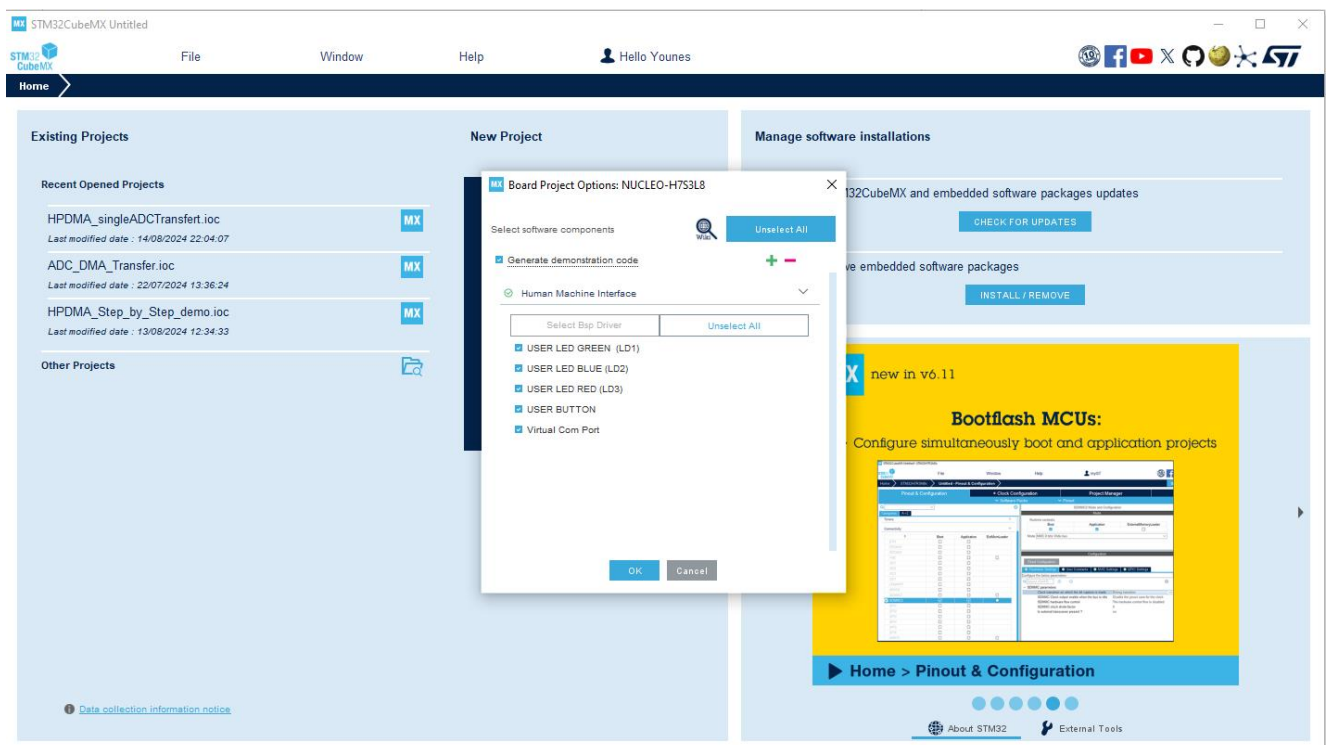


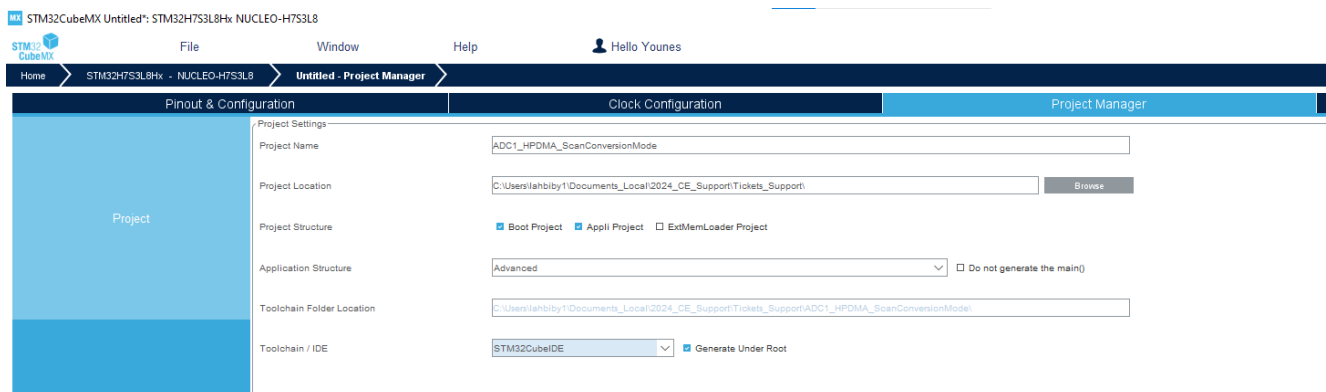
# How to configure an ADC DMA Circular mode using STM32CubeMX

