Lesson 7 - UART and Embedded C

Norman McEntire
norman.mcentire@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

life.augmented

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

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_PARTIY_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 DeInit

- HAL_UART_Init()
- HAL_UART_DeInit()
- HAL_UART_MspInit()
- HAL_UART_MspDeInit()

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

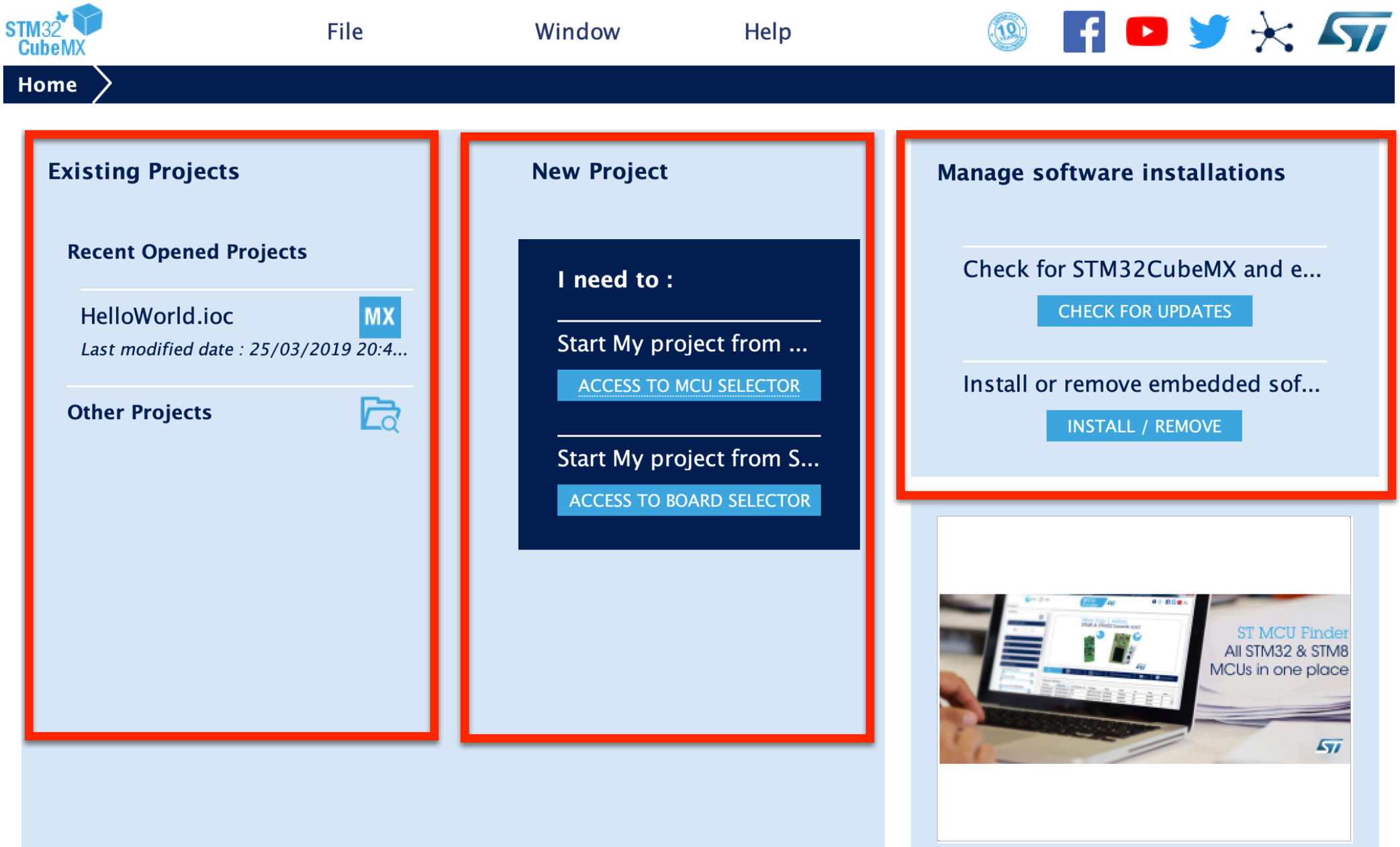
- UART_SetConfig()
- UART_AdvFeatureConfig()

HAL UART - Peripheral State and Error Functions

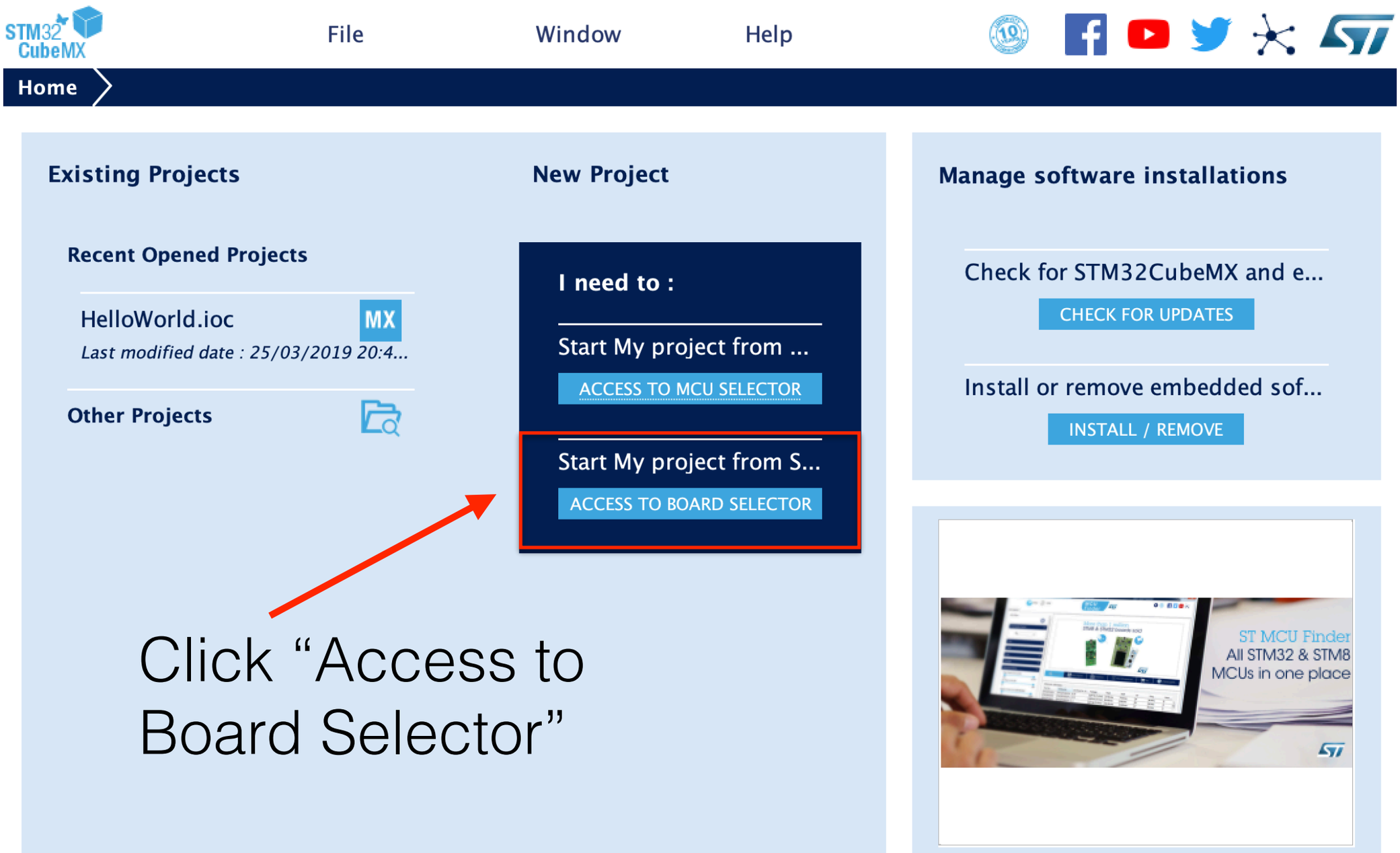
- HAL_UART_GetState()
- HAL_UART_GetError()

