Renesas RA Family

Arm® DSP Examples

Introduction

This application note describes the use of Arm® CMSIS-DSP example projects that are ported to Renesas Arm® Cortex®-M33 core based RA6M4 MCU with the Digital Signal Processing (DSP) extension and Floating Point Unit (FPU). This application note will discuss the steps to import, configure, build, and execute these DSP examples and measure their performance.

All information regarding to Arm CMSIS-DSP can be found at the below link:

https://arm-software.github.io/CMSIS_5/DSP/html/group__groupExamples.html

Target Device

This application note focuses on RA6M4 MCU. However, it also applies to any Arm Cortex-M33 core based Renesas MCU. This includes RA6M4, RA6M5, RA4M2, and RA4M3.

Contents

1. Arm CMSIS-DSP Library in Renesas Flexible Software Package (FSP)

The CMSIS-DSP software library is a suite of common signal processing functions for use on Cortex-M and Cortex-A processor based devices.

The library is divided into several functions, each covering a specific category:

- Basic math functions
- Fast math functions
- Complex math functions
- Filtering functions
- Matrix functions
- Transform functions
- Motor control functions
- Statistical functions
- Support functions
- Interpolation functions
- Support Vector Machine functions (SVM)
- Bayes classifier functions
- Distance functions

The library has generally separate functions for operating on 8-bit integers, 16-bit integers, 32-bit integer, and 32-bit floating-point values. It is ported to FSP and can be easily included into your project by using **New Stack** in the **Stacks** tab in the FSP configurator, as shown in Figure 1.

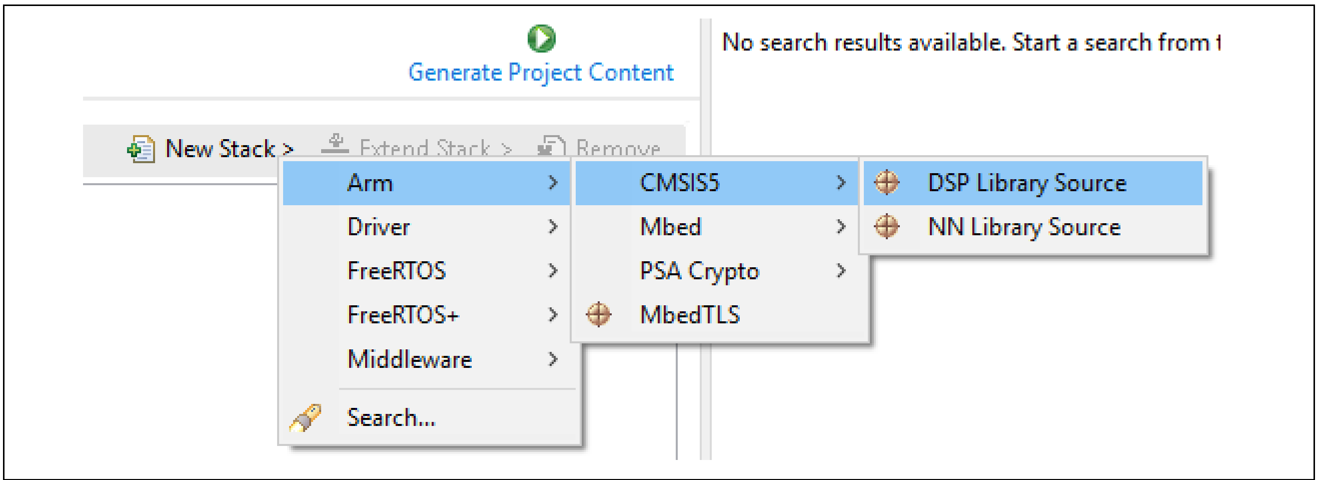


Figure 1. Adding CMSIS-DSP Library Source in FSP Configurator Using New Stack Feature

