TEHNIČKO VELEUČILIŠTE U ZAGREBU

PREDDIPLOMSKI STRUČNI STUDIJ MEHATRONIKE

Mihael Žugec

Tahometar

Seminarski rad br. 3

Zagreb, Siječanj 2024.

TEHNIČKO VELEUČILIŠTE U ZAGREBU

PREDDIPLOMSKI STRUČNI STUDIJ MEHATRONIKE

Mihael Žugec

JMBAG: 0246104197

Tahometar

Seminarski rad br. 3

Zagreb, Siječanj 2024.

Sažetak seminarskog rada

Tema seminarskog rada je bio uređaj za mjerenje broja okretaja u vremenskom rasponu, tahometar. Uređaj se sastojao od *Nucleo STM32F303K8* mikro upravljača, infra-crvenog senzora udaljenosti i 128x64p OLED displaya ukupne cijene od otprilike 20 €. Rezultat se dobivao mjerenjem vremenskih raspona između dva prolaska referentne točke ispred senzora pomoću *timera* i *interupta*. Ta vrijednost bi se pretvorila u kružnu frekvenciju te poslala na OLED display pomoću I^2C protokola serijske komunikacije.

Ključne riječi: Vremenski raspon, serijska komunikacija, kružna frekvencija, infra-crveni senzor, *timer*, *interupt, rotirajuće tijelo*

Sadržaj

Popis kratica

CAN – eng. *Controller Area Network*

CCR - eng. *Counter Compare Register*

GPIO - eng. *General Purpose Input Output*

I2C - eng. *Inter-Intergarted Circuit*

LED - eng. *Light Emitting Diode*

OLED - eng. *Organic Light-Emitting Diode*

RAM - eng. *Ready Access Memory*

SCL - eng. *Serial clock*

SDA - eng. *Serial data*

SPI - eng. *Serial Peripheral Interface*

USART - eng. *Universal Synchronous/Asynchronous Receiver/Transmitter*

Popis tablica

Popis slika

1. Uvod

Tahometar je uređaj za mjerenje brzine vrtnje nekog rotirajućeg tijela i koristi se u mnogim uređajima i strojevima svakodnevice. Mjerenje brzine vrtnje je bitno u vozilima kako bi vozač znao stanje rada motora, ili brzinu kretanja vozila pomoću brzinomjera koja se dobije pretvaranjem okretaja kotača u pravocrtnu brzinu. Neka vozila, uglavnom teretna, uz brzinomjer i tahometar okretaja motora imaju i tahograf uređaj koji bilježi brzinu kretanja vozila kroz vrijeme i on izravno ne utječe na korištenje vozila nego služi za legalne svrhe. Osim vozila mjerenje okretaja je vrlo bitno i kod raznih motora za pumpe ili ventilatore, generatora snage, turbina, brzine vrtnje alata kod alatnih strojeva itd.

Uređaj u ovom seminarskom radu je tahometar koji može mjeriti brzinu vrtnje bilo kojeg rotirajućeg tijela, ali je uglavnom zamišljen za provjeru okretaja npr. elektromotora i ručnih alata kao bušilica ili brusilica. Dodavanjem baterije uređaj može biti prijenosan.

Temeljna komponenta ovog uređaja je mikro upravljač *Nucleo STM32F303K8* pa se stoga mjerenje obavlja digitalno. Na objekt mjerenja se stavi mjerna traka te se on počne vrtjeti, mikro upravljač ima infra-crveni senzor koji bilježi svaki prolaz mjerne trake ispred njega te pomoću unutrašnjeg *timera* računa vremenski interval između pojedinačnih prolazaka. Na kraju se ta vrijednost preračuna u kružnu frekvenciju i prikaže na OLED ekranu.

2. Teorijski opis

Vrijednost mjerenja tahometra se najčešće izražava u okretajima po minuti (min^{-1}; o/min), ili se iz te vrijednosti pretvori u drugu željenu vrijednost. Postoji više načina mjerenja, analognih i digitalnih, najčešći bi bili dva diska u plošnom kontaktu gdje je jedan disk spojen na rotirajući objekt, a drugi na kazaljku i trenje između diskova uzrokuje pomicanje kazaljke, najčešći digitalni oblik, korišten također u ovom radu, je infra-crveni senzor koji mjeri vremenske intervale prolaska mjerne trake ispred njega i to pretvara u okretaje po minuti. Postoje i drugi načini mjerenja kao mjerenje brzine vrtnje *enkodera* spojenog na mjerni objekt tarnim spojem pomoću kotačića.

Uređaj u seminarskom radu *koristi Nucleo STM32F303K8* mikro upravljač, infra-crveni senzor udaljenosti temeljen na *LM393* komparatoru i *SSD1306* OLED ekran kao glavne komponente. Za mjerenje je na mjerni objekt potrebno postaviti komad mjerne trake koja se koristi za dobivanje vrijednosti vremenskih intervala okretaja. Princip rada uređaja se temelji na mjerenju tih vremenskih intervala između dva prolaza mjerne trake ispred senzora i preračunavanja te vrijednosti u okretaje po minuti prema jednadžbi (1). Simbol ω označava kružnu frekvenciju u okretajima po minuti, a T vremenski interval između dva signala u milisekundama.

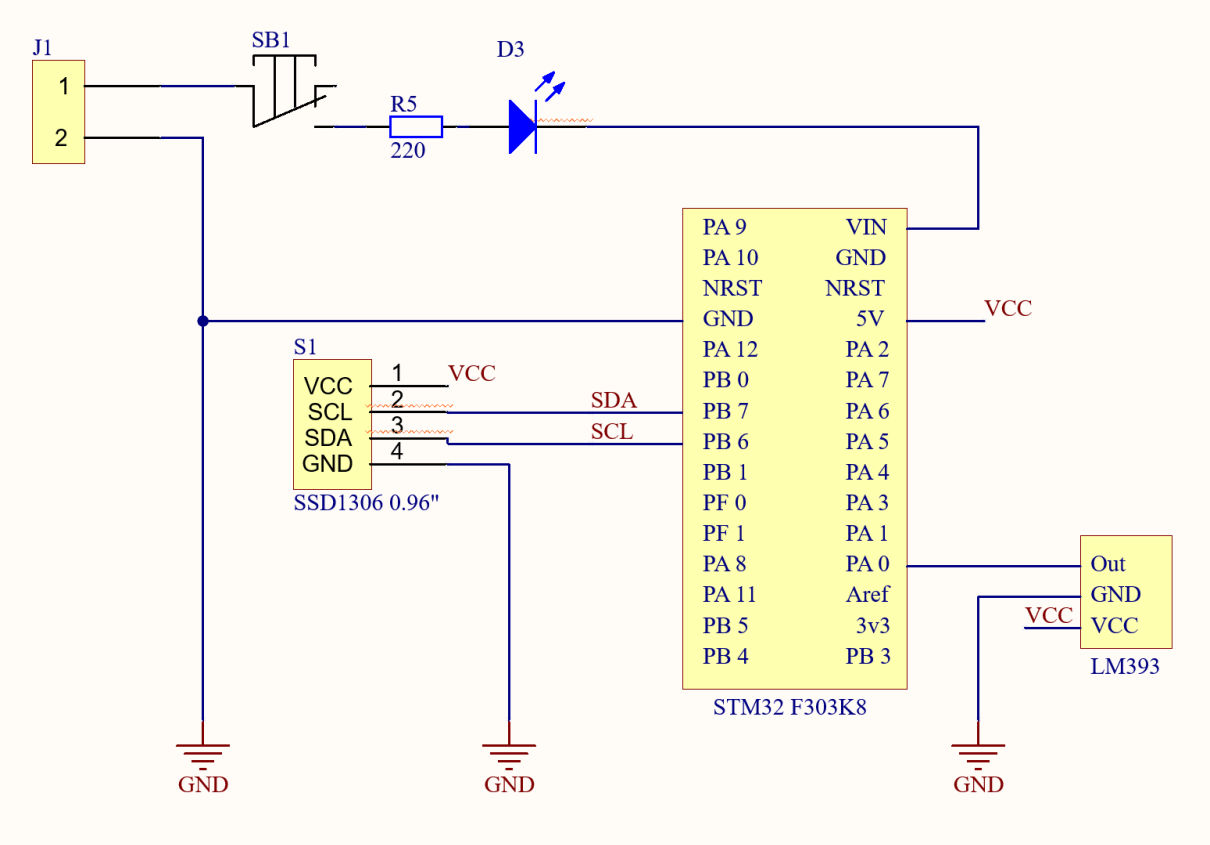
$$\omega = \frac{60\,000}{T} \quad (1)$$

Prescaler za *timer* 2 ima vrijednost od 7 999 što znači da se od osnovne frekvencije od 16 MHz dobije frekvencija mjerenja od 2 kHz, to jest rezolucija od 500 µs. Također timer broji do 65 535 pa bi najveći vremenski interval između dva prolaza mjerne trake bio 32 767,5 ms, a najmanji od 0,5 ms. Preko tih vrijednosti intervala se teoretski može mjeriti najniža kružna frekvencija od ~2 min^{-1} i najviša od 120 000 min^{-1}.