STM32CubeMX and Generation of UART Code

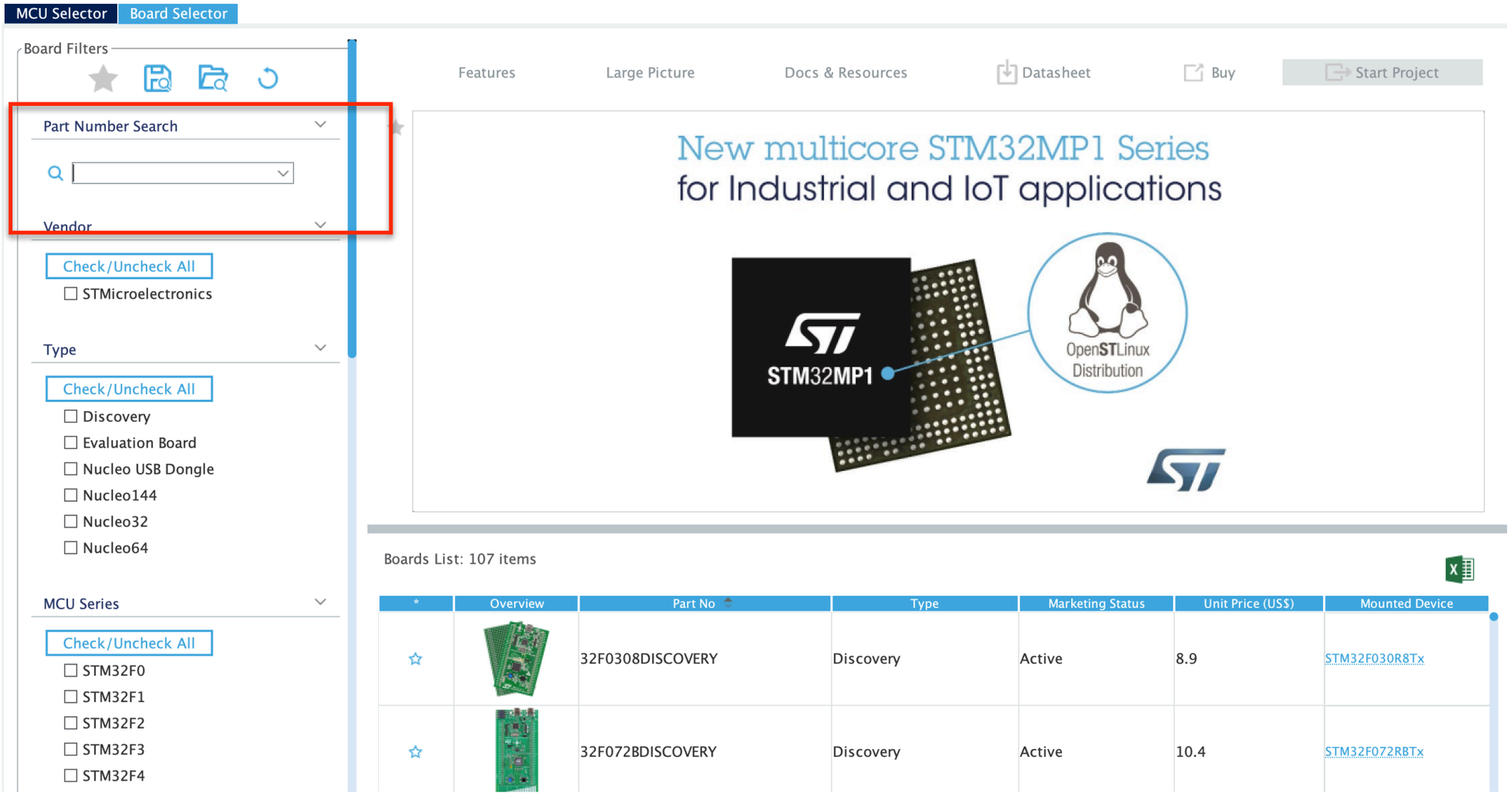
Step: Startup STM32CubeMX



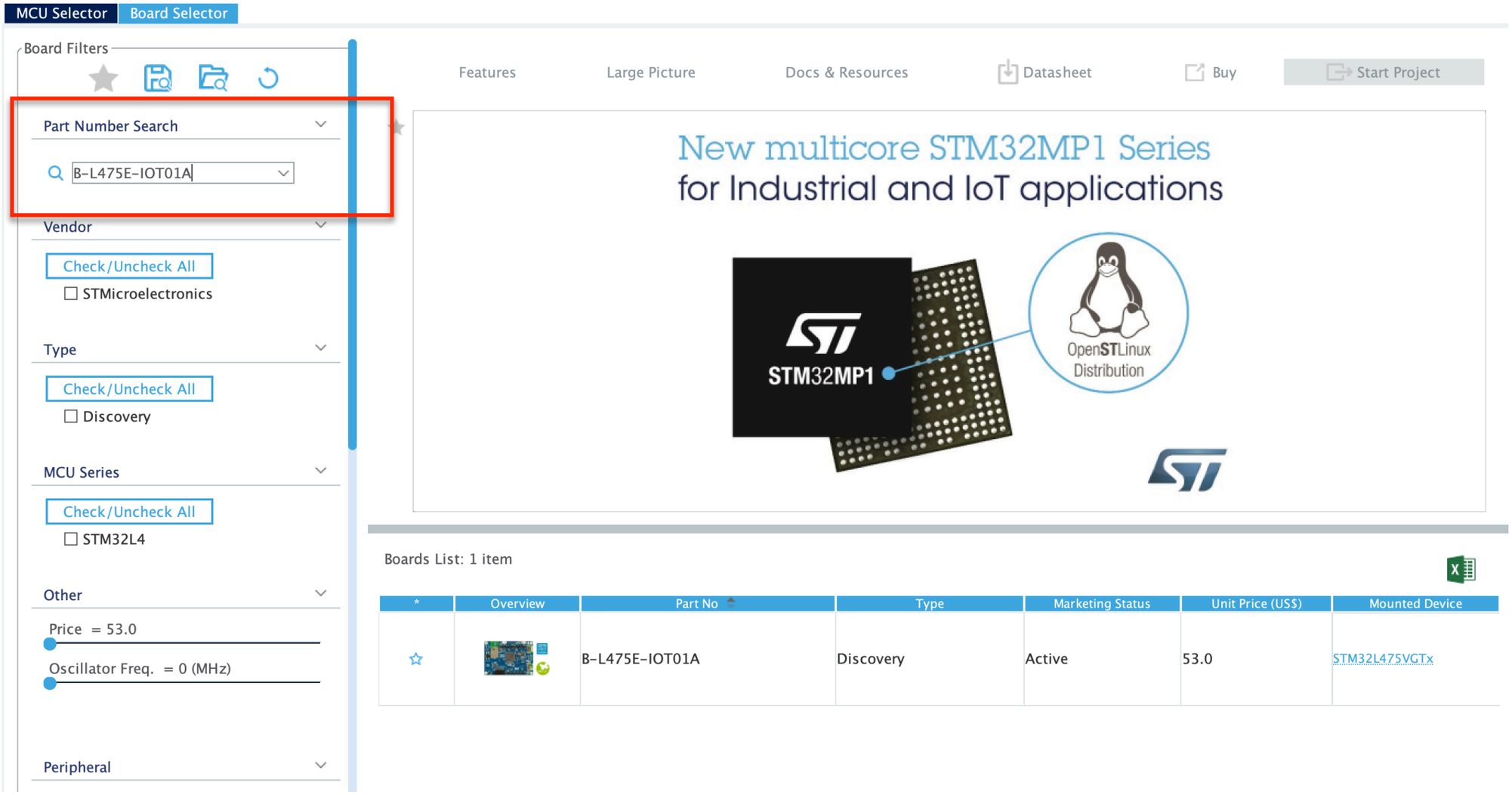
Step: Click on “Access to Board Selector”



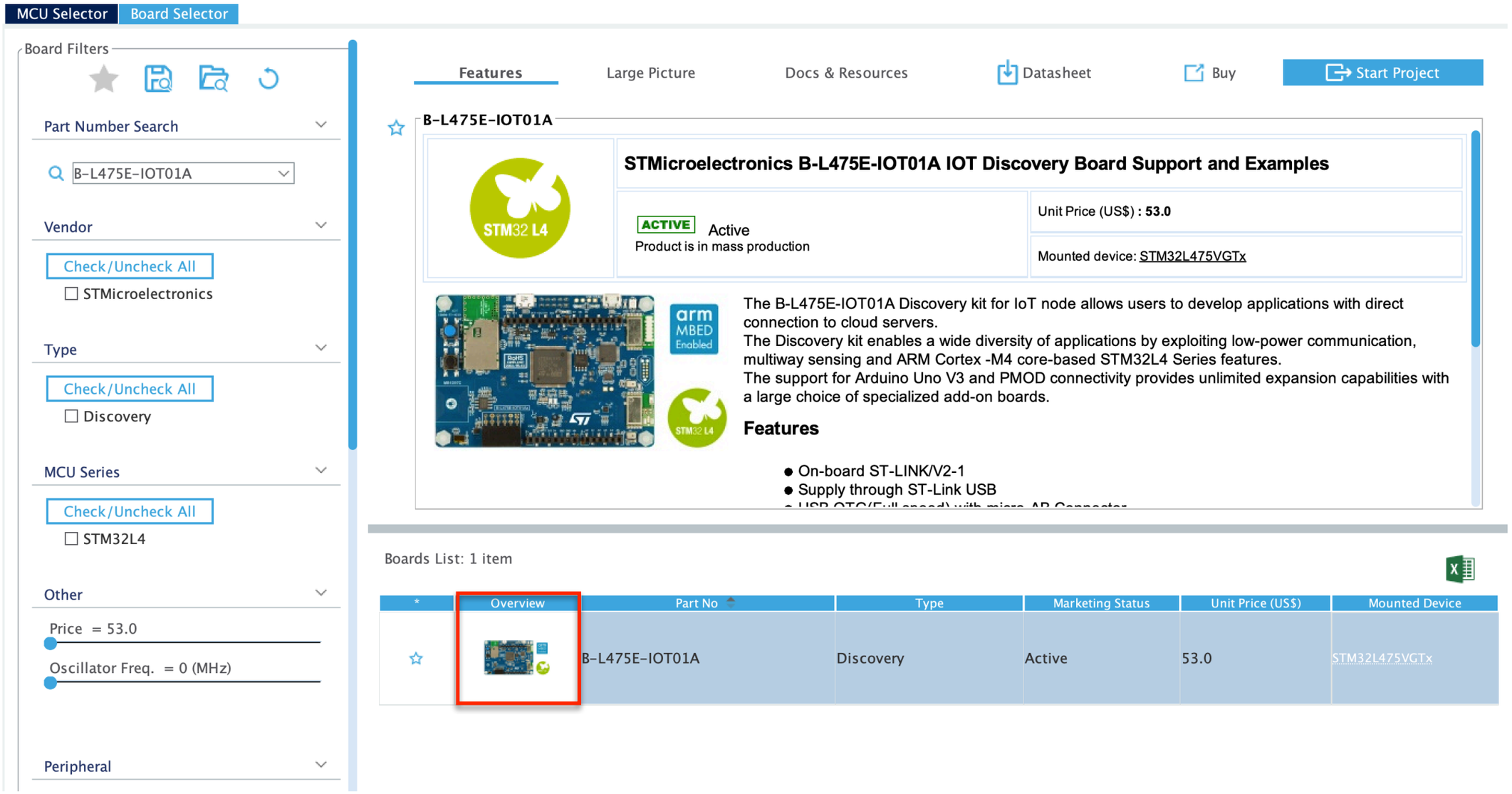
Step: Observe “Part Number Search”



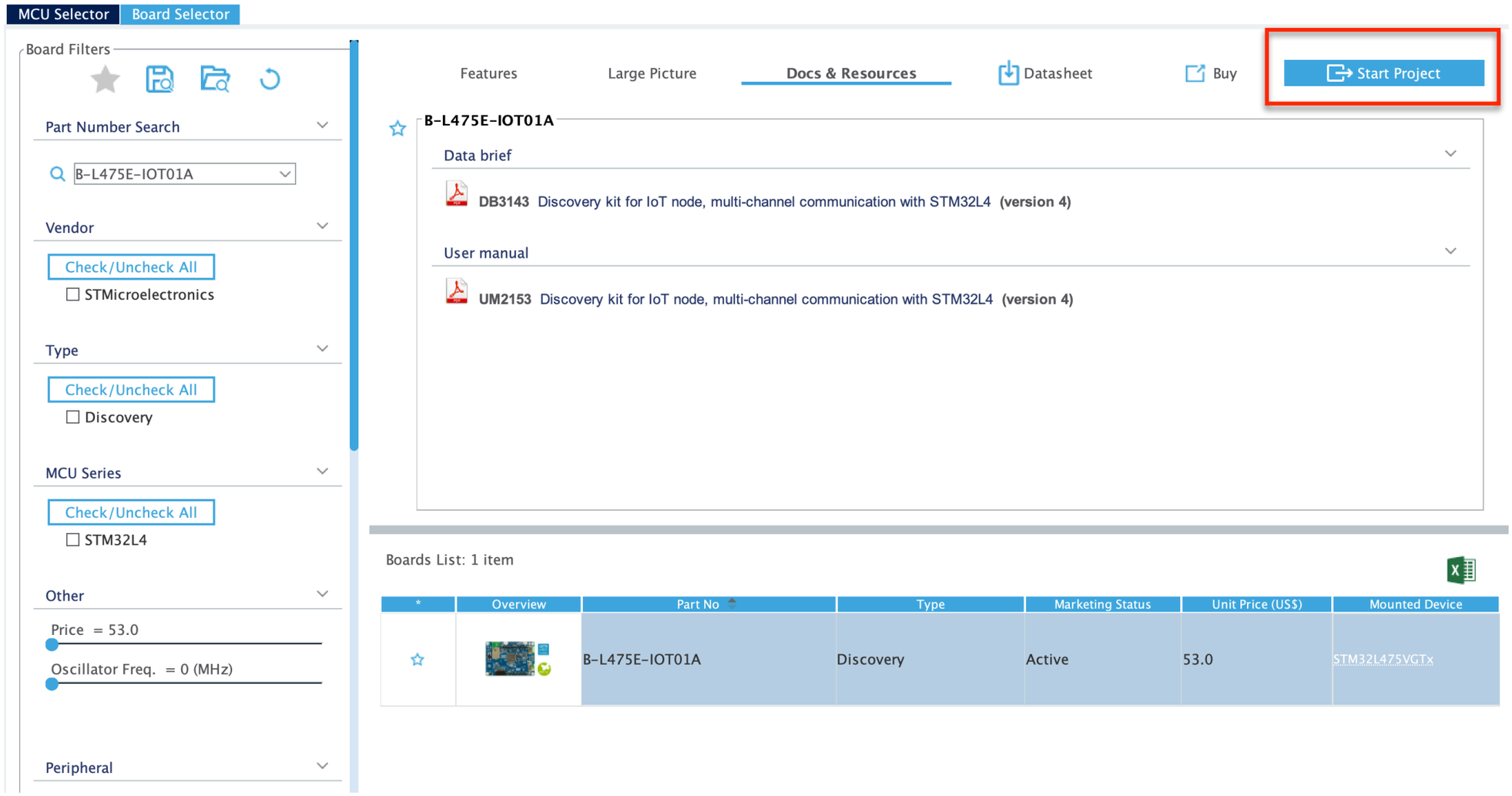
Step: Enter ”B-L475E-IOT01A”



