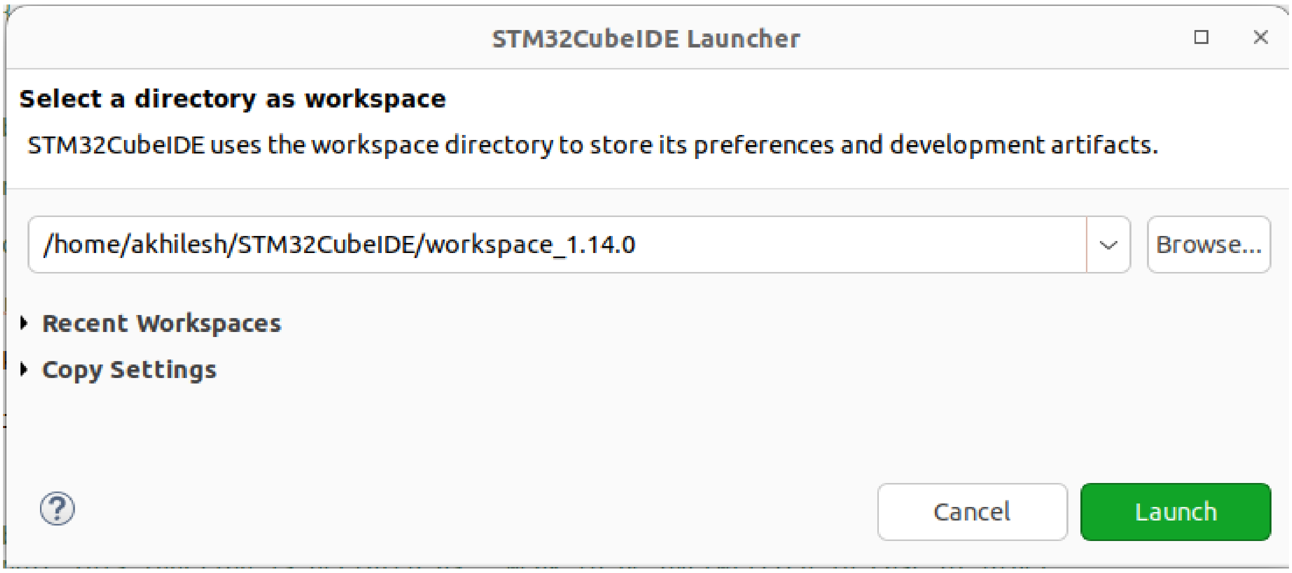
Steps for creating Workspace for FreeRTOS in STM32cube IDE

-Akhilesh Yadav

Step 1:-

Create your RTOS workspace........

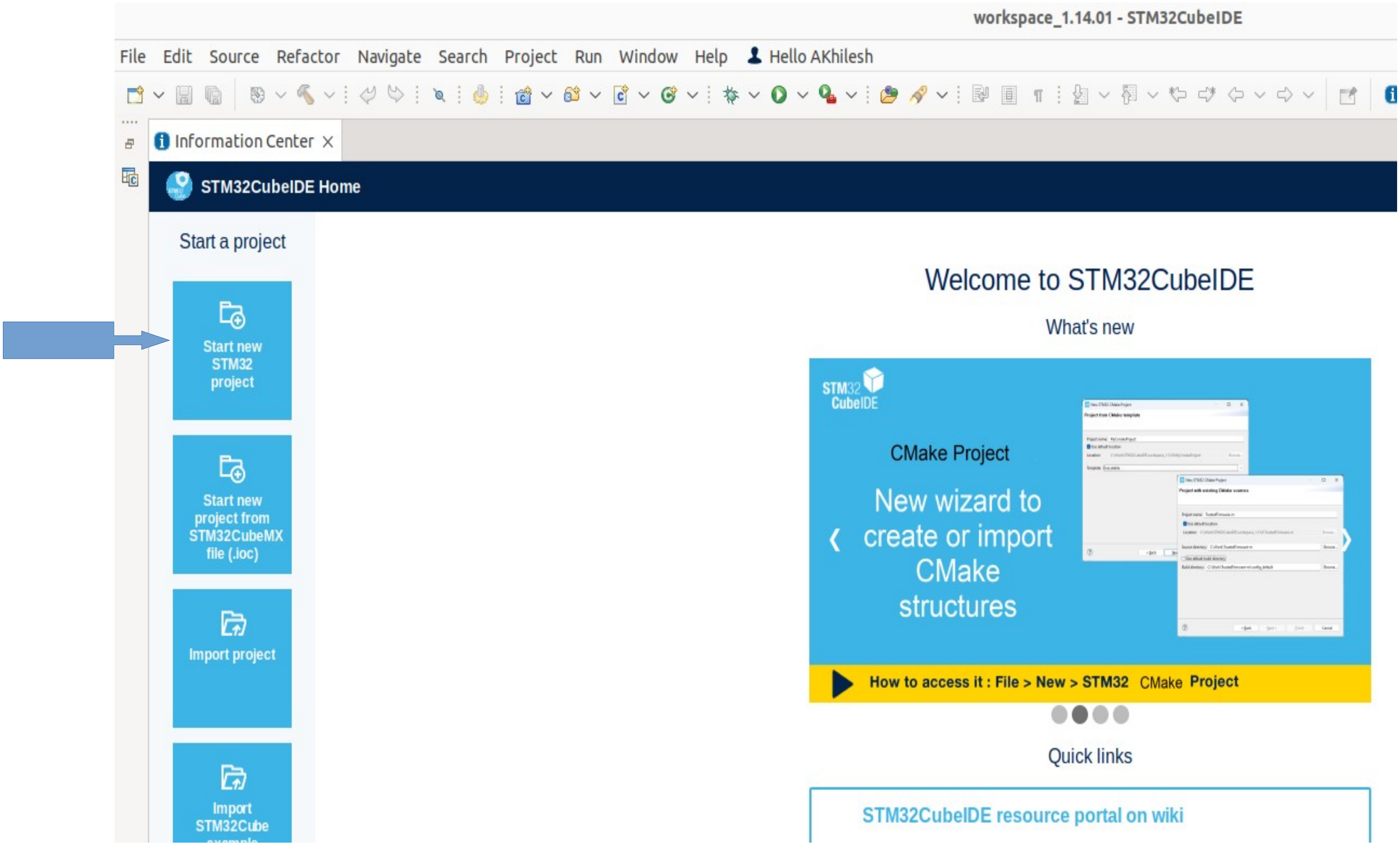


STM32CubeIDE Launcher

Select a directory as workspace

STM32CubeIDE uses the workspace directory to store its preferences and development artifacts.

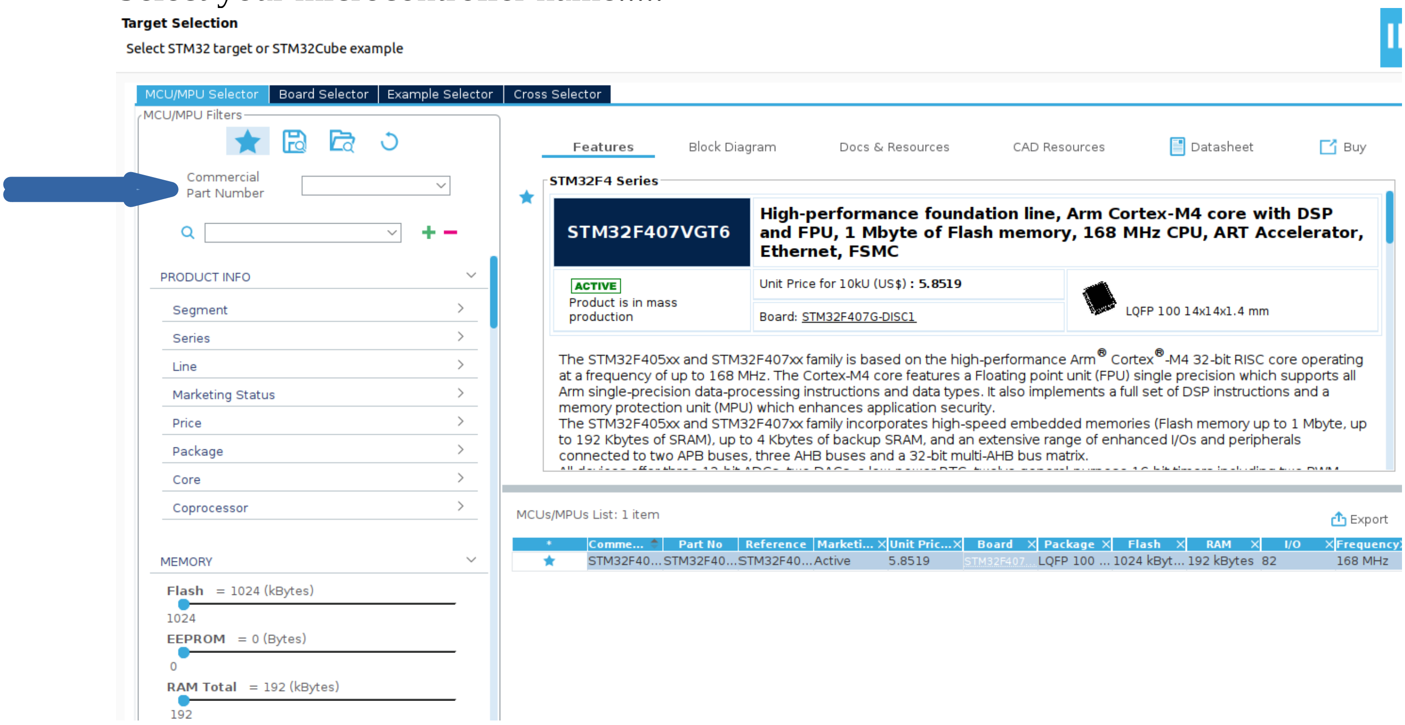
/home/akhilesh/STM32CubeIDE/workspace_1.14.0



workspace_1.14.01 - STM32CubeIDE

File Edit Source Refactor Navigate Search Project Run Window Help
Hello AKhilesh
Information Center
STM32CubeIDE Home
Start a project
Start new STM32 project
Start new project from STM32CubeMX file (.ioc)
Import project
Import STM32Cube example
Welcome to STM32CubeIDE
What's new
STM32 CubeIDE
CMake Project
New wizard to create or import CMake structures
How to access it : File > New > STM32 CMake Project
Quick links
STM32CubeIDE resource portal on wiki

Select your microcontroller name.....



Target Selection

Select STM32 target or STM32Cube example

Commercial Part Number

Flash = 1024 (kBytes)
1024

EEPROM = 0 (Bytes)
0

RAM Total = 192 (kBytes)
192

STM32F4 Series

STM32F407VGT6

High-performance foundation line, Arm Cortex-M4 core with DSP and FPU, 1 Mbyte of Flash memory, 168 MHz CPU, ART Accelerator, Ethernet, FSMC

ACTIVE
Product is in mass production

Unit Price for 10kU (US$) : **5.8519**

Board: STM32F407G-DISC1

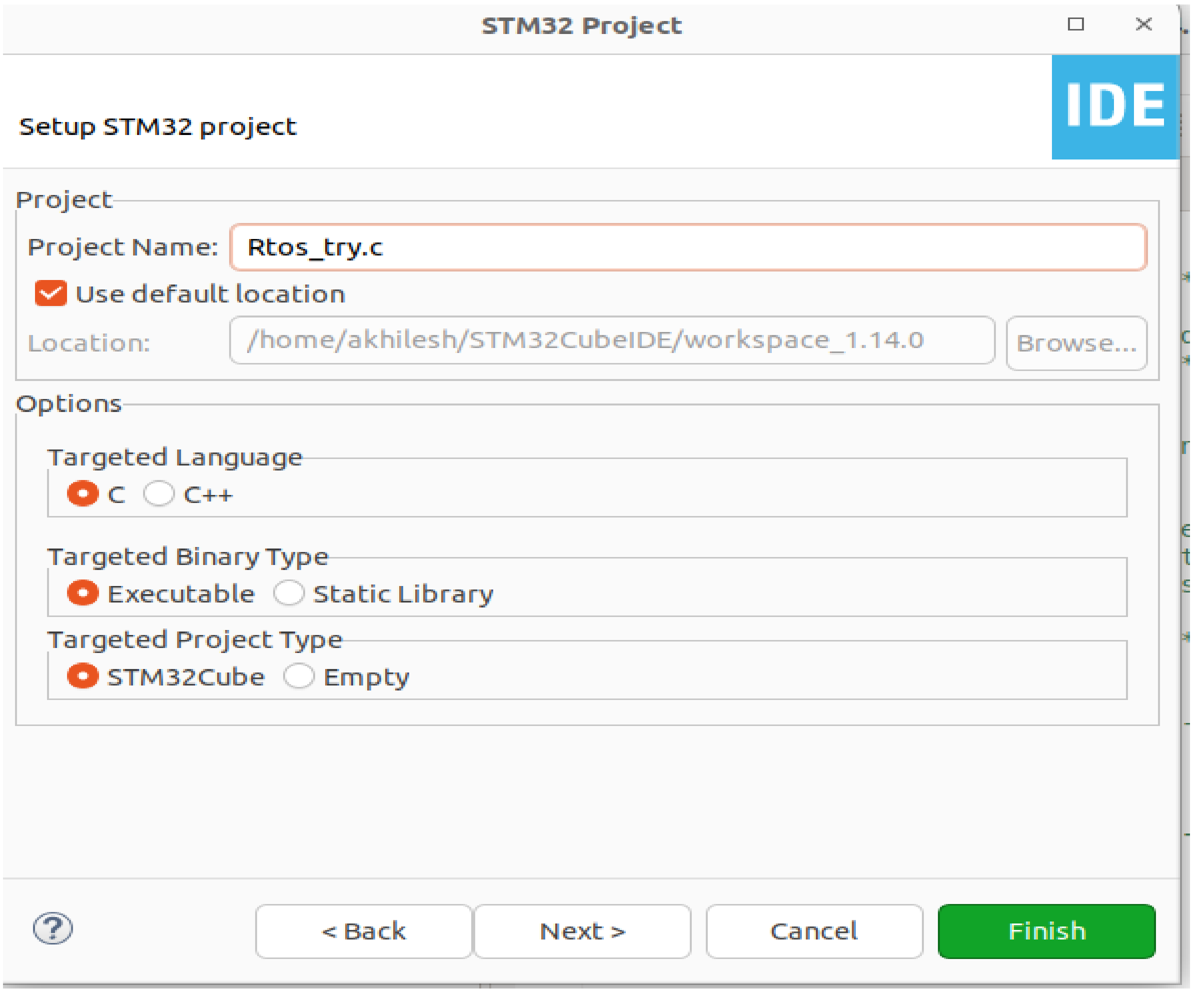
LQFP 100 14x14x1.4 mm

The STM32F405xx and STM32F407xx family is based on the high-performance Arm® Cortex®-M4 32-bit RISC core operating at a frequency of up to 168 MHz. The Cortex-M4 core features a Floating point unit (FPU) single precision which supports all Arm single-precision data-processing instructions and data types. It also implements a full set of DSP instructions and a memory protection unit (MPU) which enhances application security.
The STM32F405xx and STM32F407xx family incorporates high-speed embedded memories (Flash memory up to 1 Mbyte, up to 192 Kbytes of SRAM), up to 4 Kbytes of backup SRAM, and an extensive range of enhanced I/Os and peripherals connected to two APB buses, three AHB buses and a 32-bit multi-AHB bus matrix.

MCUs/MPUs List: 1 item

*	Comme...	Part No	Reference	Marketi...	Unit Pric...	Board	Package	Flash	RAM	I/O	Frequency
★	STM32F40...	STM32F40...	STM32F40...	Active	5.8519	STM32F407...	LQFP 100 ...	1024 kByt...	192 kBytes	82	168 MHz

Create a .c file........



STM32 Project

Setup STM32 project

Project

Project Name: Rtos_try.c

☑ Use default location

Location: /home/akhilesh/STM32CubeIDE/workspace_1.14.0

Options

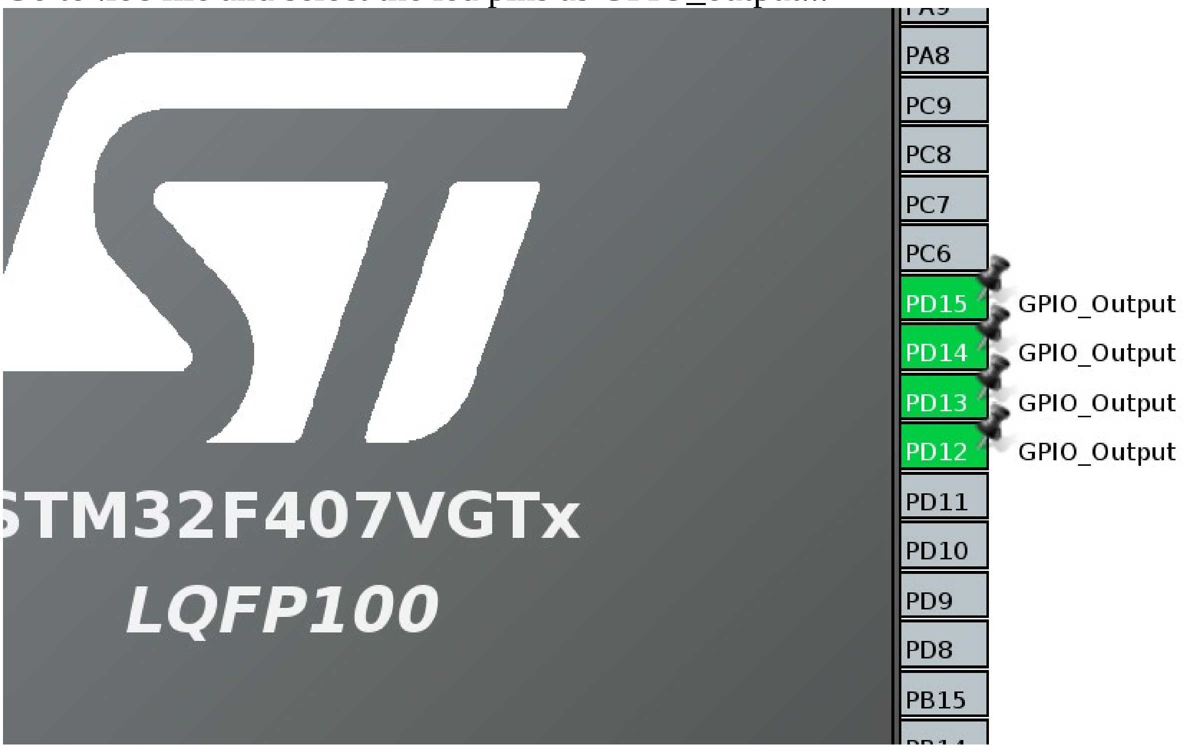
Targeted Language
● C ○ C++

Targeted Binary Type
● Executable ○ Static Library

Targeted Project Type
● STM32Cube ○ Empty

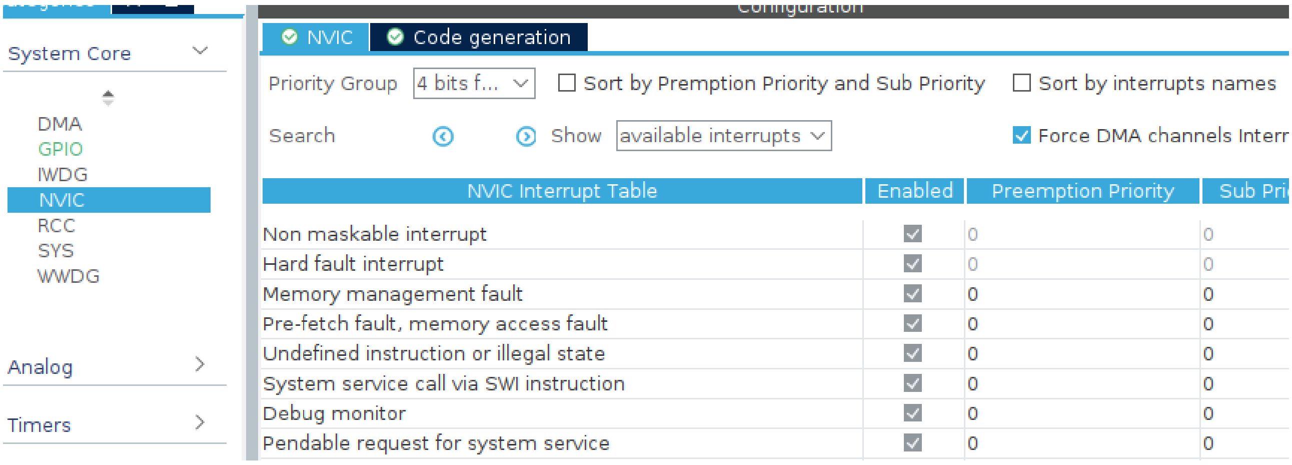
Select Finish!

Go to .ioc file and select the led pins as GPIO_output...



PD15 GPIO_Output
PD14 GPIO_Output
PD13 GPIO_Output
PD12 GPIO_Output

Setting up the workspace for FreeRTOS kernel will require setting of NVIC and Systick settings.
Since FreeRTOS needs a time base source for the Sys tick counting.

