

# STM32L4 – Putting All Together

STM32L4 Workshop





# Goal of this part

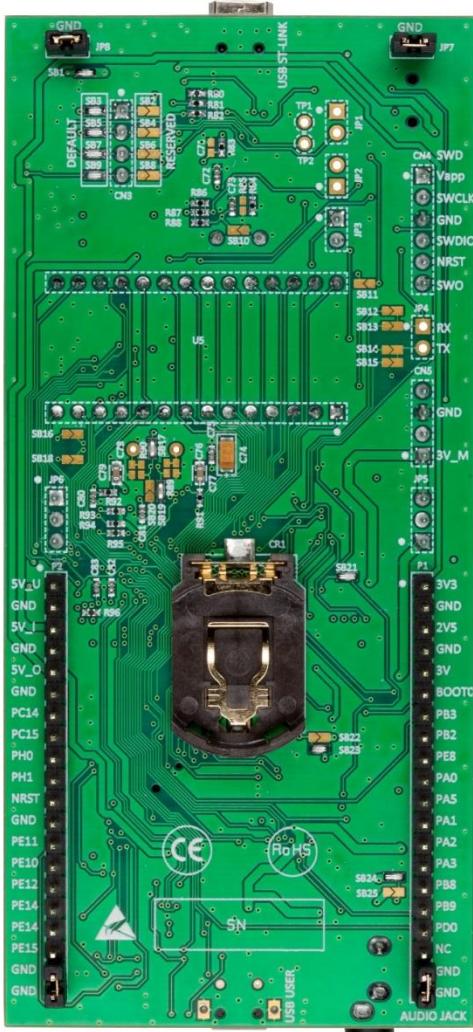
2

- Get some basic knowledge about advanced topics
- Practice a little bit with STM32CubeMX
- Understand the structure of prepared demo application
- Have some fun!



# STM32L476 Discovery

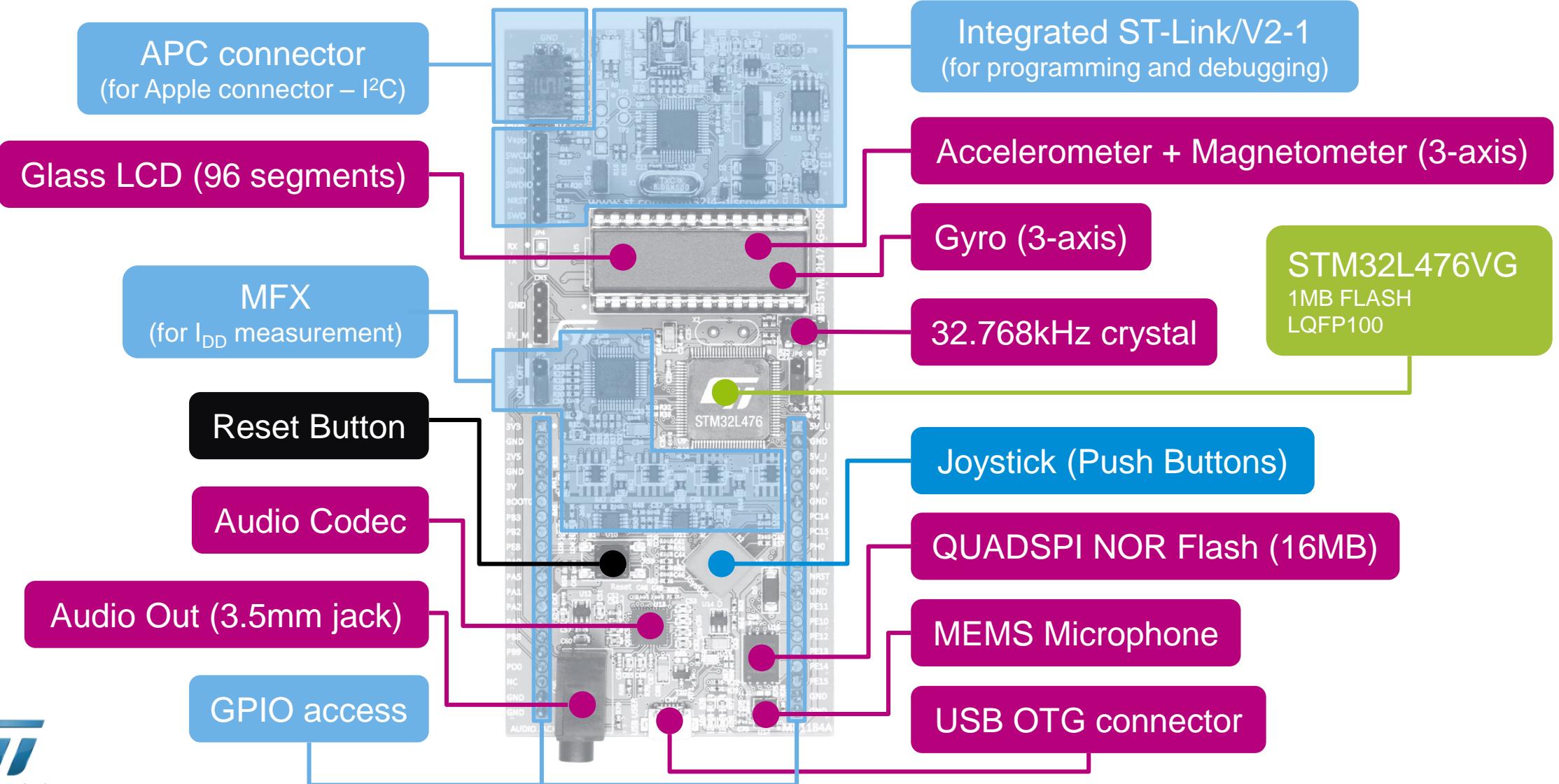
4





# STM32L476 Discovery

5

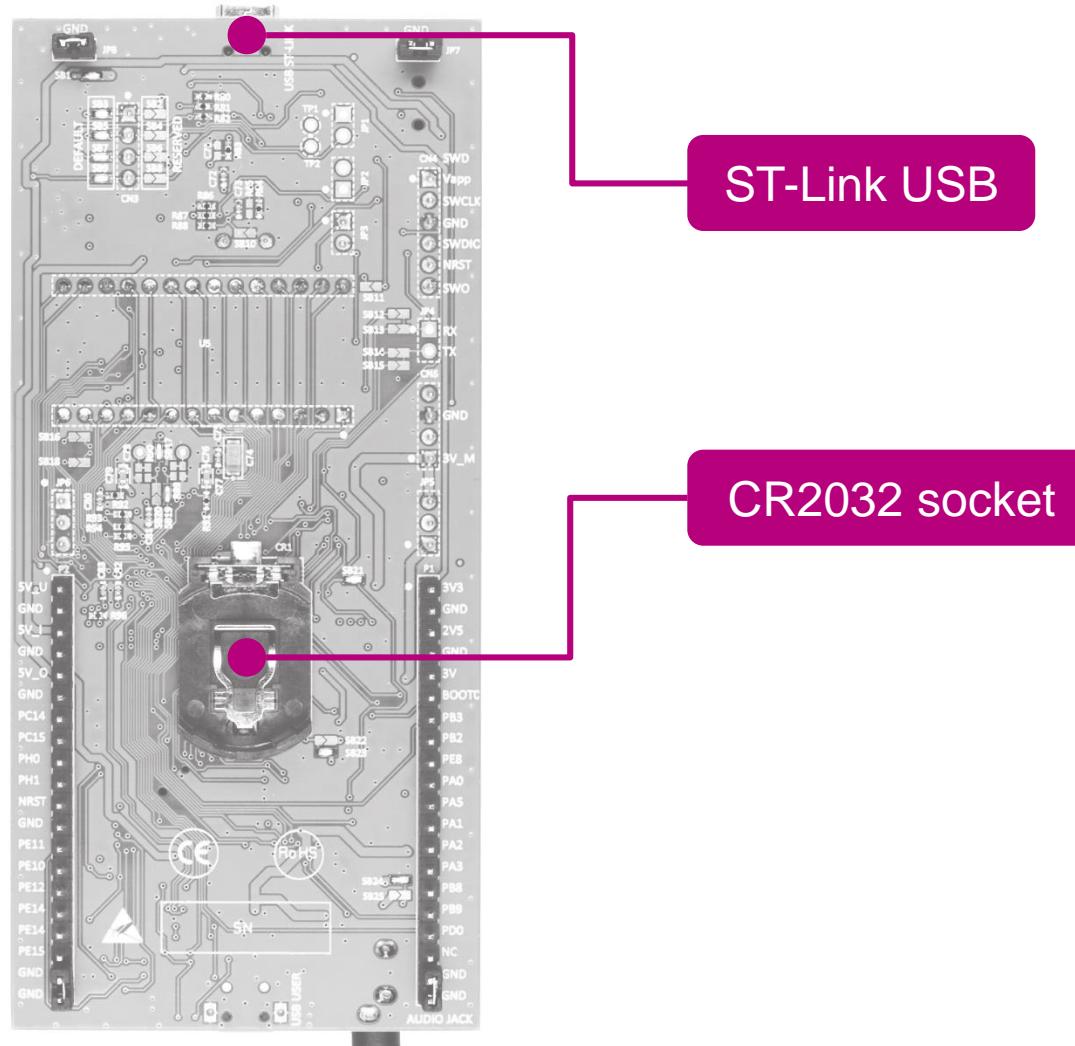




# STM32L476 Discovery

6

Flexible board  
power supply



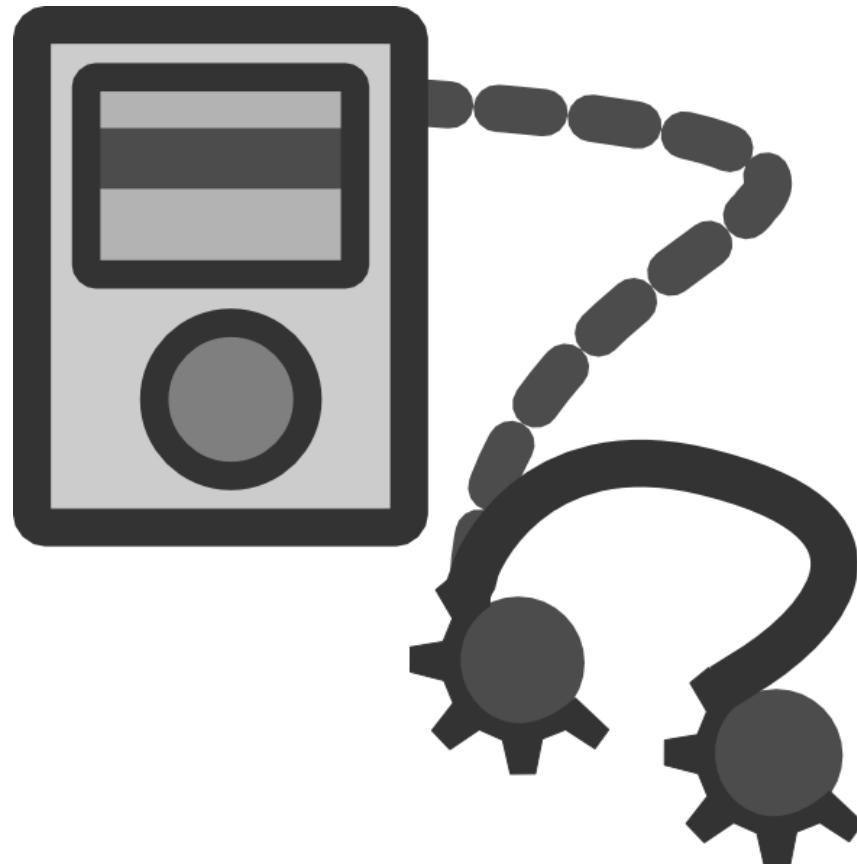
ST-Link USB

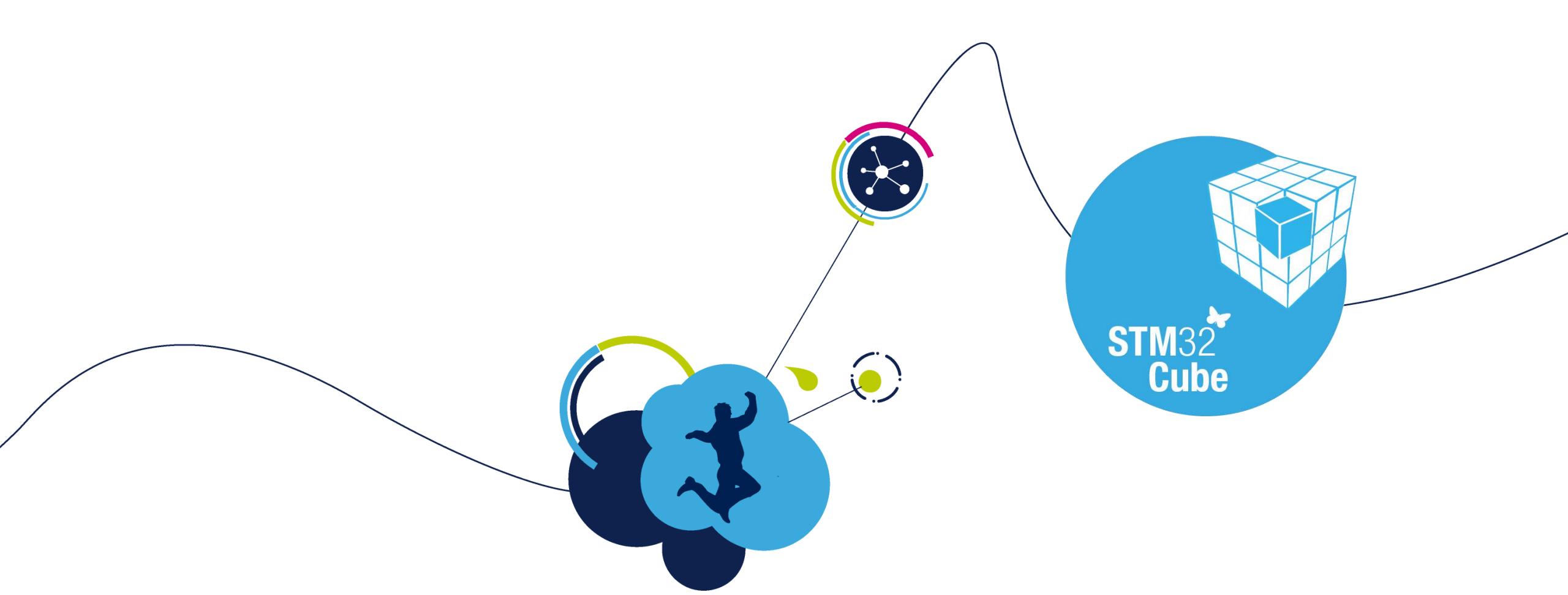
CR2032 socket

# Application idea

7

- Create simple Audio Player





Let's start to work!

# Application idea

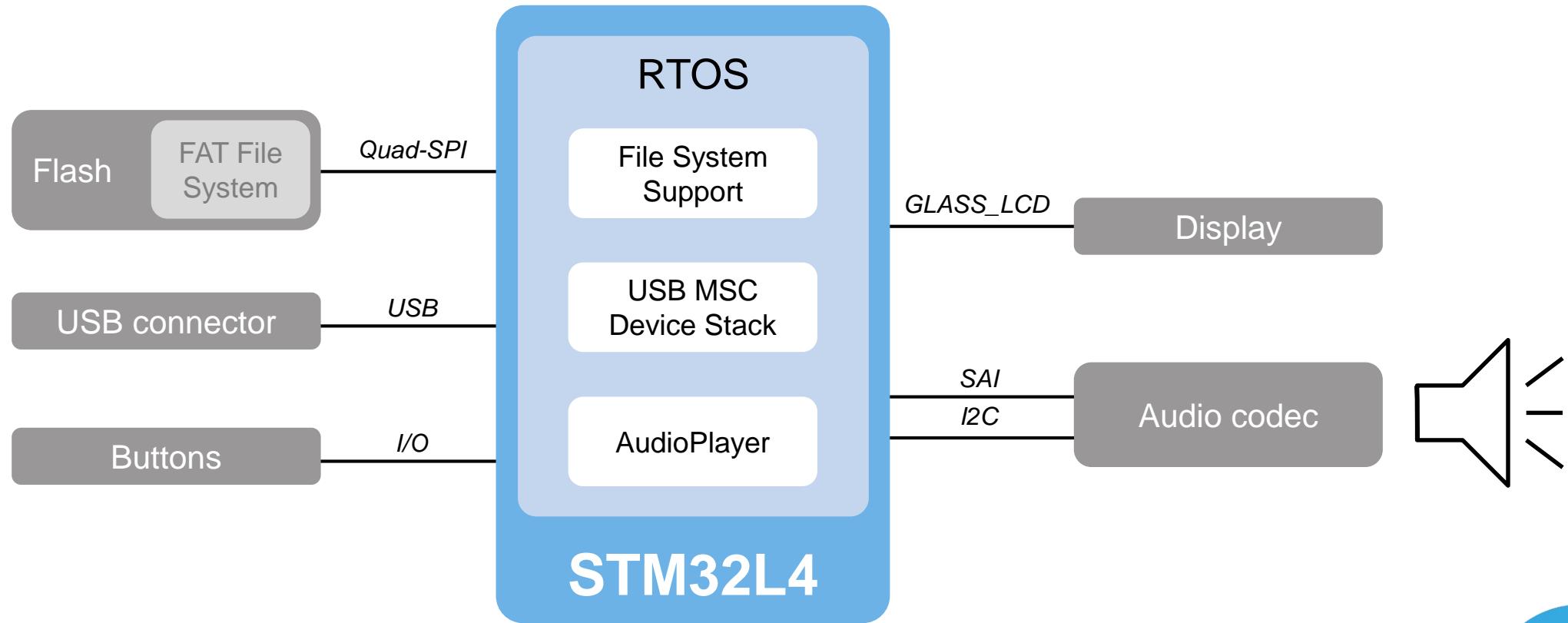
9

- Create simple Audio Player



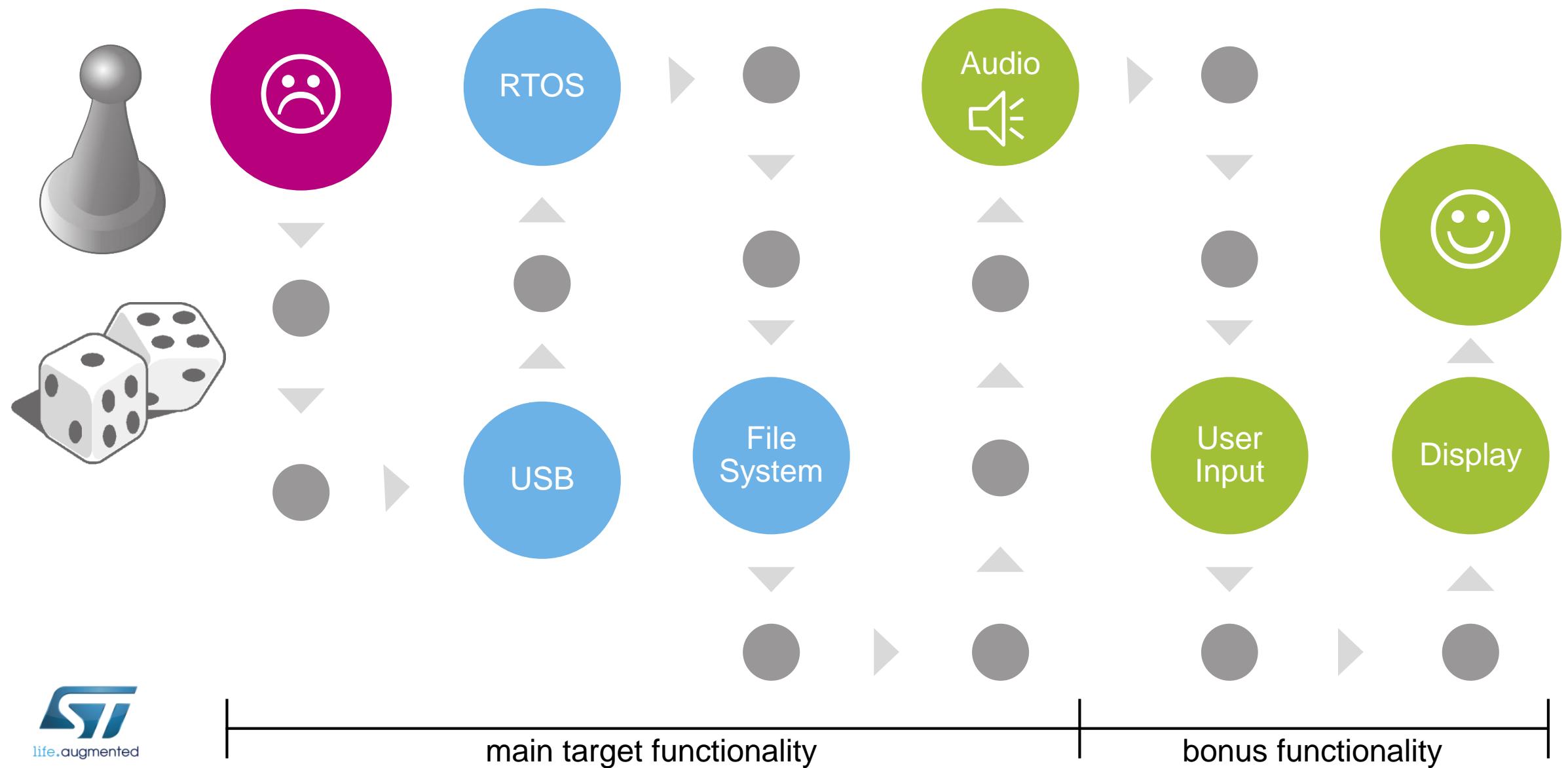
# Simplified target block diagram

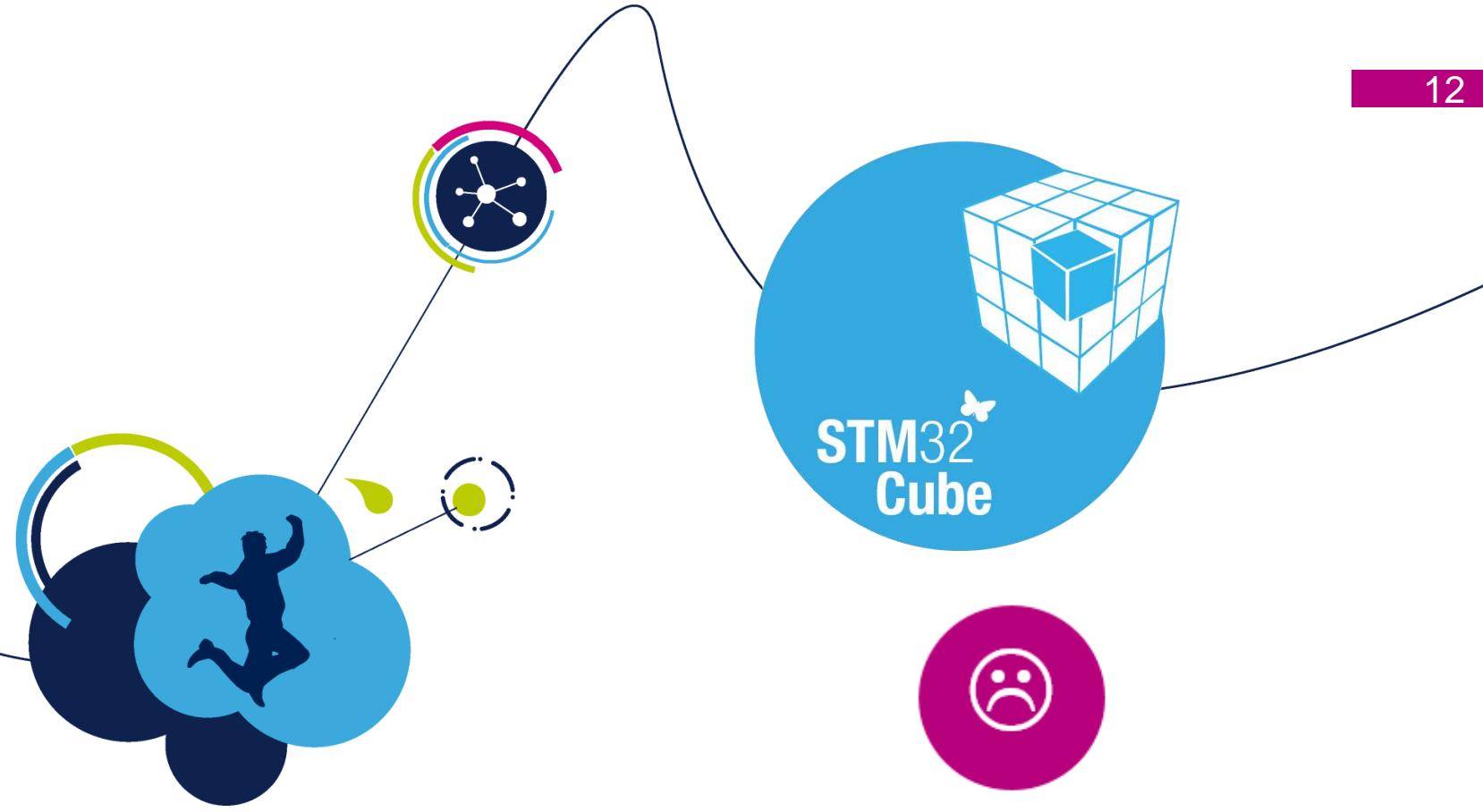
10



# „Game“ flow

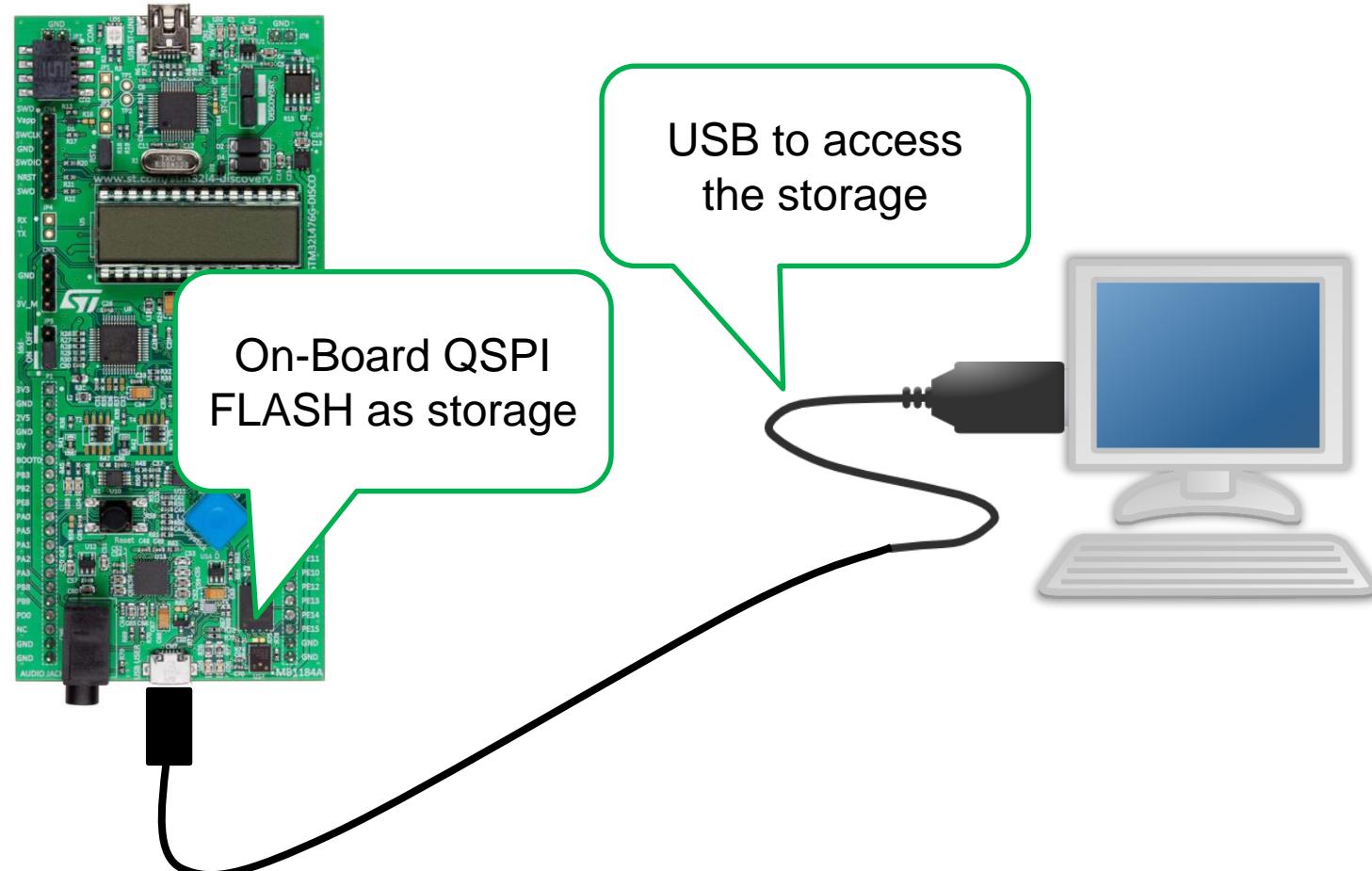
11





## STEP1 – USB MSC Device

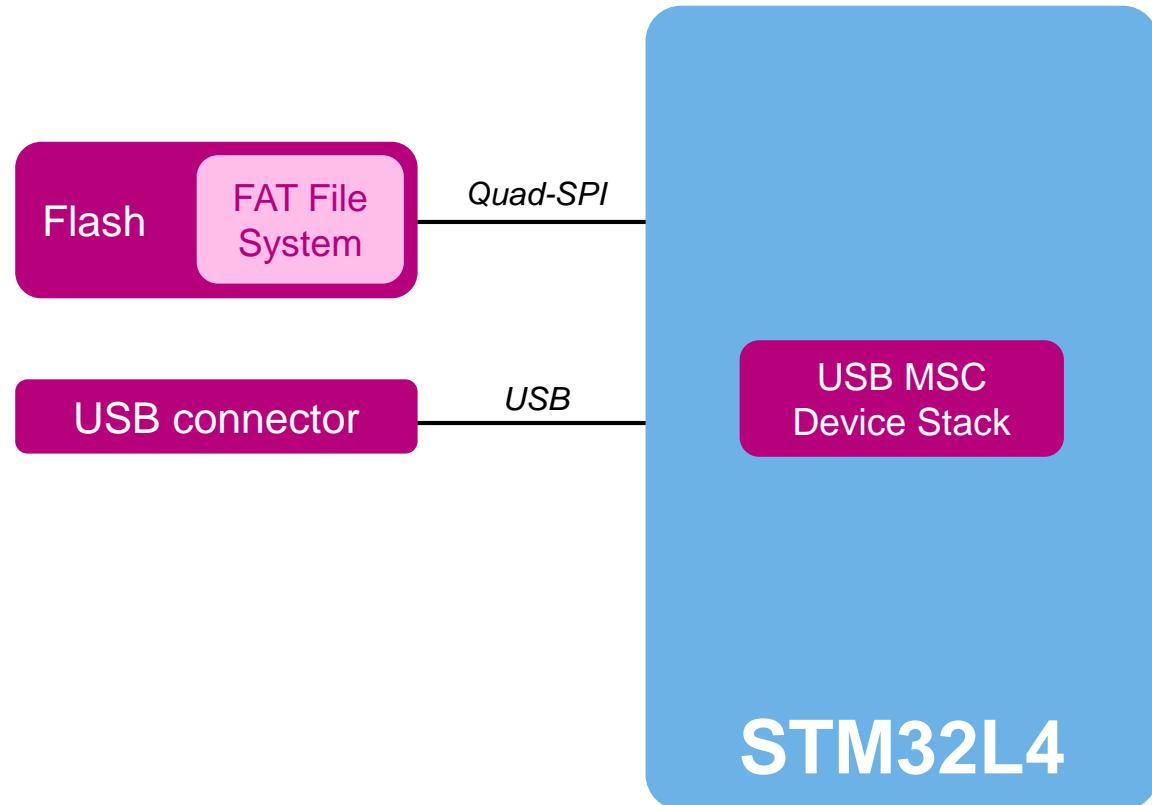
- We need storage for our audio files!



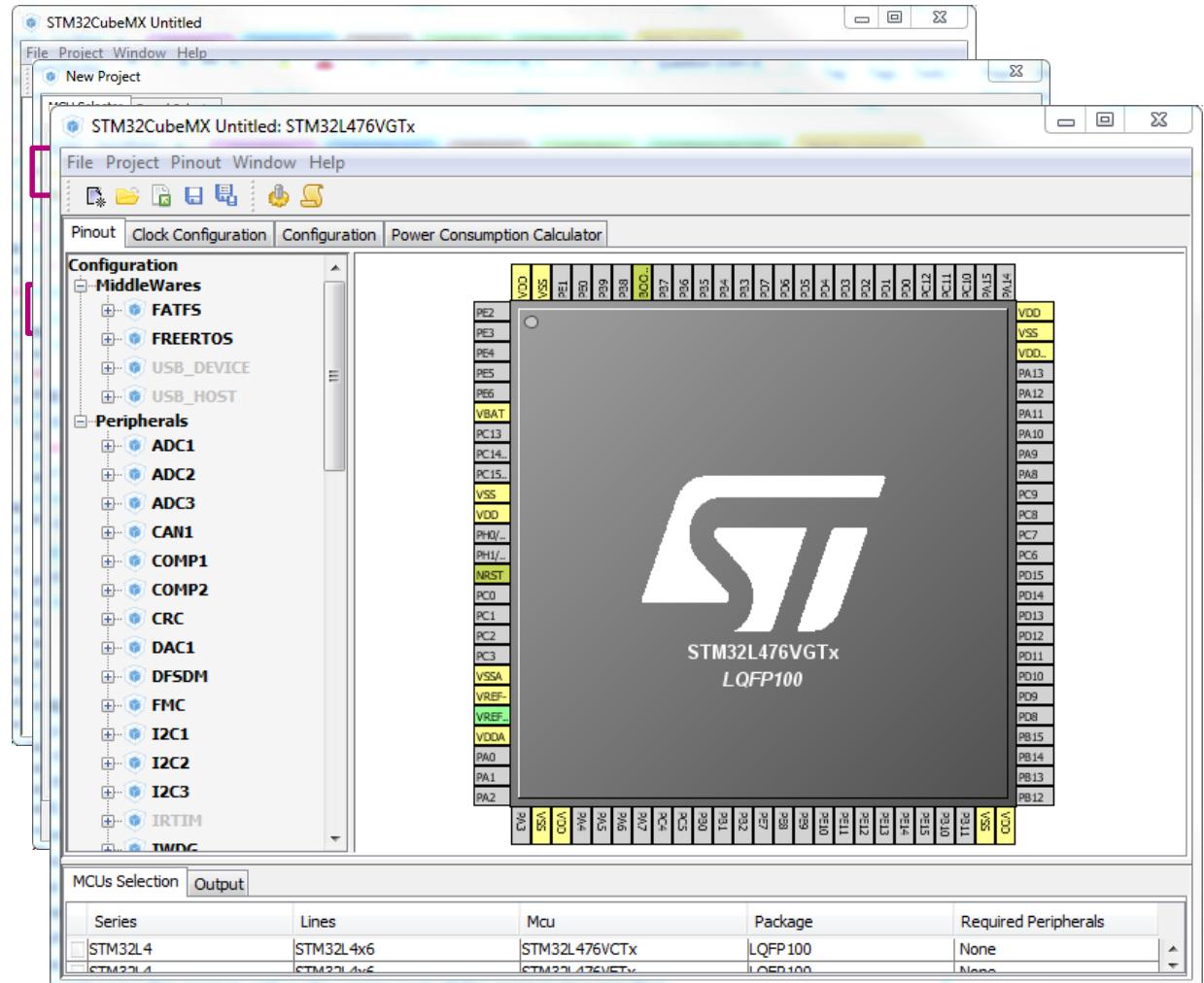
# 1 Adding USB MSC Device functionality

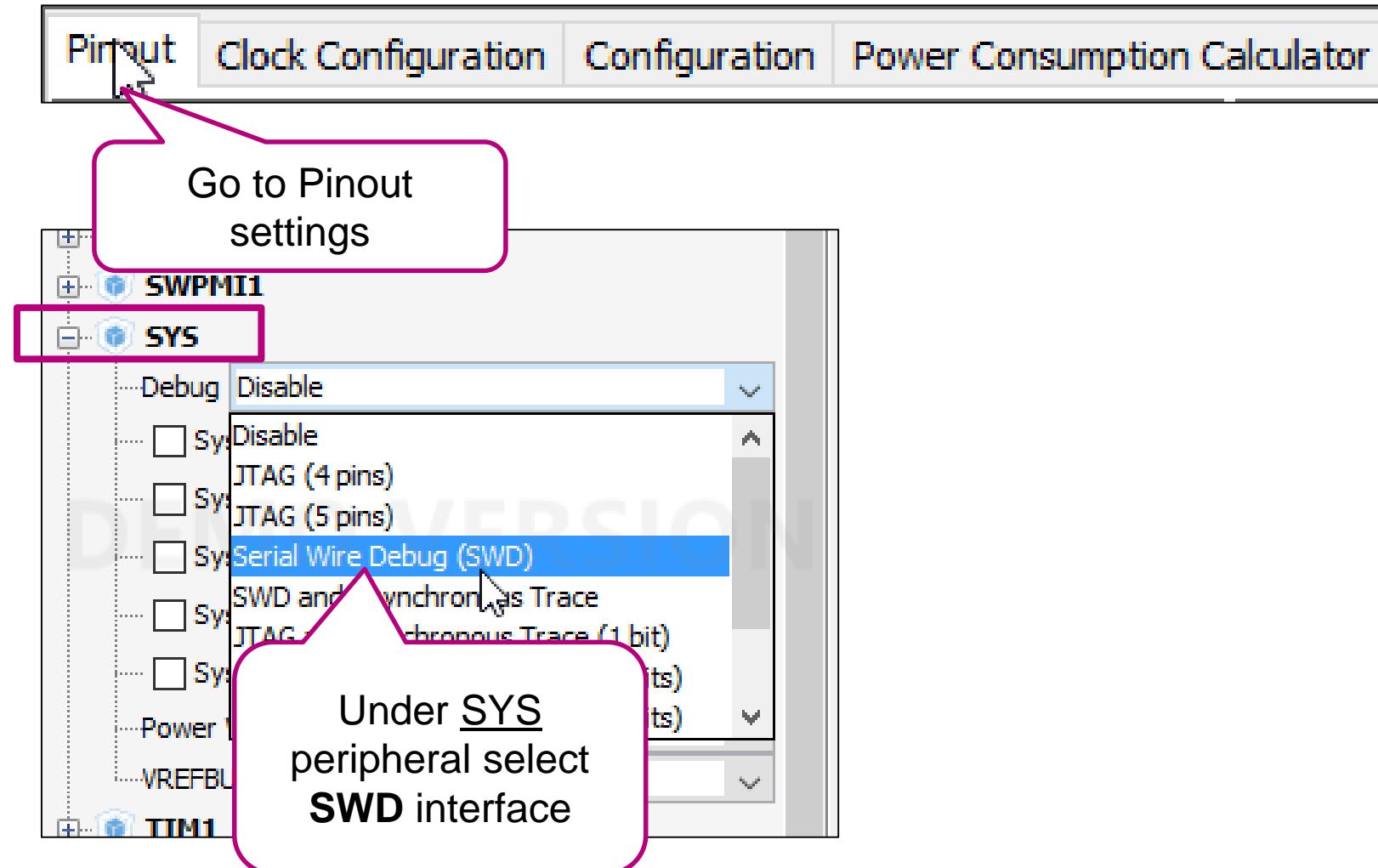
14

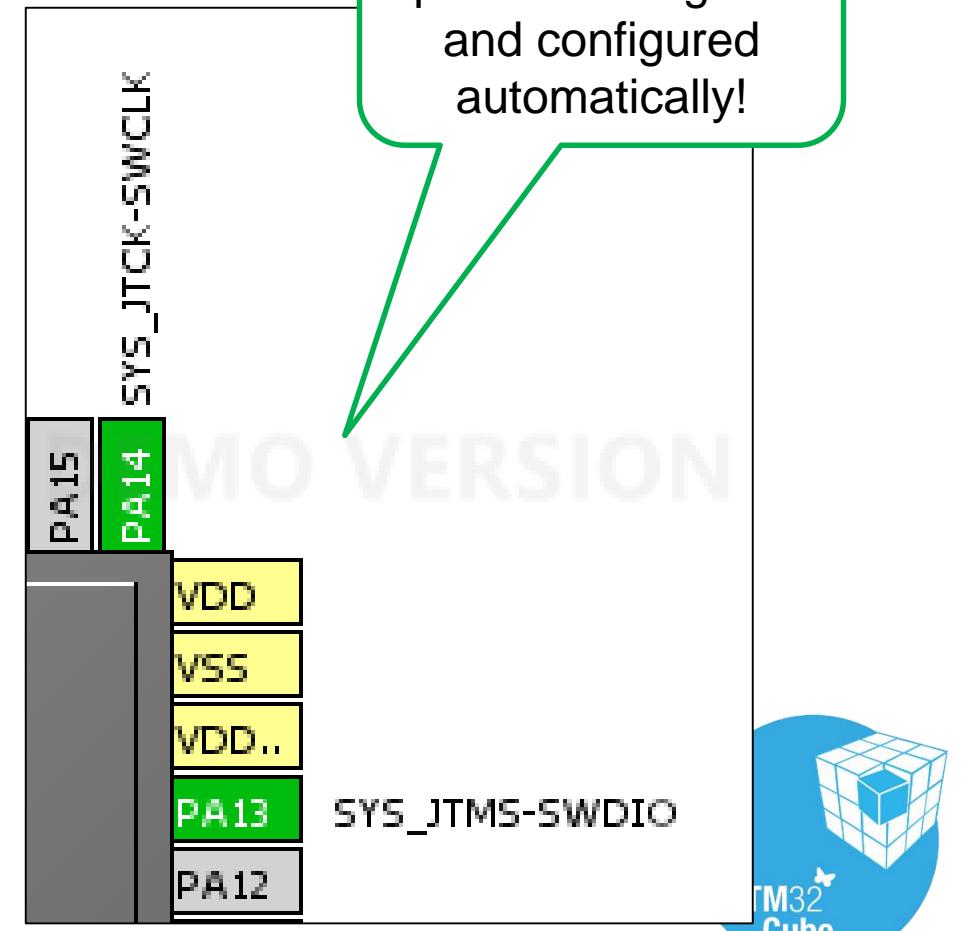
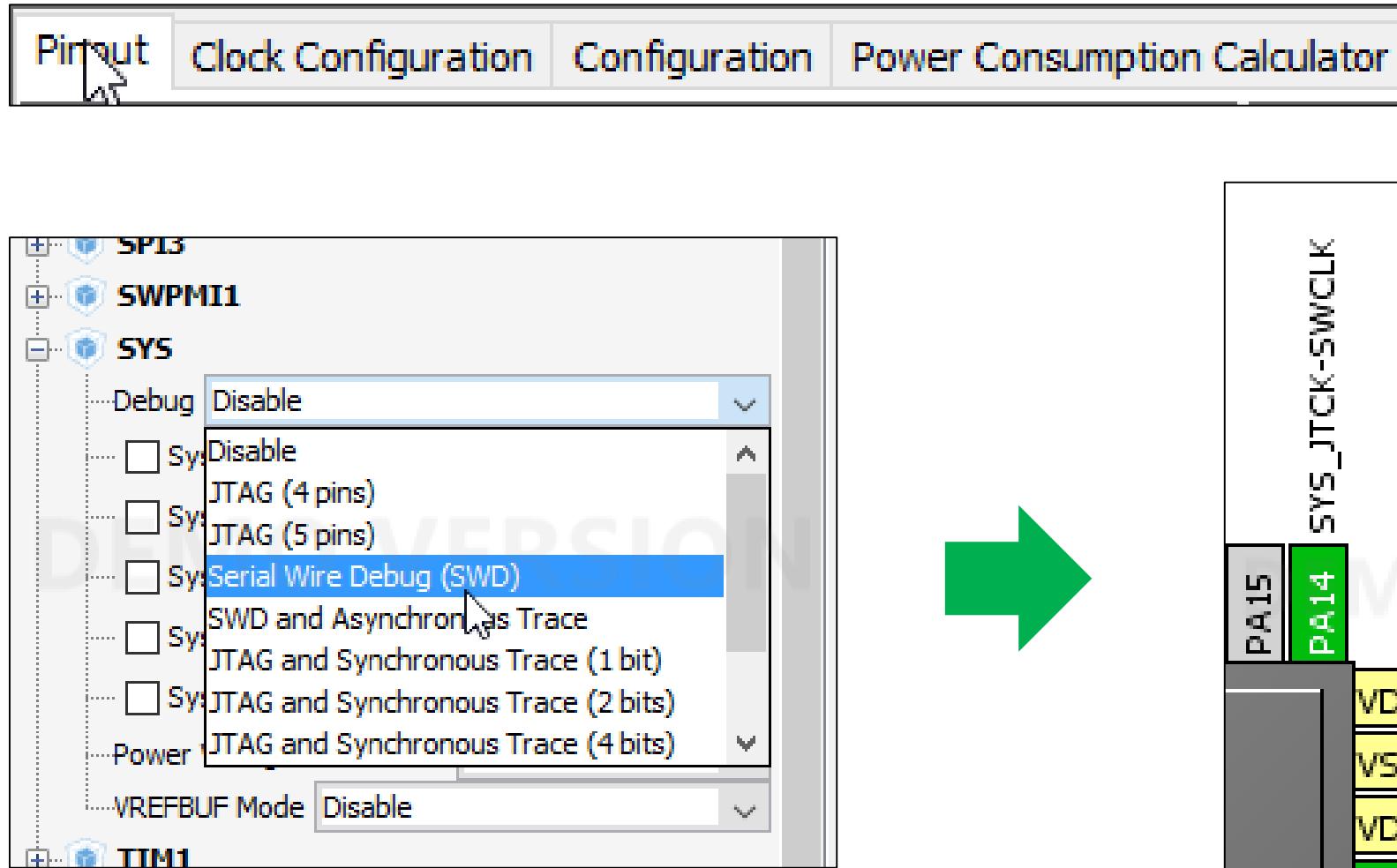
- We need storage for our audio files!



- Run CubeMX tool
- Menu->File->New Project
- Filter:
  - Series STM32L4
  - Line STM32L4x6
  - Package LQFP100
- Select STM32L476VGTx

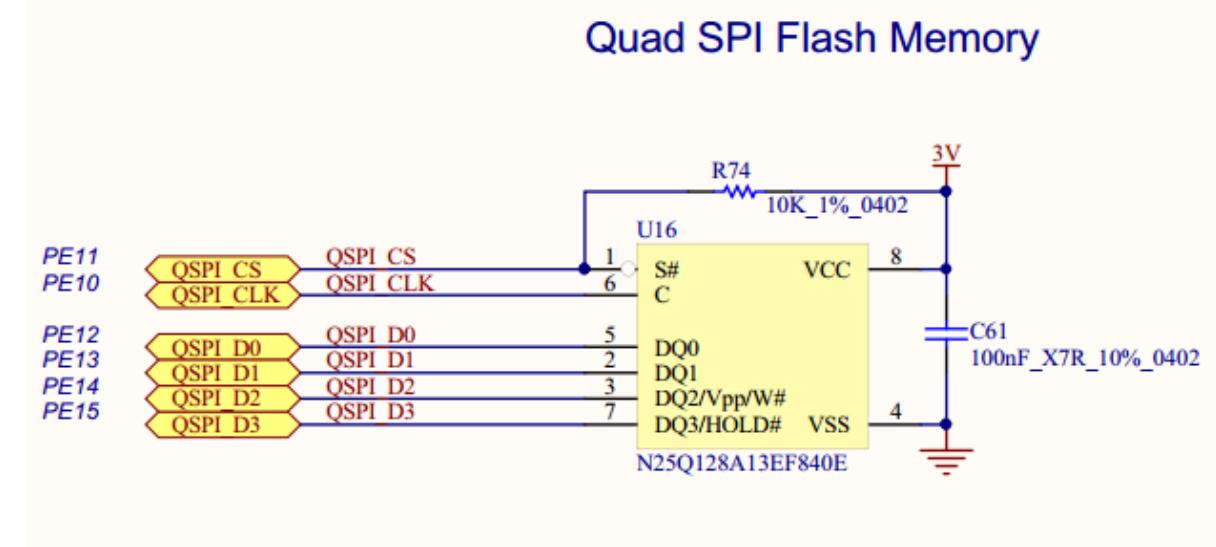
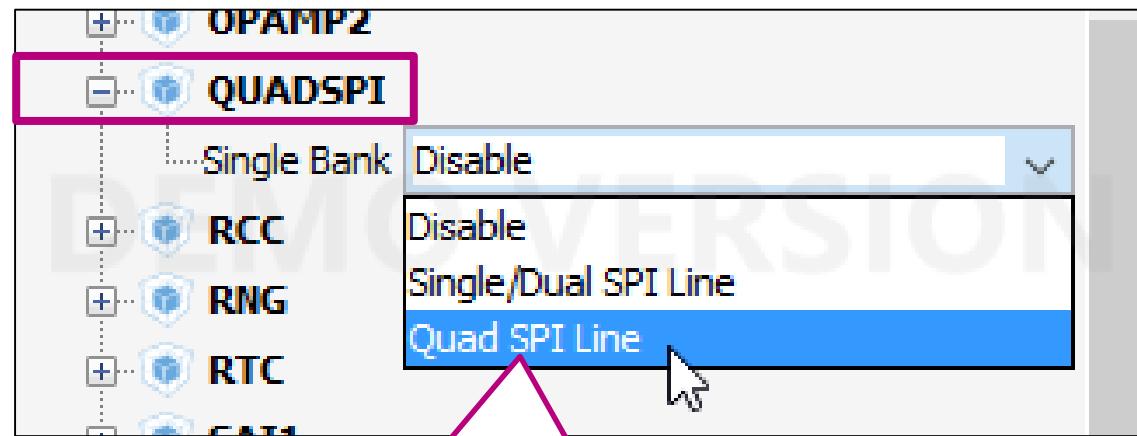
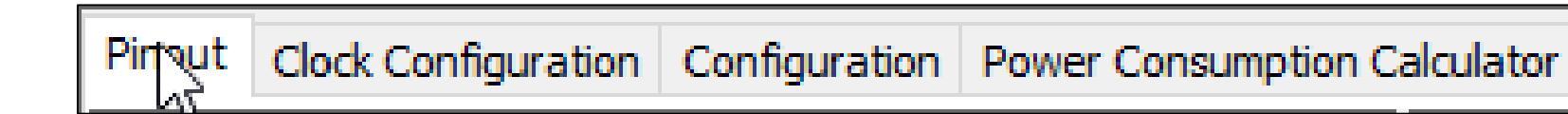






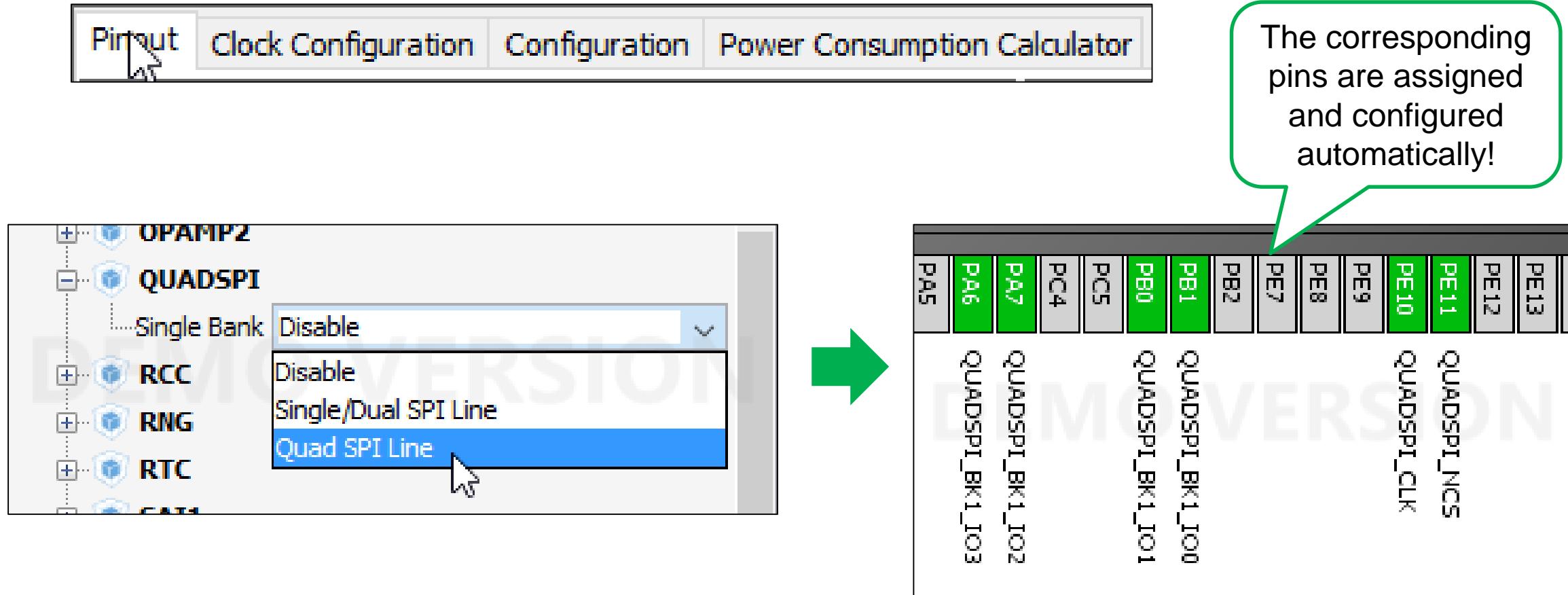
# 1 Configure QUADSPI interface

18



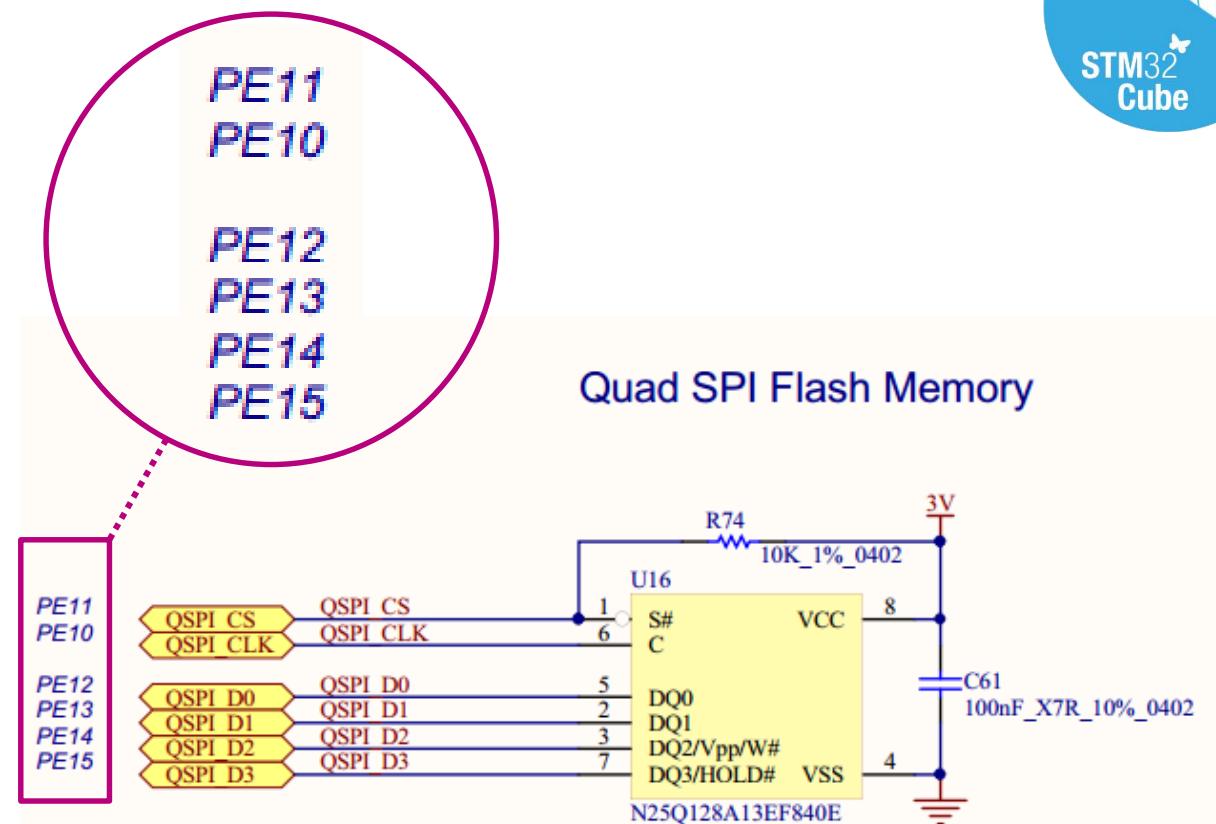
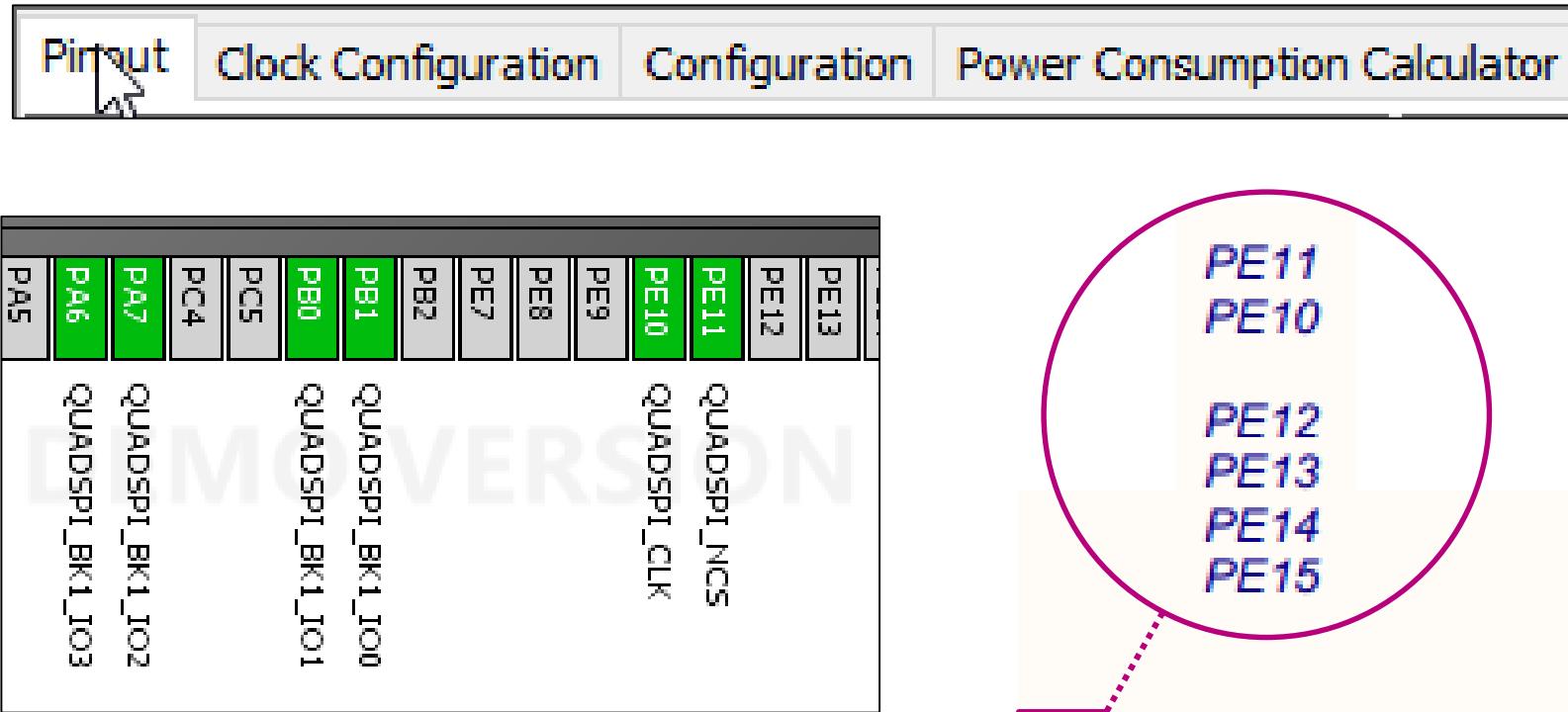
# 1 Configure QUADSPI interface

