ARM

Projekt za STM32H7 vgrajen sistem II. del

CubeIDE

STM32H750B-DK Discovery razvojni sistem

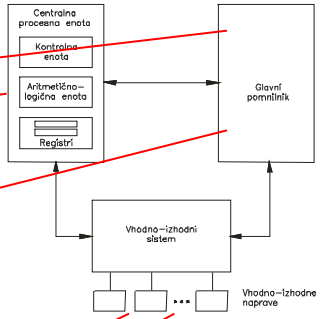
I. del

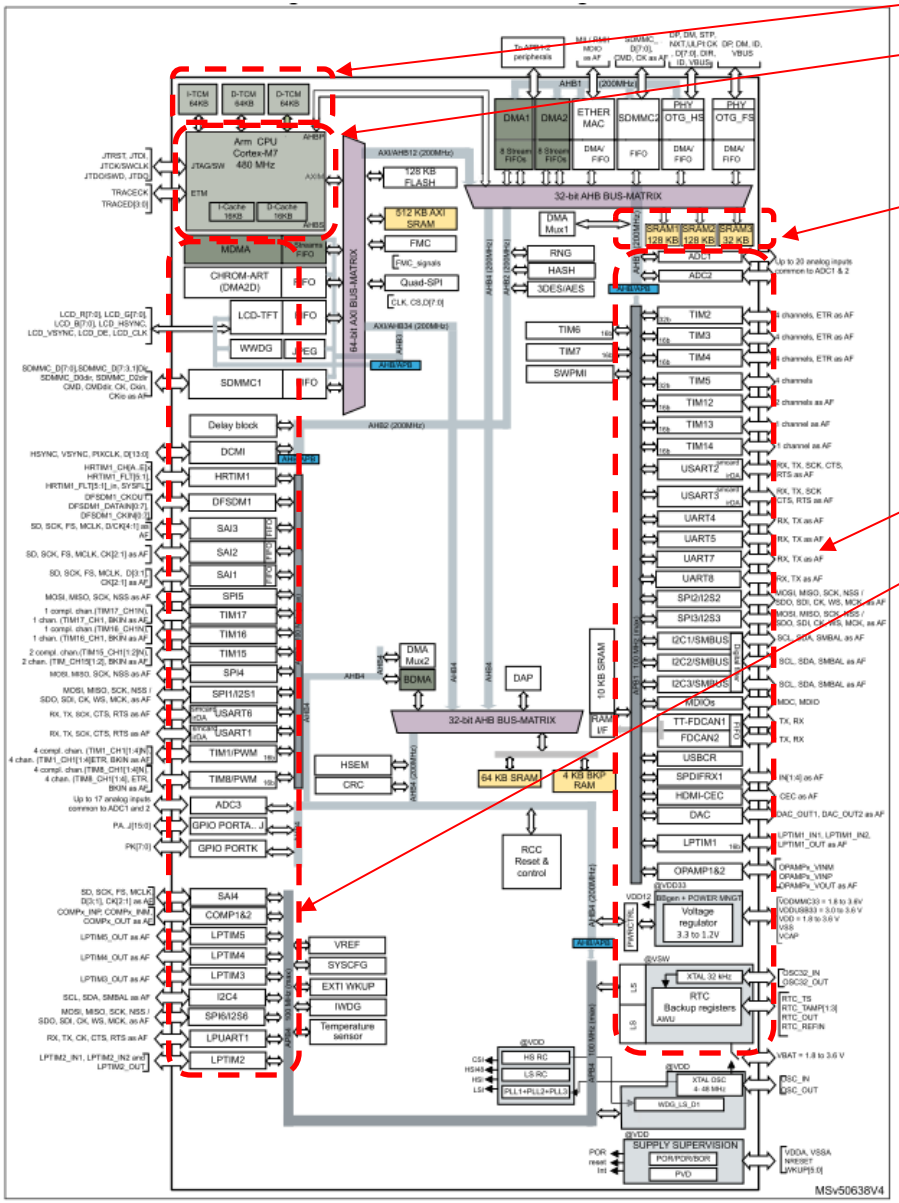
- Arm® Cortex® core-based microcontroller with 128 Kbytes (STM32H750XBH6) of Flash memory and 1 Mbyte of RAM, in TFBGA240+25 package
- 4.3” RGB interface LCD with touch panel connector
- Ethernet compliant with IEEE-802.3-2002, and POE
- USB OTG FS with Micro-AB connector
- SAI audio codec
- One ST-MEMS digital microphone
- 2 x 512-Mbit Quad-SPI NOR Flash memory
- 128-Mbit SDRAM
- 4-Gbyte on-board eMMC
- 1 user and reset push-button
- Fanout daughterboard
- 2 x FDCANs
- Board connectors:
  - USB FS Micro-AB connectors
  - ST-LINK Micro-B USB connector
  - USB power Micro-B connector
  - Ethernet RJ45
  - Stereo headset jack including analog microphone input
  - Audio header for external speakers
  - Arduino™ Uno V3 expansion connectors
  - STMod+

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

STM32H750XB

I. del





Delo na STM32H7 razvojnem sistemu

I. del

Priključitev :

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

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

- ***dodajanje vsebine (Main.s):***

CubeIDEWorkspace - stm32h7-asm/Core/Src/Main.s - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer

- CubeIDE_Workspace
  - stm32f4-asm-qemu
- Delo
  - ARM9Template
  - stm32f4-asm (in STM32AsmTemplate)
  - ARM9Template.zip
- Node_V4 (in node_v4)
- Sluzba
  - CAN_IEX_Module
  - ORLab-STM32H7
    - stm32h7-asm
      - Binaries
      - Includes
      - Core
        - Src
          - Main.s
        - Startup
          - startup_stm32h750xbhx.s
      - Debug
      - out
      - makefile
      - README.md
      - STM32H750X.svd
      - STM32H750XBHX_FLASH.ld
      - STM32H750XBHX_RAM.ld
    - README.md
  - RALab-STM32H7
    - stm32h7-asm_RA_LED
    - README.md
  - STM32_USB_Key_AdvDebug
  - STM32_USB_Key_FreeRTOS_AdvDebug
  - STM32CubeIDE_Adv_Debug
  - STM32F4_Discovery_VIN_Projects

Main.s × startup_stm32h750xbhx.s

```
12
13 ////////////////////////////////////////////////////////////////////////////
14 // Definitions
15 ////////////////////////////////////////////////////////////////////////////
16 // Definitions section. Define all the registers and
17 // constants here for code readability.
18
19 // Constants
20
21
22 // Start of data section
23         .data
24
25         .align
26
27 STEV1:  .word   0x10    // 32-bitna spr.
28 STEV2:  .word   0x40    // 32-bitna spr.
29 VSOTA:  .word   0       // 32-bitna spr.
30
31
32 // Start of text section
33         .text
34
35         .type  main, %function
36         .global main
37
38         .align
39 main:
40         ldr r0, =STEV1   // Naslov od STEV1 -> r0
41         ldr r1, [r0]     // Vsebina iz naslova v r0 -> r1
42
43         ldr r0, =STEV2   // Naslov od STEV1 -> r0
44         ldr r2, [r0]     // Vsebina iz naslova v r0 -> r2
45
46         add r3,r1,r2     // r1 + r2 -> r3
47
48         ldr r0, =VSOTA   // Naslov od STEV1 -> r0
49         str r3,[r0]      // iz registra r3 -> na naslov v r0
50
51 __end:  b    __end
52
```

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

- STM32H750B-DK Discovery kit with STM32H750XB MCU
- ORLab-STM32H7 - GitHub repozitorij
- User Manual Discovery kit stm32h750xb Uploaded 11/11/22, 10.15
- DataSheet_stm32h750xb Uploaded 11/11/22, 10.16
- Reference Manual rm0433-stm32h750xb Uploaded 11/11/22, 10.17
- Programming_Manual_pm0253-stm32h750xb Uploaded 11/11/22, 10.17
- Errata_es0396-stm32h750xb Uploaded 11/11/22, 10.19

