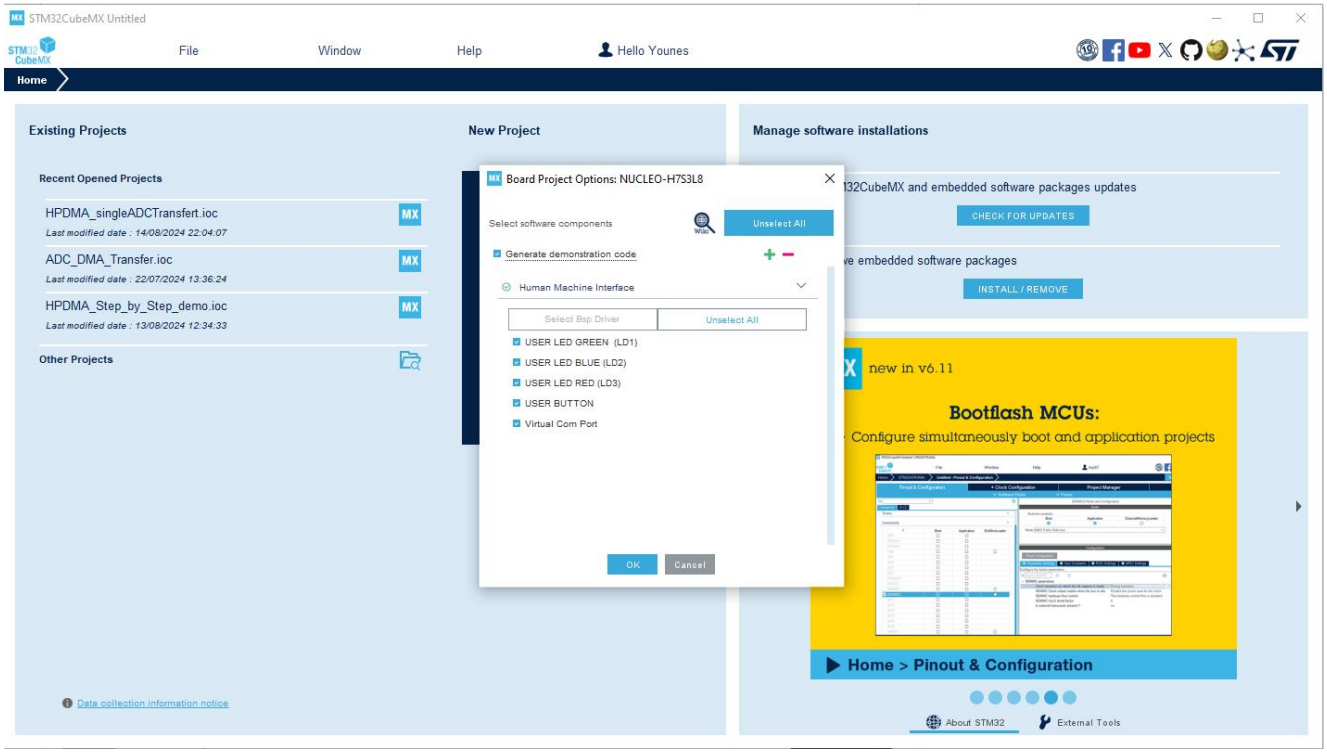
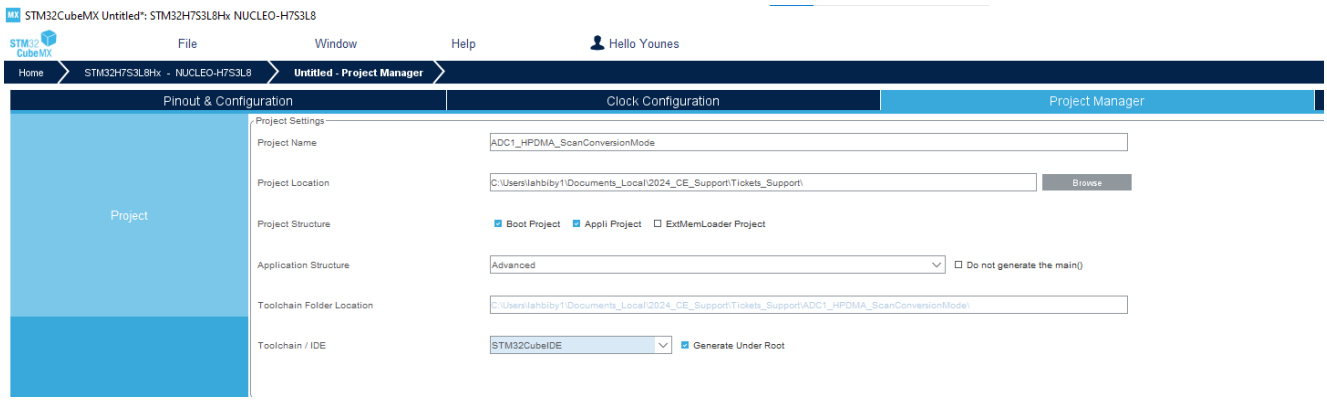
How to configure an ADC DMA Circular mode using STM32CubeMX