19



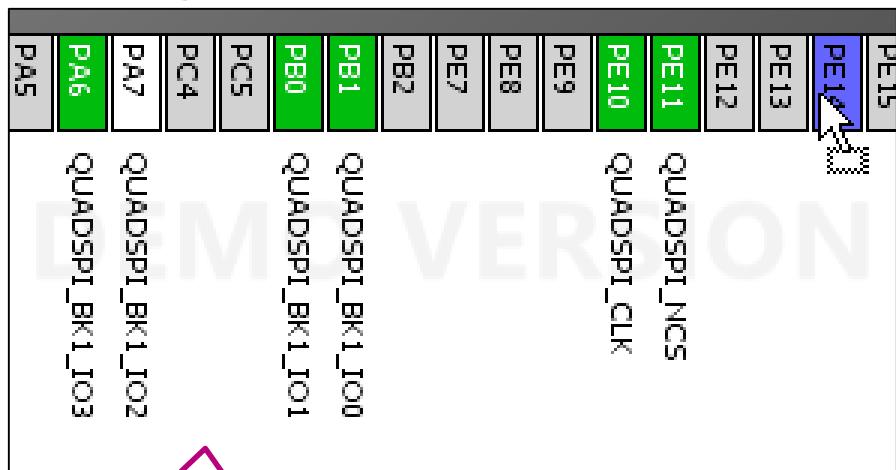
# 1 QUADSPI interface pins remapping

20



# 1 QUADSPI interface pins remapping

21

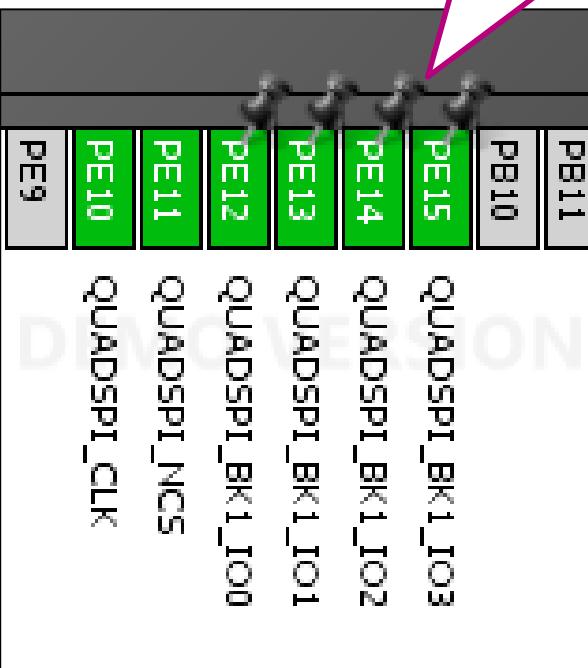
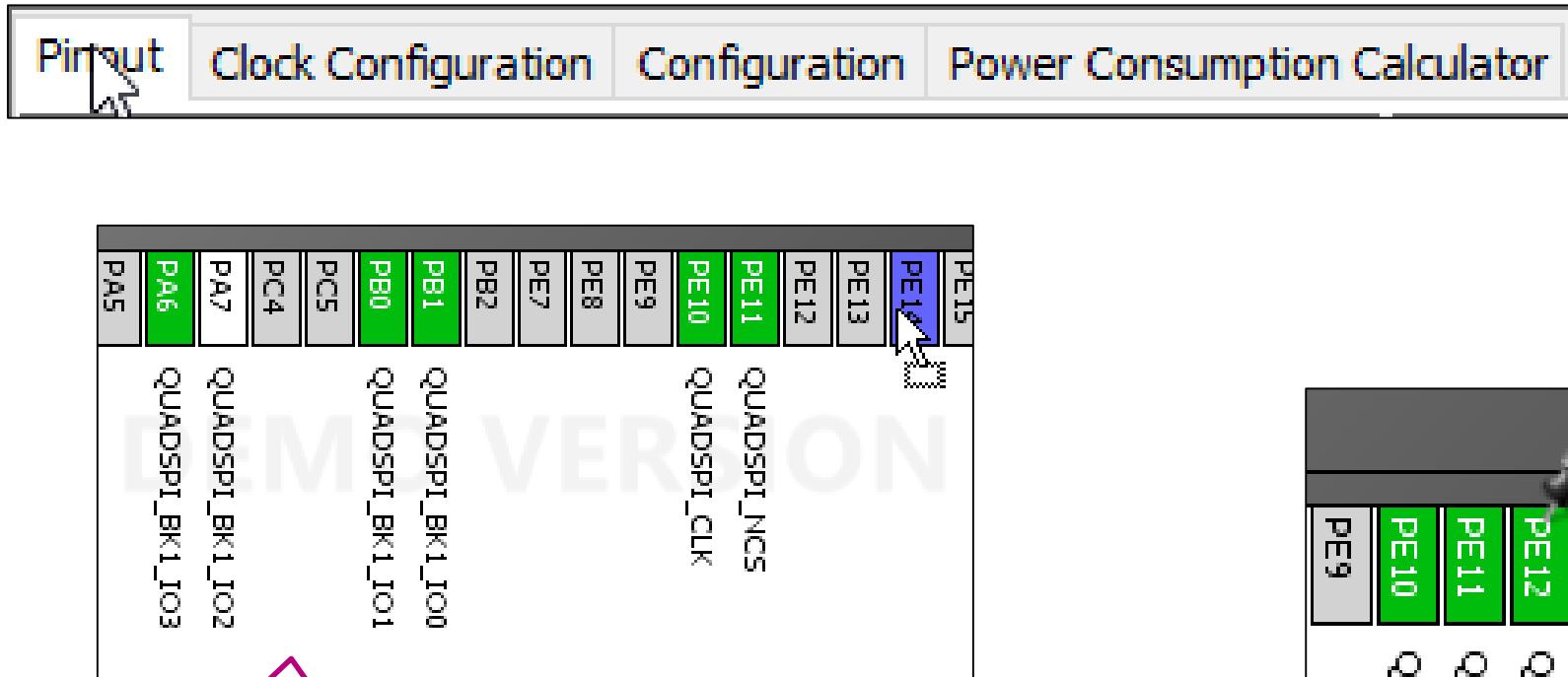


PA7 → PE14  
PA6 → PE15  
PB0 → PE13  
PB1 → PE12

Hold CTRL key → click  
on the pin → move it to  
the other possible  
configuration

# 1 QUADSPI interface pins remapping

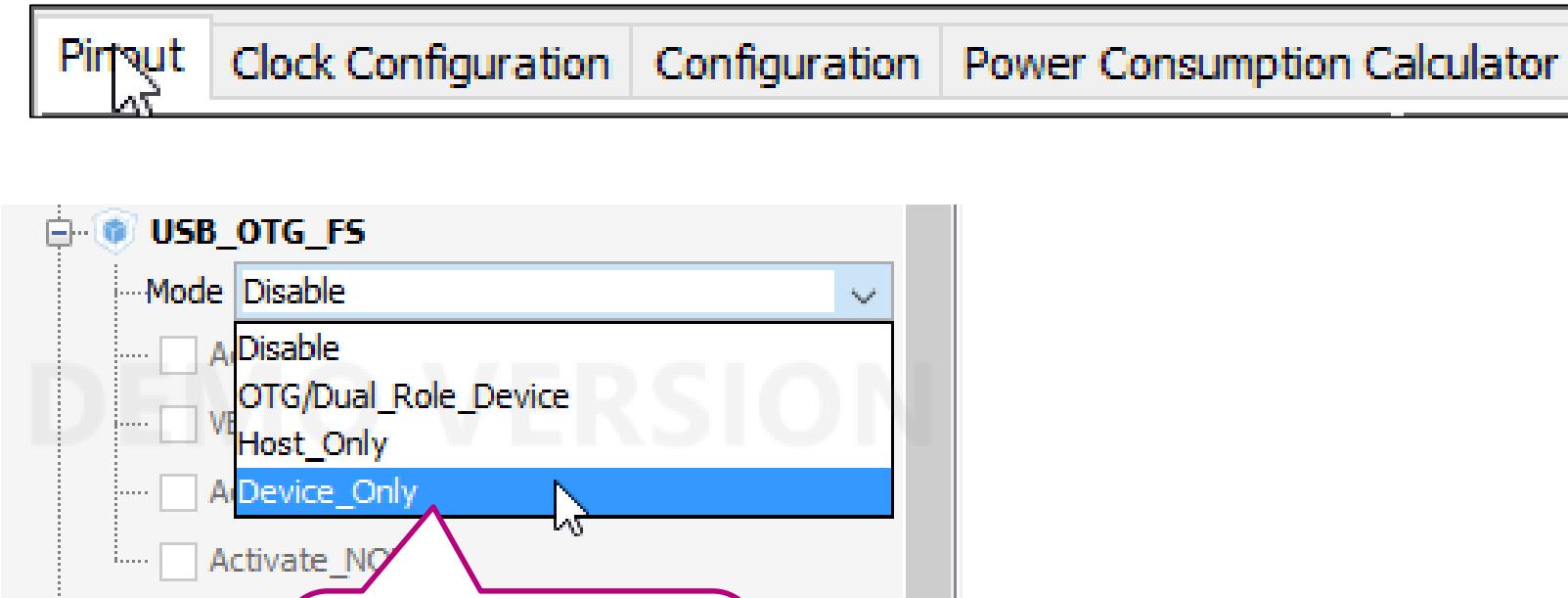
22



Desired configuration

# 1 Configure USB\_OTG\_FS interface

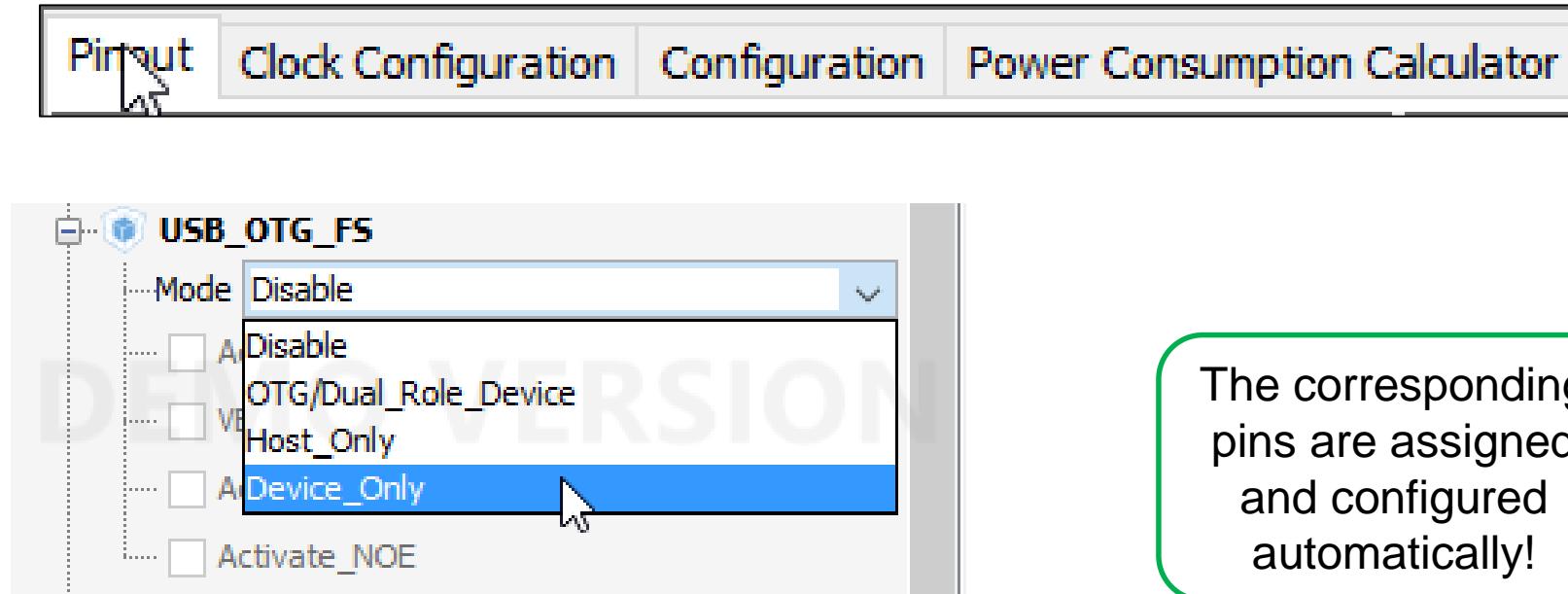
23



Under  
USB\_OTG\_FS  
peripheral select  
**Device\_Only**

# 1 Configure USB\_OTG\_FS interface

24



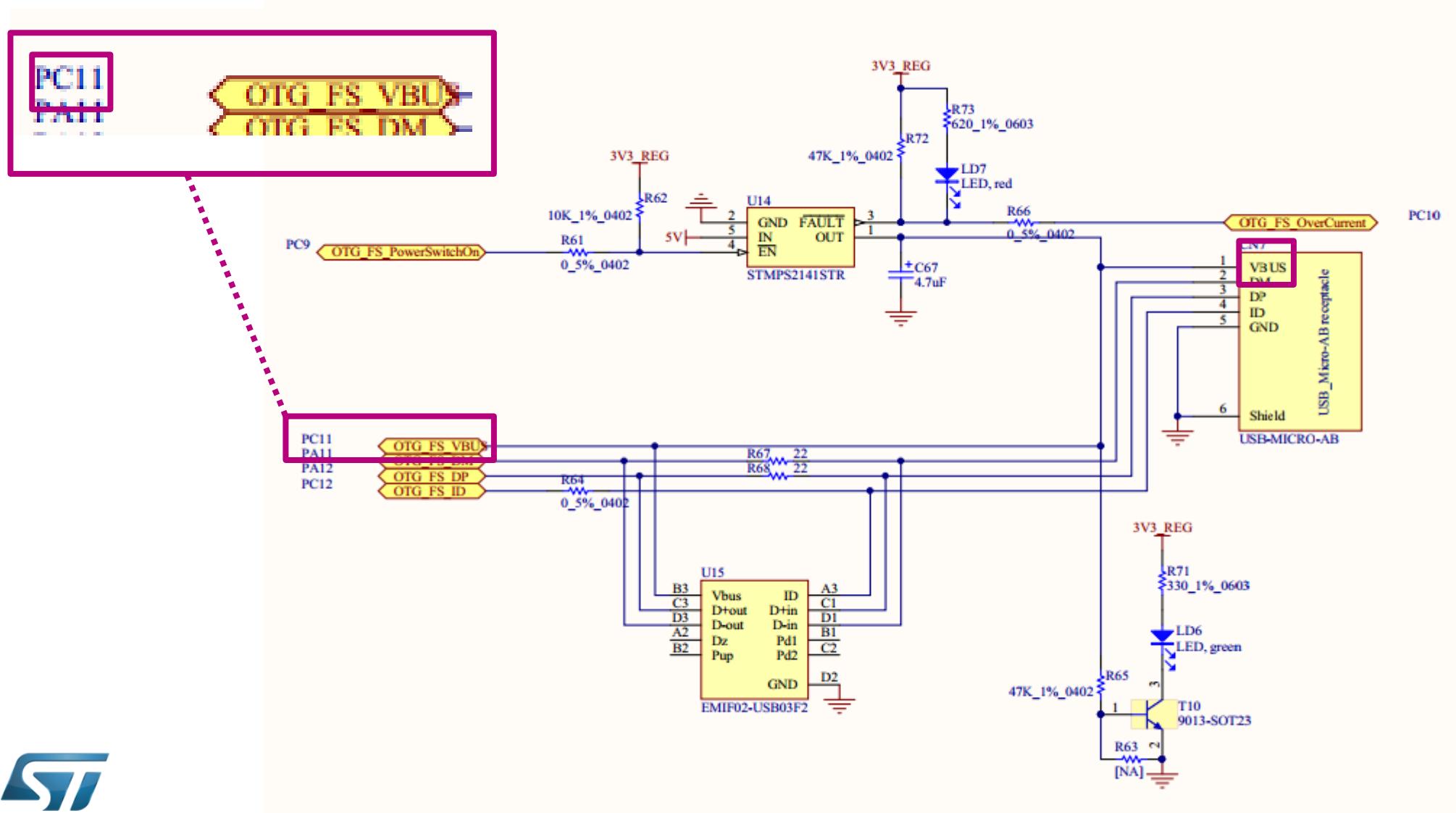
The corresponding  
pins are assigned  
and configured  
automatically!



USB\_OTG\_FS\_DP  
USB\_OTG\_FS\_DM

# 1 Configure USB\_OTG\_FS interface

25



# 1 Configure USB\_OTG\_FS interface

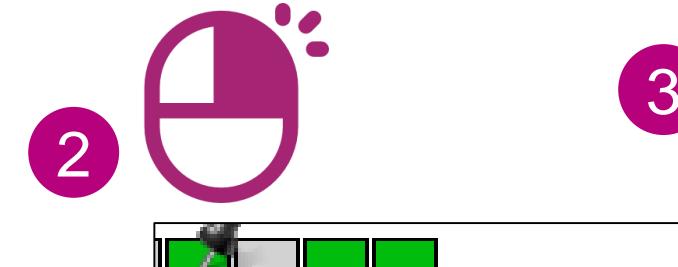
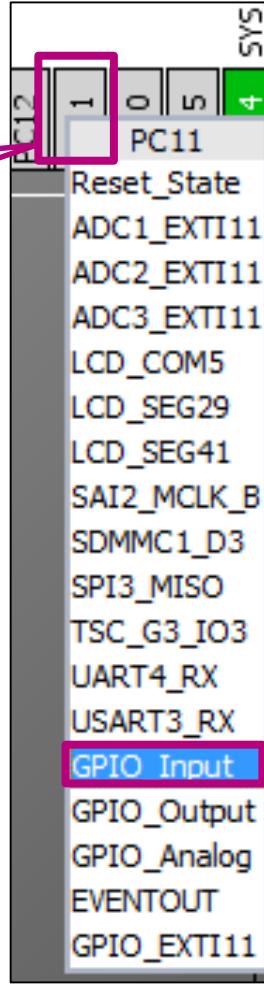
26



PC11



1



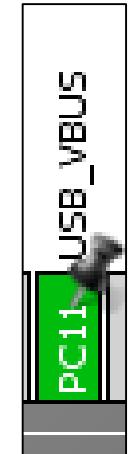
2

Select Enter User Label

3

GPIO\_Input (PC11) **USB\_VBUS**

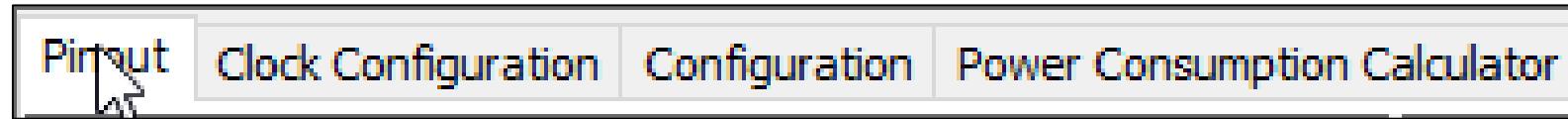
Enter **USB\_VBUS**



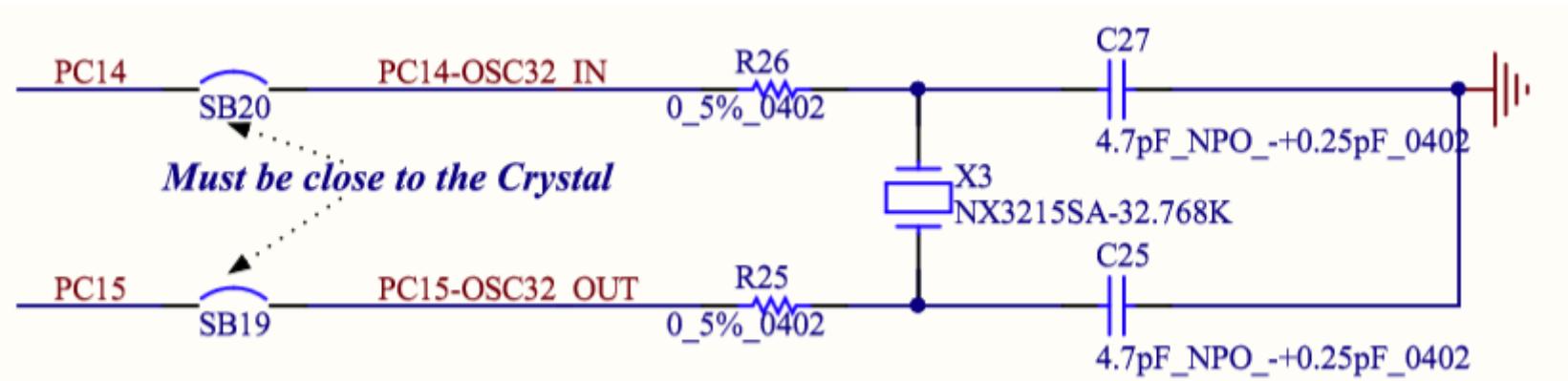
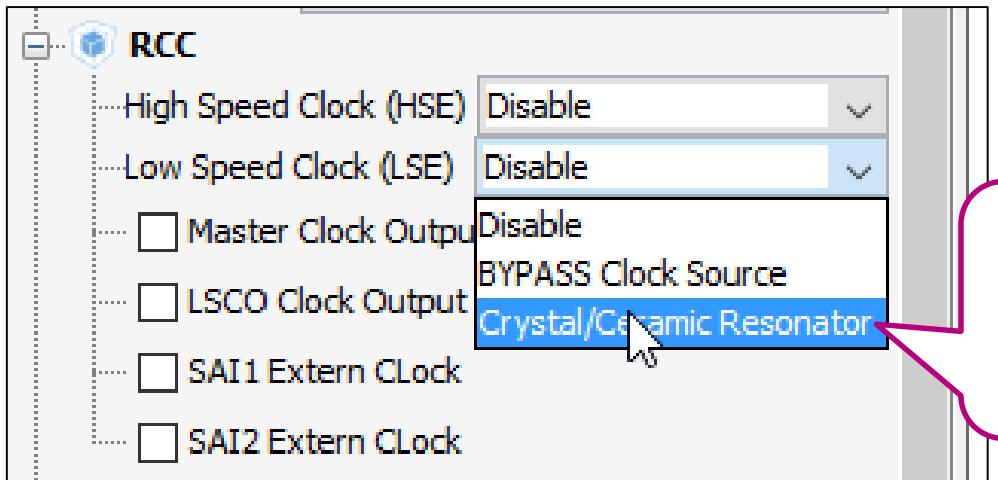
GPIO\_Input

Make sure „Show user Label“ is checked

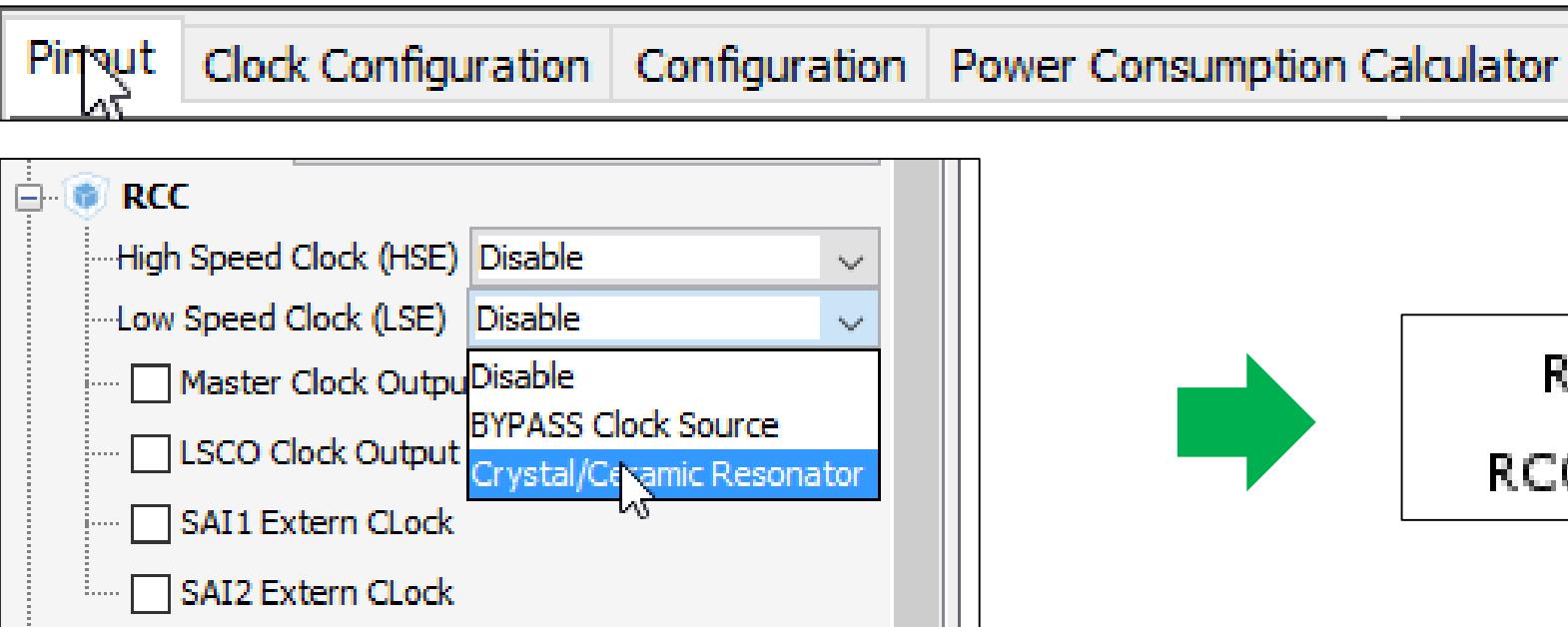




Under RCC Low Speed Clock (LSE)  
select **Crystal/Ceramic Resonator**

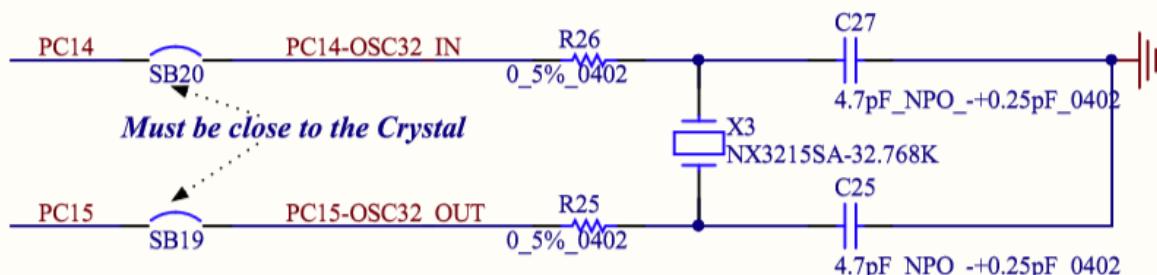
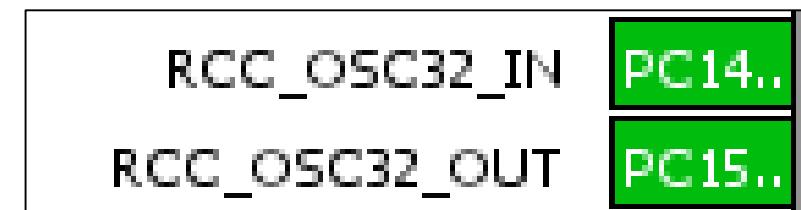


Pinout    Clock Configuration    Configuration    Power Consumption Calculator

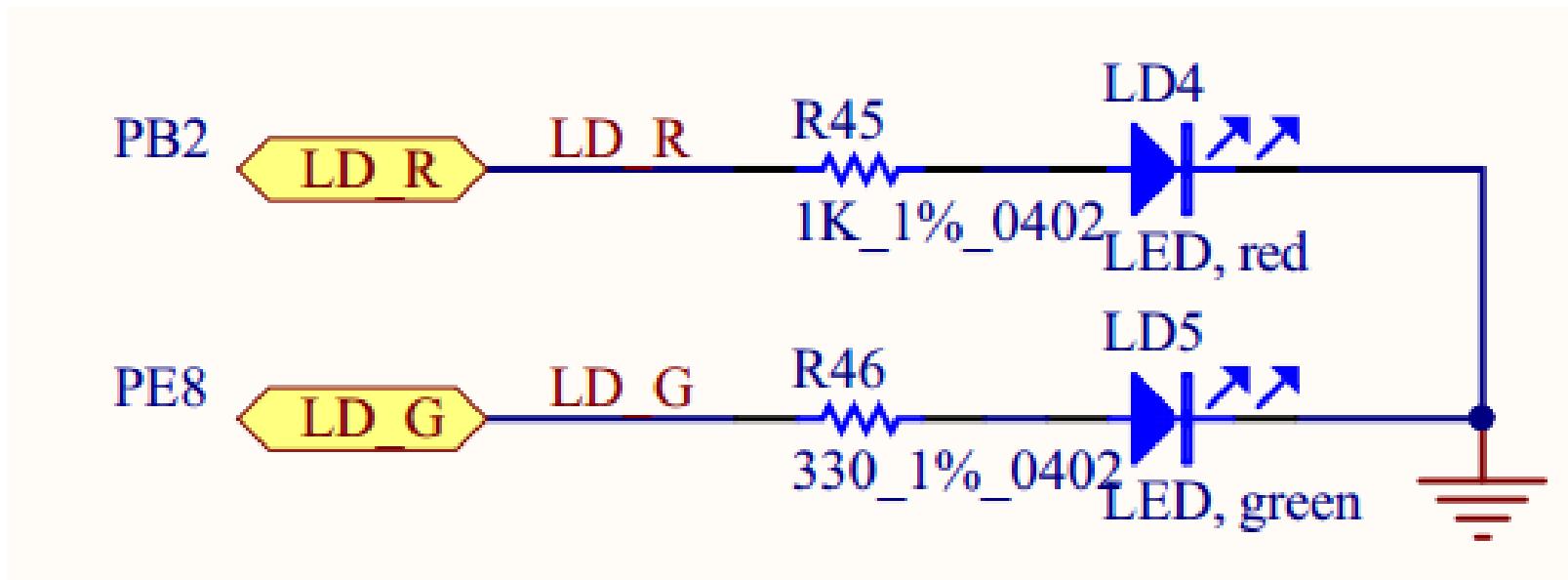


RCC

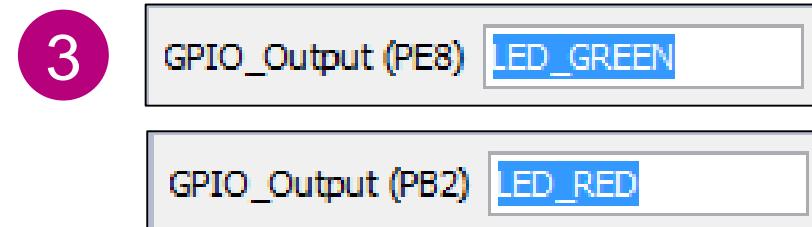
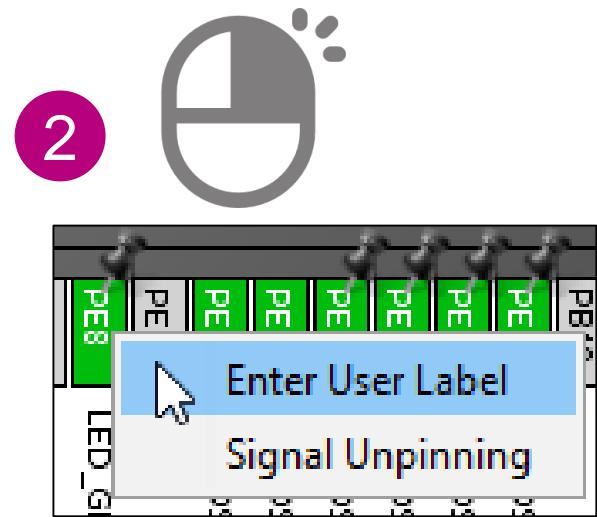
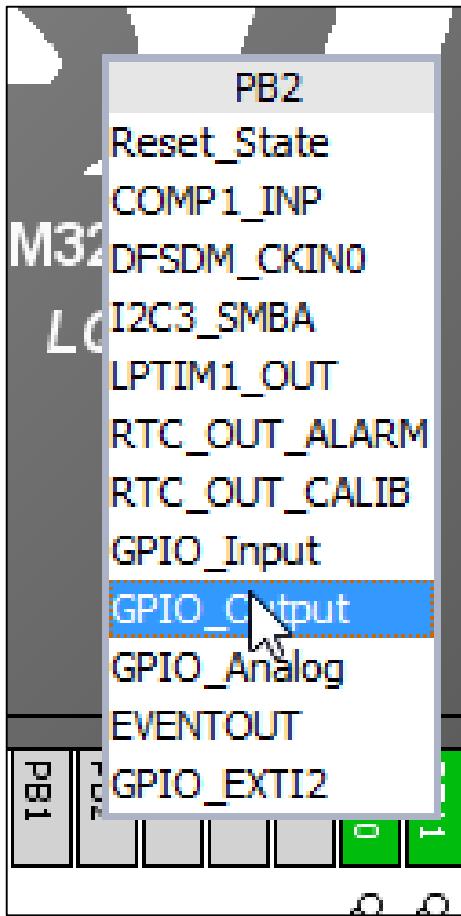
- High Speed Clock (HSE)
- Low Speed Clock (LSE)   
 Master Clock Output  
 BYPASS Clock Source  
 Crystal/Ceramic Resonator
- Master Clock Output
- LSCO Clock Output
- SAI1 Extern Clock
- SAI2 Extern Clock



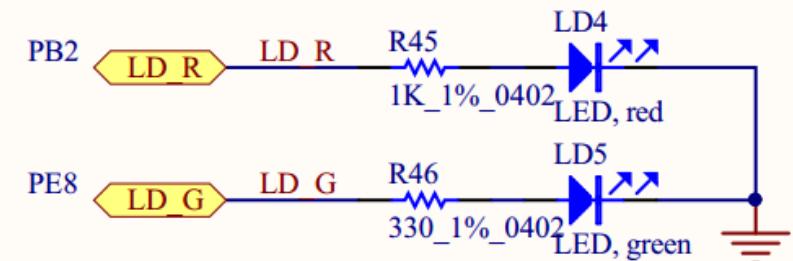
The corresponding pins are assigned and configured automatically!



# Configure LED pins



LED\_GREEN for PE8  
LED\_RED for PB2

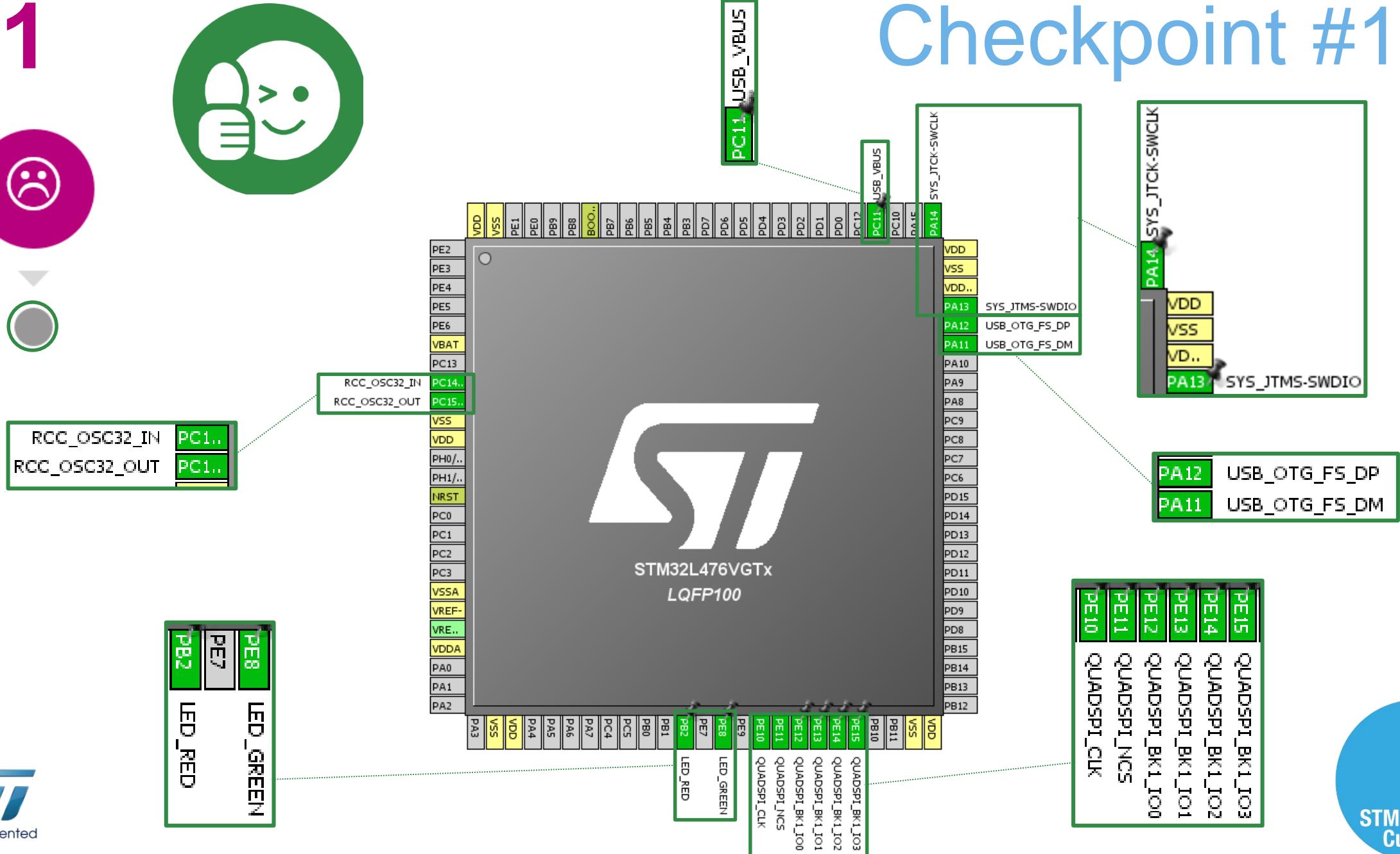


1



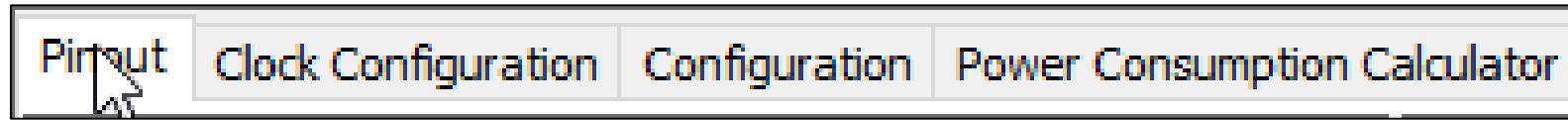
# Checkpoint #1

31

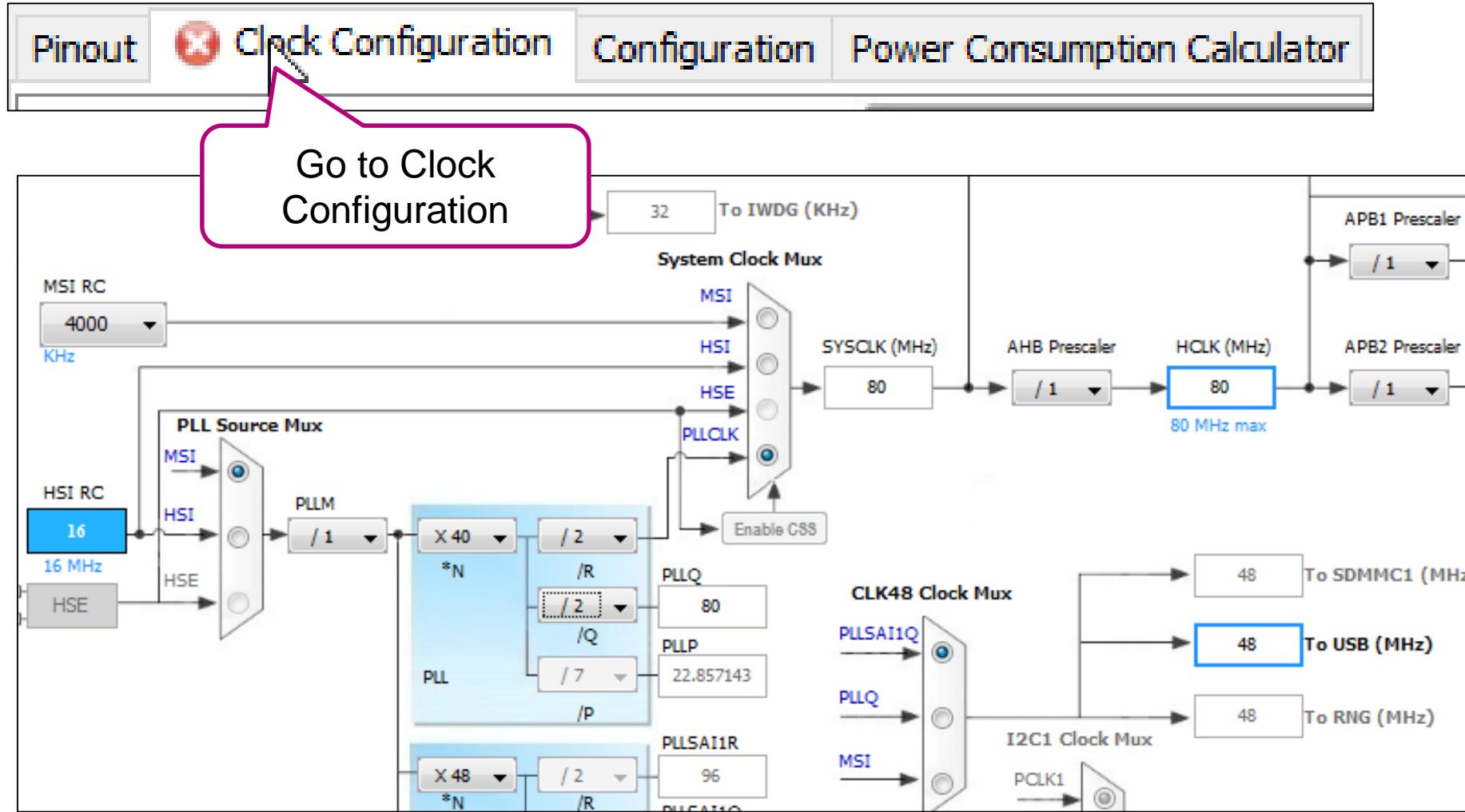


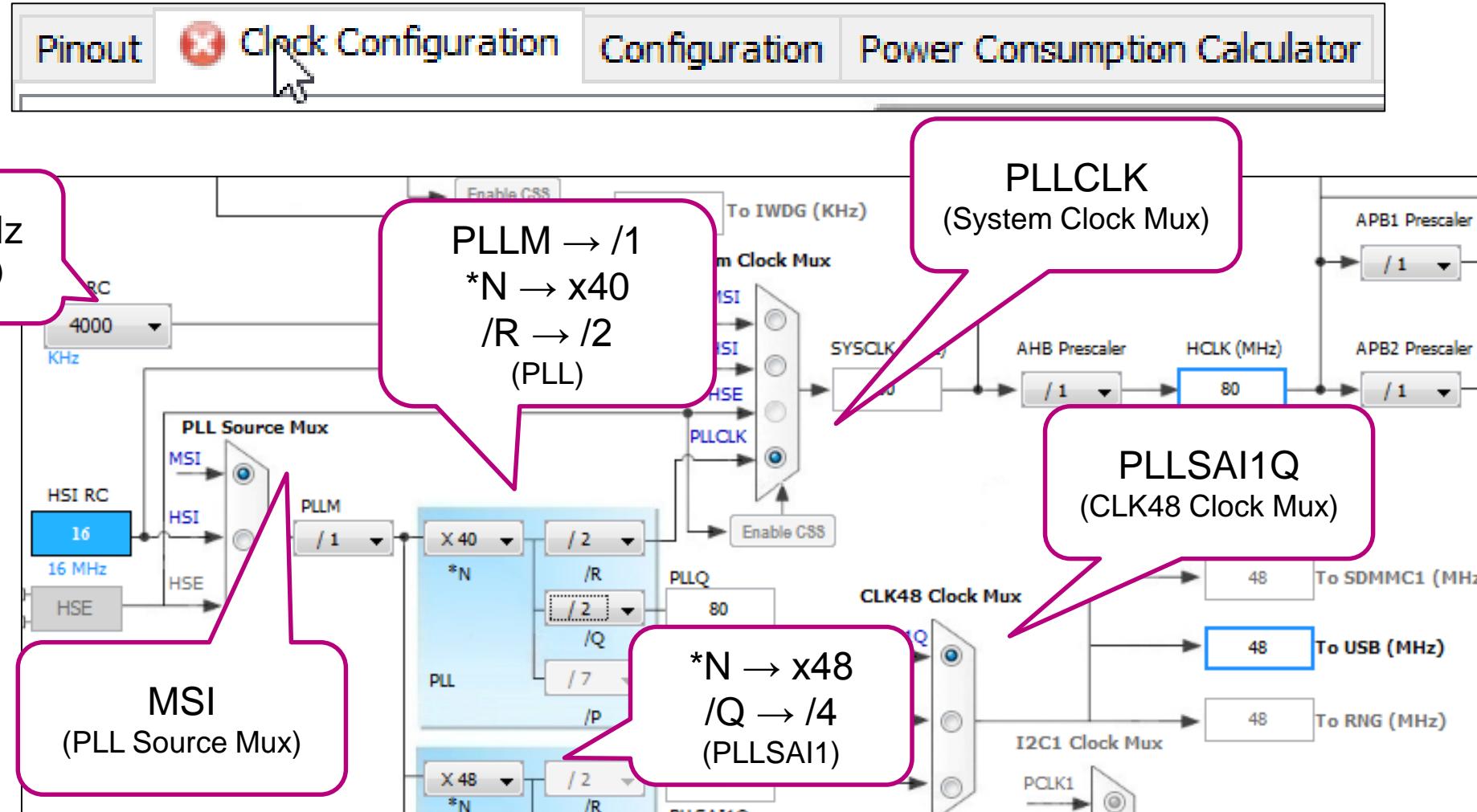
# 1 Configure USB\_DEVICE stack

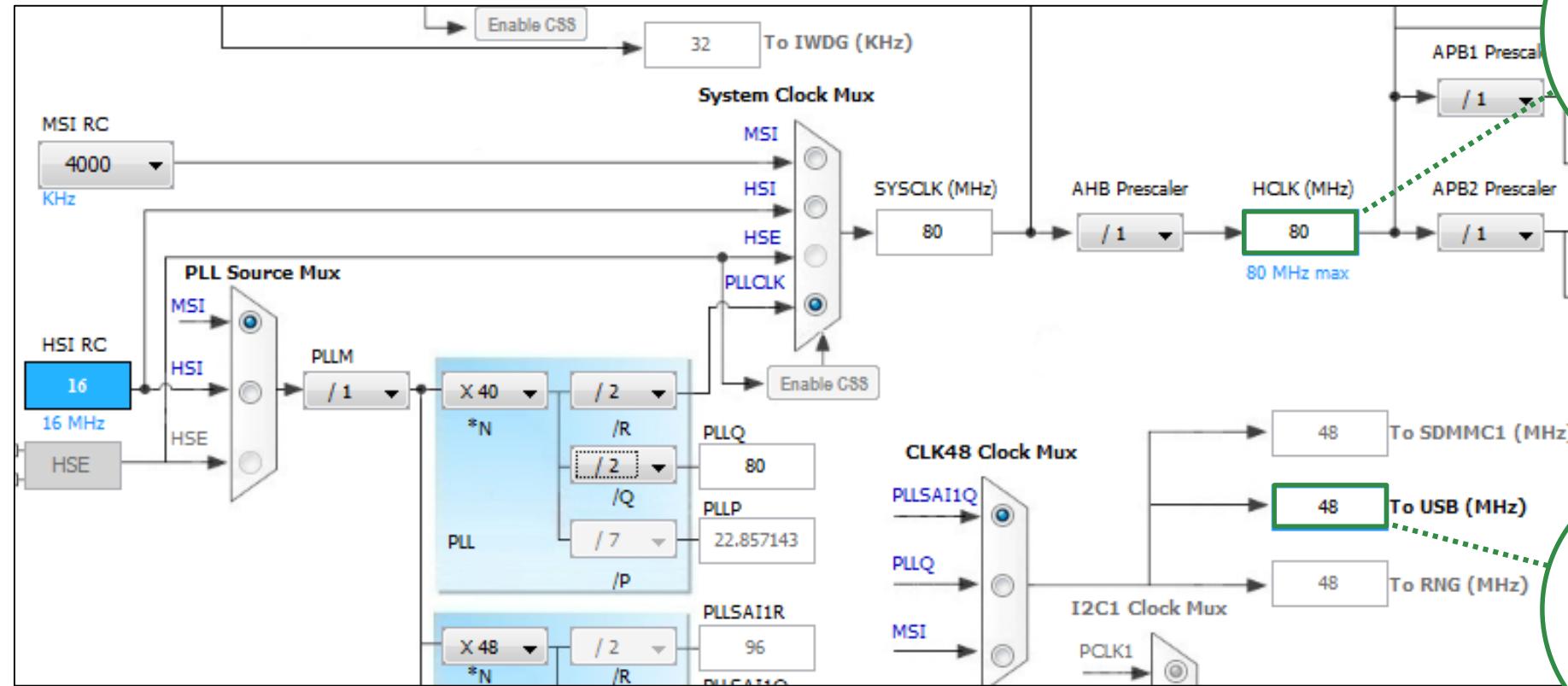
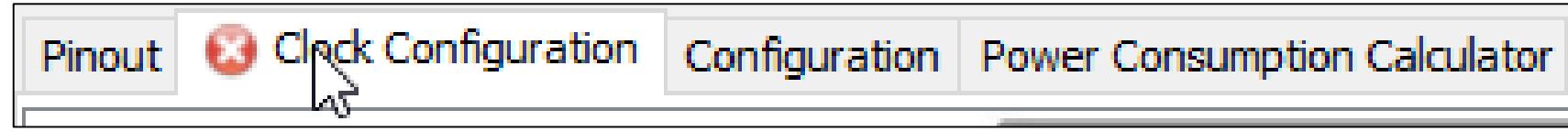
32



Under USB\_DEVICE  
MiddleWare select **Mass  
Storage Class**



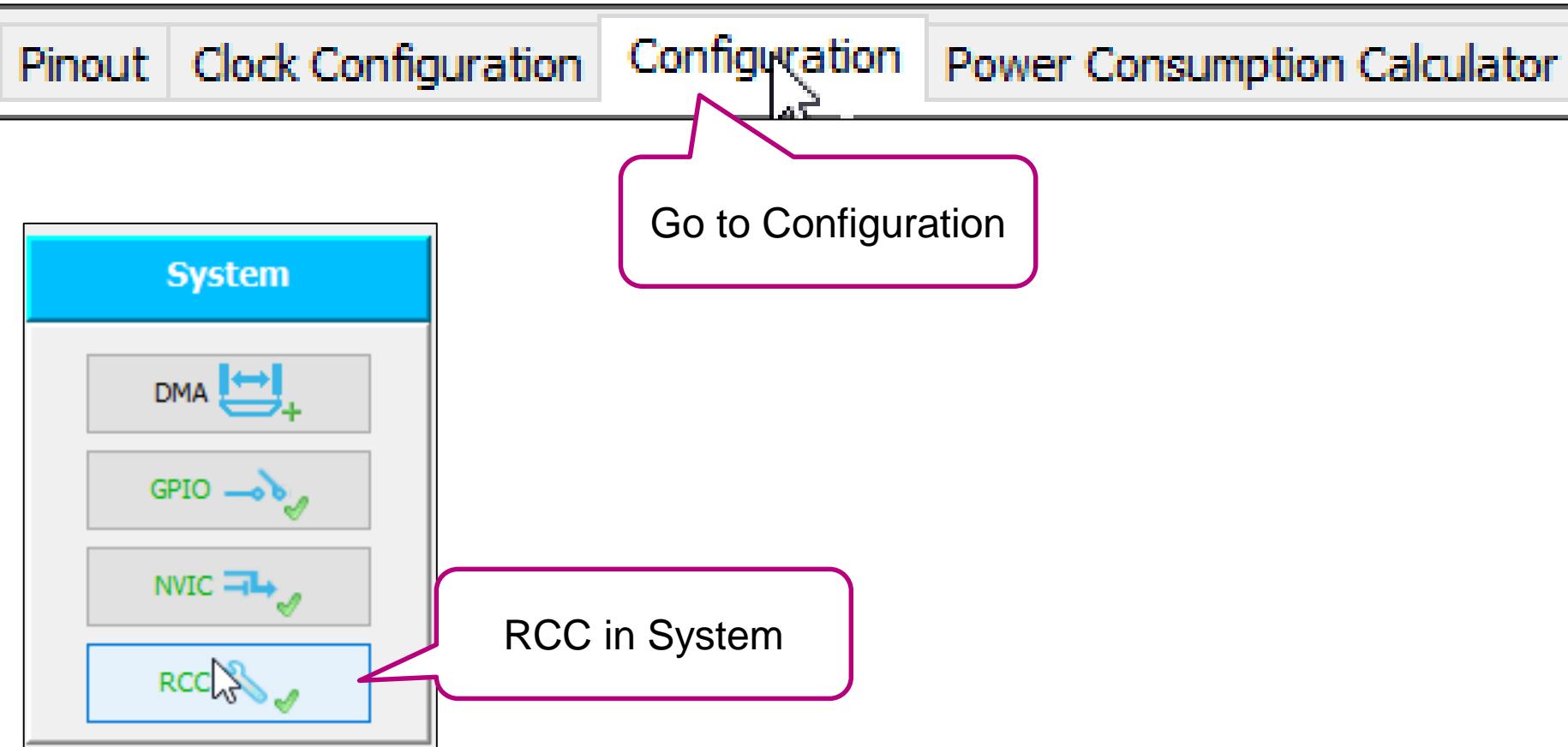


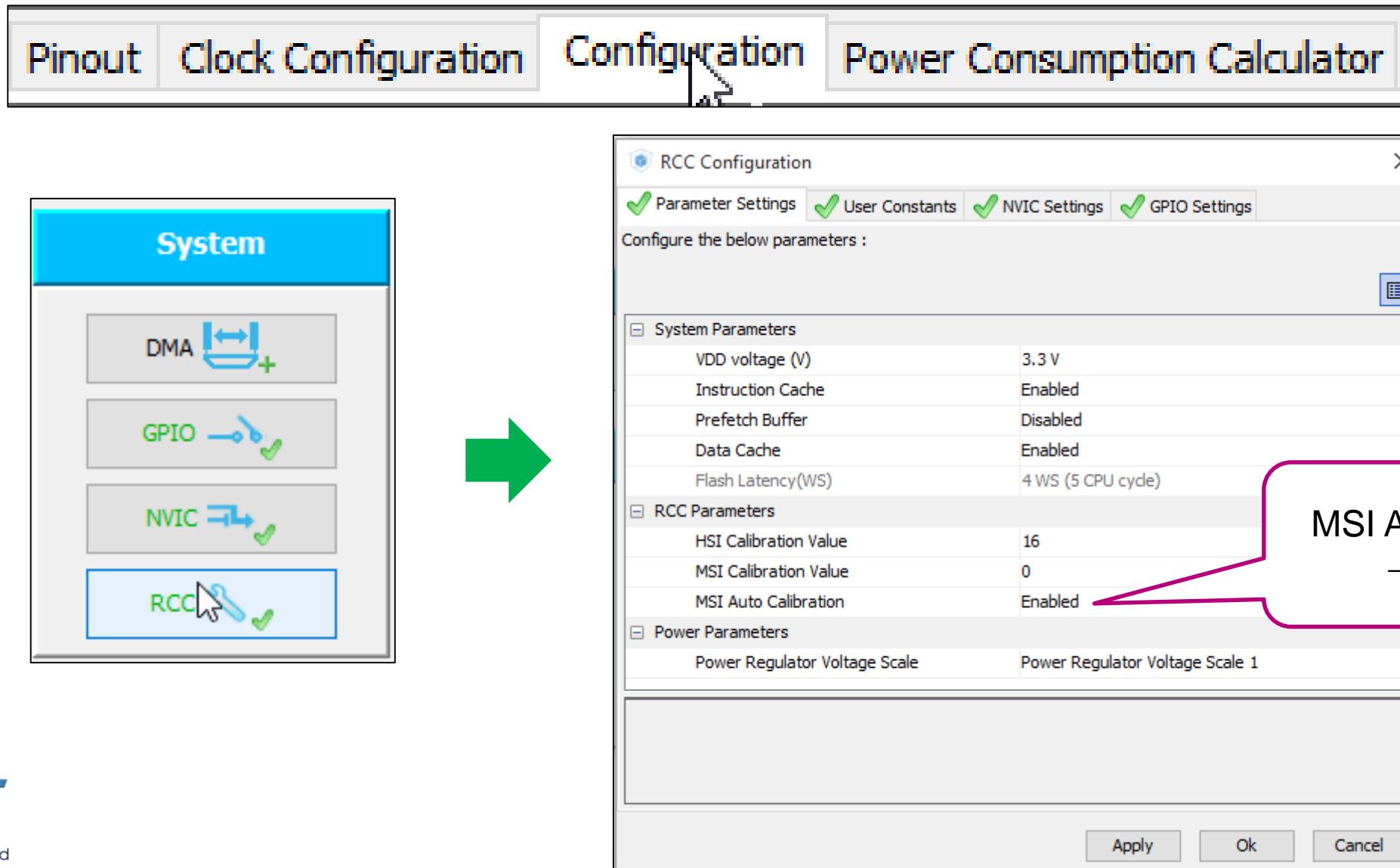


80MHz  
(HCLK)



48MHz  
(To USB)





