



life.augmented

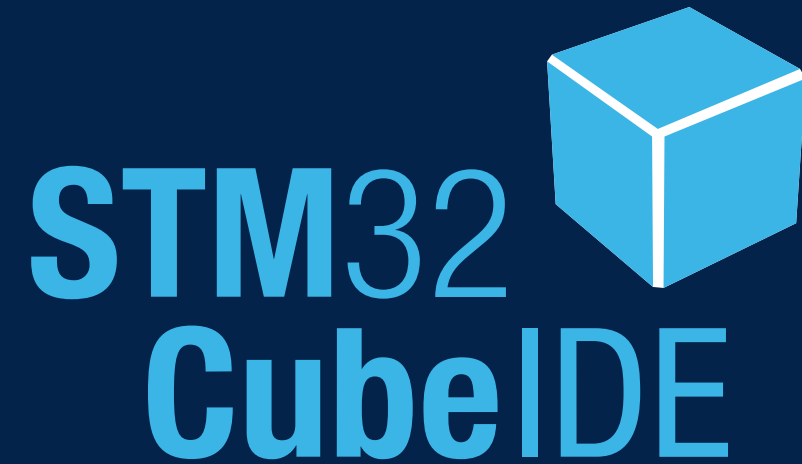
Innovations in debugging tools

Krishan Solanki

GPM Staff Engineer

3rd Aug 2023

STM32CubeIDE

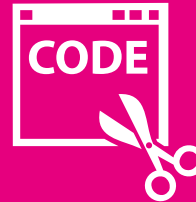


One tool for all your STM32 development

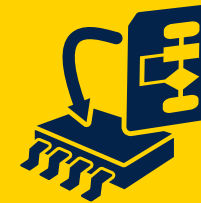
Chipset / Board
Configuration



Code
Development

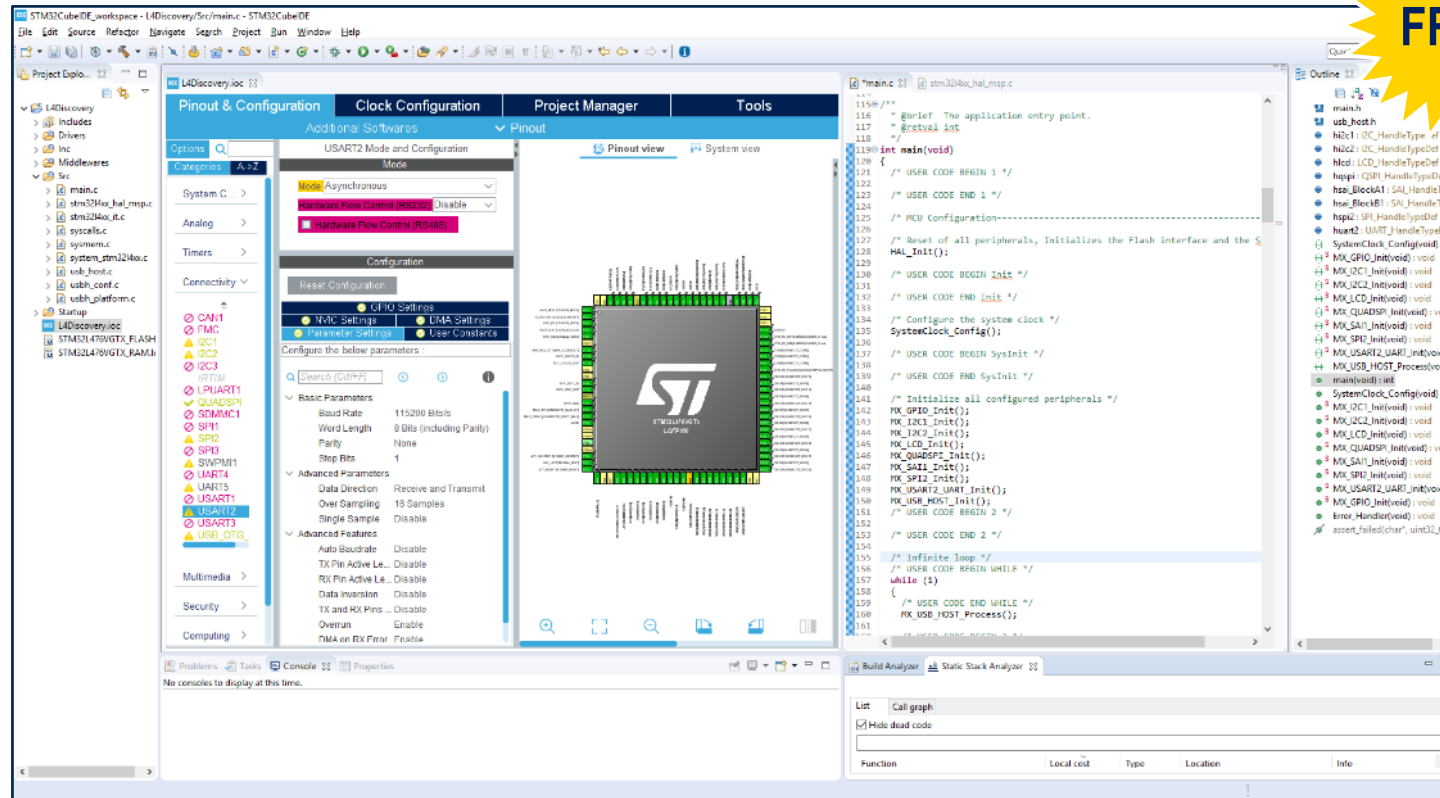


Validation
Debug



Free multi-platform development tool

FREE



macOS

Eclipse/GCC Based

Free for Commercial Development

Multi-OS Support

Create a STM32CubeIDE project for GPIO toggle with Project Name : FAE_name

OR

Open below project

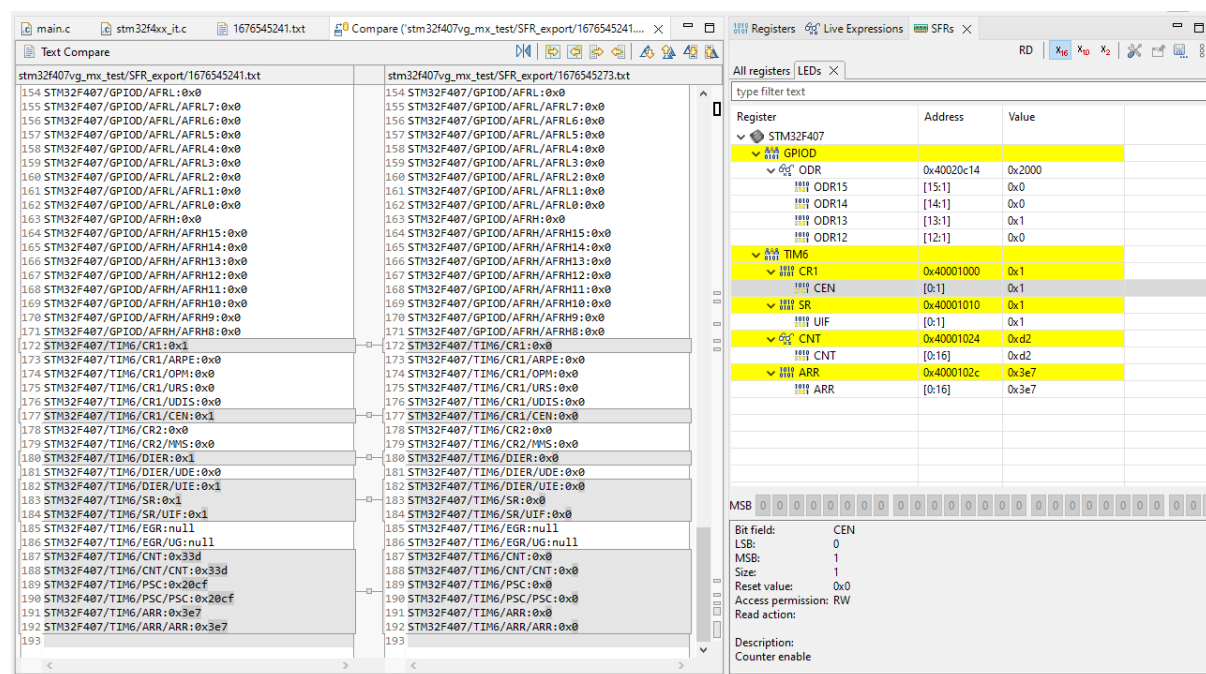
Repository\STM32Cube_FW_U5_V1.2.0\Projects\NUCLEO-U575ZI-Q\Examples\GPIO\GPIO_EXTI

- Add the GPIOC->ODR and IDR register in the favorite list
- Give the list name as “CompanyName_ FaeName”

New

SFR-view – 3 use cases

- Favorite lists
 - Focus on a subset of peripherals and registers!
- Live update
 - Live channel provides ~5Hz live update
- Log export
 - Troubleshoot your peripheral configuration by comparing log-files



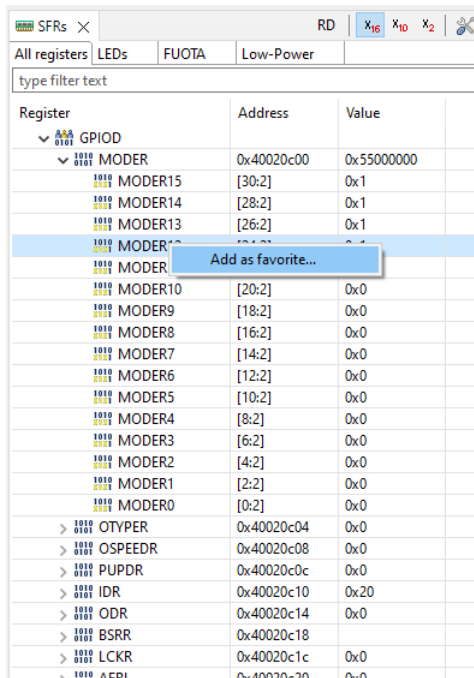
Available in STM32CubeIDE 1.11.x or later

New

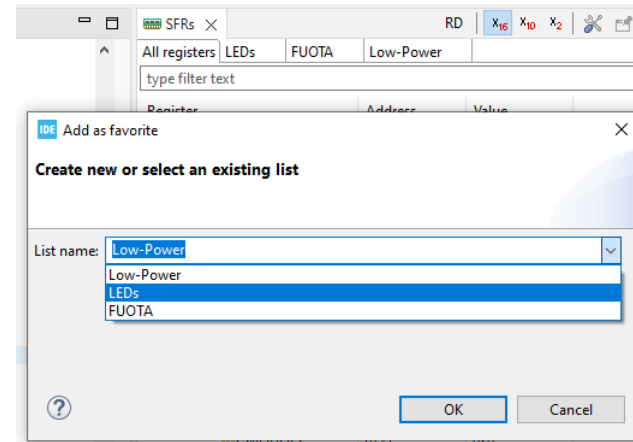
Favorite list(s)

1. Right-click on a node and click *Add as favorite...*

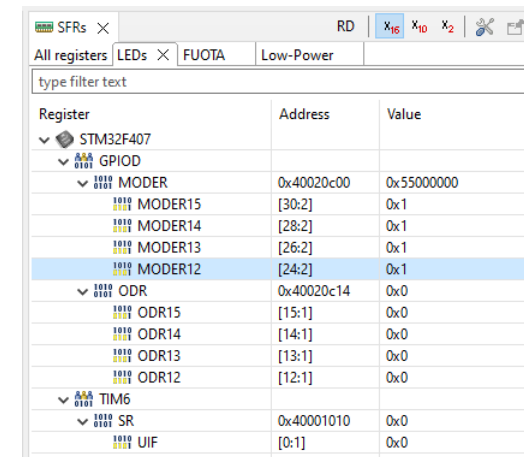
- Peripheral
- Register
- Single bit.



2. Select a *List name* or type in the name of a new *List*



3. Display only your favorites



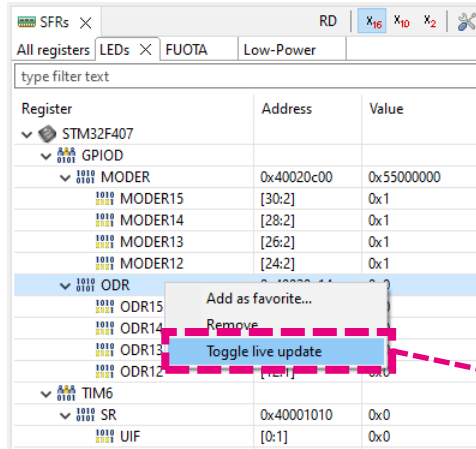
New

Live updates

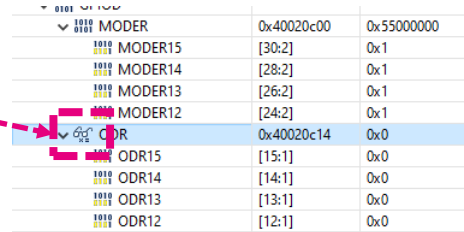
1. Right-click on register-level → *Toggle live update*

- *Only applicable at register-level since debugger always reads 32-bits.*

2. The “google” icon indicates that live-update is enabled

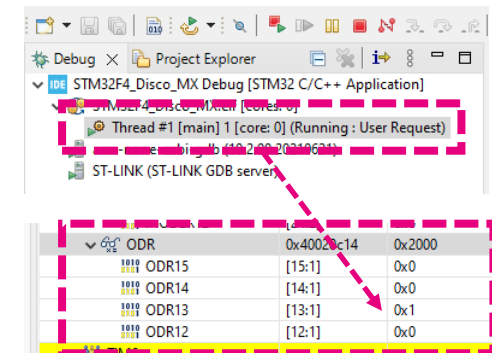


Register	Address	Value
STM32F407		
GPIO		
MODER	0x40020c00	0x55000000
MODER15	[30:2]	0x1
MODER14	[28:2]	0x1
MODER13	[26:2]	0x1
MODER12	[24:2]	0x1
ODR	0x40020c14	0x0
ODR15	[15:1]	0x0
ODR14	[14:1]	0x0
ODR13	[13:1]	0x0
ODR12	[12:1]	0x0
TIM6		
SR	0x40001010	0x0
UIF	[0:1]	0x0



Register	Address	Value
MODER	0x40020c00	0x55000000
MODER15	[30:2]	0x1
MODER14	[28:2]	0x1
MODER13	[26:2]	0x1
MODER12	[24:2]	0x1
ODR	0x40020c14	0x0
ODR15	[15:1]	0x0
ODR14	[14:1]	0x0
ODR13	[13:1]	0x0
ODR12	[12:1]	0x0

3. Register values are updated “live” while target is running



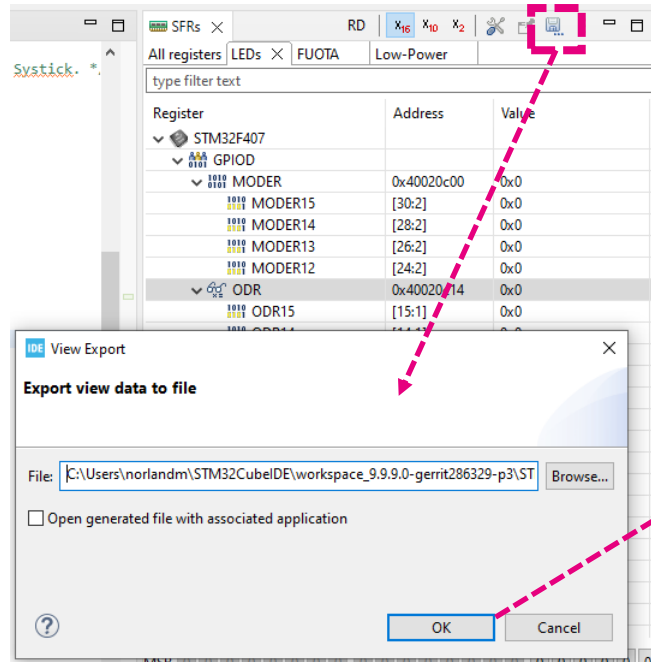
Register	Address	Value
ODR	0x40020c14	0x2000
ODR15	[15:1]	0x0
ODR14	[14:1]	0x0
ODR13	[13:1]	0x1
ODR12	[12:1]	0x0

New

Log export and compare

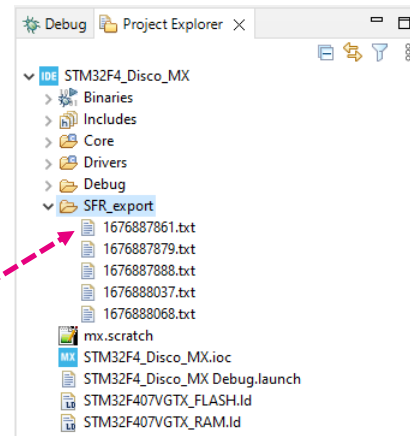
1. Select the Export scope

- All registers?
- Or just a favorite list?



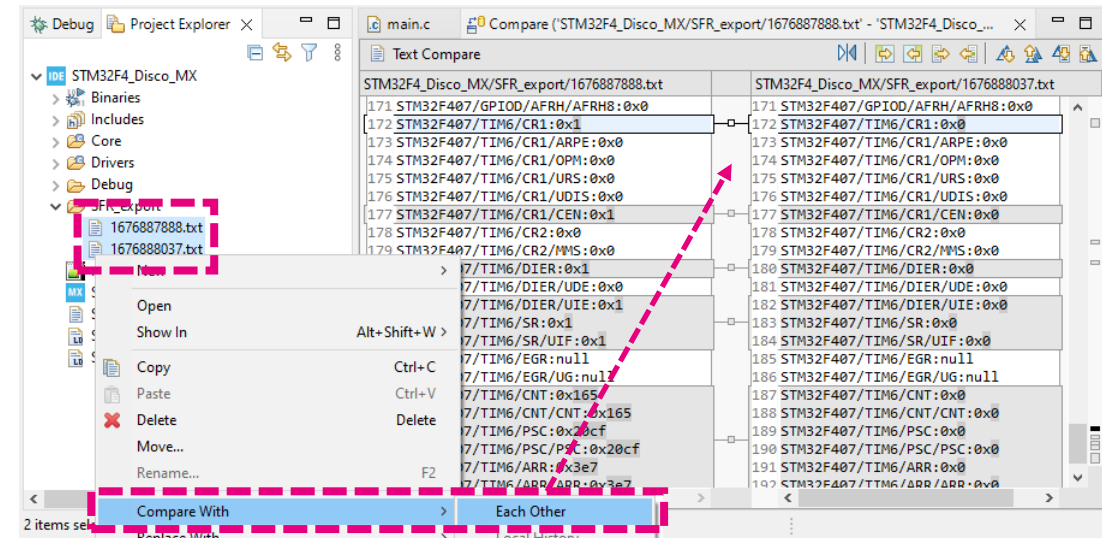
2. Logs are saved in *SFR_export*

- Named by incremental number



3. Compare two log-files

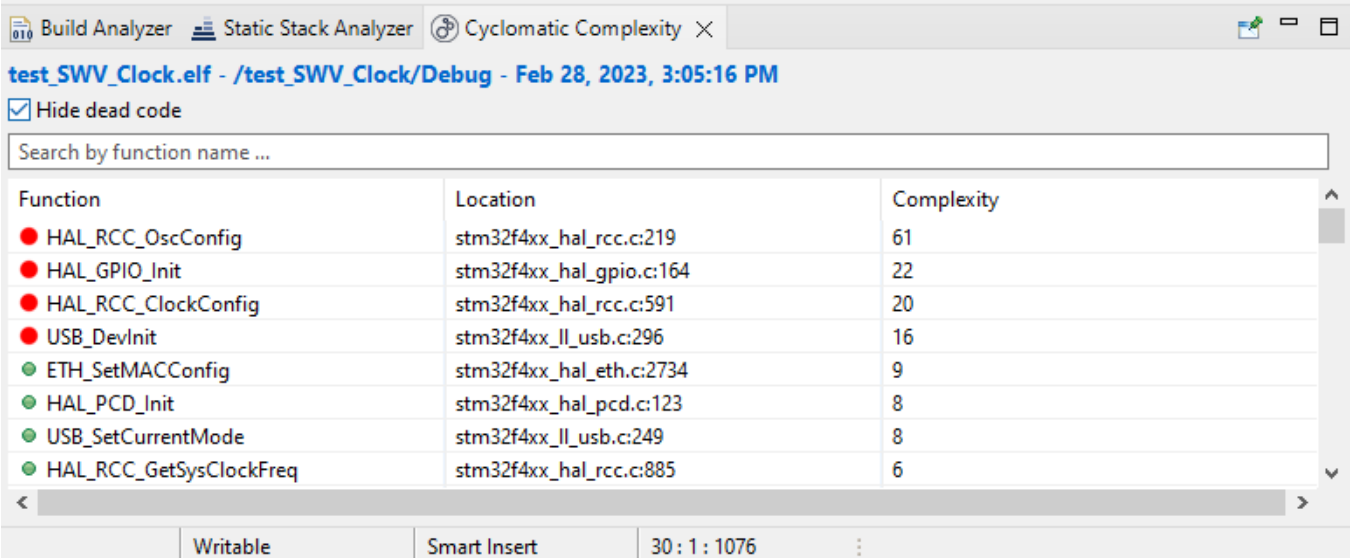
- Right-click → Compare with → each other



New

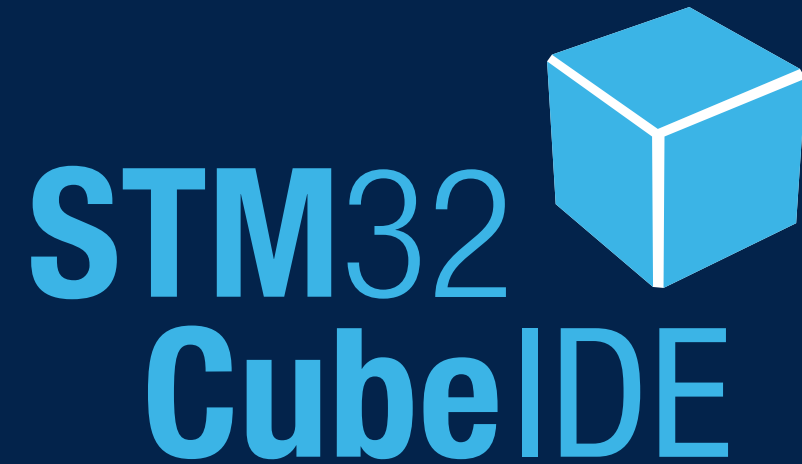
Add support for cyclomatic complexity

- Cyclomatic complexity is a measurement which give the possibility to determine the stability and level of confidence in a program.
- The new view show the level of complexity for each function.



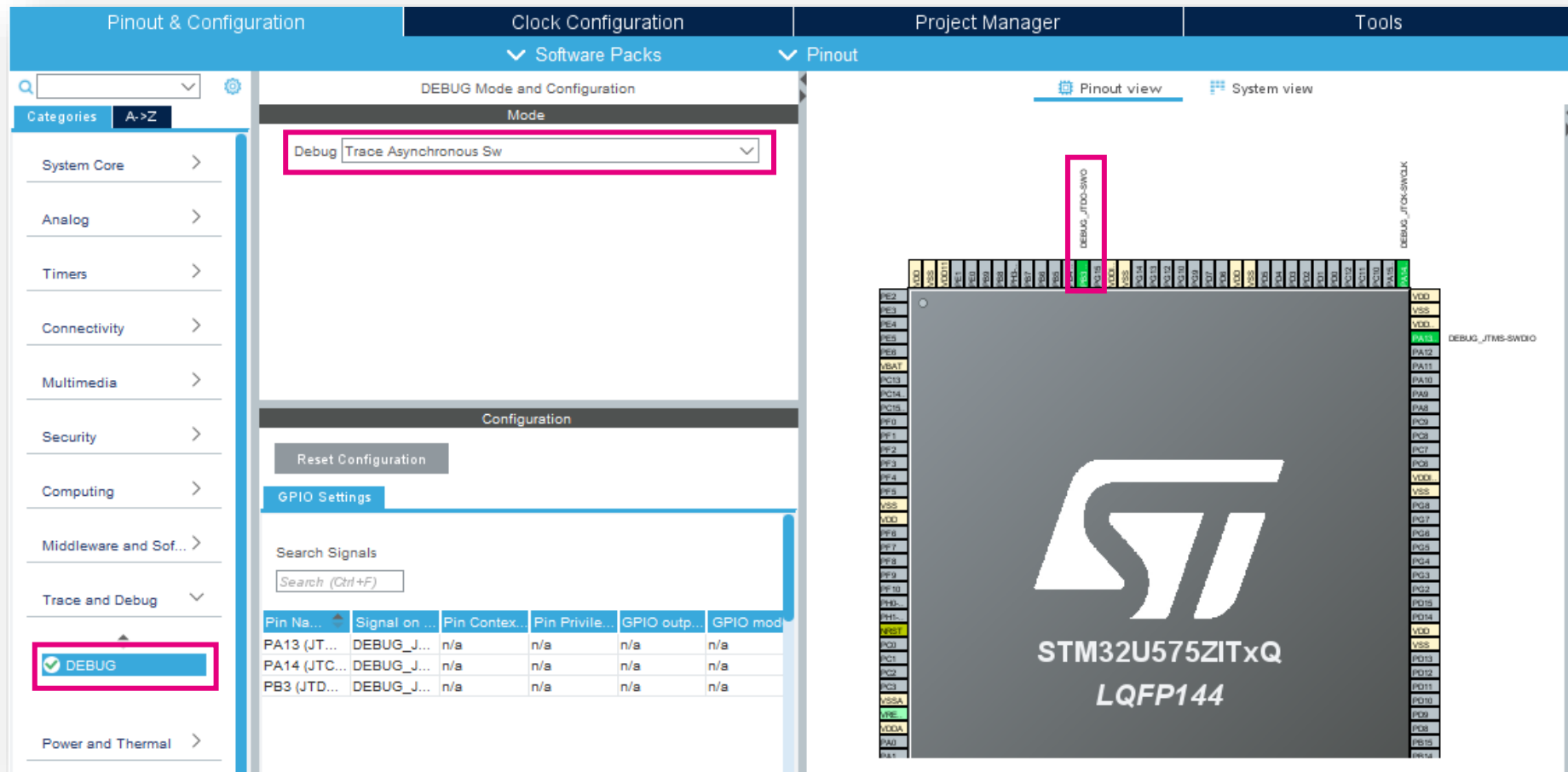
Function	Location	Complexity
● HAL_RCC_OscConfig	stm32f4xx_hal_rcc.c:219	61
● HAL_GPIO_Init	stm32f4xx_hal_gpio.c:164	22
● HAL_RCC_ClockConfig	stm32f4xx_hal_rcc.c:591	20
● USB_DevInit	stm32f4xx_ll_usb.c:296	16
● ETH_SetMACConfig	stm32f4xx_hal_eth.c:2734	9
● HAL_PCD_Init	stm32f4xx_hal_pcd.c:123	8
● USB_SetCurrentMode	stm32f4xx_ll_usb.c:249	8
● HAL_RCC_GetSysClockFreq	stm32f4xx_hal_rcc.c:885	6

STM32CubeIDE



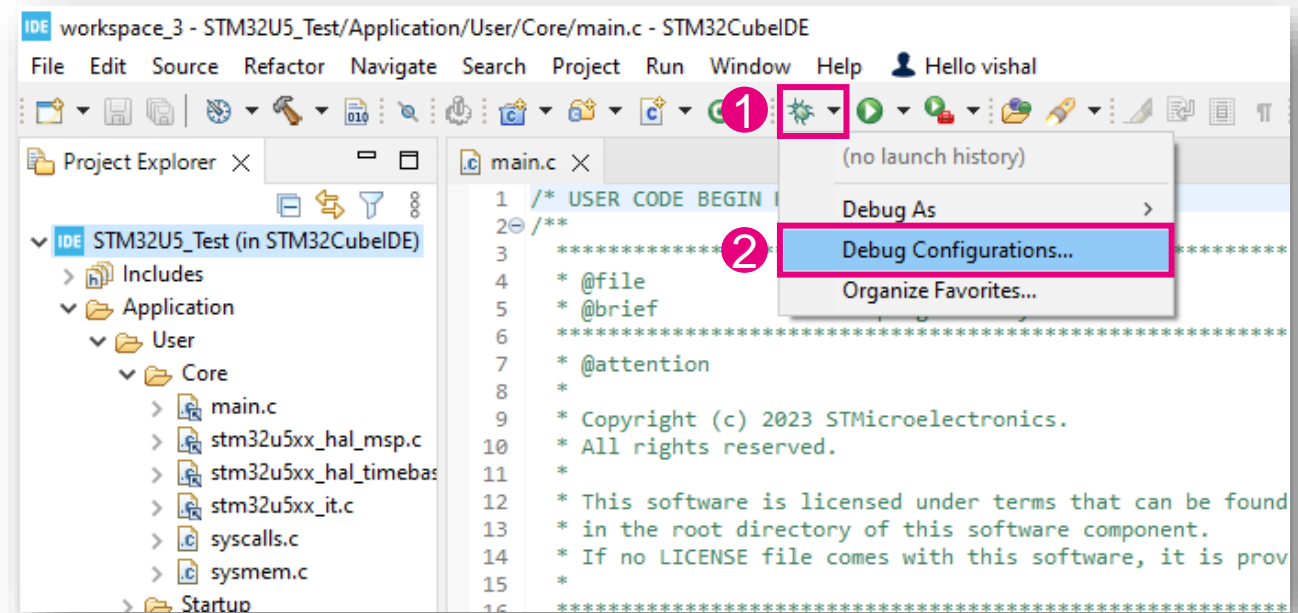
Adding SWO pin

- Enable SWO via the STM32CubeMC perspective and re-generate the code.



Enable SWV in the debugger settings(1/2)

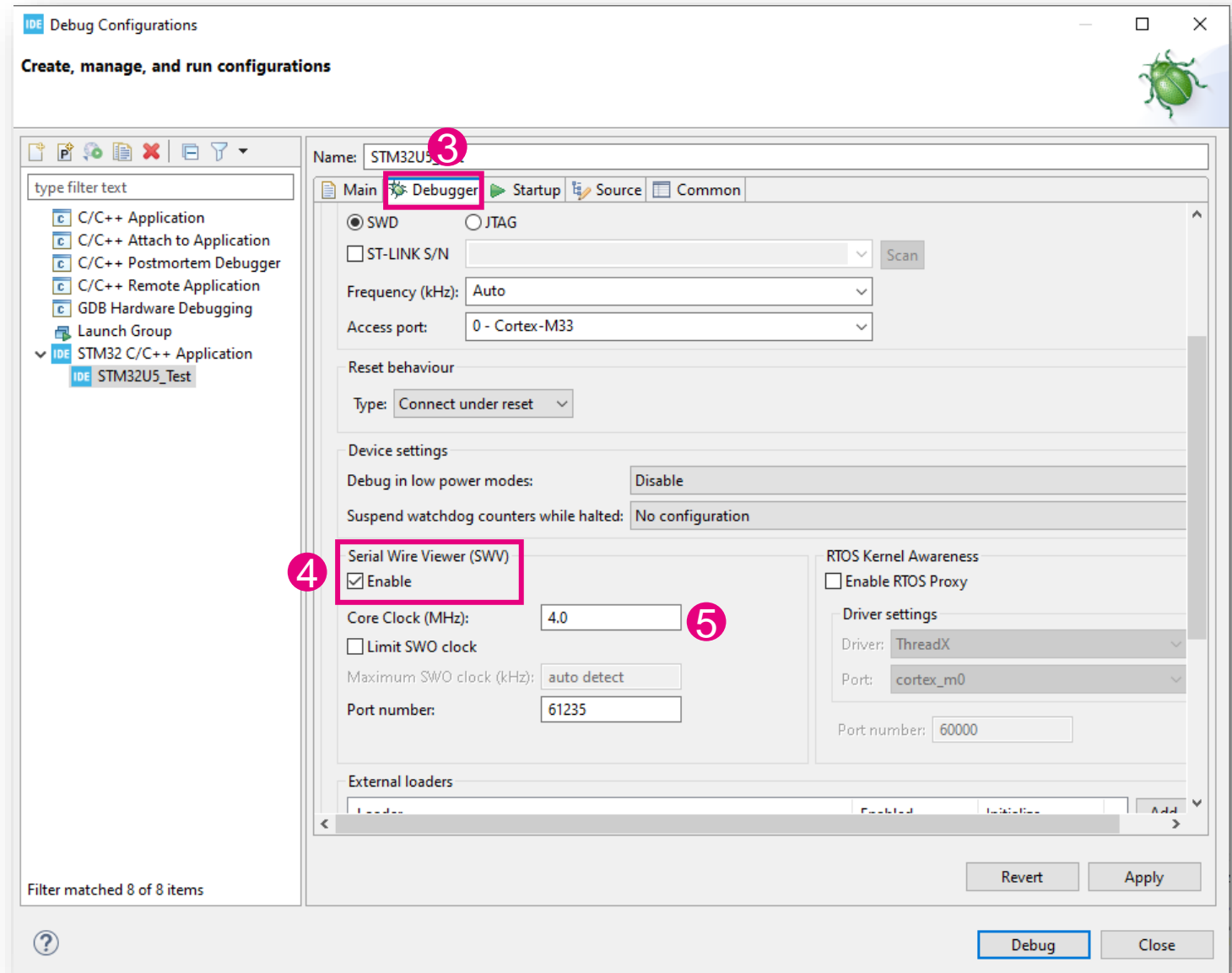
- Steps:
- Enable SWV Ftr:
 - Step 1: Go to Debug
 - Step 2: “Debug Configurations”



Enable SWV in the debugger settings (2/2)

Steps contd:

- **Step 3:** Go to “Debugger” tab
- **Step 4:** Enable SWV
- **Step 5:** SWV tracing requires manually setting the **Core Clock** to the correct value, **4 MHz** in our case
- Click “Apply” -> “Close”



SWV: features explored

- Using Data Trace & Live Watch
- Using Data Trace Timeline Graph
- printf() redirection
- Timing measurement
- Exception Trace Log & Timeline Graph



life.augmented

STM32CubeIDE

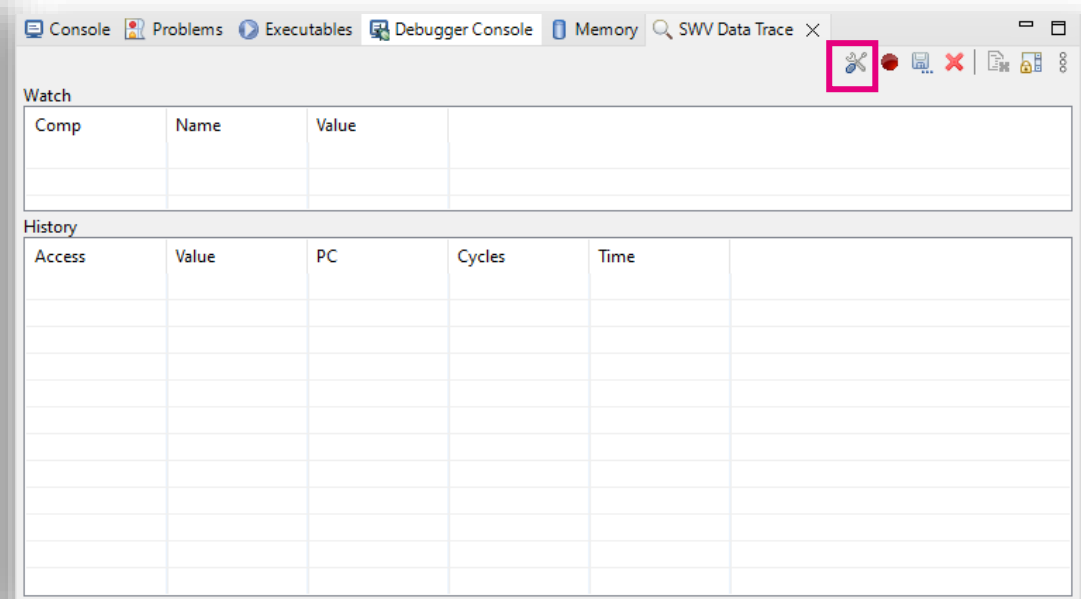
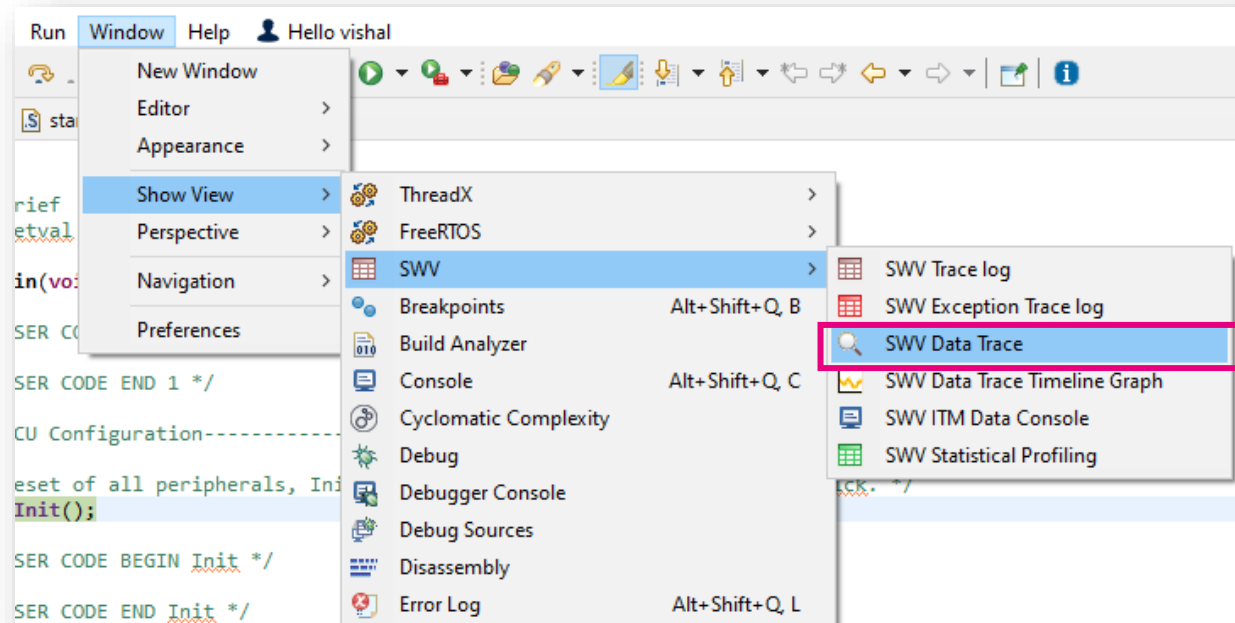
SWV: Data Trace & Live Watch

SWV: using Data Trace

- Data Trace:
 - To watch variables live
 - No need to stop on a breakpoint OR Halt execution
 - Monitor the memory reads and writes of particular variables
 - Can be used to detect unexpected RAM access.

