

How to Use ARM® Keil® RTX with MSP432 MCUs



This blog is about the RTX, ARM's free and reliable RTOS. RTX now has a BSD license and this makes it free. ARM provides all RTX source code and maintains it. While initially focused on Cortex®-M processors, RTX has been ported to Cortex-R and at least one Cortex-A9. It runs perfectly on TI's new [MSP432 MCU](#) platform, based on the ARM Cortex-M4 core.

A tutorial with RTX and DSP examples for the [MSP432 MCU LaunchPad](#) is now [available online](#).

RTOS is a full-featured RTOS that consumes minimal resources on your target and is easy to install, configure and modify. You can use RTX with any tool chain: not just Keil MDK and this includes GCC. RTX is completely CMSIS-RTOS compliant. This makes it easy to use and stable.

It took me about 5 minutes to get RTX configured and running with a minimum main.c project on the MSP432 LaunchPad using the general steps outlined in the MDK Getting Started manual. It is available here: www.keil.com/mdk5/. General information about RTX is here: www.keil.com/rl-arm/kernel.asp

RTOS uses a minimum of system resources and never turns off IRQ interrupts. RTX uses the Cortex SysTick timer with its Interrupt 15 and also has a tickless mode for low power operation. RTX uses less than 4 Kbytes for the kernel. Here are the memory requirements:

RTOS Memory Requirements	
Task Specifications	Performance
CODE Size	< 4.0 KBytes
RAM Space for Kernel	< 300 Bytes + 128 Bytes User Stack
RAM Space for a Task	TaskStackSize + 52 Bytes
RAM Space for a Mailbox	MaxMessages*4 + 16 Bytes
RAM Space for a Semaphore	8 Bytes
RAM Space for a Mutex	12 Bytes
RAM Space for a User Timer	8 Bytes
Hardware Requirements	SysTick timer

There are compelling advantages of using an RTOS in all but the simplest of designs. See http://www.keil.com/rl-arm/rtx_rtosadv.asp for more details.

The best way to get the latest RTX is to [download and install](#) MDK. MDK is free and you do not need any license. You can then obtain the RTX libraries with source code here: C:\Keil_v5\ARM\Pack\ARM\CMSIS\4.3.0\CMSIS\RTOS. You can use the kernel awareness windows provided with μ Vision® to experiment with RTX and view how your changes affect its operation.

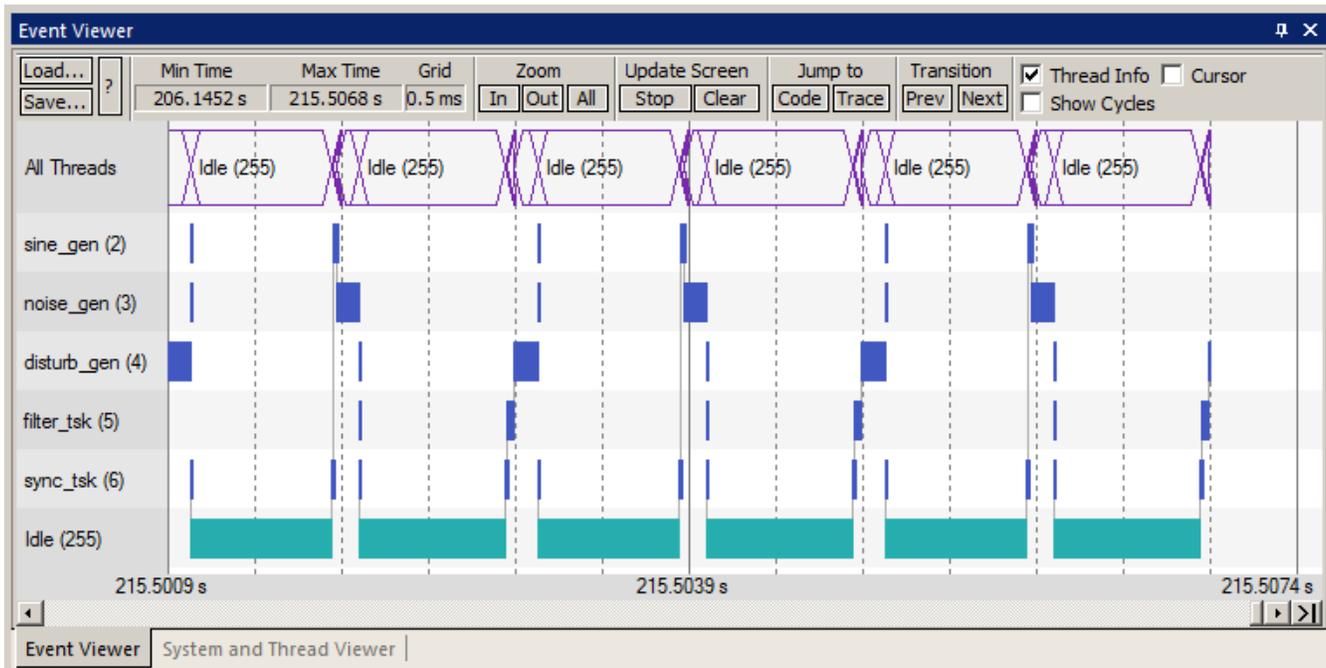
Another method is by downloading the ARM CMSIS Pack [CMSIS \(Cortex Microcontroller Software Interface Standard\)](#). This file has a .pack extension which is really a standard .zip file. The RTX files with source code are inside this zip file.

