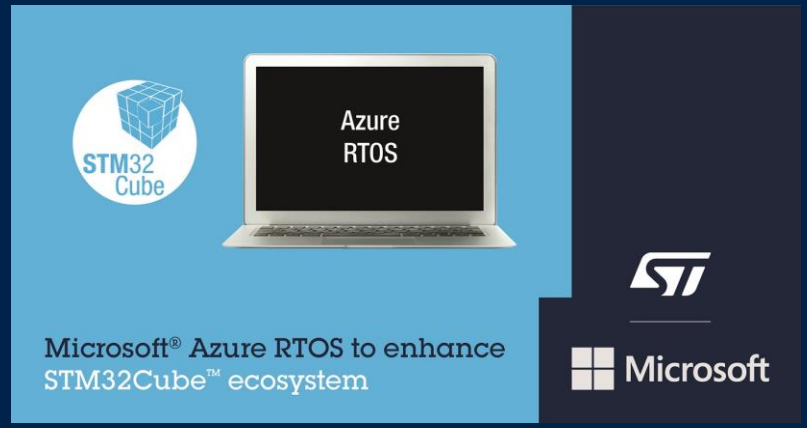
life.augmented

STM32Cube enhanced with Azure RTOS

ST MCD Korea
문현수 과장



Agenda

1 STM32Cube and Azure RTOS

2 STM32H7 Overview

3 Create Azure RTOS projects for STM32H7

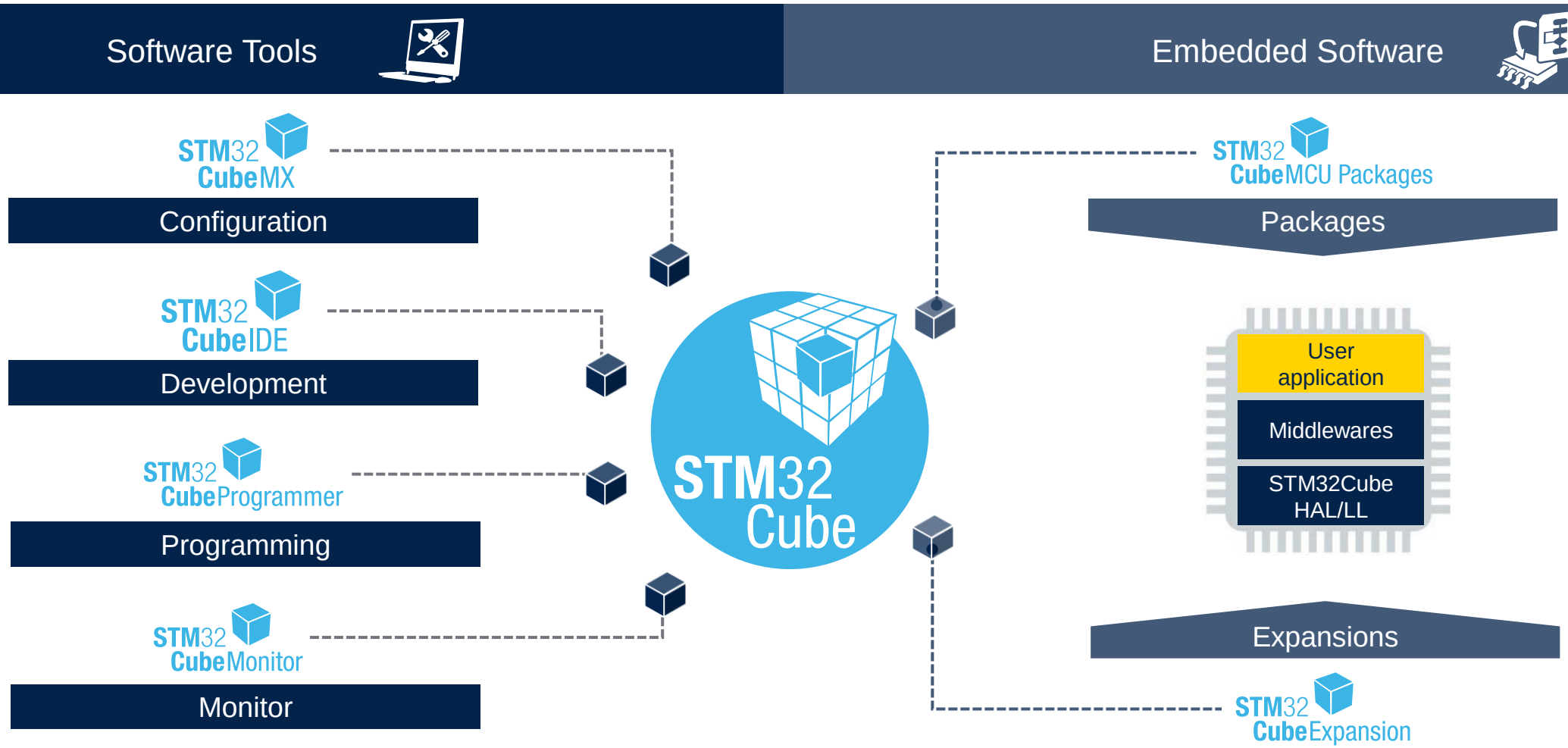
4 Resources

life.augmented

STM32Cube and Azure RTOS

life.augmented

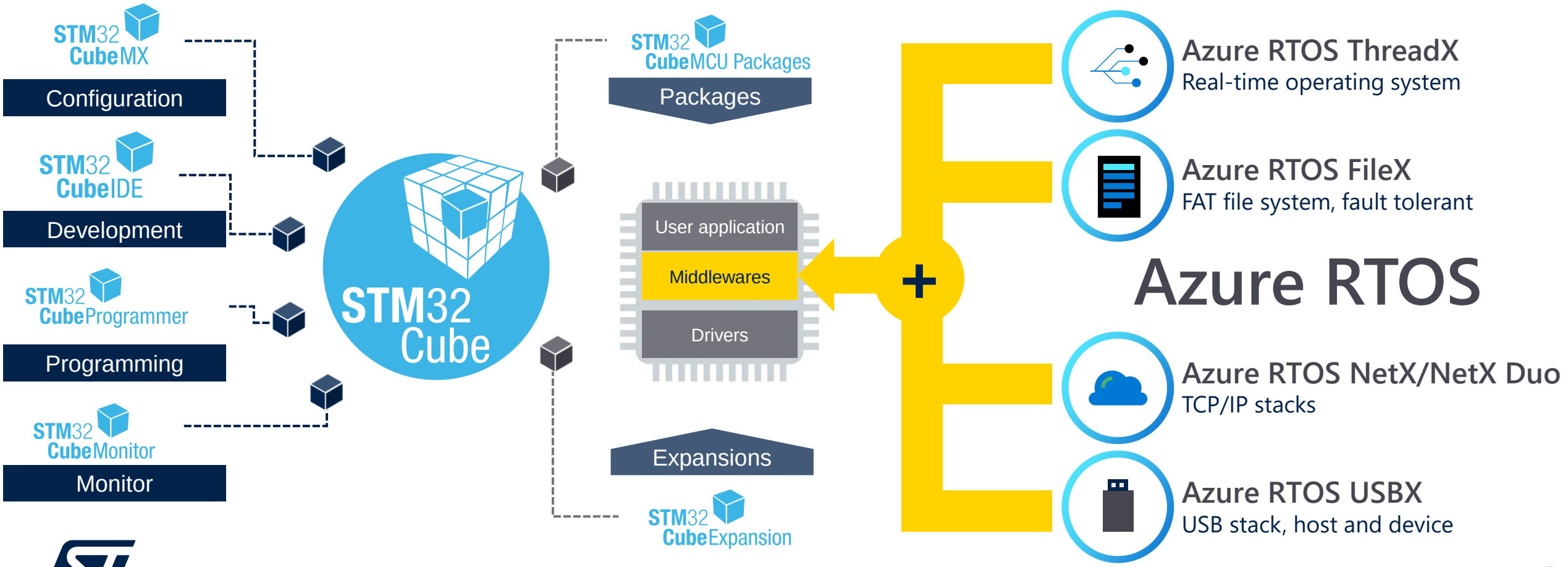
Today - STM32Cube Software Suite Offer



ST proposal for STM32, since 2014

Ongoing - STM32Cube Software Suite

Software Tools | Embedded Software | **Complemented with Microsoft Azure RTOS (2021)**



life.augmented

Ongoing - STM32Cube Software Suite

Microsoft Azure RTOS bringing additional Key benefits to STM32Cube software Suite, from 2021

Microsoft Azure

+ Azure RTOS

Faster & Easier Development

Business-friendly terms

Better Quality

+

Fast performance

Complete consistent solution

Industry certifications

life.augmented

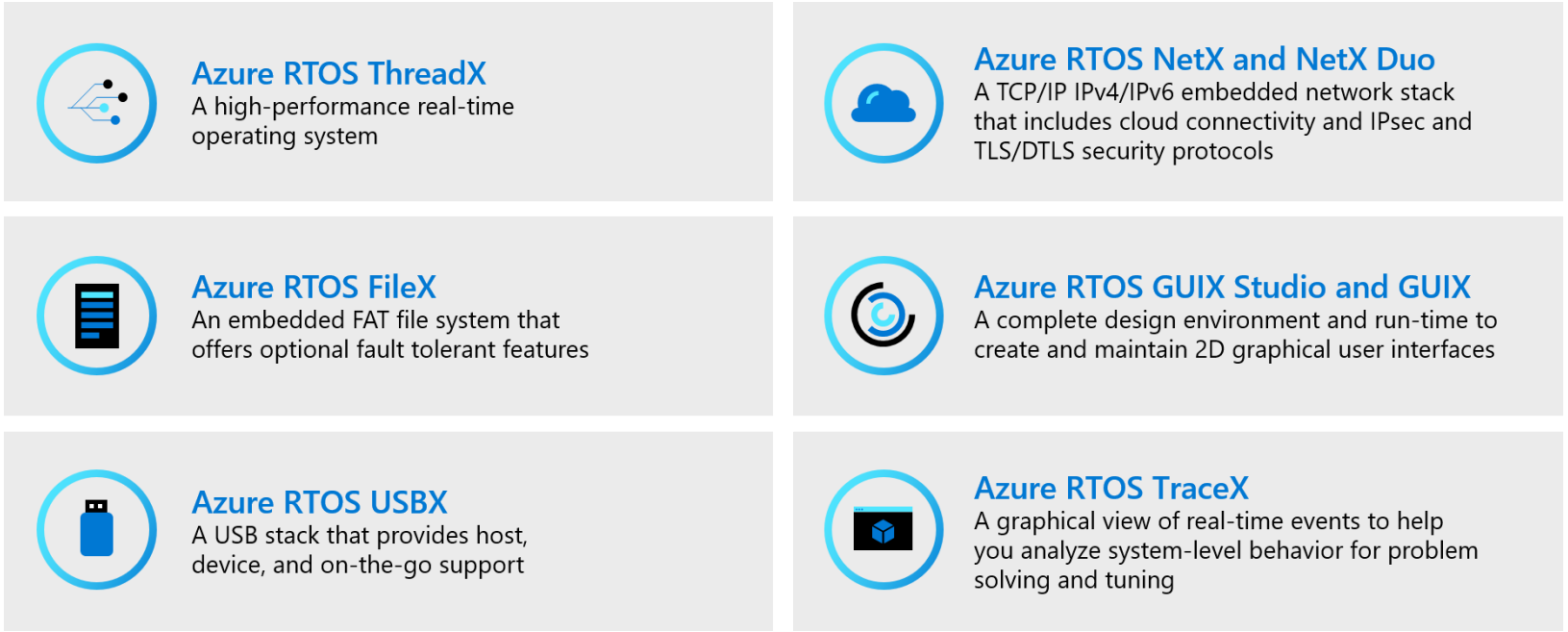
Microsoft Azure RTOS: Key Features

Professional grade, highly reliable and market proven MW suite

- **Industrial grade** networking stack: optimized for performance coming with **many IoT protocols**
- Advanced FS/FTL: **fully featured** to support **NAND/NOR Flash** memories
- USB **Host** and **Device** stacks coming with **many classes**
- **Safety pre-certifications (from MSFT)**: IEC 61508 SIL4, IEC 62304 Class C and ISO 26262 ASIL D
- **Security pre-certifications (from MSFT)**: EAL4+ for TLS/DTLS, FIPS 140-2 for SW crypto lib
- STM32 **granted production license:** https://github.com/azure-rtos/guix/blob/master/LICENSED-HARDWARE.txt

Azure RTOS Overview

- Azure RTOS is an embedded development suite including RTOS, File System, USB ,Networking, Graphics MW stacks and Tracing tools



Partnership between ST and Microsoft for Azure RTOS is mainly on ThreadX, USBX, FileX and NetX. ST can also take benefit from **GUIX** or **TraceX** at its decision, but **TouchGFX** and **STM32CubeMonitor** are considered as better options.

