STM32L4 technical training

Universal Serial Bus (USB)

Hands-on session

life.augmented

USB Lab 2

USB device in CDC class: MCU – PC communication using VCP (Virtual COM Port)

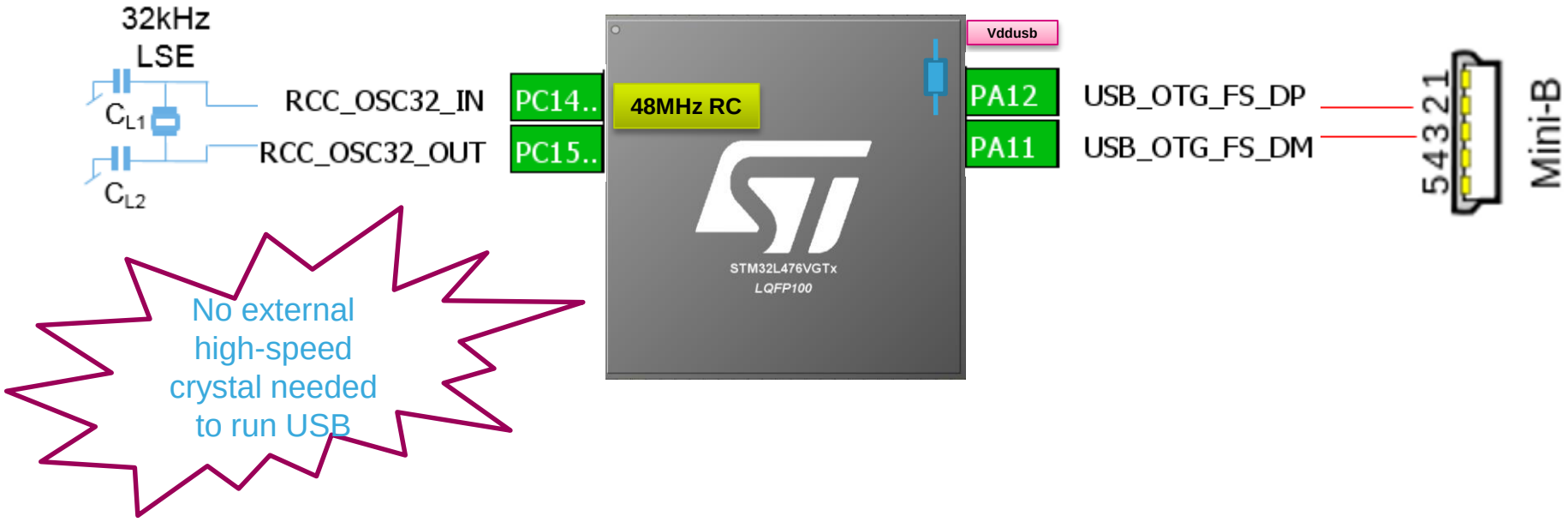
life.augmented

USB VCP communication

- Objective
  - Learn how to design USB hardware with STM32L4
  - Learn how to configure USB device (USB clock and USB CDC class) in CubeMX
  - Learn how to configure joystick (four input GPIOs) in CubeMX
  - Learn how to generate code in CubeMX and use HAL functions
- Method
  - Create a bidirectional USB VCP communication between MCU and PC terminal

User USB hardware connection

- STM32L4 is optimised in terms of BOM for USB connectivity
  - Pull-up resistor is embedded in USB PHY
  - Serial resistors are not needed
  - Internal RC 48MHz (MSI – Multi Speed Internal), which can be used to run USB, after trimming by LSE (Low Speed External)

