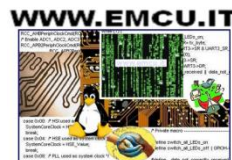




Presenter's name

STM32F4 + CUBE MX + KEIL



HW and SW tools

HW:

- NUCLEO-F401RE



NUCLEO-F401RE

- Cortex-M4 + FPU, 84MHz
- 512-KB Flash, 96-KB SRAM
- USB_OTG_FS SDIO

SW:

- KEIL Compiler
- CUBE MX
- STM32F4 HAL Library

CUBE

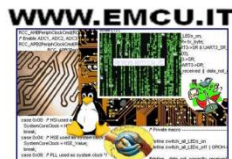
CUBE



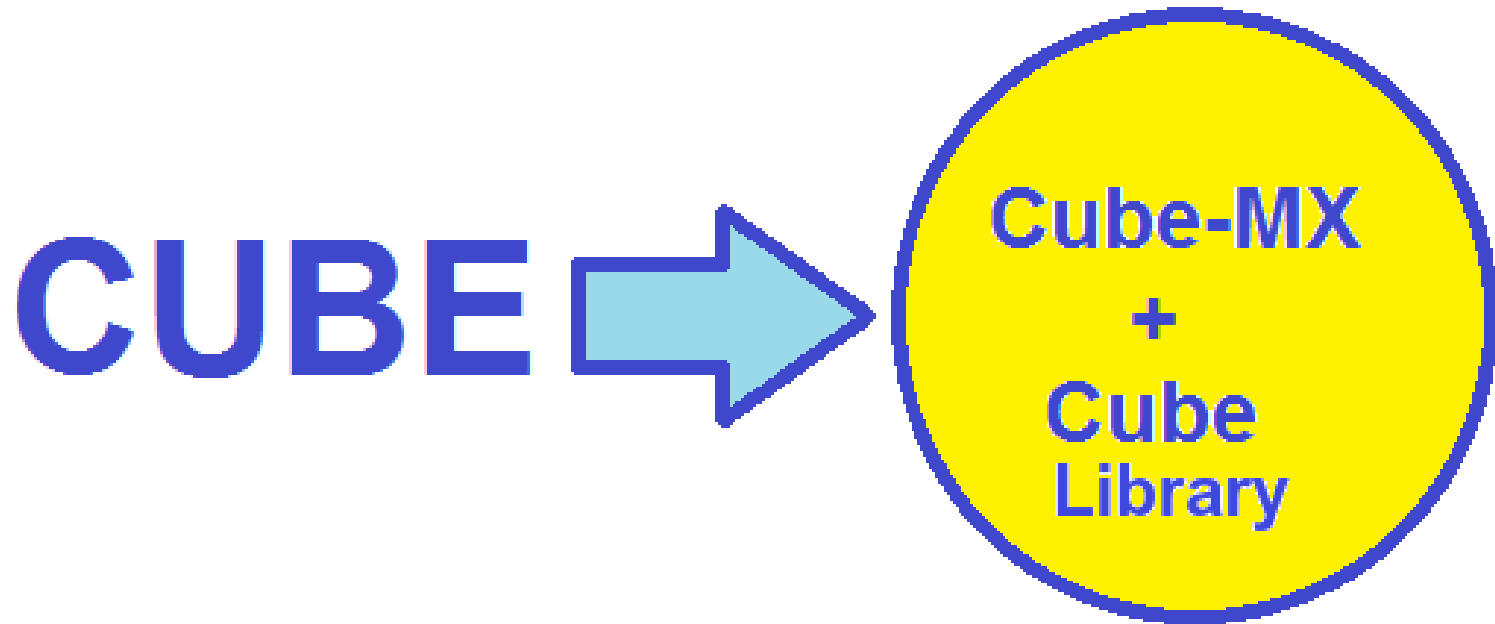
3



11 June 2016

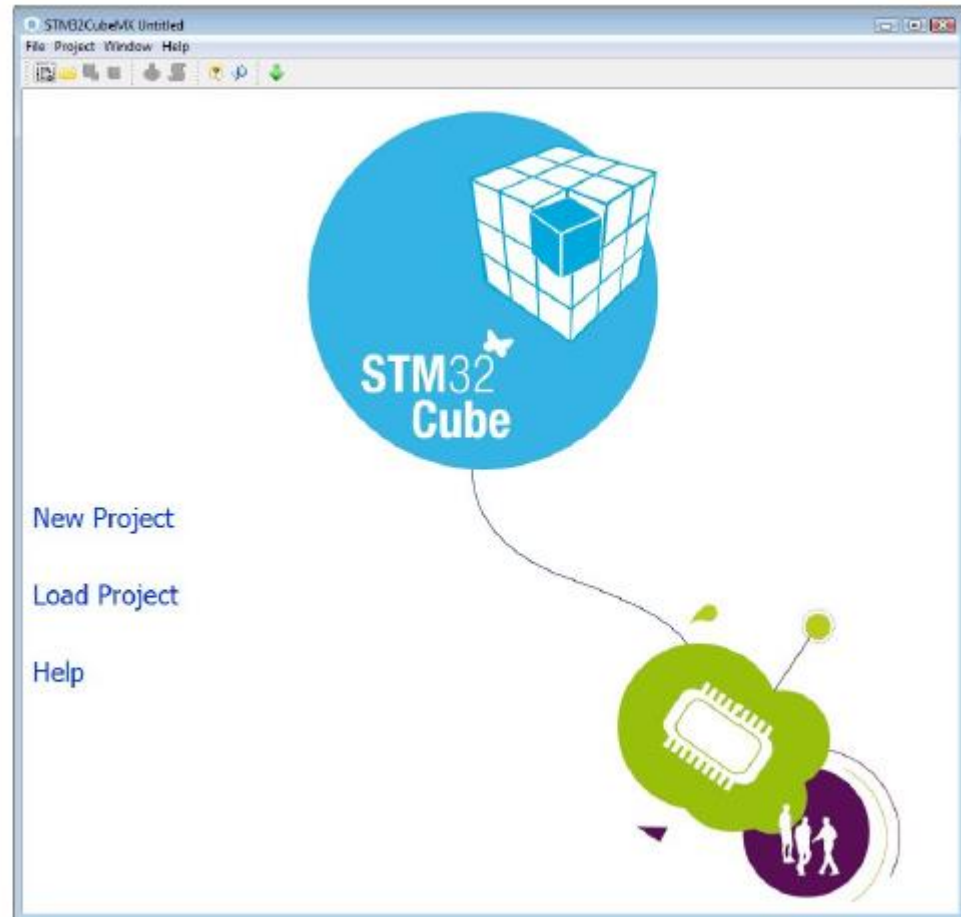


Introduction to CubeMX 1/3

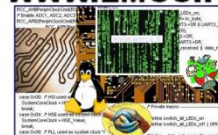


Introduction to CubeMX 2/3

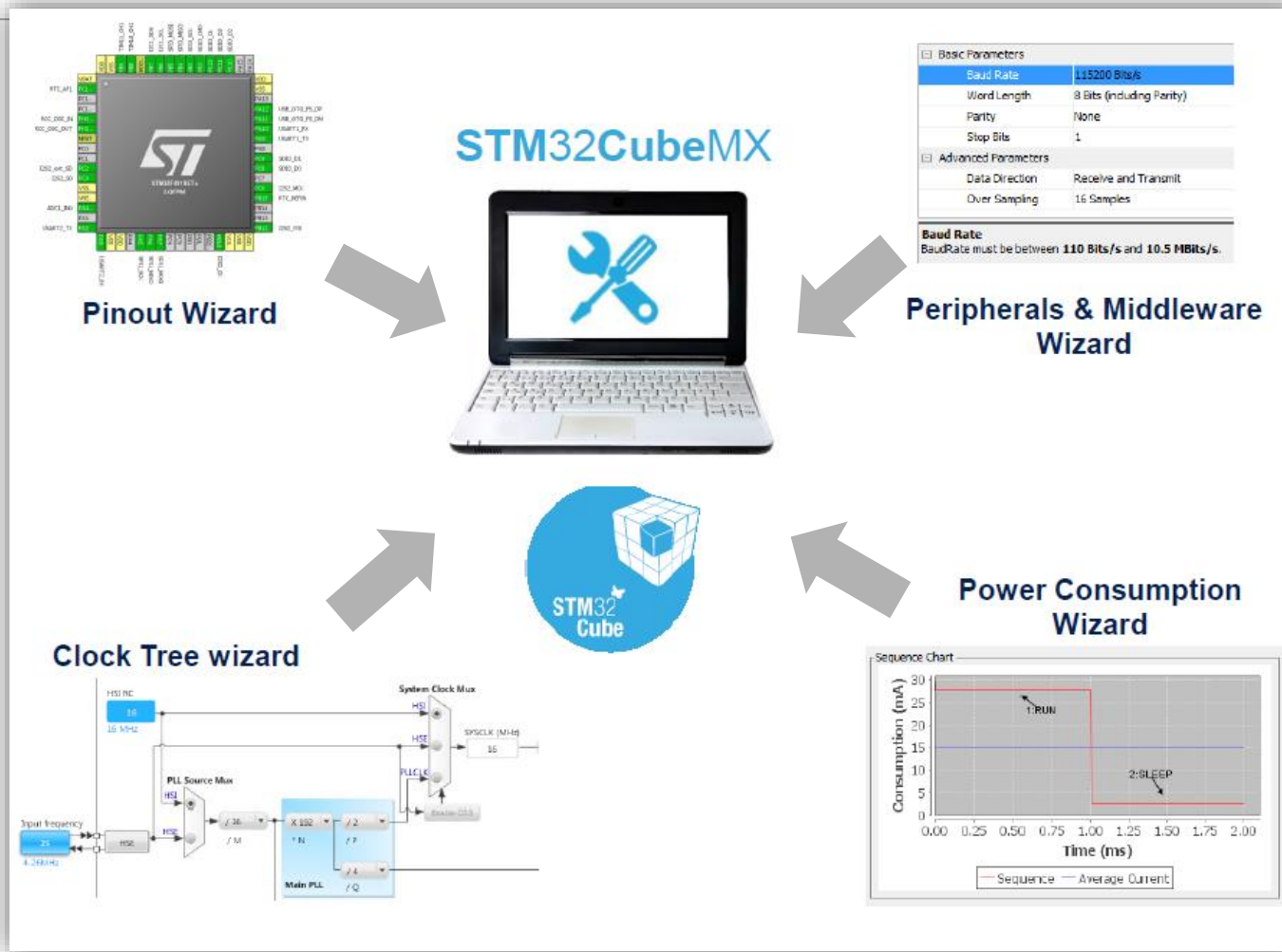
- MCU selector
- Pinout configuration
- Clock tree initialization
- Peripherals and middleware parameters
- Code generation
- Power consumption calculator



WWW.EMCU.IT

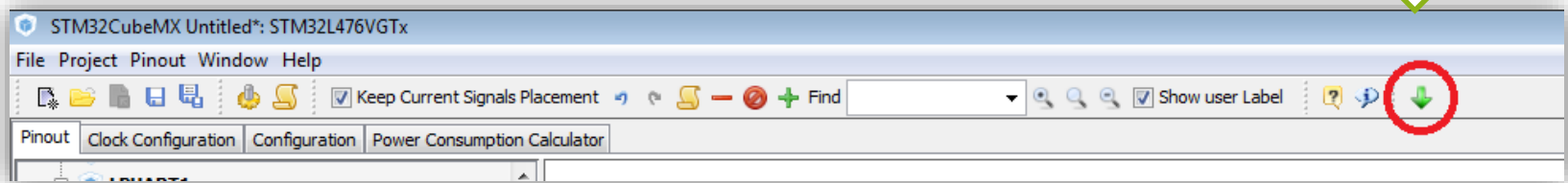
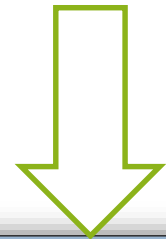


