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

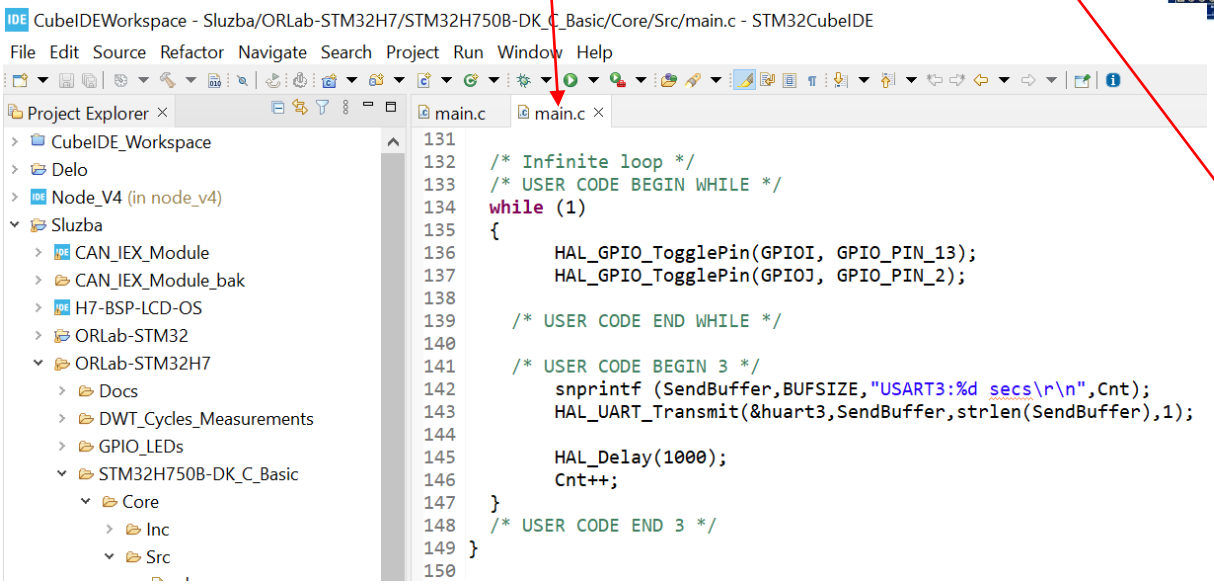
Mikro USB priklop na daljši stranici (srednji !!!)

Priključitev :

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

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

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



Lastni viri :

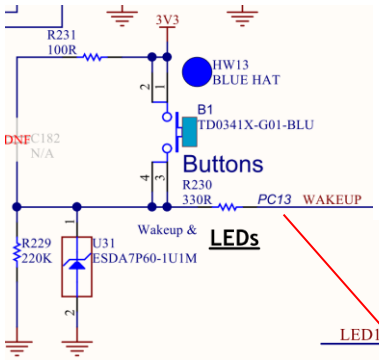
https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects

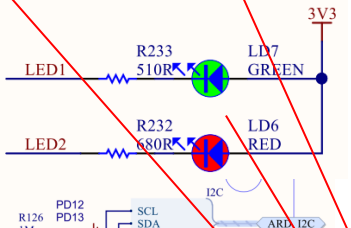
https://github.com/LAPSyLAB/ORLab-STM32H7