STM32CubeMX version 6.12
STM32CubeIDE version: 1.15.1

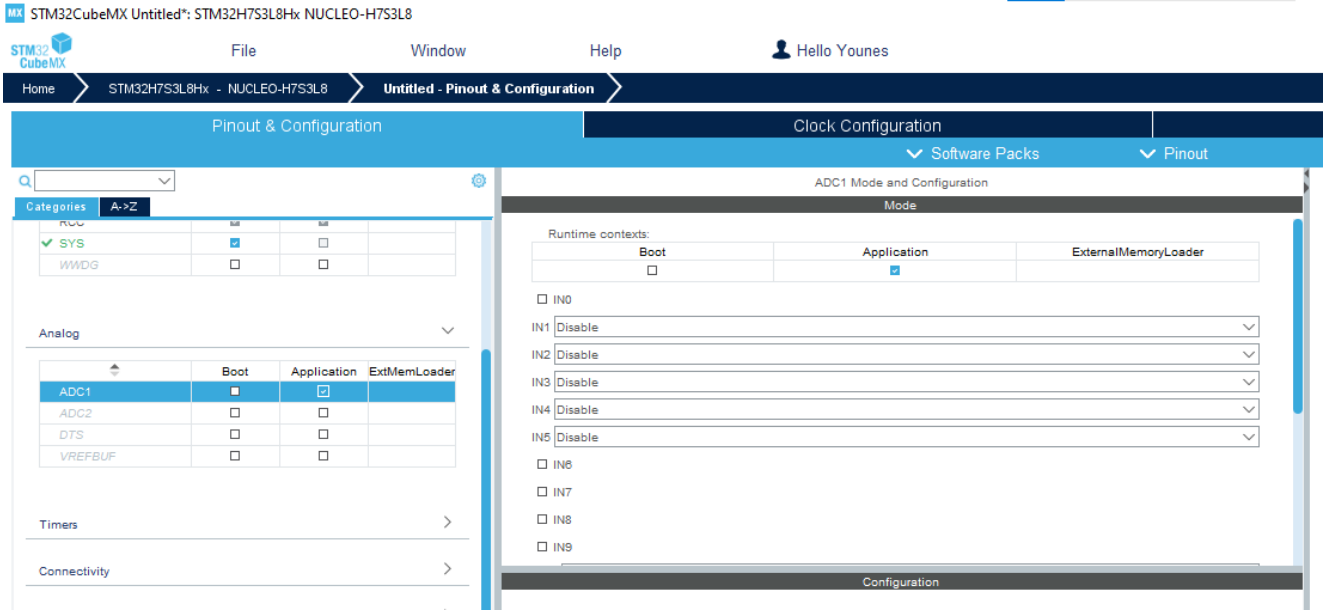
Step1: Access to Board Selector (optionally use the BSP to configure leds, button and VCP)