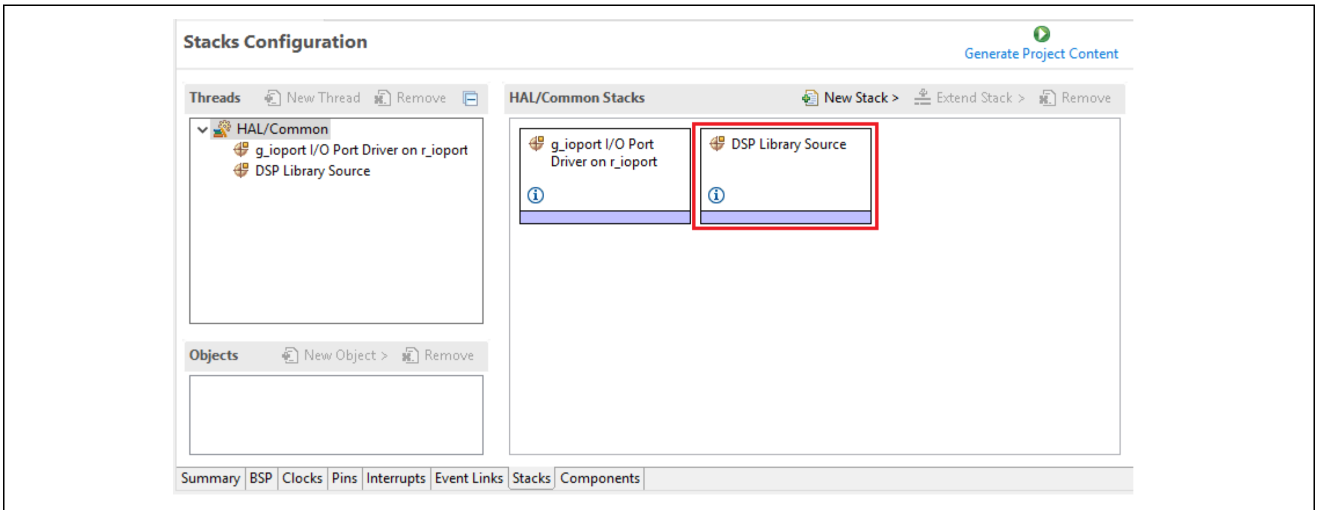


Figure 2. Adding CMSIS-DSP Source Library in FSP Configurator

Pressing **Generate Project Content** will generate the CMSIS-DSP library in your project.

Header and source files are created as shown in Figure 2.

