

# **Lesson 7 - UART and Embedded C**

Norman McEntire  
[norman.mcentre@gmail.com](mailto:norman.mcentre@gmail.com)

# Contents

- Introduction to UART and Embedded C
- STM32CubeMX and UART Code Generation
- Tour of UART
- TrueStudio and UART
  - UART Transmit and Receive

[https://www.st.com/content/ccc/resource/technical/document/user\\_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf](https://www.st.com/content/ccc/resource/technical/document/user_manual/63/a8/8f/e3/ca/a1/4c/84/DM00173145.pdf/files/DM00173145.pdf/jcr:content/translations/en.DM00173145.pdf)



## UM1884 User manual

### Description of STM32L4/L4+ HAL and low-layer drivers

---

#### Introduction

STMCube™ is STMicroelectronics's original initiative to ease developers' life by reducing development efforts, time and cost. STM32Cube covers the STM32 portfolio.

STM32Cube Version 1.x includes:

- The STM32CubeMX, a graphical software configuration tool that allows generating C initialization code using graphical wizards.
- A comprehensive embedded software platform, delivered per series (such as STM32CubeL4 for STM32L4 series and STM32L4+ series)
  - The STM32Cube Hardware Abstraction Layer (HAL), an STM32 abstraction layer embedded software ensuring maximized portability across the STM32 portfolio. The HAL is available for all peripherals.
  - The low-layer APIs (LL) offering a fast light-weight expert-oriented layer which is closer to the hardware than the HAL. The LL APIs are available only for a set of peripherals.

# Key Sections From Manual

<b>68</b>	<b>HAL UART Generic Driver.....</b>	<b>1029</b>
68.1	UART Firmware driver registers structures .....	1029
68.1.1	UART_InitTypeDef .....	1029
68.1.2	UART_AdvFeatureInitTypeDef.....	1030
68.1.3	__UART_HandleTypeDef.....	1030
68.2	UART Firmware driver API description .....	1032
68.2.1	How to use this driver .....	1032
68.2.2	Initialization and Configuration functions.....	1033
68.2.3	IO operation functions .....	1034
68.2.4	Peripheral Control functions .....	1034
68.2.5	Peripheral State and Error functions .....	1035
68.2.6	Detailed description of functions .....	1035
68.3	UART Firmware driver defines .....	1045
68.3.1	UART .....	1045

# Introduction to UART and Embedded C

# 28.2.1 UART Peripheral Features

- UART = Universal Async Receiver Transmitter
- Example: UART4 on our STM32 Discovery Board connected to Arduino D0/RX and D1/TX

# 68.2.1 How to use driver - Part 1

- Step 1. Declare a `UART_HandleTypeDef`
- Step 2. Init UART low-level resources by implementing `HAL_UART_MspInit()`
  - Enable USARTx interface clock
  - Enable interrupt handler if needed

**NVIC** =  
Nested  
Vector  
Interrupt  
Controller

# HAL UART Data Types

Drivers/STM32L4xx\_HAL\_Driver/Inc/stm32l4xx\_hal\_uart.h

# UART\_InitTypeDef

- ```
typedef struct {
    uint32_t BaudRate;
    uint32_t WordLength;
    uint32_t StopBits;
    uint32_t Parity;
    uint32_t Mode;
    uint32_t HwFlowCtl;
    uint32_t OverSampling;
    uint32_t OneBitSampling;
} UART_InitTypeDef
```

# UART\_AdvFeatureInitTypeDef

- ```
typedef struct {
    uint32_t AdvFeatureInit;
    uint32_t TxPinLevelInvert;
    uint32_t RxPinLevelInvert;
    uint32_t DataInvert;
    uint32_t Swap;
    uint32_t DMADisableonRxError;
    uint32_t AutoBaudRateEnable;
    uint32_t AutoBaudRateMode;
    uint32_t MSBFIRST;
} UART_AdvFeatureInitTypeDef
```

# HAL\_UART\_StateTypeDef

- HAL\_UART\_STATE\_RESET
- HAL\_UART\_STATE\_READY
- HAL\_UART\_STATE\_BUSY
- HAL\_UART\_STATE\_BUSY\_TX
- HAL\_UART\_STATE\_BUSY\_RX
- HAL\_UART\_STATE\_BUSY\_TX\_RX
- HAL\_UART\_STATE\_TIMEOUT
- HAL\_UART\_STATE\_ERROR

# UART\_ClockSourceTypeDef

- UART\_CLOCKSOURCE\_PCLK1
- UART\_CLOCKSOURCE\_PCLK2
- UART\_CLOCKSOURCE\_HSI
- UART\_CLOCKSOURCE\_SYSCLK
- UART\_CLOCKSOURCE\_LSE
- UART\_CLOCKSOURCE\_UNDEFINED

# UART\_HandleTypeDef

- ```
typedef struct {
    USART_TypeDef *Instance; //Base Address
    UART_InitTypeDef Init;
    UART_AdvFeatureInitTypeDef AdvancedInit;
    uint8_t *pTxBuffPtr;
    uint16_t TxXferSize;
    __IO uint16_t TxXferCount;
    uint8_t *pRxBuffPtr;
    uint16_t RxXferSize;
    __IO uint uint16_t RxXferCount;
    uint16_t Mask;
    ...
} UART_HandleTypeDef
```

# UART\_STOPBITS

- UART\_STOPBITS\_0\_5
- UART\_STOPBITS\_1
- UART\_STOPBITS\_1\_5
- UART\_STOPBITS\_2

# UART\_PARITY

- UART\_PARITY\_NONE
- UART\_PARITY EVEN
- UART\_PARITY ODD

# UART\_HWCONTROL

- UART\_HWCONTROL\_NONE
- UART\_HWCONTROL\_RTS
- UART\_HWCONTROL\_CTS
- UART\_HWCONTROL\_RTS\_CTS

# UART\_MODE

- UART\_MODE\_RX
- UART\_MODE\_TX
- UART\_MODE\_TX\_RX

# HAL UART Functions

# HAL UART

## Init and Delinit

- HAL\_UART\_Init()
- HAL\_UART\_Delinit()
- HAL\_UART\_MspInit()
- HAL\_UART\_MspDelinit()

# HAL UART - I/O Functions

## Part 1

- HAL\_UART\_Transmit()
- HAL\_UART\_Receive()
- HAL\_UART\_Transmit\_IT()
- HAL\_UART\_Receive\_IT()
- HAL\_UART\_Transmit\_DMA()
- HAL\_UART\_Receive\_DMA()
- HAL\_UART\_DMAPause()
- HAL\_UART\_DMAResume()
- HAL\_UART\_DMAStop()

# HAL UART - I/O Functions

## Part 2

- HAL\_UART\_Abort()
- HAL\_UART\_AbortTransmit()
- HAL\_UART\_AbortReceive()
- HAL\_UART\_Abort\_IT()
- HAL\_UART\_AbortTransmit\_IT()
- HAL\_UART\_AbortReceive\_IT()
- HAL\_UART\_IRQHandler()

# HAL UART - I/O Functions

## Part 3

- HAL\_UART\_TxCpltCallback()
- HAL\_UART\_TxHalfCpltCallback()
- HAL\_UART\_RxCpltCallback()
- HAL\_UART\_RxHalfCpltCallback()
- HAL\_UART\_ErrorCallback()
- HAL\_UART\_AbortCpltCallback()
- HAL\_UART\_AbortTransmitCpltCallback()
- HAL\_UART\_AbortReceiveCpltCallback()

# HAL UART - Peripheral Control Functions

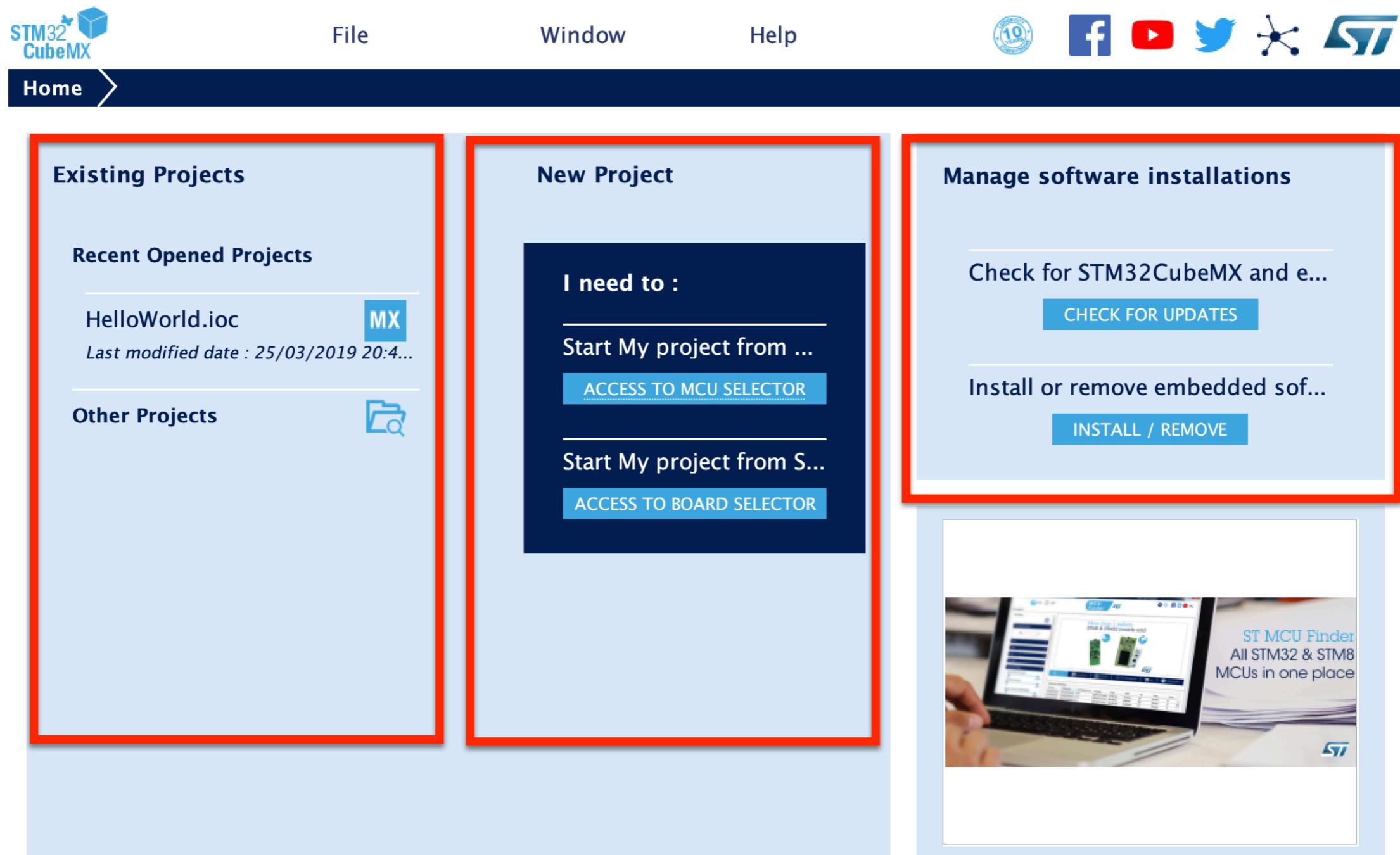
- `UART_SetConfig()`
- `UART_AdvFeatureConfig()`

