

UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Matej Božič

Vremenska postaja na platformi STM32F4

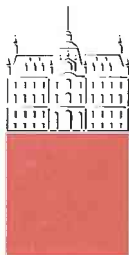
DIPLOMSKO DELO

VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM
PRVE STOPNJE RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: izr. prof. dr. Patricio Bulić

Ljubljana, 2013

Rezultati diplomskega dela so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavljane ali izkoriščanje rezultatov diplomskega dela je potrebno pisno soglasje avtorja, Fakultete za računalništvo in informatiko ter mentorja.



Št. naloge: 00401/2013

Datum: 03.04.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **MATEJ BROŽIČ**

Naslov: **OPTIČNO BRANJE UNIVERZALNEGA PLAČILNEGA NALOGA**
THE UNIVERSAL PAYMENT ORDER OPTICAL READING

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Ročno vnašanje besedila iz univerzalnega plačilnega naloga je zamudno početje. Zaradi tega se je v bankah pojavila potreba po avtomatizaciji tega postopka. V ta namen so bili razviti posebni čitalniki, katere najdemo nameščene na različnih krajih od bank do trgovskih centrov.

Kandidat naj pregleda možnost avtomatizacije zajema vsebine univerzalnega plačilnega naloga s pomočjo metod računalniškega vida. Izdela naj sistem optičnega branja plačilnega naloga in na ta način omogoči tudi uporabnikom, ki nimajo čitalnikov, avtomatsko vnašanja plačilnega naloga v sistem zgolj s pomočjo zajete slike.

Mentor:

viš. pred. dr. Borut Batagelj

Dekan:

prof. dr. Nikolaj Zimic



IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani/-a **Matej Božič**

z vpisno številko **63030122**

sem avtor/-ica diplomskega dela z naslovom:

Vremenska postaja na platformi STM32F4

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal/-a samostojno pod mentorstvom (naziv, ime in priimek) izr. 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 v zbirki »Dela FRI«.

V Ljubljani, dne 14. oktober 2013

Podpis avtorja/-ice: _____

Iskreno se zahvaljujem staršem za podporo v času študija. Za pomoč pri izdelavi diplomskega dela se zahvaljujem vsem, ki so mi pomagali, še posebna zahvala pa gre mentorju izr. prof. dr. Patriciu Buliću za potrpežljivost, nasvete ter strokovno pomoč pri pisanju.

Kazalo vsebine

Povzetek	1
Abstract	3
1 Uvod	5
2 Uporabljena strojna oprema	7
2.1 STM32F4 Discovery	7
2.1.1 Vodili AHB ter APB	8
2.1.2 Prekinitveni vektor NVIC	9
2.2 STM32F4DIS-BB ter STM32F4DIS-LCD	9
2.2.1 Krmilnik SSD2119	11
2.2.2 Krmilnik STMPE811	11
2.3 Modul za brezžično komunikacijo RFM12B	11
2.4 Senzor temperature in zračne vlage DHT11	13
2.5 Senzor temperature in zračnega pritiska BMP085	13
2.6 Anemometer ter Hall senzor	14
2.7 Merilnik smeri vetra	15
3 Programska realizacija	17
3.1 Časovniki in prekinitve	19
3.2 DHT11 komunikacija	21
3.3 BMP085 in protokol I ² C	22
3.4 Smer vetra ter krmilnik ADC	23
3.5 Hitrost vetra	24
3.6 LCD	24
3.7 Zaslona na dotik	26
3.8 Brezžični prenos podatkov	27
3.9 Kartica SD	29
4 Uporabniški vmesnik	31
4.1 Kalibracija zaslona na dotik	32
4.1.1 Implementacija kalibracijskega algoritma	34
4.2 Prikaz slike BMP	34
4.3 Prikaz znakov ASCII	36
4.4 Bresenhamov algoritem vzorčenja daljic	36
4.4.1 Implementacija algoritma	39

4.4.2 Izris grafa	39
4.5 Prikaz in kalibracija smeri vetra.....	40
5 SKLEP.....	43
Literatura in viri	45
Kazalo slik.....	47
Kazalo prilog	48

Seznam uporabljenih kratic in simbolov

- ADC - (ang.) Analog-to-digital converter (pretvornik analognega signala v digitalni signal)
- AHB - (ang.) Advance High-performance Bus (visoko zmogljivo vodilo v jedru Cortex-M4)
- APB - (ang.) Advanced Peripheral Bus (preprosto vodilo znotraj jedra Cortex-M4)
- ARM - (ang.) Advanced RISC Machines (skupina mikrokrmilnikov proizvajalca ARM Holdings)
- ASCII - (ang.) American Standard Code for Information Interchange (ameriški standardni nabor za izmenjavo informacij - 7-bitni nabor znakov)
- CPU - Centralno procesna enota
- DAC - (ang.) Digital-to-analog converter (pretvornik digitalnega signala v analogni signal)
- DMA - (ang.) Direct memory access (neposreden dostop do notranjega pomnilnika)
- E²PROM - (ang.) Electrically Erasable Programmable Read-Only Memory (Električno zbrisljiv in programirljiv bralni pomnilnik)
- FIFO - (ang.) First In, First Out (metoda za organizirane podatkovnega pomnilnika, kjer se najstarejši vpisan podatek tudi prvi prebere)
- Flash - (ang.) Flash memory (bliskovni pomnilnik)
- FSK - (ang.) Frequency-shift keying (metoda oziroma sistem frekvenčne modulacije)
- FSMC - (ang.) Flexible Static Memory Controller (namenski krmilnik za upravljanje z zunanjimi pomnilniki)
- GDDRAM - (ang.) Graphic display data RAM (pomnilnik RAM namenjen za shranjevanje podatkov o točkah za prikaz na LCD zaslon)
- GPIO - (ang.) General-purpose input/output (splošno namenski vhodi/izhodi)
- I²C - (ang.) Inter-Integrated Circuit (vodilo za povezovanje počasnejših naprav z mikrokrmilnikom)
- IRQ - (ang.) interrupt request (zahteva po prekinitvi)
- LCD - (ang.) Liquid-crystal display (zaslon s tekočimi kristali)
- LED - (ang.) Light-emitting diode (svetleča dioda)
- LSB - (ang.) Least significant bit (pozicija bita z najnižjo vrednostjo, imenovanega tudi skrajno-desni bit)
- MSB - (ang.) Most significant bit (pozicija bita z najvišjo vrednostjo, imenovanega tudi skrajno-levi bit)
- NVIC - (ang.) Nested Vectored Interrupt Controller (prekinitveni krmilnik v jedru Cortex-M4)
- PWM - (ang.) Pulse-width modulation (impulzno širinska modulacija)
- PCMCIA - (ang.) Personal Computer Memory Card International Association (periferni vmesnik načrtovan za prenosne računalnike, poznan tudi pod imenom PC Card)
- RAM - (ang.) Random Access Memory (bralno-pisalni pomnilnik)
- ROM - (ang.) Read-Only Memory (bralni pomnilnik)

RX - (ang.) Receiver/reciving (sprejemnik/sprejemanje)

SD - (ang.) Secure Digital (format spominske kartice)

SRAM - (ang.) Static Random Access Memory (statični bralno-pisalni pomnilnik)

STM32F4 - namenski mikrokontroler na STM32F4 Discovery razvojni ploščici

TX - (ang.) Transmitter/transmitting (oddajnik/oddajanje)

USART - (ang.) Universal Asynchronous Receiver/Transmitter (univerzalni asinhroni sprejemnik/oddajnik)

USB - (ang.) Universal Serial Bus (večnamensko vodilo, namenjeno priklopu različnih perifernih naprav na računalnik)

Povzetek

Cilj diplomske naloge je bil zasnovati in izdelati preprosto vremensko postajo, ki bi zbirala podatke iz ustreznih senzorjev in le-te prikazala na uporabniku prijaznem vmesniku na zaslonu LCD. V diplomski nalogi so opisana uporabljena orodja in elementi ter njihova uporaba in končna implementacija. Rezultat diplomske naloge predstavlja delujoča vremenska postaja z brezžičnim prenosom podatkov. Sistema sestavljata dva mikrokrmilnika z arhitekturo ARM z jedrom Cortex-M4F, oddajniški ter sprejemniški RF modul, ustrezni senzorji za odčitavanje najpogostejših vremenskih podatkov (temperatura, zračna vlaga, zračni pritisk, smer ter hitrost vetra) ter na dotik občutljiv zaslon. Osrednji del naloge vsebuje programsko implementacijo zajemanja in obdelave podatkov, pošiljanja le-teh preko RF vmesnika ter prikaz podatkov na sprejemniškem modulu. Na koncu je prikazan še preprost uporabniški vmesnik, ki omogoča interakcijo med uporabnikom in vremensko postajo. Predstavljen je tudi pogost problem nepravilnosti zaslonov na dotik in ustreznega kalibracijskega algoritma.

Ključne besede: ARM, STM32F4, vremenska postaja, RF, senzor, LCD

Abstract

The goal of the thesis was to design and develop a simple weather station, which would collect data from the relevant sensors and display them on a LCD within an user friendly interface. The description of tools and elements used, along with their use and implementation, are also presented. The result is a working weather station with wireless data transmission. The system consists of 2 microcontrollers, based on the ARM architecture with a Cortex-M4F core, a RF transmitter and receiver, relevant sensors for reading the basic weather data (temperature, relative humidity, atmospheric pressure, wind speed and direction) and a touch screen LCD display. The central part describes the programming implementation of data acquisition and processing, RF communication and receiver data displaying. Finally the user interface is described, which allows the interaction between the user and the weather station. Featured is also the common misalignment problem of touch screens along with its calibration algorithm.

Key words: ARM, STM32F4, weather station, RF, sensor, LCD

1 Uvod

Princip opazovanja vremena ter vremenskih pojavov pozna človeštvo že dolgo časa. Nekoč je opazovanje temeljilo zgolj na občutkih in zaznavah, skozi čas in z razvojem tehnologije pa je postajalo bolj in bolj kompleksno ter se nazadnje razvilo v lastno znanost.

Vremenska postaja je naprava, lahko digitalna ali analogna, namenjena zajemanju in beleženju podatkov o vremenu. Teh podatkov je več, odvisno od kompleksnosti same postaje. Osnovni in naj pogostejši podatki, ki se beležijo, vključujejo temperaturo zraka, zračno vlažnost, zračni pritisk, hitrost vetra, smer vetra ter merilec količine padavin. Naprednejše postaje lahko vključujejo tudi merjenje vidljivosti, UV index, nivo smoga v zraku....

Pri analognih postajah poteka odčitavanje podatkov ročno, navodno enkrat do dvakrat dnevno, dandanes pa so jih v večini že izpodrinile digitalne vremenske postaje, katere omogočajo preprostejšo zbiranje in analizo vremenskih podatkov.

V zadnjem času so priljubljene tako imenovane "hišne" oziroma "domače" vremenske postaje za lastno rabo. Pznamo več vrst, od nizkocenovnih, ki omogočajo le merjenje temperature in zračne vlage, pa do bolj naprednih, ki omogočajo tudi odčitavanje hitrosti ter smeri vetra.

Ideja diplomskega dela je bila realizirat preprosto vremensko postajo za lastno uporabo, ki bi merila osnovne podatke (temperaturo, zračno vlažnost, zračni pritisk, hitrost ter smer vetra) ter jih prikazala na naprednem uporabniškem vmesniku. Realizacija vsebuje dva modula, odajniškega in sprejemniškega. Na odajniškem modulu se nahajajo senzorji, podatki pa se preko brezžične komunikacije prenašajo na sprejemniški modul, kjer se prikažejo na zaslonu LCD.

V drugem poglavju smo pod drobnogled vzeli posamezne senzorje in module, opisali njihove lastnosti in prikazali shemo povezav z mikrokrmilnikom.

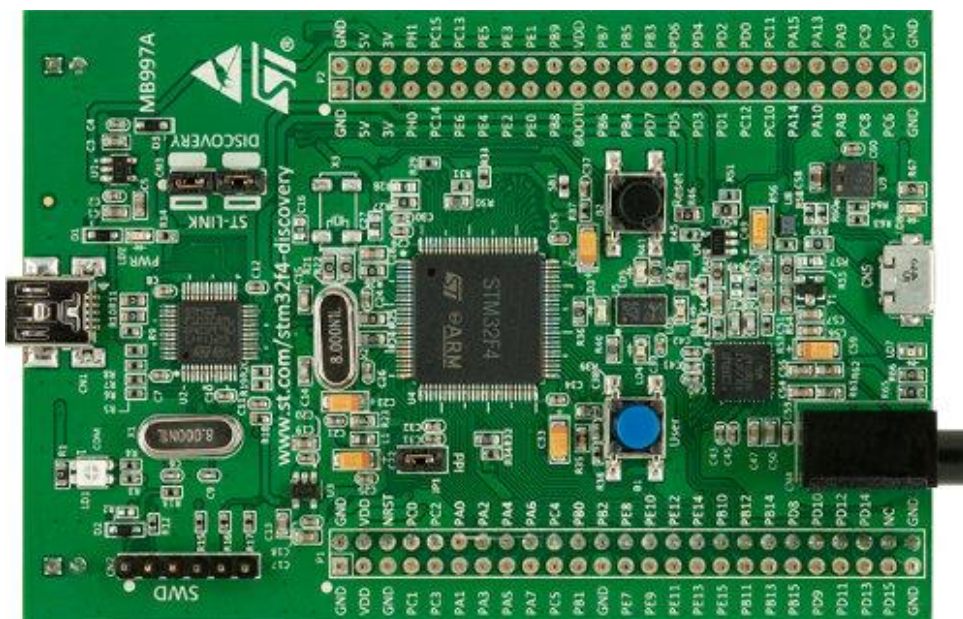
V tretjem poglavju je podrobno opisana programska realizacija. Prisotnih je bilo več različnih standardov komunikacije med mikrokrmilnikom in posameznimi moduli. Omenjeni so tudi časovniki in prekinitve, ki bistveno vplivajo na delovanje samega programa.

