )) return;
        LCD_SetCursor(x,y);
        LCD_WriteRAM_Prepare();
        LCD_WriteRAM(c);
}
```

Slika 4.28: Funkciji za zapis piksla na zaslon

4.2.3 Nastavitev krmilnika za zaznavanje dotikov ADS7843

Tudi pri krmilniku za zaznavanje dotikov je bilo treba omogočiti uro. Uro smo omogočili na splošnonamenskih vratih GPIOB, ki so povezana na vodilo AHB1. To smo storili tako kot pri LCD krmilniku, z uporabo funkcije RCC_AHB1PeriphClockCmd (Slika 4.29).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE); // Touch, Reed, Pir
```

Slika 4.29: Vklop ure za splošnonamenska vrata GPIOB

Po vklopu ure smo nastavili pine, ki smo jih uporabili za komunikacijo s krmilnikom za zaznavanje dotikov. Pine smo povezali s vmesnikom SPI2, nato smo nastavili tudi vmesnik SPI2, in sicer tako, da omogoča dvosmerne prenose, v enem prenosu pa se prenese osem bitov (Slika 4.30).

```
GPIO_InitStruct.GPIO_OType=GPIO_OType_PP;
GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_UP;

GPIO_InitStruct.GPIO_Pin=GPIO_Pin_15|GPIO_Pin_13|GPIO_Pin_14;
GPIO_Init(GPIOB,&GPIO_InitStruct);

GPIO_PinAFConfig(GPIOB, GPIO_PinSource13, GPIO_AF_SPI2);      //sclk  10       13
GPIO_PinAFConfig(GPIOB, GPIO_PinSource14, GPIO_AF_SPI2);      //mýso  11       14
GPIO_PinAFConfig(GPIOB, GPIO_PinSource15, GPIO_AF_SPI2);      //mosý  12       15

SPI_I2S_DeInit(SPI2);
SPI_InitStructure.SPI_Direction = SPI_Direction_2Lines_FullDuplex;
SPI_InitStructure.SPI_Mode = SPI_Mode_Master;
SPI_InitStructure.SPI_DataSize = SPI_DataSize_8b;
SPI_InitStructure.SPI_CPOL = SPI_CPOL_High;//SPI_CPOL_Low      SPI_CPOL_High
SPI_InitStructure.SPI_CPHA = SPI_CPHA_2Edge;
SPI_InitStructure.SPI_NSS = SPI_NSS_Soft;   //SPI_NSS_Hard     //SPI_NSS_Soft
SPI_InitStructure.SPI_BaudRatePrescaler = SPI_BaudRatePrescaler_256;  //16
SPI_InitStructure.SPI_FirstBit = SPI_FirstBit_MSB;
SPI_InitStructure.SPI_CRCPolynomial = 7;
SPI_Init(SPI2,&SPI_InitStructure);
SPI_Cmd(SPI2,ENABLE);

//CS

GPIO_InitStruct.GPIO_Mode=GPIO_Mode_OUT;
GPIO_InitStruct.GPIO_Speed=GPIO_Speed_50MHz;
GPIO_InitStruct.GPIO_OType=GPIO_OType_PP;
GPIO_InitStruct.GPIO_PuPd=GPIO_PuPd_UP;
GPIO_InitStruct.GPIO_Pin=GPIO_Pin_12; // 3
GPIO_Init(GPIOB,&GPIO_InitStruct);    // d
```

Slika 4.30: Nastavitev pinov in vmesnika SPI2

Branje koordinat X in Y poteka s pomočjo funkcij TPReadX in TPReadY (Slika 4.31). Za branje koordinate X moramo preko vmesnika SPI2 poslati ukaz 0xD0h, za koordinato Y pa ukaz 0x90h. Ker je rezultat A/D pretvorbe 12-biten, moramo za pravilen rezultat opraviti dvojno branje.

Ker pri branju koordinat X in Y pride do nihanja vrednosti koordinat, je treba opraviti več zaporednih odčitkov. To storimo z uporabo funkcij getAndNormalizeX in getAndNormalizeY, ki vrneta srednjo vrednost več zaporednih odčitkov.

Koordinate, ki smo jih prebrali s pomočjo funkcij TPReadX, TPReadY, getAndNormalizeX in getAndNormalizeY, je treba pomnožiti s kalibracijsko matriko, ki vsebuje koeficiente, s pomočjo katerih dobimo prave koordinate na LCD zaslonu. Koeficiente kalibracijske matrike izračunamo s pomočjo

algoritma za kalibracijo zaslona.

```
uint16_t TPReadX(void)
{
   uint16_t x=0;
   T_CS();
   SpiDelay(10);
   SPI_WriteByte(0xd0);
   SpiDelay(10);
   x=SPI_WriteByte(0x0);
   x<<=8;
   x |=SPI_WriteByte(0x0);
   T_DCS();
   x = x>>3;
   x &= 0xFFF;   //fff
   return (uint16_t)(x * (240.0 / 4095));
}


uint16_t TPReadY(void)
{
  int16_t y=0;
  T_CS();
  SpiDelay(10);
  SPI_WriteByte(0x90);
  SpiDelay(10);
  y=SPI_WriteByte(0x0);
  y<<=8;
  y |=SPI_WriteByte(0x0);
  T_DCS();
  y=y>>3;
  y &= 0xFFF; //fff
  y=4095-y;
  return (uint16_t)(y * (320.0 / 4095)); // You have to convert coords as follows: (y - 320) * -1
}
```

Slika 4.31: Branje koordinat X in Y

Kalibracija zaslona

Vedno več naprav uporablja zaslon na dotik. Če želimo, da zaslon na dotik pravilno deluje, ga je treba pred uporabo kalibrirati. Kalibracija je potrebna zato, ker je težko poravnati koordinate folije, občutljive na dotik, s koordinatami LCD zaslona. Če kalibracije zaslona ne opravimo, se sistem ne odziva pravilno, kadar pritisnemo na zaslon [30].

V praksi poznamo tri vire napak, ki vplivajo na pravilno delovanje zaslona na dotik [30]:

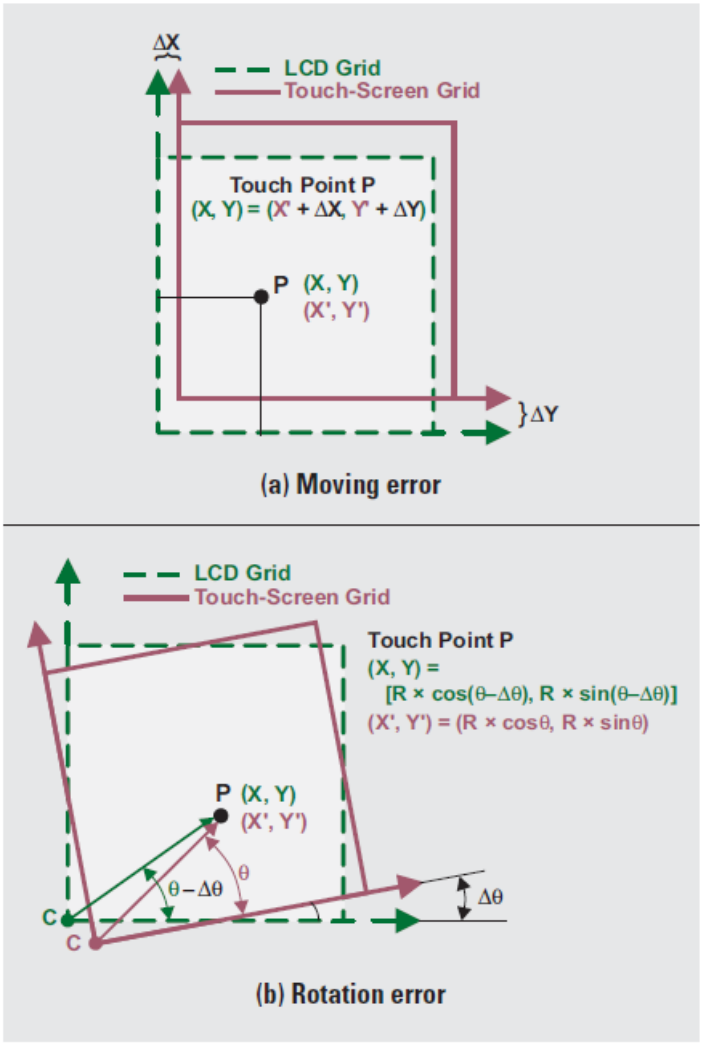
- električni šum,

- skalirni faktorji,

- mehanska neporavnanost.