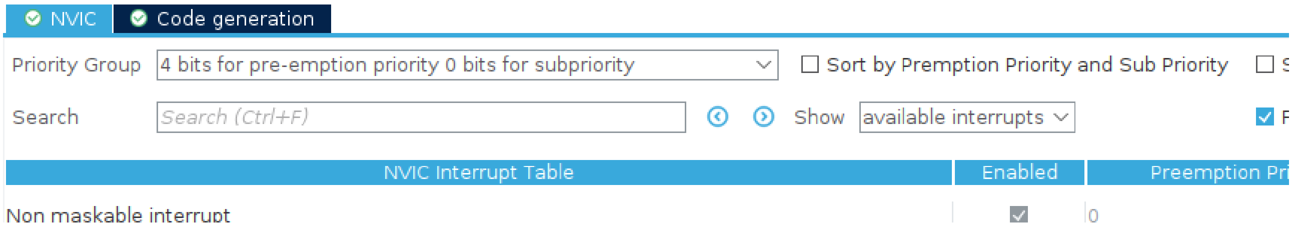


Priority Group 4 bits f... □ Sort by Premption Priority and Sub Priority □ Sort by interrupts names

Search Show available interrupts ☑ Force DMA channels Interr

NVIC Interrupt Table	Enabled	Preemption Priority	Sub Pri
Non maskable interrupt	☑	0	0
Hard fault interrupt	☑	0	0
Memory management fault	☑	0	0
Pre-fetch fault, memory access fault	☑	0	0
Undefined instruction or illegal state	☑	0	0
System service call via SWI instruction	☑	0	0
Debug monitor	☑	0	0
Pendable request for system service	☑	0	0

NVIC priority group setting for 4 bit preemption and 0 bit for sub-priority(RTOS critical)

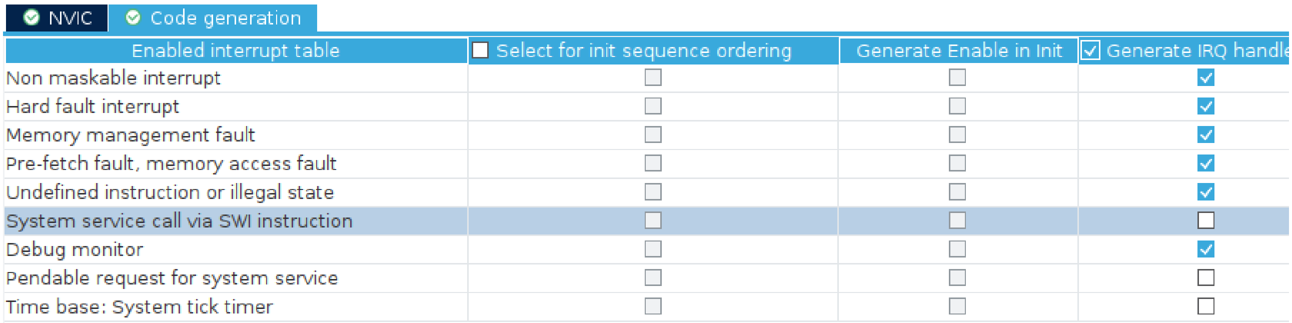


Priority Group 4 bits for pre-emption priority 0 bits for subpriority □ Sort by Premption Priority and Sub Priority □ S

Search Search (Ctrl+F) Show available interrupts ☑ F

NVIC Interrupt Table	Enabled	Preemption Pri
Non maskable interrupt	☑	0

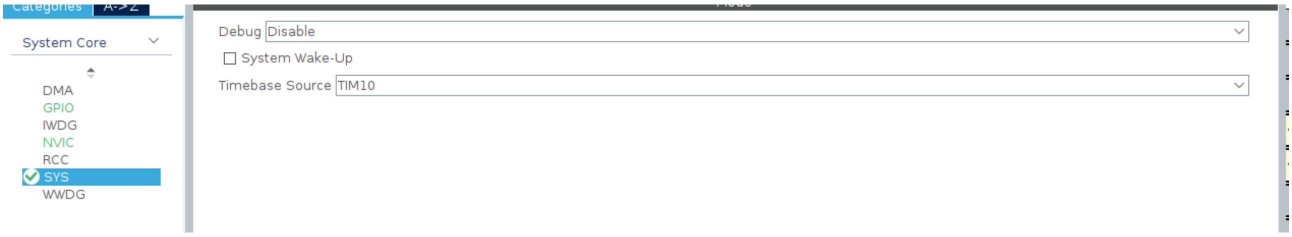
Un-check of following handlers will be needed as highloghted. Since, same interupt handler are defined by the FreeRTOS kernel and cause duplication conflict while building.



NVIC | Code generation

Enabled interrupt table	☐ Select for init sequence ordering	Generate Enable in Init	☑ Generate IRQ handle
Non maskable interrupt	☐	☐	☑
Hard fault interrupt	☐	☐	☑
Memory management fault	☐	☐	☑
Pre-fetch fault, memory access fault	☐	☐	☑
Undefined instruction or illegal state	☐	☐	☑
System service call via SWI instruction	☐	☐	☐
Debug monitor	☐	☐	☑
Pendable request for system service	☐	☐	☐
Time base: System tick timer	☐	☐	☐

Now, the HAL driver configuration uses sysTick for its time base source.FreeRTOS also uses the same SysTick for the time base source. To resolve this confllict, we can move the time base source for HAL library as one of the timer.

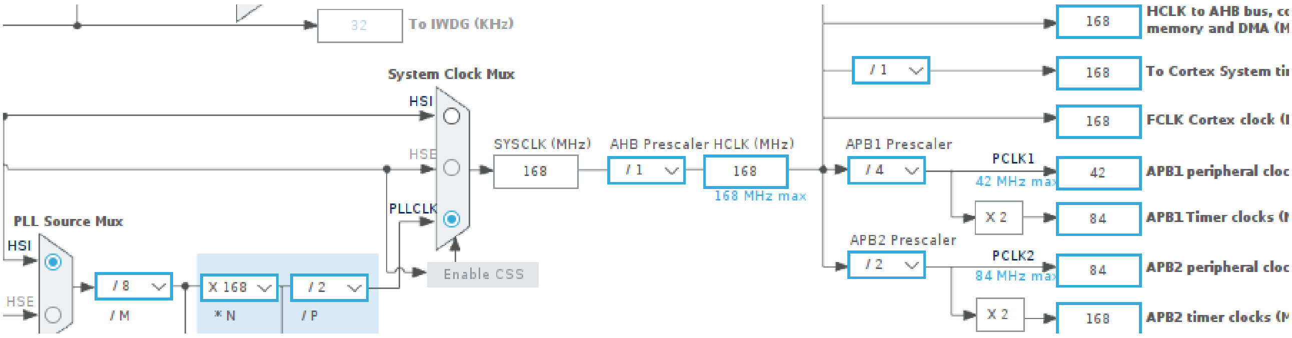


Debug Disable

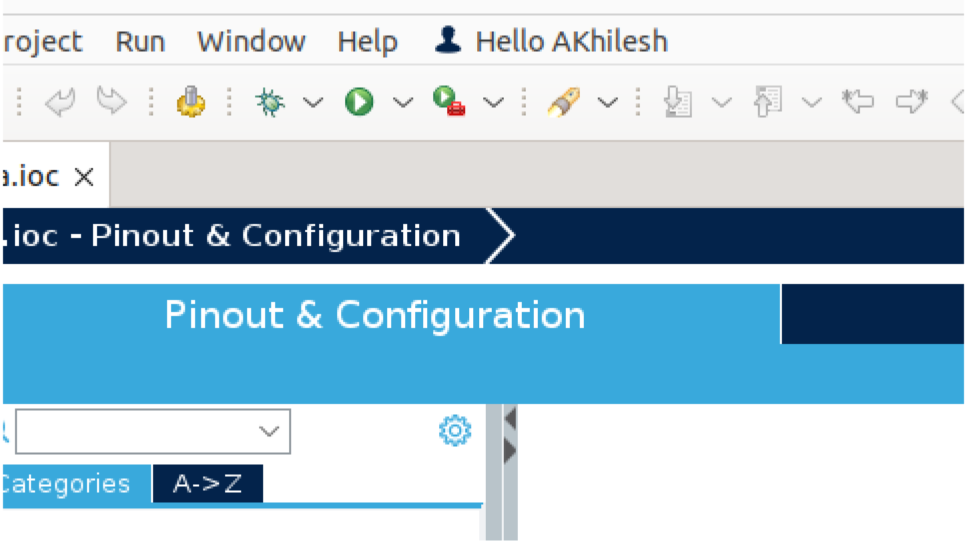
☐ System Wake-Up

Timebase Source TIM10

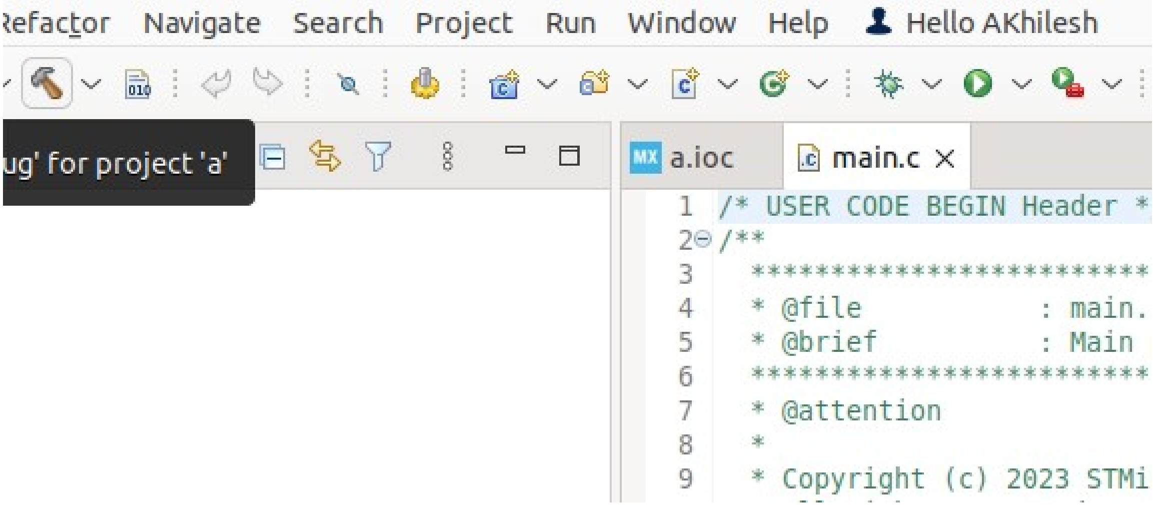
Set the HCLK(MHz) at highest value, in my case it is 168.It might be different in different uc.



Once done, we can generate the code for our configuration, same will open the c//c++ persepective.Click on th gear symbol....

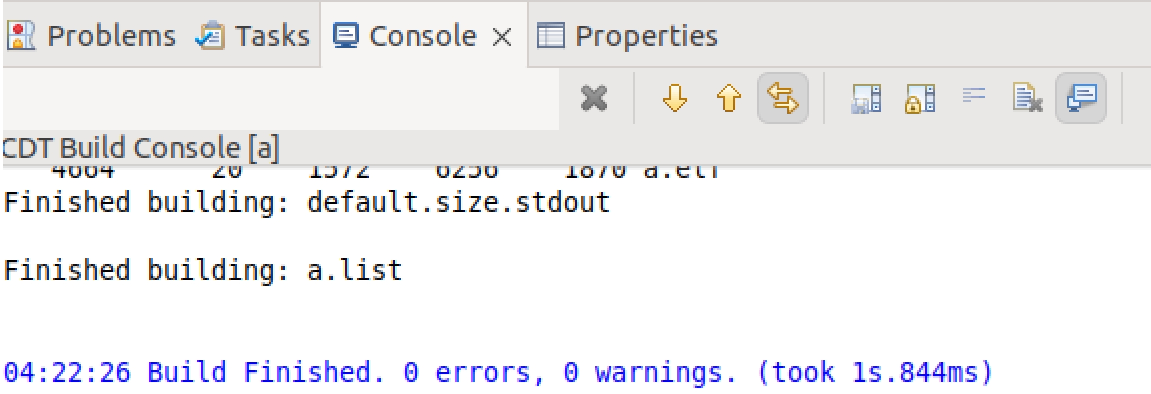


Build th code by clicking on tha hammer



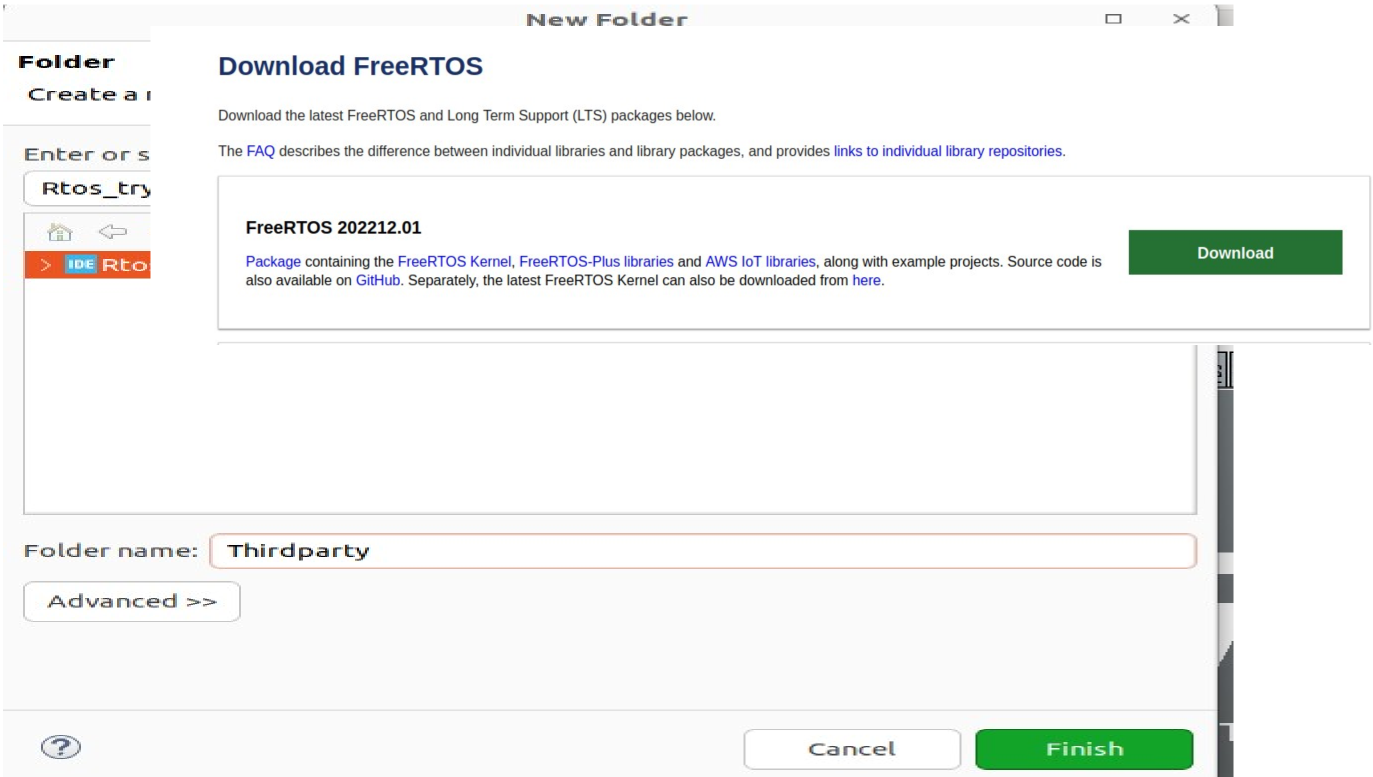
```
/* USER CODE BEGIN Header *
/**
  ************************
  * @file          : main.
  * @brief         : Main
  ************************
  * @attention
  *
  * Copyright (c) 2023 STMi
```

Succesfull build with 0 errors and 0 warnings!



```
   4064      20    1372    6256    1870 a.elf
Finished building: default.size.stdout

Finished building: a.list


04:22:26 Build Finished. 0 errors, 0 warnings. (took 1s.844ms)
```

Now we need RTOS kernel source files to project
So, first create folder “Thirdparty” uner project
Once done go to:- https://freertos.org/
and download the latest software and extract it



Folder name: Thirdparty

Download FreeRTOS

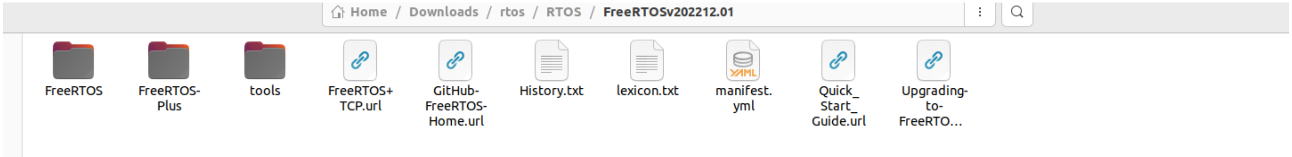
Download the latest FreeRTOS and Long Term Support (LTS) packages below.

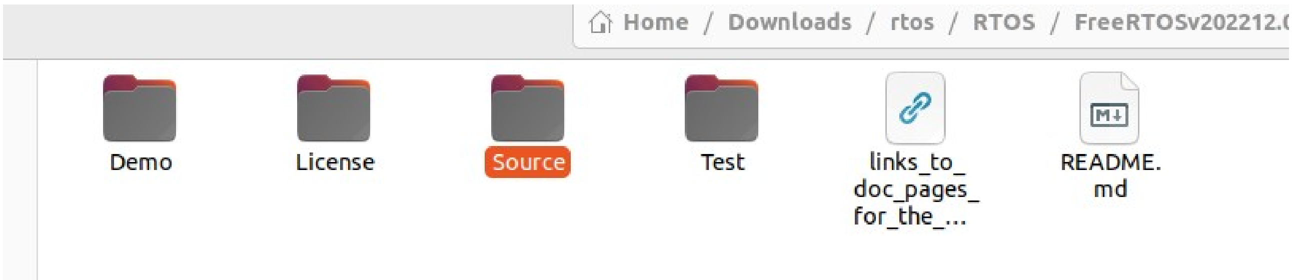
The FAQ describes the difference between individual libraries and library packages, and provides links to individual library repositories.

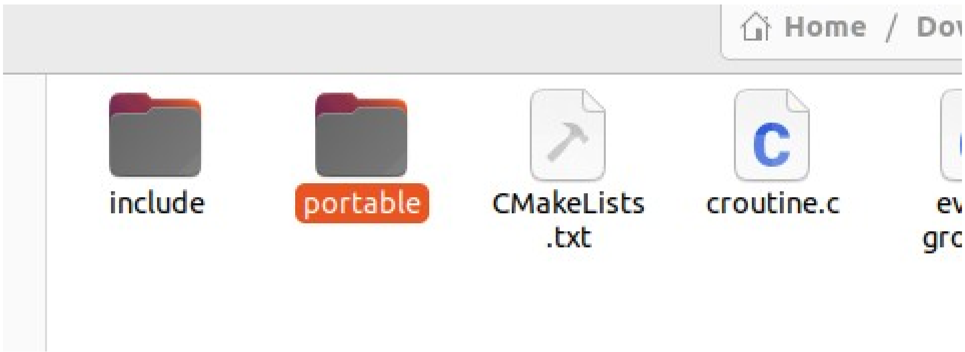
FreeRTOS 202212.01

Package containing the FreeRTOS Kernel, FreeRTOS-Plus libraries and AWS IoT libraries, along with example projects. Source code is also available on GitHub. Separately, the latest FreeRTOS Kernel can also be downloaded from here.

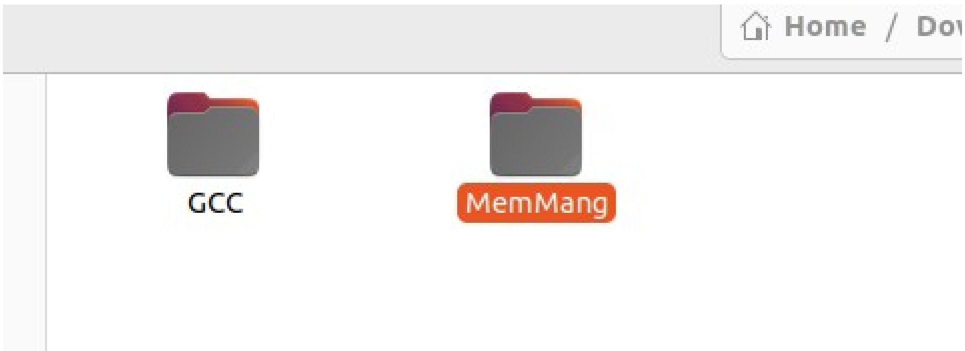
After extracting open it now it will look...



