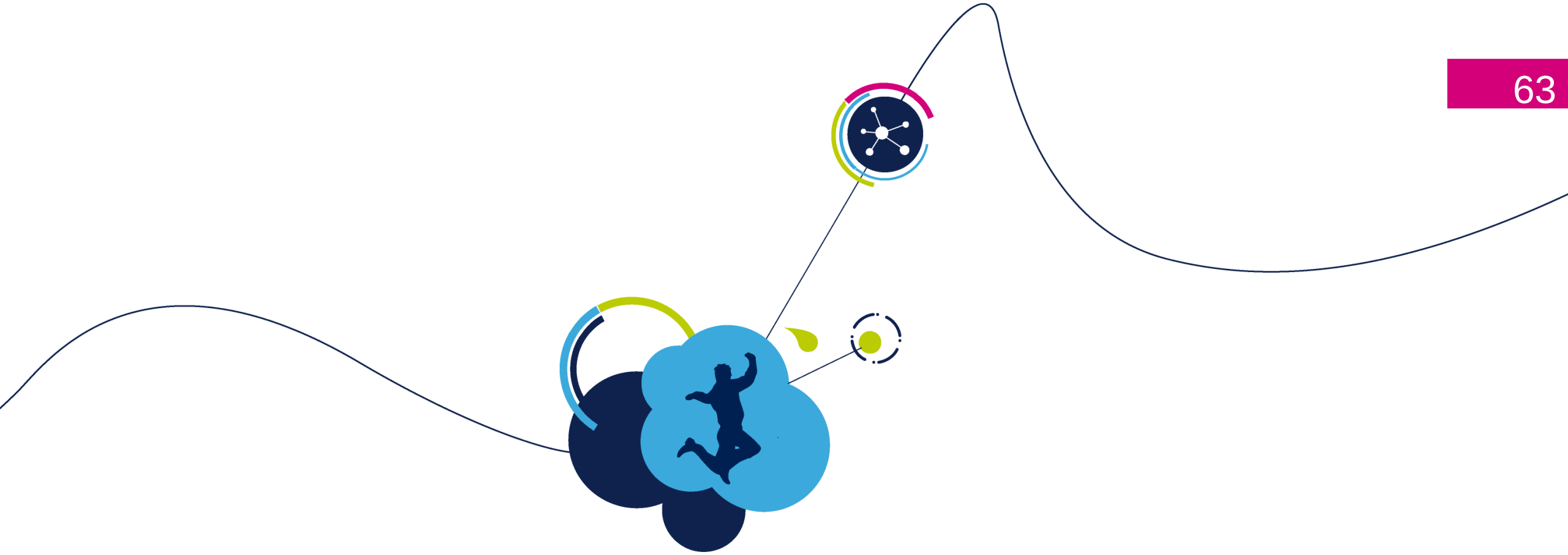


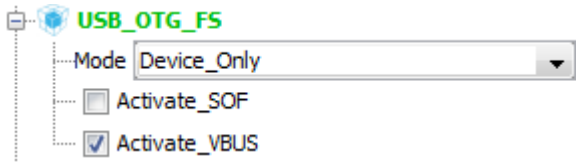
USB HID Low Power Device lab

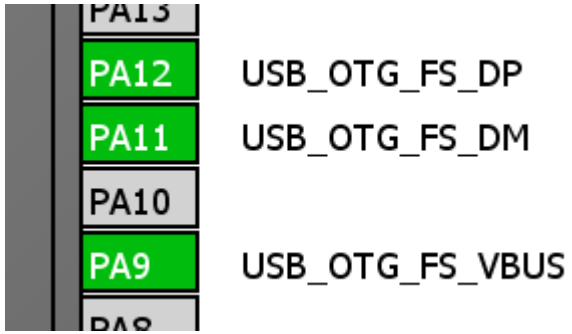
USB HID Low Power Device lab

- Demonstration of STM32 capability to work in low power modes with USB peripheral
- USB suspend remote wakeup
- VBUS sensing feature demonstration

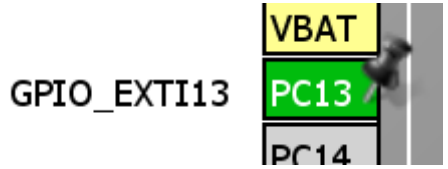
USB HID Low Power Device lab

- Create project in CubeMX, configuration is the same like for VCP device
  - Menu > File > New Project
  - Select STM32F4 > STM32F446 > LQFP144 > STM32F446ZETx
- Select USB FS OTG in device mode
  - Set Activate_VBUS option