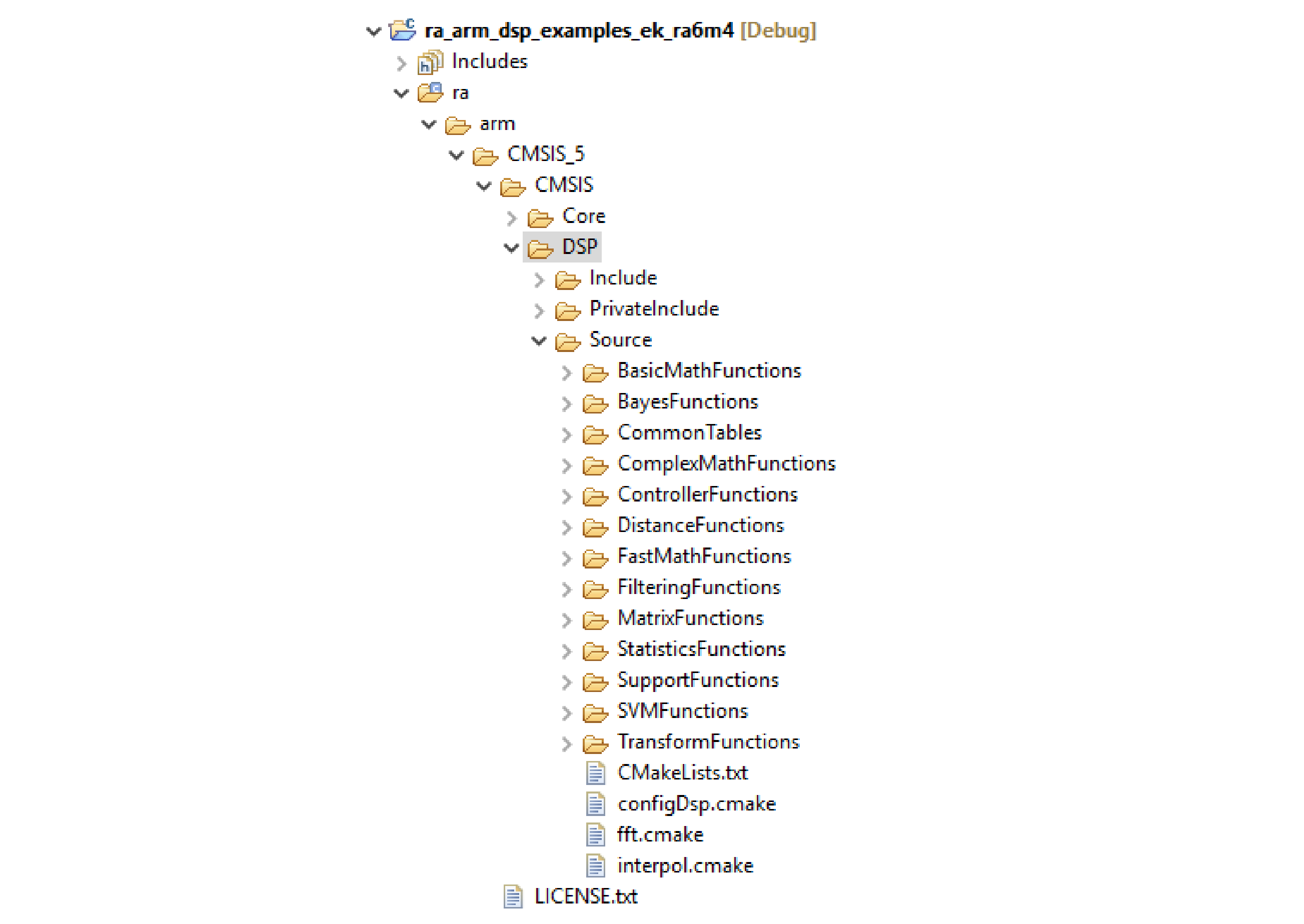


Figure 3. CMSIS-DSP Library Source in Example Project

2. Arm CMSIS-DSP Examples

Original Arm CMSIS-DSP examples are ported to the e^2 studio/FSP environment in a single project form. This example project includes multiple CMSIS-DSP examples. Below is the list of examples supported by this examples project.

- SineCosine example
- Signal Convergence example
- Linear Interpolate example
- Graphic Audio Equalizer example
- Lowpass Filter example
- Frequency Bin example
- Dot Product example
- Convolution example
- Variance example
- Matrix example
- Class Marks example
- Bayes example
- SVM example

Details of the above examples can be found here at the following link:

https://arm-software.github.io/CMSIS_5/DSP/html/group__groupExamples.html

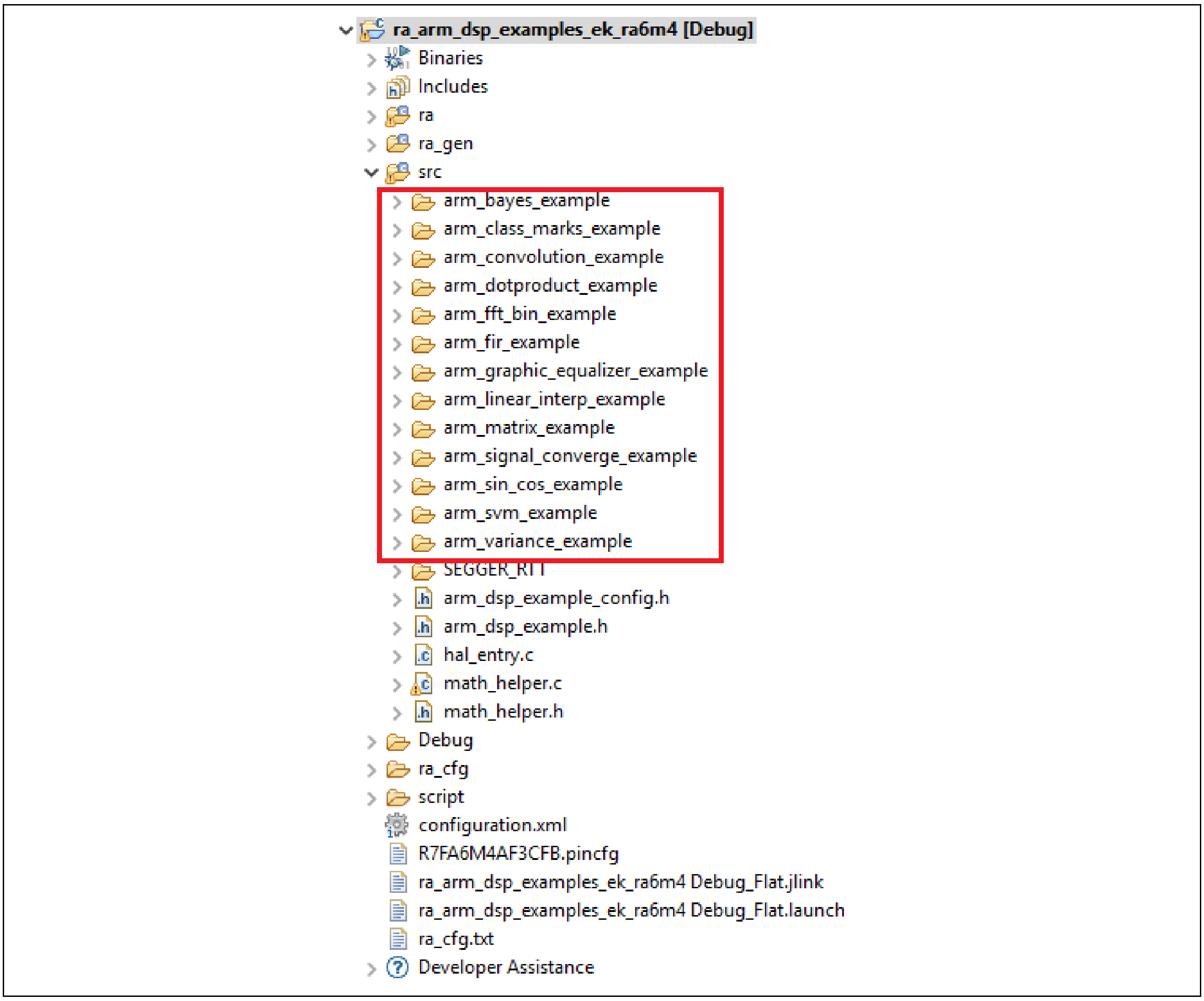


Figure 4. Example Project Imported to e² studio Including Multiple CMSIS-DSP Examples

Each CMSIS-DSP example is enabled using its macro definition in `arm_dsp_example_config.h`.

