

FreeRTOS and emWin for i.MX RT platform

Krzysztof Chojnowski



SOM

System on Module

CB

Carrier Board

DK

Development Kit

Engineering

Since 2003 delivering proven designs

Agenda

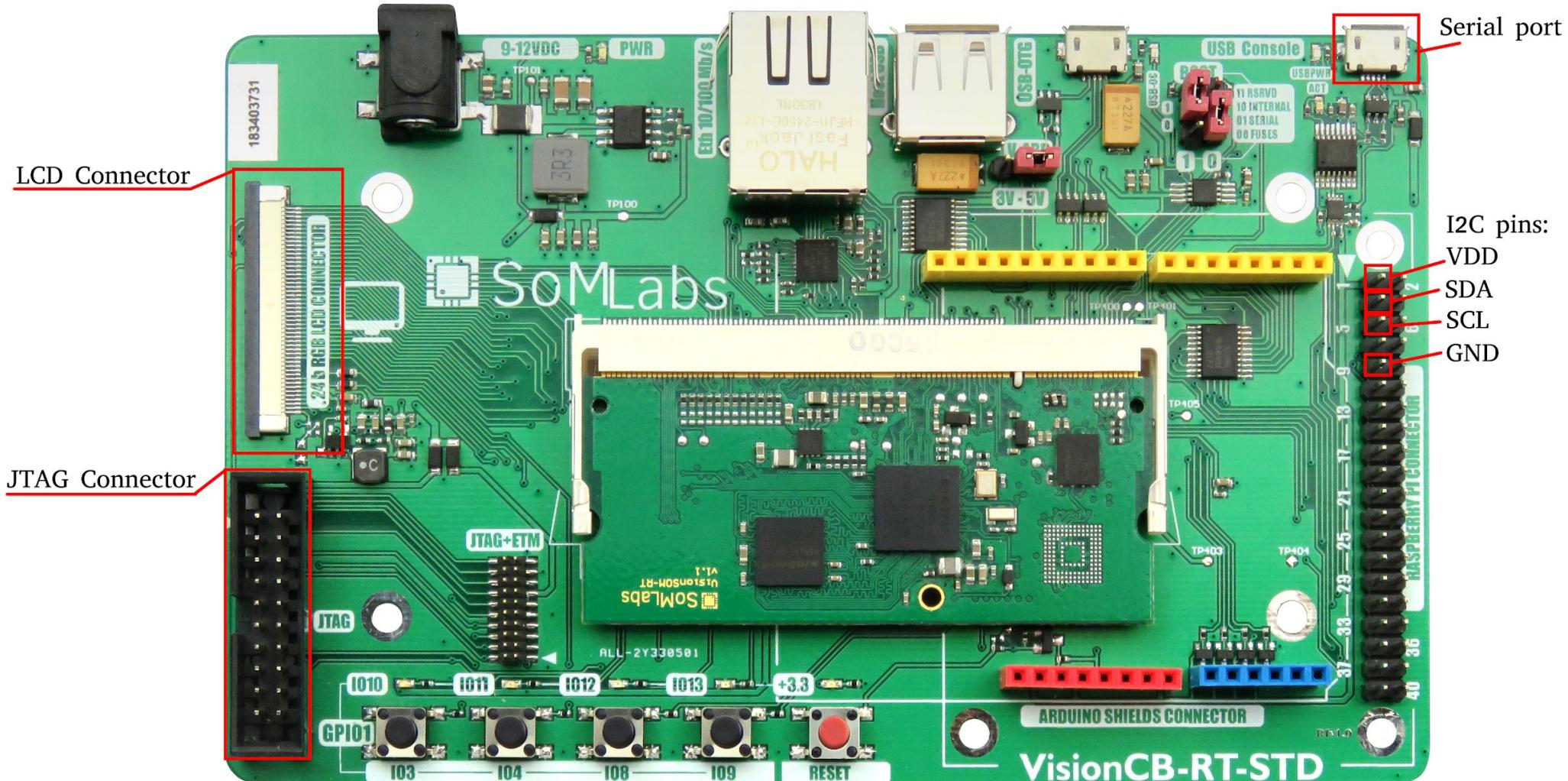
- Introduction to the development environment
- Introduction to (Free)RTOS
- Creating and synchronizing FreeRTOS tasks
- Tasks communication
- emWin library

Proven Partner

Exercises

- Exercise 0 - building and running example project
- Exercise 1 - creating tasks
- Exercise 2 - handling interrupts
- Exercise 3 - queues
- Exercise 4 - building GUI with emWin
- Exercise 5 - handling GUI events

VisionCB-RT-STD



SDK

i.MX RT1050 Crossover Processor with Arm® Cortex®-M7 core

Follow  

OVERVIEW

DOCUMENTATION

TOOLS & SOFTWARE

BUY/PARAMETRICS

PACKAGE/QUALITY

TRAINING & SUPPORT

Recommended Tools and Software



i.MX RT1050 Evaluation Kit

The i.MX RT1050 EVK is an entry-level, low-cost development platform featuring the i.MX RT1050 crossover processor based on Arm Cortex-M7.

 Buy Options



MCUXpresso Software Development Kit (SDK)

Software development kit for designing with Kinetis, LPC, i.MX controllers based on Arm Cortex-M cores, with integrated stacks, middleware, examples

Download Options

Software Development Kits



MCUXpresso Integrated Development Environment (IDE)

Easy-to-use software development tools for Kinetis, LPC, i.MX controllers based on Arm Cortex-M cores - GNU, Eclipse, profiling, debugger, trace

Download Options

Proven Partner



SoMLabs | www.somlabs.com



SDK

MCUXpresso Software Development Kit (SDK)

[Follow](#)[OVERVIEW](#)[DOCUMENTATION](#)[DOWNLOADS](#)[DEVELOPMENT TOOLS](#)[TRAINING & SUPPORT](#)

Jump To

[Overview & Features](#)
[Supported Devices](#)
[Target Applications](#)
[System Requirements](#)

Overview

The MCUXpresso SDK is a comprehensive software enablement package designed to simplify and accelerate application development with NXP's LPC and Kinetis® microcontrollers and i.MX RT crossover processors based on Arm® Cortex®-M cores. The MCUXpresso SDK includes production-grade software with integrated RTOS (optional), integrated stacks and middleware, reference software, and more.

Underscoring highest quality, the MCUXpresso SDK is MISRA compliant and checked with Coverity® static analysis tools and is available in custom downloads based on user selections of MCU, evaluation board, and optional software components.

[More ▾](#)[User Guide](#)[Download](#)

Features

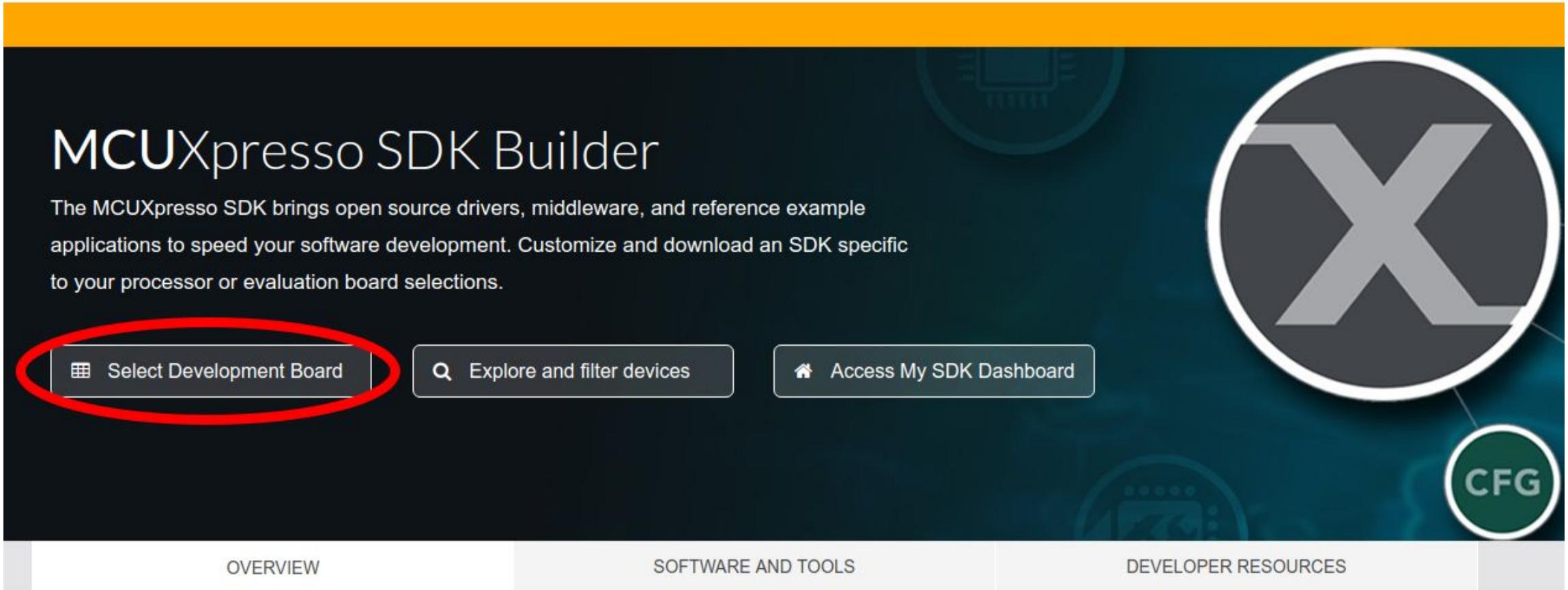
The MCUXpresso SDK consists of the following runtime software components:

- Arm® CMSIS-CORE startup and device header files and CMSIS-DSP standard libraries
- Open-source peripheral drivers that provide stateless, high performance, easy-to-use APIs
- Drivers for communication peripherals also include high-level transactional APIs for high-performance data transfers and RTOS wrappers that leverage native RTOS services to better comply with the RTOS cases
- High-quality software: all drivers and startup code are

Proven Partner

SoMLabs | www.somlabs.com

SDK



The screenshot shows the MCUXpresso SDK Builder landing page. At the top, there's a yellow header bar with the word "SDK" in green. Below it is a dark blue header with the title "MCUXpresso SDK Builder". To the right is a large circular icon containing a stylized "X". Below the title, there's a brief description: "The MCUXpresso SDK brings open source drivers, middleware, and reference example applications to speed your software development. Customize and download an SDK specific to your processor or evaluation board selections." Three buttons are visible: "Select Development Board" (circled in red), "Explore and filter devices", and "Access My SDK Dashboard". At the bottom, there are three tabs: "OVERVIEW", "SOFTWARE AND TOOLS", and "DEVELOPER RESOURCES".

SDK

Select Development Board

Search for your board or kit to get started.

Search by Name

mimxrt1052

Select a Device, Board, or Kit

▼ Boards

▼ Kits

▼ Processors

MIMXRT1052xxxxB

MIMXRT1052xxxx



Deprecated

Name your SDK

SDK_2.5.0_MIMXRT1052xxxxB

Don't use: <, >, :, ;, /, |, ?, *, \ in the name of your SDK



Hardware Details

Included Part Numbers	MIMXRT1052CVL5B, MIMXRT1052DVL6B, MIMXRT1052CVJ5B, MIMXRT1052DVJ6B
Board(s)	EVKB-IMXRT1050
Device	MIMXRT1052
Core Type / Max Freq	Cortex-M7F / 600MHz
Device Memory Size	0 KB Flash 512 KB RAM

Actions

→ Build MCUXpresso SDK



Explore selection with Clocks tool



Explore selection with Pins tool

Proven Partner



SoMLabs | www.somlabs.com



SDK

SDK Builder

Generate a downloadable SDK archive for use with desktop MCUXpresso Tools.

Developer Environment Settings

Selections here will impact files and examples projects included in the SDK and Generated Projects

Host OS: **Linux**

Toolchain / IDE: **MCUXpresso IDE**

Select Optional Middleware

Add middleware, operating systems, and software libraries to your SDK.

+ Add software component

Click the link below to request this specific MCUXpresso SDK Build

In general, SDK builds should complete within a few minutes.

You will be notified via email and notifications in the upper right corner of this webpage.