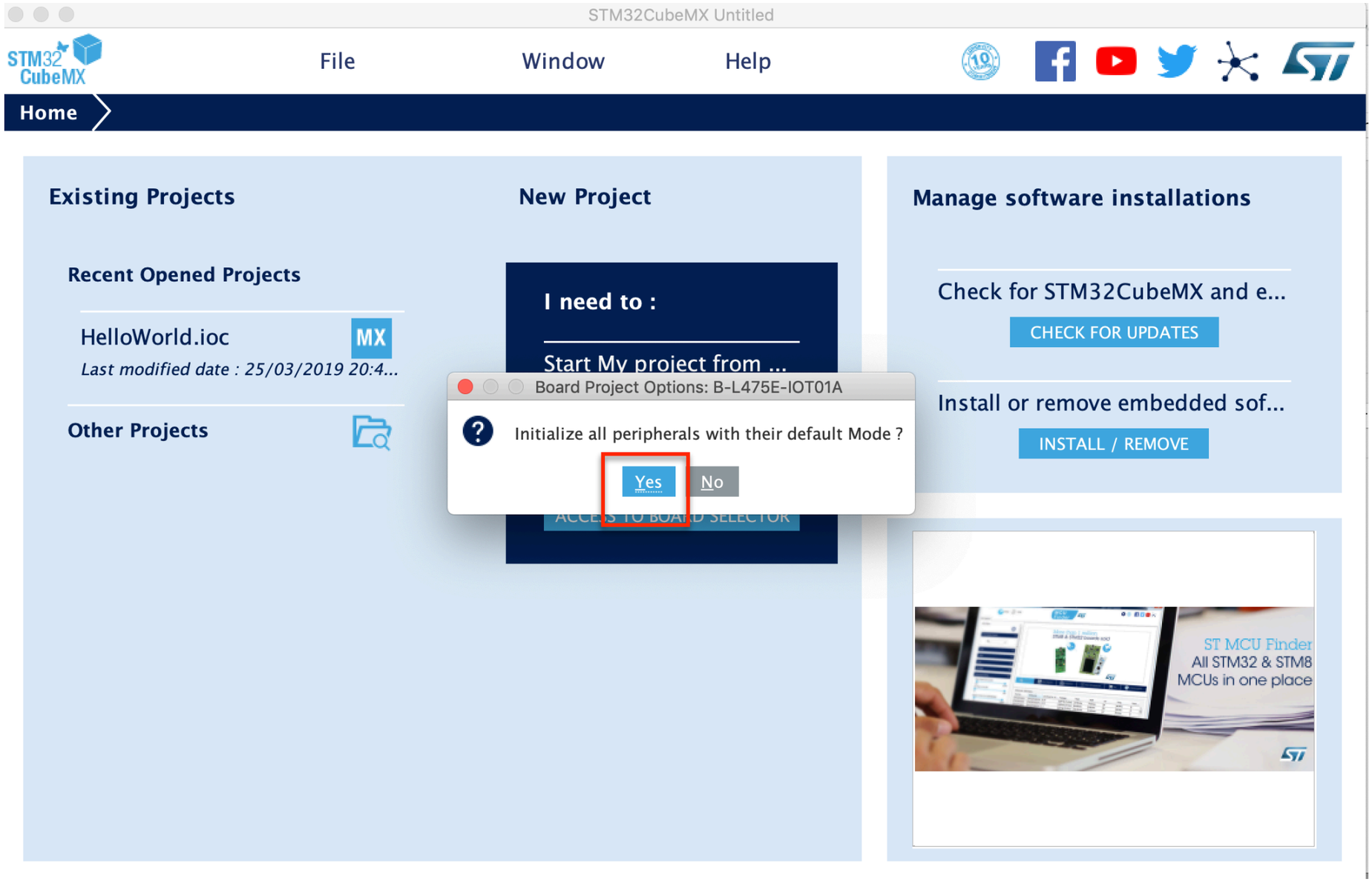
Step: Click on Image, and observe “Features”



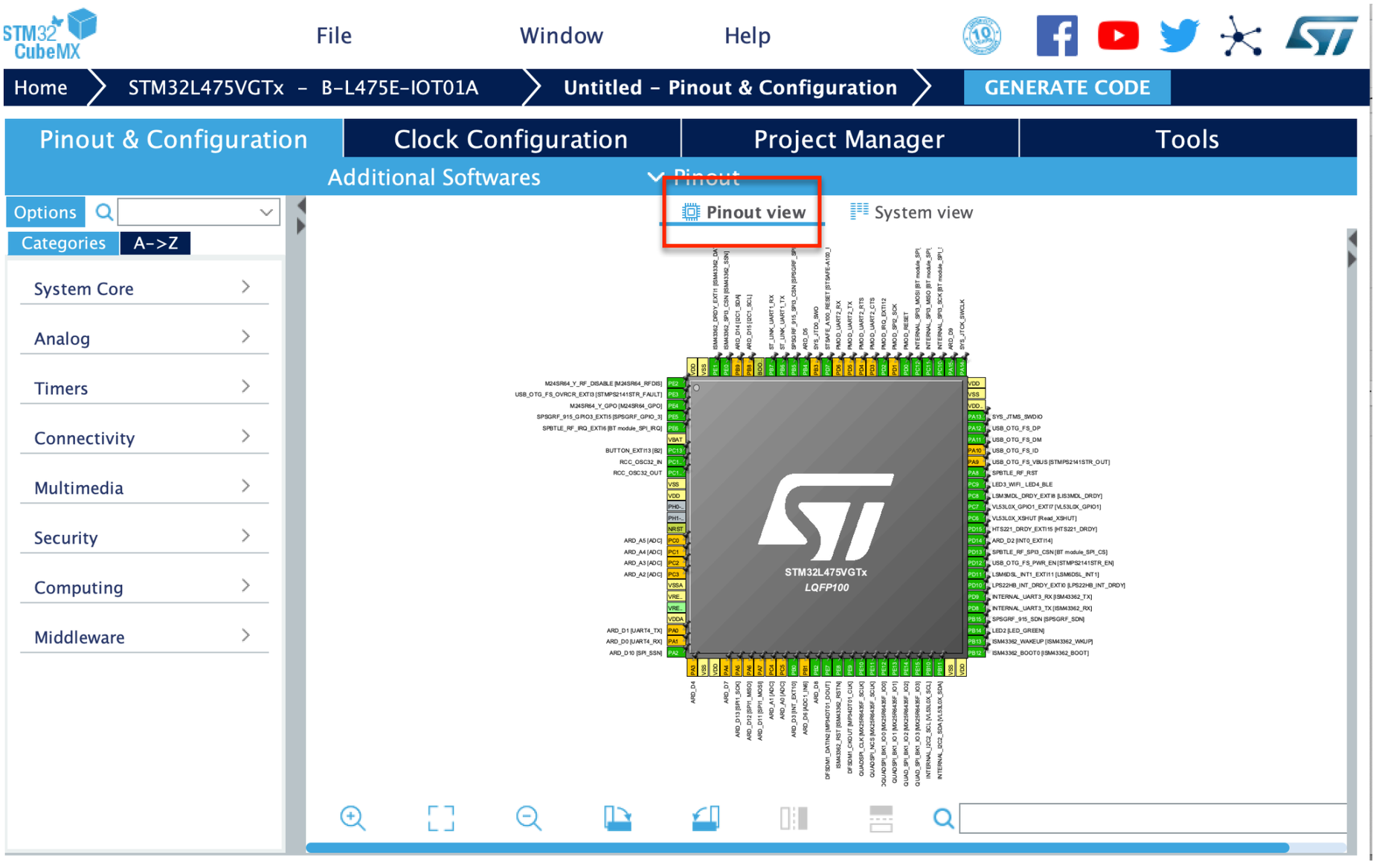
Step: Click on “Start Project”



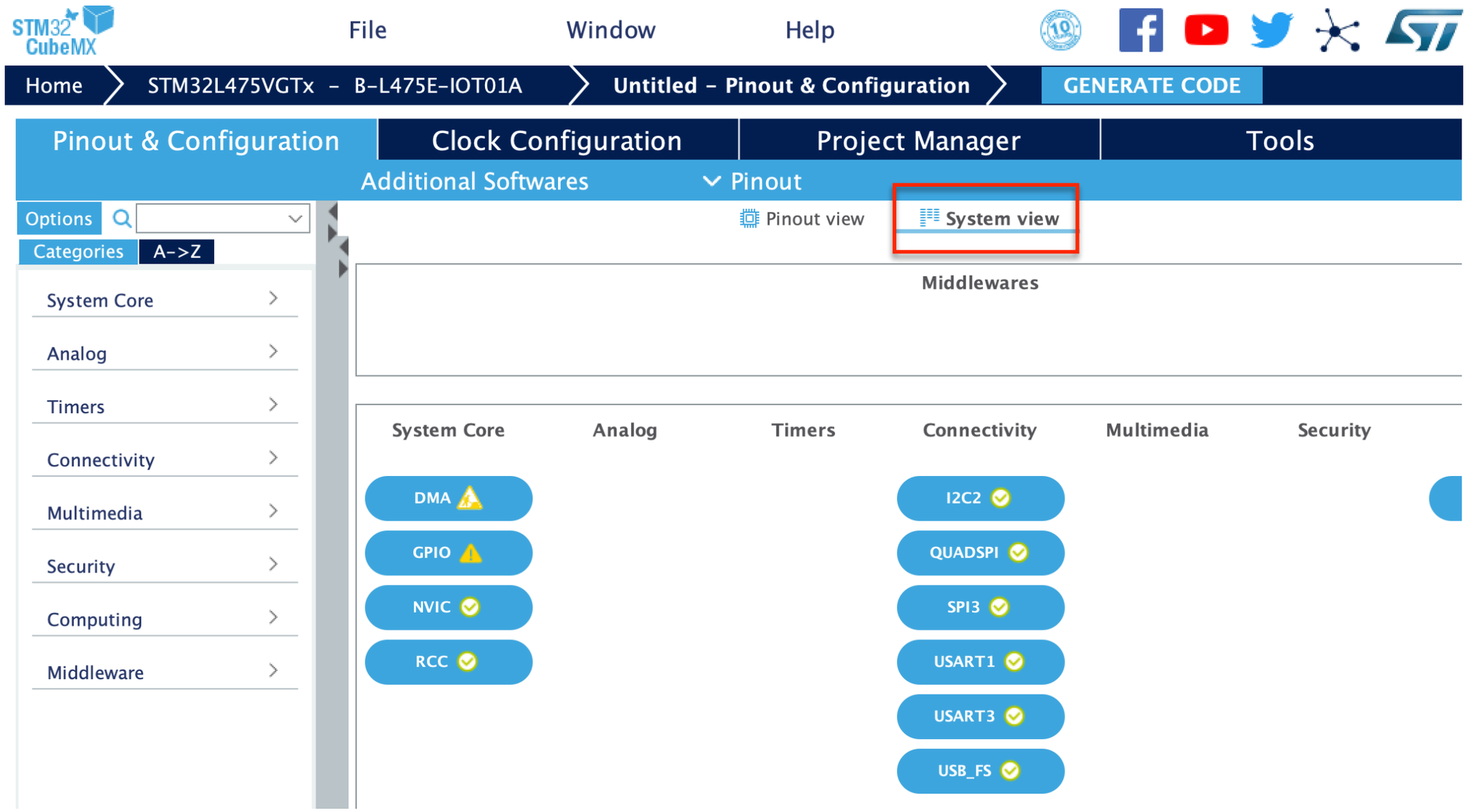
Step: Click on “Yes” (Initialize all …with default mode)