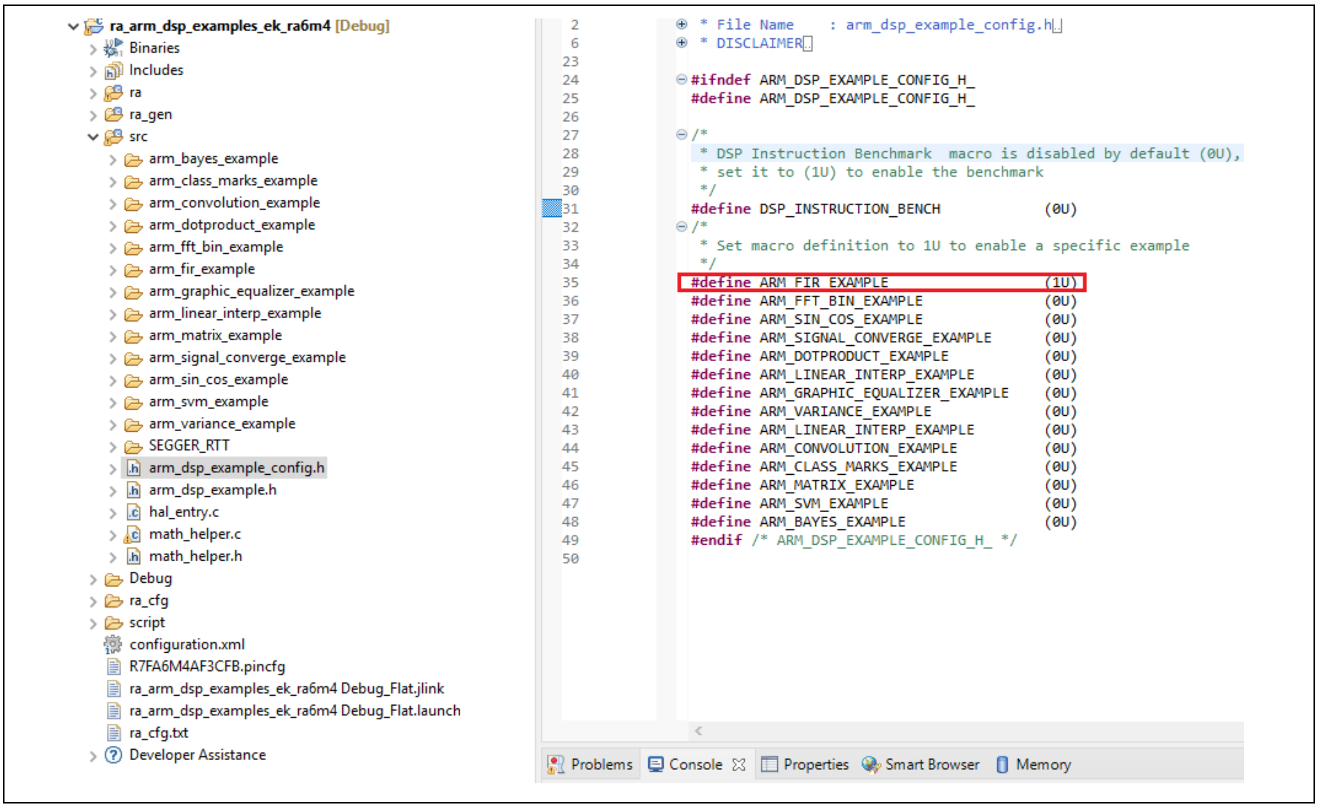
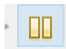


Figure 5. arm_dsp_example_config.h

2.1 Build and Run CMSIS_DSP Example

After importing the example project, enable the example project that you wish to run in `arm_dsp_example_config.h`.

You now can download and run the project. The following screen shots show an example of running the Lowpass Filter example.

After downloading and running the project, wait for a few second and hit the **Suspend** button  to stop project execution.

