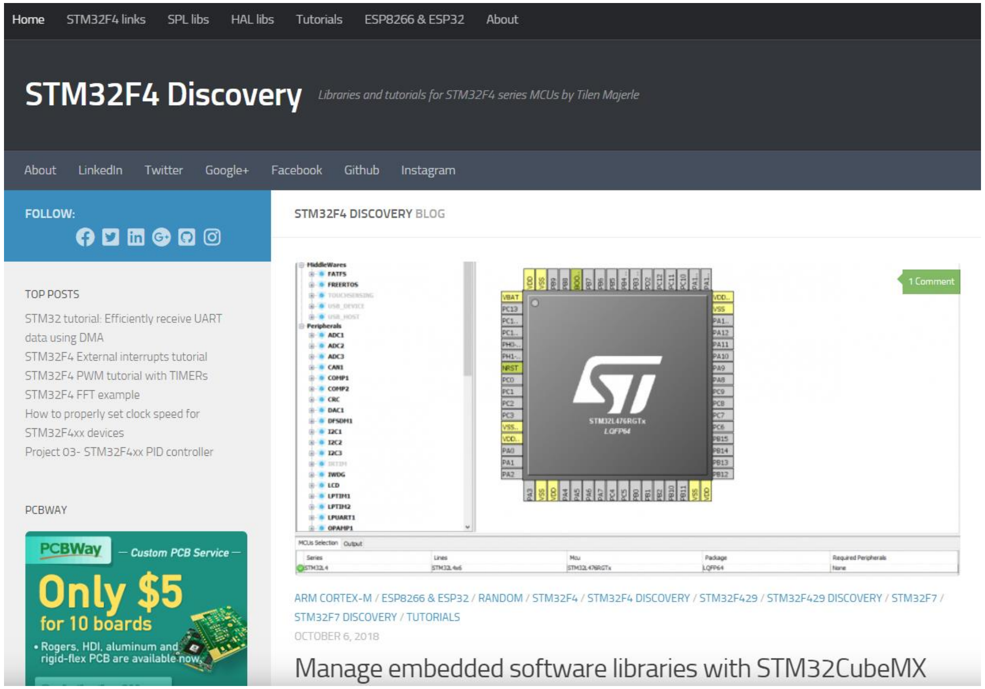
Vhodno izhodne naprave

Laboratorijska vaja 4 - VP 4
VIN projekt, „Edge AI“, Miško3,
STM32 F4/H7 PWM-I2C primeri

VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

Delo na STM32F4 razvojnem sistemu - zgodba



STM32F4 Discovery *Libraries and tutorials for STM32F4 series MCUs by Tilen Majerle*

TOP POSTS

- STM32 tutorial: Efficiently receive UART data using DMA
- STM32F4 External interrupts tutorial
- STM32F4 PWM tutorial with TIMERs
- STM32F4 FFT example
- How to properly set clock speed for STM32F4xx devices
- Project 03- STM32F4xx PID controller

PCBWAY

STM32F4 DISCOVERY BLOG

Series	Lines	Mcu	Package	Required Peripherals
STM32L4	STM32L4x6	STM32L476RGTx	LQFP64	None

ARM CORTEX-M / ESP8266 & ESP32 / RANDOM / STM32F4 / STM32F4 DISCOVERY / STM32F429 / STM32F429 DISCOVERY / STM32F7 / STM32F7 DISCOVERY / TUTORIALS

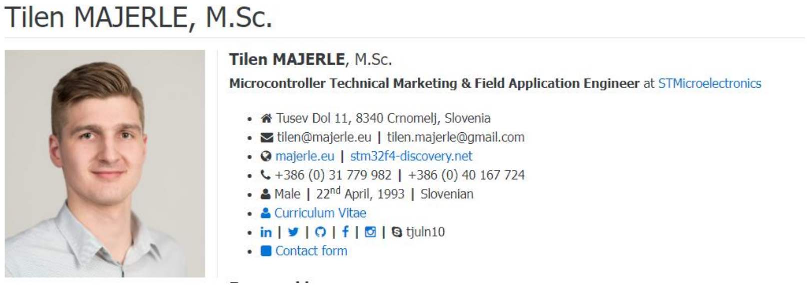
OCTOBER 6, 2018

Manage embedded software libraries with STM32CubeMX

majerle.eu

TILEN MAJERLE

Knowledge sharing is caring

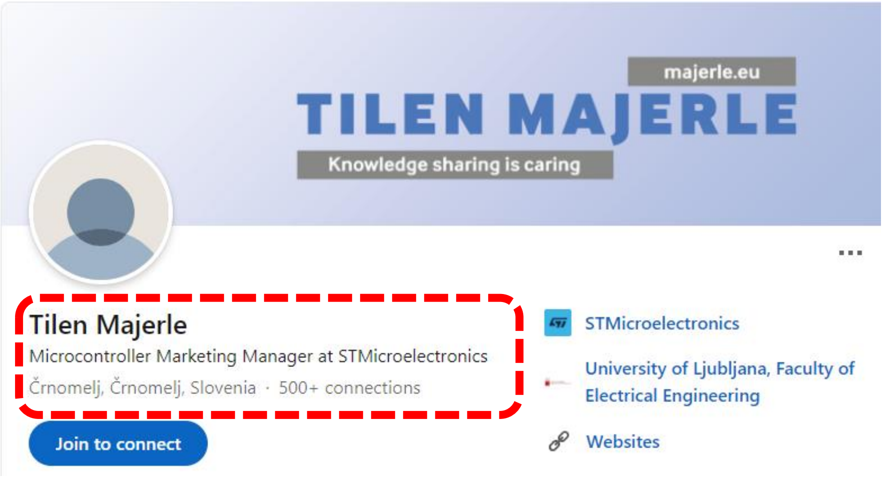


Tilen MAJERLE, M.Sc.

Tilen MAJERLE, M.Sc.

Microcontroller Technical Marketing & Field Application Engineer at STMicroelectronics

- Tusev Dol 11, 8340 Crnomelj, Slovenia
- tilen@majerle.eu | tilen.majerle@gmail.com
- majerle.eu | stm32f4-discovery.net
- +386 (0) 31 779 982 | +386 (0) 40 167 724
- Male | 22nd April, 1993 | Slovenian
- Curriculum Vitae
- in | | | f | | tjuln10
- Contact form


majerle.eu
TILEN MAJERLE
Knowledge sharing is caring
...

Tilen Majerle

Microcontroller Marketing Manager at STMicroelectronics

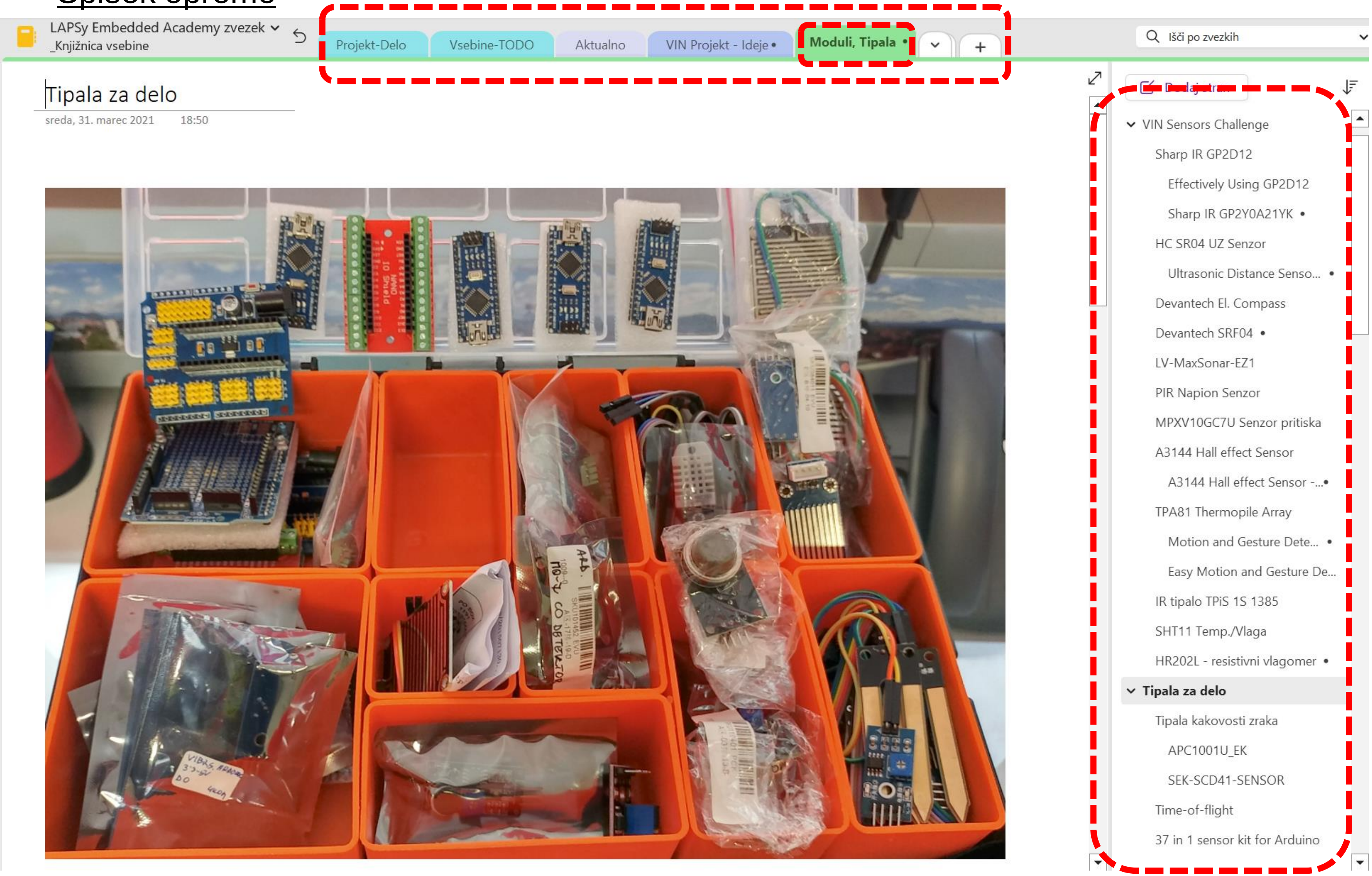
Črnomelj, Črnomelj, Slovenia · 500+ connections

Join to connect
STMicroelectronics
University of Ljubljana, Faculty of Electrical Engineering
Websites

https://stm32f4-discovery.net/

VIN projekt

Spisek opreme



LAPSy Embedded Academy zvezek
_Knjižnica vsebine

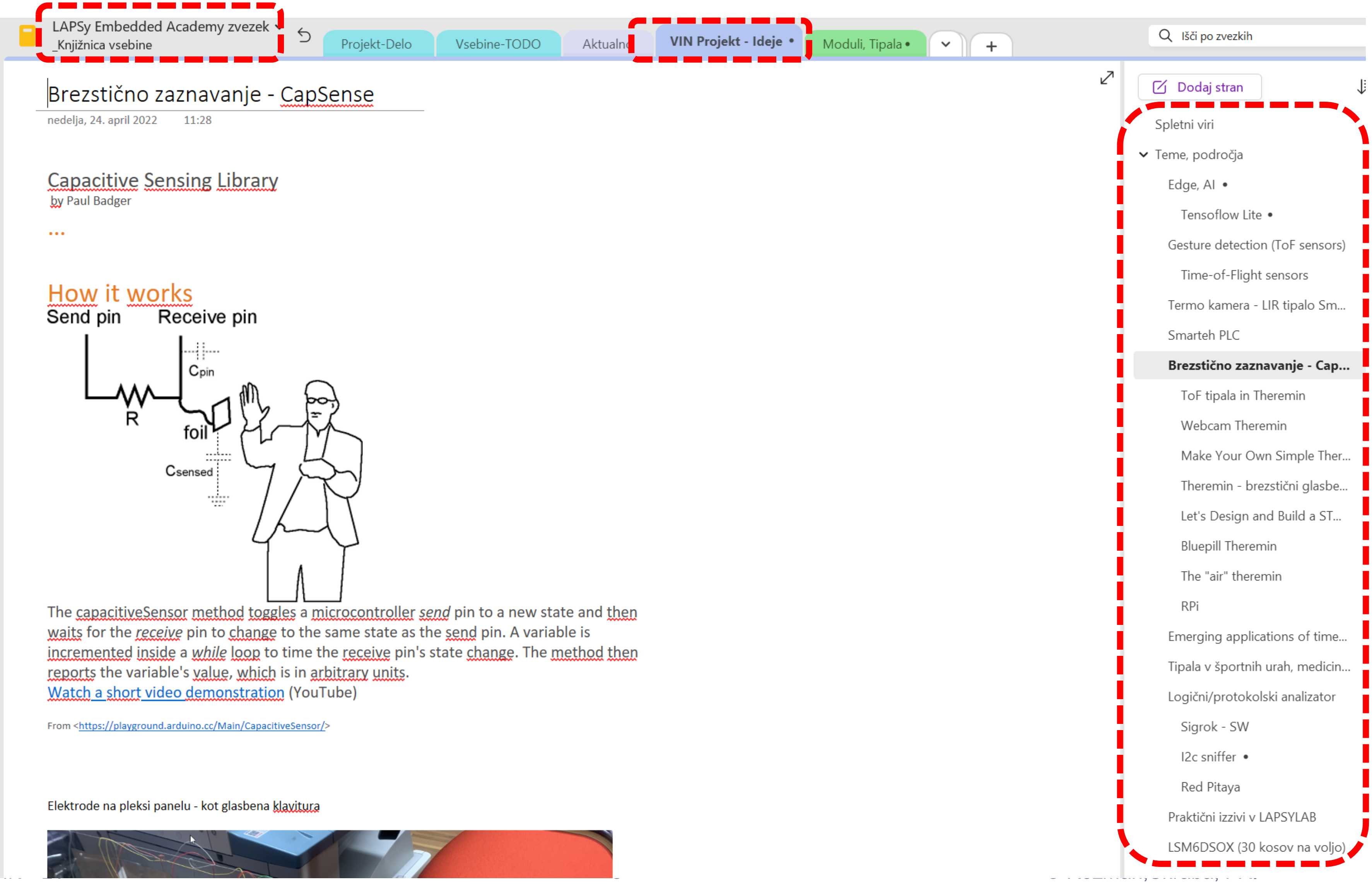
Tipala za delo

sreda, 31. marec 2021 18:50

- VIN Sensors Challenge
  - Sharp IR GP2D12
    - Effectively Using GP2D12
    - Sharp IR GP2Y0A21YK •
  - HC SR04 UZ Senzor
    - Ultrasonic Distance Senso... •
  - Devantech El. Compass
  - Devantech SRF04 •
  - LV-MaxSonar-EZ1
  - PIR Napion Senzor
  - MPXV10GC7U Senzor pritiska
  - A3144 Hall effect Sensor
    - A3144 Hall effect Sensor -...•
  - TPA81 Thermopile Array
    - Motion and Gesture Dete... •
    - Easy Motion and Gesture De...
  - IR tipalo TPiS 1S 1385
  - SHT11 Temp./Vlaga
  - HR202L - resistivni vlagomer •
- **Tipala za delo**
  - Tipala kakovosti zraka
    - APC1001U_EK
    - SEK-SCD41-SENSOR
  - Time-of-flight
  - 37 in 1 sensor kit for Arduino

VIN projekt

Ideje



LAPSy Embedded Academy zvezek
_Knjižnica vsebine

Brezstično zaznavanje - CapSense

nedelja, 24. april 2022 11:28

Capacitive Sensing Library
by Paul Badger

...

How it works

The capacitiveSensor method toggles a microcontroller *send* pin to a new state and then waits for the *receive* pin to change to the same state as the send pin. A variable is incremented inside a *while* loop to time the receive pin's state change. The method then reports the variable's value, which is in arbitrary units.
Watch a short video demonstration (YouTube)

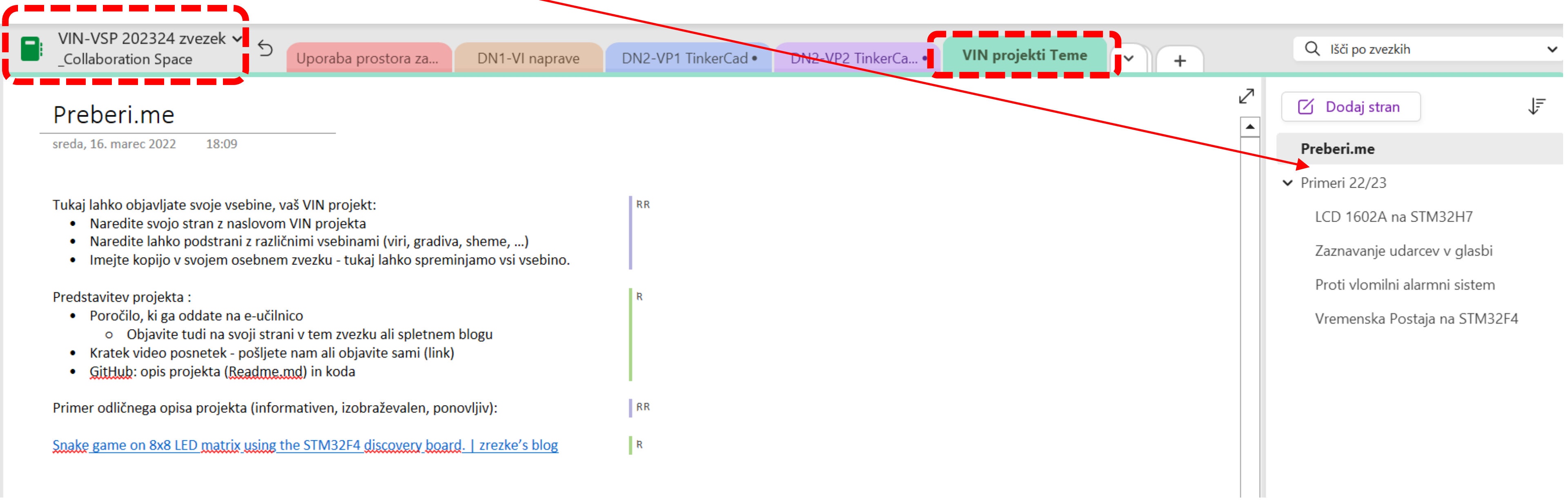
From <https://playground.arduino.cc/Main/CapacitiveSensor/>

Elektrode na pleksi panelu - kot glasbena klavitura

- Spletni viri
- Teme, področja
  - Edge, AI •
    - Tensoflow Lite •
  - Gesture detection (ToF sensors)
    - Time-of-Flight sensors
  - Termo kamera - LIR tipalo Sm...
  - Smarteh PLC
  - Brezstično zaznavanje - Cap...
    - ToF tipala in Theremin
    - Webcam Theremin
    - Make Your Own Simple Ther...
    - Theremin - brezstični glasbe...
    - Let's Design and Build a ST...
    - Bluepill Theremin
    - The "air" theremin
    - RPi
  - Emerging applications of time...
  - Tipala v športnih urah, medicin...
  - Logični/protokolski analizator
    - Sigrok - SW
    - I2c sniffer •
    - Red Pitaya
  - Praktični izzivi v LAPSYLAB
  - LSM6DSOX (30 kosov na voljo)

VIN projekt

Vaša tema ?



Preberi.me

sreda, 16. marec 2022 18:09

Tukaj lahko objavljate svoje vsebine, vaš VIN projekt:

- Naredite svojo stran z naslovom VIN projekta
- Naredite lahko podstrani z različnimi vsebinami (viri, gradiva, sheme, ...)
- Imejte kopijo v svojem osebnem zvezku - tukaj lahko spreminjamo vsi vsebino.

Predstavitev projekta :

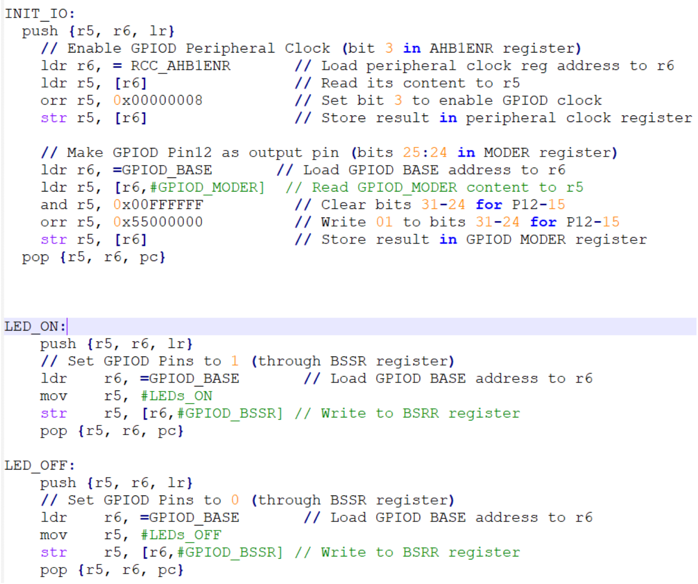
- Poročilo, ki ga oddate na e-učilnico
    - Objavite tudi na svoji strani v tem zvezku ali spletnem blogu
- Kratek video posnetek - pošljete nam ali objavite sami (link)
- GitHub: opis projekta (Readme.md) in koda

Primer odličnega opisa projekta (informativen, izobraževalen, ponovljiv):

Snake game on 8x8 LED matrix using the STM32F4 discovery board. | zrezke's blog

Osnovni projekt CubeIDE – GPIO – nivoji programiranja

Baremetal - zbirnik



```
INIT_IO:
  push {r5, r6, lr}
    // Enable GPIOD Peripheral Clock (bit 3 in AHB1ENR register)
    ldr r6, = RCC_AHB1ENR       // Load peripheral clock reg address to r6
    ldr r5, [r6]                // Read its content to r5
    orr r5, 0x00000008          // Set bit 3 to enable GPIOD clock
    str r5, [r6]                // Store result in peripheral clock register

    // Make GPIOD Pin12 as output pin (bits 25:24 in MODER register)
    ldr r6, =GPIOD_BASE       // Load GPIOD BASE address to r6
    ldr r5, [r6,#GPIOD_MODER]  // Read GPIOD_MODER content to r5
    and r5, 0x00FFFFFF          // Clear bits 31-24 for P12-15
    orr r5, 0x55000000          // Write 01 to bits 31-24 for P12-15
    str r5, [r6]                // Store result in GPIOD MODER register
  pop {r5, r6, pc}


LED_ON:
    push {r5, r6, lr}
    // Set GPIOD Pins to 1 (through BSSR register)
    ldr    r6, =GPIOD_BASE        // Load GPIOD BASE address to r6
    mov    r5, #LEDs_ON
    str    r5, [r6,#GPIOD_BSSR] // Write to BSRR register
    pop {r5, r6, pc}

LED_OFF:
    push {r5, r6, lr}
    // Set GPIOD Pins to 0 (through BSSR register)
    ldr    r6, =GPIOD_BASE        // Load GPIOD BASE address to r6
    mov    r5, #LEDs_OFF
    str    r5, [r6,#GPIOD_BSSR] // Write to BSRR register
    pop {r5, r6, pc}
```

https://github.com/LAPSyLAB/ORLab-STM32/tree/main/GPIO_LEDs

Baremetal - C

```
  /* USER CODE BEGIN 2 */


  RCC->AHB1ENR |= 0x08;
// Enable clock for GPIOD
  GPIOD->MODER |= 0x01000000;        //
MODE Register: bit 12 == out

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    GPIOD->ODR ^= 0x1000;            //
Toggle PD12

/* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
    for (int i=0; i<0x1000000; i++) {};
// waste some time
  }
  /* USER CODE END 3 */
```

