

# **Lesson 6 - ADC and Embedded C**

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

# Contents

- Introduction to ADC and Embedded C
- STM32CubeMX and ADC Code Generation
- Tour of ADC
- TrueStudio and ADC
  - Read ADC value

[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>6</b>	<b>HAL ADC Generic Driver.....</b>	<b>97</b>
6.1	ADC Firmware driver registers structures .....	97
6.1.1	ADC_OversamplingTypeDef .....	97
6.1.2	ADC_InitTypeDef.....	97
6.1.3	ADC_ChannelConfTypeDef .....	100
6.1.4	ADC_AnalogWDGConfTypeDef.....	101
6.1.5	ADC_InjectionConfigTypeDef .....	102
6.1.6	ADC_HandleTypeDef.....	102
6.2	ADC Firmware driver API description.....	102
6.2.1	ADC peripheral features .....	102
6.2.2	How to use this driver .....	103
6.2.3	Peripheral Control functions .....	105
6.2.4	Peripheral state and errors functions .....	105
6.2.5	Detailed description of functions .....	105
6.3	ADC Firmware driver defines .....	114
6.3.1	ADC .....	114

# Introduction to ADC and Embedded C

# 28.2.1 ADC Peripheral Features

- ADC = Analog to Digital Conversion
- **12-bit**, 10-bit, 8-bit, 6-bit resolution
- Interrupt at end of conversion
- Single or continuous conversion
- Programmable sample time
- External trigger (timer or EXTI)

## 28.2.2 How to use driver - Part 1

- Step 1. Enable ADC Interface
  - `_HAL_RCC_GPIOx_CLK_ENABLE()`
- Step 2. Configure ADC Pins
  - `HAL_GPIO_Init()`
- Step 3. (If using input for interrupts). Configure NVIC IRQ Priority mapped to the EXTI line
  - `HAL_NVIC_SetPriority()`
  - `HAL_NVIC_EnableIRQ()`

**NVIC** =  
Nested  
Vector  
Interrupt  
Controller

# HAL ADC Data Types

Drivers/STM32L4xx\_HAL\_Driver/Inc/**stm32l4xx\_adc.h**

# ADC\_OversamplingTypeDef

- ```
typedef struct {
    uint32_t Ratio;
    uint32_t RightBitShift;
    uint32_t TriggeredMode;
    uint32_t OversamplingStopReset;
} ADC_OversamplingTypeDef
```

# ADC\_InitTypeDef

- ```
typedef struct {
    uint32_t ClockPrescaler;
    uint32_t Resolution;
    uint32_t DataAlign;
    uint32_t ScanConvMode;
    uint32_t EOCSelection;
    FunctionalState LowPowerAutoWait;
    FunctionalState ContinuousConvMode;
    uint32_t NbrOfConversion;
    FunctionalState DiscontinuousConvMode;
    uint32_t NbrOfDiscConversion;
    uint32_t ExternalTrigConv;
    uint32_t ExternalTrigConvEdge;
    FunctionalState DMAContinuousRequests;
    uint32_t Overrun;
    FunctionalState OversamplingMode;
    ADC_OversamplingTypeDef Oversampling;
} ADC_OversamplingTypeDef
```

# ADC\_ChannelConfTypeDef

- ```
typedef struct {
    uint32_t Channel;
    uint32_t Rank;
    uint32_t SamplingTime;
    uint32_t SingleDiff;
    uint32_t OffsetNumber;
    uint32_t Offset;
} ADC_ChannelConfTypeDef
```

# ADC\_AnalogWDGConfTypeDef

- ADC Analog Watchdog
- ```
typedef struct {
    uint32_t WatchDogNumber;
    uint32_t WatchDogMode;
    uint32_t Channel;
    FunctionalState ITMode;
    uint32_t HighThreshold;
    uint32_t LowThreshold;
} ADC_AnalogWDGConfTypeDef
```

# HAL ADC State Machine

## States of ADC Global Scope

- HAL\_ADC\_STATE\_RESET
- HAL\_ADC\_STATE\_READY
- HAL\_ADC\_STATE\_BUSY\_INTERNAL
- HAL\_ADC\_STATE\_TIMEOUT
- Example
  - if (HAL\_ADC\_GET\_STATE(hadc1) &  
HAL\_ADC\_STATE\_READY != 0) { ... }

# HAL ADC State Machine

## States of ADC Errors

- HAL\_ADC\_STATE\_ERROR\_INTERNAL
- HAL\_ADC\_STATE\_ERROR\_CONFIG
- HAL\_ADC\_STATE\_ERROR\_DMA

# HAL ADC State Machine

## States of ADC **Group Regular**

- HAL\_ADC\_STATE\_REG\_BUSY
- HAL\_ADC\_STATE\_REG\_EOC
  - End of Conversion
- HAL\_ADC\_STATE\_REG\_OVR
  - Overrun
- HAL\_ADC\_STATE\_REG\_EOSMP
  - End of Sample

# HAL ADC State Machine

## States of ADC **Analog Watchdogs**

- These are watchdogs for out-of-window occurrence
- HAL\_ADC\_STATE\_AWD1
- HAL\_ADC\_STATE\_AWD2
- HAL\_ADC\_STATE\_AWD3

# ADC\_HandleTypeDefDef

- ```
typedef struct {
    ADC_TypeDef *instance; //Register base address
    ADC_InitTypeDef Init;
    DMA_HandleTypeDef *DMA_Handle;
    HAL_LockTypeDef Lock;
    __IO uint32_t State;
    __IO uint32_t ErrorCode;
    ADC_InjectionConfigTypeDef InjectionConfig;
#if (USE_HAL_ADC_REGISTER_CALLBACKS = 1)
    ...
#endif
} ADC_HandleTypeDef
```

# ADC\_RESOLUTION

- ADC\_RESOLUTION\_12B
- ADC\_RESOLUTION\_10B
- ADC\_RESOLUTION\_8B
- ADC\_RESOLUTION\_6B

# ADC\_DATAALIGN

- ADC\_DATAALIGN\_RIGHT
- ADC\_DATAALIGN\_LEFT

# ADC\_CHANNEL

- ADC\_CHANNEL\_0
- ADC\_CHANNEL\_1
- ADC\_CHANNEL...
- ADC\_CHANNEL\_VREFINT
  - Internal voltage reference
- ADC\_CHANNEL\_TEMPSENSOR
  - Internal temperature sensor
- ADC\_CHANNEL\_VBAT
  - VBAT voltage through a divider ladder of factor 1/3 to have VBAT always below VDDA

# HAL ADC Functions

# HAL ADC

## Initialization/DelInitialization

- HAL\_StatusTypeDef  
HAL\_ADC\_Init(ADC\_HandleTypeDef \*hadc)
- HAL\_StatusTypeDef  
HAL\_ADC\_DelInit(ADC\_HandleTypeDef \*hadc)
- void  
HAL\_ADC\_MspInit(ADC\_HandleTypeDef \*hadc)
- void  
HAL\_ADC\_MspDelInit(ADC\_HandleTypeDef \*hadc)

# HAL ADC Callbacks

- HAL\_StatusTypeDef  
`HAL_ADC_RegisterCallback(ADC_HandleTypeDef *adc, HAL_ADC_CallbackIDTypeDef CallbackID, pADC_CallbackTypeDef pCallback)`
- HAL\_StatusTypeDef  
`HAL_ADC_UnRegisterCallback(ADC_HandleTypeDef *adc, HAL_ADC_CallbackIDTypeDef CallbackID)`

# HAL ADC

## Blocking Mode - Polling

- HAL\_StatusTypeDef  
HAL\_ADC\_Start(ADC\_HandleTypeDef \*hadc)
- HAL\_StatusTypeDef  
HAL\_ADC\_Stop(ADC\_HandleTypeDef \*hadc)
- HAL\_StatusTypeDef  
HAL\_ADC\_PollForConversion(ADC\_HandleTypeDef \*hadc,  
uint32\_t Timeout)
- HAL\_StatusTypeDef  
HAL\_ADC\_PollForEvent(ADC\_HandleTypeDef \*hadc, uint32\_t  
EventType, uint32\_t Timeout)

# HAL ADC

## Non Blocking Mode - Interrupt

- HAL\_StatusTypeDef  
HAL\_ADC\_Start\_IT(ADC\_HandleTypeDef \*hadc)
- HAL\_StatusTypeDef  
HAL\_ADC\_Stop\_IT(ADC\_HandleTypeDef \*hadc)

# HAL ADC

## Non Blocking Mode - **DMA**

- HAL\_StatusTypeDef  
HAL\_ADC\_Start\_**DMA**(ADC\_HandleTypeDef \*hadc,  
uint32\_t \*data, uint32\_t length)
- HAL\_StatusTypeDef  
HAL\_ADC\_Stop\_**DMA**(ADC\_HandleTypeDef \*hadc)

# HAL ADC

## Retrieve Conversion Value (Polled and Interrupt Driven)

- `uint32_t  
HAL_ADC_GetValue(ADC_HandleTypeDef *hadc)`

# HAL ADC IRQ Handler

- void  
HAL\_ADC\_IRQHandler(ADC\_HandleTypeDef  
\*hadc)

# HAL ADC Peripheral Control

- HAL\_StatusTypeDef  
HAL\_ADC\_ConfigChannel(ADC\_HandleTypeDef  
\*hdac, ADC\_ChannelConfTypeDef \*sConfig)
- HAL\_StatusTypeDef  
HAL\_ADC\_AnalogWDGConfig(ADC\_HandleTypeDef  
\*hadc, ADC\_AnalogWDGConfTypeDef  
\*AnalogWDGConfig)