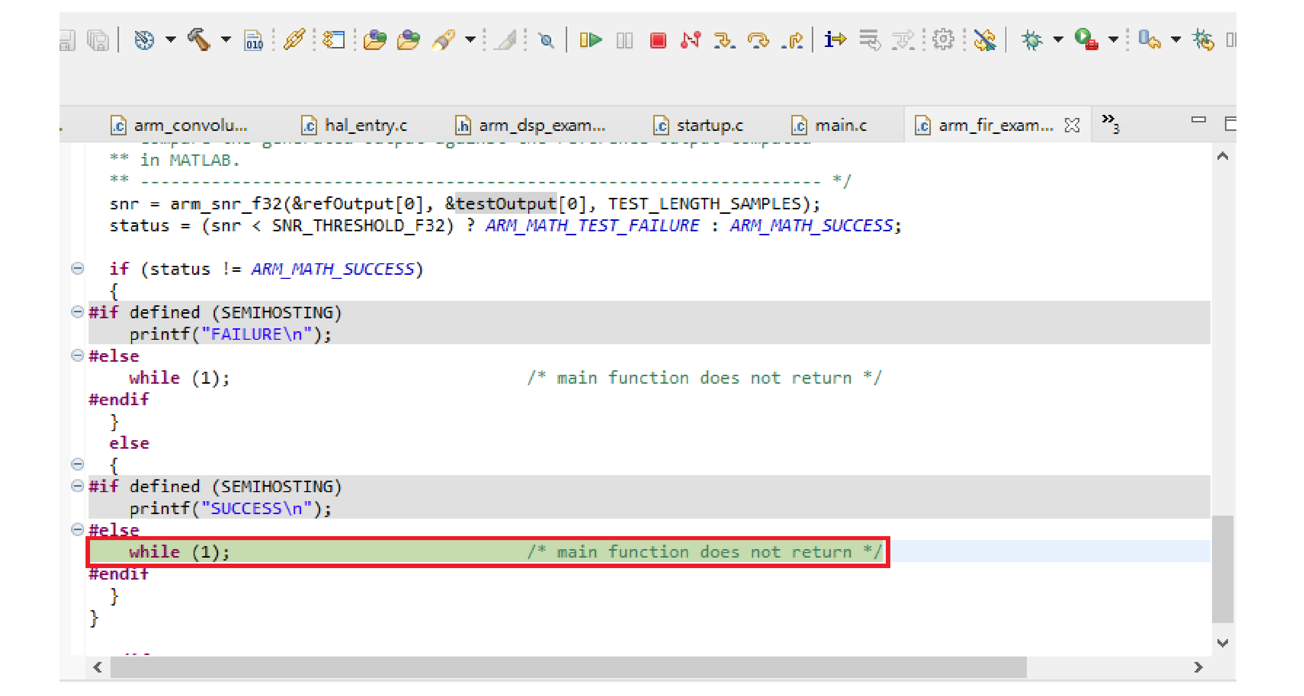
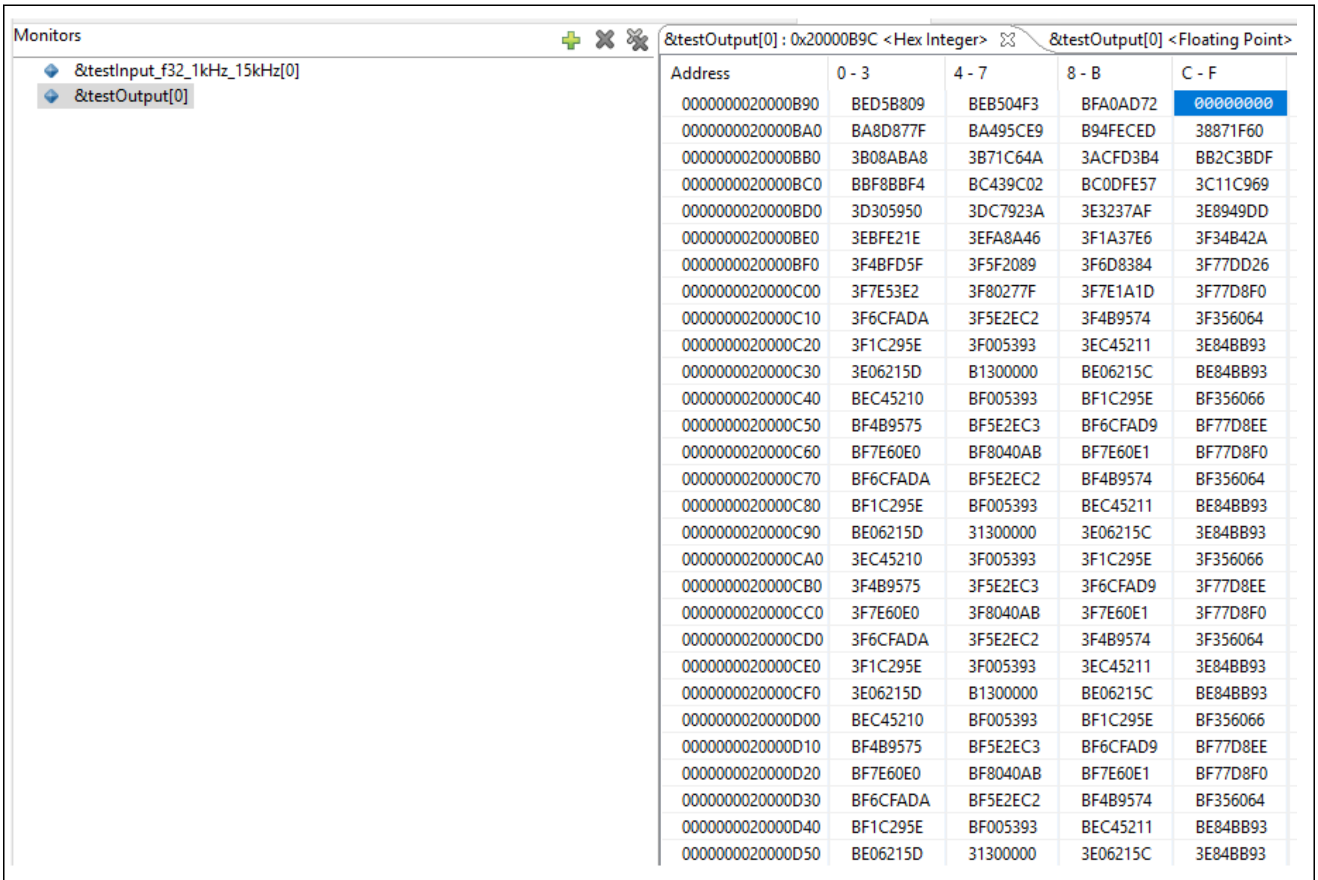


