

Vhodno izhodne naprave

Laboratorijska vaja 2 - VP 2
STM32-CubeIDE projekt II, FreeRTOS,
BSP

VIN projekt - Izhodiščni projekti v CubeIDE STM32H7

Vsi projekti temeljijo na uporabi HAL knjižnice

Osnova CubeMX grafični konfigurator

■ Osnovni projekt CubeMX:

- Generiran s CubeMX(HAL knjižnica)

- https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic

■ Projekt CubeMX + FreeRTOS:

- Osnovni projekt CubeMX in dodan FreeRTOS

- FreeRTOS – realno-časni operacijski sistem

- https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic_FreeRTOS

Osnova BSP ročno skodiran projekt

■ Osnovni projekt z vključenim BSP (LCD in Touch demo):

- https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

■ Osnovni projekt BSP in dodan FreeRTOS (SDRAM, LCD demo):

- FreeRTOS – realno-časni operacijski sistem

- https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

VIN LAB 2: VP2 - STM32-CubeIDE projekt, USART, GPIO (LED, tipka), BSP

■ Osnovna projekta v CubeIDE:

□ CubeMX (HAL knjižnica) – STM32H7

■ „izziv glavne zanke“

■ FreeRTOS – realno-časni operacijski sistem

□ reši „izziv glavne zanke“

□ https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic_FreeRTOS

□ BSP (HAL knjižnica) – STM32H7

■ BSP – Board Support Package

□ Zbirka gonilnikov za naprave na plošči (LCD, Touch, Audio, SDRAM, ...)

□ Ni najbolj kompatibilen s CubeMX („ročne“ konfiguracije, temelji na HAL knjižnici)

■ BSP – osnovni demo projekt (Touch, LCD demo)

□ https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

VIN LAB 2: VP2 - STM32-CubeIDE projekt, USART, GPIO (LED, tipka), BSP

■ Osnovna projekta v CubeIDE:

□ CubeMX (HAL knjižnica) – STM32H7

- „izziv glavne zanke“
- FreeRTOS – realno-časni operacijski sistem

□ reši „izziv glavne zanke“

□ https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic_FreeRTOS

□ BSP (HAL knjižnica) – STM32H7

■ BSP – Board Support Package

- Zbirka gonilnikov za naprave na plošči (LCD, Touch, Audio, SDRAM, ...)
- Ni najbolj kompatibilen s CubeMX („ročne“ konfiguracije, temelji na HAL knjižnici)

■ BSP – osnovni demo projekt (Touch, LCD demo)

□ https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

Delo na STM32H7 razvojnem sistemu

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

Priključitev :

- Mikro USB priključek na daljši stranici (srednji !!!)

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

- dodajanje vsebine (main.c):



```
CubelDEWorkspace - Sluzba/ORLab-STM32H7/STM32H750B-DK_C_Basic/Core/Src/main.c - STM32CubelDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
CubelDE_Workspace
Delo
Node_V4 (in node_v4)
Sluzba
  CAN_IEX_Module
  CAN_IEX_Module_bak
  H7-BSP-LCD-OS
  ORLab-STM32
  ORLab-STM32H7
    Docs
    DWT_Cycles_Measurements
    GPIO_LEDs
    STM32H750B-DK_C_Basic
      Core
        Inc
        Src
main.c
131
132 /* Infinite loop */
133 /* USER CODE BEGIN WHILE */
134 while (1)
135 {
136     HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13);
137     HAL_GPIO_TogglePin(GPIOJ, GPIO_PIN_2);
138
139     /* USER CODE END WHILE */
140
141     /* USER CODE BEGIN 3 */
142     snprintf (SendBuffer,BUFSIZE,"USART3:%d secs\r\n",Cnt);
143     HAL_UART_Transmit(&huart3,SendBuffer,strlen(SendBuffer),1);
144
145     HAL_Delay(1000);
146     Cnt++;
147 }
148 /* USER CODE END 3 */
149 }
150
```



Lastni viri :

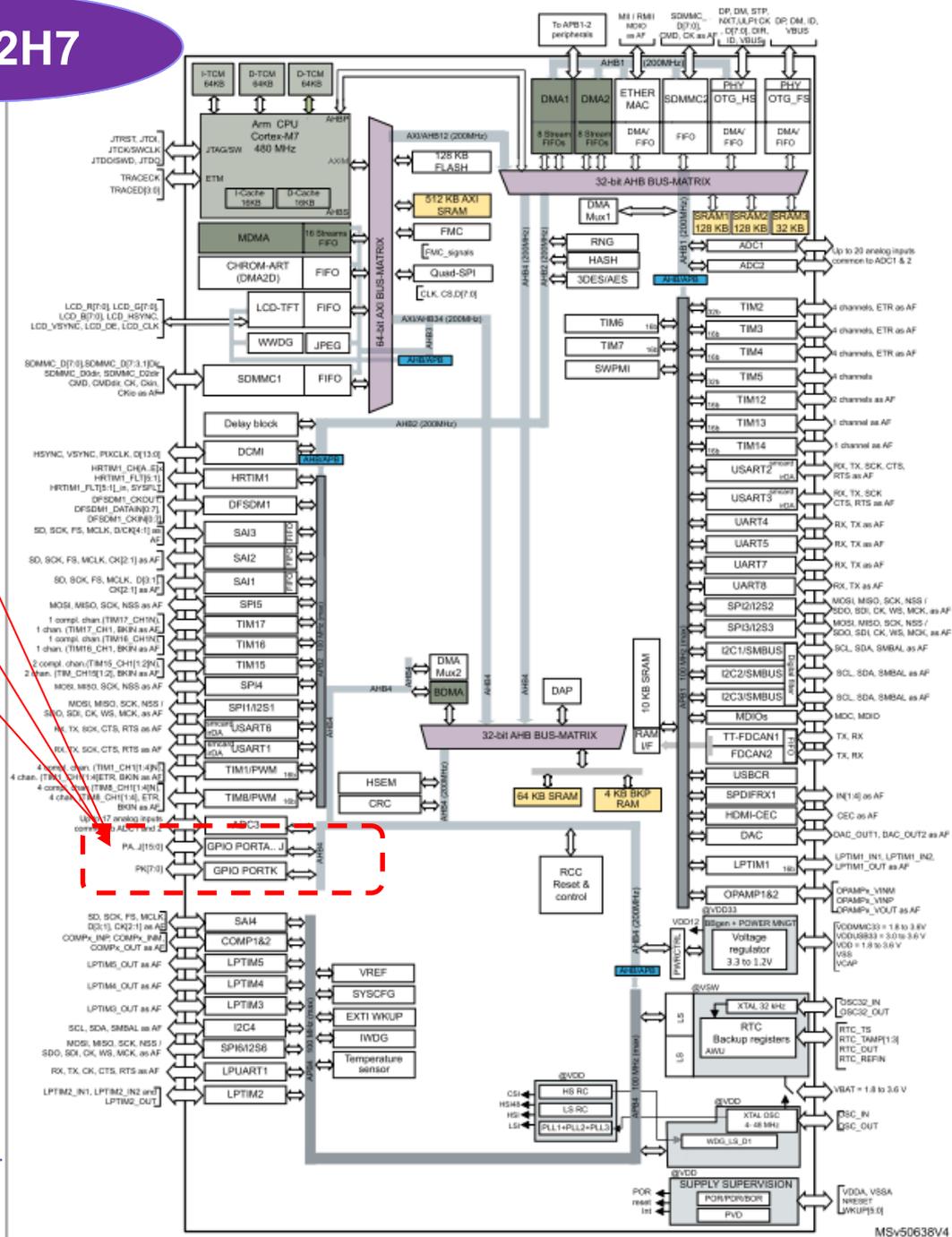
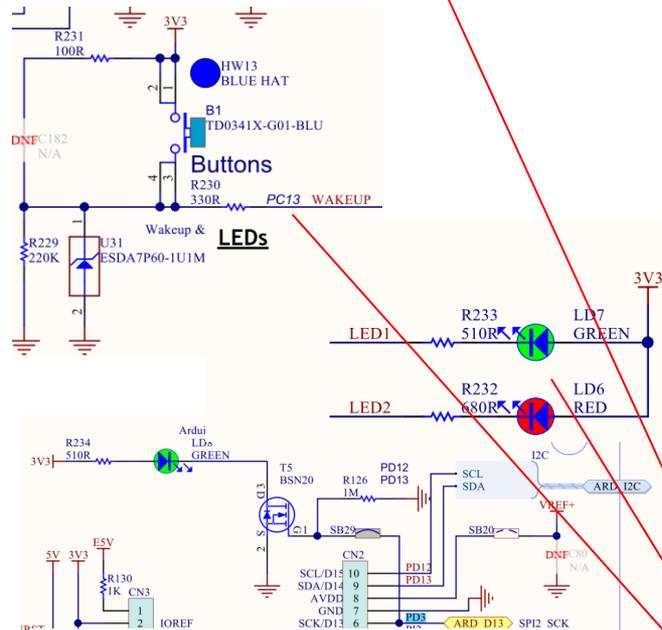
https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects

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



STM32H7

GPIO Krmilnik



U6B STM32H750XBH6

T8	PG0	PI0	A16	LCD G5
U8	PG1	PI1	A15	LCD G6
H16	PG2	PI2	B15	ARD D12
H15	PG3	PI3	C14	STM0D#8-MOSIs
H14	PG4	PI4	A4	SAI2 MCLKA
G14	PG5	PI5	A3	SAI2 SCKA
G15	PG6	PI6	A2	SAI2 SDA
F16	PG7	PI7	B3	SAI2 FSA
F15	PG8	PI8	E4	ARD D7
A10	PG9	PI9	E2	LCD VSYNC
A9	PG10	PI10	F3	MII RX ER
B9	PG11	PI11	F4	STM0D#18
C9	PG12	PI12	H1	LCD HSYNC
D9	PG13	PI13	H2	LED2
D8	PG14	PI14	H3	LCD CLK
D6	PG15	PI15	P5	LCD R0

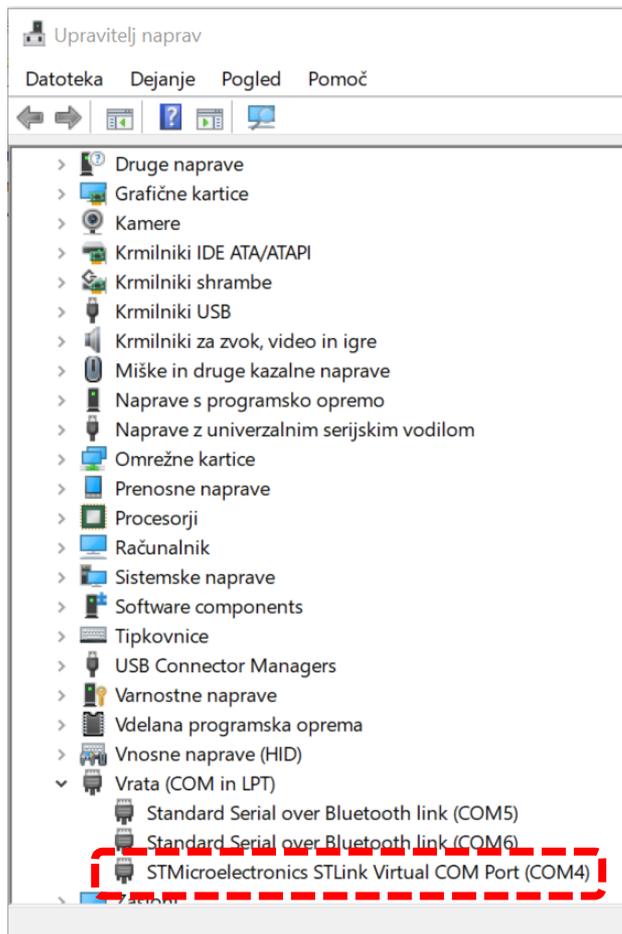
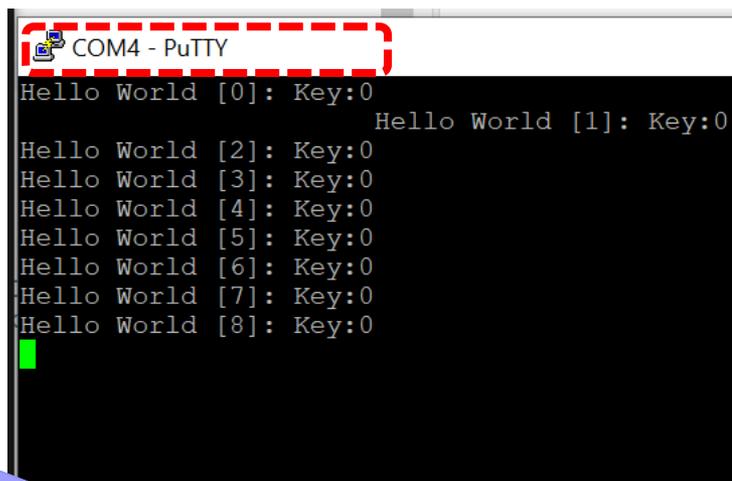
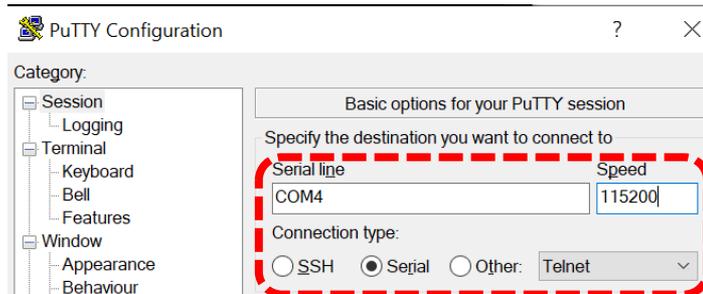
PJ0	P6	LCD R2
PJ1	T6	LED1
PJ2	IT6	LCD R4