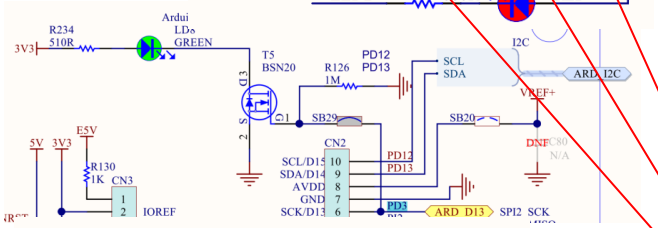
STM32H7

STM32H7

GPIO Krmilnik



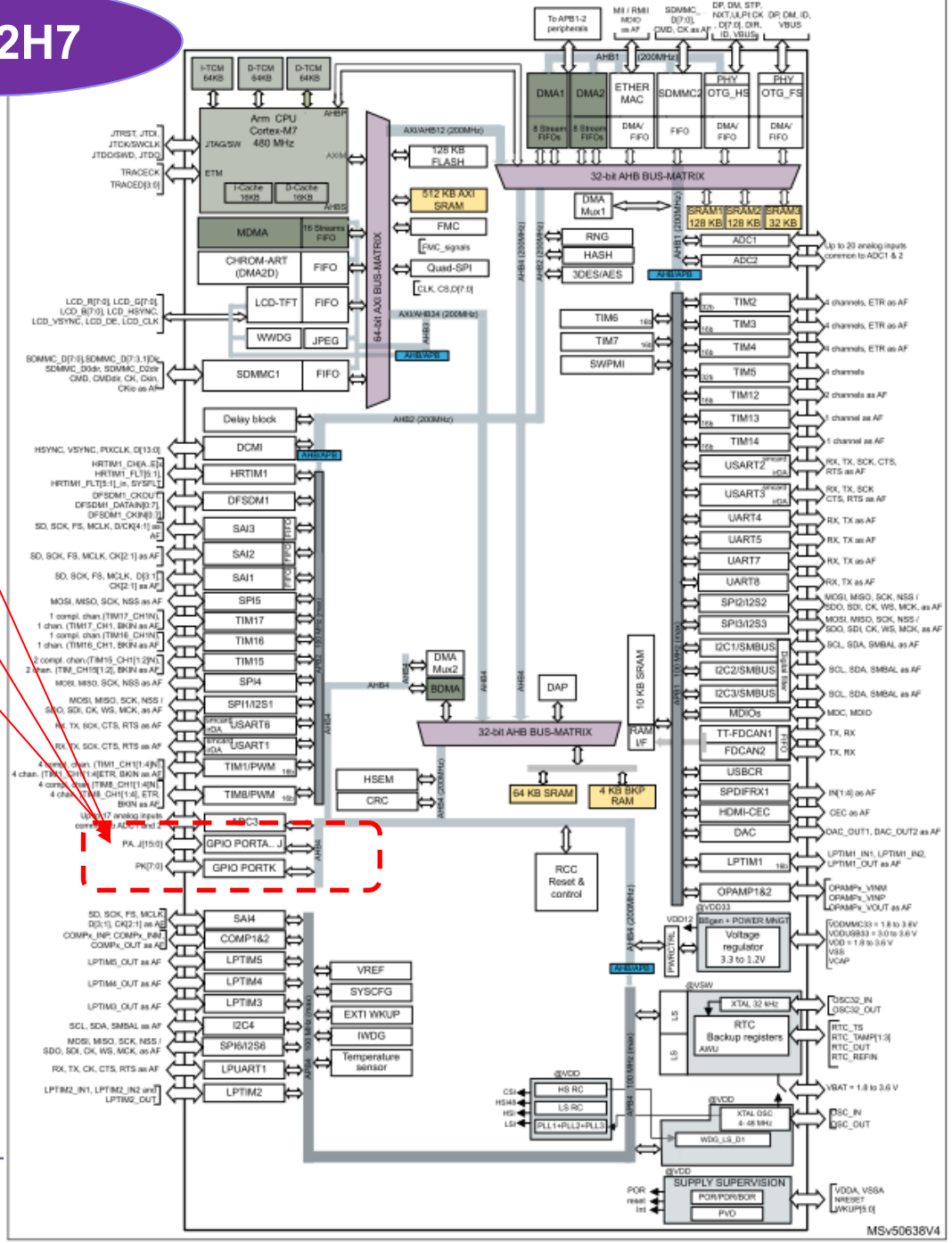




U6B
STM32H750XBH6

T8	PG0	PI0	A16	LCD_G5
U8	PG1	PI1	A15	LCD_G6
H16	PG2	PI2	B15	ARD_D12
H15	PG3	PI3	C14	STMOD#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	STMOD#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		U6	LCD_R4

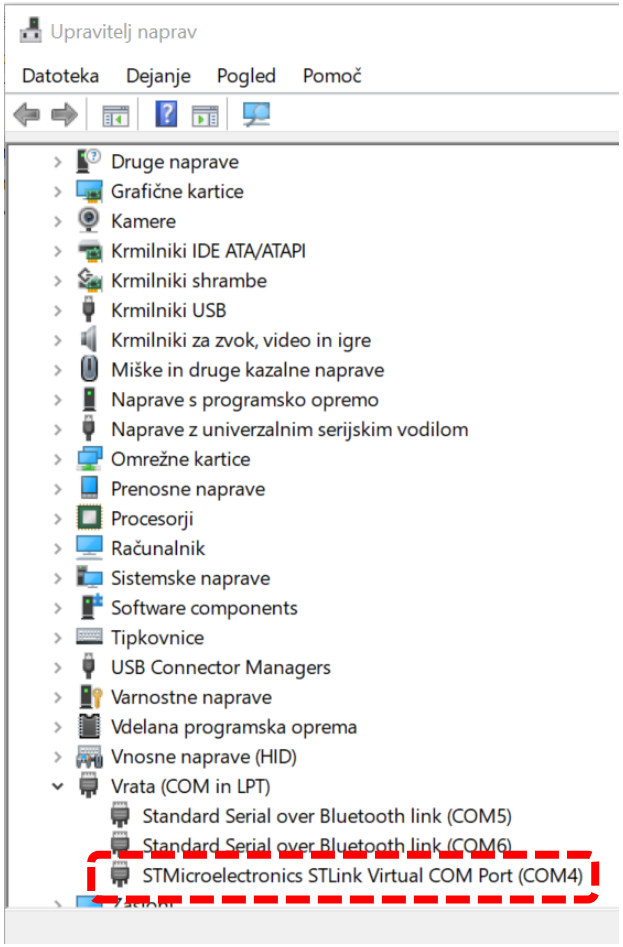
LED: rdeča PI13, zelena PJ2
zelena PD3



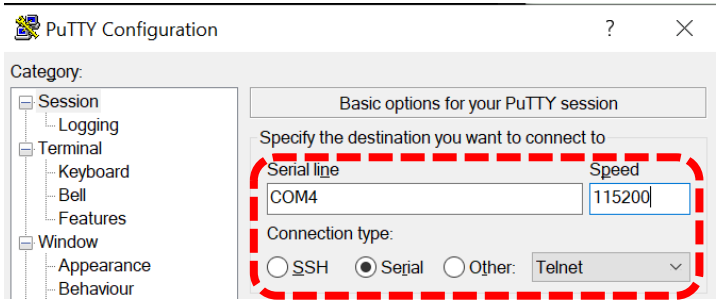
VIN - LV

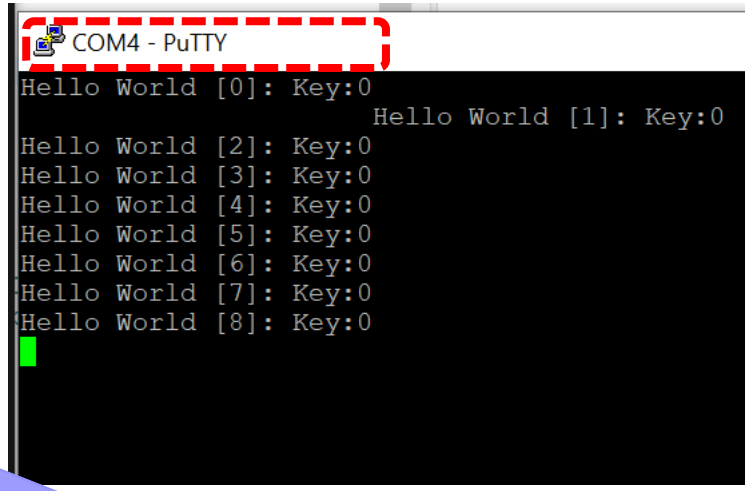
Osnovni projekt CubeIDE – USB Virtual COM Port (USART3 na STM strani)

