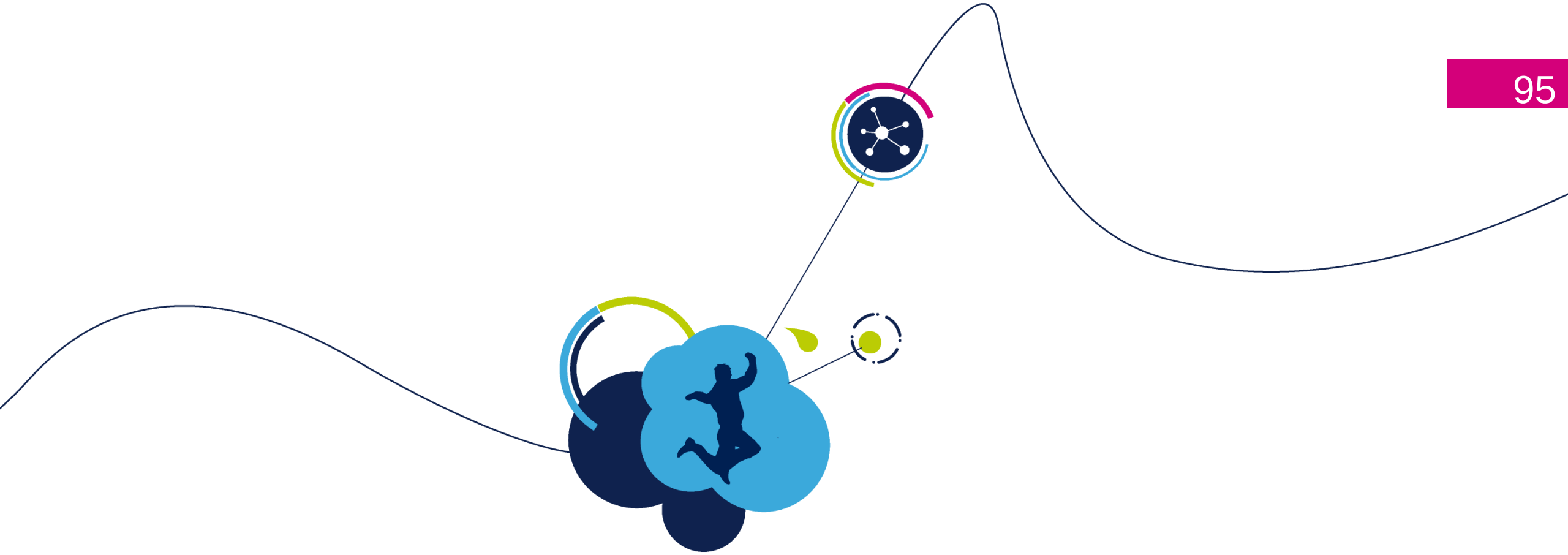
USB Custom HID Device lab

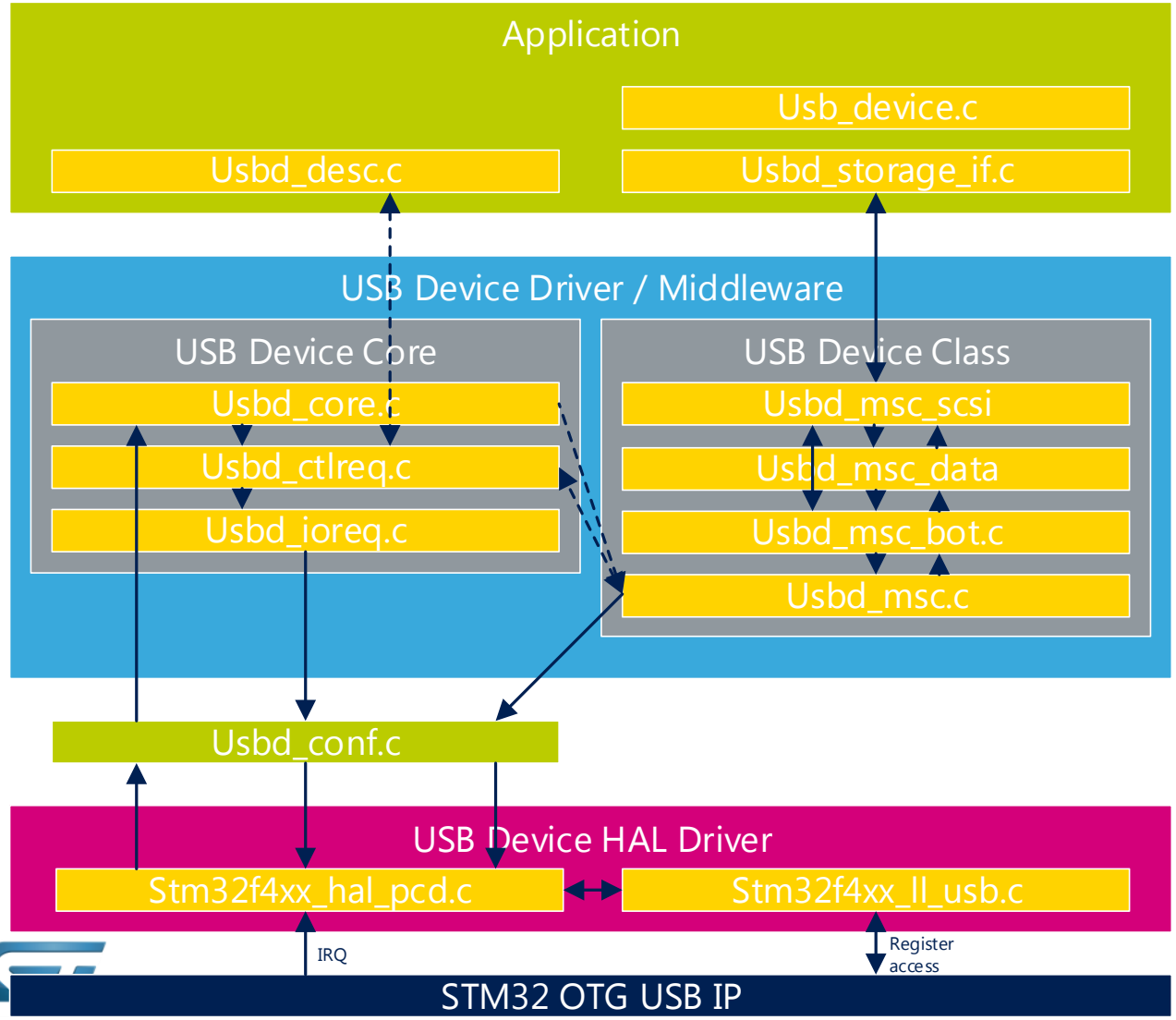
- Now is the device ready for test
- For communication on host side you can use attached HID terminal (C#)