LED: rdeča PI13, zelena PJ2
zelena PD3

VIN - LV

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

HAL - C

UM2217

User manual

Description of STM32H7 HAL and low-layer drivers

35.2.4

IO operation functions

This section contains the following APIs:

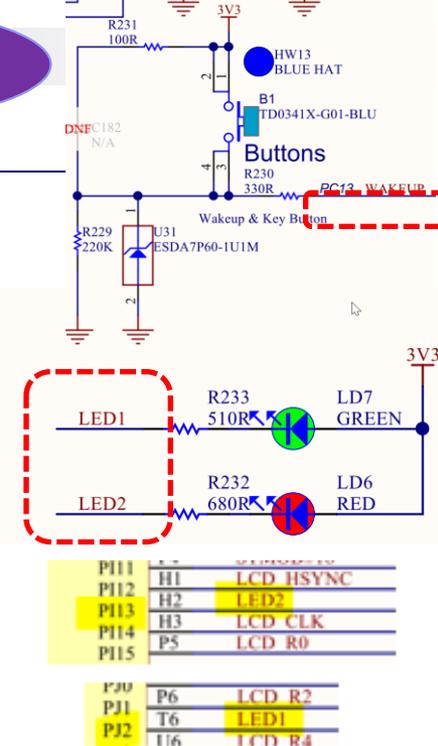
- HAL_GPIO_ReadPin()
- HAL_GPIO_WritePin()
- HAL_GPIO_TogglePin()
- HAL_GPIO_LockPin()
- HAL_GPIO_EXTI_IRQHandler()
- HAL_GPIO_EXTI_Callback()

GPIO

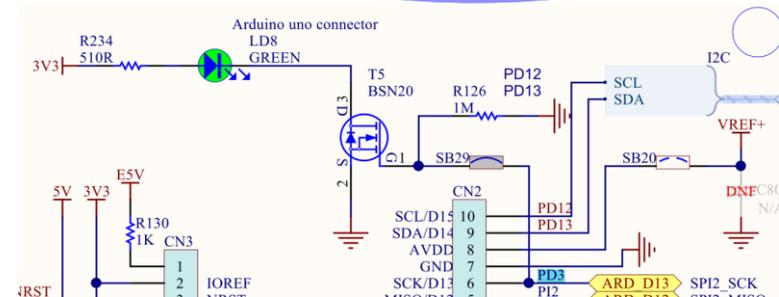
izziv – kako uskladiti, uravnovežiti?

Delay

LEDs



USART COM Port



Načini programiranja „glavne zanke“

Ena zanka
(enostavnejši)

```
{ ...  
  if (Timer_1sec) {  
    readSensors(&data);  
    send_data(&data);  
    Timer_1sec = 0;  
  }  
  
  if (Timer_50msec) {  
    readKeys(&keys);  
    readInputs(&inputs);  
    Timer_50msec = 0;  
  }  
}
```



Tevel

Merilnik konc. plinov

Končni avtomat
(kompleksnejši)

```
switch (FSM.State) {  
  case CHECK_REASON:  
    ///  
    FSM.State: after reset.  
  
    if VSE_OK then  
      FSM.State = CHECK_BAUDRATE  
  
      break;  
  
  case CHECK_BAUDRATE:  
    ///  
    FSM.State: after reset.  
  
    ...  
  
    break;  
  
  ...  
}
```



Cubesensor

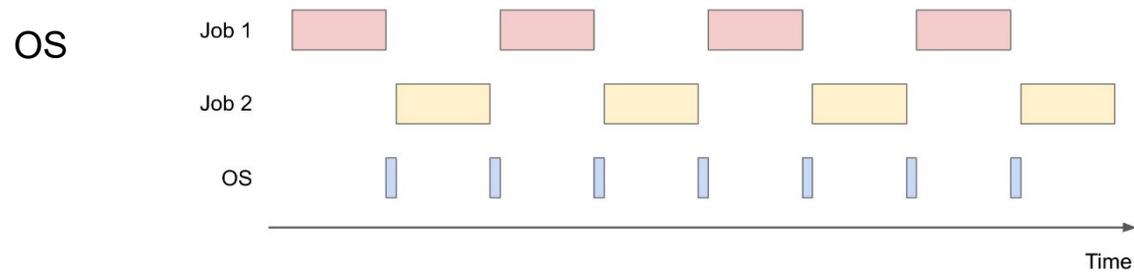
Merilnik kakovosti zraka

RTOS
(ločeni procesi)

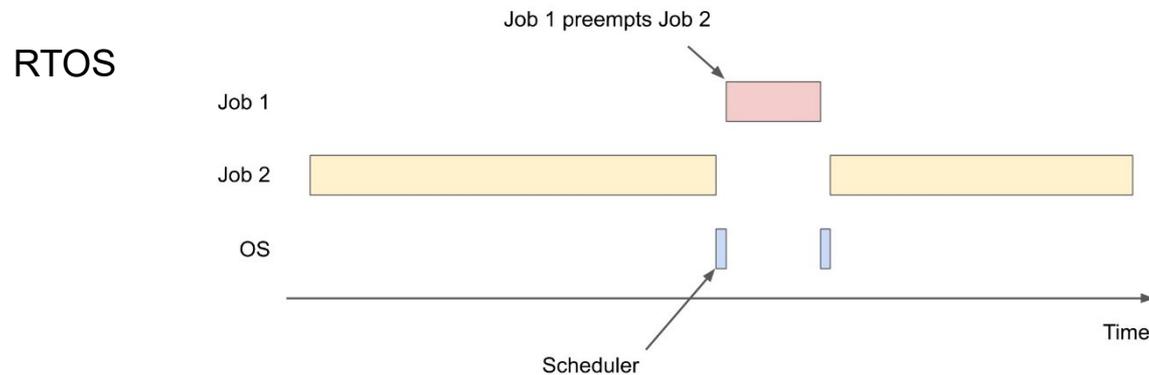
```
void StartTask02(void *argument)  
{  
  /* USER CODE BEGIN StartTask02 */  
  /* Infinite loop */  
  for(;;)  
  {  
    HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);  
    osDelay(1000);  
  }  
  /* USER CODE END StartTask02 */  
}  
  
void StartTask01(void *argument)  
{  
  /* USER CODE BEGIN StartTask01 */  
  /* Infinite loop */  
  for(;;)  
  {  
    HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);  
  
    osDelay(1000);  
  
  }  
  /* USER CODE END StartTask01 */  
}
```