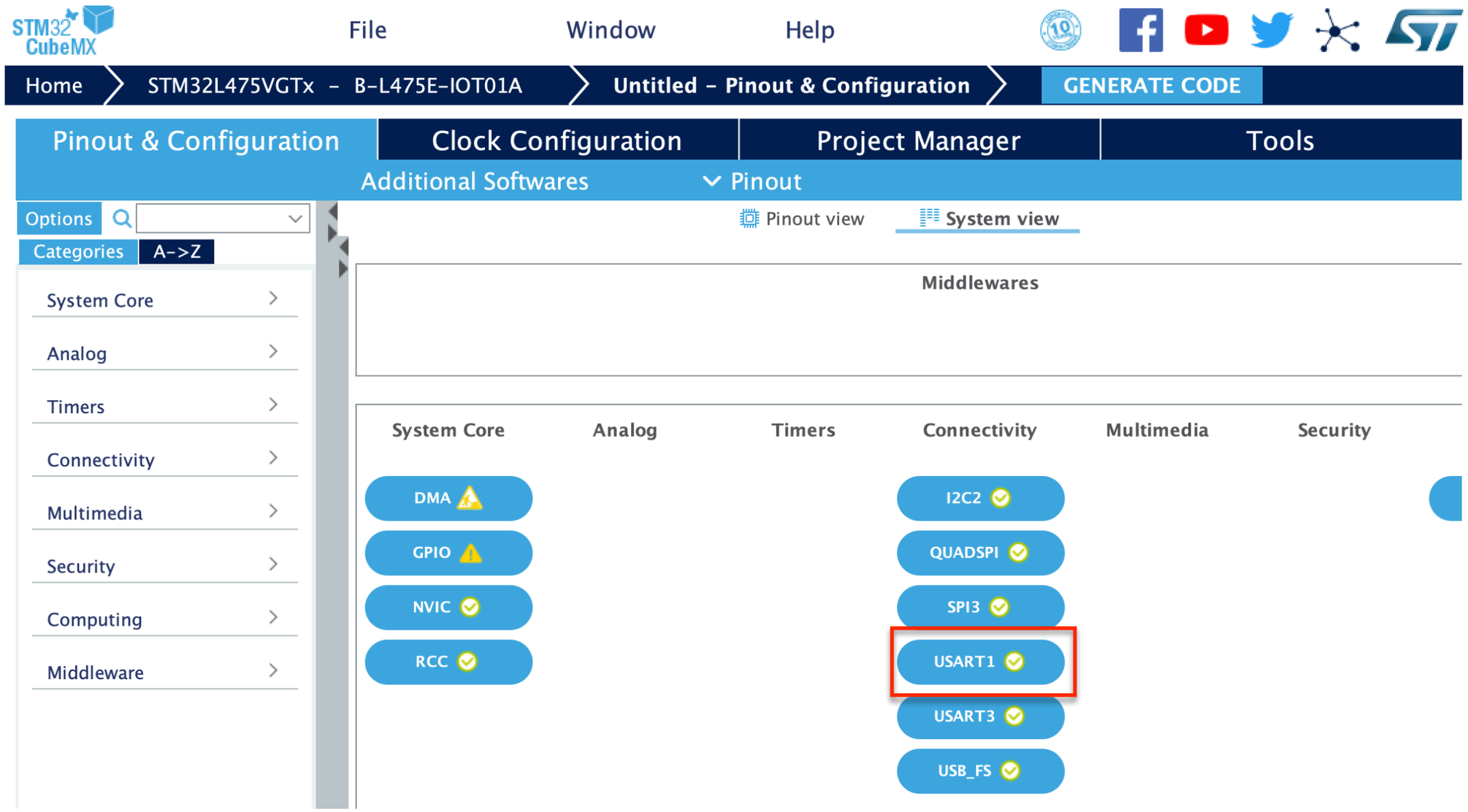
Step: Observe “Pinout View”



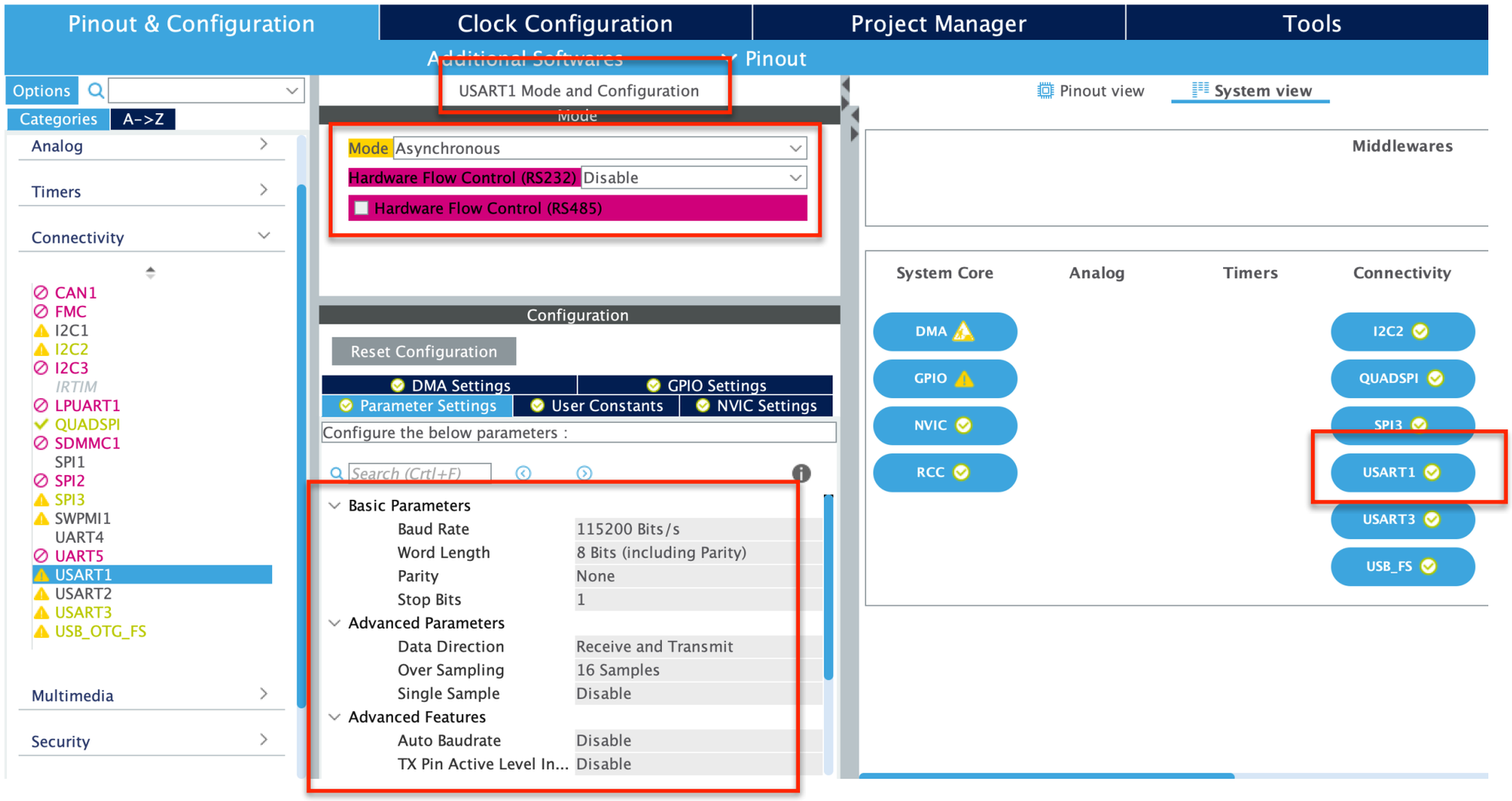
Step: Select “System View”



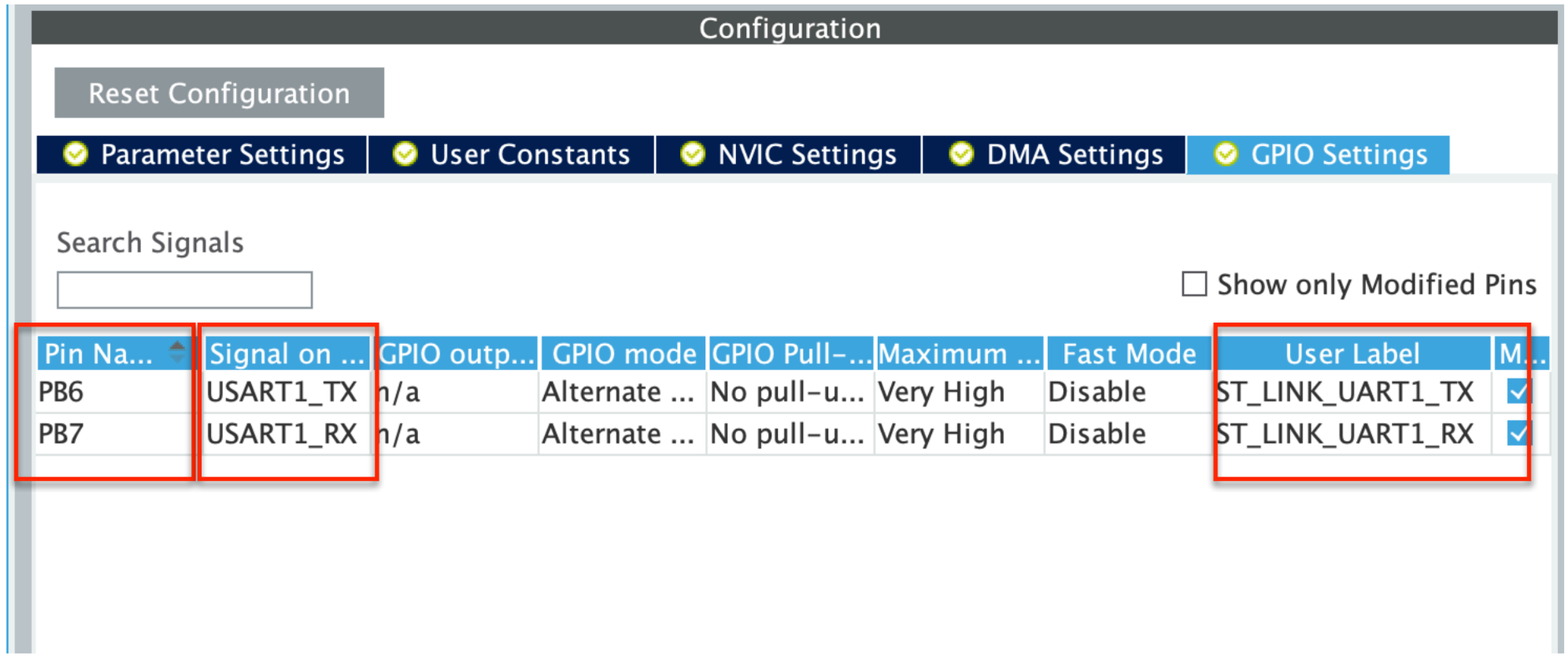
Step: Select “USART1”