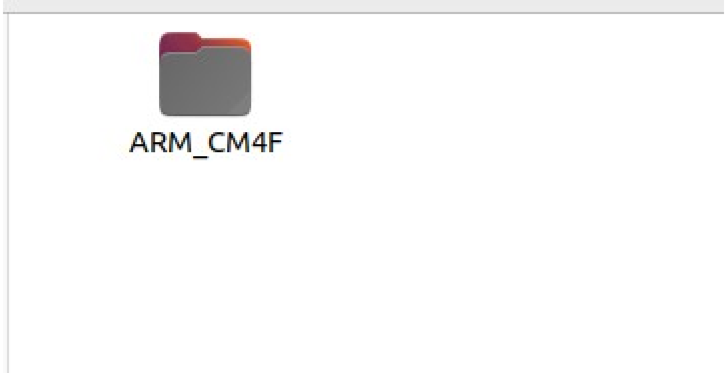

Home / Downloads / rtos / RTOS / FreeRTOSv202212.0
Demo
License
Source
Test
links_to_doc_pages_for_the_...
README.md



delete all the files rather than these inside the portabe

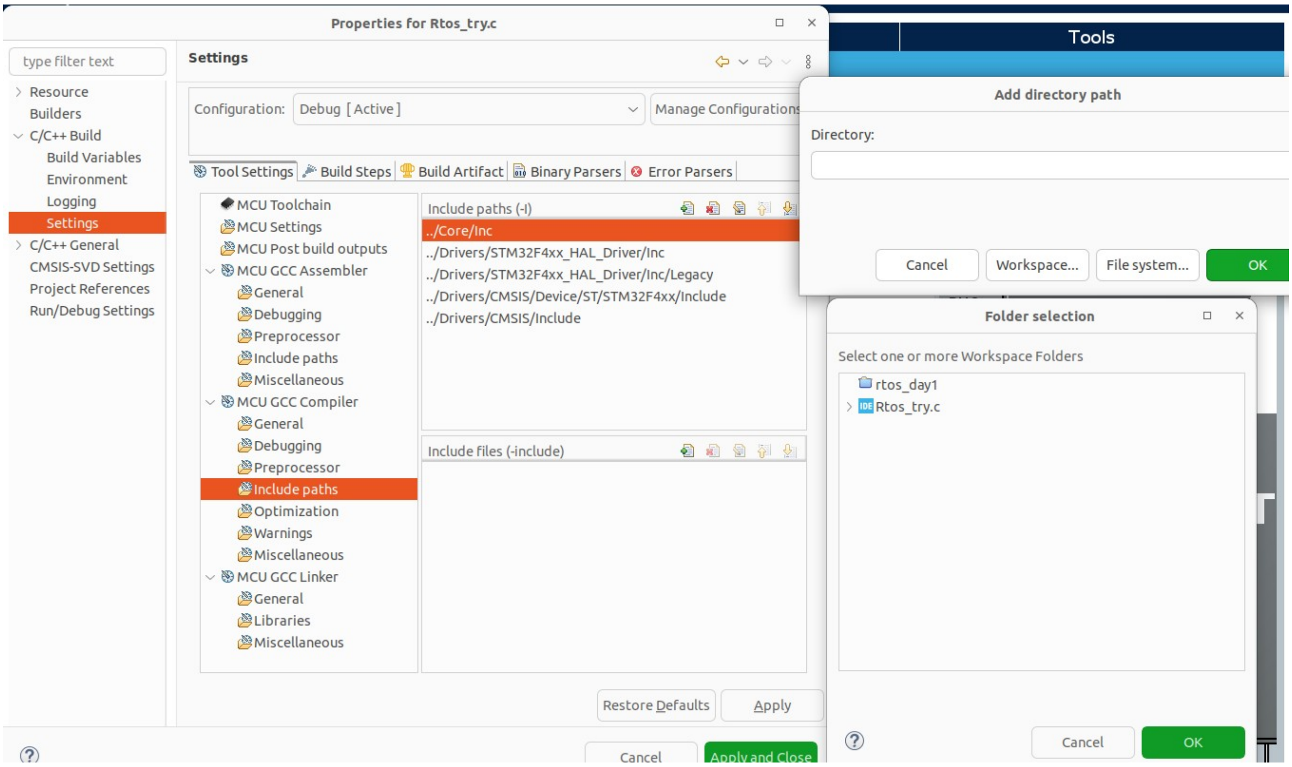


then, step into the GCC file and delete all the files other than our architecture specific in our case we are using Arm cortex-m4F



once done copy whole directory of free directory of freertos and paste on the thirdparty in our project.

Once done Right click on the project folder and select properties the go according to arrow instructions.....

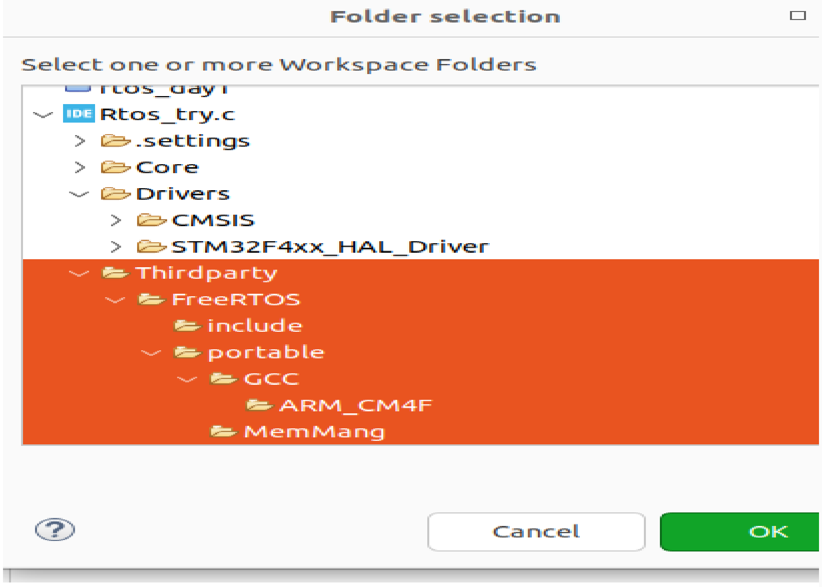


Configuration: Debug [Active]   Manage Configurations

Include paths (-I)

```
../Core/Inc
../Drivers/STM32F4xx_HAL_Driver/Inc
../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy
../Drivers/CMSIS/Device/ST/STM32F4xx/Include
../Drivers/CMSIS/Include
```

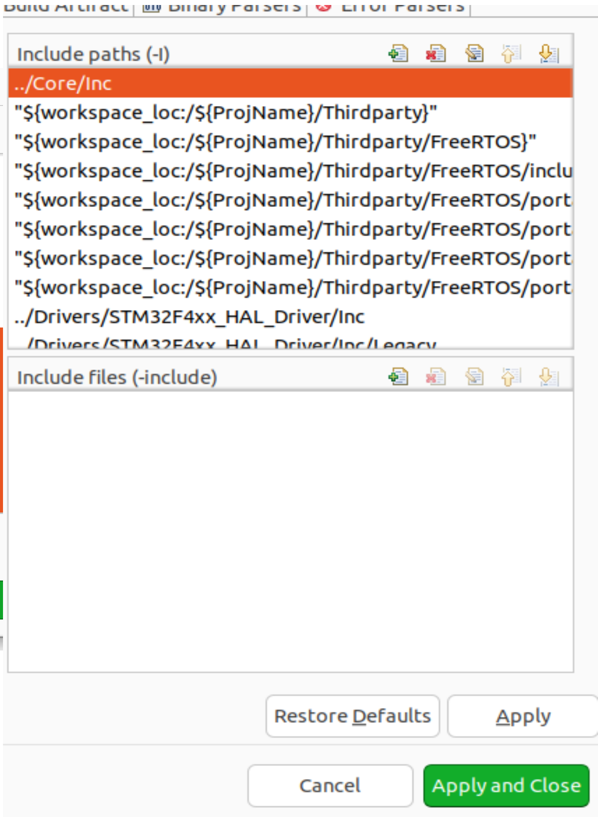
We have to add the paths of the kernel source files....
Append all the folder in th Thirdparty and select evreone like in next page....



Folder selection

Select one or more Workspace Folders

- rtos_day1
- Rtos_try.c
  - .settings
  - Core
  - Drivers
    - CMSIS
    - STM32F4xx_HAL_Driver
  - Thirdparty
    - FreeRTOS
      - include
      - portable
        - GCC
          - ARM_CM4F
        - MemMang



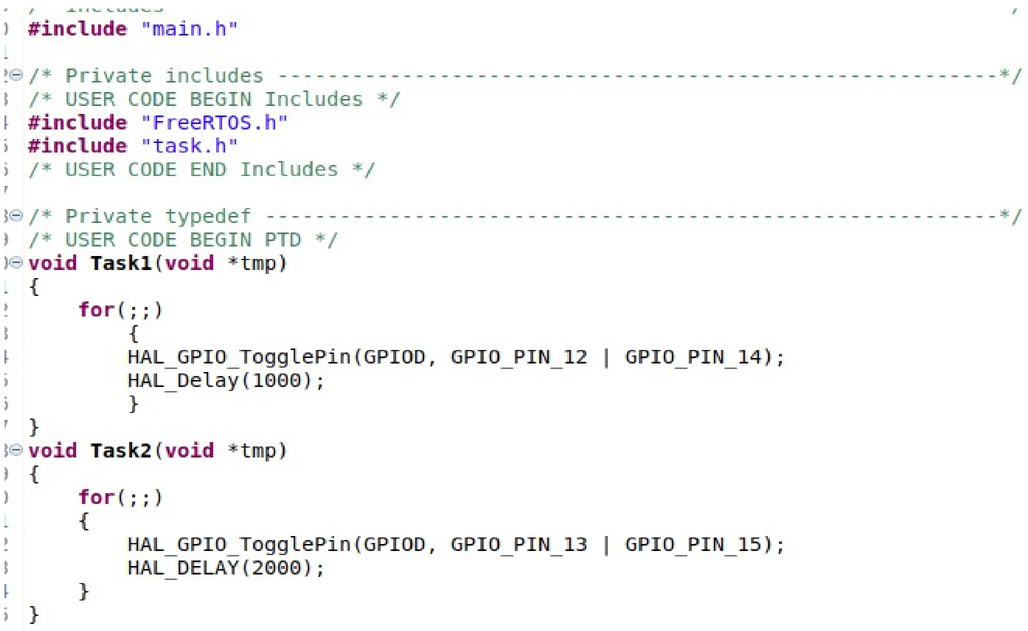
Include paths (-I)

```
../Core/Inc
"${workspace_loc:/${ProjName}/Thirdparty}"
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS}"
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS/inclu
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS/port
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS/port
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS/port
"${workspace_loc:/${ProjName}/Thirdparty/FreeRTOS/port
../Drivers/STM32F4xx_HAL_Driver/Inc
../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy
```

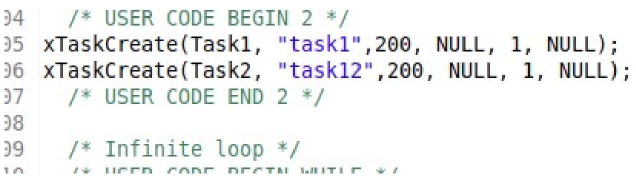
Include files (-include)

Apply and close......

To test is it working or not we need to make some task using FreeRTOS and run it!

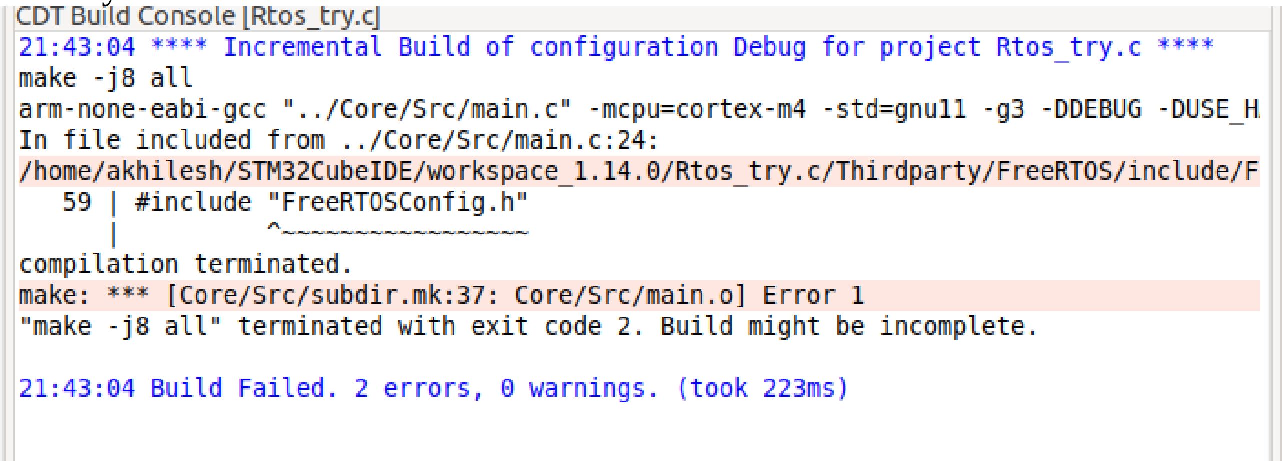


```
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
#include "FreeRTOS.h"
#include "task.h"
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
void Task1(void *tmp)
{
    for(;;)
        {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12 | GPIO_PIN_14);
        HAL_Delay(1000);
        }
}
void Task2(void *tmp)
{
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13 | GPIO_PIN_15);
        HAL_DELAY(2000);
    }
}
```



```
  /* USER CODE BEGIN 2 */
xTaskCreate(Task1, "task1",200, NULL, 1, NULL);
xTaskCreate(Task2, "task12",200, NULL, 1, NULL);
  /* USER CODE END 2 */

  /* Infinite loop */
```

Now try to buil it......

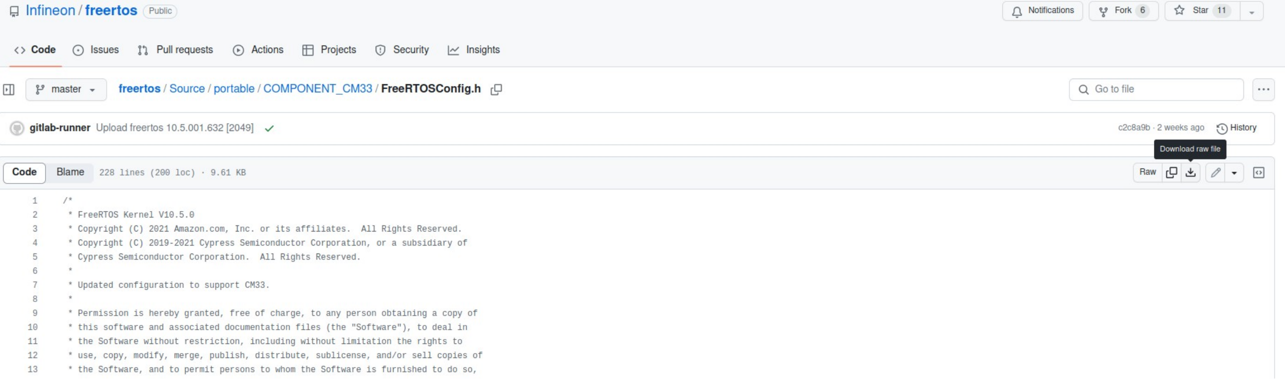


CDT Build Console [Rtos_try.c]

```
21:43:04 **** Incremental Build of configuration Debug for project Rtos_try.c ****
make -j8 all
arm-none-eabi-gcc "../Core/Src/main.c" -mcpu=cortex-m4 -std=gnu11 -g3 -DDEBUG -DUSE_H
In file included from ../Core/Src/main.c:24:
/home/akhilesh/STM32CubeIDE/workspace_1.14.0/Rtos_try.c/Thirdparty/FreeRTOS/include/F
   59 | #include "FreeRTOSConfig.h"
      |          ^~~~~~~~~~~~~~~~~~
compilation terminated.
make: *** [Core/Src/subdir.mk:37: Core/Src/main.o] Error 1
"make -j8 all" terminated with exit code 2. Build might be incomplete.

21:43:04 Build Failed. 2 errors, 0 warnings. (took 223ms)
```

so we got some error because we are not included configuration file of FreeRTOS
To download it :-
https://github.com/Infineon/freertos/blob/master/Source/portable/COMPONENT_CM33/FreeRTOSConfig.h

It will opened like this now download raw file.....

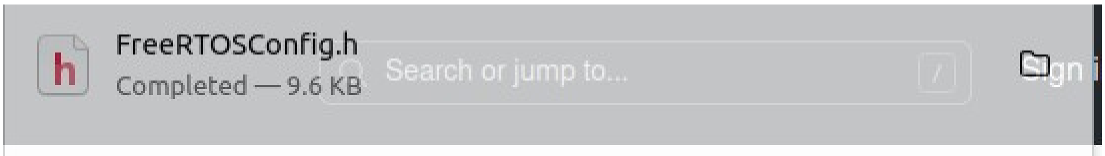


freertos / Source / portable / COMPONENT_CM33 / FreeRTOSConfig.h

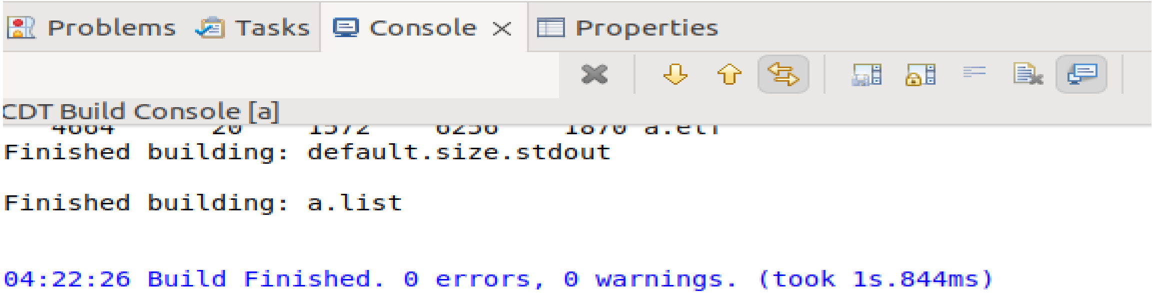
gitlab-runner Upload freertos 10.5.001.632 [2049] ✓ c2c8a9b · 2 weeks ago

228 lines (200 loc) · 9.61 KB

```
/*
 * FreeRTOS Kernel V10.5.0
 * Copyright (C) 2021 Amazon.com, Inc. or its affiliates.  All Rights Reserved.
 * Copyright (C) 2019-2021 Cypress Semiconductor Corporation, or a subsidiary of
 * Cypress Semiconductor Corporation.  All Rights Reserved.
 *
 * Updated configuration to support CM33.
 *
 * Permission is hereby granted, free of charge, to any person obtaining a copy of
 * this software and associated documentation files (the "Software"), to deal in
 * the Software without restriction, including without limitation the rights to
 * use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
 * the Software, and to permit persons to whom the Software is furnished to do so,
```



Now copy this file and paste it into the FreeRTOS inside the Thirdpart in your project.
Now save the program and build again.... Build succesfully!!!!



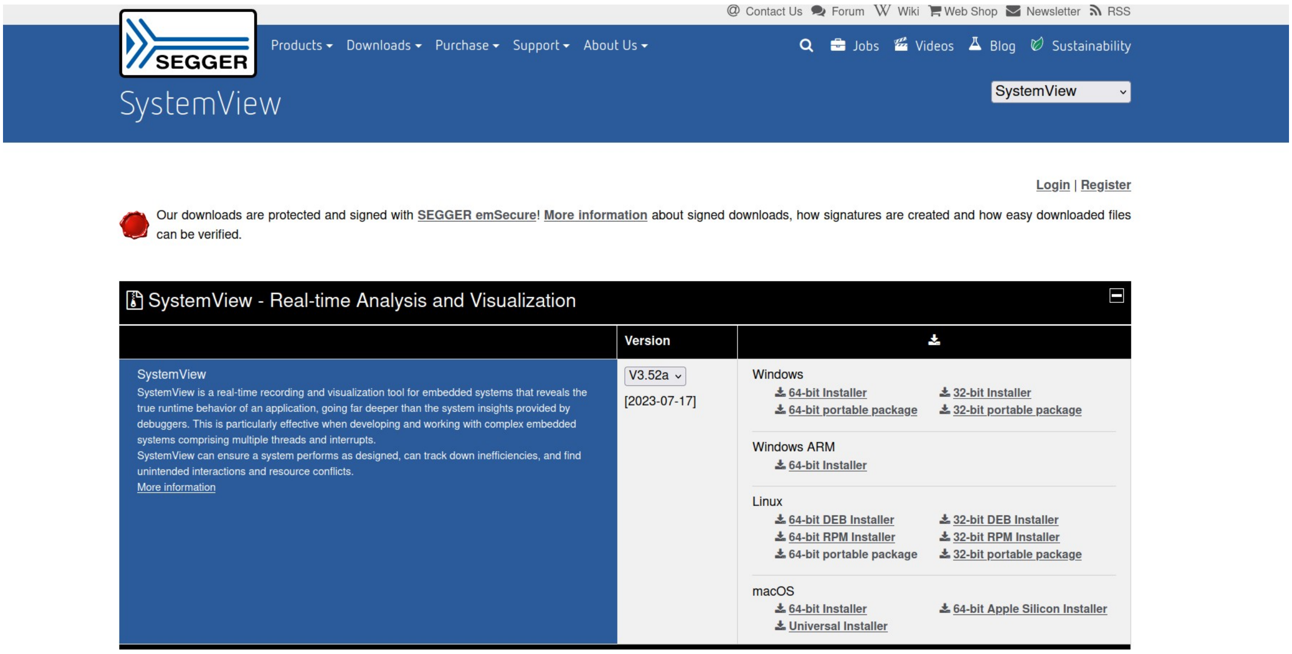
CDT Build Console [a]

```
   4604      20    1572    6256    1870 a.elf
Finished building: default.size.stdout

Finished building: a.list


04:22:26 Build Finished. 0 errors, 0 warnings. (took 1s.844ms)
```

Steps for creating for SEGGER Systemview in STM32Cube IDE

First we need to downlaod the source file of SEGGER Systemview..
Download link:- https://www.segger.com/downloads/systemview/

Now screen will be look like this..