Programiranje vgrajenih sistemov – OS vs. RTOS

General Purpose Operating System



Real-Time Operating System (RTOS)



Splošno o RTOS - Real Time Operating System

- RTOS upravlja čas in procese na mikroprocesorju ali mikrokrmilniku
 - v bistvu: „poenostavljen operacijski sistem“

- Funkcionalnosti RTOS:
 - **Večopravilnost** (multi-tasking)
 - Dodeljevanje opravil CPE s prioritetai
 - **Sinhronizacija** dostopov do virov:
 - V/I naprav
 - Pomnilnika (podatkovnih struktur)
 - **Komunikacija** med procesi (Inter-task communication)
 - Časovna predvidljivost (realno-časovna odzivnost)
 - Servisiranje prekinitev

Zakaj uporabiti RTOS?

- Uporaba V/I naprav (že pripravljene **driverji** (TCP, ETH, CANBUS,...))
- **Se splača** vse razviti iz nič (npr. svoj dodeljevalnik) ?
 - Diploma : Fabčič – 2021 – lasten RTOS „from scratch“
- Večopravilnost z možnostjo sinhronizacije
- **Prenosljivost** kode na druge CPE
- **Upravljanje z viri**
- Možnost dopolnitve z lastnimi funkcijami
- **Obstoječa podpora** za nekatere razširjene protokole:
 - TCP/IP, USB, Flash Systems, Web Servers,
 - CAN protocols, GUI, SSL, SNMP

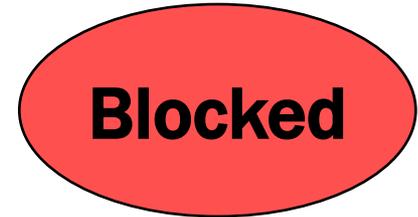
Nekatere potencialne prednosti se hitro sprevržejo v težave in dodatno delo...

the task is ready and is running because it's the highest-priority ready task

**Task Finishes
Explicit
Termination**

**Blocking
Call**

*the task is blocked
and therefore not
ready
it's waiting for a
condition to be
true*



**Higher-priority Task
becomes Ready
Time Slice Expires
Interrupt comes in**

Context Switch



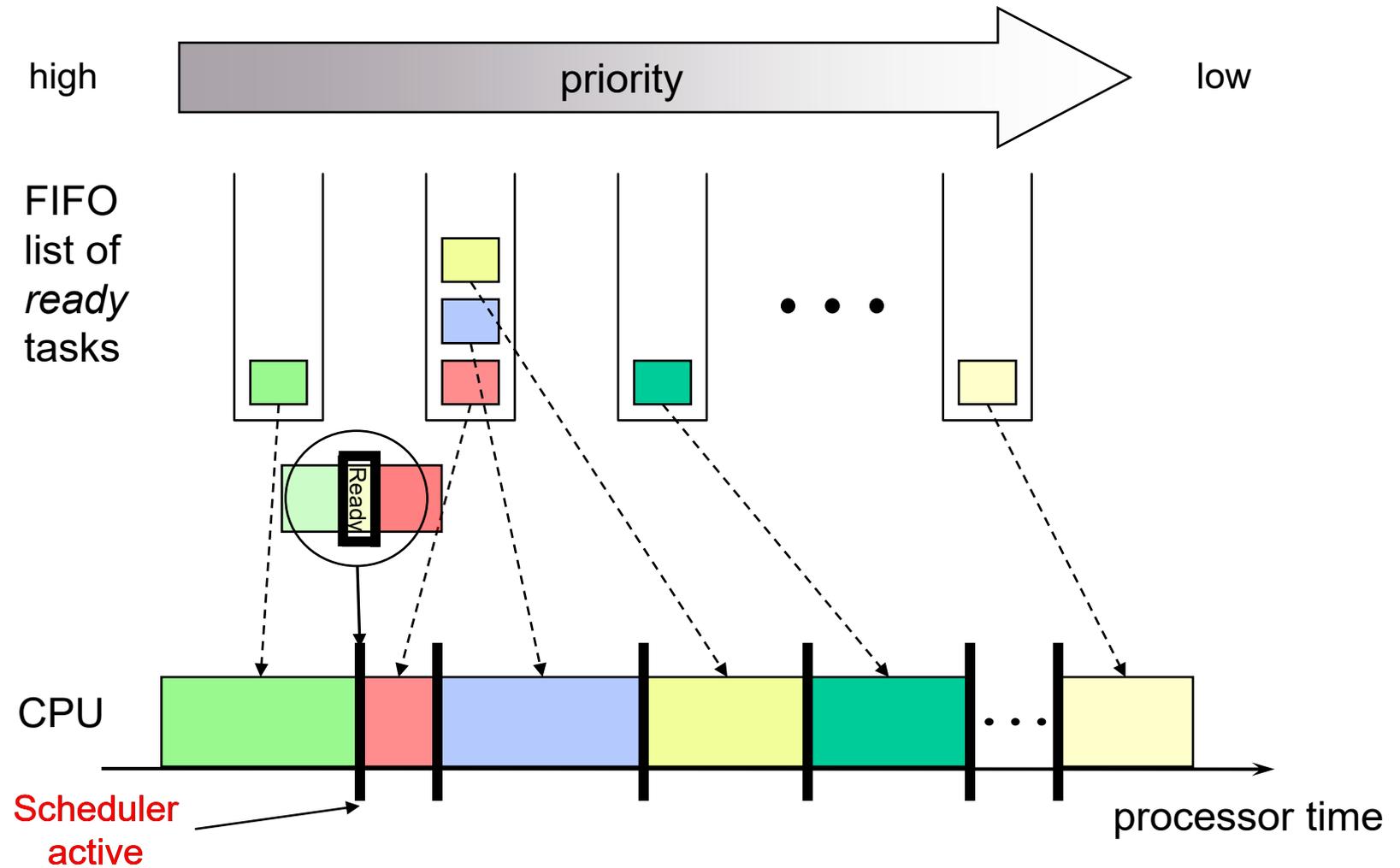
*the task has finished all
its work, or was explicitly
destroyed*