Pinout Clock Configuration Configuration Power Consumption Calculator

System

DMA

GPIO

NVIC

RCC

RCC Configuration

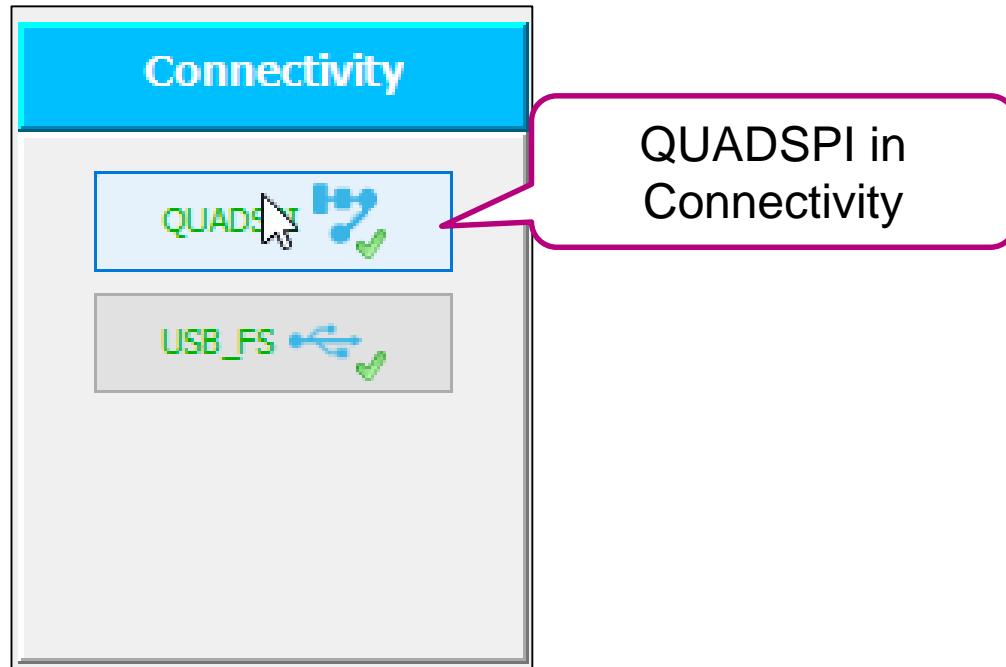
Parameter Settings User Constants NVIC Settings GPIO Settings

Configure the below parameters :

System Parameters	
VDD voltage (V)	3.3 V
Instruction Cache	Enabled
Prefetch Buffer	Disabled
Data Cache	Enabled
Flash Latency(WS)	4 WS (5 CPU cycle)
RCC Parameters	
HSI Calibration Value	16
MSI Calibration Value	0
MSI Auto Calibration	Enabled
Power Parameters	
Power Regulator Voltage Scale	Power Regulator Voltage Scale 1

MSI Auto Calibration → Enabled

Apply Ok Cancel



Pinout Clock Configuration Configuration Power Consumption Calculator

Connectivity

QUADSPI

USB\_FS

QUADSPI Configuration

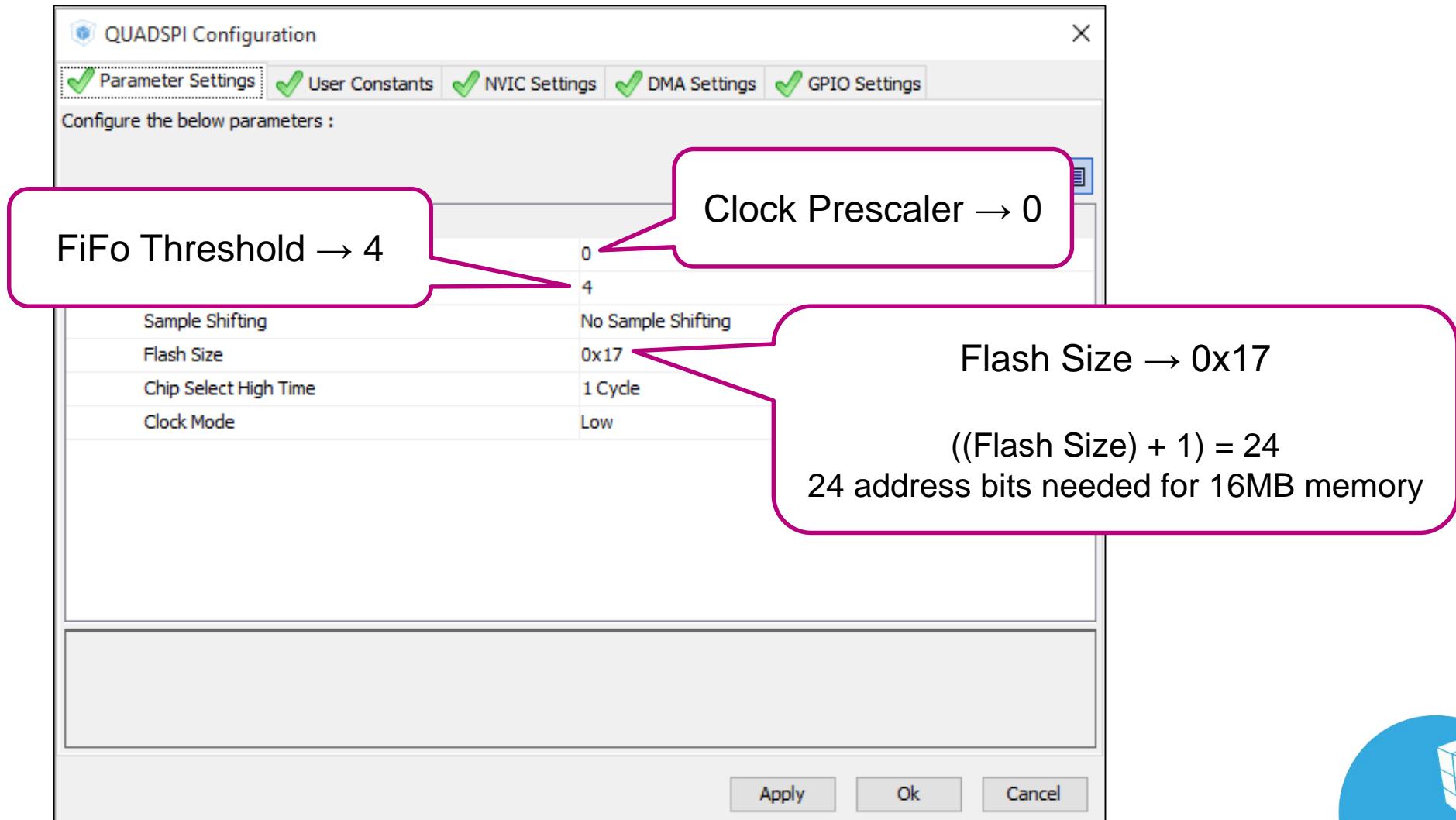
Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Configure the below parameters :

General Parameters

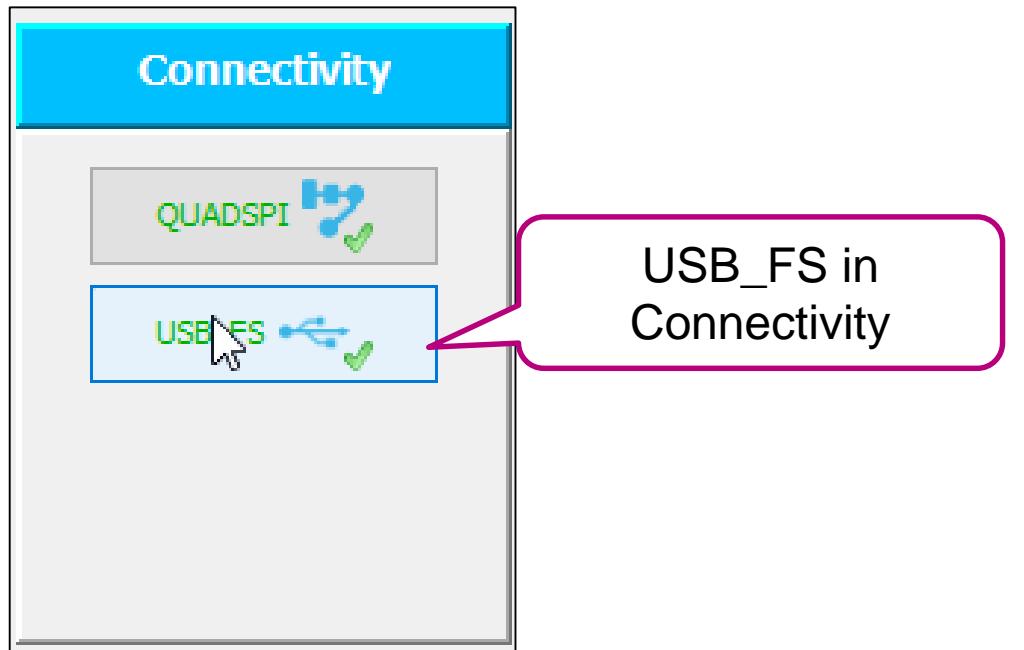
Clock Prescaler	255
Fifo Threshold	1
Sample Shifting	No Sample Shifting
Flash Size	1
Chip Select High Time	1 Cycle
Clock Mode	Low

Apply Ok Cancel



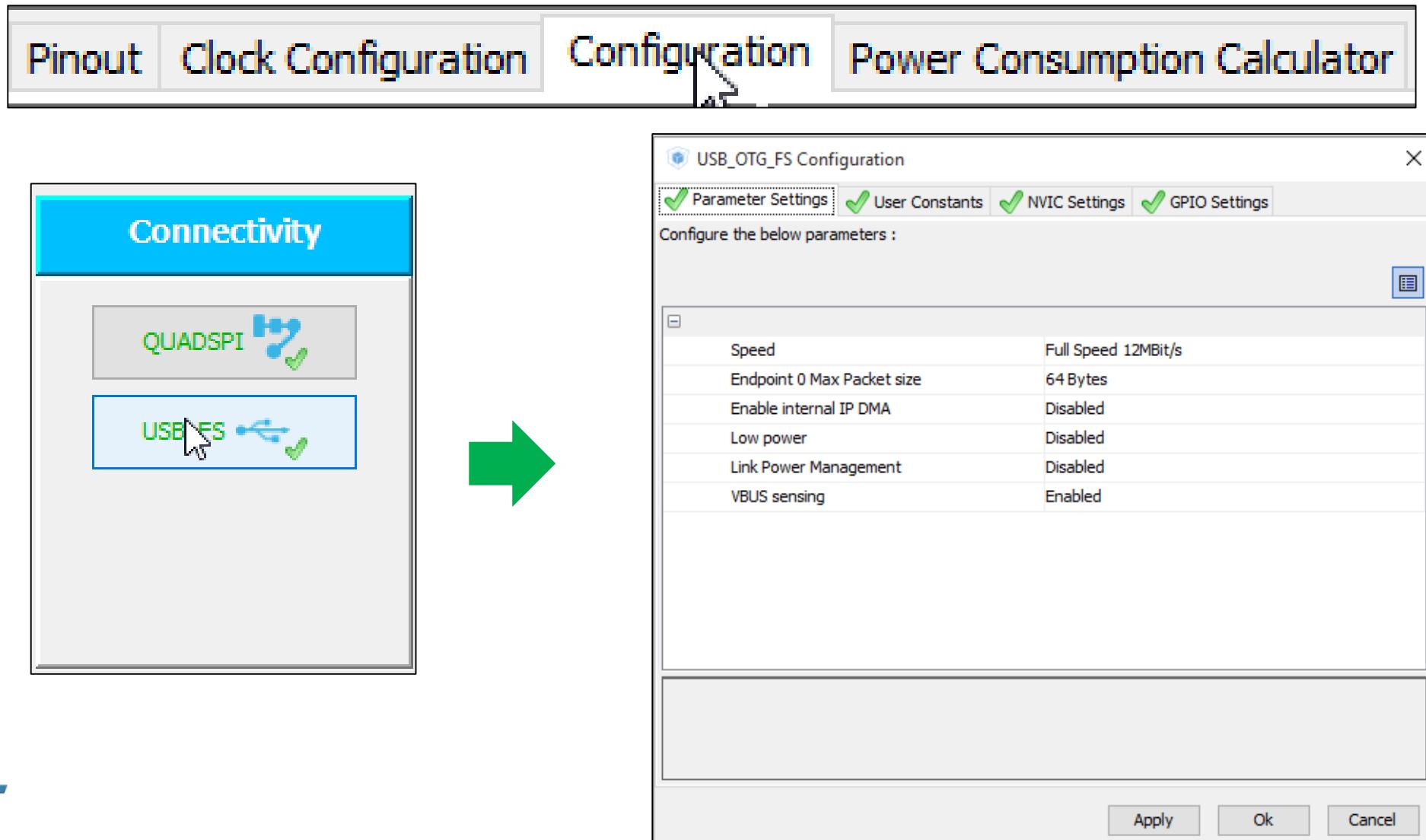
# 1 USB\_OTF\_FS Configuration

41

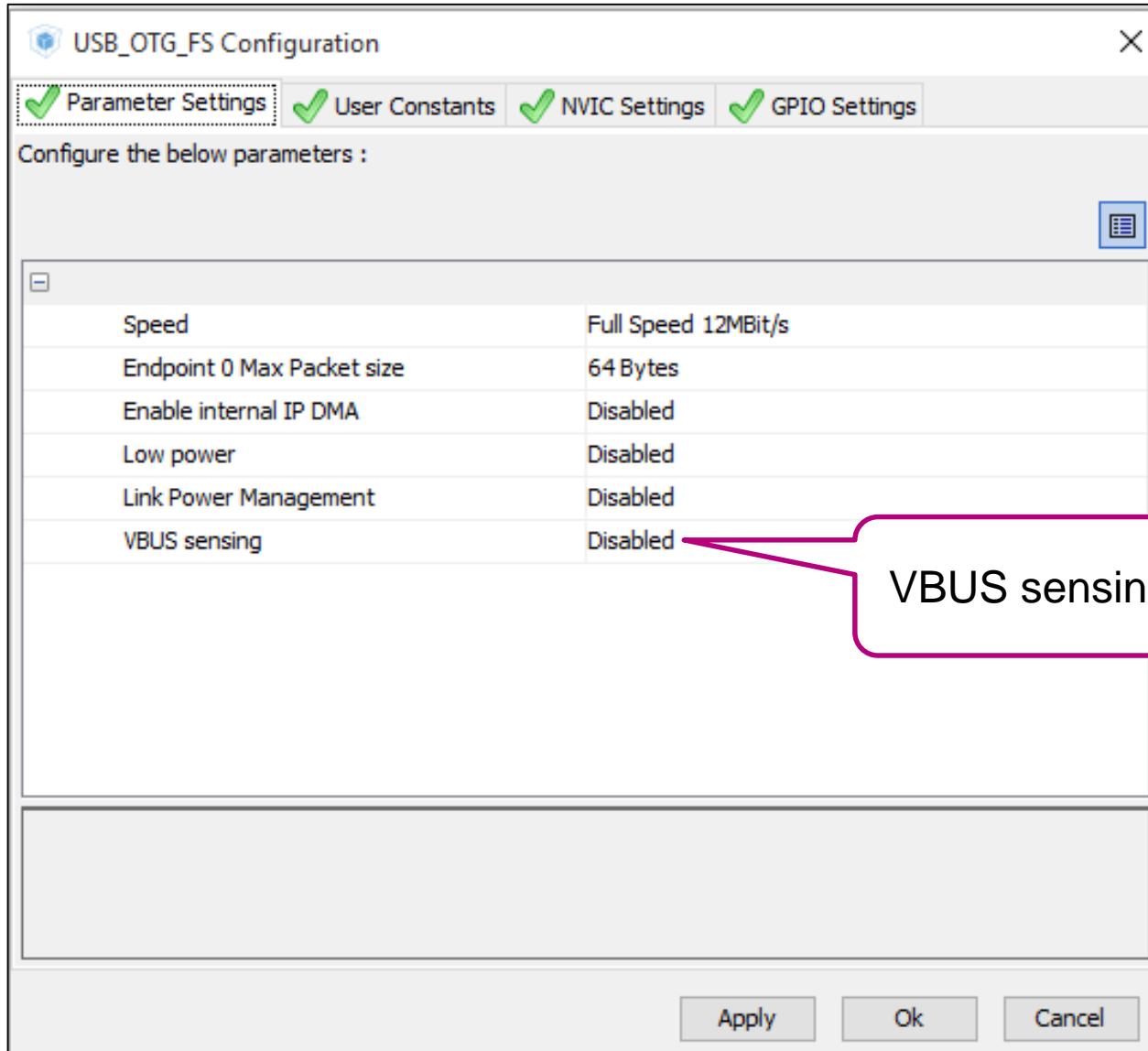


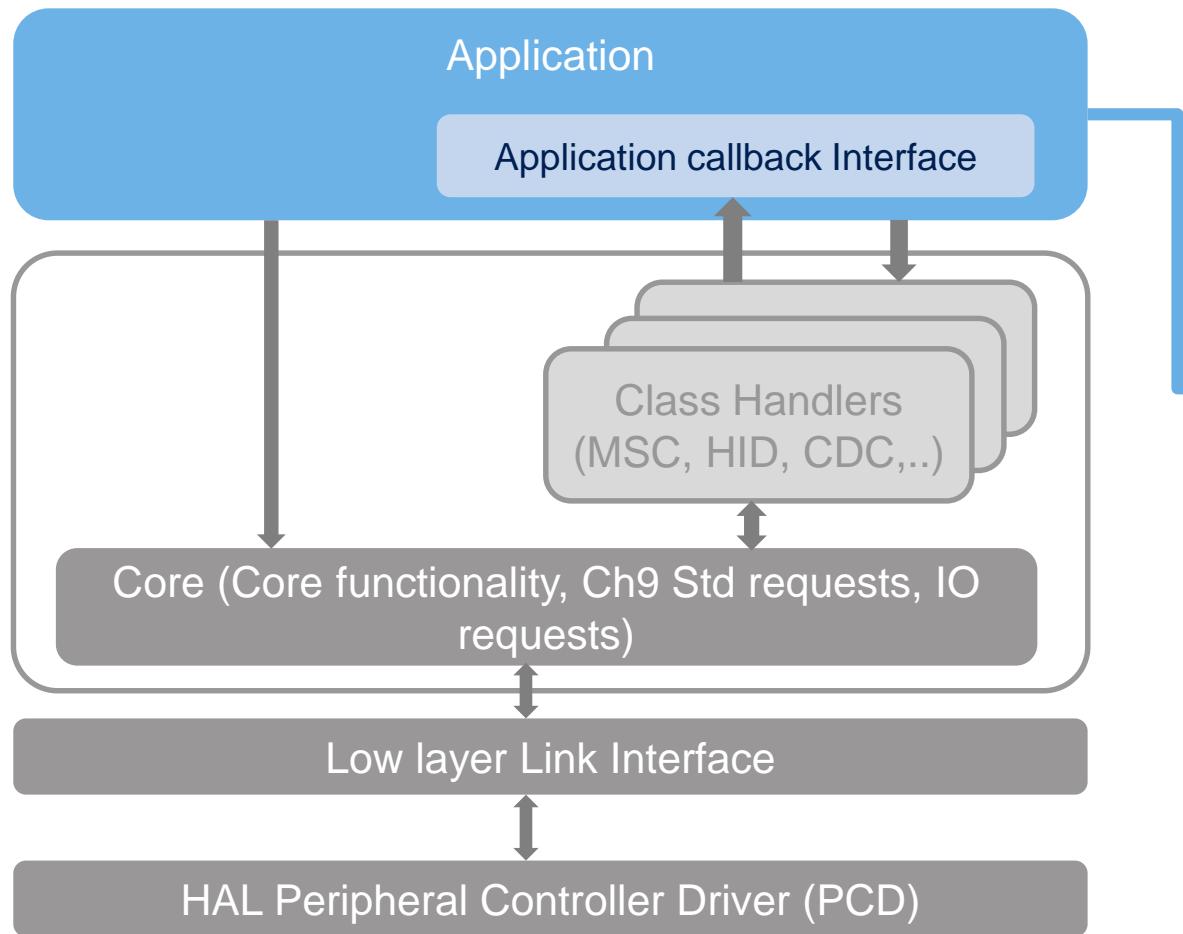
# 1 USB\_OTG\_FS Configuration

42



# 1 USB\_OTG\_FS configuration





- USB wrapper for SCSI commands



SCSI = Small Computer System Interface

#### SCSI command subset for USB

- Init
- Read
- Write
- Is Ready
- Get Capacity
- Get Max LUN
- Is Write Protected

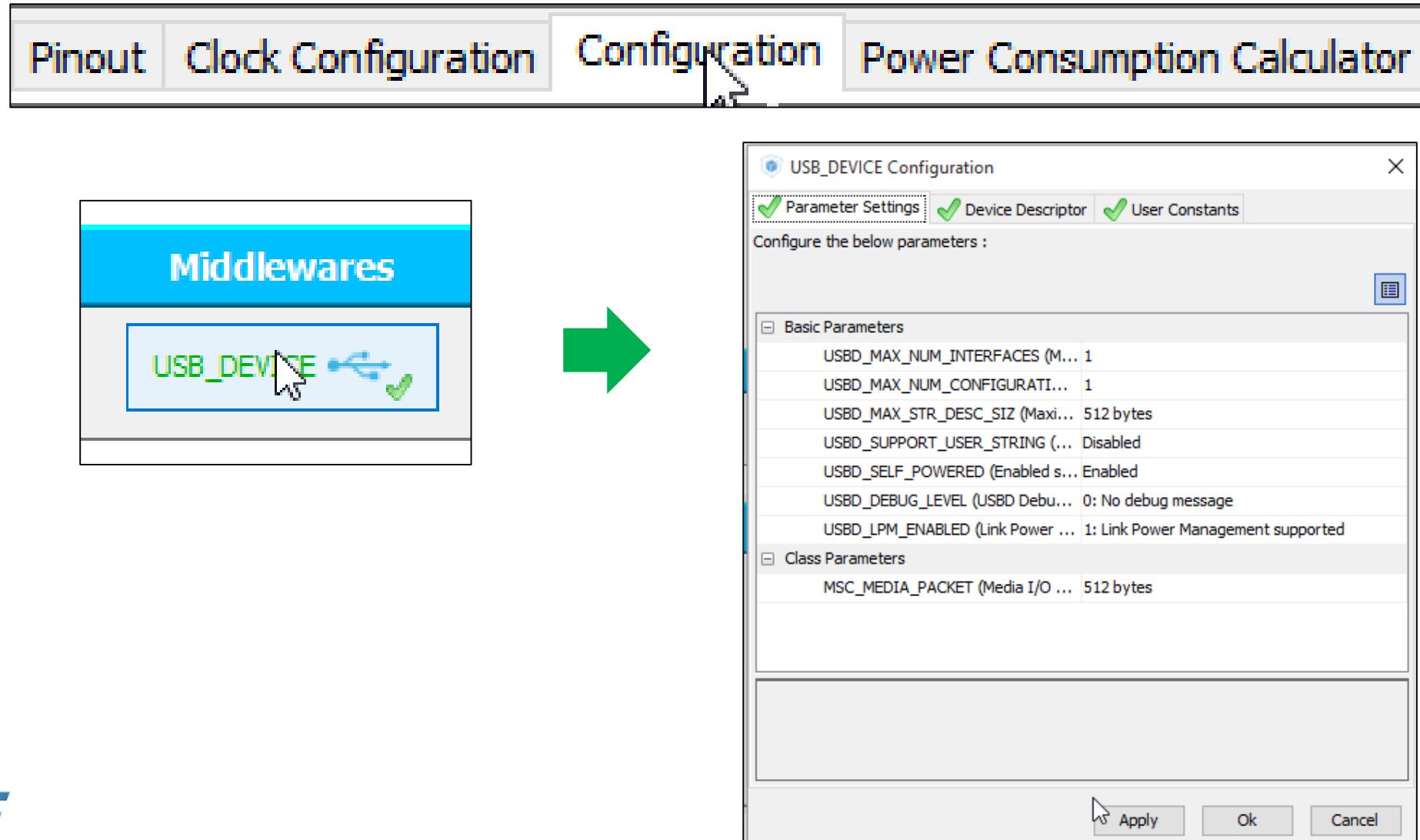
# 1 USB\_DEVICE MiddleWare configuration

45

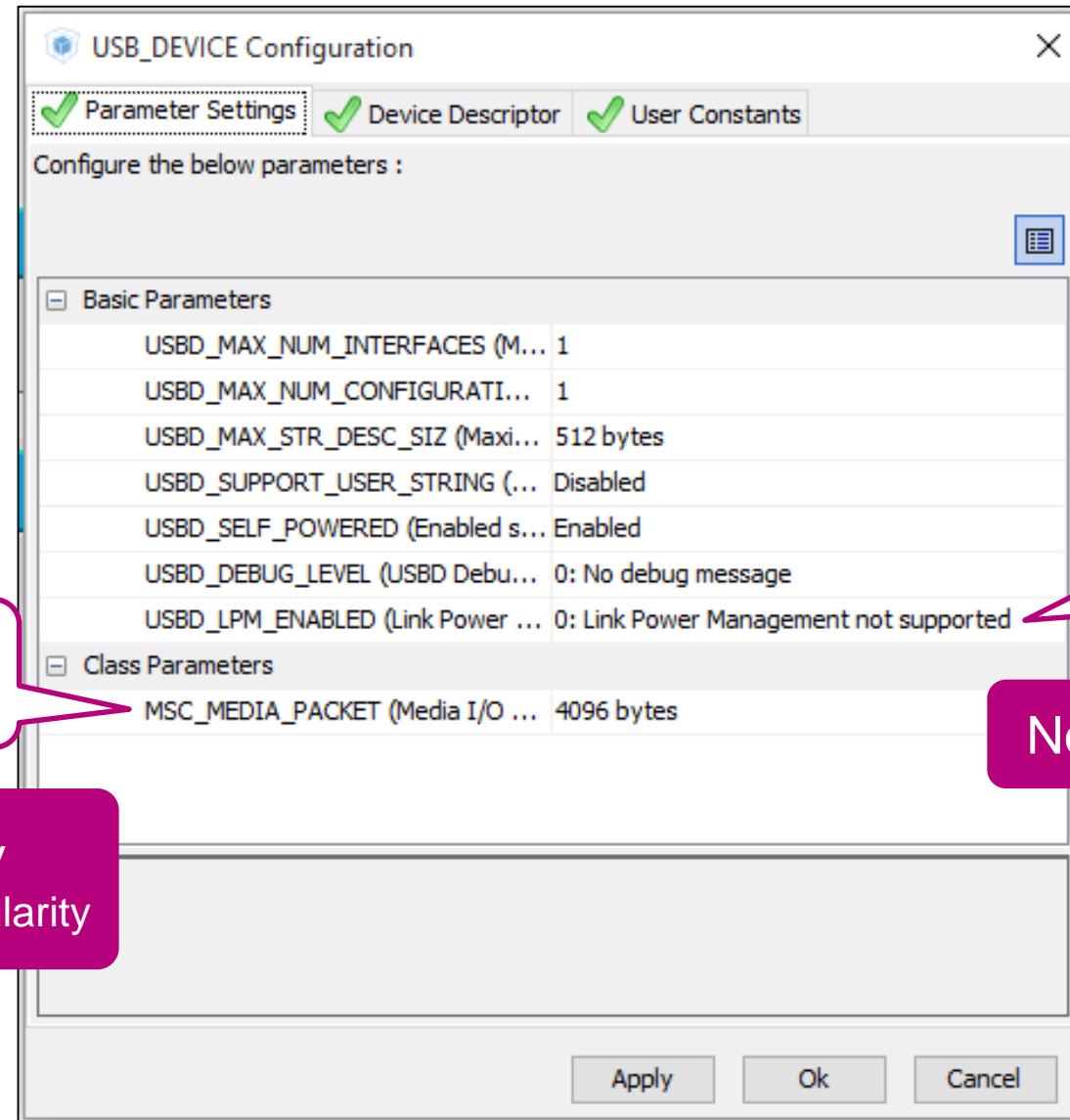


# 1 USB\_DEVICE MiddleWare configuration

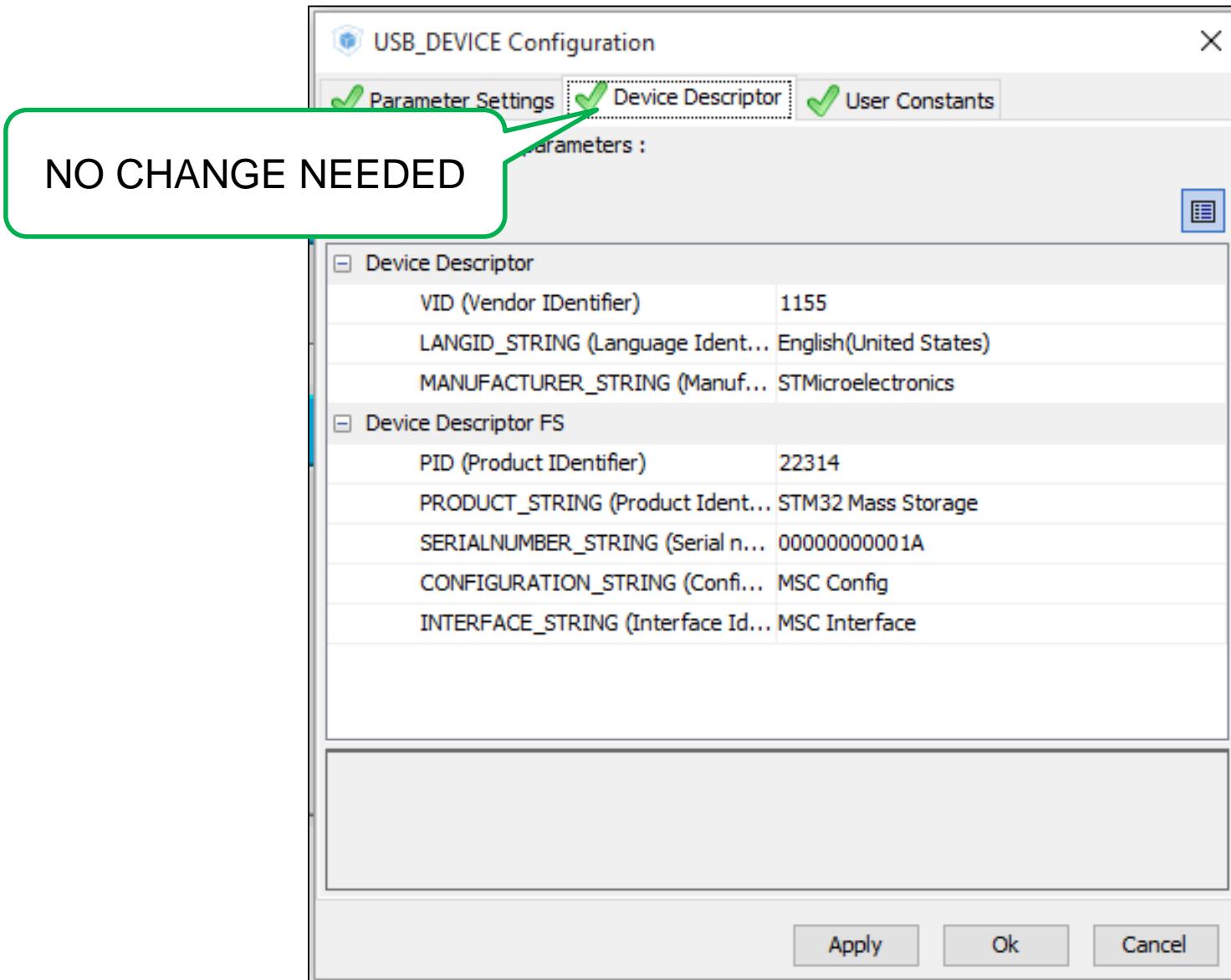
46

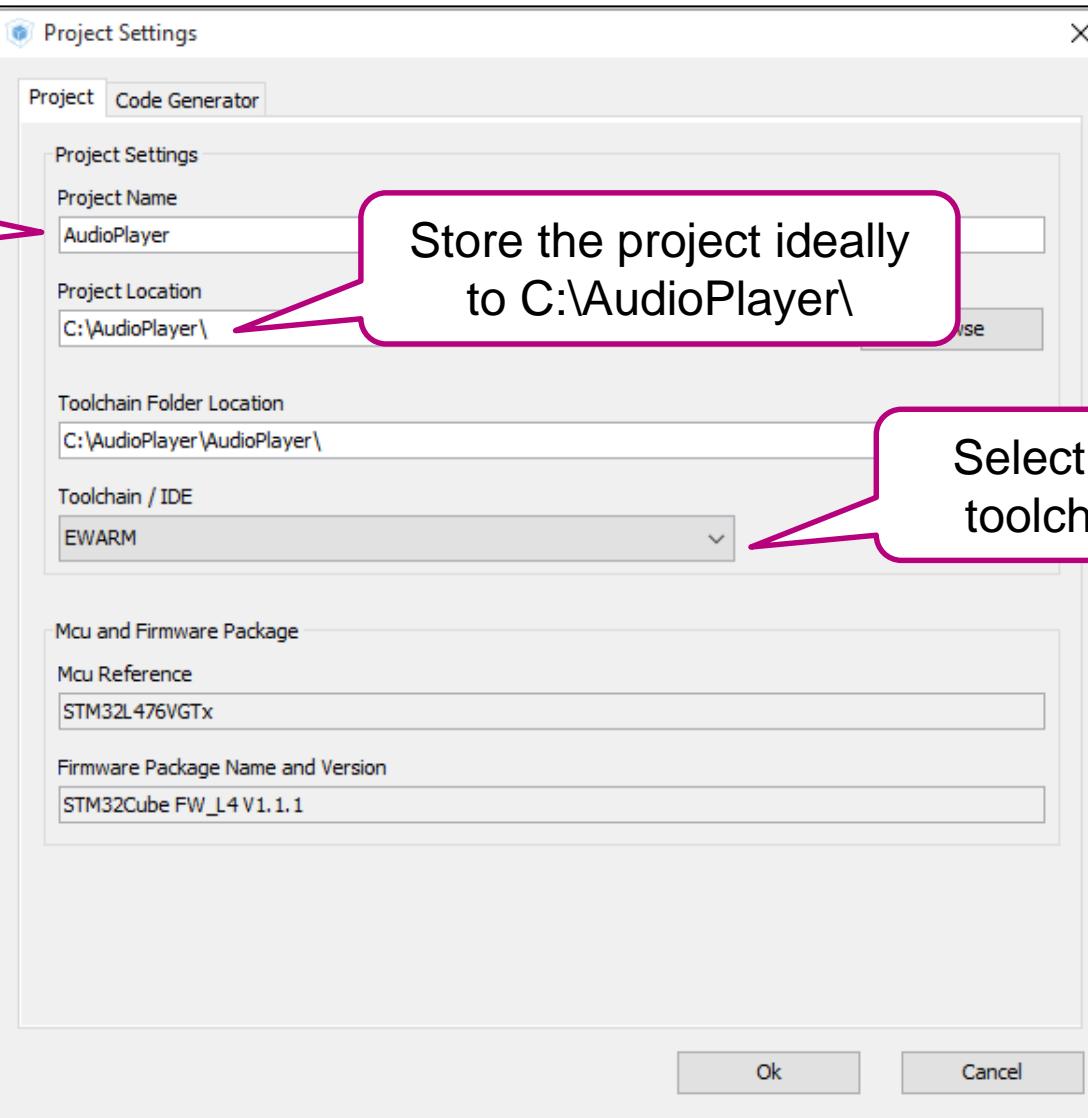


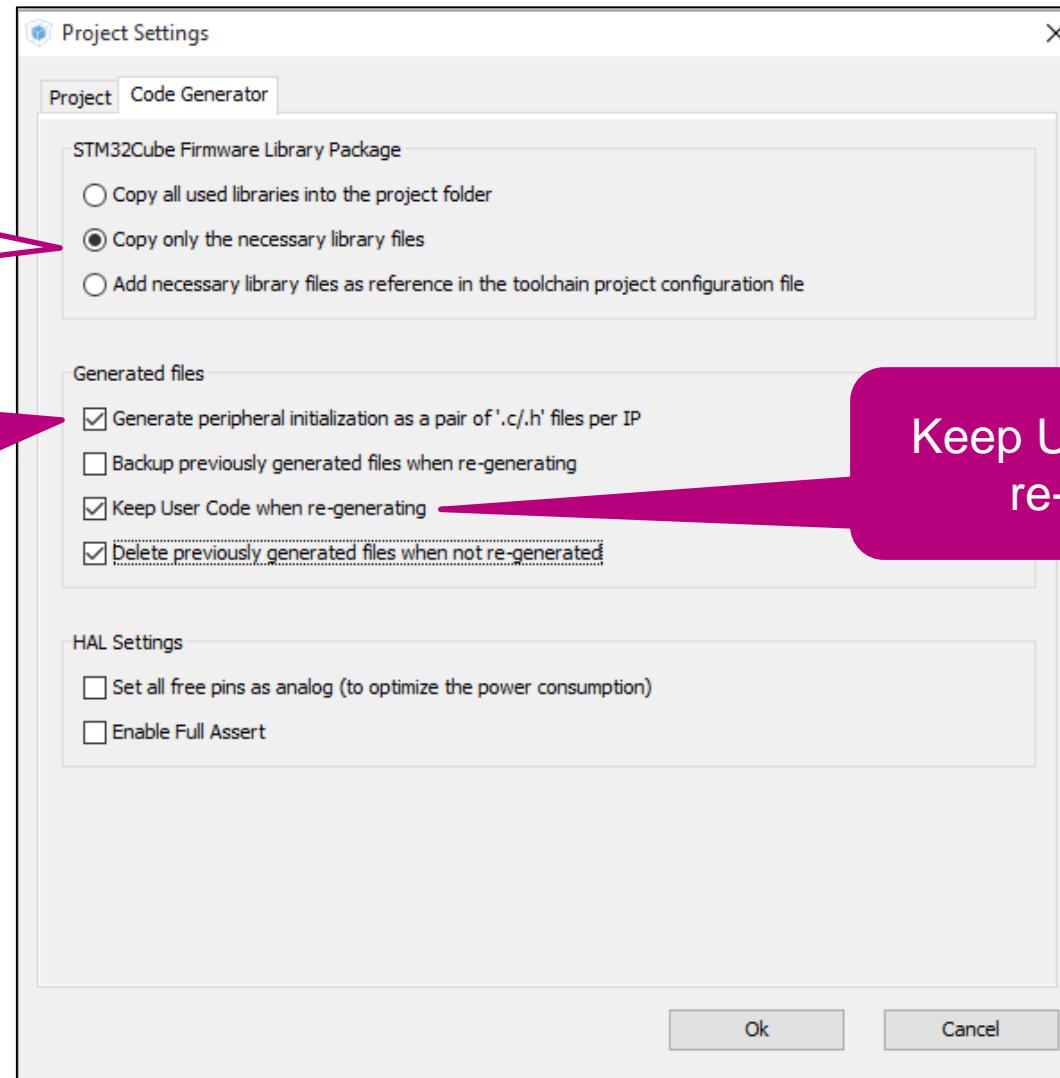
# 1 USB\_DEVICE MiddleWare configuration



# 1 USB\_DEVICE MiddleWare configuration







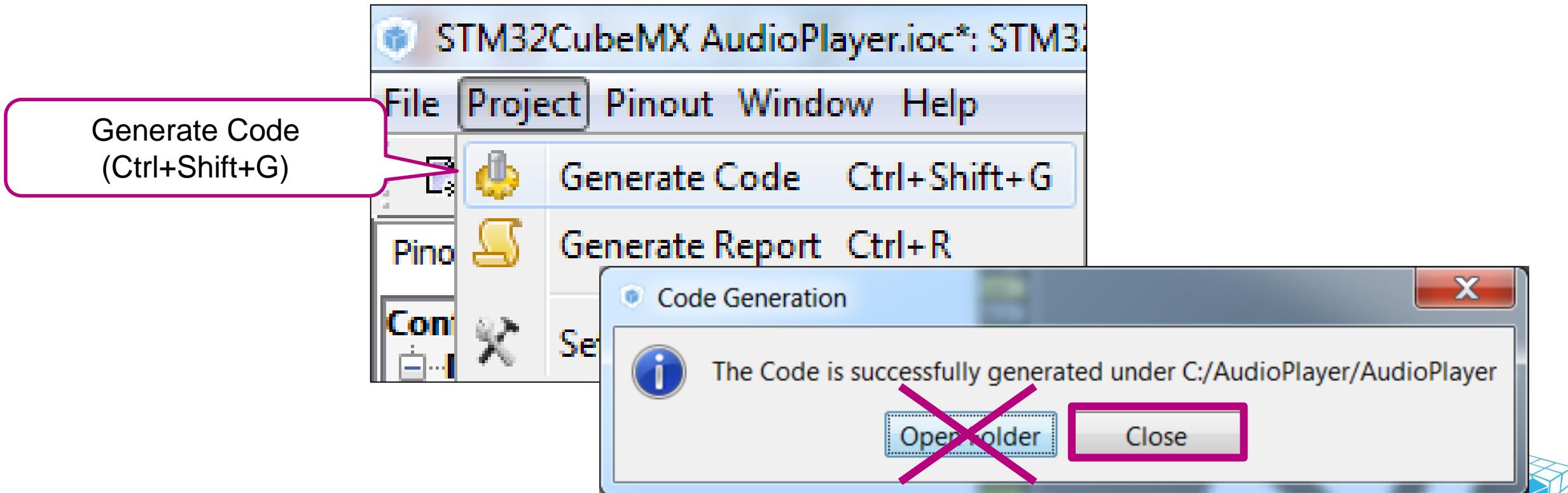
Copy only the necessary library files

Generate peripheral initialization as a pair of .c/.h files per IP

Keep User Code when re-generating

# Save and generate the project

Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP1\_USB\_MSC\_Device**  
to project folder

(C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Inc\

quadspi.h

n25q128a.h

.\Src\

usbd\_storage\_if.c

quadspi.c

This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

- quadspi.c

QSPI\_Init(void)

QSPI\_DelInit(void)

Initialization

QSPI\_Read(...)

QSPI\_Write(...)

Read / Write access

QSPI\_Erase\_Block (...)

QSPI\_Erase\_Sector(...)

QSPI\_Erase\_Chip(void)

Erase operations

QSPI\_GetStatus(void)

QSPI\_GetInfo(...)

QUADSPI memory info / status

Based on BSP driver in STM32CubeL4 HAL

- usbd\_storage\_if.c

STORAGE\_Init\_FS (...)

STORAGE\_Read\_FS (...)

STORAGE\_Write\_FS (...)

STORAGE\_IsReady\_FS (...)

STORAGE\_GetCapacity\_FS (...)

Most important and implemented  
Support for 4kB sector size only  
(due to 4kB block erase feature of used memory)

STORAGE\_GetMaxLun\_FS (void)

STORAGE\_IsWriteProtected\_FS (...)

Not so important

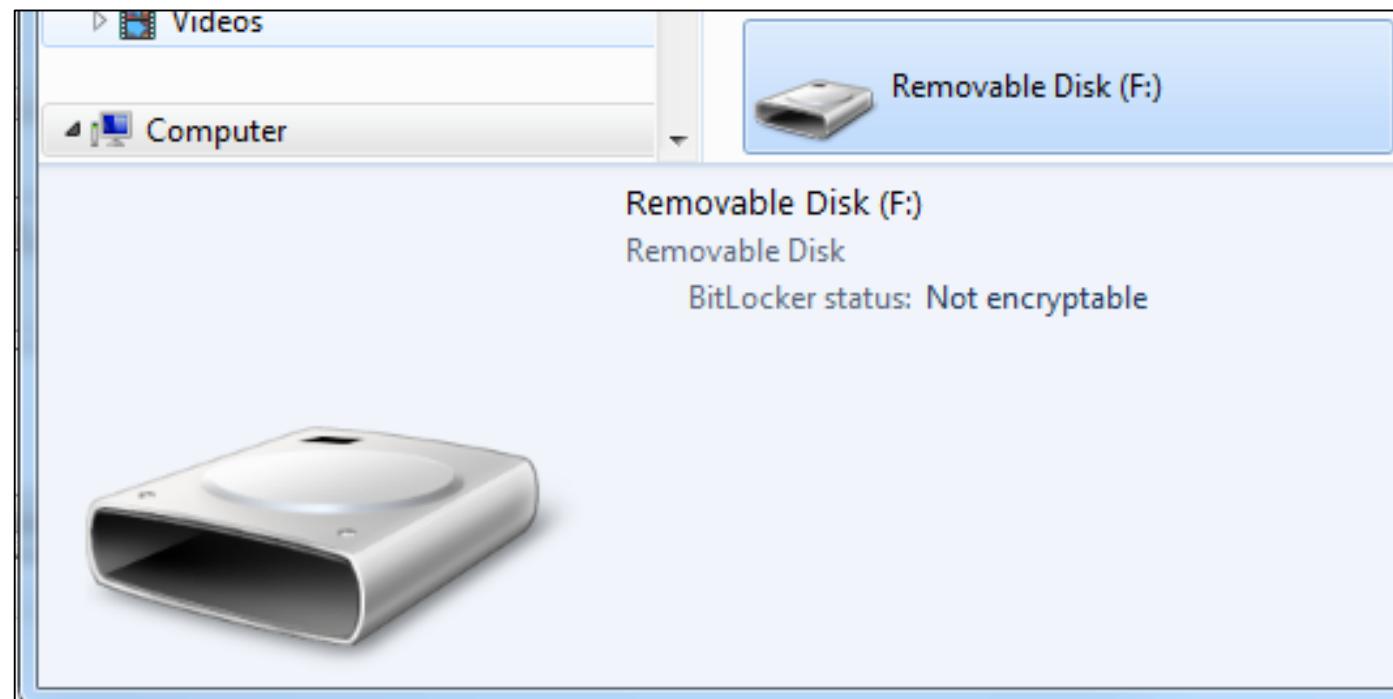
Templates generated by STM32CubeMX, just the bodies  
of the functions were implemented.

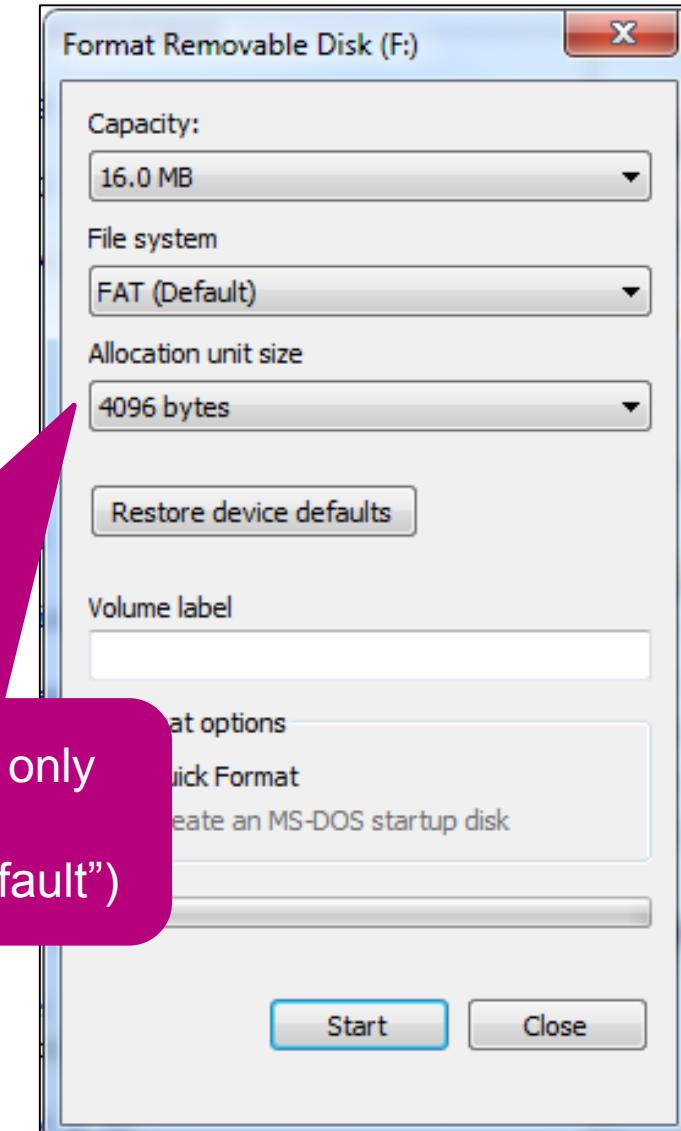
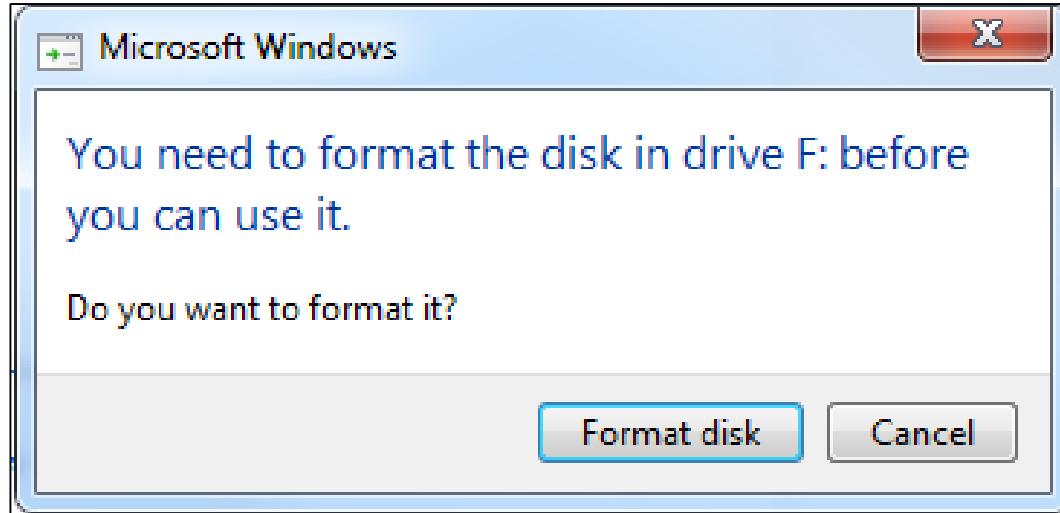
1. Open the project
2. Connect discovery kit over miniUSB cable (ST-Link)
3. Compile and download
4. Press the reset button on discovery once download has finished

# 1 Connect the new “FLASH stick”

56

- Plug-In microUSB cable
- Check if new Mass Storage Class device appeared





Actual tiny implementation supports only  
native 4kB sector size  
(Windows 10 users should select "default")

- Now try to copy the **audio.wav** file from the package to the new

RED LED  
(LD4)



GREEN LED  
(LD5)



ERASING

RED LED  
(LD4)



GREEN LED  
(LD5)

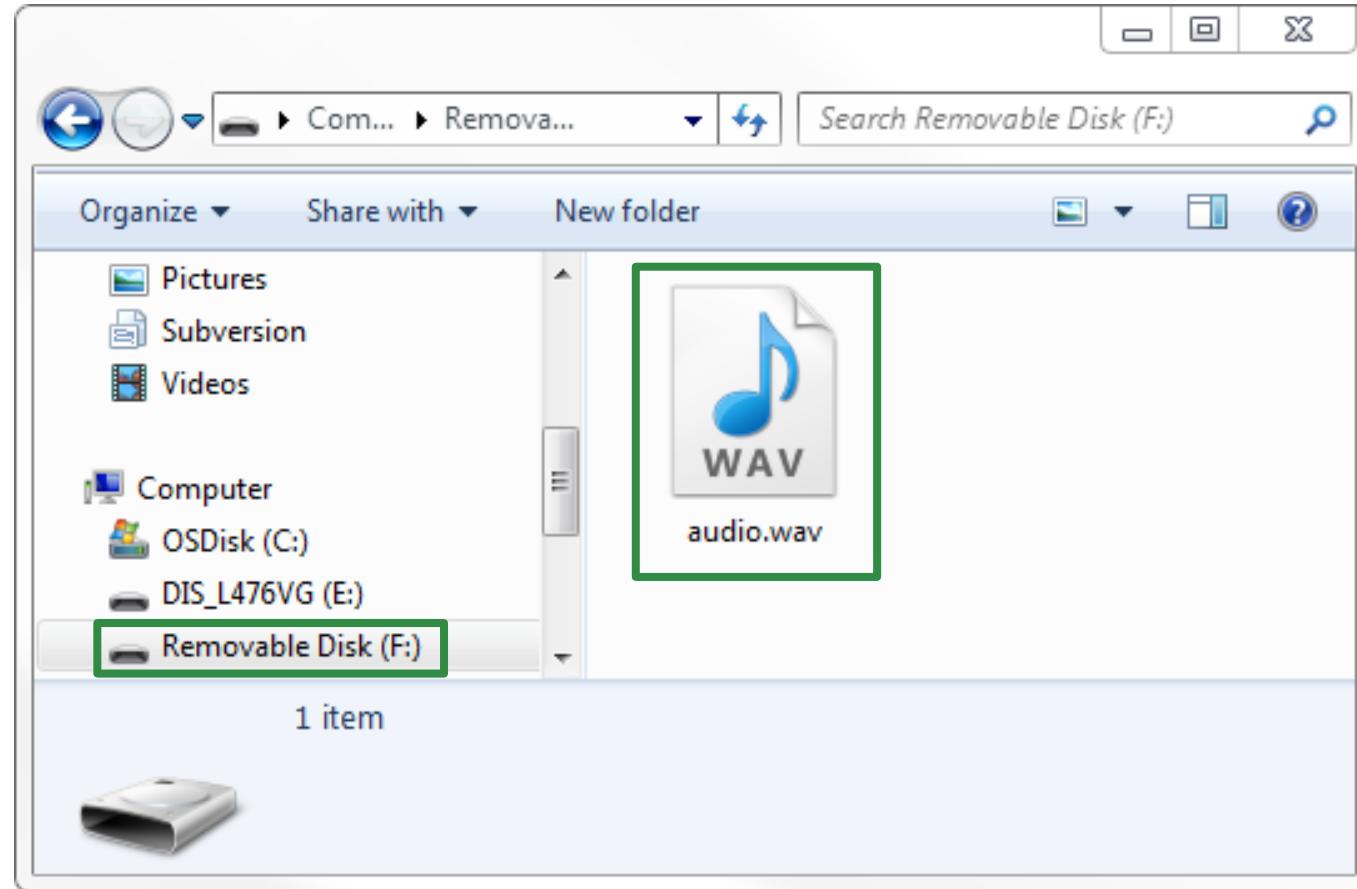


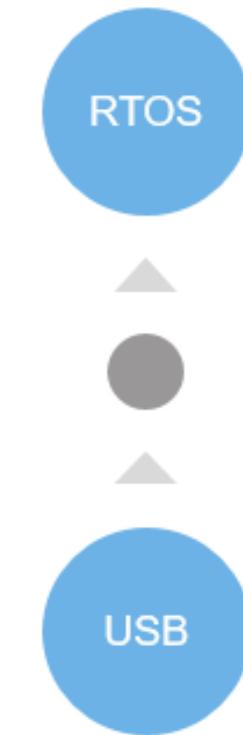
WRITING

# 1

# Checkpoint #3

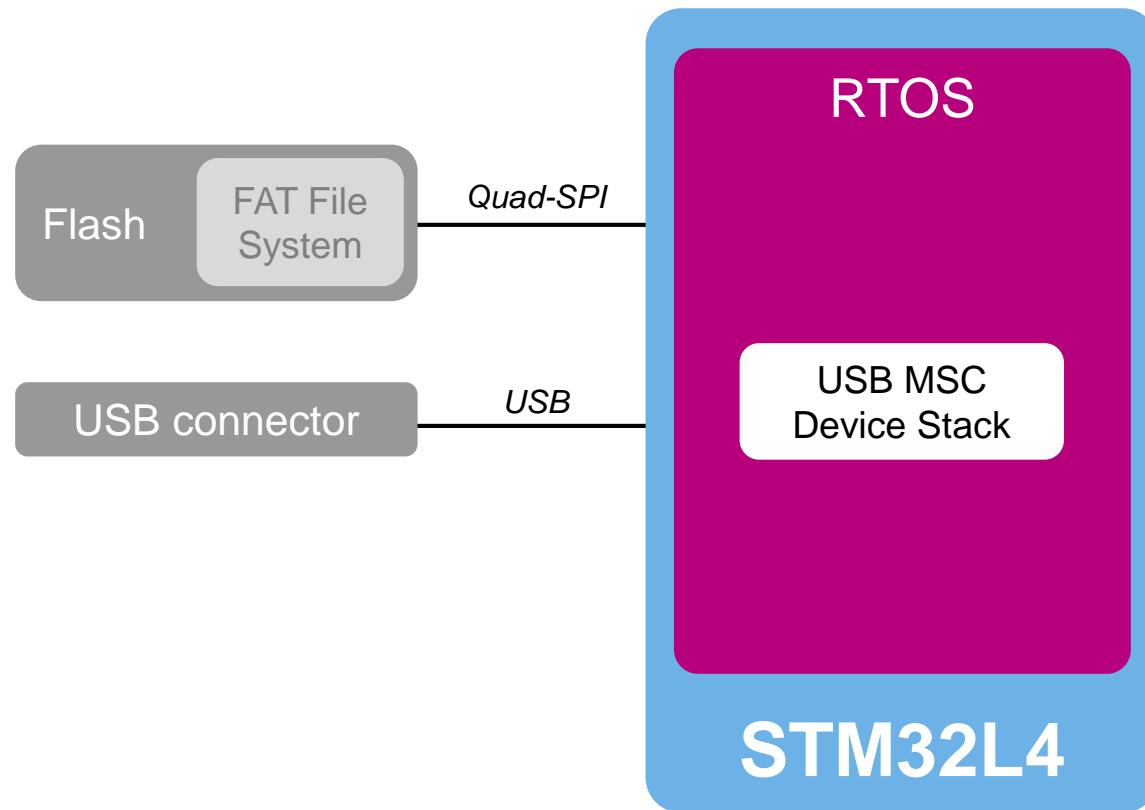
59





## STEP2 – RTOS

- We want to have the application running on RTOS



- To have the application modular (design not closed yet)
  - add/remove functionality easily
- We want to implement application with multiple functionality quickly
  - we don't want to care about the operations synchronization and flow management
- Because we can
  - far more than enough flash, RAM, performance
  - don't need to limit ourselves

## Conventional app

```
void main(void)
{
    while(1)
    {
        switch task {
        case 1:
            ....
            break;
        case 2:
            ....
            break;
        case 3:
            ....
            break;
        }
    }
}
```

## RTOS app

## Task 1 (Thread)

```
void main1(void)
{
    while(1)
    {
        ....
    }
}
```

Stack 1

## Task 2 (Thread)

```
void main2(void)
{
    while(1)
    {
        ....
    }
}
```

Stack 2

## Task 3 (Thread)

```
void main3(void)
{
    while(1)
    {
        ....
    }
}
```

Stack 3

Scheduler (RTOS kernel)  
switching based on event, time, priority...