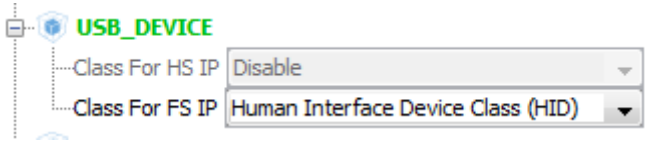
- Select HSE clock
  - (Bypass HSE from STlink)
- Configure PC13 as GPIO exti



life.augmented

USB HID Low Power Device lab

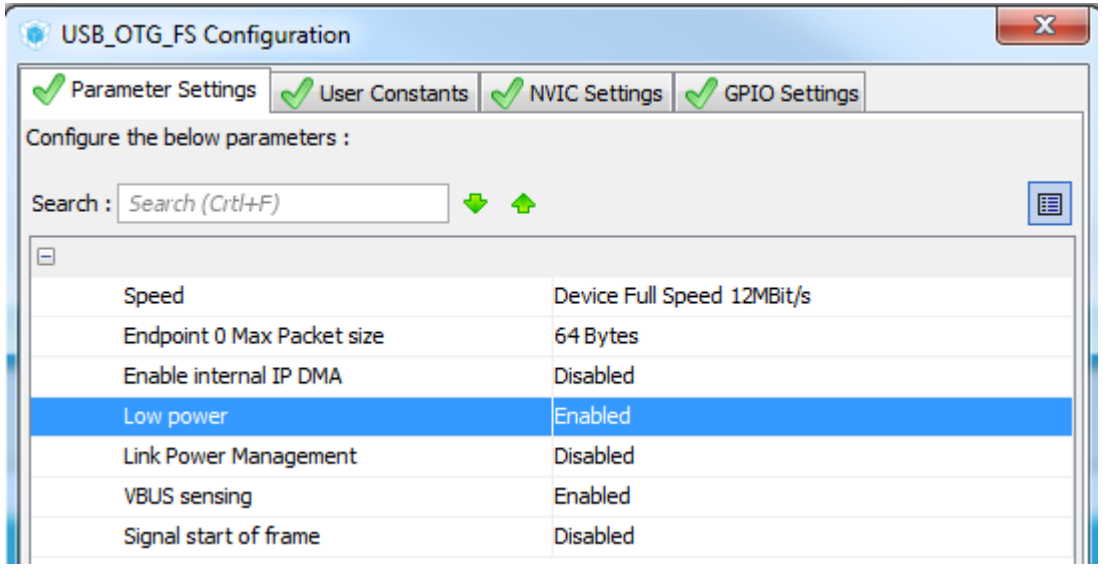
- Configure GPIOs connected to LEDs as GPIO output – PB0, PB7 and PB14
- Select HID class in MiddleWares



- Configure RCC clocks
  - Set 8 MHz HSE as PLL input
  - Set HCLK frequency 168 MHz
  - PLL parameters will be computed automatically