# HAL UART - Peripheral State and Error Functions

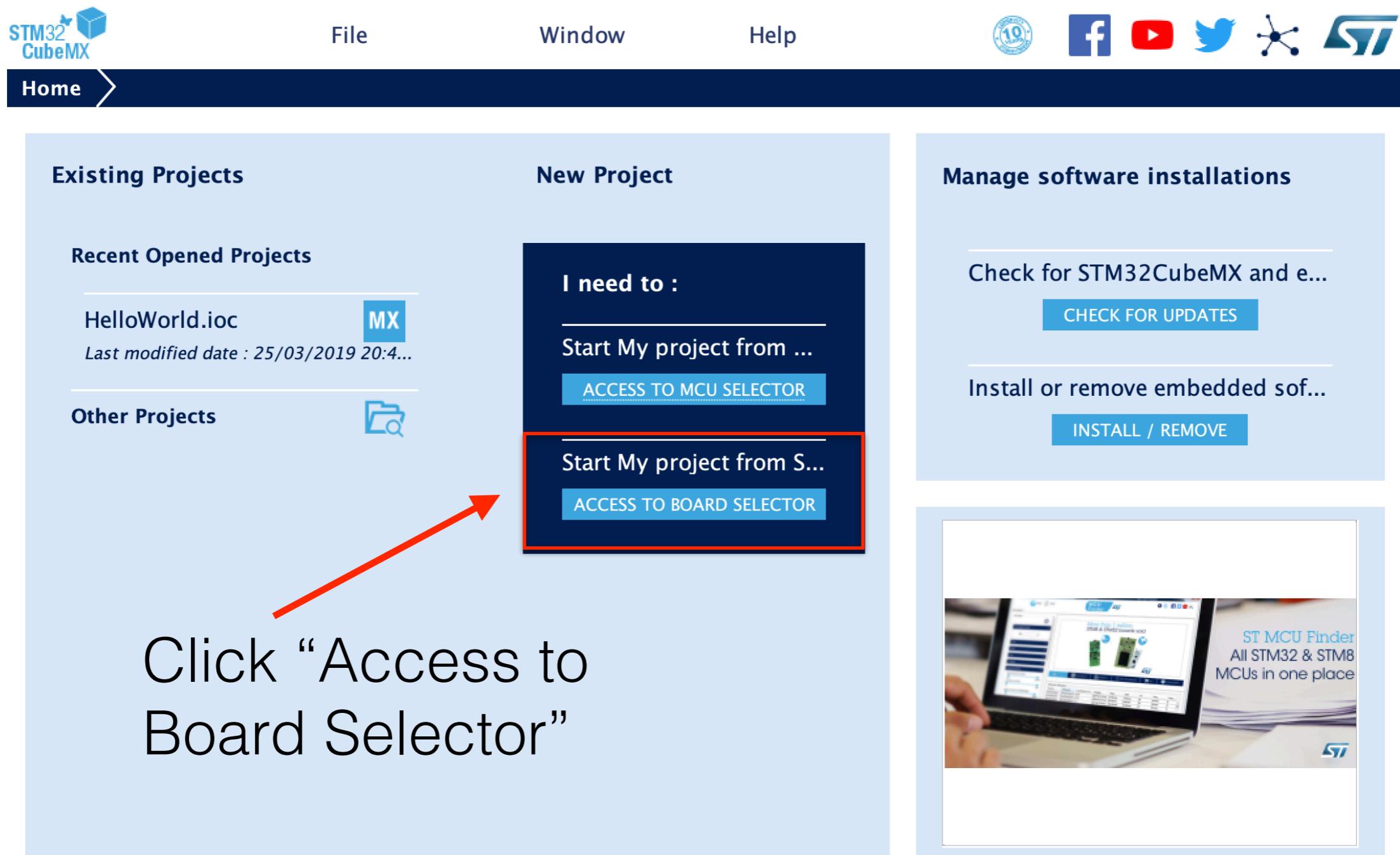
- HAL\_UART\_GetState()
- HAL\_UART\_GetError()

# STM32CubeMX and Generation of UART Code

# Step: Startup STM32CubeMX



# Step: Click on “Access to Board Selector”



# Step: Observe “Part Number Search”

The screenshot shows the STMicroelectronics website interface for searching boards. A red box highlights the 'Part Number Search' input field and its dropdown menu. Below it, another red box highlights the 'Vendor' dropdown menu. The main content area features a banner for the 'New multicore STM32MP1 Series for Industrial and IoT applications'. It includes an image of an STM32MP1 chip, a Linux logo, and the text 'OpenSTLinux Distribution'. Below the banner is a table titled 'Boards List: 107 items' showing two rows of board details.

| * | Overview | Part No          | Type      | Marketing Status | Unit Price (US\$) | Mounted Device                |
|---|----------|------------------|-----------|------------------|-------------------|-------------------------------|
| ☆ |          | 32F0308DISCOVERY | Discovery | Active           | 8.9               | <a href="#">STM32F030R8Tx</a> |
| ☆ |          | 32F072BDISCOVERY | Discovery | Active           | 10.4              | <a href="#">STM32F072RBTx</a> |

# Step: Enter "B-L475E-IOT01A"

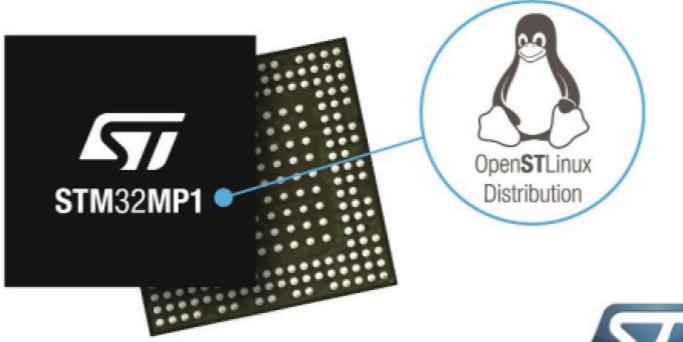
MCU Selector | Board Selector

Board Filters

- Part Number Search: B-L475E-IOT01A (highlighted with a red box)
- Vendor: STMicroelectronics
- Type: Discovery
- MCU Series: STM32L4
- Other: Price = 53.0, Oscillator Freq. = 0 (MHz)
- Peripheral

Features | Large Picture | Docs & Resources | Datasheet | Buy | Start Project

New multicore STM32MP1 Series for Industrial and IoT applications



STM32MP1

OpenSTLinux Distribution

Boards List: 1 item

| * | Overview                                                                             | Part No        | Type      | Marketing Status | Unit Price (US\$) | Mounted Device |
|---|--------------------------------------------------------------------------------------|----------------|-----------|------------------|-------------------|----------------|
| ★ |  | B-L475E-IOT01A | Discovery | Active           | 53.0              | STM32L475VGTx  |

Step: Click on Image, and observe “Features”

MCU Selector Board Selector

Board Filters

- 
- 
- 
- 

Part Number Search

B-L475E-IOT01A

Vendor

Check/Uncheck All

STMicroelectronics

Type

Check/Uncheck All

Discovery

MCU Series

Check/Uncheck All

STM32L4

Other

Price = 53.0

Oscillator Freq. = 0 (MHz)

Peripheral

Features Large Picture Docs & Resources Datasheet Buy Start Project

**B-L475E-IOT01A**



**STMicroelectronics B-L475E-IOT01A IOT Discovery Board Support and Examples**

**ACTIVE** Active  
Product is in mass production

Unit Price (US\$) : 53.0

Mounted device: [STM32L475VGTx](#)

The B-L475E-IOT01A Discovery kit for IoT node allows users to develop applications with direct connection to cloud servers. The Discovery kit enables a wide diversity of applications by exploiting low-power communication, multiway sensing and ARM Cortex -M4 core-based STM32L4 Series features. The support for Arduino Uno V3 and PMOD connectivity provides unlimited expansion capabilities with a large choice of specialized add-on boards.

**Features**

- On-board ST-LINK/V2-1
- Supply through ST-Link USB
- USB OTG(Full speed) with micro AB Connector

Boards List: 1 item

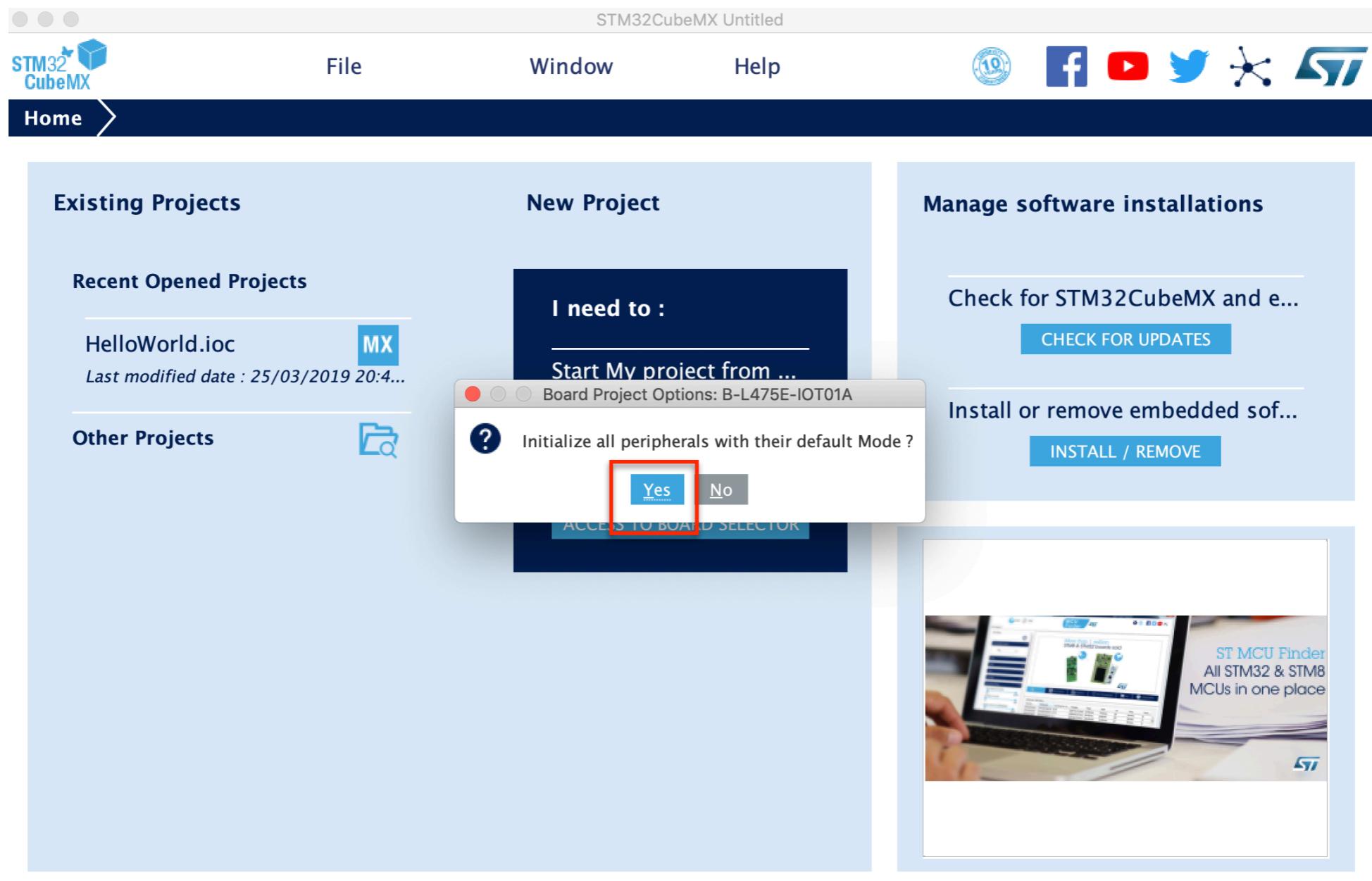
| *                                                                                    | Overview                       | Part No | Type      | Marketing Status | Unit Price (US\$) | Mounted Device                |
|--------------------------------------------------------------------------------------|--------------------------------|---------|-----------|------------------|-------------------|-------------------------------|
|  | <a href="#">B-L475E-IOT01A</a> |         | Discovery | Active           | 53.0              | <a href="#">STM32L475VGTx</a> |