Request Build →

Archive Name:

SDK_2.5.0_MIMXRT1052xxxxB (3)

Don't use: <,>,;,",/,?,*,\ in the name of your SDK



Hardware Details

Included Part Numbers	MIMXRT1052CVL5B, MIMXRT1052DVL6B, MIMXRT1052CVJ5B, MIMXRT1052DVJ6B
Board(s)	EVKB-IMXRT1050
Device	MIMXRT1052
Core Type / Max Freq	Cortex-M7F / 600MHz
Device Memory Size	0 KB Flash 512 KB RAM

SDK Details

SDK Version:	2.5.0 (released 2018-12-17)
Host OS:	Linux
Toolchain:	MCUXpresso IDE
Middleware:	emWin, Amazon-FreeRTOS Kernel
Documentation:	MCUXpresso SDK API Reference Manual

i **SDK v2.5.x requires MCUXpresso IDE v10.3.x or later**

Proven Partner

SDK

MCUXpresso SDK Dashboard

Access, Download, and Share your requested SDK Builds.

My Recent SDKs

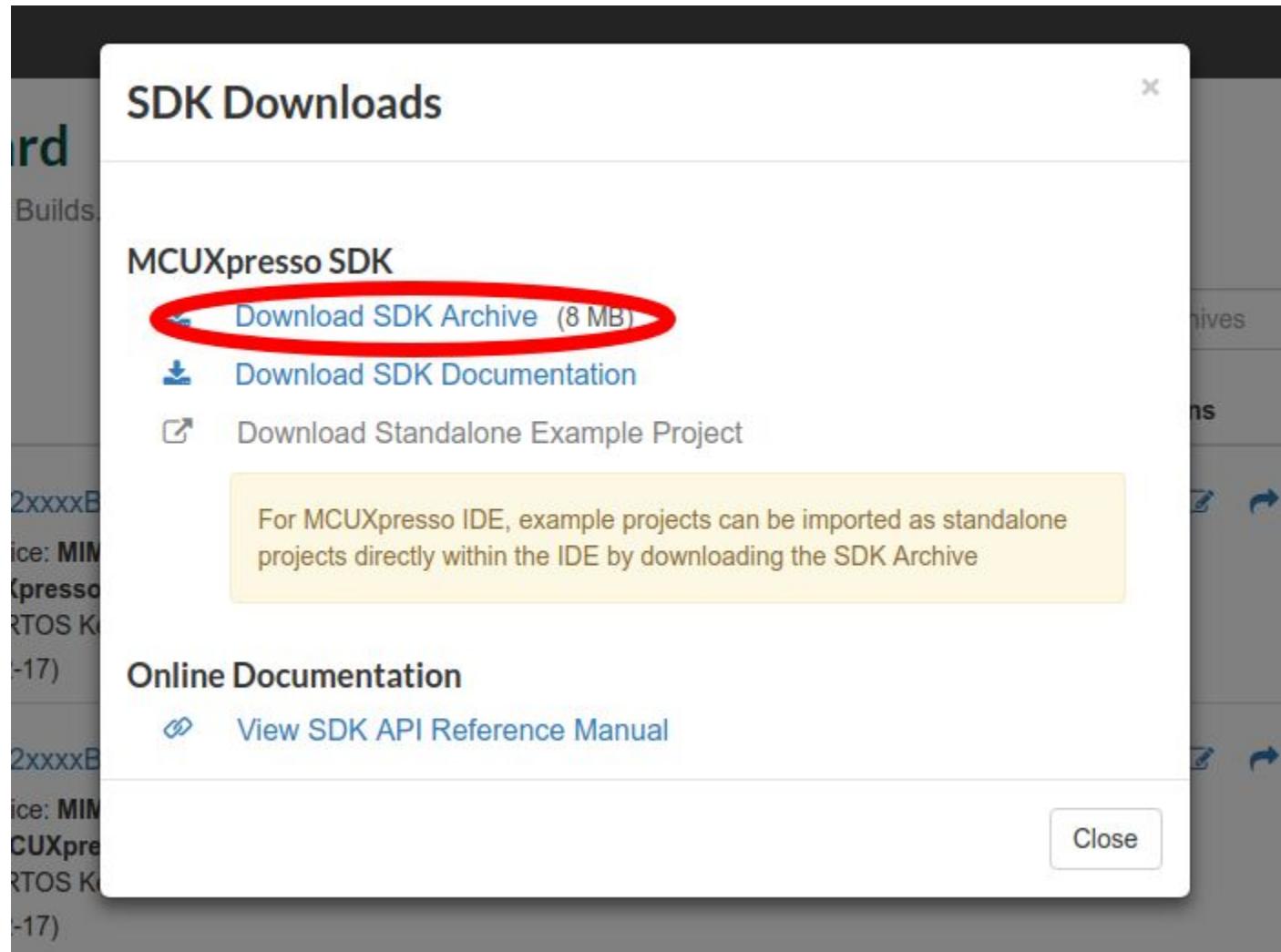
Search all archives



SDK Archive Details	Actions
 SDK_2.5.0_MIMXRT1052xxxxB (3)  NEW Build Date: 2019-02-13, Device: MIMXRT1052xxxxB OS: Linux, Toolchain: MCUXpresso IDE Components: Amazon-FreeRTOS Kernel, emWin SDK Version: 2.5.0 (2018-12-17)	    
 SDK_2.5.0_MIMXRT1052xxxxB  Build Date: 2019-01-09, Device: MIMXRT1052xxxxB OS: Windows, Toolchain: MCUXpresso IDE Components: Amazon-FreeRTOS Kernel, emWin SDK Version: 2.5.0 (2018-12-17)	    
 EVKB-IMXRT1050  Build Date: 2018-09-01, Board: EVKB-IMXRT1050 OS: Linux, Toolchain: MCUXpresso IDE Components: CMSIS DSP Library, Amazon-FreeRTOS Kernel, emWin, lwIP, FatFS, USB stack SDK Version: KSDK 2.4.2 (2018-08-02)	     Update Available

