

[Športna](https://github.com/ZanRescic/STM32H750B-DK_BouncingBall) ura

Poročilo za projektno nalogo pri predmetu Vhodno-izhodne naprave

Avtor: Žan Reščič, 63220274; Matija Baša 63220014

Mentor: viš. pred. dr. Robert Rozman

Datum: 12.9.2024

# UVOD

Za VIN projekt sva se odločila narediti prototip športne ure, katera s pomočjo pulznega oksimetra meri srčni utrip in kisik v krvi (Sp02) in to prikazuje na zaslon. Uporabila sva mikrokontroler STM32H750B-DK, ploščico Oximeter 5 Click, na kateri je senzor MAX30102, ki vsebuje integriran pulzni oksimeter in TFT LCD zaslon z GC9A01 gonilnikom. Projekt je bil spisan v STM32CubeIDE.

Slika, ki vsebuje besede elektronsko inženirstvo, elektronska komponenta, elektronika, pasivna komponenta vezja

Opis je samodejno ustvarjen Slika, ki vsebuje besede elektronika, elektronska komponenta, elektronsko inženirstvo, pasivna komponenta vezja

Opis je samodejno ustvarjen

Slika, ki vsebuje besede elektronika, elektronsko inženirstvo, kabel, električna napeljava

Opis je samodejno ustvarjen Slika, ki vsebuje besede elektronika, besedilo, elektronsko inženirstvo, kabel

Opis je samodejno ustvarjen

Slike meritev so le simbolične in ne dejansko izmerjene vrednosti z uporabo Oximeter 5 click.

OXIMETER 5 CLICK

# PREDSTAVITEV SENZORJEV IN NAPRAV

Oximeter 5 click vsebuje senzor MAX30102, ki je integriran pulzni oksimeter in monitor srčnega utripa. Vključuje notranje LED diode, detektor svetlobe, optične elemente in nizko šumno elektroniko z zavrnitvijo svetlobe okolice. Deluje z enim samim napajalnikom 1,8 V, pridobljenim iz obeh napajalnih pinov mikroBUS-a za notranje LED diode. Komunicira prek standardnega vmesnika, združljivega z I2C.

Deluje tako, da ima MAX30102 integrirane Red in IR (Infra red) LED diode katere oddajajo svetlobo, ki se vpije oz. odbije od kože. Vsebuje tudi detektor svetlobe kateri lovi odbitke svetlobe od kože preko katere se nato izračuna Sp02 in srčni utrip. Kri, ki vsebuje več kisika absorbira več IR svetlobe in odbija več Red svetlobe. Kri, ki vsebuje manj kisika pa absorbira več Red svetlobe in odbija več IR svetlobe. Takšni tehniki pravimo fotopletizmografija (photoplethysmography- PPG).

Slika, ki vsebuje besede elektronika, vezje, besedilo, elektronsko inženirstvo

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, številka, pisava

Opis je samodejno ustvarjen

# KOMUNIKACIJA

Z Oximeter 5 Click-om komuniciramo preko I2C (Inter-Integrated Circuit) protokola. Deluje tako, da uporablja dve žici za komunikacijo in sicer SCL (Serial Clock Line) za časovno usklajevanje in SDL (Serial Data Line) za prenos podatkov. Komunikacija poteka med master napravo in slave slave napravo. V našem primeru je master STM32H750B-KD in slave Oximeter 5 Click.

# INICIALIZACIJA

Za inicializiacijo uporabljamo metodo **oximeter5\_init()**, katera preveri, če je naprava dosegljiva na svojem naslovu (0x57 oz. 0x AE, ko naslov shift-amo za 1 v levo). Če je izpiše, da je naprava pripravljena, drugače izpiše da ni.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, programska oprema

Opis je samodejno ustvarjen

V metodi **oximeter5\_default\_cfg()** resetiramo programsko opremo, registre in kazalce za fifo buffer, registre prekinitev, amplitude LED diod. To naredimo tako, da z metodo **oximeter5\_generic\_write()** zapišemo na naslove želenih registrov določeno vrednost.

Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema

Opis je samodejno ustvarjen

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, programska oprema

Opis je samodejno ustvarjen

# BRANJE IN PISANJE

Branje in pisanje je implementirano z uporabo metod **oximeter5\_generic\_write()** in **oximeter5\_generic\_read()**. Write deluje tako, da kliče metodo **HAL\_I2C\_Master\_Transmit()**, katera pošilja podatke slave-om. Prejme parametre hi2c4 ročico, naslov Oximetra, podatke za prenos, število podatkov, ki se bo preneslo in časovno omejitev.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Read deluje podobno, le da prvo kliče write, da pove določenemu registru, da bo iz njega brala podatke in nato kliče metodo **HAL\_I2C\_Master\_Receive()** katera prejme podatke. Parametri metode so enaki kot pri write, le da tukaj se podatki sprejemajo v write pa pošiljajo.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

# MERJENJE Sp02 IN SRČNEGA UTRIPA

Za merjenje najprej kličemo metodo oximeter5\_read\_sensor\_data(), katera iz senzorja bere z hitrostjo 25 smp/s (samples/seconds) štiri sekunde, tako da prejme 100 podatkov. Podatki se hranijo v FIFO bufferjih kar pomeni, da je hranjenim samo prvih 100 podatkov. Imamo dva bufferja in sicer RED buffer in IR buffer.

Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema

Opis je samodejno ustvarjen

Ko imamo podatke v bufferjih jih lahko začnemo procesirati za Sp02 in srčni utrip. Za meritev Sp02 kličemo metodo **oximeter5\_get\_oxygen\_saturation()**. Metoda podatke v bufferjih procesira s pristopom digitalnega procesiranja signalov (DPS). Odstraniti mora vrhove v signalu kateri so preveč blizu, odstraniti mora prenizke vrhove kateri bi pokvarili rezultat in jih sortirati (koda metode je na github repozitoriju, ker je preveč dolga).

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Za meritev srčnega utripa uporabljamo metodo **oximeter5\_get\_heart\_rate()**. Metoda najprej podatke v bufferjih pripravi za procesiranje in nato jih s pristopom digitalnega procesiranja signalov pripravi v sprejemljive za izračun srčnega utripa (koda metode je na github repozitoriju, ker je preveč dolga).

Slika, ki vsebuje besede besedilo, posnetek zaslona

Opis je samodejno ustvarjen

# POMOŽNE METODE

Metode, katere smo uporabili za procesiranje surovega signala so **dev\_find\_peaks()**, katere najde prave vrhove v signalu. Pomožni metodi, ki jih uporabi sta **dev\_peaks\_above\_min\_height()** in **dev\_remove\_close\_peaks()**. Prva odstrani vrhove, kateri so prenizki za pravilen izračun tako Sp02, kot srčnega utripa.

Slika, ki vsebuje besede besedilo, posnetek zaslona, programska oprema

Opis je samodejno ustvarjen

Drugo metodo, katero uporabi odstrani vrhove signala, kateri so si med seboj preblizu.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Uporabljene so še sortirne metode, katere uredijo podatke naraščajoče oz. padajoče.

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

LCD ZASLON

# OPIS NAPRAVE

Naprava, na katero bomo izpisovali rezultate je okrogel LCD zaslon, ki ima naslednje specifikacije:

* Velikost 1.28 palčni oziroma 3.25cm
* Ločljivost: 240x240
* IC: GC9A01

# POVEZAVA

Naprava ima 6 priključitvenih pinov in sicer:

* VCC, povežemo ga na napajanje 3.3V
* GND, povežemo ga na ozemljitveni pin na naši napravi
* SCL, povežemo ga na urin signal
* SDA, povežemo na master out slave in
* DC, povežemo na GPIO output
* CS, povežemo na GPIO output
* RST, povežemo na GPIO output

# INICIALIZACIJA NA STM32H7

Za upravljanje zaslona sem uporabil STM32H7 board. Za komunikacijo bomo uporabili SPI. Zaslon na STM23H7 povežemo na naslednji način za privzeto inicializiran SPI2:

* VCC -> 3v3 pin na H7
* GND -> GND pin na H7
* SCL -> PD3 pin na H7 (SCK/D13)
* SDA -> PB15 pin na H7 (PWM/MOSI/D11)
* DC -> katerikoli prosti GPIO pin, jaz sem izbral PE6
* CS -> katerikoli prosti GPIO pin, jaz sem izbral PI8
* RST -> katerikoli prosti GPIO pin, jaz sem izbral PE3