Introduction to CubeMX 3/3

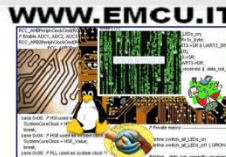
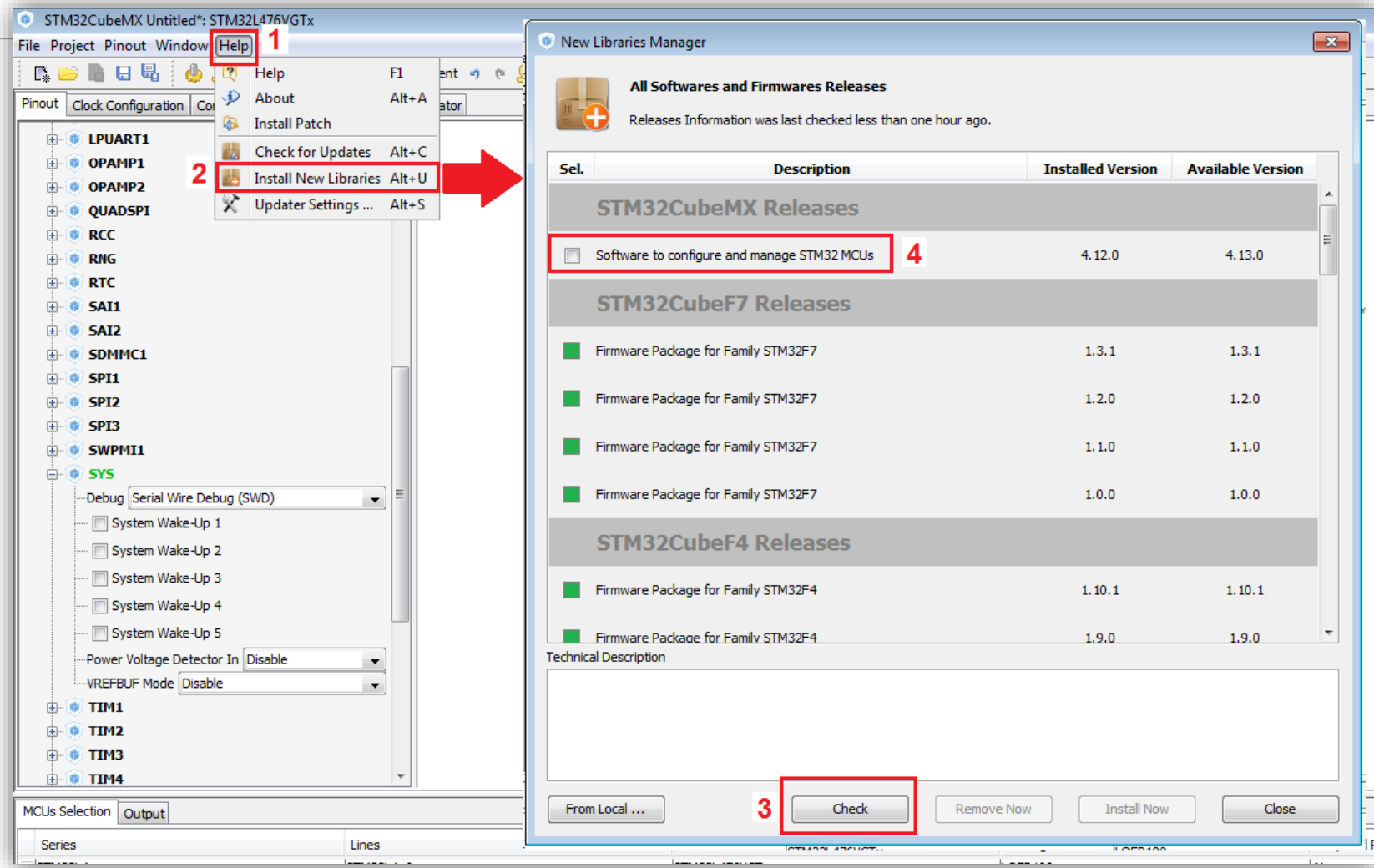


CubeMX request update 1/2

The **green arrow** indicate that are presents some updates.



CubeMX request update 2/2



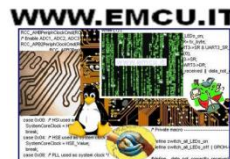
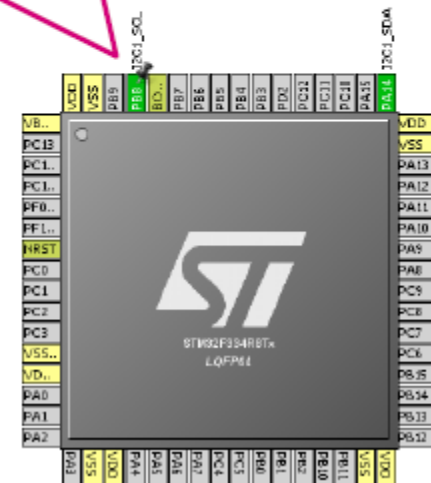
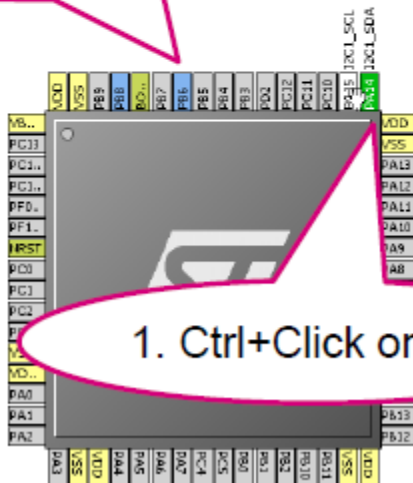
CubeMX: Pinout configuration

- Signals can be set/moved directly from the pinout view
 - To see alternate pins for a signal Ctrl+Click on the signal, you can then drag and drop the signal to the new pin (keep pressing the Ctrl key)

2. Show alternative positions

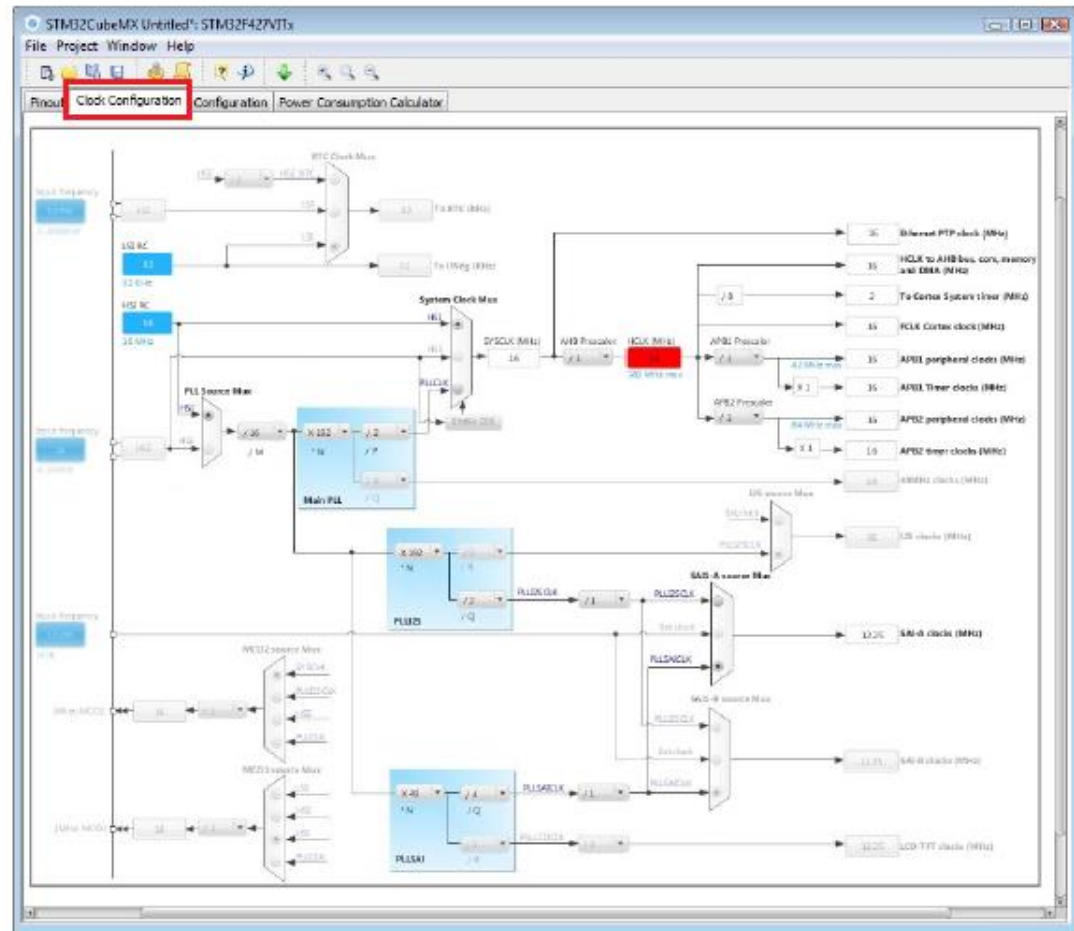
3. Move pin to new position

1. Ctrl+Click on pin



CubeMX: Clock tree

- Immediate display of all clock values
- Management of all clock constraints
- Highlight of errors