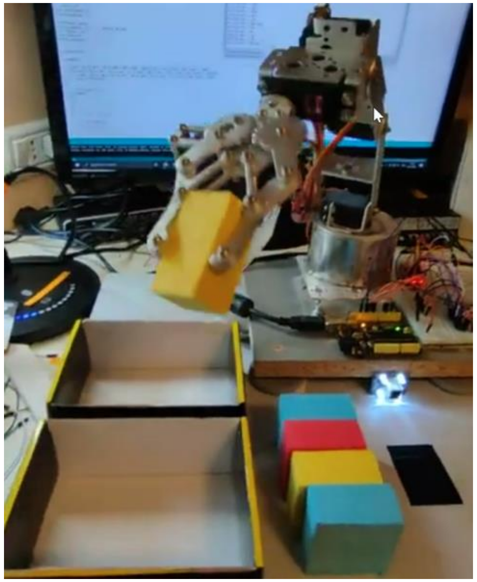
https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/LED_GPIO_C_Baremetal_C

HAL - C

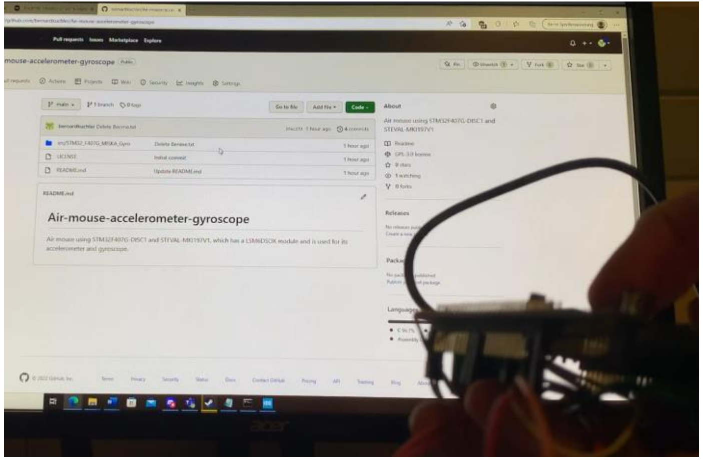
```
/* Infinite loop */
 /* USER CODE BEGIN WHILE */
 while (1)
 {

HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);

   /* USER CODE END WHILE */

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



void HAL_GPIO_TogglePin(GPIO_TypeDef* GPIOx,
uint16_t GPIO_Pin)
{
  uint32_t odr;

  /* Check the parameters */
  assert_param(IS_GPIO_PIN(GPIO_Pin));

  /* get current Ouput Data Register value
*/
  odr = GPIOx->ODR;

  /* Set selected pins that were at low
level, and reset ones that were high */
  GPIOx->BSRR = ((odr & GPIO_Pin) <<
GPIO_NUMBER) | (~odr & GPIO_Pin);
}
```

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

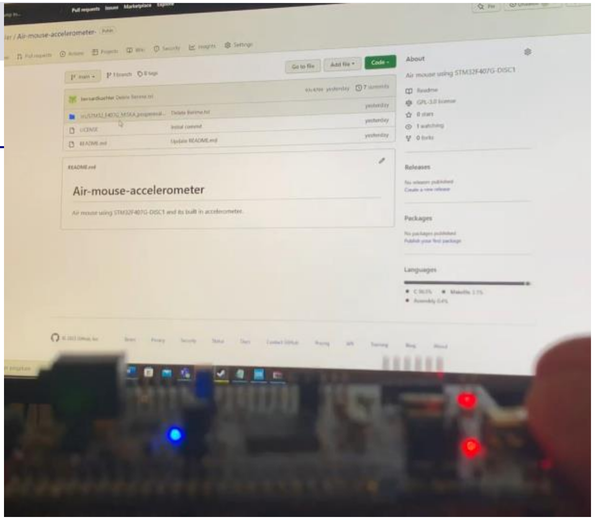
VP – Primeri projektov STM32F4, H7 – 21/22



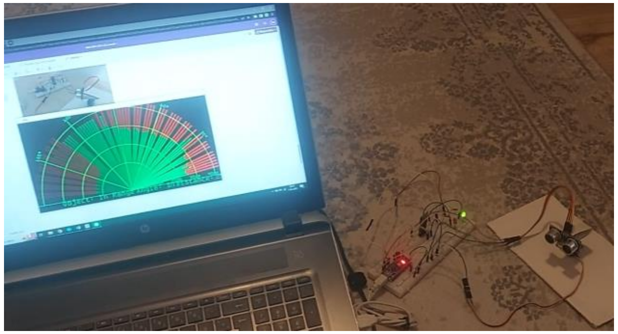
Robot: Colour Box sorter



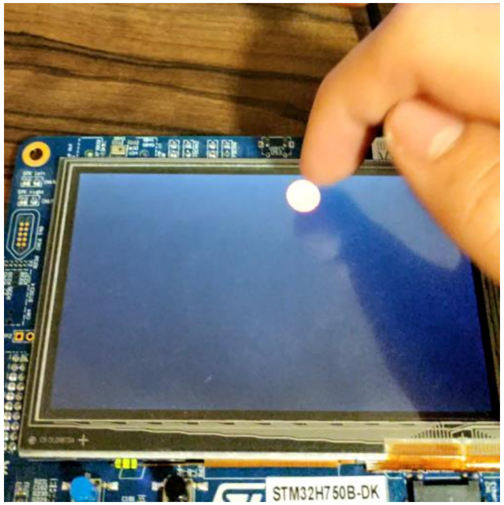
F4: LSM6DSOX – Air Mouse



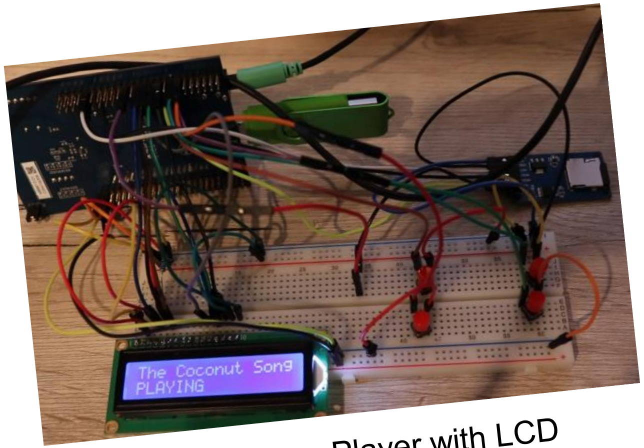
F4: Air Mouse



3D Sonar



H7: Circle Popper

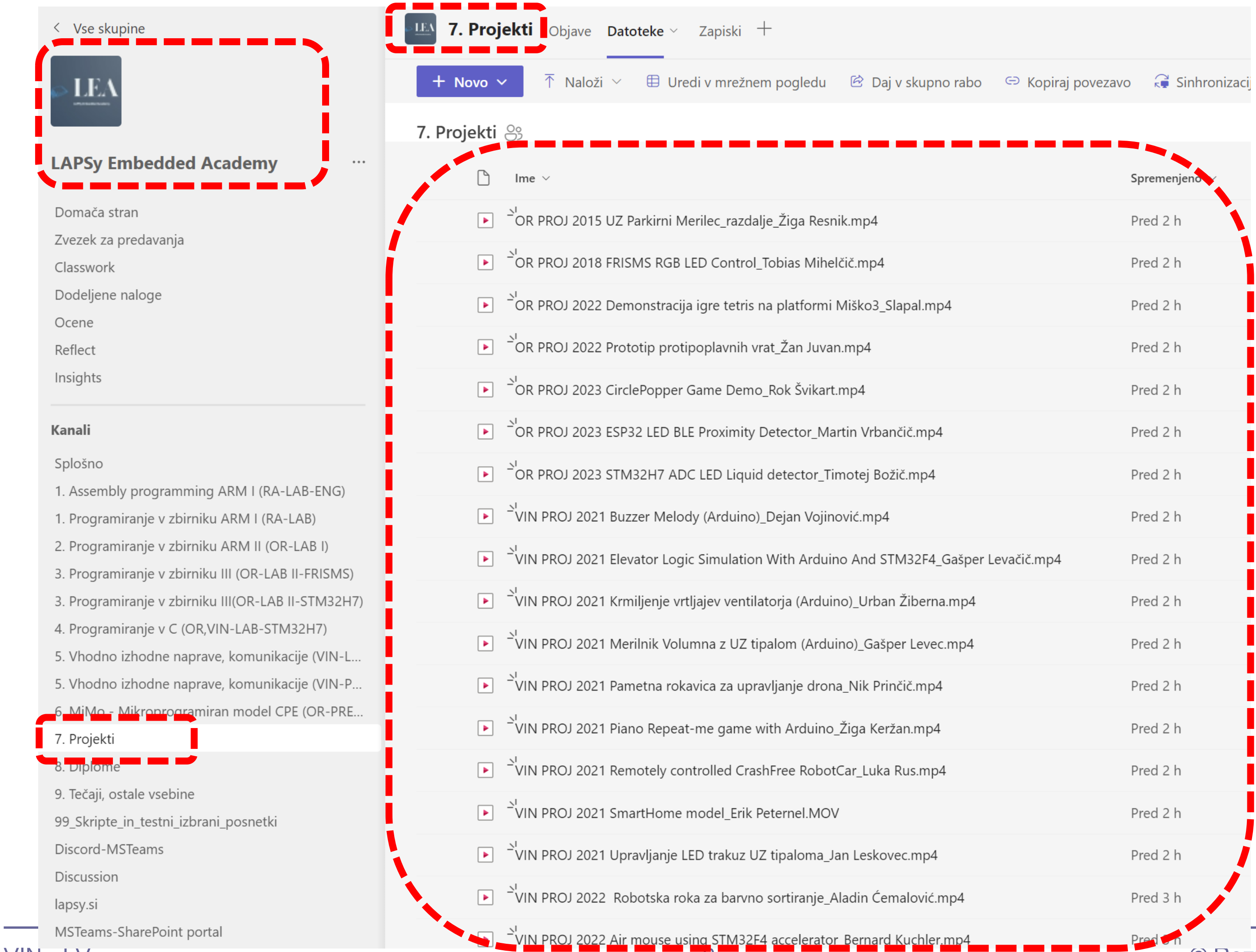


F4: Wave Player with LCD

https://github.com/LAPSyLAB/STM32F4_Docs_and_Examples/

VIN projekt

Video posnetki VIN projektov iz prejšnjih let



< Vse skupine

LAPSy Embedded Academy ...

Domača stran
Zvezek za predavanja
Classwork
Dodeljene naloge
Ocene
Reflect
Insights

Kanali

Splošno
1. Assembly programming ARM I (RA-LAB-ENG)
1. Programiranje v zbirniku ARM I (RA-LAB)
2. Programiranje v zbirniku ARM II (OR-LAB I)
3. Programiranje v zbirniku III (OR-LAB II-FRISMS)
3. Programiranje v zbirniku III(OR-LAB II-STM32H7)
4. Programiranje v C (OR,VIN-LAB-STM32H7)
5. Vhodno izhodne naprave, komunikacije (VIN-L...
5. Vhodno izhodne naprave, komunikacije (VIN-P...
6. MiMo - Mikroprogramiran model CPE (OR-PRE...
7. Projekti
8. Diplome
9. Tečaji, ostale vsebine
99_Skripte_in_testni_izbrani_posnetki
Discord-MSTeams
Discussion
lapsy.si
MSTeams-SharePoint portal

7. Projekti Objave Datoteke Zapiski +

+ Novo | Naloži | Uredi v mrežnem pogledu | Daj v skupno rabo | Kopiraj povezavo | Sinhronizacij

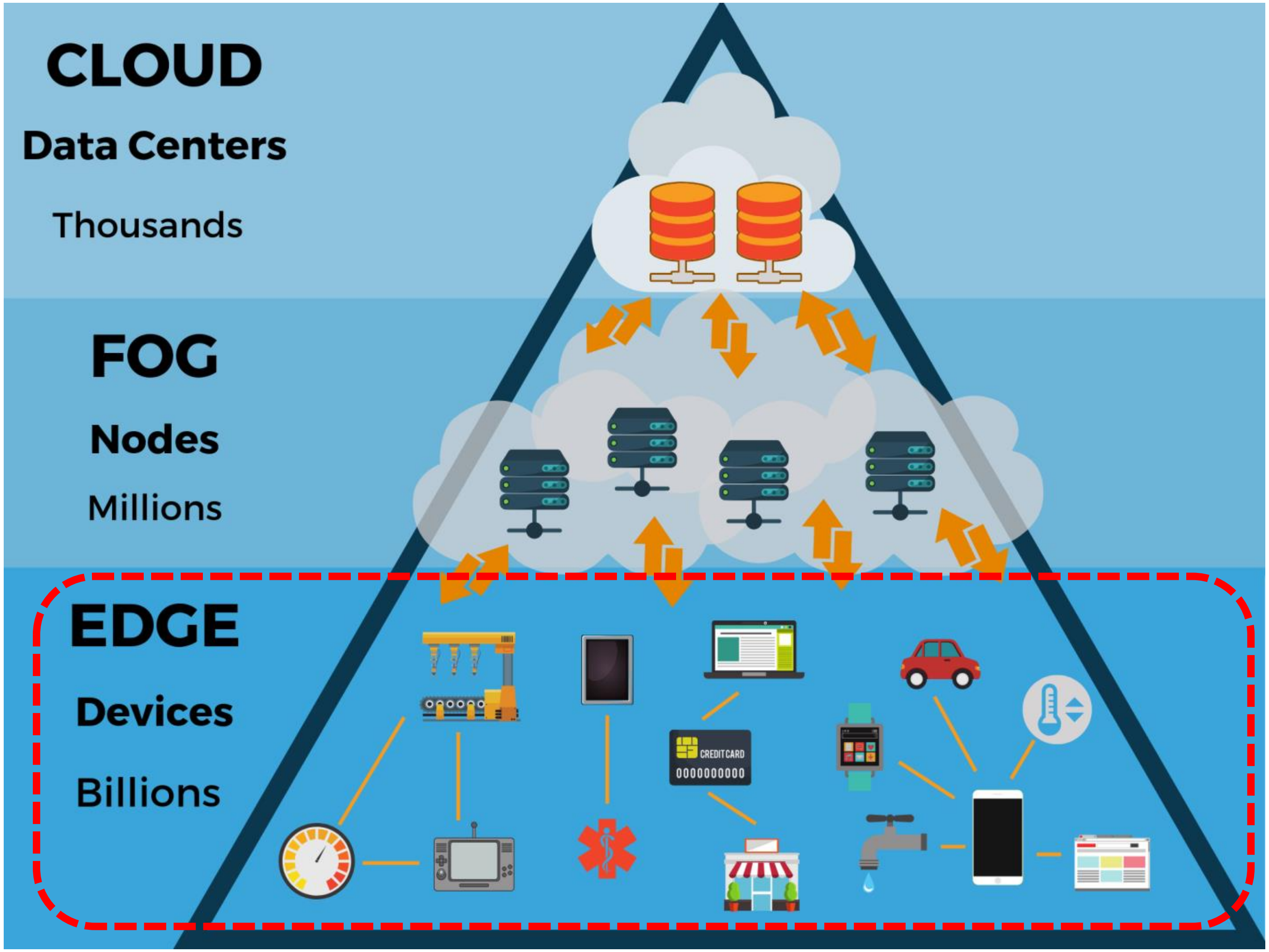
7. Projekti

Ime	Spremenjeno
OR PROJ 2015 UZ Parkirni Merilec_razdalje_Žiga Resnik.mp4	Pred 2 h
OR PROJ 2018 FRISMS RGB LED Control_Tobias Mihelčič.mp4	Pred 2 h
OR PROJ 2022 Demonstracija igre tetris na platformi Miško3_Slapal.mp4	Pred 2 h
OR PROJ 2022 Prototip protipoplavnih vrat_Žan Juvan.mp4	Pred 2 h
OR PROJ 2023 CirclePopper Game Demo_Rok Švikart.mp4	Pred 2 h
OR PROJ 2023 ESP32 LED BLE Proximity Detector_Martin Vrbančič.mp4	Pred 2 h
OR PROJ 2023 STM32H7 ADC LED Liquid detector_Timotej Božič.mp4	Pred 2 h
VIN PROJ 2021 Buzzer Melody (Arduino)_Dejan Vojinović.mp4	Pred 2 h
VIN PROJ 2021 Elevator Logic Simulation With Arduino And STM32F4_Gašper Levačič.mp4	Pred 2 h
VIN PROJ 2021 Krmiljenje vrtljajev ventilatorja (Arduino)_Urban Žiberna.mp4	Pred 2 h
VIN PROJ 2021 Merilnik Volumna z UZ tipalom (Arduino)_Gašper Levec.mp4	Pred 2 h
VIN PROJ 2021 Pametna rokavica za upravljanje drona_Nik Prinčič.mp4	Pred 2 h
VIN PROJ 2021 Piano Repeat-me game with Arduino_Žiga Keržan.mp4	Pred 2 h
VIN PROJ 2021 Remotely controlled CrashFree RobotCar_Luka Rus.mp4	Pred 2 h
VIN PROJ 2021 SmartHome model_Erik Peternel.MOV	Pred 2 h
VIN PROJ 2021 Upravljanje LED trakuz UZ tipaloma_Jan Leskovec.mp4	Pred 2 h
VIN PROJ 2022 Robotska roka za barvno sortiranje_Aladin Ćemalović.mp4	Pred 3 h
VIN PROJ 2022 Air mouse using STM32F4 accelerator_Bernard Kuchler.mp4	Pred 3 h

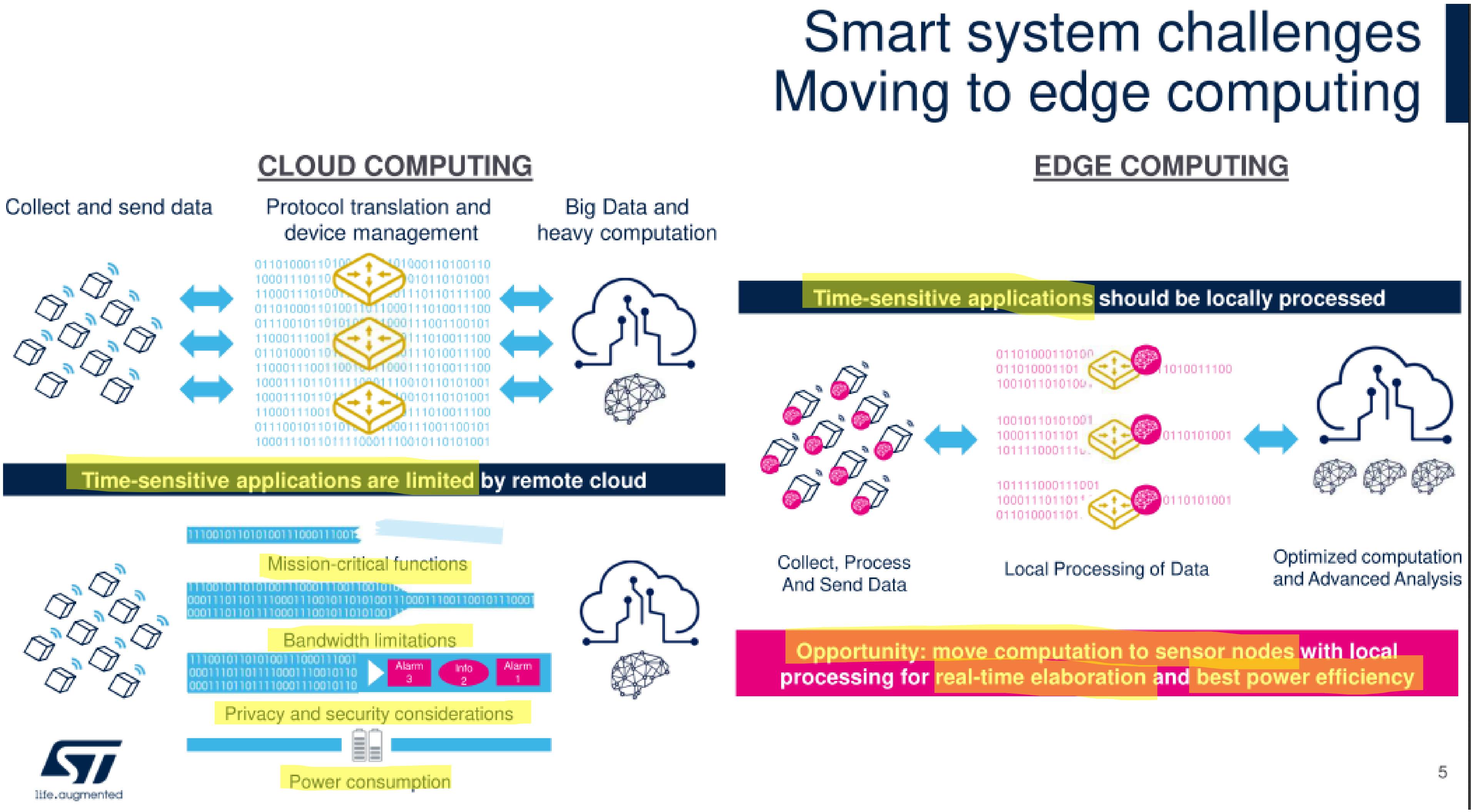
VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

Edge computing



Edge computing



Smart system challenges
Moving to edge computing

CLOUD COMPUTING

Collect and send data

Protocol translation and device management

Big Data and heavy computation

Time-sensitive applications are limited by remote cloud

Mission-critical functions

Bandwidth limitations

Privacy and security considerations

Power consumption

EDGE COMPUTING

Time-sensitive applications should be locally processed

Collect, Process And Send Data

Local Processing of Data

Optimized computation and Advanced Analysis

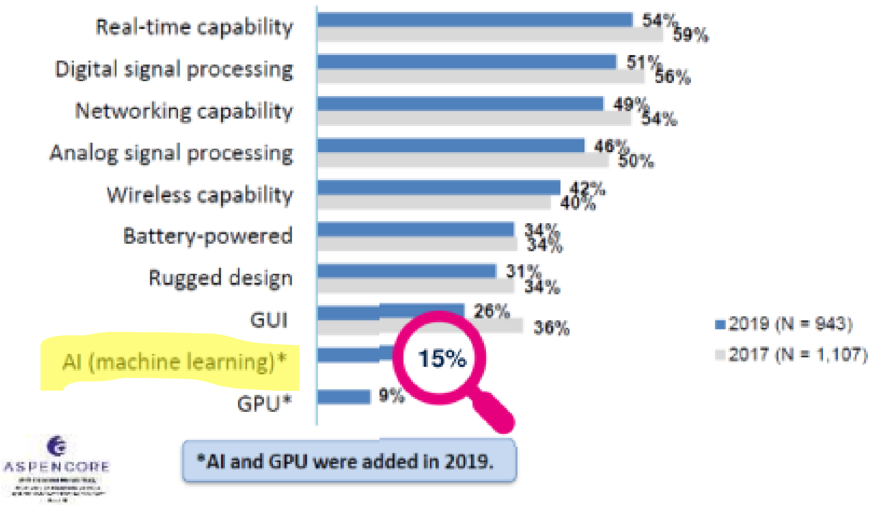
Opportunity: move computation to sensor nodes with local processing for real-time elaboration and best power efficiency

5

Edge computing

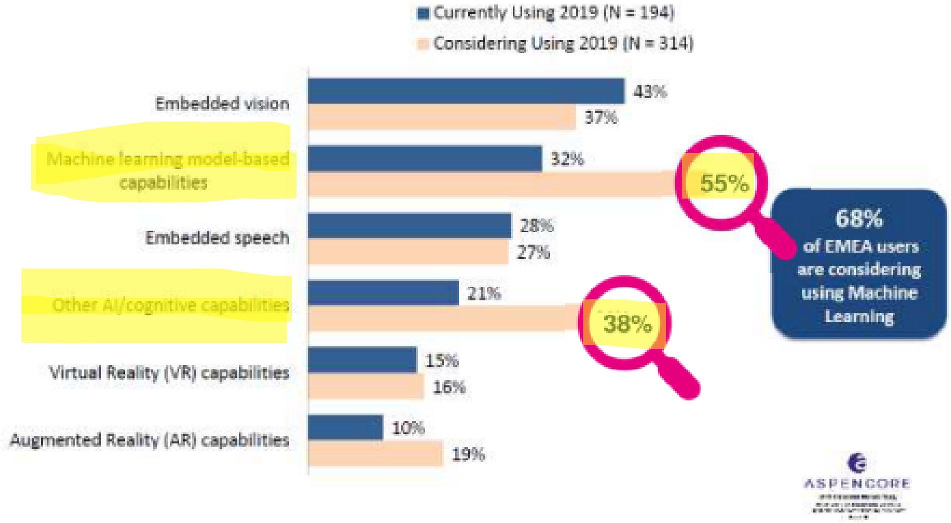
AI is moving to the edge

Capabilities included in a project



*AI and GPU were added in 2019.