## CPU time and scheduling

- A timeout parameter is incorporated in RTOS functions to avoid system lockup
- When a timeout is specified the system waits until a resource is available or event occurs
- While waiting, other threads are scheduled
- [osDelay](#) function puts a thread into the state WAITING for a specified period of time

## Threads and ISR communication

## • Signal

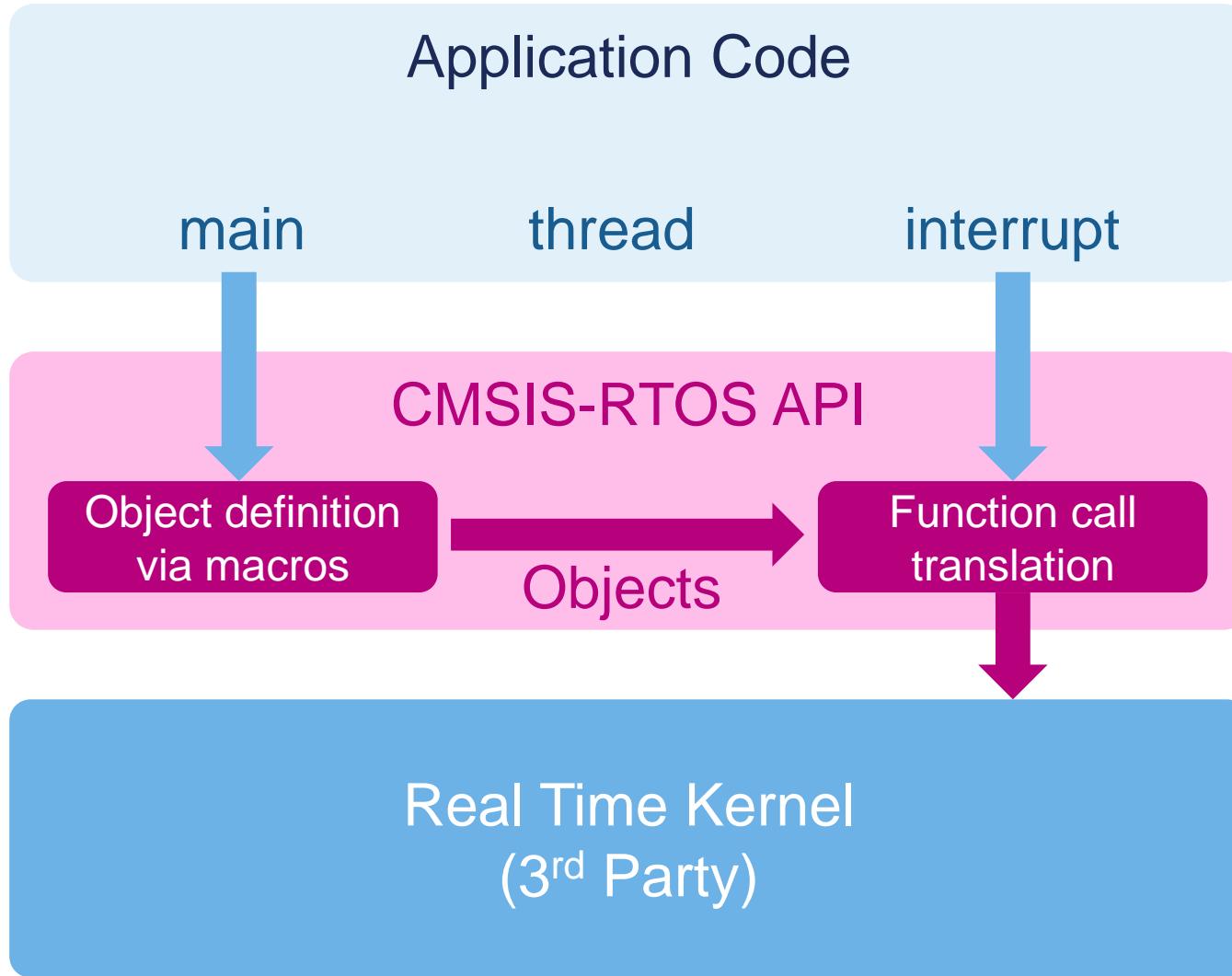
- flag that may be used to signal specific condition to a thread
- can be modified in an ISR or set from other threads.

## • Message

- 32-bit value that can be sent to a thread or an ISR
- buffered in a queue, type and queue size defined in a descriptor

## • Mail

- fixed-size memory block that can be sent to a thread or an ISR
- buffered in a queue and memory allocation is provided
- type and queue size are defined in a descriptor.



FreeRTOS



- UM1722 – Developing Applications on STM32Cube with RTOS

[http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user\\_manual/DM00105262.pdf](http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/DM00105262.pdf)

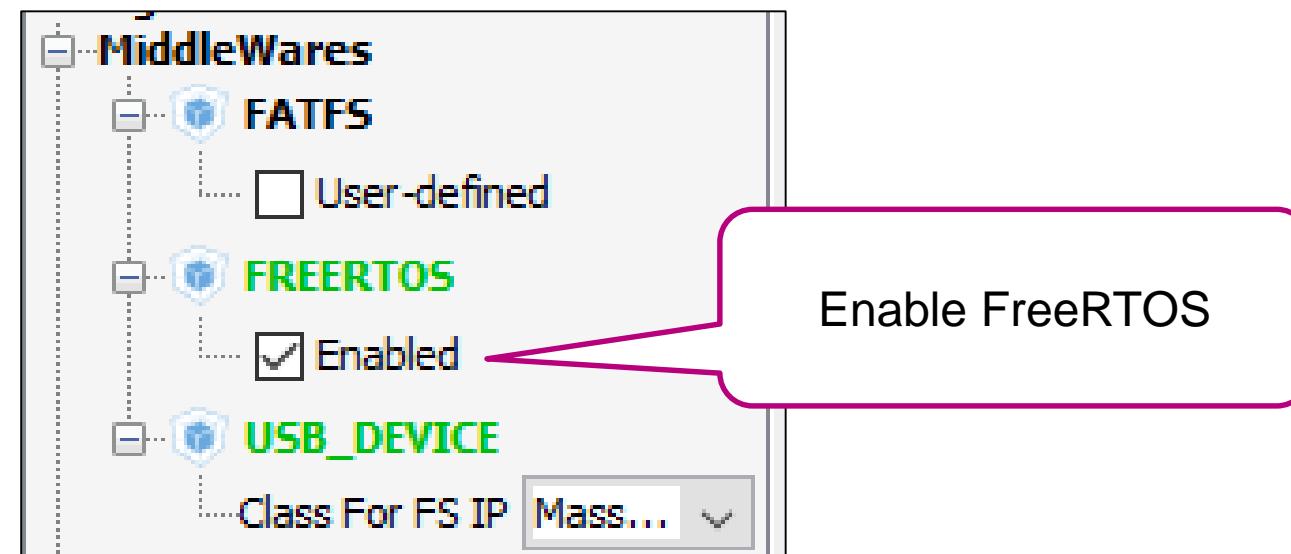
- <http://www.freertos.org>

- Using The FreeRTOS Real Time Kernel - Cortex-M3 edition

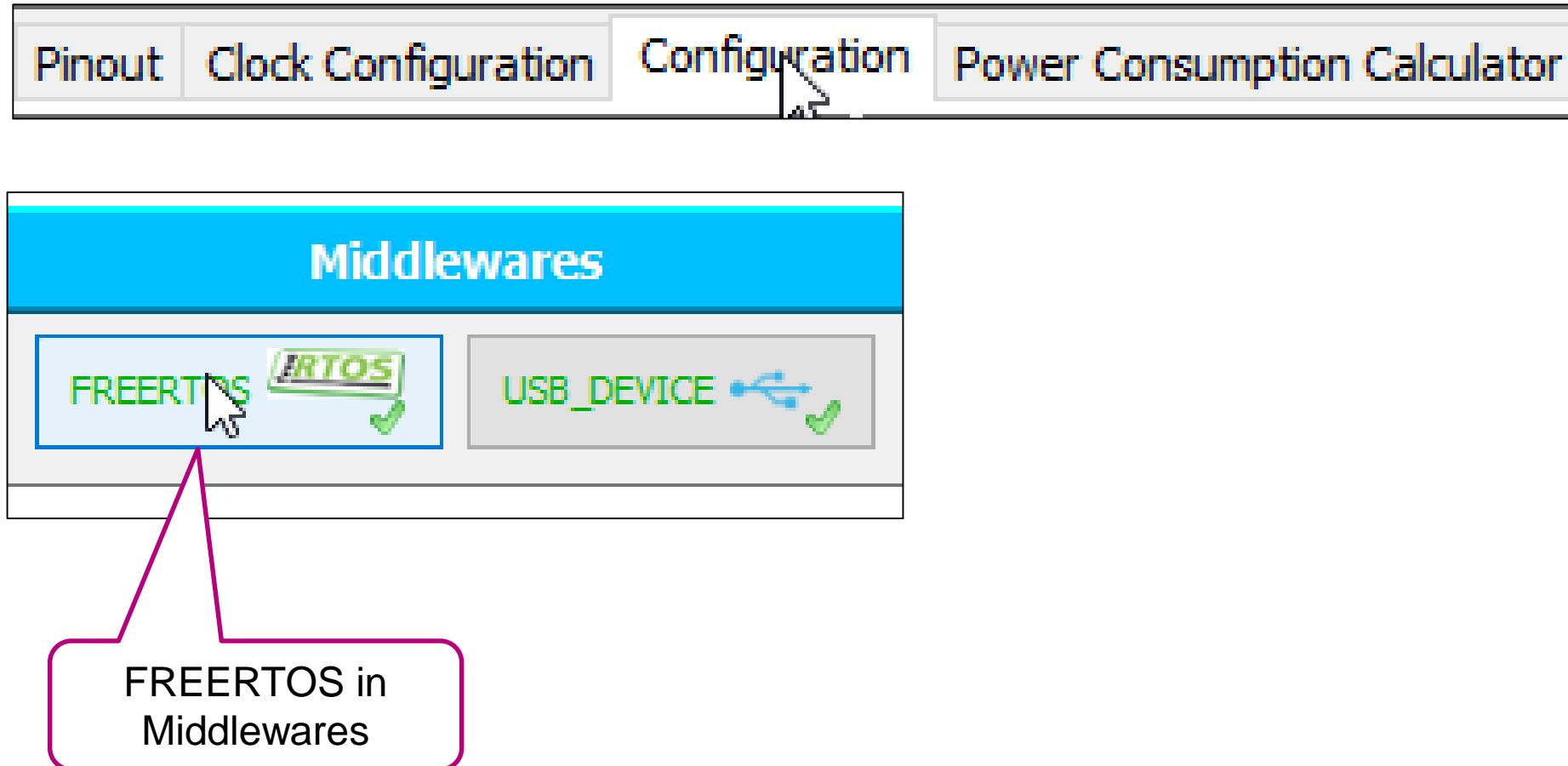
[http://shop.freertos.org/FreeRTOS\\_Tutorial\\_Book\\_Generic\\_Cortex\\_M3\\_Edition\\_p/pdf\\_cortex-m3\\_tutorial\\_book.htm](http://shop.freertos.org/FreeRTOS_Tutorial_Book_Generic_Cortex_M3_Edition_p/pdf_cortex-m3_tutorial_book.htm)



Go to Configuration



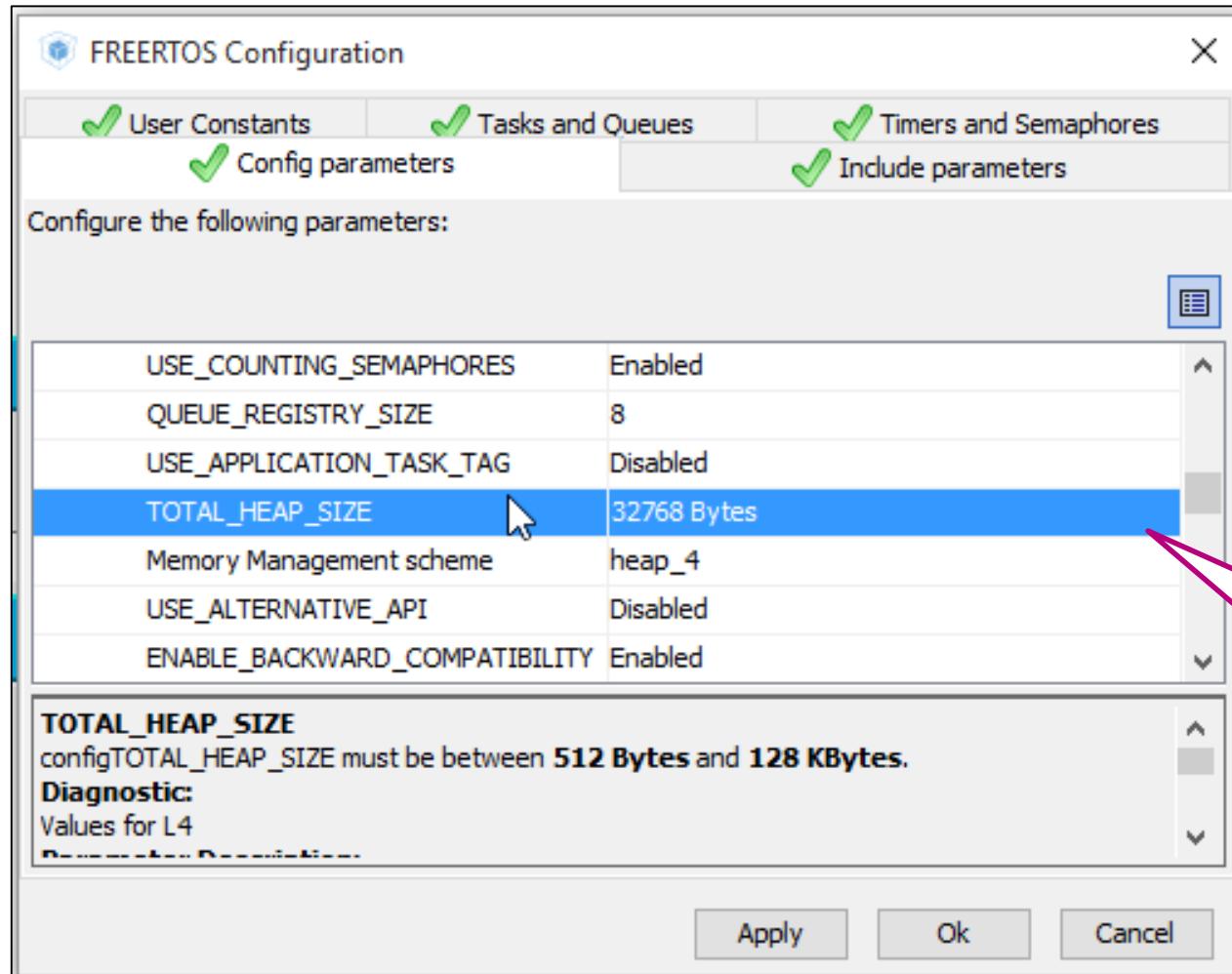
Enable FreeRTOS



The image shows the STM32CubeMX software interface. At the top, there is a navigation bar with tabs: Pinout, Clock Configuration, Configuration (which is currently selected), and Power Consumption Calculator. Below the navigation bar, there is a section titled "Middlewares" containing icons for "FREERTOS" and "USB\_DEVICE". A large green arrow points from the "FREERTOS" icon towards a detailed configuration window on the right. This window is titled "FREERTOS Configuration" and contains tabs for Tasks and Queues, Timers and Semaphores, Config parameters (which is selected), Include parameters, and User Constants. The "Config parameters" tab displays various kernel settings with their current values:

Setting	Value
CMSIS-RTOS version	1.02
FreeRTOS version	8.2.1
Kernel settings	
USE_PREEMPTION	Enabled
CPU_CLOCK_HZ	SystemCoreClock
TICK_RATE_HZ	1000
MAX_PRIORITIES	7
MINIMAL_STACK_SIZE	128 Words
MAX_TASK_NAME_LEN	16
USE_16_BIT_TICKS	Disabled
IDLE_SHOULD_YIELD	Enabled
USE_MUTEXES	Enabled
USE_RECURSIVE_MUTEXES	Enabled
USE_COUNTING_SEMAPHORES	Enabled
QUEUE_REGISTRY_SIZE	8
USE_APPLICATION_TASK_TAG	Disabled
TOTAL_HEAP_SIZE	3000 Bytes

At the bottom of the configuration window are buttons for Apply, Ok, and Cancel.



2x

FREERTOS Configuration

Config parameters

User Constants

Tasks and Queues

Tasks

Name	Task Priority	Stack size (Words)	Entry function
defaultTask	osPriorityNormal	128	StartDefaultTask

Queues

Name

Add Delete

Item size

Add Delete

Apply Ok Cancel

Go to Tasks and Queues configuration

appTask

Modify the default task

Edit Task

Name

osPriorityNormal

Stack size (Words)

4096

Entry function

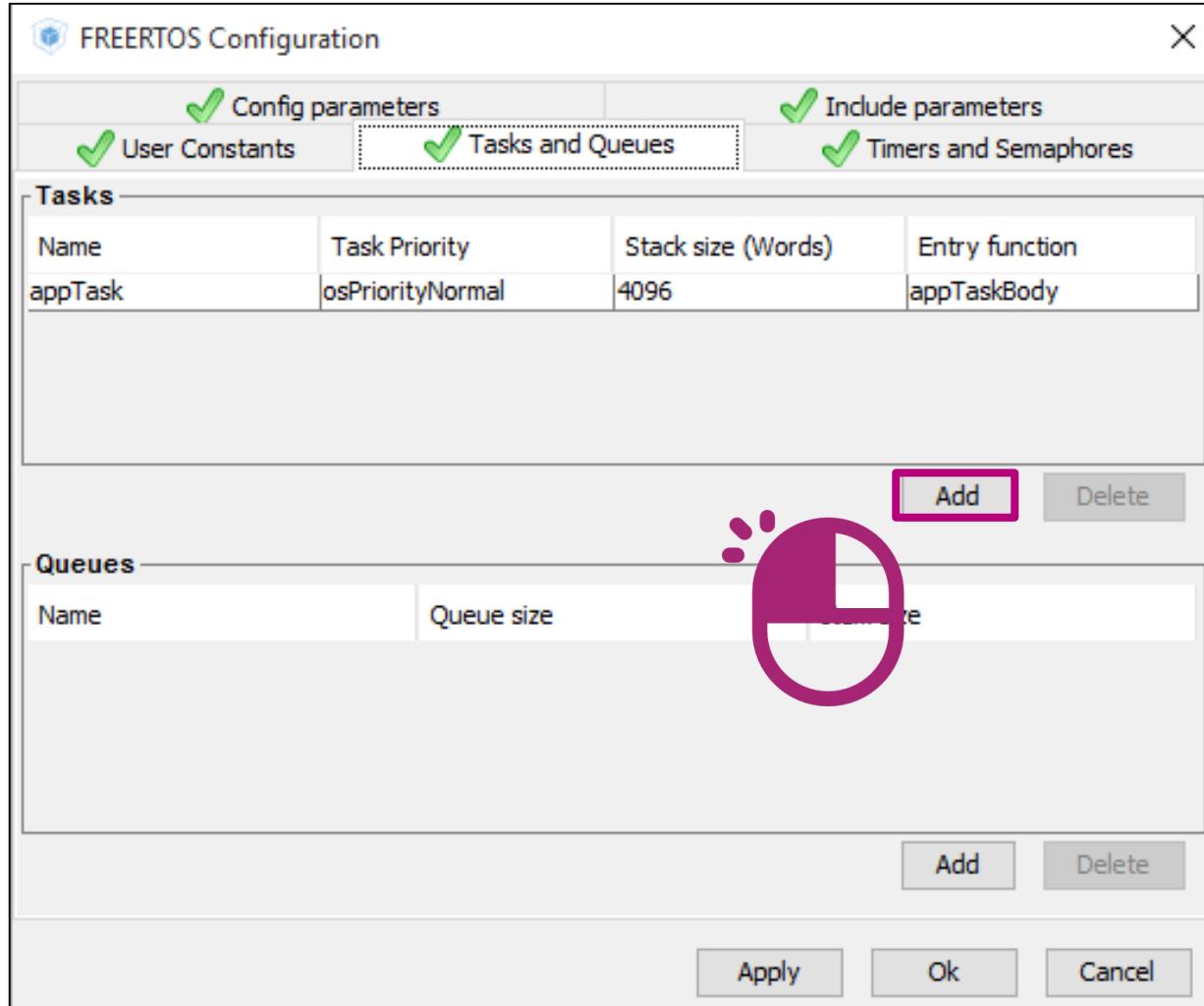
appTaskBody

OK Cancel

appTask

4096

appTaskBody



usbTask

usbTask

New Task

Name

osPriorityHigh

usbTask

osPriorityHigh

256

Stack size (Words)

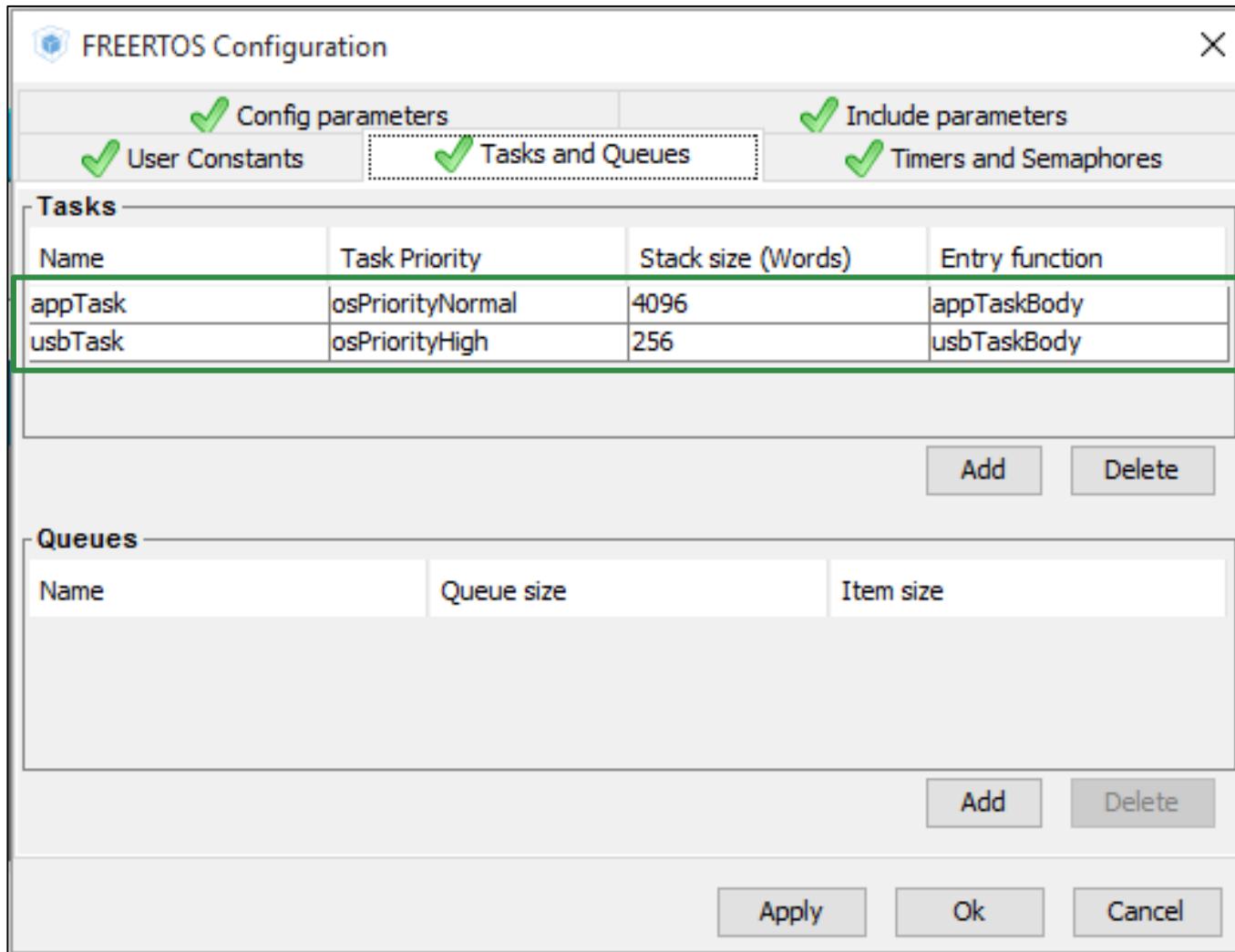
256

Entry function

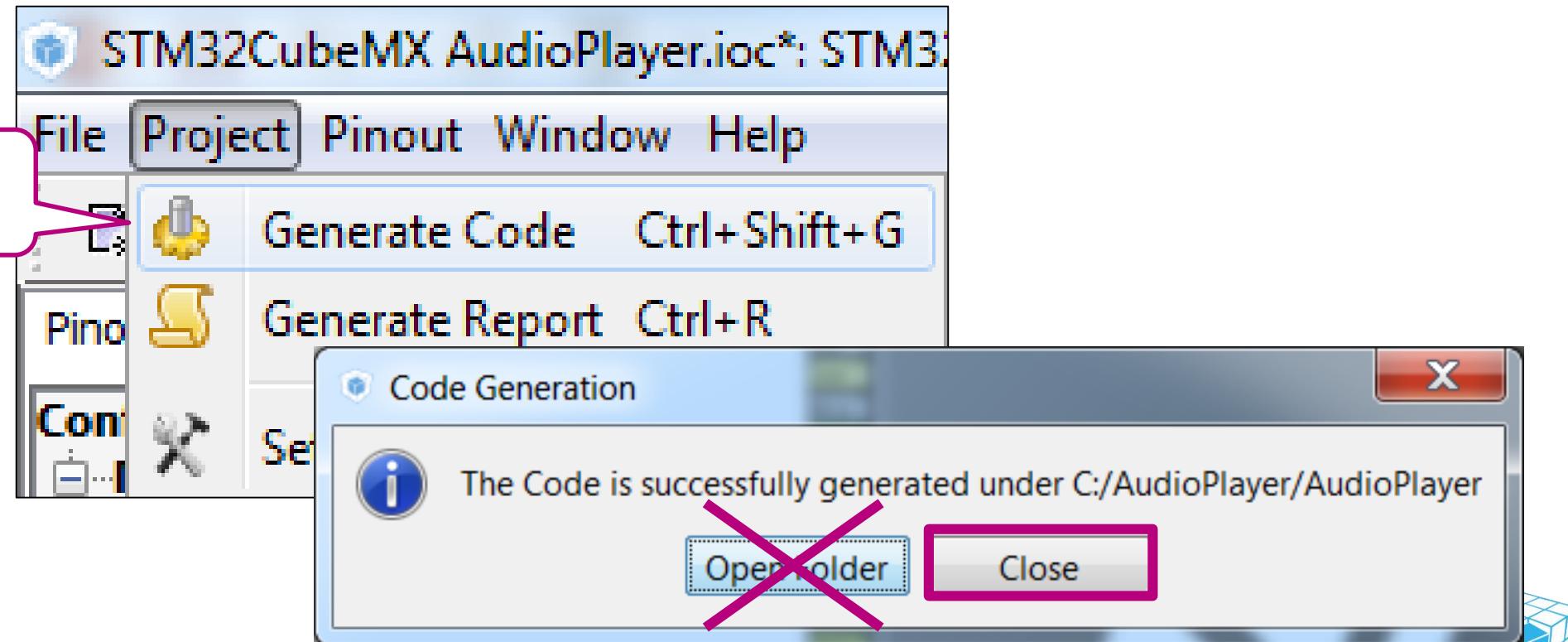
usbTaskBody

256

usbTaskBody



Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP2\_RTO**

to root folder of your project

(C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Src\

freertos.c

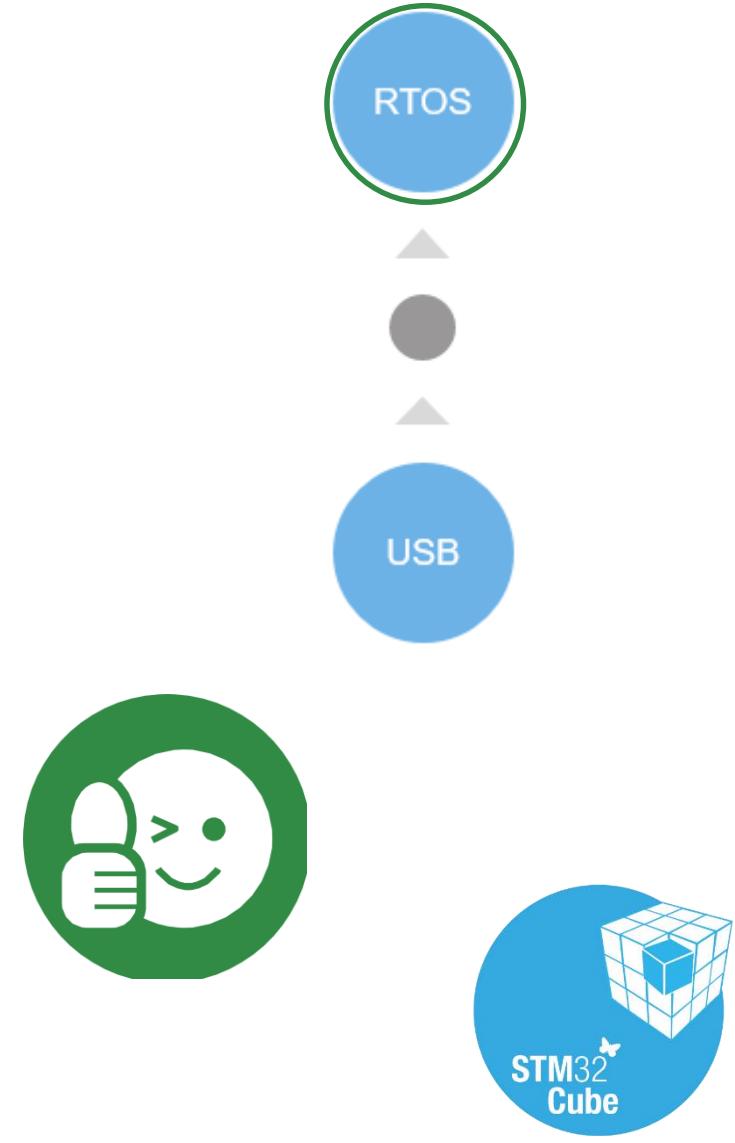
usbd\_storage\_if.c

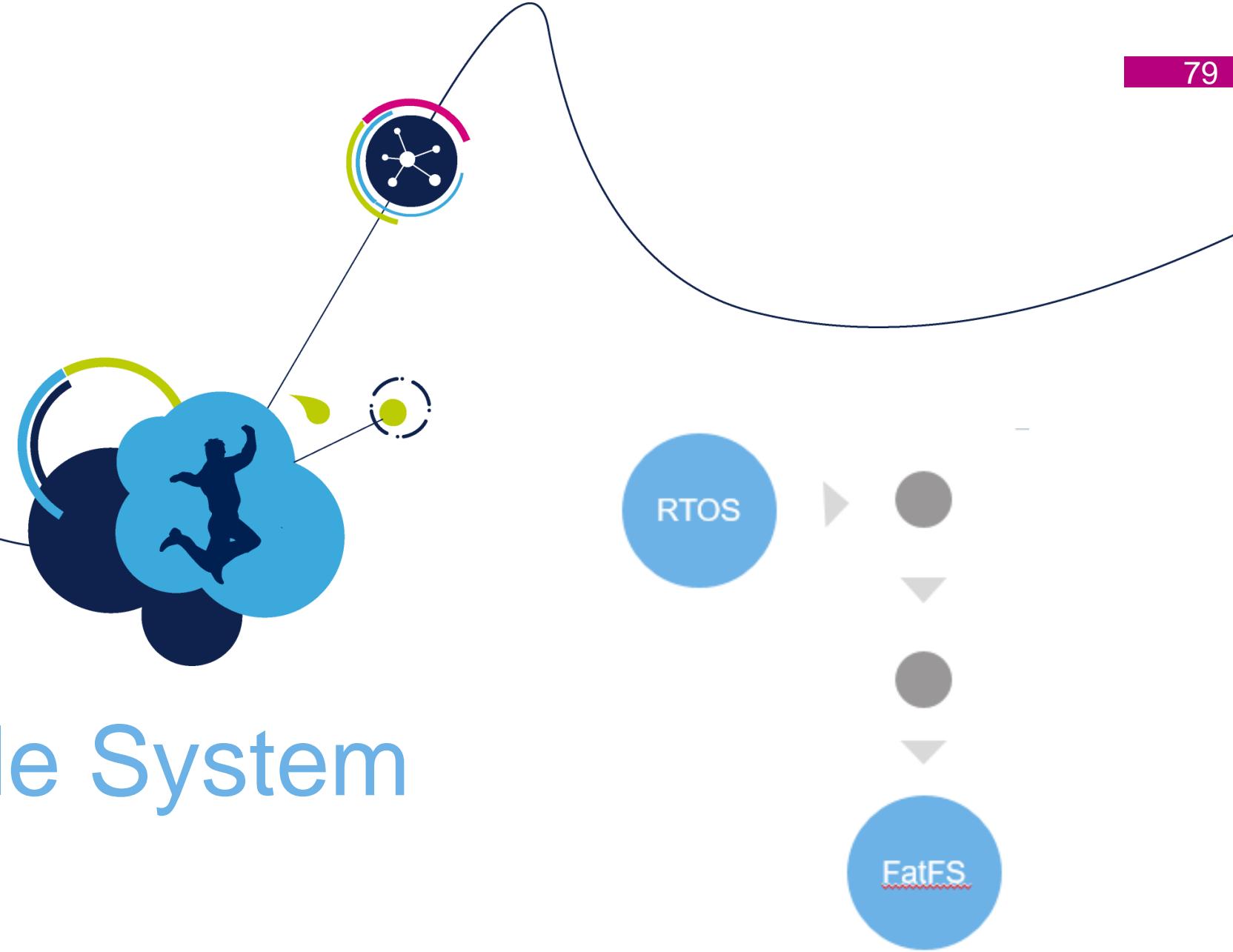
This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

- usbd\_storage\_if.c – change in LEDs use, we need them for other purpose

1. Open the project
2. Compile and download
3. Press the reset button on discovery once download has finished

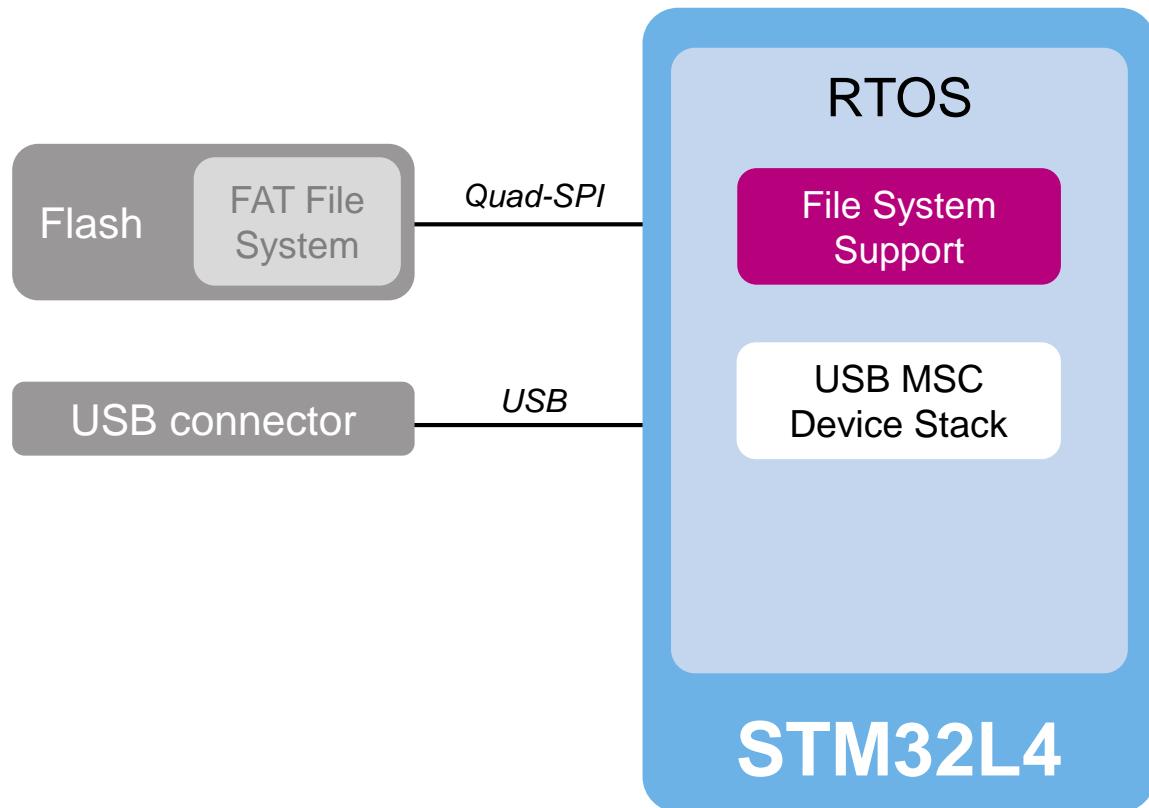
- appTask – GREEN LED blinking
- usbTask – USB connected → RED LED ON
- usbTask – USB connected → RED LED OFF





## STEP3 – File System

- We want to have the access to our “audio.wav” file from RTOS



## PC application in C language

## C standard libraries (stdio.h)

- `fopen(...)` – opens a file
- `fclose(...)` – closes a file
- `fread(...)` – reads from a file
- `fwrite(...)` – writes to a file
- `fprintf(...)` – formatted output to a file
- `fscanf(...)` – formatted input from a file

## MCU application in C language

## C standard libraries (stdio.h)

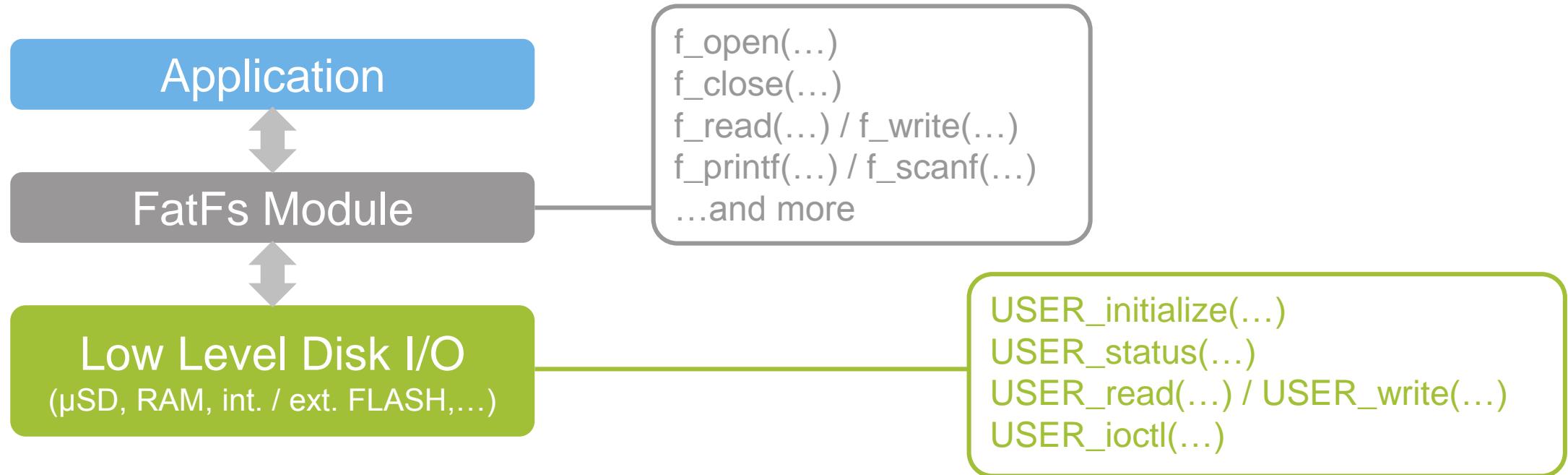
- `fopen(...)` – opens a file
- `fclose(...)` – closes a file
- `fread(...)` – reads from a file
- `fwrite(...)` – writes to a file
- `fprintf(...)` – formatted output to a file
- `fscanf(...)` – formatted input from a file

No OS + No standard File System =

No native support by  
C standard IO library

Let's add “special” library to support that

- Windows compatible FAT file system support



UM1721 – Developing Applications on STM32Cube with FatFs

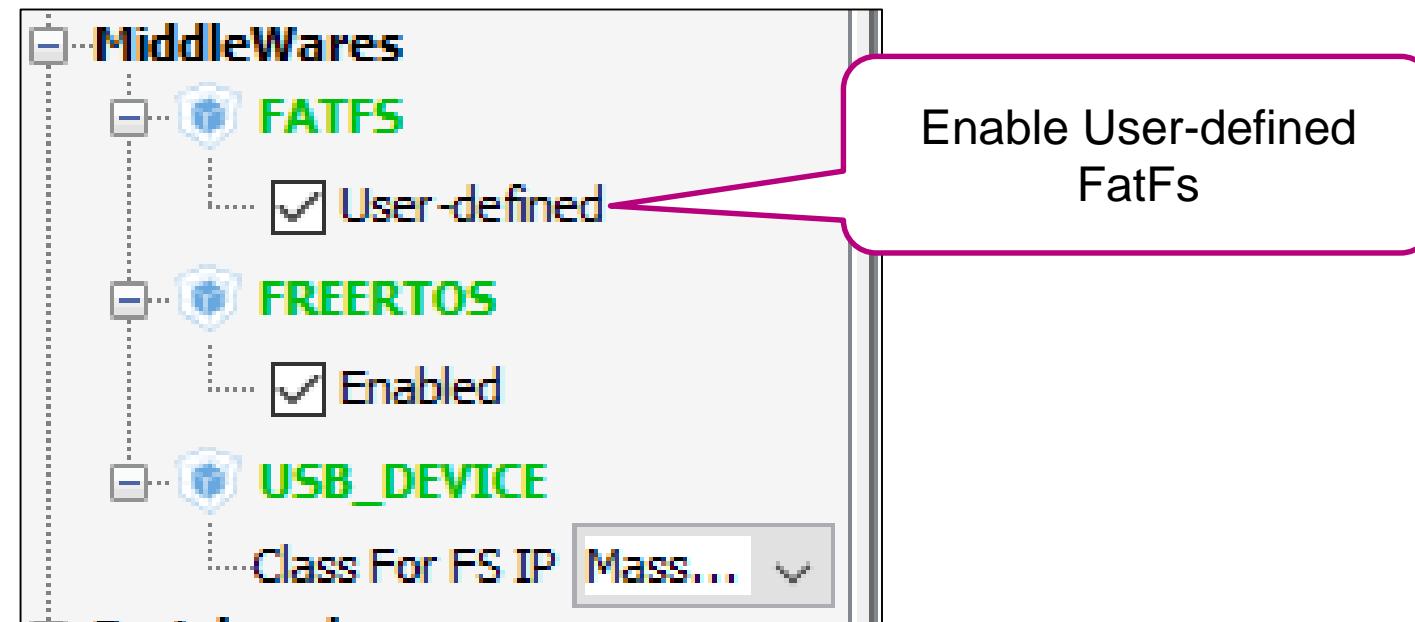
[http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user\\_manual/DM00105259.pdf](http://www.st.com/st-web-ui/static/active/en/resource/technical/document/user_manual/DM00105259.pdf)

FatFs project home page

[http://elm-chan.org/fsw/ff/00index\\_e.html](http://elm-chan.org/fsw/ff/00index_e.html)



Go to Configuration



Enable User-defined  
FatFs



FATFS in  
Middlewares

**FATFS Configuration**

Configure the below parameters :

<input checked="" type="checkbox"/> Set Defines	<input checked="" type="checkbox"/> User Constants																		
Configure the below parameters :																			
<b>Version</b> FATFS version R.0.11																			
<b>Function Parameters</b> <table border="1"> <tbody> <tr> <td>FS_TINY (Tiny mode)</td> <td>Enabled</td> </tr> <tr> <td>FS_READONLY (Read-only mode)</td> <td>Disabled</td> </tr> <tr> <td>FS_MINIMIZE (Minimization level)</td> <td>Disabled</td> </tr> <tr> <td>USE_STRFUNC (String functions)</td> <td>Enabled with LF -&gt; CRLF conversion</td> </tr> <tr> <td>USE_FIND (Find functions)</td> <td>Enabled</td> </tr> <tr> <td>USE_MKFS (Make filesystem function)</td> <td>Enabled</td> </tr> <tr> <td>USE_FORWARD (Forward function)</td> <td>Disabled</td> </tr> <tr> <td>USE_LABEL (Volume label functions)</td> <td>Enabled</td> </tr> <tr> <td>USE_FASTSEEK (Fast seek function)</td> <td>Enabled</td> </tr> </tbody> </table>		FS_TINY (Tiny mode)	Enabled	FS_READONLY (Read-only mode)	Disabled	FS_MINIMIZE (Minimization level)	Disabled	USE_STRFUNC (String functions)	Enabled with LF -> CRLF conversion	USE_FIND (Find functions)	Enabled	USE_MKFS (Make filesystem function)	Enabled	USE_FORWARD (Forward function)	Disabled	USE_LABEL (Volume label functions)	Enabled	USE_FASTSEEK (Fast seek function)	Enabled
FS_TINY (Tiny mode)	Enabled																		
FS_READONLY (Read-only mode)	Disabled																		
FS_MINIMIZE (Minimization level)	Disabled																		
USE_STRFUNC (String functions)	Enabled with LF -> CRLF conversion																		
USE_FIND (Find functions)	Enabled																		
USE_MKFS (Make filesystem function)	Enabled																		
USE_FORWARD (Forward function)	Disabled																		
USE_LABEL (Volume label functions)	Enabled																		
USE_FASTSEEK (Fast seek function)	Enabled																		
<b>Locale and Namespace Parameters</b> <table border="1"> <tbody> <tr> <td>CODE_PAGE (Code page on target)</td> <td>Latin 1 (Windows)</td> </tr> <tr> <td>USE_LFN (Use Long Filename)</td> <td>Disabled</td> </tr> <tr> <td>MAX_LFN (Max Long Filename)</td> <td>255</td> </tr> <tr> <td>LFN_UNICODE (Enable Unicode)</td> <td>ANSI/OEM</td> </tr> <tr> <td>STRF_ENCODE (Character encoding)</td> <td>UTF-8</td> </tr> <tr> <td>FS_RPATH (Relative Path)</td> <td>Disabled</td> </tr> </tbody> </table>		CODE_PAGE (Code page on target)	Latin 1 (Windows)	USE_LFN (Use Long Filename)	Disabled	MAX_LFN (Max Long Filename)	255	LFN_UNICODE (Enable Unicode)	ANSI/OEM	STRF_ENCODE (Character encoding)	UTF-8	FS_RPATH (Relative Path)	Disabled						
CODE_PAGE (Code page on target)	Latin 1 (Windows)																		
USE_LFN (Use Long Filename)	Disabled																		
MAX_LFN (Max Long Filename)	255																		
LFN_UNICODE (Enable Unicode)	ANSI/OEM																		
STRF_ENCODE (Character encoding)	UTF-8																		
FS_RPATH (Relative Path)	Disabled																		

<b>Physical Drive Parameters</b>	
VOLUMES (Logical drives)	1
MAX_SS (Maximum Sector Size)	4096
MIN_SS (Minimum Sector Size)	4096
MULTI_PARTITION (Volume partitio...)	Disabled
USE_TRIM (Erase feature)	Disabled
FS_NOFSINFO (Force full FAT scan)	0
<b>System Parameters</b>	
FS_NORTC (Timestamp feature)	Dynamic timestamp
NORTC_YEAR (Year for timestamp)	2015
NORTC_MON (Month for timestamp)	6
NORTC_MDAY (Day for timestamp)	4
WORD_ACCESS (Platform depende...)	Byte access
FS_REENTRANT (Re-Entrancy)	Enabled
FS_TIMEOUT (Timeout ticks)	1000
SYNC_t (O/S sync object)	osSemaphoreId
FS_LOCK (Number of files opened si...)	2

MAX\_SS → 4096  
MIN\_SS → 4096

CODE\_PAGE  
→ Latin 1 (Windows)

USE\_LFN  
→ Disabled

LFN\_UNICODE  
→ ANSI/OEM

- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex

appTask  
FatFs

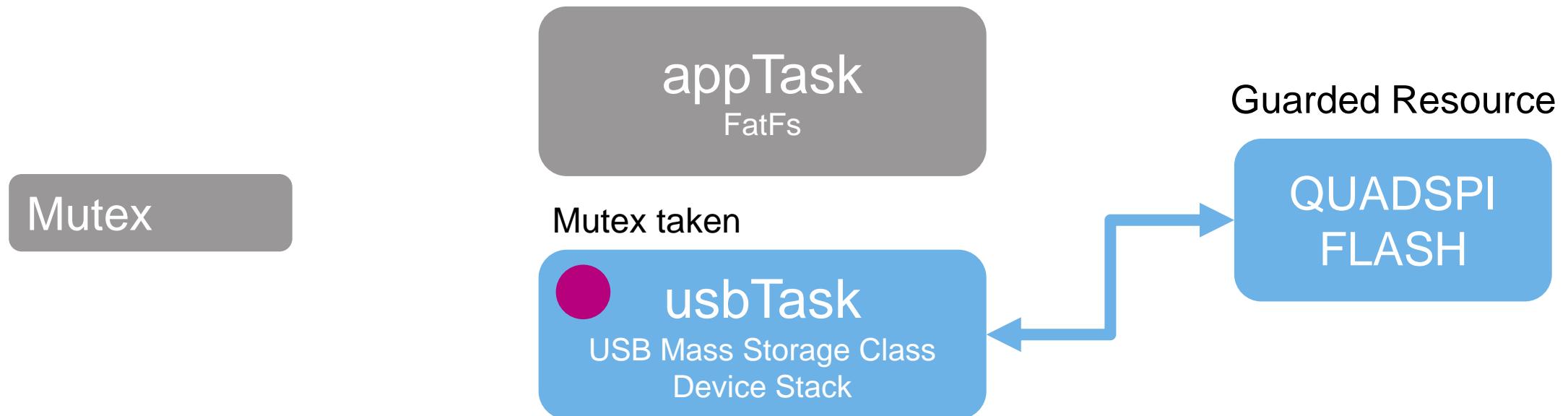
osMutexWait(...)

usbTask  
USB Mass Storage Class  
Device Stack

Guarded Resource

QUADSPI  
FLASH

- Mutex needed as we can't have concurrent access to the QSPI FLASH



- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex

osMutexWait(...)

appTask  
FatFs

Guarded Resource

QUADSPI  
FLASH

usbTask  
USB Mass Storage Class  
Device Stack

- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex

Waiting for Mutex

appTask  
FatFs

osMutexRelease(...)

usbTask  
USB Mass Storage Class  
Device Stack

Guarded Resource

QUADSPI  
FLASH

- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex 

Waiting for Mutex

appTask  
FatFs

Mutex released

usbTask  
USB Mass Storage Class  
Device Stack

Guarded Resource

QUADSPI  
FLASH

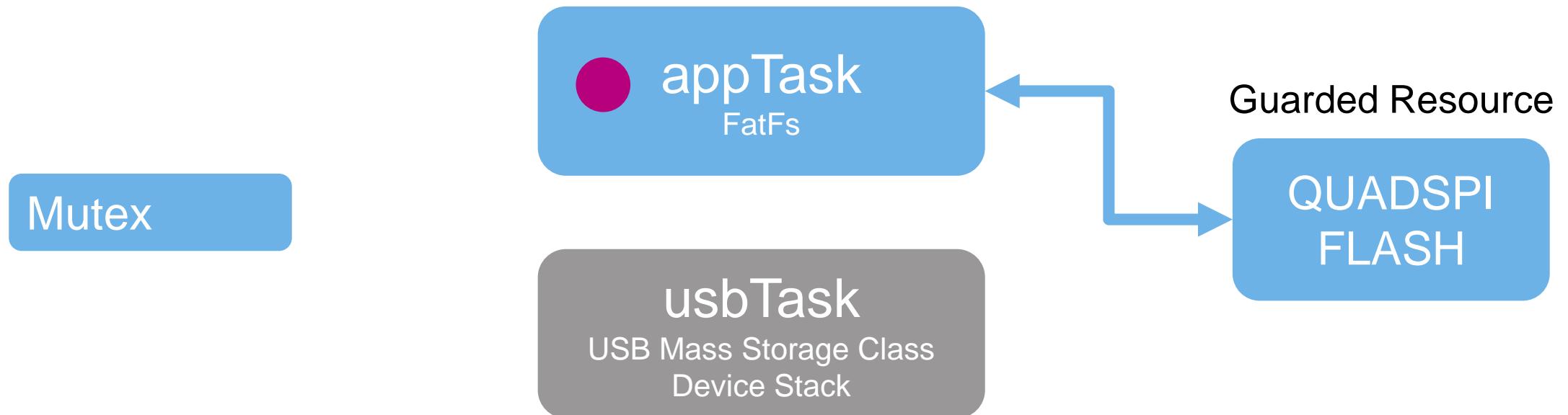
- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex



Guarded Resource

- Mutex needed as we can't have concurrent access to the QSPI FLASH



- Mutex needed as we can't have concurrent access to the QSPI FLASH

Mutex

osMutexRelease(...)