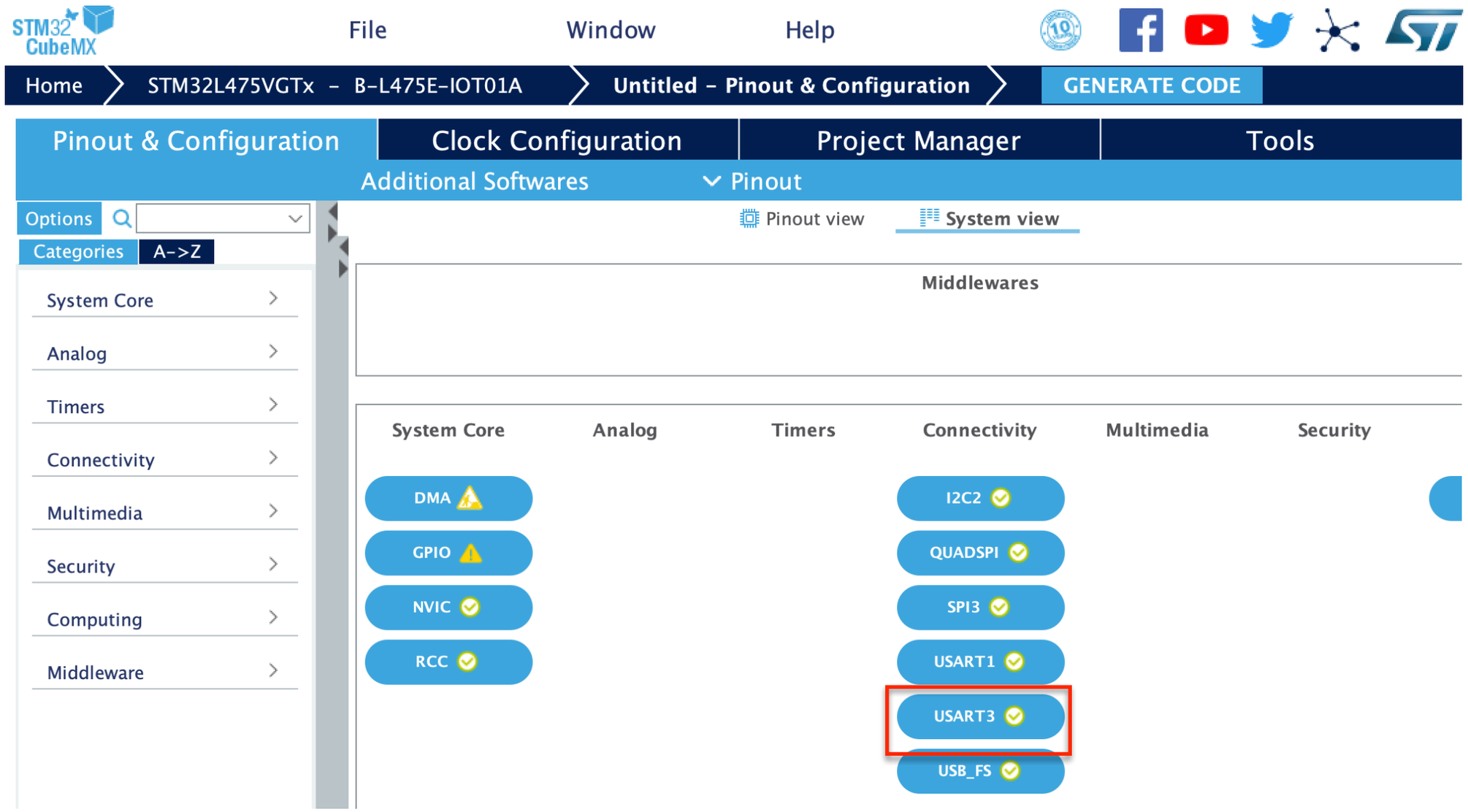
USART1 Options - Part 1



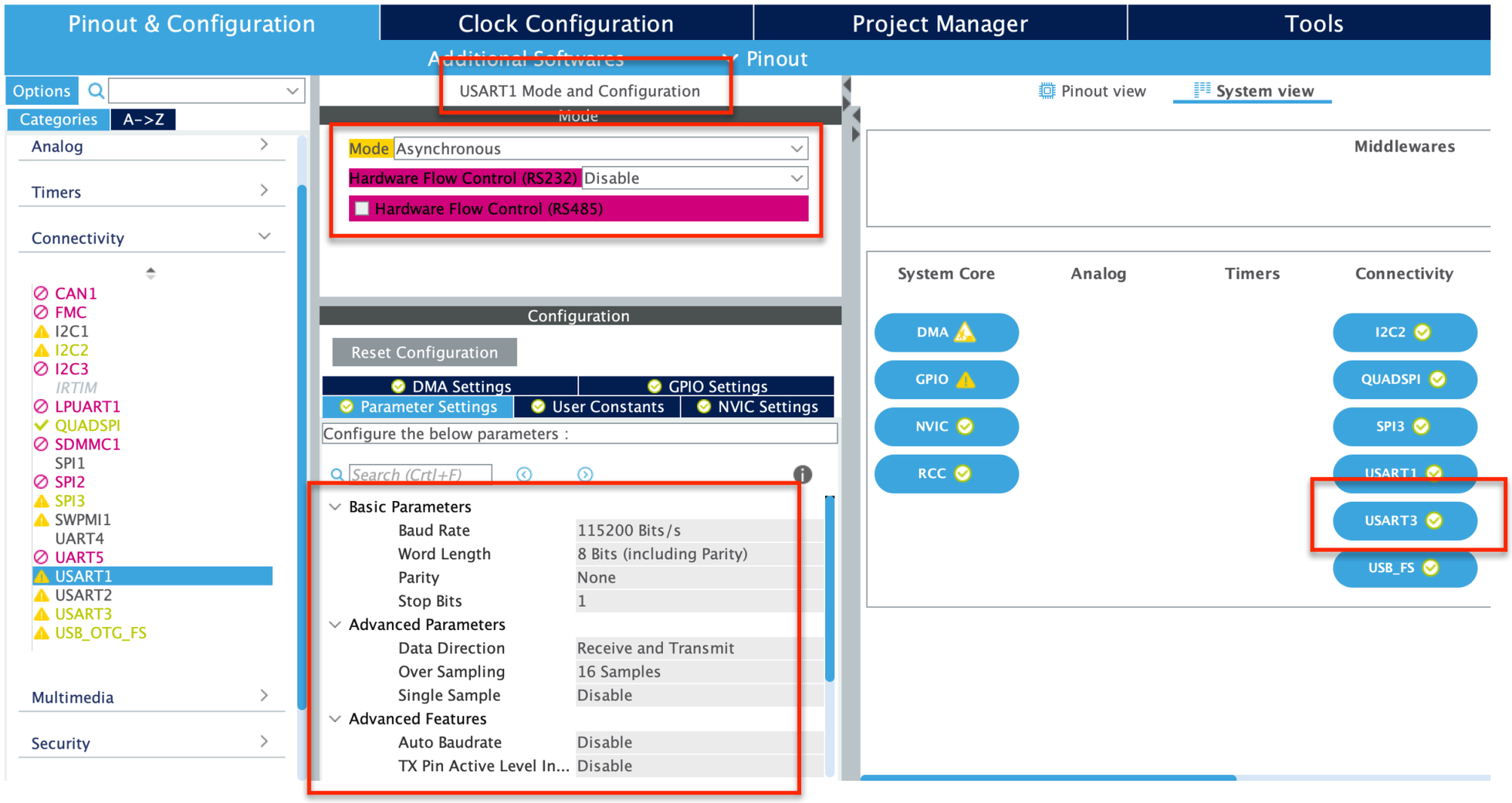
USART1 Options - Part 2



Step: Select “USART3”



USART3 Options - Part 1



USART3 Options - Part 2

Configuration

Reset Configuration

Parameter Settings | User Constants | NVIC Settings | DMA Settings | GPIO Settings

Search Signals

Search (Crtl+F)

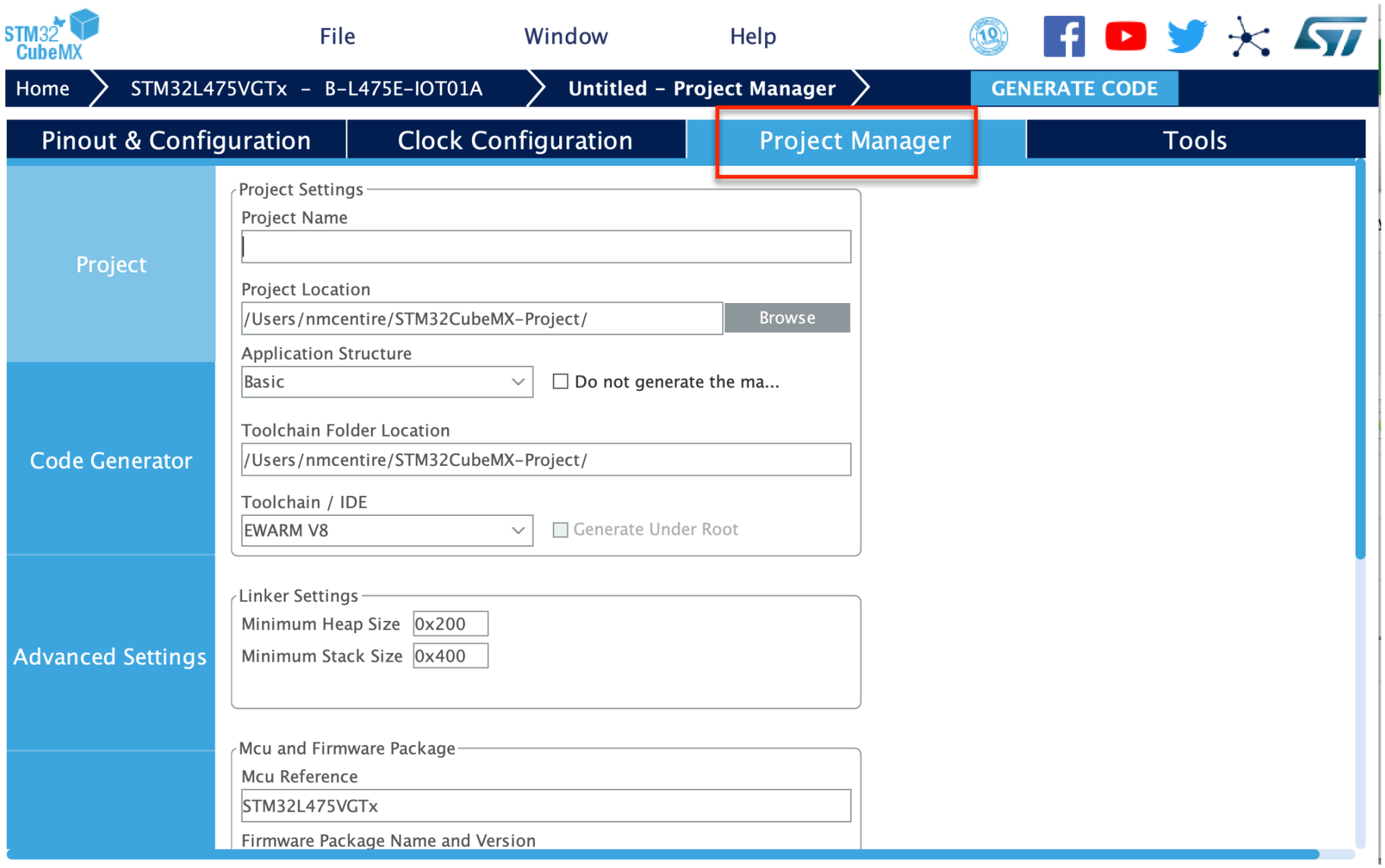
Show only Modified Pins

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]	☑
PD9	USART3_RX	n/a	Alternate Func...	No pull-up an...	Very High	n/a	INTERNAL_UART3_RX [ISM43362_TX]	☑