Enable SWV Data Trace

- Start Debug session
- Go to **Window -> Show View -> SWV** and click **SWV Data Trace**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



Serial Wire Viewer settings

- **Step 1:** Enable “Timestamps”
- **Step 2:** Enable **Trace Comparator 0**
- **Step 3:** Add the variable name/address to be traced (LoopCounter)
- **Step 4:** Select “Write” in the Access list (only monitor Write access)
- **Step 5:** Select “Data Value” to be logged, click OK
- **Note:** Keep other trace events disabled to avoid chances of overflow in the SWO line

Serial Wire Viewer settings for STM32U5_Test

Clock Settings
Core Clock: 4 MHz
Clock Prescaler: 1
SWO Clock: 4000.0 kHz

Trace Events
☐ CPI: Cycles per instruction ☐ EXC: Exception overhead
☐ SLEEP: Sleep cycles ☐ LSU: Load store unit cycles
☐ FOLD: Folded instructions ☐ EXETRC: Trace Exceptions

PC Sampling
☐ Enable Resolution: 16384 Cycles/sample

Timestamps
☒ Enable Prescaler: 1

Data Trace

Comparator 0
☒ Enable
Var/Addr: LoopCounter
Access: Write
Size: Word
Generate: Data Value

Comparator 1
☐ Enable
Var/Addr: 0x0
Access: Read/Write
Size: Word
Generate: Data Value

Comparator 2
☐ Enable
Var/Addr: 0x0
Access: Read/Write
Size: Word
Generate: Data Value

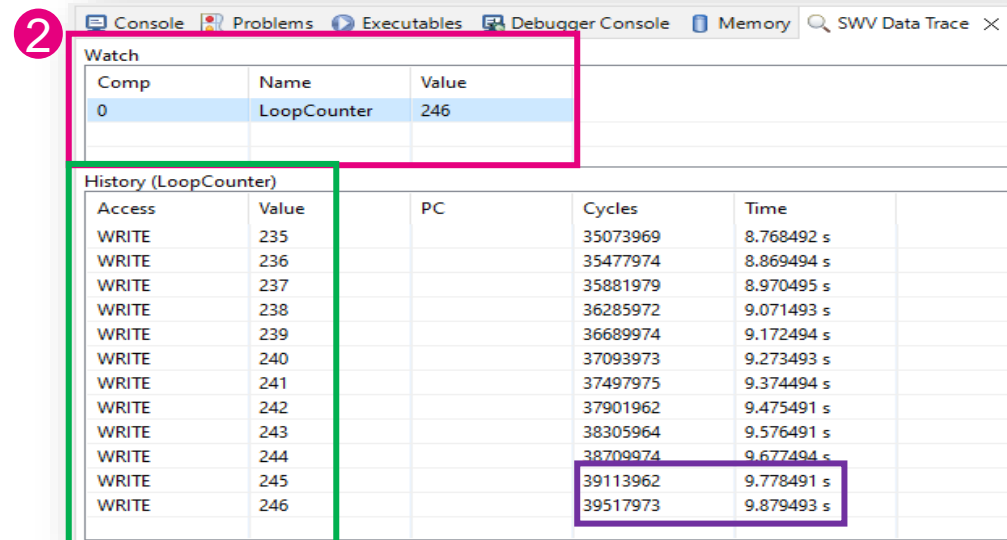
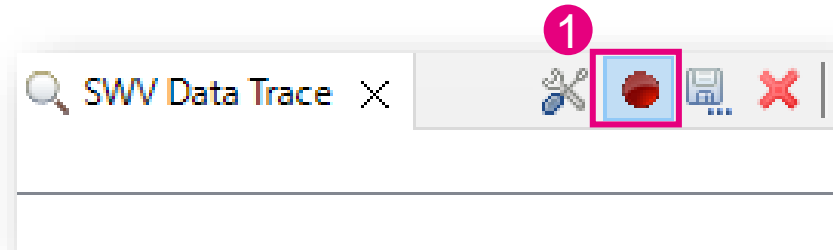
Comparator 3
☐ Enable
Var/Addr: 0x0
Access: Read/Write
Size: Word
Generate: Data Value

ITM Stimulus Ports
Enable port: 31 ☐ 24 ☐ 23 ☐ 16 ☐ 15 ☐ 8 ☐ 7 ☐ 0 ☐
Privileged only ports: ☐ Port 31..24 ☐ Port 23..16 ☐ Port 15..8 ☐ Port 7..0

OK Cancel

SWV: Using Data Trace

- **Step 1:** Click on “Start Trace Button”
- Start Execution
- **Step 2:** Observe the Data live on the “SWV Data Trace Console”
 - Live Watch
 - History
 - Timestamps(~101 ms delay between each **Write** in our Case)



Comp	Name	Value
0	LoopCounter	246

History (LoopCounter)				
Access	Value	PC	Cycles	Time
WRITE	235		35073969	8.768492 s
WRITE	236		35477974	8.869494 s
WRITE	237		35881979	8.970495 s
WRITE	238		36285972	9.071493 s
WRITE	239		36689974	9.172494 s
WRITE	240		37093973	9.273493 s
WRITE	241		37497975	9.374494 s
WRITE	242		37901962	9.475491 s
WRITE	243		38305964	9.576491 s
WRITE	244		38709974	9.677494 s
WRITE	245		39113962	9.778491 s
WRITE	246		39517973	9.879493 s



life.augmented

STM32CubeIDE

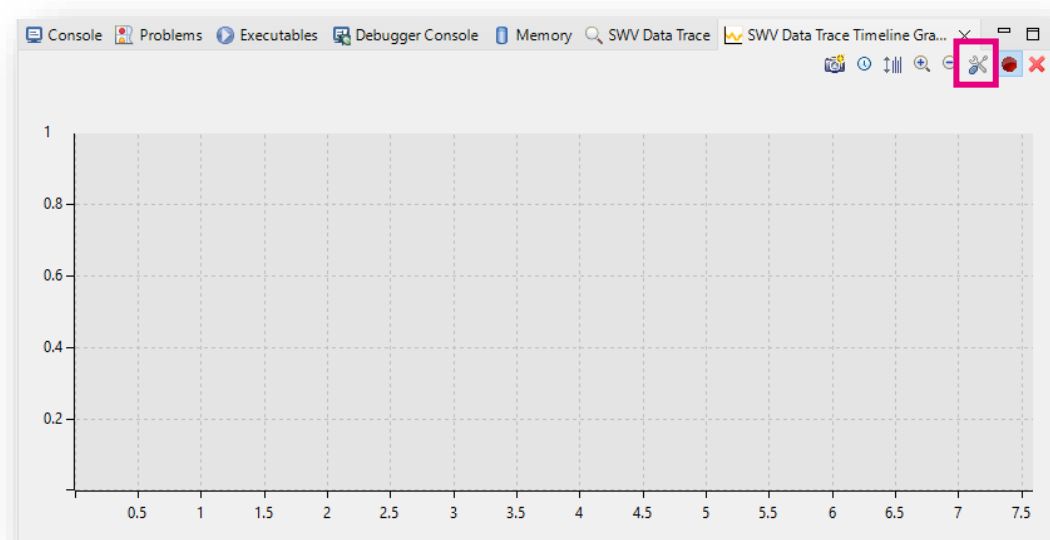
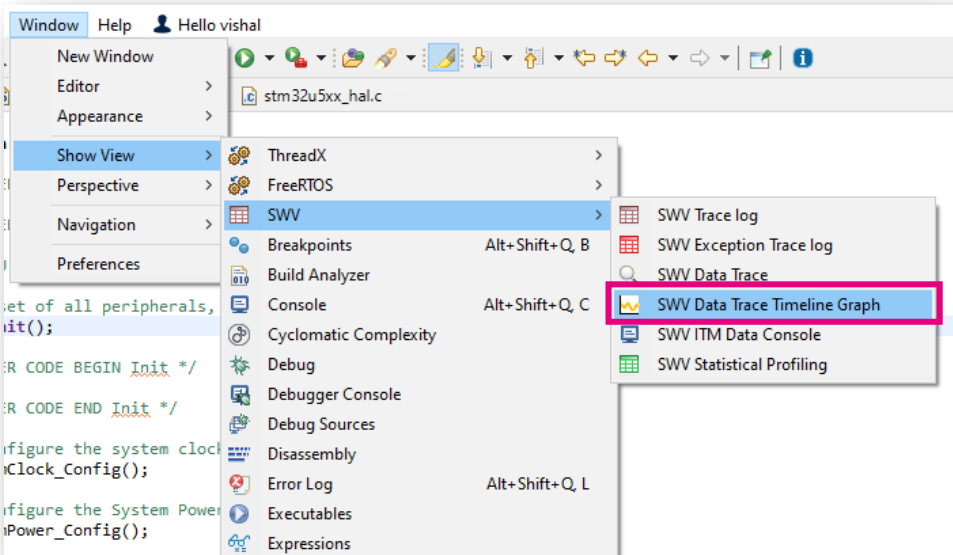
SWV: Data Trace Timeline Graph

SWV: Using Data Trace Time Graph

- Used for Graphical Data plot of variables in real-time
- An “Oscilloscope style” data plot graph
- Plots the Graph of Data Vs Time OR Data Vs Clock Cycles
- Also allows to save the graphs as images to study later

Enable SWV Data Trace Timeline Graph

- Start Debug session
- Go to **Window -> Show View -> SWV** and click **SWV Data Trace Timeline Graph**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



Configure SWV settings

- **Step 1:** In the “Data Trace” settings, add Comparator 1
- **Step 2:** add “Sensor_Data” and also set the other fields of Access, Size, Data Value as shown here.

```
/* USER CODE BEGIN WHILE */
while (1)
{
    LoopCounter++;
    HAL_Delay(500);

    if(LoopCounter>100)
        LoopCounter=0;

    //Used to simulate general data pattern, to be observed in Trace Logs.
    if(LoopCounter<25)
    {
        Sensor_Data = Sensor_Data + 5;
    }
    else
    {
        Sensor_Data = Sensor_Data -1000;
    }
}

/* USER CODE END WHILE */
```

Serial Wire Viewer settings for STM32U5_Test

Clock Settings
Core Clock: 4 MHz
Clock Prescaler: 1
SWO Clock: 4000.0 kHz

Trace Events
☐ CPI: Cycles per instruction ☐ EXC: Exception overhead
☐ SLEEP: Sleep cycles ☐ LSU: Load store unit cycles
☐ FOLD: Folded instructions ☐ EXETRC: Trace Exceptions

PC Sampling
☐ Enable Resolution: 16384 Cycles/sample

Timestamps
☒ Enable Prescaler: 1

Data Trace

Comparator 0
☒ Enable
Var/Addr: LoopCounter
Access: Write
Size: Word
Generate: Data Value

Comparator 1 ①
☒ Enable
Var/Addr: Sensor_Data
Access: Write ②
Size: Word
Generate: Data Value

Comparator 2
☐ Enable
Var/Addr: 0x0
Access: Read/Write
Size: Word
Generate: Data Value

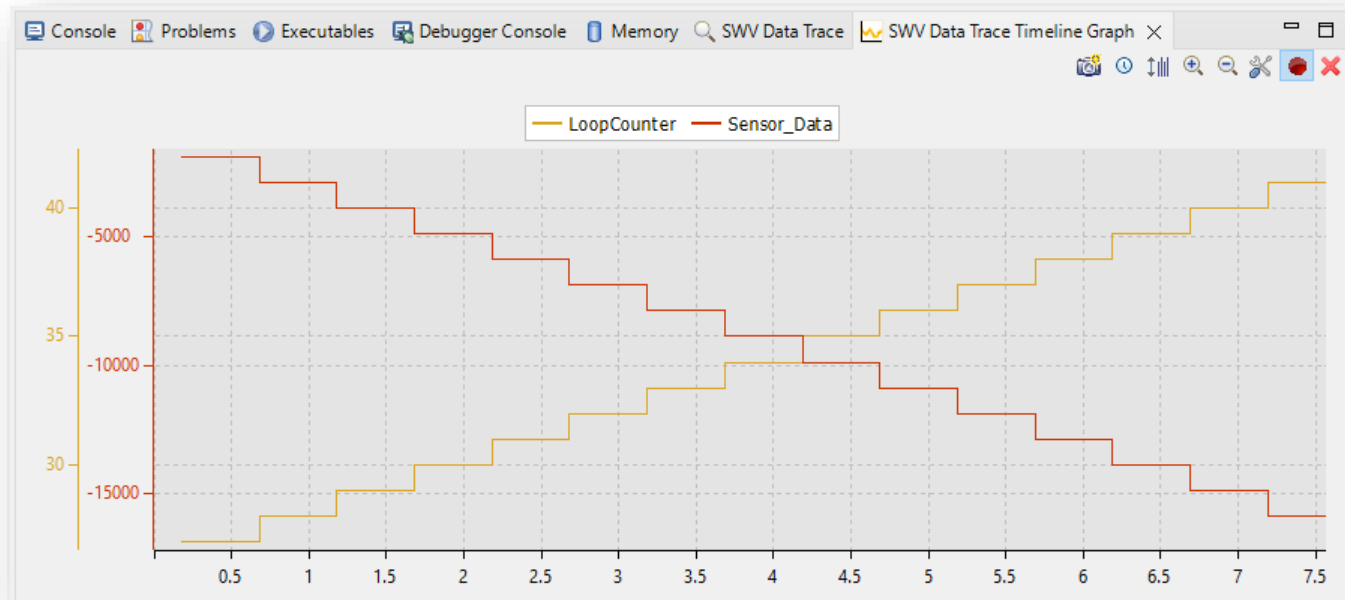
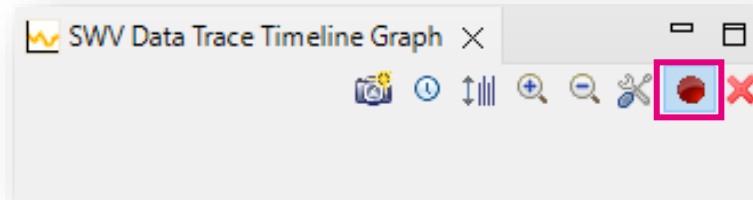
Comparator 3
☐ Enable
Var/Addr: 0x0
Access: Read/Write
Size: Word
Generate: Data Value

ITM Stimulus Ports
Enable port: 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Privileged only ports: ☐ Port 31..24 ☐ Port 23..16 ☐ Port 15..8 ☐ Port 7..0

OK Cancel

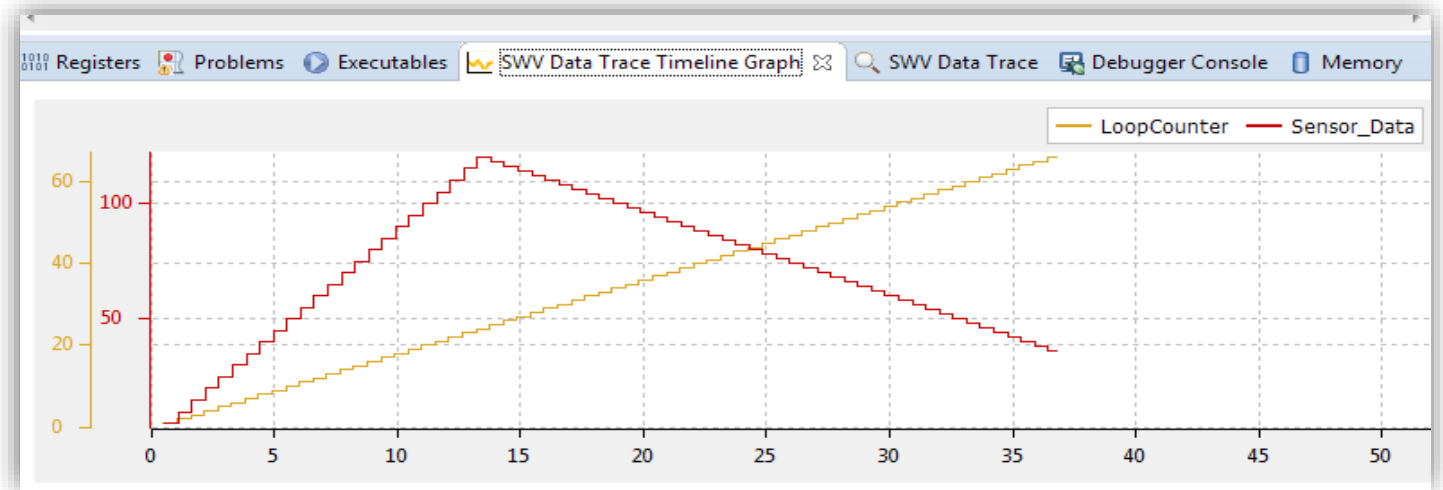
SWV: Using Data Trace Timeline Graph

- Click on “Start Trace” button
- Start Execution
- You shall now start seeing some graphs being plotted in the console
- Note: Keep other trace events disabled to avoid chances of overflow in the SWO line



SWV: Using Data Trace Timeline Graph

- Few Sample Graphs generated
- Multiple Data variables can be plotted
- Color coding to identify different variables





life.augmented

STM32CubeIDE

SWV: printf() redirection

SWV: printf() redirection

- Typically, we use printf() messages sent via UART to troubleshoot the application during run time
- SWV can be used for **printf() re-direction**
- Debug messages are sent to a “Console” window in the debugger using the **Serial Wire Output (SWO)** pin
- No need for a USB or UART cable

Redirect printf() to SWO pin

- Include stdio.h library to declare printf functionality
- Add code to print messages
- To redirect printf() to SWO pin
 - Redefine _write() function to send data to SWO pin using ITM_SendChar() function
 - Weak _write() included in syscalls.c file
 - Same technique can be used to print to USART pins

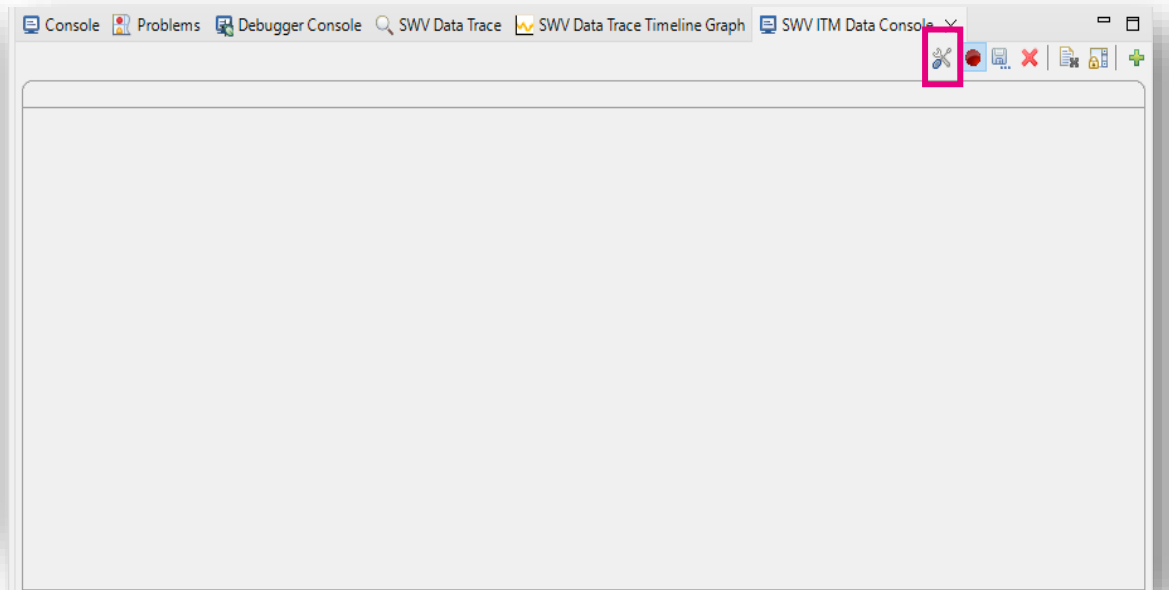
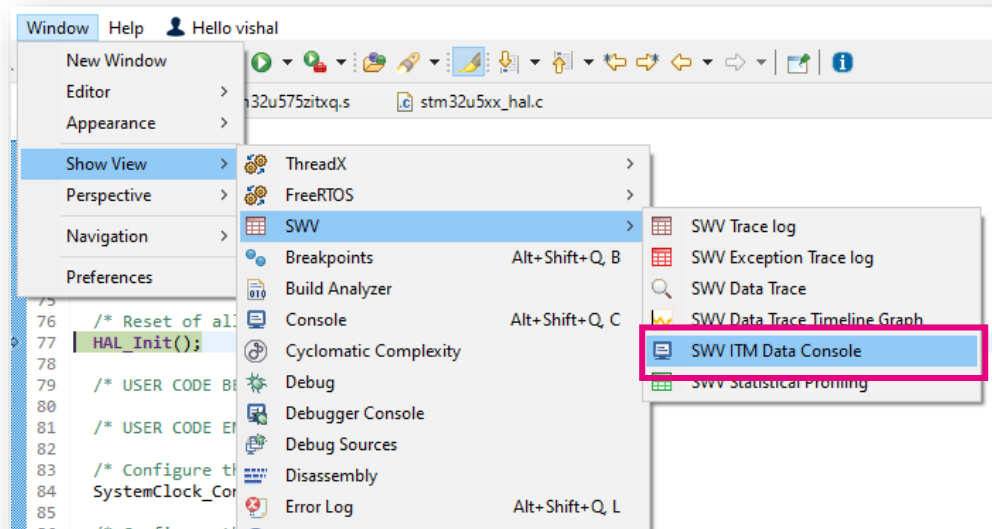
```
/* USER CODE BEGIN Includes */  
#include <stdio.h>  
/* USER CODE END Includes */
```

```
109 /* Initialize all configured peripherals */  
110 MX_GPIO_Init();  
111 /* USER CODE BEGIN 2 */  
112 printf("GPIO Init done\n");  
113 /* USER CODE END 2 */  
114  
115 /* Infinite loop */  
116 /* USER CODE BEGIN WHILE */  
117 while (1)  
118 {  
119     /* USER CODE END WHILE */  
120  
121     /* USER CODE BEGIN 3 */  
122     printf("In while loop \n");  
123     HAL_GPIO_TogglePin(LED1_GPIO_Port, LED1_Pin);  
124     HAL_Delay(500);  
125     printf("Counter = %i\n", LoopCounter++);
```

```
/* USER CODE BEGIN 4 */  
  
int _write(int file, char *ptr, int len)  
{  
    int DataIdx;  
  
    for(DataIdx=0; DataIdx<len; DataIdx++)  
    {  
        ITM_SendChar(*ptr++);  
    }  
    return len;  
}  
  
/* USER CODE END 4 */
```

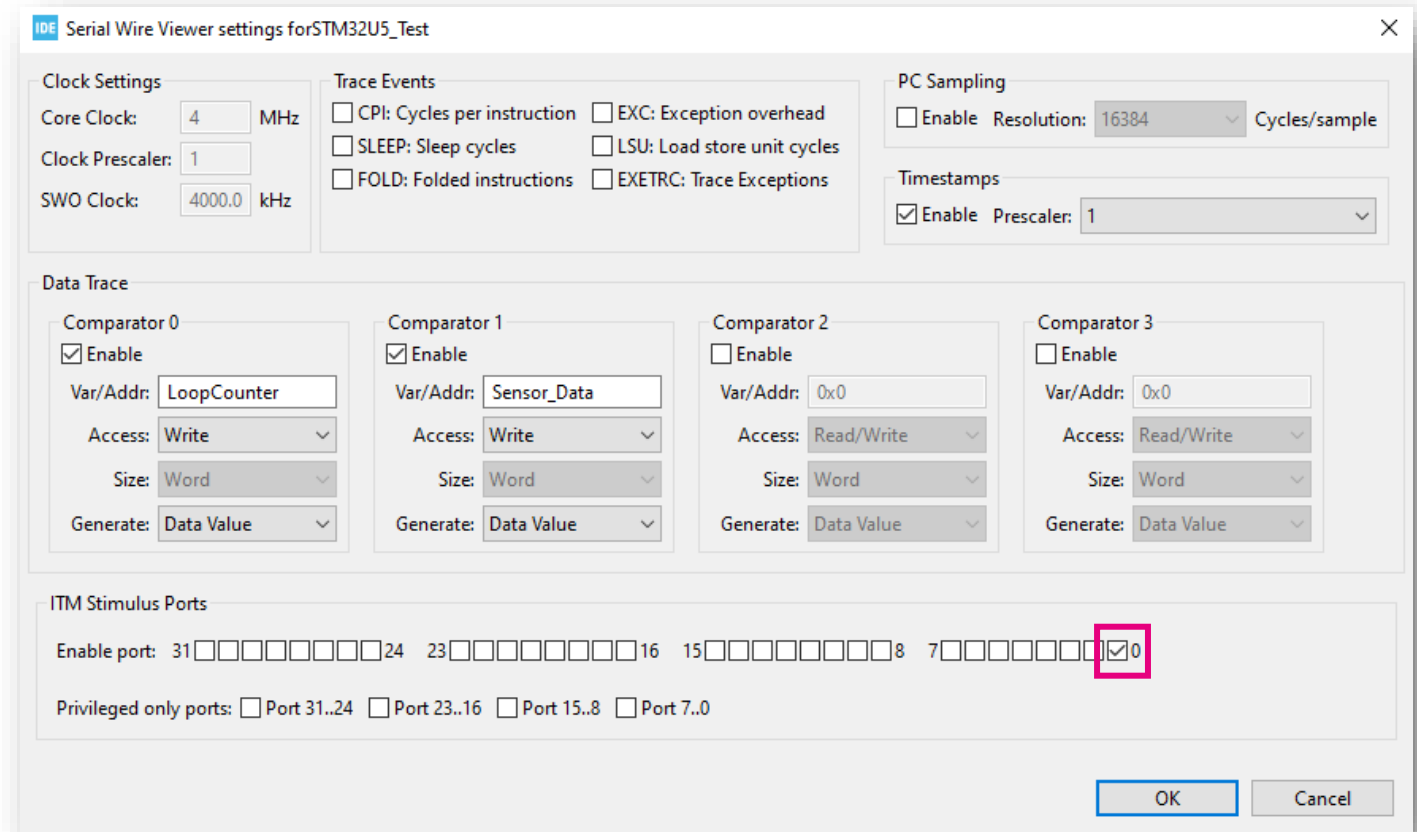
Enable SWV ITM Data Console

- Start Debug session
- Go to **Window -> Show View -> SWV** and click **SWV ITM Data Console**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



Configure SWV settings

- Enable Channel 0 from the ITM Ports
- Note: Keep other trace events disabled.

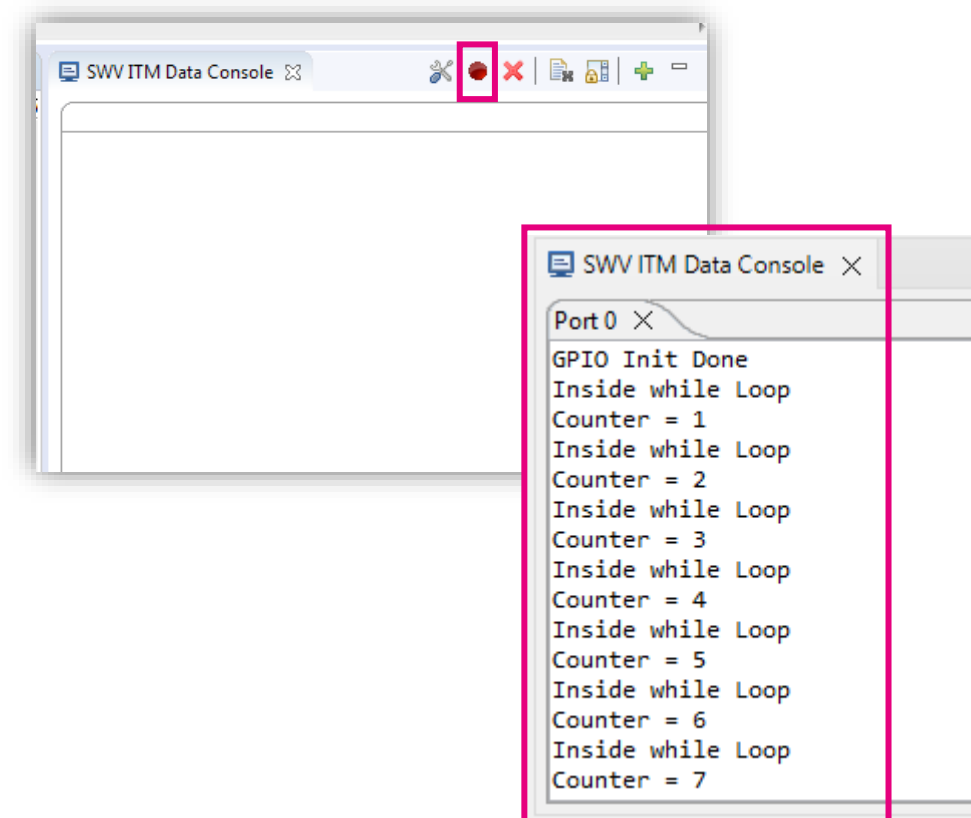


The image shows the 'Serial Wire Viewer settings for STM32U5_Test' dialog box. The 'ITM Stimulus Ports' section at the bottom has a row of checkboxes for ports 31 down to 0. Port 0 is checked and highlighted with a red square. Other sections include 'Clock Settings' (Core Clock: 4 MHz, SWO Clock: 4000.0 kHz), 'Trace Events' (CPI, SLEEP, FOLD, EXC, LSU, EXETRC), 'PC Sampling' (Resolution: 16384 Cycles/sample), 'Timestamps' (Prescaler: 1), and 'Data Trace' (Comparators 0-3). Comparators 0 and 1 are enabled with specific variables and access types. Comparators 2 and 3 are disabled.

Port	Enable
31	<input type="checkbox"/>
24	<input type="checkbox"/>
23	<input type="checkbox"/>
16	<input type="checkbox"/>
15	<input type="checkbox"/>
8	<input type="checkbox"/>
7	<input type="checkbox"/>
0	<input checked="" type="checkbox"/>

SWV printf() redirection

- Click on “Start Trace” button
- Start Execution
- View printf() messages in the “SWV Data Console”





life.augmented

STM32CubeIDE

SWV: Timing Measurement

SWV: Timing Measurement

- Used to **measure execution timings** for a process/function
- Step 1:** Add ITM Stimulus Port register definition MACRO to your source code (main.c), which allows us to write to an ITM Port
 - `#define ITM_Port32(n) ((volatile unsigned long *) (0xE0000000+4*n))`
- Step 2:** Write 2 different values to the **ITM Port 31** **before** and **after** the desired functions
 - `ITM_Port32(31) = 1;`
 - `ITM_Port32(31) = 2;`

1

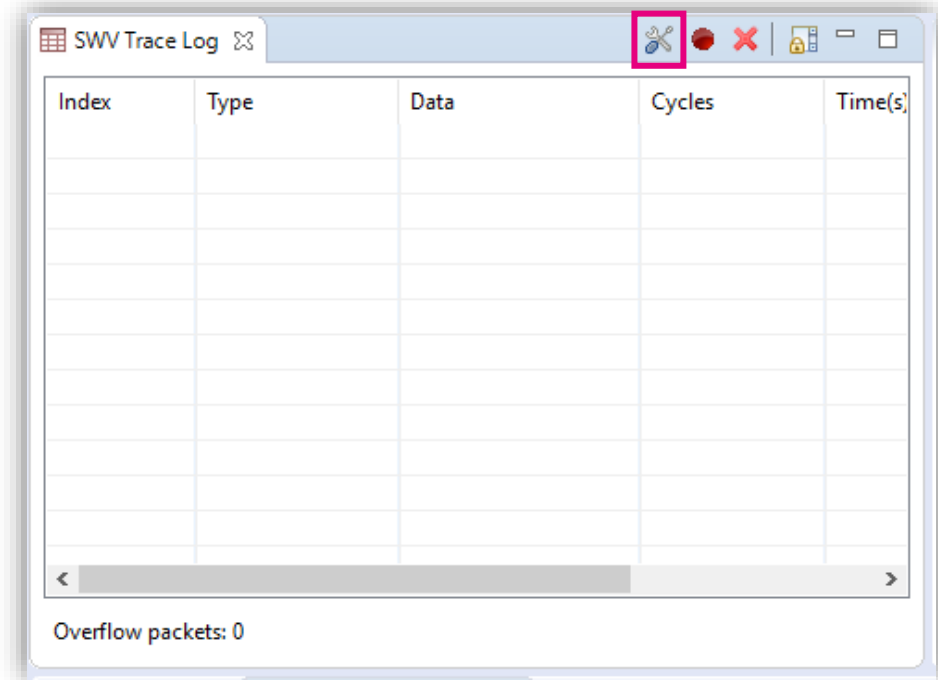
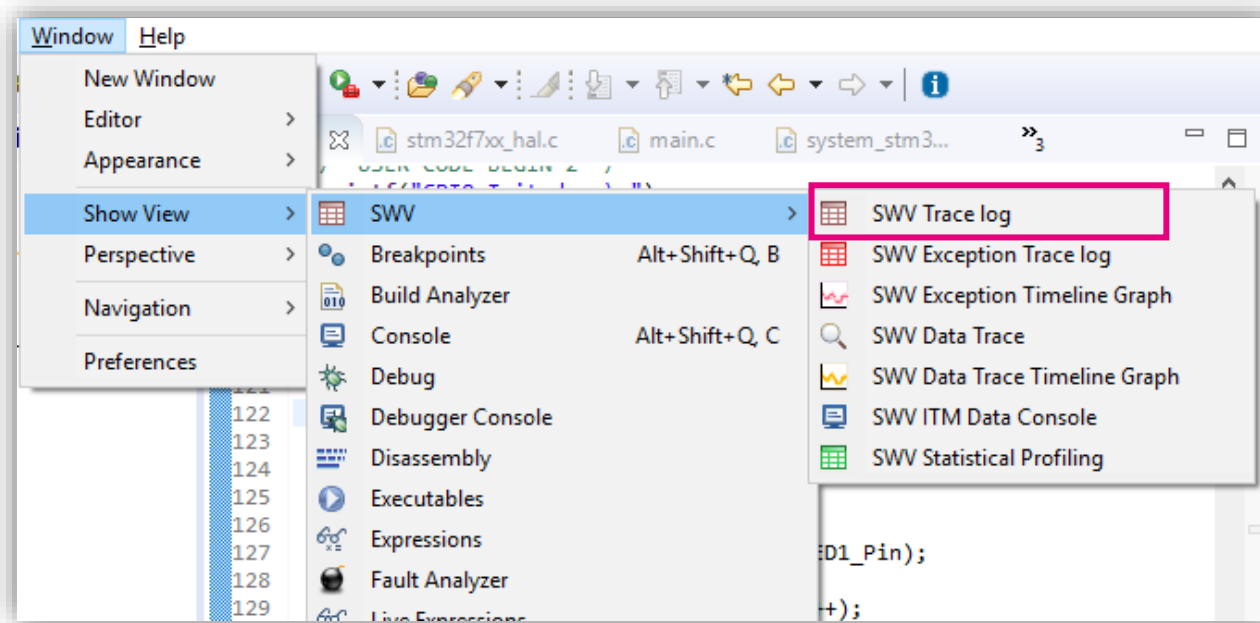
```
78 /* Private user code -----  
79 /* USER CODE BEGIN 0 */  
80 #define ITM_Port32(n) ((volatile unsigned long *) (0xE0000000+4*n))  
81 /* USER CODE END 0 */  
82  
83 /**  
84  * @brief The application entry point.  
85  * @retval int  
86  */  
87 int main(void)  
88 {  
89  /* USER CODE BEGIN 1 */
```

2

```
main.c syscalls.c stm32f7xx_hal.c startup_s  
106 printf("System Init done\n");  
107 /* USER CODE END SysInit */  
108  
109 /* Initialize all configured peripherals */  
110 ITM_Port32(31) = 1;  
111  
112  
113 MX_GPIO_Init();  
114 /* USER CODE BEGIN 2 */  
115 printf("GPIO Init done\n");  
116  
117  
118 ITM_Port32(31) = 2;  
119 /* USER CODE END 2 */  
120
```

Enable SWV ITM Data Console

- Start Debug Session
- Go to **Window -> Show View -> SWV** and click **SWV Trace Log**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



SWV: Timing Measurement

- Enable **Timestamps**
- Enable Channel 31 *only* from the ITM Ports
- Note: Keep other trace events disabled to avoid chances of overflow in the SWO line

IDE Serial Wire Viewer settings for STM32U5_Test

Clock Settings
Core Clock: 4 MHz
Clock Prescaler: 1
SWO Clock: 4000.0 kHz

Trace Events
☐ CPI: Cycles per instruction ☐ EXC: Exception overhead
☐ SLEEP: Sleep cycles ☐ LSU: Load store unit cycles
☐ FOLD: Folded instructions ☐ EXETRC: Trace Exceptions

PC Sampling
☐ Enable Resolution: 16384 Cycles/sample

Timestamps
☒ Enable Prescaler: 1

Data Trace

Comparator 0	Comparator 1	Comparator 2	Comparator 3
<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable
Var/Addr: LoopCounter	Var/Addr: Sensor_Data	Var/Addr: 0x0	Var/Addr: 0x0
Access: Write	Access: Write	Access: Read/Write	Access: Read/Write
Size: Word	Size: Word	Size: Word	Size: Word
Generate: Data Value	Generate: Data Value	Generate: Data Value	Generate: Data Value

ITM Stimulus Ports

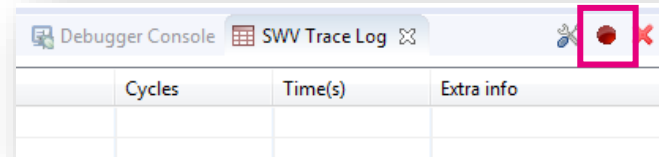
Enable port: ☒ 31 ☐ 30 ☐ 29 ☐ 28 ☐ 27 ☐ 26 ☐ 25 ☐ 24 ☐ 23 ☐ 22 ☐ 21 ☐ 20 ☐ 19 ☐ 18 ☐ 17 ☐ 16 ☐ 15 ☐ 14 ☐ 13 ☐ 12 ☐ 11 ☐ 10 ☐ 9 ☐ 8 ☐ 7 ☐ 6 ☐ 5 ☐ 4 ☐ 3 ☐ 2 ☐ 1 ☐ 0

Privileged only ports: ☐ Port 31..24 ☐ Port 23..16 ☐ Port 15..8 ☐ Port 7..0

OK Cancel

SWV: Timing Measurement

- Click on “Start Trace” button
- Start Execution
- Observe the Trace values & timestamps in the “SWV Trace Log Console”



A screenshot of the 'SWV Trace Log' window showing a table of trace data. The table has columns 'Index', 'Type', 'Data', 'Cycles', and 'Time(s)'. The 'Time(s)' column is highlighted with a red rectangular box. The table contains two rows of data.

Index	Type	Data	Cycles	Time(s)
0	ITM Port 31	1	7101	1.775250 ms
1	ITM Port 31	2	15790	3.947500 ms



life.augmented

STM32CubeIDE

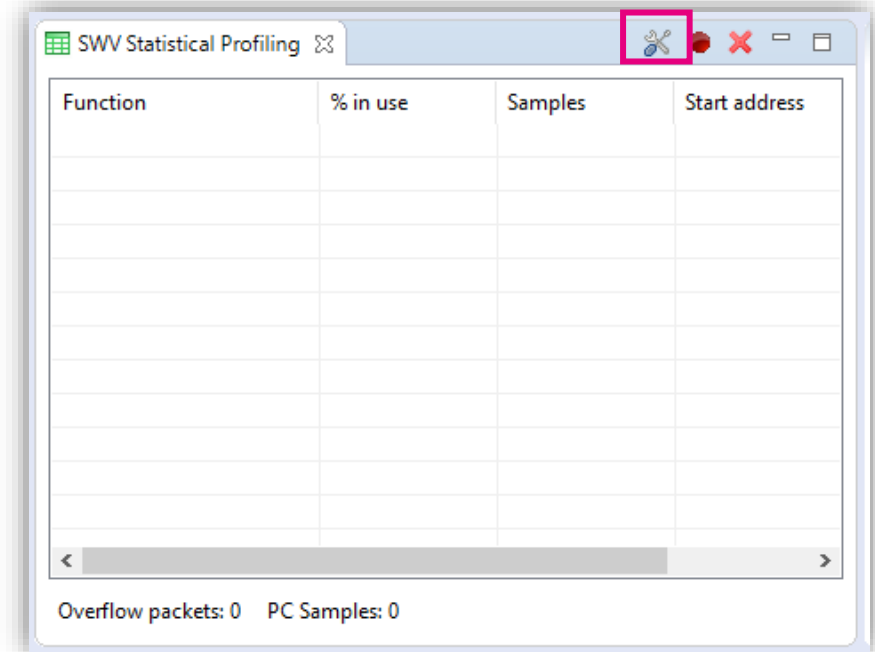
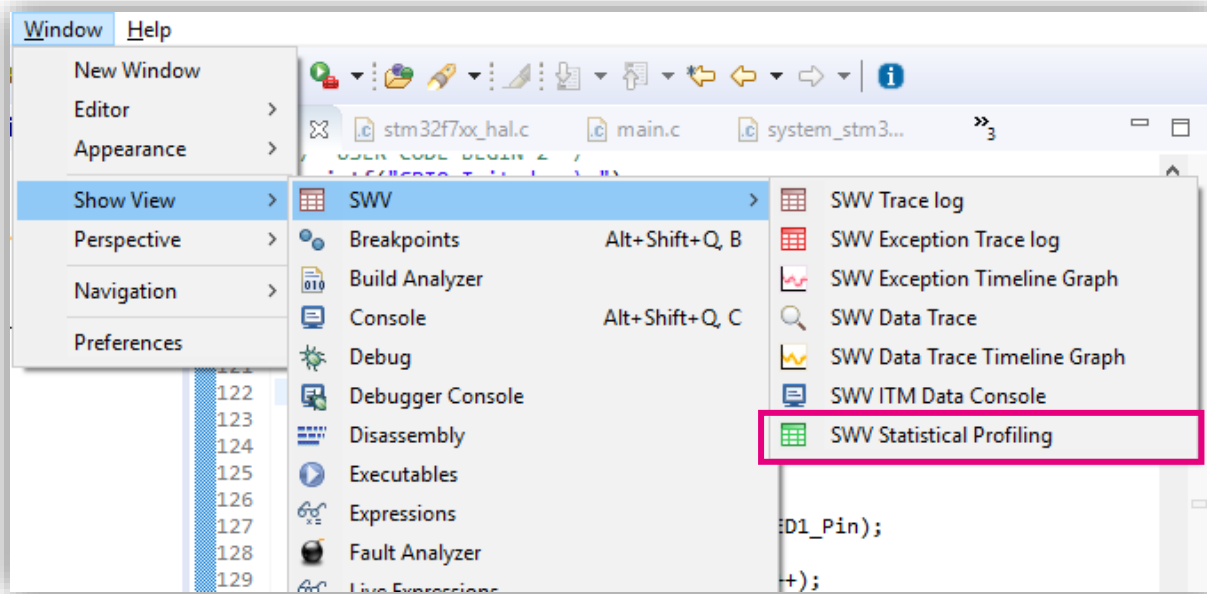
SWV: Statistical Profiling

SWV: Statistical Profiling

- Statistical profiling feature of the SWV provides a way to quickly analyze the performance of the application code and optimize speed.

Enable SWV Statistical Profiling

- Start Debug Session
- Go to **Window -> Show View -> SWV** and click **SWV Statistical Profiling**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



Configure SWV settings

- Enable PC Sampling and click ok
- Note: Keep other trace events disabled to avoid chances of overflow in the SWO line

Serial Wire Viewer settings for STM32U5_Test

Clock Settings
Core Clock: 4 MHz
Clock Prescaler: 1
SWO Clock: 4000.0 kHz

Trace Events
☐ CPI: Cycles per instruction ☐ EXC: Exception overhead
☐ SLEEP: Sleep cycles ☐ LSU: Load store unit cycles
☐ FOLD: Folded instructions ☐ EXETRC: Trace Exceptions

PC Sampling
☒ Enable Resolution: 16384 Cycles/sample

Timestamps
☐ Enable Prescaler: 1

Data Trace

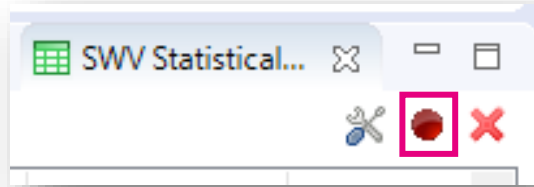
Comparator 0	Comparator 1	Comparator 2	Comparator 3
<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable
Var/Addr: LoopCounter	Var/Addr: Sensor_Data	Var/Addr: 0x0	Var/Addr: 0x0
Access: Write	Access: Write	Access: Read/Write	Access: Read/Write
Size: Word	Size: Word	Size: Word	Size: Word
Generate: Data Value	Generate: Data Value	Generate: Data Value	Generate: Data Value

ITM Stimulus Ports
Enable port: 31 24 23 16 15 8 7 0
Privileged only ports: ☐ Port 31..24 ☐ Port 23..16 ☐ Port 15..8 ☐ Port 7..0

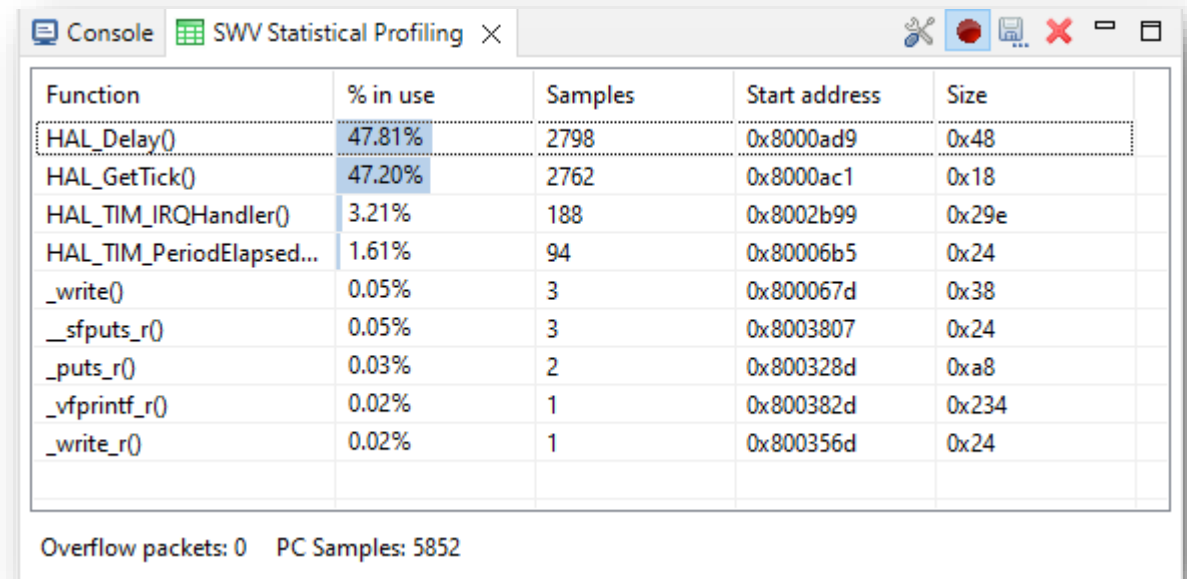
OK Cancel

Start Statistical Profiling view

- Click on the Start Trace button.



- Execute the code.
- Halt the execution in order to see a statistical profile of the application when it was running at that particular time frame.

A screenshot of a software window titled "SWV Statistical Profiling". The window contains a table with the following data:

Function	% in use	Samples	Start address	Size
HAL_Delay()	47.81%	2798	0x8000ad9	0x48
HAL_GetTick()	47.20%	2762	0x8000ac1	0x18
HAL_TIM_IRQHandler()	3.21%	188	0x8002b99	0x29e
HAL_TIM_PeriodElapsed...	1.61%	94	0x80006b5	0x24
_write()	0.05%	3	0x800067d	0x38
_sfputs_r()	0.05%	3	0x8003807	0x24
_puts_r()	0.03%	2	0x800328d	0xa8
_vfprintf_r()	0.02%	1	0x800382d	0x234
_write_r()	0.02%	1	0x800356d	0x24

Below the table, it says "Overflow packets: 0 PC Samples: 5852".