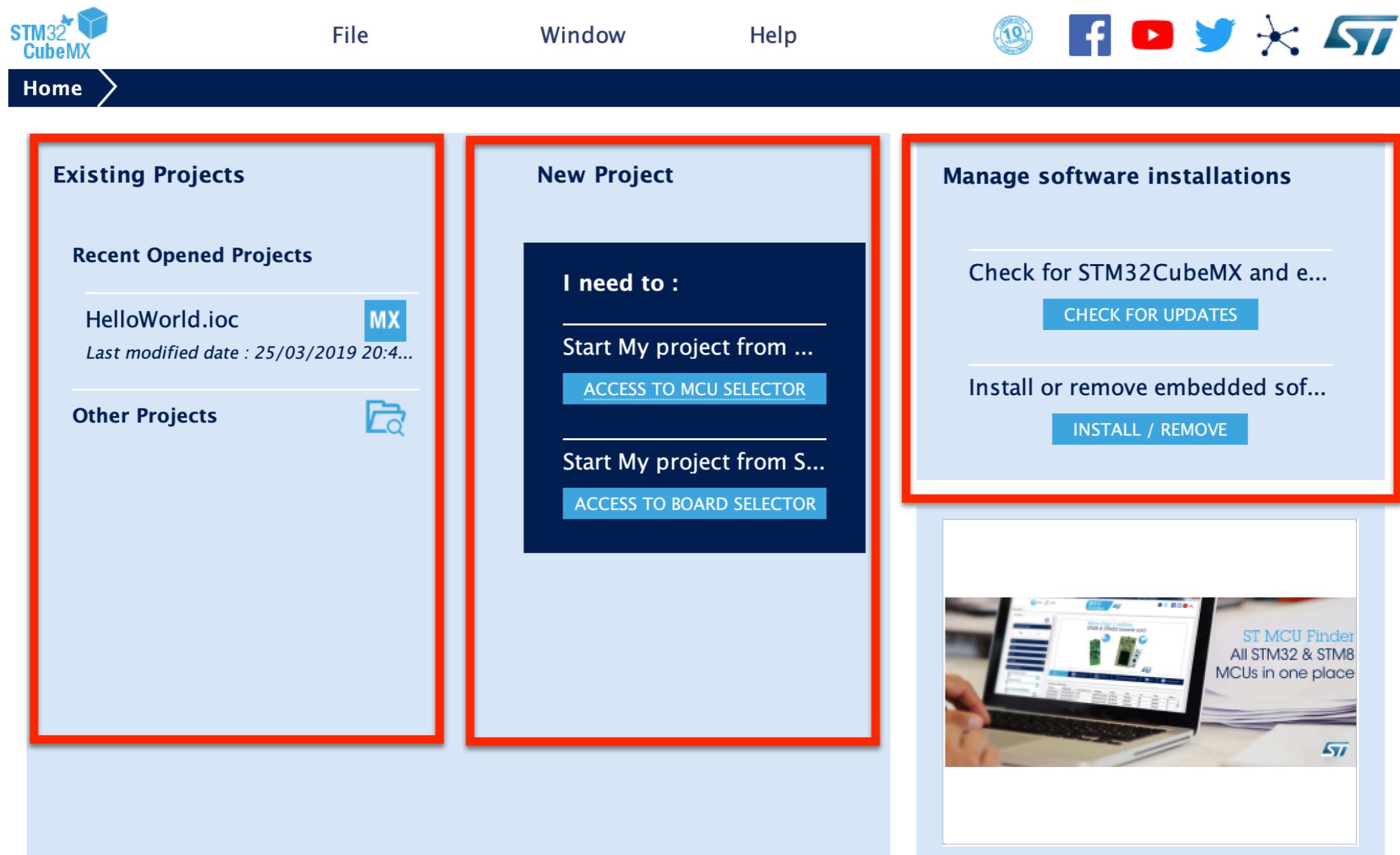
# HAL ADC

## Other Functions

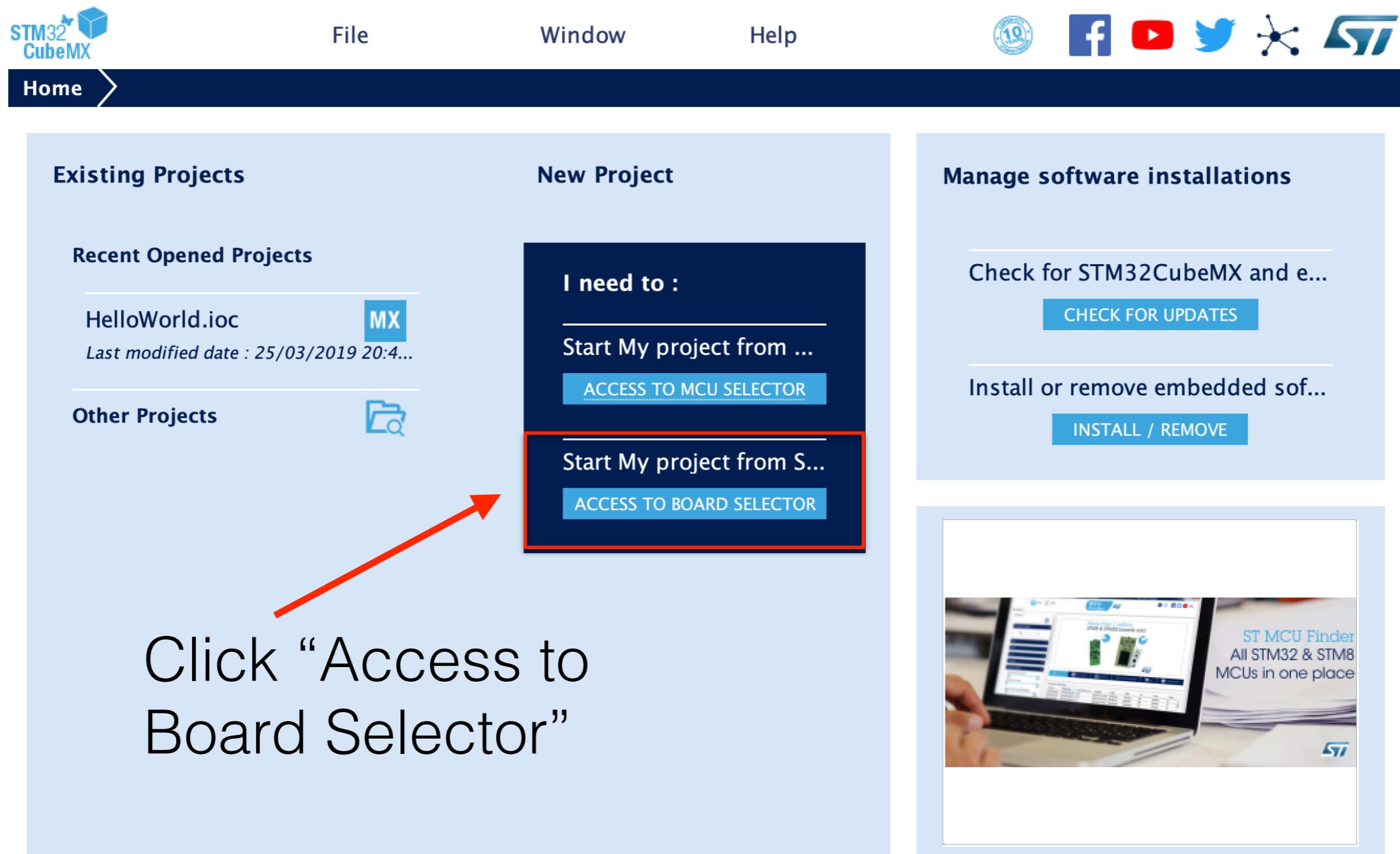
- `uint32_t  
HAL_ADC_GetState(ADC_HandleTypeDef *hadc)`
- `uint32_t  
HAL_ADC_GetError(ADC_HandleTypeDef *hadc)`

# STM32CubeMX and Generation of ADC 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 penguin icon, and the text 'OpenSTLinux Distribution'. A table below lists 'Boards List: 107 items' with columns for Overview, Part No, Type, Marketing Status, Unit Price (US\$), and Mounted Device. Two rows are visible:

| * | 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"

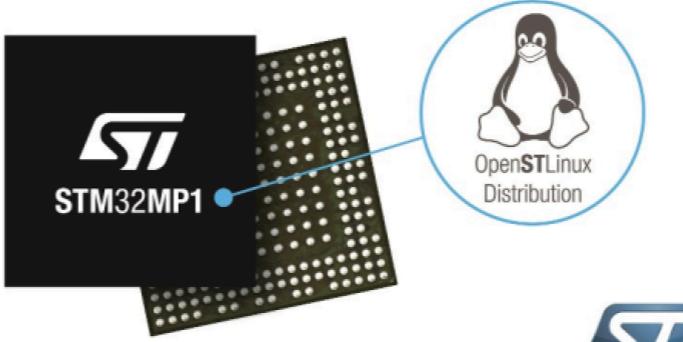
MCU Selector | Board Selector

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**

**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**

**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](#)



arm MBED Enabled



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                |
|---|--------------------------------------------------------------------------------------|----------------|-----------|------------------|-------------------|-------------------------------|
|   |  | B-L475E-IOT01A | 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 is a sidebar with various filters:

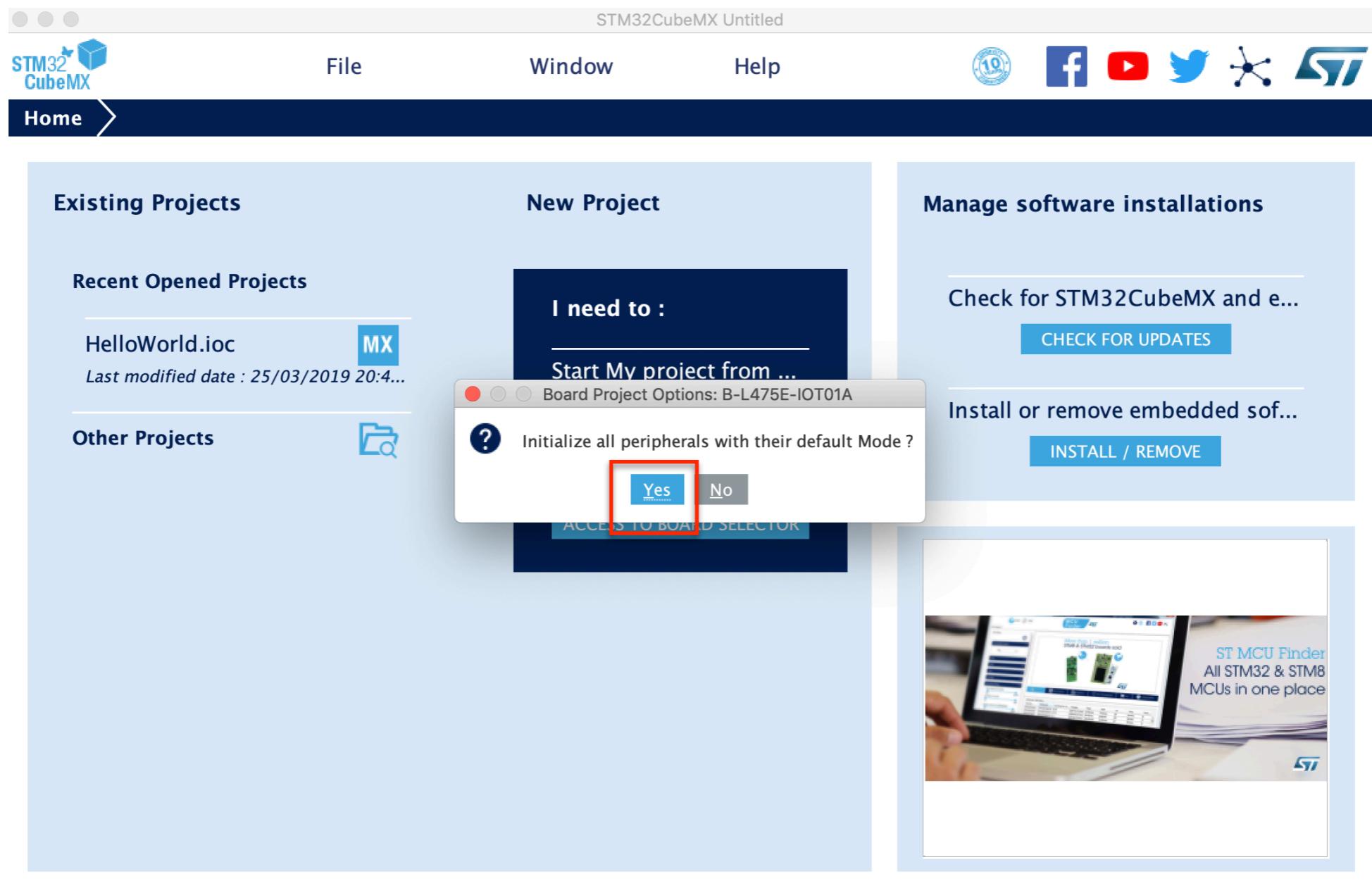
- Board Filters: Includes a star icon, a file icon, a magnifying glass icon, and a refresh icon.
- Part Number Search: A dropdown menu showing "B-L475E-IOT01A".
- Vendor: A dropdown menu with "Check/Uncheck All" and a checkbox for "STMicroelectronics".
- Type: A dropdown menu with "Check/Uncheck All" and a checkbox for "Discovery".
- MCU Series: A dropdown menu with "Check/Uncheck All" and a checkbox for "STM32L4".
- Other: Includes fields for "Price = 53.0" and "Oscillator Freq. = 0 (MHz)".
- Peripheral: A dropdown menu.

The main content area displays the following information for the B-L475E-IOT01A board:

- Features: Large Picture, Docs & Resources (highlighted with a red box), Datasheet, Buy.
- Docs & Resources: DB3143 Discovery kit for IoT node, multi-channel communication with STM32L4 (version 4).
- User manual: UM2153 Discovery kit for IoT node, multi-channel communication with STM32L4 (version 4).
- Boards List: 1 item
- Table:

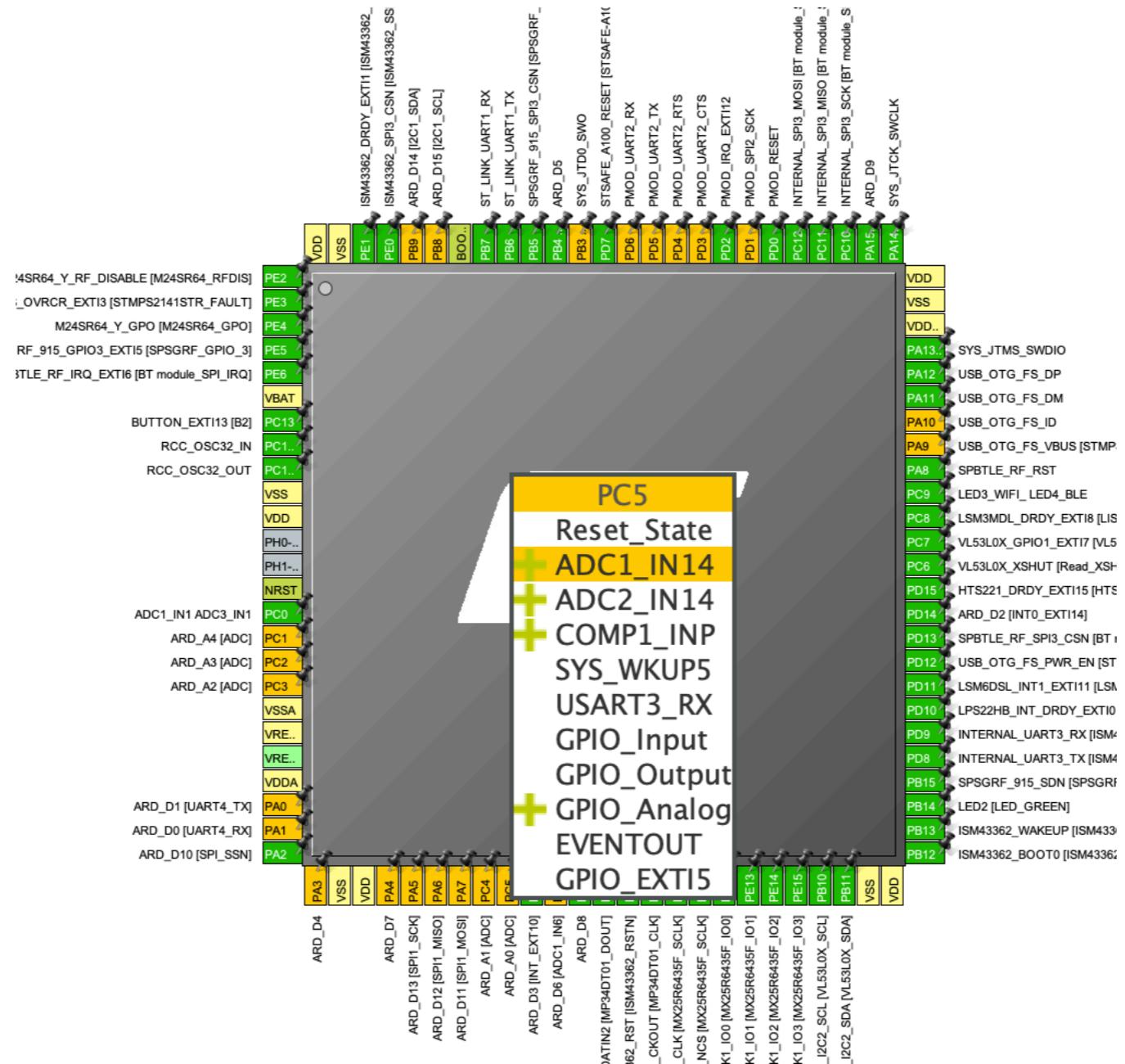
| * | Overview | Part No        | Type      | Marketing Status | Unit Price (US\$) | Mounted Device |
|---|----------|----------------|-----------|------------------|-------------------|----------------|
| ☆ |          | B-L475E-IOT01A | Discovery | Active           | 53.0              | STM32L475VGTx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