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

STM32CubeMX Untitled\*: STM32H7S3L8Hx NUCLEO-H7S3L8

File Window Help Hello Younes

Home STM32H7S3L8Hx - NUCLEO-H7S3L8 Untitled - Pinout & Configuration

Pinout & Configuration Clock Configuration Software Packs Pinout

ADC1 Mode and Configuration

Mode

Runtime contexts:

Boot	Application	ExternalMemoryLoader
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

IN0

IN1 Disable

IN2 Disable

IN3 Disable

IN4 Disable

IN5 Disable

IN6

IN7

IN8

IN9

Configuration

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

STM32CubeMX Untitled\*: STM32H7S3L8Hx NUCLEO-H7S3L8

File Window Help Hello Younes

Home STM32H7S3L8Hx - NUCLEO-H7S3L8 Untitled - Pinout & Configuration

Pinout & Configuration Clock Configuration Software Packs Pinout

ADC1 Mode and Configuration

Mode

Runtime contexts:

Boot	Application	ExternalMemoryLoader
<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

IN0

IN1 IN1 Single-ended

IN2 IN2 Single-ended

IN3 IN3 Single-ended

IN4 IN4 Single-ended

IN5 Disable

IN4 Differential

IN4 Single-ended

IN7

IN8

IN9

Configuration

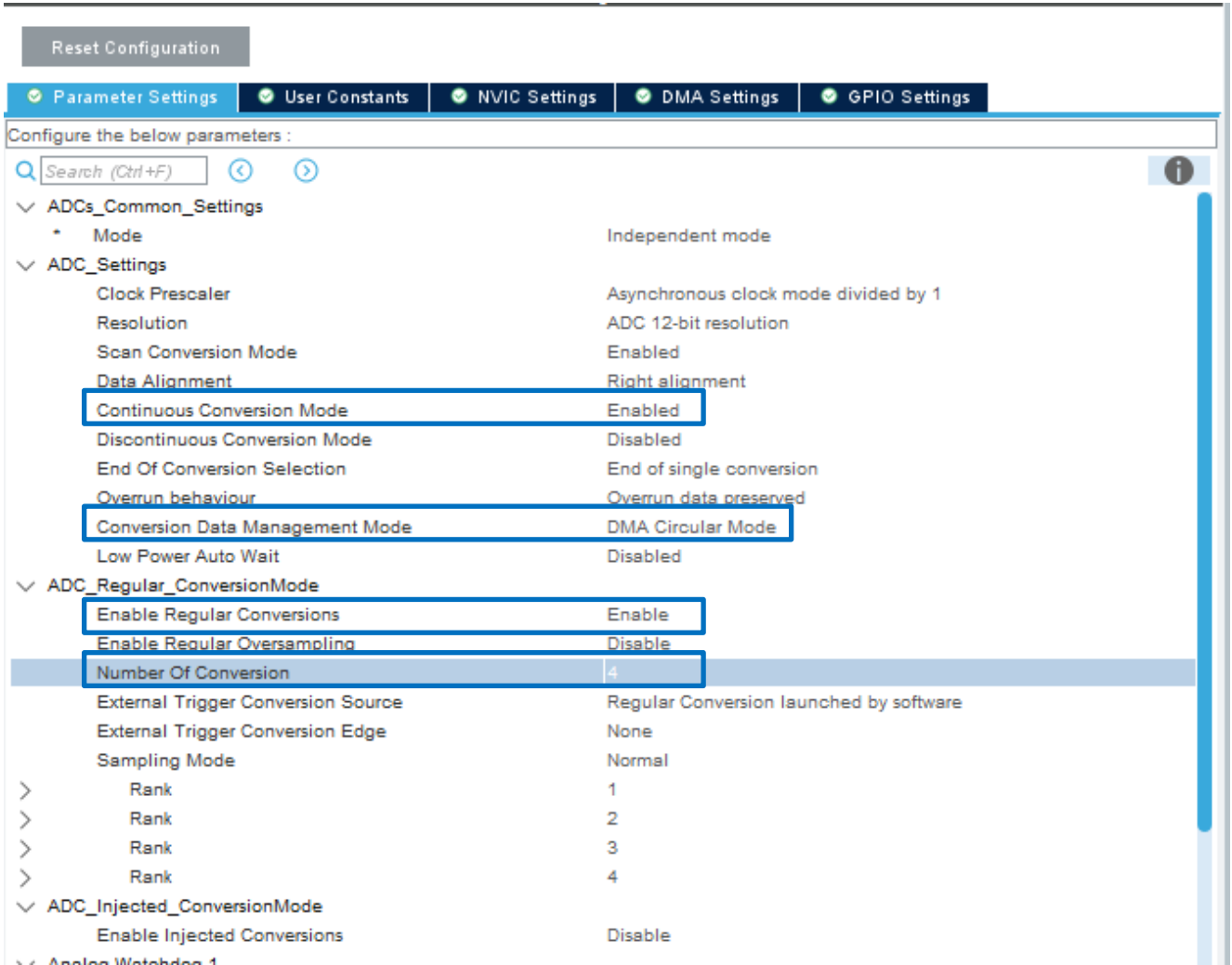
Reset Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