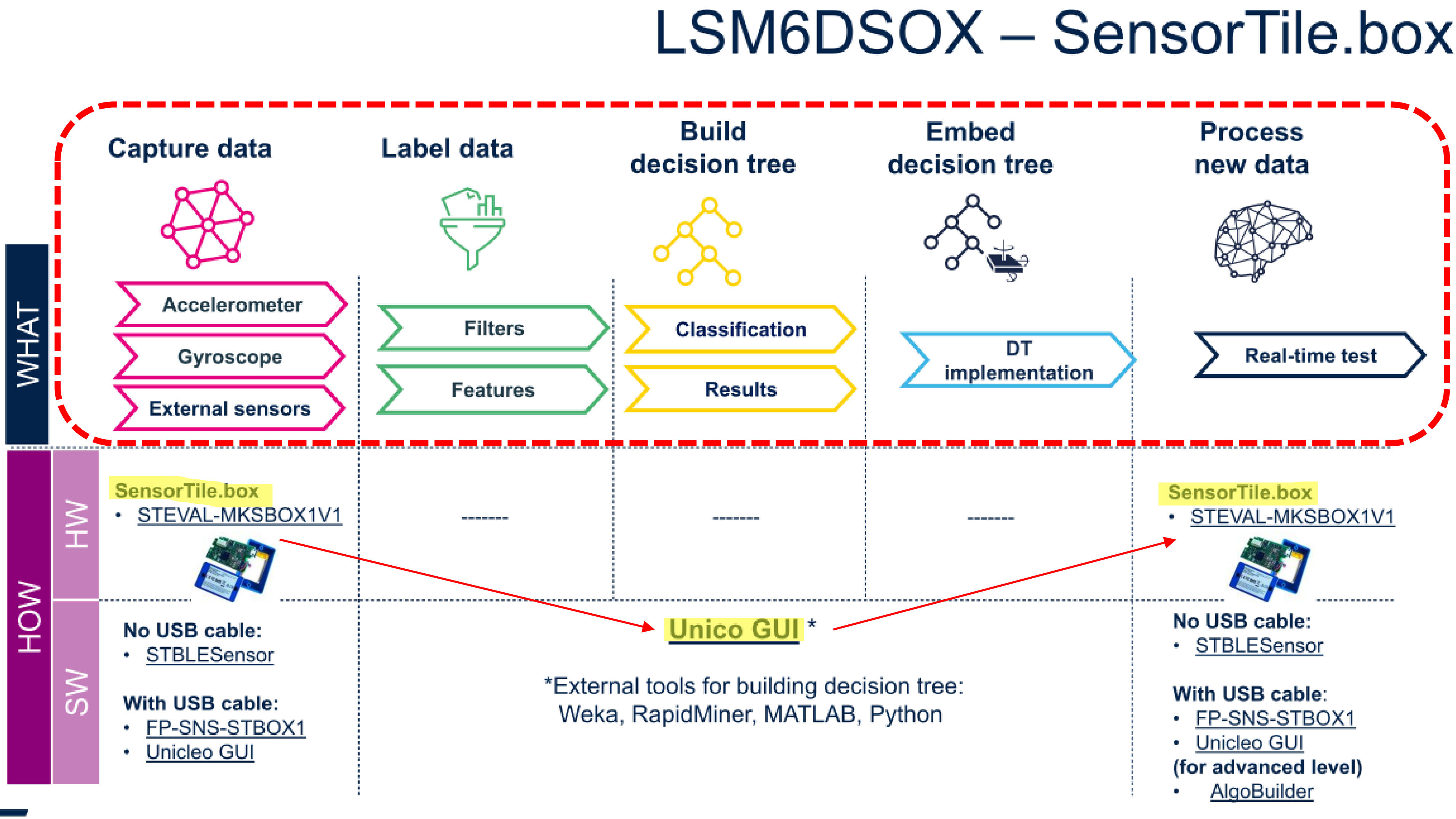
Advanced technology in a project



68% of EMEA users are considering using Machine Learning

- 15% of embedded projects already include AI in 2019
- Pervasion of Machine Learning and other AI capabilities

Edge computing – moduli, tipala

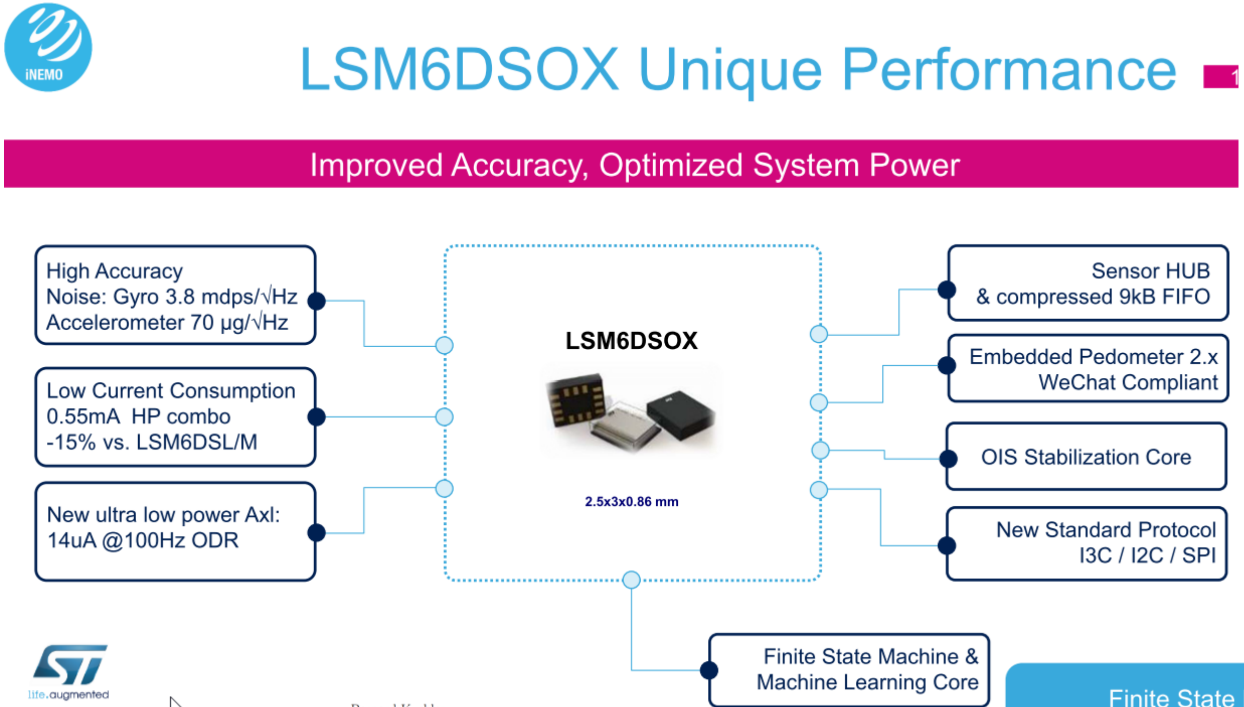


LSM6DSOX – SensorTile.box

		Capture data	Label data	Build decision tree	Embed decision tree	Process new data
WHAT		Accelerometer Gyroscope External sensors	Filters Features	Classification Results	DT implementation	Real-time test
HOW	HW	SensorTile.box • STEVAL-MKSBOX1V1	-------	-------	-------	SensorTile.box • STEVAL-MKSBOX1V1
	SW	**No USB cable:** • STBLESensor **With USB cable:** • FP-SNS-STBOX1 • Unicleo GUI	Unico GUI *			**No USB cable:** • STBLESensor **With USB cable:** • FP-SNS-STBOX1 • Unicleo GUI **(for advanced level)** • AlgoBuilder

*External tools for building decision tree:
Weka, RapidMiner, MATLAB, Python

Edge computing – moduli, tipala



LSM6DSOX Unique Performance

Improved Accuracy, Optimized System Power

High Accuracy
Noise: Gyro 3.8 mdps/√Hz
Accelerometer 70 µg/√Hz

Low Current Consumption
0.55mA HP combo
-15% vs. LSM6DSL/M

New ultra low power Axl:
14uA @100Hz ODR

LSM6DSOX

2.5x3x0.86 mm

Sensor HUB
& compressed 9kB FIFO

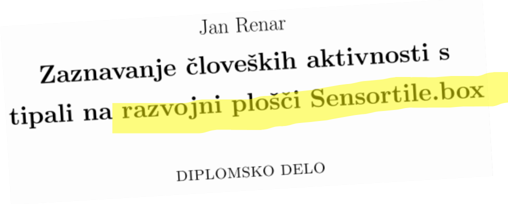
Embedded Pedometer 2.x
WeChat Compliant

OIS Stabilization Core

New Standard Protocol
I3C / I2C / SPI

Finite State Machine &
Machine Learning Core

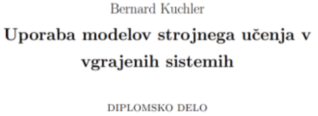
LSM6DSOX adapter board
for a standard DIL24 socket



Jan Renar

Zaznavanje človeških aktivnosti s tipali na razvojni plošči Sensortile.box

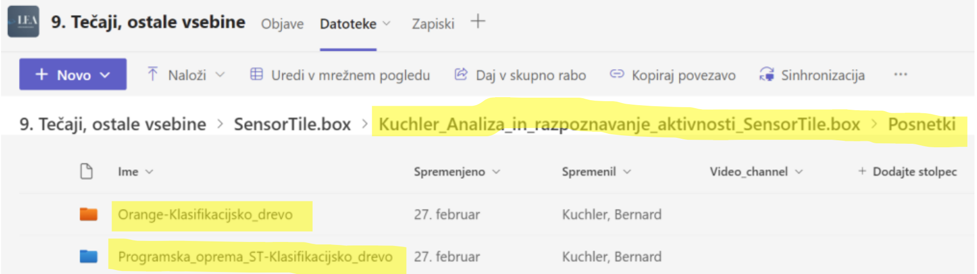
DIPLOMSKO DELO



Bernard Kuchler

Uporaba modelov strojnega učenja v vgrajenih sistemih

DIPLOMSKO DELO

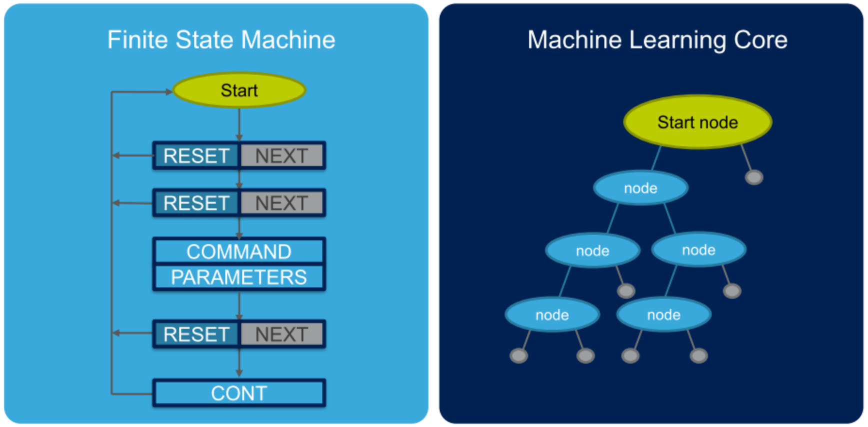


9. Tečaji, ostale vsebine Objave Datoteke Zapiski +

+ Novo Naloži Uredi v mrežnem pogledu Daj v skupno rabo Kopiraj povezavo Sinhronizacija ···

9. Tečaji, ostale vsebine > SensorTile.box > Kuchler_Analiza_in_razpoznavanje_aktivnosti_SensorTile.box > Posnetki

Ime	Spremenjeno	Spremenil	Video_channel	+ Dodajte stolpec
Orange-Klasifikacijsko_drevo	27. februar	Kuchler, Bernard		
Programska_oprema_ST-Klasifikacijsko_drevo	27. februar	Kuchler, Bernard		



FSM & MLC allows sensors to process data with reduced help of a host MCU

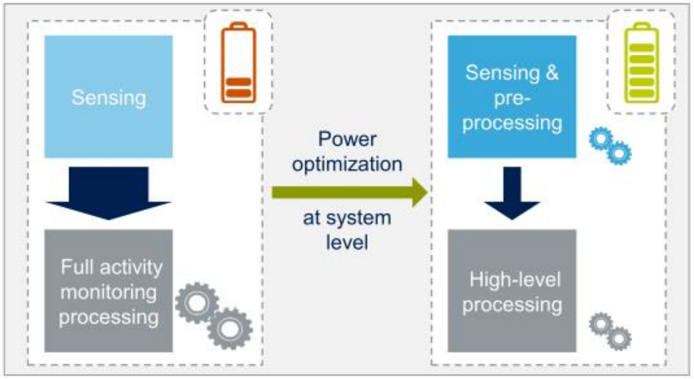
Edge computing – moduli, tipala

BHI260AP

Ultra-low power, high performance, self-learning AI smart sensor with integrated accelerometer and gyroscope

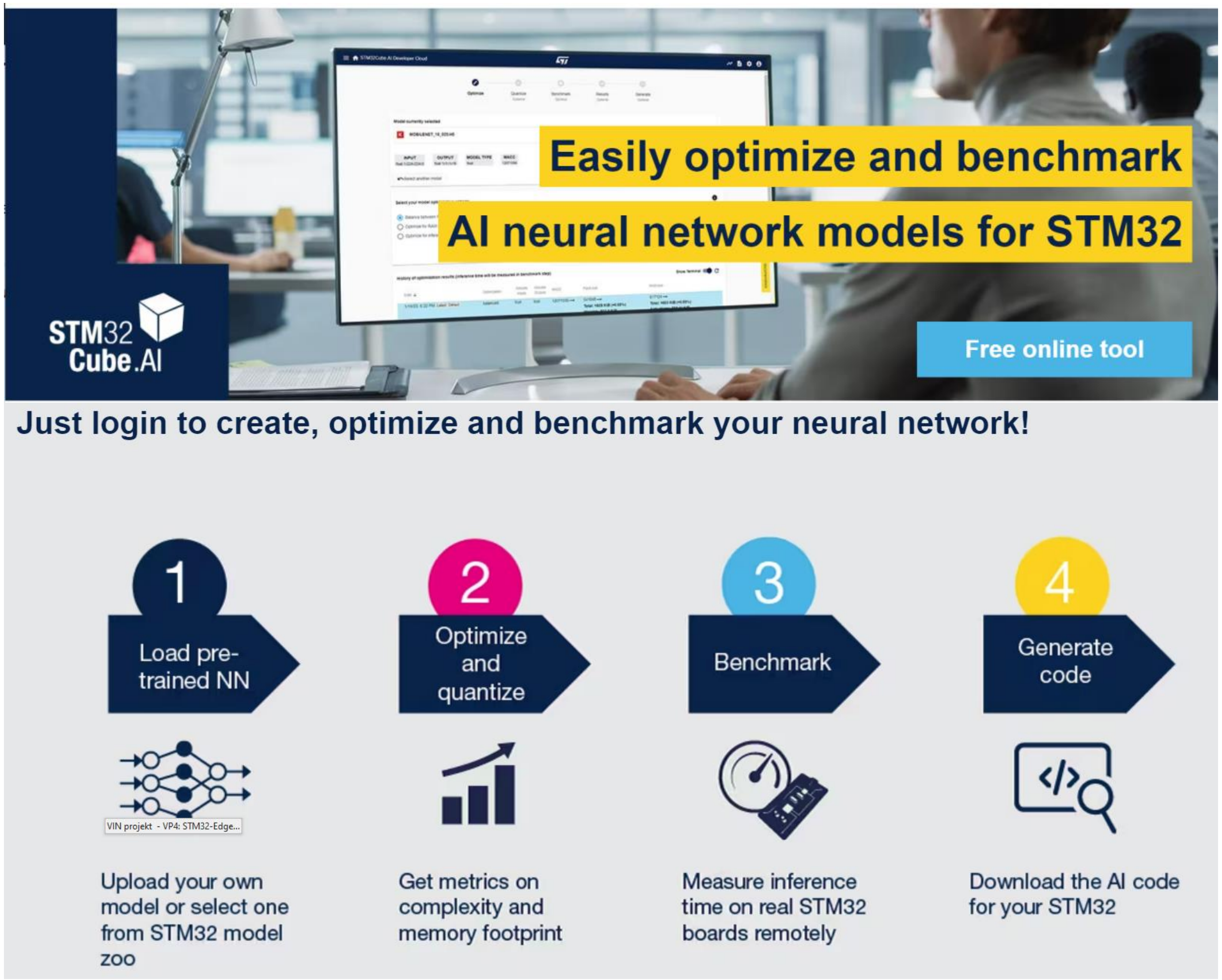
BOSCH

LSM6DSOX Unique Performance



LSM6DSOX adapter board for a standard DIL24 socket

Edge computing – Optimizacija AI modelov



Easily optimize and benchmark

AI neural network models for STM32

Free online tool

STM32 Cube.AI

Just login to create, optimize and benchmark your neural network!

1 Load pre-trained NN

2 Optimize and quantize

3 Benchmark

4 Generate code

VIN projekt - VP4: STM32-Edge...

Upload your own model or select one from STM32 model zoo

Get metrics on complexity and memory footprint

Measure inference time on real STM32 boards remotely

Download the AI code for your STM32

https://stm32ai-cs.st.com/home

Edge computing – Tensor Flow Lite in ARM Cortex-M4

MAKING MACHINE LEARNING ARDUINO COMPATIBLE

A GAMING HANDHELD THAT RUNS NEURAL NETWORKS

adafruit PyGamer
Test the TensorFlow lite voice model. Press and hold button, say YES or NO, and see if the machine learned!
version 1.12.2
TensorFlowLite
SELECT
START
A
B

A BIT OF FUN: After she created an Arduino-compatible version of TensorFlow Lite, the author adapted a voice-recognition demo so that pressing a button and speaking into a microphone attached to a SAMD51-based PyGamer would play back different animations.

A BIT OF FUN: After she created an Arduino-compatible version of TensorFlow Lite, the author adapted a voice-recognition demo so that pressing a button and speaking into a microphone attached to a SAMD51-based PyGamer would play back different animations.

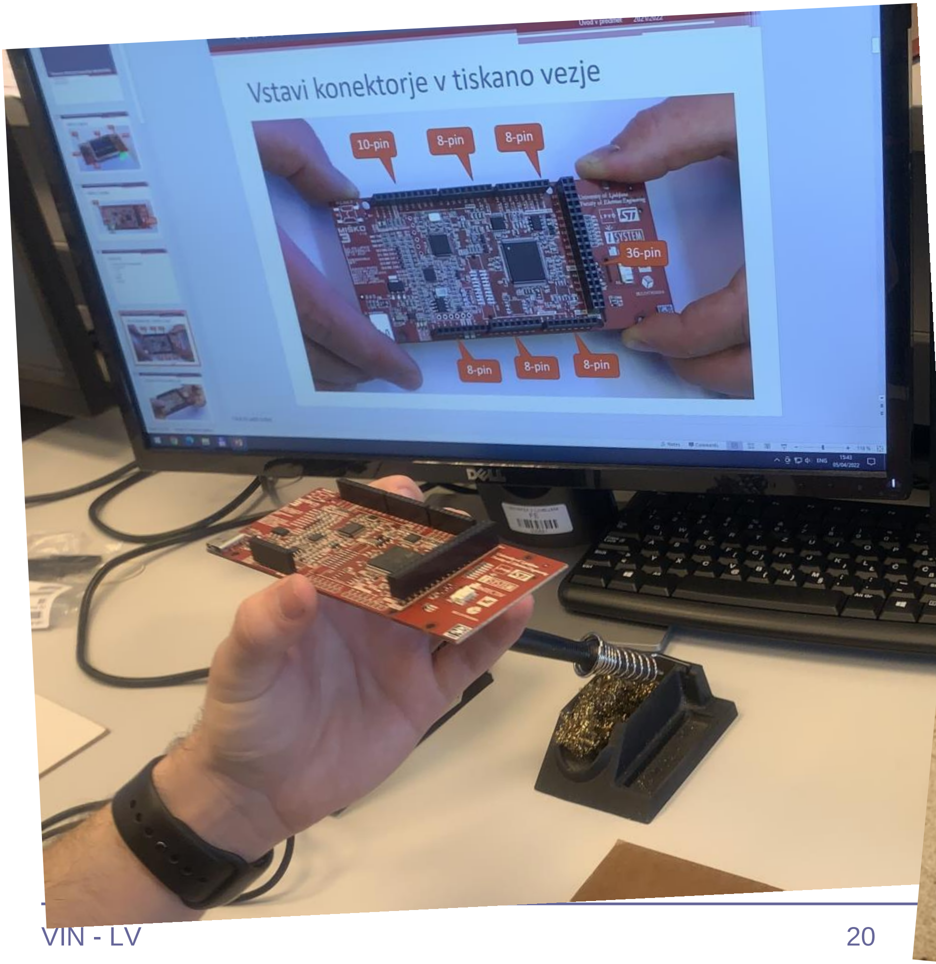
https://spectrum.ieee.org/machine-learning-thats-light-enough-for-an-arduino

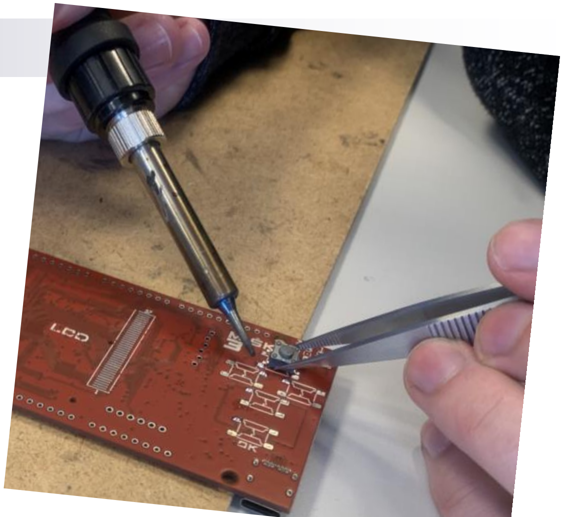
VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