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



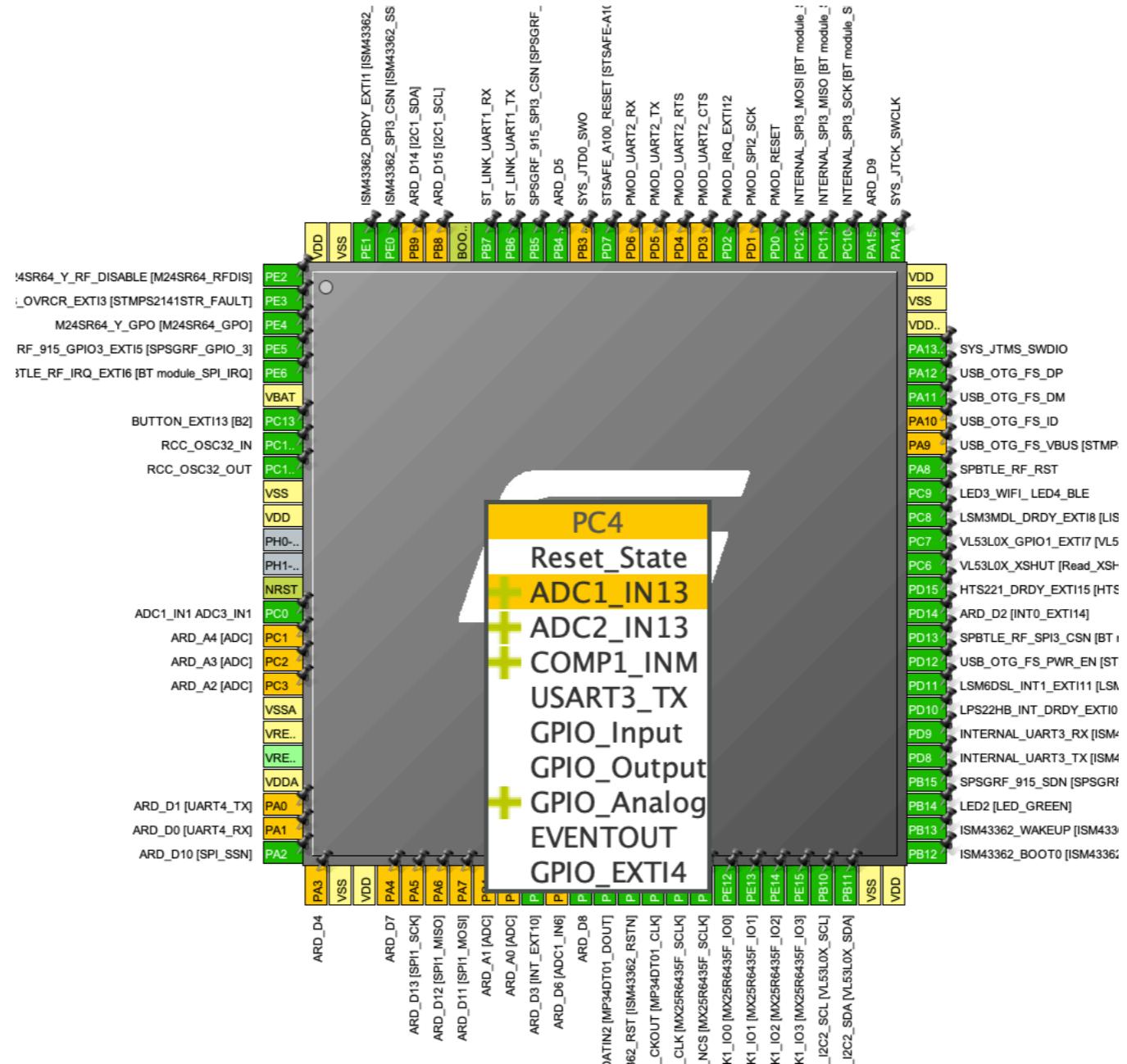
# Step: Pin View

## PC5 - ADC1 IN14 - ARD A0



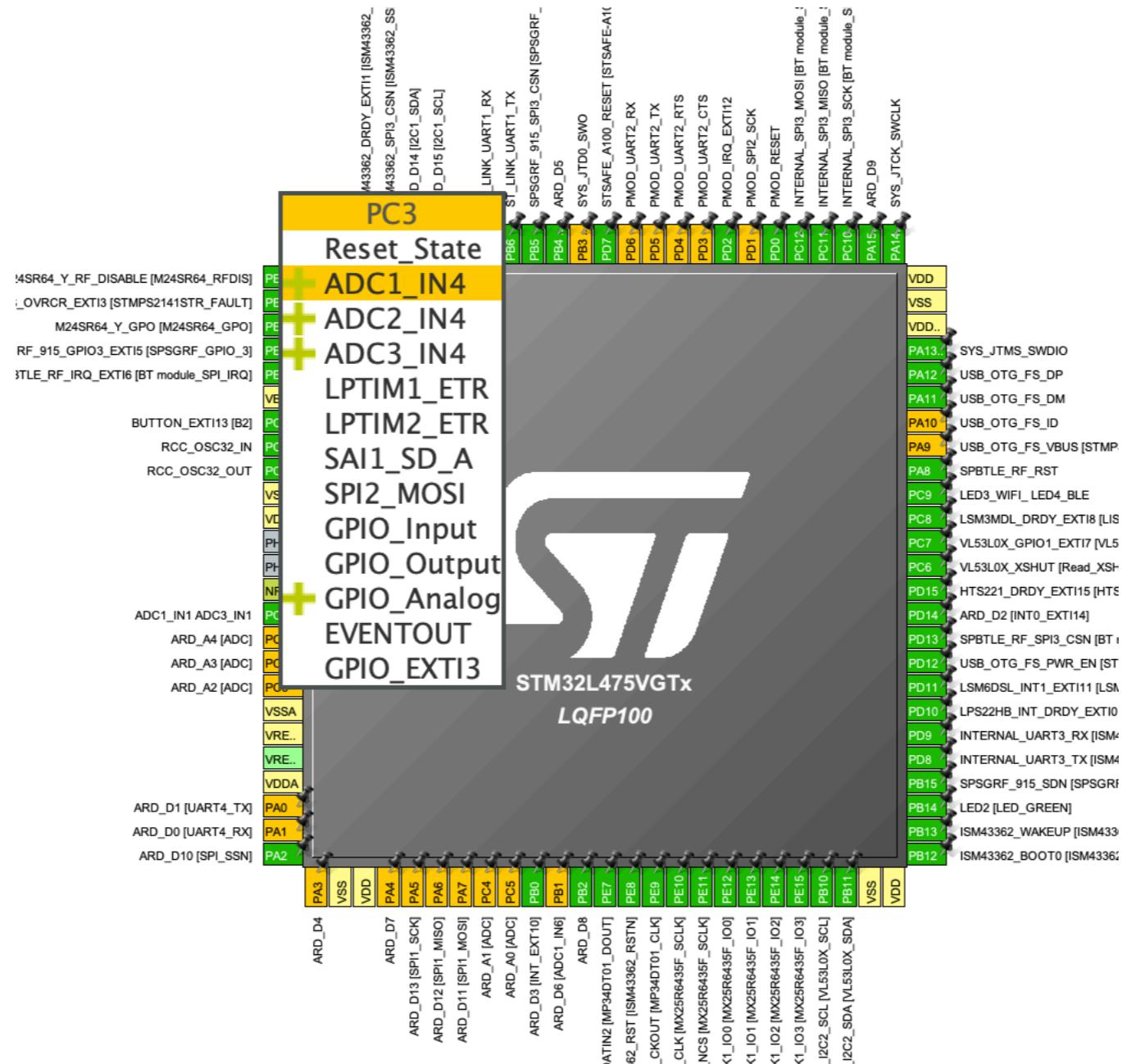
# Step: Pin View

## PC4 - ADC1\_IN13 - ARD\_A1



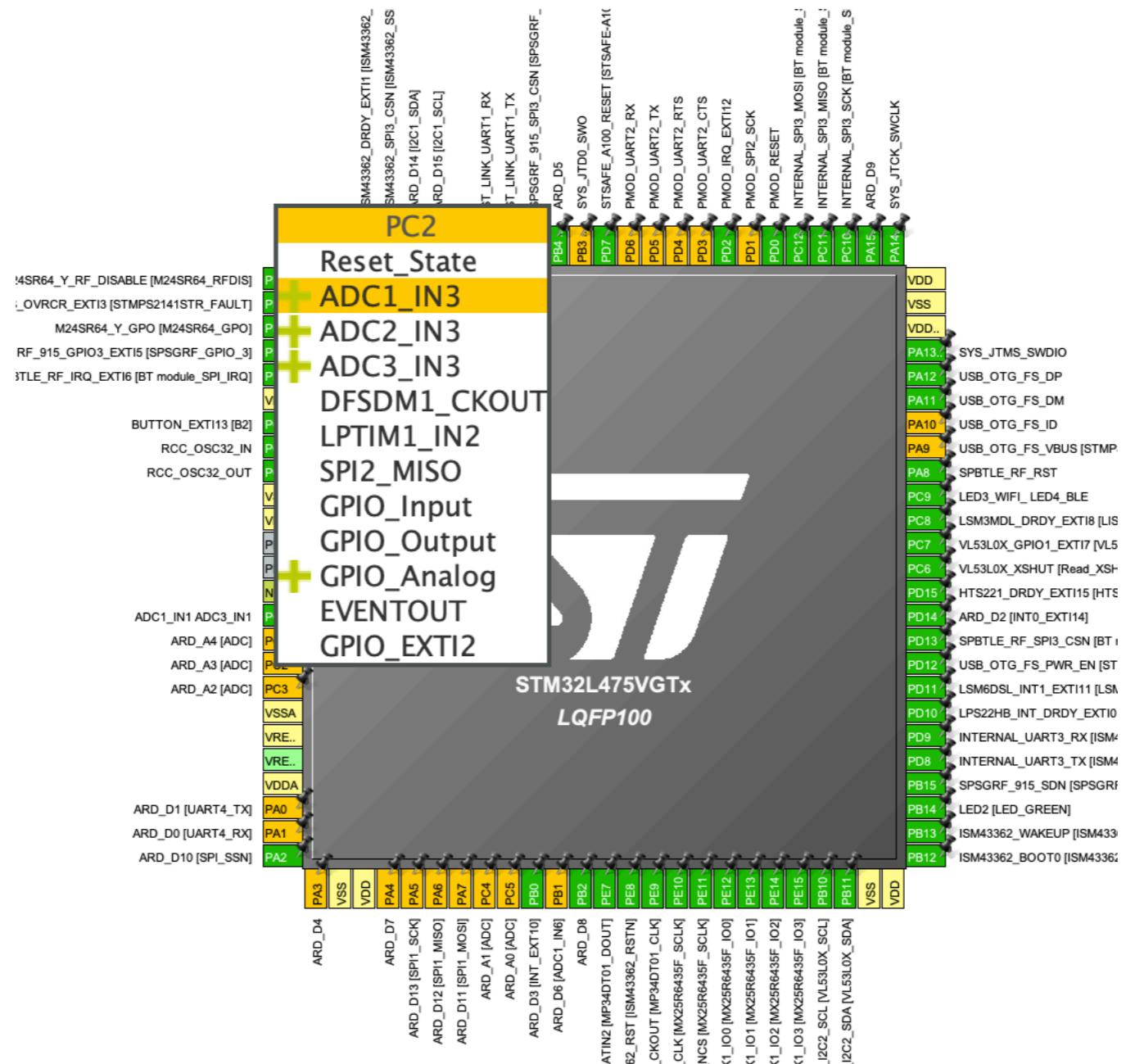
# Step: Pin View

## PC3 - ADC1\_IN4 - ARD\_A2



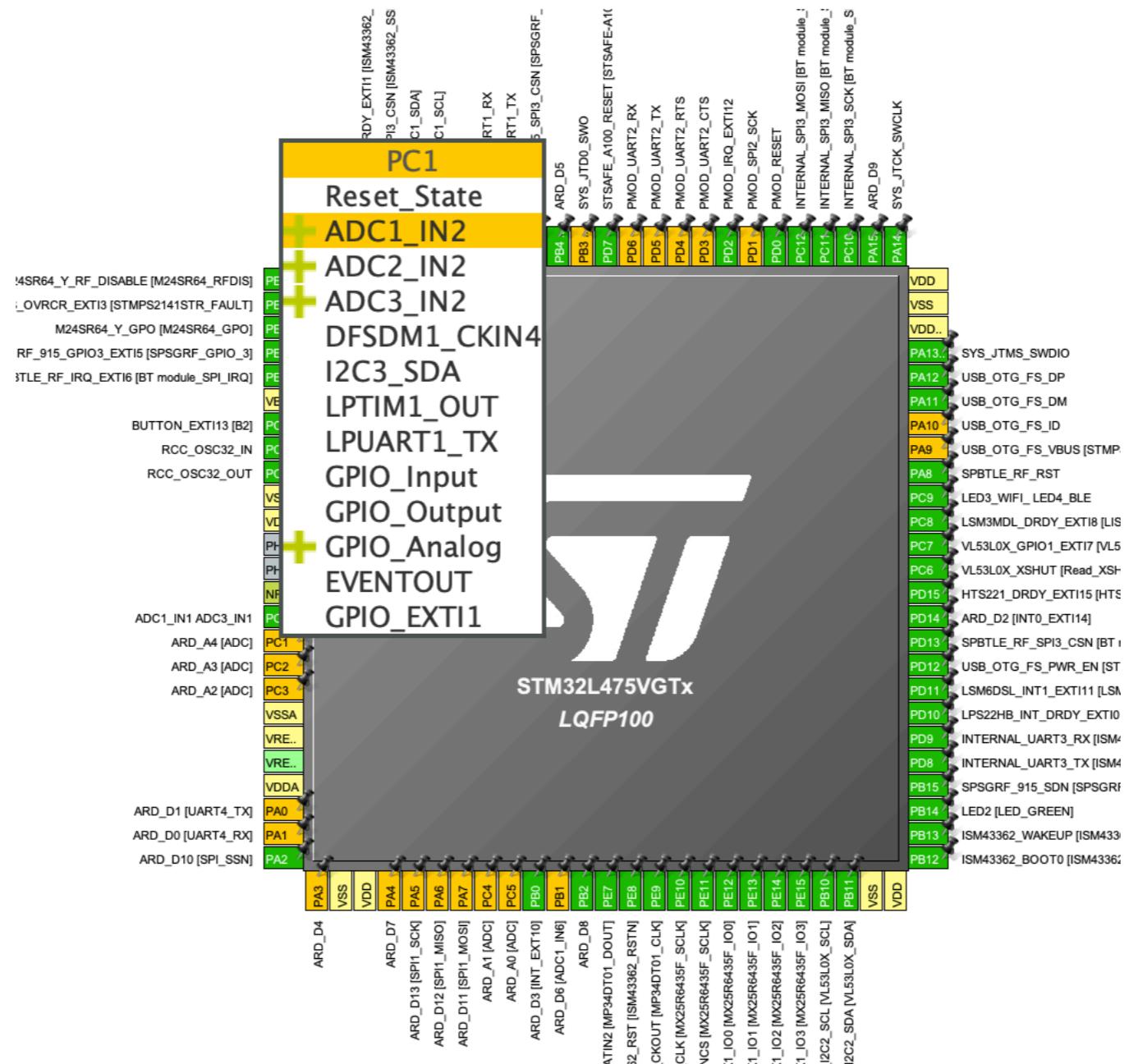
# Step: Pin View

## PC2 - ADC1 IN3 - ARD A3

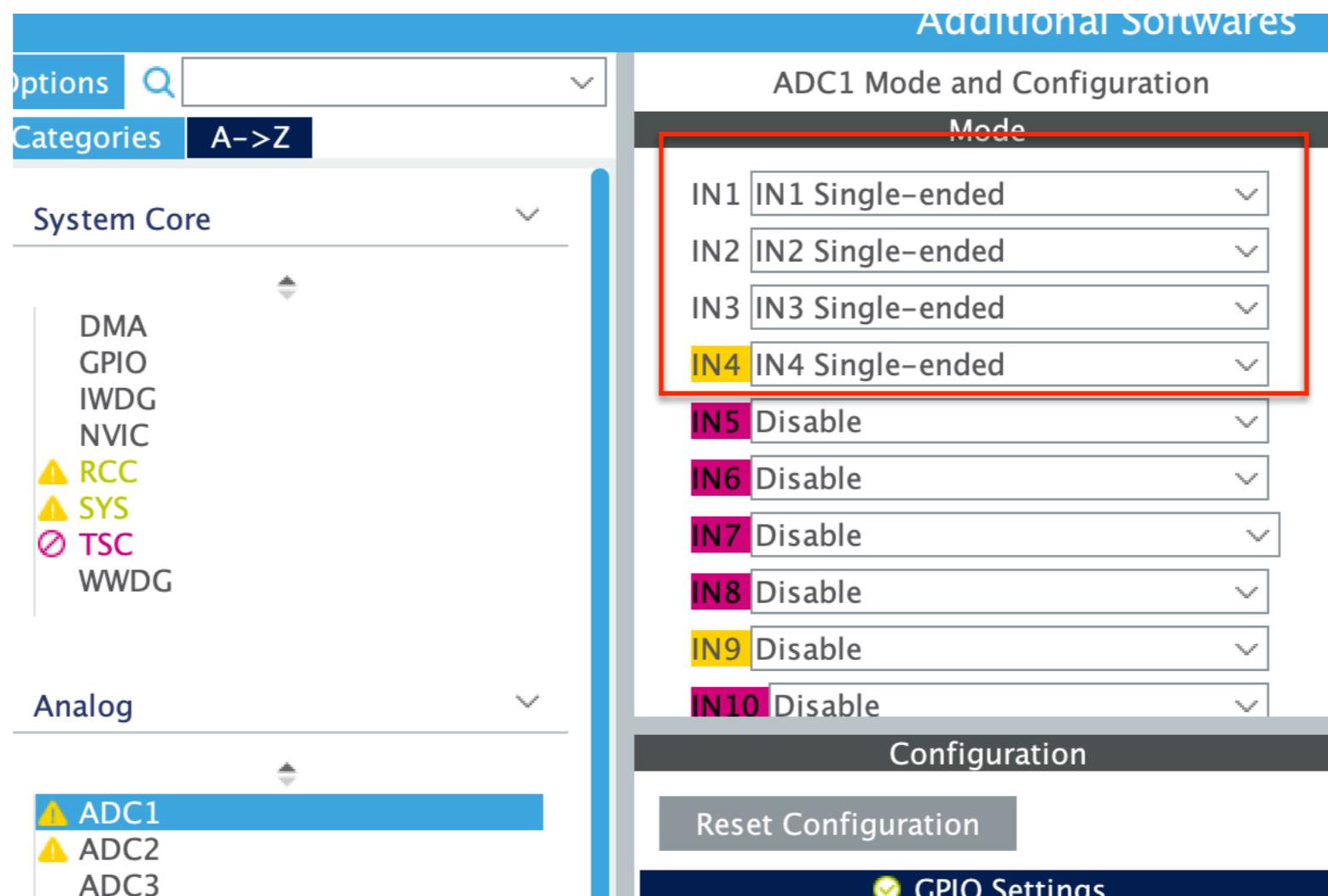