**Task
Starts**

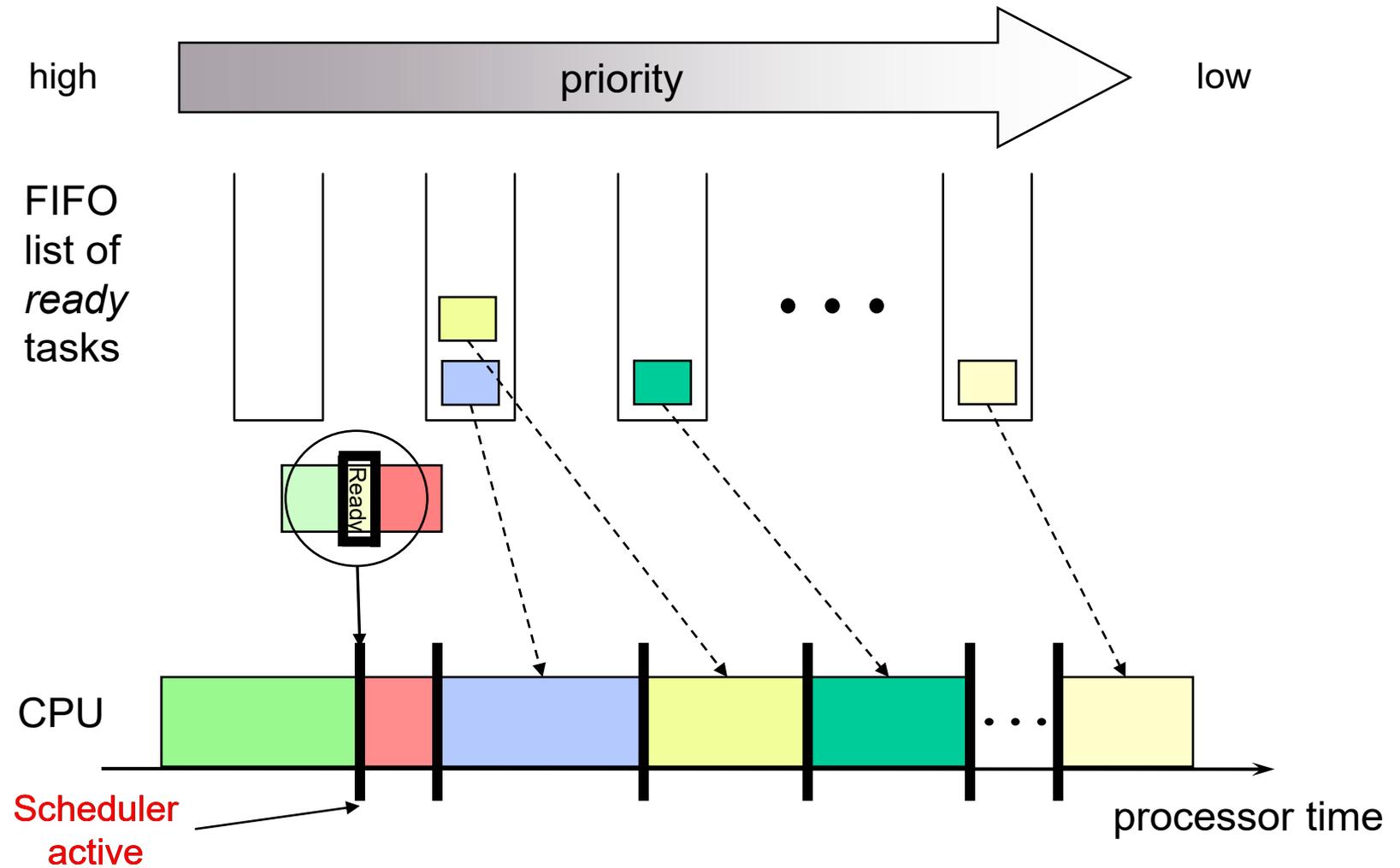
**Object
Available
Timeout Expires**

*the task is ready, but it's not
running because it isn't the
highest-priority ready task*