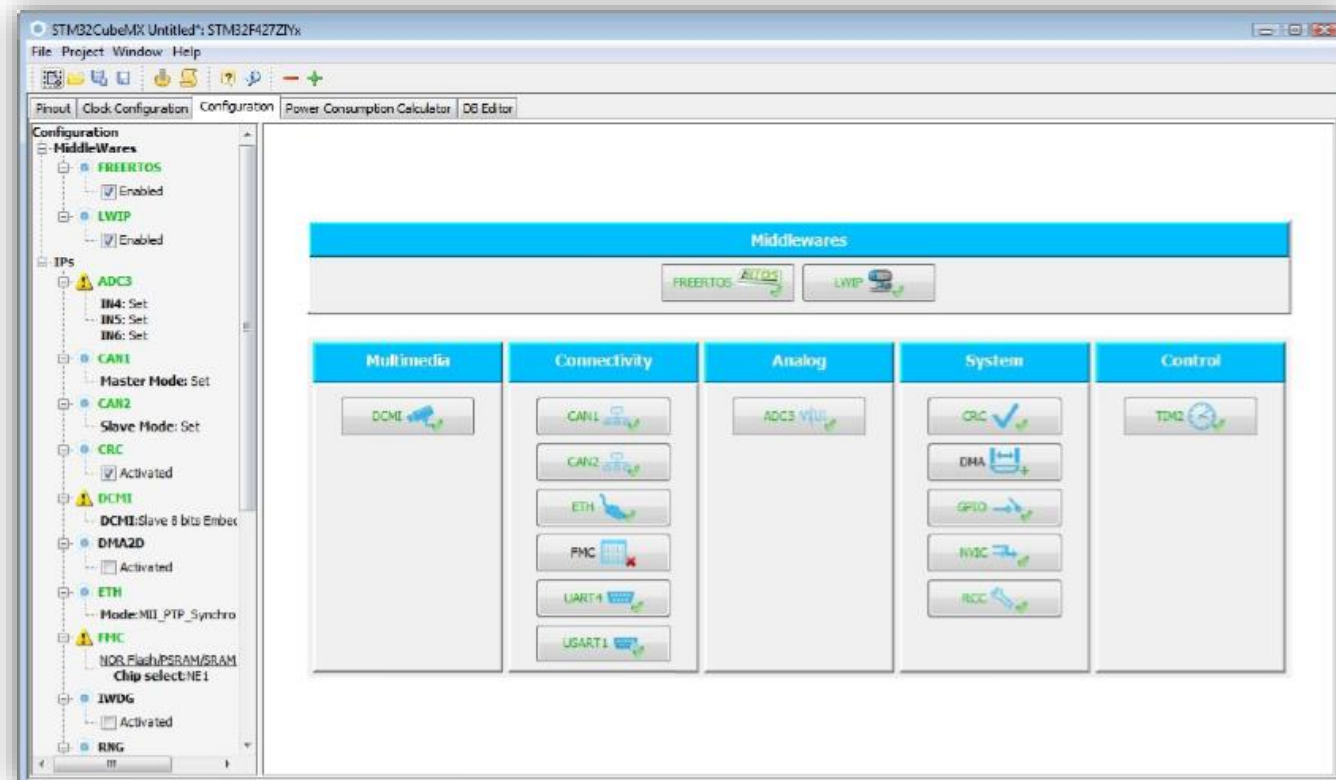


WWW.EMCU.IT



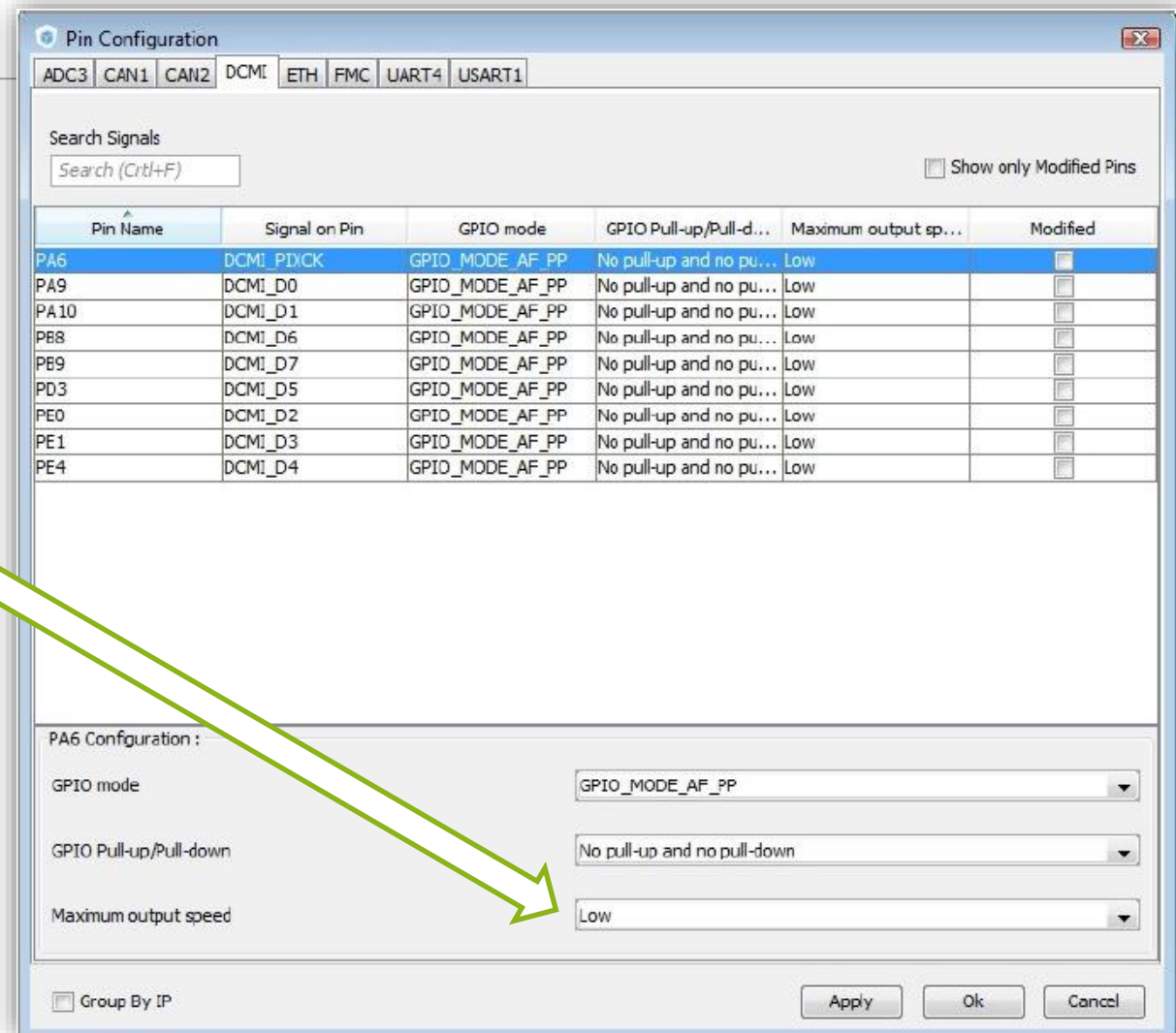
CubeMX: Peripheral and middleware configuration

- Global view of used peripherals and middleware
- Highlight of configuration errors
 - + Not configured
 - ✓ OK
 - x Error
- Read only tree view on the left with access to IPs / Middleware having no impact on the pinout



CubeMX: GPIO Panel

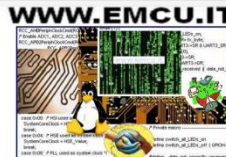
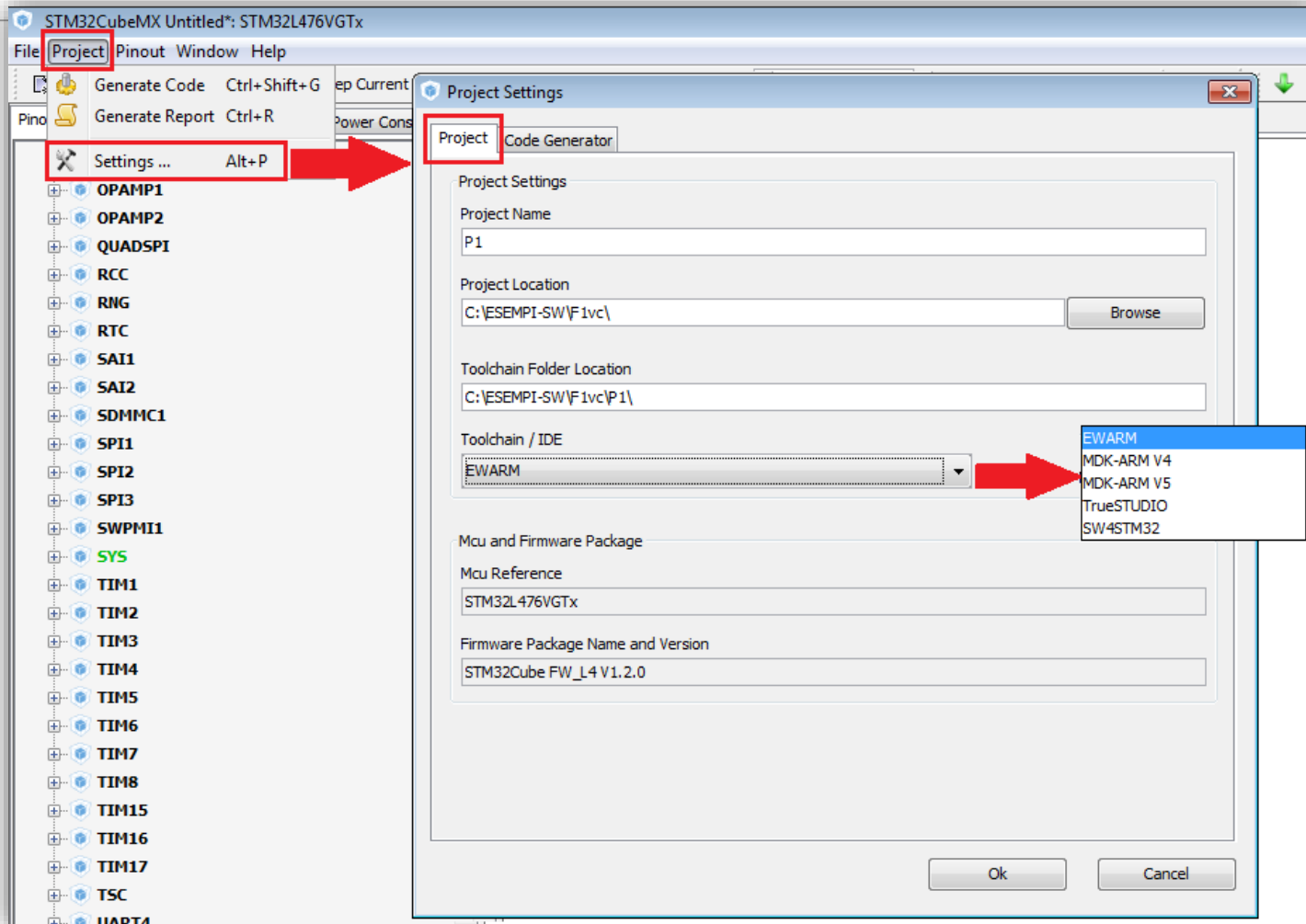
- Most of the GPIO parameters are set by default to the correct value
- You may want to change the maximum output speed
- You can select multiple pin at a time to set the same parameter



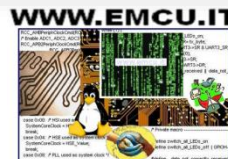
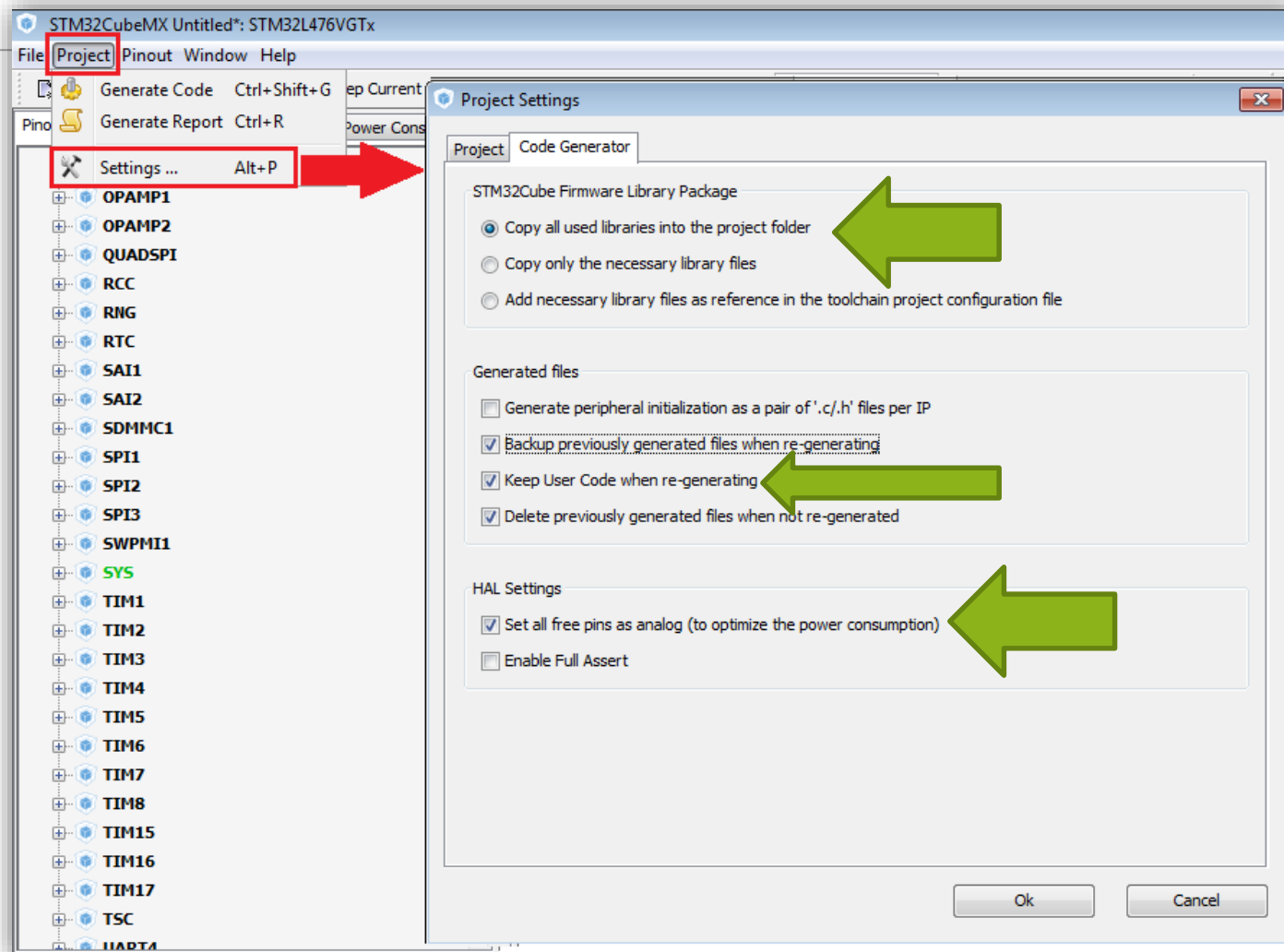
WWW.EMCU.IT



CubeMX generate the code for some GUI 1/3



CubeMX generate the code for some GUI 2/3



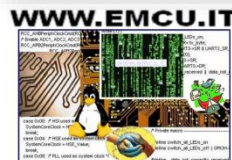
CubeMX generate the code for some GUI 3/3

- Generation of all the C initialization code
- Automatic integration with partners toolchains
- User code can be added in dedicated sections and will be kept upon regeneration

Generated files

- ☐ Generate peripheral initialization as a pair of '.c/.h' files per IP
- ☐ Backup previously generated files when re-generating
- ☒ Keep User Code when re-generating
- ☒ Delete previously generated files when not re-generated

```
22  /*
23  */
24  /* Includes -----
25  #include "stm32f4xx_hal.h"
26  #include "cmsis_os.h"
27  #include "lwip.h"
28  #include "usb_device.h"
29
30  /* Define structures */
31  ADC_HandleTypeDef hadc1;
32
33
34  /* USER CODE BEGIN 0 */
35
36  /* USER CODE END 0 */
37  /* Private function prototypes -----
38  static void SystemClock_Config(void);
39  static void StartThread(void const * argument);
40  static void MX_GPIO_Init(void);
41  static void MX_ADC1_Init(void);
42  static void MX_NVIC_Init(void);
43
44  int main(void)
45  {
46  /* USER CODE BEGIN 1 */
47
48  /* USER CODE END 1 */
49  /* MCU Configuration-----
50  /* Reset of all peripherals, Initializes the Flash interface
51  HAL_Init();
52  /* Configure the system clock */
```



CubeMX: Power consumption calculator

STM32CubeMX Fiorentini-STM32L053C6.ioc: STM32L053C6Tx

File Project Power Window Help

Pinout Clock Configuration Configuration Power Consumption Calculator

Microcontroller Selected

Series: STM32L0
Line: STM32L0x3
MCU: STM32L053C6Tx
[Datasheet:](#) 025844_Rev4

Parameter Selection

Ambient Temperature (°C): 25
Vdd Power Supply (V): 3.0

Battery Selection

Select

Battery: LI-SOCL2(A3400)
In Series: 1
In Parallel: 1
Capacity: 3400.0 mAh
Self Discharge: 0.08 %/month
Nominal Voltage: 3.6 V
Max Cont Current: 100.0 mA
Max Pulse Current: 200.0 mA

Information Notes

Help

Sequence

Load Save Delete Compare

Transitions checker
☐ Enabled Show log

Sequence Table

Step	Mode	Vdd	Range/Scale	Memory	CPU/Bus Freq	Clock Config	Src Freq	Peripherals	Add. Current	Step Current	Duration	DMIPS	Volta...	Ta ...	C...
1	RUN	3.0	Range2-Medium	RAM	4.0 MHz	HSEBYP_4MHz PLL_OFF	4.0 MHz	GPIOA GPI...	0 mA	615 µA	3 ms	3.812	Battery	85.0	Da...
2	RUN	3.0	Range1-High	FLASH	8.0 MHz	HSEBYP_8MHz PLL_OFF	8.0 MHz	GPIOA GPI...	0 mA	1.77 mA	1 ms	7.624	Battery	85.0	Da...
3	STOP	3.0	NoRange	n/a	0 Hz	LSE RTC_ON IWDG_O...	32.768 kHz	GPIOA GPI...	0 mA	4 µA	100 ms	0.0	Battery	85.0	Da...