V četrtem poglavju je opisan uporabniški vmesnik, ki smo ga zasnovali tako, da prikazuje čim več podatkov in hkrati ni prenatrpan. Poleg podatkov iz senzorjev prikazuje tudi dva podatka o temperaturi, ki ga človeško telo zaznava bodisi pozimi bodisi poleti.

2 Uporabljena strojna oprema

2.1 STM32F4 Discovery

Za jedro vremenske postaje smo uporabili razvojno ploščo STM32F4 Discovery (slika 1) ARM proizvajalca STMicroelectronics (www.st.com), na kateri se nahaja mikrokrmilnik STM32F407VGT6. Gre za 32-bitni mikrokrmilnik, ki se ponaša z jedrom Cortex-M4F, 1MB Flash ter 192kB pomnilnika RAM.



Slika 1: Razvojna plošča STM32F4 Discovery proizvajalca STMicroelectronics.

Osnovne značilnosti mikrokrmilnika:

- jedro: 32-bitni Cortex-M4F s frekvenco 168MHz,
- pomnilnik Flash: 1MB,
- pomnilnik RAM: 192kB,
- ločena enota za operande v plavajoči vejici (*ang. Floating Point Unit*),
- vgrajen prekinitveni krmilnik (*ang. Nested Vectored Interrupt Controller*),
- 2x 12-bitni pretvornik DAC ter 3x 12-bitni pretvorniki ADC,
- 2x krmilnik DMA (*ang. Direct Memory Access*),
- 12x 16-bitnih časovnikov ter 2x 32-bitna časovnika,
- 3x vmesnik I²C (*ang. Inter-Integrated Circuit*),
- vmesnik SDIO (*ang. Secure Digital Input Output*),
- 3x vmesnik SPI (*ang. Serial Peripheral Interface*),
- 4x vmesnik USART (*ang. Universal Asynchronous Receiver/Transmitter*),

- 2x vmesnik CAN (*ang. Controller Area Network*).

Na razvojni plošči STM32F4 Discovery najdemo:

- 84-splošno namenskih vhodov/izhodov (GPIO),
- 2x 5V pin, 2x 3V pin, 3x Vdd pin, 10x GND pin
- 1x Reset, 1x User gumb,
- 1x priključek SWD (*ang. Serial Wire Debug*) – za programiranje mikrokrmilnika,
- 1x priključek USB – komunikacija z računalnikom in napajanje,
- 4x programabilna LED dioda,
- 1x 3.5mm audio konektor,
- 1x konektor USB OTG (*ang. USB On-The-Go*),
- 1x vgrajen pospeškometer.

2.1.1 Vodili AHB ter APB

Na mikrokrmilniku najdemo dva tipa preprostih vodil, ki služita za komunikacijo med programsko opremo (ki teče na CPU) in zunanji napravami na vgrajenem sistemu [13].

AHB (*ang. Advance High-performance Bus*) je visoko zmogljivo vodilo, ki omogoča da imamo več gospodarjev vodila kar naknadno omogoča po več hkratnih prenosov po vodilu med različnimi gospodarji in sužnji.

Na mikrokrmilniku STM32F4 imamo osem gospodarjev: I, D in S-vodila jedra Cortex M4, vodili DMA1 ter DMA2 za pomnilnik, vodilo DMA2 za periferne naprave, vodilo ethernet DMA ter vodilo USB OTG HS DMA. Sužnjev pa je sedem, med njimi mostički AHB-APB ter FSMC (več v poglavju 3.6) [16].

APB (*ang. Advanced Peripheral Bus*) je preprosto vodilo namenjeno priključevanju počasnih naprav. Imamo samo enega gospodarja - mostiček AHB-APB kar omogoča le en prenos v določenem trenutku. Na mikrokrmilniku STM32F4 imamo dva taka mostička: AHB-APB1 ter AHB-APB2, ki povezujeta vodili APB1/APB2 z vodilom AHB. Sužnji na vodilu pa so naprave: I²C (več v poglavju 3.3), USART, splošno namenski vhodni/izhodni pini (GPIO) [19]...

Frekvenco vodil lahko po želji spreminjamo z delilniki (*ang. prescaler*), za osnovne nastavitve pa velja (razbrano iz `system_stm32f4xx.c` sistemske kode): AHB deluje s frekvenco ure 168MHz (delilnik 1), APB1 s frekvenco 42MHz (delilnik 4), APB2 pa s frekvenco 86MHz (delilnik 2) [16].

Zaradi varčevanja z energijo so ob zagonu mikrokontrolerja vse periferne ure izklopljene. Če želimo posamezno napravo uporabljati, ji moramo vključiti urin signal.

2.1.2 Prekinitveni vektor NVIC

NVIC podpira do 240 prekinitev z do 256 nivojev prioritet na vsako prekinitev, te pa je mogoče spreminjati tudi dinamično. Krmilnik in jedro procesorja sta tesno povezana, kar omogoča nizko latenco obdelave prekinitev ter učinkovitejšo procesiranje poznih prekinitev [1].

Izjeme in prekinitve so vektorske, kar pomeni, da vsaki izjemi in prekinitvi pripada fiksni prekinitveni vektor. Ta vektor je dejansko naslov pomnilniške besede, na kateri se nahaja naslov prvega ukaza prekinitveno servisnega podprograma [13].

V STM32F407xx imamo 82 prekinitvenih vhodov (ki ne vključujejo 16 prekinitvenih linij jedra procesorja), vsaka z do 16 prioritetnih nivojev (4 biti), le-te je mogoče tudi maskirati [16].

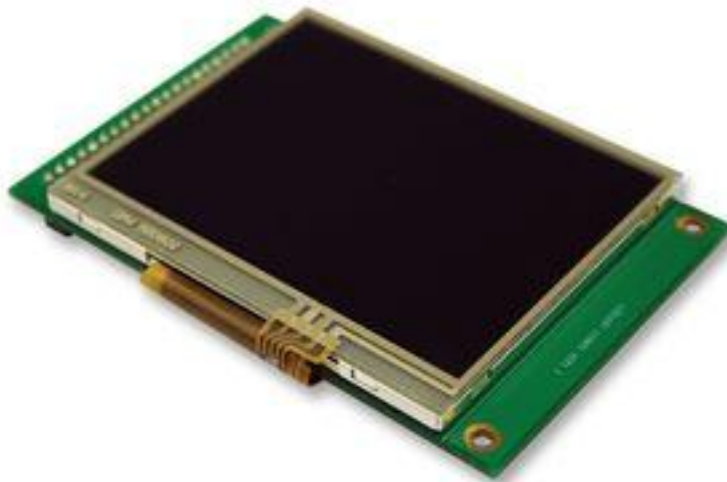
2.2 STM32F4DIS-BB ter STM32F4DIS-LCD

Uporabili smo tudi razširitevno vezje (slika 2) proizvajalca Embest Technology (www.embest-tech.com), katero je mogoče priklopiti modul LCD (slika 3) istega proizvajalca. Poleg tega se na vezju nahaja tudi konektor Ethernet, modul za kartico mikro SD, konektor za kamero (potrebuje dodaten modul) ter konektor RS232.



Slika 2: Razširitveno vezje STM32F4DIS-BB proizvajalca Embest Technology

Za prikaz podatkov ter interakcijo z vremensko postajo smo uporabili na dotik občutljiv zaslon LCD istega proizvajalca. Ponaša se s 3,5 palcev velikim zaslonom, do 320x240 pik prikaza ter do 262K barv. Za prikaz slike ima vgrajen krmilnik SSD2119, za krmljenje na dotik občutljivega zaslona pa skrbi krmilnik STMPE811.



Slika 3: STM32F4DIS-LCD

2.2.1 Krmilnik SSD2119

S krmilnikom komuniciramo preko kontrolnih registrov ter prikazovalnega pomnilnika.

S pisanjem v kontrolne registre upravljamo z nastavitvami krmilnika. Ob inicializaciji je treba nastaviti za napajanje, format prikaza, in ostale nastavitve (več v poglavju 3.6).

Prikazovalni pomnilnik nam služi za prikazovanje podatkov na zaslon. V njem se hranijo podatki o posameznih točkah in barvah le-teh. Sestavlja ga 172.800 (320 x 240 x 18 / 8) bajtov pomnilnika GDDRAM. Z mikrokrmilnikom komunicira preko 8-, 9-, 16- ali 18-bitnega paralelnega vmesnika, združljivega s standardom 6800 ali Intel 8080, podprt pa je tudi vmesnik SPI [15].

Z razširitvenim vezjem je (skupaj s krmilnikom dotika) povezan preko 40-žilnega podatkovnega kabla, komunikacija pa je nastavljena na 16-bitni 8080-kompatibilni paralelni vmesnik, na STM32F4 pa smo uporabili namenski krmilnik FSMC (več v poglavju 3.6).

2.2.2 Krmilnik STMPE811

Rezistivni zasloni na dotik so sestavljeni iz dveh fleksibilnih kovinskih plasti, ločenih s tanko zračno režo. Plasti sta izmenično pod napetostjo, ko pride do pritiska krmilnik izmeri napetost na plasti, ki ni pod napetostjo, nato preklopi napajanje na drugo plast in ponovi postopek – tako dobimo točke dotika [14].

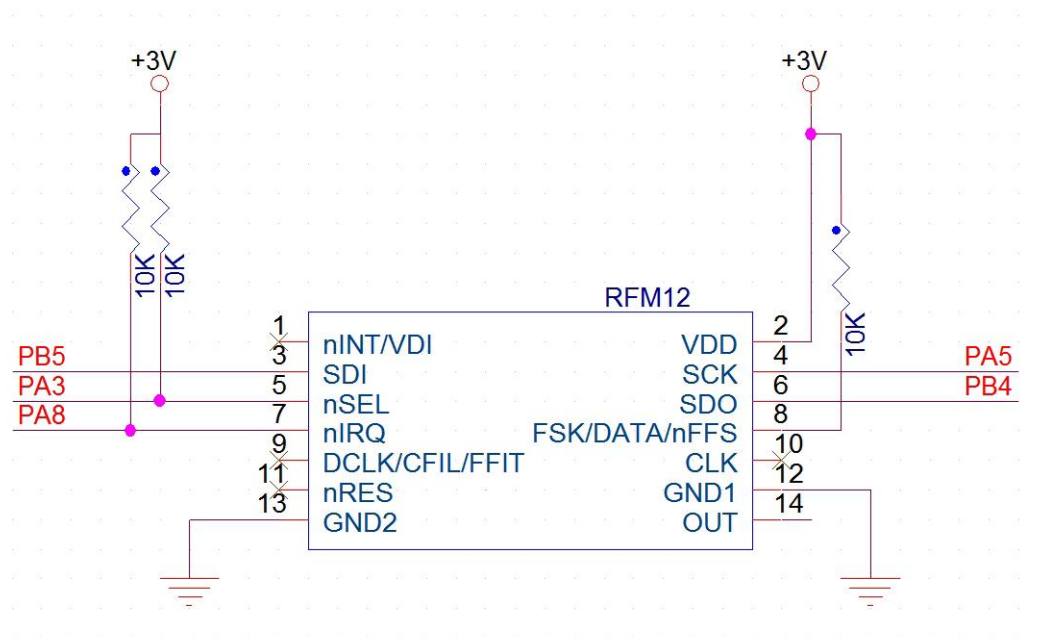
Zaslon na dotik z mikrokrmilnikom komunicira preko povezave I²C z ločenim izhodom za prekinitev, ima 12-bitni pretvornik ADC (vsaka koordinata je predstavljena z vrednostjo med 0 in 4095), na njem pa najdemo tudi temperaturni senzor.

2.3 Modul za brezžično komunikacijo RFM12B

Za komunikacijo med obema ločenima modula vremenske postaje je bil uporabljen RF modul proizvajalca Hope Microelectronics (www.hoperf.com). Modul deluje v frekvenčnem območju 868MHz, z mikrokrmilnikom pa komunicira preko vmesnika SPI (več v poglavju 3.8). Podatki (biti) se prenašajo s pomočjo frekvenčne modulacije FSK signala, kjer odstopanje od centralne frekvence predstavlja logično vrednost bita.

Gre za serijsko komunikacijo z uporabo štirih linij (slika 4):

- SCLK (SCK) – urin signal, tvori ga gospodar,
- NSS (nSEL) – linija za izbiro sužnja; za vsakega sužnja potrebujemo eno linijo; aktivna je v nizkem stanju,
- MOSI (SDI) – podatkovna linija za prenos podatkov od gospodarja do sužnja,
- MISO (SDO) – podatkovna linija za prenos podatkov od sužnja do gospodarja.