Delo na STM32H7 razvojnem sistemu

I. del

Priključitev :

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

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

- ***začetne nastavitve*** (*startup_stm32h750xbhx.s*) :
  - Pustimo default nastavitve:
    - 64MHz frekvenca urinega signala
      - (višja poveča porabo!)
    - izklop predpomnilnikov
  - inicializacija sklada oz. SP – kazalca na sklad

- ***dodajanje vsebine (Main.s):***

  - *podatki/operandi:*
    - dodamo v .data sekcijo, končamo z .align

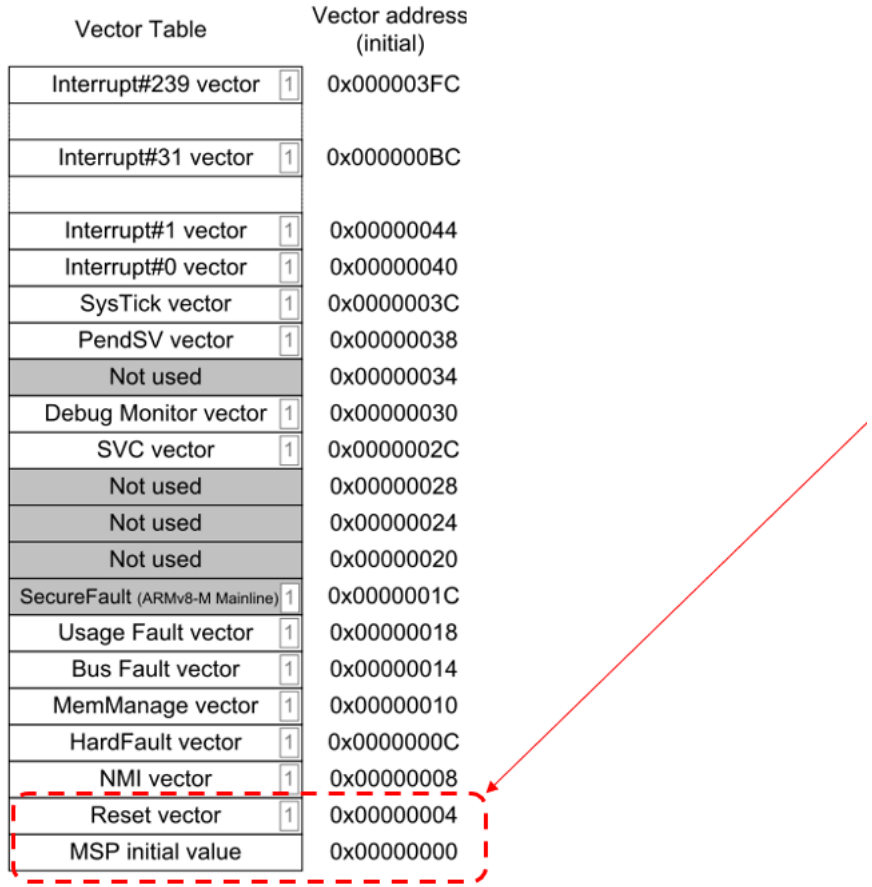
  - *program* (dodamo v .text sekcijo) :
    - dodamo od oznake `main:` naprej
    - na koncu programa je mrtva zanka (`__end: b __end`)
    - podprograme dodamo za mrtvo zanko

Inicializacija sistema – začetno stanje

startup_stm32h750xbhx.s :

```
g_pfnVectors:
  .word  _estack
  .word  Reset_Handler
  .word  NMI_Handler
  .word  HardFault_Handler
  .word  MemManage_Handler
  .word  BusFault_Handler
  .word  UsageFault_Handler
  .word  0
  .word  0
  .word  0
  .word  0
  .word  SVC_Handler
  .word  DebugMon_Handler
  .word  0
  .word  PendSV_Handler
  .word  SysTick_Handler
```

ARM Cortex M – Vektorska tabela



Inicializacija sistema – začetno stanje

I. del

startup_stm32h750xbhx.s :

```
Reset_Handler:
  ldr   sp, =_estack      /* set stack pointer */

/* Copy the data segment initializers from flash to SRAM */
CopyDataInit: …

FillZerobss: …

// Initialize DWT counters - added for cycle measurements
     …

/* Call the application's entry point.*/
  bl  main
  bx  lr
```

Povezovalna skripta („linker script“)

II. Del
(v nadaljevanju)

Introduction to the STM32 Memory Model

The memory of STM32 microcontrollers has two main areas :

1. Flash Memory :

- Read-Only
- Begins at address 0x08000000
- Size depends on specific STM32 microcontroller
- Program code is stored here.
- Contains a vector table at address 0x08000004

2. SRAM Memory :

- Read and Write
- Begins at 0x20000000
- Size depends on specific STM32 microcontroller
- Variables and Stack are stored here

```
MEMORY
{
  FLASH(rx):ORIGIN =0x08000000,LENGTH =512K
  SRAM(rwx):ORIGIN =0x20000000,LENGTH =128K
}
```

```
SECTIONS
{
    .text :
     {
      *(.text)     /*merge all .text sections of input files */
     }> FLASH
}
```

https://go.embeddedexpert.io/article1635692200758

Povezovalna skripta („linker script“) - CubeIDE

Linker skripta določi razpored po pomnilnikih

Introduction to STM32 Memory Model and Linker Scripts

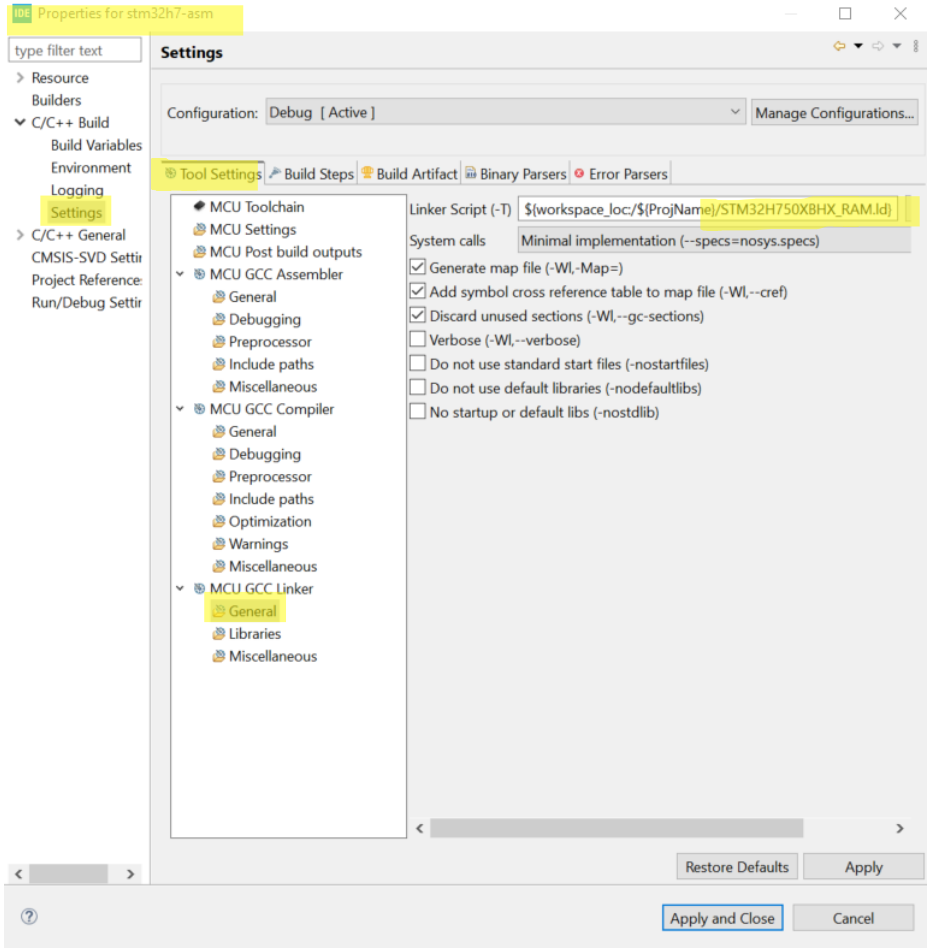
Introduction to the STM32 Memory Model

The memory of STM32 microcontrollers has two main areas :

1. Flash Memory :
•Read-Only
•Begins at address 0x08000000
•Size depends on specific STM32 microcontroller
•Program code is stored here.