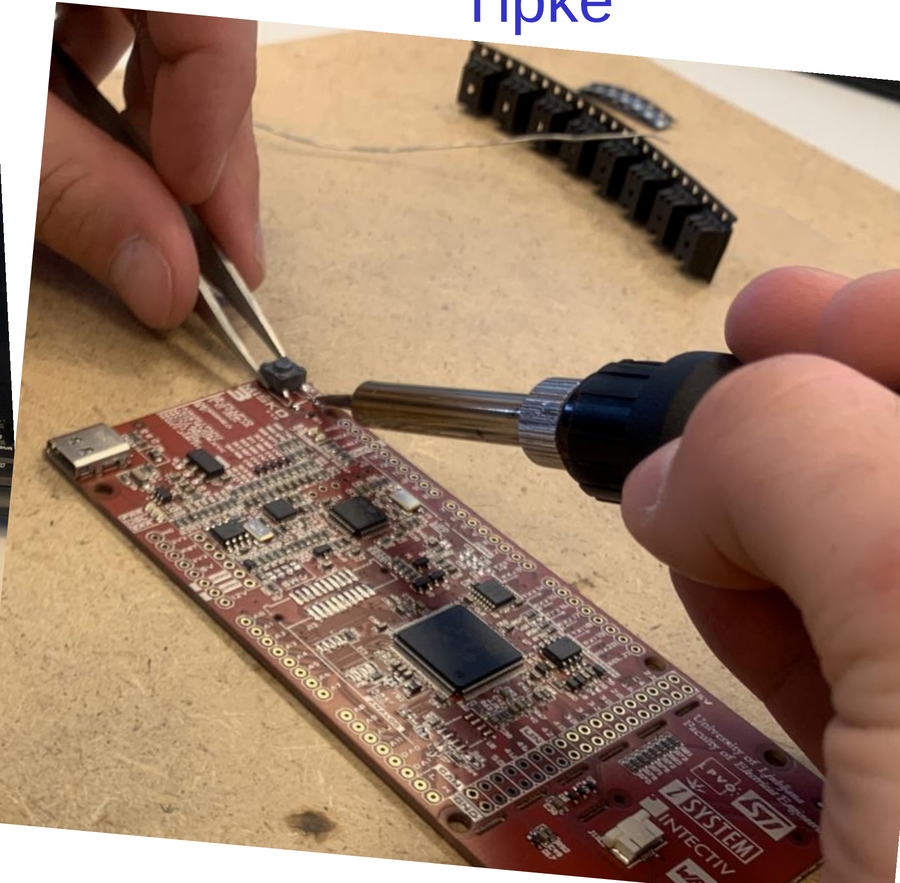
Miško 3 in „Spajka“ party 2022

Konektorji

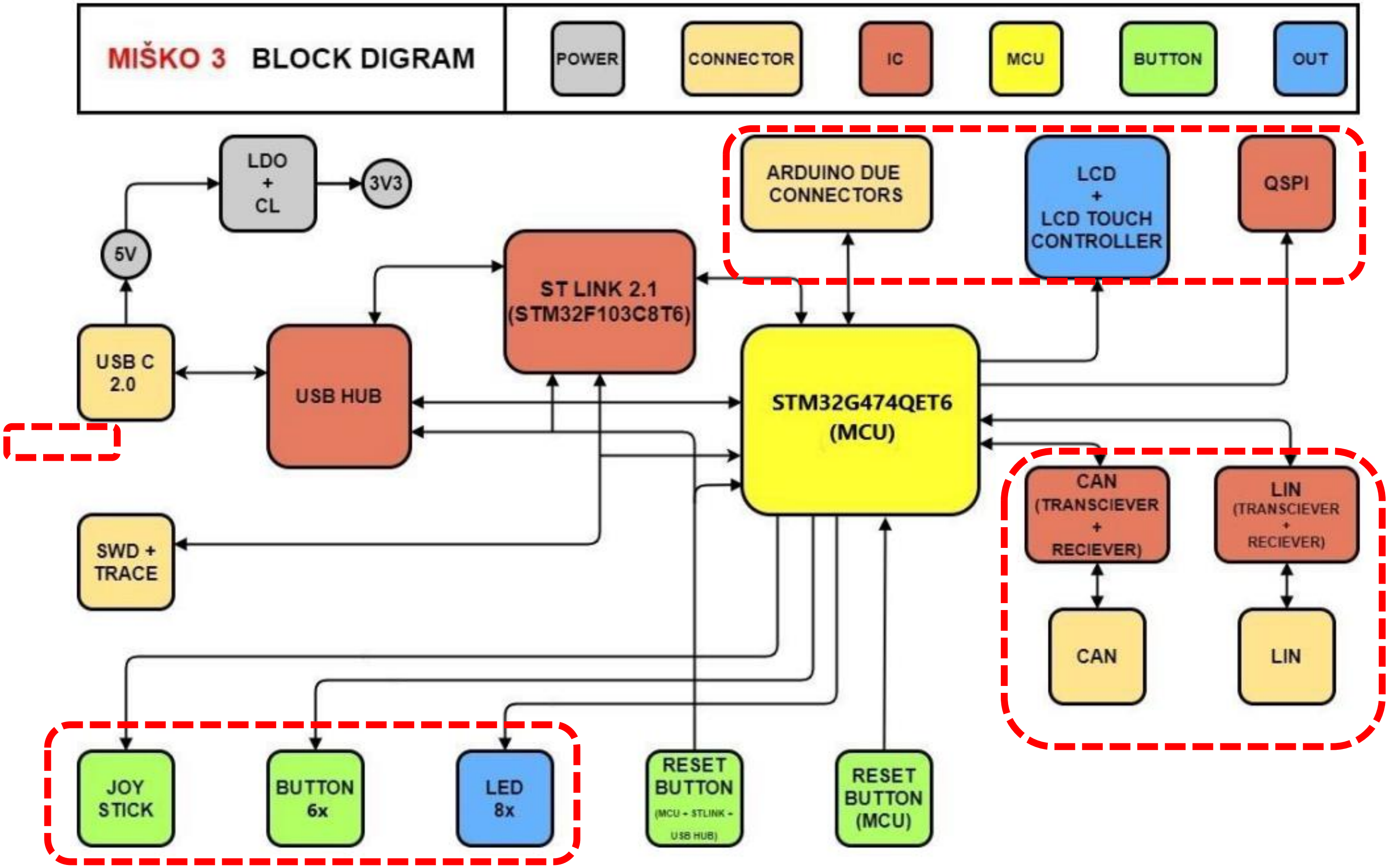




Tipke



VP – Miško 3



Slika 1: Bločni diagram razvojnega Sistema MiŠKo 3

VP – Miško 3 - Inicializacija

```c
int main(void)
{
  /* USER CODE BEGIN 1 */
coord_t joystick_raw, joystick_out;
joystick_t joystick;
uint8_t MSG[100]={0};
uint16_t touch_x = 0, touch_y = 0;

char str[10];
float framerate;

  /* USER CODE END 1 */

  /* MCU Configuration------------------
--------------------------------------*/

  /* Reset of all peripherals, Initializes
 the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */
```

```c
/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_ADC1_Init();
MX_ADC2_Init();
MX_FMC_Init();
MX_I2C2_Init();
MX_UART4_Init();
MX_UART5_Init();
MX_USART1_UART_Init();
MX_USART2_UART_Init();
MX_QUADSPI1_Init();
MX_SPI1_Init();
MX_TIM5_Init();
MX_TIM8_Init();
MX_TIM20_Init();
MX_ADC3_Init();
MX_DAC1_Init();
MX_DAC2_Init();
MX_FDCAN2_Init();
MX_I2C1_Init();
MX_TIM15_Init();
MX_USART3_UART_Init();
MX_ADC4_Init();
MX_USB_Device_Init();
MX_DMA_Init();
MX_CRC_Init();
MX_TIM6_Init();
```

```c
  /* USER CODE BEGIN 2 */

  LED_init();
  KBD_init();
  SCI_init();
  joystick_init(&joystick);

  for (uint8_t i=0;i<3;i++)
  {
  HAL_Delay(250);
  LEDs_on(0xFF);
  HAL_Delay(250);
  LEDs_off(0xFF);
  }

  LCD_Init();
  UG_Init(&gui, UserPixelSetFunction,
ILI9341_GetParam(LCD_WIDTH),
ILI9341_GetParam(LCD_HEIGHT));
  UG_FontSelect(&FONT_8X12);
  UG_SetForecolor(C_WHITE);
  UG_SetBackcolor(C_BLACK);
  UG_DriverRegister(DRIVER_FILL_FRAME, (void
*)_HW_FillFrame_);
  UG_DriverEnable(DRIVER_FILL_FRAME);

  DrawStartScreen();
  framerate = DrawColors(80);

  UG_SetForecolor(C_WHITE);
  UG_FontSelect(&FONT_16X26);
  sprintf(str,"%.0f fps",framerate);
  UG_PutString(5,105,str);

  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
```

https://github.com/LAPSyLAB/Misko3_Docs_and_Projects

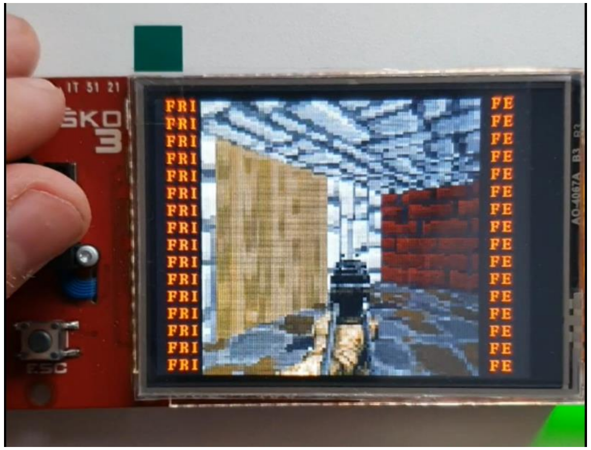
VP – Miško 3 – Glavna zanka

```
 while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */

//LEDs
  LED_set(LED0, !KBD_get_button_state(BTN_OK));
  LED_set(LED1, !KBD_get_button_state(BTN_DOWN));
  LED_set(LED2, !KBD_get_button_state(BTN_RIGHT));
  LED_set(LED3, !KBD_get_button_state(BTN_UP));
  LED_set(LED4, !KBD_get_button_state(BTN_LEFT));
  LED_set(LED6, !KBD_get_button_state(BTN_ESC));
  LED_set(LED7, !KBD_get_button_state(BTN_JOY));

 // Joystick
 HAL_ADC_Start(&hadc4);
 HAL_ADC_PollForConversion(&hadc4,10);// Waiting for ADC conversion
 joystick_raw.x=HAL_ADC_GetValue(&hadc4);

 HAL_ADC_Start(&hadc4);
 HAL_ADC_PollForConversion(&hadc4,10);// Waiting for ADC conversion
 joystick_raw.y=HAL_ADC_GetValue(&hadc4);
 HAL_ADC_Stop(&hadc4);

 joystick_get(&joystick_raw, &joystick_out, &joystick);
 UG_DrawCircle(joystick_out.x+250, joystick_out.y+50,5, C_YELLOW);
```

```
// Touchscreen
if(XPT2046_TouchPressed())
{
uint16_t x = 0, y = 0;

if(XPT2046_TouchGetCoordinates(&x, &y, 0))
{
touch_x = x;
touch_y = y;
UG_FillCircle(x, y,2, C_GREEN);
UG_FillCircle(250, 50, 49, C_BLACK);
}
}

 sprintf(MSG, "Joystick X:%05d, Y:%05d, Touch: X:%05d, Y:%05d
\r",joystick_out.x,joystick_out.y, touch_x, touch_y);

SCI_send_string(MSG);
 CDC_Transmit_FS(MSG, strlen(MSG));
 UG_DrawCircle(250, 50, 50, C_RED);

 HAL_Delay(20);
  }

  /* USER CODE END 3 */
```

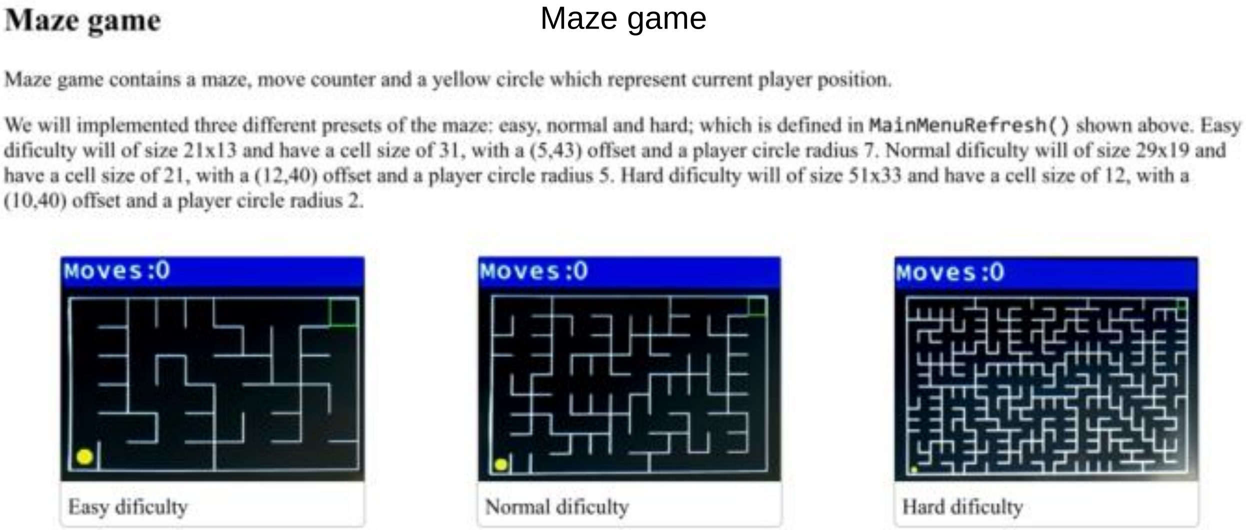
https://github.com/LAPSyLAB/Misko3_Docs_and_Projects

VP – Miško 3 – Projekti VIN 21/22



Doom

Maze game



Maze game

Maze game contains a maze, move counter and a yellow circle which represent current player position.

We will implemented three different presets of the maze: easy, normal and hard; which is defined in `MainMenuRefresh()` shown above. Easy dificulty will of size 21x13 and have a cell size of 31, with a (5,43) offset and a player circle radius 7. Normal dificulty will of size 29x19 and have a cell size of 21, with a (12,40) offset and a player circle radius 5. Hard dificulty will of size 51x33 and have a cell size of 12, with a (10,40) offset and a player circle radius 2.

Easy dificulty

Normal dificulty

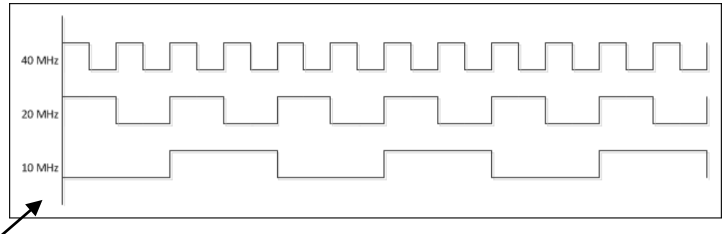
Hard dificulty

https://github.com/LAPSyLAB/Misko3_Docs_and_Projects

VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

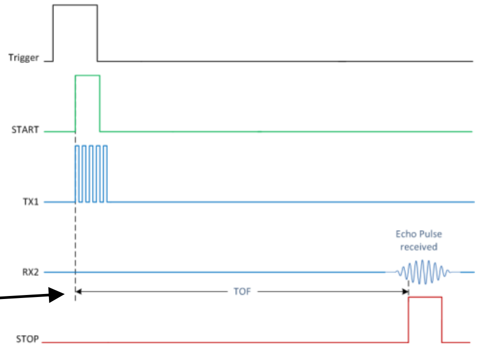
- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

Timer Counter (časovnik / števec)



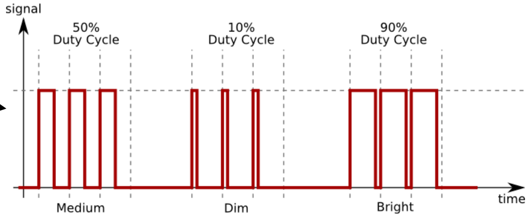
- Običajno več enakovrednih kanalov

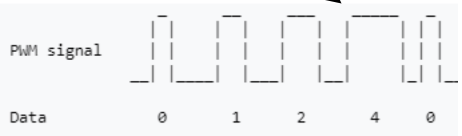
- Uporabni za
  - štetje dogodkov (Capture)
  - tvorjenje časovnih signalov (Waveform)
  - zakasnitve (DELAY s časovnikom !)
  - merjenje intervalov
  - periodične prekinitve
  - pulzno širinska modulacija (PWM)

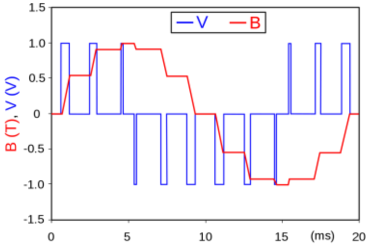


Variacije „Duty Cycle“:

- LED „dimmer“
- krmiljenje hitrosti motorjev
- „enostavni“ DAC – povprečje
- kodiranje podatkov

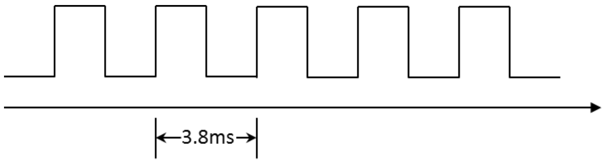






Variacije periode:

- krmiljenje servo motorjev
- približek sinusnih tonov (50% duty)
  - npr. nota C2 = 262Hz, T=1/262=3.8ms



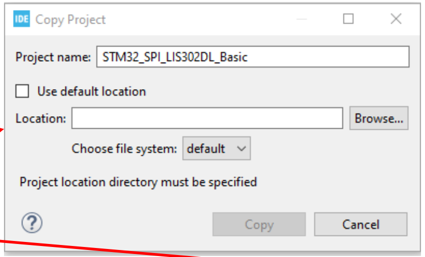
CubeIDE – delo s projekti

Kopiranje/preimenovanje projekta :

- **Kopiranje projekta Cube MX I**:
  - Znotraj CubeIDE

Kopiranje CubeIDE projekta z CubeMX .ioc datoteko

1) Edit > Copy.
2) Edit > Paste.
3) Preimenuj .ioc datoteko.
4) Zbriši Debug.launch datoteko.
5) Project > Clean.
6) Generiraj kodo s CubeMX.
7) Project > Build Project.
8) Debug As Stm32 Application.
9) Debug aplikacije.



Copy Project

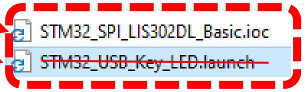
Project name: STM32_SPI_LIS302DL_Basic

☐ Use default location

Location:

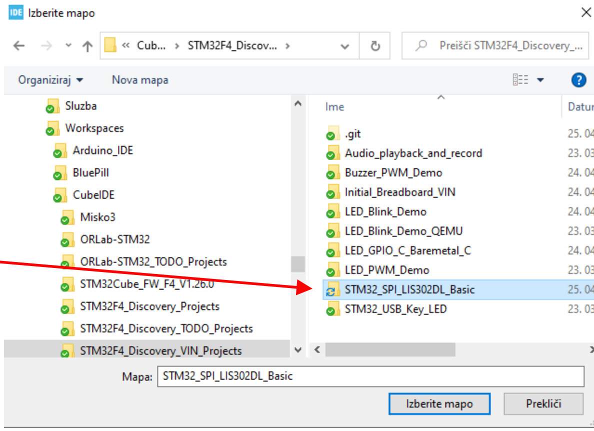
Choose file system: default

Project location directory must be specified



STM32_SPI_LIS302DL_Basic.ioc

~~STM32_USB_Key_LED.launch~~



Izberite mapo

« Cub... › STM32F4_Discov... ›

Preišči STM32F4_Discovery_...

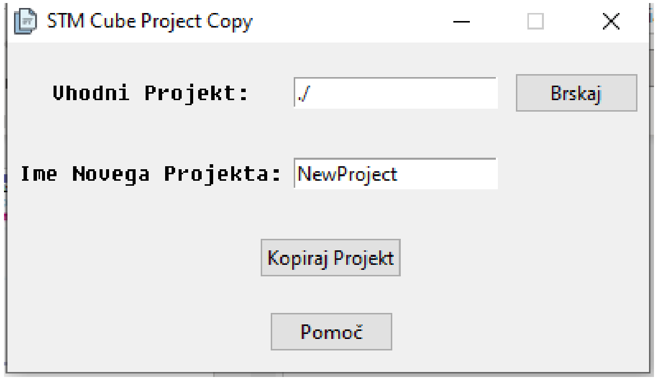
Sluzba
Workspaces
Arduino_IDE
BluePill
CubeIDE
Misko3
ORLab-STM32
ORLab-STM32_TODO_Projects
STM32Cube_FW_F4_V1.26.0
STM32F4_Discovery_Projects
STM32F4_Discovery_TODO_Projects
STM32F4_Discovery_VIN_Projects

Ime	Datu
.git	25. 04
Audio_playback_and_record	23. 03
Buzzer_PWM_Demo	24. 04
Initial_Breadboard_VIN	24. 04
LED_Blink_Demo	24. 04
LED_Blink_Demo_QEMU	23. 03
LED_GPIO_C_Baremetal_C	24. 04
LED_PWM_Demo	23. 03
STM32_SPI_LIS302DL_Basic	25. 04
STM32_USB_Key_LED	23. 03

Mapa: STM32_SPI_LIS302DL_Basic

Skopiram, preimenujem ioc, generiram kodo, brišem Debug.launch, clean in build

- **Kopiranje projekta Cube MX II**:
  - Uporaba orodja



STM Cube Project Copy

Vhodni Projekt: ./

Ime Novega Projekta: NewProject

https://github.com/LAPSyLAB/STM32F4_Docs_and_Examples/tree/main/Documentation/CubeIDE

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

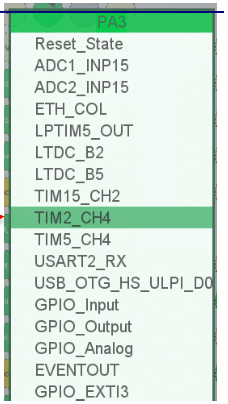
2. Spremeniti pin PA3 v TIM2->CH4
tim2 kanal 4 spremeniti na PWM Generation CH4

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

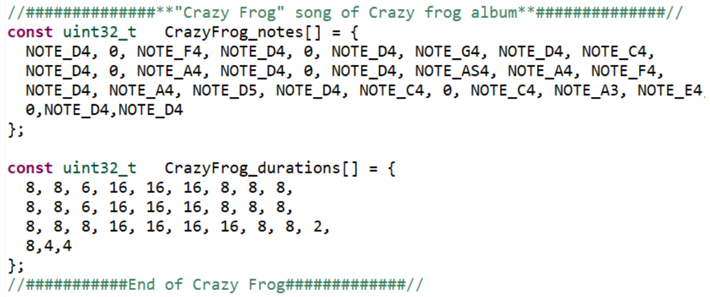
Perioda štetja se bo določala glede na noto (duty cycle je vedno 50%)
Counter Period (AutoReload Register - 16 bits value) =
ARR = 1000000 (ura števca) / Frekv.note[Hz]
CCR1 bo vedno ARR/2 (50% duty cycle)