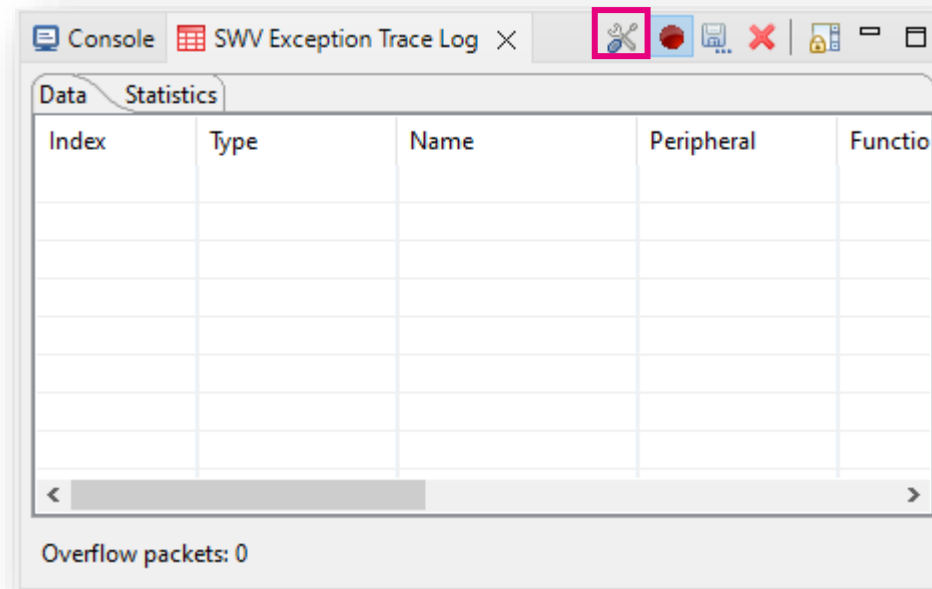
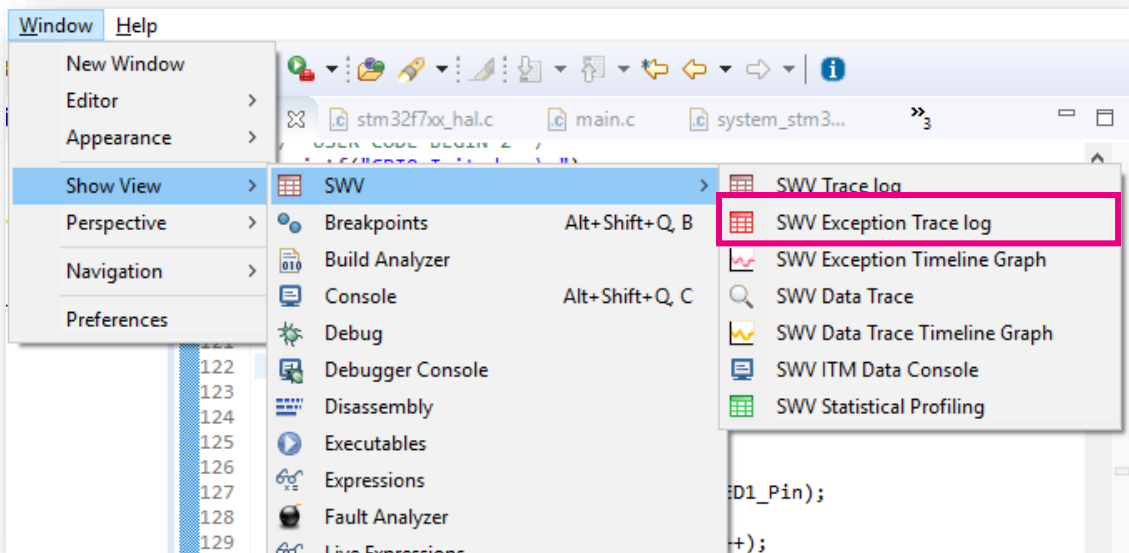
life.augmented

STM32CubeIDE

SWV: Exception Trace Log

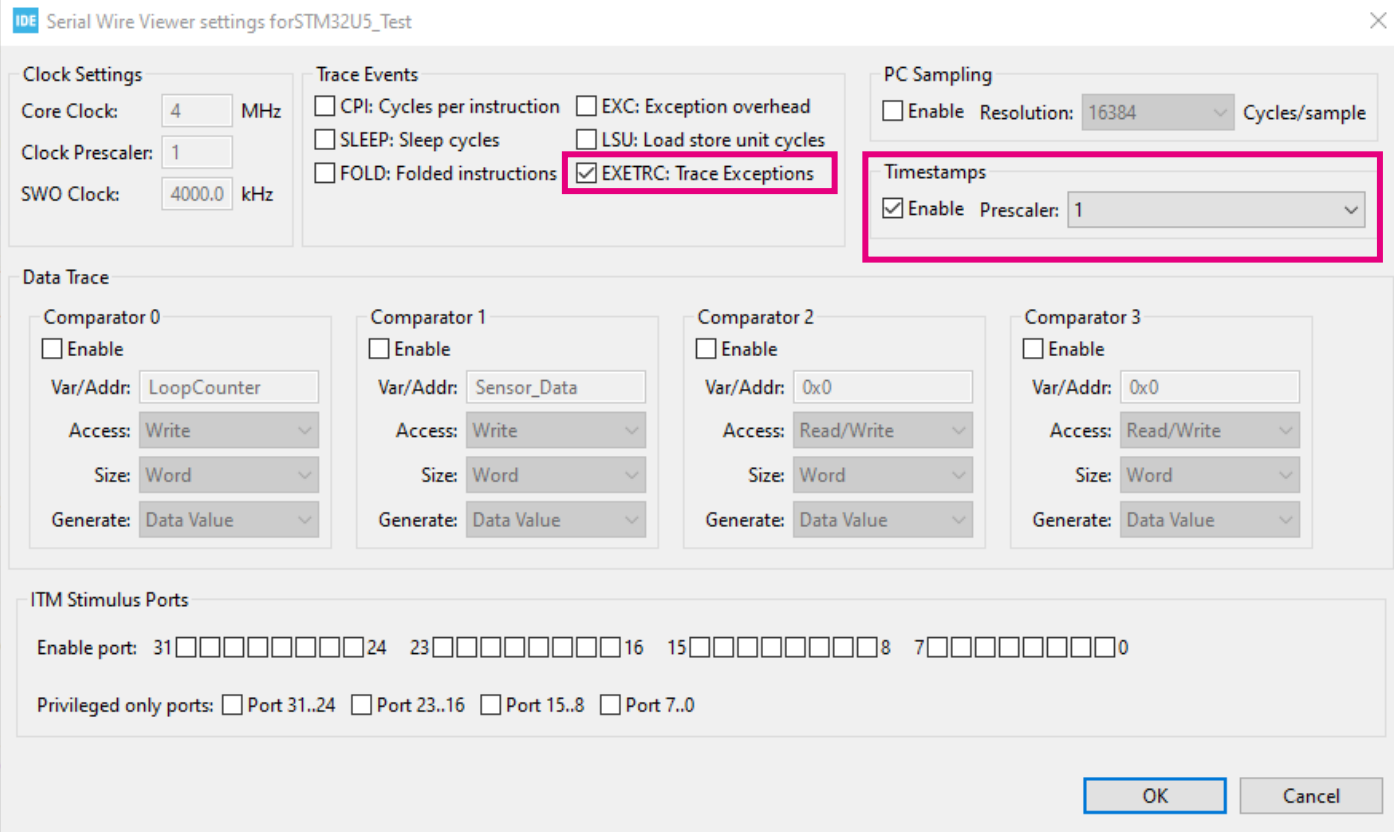
Enable SWV Exception Trace Log

- Start Debug session
- Go to **Window -> Show View -> SWV** and click **SWV Exception Trace Log**
- Click on “Configure Trace” button to open the Serial Wire Viewer settings



Configure SWV settings

- Enable Timestamps
- Enable EXETRC: Trace Exceptions
- Click OK
- Note: Keep other trace events disabled to avoid chances of overflow in the SWO line



The image shows the 'Serial Wire Viewer settings for STM32U5_Test' dialog box. The 'Trace Events' section has 'EXETRC: Trace Exceptions' checked and highlighted with a red box. The 'Timestamps' section has 'Enable' checked and 'Prescaler' set to 1, also highlighted with a red box. The 'Data Trace' section shows four comparators, all with 'Generate' set to 'Data Value'. The 'ITM Stimulus Ports' section shows 'Enable port' settings for ports 31, 24, 23, 16, 15, 8, 7, and 0, with 'Privileged only ports' set to 'Port 31..24', 'Port 23..16', 'Port 15..8', and 'Port 7..0'.

Serial Wire Viewer settings for STM32U5_Test

Clock Settings
Core Clock: 4 MHz
Clock Prescaler: 1
SWO Clock: 4000.0 kHz

Trace Events
☐ CPI: Cycles per instruction
☐ EXC: Exception overhead
☐ SLEEP: Sleep cycles
☐ LSU: Load store unit cycles
☐ FOLD: Folded instructions
☒ EXETRC: Trace Exceptions

PC Sampling
☐ Enable Resolution: 16384 Cycles/sample

Timestamps
☒ Enable Prescaler: 1

Data Trace

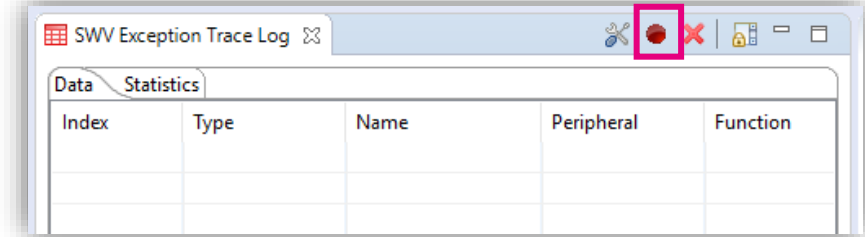
Comparator 0	Comparator 1	Comparator 2	Comparator 3
<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable	<input type="checkbox"/> Enable
Var/Addr: LoopCounter	Var/Addr: Sensor_Data	Var/Addr: 0x0	Var/Addr: 0x0
Access: Write	Access: Write	Access: Read/Write	Access: Read/Write
Size: Word	Size: Word	Size: Word	Size: Word
Generate: Data Value	Generate: Data Value	Generate: Data Value	Generate: Data Value

ITM Stimulus Ports
Enable port: 31 ☐ 24 ☐ 23 ☐ 16 ☐ 15 ☐ 8 ☐ 7 ☐ 0 ☐
Privileged only ports: ☐ Port 31..24 ☐ Port 23..16 ☐ Port 15..8 ☐ Port 7..0

OK Cancel

SWV: Exception Trace Log

- Click on Start Trace button
- Execute the code to see exception logs
- Click on the statistics tab to the exception profile.



The screenshot shows the 'SWV Exception Trace Log' window with the 'Data' tab selected. The table displays a list of exception events. The 'Name' column shows 'SYSTICK (EXC 15)' for entry and exit events, and 'N/A (EXC 0)' for return events. The 'Function' column shows 'SysTick_Handler()' for entry and exit events.

Index	Type	Name	Peripheral	Function
46226	Exception exit	SYSTICK (EXC 15)		SysTick_Handler()
46227	Exception return	N/A (EXC 0)		
46228	Exception entry	SYSTICK (EXC 15)		SysTick_Handler()
46229	Exception exit	SYSTICK (EXC 15)		SysTick_Handler()
46230	Exception return	N/A (EXC 0)		
46231	Exception entry	SYSTICK (EXC 15)		SysTick_Handler()
46232	Exception exit	SYSTICK (EXC 15)		SysTick_Handler()
46233	Exception return	N/A (EXC 0)		
46234	Exception entry	SYSTICK (EXC 15)		SysTick_Handler()
46235	Exception exit	SYSTICK (EXC 15)		SysTick_Handler()
46236	Exception return	N/A (EXC 0)		

Overflow packets: 0

The screenshot shows the 'SWV Exception Trace Log' window with the 'Statistics' tab selected. The table displays summary statistics for the exception events. The 'Exception' column shows 'SYSTICK (EXC 15)', the 'Handler' column shows 'SysTick_Handler()', the '% of' column shows '100.0000%', the 'Number of' column shows '108957', and the '% of exc' column shows '100.0000%'. The 'Total for all' row shows the total number of exceptions.

Exception	Handler	% of	Number of	% of exc
SYSTICK (EXC 15)	SysTick_Handler()	100.0000%	108957	100.0000%
Total for all			108957	

Overflow packets: 0



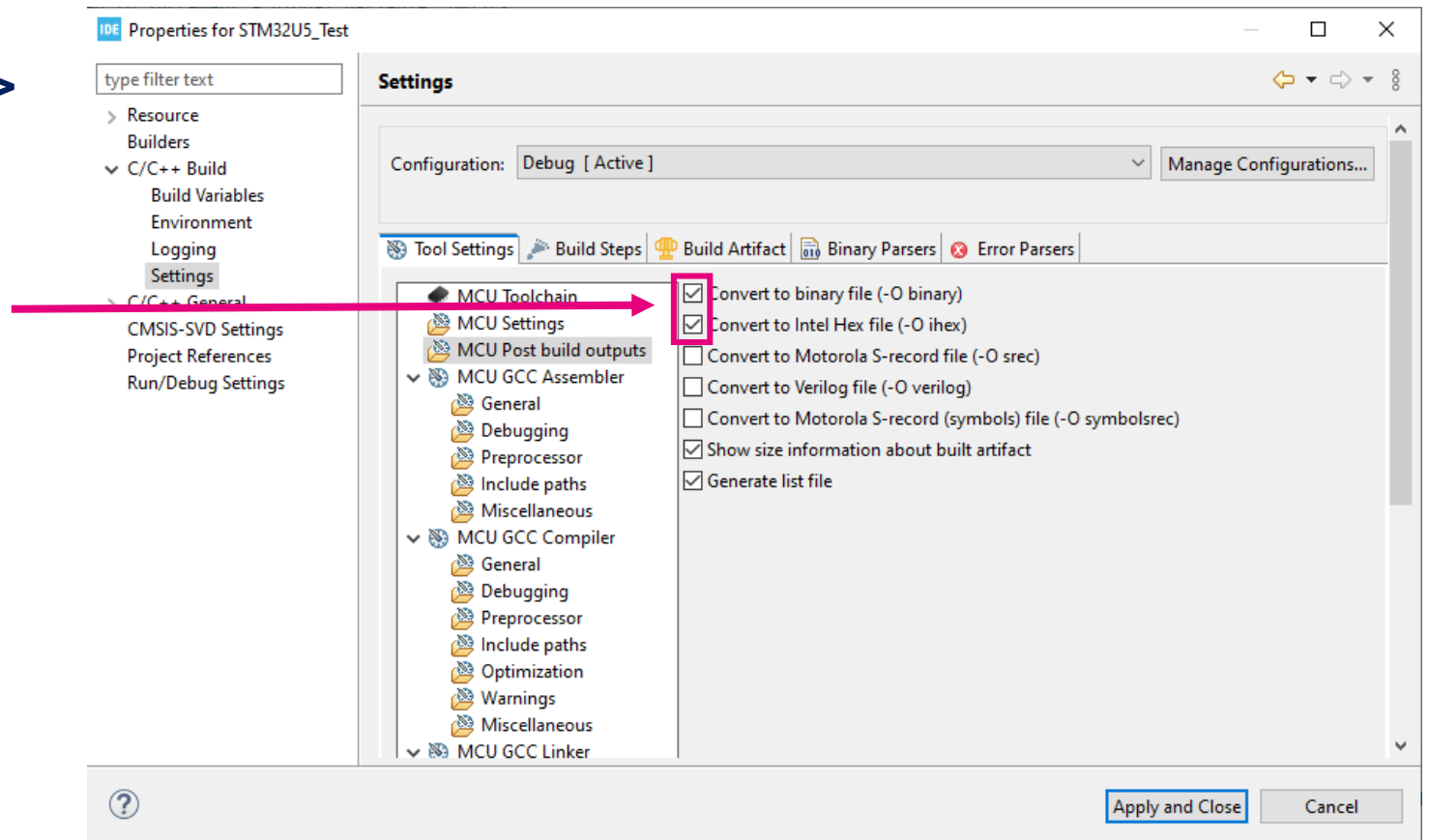
life.augmented

STM32CubeIDE

More IDE Options

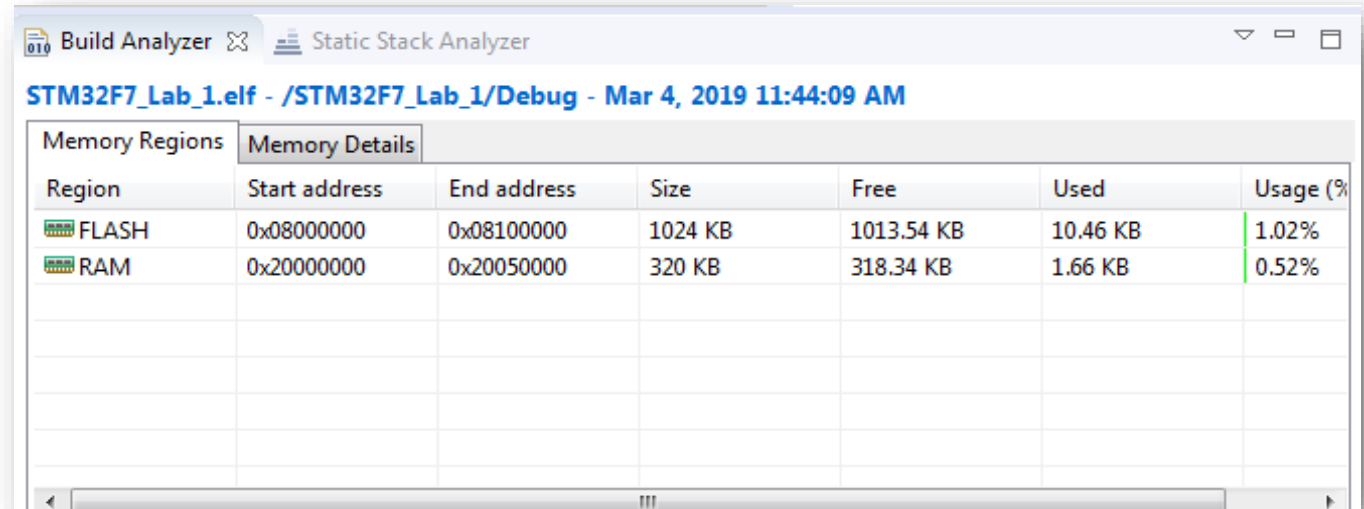
IDE options: Good-to-know

- To generate binary (.bin) and hex (.hex) output files
 - Go to **Project >> Properties >> C/C++ Build >> Settings >> Tool Settings >> MCU Post build outputs**
 - Enable these 2 checkboxes to generate binary (.bin) and hex (.hex) output files



IDE options: Good-to-know

- To check the **Flash and RAM consumption** : **Build Analyzer** -> **Memory Regions**

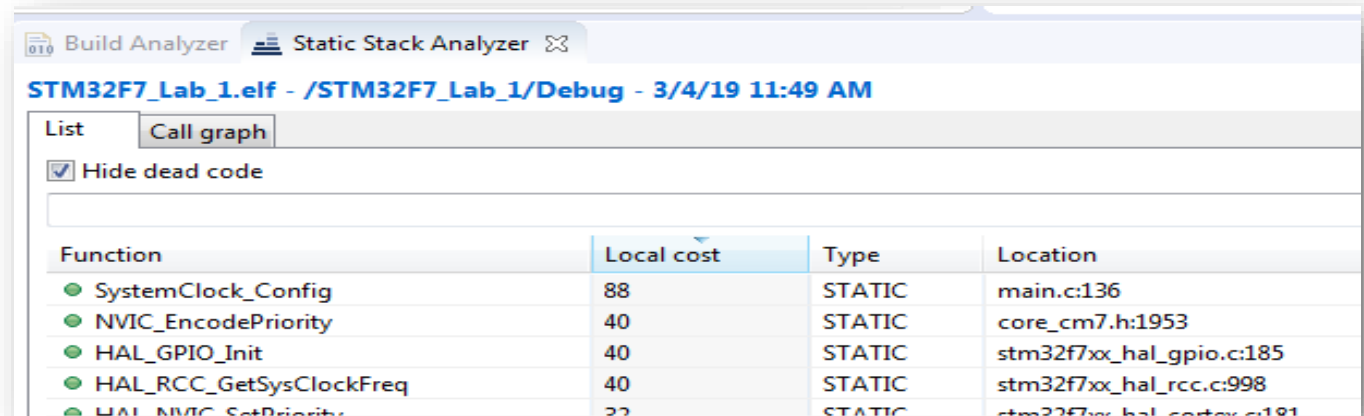


Build Analyzer Static Stack Analyzer

STM32F7_Lab_1.elf - /STM32F7_Lab_1/Debug - Mar 4, 2019 11:44:09 AM

Memory Regions	Memory Details					
Region	Start address	End address	Size	Free	Used	Usage (%)
FLASH	0x08000000	0x08100000	1024 KB	1013.54 KB	10.46 KB	1.02%
RAM	0x20000000	0x20050000	320 KB	318.34 KB	1.66 KB	0.52%

- To check **stack** consumption of individual functions: **Static Stack Analyzer** -> **List**



Build Analyzer Static Stack Analyzer

STM32F7_Lab_1.elf - /STM32F7_Lab_1/Debug - 3/4/19 11:49 AM

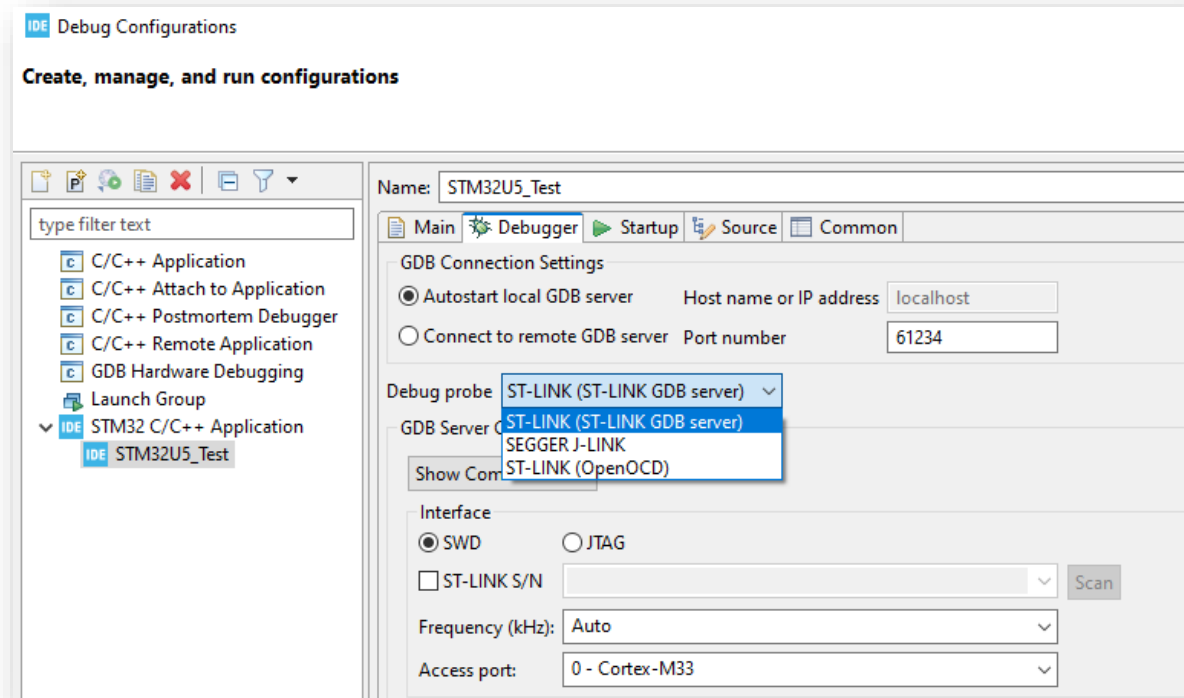
List Call graph

☒ Hide dead code


Function	Local cost	Type	Location
SystemClock_Config	88	STATIC	main.c:136
NVIC_EncodePriority	40	STATIC	core_cm7.h:1953
HAL_GPIO_Init	40	STATIC	stm32f7xx_hal_gpio.c:185
HAL_RCC_GetSysClockFreq	40	STATIC	stm32f7xx_hal_rcc.c:998
HAL_NVIC_SetPriority	22	STATIC	stm32f7xx_hal_nvic.c:191

IDE options: Good-to-know


- STM32CubeIDE supports below Debuggers
 - **ST-LINK (GDB server)**
 - **ST-LINK (OpenOCD)**
 - **SEGGER J-LINK**





STM32CubeMonitor

STM32 
CubeMonitor

Several monitors

STM32 
CubeMonitor

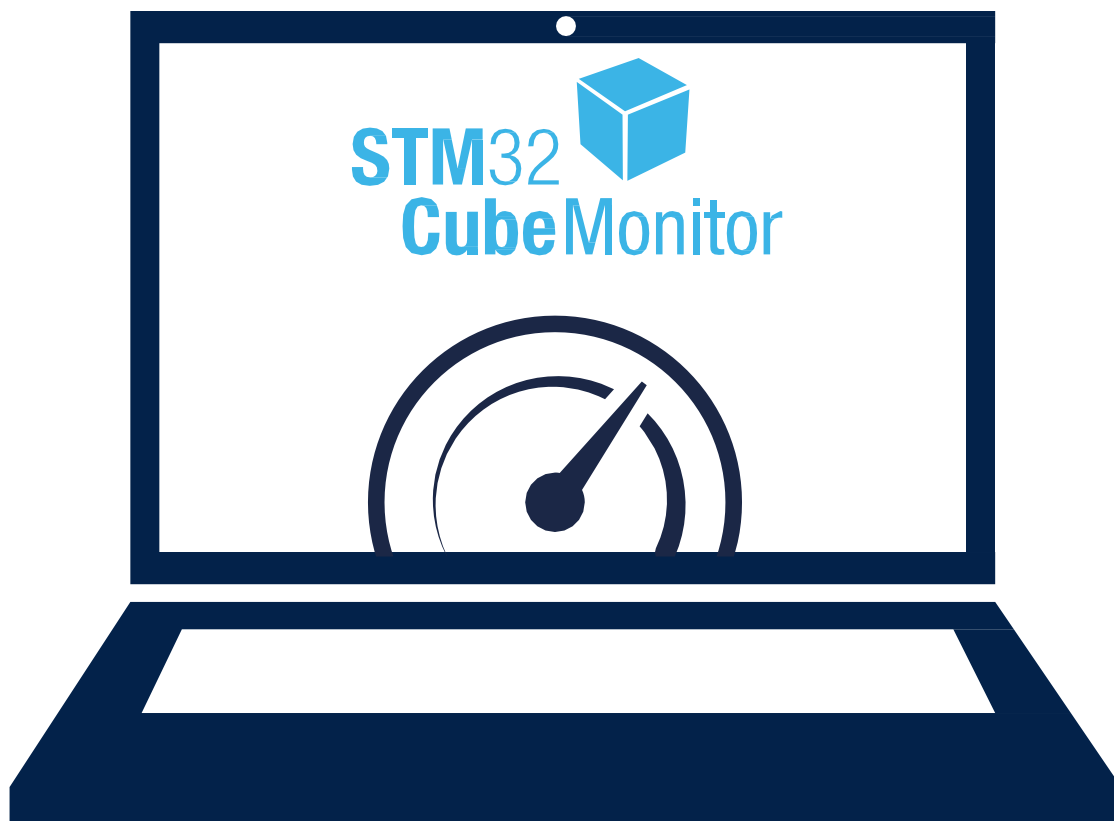
STM32 
CubeMonitor-RF

STM32 
CubeMonitor

STM32 
CubeMonitor-UCPD

STM32 
CubeMonitor-Power

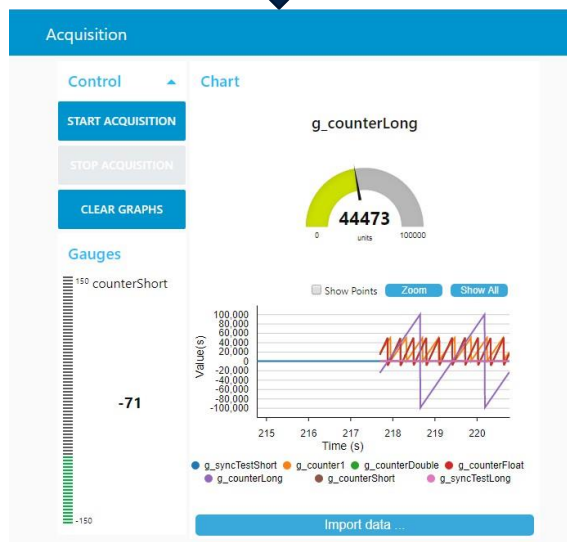
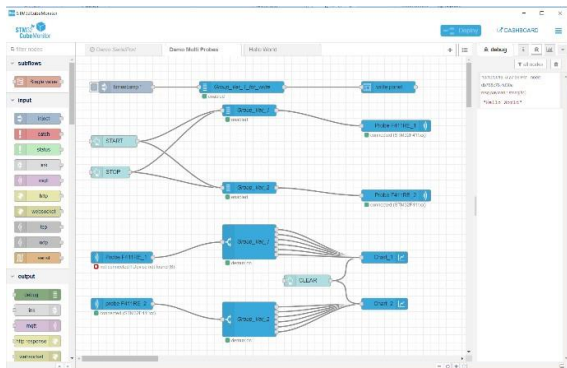
Introducing STM32CubeMonitor



Real-time variable monitoring
during application runtime

Flexible graphical visualization

Direct support of the Node-RED®
open community



Monitoring application variables during runtime

- Non-intrusive tool to follow application behavior without interruption
- Real-time analysis to finetune application configuration

Drag & drop creation of dashboard UI

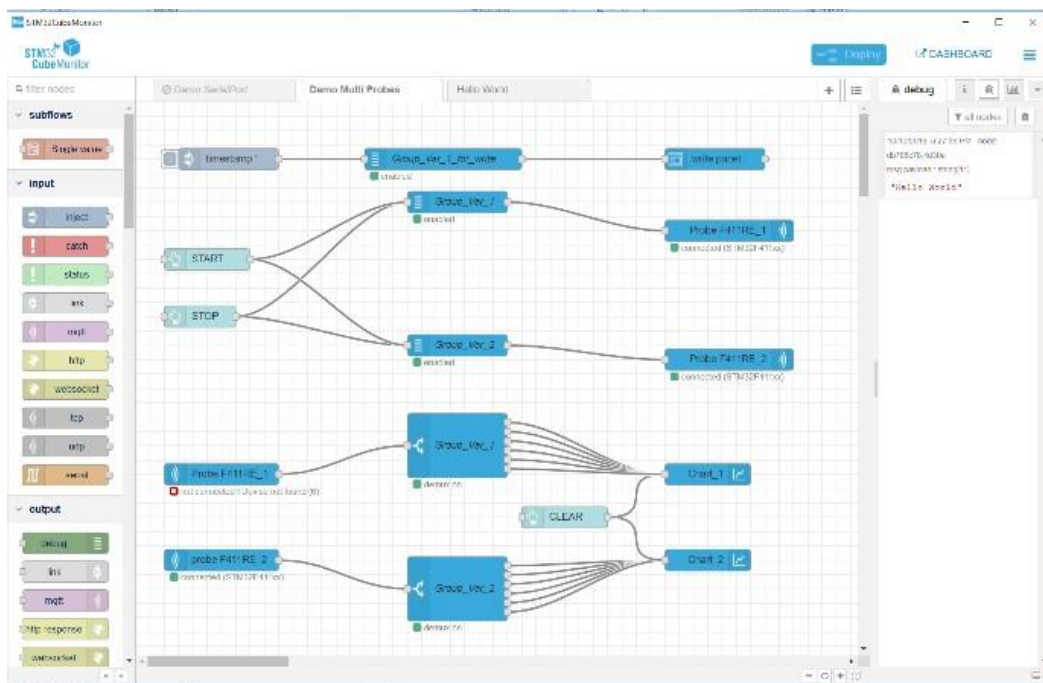
- Large choice of graphical components (gauges, bar graphs, plots...)
- Customize settings. No need for programming.

Graphical visualization on any display

- Multi-OS tool: direct support of PC, tablets and smartphones
- Remote monitoring

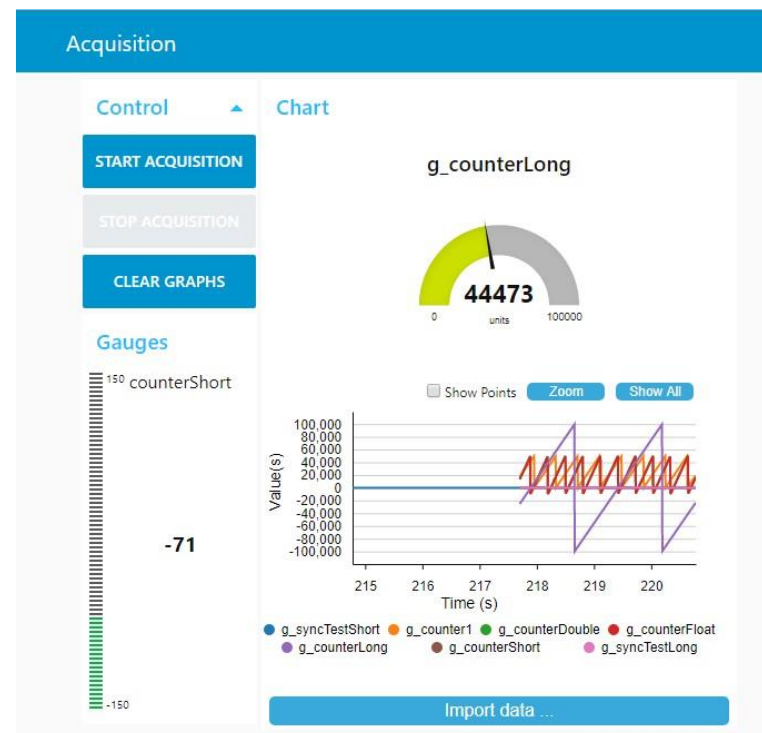
Design mode to create

Build and edit the logical data flow and graphical rendering of the custom monitoring UI.

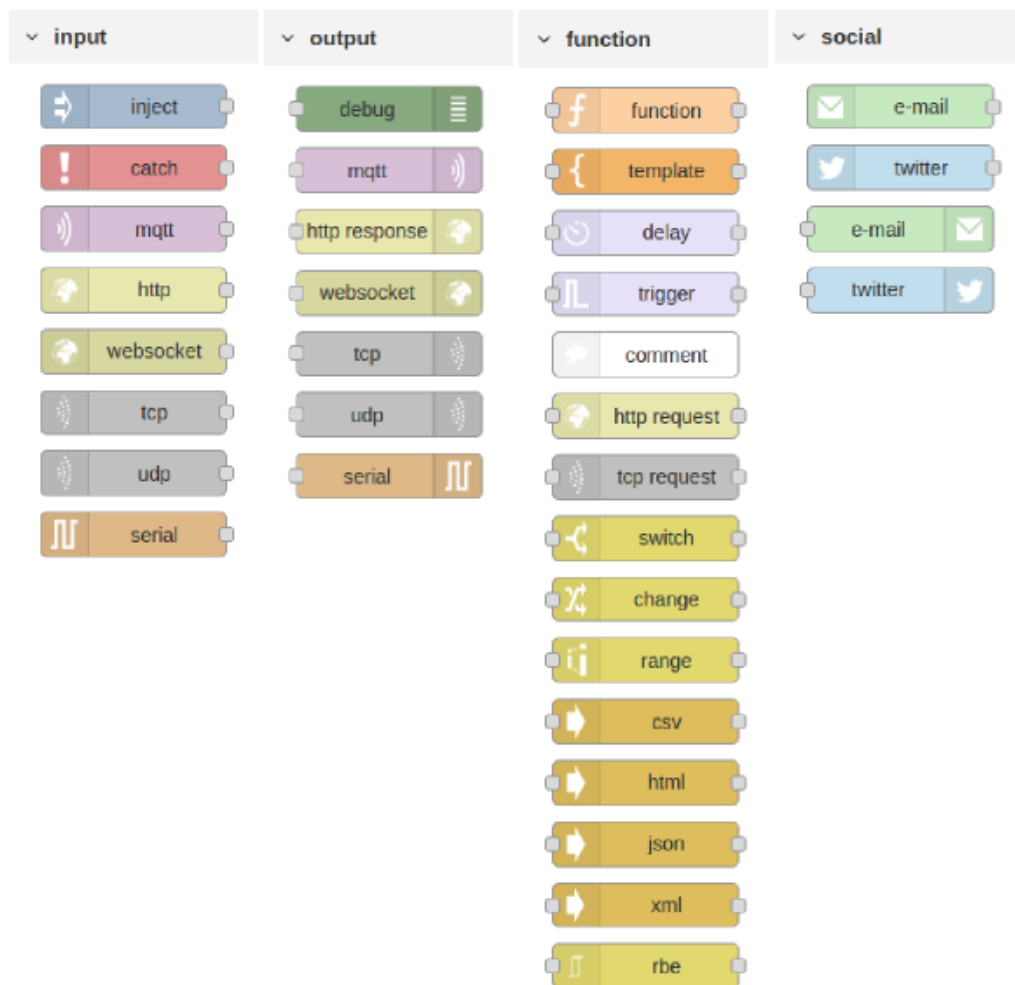


Dashboard mode to visualize

Use the monitoring UI built previously and visualize locally or remotely.



Open community



Direct support of Node-RED community

Flow-based visual programming for connecting hardware devices, APIs and online services (a flow is composed of connected nodes: input, output, function etc.)

Unlimited possibilities

Large choice of ready-to-use nodes to easily expand STM32CubeMonitor default palette.

Presentation of the tool

- STM32CubeMonitor performs data acquisition by reading in the target MCU memory through ST-Link and SWD/JTAG connection.
- The elements involved are :
 - STM32 JTAG/SWD access port
used to connect the MCU debug block to ST-Link.
 - ST-Link device.
 - STM32CubeMonitor software.
- **All the MCU address** space is accessible.

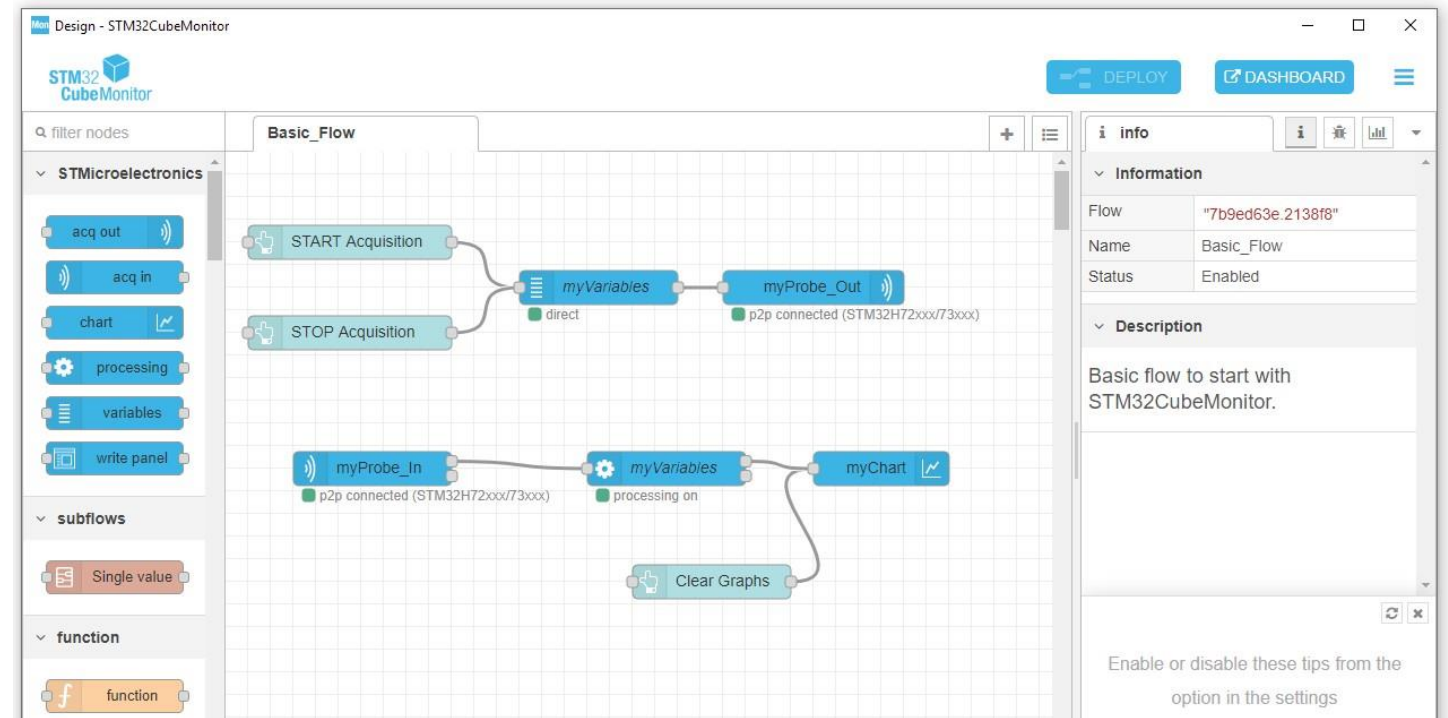


Let's start with STM32CubeMonitor

STM32 
CubeMonitor

The design mode

- Let's start from this point
 - Please plug in your STM32 board to your PC
 - Launch STM32CubeMonitor
- STM32CubeMonitor starts in design mode
 - The basic flow is displayed



Flow structure

- One TX path from PC to probe

- START Acquisition

Messages are sent on the ST-Link:

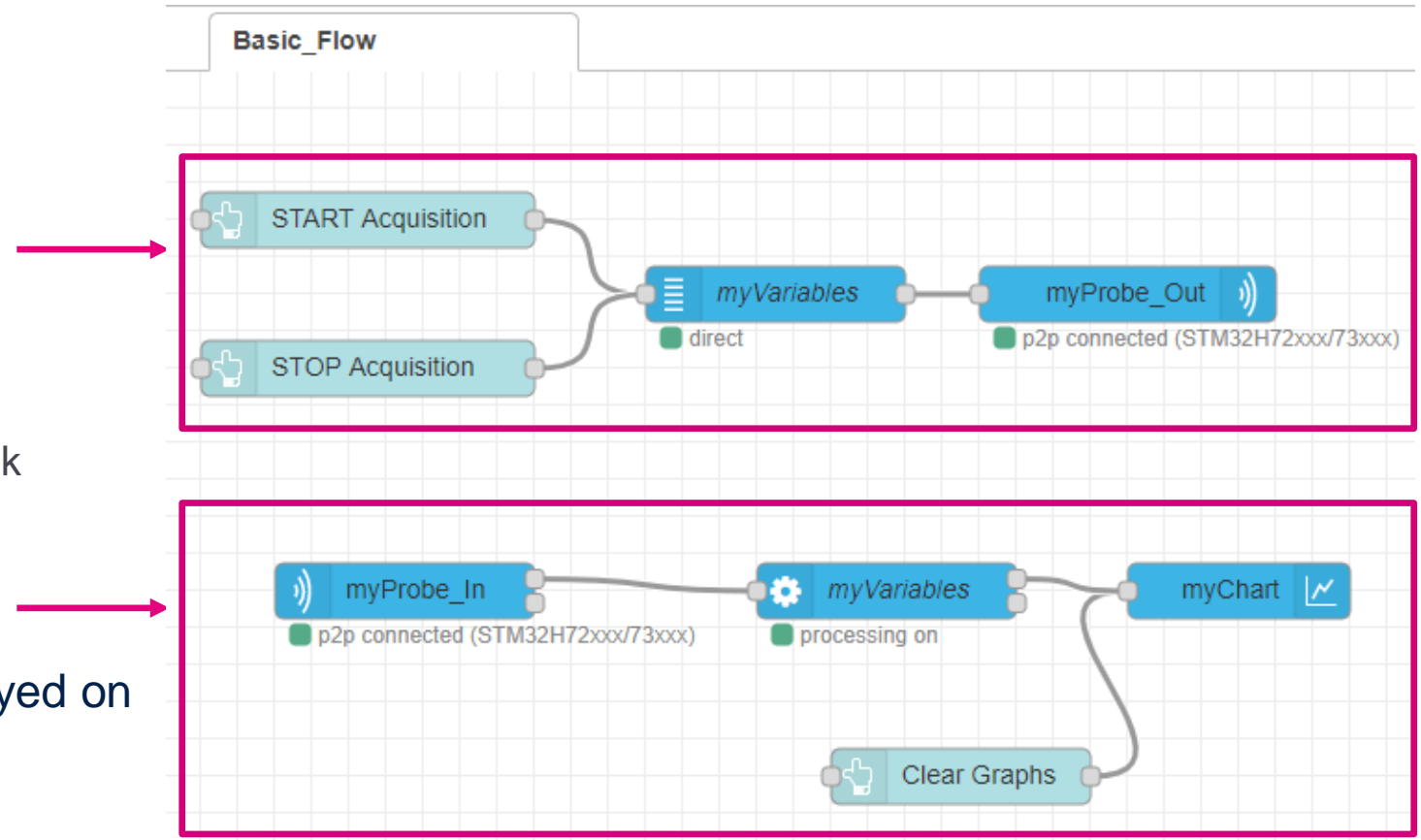
- "Read sequentially myVariables"

- STOP Acquisition

No more messages are sent on the ST-Link

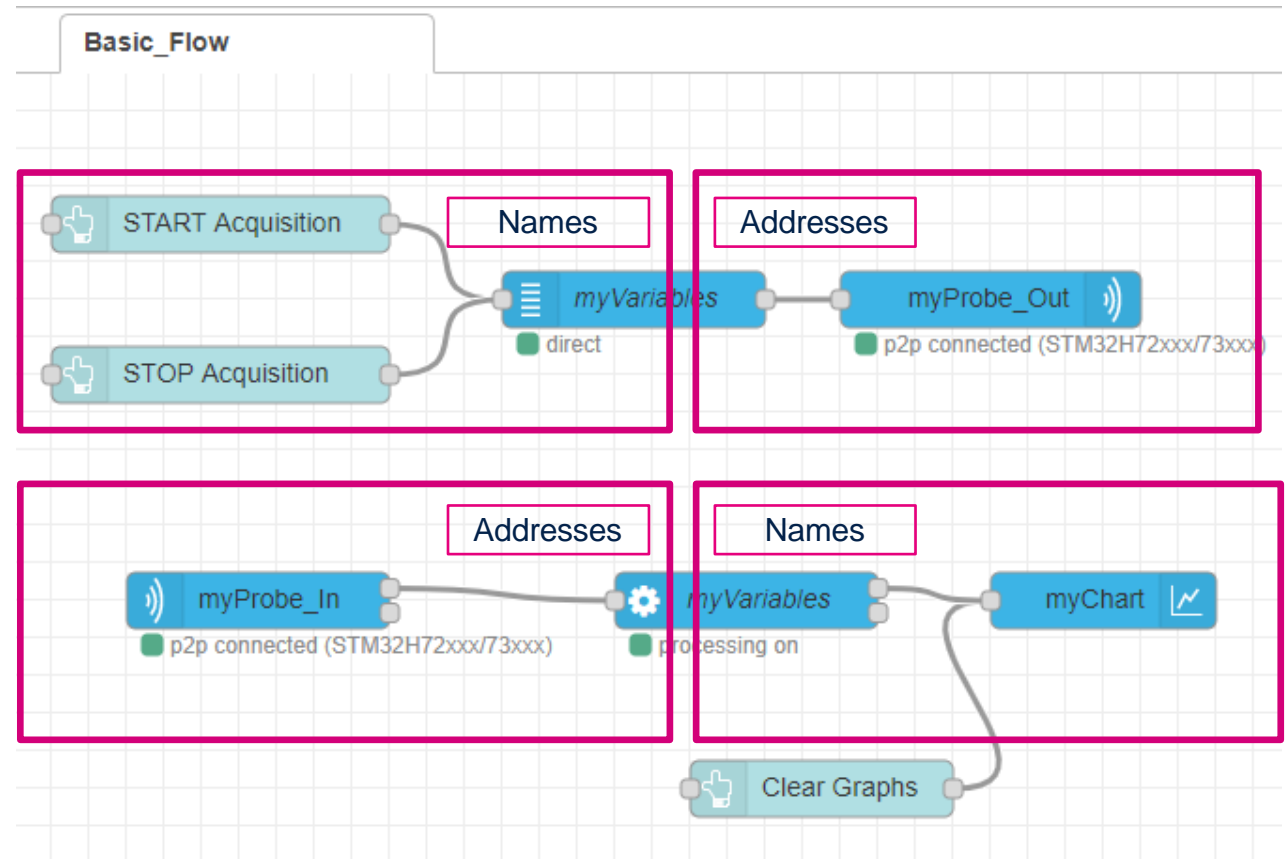
- One RX path from probe to PC

- MyVariables are received and displayed on the chart.