Tokom cijelog mjerenja *timer* mikro upravljača broji i svaki put kada mjerna traka prođe ispred senzora se aktivira prekidna funkcija koja zapisuje vrijednost *timera* u varijablu „T1“ te potom „T2“ pomoću *Input Capture Mode*. Zatim se varijabla „T“ dobiva oduzimanjem „T1“ od „T2“ i množenjem s 0,5 kako bi se dobile milisekunde

za korištenje u jednadžbi (1). Na kraju se varijabla „W“, kružna frekvencija u obliku cijelog broja, pretvara u tekst i ispisuje na ekranu uređaja.

S ovim komponentama postoji i drugi način dobivanja vrijednosti kružne frekvencije, a to je mjerenje broja prolazaka mjerne trake unutar nekog vremenskog intervala, npr. 60 s čime bi se izravno dobila vrijednost u okretajima po minuti. Ovaj pristup je jednako jednostavan kao i izabrani, ali mana mu je što se za dobivanje vrijednosti mjerenja treba čekati cijela minuta, dok kod izabranog postupka rezultat se dobije praktički odmah.



Slika 1 Shema spajanja tahometra

Mikro upravljač je napajan pomoću *Micro USB* kabela. Izlaz senzora je spojen je na nožicu PA 0 koja aktivira *interupt* protokol za TIM2 preko *Input capture mode*. Kod serijske komunikacije pomoću I2C protokola sa OLED ekranom se za podatkovnu liniju SDA koristi nožica PB 7, a za liniju takta SCL nožica PB 6.

2. Popis komponenata

1. Mikro upravljač

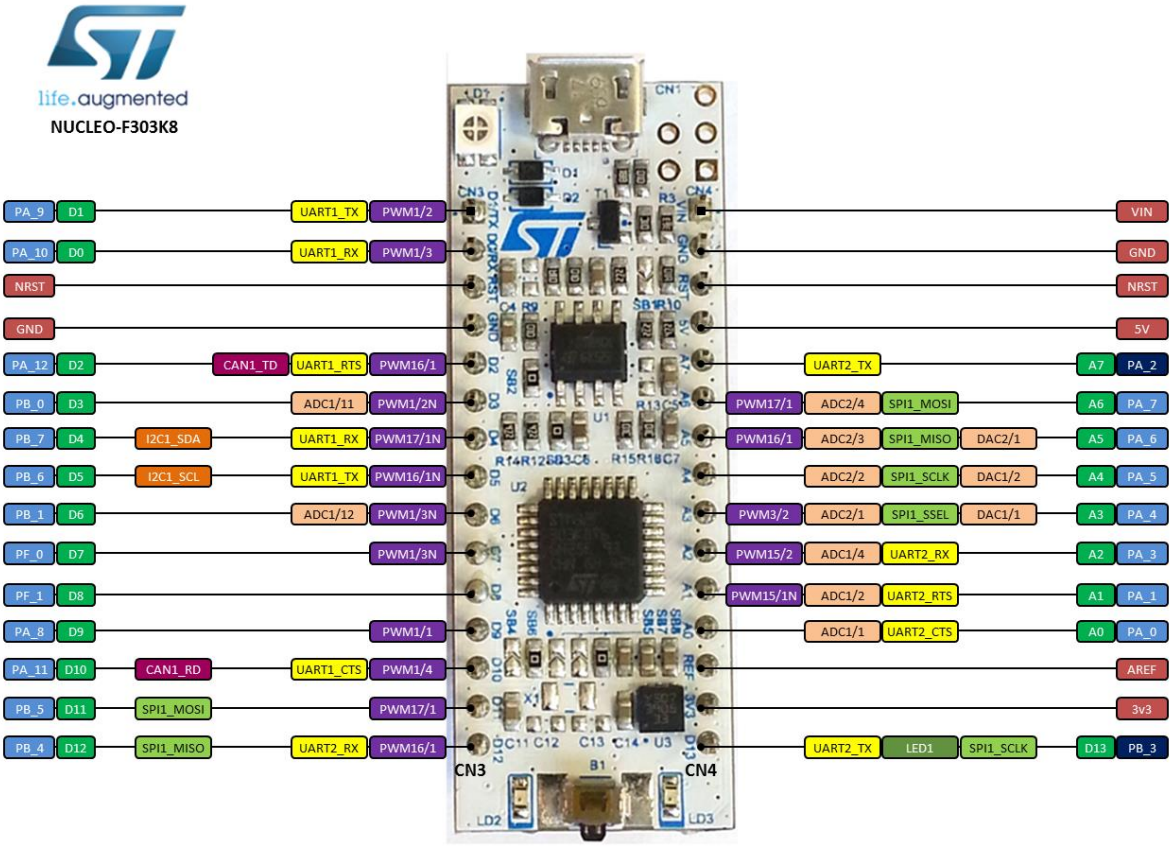
Slika 2 Nucleo STM32F303K8 [1]

Nucleo STM32F303K8 je mikro upravljač tipa *Nucleo 32* temeljen na *STM32 F3* obitelji mikro upravljačkih integriranih krugova. Mikro upravljač sadrži STM32 mikroprocesor, *flash* memoriju, RAM memoriju i *STLINK debug* sučelje. Mikroprocesor je temeljen na Cortex-M4 32 bitnom ARM jezgrom sa radnom frekvencijom od 72 MHz. Memorija mikroupravljača se sastoji 12kB RAM memorije, 64 kB *flash* memorije i jedinstvenog 96 bitnog identifikacijskog broja. [2]

Mikro upravljač se može napajati naponom od 3,3 do 12 V, a VDD *pin* daje napon od 2 do 3,6 V, na pločici su 25 *GPIO pina* s mogućnošću korištenja vanjske prekidne funkcije. Ugrađeno je osam *timera* od kojih su dva „Basic“, odnosno služe

samo za generiranje takta, pet univerzalnih *timera* koji uz brojanje imaju ugrađene mogućnosti zapisivanja vrijednosti *timera* prema vanjskoj prekidnoj funkciji, uspoređivanja vrijednosti *timera sa zadanom vrijednošću* i mogućnost stvaranja pulsno širinsko moduliranog signala i na kraju jedan „napredni“ *timer* s raznim mogućnostima za upravljanje motorima. Za komunikaciju s drugim uređajima *Nucleo STM32F303K8* može koristiti SPI, I2C, USART i CAN protokole. [1]

Posljednje bitne značajke *STM32 F3xx* serije mikro upravljača su četiri neovisna 12 bitna analogno-digitalna pretvarača koji se mogu *multiplexirat*i na 30 zasebnih kanala, te četiri operacijskih pojačala frekvencije 8 MHz za razne primjene. [2]