HID_terminal.zip



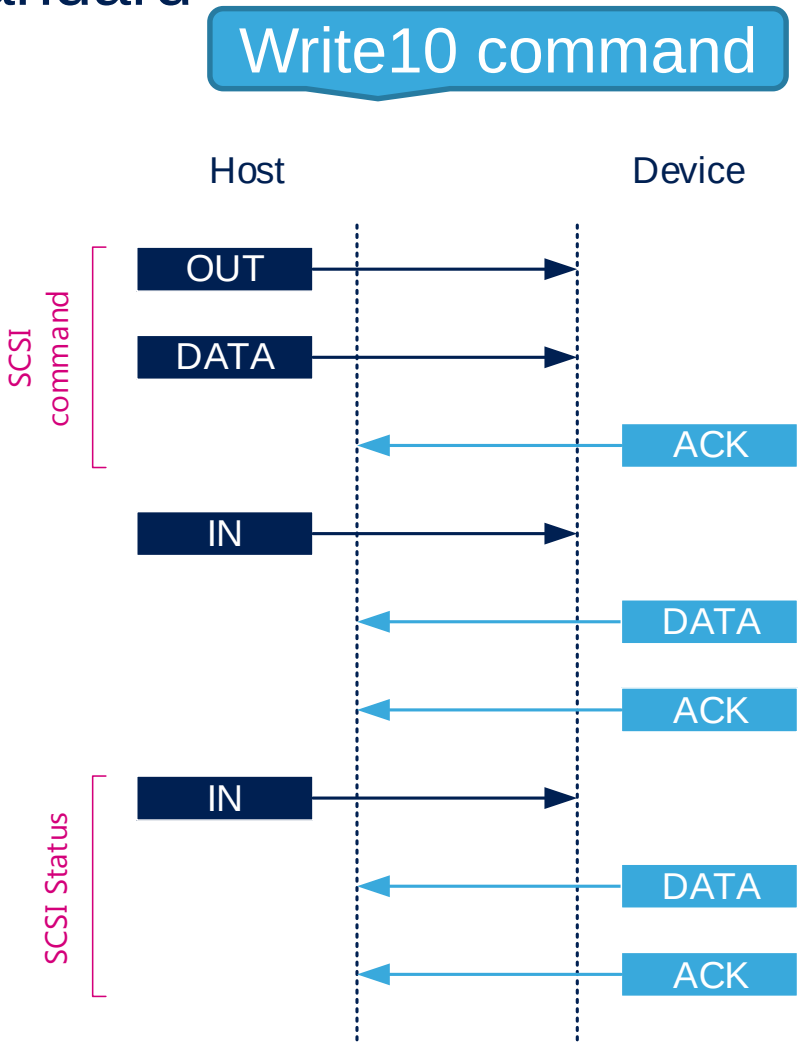
USB Mass storage Device lab

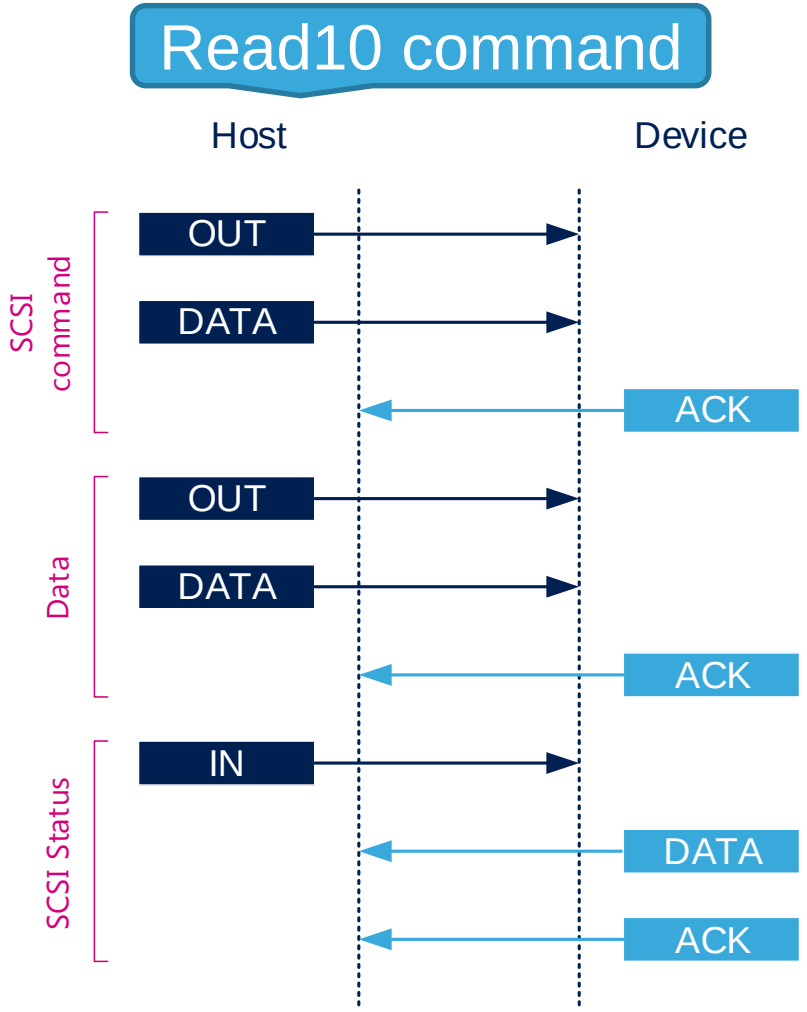
Device MSC structure



SCSI commands

- Defined by SCSI standard
- Structure
  - Command
  - Data
  - Status





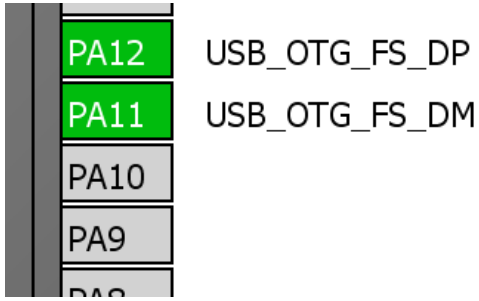
life.augmented

USB Mass storage Device lab

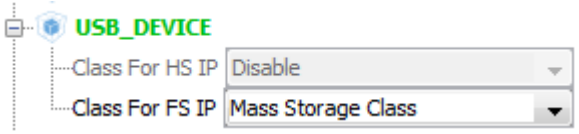