Step

Add Delete Duplicate Up Down Undo Redo

Display
Plot: All Steps Ext. Display

Results Charts

Consumption Profile by Step

Consumption (mA)

Time (ms)

1: RUN 2: RUN 3: STOP

— Idd by Step — Average Current

Results Summary

Sequence Time / Ta Max 104 ms / 85.0 °C
Battery Life Estimation 9 years, 1 month, 27 days & 11 hours

Average Consumption 38.61 µA
Average DMIPS 4.75 DMIPS

WWW.EMCU.IT



HAL library

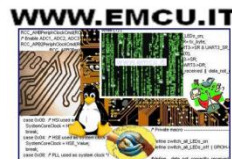
HAL library



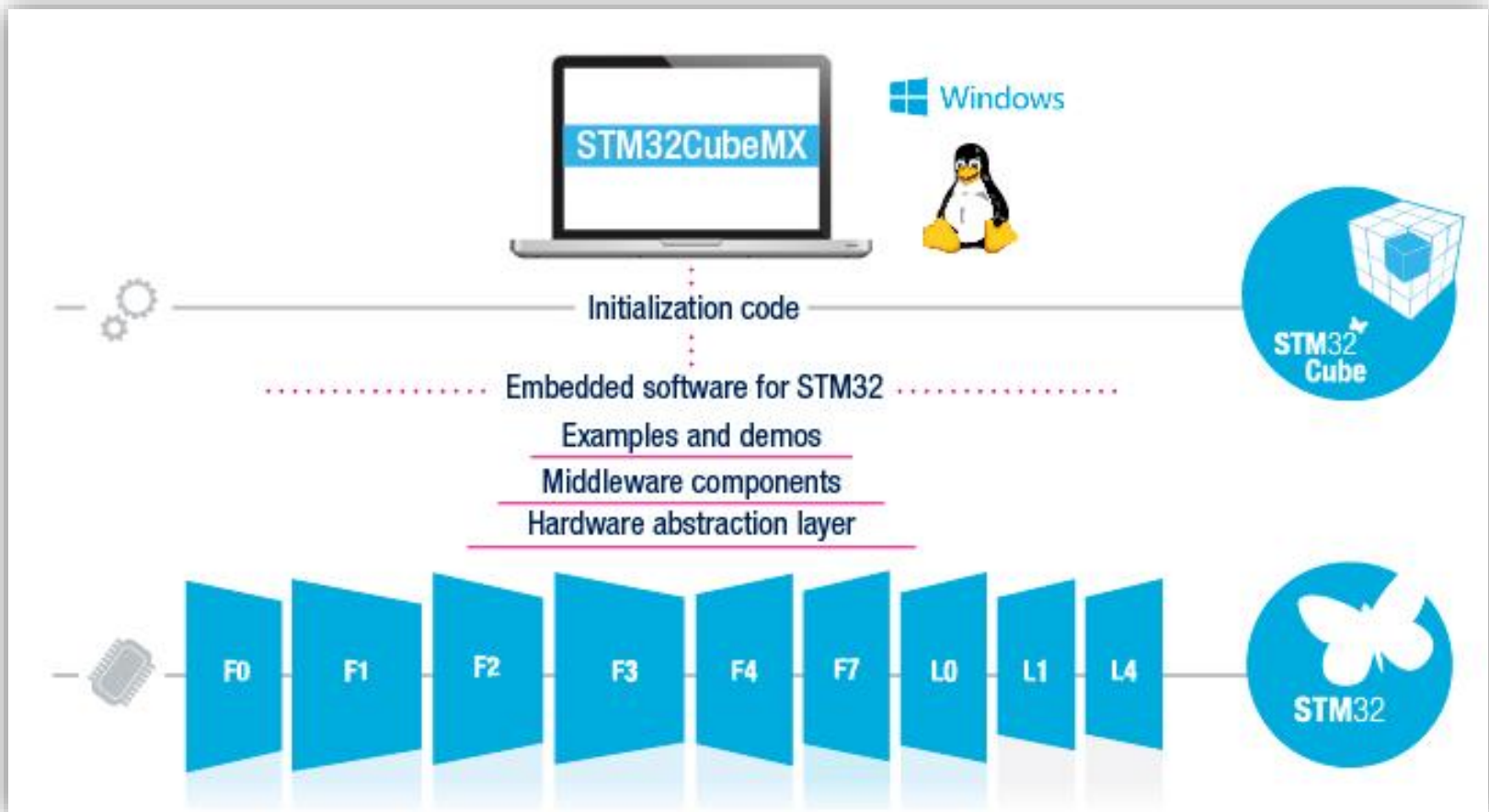
17



11 June 2016



HAL library – HAL == hardware abstraction layer

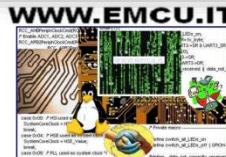
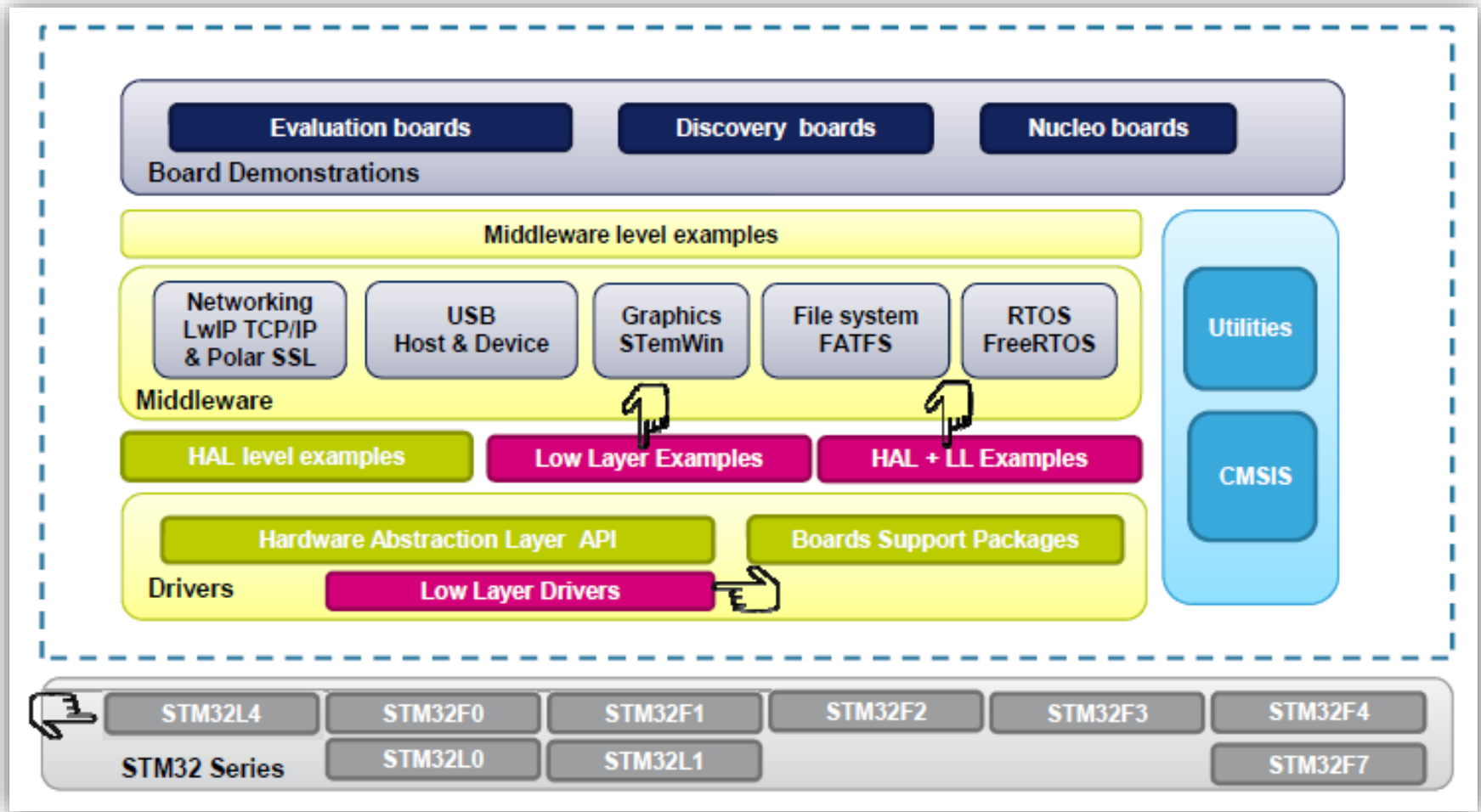


WWW.EMCU.IT



HAL library

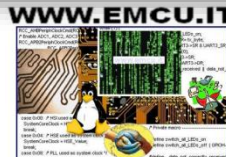
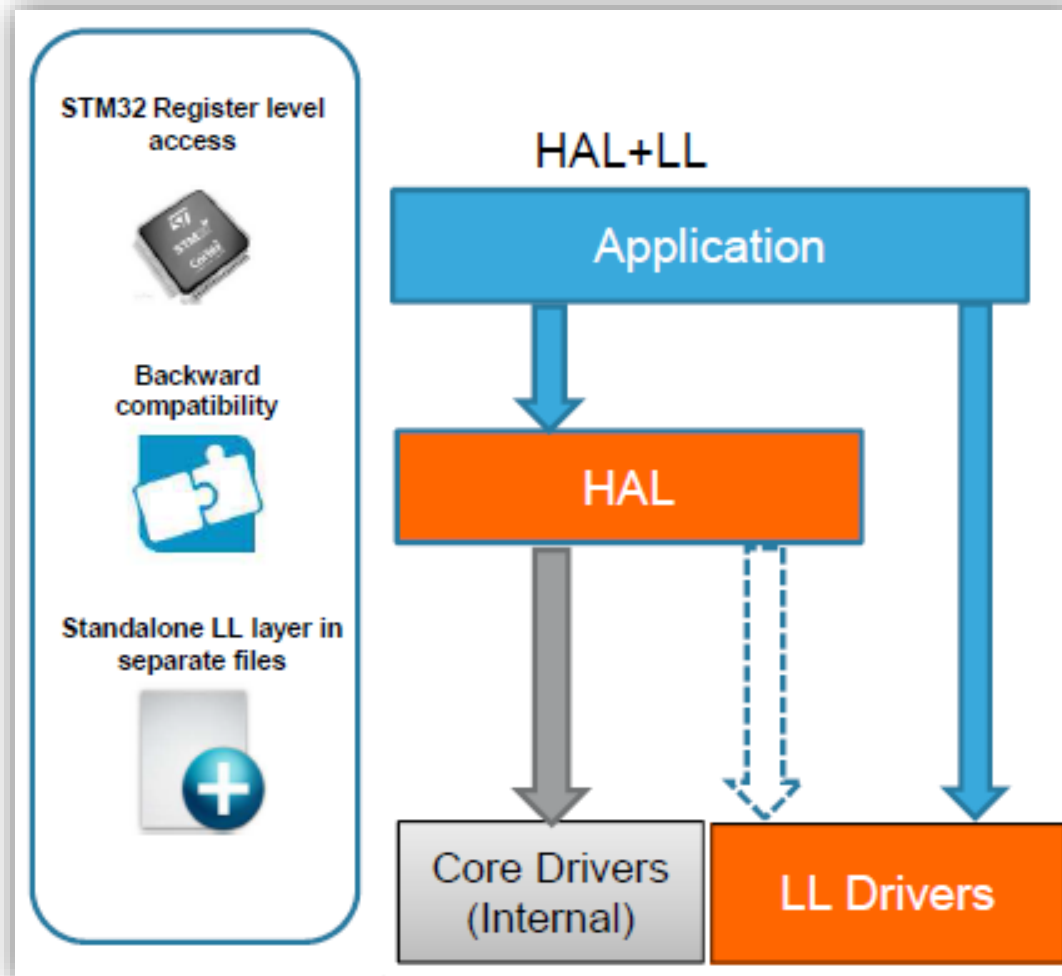
The HAL library are [here](#).