•Contains a vector table at address 0x08000004

2. SRAM Memory :
•Read and Write
•Begins at 0x20000000
•Size depends on specific STM32 microcontroller
•Variables and Stack are stored here



The 4 features of a linker script :

•Memory Layout : Available memory types
•Section Definitions : Placing specific parts at specific locations
•Options : Commands e.g. ENTRY POINT
•Symbols : Variables to inject into program at link time

https://go.embeddedexpert.io/article1635692200758>

Povezovalna skripta („linker script“) - CubeIDE

STM32H750XBHX_FLASH.ld :

```
/* Specify the memory areas */
MEMORY
{
  FLASH (rx)      : ORIGIN = 0x08000000, LENGTH = 128K
  DTCMRAM (xrw)   : ORIGIN = 0x20000000, LENGTH = 128K
  RAM_D1 (xrw)    : ORIGIN = 0x24000000, LENGTH = 512K
  RAM_D2 (xrw)    : ORIGIN = 0x30000000, LENGTH = 288K
  RAM_D3 (xrw)    : ORIGIN = 0x38000000, LENGTH = 64K
  ITCMRAM (xrw)   : ORIGIN = 0x00000000, LENGTH = 64K
}
…
SECTIONS
{
  /* The startup code goes first into FLASH */
  .isr_vector :
  {   …
  } >FLASH
/* The program code and other data goes into FLASH */
  .text :
  {    …
  } >FLASH

 /* Initialized data sections goes into RAM, load LMA copy*/
  .data :
  {    …
  } >RAM_D1 AT> FLASH
}
```

STM32H750XBHX_RAM.ld :

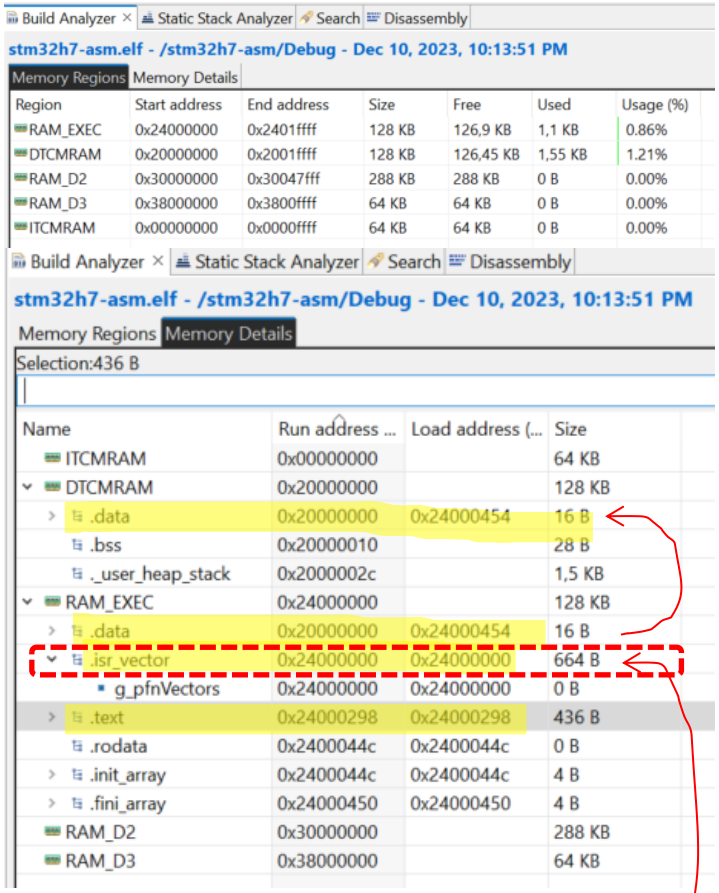
```
/* Specify the memory areas */
MEMORY
{
  RAM_EXEC (xrw)  : ORIGIN = 0x24000000, LENGTH = 128K
  DTCMRAM (xrw)   : ORIGIN = 0x20000000, LENGTH = 128K
  RAM_D2 (xrw)    : ORIGIN = 0x30000000, LENGTH = 288K
  RAM_D3 (xrw)    : ORIGIN = 0x38000000, LENGTH = 64K
  ITCMRAM (xrw)   : ORIGIN = 0x00000000, LENGTH = 64K
}
…

SECTIONS
{
  /* The startup code goes first into FLASH */
  .isr_vector :
  {   …
  } >RAM_EXEC
/* The program code and other data goes into FLASH */
  .text :
  {    …
  } >RAM_EXEC

 /* Initialized data sections goes into RAM, load LMA copy*/
  .data :
  {    …
  } >DTCMRAM AT> RAM_EXEC
}
```

CubeIDE – Izvedba programa - RAM

STM32H750XBHX_RAM.ld:

```
/* Define output sections */
SECTIONS
{
/* The startup code goes first into RAM_EXEC (D1) */
.isr_vector :
{
…
} >RAM_EXEC


/* The program code and other data goes into RAM_EXEC(D1)*/
.text :
{
…
} >RAM_EXEC

/* Initialized data sections go into DTCMRAM from RAM_EXEC*/
.data :
{
…
} >DTCMRAM AT> RAM_EXEC


/* Uninitialized data section goes into DTCMRAM */
.bss :
{
…
} >DTCMRAM
```

```
MEMORY /* Specify the memory areas */
{
RAM_EXEC (xrw) : ORIGIN = 0x24000000, LENGTH = 128K
DTCMRAM (xrw) : ORIGIN = 0x20000000, LENGTH = 128K
RAM_D2 (xrw) : ORIGIN = 0x30000000, LENGTH = 288K
RAM_D3 (xrw) : ORIGIN = 0x38000000, LENGTH = 64K
ITCMRAM (xrw) : ORIGIN = 0x00000000, LENGTH = 64K
}
```