Več informacij : BeriMe.txt

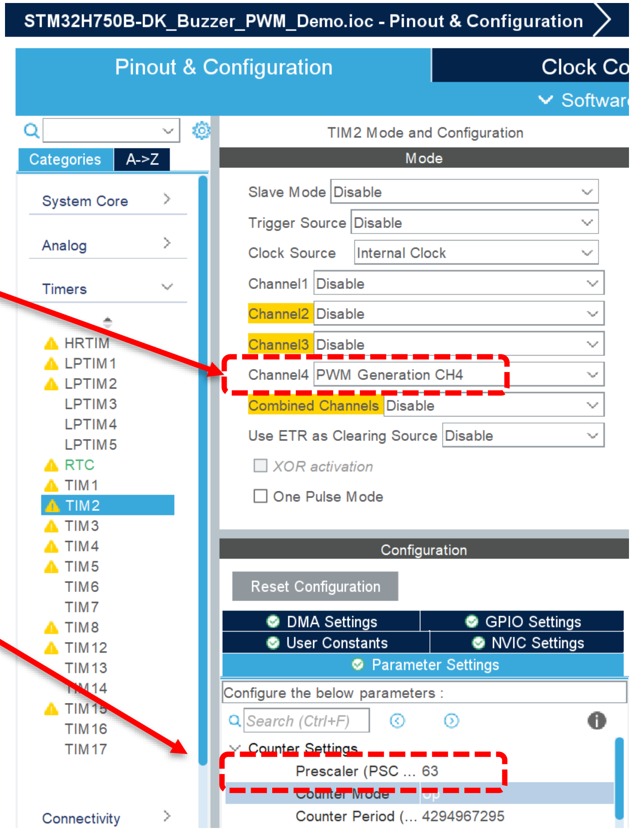
Primer: Nota A4 = 440 Hz
Perioda[s]=1/440Hz=0.00227272
Perioda[us]=1/440*1000000=2272



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############//
```



STM32H750B-DK_Buzzer_PWM_Demo.ioc - Pinout & Configuration

TIM2 Mode and Configuration

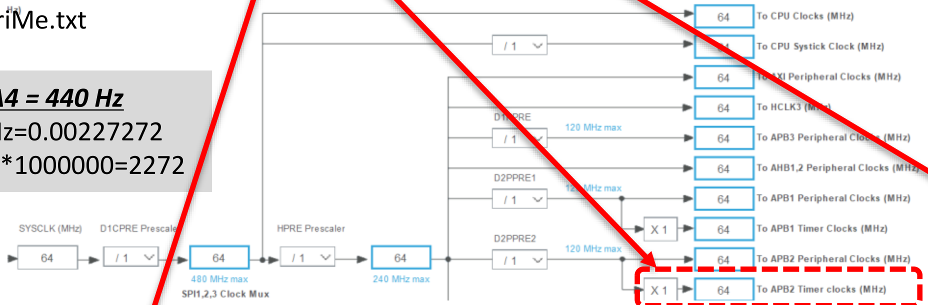
Mode

Slave Mode Disable
Trigger Source Disable
Clock Source Internal Clock
Channel1 Disable
Channel2 Disable
Channel3 Disable
Channel4 PWM Generation CH4
Combined Channels Disable
Use ETR as Clearing Source Disable
☐ XOR activation
☐ One Pulse Mode

Configuration

Configure the below parameters :

Counter Settings
Prescaler (PSC ... 63
Counter Mode Up
Counter Period (... 4294967295


SYSCLK (MHz)
64
D1CPRE Prescaler
/ 1
64
480 MHz max
SPI1,2,3 Clock Mux
HPRE Prescaler
/ 1
64
240 MHz max
/ 1
D1PPRE
/ 1
120 MHz max
D2PPRE1
/ 1
120 MHz max
D2PPRE2
/ 1
120 MHz max
X 1
X 1
64
To CPU Clocks (MHz)
64
To CPU Systick Clock (MHz)
64
To AXI Peripheral Clocks (MHz)
64
To HCLK3 (MHz)
64
To APB3 Peripheral Clocks (MHz)
64
To AHB1,2 Peripheral Clocks (MHz)
64
To APB1 Peripheral Clocks (MHz)
64
To APB1 Timer Clocks (MHz)
64
To APB2 Peripheral Clocks (MHz)
64
To APB2 Timer clocks (MHz)

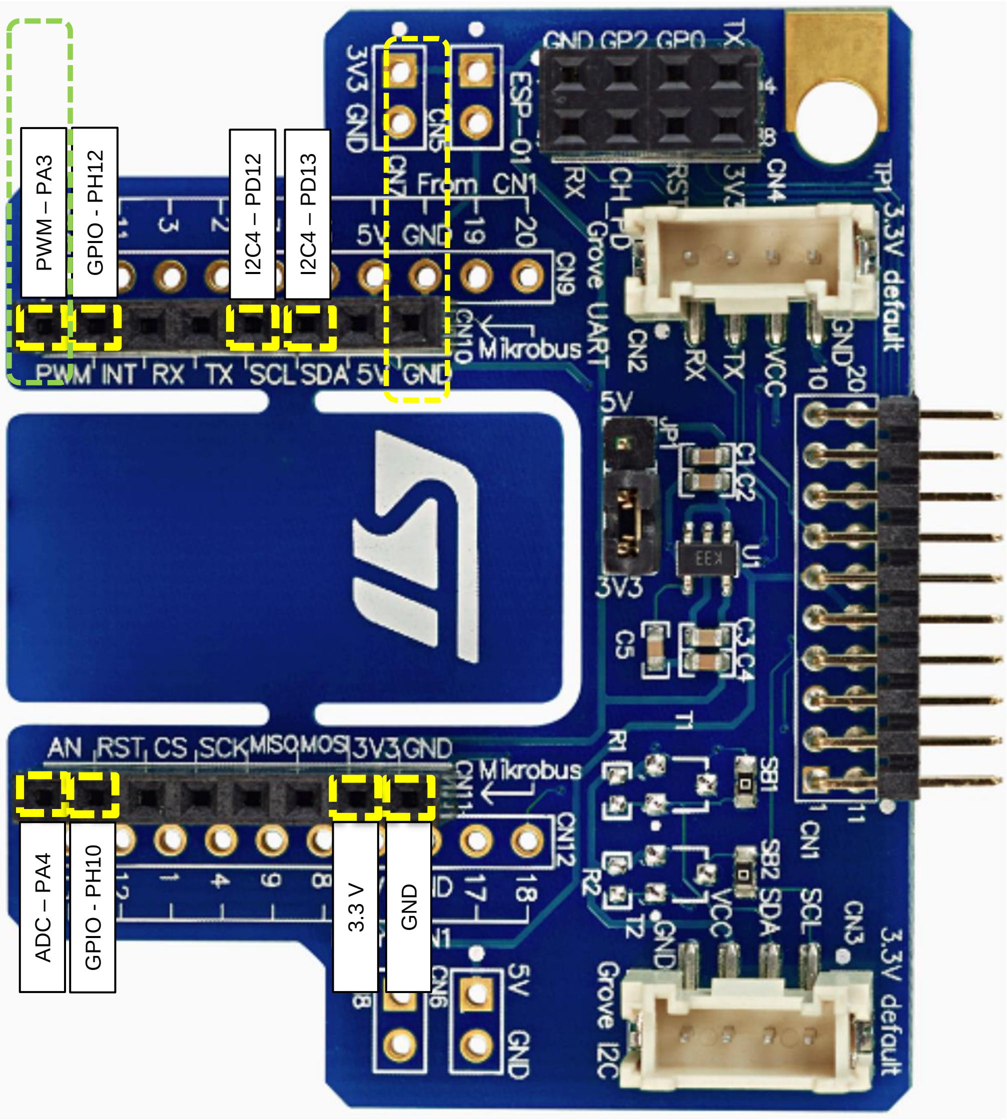
Nota:

```
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];
```

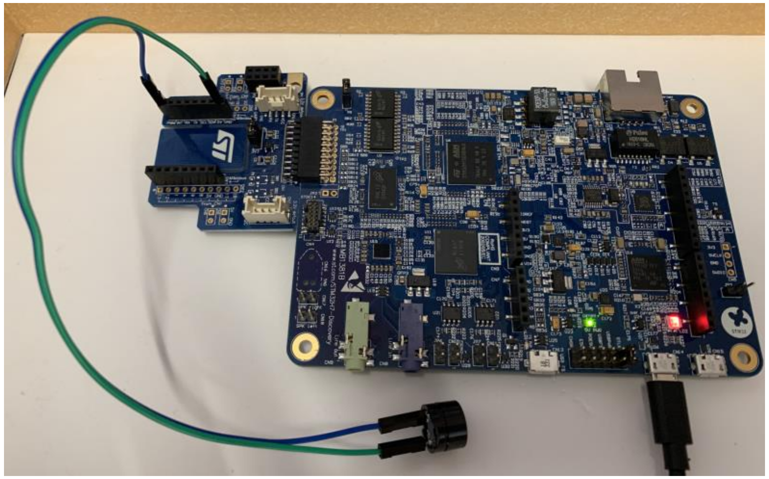
STM32H7

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

STM32H7 – PWM signali/melodija za brenčača (Buzzer)



Pravilna priključitev



Nepravilna priključitev

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

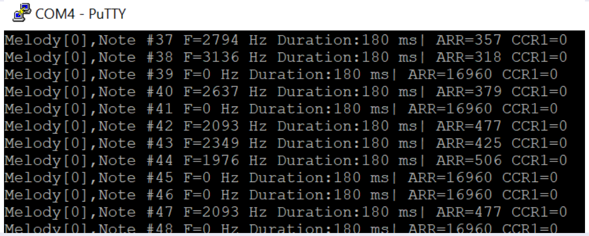
STM32H7 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

```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:

```c
//##############**"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:

```c
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)

STM32F4

STM32H7

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));

}
…
```

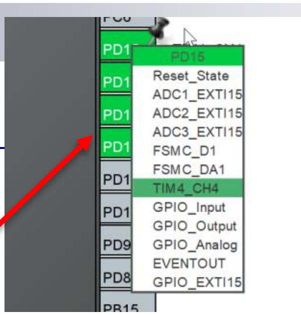
Melody.h:

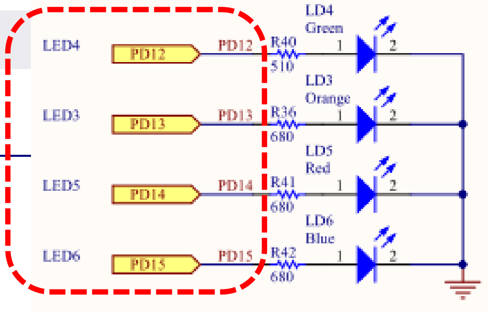
```
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

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############//
```

STM32F4 – PWM signali za LED diode (LED dimmer)





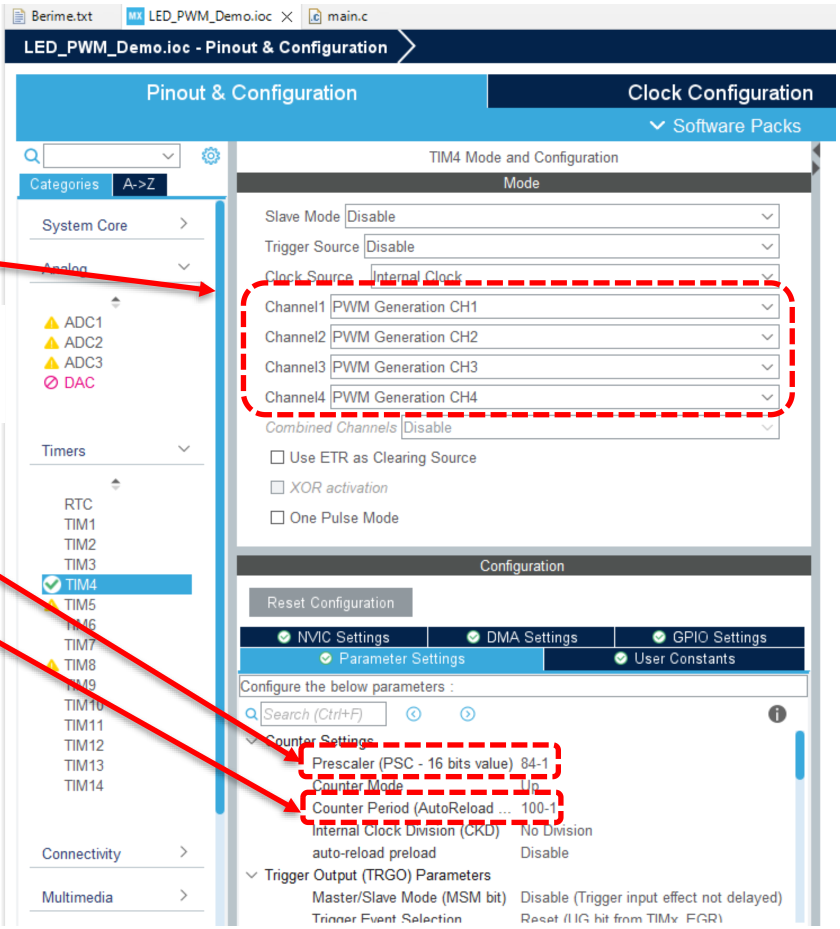
HAL - C

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 pine PD12-PD15 (LED diode) v TIM4_CH1-4
   tim4 Vse kanale spremeniti na PWM Generation CH1-4

4. Clock :
   Ura števca = 1 MHz
   Prescaler (PSC - 16 bits value) Prescaler (PSC - 16 bits v
   must be between 0 and 65 535 = **84-1 = 83** (clock 1M
   Perioda štetja je 100 (duty cycle pa lahko 0-100)
   Counter Period (AutoReload Register - 16 bits value) Counter
   Period (AutoReload Register - 16 bits value) **= 100-1 = 99**



TIM4 Mode and Configuration

Mode

Slave Mode: Disable
Trigger Source: Disable
Clock Source: Internal Clock
Channel1: PWM Generation CH1
Channel2: PWM Generation CH2
Channel3: PWM Generation CH3
Channel4: PWM Generation CH4
Combined Channels: Disable
☐ Use ETR as Clearing Source
☐ XOR activation
☐ One Pulse Mode

Configuration

Configure the below parameters :

Counter Settings
- Prescaler (PSC - 16 bits value): 84-1
- Counter Mode: Up
- Counter Period (AutoReload ...: 100-1
- Internal Clock Division (CKD): No Division
- auto-reload preload: Disable

Trigger Output (TRGO) Parameters
- Master/Slave Mode (MSM bit): Disable (Trigger input effect not delayed)
- Trigger Event Selection: Reset (UG bit from TIMx_EGR)

STM32F4

Osnovni projekt CubeIDE – GPIO – PWM, LED diode

HAL - C

```
  /* USER CODE BEGIN PV */
  #define BUFSIZE 256
  char SendBuffer[BUFSIZE];

  /* USER CODE END PV */
  /* USER CODE BEGIN 2 */

 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_2);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_3);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_4);

 /* USER CODE END 2 */

/* Infinite loop */
 /* USER CODE BEGIN WHILE */
 while (1)
 {
   htim4.Instance->CCR1 = duty;
   htim4.Instance->CCR2 = 100-duty;
   htim4.Instance->CCR3 = duty;
   htim4.Instance->CCR4 = 100-duty;

   /* USER CODE END WHILE */

   /* USER CODE BEGIN 3 */
   snprintf (SendBuffer,BUFSIZE,"USB:0.1 secs. Duty=%d%%\r\n",duty);
   CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

     duty = (duty + 1) ;
     if (duty > 100 )
       duty = 0;


   HAL_Delay(100);

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

CubeMX - dodatne spremembe osnovnega projekta :

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 pine PD12-PD15 (LED diode) v TIM4_CH0-3
   tim4 Vse kanale spremeniti na PWM Generation CH0-3

4. Clock :
   Ura števca = 1 MHz
   Prescaler (PSC - 16 bits value) Prescaler (PSC - 16 bits value) must be between 0 and 65 535 = 84-1 = 83 (clock 1Mhz)

   Perioda štetja je 100 (duty cycle pa lahko 0-100)
   Counter Period (AutoReload Register - 16 bits value) Counter Period (AutoReload Register - 16 bits value) = 100-1 = 99

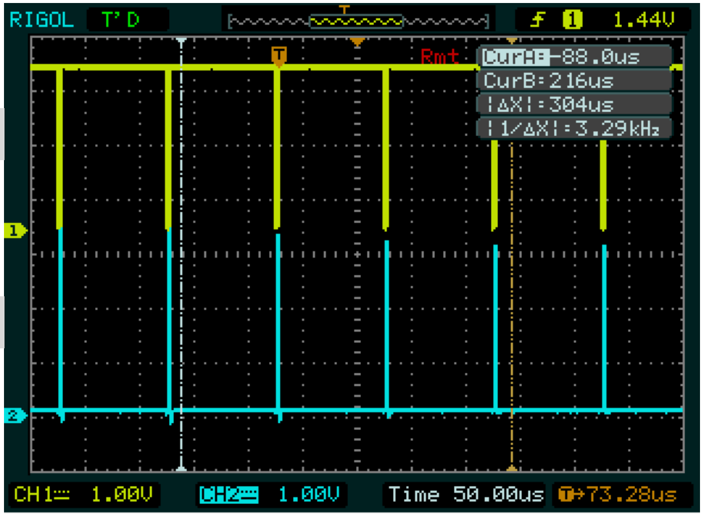
https://github.com/LAPSyLAB/STM32F4_Discovery_VIN_Projects/tree/main/LED_PWM_Demo

STM32F4

Osnovni projekt CubeIDE – GPIO – PWM, LED diode

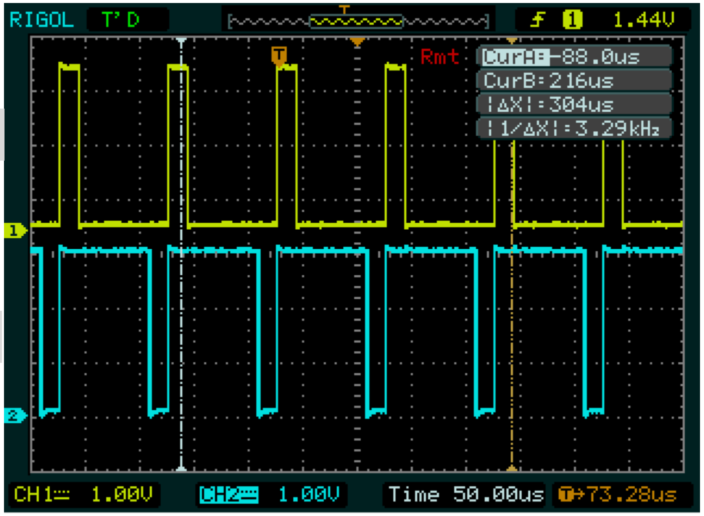
HAL - C

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
char SendBuffer[BUFSIZE];

/* USER CODE END PV */
/* USER CODE BEGIN 2 */

 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_1);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_2);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_3);
 HAL_TIM_PWM_Start(&htim4, TIM_CHANNEL_4);

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  htim4.Instance->CCR1 = duty;
  htim4.Instance->CCR2 = 100-duty;
  htim4.Instance->CCR3 = duty;
  htim4.Instance->CCR4 = 100-duty;

  /* USER CODE END WHILE */

  /* USER CODE BEGIN 3 */
  snprintf (SendBuffer,BUFSIZE,"USB:0.1 secs. Duty=%d%%\r\n",duty);
  CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

    duty = (duty + 1) ;
    if (duty > 100 )
      duty = 0;


  HAL_Delay(100);

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



Max Duty

Min Duty



Min Duty

Max Duty

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

STM32F4 – PWM signali/melodija za brenčača (Buzzer)

HAL - C

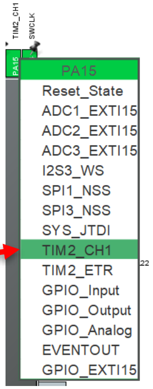
Brencac se prikljuci na **PA15 (TIM2->CH1) in GND**

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

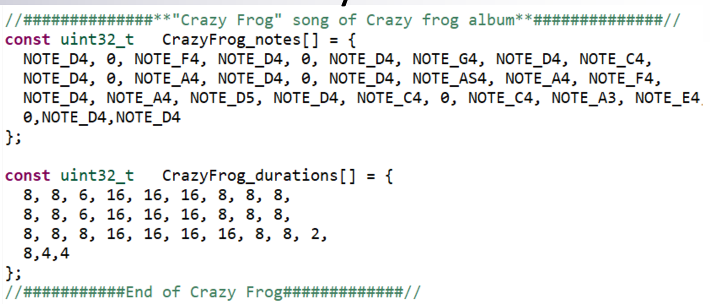
3. Spremeniti pin PA15 v TIM2->CH1
tim2 kanal 1 spremeniti na PWM Generation CH1

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)

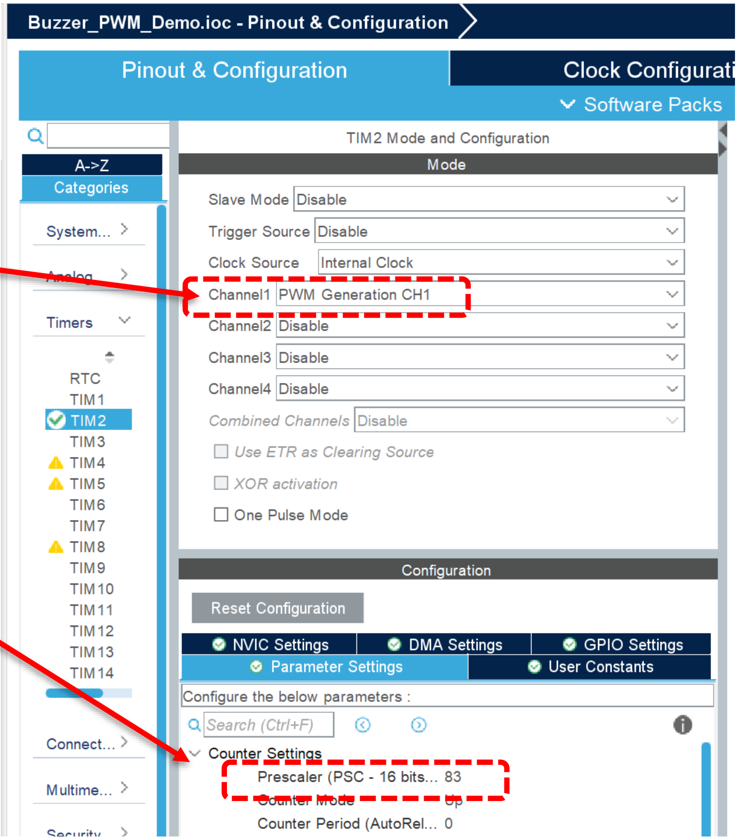
Več informacij : BeriMe.txt



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############//
```



Buzzer_PWM_Demo.ioc - Pinout & Configuration

Pinout & Configuration | Clock Configurati | Software Packs

TIM2 Mode and Configuration

Mode

Slave Mode: Disable
Trigger Source: Disable
Clock Source: Internal Clock
Channel1: PWM Generation CH1
Channel2: Disable
Channel3: Disable
Channel4: Disable
Combined Channels: Disable
□ Use ETR as Clearing Source
□ XOR activation
□ One Pulse Mode

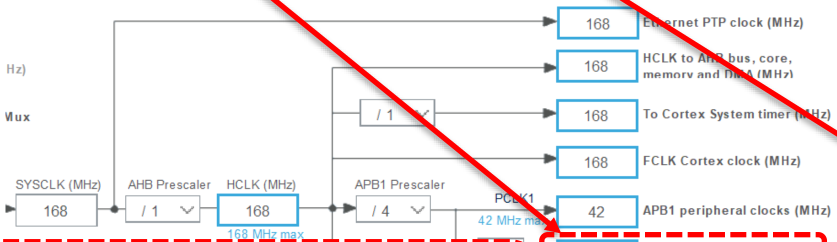
Configuration

Reset Configuration

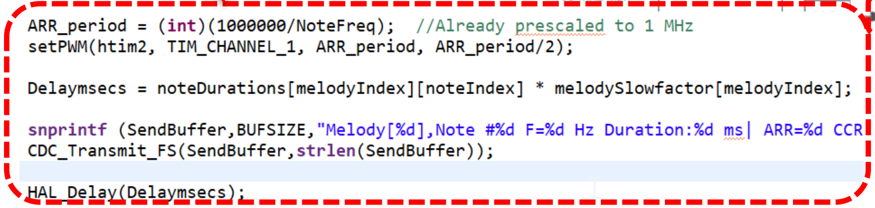
NVIC Settings | DMA Settings | GPIO Settings | Parameter Settings | User Constants

Configure the below parameters :

Counter Settings
Prescaler (PSC - 16 bits... 83
Counter Mode Up
Counter Period (AutoRel... 0



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#############//
```