Getting started with Azure RTOS examples

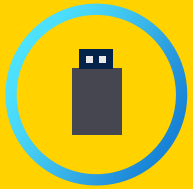
Start from STM32CubeMX or from ready-to-use examples to easily get up to speed with Azure RTOS

Azure RTOS ThreadX

Thread
Creation, Synchronization, message queue

OS wrappers
FreeRTOS
CMSIS OS

Azure RTOS USBX



Host
MSC, HID, CDC ACM
Dual class

Device
MSC, CDC ACM,
HID, CDC ECM,
HID CDC ACM (dual-class)

Azure RTOS NetX Duo

TCP
Server, Client

UDP
Server, Client

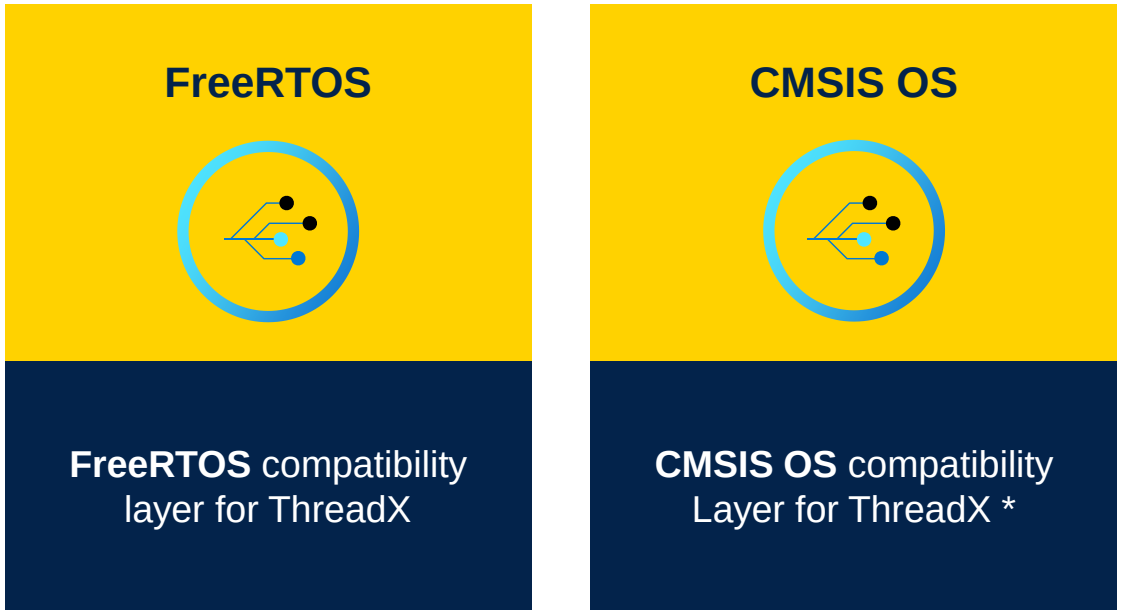
Application
Web server, MQTT client, SNTP client

Azure RTOS FileX

Micro SD File edit
Multi-thread access
NOR memory File RW
NAND memory File RW
Multi-instance
In-Application-Programming

Migrate to Azure RTOS

Keep your application layer, simply integrate an industry-leading real time operating system



* Offer exclusive to the STM32

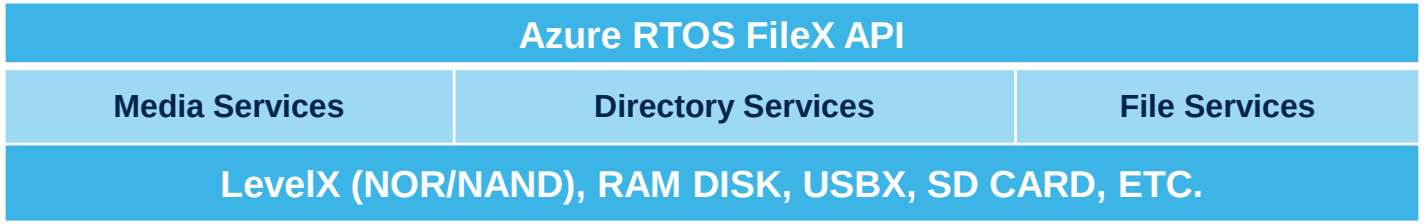
Azure RTOS: ThreadX

- **Small:** ~ 2KB Minimal Footprint
- **Fast:** Sub microsecond context switch, APIs
- **Safe:** SIL 4, ASIL D, Medical Class C
- **Security:** Extensive Pen Testing, Part of EAL4+, FIPS 140-2
- **Advanced:** Preemption-threshold, Event Chaining, Auto Scaling
- **Easy:** Consistent API, Extensive out-of-box examples

Azure RTOS ThreadX API	
Thread Service	Messaging Queues
Counting Semaphores	Mutexes
Event Flags	Block Memory Pools
Byte Memory Pools	Application Timers
Azure RTOS ThreadX Core Scheduler	

Azure RTOS: FileX

- **Small:** ~ 9KB Minimal Footprint
- **Fast:** Direct Data Write, Cache optimized for speed
- **Safe:** SIL 4, ASIL D, Medical Class C
- **Advanced:** Fault tolerant, FAT 12/16/32/exFAT, Extensive Cache Support, NAND/NOR Wear Leveling, Auto Scaling
- **Easy:** Consistent API, Extensive out-of-box examples



Azure RTOS: NetX Duo

- **Small:** ~50KB Device-to-Cloud
- **Fast:** Near Wire Speed, MinmalCPU usage
- **Safe:** SIL 4, ASIL D, Medical Class C
- **Security:** Extensive Pen Testing, EAL4+, FIPS 140-2
- **Advanced:** Extensive Components, Zero Copy, Auto Scaling
- **Easy:** Consistent API, Extensive out-of-box examples

<table>
<tr><td>MQTT</td><td>CoAP</td><td>LwM2M</td></tr>
<tr><td>Auto IP</td><td>HTTP, HTTPS</td><td>SMTP</td></tr>
<tr><td>DHCP</td><td>NAT</td><td>SNMP</td></tr>
<tr><td>DNS, mDNS, DNS-SD</td><td>POP3</td><td>Telnet</td></tr>
<tr><td>FTP, TFTP</td><td>PPP, PPPoE</td><td>PTP, SNTP</td></tr>
<tr><td colspan="3">Azure RTOS NetX Duo</td></tr>
<tr><td>IGMP</td><td>NetX Secure TLS</td><td>NetX Secure DTLS</td></tr>
<tr><td>ICMP</td><td>IPV4 & IPV6</td><td rowspan="2">Azure RTOS NetX Secure IPsec</td></tr>
<tr><td rowspan="2">ARP/PAPR</td><td>UDP</td></tr>
<tr><td>6LoWPAN</td><td>TCP</td></tr>
<tr><td colspan="3">Ethernet, Wi-Fi, Bluetooth LE, 15.4, etc.</td></tr>
<tr><td colspan="3">Azure RTOS ThreadX</td></tr>
</table>

life.augmented

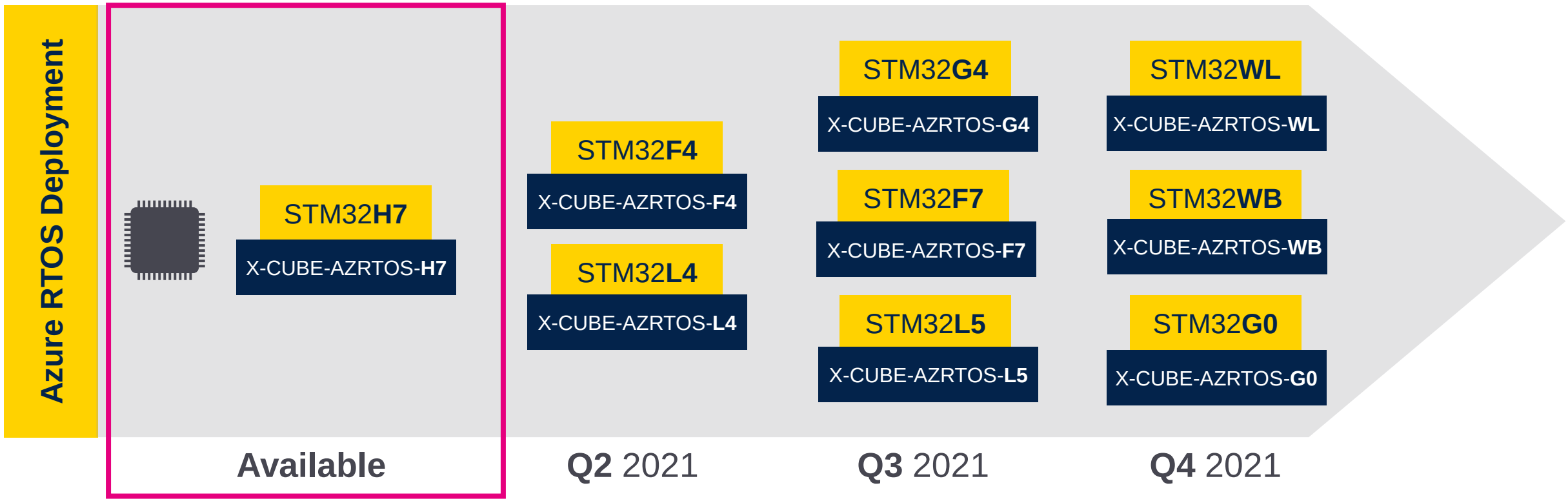
Azure RTOS: USBX

- **Small:** ~8.5KB Device, ~12KB Host
- **Fast:** Leverages DMA, Minimal Function Call Layering
- **Safe:** SIL 4, ASIL D, Medical Class C
- **Advanced:** Comprehensive class support
- **Easy:** Consistent API, Extensive out-of-box examples and device/host controller integration

<table>
<tr><th colspan="2">Azure RTOS USBX Host API</th><th>USB Device API</th></tr>
<tr><td>ASIX</td><td>Hub</td><td>CDC/ACM</td></tr>
<tr><td>Audio</td><td>PIMA(PTP/MTP)</td><td>CDC/ECM</td></tr>
<tr><td>CDC/ACM</td><td>Printer</td><td>DFU</td></tr>
<tr><td>GSER</td><td>Prolific</td><td>HID</td></tr>
<tr><td>HID</td><td>Storage</td><td>PIMA(PTP/MTP)</td></tr>
<tr><td>CDC/ECM</td><td></td><td>Storage</td></tr>
<tr><td colspan="2">Azure RTOS USBX Host Stack</td><td>RNDIS</td></tr>
<tr><td colspan="2"></td><td>USBX Device Stack</td></tr>
<tr><td colspan="2">OHCI, EHCI, Proprietary Host Controllers</td><td>Proprietary Device Controllers</td></tr>
</table>

Azure RTOS Deployment within STM32 and STM32Cube portfolio

STM32Cube Expansions for existing STM32 series, enhanced for ST Toolset

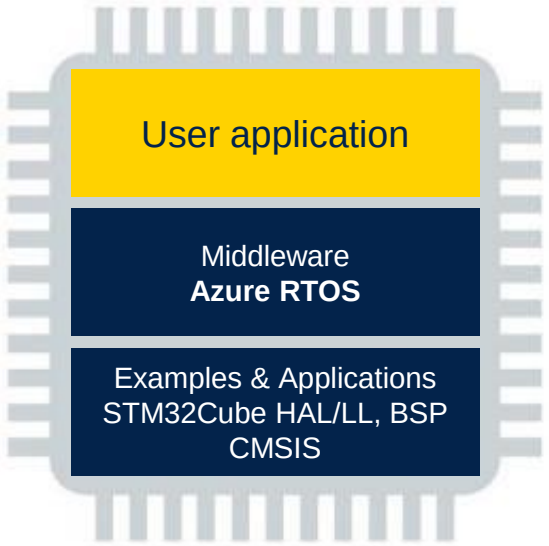


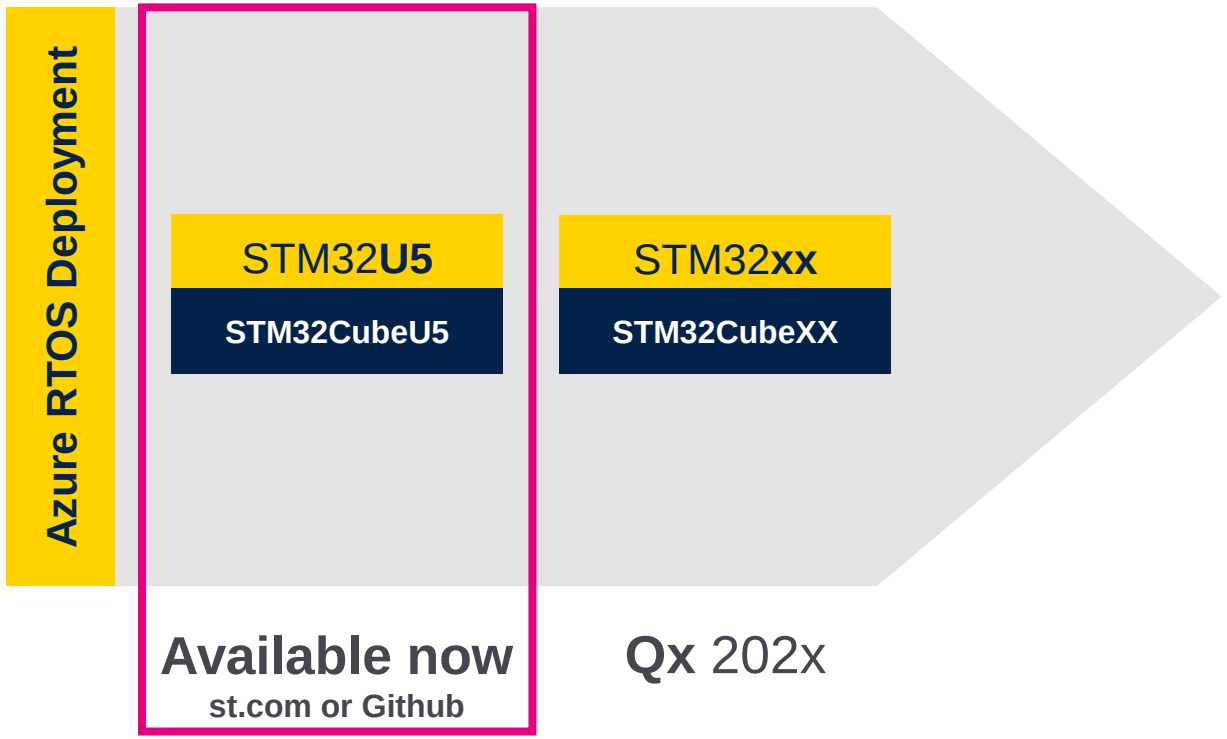
www.st.com/x-cube

Azure RTOS deployment within STM32 and STM32Cube portfolio

STM32Cube native support for new STM32 series from 2021 onward

STM32Cube MCU package





Azure RTOS and STM32Cube: User Benefits

Getting the most out of Azure RTOS and STM32 MCUs is now easier than ever with STM32Cube

Source code available
Free of charge
User-friendly license terms

Drastically reduced learning curve:
- Azure RTOS made compatible with ST Toolset
- Many applicative examples provided