Slika 4: Shema povezave RFM12B s STM32F4 Discovery

Zahteva za komunikacijo z mikrokrmilnikom s strani RF modula predstavlja generiranje prekinitve preko linije nIRQ. Opcijsko se lahko uporablja tudi linijo VDI (*ang. Valid data indicator*).

Antena modulu ni bila priložena. Izdelana je bila iz preproste ravne bakrene žice, dolžina pa izračunana po enačbi (1). $1/4$ predstavlja faktor pri katerem antena resonira.

$$l = 1/4 \lambda \quad (1)$$

λ izračunamo z enačbo (2), kjer c predstavlja svetlobno hitrost, f pa frekvenco oddajanja. Elektro-magnetno valovanje signala potuje počasneje skozi razne materiale, zato je treba zmanjšati svetlobno hitrost za faktor hitrosti širjenja valovanja [3]. Za bakreno žico velja faktor 0,951.

$$\lambda = c \times 0,951/f \quad (2)$$

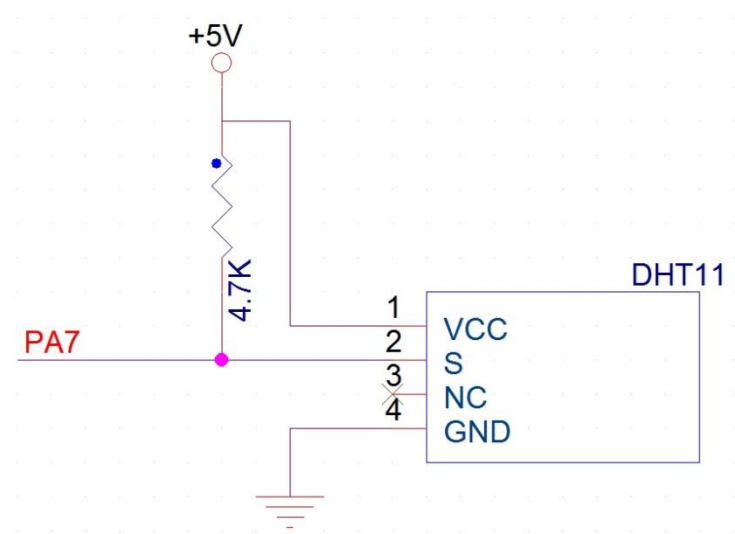
Izračunana dolžina antene v našem primeru znaša 82,2mm.

2.4 Senzor temperature in zračne vlage DHT11

Komunikacija z mikrokrmilnikom poteka preko enostavnega eno-žičnega komunikacijskega protokola [4]. Komunikacija se prične s START ukazom s strani mikrokrmilnika, nato sledita dva signala s strani DHT11, ki nakazujeta začetek prenosa podatkov, zatem pa pošlje 40-bitov podatkov za temperaturo in vlago (več v poglavju 3.2).

DHT11 omogoča merjenje temperature do natančnosti 1°C ter vlage do 1%.

Senzor se napaja med 3 do 5,5V, proizvajalec pa tudi priporoča en pull-up upor v vrednosti 5kOhm do razdalje 20m. Zaradi občasne nestabilnosti (vsaj na testni plošči) smo izbrali napajanje 5V (slika 5).



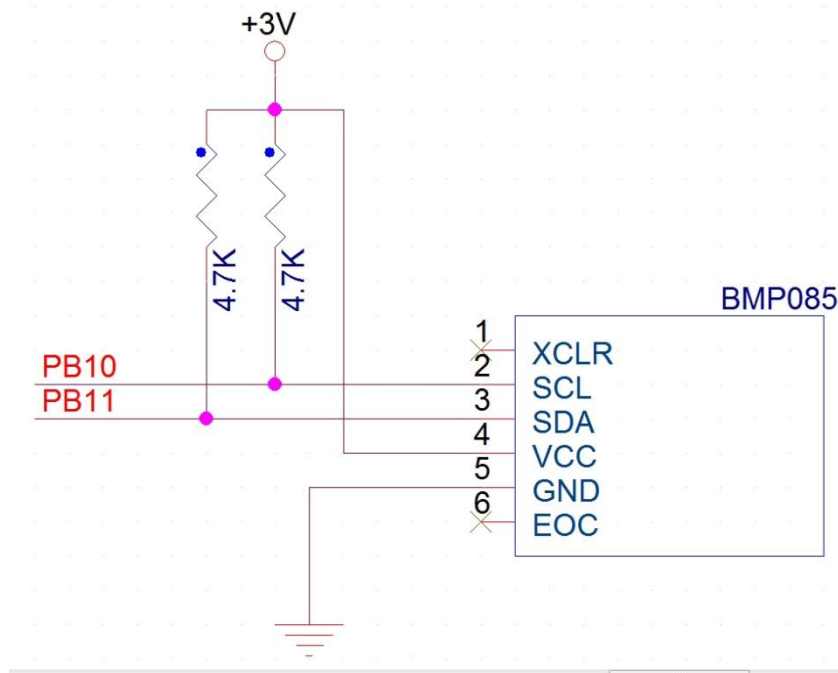
Slika 5: Shema povezave DHT11 s STM32F4 Discovery

2.5 Senzor temperature in zračnega pritiska BMP085

Izbran je bil senzor proizvajalca Bosch. Za razliko od DHT11 nam ta omogoča merjenje temperature do natančnosti $0,1^{\circ}\text{C}$, zračni pritisk pa do natančnosti $0,01\text{hPa}$. Komunikacija z mikrokrmilnikom poteka preko vodila I²C.

I²C komunikacija poteka preko dveh open-drain linij – SCL ter SDA. Obvezna je uporaba pull-up uporov (tipično 4,7k Ohm pri 3,3V) (slika 6).

Gre za gospodar-suženj vodilo (lahko imamo več gospodarjev), naprave lahko uporabljajo 7 ali 10-bitni naslov za komunikacijo. Tipična hitrost vodila je 100kbit/s v standardnem načinu, 10kbit/s v počasnem načinu ter 400kbit/s v hitrem načinu.



Slika 6: Shema povezave BMP085 s STM32F4 Discovery

2.6 Anemometer ter Hall senzor

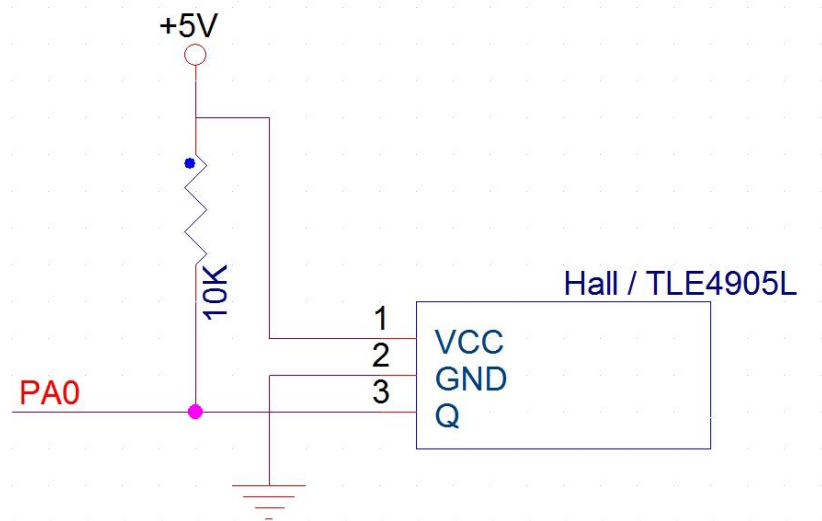
Osnovni princip merilnika hitrosti vetra je zaznavanje števila obratov v določenem času. V ta namen smo izkoristili senzor na Hallov pojav skupaj z neodim magnetom. Sam magnet je pritrjen na vetrnico in se vrti z njo, Hallov senzor pa zaznava prehode magneta ter generira urino fronto. Le-to pa zaznavamo na mikrokontrolerju kot prekinitve.

Hallove senzorje delimo v dve večji skupini:

Linearni Hall senzor generira izhodno napetost ki je proporcionalna glede na jakost magnetnega polja pravokotno na površino senzorja. Izhod je dejansko analogna vrednost. Tipičen senzor ob odsotnosti magnetnega polja daje izhodno napetost, ki je pol manjša od vhodne napetosti. Ko približamo južno stran magneta, izhodna napetost začne naraščati (največ do Vcc) ter obratno, s severno stranjo magneta izhodna napetost pada (do 0V).

Hall switch senzorji imajo s pomočjo vgrajenega Schmitt-triggerja digitalni izhod signala. Ker je izhod tipa open-drain, je potreben pri teh senzorjih en pull-up upor (slika 7).

V projektu smo uporabili Hall switch senzor TLE4905L proizvajalca Infineon Technologies [10]. Gre za unipolarni senzor, ki nam da nizko stanje ob prisotnosti pozitivnega magnetnega polja (približanje severnega pola magneta).



Slika 7: Shema vezave Hallovega senzorja s STM32F4 Discovery

2.7 Merilnik smeri vetra

Za merilnik smeri vetra smo uporabili preprost potenciometer s kontinuirano rotacijo. Take potenciometre navadno dobimo v preprostih napravah kot so glasbeni stolpi, avtoradii...

Upornost potenciometra narašča med 0 Ohm pa vse do 5kOhm, ko doseže najvišjo vrednost, začne spet od 0 Ohm.

Izhodno analogno napetost preprosto odčitavamo in s pretvornikom ADC na STM32F4 pretvorimo v 12-bitno vrednost, le-to pa konvertiramo v kot zasuka v stopinjah.

3 Programska realizacija

Za programiranje mikrokrmilnikov STM je na voljo več razvojnih orodij [20]: IAR EWARM, Keil MDK-ARM, Atolic TrueStudio, in drugi, ki podpirajo vgrajen razhroščevalnik ST-LINK/V2.

Med sabo so si orodja precej podobna, zato smo se odločili uporabiti kar prvega – Embedded Workbench for ARM proizvajalca IAR Systems (www.iar.com).

IAR nam ponuja dve licenci programja:

- kickstart, omejena na 32kB velikosti izvorne kode,
- časovno omejena na 30 dni (pod pogojem, da se izdelek ne uporablja v komercialne namene).

Uporabili smo kickstart verzijo.

Pri programiranju mikrokrmilnika smo izkoristili tudi knjižnice proizvajalca STMicroelectronics [17]. Pričeli smo že z ogrođjem, ki vsebuje nastavitve za osnovno delovanje mikrokrmilnika, kot so: nastavitve sistemske ure, delilniki ur, nastavitve skladovnega kazalca ter prekinitvenega vektorja, inicializacija pomnilnika Flash, ...

Vremenska postaja je razdeljena na dva modula – oddajniški in sprejemniški. Na obeh se program prične z inicializacijo omenjenih nastavitvev (slika 8–9).

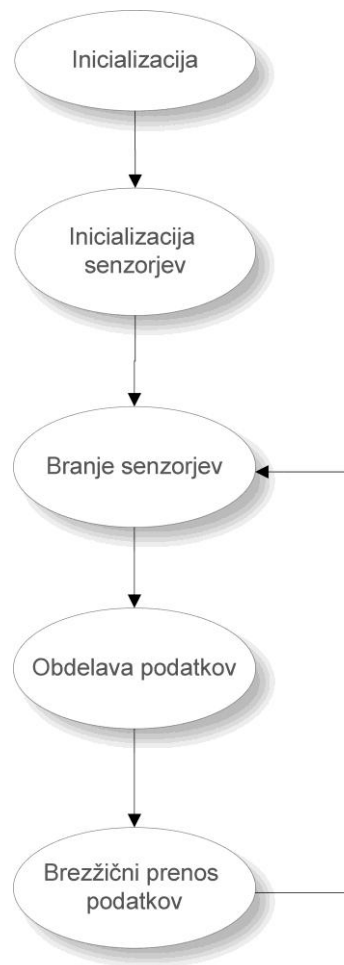
Pri tem omenimo, da nam jedro procesorja ter vodilo AHB tečeta na 168MHz, vodilo APB1 na 42MHz ter vodilo APB2 na 86MHz.

Na oddajniškem modulu (slika 8) sledi inicializacija senzorjev, za tem pa še branje le-teh. Pri samih inicializacijah ne smemo pozabiti vključiti ure na perifernih napravah, ki jih želimo uporabiti..

Pred pošiljanjem podatkov po brezžičnem prenosu pa se morajo ti ustrezno predelati v nepredznačene 8-bitne vrednosti (jih ustrezno omejimo in prištejemo oziroma odštejemo usrezne vrednosti).

Nato sledi prenos podatkov, ki ga ponovimo dvakrat, če pride do napake pri prvem prenosu.

Branje senzorjev, obdelava ter prenos podatkov se periodično ponavlja na vsaki dve minuti.