Ports are included for Keil, GCC and IAR toolchains.

This is the **System and Thread Viewer** displaying information real-time about threads: You do not need to stop the program to view this window.

System and Thread Viewer																																																																	
Property	Value																																																																
System	<table border="1"> <thead> <tr> <th>Item</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Tick Timer:</td> <td>1.000 mSec</td> </tr> <tr> <td>Round Robin Timeout:</td> <td>5.000 mSec</td> </tr> <tr> <td>Default Thread Stack Size:</td> <td>200</td> </tr> <tr> <td>Thread Stack Overflow Check:</td> <td>Yes</td> </tr> <tr> <td>Thread Usage:</td> <td>Available: 6, Used: 7 + os_idle_demon</td> </tr> </tbody> </table>	Item	Value	Tick Timer:	1.000 mSec	Round Robin Timeout:	5.000 mSec	Default Thread Stack Size:	200	Thread Stack Overflow Check:	Yes	Thread Usage:	Available: 6, Used: 7 + os_idle_demon																																																				
Item	Value																																																																
Tick Timer:	1.000 mSec																																																																
Round Robin Timeout:	5.000 mSec																																																																
Default Thread Stack Size:	200																																																																
Thread Stack Overflow Check:	Yes																																																																
Thread Usage:	Available: 6, Used: 7 + os_idle_demon																																																																
Threads	<table border="1"> <thead> <tr> <th>ID</th> <th>Name</th> <th>Priority</th> <th>State</th> <th>Delay</th> <th>Event Value</th> <th>Event Mask</th> <th>Stack Usage</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>main</td> <td>Normal</td> <td>Wait_DLY</td> <td></td> <td></td> <td></td> <td>32%</td> </tr> <tr> <td>2</td> <td>phaseA</td> <td>Normal</td> <td>Wait_AND</td> <td></td> <td></td> <td></td> <td>40%</td> </tr> <tr> <td>3</td> <td>phaseB</td> <td>Normal</td> <td>Wait_AND</td> <td></td> <td>0x0000</td> <td>0x0001</td> <td>40%</td> </tr> <tr> <td>4</td> <td>phaseC</td> <td>Normal</td> <td>Wait_DLY</td> <td>366</td> <td>0x0000</td> <td>0x0001</td> <td>44%</td> </tr> <tr> <td>5</td> <td>phaseD</td> <td>Normal</td> <td>Wait_AND</td> <td></td> <td>0x0000</td> <td>0x0001</td> <td>40%</td> </tr> <tr> <td>6</td> <td>clock</td> <td>Normal</td> <td>Wait_AND</td> <td></td> <td>0x0000</td> <td>0x0100</td> <td>40%</td> </tr> <tr> <td>255</td> <td>os_idle_demon</td> <td>None</td> <td>Running</td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	ID	Name	Priority	State	Delay	Event Value	Event Mask	Stack Usage	1	main	Normal	Wait_DLY				32%	2	phaseA	Normal	Wait_AND				40%	3	phaseB	Normal	Wait_AND		0x0000	0x0001	40%	4	phaseC	Normal	Wait_DLY	366	0x0000	0x0001	44%	5	phaseD	Normal	Wait_AND		0x0000	0x0001	40%	6	clock	Normal	Wait_AND		0x0000	0x0100	40%	255	os_idle_demon	None	Running				
ID	Name	Priority	State	Delay	Event Value	Event Mask	Stack Usage																																																										
1	main	Normal	Wait_DLY				32%																																																										
2	phaseA	Normal	Wait_AND				40%																																																										
3	phaseB	Normal	Wait_AND		0x0000	0x0001	40%																																																										
4	phaseC	Normal	Wait_DLY	366	0x0000	0x0001	44%																																																										
5	phaseD	Normal	Wait_AND		0x0000	0x0001	40%																																																										
6	clock	Normal	Wait_AND		0x0000	0x0100	40%																																																										
255	os_idle_demon	None	Running																																																														

This is the **Event Viewer** and it also updates in real-time while your program is running: It displays in a graphical format when each thread is running with timestamps. Note the idle daemon is running most of the time. You can change these timing parameters easily and see the results immediately in the Event Viewer. You can measure timings with the cursors. Serial Wire Viewer (SWV) support is needed for the Event Viewer. This is supplied with a Keil ULINK[®]2, ULINK_{pro} or a Segger J-Link.



I created a hands-on tutorial using Keil MDK with the MSP432 MCU and it includes references to RTX. There are several examples and two of them use RTX. Blinky is a small bare metal program and RTX_Blinky has RTX added to it. The DSP example uses RTX to allocate processing time between four threads. There is a section on

adding RTX to a project and then adding a thread. This action shows up in the kernel awareness windows. This lab is located here: www.keil.com/appnotes/docs/apnt_276.asp

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES "AS IS" AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI's products are provided subject to [TI's Terms of Sale](#) or other applicable terms available either on [ti.com](https://www.ti.com) or provided in conjunction with such TI products. TI's provision of these resources does not expand or otherwise alter TI's applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2023, Texas Instruments Incorporated