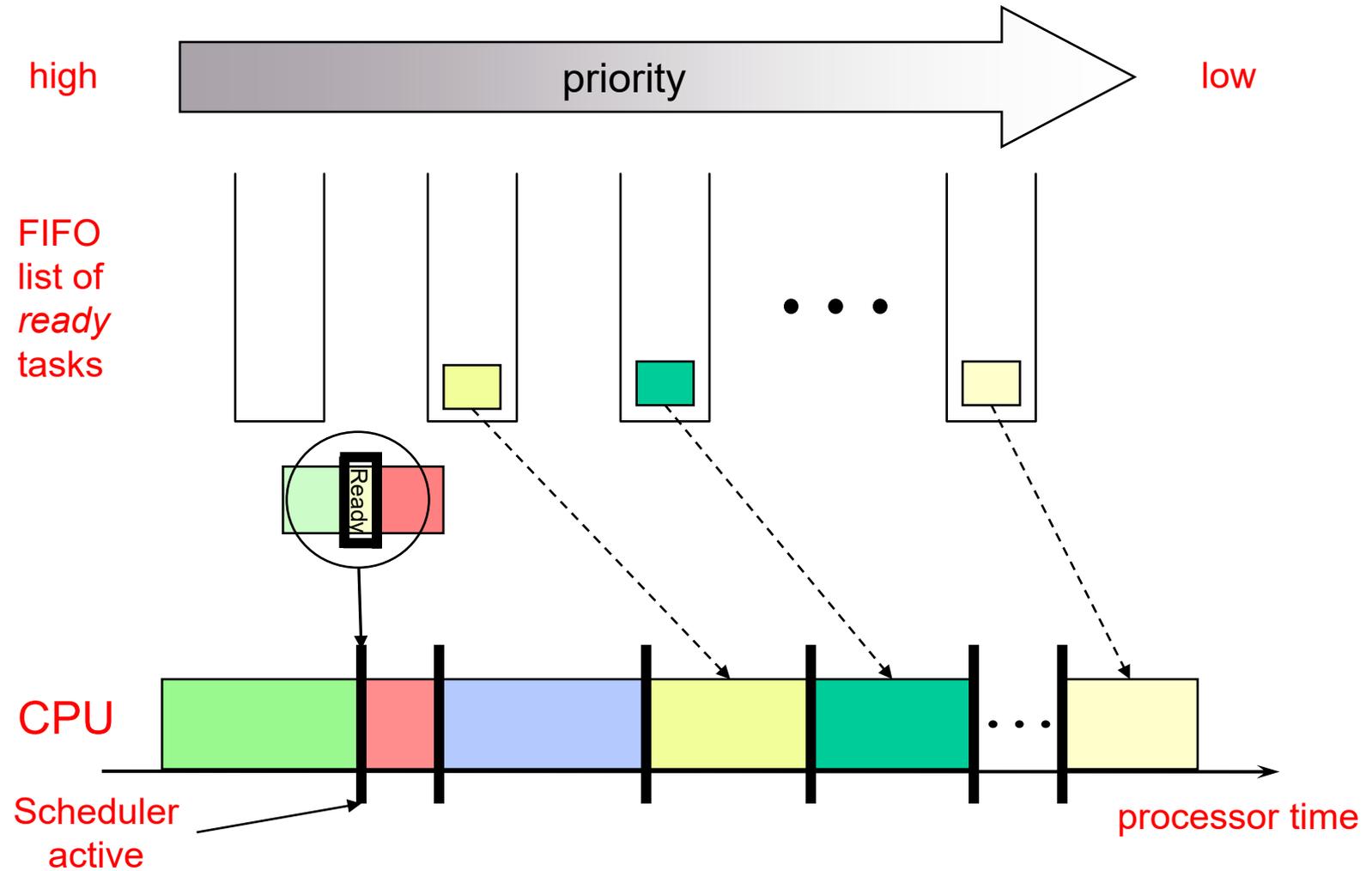
Priority Based FIFO Scheduling



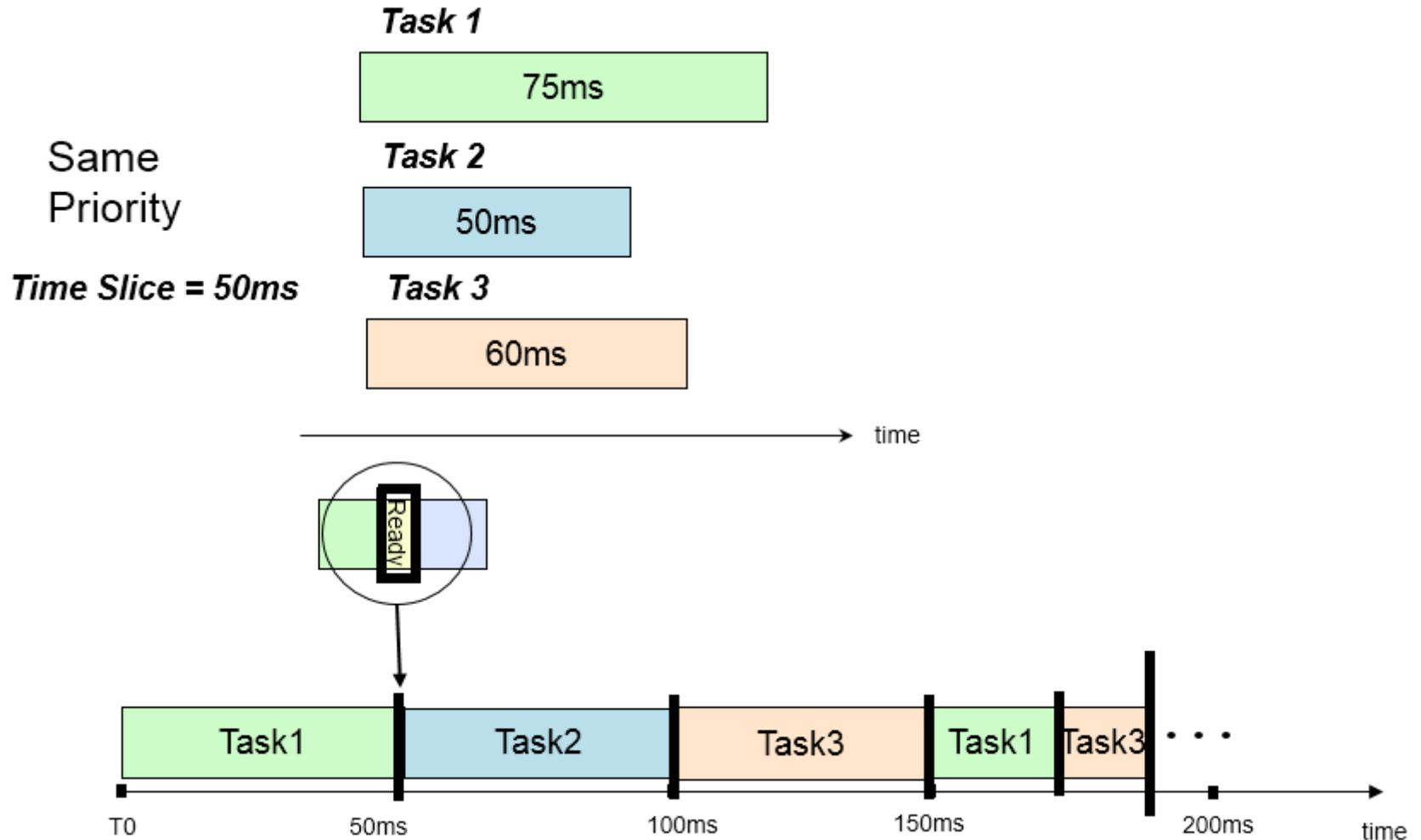
Priority Based FIFO Scheduling



Priority Based FIFO Scheduling



Same priority – Round Robin Scheduling



RTOS - Opravila

- Sistem oz. aplikacija je sestavljena iz več opravil
- Opravila se izmenjaje izvajajo
- V nekem trenutku je aktivno natanko eno opravilo (se izvaja na procesorju)
- RTOS odloča, kako si opravila delijo procesor („context switching“)
- Vsebina opravila („Task Context“)
 - Podatkovna struktura lastna vsakemu opravilu:
 - Vsebuje vse potrebne podatke za izvedbo opravila:
 - npr. spremenljivke, registre in sezname vseh uporabljenih virov

Tipična struktura kode opravila

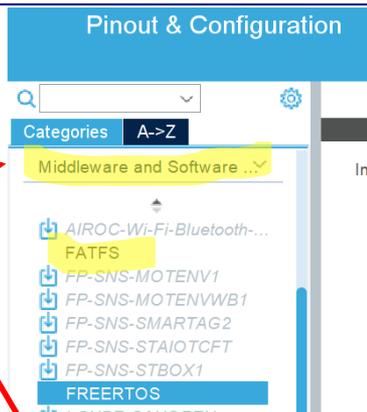
```
void StartTaskLED(void *argument)
{
/* USER CODE BEGIN StartTaskLED */
/* Infinite loop */
for(;;)
{
    HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13); // Negiraj dig. izhod za rdečo LED diodo
    HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_3); // Negiraj dig. izhod za zeleno LED diodo

    osDelay(1000);
}
/* USER CODE END StartTaskLED */
}
void StartTaskKEYLED(void *argument)
{
/* USER CODE BEGIN StartTaskKEYLED */
/* Infinite loop */
for(;;)
{
    KeyState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13); // Preberi stanje dig. vhoda (USER tipka) v KeyState
    HAL_GPIO_WritePin(GPIOJ, GPIO_PIN_2, KeyState); // Zapiši stanje tipke na dig. izhod za zeleno LED diodo
    osDelay(50);
}
/* USER CODE END StartTaskKEYLED */
}
```

```
void mytask(uint_32 startup_parameter) {
    /* Task initialization code */
    ....
    while (1) {
        /* Task body */
        ....
        ....
    }
}
```