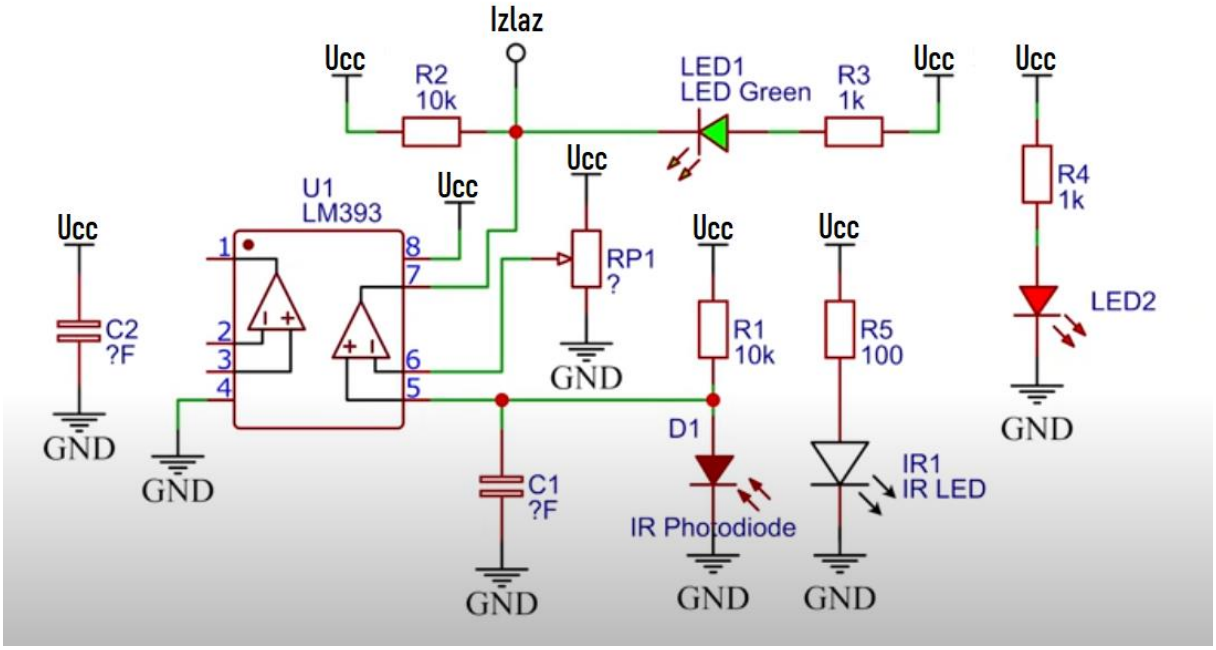
Slika 3 Izlazni pinovi mikro upravljača Nucleo STM32F303K8 [1]

2. Senzor udaljenosti

Slika 4 Senzor udaljenosti temeljen na komparatoru LM393 [3]

Senzor tahometra je temeljen na čipu LM393. Prema prilogu LM393 je čip s dva diferencijalna komparatora koji se mogu koristiti za razne primjene kao mjerenje udaljenosti, zvuka, svjetlosti i sl. Komparatori imaju najveći napon razlike do ±38 V ovisno o inačici čipa. LM393B radi sa strujom od 200 μA za svaki komparator, ima vrijeme odziva od 1 μs i namijenjen je za rad u temperaturnim uvjetima od -40 °C do 85 °C. Također izlazi su izvedeni u *open-drain* načinu što omogućava korištenje čip kao I logičke sklopke i podešavanje vrijednosti izlaznog napona u visokom logičkom stanju.

Senzor udaljenosti ima 3 nožice, napajanje, uzemljenje i izlaznu nožicu. Senzor ne može mjeriti promjenjivu udaljenost od prepreke, nego samo uočava da li se prepreka nalazi na nekoj zadanoj udaljenosti i prema tome na izlaznoj nožici daje napon od otprilike 0 ili 3,3 V, logičku 0 ili 1. Radi tako da pri dovođenju napajanja na senzor počnu svijetliti zelena signalna LED i infra-crvena LED. Na invertirajući ulaz komparatora se dovodi određeni napon preko potenciometra spojenog na napajanje. Kada je senzor udaljen od ikakvih prepreka zrake infra-crvene LED se ne vraćaju natrag na infra-crvenu foto-diodu pa je napon na neinvertirajućem ulazu komparatora jednak naponu napajanja od 3,3 V preko otpornika R1. Zbog toga na izlazu komparator također daje napon od 3,3 V kojih se poništi s naponom napajanja preko otpornika R2 te je na izlaznoj nožici 0 V. Kada je senzor dovoljno blizu nekoj prepreci dio zraka infra-crvene LED se odbija i vraća do foto-diode te ona počne voditi. Zbog vođenja struje foto-diode padne napon na neinvertirajućem ulazu komparatora i ako padne ispod vrijednosti napona na invertirajućem ulazu na izlazu komparatora će biti 0 V. Kroz drugu signalnu LED će poteći struja i na izlaznoj nožici će biti napon nešto manji od 3,3 V, logička jedinica. Tako prema ovom principu potenciometar na senzoru služi za namještanje udaljenosti na kojoj će izlaz dati logičku jedinicu.



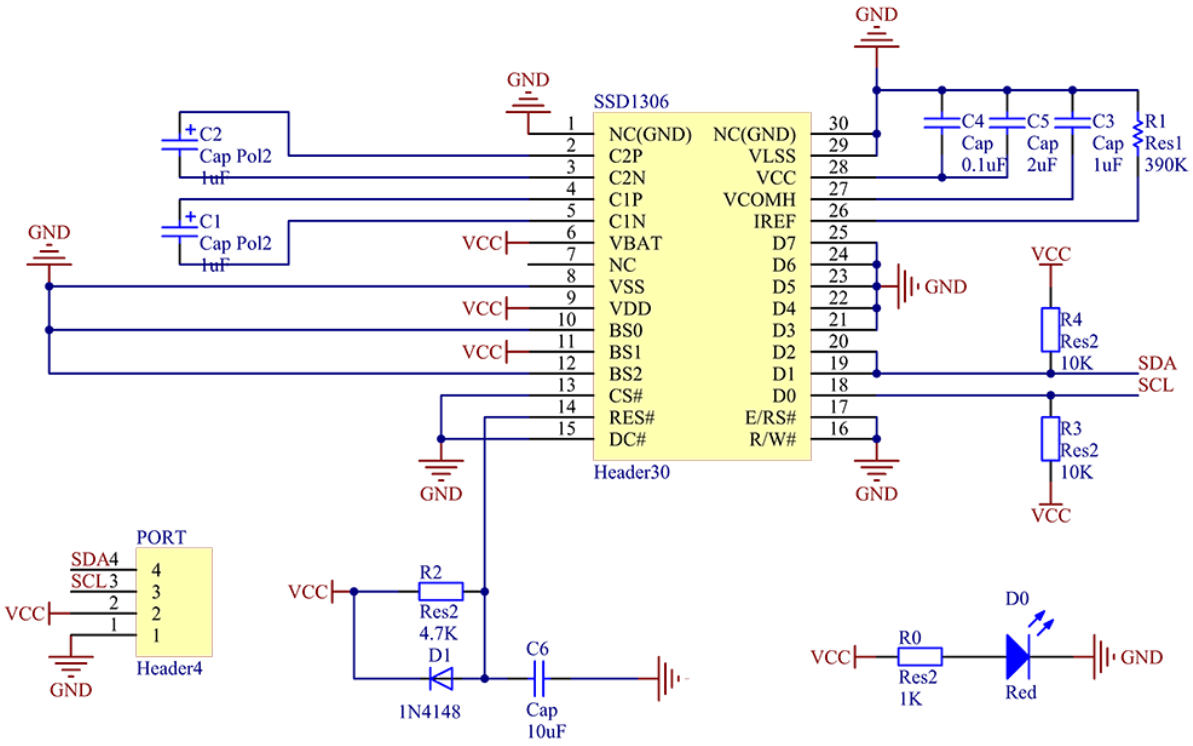
Slika 5 Shema senzora udaljenosti [4]

3. Ekran

Slika 6 SSD1306 OLED ekran [5]

Za ekran se koristi *SSD1306* OLED ekran sa rezolucijom od 128 x 64 piksela kojim se može upravljati pomoću samo dvije nožice korištenjem I2C serijskog komunikacijskog protokola i korištenjem naredbi iz *„Adafruit SSD1306“* biblioteke u samom programu. Kratice OLED na engleskom jeziku znači *„Organic Light-Emitting Diode“* i odnosi se na sloj organskog materija koji se nalazi između dvije elektrode ekrana od kojih je barem jedna prozirna i taj organski sloj emitira svjetlost prolaskom struje kroz njega. Princip rada OLED ekrana je znatno drukčiji od principa rada LED ekrana jer OLED matrica sama stvara svoju svjetlost te ne treba pozadinsku i pri prikazu crne boje ona je znatno tamnija nego kod LED ekrana. Uz to OLED ekrani su tanji u odnosu na LED ekrane.