# Step: Click on “Start Project”

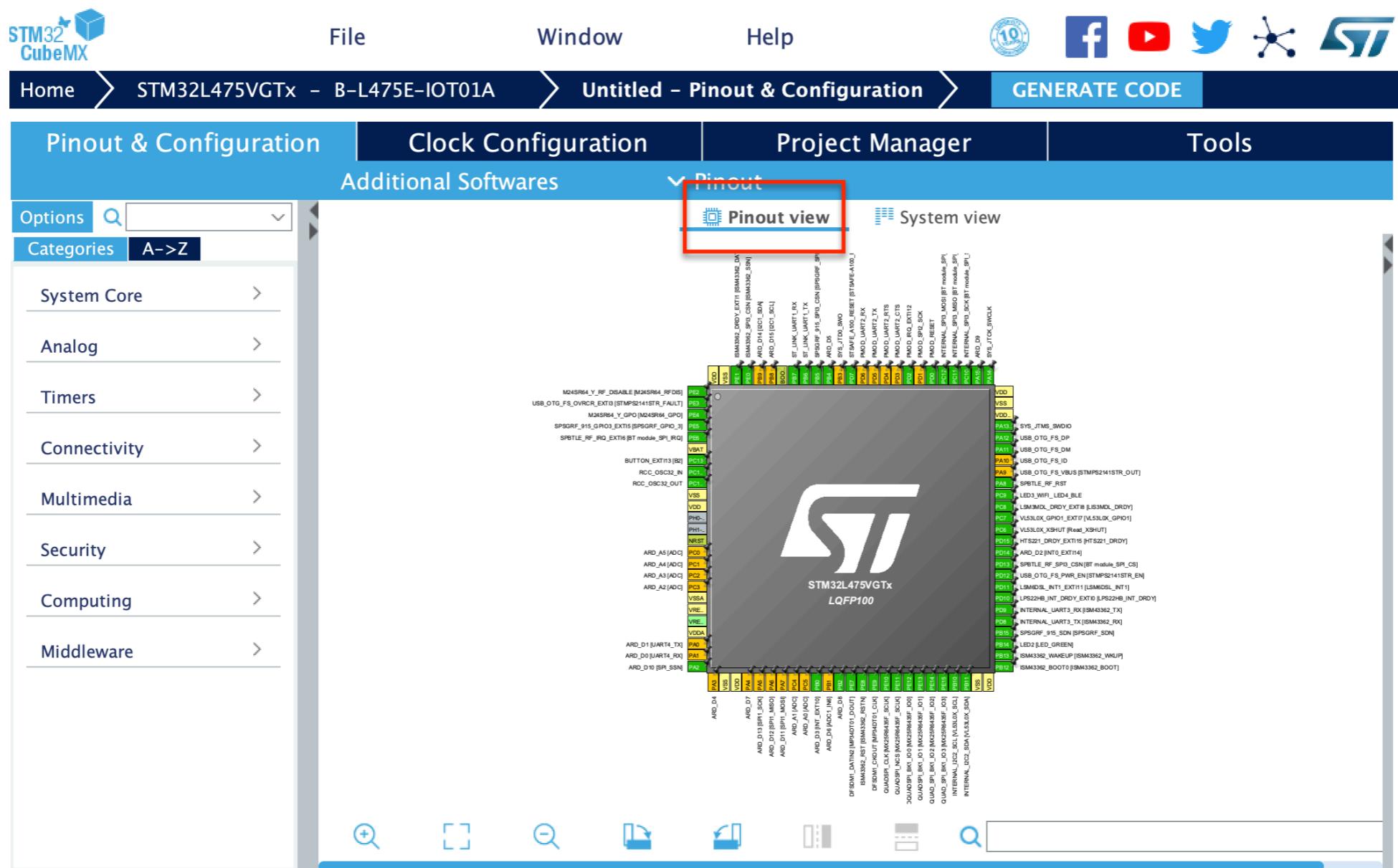
The screenshot shows a web-based board selector interface. On the left, there are several filter sections: 'Board Filters' (Part Number Search set to B-L475E-IOT01A, with icons for star, favorite, file, and search); 'Vendor' (Check/Uncheck All, STMMicroelectronics selected); 'Type' (Check/Uncheck All, Discovery selected); 'MCU Series' (Check/Uncheck All, STM32L4 selected); 'Other' (Price = 53.0, Oscillator Freq. = 0 (MHz)); and 'Peripheral'. The main content area has tabs for 'Features', 'Large Picture', 'Docs & Resources' (which is active and underlined), 'Datasheet', 'Buy', and a prominent 'Start Project' button, all within a red box. Below this, the board is identified as 'B-L475E-IOT01A'. It lists 'Data brief' (DB3143) and 'User manual' (UM2153). At the bottom, a 'Boards List: 1 item' table is shown:

| * | Overview | Part No        | Type      | Marketing Status | Unit Price (US\$) | Mounted Device |
|---|----------|----------------|-----------|------------------|-------------------|----------------|
| ☆ |          | B-L475E-IOT01A | Discovery | Active           | 53.0              | STM32L475VGTx  |

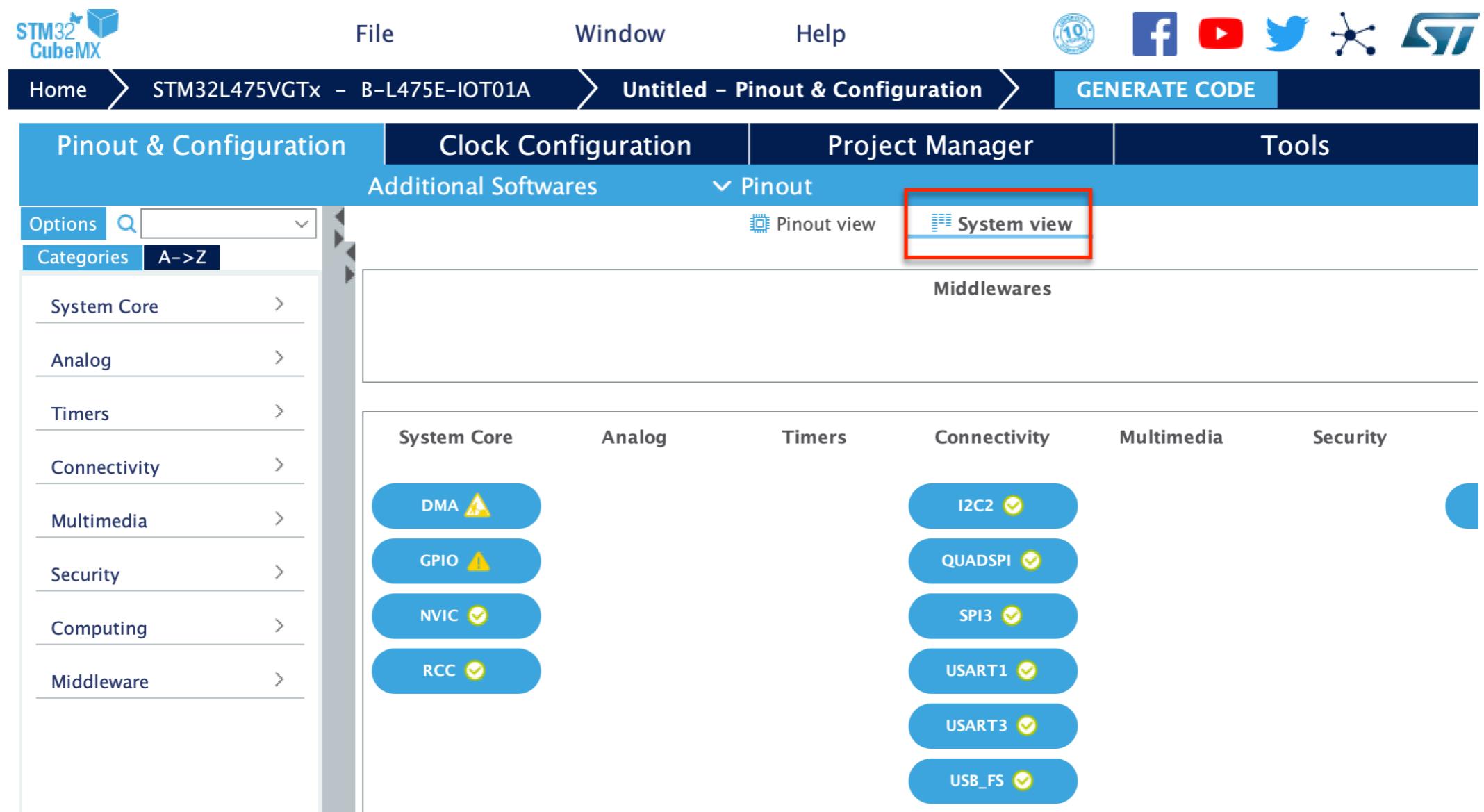
# Step: Click on “Yes” (Initialize all ...with default mode)



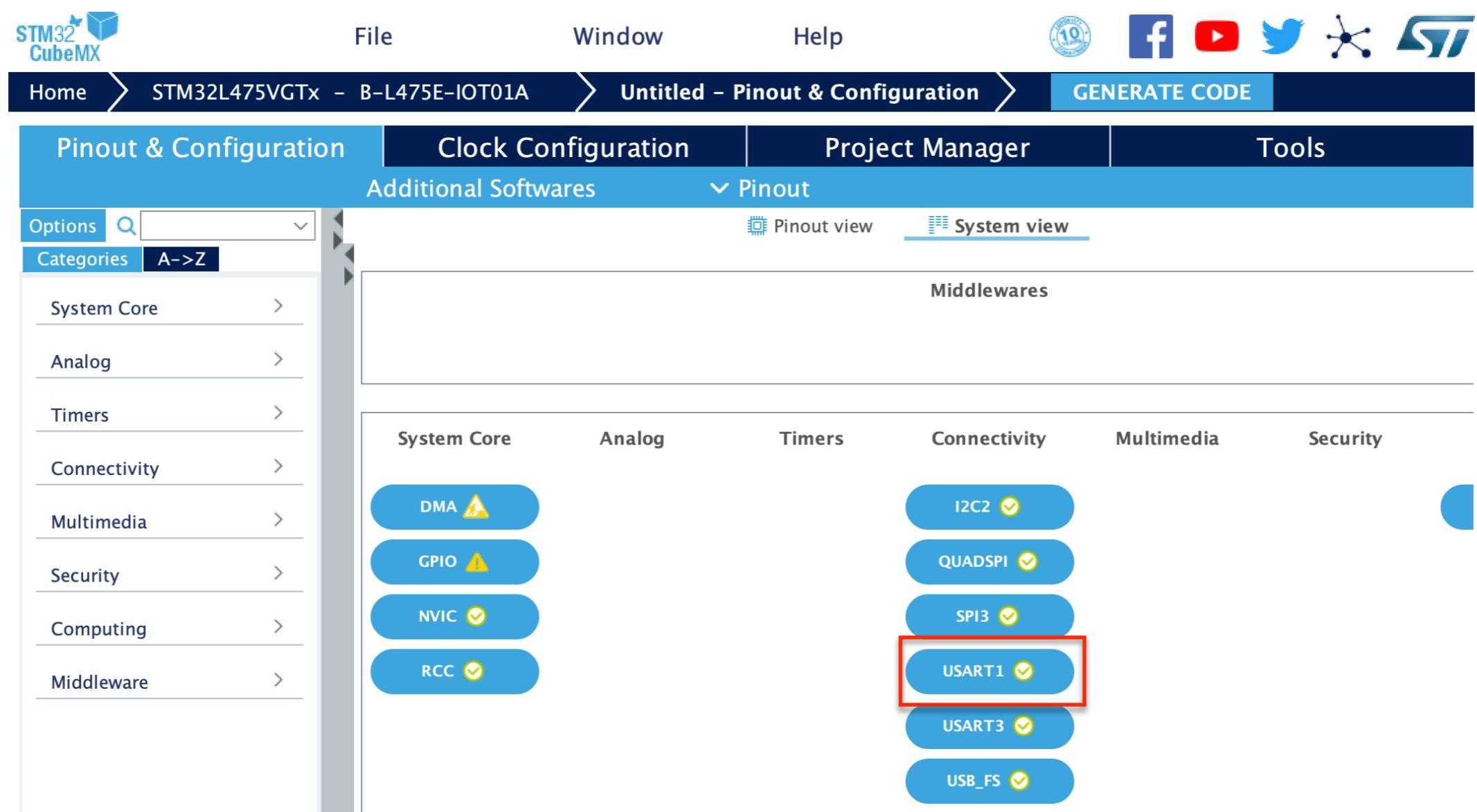
# Step: Observe “Pinout View”



# Step: Select “System View”



# Step: Select “USART1”



# USART1 Options - Part 1

The screenshot shows a software interface with several tabs at the top: Pinout & Configuration, Clock Configuration, Project Manager, and Tools. The Pinout & Configuration tab is active.