Električni šum nastane zaradi okolja, zaslona in osvetlitve zaslona. Skalirni faktorji in mehanska neporavnanost so posledica neusklajenosti med folijo, občutljivo na dotik, in LCD zaslonom. Napake zaradi mehanske neporavnanosti in skalirnih faktorjev odpravljamo s pomočjo kalibracije.

Na Sliki 4.32 vidimo, da mehansko neporavnanost sestavljajo "premikalne"napake (ang. *moving errors*) in "krožne"napake (ang. *rotation errors*). P (X, Y) je točka na zaslonu, P (X', Y') pa točka na foliji, občutljivi na dotik. ΔX je sprememba v smeri X, ΔY pa je sprememba v smeri Y. R je razdalja od kordinatnega izhodišča do točke P. $\Delta\theta$ je relativni zamik med LCD zaslonom in folijo, občutljivo na dotik.



Slika 4.32: Mehanska neporavnanost zaslona, vir: [30]

Z upoštevanjem skalirnih faktorjev, "premikalne"napake in "krožne"napake lahko koordinato X zapišemo kot:

$$\begin{aligned} X &= Kx * R * cos(\theta - \Delta\theta) + \Delta X \\ &= Kx * R * cos\theta * cos(\Delta\theta) + Kx * R * sin\theta * sin(\Delta\theta) + \Delta X \\ &= Kx * X' * cos(\Delta\theta) + Kx * Y' * sin(\Delta\theta) + \Delta X \\ &= \alpha x * X' + \beta x * Y' + \Delta X \end{aligned} \tag{4.1}$$

Koordinato Y pa zapišemo kot:

$$\begin{aligned} Y &= Ky * R * sin(\theta - \Delta\theta) + \Delta Y \\ &= Ky * R * sin\theta * cos(\Delta\theta) - Ky * R * cos\theta * sin(\Delta\theta) + \Delta Y \\ &= Ky * Y' * cos(\Delta\theta) - Ky * X' * sin(\Delta\theta) + \Delta Y \\ &= \alpha y * X' + \beta y * Y' + \Delta Y \end{aligned} \tag{4.2}$$

Iz enačb 4.1 in 4.2 je razvidno, da potrebujemo najmanj tri nekolinearne točke, če želimo izračunati koeficiente αx, αy, βx, βy, ΔX in ΔY. Predpostavimo, da imamo tri točke, (X1, Y1), (X2, Y2) in (X3, Y3), na LCD zaslonu. Točke (X1', Y1'), (X2', Y2') in (X3', Y3') ustrezajo točkam na foliji, občutljivi na dotik. Enačbi 4.1 in 4.2 lahko uporabimo, da sestavimo naslednji sistem enačb:

$$\begin{aligned} X1 &= \alpha x * X1' + \beta x * Y1' + \Delta X \\ X2 &= \alpha x * X2' + \beta x * Y2' + \Delta X \\ X3 &= \alpha x * X3' + \beta x * Y3' + \Delta X \end{aligned} \tag{4.3}$$

$$\begin{aligned} Y1 &= \alpha y * X1' + \beta y * Y1' + \Delta Y \\ Y2 &= \alpha y * X2' + \beta y * Y2' + \Delta Y \\ Y3 &= \alpha y * X3' + \beta y * Y3' + \Delta Y \end{aligned}$$

Enačbo 4.3 lahko zapišemo tudi v matrični obliki:

$$\begin{bmatrix} X1 \\ X2 \\ X3 \end{bmatrix} = A * \begin{bmatrix} \alpha x \\ \beta x \\ \Delta X \end{bmatrix} in \begin{bmatrix} Y1 \\ Y2 \\ Y3 \end{bmatrix} = A * \begin{bmatrix} \alpha y \\ \beta y \\ \Delta Y \end{bmatrix} \tag{4.4}$$

kjer je

$$A = \begin{bmatrix} X1' & Y1' & 1 \\ X2' & Y2' & 1 \\ X3' & Y3' & 1 \end{bmatrix}$$

Za izračun vseh potrebnih koeficientov je treba rešiti enačbo 4.4 [30].

Algoritem za kalibracijo zaslona je sestavljen iz naslednjih korakov [30]:

- izberemo kalibracijske točke na zaslonu, (Xk, Yk) za k = $1, 2, ..., n$ in $n \geq 3$,
- treba je poklicati funkcijo za pridobitev koordinat iz krmilnika za zaznavanje dotikov,
- dotakniti se je treba prve točke (X1, Y1) na zaslonu, pridobiti koordinati X in Y iz krmilnika za zaznavanje dotikov, nato pa koordinati shraniti kot (X1', Y2'),
- ponoviti je treba predhodni korak, da pridobimo vse točke (Xk', Yk') za k = $2, 3, ..., n$ in $n \geq 3$,
- izračunati je treba koeficiente $\alpha x, \beta x, \Delta X, \alpha y, \beta y$ in ΔY.

Izračunane koeficiente $\alpha x, \beta x, \Delta X, \alpha y, \beta y$ in ΔY uporabimo v enačbah 4.1 in 4.2. Za izračun koeficientov smo uporabili funkcijo calibrateTouch, ki smo jo sami napisali.

4.2.4 Nastavitev RFID čitalca

Za pravilno delovanje RFID čitalca je bilo treba omogočiti uro na splošnonamenskih vratih GPIOC in vmesniku UART4. To smo storili z uporabo funkcij RCC_APB1PeriphClockCmd in RCC_AHB1PeriphClockCmd (Slika 4.33).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE); // Enable clock for GPIOC...
RCC_APB1PeriphClockCmd(RCC_APB1Periph_UART4, ENABLE); // Enable UART4 clock
```

Slika 4.33: Vklop ure za splošnonamenska vrata GPIOC ter vmesnik UART4

Nato smo nastavili pine, ki smo jih povezali z vmesnikom UART4. Poleg pinov je bilo treba nastaviti tudi vmesnik UART4. Hitrost prenosa na vmesniku UART4 smo nastavili na 9600 bps. Prenos podatkov poteka v formatu 8N1, kar pomeni, da prenašamo osem podatkovnih bitov in en stop bit, ne prenašamo pa paritetnega bita (Slika 4.34).

