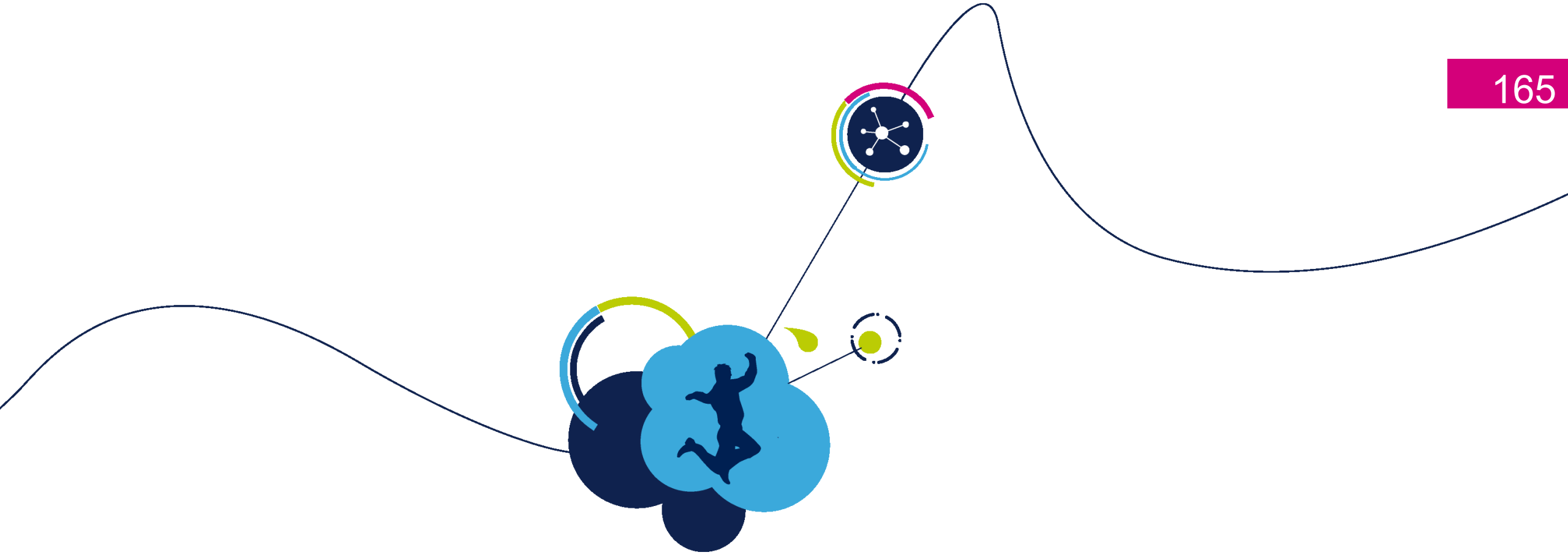


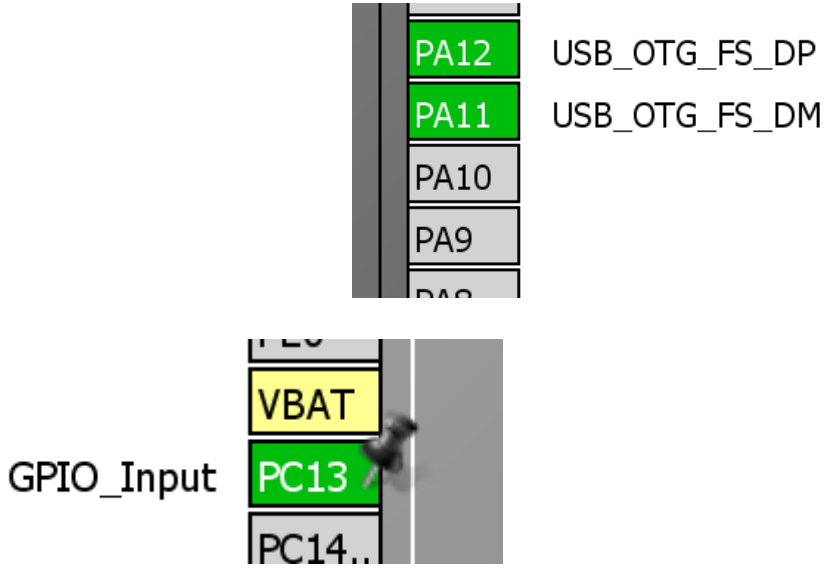
USB MSC Host

USB MSC Host

- MSC host shall handle filesystem of connected device
- SCSI commands usage
- Most USB flash stick are not certified
  - USB host library is not so universal to support devices, which are not following strictly USB specification
  - User may modify USB MSC host library in such cases to enable functionality with specific device, but interface universality may be lost