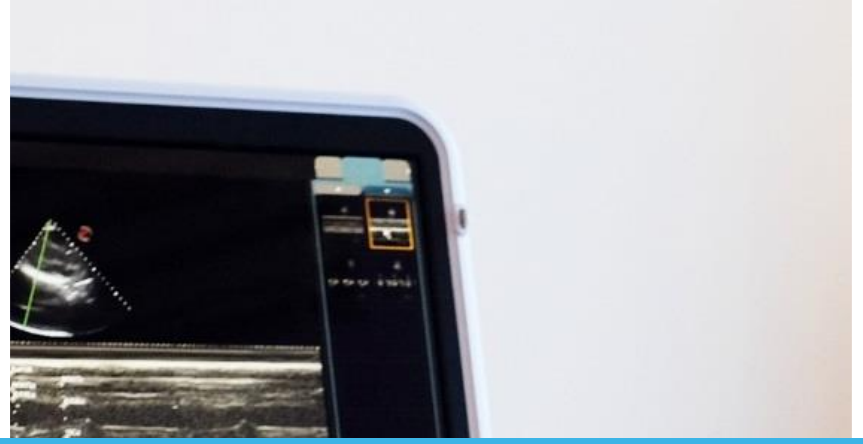
Developers can focus on their application and differentiators

Expanding the ecosystem of existing and upcoming STM32 series by leveraging Azure RTOS middleware

www.st.com/stm32cube

Azure RTOS: License

- Source code is available
- User is granted modification
- User is granted redistribution of source (modified or not) and/or binary, provided he develops on an MCU/MPU that is listed officially by Microsoft
- User is granted production, provided he develops on an MCU that is listed officially by Microsoft
- User is forbidden to reuse provided software to create a competing offer

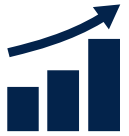


STM32

STM32H7

STM32H7 series

New product lines expanding the STM32 portfolio



New Performance Record
Up to 2424 + 800 CoreMark (Cortex©-M7 @480MHz + Cortex©-M4 @240MHz) in Dual core
Up to 2778 CoreMark (Cortex©-M7 @550MHz) in Single core

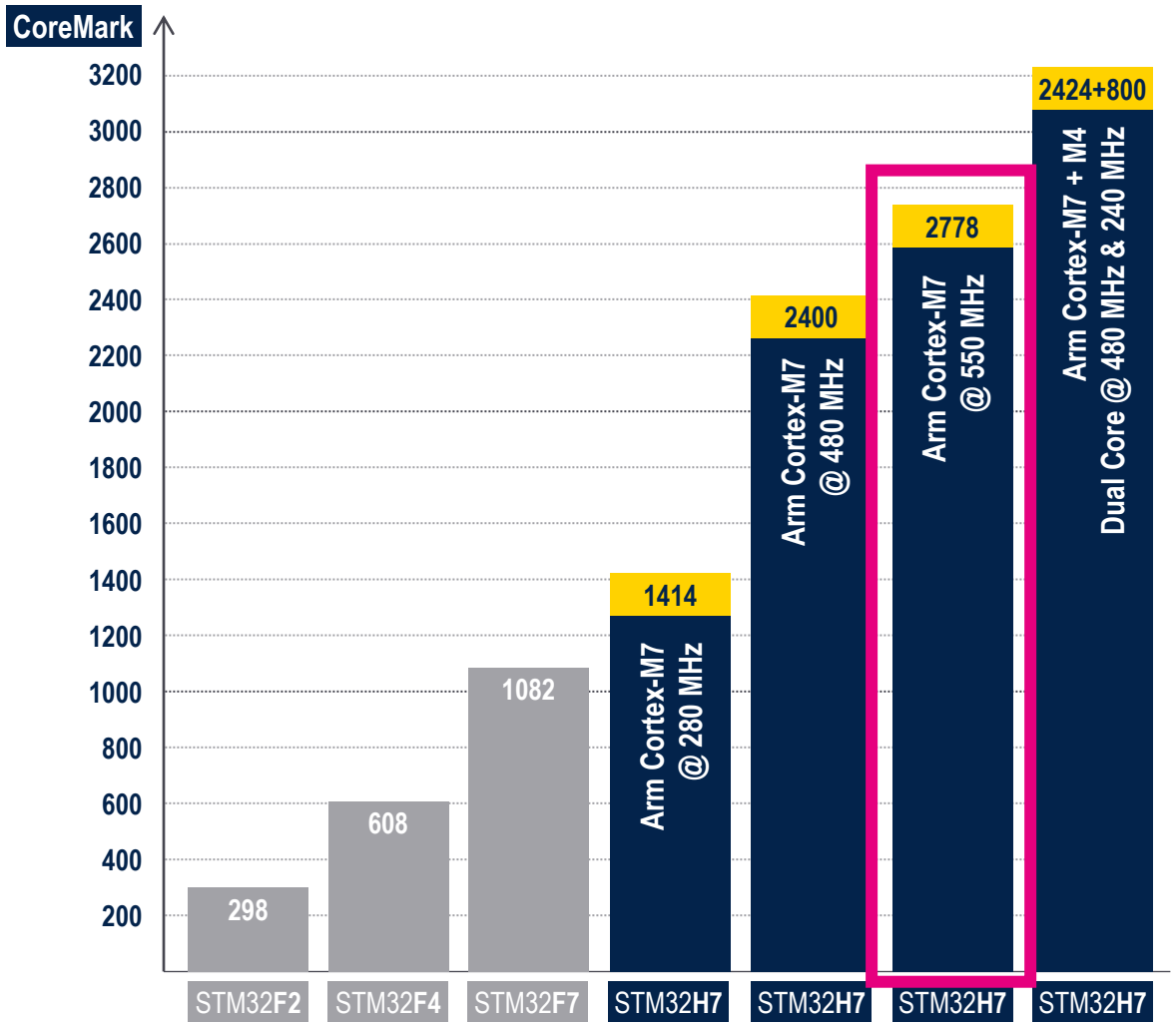
Single and Dual-core flexible architecture for industrial, security or AI applications
Accelerated graphics, fast data transfer, advanced peripherals

Advanced security features
Crypto Hash, Cortex©-M7 Security services

Rich eco-system to speed-up your design
SW tools, HW boards, community and partners

life.augmented

High performance range



Arm® Cortex® -M7 up to 550 MHz

Most powerful Cortex core with double precision FPU, MPU, advanced DSP and L1 cache

Arm® Cortex® -M4 @240 MHz

Best in class core for **real-time** with single precision FPU, DSP, MPU and ART Accelerator™

life.augmented

Powerful cores supported by a powerful architecture

Display nice graphic
The Chrom-ART Accelerator™
and MJPEG codec
offload the CPU by more than 90%

Manage security
Use dedicated **cryptography**
and **Hashing** HW acceleration
to **offload the CPU by more than 90%**

Transfer data efficiently across peripherals
The Main DMA takes care
of the most complex schemes
between memories and peripherals
with **up to 16 channels to offload the CPU**

Generate complex wave forms
High-Resolution timer (2.1ns)
can generate complex wave forms
synchronized on multiples events,
with no CPU assist

STM32H7 MCU series

MPU					STM32**MP1** 4158 CoreMark 650 MHz Cortex –A7 209 MHz Cortex –M4
High Perf MCUs		STM32**F2** Up to 398 CoreMark 120 MHz Cortex-M3	STM32**F4** Up to 608 CoreMark 180 MHz Cortex-M4	STM32**F7** 1082 CoreMark 216 MHz Cortex-M7	STM32**H7** **Up to 3224 CoreMark** **Up to 550 MHz Cortex -M7** **240 MHz Cortex -M4**
Mainstream MCUs	STM32**F0** 106 CoreMark 48 MHz Cortex-M0	STM32**G0** 142 CoreMark 64 MHz Cortex-M0+	STM32**F1** 177 CoreMark 72 MHz Cortex-M3	STM32**F3** ● 245 CoreMark 72 MHz Cortex-M4	STM32**G4** ● 550 CoreMark 170 MHz Cortex-M4
			Optimized for mixed-signal Applications		
Ultra-low Power MCUs	STM32**L0** 75 CoreMark 32 MHz Cortex-M0+	STM32**L1** 93 CoreMark 32 MHz Cortex-M3	STM32**L4** 273 CoreMark 80 MHz Cortex-M4	STM32**L4+** 409 CoreMark 120 MHz Cortex-M4	STM32**L5** 443 CoreMark 110 MHz Cortex-M33	STM32**U5** 651 CoreMark 160 MHz Cortex-M33
Wireless MCUs			STM32**WL** 162 CoreMark 48 MHz Cortex-M4 48 MHz Cortex-M0+	STM32**WB** ● 216 CoreMark 64 MHz Cortex-M4 32 MHz Cortex-M0+	

LONGEVITY 10 YEARS COMMITMENT

life.augmented

● Optimized for mixed-signal applications

● Cortex-M0+ Radio co-processor

Extensive STM32H7 portfolio

Dual-core Line

STM32H745/755
480+240 MHz SMPS
1027 + 300 DMIPS
RAM 1 MB
Flash up to 2 MB

STM32H747/757
480+240 MHz SMPS
1027 + 300 DMIPS
RAM 1 MB
Flash up to 2 MB

Single-core Line

STM32H7A3/B3
280 MHz LDO
599 DMIPS
RAM 1.4 MB
Flash up to 2 MB

STM32H742
480 MHz LDO
1027 DMIPS
RAM 692 KB
Flash up to 2 MB

STM32H743/753
480 MHz LDO
1027 DMIPS
RAM 1 MB
Flash up to 2 MB

STM32H723/733
550 MHz LDO
1177 DMIPS
RAM 564 KB
Flash up to 1 MB

STM32H725/735
550 MHz SMPS
1177 DMIPS
RAM 564 KB
Flash up to 1 MB

Extended temperature range 125 ° C ambient

Value Line

STM32H7B0
280 MHz LDO
599 DMIPS
RAM 1.4 MB
Flash 128 KB

STM32H750
480 MHz LDO
1027 DMIPS
RAM 1 MB
Flash 128 KB

STM32H730
550 MHz LDO
1177 DMIPS
RAM 564 KB
Flash 128 KB

STM32H730Q
550 MHz SMPS
1177 DMIPS
RAM 564 KB
Flash 128 KB

Arm® Cortex® core — **Cortex-M7** — **Cortex-M7 & -M4**

life.augmented

Pick the right STM32H7 development tool

STM32H7 class	Cores/Speed	Part numbers	Evaluation boards	Discovery Kits	Nucleo boards
STM32H74/5	Single Core 480 MHz	**STM32H743**	STM32H743I-EVAL2	-	NUCLEO-H743ZI2
		STM32H753, Crypto enabled	STM32H753I-EVAL2	-	NUCLEO-H753ZI
		STM32H750 Value Line, Crypto enabled	-	STM32H750B-DK	-
	Dual Core 480 MHz + 240 MHz	**STM32H745**	-	STM32H745I-DISCO	NUCLEO-H745ZI-Q
		STM32H747	STM32H747I-EVAL	STM32H747I-DISCO STM32H747I-DISC1	-
		STM32H755/757, Crypto enabled	STM32H757I-EVAL	-	NUCLEO-H755ZI-Q
STM32H7A/B	Single Core 280 MHz	**STM32H7A3**	-	-	NUCLEO-H7A3ZI-Q
		STM32H7B3, Crypto enabled	STM32H7B3I-EVAL	STM32H7B3I-DK	-
		STM32H7B0, Value line, Crypto enabled	*STM32H7B3I-EVAL* *	*STM32H7B3I-DK* *	-
STM32H72/3	Single Core 550 MHz	**STM32H723/733**	-	-	NUCLEO-H723ZG
		STM32H725/735	-	STM32H735G-DK	-
		STM32H730, Value line, Crypto enabled	-	*STM32H735G-DK* *	-

* Recommended board (no dedicated board for this part number)

life.augmented

Create Azure RTOS projects for STM32H7

life.augmented

Prerequisites

- STM32CubeMX

- https://www.st.com/en/development-tools/stm32cubemx.html

- STM32CubeIDE

- https://www.st.com/en/development-tools/stm32cubeide.html

- X-CUBE-AZRTOS-H7

- https://www.st.com/en/embedded-software/x-cube-azrtos-h7.html

X-CUBE-AZRTOS-H7

Start from st.com

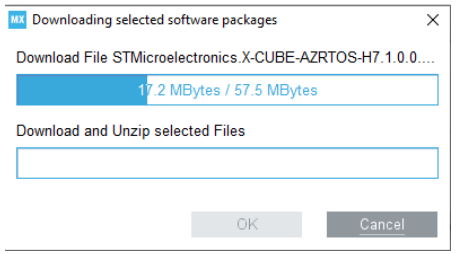
- en.x-cube-azrtos-h7
  - _htmresc
  - Drivers
  - Middlewares
    - ST
      - cmsis_rtos_threadx
        - cmsis_os2.c
        - README.md
      - filex
      - levelx
      - netxduo
      - threadx
      - usbx
  - Projects
    - NUCLEO-H723ZG
    - STM32H735G-DK
      - Applications
        - FileX
        - NetXDuo
        - ThreadX
          - Tx_FreeRTOS_Wrapper
          - Tx_Thread_Sync
            - EWARM
            - Inc
            - MDK-ARM
            - Src
            - STM32CubeIDE
            - .extSettings
            - readme.html
            - README.md
            - Tx_Thread_Sync.ioc
        - USBX
    - STM32H743I-EVAL
    - STM32H747I-DISCO
    - X-CUBE-AZRTOS-H7_ProjectsList.html
  - STM32CubeMX
    - templates
    - readme.txt
    - STMicroelectronics.X-CUBE-AZRTOS-H7.1.0.0_Configs.xml
    - STMicroelectronics.X-CUBE-AZRTOS-H7.1.0.0_Modes.xml
  - Package_License.html
  - Release_Notes.html
  - STMicroelectronics.X-CUBE-AZRTOS-H7.pdsc