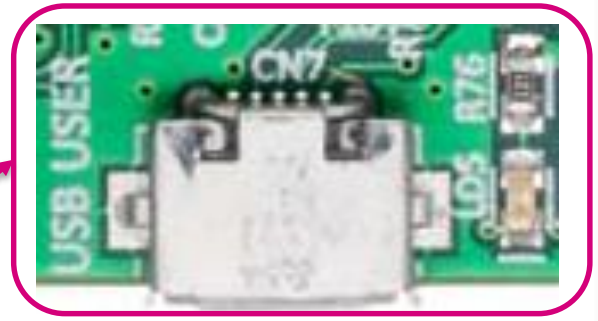


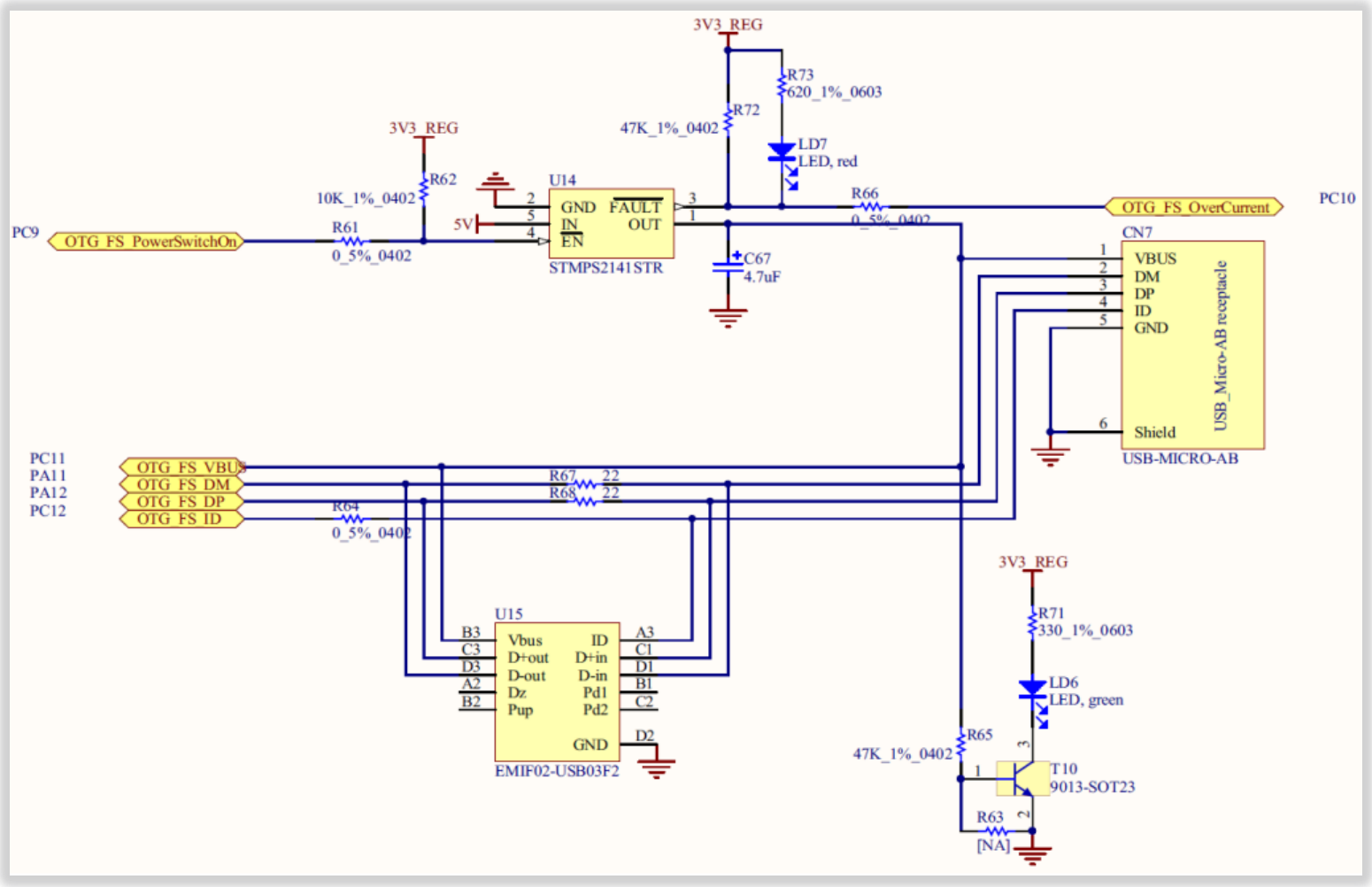
life.augmented

User USB connection

STM32L476RG-Discovery

- STM32F476RG-Discovery is equipped User USB connector. Pins assignment:
  - PA11 (USB D-)
  - PA12 (USB D+)



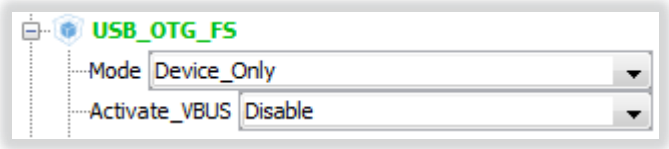


STM32CubeMX

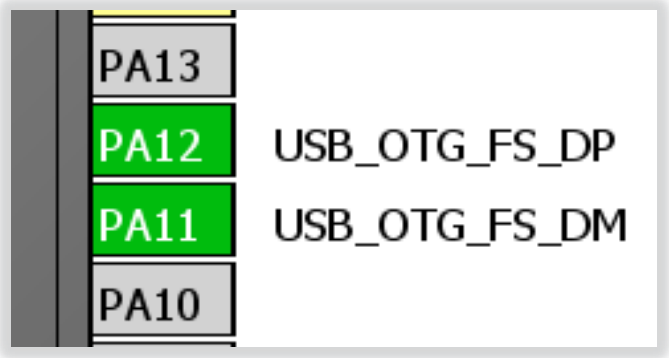
Selecting USB interface and USB class

- Create project in STM32CubeMX
  - Menu > File > New Project
  - Select STM32L4 -> STM32L4x6 -> LQFP100 package -> STM32L476VGTx

- Select USB:
  - Select "**Device_Only**" for **Mode** of **USB_OTG_FS**



  - Select „**Communication Device Class (Virtual Port COM)**" for **Class For FS IP** of **USB_DEVICE**

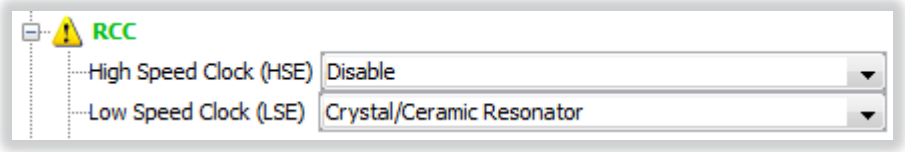


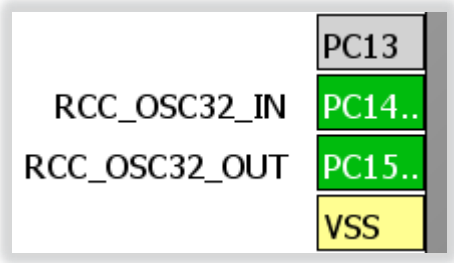
life.augmented

STM32CubeMX

Selecting LSE clock and Joystick buttons

- Select LSE:
  - Select “**Crystal/Ceramic Resonator**” for **Low Speed Clock (LSE)** of **RCC**



