Vhodno izhodne naprave

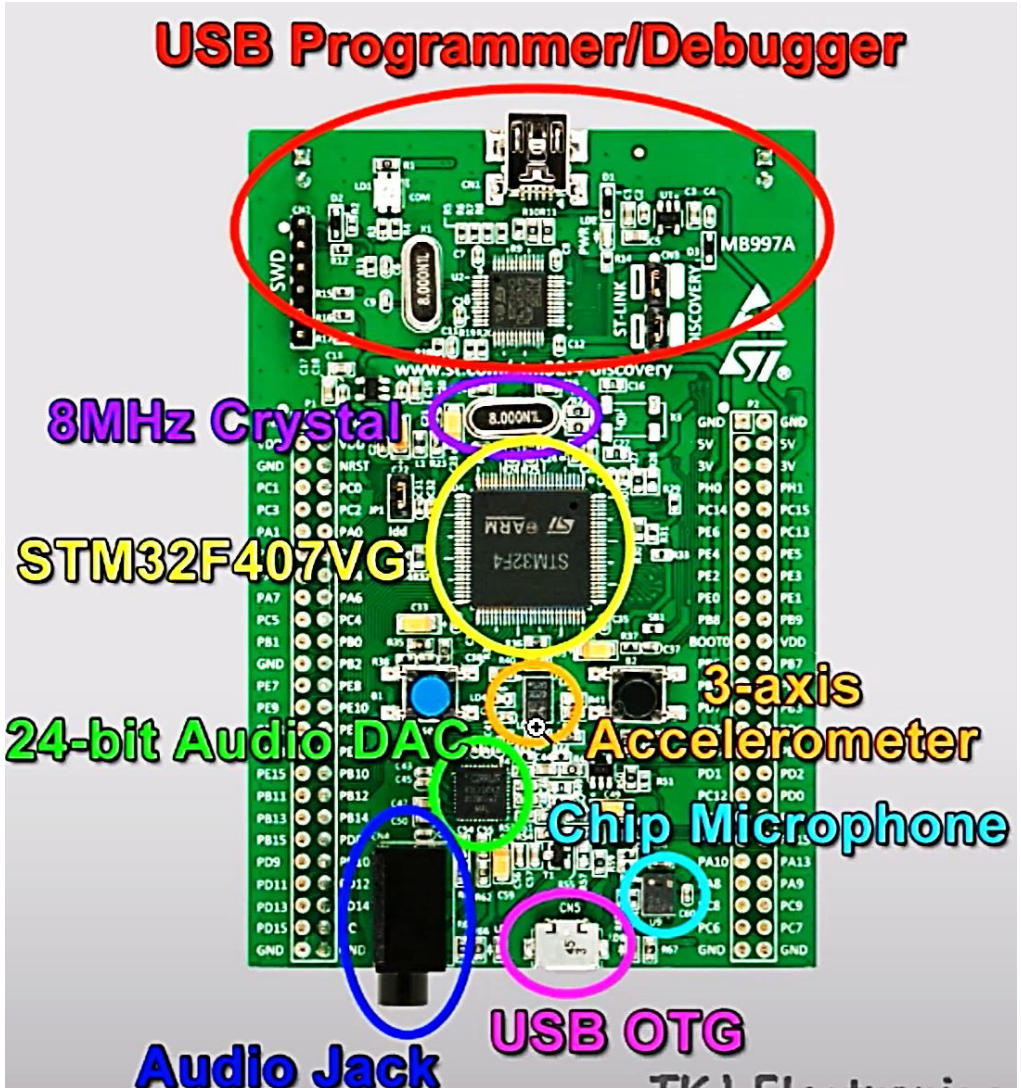
Laboratorijska vaja 5 - VP 5
STM32-CubeIDE projekt, breadboard vezave

VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

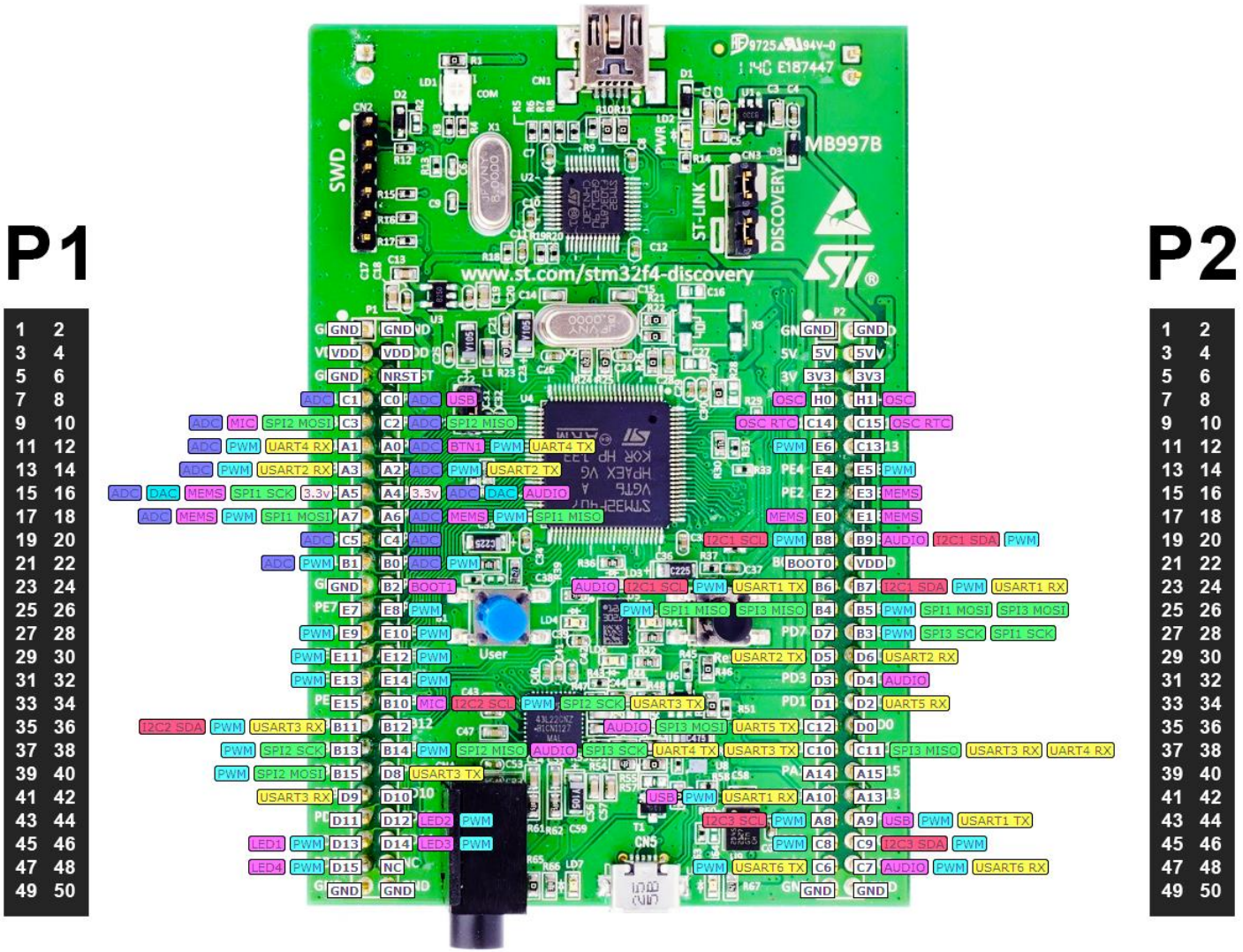
STM32F4DISCOVERY

3.3V !!!



STM32F4DISCOVERY

3.3V !!!



STM32F4DISCOVERY

3.3V !!!

Table 11. Voltage characteristics

Symbol	Ratings	Min	Max	Unit
V_{DD}–V_{SS}	External main supply voltage (including V_{DDA}, V_{DD})(1)	–0.3	4.0	V
V_{IN}	Input voltage on five-volt tolerant pin(2)	V_{SS}–0.3	V_{DD}+4	
	Input voltage on any other pin	V_{SS}–0.3	4.0	
$\lvert\Delta V_{DDx}\rvert$	Variations between different V_{DD} power pins	-	50	mV
$\lvert V_{SSX} - V_{SS}\rvert$	Variations between all the different ground pins including V_{REF-}	-	50	
$V_{ESD(HBM)}$	Electrostatic discharge voltage (human body model)	see *Section 5.3.14: Absolute maximum ratings (electrical sensitivity)*		

Table 12. Current characteristics

Symbol	Ratings	Max.	Unit
I_{VDD}	Total current into V_{DD} power lines (source)(1)	240	mA
I_{VSS}	Total current out of V_{SS} ground lines (sink)(1)	240	
I_{IO}	Output current sunk by any I/O and control pin	25	
	Output current source by any I/Os and control pin	25	
$I_{INJ(PIN)}$ (2)	Injected current on five-volt tolerant I/O(3)	–5/+0	
	Injected current on any other pin(4)	±5	
$\Sigma I_{INJ(PIN)}$(4)	Total injected current (sum of all I/O and control pins)(5)	±25	

Delo na STM32F4 razvojnem sistemu

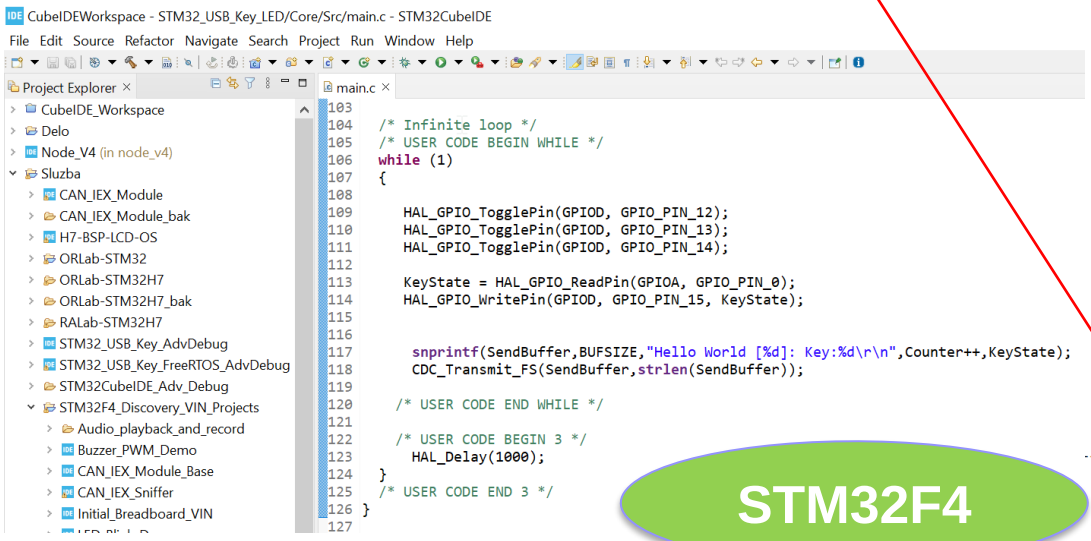
Mikro USB VCom-port

Priključitev :

- **Mini USB** priklop na **krajši stranici**, svetita rdeči **LED** diodi

Poseben začetni projekt za *STM32F4* (e-učilnica) :

- ***dodajanje vsebine (main.c):***



STM32F4

--------- Razvojni sistem STM32F407 Discovery ----------

- STM32F4DISCOVERY Discovery kit with STM32F407VG MCU
- VINLab-STM32 - GitHub repozitorij
- ORLab-STM32 - GitHub repozitorij
- STM32F4 - Dokumentacija

Lastni viri :

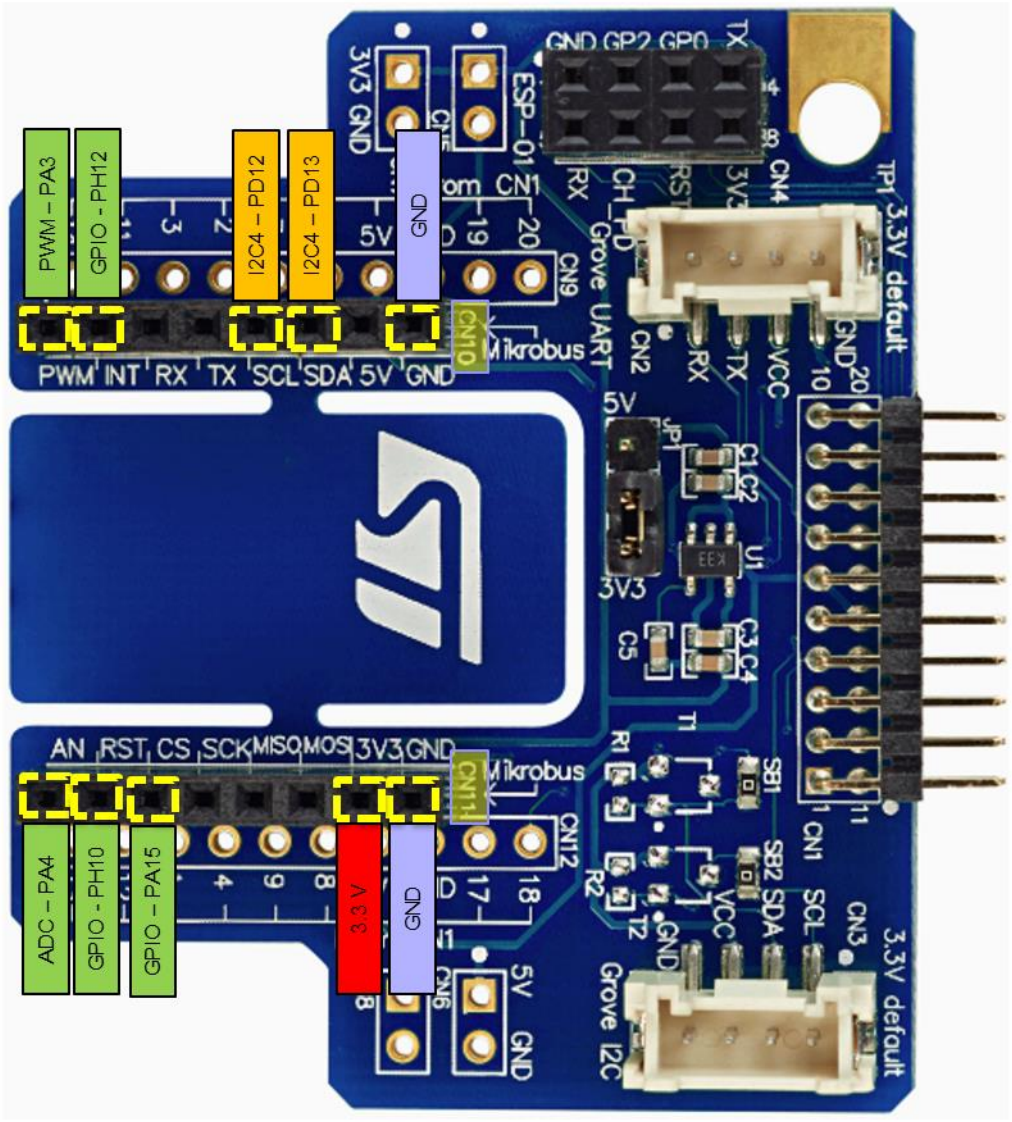
https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects

https://github.com/LAPSyLAB/STM32F4_Docs_and_Examples

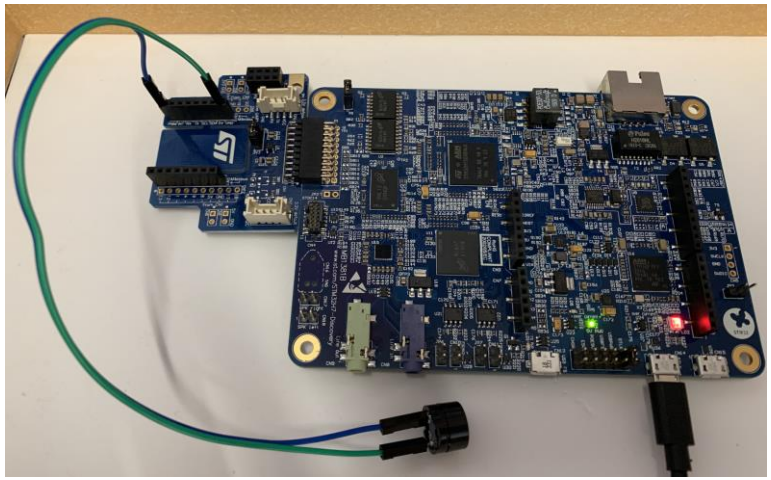
https://github.com/LAPSyLAB/ORLab-STM32

3.3V !!!