# Step: Pin View

## PC1 - ADC1 IN2 - ARD A4



# Step: Enable 4 channels for ADC1



# Step: System View, GPIO, Single Mapped Signals

Additional Softwares ▾ Pinout

GPIO Mode and Configuration

Configuration

Group By Peripherals

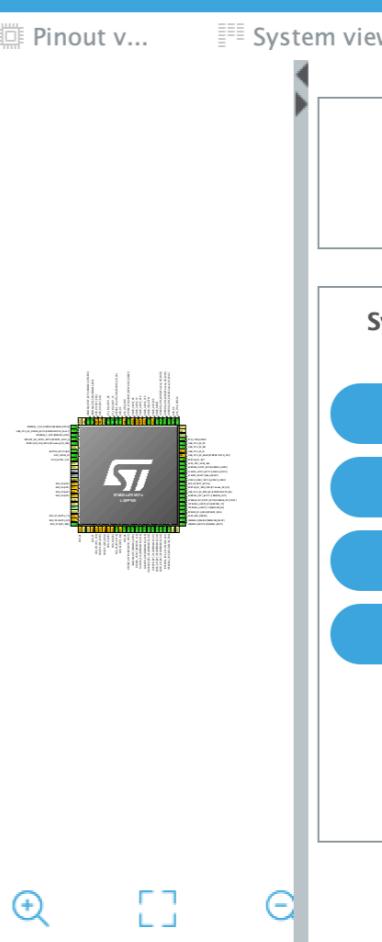
|      |                       |        |        |            |      |
|------|-----------------------|--------|--------|------------|------|
| SPI3 | SYS                   | USART1 | USART3 | USB_OTG_FS | NVIC |
| GPIO | Single Mapped Signals | DFSDM1 | I2C2   | QUADSPI    | RCC  |