Dijagonala ekrana je duljine od 0,96“ (~24 mm), a PCB je dimenzija 26 x 26 mm. Matrica organskih LED je izvedena sa zajedničkom katodom i koristi jedan CMOS za kontrolu te se nalazi na običnoj obostranoj tiskanoj pločici. Ekran se napaja naponom od 2,7 V do 5,5 V i potrošnja energije tokom rada je minimalna na 40 mW. Na vrhu tiskane pločice se nalaze četiri nožice slijeva, GND kao uzemljenje, VCC za napajanje, SCL nožica I2C protokola za sinkronizaciju slanja podataka sa mikro upravljača i nožica SDA za slanje podataka na ekran. [6]



Slika 7 Shema SSD1306 ekrana [6]

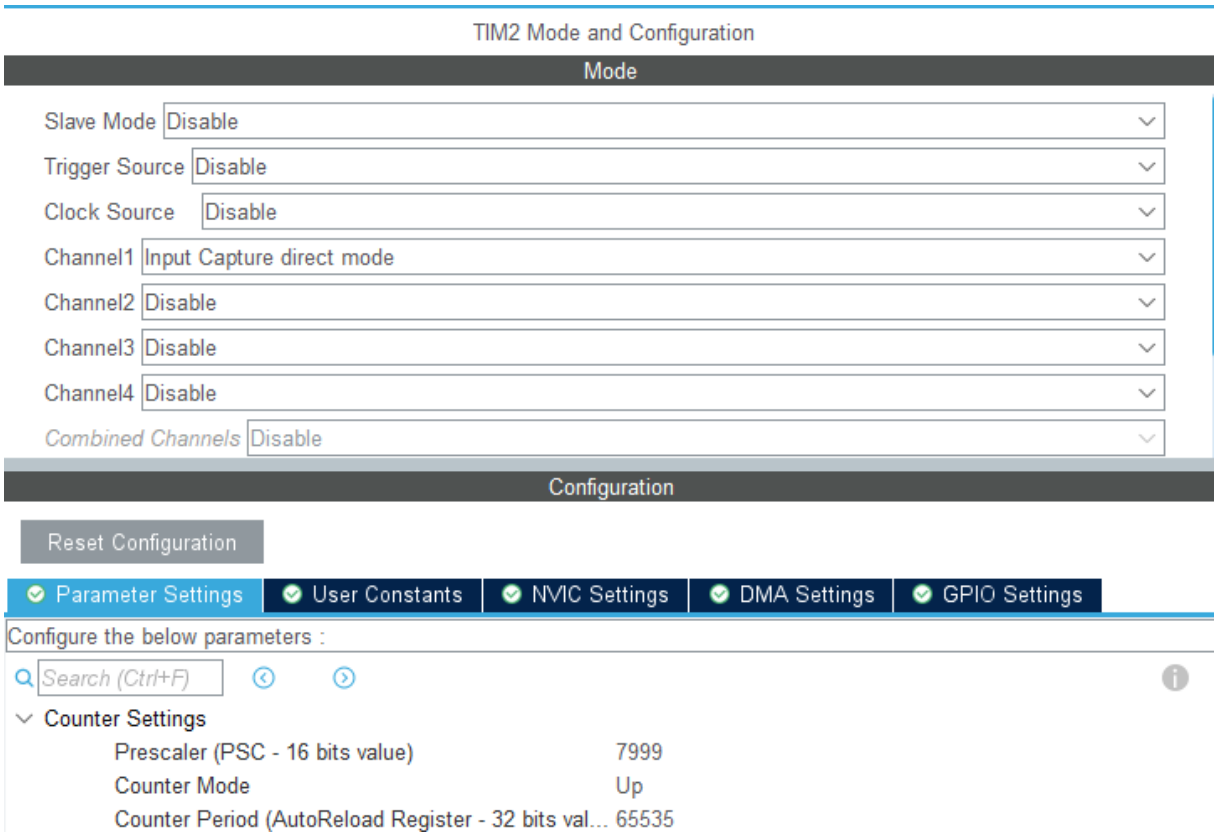
Također, *SSD1306* ima ugrađenu RAM memoriju i oscilator te mogućnost upravljanja jačinom svjetlosti piksela sa 256 razina osvjetljenja. Slika na ekranu je vidljiva pod kutom od 80° od okomice ekrana u bilo kojem smjeru. [6]

3. Program

Cijeli kod je spremljen u GitHub repozitoriju seminarskog rada.

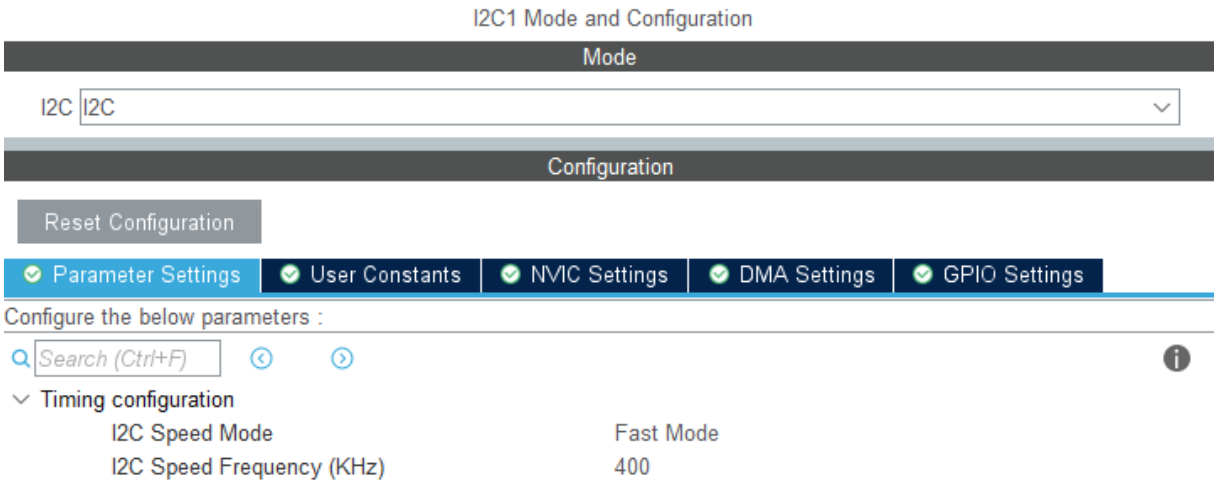
1. Postavljanje programa u STM32CubeMX

U aplikaciji STM32CubeMX je odabran mikro upravljač STM32F303K8. U kartici *„Pinout & configuration“* pod *„Timers“* izabran TIM2, pod opcijom *„Channel 1“* izabrano je *„Input capture direct mode“* čime se automatski na desnoj strani ekrana rezervira nožica mikro upravljača PA 0 kao *„TIM2_CH1“* koja služi za aktiviranje *interupt* funkcije. Niže u postavkama parametar *„Prescaler“* je postavljen na vrijednost 7 999 čime se od osnovne frekvencije 16 MHz dobije frekvencija od 2 kHz, ili period od 500 µs. Parametar „Counter period“ je postavljen na vrijednost 65 535 što znači da timer broji 65 535 takta po 500 µs, ili 32,7 s.