```
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_11; // Pin for RX
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_Init(GPIOC,&GPIO_InitStructure);

GPIO_PinAFConfig(GPIOC,GPIO_PinSource11,GPIO_AF_UART4); // AF function..rx

USART_InitStructure.USART_BaudRate = 9600;
USART_InitStructure.USART_WordLength = USART_WordLength_8b;
USART_InitStructure.USART_StopBits = USART_StopBits_1;
USART_InitStructure.USART_Parity = USART_Parity_No;
USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_InitStructure.USART_Mode = USART_Mode_Rx;
USART_Init(UART4, &USART_InitStructure);
```

Slika 4.34: Nastavitev pinov in vmesnika UART4

Za prenos podatkov iz RFID čitalca smo uporabili DMA krmilnik, s pomočjo katerega smo dosegli, da se prekinitev sproži samo takrat, ko se v pomnilnik prenese celotna vsebina RFID značke (14 bajtov podatkov). Na ta način smo

procesor razbremenili tega, da za vsak preneseni znak odgovori na prekinitveno zahtevo, ki jo sproži vmesnik UART4.

DMA krmilnik

DMA krmilnik napravam daje neposredni dostop do pomnilnika, kar omogoča prenašanje podatkov iz naprave v pomnilnik ali iz pomnilnika v napravo brez vpletanja centralne procesne enote. Ta pri postopku sodeluje samo na začetku, ko DMA krmilniku poda zahtevek za prenos, ter na koncu prenosa, ko DMA krmilnik sproži prekinitveno zahtevo [31].

Mikrokrmilnik STM32F407VG vsebuje dva DMA krmilnika. Vsak DMA krmilnik vsebuje osem tokov (ang. *streams*), vsak tok pa osem kanalov (ang. *channels*). Tokovi omogočajo dvosmerno povezavo med DMA krmilnikom in napravami, na vsak tok pa je priklopljenih osem kanalov, preko katerih naprave prožijo DMA zahteve. DMA krmilnika omogočata več vrst prenosov [32]:

- iz naprave v pomnilnik,
- iz pomnilnika v napravo,
- iz pomnilnika v pomnilnik (to omogoča samo krmilnik DMA2).

Za prenos podatkov iz RFID čitalca v pomnilnik smo uporabili krmilnik DMA1. Da bi krmilnik DMA1 deloval, je bilo najprej treba vklopiti uro. To smo storili s funkcijo RCC_AHB1PeriphClockCmd (Slika 4.35).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_DMA1, ENABLE); // Enable DMA1 clock
```

Slika 4.35: Vklop ure na krmilniku DMA1

Po vklopu ure smo krmilnik DMA1 nastavili tako, da uporablja kanal štiri (UART rx) in tok dve. Izbrali smo prenos iz naprave v pomnilnik, prebrane vrednosti pa se shranjujejo v polje tagValues. Nastavili smo tudi, da

prenašamo podatke dolžine enega bajta. Poleg dolžine smo morali nastaviti tudi število prenesenih bajtov, ki je v našem primeru štirinajst. Omogočili smo tudi prekinitev, ki se sproži, ko je prenesenih vseh štirinajst bajtov (Slika 4.36).

```
DMA_InitStruct.DMA_Channel = DMA_Channel_4; // Channel 4(UART4 rx)
DMA_InitStruct.DMA_DIR = DMA_DIR_PeripheralToMemory; // Transfer from uart4 to buffer
DMA_InitStruct.DMA_PeripheralBaseAddr = (uint32_t)&(UART4->DR); // Uart4 data register
DMA_InitStruct.DMA_Memory0BaseAddr = (uint32_t)&tagValues; // Compy data to this buffer
DMA_InitStruct.DMA_PeripheralInc = DMA_PeripheralInc_Disable; // Don't increment the peripheral 'memory
DMA_InitStruct.DMA_MemoryInc = DMA_MemoryInc_Enable; // Increment the memory location
DMA_InitStruct.DMA_PeripheralDataSize = DMA_PeripheralDataSize_Byte; // 8b size transfer
DMA_InitStruct.DMA_MemoryDataSize = DMA_MemoryDataSize_Byte; // 8b size transfer
DMA_InitStruct.DMA_Mode = DMA_Mode_Normal; // Normal mode. Not repeating
DMA_InitStruct.DMA_BufferSize = 14; // We transfer 14 bytes
DMA_InitStruct.DMA_Priority = DMA_Priority_High; // High priority... ???
DMA_InitStruct.DMA_MemoryBurst = DMA_MemoryBurst_Single; // No bursts
DMA_InitStruct.DMA_PeripheralBurst = DMA_PeripheralBurst_Single; // no bursts
DMA_InitStruct.DMA_FIFOMode = DMA_FIFOMode_Disable; // FIFO mode disabled(direct mode)
DMA_Init(DMA1_Stream2, &DMA_InitStruct);

DMA_ITConfig(DMA1_Stream2, DMA_IT_TC, ENABLE); // Enable the transfer complete interrupt for DMA1 stream 2

USART_DMACmd(UART4, USART_DMAReq_Rx, ENABLE); // Enable Uart dma rx request

NVIC_InitStructure.NVIC_IRQChannel = DMA1_Stream2_IRQn;
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0x00;
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0x00;
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;

NVIC_Init(&NVIC_InitStructure);
```

Slika 4.36: Nastavitev krmilnika DMA1

4.2.5 Nastavitev senzorjev

Alarmni sistem uporablja več vrst senzorjev, ki jih je bilo treba pravilno nastaviti. Za vsak senzor je bilo treba najprej vklopiti uro. Za magnetna stikala je bilo treba vklopiti uro na splošnonamenskih vratih GPIOB in GPIOD. Pri senzorjih za zaznavanje gibanja je bilo treba vklopiti uro na splošnonamenskih vratih GPIOB in GPIOA. Senzor plinov potrebuje uro na splošnonamenskih vratih GPIOE, medtem ko senzorji loma stekla potrebujejo uro na splošnonamenskih vratih GPIOA. Uro smo omogočili z uporabo funkcije RCC_AHB1PeriphClockCmd (Slika 4.37).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA , ENABLE); // Lcd, Pir
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOB, ENABLE); // Touch, Reed, Pir
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE); // Touch, Ethernet, LCD
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE, ENABLE); // Lcd
```

Slika 4.37: Vklop ure na splošnonamenskih vratih

Nato je bilo treba nastaviti pine vsakega senzorja posebej. Vsi pini so nastavljeni kot vhod (Slika 4.38).

```
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2; // Reed 1
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IN;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_DOWN;
GPIO_Init(GPIOB,&GPIO_InitStructure);
```

Slika 4.38: Primer nastavitve pina za magnetno stikalo

4.2.6 Nastavitev GSM modula

Tako kot vse prejšnje komponente je bilo treba nastaviti tudi GSM modul. Ker GSM modul za komunikacijo z mikrokrmilnikom uporablja UART vmesnik, je bilo najprej treba omogočiti uro za splošnonamenska vrata GPIOC in vmesnik USART6. To smo storili s klicem funkcij RCC_AHB1PeriphClockCmd in RCC_APB2PeriphClockCmd (Slika 4.39).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE); // Enable clock for GPIOC...
RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART6, ENABLE); // Enable USART6 clock
```

Slika 4.39: Vklop ure za splošnonamenska vrata GPIOC in vmesnik USART6

Po vklopu ure je bilo treba nastaviti pin za pošiljanje. Pin smo povezali z vmesnikom USART6, ki smo ga nastavili tako, da deluje s hitrostjo 9600 bps. Poleg tega smo nastavili ostale parametre tako, da se sporočila pošiljajo v formatu 8N1, kar pomeni, da pošiljamo osem podatkovnih bitov in en stop bit, ne pošiljamo pa paritetnega bita (Slika 4.40).

```
// Init GPIO

GPIO_InitStruct.GPIO_Pin = GPIO_Pin_6; // tx
GPIO_InitStruct.GPIO_Mode = GPIO_Mode_AF; // Connect pins with usart interface
GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStruct.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_InitStruct.GPIO_OType = GPIO_OType_PP;
GPIO_Init(GPIOC, &GPIO_InitStruct);

// Connect GPIO pins with USART interface

GPIO_PinAFConfig(GPIOC, GPIO_PinSource6, GPIO_AF_USART6);

// init USART

USART_InitStruct.USART_BaudRate = 9600; // Baudrate
USART_InitStruct.USART_HardwareFlowControl = USART_HardwareFlowControl_None; //
USART_InitStruct.USART_Mode = USART_Mode_Tx;
USART_InitStruct.USART_Parity = USART_Parity_No; // Without parity bit
USART_InitStruct.USART_StopBits = USART_StopBits_1; // One stop bit
USART_InitStruct.USART_WordLength = USART_WordLength_8b; // Data length = 8 bits
USART_Init(USART6, &USART_InitStruct);
USART_Cmd(USART6, ENABLE); // Enable USART6
```

Slika 4.40: Nastavitev pina in vmesnika USART6

Po končani nastavitvi pina in vmesnika UART6 je treba GSM modulu poslati tudi AT ukaz, s katerim modul postavimo v način za pošiljanje SMS-ov. To storimo tako, da preko vmesnika UART6 pošljemo niz "AT+CMGF=1"[27].