Proven Partner

SDK



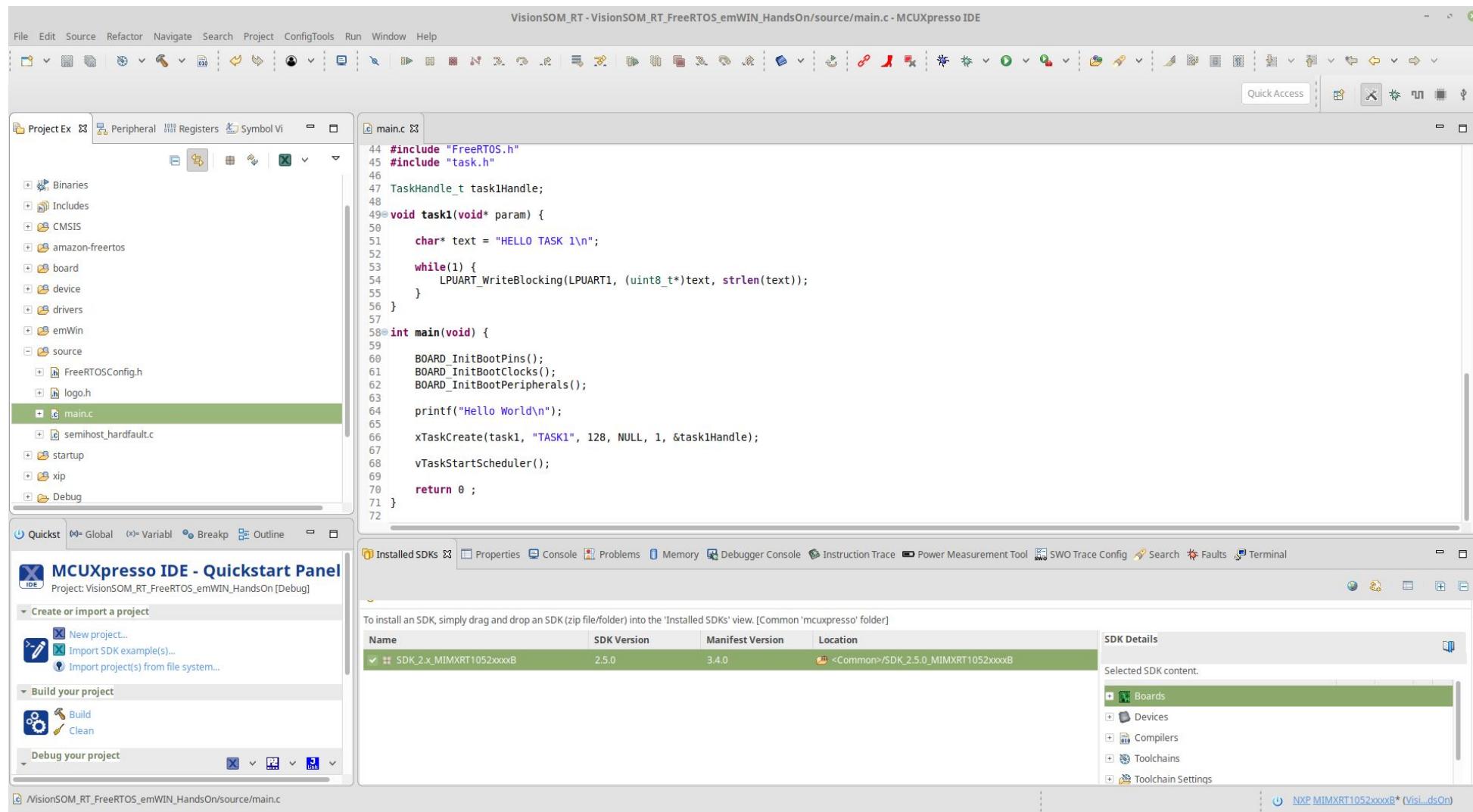
Proven Partner

SDK

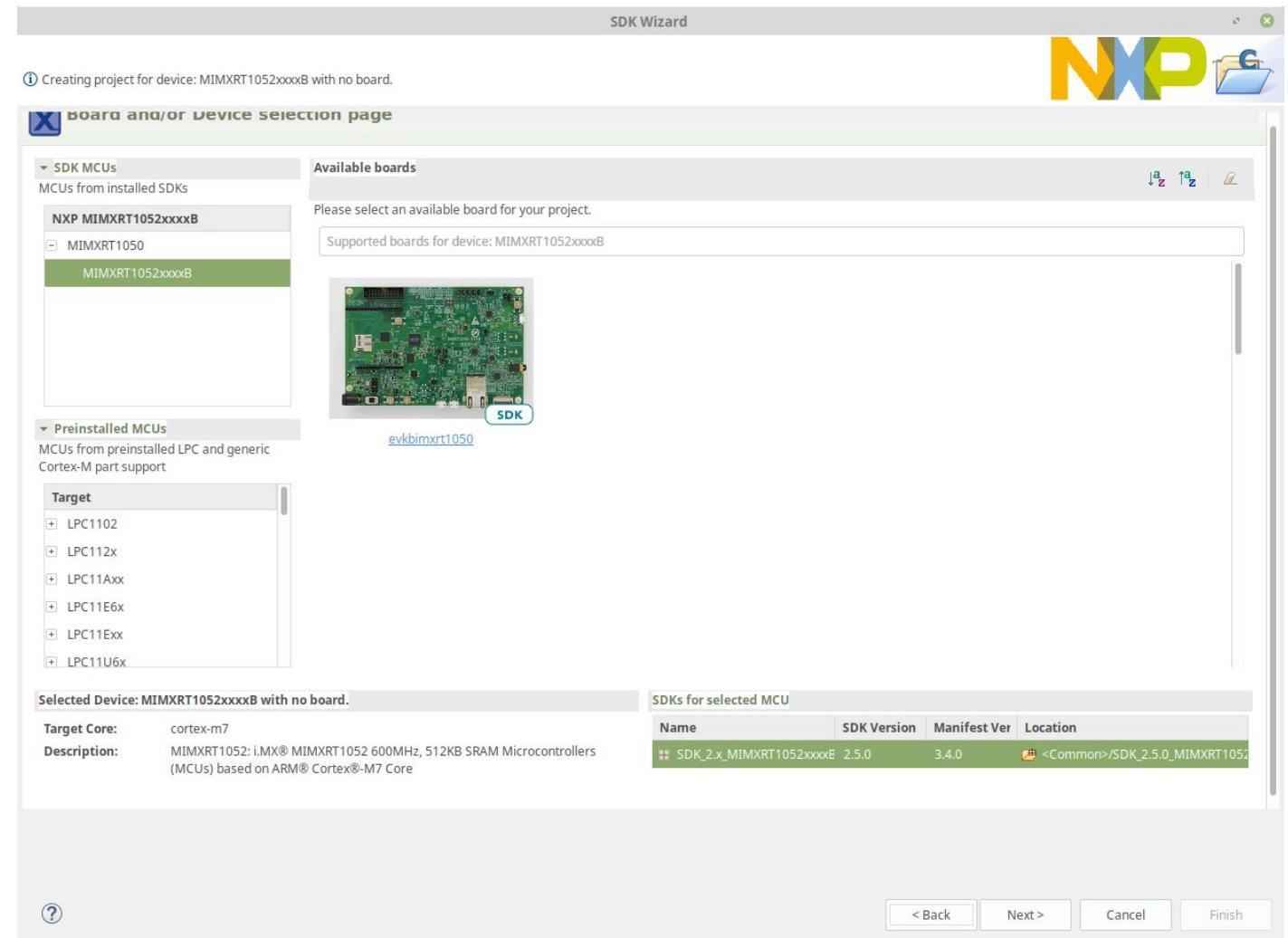
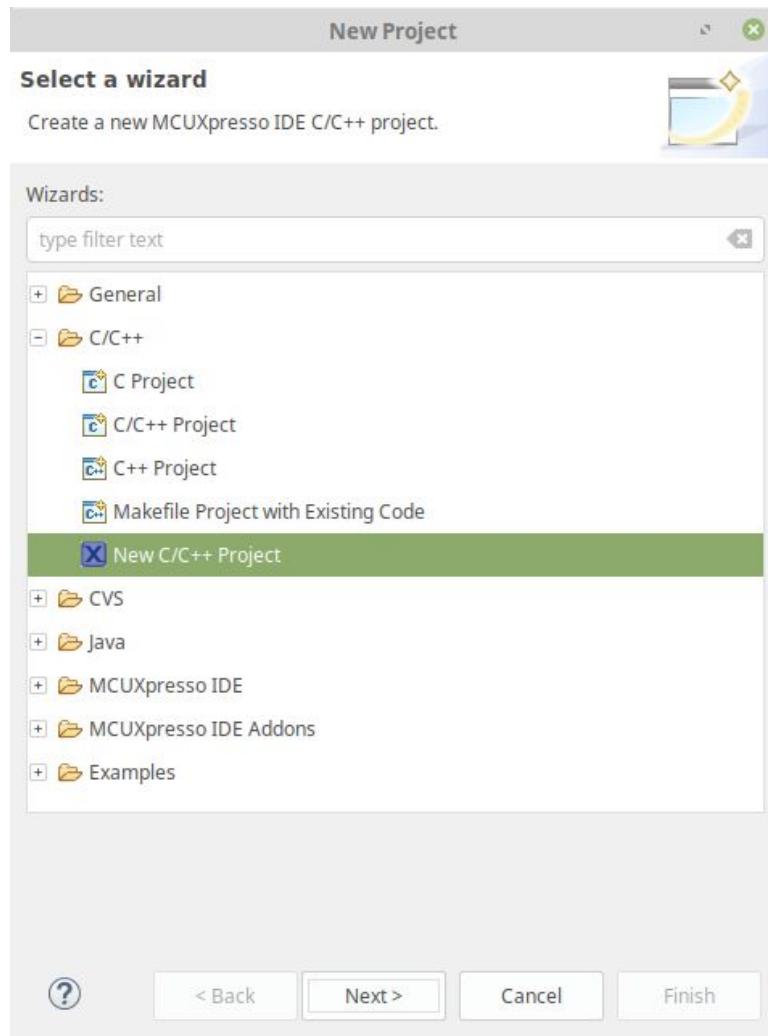
Name	Size	Type	Modified
tools	4,9 kB	Folder	13 February 2019, 17:01
components	376,9 kB	Folder	13 February 2019, 17:01
rtos	929,4 kB	Folder	13 February 2019, 17:01
docs	1,6 MB	Folder	13 February 2019, 17:01
CMSIS	1,9 MB	Folder	13 February 2019, 17:01
middleware	7,1 MB	Folder	13 February 2019, 17:01
boards	12,3 MB	Folder	13 February 2019, 17:01
devices	18,9 MB	Folder	13 February 2019, 17:01
SW-Content-Register.txt	5,5 kB	plain text docu...	12 February 2019, 17:01
</> MIMXRT1052xxxxB_manifest_v3_3.xml	147,1 kB	XML document	12 February 2019, 17:01
</> MIMXRT1052xxxxB_manifest_v3_4.xml	195,2 kB	XML document	12 February 2019, 17:01

Proven Partner

MCUXpresso



MCUXpresso



MCUXpresso

SDK Wizard

NXP

Configure the project

Project name: MIMXRT1052xxxxB_Project Project name suffix:

Use default location Location: /home/krzysiek/workspace/VisionSOM_RT/MIMXRT1052xxxxB_Project

Device Packages	Board	Project Type	Project Options
<input checked="" type="radio"/> MIMXRT1052CVL5B <input type="radio"/> MIMXRT1052CVJ5B ...	No board selected	<input checked="" type="radio"/> C Project <input type="radio"/> C++ Project <input type="radio"/> C Static Library <input type="radio"/> C++ Static Library	SDK Debug Console <input checked="" type="radio"/> Semihost <input type="radio"/> U... <input checked="" type="checkbox"/> CMSIS-Core <input checked="" type="checkbox"/> Copy sources <input checked="" type="checkbox"/> Import other files

OS	driver	CMSIS_driver	utilities	middleware
<input type="text" value="type to filter"/>	<input type="text" value="type to filter"/>	<input type="text" value="type to filter"/>	<input type="text" value="type to filter"/>	<input type="text" value="type to filter"/>
Name	Name	Name	Name	Name
<input type="checkbox"/> Amazon-FreeRTOS 10.0.1 <input checked="" type="checkbox"/> baremetal 1.0.0	<input type="checkbox"/> adc_12b1msps_sa 2.0.1 <input type="checkbox"/> adc_etc 2.0.1 <input type="checkbox"/> aipstz 2.0.0 <input type="checkbox"/> aoi 2.0.0 <input type="checkbox"/> bee 2.0.0 <input type="checkbox"/> cache 2.0.1 <input type="checkbox"/> clock 2.1.0 <input type="checkbox"/> cmp 2.0.0	<input checked="" type="checkbox"/> i2c_cmsis 2.0.0 <input type="checkbox"/> lpspi_cmsis 2.0.0 <input type="checkbox"/> lpuart_cmsis 2.0.1	<input type="checkbox"/> assert 1.0.0 <input type="checkbox"/> debug_console 1.0.0 <input type="checkbox"/> lpuart_adapter 1.0.0 <input type="checkbox"/> misc_utilities 1.0.0 <input type="checkbox"/> notifier 1.0.0 <input type="checkbox"/> serial_manager 1.0.0 <input type="checkbox"/> serial_manager_uai 1.0.0 <input type="checkbox"/> shell 1.0.0	<input type="checkbox"/> Image

?

< Back Next > Cancel Finish

MCUXpresso

SDK Wizard

NXP

Advanced project settings

C/C++ Library Settings

Set library type (and hosting variant) Redlib (semihost-nf)

- Redlib: Use floating point version of printf
- Redlib: Use character rather than string based printf
- Redirect SDK "PRINTF" to C library "printf"
- Include semihost HardFault handler
- NewlibNano: Use floating point version of printf
- NewlibNano: Use floating point version of scanf
- Redirect printf/scanf to ITM
- Redirect printf/scanf to UART

Hardware settings

Set Floating Point type FPv5-SP-D16 (HardABI)

MCU C Compiler

Language standard Compiler default

MCU Linker

Link application to RAM

Memory Configuration

Memory details

Default LinkServer Flash Driver

Type	Name	Alias	Location	Size	Driver
RAM	SRAM_DTC	RAM	0x2000000	0x20000	
RAM	SRAM_ITC	RAM2	0x0	0x20000	
RAM	SRAM_OC	RAM3	0x20200000	0x40000	