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



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





10 – Test project

Osnovni projekt CubeIDE II – GPIO – tipka, LED diode

HAL - C

```
/* 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); // Negiraj dig. izhod za rdečo LED diodo
HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_3); // Negiraj dig. izhod za zeleno LED diodo

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

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

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

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

GPIO

Delay

izziv – kako uskladiti, uravnotežiti?

USART COM Port

UM2217

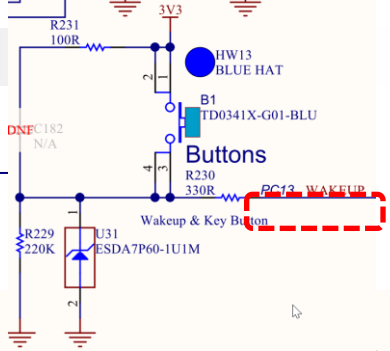
User manual

Description of STM32H7 HAL and low-layer drivers

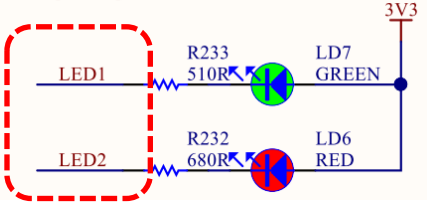
35.2.4 **IO operation functions**

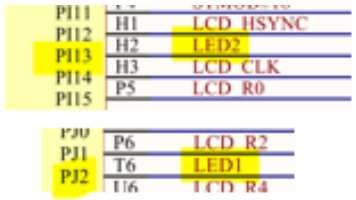
This section contains the following APIs:

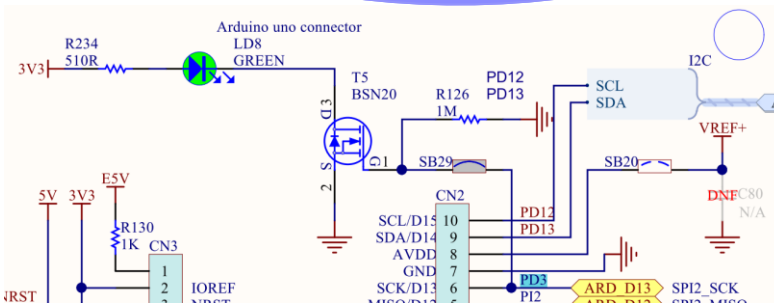
- *HAL_GPIO_ReadPin()*
- *HAL_GPIO_WritePin()*
- *HAL_GPIO_TogglePin()*
- *HAL_GPIO_LockPin()*
- *HAL_GPIO_EXTI_IRQHandler()*
- *HAL_GPIO_EXTI_Callback()*



LEDs







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

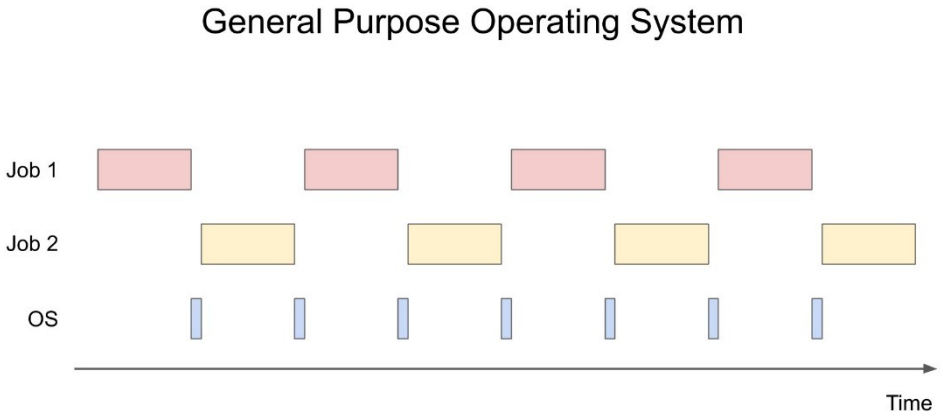
Merilnik kakovosti zraka

RTOS
(ločeni procesi)

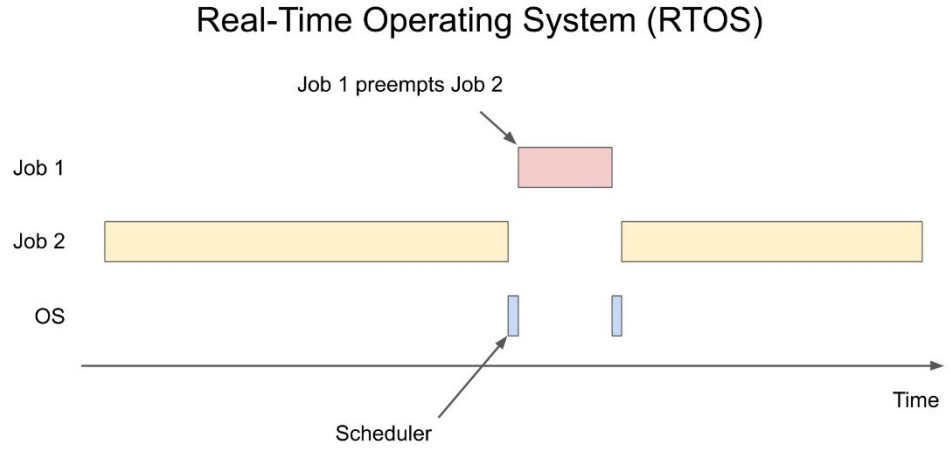
```
void StartTask02(void *argument)
{
  /* USER CODE BEGIN StartTask02 */
  /* Infinite loop */
  for(;;)
  {
   HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
    osDelay(1000);
  }
  /* USER CODE END StartTask02 */
}