**Pinout & Configuration Tab:**

- Options: Q
- Categories: A-Z
- Listed components include CAN1, FMC, I2C1, I2C2, I2C3, IRTIM, LPUART1, QUADSPI, SDMMC1, SPI1, SPI2, SPI3, SWPMI1, UART4, USART5, USART1 (selected), USART2, USART3, and USB\_OTG\_FS.

**Clock Configuration Tab:**

Additional Softwares: USART1 Mode and Configuration

Mode: Asynchronous

Hardware Flow Control (RS232): Disable

Hardware Flow Control (RS485):

**Project Manager Tab:**

Pinout view (disabled)

System view (enabled)

Middlewares

System Core: DMA (yellow warning icon), GPIO (yellow warning icon), NVIC (green checkmark), RCC (green checkmark)

Analog: I2C2 (green checkmark), QUADSPI (green checkmark), SPI3 (green checkmark), USART1 (selected, highlighted with a red box), USART3 (green checkmark), USB\_FS (green checkmark)

Timers: None

Connectivity: None

**Tools Tab:**

Pinout view (disabled)

System view (enabled)

Middlewares

System Core: DMA (yellow warning icon), GPIO (yellow warning icon), NVIC (green checkmark), RCC (green checkmark)

Analog: I2C2 (green checkmark), QUADSPI (green checkmark), SPI3 (green checkmark), USART1 (selected, highlighted with a red box), USART3 (green checkmark), USB\_FS (green checkmark)

Timers: None

Connectivity: None

# USART1 Options - Part 2

Configuration

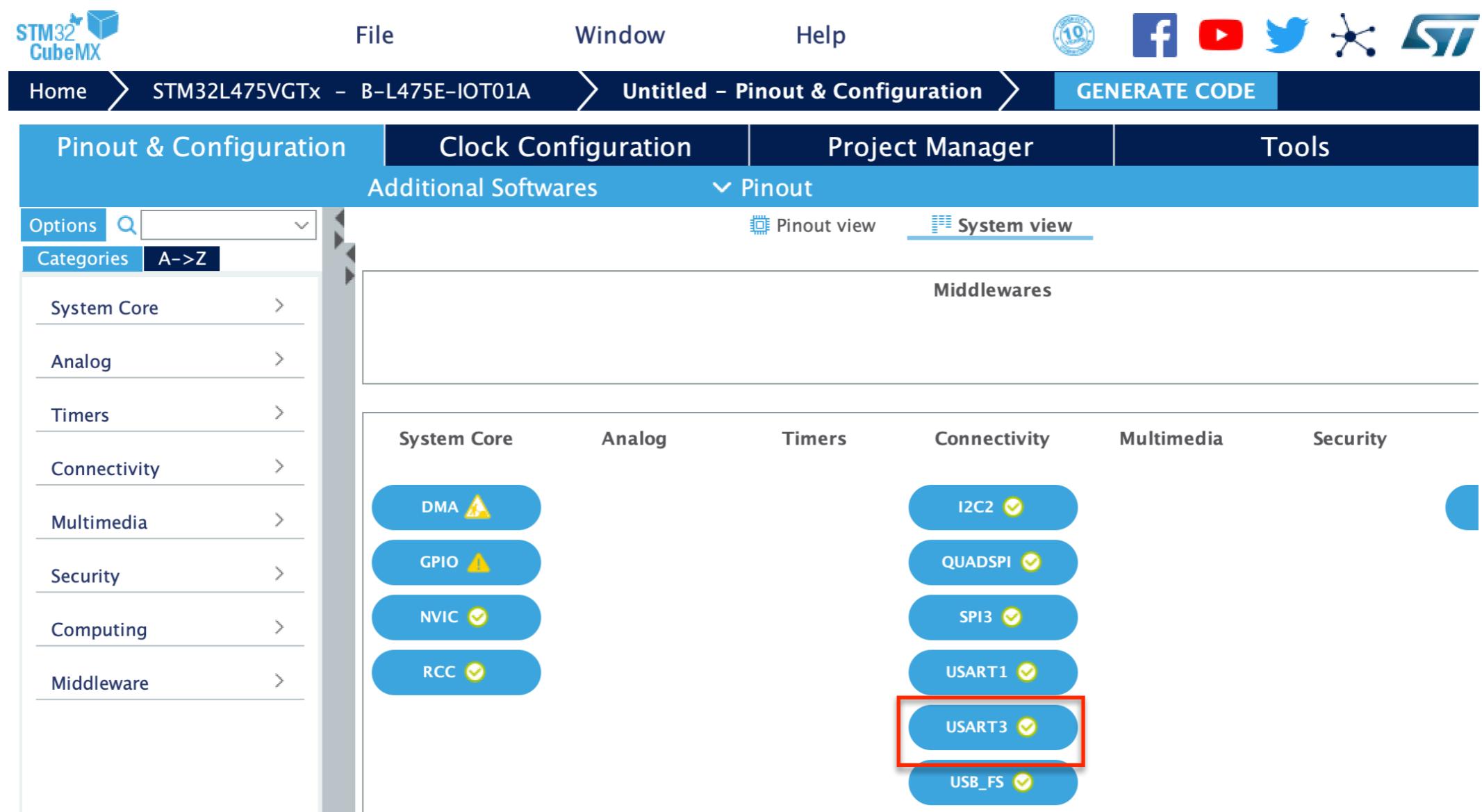
Reset Configuration

Parameter Settings    User Constants    NVIC Settings    DMA Settings    GPIO Settings

Search Signals   Show only Modified Pins

| Pin Na... | Signal on ... | GPIO outp... | GPIO mode     | GPIO Pull-... | Maximum ... | Fast Mode | User Label       | M...                                |
|-----------|---------------|--------------|---------------|---------------|-------------|-----------|------------------|-------------------------------------|
| PB6       | USART1_TX     | n/a          | Alternate ... | No pull-u...  | Very High   | Disable   | ST_LINK_UART1_TX | <input checked="" type="checkbox"/> |
| PB7       | USART1_RX     | n/a          | Alternate ... | No pull-u...  | Very High   | Disable   | ST_LINK_UART1_RX | <input checked="" type="checkbox"/> |

# Step: Select “USART3”



# USART3 Options - Part 1

The screenshot shows a software interface for configuring USART3. The main window is titled "Clock Configuration" and displays "USART1 Mode and Configuration". A red box highlights the "Mode" dropdown set to "Asynchronous" and the "Hardware Flow Control (RS232)" dropdown set to "Disable". Another red box highlights the "Hardware Flow Control (RS485)" checkbox, which is currently unchecked. Below this, there's a "Configuration" section with "Reset Configuration" and several checked settings: DMA Settings, GPIO Settings, Parameter Settings, User Constants, and NVIC Settings. A search bar and a basic parameters table are also visible. To the right, a "Project Manager" tab shows a "Pinout view" and a "System view" (which is selected). The "System view" lists various system components with checkboxes: DMA (unchecked), GPIO (unchecked), NVIC (unchecked), RCC (unchecked), I2C2 (checked), QUADSPI (checked), SPI3 (checked), USART1 (checked), USART3 (unchecked), and USB\_FS (checked). The left sidebar has categories like Analog, Timers, Connectivity, and a list of peripherals including CAN1, FMC, I2C1, I2C2, I2C3, IRTIM, LPUART1, QUADSPI, SDMMC1, SPI1, SPI2, SPI3, SWPMI1, UART4, USART5, USART1, USART2, USART3, and USB\_OTG\_FS. USART1 is currently selected.

# USART3 Options - Part 2

Configuration

Reset Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Search Signals

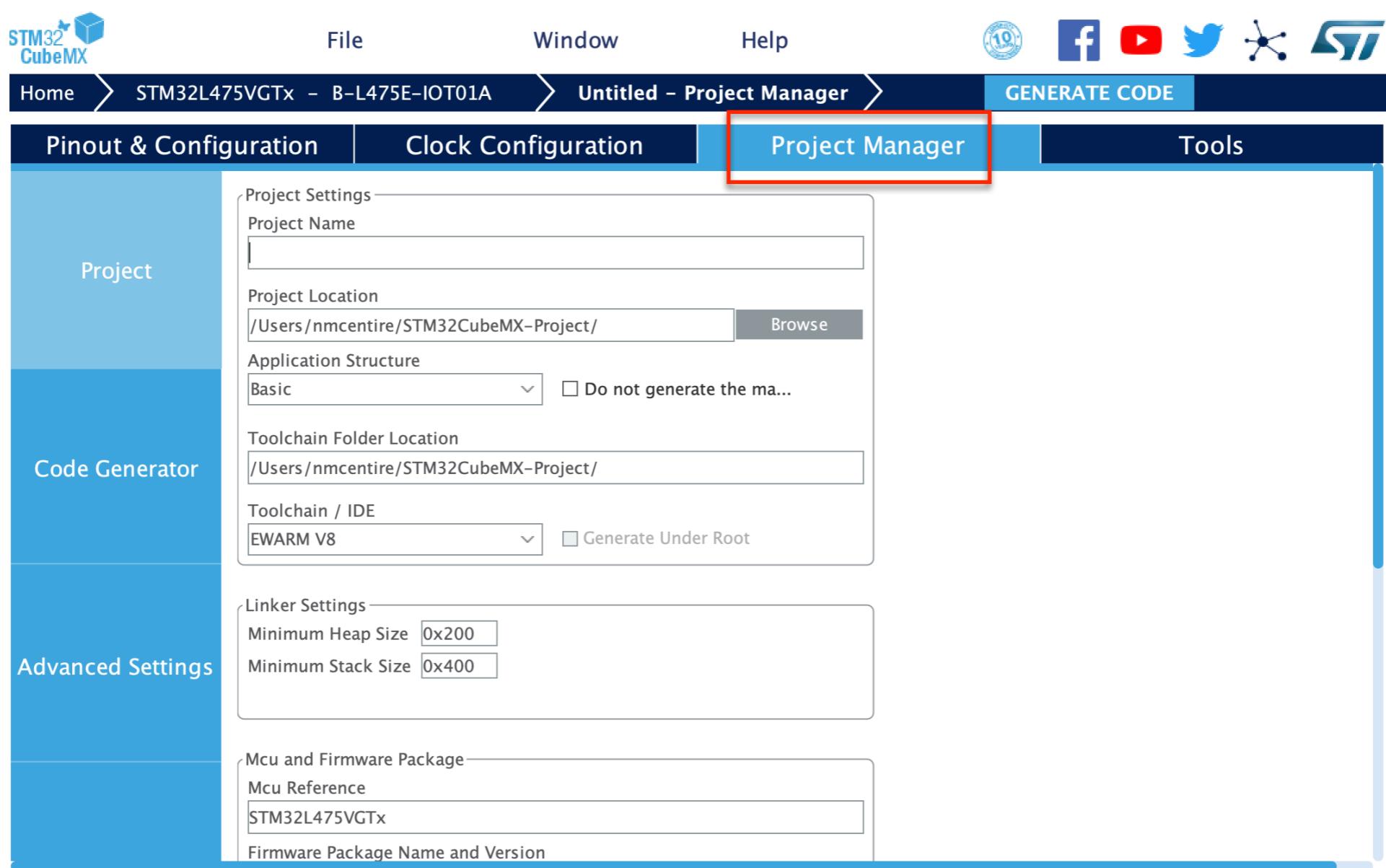
| Pin Name | Signal on Pin | GPIO output le... | GPIO mode         | GPIO Pull-up/... | Maximum out... | Fast M... | User Label                      | ...                                 |
|----------|---------------|-------------------|-------------------|------------------|----------------|-----------|---------------------------------|-------------------------------------|
| PD8      | USART3_TX     | n/a               | Alternate Func... | No pull-up an... | Very High      | n/a       | INTERNAL_UART3_TX [ISM43362_RX] | <input checked="" type="checkbox"/> |
| PD9      | USART3_RX     | n/a               | Alternate Func... | No pull-up an... | Very High      | n/a       | INTERNAL_UART3_RX [ISM43362_TX] | <input checked="" type="checkbox"/> |