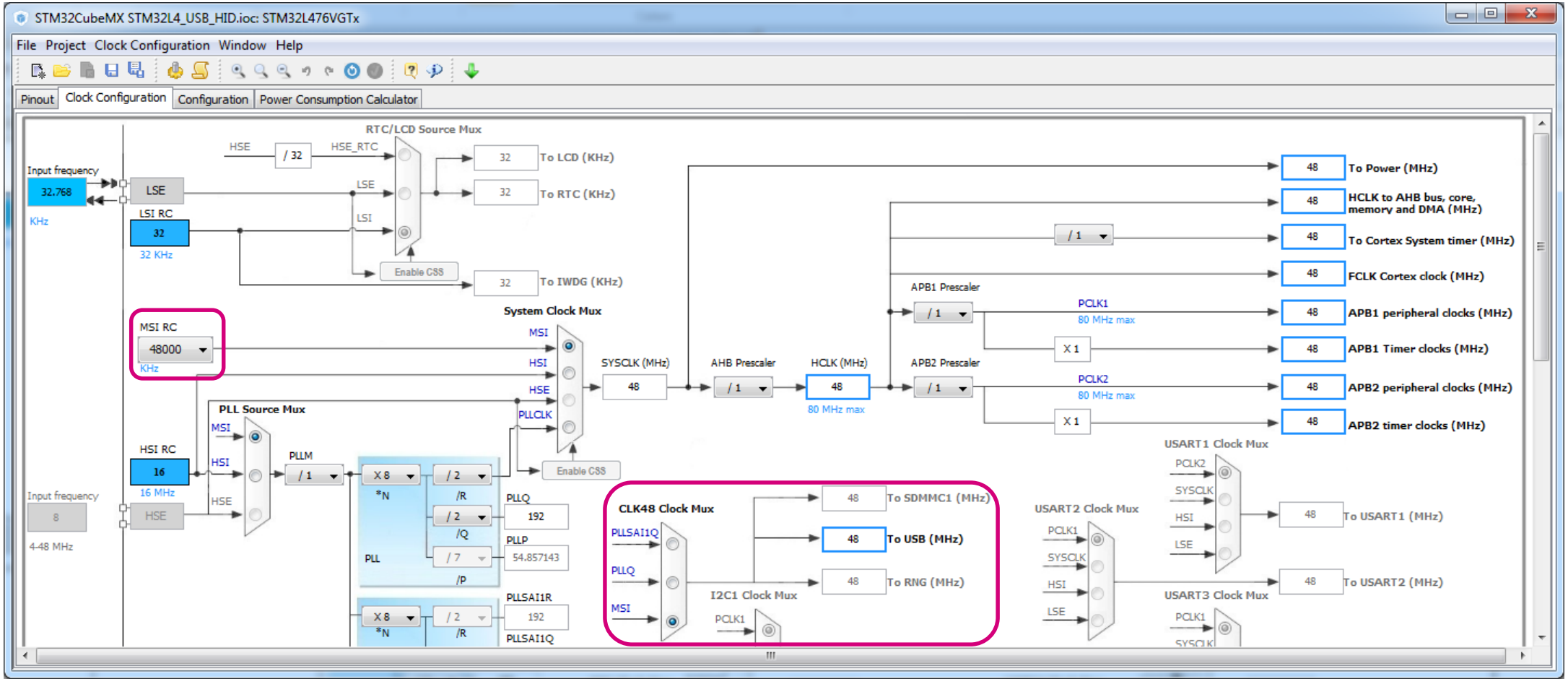


LSE is needed to trimm the MSI

STM32CubeMX

clock configuration

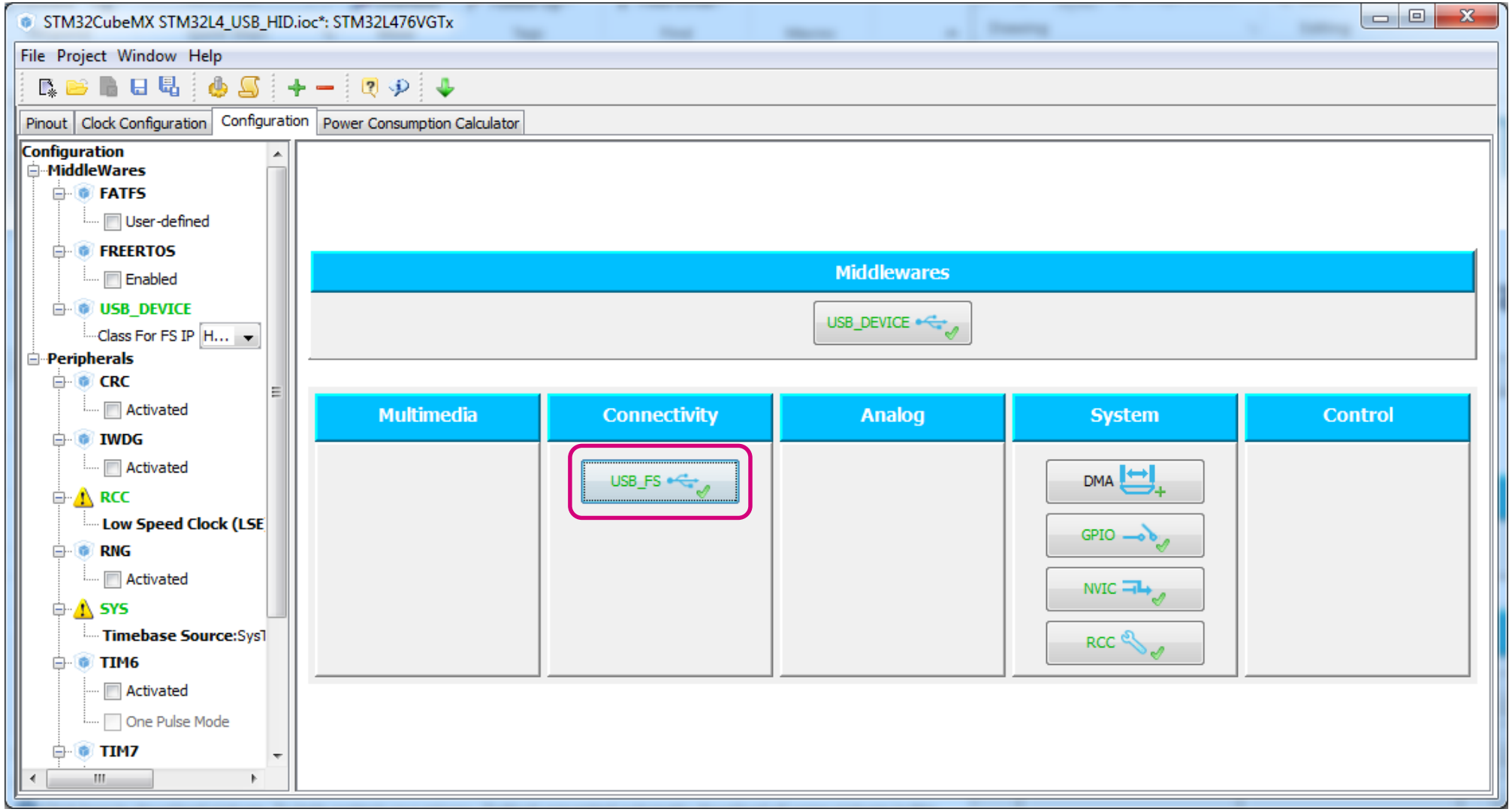
- Go to Clock Configuration tab and configure MCU clock system:
  1. Change MSI default value (4 MHz) to 48 MHz
  2. Select MSI as a clock source for USB