STM32H750B – DISCOVERY StMod+ konektor



Pravilna priključitev

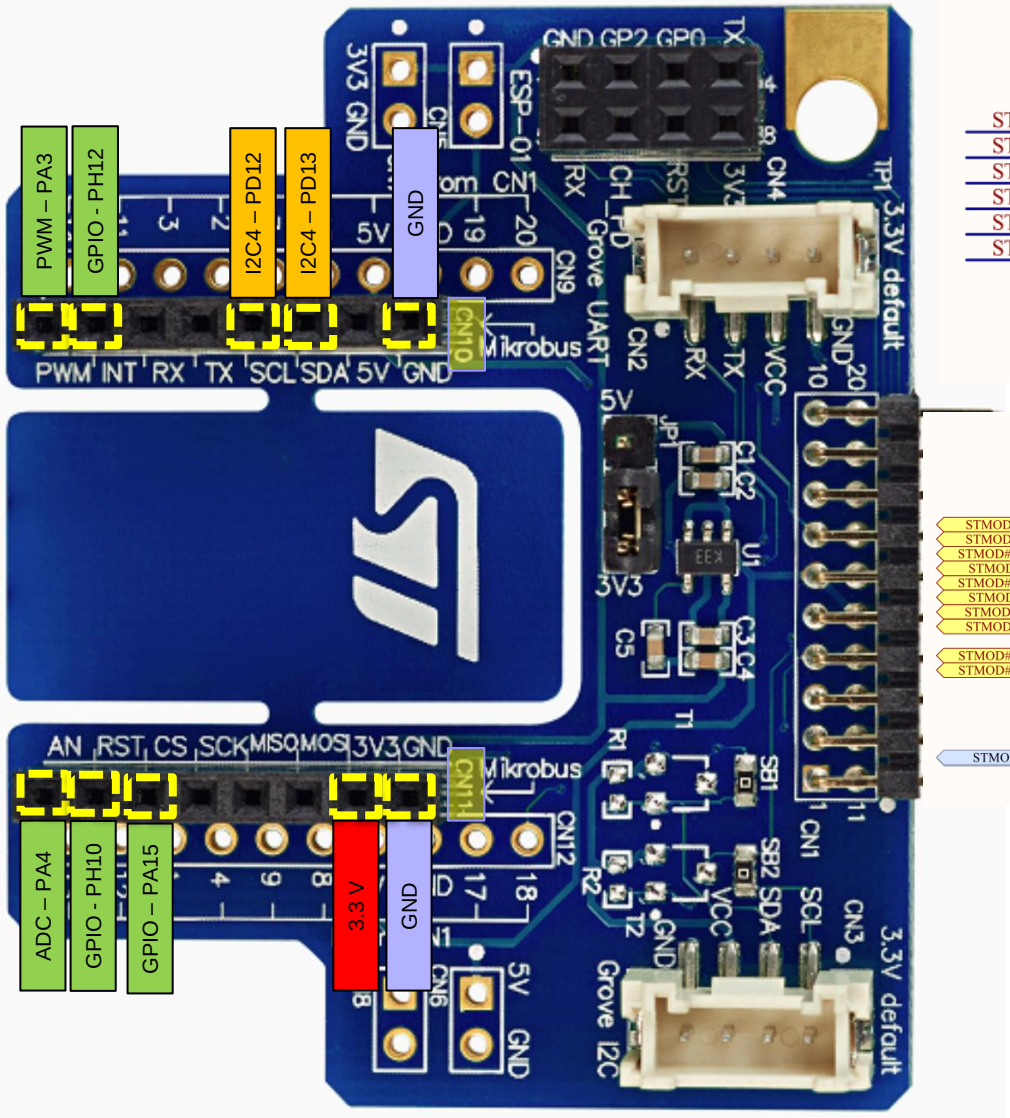


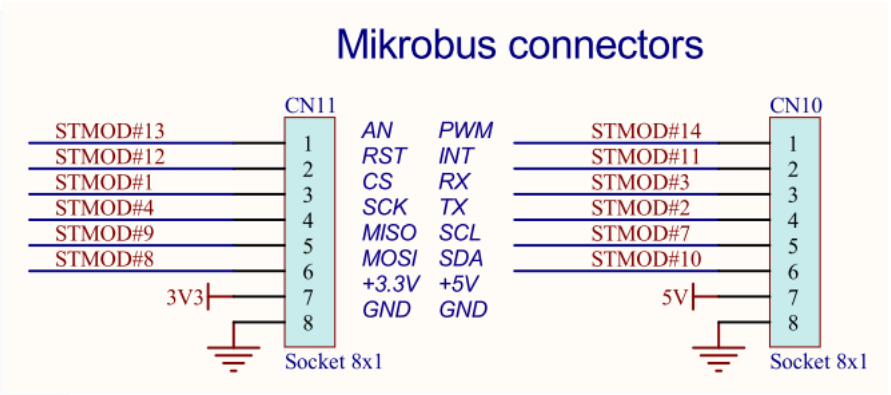
Nepravilna priključitev

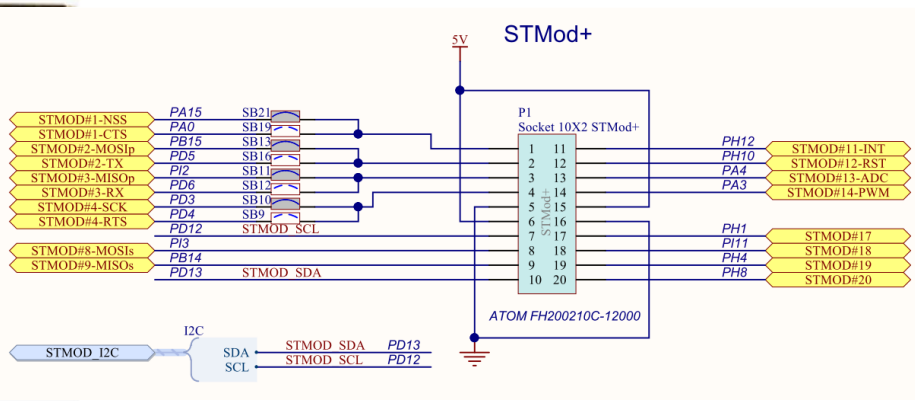
https://www.st.com/en/evaluation-tools/stm32h750b-dk.html

3.3V !!!

STM32H750B – DISCOVERY StMod+ konektor

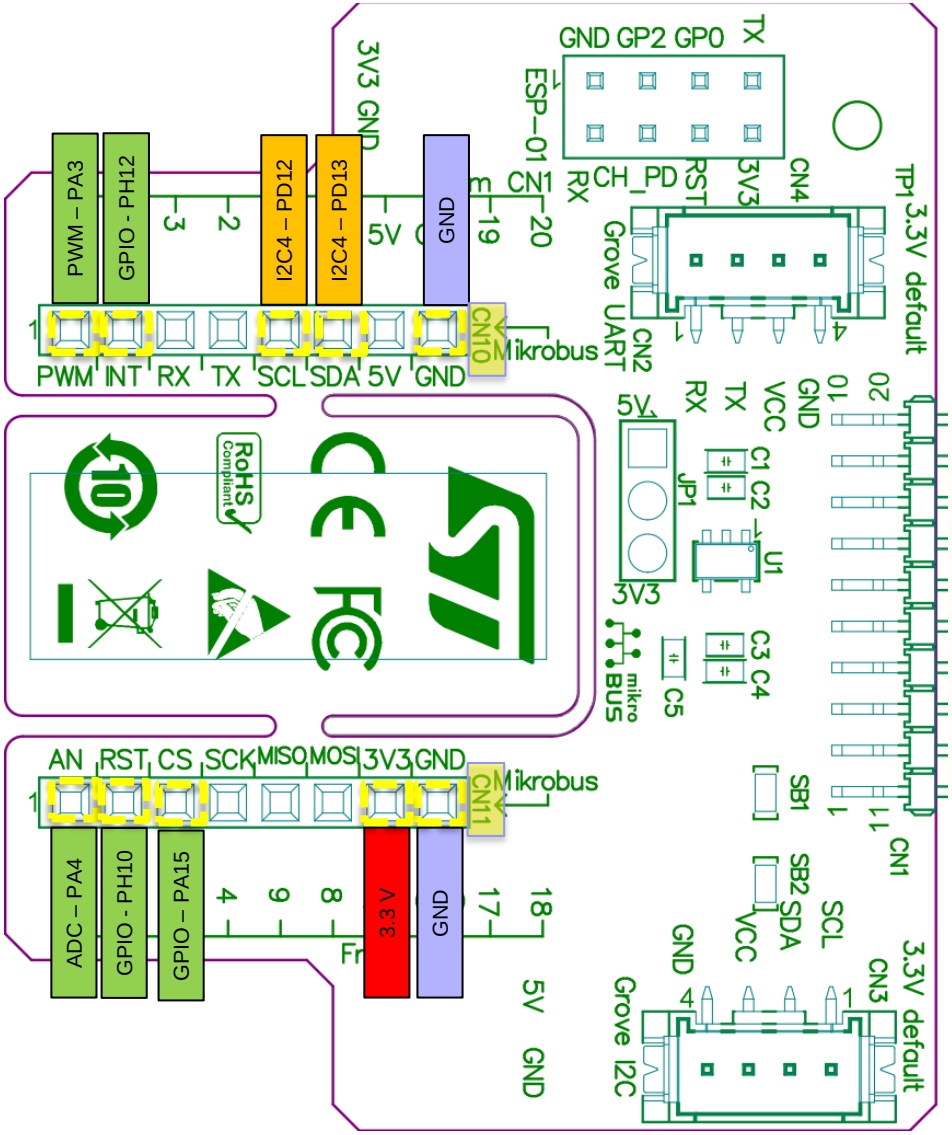


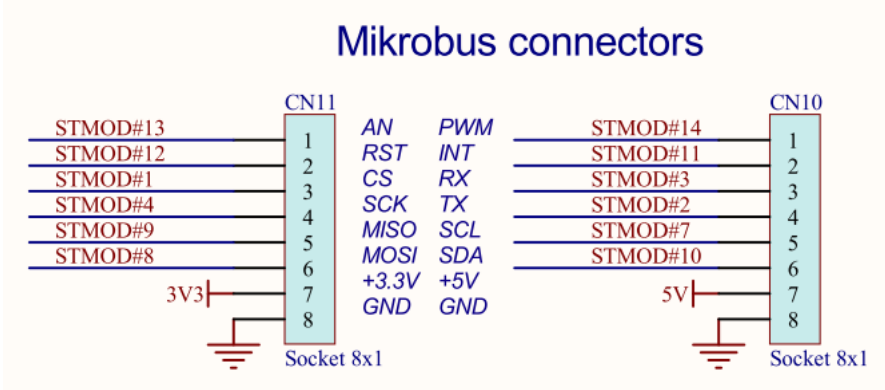


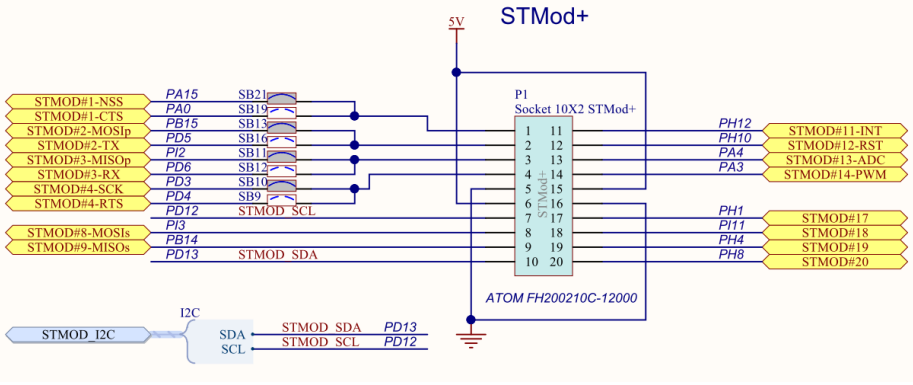


3.3V !!!

STM32H750B – DISCOVERY StMod+ konektor







STM32H750B – DISCOVERY

StMod+ konektor

3.3V !!!

3. This formula has to be applied on power supplies related to the IO structure described by the pin definition table.
4. To sustain a voltage higher than 4V the internal pull-up/pull-down resistors must be disabled.

Table 20. Current characteristics

Symbols	Ratings	Max	Unit
ΣIV_{DD}	Total current into sum of all V_{DD} power lines (source)(1)	620	mA
ΣIV_{SS}	Total current out of sum of all V_{SS} ground lines (sink)(1)	620	
IV_{DD}	Maximum current into each V_{DD} power pin (source)(1)	100	
IV_{SS}	Maximum current out of each V_{SS} ground pin (sink)(1)	100	
I_{IO}	Output current sunk by any I/O and control pin	20	
$\Sigma I_{(PIN)}$	Total output current sunk by sum of all I/Os and control pins(2)	140	
	Total output current sourced by sum of all I/Os and control pins(2)	140	
$I_{INJ(PIN)}$(3)(4)	Injected current on FT_xxx, TT_xx, RST and B pins except PA4, PA5	−5/+0	
	Injected current on PA4, PA5	−0/0	
$\Sigma I_{INJ(PIN)}$	Total injected current (sum of all I/Os and control pins)(5)	±25	

Output driving current

The GPIOs (general purpose input/outputs) can sink or source up to ±8 mA, and sink or source up to ±20 mA (with a relaxed V_{OL}/V_{OH}).

In the user application, the number of I/O pins which can drive current must be limited to respect the absolute maximum rating specified in *Section 6.2*. In particular:

Delo na STM32H7 razvojnem sistemu

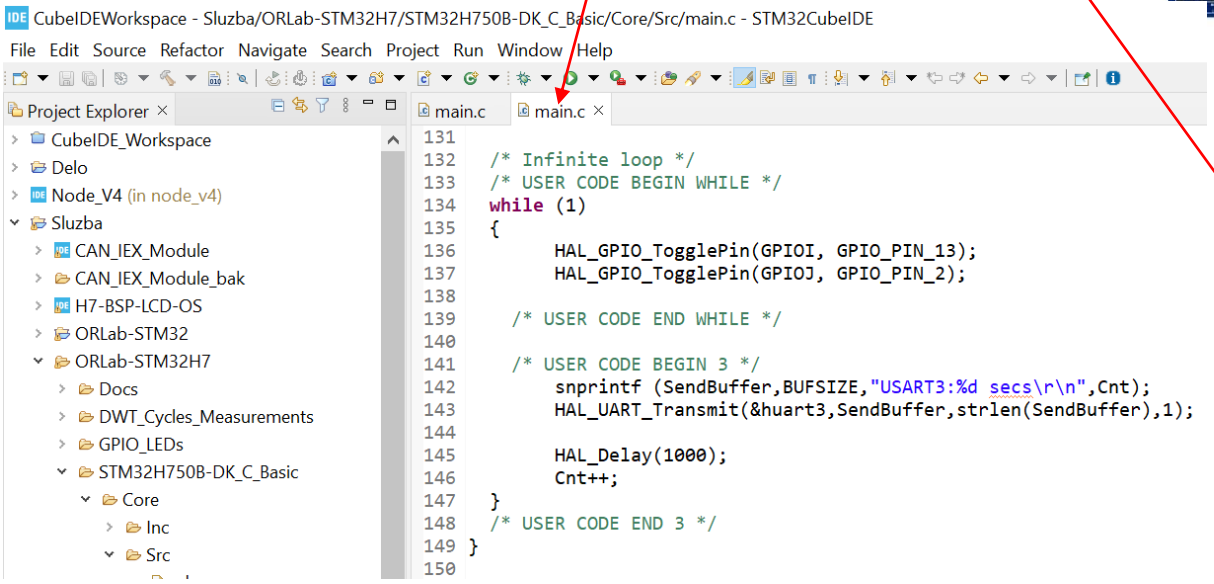
Mikro USB priklop na daljši stranici (srednji !!!)

Priključitev :

- **Mikro USB** priklop na **daljši stranici (srednji !!!)**

Poseben začetni projekt (github) in info za *STM32H7* (e-učilnica):

- ***dodajanje vsebine (main.c):***



--------- Razvojni sistem STM32H750-DK ----------

- STM32H750B-DK Discovery kit with STM32H750XB MCU
- VINLab-STM32H7 - GitHub repozitorij
- STM32H7-online training (tutorials from ST)
- ORLab-STM32H7 - GitHub repozitorij
- STM32H7 - Dokumentacija

Lastni viri :

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects

https://github.com/LAPSyLAB/ORLab-STM32H7

STM32H7

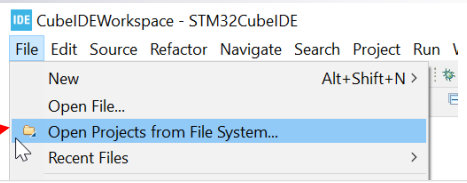
Delo v CubeIDE

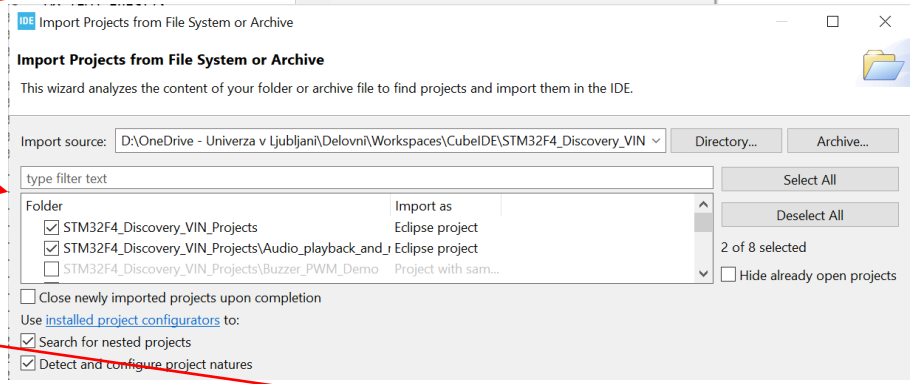
Vzpostavitev začetnega projekta :

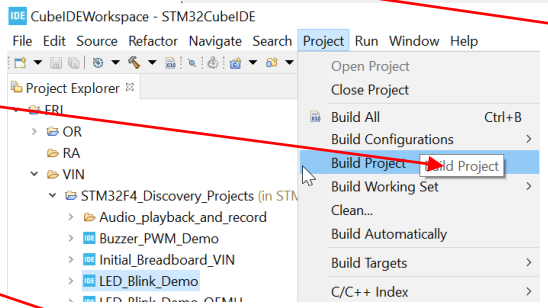
- **Uvoz obstoječega**
  - Open projects from File System
  - Select project(s)
- **Nov projekt Cube MX**
- **Kopiranje obstoječega**

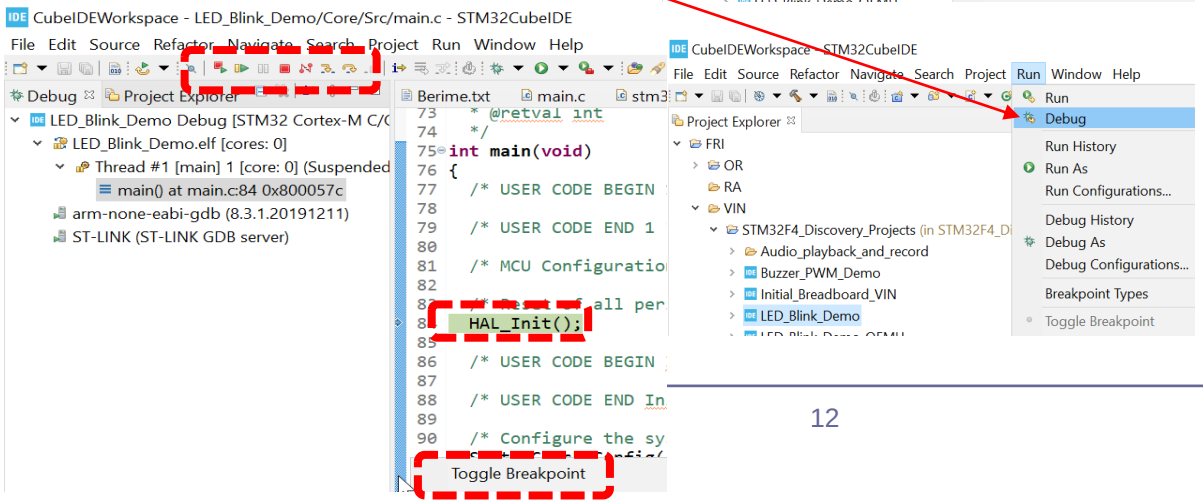
Prevajanje, zagon :

- Project -> Build Project
- Run -> Debug
- Step (Into,Over), Breakpoints









Navodila :

- CubeIDE asm projekt
  1) Edit > Copy.
  2) Edit > Paste.
  3) Delete the Debug.launch file.
  4) Project > Clean.
  5) Project > Build Project.
  6) Debug As Stm32 Application.
  7) And debug the application
  8) Add breakpoint on first instruction if necessary
- CubeIDE projekt z CubeMX
  1) Edit > Copy.
  2) Edit > Paste.
  3) Rename the ioc files.
  4) Delete the Debug.launch file.
  5) Project > Clean.
  6) Generate the CubeMX.
  7) Project > Build Project.
  8) Debug As Stm32 Application.
  9) And debug the application.