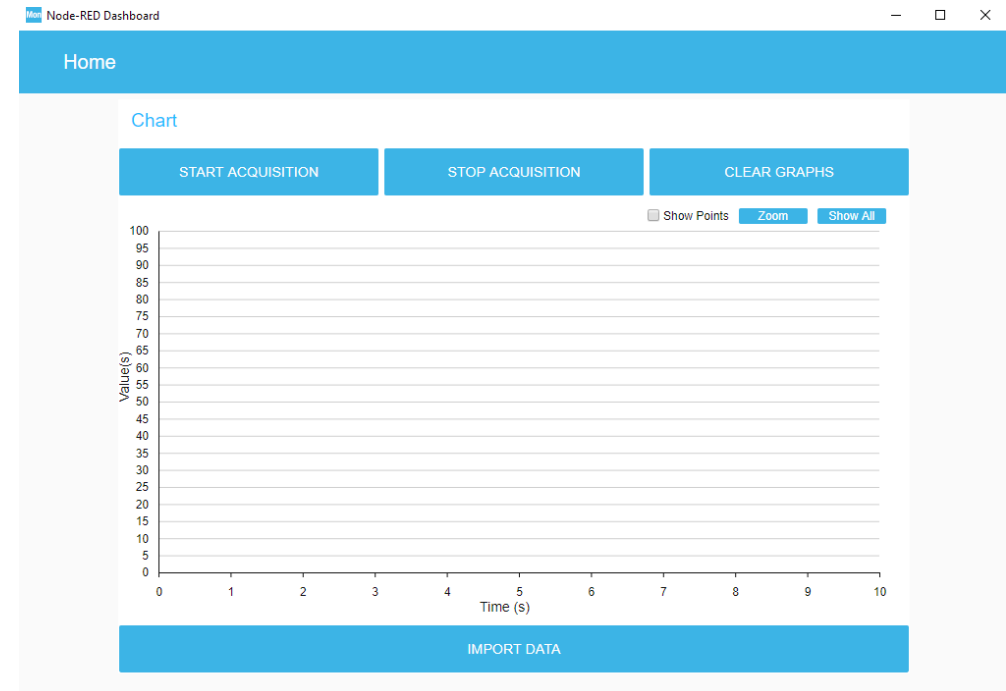
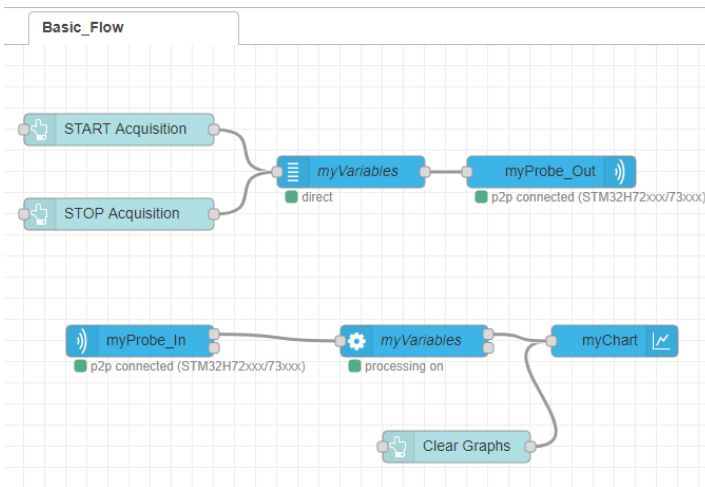
Name and address

- The MCU and its MCU debug block use addresses
- Humans prefer names
- PC software also!
 - The nodes “myVariables” manages the translation




The dashboard mode

- The widgets seen on the flow are displayed on the dashboard.
 - 3 Buttons
 - START, STOP, CLEAR
 - Chart



Read variables from MCU RAM memory

STM32 
CubeMonitor

Variable name

- Let's access to some variable of the firmware

- The code defines the type of "basicCounter"

- Unsigned 16-bit

```
70  
71 uint16_t basicCounter = 0;  
72
```


- The code increment the "basicCounter" every 500ms

```
134  
135 /* Infinite loop */  
136 /* USER CODE BEGIN WHILE */  
137 while (1)  
138 {  
139     /* USER CODE END WHILE */  
140  
141     /* USER CODE BEGIN 3 */  
142     HAL_Delay(500);  
143     HAL_GPIO_TogglePin(GPIOC, USER_LED1_Pin);  
144     basicCounter += 1;  
145
```

Get the names from elf

- The variable names are in the elf executable file

- generated during the project build in STM32CubeIDE
- STM32H735-DK.elf



```
arm-none-eabi-gcc -o "STM32H735G-DK.elf" @"objects.list" -mcpu=cortex-m7 -T"C:\STM32H735G-DK\STM32H735G-DK.ld"
Finished building target: STM32H735G-DK.elf

arm-none-eabi-size STM32H735G-DK.elf
arm-none-eabi-objdump -h -S STM32H735G-DK.elf > "STM32H735G-DK.list"
arm-none-eabi-objcopy -O binary STM32H735G-DK.elf "STM32H735G-DK.bin"
  text    data    bss     dec     hex filename
 7692     40    1568    9300    2454 STM32H735G-DK.elf
Finished building: default.size.stdout

Finished building: STM32H735G-DK.list

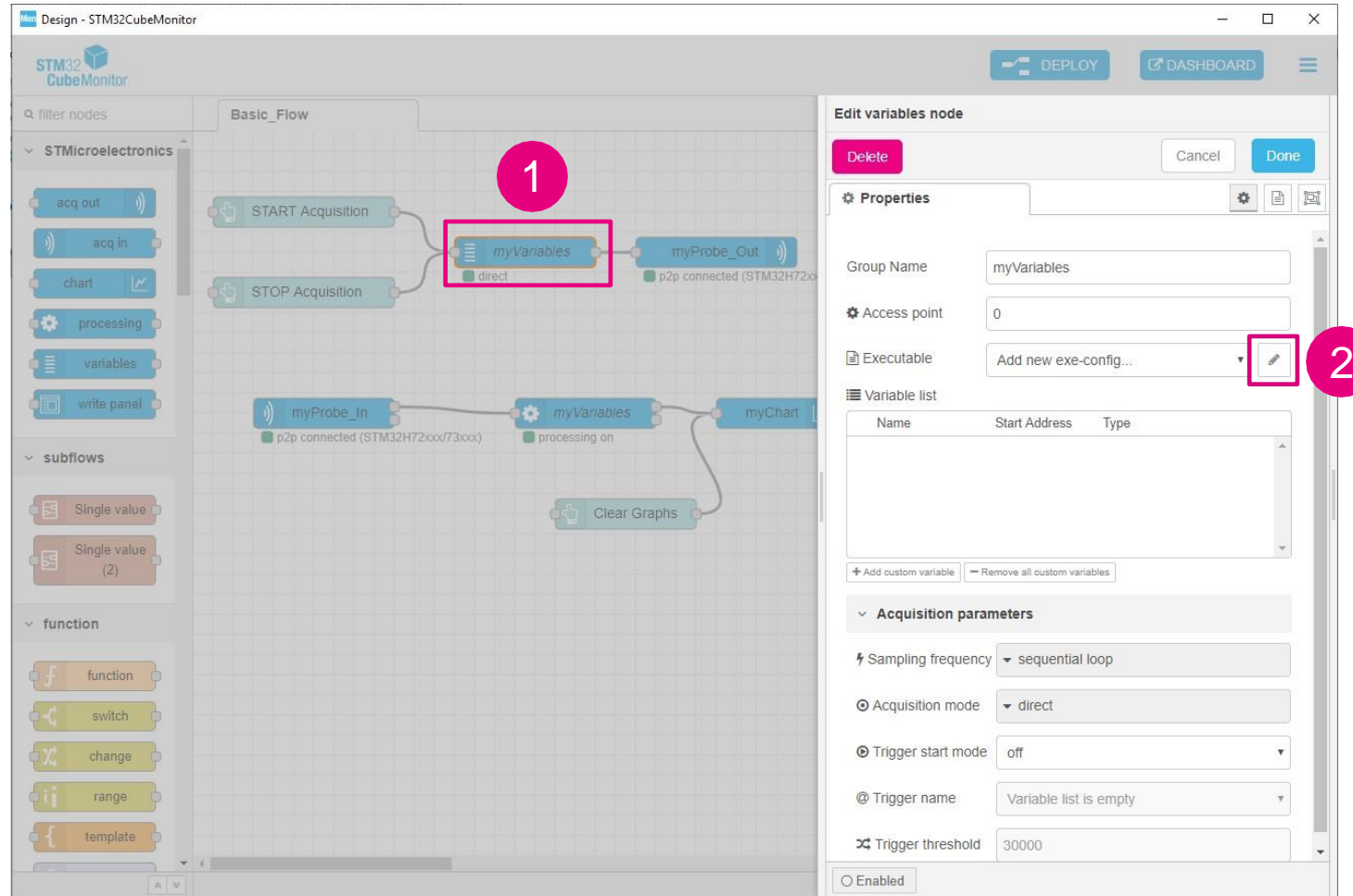
Finished building: STM32H735G-DK.bin

00:05:51 Build Finished. 0 errors, 2 warnings. (took 17s.365ms)
```

- .out and .axf extensions also supported

Get the names from elf

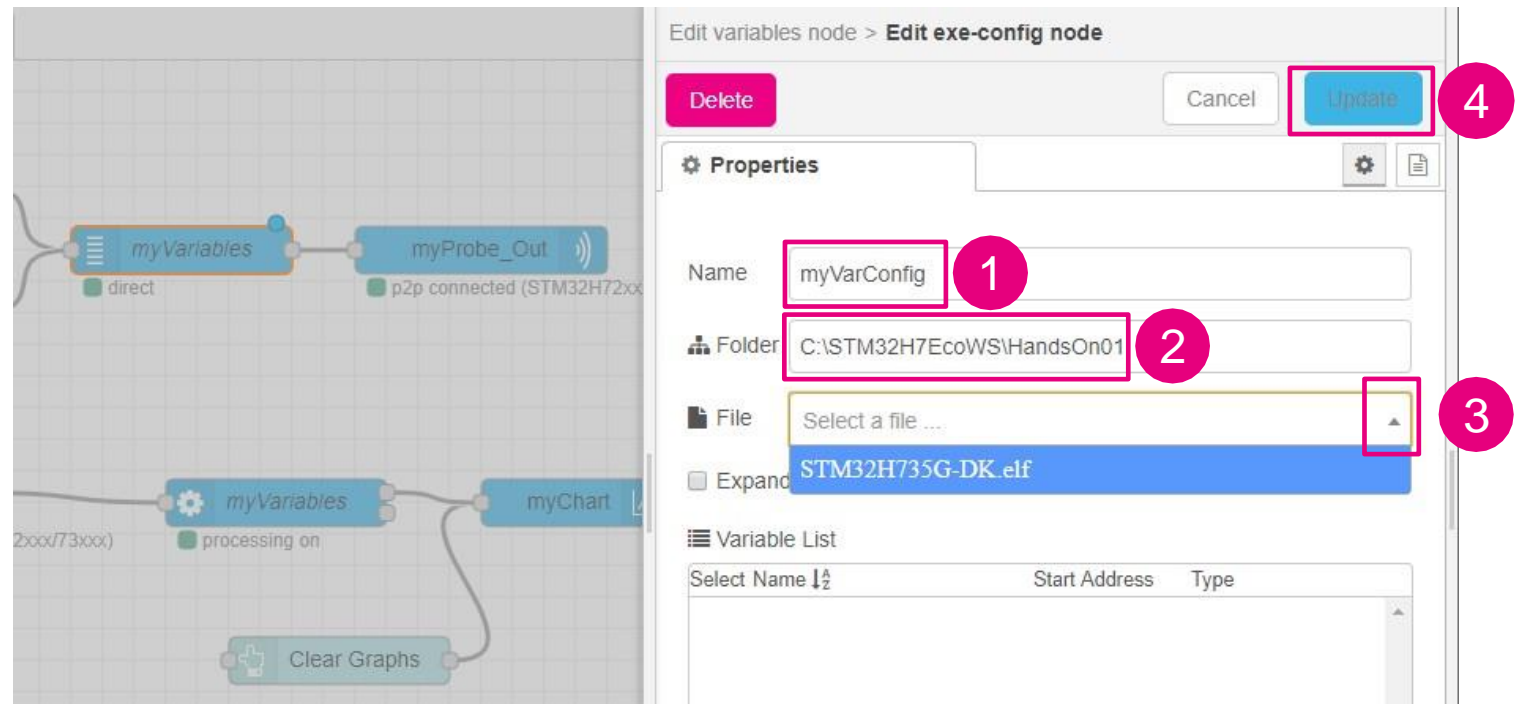
- Add new exe-config...
 1. Double-click on “myVariables”
 - The variables node properties are displayed
 2. Click on the pen of “Executable”



Get the names from elf

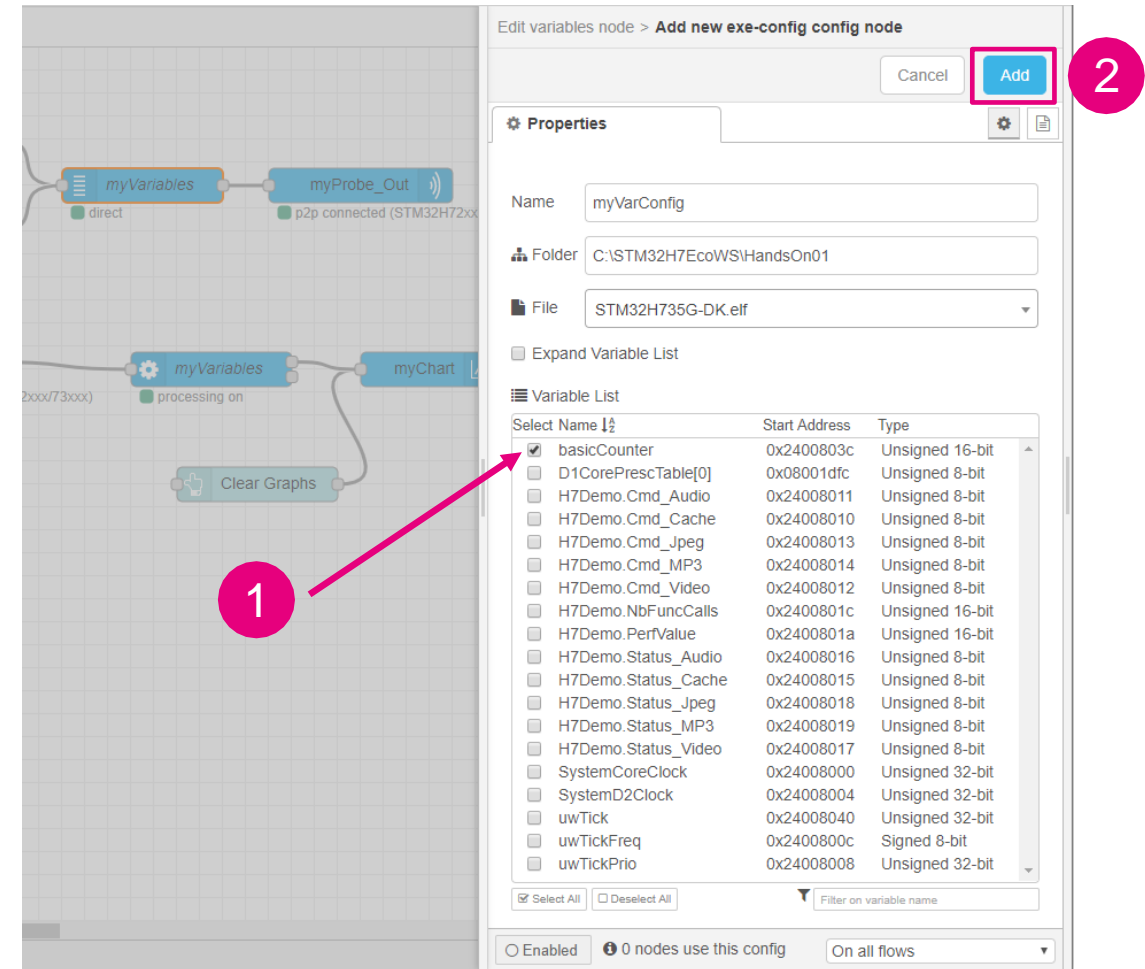
- Create the exe-config

1. Give a name to the configuration
 - C:\STM32H7EcoWS\HandsOn_01
2. Fill "Folder" with the folder containing the elf file
 - C:\STM32H7EcoWS\HandsOn_01
3. Select the elf file from the drop-down list
 - STM32H735G-DK.elf
4. Click in Update



Get the names from elf

- Select the variables in the “Variable List”
 1. Select “basicCounter”
 2. Click on “Add”
 - Adds the configuration to the “myVariables”

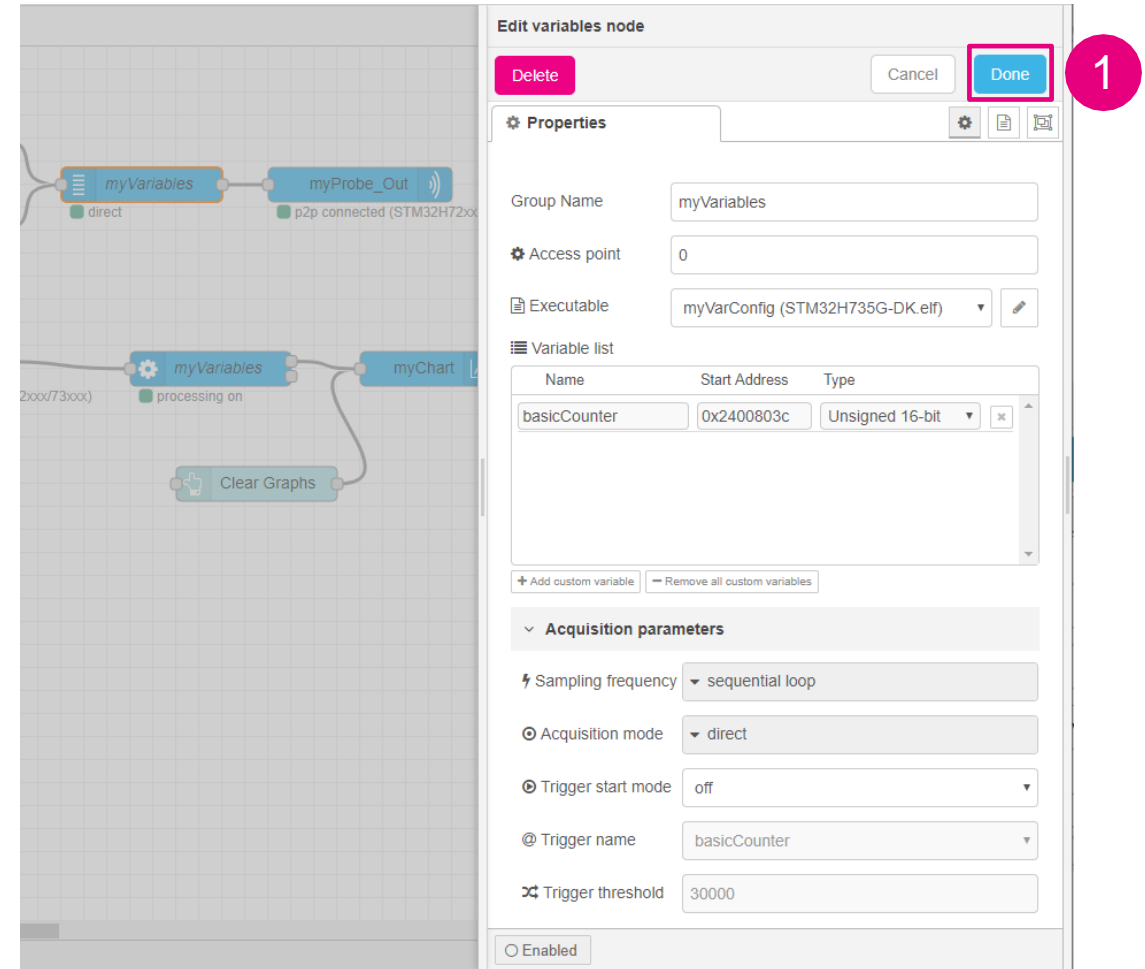


Get the names from elf

- The variable list is configured

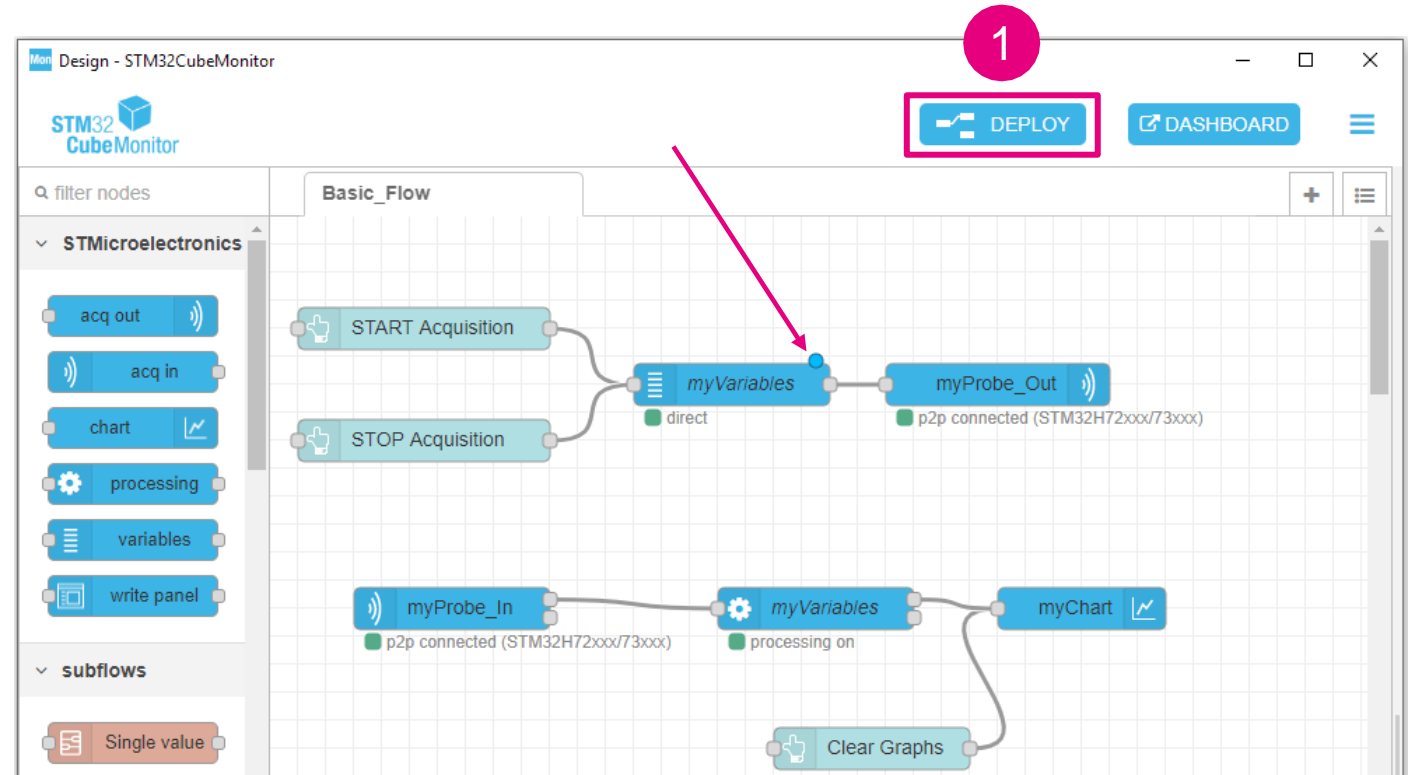
- Name
 - basicCounter
- Address
 - 0x2400803C
- Type
 - Unsigned 16-bit

1. Click on “Done”



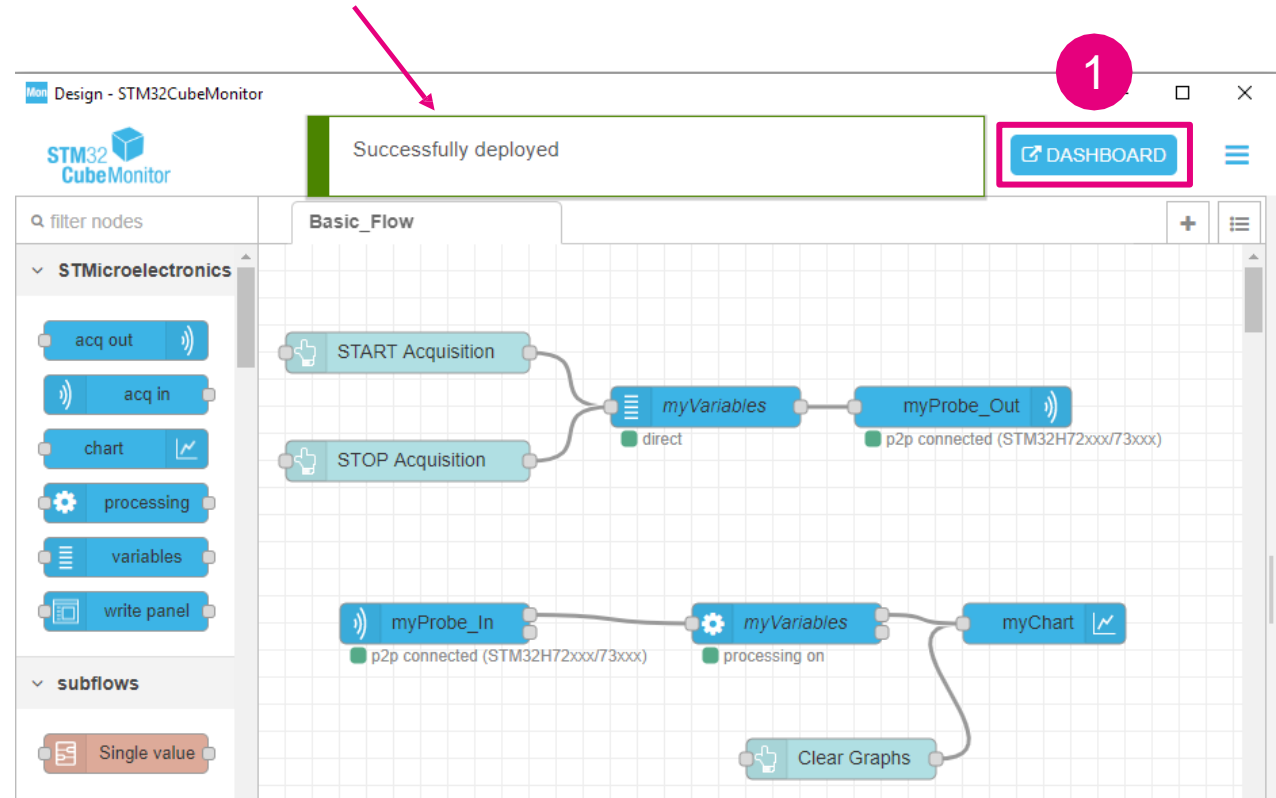
Get the names from elf

- Save the flow
 - A blue point on “myVariables” shows that the configuration has been modified.
1. Click on “Deploy”



Monitor and visualize read in MCU RAM memory

- Launch the dashboard
 - “Successfully deployed” is displayed
- 1. Click on “Dashboard”

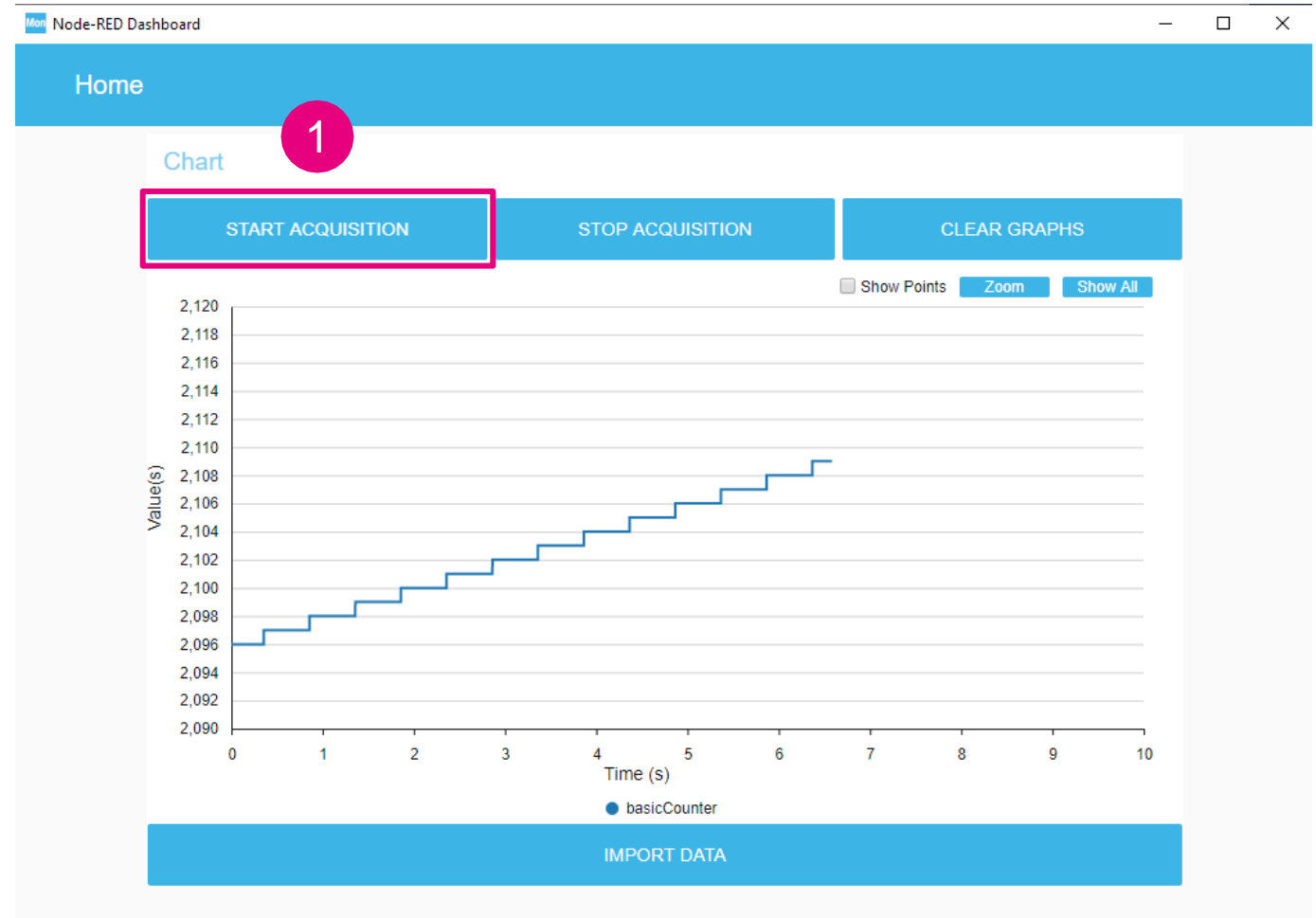


Monitor and visualize read in MCU RAM memory

- Launch the acquisition

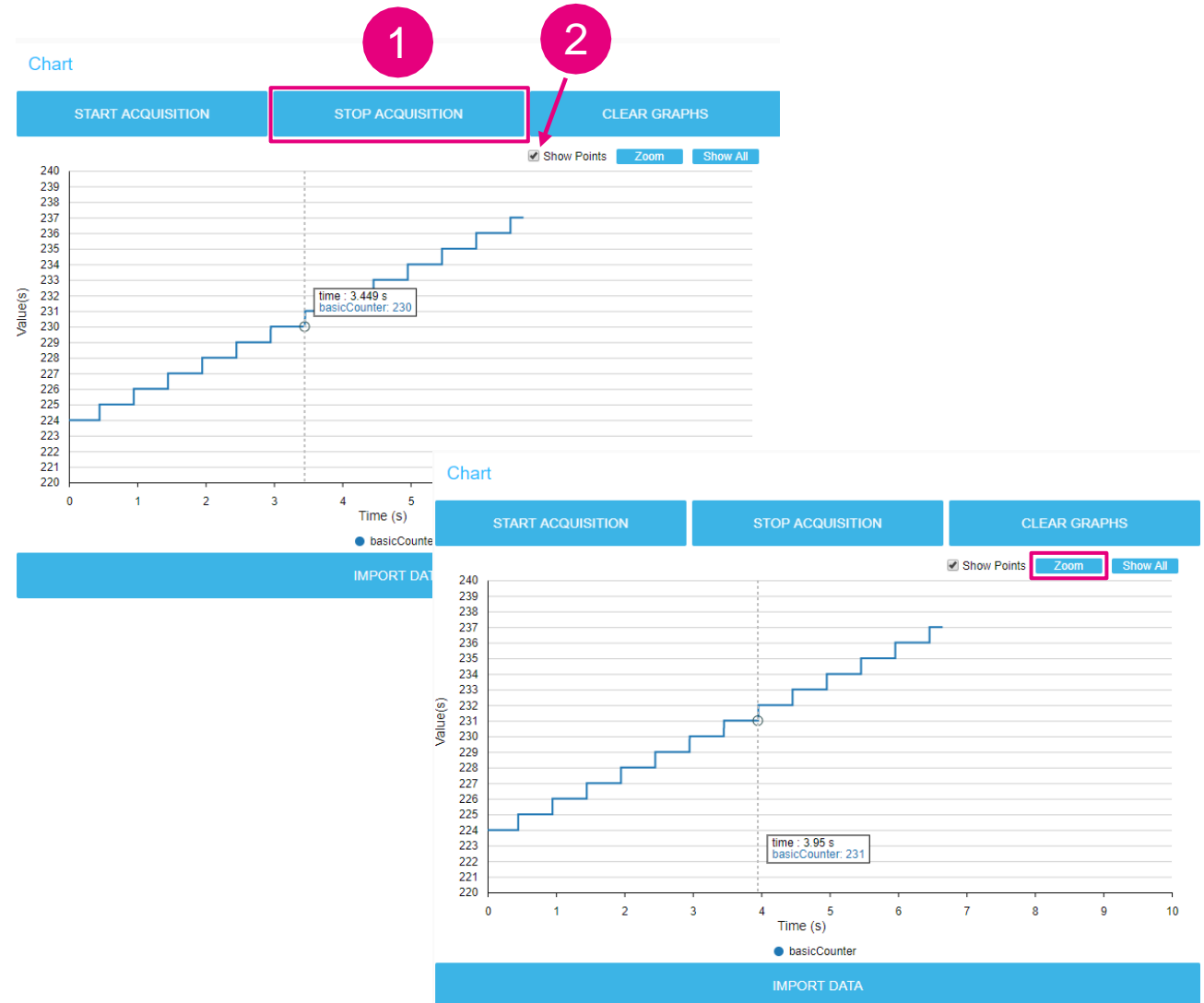
1. Click on “STARTACQUISITION”

- “basicCounter” is sequentially read from the MCU RAM memory
- “basicCounter” is regularly incremented




Monitor and visualize read in MCU RAM memory

- Stop reading the variable
 1. Click on “STOP ACQUISITION”
- The times and values can be visualized
 2. Click the “Show Points” checkbox
 - ‘basicCounter’ is incremented by 1
 - “basicCounter” is incremented every 500ms
- Use the zoom for more precision
 - Zoom or Brush is selectable.
 - In zoom mode, it's possible to zoom on the curve using the mouse to select a zone.

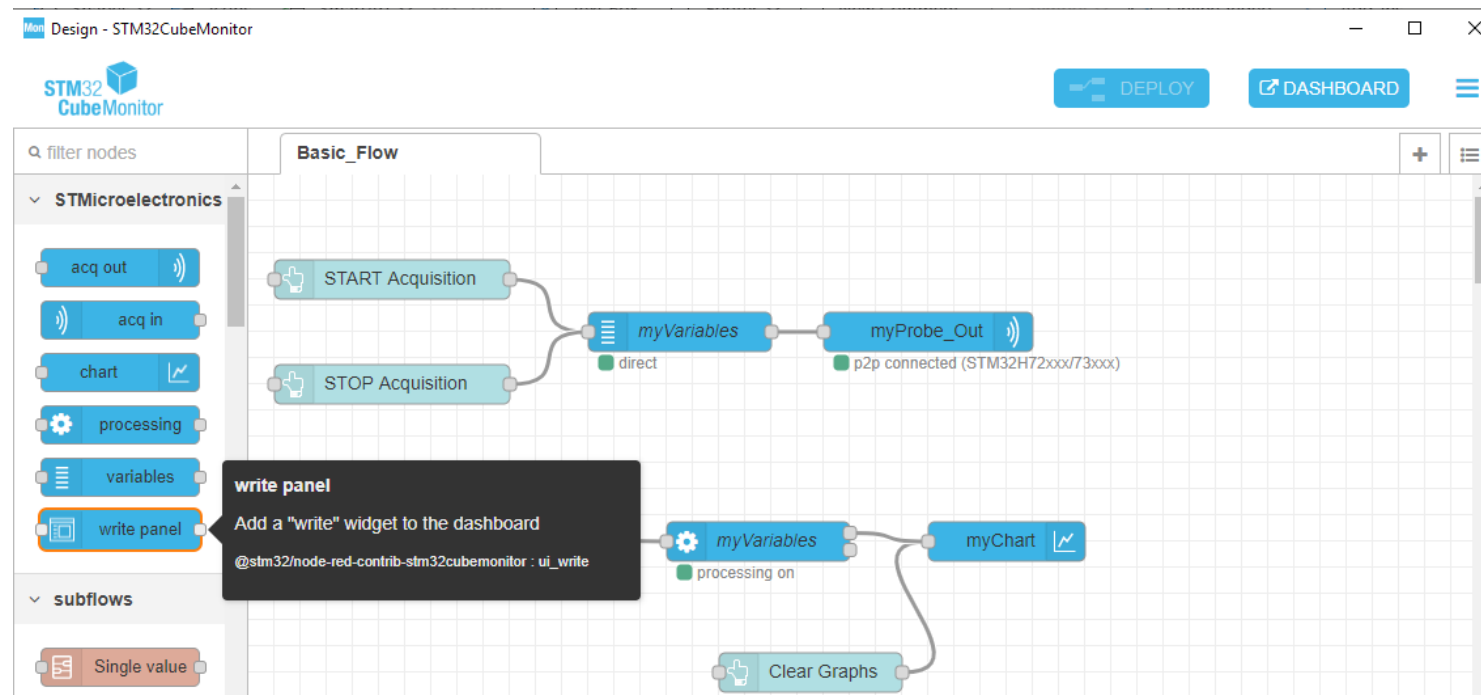


Write variables to MCU RAM memory

STM32 
CubeMonitor

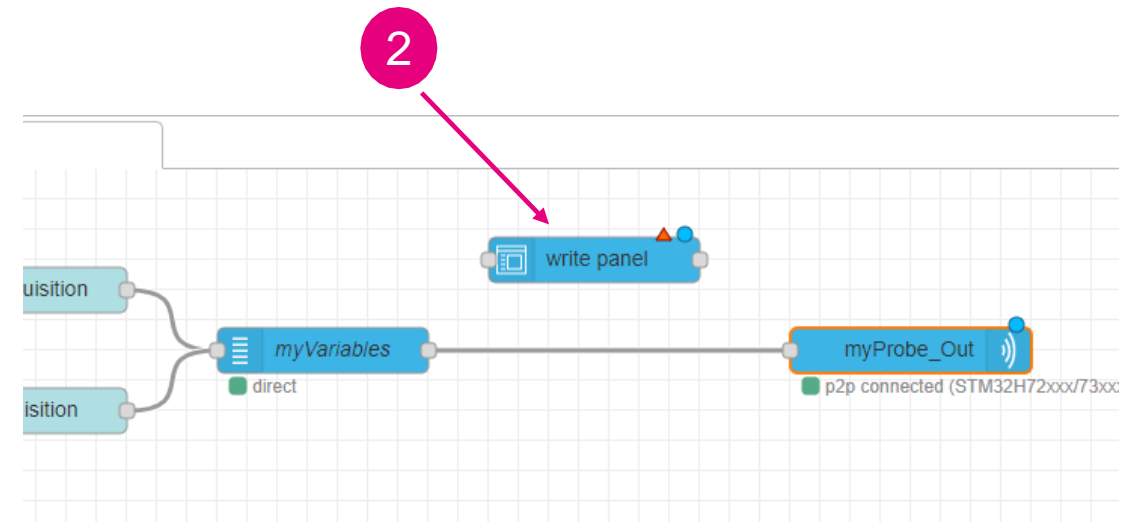
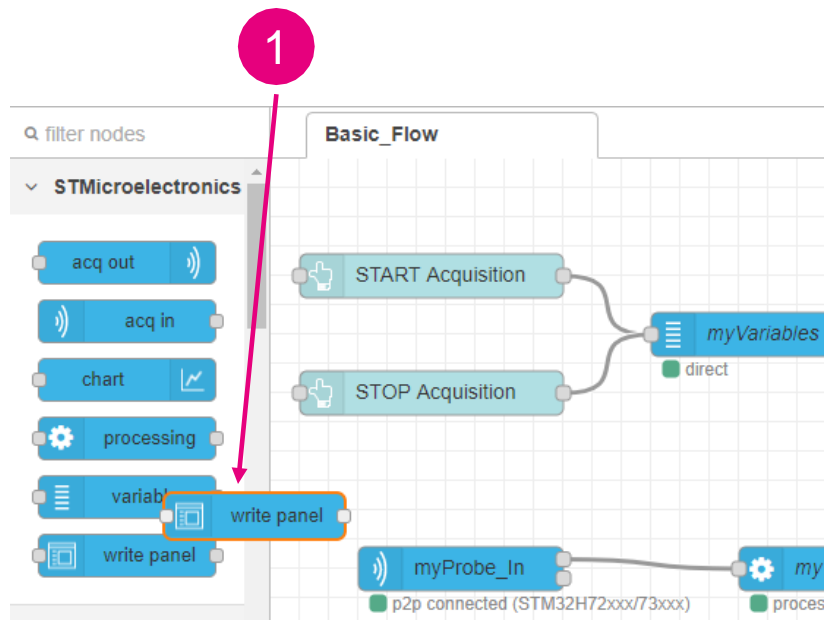
Add a widget

- Currently, no way to write a value to a variable.
 - The “write panel” widget in the node palette will be used



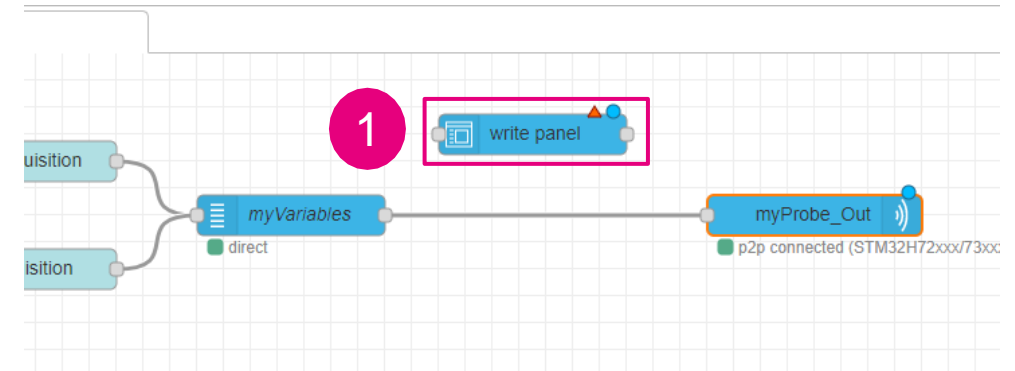
Add a widget

- Drag & drop the “write panel” in the flow editor
 1. Click on the “write panel” node and hold the button down
 2. Drag the “write panel” node to the flow editor
 - “myprobe_Out” can be dragged a bit further



Add a widget

- Set the “write panel” properties
 - A red triangle is visible on the “write panel” node
 - Properties are not configured
 - The “write panel” widget will not be added to the dashboard!