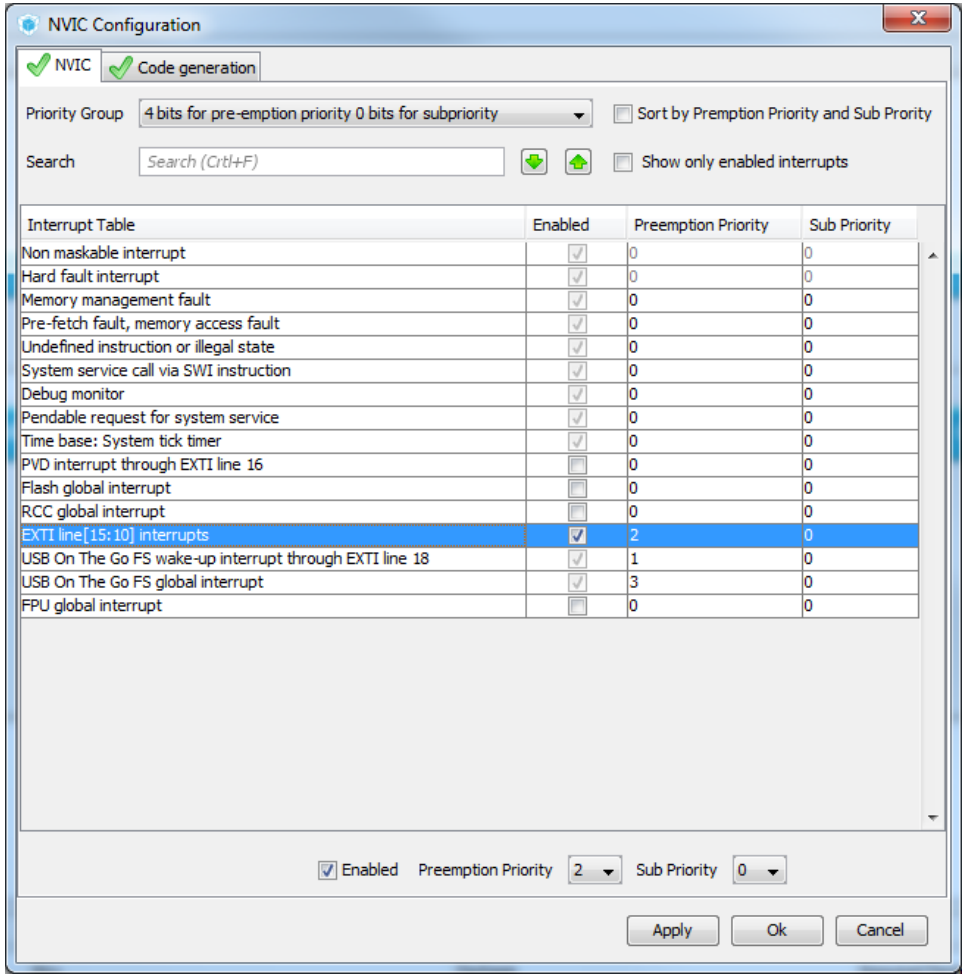
USB HID Low Power Device lab

- In tab configuration -> USB_FS configuration Enable Low power option



USB HID Low Power Device lab

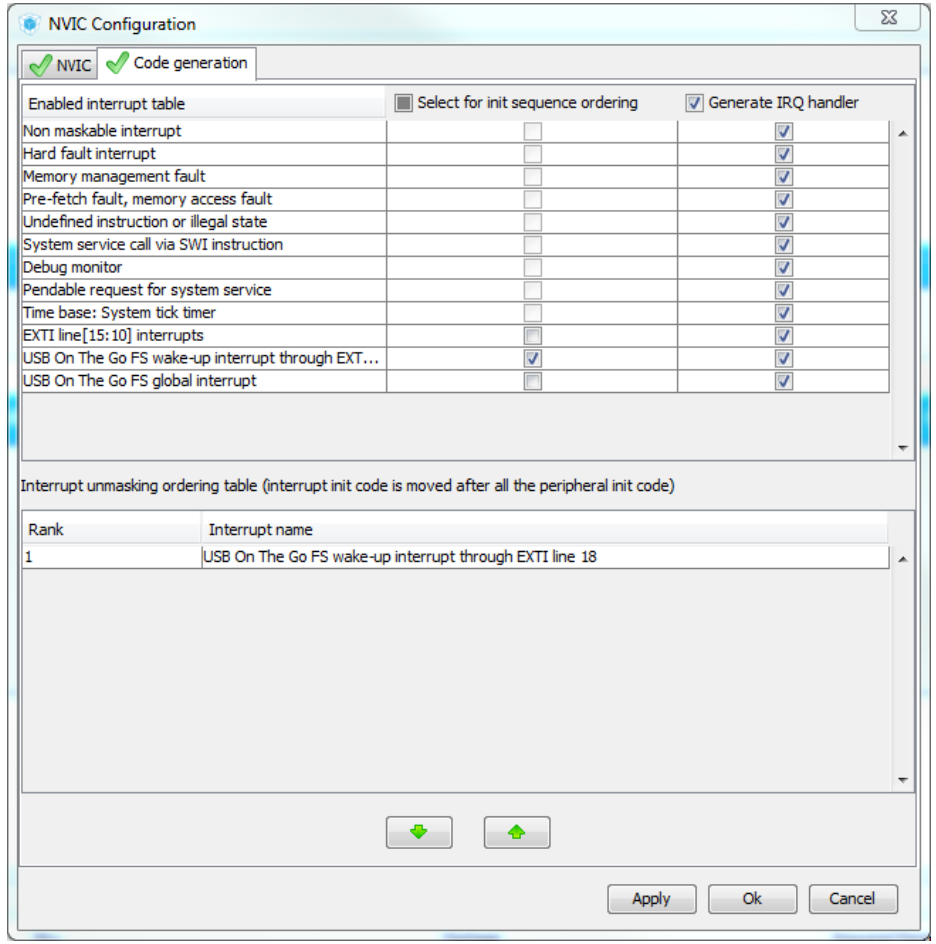
- In tab configuration -> NVIC configuration decrease USB interrupt priority to be lower than System tick timer
  - HAL_Delay is used in button interrupt routine
- Enable EXTI line interrupt