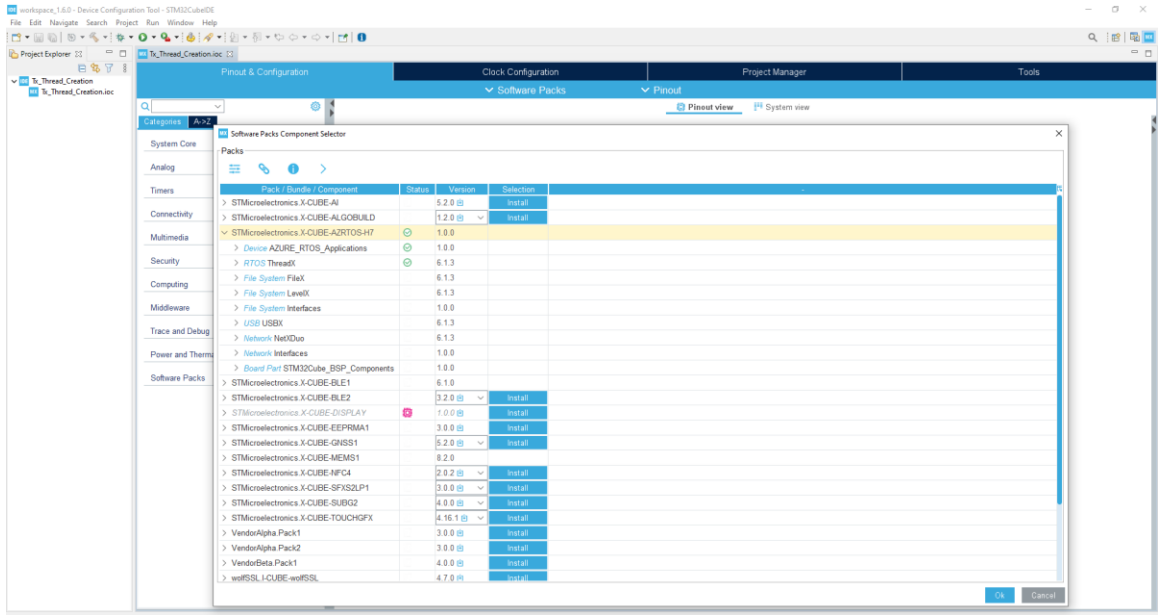
Self contained package with Azure RTOS MW, CubeMX files & ready to use examples

Start from CubeMX/CubeIDE

STM32 CubeMX

STM32 CubeIDE





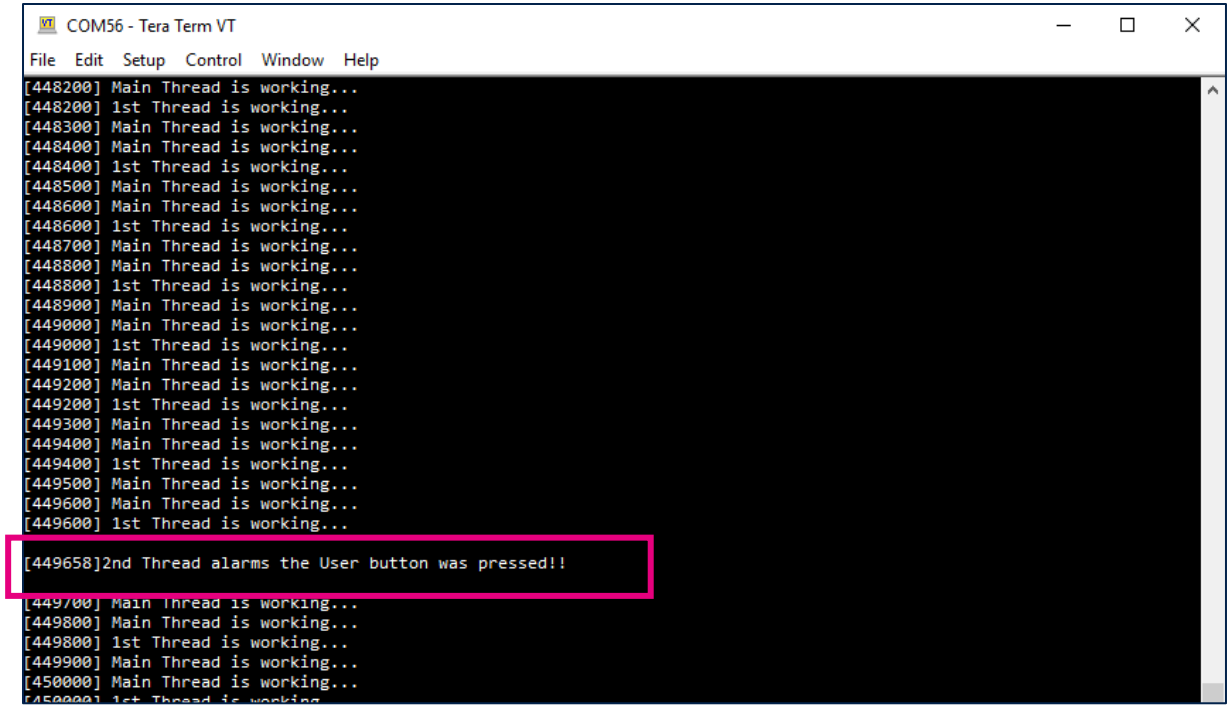
Start from GitHub

- x-cube-azrtos-h7-main
  - .github
  - _htmresc
  - Drivers
  - Middlewares
    - ST
      - cmsis_rtos_threadx
        - cmsis_os2.c
        - README.md
      - filex
      - levelx
      - netxduo
      - threadx
      - usbx
  - Projects
    - NUCLEO-H723ZG
    - STM32H735G-DK
      - Applications
        - FileX
        - NetXDuo
        - ThreadX
          - Tx_FreeRTOS_Wrapper
          - Tx_Thread_Sync
            - EWARM
            - Inc
            - MDK-ARM
            - Src
            - STM32CubeIDE
            - readme.html
            - README.md
        - USBX
    - STM32H743I-EVAL
    - STM32H747I-DISCO
    - X-CUBE-AZRTOS-H7_ProjectsList.html
  - .gitignore
  - CODE_OF_CONDUCT.md
  - CONTRIBUTING.md
  - LICENSE.md
  - README.md

No possibility to configure Azure RTOS MW in CubeMX & regenerate code

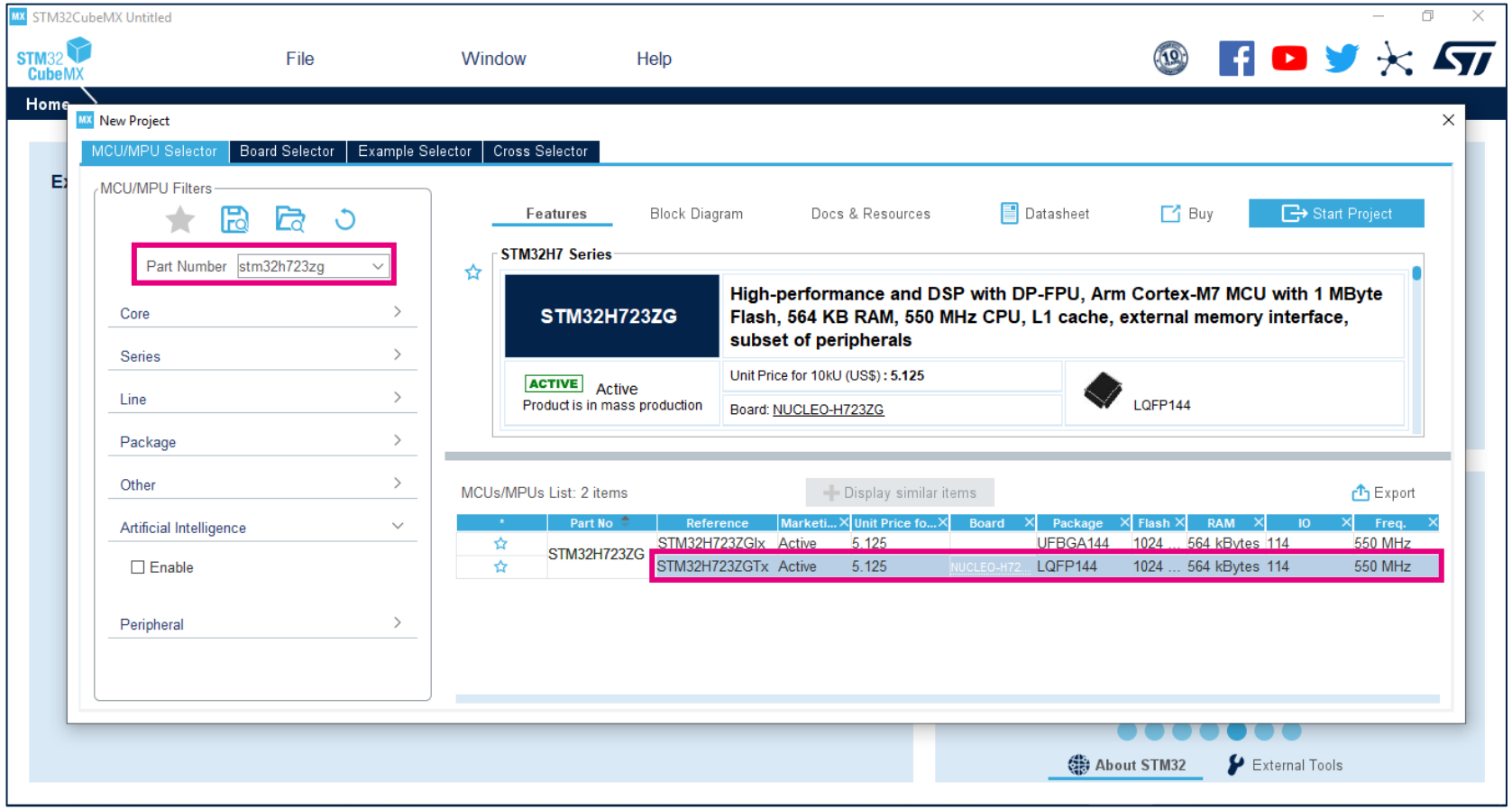
Example overview

- 3개의 Thread 동작
  - Main Thread: 1s 마다 로깅
  - First Thread: 2s 마다 로깅
  - Second Thread: Key Event Flag Getting
    - User button 입력 시 마다 로깅
- User button 으로 키 입력을 받음
- EXTI interrupt에서 Event Flag Setting