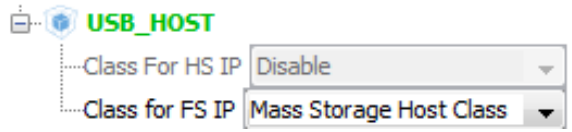
USB MSC Host

- Create project in CubeMX
  - Menu > File > New Project
  - Select STM32F4 > STM32F446 > LQFP144 > STM32F446ZETx
- Select USB FS OTG in host mode
- Select HSE clock
  - (Bypass HSE from STlink)
- Configure PC13 as input – key button
- Configure GPIOs connected to LEDs as GPIO output – PB0, PB7 and PB14

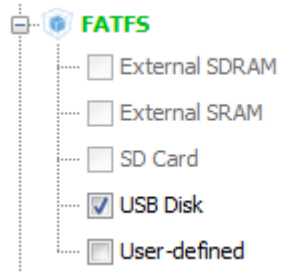


USB MSC Host

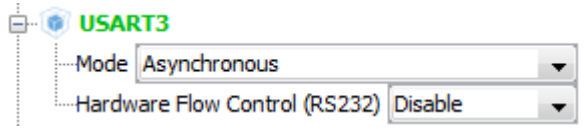
- Select Communication host class in MiddleWares



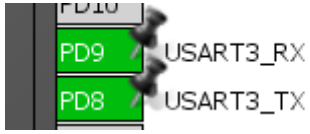
- Configure FAT files system on USB disk



- Configure RCC clocks
  - Set 8 MHz HSE as PLL input and HCLK frequency 168 MHz
- Add USART3 for debug purposes
  - USART3 is connected to STlink virtual COM port functionality