life.augmented

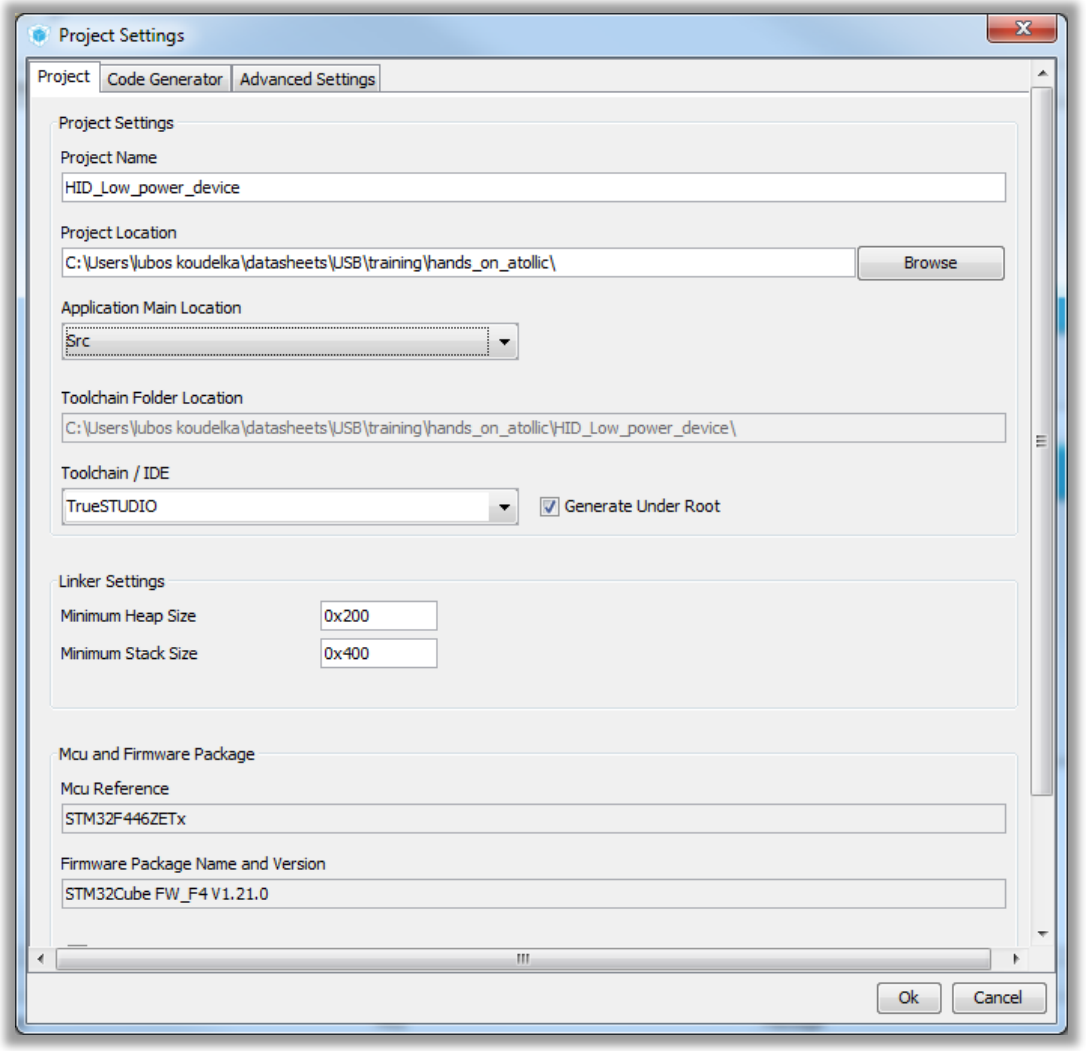
USB HID Low Power Device lab

- Add sequence ordering for USB wake-up interrupt
  - This interrupt will be enabled after complete USB peripheral initialization
- This setting is inside Configuration -> NVIC, Code generation tab