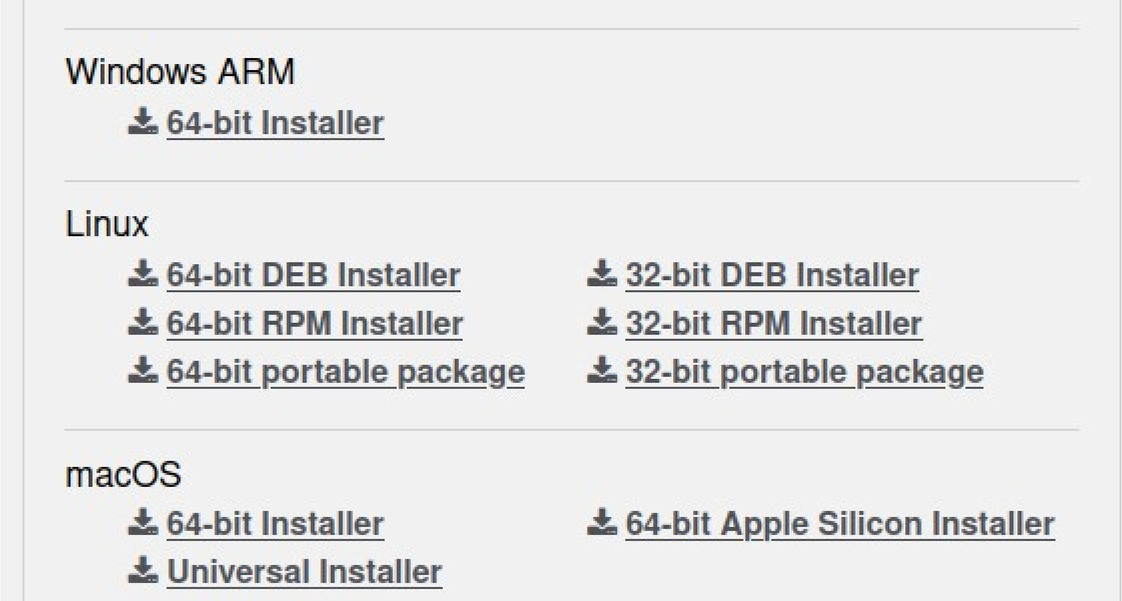


Our downloads are protected and signed with SEGGER emSecure! More information about signed downloads, how signatures are created and how easy downloaded files can be verified.

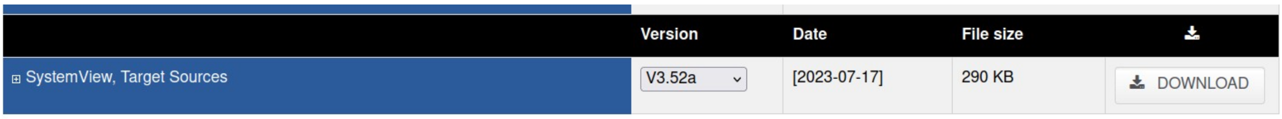
SystemView - Real-time Analysis and Visualization

	Version	
SystemView SystemView is a real-time recording and visualization tool for embedded systems that reveals the true runtime behavior of an application, going far deeper than the system insights provided by debuggers. This is particularly effective when developing and working with complex embedded systems comprising multiple threads and interrupts. SystemView can ensure a system performs as designed, can track down inefficiencies, and find unintended interactions and resource conflicts. More information	V3.52a [2023-07-17]	Windows 64-bit Installer, 32-bit Installer 64-bit portable package, 32-bit portable package Windows ARM 64-bit Installer Linux 64-bit DEB Installer, 32-bit DEB Installer 64-bit RPM Installer, 32-bit RPM Installer 64-bit portable package, 32-bit portable package macOS 64-bit Installer, 64-bit Apple Silicon Installer Universal Installer

In my case i am using linux and 64-bit architecture.
Therefore, i downloaded the linux--> 64-bit portable package

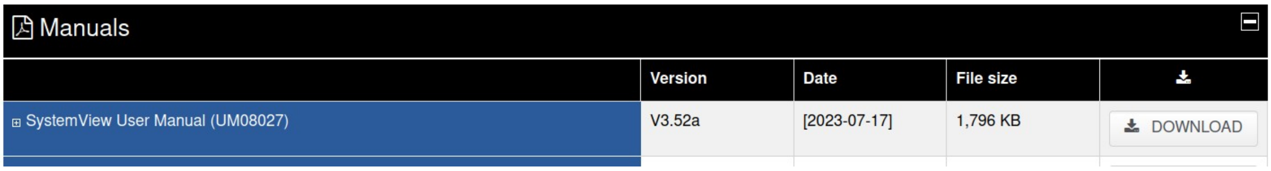


Now we need the target source...



	Version	Date	File size	
⊞ SystemView, Target Sources	V3.52a	[2023-07-17]	290 KB	DOWNLOAD

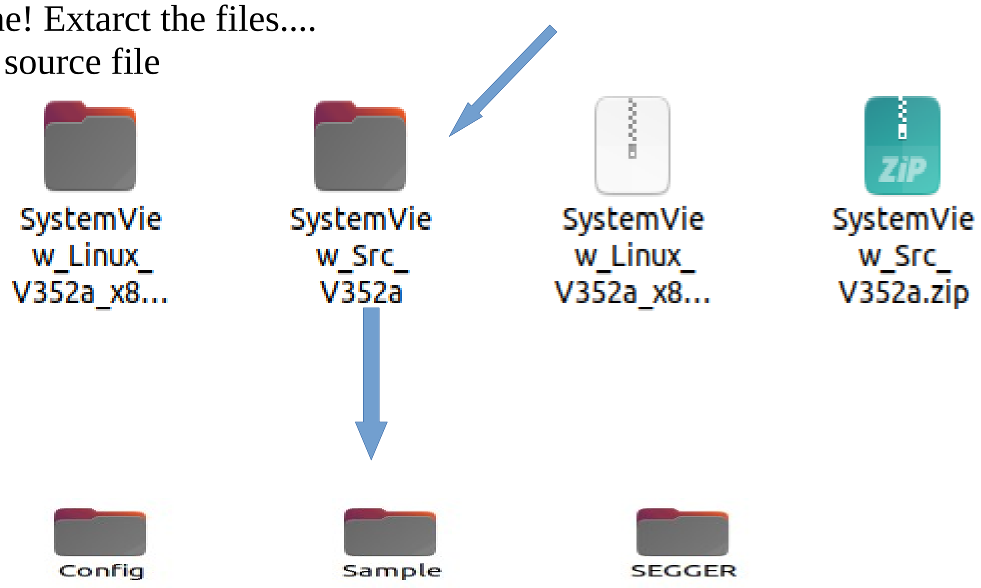
Optional, you can downlaod User manual for SystemView....



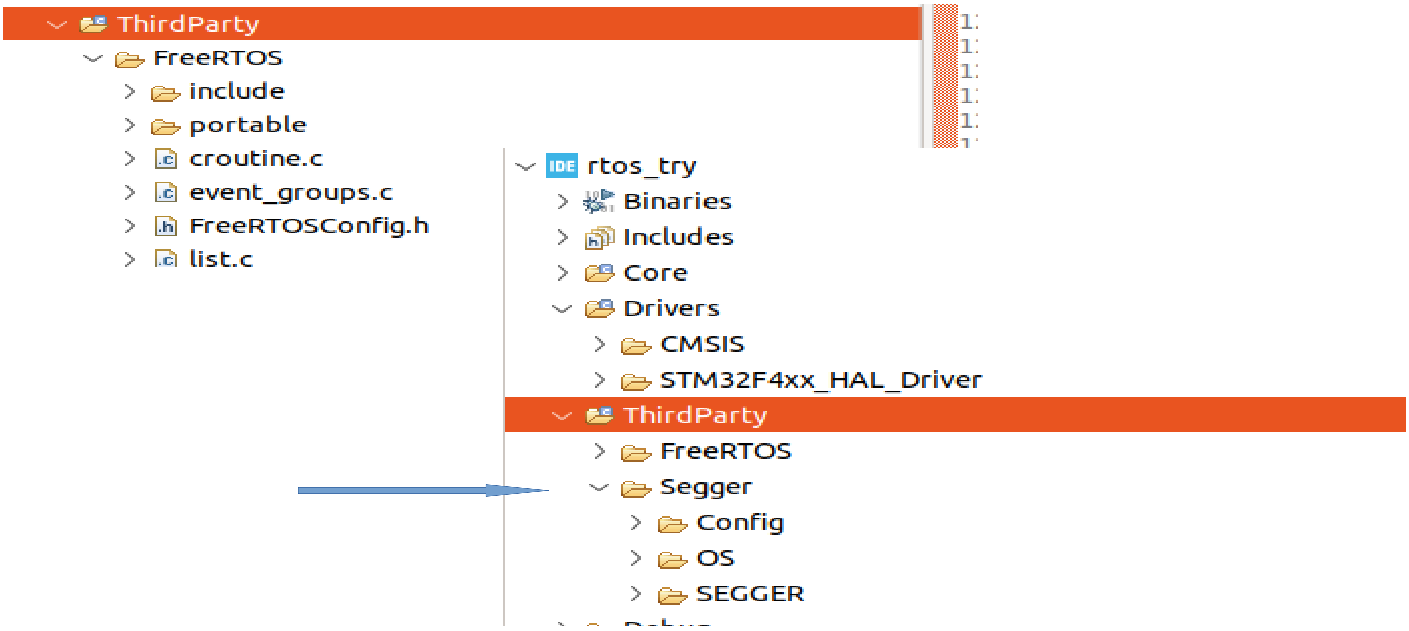
Manuals

	Version	Date	File size	
⊞ SystemView User Manual (UM08027)	V3.52a	[2023-07-17]	1,796 KB	DOWNLOAD

Once done! Extarct the files....
Step into source file

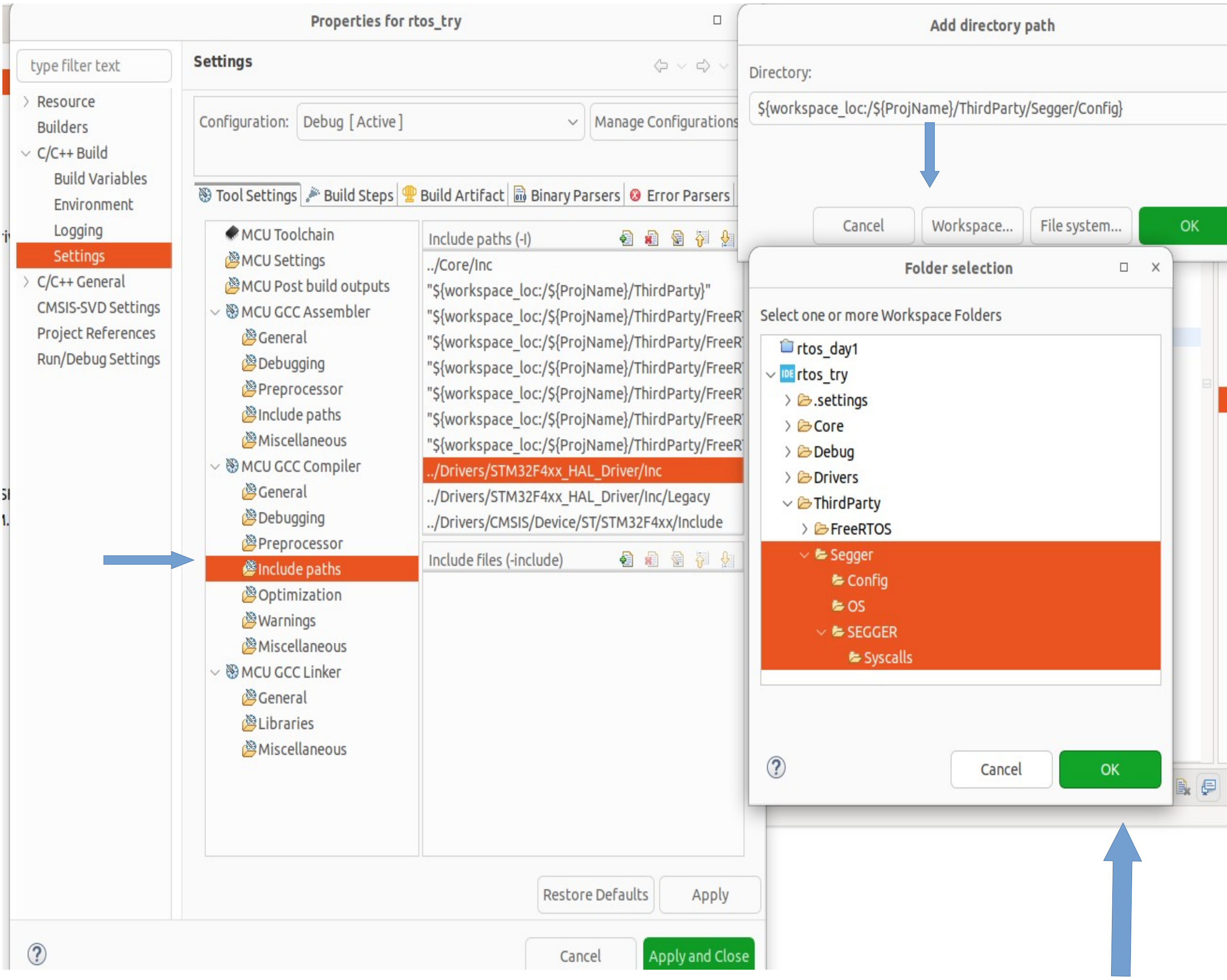


Create a folder “segger” into Thirdparty in your project parallel to FreeRTOS file . Now copy all the files(Config,Sample and Segger) and paste into thirdparty folder in your project...



Once done we need to include paths on the top of the project as we done previous for FreeRTOS in our project.

Select properties of your prjoect



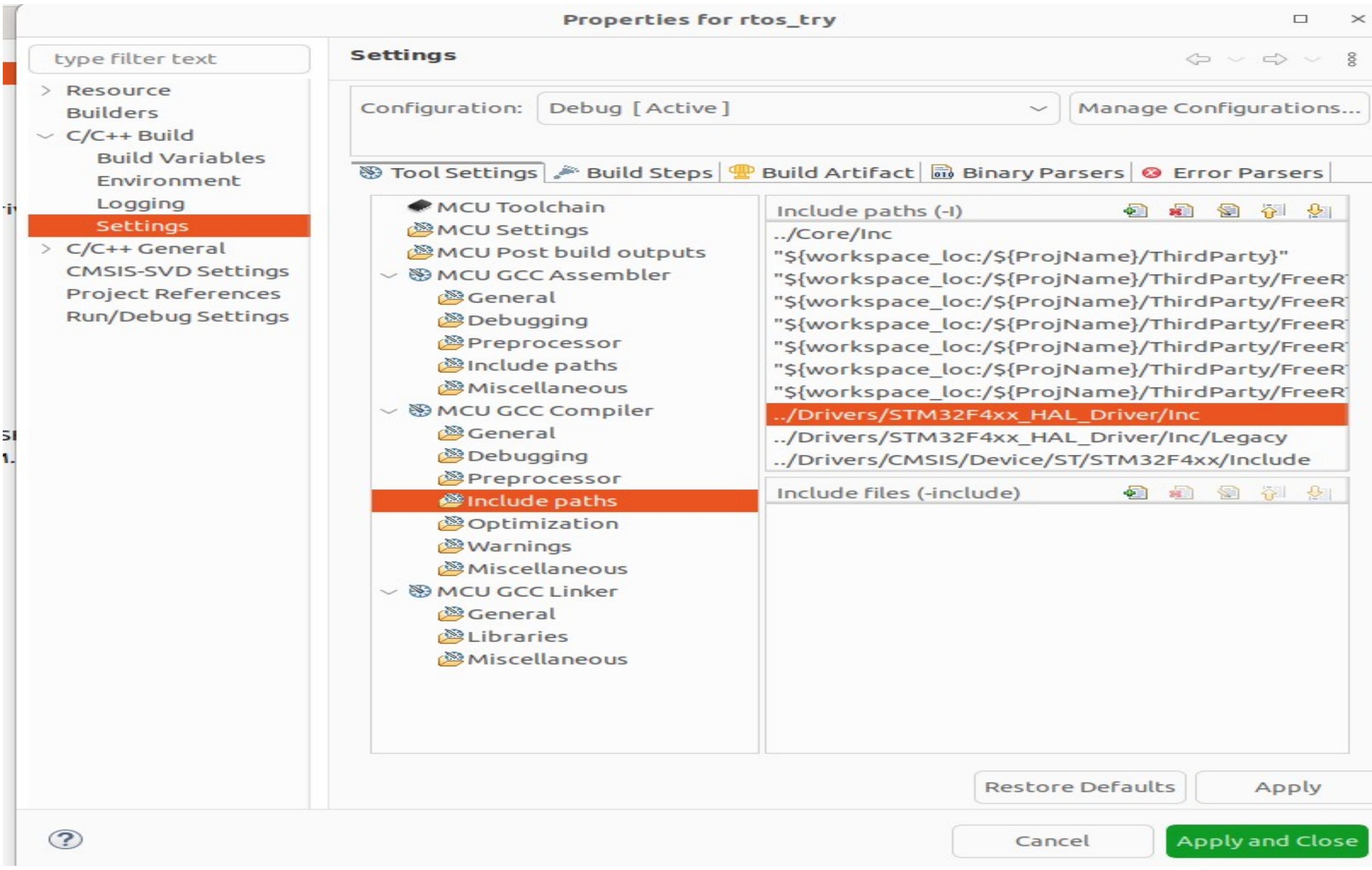
Configuration: Debug [Active]

Include paths (-I)

```
../Core/Inc
"${workspace_loc:/${ProjName}/ThirdParty}"
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
../Drivers/STM32F4xx_HAL_Driver/Inc
../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy
../Drivers/CMSIS/Device/ST/STM32F4xx/Include
```

Directory: ${workspace_loc:/${ProjName}/ThirdParty/Segger/Config}

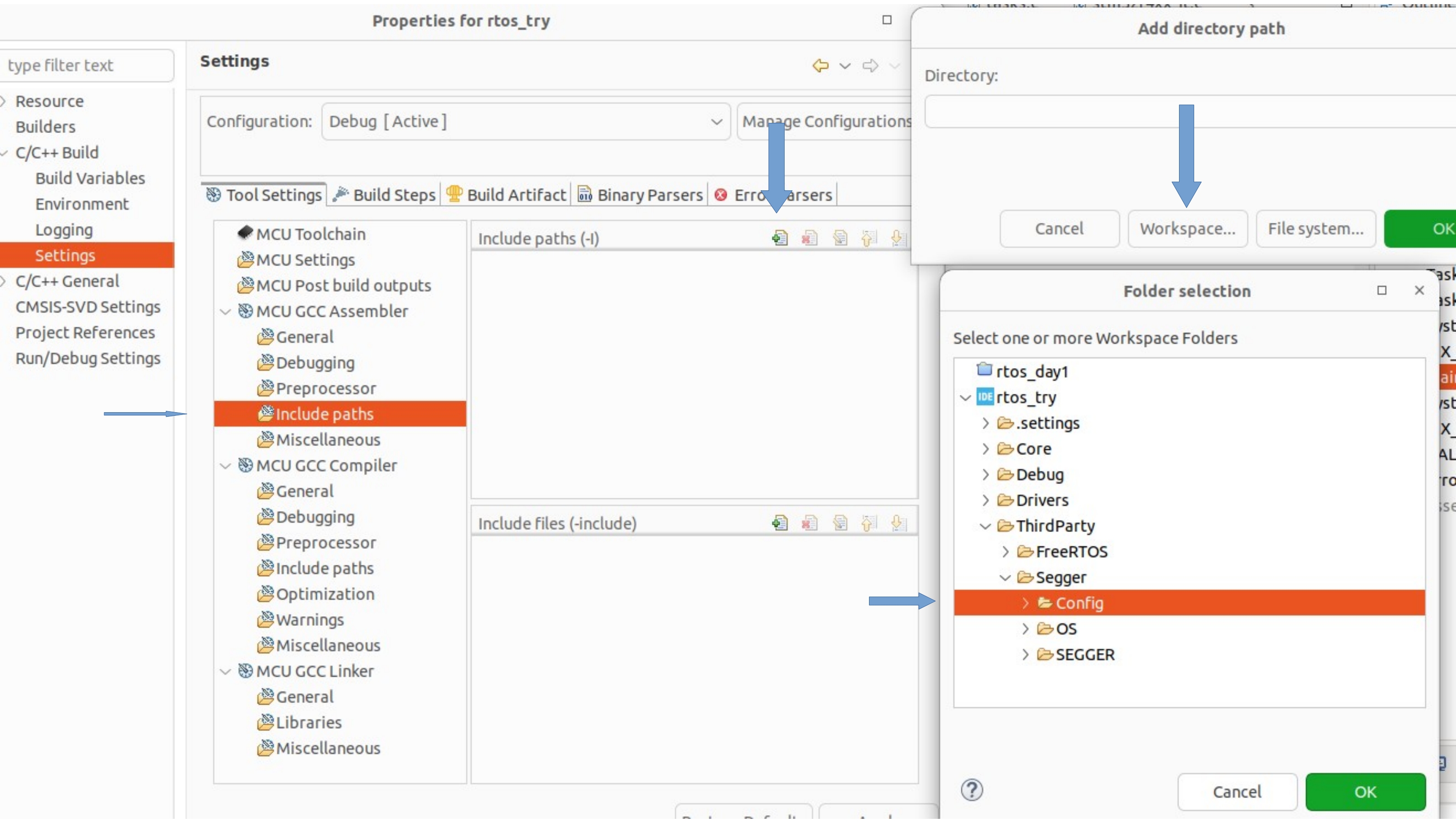
Once done it will look like this....P.T.O



Include paths (-I)

```
../Core/Inc
"${workspace_loc:/${ProjName}/ThirdParty}"
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
"${workspace_loc:/${ProjName}/ThirdParty/FreeR
../Drivers/STM32F4xx_HAL_Driver/Inc
../Drivers/STM32F4xx_HAL_Driver/Inc/Legacy
../Drivers/CMSIS/Device/ST/STM32F4xx/Include
```

Now we have to add path for config file of segger in assembler...