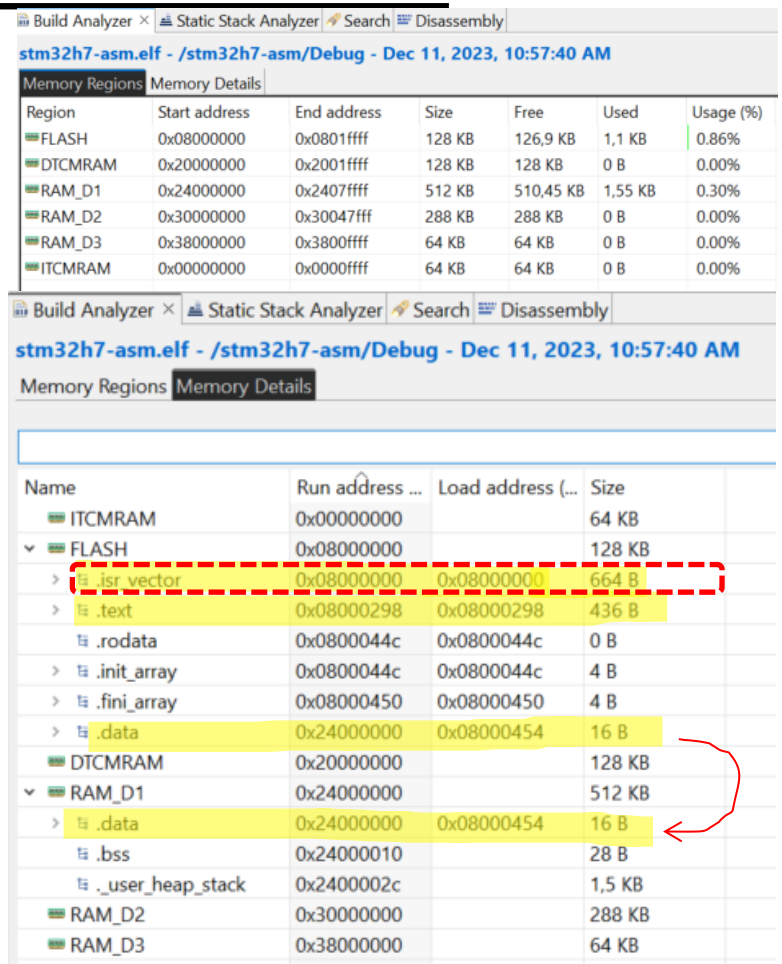



```
Potrebno spremeniti naslov vektorske tabele !!!
//Set Vector table addr. to 0x24000000
            ldr r1, =VTOR
            ldr r0, =0x24000000
            str r0, [r1]
```

CubeIDE – Izvedba programa - Flash

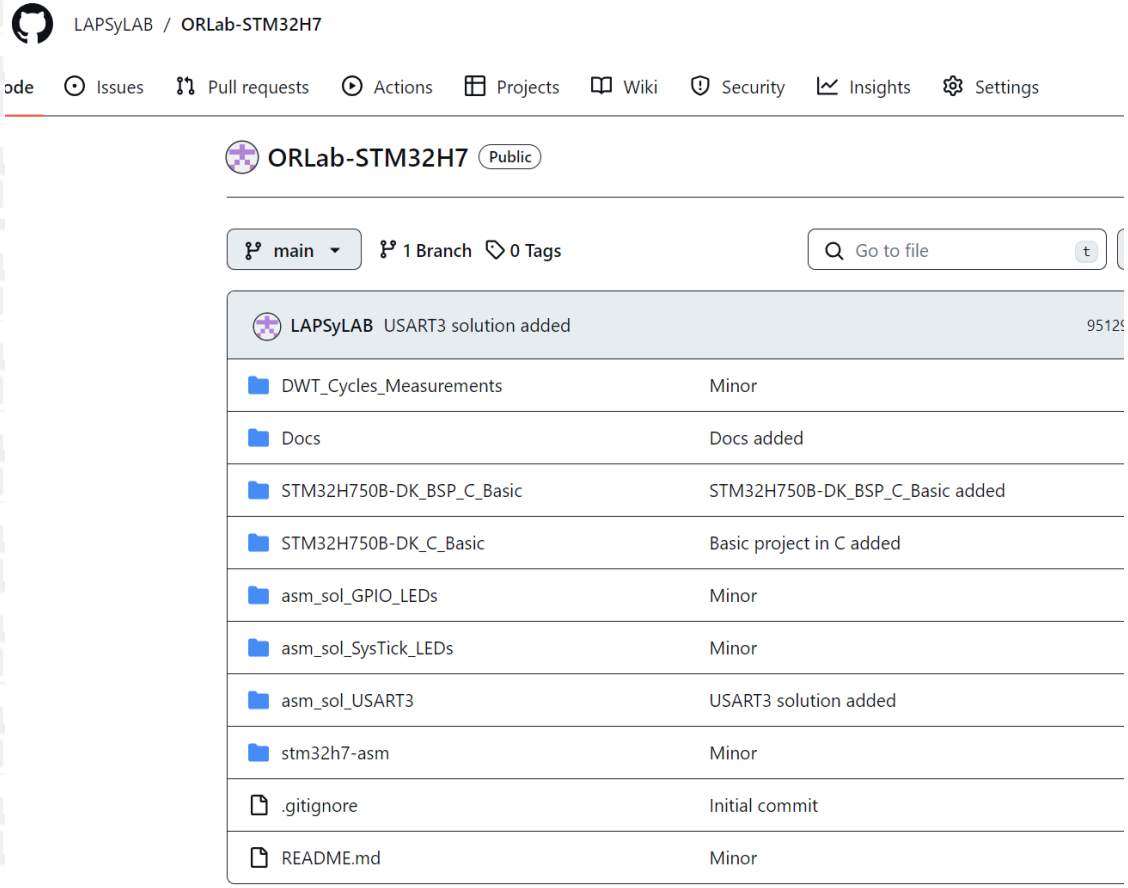
STM32H750XBHX_FLASH.ld:

```
/* Define output sections */
SECTIONS
{
/* The startup code goes first into FLASH */
.isr_vector :
{
…
} >FLASH


/* The program code and other data goes into FLASH */
.text :
{
…
} >FLASH

/* Initialized data sections go into RAM_D1 from FLASH*/
.data :
{
…
} >RAM_D1 AT> FLASH


/* Uninitialized data section */
.bss :
{
…
} >RAM_D1
```




```
MEMORY /* Specify the memory areas */
{
FLASH (rx) : ORIGIN = 0x08000000, LENGTH = 128K
DTCMRAM (xrw) : ORIGIN = 0x20000000, LENGTH = 128K
RAM_D1 (xrw) : ORIGIN = 0x24000000, LENGTH = 512K
RAM_D2 (xrw) : ORIGIN = 0x30000000, LENGTH = 288K
RAM_D3 (xrw) : ORIGIN = 0x38000000, LENGTH = 64K
ITCMRAM (xrw) : ORIGIN = 0x00000000, LENGTH = 64K
}
```

GitHub repozitorij ORLab-STM32H7



Projekti:

- stm32h7-asm
  - **Osnovni projekt za zbirnik**
- STM32H750B-DK_BSP_C_Basic
  - **Osnovni projekt z BSP (brez CubeMx)**
- STM32H750B-DK_C_Basic
  - **Osnovni projekt s CubeMx**
- DWT_Cycles_Measurements
  - Main.s za meritve hitrosti izvajanja delov kode in preučevanje delovanja cevovoda

Docs:

- `Dokumentacija`

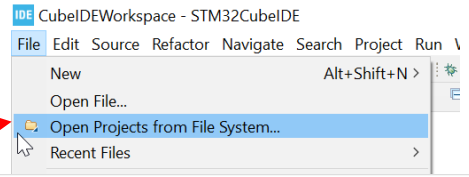
Rešitve:

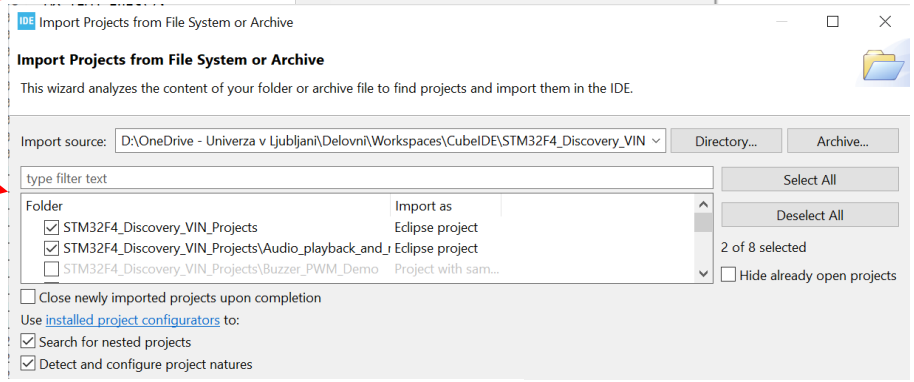
- `asm_sol…`
  - `Main.s datoteke za rešitev naloge`