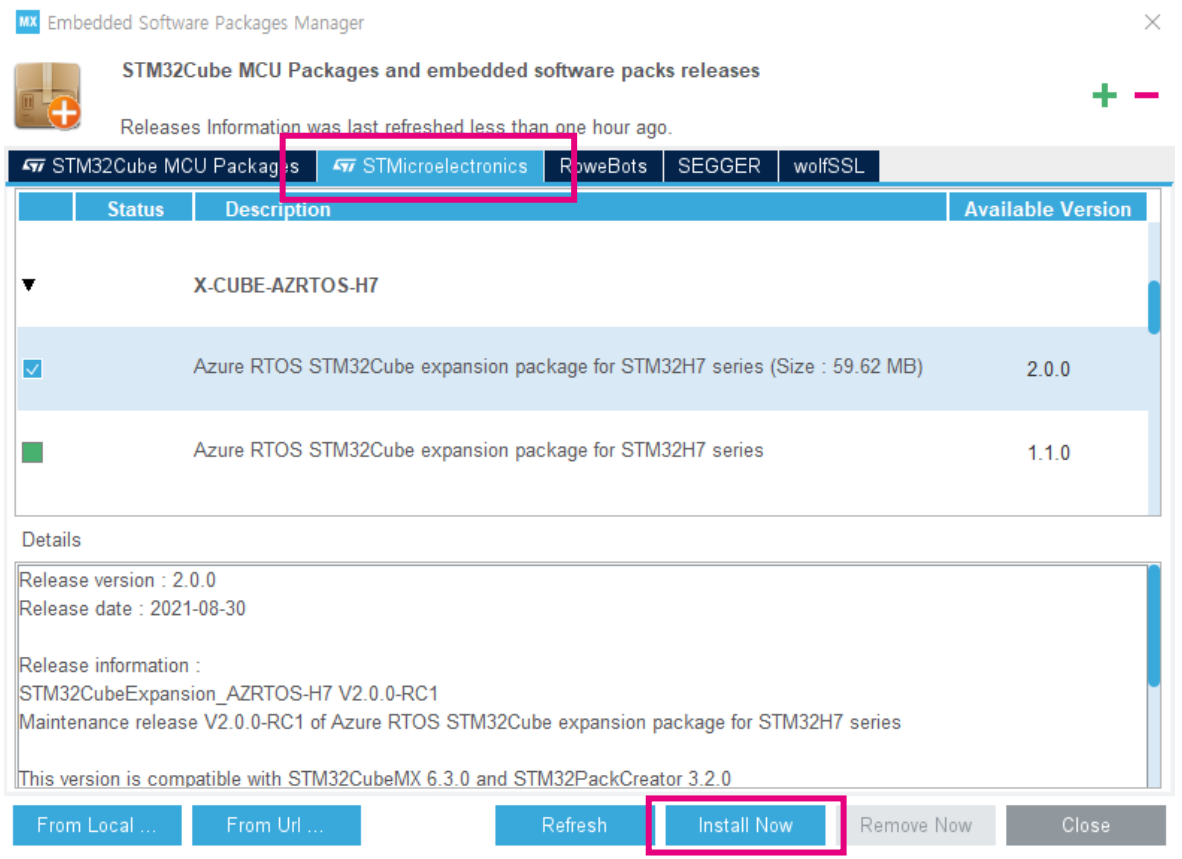
STM32CubeMX 설정

1. File > New Project
2. Part Number
   - **stm32h723zg**
3. MCU List
   - **Nucleo-H723**



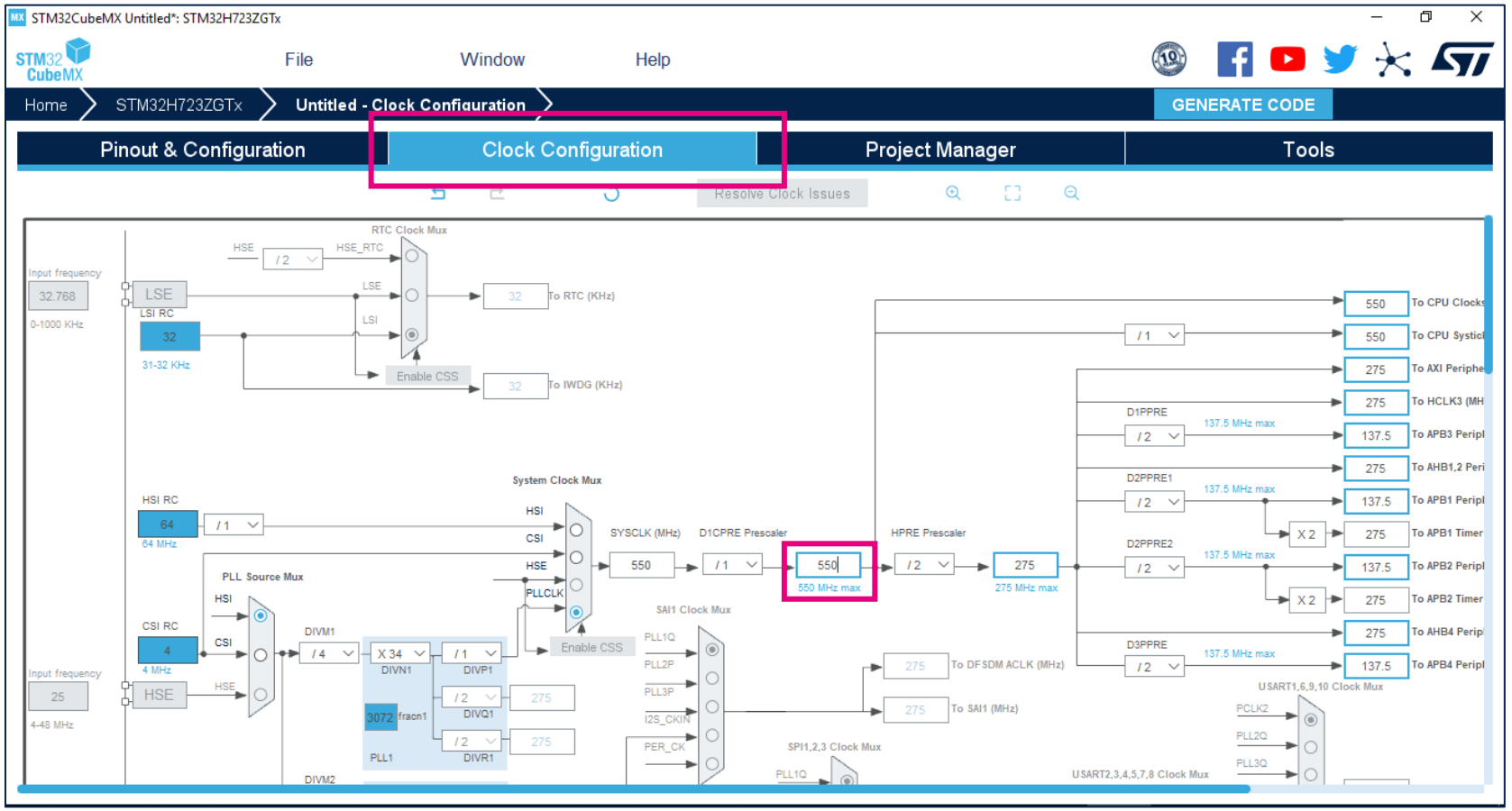
STM32CubeMX 설정

- STM32CubeMX에 X-CUBE-AZRTOS-H 확장팩 설치
  1. Help > Manage Embedded Software Package
  2. STMicroelectronics Tab
  3. X-CUBE-AZRTOS-H7 최신 버전 설치



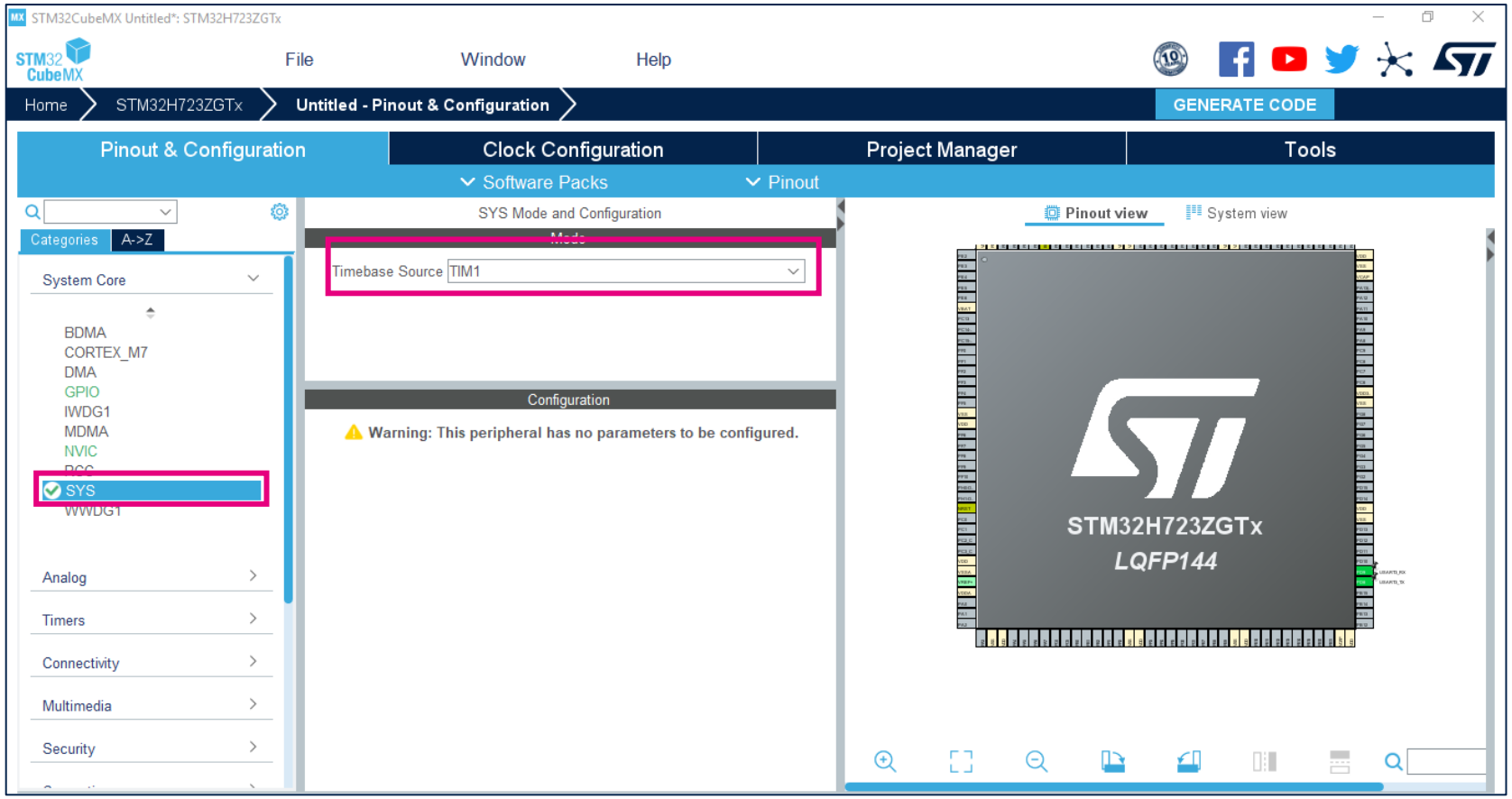
STM32CubeMX 설정

- Clock Configuration
  - 550MHz



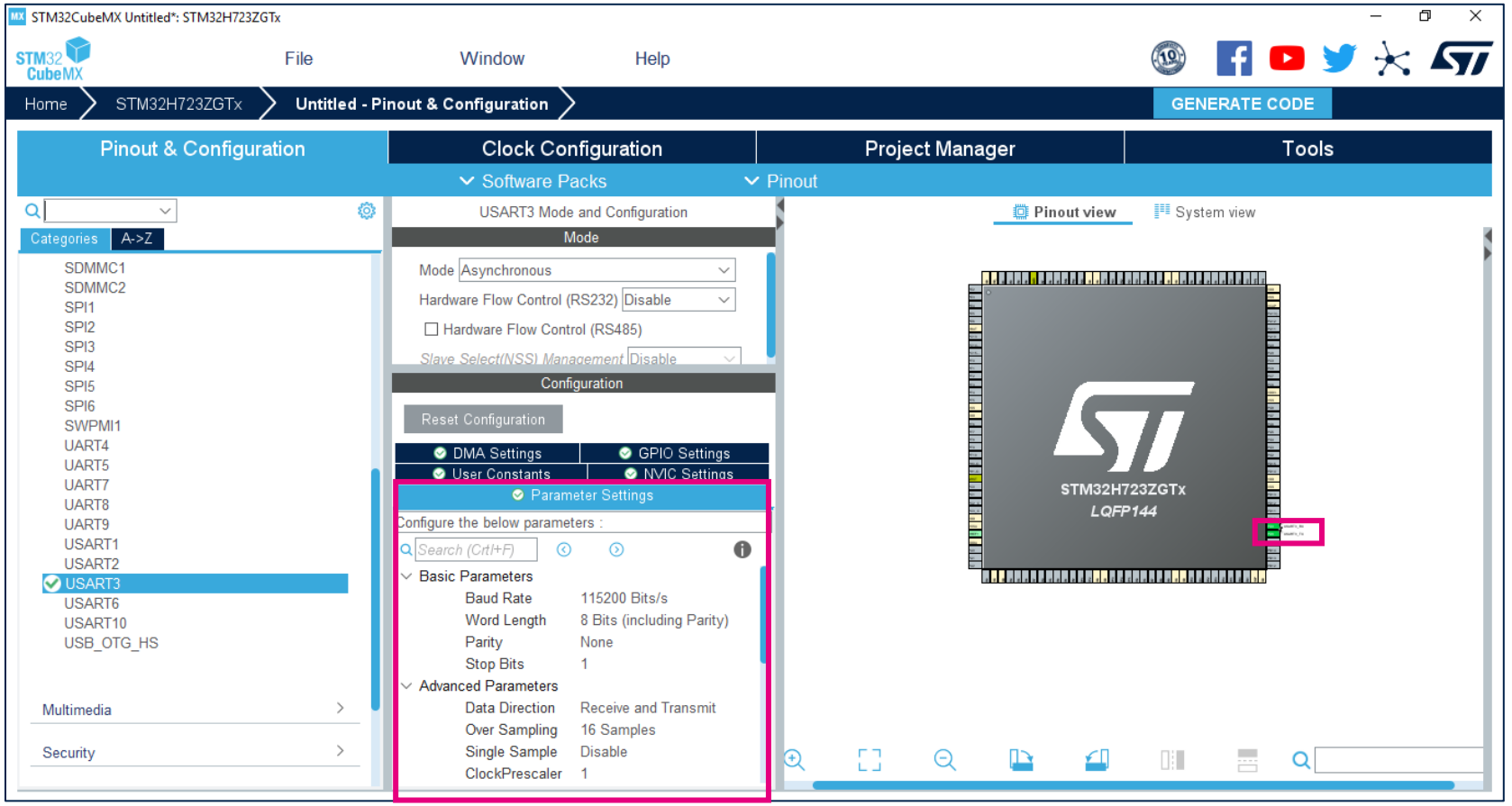
STM32CubeMX 설정

- System timer 변경
  - System Core > SYS
  - Timebase Source
    - TIM1 으로 변경



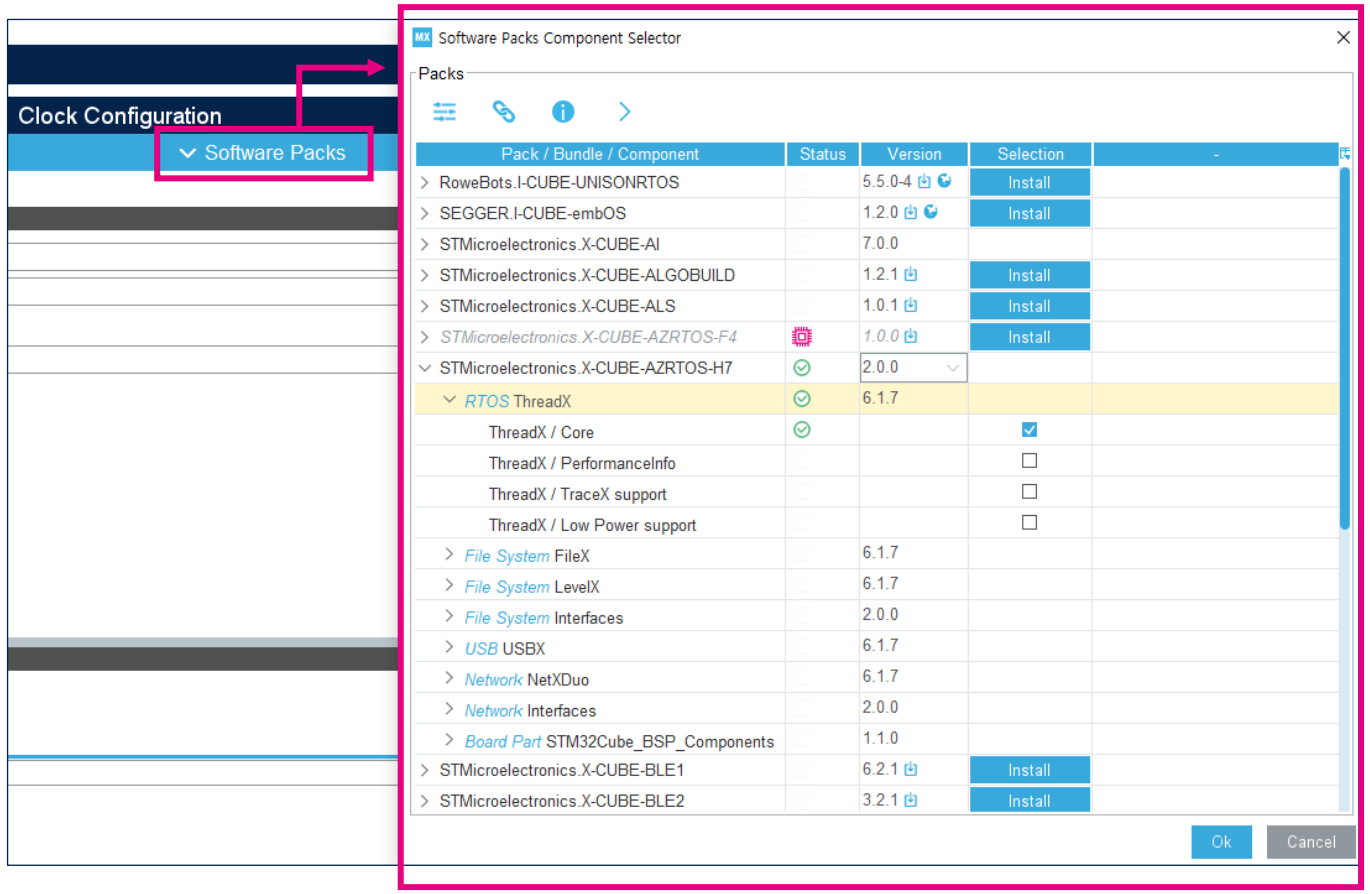
STM32CubeMX 설정

- 디버깅 용 UART 설정
  1. Connectivity:
     USART3
  2. Pin assigned:
     PD8(TX), PD9(RX)
  3. Mode:
     Asynchronous
  4. Parameter:
     Baud Rate - 115200
     Word Length - 8bits
     Parity - None
     Stop Bits - 1