Podatke pošiljamo s pomočjo funkcije sendData, s pomočjo funkcije sendSms pa pošljemo SMS sporočilo na izbrano številko.

4.2.7 Nastavitev ethernet modula

Namen ethernet modula je pošiljanje podatkov nadzornemu programu. Če želimo, da modul pravilno deluje, ga je treba pravilno nastaviti. Najprej je bilo treba vklopiti uro na splošnonamenskih vratih GPIOA, GPIOC in GPIOD ter na vmesniku UART5. To smo storili s klicem funkcij RCC_AHB1PeriphClockCmd in RCC_APB1PeriphClockCmd (Slika 4.41).

```
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOA , ENABLE); // Lcd, Pir
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOC, ENABLE); // Enable clock for GPIOC
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOD, ENABLE); // Touch, Ethernet, LCD
RCC_APB1PeriphClockCmd(RCC_APB1Periph_UART5, ENABLE); // Enable UART5 clock
```

Slika 4.41: Vklop ure na splošnonamenskih vratih in vmesniku UART5

Po vklopu ure je bilo treba nastaviti vse potrebne pine. Pine za sprejemanje in pošiljanje podatkov smo povezali z vmesnikom UART5, nastavitveni pin pa smo nastavili kot izhod. Vmesnik UART5 smo nastavili tako, da deluje s hitrostjo 9600 bps. Poleg tega smo druge parametre nastavili tako, da se sporočila pošiljajo v formatu 8N1, kar pomeni, da pošiljamo osem podatkovnih bitov in en stop bit, ne pošiljamo pa paritetnega bita (Slika 4.42).

```
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_12; // Pin for TX
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_Init(GPIOC,&GPIO_InitStructure);

GPIO_PinAFConfig(GPIOC,GPIO_PinSource12,GPIO_AF_UART5);

GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2; // Pin for RX
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_UP;
GPIO_Init(GPIOD,&GPIO_InitStructure);

GPIO_PinAFConfig(GPIOD,GPIO_PinSource2,GPIO_AF_UART5);

// Uart config

USART_InitStructure.USART_BaudRate = 9600;
USART_InitStructure.USART_WordLength = USART_WordLength_8b;
USART_InitStructure.USART_StopBits = USART_StopBits_1;
USART_InitStructure.USART_Parity = USART_Parity_No;
USART_InitStructure.USART_HardwareFlowControl = USART_HardwareFlowControl_None;
USART_InitStructure.USART_Mode = USART_Mode_Rx | USART_Mode_Tx;
USART_Init(UART5, &USART_InitStructure);

USART_Cmd(UART5, ENABLE); // Enable UART


// Config cfg pin

GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2; // Cfg pin.
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;;

GPIO_Init(GPIOA,&GPIO_InitStructure);
```

Slika 4.42: Nastavitev pinov in vmesnika UART5

Po končani nastavitvi pinov in vmesnika UART5 je bilo treba ethernet modulu nastaviti vse potrebne omrežne parametre. Tako je bilo treba modulu nastaviti IP (ang. *Internet Protocol*) naslov, vrata (ang. *port*), privzeti prehod (ang. *gateway*), način delovanja (TCP client), IP serverja in vrata serverja. To smo storili tako, da smo nastavitveni pin postavili v nizko sta-

nje, nato pa počakali, da nam je modul preko vmesnika UART5 poslal znak 'U'. Po prejetem znaku smo začeli modulu preko vmesnika UART5 pošiljati vse zgoraj omenjene parametre (Slika 4.43). Če je bila nastavitev uspešna, je modul poslal znak 'K', v nasprotnem primeru pa znak 'E'. Nastavitev smo končali tako, da smo nastavitveni pin postavili nazaj v visoko stanje.

```
/*
    prefix(0x55,0xaa)
    dest ip(0x04,0x02,0xA8,0xC0)
    dest port(0x2A,0x50)
    module ip(0x4D,0x02,0xA8,0xC0)
    module port(0x8C,0x4E)
    gateway(0x01,0x02,0xA8,0xC0)
    work mode(0x01)
    baud rate(0x80,0x25,0x00)
    reserved( i put 0 in here.. 0x00)
    checksum(calculated during configuration mode... See below)
*/
```

Slika 4.43: Parametri za nastavitev ethernet modula

Podatke modulu pošiljamo s pomočjo funkcije send. Funkcija authenticate se uporablja za avtentikacijo uporabnika z RFID značko in PIN številko. Funkcija podatke zakriptira s kriptirnim algoritmom AES in jih pošlje nadzornemu programu. Prejete podatke funkcija odkriptira in preveri, ali je bila avtentikacija uspešna. Funkcija getNumSendSms se uporablja za pošiljanje SMS sporočil v primeru vdora v objekt. Telefonske številke, pridobljene od nadzornega programa, funkcija odkriptira s pomočjo kriptirnega algoritma AES in s pomočjo funkcije sendSms pošlje SMS sporočilo.

Kriptirni algoritem AES

Kriptirni algoritem AES deluje na principu substitucijsko-permutacijskega omrežja in je hiter tako v programski kot strojni izvedbi. AES izvaja operacije na bloku podatkov velikosti 128 bitov, uporablja pa lahko 128-, 192- in 256-bitni ključ [33].

Dolžina ključa pri kriptiranju pomeni tudi določeno število ponovitev transformacijskih ciklov, ki pretvorijo nekriptiran tekst v kriptirano obliko. Število ciklov se določi na naslednji način [33]:

- 10 ciklov za ključ dolžine 128 bitov,
- 12 ciklov za ključ dolžine 192 bitov,
- 14 ciklov za ključ dolžine 256 bitov.

Vsak cikel je sestavljen iz več korakov, od katerih vsak vsebuje štiri podobne, vendar različne faze. Za pridobitev originalnega teksta je treba cikle ponoviti v obratnem vrstnem redu, z uporabo enakega ključa, s katerim je tekst zakriptiran [33].

Algoritem AES je sestavljen iz več faz. V prvi fazi iz glavnega ključa izpeljemo podključe, ki jih algoritem uporabi v posameznih transformacijskih ciklih. V drugi fazi, ki se imenuje začetna faza, je treba za vsak bajt 128-bitnega podatkovnega bloka opraviti operacijo XOR (ang. *Exclusive OR*) s podključem. V tretji fazi, imenovani ciklična faza, je najprej treba opraviti substitucijo, s pomočjo katere vsak bajt 128-bitnega podatkovnega bloka zamenjamo glede na preslikovalno tabelo. Nato je treba opraviti transpozicijo, s pomočjo katere naredimo ciklični zamik v zadnjih treh vrsticah 128-bitnega podatkovnega bloka. Nato sledi premešavanje stolpcev, pri katerem vsebino posameznega stolpca pomnožimo s posebno matriko. Na koncu je treba za vsak bajt 128-bitnega podatkovnega bloka opraviti operacijo XOR s podključem. Ciklično fazo je treba ponoviti večkrat zaporedoma. Število ponovitev je odvisno od dolžine AES ključa. Nato sledi četrta faza, imenovana zadnja faza, ki deluje enako kot ciklična faza, vendar s to razliko, da ne vključuje premešavanja stolpcev in večkratnega ponavljanja [33].