CubeIDE: Uvoz in delo na projektu

Vzpostavitev začetnega projekta :

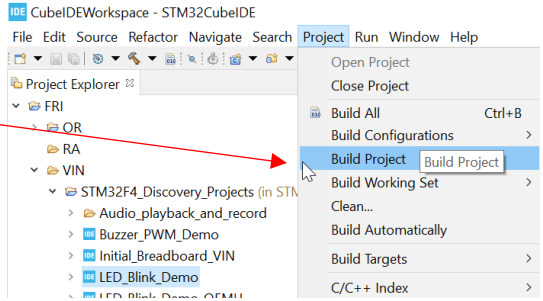
- **Uvoz obstoječega (BSP)**
  - Open projects from File System
  - Select project(s)
- **Nov projekt CubeMX ->**
  (v nadaljevanju)

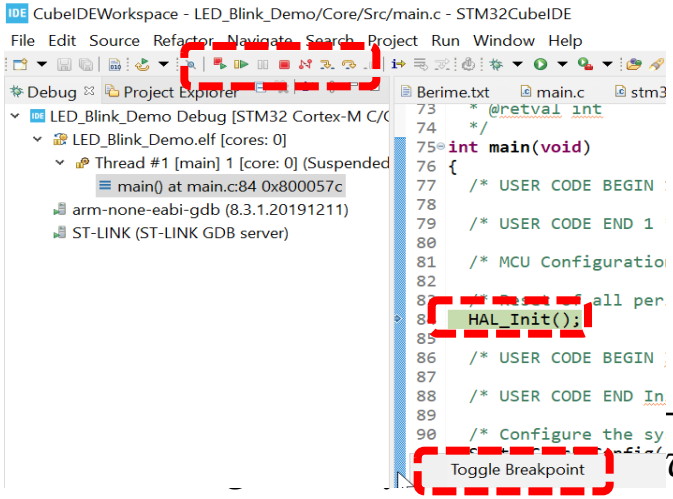


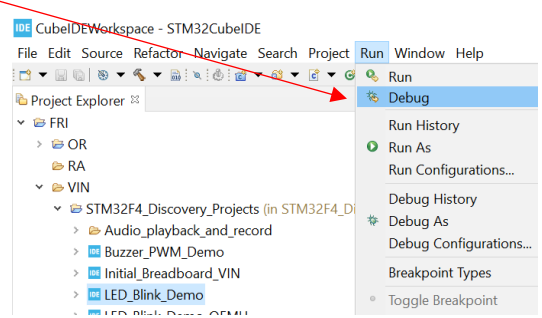


Prevajanje, zagon :

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

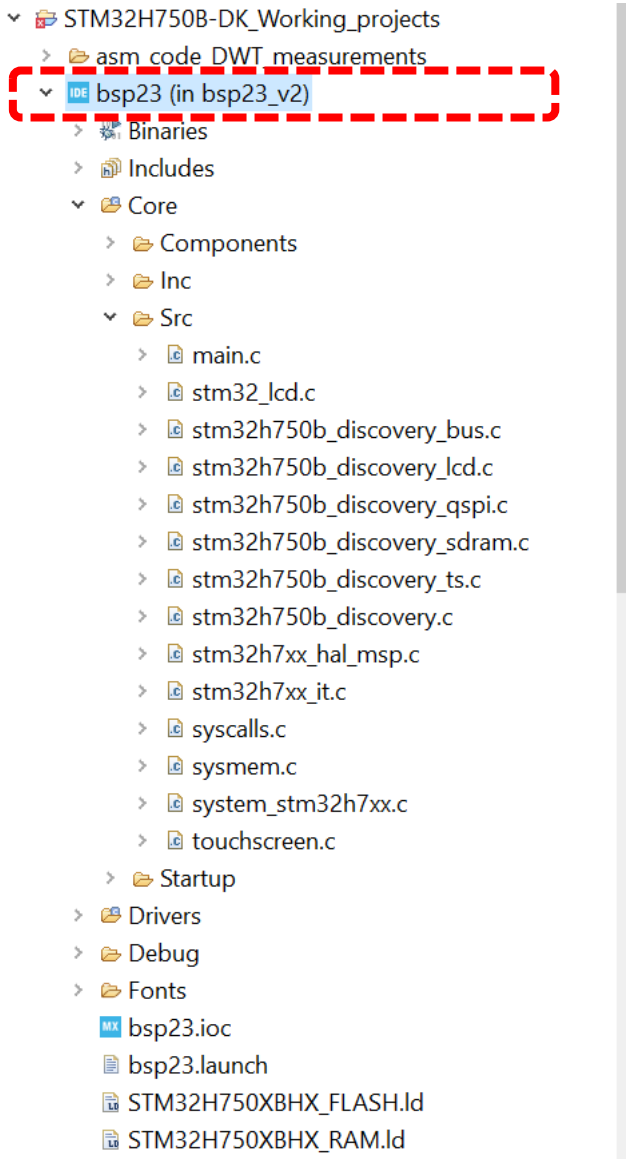






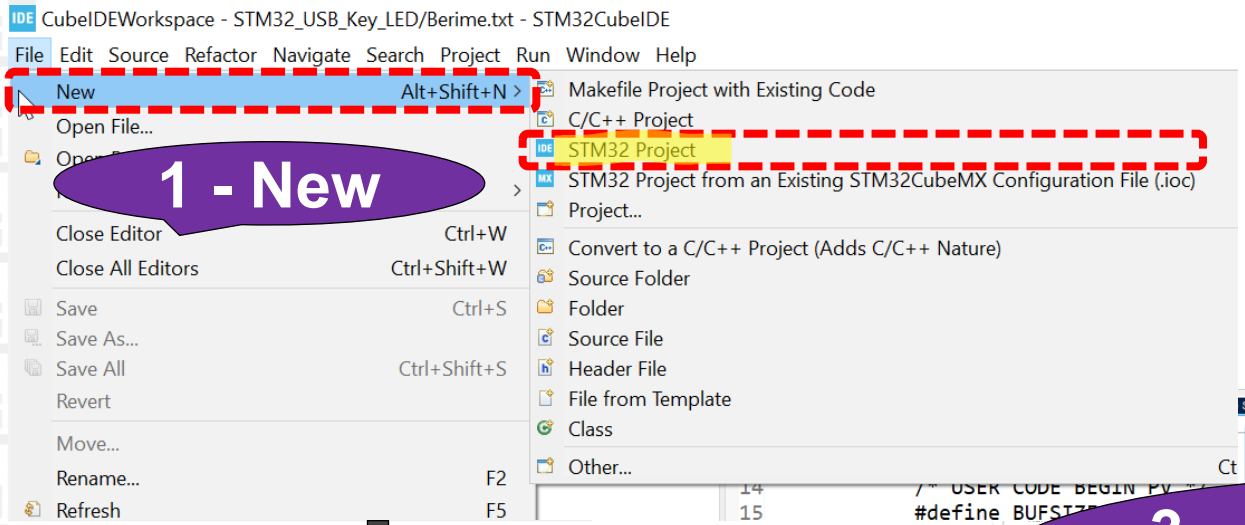
CubeIDE: kopiranje projekta

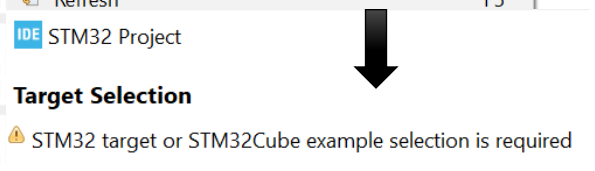
- **Kopiranje CubeIDE projekta z CubeMX .ioc datoteko**
  - 1) Edit > Copy (obstoječi projekt).
  - 2) Edit > Paste (nova lokacija).
  - 3) Preimenuj .ioc datoteko.
  - 4) Zbriši Debug.launch datoteko.
  - 5) Project > Clean.
  - 6) Generiraj kodo s CubeMX.
  - 7) Project > Build Project.
  - 8) Debug As Stm32 Application.
  - 9) Debug aplikacije.
  -
- **Kopiranje osnovnih CubeIDE asm,BSP C projekta**
  - 1) Edit > Copy (obstoječi projekt).
  - 2) Edit > Paste (nova lokacija).
  - 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