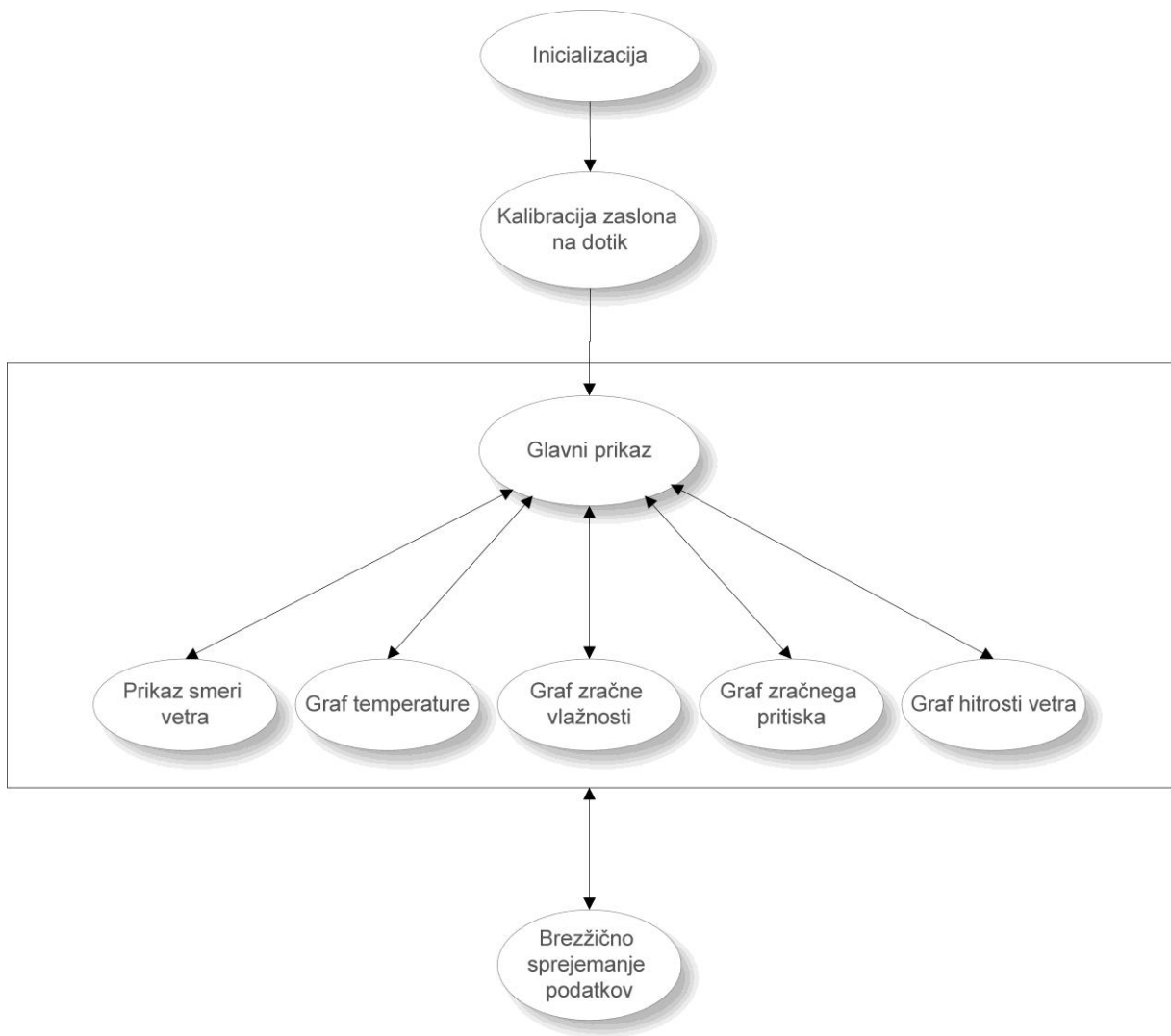


Slika 8: Shema delovanja oddajniškega modula.

Na sprejemniškem modulu (slika 9) sledi inicializacija zaslona na dotik ter kalibracija le-tega. Znotraj te zanke čakamo, dokler se ne zgodita dotik in odčitavanje treh točk ter konča sam postopek kalibracije. Pomembno je, da so v tem trenutku prekinitve zaslona izključene, saj bi motile zanko, kjer čakamo na dotik.

Po kalibraciji nastopi glavna zanka sistema, ki se prične z glavnim prikazom, kjer prikazujemo vse aktualne podatke. S klikom na zaslon lahko menjujemo med različnimi prikazi.

V času neaktivnosti prikazov poteka odčitavanje sprejemniškega modula za morebitne nove podatke.



Slika 9: Shema delovanja sistema.

3.1 Časovniki in prekinitve

Pri inicializacijah naleteli na prvo težavo – kako nastaviti preprosto zakasnitev.

Ko delamo s počasnimi napravami, od njih ne moremo zahtevati hitrega preklapljanja med stanji, mikrokrmilnik se jim mora prilagoditi, drugače pride do napak pri komunikacijah. Primer sta senzorja DHT11 in BMP085. DHT11 potrebuje vsaj sekundo po vključenem napajanju, da preide v stabilno stanje, BMP085 pa čas, da preračuna vrednosti, preden jih pošlje mikrokrmilniku.

Zakasnitve lahko realiziramo na več načinov, še najbolj točne pa so s časovniki. Kot smo omenili, ima STM32F4 14 časovnikov, oštevilčenih od TIM1 do TIM14. Le-te delimo v štiri skupine [16]:

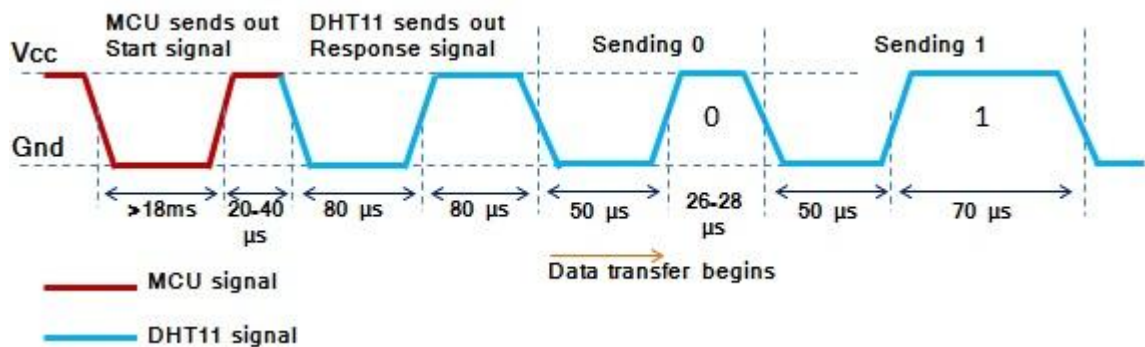
našem primeru ima časovnik TIM2 IRQ številko 28, zunanja prekinitve zaslon na dotik pa številko 40.

Vsem prekinitvam lahko nastavimo še prioriteto. Skupaj imamo 256 nivojev prioritete, razdeljenih v 16 skupin ter 16 nivojev znotraj posamezne skupine.

V našem programu smo časovnikom dodelili najvišjo prioriteto, najnižjo pa zaslonu na dotik.

3.2 DHT11 komunikacija

Na sliki 11 vidimo preprost časovni diagram komunikacije *1-wire*, ki jo uporablja senzor DHT11.



Slika 11: Časovni diagram komunikacije senzorja DHT11.

Komunikacijo začnemo s strani mikrokontrolerja [4]. Izhodno stanje postavimo na nizko ter počakamo 20ms, nato pa spet na visoko in počakamo 35µs. Nato spremenimo nastavev pina na vhodno in preverimo odgovor od DHT11.

Branje signala senzorja je realizirano s preprosto zanko in odčitavanjem stanja pina (slika 12). Znotraj vsake zanke je dodana še časovna omejitev, da se ne ujamemo v neskončni zanki.

```
cicles = 0;
while (GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_5) == Bit_SET) {
    cicles++;
    if (cicles > MAX_TICS) {
        return DHT_ERROR_TIMEOUT;
    }
}
```

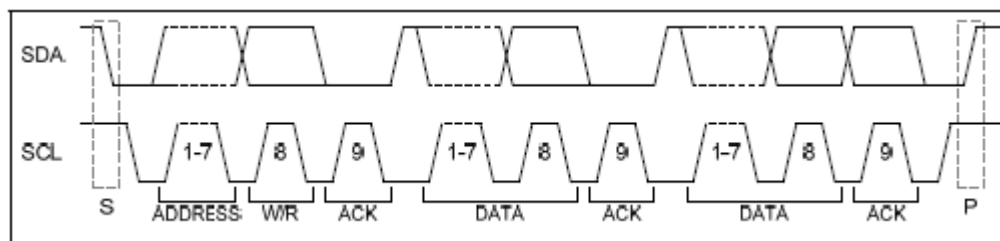
Slika 12: Preprosta bralna zanka visokega stanja enega signala.

Vrednost podatka definiramo tako, da najprej odčitamo nizko stanje, nato pred odčitavanjem visokega stanja poženemo štetje časa v mikro sekundah (realizirano s časovnikom), odčitamo visoko stanje ter prekinemo štetje časa. Če je pretečen čas visokega stanja manjši od 40 μ s, pomeni, da smo sprejeli bit 0, drugače pa bit 1. To ponovimo za vseh 40 bitov.

40-bitov prenosa je sestavljenih iz: 16-bitov (vlaga) + 16-bitov (temperatura) + 8-bitov (kontrolna vsota).

3.3 BMP085 in protokol I²C

I²C je sinhronska, half-duplex komunikacija, ki uporablja dve liniji. SCL predstavlja urin signal, SDA pa podatkovno linijo. Slika 13 prikazuje preprost protokol pri prenosu naslova ter dveh sklopov po 8 bitov podatkov.



Slika 13: I²C protokol.

Na vodilu I²C poznamo 4 načine delovanja:

- gospodar pošilja podatke,
- gospodar sprejema podatke,
- suženj pošilja podatke,
- suženj sprejema podatke

Prenos podatkov se začne tako, da gospodar pošlje bit START (prvo linijo SDA spusti v nizko stanje, zatem še linijo SCL v nizko). Zatem sledi 7-bitni naslov naprave ter dodaten RW bit, ki določa smer prenosa podatkov [13].

Ko suženj prepozna svoj naslov odgovori z bitom ACK na vodilu SDA. Zatem sledi prenos 8-bitnih podatkov, ki se konča z bitom ACK (s strani naprave, ki je sprejemala podatke).

Komunikacija se konča z bitom STOP.

Vmesnik I²C na STM32F4 nam omogoča komunikacijo z napravo, ne da bi se spuščali na fizično plast, poznati moramo le protokol. Za lažje delo smo izkoristili tudi knjižnico *stm32f4xx_i2c.c*.

Sam postopek komunikacije s strani gospodarja (gospodar pošilja podatke) je sledeč:

1. Preverimo če I²C ni zaseden – *I2C_GetFlagStatus(I2C_FLAG_BUSY)*.
2. Pošljemo bit START – *I2C_GenerateSTART()*.
3. Zastavica SB (*start bit*) se postavi, moramo jo pobrisati.
4. Pošljemo 7-bitni naslov sužnja in smer prenosa – *I2C_Send7bitAddress()*.
5. Ko gospodar dobi potrditev od sužnja, se postavi zastavica ADDR, moramo jo pobrisati.
6. Pošljemo bajt podatkov – *I2C_SendData()*.
7. Preverimo, če sta zastavici TxE (*data register empty*) ter BTF (*byte transfer finished*) postavljeni, nato ju zberemo. Če želimo ponovno poslati, ponovimo od točke 6.
8. Zaključimo komunikacijo – *I2C_GenerateSTOP()*.

Pri gospodarju kot sprejemniku so točke 1–5 enake, nakar se postopek razveja v dve smeri, odvisno od tega, ali sprejemamo le 1 bajt (oziroma zadnji bajt v sekvenci) ali nameravamo sprejeti več bajtov.

6. Če sprejemamo več kot 1 bajt podatkov omogočimo generiranje ACK – *I2C_AcknowledgeConfig(ENABLE)*.
7. Če sprejemamo 1 bajt (oziroma zadnji bajt v sekvenci) podatkov moramo onemogočiti generiranje ACK – *I2C_AcknowledgeConfig(DISABLE)*.
8. Preberemo podatke – *I2C_ReceiveData()*.
9. Ustrezno ponovimo točki 6 ali 7.
10. Zaključimo z bitom STOP.

Pri inicializaciji BMP085 moramo prebrati ustrezne 16-bitne kalibracijske vrednosti, ki se nahajajo v integriranem E²PROM. S pomočjo teh preračunamo končno vrednost temperature in zračnega pritiska. Vrednost teh registrov je fiksna, vendar se od naprave do naprave razlikujejo [2].

Sam postopek odčitavanja in preračunavanja vrednosti je prikazan v prilogi A.

3.4 Smer vetra ter krmilnik ADC

STM32F4 ima 3 analogno-digitalne pretvornike, vse tri vezane na vodilo APB2 [19]. Natančnost vzorčenja le-teh je do 12-bitov, seveda lahko izberemo tudi manj (6, 8 ali 10-bitno). Na voljo je 19 multipleksiranih kanalov ADC od tega 16 na voljo za merjenje iz zunanjih virov.

V našem primeru smo izbrali kar ADC1; potenciometer smo vezali na pin PC0, ki predstavlja kanal 10 (ADC123_IN10) [19].