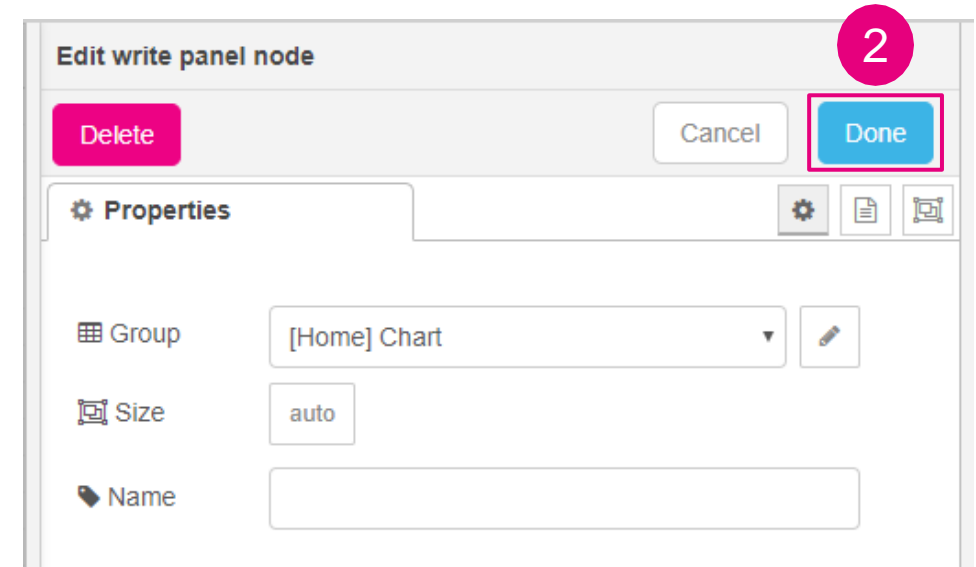


1. Double-click on the “write panel”

- “Group” has been set automatically as only one exists
 - UI group on the dashboard = Chart
 - Tab = Home
- Several groups can be created on the dashboard
- Several flows can be opened in the flow editor
 - The dashboard will then have several tabs with several UIs

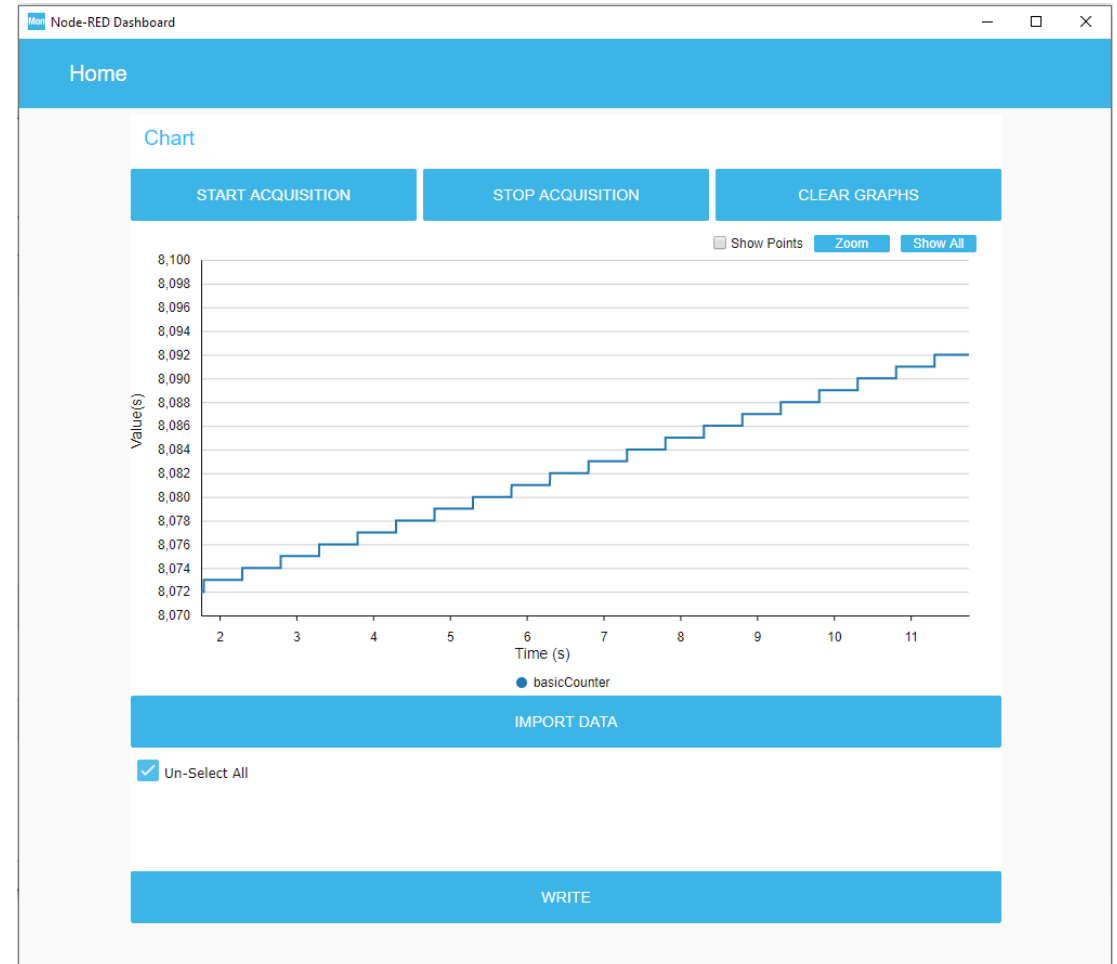
2. Click on “Done”

- The name is optional



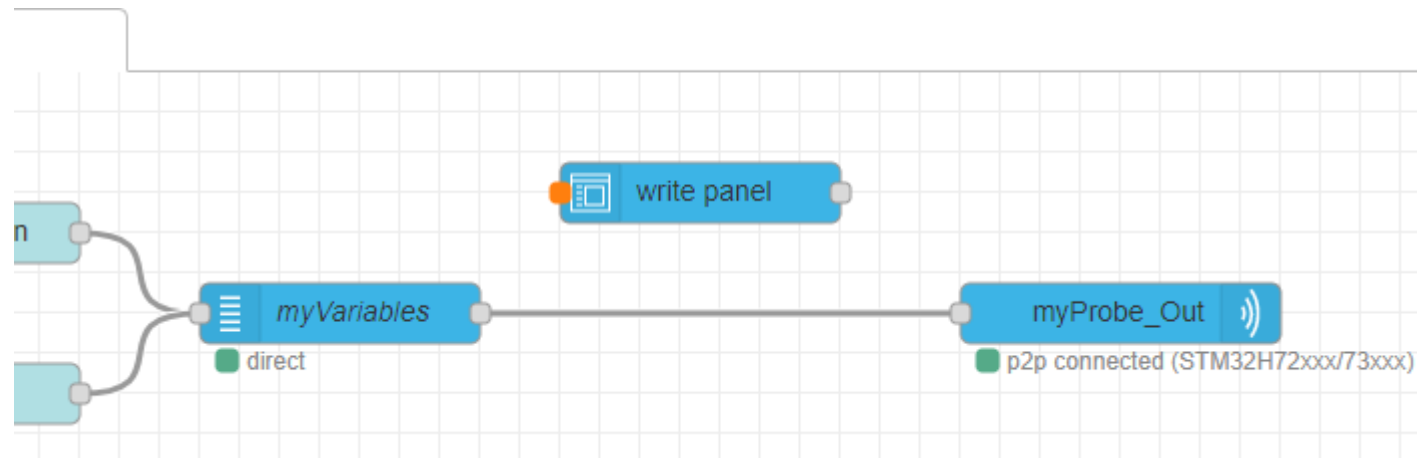
Add a widget – done!

- Launch the dashboard
 1. Click on “DEPLOY”
 2. Click on “Dashboard”
- The write panel widget has been added
 - Visual programming
 - A drag & drop and some clicks
- Let's add a variable in the panel



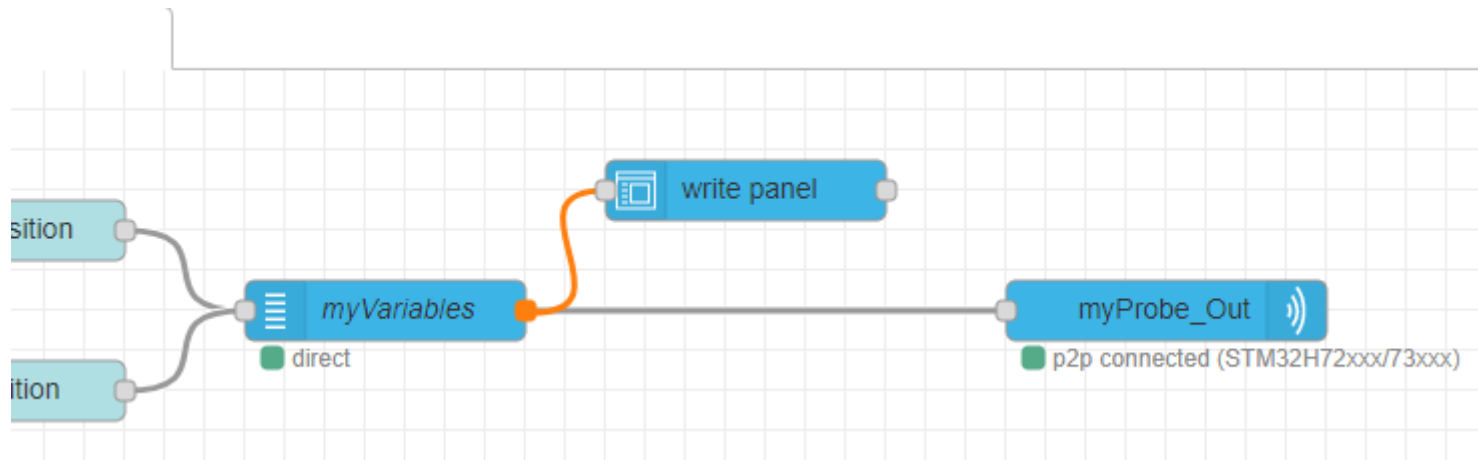
Add a variable

- Add a link with “myVariables” to populate the write panel
 1. Put the mouse on the input of the “write panel” node
 - The input point is displayed in orange



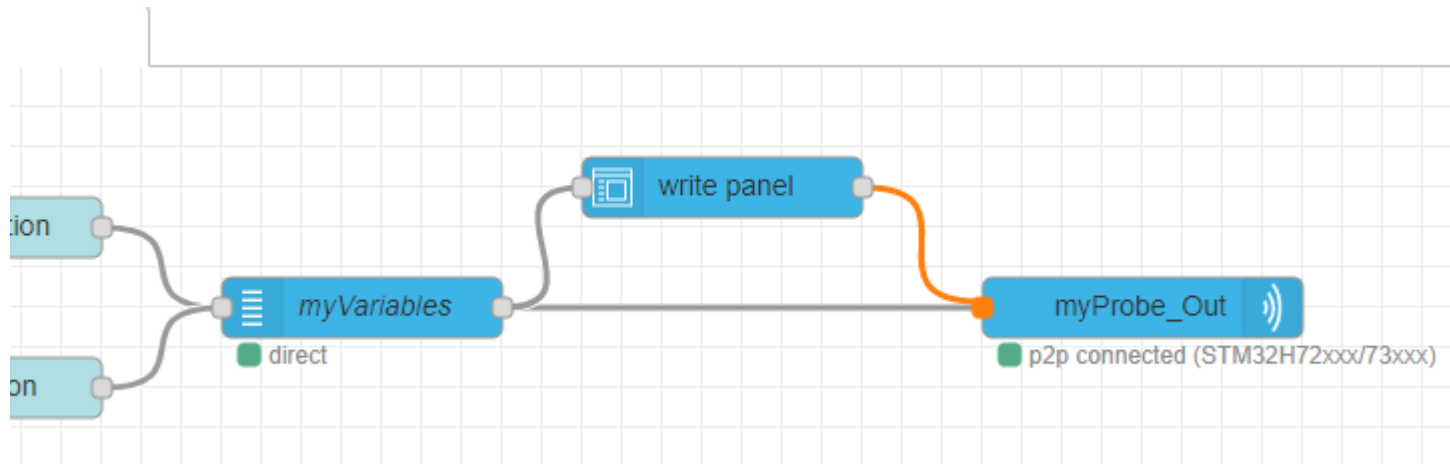
Add a variable

- Add a link with “myVariables” to populate the write panel
 2. Click and drag to the output of “myVariable”
 - Hold the mouse button down until the output point is displayed in orange
 - The link is done
 - The list of variables will appear in the write panel as soon as the “STARTACQUISITION” button is clicked



Activate the “WRITE” button

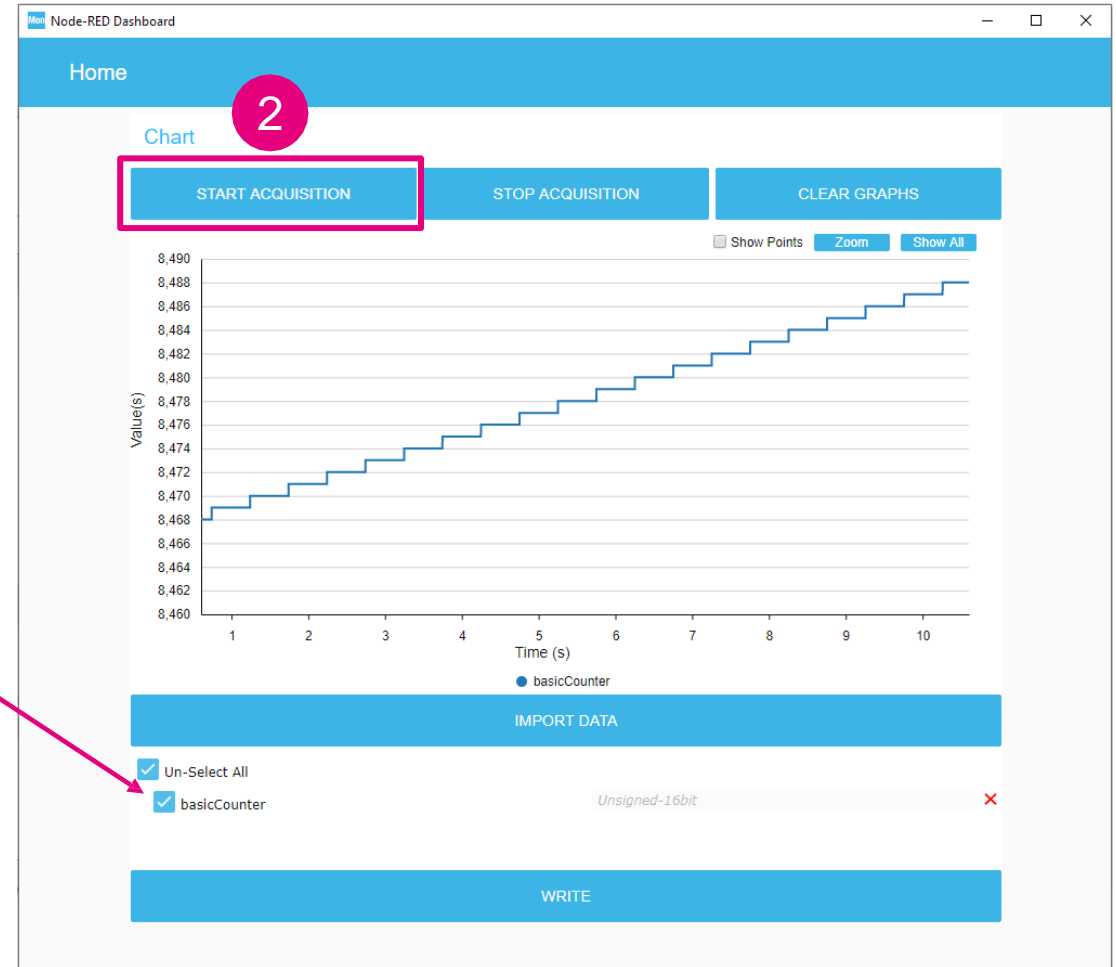
- Add a link with “myProbe_Out”
 1. Drag a link between the output of the “write panel” and the input of “myProbe_Out”
 - The link is done
 - A click on the “WRITE” button will produce a message sent on the ST-LINK:
 - “Write the selected variables”



Monitor and visualize write in MCU RAM memory

- Launch the acquisition
 1. Click on “DEPLOY”, then “DASHBOARD”
 2. Click on “START ACQUISITION”

➤ “basicCounter” is in the write panel list



Monitor and visualize write in MCU RAM memory

- Write to the basicCounter variable

1. Write a value in the text box beside “basicCounter” name
2. Click on “WRITE”

➤ “basicCounter” value is changed to the written number.



Read or Write a peripheral register

STM32 
CubeMonitor

Read/write to an address

- **All the MCU address** space is accessible.
- But some addresses don't have a variable name in the elf file.
 - Registers
- How to read/write an address without a name?
 - Let's take the example of an I/O port

- The GPIOC base address is 0x58020800
- The ODR register has the offset 0x14
 - The lowest byte of ODR register is at 0x58020814

11.4.6 GPIO port output data register (GPIOx_ODR) (x = A to H, J, K)

Address offset: 0x14

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.	Res.
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW	rW

Table 7. Register boundary addresses⁽¹⁾

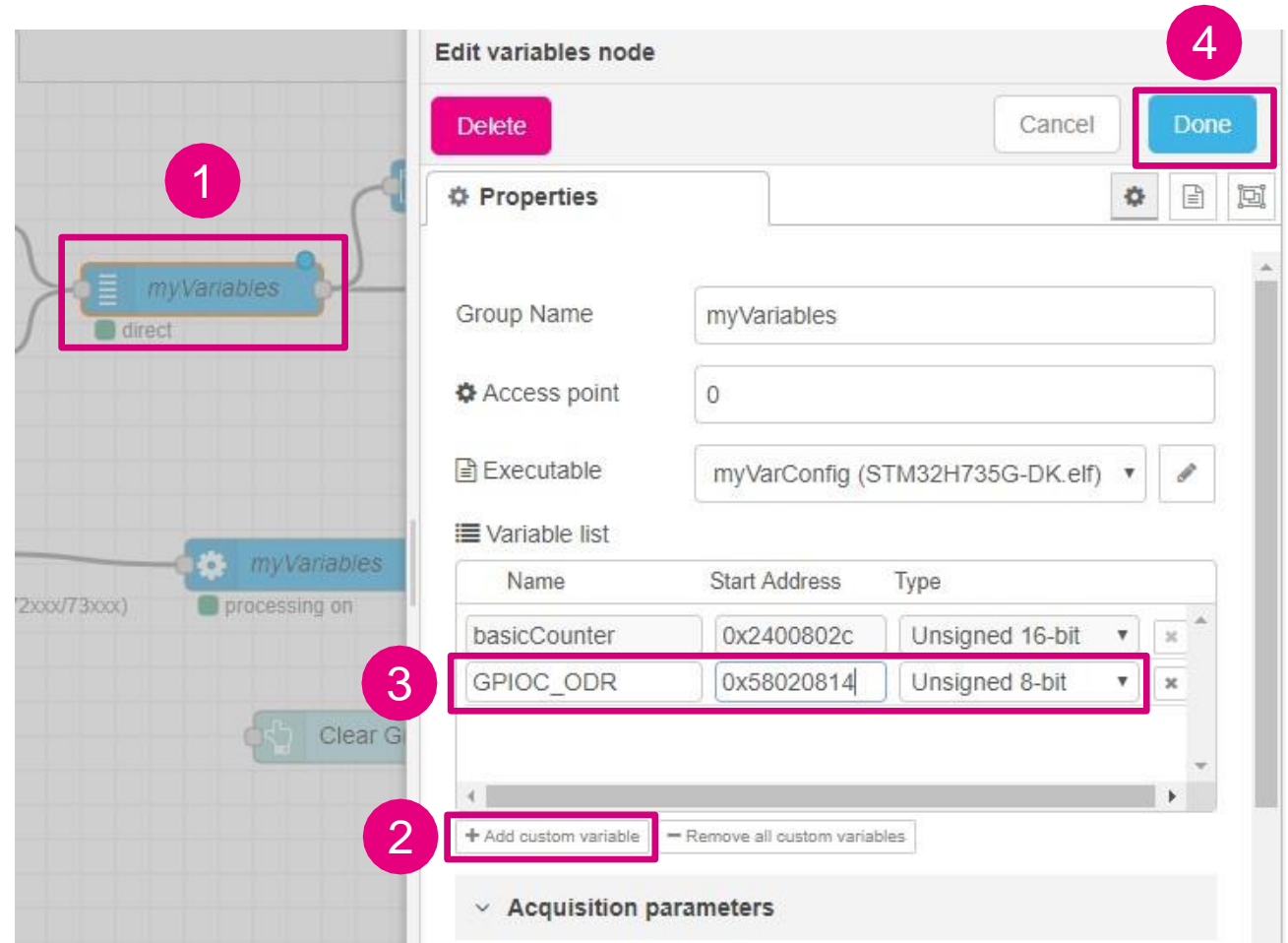
Boundary address	Peripheral	Bus	Register map
0x58026400 - 0x580267FF	HSEM	AHB4 (D3)	Section 10.4: HSEM registers
0x58026000 - 0x580263FF	ADC3		Section 28.7: ADC common registers
0x58025800 - 0x58025BFF	DMAMUX2		Section 17.6: DMAMUX registers
0x58025400 - 0x580257FF	BDMA		Section 16.6: BDMA registers
0x58024C00 - 0x58024FFF	CRC		Section 21.4: CRC registers
0x58024800 - 0x58024BFF	PWR		Section 6.8: PWR registers
0x58024400 - 0x580247FF	RCC		Section 8.7: RCC registers
0x58022800 - 0x58022BFF	GPIOK		Section 11.4: GPIO registers
0x58022400 - 0x580227FF	GPIOJ		Section 11.4: GPIO registers
0x58021C00 - 0x58021FFF	GPIOH		Section 11.4: GPIO registers
0x58021800 - 0x58021BFF	GPIOG		Section 11.4: GPIO registers
0x58021400 - 0x580217FF	GPIOF		Section 11.4: GPIO registers
0x58021000 - 0x580213FF	GPIOE		Section 11.4: GPIO registers
0x58020C00 - 0x58020FFF	GIOD		Section 11.4: GPIO registers
0x58020800 - 0x58020BFF	GPIOC		Section 11.4: GPIO registers
0x58020400 - 0x580207FF	GPIOB		Section 11.4: GPIO registers
0x58020000 - 0x580203FF	GPIOA		Section 11.4: GPIO registers

Add the address

- Add the address in the variable list

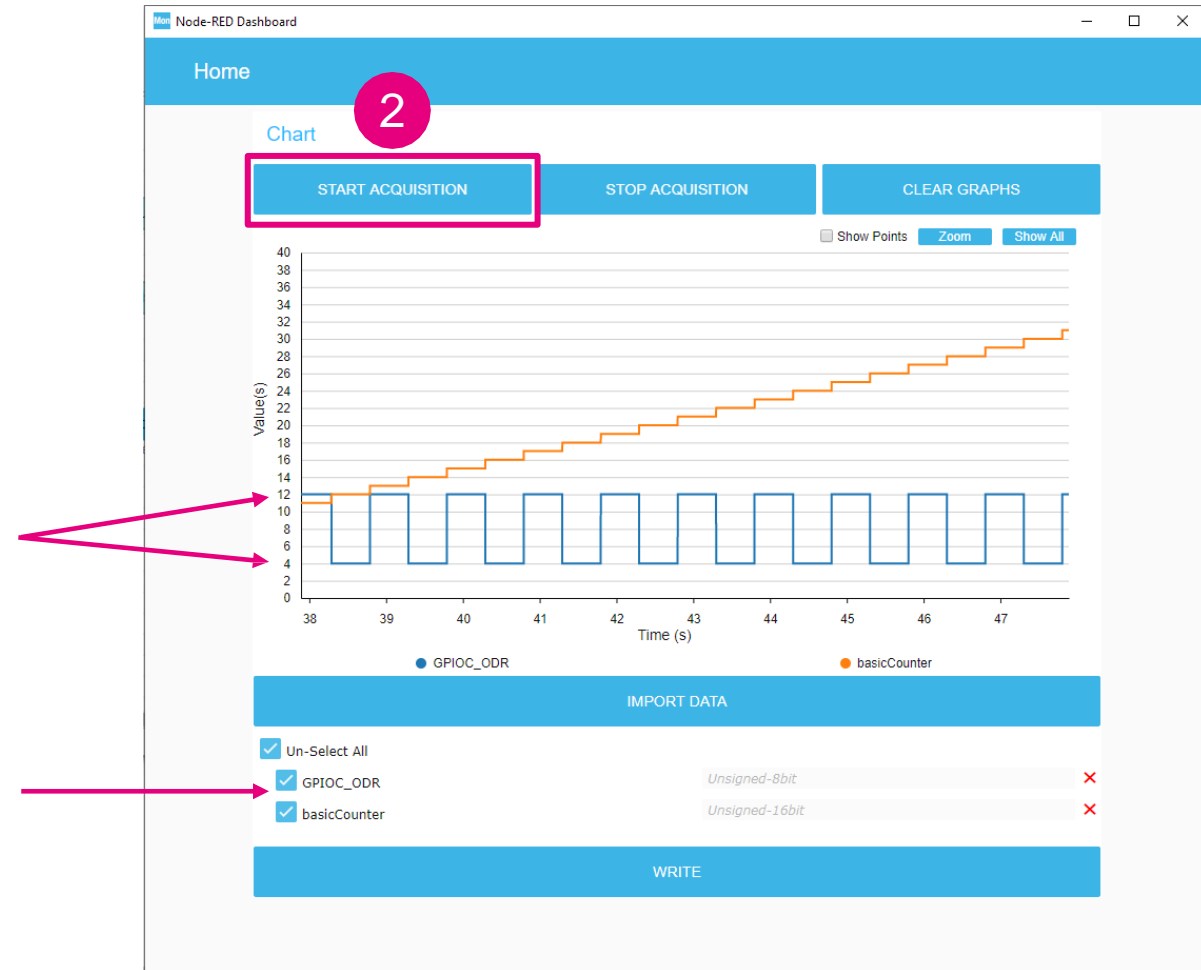
1. Open “myVariables” properties
2. Click on “Add custom variable”
3. Fill the fields
 - Name
 - GPIOC_ODR
 - Address
 - 0x58020814
 - Type
 - Unsigned 8-bit
4. Click on “Done”

- The name is mandatory
 - For humans
 - For STM32CubeMonitor software



Monitor and visualize Read in MCU register fixed address

- Launch the acquisition
 1. Click on “DEPLOY”, then “DASHBOARD”
 2. Click on “START ACQUISITION”
 - “basicCounter” is displayed
 - The GPIOC_ODR is also displayed



We have seen...

- How to **configure** the flow
 - How to connect to the STM32 MCU board through ST-Link
 - How to get the variable names from the executable file
- How to **access** to the STM32 memory
 - Read / Write using the variable name
 - Read / Write using the address
 - All the STM32 address space is accessible
- How to add a **widget** to the dashboard
 - Drag & drop on the flow editor
 - Insert in the flow
 - Make the connection with previous and next node

STMH72x/3x Demo flow

STM32 
CubeMonitor

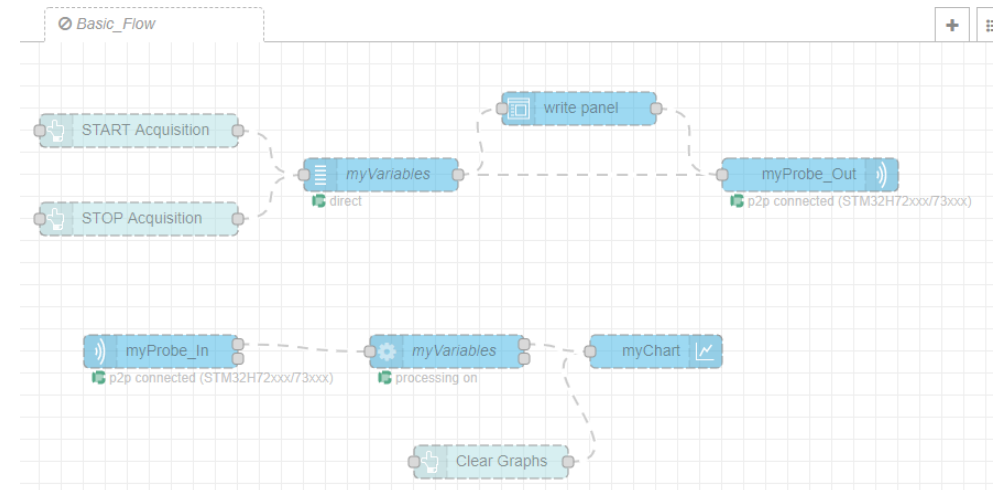
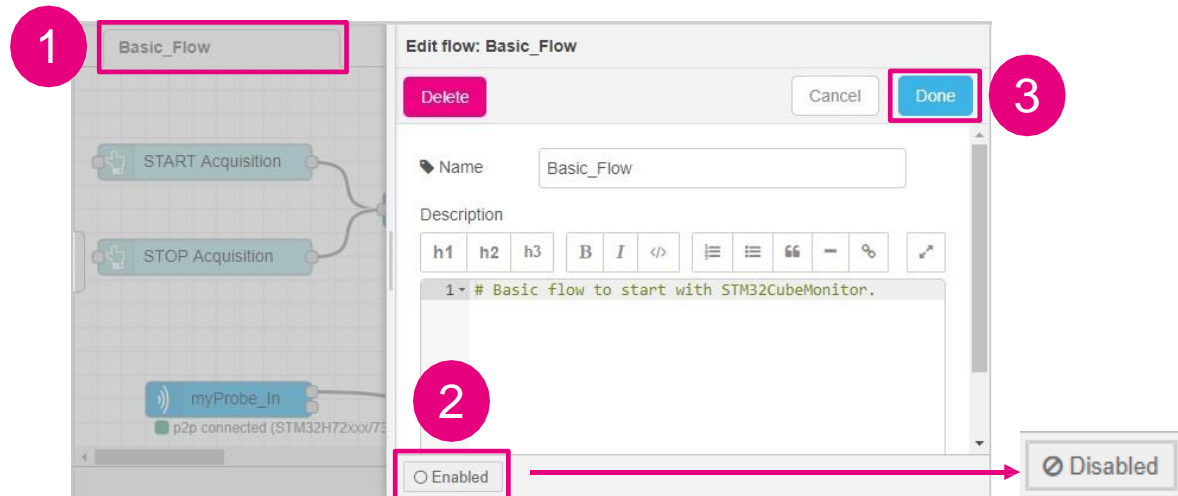
STMH72X/3X Demo flow

- An example of customization
 - More widgets
 - But same concepts
- Let's import the flow

Disable a flow

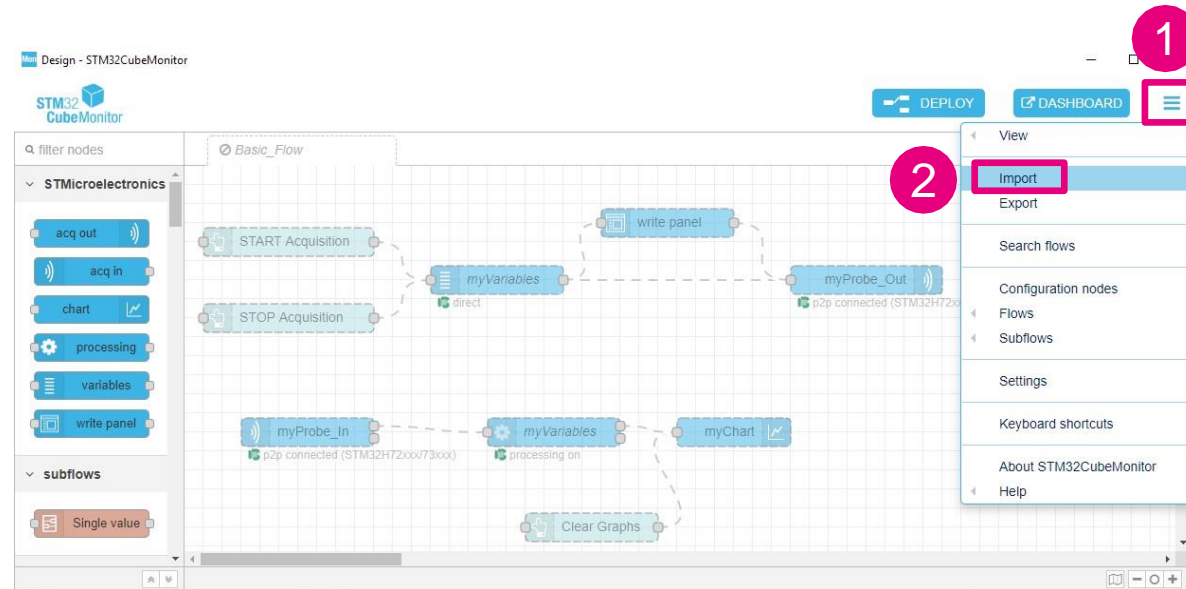
- Disable the “Basic_Flow”

1. Double-click on the “Basic_Flow” tab
 - It is changed to disabled
2. Click in Enabled
 - The “Basic_Flow” tab remains in the flow editor but no “Home” tab will be accessible in the dashboard.
3. Click on Done



Import a flow

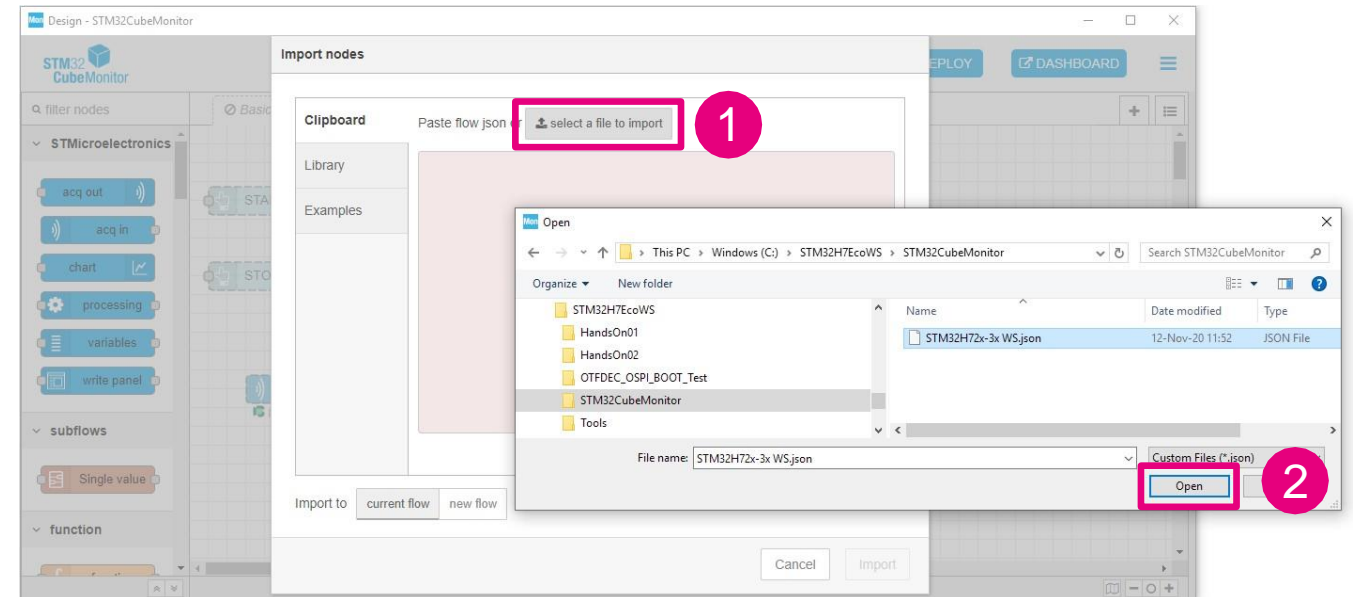
- **Please click on “DEPLOY”**
 - If not closed and acquisition not stopped, the dashboard is disabled and the ST-Link is relaxed.
- Open the menu and select import
 1. Click on the menu
 2. Click on “Import”



Import a flow

- Import from a file

1. Click on “select a file to import”
2. Open the STMH72x/3X Demo flow
 - In C:\STM32H7EcoWS\STM32CubeMonitor
 - STM32H72x-3x WS.json



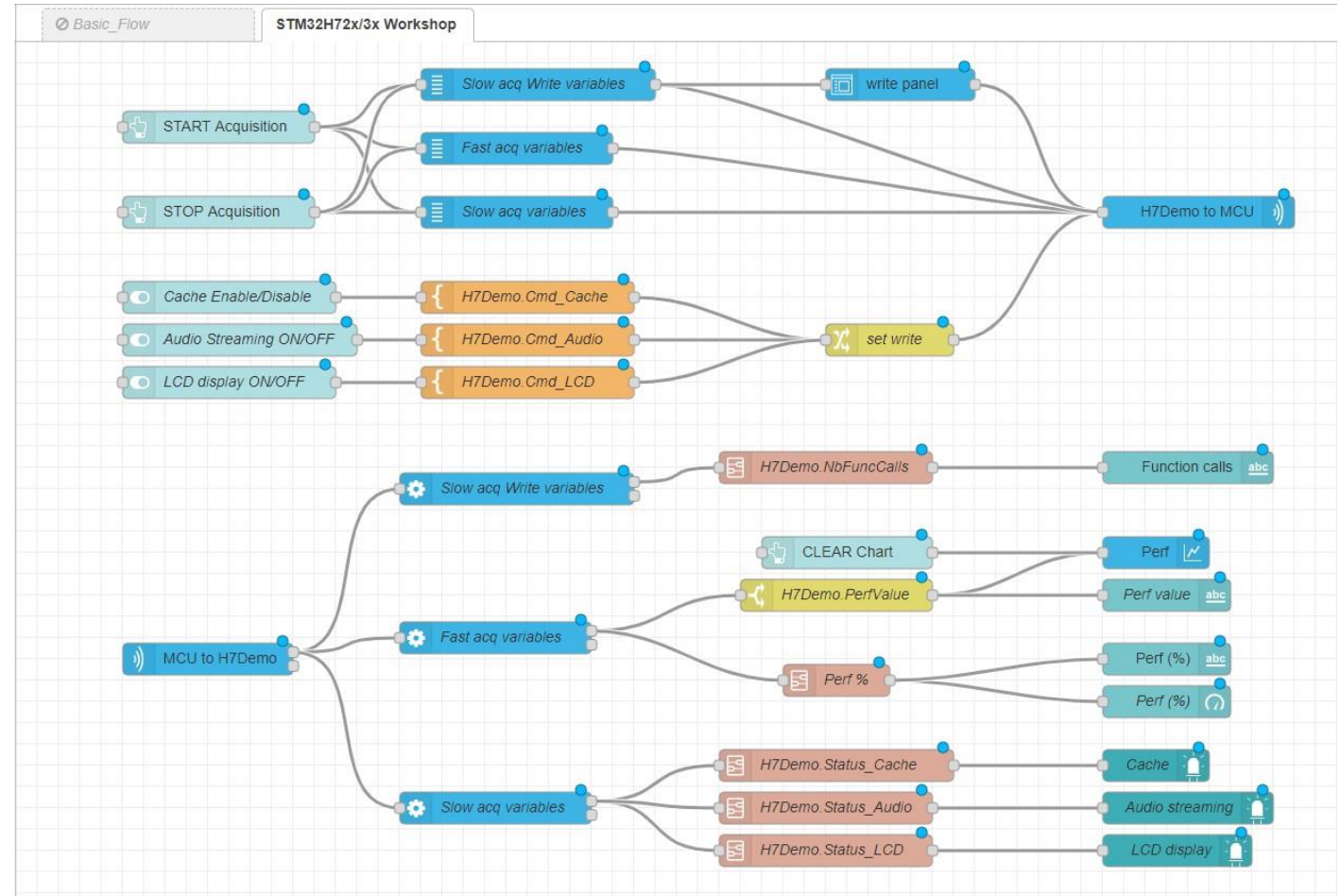
1

1. Click on “Import”



Import a flow

- The STMH72x/3X Demo flow is now visible in the flow editor
- More nodes, more links, but...