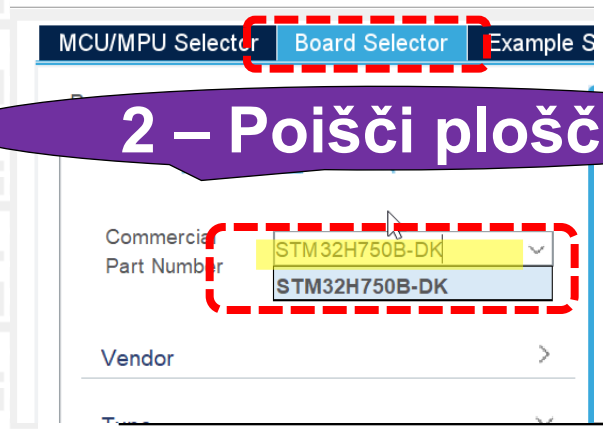


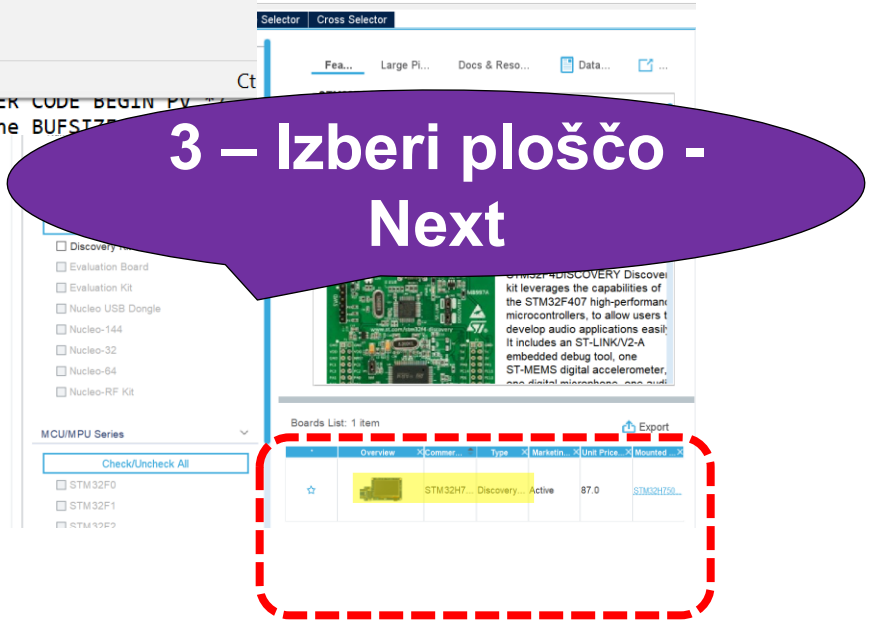
CubeIDE – Vzpostavitev novega projekta s CubeMX

Nov projekt :