Ko vzorčimo vhodno napetost dobimo kot rezultat 12-bitno vrednost, ki je med 0 in 4.095. To vrednost moramo pretvoriti v kot v stopinjah, kalibracijo smeri vetra opravimo v uporabniškem vmesniku (več v poglavju 4.5).

3.5 Hitrost vetra

Ko Hall senzor proži nizko fronto, to zaznamo na mikrokrmilniku, proži se prekinitev. Znotraj prekinitveno servisne rutine preprosto povečujemo števec, ki nam predstavlja število obratov. Dlje kot merimo število obratov, večjo natančnost imamo.

Ko pride do periodičnega odčitavanja senzorjev, zaradi večje natančnosti, beremo število obratov pred branjem ostalih senzorjev.

Kalibracijo smo opravili s pomočjo ročnega anemometra. Ker hitrost linearno narašča je zadostovalo kalibriranje pri dveh do treh različnih hitrostih. Po naših izračunih en obrat na sekundo predstavlja približno 0,4398 m/s oz. 1,583 km/h.

3.6 LCD

LCD modul krmilimo s pomočjo krmilnika FSMC. FSMC je namenski krmilnik v STM32F družini mikrokrmilnikov za upravljanje z zunanjimi spomini (SRAM, ROM, NOR Flash...) ter 16-bit PC Card (bol poznan kot PCMCIA) kompatibilnimi napravami [16].

Z hranjenjem vrednosti spremenljivk, smo seveda omejeni s pomnilniškim prostorom na notranjem pomnilniku RAM. S pomočjo FSMC lahko ta pomnilniški prostor razširimo. To lahko uporabimo tudi s krmilnikom SSD2119 zaslona LCD in na ta način preslikamo notranje registre v pomnilniški prostor. Tako poteka upravljanje in izrisovanje s preprostim vpisom vrednosti v "spremenljivko". To spremenljivko moramo najprej definirat kot kazalec na željen pomnilniški prostor.

Za konfiguracijo vmesnika FSMC smo si pomagali z demo primeri proizvajalca modula LCD [5].

Postopek inicializacije krmilnika SSD2119 je sledeč[5, 15]:

1. Vstopimo v način mirovanja, če že nismo v njem.
2. Nastavimo parametre napajanja.
3. Vključimo notranji oscilator.
4. Nastavimo orientacijo zaslona, način prikaza.

5. Izstopimo iz načina mirovanja, počakamo vsaj 30ms.
6. Nastavimo način prikaza barv ter vklopimo avtomatsko večanje naslovnega števca pomnilnika GDDRAM.
7. Nastavimo voltažo jedra ter ostale analogne parametre.
8. Vključimo zaslon.
9. Nastavimo napajanje VCIX2 na 6,1V.
10. Nastavimo gamma parametre.
11. Nastavimo napajanja Vlcd63 ter VcomL.
12. Nastavimo velikost zaslona.
13. Pobrišemo zaslon.

Kot smo omenili SSD2119 omogoča izris do 18-bitne barvne globino (262.144 oz 2^{6+6+6} barv). Na modulu pa imamo povezanih samo 16 podatkovnih linij, kar pomeni, da ne moremo poslati vseh 18 bitov naenkrat. Tako bi potrebovali vsaj dva prenosa (slika 14) po vodilu za prikaz ene točke.

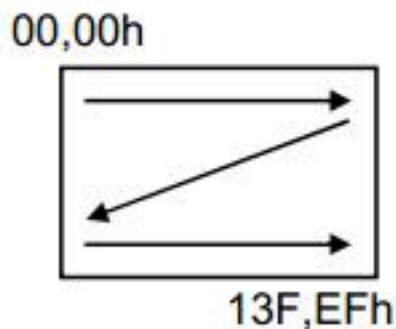
			Hardware pins																	
Interface	Color mode	Cycle	D17	D16	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0
16 bit	262k Type A	1 st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
		2 nd	B5	G4	B3	B2	B1	B0	x	x		R5	R4	R3	R2	R1	R0	x	x	
		3 rd	G5	G4	G3	G2	G1	G0	x	x		B5	G4	B3	B2	B1	B0	x	x	
	262k Type B	1 st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
		2 nd	x	x	x	x	x	x	x	x		B5	G4	B3	B2	B1	B0	x	x	
	262k Type C	1 st	R5	R4	R3	R2	R1	R0	x	x		G5	G4	G3	G2	G1	G0	x	x	
2 nd		B5	G4	B3	B2	B1	B0	x	x		x	x	x	x	x	x	x	x		
Remark :		x	Don't care bits Not connected pins																	

Slika 14: Trije načini pošiljanja 18-bitne barvne globine.

Zato smo se iz praktičnega razloga odločili uporabiti 16-bitno RGB565 barvno globino (65.536 oz 2^{5+6+5} barv). Ta način uporablja 5-bitov za rdečo in modro ter 6-bitov za zeleno barvo.

Za izris na zaslon moramo najprej določiti koordinato točke (X med 0 in 319 ter Y med 0 in 239), nato poslati ukaz, da pišemo v GDDRAM, šele za tem pa lahko pošljemo 16-bitno vrednost barve.

Ker pa smo vključili avtomatsko večanje naslovnega števca in so naslovi znotraj grafičnega pomnilnika (po X-osi) zaporedni, lahko pošljemo 16-bitne podatke v sekvenci, ne da bi prej določili koordinato izrisa (slika 15).



Slika 15: Prikaz večanja naslovnega števca grafičnega pomnilnika.

3.7 Zaslon na dotik

Kot smo omenili, zaslon na dotik uporablja komunikacijo I²C. Princip delovanja je isti kot pri senzorju BMP085, le za prenos vrednosti je bil uporabljen DMA.

Princip DMA je, da razbremeni procesor pri prenosu podatkov med pomnilnikom in različnimi perifernimi napravami (kot so I²C, ADC, SPI ...). Smiselno ga je uporabljati tam, kjer so potrebni časovno kritični prenosi.

V našem primeru nam prenos DMA zamenja del komunikacije, kjer bi uporabili *I2C_SendData()* ter *I2C_ReceiveData()*.

Postopek inicilizacije krmilnika STMPE811 je sledeč [18]:

1. Vključimo uro ter ADC na krmilniku.
2. Vključimo globalne prekinitve (čeprav so vključene, se v tem trenutku še ne prožijo).
3. Nastavimo vgrajen ADC (čas vzorčenja, 10- ali 12-bitno vzorčenje ter referenčno napetost).
4. Počakamo približno 20ms.
5. Izberemo 3.25MHz frekvenco na ADC.
6. Nastavimo notranje pine GPIO.
7. Nastavimo kontrolni register (število vzorčenj ter zakasnitve).
8. Nastavimo omejitev notranje vrste FIFO (izberemo 1 – hrani vrednost le enega dotika).
9. Izključimo ter ponovno vključimo vrsto FIFO – jo pobrišemo.
10. Omejimo tok znotraj pinov na 50mA.
11. Vključimo zaslon na dotik.
12. Pobrišemo vse prekinitve.

Prekinitve vključimo šele po kalibraciji zaslona. Koordinate dotika zaslona beremo v prekinitveno servisnem podprogramu. Po končanem branju koordinat pa moramo pobrisati register INT_STA, kjer se hrani stanje prekinitve. Če ga ne zberemo, bo krmilnik STMPE811 konstantno držal nizko stanje na liniji (ne bo več prožil prekinitve).

Pri odčitavanju vrednosti ADC, dotika koordinat X ter Y, smo naleteli na nekaj težav. Za koordinato X so namreč precej odstopale vrednosti med različnimi dotiki na isti točki, kar je kasneje tudi povzročalo težave pri kalibraciji. Težavo smo rešili tako, da smo povečali število branj registrov ADC z 1 na 16, ter na koncu izračunali povprečno vrednost.

Poleg koordinat X in Y krmilnik omogoča tudi koordinato Z, le-ta nam dejansko predstavlja jakost pritiska na zaslon. V našem programu je ne uporabljamo, vendar vseeno beremo, da se sproti briše vrsta FIFO.

3.8 Brežžični prenos podatkov

Komunikacija preko SPI se prične tako, da gospodar postavi NSS (v našem primeru nSEL) linije v nizko stanje (s tem izberemo sužnja). Nato se začne tvoriti urin signal ter postavljanje bitov na MOSI in branje iz MISO [13].

Protokol SPI pozna štiri načine delovanja, ki določajo ob kateri urini fronti se podatki prenašajo ter stanje SCLK linije v času mirovanja. Modul RFM12B tako prenaša podatke na vsako pozitivno fronto, SCLK linija pa je v mirovanju v nizkem stanju.

Za nastavitve ter komunikacijo z modulom RFM12B imamo 17 različnih 16-bitnih ukazov. Povratna zahteva za komunikacijo pa predstavlja prekinitiv na nIRQ liniji, le-ta je aktivna v nizkem stanju. Prekinitvenih izvorov je lahko več, to lahko odčitamo z ukazom za branje stanja (*ang. Status read command*), s tem pa tudi izbrišemo aktualno prekinitiv.

Sam postopek inicializacije je sledeč [7, 8]:

1. Izberemo frekvenčno območje, vključimo notranje podatkovne registre ter vrsto FIFO.
2. Izberemo način delovanja (oddajanje, sprejemanje, mirovanje).
3. Nastavimo centralno frekvenco f_0 .
4. Nastavimo bitno hitrost prenosa.
5. Na strani sprejemnika nastavimo ustrezne RX parametre, filter podatkov, nastavitve vrste FIFO ter deviacijo sprejemne frekvence v primerjavi z oddajno.
6. Na strani oddajnika nastavimo ustrezne TX parametre.
7. Po želji nastavimo še časovnik bujenja iz stanja mirovanja ter detekcijo stanja opsijske zunanje baterije.

Sam protokol komunikacije je definiran s strani proizvajalca. Podatki se v 16-bitni (sestavljeno iz dveh 8-bitnih pomikalnih pomnilnikov) pomnilnik modula prenesejo sinhronsko, bit MSB prvi, po 8-bitov na en prenos.

Oddajniški modul začne z oddajanjem na zahtevo mikrokrmilnika, ko je oddajanje končano pa postavi nIRQ v nizko stanje in sproži ustrezno prekinitev, takrat se lahko prenese nov bajt. Najprej se prenesejo dva uvodna bajta AA_{16} (*ang. preamble*), ki nakazujeta na začetek prenosa. Uvodu sledita bajta za sinhronizacijo oddajne in sprejemne ure $2D_{16}$ $D4_{16}$. Nato se prenesejo podatki, zaključek prenosa pa predstavlja bajt AA_{16} .

Slika 16 prikazuje podatkovni okvir enega brezžičnega prenosa, sliki 17 in 18 pa vsebine podatkov.

AA	AA	2D	D4	podatki	AA
----	----	----	----	---------	----

Slika 16: Podatkovni okvir.

dolžina	kontrolna vsota	meritve senzorjev
---------	-----------------	-------------------

Slika 17: Vsebina polja "podatki".

temperatura 1	zračna vlažnost	temperatura 2	zračni tlak	hitrost vetra	smer vetra
------------------	--------------------	------------------	----------------	------------------	---------------

Slika 18: Vsebina polja "meritve senzorjev".

Vsak od podatkov je predstavljen kot nepredznačeno 8-bitno število. V primeru temperature iz senzorja BMP085 smo morali podatek razdeliti v 2 bajta. Prvi bajt predstavlja celoštevilski del meritve, drugi pa decimalni del. Podobno velja tudi za smer vetra, ker vrednost prekorači velikost 1 bajt.

Kontrolna vsota je enaka zadnjim 8 bitom vsote vseh podatkov ter dolžine.

Sprejemanje poteka tako, da modul zaznane podatke začne vpisovati v vrsto FIFO. Ko vrsta doseže definiran limit bitov (v našem primeru smo nastavili na 8). se sproži prekinitev, ki naznanjuje mikrokrmilniku, da lahko prebere ustrezen bajt.

Uvodna bajta, bajta za sinhronizacijo ure ter zaključni bajt se sami izločijo na sprejemniškem modulu, tako da program ni treba skrbeti za to.

3.9 Kartica SD

Za interakcijo s kartico SD smo izkoristili rešitev (*stm32f4_discovery_sdio_sd.c*) od MCD Application Team [5] ter odprto-kodno rešitev FatFS R0.09b [6] za interakcijo z datotečnim sistemom FAT.

FatFS je namenski modul za majhne vgrajene sisteme s precej nizko porabo pomnilnika. Ponuja nam osnovna orodja za delo s particijami in datotekami (*f_mount()*, *f_open()*, *f_write()* ...).

4 Uporabniški vmesnik

Z uporabniškim vmesnikom želimo prikazati čim več podatkov, hkrati pa ta ne sme biti prenatrpan. Omislili smo si več prikazov:

- **glavni prikaz** (slika 19) – prikazujemo osnovne podatke senzorjev (temperatura, zračna vlaga, zračni pritisk, smer vetra ter hitrost vetra). Poleg teh pa še dva podatka o zaznavani temperaturi – *wind chill* (temperaturo, ki jo telo zaznava pozimi, ko je zunanja temperatura pod 10°C, in hitrost vetra nad 4,8km/h) ter *heat index* (temperaturo, ki jo telo zaznava poleti, za temperature nad 26°C ter relativno zračno vlago nad 40%);
- **grafični prikaz** preteklih podatkov za vsaj pol dneva – izbrali smo 14 ur (zaradi lažjega prikaza na zaslon). Grafe (koordinatni sistem je dejansko slika s kartice SD) prikazujemo za temperaturo, zračno vlago, zračni pritisk ter hitrost vetra;
- **vetrnica smeri vetra** – ideja je prikazat smer vetra na predefinirani vetrnici (slika, ki jo naložimo s kartice SD), podatke osveževati in ponuditi uporabniku možnost kalibriranja z dotikom na zaslon.