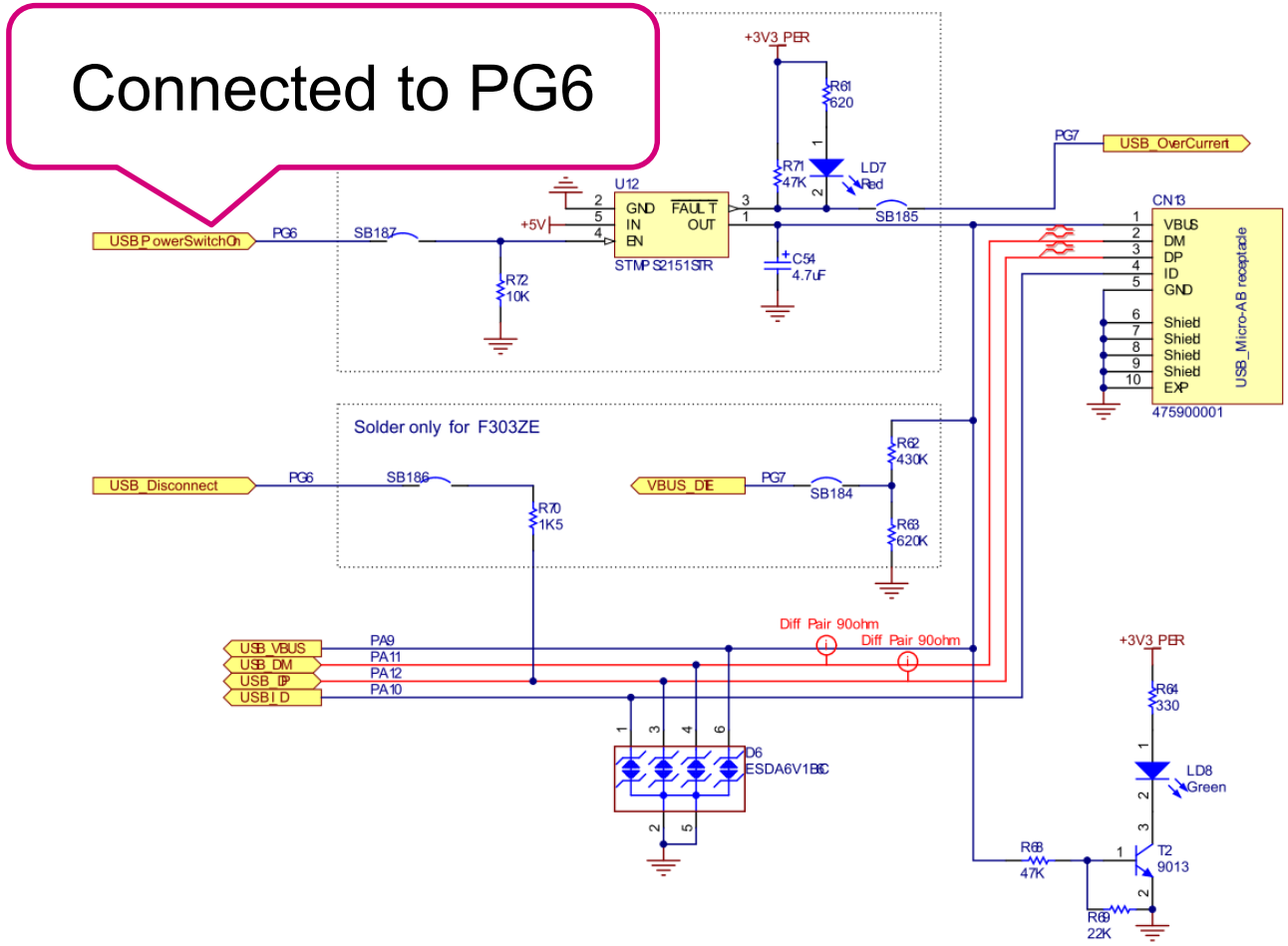
  - PD9 – USART3_RX
  - PD8 – USART3_TX



- For easier handling more convenient DMA implementation is not used

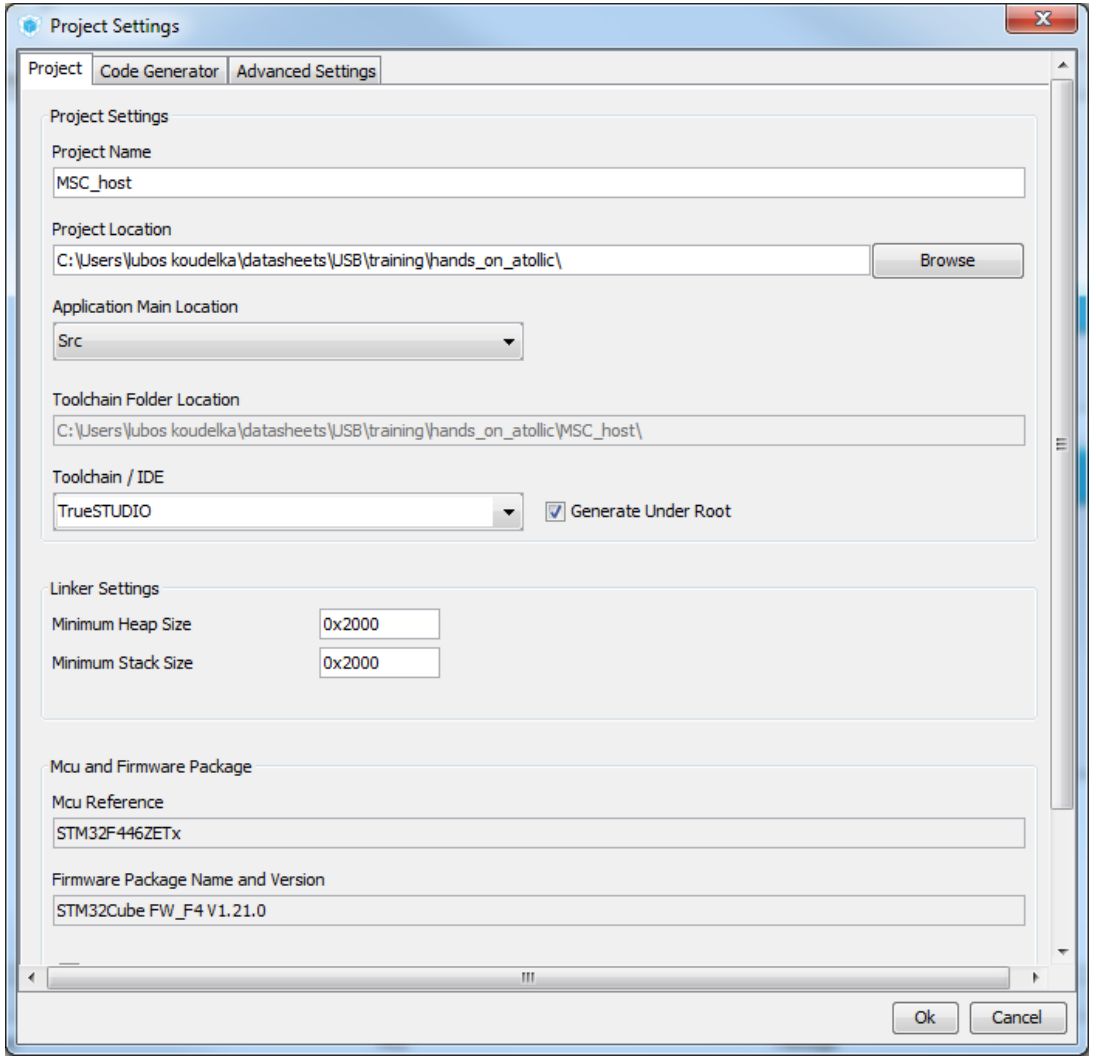
USB MSC Host

- HOST must also power the device -> we need to enable voltage regulator connected to VBUS line
- Set PG6 as GPIO output



USB MSC Host

- Now we set the project details for generation
  - Menu > Project > Project Settings
  - Set the project name
  - Project location
  - Type of toolchain
- Linker Settings
  - Increase Heap size to 0x2000
  - Increase Stack size to 0x2000
- Now we can Generate Code
  - Menu > Project > Generate Code



life.augmented

USB MSC Host

- In usbh_conf.c is function for handling USB VBUS voltage level - USBH_LL_DriverVBUS
- Pin PG6 controls power source for USB VBUS

```
USBH_StatusTypeDef  USBH_LL_DriverVBUS
(USBH_HandleTypeDef *phost, uint8_t state)
{
  /* USER CODE BEGIN 0 */
  /* USER CODE END 0*/
  if (phost->id == HOST_FS)
  {
    if (state == 0)
    {
      /* Deactivate Charge pump */
     HAL_GPIO_WritePin(GPIOG,GPIO_PIN_6,GPIO_PIN_RESET);
      /* USER CODE END DRIVE_HIGH_CHARGE_FOR_FS */
    }
    else
    {
      /* Activate Charge pump */
      HAL_GPIO_WritePin(GPIOG,GPIO_PIN_6,GPIO_PIN_SET);