Same flow structure

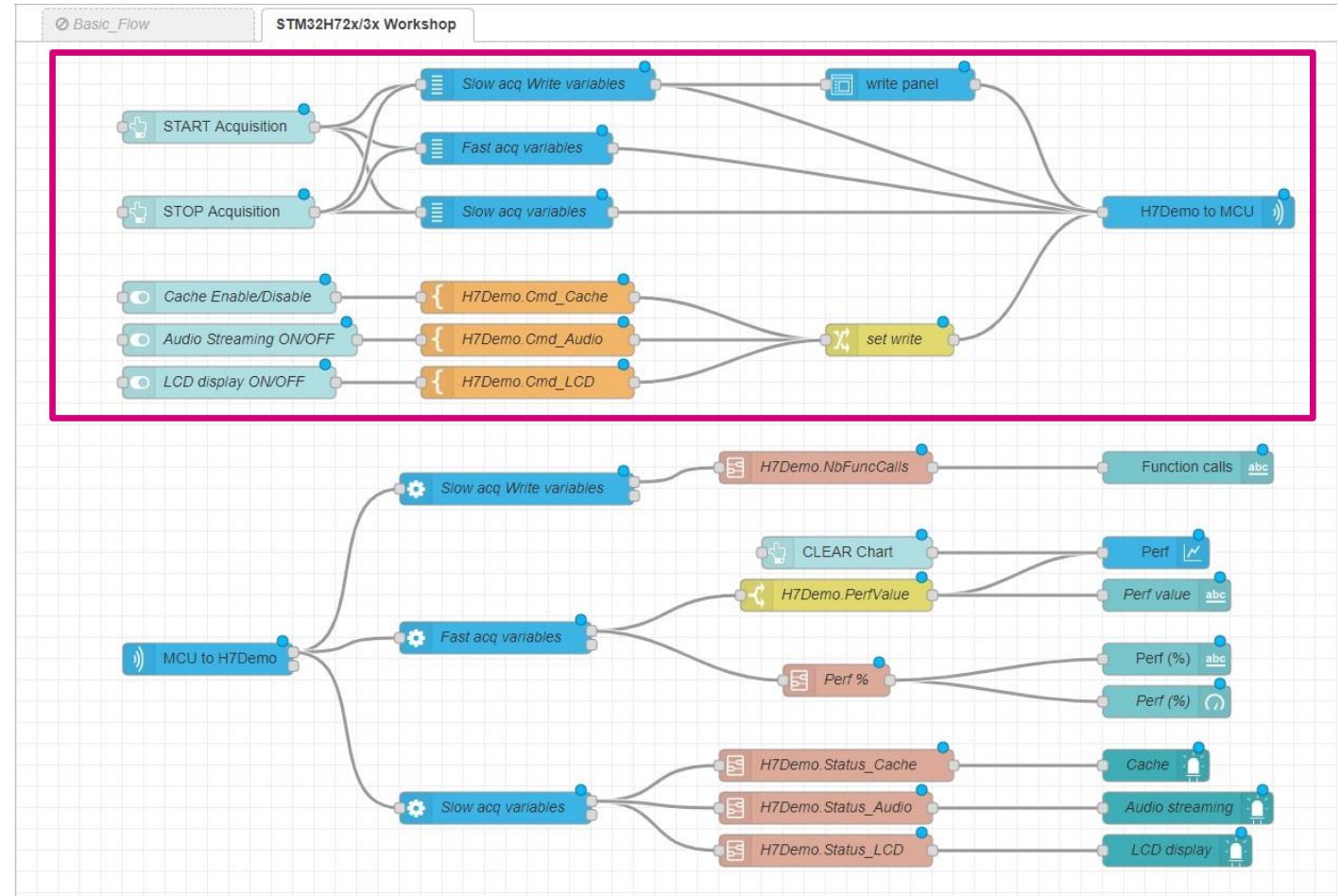
- One TX path from PC to probe

- But 3 variable lists

- Only 1 variable can be written in the write panel
- The sampling frequency is slow or fast

- And a new way to write in MCU RAM memory

- 3 switch widgets allows to write to 3 variables
- But the same message to the ST-Link probe than the write panel
- “write this value at that address”
- 0 for switch OFF, 1 for switch ON



Same flow structure

- One RX path from probe to PC

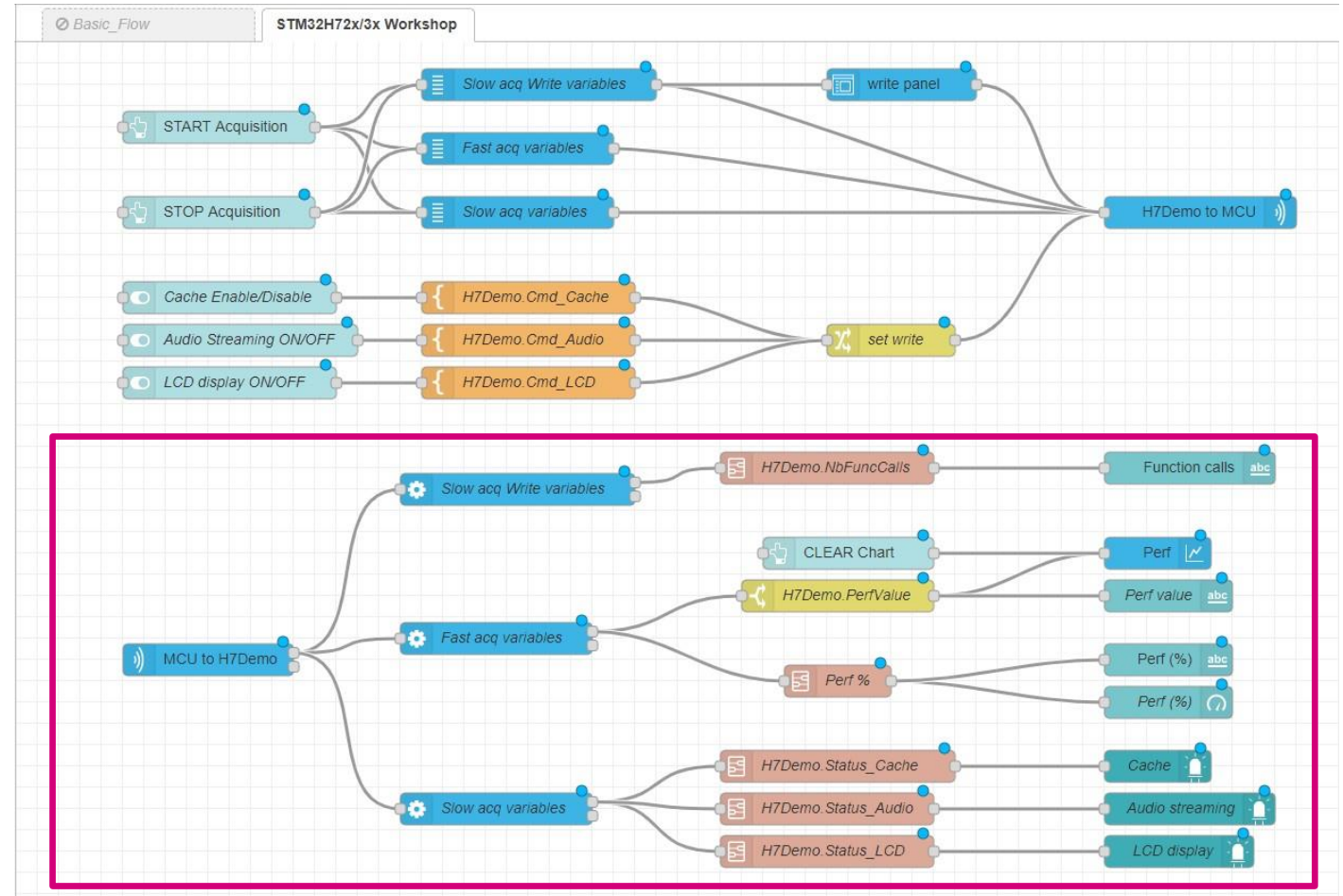
- 3 variable processing nodes
 - Paired with each variable list

- Again 1 chart but also other output widgets
 - Text, gauge, LEDs

- In between, selection nodes
 - Select which variable is displayed on which widget
 - A variable can be displayed on several widgets
 - Several variables can be displayed on a widget

- **More customization** but...

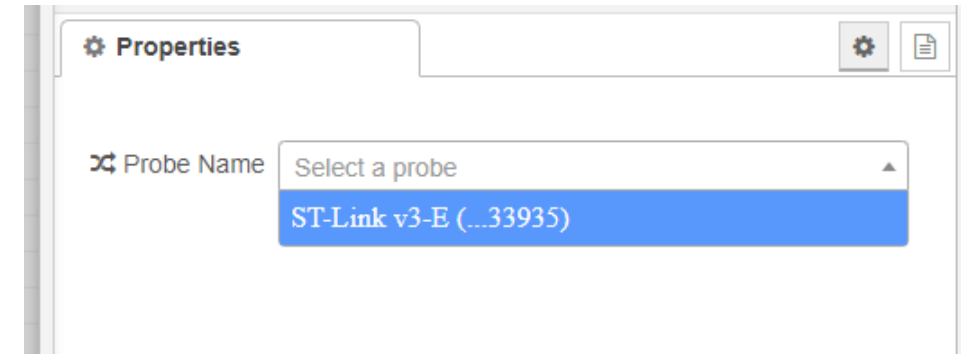
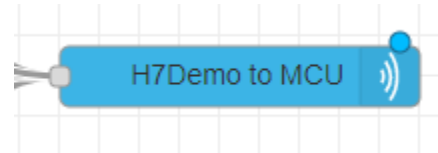
- **Nothing more difficult** than "Basic_Flow"



Prepare the flow

- Set ST-Link (Monitor to MCU)

1. Double-click on “H7Demo to MCU”
2. Click on the pen of “Probe Config”
3. Select the ST-Link probe
4. Click on “Add”
5. Click on “Done”



- Set ST-Link (MCU to Monitor)

- Do the same for “MCU to H7Demo”



Launch the dashboard

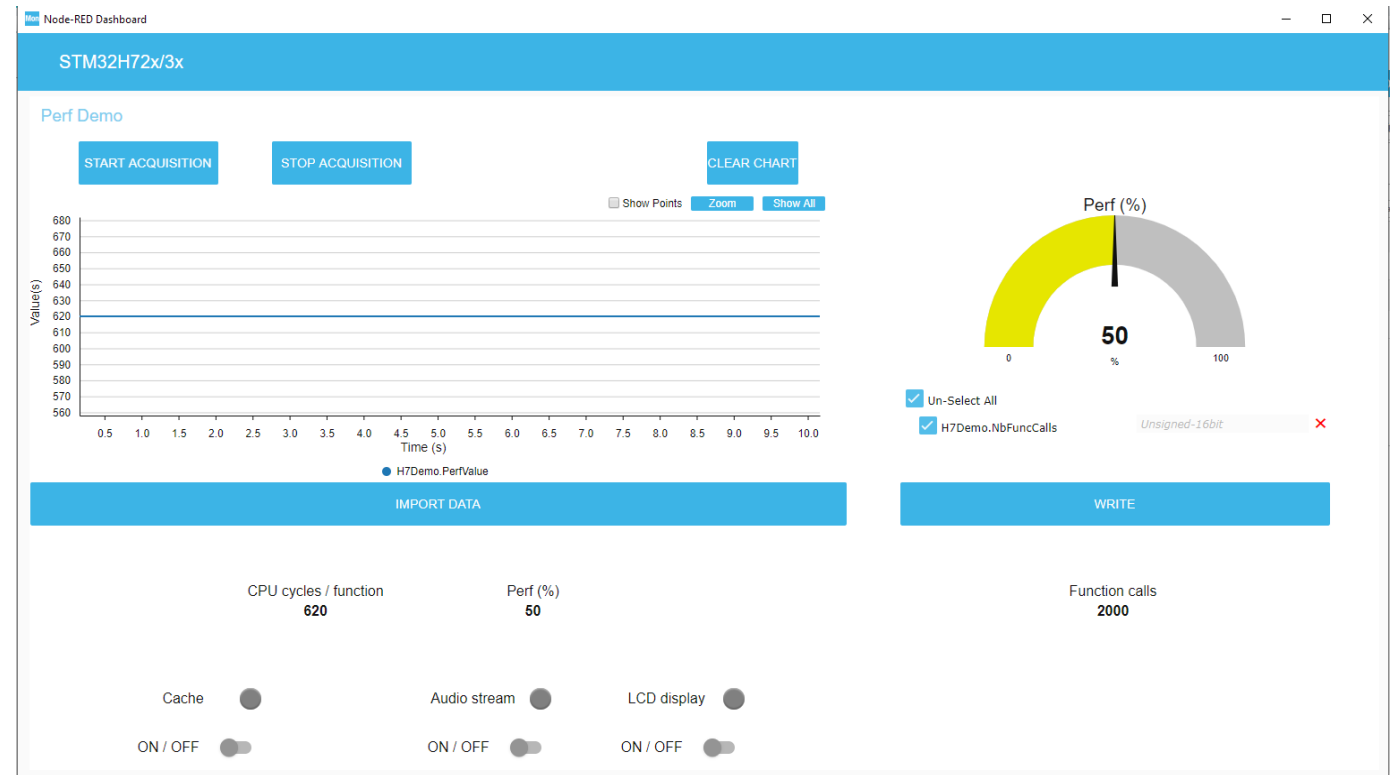
- The dashboard displays

- Known widgets

- 3 Buttons
 - START, STOP, CLEAR
- 1 chart, 1 write panel

- New widgets

- 1 gauge
- 3 text fields
- 3 switches and LEDs



Variables at fixed address

- Other firmware will be downloaded to the board for HandsOn2 and HandsOn3
- But the same variables will be used.
 - Variables in a structure at fixed address
 - H7Demo - 0x24000000 to 0x24000008
 - Reads and writes at the same addresses
 - No need to update the elf file

The screenshot displays the STM32H72x/3x Workshop interface. On the left, a flowchart shows nodes for 'START Acquisition', 'STOP Acquisition', 'Cache Enable/Disable', 'Audio Streaming ON/OFF', and 'LCD display ON/OFF'. These nodes are connected to 'Slow acq Write variables', 'Fast acq variables', and 'Slow acq variables' blocks. A 'MCU to H7Demo' block is also present, connected to the 'Fast acq variables' and 'Slow acq variables' blocks. On the right, the 'Edit variables node > Edit exe-config node' dialog is open. It shows the 'Properties' tab with fields for 'Name' (MySlowVarList), 'Folder' (C:\STM32H7EcoWS\HandsOn01), and 'File' (STM32H735G-DK.elf). Below these fields is an 'Expand Variable List' checkbox and a 'Variable List' table.

Select	Name	Start Address	Type
<input type="checkbox"/>	basicCounter	0x2400802c	Unsigned 16-bit
<input type="checkbox"/>	D1CorePrescTable[0]	0x08001dfc	Unsigned 8-bit
<input type="checkbox"/>	H7Demo.Cmd_Audio	0x24000001	Unsigned 8-bit
<input type="checkbox"/>	H7Demo.Cmd_Cache	0x24000000	Unsigned 8-bit
<input type="checkbox"/>	H7Demo.Cmd_LCD	0x24000002	Unsigned 8-bit
<input type="checkbox"/>	H7Demo.NbFuncCalls	0x24000008	Unsigned 16-bit
<input type="checkbox"/>	H7Demo.PerfValue	0x24000006	Unsigned 16-bit
<input checked="" type="checkbox"/>	H7Demo.Status_Audio	0x24000004	Unsigned 8-bit
<input checked="" type="checkbox"/>	H7Demo.Status_Cache	0x24000003	Unsigned 8-bit
<input checked="" type="checkbox"/>	H7Demo.Status_LCD	0x24000005	Unsigned 8-bit
<input type="checkbox"/>	SystemCoreClock	0x24008000	Unsigned 32-bit
<input type="checkbox"/>	SystemD2Clock	0x24008004	Unsigned 32-bit
<input type="checkbox"/>	uwTick	0x24008030	Unsigned 32-bit
<input type="checkbox"/>	uwTickFreq	0x2400800c	Signed 8-bit
<input type="checkbox"/>	uwTickPrio	0x24008008	Unsigned 32-bit

Launch the dashboard

- The current firmware is a simulation firmware
 - The same logic but only **dummy actions**
- Example of the cache switch
 - When HandsOn2 firmware will do

```
if(H7Demo.Status_Cache != H7Demo.Cmd_Cache){  
    if (H7DEMO_ON == H7Demo.Cmd_Cache){  
        SCB_EnableICache();  
        H7Demo.Status_Cache = H7DEMO_ON;
```

 - The Perf varies because of cache on/off
 - The current firmware does

```
PerfPercent += 50;  
H7Demo.Status_Cache = H7DEMO_ON;
```

 - The cache is **always off**
 - The Perf is **dummy**



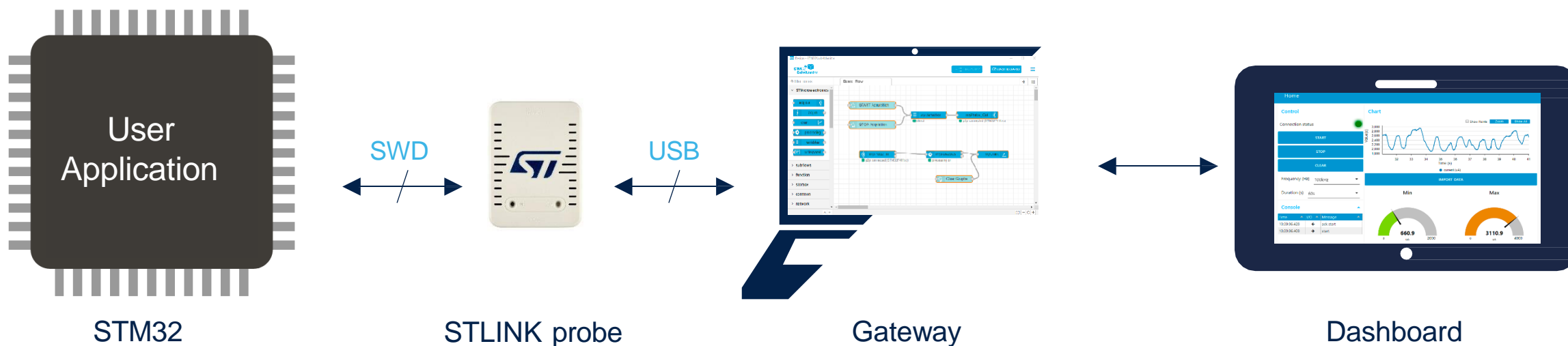
Remote monitoring

Native support of multi-format displays

Dynamic layout of dashboard UI on PCs, tablets, smartphones.

Remote data acquisition with web server technology


Monitor across a network with a web browser



Conclusion

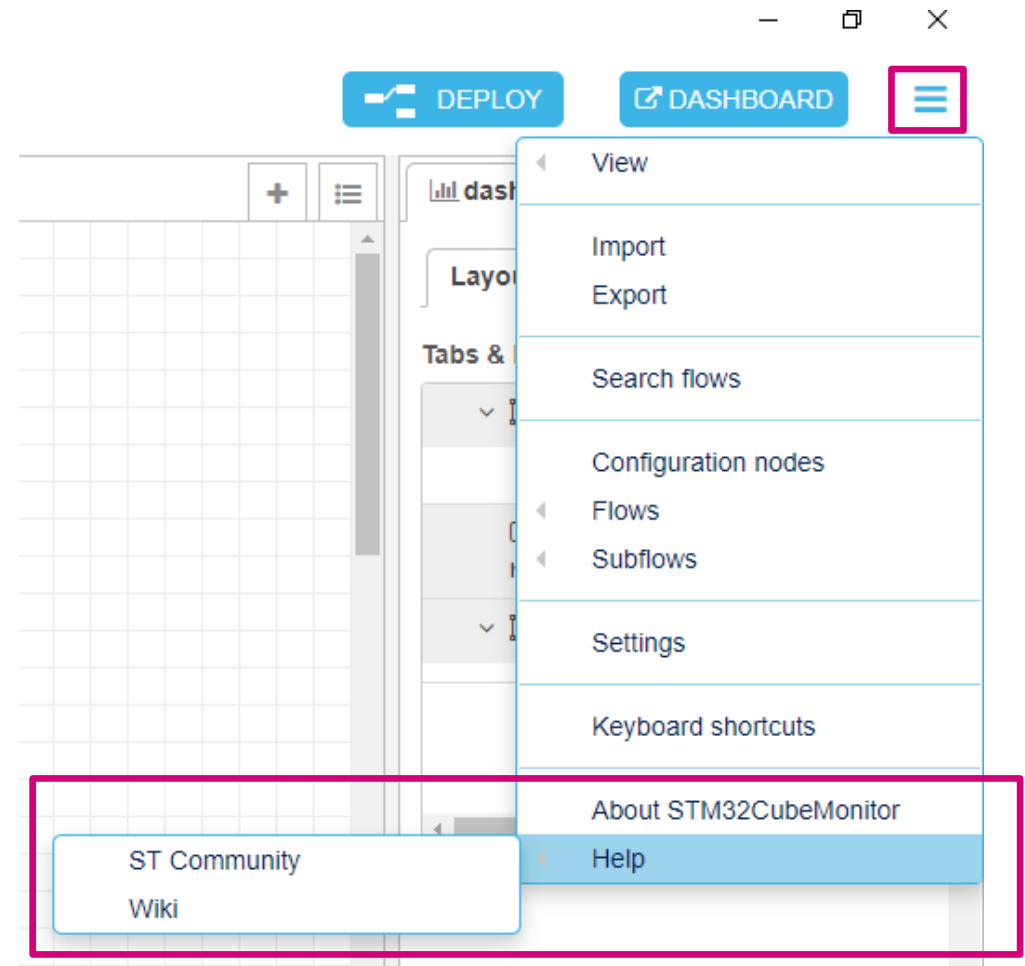
- STM32CubeMonitor is a non-intrusive tool to monitor a running application
 - Uses the MCU debug bloc and the SWD protocol
- STM32CubeMonitor is easily customizable.
 - We have seen the basics:
 - How to **configure** the flow
 - How to **access** to the STM32 memory
 - How to add **widgets** to the dashboard
 - You are **ready to design your own** dashboard

Going further - Documentation & Links

STM32 
CubeMonitor

To go further

- The help is directly accessible through the menu
 - Allow to find information to guide the user
 - ST Community
 - STM32 MCU wiki



Resources and useful links

[Visit our STM32CubeMonitor web page](#)



[Read our technical documentation on STM32CubeMonitor](#)



[Share use cases with STM32 community](#)



[Watch our Getting started webinar \(video\)](#)



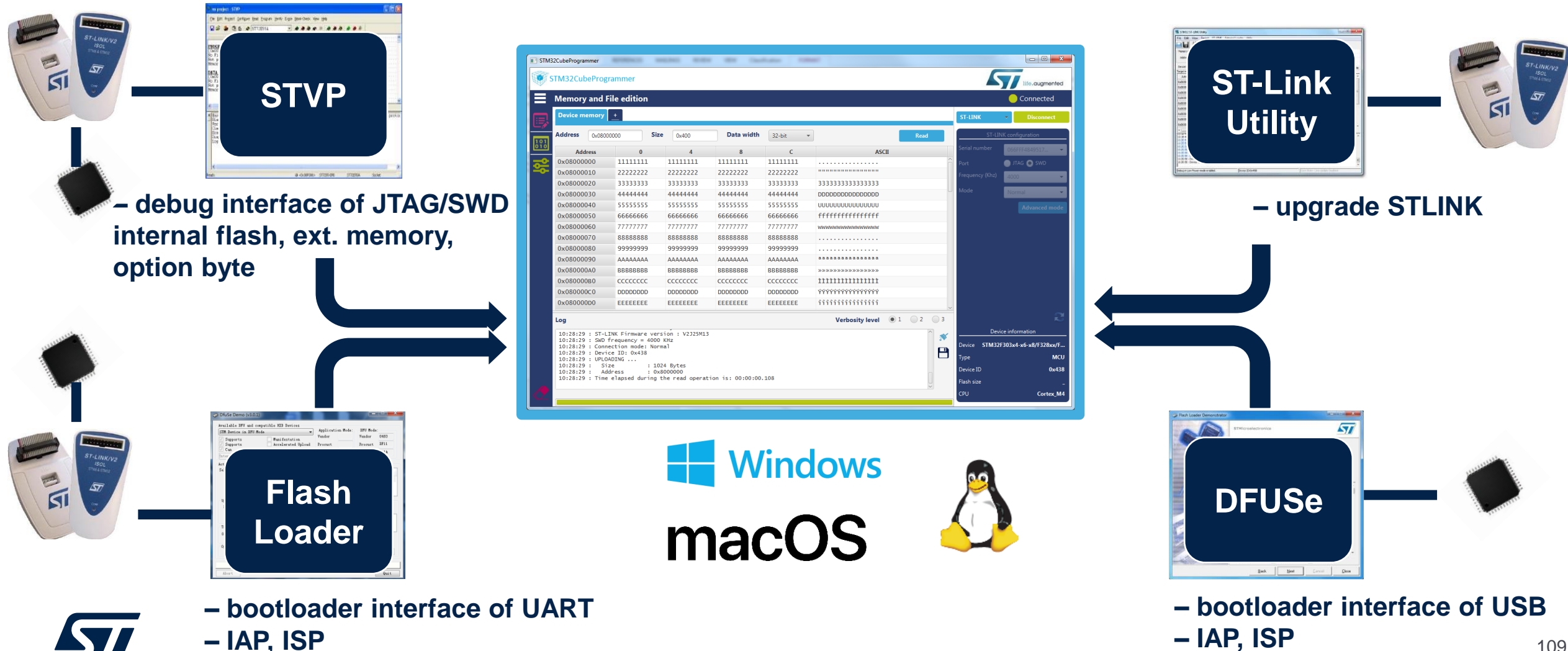
[Read our latest blog post](#)



STM32CubeProgrammer

STM32 
CubeProgrammer

From multiple utilities to STM32CubeProgrammer



All-in-one programming software tool



Intuitive GUI

Multi-platform
(Windows, Linux, macOS)

STLink Direct Support
(JTAG, SWD)

Automatic Mode

Option Bytes
Program & Upload

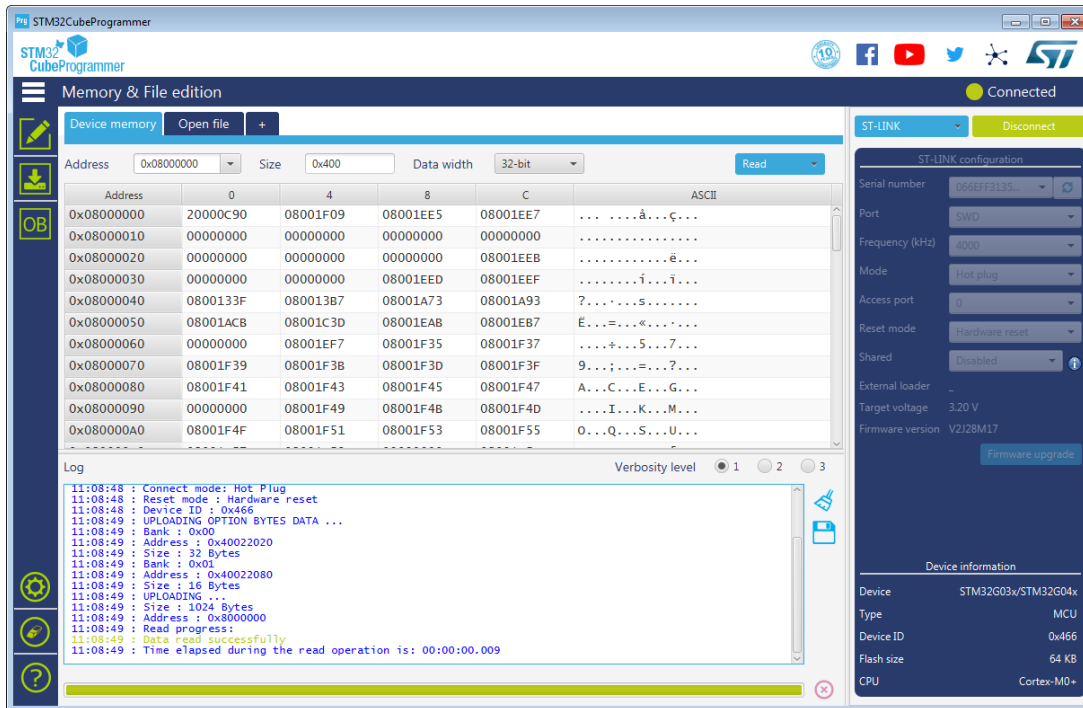
Bootloader Interface Support
(USB, UART, SPI, I2C, CAN)

Internal/External
Flash Services

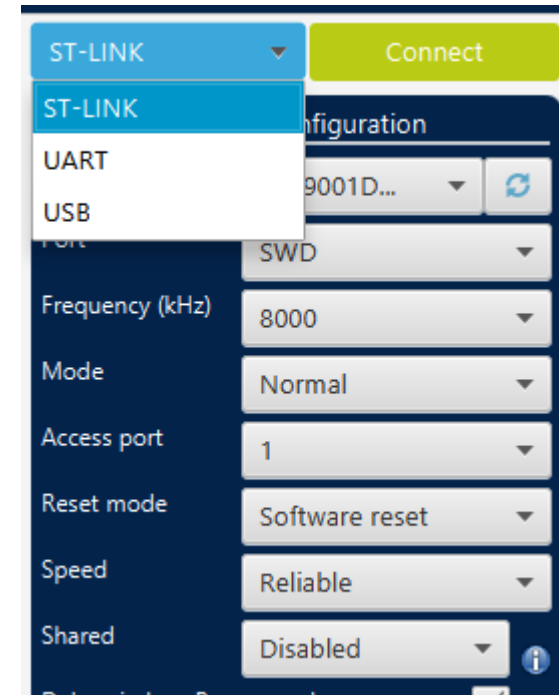
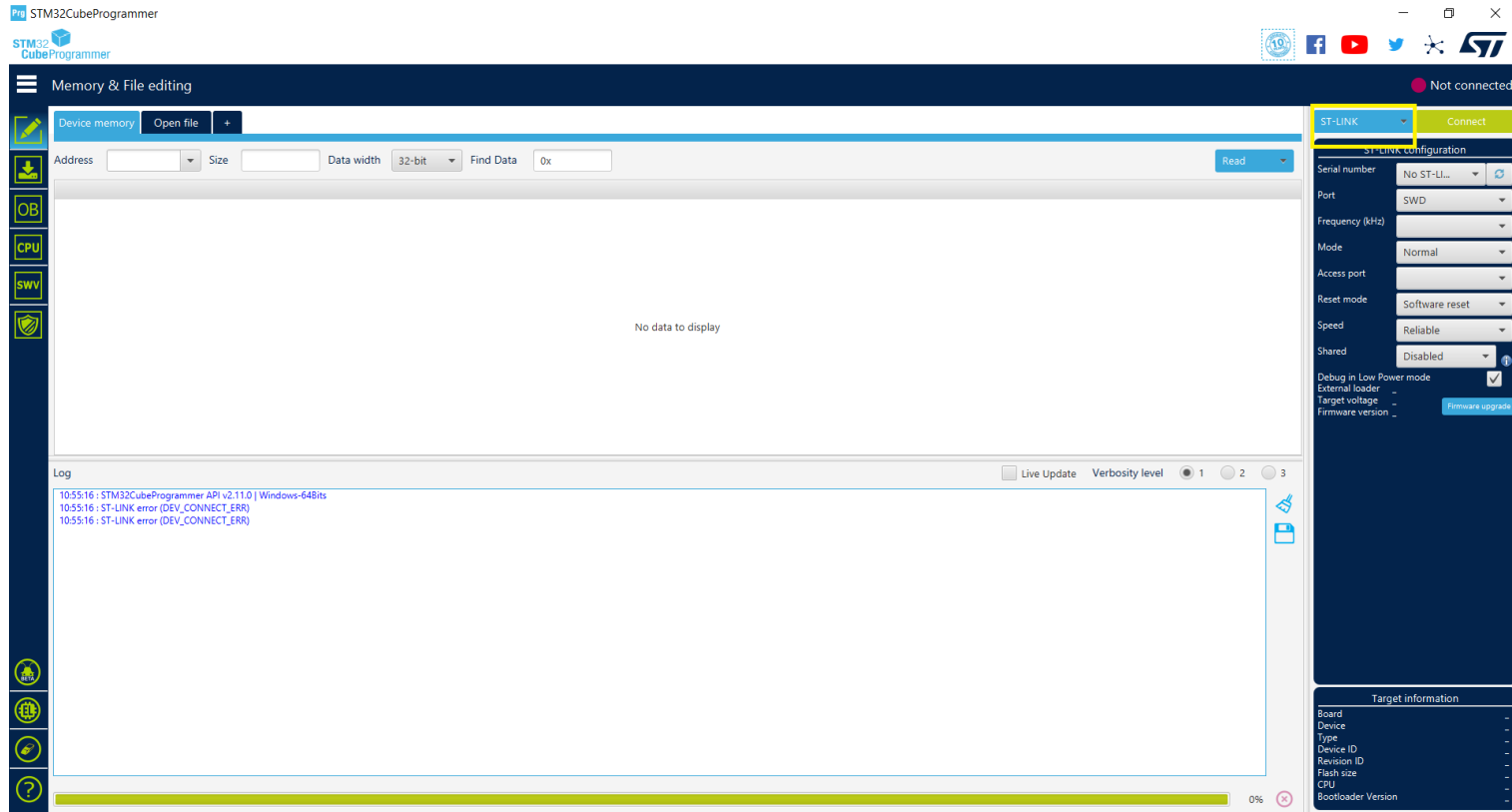
API DLL
for Custom Integration

Command Line Interface
for Scripting

Trusted Package Creator
(secure programming)



Supported interfaces

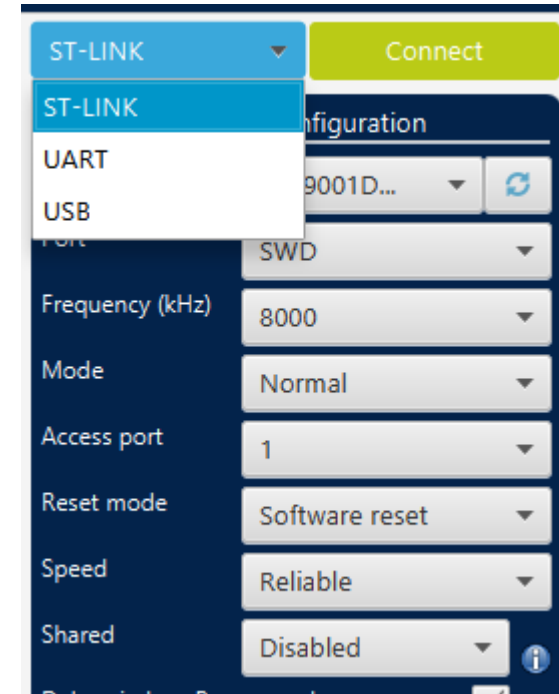
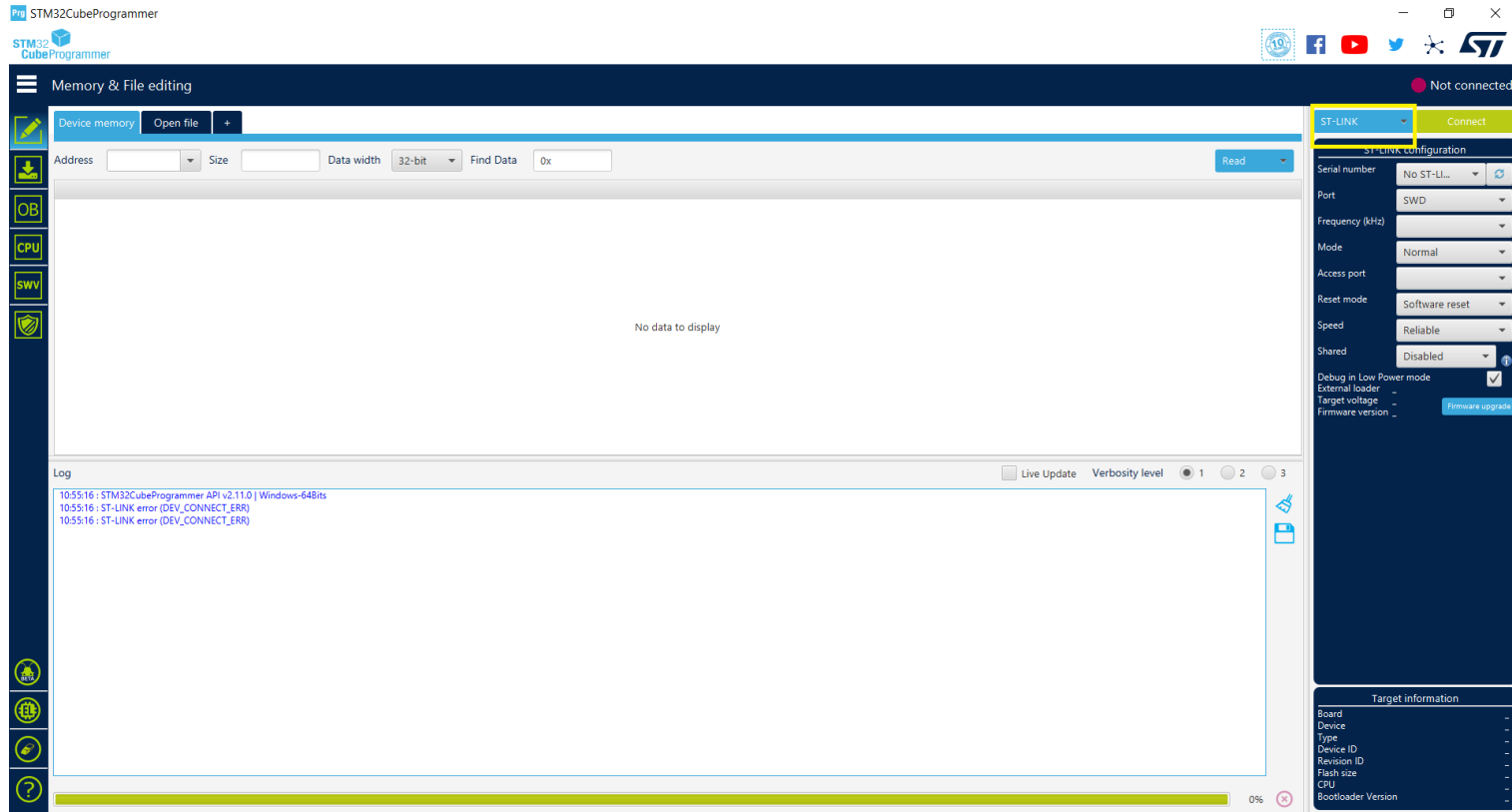


ST-LINK

UART

USB

Supported interfaces



SPI

CAN

I2C

Memory & file editing

Read

Edit

Download

Verify

STM32CubeProgrammer

Memory & File editing

Device memory | Open file | +

Address: 0x08000000 | Size: 0x400 | Data width: 32-bit | Find Data: 0x | Read

Address	0	4	8	C	ASCII
0x08000000	B92788CC	10DEF450	1703D969	1651E08B	1.'Pôp.iü...àQ.
0x08000010	825D332F	39B86DCC	1747631E	39C2095D	/3].Im>9.cG.].A9
0x08000020	0B027882	5C30AA28	B52DD807	CC82A259	.x..(°0\,0-µY€.I
0x08000030	737ED87F	68B169D0	13EAA34E	C9539456	.0~sDi±hNfê.v.SÊ
0x08000040	34699659	C654A72F	59A2916C	37BE8650	Y.i4/\$TÆ1.çYP.¾7
0x08000050	852DAF3C	CF8C0E32	F007BDA0	2F6D67D3	<~-.2..I ½.ð0gm/
0x08000060	6D9A47C8	315B89A5	46B7DABF	A7280566	ÊG.m¥.[1¿Ü·FF.(§
0x08000070	4EC69CB6	17CD4349	605E7E04	F10CFEE5	¶.ÆNICI...~^`âp.ñ
0x08000080	7B8A1103	AA877547	9B2992D0	410C28A0	...{Gu."-.).(.A
0x08000090	CFA2B8E3	BB6B4CEA	55EB7897	9A5CF0C4	ä.«IêLk».xêUÄð\.
0x080000A0	A1909075	D324A604	614D5742	4182F6A5	u..j.{\$0BwMaYö.A
0x080000B0	98FCA332	AEF8A38C	4FFC41F9	4F8F31AA	2Ëü..f0°üAü0*1.0
0x080000C0	D279E07B	84542D72	E465E53F	1AEDDACD	{âyÖr-T.7äeäIÜí.

Log

10:58:48: Disconnected from device.
 10:58:52: ST-LINK error (DEV_CONNECT_ERR)
 10:58:52: ST-LINK SN : 0669FF544949878667183240
 10:58:52: ST-LINK FW : V2J28M18
 10:58:52: Board : NUCLEO-L433RC-P
 10:58:52: Voltage : 3.26V
 10:58:52: SWD freq : 4000 KHz
 10:58:52: Connect mode: Normal
 10:58:52: Reset mode: Software reset
 10:58:52: Device ID : 0x435
 10:58:52: Revision ID : Rev Z
 10:58:52: Debug in Low Power mode is not supported for this device.
 10:58:52: UPLOADING OPTION BYTES DATA ...
 10:58:52: Bank : 0x00
 10:58:52: Address : 0x40022020
 10:58:52: Size : 20 Bytes
 10:58:52: UPLOADING ...
 10:58:52: Size : 1024 Bytes
 10:58:52: Address : 0x08000000
 10:58:52: Read progress:
 10:58:52: Data read successfully
 10:58:52: Time elapsed during the read operation is: 00:00:00.007

ST-LINK configuration

Serial number: 0669FF5...
 Port: SWD
 Frequency (kHz): 4000
 Mode: Normal
 Access port: 0
 Reset mode: Software reset
 Speed: Reliable
 Shared: Disabled
 Debug in Low Power mode: ☒
 External loader: ☒
 Target voltage: 3.26 V
 Firmware version: V2J28M18

Target information

Board: NUCLEO-L433RC-P
 Device: STM32L43xxx/STM32L44xxx
 Type: MCU
 Device ID: 0x435
 Revision ID: Rev Z
 Flash size: 256 KB
 CPU: Cortex-M4
 Bootloader Version: --

Download

STM32CubeProgrammer

STM32CubeProgrammer

Connected

Memory & File editing

Device memory FlashLayout_sdcard_stm32mp135f-dk-optee.tsv

Address Size Data width 32-bit Find Data 0x Download

Select	Opt	Id	Name	Type	IP	Offset	Binary
<input checked="" type="checkbox"/>	-	0x1	fsbl1-boot	Binary	none	0x00000000	tf-a-stm32mp135f-dk-usb.stm32
<input checked="" type="checkbox"/>	-	0x3	fip-boot	Binary	none	0x00000000	fip-stm32mp135f-dk-optee.bin
<input checked="" type="checkbox"/>	P	0x4	fsbl1	Binary	mmc0	0x00004400	tf-a-stm32mp135f-dk-sdcard.stm32
<input checked="" type="checkbox"/>	P	0x5	fsbl2	Binary	mmc0	0x00044400	tf-a-stm32mp135f-dk-sdcard.stm32
<input checked="" type="checkbox"/>	PD	0x6	fip	Binary	mmc0	0x00084400	fip-stm32mp135f-dk-optee.bin
<input checked="" type="checkbox"/>	P	0x10	bootfs	System	mmc0	0x00484400	st-image-bootfs-stm32mp-valid-stm32mp13-valid.ext4

Binaries path C:\PRG MPU\Flashing Browse

Log

Live Update Verbosity level 1 2 3

```

14:27:29 : File : fip-stm32mp135f-dk-optee.bin
14:27:29 : Size : 1.48 MB
14:27:29 : Partition ID : 0x06
14:27:29 : Download in Progress:
14:27:31 : File download complete
14:27:31 : Time elapsed during download operation: 00:00:02.365
14:27:31 : RUNNING Program ...
14:27:31 : PartID: :0x06
14:27:31 : Start operation done successfully at partition 0x06
14:27:31 : Memory Programming ...
14:27:31 : Opening and parsing file: st-image-bootfs-stm32mp-valid-stm32mp13-valid.ext4
14:27:32 : File : st-image-bootfs-stm32mp-valid-stm32mp13-valid.ext4
14:27:32 : Size : 64.00 MB
14:27:32 : Partition ID : 0x10
14:27:32 : Download in Progress:
14:28:10 : File download complete
14:28:10 : Time elapsed during download operation: 00:00:38.477
14:28:10 : RUNNING Program ...
14:28:10 : PartID: :0x10
14:28:10 : Start operation done successfully at partition 0x10
14:28:10 : Flashing service completed successfully
  
```

USB configuration

Port USB1

Serial number 8018800235305104383435...

PID 0xdf11

VID 0x0483

Read Unprotect (MCU)

TZEN Regression (MCU)

Target information

Board --

Device STM32MP13xx

Type MPU

Device ID 0x501


Revision ID --

Flash size --

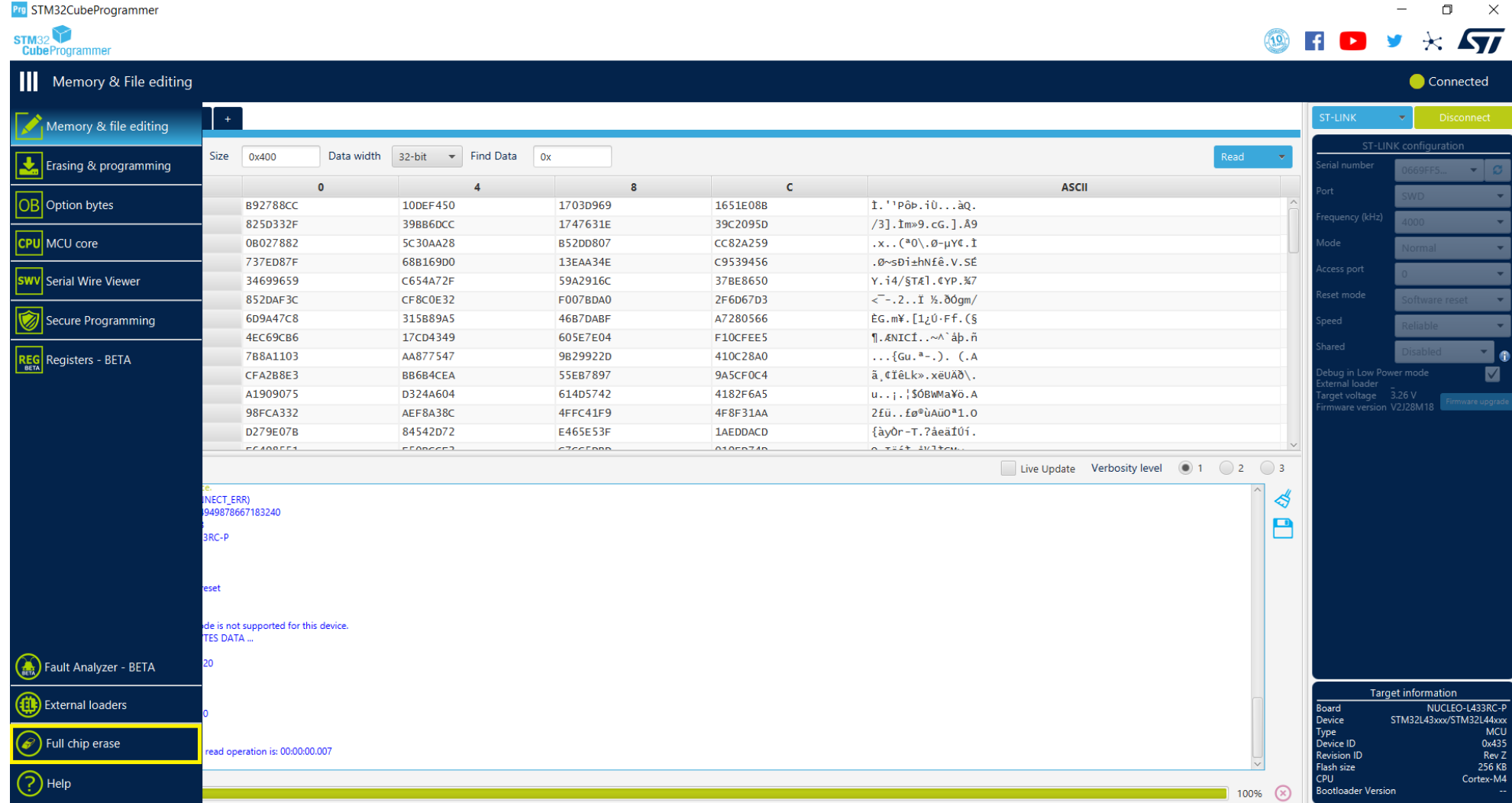
CPU Cortex-A7

Bootloader Version --

0%

 Full chip erase

Erase entire
chip memory



The screenshot shows the STM32CubeProgrammer application window. The left sidebar contains several menu items, with 'Full chip erase' highlighted in yellow. The main window displays a memory map table with columns for Size, Data width, Find Data, and ASCII. The table lists memory addresses and their corresponding data. The bottom status bar shows 'read operation is: 00:00:00.007' and a progress bar at 100%.