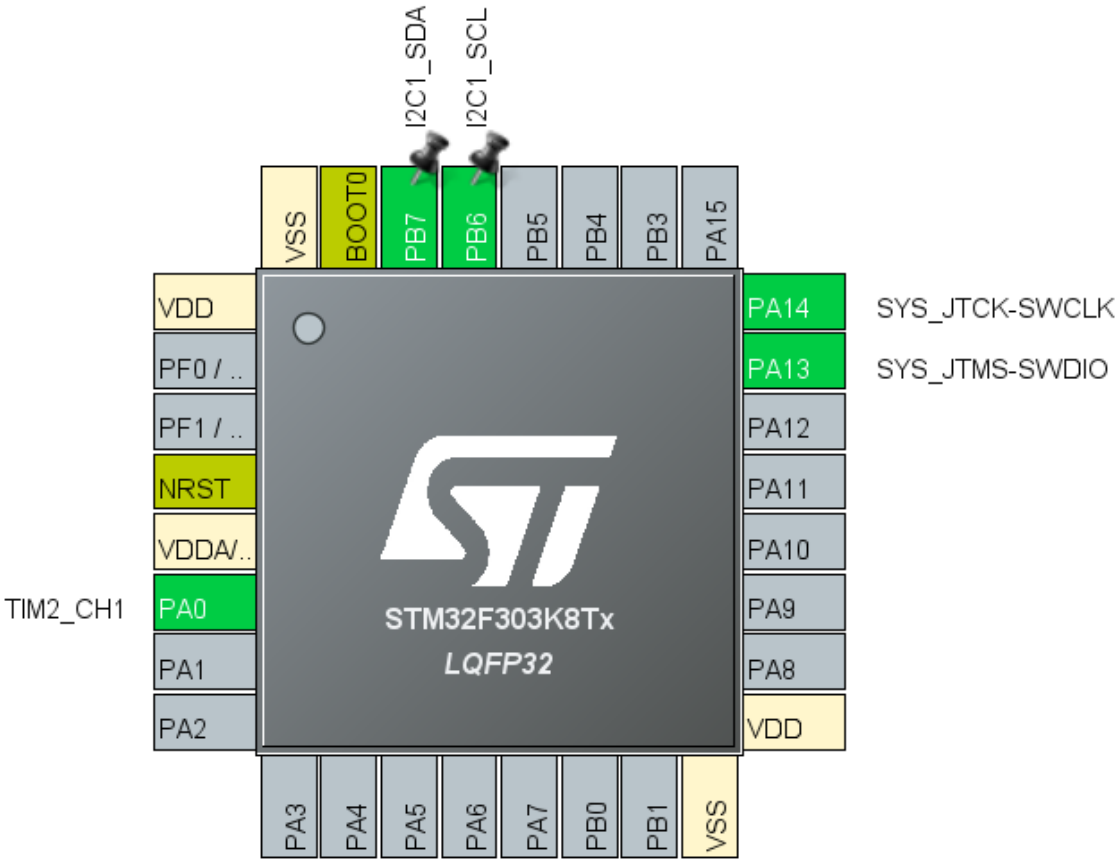


Slika 8 Konfiguracija timera 2

Za uspostavljanje serijske komunikacije je u kartici *„Pinout & configuration“* pod *„Connectivity“* izabran I2C1 i uključen I2C protokol. Time se rezerviraju nožice PA 14 i PA 15 za podatkovnu i liniju takta. Zatim se u postavkama opcija *„I2C Speed mode“* namjesti na *„Fast mode“* što postavlja frekvenciju serijske komunikacije na 400 kHz, ili 400 kb/s. Za OLED ekran *SSD1306* rezolucije 128 x 64 piksela jedna puna slika sadrži 8 192 bita, što znači da se sa frekvencijom od 400 kHz ekran može osvježavati frekvencijom od 48 Hz. Da se koristi više slave i master uređaja bilo bi potrebno za svaki postaviti jedinstvenu adresu, ali u ovom slučaju je samo jedan *master* i *slave* uređaj gdje *slave* može samo primati podatke.

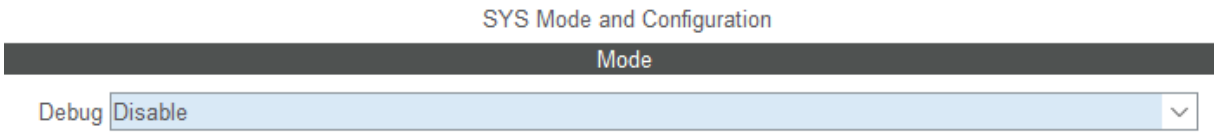


Slika 9 Konfiguracija I2C protokola



Slika 10 Prikaz korištenih nožica mikro upravljača

Podatkovna linija i linija takta, SDA i SCL, su premještene na nožice PB 7 i PB 6 jer se kartici *„Pinout & configuration“* pod *„System core“* treba izabrati SYS i opcija *„Debug“* namjestiti na *„Serial wire“* što rezervira nožice PA 13 i PA 14.



Slika 11 Konfiguracija SYS moda

2. I2C.c datoteka

Kod datoteke:

```
1
2 #include "i2c.h"
3
4 I2C_HandleTypeDef hi2c1;
5
6
7 void MX_I2C1_Init(void)
8 {
9   hi2c1.Instance = I2C1;
10  hi2c1.Init.Timing = 0x0000020B;
11  hi2c1.Init.OwnAddress1 = 0;
12  hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
13  hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
14  hi2c1.Init.OwnAddress2 = 0;
15  hi2c1.Init.OwnAddress2Masks = I2C_OA2_NOMASK;
16  hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
17  hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
18  if (HAL_I2C_Init(&hi2c1) != HAL_OK)
19  {
20    Error_Handler();
21  }
22/** Configure Analogue filter
23  */
24  if (HAL_I2CEx_ConfigAnalogFilter(&hi2c1,
I2C_ANALOGFILTER_ENABLE) != HAL_OK)
25  {
26    Error_Handler();
27  }
28/** Configure Digital filter
29  */
30  if (HAL_I2CEx_ConfigDigitalFilter(&hi2c1, 0) != HAL_OK)
31  {
32    Error_Handler();
33  }
34}
35
36void HAL_I2C_MspInit(I2C_HandleTypeDef* i2cHandle)
37{
38
39  GPIO_InitTypeDef GPIO_InitStruct = {0};
40  if(i2cHandle->Instance==I2C1)
41  {
42    __HAL_RCC_GPIOB_CLK_ENABLE();
43    /**I2C1 GPIO Configuration
```

```
44     PB6      ------> I2C1_SCL
45     PB7      ------> I2C1_SDA
46     */
47     GPIO_InitStruct.Pin = GPIO_PIN_6|GPIO_PIN_7;
48     GPIO_InitStruct.Mode = GPIO_MODE_AF_OD;
49     GPIO_InitStruct.Pull = GPIO_NOPULL;
50     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_HIGH;
51     GPIO_InitStruct.Alternate = GPIO_AF4_I2C1;
52     HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
53
54     /* I2C1 clock enable */
55     __HAL_RCC_I2C1_CLK_ENABLE();
56   }
57}
58
59void HAL_I2C_MspDeInit(I2C_HandleTypeDef* i2cHandle)
60{
61
62   if(i2cHandle->Instance==I2C1)
63   {
64     /* Peripheral clock disable */
65     __HAL_RCC_I2C1_CLK_DISABLE();
66
67     /**I2C1 GPIO Configuration
68     PB6      ------> I2C1_SCL
69     PB7      ------> I2C1_SDA
70     */
71     HAL_GPIO_DeInit(GPIOB, GPIO_PIN_6);
72
73     HAL_GPIO_DeInit(GPIOB, GPIO_PIN_7);
74   }

75}
```

I2C je *half-duplex* protokol serijske komunikacije što znači da *master* i *slave* uređaji mogu primati i slati podatke, ali samo na zahtjev *master* uređaja. S obzirom da može biti više *master* i *slave* uređaja inače za svaki treba postaviti jedinstvenu 7 bitnu adresu u parametru „*hi2c1.Init.OwnAddress*“, 11. i 14. red. U ovom slučaju se komunikacija odvija između jednog *master* i jednog *slave* uređaja pa to nije potrebno. Slave uređaj čak može imati i dvije adrese uključivanjem parametra „*hi2c1.Init.DualAddressMode“* u 13. redu, što omogućava da dva različita *master* uređaja mogu pozvati isti *slave* uređaj korištenjem različitih adresu pri pozivanju. Pri slanju podataka *master* uređaj svaki put treba definirati adresu *slave* uređaja kojem šalje podatak, ali u slučaju da se svim *slave* uređajima treba poslati isti podatak u 16. redu postoji opcija „*i2c1.Init.GeneralCallMode*“ koja to omogućava bez upisivanja

svake pojedinačne adrese *slave* uređaja. Kada se koristi više *master* uređaja može se dogoditi da više njih u isto vrijeme šalje podatak na isti *slave* uređaj i preplavi ga. U tom slučaju postoji parametar „*hi2c1.Init.NoStretchMode*“, 17. redak, koji omogućuje slave uređaj da zaustavi naredbe koje pristižu dok do kraja ne izvrši započete naredbe.