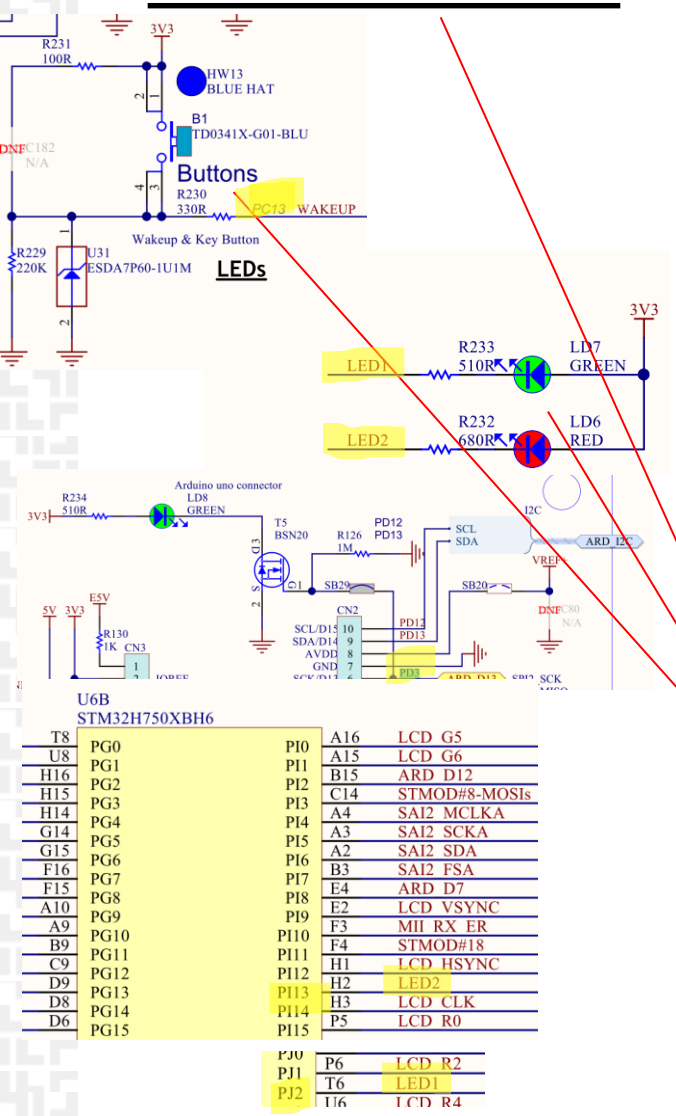




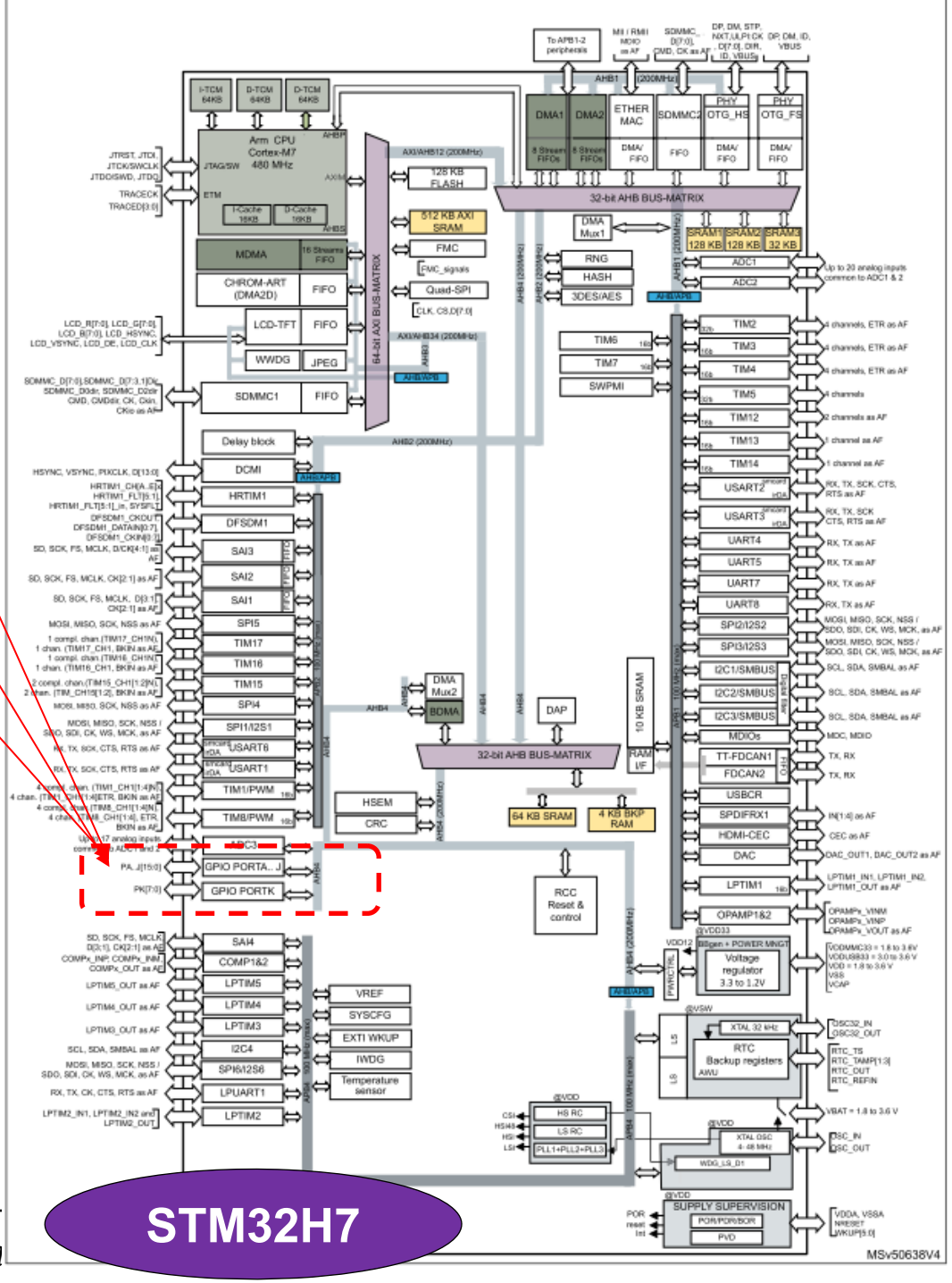




GPIO Krmilnik



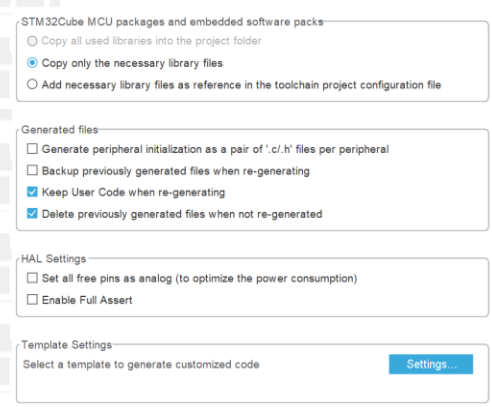
LED: rdeča PI13, zelena PJ2
zelena PD3

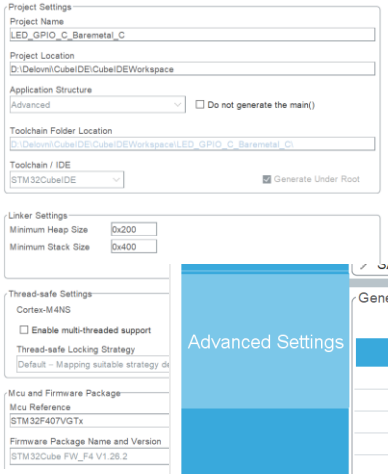


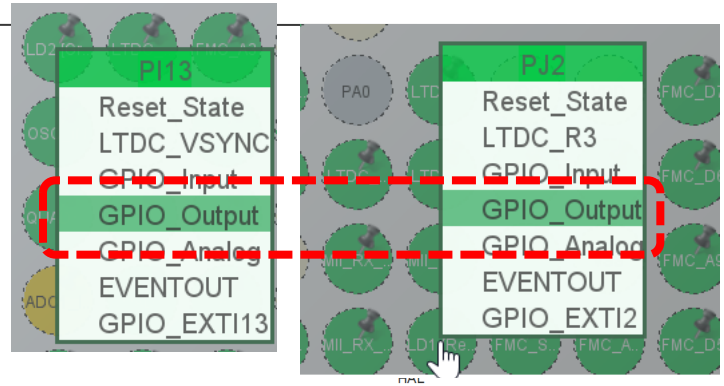
STM32H7

Osnovni projekt CubeIDE – CubeMX

Konfiguracija : priključki, knjižnice STM32H7



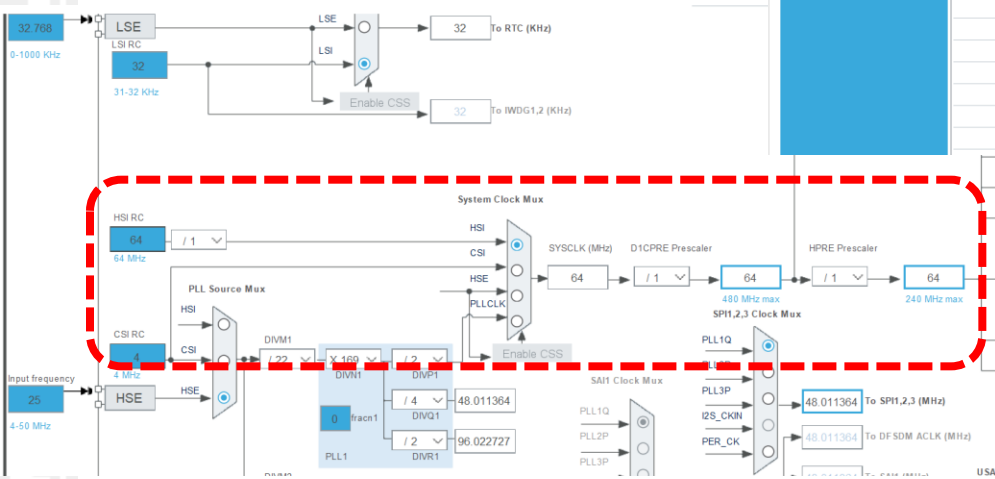




Advanced Settings

Generated Function Calls

Generate Code	Rank	Function Name	Peripheral Instance Name	Do Not Generate Function Call
☑	1	SystemClock_Config	RCC	☐
☑	2	MX_GPIO_Init	GPIO	☐
☑	3	MX_ADC1_Init	ADC1	☐
☑	4	MX_ADC2_Init	ADC2	☐
☑	5	MX_ADC3_Init	ADC3	☐
☑	6	MX_ETH_Init	ETH	☐
☑	7	MX_FDCAN1_Init	FDCAN1	☐
☑	8	MX_FDCAN2_Init	FDCAN2	☐
☑	9	MX_FMC_Init	FMC	☐
☑	10	MX_LTDC_Init	LTDC	☐
☑	11	MX_QUADSPI_Init	QUADSPI	☐
☑	12	MX_RTC_Init	RTC	☐
☑	13	MX_SAI2_Init	SAI2	☐
☑	14	MX_SDMMC1_MMC_Init	SDMMC1	☑