Search Signals

Show only Modified Pins

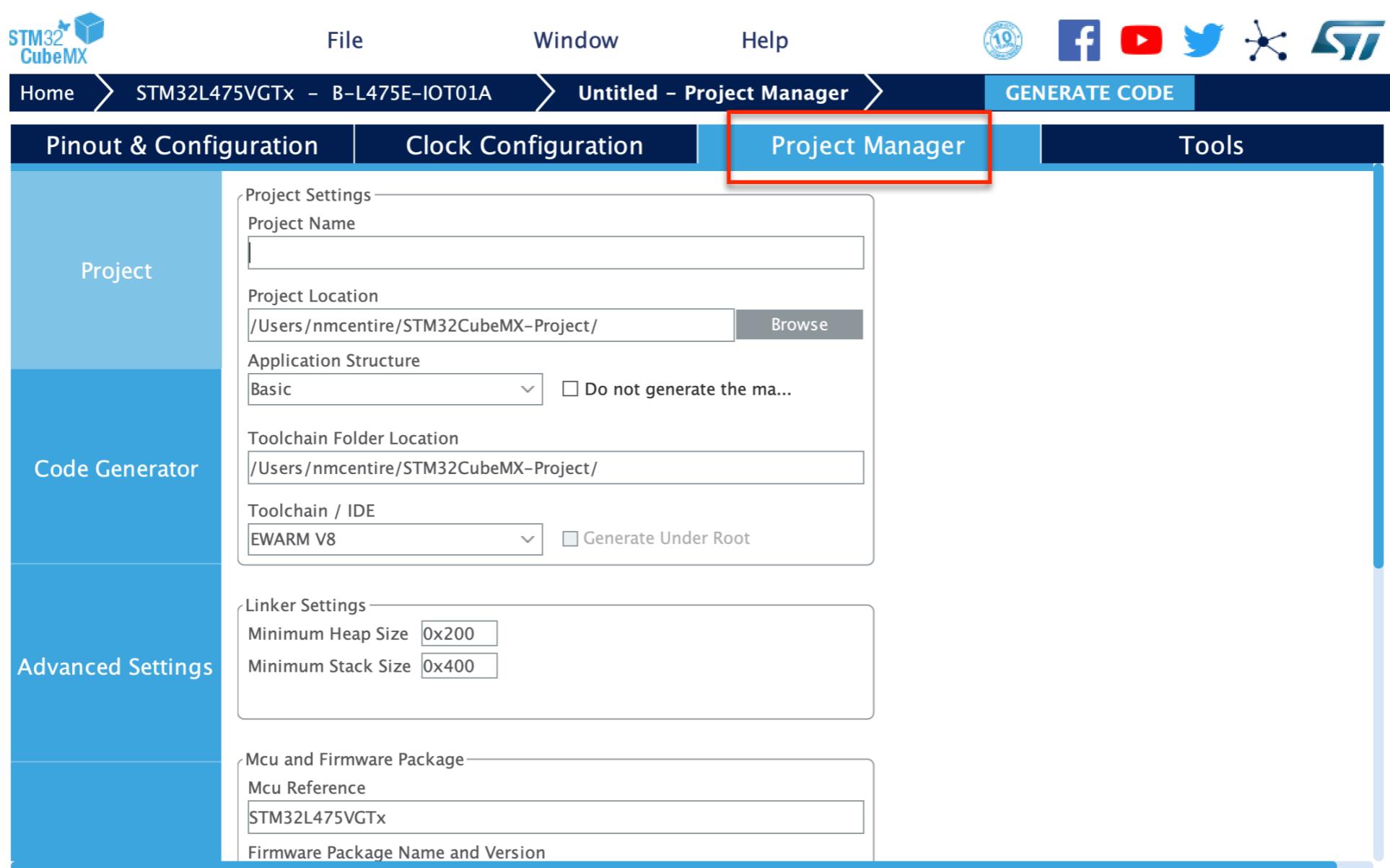
| Pin Name       | Signal on Pin | GPIO output... | GPIO mode       | GPIO Pull-u... | Maximum o... | Fast Mode | User Label     | Modified                            |
|----------------|---------------|----------------|-----------------|----------------|--------------|-----------|----------------|-------------------------------------|
| PA9            | USB_OTG_F...  | n/a            | Input mode      | No pull-up ... | n/a          | n/a       | USB_OTG_F...   | <input checked="" type="checkbox"/> |
| PA10           | USB_OTG_F...  | n/a            | Alternate Fu... | No pull-up ... | Very High    | n/a       | USB_OTG_F...   | <input checked="" type="checkbox"/> |
| PB1            | ADC1_IN16     | n/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_D6 [A...]  | <input checked="" type="checkbox"/> |
| PB3 (JTDO-...) | SYS_JTDO-S... | n/a            | n/a             | n/a            | n/a          | n/a       | SYS_JTDO_S...  | <input checked="" type="checkbox"/> |
| PB8            | I2C1_SCL      | n/a            | Alternate Fu... | Pull-up        | Very High    | Disable   | ARD_D15 [I...] | <input checked="" type="checkbox"/> |
| PB9            | I2C1_SDA      | n/a            | Alternate Fu... | Pull-up        | Very High    | Disable   | ARD_D14 [I...] | <input checked="" type="checkbox"/> |
| PC0            | ADC1_IN1      | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A5 [A...]  | <input checked="" type="checkbox"/> |
| PC1            | ADC1_IN2      | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A4 [A...]  | <input checked="" type="checkbox"/> |
| PC2            | ADC1_IN3      | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A3 [A...]  | <input checked="" type="checkbox"/> |
| PC3            | ADC1_IN4      | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A2 [A...]  | <input checked="" type="checkbox"/> |
| PC4            | ADC1_IN13     | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A1 [A...]  | <input checked="" type="checkbox"/> |
| PC5            | ADC1_IN14     | r/a            | Analog mod...   | No pull-up ... | n/a          | n/a       | ARD_A0 [A...]  | <input checked="" type="checkbox"/> |
| PD1            | SPI2_SCK      | n/a            | Alternate Fu... | No pull-up ... | Very High    | n/a       | PMOD_SPI2...   | <input checked="" type="checkbox"/> |
| PD3            | USART2_CTS    | n/a            | Alternate Fu... | No pull-up ... | Very High    | n/a       | PMOD_UAR...    | <input checked="" type="checkbox"/> |

Select Pins from table to configure them. Multiple selection is Allowed.

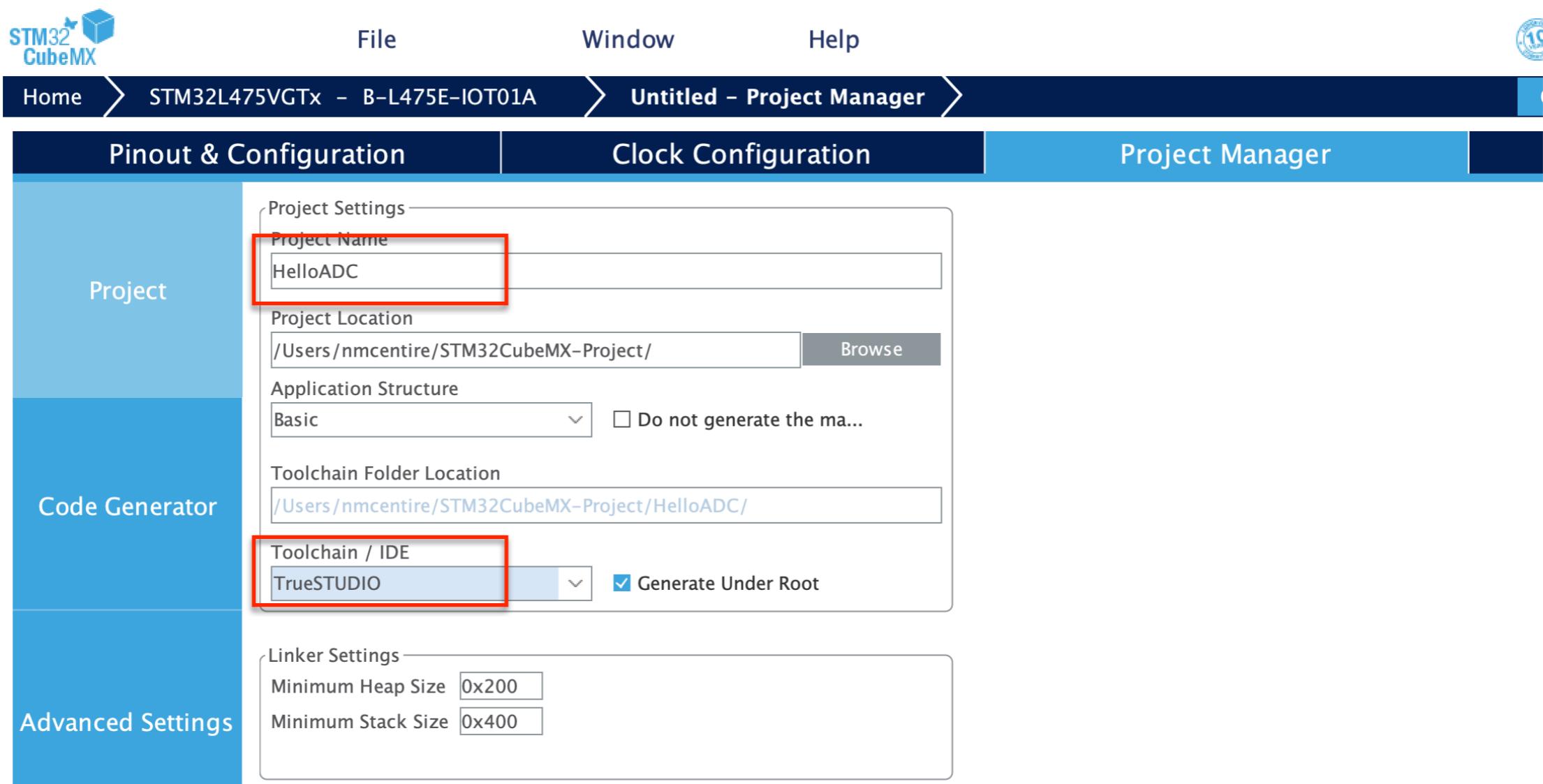


The screenshot shows the STM32F4 Discovery Board's pinout configuration. The main window displays a table of pins with columns for Pin Name, Signal on Pin, GPIO output..., GPIO mode, GPIO Pull-u..., Maximum o..., Fast Mode, User Label, and Modified. Several pins are highlighted with red boxes: PC0, PC1, PC2, PC3, PC4, PC5, PD1, and PD3. These correspond to ADC1\_IN1 through ADC1\_IN14, SPI2\_SCK, USART2\_CTS, and PMOD\_SPI2 respectively. The 'Modified' column for these pins also contains checked checkboxes. A legend at the bottom left indicates that a yellow question mark icon means 'Select Pins from table to configure them. Multiple selection is allowed.' The top navigation bar includes tabs for 'Additional Softwares' and 'Pinout', and icons for 'Pinout v...' and 'System view'.