Slika 19: Glavni prikaz podatkov.

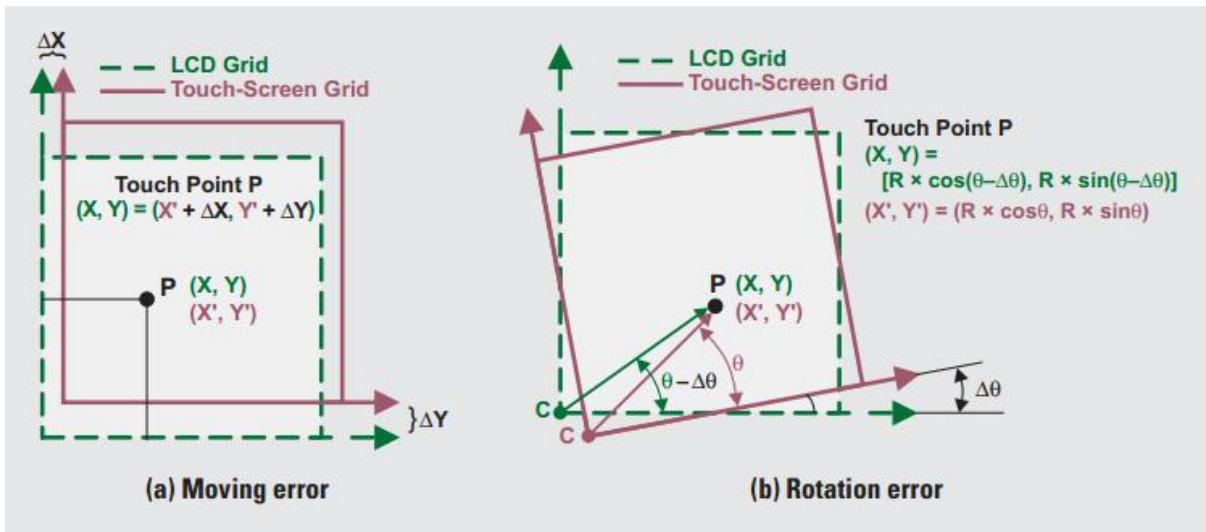
Med različnimi prikazi menjujemo s pritiskom na zaslon, točka pritiska v glavnem prikazu določa kateri pod-prikaz se bo prikazal.

Če se nahajamo v grafičnem prikazu, nas dotik na zaslon povrne v glavni prikaz.

Ko pa se nahajamo v prikazu smeri vetra, nam dotik znotraj vetrnice (X je med 50 in 280) sproži kalibracijo smeri vetra, dotik izven vetrnice pa vrnitev v glavni prikaz.

4.1 Kalibracija zaslona na dotik

Neobdelani podatki o koordinatah dotika so predstavljeni z 12-bitno vrednostjo, to je med 0 in 4.095. Le-te moramo ustrezno pretvoriti (skalirati) v koordinate našega zaslona. Po vrhu so mehanske nepravilnosti (premiki ter zasuki) med zaslonom LCD ter plastmi zaslona na dotik dokaj pogoste (slika 20).



Slika 20: Mehanska nepravilnost pri zaslonih na dotik.

Točka (X', Y') predstavlja dotik na zaslonu na dotik, (X, Y) pa točko na zaslonu LCD. Z nadaljnjimi enačbami bomo opisali korelacijo med točkami ter preprosto kalibracijo zaslona z metodo treh točk [21].

$$\begin{aligned}
 X &= k_X \times R \cos(\theta - \Delta\theta) + \Delta X \\
 X &= k_X \times R \cos \theta \times \cos(\Delta\theta) + k_X \times R \times \sin \theta \times \sin(\Delta\theta) + \Delta X \\
 X &= k_X \times X' \times \cos(\Delta\theta) + k_X \times Y' \times \sin(\Delta\theta) + \Delta X \\
 X &= \alpha_X \times X' + \beta_X \times Y' + \Delta X
 \end{aligned} \tag{3}$$

Kjer velja:

- k_X – skaliranje na osi X,
- k_Y – skaliranje na osi Y,
- R – razdalja od izhodišča (0, 0) do točke P,
- $\Delta\theta$ – razlika v zasuku med LCD in zaslonom na dotik,
- ΔX – premik po osi X,
- ΔY – premik po osi Y.

V enačbah (3) je predstavljen zapis koordinate X na zaslonu LCD z upoštevanjem vseh treh transformacij.

Na podoben način dobimo enačbo (4) koordinate Y

$$Y = \alpha_Y \times X' + \beta_Y \times Y' + \Delta Y \quad (4)$$

Iz enačb (3) ter (4) je razvidno, da za koeficiente α_X , α_Y , β_X , β_Y , ΔX ter ΔY potrebujemo vsaj tri nekolinearne točke, od tod dobimo enačbe (5).

$$\begin{aligned} X_1 &= \alpha_X \times X'_1 + \beta_X \times Y'_1 + \Delta X \\ X_2 &= \alpha_X \times X'_2 + \beta_X \times Y'_2 + \Delta X \\ X_3 &= \alpha_X \times X'_3 + \beta_X \times Y'_3 + \Delta X \end{aligned} \quad (5)$$

$$\begin{aligned} Y_1 &= \alpha_Y \times X'_1 + \beta_Y \times Y'_1 + \Delta Y \\ Y_2 &= \alpha_Y \times X'_2 + \beta_Y \times Y'_2 + \Delta Y \\ Y_3 &= \alpha_Y \times X'_3 + \beta_Y \times Y'_3 + \Delta Y \end{aligned}$$

Le-te lahko zapišemo matrično (6)

$$\begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} = A \times \begin{pmatrix} \alpha_X \\ \beta_X \\ \Delta X \end{pmatrix} \text{ ter } \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} = A \times \begin{pmatrix} \alpha_Y \\ \beta_Y \\ \Delta Y \end{pmatrix}$$

kjer

$$A = \begin{pmatrix} X'_1 & Y'_1 & 1 \\ X'_2 & Y'_2 & 1 \\ X'_3 & Y'_3 & 1 \end{pmatrix} \quad (6)$$

Zdaj lahko izračunamo koeficiente za izračun točk na zaslonu. Za razrešitev enačbe (6) obe strani pomnožimo z inverzno matriko A , tako dobimo enačbo (7).

$$\begin{pmatrix} \alpha_X \\ \beta_X \\ \Delta X \end{pmatrix} = A^{-1} \times \begin{pmatrix} X_1 \\ X_2 \\ X_3 \end{pmatrix} \text{ ter } \begin{pmatrix} \alpha_Y \\ \beta_Y \\ \Delta Y \end{pmatrix} = A^{-1} \times \begin{pmatrix} Y_1 \\ Y_2 \\ Y_3 \end{pmatrix} \quad (7)$$

Zdaj lahko izračunamo kalibracijske koeficiente.

Lokacijo kalibracijskih točk je treba tudi praktično izbrati [9]. Vemo, da ne smejo biti kolinearne med sabo. Paziti moramo tudi, da niso preveč blizu roba zaslona, kjer so nelinearnosti bolj verjetne. Zagotoviti moramo tudi, da so čim bolj oddaljene druga od druge, da zagotovimo natančnejše meritve. Na sliki 21 vidimo praktično izbiro točk, le-te se nahajajo približno 10% do 15% od roba zaslona.



Slika 21: Praktična izbira kalibracijskih točk.

4.1.1 Implementacija kalibracijskega algoritma

Princip je sledeč [21]:

1. Izberemo kalibracijske točke (X_k, Y_k) na zaslonu LCD, kjer je $k = 1, 2, \dots, n$ in $n \geq 3$.
2. Postopoma prikazujemo točke (X_k, Y_k) in se dotikamo zaslona ter odčitavamo točke (X'_k, Y'_k) , postopek ponovimo za vse k -je.
3. Izračunamo $\alpha_X, \alpha_Y, \beta_X, \beta_Y, \Delta X$ ter ΔY .

Za izračun $\alpha_X, \alpha_Y, \beta_X, \beta_Y, \Delta X$ ter ΔY izračunamo determinante sedmih matrik (priloga B), sestavljenih iz kalibracijskih točk na zaslonu LCD (X_k, Y_k) ter zbranih točk (X'_k, Y'_k) z zaslona na dotik.

Ker smo imeli opravka s precej velikimi števili (v enem primeru nam je vrednost ene determinante celo presegla 64-bitno vrednost tipa *long*), smo bili prisiljeni deliti vrednosti s 100.

4.2 Prikaz slike BMP

Za popestritev prikaza smo se odločili sestaviti namenske slike (v velikosti zaslona) ter napisati funkcijo za prikaz le-teh s kartice SD. Trenutno program podpira le slike BMP z barvno globino 24-bitov, shranjene v formatu Windows.

Obstaja več formatov datotek BMP, v nadaljevanju se bomo omejili na najbolj razširjeno verzijo Windows 3.x. Poglejmo bolj podrobno kako je format sestavljen [11].

Glava datoteke (ang. *file header*) – 14 bajtov:

- 2 bajta tipa datoteke (ang. *file type*) (tipična vrednost je "BM"),
- 4 bajti o velikosti datoteke, podano v bajtih,
- 2 bajta rezerviranega prostora,
- 2 bajta rezerviranega prostora,
- 4 bajti zamika – pove, kje se nahaja polje barv točk.

Bitna info glava (ang. *bitmap information header*) – 40 bajtov:

- 4 bajti o velikosti info glave (s pomočjo te ugotavljamo različne formate datotek BMP),
- 4 bajti o širini slike,
- 4 bajti o višini slike,
- 2 bajta števila barvnih ravnin (format BMP podpira le 1),
- 2 bajta BPP (ang. *bits per pixel*) vrednosti (v našem primeru 24),
- 4 bajti o kompresiji slike (v našem primeru je 0),
- 4 bajti o velikosti barvne tabele točk,
- 4 bajti o horizontalni resoluciji slike (podano v točkah na meter),
- 4 bajti o horizontalni resoluciji slike,
- 4 bajti o številu barv v barvni paleti (navadno je 0, kar predstavlja maksimalno število barv glede na BBP),
- 4 bajti o pomembnih barvah (navadno 0, ko so vse barve pomembne).

Barvna paleta (ang. *color palette*) - 4 bajti (opsijska, navadno je opuščena za 16- in več bitne datoteke BMP):

- 1 bajt za modro barvo,
- 1 bajt za zeleno barvo,
- 1 bajt za rdečo barvo
- 1 bajt rezerviranega prostora.

Polje barv točk (ang. *bitmap data array*):

- Polje je razporejeno po vrsticah, vsaka točka je sestavljena iz vrednosti R-G-B (3 bajt).
- Na koncu vsake vrstice se dodaja od 0 pa do 3 bajte "polnila" (ang. *padding*).
- Ta polnila so odvisna od dolžine vrstic. Velja pravilo, da se mora vsaka vrstica končati s številom bajtov, ki je večkratnik števila 4.

Velja še omeniti, da je zapis podatkov v datoteki BMP obraten – vrstice so podane od zadnje proti prvi, barva točk je podana kot B-G-R.

Naš algoritem za izris slike BMP na zaslon je preprost. Postopoma beremo datoteko bajt po bajtu ter luščimo ven podatke.

Ker imamo 24-bitno globino točk, na zaslonu pa izrisujemo le 16-bitno, moramo te vrednosti pretvoriti. Pretvorbo smo naredili tako, da smo nepotrebne bite LSB izloščili, ter ustrezno spojili preostale bite skupaj. Še prej pa moramo ustrezno maskirati bite, ki jih želimo obdržati (slika 22).

```
void Write24bitPixel(uint16_t R, uint16_t G, uint16_t B){
    LCD_WriteRAM((((R & 0xf8) << 8) | ((G & 0xfc) << 3) | ((B & 0xf8) >> 3));
}
```

Slika 22: Pretvorba v 16-bitni format.

4.3 Prikaz znakov ASCII

Za prikaz raznih podatkov smo si morali omisliti tudi prikaz teksta. Izbrana je bila fiksna velikost znakov 16 x 24 točk, podatki o točkah pa so predstavljeni v polju. Vsak znak je prikazan iz 24 16-bitnih vrednosti (slika 23), posamezni biti znotraj teh določajo, ali se bo določena točka obarvala ali ne (priloga C).

'A'

```
0x0000, 0x0380, 0x0380, 0x06C0, 0x06C0, 0x06C0, 0x0C60, 0x0C60,
0x1830, 0x1830, 0x1830, 0x3FF8, 0x3FF8, 0x701C, 0x600C, 0x600C,
0xC006, 0xC006, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000, 0x0000,
```

Slika 23: Tabela vrednosti za prikaz znaka "A".