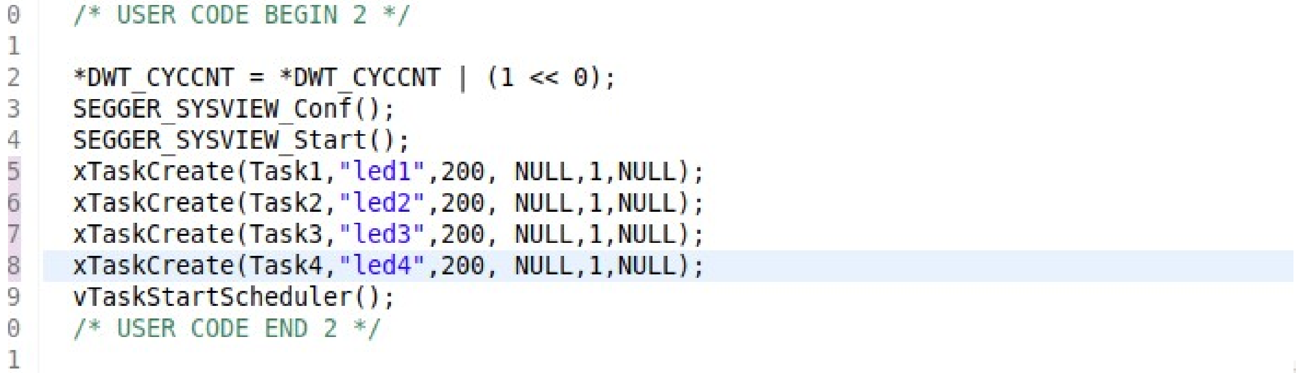


->Need to add one patch one more patch that is setup configuration of SystemView with FreeRTOS.Download link:-

https://github.com/adafruit/Adafruit_nRF52_Arduino/blob/master/cores/nRF5/sysview/Config/SEGGER_SYSVIEW_Config_FreeRTOS.c

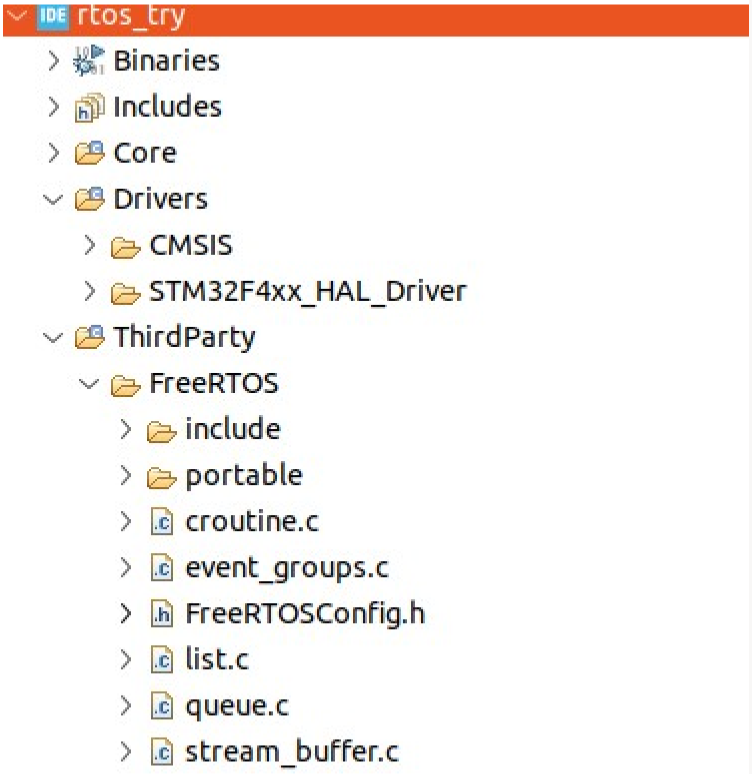
-> and copy it into the config file of segger of Thirdparty in project.

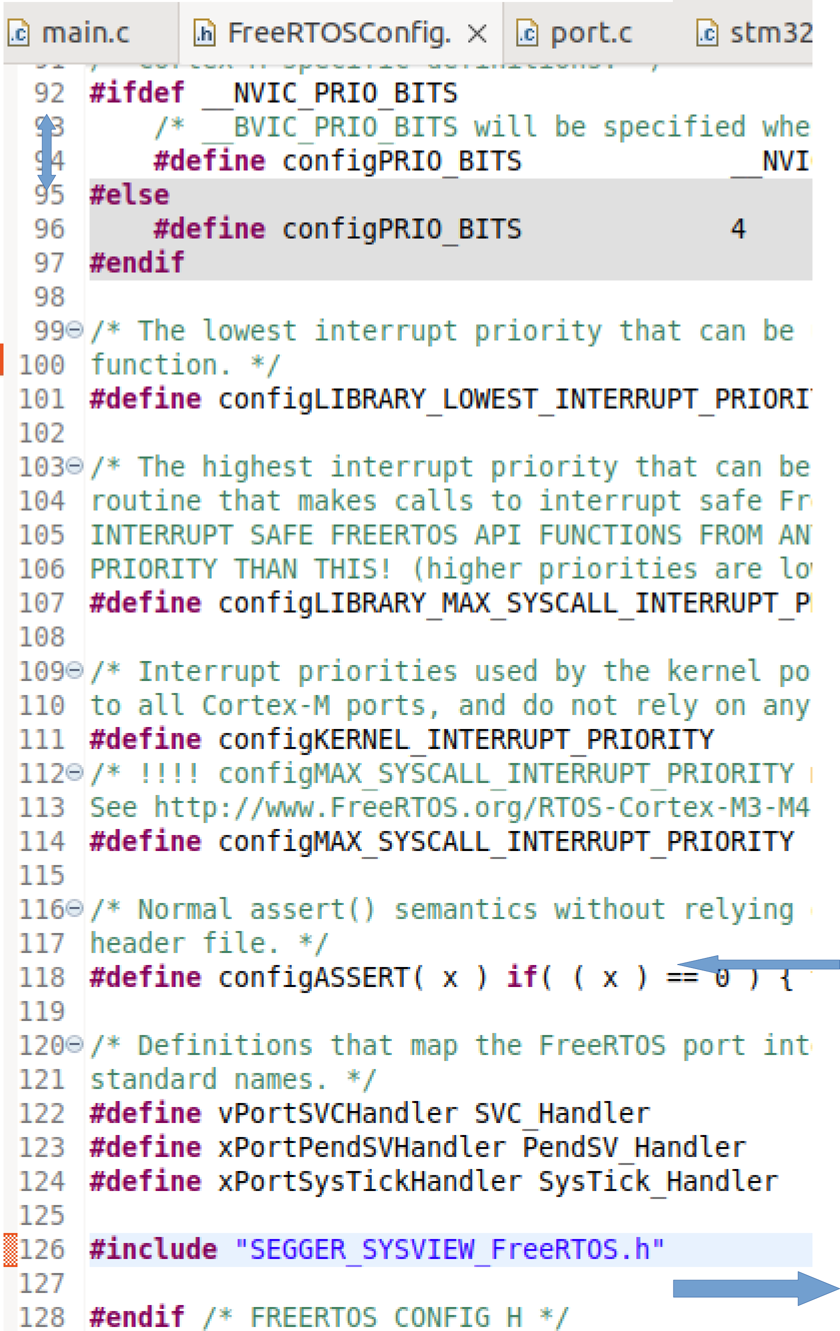
Once done need to add below threeee lines.



```
/* USER CODE BEGIN 2 */

*DWT_CYCCNT = *DWT_CYCCNT | (1 << 0);
SEGGER_SYSVIEW_Conf();
SEGGER_SYSVIEW_Start();
xTaskCreate(Task1,"led1",200, NULL,1,NULL);
xTaskCreate(Task2,"led2",200, NULL,1,NULL);
xTaskCreate(Task3,"led3",200, NULL,1,NULL);
xTaskCreate(Task4,"led4",200, NULL,1,NULL);
vTaskStartScheduler();
/* USER CODE END 2 */
```

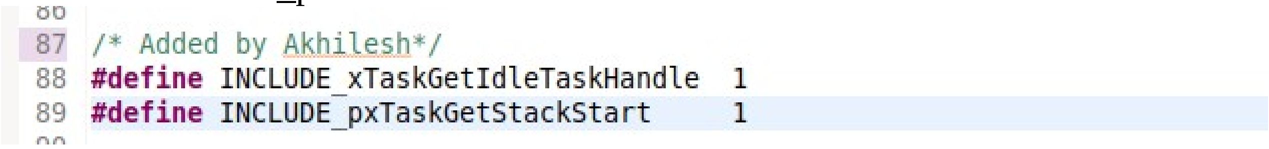
Include header file of SEGGER_SYSVIEW_FreeRTOS.h in FreeRTOSConfig.h file





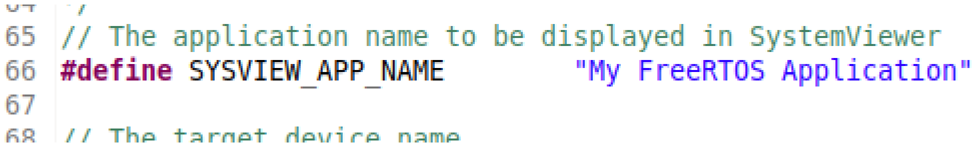
```
#ifdef __NVIC_PRIO_BITS
    /* __BVIC_PRIO_BITS will be specified whe
    #define configPRIO_BITS              __NVI
#else
    #define configPRIO_BITS              4
#endif

/* The lowest interrupt priority that can be
function. */
#define configLIBRARY_LOWEST_INTERRUPT_PRIORI

/* The highest interrupt priority that can be
routine that makes calls to interrupt safe Fr
INTERRUPT SAFE FREERTOS API FUNCTIONS FROM AN
PRIORITY THAN THIS! (higher priorities are lo
#define configLIBRARY_MAX_SYSCALL_INTERRUPT_P

/* Interrupt priorities used by the kernel po
to all Cortex-M ports, and do not rely on any
#define configKERNEL_INTERRUPT_PRIORITY
/* !!!! configMAX_SYSCALL_INTERRUPT_PRIORITY
See http://www.FreeRTOS.org/RTOS-Cortex-M3-M4
#define configMAX_SYSCALL_INTERRUPT_PRIORITY

/* Normal assert() semantics without relying
header file. */
#define configASSERT( x ) if( ( x ) == 0 ) {

/* Definitions that map the FreeRTOS port int
standard names. */
#define vPortSVCHandler SVC_Handler
#define xPortPendSVHandler PendSV_Handler
#define xPortSysTickHandler SysTick_Handler

#include "SEGGER_SYSVIEW_FreeRTOS.h"

#endif /* FREERTOS_CONFIG_H */
```

In "FreeRTOSConfig.h" include the following macro switches
#define INCLUDE_xTaskGetIdleTaskHandle 1
#define INCLUDE_pxTaskGetStackStart



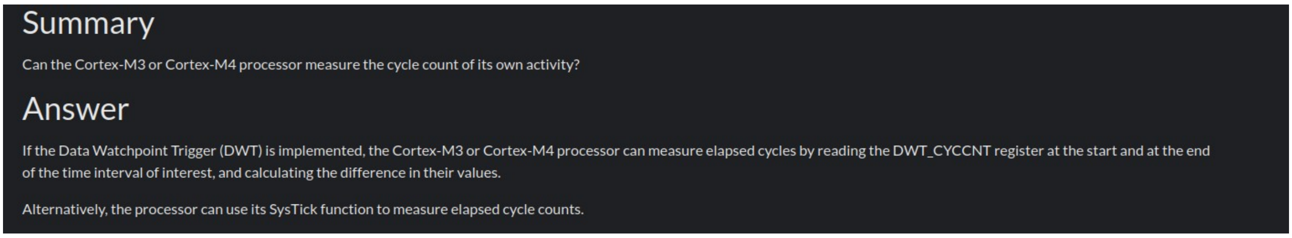
```
/* Added by Akhilesh*/
#define INCLUDE_xTaskGetIdleTaskHandle  1
#define INCLUDE_pxTaskGetStackStart     1
```

Application specific information in the "SEGGER_SYSVIEW_Config_FreeRTOS.c"



```
// The application name to be displayed in SystemViewer
#define SYSVIEW_APP_NAME        "My FreeRTOS Application"

// The target device name
```

Once done, we need to enable the time stamp information to be dumped by our RTOS application the same is needed for monitoring of the events at specific time frames. Which then shows what event was configured at what stage.
In order for our STM32 board to monitor the time stamp information, we need to enable the same at the hardware level using the "Cycle Counter" of ARM M4F
For more information refer to ARM website.
https://developer.arm.com/documentation/ka001406/latest



Summary

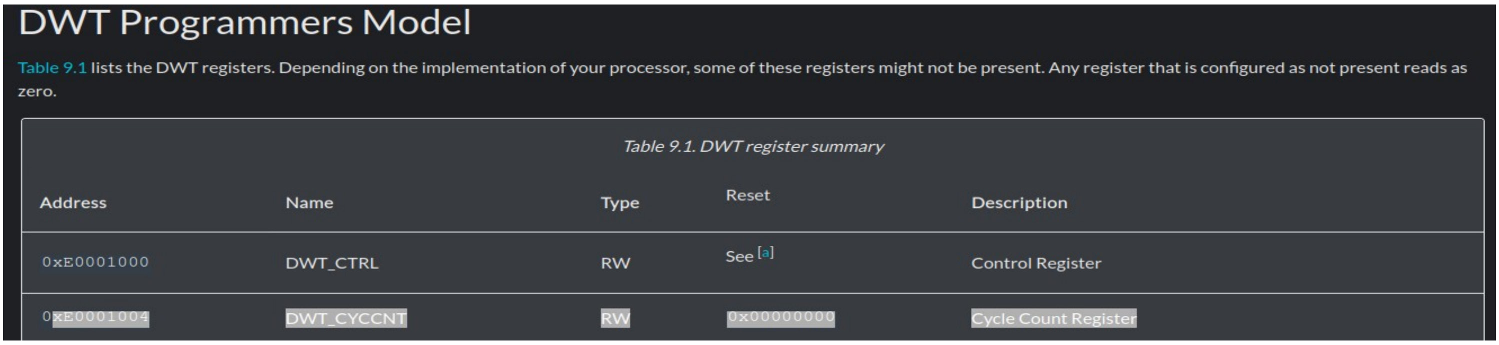
Can the Cortex-M3 or Cortex-M4 processor measure the cycle count of its own activity?

Answer

If the Data Watchpoint Trigger (DWT) is implemented, the Cortex-M3 or Cortex-M4 processor can measure elapsed cycles by reading the DWT_CYCCNT register at the start and at the end of the time interval of interest, and calculating the difference in their values.

Alternatively, the processor can use its SysTick function to measure elapsed cycle counts.

DWT_CYCCNT register of the ARM Cortex M3/4/4F processor will store the number of clock cycles since the processor was POR or started after reset.
For ARM Cortex M3/4 based controllers, the given register is available at location 0xE0001000. This register addresses are managed by the ARM and the CMSIS is configuring the same.
https://developer.arm.com/documentation/ddi0439/b/Data-Watchpoint-and-Trace-Unit/DWT-Programmers-Model



DWT Programmers Model

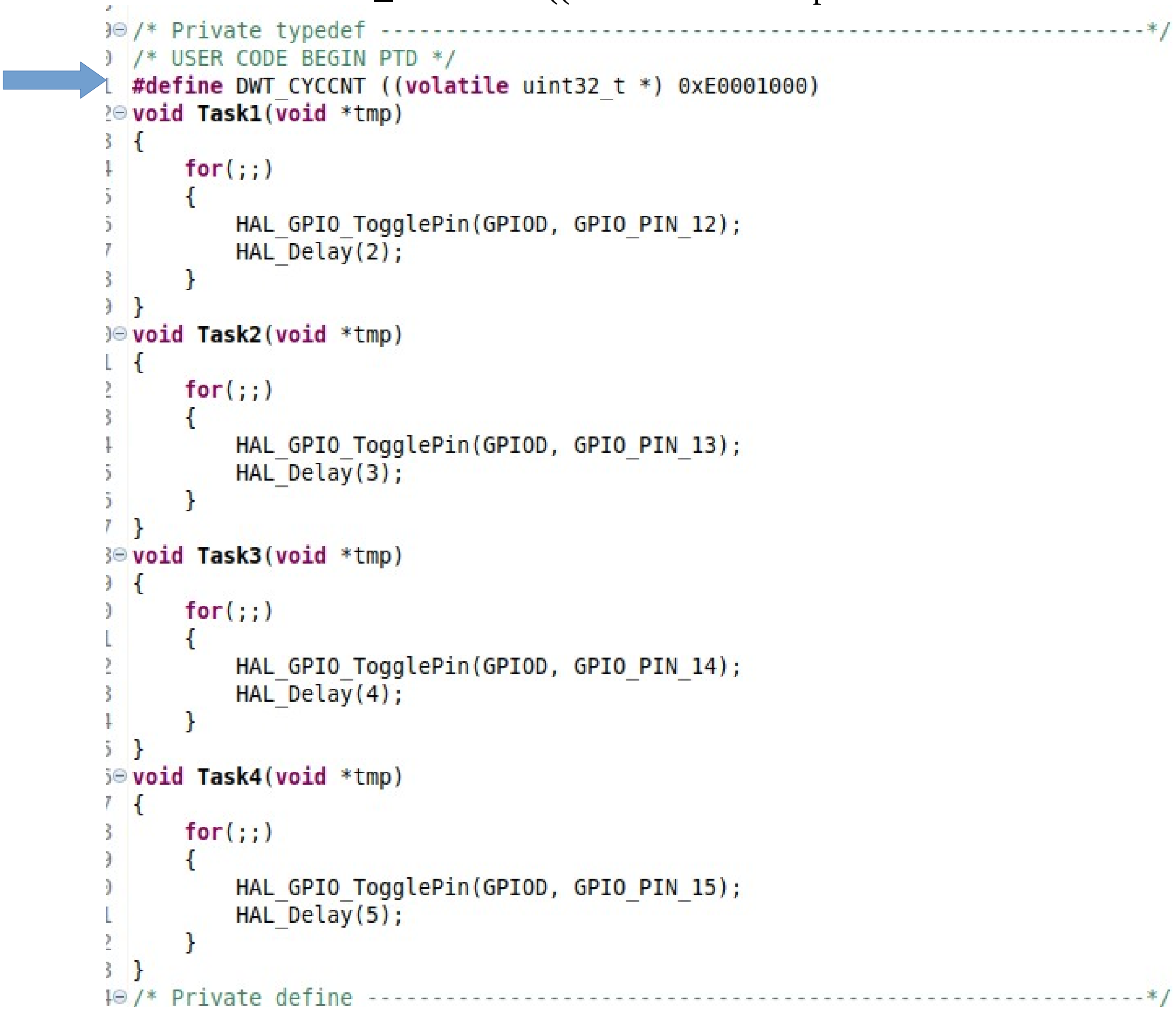
Table 9.1 lists the DWT registers. Depending on the implementation of your processor, some of these registers might not be present. Any register that is configured as not present reads as zero.