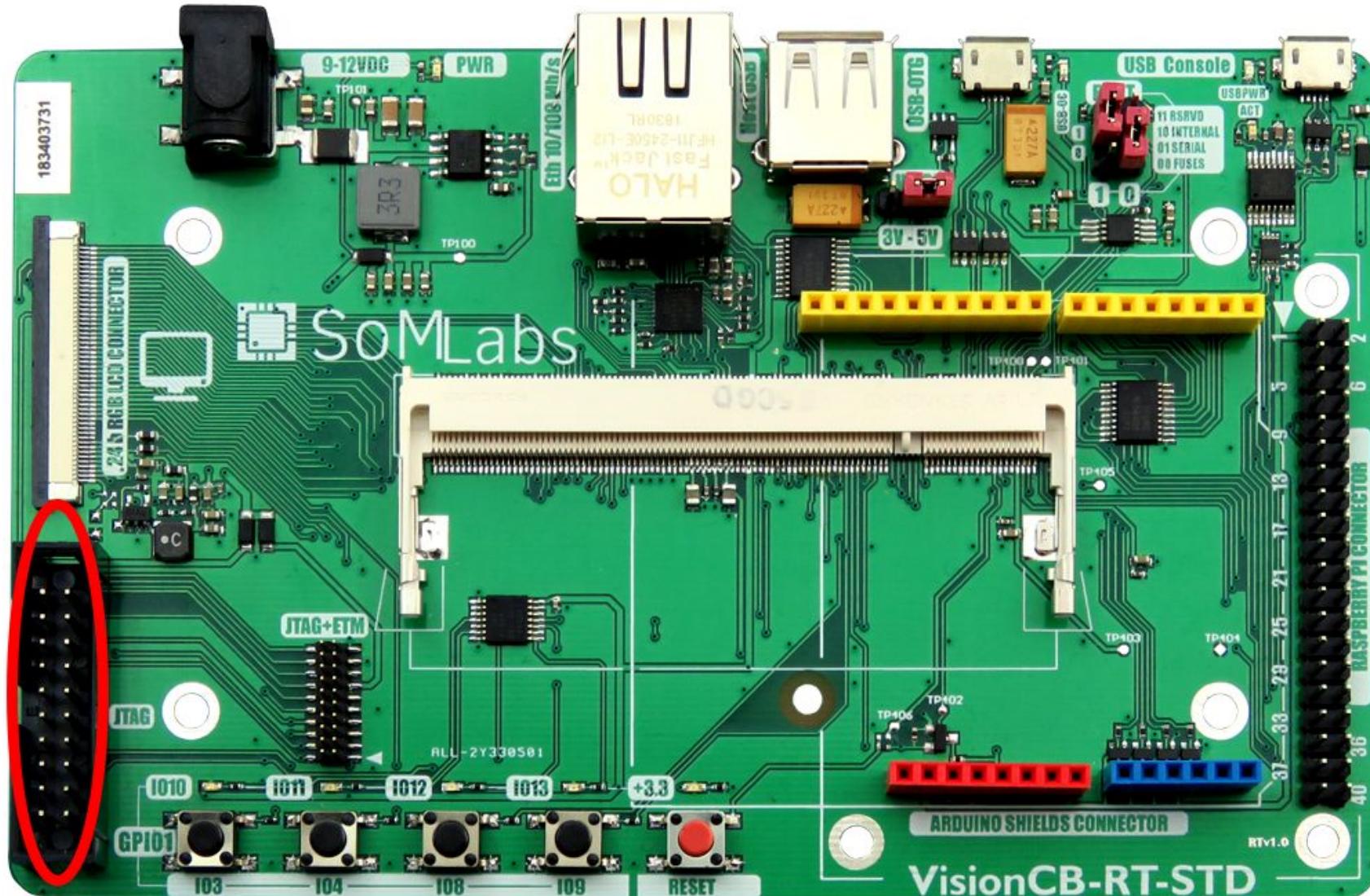
< Back Next > Cancel Finish

Exercise 0

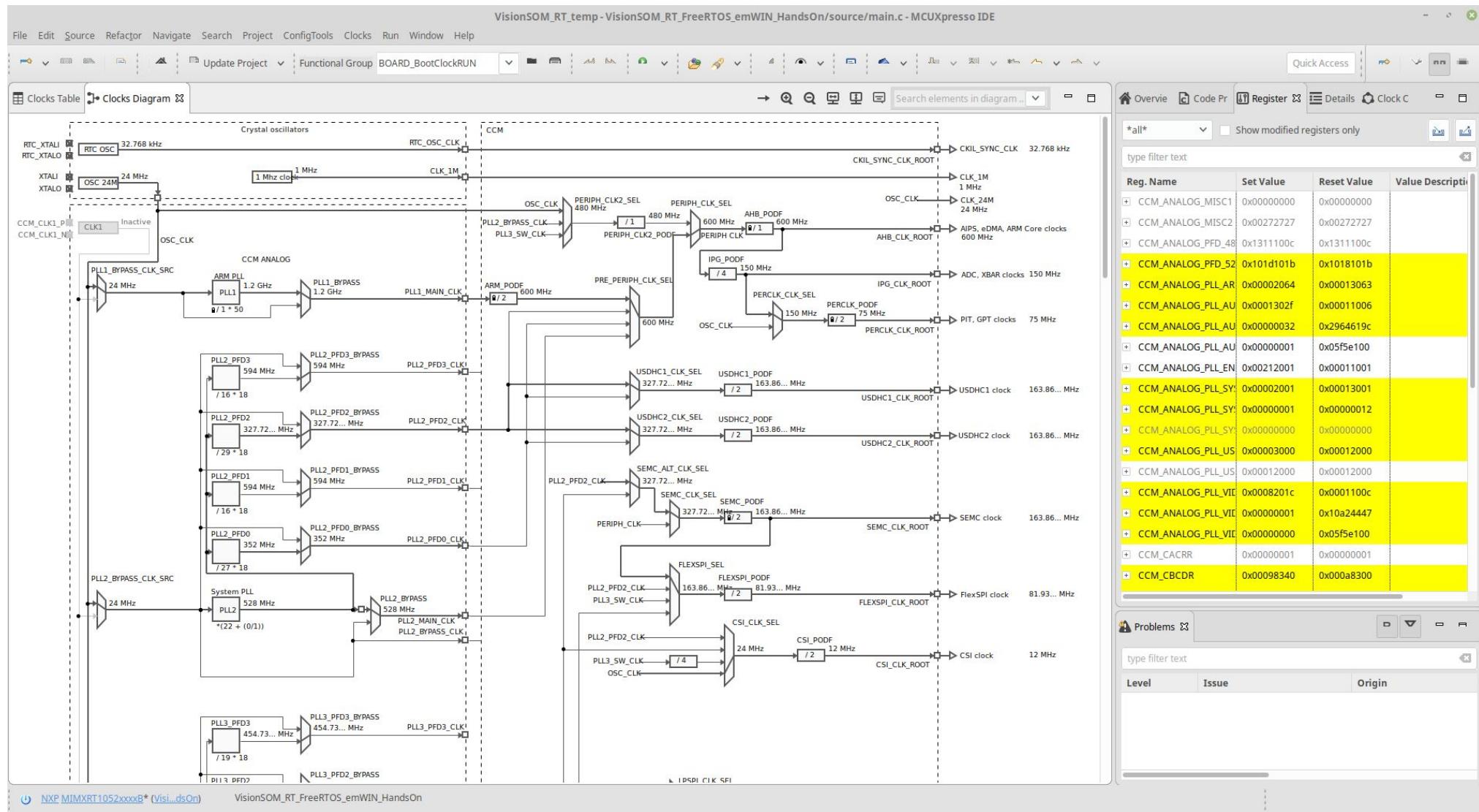
- Import SDK
- Import a template project
- build project

Proven Partner

Exercise 0



MCUXpresso



MCUXpresso

VisionSOM_RT_temp - VisionSOM_RT_FreeRTOS_emWIN_HandsOn/source/main.c - MCUXpresso IDE

The screenshot shows the MCUXpresso IDE interface with the following panels:

- Pins & Peripheral Signals**: A table listing pins (A1 to H1, J1 to P1) with their names, identifiers, and functional groups (GPIO, LPUART, PWM, FLEXIO).
- Package [Pins Bottom]**: A matrix diagram showing pin connections across multiple packages (A through P).
- Registers**: A table showing memory register settings for CCM_CCGR2 through DMAMUX_CHCFG15.
- Routed Pins**: A table detailing the routing of pins for BOARD_InitPins, including peripherals, signals, and route details.
- Problems**: A list of issues, including a warning about the LCDIF peripheral not being initialized.

MCUXpresso

The screenshot shows the MCUXpresso IDE interface for a project named "VisionSOM_RT_temp". The main window displays the configuration for the "Low Power Universal Asynchronous Receiver/Transmitter (LPUART)".

Peripheral Configuration:

- Name:** LPUART_1
- Mode:** Transfer
- Peripheral:** LPUART1

LPUART general configuration:

- Clock source: UART_CLK_ROOT - BOARD_BootClockRUN: 80 MHz
- Clock source frequency: 80 MHz (BOARD_BootClockRUN)
- LPUART baud rate: 115200
- Parity mode: Parity disabled
- Data size: Eight data bit
- Enable MSB data bits order:
- Number of stop bits: One stop bit
- TX FIFO watermark: 0
- RX FIFO watermark: 1
- RX RTS enable:
- TX CTS enable:
- TX CTS source: LPUART CTS pin
- TX CTS configuration: Sampled at the start
- RX IDLE type: Start counting after a valid start bit
- RX IDLE configuration: 1 idle character
- Enable TX:
- Enable RX:

Code Preview:

```
baudRate_Bps: '100000'  
/* BE CAREFUL MODIFYING THIS COMMENT - IT IS YAML SETTINGS */  
/* clang-format on */  
/* FlexIO peripheral configuration */  
FLEXIO_I2C_Type FlexIO_I2C_1_peripheralConfig = {  
    .flexioBase = FLEXIO_I2C_1_PERIPHERAL,  
    .SDAPinIndex = 29,  
    .SCLPinIndex = 28,  
    .shifterIndex = {0, 1},  
    .timerIndex = {0, 1}  
};  
/* I2C master configuration */  
flexio_i2c_master_config_t FlexIO_I2C_1_config = {  
    .enableMaster = true,  
    .enableInDoze = false,  
    .enableInDebug = true,  
    .enableFastAccess = false,  
    .baudRate_Bps = 100000  
};  
  
void FlexIO_I2C_1_init(void)  
{  
    /* Master initialization */  
    FLEXIO_I2C_MasterInit(&FlexIO_I2C_1_peripheralConfig, &FlexIO_I2C_1_config);  
}  
  
/* ***** * Initialization functions * ***** */  
void BOARD_InitPeripherals(void)  
{  
    /* Initialize components */  
    LPUART_1_init();  
    GPIO_1_init();  
    FlexIO_I2C_1_init();  
}
```

Problems:

Level	Issue	Origin
Warning	When GPIO1 interrupt 0-15 is disa	Peripherals:BOARD_InitPeriph
Warning	When GPIO1 interrupt 16-31 is dis	Peripherals:BOARD_InitPeriph
Warning	Peripheral LCDIF is not initialized	Pins:BOARD_InitPins

SEGGER J-Link

- Modification of the JLinkDevices.xml script for QSPI programming

/opt/SEGGER/JLink
C:\Program Files (x86)\SEGGER\JLink_V640

- Section

```
<!--          -->  
<!-- NXP (iMXRT105x) -->  
<!--          -->
```

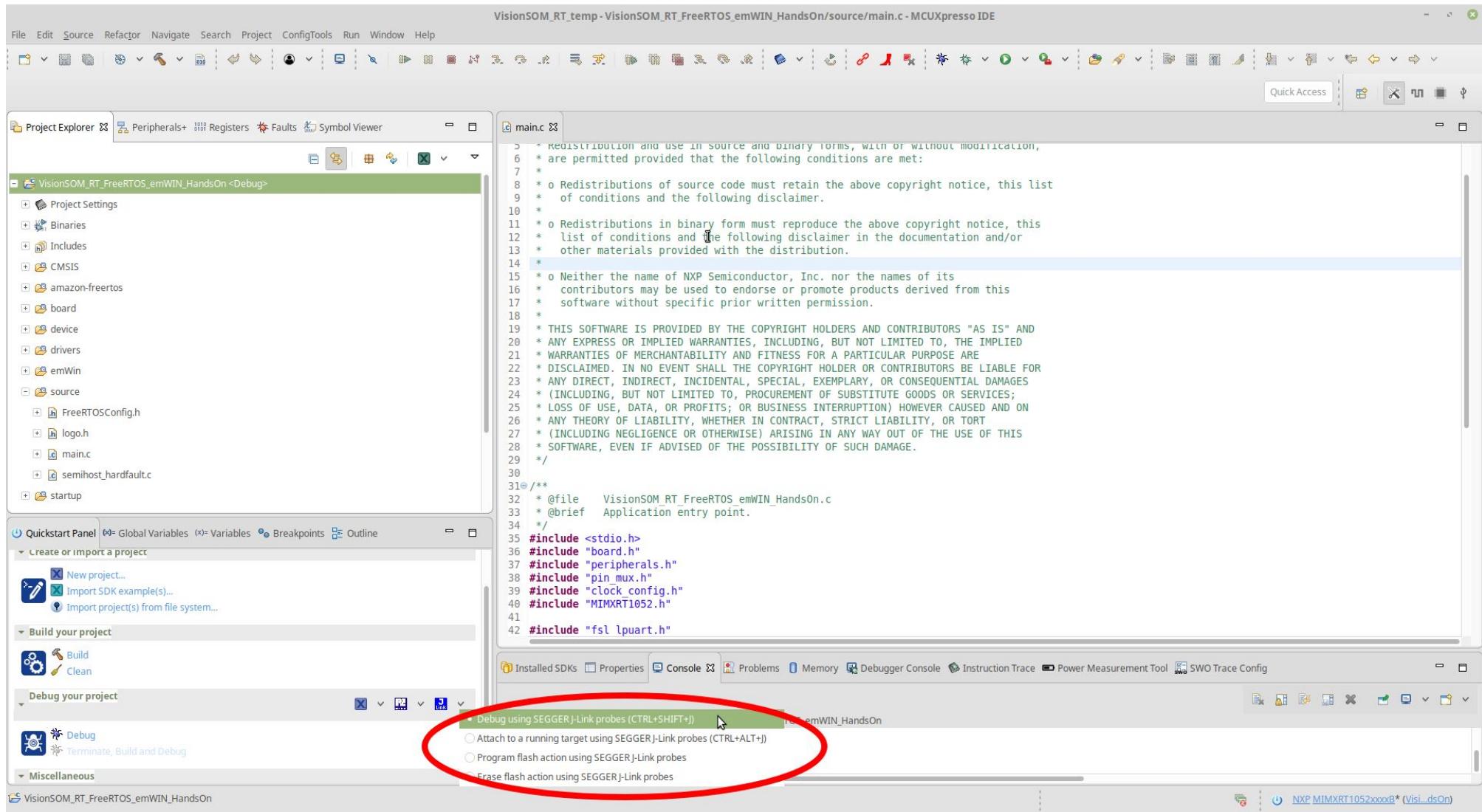
- Loader value

Devices/NXP/iMXRT105x/NXP_iMXRT105x_HyperFlash.elf
should be:

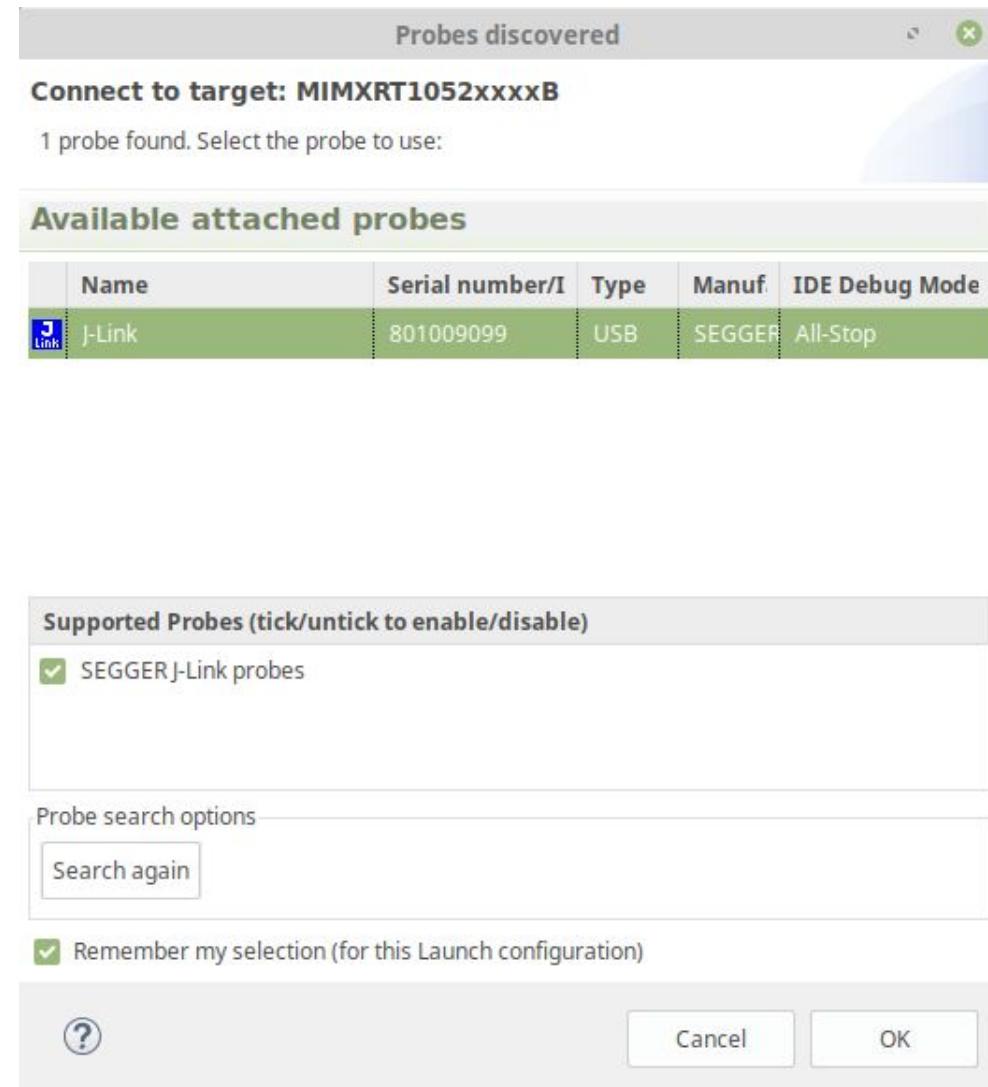
Devices/NXP/iMXRT105x/NXP_iMXRT105x_QSPI.elf