Size	Data width	Find Data	0	4	8	C	ASCII
B92788CC	10DEF450	1703D969	1651E08B	I. 'Pô».iû...àQ.			
825D332F	39BB60CC	1747631E	39C2095D	/3].îm»9.cG.] .A9			
0B027882	5C30AA28	B52D0807	CC82A259	.x..(*0\,0-µY€.I			
737ED87F	68B169D0	13EAA34E	C9539456	.0~sDî±hNfê.V.SÉ			
34699659	C65A72F	59A2916C	37BE8650	Y.i4/§T&1.CYP.¾7			
852DAF3C	CF8C0E32	F007BDA0	2F6D67D3	<"-..2..I %..ðógm/			
6D9A47C8	315B89A5	46B7DABF	A7280566	ÊG.m¥.[1¿Ü.Ff.(§			
4EC69CB6	17CD4349	605E7E04	F10CFEE5	¶.ÆNICÍ...~^`âp.ñ			
7B8A1103	AA877547	9B29922D	410C28A0	...{Gu.ª-). (.A			
CFA2B8E3	BB6B4CEA	55EB7897	9A5CF0C4	ä.«IêLk».xèUAd\.			
A1909075	D324A604	614D5742	4182F6A5	u..j.¡\$ÓBwMaYö.A			
98FCA332	AEF8A38C	4FFC41F9	4F8F31AA	2Êü..Êø*ûAu0*1.O			
D279E07B	84542D72	E465E53F	1AEDDACD	{âyÖr-T.¿âëäFüí.			

Target information:

- Board: NUCLEO-L433RC-P
- Device: STM32L43xxx/STM32L44xxx
- Type: MCU
- Device ID: 0x435
- Revision ID: Rev Z
- Flash size: 256 KB
- CPU: Cortex-M4
- Bootloader Version: --

Erasing & programming



File programming
(.bin, .hex, .srec, .elf)

Automatic Mode

Sector erase

STM32CubeProgrammer

Erasing & Programming

Download

File path: C:\128KB.bin

Start address: 0x08000000

☐ Skip flash erase before programming
☒ Verify programming
☐ Run after programming

Start Programming

Automatic Mode

☒ Full chip erase
☐ Download file
☐ Option bytes commands

Start automatic mode

Erase flash memory | Erase external memory

Erase selected sectors | Full chip erase

Select	Index	Start Address	Size
<input type="checkbox"/>	0	0x08000000	2K
<input type="checkbox"/>	1	0x08000800	2K
<input type="checkbox"/>	2	0x08001000	2K
<input type="checkbox"/>	3	0x08001800	2K
<input type="checkbox"/>	4	0x08002000	2K
<input type="checkbox"/>	5	0x08002800	2K
<input type="checkbox"/>	6	0x08003000	2K
<input type="checkbox"/>	7	0x08003800	2K
<input type="checkbox"/>	8	0x08004000	2K
<input type="checkbox"/>	9	0x08004800	2K

Log

10:58:48 : Disconnected from device.
 10:58:52 : ST-LINK error (DEV_CONNECT_ERR)
 10:58:52 : ST-LINK SN : 0669FF544949878667183240
 10:58:52 : ST-LINK FW : V2J28M18
 10:58:52 : Board : NUCLEO-L433RC-P
 10:58:52 : Voltage : 3.26V
 10:58:52 : SWD freq : 4000 KHz
 10:58:52 : Connect mode: Normal
 10:58:52 : Reset mode : Software reset
 10:58:52 : Device ID : 0x435
 10:58:52 : Revision ID : Rev Z
 10:58:52 : Debug in Low Power mode is not supported for this device.
 10:58:52 : UPLOADING OPTION BYTES DATA ...
 10:58:52 : Bank : 0x00
 10:58:52 : Address : 0x40022020
 10:58:52 : Size : 20 Bytes
 10:58:52 : UPLOADING ...
 10:58:52 : Size : 1024 Bytes
 10:58:52 : Address : 0x08000000
 10:58:52 : Read progress:
 10:58:52 : Data read successfully
 10:58:52 : Time elapsed during the read operation is: 00:00:00.007

ST-LINK configuration

Serial number: 0669FF5...

Port: SWD

Frequency (kHz): 4000

Mode: Normal

Access port: 0

Reset mode: Software reset

Speed: Reliable

Shared: Disabled

Debug in Low Power mode: ☒

External loader: Target voltage: 3.26 V

Firmware version: V2J28M18

Target information

Board: NUCLEO-L433RC-P

Device: STM32L43xxx/STM32L44xxx

Type: MCU

Device ID: 0x435

Revision ID: Rev Z

Flash size: 256 KB

CPU: Cortex-M4

Bootloader Version: --

100%

STM32CubeProgrammer

External loaders

Available external loaders:

Select	Name	Board	Start Address	Memory Size	Page Size	Type
<input type="checkbox"/>	512W3A_STM3210E-EVAL	STM3210E-EVAL	0x70000000	64M	0x200	NAND_FLASH
<input type="checkbox"/>	IS42S32400F_STM32F469I-DK	STM32F469I-DK	0xC0000000	16M	0x1000000	SRAM
<input type="checkbox"/>	IS42S32800G_STM32769I-EVAL	STM32769I-EVAL	0xC0000000	32M	0x2000000	SRAM
<input type="checkbox"/>	IS61WV102416BLL_STM324x9I-EVAL	STM324x9I-EVAL	0x64000000	2M	0x200000	SRAM
<input type="checkbox"/>	IS61WV102416BLL_STM324xG-EVAL	STM324xG-EVAL	0x64000000	2M	0x200000	SRAM
<input type="checkbox"/>	IS61WV102416BLL_STM32769I-EVAL	STM32769I-EVAL	0x68000000	2M	0x200000	SRAM
<input type="checkbox"/>	IS61WV51216BLL_STM3210E-EVAL	STM3210E-EVAL	0x68000000	1M	0x10000	SRAM
<input type="checkbox"/>	IS66WV51216BLL_STM32723E-DISCO	STM32723E-DISCO	0x60000000	512K	0x80000	SRAM
<input type="checkbox"/>	IS66WV51216BLL_STM32F413H-DISCO	STM32F413H-DISCO	0x60000000	512K	0x80000	SRAM
<input type="checkbox"/>	M24LR-A_STM32303C-EVAL	STM32303C-EVAL	0x00000000	8K	0x2000	I2C_EEPROM
<input type="checkbox"/>	M24LR-A_STM32373C-EVAL	STM32373C-EVAL	0x00000000	8K	0x2000	I2C_EEPROM
<input type="checkbox"/>	M24LR-A_STM324x9I-EVAL	STM324x9I-EVAL	0x00000000	8K	0x2000	I2C_EEPROM

Log

```

10:58:48 : Disconnected from device.
10:58:52 : ST-LINK error (DEV_CONNECT_ERR)
10:58:52 : ST-LINK SN : 0669FF544949878667183240
10:58:52 : ST-LINK FW : V2J28M18
10:58:52 : Board : NUCLEO-L433RC-P
10:58:52 : Voltage : 3.26V
10:58:52 : SWD freq : 4000 KHz
10:58:52 : Connect mode: Normal
10:58:52 : Reset mode : Software reset
10:58:52 : Device ID : 0x435
10:58:52 : Revision ID : Rev Z
10:58:52 : Debug in Low Power mode is not supported for this device.
10:58:52 : UPLOADING OPTION BYTES DATA ...
10:58:52 : Bank : 0x00
10:58:52 : Address : 0x40022020
10:58:52 : Size : 20 Bytes
10:58:52 : UPLOADING ...
10:58:52 : Size : 1024 Bytes
10:58:52 : Address : 0x80000000
10:58:52 : Read progress:
10:58:52 : Data read successfully
10:58:52 : Time elapsed during the read operation is: 00:00:00.007
    
```

ST-LINK configuration

Serial number: 0669FF544949878667183240

Port: SWD

Frequency (kHz): 4000

Mode: Normal

Access port: 0

Reset mode: Software reset

Speed: Reliable

Shared: Disabled

Debug in Low Power mode: ☒

External loader: ☐

Target voltage: 3.26 V

Firmware version: V2J28M18

Firmware upgrade

Target information

Board: NUCLEO-L433RC-P

Device: STM32L43xxx/STM32L44xxx

Type: MCU

Device ID: 0x435

Revision ID: Rev Z

Flash size: 256 KB

CPU: Cortex-M4

Bootloader Version: --

100%

External loaders

Program External memory

OB Option bytes

Read & Program
option Bytes

STM32CubeProgrammer

Option bytes

Read Out Protection

Name	Value	Description
RDP	AA	Read protection option byte The read protection is used to protect the software code stored in Flash memory. AA : Level 0, no protection BB : or any value other than 0xAA and 0xCC: Level 1, read protection CC : Level 2, chip protection

BOR Level

User Configuration

Name	Value	Description
nRST_STOP	<input checked="" type="checkbox"/>	Unchecked : Reset generated when entering Stop mode Checked : No reset generated when entering Stop mode
nRST_STDBY	<input checked="" type="checkbox"/>	Unchecked : Reset generated when entering Standby mode Checked : No reset generated when entering Standby mode
nRST_SHDW	<input checked="" type="checkbox"/>	Unchecked : Reset generated when entering the Shutdown mode Checked : No reset generated when entering the Shutdown mode
IWDG_SW	<input checked="" type="checkbox"/>	Unchecked : Hardware independant watchdog Checked : Software independant watchdog
IWDG_STOP	<input checked="" type="checkbox"/>	Unchecked : Freeze IWDG counter in stop mode

Some of the option bytes might be hidden or clipped, Use the mouse wheel or the touch pad to scroll down

Apply Read

Log

Live Update Verboosity level 1 2 3

```

10:58:48 : Disconnected from device.
10:58:52 : ST-LINK error (DEV_CONNECT_ERR)
10:58:52 : ST-LINK SN : 0669FF544949878667183240
10:58:52 : ST-LINK FW : V2J28M18
10:58:52 : Board : NUCLEO-L433RC-P
10:58:52 : Voltage : 3.26V
10:58:52 : SWD freq : 4000 KHz
10:58:52 : Connect mode: Normal
10:58:52 : Reset mode : Software reset
10:58:52 : Device ID : 0x435
10:58:52 : Revision ID : Rev Z
10:58:52 : Debug in Low Power mode is not supported for this device.
10:58:52 : UPLOADING OPTION BYTES DATA ...
10:58:52 : Bank : 0x00
10:58:52 : Address : 0x40022020
10:58:52 : Size : 20 Bytes
10:58:52 : UPLOADING ...
10:58:52 : Size : 1024 Bytes
10:58:52 : Address : 0x8000000
10:58:52 : Read progress:
10:58:52 : Data read successfully
10:58:52 : Time elapsed during the read operation is: 00:00:00.007
    
```

100%

ST-LINK configuration

ST-LINK

Disconnect

Serial number 0669FF5...

Port SWD

Frequency (KHz) 4000

Mode Normal

Access port 0

Reset mode Software reset

Speed Reliable

Shared Disabled

Debug in Low Power mode ☒

External loader

Target voltage 3.26 V

Firmware version V2J28M18

Firmware upgrade

Target information

Board NUCLEO-L433RC-P

Device STM32L43xxx/STM32L44xxx

Type MCU

Device ID 0x435

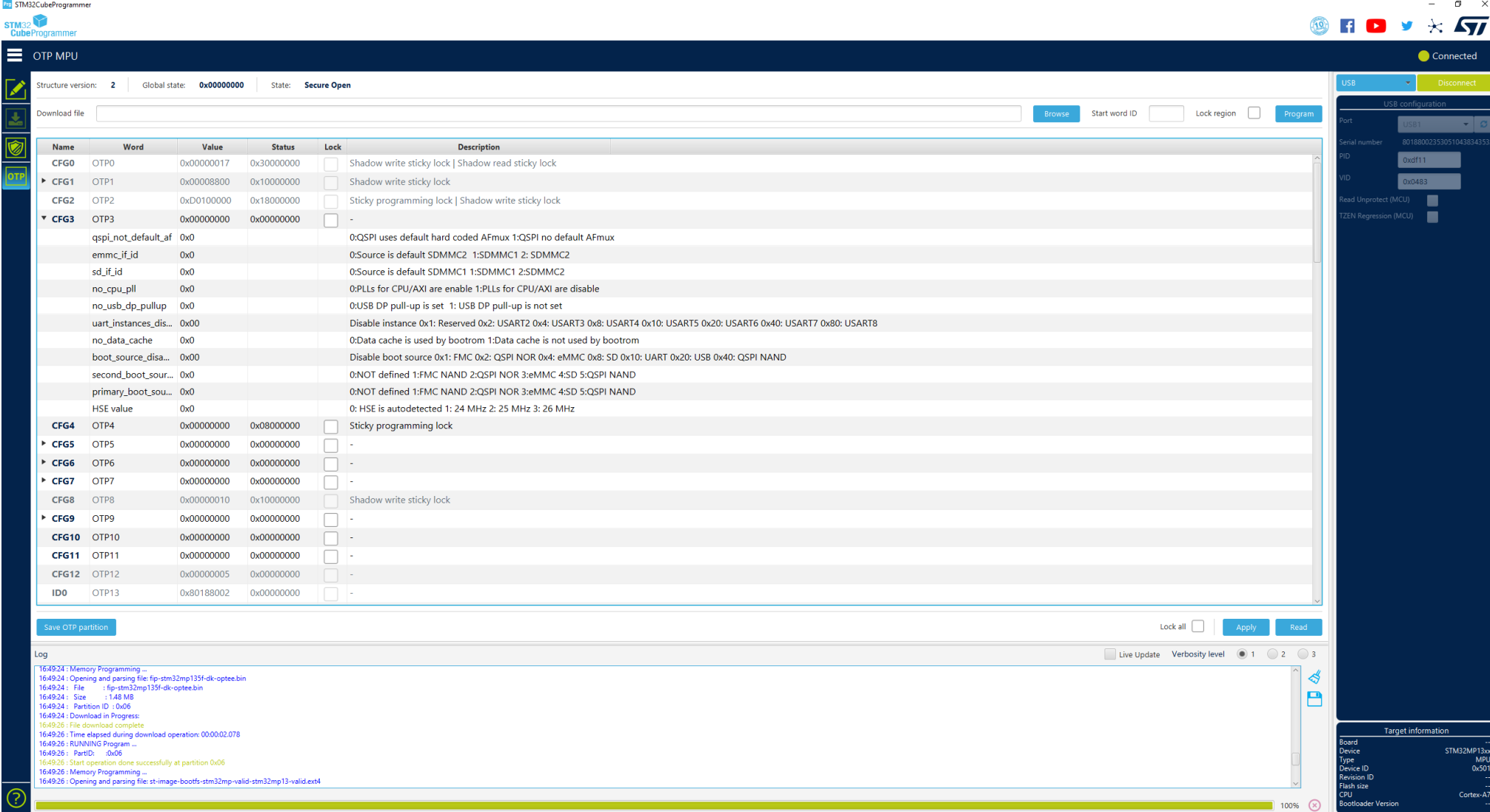
Revision ID Rev Z

Flash size 256 KB

CPU Cortex-M4

Bootloader Version --

OTP programming



The screenshot displays the STM32CubeProgrammer interface for OTP MPU programming. The main window shows a table of OTP configurations with columns for Name, Word, Value, Status, Lock, and Description. The table lists various OTPs (CFG0 to ID0) and their corresponding values and descriptions. The interface includes a 'Download file' section, a 'Save OTP partition' button, and a 'Log' section at the bottom showing the progress of the programming operation.

Name	Word	Value	Status	Lock	Description
CFG0	OTP0	0x00000017	0x30000000	<input type="checkbox"/>	Shadow write sticky lock Shadow read sticky lock
CFG1	OTP1	0x00008800	0x10000000	<input type="checkbox"/>	Shadow write sticky lock
CFG2	OTP2	0xD0100000	0x18000000	<input type="checkbox"/>	Sticky programming lock Shadow write sticky lock
CFG3	OTP3	0x00000000	0x00000000	<input type="checkbox"/>	-
	qspi_not_default_af	0x0			0:QSPI uses default hard coded Afmux 1:QSPI no default Afmux
	emmc_if_id	0x0			0:Source is default SDMMC2 1:SDMMC1 2:SDMMC2
	sd_if_id	0x0			0:Source is default SDMMC1 1:SDMMC1 2:SDMMC2
	no_cpu_pll	0x0			0:PLLs for CPU/AXI are enable 1:PLLs for CPU/AXI are disable
	no_usb_dp_pullup	0x0			0:USB DP pull-up is set 1:USB DP pull-up is not set
	uart_instances_dis...	0x00			Disable instance 0x1: Reserved 0x2: USART2 0x4: USART3 0x8: USART4 0x10: USART5 0x20: USART6 0x40: USART7 0x80: USART8
	no_data_cache	0x0			0:Data cache is used by bootrom 1:Data cache is not used by bootrom
	boot_source_disa...	0x00			Disable boot source 0x1: FMC 0x2: QSPI NOR 0x4: eMMC 0x8: SD 0x10: UART 0x20: USB 0x40: QSPI NAND
	second_boot_sour...	0x0			0:NOT defined 1:FMC NAND 2:QSPI NOR 3:eMMC 4:SD 5:QSPI NAND
	primary_boot_sou...	0x0			0:NOT defined 1:FMC NAND 2:QSPI NOR 3:eMMC 4:SD 5:QSPI NAND
	HSE value	0x0			0: HSE is autodetected 1: 24 MHz 2: 25 MHz 3: 26 MHz
CFG4	OTP4	0x00000000	0x08000000	<input type="checkbox"/>	Sticky programming lock
CFG5	OTP5	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG6	OTP6	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG7	OTP7	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG8	OTP8	0x00000010	0x10000000	<input type="checkbox"/>	Shadow write sticky lock
CFG9	OTP9	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG10	OTP10	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG11	OTP11	0x00000000	0x00000000	<input type="checkbox"/>	-
CFG12	OTP12	0x00000005	0x00000000	<input type="checkbox"/>	-
ID0	OTP13	0x80188002	0x00000000	<input type="checkbox"/>	-

Log:

```

16:49:24 : Memory Programming ...
16:49:24 : Opening and parsing file: fip-stm32mp135f-dk-optee.bin
16:49:24 : File : fip-stm32mp135f-dk-optee.bin
16:49:24 : Size : 148 MB
16:49:24 : Partition ID : 0x06
16:49:24 : Download in Progress:
16:49:25 : File download complete
16:49:26 : Time elapsed during download operation: 00:00:02.078
16:49:26 : RUNNING Program ...
16:49:26 : PartID: 0x06
16:49:26 : Start operation done successfully at partition 0x06
16:49:26 : Memory Programming ...
16:49:26 : Opening and parsing file: st-image-boots-stm32mp-valid-stm32mp13-valid.ext4
  
```

STM32CubeProgrammer

MCU core

Connected

ST-LINK configuration

Serial number: 00210028474150042...

Port: SWD

Frequency (kHz): 24000

Mode: Normal

Access port: 0

Reset mode: Software reset

Speed: Reliable

Shared: Disabled

Debug in Low Power mode: ☒

External loader: MX25LM51245G_STM32H573I-DK.stldr

Target voltage: 3.31 V

Firmware version: V3J9M3

Firmware upgrade

Target information

Board: STM32H573I-DK

Device: STM32H5xx

Type: MCU

Device ID: 0x484

Revision ID: Rev A

Flash size: 2 MB

CPU: Cortex-M33

Bootloader Version: 0xFF

Core ID: Cortex-M33

Core State: Halted

CPU Security state: -

Log

14:54:03 : UPLOADING ...

14:54:03 : Size : 2048 Bytes

14:54:03 : Address : 0x8000000

14:54:03 : Read progress:

14:54:03 : Data read successfully

14:54:03 : Time elapsed during the read operation is: 00:00:00.005

14:54:20 : Flash sector erase ...

14:54:20 : Existing specified sectors are erased successfully Protected sectors are not erased

16:09:21 : UPLOADING ...

16:09:21 : Size : 4 Bytes

16:09:21 : Address : 0xE000EE08

16:09:21 : Read progress:

16:09:21 : Data read successfully

16:09:21 : Time elapsed during the read operation is: 00:00:00.000

100%

CPU MCU core

Manipulate register

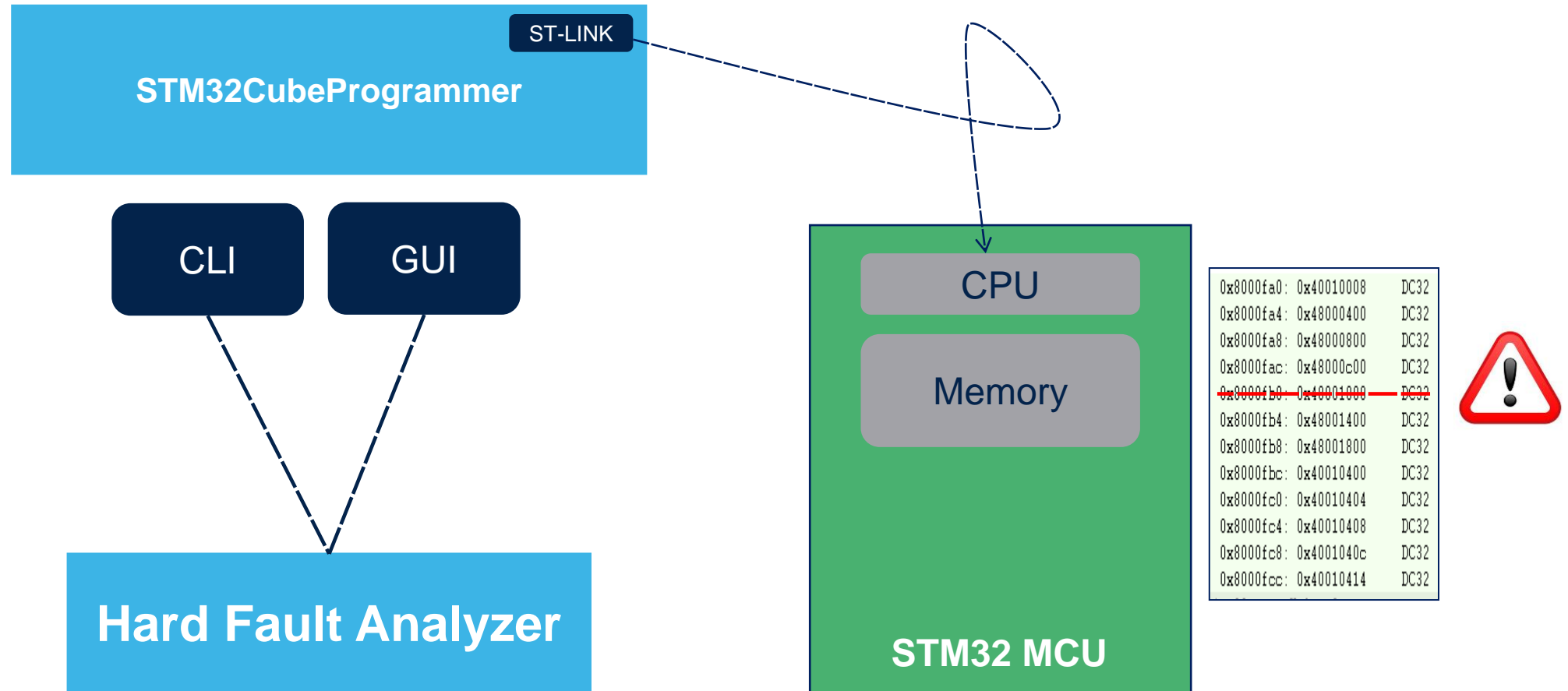


life.augmented

Hard Fault Analyzer

STM32CubeProgrammer

Overview



Hard Fault Analyzer UI

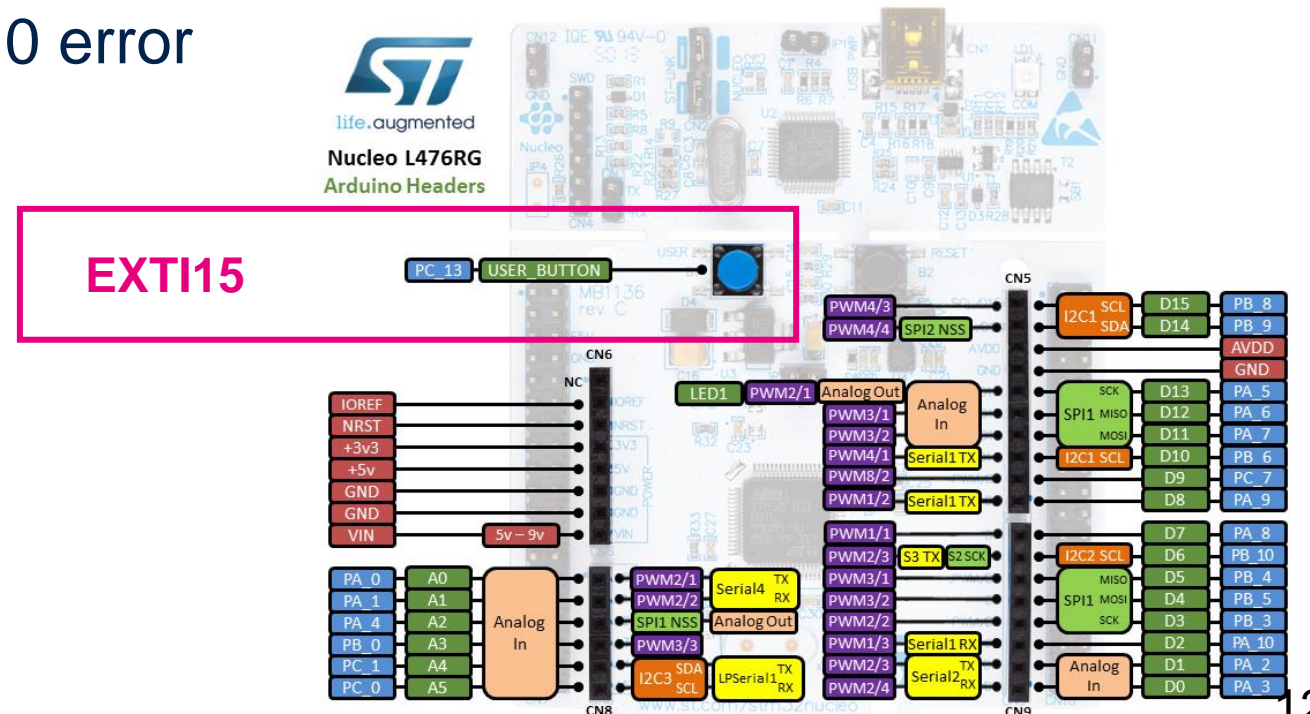
The screenshot displays the STM32CubeProgrammer interface, specifically the Hard Fault Analyzer section. The UI is divided into several panels and sections, with five numbered callouts highlighting key features:

- 1**: The top status bar showing "No Hard Fault detected" and buttons for "Start Analysis", "Clear Window", and "Note".
- 2**: The "CPU: RUNNING" status indicator and the "Reset" button.
- 3**: The "Hard Faults" section, which lists various fault types such as "Bus error on a vector read error (VECTBL)", "Fault that is escalated to a hard fault (FORCED)", "Fault on breakpoint escalation (DEBUGVT)", "Faulty instruction address", and "Faulty called function address".
- 4**: The "Usage Faults" section, which lists faults like "Undefined instruction (UNDEFINSTR)", "Attempt to enter an invalid instruction set state (INVSTATE)", "Failed integrity check on exception return (INVPC)", "Attempt to access a non-existing coprocessor (NOCP)", "Illegal unaligned load or store (UNALIGNED)", "Stack overflow (STKOF)", and "Divide by 0 (DIVBYZERO)".
- 5**: The "CPU capture during exception" section, which displays a table of registers and values, including "NVIC position", "Execution mode", and "Stack memory region".

Additional UI elements include a sidebar on the left with icons for "REG BETA", "SWV", "CPU", "OB", and "Download". The bottom right corner shows the "ST-LINK configuration" panel with settings for "Serial number", "Port", "Frequency (kHz)", "Mode", "Access port", "Reset mode", "Speed", "Shared", "Debug in Low Power mode", "External loader", "Target voltage", and "Firmware version". The "Target information" panel at the bottom right provides details about the board, device, type, device ID, revision ID, flash size, CPU, and bootloader version.

Demo 1: Division By Zero

- **Device** : STM32L476 Nucleo Board
- **Interface** : STLINK via SWD
- **Scenario** : Division By Zero from an EXTI Handler
- **Fault type** : Usage Fault, Devide by 0 error (DIVBYZERO)



Demo 2: illegal READ

- **Device** : STM32U574ZI-Q Nucleo Board
- **Interface** : STLINK via SWD
- **Scenario** : Read a non accessible memory region
- **Fault type** : Bus Fault, Precise data access error (PRECISERR)



life.augmented

Register Viewer

Overview

- Functionalities:
 - ✓ Visualize all the MCU and core registers.
 - ✓ Modify the MCU registers values.
 - ✓ Save MCU registers values into a log file.

CLI Command:

-regdump <file_path.log> [choice=<number>]

Note: Access using SWD/JTAG.



Registers - BETA

Device: STM32L0x0 Search: Peripheral: TIM21 Save to file

Name	Value	Access	Address
▶ CR1	0x00000000	ReadWrite	@ 0x40010800
▶ CR2	0x00000000	ReadWrite	@ 0x40010804
▶ SMCR	0x00000000	ReadWrite	@ 0x40010808
▶ DIER	0x00000000	ReadWrite	@ 0x4001080C
▶ SR	0x00000000	ReadWrite	@ 0x40010810
▶ EGR	WriteOnly	WriteOnly	@ 0x40010814
▶ CCMR1_Output	0x00000000	ReadWrite	@ 0x40010818
▶ CCMR1_Input	0x00000000	ReadWrite	@ 0x40010818
▶ CCER	0x00000000	ReadWrite	@ 0x40010820
▶ CNT	0x00000000	ReadWrite	@ 0x40010824
▶ PSC	0x00000000	ReadWrite	@ 0x40010828
▶ ARR	0x00000000	ReadWrite	@ 0x4001082C
▶ CCR1	0x00000000	ReadWrite	@ 0x40010834
▶ CCR2	0x00000000	ReadWrite	@ 0x40010838
▶ OR	0x00000000	ReadWrite	@ 0x40010850

Apply Read CPU: RUNNING Reset

Log

12:43:30 : ST-LINK FW : V2J29M18
12:43:30 : Board : NUCLEO-L010RB
12:43:31 : Voltage : 3.26V
12:43:31 : SWD freq : 4000 KHz
12:43:31 : Connect mode: Hot Plug
12:43:31 : Reset mode : Software reset
12:43:31 : Device ID : 0x447
12:43:31 : Revision ID : Rev Z
12:43:31 : Debug in Low Power mode is not supported for this device.
12:43:31 : UPLOADING OPTION BYTES DATA ...
12:43:31 : Bank : 0x00
12:43:31 : Address : 0x4002201c
12:43:31 : Size : 104 Bytes

100%

ST-LINK

Disconnect

ST-LINK configuration

Serial number 066EFF53515587828110...

Port SWD

Frequency (kHz) 4000

Mode Hot plug

Access port 0

Reset mode Software reset

Speed Reliable

Shared Disabled

Debug in Low Power mode ☒

External loader

Target voltage 3.26 V

Firmware version V2J29M18

Firmware upgrade

Target information

Board NUCLEO-L010RB

Device STM32L07x/L08x/L010

Type MCU

Device ID 0x447

Revision ID Rev Z

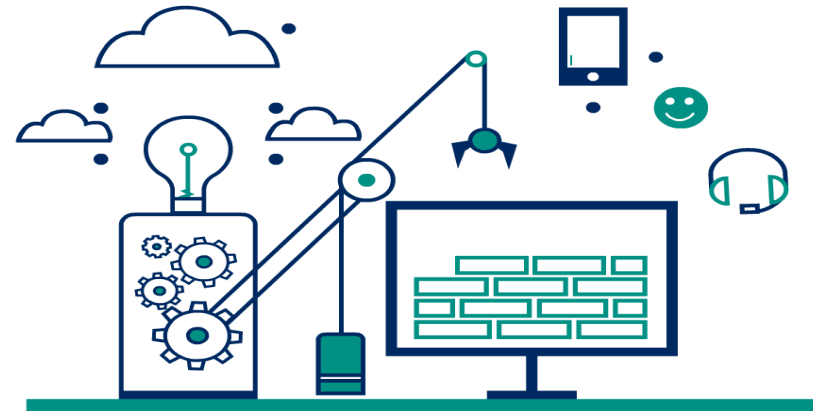
Flash size 128 KB

CPU Cortex-M0+

Bootloader Version 0xB2

Register Viewer

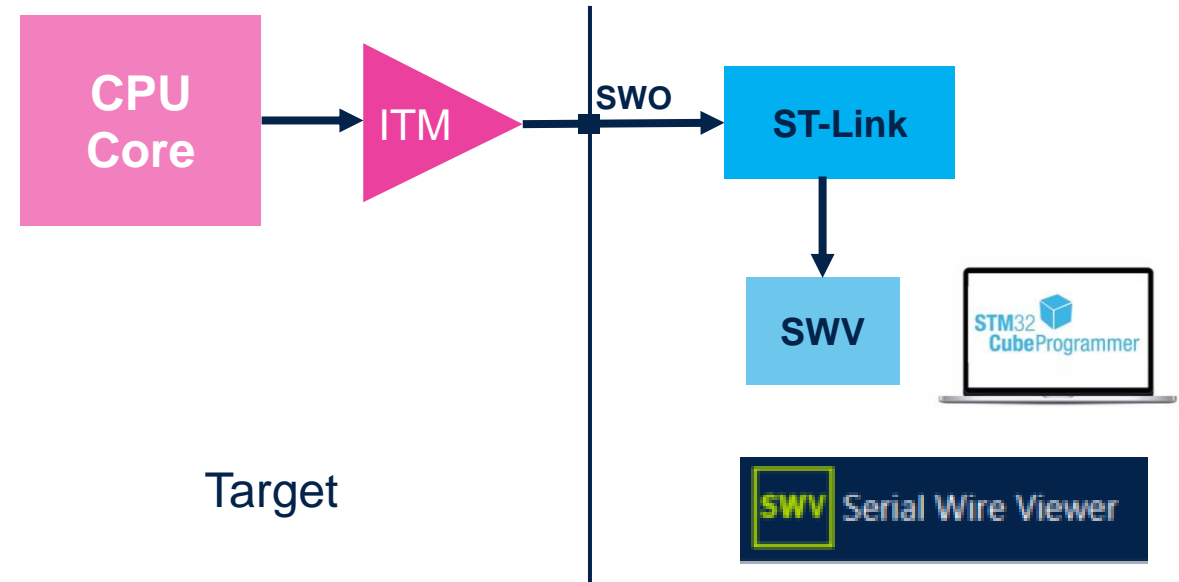
Demo



Serial Wire Viewer

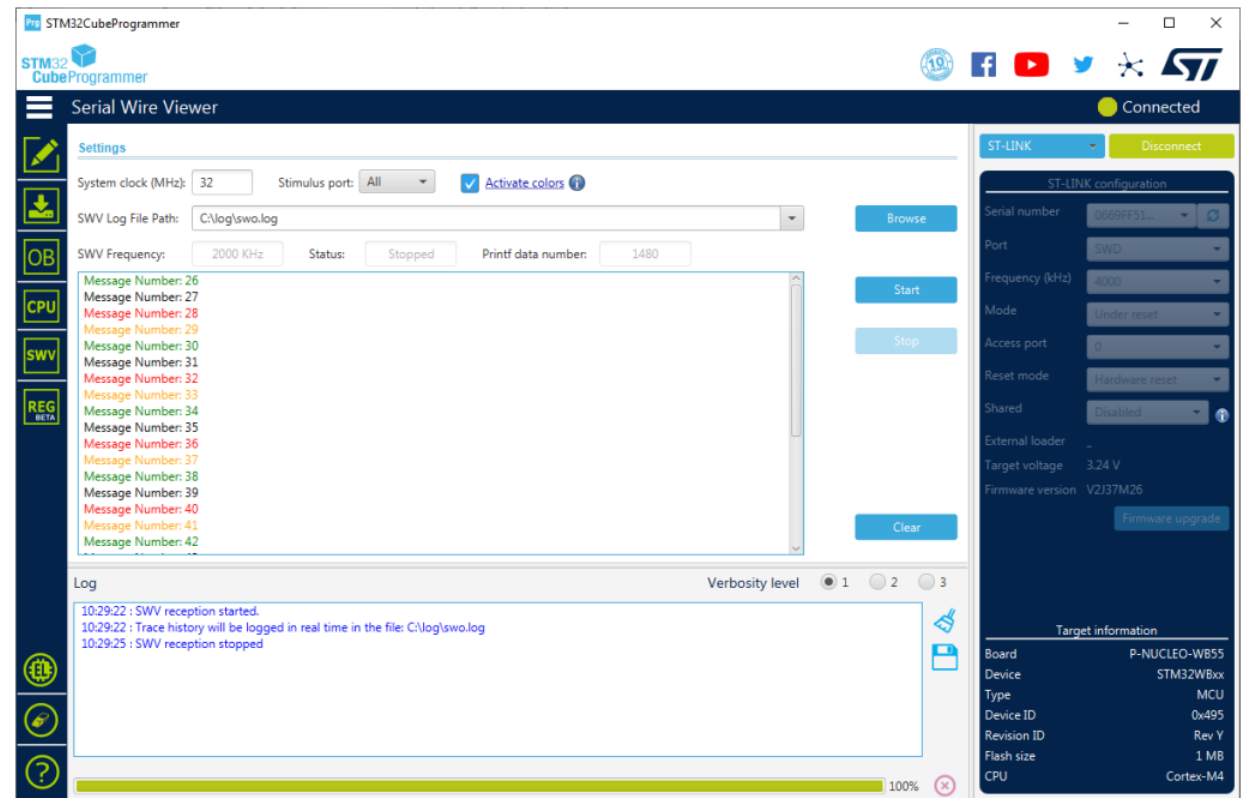
Overview

- Display printf data sent from the target
- Available through the SWD interface
- User application should configure the ITM and SWO pin to generate the trace.
- Less intrusive compared to UART/VCP.



Functional description

- Specify target system clock frequency.
- Select the corresponding stimulus port at which the application is sending data.
- Click on the start button.
- Optionally generate a .log file.
- Optionally enable colored traces output.
- Feature available on both CLI and GUI.



STM32CubeProgrammer v2.14.0 release information

- Updated the support for the entire STM32H5 series:
 - Debug authentication with password or certificate
 - Authentication key provisioning
 - Key generation
 - Firmware encryption and signing
 - Certificate generation
 - SFI support, secure manager install, and module install/update
- Added support for microcontrollers in the STM32L5 series and STM32WL series:
 - SFI integrity check
- Updated support for the STM32U5 series:
 - SFI support for the STM32U535/545, STM32U575/585, STM32U595/5A5, and STM32U599/5A9 product lines
- Added support for the STM32U5Fxxx and STM32U5Gxxx microcontrollers:
 - SFI support: user interface and command-line interface
 - SFIx support:
 - Via debug: user interface and command-line interface
 - Via bootloader: command-line interface only

Refer to the STM32CubeProgrammer software description user manual (UM2237) for details.