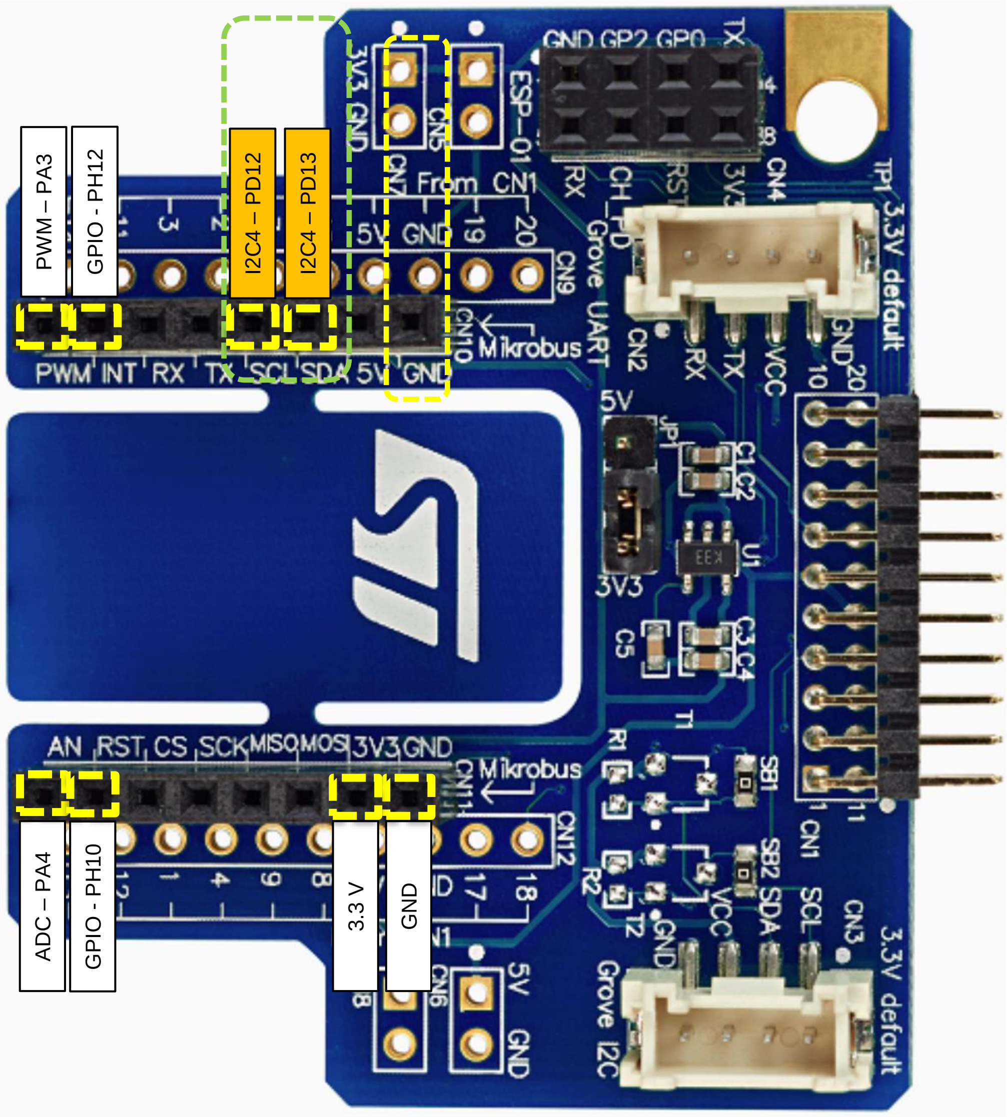
```
Melody.h:
const uint32_t* melody[] ={marioMelody, secondMelody,
Titanic_Melody,Pirates_notes,CrazyFrog_notes};
const uint32_t* noteDurations[] = {marioDuration, secondDuration,
Titanic_duration,Pirates_durations,CrazyFrog_durations};
const uint16_t melodySlowfactor[] ={15, 30, 20, 20, 20};

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

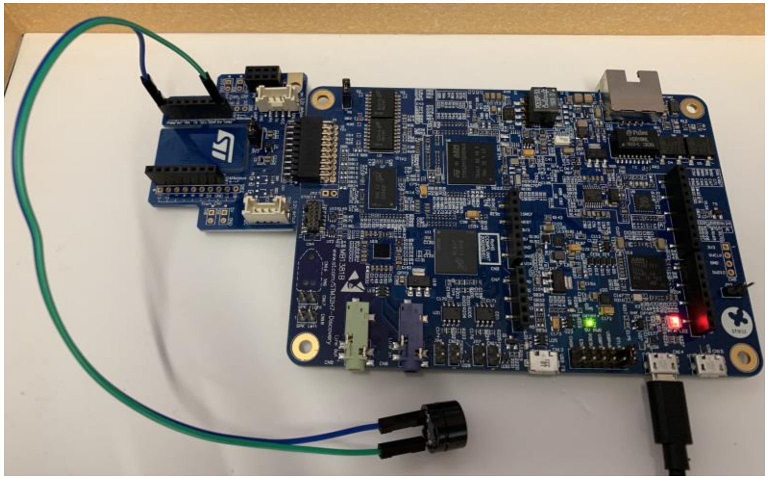
VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

STM32H7 – PWM signali/melodija za brenčača (Buzzer)



Pravilna priključitev



Nepravilna priključitev

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

VP 4 - STM32H7 CubeIDE, I2C4 scanner

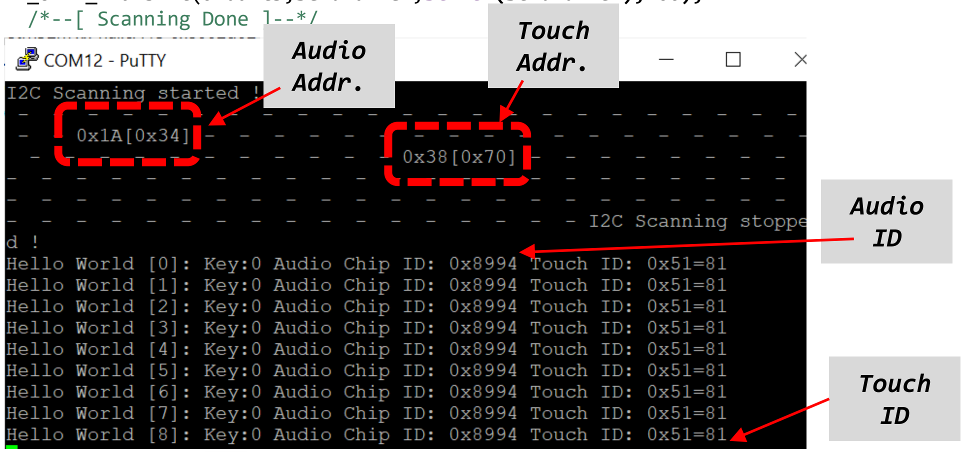
Spremenljivke

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
charSendBuffer[BUFSIZE];
intCounter;
int KeyState=0;
uint8_t dataBuffer[10];

HAL_StatusTypeDef retval;
uint8_t Answer;
```

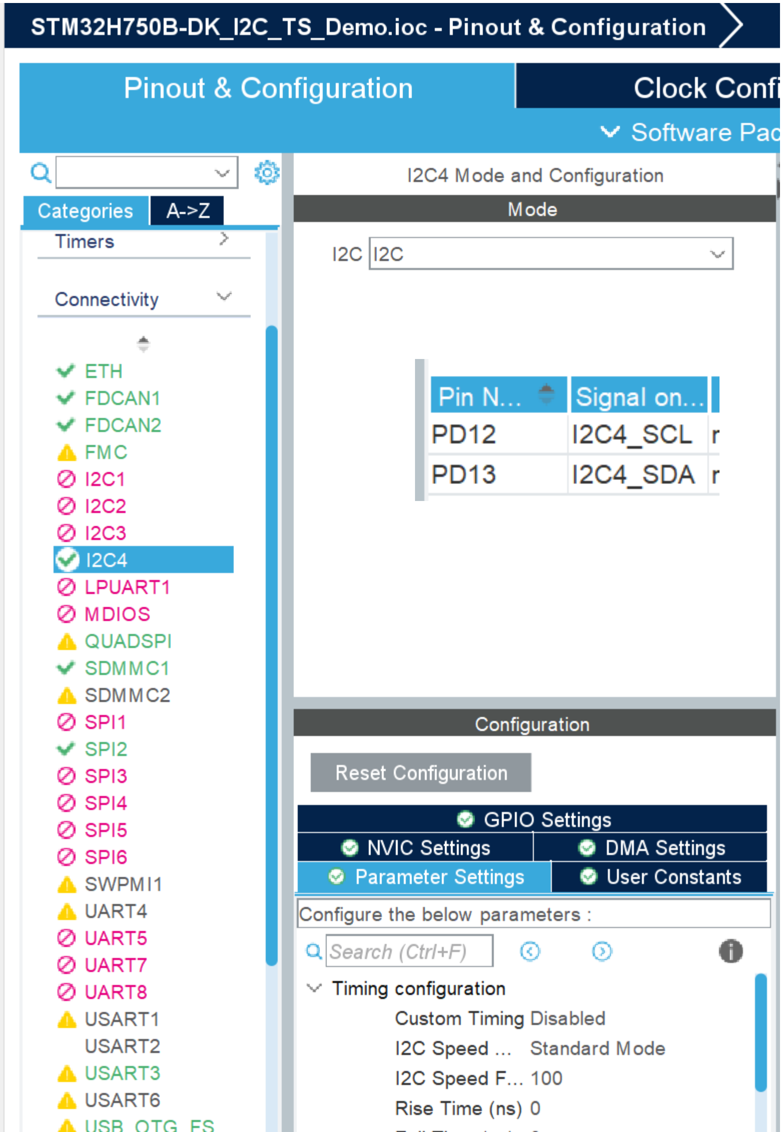
main.c : dodana koda

I2C Scan (vseh naprav)

```
HAL_GPIO_WritePin(GPIOB, GPIO_PIN_12, 1);    // Set LCD_RST to high

 /*-[ I2C Bus Scanning ]-*/
snprintf(SendBuffer,BUFSIZE,"I2C Scanning started !\n\r");
HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);

 for(i=1; i<128; i++)
 {
        retval = HAL_I2C_IsDeviceReady(&hi2c4, (uint16_t)(i<<1), 3, 5);
        if (retval != HAL_OK) /* No ACK Received At That Address */
        {
                HAL_UART_Transmit(&huart3, Space, sizeof(Space), 100);
        }
        else if(retval == HAL_OK)
        {
                snprintf(SendBuffer,BUFSIZE,"0x%02X[0x%02X]", i, i<<1);

                HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),1);
        }
 }
snprintf(SendBuffer,BUFSIZE,"I2C Scanning stopped !\n\r");
HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);
   /*--[ Scanning Done ]--*/
```



```
I2C Scanning started !
 - - - - - - - - - - - - - - - - - - - - - - -
 -   0x1A[0x34]- - - - - - - - - - - - - - - - - -
  - - - - - - - - - - - 0x38[0x70]- - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - I2C Scanning stoppe
d !
Hello World [0]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [1]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [2]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [3]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [4]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [5]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [6]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [7]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [8]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
```



I2C4 Mode and Configuration

Mode

I2C: I2C

Pin N...	Signal on...
PD12	I2C4_SCL
PD13	I2C4_SDA

Configuration

Configure the below parameters :

Timing configuration
- Custom Timing Disabled
- I2C Speed ... Standard Mode
- I2C Speed F... 100
- Rise Time (ns) 0

STM32H7

STM32H7

VP 4 - STM32H7 CubeIDE, I2C4 scanner

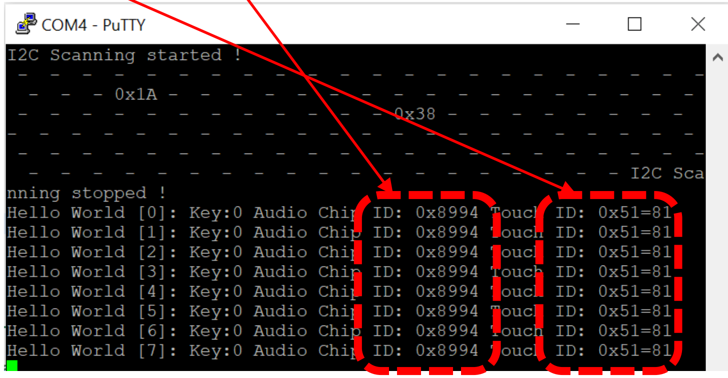
main.c : dodana koda

Spremenljivke

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
charSendBuffer[BUFSIZE];
intCounter;
int KeyState=0;
uint8_t dataBuffer[10];
int i=0;
uint8_t Space[] = " - ";

HAL_StatusTypeDef retval;
/* USER CODE END PV */
```

Glavna zanka

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

    KeyState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13);
    HAL_GPIO_WritePin(GPIOJ, GPIO_PIN_2, KeyState);

…

// Reading from address 0x1a register R0 (addr. 0x00) default value should be 0x8994 - Both variations work !
    //dataBuffer[0] = 0; dataBuffer[1] = 0x00;
    //retval = HAL_I2C_Master_Transmit(&hi2c4, (0x1a << 1), dataBuffer, 2, HAL_MAX_DELAY);
    //retval = HAL_I2C_Master_Receive(&hi2c4, (0x1a << 1), dataBuffer, 2, HAL_MAX_DELAY);
    retval = HAL_I2C_Mem_Read(&hi2c4, (0x1a << 1), 0, I2C_MEMADD_SIZE_16BIT,dataBuffer, 2, HAL_MAX_DELAY);

// Reading from address 0x38 register Vendor's Chip ID (addr. 0xA8) default value should be 0x51=81  - Both variations work !
    //dataBuffer[5] = 0xA8;
    //retval = HAL_I2C_Master_Transmit(&hi2c4, (0x38 << 1), &dataBuffer[5], 1, HAL_MAX_DELAY);
    //retval = HAL_I2C_Master_Receive(&hi2c4, (0x38 << 1), &dataBuffer[5], 1, HAL_MAX_DELAY);
    retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0xA8, I2C_MEMADD_SIZE_8BIT,&dataBuffer[5], 1, HAL_MAX_DELAY);


    snprintf(SendBuffer,BUFSIZE,"Hello World [%d]: Key:%d Audio Chip ID: 0x%4x Touch ID: 0x%2x=%d\n\r",Counter++,KeyState,
dataBuffer[0]*256+dataBuffer[1],dataBuffer[5],dataBuffer[5]);
    HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);

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

    /* USER CODE BEGIN 3 */

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



COM4 - PuTTY

```
I2C Scanning started !
 - - - - - - - - - - - - - - - - - - - - - -
  - - - 0x1A - - - - - - - - - - - - - - - - -
 - - - - - - - - - - - - 0x38 - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - -
 - - - - - - - - - - - - - - - - - - - - - -
  - - - - - - - - - - - - - - - - - - - I2C Sca
nning stopped !
Hello World [0]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [1]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [2]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [3]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [4]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [5]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [6]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
Hello World [7]: Key:0 Audio Chip ID: 0x8994 Touch ID: 0x51=81
```

VP 4 - STM32H7 CubeIDE, I2C Audio WM9884 Gradiva

CIRRUS LOGIC®

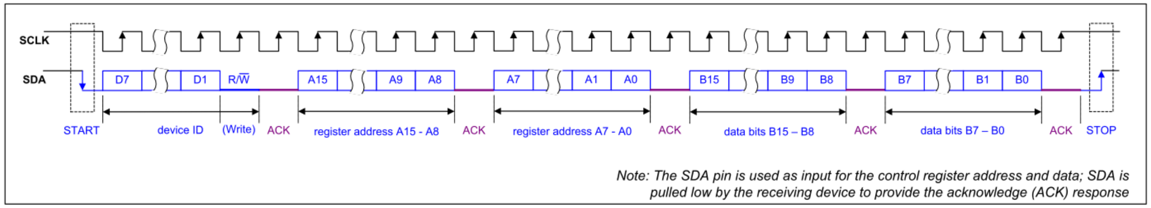
WM8994

Multi-channel Audio Hub CODEC for Smartphones

16-bitni naslovi in 16-bitni registri !

2-WIRE (I2C) CONTROL MODE

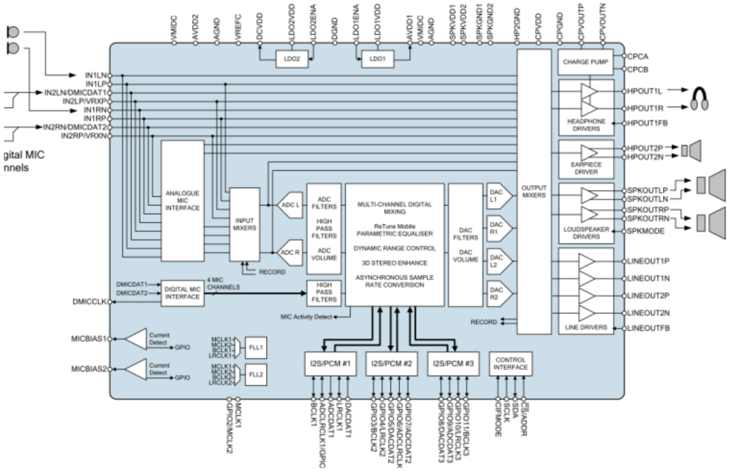
The sequence of signals associated with a single register write operation is illustrated in Figure 72.



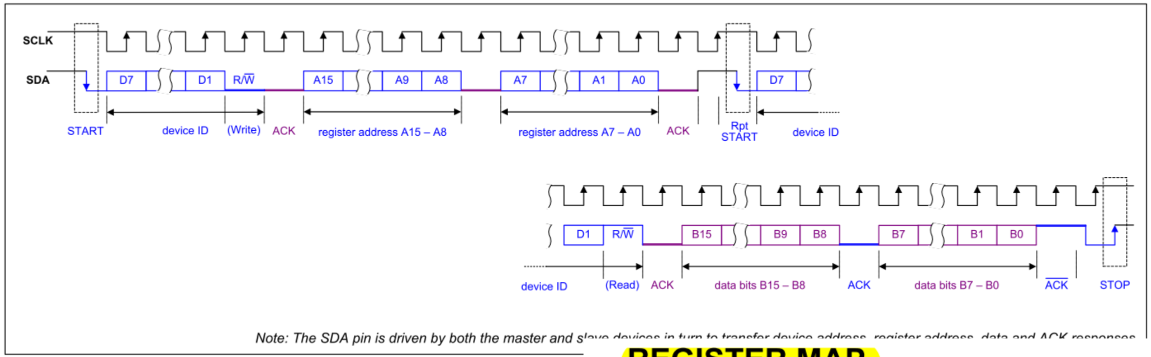
Note: The SDA pin is used as input for the control register address and data; SDA is pulled low by the receiving device to provide the acknowledge (ACK) response

[Brez naslova]

Figure 72 Control Interface 2-wire (I2C) Register Write



The sequence of signals associated with a single register read operation is illustrated in Figure 73.



Note: The SDA pin is driven by both the master and slave devices in turn to transfer device address, register address, data and ACK responses

Figure 73 Control Interface 2-wire (I2C) Register Read

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_TS_Demo

REGISTER MAP

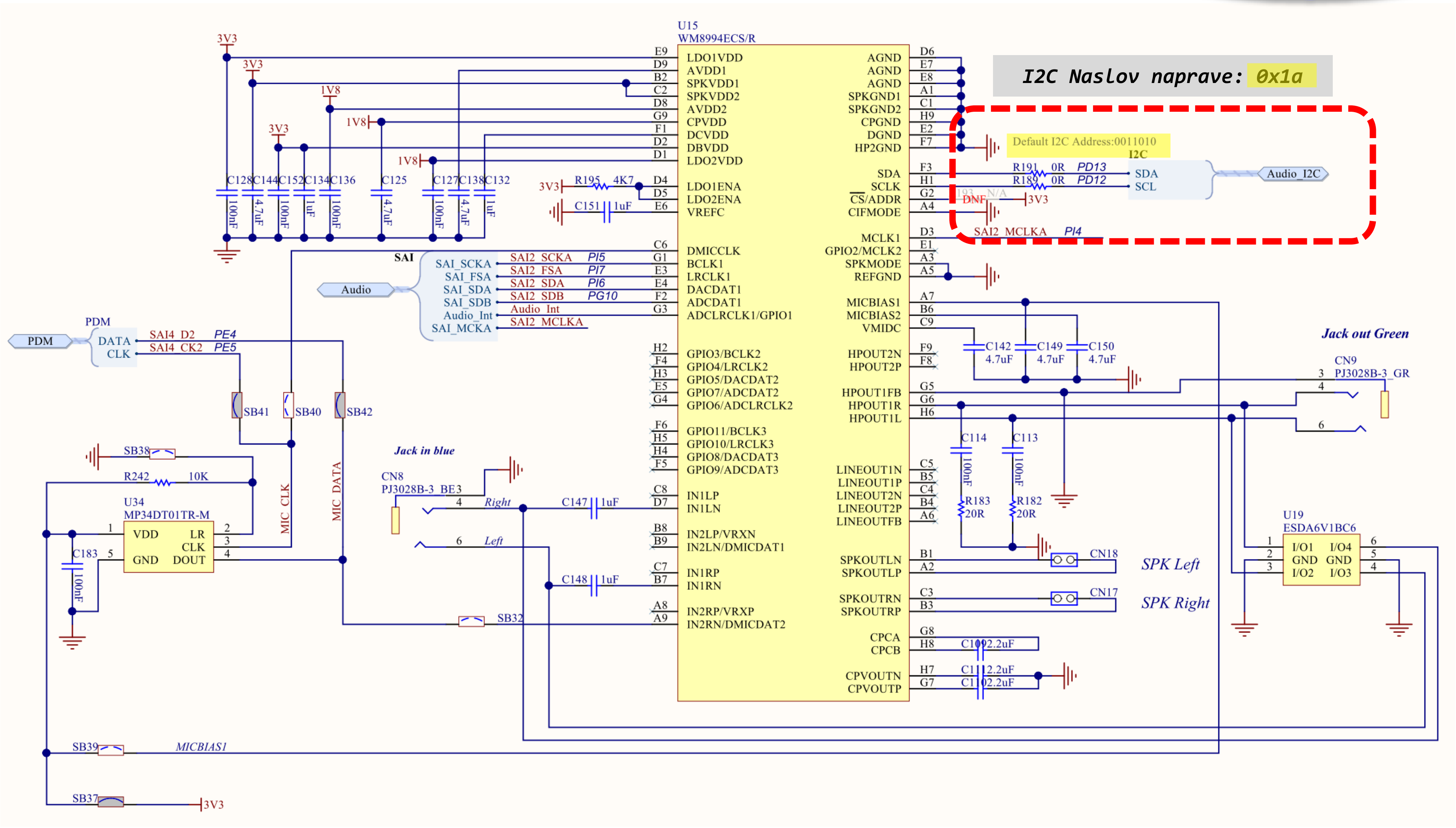
The WM8994 control registers are listed below. Note that only the register addresses described here should be accessed; writing to other addresses may result in undefined behaviour. Register bits that are not documented should not be changed from the default values.

REG	NAME	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	DEFAULT
R0 (0h)	Software Reset	SW_RESET [15:0]																0000h
R1 (1h)	Power Management (1)	0	0	SPKOUTR_ENA	SPKOUTL_ENA	HPOUT2_ENA	0	HPOUT1L_ENA	HPOUT1R_ENA	0	0	MICB2_ENA	MICB1_ENA	0	VMID_SEL [1:0]		BIAS_ENA	0000h
R2 (2h)	Power Management (2)	0	TSHUT_ENA	TSHUT_OPDIS	0	OPCLK_ENA	0	MIXINL_ENA	MIXINR_ENA	IN2L_ENA	IN1L_ENA	IN2R_ENA	IN1R_ENA	0	0	0	0	6000h

STM32H7

VP 4 - STM32H7 CubeIDE, I2C Audio WM9884 – vezalna shema

STM32H7



I2C Naslov naprave: 0x1a

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_TS_Demo

VP 4 - STM32H7 CubeIDE, I2C4 branje

Spremenljivke

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
charSendBuffer[BUFSIZE];
intCounter;
int KeyState=0;
uint8_t dataBuffer[10];

HAL_StatusTypeDef retval;
/* USER CODE END PV */
```

main.c : dodana koda

```
// Reading from address 0x1a register R0 (addr. 0x00) default value should be 0x8994
   dataBuffer[0] = 0; dataBuffer[1] = 0x00;
   retval = HAL_I2C_Master_Transmit(&hi2c4, (0x1a << 1), dataBuffer, 2, HAL_MAX_DELAY);
   retval = HAL_I2C_Master_Receive(&hi2c4, (0x1a << 1), dataBuffer, 2, HAL_MAX_DELAY);

   snprintf(SendBuffer,BUFSIZE,"Hello World [%d]: Key:%d
Reg.value1:0x%\n\r",Counter++,KeyState, dataBuffer[0]*256+dataBuffer[1]);

   HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);
```