ISM43362 - Wifi Module

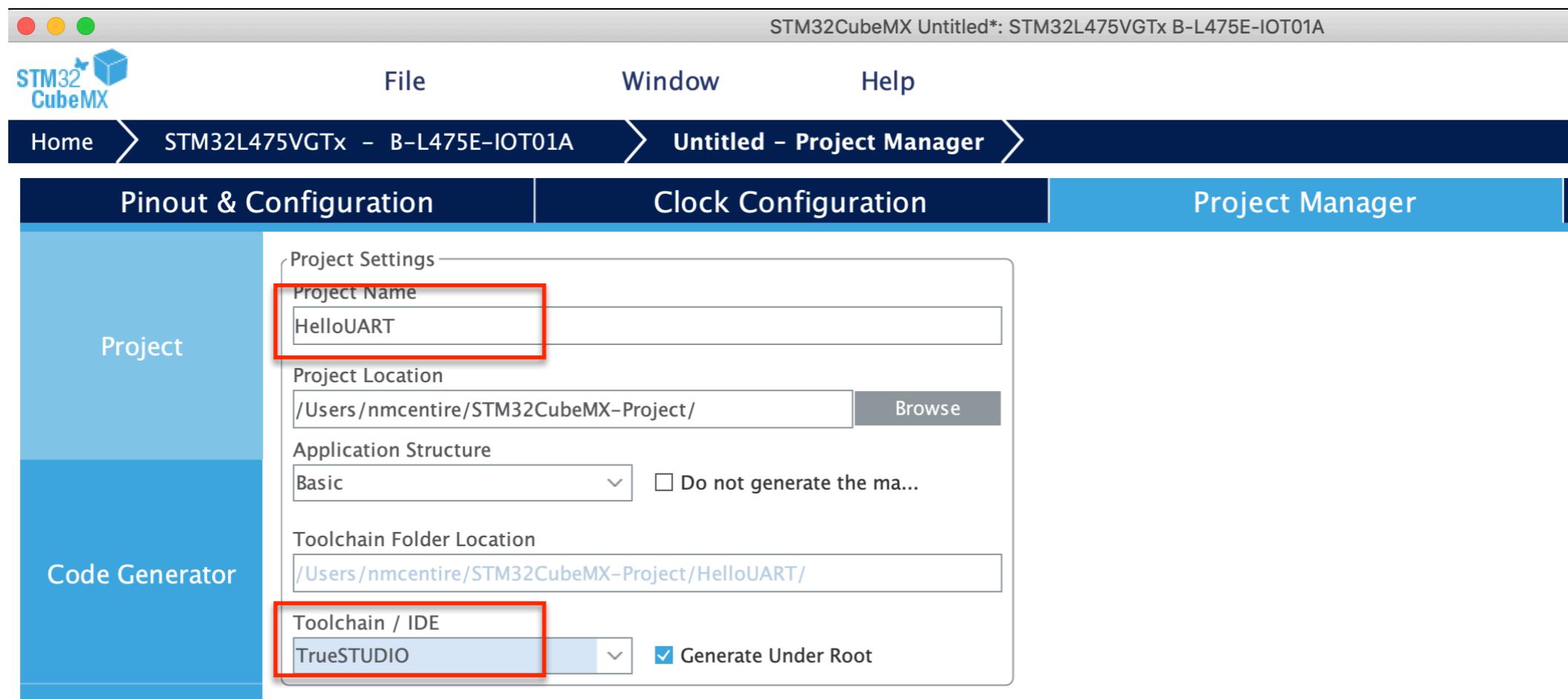
**ISM43362-M3G-L44-E/U Serial-to-WiFi  
Module**



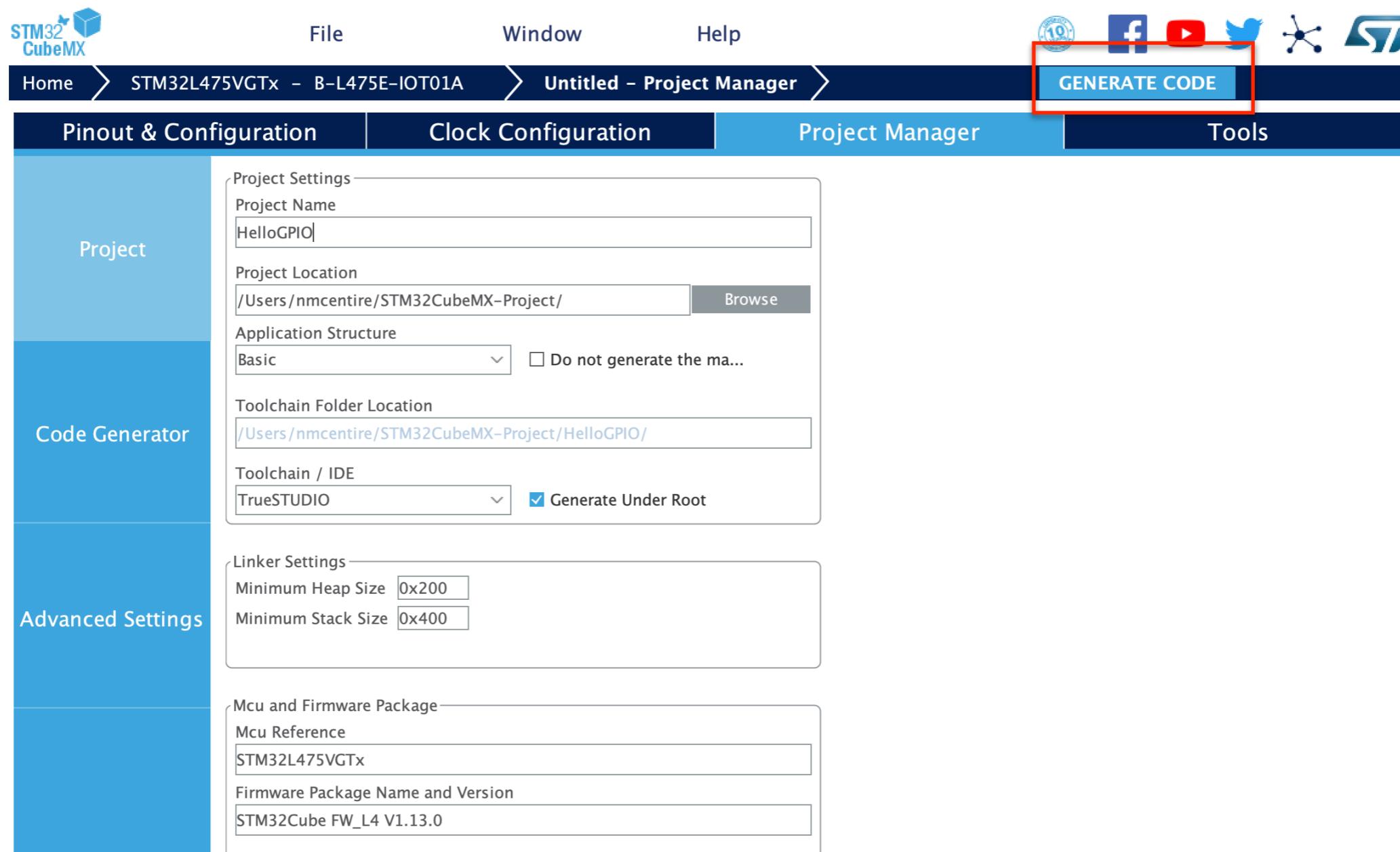
# Step: Observe “Project Manager”



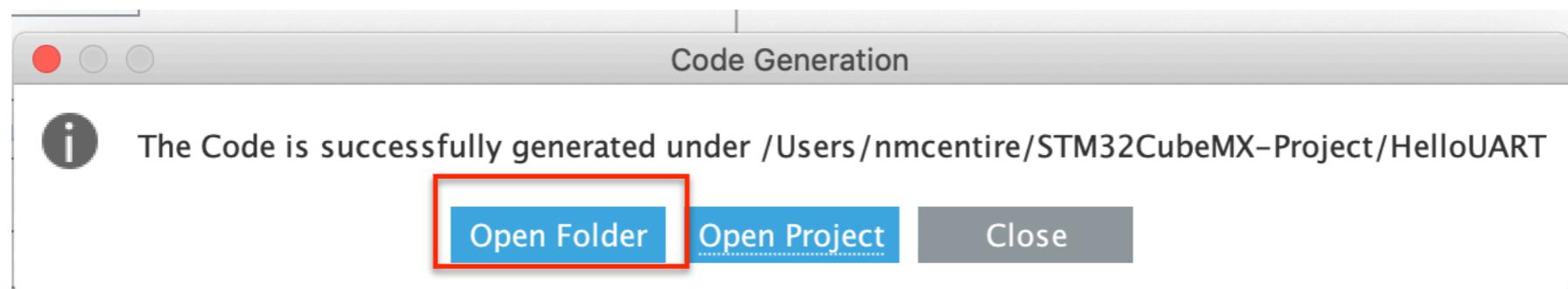
# Step: Enter “HelloUART” for Project Name, TrueStudio Toolchain