Za uspješnu uspostavu komunikacije između uređaja isto tako je potrebno definirati nožice podatkovne i linije takta. U ovom slučaju to su nožice PB 7 i PB 6 s definiranim parametrima od 47. do 52. retka. Prvo je potrebno nožice PB 6 i 7 definirati kao *GPIO,* univerzalni ulaz-izlaz (eng. *General Purpose Input Output*). Zatim se definira način rada, eng. *mode*, kao „*GPIO_MODE_AF_OD*“ gdje „*AF*“ znači „*Alternate function*“, a „*OD*“ znači „*Open drain*“. *Alternate function* znači da se nožica koristi za neku drugu svrhu osim uobičajenog ulaza ili izlaza kao u ovom slučaju linija takta i podatkovna linija za I2C serijski protokol. Točna svrha nožice ako nije samo ulaz ili izlaz se definira pod parametrom „*GPIO_InitStruct.Alternate*“ kao *„GPIO_AF4_I2C1“ u* 51. retku. *Open Drain* znači da digitalni izlaz djeluje kao ponor, to jest na nožici je visoko stanje od 3,3 V sve dok mikro upravljač na izlaz ne pošalje signal koji preko tranzistora kratko spoji izlaz sa uzemljenjem što uzrokuje nisku logičku razinu na izlaznoj nožici. Osim *Open Drain* načina rada postoji i „*Push-Pull*“ način rada. Odnosi si se na logičku naponsku razinu nožice kada nema jasnog signala sa mikro upravljača ili nekog vanjskog sklopa, ovisno da li je nožica izlaz ili ulaz. Za način rada kao ulaz ili izlaz, ako je naponska razina u zabranjenom području pri načinu rada „*Pull up*“ će se logička razina povući na 3,3 V, a kod „*Pull down*“ će se povući na 0. Za I2C komunikacijski protokol postoji nekoliko razina brzine prijenosa: *Standard mode* od 100 kb/s, *Fast mode* od 400kb/s, *Fast mode plus* od 1Mb/s, *High speed mode* od 3,4 Mb/s i *Ultra fast mode* od 5 Mb/s. *Nucleo STM32F303K8* podupire *Standard, Fast i Fast plus* brzinu prijenosa definiranu u 50. retku pod parametrom „*GPIO_InitStruct.Speed*“.

3. Main.c datoteka

Kod datoteke:

```
1 #include "main.h"
2 #include "i2c.h"
3 #include "tim.h"
4 #include "gpio.h"
5
6 #include "ssd1306.h"
7 #include "ssd1306_tests.h"
8
9 #include <stdio.h>
10#include <stdlib.h>
11#include <string.h>
12
13void SystemClock_Config(void);
14
15uint16_t T1=0, T2=0, zastavica=1;
16float W, T;
17char tekst[16];
18
19int main(void)
20{
21  HAL_Init();
22
23  SystemClock_Config();
24
25  MX_GPIO_Init();
26  MX_I2C1_Init();
27  MX_TIM2_Init();
28
29  HAL_TIM_IC_Start_IT(&htim2, TIM_CHANNEL_1);
30  ssd1306_Init();
31
32  while (1)
33  {
34     ssd1306_Fill(Black); //ispuna ekrane crnom bojom
35
36     ssd1306_SetCursor(25, 19); //postavljanje pokazivača
37     ssd1306_WriteString(tekst, Font_16x26, White);
//ispisivanje teksta
38
39     ssd1306_UpdateScreen(); //osvježavanje ekrana
40     HAL_Delay(200); //odgoda
41  }
42  /
43}
44
```

```
45void HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef* htim){
46
47   if(zastavica==1){
48        T1=TIM2->CCR1; //zapisivanje vrijednosti timera kod
prvog prolaza
49        zastavica=2;
50   }
51
52   else if(zastavica==2){
53
54        T2=TIM2->CCR1; //zapisivanje vrijednosti timera kod
drugog prolaza
55        T=(T2-T1)/2.0; //računanje razlike
56        W=60000.0/T; //računanje kružne frekvencije
57        itoa(W,tekst,10); //pretvaranje float varijable u
string
58        zastavica=1;
59        T1=0; //resetiranje varijabli
60        T2=0;
61   }

62}
```

„*Main*“ datoteka je glavna datoteka u koju se upisuje korisnički kod. Prvo se upisuju „*include*“ funkcije koje povezuju *main* datoteku s drugim datotekama koje sadrže razne funkcije i makro naredbe za uspješno izvršenje koda. Svaki kod uključuje barem „*main.h*“ datoteku, a u ovom slučaju također je uključena „*i2c.h*“ datoteka za serijsku komunikaciju, „*tim.h*“ datoteka za korištenje brojača i „*gpio.h*“ za korištenje ulazno-izlaznih nožica. Te datoteke se automatski uključuju u kod pri postavljanju funkcija programa u aplikaciji *STM32CubeMX.* Za ispis vrijednosti na ekran je potrebno uključiti dodatne datoteke koje su se morale posebno instalirati kao „*ssd1306.h*“ za korištenje ugrađenih funkcija „*Adafruit SSD1306“* biblioteke. Datoteka „*ssd1306_tests.h*“ sadrži primjere gotovih funkcija za ispis raznih vrijednosti i slika na ekran, iz datoteke se može vidjeti kako koja funkcija radi za daljnju primjenu. Na kraju je potrebno uključiti „*stdio.h*“, „*stdlib.h*“ i „*string.h*“ datoteke za obradu teksta, potrebno je brojčane vrijednosti pretvarati u tekst za ispis na ekran.

Nakon uključenja svih potrebnih datoteka slijedeći dio programa od reda 13 do 17 konfigurira takt cijelog sustava u redu 13 naredbom „*SystemClock_Config(void)“* i korisnik inicijalizira sve potrebne varijable. Oznaka „*uint16_t*“ označava 16 bitni cijeli

broj bez predznaka i u kodu se koriste četiri korisničke varijable tog tipa: *T1* za prvo mjerenje, *T2* za drugo mjerenje i *zastavica* za naizmjenični upis između varijabli *T1* i *T2*. Za decimalne brojeve se koristi oznaka „*float*“ i koriste ju dvije varijable: *T* za računanje vremenskog perioda između dva prolaza mjerne trake i *W* za pohranu konačne vrijednosti mjerenja u okretajima po minuti. Za ispis na ekran je bilo kakvu vrijednost potrebno pretvoriti u tekst pa se stoga definira varijabla *tekst* tipa *char* (eng. Character) veličine 16 bita.

Slijedi glavni dio programa između retka 19 i 43. Najprije je nužno inicijalizirati HAL biblioteku naredbom „*HAL_Init()*“ kako bi se mogle koriste sve HAL funkcije. Dalje se poziva funkcija „SystemClock_Config**()**“ sa svim korisničkim varijablama i inicijaliziraju se ulazno-izlazne nožice, svi parametri za serijsku komunikaciju i korištenje vremenskih brojača. Prije korisničkog koda se još poziva naredba „*HAL_TIM_IC_Start_IT**(&**htim2**,** TIM_CHANNEL_1**)**“ kojoj TIM2 počinje odbrojavanje u *Input capture mode* i inicijalizira se SSD1306 biblioteka. Korisnički dio programa koji se izvršava beskonačno se zapisuje u *while* petlju između redaka 32 i 41. Svaki ciklus započinje s naredbom „*ssd1306_Fill**(**Black**)**“ koja ekran ispuni crnom bojom, ovo je nužan korak jer inače bi svaki slijedeći ciklus nadodavao na prijašnju sliku prikazanu na ekranu. Naredba „*ssd1306_SetCursor**(**25**,** 19**)**“ postavlja pokazivač na željenu točku ekrana, to jest označava u odnosu na koju točku će se ispisivati tekst na ekranu. Bez definicije točke sav tekst će se ispisivati u gornjem lijevom kutu ekrana. Za slanje informacije ekranu o ispisu teksta koristi se naredba „*ssd1306_WriteString**(**tekst**,** Font_16x26**,** White**)**“ u čijoj se sintaksi definira koji tekst se ispisuje, veličina teksta i boja. Na ekranu se ispisuje tekst sadržan u varijabli *tekst*, ali u početku neće pisati ništa sve dok se ne izračuna neka vrijednost tokom mjerenja. Ne ekranu će tekst postati vidljiv tek nakon što se pozove naredba „*ssd1306_UpdateScreen**()**“ za osvježenje ekrana, do tada će biti prikazan tekst od prijašnjeg osvježenja bez obzira koliko puta se pozvala naredba „*ssd1306_WriteString**“.*** Na kraju korisničkog koda je upotrijebljena naredba „*HAL_Delay(200)*“ koja zaustavi program radi lakšeg očitanja vrijednosti na ekranu. Naredba zaustavi program na 200 ms što znači da se ekran osvježava frekvencijom od 5 Hz i bez nje je teže očitati vrijednost jer stalno ona skakuće oko neke srednje vrijednosti.