HAL library – LL now (June 2016) only available on L4

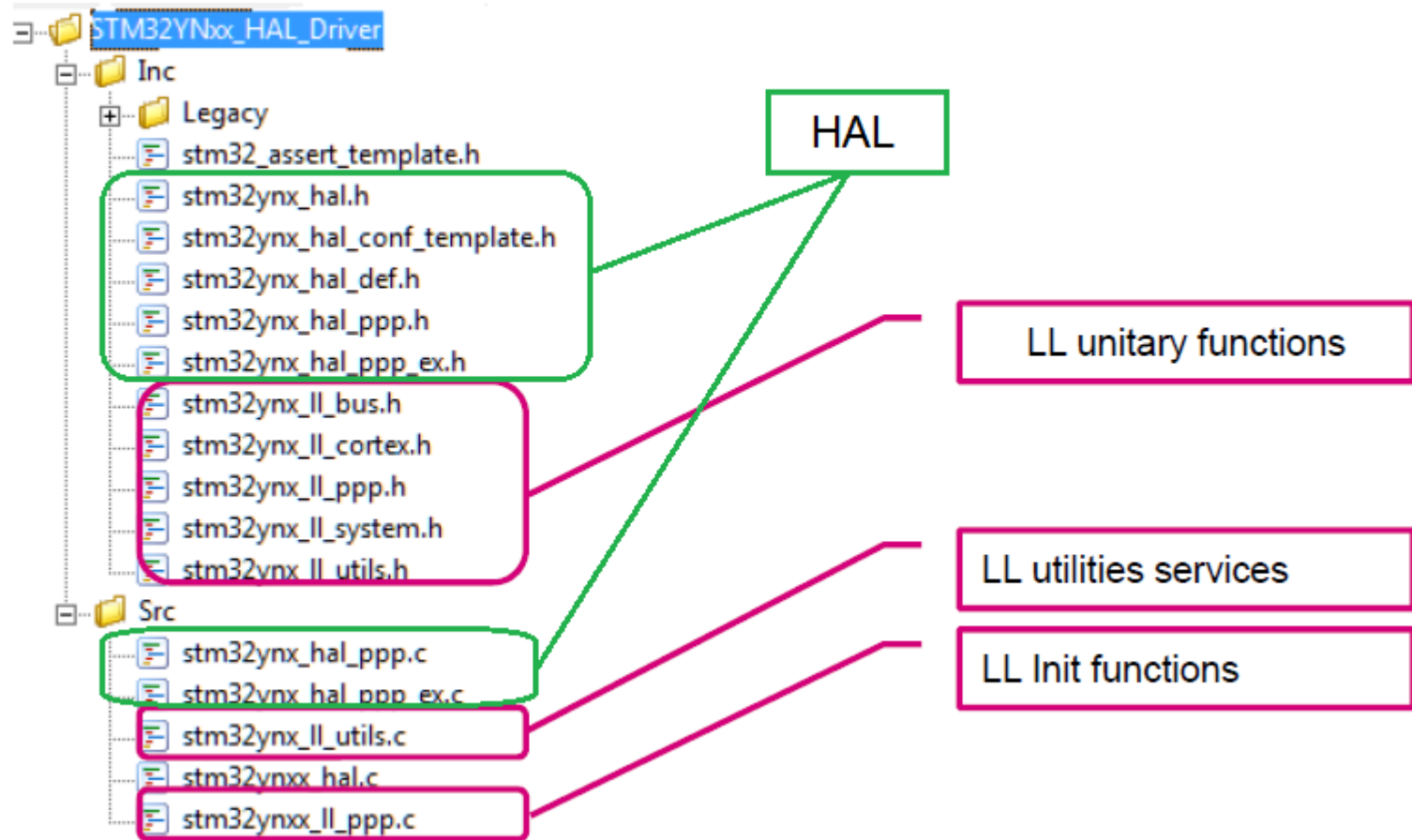
- **STM32Cube HAL & LL** are complementary and covers a wide range of applications requirements:
 - HAL offers high level and functionalities oriented APIs, with high portability level and hide product/IPs complexity to end user
 - LL offers low level APIs at registers level, w/ better optimization but less portability and require deep knowledge of the product/IPs specification
- The new Low Layer (LL) is offering the following services:
 - **Unitary static inline functions for direct register access** (provided in *.h files)
 - One-shot operations that can be used by the HAL drivers or from application level.
 - Independant from HAL and can be used in standalone usage (without HAL drivers)
 - Full features coverage of the supported IP
 - **Init functions** (provided in *.c files)
 - compatible with Standard peripheral library

HAL library – LL now (June 2016) only available on L4



HAL library – LL now (June 2016) only available on L4

LL drivers are located in the Src/Inc HAL Driver folders



WWW.EMCU.IT



HAL library – LL now (June 2016) only available on L4

Covered peripherals (1/2)

Peripherals (IPs)		STM32Cube Support	
System	Flash	HAL Yes	LL No (some of the Flash features need to be handled in the MISC file to prevent dependency with HAL when using LL PWR driver)
System	EXTI	Yes	Yes
	GPIO	Yes	Yes
	DMAs	Yes	Yes
	PWR	Yes	Yes
	RCC	Yes	Yes
	Cortex	Yes	No (some of the cortex features added: MPU, SYSTICK, CPUID, SLEELDEEP)
	SYSCFG	Yes	Yes
Analog	ADC	Yes	Yes
	SDADC	Yes	Yes
	DAC	Yes	Yes
	COMP	Yes	Yes
	DFSDM	Yes	No
	OPAMP	Yes	Yes
	RTC	Yes	Yes
Timers	TIM	Yes	Yes
	LPTIM	Yes	Yes
	HRTIM	Yes	Yes
	WWDG	Yes	Yes
	IWDG	Yes	Yes
	CRC	Yes	Yes
	CRYP	Yes	No
Cryptography	HASH	Yes	No
	RNG	Yes	Yes

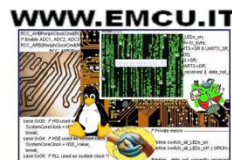
WWW.EMCU.IT



HAL library – LL now (June 2016) only available on L4

Covered peripherals (2/2)

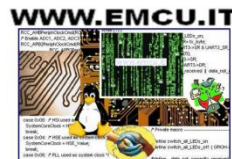
Peripherals (IPs)		STM32Cube Support	
		HAL	LL
Basic Connectivity	I2C/SMBUS	Yes	Yes
	UART/USART/LPUART	Yes	Yes
	SWPMI	Yes	Yes
	SPI/I2S	Yes	Yes
	SDMMC(SDIO)	Yes	No
	CAN	Yes	No
	CEC	Yes	No
Advanced Connectivity	USB-FS-Device	Yes	No
	USB-OTG-FS/HS	Yes	No
	Ethernet	Yes	No
	MDIOS	Yes	No
Interface	FSMC(FMC)	Yes	No
	LCD"Glass"	Yes	No
	LTDC	Yes	No
	DSI	Yes	No
	DMA2D	Yes	Yes
	JPEG	Yes	No
	DCMI	Yes	No
	QSPI	Yes	No
	SPDIF-IN	Yes	No
	SAI	Yes	No



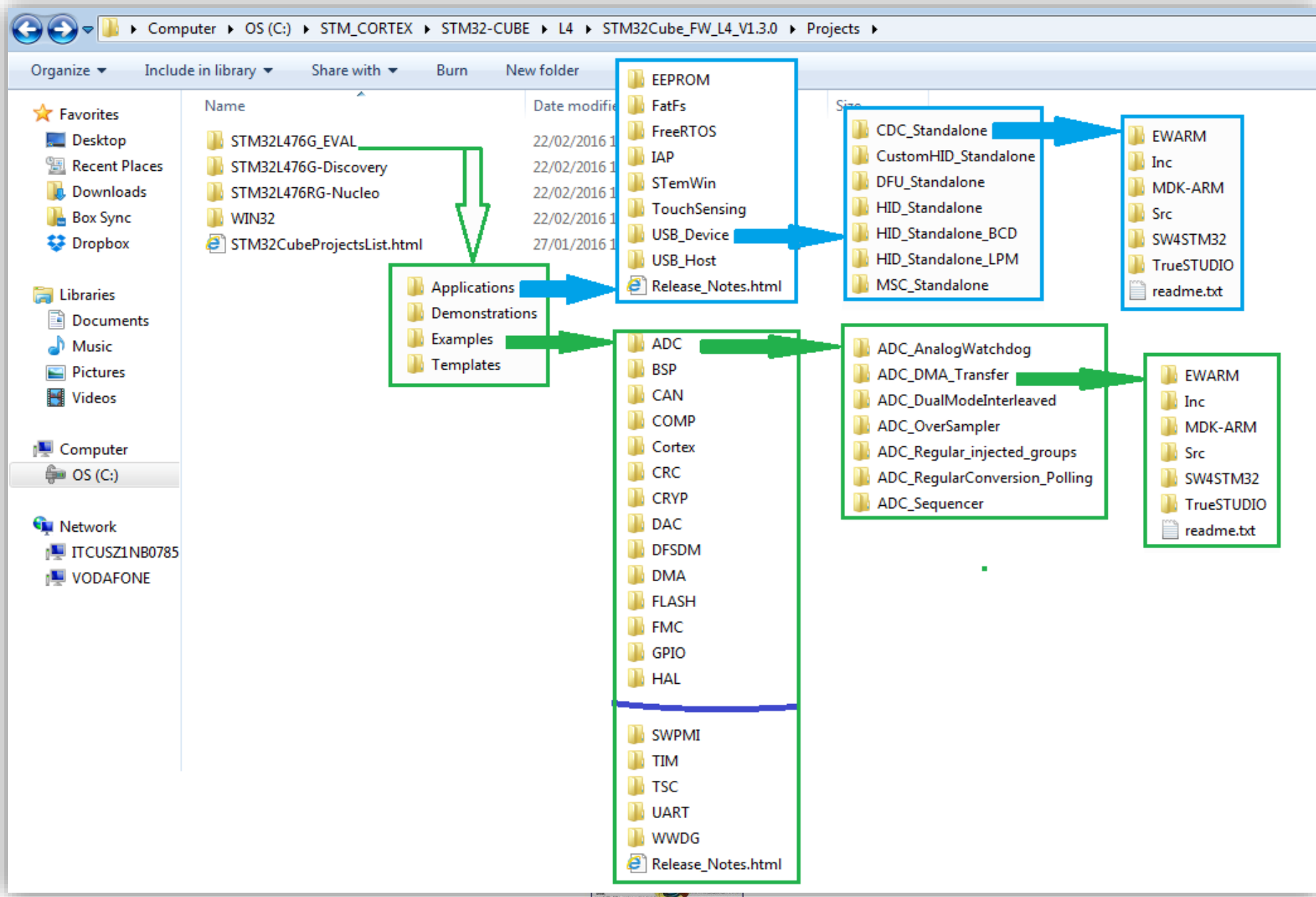
HAL library – LL now (June 2016) only available on L4

HAL vs. LL usage

- To cohabitate the HAL with the LL, user has to be aware about some HAL concepts.
- Main constraint is when the LL overwrites some registers that the content is mirrored in the HAL handles.
- The Low Layer drivers cannot be automatically used with the HAL for the same peripheral instance: mainly can't run concurrent process on the same IP using both APIs, however sequential use is allowed.
- The low layer drivers can be used without any constraint with all the HAL drivers that are not based on handle objects (RCC, Cortex, common HAL, flash and GPIO)
- The LL is intended to be used in expert mode (high knowledge on STM32 hardware aspect)



HAL library - Where to find examples ready to use ?



Start new project

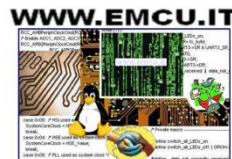
New Project



27

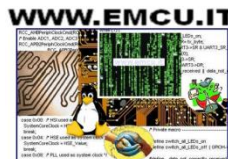
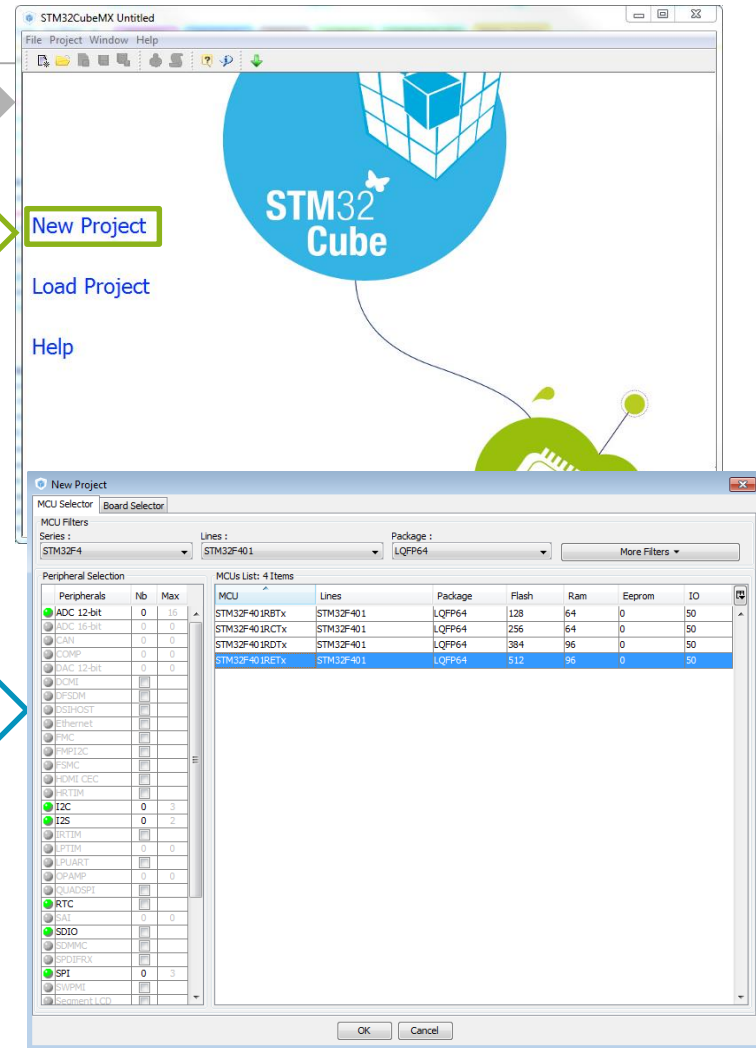