### Step5:

- Set *Continuous conversion mode* to **Enable**: This option run ADC in loops. When ADC finishes converting all its channels 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.



Reset Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Configure the below parameters :

Search (Ctrl+F)

ADCs\_Common\_Settings

- Mode Independent mode

ADC\_Settings

- Clock Prescaler Asynchronous clock mode divided by 1
- Resolution ADC 12-bit resolution
- Scan Conversion Mode Enabled
- Data Alignment Right alignment
- Continuous Conversion Mode Enabled
- Discontinuous Conversion Mode Disabled
- End Of Conversion Selection End of single conversion
- Overrun behaviour Overrun data preserved
- Conversion Data Management Mode DMA Circular Mode
- Low Power Auto Wait Disabled

ADC\_Regular\_ConversionMode

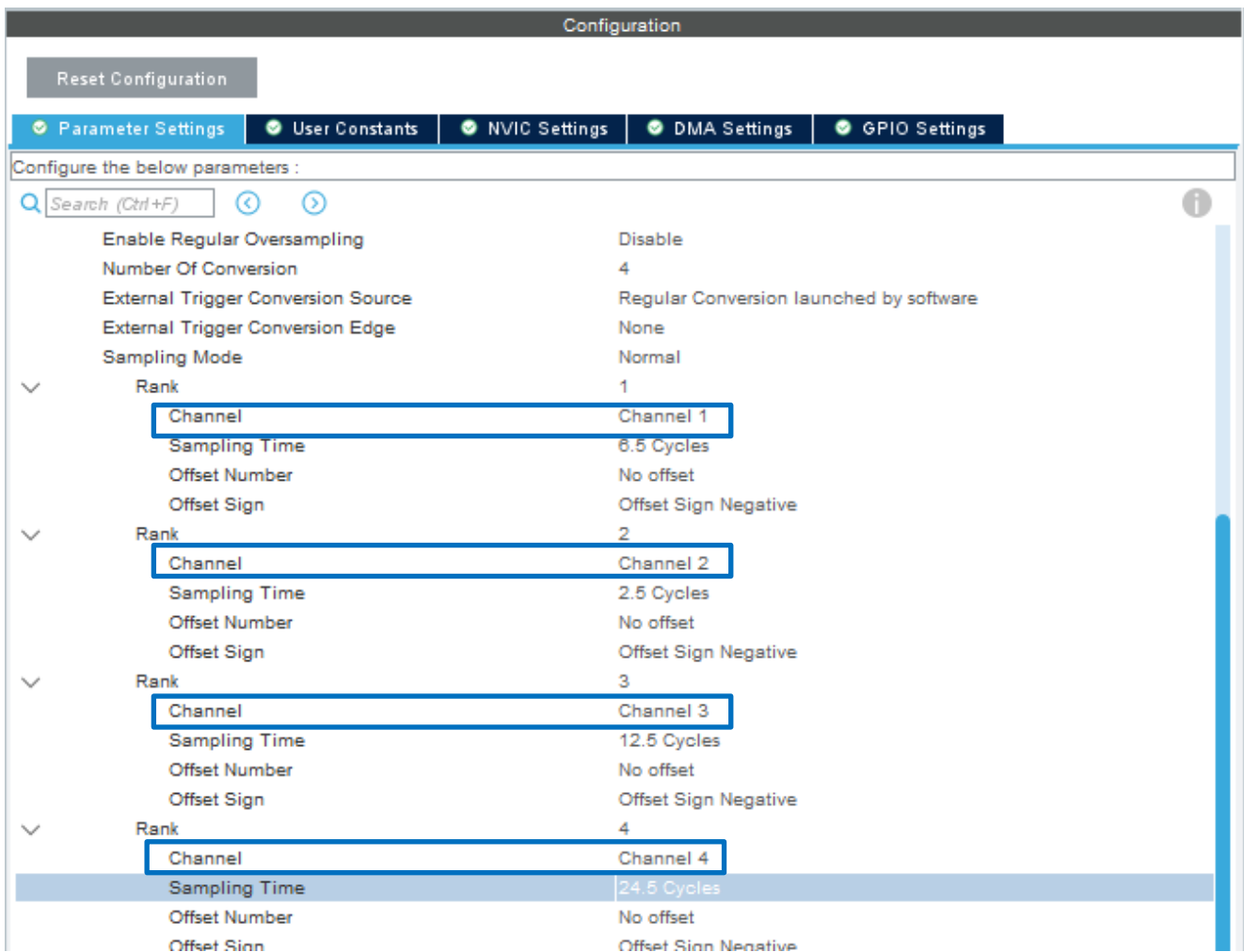
- Enable Regular Conversions Enable
- Enable Regular Oversampling Disable
- Number Of Conversion 4
- External Trigger Conversion Source Regular Conversion launched by software
- External Trigger Conversion Edge None
- Sampling Mode Normal
- Rank 1
- Rank 2
- Rank 3
- Rank 4

ADC\_Injected\_ConversionMode

- Enable Injected Conversions Disable

ADCs\_Watchdog 1

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

ADC1 Mode and Configuration

Mode

Runtime contexts:

	Boot	Application	ExternalMemoryLoader
<input type="checkbox"/> IN0		<input checked="" type="checkbox"/>	
IN1 IN1 Single-ended			
IN2 IN2 Single-ended			
IN3 IN3 Single-ended			
IN4 IN4 Single-ended			
IN5 Disable			
<input type="checkbox"/> 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](#)

Categories A-Z

RTOS

	Boot	Application	ExtMemLoader
<input checked="" type="checkbox"/> SYS	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/> WWDG	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Analog

	Boot	Application	ExtMemLoader
<input checked="" type="checkbox"/> ADC1	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ADC2	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
DTS	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
VREFBUF	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Timers >