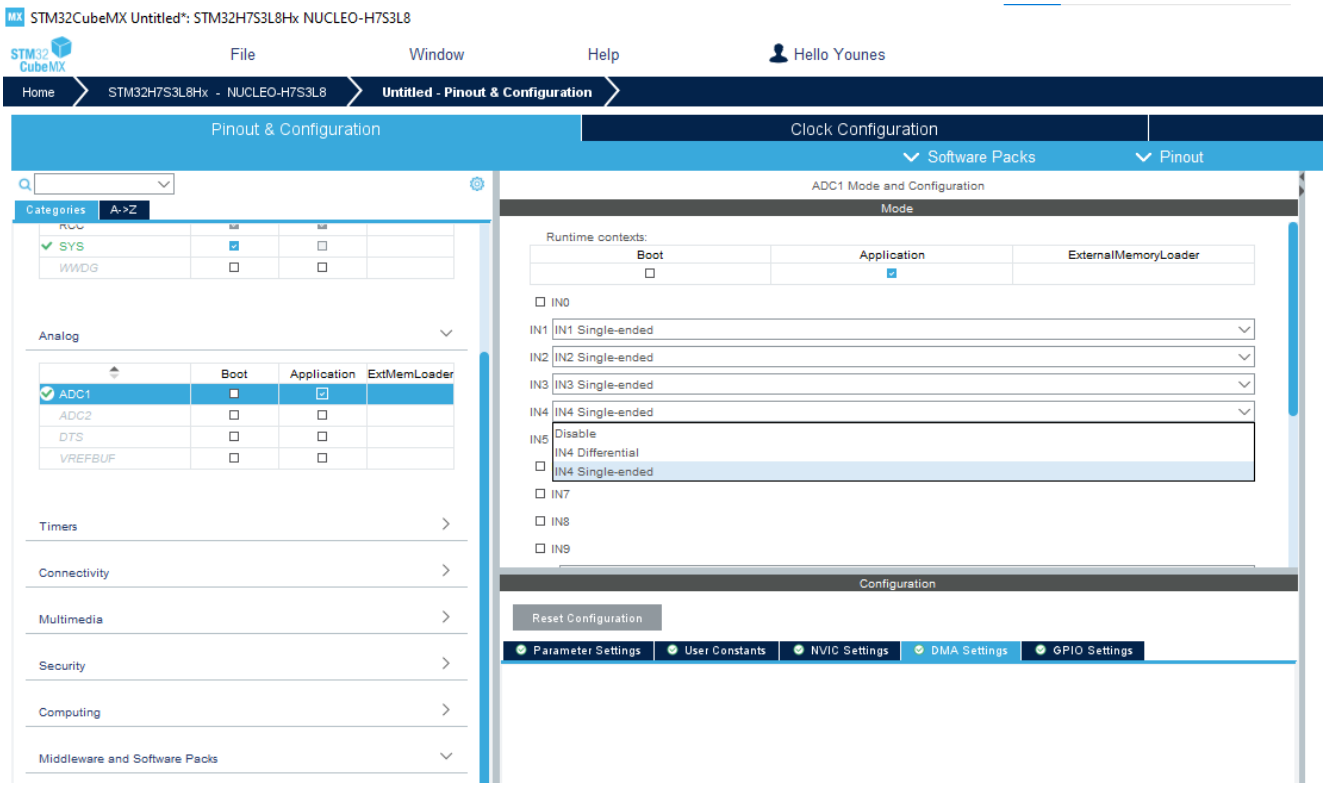
Step2: choose Appli and boot projects generation as we will put the application code into an external memory.



Step3: select ADC1 and assign it to Appli



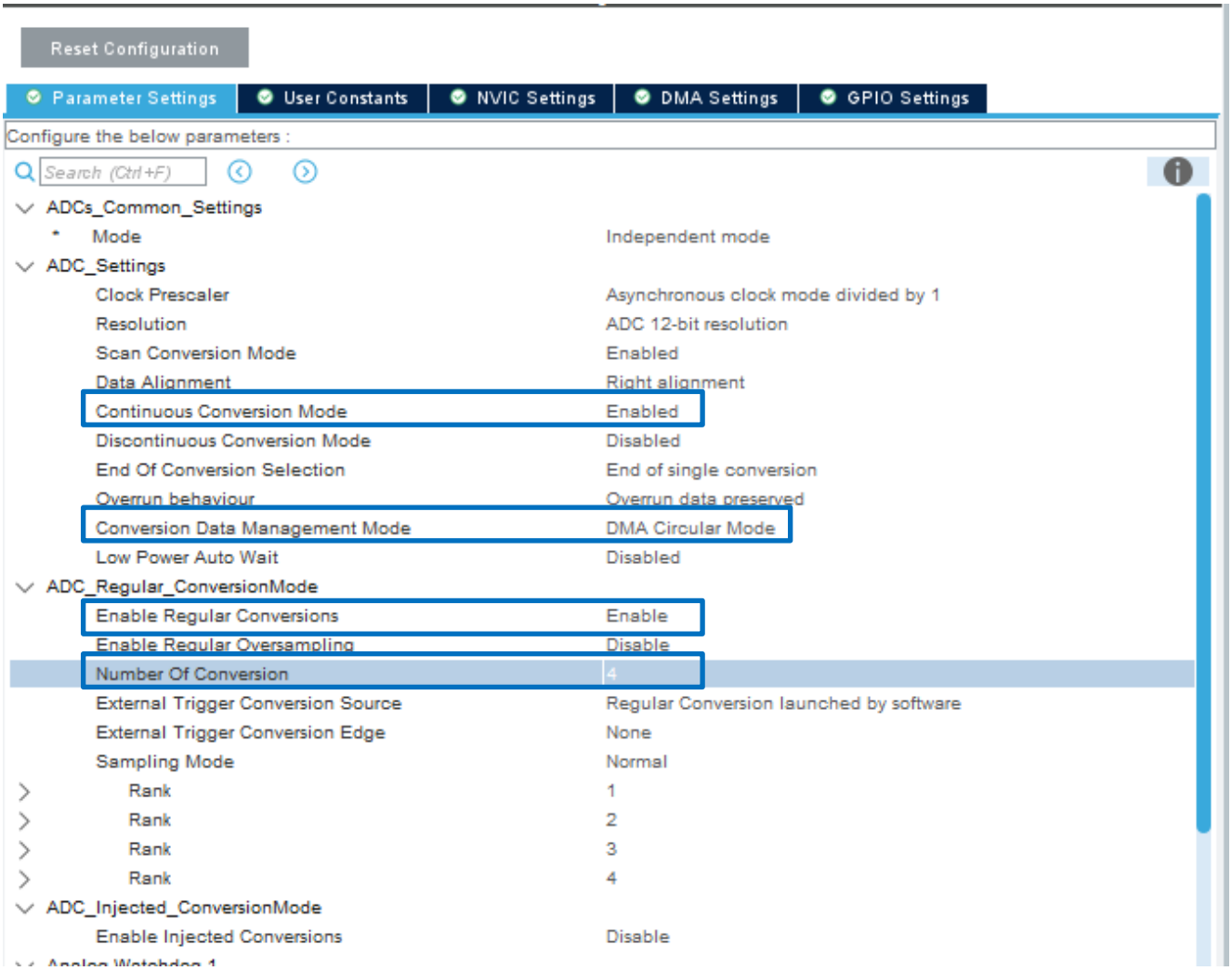
Step4: Enable the ADC channels you need (here 4 channels are enabled)