11 June 2016

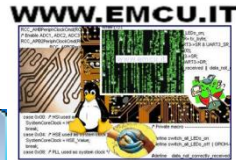


Create new project using CubeMX

- Run CubeMX tool
- Start **new project**
 - Click “New Project” desktop shortcut, or
 - Go to “Menu->File->New Project”
- Filter:
 - Series: STM32F4
 - Line: STM32F401
 - Package: LQFP64
- Select: **STM32F401RE**



Configure debug interface



STM32CubeMX Untitled*: STM32F401RETx

File Project Pinout Window Help

Pinout Clock Configuration Configuration Power Consumption Calculator

Go to Pinout settings

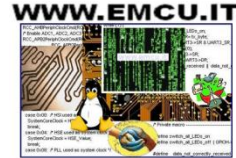
Under SYS peripheral select **SWD interface**

Debug: Serial Wire Debug (SWD)

System Wake-Up

Series	Lines	Mcu	Package	Required Peripherals
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RBTx	LQFP64	None
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RCTx	LQFP64	None
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RDTx	LQFP64	None
<input checked="" type="checkbox"/> STM32F4	STM32F401	STM32F401RETx	LQFP64	None

Configure LSE resonator (32,768 KHz)



STM32CubeMX Untitled*: STM32F401RETx

File Project Pinout Window Help

Pinout Clock Configuration Configuration Power Consumption Calculator

ADC1
CRC
I2C1
I2C2
I2C3
I2S2
I2S3
IWDG
RCC
RTC
SDIO
SPI1
SPI2

High Speed Clock (HSE) Disable
Low Speed Clock (LSE) Crystal/Ceramic Res...
Master Clock Output 1
Master Clock Output 2
Audio Clock Input (I2S_CKIN)

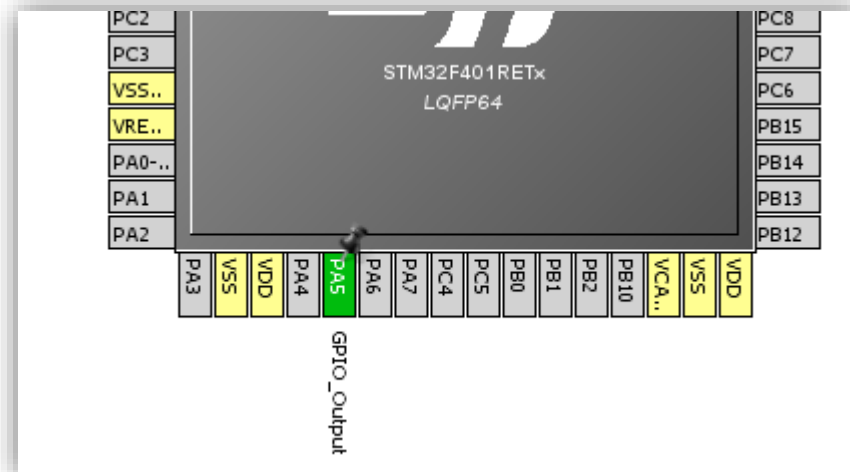
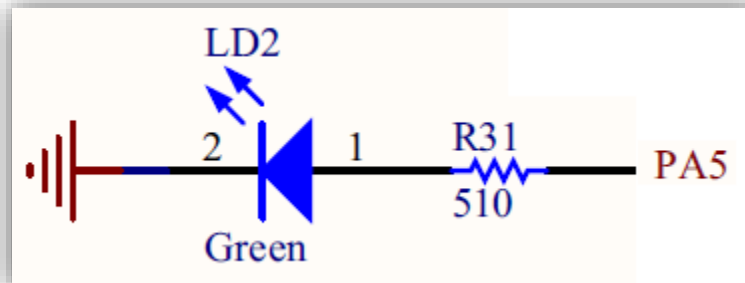
STM32F401RETx
LQFP64

MCUs Selection Output

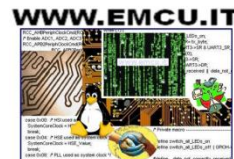
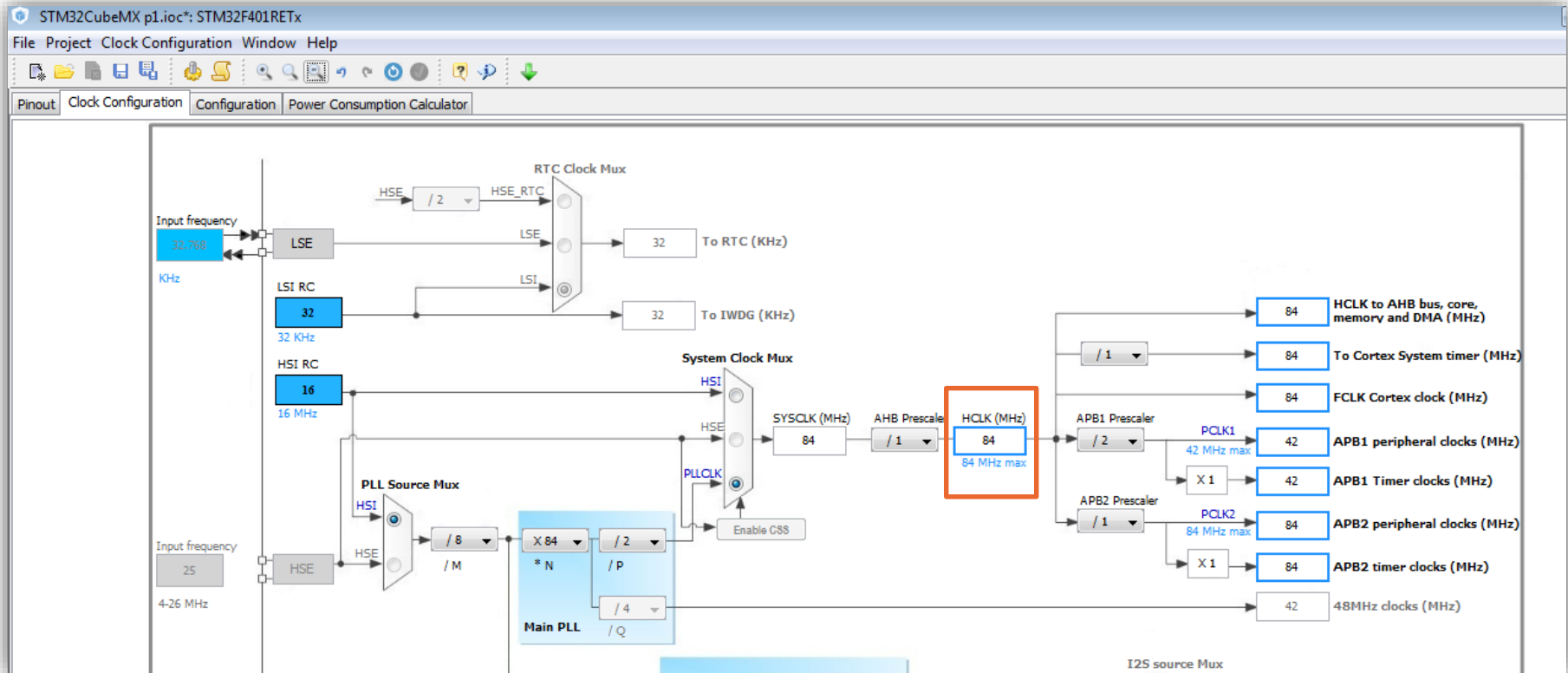
Series	Lines	Mcu	Package	Required Peripherals
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RBTx	LQFP64	None
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RCTx	LQFP64	None
<input type="checkbox"/> STM32F4	STM32F401	STM32F401RDTx	LQFP64	None
<input checked="" type="checkbox"/> STM32F4	STM32F401	STM32F401RETx	LQFP64	None