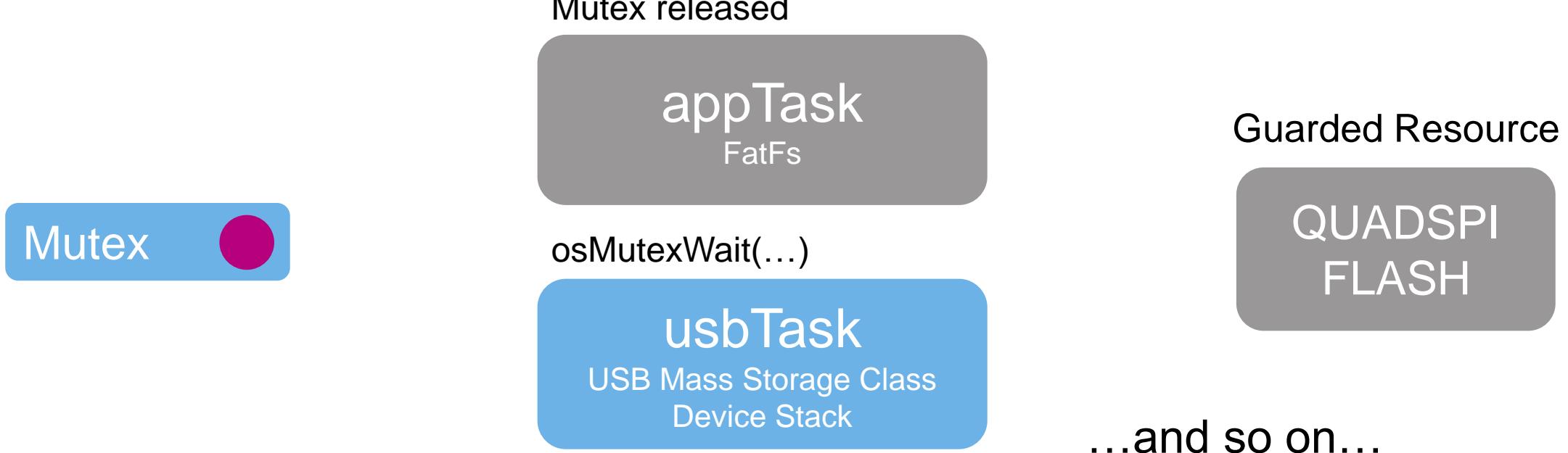
appTask  
FatFs

usbTask  
USB Mass Storage Class  
Device Stack

Guarded Resource

QUADSPI  
FLASH

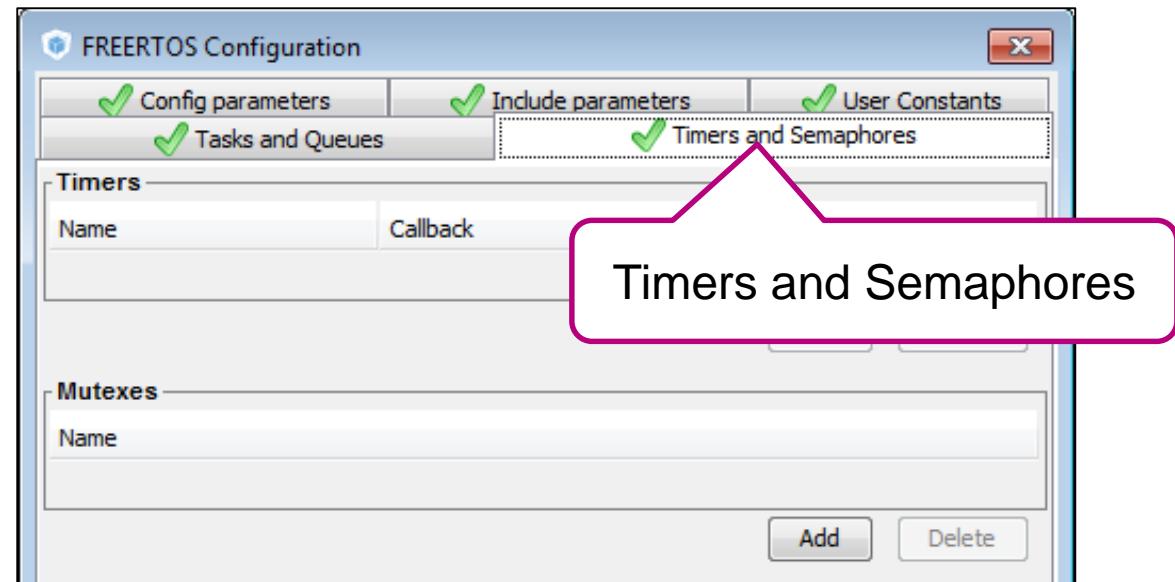
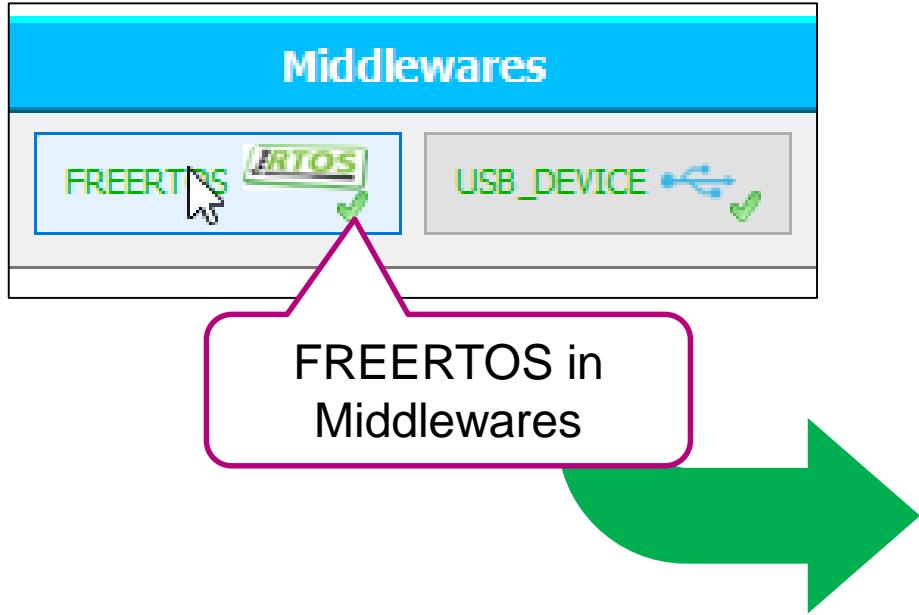
- Mutex needed as we can't have concurrent access to the QSPI FLASH

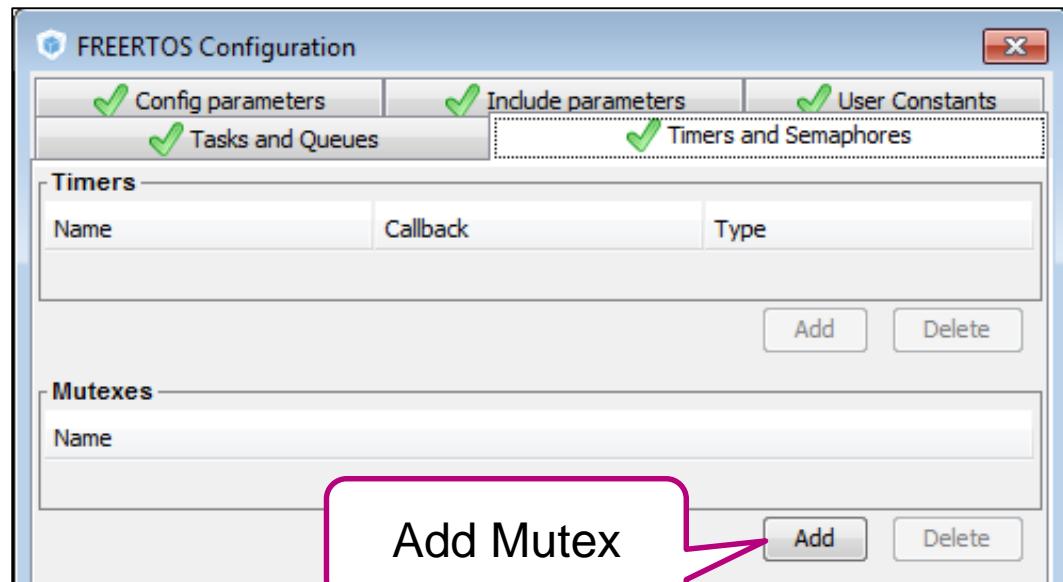
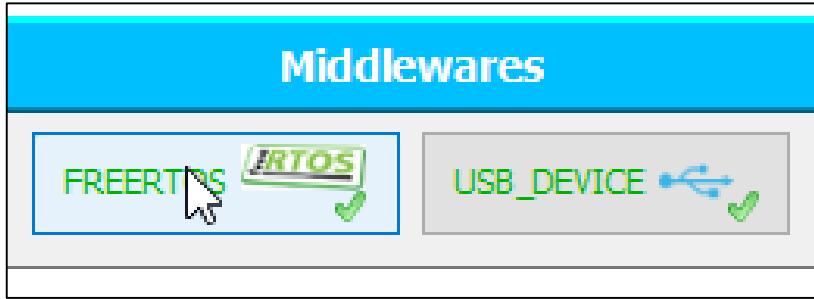


Single memory  
Single QUADSPI interface

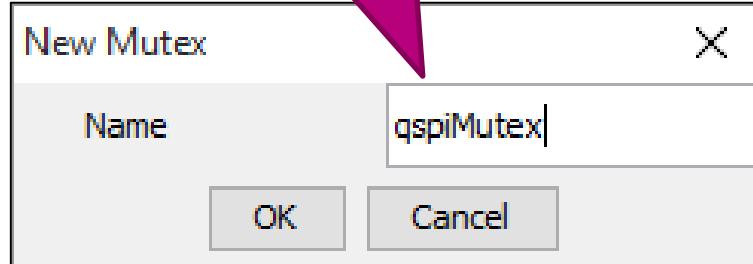


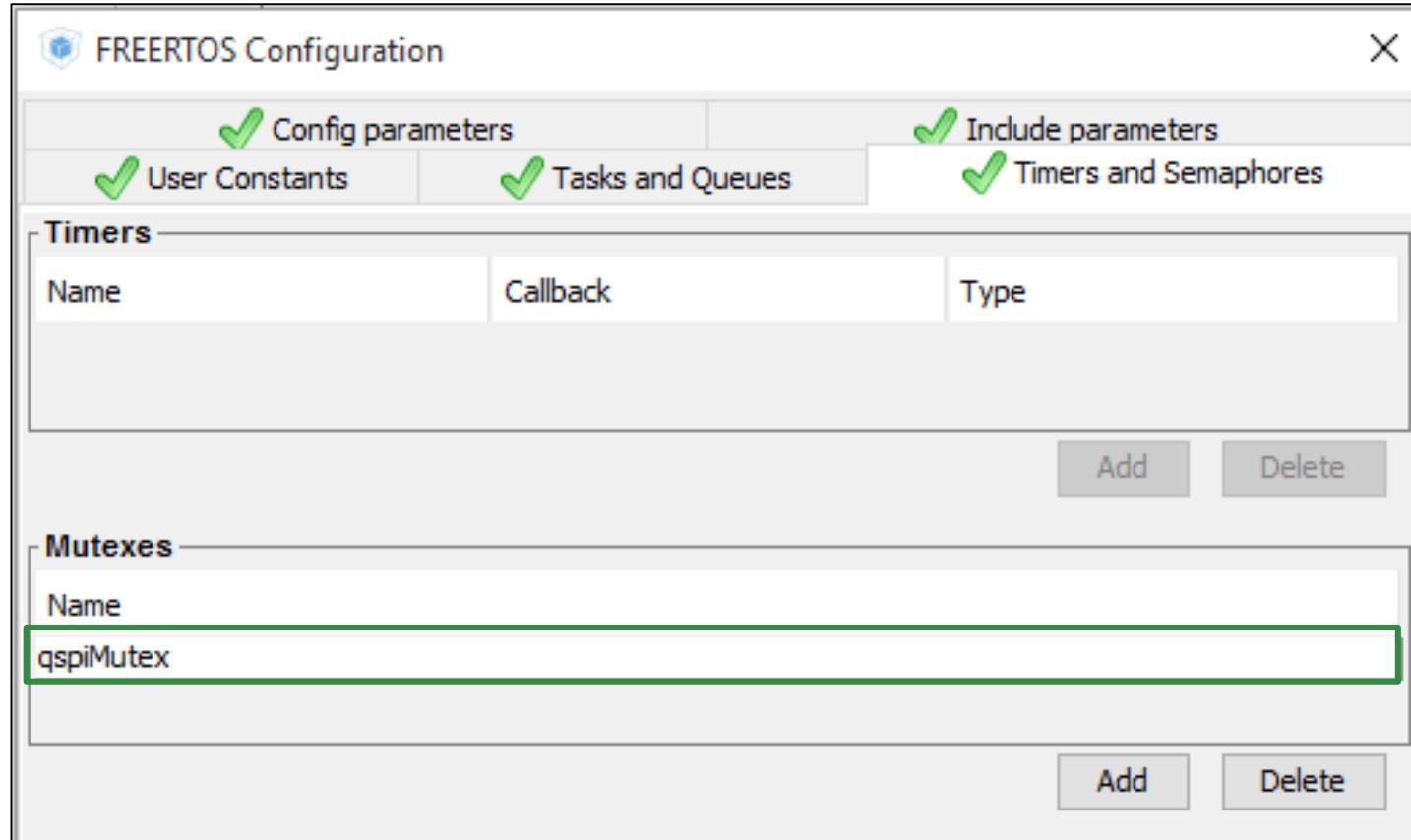
Concurrent access not  
allowed



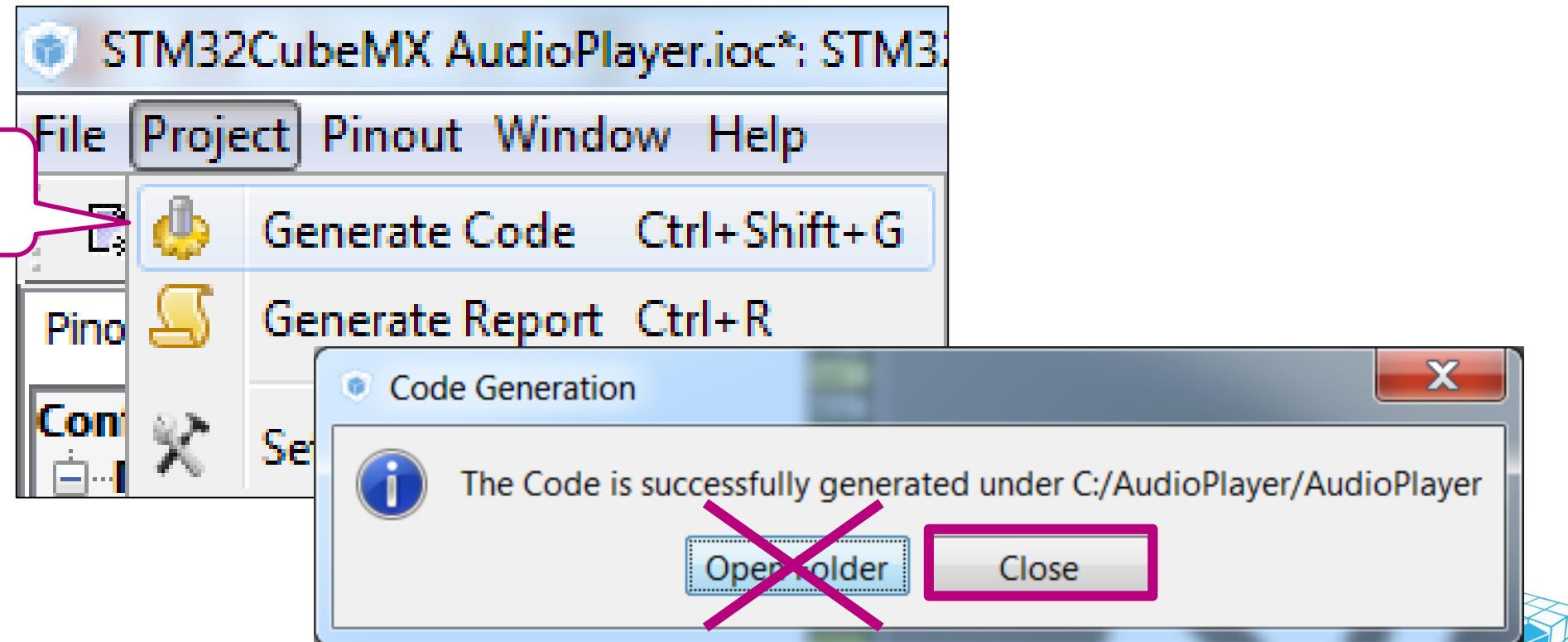


Name → qspiMutex





Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP3\_FileSystem**  
to root folder of your project

(C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Src\

freertos.c

user\_diskio.c

This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

- These files contains the main needed source code the user has to add to have the FatFS working together with QSPI FLASH storage

- user\_diskio.c

USER\_initialize (...)

USER\_status (...)



Storage initialization and Status info

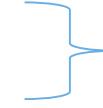
USER\_read (...)

USER\_write (...)



Read / Write operations

USER\_ioctl (...)



IO control

- `f_mount(...)` – mount the storage
  - `f_mkfs(...)` – create filesystem if not existing
- `f_open(...)` – open/create file with name FATFSOK
- `f_printf(...)` – write “FatFS is working properly.” into FATFSOK file
- `f_close(...)` – close the file, no more needed

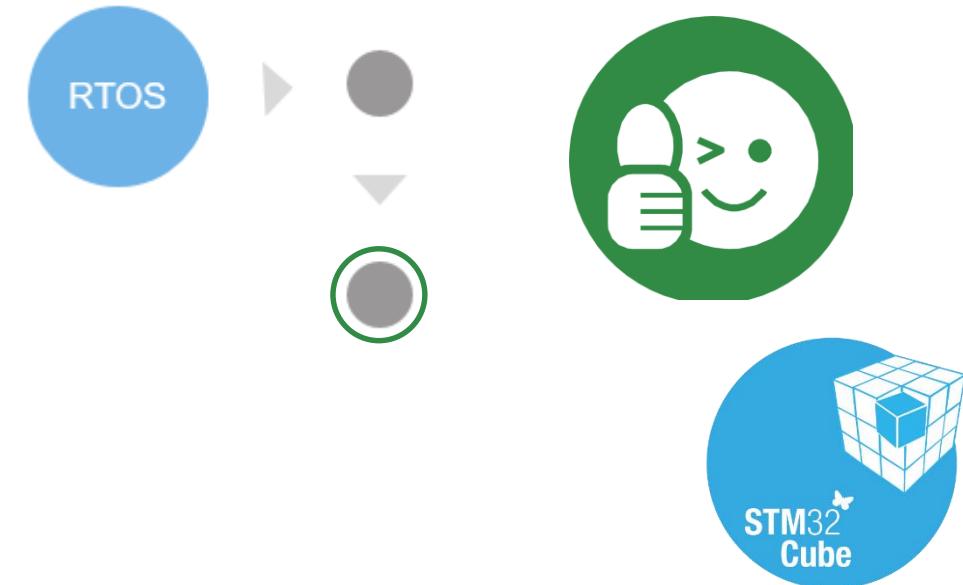
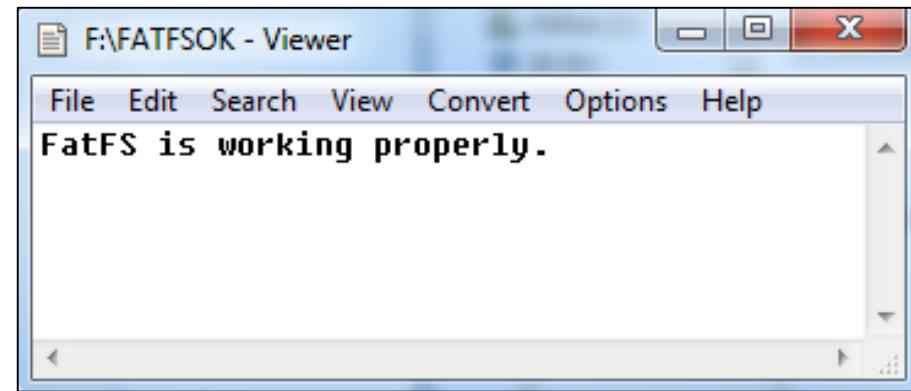
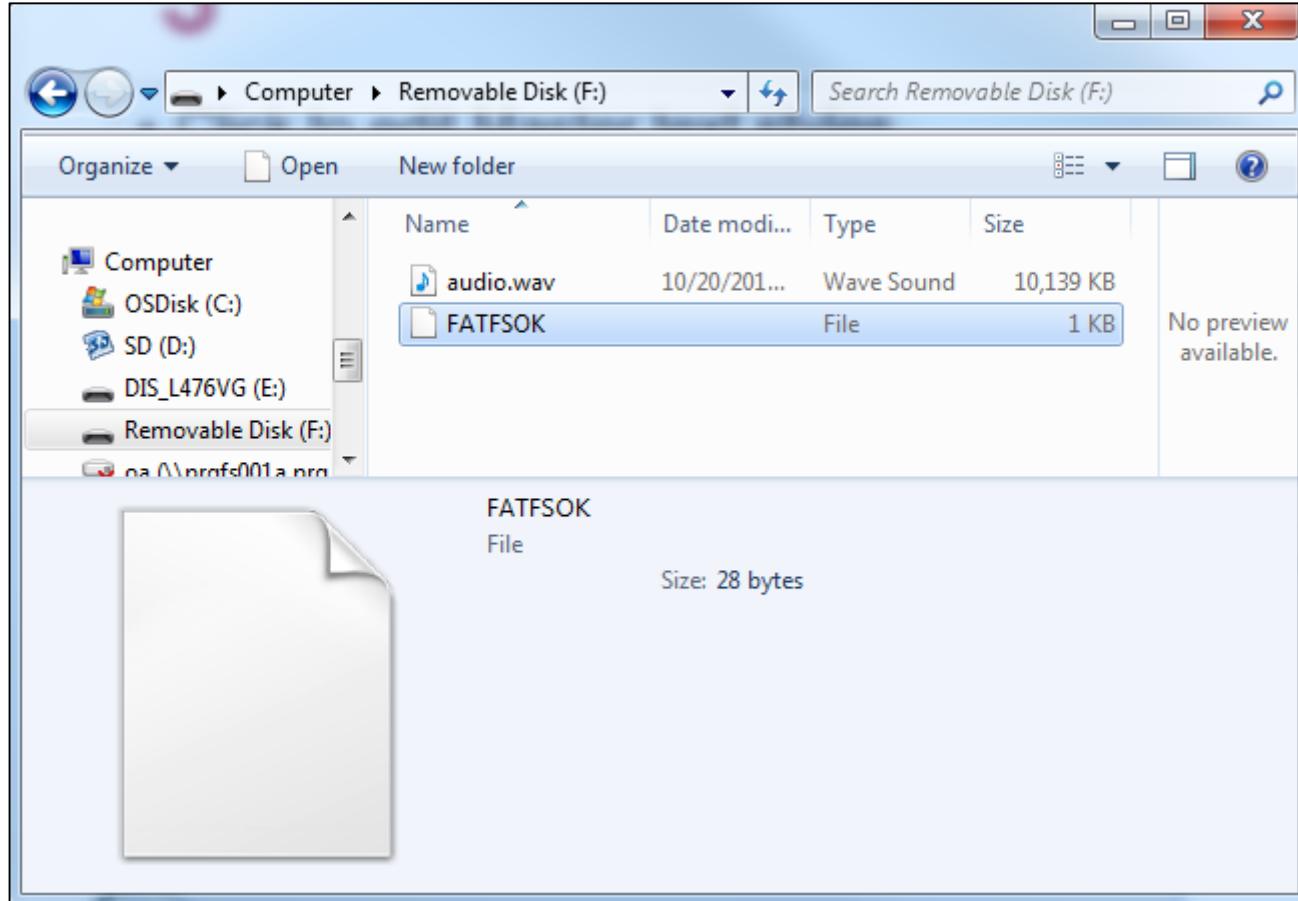
**FATFSOK file in the root of the connected storage  
Content should be (FatFS is working properly.)**

1. Open the project
2. Compile and download
3. Press the reset button on discovery once download has finished
4. Try to connect microUSB to PC

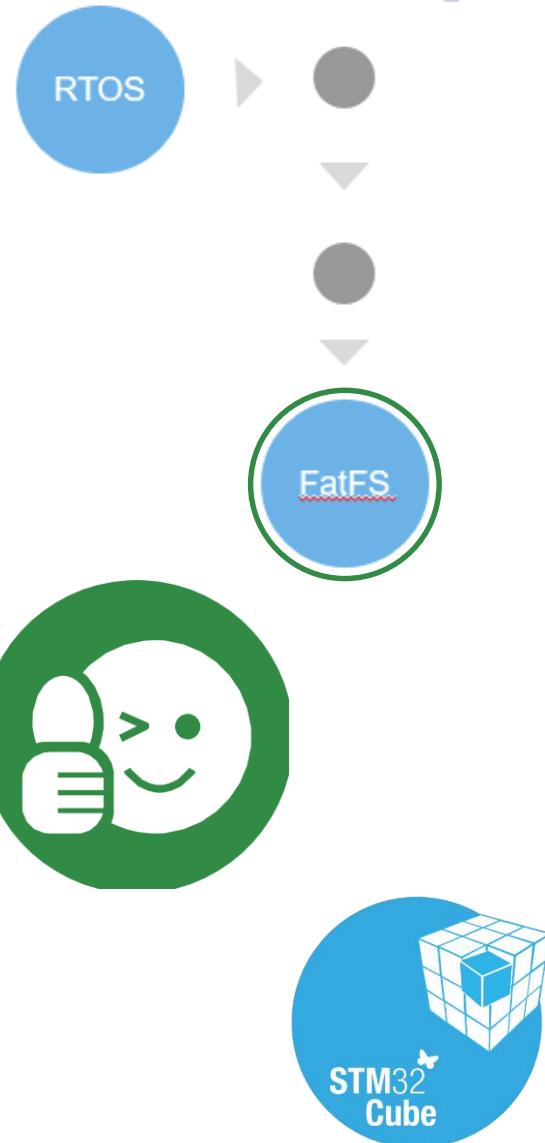
# 3

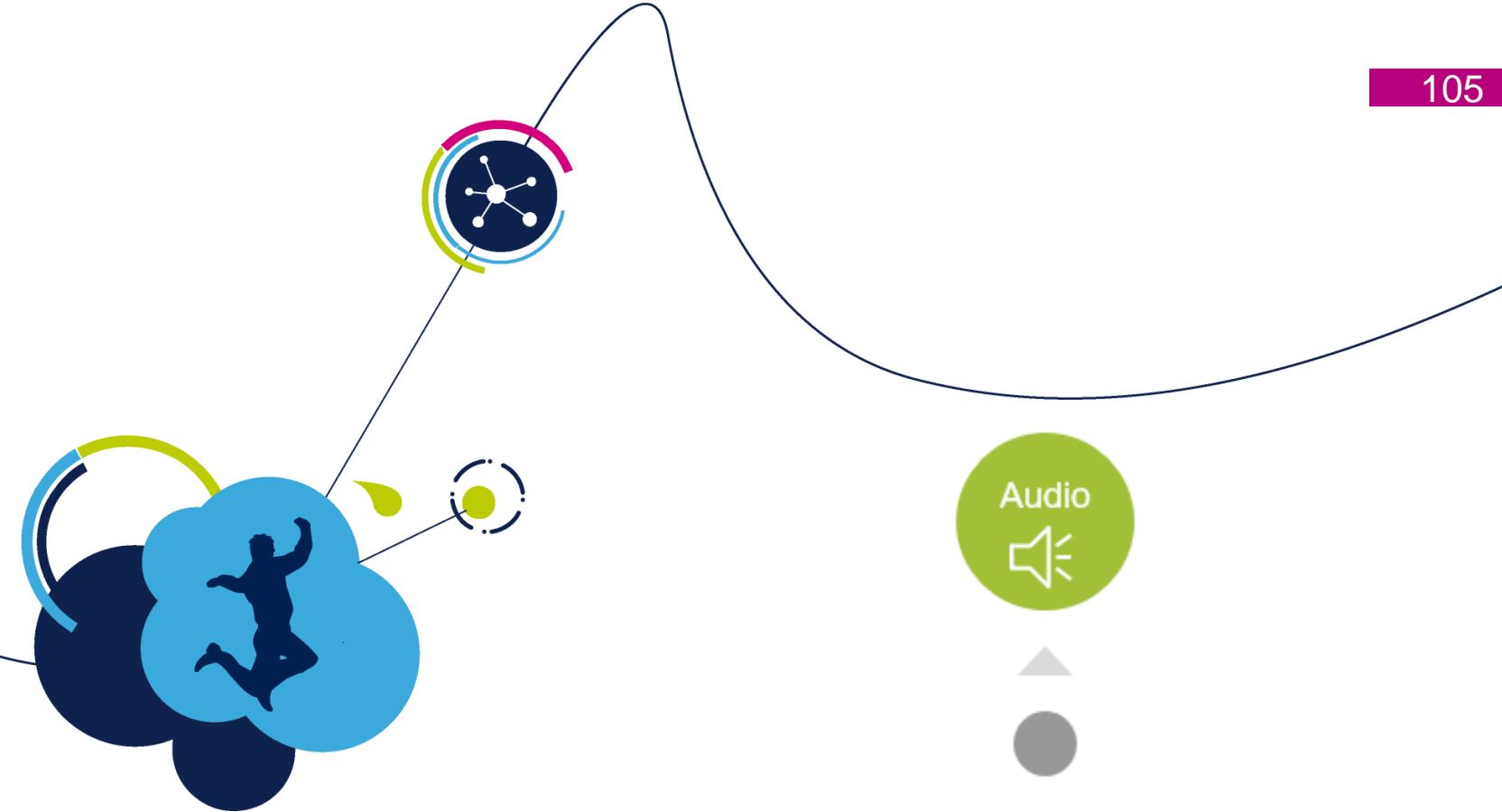
# Checkpoint #7

103



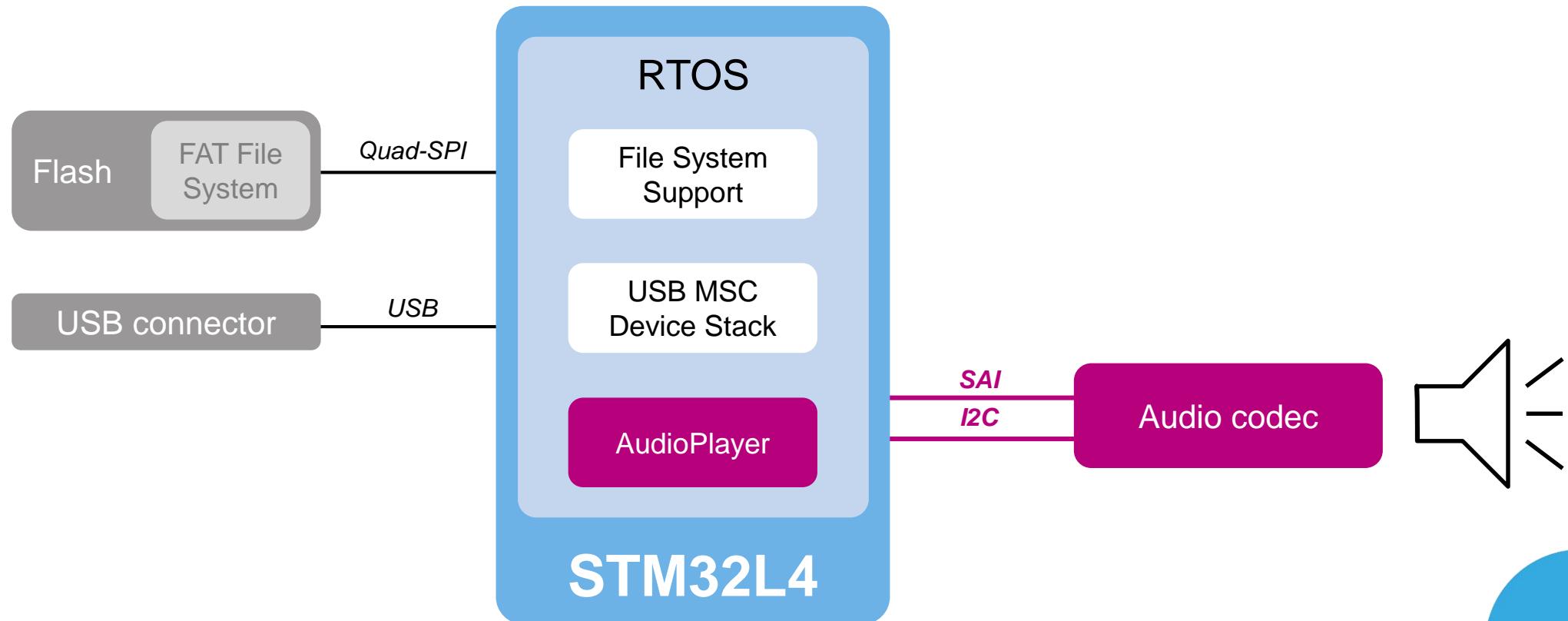
- LED\_GREEN ON before qspiMutex take in appTask
- LED\_GREEN OFF after qspiMutex release in appTask
- LED\_RED ON on qspiMutex take in usbTask
- LED\_RED OFF on qspiMutex release in usbTask

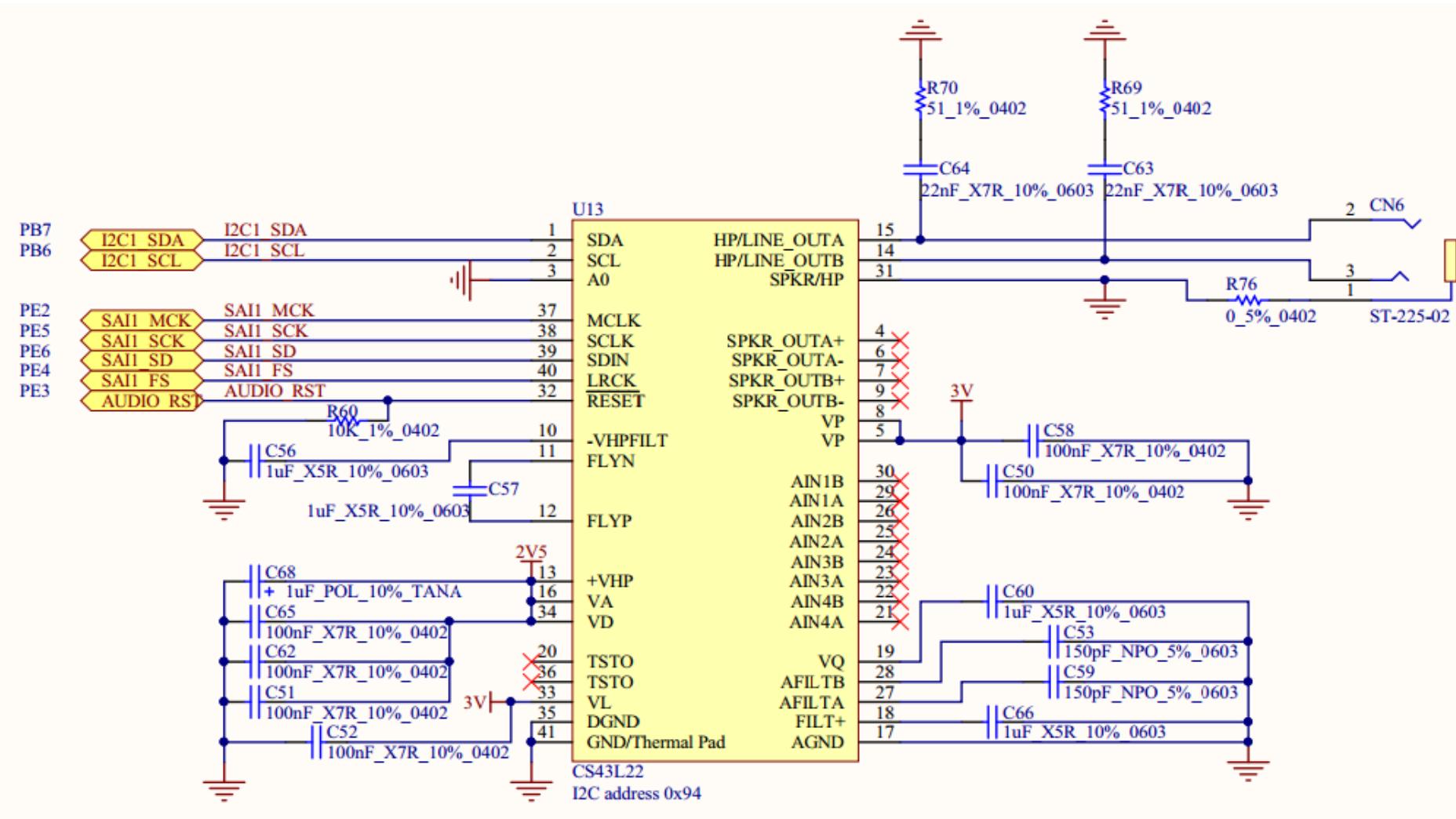


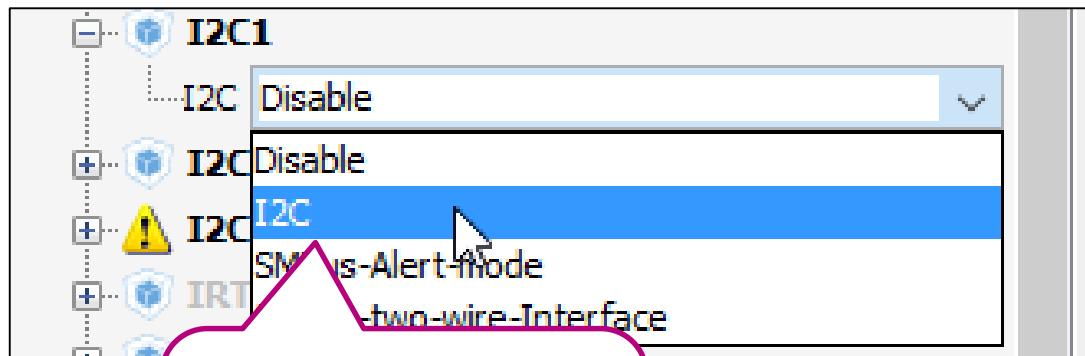
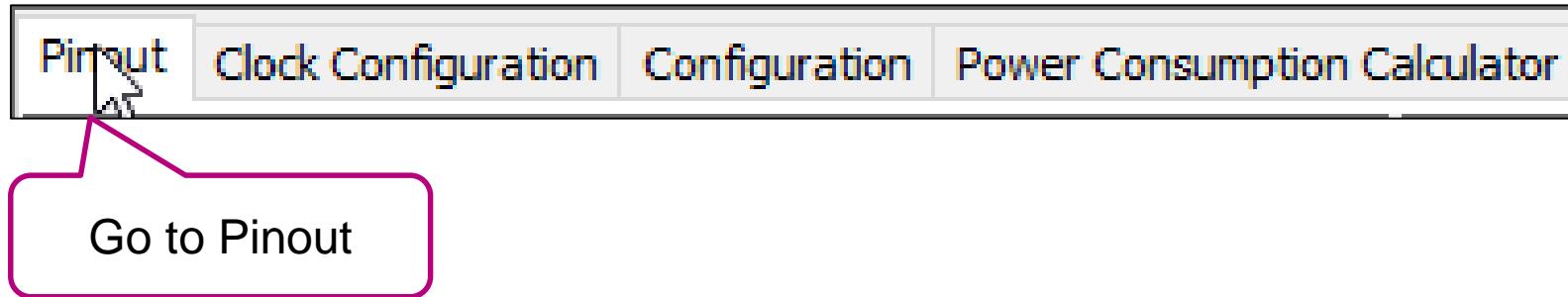


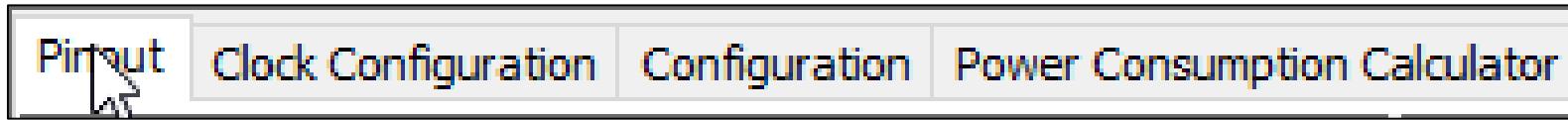
## STEP4 – Audio

- We want to hear our “audio.wav” file in headphones

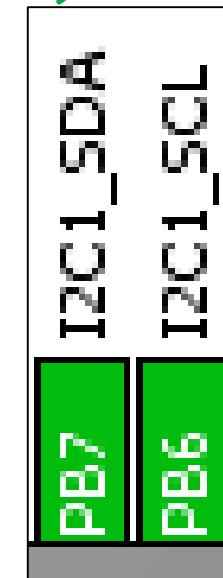
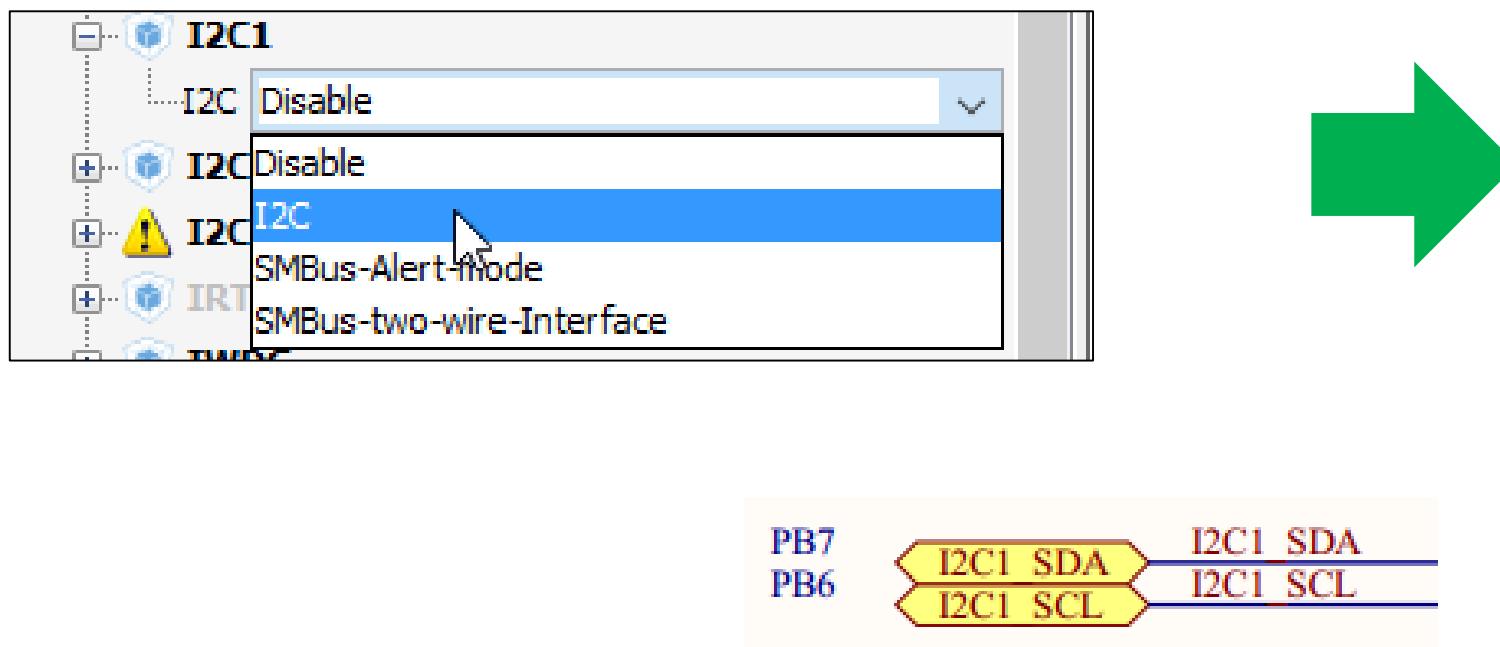


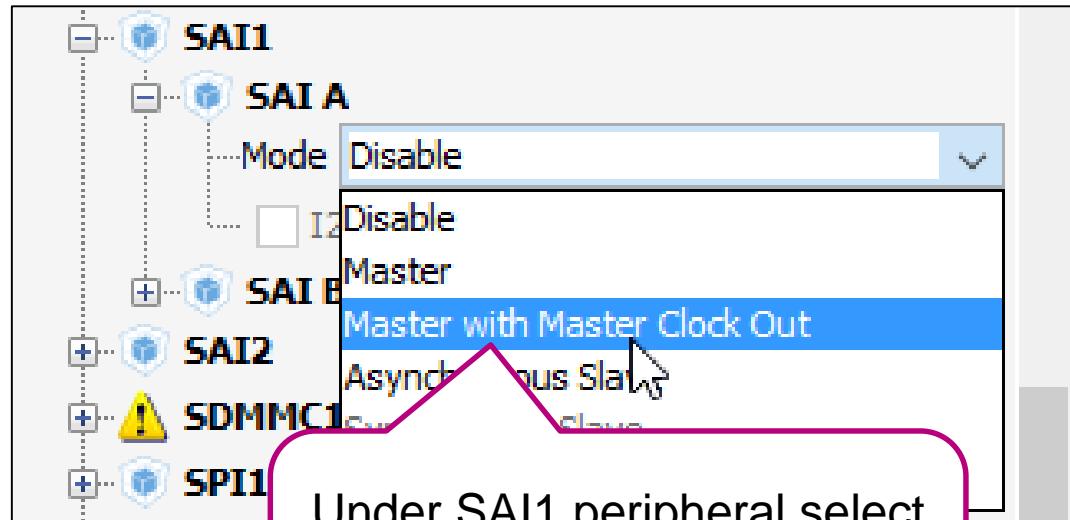
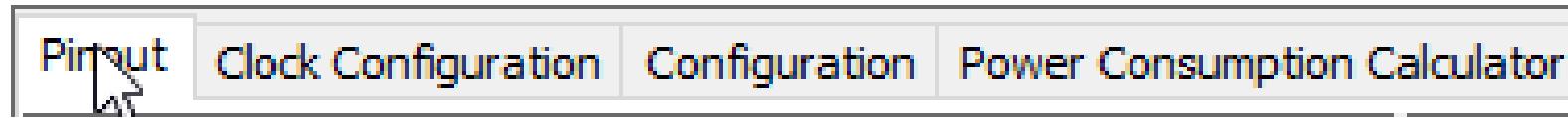




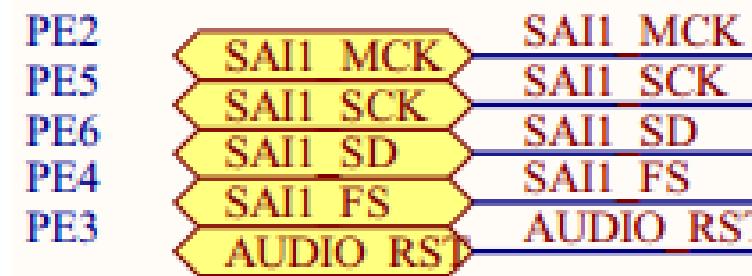


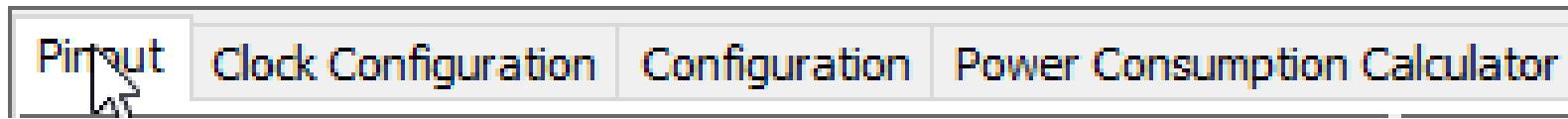
The corresponding pins are assigned and configured automatically!



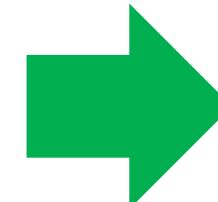
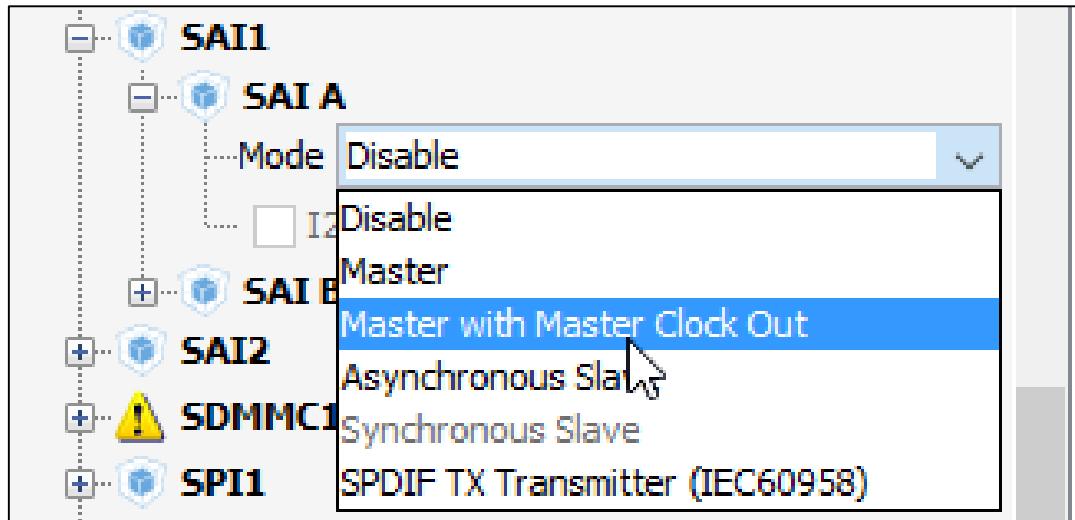


Under SAI1 peripheral select  
**SAI A** interface to **Master**  
**with Master Clock Out**

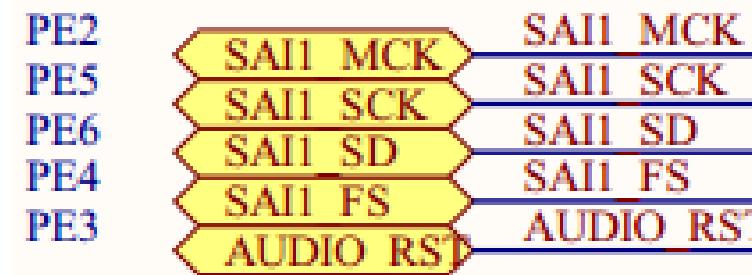


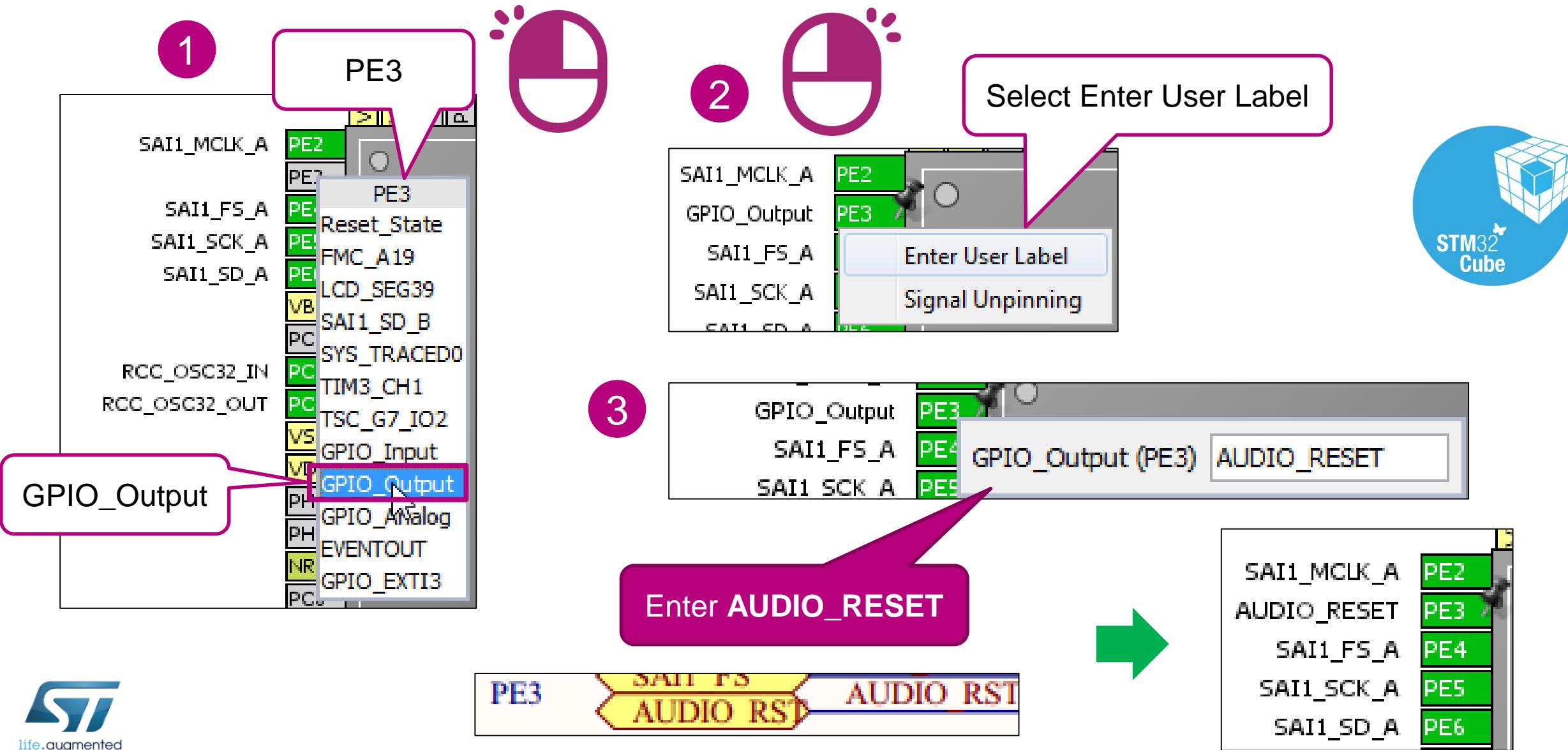


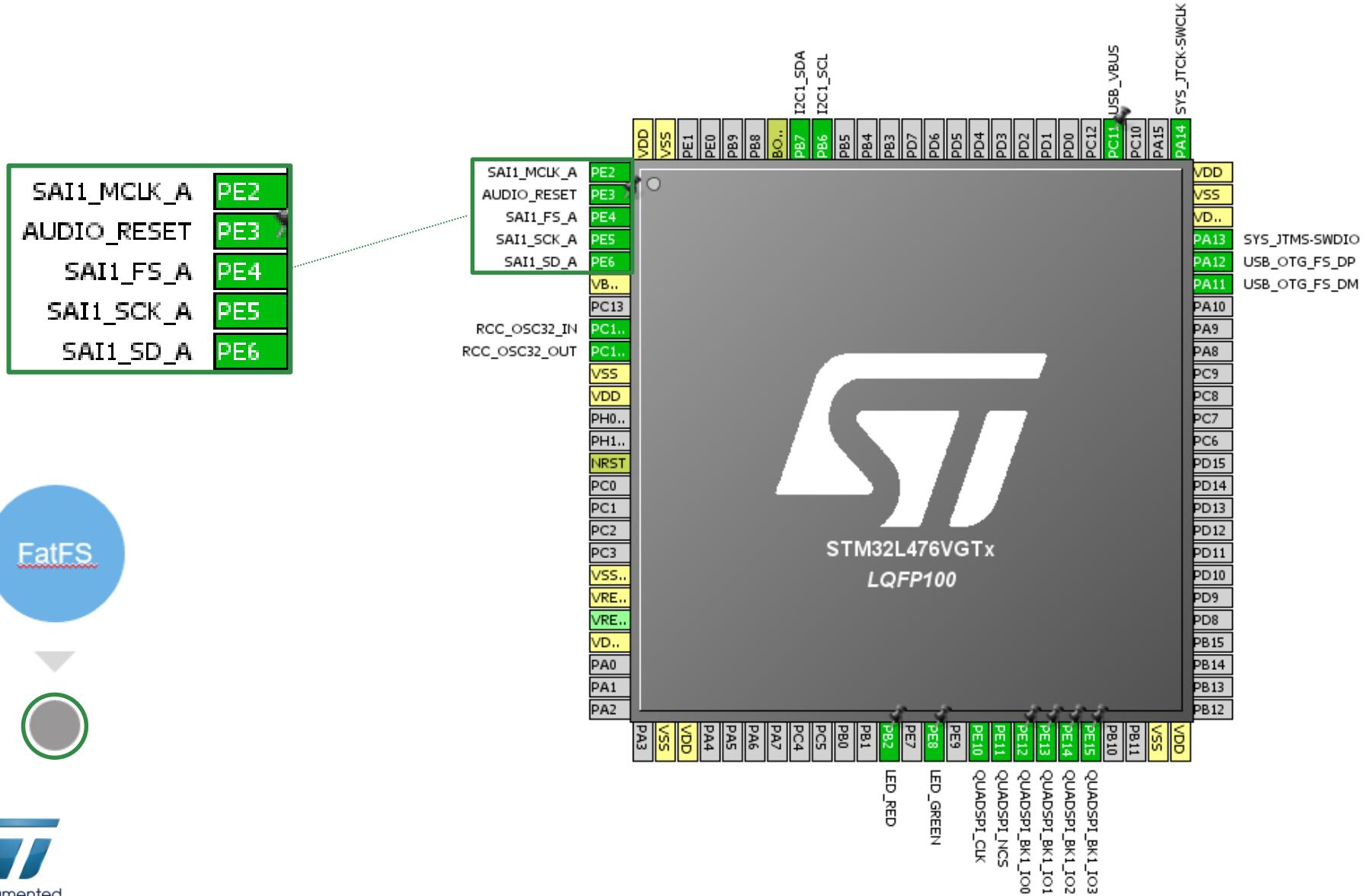
The corresponding pins are assigned and configured automatically!



SAI1_MCLK_A	PE2
SAI1_FS_A	PE3
SAI1_SCK_A	PE4
SAI1_SD_A	PE5







## SAI MCLK configuration

audio.wav	
WAVE Information	
Length	00:58
Compression	PCM
Bitrate (bps)	16
Frequency (Hz)	44 100
Mode	Stereo



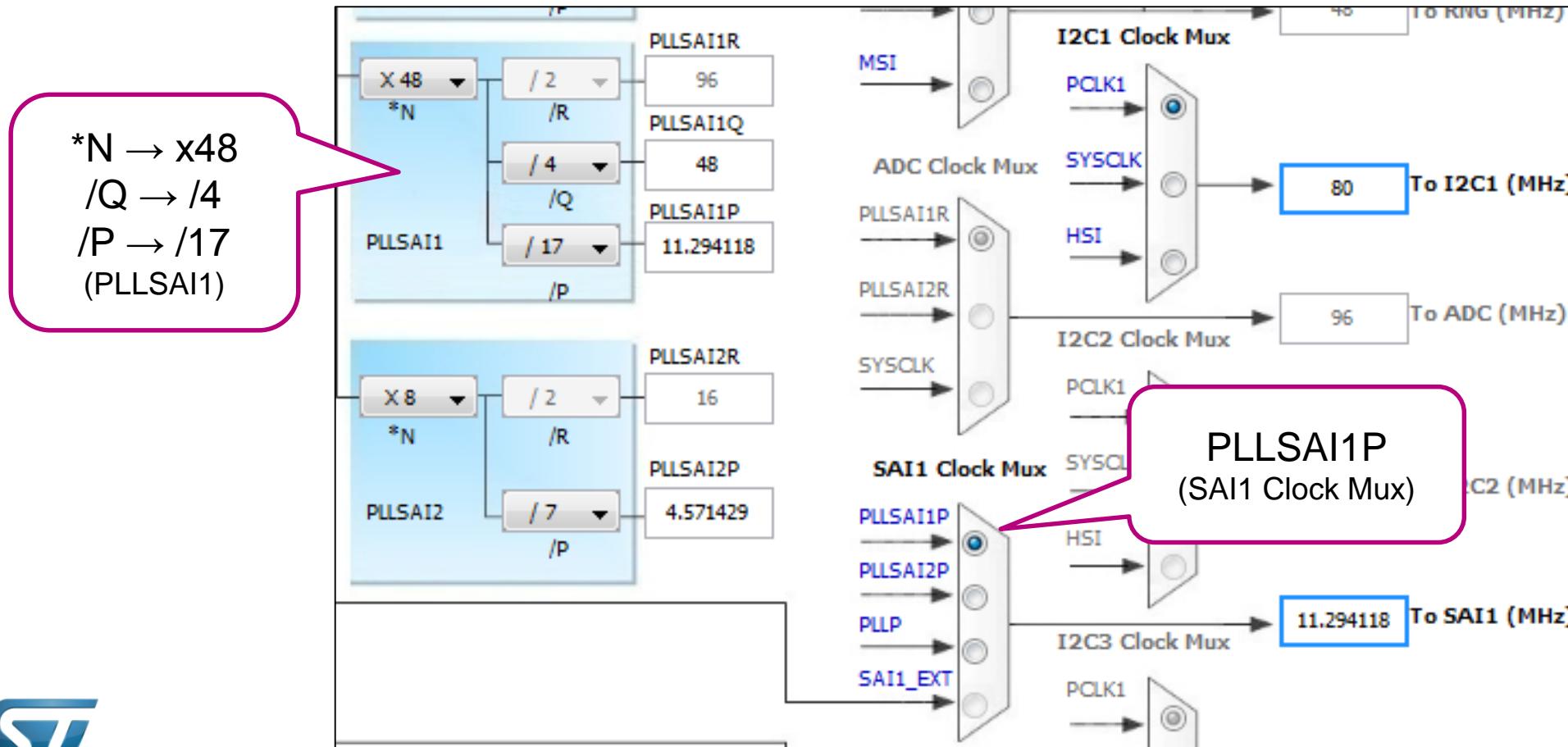
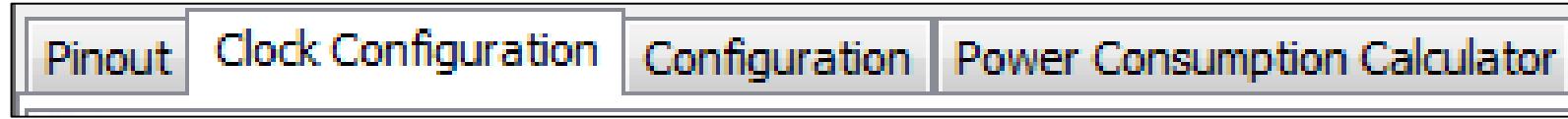
CS43L22

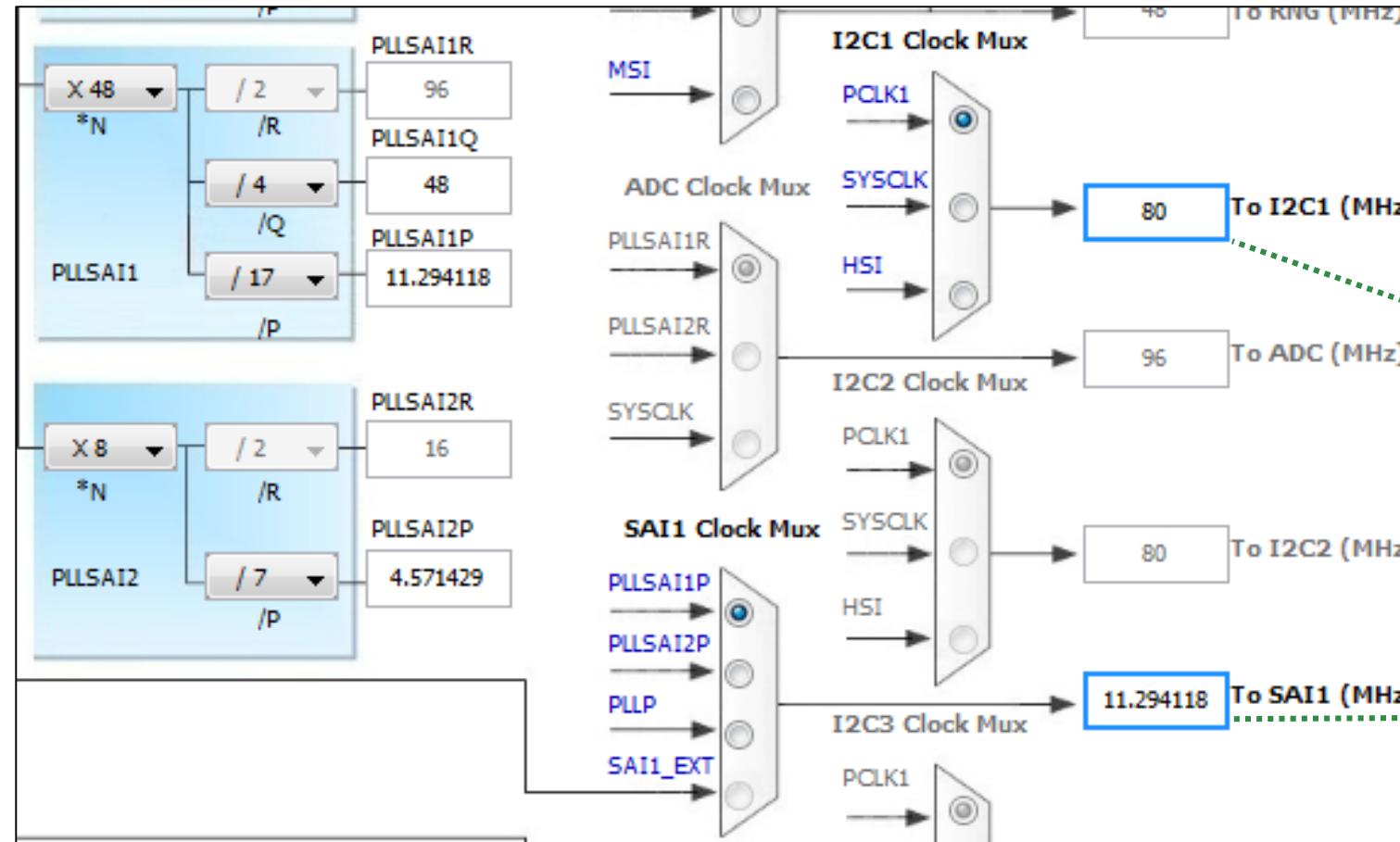
## 4.6 Serial Port Clocking

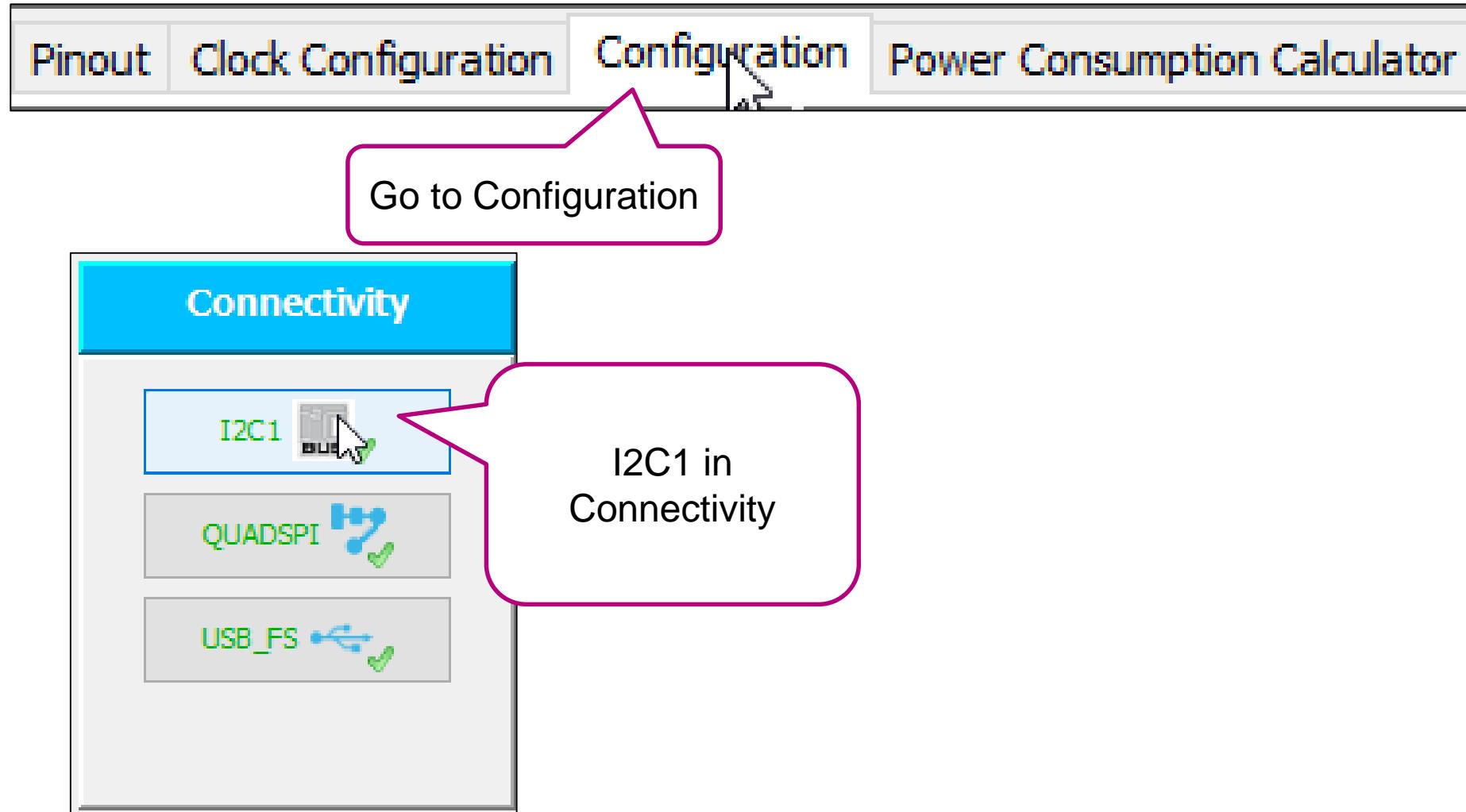
The CS43L22 serial audio interface port operates either as a slave or master, determined by the M/S bit. It accepts externally generated clocks in Slave Mode and will generate synchronous clocks derived from an input master clock (MCLK) in Master Mode. Refer to the tables below for the required setting in register 05h and 06h associated with a given MCLK and sample rate.