Skopiram, preimenujem, generiram ioc, clean in build

VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

Uporovna tipala

LDR – Light Dependent Resistor PGM5337

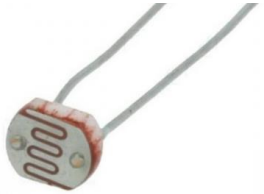


FOTO UPOR PGM5537 100mW 16-50kR 540nm

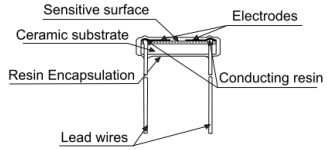
Electronics Characteristics

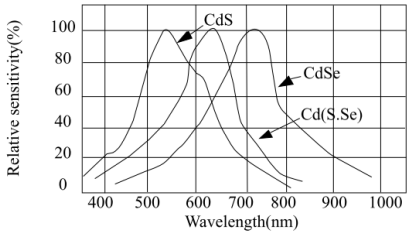
Model	Vmax (VDC)	Pmax (mW)	Ambient Temp (°C)	Spectral Peak (nm)	Photo Resistance (10Lx) (KΩ)	Dark Resistance (MΩ)min	γ min	ResponseTime (ms)	
								Rise	Decay
PGM5506	100	90	-30 ~ +70	540	2 ~ 6	0.15	0.6	30	40
PGM5516	100	90	-30 ~ +70	540	5 ~ 10	0.2	0.6	30	40
PGM5526	150	100	-30 ~ +70	540	8 ~ 20	1.0	0.6	20	30
PGM5537	150	100	-30 ~ +70	540	16 ~ 50	2.0	0.7	20	30

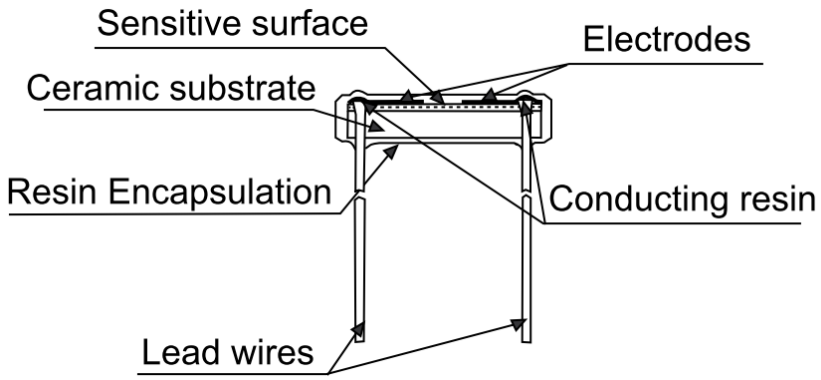
TOKEN **PGM CDS Photoresistors**

Terminology

- **Light Resistance :**
  Measured at 10 lux with standard light A (2854K-color temperature) and 2hr. preillumination at 400-600 lux prior testing.
- **Dark Resistance :**
  Measured at 10th seconds after closing 10 lux.
- **Gamma characteristic :**
  Under 10 lux and 100 lux and given by γ = log(R10/R100) / log(100/10) = log(R10/R100)
  R10, R100: resistance at 10 lux and 100 lux.
  The tolerance of γ is ±0.1.
- **Pmax :**
  Max. power dissipation at ambient temperature of 25°C.At higher ambient temperature,the maximum power permissible may be lowered.
- **Vmax :**
  Max. voltage in darkness that may be applied to the device continuously.
- **Spectral peak :**
  Spectral sensitivity of photoresistors depends on the wavelength of light they are exposed to and in accordance with figure 'Spectral Response'.
  The tolerance of spectral peak is ±50nm.







UPOR NTC 2K2 5%

Uporovna tipala

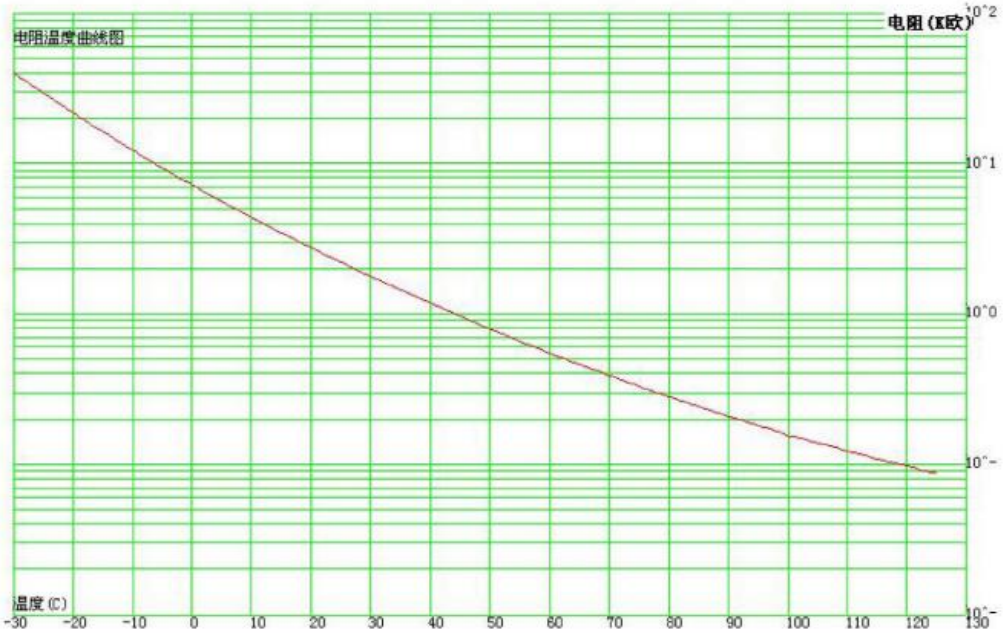
NTC – Termistor NTCC-2K2

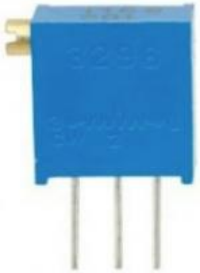
NTCC-2K2 SR PASSIVES
NTC thermistor; 2.2kΩ; THT; 3900K; -55÷125° C; 500mW; Ø6.5mm

R25 = 2.2kΩ

B25/50 = 3950K

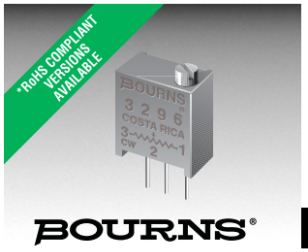
T°C	R kΩ	T°C	R kΩ	T°C	R kΩ	T°C	R kΩ	T°C	R kΩ	T°C	R kΩ
-30	40.128	-4	8.871	22	2.511	48	0.857	74	0.341	100	0.154
-29	37.694	-3	8.415	23	2.402	49	0.824	75	0.33	101	0.154
-28	35.418	-2	7.986	24	2.298	50	0.794	76	0.319	102	0.15
-27	33.289	-1	7.581	25	2.2	51	0.764	77	0.309	103	0.146
-26	31.298	0	7.238	26	2.105	52	0.736	78	0.3	104	0.142
-25	29.436	1	6.839	27	2.016	53	0.709	79	0.29	105	0.139
-24	27.693	2	6.499	28	1.931	54	0.683	80	0.281	106	0.135
-23	26.064	3	6.179	29	1.85	55	0.659	81	0.273	107	0.132
-22	24.539	4	5.876	30	1.773	56	0.635	82	0.265	108	0.129
-21	23.112	5	5.59	31	1.699	57	0.612	83	0.257	109	0.125
-20	21.776	6	5.32	32	1.629	58	0.59	84	0.249	110	0.122
-19	20.526	7	5.064	33	1.562	59	0.57	85	0.242	111	0.12
-18	19.355	8	4.823	34	1.498	60	0.55	86	0.234	112	0.117
-17	18.258	9	4.594	35	1.437	61	0.53	87	0.228	113	0.114
-16	17.231	10	4.378	36	1.379	62	0.512	88	0.221	114	0.111
-15	16.267	11	4.173	37	1.324	63	0.494	89	0.215	115	0.109
-14	15.364	12	3.979	38	1.271	64	0.477	90	0.208	116	0.106
-13	14.517	13	3.795	39	1.221	65	0.461	91	0.202	117	0.104
-12	13.722	14	3.62	40	1.172	66	0.445	92	0.197	118	0.102
-11	12.976	15	3.455	41	1.126	67	0.43	93	0.191	119	0.099
-10	12.275	16	3.298	42	1.082	68	0.416	94	0.186	120	0.097
-9	11.617	17	3.15	43	1.04	69	0.402	95	0.181	121	0.095
-8	10.999	18	3.008	44	1	70	0.389	96	0.176	122	0.093
-7	10.417	19	2.874	45	0.962	71	0.376	97	0.171	123	0.091
-6	9.87	20	2.747	46	0.925	72	0.364	98	0.167	124	0.089
-5	9.356	21	2.626	47	0.89	73	0.352	99	0.162	125	0.088





Uporovna tipala

TrimPot – Trimer Potenciometer TSR-3296Z-104



Features

- Multiturn / Cermet / Industrial / Sealed
- 5 terminal styles
- Tape and reel packaging available
- Chevron seal design
- Listed on the QPL for style RJ24 per MIL-R-22097 and RJR24 per High-Rel Mil-R-39035
- Mounting hardware available (H-117P)
- RoHS compliant* version available
- For trimmer applications/processing guidelines, click here

3296 - 3/8 " Square Trimpot® Trimming Potentiometer

Proizvajalec	Suntan
Številka proizvajalca	TSR-3296Z-104

Proizvajalec	Suntan
Številka proizvajalca	TSR-3296Z-104
Upornost	100KOhm
Moč	0.5W
Toleranca	± 10%
Tip	THT

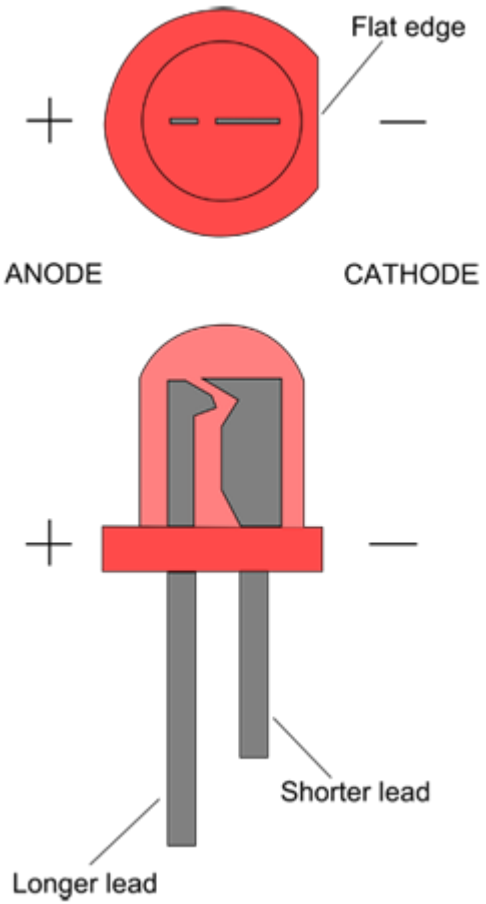
Standard Resistance Table

Resistance (Ohms)	Resistance Code
10	100
20	200
50	500
100	101
200	201
500	501
1,000	102
2,000	202
5,000	502
10,000	103
20,000	203
25,000	253
50,000	503
100,000	104

https://www.ic-elect.si/trimpot-cer-64z-100k-tsr-3296z.html

Elektronske komponente

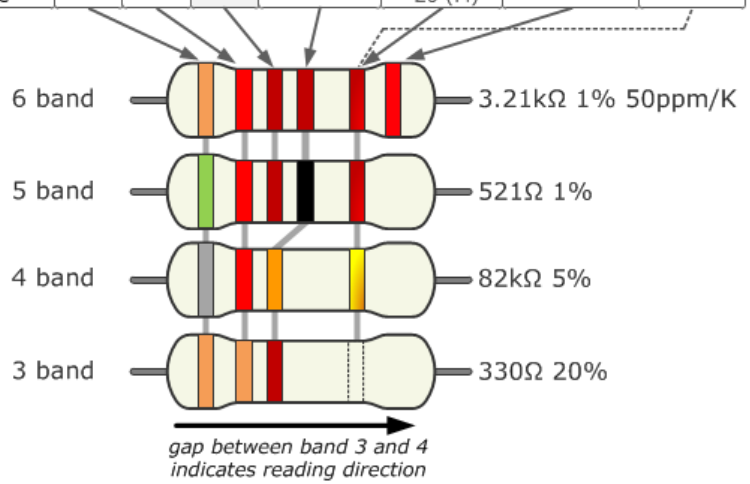
Dioda LED



Uporniki

www.resistorguide.com

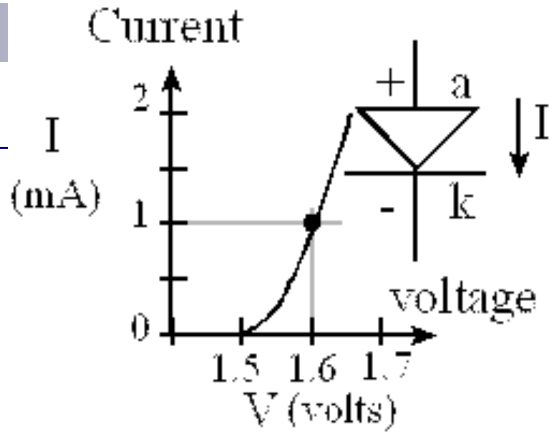
	Color	Signficant figures			Multiply	Tolerance (%)	Temp. Coeff. (ppm/K)	Fail Rate (%)
Bad	black	0	0	0	x 1		250 (U)	
Beer	brown	1	1	1	x 10	1 (F)	100 (S)	1
Rots	red	2	2	2	x 100	2 (G)	50 (R)	0.1
Our	orange	3	3	3	x 1K		15 (P)	0.01
Young	yellow	4	4	4	x 10K		25 (Q)	0.001
Guts	green	5	5	5	x 100K	0.5 (D)	20 (Z)	
But	blue	6	6	6	x 1M	0.25 (C)	10 (Z)	
Vodka	violet	7	7	7	x 10M	0.1 (B)	5 (M)	
Goes	grey	8	8	8	x 100M	0.05 (A)	1(K)	
Well	white	9	9	9	x 1G			
Get	gold			3th digit only for 5 and 6 bands	x 0.1	5 (J)		
Some	silver				x 0.01	10 (K)		
Now!	none					20 (M)		

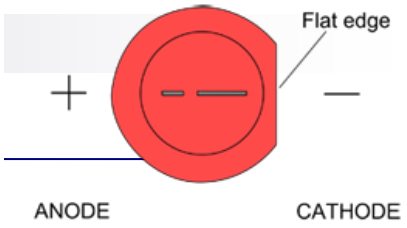


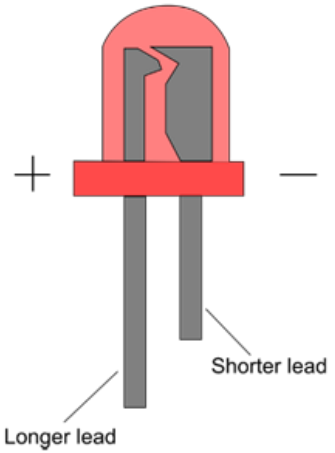
Resistor Color Code Calculator and Chart (4-band, 5-band or 6-band)

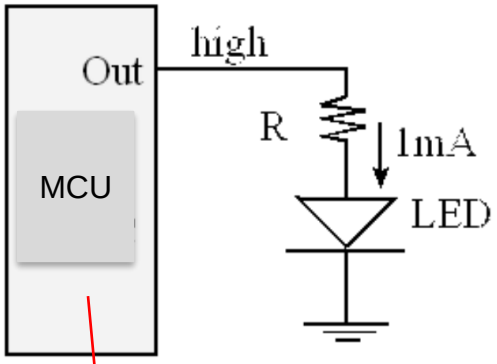
Z naslova <https://www.allaboutcircuits.com/tools/resistor-color-code-calculator/>

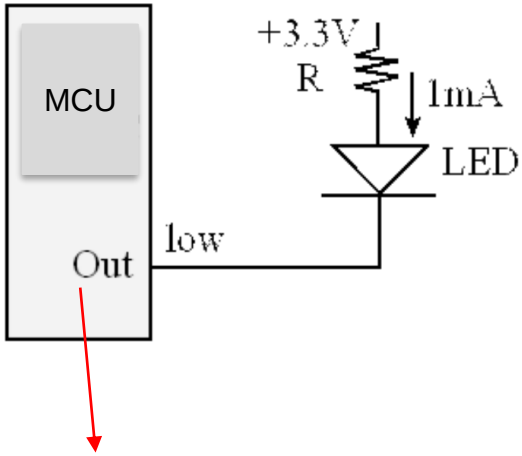
Dioda LED
Izračun upora za omejitev toka skozi tokom











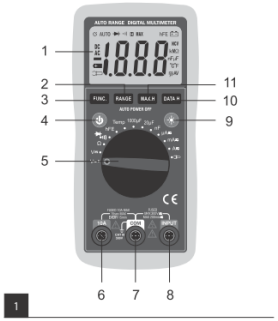
$$R = \frac{V_{OH} - V_d}{I_d} = \frac{2.4 - 1.6}{0.001} = 800\,\Omega$$

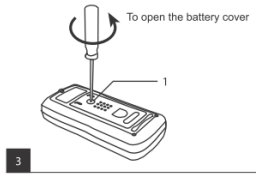
$$R = \frac{3.3 - V_d - V_{OL}}{I_d} = \frac{3.3 - 1.6 - 0.4}{0.001} = 1.3\,k\Omega$$

Z naslova <http://users.ece.utexas.edu/~valvano/Volume1/E-Book/C8_SwitchLED.htm>

Orodja

Multimeter EMOS MD-420





Preverjanje povezav

Merjenje upornosti

Merjenje el. napetosti
- enosmerna DC
- vzporedna vezava !!!
- visoka upornost

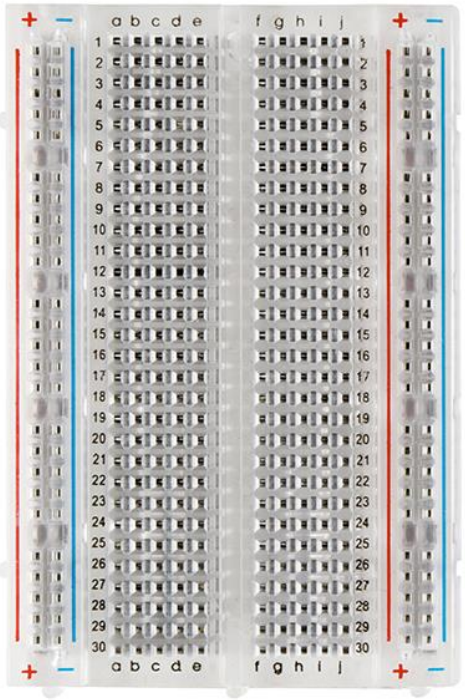
Merjenje el. toka
- zaporedna vezava !!!
- nizka upornost