Vremenski brojač je povezan sa *interupt* funkcijom i kod koji se izvršava kod svakog poziva *interupt* funkcije se nalazi između 45. i 61. reda. Općenito kada se pozove *interupt* funkcija prvo se pozove glavna funkcija koja zatim pozove funkciju koja označava točnu nožicu na kojoj se dogodio *interupt* i na kraju se poziva „*Callback*“ funkcija u koju korisnik upisuje naredbe za izvršavanje nakon ulaska u interupt funkciju. Za brojače koji koriste Input capture mode prva funkcija je *„TIM2_IRQHandler“* koja poziva drugu, *„HAL_TIM_IRQHandler(&htim2)“,* i koja na kraju poziva *„HAL_TIM_IC_CaptureCallback(TIM_HandleTypeDef* htim)“.* Svaki put kada mjerna traka prođe ispred senzora aktivira se *interupt* funkcija. S obzirom da je varijabla *zastavica* isprva vrijednosti 1 prvom ulazu u i*nterupt* funkciju će se na kraju ući u prvu *if* funkciju, red 47, čiji je uvjet ***(zastavica==1)*** pa se u varijablu T1 upisuje trenutačna vrijednost timera iz registra CCR1 (eng. *Counter compare register*). Na kraju *if* funkcije se varijabla *zastavic*a postavlja na vrijednost 2. Kod slijedećeg prolaza mjerna trake ispred senzora će se ući u *else if* funkciju s obzirom da je varijabla zastavica vrijednosti 2. Tada se vrijednost registra CCR1 upisuje u varijablu *T2* te se računa razlika između *T2* i *T1* i dijeli sa 2 da se dobije vrijeme u milisekundama i ta vrijednost se zapisuje u varijablu *T.* Za dobivanje vrijednosti okretaja u minuti se broj 60 000 dijeli sa varijablom *T*, dijeli se broj 60 000 jer je varijabla *T* u milisekundama pa se 60 s pretvori u 60 000 ms. Brojčanu varijablu treba pretvoriti u tekst naredbom *„itoa(W,tekst,10)“* u čijoj sintaksi prva varijabla je broj, u drugu varijablu se sprema tekst, a na zadnjem mjestu se određuje veličina tekstne varijable. Na kraju se varijable T1, T2 i zastavica postave na njihove početne vrijednosti i mjerenje počinje ispočetka. Sada kada while petlja uđe u novi krug i dođe do naredbe *„ssd1306_WriteString(tekst, Font_16x26, White)“* će se na ekranu prikazati izmjerena vrijednost.

4. Tim.c datoteka

Kod datoteke:

```
1 #include "tim.h"
2
3 TIM_HandleTypeDef htim2;
4
5 void MX_TIM2_Init(void)
6 {
7   TIM_MasterConfigTypeDef sMasterConfig = {0};
8   TIM_IC_InitTypeDef sConfigIC = {0};
9
10  htim2.Instance = TIM2;
11  htim2.Init.Prescaler = 7999;
12  htim2.Init.CounterMode = TIM_COUNTERMODE_UP;
13  htim2.Init.Period = 65535;
14  htim2.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
15  htim2.Init.AutoReloadPreload =
TIM_AUTORELOAD_PRELOAD_DISABLE;
16  if (HAL_TIM_IC_Init(&htim2) != HAL_OK)
17  {
18    Error_Handler();
19  }
20  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
21  sMasterConfig.MasterSlaveMode =
TIM_MASTERSLAVEMODE_DISABLE;
22  if (HAL_TIMEx_MasterConfigSynchronization(&htim2,
&sMasterConfig) != HAL_OK)
23  {
24    Error_Handler();
25  }
26  sConfigIC.ICPolarity = TIM_INPUTCHANNELPOLARITY_RISING;
27  sConfigIC.ICSelection = TIM_ICSELECTION_DIRECTTI;
28  sConfigIC.ICPrescaler = TIM_ICPSC_DIV1;
29  sConfigIC.ICFilter = 0;
30  if (HAL_TIM_IC_ConfigChannel(&htim2, &sConfigIC,
TIM_CHANNEL_1) != HAL_OK)
31  {
32    Error_Handler();
33  }
34}
35
36void HAL_TIM_IC_MspInit(TIM_HandleTypeDef* tim_icHandle)
37{
38
39  GPIO_InitTypeDef GPIO_InitStruct = {0};
40  if(tim_icHandle->Instance==TIM2)
41  {
```

```
42    /* TIM2 clock enable */
43    __HAL_RCC_TIM2_CLK_ENABLE();
44
45    __HAL_RCC_GPIOA_CLK_ENABLE();
46    /**TIM2 GPIO Configuration
47    PA0     ------> TIM2_CH1
48    */
49    GPIO_InitStruct.Pin = GPIO_PIN_0;
50    GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
51    GPIO_InitStruct.Pull = GPIO_NOPULL;
52    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
53    GPIO_InitStruct.Alternate = GPIO_AF1_TIM2;
54    HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
55
56    /* TIM2 interrupt Init */
57    HAL_NVIC_SetPriority(TIM2_IRQn, 0, 0);
58    HAL_NVIC_EnableIRQ(TIM2_IRQn);
59  }
60}
61
62void HAL_TIM_IC_MspDeInit(TIM_HandleTypeDef* tim_icHandle)
63{
64
65  if(tim_icHandle->Instance==TIM2)
66  {
67    /* Peripheral clock disable */
68    __HAL_RCC_TIM2_CLK_DISABLE();
69
70    /**TIM2 GPIO Configuration
71    PA0     ------> TIM2_CH1
72    */
73    HAL_GPIO_DeInit(GPIOA, GPIO_PIN_0);
74
75    /* TIM2 interrupt Deinit */
76    HAL_NVIC_DisableIRQ(TIM2_IRQn);
77  }
78}
```

U *tim.c* datoteci su definirani svi parametri vezani za *timere*. Prvo je potrebno uključiti „*tim.h*“ datoteku kako bi mogli koristiti sve funkcije vezane za *timere* te se definira koji *timer* koristimo naredbom „“ u 3. retku, u ovom slučaju *timer* TIM2. Namještene postavke iz aplikacije *STM32CubeMX* kao *Prescaler* i *Counter* period su spremljene u ovoj datoteci između redaka 10 i 13. Vrijednost djelitelja osnovne frekvencije, *Prescalera*, je 7 999, a broj do kojeg brojač broji, *Counter* period, je 65 535. Također *timer* je automtaski namješten da broji unaprijed, od 0 nadalje. Nožica

preko koje se aktivira ugrađena prekidna funkcija između retka 45 i 54. Prvo se definira da se radi o nožici grupe A, PA , naredbom „*__HAL_RCC_GPIOA_CLK_ENABLE()*“ i naredbom „*GPIO_InitStruct.Pin = GPIO_PIN_0*“ u 49. retku se definira broj nožice 0, PA 0. Način rada nožice se definira naredbom „*GPIO_InitStruct.Mode = GPIO_MODE_AF_PP*“ gdje kao i za podatkovnu liniju i liniju takta I2C komunikacijskog protokola način rada je „*Alternate function*, AF, što znači da se nožica koristi za neku drugu svrhu osim običnog ulaza ili izlaza. Ovdje se nožica koristi za aktivaciju prekidne funkcije TIM2. Nožica PA 0 je namještena kao „*Push-Pull“,* PP, što znači da bi se njezino stanje moglo povući na logičku 0 ili 1 da u 51. retku parametar „*GPIO_InitStruct.Pull*“ nije definiran kao „*GPIO_NOPULL*“. Brzina osvježavanja stanja nožice je definirana u retku 52, „*GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW*“, kao standardna brzina od 100kb/s. Konkretna alternativna funkcija nožice je definirana u 53. redu naredbom „*GPIO_InitStruct.Alternate = GPIO_AF1_TIM2*“. Broj prioriteta prekidne funkcije je 0, inače ako se koristi više prekidnih funkcija ona s manjim dodijeljenim brojem ima veći prioritet, ali u ovom slučaju se koristi samo jedna funkcija.

4. Rezultati

Za provjeru točnosti je mjerena brzina okretaja stupne bušilice s pet brzina.