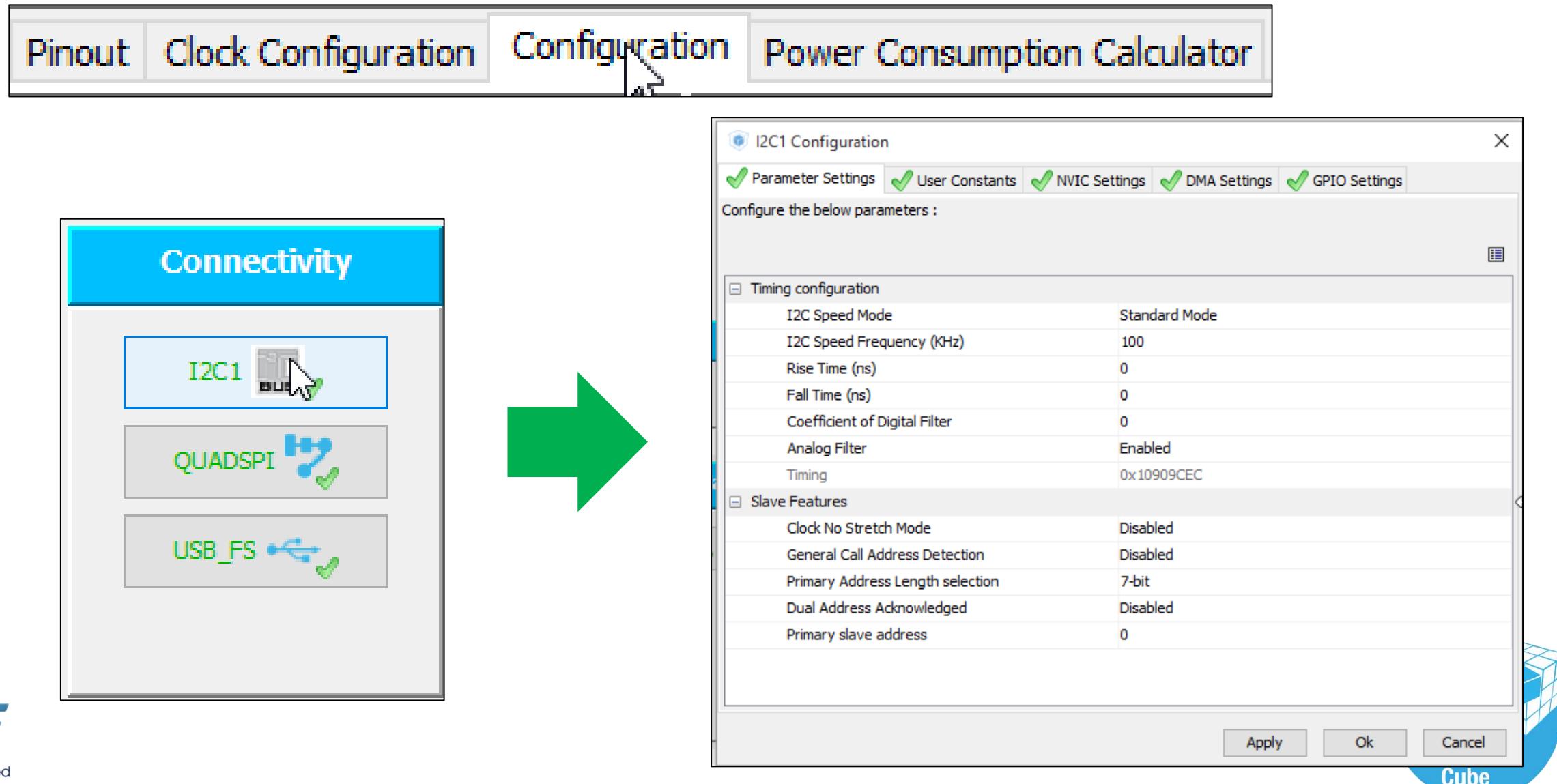
Referenced Control	Register Location
M/S.....	"Master/Slave Mode" on page 40
Register 05h.....	"Clocking Control (Address 05h)" on page 38
Register 06h.....	"Interface Control 1 (Address 06h)" on page 40

MCLK (MHz)	Sample Rate, Fs (kHz)	SPEED[1:0] (AUTO='0'b)	32kGROUP	VIDEOCLK	RATIO[1:0]	MCLKDIV2
12.2880	8.0000	11	1	0	00	0
	12.0000	11	0	0	00	0
	16.0000	10	1	0	00	0
	24.0000	10	0	0	00	0
	32.0000	01	1	0	00	0
	48.0000	01	0	0	00	0
	96.0000	00	0	0	00	0
11.2896	11.0250	11	0	0	00	0
	22.0500	10	0	0	00	0
	44.1000	01	0	0	00	0
	88.2000	00	0	0	00	0









The image shows the STM32CubeMX software interface. On the left, a 'Connectivity' configuration window is open, displaying three options: I2C1, QUADSPI, and USB\_FS. The I2C1 option is selected, indicated by a green checkmark icon. A large green arrow points from this window to a detailed configuration dialog box on the right. The dialog box is titled 'I2C1 Configuration' and contains tabs for Parameter Settings, User Constants, NVIC Settings, DMA Settings, and GPIO Settings. The 'Parameter Settings' tab is selected. The configuration parameters are listed under 'Timing configuration' and 'Slave Features'.

**I2C1 Configuration**

Configure the below parameters :

Timing configuration	
I2C Speed Mode	Standard Mode
I2C Speed Frequency (KHz)	100
Rise Time (ns)	0
Fall Time (ns)	0
Coefficient of Digital Filter	0
Analog Filter	Enabled
Timing	0x10909CEC
Slave Features	
Clock No Stretch Mode	Disabled
General Call Address Detection	Disabled
Primary Address Length selection	7-bit
Dual Address Acknowledged	Disabled
Primary slave address	0

Buttons at the bottom: Apply, Ok, Cancel



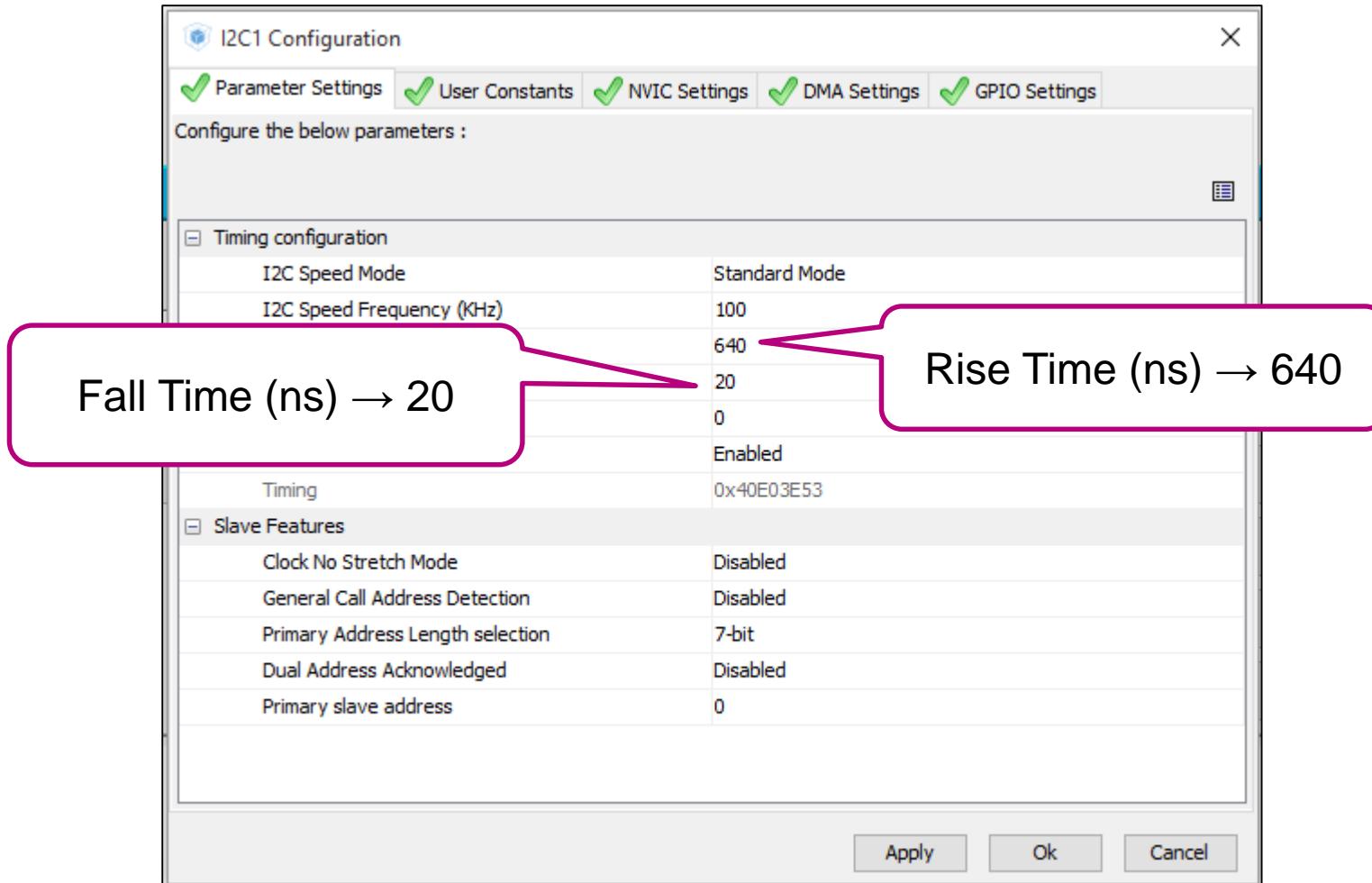
**CIRRUS LOGIC®**

**CS43L22**

**SWITCHING SPECIFICATIONS - I<sup>2</sup>C CONTROL PORT**

Inputs: Logic 0 = DGND; Logic 1 = V; SDA C<sub>L</sub> = 30 pF.

Parameters	Symbol	Min	Max	Unit
SCL Clock Frequency	f <sub>scl</sub>	-	100	kHz
RESET Rising Edge to Start	t <sub>irs</sub>	550	-	ns
Bus Free Time Between Transmissions	t <sub>buf</sub>	4.7	-	μs
Start Condition Hold Time (prior to first clock pulse)	t <sub>hdst</sub>	4.0	-	μs
Clock Low time	t <sub>low</sub>	4.7	-	μs
Clock High Time	t <sub>high</sub>	4.0	-	μs
Setup Time for Repeated Start Condition	t <sub>sust</sub>	4.7	-	μs
SDA Hold Time from SCL Falling	(Note 13) t <sub>hdd</sub>	0	-	μs
SDA Setup time to SCL Rising	t <sub>sud</sub>	250	-	ns
Rise Time of SCL and SDA	t <sub>rc</sub>	-	1	μs
Fall Time SCL and SDA	t <sub>fc</sub>	-	300	ns
Setup Time for Stop Condition	t <sub>susp</sub>	4.7	-	μs
Acknowledge Delay from SCL Falling	t <sub>ack</sub>	300	1000	ns





SAI1 in Multimedia

**Multimedia**

**SAI**

**Configuration**

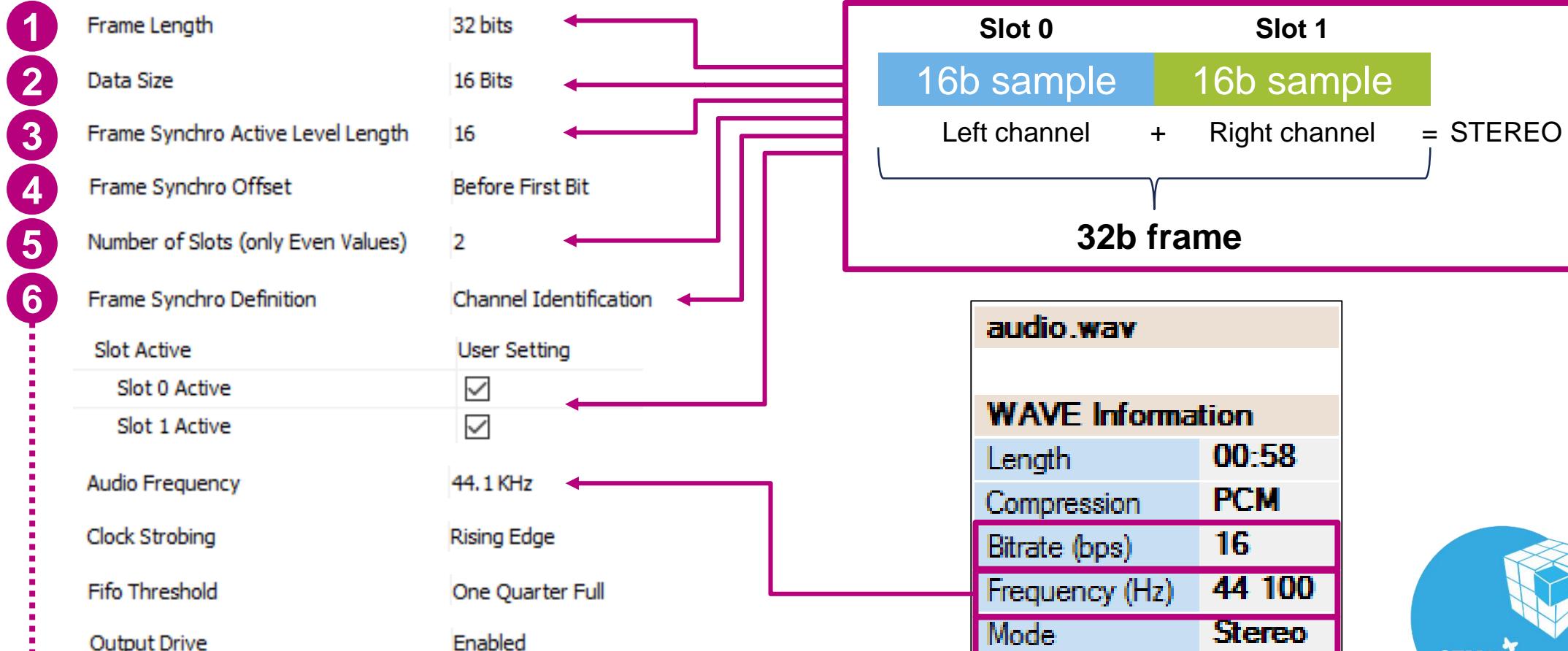
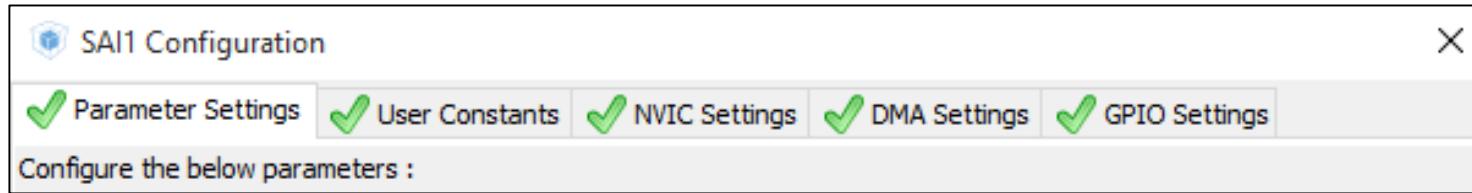
**Power Consumption Calculator**

**SAI1 Configuration**

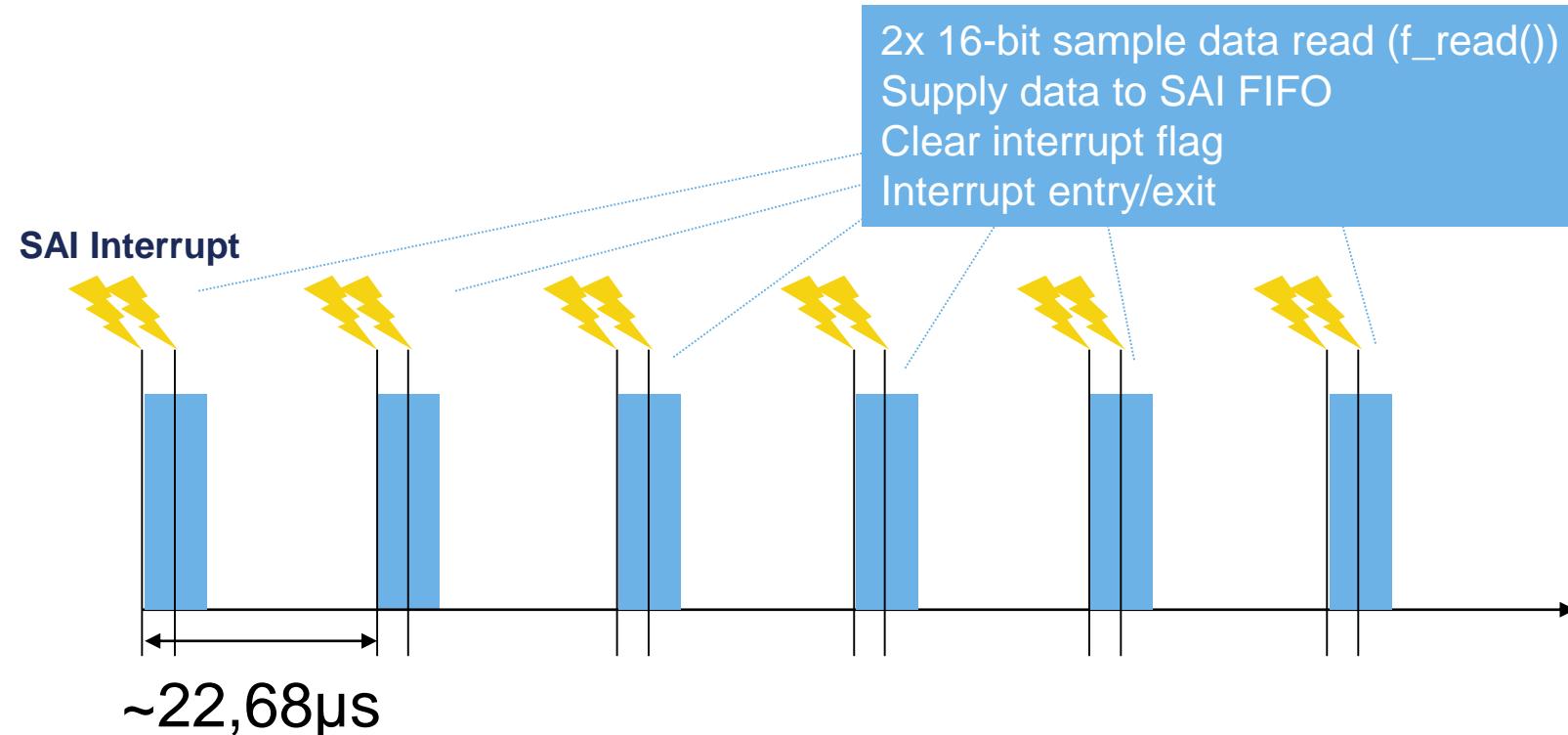
Configure the below parameters :

SAI A	
Basic Parameters	
Protocol	Free
Audio Mode	Master Transmit
Frame Length	8 bits
Data Size	24 Bits
Slot Size	DataService
Output Mode	Stereo
Companding Mode	No companding mode
SAI SD Line Output Mode	Driven
Frame Parameters	
First Bit	MSB First
Frame Synchro Active Level Length	1
Frame Synchro Definition	Start Frame
Frame Synchro Polarity	Active Low
Frame Synchro Offset	First Bit
Slot Parameters	
First Bit Offset	0
Number of Slots	1
Slot Active Final Value	0x00000000
Slot Active	Neither
Clock Parameters	
Master Clock Divider	Enabled
Audio Frequency	192 KHz
Real Audio Frequency	107.142 KHz
Error between Selected	-44.19 %
Clock Strobing	Falling Edge
Advanced Parameters	
Fifo Threshold	Empty
Output Drive	Disabled
Synchronization External	Disabled

Apply   Ok   Cancel



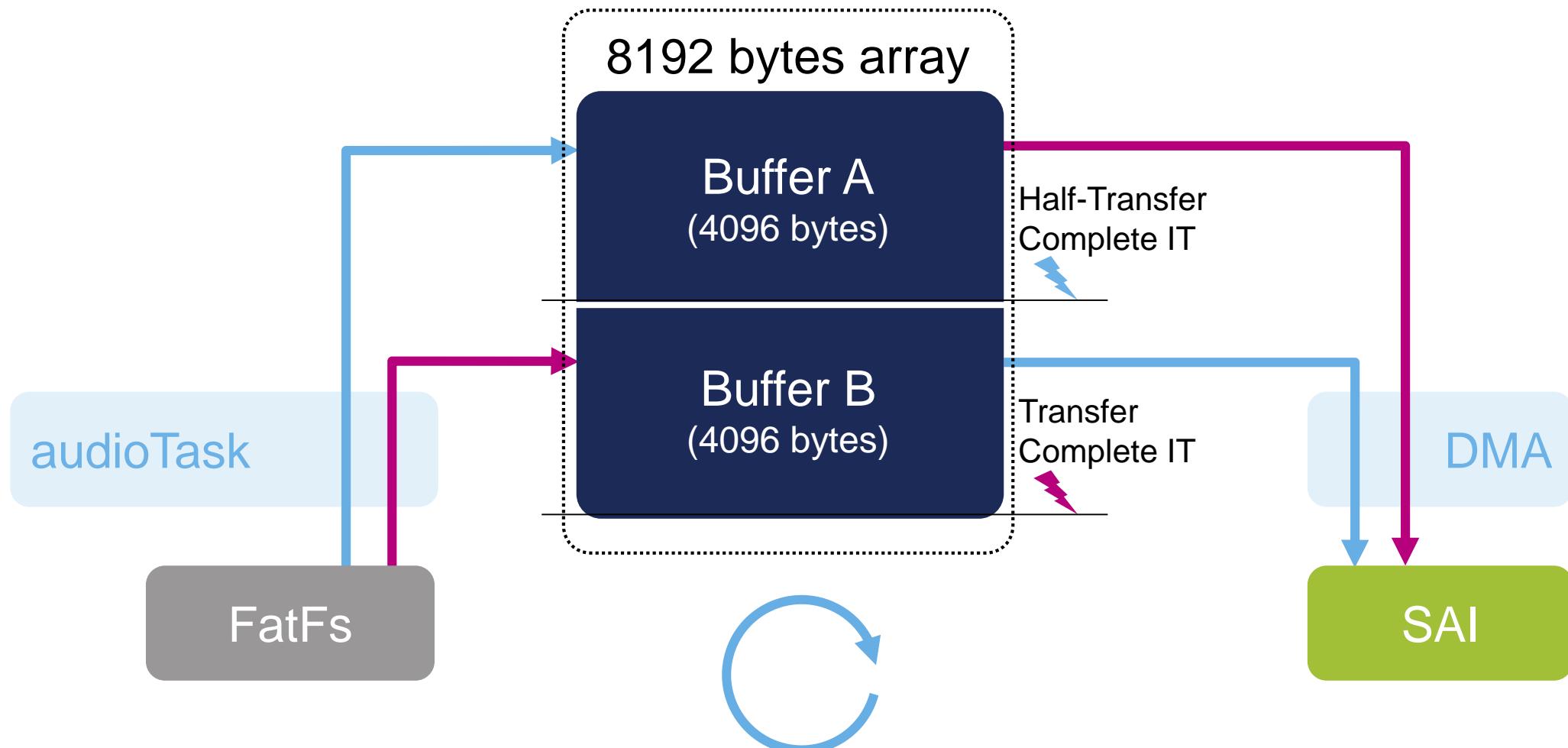
audio.wav	
WAVE Information	
Length	00:58
Compression	PCM
Bitrate (bps)	16
Frequency (Hz)	44 100
Mode	Stereo



~1814 clk cycles @80 MHz → **far enough for the task**

But, is it “best practices” implementation of such task?

NO!



SAI1 Configuration

Parameter Settings User Constants NVIC Settings DMA Settings

DMA Request	Channel	Direction	Priority
SAI1_A	DMA2 Channel 1	Memory To Peripheral	High

DMA Request → SAI1\_A

Channel → DMA2 Channel 1

Direction → Memory To Peripheral

Priority → High

Mode → Circular

Data Width → Half Word (both)

Increment Address → Memory (only)

Add Delete

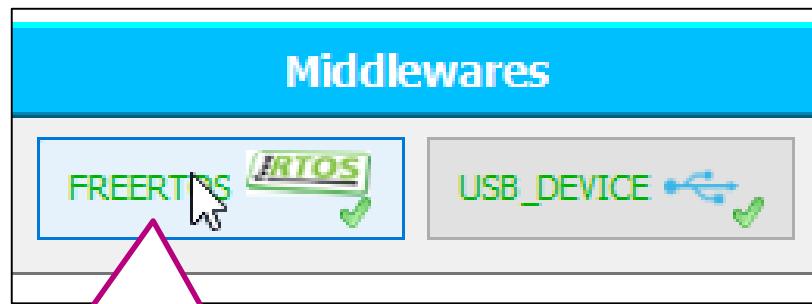
DMA Request Settings

Mode: Circular

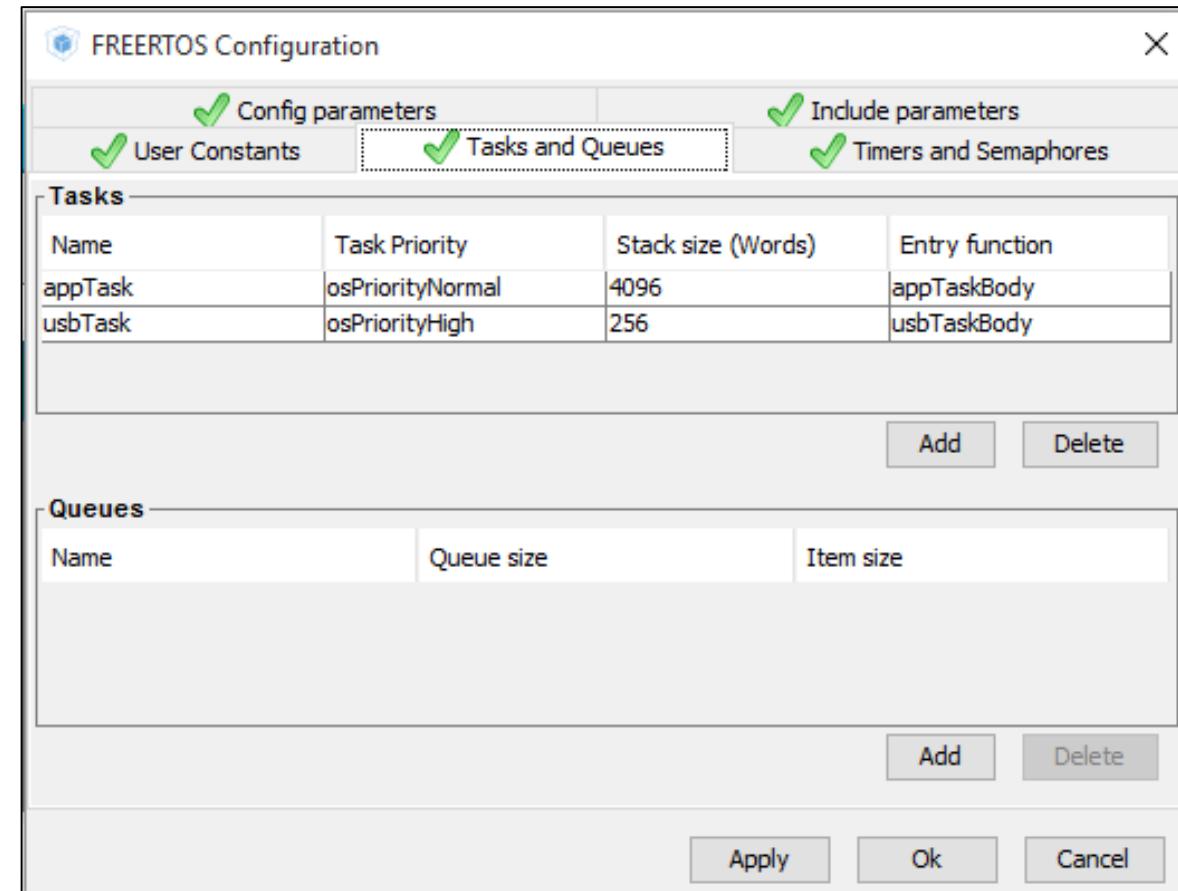
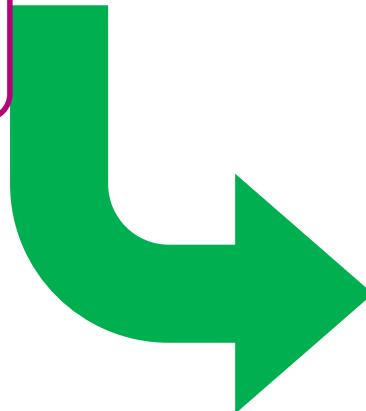
Increment Address:  Peripheral  Memory

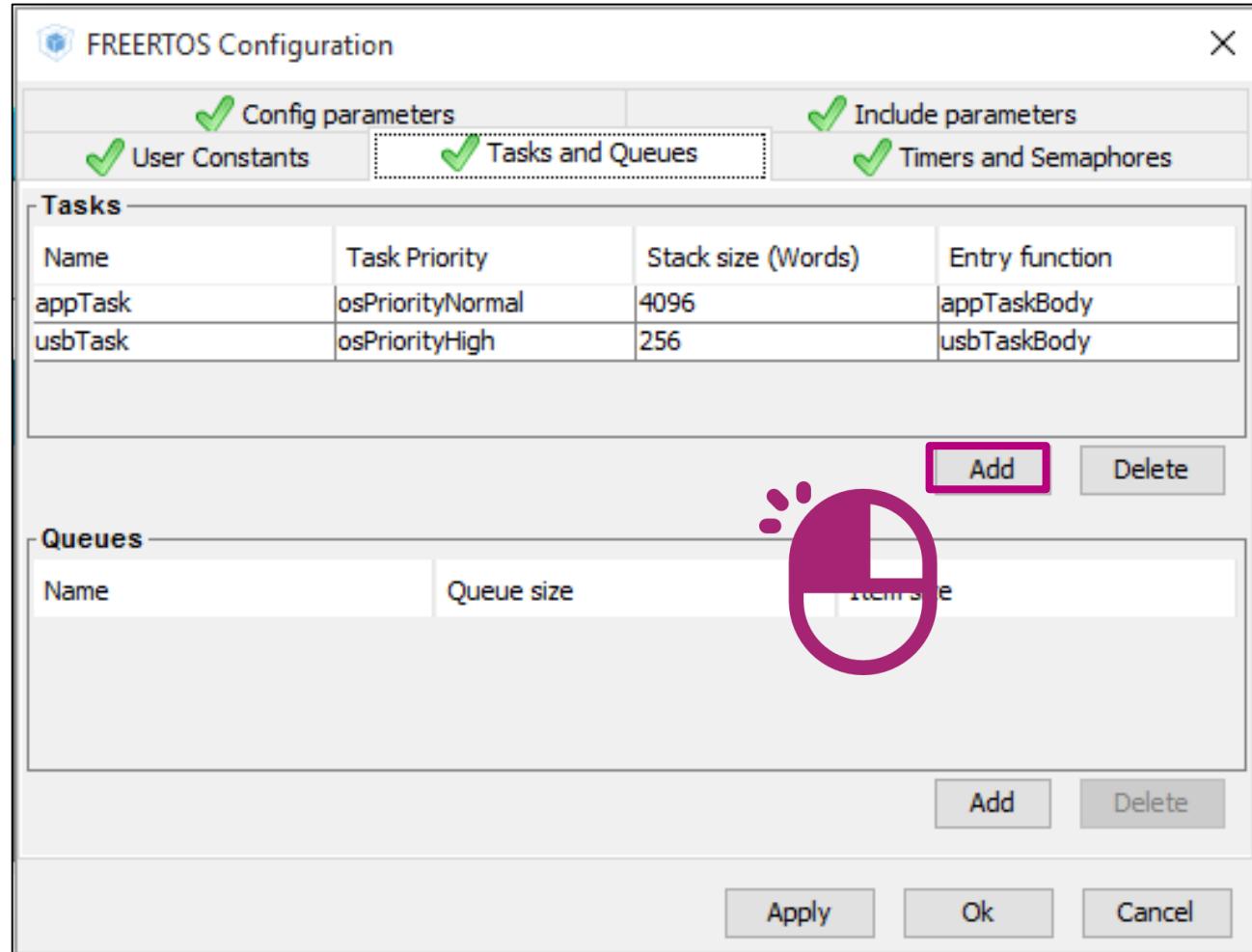
Data Width: Half Word (both)

Apply OK Cancel



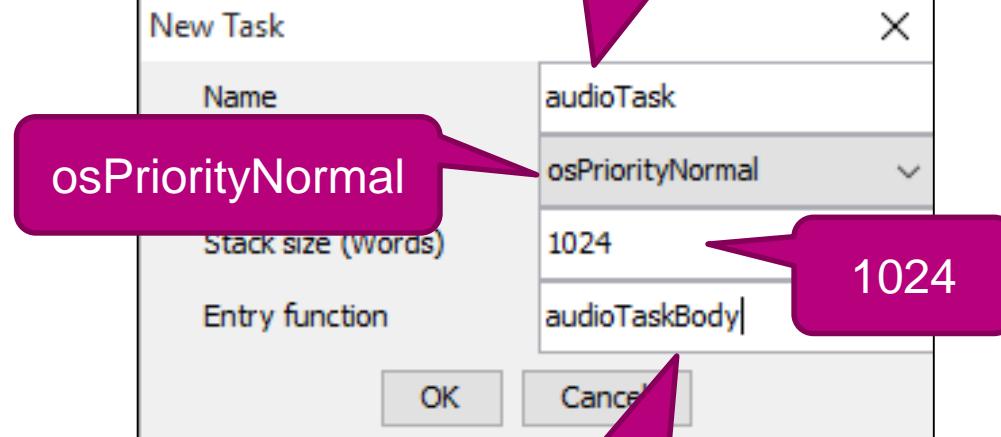
FREERTOS in  
Middlewares





audioTask

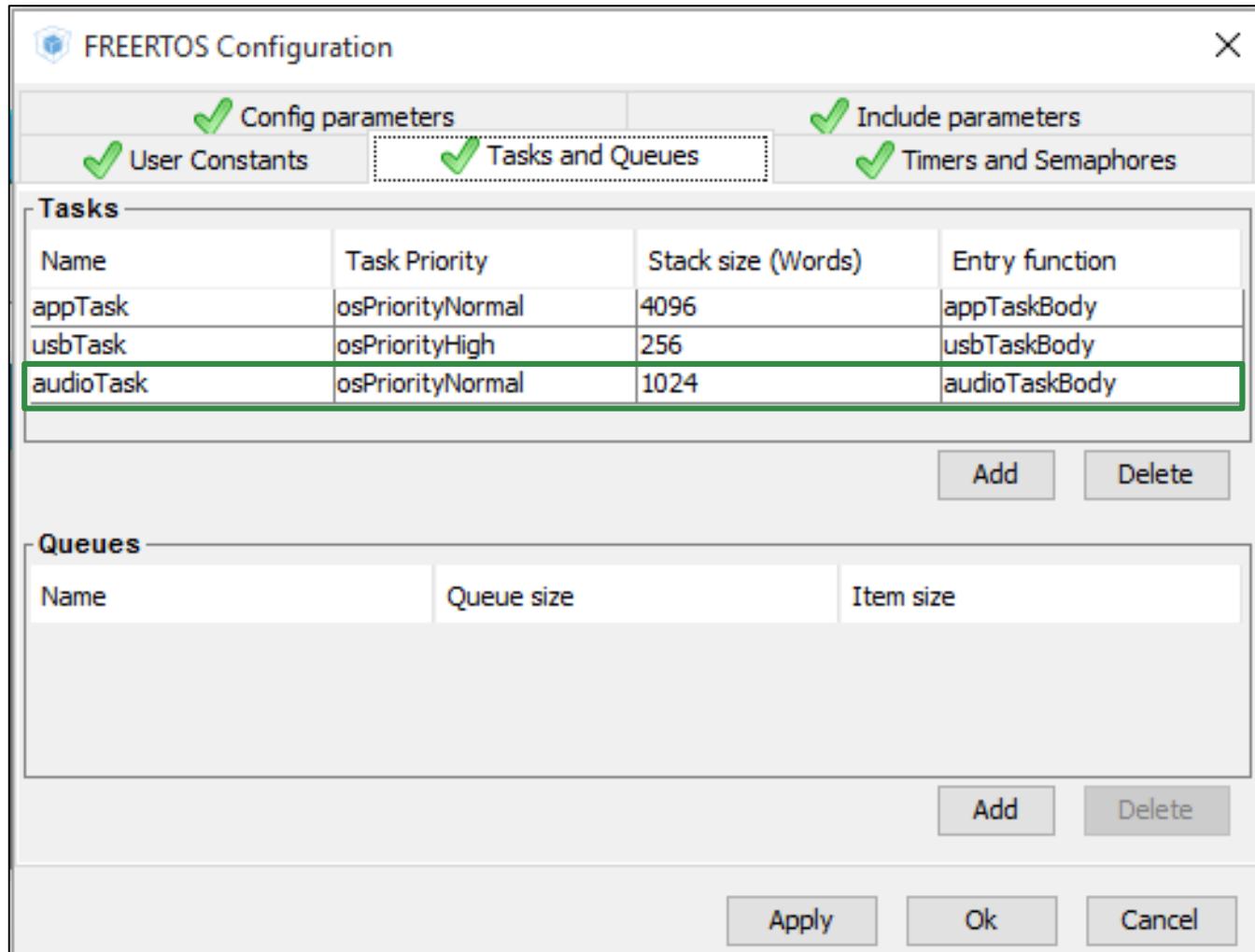
audioTask

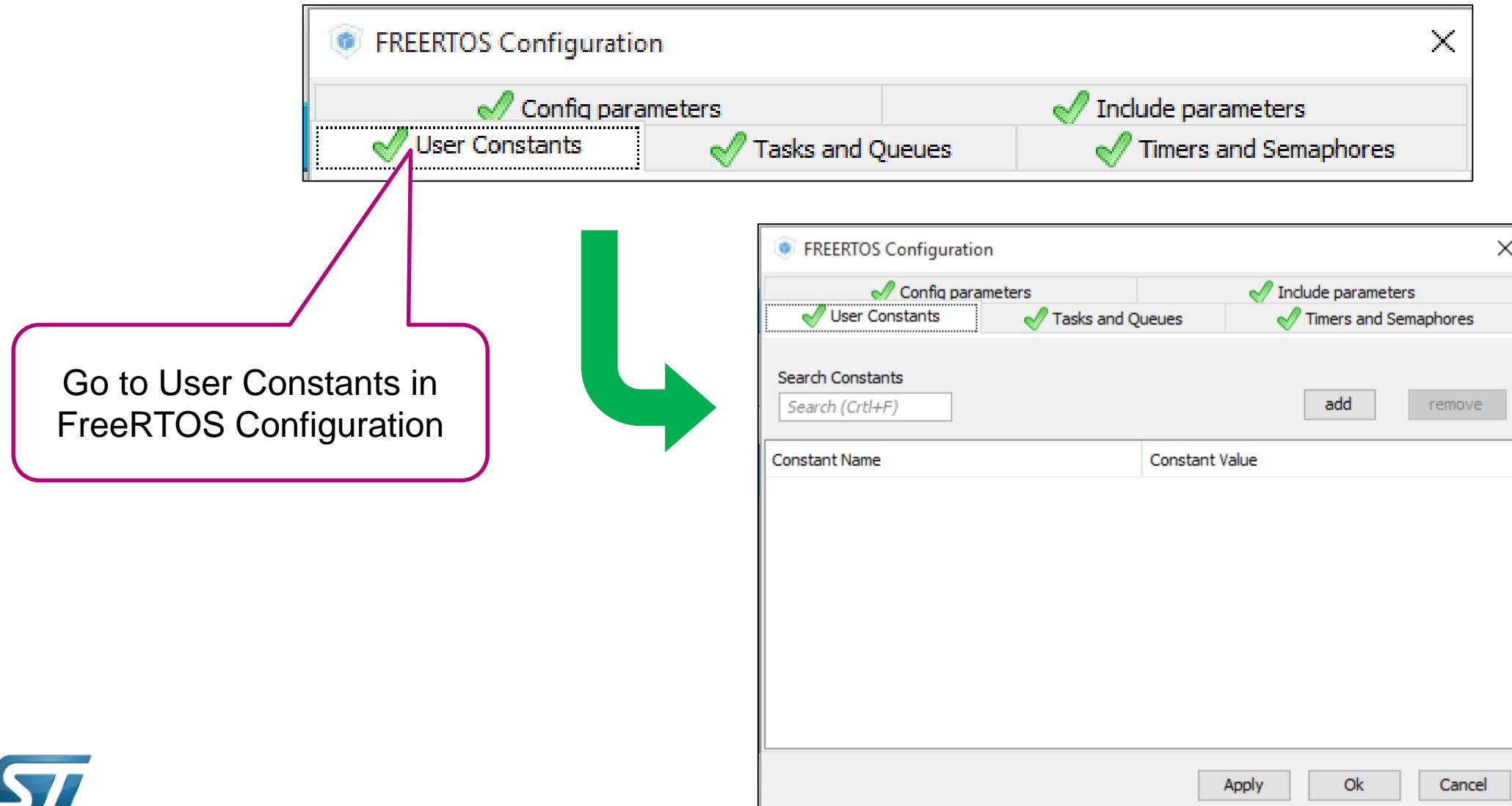


osPriorityNormal

1024

audioTaskBody





A

User Constants	
constant Name	A
constant Value	0x1

0x1

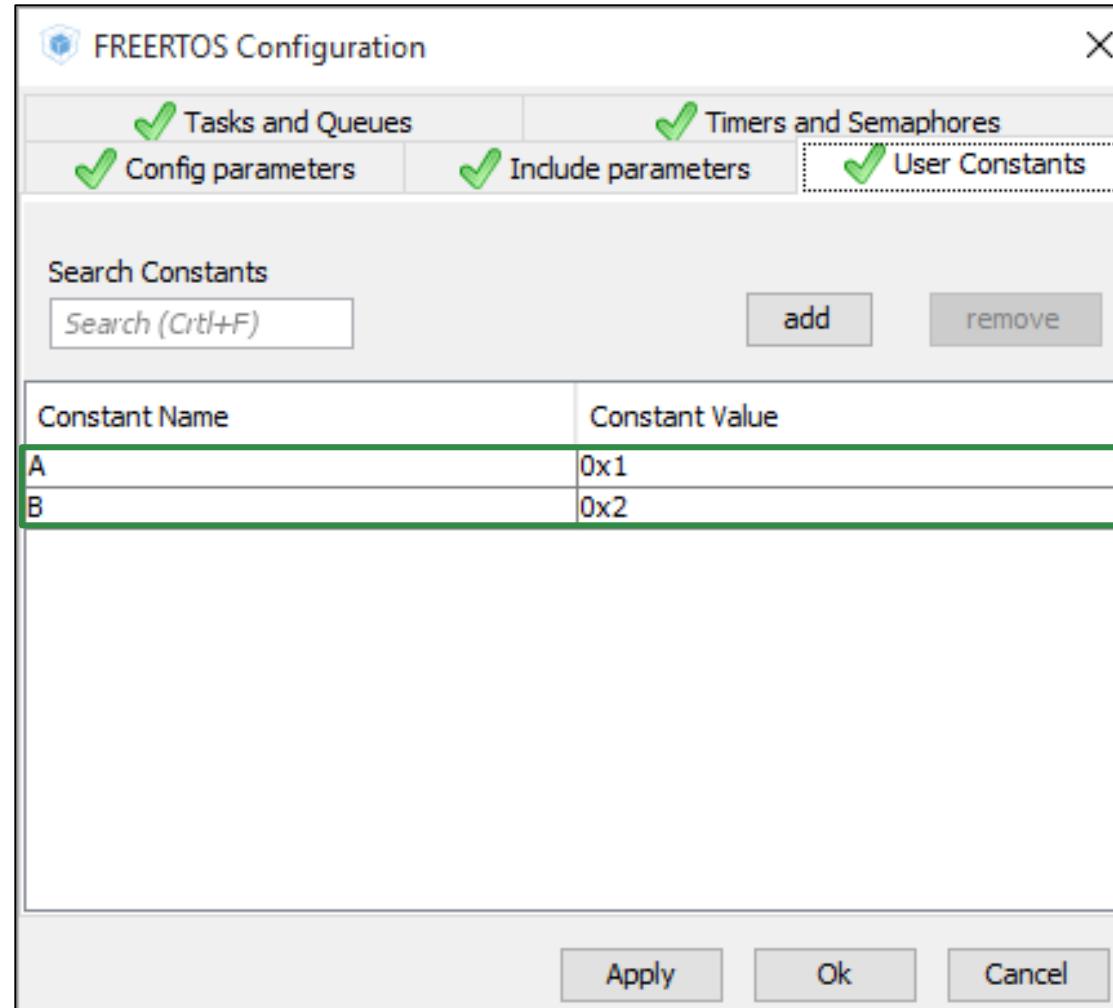
OK Cancel

B

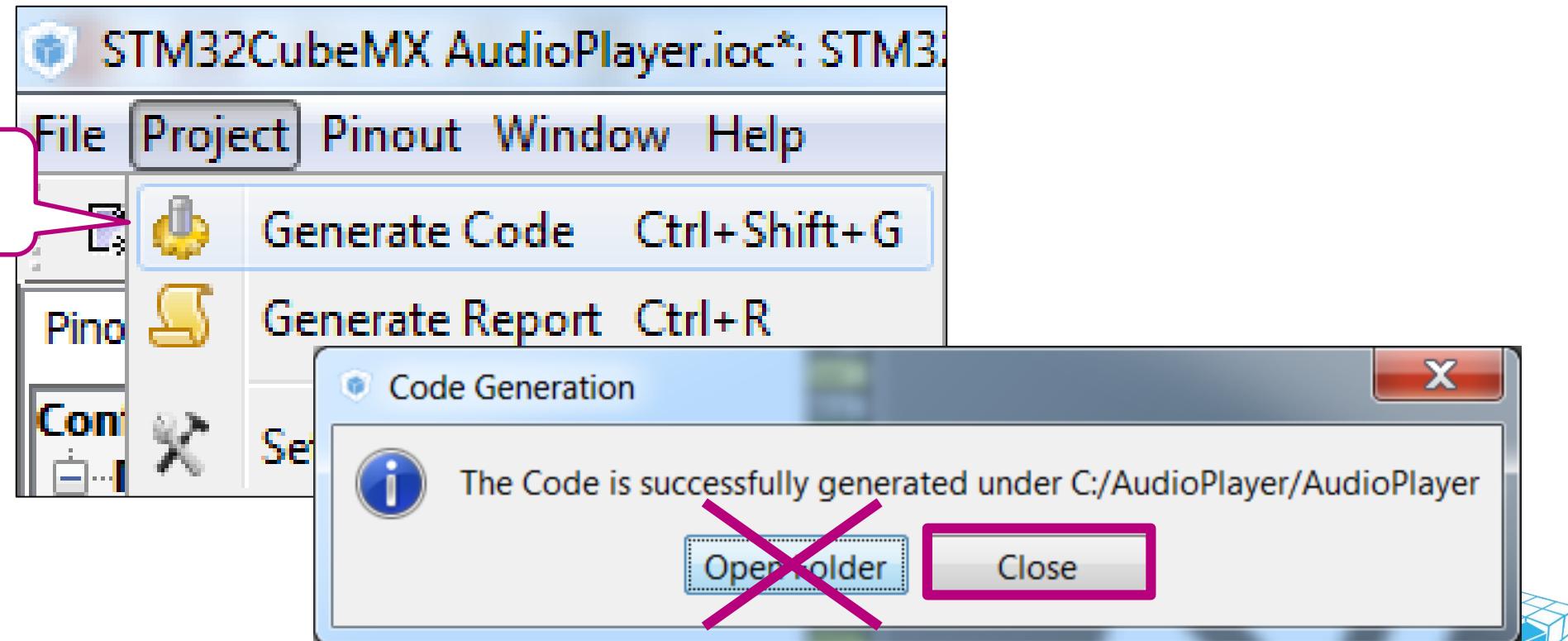
User Constants	
constant Name	B
constant Value	0x2

0x2

OK Cancel



Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP4\_Audio**  
to root folder of your project (C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Inc\

cs43l22.h

i2c.h

.\Src\

freertos.c

i2c.c

This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

- These files contains the main needed source code the user has to add to be able to work with the external audio codec (cd43l22)

- i2c.c

`AUDIO_Init(...)`

`AUDIO_DelInit(void)`

`AUDIO_ReadID(...)`

`AUDIO_Play(...)`

`AUDIO_Pause(...)`

`AUDIO_Resume(...)`

`AUDIO_Stop(...)`

`AUDIO_SetVolume(...)`

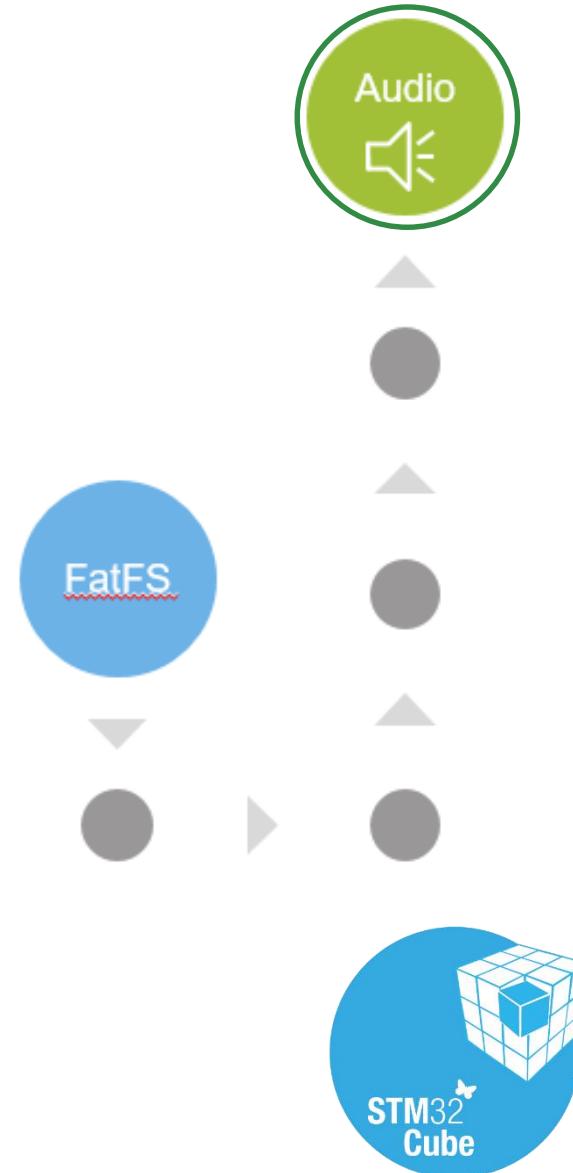
`AUDIO_SetFrequency(...)`

`AUDIO_SetMute(...)`

`AUDIO_SetOutputMode(...)`

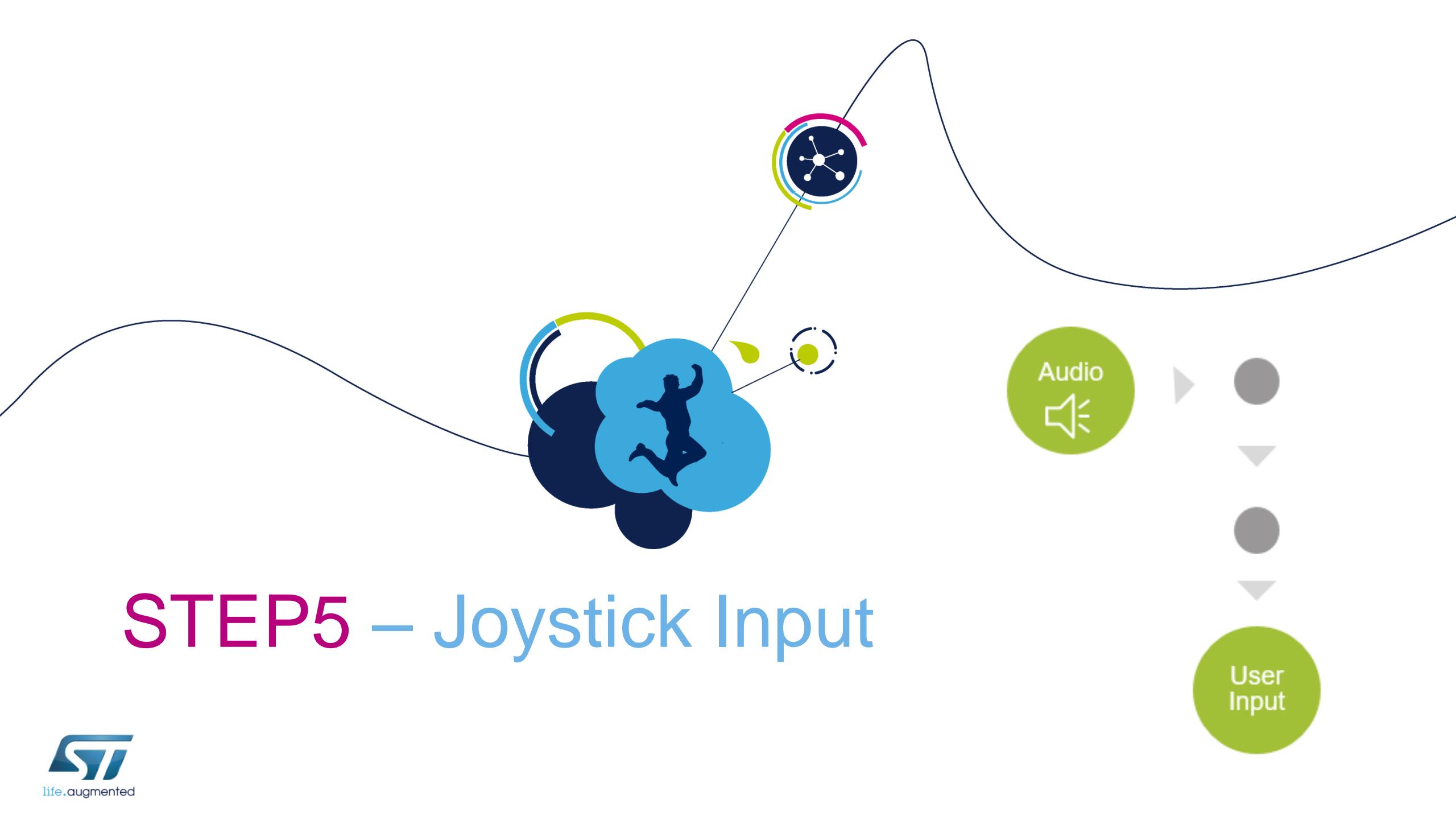
1. Open the project
2. Compile and download
3. Press the reset button to restart the application

- Connect headphones
- Do you hear music?
- LED\_GREEN being toggled when playing audio file