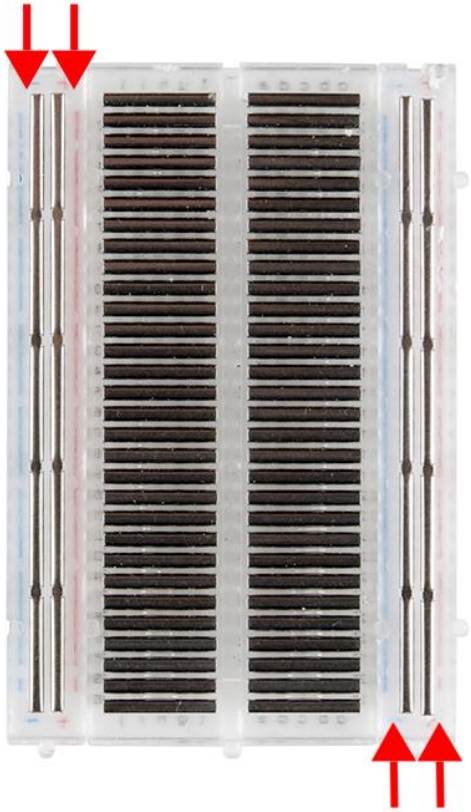
Praktični nasveti :
- *Večinoma merimo napetost, upornost*
- *upornost samo izven tokokroga*
- *pazimo, da ne sklenemo kratkega stika z merilno sondo*
- *pazimo predvsem na majhne upornosti:*
  - *Med +V in GND*
  - *Na izhodih, vhodih mikrokrmilnikov*

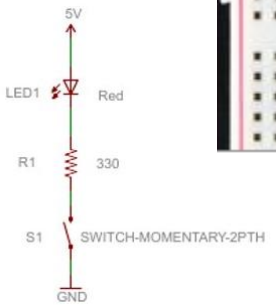
https://www.emos-si.si/multimeter-md-420

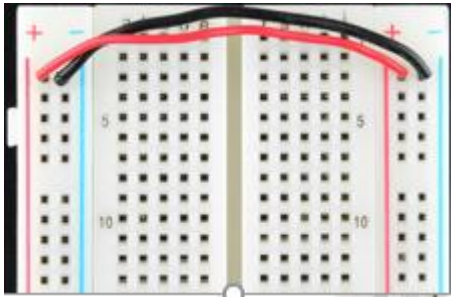
Ponovitev

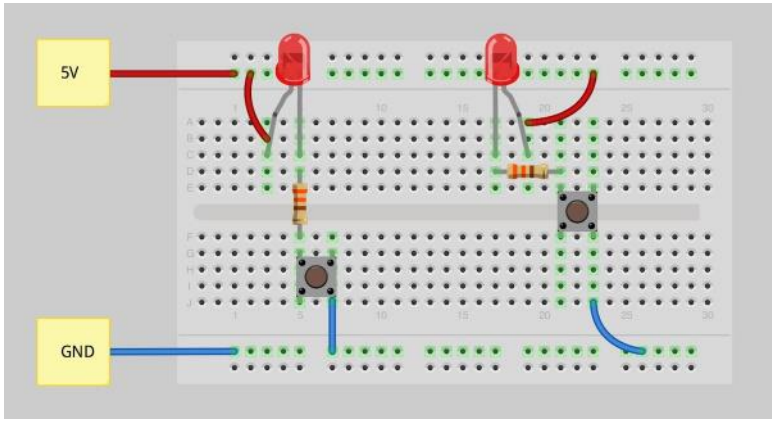
VIN projekt : TinkerCad
Breadboard vezave







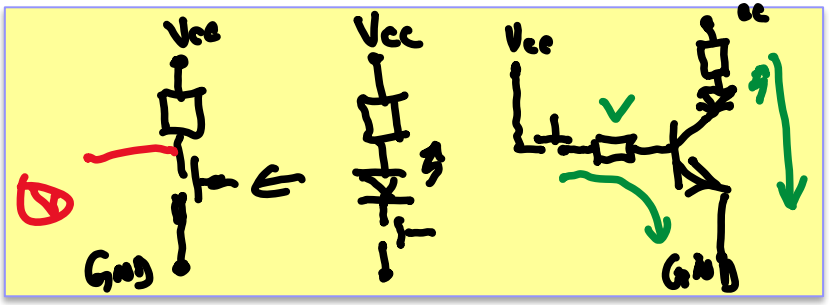


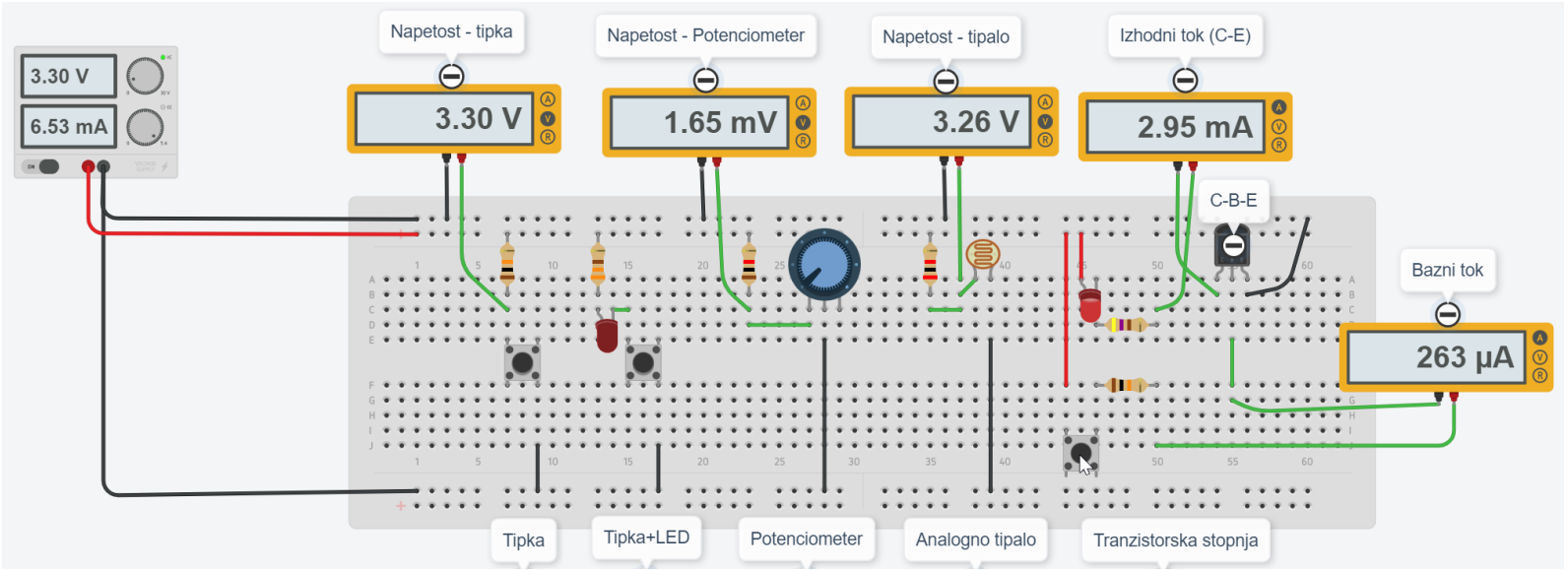


Viri

- https://learn.sparkfun.com/tutorials/how-to-use-a-breadboard/
- https://www.sciencebuddies.org/science-fair-projects/references/how-to-use-a-breadboard

VIN projekt : TinkerCad – LAB 2
Breadboard vezave – VP2 primeri vezav

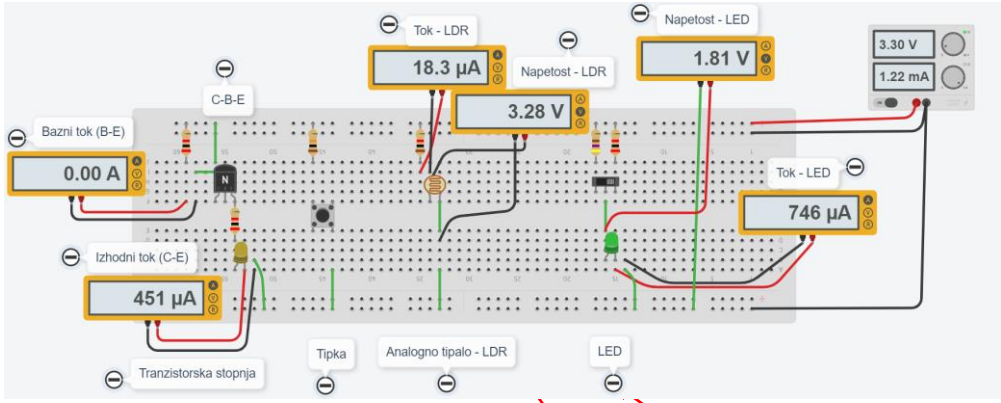


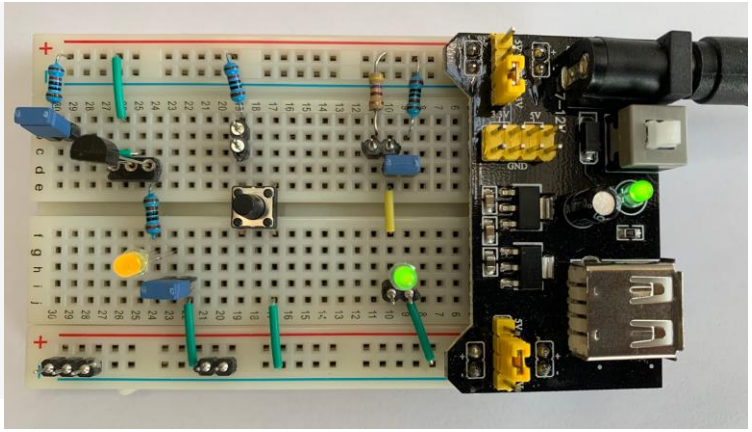


https://www.tinkercad.com/things/9o9pEq418de

Breadboard vezave – izhodišči za delo :

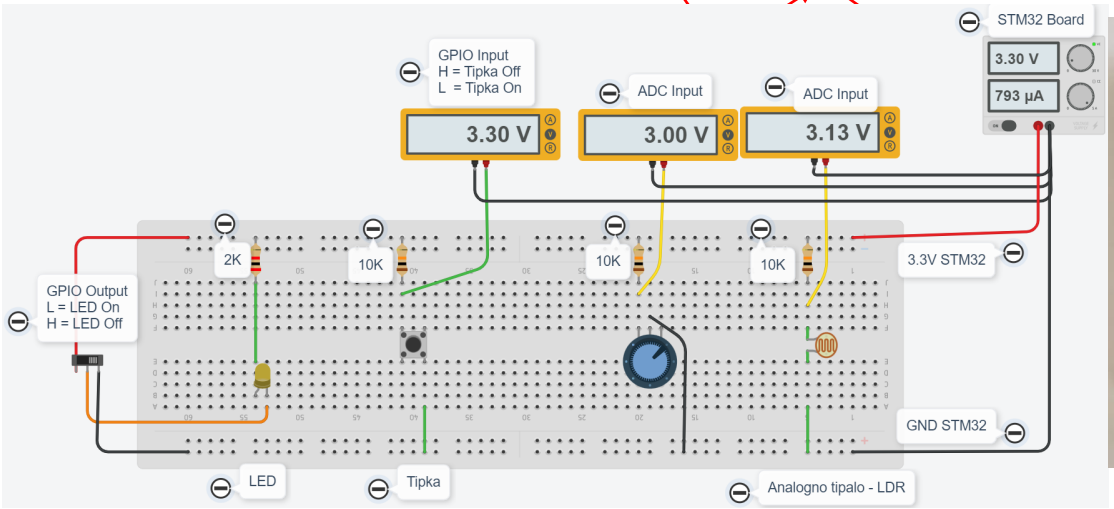
- „VIN LAB Breadboard Demo“ : demo breadboard „merilna“ (napajana) vezava (za meritve, poskuse, …)

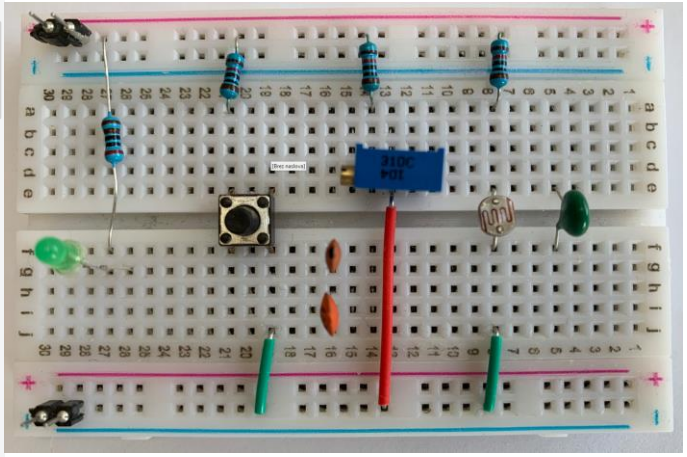




https://www.tinkercad.com/things/1UQpxVO5DSY

- „VIN LAB Breadboard STM32 IO Demo“ : demo breadboard vezava za povezavo s STM32 sistemoma

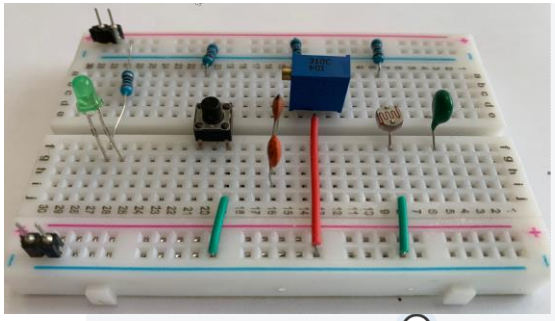




https://www.tinkercad.com/things/cZld7zNU6Yd

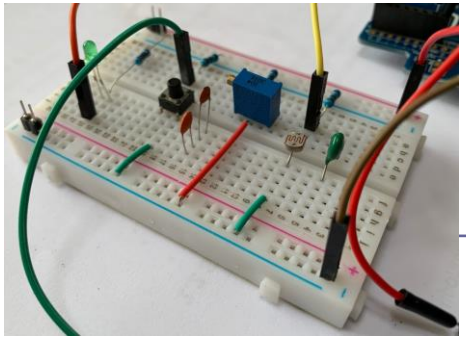
Breadboard vezava

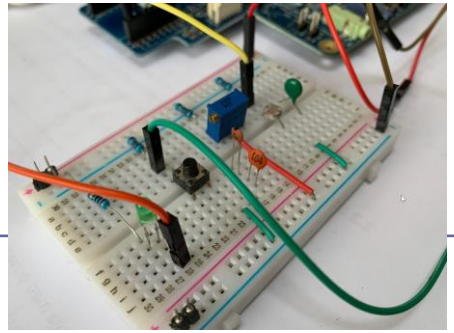
- „VIN LAB Breadboard STM32 IO Demo“ : demo breadboard vezava za povezavo s STM32 sistemoma



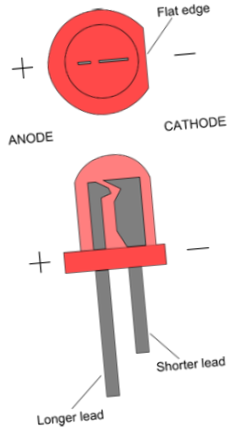
https://www.tinkercad.com/things/cZld7zNU6Yd

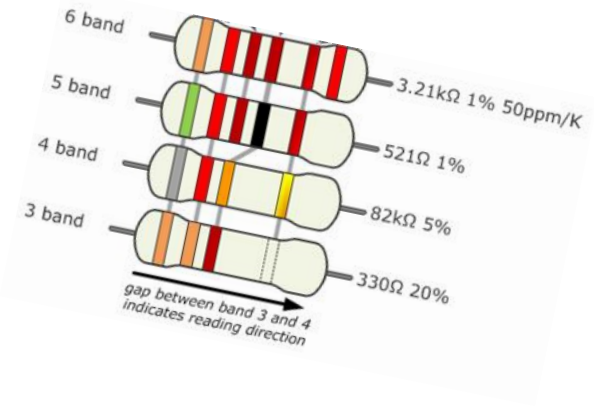
Priključitev na STM32 : ADC: 1x analogni, GPIO: 1x digitalni vhod (GPIO), 1x digitalni izhod





Breadboard vezava - pripomočki





Osnovna priporočila za potek praktičnega dela vaje :

Napajanje 3V !!!
Priključite nazadnje !

- z multimetri najprej preverite
  - „napajano“ vezavo - „VIN LAB Breadboard Demo“ (po želji)
  - vezavo - „VIN LAB Breadboard STM32 IO Demo“

- ***1. Vezava na breadboardu:***
  - v dvoje izvedite lastno vezavo (GPIO: tipka, led, ADC: uporovno tipalo)
    - previdno priključujte žice, konektorje

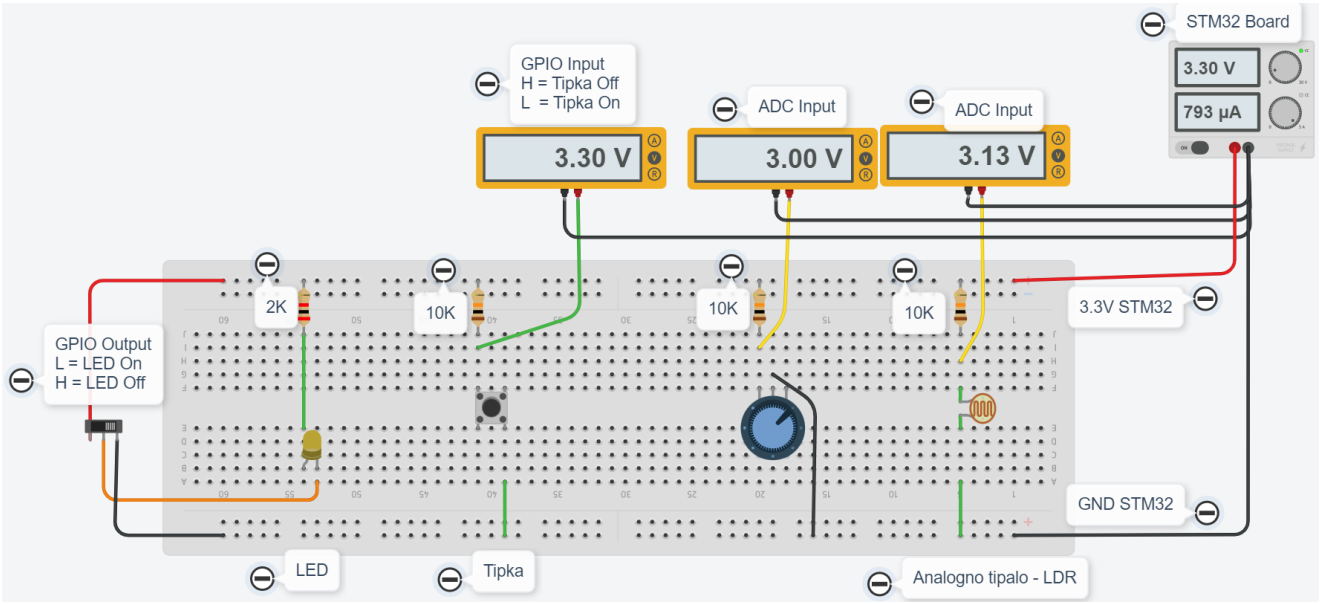
- ***2. Preveritev vezave:***
  - z multimetri še brez povezave s STM32 sistemom
    - preverite posamezne komponente pred vezavo (upornosti, …)
    - po vezavi: posamezne veje, stike, povezave, upornosti med Vcc in GND
  - preverite logiko in pravilnost povezav
    - preverite tudi slike povezanih sistemov

- ***3. Povezava s STM32 in programiranje:***
  - povežite s STM32 sistemom (naj bo izkopljen)
  - STM32: vklop in delo na STM32 programu

VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

Izhodišče : **VIN LAB Breadboard STM32 IO Demo**



Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Testno vezje (primer) :

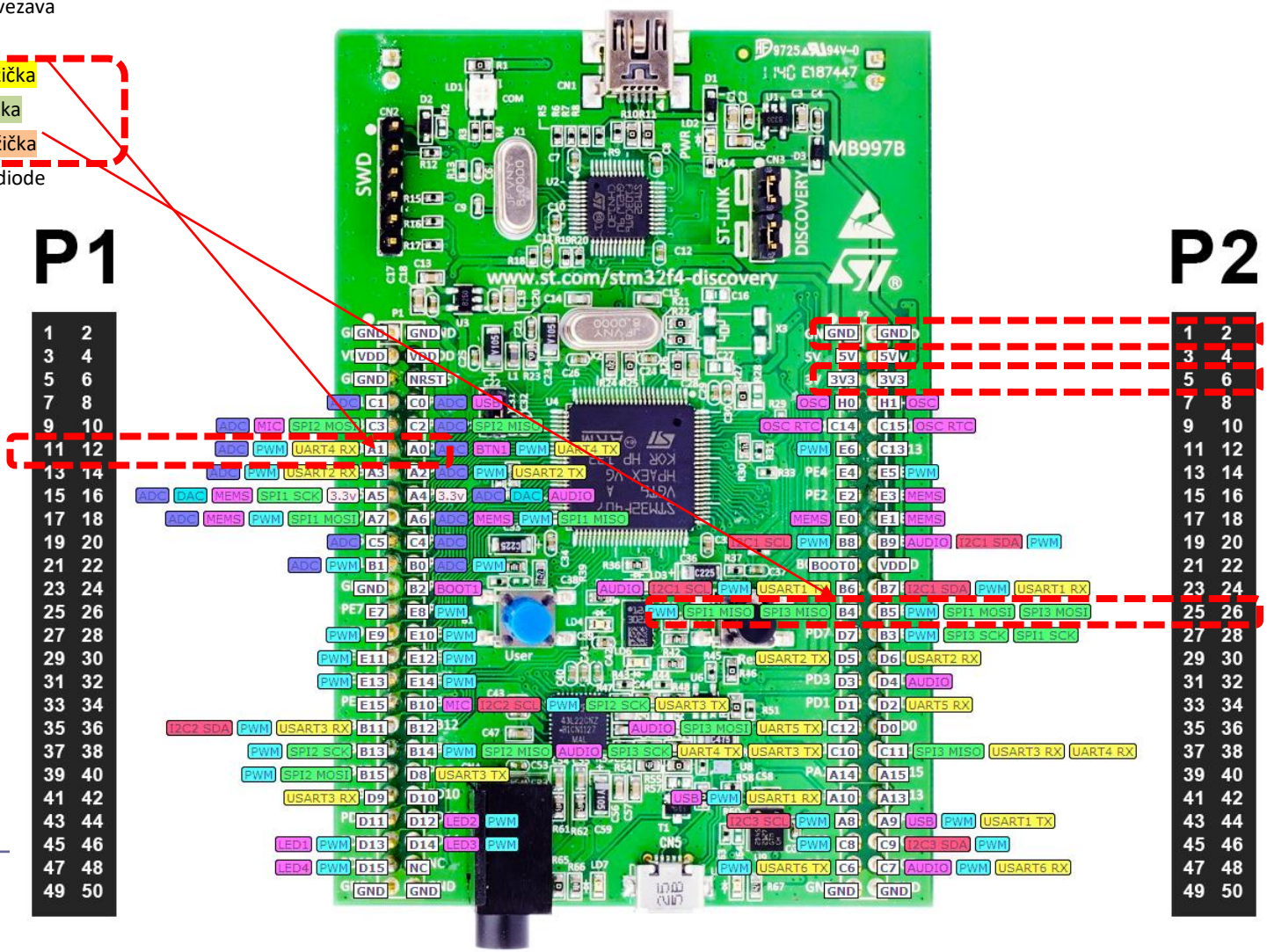
GPIO	Vrsta	Povezava
PA0	User tipka	
PA1	Analogni vhod	Rumena žička
PB4	Dig. Vhod	Zelena žička
PB5	Dig. Izhod - LED	Oranžna žička
PD12-PD15	Dig. Izhodi	vgr. LED diode

Breadboard vezava

Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Testno vezje (primer) :

GPIO	Vrsta	Povezava
PA0	User tipka	
PA1	Analogni vhod	Rumena žička
PB4	Dig. Vhod	Zelena žička
PB5	Dig. Izhod - LED	Oranžna žička
PD12-PD15	Dig. Izhodi	vgr. LED diode

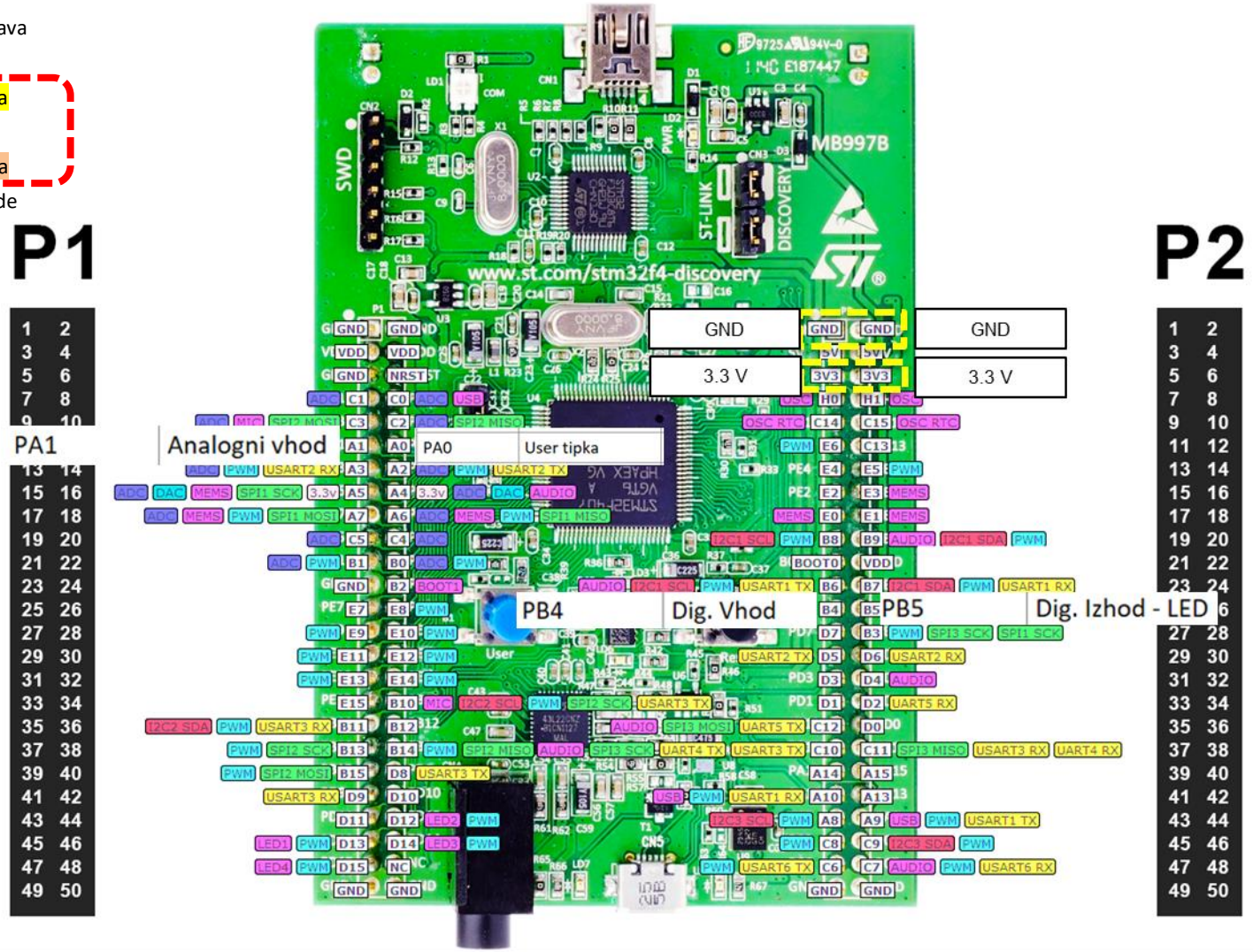


Breadboard vezava

Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Testno vezje (primer) :

GPIO	Vrsta	Povezava
PA0	User tipka	
PA1	Analogni vhod	Rumena žička
PB4	Dig. Vhod	Zelena žička
PB5	Dig. Izhod - LED	Oranžna žička
PD12-PD15	Dig. Izhodi	vgr. LED diode

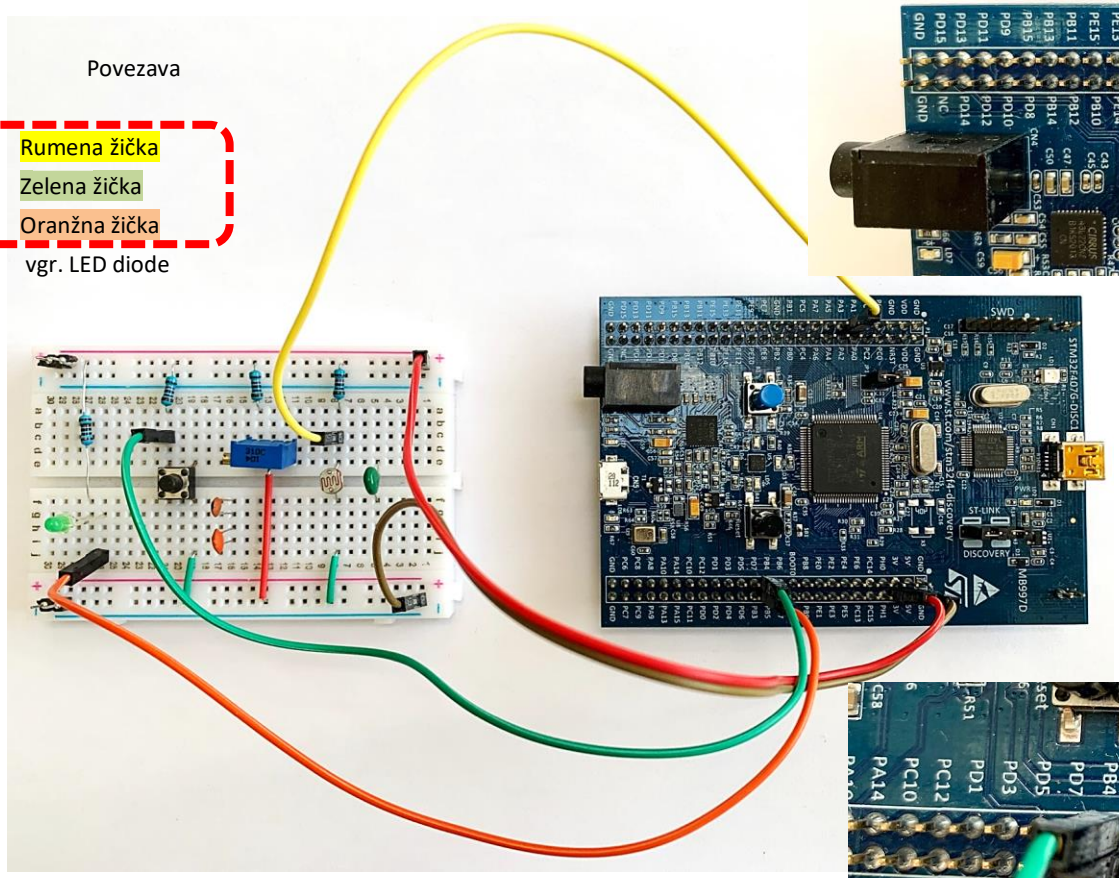


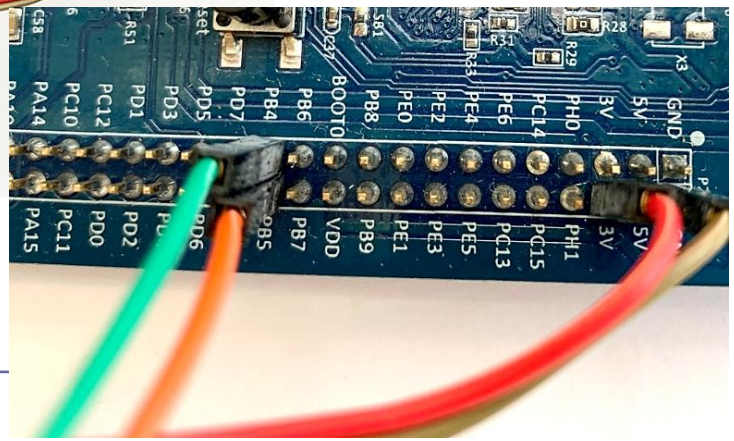
Breadboard vezava

Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Testno vezje (primer) :

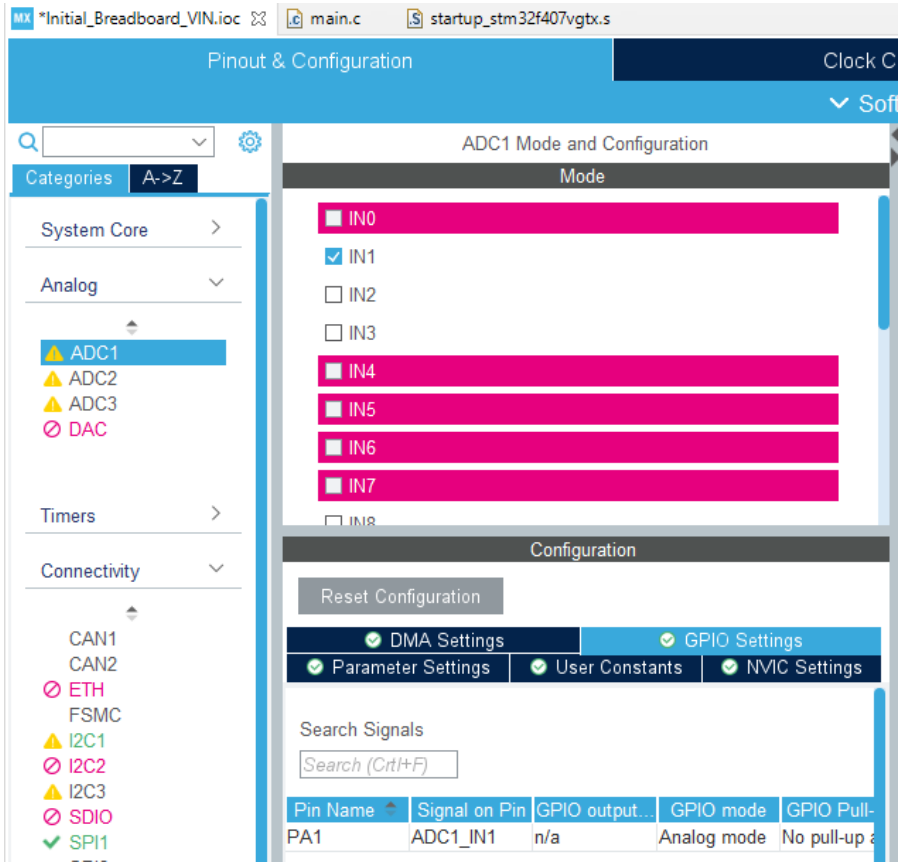
GPIO	Vrsta	Povezava
PA0	User tipka	
PA1	Analogni vhod	Rumena žička
PB4	Dig. Vhod	Zelena žička
PB5	Dig. Izhod - LED	Oranžna žička
PD12-PD15	Dig. Izhodi	vgr. LED diode

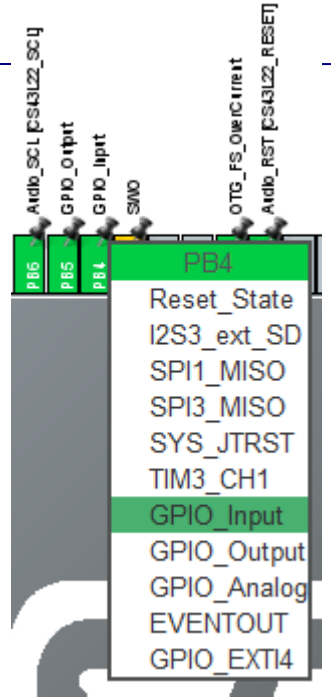


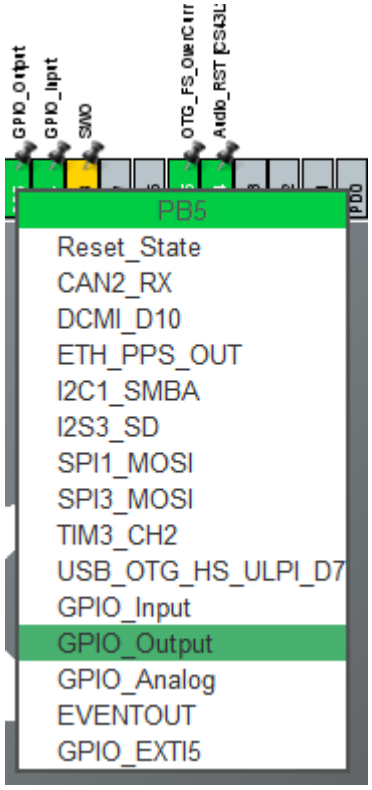


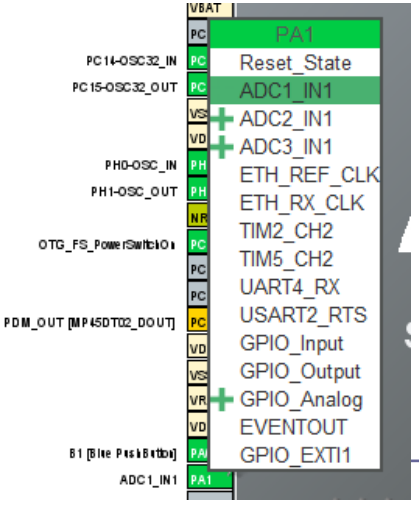
VIN projekt - P5.2 : CubeIDE in STM32

Konfiguracija 2: (PB5 DIG_OUT, PB4 DIG_INP, PA1 ADC1_IN1)