# Step: Click on “Generate Code”

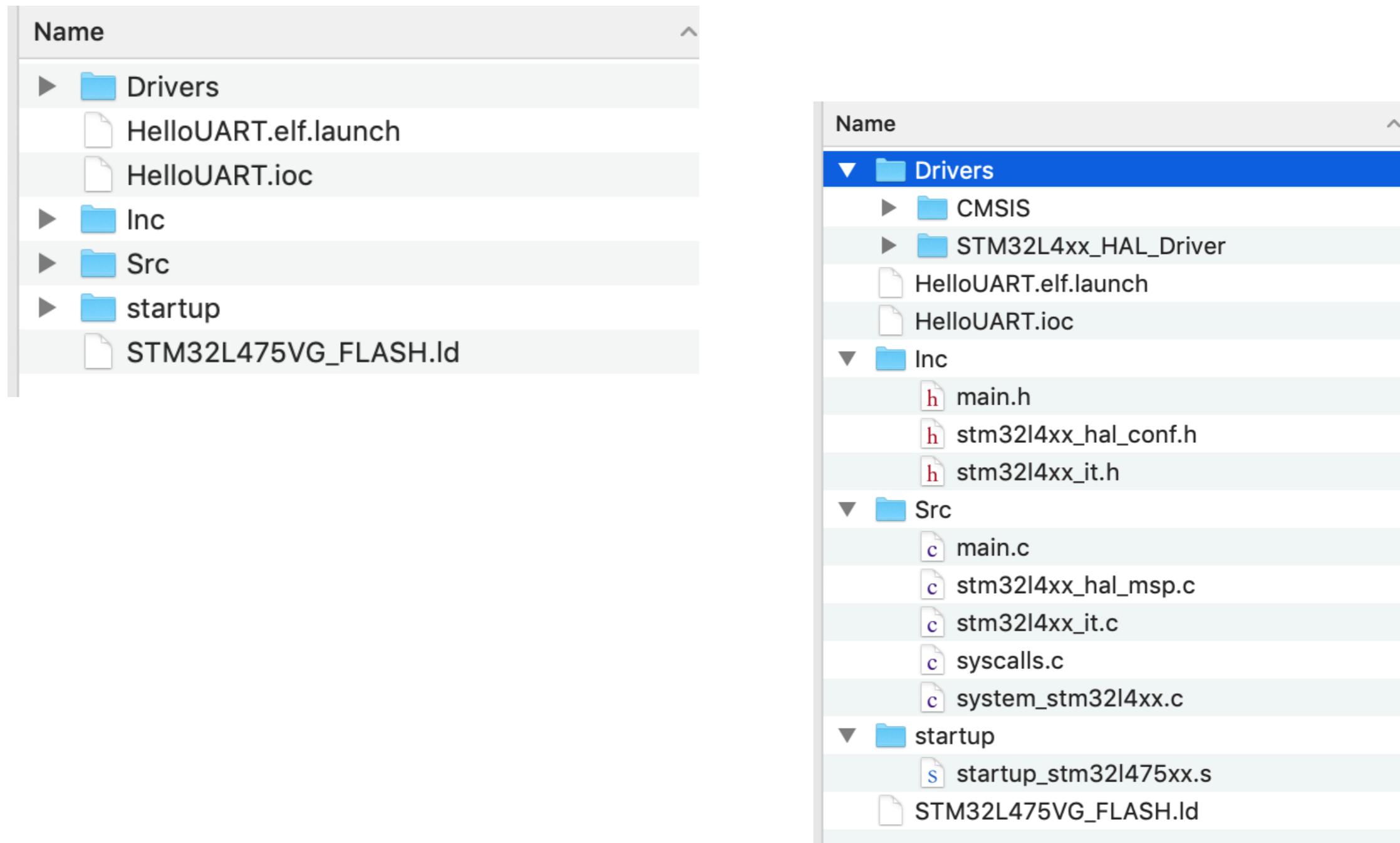


# Step: Select “Open Folder”

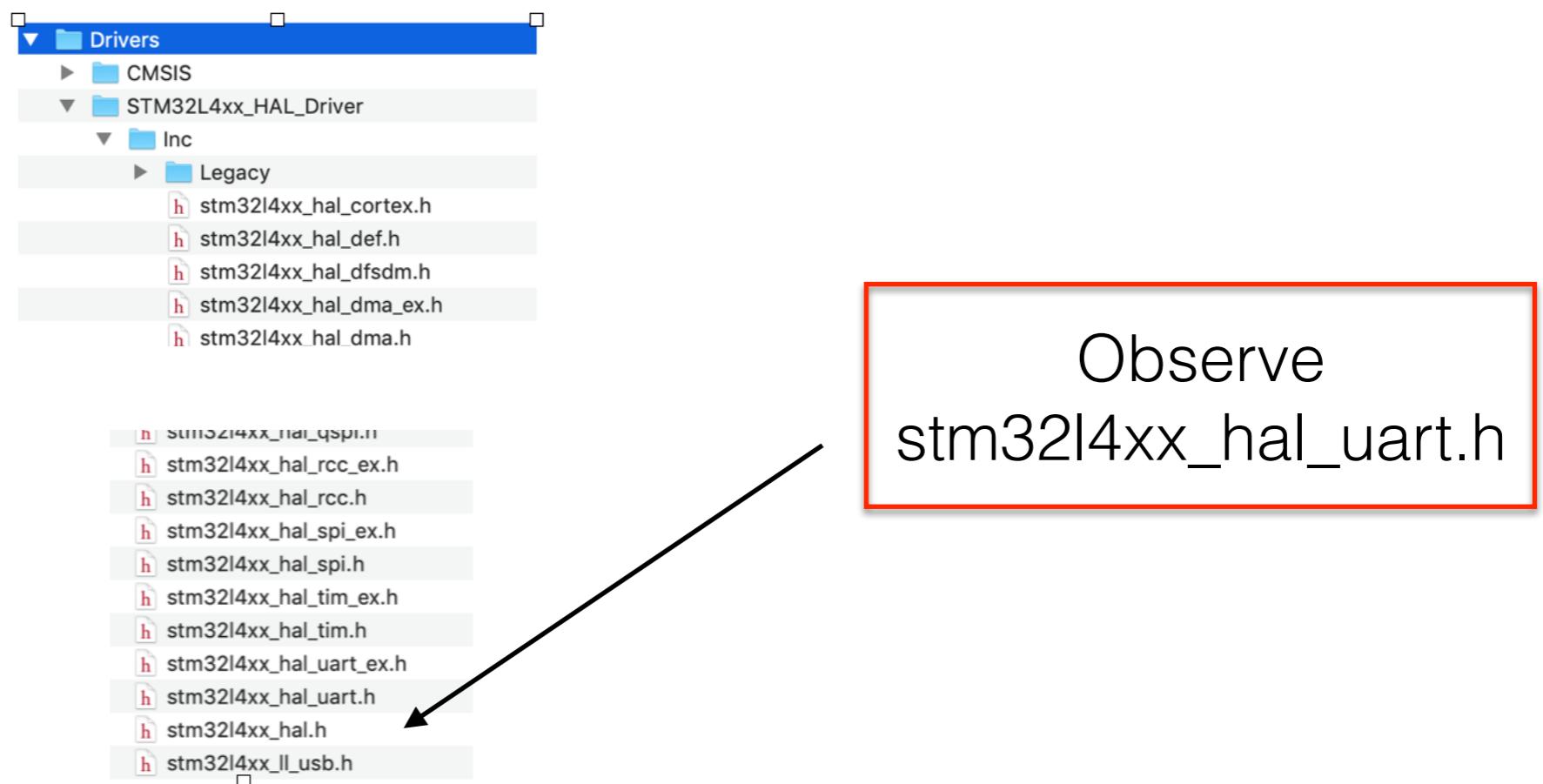


# Tour of Generated Project

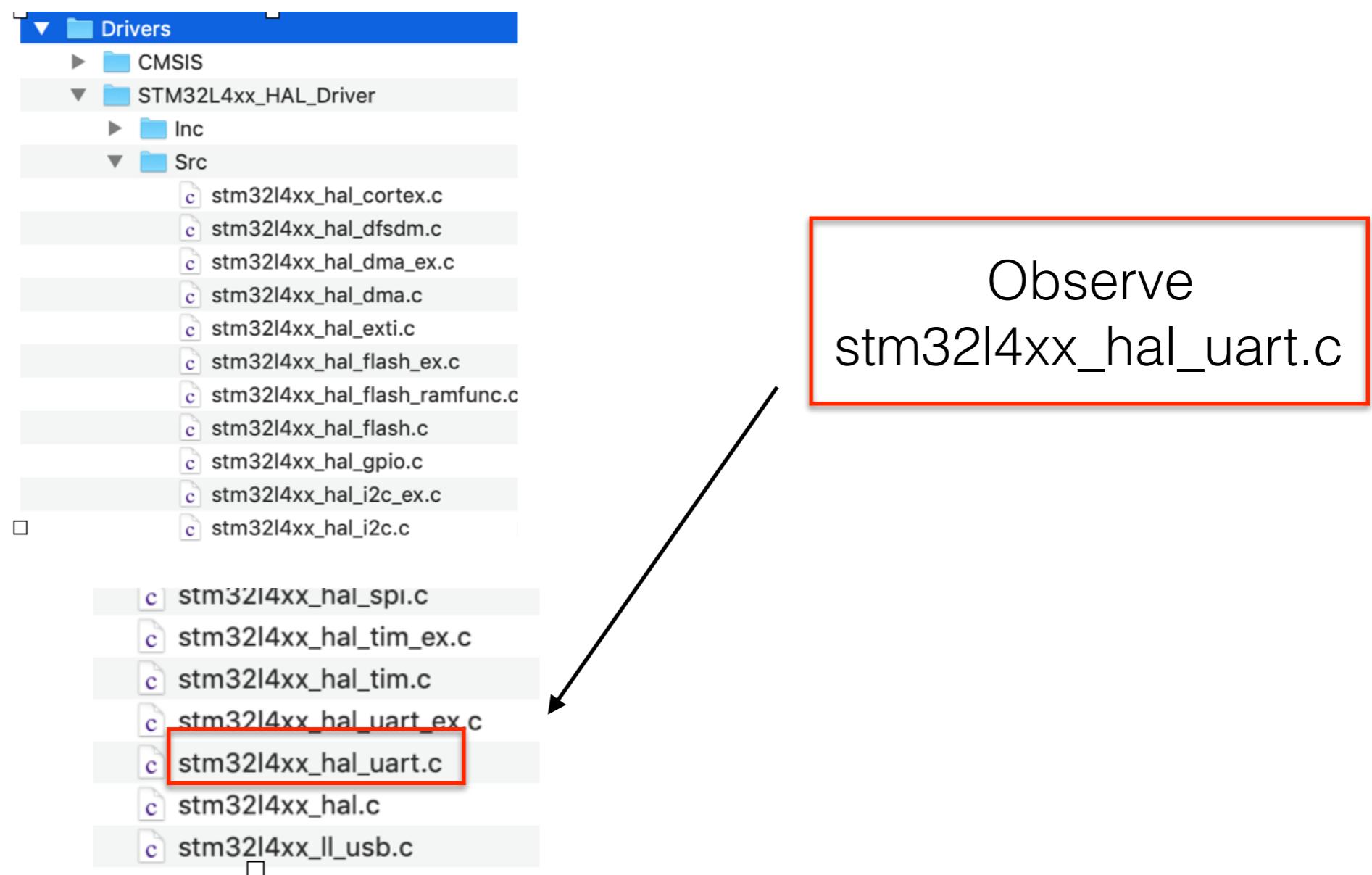
# Step: View Open Folder



# Step: View “Drivers/STM32L4xx\_HAL\_Driver/Inc” directory



# Step: View “Drivers/STM32L4xx\_HAL\_Driver/Src” directory



Observe  
stm32l4xx\_hal\_uart.c

# main.h - Part 1

```
214 #define ST_LINK_UART1_TX_Pin GPIO_PIN_6  
215 #define ST_LINK_UART1_TX_GPIO_Port GPIOB  
216 #define ST_LINK_UART1_RX_Pin GPIO_PIN_7  
217 #define ST_LINK_UART1_RX_GPIO_Port GPIOB
```

```
201 #define PMOD_UART2_RTS_GPIO_Port GPIOD  
202 #define PMOD_UART2_TX_Pin GPIO_PIN_5  
203 #define PMOD_UART2_TX_GPIO_Port GPIOD  
204 #define PMOD_UART2_RX_Pin GPIO_PIN_6  
205 #define PMOD_UART2_RX_GPIO_Port GPIOD
```

```
140 #define INTERNAL_UART3_TX_Pin GPIO_PIN_8  
141 #define INTERNAL_UART3_TX_GPIO_Port GPIOD  
142 #define INTERNAL_UART3_RX_Pin GPIO_PIN_9  
143 #define INTERNAL_UART3_RX_GPIO_Port GPIOD
```

# main.c - Part 1

```
81  /*
85  int main(void)
86  {
87      /* USER CODE BEGIN 1 */
88
89      /* USER CODE END 1 */
90
91      /* MCU Configuration-----*/
92
93      /* Reset of all peripherals,
94      HAL_Init();
95
```

```
113      MX_USART1_UART_Init();
114      MX_USART3_UART_Init();
```