STM32CubeMX

Configure USB

- Go to Configuration tab and select USB peripheral

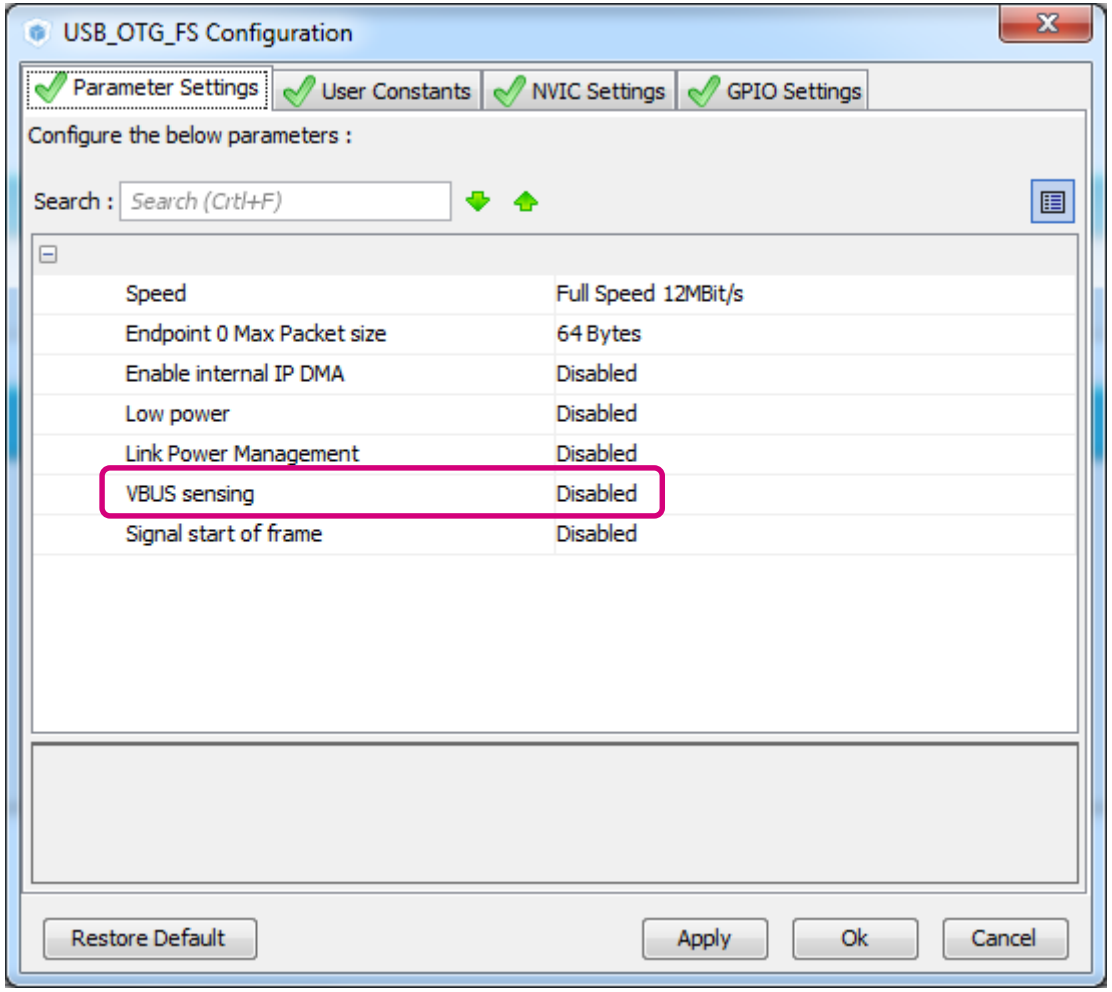


life.augmented

STM32CubeMX

configuration of USB VBUS

- Select Parameter Settings tab
  - Disable VBUS sensing
- Press **Ok** to confirm the configuration