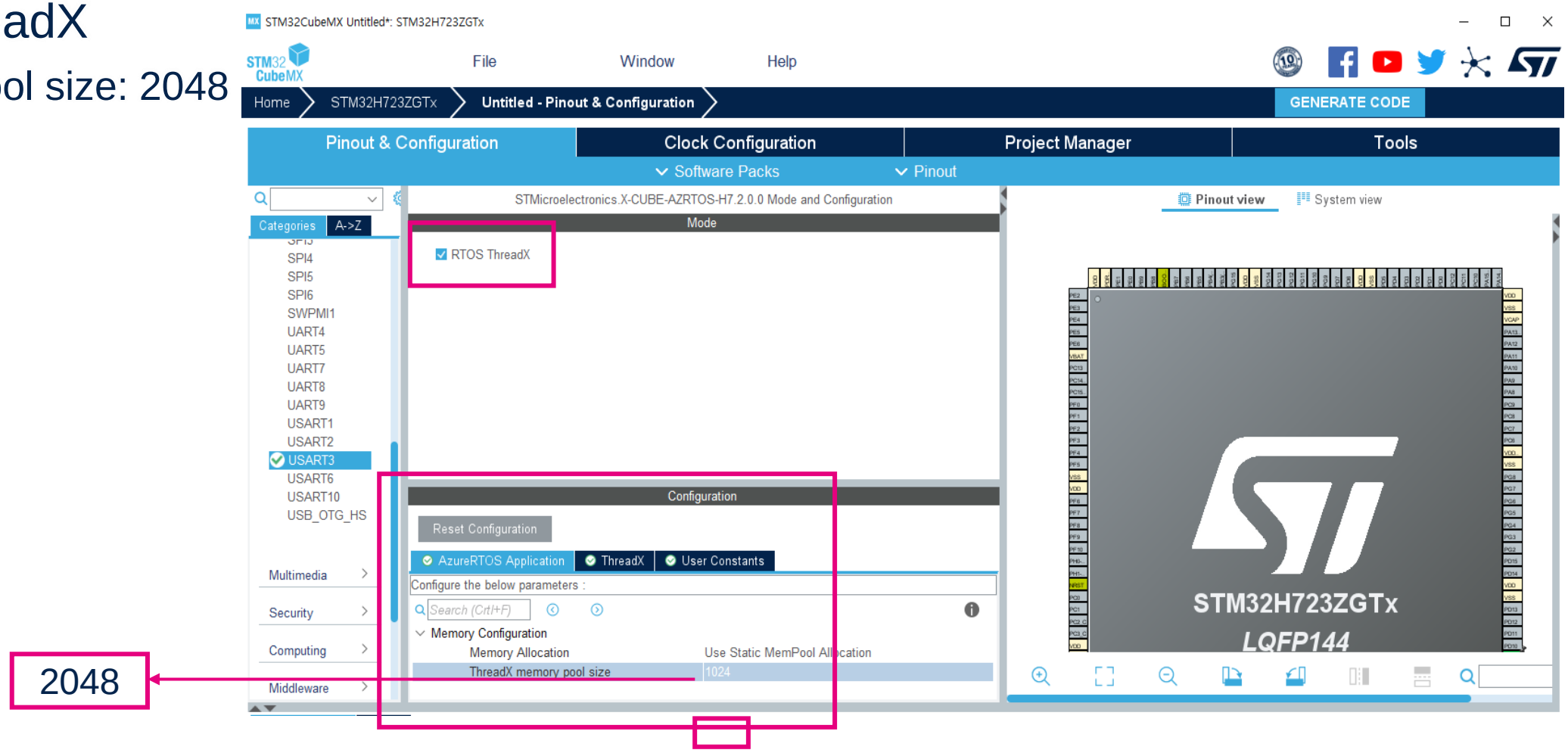
STM32CubeMX 설정

- Software Packs Component Selector
  1. Device AZURE_RTOS Applications
     - HW_Profile: **STM32H723**
     - Application: **azure_rtos_app**
  2. RTOS ThreadX
     - **ThreadX / Core**



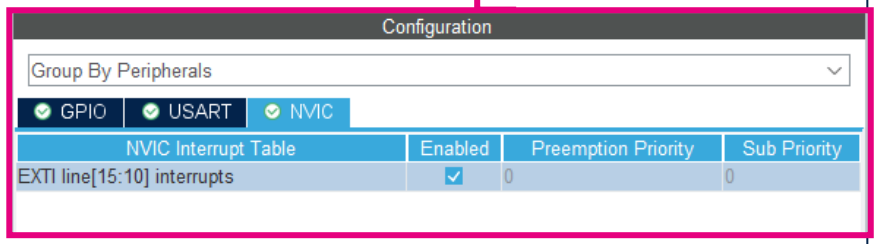
STM32CubeMX 설정

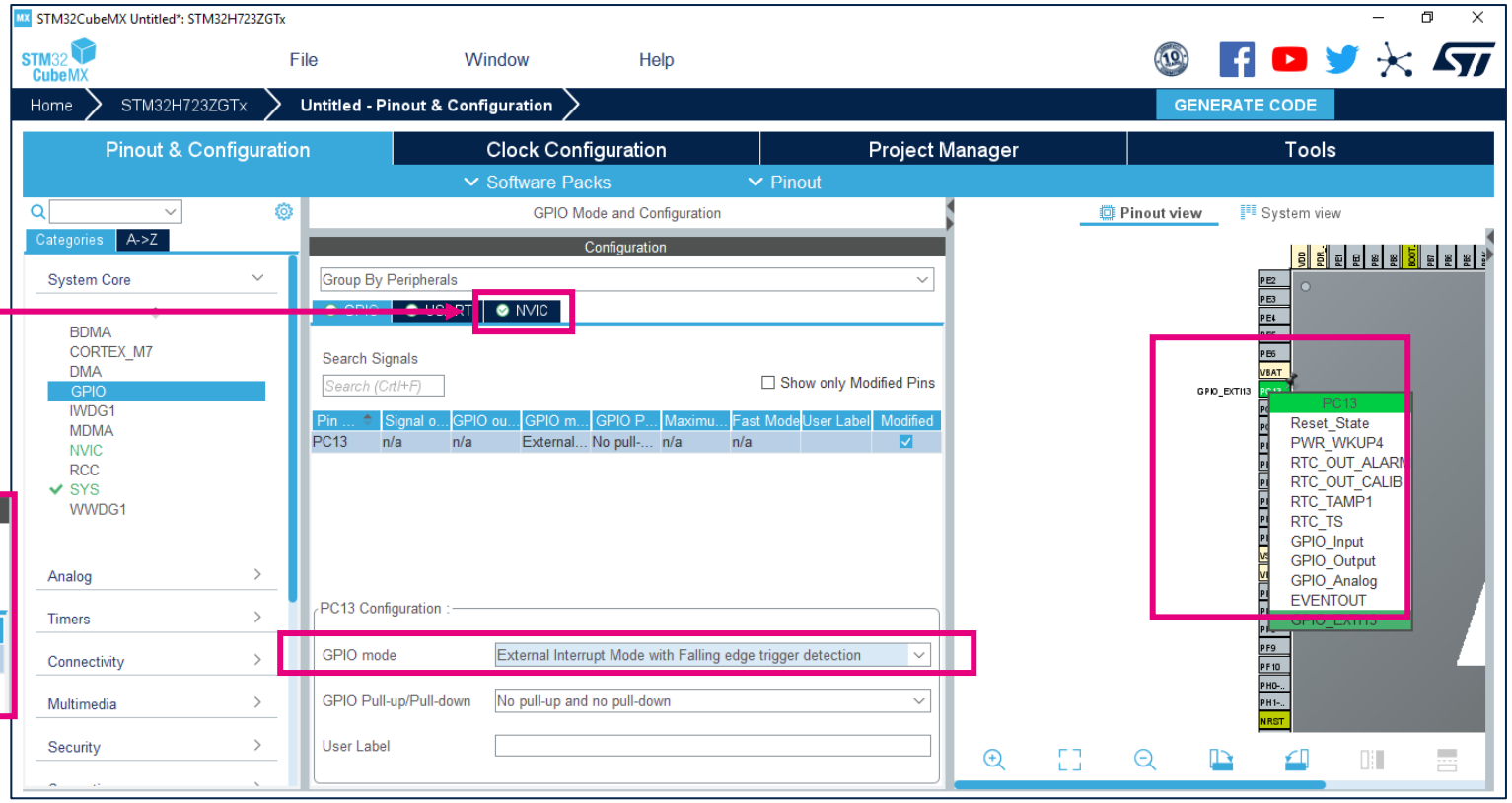
- RTOS ThreadX
  - memory pool size: 2048



STM32CubeMX 설정

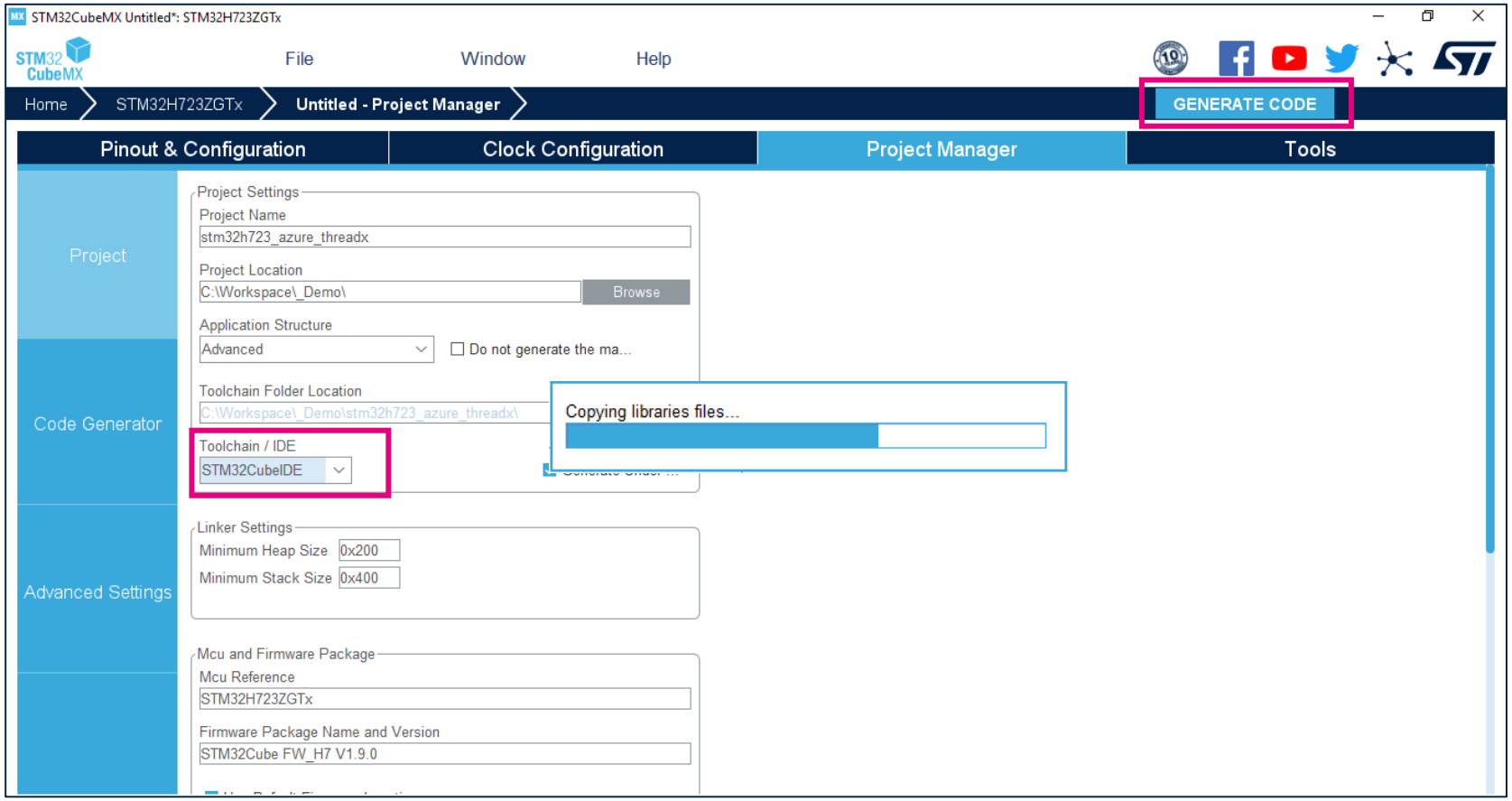
- User button 설정
  - PC13: **GPIO_EXTI13**
  - GPIO mode:
    Falling edge trigger detection
  - NVIC:
    EXTI line[15:10] interrupts