Ko algoritem konča vse zgoraj opisane faze, na izhodu dobimo zakriptiran tekst.

4.2.8 FreeRTOS in opravila

FreeRTOS je majhen in enostaven operacijski sistem za vgrajene sisteme, ki daje prednost kompaktnosti in hitremu izvajanju. FreeRTOS nima implementiranih nobenih naprednih lastnosti, kot so gonilniki naprav, napredno upravljanje pomnilnika, uporabniški računi in mreženje, ki jih srečamo v operacijskih sistemih, kot sta Linux in Windows [34].
Operacijski sistem FreeRTOS sestavljajo naslednji pomembni elementi [34]:

- opravila,
- ključavnice,
- semaforji,
- programski časovniki.

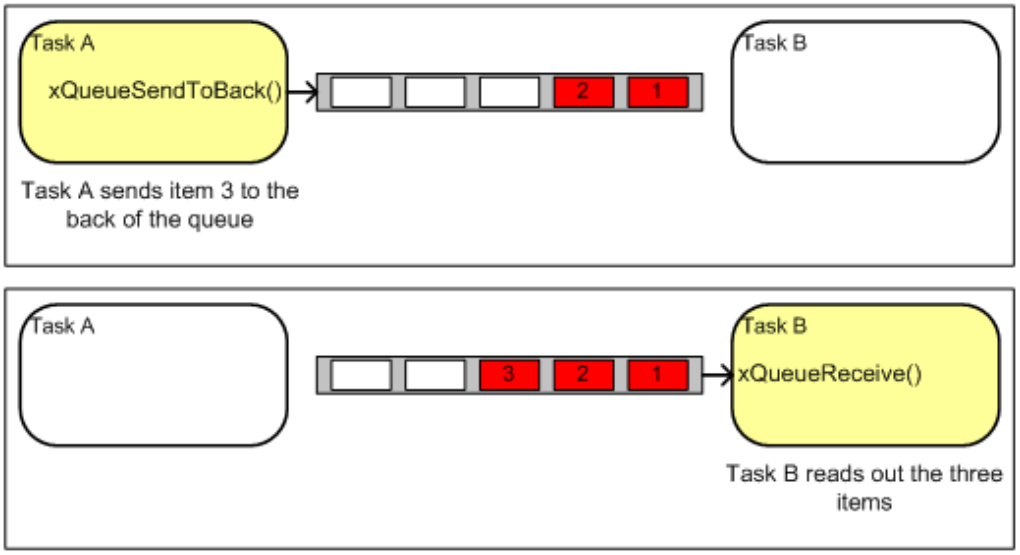
FreeRTOS omogoča izvajanje več opravil s pomočjo preklapljanja med njimi. Preklapljanje je implementirano s pomočjo prekinitev, ki se dogajajo v določenih intervalih. Izbira opravila za izvajanje temelji na algoritmu Round Robin in prioritetah [34].

Opravilo ustvarimo s pomočjo funkcije xTaskCreate, ki ustvari novo opravilo in ga doda na seznam opravil, ki so pripravljena za izvajanje. Funkcija sprejme večje število argumentov, kot so na primer kazalec na funkcijo (opravilo), ime opravila, velikost sklada in prioriteta opravila [35].

Ko so ustvarjena vsa opravila, je treba zagnati razporejevalnik opravil. Tega zaženemo tako, da pokličemo funkcijo vTaskStartScheduler. Po klicu funkcije ima jedro operacijskega sistema nadzor nad tem, katero opravilo se bo naslednje izvedlo [35].

Komunikacija med opravili poteka s pomočjo vrst, ki so primarna oblika medopravilne komunikacije v operacijskem sistemu FreeRTOS (Slika 4.44). S pomočjo vrst pošiljamo sporočila med opravili, dopuščajo pa tudi komunikacijo med opravili in prekinitvami. V večini primerov se vrste uporabljajo po principu FIFO (ang. *First In First Out*), kar pomeni, da se podatki zapišejo na konec vrste. Poleg tega je mogoče podatke zapisati tudi na začetek vrste.

Vrsto ustvarimo s pomočjo funkcije xQueueCreate, v njo pa pišemo s pomočjo funkcij xQueueSendToBack, xQueueSendToFront, xQueueSendToBackFromISR in xQueueSendToFrontFromISR. Branje podatkov iz vrste poteka s pomočjo funkcij xQueueReceive in xQueueReceiveFromISR [35].



Slika 4.44: Pisanje v vrsto in branje iz nje, vir: [34]

Opis opravil alarmnega sistema

Program alarmnega sistema smo razdelili na več opravil, ki med seboj komunicirajo s pomočjo vrst. Opravila so sledeča:

- autoSendStatus,
- alarmSysControl,
- alarmExpired,
- reedSensor1,
- reedSensor2,
- pirSensor1,

- pirSensor2,
- glassSensor1,
- glassSensor2,
- gasSensor.

Opravila reedSensor1, reedSensor2, pirSensor1, pirSensor2, glassSensor1, glassSensor2 in gasSensor so namenjena spremljanju stanja vseh senzorjev, ki so priklopljeni na alarmni sistem (Slika 4.45). Če je alarmni sistem v aktiviranem stanju in če se izvaja opravilo, katerega senzor je zaznal nenavaden dogodek, potem se v vrsto triggeredSensor zapiše številka tega senzorja.

```
void reedSensor1(void *pvParameters){

  uint8_t alarmType = 8;
  uint8_t pit = 0;

  while(1){

    if(xQueuePeek(triggeredSensor, &pit, 0) == pdFALSE &&
       xQueuePeek(alarmTaskEnable, &pit, 0) == pdTRUE && getReed1Value() == 1)
    {

      xQueueSend(triggeredSensor, (void*)&alarmType, 10);

      enableTim5(30000); // 15 seconds

    }

  }

}
```

Slika 4.45: Primer senzorskega opravila

Opravilo autoSendStatus (Slika 4.46) je namenjeno periodičnemu sporočanju stanja alarmnega sistema centralnemu nadzornemu programu. Opravilo pošilja informacijo o tem, kateri senzor je sprožil alarm. Poleg tega pa pošilja tudi informacijo o tem, ali je alarmni sistem v aktiviranem ali deaktiviranem stanju. Opravilo dobi informacije o stanju alarmnega sistema

z branjem vrst autoSendStatusEnable in alarmStatus. Opravilo je uporabno zato, ker redno osvežuje podatke na grafičnem vmesniku centralnega nadzornega programa. Če pride do prekinitve omrežne povezave med alarmnim sistemom in centralnim nadzornim programom ter se medtem sproži alarm, se ob ponovni vzpostavitvi povezave med alarmnim sistemom in centralnim nadzornim programom stanje grafičnega vmesnika osveži.

```
while(1){

  // Check if alarm system is activated(autoSendStatusEnable queue)

  if(xQueuePeek(autoSendStatusEnable, &pit, 0) == pdTRUE)
  {

    // Send message for activated

    send(0, 0, 24); // 24... Alarm activate status message

  }
  else
  {

    // Send message for deactivated

    send(0, 0, 25); // 25... Alarm deactivate status message

  }

  // Check if alarm is triggered

  if(xQueuePeek(autoSendStatusEnable, &pit, 0) == pdTRUE &&
     xQueuePeek(alarmStatus, &pit, 0) == pdTRUE)
  {

    // Save alarm type into var

    xQueuePeek(alarmStatus, &alarmType, 0);

    // Send message for alarm to the server... Offset is 10

    send(0, 0, alarmType); // Offset 10... Check message types

  }
  else
  {

    // Send message for alarm Off

    send(0, 0, 23);

  }


  vTaskDelay(10000);

}
```