# main.c - Part 2

```
368 static void MX_USART1_UART_Init(void)
369 {
370
371     /* USER CODE BEGIN USART1_Init_0 */
372
373     /* USER CODE END USART1_Init_0 */
374
375     /* USER CODE BEGIN USART1_Init_1 */
376
377     /* USER CODE END USART1_Init_1 */
378     huart1.Instance = USART1;
379     huart1.Init.BaudRate = 115200;
380     huart1.Init.WordLength = UART_WORDLENGTH_8B;
381     huart1.Init.StopBits = UART_STOPBITS_1;
382     huart1.Init.Parity = UART_PARITY_NONE;
383     huart1.Init.Mode = UART_MODE_TX_RX;
384     huart1.Init.HwFlowCtl = UART_HWCONTROL_NONE;
385     huart1.Init.OverSampling = UART_OVERSAMPLING_16;
386     huart1.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
387     huart1.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
388     if (HAL_UART_Init(&huart1) != HAL_OK)
389     {
390         Error_Handler();
391     }
```

# main.c - Part 3

```
402 |     */
403 static void MX_USART3_UART_Init(void)
404 {
405
406     /* USER CODE BEGIN USART3_Init_0 */
407
408     /* USER CODE END USART3_Init_0 */
409
410     /* USER CODE BEGIN USART3_Init_1 */
411
412     huart3.Instance = USART3;
413     huart3.Init.BaudRate = 115200;
414     huart3.Init.WordLength = UART_WORDLENGTH_8B;
415     huart3.Init.StopBits = UART_STOPBITS_1;
416     huart3.Init.Parity = UART_PARITY_NONE;
417     huart3.Init.Mode = UART_MODE_TX_RX;
418     huart3.Init.HwFlowCtl = UART_HWCONTROL_NONE;
419     huart3.Init.OverSampling = UART_OVERSAMPLING_16;
420     huart3.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
421     huart3.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
422
423     if (HAL_UART_Init(&huart3) != HAL_OK)
424     {
425         Error_Handler();
426     }
}
```

# main.c - Part 4

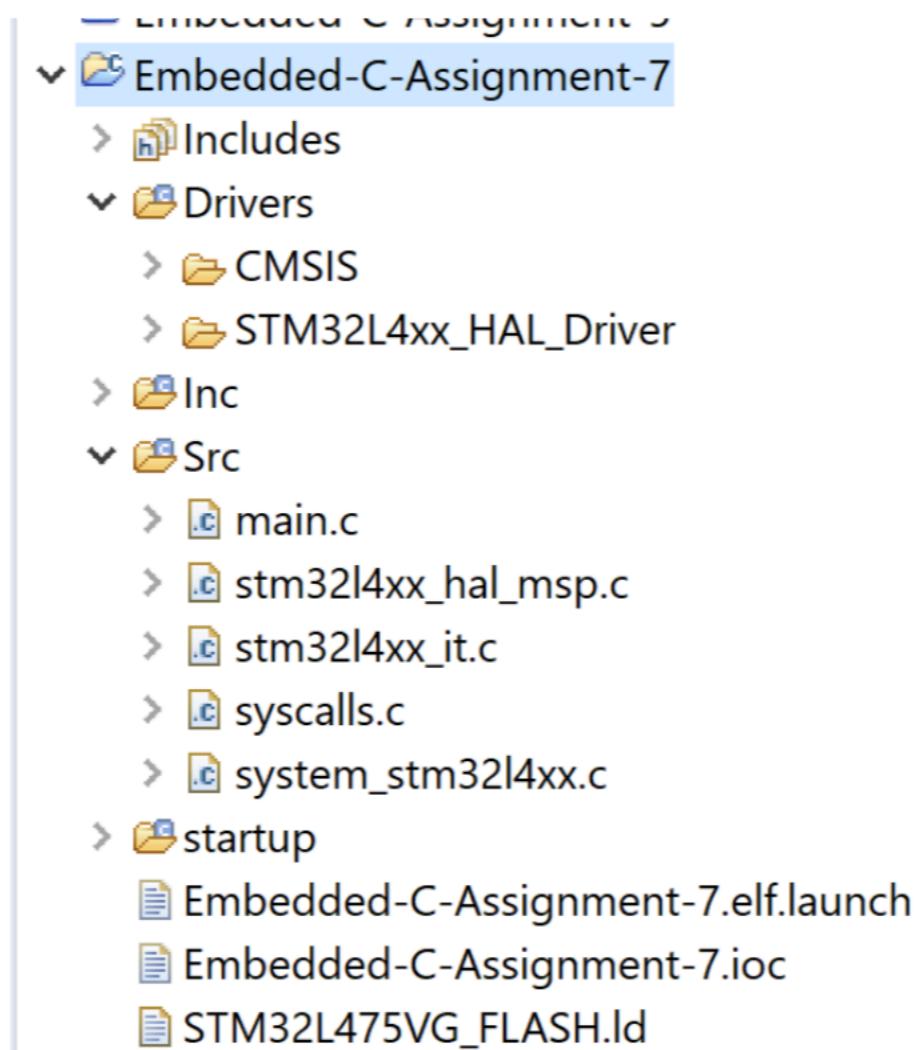
```
473 static void MX_GPIO_Init(void)
474 {
475     GPIO_InitTypeDef GPIO_InitStruct = {0};
476
477     /**
478      * Configure GPIO pins : ARD_D1_Pin ARD_D0_Pin */
479     GPIO_InitStruct.Pin = ARD_D1_Pin|ARD_D0_Pin;
480     GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
481     GPIO_InitStruct.Pull = GPIO_NOPULL;
482     GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
483     GPIO_InitStruct.Alternate = GPIO_AF8_UART4;
484     HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
485 }
```

# Summary so far...

- STM32CubeMX generates UART Code
- HAL Driver Code
  - Drivers/STM32L4xx\_HAL\_Driver/Inc
    - stm32l4xx\_hal\_uart.h
  - Drivers/STM32L4xx\_HAL\_Driver/Src
    - stm32l4xx\_hal\_uart.c

Use TrueStudio  
To send the pattern “0123456789”,  
one character at a time, every  
second, to UART1