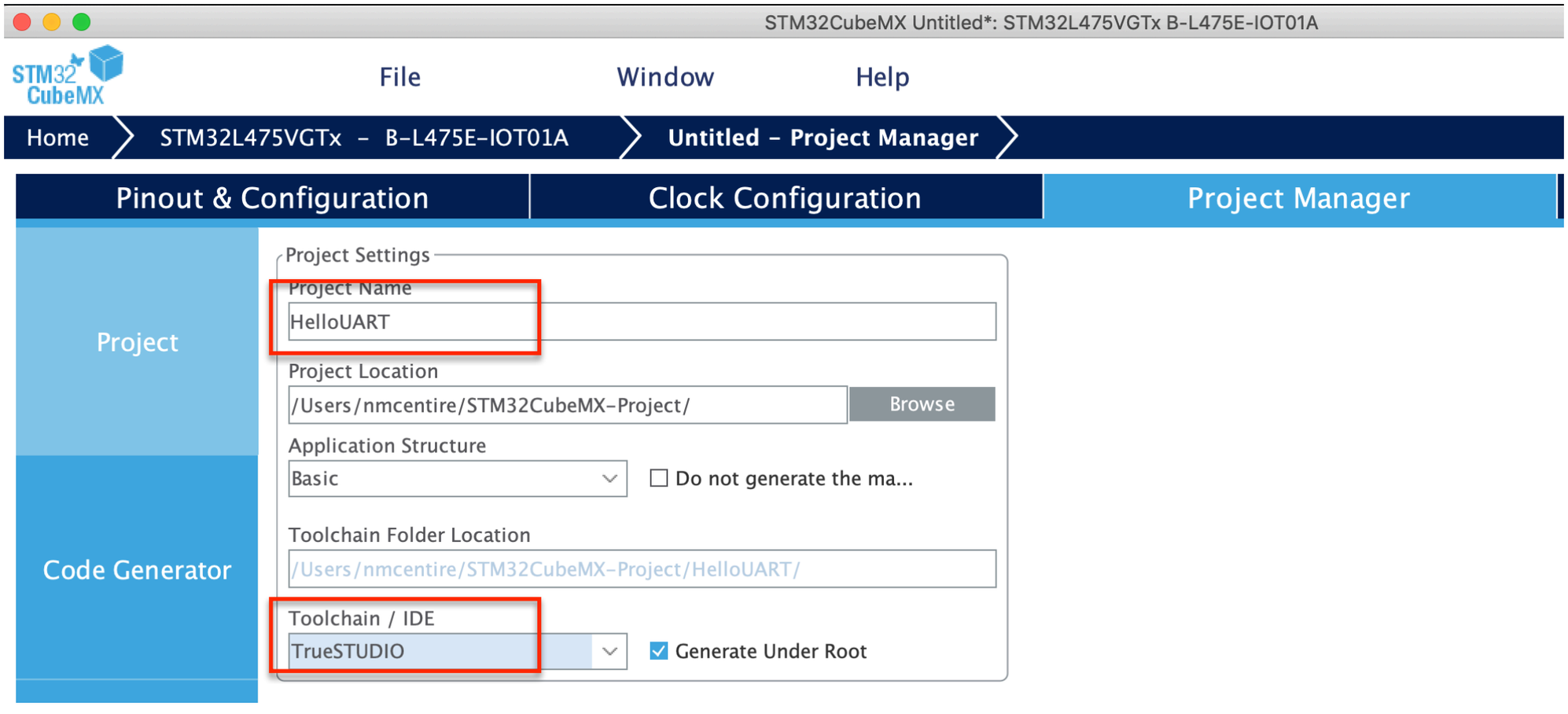
ISM43362 - Wifi Module

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

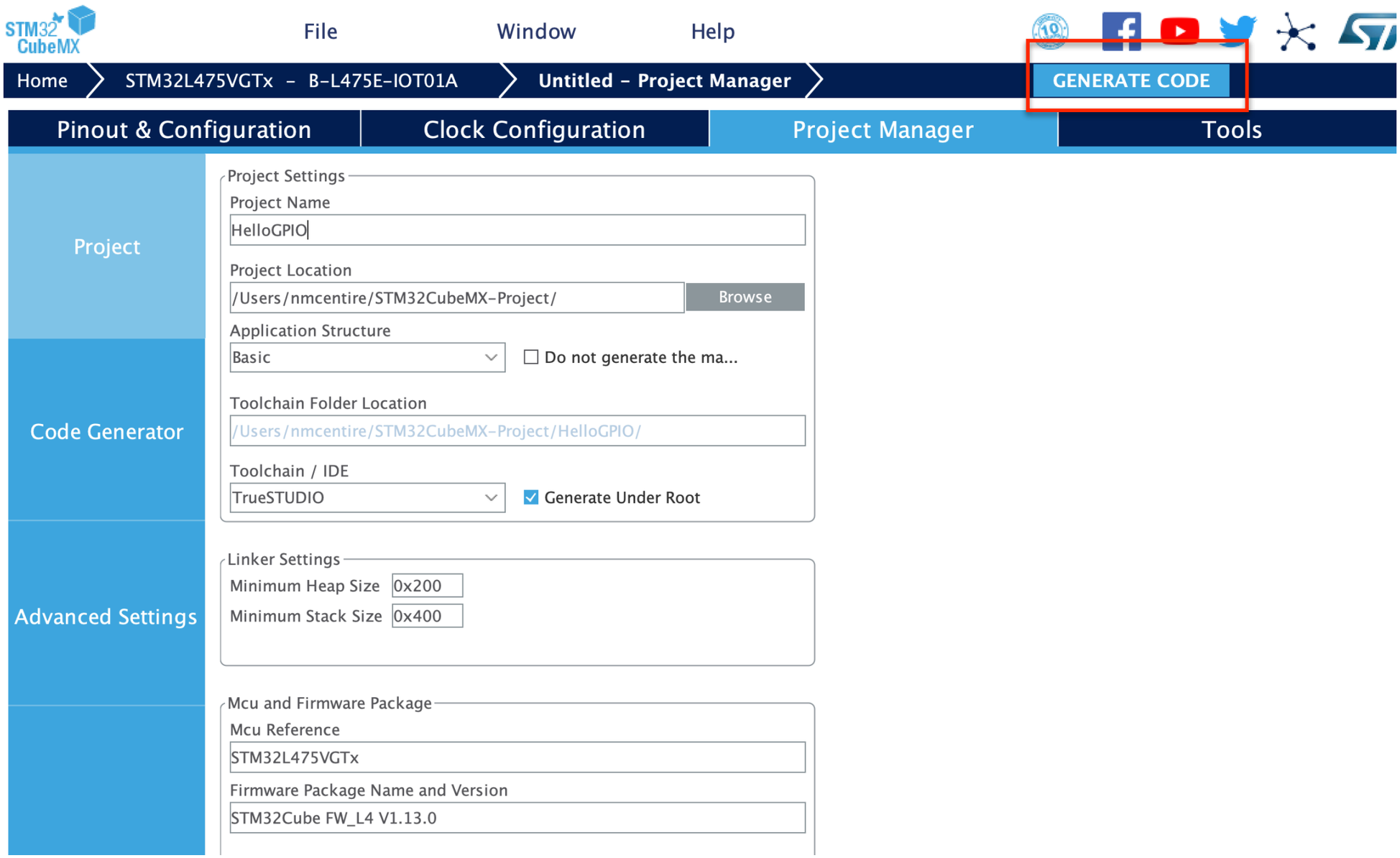
Step: Observe “Project Manager”



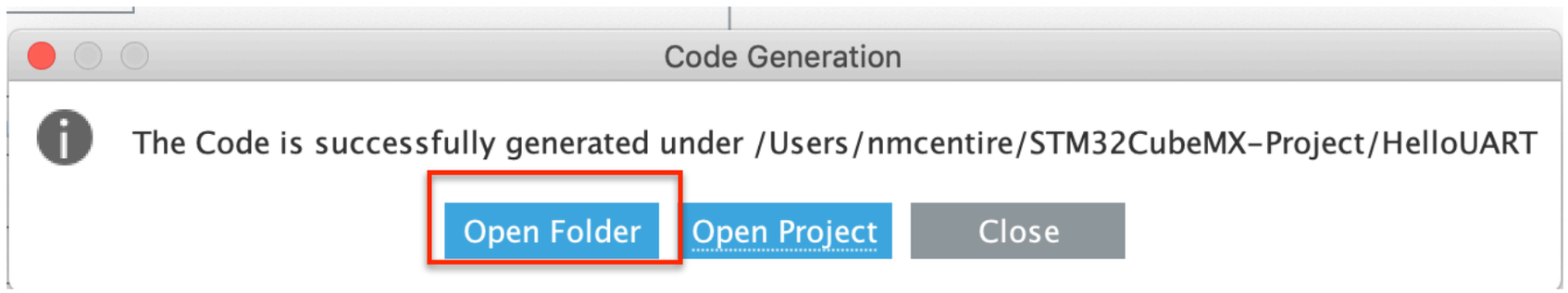
Step: Enter “HelloUART” for Project Name, TrueStudio Toolchain



Step: Click on “Generate Code”

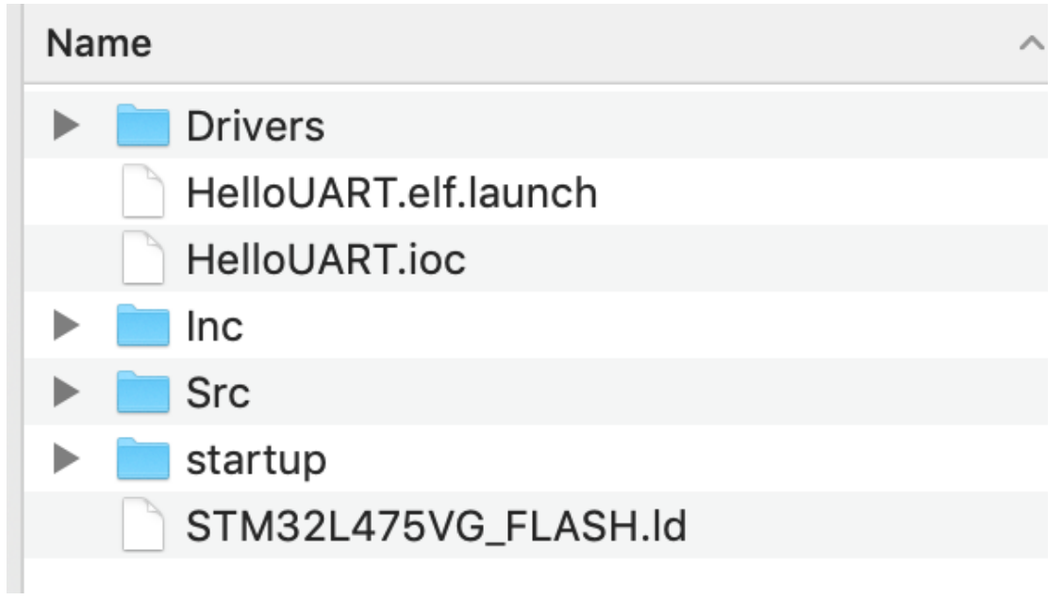


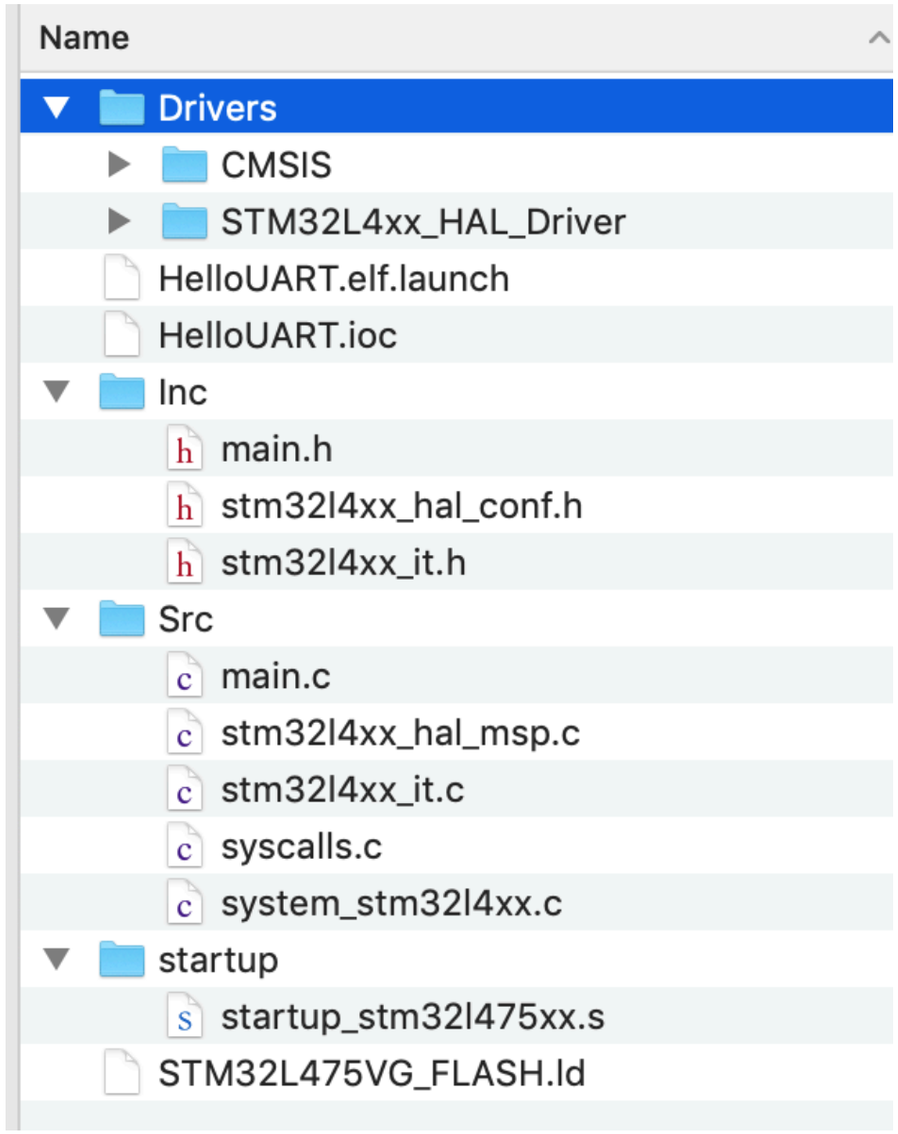
Step: Select “Open Folder”