- Used for USB flash keys
- No need for driver on Microsoft Windows
- In the example internal SRAM is used for as storage for USB mass storage

USB Mass storage Device lab

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



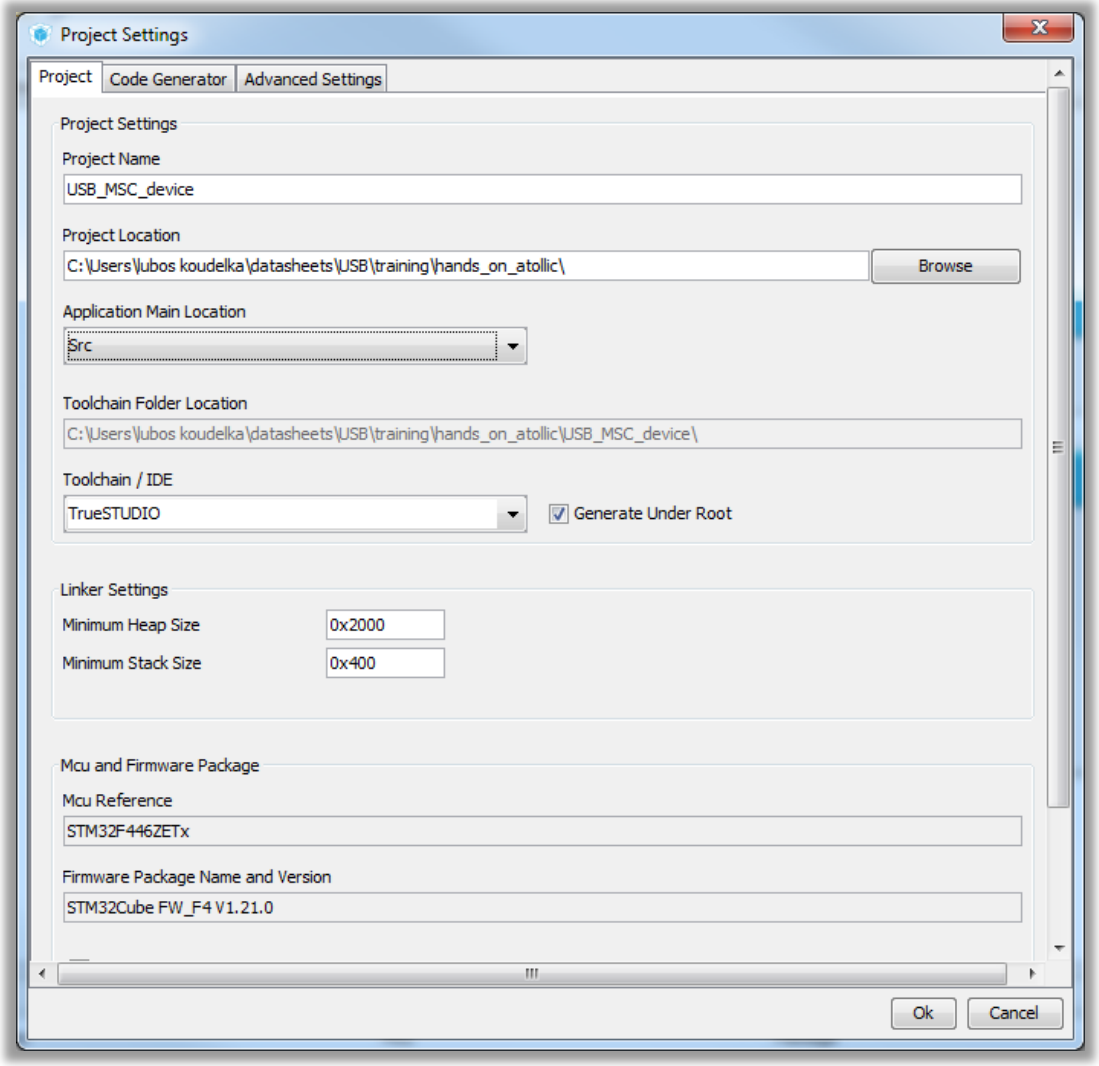
- Select HSE clock
  - (Bypass HSE from STlink)
- Select MSC class in MiddleWares