Table 9.1. DWT register summary

Address	Name	Type	Reset	Description
0xE0001000	DWT_CTRL	RW	See [a]	Control Register
0xE0001004	DWT_CYCCNT	RW	0x00000000	Cycle Count Register

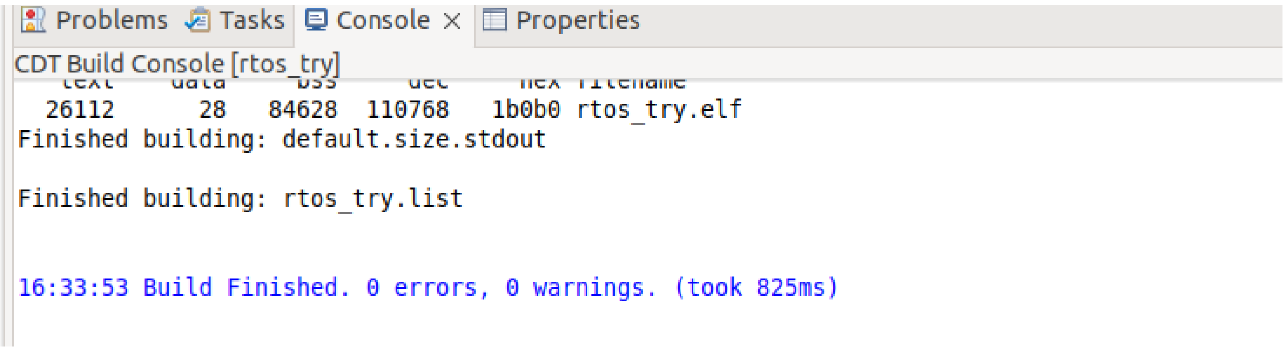
These are mainly used for the tracing functionality, when it is enabled in the system/build.

Now, we can easlity set the required biy in register "DWT_CYCCNT" by performing a simple pointer operation Control register, DWT_CTRL,we need to set the bit...
*DWT_CYCCNT |= (1 << 0);
Add #define DWT_CYCCNT ((volatile...... on top of the task



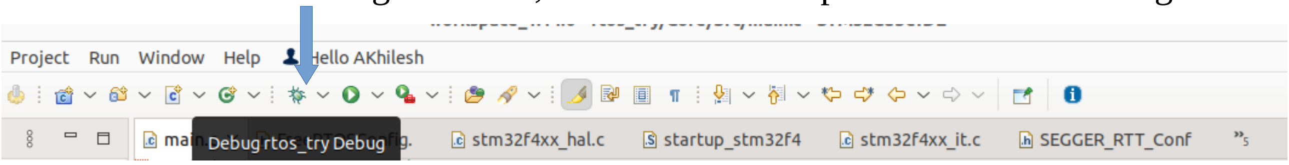
```
/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */
#define DWT_CYCCNT ((volatile uint32_t *) 0xE0001000)
void Task1(void *tmp)
{
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);
        HAL_Delay(2);
    }
}
void Task2(void *tmp)
{
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
        HAL_Delay(3);
    }
}
void Task3(void *tmp)
{
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_14);
        HAL_Delay(4);
    }
}
void Task4(void *tmp)
{
    for(;;)
    {
        HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_15);
        HAL_Delay(5);
    }
}
/* Private define ------------------------------------------------------------*/
```

Once done! Now build the program....

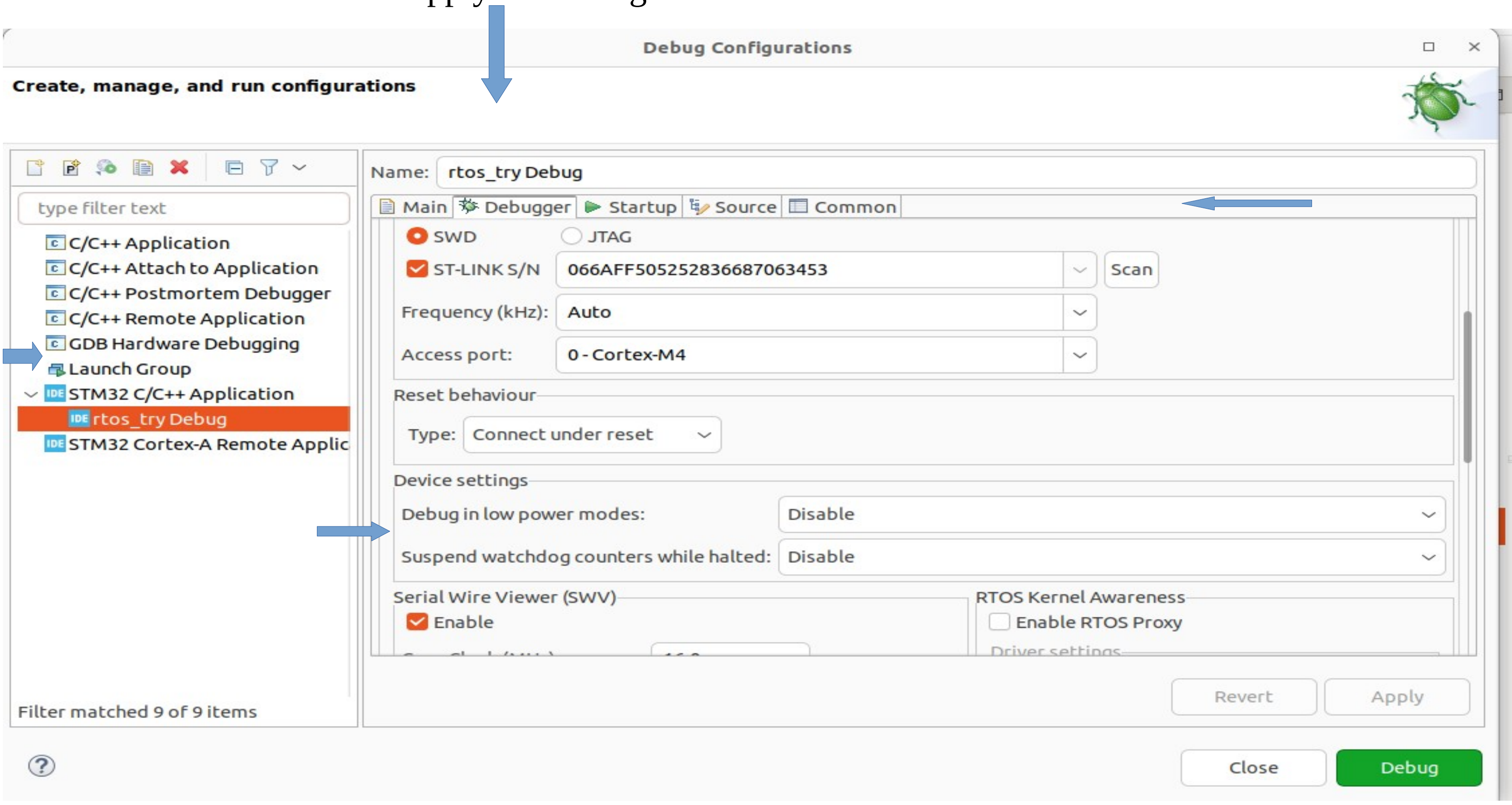


```
   text    data     bss     dec     hex filename
  26112      28   84628  110768   1b0b0 rtos_try.elf
Finished building: default.size.stdout

Finished building: rtos_try.list


16:33:53 Build Finished. 0 errors, 0 warnings. (took 825ms)
```

Successfully build!
Now we need to debug the code.,now follow the steps for serial wire debug...



Once done! Select apply and debug...



Debug Configurations

Create, manage, and run configurations

- C/C++ Application
- C/C++ Attach to Application
- C/C++ Postmortem Debugger
- C/C++ Remote Application
- GDB Hardware Debugging
- Launch Group
- STM32 C/C++ Application
  - rtos_try Debug
- STM32 Cortex-A Remote Applic

Filter matched 9 of 9 items

Name: rtos_try Debug

Main | Debugger | Startup | Source | Common

● SWD ○ JTAG

☑ ST-LINK S/N: 066AFF505252836687063453 (Scan)

Frequency (kHz): Auto

Access port: 0 - Cortex-M4

Reset behaviour

Type: Connect under reset

Device settings

Debug in low power modes: Disable

Suspend watchdog counters while halted: Disable

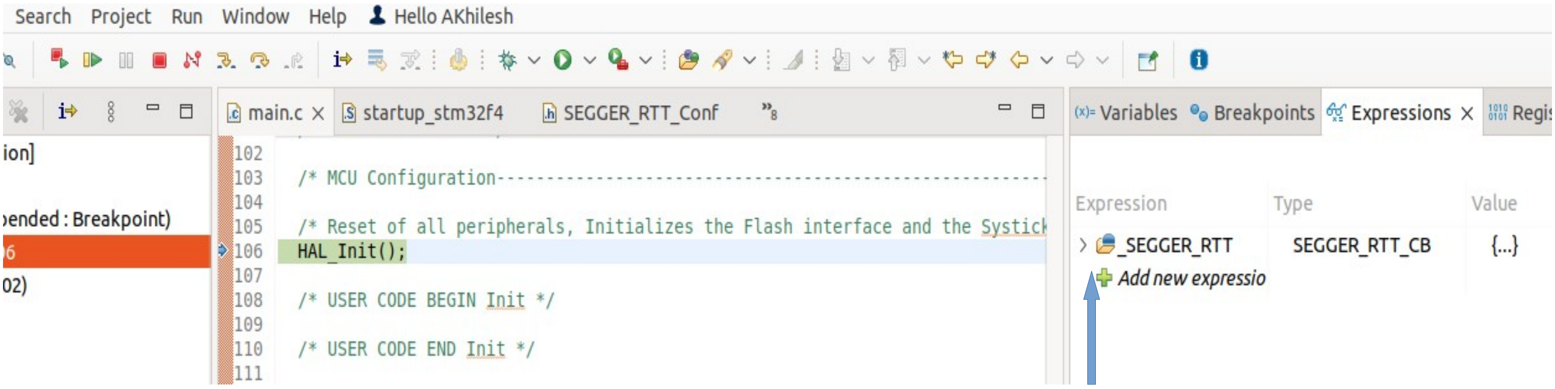
Serial Wire Viewer (SWV)

☑ Enable

RTOS Kernel Awareness

☐ Enable RTOS Proxy

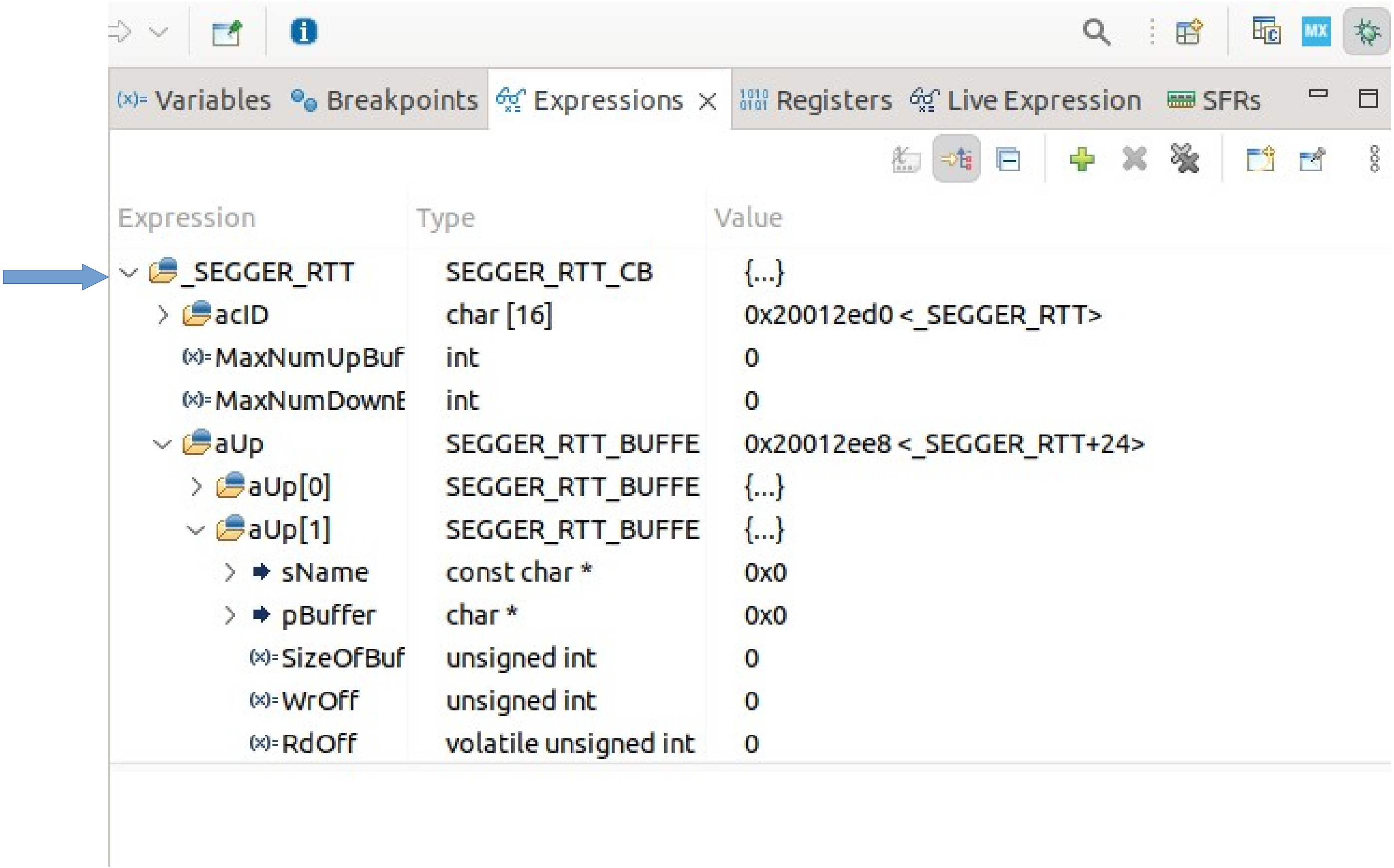
Add _SEGGER_RTT in the Expression view...



```
102
103   /* MCU Configuration--------------------------------------------------------
104
105   /* Reset of all peripherals, Initializes the Flash interface and the Systick
106   HAL_Init();
107
108   /* USER CODE BEGIN Init */
109
110   /* USER CODE END Init */
111
```

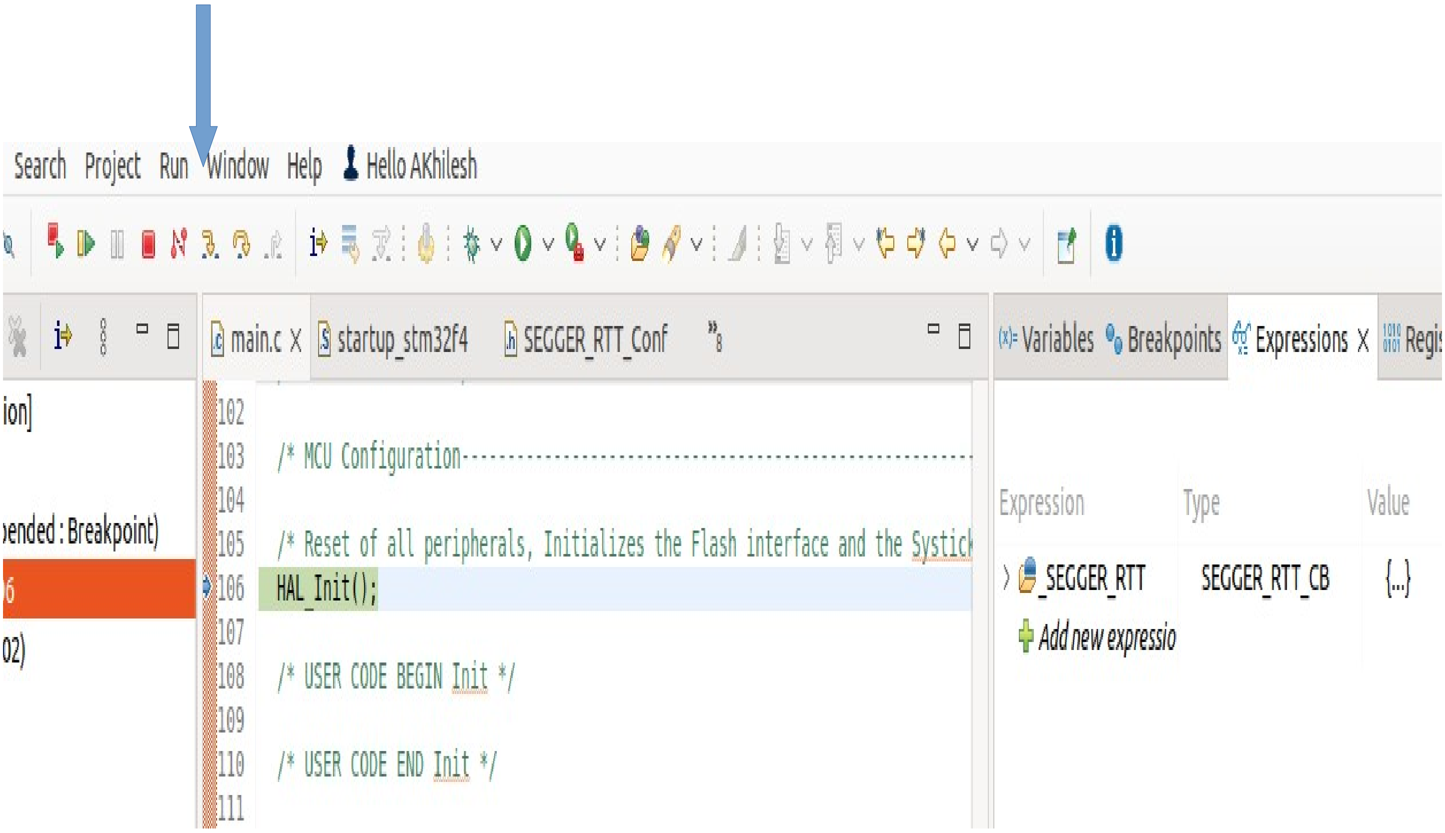
Expression	Type	Value
_SEGGER_RTT	SEGGER_RTT_CB	{...}
Add new expressio		

Append SEGGER_RTT like that



Expression	Type	Value
_SEGGER_RTT	SEGGER_RTT_CB	{...}
acID	char [16]	0x20012ed0 <_SEGGER_RTT>
MaxNumUpBuf	int	0
MaxNumDownE	int	0
aUp	SEGGER_RTT_BUFFE	0x20012ee8 <_SEGGER_RTT+24>
aUp[0]	SEGGER_RTT_BUFFE	{...}
aUp[1]	SEGGER_RTT_BUFFE	{...}
sName	const char *	0x0
pBuffer	char *	0x0
SizeOfBuf	unsigned int	0
WrOff	unsigned int	0
RdOff	volatile unsigned int	0

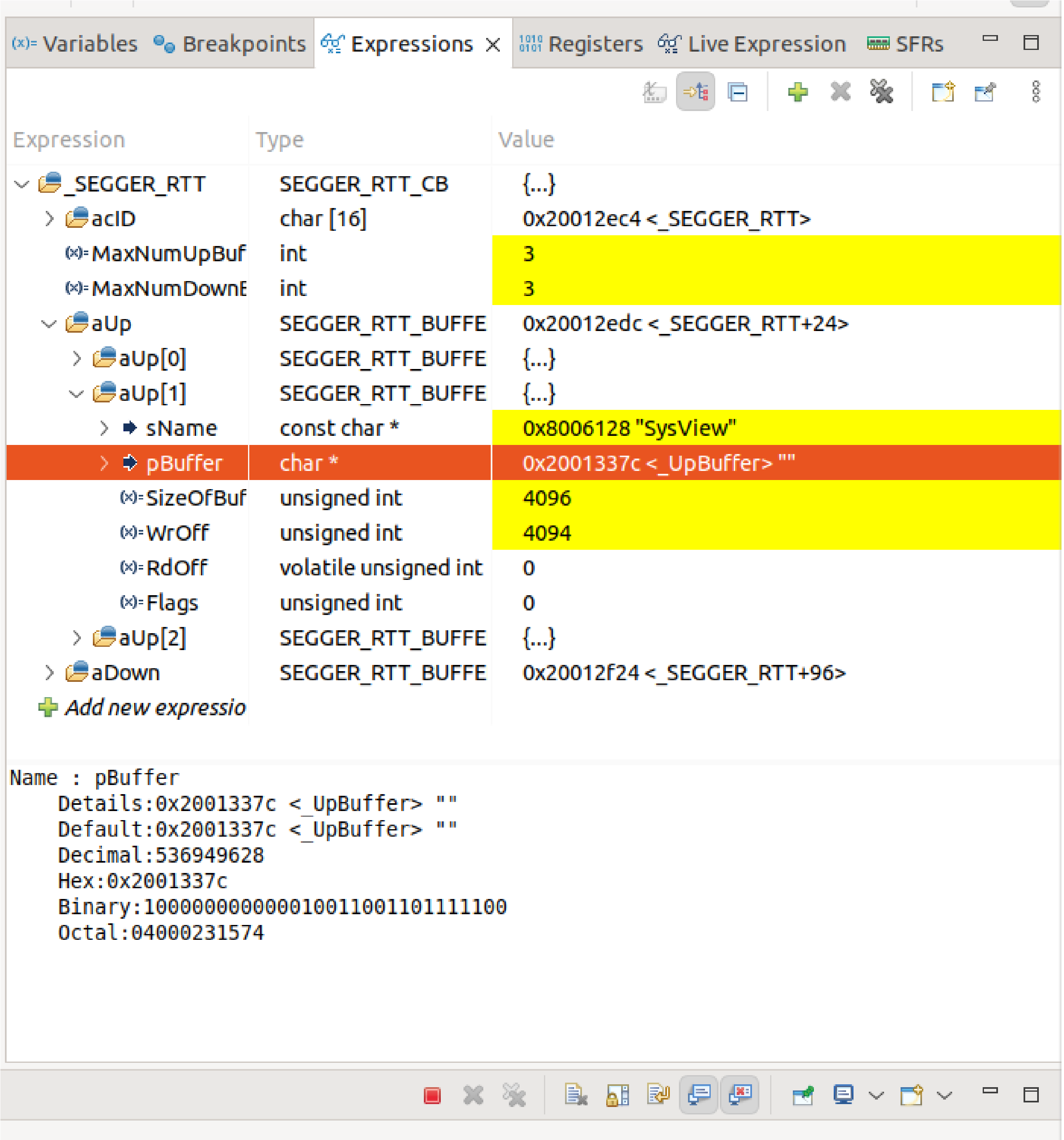
now resume the program for 3 second and then suspend it ....



```

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

  /* Reset of all peripherals, Initializes the Flash interface and the Systick

  HAL_Init();

  /* USER CODE BEGIN Init */

  /* USER CODE END Init */

```

Expression	Type	Value
_SEGGER_RTT	SEGGER_RTT_CB	{...}
Add new expressio		

now the values are changed like that....