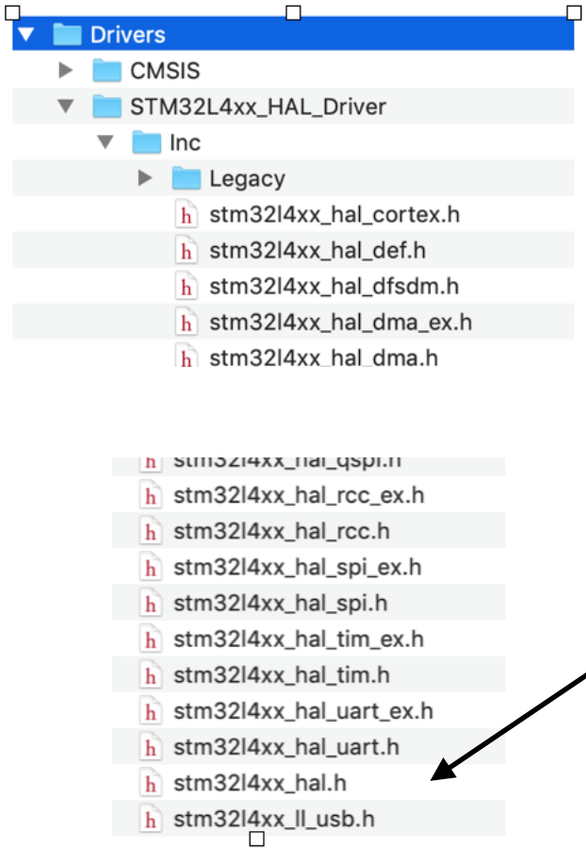
Tour of Generated Project

Step: View Open Folder





Step: View “Drivers/ STM32L4xx_HAL_Driver/Inc” directory



Observe stm32l4xx_hal_uart.h

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

```
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
412    /* USER CODE END USART3_Init 1 */
413    huart3.Instance = USART3;
414    huart3.Init.BaudRate = 115200;
415    huart3.Init.WordLength = UART_WORDLENGTH_8B;
416    huart3.Init.StopBits = UART_STOPBITS_1;
417    huart3.Init.Parity = UART_PARITY_NONE;
418    huart3.Init.Mode = UART_MODE_TX_RX;
419    huart3.Init.HwFlowCtl = UART_HWCONTROL_NONE;
420    huart3.Init.OverSampling = UART_OVERSAMPLING_16;
421    huart3.Init.OneBitSampling = UART_ONE_BIT_SAMPLE_DISABLE;
422    huart3.AdvancedInit.AdvFeatureInit = UART_ADVFEATURE_NO_INIT;
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
```

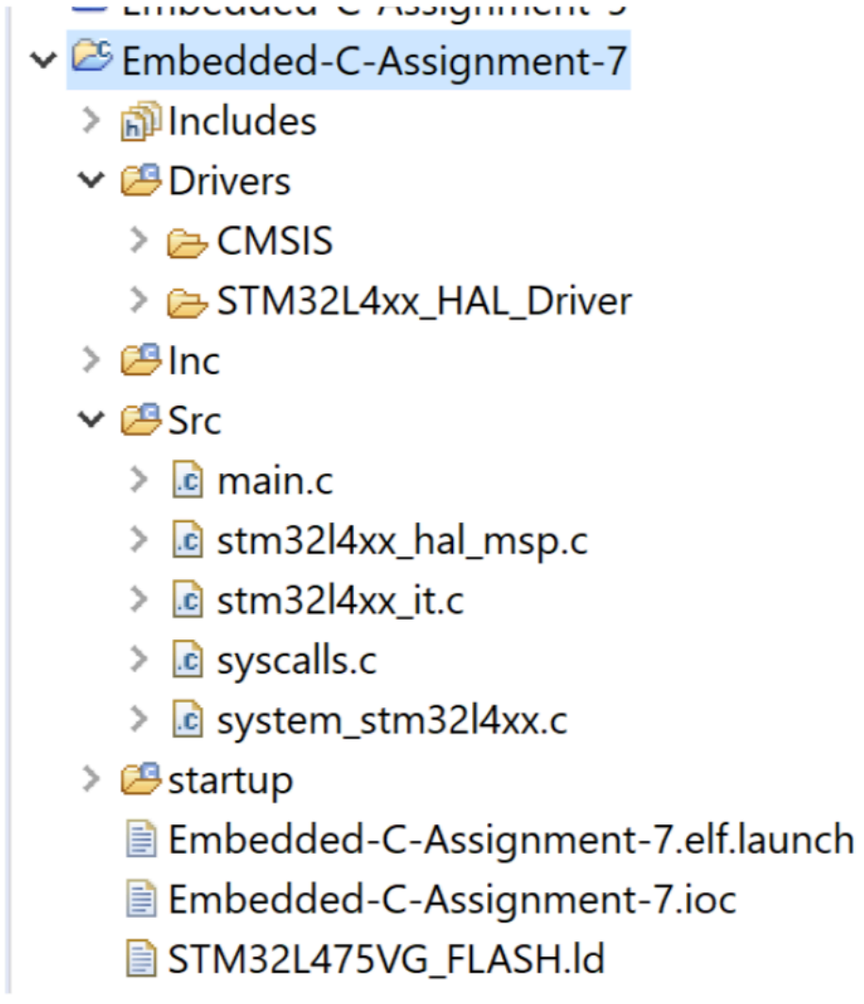
```
536    /*Configure GPIO pins : ARD_D1_Pin ARD_D0_Pin */
537    GPIO_InitStruct.Pin = ARD_D1_Pin|ARD_D0_Pin;
538    GPIO_InitStruct.Mode = GPIO_MODE_AF_PP;
539    GPIO_InitStruct.Pull = GPIO_NOPULL;
540    GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_VERY_HIGH;
541    GPIO_InitStruct.Alternate = GPIO_AF8_UART4;
542    HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
```