Slika 4.46: Opravilo autoSendStatus

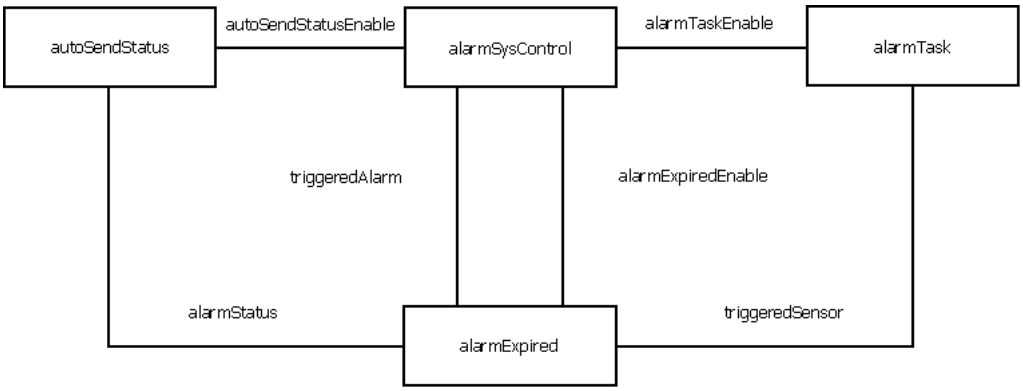
Poleg že omenjenih opravil smo ustvarili tudi opravilo alarmSysControl, ki skrbi za vklop in izklop alarmnega sistema ter izklop sproženega alarma. Opravilo deluje tako, da preverja vrednost spremenljivke tagRead. Če ugotovi, da je vrednost spremenljivke enaka ena, potem je uporabnik svojo RFID značko približal čitalcu, ki jo je prebral. V primeru, da je RFID značka prebrana, se izvedeta funkciji enterPin in authenticate, s pomočjo katerih uporabnik vnese PIN številko in se avtenticira (Slika 4.47). Če je avtentikacija uspešna, se najprej preveri stanje vrste triggeredAlarm, da se ugotovi, ali je alarm sprožen. Če je alarm sprožen, se alarm izklopi in deaktivira alarmni sistem. V nasprotnem primeru se alarmni sistem samo aktivira oziroma deaktivira. Ali je alarmni sistem aktiviran ali deaktiviran, se ugotovi s pomočjo stanja vrste autoSendStatusEnable.

```
while(1){

  if(tagRead == 1)
  {

    vTaskSuspendAll();

    // Check if auth is ok

    if(enterPin(KX1, KX2, KX3, KY1, KY2, KY3) == 1 && authenticate() == 1)
    {
```

Slika 4.47: Preverjanje spremenljivke tagRead in avtentikacija

Opravilo alarmExpired je namenjeno ugotavljanju, ali je potekel časovnik, ki ga sprožijo senzorska opravila, ko senzorji zaznajo nenavaden dogodek. Če je časovnik potekel, opravilo alarmExpired preko vrste triggeredAlarm opravilu alarmSysControl sporoči, da je alarm sprožen. Opravilo alarmExpired nato opravilu autoSendStatus sporoči, kateri senzor je sprožil alarm. To izvede preko vste alarmStatus. V primeru sprožitve alarma opravilo alarmExpired pošlje tudi SMS sporočila uporabnikom objekta, da je prišlo do morebitnega vdora.

Slika 4.48 prikazuje vsa opravila in vrste, ki so bile uporabljene za komunikacijo med opravili.



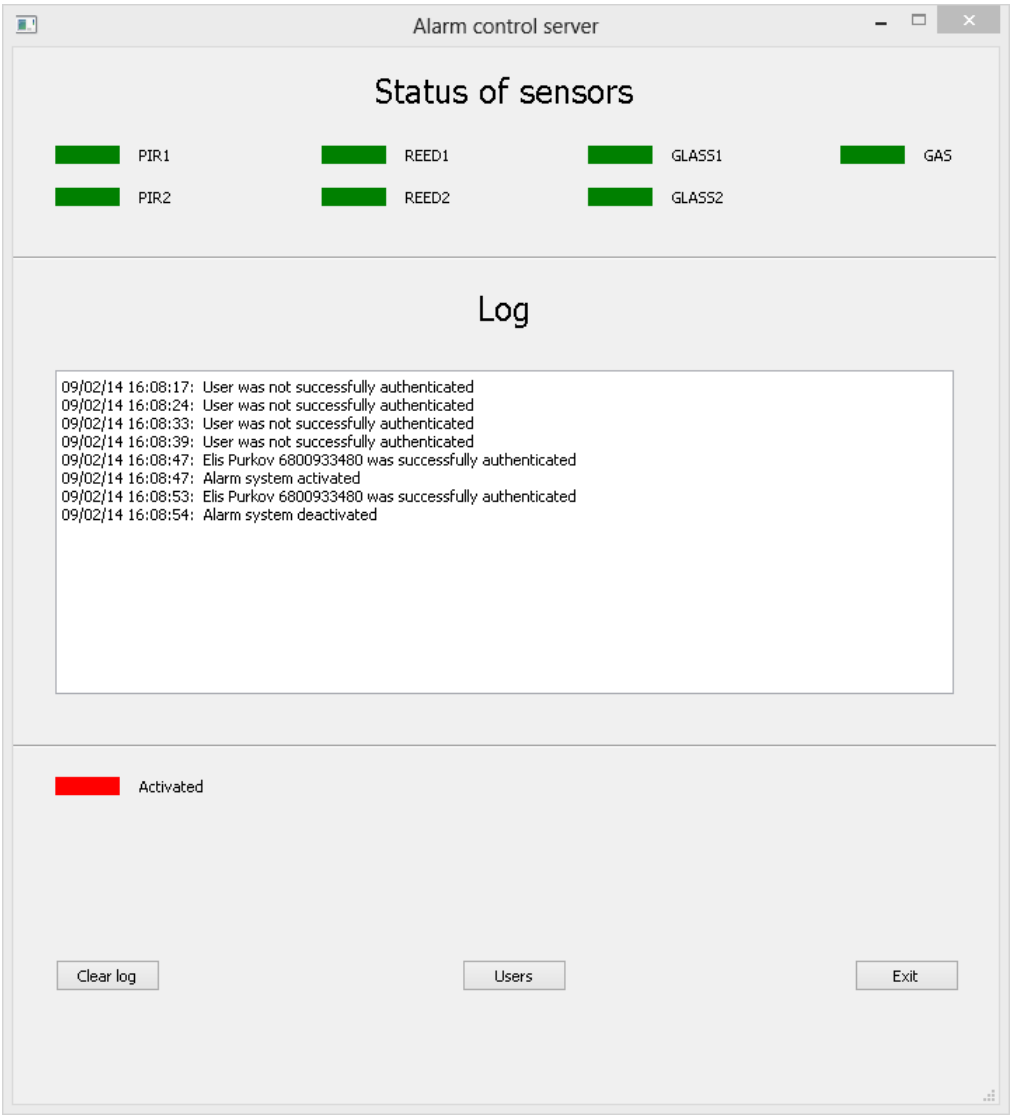
Slika 4.48: Diagram opravil in vrst

4.2.9 Nadzorni program

Poleg alarmnega sistema je bilo treba izdelati nadzorni program, ki prikazuje stanje alarmnega sistema. Program je bil napisan v programskem jeziku Python [36], za prikaz grafičnih elementov pa smo uporabili knjižnico gradnikov, imenovano Qt [37]. Nadzorni program prikazuje stanje vseh senzorjev, ki so priklopljeni na alarmni sistem. Poleg stanja senzorjev je mogoče spremljati tudi, ali je alarmni sistem v aktiviranem ali neaktiviranem stanju. Nadzorni program omogoča tudi spremljanje podatkov o preteklih uspešnih oziroma neuspešnih avtentikacijah uporabnikov. Poleg spremljanja avtentikacij program omogoča tudi pregled nad tem, kdaj je bil alarmni sistem aktiviran oziroma deaktiviran.