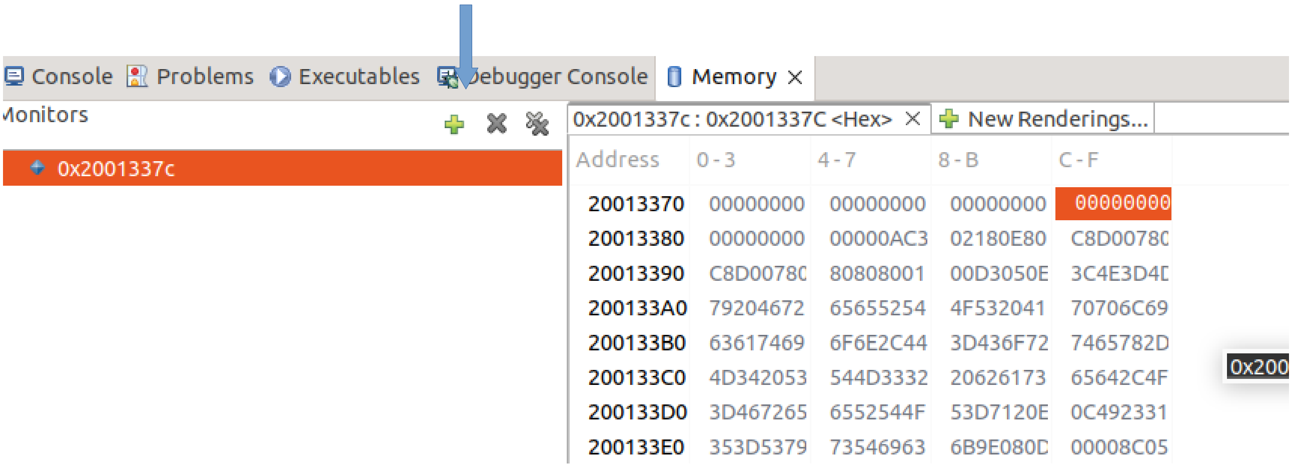


Expression	Type	Value
_SEGGER_RTT	SEGGER_RTT_CB	{...}
acID	char [16]	0x20012ec4 <_SEGGER_RTT>
MaxNumUpBuf	int	3
MaxNumDownE	int	3
aUp	SEGGER_RTT_BUFFE	0x20012edc <_SEGGER_RTT+24>
aUp[0]	SEGGER_RTT_BUFFE	{...}
aUp[1]	SEGGER_RTT_BUFFE	{...}
sName	const char *	0x8006128 "SysView"
pBuffer	char *	0x2001337c <_UpBuffer> ""
SizeOfBuf	unsigned int	4096
WrOff	unsigned int	4094
RdOff	volatile unsigned int	0
Flags	unsigned int	0
aUp[2]	SEGGER_RTT_BUFFE	{...}
aDown	SEGGER_RTT_BUFFE	0x20012f24 <_SEGGER_RTT+96>
Add new expressio		

```
Name : pBuffer
    Details:0x2001337c <_UpBuffer> ""
    Default:0x2001337c <_UpBuffer> ""
    Decimal:536949628
    Hex:0x2001337c
    Binary:100000000000010011001101111100
    Octal:04000231574
```

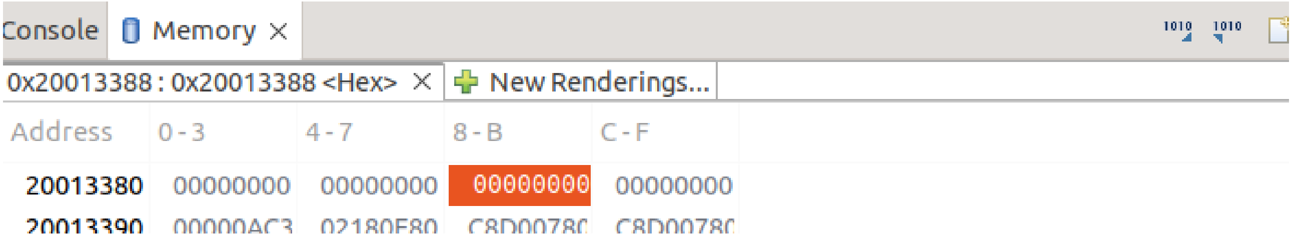
now copy the values of pbuffer as “0X2001337c”

and add into the memory



Address	0 - 3	4 - 7	8 - B	C - F
20013370	00000000	00000000	00000000	00000000
20013380	00000000	00000AC3	02180E80	C8D0078C
20013390	C8D0078C	80808001	00D3050E	3C4E3D4D
200133A0	79204672	65655254	4F532041	70706C69
200133B0	63617469	6F6E2C44	3D436F72	7465782D
200133C0	4D342053	544D3332	20626173	65642C4F
200133D0	3D467265	6552544F	53D7120E	0C492331
200133E0	353D5379	73546963	6B9E080D	00008C05

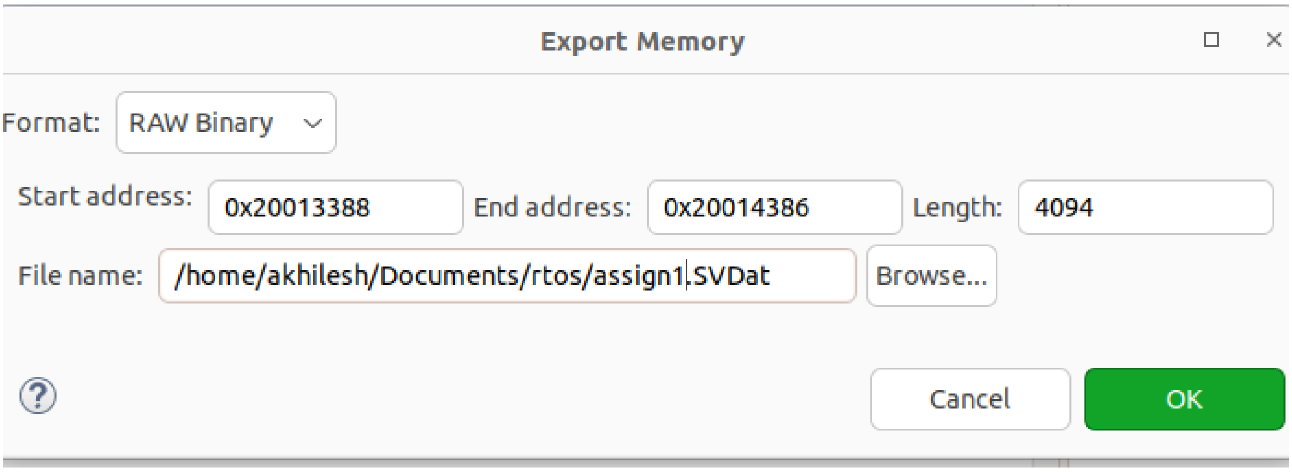
After that export that it by select...



0x20013388 : 0x20013388 <Hex>

```
Address   0-3       4-7       8-B       C-F
20013380  00000000  00000000  00000000  00000000
20013390  00000AC3  02180E80  C8D0078C  C8D0078C
```

Select format as Raw binay and length as we got 4094 and file name with extention of “ .SVDat ”



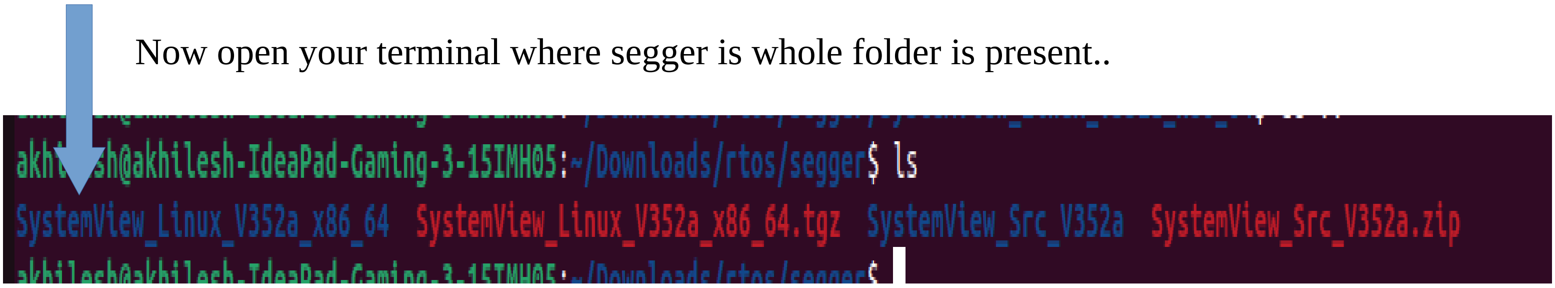
Export Memory

Format: RAW Binary

Start address: 0x20013388
End address: 0x20014386
Length: 4094

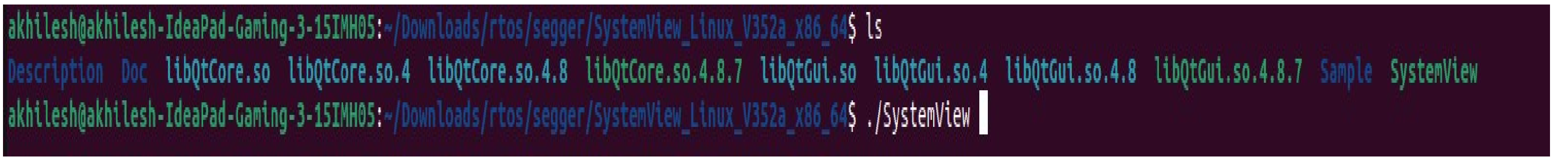
File name: /home/akhilesh/Documents/rtos/assign1.SVDat

Now open your terminal where segger is whole folder is present..



```
akhilesh@akhilesh-IdeaPad-Gaming-3-15IMH05:~/Downloads/rtos/segger$ ls
SystemView_Linux_V352a_x86_64  SystemView_Linux_V352a_x86_64.tgz  SystemView_Src_V352a  SystemView_Src_V352a.zip
akhilesh@akhilesh-IdeaPad-Gaming-3-15IMH05:~/Downloads/rtos/segger$
```

now step into the directory that one marked in uper image.

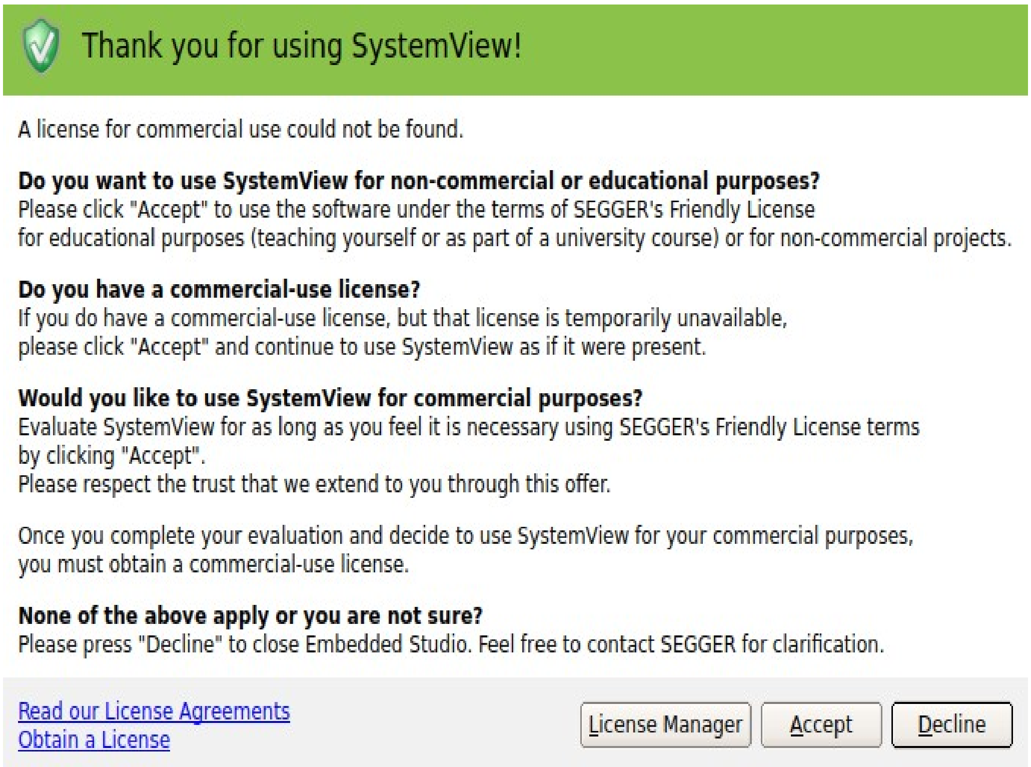


```
akhilesh@akhilesh-IdeaPad-Gaming-3-15IMH05:~/Downloads/rtos/segger/SystemView_Linux_V352a_x86_64$ ls
Description  Doc  libQtCore.so  libQtCore.so.4  libQtCore.so.4.8  libQtCore.so.4.8.7  libQtGui.so  libQtGui.so.4  libQtGui.so.4.8  libQtGui.so.4.8.7  Sample  SystemView
akhilesh@akhilesh-IdeaPad-Gaming-3-15IMH05:~/Downloads/rtos/segger/SystemView_Linux_V352a_x86_64$ ./SystemView
```

Now type on the terminal...

$ ./SystemView

After that pop up comes like this...
Select Acccept



Thank you for using SystemView!

A license for commercial use could not be found.

Do you want to use SystemView for non-commercial or educational purposes?
Please click "Accept" to use the software under the terms of SEGGER's Friendly License
for educational purposes (teaching yourself or as part of a university course) or for non-commercial projects.

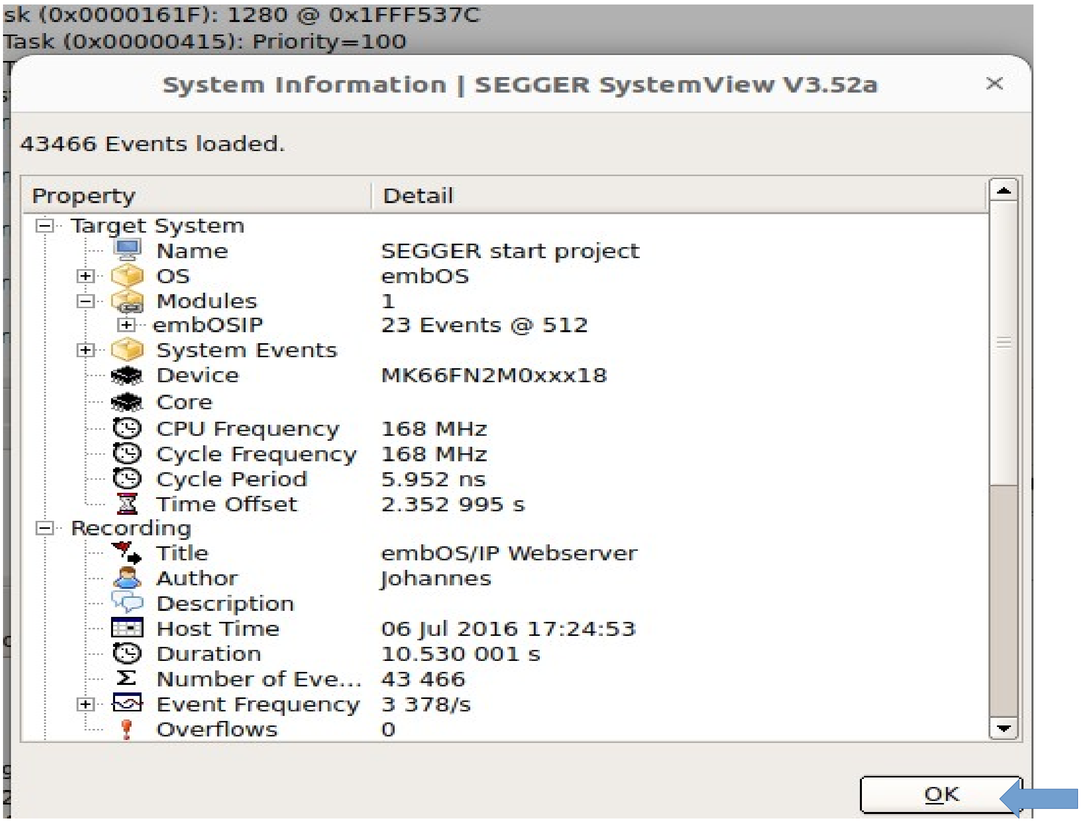
Do you have a commercial-use license?
If you do have a commercial-use license, but that license is temporarily unavailable,
please click "Accept" and continue to use SystemView as if it were present.

Would you like to use SystemView for commercial purposes?
Evaluate SystemView for as long as you feel it is necessary using SEGGER's Friendly License terms
by clicking "Accept".
Please respect the trust that we extend to you through this offer.

Once you complete your evaluation and decide to use SystemView for your commercial purposes,
you must obtain a commercial-use license.

None of the above apply or you are not sure?
Please press "Decline" to close Embedded Studio. Feel free to contact SEGGER for clarification.

Read our License Agreements
Obtain a License



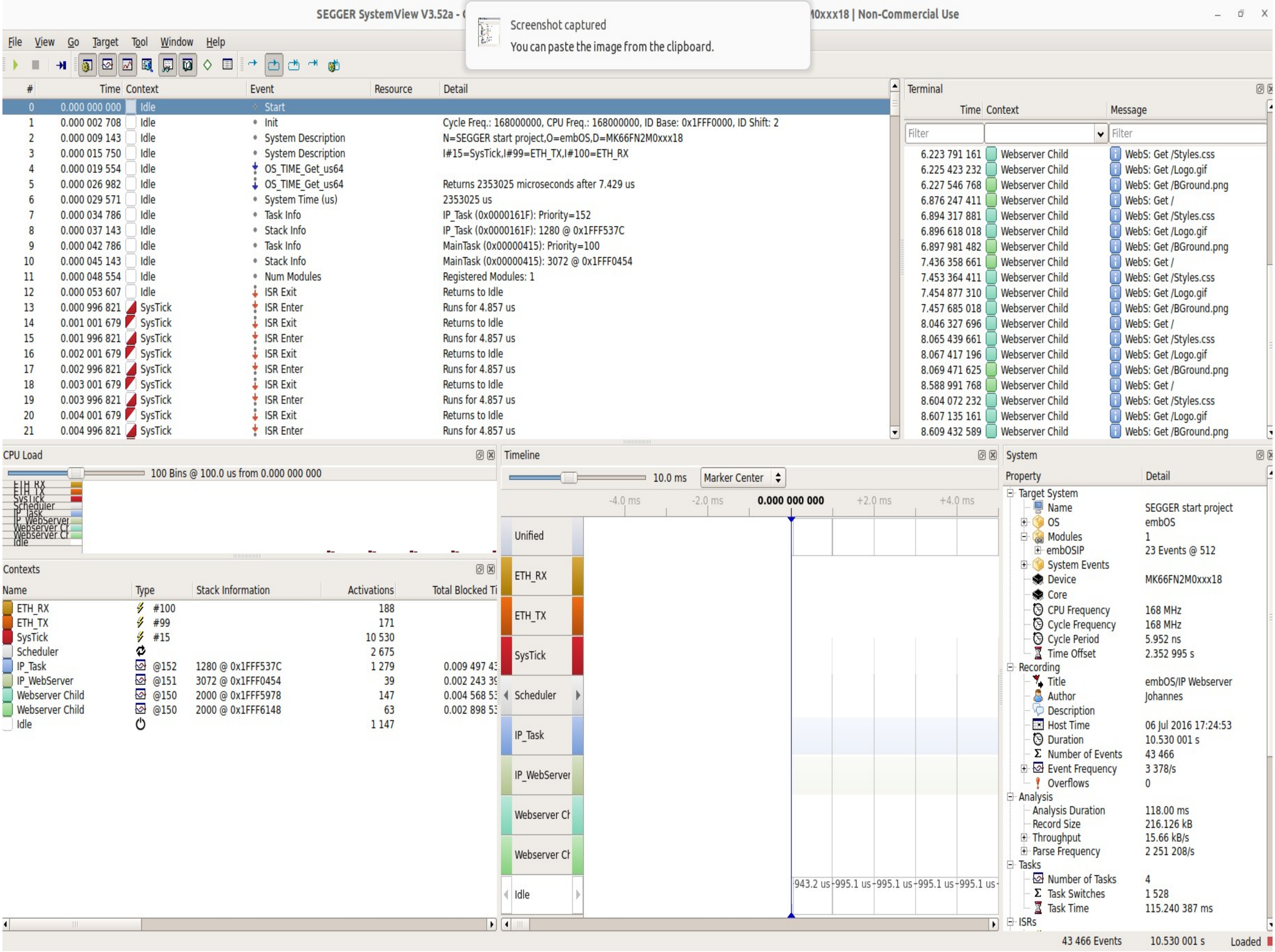
sk (0x0000161F): 1280 @ 0x1FFF537C
Task (0x00000415): Priority=100

System Information | SEGGER SystemView V3.52a

43466 Events loaded.

Property	Detail
Target System	
Name	SEGGER start project
OS	embOS
Modules	1
embOSIP	23 Events @ 512
System Events	
Device	MK66FN2M0xxx18
Core	
CPU Frequency	168 MHz
Cycle Frequency	168 MHz
Cycle Period	5.952 ns
Time Offset	2.352 995 s
Recording	
Title	embOS/IP Webserver
Author	Johannes
Description	
Host Time	06 Jul 2016 17:24:53
Duration	10.530 001 s
Number of Eve...	43 466
Event Frequency	3 378/s
Overflows	0

Once done ! The SEGGER Systmeview open like that...