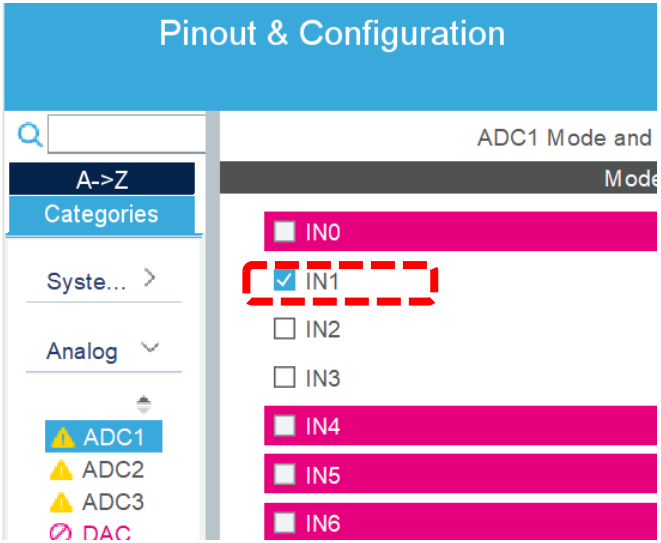


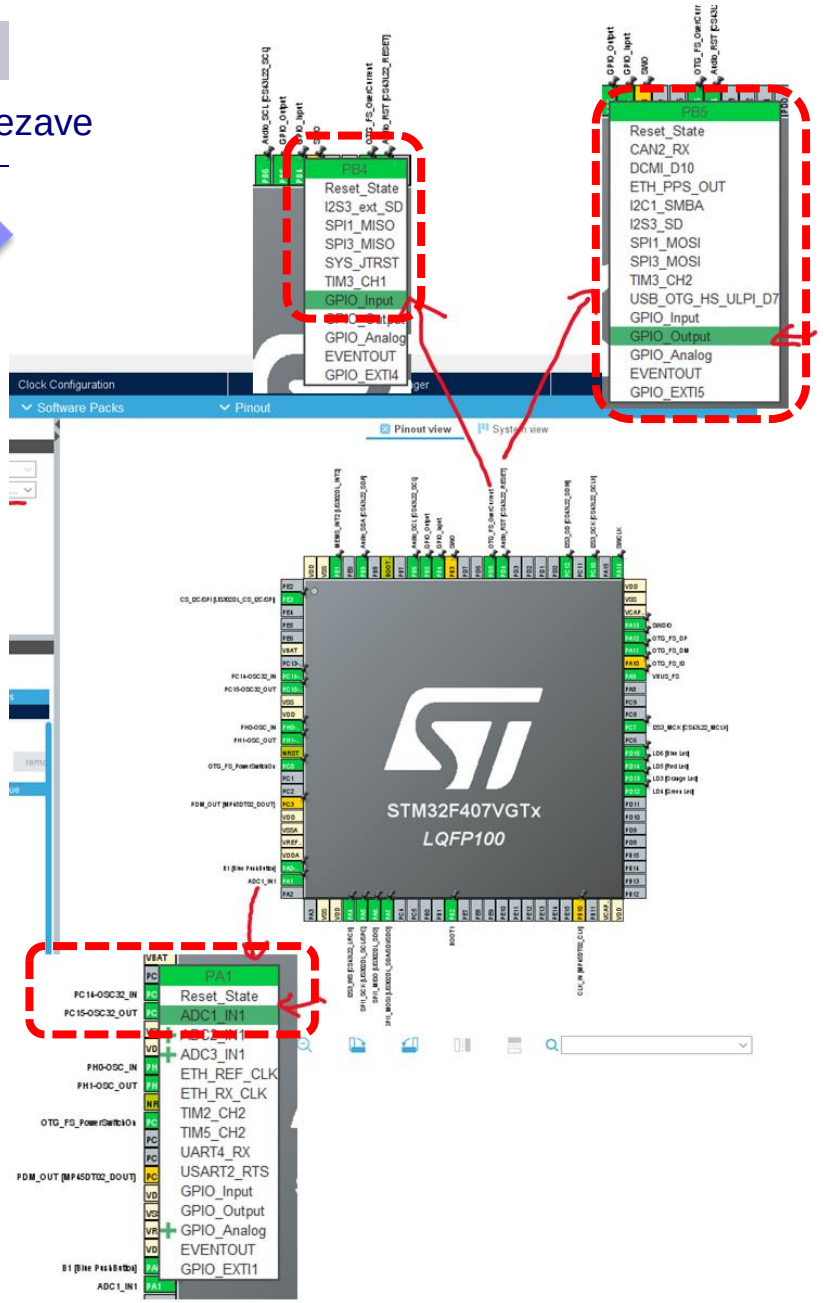


VIN projekt - VP 5 STM32-CubeIDE projekt, breadboard vezave

Spremembe v projektu :

STM32F4





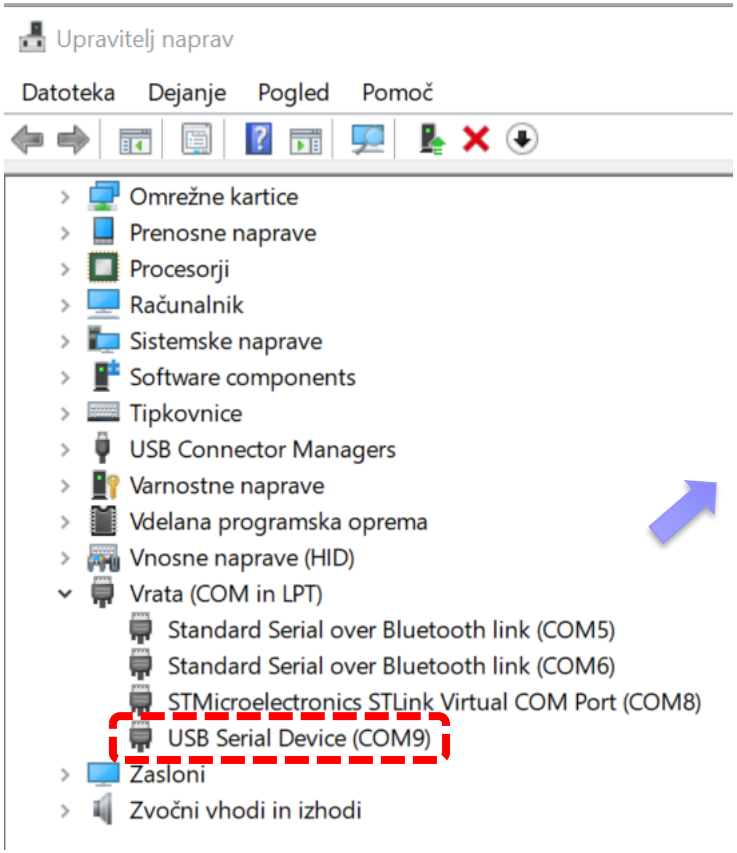
VIN projekt - VP 5 STM32-CubeIDE projekt, breadboard vezave

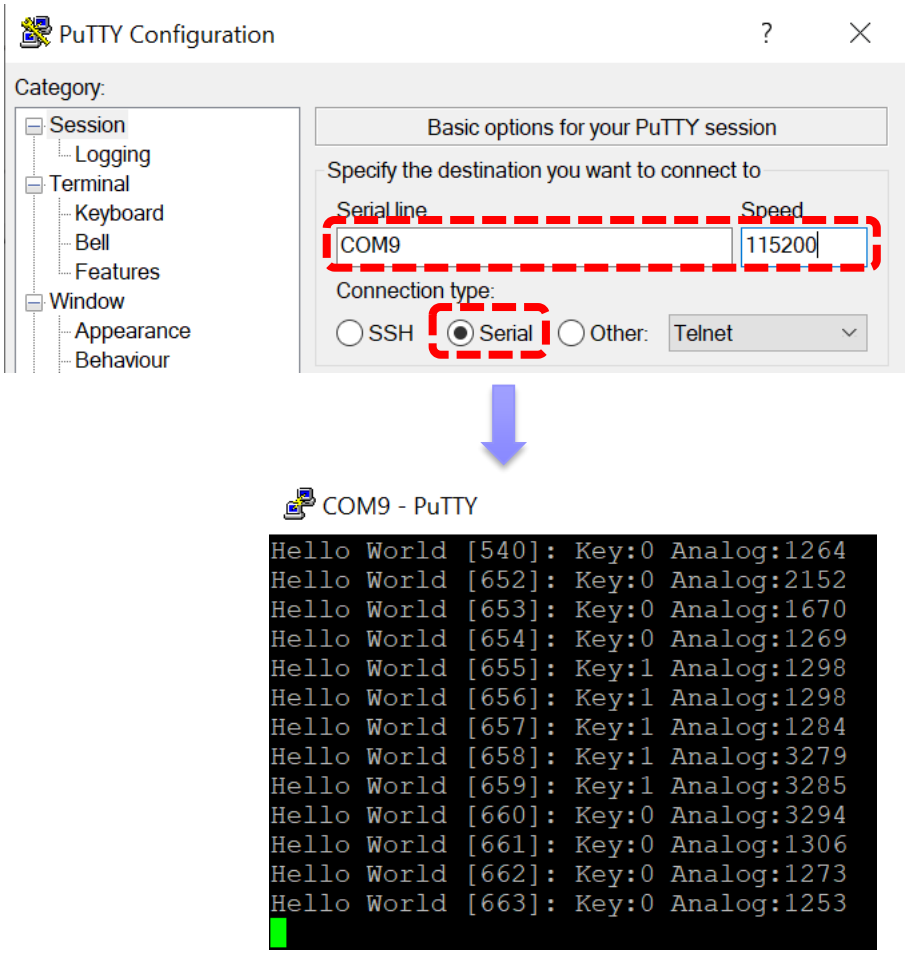
Program : za branje tipal in pošiljanje po USB Virtual COM Port

```
/* Infinite loop */
 /* USER CODE BEGIN WHILE */
 while (1)
 {

    HAL_ADC_Start(&hadc1);
    HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
    AnalogValue = HAL_ADC_GetValue(&hadc1);

    HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);   // On-board LED

    HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_5);   //External LED on PB5
    KeyState = HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_4); //External Key on PB4

    snprintf (SendBuffer,BUFSIZE,"Hello [%d]: Key:%d Analog:%d\r\n",Counter++, 1-KeyState, AnalogValue);
    CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
        HAL_Delay(1000);
 }
 /* USER CODE END 3 */
```

Osnovni projekt CubeIDE – USB Virtual COM Port

Program : sprejem na PC strani (povezava z dodatnim Micro-USB kablom)





VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

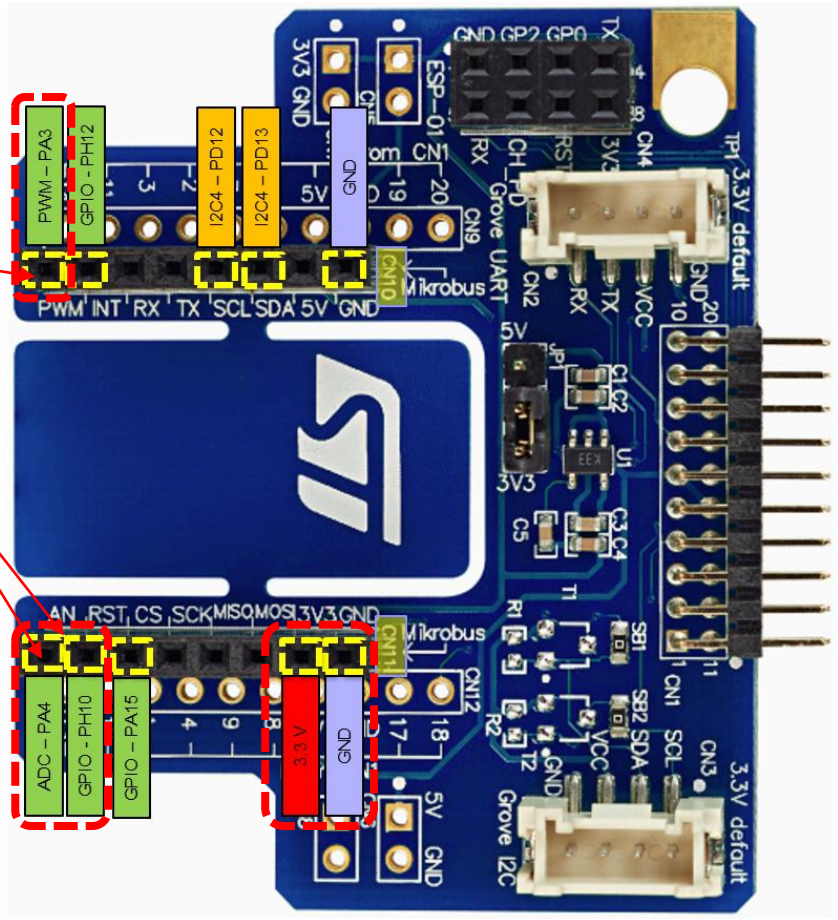
- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

Breadboard vezava

Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Testno vezje (primer) - STM32H7 :

GPIO	Vrsta	Povezava
PC13	User tipka	Modra tipka
PA4	Analogni vhod	Rumena žička
PH10	Dig. Vhod	Zelena žička
PA3	Dig. Izhod - LED	Oranžna žička
PJ2,Pi13	Dig. Izhodi	vgr. LED diode

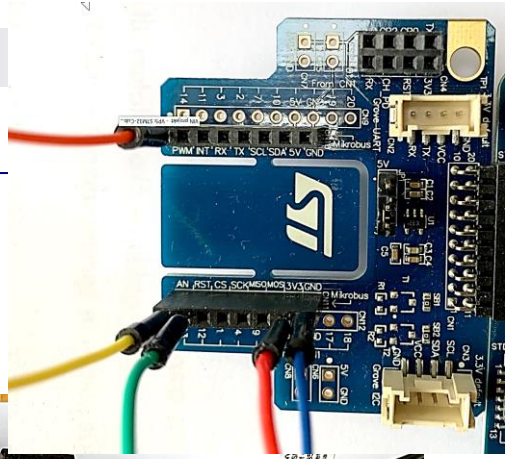


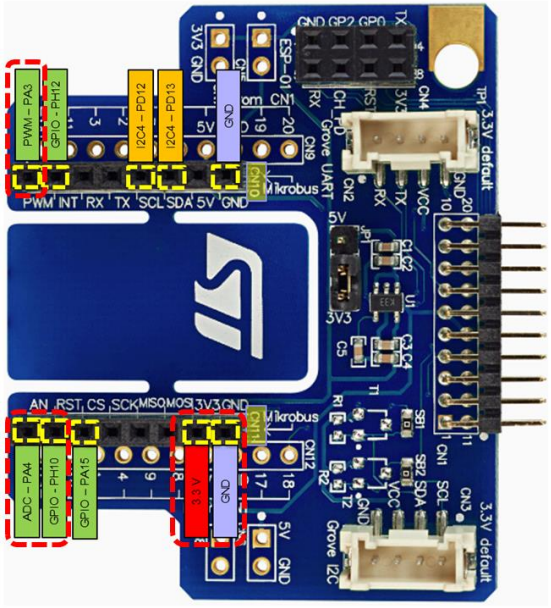
Breadboard vezava

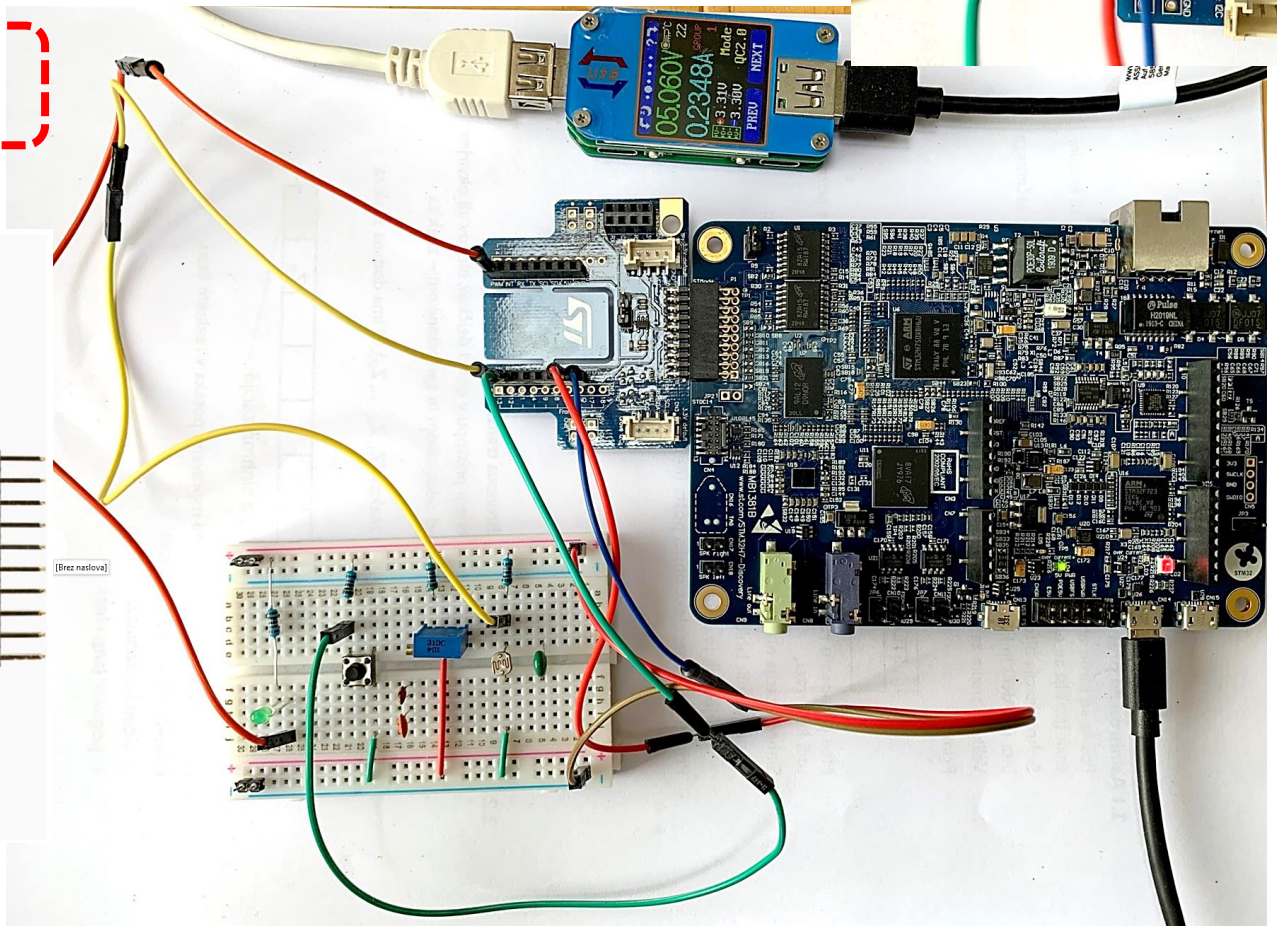
Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod

- Testno vezje (primer) - STM32H7 :