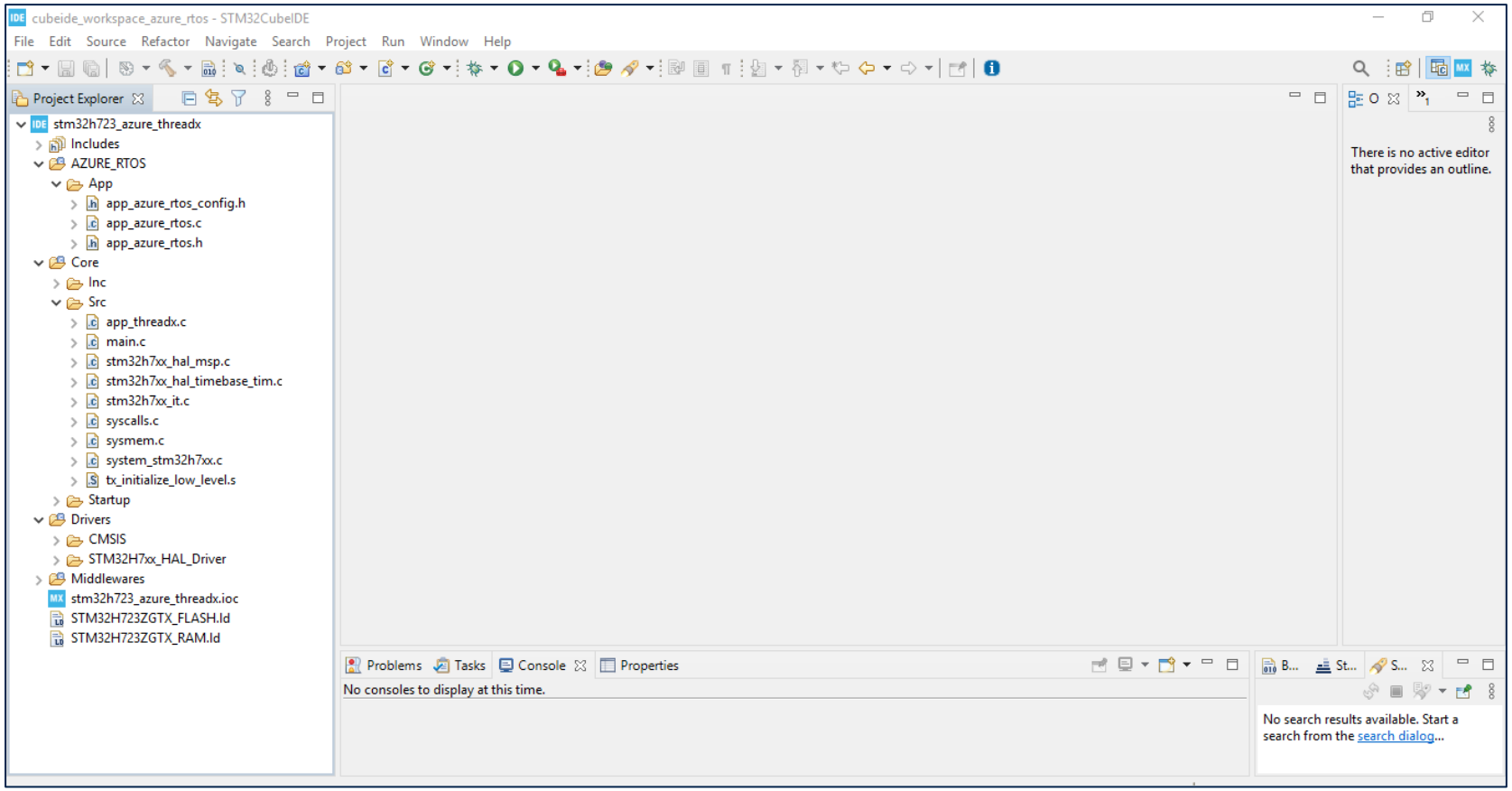
STM32CubeMX 설정

- Toolchain / IDE
  - **STM32CubeIDE**
- GENERATE CODE



STM32CubeIDE 설정

- AZURE_RTOS
- Core
- Drivers
- Middlewares



USART printf 함수 구현

- USART3을 이용하여 printf() 함수 Overring
- Serial console 확인 가능

```
int _write(int32_t file, uint8_t *ptr, int32_t len)
{
  if(HAL_UART_Transmit(&huart3, ptr, len, len) == HAL_OK )
  {
    return len;
  }
  else
  {
    return 0;
  }
}
```

life.augmented

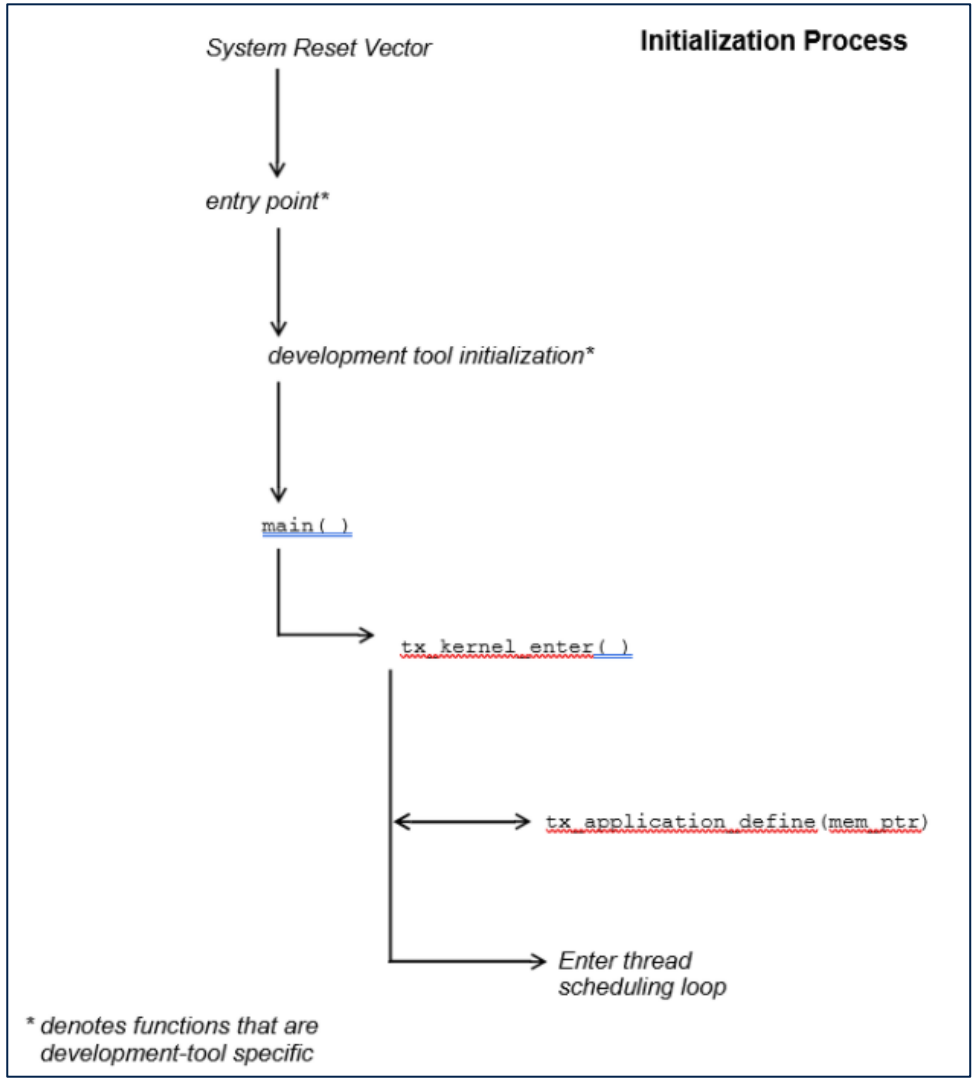
Main.c

- MX_AZURE_RTOS_Init
  - Call tx_kernel_enter()

```
int main(void)
{
  /* USER CODE BEGIN 1 */

  /* USER CODE END 1 */

  /* MCU Configuration--------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

  /* Configure the system clock */
  SystemClock_Config();

  /* USER CODE BEGIN SysInit */

  /* USER CODE END SysInit */

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_USART3_UART_Init();
  MX_AZURE_RTOS_Init();
  /* USER CODE BEGIN 2 */
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
```

life.augmented

Azure RTOS Workflow



이미지 출처: ThreadX User guide - https://docs.microsoft.com/en-us/azure/rtos/threadx/chapter3

Creation Memory Pool

- app_azure_rtos.c 에 자동 생성됨

```
#if (USE_MEMORY_POOL_ALLOCATION == 1)
/* USER CODE BEGIN TX_Pool_Buffer */
/* USER CODE END TX_Pool_Buffer */
static UCHAR tx_byte_pool_buffer[TX_APP_MEM_POOL_SIZE];
static TX_BYTE_POOL tx_app_byte_pool;

#endif
```

```
VOID tx_application_define(VOID *first_unused_memory)
{
    /* USER CODE BEGIN  tx_application_define_1*/

    /* USER CODE END  tx_application_define_1 */
#if (USE_MEMORY_POOL_ALLOCATION == 1)
  VOID *memory_ptr;

  if (tx_byte_pool_create(&tx_app_byte_pool, "Tx App memory pool", tx_byte_pool_buffer, TX_APP_MEM_POOL_SIZE) != TX_SUCCESS)
  {
    /* USER CODE BEGIN TX_Byte_Pool_Error */

    /* USER CODE END TX_Byte_Pool_Error */
  }
  else
  {
    /* USER CODE BEGIN TX_Byte_Pool_Success */

    /* USER CODE END TX_Byte_Pool_Success */

    memory_ptr = (VOID *)&tx_app_byte_pool;

    if (App_ThreadX_Init(memory_ptr) != TX_SUCCESS)
    {
        /* USER CODE BEGIN  App_ThreadX_Init_Error */

        /* USER CODE END  App_ThreadX_Init_Error */
    }
```

life.augmented

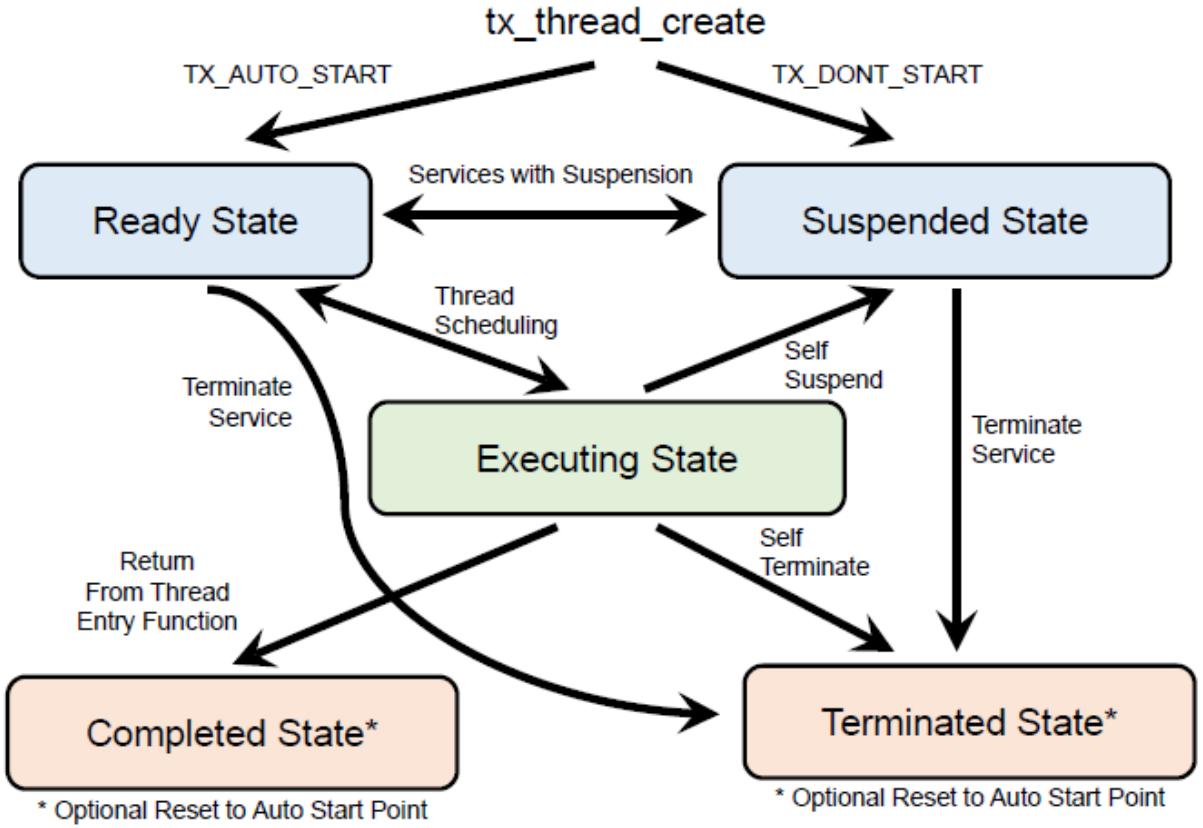
Memory Management

- 두개의 메모리 할당 방식

Memory Byte Pool	Memory Block Pool
• 표준 C 힙(malloc)과 비슷함 • 유연한 메모리 할당 제공 • 메모리 단편화에 따른 성능 저하 • 메모리 병합을 위한 단편화 모음 필요	• 고정 크기 블록으로 구성 • 메모리 단편화에 따른 성능 저하가 없음 • 블록 할당 및 해제 시간이 짧음 • 유연성으로 인한 메모리 낭비

Thread Feature

- Thread State Transition



이미지 출처: ThreadX User guide - https://docs.microsoft.com/en-us/azure/rtos/threadx/chapter3