Dodatek FreeRTOS:

1. „Middleware and SW packs“
2. FreeRTOS activate (CMSIS V2)
3. Kreiraj opravila (tasks):
 1. LED Blinking task
 2. KEY & LED task
 3. defaultTask (že obstaja)



GPIO – GPIO_Output za LED:

LED: rdeča PI13, zelena PJ2

zelena PD3

PI13	PJ2	PD3
Reset_State	Reset_State	Reset_State
LTDC_VSYNC	LTDC_R3	DCMI_D5
GPIO_Input	GPIO_Input	DFSDM1_CKOUT
GPIO_Output	GPIO_Output	FMC_CLK
GPIO_Analog	GPIO_Analog	I2S2_CK
EVENTOUT	EVENTOUT	LTDC_G7
GPIO_EXTI13	GPIO_EXTI2	SPI2_SCK
		USART2_CTS
		USART2_NSS
		GPIO_Input
		GPIO_Output
		GPIO_Analog
		EVENTOUT
		GPIO_EXTI3

New Task

Task Name: LED_Blinking

Priority: osPriorityLow

Stack Size (Words): 128

Entry Function: StartTaskLED

Code Generation Option: Default

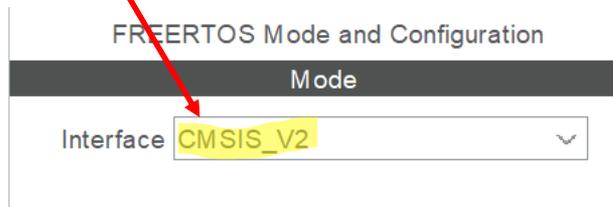
Parameter: NULL

Allocation: Static

Buffer Name: LED_BlinkingBuffer

Control Block Name: LED_BlinkingControlBlock

OK Cancel



Tasks and Queues		Timers and Semaphores		Mutexes		Events		FreeRTOS Heap Usage	
Config parameters		Include parameters		Advanced settings		User Constants			
Task Name	Priority	Stack Size (Wor...	Entry Function	Code Generatio...	Parameter	Allocation	Buffer Name	Control Block N...	
defaultTask	osPriorityNormal	128	StartDefaultTask	Default	NULL	Dynamic	NULL	NULL	
LED_Blinking	osPriorityLow	128	StartTaskLED	Default	NULL	Static	LED_BlinkingBuf...	LED_BlinkingCo...	
KEY_LED	osPriorityLow	128	StartTaskKEYLED	Default	NULL	Static	KEY_LEDBuffer	KEY_LEDContr...	

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

If configUSE_NEWLIB_REENTRANT is set to 1 then a newlib reent structure will be allocated for each created task.

Dodatek FreeRTOS:

4. Newlib reentrant
5. HAL Timebase Source to TIM6 (FreeRTOS has SysTick)

The screenshot shows the STM32CubeIDE configuration interface. At the top, there are several configuration categories: Mutexes, Events, FreeRTOS Heap Usage, User Constants, Tasks and Queues, Timers and Semaphores, Config parameters, Include parameters, and Advanced settings. Below these, a search bar is visible. The 'Newlib settings (see parameter descri...)' section is expanded, showing 'USE_NEWLIB_REENTRANT' set to 'Enabled'. Below this, the 'Pinout & Configuration' and 'Clock Configuration' tabs are visible. The 'Pinout & Configuration' tab is active, showing a list of components under 'System Core'. The 'SYS' component is highlighted. The 'Clock Configuration' tab is also visible, showing 'SYS Mode and Configuration' with 'Mode' set to 'Timebase Source' and 'TIM6' selected. A warning message is displayed at the bottom: 'Warning: This peripheral has no parameters to be co...'

⚠ WARNINGS:

- When RTOS is used, it is strongly recommended to use a HAL timebase source other than the SysTick. The HAL timebase source can be changed from the Pinout tab under SYS

- The USE_NEWLIB_REENTRANT must be set in order to make sure that newlib is fully reentrant. The option will increase the RAM usage. Enable this option under FreeRTOS > Advanced Settings > USE_NEWLIB_REENTRANT

https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic_FreeRTOS

Dodatek FreeRTOS:

4. Koda za opravila
5. Compile & Debug

```
void StartDefaultTask(void *argument)
{
  /* USER CODE BEGIN 5 */
  /* Infinite loop */
  for(;;)
  {
```

```
  snprintf(SendBuffer, BUFSIZE, "Hello World [%d]:
  Key:%d\n\r", Counter++, KeyState); // Pripravi tekstovno sporočilo s stanjem v
  polje znakov - string
  HAL_UART_Transmit(&huart3, SendBuffer, strlen(SendBuffer), 100); // Pošlji
  sporočilo s stanjem po USART povezavi v PC
```

```
  osDelay(2000);
}
/* USER CODE END 5 */
}
```

```
void StartTaskLED(void *argument)
{
  /* USER CODE BEGIN StartTaskLED */
  /* Infinite loop */
  for(;;)
  {
    HAL_GPIO_TogglePin(GPIOI, GPIO_PIN_13); // Negiraj dig. izhod za rdečo LED diodo
    HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_3); // Negiraj dig. izhod za zeleno LED diodo

    osDelay(1000);
  }
  /* USER CODE END StartTaskLED */
}
```

```
void StartTaskKEYLED(void *argument)
{
  /* USER CODE BEGIN StartTaskKEYLED */
  /* Infinite loop */
  for(;;)
  {
    KeyState = HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13); // Preberi stanje dig. vhoda
    (USER tipka) v KeyState
    HAL_GPIO_WritePin(GPIOD, GPIO_PIN_2, KeyState); // Zapiši stanje tipke na dig.
    izhod za zeleno LED diodo
    osDelay(50);
  }
  /* USER CODE END StartTaskKEYLED */
}
```

HAL_Delay(): The CPU Hog (Blocking Delay)

When you call `HAL_Delay(100)`, the CPU sits in a tight `while` loop, constantly checking if 100 milliseconds have passed on the HAL timebase (which we just set to TIM6).

osDelay(): The Polite Yielder (Non-Blocking Delay)

When you call `osDelay(100)` inside a task, you are telling the RTOS scheduler: "I don't need to do anything for the next 100 OS ticks. Put me to sleep and let someone else use the CPU."

VIN LAB 2: VP2 - STM32-CubeIDE projekt, USART, GPIO (LED, tipka), BSP

■ Osnovna projekta v CubeIDE:

□ CubeMX (HAL knjižnica) – STM32H7

■ „izziv glavne zanke“

■ FreeRTOS – realno-časni operacijski sistem

□ reši „izziv glavne zanke“

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

□ BSP (HAL knjižnica) – STM32H7

■ BSP – Board Support Package

□ Zbirka gonilnikov za naprave na plošči (LCD, Touch, Audio, SDRAM, ...)