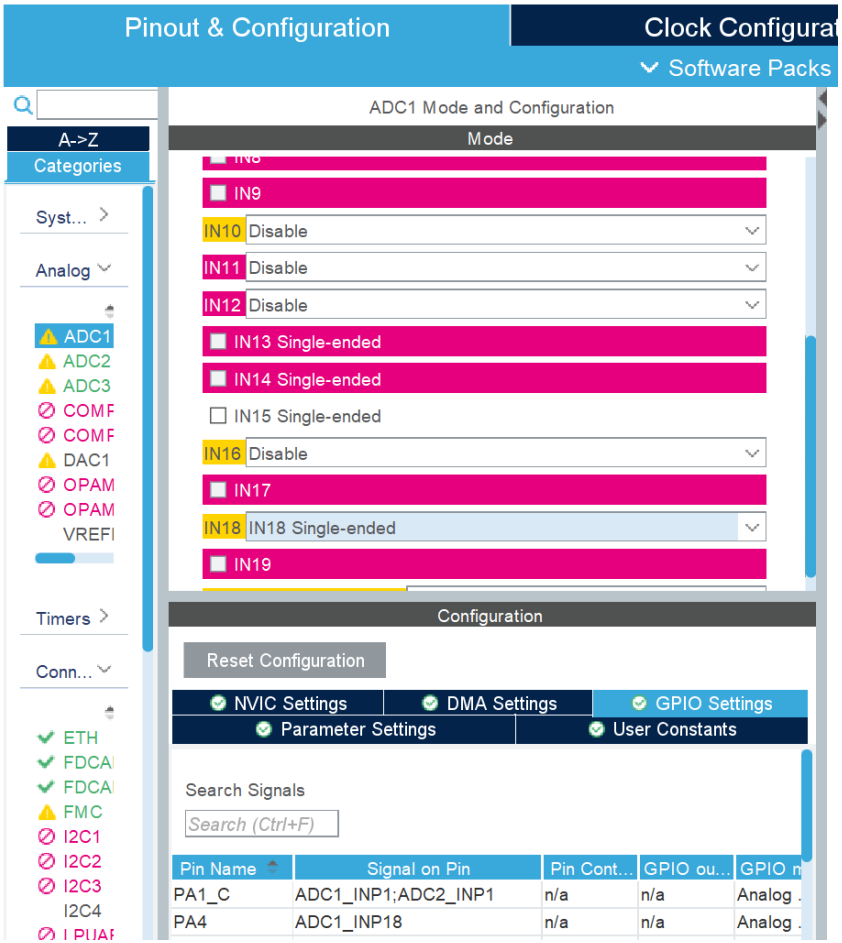
GPIO	Vrsta	Povezava
PC13	User tipka	Modra tipka
PA4	Analogni vhod	Rumena žička
PH10	Dig. Vhod	Zelena žička
PA3	Dig. Izhod - LED	Oranžna žička
PJ2,Pi13	Dig. Izhodi	vgr. LED diode

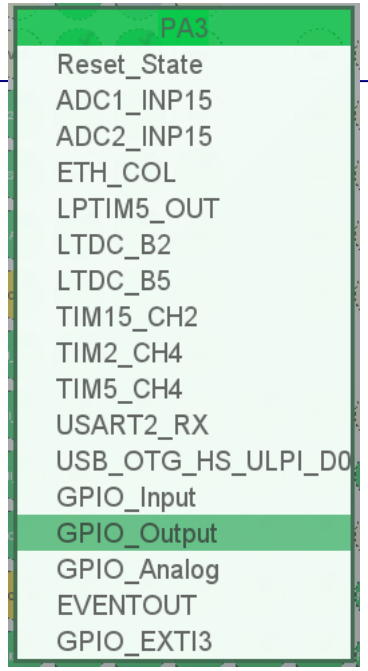


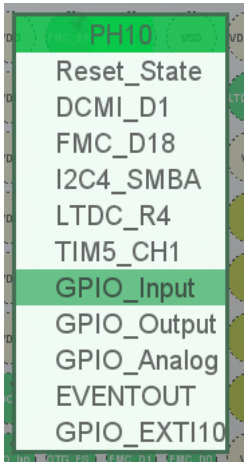




Konfiguracija 2: (PB5 DIG_OUT, PB4 DIG_INP, PA1 ADC1_IN1)







Testno vezje (primer) - STM32H7 :

GPIO	Vrsta	Povezava
PC13	User tipka	Modra tipka
PA4	Analogni vhod	Rumena žička
PH10	Dig. Vhod	Zelena žička
PA3	Dig. Izhod - LED	Oranžna žička
PJ2,Pi13	Dig. Izhodi	vgr. LED diode

VIN projekt - VP 5 STM32-CubeIDE projekt, breadboard vezave

Program : za branje tipal in pošiljanje po USB Virtual COM Port

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
char SendBuffer[BUFSIZE];
int Counter;
int KeyState=0;
int AnalogValue;

/* USER CODE END PV */
```

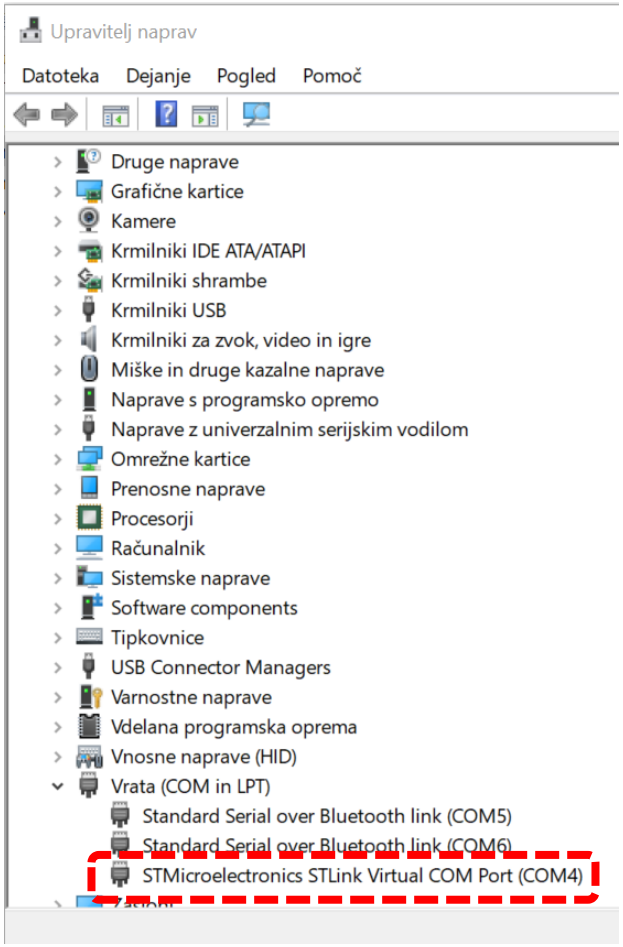
```
/* Infinite loop */
 /* USER CODE BEGIN WHILE */
 while (1)
 {
   HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13);

   HAL_ADC_Start(&hadc1);
   HAL_ADC_PollForConversion(&hadc1, HAL_MAX_DELAY);
   AnalogValue = HAL_ADC_GetValue(&hadc1);          // Read ADC value on analog input


   KeyState = HAL_GPIO_ReadPin(GPIOH, GPIO_PIN_10);   // Read state of PH10
   HAL_GPIO_WritePin(GPIOA, GPIO_PIN_3, KeyState);    // Write to PA3 accordingly


   snprintf(SendBuffer,BUFSIZE,"Hello World [%d]: Key:%d | ADC:%d\n\r",Counter++,KeyState,AnalogValue);
   HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);

   HAL_Delay(1000);
   /* USER CODE END WHILE */

   /* USER CODE BEGIN 3 */

 }
 /* USER CODE END 3 */
```

Testno vezje (primer) - STM32H7 :

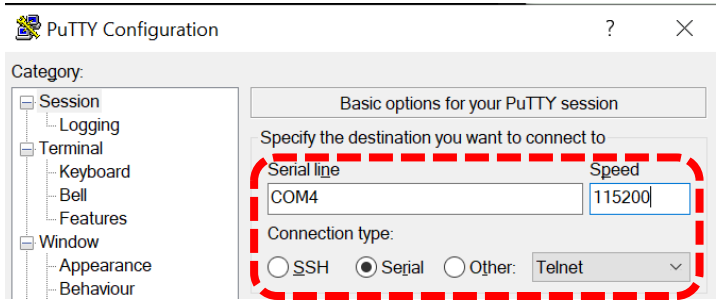
GPIO	Vrsta	Povezava
PC13	User tipka	Modra tipka
PA4	Analogni vhod	Rumena žička
PH10	Dig. Vhod	Zelena žička
PA3	Dig. Izhod - LED	Oranžna žička
PJ2,Pi13	Dig. Izhodi	vgr. LED diode

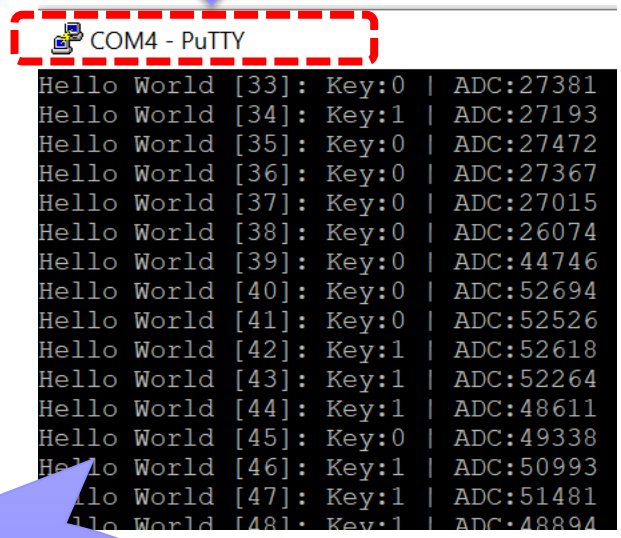
Osnovni projekt CubeIDE – USB Virtual COM Port (USART3 na STM strani)

Program : sprejem na PC strani (povezava že vzpostavljena z Micro-USB kablom)



https://the.earth.li/~sgtatham/putty/latest/w64/putty.exe





Test serial comm.

VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

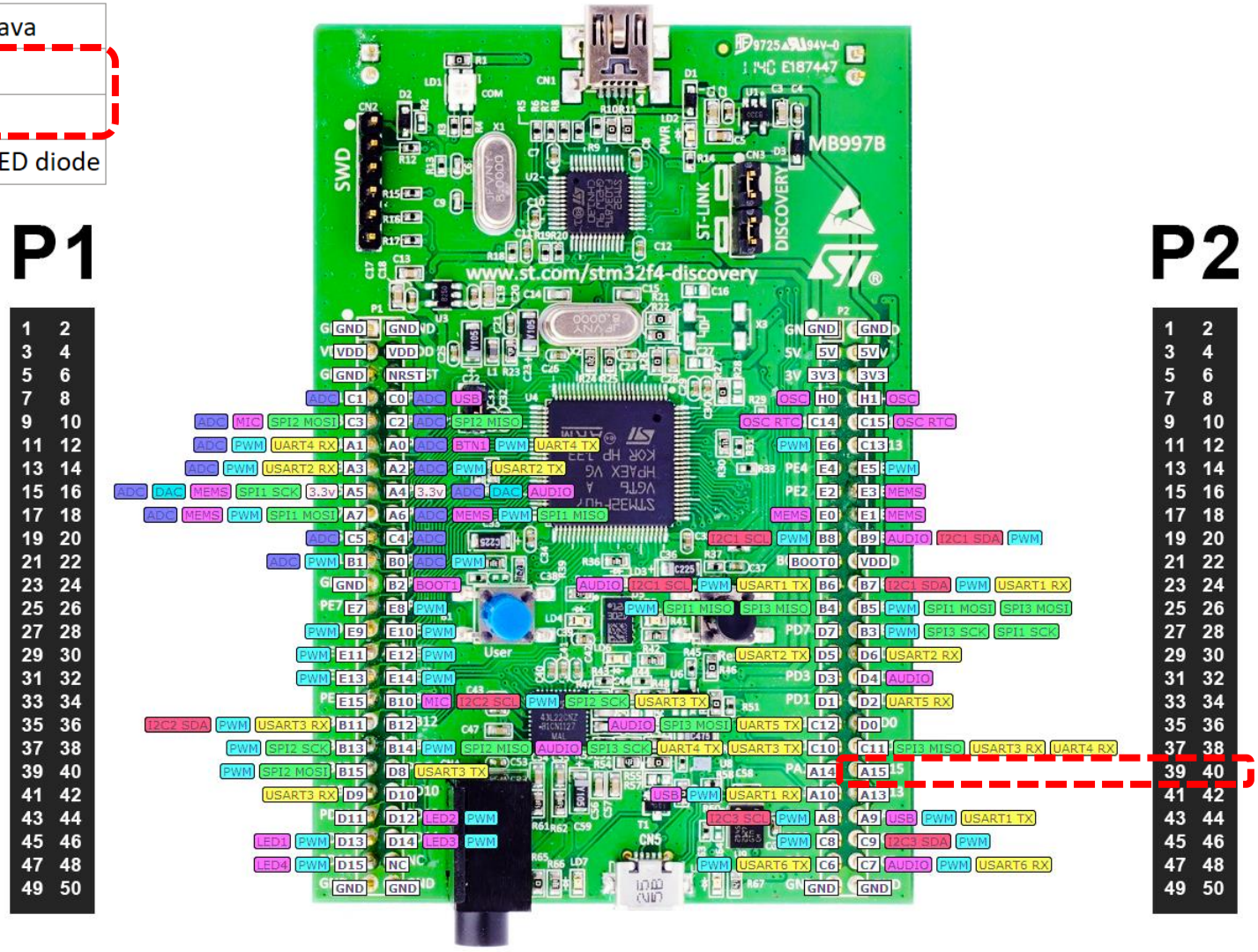
- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

Vezava brenčača

Priključitev na STM32 : 1x PWM izhod, 1x GND, 4x vgrajene LED diode

Testno vezje (primer) :

GPIO	Vrsta	Povezava
PA15	Brencac	+
GND	Brencac	-
PD12-PD15	Dig. Izhodi	vgr. LED diode



STM32F4 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

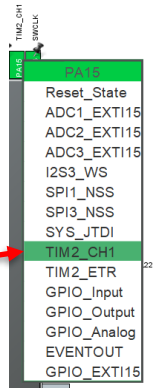
Brencac se prikljuci na **PA15 (TIM2->CH1) in GND**

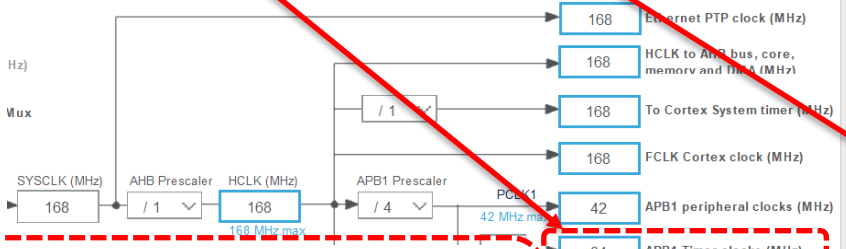
CubeMX :
1. New project -> STM32 Project -> Board -> 407DISC1
2. CubeMX: Spremeniti USB Host v USB Device :
Connectivity -> USB_OTG_FS -> Mode v Device Only
Middleware -> DEVICE_USB in Class Virtual Com Port

3. Spremeniti pin PA15 v TIM2->CH1
tim2 kanal 1 spremeniti na PWM Generation CH1

4. Clock :
Ura števca = 1 MHz
Prescaler (PSC - 16 bits value) = **84-1 = 83 (clock 1Mhz)**
Perioda štetja se bo določala glede na noto (duty cycle je vedno 50%)
Counter Period (AutoReload Register - 16 bits value)
ARR = 1000000 (ura števca) / Frekv.note[Hz]
CCR1 bo vedno ARR/2 (50% duty cycle)

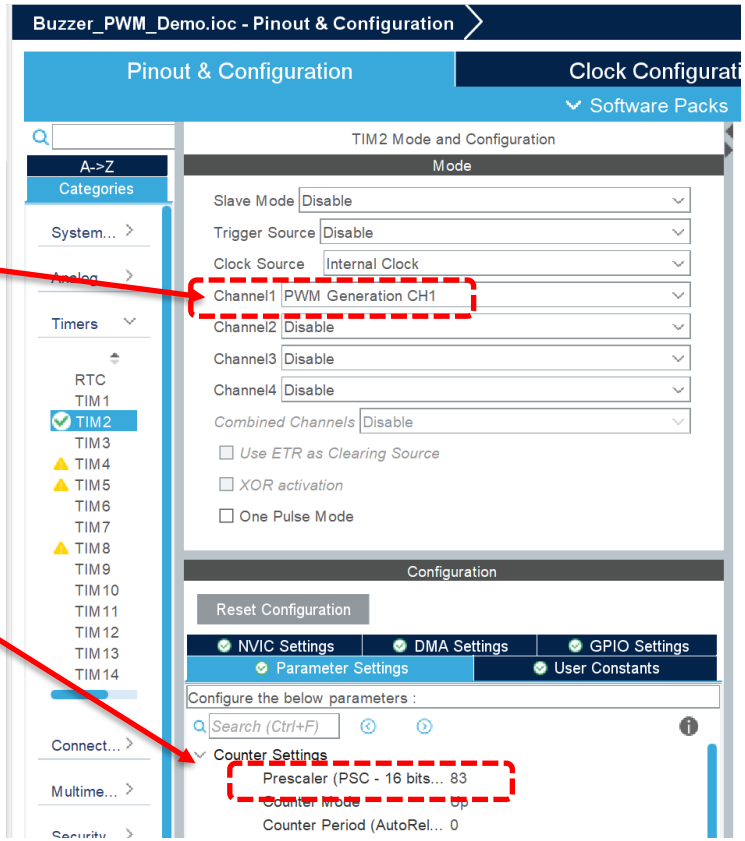
Več informacij : BeriMe.txt





melody.h

```
//##############**"Crazy Frog" song of Crazy frog album**##############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4, NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4, NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0, NOTE_C4, NOTE_A3, NOTE_E4
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```



Nota:

```
ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
setPWM(htim2, TIM_CHANNEL_1, ARR_period, ARR_period/2);

Delaymsecs = noteDurations[melodyIndex][noteIndex] * melodySlowfactor[melodyIndex];

snprintf (SendBuffer,BUFSIZE,"Melody[%d],Note #%d F=%d Hz Duration:%d ms| ARR=%d CCR
CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

HAL_Delay(Delaymsecs);
```

STM32F4

https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/Buzzer_PWM_Demo

STM32F4 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{

melodyCount = sizeof(melodySizes)/ sizeof(uint32_t);

for(melodyIndex = 0; melodyIndex < melodyCount; melodyIndex++)
{
        for(noteIndex = 0; noteIndex < melodySizes[melodyIndex]; noteIndex++)
          {
              //    buzzerSetNewFrequency(melody[melodyIndex][noteIndex]);
                NoteFreq = melody[melodyIndex][noteIndex];
                if (NoteFreq == 0) NoteFreq = 1;

                ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
                setPWM(htim2, TIM_CHANNEL_1, ARR_period, ARR_period/2);

                Delaymsecs = noteDurations[melodyIndex][noteIndex] * melodySlowfactor[melodyIndex];

                HAL_Delay(Delaymsecs);
          }
        snprintf (SendBuffer,BUFSIZE,"\r\n\r\nEnd of Melody[%d]\r\n\r\n",melodyIndex);
            CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

}
```

Melody.h:

```
//###############**"Crazy Frog" song of Crazy frog album**###############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4, NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4, NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0, NOTE_C4, NOTE_A3, NOTE_E4,
NOTE_D4,
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```

Melody.h:

```
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

const uint32_t melodySizes[] ={sizeof(marioMelody)/sizeof(uint32_t),
sizeof(secondDuration)/sizeof(uint32_t),
sizeof(Titanic_duration)/sizeof(uint32_t),
sizeof(Pirates_durations)/sizeof(uint32_t),
sizeof(CrazyFrog_durations)/sizeof(uint32_t)};
```

https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/Buzzer_PWM_Demo

STM32F4, H7 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{

melodyCount = sizeof(melodySizes)/ sizeof(uint32_t);

for(melodyIndex = 0; melodyIndex < melodyCount; melodyIndex++)
{
        for(noteIndex = 0; noteIndex < melodySizes[melodyIndex]; noteIndex++)
          {
              //    buzzerSetNewFrequency(melody[melodyIndex][noteIndex]);
                NoteFreq = melody[melodyIndex][noteIndex];
                if (NoteFreq == 0) NoteFreq = 1;

                ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
                setPWM(htim2, TIM_CHANNEL_1, ARR_period, ARR_period/2);

                Delaymsecs = noteDurations[melodyIndex][noteIndex] * melodySlowfactor[melodyIndex];

                HAL_Delay(Delaymsecs);
          }
        snprintf (SendBuffer,BUFSIZE,"\r\n\r\nEnd of Melody[%d]\r\n\r\n",melodyIndex);
            CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

}
…
```

Melody.h:

```
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

const uint32_t melodySizes[] ={sizeof(marioMelody)/sizeof(uint32_t),
sizeof(secondDuration)/sizeof(uint32_t),
sizeof(Titanic_duration)/sizeof(uint32_t),
sizeof(Pirates_durations)/sizeof(uint32_t),
sizeof(CrazyFrog_durations)/sizeof(uint32_t)};
```

Melody.h:

```
// Zapisi not v [Hz]
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440


// Zapisi melodij v notah [Hz] in trajanju
//###############**"Crazy Frog" song of Crazy frog
album**###############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4,
NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4,
NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0,
NOTE_C4, NOTE_A3, NOTE_E4, NOTE_D4,
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```

VIN projekt - VP5: STM32-CubeIDE projekt, breadboard vezave

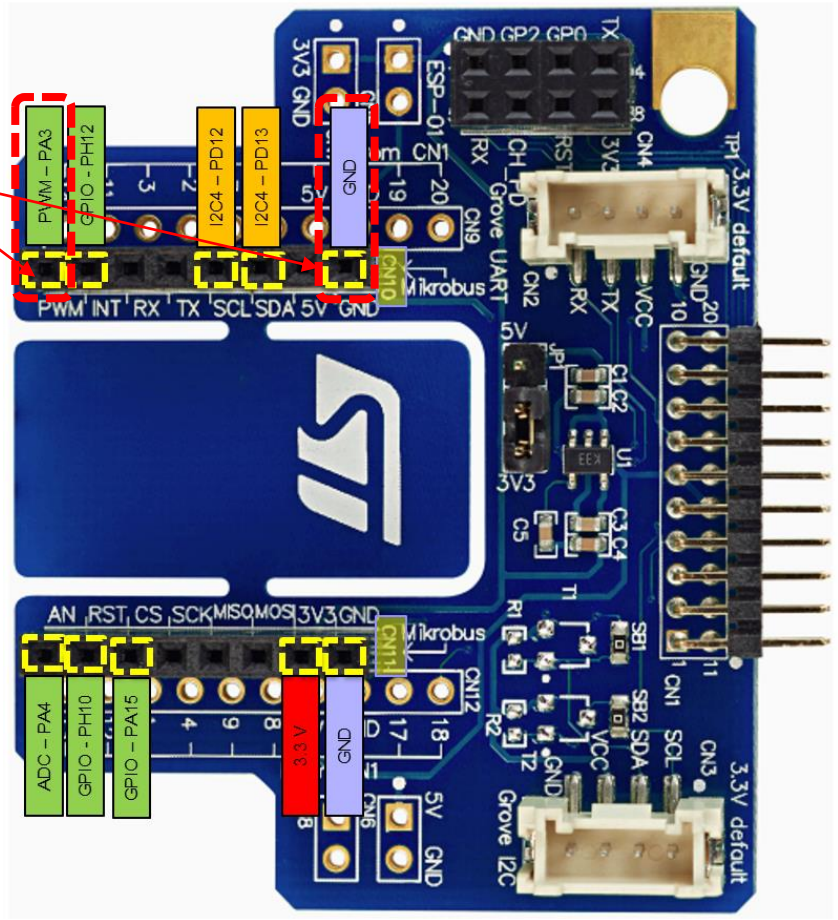
- Osvežitev: STM32 sistema
- Priprava na povezovanje
- STM32 CubeIDE + Breadboard
  - LED, tipka, potenciometer, uporovna tipala
    - STM32F4
    - STM32H7
  - PWM brenčač z melodijami
    - STM32F4
    - STM32H7

Breadboard vezava

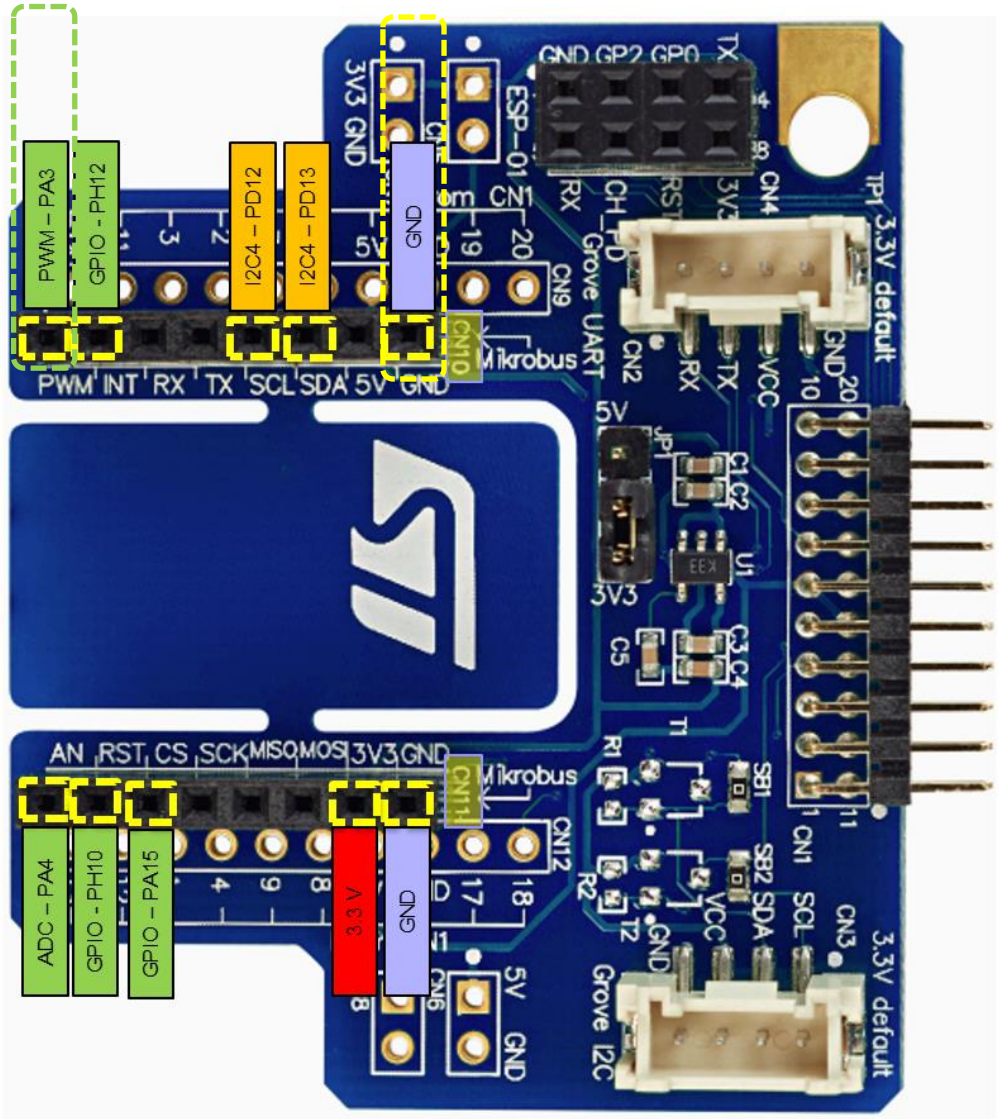
Priključitev na STM32 : 1x analogni, 1x digitalni vhod, 1x digitalni izhod, 4x vgrajene LED diode

Povezava brenčača - STM32H7:

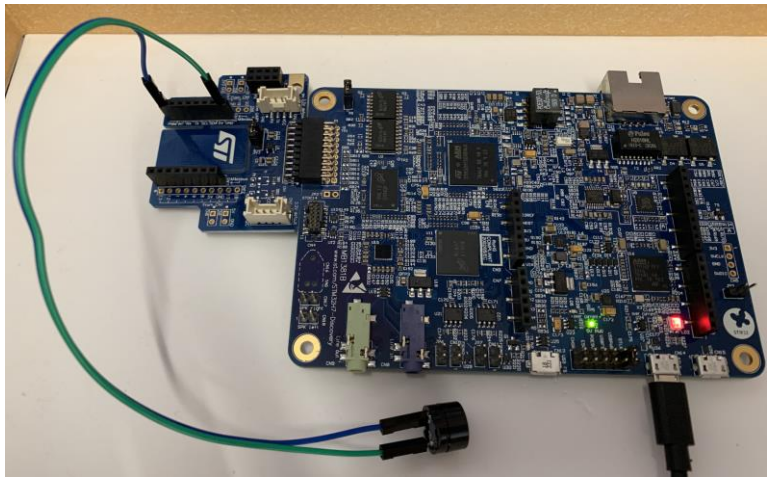
GPIO	Vrsta	Povezava
PA3	Brenčač	+
GND	Brenčač	-



STM32H7 – PWM signali/melodija za brenčača (Buzzer)



Pravilna priključitev



Nepravilna priključitev

https://www.st.com/en/evaluation-tools/stm32h750b-dk.html

STM32H7 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

Brencac se prikljuci na **PA3-PWM na STMod+ Clickboard (TIM2->CH4) in GND (priključitev lahko naredimo skupaj na naslednji vaji – VP5)**

CubeMX :

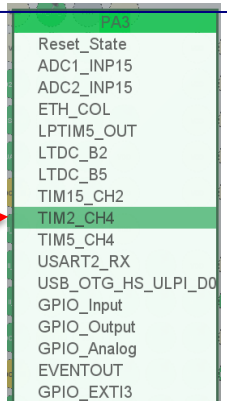
1. Osnovna nastavitev plošče

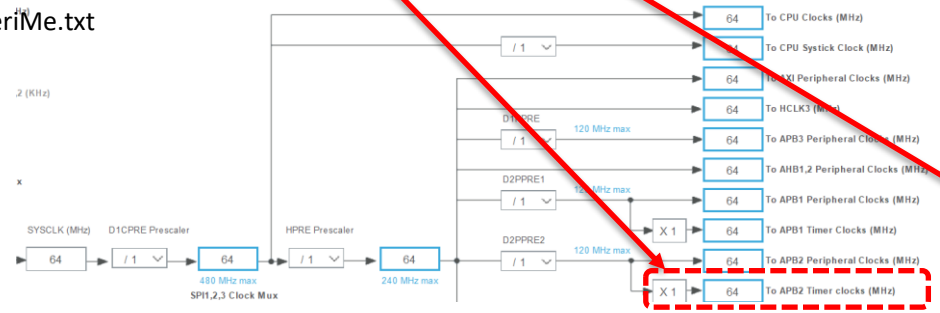
2. Spremeniti pin PA3 v TIM2->CH4
tim2 kanal 4 spremeniti na PWM Generation CH4

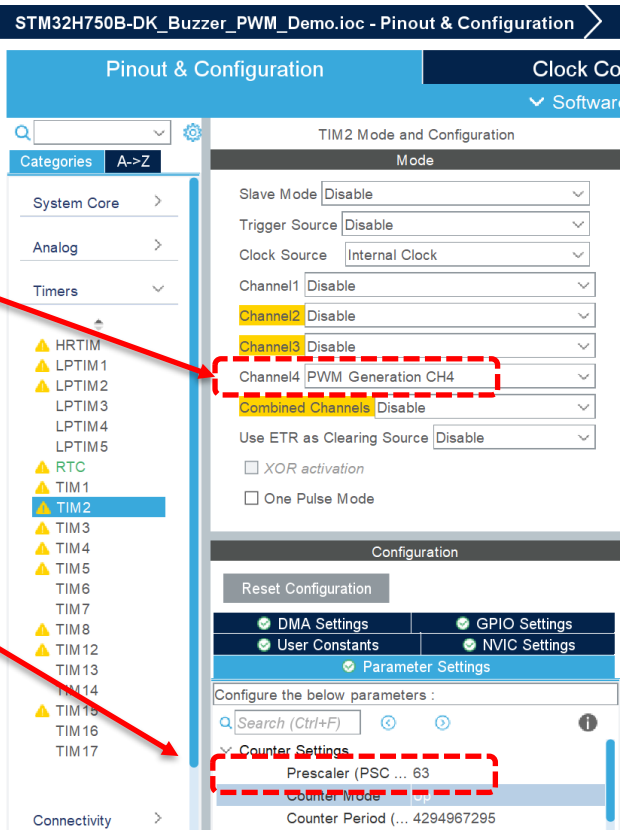
3. Clock & TIM2:
Ura števca = 1 MHz
Prescaler (PSC - 16 bits value) = 64-1 = 63 (clock 1Mhz)

Perioda štetja se bo določala glede na noto (duty cycle je vedno 50%)
Counter Period (AutoReload Register - 16 bits value) =
ARR = 1000000 (ura števca) / Frekv.note[Hz]
CCR1 bo vedno ARR/2 (50% duty cycle)

Več informacij BeriMe.txt







melody.h

```
//##############**"Crazy Frog" song of Crazy frog album**##############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4, NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4, NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0, NOTE_C4, NOTE_A3, NOTE_E4
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```

Nota:

```
ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
    setPWM(htim2, TIM_CHANNEL_4, ARR_period, ARR_period/2);

Delaymsecs = noteDurations[melodyIndex][noteIndex] * melodySlowfactor[melodyIndex];
```

STM32H7

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_Buzzer_PWM_Demo

STM32H7 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{

melodyCount = sizeof(melodySizes)/ sizeof(uint32_t);

for(melodyIndex = 0; melodyIndex < melodyCount; melodyIndex++)
{
        for(noteIndex = 0; noteIndex < melodySizes[melodyIndex]; noteIndex++)
          {
              //    buzzerSetNewFrequency(melody[melodyIndex][noteIndex]);
                NoteFreq = melody[melodyIndex][noteIndex];
                if (NoteFreq == 0) NoteFreq = 1;

                ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
                setPWM(htim2, TIM_CHANNEL_4, ARR_period, ARR_period/2);

                Delaymsecs = noteDurations[melodyIndex][noteIndex] *
                  melodySlowfactor[melodyIndex];

                HAL_Delay(Delaymsecs);
                snprintf (SendBuffer,BUFSIZE,"Melody[%d],Note #%d F=%d Hz Duration:%d ms|
                  CCR1=%d\r\n",melodyIndex,noteIndex,melody[melodyIndex][noteIndex],Delay
                  m2.Instance->ARR,htim2.Instance->CCR1);
                HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);

          }

}
```

Melody.h:

```
//###############**"Crazy Frog" song of Crazy frog album**###############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4, NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4, NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0, NOTE_C4, NOTE_A3, NOTE_E4,
NOTE_D4,
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```

COM4 - PuTTY

```
Melody[0],Note #37 F=2794 Hz Duration:180 ms| ARR=357 CCR1=0
Melody[0],Note #38 F=3136 Hz Duration:180 ms| ARR=318 CCR1=0
Melody[0],Note #39 F=0 Hz Duration:180 ms| ARR=16960 CCR1=0
Melody[0],Note #40 F=2637 Hz Duration:180 ms| ARR=379 CCR1=0
Melody[0],Note #41 F=0 Hz Duration:180 ms| ARR=16960 CCR1=0
Melody[0],Note #42 F=2093 Hz Duration:180 ms| ARR=477 CCR1=0
Melody[0],Note #43 F=2349 Hz Duration:180 ms| ARR=425 CCR1=0
Melody[0],Note #44 F=1976 Hz Duration:180 ms| ARR=506 CCR1=0
Melody[0],Note #45 F=0 Hz Duration:180 ms| ARR=16960 CCR1=0
Melody[0],Note #46 F=0 Hz Duration:180 ms| ARR=16960 CCR1=0
Melody[0],Note #47 F=2093 Hz Duration:180 ms| ARR=477 CCR1=0
Melody[0],Note #48 F=0 Hz Duration:180 ms| ARR=16960 CCR1=0
```

Melody.h:

```
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

const uint32_t melodySizes[] ={sizeof(marioMelody)/sizeof(uint32_t),
sizeof(secondDuration)/sizeof(uint32_t),
sizeof(Titanic_duration)/sizeof(uint32_t),
sizeof(Pirates_durations)/sizeof(uint32_t),
sizeof(CrazyFrog_durations)/sizeof(uint32_t)};
```

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_Buzzer_PWM_Demo

STM32F4, H7 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{

melodyCount = sizeof(melodySizes)/ sizeof(uint32_t);

for(melodyIndex = 0; melodyIndex < melodyCount; melodyIndex++)
{
        for(noteIndex = 0; noteIndex < melodySizes[melodyIndex]; noteIndex++)
          {
              //    buzzerSetNewFrequency(melody[melodyIndex][noteIndex]);
                NoteFreq = melody[melodyIndex][noteIndex];
                if (NoteFreq == 0) NoteFreq = 1;

                ARR_period = (int)(1000000/NoteFreq);  //Already prescaled to 1 MHz
                setPWM(htim2, TIM_CHANNEL_1, ARR_period, ARR_period/2);

                Delaymsecs = noteDurations[melodyIndex][noteIndex] * melodySlowfactor[melodyIndex];

                HAL_Delay(Delaymsecs);
          }
        snprintf (SendBuffer,BUFSIZE,"\r\n\r\nEnd of Melody[%d]\r\n\r\n",melodyIndex);
            CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

}
…
```

Melody.h:

```
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

const uint32_t melodySizes[] ={sizeof(marioMelody)/sizeof(uint32_t),
sizeof(secondDuration)/sizeof(uint32_t),
sizeof(Titanic_duration)/sizeof(uint32_t),
sizeof(Pirates_durations)/sizeof(uint32_t),
sizeof(CrazyFrog_durations)/sizeof(uint32_t)};
```

Melody.h:

```
// Zapisi not v [Hz]
#define NOTE_C4  262
#define NOTE_CS4 277
#define NOTE_D4  294
#define NOTE_DS4 311
#define NOTE_E4  330
#define NOTE_F4  349
#define NOTE_FS4 370
#define NOTE_G4  392
#define NOTE_GS4 415
#define NOTE_A4  440


// Zapisi melodij v notah [Hz] in trajanju
//###############**"Crazy Frog" song of Crazy frog
album**###############//
const uint32_t   CrazyFrog_notes[] = {
  NOTE_D4, 0, NOTE_F4, NOTE_D4, 0, NOTE_D4, NOTE_G4,
NOTE_D4, NOTE_C4,
  NOTE_D4, 0, NOTE_A4, NOTE_D4, 0, NOTE_D4, NOTE_AS4,
NOTE_A4, NOTE_F4,
  NOTE_D4, NOTE_A4, NOTE_D5, NOTE_D4, NOTE_C4, 0,
NOTE_C4, NOTE_A3, NOTE_E4, NOTE_D4,
  0,NOTE_D4,NOTE_D4
};

const uint32_t   CrazyFrog_durations[] = {
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 6, 16, 16, 16, 8, 8, 8,
  8, 8, 8, 16, 16, 16, 16, 8, 8, 2,
  8,4,4
};
//############End of Crazy Frog##############//
```