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

OS



RTOS



Splošno o RTOS - Real Time Operating System

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

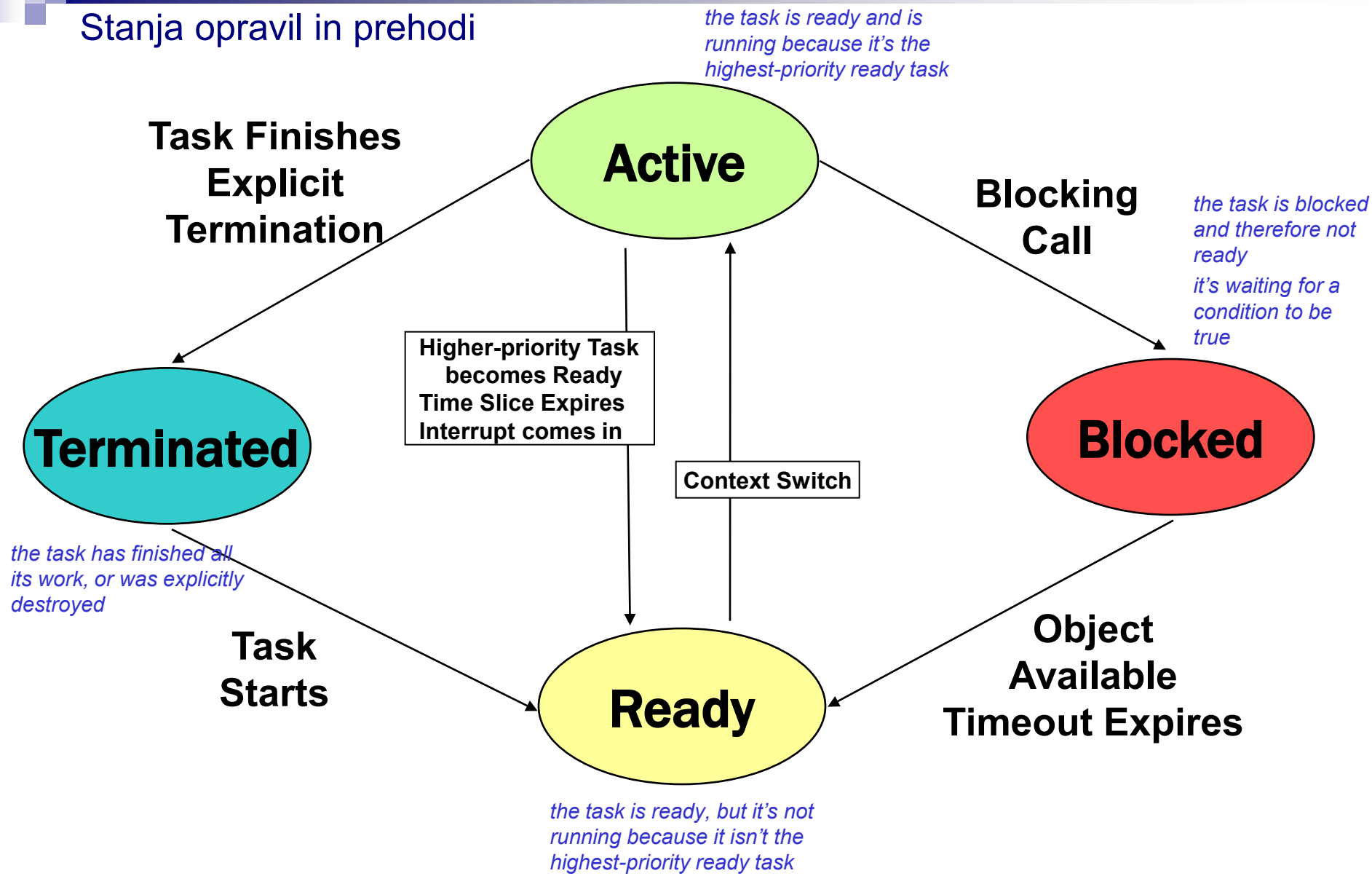
- Funkcionalnosti RTOS:
  - Večopravilnost (multi-tasking)
  - Dodeljevanje opravil CPE s prioritetami
  - Sinhronizacija dostopov do virov:
    - V/I naprav
    - Pomnilnika (podatkovnih struktur)
  - Komunikacija med procesi (Inter-task communication)
  - Časovna predvidljivost (realno-časovna odzivnost)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

Zakaj uporabiti RTOS?

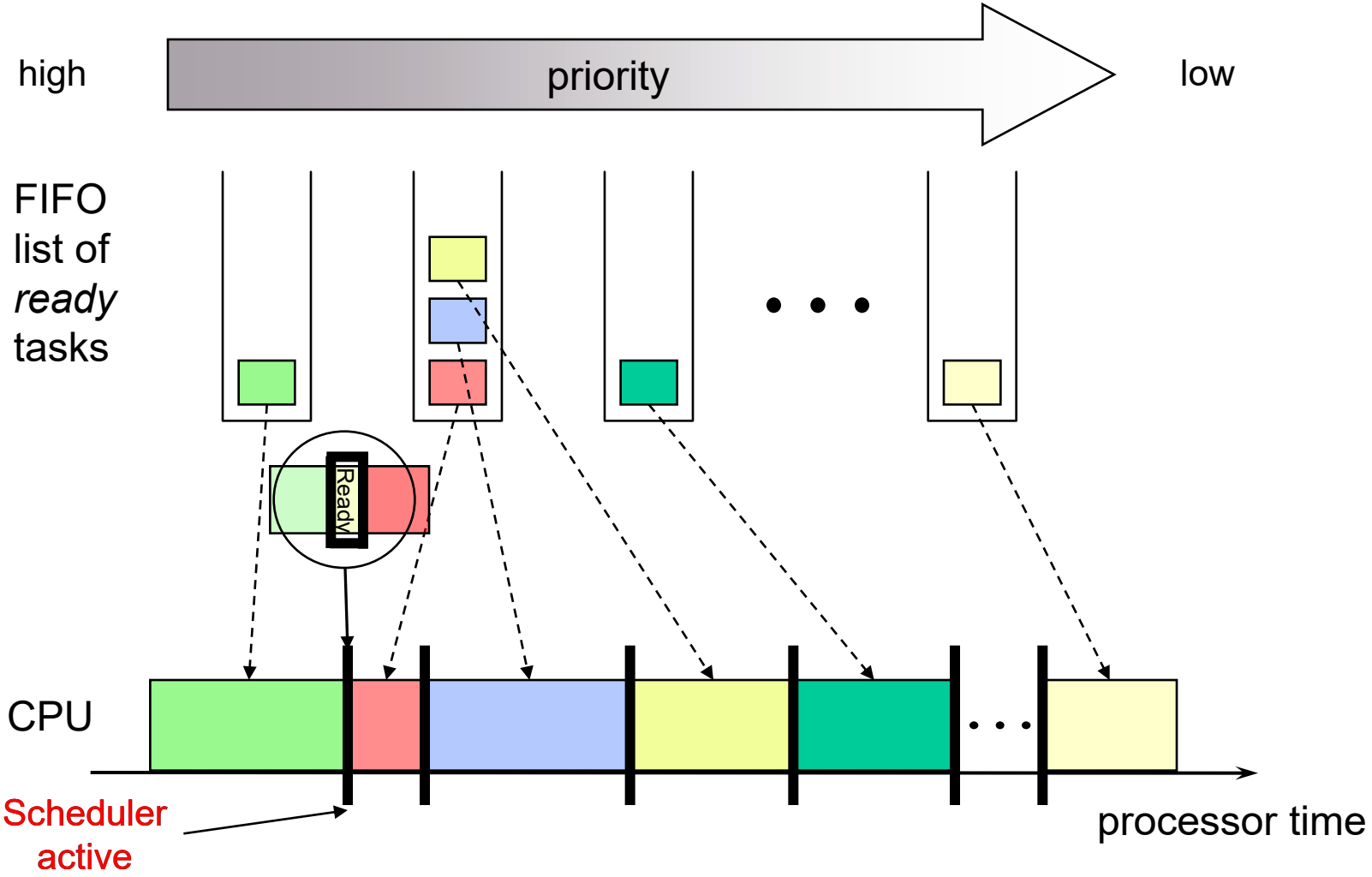
- Uporaba V/I naprav (že pripravljeni driverji (TCP, ETH, CANBUS,…)
- Se splača vse razviti iz nič (npr. svoj dodeljevalnik) ?
  - Diploma : Fabčič – 2021 – lasten RTOS „from scratch“
- Večopravilnost z možnostjo sinhronizacije
- Prenosljivost kode na druge CPE
- Upravljanje z viri
- Možnost dopolnitve 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…