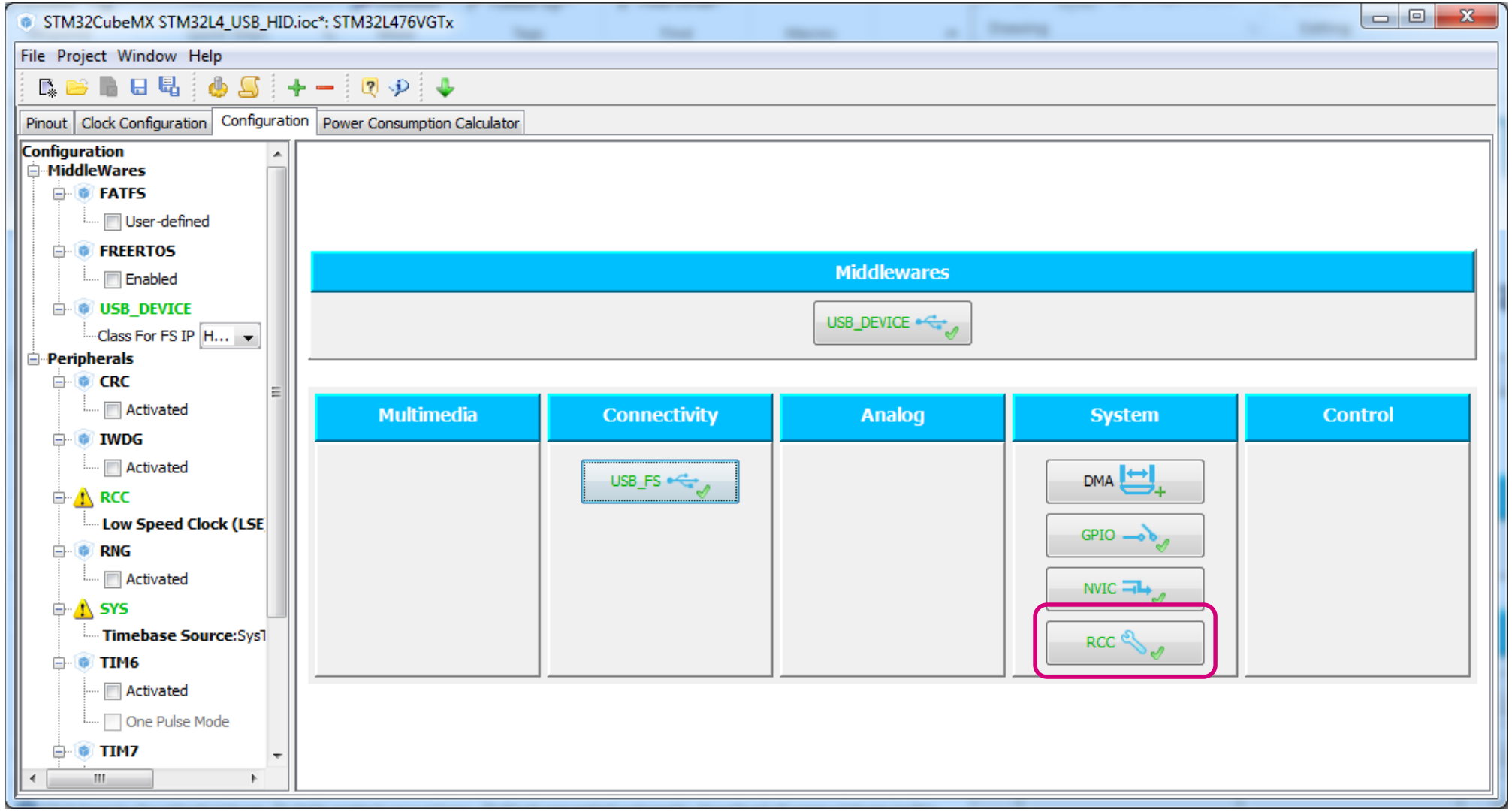


life.augmented

STM32CubeMX

Configure clock

- Go to Configuration tab and select RCC peripheral

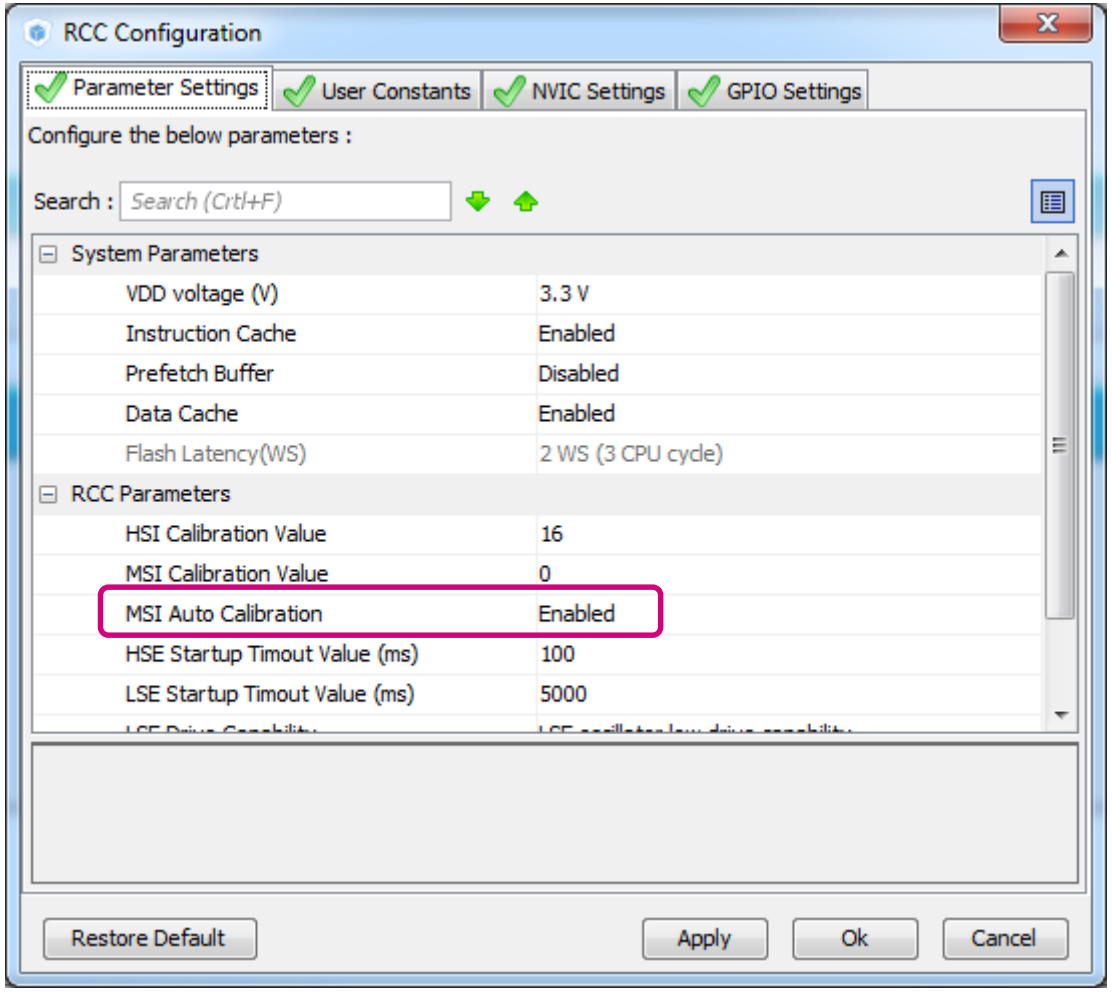


life.augmented

STM32CubeMX

configuration of the MSI calibration with LSE

- Select Parameter Settings tab
  - Enable MSI Auto Calibration
- Press **Ok** to confirm the configuration