      /* USER CODE END DRIVE_LOW_CHARGE_FOR_FS */
    }
  }
  HAL_Delay(200);
  return USBH_OK;
}
```

USB MSC Host

- If the Device is connected and enumerated, into Appli_state is stored APPLICATION_READY state and we can commutate with device
  - Storage mount and debug UART message print is added in usb_host.c

```
static void USBH_UserProcess(USBH_HandleTypeDef *phost, uint8_t id) {
/* USER CODE BEGIN 1 */
switch (id) {
case HOST_USER_SELECT_CONFIGURATION:
break;

case HOST_USER_DISCONNECTION:
Appli_state = APPLICATION_DISCONNECT;
break;

case HOST_USER_CLASS_ACTIVE:
Appli_state = APPLICATION_READY;
uart_length=sprintf(uart_tx_buffer, "application ready \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
if(f_mount(&USBH_fatfs, USBHPath, 0) != FR_OK)
    {
  uart_length=sprintf(uart_tx_buffer, "f_mount fail \n");
  HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
    }
break;

case HOST_USER_CONNECTION:
Appli_state = APPLICATION_START;
break;
default:
break;
}
/* USER CODE END 1 */
}
```

Device can communicate

Device not connected

USB MSC Host

- Operation with files on connected MSC device is done inside userFunction, check of Appli_state is inside of the userFunction