4 – Preveri nastavitve

STM32H7

HAL - C

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

/* USER CODE END PV */
```

UM2217

User manual

Description of STM32H7 HAL and low-layer drivers

35.2.4 **IO operation functions**

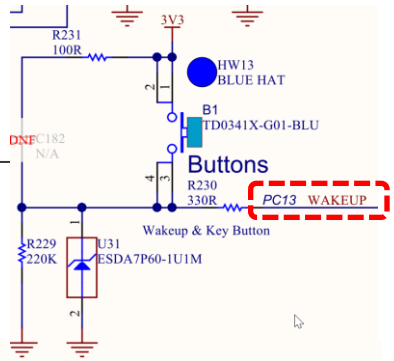
This section contains the following APIs:

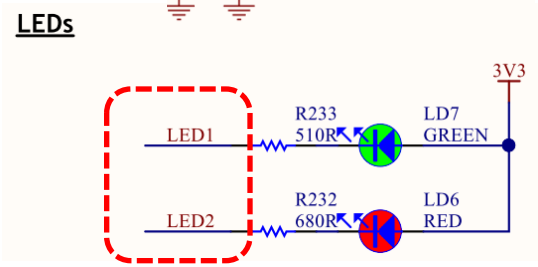
- *HAL_GPIO_ReadPin()*
- *HAL_GPIO_WritePin()*
- *HAL_GPIO_TogglePin()*
- *HAL_GPIO_LockPin()*
- *HAL_GPIO_EXTI_IRQHandler()*
- *HAL_GPIO_EXTI_Callback()*

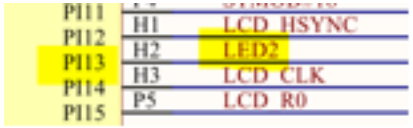
```
/* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {

    HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13);

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

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

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

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

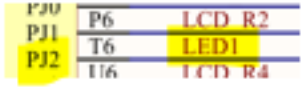
5 – GPIO

6 – USART COM Port



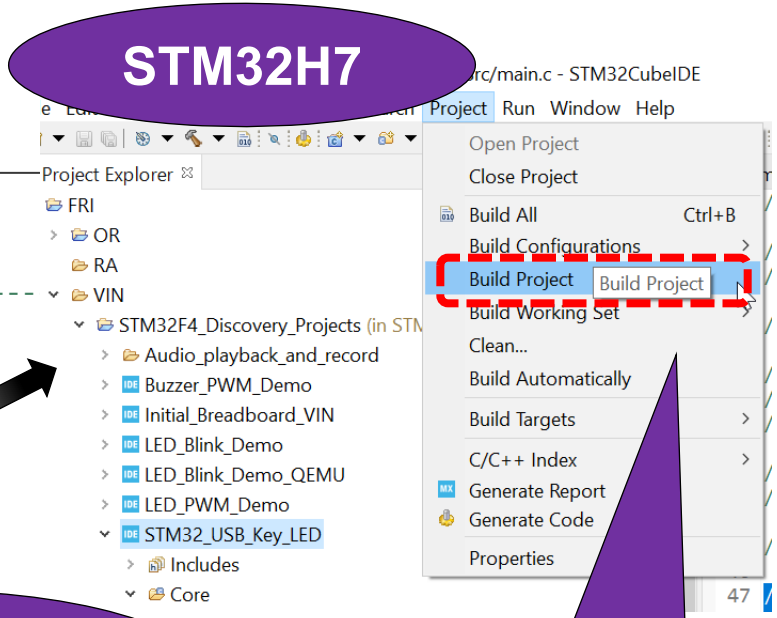






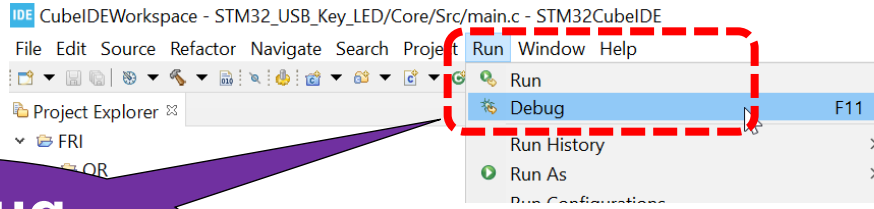
Program : za pošiljanje po USB Virtual COM Port

```
/* Private variables ---------------------------------------

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

/* USER CODE END PV */

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

   HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13);

   KeyState = HAL_GPIO_ReadPin(GPIOC      _PIN_13);
   HAL_GPIO_WritePin(GPIOJ, GPIO     2, KeyState);


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

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

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



7 – Delay

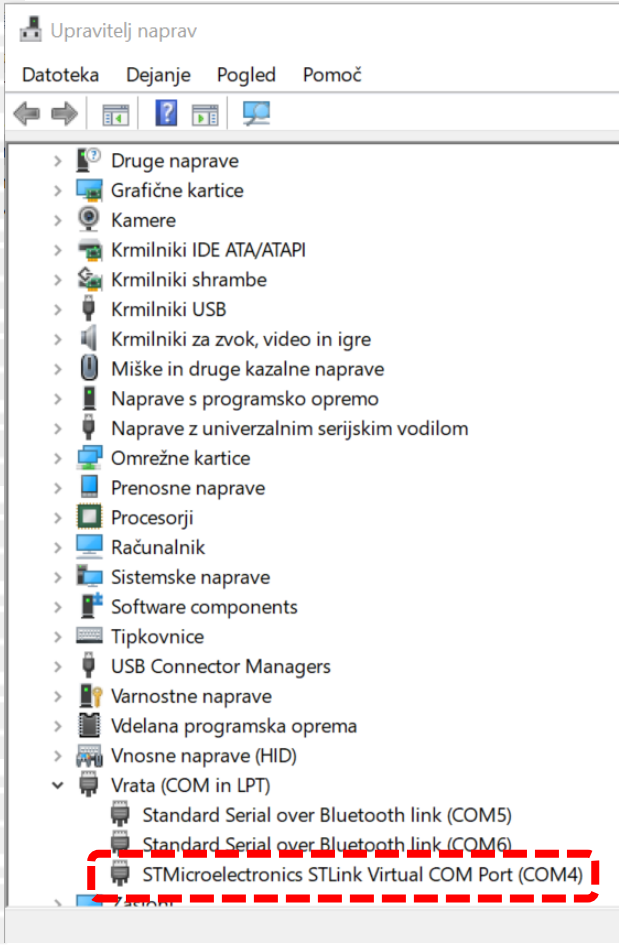
8 – Build project



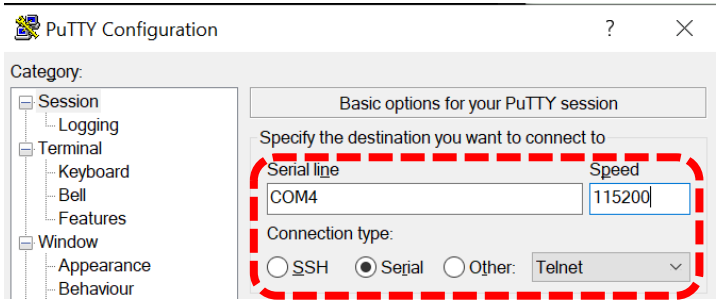
9 – Debug project

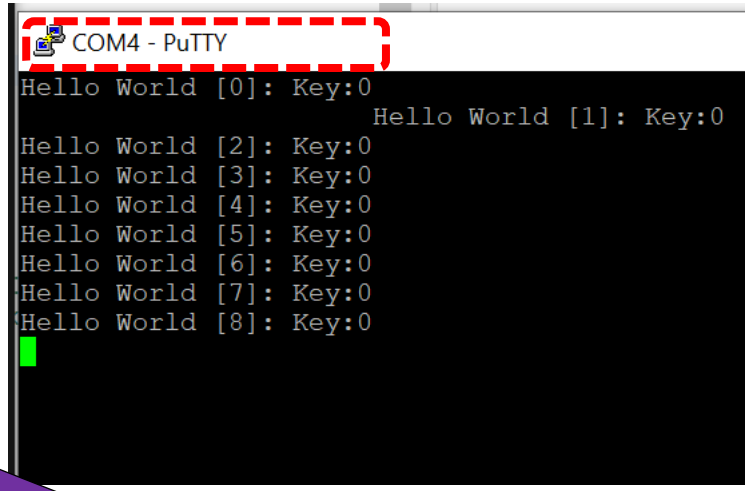
Osnovni projekt CubeIDE – USB Virtual COM Port (USART3 na STM strani)

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



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





10 – Test project