Connectivity >

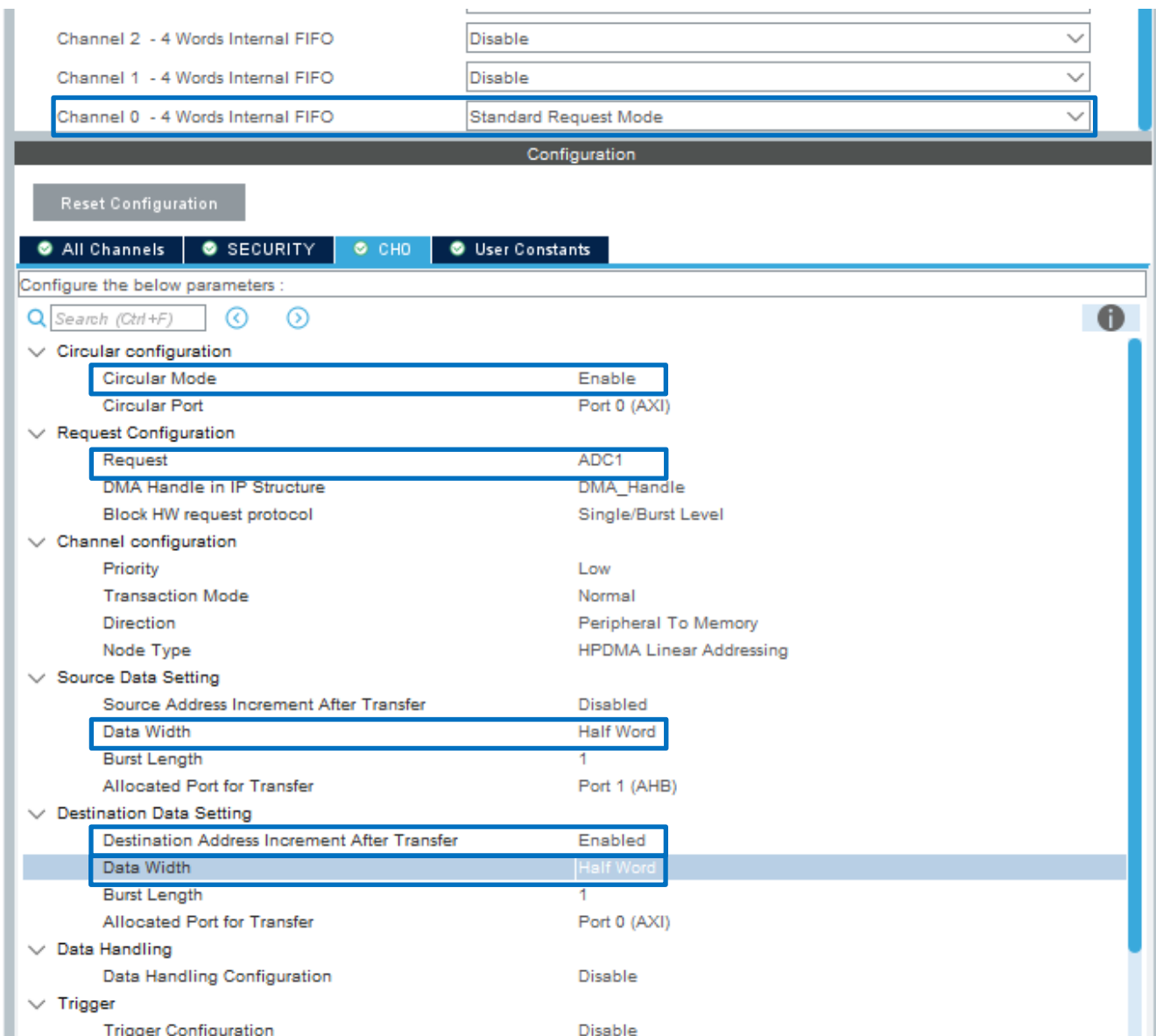
Multimedia >

Security >

Computing >

Middleware and Software Packs

	Boot	Application	ExtMemLo...
<input type="checkbox"/> AIROC-Wi-Fi-Bluetooth-ST...			
EXTMEM_LOADER			<input type="checkbox"/>
<input checked="" type="checkbox"/> EXTMEM_MANAGER	<input checked="" type="checkbox"/>		<input type="checkbox"/>
EXTMEM_MANAGER_APPLI...		<input type="checkbox"/>	
FATFS	<input type="checkbox"/>	<input type="checkbox"/>	
FP-SNS-MOTENVWB1			
FP-SNS-SMARTAG2			
FP-SNS-STBOX1			
FREERTOS	<input type="checkbox"/>	<input type="checkbox"/>	
ICUBE-CANOPEN			



Channel 2 - 4 Words Internal FIFO: Disable

Channel 1 - 4 Words Internal FIFO: Disable

Channel 0 - 4 Words Internal FIFO: Standard Request Mode

Configuration

Reset Configuration

All Channels SECURITY CH0 User Constants

Configure the below parameters :