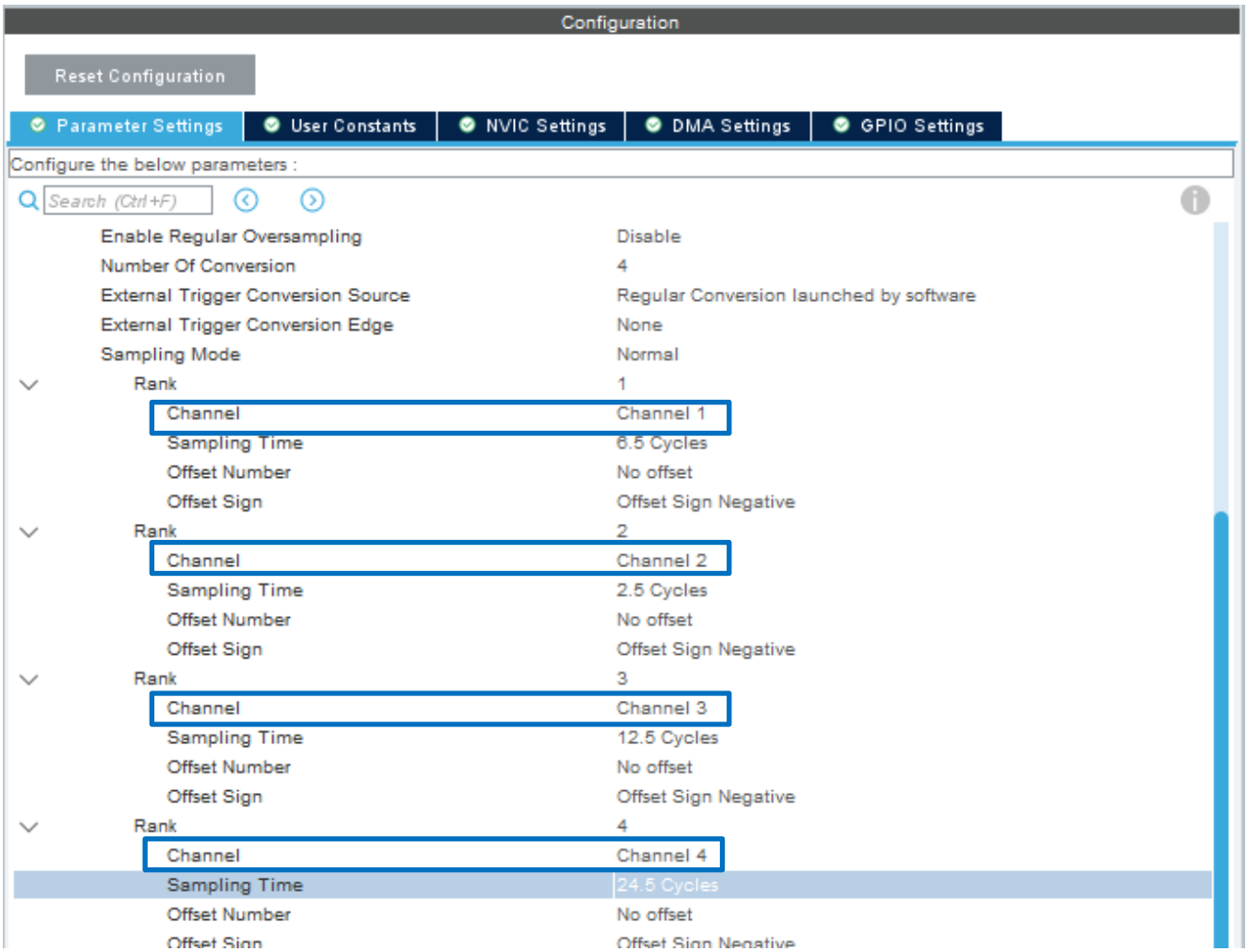
Step5:

- Set ***`Continuous conversion mode`*** to **Enable:** This optiion run ADC in loops. When ADC finishes converting all its channesl it will start again from beginning.
- Set ***`Enable Regular Conversions`*** to **Enable**

- Configure ADC 1: Set ***Conversion Data Management Mode*** to **DMA Circular Mode.** After ADC converts value, it will create request for DMA. Circular mode here means that after ADC finishes all regular channels it will continue generate DMA request in next run too.
- Set ***number of conversion*** to **4 (or your preferred value):** This will set ADC to do 4 ADC conversion which we can set.

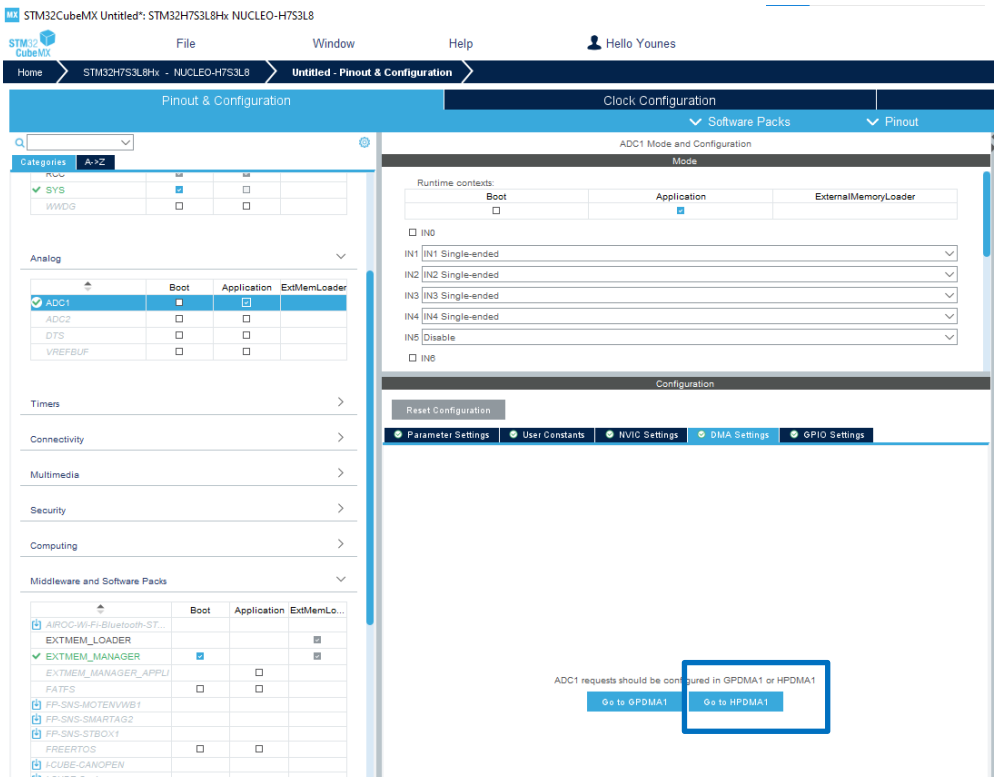


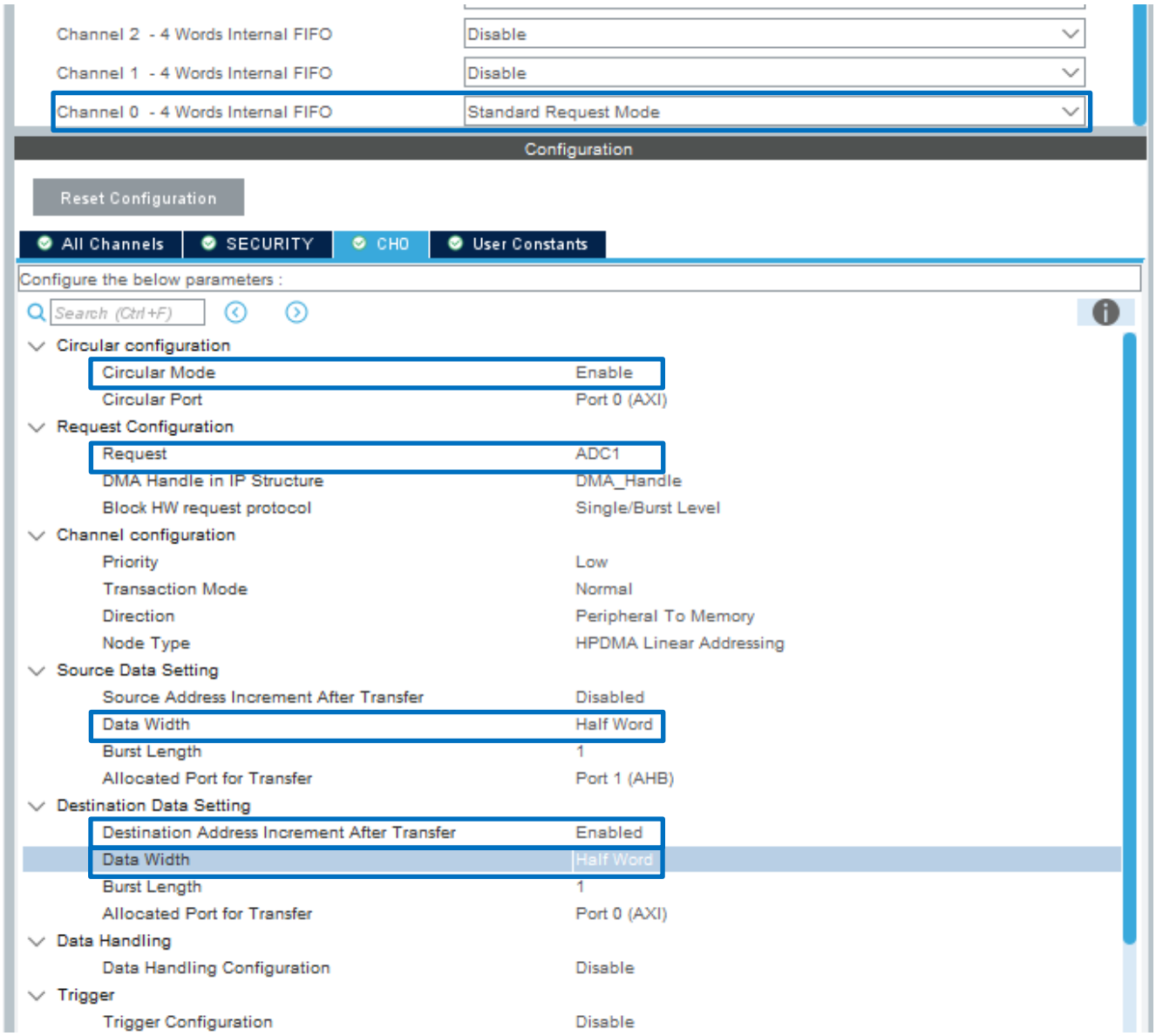
Step6: choose the sequence. You can set ADC channel for each ***Rank***. Each rank will be assigned one ADC channel to convert. It is possible to select same channel each time.



Step7: go to HPDMA settings.

1. Select CH0
2. Set ***Circular Mode*** to **Enable**
3. Set ***Request*** to **ADC1**
4. Set ***Source Data Settings***
    1. ***Data Width*** to **Half Word**