4.4 Bresenhamov algoritem vzorčenja daljic

Kot vemo je daljica med dvema točkama definirana z zvezno funkcijo, to moramo ustrezno vzorčiti, v končno množico točk, za prikaz na zaslonu LCD.

Obstaja več načinov vzorčenja, najbolj preprosta je iteracija za vse vrednosti x_i ter izračun ustreznega y_i glede na enačbo (8), za vse točke, ki se nahajajo med krajišči daljice.

$$y_i = k \times x_i + b \quad (8)$$

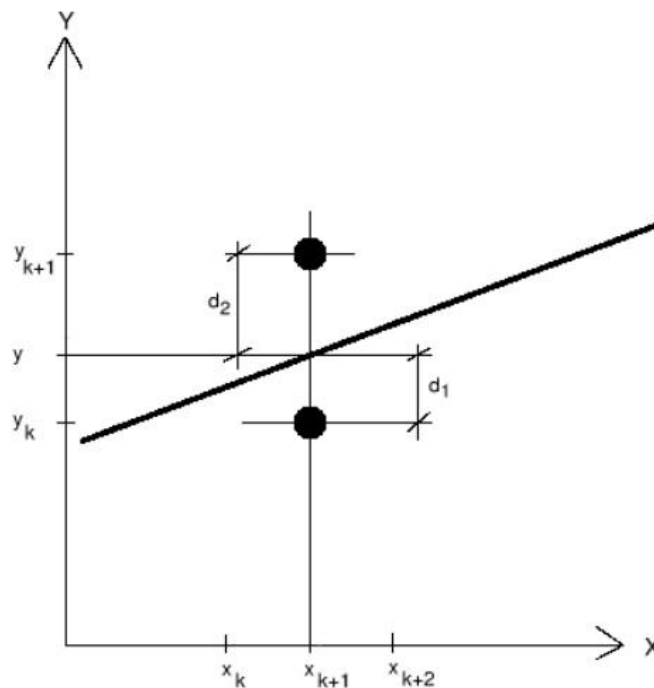
Ker govorimo o rasterskih zaslonih in imamo operacije v plavajoči vejici, moramo končne rezultate y_i zaokroževati.

Bresenhamov algoritem se izogne operacijam v plavajoči vejici. Osnovni princip algoritma je poiskati najbližjo točko idealni liniji daljice v dani iteraciji [12].

$$y = k \times x + b \quad (9)$$

Z enačbo (9) opišemo daljico, ki jo želimo vzorčiti. Definiramo tudi, da začetna točka T_1 leži levo od končne točke T_2 (kar pomeni, da daljica narašča po osi x) v našem koordinatnem sistemu, ter da za k velja: $0 \leq k \leq 1$.

Predpostavimo, da smo že izračunali željeno točko (x_k, y_k) , ki je najbližja idealni liniji daljice. Sedaj bi radi izračunali $(k + 1)$ točko (slika 24). Ker smo omejili naš koeficient naklona daljice vemo, da lahko izbiramo med dvema točkama: $(x_k + 1, y_k)$ ter $(x_k + 1, y_k + 1)$.



Slika 24: Oddaljenost točk (x_{k+1}, y_k) in (x_{k+1}, y_{k+1}) od idealne linije daljice.

Vrednosti d_1 ter d_2 sta definirani z enačbo (10).

$$d_1 = y - y_k = k \times (x_k + 1) + b - y_k \quad (10)$$

$$d_2 = (y_k + 1) - y = y_k + 1 - k \times (x_k + 1) - b$$

Vpeljemo novo spremenljivko p_k , kjer velja:

$$p_k = \Delta x \times (d_1 - d_2) \quad (11)$$

Ker velja $\Delta x = x_2 - x_1$ in ker smo predpostavili, da daljica narašča po x-osi vidimo da razlika $(d_1 - d_2)$ iz enačbe (11) tudi določa predznak same spremenljivke p_k

Če je točka $(x_k + 1, y_k)$ bližja idealni liniji potem velja da $d_1 < d_2$ in je vrednost spremenljivke p_k negativna.

Tako lahko definiramo našo y_{k+1} z enačbo (12):

$$y_{k+1} = \begin{cases} y_k & ; p_k < 0 \\ y_k + 1 & ; p_k \geq 0 \end{cases} \quad (12)$$

Zdaj pa želimo dobiti vrednost spremenljivke p_{k+1} v naslednjem koraku. Preuredimo enačbo (11), kjer velja da je $k = \Delta y / \Delta x$:

$$p_k = \Delta x \times (d_1 - d_2) = 2 \times \Delta y \times x_k - 2 \times \Delta x \times y_k + c \quad (13)$$

kjer s c označujemo konstanto, ki ne vpliva na predznak p_k . Izkoristimo enačbo (13) še za p_{k+1} , ter dobimo:

$$p_{k+1} = 2 \times \Delta y \times x_{k+1} - 2 \times \Delta x \times y_{k+1} + c \quad (14)$$

Od enačbe (14) odštejemo enačbo (13), velja tudi $x_{k+1} = x_k + 1$, dobimo (15):

$$p_{k+1} = p_k + 2 \times \Delta y - 2 \times \Delta x \times (y_{k+1} - y_k) \quad (15)$$

kjer velja:

$$y_{k+1} - y_k = \begin{cases} 0 & ; p_k < 0 \\ 1 & ; p_k \geq 0 \end{cases} \quad (16)$$

Iz enačb (15, 16) lahko zdaj zapišemo spremenljivko p_{k+1} , ki jo uporabimo za določitev točke v stolpcu x_{k+2} . Dobimo enačbo (17):

$$p_{k+1} = \begin{cases} p_k + 2 \times \Delta y & ; p_k < 0 \\ p_k + 2 \times \Delta y - 2 \times \Delta x & ; p_k \geq 0 \end{cases} \quad (17)$$

Zdaj nam preostane le še določiti začetno p_0 v točki (x_0, y_0) , vnesemo spremenljivke v enačbo (13), preuredimo in dobimo enačbo (18):

$$p_0 = 2 \times \Delta y - \Delta x \quad (18)$$

Kot vidimo so, če je p_0 celoštevilski, vsi nadaljnji izračuni prav tako celoštevilski.

Spomnimo se še, da smo postopek omejili na $0 \leq k \leq 1$. Če hočemo isti postopek uporabiti za $k > 1$ zadostuje zamenjava spremenljivk x ter y .

Za vrednosti $0 \geq k \geq -1$ pa velja, da nam y koordinata pada, zato preprosto zmanjšujemo y_{k-1} .

4.4.1 Implementacija algoritma

V programu imamo razširjen Bresenhamov algoritem, ki nam izrisuje daljico med poljubnima dvema točkama.

Postopek implementacije je sledeč:

1. Izračunamo Δx ter Δy (absolutna vrednost).
2. Definiramo začetne inkremente posameznih koordinat (štiri opcije), tako dejansko določimo smer izrisa.
3. Razrešimo koeficient naklona (ali velja $\Delta x \geq \Delta y$ ali obratno), definiramo začetni p_0 ter število iteracij.
4. V glavni zanki računamo p_k , ustrezno povečujemo x in y koordinato ter izrisujemo točko na zaslon.

4.4.2 Izris grafa

Omenjen algoritem smo izkoristili za izris grafa preteklih stanj temperature, zračne vlage, zračnega pritiska ter hitrosti vetra. Pretekle podatke shranjujemo na vsako uro v preprosto tabelo, pri izrisu pa v zanki postopoma podajamo koordinate dveh točk funkciji za izris daljice.

Pred izrisom moramo tudi pravilno pretvoriti (skalirati) vrednosti podatkov, da se ujemajo s koordinatami našega prikaza.

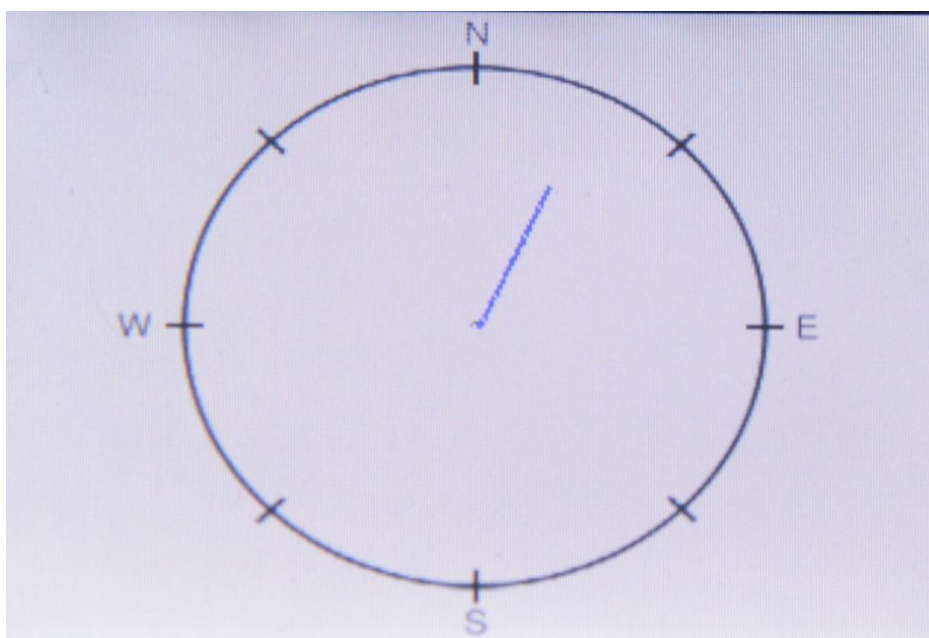
Na grafu prikazujemo podatke do 14 ur nazaj (slika 25), poleg samega grafa pa v zgornjem desnem kotu prikažemo tudi aktualne meritve.



Slika 25: Izris grafa temperature.

4.5 Prikaz in kalibracija smeri vetra

Odločili smo se prikazati preprosto vetrnico smeri neba. Na njej izrisujemo črto, ki nam predstavlja trenutno smer (slika 26).



Slika 26: Prikaz smeri vetra.

Za izris smeri (daljice) smo uporabili preproste rotacije nad referenčno daljico. Referenčna daljica je podana kot tabela koordinat. Nove koordinate izračunamo s pomočjo enačbe (19).

$$\begin{aligned}x' &= x \times \cos \Theta - y \times \sin \Theta \\y' &= x \times \sin \Theta + y \times \cos \Theta\end{aligned}\tag{19}$$

Preden uporabimo enačbo 19, moramo tudi naše koordinate (x, y) premakniti v koordinatno izhodišče.

Kot zasuka Θ dobimo iz branja senzorja vetrnice ter prištevkem kalibracijskega kota.

Kalibracijski kot dobimo z dotikom na zaslon znotraj vetrnice. Kot dobimo tako, da odčitamo koordinate dotika in jih prav tako premaknemo v koordinatno izhodišče ($x' = x - 160$ ter $y' = y - 120$), zatem izračunamo koeficient naklona k ter kot (20):

$$\Theta_{\text{offset}} = \tan^{-1} \left(\frac{y'}{x'} \right)\tag{20}$$

Pri tem moramo paziti, da v primeru, ko je koordinata $x' < 0$, našemu kotu Θ_{offset} prištejemo še 180° .

Izračun rotacije posameznih točk na referenčni daljici predstavlja precej potratno računanje v plavajoči vejici. Poleg tega nam tabela koordinat zaseda nepotreben pomnilniški prostor. V ta namen smo z enačbo (17) rotirali le zadnjo točko referenčne daljice, za izris daljice pa preprosto uporabili Bresenhamov algoritem med dvema točkama.

5 SKLEP

Cilj diplomske naloge je bil zasnovati in izdelati preprosto vremensko postajo za lastno uporabo. Postajo sestavljata dva modula, oddajniški ter sprejemniški. Na oddajniškem modulu periodično odčitavamo senzorje, le-te ustrezno obdelujemo in preko RF povezave pošljemo na sprejemniški modul. Na sprejemniškem modulu te podatke naprej obdelamo in predstavimo na zaslonu LCD. Poleg tega nam ta modul omogoča tudi prikaz preteklih stanj do 14 ur nazaj v obliki grafa, prikaz smeri vetra z možnostjo kalibracije le-tega znotraj vmesnika ter prikaz dveh dodatnih vrednosti temperatur, ki jih človeško telo zaznava poleti oziroma pozimi.

Na trgu je precej podobnih "hišnih" vremenskih postaj. Od cenovno ugodnih s funkcijami temperature ter zračne vlage, do dražjih, ki vključujejo vse meritve, ki jih beleži tudi naša. Manj pa je takih, ki vključujejo na dotik občutljiv zaslon z lastno prirejenim uporabniškim vmesnikom.

Pri razvoju se je pojavilo nekaj težav, ki so posledice mojega neznanja s področja elektronike kar je tudi vplivalo na čas izpeljave projekta. Pri programiranju mikrokrmilnikov sem že imel izkušnje (s programiranjem krmilnikov Atmel AVR), vendar je STM32 vseeno predstavljal velik skok v zahtevanem znanju iz tega področja. Pri RF modulu RFM12B pa je bila težava tudi v pomankljivi dokumentaciji.

Vseeno menim, da nam je v nalogi uspelo doseči zastavljene cilje.