- Configure RCC clocks
  - Set 8 MHz HSE as PLL input and HCLK frequency 168 MHz

USB Mass storage 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
  - Increase Heap size to 0x2000
- Now we can Generate Code
  - Menu > Project > Generate Code



life.augmented

USB Mass storage Device lab

- All changes in this example will be done in usbd_storage_if.c
- First, storage size defines need to be decreased to fit into SRAM memory

```
#define STORAGE_LUN_NBR                  1
#define STORAGE_BLK_NBR                  0x80
#define STORAGE_BLK_SIZ                  0x200
```

- And buffer for user data is created

```
/* USER CODE BEGIN PRIVATE_VARIABLES */
uint8_t buffer[STORAGE_BLK_NBR*STORAGE_BLK_SIZ];
/* USER CODE END PRIVATE_VARIABLES */
```

USB Mass storage Device lab

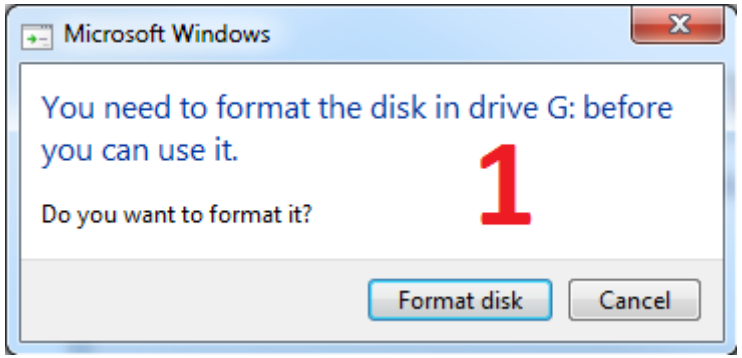
- Functions for Storage read and write are modified to use internal SRAM buffer