5. Set ***Destination Data Settings***
    1. ***Data Width*** to **Half Word**
6. Set ***Destination Data Settings***
    1. ***Destination Address Increment After Transfer*** to **Enable**


STM32CubeMX Untitled*: STM32H7S3L8Hx NUCLEO-H7S3L8
File
Window
Help
Hello Younes
Home
STM32H7S3L8Hx - NUCLEO-H7S3L8
Untitled - Pinout & Configuration
Pinout & Configuration
Clock Configuration
Software Packs
Pinout
Categories
A->Z
SYS
WWDG
Analog
Boot
Application
ExtMemLoader
ADC1
ADC2
DTS
VREFBUF
Timers
Connectivity
Multimedia
Security
Computing
Middleware and Software Packs
AIROC-Wi-Fi-Bluetooth-ST...
EXTMEM_LOADER
EXTMEM_MANAGER
EXTMEM_MANAGER_APPLI
FATFS
FP-SNS-MOTENVWB1
FP-SNS-SMARTAG2
FP-SNS-STBOX1
FREERTOS
I-CUBE-CANOPEN
ADC1 Mode and Configuration
Mode
Runtime contexts:
Boot
Application
ExternalMemoryLoader
IN0
IN1 IN1 Single-ended
IN2 IN2 Single-ended
IN3 IN3 Single-ended
IN4 IN4 Single-ended
IN5 Disable
IN6
Configuration
Reset Configuration
Parameter Settings
User Constants
NVIC Settings
DMA Settings
GPIO Settings
ADC1 requests should be configured in GPDMA1 or HPDMA1
Go to GPDMA1
Go to HPDMA1