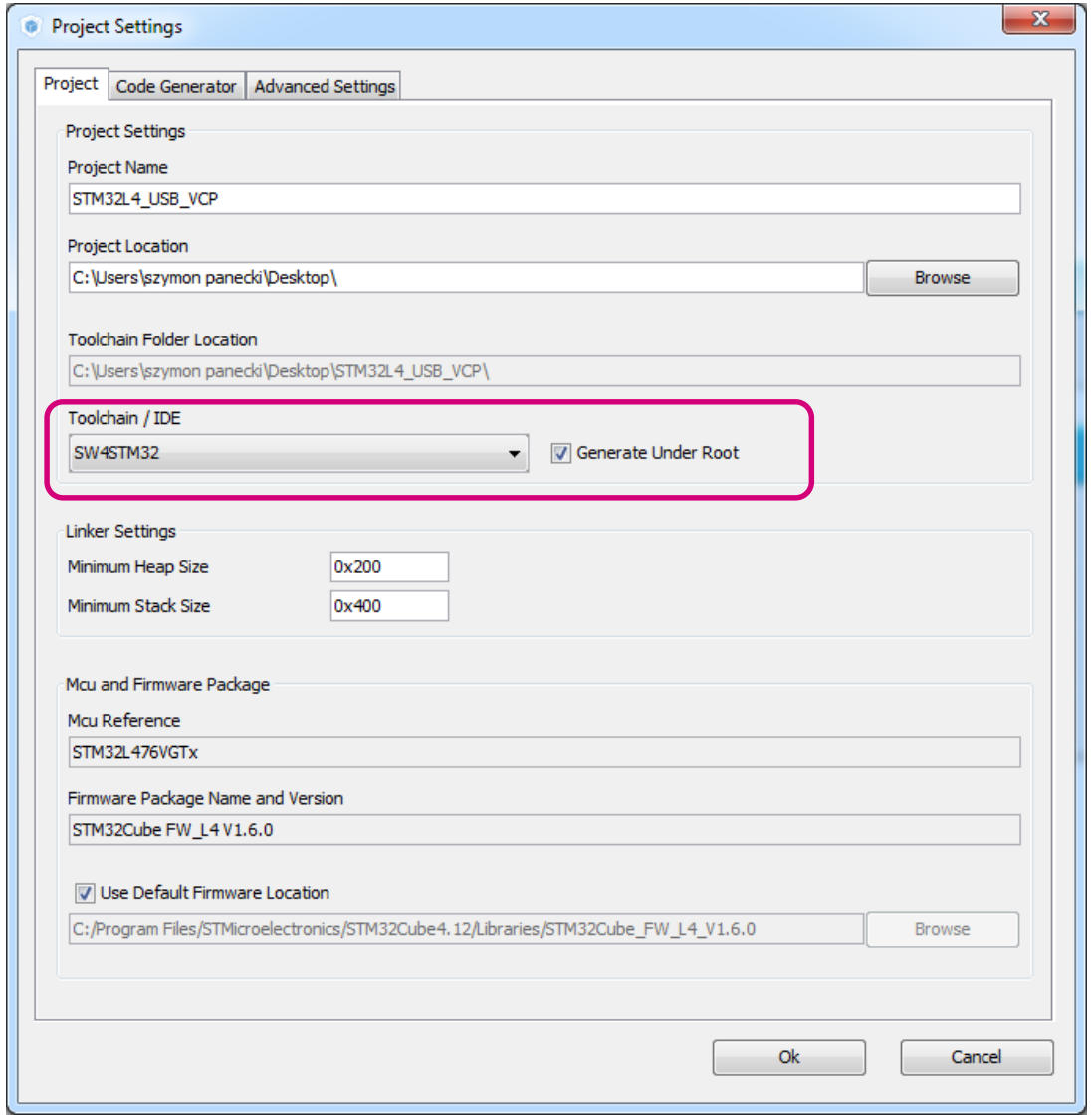


life.augmented

STM32CubeMX

Project generation

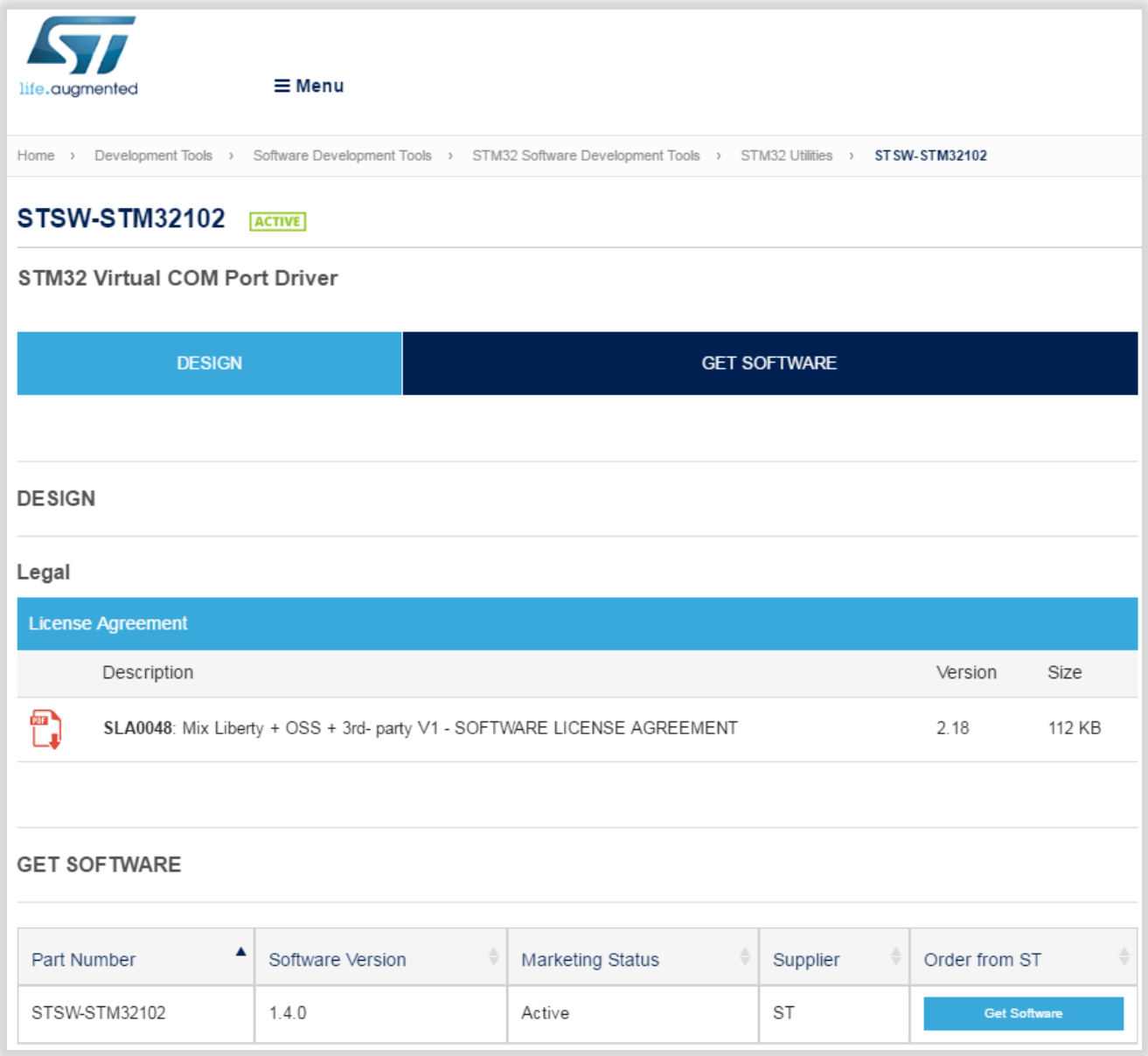
- Now we set the project details for generation
  - Menu > Project > Project Settings
  - Set the project name
  - Project location
  - Type of toolchain
- Now we can Generate Code
  - Menu > Project > Generate Code



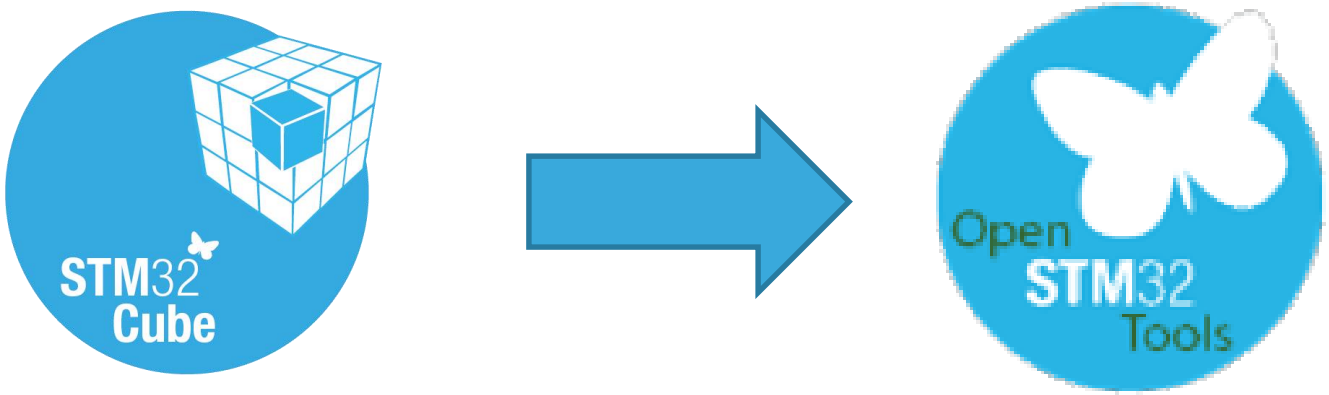
life.augmented

STM32 VCP driver

- In order to communicate between STM32 and PC terminal via VCP install driver
  - In www.st.com find **STSW-STM32102**
  - Click on **Get Software** button
  - Install downloaded driver on PC



life.augmented



- After successful code generation by STM32CubeMX this is the right time to import it into SW4STM32 toolchain for further processing

Modifying the code

data declaration and its sending - main.c file

Tasks:

1. Create the buffers with size of 2048 bytes for received and transmitted data

```
/* USER CODE BEGIN PV */
/* Private variables ---------------------------------------------------------*/
uint8_t UserTxBuffer[2048] = {'S', 'T', 'M', '3', '2', ' '};
uint8_t UserRxBuffer[2048];

/* USER CODE END PV */
```

Creation of two matrixes: first one to send the data and second one to receive the data

Tasks:

1. Call a function to associate previosulty created matrix with transmission buffer
2. Call a function to transmit packet with content from transmission buffer
3. Call a delay function to create a delay between sending of data in the loop

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

  /* USER CODE BEGIN 3 */
  USBD_CDC_SetTxBuffer(&hUsbDeviceFS, UserTxBuffer, 6);
  USBD_CDC_TransmitPacket(&hUsbDeviceFS);
  HAL_Delay(500);

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

Function call to assign matrix to transmission buffer

Function call to send the buffer via USB VCP

Function call to create a delay between two consecutive data sendings

Modifying the code

receiving of VCP data - usbd_cdc_if.c file

Tasks:

1. Refer to the buffer, which was previously created for received data
2. In USB data reception handler call a function to copy received data to received buffer

```
/* USER CODE BEGIN PRIVATE_TYPES */
extern uint8_t UserRxBuffer[];

/* USER CODE END PRIVATE_TYPES */
```

Creation of external matrix (reference to the same one from main.c) for data reception

```
static int8_t CDC_Receive_FS (uint8_t* Buf, uint32_t *Len)
{
  /* USER CODE BEGIN 6 */
  USBD_CDC_SetRxBuffer(&hUsbDeviceFS, &Buf[0]);
  USBD_CDC_ReceivePacket(&hUsbDeviceFS);
  strlcpy(UserRxBuffer, Buf, (*Len) + 1);

  return (USBD_OK);
  /* USER CODE END 6 */
}
```

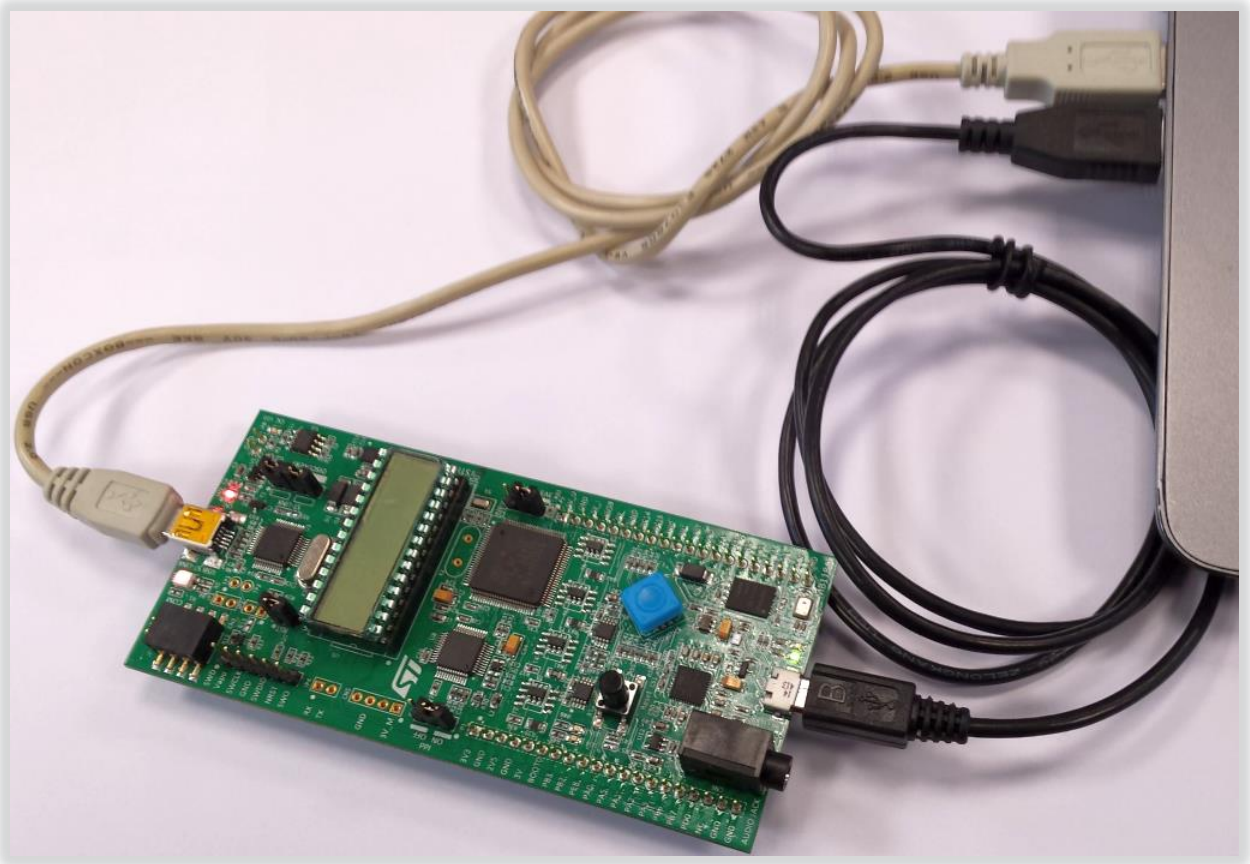
Function call to assign matrix to reception buffer

Function call to receive data via USB VCP

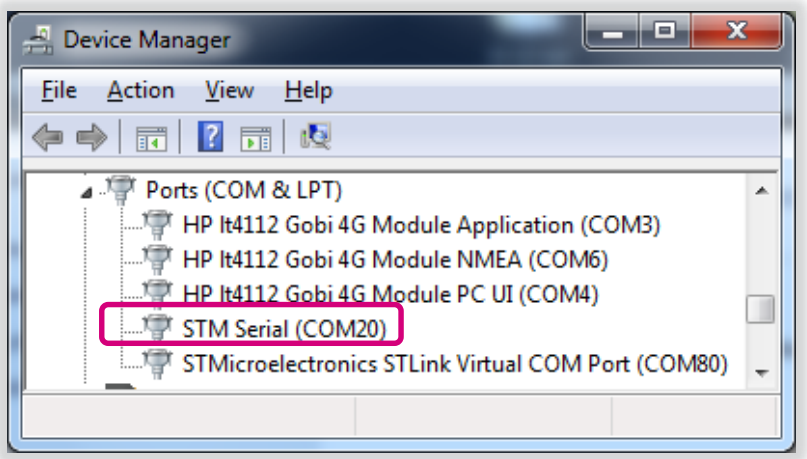
Function call to copy content of local matrix to global matrix

Running the application

- Connect STM32L476RG-Discovery with PC using micro USB cable



- Identify number of COM Port, which was assigned by PC's operating system to STM32L476RG-Discovery