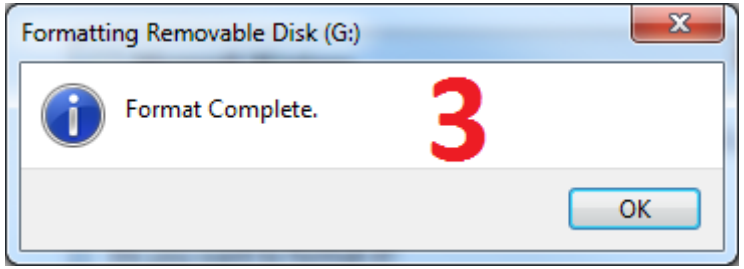
```
int8_t STORAGE_Read_FS (uint8_t lun,
                        uint8_t *buf,
                        uint32_t blk_addr,
                        uint16_t blk_len)
{
  /* USER CODE BEGIN 6 */
  memcpy(buf, &buffer[blk_addr*STORAGE_BLK_SIZ],blk_len*STORAGE_BLK_SIZ);
  return (USBD_OK);
  /* USER CODE END 6 */
}
```

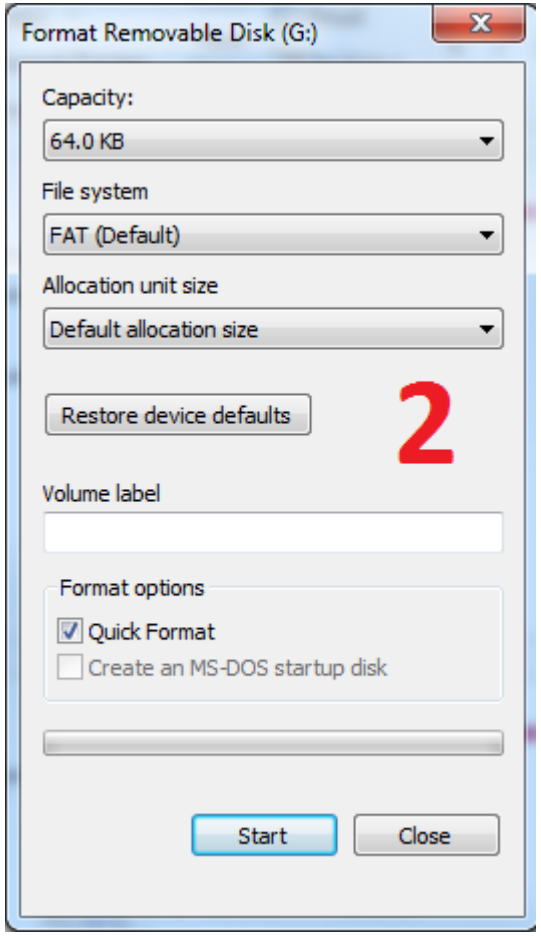
```
int8_t STORAGE_Write_FS (uint8_t lun,
                         uint8_t *buf,
                         uint32_t blk_addr,
                         uint16_t blk_len)
{
  /* USER CODE BEGIN 7 */
  memcpy(&buffer[blk_addr*STORAGE_BLK_SIZ], buf, blk_len*STORAGE_BLK_SIZ);
  return (USBD_OK);
  /* USER CODE END 7 */
}
```

USB Mass storage Device lab

- Firmware for USB MSC device is ready for test
- Connect device to PC and format memory

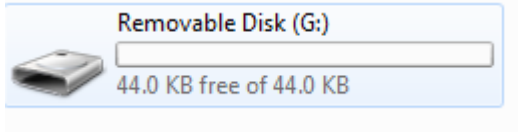






USB Mass storage Device lab

- Now is the device visible in file explorer and can be used as regular FAT medium



- The content on the Removable disk is preserved until reset or MCU power disconnection, then is necessary to format device again – problem of SRMA usage
- With nonvolatile (flash for example) memory would be device behavior as standard
- If user USB user is disconnected and connected with this example, data on Removable Disk are preserved