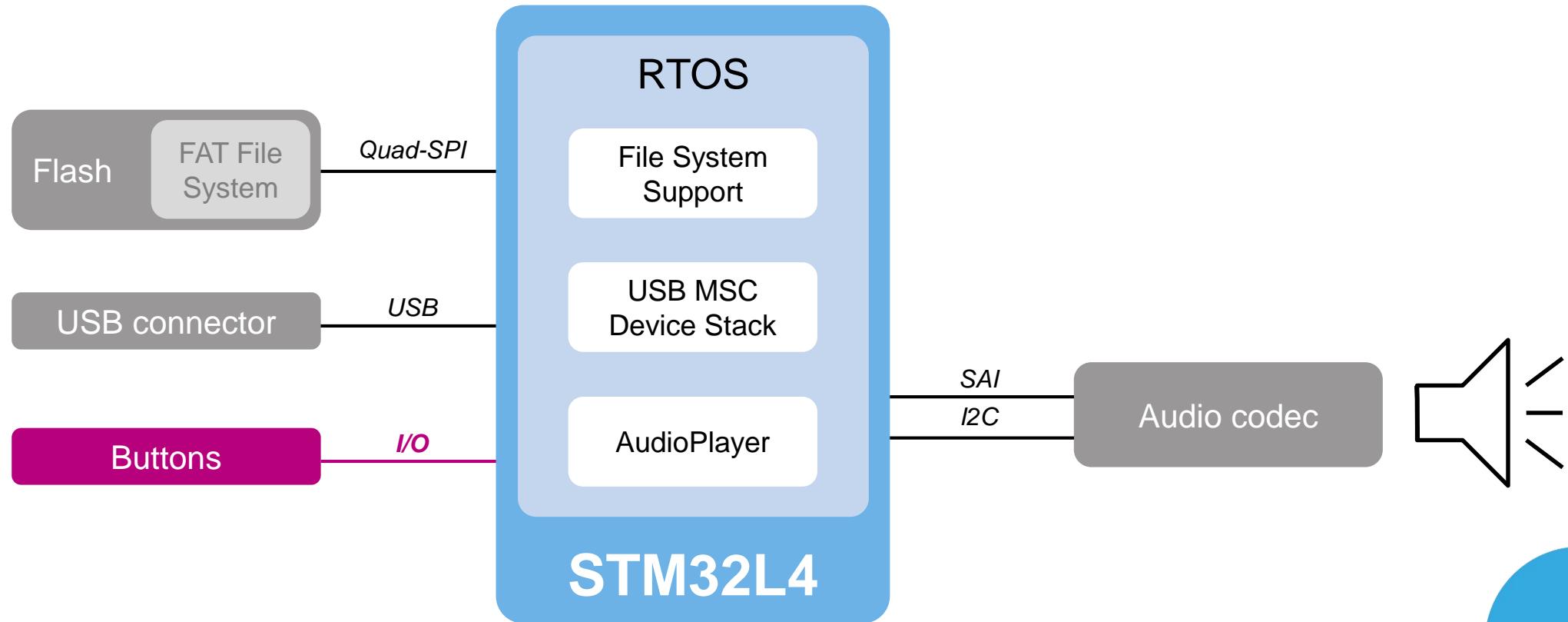


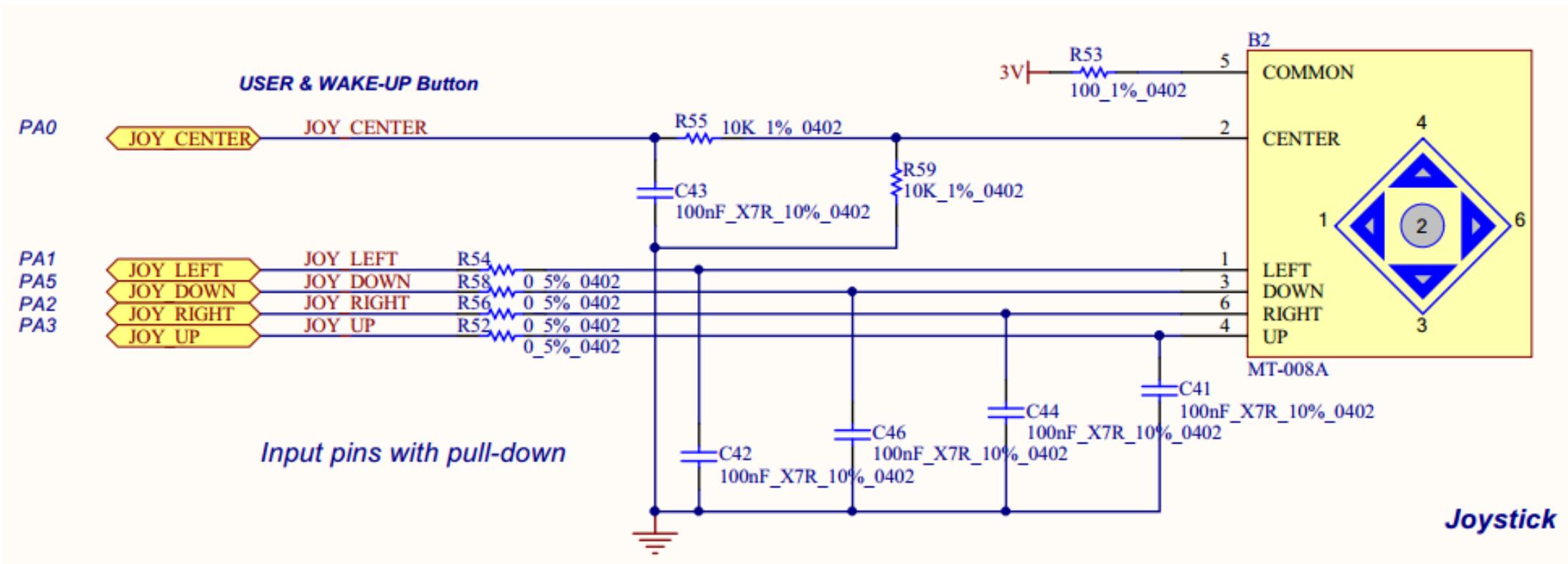
## Synchronization of the dual buffer filling mechanism:

- if both at the same time → error, we are not fast enough maybe
- “Low cost” => fast management by Scheduler, 0 memory usage



- We want to control our application state
  - like “play”, “stop”, “pause” commands

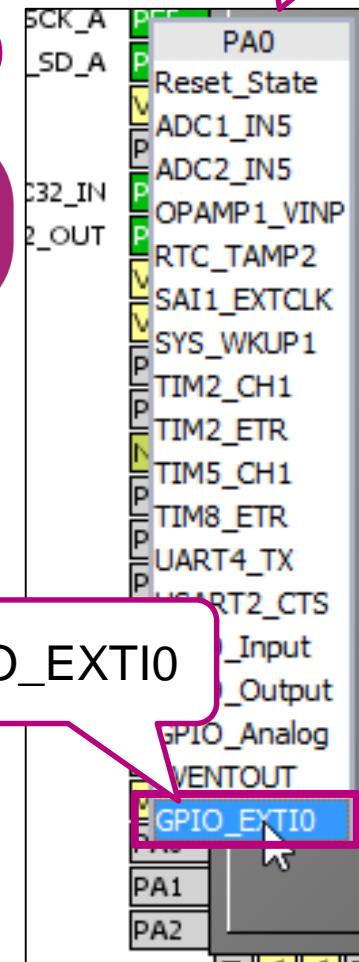




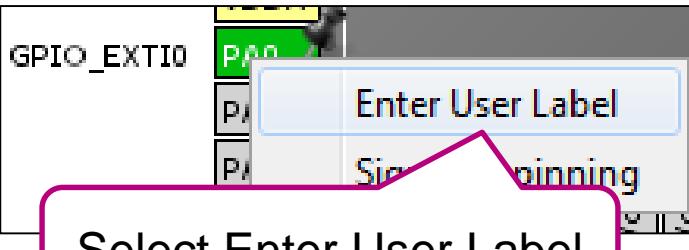
5

PA0

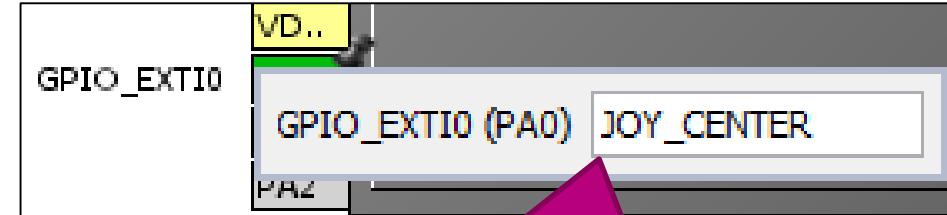
1



2

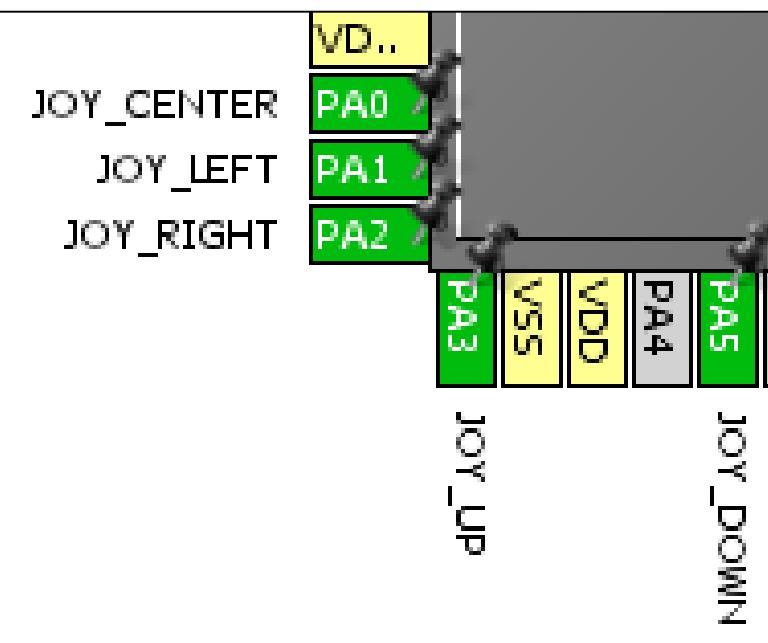
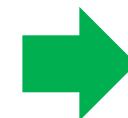


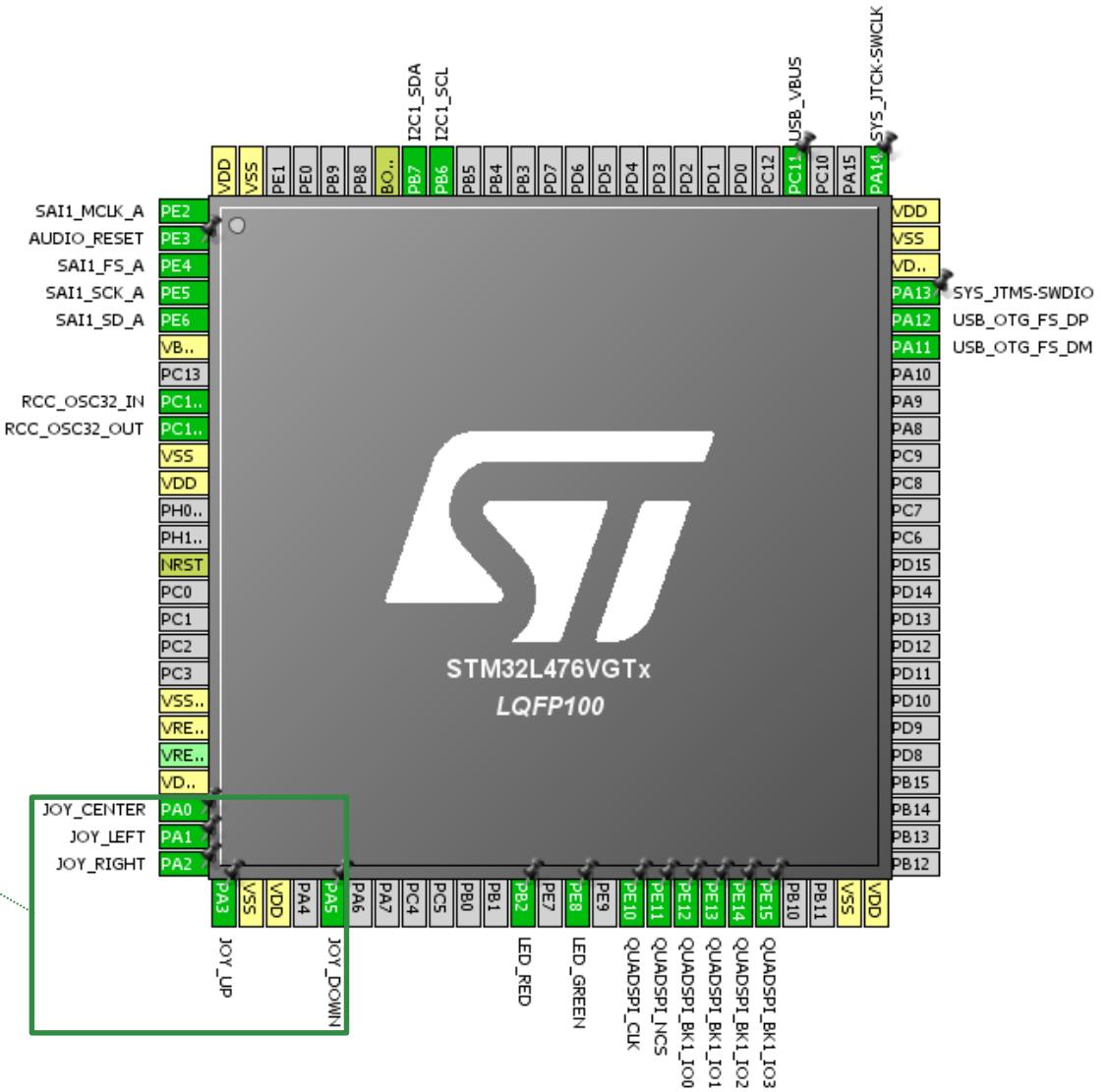
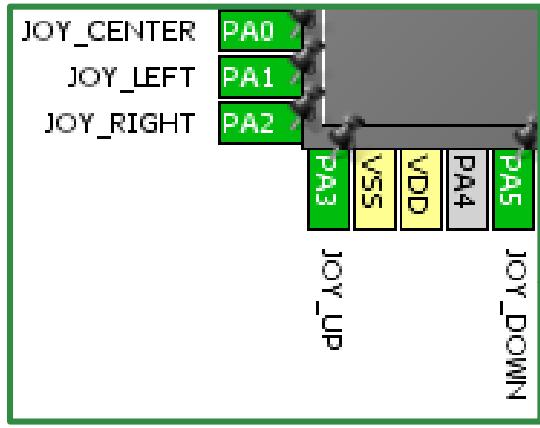
3

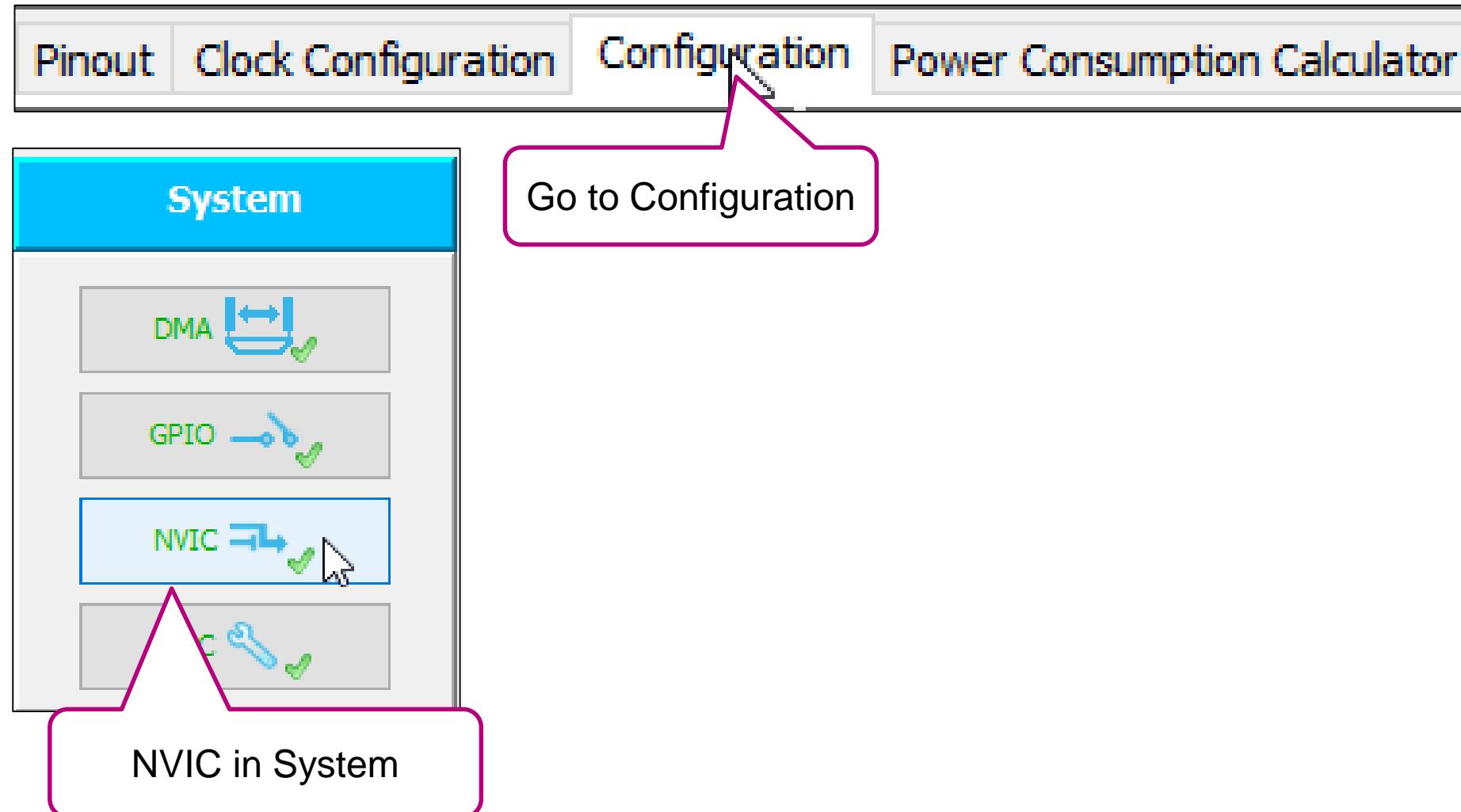


Enter JOY\_CENTER

PA0 → JOY\_CENTER  
 PA1 → JOY\_LEFT  
 PA2 → JOY\_RIGHT  
 PA3 → JOY\_UP  
 PA5 → JOY\_DOWN



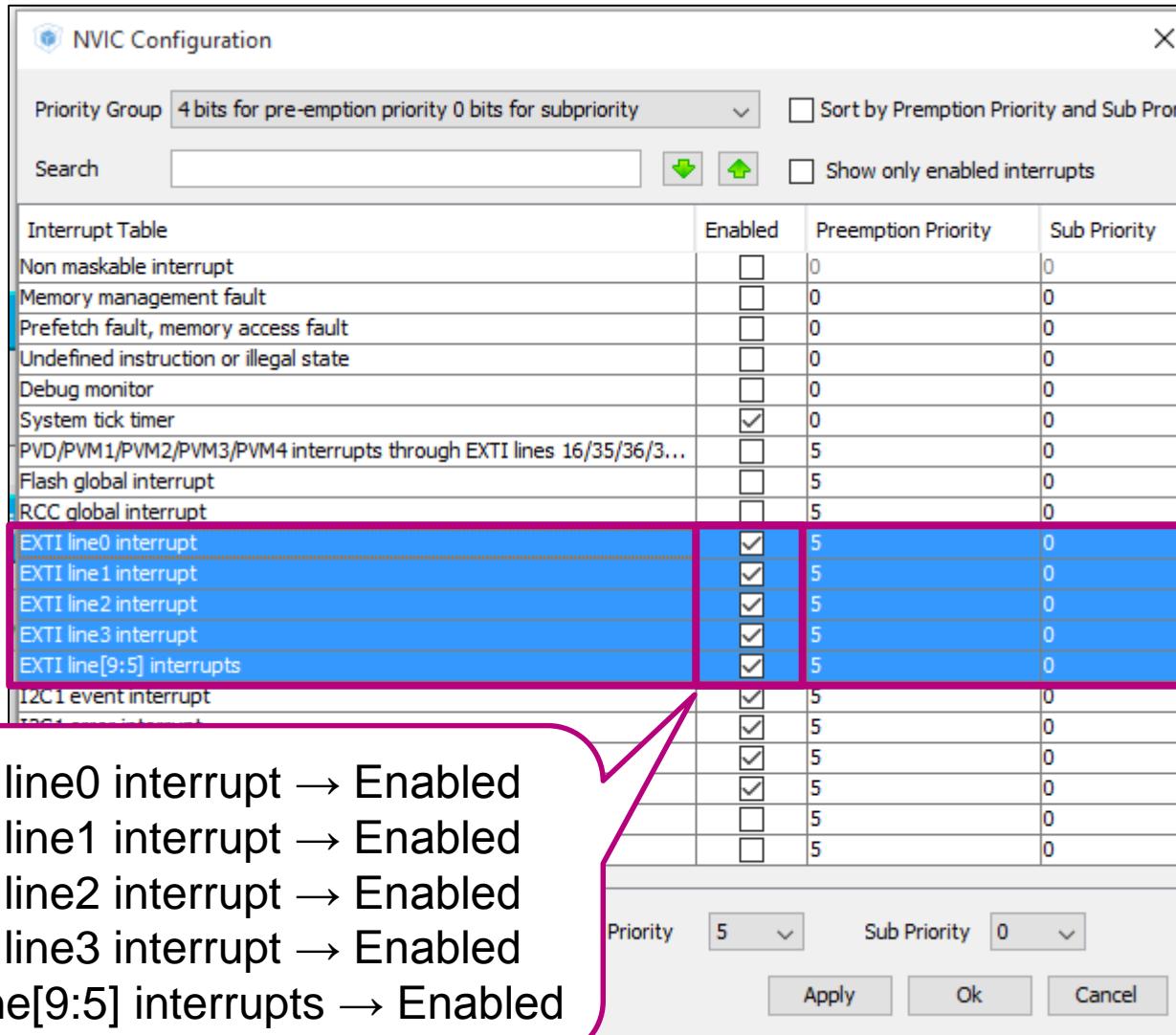




The image shows the STM32CubeMX software interface. At the top, there are tabs: Pinout, Clock Configuration, Configuration (which is selected), and Power Consumption Calculator. Below the tabs, the main window is titled 'System' and contains several configuration blocks: DMA, GPIO, NVIC, and RCC. The 'NVIC' block is highlighted with a blue border and a cursor is hovering over it. A large green arrow points from the 'NVIC' block to a detailed 'NVIC Configuration' dialog box. This dialog box has a table listing various interrupt types, their enable status, and their preemption and sub priorities. The table is as follows:

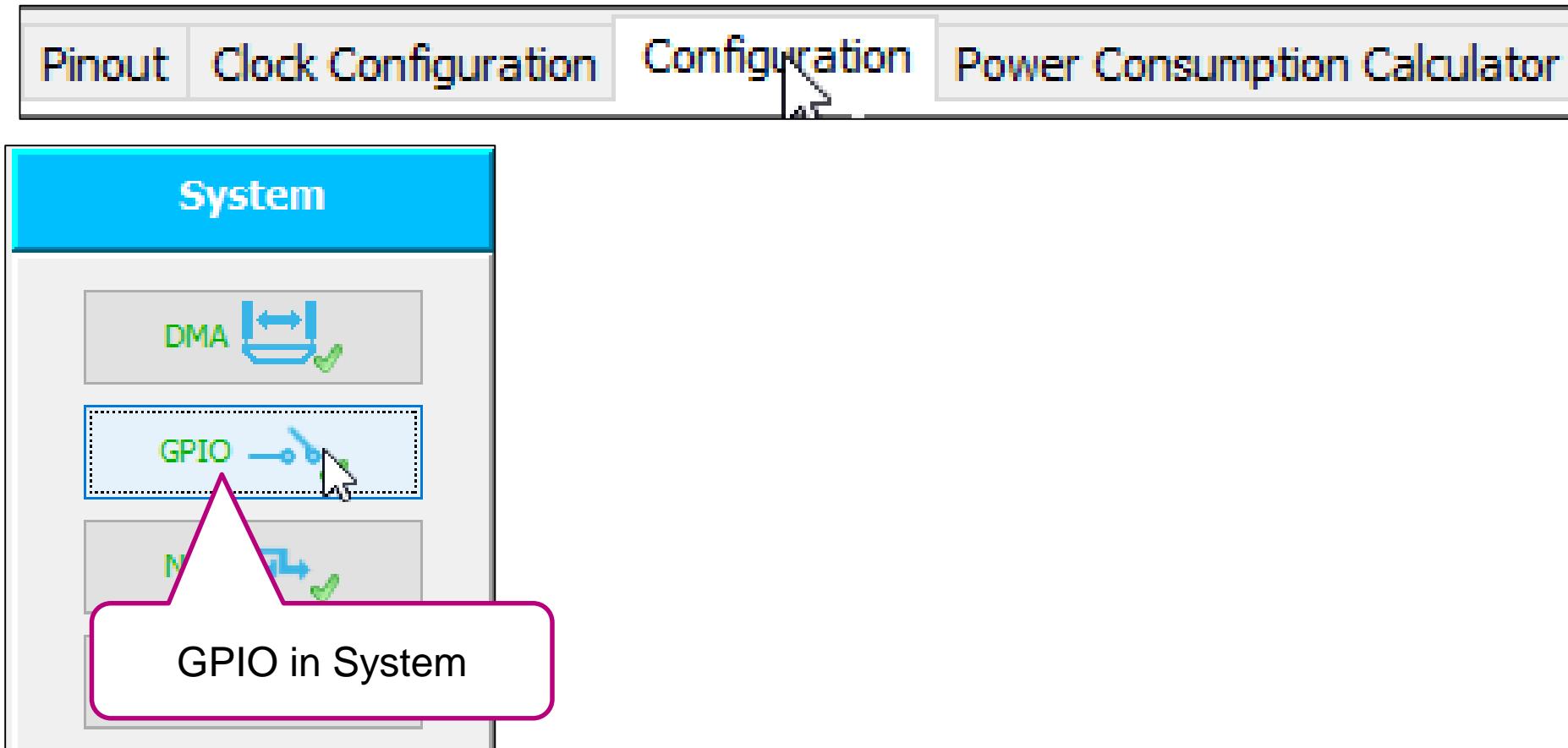
Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	<input type="checkbox"/>	0	0
Memory management fault	<input type="checkbox"/>	0	0
Prefetch fault, memory access fault	<input type="checkbox"/>	0	0
Undefined instruction or illegal state	<input type="checkbox"/>	0	0
Debug monitor	<input type="checkbox"/>	0	0
System tick timer	<input checked="" type="checkbox"/>	0	0
PVD/PVM1/PVM2/PVM3/PVM4 interrupts through EXTI lines 16/35/36/3...	<input type="checkbox"/>	5	0
Flash global interrupt	<input type="checkbox"/>	5	0
RCC global interrupt	<input type="checkbox"/>	5	0
EXTI line0 interrupt	<input type="checkbox"/>	5	0
EXTI line1 interrupt	<input type="checkbox"/>	5	0
EXTI line2 interrupt	<input type="checkbox"/>	5	0
EXTI line3 interrupt	<input type="checkbox"/>	5	0
EXTI line[9:5] interrupts	<input type="checkbox"/>	5	0
I2C1 event interrupt	<input checked="" type="checkbox"/>	5	0
I2C1 error interrupt	<input checked="" type="checkbox"/>	5	0
DMA2 channel1 global interrupt	<input checked="" type="checkbox"/>	5	0
USB OTG FS global interrupt	<input checked="" type="checkbox"/>	5	0
QUADSPI global interrupt	<input type="checkbox"/>	5	0
SAI1 global interrupt	<input type="checkbox"/>	5	0

At the bottom of the dialog box are buttons for Enabled, Preemption Priority, Sub Priority, Apply, Ok, and Cancel.



Interrupt Table	Enabled	Preemption Priority	Sub Priority
Non maskable interrupt	<input type="checkbox"/>	0	0
Memory management fault	<input type="checkbox"/>	0	0
Prefetch fault, memory access fault	<input type="checkbox"/>	0	0
Undefined instruction or illegal state	<input type="checkbox"/>	0	0
Debug monitor	<input type="checkbox"/>	0	0
System tick timer	<input checked="" type="checkbox"/>	0	0
PVD/PVM1/PVM2/PVM3/PVM4 interrupts through EXTI lines 16/35/36/3...	<input type="checkbox"/>	5	0
Flash global interrupt	<input type="checkbox"/>	5	0
RCC global interrupt	<input type="checkbox"/>	5	0
EXTI line0 interrupt	<input checked="" type="checkbox"/>	5	0
EXTI line1 interrupt	<input checked="" type="checkbox"/>	5	0
EXTI line2 interrupt	<input checked="" type="checkbox"/>	5	0
EXTI line3 interrupt	<input checked="" type="checkbox"/>	5	0
EXTI line[9:5] interrupts	<input checked="" type="checkbox"/>	5	0
I2C1 event interrupt	<input checked="" type="checkbox"/>	5	0
...	<input checked="" type="checkbox"/>	5	0
	<input checked="" type="checkbox"/>	5	0
	<input checked="" type="checkbox"/>	5	0
	<input checked="" type="checkbox"/>	5	0
	<input type="checkbox"/>	5	0
	<input type="checkbox"/>	5	0

EXTI line0 interrupt → Enabled  
EXTI line1 interrupt → Enabled  
EXTI line2 interrupt → Enabled  
EXTI line3 interrupt → Enabled  
EXTI line[9:5] interrupts → Enabled



The image shows the STM32CubeMX software interface. On the left, the 'System' tab is selected, displaying icons for DMA, GPIO, NVIC, and RCC. A green arrow points from the GPIO icon to the 'Pin Configuration' dialog box on the right. The 'Pin Configuration' dialog box has tabs for GPIO, I2C1, QUADSPI, RCC, SAI1, SYS, and USB\_OTG\_FS. The GPIO tab is selected. It includes a search bar and a table of pin configurations. The table data is as follows:

Pin Name	Signal on Pin	GPIO mode	GPIO Pull-up...	Maximum out...	Fast Mode	User Label	Modified
PA0	n/a	External Interr...	No pull-up and...	n/a	n/a	JOY_CENTER.	<input checked="" type="checkbox"/>
PA1	n/a	External Interr...	No pull-up and...	n/a	n/a	JOY_LEFT	<input checked="" type="checkbox"/>
PA2	n/a	External Interr...	No pull-up and...	n/a	n/a	JOY_RIGHT	<input checked="" type="checkbox"/>
PA3	n/a	External Interr...	No pull-up and...	n/a	n/a	JOY_UP	<input checked="" type="checkbox"/>
PA5	n/a	External Interr...	No pull-up and...	n/a	n/a	JOY_DOWN	<input checked="" type="checkbox"/>
PB2	n/a	Output Push Pull	No pull-up and...	Low	n/a	LED_RED	<input checked="" type="checkbox"/>
PC11	n/a	Input mode	No pull-up and...	n/a	n/a	USB_VBUS	<input checked="" type="checkbox"/>
PE3	n/a	Output Push Pull	No pull-up and...	Low	n/a	AUDIO_RESET	<input checked="" type="checkbox"/>
PE8	n/a	Output Push Pull	No pull-up and...	Low	n/a	LED_GREEN	<input checked="" type="checkbox"/>

At the bottom of the dialog box, there is a note: "Select Pins from table to configure them. **Multiple selection is Allowed.**" and checkboxes for "Group By IP" and "Apply", "Ok", and "Cancel".

Pin Configuration

GPIO I2C1 QUADSPI RCC SAI1 SYS USB\_OTG\_FS

Search Signals

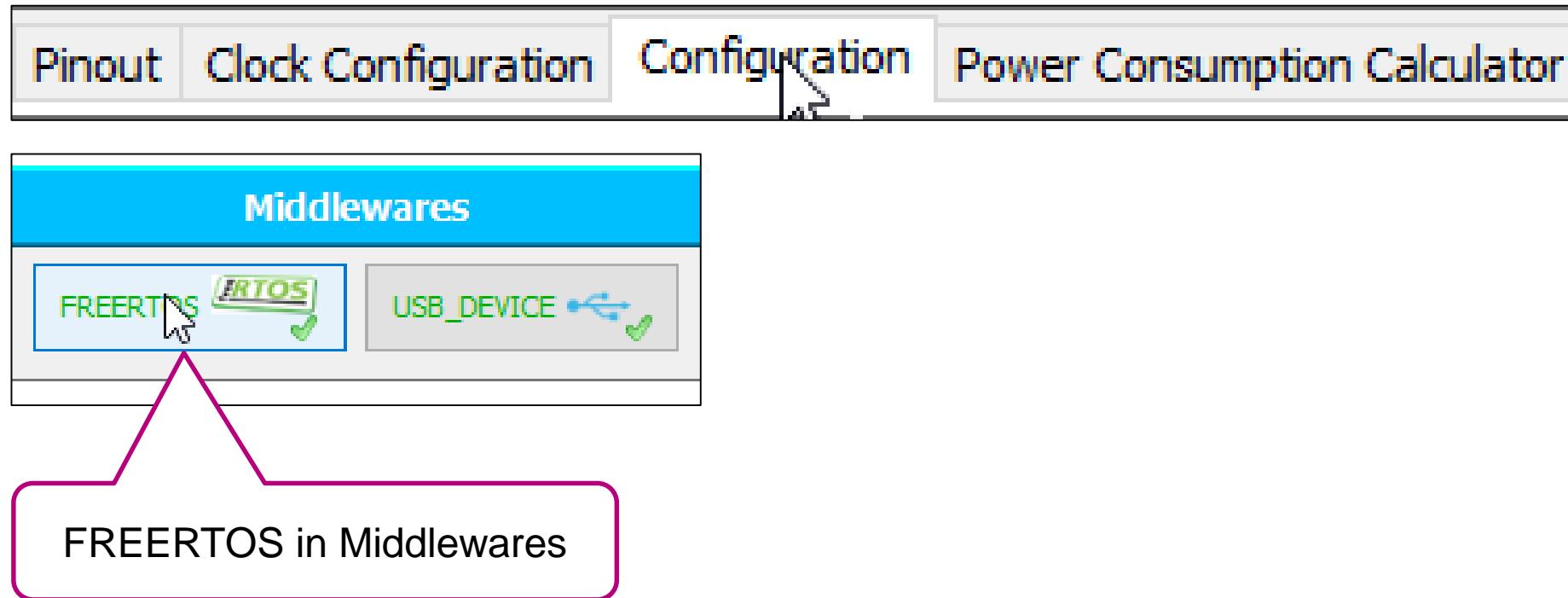
Pin Name	Signal on Pin	GPIO mode	GPIO Pull-u...	Maximum o...	Fast Mode	User Label	Modified
PA0	n/a	External Int...	Pull-down	n/a	n/a	JOY_CENTER	<input checked="" type="checkbox"/>
PA1	n/a	External Int...	Pull-down	n/a	n/a	JOY_LEFT	<input checked="" type="checkbox"/>
PA2	n/a	External Int...	Pull-down	n/a	n/a	JOY_RIGHT	<input checked="" type="checkbox"/>
PA3	n/a	External Int...	Pull-down	n/a	n/a	JOY_UP	<input checked="" type="checkbox"/>
PA5	n/a	External Int...	Pull-down	n/a	n/a	JOY_DOWN	<input checked="" type="checkbox"/>
PB2	n/a	Output Push...	No pull-up a...	Low	n/a	LED_RED	<input checked="" type="checkbox"/>
			No pull-up a...	n/a	n/a	USB_VBUS	<input checked="" type="checkbox"/>
			No pull-up a...	Low	n/a	AUDIO_RESET	<input checked="" type="checkbox"/>
			No pull-up a...	Low	n/a	LED_GREEN	<input checked="" type="checkbox"/>

GPIO Pull-Up/Pull-Down

PA0 → Pull-down  
PA1 → Pull-down  
PA2 → Pull-down  
PA3 → Pull-down  
PA5 → Pull-down

User Label: JOY\_CENTER

Group By IP

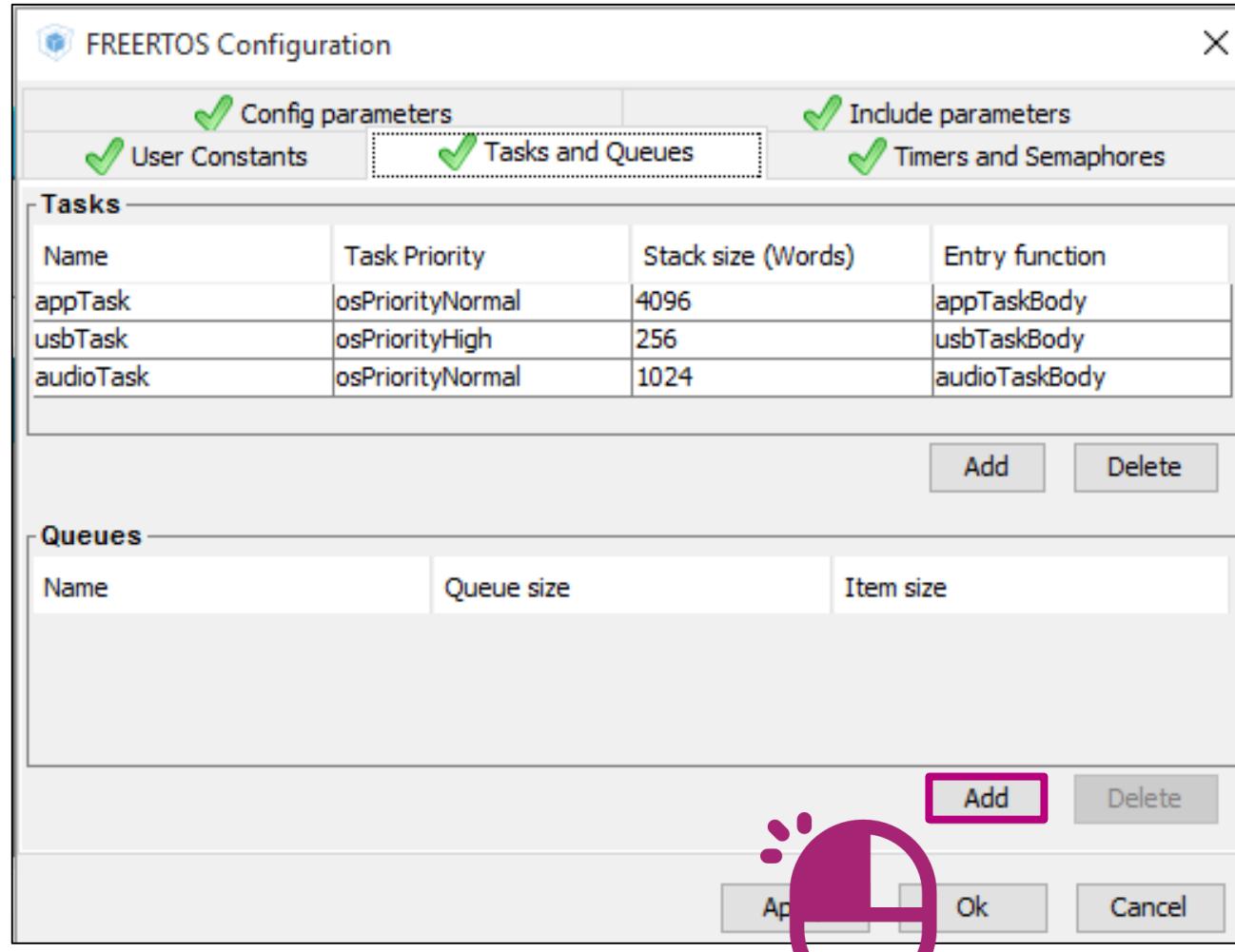


The screenshot shows the STM32CubeMX software interface. The top navigation bar includes tabs for Pinout, Clock Configuration, Configuration (which is currently selected), and Power Consumption Calculator. Below the navigation bar, the main window displays the 'Middlewares' section. This section contains two items: 'FREERTOS' and 'USB\_DEVICE'. The 'FREERTOS' item is highlighted with a blue border and a cursor icon pointing to it. A pink callout box with a triangular pointer is positioned below the 'FREERTOS' item, containing the text 'FREERTOS in Middlewares'.

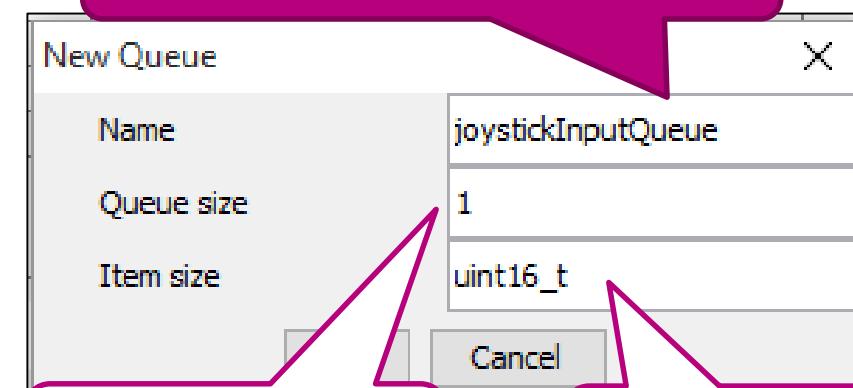
The screenshot shows the STM32CubeMX software interface. The top navigation bar includes tabs for Pinout, Clock Configuration, Configuration (which is selected), and Power Consumption Calculator. Below the navigation bar is a 'Middlewares' section with a 'FREERTOS' tab and a 'USB\_DEVICE' tab. A large green arrow points from the 'FREERTOS' tab to a detailed view of the 'FREERTOS Configuration' dialog box. This dialog box contains sections for Config parameters, User Constants, Tasks, and Queues. The 'Tasks' section lists three tasks with their respective priorities and stack sizes:

Name	Task Priority	Stack size (Words)	Entry function
appTask	osPriorityNormal	4096	appTaskBody
usbTask	osPriorityHigh	256	usbTaskBody
audioTask	osPriorityNormal	1024	audioTaskBody

The 'Queues' section is currently empty. At the bottom of the dialog box are buttons for Add, Delete, Apply, Ok, and Cancel.

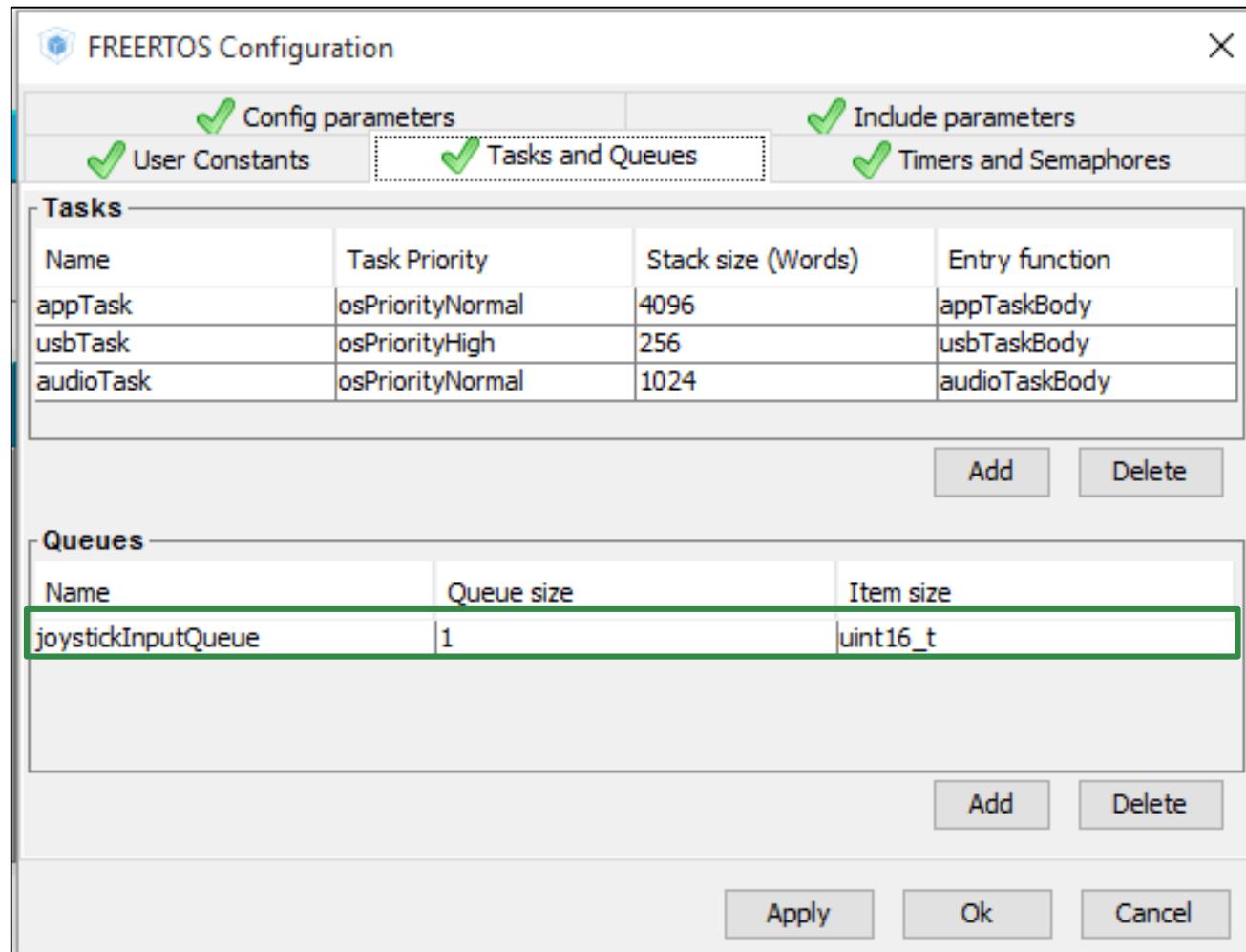


Name → joystickInputQueue

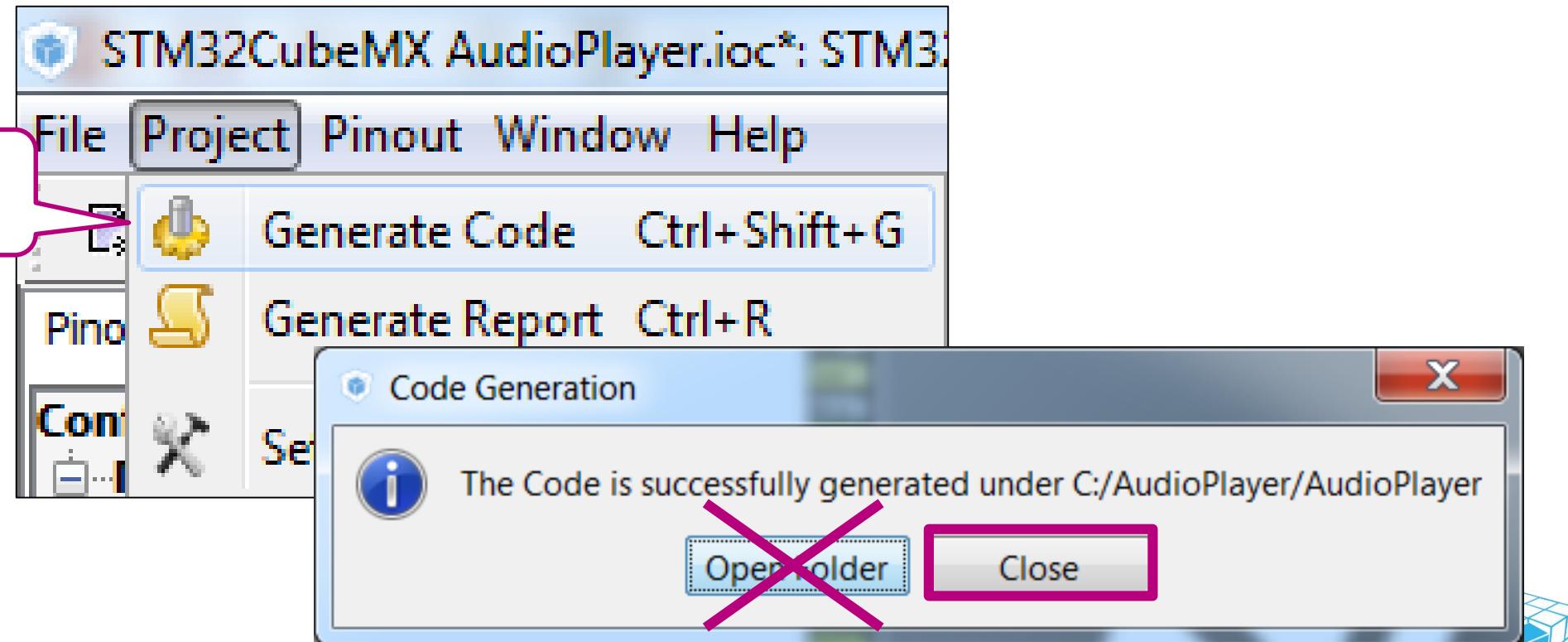


Queue size → 1

Item size  
→ uint16\_t



Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP5\_ControlInput**  
to root folder of your project

(C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Src\

freertos.c

This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

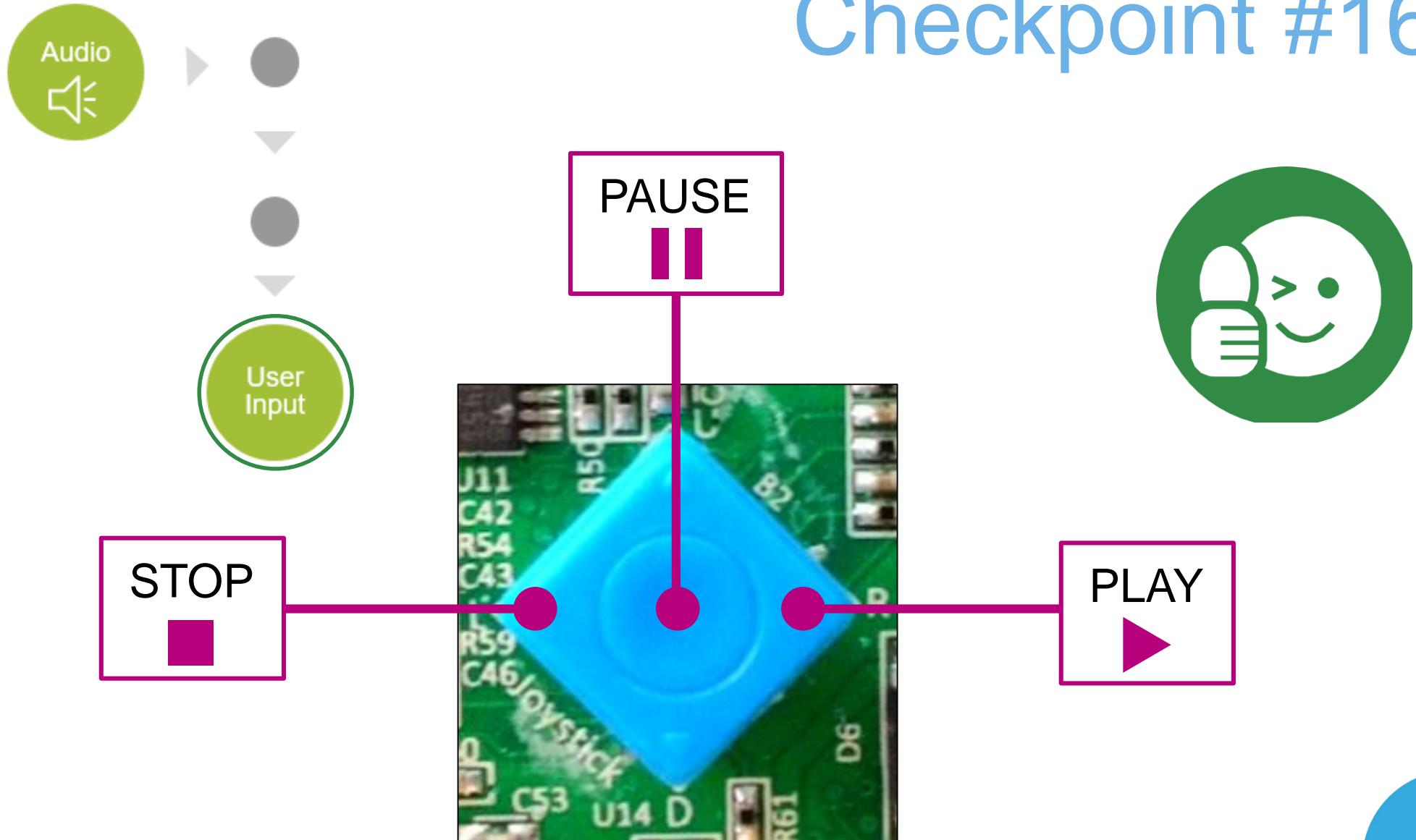
- This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

1. Open the project
2. Compile and download
3. Press the reset button on discovery once download has finished

5

# Checkpoint #16

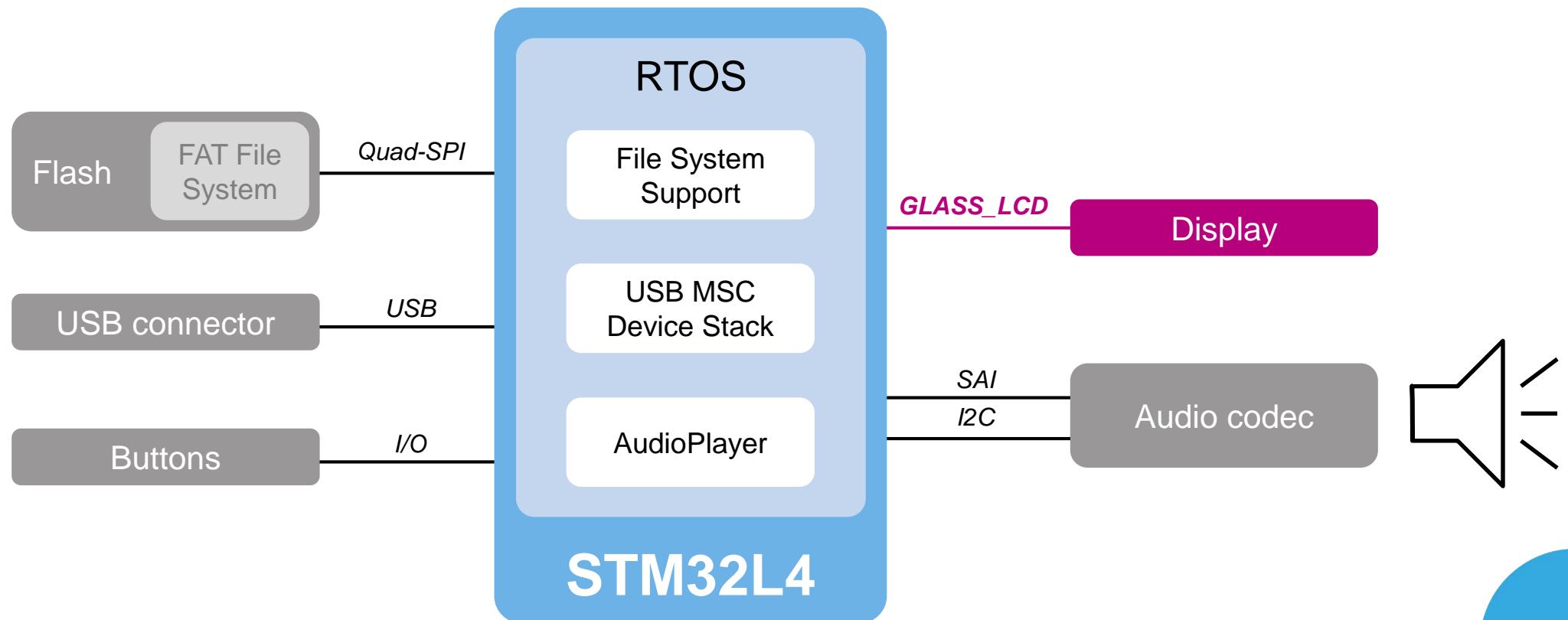
157



- Because we want to process just one input event at a time
- We are passing the command (queue item) from the EXTI ISR
- Why not signal?
  - Multiple signals can be set at a time



- We want to see the actual application state
  - like “PLAYING”, “STOPPED”, “PAUSED”

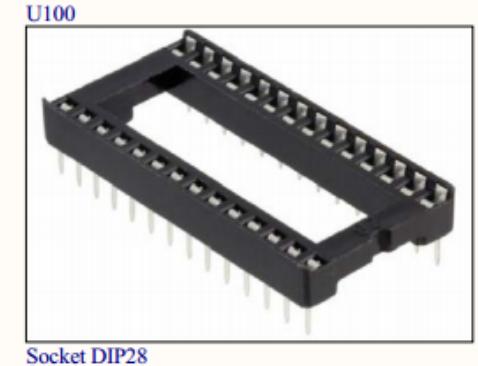
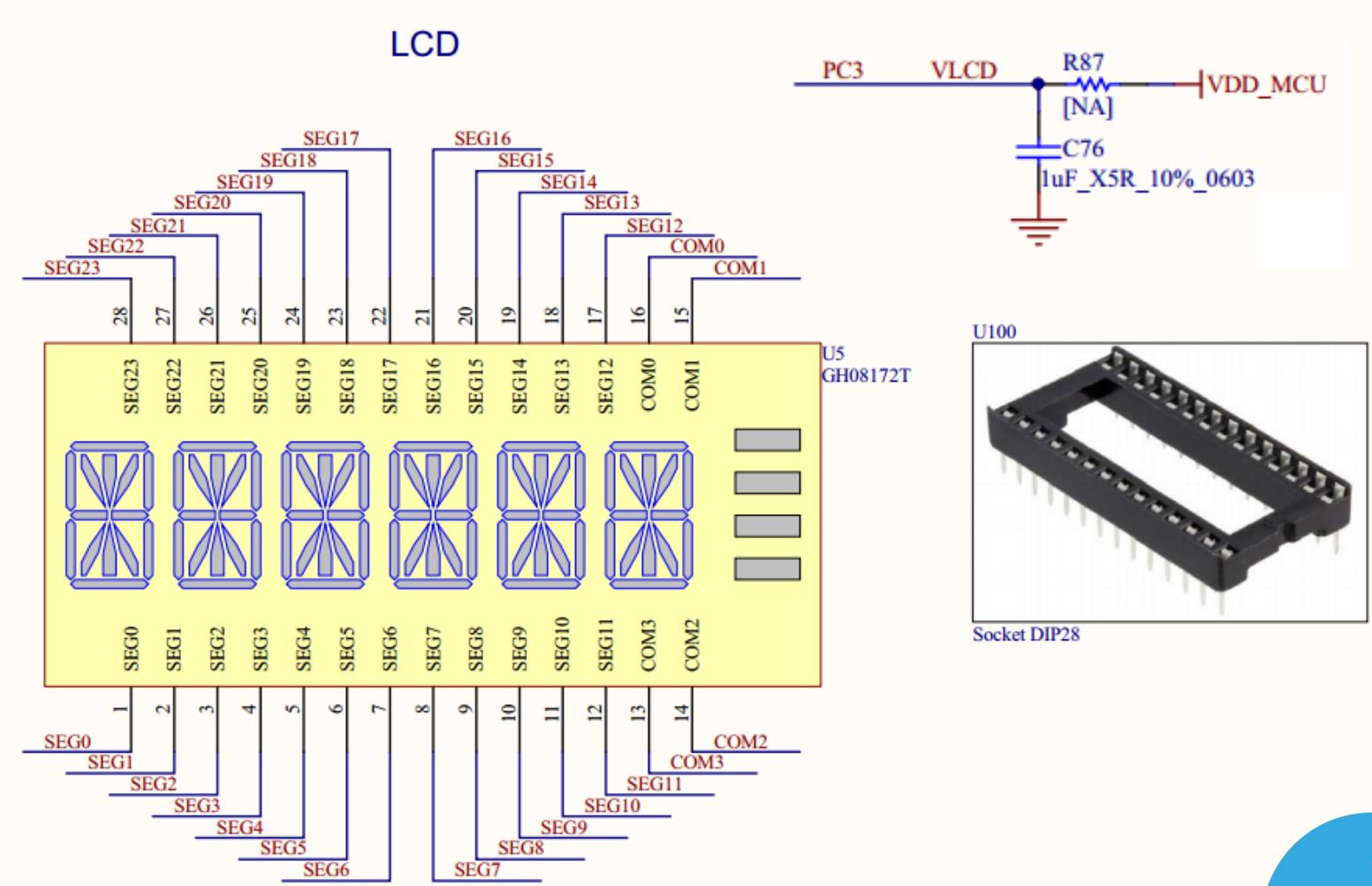