Glavna zanka

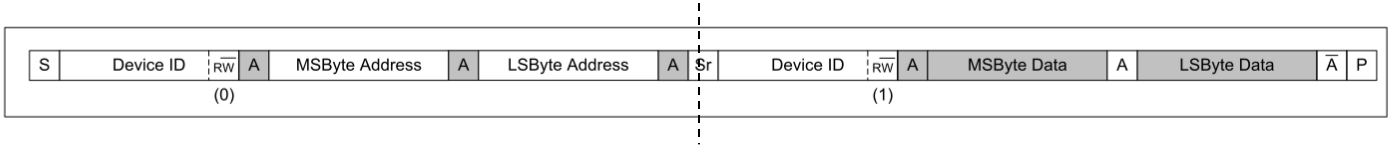
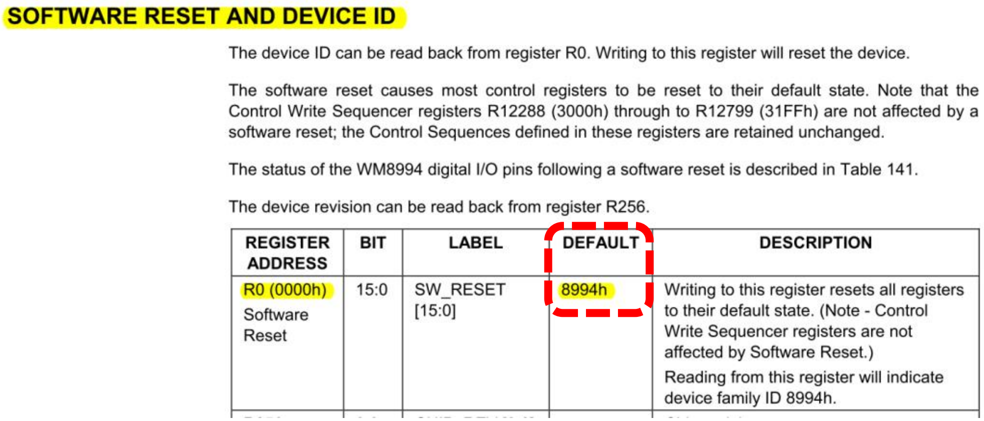


Figure 75 Single Register Read from Specified Address



SOFTWARE RESET AND DEVICE ID

The device ID can be read back from register R0. Writing to this register will reset the device.

The software reset causes most control registers to be reset to their default state. Note that the Control Write Sequencer registers R12288 (3000h) through to R12799 (31FFh) are not affected by a software reset; the Control Sequences defined in these registers are retained unchanged.

The status of the WM8994 digital I/O pins following a software reset is described in Table 141.

The device revision can be read back from register R256.

REGISTER ADDRESS	BIT	LABEL	DEFAULT	DESCRIPTION
R0 (0000h) Software Reset	15:0	SW_RESET [15:0]	8994h	Writing to this register resets all registers to their default state. (Note - Control Write Sequencer registers are not affected by Software Reset.) Reading from this register will indicate device family ID 8994h.

STM32H7

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_TS_Demo

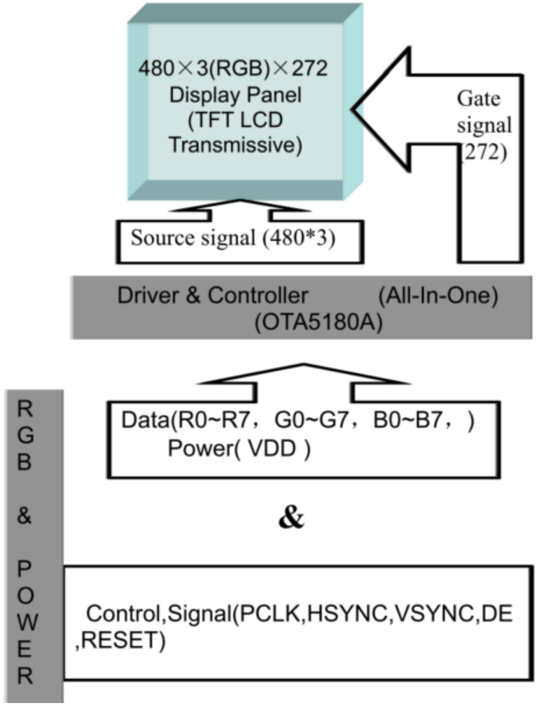
VP 4 - STM32H7 CubeIDE, I2C LCD-Touch RK043FN48H

8-bitni naslovi in 8-bitni registri !

Module P/N: RK043FN48H-CT672B

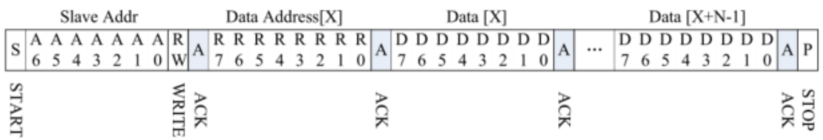
Version: 1.0

Description : 4.3 inch TFT 480*272 Pixels with LED Backlight and capacitive touch Panel

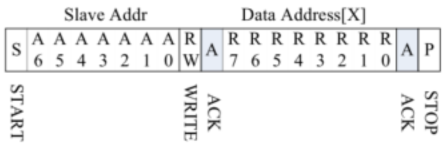


1.2 I²C Read/Write Interface description

Write N bytes to I2C slave



Set Data Address

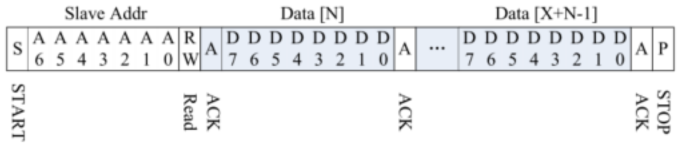


2.1.26 ID_G_FT5201ID

This register describes vendor's chip id

Address	Bit Address	Register Name	Description
A8h	7:0	ID_G_FT5201ID	R: xx

Read X bytes from I²C Slave



2.1 Work Mode

In this mode the CTP is fully functional as a touch screen controller. Read and write access address is ju logical address which is not enforced by hardware or firmware. Here is the operating mode register map.

Work Mode Register Map

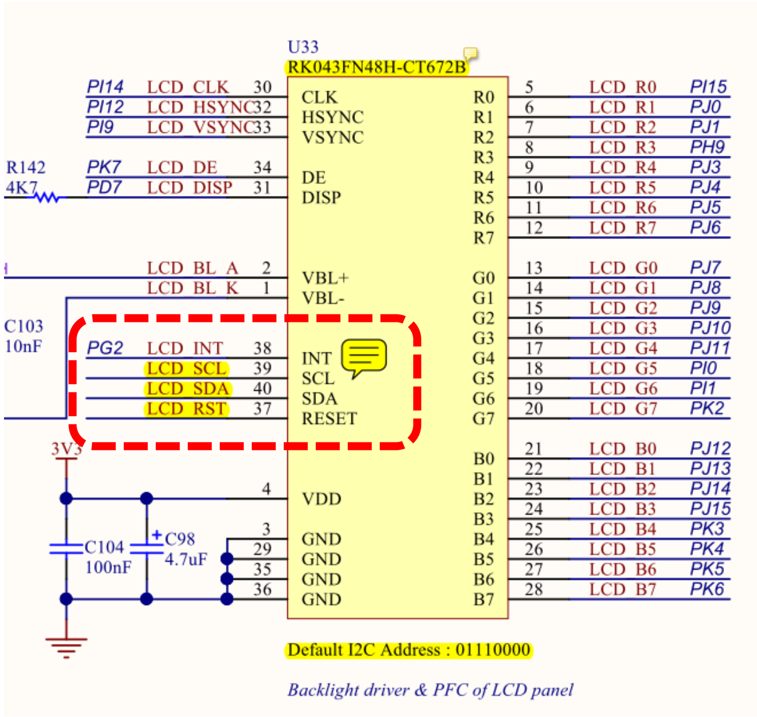
Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access
00h	DEVIDE_MODE		Device Mode[2:0]							RW

Device Mode	Val	Description
Work	000b	Read touch point and gesture
Factory	100b	Read raw data

STM32H7

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Basic_Demo

VP 4 - STM32H7 CubeIDE, I2C LCD-Touch RK043FN48H – vezalna shema



I2C Naslov naprave: 0x38 (ali 0x70 s pomikom na levo – sprostimo prostor za R/W bit)

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Basic_Demo

VP 4 - STM32H7 CubeIDE, I2C4 branje

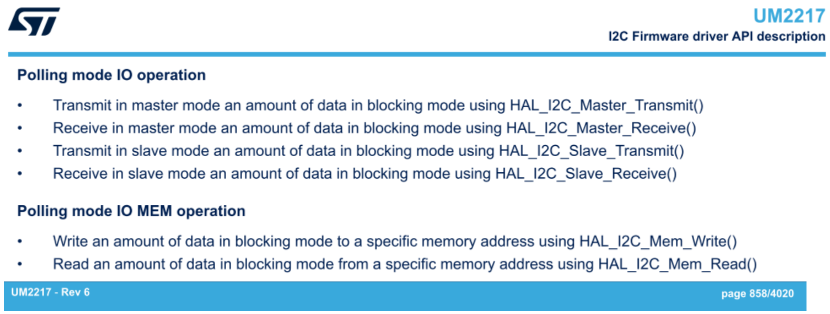
main.c : dodana koda

Spremenljivke

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
charSendBuffer[BUFSIZE];
intCounter;
int KeyState=0;
uint8_t dataBuffer[10];

HAL_StatusTypeDef retval;
/* USER CODE END PV */
```

```
 // Reading from address 0x38 register Vendor's Chip ID (addr. 0xA8) default value should be 0x51=81  - Both variations work !
    //dataBuffer[5] = 0xA8;
    //retval = HAL_I2C_Master_Transmit(&hi2c4, (0x38 << 1), &dataBuffer[5], 1, HAL_MAX_DELAY);
    //retval = HAL_I2C_Master_Receive(&hi2c4, (0x38 << 1), &dataBuffer[5], 1, HAL_MAX_DELAY);
    retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0xA8, I2C_MEMADD_SIZE_8BIT,&dataBuffer[5], 1, HAL_MAX_DELAY);


    snprintf(SendBuffer,BUFSIZE,"Hello World [%d]: Key:%d Audio Chip ID: 0x%4x Touch ID: 0x%2x=%d\n\r",Counter++,KeyState,
dataBuffer[0]*256+dataBuffer[1],dataBuffer[5],dataBuffer[5]);
    HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),100);

    HAL_Delay(1000);
```

Glavna zanka



UM2217

I2C Firmware driver API description

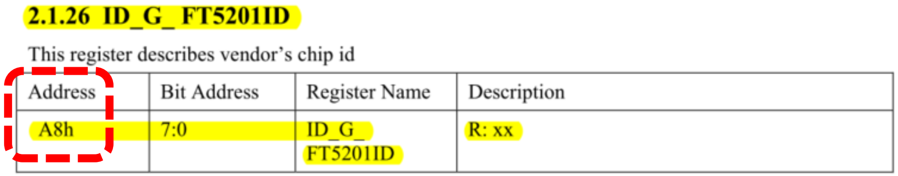
Polling mode IO operation

- Transmit in master mode an amount of data in blocking mode using HAL_I2C_Master_Transmit()
- Receive in master mode an amount of data in blocking mode using HAL_I2C_Master_Receive()
- Transmit in slave mode an amount of data in blocking mode using HAL_I2C_Slave_Transmit()
- Receive in slave mode an amount of data in blocking mode using HAL_I2C_Slave_Receive()

Polling mode IO MEM operation

- Write an amount of data in blocking mode to a specific memory address using HAL_I2C_Mem_Write()
- Read an amount of data in blocking mode from a specific memory address using HAL_I2C_Mem_Read()

UM2217 - Rev 6 page 858/4020



2.1.26 ID_G_ FT5201ID

This register describes vendor's chip id

Address	Bit Address	Register Name	Description
A8h	7:0	ID_G_ FT5201ID	R: xx

STM32H7

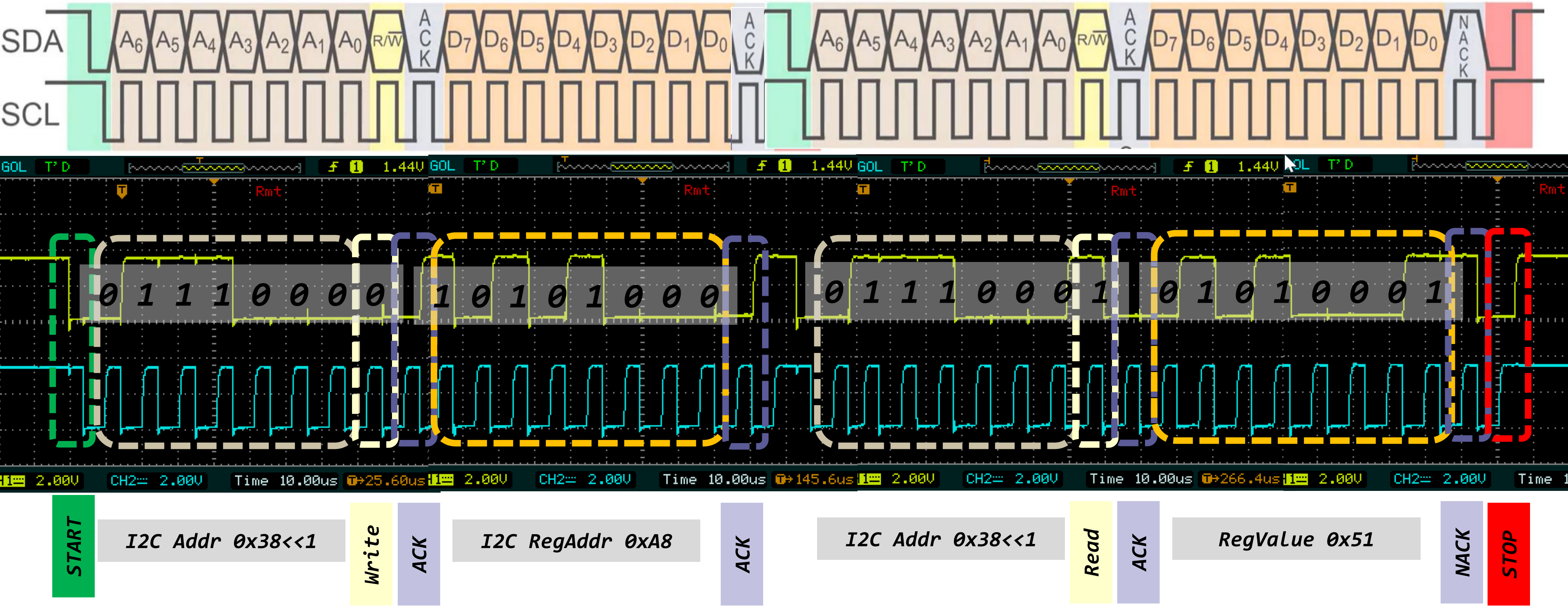
https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Basic_Demo

VP 4 - STM32H7 CubeIDE, I2C4 branje

I2C branje

main.c : dodana koda

```
// Reading from address 0x38 register Vendor's Chip ID (addr. 0xA8) default value should be 0x51=81

retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0xA8, I2C_MEMADD_SIZE_8BIT,&dataBuffer[5], 1, HAL_MAX_DELAY);
```



https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Basic_Demo

Primer I2C komunikacije STM32H7 - Touch

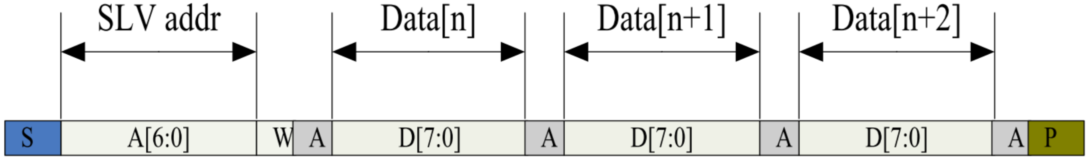


Figure 2-5 I2C master write, slave read

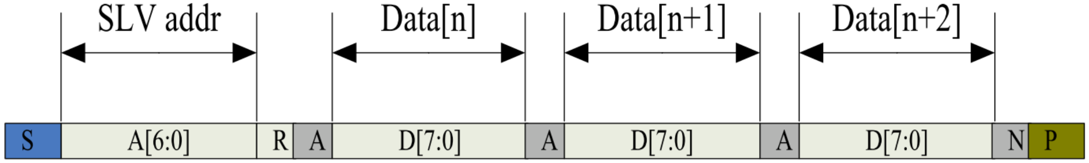


Figure 2-6 I2C master read, slave write

FT5336GQQ

True Multi-Touch Capacitive Touch Panel Controller

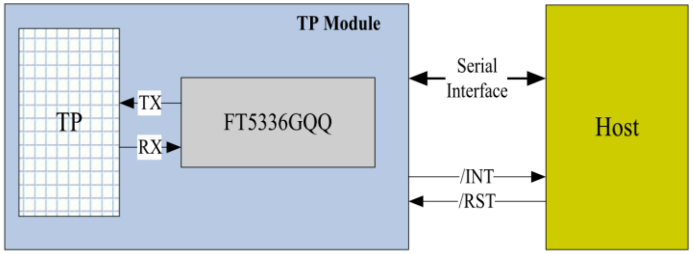
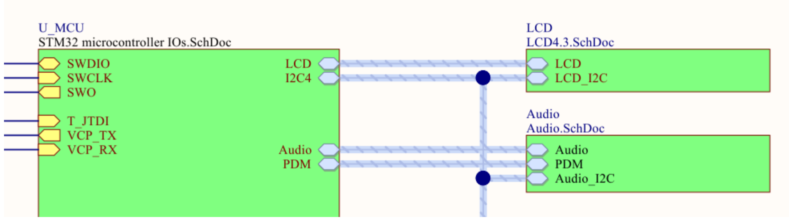
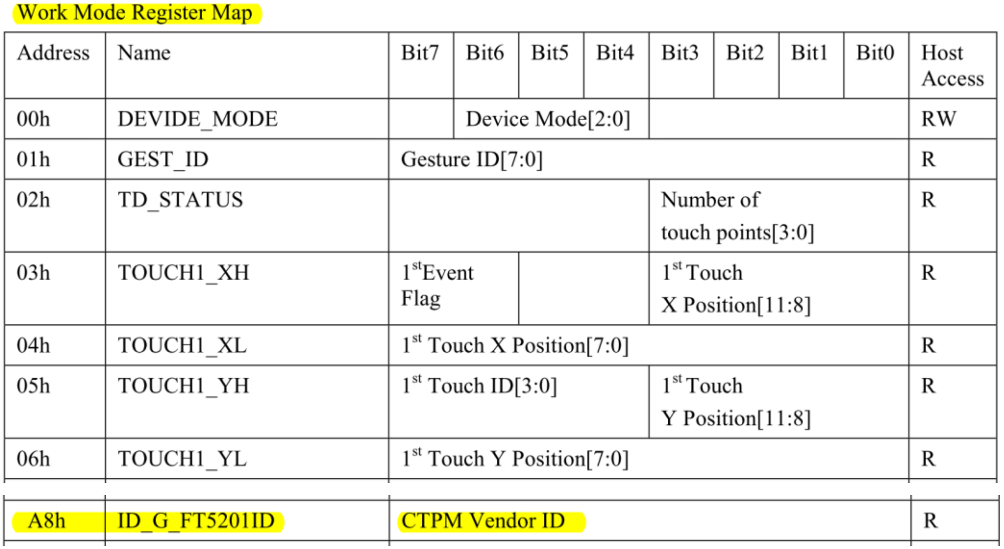


Figure 2-3 Host Interface Diagram

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Touch_Demo

8-bitni naslovi in registri





Work Mode Register Map

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access
00h	DEVIDE_MODE		Device Mode[2:0]							RW
01h	GEST_ID	Gesture ID[7:0]								R
02h	TD_STATUS					Number of touch points[3:0]				R
03h	TOUCH1_XH	1^{st}Event Flag				1^{st} Touch X Position[11:8]				R
04h	TOUCH1_XL	1^{st} Touch X Position[7:0]								R
05h	TOUCH1_YH	1^{st} Touch ID[3:0]				1^{st} Touch Y Position[11:8]				R
06h	TOUCH1_YL	1^{st} Touch Y Position[7:0]								R
A8h	ID_G_FT5201ID	CTPM Vendor ID								R

STM32H7

Primer I2C komunikacije STM32H7 - Touch

STM32H7