□ Ni najbolj kompatibilen s CubeMX („ročne“ konfiguracije, temelji na HAL knjižnici)

■ BSP – osnovni demo projekt (Touch, LCD demo)

□ https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

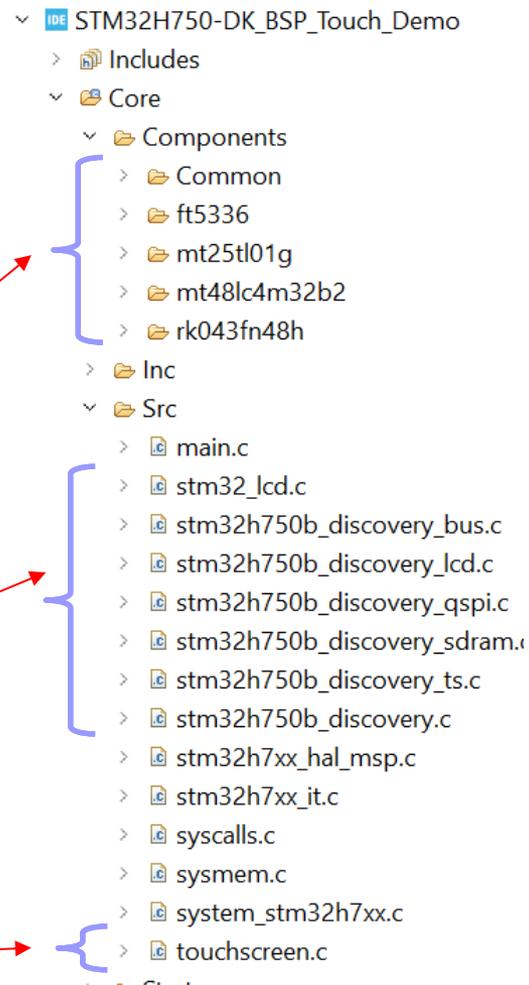
CubeIDE – BSP projekt

■ BSP – Board Support Package

- zbirka gonilnikov za naprave na plošči (LCD, Touch, Audio, SDRAM, ...)
- „ni najbolj kompatibilen“ s CubeMX („ročne“ konfiguracije, temelji na HAL knjižnici)

■ BSP – osnovni demo projekt (Touch, LCD demo)

- Pripravljen projekt, ki vključuje
 - BSP in
 - osnovni demo touch, LCD



https://github.com/LAPSYLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750-DK_BSP_Touch_Demo

CubeIDE – Uvoz obstoječega BSP projekta

Vzpostavitev začetnega projekta :

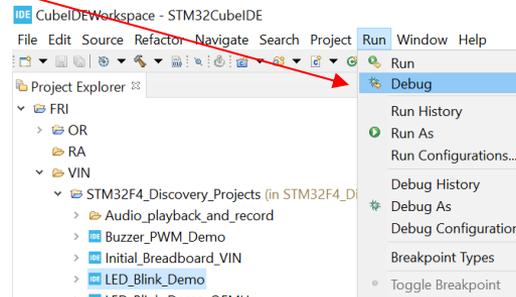
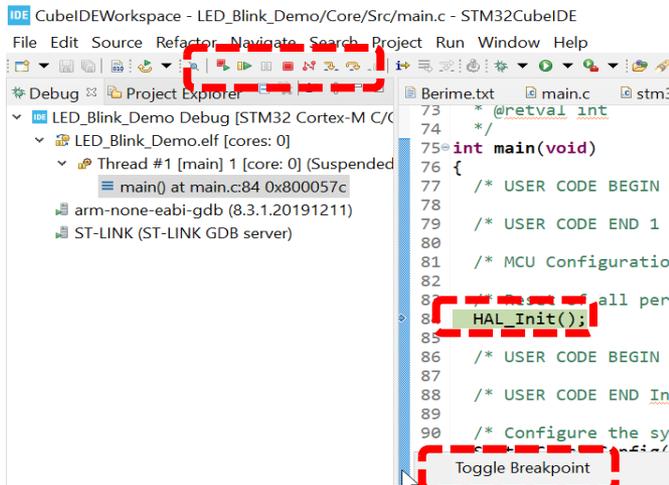
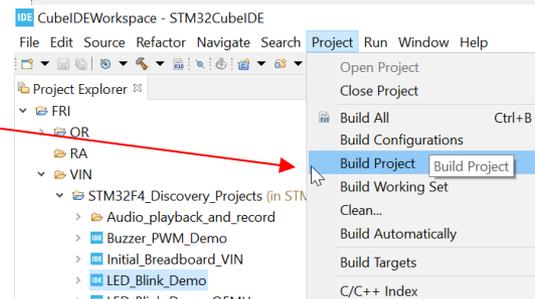
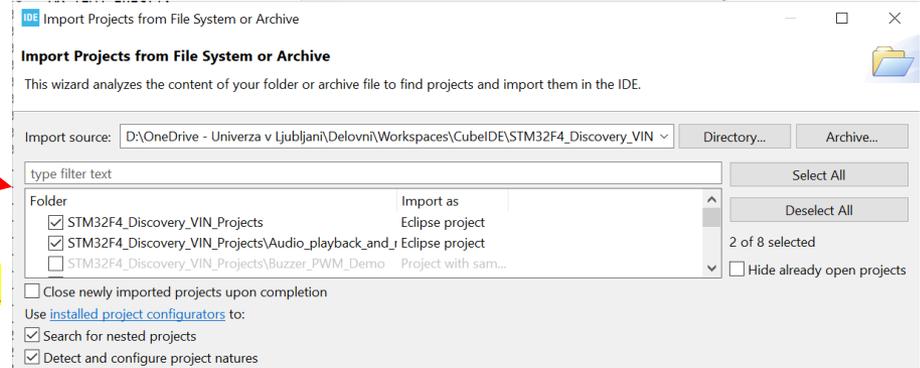
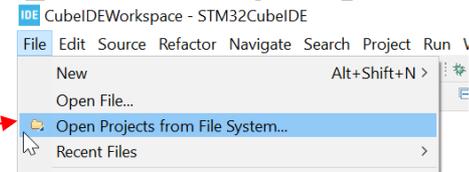
- **Uvoz obstoječega (npr. BSP)**
 - Open projects from File System
 - Select project(s)

1 - Import

Vsi ostali koraki enaki kot pri novem projektu

Prevajanje, zagon :

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



Naslednja LAB vaja

- TinkerCad.com (naredite si račun):
 - <https://www.tinkercad.com/>