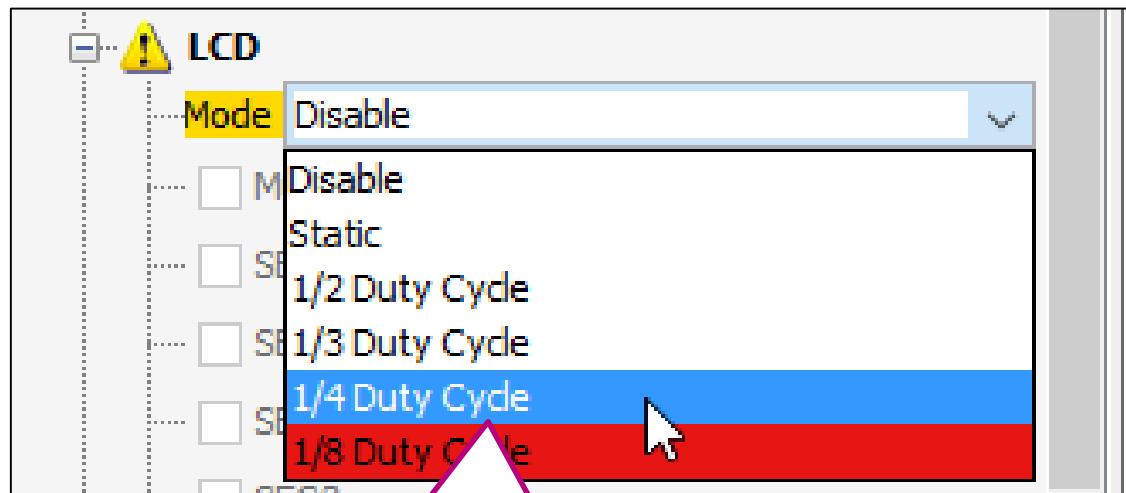
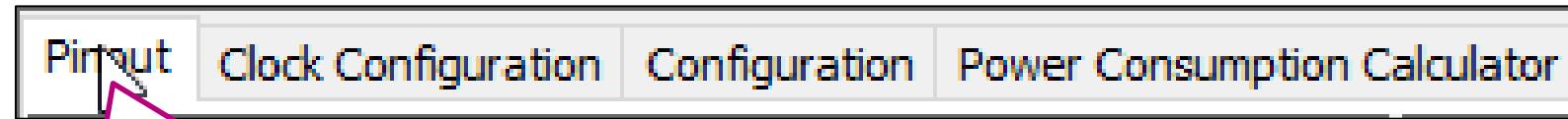


PA8	COM0
PA9	COM1
PA10	COM2
PB9	COM3

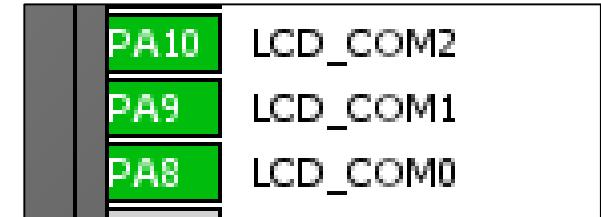
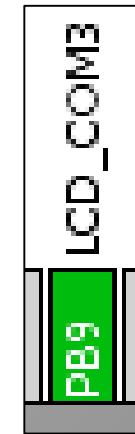
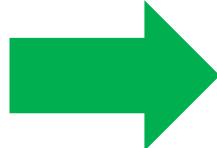
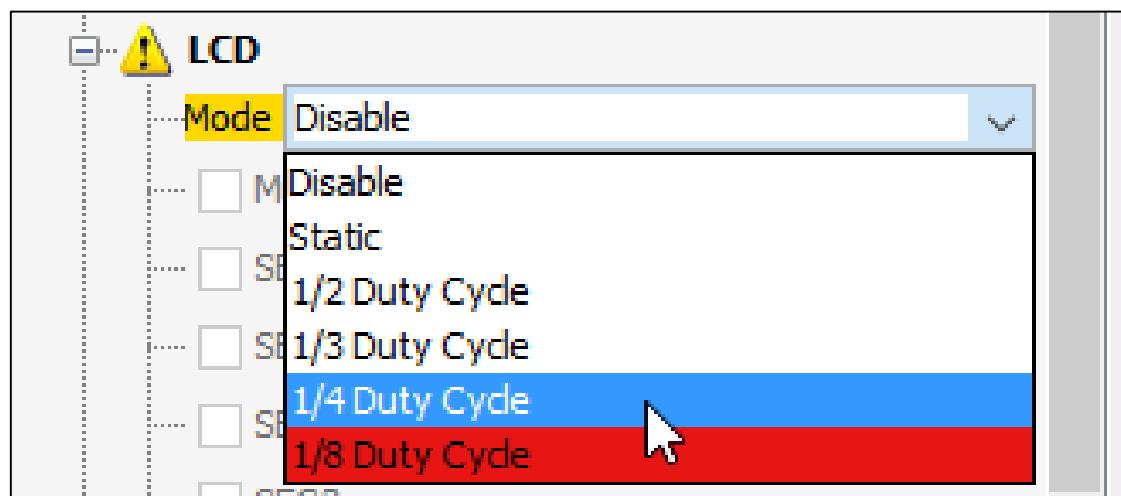
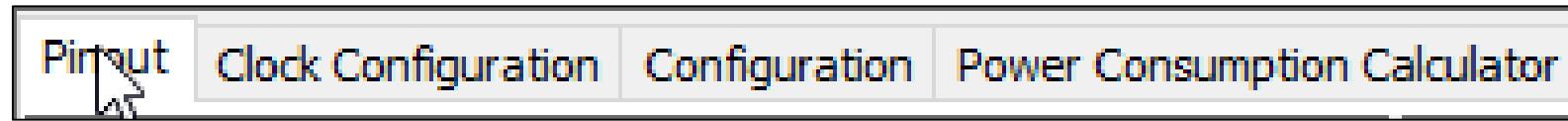
PA7	SEG0
PC5	SEG1
PB1	SEG2
PB13	SEG3
PB15	SEG4
PD9	SEG5
PD11	SEG6
PD13	SEG7
PD15	SEG8
PC7	SEG9
PA15	SEG10
PB4	SEG11
PB5	SEG12

PC8	SEG13
PC6	SEG14
PD14	SEG15
PD12	SEG16
PD10	SEG17
PD8	SEG18
PB14	SEG19
PB12	SEG20
PB0	SEG21
PC4	SEG22
PA6	SEG23

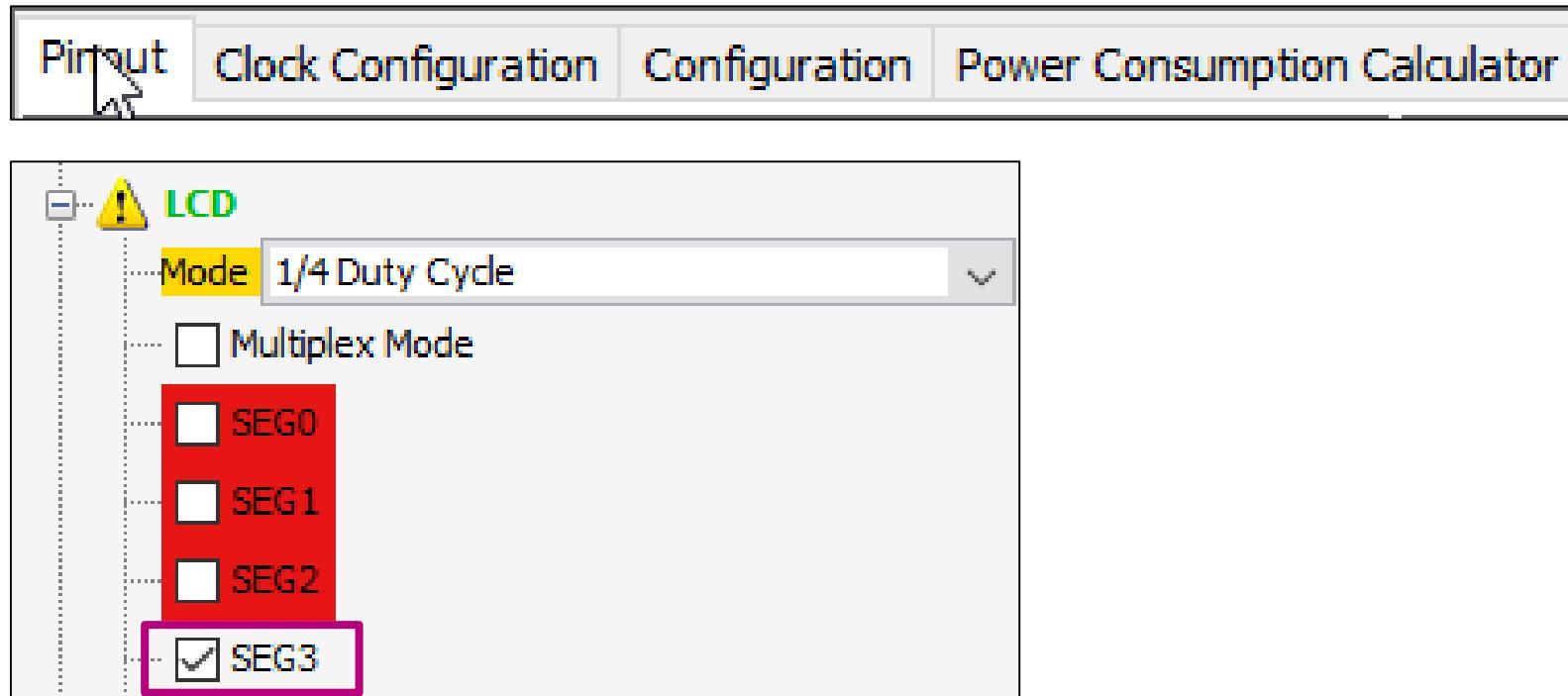




Under LCD peripheral select  
**1/4 Duty Cycle** mode



The corresponding pins are assigned and configured automatically!



Pinsout Clock Configuration Configuration Power Consumption Calculator

**LCD**

Mode **1/4 Duty Cycle**

Multiplex Mode

SEG0

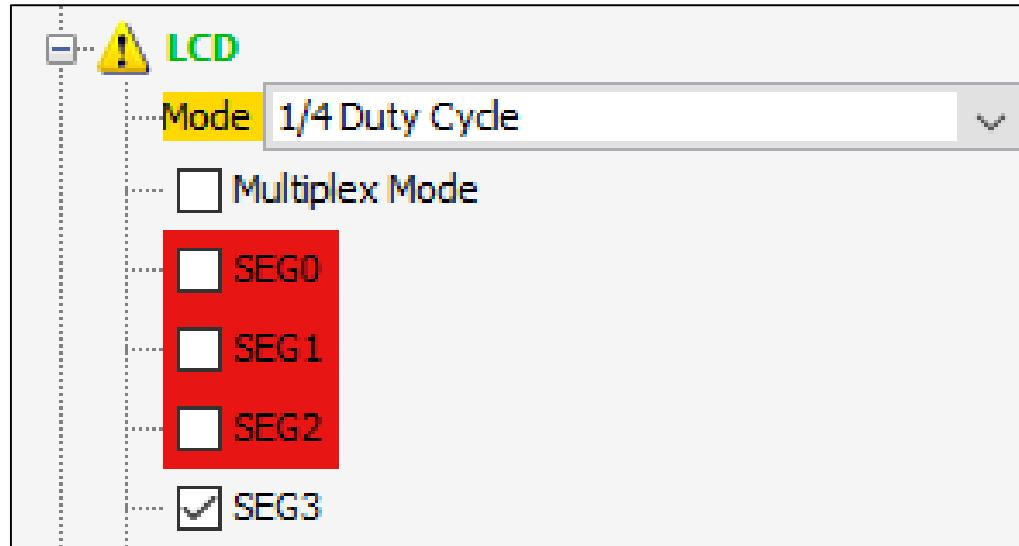
SEG1

SEG2

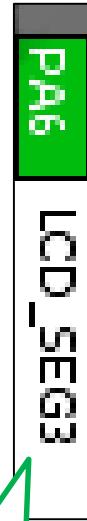
SEG3

x	3	4	5	6	8	9	12	13	14	15	17	22
	23	24	25	26	28	29	30	31	32	33	34	35

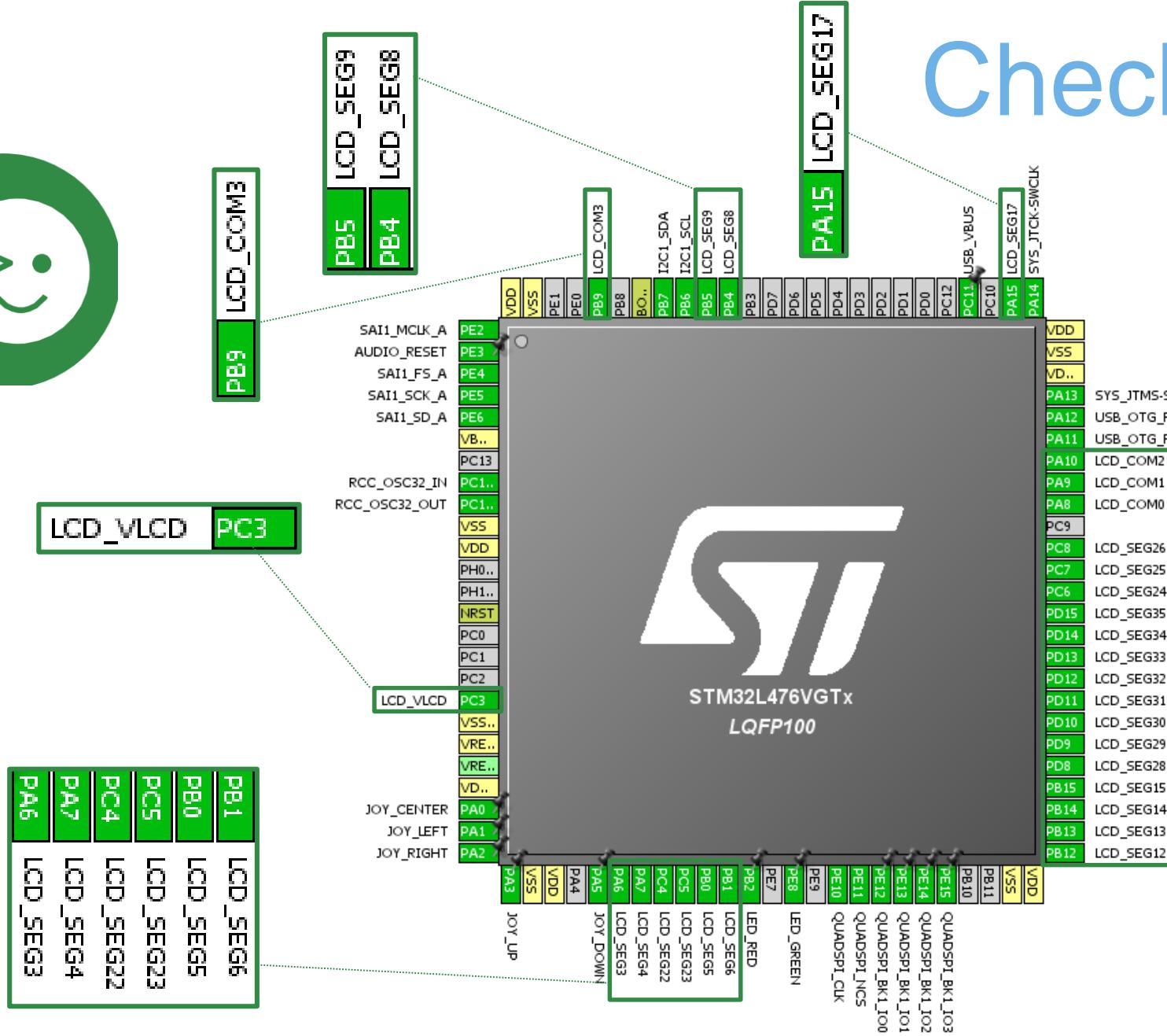
Under LCD peripheral check all the above **SEGx**



x	3	4	5	6	8	9	12	13	14	15	17	22
	23	24	25	26	28	29	30	31	32	33	34	35

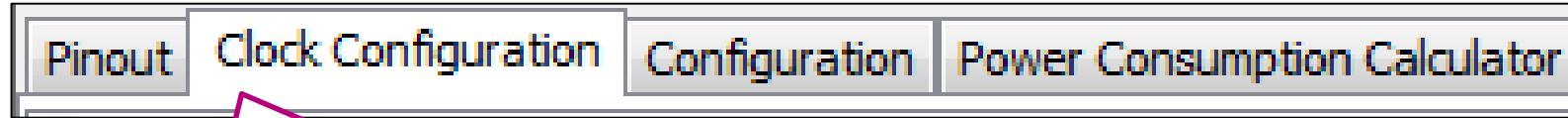


The corresponding pins are assigned and configured automatically!

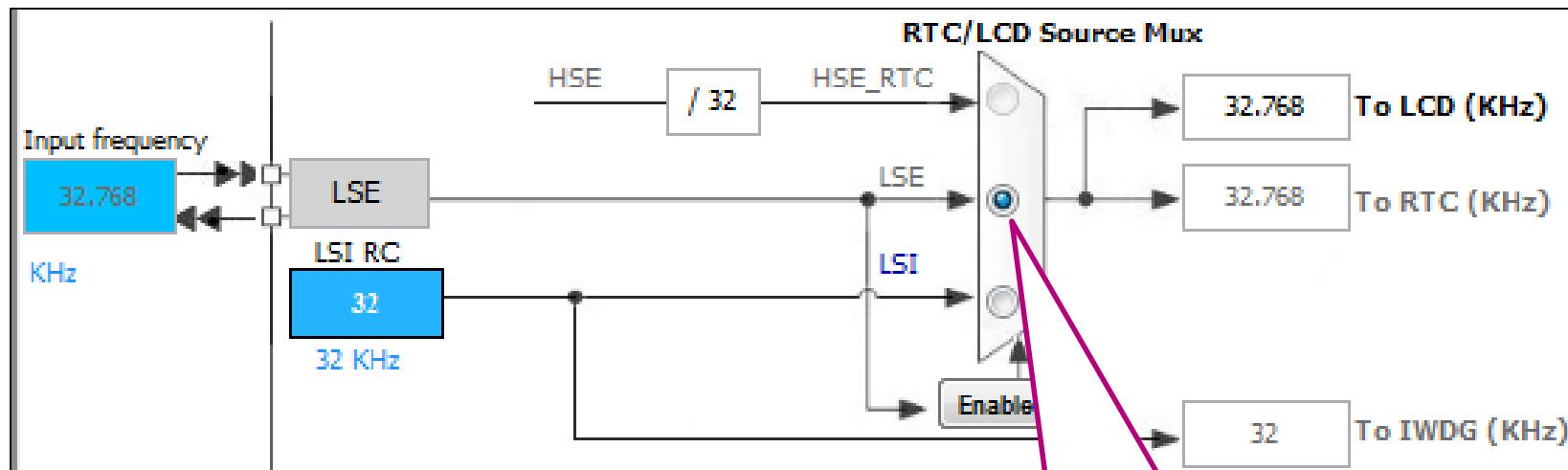


PA10	LCD_COM2
PA9	LCD_COM1
PA8	LCD_COM0
PC9	
PC8	LCD_SEG26
PC7	LCD_SEG25
PC6	LCD_SEG24
PD15	LCD_SEG35
PD14	LCD_SEG34
PD13	LCD_SEG33
PD12	LCD_SEG32
PD11	LCD_SEG31
PD10	LCD_SEG30
PD9	LCD_SEG29
PD8	LCD_SEG28
PB15	LCD_SEG15
PB14	LCD_SEG14
PB13	LCD_SEG13
PB12	LCD_SEG12

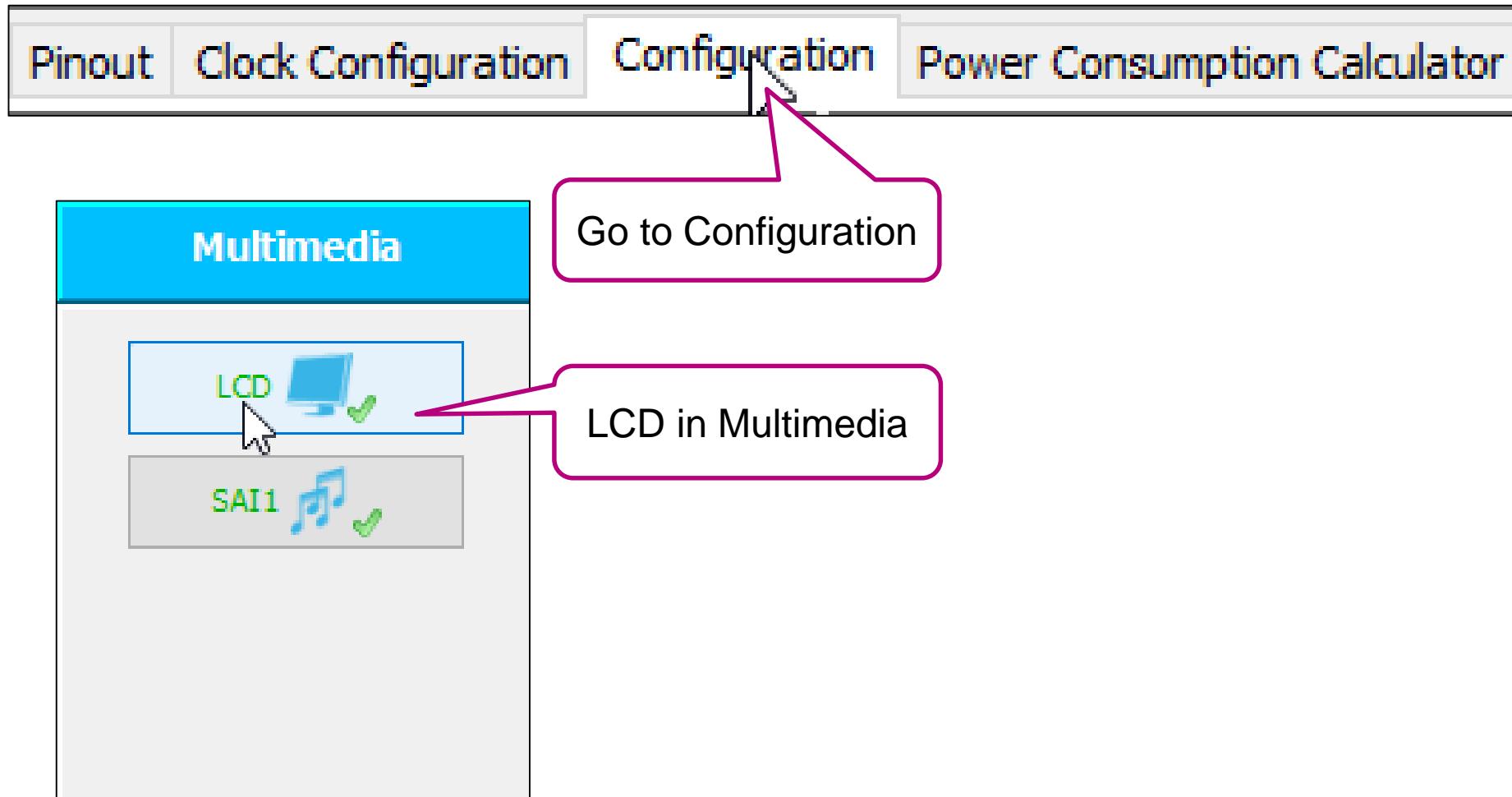


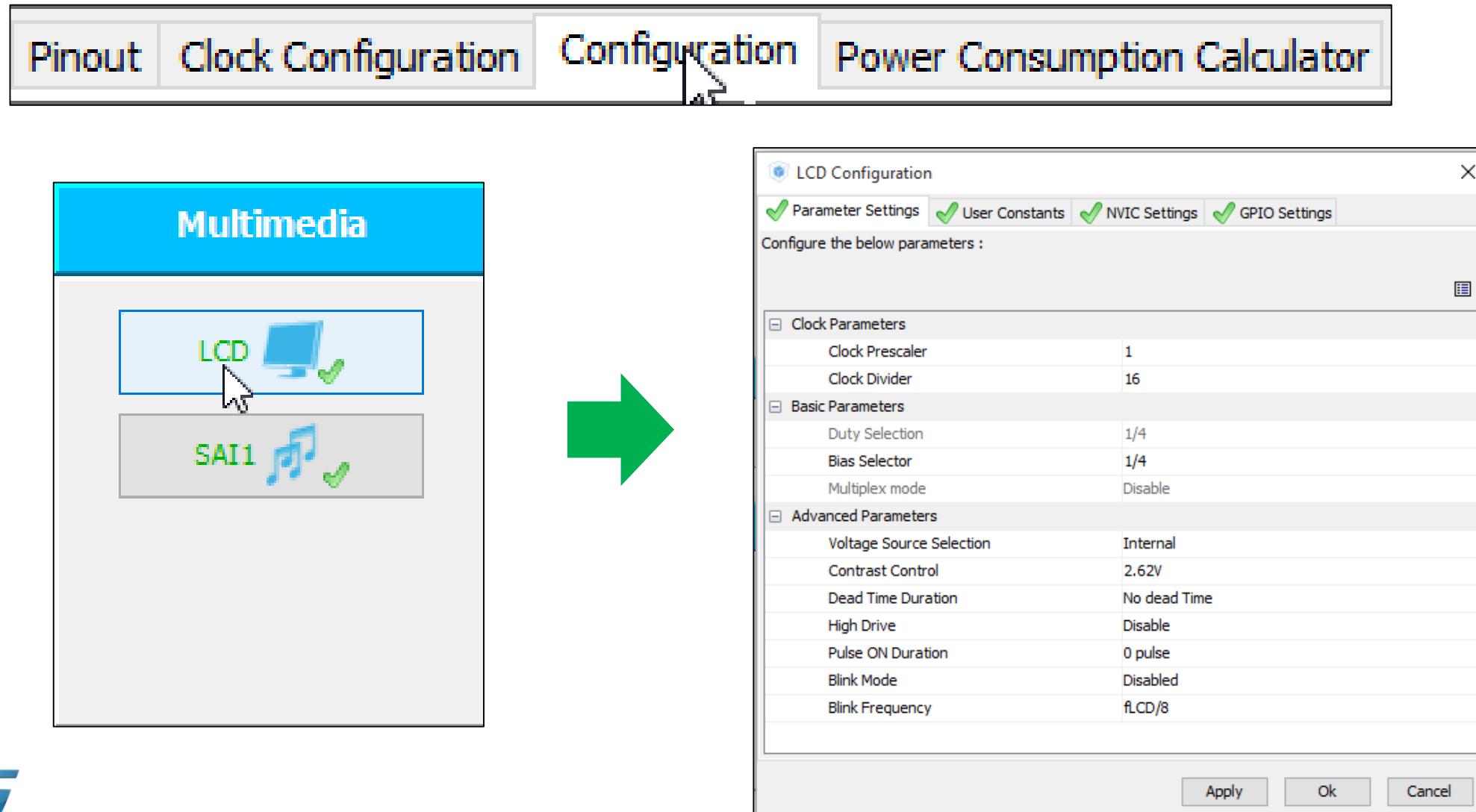


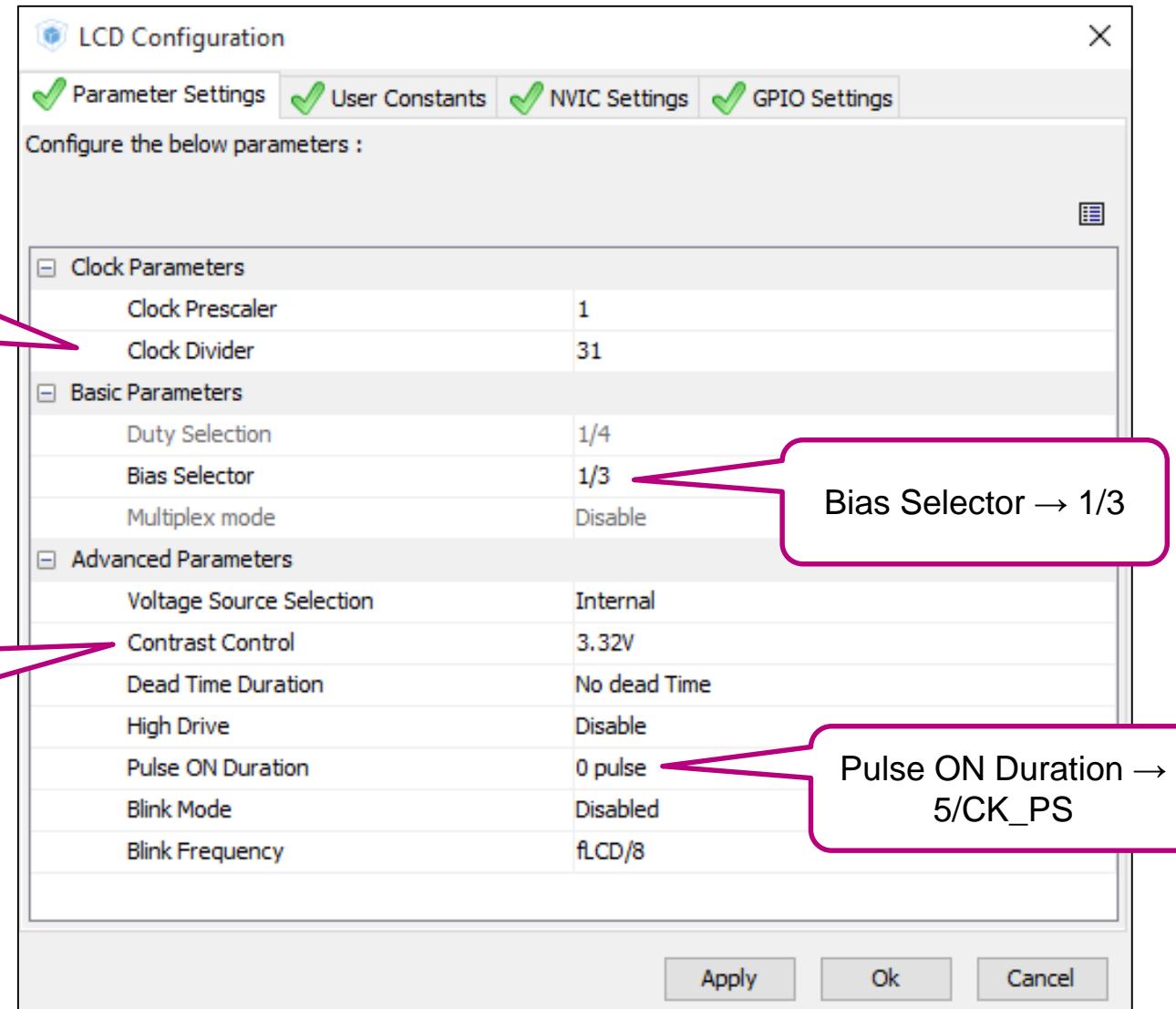
Go to Clock Configuration

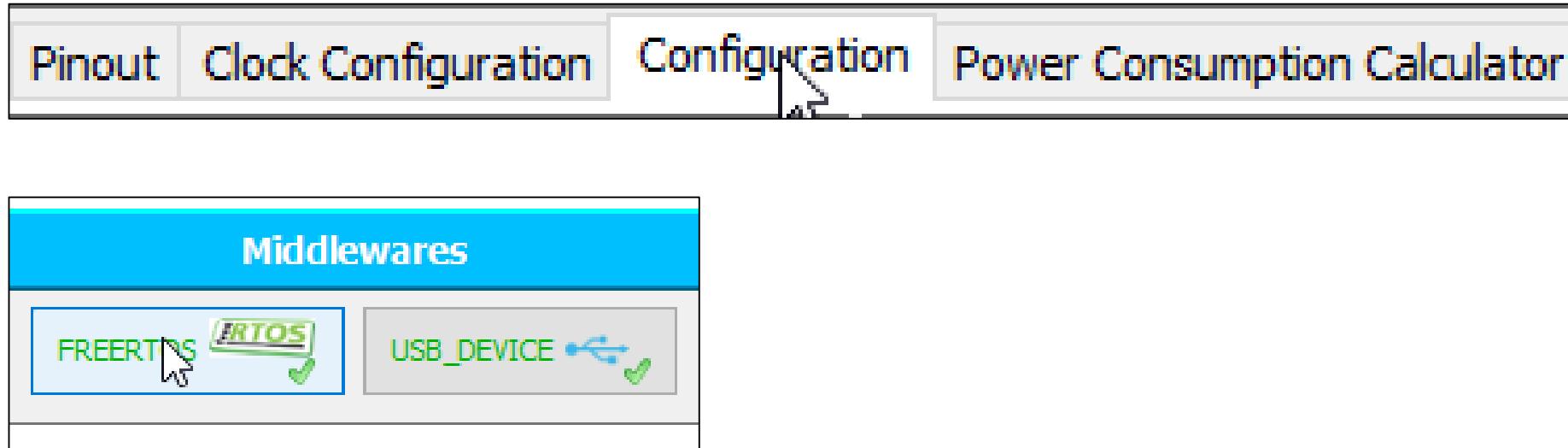


LSE  
(RTC/LCD Source Mux)





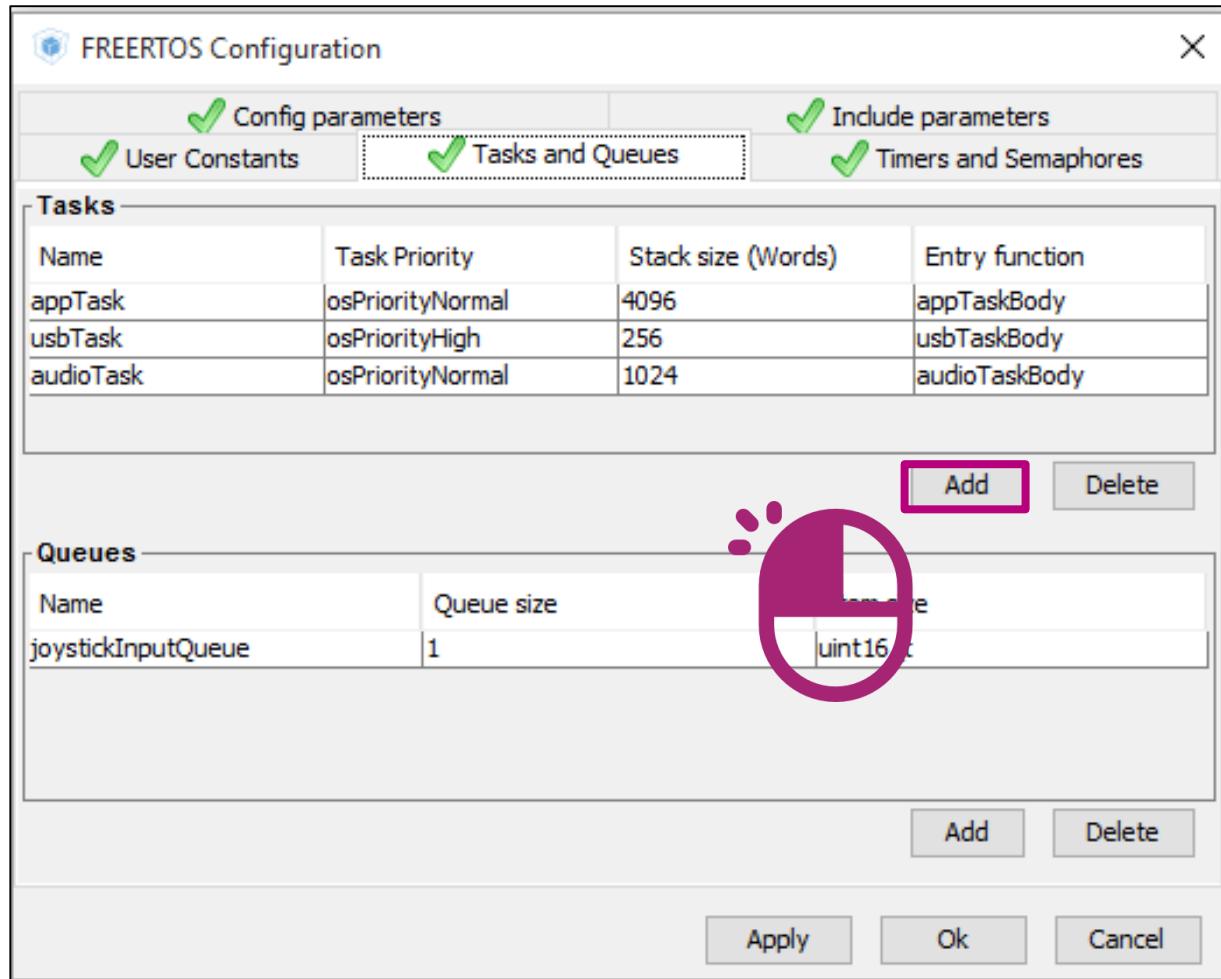




The screenshot shows the STM32CubeMX software interface with the Configuration tab selected. The Middlewares section is expanded, showing the configuration for the FREERTOS and USB\_DEVICE middlewares. A large green arrow points from the FREERTOS configuration table to the right. The table lists three tasks: appTask, usbTask, and audioTask, along with a queue for joystick input.

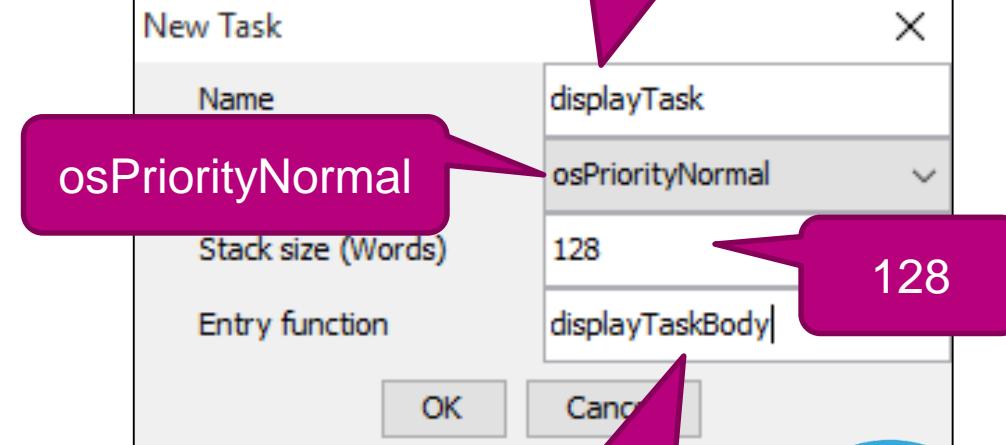
Name	Task Priority	Stack size (Words)	Entry function
appTask	osPriorityNormal	4096	appTaskBody
usbTask	osPriorityHigh	256	usbTaskBody
audioTask	osPriorityNormal	1024	audioTaskBody

Name	Queue size	Item size
joystickInputQueue	1	uint16_t

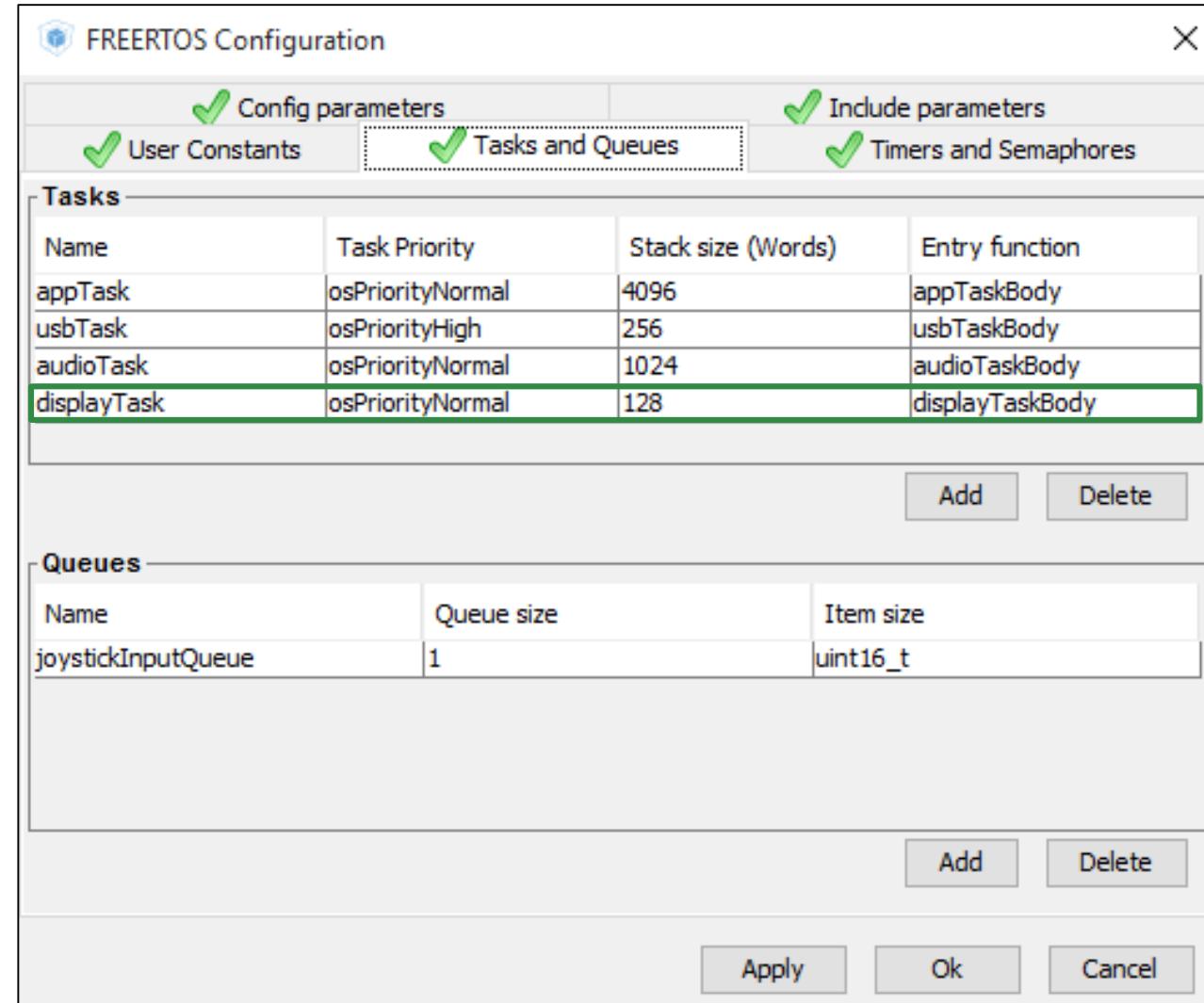


displayTask

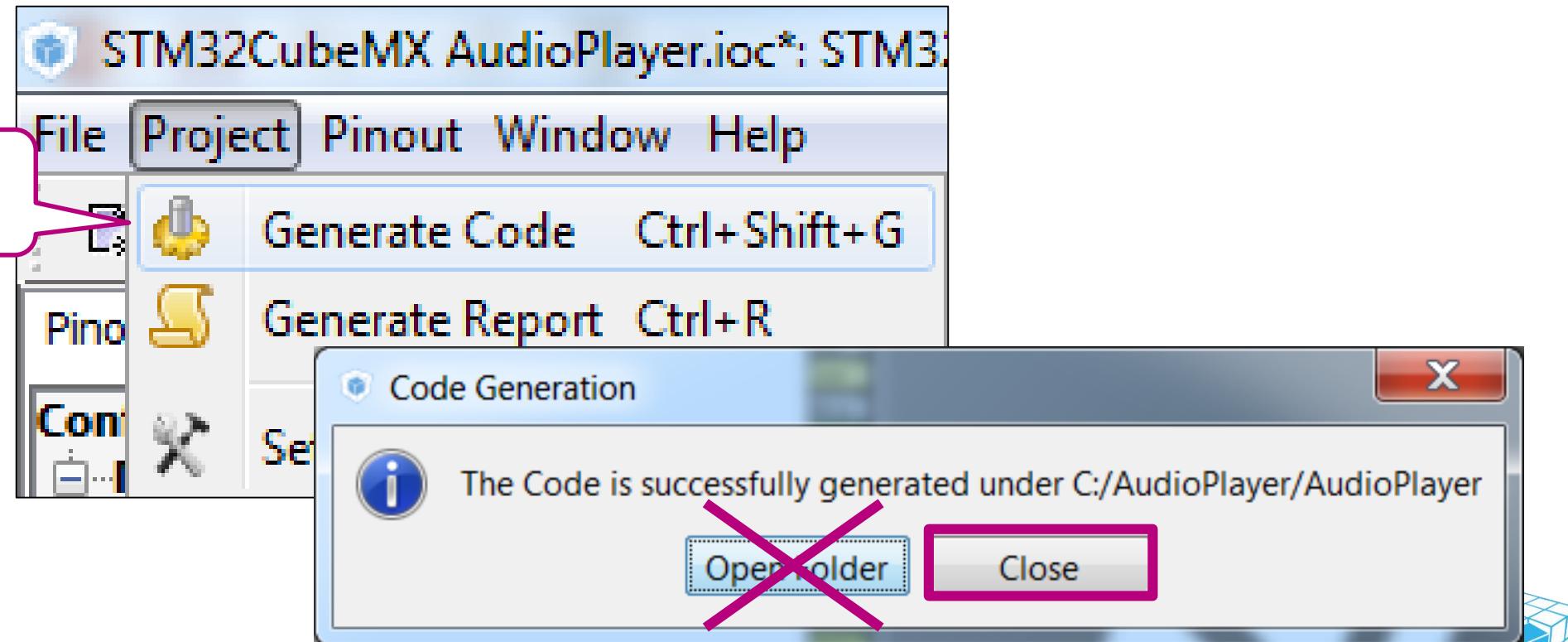
displayTask



displayTaskBody



Do not open the project yet



- Apply patch from folder

**C:\STM32L4\_Workshop\HandsOn\2\_Putting\_All\_Together\Patch\STEP6\_Display**

to root folder of your project

(C:\AudioPlayer\AudioPlayer\)

- It contains the following files:

.\Inc\

lcd.h

.\Src\

freertos.c

lcd.c

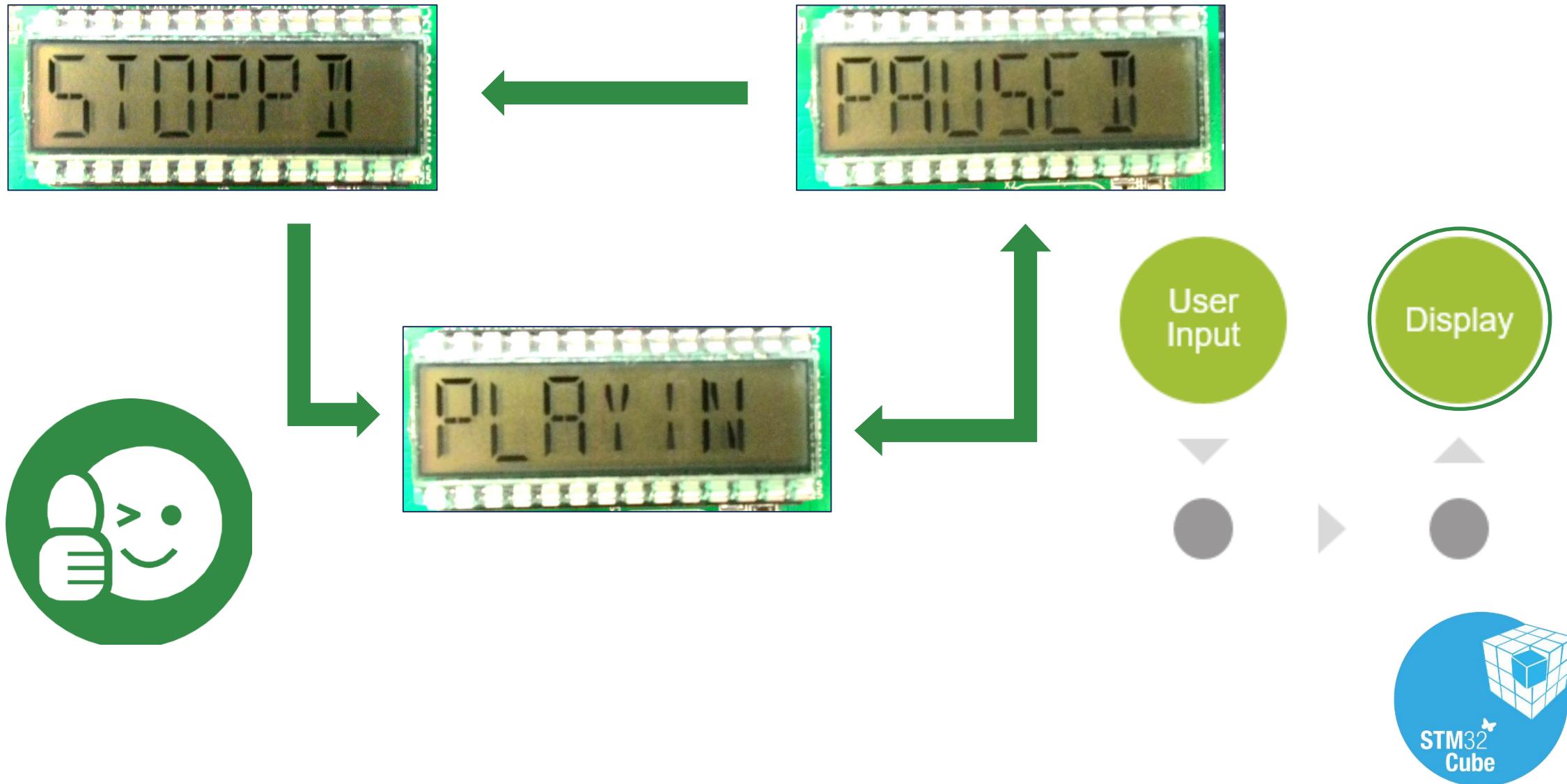
This operation is STM32CubeMX non-intrusive. You can re-generate the project later and the modifications will be retained.

- These files contains the main needed source code the user has to add to work with the particular glass LCD available on the STM32L4 discovery kits

- lcd.c

```
LCD_GLASS_Clear(void);  
LCD_GLASS_DisplayString(...);
```

1. Open the project
2. Compile and download
3. Press the reset button on discovery once download has finished

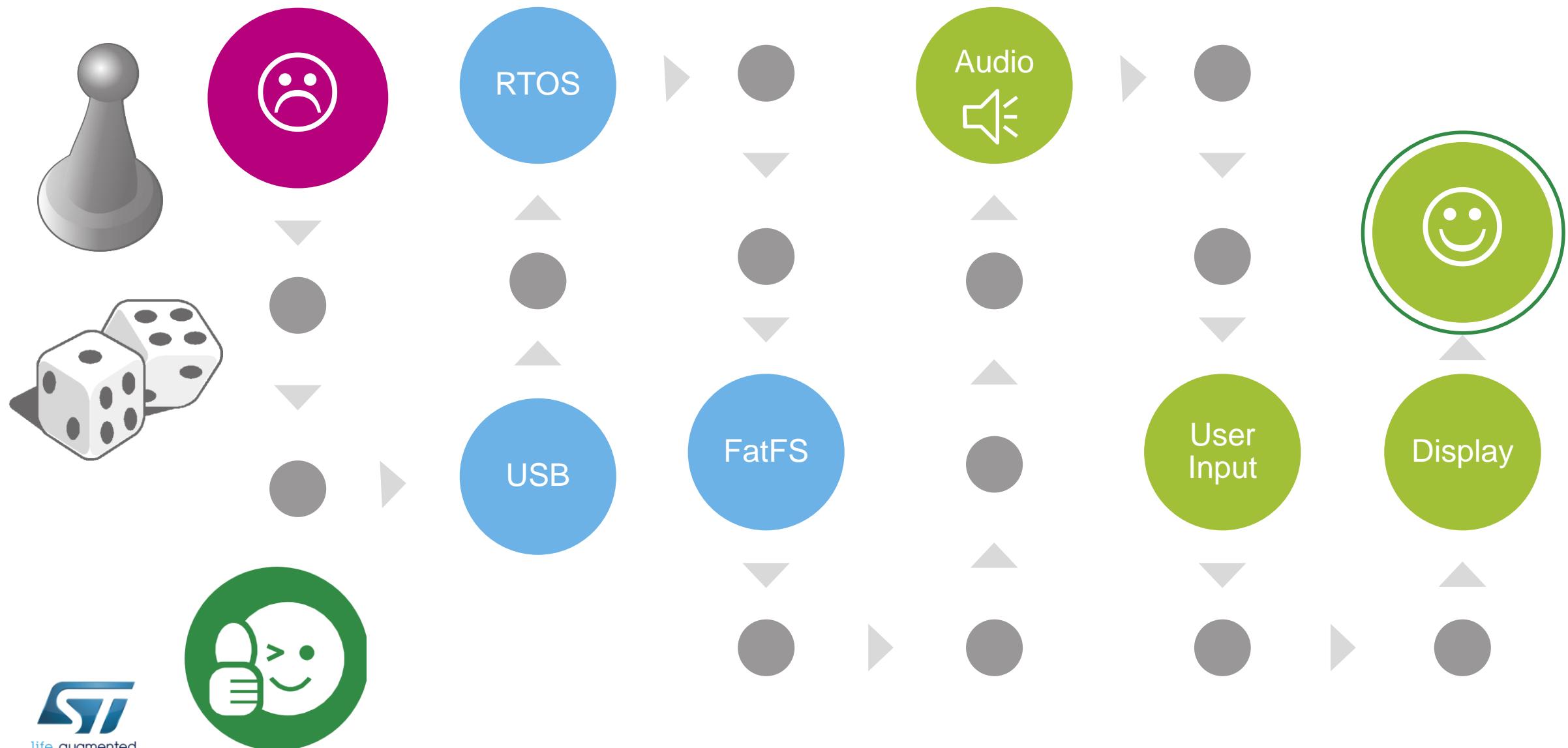


Because it can run completely independent on the other threads in the application

- You can implement previous state change checking mechanism

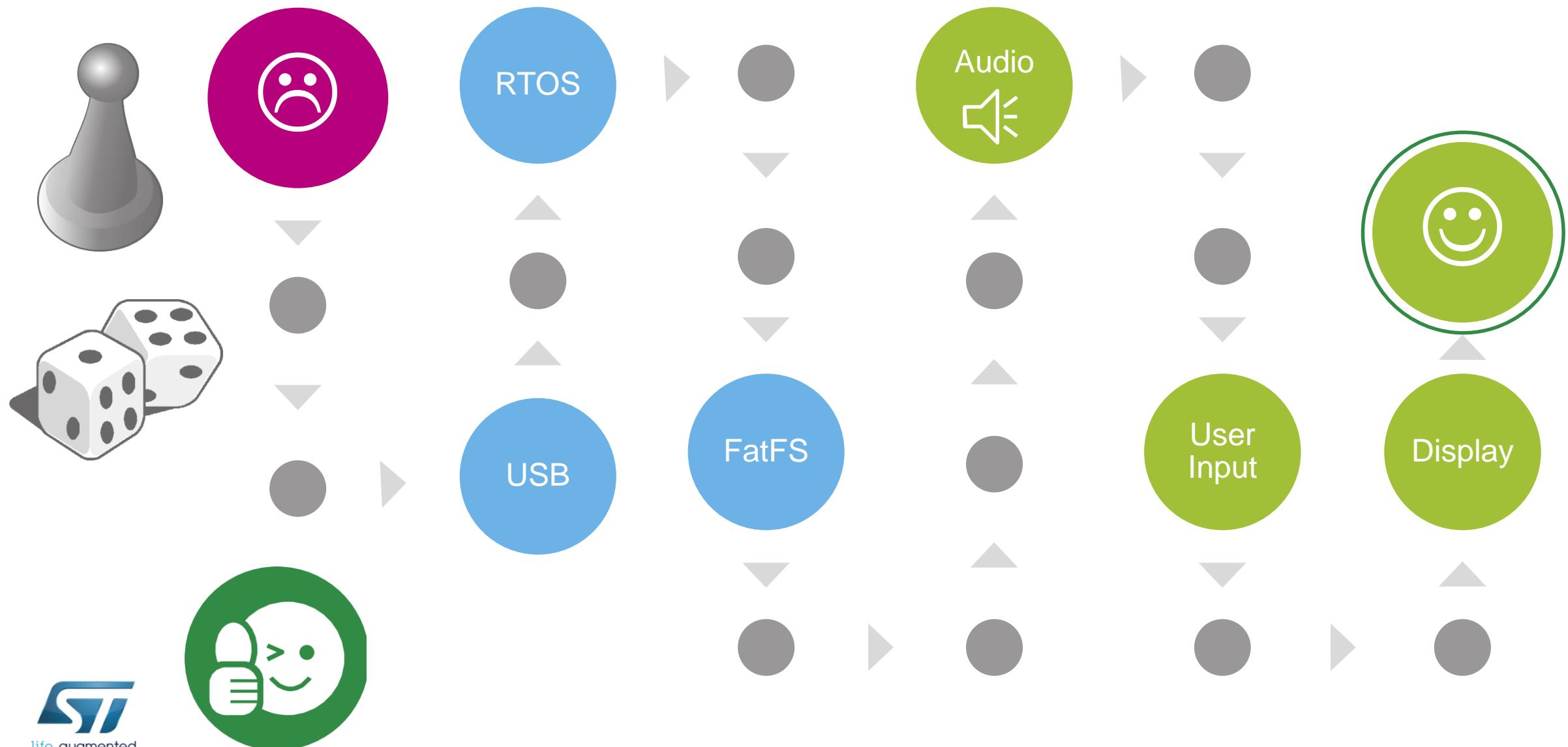
#### EXAMPLE USE-CASE

You may want to add animations or blinking segments with changing refresh rate related to the timeout variable used as argument of the osDelay(...) call.



# What you have learned?

182



# Enjoy!

7



A man in a purple tank top and red pants is captured in mid-air, performing a complex breakdance move. To his right is a graphic featuring a green circle with a white butterfly icon and the text 'STM32 L4'. To the right of that is a blue circular graphic with a white silhouette of a person jumping, surrounded by small icons of a smartphone, a cube, and a microchip. At the bottom, there are social media links: a Facebook icon with '/STM32', a Twitter icon with '@ST\_World', and a logo with 'e2e' and the URL 'st.com/e2e'.

[www.st.com/stm32l4](https://www.st.com/stm32l4)