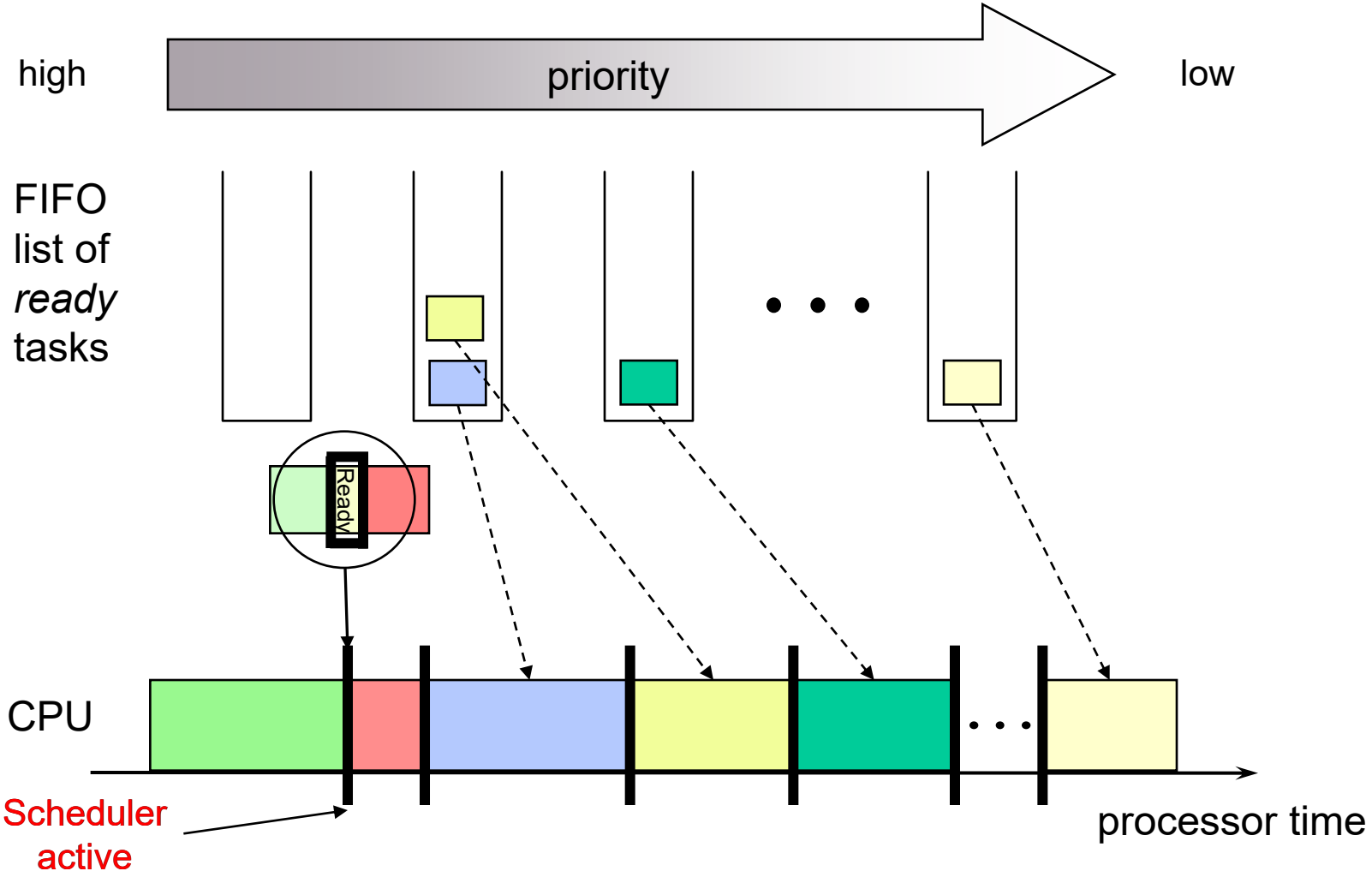
Stanja opravil in prehodi



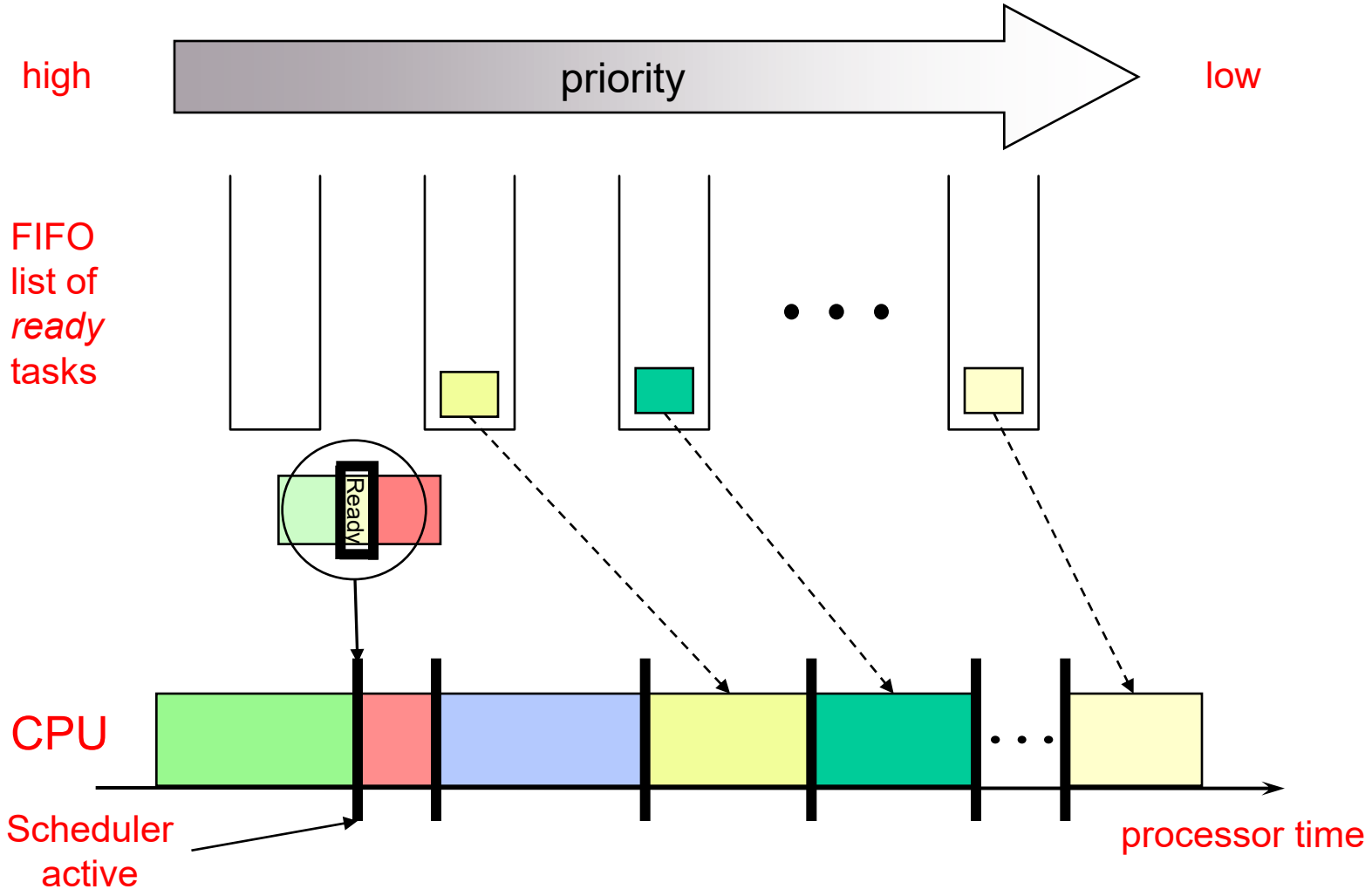
Priority Based FIFO Scheduling



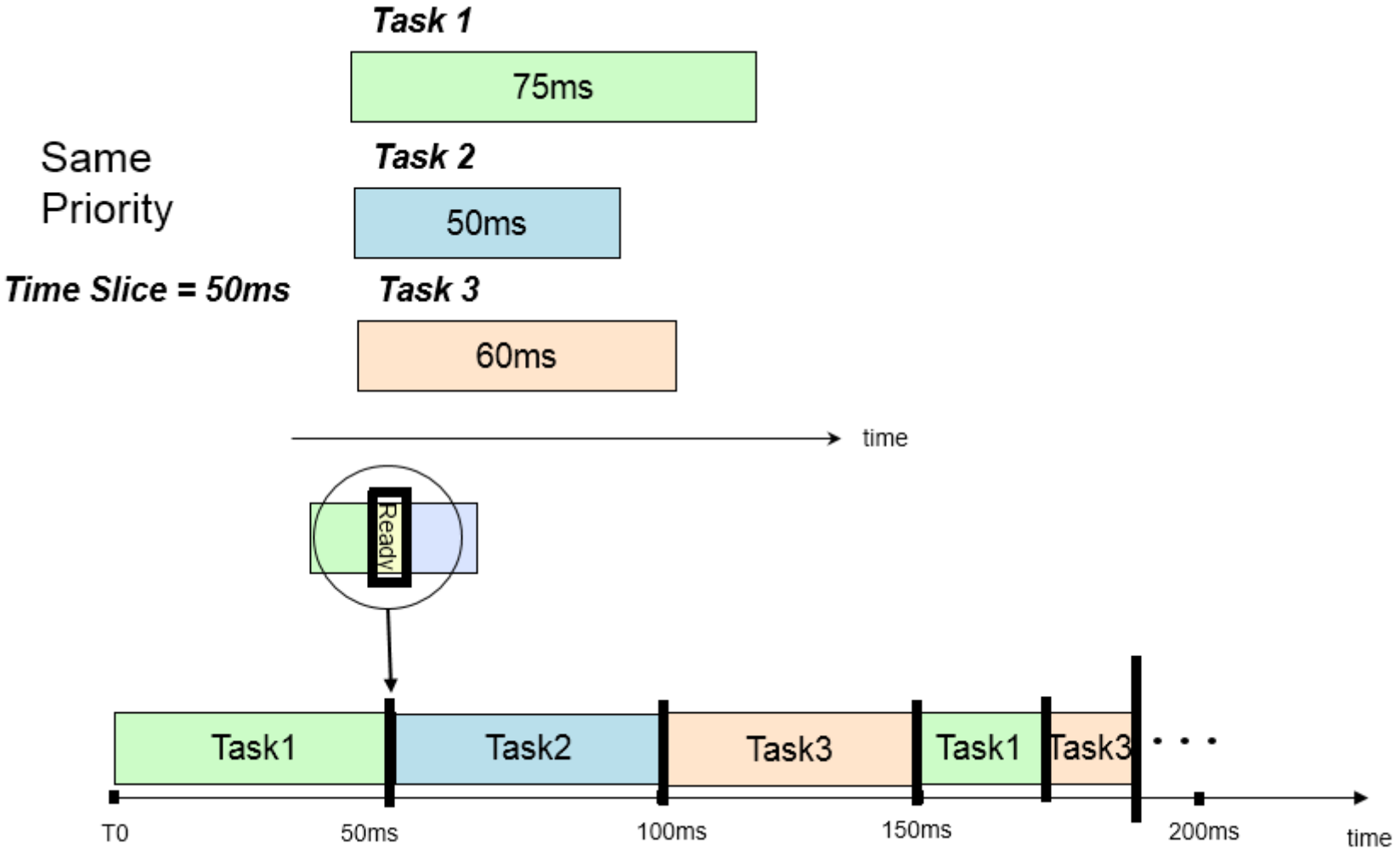
Priority Based FIFO Scheduling



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

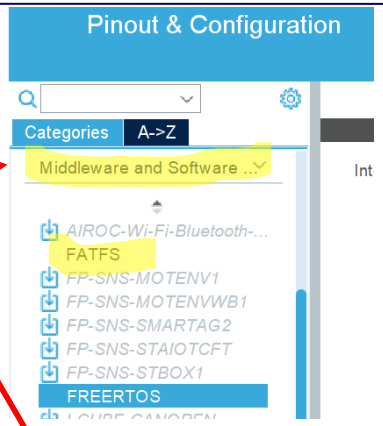
```
void mytask(uint_32 startup_parameter)  {
   /* Task initialization code */
   ....
   while (1) {
      /* Task body */