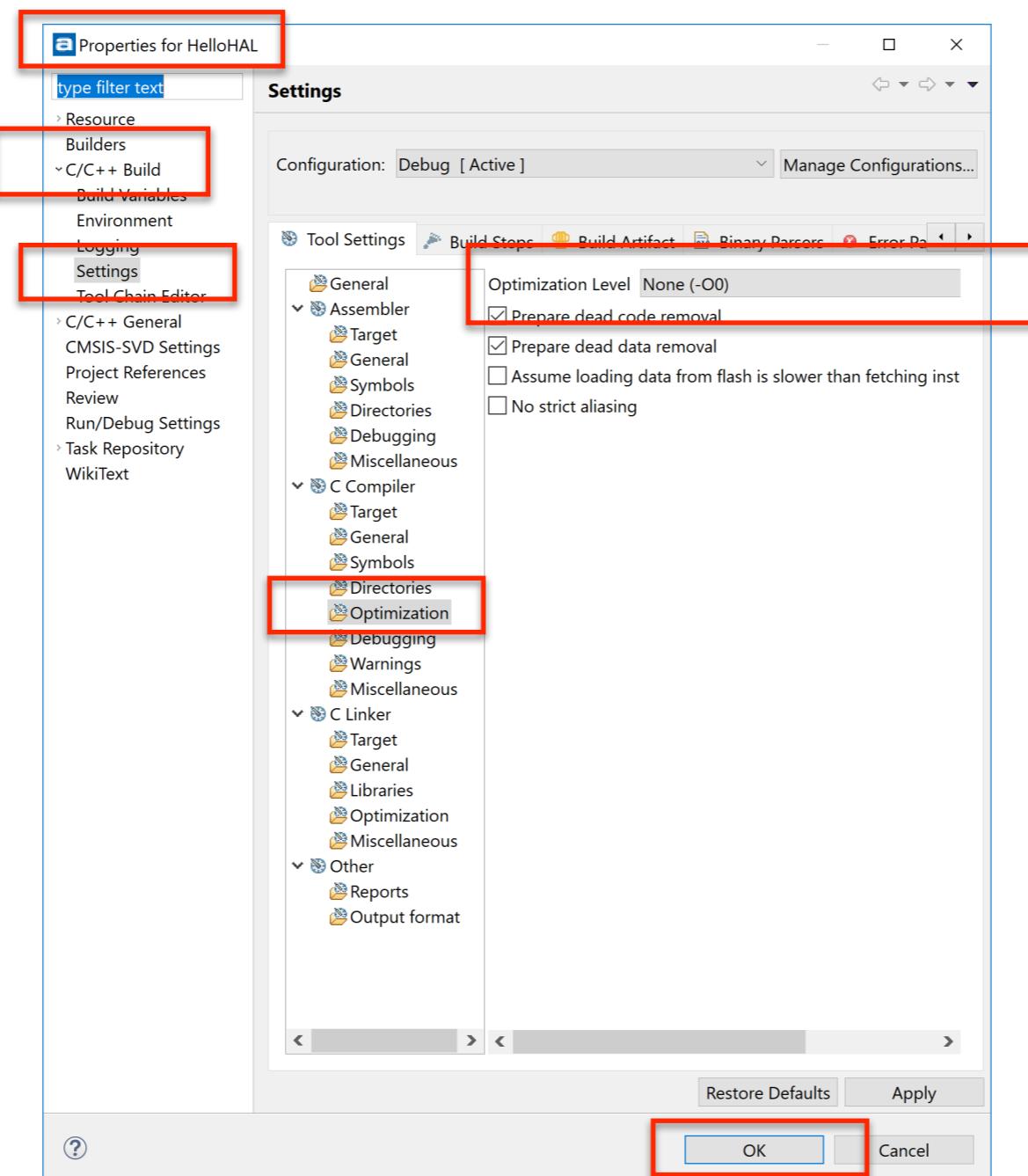
# Step: Project



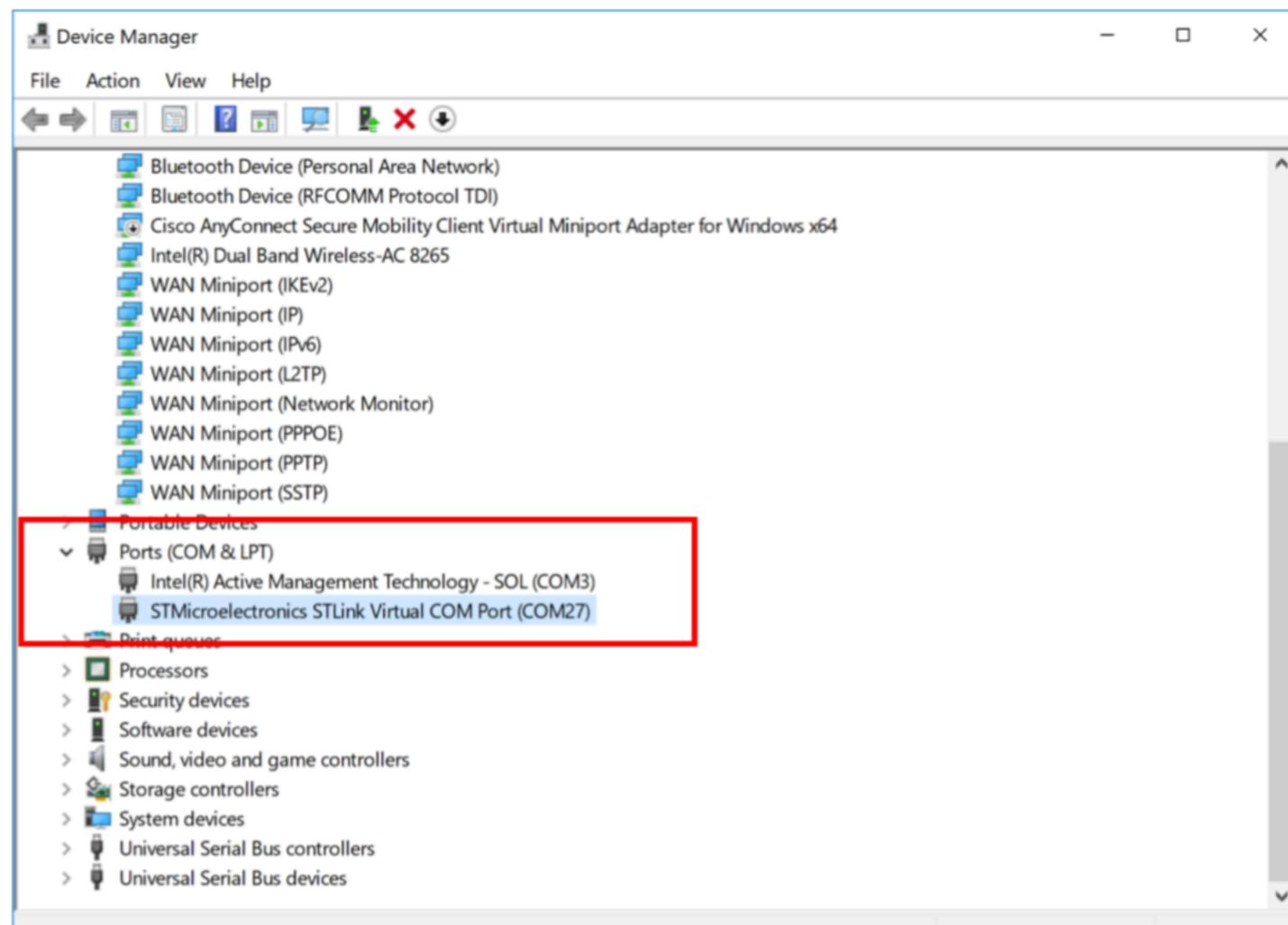
# Step: Edit main.c

```
119  /* USER CODE END 2 */  
120  
121  /* Infinite loop */  
122  /* USER CODE BEGIN WHILE */  
123  char *msg = "0123456789\n";  
124  int index = 0;  
125  int len = strlen(msg);  
126  
127  while (1)  
128  {  
129      HAL_UART_Transmit(&huart1, (uint8_t *)&msg[index], 1, 0);  
130  
131      index++;  
132      if (index == len) {  
133          index = 0;  
134      }  
135  
136      HAL_Delay(1000);  
137  
138  /* USER CODE END WHILE */  
139  
140  /* USER CODE BEGIN 3 */  
141 }
```

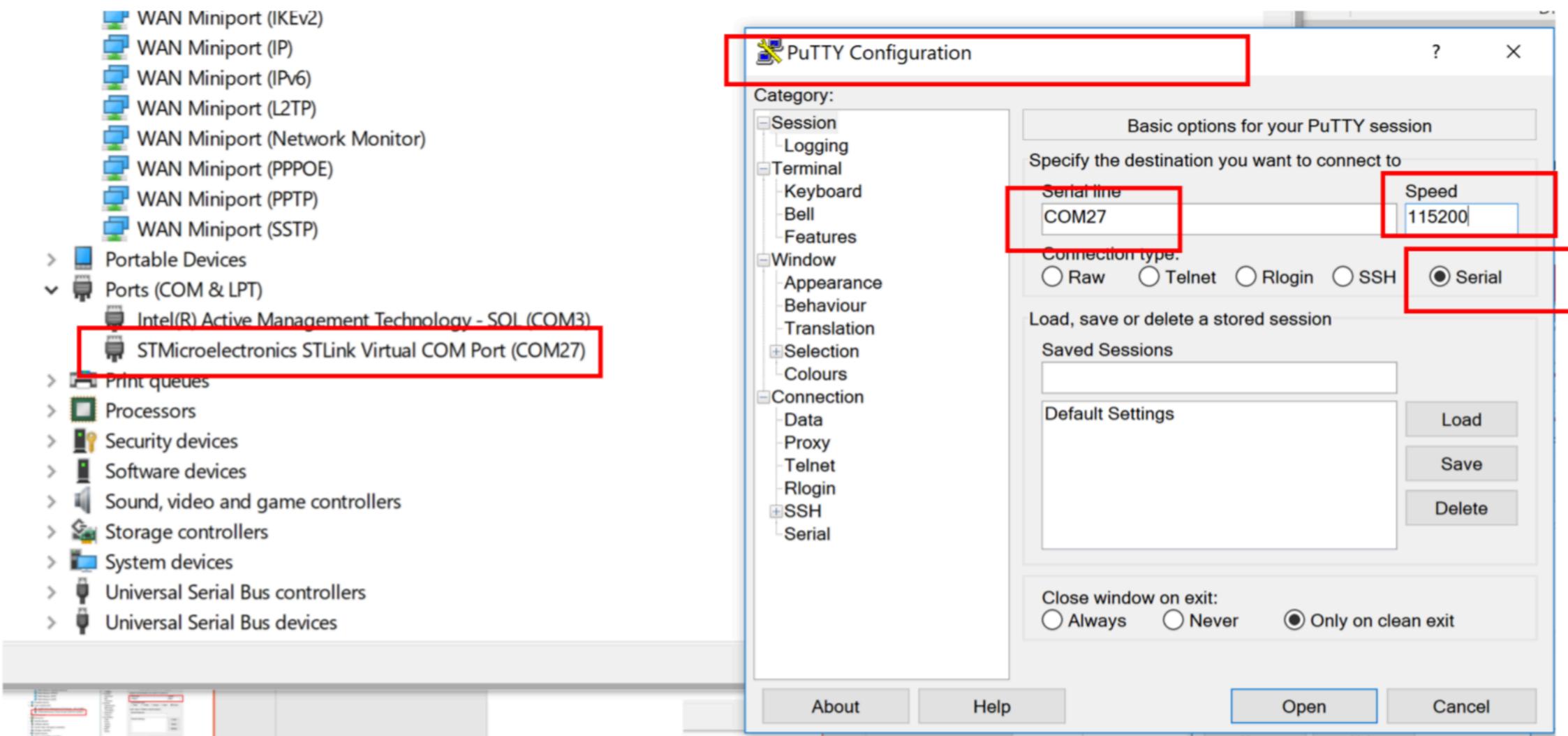
# Step: Project, Properties, C/C++ Build, Settings, C Compiler, Optimization, None



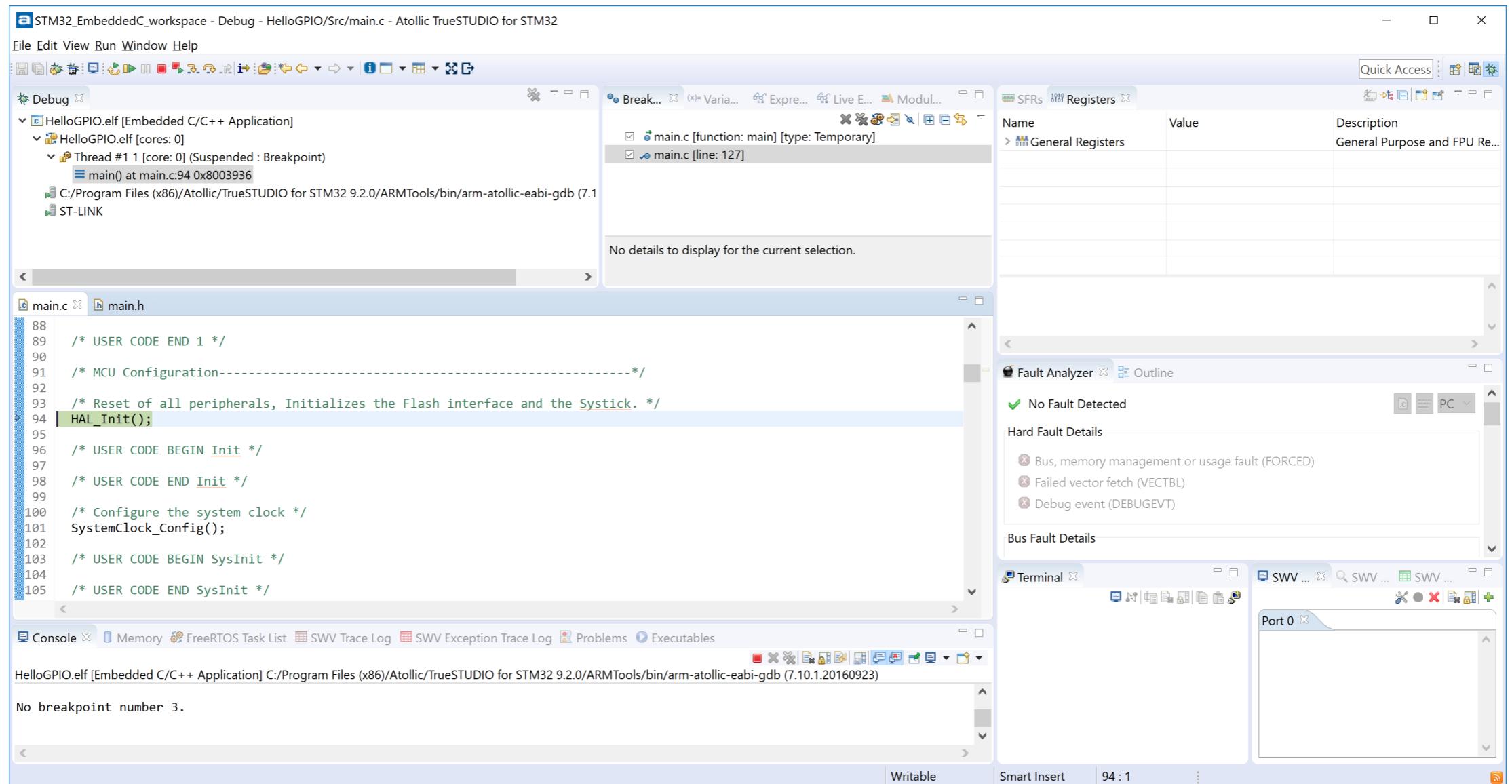
# Step: Plug in STM Board, and observe Device Manager



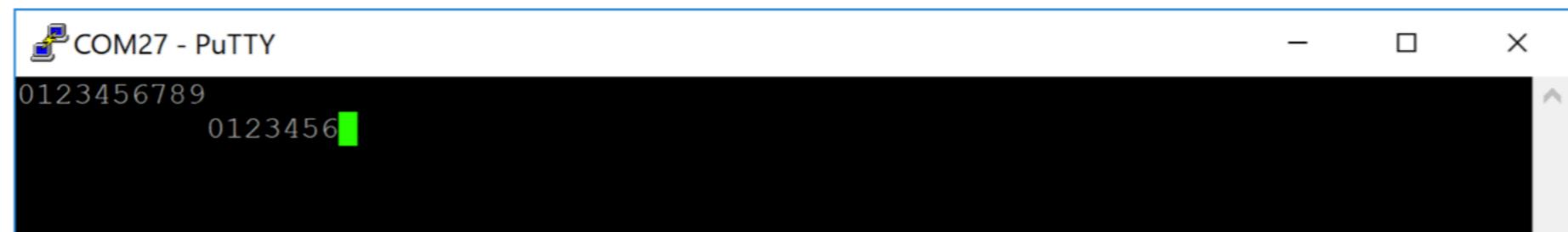
# Step: Startup PuTTY, and configure for STM COM Port



# Step: Build and Run Project in Debug Mode



# Output on PuTTY Serial Port



# Summary

- Introduction to UART and Embedded C
- STM32CubeMX and UART Code Generation
- Tour of UART
- TrueStudio and UART
  - UART Transmit and Receive