USB HID Low Power Device lab

- Now we set the project details for generation
  - Menu > Project > Project Settings
  - Set the project name
  - Project location
  - Type of toolchain
- Linker Settings
  - For HID class default Heap size is sufficient
- Now we can Generate Code
  - Menu > Project > Generate Code



USB HID Low Power Device lab

- First part of SW change will be the same as for simple HID device:
- And include hid header file

```
/* USER CODE BEGIN Includes */
#include "usbd_hid.h"
/* USER CODE END Includes */
```

- Define buffer which will be send to the host and state variable

```
/* USER CODE BEGIN PFP */
uint8_t buffer[4];
uint8_t i=0;
/* USER CODE END PFP */
```

USB HID Low Power Device lab

- USBD_HID_SendReport will be sent periodically, button will be spared for wakeup functionality

```
while (1) {
	/* USER CODE END WHILE */
	/* USER CODE BEGIN 3 */
	if ((&hUsbDeviceFS)->dev_state == USBD_STATE_CONFIGURED
			&& (&hUsbDeviceFS)->pClass != NULL) {
		USBD_HID_SendReport(&hUsbDeviceFS, buffer, 4);
		i++;
		if (i == 10) {
			buffer[1] = -30;
		} else if (i == 20) {
			buffer[1] = 30;
			i = 0;
		}
	}
	HAL_Delay(500);
	HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_7);
}
```

USB HID Low Power Device lab

- Add button callback to usbd_conf.c – wake up MCU from low power mode and send wake up signal on the bus

```
/* USER CODE BEGIN 0 */
uint8_t button_wakeup=0;
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  if (GPIO_Pin == GPIO_PIN_13)
  {
    if ((((USBD_HandleTypeDef *) hpcd_USB_OTG_FS.pData)->dev_remote_wakeup == 1) &&
        (((USBD_HandleTypeDef *) hpcd_USB_OTG_FS.pData)->dev_state == USBD_STATE_SUSPENDED))
    {
      if ((&hpcd_USB_OTG_FS)->Init.low_power_enable)
      {
        /* Reset SLEEPDEEP bit of Cortex System Control Register */
        SCB->SCR &=
          (uint32_t) ~
          ((uint32_t) (SCB_SCR_SLEEPDEEP_Msk | SCB_SCR_SLEEPONEXIT_Msk));

        SystemClock_Config();
      }
      button_wakeup=1;
      …
```

USB HID Low Power Device lab

- Add button callback to usbd_conf.c – wake up MCU from low power mode and send wake up signal on the bus

```
      …
      /* Ungate PHY clock */
      __HAL_PCD_UNGATE_PHYCLOCK((&hpcd_USB_OTG_FS));
      /* Activate Remote wakeup */
      HAL_PCD_ActivateRemoteWakeup((&hpcd_USB_OTG_FS));
      /* Remote wakeup delay */
      HAL_Delay(10);
      /* Disable Remote wakeup */
      HAL_PCD_DeActivateRemoteWakeup((&hpcd_USB_OTG_FS));

      /* change state to configured */
      ((USBD_HandleTypeDef *) hpcd_USB_OTG_FS.pData)->dev_state =
USBD_STATE_CONFIGURED;

      /* Change remote_wakeup feature to 0 */
      ((USBD_HandleTypeDef *) hpcd_USB_OTG_FS.pData)->dev_remote_wakeup = 0;
    }
  }
}
/* USER CODE END 0 */
```

USB HID Low Power Device lab

- Modify WKUP_IRQHandler – avoid second execution of SystemClock_config() function if wake-up is done by the button
  - clock is already configured in HAL_GPIO_EXTI_Callback, which is also source of wakeup source for this interrupt
  - SystemClock_config function ends in error - incorrect sequence for clock reinit
  - This change won’t be preserved after project regeneration in CubeMX