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

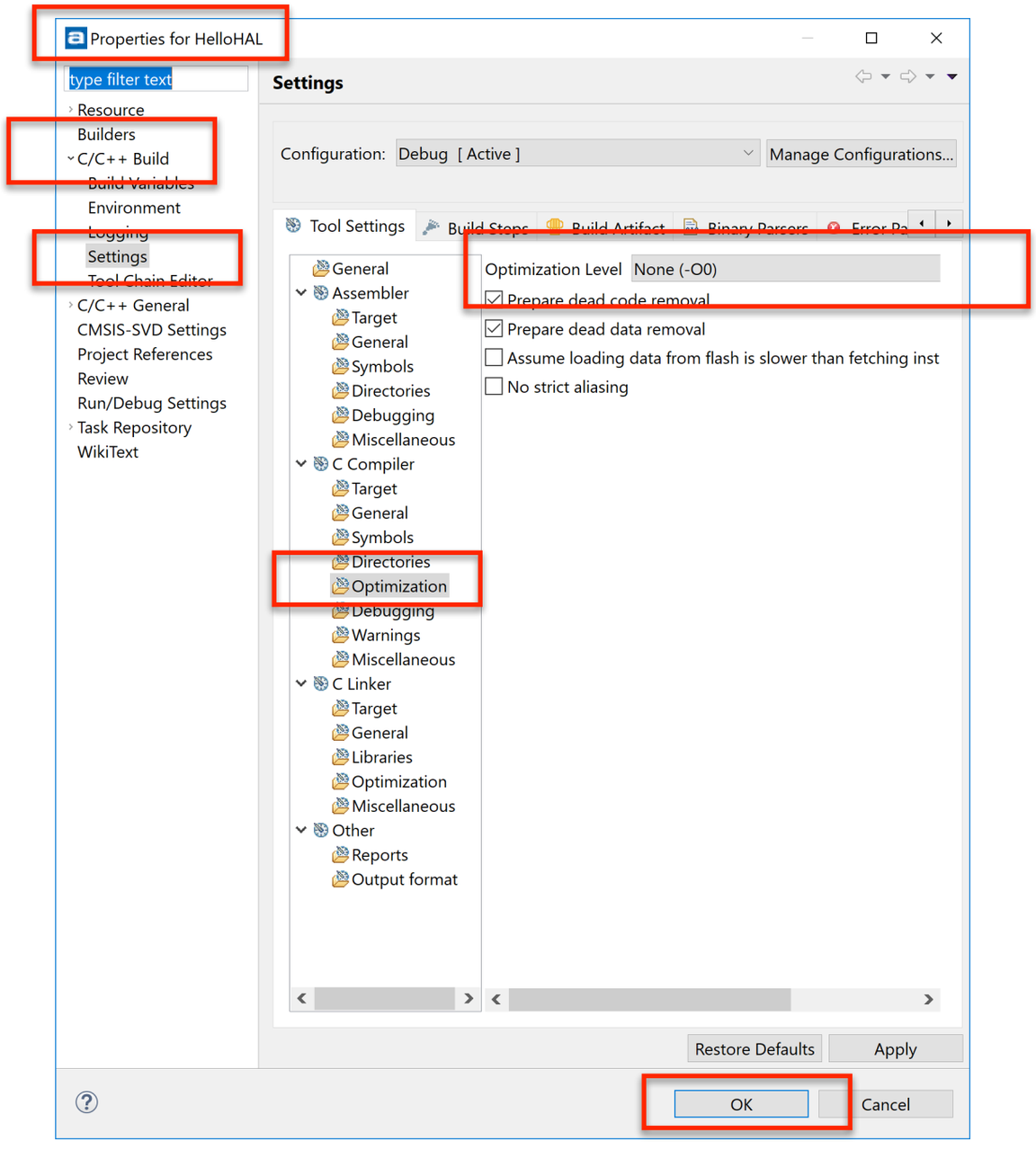
Step: Project



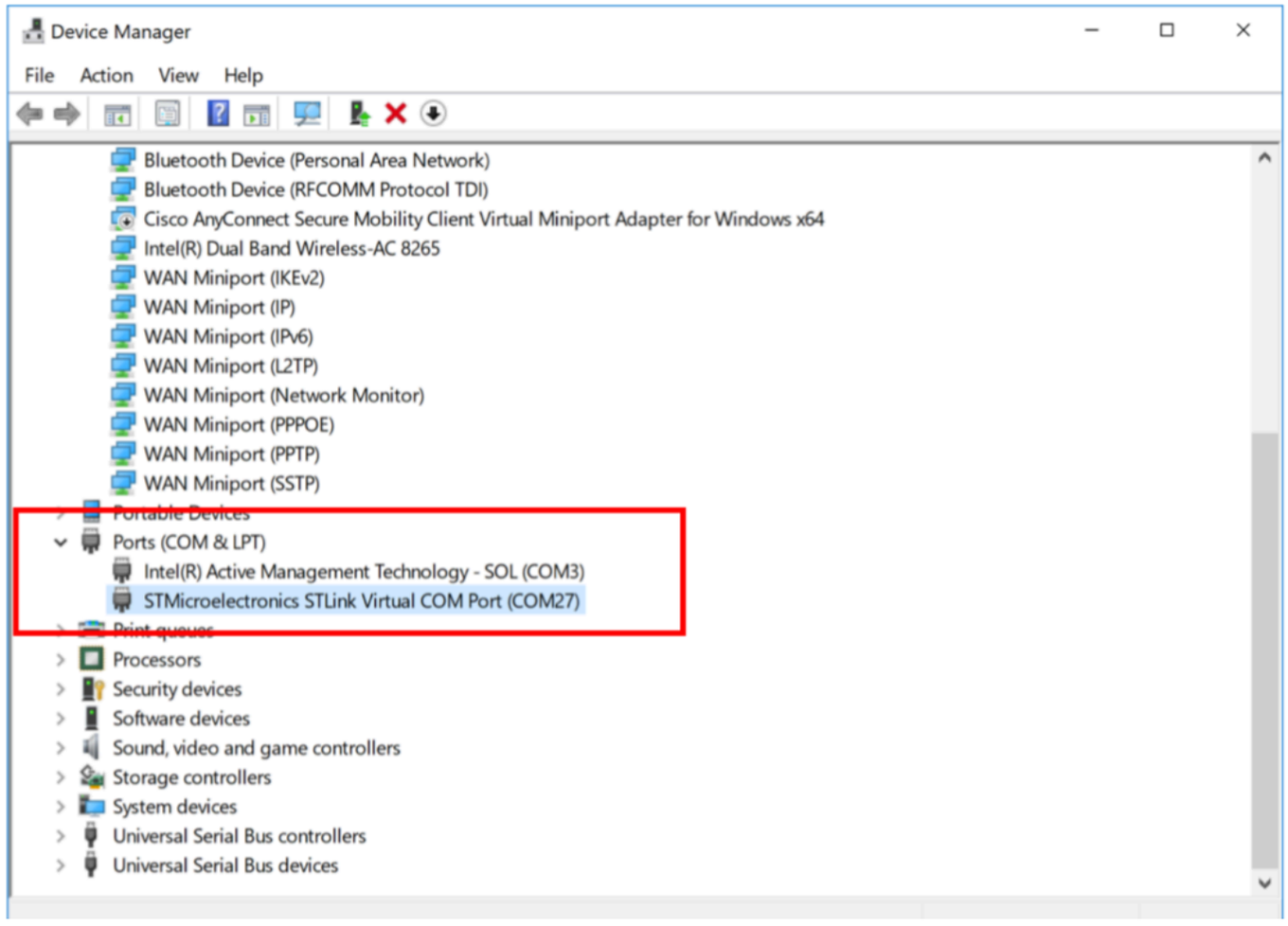
Step: Edit main.c

```
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   }
```

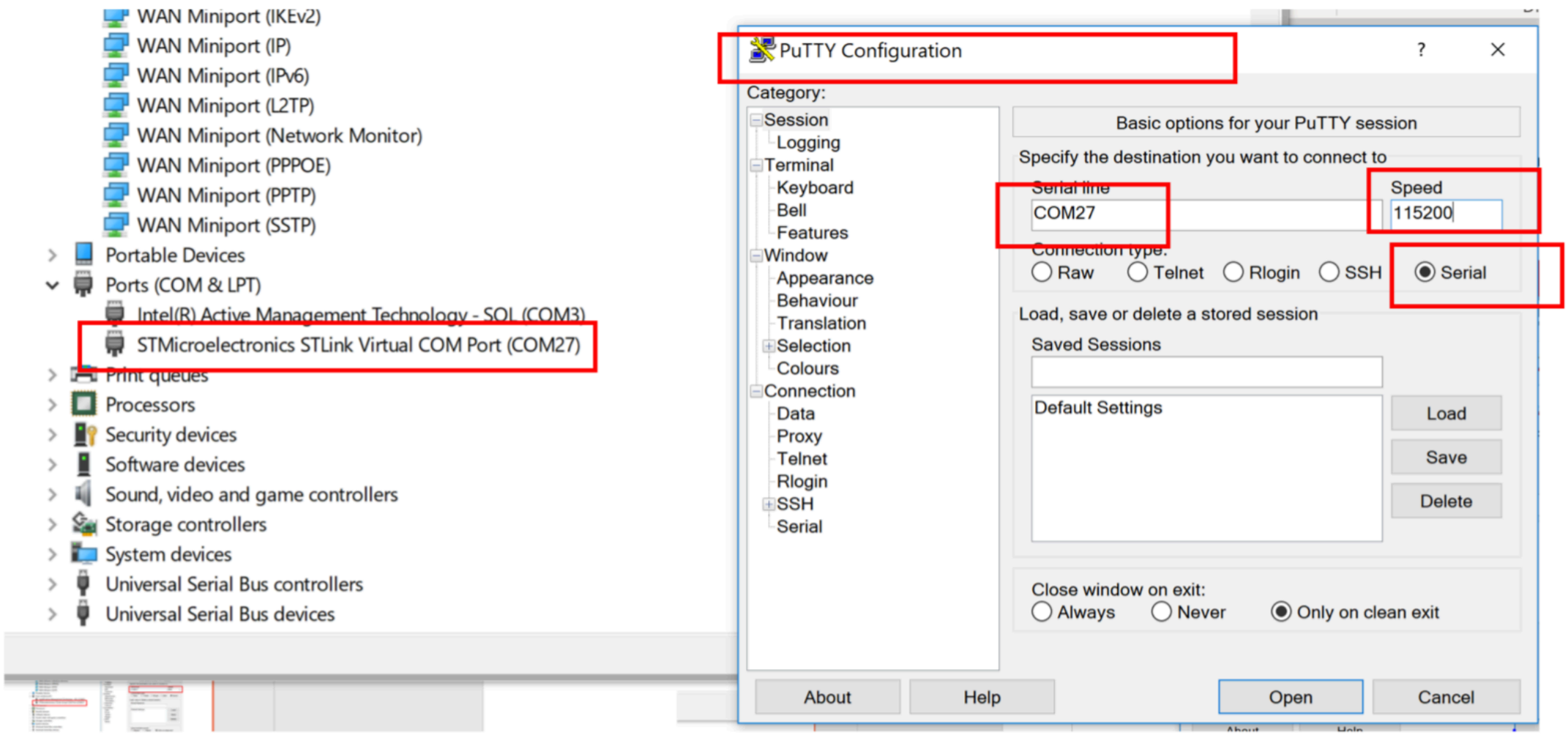
Step: Project, Properties, C/C++ Build, Settings, C Compiler, Optimization, None



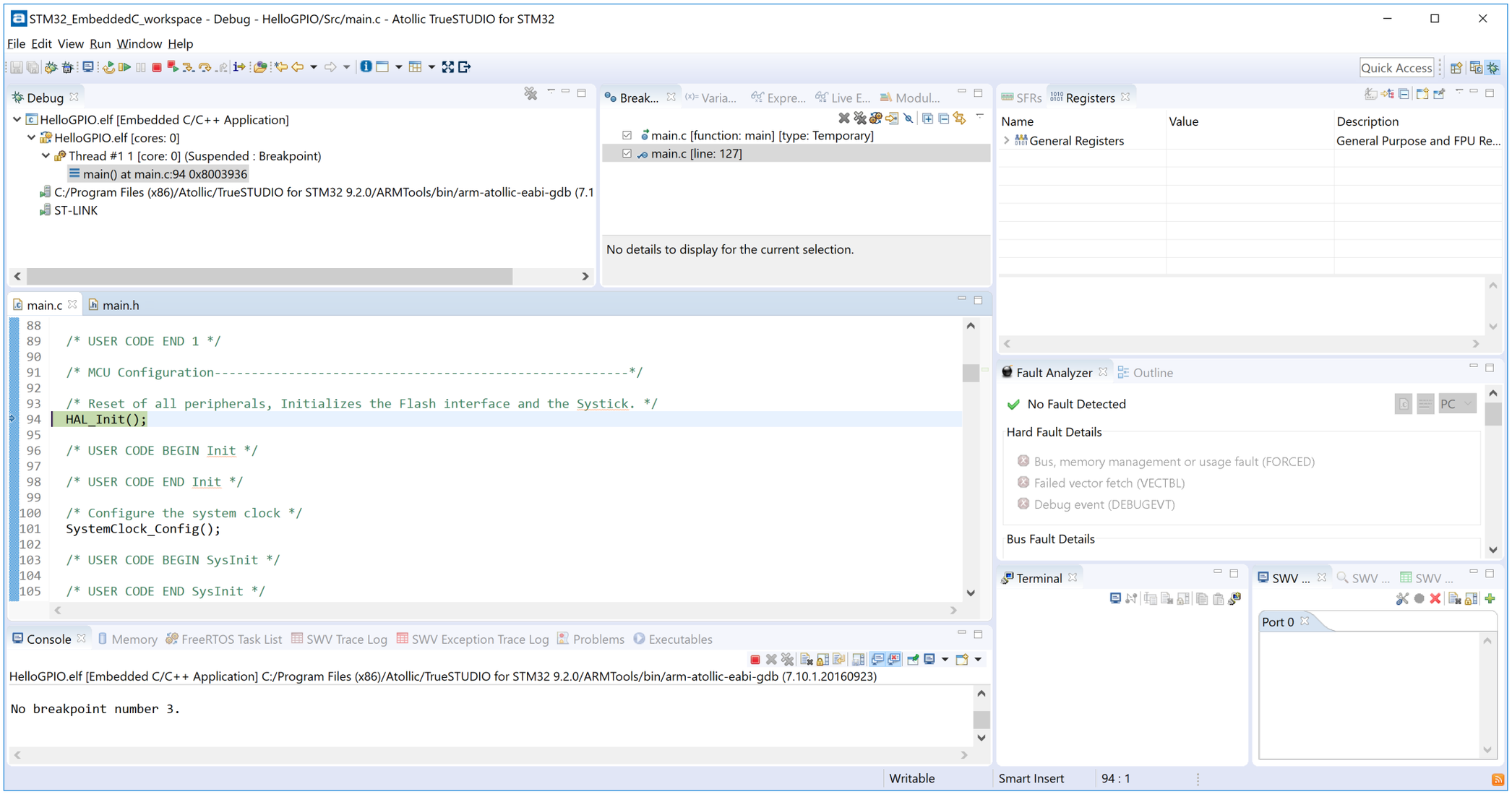
Step: Plug in STM Board, and observe Device Manager



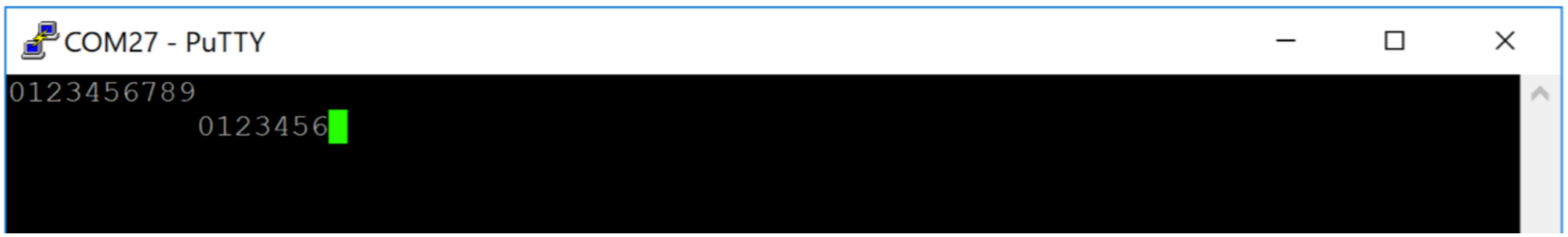
Step: Startup PuTTY, and configure for STM COM Port



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