Proven Partner

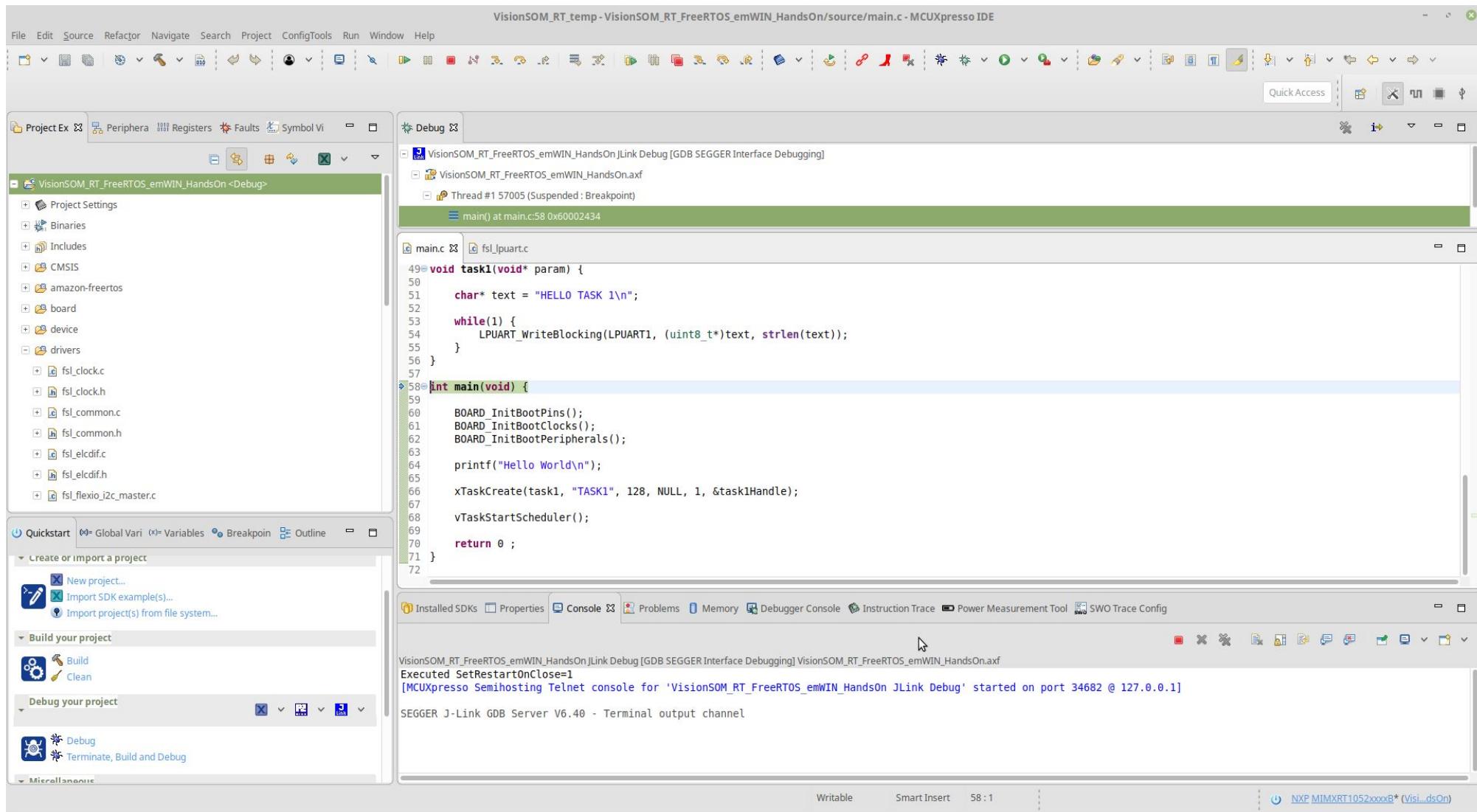
SEGGER J-Link



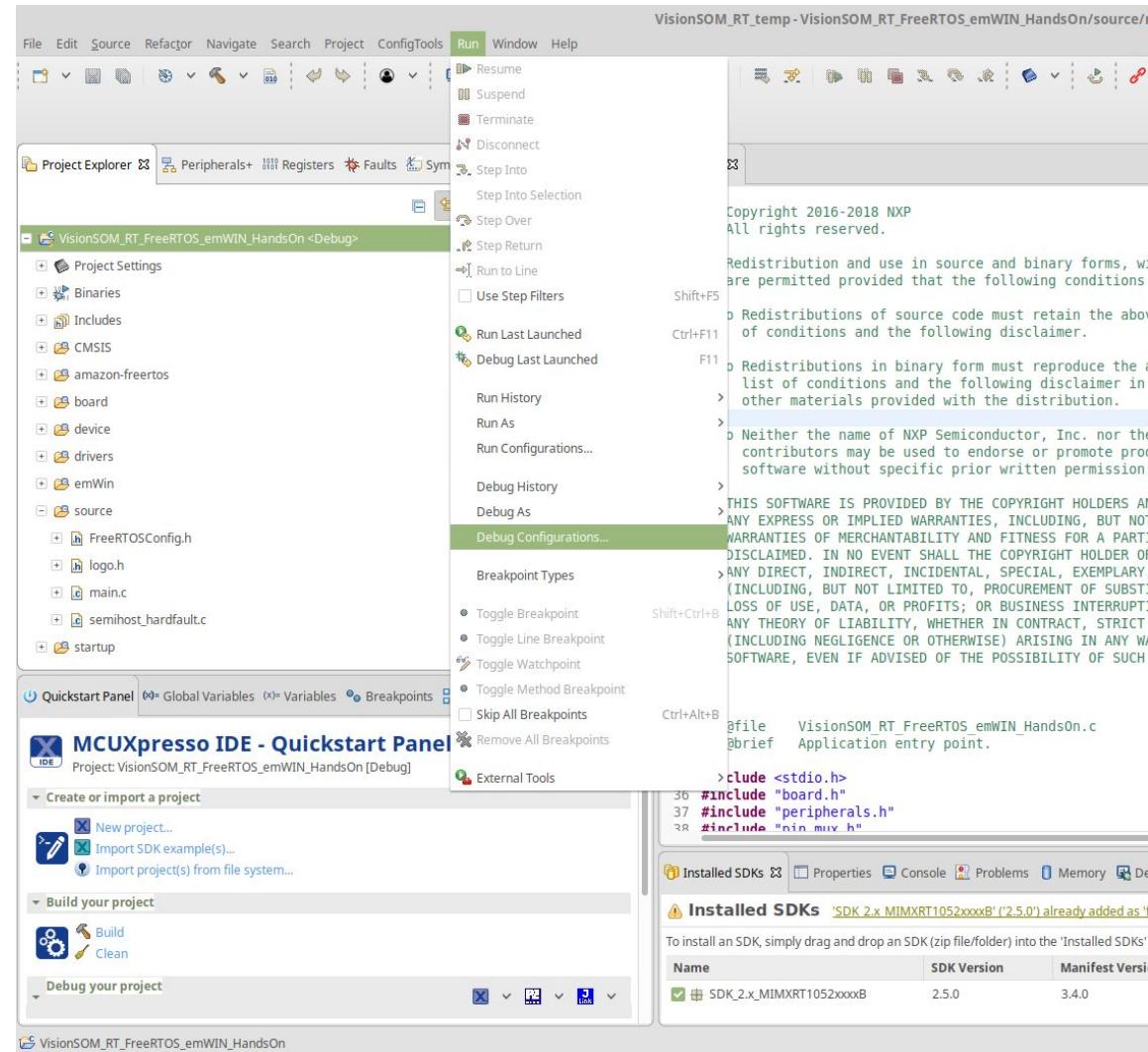
SEGGER J-Link



SEGGER J-Link



SEGGER J-Link



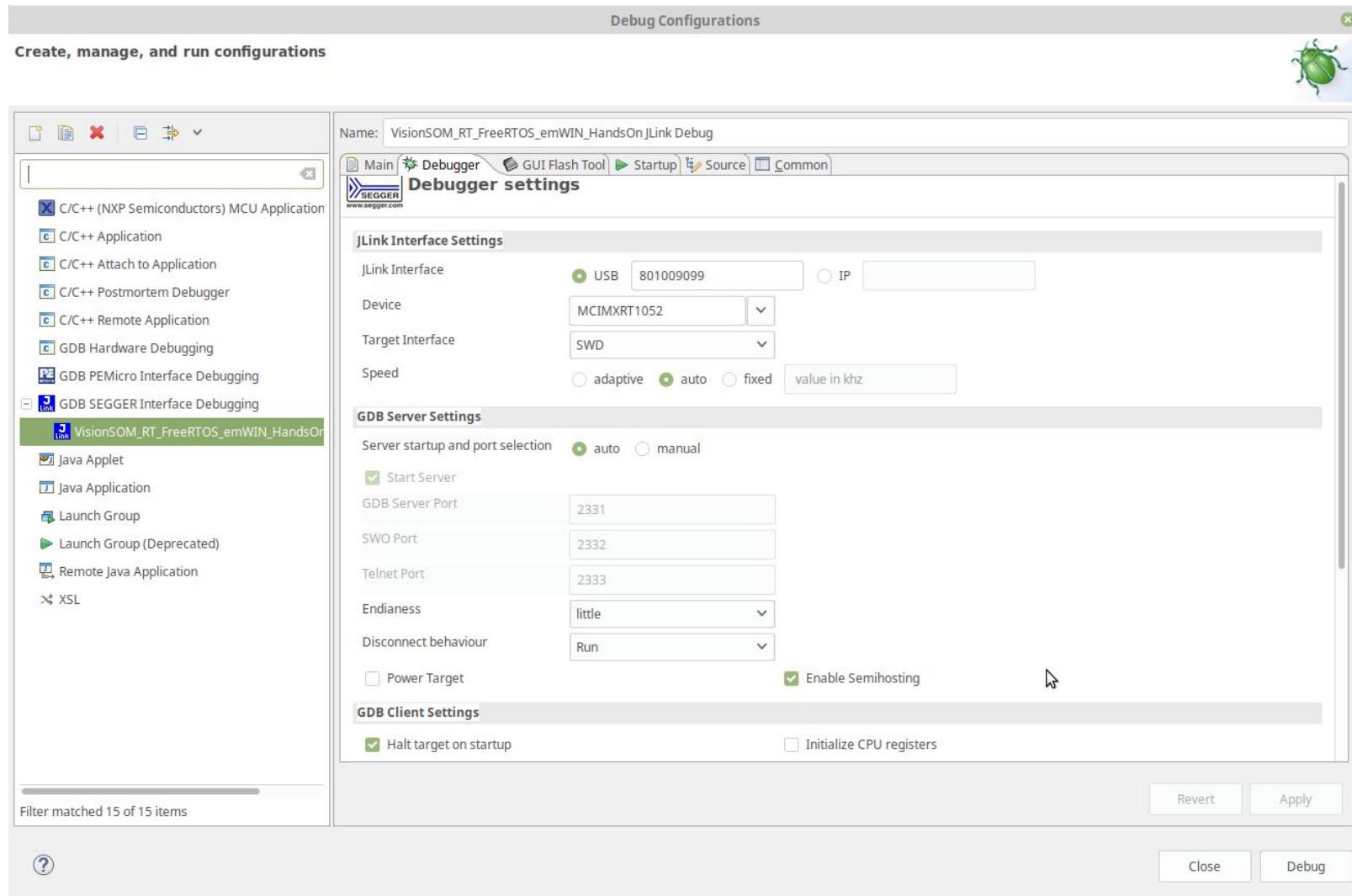
Proven Partner



SoMLabs | www.somlabs.com



SEGGER J-Link



SEGGER J-Link

The screenshot shows the MCUXpresso IDE interface with the following details:

- Project Explorer:** Shows the project "VisionSOM_RT_FreeRTOS_emWIN_HandsOn" with a suspended breakpoint at Thread #1 (57005) in main.c:54.
- Variables View:** Displays a variable "param" of type void* with the value "<optimized out>".
- Code Editor:** The main.c file is open, showing the following code:

```
#include "task.h"
...
void task1(void* param) {
    char* text = "HELLO TASK 1\n";
    while(1) {
        LPUART_WriteBlocking(LPUART1, (uint8_t*)text, strlen(text));
    }
}
int main(void) {
    BOARD_InitBootPins();
    BOARD_InitBootClocks();
    BOARD_InitBootPeripherals();
    printf("Hello World\n");
    xTaskCreate(task1, "TASK1", 128, NULL, 1, &task1Handle);
}
```
- Outline View:** Lists the project files and symbols: stdio.h, board.h, peripherals.h, pin_mux.h, clock_config.h, MIMXRT1052.h, fsl_lpuart.h, FreeRTOS.h, task.h, task1Handle: TaskHandle_t, task1(void*): void, and main(void): int.
- Task List (FreeRTOS) View:** Shows the task states:

TCB#	Task Name	Task Handle	Task State	Priority	Stack Usage	Event Object	Runtime
b 1	TASK1	0x200002b8	Running	1 (1)	0 B / 436 B		
b 2	IDLE	0x200004a8	Ready	0 (0)	0 B / 284 B		
b 3	Tmr Svc	0x200008f8	Suspended	4 (4)	0 B / 604 B	TmrQ (Rx)	

FreeRTOS

- RTOS (Real Time Operating System)
- RT - Ensures the compliance with time constraints
 - deterministic scheduler
 - fixed priorities
- OS - Kernel executing multiple tasks
 - independent tasks
 - synchronization and communication

Proven Partner

FreeRTOS - tasks

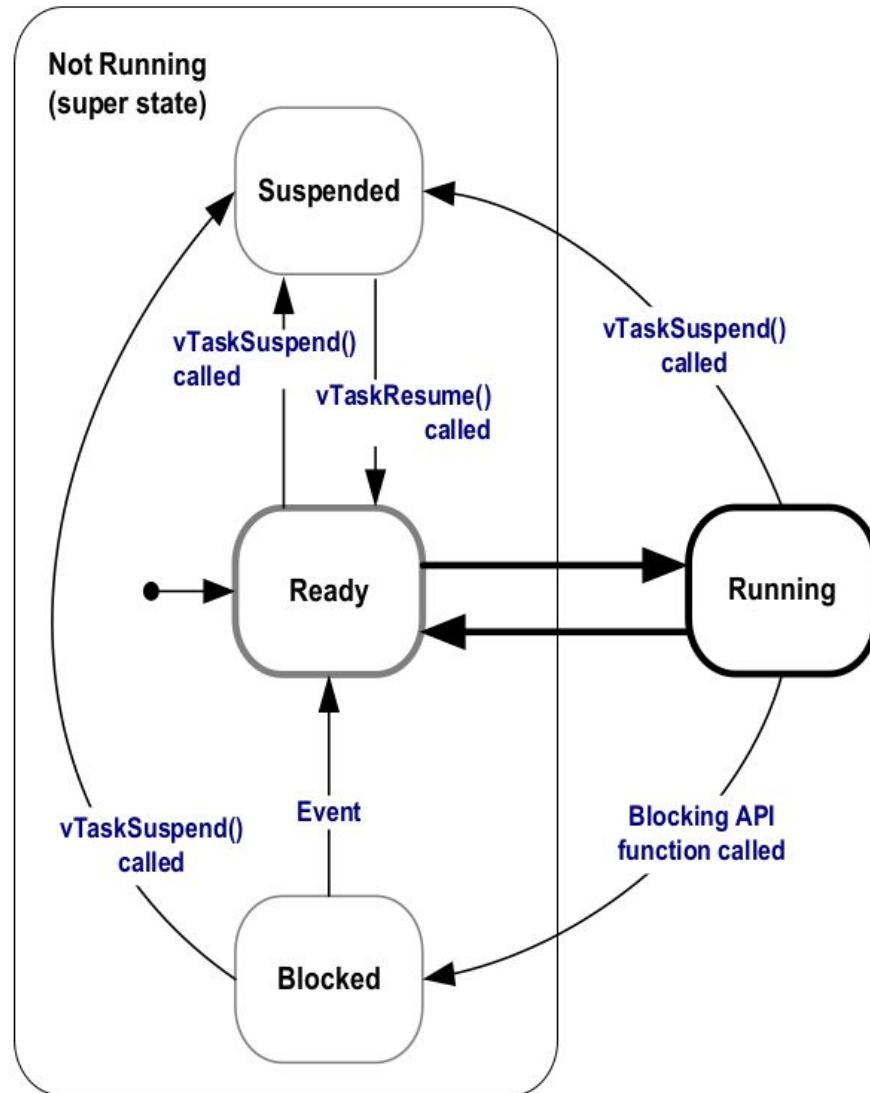
RTOS approach

```
int main(void) {  
    // create tasks  
    // run scheduler  
}  
  
void task1(void*) {  
    // initialization  
    while(1) {  
        // processing  
    }  
}  
void task2(void*) {  
    // initialization  
    while(1) {  
        // processing  
    }  
}
```

Classic approach

```
int main(void) {  
    // initialization  
  
    while(1) {  
        // processing  
    }  
}
```

FreeRTOS - tasks



FreeRTOS - scheduling

	configUSETIME_SLICING = 0	configUSETIME_SLICING = 1
configUSEPREEMPTION = 0	Co-operative Scheduling	
configUSEPREEMPTION = 1	Prioritized Pre-emptive Scheduling	Prioritized Pre-emptive Scheduling with Time Slicing

FreeRTOS - memory management

- static memory allocation
- dynamic allocation algorithms
 - heap_1 - simple allocation without freeing
 - heap_2 - allocation and freeing without blocks combining
 - heap_3 - malloc() and free()
 - heap_4 - allocation and freeing with blocks combining
 - heap_5 - allocation and freeing with blocks combining with multiple regions support

FreeRTOS - Idle task

- created automatically
- lowest priority
- responsible for freeing memory
- must not be starved!

Proven Partner



SomLabs | www.somlabs.com

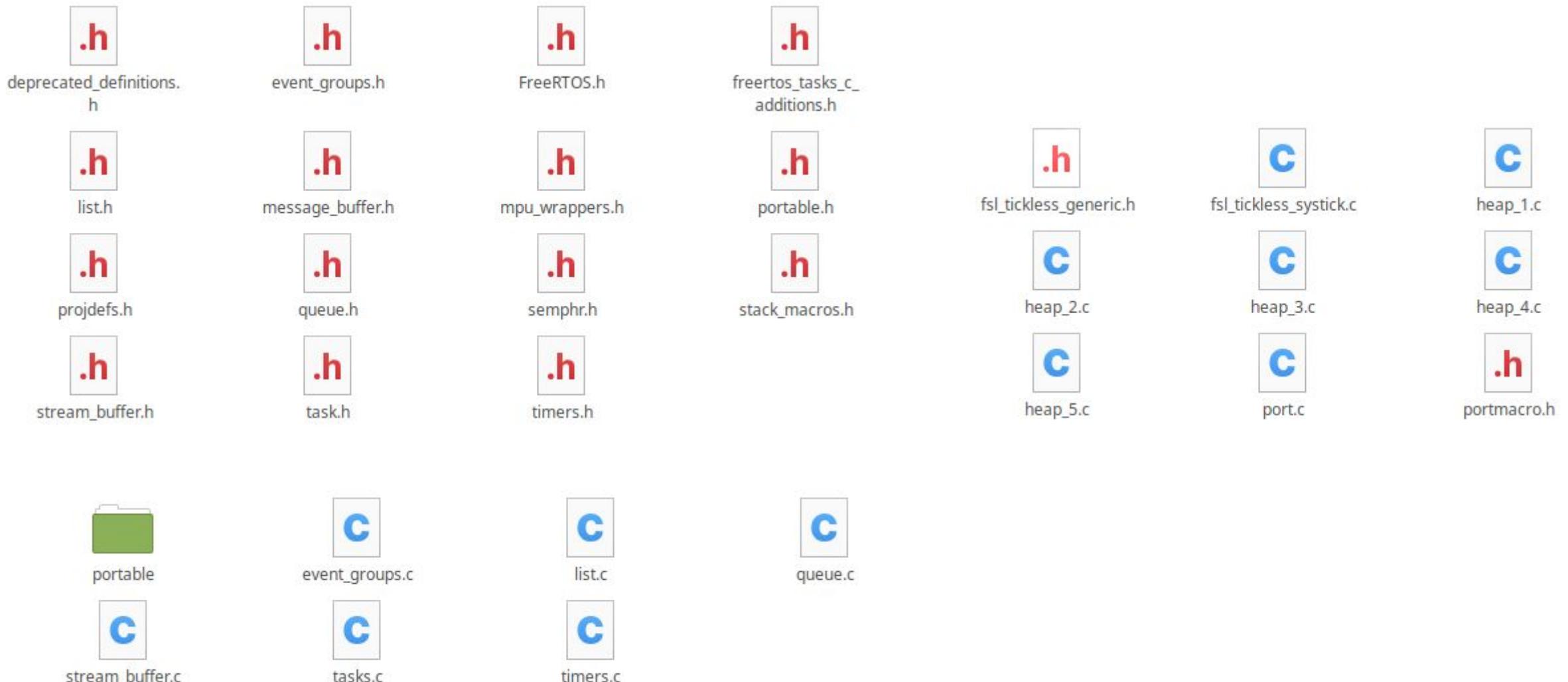


FreeRTOS - components

- Semaphores
- Mutexes
- Queues
- Notifications
- Software timers

Proven Partner

FreeRTOS - file structure



FreeRTOS - configuration

- FreeRTOSConfig.h

```
#define configUSE_PREEMPTION          1
#define configUSE_TICKLESS_IDLE        0
#define configCPU_CLOCK_HZ            (SystemCoreClock)
#define configTICK_RATE_HZ           ((TickType_t)200)
#define configMAX_PRIORITIES         5
#define configMINIMAL_STACK_SIZE     ((unsigned short)90)
#define configMAX_TASK_NAME_LEN      20
#define configUSE_16_BIT TICKS        0
#define configIDLE_SHOULD_YIELD      1
#define configUSE_TASK_NOTIFICATIONS 1
#define configUSE_MUTEXES             1
#define configUSE_RECURSIVE_MUTEXES   1
#define configUSE_COUNTING_SEMAPHORES 1
#define configUSE_ALTERNATIVE_API     0 /* Deprecated! */
#define configQUEUE_REGISTRY_SIZE     8
#define configUSE_QUEUE_SETS          0
#define configUSE_TIME_SLICING        1
#define configUSE_NEWLIB_REENTRANT    0
#define configENABLE_BACKWARD_COMPATIBILITY 1
#define configNUM_THREAD_LOCAL_STORAGE_POINTERS 5
#define configUSE_APPLICATION_TASK_TAG 0
```