Search (Ctrl+F)

Circular configuration

Circular Mode: Enable

Circular Port: Port 0 (AXI)

Request Configuration

Request: ADC1

DMA Handle in IP Structure: DMA\_Handle

Block HW request protocol: Single/Burst Level

Channel configuration

Priority: Low

Transaction Mode: Normal

Direction: Peripheral To Memory

Node Type: HPDMA Linear Addressing

Source Data Setting

Source Address Increment After Transfer: Disabled

Data Width: Half Word

Burst Length: 1

Allocated Port for Transfer: Port 1 (AHB)

Destination Data Setting

Destination Address Increment After Transfer: Enabled

Data Width: Half Word

Burst Length: 1

Allocated Port for Transfer: Port 0 (AXI)

Data Handling

Data Handling Configuration: Disable

Trigger

Trigger Configuration: Disable

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 (@0x00000000 / @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
                      )

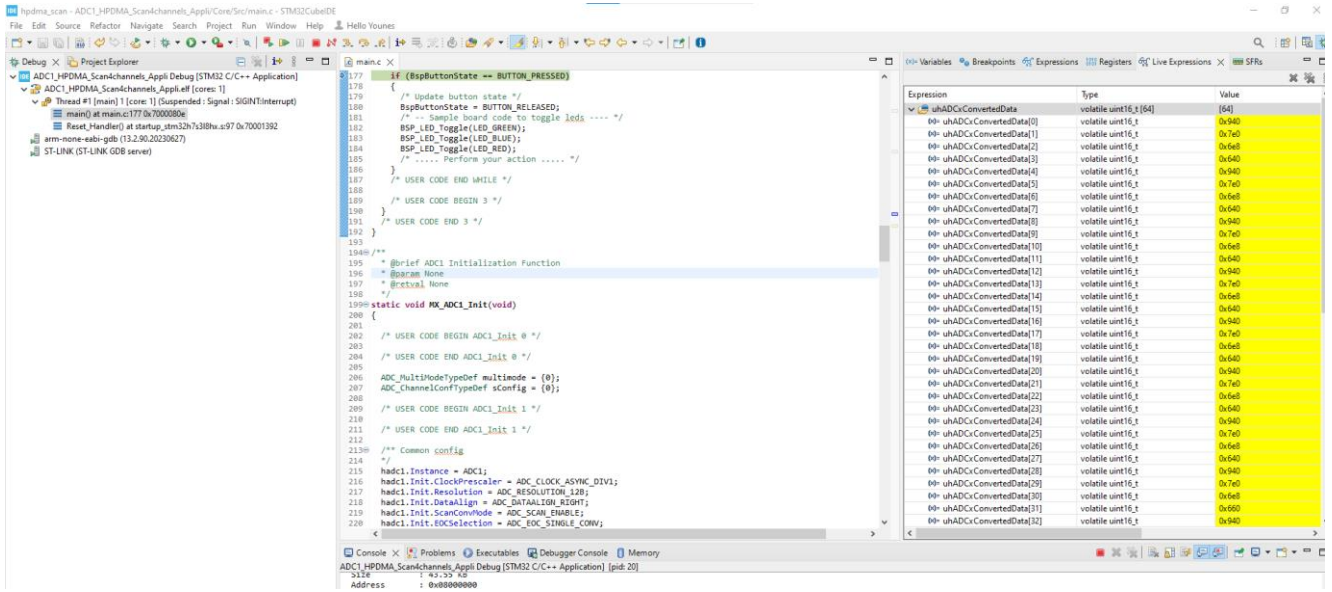
    ...

    ...
}

....
```

See complete code

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



The screenshot shows an IDE with a C source file and a variable watch window. The code defines an initialization function for an ADC channel.