Step8: generate the code
Create a table of size 64 (or multiple of 8 for example)
As the processor is Cortex-M7, place the table into non-cacheable region (ITCM/DTCM memories (@0x0000000 / @0x20000000: not cacheable and only accessible by the Cortex M7 and the GPDMA/HPDMA).
If the application needs to put DMA buffers in AXI SRAM (starting from @0x24000000), the user must:

- either define a non-cacheable region in the MPU and linker configuration file to locate DMA buffers **(this is our choice in this example).**
- or to ensure cache maintenance operations to ensure the cache coherence between the CPU and the DMAs.
- or disable the Dcache (not recommended)

```
/* USER CODE BEGIN PD */
#define ADC_CONVERTED_DATA_BUFFER_SIZE 64
/* USER CODE END PD */

/* USER CODE BEGIN PV */
__IO uint16_t uhADCxConvertedData[ADC_CONVERTED_DATA_BUFFER_SIZE]
__attribute__((section("noncacheable_buffer")));
/* USER CODE END PV */

/* USER CODE BEGIN PFP */
static void MPU_AdjustRegionAddressSize(uint32_t Address, uint32_t Size,
MPU_Region_InitTypeDef* pInit);
static void MPU_Config(void);
/* USER CODE END PFP */
```

int main(void)

{

```
  MPU_Config(void);

  ….

   HAL_ADC_Start_DMA(&hadc1,
                        (uint32_t *)uhADCxConvertedData,
                        ADC_CONVERTED_DATA_BUFFER_SIZE
                       )


  …
```

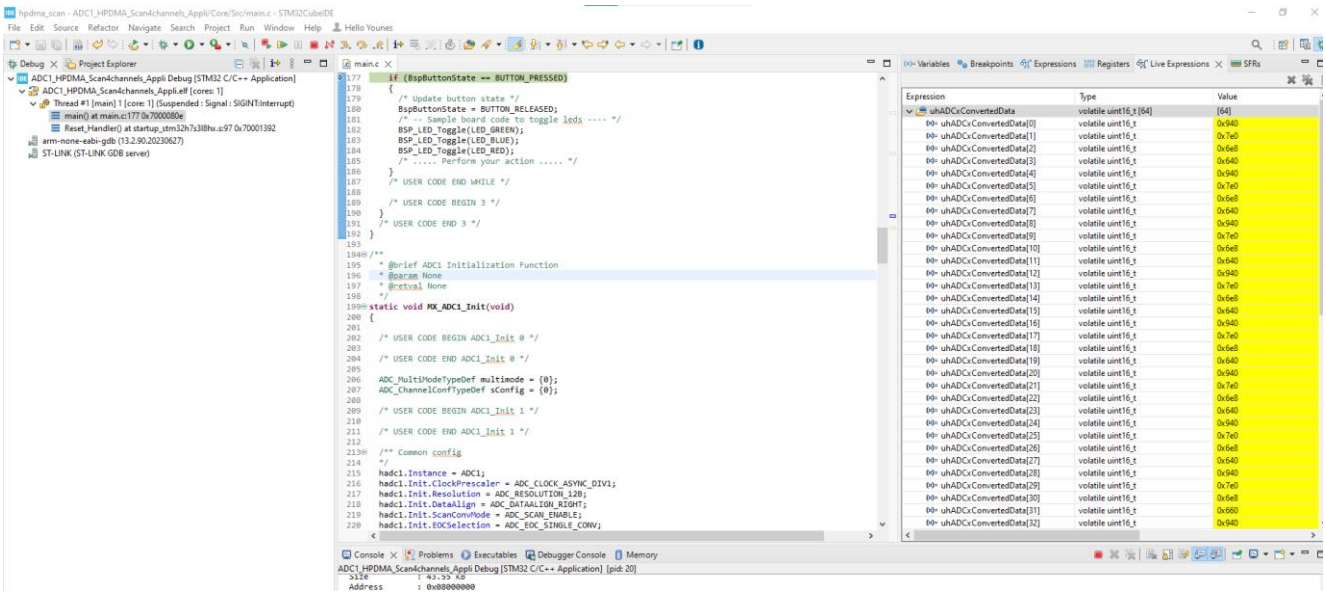
…

}

….