      ....
      ....
   }

}
```

```
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 */
}
```

Osnovni projekt CubeIDE II – GPIO – tipka, LED diode in FreeRTOS I

Dodatek FreeRTOS:

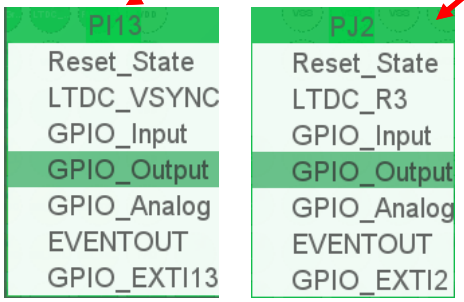
1. „Middleware and SW packs“
2. FreeRTOS activate (CMSIS V2)
3. Kreiraj opravila (tasks):
    1. LED Blinking task
    2. KEY & LED task
    3. defaultTask (že obstaja)

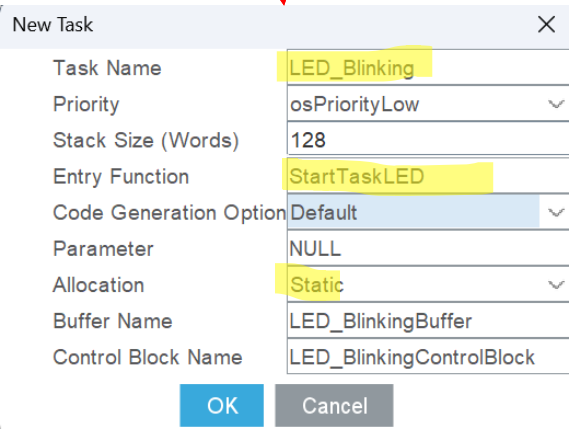