STLink portfolio



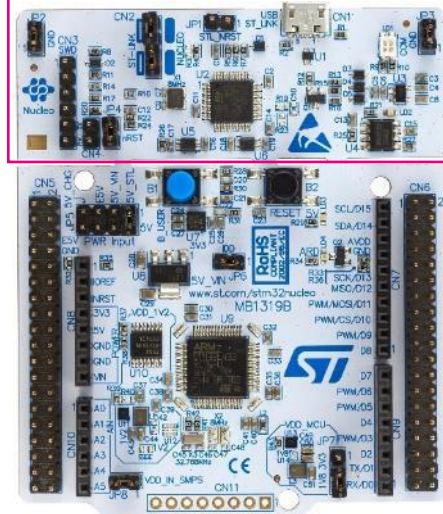
From ST-Link V2 to STLink-V3

ST-Link V2

ST-LINK/V2

ST-LINK/V2-ISOL

ST-LINK/V2-1



STLink-V3

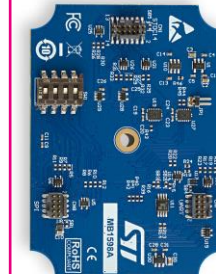
STLINK-V3MINIE

STLINK-V3MODS

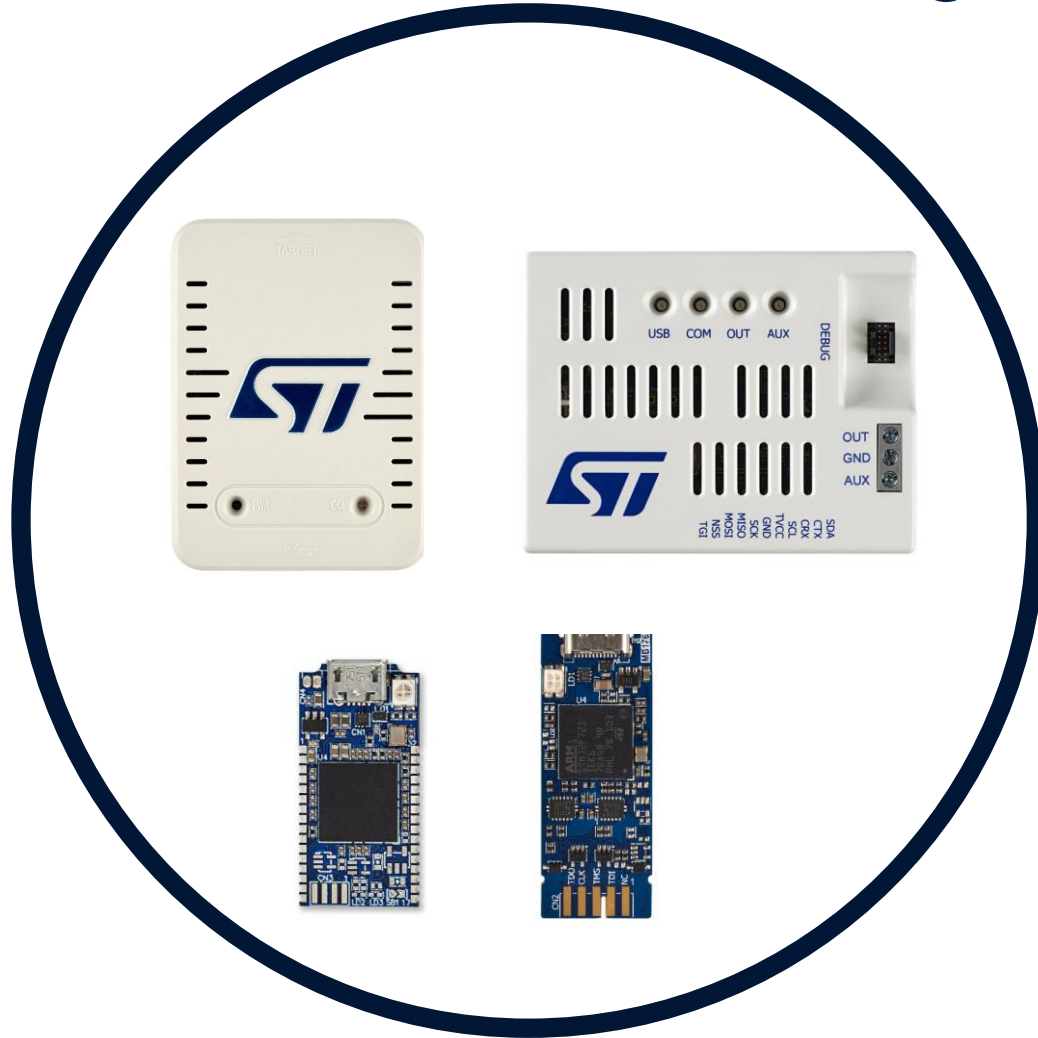
B-STLINK-VOLT

B-STLINK-ISOL

STLINK-V3SET



STLINK-V3 tools enable developers to debug / program with more efficiency



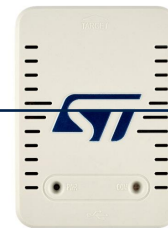
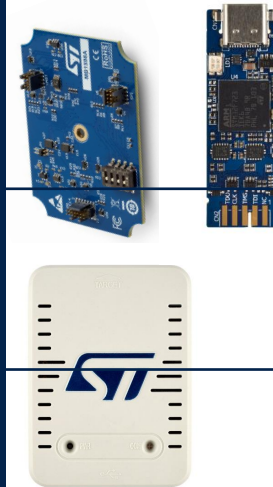
Easier

Faster

More Flexible

STLINK evolution

Energy profiler					****
Voltage adaptation 1.65-3.3 V				***	***
Virtual COM port/ USB bridge		**		**	**
Debug/Program	*				



STLink/V2

STLink-V3SET
STLink-V3MODS

B-STLink-VOLT
STLink-V3MINIE

STLink-V3PWR

...to address a large diversity of needs



STLINK-V3SET

**Easy addition of
multi-path bridge
and
debug connector
formats**

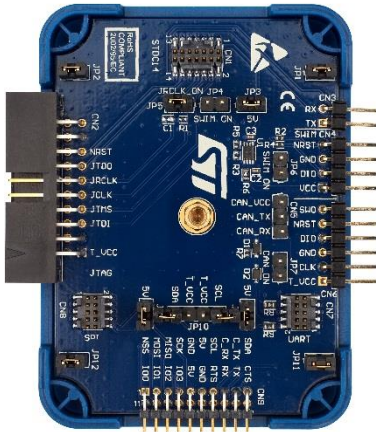


Adapter board

USB to :
UART
SPI
CAN
I2C
:
GPIOs

Connector formats :
STDC-14
JTAG-20

STLINK-V3SET a scalable debugger/programmer



STLINK-V3SET

35\$

Easier

- JTAG / SWD / SWV (STM32 debug)
- SWIM (STM8 debug)
- Drag and drop flash programming
- Virtual COM port

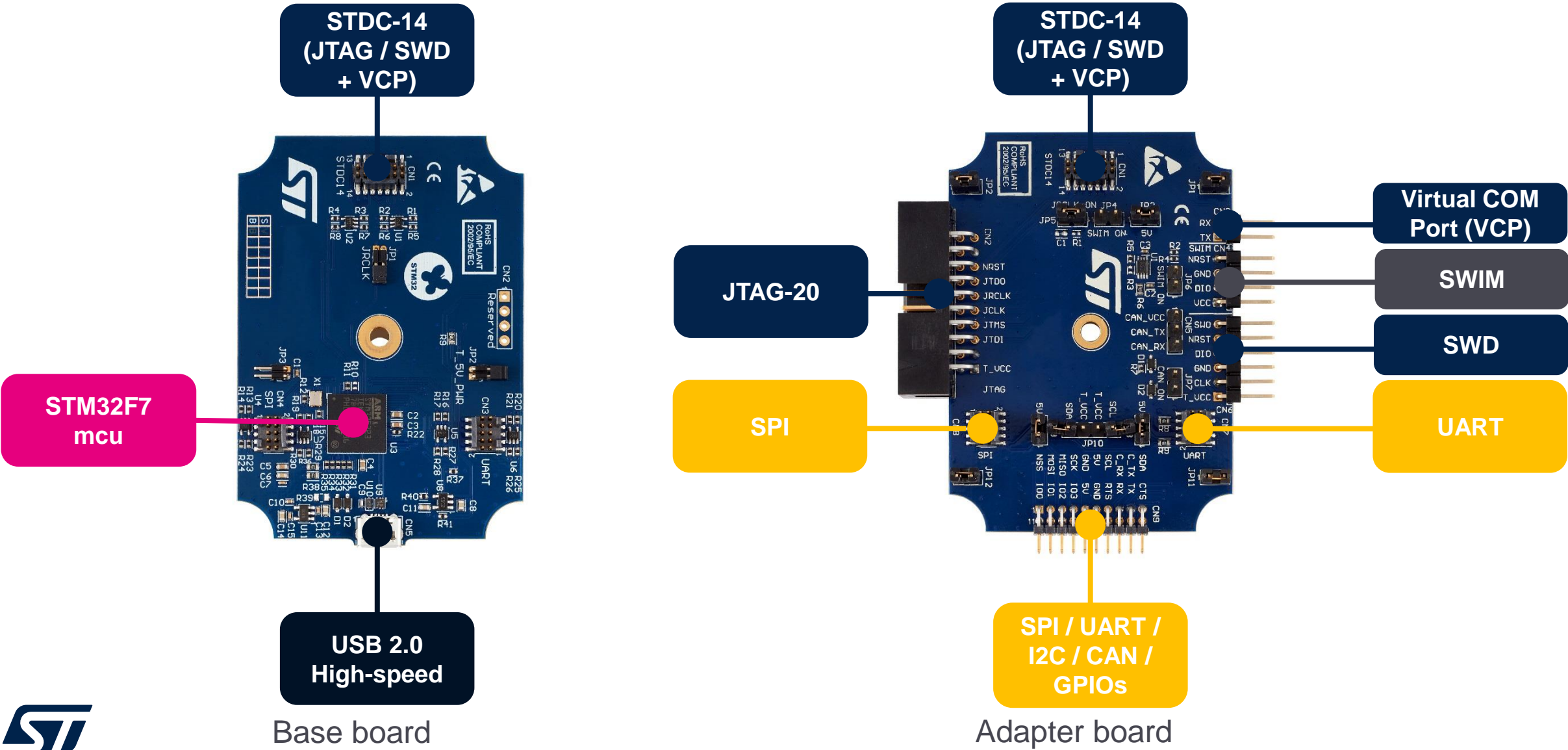
Faster

- Performance boost (vs STLink/V2)
- Optimized algorithms
- USB 2.0 High Speed interface

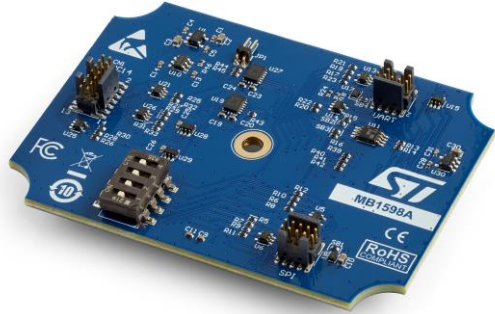
More flexible

- Extension boards
- Multi-path bridge (through adapter board)

STLINK-V3SET anatomy



STLINK extension boards



B-STLINK-VOLT

20\$

Voltage adaptation

- 1.65 – 3.3 V voltage adaptation for debug / virtual COM port / bridge signals
- Compatible with STLINK-V3SET casing



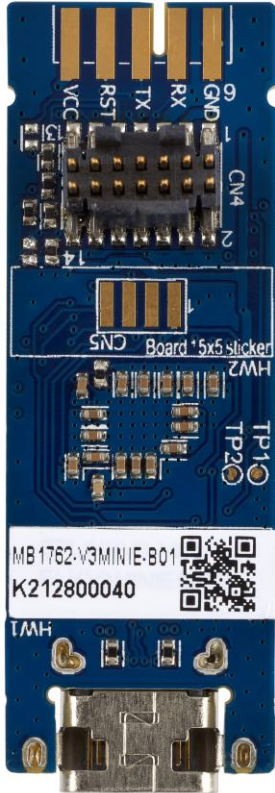
B-STLINK-ISOL

40\$

Galvanic isolation & Voltage adaptation

- 1.65 – 3.3 V galvanic isolation and voltage adaptation for debug / virtual COM port / bridge signals
- Compatible with STLINK-V3SET casing

Small STLINK-V3MINIE makes big difference



STLINK-V3MINIE

11\$

More performance at lower price (vs STLink/V2)

- JTAG / SWD / SWV debug interfaces
- USB Type-C High Speed interface

Facilitate in-the-field FW update

- Virtual COM port
- Tiny size 15 x 42 mm
- 3D printer files for slim casing

Direct support of power-constrained IoT products

- Voltage adaptation 1.65-3.3V

STLINK-V3MODS, a ready-to-use module for your custom boards



STLINK-V3SET



STLINK-V3MODS

8.25\$

Performance boost (vs STLink/V2)

- JTAG / SWD / SWV debug interfaces
- USB 2.0 High Speed interface

Small size, small price

- Footprint 15 x 30 mm
- Price below 9\$

Multi-path bridge

- Virtual COM port
- Drag and drop programming
- USB to UART / SPI / I2C / CAN / GPIOs

STLINK comparison

Features	STLink/ V2	STLink- V3MODS	STLink- V3MINIE	STLink- V3SET	B-STLink- ISOL	STLink- V3PWR
Mcus supported	STM32 STM8	STM32	STM32	STM32 STM8	STM32	STM32
Casing	*	(on-board module)	STL file	*	V3SET extension	*
USB	Full Speed (12Mbit/s)	High Speed (480Mbit/s)	High Speed (480Mbit/s)	High Speed (480Mbit/s)	V3SET extension	High Speed (480Mbit/s)
SWD max read data rate	150 kByte/s	800 kByte/s	800 kByte/s	800 kByte/s		475 kByte/s
Virtual COM port	No	16 MHz	16 MHz	16 MHz	10 MHz	12 MHz
Multi-path bridge	No	*	No	*	*	*
Target voltage	3.3V	3.3 V	1.65-3.3 V	3.3 V	1.65-3.3 V	1.6-3.6 V
Energy profiler						*
Price	21\$	8.25\$	11\$	35\$	40\$	95\$

Software support and Product references

STLINK-V3MINIE / V3MODS
STLINK-V3SET / V3PWR


STM32
CubeMonitor

STM32CubeMonitor
STM32CUBEMON


STM32
CubeProgrammer

STM32CubeProgrammer
STM32CUBEPROG


STM32
CubeIDE

STM32CubeIDE

arm KEIL

IDE toolset
MDK-ARM



IDE toolset
EWARM

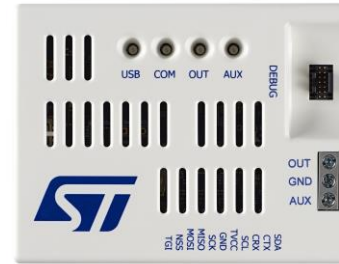


life.augmented



STLink-V3Power energy profiling tool

...Enables energy profiling of any STM32 MCU



Dvt tools	Power shield	Energy meter	STLINK-V3 Power
Dynamic current ranges	100nA / 50mA	300nA / 150mA	nA up to 500mA
Target MCUs	STM32L0, L4 STM32WB	STM32L0, L4, STM32L5, U5, STM32WB, WL	All STM32 MCUs
	X-NUCLEO-LPM01A	STM32L562E-DK	STLINK-V3PWR

Optimize the energy efficiency of STM32 mcu applications

Measure dynamic current consumption with high accuracy

Visualize energy consumption with STM32CubeMonitor-Power SW tool

Debug code in sync with energy consumption measurements

Direct support of Keil and IAR IDEs for power profiling

Programmer with multi-path bridge



STLINK-V3PWR

95\$

STLINK-V3PWR powerful easy-to-use energy profiler



Current measurement
with wide dynamic range
nA-500mA

High accuracy (up to +/-0.5%)
Resolution up to 2nA

Programmable output
voltage source
1.6 - 3.6V (under up to 2A)

STLINK-V3PWR overview

USB Type-C high-speed interface

- Simultaneous access of CubeMonitor-Power and IDEs
- Fast programming with CubeProgrammer

Debugger / Programmer

- SWD / JTAG (STDC14 connector)
- Virtual COM port

Multi-path bridge USB to UART / SPI / I2C / CAN / GPIOs

Debug code in sync with power consumption measurements

- Programmable voltage
supply for target



Visualize power data with STM32CubeMonitor-Power tool



Display

- Graphical rendering in real-time (up to 100 kSPS)
- Acquisition log over large period of time

Analyze

- Intuitive zoom and navigation into energy consumption data

Benchmark

- Fast computation of EEMBC ULPMark-CP scores

Software support and Product references

STLINK-V3PWR

 STM32
CubeMonitor-Power

STM32CubeMonitor-Power
STM32CUBEMONPWR

From release v1.2

 STM32
CubeProgrammer

STM32CubeProgrammer
STM32CUBEPROG

From release v2.13

 STM32
CubeIDE

STM32CubeIDE

From release v1.12

 arm KEIL

IDE toolset
MDK-ARM





From release v5.38a

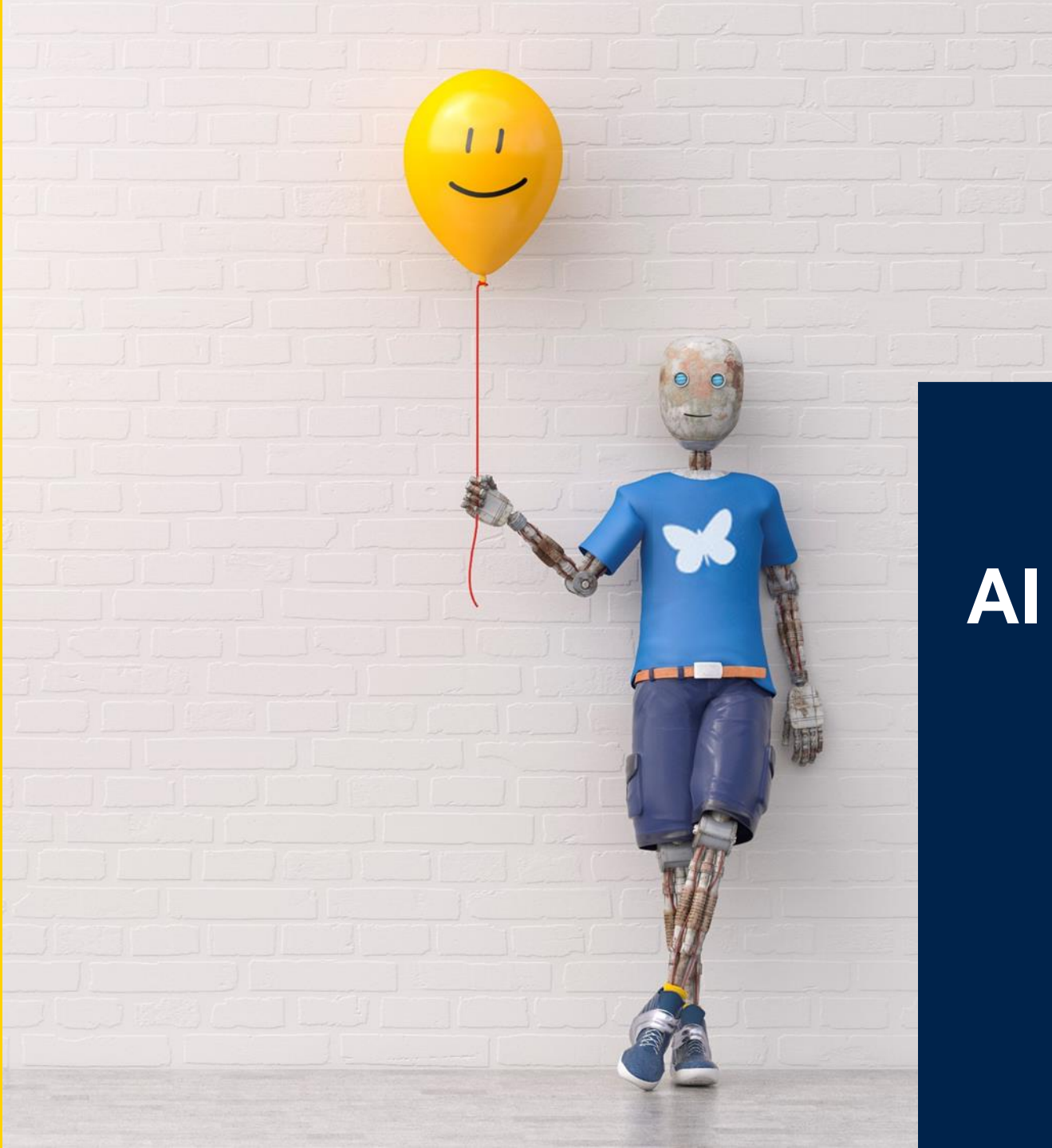
 iar

IDE toolset
EWARM

From release v9.32.2

Summary slide

	Configuration	Code generation	Code editor	Debug	Simple Programming	Advanced programming	Monitoring
	Main feature	Main feature					
	Available thanks to embedded STM32CubeMx Features	Available thanks to embedded STM32CubeMx Features	Main feature	Main feature	Programming of the flash		Limited monitoring via SWD + IDE feature
					Main feature	Main feature Please refer to the slides for details	
							Main feature Advanced monitoring



life.augmented

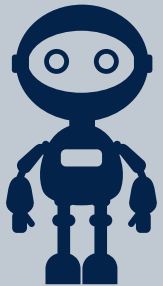
AI solutions

What is AI?

The evolution of AI

Artificial Intelligence (AI)

Early Artificial Intelligence stirs excitement



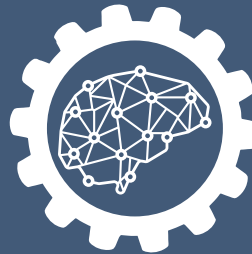
Machine Learning (ML)

Machine Learning begins to develop



Deep Learning

Deep Learning breakthroughs drive AI boom



Any technique that enables computer to mimic **human behavior**

Subset of AI. Algorithms and methodologies that improve over time through **learning from data**

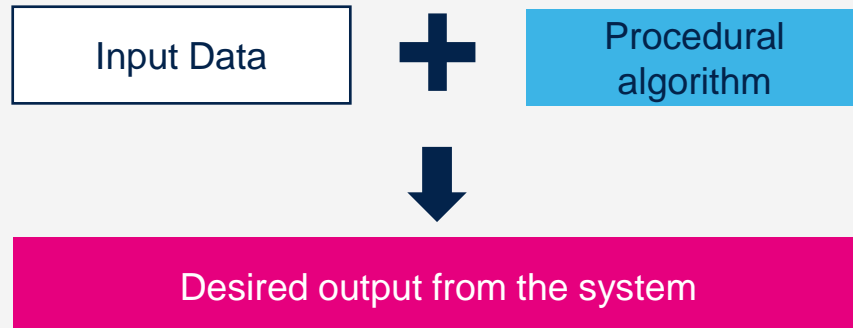
Subset of ML. Learning algorithms that derive meaning from a **huge amount of data**, by using a hierarchy of multiple layers that **mimic the neural networks of the human brain**

1950 1960 1970 1980 1990 2000 2010 2020

A new way to add environment awareness to your products

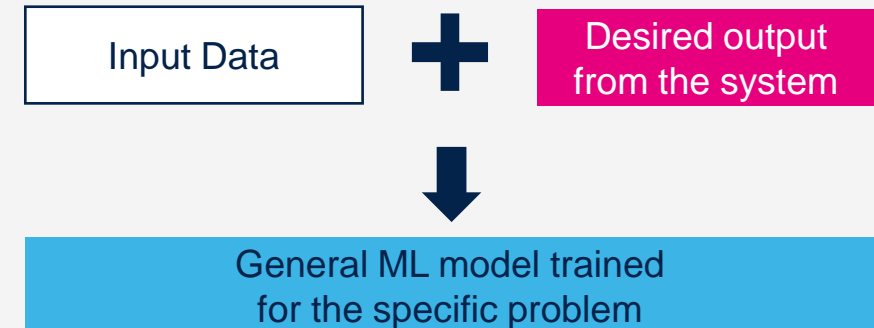
Create more robust software using Machine Learning on STM32

Standard programming Handcrafted rules based on experience



- Requires domain expertise to code
- Need to rewrite if environment evolves

Machine Learning Rules learnt from real-world data



- Generate code from real-world observations
- Re-learn from data if environment evolves

AI is used today in almost every market segment

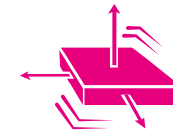
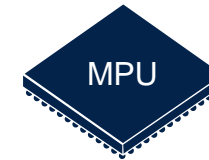
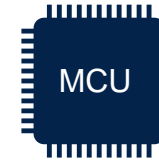


Embedded AI technology trend

**“Global Shipments of Deep Edge AI Devices
to Reach 2.5 Billion by 2030”**

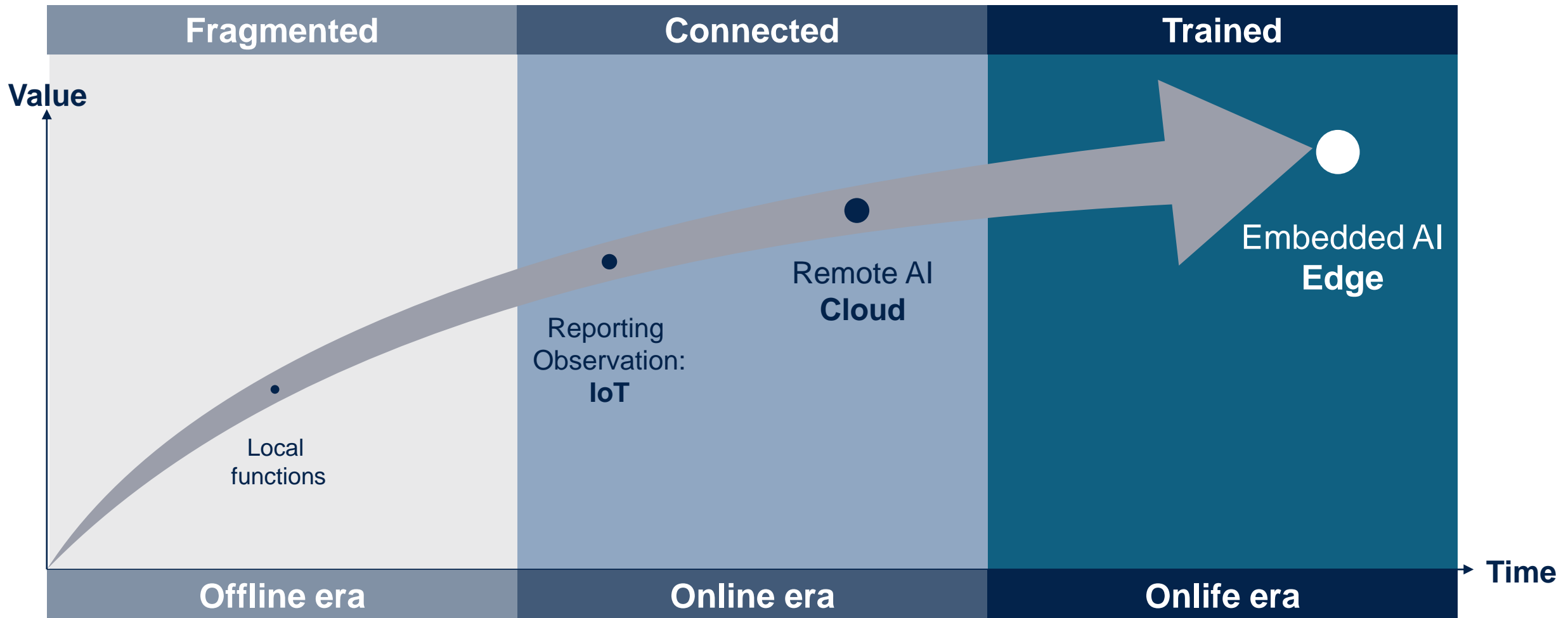
Source: ABI Research

AI technologies are now
embedded inside end devices
(MPU, MCU and sensors).



Growing community and ecosystem of **Deep Edge AI** technologies focusing on standalone, low-power and cost-efficient embedded solutions.

The quest for an ever-SMARTER infrastructure



Artificial intelligence at the Edge

Moving part of Artificial Intelligence closer to the data acquisition brings several benefits



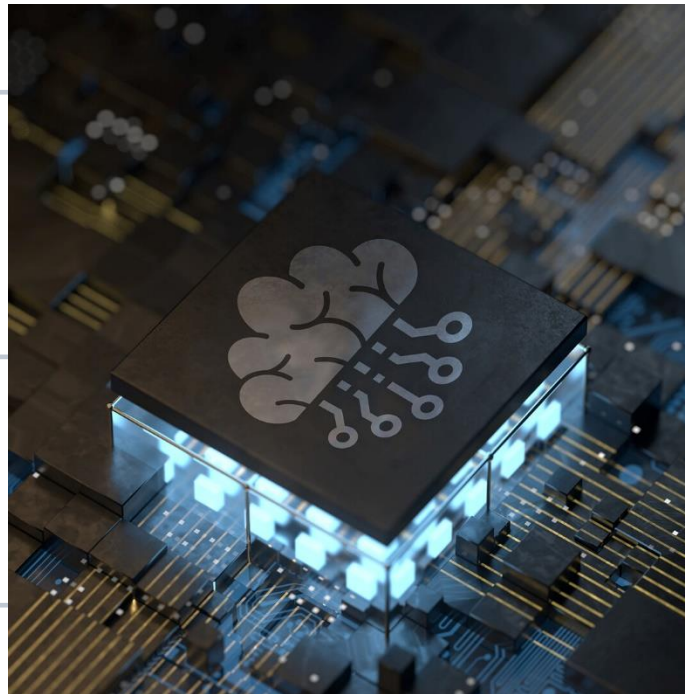
Ultra-low latency
Real-time applications



More reliability



Security of data
No sharing in the cloud



Privacy by design
GDPR compliant



Sustainable on energy
Low-power consumption



Better user experience

A step too big to climb

Organizations main challenges – AI Skills & datasets

FAILURE RATE FOR AI/OT PROJECTS

(Cisco Connected Research Futures)

74%

Lack of adequate data sets

15Months

Average time to collect & label data
(Mc Kinsey)

Lack of data scientists

130,000

Data Scientist shortfall in the US
(IBM Quant Crunch Report)

EMBEDDED DEVELOPERS

(IDC)

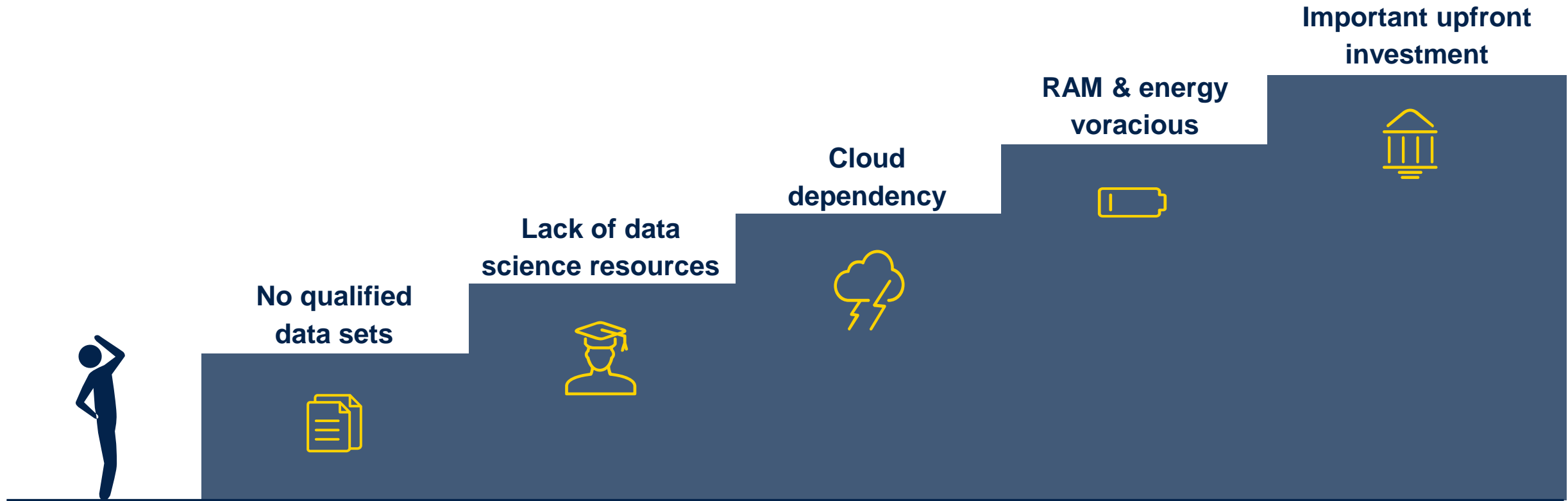
1.2Million

0.2%

With some AI skills

For most companies, creating an edge AI device is a long journey with extraordinary challenges

Investment, complexity and development time are often barriers to AI adoption















Start today with deep edge AI

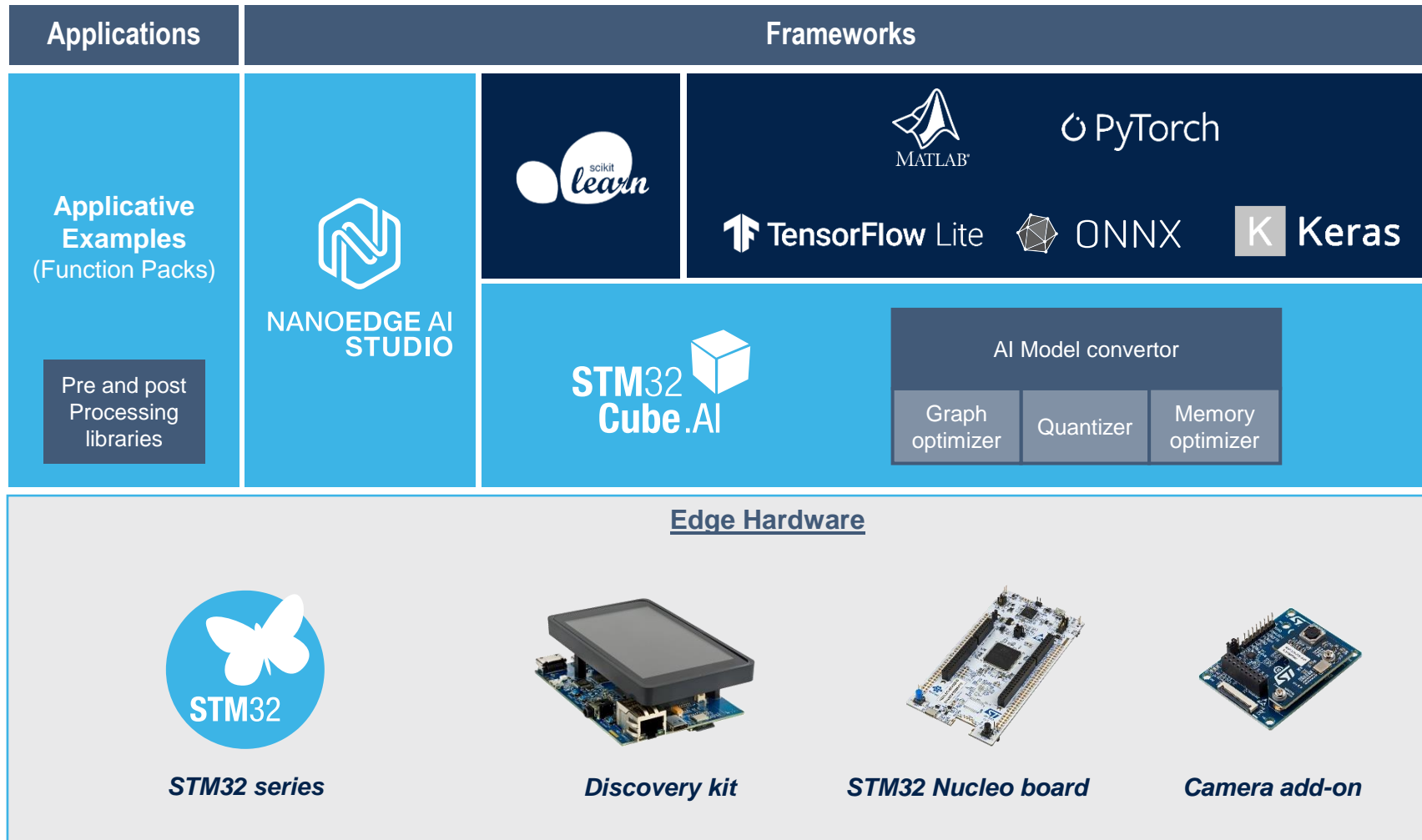
**“ If only
I had solutions to overcome
AI design challenges**

This is where we come in

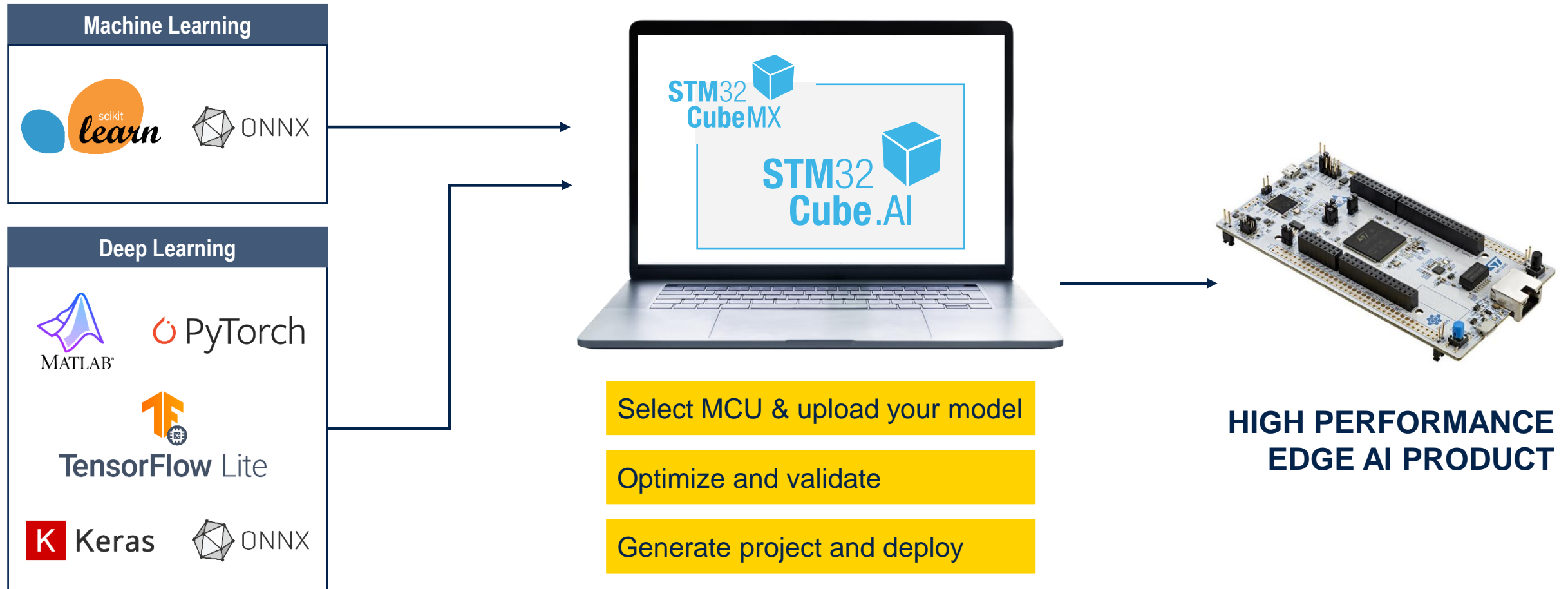
Whatever your company's profile, you will find an AI solution suited to your needs

COMPANY'S PROFILE	USE CASES		
	Anomaly detection	Classification	Deep Learning
  Embedded developers <ul style="list-style-type: none">▪ No dataset available▪ No dedicated AI Team			 
  Team with AI expertise <ul style="list-style-type: none">▪ Dataset available▪ AI Team		 	

STM32 comprehensive AI ecosystem



A tool to seamlessly integrate AI in your projects



For embedded developers

NanoEdge™ AI Studio, an automated ML design solution

NANOEDGE AI
STUDIO 



A unique solution thought from scratch

HOW DID WE DO IT?

We re-wrote, from the algebra, ML and signal processing algorithms so that they can **LEARN** and **INFER** inside STM32

- Patented technology
- Designed for embedded developers
- Ultra memory efficient (Flash and RAM)
- Unsupervised learning in the device
- No dataset need
- Superior security
- Small footprint, runs on any STM32 microcontroller
- Close to 100% accuracy and confidence
- For anomaly detection, classification and extrapolation problems¹⁶⁷

For teams with AI expertise

STM32Cube.AI helps you accelerate your embedded development



Easily evaluate, convert and deploy Machine Learning and Deep Neural Networks on STM32

An AI extension integrated with STM32Cube MCU development environment to **OPTIMIZE** and **TUNE** models, directly on target

- Develop and train your model with major AI frameworks



- Best ML performance on STM32 (MLPerf™ Tiny benchmarks)
- Validate performance directly on target
- Small footprint, runs on any STM32
- To address any kind of problem with ML or CNN libraries



Making Edge AI accessible to all STM32 portfolio

STM32Cube.AI & NanoEdge AI are compatible with all STM32 series



MPU

STM32MP1

4158 CoreMark
Up to 800 MHz Cortex –A7
209 MHz Cortex –M4



High Perf
MCUs

STM32F3

245 CoreMark
72 MHz Cortex-M4

STM32G4

569 CoreMark
170 MHz Cortex-M4

STM32F2

Up to 398 CoreMark
120 MHz Cortex-M3

STM32F4

Up to 608 CoreMark
180 MHz Cortex-M4

STM32F7

1082 CoreMark
216 MHz Cortex-M7

STM32H7

Up to 3224 CoreMark
Up to 550 MHz Cortex -M7
240 MHz Cortex -M4

Optimized for mixed-signal Applications



Mainstream
MCUs

STM32F0

106 CoreMark
48 MHz Cortex-M0

STM32G0

142 CoreMark
64 MHz Cortex-M0+

STM32F1

177 CoreMark
72 MHz Cortex-M3



Ultra-low Power
MCUs

STM32L0

75 CoreMark
32 MHz Cortex-M0+

STM32L1

93 CoreMark
32 MHz Cortex-M3

STM32L4

273 CoreMark
80 MHz Cortex-M4

STM32L4+

409 CoreMark
120 MHz Cortex-M4

STM32L5

443 CoreMark
110 MHz Cortex-M33

STM32U5

651 CoreMark
160 MHz Cortex-M33



Wireless
MCUs

STM32WL

162 CoreMark
48 MHz Cortex-M4
48 MHz Cortex-M0+

STM32WB

216 CoreMark
64 MHz Cortex-M4
32 MHz Cortex-M0+

Latest product generation

Integrate your ML models more easily with our application-oriented code examples

Time series-based monitoring



FP-AI-MONITOR1

- Predictive maintenance and much more sensor-monitoring apps
- Runs Libraries from NanoEdge™ AI Studio

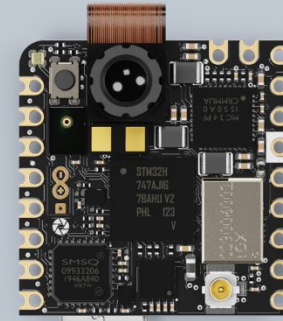
Audio and Sensing



FP-AI-SENSING1

- Human Activity Recognition
- Acoustic Scene Classification
- Data logging, labeling and result on BLE applications

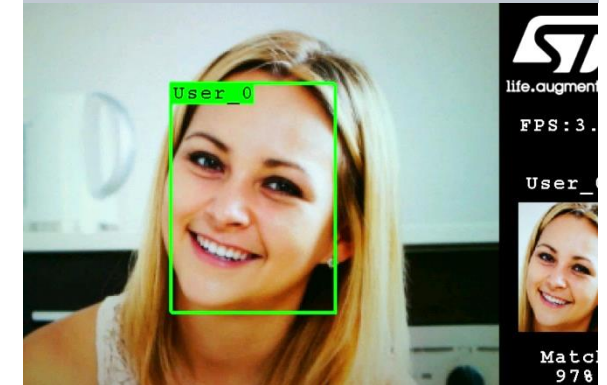
Computer Vision



FP-AI-VISION1

- Food recognition (CNN)
- Person presence detection (CNN)
- People counting (Object detection NN)
- Image processing Library

Face recognition

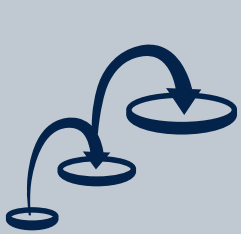


FP-AI-FACEREC1

- Face detection and recognition
- Fully functional without cloud connection

We provide everything to kick off your project

Design documentation



Getting started

Be guided step-by-step to learn STM32 ecosystem

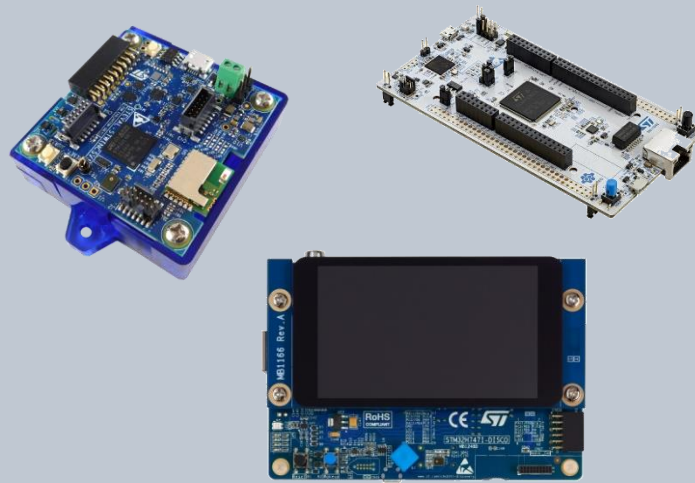


Development zone

Get started on application development and project sharing

- **Wiki by ST** is a great forum to learn and start developing AI on STM32!
- Videos of application examples
- Massive Open Online Course (MOOC)

Hardware and software tools



- Evaluation platforms for STM32 MCU/MPU
- Extra sensor boards
- Full software suite

Support & Updates



- **ST Community:** STM32 ML & AI group
- Competence center
- Distributor certified FAE
- Support center
- Newsletter

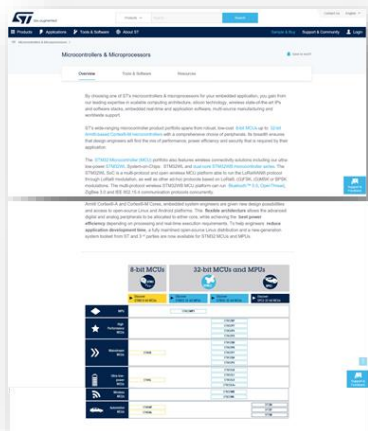
Stay focused on your expertise, we bring you everything else



Information and sharing: ST worldwide

Get connected to STM32 world!

st.com/stm32



Information



[ST MCU Finder](#)



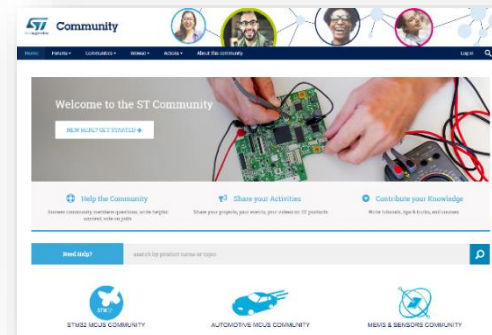
Product Selector



community.st.com



github.com/STMicroelectronics



Community/GitHub



facebook.com/stm32



youtube.com/STonlineMedia



twitter.com/@ST_World



linkedin.com/stmicroelectronics



21IC



YouKu

Social Media

► STM32 Education is now available [here](http://www.st.com/stm32education) (MOOC, OLT, Textbooks, Training Courses)

www.st.com/stm32education

Our technology starts with You



Find out more at www.st.com

© STMicroelectronics - All rights reserved.

ST logo is a trademark or a registered trademark of STMicroelectronics International NV or its affiliates in the EU and/or other countries.

For additional information about ST trademarks, please refer to www.st.com/trademarks.

All other product or service names are the property of their respective owners.



life.augmented