Nadzorni program omogoča tudi upravljanje z uporabniki, ki imajo dovoljenje za interakcijo z alarmnim sistemom. Za interakcijo s sistemom potrebujejo pripadajočo RFID značko in PIN številko. Program omogoča dodajanje in brisanje pa tudi spreminjanje podatkov obstoječih uporabnikov.

Na Sliki 4.49 je prikazan grafični vmesnik, ki se izriše ob zagonu nadzornega programa.



Slika 4.49: Grafični vmesnik nadzornega programa

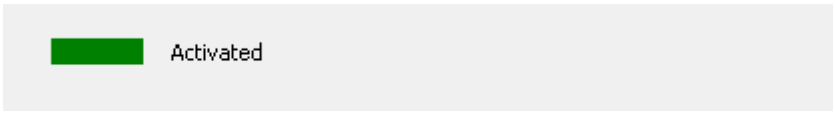
Vrstica, ki prikazuje stanje senzorjev, je razvidna s Slike 4.50. Če je kvadratek zelene barve, senzor ni zaznal nobenega nenavadnega dogodka. Če senzor zazna nenavaden dogodek, se kvadratek obarva rdeče (Slika 4.50).

Slika 4.50: Vrstica prikazuje stanje senzorjev

Dnevniško polje prikazuje Slika 4.51. Vanj se zapisujejo dogodki, povezani z alarmnim sistemom, kot so uporabniške avtentikacije, aktivacije in deaktivacije alarmnega sistema ter izklopi alarma.

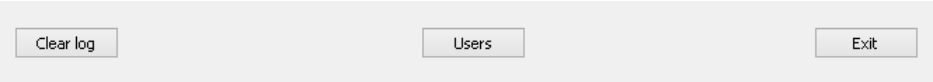
Slika 4.51: Dnevniško polje

Poleg dnevniškega polja ter vrstice, ki prikazuje stanje senzorjev, je vrstica, ki prikazuje, ali je alarmni sistem aktiviran ali deaktiviran. Če je kvadratek obarvan zeleno, je alarmni sistem aktiviran, rdeče obarvan kvadratek pa pomeni, da je alarmni sistem deaktiviran (Slika 4.52).



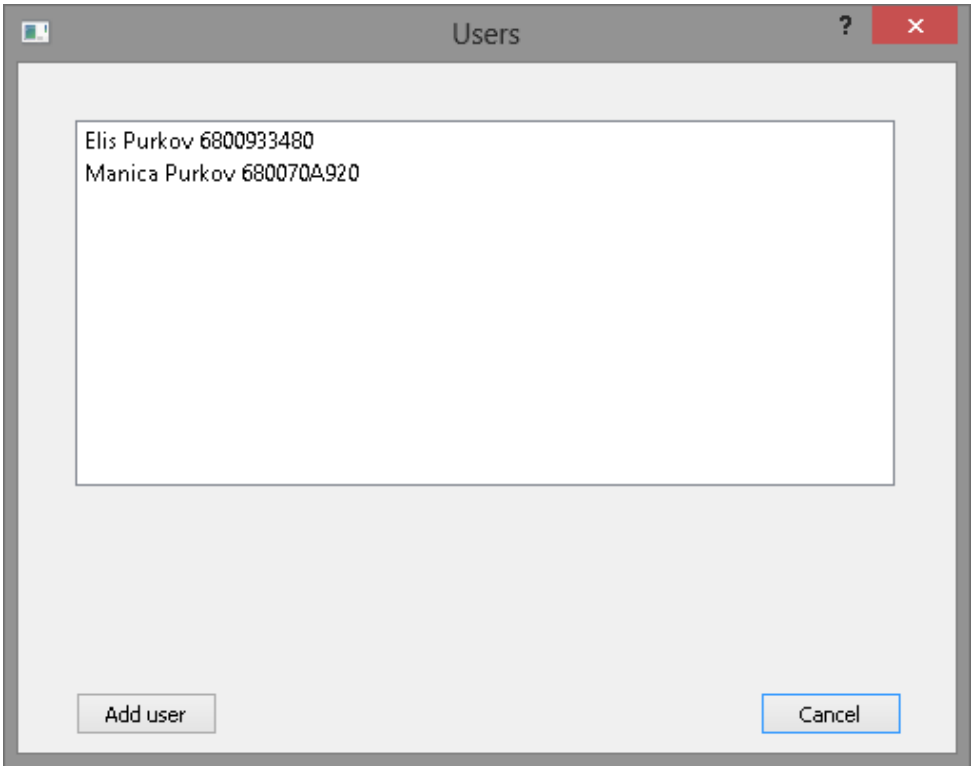
Slika 4.52: Vrstica prikazuje stanje alarmnega sistema

Na koncu grafičnega vmesnika imamo tri gumbe, katerih namen je upravljanje uporabnikov, izbris dnevniškega polja ter izklop programa. Za izbris dnevniškega polja pritisnemo gumb, imenovan Clear log. Z gumbom Users odpremo okno, s pomočjo katerega upravljamo uporabnike sistema. Gumb Exit je namenjen izhodu iz programa (Slika 4.53).



Slika 4.53: Gumbi za upravljanje uporabnikov in dnevnika ter izhod iz programa

Za upravljanje uporabnikov je treba pritisniti gumb Users. Ob pritisku gumba se odpre novo okno, v katerem se izpišejo vsi uporabniki sistema. Okno ima tudi dva gumba, ki sta namenjena dodajanju novih uporabnikov ter zapiranju okna. Okno za upravljanje uporabnikov je razvidno s Slike 4.54.

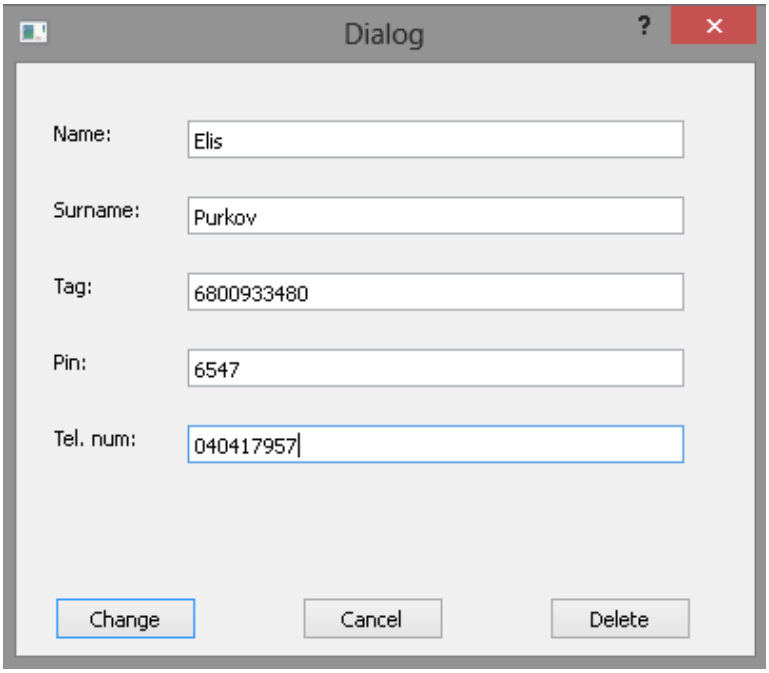


Slika 4.54: Okno za upravljanje uporabnikov

Uporabnik programa lahko doda novega uporabnika tako, da s klikom na gumb Add user odpre obrazec, ki ga nato izpolni. Vnesti je treba ime, priimek, številko RFID značke, PIN številko ter telefonsko številko. Obrazec za dodajanje uporabnika je razviden s Slike 4.55.