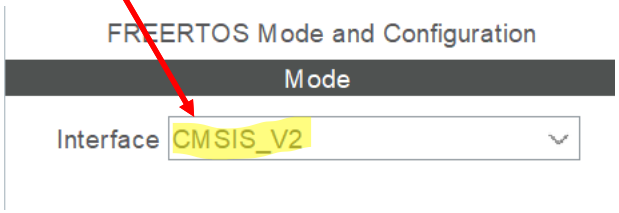
GPIO – GPIO_Output za LED:

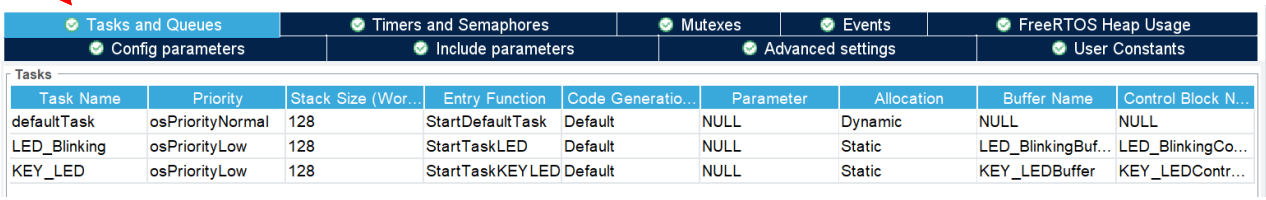
LED: rdeča PI13, zelena PJ2

zelena PD3









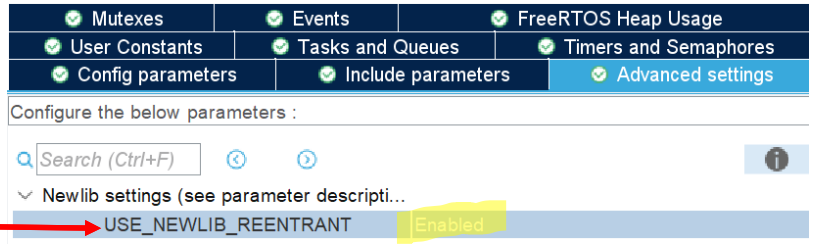
https://github.com/LAPSyLAB/STM32H7_Discovery_VIN_Projects/tree/main/STM32H750B-DK_VIN_Basic_FreeRTOS

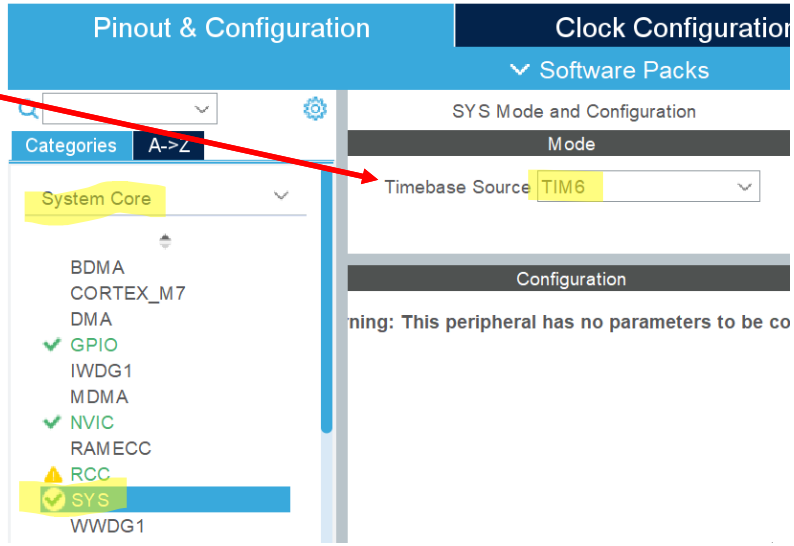
Osnovni projekt CubeIDE II – GPIO – tipka, LED diode in FreeRTOS II