See complete code

Results are always updated, suspend debug to analyze the values.


hpdma_scan - ADC1_HPDMA_Scan4channels_Appli/Core/Src/main.c - STM32CubeIDE
File Edit Source Refactor Navigate Search Project Run Window Help Hello Younes
Debug Project Explorer
ADC1_HPDMA_Scan4channels_Appli Debug [STM32 C/C++ Application]
ADC1_HPDMA_Scan4channels_Appli.elf [cores: 1]
Thread #1 [main] 1 [core: 1] (Suspended : Signal : SIGINT:Interrupt)
main() at main.c:177 0x700008de
Reset_Handler() at startup_stm32h7s3lhx.s:97 0x70001392
arm-none-eabi-gdb (13.2.90.20230627)
ST-LINK (ST-LINK GDB server)
main.c
if (BspButtonState == BUTTON_PRESSED)
{
/* Update button state */
BspButtonState = BUTTON_RELEASED;
/* -- Sample board code to toggle leds ---- */
BSP_LED_Toggle(LED_GREEN);
BSP_LED_Toggle(LED_BLUE);
BSP_LED_Toggle(LED_RED);
/* ..... Perform your action ..... */
}
/* USER CODE END WHILE */
/* USER CODE BEGIN 3 */
}
/* USER CODE END 3 */
}
/**
* @brief ADC1 Initialization Function
* @param None
* @retval None
*/
static void MX_ADC1_Init(void)
{
/* USER CODE BEGIN ADC1_Init 0 */
/* USER CODE END ADC1_Init 0 */
ADC_MultiModeTypeDef multimode = {0};
ADC_ChannelConfTypeDef sConfig = {0};
/* USER CODE BEGIN ADC1_Init 1 */
/* USER CODE END ADC1_Init 1 */
/** Common config
*/
hadc1.Instance = ADC1;
hadc1.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;
hadc1.Init.Resolution = ADC_RESOLUTION_12B;
hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
hadc1.Init.ScanConvMode = ADC_SCAN_ENABLE;
hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
Variables Breakpoints Expressions Registers Live Expressions SFRs
Expression Type Value
uhADCxConvertedData volatile uint16_t [64] [64]
uhADCxConvertedData[0] volatile uint16_t 0x940
uhADCxConvertedData[1] volatile uint16_t 0x7e0
uhADCxConvertedData[2] volatile uint16_t 0x6e8
uhADCxConvertedData[3] volatile uint16_t 0x640
uhADCxConvertedData[4] volatile uint16_t 0x940
uhADCxConvertedData[5] volatile uint16_t 0x7e0
uhADCxConvertedData[6] volatile uint16_t 0x6e8
uhADCxConvertedData[7] volatile uint16_t 0x640
uhADCxConvertedData[8] volatile uint16_t 0x940
uhADCxConvertedData[9] volatile uint16_t 0x7e0
uhADCxConvertedData[10] volatile uint16_t 0x6e8
uhADCxConvertedData[11] volatile uint16_t 0x640
uhADCxConvertedData[12] volatile uint16_t 0x940
uhADCxConvertedData[13] volatile uint16_t 0x7e0
uhADCxConvertedData[14] volatile uint16_t 0x6e8
uhADCxConvertedData[15] volatile uint16_t 0x640
uhADCxConvertedData[16] volatile uint16_t 0x940
uhADCxConvertedData[17] volatile uint16_t 0x7e0
uhADCxConvertedData[18] volatile uint16_t 0x6e8
uhADCxConvertedData[19] volatile uint16_t 0x640
uhADCxConvertedData[20] volatile uint16_t 0x940
uhADCxConvertedData[21] volatile uint16_t 0x7e0
uhADCxConvertedData[22] volatile uint16_t 0x6e8
uhADCxConvertedData[23] volatile uint16_t 0x640
uhADCxConvertedData[24] volatile uint16_t 0x940
uhADCxConvertedData[25] volatile uint16_t 0x7e0
uhADCxConvertedData[26] volatile uint16_t 0x6e8
uhADCxConvertedData[27] volatile uint16_t 0x640
uhADCxConvertedData[28] volatile uint16_t 0x940
uhADCxConvertedData[29] volatile uint16_t 0x7e0
uhADCxConvertedData[30] volatile uint16_t 0x6e8
uhADCxConvertedData[31] volatile uint16_t 0x660
uhADCxConvertedData[32] volatile uint16_t 0x940
Console Problems Executables Debugger Console Memory
ADC1_HPDMA_Scan4channels_Appli Debug [STM32 C/C++ Application] [pid: 20]
Address : 0x08000000