```

177 if (BspButtonState == BUTTON_PRESSED)
178 {
179     /* Update button state */
180     BspButtonState = BUTTON_RELEASED;
181     /* -- Sample board code to toggle leds ---- */
182     BSP_LED_Toggle(LED_GREEN);
183     BSP_LED_Toggle(LED_BLUE);
184     BSP_LED_Toggle(LED_RED);
185     /* ----- Perform your action ----- */
186 }
187 /* USER CODE END WHILE */
188
189 /* USER CODE BEGIN 3 */
190 }
191 /* USER CODE END 3 */
192
193
194 /**
195  * @brief ADC1 Initialization Function
196  * @param None
197  * @retval None
198  */
199 static void MX_ADC1_Init(void)
200 {
201     /* USER CODE BEGIN ADC1_Init 0 */
202     /* USER CODE END ADC1_Init 0 */
203     /* USER CODE BEGIN ADC1_Init 1 */
204     ADC_ChannelConfTypeDef sConfig = {0};
205
206     /* USER CODE BEGIN ADC1_Init 1 */
207     ADC_MultiModeTypeDef multimode = {0};
208     ADC_ChannelConfTypeDef sConfig = {0};
209     /* USER CODE BEGIN ADC1_Init 1 */
210     /* USER CODE END ADC1_Init 1 */
211     /* USER CODE BEGIN ADC1_Init 1 */
212
213     /** Common config
214     */
215     hadc1.Instance = ADC1;
216     hadc1.Init.ClockPrescaler = ADC_CLOCK_ASYNC_DIV1;
217     hadc1.Init.Resolution = ADC_RESOLUTION_12B;
218     hadc1.Init.DataAlign = ADC_DATAALIGN_RIGHT;
219     hadc1.Init.ScanConvMode = ADC_SCAN_ENABLE;
220     hadc1.Init.EOCSelection = ADC_EOC_SINGLE_CONV;
221 }
  
```

The Variable Watch window displays the following data:

Expression	Type	Value
uADCx_ConvertedData[0]	volatile uint16_t [64]	0x940
uADCx_ConvertedData[1]	volatile uint16_t	0x760
uADCx_ConvertedData[2]	volatile uint16_t	0x668
uADCx_ConvertedData[3]	volatile uint16_t	0x640
uADCx_ConvertedData[4]	volatile uint16_t	0x940
uADCx_ConvertedData[5]	volatile uint16_t	0x760
uADCx_ConvertedData[6]	volatile uint16_t	0x668
uADCx_ConvertedData[7]	volatile uint16_t	0x640
uADCx_ConvertedData[8]	volatile uint16_t	0x940
uADCx_ConvertedData[9]	volatile uint16_t	0x760
uADCx_ConvertedData[10]	volatile uint16_t	0x668
uADCx_ConvertedData[11]	volatile uint16_t	0x640
uADCx_ConvertedData[12]	volatile uint16_t	0x940
uADCx_ConvertedData[13]	volatile uint16_t	0x760
uADCx_ConvertedData[14]	volatile uint16_t	0x668
uADCx_ConvertedData[15]	volatile uint16_t	0x640
uADCx_ConvertedData[16]	volatile uint16_t	0x940
uADCx_ConvertedData[17]	volatile uint16_t	0x760
uADCx_ConvertedData[18]	volatile uint16_t	0x668
uADCx_ConvertedData[19]	volatile uint16_t	0x640
uADCx_ConvertedData[20]	volatile uint16_t	0x940
uADCx_ConvertedData[21]	volatile uint16_t	0x760
uADCx_ConvertedData[22]	volatile uint16_t	0x668
uADCx_ConvertedData[23]	volatile uint16_t	0x640
uADCx_ConvertedData[24]	volatile uint16_t	0x940
uADCx_ConvertedData[25]	volatile uint16_t	0x760
uADCx_ConvertedData[26]	volatile uint16_t	0x668
uADCx_ConvertedData[27]	volatile uint16_t	0x640
uADCx_ConvertedData[28]	volatile uint16_t	0x940
uADCx_ConvertedData[29]	volatile uint16_t	0x760
uADCx_ConvertedData[30]	volatile uint16_t	0x668
uADCx_ConvertedData[31]	volatile uint16_t	0x640
uADCx_ConvertedData[32]	volatile uint16_t	0x940

The console shows the following output:

```

ADC1_HPDMAScanChannels_App0 Debug [STM32 C/C++ Application] [pid:20]
3158 : 43.33 KHz
Address : 0x80000000
  
```