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)





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

Osnovni projekt CubeIDE II – GPIO – tipka, LED diode in FreeRTOS III

Dodatek FreeRTOS:

4. Koda za opravila
5. Compile & Debug

```
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 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 */
}
```

`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)

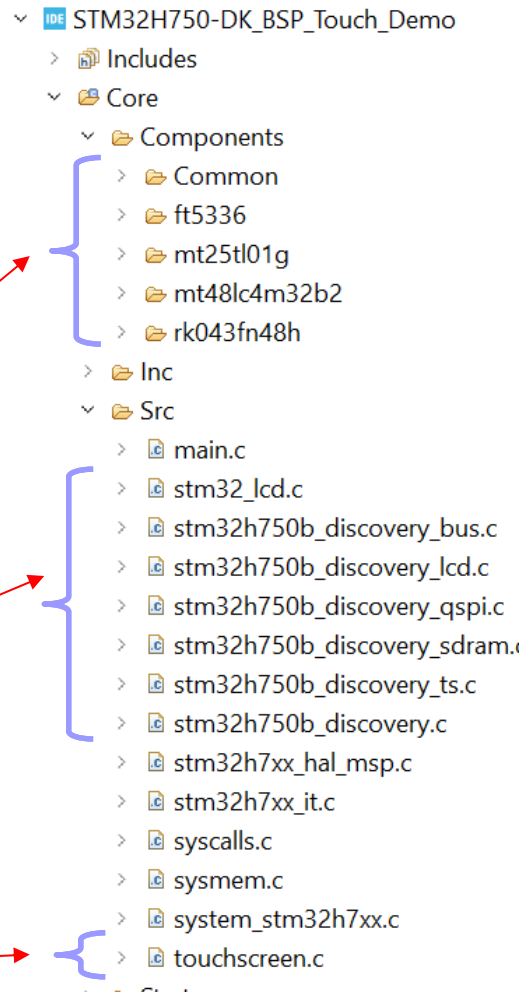
When you call `osDelay(100)` inside a task, you are telling the RTOS scheduler: *"I don't need to do anything for the next 100 OS ticks. Put me to sleep and let someone else use the CPU."*

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

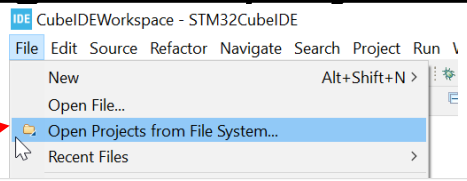
CubeIDE – Uvoz obstoječega BSP projekta

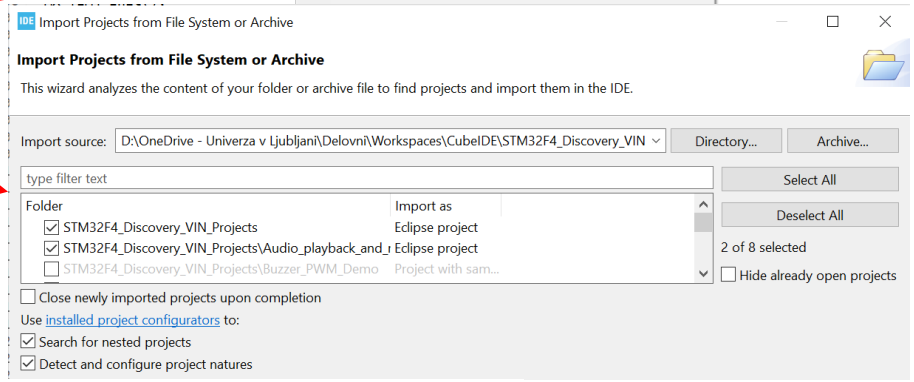
Vzpostavitev začetnega projekta :

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

1 - Import

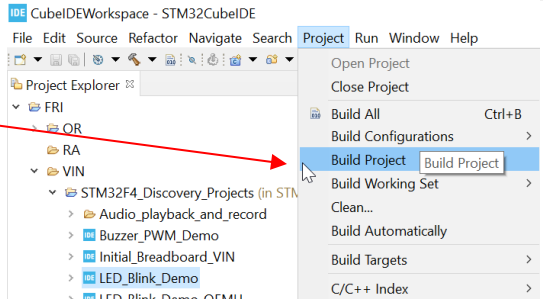
Vsi ostali koraki enaki kot pri novem projektu

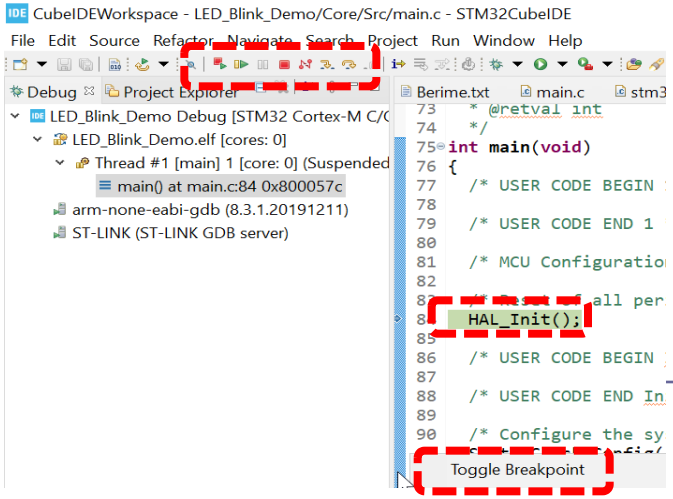


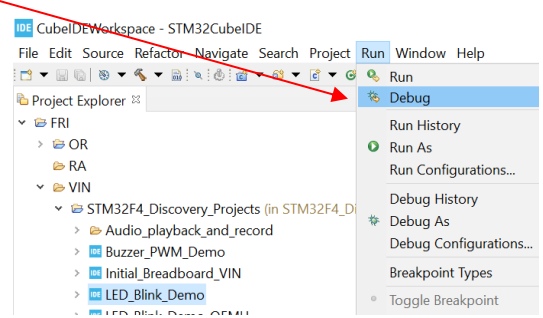


Prevajanje, zagon :

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







Naslednja LAB vaja

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