Slika 4.55: Obrazec za dodajanje novega uporabnika

Poleg dodajanja novega uporabnika program omogoča tudi urejanje in brisanje obstoječih uporabnikov. Z dvojnim klikom na uporabnika se odpre okno za urejanje tega uporabnika. Spreminjati je mogoče ime, priimek, številko RFID značke, PIN številko ter telefonsko številko. S klikom na gumb Delete uporabnika izbrišemo. Obrazec za urejanje in brisanje obstoječega uporabnika je razviden s Slike 4.56.



Slika 4.56: Obrazec za urejanje in brisanje obstoječega uporabnika

Poglavje 5

Zaključek

Cilj diplomskega dela je bil izdelati alarmni sistem za učinkovito varovanje stanovanjskih objektov in hiš. Alarmni sistem, ki smo ga razvili, ima to prednost, da je poceni, hkrati pa ponuja nadzorni program, primerljiv s tistimi v alarmnih sistemih višjega cenovnega razreda. Podobni alarmni sistemi v istem cenovnem razredu ne omogočajo neomejenega števila uporabnikov, za interakcijo s sistemom pa uporabljajo navadno številčnico, medtem ko naš sistem omogoča interakcijo preko zaslona na dotik.

Poleg prednosti ima naš alarmni sistem tudi nekatere slabosti. Ena izmed glavnih je ta, da ne deluje brez povezave z nadzornim programom. Težavo bi lahko rešili z vzpostavitvijo neposredne povezave med alarmnim sistemom in nadzornim programom, vendar bi bilo to smiselno samo v primeru, da bi bila nadzorna soba zelo blizu varovanega objekta. Poleg tega nadzorni program ne omogoča povezave z večjim številom alarmnih sistemov, kar pomeni, da vsak alarmni sistem potrebuje svoj nadzorni program. Težavo bi lahko rešili s spremembo nadzornega programa, ki bi mu omogočila sprejemanje večjega števila povezav. To bi storili z uporabo niti, od katerih bi vsaka komunicirala z enim alarmnim sistemom.

Pri izdelavi alarmnega sistema smo naleteli tudi na nekatere težave. Ena izmed največjih je bila dobava sestavnih delov. Senzorje in druge komponente smo namreč naročili preko spletne trgovine Ebay, naročenega pa nismo dobili.

Zato je bilo treba nekatere komponente ponovno naročiti, kar je podaljšalo čas izdelave alarmnega sistema.

Zastavljeni cilj diplomskega dela smo uspešno dosegli, izdelek pa svoje delo opravlja zadovoljivo. Če bi želeli izdelek prodajati na trgu, bi bilo treba odpraviti vse zgoraj omenjene slabosti, poleg tega pa izdelati tudi primerno ohišje.

Slike

Literatura

[1] (2014) Protivlomni sistemi. Dostopno na:
http://www.varendom.com/alarmni-sistemi.html

[2] (2014) Zaslon na dotik. Dostopno na:
http://sl.wikipedia.org/wiki/Zaslon_na_dotik

[3] (2014) Touchscreen. Dostopno na:
http://en.wikipedia.org/wiki/Touchscreen

[4] (2014) Touchscreens or Human Machine Interface (HMI). Dostopno na:
http://www.engineersgarage.com/articles/touchscreen-technology-working

[5] (2014) 4-Wire Analog-Resistive Touch Screens. Dostopno na:
https://www.sparkfun.com/datasheets/LCD/HOW%20DOES%20IT%20WORK.pdf

[6] (2014) Voltage divider. Dostopno na:
http://en.wikipedia.org/wiki/Voltage_divider

[7] (2014) Radiofrekvenčna identifikacija. Dostopno na:
http://sl.wikipedia.org/wiki/Radiofrekven%C4%8Dna_identifikacija

[8] (2014) Radiofrekvenčna identifikacija. Dostopno na:
http://www.leoss.si/index.php?vie=ctl&gr1=strSvt&gr2=&id=2006102514103146

[9] (2014) RFID basics. Dostopno na:
http://www.priority1design.com.au/rfid_design.html

[10] (2014) Passive RFID Basics. Dostopno na:
http://ww1.microchip.com/downloads/en/devicedoc/51115f.pdf

[11] (2014) STM32F407VG. Dostopno na:
http://www.st.com/web/catalog/mmc/FM141/SC1169/SS1577/LN11/PF252140

[12] (2014) STM32F4DISCOVERY. Dostopno na:
http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419

[13] (2014) ARM Cortex-M4 Processor Technical Reference Manual. Dostopno na:
http://infocenter.arm.com/help/topic/com.arm.doc.dui0553a/index.html

[14] (2014) STM32F3 and STM32F4 Series Cortex®-M4 programming manual. Dostopno na:
http://www.st.com/st-web-ui/static/active/en/resource/technical/document/programming_manual/DM00046982.pdf

[15] (2014) Advanced Microcontroller Bus Architecture (AMBA). Dostopno na:
http://www.icverification.com/BusProtocols/AmbaOverview.php

[16] (2014) AMBA Advanced Peripheral Bus. Dostopno na:
http://www.icverification.com/BusProtocols/AmbaAPB.php

[17] (2014) AMBA Advanced High-performance Bus (AHB). Dostopno na:
http://www.icverification.com/BusProtocols/AmbaAHB.php

[18] (2014) RDM630 Specification. Dostopno na:
http://www.seeedstudio.com/depot/datasheet/RDM630-Spec..pdf

[19] (2014) SSD1289. Dostopno na:
http://www.datasheetdir.com/SSD1289+download

[20] (2014) Touch screen controller. Dostopno na:
http://www.ti.com/lit/ds/sbas090b/sbas090b.pdf

[21] (2014) How PIRs Work. Dostopno na:
https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/how-pirs-work

[22] (2014) Reed Switch Info. Dostopno na:
http://reed-switch-info.com

[23] (2014) Glass break detector. Dostopno na:
http://en.wikipedia.org/wiki/Glass_break_detector

[24] (2014) Gas detector. Dostopno na:
http://en.wikipedia.org/wiki/Gas_detector

[25] (2014) Serial to Ethernet converter module. Dostopno na:
http://www.tcp232.net/download/USR-TCP232-T-EN.pdf

[26] (2014) TC35 / TC37 Hardware Interface Description. Dostopno na:
http://www.robotshop.com/media/files/pdf/datasheet-gsm-tc35.pdf

[27] (2014) AT Command Set Siemens Cellular Engines. Dostopno na:
http://tc485.od.ua/docs/at_commands_tc35.pdf

[28] (2014) IAR Embedded Workbench, dostopno na:
http://www.iar.com/Products/IAR-Embedded-Workbench/

[29] (2014) Discovery kit for STM32F407/417 lines - with STM32F407VG MCU. Dostopno na:
http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419

[30] (2014) Calibration in touch-screen systems. Dostopno na:
http://www.ti.com/lit/an/slyt277/slyt277.pdf

[31] (2014) Direct memory access. Dostopno na:
http://en.wikipedia.org/wiki/Direct_memory_access

[32] (2014) Using the STM32F2 and STM32F4 DMA controller. Dostopno na:
http://www.st.com/web/en/resource/technical/document/application_note/DM00046011.pdf

[33] (2014) Advanced Encryption Standard. Dostopno na:
http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[34] (2014) FreeRTOS. Dostopno na:
http://en.wikipedia.org/wiki/FreeRTOS

[35] (2014) About FreeRTOS. Dostopno na:
http://www.freertos.org/RTOS.html

[36] (2014) Python. Dostopno na:
https://www.python.org/

[37] (2014) Qt. Dostopno na:
http://qt-project.org/