Ko sem imel napravo uspešno povezano z kabli sem moral še projekt inicializirati v .ioc datoteki v CubeIDE. Najprej sem inicializiral PGIO pine potrebne za komunikacijo. V pinout sem inicializiral 3 GPIO pine kot GPIO\_Output in sicer DC, CS, RST. Poimenoval sem jih LCD\_DC\_R, LCD\_CS\_R, LCD\_RST\_R. Poimenovanje sicer ni tako pomembno mi je pa bilo lažje potem v kodi da sem vedel kateri pin je kateri. Nato sem pod Connectivity zavihkom izbral SPI2. Tukaj sem najprej spremenil data size iz 4b na 8b. Nato sem moral še zmanjšati boud rate, saj je bil privzeto zelo previsok. Zmanjšal sem ga na 12MBits/s. Kot zadnje sem inicializiral še DMA. Dodal sem SPI2\_RX kot SPI2\_TX. DMA sicer nebi bil obvezen za upravljanje naprave, ampak bi bila komunikacija bistveno prepočasna za uporabo zaslona.

# IZPISOVANJE NA ZASLON

Za izpisovanje sem si pomagal z minimalnim driverjem za GC9A01. Za začetek je potrebno dodati pine, ki smo jih inicialzirali v .ioc datoteki. To naredimo na naslednji način:

Slika, ki vsebuje besede besedilo, pisava, posnetek zaslona

Opis je samodejno ustvarjen

Tukaj imamo nizka in visoka stanja za vse 3 pine. Imena so tukaj ista kot v pinout.

Za pisanje uporabimo naslednjo funkcijo:  
Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava, vrstica

Opis je samodejno ustvarjen

Za zapis uporabimo HAL\_SPI\_Transmit\_DMA() jas je tako najhitrejše. Zanko na koncu potrebujemo zato, da vemo kdaj je SPI pripravljen za naslednji zapis.

Za izbiro naslovnega okna oziroma izbire na kateri del zaslona bomo risali uporabimo naslednjo funkcijo:

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Write\_Cmd(0x2a) in Write\_Cmd(0x2b), ki jim potem sledijo podatki za začetek in konec nam x in y koordinate okna. Nato pa imamo še Write\_Cmd(0x2c), ki nam pove, da bomo od sedaj naprej pisali pixle na zaslon, se pravi vsi podatki, ki jih bomo poslali bodo interpretirani kot podatki za prižiganje pixlov na območju, ki smo ga definirali.

Za dejansko prižiganje pixlov pa uporabimo naslednje funkcije:  
Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Funkcija Write\_Data\_U16() nam razdeli 16bitno integer število na dva 8bitnadela in jih pošlje v funkcijo Write\_Data(). Tukaj najprej chip select nastavimo na nizko stanje in LCD\_DC na visoko kar določi, da bomo pošiljali podatke in ne ukaze. Nate z prejšnje implementirano funkcijo pošljemo oba dela in postavimo CS na visoko stanje.

Sedaj če bi želeli celoten zaslon spremeniti v eno barvo bi uporabili naslednjo kodo:  
Slika, ki vsebuje besede besedilo, pisava, posnetek zaslona, vrstica

Opis je samodejno ustvarjen

Parameter, ki ga funkcija dobi je barva, ki je v formatu RGB565 (5bit rdeča, 6bit zelena in 5bit modra). Ta funkcija sicer deluje, je pa počasna, saj moramo vsak pixel posebej nastaviti.

To bi pohitrili z DMA in sicer da namesto vsakega pixla posebej pošljemo cel buffer.  
Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

To funkcijo pa lahko kličemo na naslednji način:

Slika, ki vsebuje besede besedilo, posnetek zaslona, pisava

Opis je samodejno ustvarjen

Ta funkcija bistveno pohitri delovanje, saj ne pošiljamo vseh 240\*240 pixlov, vendar pošiljamo po kosih velikih 512 bajtov. Na tak način lahko na zaslon izpisujemo kar želimo, le da je ta način bistveno hitrejši.

Ko imamo vse pripravljeno, lahko te funkcije kličemo na naslednji način:  


Prva funkcija inicializira zaslon, da je pripravljen za komunikacijo, druga pa je funkcija, ki kot zgoraj nastavi cel zaslon na eno barvo, v tem primeru bela.

# GITHUB

Na naslednji povezavi se nahaja naše celoten projekt, ki vključuje datoteke oximeter5custom.c, ki vključuje vso kodo povezano z Oximeter 5 click. In gc9a01.c, ki vključuje vso kodo povezano z LCD zaslonom.

https://github.com/ZanRescic/VIN\_Oximeter\_5\_Click