Screenshot captured

You can paste the image from the clipboard.

#	Time	Context	Event	Resource	Detail
0	0.000 000 000	Idle	Start		
1	0.000 002 708	Idle	Init		Cycle Freq.: 168000000, CPU Freq.: 168000000, ID Base: 0x1FFF0000, ID Shift: 2
2	0.000 009 143	Idle	System Description		N=SEGGER start project,O=embOS,D=MK66FN2M0xxx18
3	0.000 015 750	Idle	System Description		I#15=SysTick,I#99=ETH_TX,I#100=ETH_RX
4	0.000 019 554	Idle	OS_TIME_Get_us64		
5	0.000 026 982	Idle	OS_TIME_Get_us64		Returns 2353025 microseconds after 7.429 us
6	0.000 029 571	Idle	System Time (us)		2353025 us
7	0.000 034 786	Idle	Task Info		IP_Task (0x0000161F): Priority=152
8	0.000 037 143	Idle	Stack Info		IP_Task (0x0000161F): 1280 @ 0x1FFF537C
9	0.000 042 786	Idle	Task Info		MainTask (0x00000415): Priority=100
10	0.000 045 143	Idle	Stack Info		MainTask (0x00000415): 3072 @ 0x1FFF0454
11	0.000 048 554	Idle	Num Modules		Registered Modules: 1
12	0.000 053 607	Idle	ISR Exit		Returns to Idle
13	0.000 996 821	SysTick	ISR Enter		Runs for 4.857 us
14	0.001 001 679	SysTick	ISR Exit		Returns to Idle
15	0.001 996 821	SysTick	ISR Enter		Runs for 4.857 us
16	0.002 001 679	SysTick	ISR Exit		Returns to Idle
17	0.002 996 821	SysTick	ISR Enter		Runs for 4.857 us
18	0.003 001 679	SysTick	ISR Exit		Returns to Idle
19	0.003 996 821	SysTick	ISR Enter		Runs for 4.857 us
20	0.004 001 679	SysTick	ISR Exit		Returns to Idle
21	0.004 996 821	SysTick	ISR Enter		Runs for 4.857 us

Terminal

Time	Context	Message
Filter		Filter
6.223 791 161	Webserver Child	WebS: Get /Styles.css
6.225 423 232	Webserver Child	WebS: Get /Logo.gif
6.227 546 768	Webserver Child	WebS: Get /BGround.png
6.876 247 411	Webserver Child	WebS: Get /
6.894 317 881	Webserver Child	WebS: Get /Styles.css
6.896 618 018	Webserver Child	WebS: Get /Logo.gif
6.897 981 482	Webserver Child	WebS: Get /BGround.png
7.436 358 661	Webserver Child	WebS: Get /
7.453 364 411	Webserver Child	WebS: Get /Styles.css
7.454 877 310	Webserver Child	WebS: Get /Logo.gif
7.457 685 018	Webserver Child	WebS: Get /BGround.png
8.046 327 696	Webserver Child	WebS: Get /
8.065 439 661	Webserver Child	WebS: Get /Styles.css
8.067 417 196	Webserver Child	WebS: Get /Logo.gif
8.069 471 625	Webserver Child	WebS: Get /BGround.png
8.588 991 768	Webserver Child	WebS: Get /
8.604 072 232	Webserver Child	WebS: Get /Styles.css
8.607 135 161	Webserver Child	WebS: Get /Logo.gif
8.609 432 589	Webserver Child	WebS: Get /BGround.png

100 Bins @ 100.0 us from 0.000 000 000

ETH_RX
ETH_TX
SysTick
Scheduler
IP_Task
IP_WebServer
Webserver Ch
Webserver Ch
Idle

Contexts

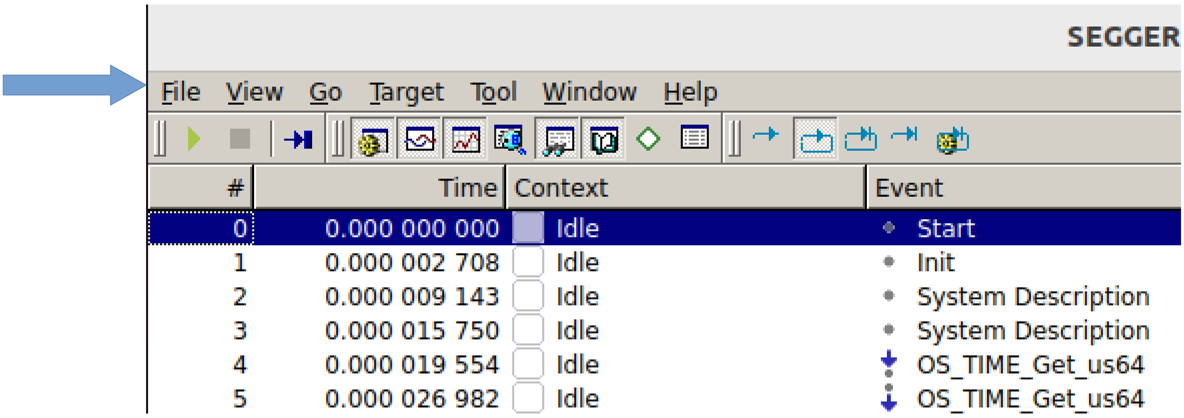
Name	Type	Stack Information	Activations	Total Blocked Ti
ETH_RX	#100		188	
ETH_TX	#99		171	
SysTick	#15		10 530	
Scheduler			2 675	
IP_Task	@152	1280 @ 0x1FFF537C	1 279	0.009 497 43
IP_WebServer	@151	3072 @ 0x1FFF0454	39	0.002 243 39
Webserver Child	@150	2000 @ 0x1FFF5978	147	0.004 568 53
Webserver Child	@150	2000 @ 0x1FFF6148	63	0.002 898 53
Idle			1 147	

System

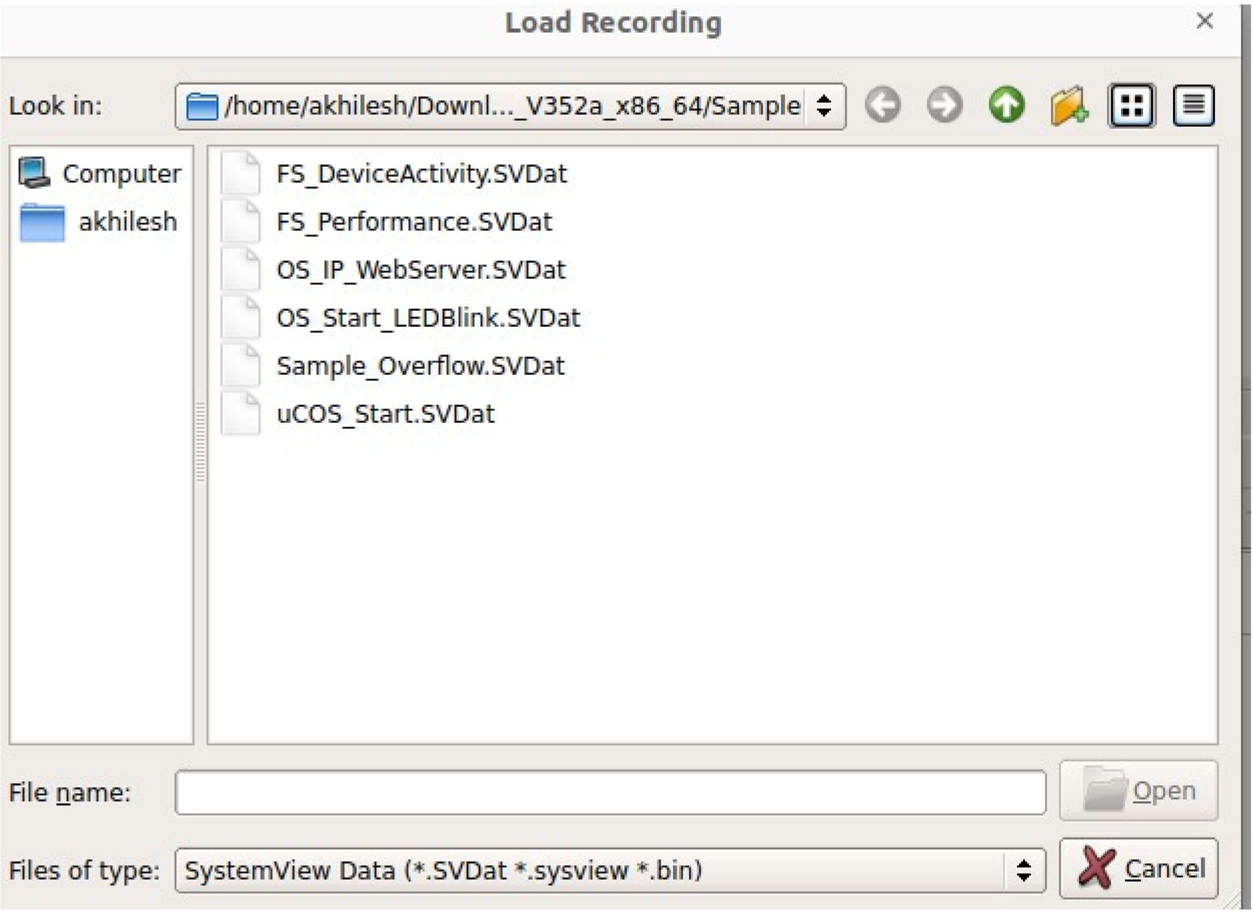
Property	Detail
Target System	
Name	SEGGER start project
OS	embOS
Modules	1
embOSIP	23 Events @ 512
System Events	
Device	MK66FN2M0xxx18
Core	
CPU Frequency	168 MHz
Cycle Frequency	168 MHz
Cycle Period	5.952 ns
Time Offset	2.352 995 s
Recording	
Title	embOS/IP Webserver
Author	Johannes
Description	
Host Time	06 Jul 2016 17:24:53
Duration	10.530 001 s
Number of Events	43 466
Event Frequency	3 378/s
Overflows	0
Analysis	
Analysis Duration	118.00 ms
Record Size	216.126 kB
Throughput	15.66 kB/s
Parse Frequency	2 251 208/s
Tasks	
Number of Tasks	4
Task Switches	1 528
Task Time	115.240 387 ms
ISRs	

43 466 Events   10.530 001 s   Loaded

Now open your raw binary file(.SVDat) file that we are generated in the STMCube32 IDE...



#	Time	Context	Event
0	0.000 000 000	Idle	Start
1	0.000 002 708	Idle	Init
2	0.000 009 143	Idle	System Description
3	0.000 015 750	Idle	System Description
4	0.000 019 554	Idle	OS_TIME_Get_us64
5	0.000 026 982	Idle	OS_TIME_Get_us64



Load Recording

Look in: /home/akhilesh/Downl..._V352a_x86_64/Sample

Computer
akhilesh

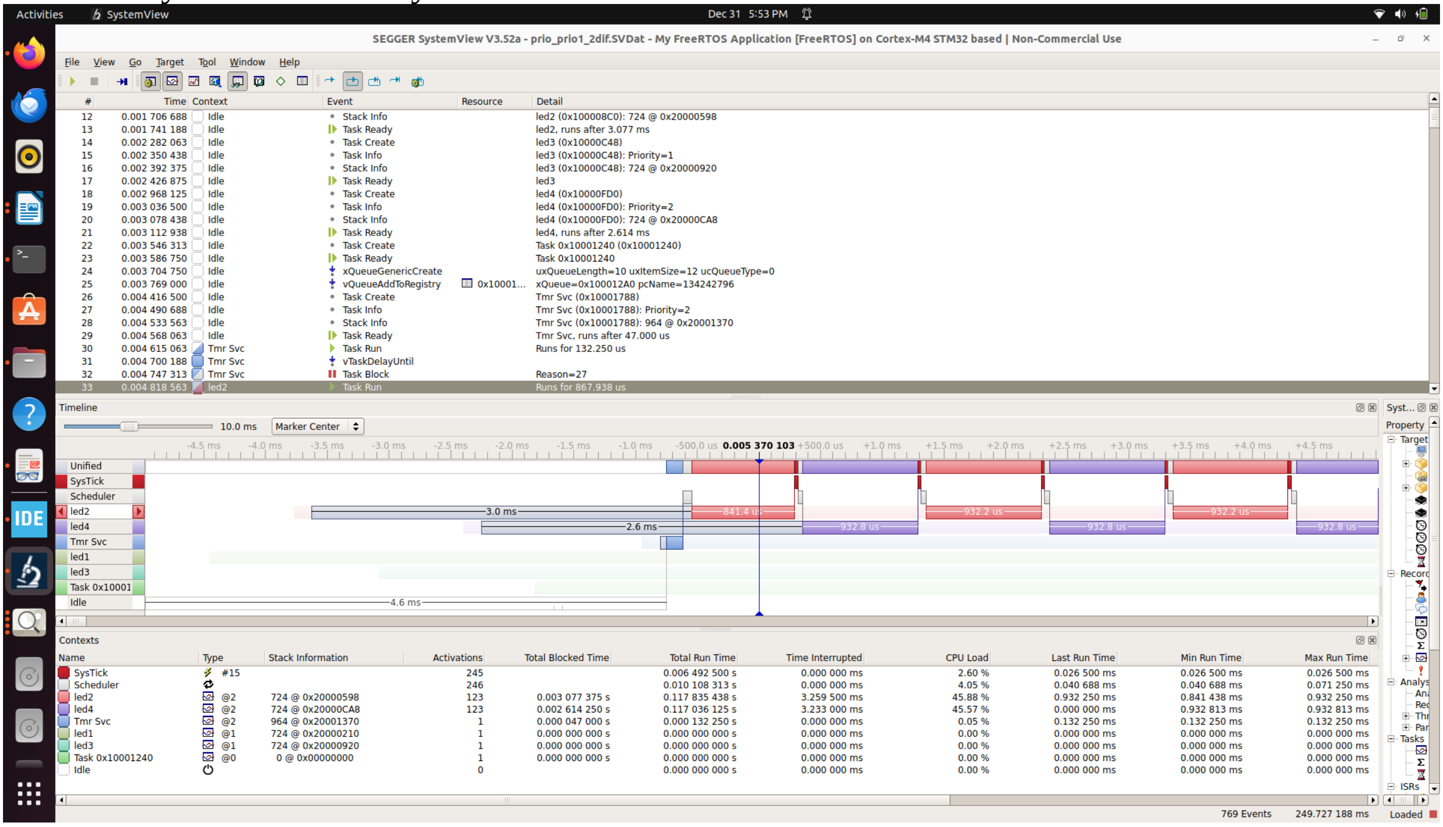
FS_DeviceActivity.SVDat
FS_Performance.SVDat
OS_IP_WebServer.SVDat
OS_Start_LEDBlink.SVDat
Sample_Overflow.SVDat
uCOS_Start.SVDat

File name:

Files of type: SystemView Data (*.SVDat *.sysview *.bin)

Select your file and select the open!

Now you can visulise your task.....



Activities   SystemView   Dec 31 5:53 PM

SEGGER SystemView V3.52a - prio_prio1_2dif.SVDat - My FreeRTOS Application [FreeRTOS] on Cortex-M4 STM32 based | Non-Commercial Use

#	Time	Context	Event	Resource	Detail
12	0.001 706 688	Idle	Stack Info		led2 (0x100008C0): 724 @ 0x20000598
13	0.001 741 188	Idle	Task Ready		led2, runs after 3.077 ms
14	0.002 282 063	Idle	Task Create		led3 (0x10000C48)
15	0.002 350 438	Idle	Task Info		led3 (0x10000C48): Priority=1
16	0.002 392 375	Idle	Stack Info		led3 (0x10000C48): 724 @ 0x20000920
17	0.002 426 875	Idle	Task Ready		led3
18	0.002 968 125	Idle	Task Create		led4 (0x10000FD0)
19	0.003 036 500	Idle	Task Info		led4 (0x10000FD0): Priority=2
20	0.003 078 438	Idle	Stack Info		led4 (0x10000FD0): 724 @ 0x20000CA8
21	0.003 112 938	Idle	Task Ready		led4, runs after 2.614 ms
22	0.003 546 313	Idle	Task Create		Task 0x10001240 (0x10001240)
23	0.003 586 750	Idle	Task Ready		Task 0x10001240
24	0.003 704 750	Idle	xQueueGenericCreate		uxQueueLength=10 uxItemSize=12 ucQueueType=0
25	0.003 769 000	Idle	vQueueAddToRegistry	0x10001...	xQueue=0x100012A0 pcName=134242796
26	0.004 416 500	Idle	Task Create		Tmr Svc (0x10001788)
27	0.004 490 688	Idle	Task Info		Tmr Svc (0x10001788): Priority=2
28	0.004 533 563	Idle	Stack Info		Tmr Svc (0x10001788): 964 @ 0x20001370
29	0.004 568 063	Idle	Task Ready		Tmr Svc, runs after 47.000 us
30	0.004 615 063	Tmr Svc	Task Run		Runs for 132.250 us
31	0.004 700 188	Tmr Svc	vTaskDelayUntil		
32	0.004 747 313	Tmr Svc	Task Block		Reason=27
33	0.004 818 563	led2	Task Run		Runs for 867.938 us

Timeline

10.0 ms   Marker Center

Contexts

Name	Type	Stack Information	Activations	Total Blocked Time	Total Run Time	Time Interrupted	CPU Load	Last Run Time	Min Run Time	Max Run Time
🟥 SysTick	#15		245		0.006 492 500 s	0.000 000 ms	2.60 %	0.026 500 ms	0.026 500 ms	0.026 500 ms
Scheduler			246		0.010 108 313 s	0.000 000 ms	4.05 %	0.040 688 ms	0.040 688 ms	0.071 250 ms
led2	@2	724 @ 0x20000598	123	0.003 077 375 s	0.117 835 438 s	3.259 500 ms	45.88 %	0.932 250 ms	0.841 438 ms	0.932 250 ms
led4	@2	724 @ 0x20000CA8	123	0.002 614 250 s	0.117 036 125 s	3.233 000 ms	45.57 %	0.000 000 ms	0.932 813 ms	0.932 813 ms
Tmr Svc	@2	964 @ 0x20001370	1	0.000 047 000 s	0.000 132 250 s	0.000 000 ms	0.05 %	0.132 250 ms	0.132 250 ms	0.132 250 ms
led1	@1	724 @ 0x20000210	1	0.000 000 000 s	0.000 000 000 s	0.000 000 ms	0.00 %	0.000 000 ms	0.000 000 ms	0.000 000 ms
led3	@1	724 @ 0x20000920	1	0.000 000 000 s	0.000 000 000 s	0.000 000 ms	0.00 %	0.000 000 ms	0.000 000 ms	0.000 000 ms
Task 0x10001240	@0	0 @ 0x00000000	1	0.000 000 000 s	0.000 000 000 s	0.000 000 ms	0.00 %	0.000 000 ms	0.000 000 ms	0.000 000 ms
Idle			0		0.000 000 000 s	0.000 000 ms	0.00 %	0.000 000 ms	0.000 000 ms	0.000 000 ms

769 Events   249.727 188 ms   Loaded

All done!!