Configure GPIO for LED toggling

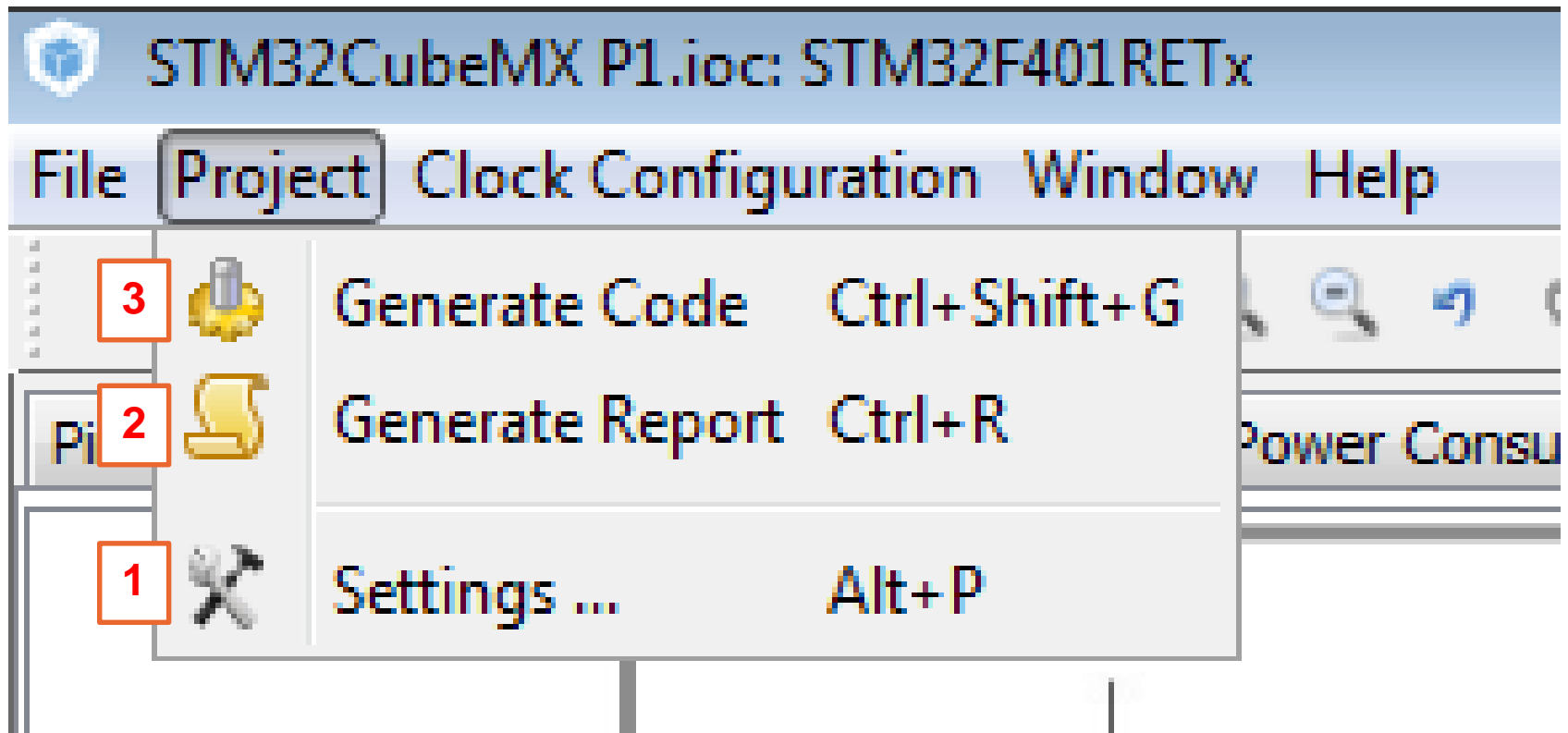
Configure LED pin as GPIO_Output



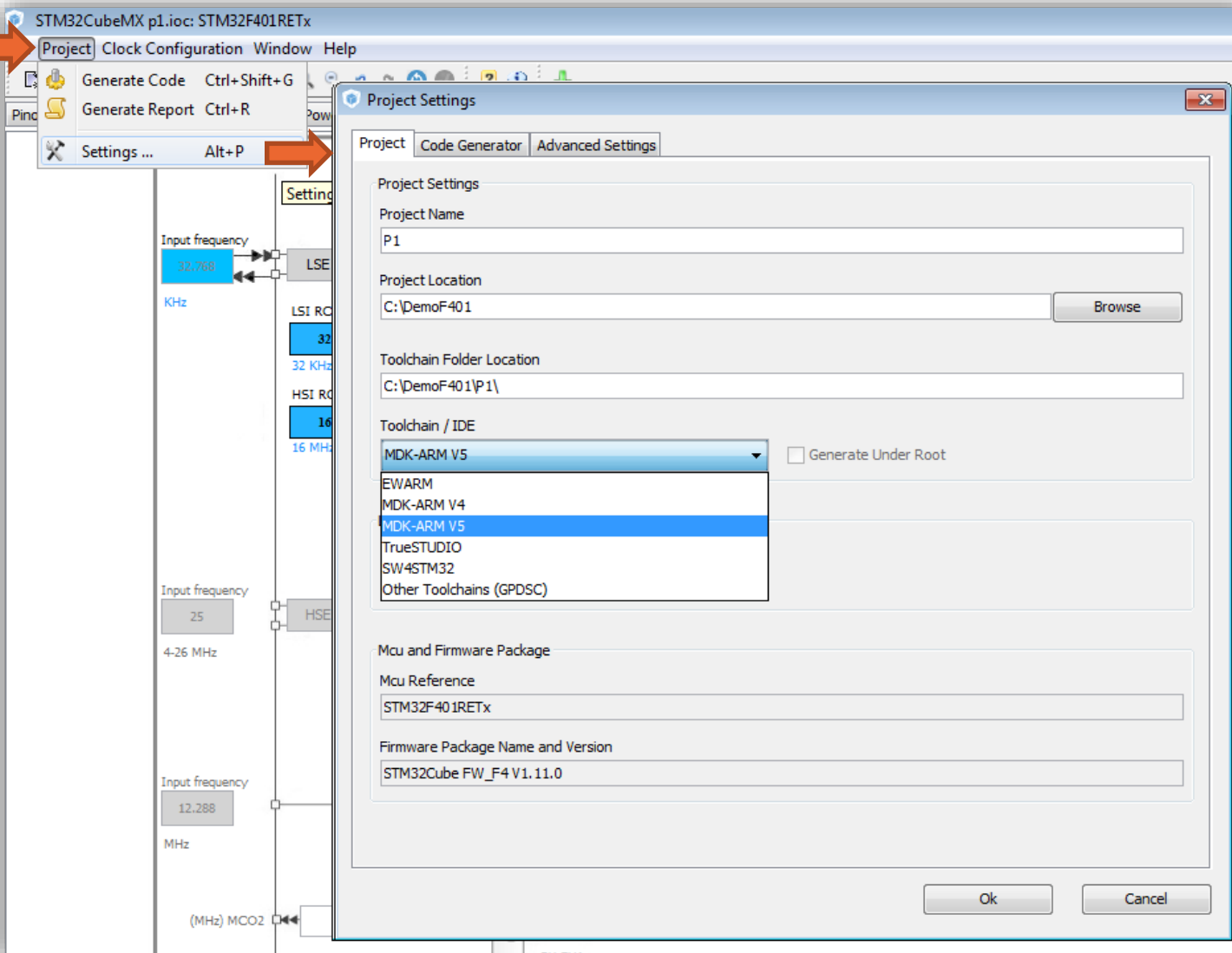
Clock configuration



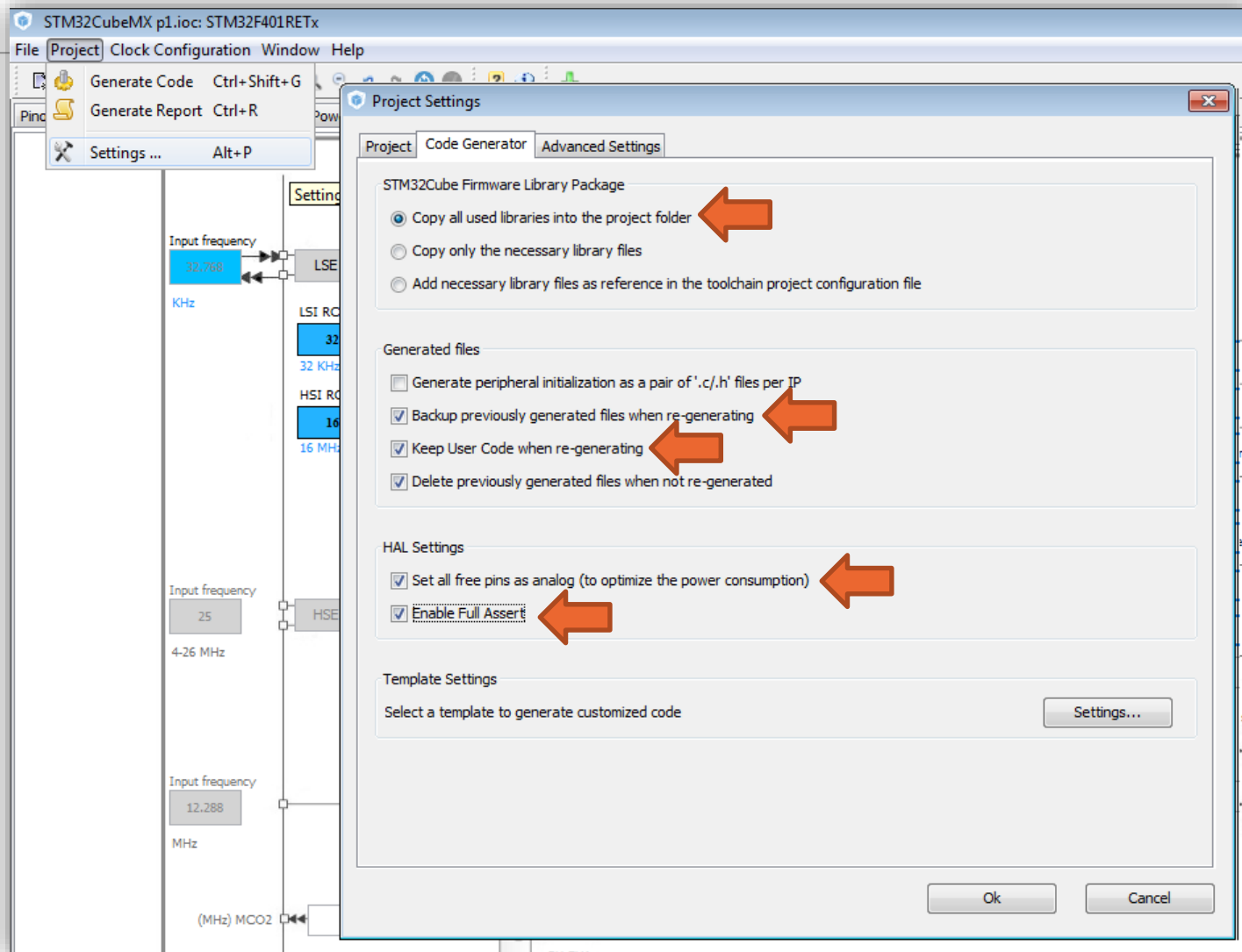
CubeMX generate the code for some GUI 1/3



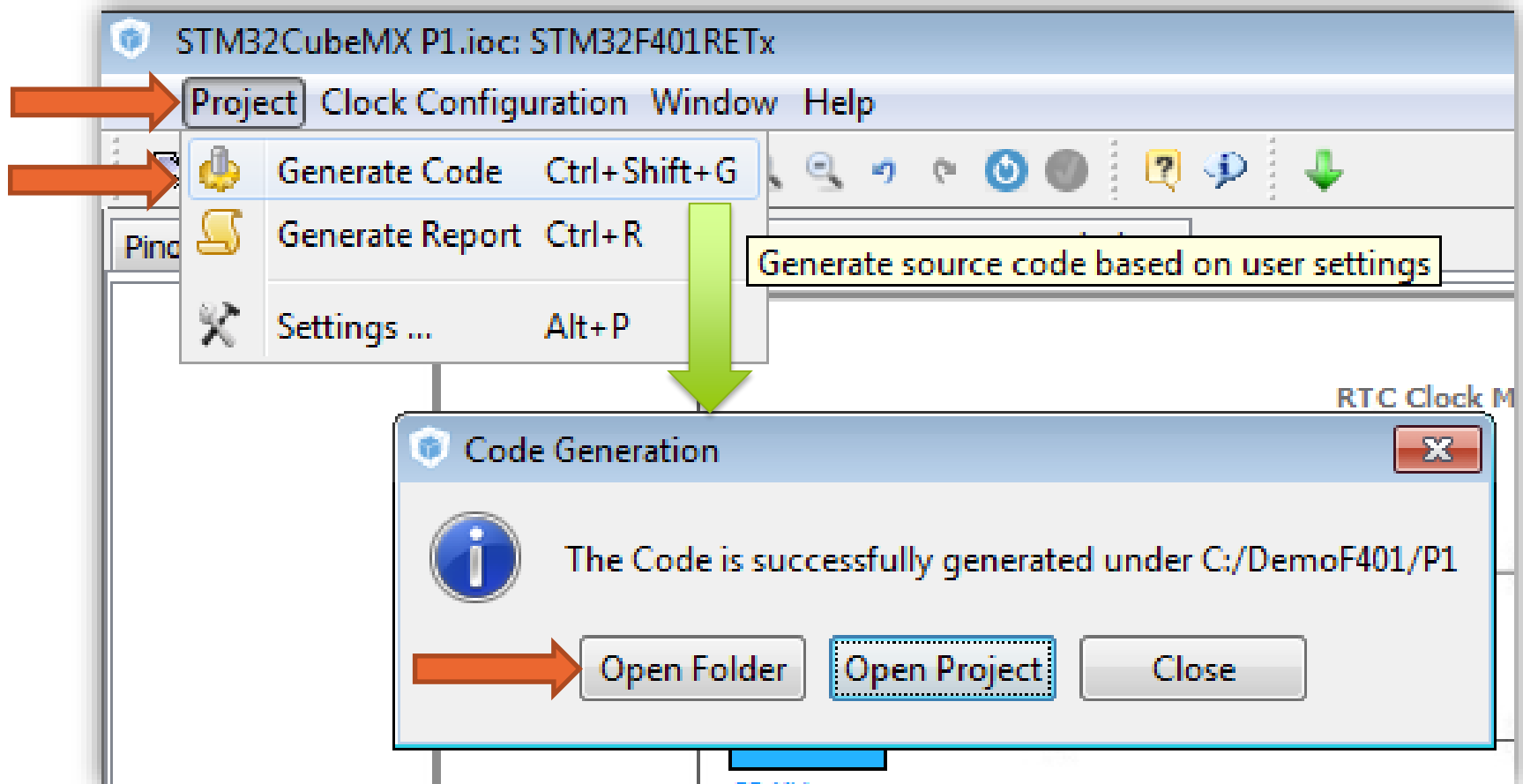
CubeMX generate the code for some GUI 2/3



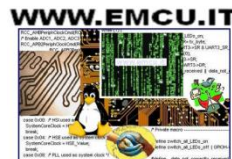
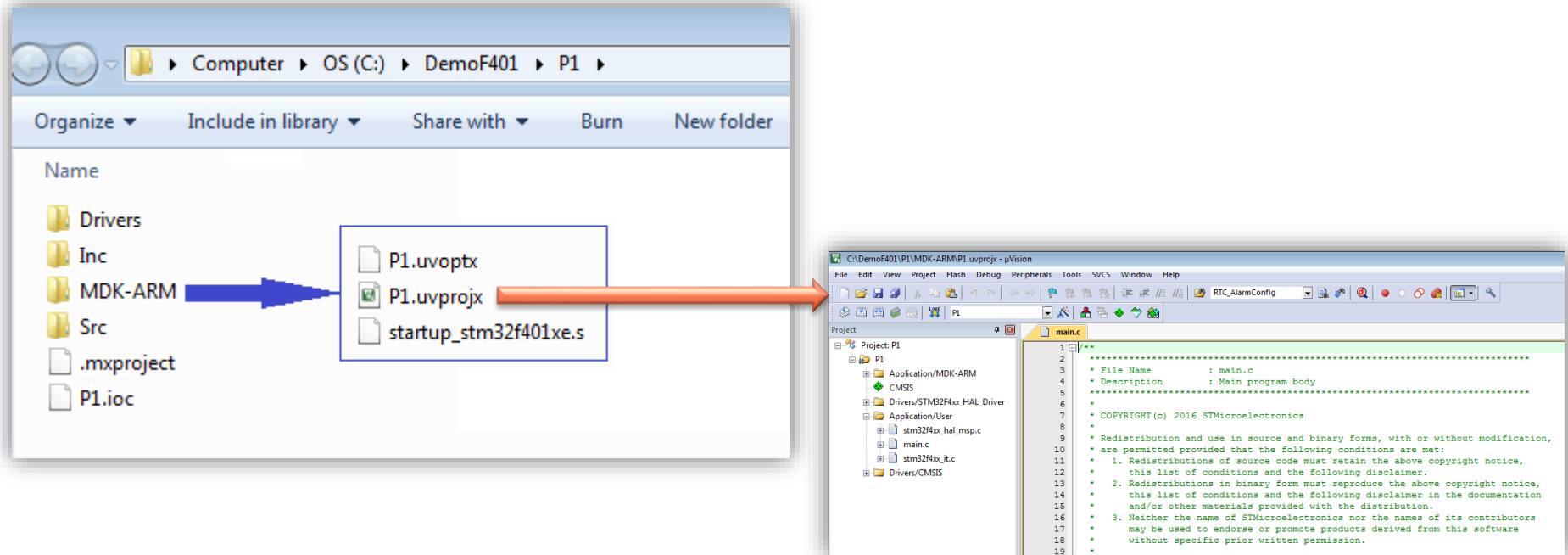
CubeMX generate the code for some GUI 3/3