Tablica 1 Rezultati mjerenja brzine stupne bušilice

Brzina prema tablici stroja [min^{-1}]	Izmjerena brzina [min^{-1}]	Odstupanje [min^{-1}]	Odstupanje [%]
515	616	101	19,6
915	923	8	0,9
1430	1395	35	2,4
1950	1818	132	6,8
2580	2608	28	1,1
Srednje odstupanje		60,8 min^{-1}	6,2 %

Prema tablici 1 vidljivo je da je točnost mjerenja veća pri većim brzinama. Za brzine ispod ~900 min^{-1} odstupanje je veće od ±100 min^{-1} ,ili ±20 %. Za brzine iznad 900 min^{-1}, iznos odstupanja mjerenja dosta skače ali sveukupno je manji u odnosu na red veličine teoretske brzine vrtnje pa je greška u postotcima znatno manja. Sve u svemu mjerenjima tahometra može se vjerovati u ±100 min^{-1}.

U samom pisanju koda nije bilo problema što se tiče bilježenja vrijednosti *timera* pri prolasku mjerne trake i preračunavanja u okretaje po minuti. Problem se pojavio u rezultatima mjerenja koji su većinu vremena ispadali okrugli u znamenku stotica bez obzira je li brzina okretaja stalna ili se kontinuirano mijenja. Tada je djelitelj osnovne frekvencije za TIM2 bio vrijednosti 15 999, što daje frekvenciju od 1 kHz ili puls od 1 ms. Pri promjeni tog parametra na vrijednost od 7 999, frekvencija 2 kHz, dobiveni su naizgled točniji rezultati. Zbog promjene tog parametra se varijabla *T* mora dijeliti sa 2. Još jedan manji problem se pojavio pri prikazu vrijednosti mjerenja na ekran, kada bi mjerena vrijednost pala iz npr. reda tisuća u red stotih okretaja, znamenka tisućice bi i dalje ostala na ekranu i mjerenje bi ispalo krivo. Tada je kod bi napisan tako da naredbom „*ssd1306_WriteString*“ prikaže mjerena vrijednost na ekranu i da se ekran osvježi naredbom „*ssd1306_UpdateScreen()“.* Međutim to ne znači da će se prijašnja slika obrisati sa ekrana nego će se ažurirati samo određena polja, a ostala će ostati nepromijenjena sa istim tekstom. Rješenje se nalazilo u naredbi „*ssd1306_Fill(Black)*“ kojom se prije svakog osvježenja ekran ispuni crnom bojom kako bi ekran bio prazan prije prikaza slijedeće slike.

Za ikakve buduće pokušaje u poboljšanju tahometra ili pravljenju novog prije svega bi se trebali osigurati adekvatni uvjeti mjerenja. Mjerna traka mora biti kontrastne boje u odnosu na pozadinu i mjereni objekt, te osvjetljenje ne bi smjelo biti prejako. U suprotnom vrijednost mjerenja znatno skače tokom mjerenja i teško je procijeniti koja vrijednost prikazana na ekranu je točna. Također za poboljšanje točnosti možda postoji optimalna frekvencija mjerenja koja nije nužno korištena u ovom radu, može se uspostaviti isprobavanjem raznih frekvencija i utvrđivanjem one s najboljom točnošću.

5. Zaključak

Napravljeni uređaj više nego ispunjava očekivanja uzeći u obzir jednostavnost komponenti, cijelog sklopa i relativnu jednostavnost koda. Na tržištu postoje razni proizvodi u ovom području mjerenja i sudeći po recenzijama točnost nekih od njih je čak i gora za višu cijenu. To nije nužno slučaj za svaki proizvod, ali ovaj uređaj postiže pristojne rezultate za otprilike dva puta manju cijenu komponenti od 20 €. Naravno postoje znatno precizniji uređaji koji mogu mjeriti brzinu okretaja čak i do dvije decimale, ali za korištenje u svrhu hobija i nekih općih mjerenja uređaj iz ovog seminarskog rada je dostatan.

Ovim naporom je postignut jeftin uređaj jednostavne konstrukcije i načina rada koji je primjenjiv u raznim područjima kao npr. školstvo ili kućne radinosti. Ta jednostavnost rezultira dobrom pouzdanošću i lakoćom popravka.

Za buduće pokušaje u pravljenju istog ili sličnog uređaja najbolje bi bilo i dalje koristiti troškovno prihvatljive komponente kako bi se sačuvao duh ovog rada. Za poboljšanje točnosti možda i postoje bolje komponente ili bolje napisan kod što bi se utvrdilo iterativno.

Literatura

[1] armMBED, »NUCLEO-F303K8,« [Mrežno]. Available: https://os.mbed.com/platforms/ST-Nucleo-F303K8/. [Pokušaj pristupa 3 siječanj 2024].

[2] »Wikipedia - STM32,« [Mrežno]. Available: https://en.wikipedia.org/wiki/STM32. [Pokušaj pristupa 3 siječanj 2024].

[3] »IndiaMART,« [Mrežno]. Available: https://m.indiamart.com/proddetail/digital-ir-sensor-module-ir-proximity-lm393-2852131994748.html. [Pokušaj pristupa 3 siječanj 2024].

[4] »Instructables - DIY Tachometer (RPM Meter),« [Mrežno]. Available: https://www.instructables.com/DIY-Tachometer-RPM-Meter/. [Pokušaj pristupa 3 siječanj 2024].

[5] »Majju,« [Mrežno]. Available: https://www.majju.pk/product/oled-0-96-inch-display-ssd1306-oled-i2c-128x64-display/. [Pokušaj pristupa 3 siječanj 2024].

[6] »Sunfounder - OLED-SSD1306 Module,« [Mrežno]. Available: http://wiki.sunfounder.cc/index.php?title=OLED-SSD1306_Module. [Pokušaj pristupa 4 siječanj 2024].

Summary

The theme of this thesis was a device for measuring the number of rotations in a certain time span, a tachometer. The device consisted of a *Nucleo STM32F303K8* micro controller, an infra-red distance sensor and a 128 x 64 p OLED display with the total price of approximately 20 €. The result was gotten by measuring the time spans of two passing of the reference point in front of the sensor with the help of a timer and an interrupt. This value would be converted in angular frequency and be sent to the OLED display with the help of I2C serial communication protocol.

Key words: Time span, serial communication, angular frequency, infra-red sensor, timer, interrupt, rotating body

PRILOG – Tehničke specifikacije LM393B komparatora

LM393B.PDF

NEW LM393B and LM2903B
Improved specifications of B-version
– Maximum rating: up to 38 V
– ESD rating (HBM): 2k V
– Low input offset: 0.37 mV
– Low input bias current: 3.5 nA
– Low supply-current: 200 µA per comparator
– Faster response time of 1 µsec
– Extended temperature range for LM393B
– Available in tiny 2 x 2mm WSON package
B-version is drop-in replacement for LM293, LM393 and LM2903, A and V versions
Common-mode input voltage range includes ground
Differential input voltage range equal to maximum-rated supply voltage: ±38 V
Low output saturation voltage
Output compatible with TTL, MOS, and CMOS

Specification	LM393B	LM2903B	LM393 LM393A	LM2903	LM2903V LM2903AV	LM193	LM293 LM293A	Units
Supply Votlage	3 to 36	3 to 36	2 to 30	2 to 30	2 to 32	2 to 30	2 to 30	V
Total Supply Current (5V to 36V max)	0.6 to 0.8	0.6 to 0.8	1 to 2.5	1 to 2.5	1 to 2.5	1 to 2.5	1 to 2.5	mA
Temperature Range	−40 to 85	−40 to 125	0 to 70	−40 to 125	−40 to 125	−55 to 125	−25 to 85	°C
ESD (HBM)	2000	2000	1000	1000	1000	1000	1000	V
Offset Voltage (Max over temp)	± 4	± 4	± 9 ± 4	± 15	± 15 ± 4	± 9	± 9 ± 4	V
Input Bias Current (typ / max)	3.5 / 25	3.5 / 25	25 / 250	25 / 250	25 / 250	25 / 100	25 / 250	nA
Response Time (typ)	1	1	1.3	1.3	1.3	1.3	1.3	µsec

The comparator consists of a PNP darlington pair input, allowing the device to operate with very high gain and fast response with minimal input bias current. The input Darlington pair creates a limit on the input common mode voltage capability, allowing the comparator to accurately function from ground to V_{CC}– 1.5 V input. Allow for V_{CC}– 2 V at cold temperature.

The output consists of an open drain NPN (pull-down or low side) transistor. The output NPN sinks current when the negative input voltage is higher than the positive input voltage and the offset voltage. The V_{OL} is resistive and scales with the output current. See Figure 7-3 for V_{OL} values with respect to the output current.