Če odpišemo omejitev Kickstarter verzije programskega okolja IAR je še precej prostora za razširitve in optimizacijo sistema. Na oddajniški modul bi lahko priključili še ostale senzorje (merilnik količine padavin, merilnik svetlobe in UV sevanja, ...) na sprejemniškem modulu pa vzpostavili Ethernet povezavo ter preprost spletni strežnik za deljenje meritev s spletom.

Namesto našega uporabniškega vmesnika bi lahko uporabili eno izmed RTOS (*ang. Real-time operating system*) rešitev (kot so FreeRTOS, ChibiOS/RT, CoCoX CoOS, in drugi). Prednost operacijskih sistemov v realnem času je v več-opravnosti, to bi lahko recimo izkoristili pri (navidezno) istočasnem izvajanju prikaza in sprejemanja novih podatkov ter morebitnim spletnim strežnikom.

Literatura in viri

- [1] (2010) ARM Cortex M4 Technical reference manual. Dostopno na:
http://infocenter.arm.com/help/topic/com.arm.doc.ddi0439c/DDI0439C_cortex_m4_r0p1_trm.pdf

- [2] (2009) Bosch, BMP085 Digital Pressure Sensor. Dostopno na:
<https://www.sparkfun.com/datasheets/Components/General/BST-BMP085-DS000-05.pdf>

- [3] (2013) Calculating the length of an Antenna. Dostopno na:
<http://www.air-stream.org/technical/calculating-length-antenna>

- [4] (2010) DHT11 Humidity & Temperature Sensor. Dostopno na:
<http://www.micro4you.com/files/sensor/DHT11.pdf>.

- [5] (2013) Element 14, STM32F4Discovery Expansion Boards. Dostopno na:
http://www.element14.com/community/community/knode/dev_platforms_kits/element14_dev_kits/stm32f4-discovery-expansion-boards

- [6] (2013) FatFs - Generic FAT File System Module, R0.09b. Dostopno na:
http://elm-chan.org/fsw/ff/00index_e.html

- [7] (2006) Hope RF, RF12B programming guide. Dostopno na:
http://www.hoperf.com/upload/rf/RF12B_code.pdf

- [8] (2006) Hope RF, RFM12B universal ISM band FSK transceiver. Dostopno na:
<http://www.hoperf.com/upload/rf/rfm12b.pdf>

- [9] (2013) How to calibrate touch screens. Dostopno na:
<http://www.embedded.com/design/system-integration/4023968/How-To-Calibrate-Touch-Screens>

- [10] (2007) Infineon Technologies, TLE4905L. Dostopno na:
http://www.produktinfo.conrad.com/datenblaetter/150000-174999/153775-da-01-en-HALL_SENSOR_TLE_4935L.pdf

- [11] (2013) Microsoft Windows Bitmap File Format Summary. Dostopno na:
<http://www.fileformat.info/format/bmp/egff.htm>
- [12] P. Bulić, Osnovni postopki v računalniški grafiki, 2006. Dostopno na:
<http://www.fri.uni-lj.si/file/58846/tsnaxldwxb.pdf>
- [13] P. Bulić, Prosojnice za predmet Vgrajeni sistemi, 2012. Dostopno na:
<https://ucilnica.fri.uni-lj.si/course/view.php?id=39>
- [14] (2013) Resistive Touch Screen. Dostopno na:
<http://resistivetouchscreen.org/>
- [15] (2009) Solomon Systech, SSD2119. Dostopno na:
<http://www.crystalfontz.com/controllers/SSD2119.pdf>
- [16] (2013) STMicroelectronics, RM0090 Reference Manual. Dostopno na:
http://www.st.com/web/en/resource/technical/document/reference_manual/DM00031020.pdf
- [17] (2013) STMicroelectronics, STM32F4DISCOVERY board firmware package. Dostopno na:
<http://www.st.com/web/en/catalog/tools/PF257904>
- [18] (2011) STMicroelectronics, STMPE811. Dostopno na:
<http://www.st.com/web/en/resource/technical/document/datasheet/CD00186725.pdf>
- [19] (2012) STMicroelectronics, UM1472 User Manual. Dostopno na:
http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/DM00039084.pdf
- [20] (2011) STMicroelectronics, UM1467 User Manual. Dostopno na:
http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/DM00037368.pdf
- [21] (2007) Texas Instruments Incorporated, Calibration in Touch-Screen Systems. Dostopno na:
<http://www.ti.com/lit/an/slyt277/slyt277.pdf>

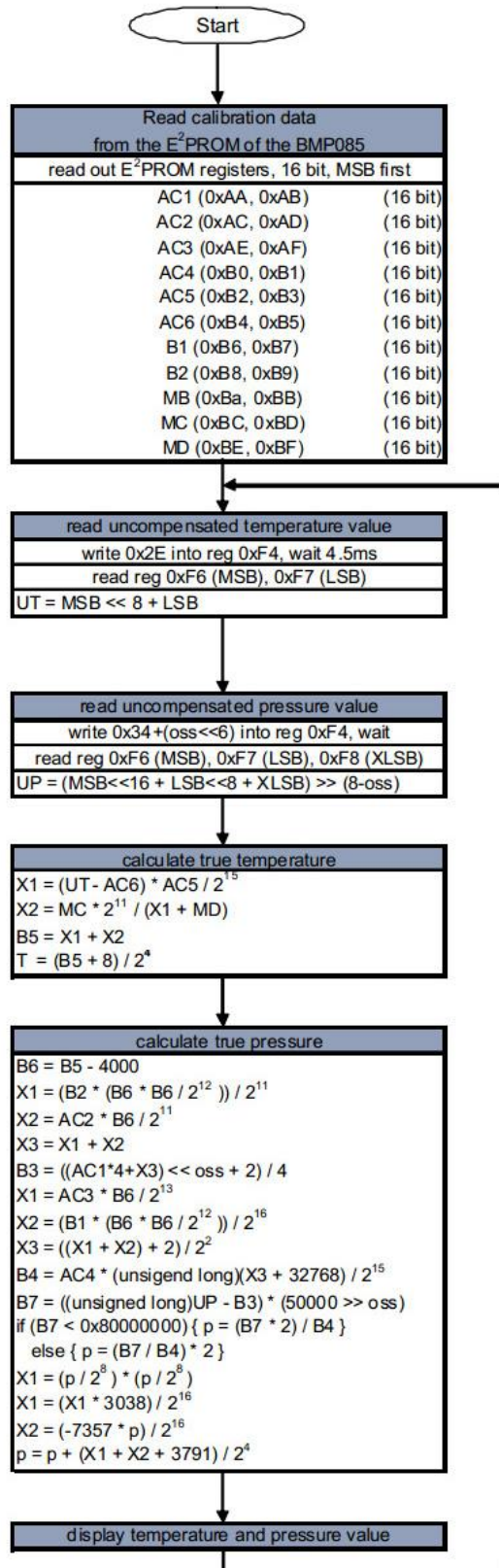
Kazalo slik

Slika 1: Razvojna plošča STM32F4 Discovery proizvajalca STMicroelectronics.	7
Slika 2: Razširitevno vezje STM32F4DIS-BB proizvajalca Embest Technology	10
Slika 3: STM32F4DIS-LCD.....	10
Slika 4: Shema povezave RFM12B s STM32F4 Discovery	12
Slika 5: Shema povezave DHT11 s STM32F4 Discovery	13
Slika 6: Shema povezave BMP085 s STM32F4 Discovery	14
Slika 7: Shema vezave Hallovega senzorja s STM32F4 Discovery.....	15
Slika 8: Shema delovanja oddajniškega modula.	18
Slika 9: Shema delovanja sistema.	19
Slika 10: Odsek urinega diagrama [16].	20
Slika 11: Časovni diagram komunikacije senzorja DHT11.	21
Slika 12: Preprosta bralna zanka visokega stanja enega signala.	21
Slika 13: I ² C protokol.....	22
Slika 14: Trije načini pošiljanja 18-bitne barvne globine.....	25
Slika 15: Prikaz večanja naslovnega števca grafičnega pomnilnika.	26
Slika 16: Podatkovni okvir.	28
Slika 17: Vsebina polja "podatki".	28
Slika 18: Vsebina polja "meritve senzorjev".	28
Slika 19: Glavni prikaz podatkov.	31
Slika 20: Mehanska nepravilnost pri zaslonih na dotik.	32
Slika 21: Praktična izbira kalibracijskih točk.	34
Slika 22: Pretvorba v 16-bitni format.	36
Slika 23: Tabela vrednosti za prikaz znaka "A".	36
Slika 24: Oddaljenost točk $(xk + 1, yk)$ in $xk + 1, yk + 1$ od idealne linije daljice.....	37
Slika 25: Izris grafa temperature.	40
Slika 26: Prikaz smeri vetra.....	40

Kazalo prilog

Priloga A: BMP085 Izračun temperature in zračnega pritiska	1
Priloga B: Determinante kalibracijskih matrik	2
Priloga C: Prikaz tabele za izris črke "A"	3

Priloga A: BMP085 Izračun temperature in zračnega pritiska



example:

C code function: type:

bmp085_get_cal_param

AC1 =	408	short
AC2 =	-72	short
AC3 =	-14383	short
AC4 =	32741	unsigned short
AC5 =	32757	unsigned short
AC6 =	23153	unsigned short
B1 =	6190	short
B2 =	4	short
MB =	-32768	short
MC =	-8711	short
MD =	2868	short

bmp085_get_ut

UT = **27898** long

oss = 0
= oversampling_setting
(ultra low power mode)

short (0 .. 3)

bmp085_get_up

UP = **23843** long

bmp085_get_temperature

X1 = 4743 long

X2 = -2344 long

B5 = 2399 long

T = **150** temp in 0.1°C long

BMP085_calpressure

B6 = -1601 long

X1 = 1 long

X2 = 56 long

X3 = 57 long

B3 = 422 long

X1 = 2810 long

X2 = 59 long

X3 = 717 long

B4 = 33457 unsigned long

B7 = 1171050000 unsigned long

p = 70003 long

X1 = 74529 long

X1 = 3454 long

X2 = -7859 long

p = **69964** press. in Pa long

Priloga B: Determinante kalibracijskih matrik

$$\alpha_x = \Delta_{x1}/\Delta, \beta_x = \Delta_{x2}/\Delta, \Delta X = \Delta_{x3}/\Delta$$

$$\alpha_y = \Delta_{y1}/\Delta, \beta_y = \Delta_{y2}/\Delta, \text{ and } \Delta Y = \Delta_{y3}/\Delta$$

$$\Delta = \det(A) = \begin{vmatrix} X'_1 & Y'_1 & 1 \\ X'_2 & Y'_2 & 1 \\ X'_3 & Y'_3 & 1 \end{vmatrix} = (X'_1 - X'_3) \times (Y'_2 - Y'_3) - (X'_2 - X'_3) \times (Y'_1 - Y'_3)$$

$$\Delta_{x1} = \det(A_{x1}) = \begin{vmatrix} X_1 & Y'_1 & 1 \\ X_2 & Y'_2 & 1 \\ X_3 & Y'_3 & 1 \end{vmatrix} = (X_1 - X_3) \times (Y'_2 - Y'_3) - (X_2 - X_3) \times (Y'_1 - Y'_3)$$

$$\Delta_{x2} = \det(A_{x2}) = \begin{vmatrix} X'_1 & X_1 & 1 \\ X'_2 & X_2 & 1 \\ X'_3 & X_3 & 1 \end{vmatrix} = (X'_1 - X'_3) \times (X_2 - X_3) - (X'_2 - X'_3) \times (X_1 - X_3)$$

$$\Delta_{x3} = \det(A_{x3}) = \begin{vmatrix} X'_1 & Y'_1 & X_1 \\ X'_2 & Y'_2 & X_2 \\ X'_3 & Y'_3 & X_3 \end{vmatrix} = X_1 \times (X'_2 Y'_3 - X'_3 Y'_2) - X_2 \times (X'_1 Y'_3 - X'_3 Y'_1) + X_3 \times (X'_1 Y'_2 - X'_2 Y'_1)$$

$$\Delta_{y1} = \det(A_{y1}) = \begin{vmatrix} Y_1 & Y'_1 & 1 \\ Y_2 & Y'_2 & 1 \\ Y_3 & Y'_3 & 1 \end{vmatrix} = (Y_1 - Y_3) \times (Y'_2 - Y'_3) - (Y_2 - Y_3) \times (Y'_1 - Y'_3)$$

$$\Delta_{y2} = \det(A_{y2}) = \begin{vmatrix} X'_1 & Y_1 & 1 \\ X'_2 & Y_2 & 1 \\ X'_3 & Y_3 & 1 \end{vmatrix} = (X'_1 - X'_3) \times (Y_2 - Y_3) - (X'_2 - X'_3) \times (Y_1 - Y_3)$$

$$\Delta_{y3} = \det(A_{y3}) = \begin{vmatrix} X'_1 & Y'_1 & Y_1 \\ X'_2 & Y'_2 & Y_2 \\ X'_3 & Y'_3 & Y_3 \end{vmatrix} = Y_1 \times (X'_2 Y'_3 - X'_3 Y'_2) - Y_2 \times (X'_1 Y'_3 - X'_3 Y'_1) + Y_3 \times (X'_1 Y'_2 - X'_2 Y'_1)$$