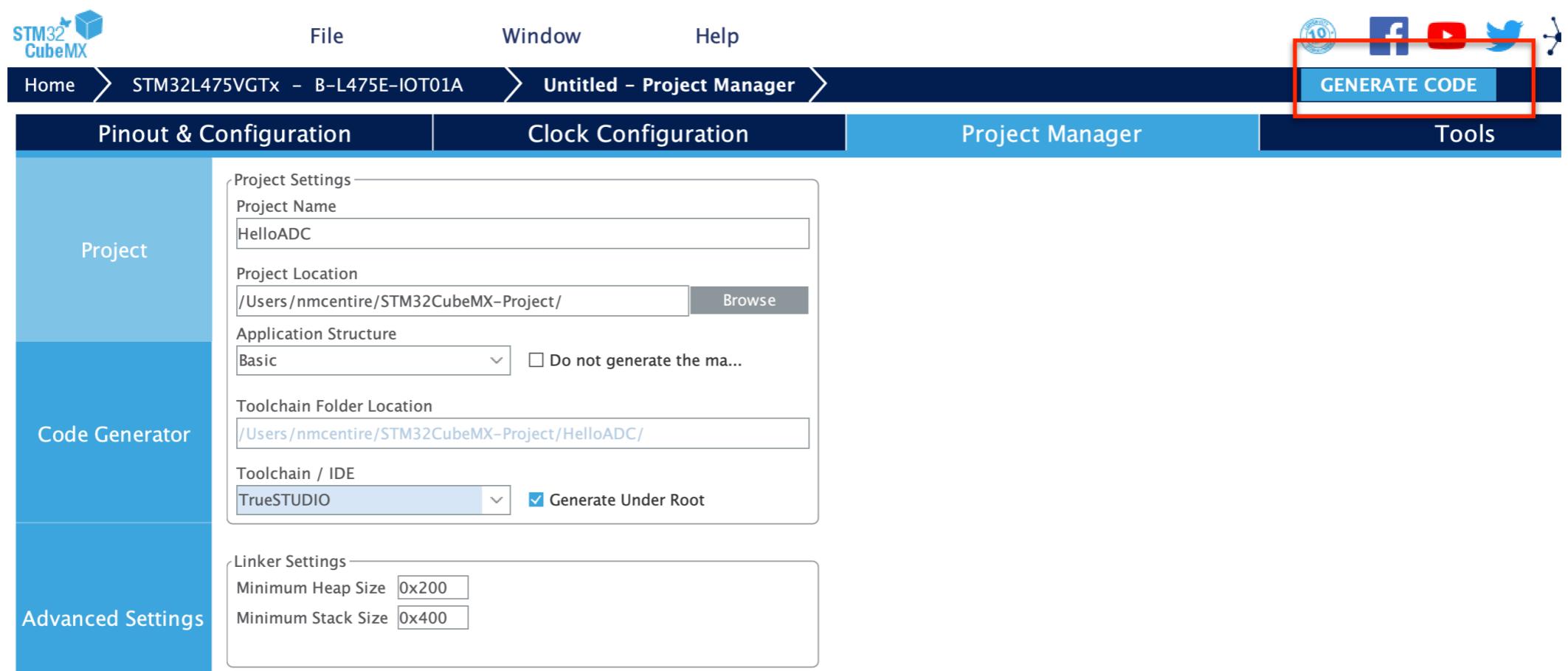
# Step: Observe “Project Manager”