- User function declaration need to added into usb_host.h

```
void userFunction(void);
```

- And put userFunction in while loop
  - In MX_USB_HOST_Process is handled device enumeration

```
while (1)
 {
 /* USER CODE END WHILE */
   MX_USB_HOST_Process();
 /* USER CODE BEGIN 3 */
   userFunction();
 }
```

USB MSC Host

- Functionality is added in usb_host.c

```
/* USER CODE BEGIN 0 */
#include "ff.h"
FATFS USBH_fatfs;
FIL MyFile;
FRESULT res;
uint32_t bytesWritten;
uint8_t rtext[200];
uint8_t wtext[] = "USB Host Library : Mass Storage Example";
uint8_t name[10];//name of the file
uint16_t counter=0;
uint32_t i=0;
static int32_t uart_length=0;
extern char USBHPath [];  /* USBH logical drive path */

extern UART_HandleTypeDef huart3;
uint8_t uart_tx_buffer[100];

void userFunction(void) {
uint16_t bytesread;
if (Appli_state == APPLICATION_READY) {
if((HAL_GPIO_ReadPin(GPIOC,GPIO_PIN_13)==GPIO_PIN_SET) && i>0xfffff){
i=0;
```

USB MSC Host

- ...

```
sprintf(name,"%d.txt",counter++);
/*open file*/
if (f_open(&MyFile, name, FA_CREATE_ALWAYS | FA_WRITE) != FR_OK) {
/*file open failed*/
uart_length=sprintf(uart_tx_buffer, "Cannot open %s file \n", name);
HAL_UART_Transmit(&huart3, uart_tx_buffer, (uint16_t)uart_length,1000);

} else {
/*write message to the file*/
uart_length=sprintf(uart_tx_buffer, "file %s created \n", name);
HAL_UART_Transmit(&huart3, uart_tx_buffer, (uint16_t)uart_length,1000);
res = f_write(&MyFile, wtext, sizeof(wtext),
(void *) &bytesWritten);

if (f_close(&MyFile) != FR_OK) {
/*file closing failure*/
uart_length=sprintf(uart_tx_buffer, "fclose fail \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
while (1){}
}
/*check number of written bytes*/
if ((bytesWritten == 0) || (res != FR_OK)) {
/*error during writing*/
uart_length=sprintf(uart_tx_buffer, "write error \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
}
```

USB MSC Host

- ...

```
else {
/*open file to verification*/
if (f_open(&MyFile, name, FA_READ) != FR_OK) {
/*file open failure*/
uart_length=sprintf(uart_tx_buffer,"Cannot open %s file for verify \n",
name);
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
} else {
/*read file to verification*/
res = f_read(&MyFile, rtext, sizeof(rtext),(void *) &bytesread);

if ((bytesread == 0) || (res != FR_OK)) {
/*read fail*/
uart_length=sprintf(uart_tx_buffer,"Cannot read file for verification
\n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
} else {
/*read success*/
}
if (f_close(&MyFile) != FR_OK) {
/*check number of written bytes*/
uart_length=sprintf(uart_tx_buffer, "fclose fail \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
while (1){}
}
}
```

USB MSC Host

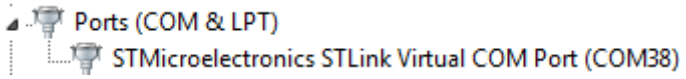
- Last changes before example is finished

```
/* Compare read data with the expected data */
if ((bytesread == bytesWritten)) {
/*verification success full - number of written bytes is equal to number
of read bytes*/
uart_length=sprintf(uart_tx_buffer,"verification OK - read number of bytes
is equal to written number of bytes \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,((uint16_t)uart_length), 5000);

} else {
/*verification failed - number of written bytes is not equal to number of
read bytes*/
uart_length=sprintf(uart_tx_buffer, "verify fail \n");
HAL_UART_Transmit(&huart3, uart_tx_buffer,(uint16_t)uart_length, 1000);
}
/*end program execution after verification*/
}
}
}
i++;
}
}
/* USER CODE END 0 */
```

USB MSC Host

- File is written on each button press when USB MSC device connected
- Then in device manager find COM port number of connected host board STlink



- Debug output can be view in any COM port terminal application