FreeRTOS - memory overhead

- IAR STR71x ARM7 port
- Full optimisation
- Four priorities
- <https://www.freertos.org/FAQMem.html>

Scheduler	236 bytes
Queue	76 bytes + queue storage area
Task	64 bytes + the task stack size

+ 5-10 kB Flash



FreeRTOS - time overhead

- FreeRTOS ARM Cortex-M3 port for the Keil compiler
- Stack overflow checking turned off
- Trace features turned off
- Run-time stats feature turned off
- Compiler set to optimise for speed
- <https://www.freertos.org/FAQMem.html>

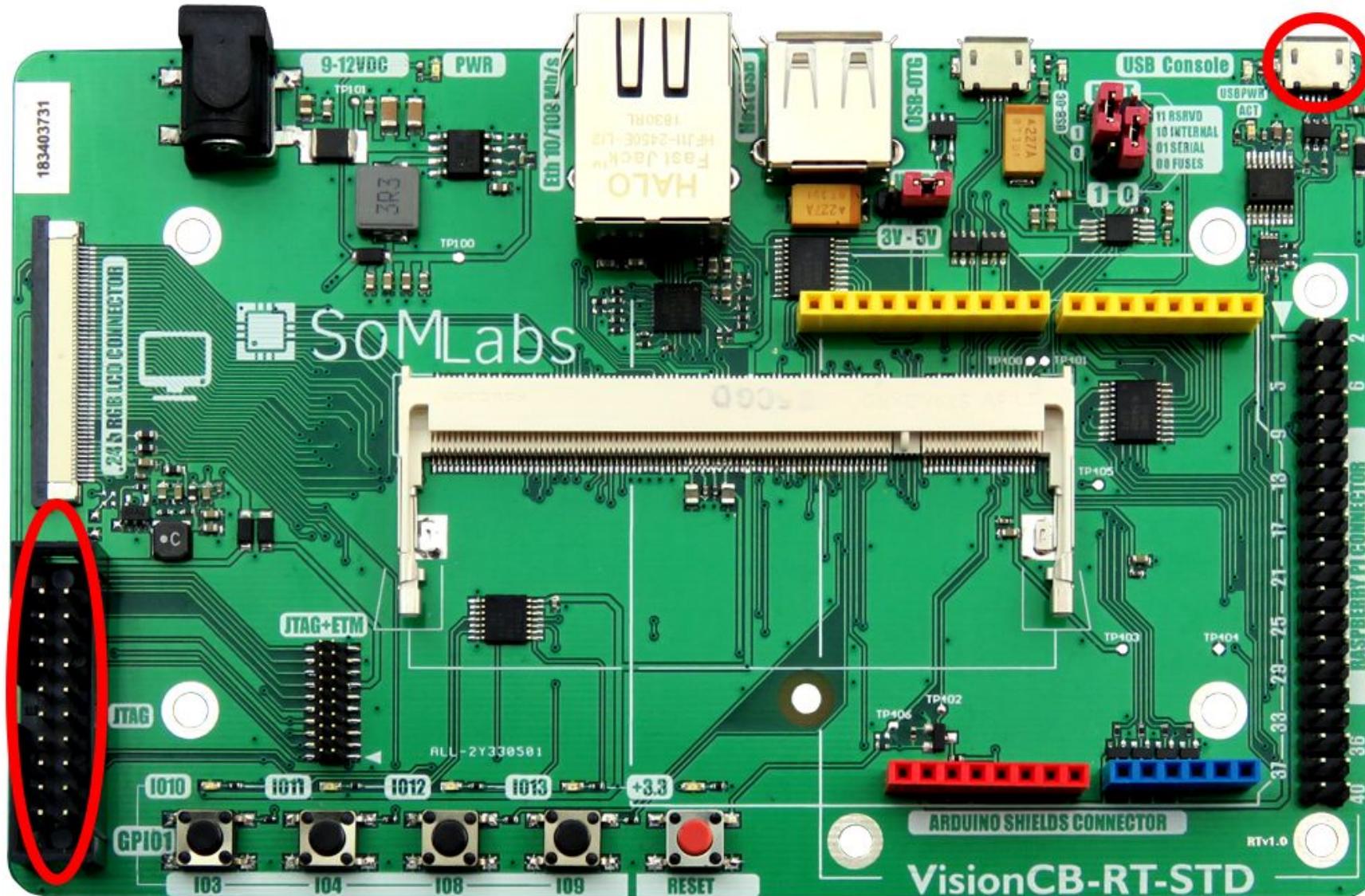
Context switching: 84 CPU cycles

Proven Partner

Exercise 1

- Create a new task
- Synchronize access to the common resource (UART port) using a mutex
- Use the same code for multiple tasks
- Synchronize access to the common resource (UART port) by changing the scheduling algorithm

Exercise 1



Exercise 1

- Creating a new task

```
BaseType_t xTaskCreate( TaskFunction_t pvTaskCode,  
                        const char * const pcName,  
                        unsigned short usStackDepth,  
                        void *pvParameters,  
                        UBaseType_t uxPriority,  
                        TaskHandle_t *pxCreatedTask  
) ;  
  
void vTaskCode( void * pvParameters ) ;
```

Exercise 1

- Using the mutex

```
SemaphoreHandle_t xSemaphoreCreateMutex( void );
```

```
BaseType_t xSemaphoreTake( SemaphoreHandle_t xSemaphore,  
                           TickType_t xTicksToWait );
```

```
BaseType_t xSemaphoreGive( SemaphoreHandle_t xSemaphore );
```

Exercise 1

- Changing scheduling algorithm (FreeRTOSConfig.h)

configUSE_PREEMPTION

configUSE_TIME_SLICING

FreeRTOS - interrupts

- No requirements regarding the interrupts handling
- Functions dedicated for the interrupts - **FromISR*
- Task-interrupt cooperation
 - Synchronization
 - Processing deferring

Exercise 2

- Send a notification from the button interrupt
- Add a higher priority ‘busy task’
- Observe the button-LED responsiveness
- Switch the priorities
- Change the scheduling algorithm

Exercise 2

- Using notifications

```
BaseType_t xTaskNotifyFromISR(
    TaskHandle_t xTaskToNotify,
    uint32_t ulValue,
    eNotifyAction eAction,
    BaseType_t *pxHigherPriorityTaskWoken );

BaseType_t xTaskNotifyWait(
    uint32_t ulBitsToClearOnEntry,
    uint32_t ulBitsToClearOnExit,
    uint32_t *pulNotificationValue,
    TickType_t xTicksToWait );
```

Exercise 2

- Busy task

```
void busyTask(void* param) {  
  
    const TickType_t delayMs = 10 / portTICK_PERIOD_MS;  
    char* text = "Calculation finished\n";  
  
    while(1) {  
        for(int i = 0; i < 0xFFFFFFFF; i++)  
            ;  
  
        LPUART_WriteBlocking(LPUART1, (uint8_t*)text, strlen(text));  
        vTaskDelay(delayMs);  
    }  
  
}  
  
xTaskCreate(busyTask, "BUSY_TASK", 128, busyTask, 2, &busyTaskHandle);
```

Exercise 2

- Switching priorities

```
xTaskCreate(ledTask, "LED_TASK", 128, ledTask, 2, &ledTaskHandle);  
xTaskCreate(busyTask, "BUSY_TASK", 128, busyTask, 1, &busyTaskHandle);
```

- Changing scheduling algorithm

```
#define configUSE_PREEMPTION    0  
#define configUSE_TIME_SLICING  1
```

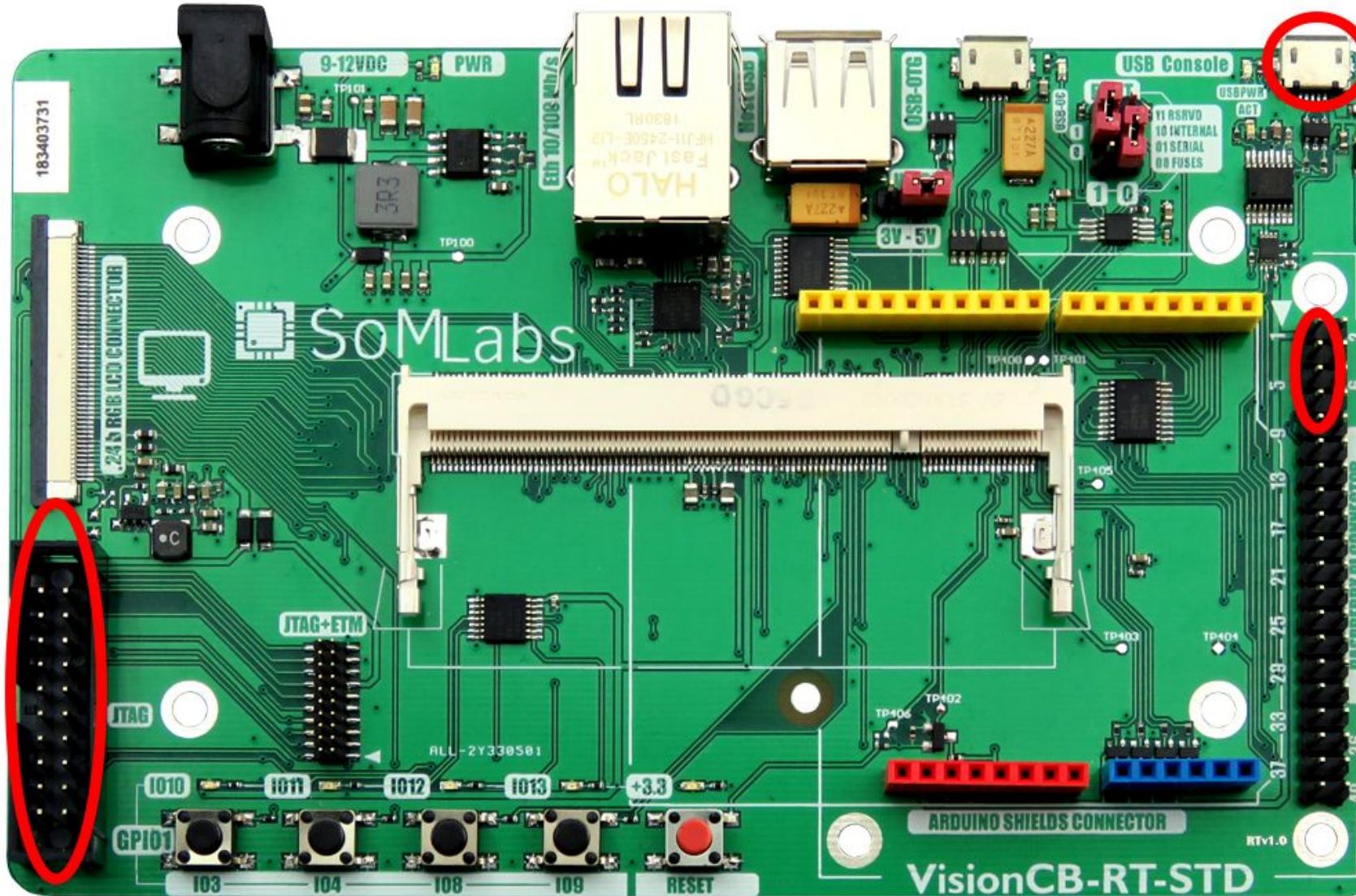
FreeRTOS - queues

- FIFO queues used for sending data
 - task - task
 - task - interrupt
- Data type defined by the programmer:
 - simple type
 - structure
 - pointer

Exercise 3

- Read temperature and pressure from MPL3115A2
- Create a queue for sending:
 - simple types
 - structures
 - pointersbetween two tasks.