# Step: Enter “HelloADC” for Project Name, TrueStudio for IDE



# Step: Click on “Generate Code”

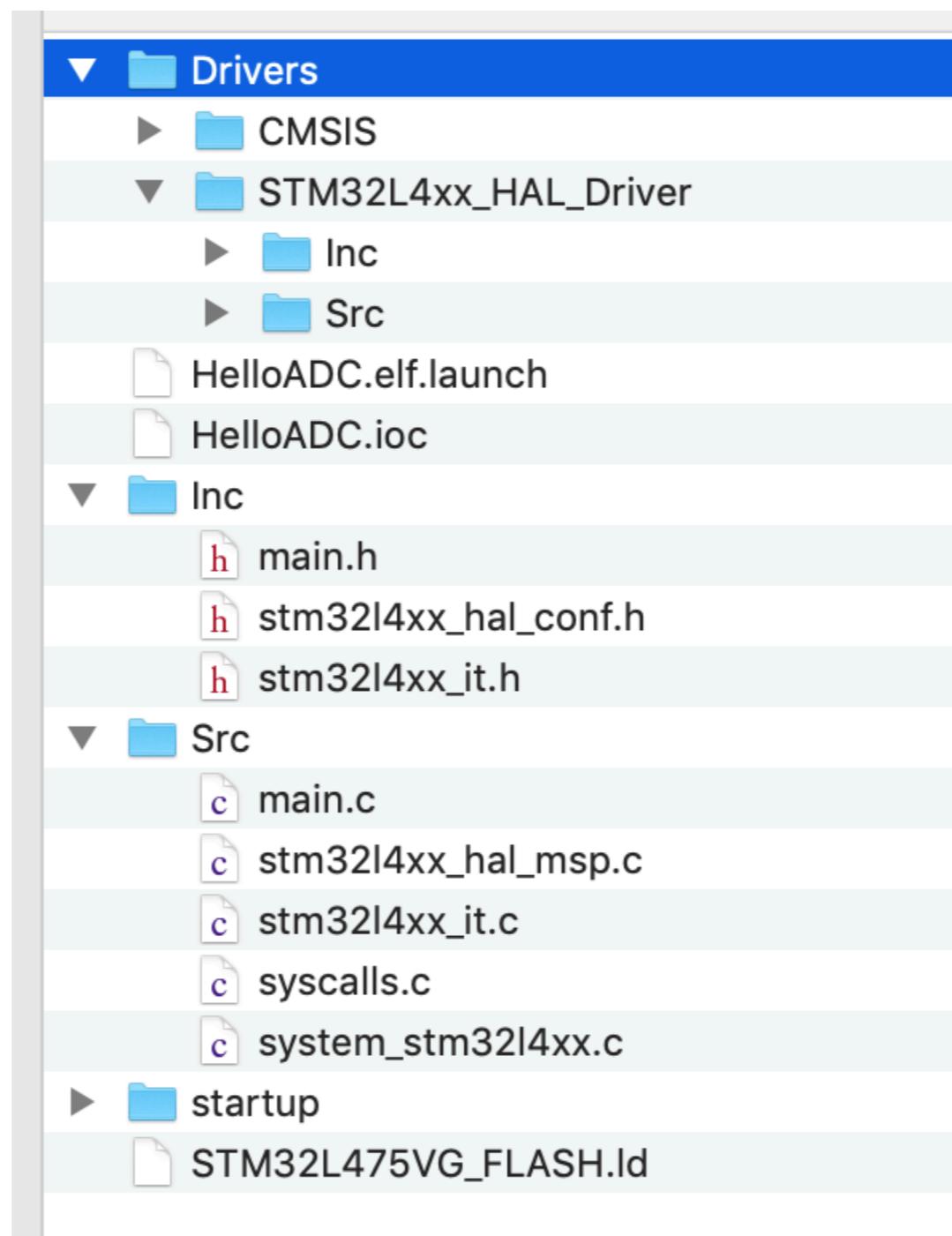


# Step: Select “Open Folder”

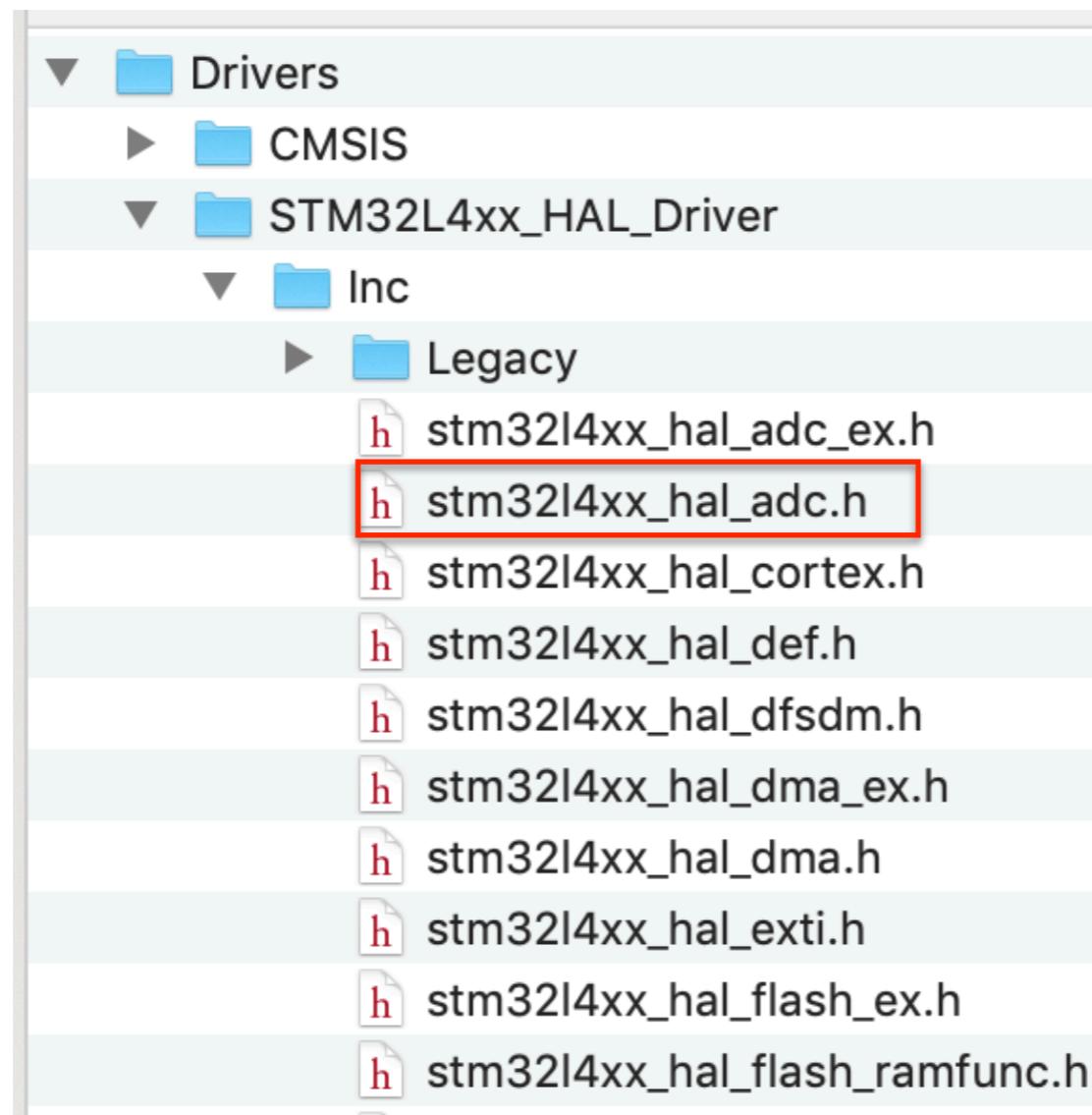


# Tour of Generated Project

# Step: View Open Folder

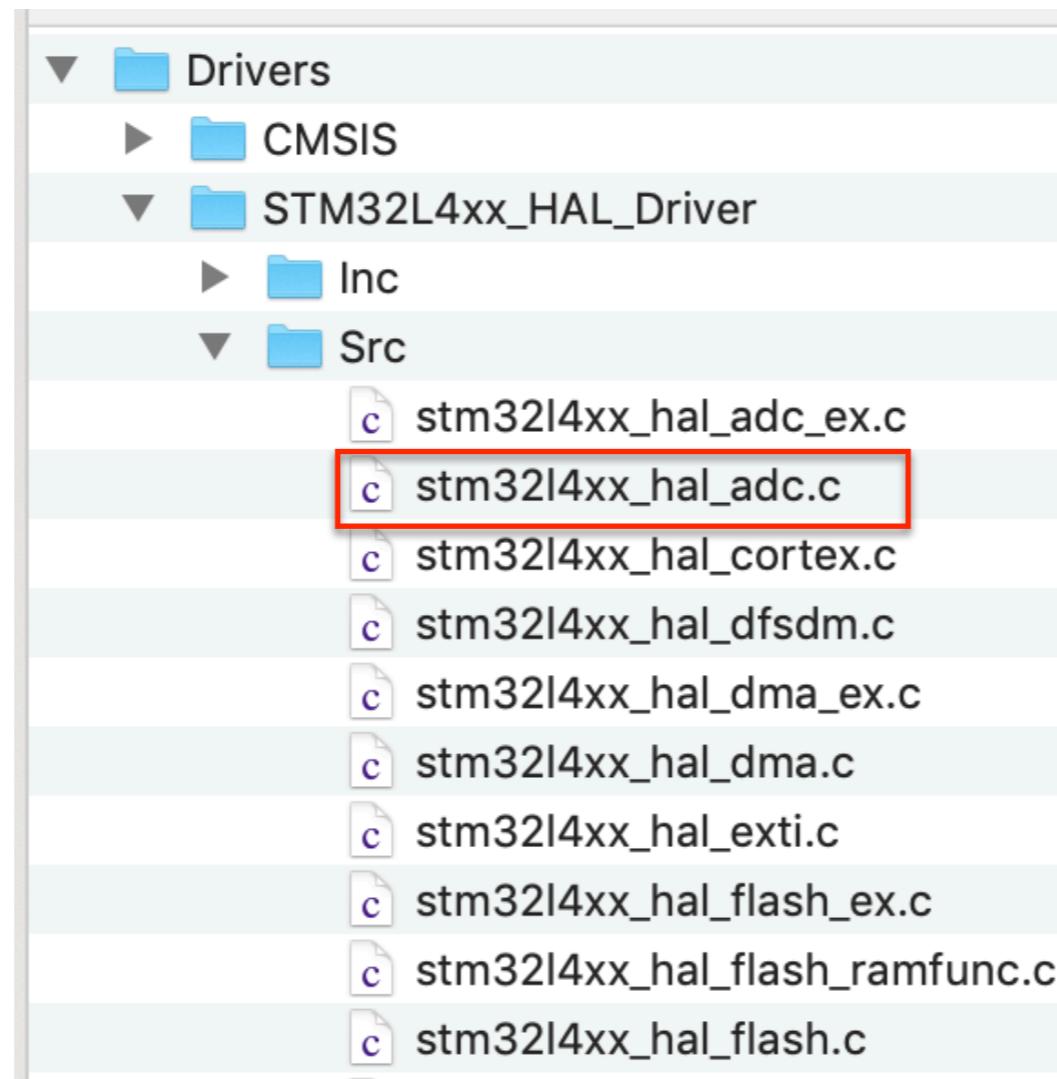


# Step: View “Drivers/STM32L4xx\_HAL\_Driver/Inc” directory



Observe  
stm32l4xx\_hal\_adc.h

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



Observe  
stm32l4xx\_hal\_adc.c

# main.c - Part 1

```
--  
44  /* Private variables -----  
45  ADC_HandleTypeDef hadc1;  
46  
  
--  -----  
74  static void MX_ADC1_Init(void);  
  
88  int main(void)  
89  {  
90  |  /* USER CODE BEGIN 1 */  
--
```

# main.c - Part 2

```
216 static void MX_ADC1_Init(void)
217 {
218     /* USER CODE BEGIN ADC1_Init_0 */
219
220
221     /* USER CODE END ADC1_Init_0 */
222
223     ADC_MultiModeTypeDef multimode = {0};
224     ADC_ChannelConfTypeDef sConfig = {0};
225 }
```

# main.c - Part 3

```
---  
229  /** Common config  
230  */  
231  hadc1.Instance = ADC1;  
232  hadc1.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;  
233  hadc1.Init.Resolution = ADC_RESOLUTION_12B;  
234  hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;  
235  hadc1.Init.ScanConvMode = ADC_SCAN_DISABLE;  
236  hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;  
237  hadc1.Init.LowPowerAutoWait = DISABLE;  
238  hadc1.Init.ContinuousConvMode = DISABLE;  
239  hadc1.Init.NbrOfConversion = 1;  
240  hadc1.Init.DiscontinuousConvMode = DISABLE;  
241  hadc1.Init.ExternalTrigConv = ADC_SOFTWARE_START;  
242  hadc1.Init.ExternalTrigConvEdge = ADC_EXTERNALTRIGCONVEDGE_NONE;  
243  hadc1.Init.DMAContinuousRequests = DISABLE;  
244  hadc1.Init.Overrun = ADC_OVR_DATA_PRESERVED;  
245  hadc1.Init.OversamplingMode = DISABLE;  
246  if (HAL_ADC_Init(&hadc1) != HAL_OK)  
247  {  
248    | Error Handler():
```

# main.c - Part 4

```
250  /** Configure the ADC multi-mode
251  */
252  multimode.Mode = ADC_MODE_INDEPENDENT;
253  if (HAL_ADCEx_MultiModeConfigChannel(&hadc1, &multimode) != HAL_OK)
254  {
255  | Error_Handler();
256 }
```

# main.c - Part 5

```
---  
257     /** Configure Regular Channel  
258     */  
259     sConfig.Channel = ADC_CHANNEL_1;  
260     sConfig.Rank = ADC_REGULAR_RANK_1;  
261     sConfig.SamplingTime = ADC_SAMPLETIME_2CYCLES_5;  
262     sConfig.SingleDiff = ADC_SINGLE_ENDED;  
263     sConfig.OffsetNumber = ADC_OFFSET_NONE;  
264     sConfig.Offset = 0;  
265     if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)  
266     {  
267         Error_Handler();  
268     }
```

# main.c - Part 6

# Summary so far...

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

Use TrueStudio  
To Read the value of  
Arduino A0 (analog read)

# Step: Edit main.c - Part 1

```
static void MX_ADC1_Init(void)
{
    s
    /** Configure Regular Channel
     */
    sConfig.Channel = ADC_CHANNEL_14;
    sConfig.Rank = ADC_REGULAR_RANK_1;
    sConfig.SamplingTime = ADC_SAMPLETIME_2CYCLES_5;
    sConfig.SingleDiff = ADC_SINGLE_ENDED;
    sConfig.OffsetNumber = ADC_OFFSET_NONE;
    sConfig.Offset = 0;
    if (HAL_ADC_ConfigChannel(&hadc1, &sConfig) != HAL_OK)
    {
        Error_Handler();
    }
}
```

ARD\_A0 => PC5 => ADCX1\_IN14

# Step: Edit main.c - Part 2

```
87 // Main program entry point
88 int main(void)
89 {
90
91     uint32_t adcResult1;
92     HAL_StatusTypeDef halResult1;
93
94     int main(void)
95     {
96
97         HAL_ADC_Start(&hadc1);
98         halResult1 = HAL_ADC_PollForConversion(&hadc1, 100);
99         adcResult1 = HAL_ADC_GetValue(&hadc1);
100
101         HAL_Delay(200);
102
103     }
104 }
```

# Summary

- Introduction to ADC and Embedded C
- STM32CubeMX and ADC Code Generation
- Tour of ADC
- TrueStudio and ADC
  - Read ADC value