USB HID Low Power Device lab


```
extern uint8_t button_wakeup;
void OTG_FS_WKUP_IRQHandler(void){
  /* USER CODE BEGIN OTG_FS_WKUP_IRQn 0 */
if (button_wakeup==0)    {
  /* USER CODE END OTG_FS_WKUP_IRQn 0 */
  if ((&hpcd_USB_OTG_FS)->Init.low_power_enable)
 {
    /* Reset SLEEPDEEP bit of Cortex System Control Register */
    SCB->SCR &=(uint32_t)~((uint32_t)(SCB_SCR_SLEEPDEEP_Msk|SCB_SCR_SLEEPONEXIT_Msk));
    SystemClock_Config();
  }
  __HAL_PCD_UNGATE_PHYCLOCK(&hpcd_USB_OTG_FS);
  /* Clear EXTI pending bit */
  /* USER CODE BEGIN OTG_FS_WKUP_IRQn 1 */
    }else{
      button_wakeup=0;
    }
__HAL_USB_OTG_FS_WAKEUP_EXTI_CLEAR_FLAG();
  /* USER CODE END OTG_FS_WKUP_IRQn 1 */
}
```

USB HID Low Power Device lab

- Now the demo can be tested
- As long as device is configured, LEDs are blinking and mouse cursor is moved
- After detecting idle condition, device is switched to suspend mode, STM32 is put into stop mode
  - Disabling of unused peripheral should be done if lowest possible power consumption is intended
- Functionality is restored when wake-up event is detected on the bus or device can produce wake-up condition on button press

USB HID Low Power Device lab

- In the second part of this example will be demonstrated purpose of VBUS sensing
- Without VBUS sensing is not possible to differentiate suspend and disconnect event
- Few more modification in the project is needed now

USB HID Low Power Device lab

- Add include to usbd_conf.c

```
/* USER CODE BEGIN Includes */
#include "stm32f4xx_hal.h"
/* USER CODE END Includes */
```

- Modify HAL_PCD_SuspendCallback
  - For simplification of the example is used blocking delay inside of this callback – more sophisticated methods shall be used in real usage, for example timer interrupt

USB HID Low Power Device lab


```
void HAL_PCD_SuspendCallback(PCD_HandleTypeDef *hpcd) {
USB_OTG_GlobalTypeDef *USBx = hpcd->Instance;

/*delay in interrupt callback -incorrect, but the simplest solution
 Reason for using this interrupt here is, that thanks to capacitor connected to VBUS on nucleo board, suspend
event (3ms) is detected much faster compared to disconnect event
 According to USB specification, power consumption of USB device shall be reduced bellow 2.5 mA for High-power
device (500 uA for Low-power device) within 10 ms afer suspend state detection*/
HAL_Delay(2000);
/* Check if device is still connected to the host */
if (((hpcd->Instance->GOTGCTL) & USB_OTG_GOTGCTL_BSESVLD)== USB_OTG_GOTGCTL_BSESVLD) {
  /* Check if suspend state is still detected on USB bus */
  if ((USBx_DEVICE->DSTS & USB_OTG_DSTS_SUSPSTS) == USB_OTG_DSTS_SUSPSTS) {
    HAL_GPIO_TogglePin(GPIOB, GPIO_PIN_14);
    /* Inform USB library that core enters in suspend Mode */
    USBD_LL_Suspend((USBD_HandleTypeDef*) hpcd->pData);

    __HAL_PCD_GATE_PHYCLOCK(hpcd);
    /* Enter in STOP mode. */
    if (hpcd->Init.low_power_enable) {
      /* Set SLEEPDEEP bit and SleepOnExit of Cortex System Control Register. */
      SCB->SCR |= (uint32_t) ((uint32_t) (SCB_SCR_SLEEPDEEP_Msk| SCB_SCR_SLEEPONEXIT_Msk));
}}}}
```

USB HID Low Power Device lab

- Add led toggling inside disconnect callback to recognize disconnection

```
void HAL_PCD_DisconnectCallback(PCD_HandleTypeDef *hpcd) {
  HAL_GPIO_TogglePin(GPIOB,GPIO_PIN_0);
  USBD_LL_DevDisconnected((USBD_HandleTypeDef*) hpcd->pData);
}
```

- Now is the example ready for test
  - On suspend detection - LD3 toggle and MCU is put to stop mode – LD2 toggling is stopped
  - On disconnect detection LD1 toggle and MCU continue in run mode – LD2 is toggling continuously

USB HID Low Power Device lab

- Now disable in usbd_conf.c

```
hpcd_USB_OTG_FS.Init.vbus_sensing_enable = DISABLE;
```

- Without vbus sensing feature MCU is not able to recognize disconnection, and both disconnect and suspend event are recognized as suspend