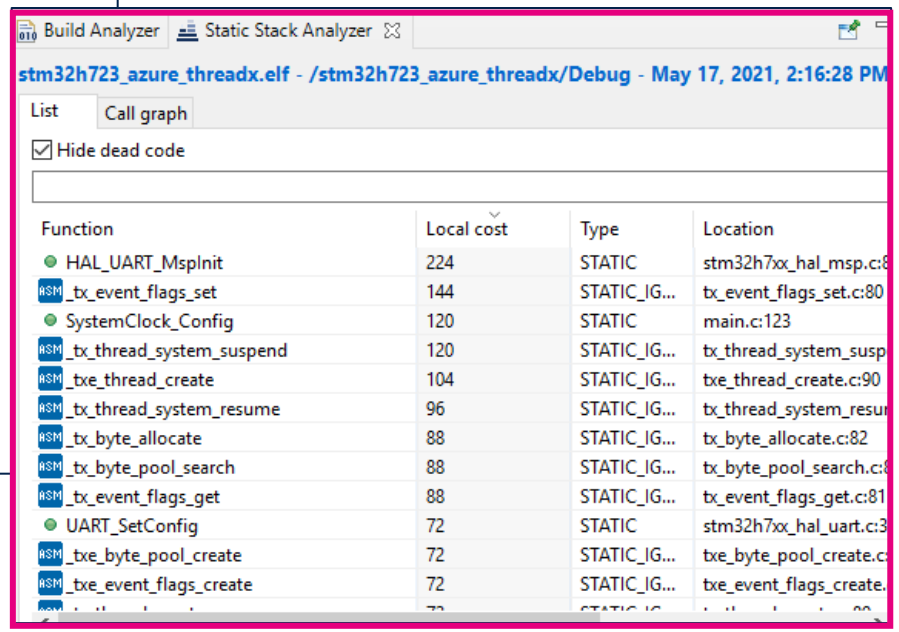
Thread 생성 파라메타

- tx_thread_create Parameters
  - thread control block
  - thread name
  - thread entry function name
  - user input
  - stack pointer
  - stack size
  - priority
  - preemption-threshold
  - time slice option
  - auto start option

Thread 생성

- app_threadx.c
  - App_ThreadX_Init 함수에 Thread 생성 코드 추가

```
UINT App_ThreadX_Init(VOID *memory_ptr)
{
  UINT ret = TX_SUCCESS;
  TX_BYTE_POOL *byte_pool = (TX_BYTE_POOL*)memory_ptr;

  /* USER CODE BEGIN App_ThreadX_Init */
  CHAR *pointer;

  if (tx_byte_allocate(byte_pool, (VOID **) &pointer,
                       APP_STACK_SIZE, TX_NO_WAIT) != TX_SUCCESS)
  {
    ret = TX_POOL_ERROR;
  }

  /* Create MainThread.  */
  if (tx_thread_create(&MainThread, "Main Thread", MainThread_Entry, 0,
                       pointer, APP_STACK_SIZE,
                       MAIN_THREAD_PRIO, MAIN_THREAD_PREEMPTION_THRESHOLD,
                       TX_NO_TIME_SLICE, TX_AUTO_START) != TX_SUCCESS)
  {
    ret = TX_THREAD_ERROR;
  }
```



Thread implements

- MainThread

```
void MainThread_Entry(ULONG thread_input)
{
  printf("This is Main Thread Entry\r\n");
  while(1)
  {
    printf("[%d] Main Thread is working...\r\n", (int)tx_time_get());
    tx_thread_sleep(100);
  }
}
```

- FirstThread

```
void FirstThread_Entry(ULONG thread_input)
{
  printf("This is 1st Thread Entry\r\n");
  while(1)
  {
    printf("[%d] 1st Thread is working...\r\n", (int)tx_time_get());
    tx_thread_sleep(200);
  }
}
```

Thread implements

- SecondThread
  - tx_event_flags_get

```
void SecondThread_Entry(ULONG thread_input)
{
  ULONG actual_flags = 0;

  printf("This is 2nd Thread Entry\r\n");

  while(1)
  {
    if (tx_event_flags_get(&EventFlag, THREAD_EXTI_EVT, TX_OR_CLEAR,
                           &actual_flags, TX_WAIT_FOREVER) != TX_SUCCESS)
    {
      Error_Handler();
    }
    else
    {
      printf("\r\n[%d]2nd Thread alarms the User button was pressed!!\r\n\r\n", (int)tx_time_get());
    }
  }
}
```

life.augmented

User button 처리

- User button (PC13) EXTI handler
  - stm32h7xx_it.c 에 아래 코드 확인

```
void EXTI15_10_IRQHandler(void)
{
  /* USER CODE BEGIN EXTI15_10_IRQn 0 */

  /* USER CODE END EXTI15_10_IRQn 0 */
  HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_13);
  /* USER CODE BEGIN EXTI15_10_IRQn 1 */

  /* USER CODE END EXTI15_10_IRQn 1 */
}
```

- HAL_GPIO_EXTI_Callbak 구현

```
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
  if(GPIO_Pin == GPIO_PIN_13)
  {
    if (tx_event_flags_set(&EventFlag, THREAD_EXTI_EVT, TX_OR) != TX_SUCCESS)
    {
      Error_Handler();
    }
  }
}
```

life.augmented

Demo

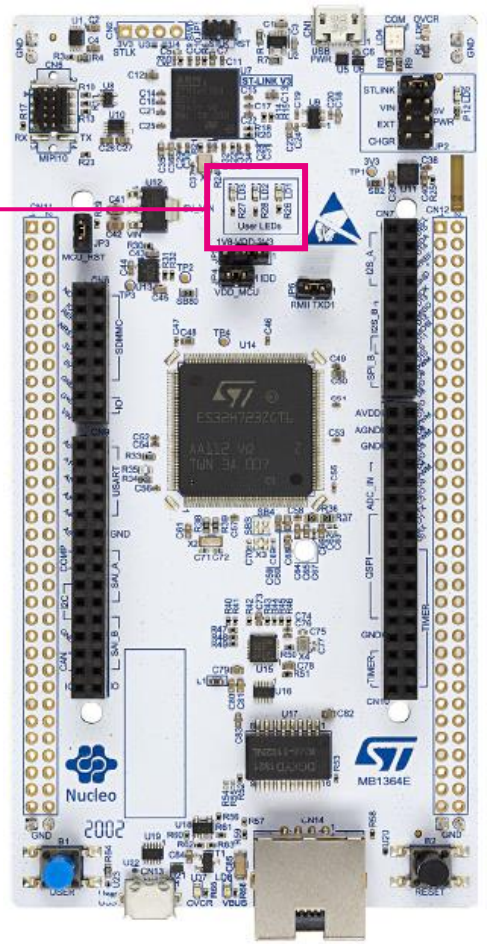
- Serial console 로 Logging 확인
- User button 을 눌러 Key event 처리
- LED 동작 확인

LED Indicator
- LED 1: Main Thread
- LED 2: First Thread
- LED 3: Key Event

COM56 - Tera Term VT

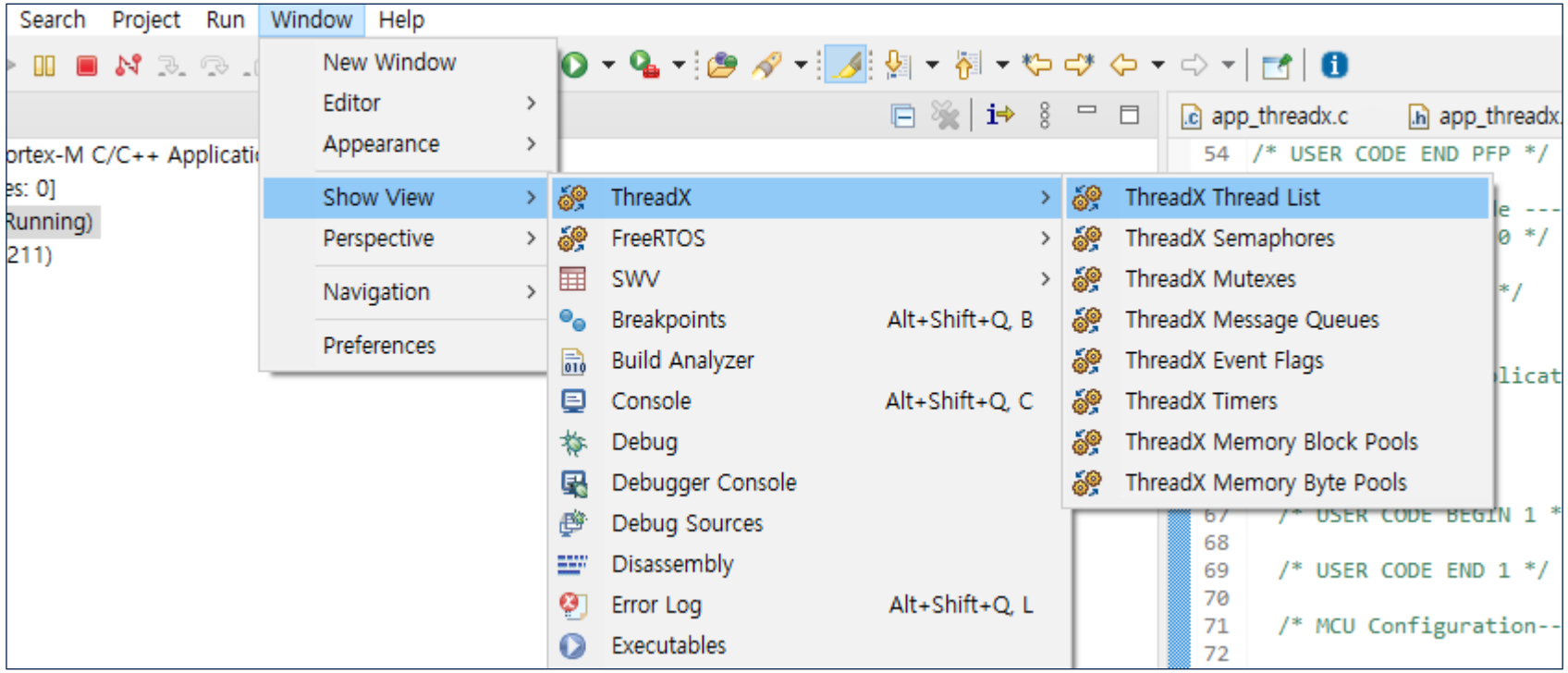
File Edit Setup Control Window Help

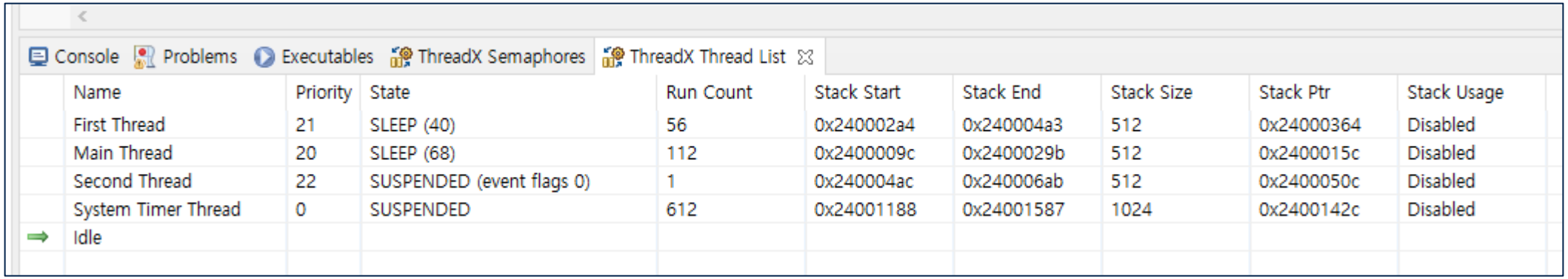
```
[448200] Main Thread is working...
[448200] 1st Thread is working...
[448300] Main Thread is working...
[448400] Main Thread is working...
[448400] 1st Thread is working...
[448500] Main Thread is working...
[448600] Main Thread is working...
[448600] 1st Thread is working...
[448700] Main Thread is working...
[448800] Main Thread is working...
[448800] 1st Thread is working...
[448900] Main Thread is working...
[449000] Main Thread is working...
[449000] 1st Thread is working...
[449100] Main Thread is working...
[449200] Main Thread is working...
[449200] 1st Thread is working...
[449300] Main Thread is working...
[449400] Main Thread is working...
[449400] 1st Thread is working...
[449500] Main Thread is working...
[449600] Main Thread is working...
[449600] 1st Thread is working...

[449658]2nd Thread alarms the User button was pressed!!

[449700] Main Thread is working...
[449800] Main Thread is working...
[449800] 1st Thread is working...
[449900] Main Thread is working...
[450000] Main Thread is working...
```



life.augmented

STM32CubeIDE ThreadX Debug Tools



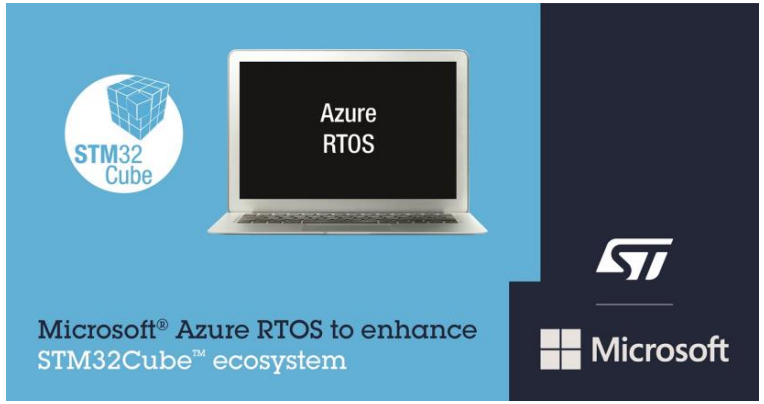


Name	Priority	State	Run Count	Stack Start	Stack End	Stack Size	Stack Ptr	Stack Usage
First Thread	21	SLEEP (40)	56	0x240002a4	0x240004a3	512	0x24000364	Disabled
Main Thread	20	SLEEP (68)	112	0x2400009c	0x2400029b	512	0x2400015c	Disabled
Second Thread	22	SUSPENDED (event flags 0)	1	0x240004ac	0x240006ab	512	0x2400050c	Disabled
System Timer Thread	0	SUSPENDED	612	0x24001188	0x24001587	1024	0x2400142c	Disabled
Idle								

Resources

Resources

- STM32Cube Ecosystem: https://www.st.com/en/ecosystems/stm32cube.html
- X-Cube-AZRTOS-H7: https://www.st.com/en/embedded-software/x-cube-azrtos-h7.html
- X-Cube-AZRTOS-H7 Github: https://github.com/STMicroelectronics/x-cube-azrtos-h7
- STM32 MCU Wiki: https://wiki.st.com/stm32mcu/wiki/Main_Page
- Azure RTOS Github: https://github.com/azure-rtos
- Azure RTOS ThreadX 설명서: https://docs.microsoft.com/ko-kr/azure/rtos/threadx/



Thank you

life.augmented