CubeMX generate the code 1/3



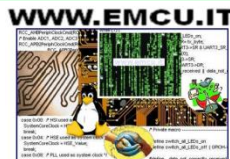
CubeMX generate the code 2/3



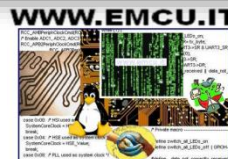
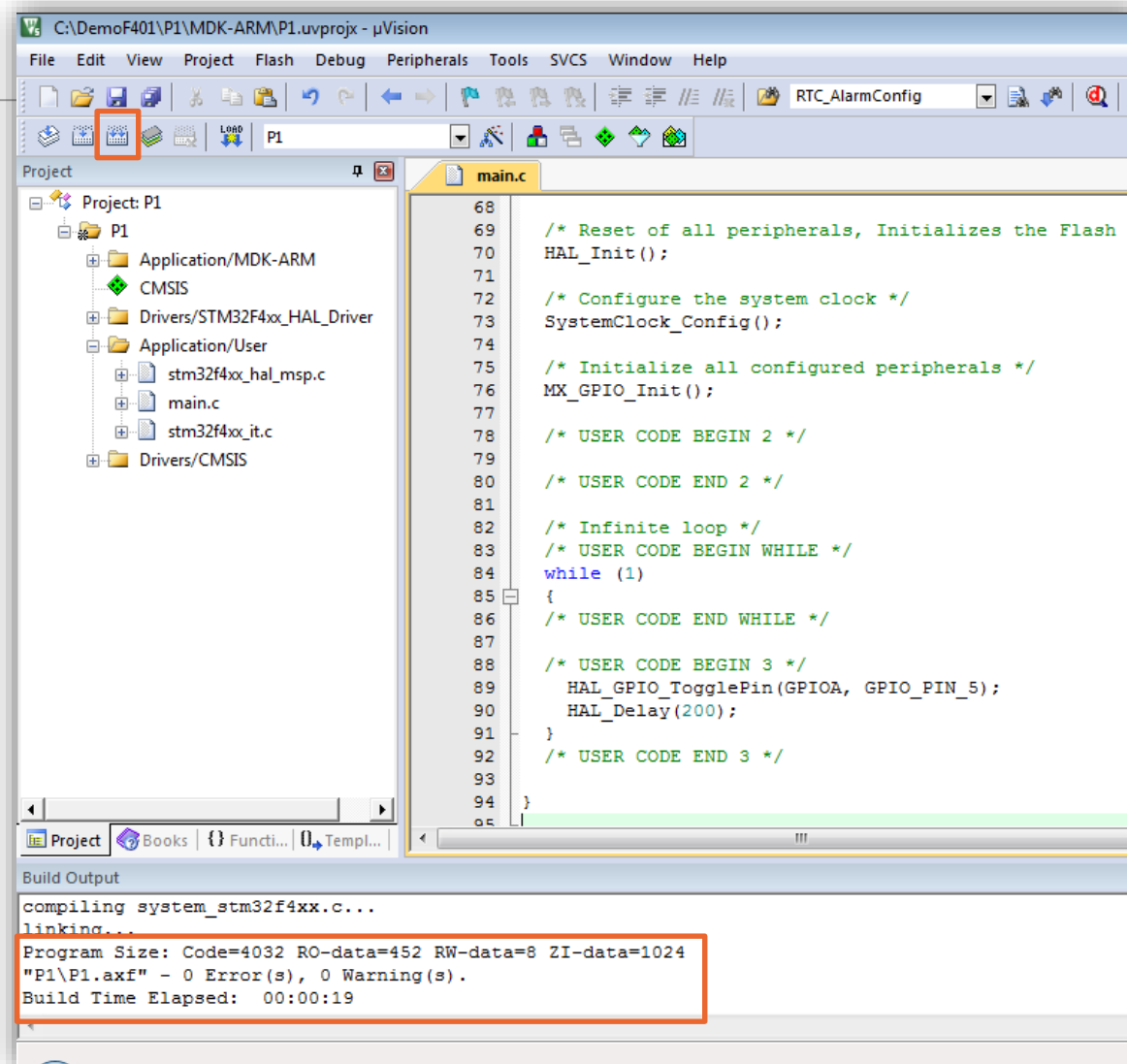
CubeMX add code for flashing LEDs

```
68
69  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
70  HAL_Init();
71
72  /* Configure the system clock */
73  SystemClock_Config();
74
75  /* Initialize all configured peripherals */
76  MX_GPIO_Init();
77
78  /* USER CODE BEGIN 2 */
79
80  /* USER CODE END 2 */
81
82  /* Infinite loop */
83  /* USER CODE BEGIN WHILE */
84  while (1)
85  {
86    /* USER CODE END WHILE */
87
88    /* USER CODE BEGIN 3 */
89    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
90    HAL_Delay(200);
91  }
92  /* USER CODE END 3 */
93
94 }
```

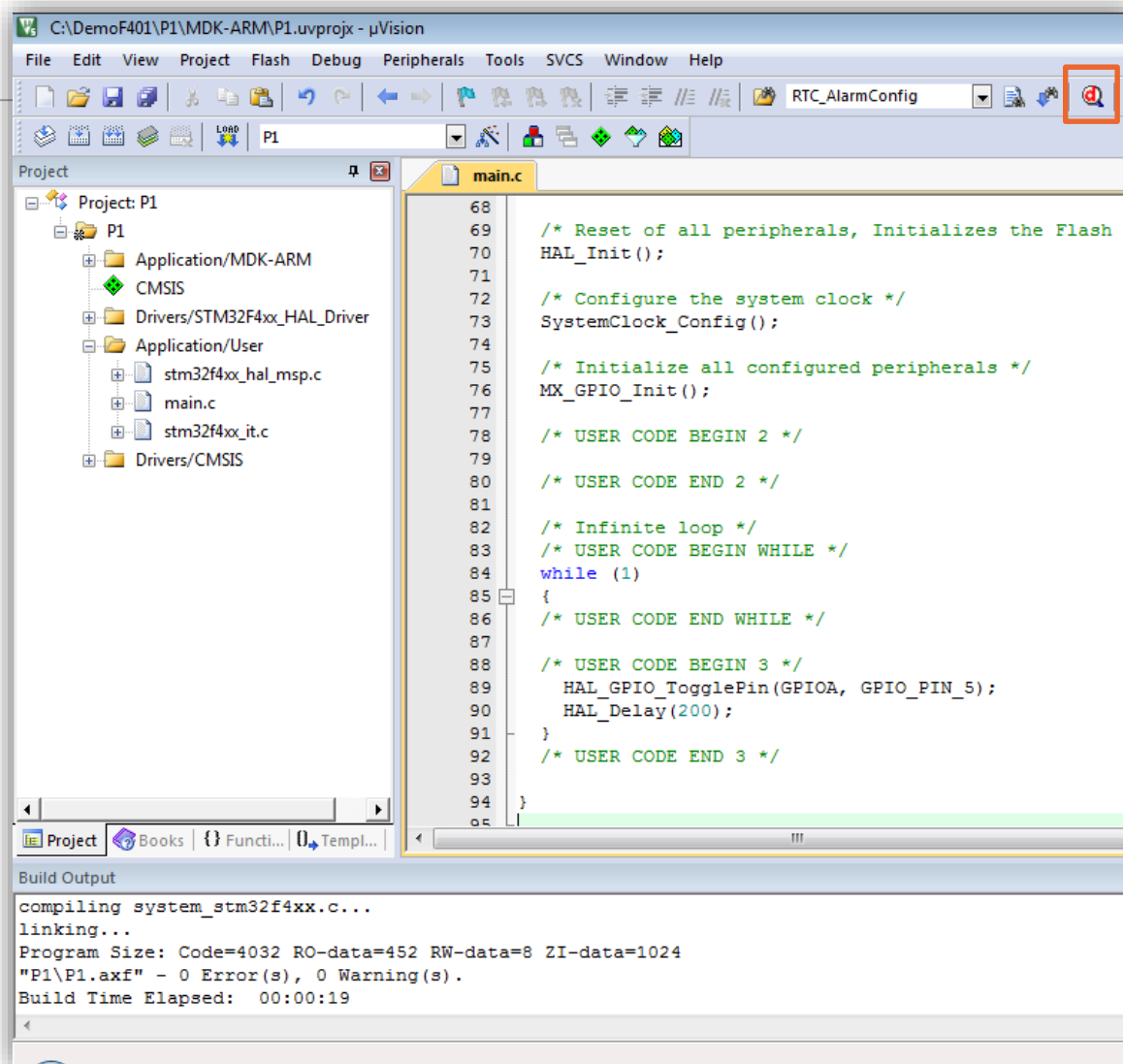
See the:
UM1725 - Description
of STM32F4 HAL
drivers



CubeMX compile and debug – 1/3



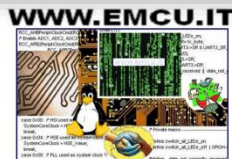
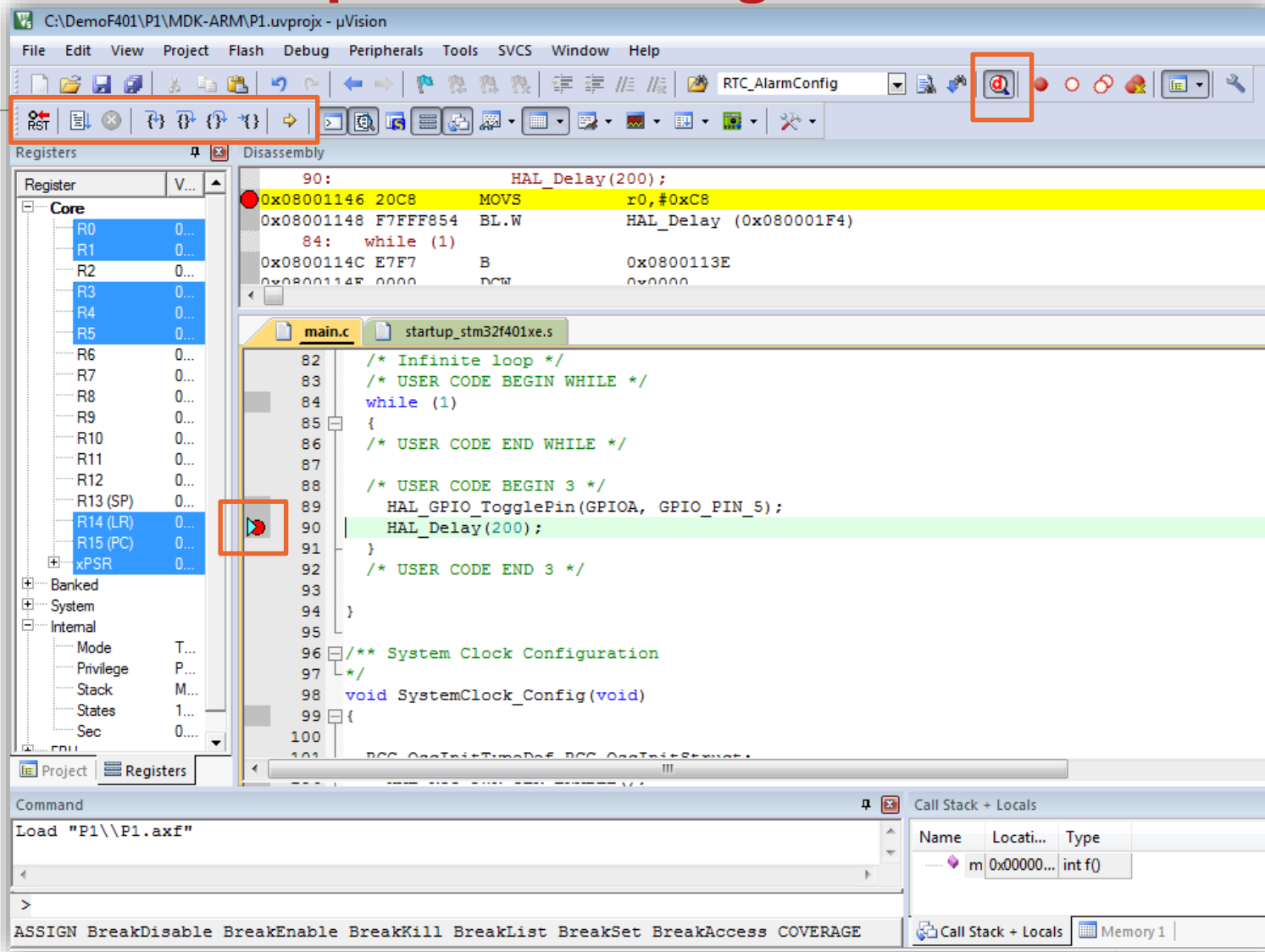
CubeMX compile and debug – 2/3



WWW.EMCU.IT



CubeMX compile and debug – 3/3





Thank you.

WWW.EMCU.IT



42



11 June 2016



life.augmented