Exercise 3



PIN	
1	VCC
3	SDA
5	SCL
9	GND



Exercise 3

- Using a queue

```
QueueHandle_t xQueueCreate(  UBaseType_t uxQueueLength,  
                            UBaseType_t uxItemSize  );  
  
 BaseType_t xQueueSend( QueueHandle_t xQueue,  
                        const void * pvItemToQueue,  
                        TickType_t xTicksToWait  );  
  
 BaseType_t xQueueReceive( QueueHandle_t xQueue,  
                           void *pvBuffer,  
                           TickType_t xTicksToWait  );
```

emWin

- GUI library based on windows and widgets
 - Touch panel support
 - Easy to use with any RTOS or bare-metal application
 - Examples for multiple microcontrollers
- <https://www.segger.com/evaluate-our-software/>

Proven Partner



SoMLabs | www.somlabs.com



emWin - configuration

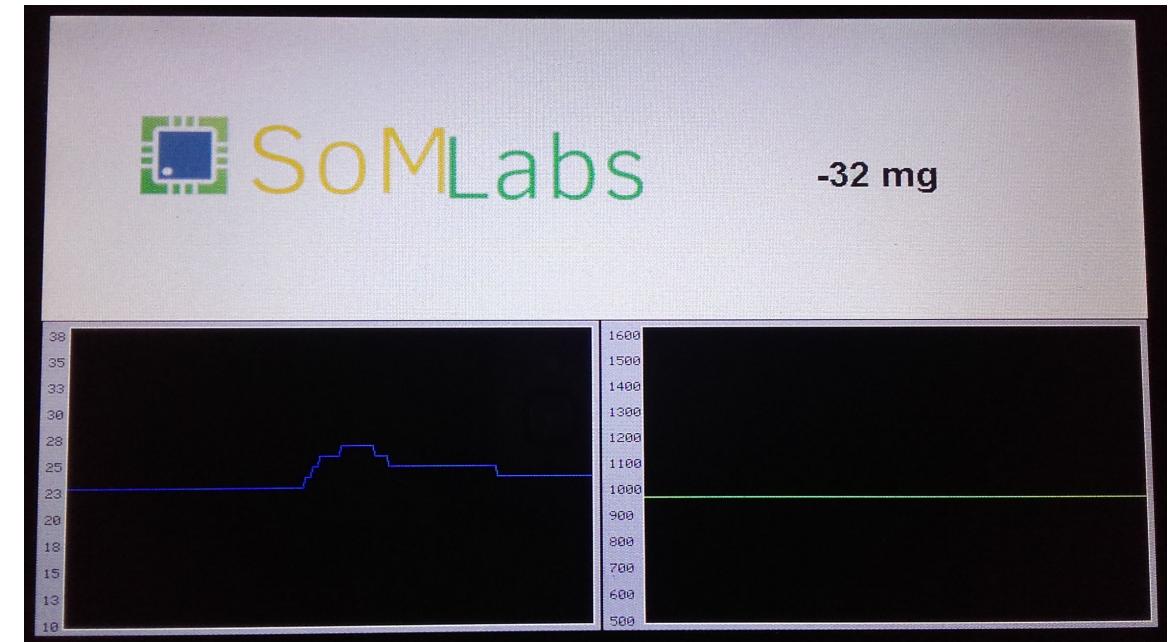
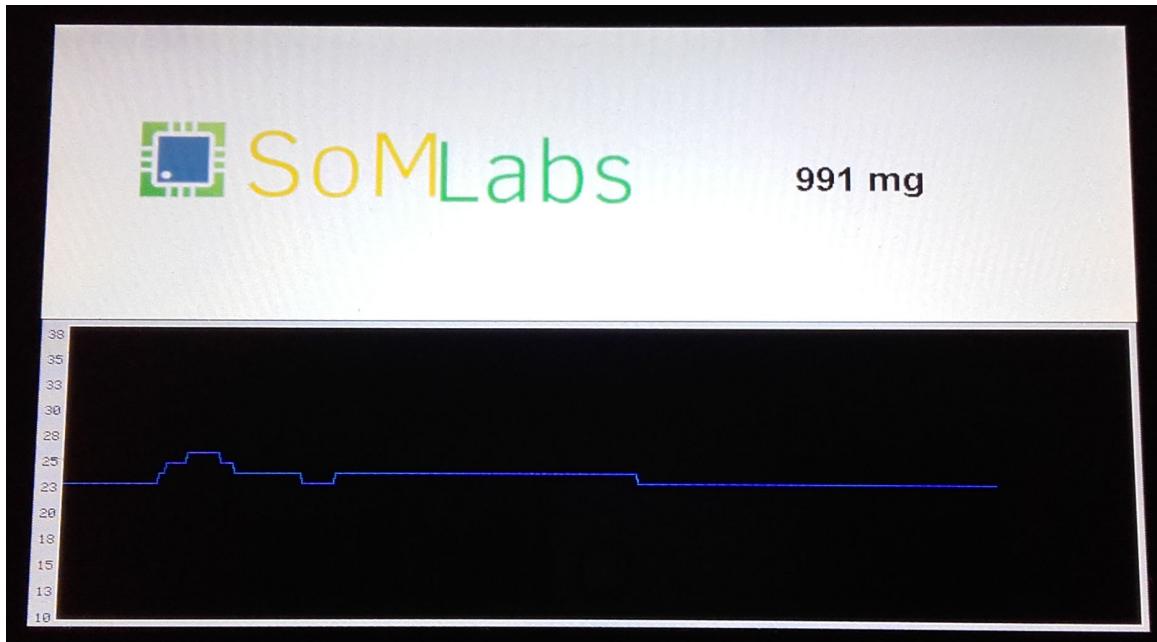
Configuration files:

- GUIConf.h
- LCDConf.h
- emwin_support.c
- emwin_support.h

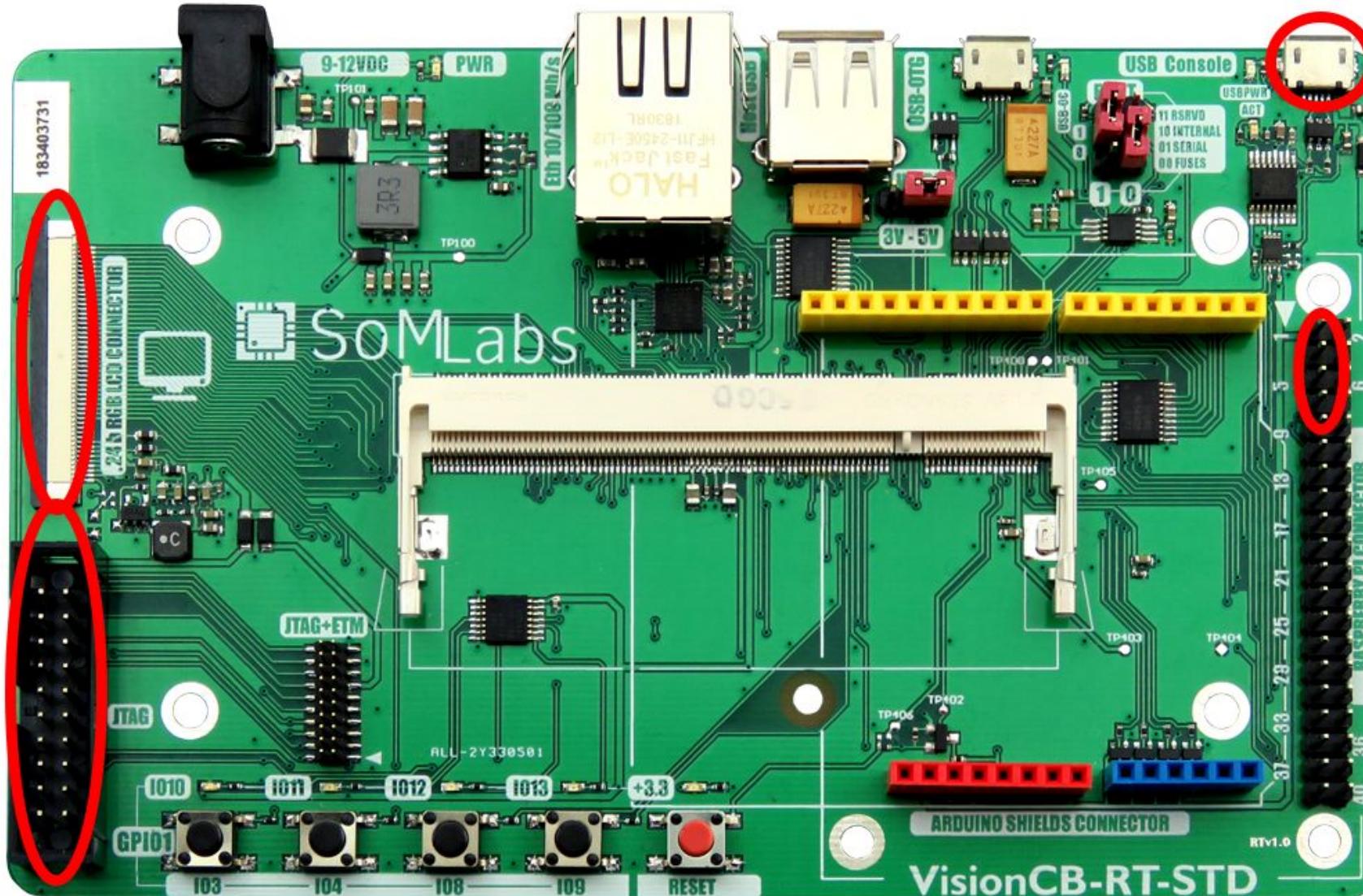
```
#define GUI_MEMORY_ADDR ((uint32_t)s_gui_memory)
#define VRAM_ADDR    ((uint32_t)s_vram_buffer)
```

Exercise 4

- Analyze the GUI building code in the *guiTask*
- Create another GRAPH component for pressure data



Exercise 4



PIN	
1	VCC
3	SDA
5	SCL
9	GND



Exercise 4

- Creating the GRAPH widget

```
GRAPH_Handle GRAPH_CreateEx( int x0, int y0,  
                                int xSize, int ySize,  
                                WM_HWIN hParent,  
                                int WinFlags,  
                                int ExFlags, int Id);
```

emWin - touch panel

- PID (Pointer Input Device) and MultiTouch support
- Providing touch events to the library
- Handling the input events
- Windows notifications

Proven Partner

Exercise 5

- Analyze handling the touch events in the software timer code
- Create buttons
- Use the buttons for acceleration axis selection
- Handle the input events



Exercise 5

- Creating buttons

```
BUTTON_Handle BUTTON_Create( int x0, int y0,  
                             int xSize, int ySize,  
                             WM_HWIN hParent,  
                             int WinFlags,  
                             int ExFlags,  
                             int Id);
```

Exercise 5

- Handling events

```
void eventCallback(WM_MESSAGE * pMsg) {
    if(pMsg->MsgId == WM_NOTIFY_PARENT) {
        if (pMsg->hWinSrc == buttonX) {
            int code = pMsg->Data.v;
            if (code == WM_NOTIFICATION_RELEASED) {
                activeAxis = ACTIVE_AXIS_X;
            }
        }
    }
    return;
}
```

Why to use RTOS?

- Easy timing management
- Application modularity
- Team development
- Schedulability design and analysis
 - MAST (mast.unican.es)
 - SimSo (projects.laas.fr/simso)

Proven Partner

Why to use FreeRTOS?

- One of the most popular RTOS
- Very well documented
- Open-source
- FreeRTOS+ Ecosystem
- OpenRTOS an SafeRTOS

Proven Partner



- Connecting the Future
- Customer Centric