```
// Reading from address 0x38 register Vendor's Chip ID (addr. 0xA8) default value should be 0x51=81

retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0xA8, I2C_MEMADD_SIZE_8BIT,&VendorID, 1, HAL_MAX_DELAY);

retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0x00, I2C_MEMADD_SIZE_8BIT,&DeviceMode, 1, HAL_MAX_DELAY);
retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0x01, I2C_MEMADD_SIZE_8BIT,&Gesture, 1, HAL_MAX_DELAY);
retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0x02, I2C_MEMADD_SIZE_8BIT,&Status, 1, HAL_MAX_DELAY);

retval = HAL_I2C_Mem_Read(&hi2c4, (0x38 << 1), 0x03, I2C_MEMADD_SIZE_8BIT,&dataBuffer, 4, HAL_MAX_DELAY);
if (Status != 0) {
        TouchX = ( (dataBuffer[0] & 0b1111) << 8) + dataBuffer[1];
        TouchY = ( (dataBuffer[2] & 0b1111) << 8) + dataBuffer[3];
} else {
        TouchX = 0;
        TouchY = 0;
}
```

8-bitni naslovi in registri

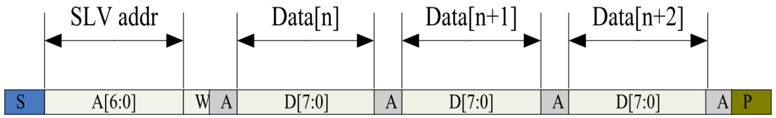


Figure 2-5 I2C master write, slave read

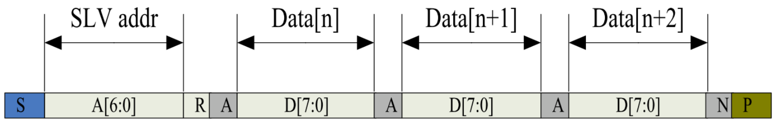
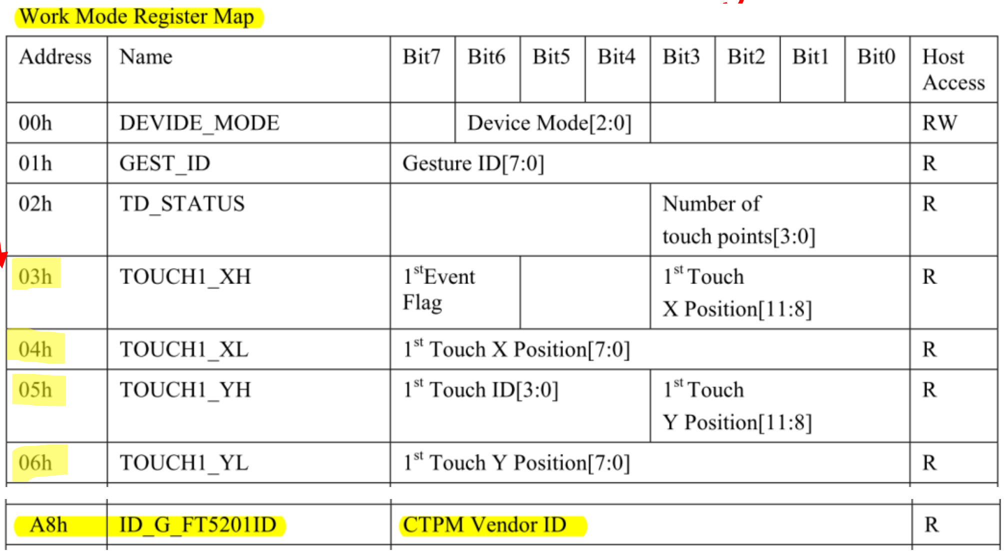


Figure 2-6 I2C master read, slave write



Work Mode Register Map

Address	Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0	Host Access
00h	DEVIDE_MODE		Device Mode[2:0]							RW
01h	GEST_ID	Gesture ID[7:0]								R
02h	TD_STATUS					Number of touch points[3:0]				R
03h	TOUCH1_XH	1stEvent Flag				1st Touch X Position[11:8]				R
04h	TOUCH1_XL	1st Touch X Position[7:0]								R
05h	TOUCH1_YH	1st Touch ID[3:0]				1st Touch Y Position[11:8]				R
06h	TOUCH1_YL	1st Touch Y Position[7:0]								R
A8h	ID_G_FT5201ID	CTPM Vendor ID								R

https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_I2C_Touch_Demo

VIN projekt - VP4: STM32-Edge computing, CubeIDE primeri, Miško3

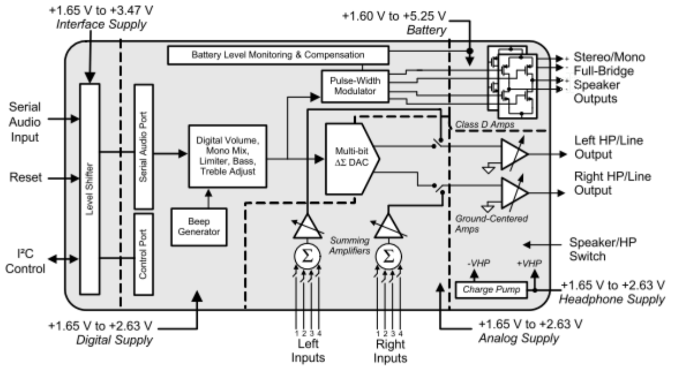
- VIN projekt
- AI v vgrajenih napravah („Edge Computing“)
- Miško3 – demo projekt
- STM32 CubeIDE H7,F4 – PWM izhodi
- STM32H7 CubeIDE, I2C (Scan, WM9884, Touch)
- STM32F4 CubeIDE: I2C in CS43L22

VP 4 - STM32 CubeIDE, I2C in CS43L22 Gradiva

CIRRUS LOGIC®

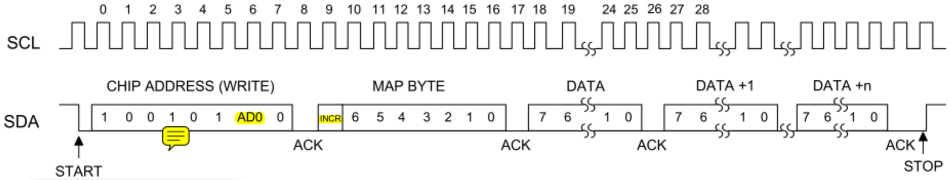
CS43L22

Low Power, Stereo DAC w/Headphone & Speaker Amps



5.1 I²C Control

The upper 6 bits of the address field are fixed at 100101. To communicate with the CS43L22, the chip address field, which is the first byte sent to the CS43L22, should match 100101 followed by the setting of the AD0 pin. The eighth bit of the address is the R/W bit. If the operation is a write, the next byte is the Memory Address Pointer (MAP), which selects the register to be read or written. If the operation is a read, the contents of the register pointed to by the MAP will be output. Setting the auto-increment bit in MAP allows successive reads or writes of consecutive registers. Each byte is separated by an acknowledge bit. The ACK bit is output from the CS43L22 after each input byte is read and is input to the CS43L22 from the microcontroller after each transmitted byte.



rozman 26. 04. 2022, 1...

AD0 -> GND Addr=0x94

Figure 16. Control Port Timing, I²C Write

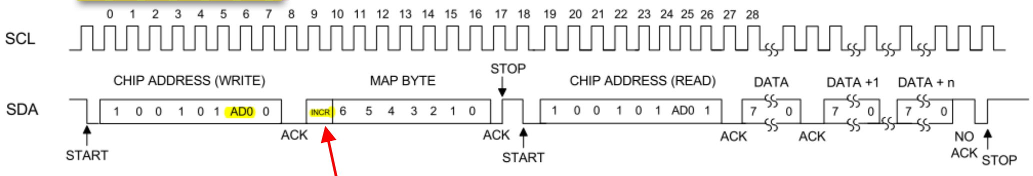


Figure 17. Control Port Timing, I²C Read

5.1.1 *Memory Address Pointer (MAP)*

The MAP byte comes after the address byte and selects the register to be read or written. Refer to the pseudo code above for implementation details.

5.1.1.1 *Map Increment (INCR)*

The device has MAP auto-increment capability enabled by the INCR bit (the MSB) of the MAP. If INCR is set to 0, MAP will stay constant for successive I²C writes or reads. If INCR is set to 1, MAP will auto-increment after each byte is read or written, allowing block reads or writes of successive registers.

7. REGISTER DESCRIPTION

All registers are read/write except for the chip I.D. and Revision Register and Interrupt Status Register which are read only. See the following bit definition tables for bit assignment information. The default state of each bit after a power-up sequence or reset is shown as shaded in the table. Unless otherwise specified, all "Reserved" bits must maintain their default value.

7.1 Chip I.D. and Revision Register (Address 01h) *(Read Only)*

7	6	5	4	3	2	1	0
CHIPID4	CHIPID3	CHIPID2	CHIPID1	CHIPID0	REVID2	REVID1	REVID0

7.1.1 *Chip I.D. (Read Only)*

I.D. code for the CS43L22.

CHIPID[4:0]	Device
11100	CS43L22

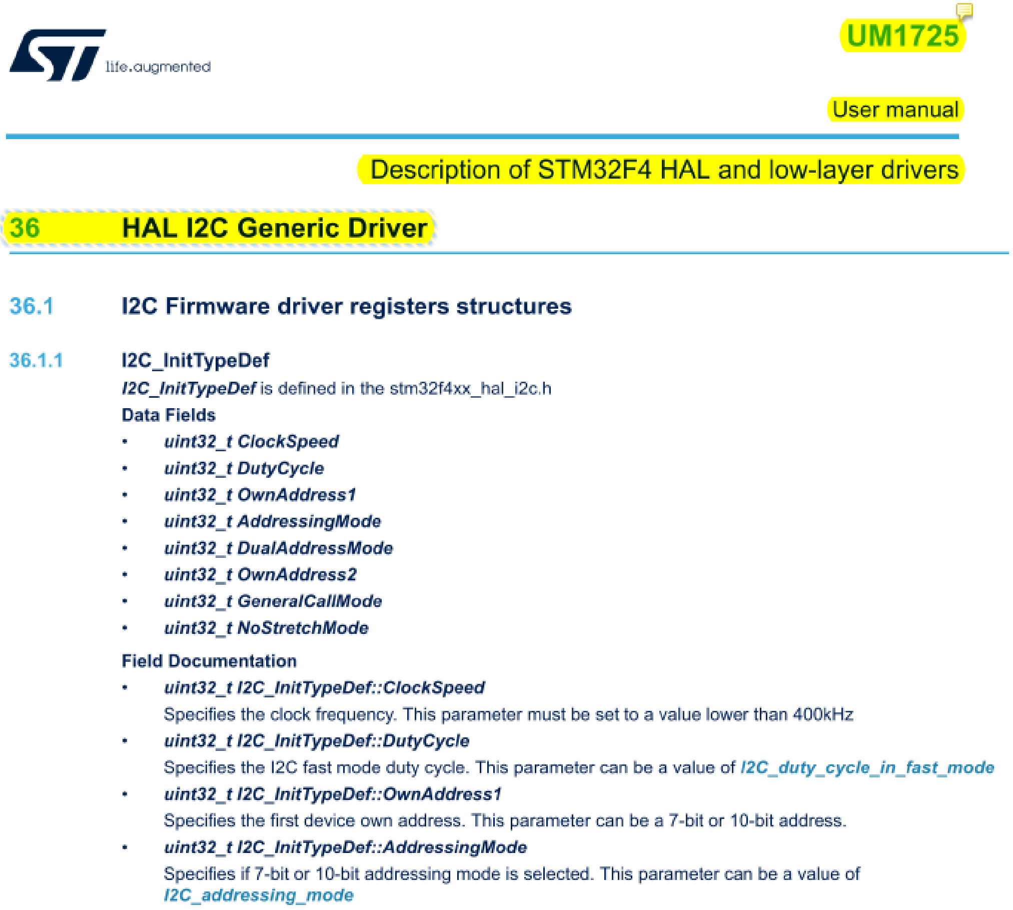
7.1.2 *Chip Revision (Read Only)*

CS43L22 revision level.

REVID[2:0]	Revision Level
000	A0
001	A1
010	B0
011	B1

https://github.com/LAPSyLAB/STM32F4_Docs_and_Examples/blob/main/STM32F407_Discovery_kit/CS43L22_F2.pdf

Delo na STM32F4 razvojnem sistemu



life.augmented

UM1725

User manual

Description of STM32F4 HAL and low-layer drivers

36 HAL I2C Generic Driver

36.1 I2C Firmware driver registers structures

36.1.1 I2C_InitTypeDef

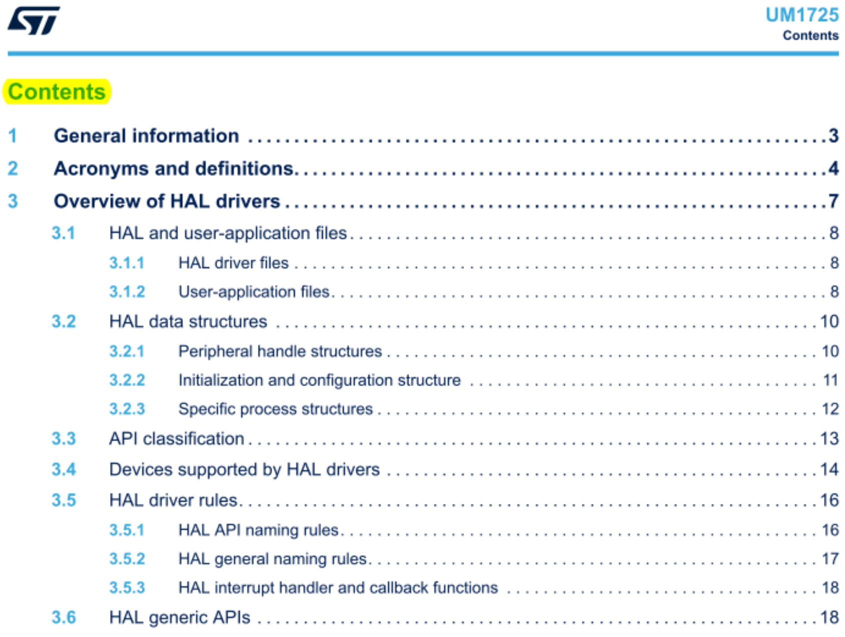
I2C_InitTypeDef is defined in the stm32f4xx_hal_i2c.h

Data Fields

- ***uint32_t ClockSpeed***
- ***uint32_t DutyCycle***
- ***uint32_t OwnAddress1***
- ***uint32_t AddressingMode***
- ***uint32_t DualAddressMode***
- ***uint32_t OwnAddress2***
- ***uint32_t GeneralCallMode***
- ***uint32_t NoStretchMode***

Field Documentation

- ***uint32_t I2C_InitTypeDef::ClockSpeed***
  Specifies the clock frequency. This parameter must be set to a value lower than 400kHz
- ***uint32_t I2C_InitTypeDef::DutyCycle***
  Specifies the I2C fast mode duty cycle. This parameter can be a value of ***I2C_duty_cycle_in_fast_mode***
- ***uint32_t I2C_InitTypeDef::OwnAddress1***
  Specifies the first device own address. This parameter can be a 7-bit or 10-bit address.
- ***uint32_t I2C_InitTypeDef::AddressingMode***
  Specifies if 7-bit or 10-bit addressing mode is selected. This parameter can be a value of ***I2C_addressing_mode***



UM1725
Contents

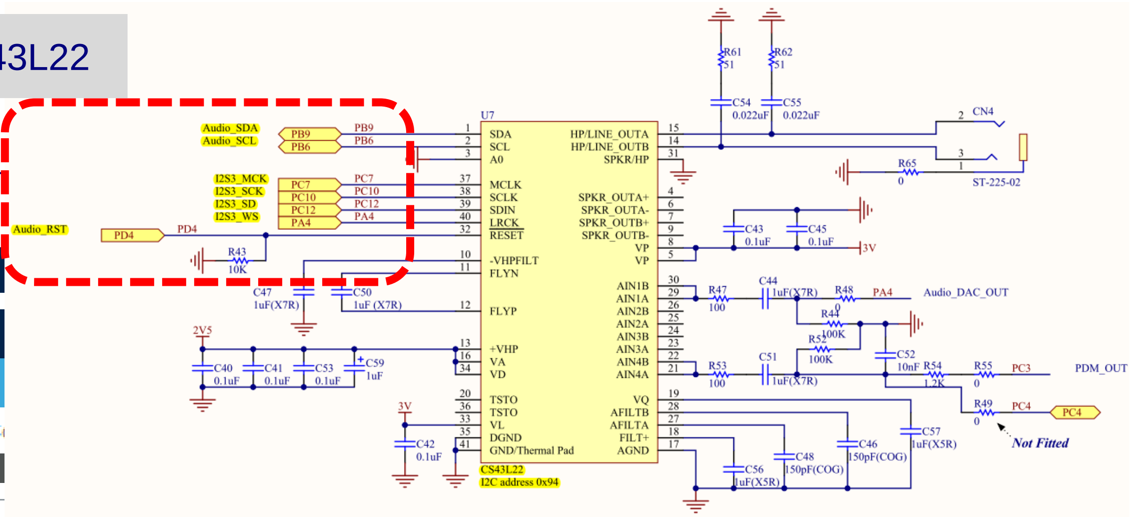
Contents

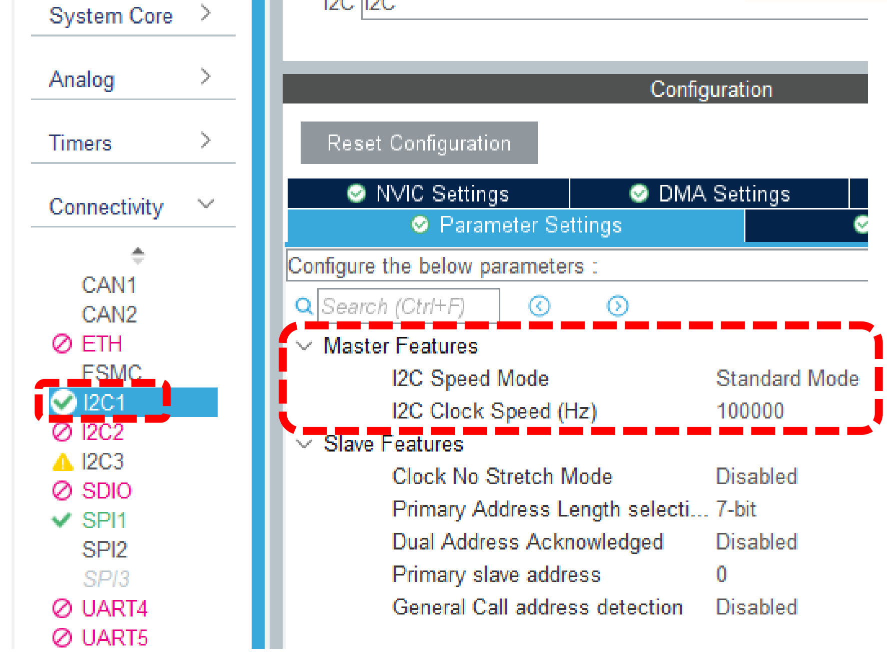
Lastni viri :

https://github.com/LAPSyLAB/STM32F4_Docs_and_Examples

VP 4 - STM32 CubeIDE, I2C in CS43L22

CubeMX nastavitve (I2C1 že nastavljen)





STM32_I2C_CS43L22_Basic.ioc - Pinout & Configuration

Pinout & Configuration

Categories | A->Z

- System Core >
- Analog >
- Timers >
- Connectivity ⌄
  - CAN1
  - CAN2
  - ⊘ ETH
  - FSMC
  - ✓ I2C1
  - ⊘ I2C2
  - ⚠ I2C3
  - ⊘ SDIO
  - ✓ SPI1
  - SPI2
  - SPI3
  - ⊘ UART4
  - ⊘ UART5

I2C1 Mode and Co

Mode

I2C I2C

Configuration

Reset Configuration

NVIC Settings | DMA Settings | Parameter Settings

Configure the below parameters :

Search (Ctrl+F)

⌄ Master Features

I2C Speed Mode	Standard Mode
I2C Clock Speed (Hz)	100000

⌄ Slave Features

Clock No Stretch Mode	Disabled
Primary Address Length selecti...	7-bit
Dual Address Acknowledged	Disabled
Primary slave address	0
General Call address detection	Disabled

i2c.c:

```c
/* I2C1 init function */
void MX_I2C1_Init(void)
{
  /* USER CODE BEGIN I2C1_Init 0 */
  /* USER CODE END I2C1_Init 0 */

  /* USER CODE BEGIN I2C1_Init 1 */
  /* USER CODE END I2C1_Init 1 */

  hi2c1.Instance = I2C1;
  hi2c1.Init.ClockSpeed = 100000;
  hi2c1.Init.DutyCycle = I2C_DUTYCYCLE_2;
  hi2c1.Init.OwnAddress1 = 0;
  hi2c1.Init.AddressingMode = I2C_ADDRESSINGMODE_7BIT;
  hi2c1.Init.DualAddressMode = I2C_DUALADDRESS_DISABLE;
  hi2c1.Init.OwnAddress2 = 0;
  hi2c1.Init.GeneralCallMode = I2C_GENERALCALL_DISABLE;
  hi2c1.Init.NoStretchMode = I2C_NOSTRETCH_DISABLE;
  if (HAL_I2C_Init(&hi2c1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN I2C1_Init 2 */

  /* USER CODE END I2C1_Init 2 */

}
```

VP 4 - STM32 CubeIDE, I2C in CS43L22

main.c : dodana koda

Spremenljivke

```
/* USER CODE BEGIN PV */
#define BUFSIZE 256
charSendBuffer[BUFSIZE];
intCounter;
int KeyState=0;

HAL_StatusTypeDef retval;
uint8_t ChipID;
/* USER CODE END PV */
```

Inicializacija

```
/* USER CODE BEGIN 2 */

HAL_GPIO_WritePin(GPIOD, GPIO_PIN_4,GPIO_PIN_SET);   // Set Reset line to 1 (switch device on)

HAL_Delay(1000);   // recomended by datasheet

// From Device with address=0x94, Read register with address 0x01 and put value in ChipID
// DevAddress_0x94, tMemAddress=0x01, MemAddSize=8b, *pData,Size, Timeout);
retval = HAL_I2C_Mem_Read(&hi2c1, 0x94, 0x01, I2C_MEMADD_SIZE_8BIT, &ChipID, 1, 1000);

/* USER CODE END 2 */
```

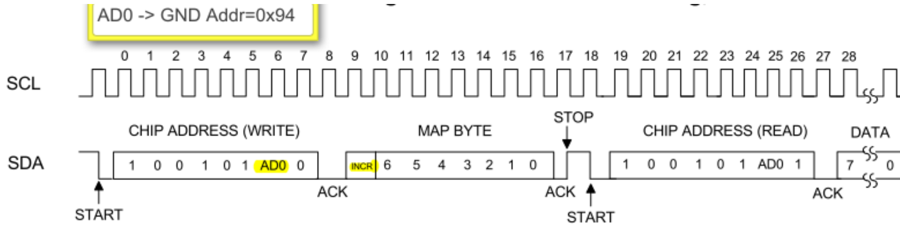


Figure 17. Control Port Timing, I²C Read

Glavna zanka

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

 HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);
 HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
 HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_14);

 KeyState = HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0);
 HAL_GPIO_WritePin(GPIOD, GPIO_PIN_15, KeyState);


  snprintf(SendBuffer,BUFSIZE,"Hello World [%d]: Key:%d | Id:%02x \r\n",Counter++,KeyState,ChipID);
  CDC_Transmit_FS(SendBuffer,strlen(SendBuffer));

    /* USER CODE END WHILE */

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

Primer kompleksnejše demo USB-Audio aplikacije :

"Wave player - Predvajalnik .wav datotek iz USB ključka na izhod za slušalke"

WAVEPLAYER using STM32 || I2S AUDIO || CS43L22 || F4 DISCOVERY

From <https://www.youtube.com/watch?v=_Pm0L1ropJs>

AN3997 Application note Audio playback and recording using the STM32F4DISCOVERY

https://www.st.com/resource/en/application_note/an3997-audio-playback-and-recording-using-the-stm32f4discovery-stmicroelectronics.pdf

WavePlayer using STM32 Discovery

From <https://controllerstech.com/waveplayer-using-stm32-discovery/>

VIN projekt - VP5: STM32-Edge computing, CubeIDE projekti, Miško3

- Diskusija, vprašanja ?