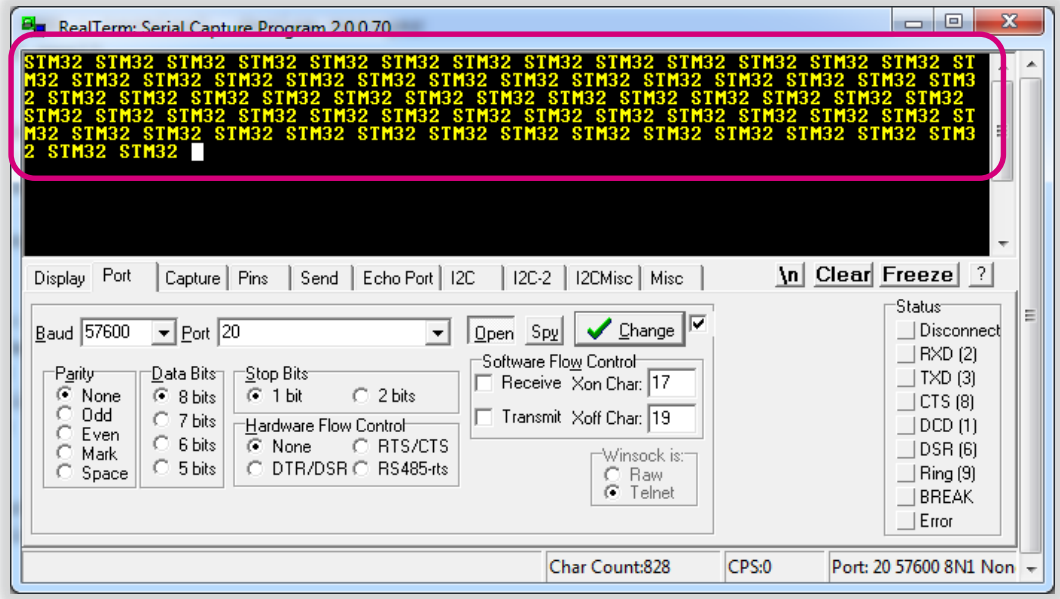


life.augmented

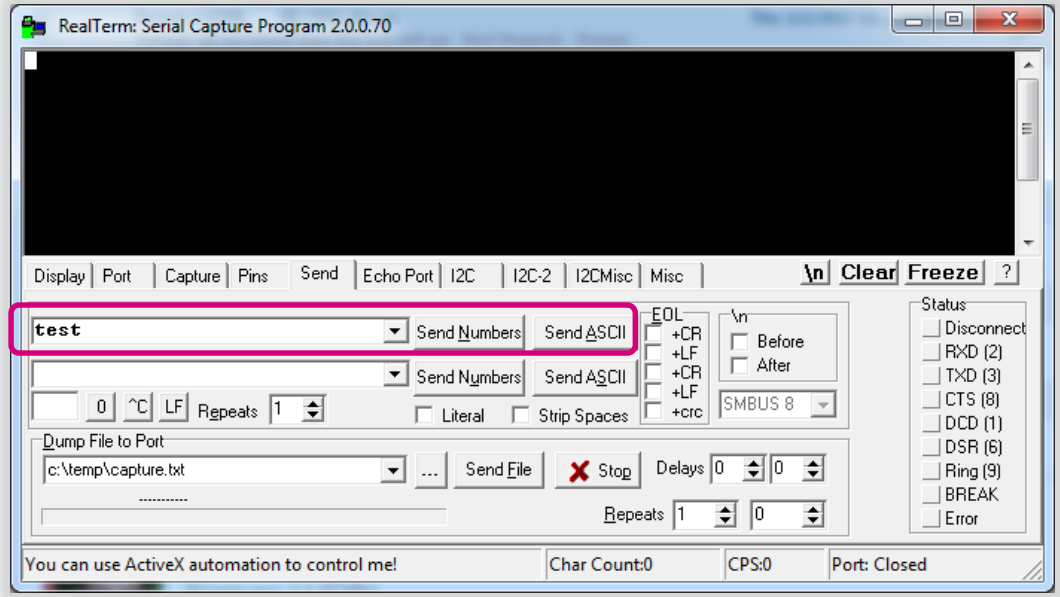
Running the application

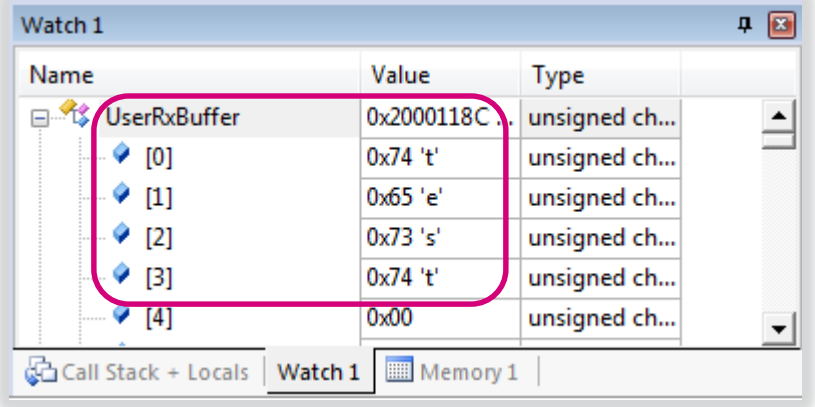
- MCU -> PC communication
  - Open PC terminal (for example RealTerm), connect to identified COM Port and observe the traffic

Any configuration of baudrate, stop/data bits and parity is ok



- PC -> MCU communication
  - Open PC terminal (for example RealTerm), connect to identified COM and send some data
  - In debug session observe content of the reception buffer

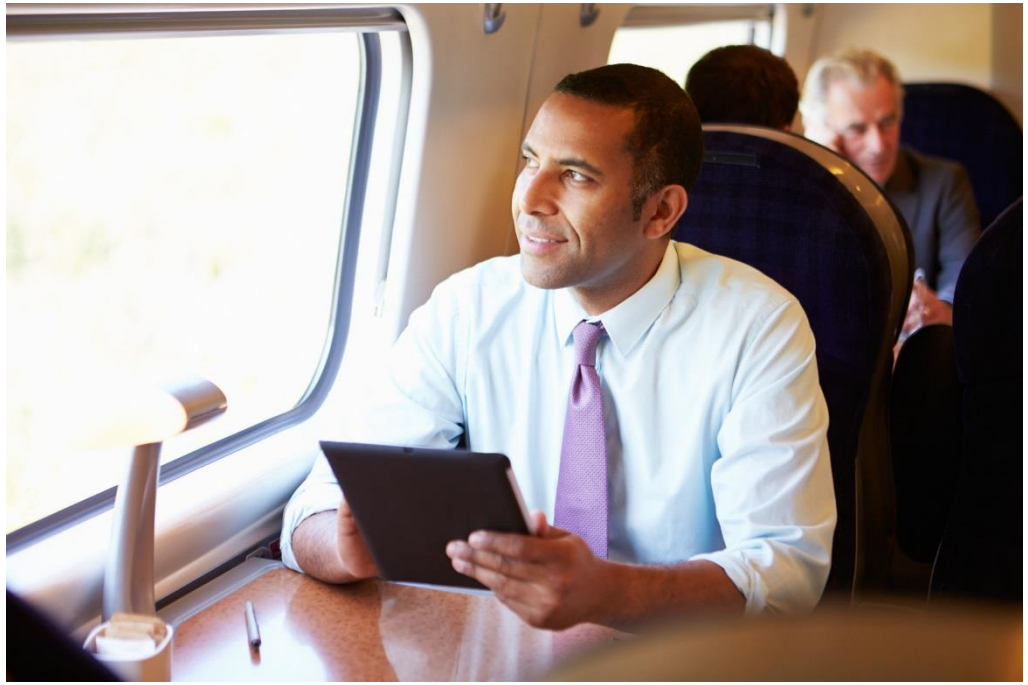




life.augmented

Further reading

- **UM1734** – STM32Cube USB device library user manual
- **STSW-STM32102** – STM32 VCP driver



life.augmented

Enjoy!

www.st.com/mcu

life.augmented