Figure 6. Lowpass Filter Example Ran Successfully and Stopped at "SUCCESS" while (1)

The waveforms of input and output buffers can be used to visually verify the result. Add `testOutput` buffer to the **Memory** tab as shown in Figure 6, then add a new **Fixed Floating Point** rendering for the same buffer.



Address	0 - 3	4 - 7	8 - B	C - F
0000000020000B90	BED5B809	BEB504F3	BFA0AD72	00000000
0000000020000BA0	BA8D877F	BA495CE9	B94FECED	38871F60
0000000020000BB0	3B08ABA8	3B71C64A	3ACFD3B4	BB2C3BDF
0000000020000BC0	BBF8BBF4	BC439C02	BC0DFE57	3C11C969
0000000020000BD0	3D305950	3DC7923A	3E3237AF	3E8949DD
0000000020000BE0	3EBFE21E	3EFA8A46	3F1A37E6	3F34B42A
0000000020000BF0	3F4BFD5F	3F5F2089	3F6D8384	3F77DD26
0000000020000C00	3F7E53E2	3F80277F	3F7E1A1D	3F77D8F0
0000000020000C10	3F6CFADA	3F5E2EC2	3F4B9574	3F356064
0000000020000C20	3F1C295E	3F005393	3EC45211	3E84BB93
0000000020000C30	3E06215D	B1300000	BE06215C	BE84BB93
0000000020000C40	BEC45210	BF005393	BF1C295E	BF356066
0000000020000C50	BF4B9575	BF5E2EC3	BF6CFAD9	BF77D8EE
0000000020000C60	BF7E60E0	BF8040AB	BF7E60E1	BF77D8F0
0000000020000C70	BF6CFADA	BF5E2EC2	BF4B9574	BF356064
0000000020000C80	BF1C295E	BF005393	BEC45211	BE84BB93
0000000020000C90	BE06215D	31300000	3E06215C	3E84BB93
0000000020000CA0	3EC45210	3F005393	3F1C295E	3F356066
0000000020000CB0	3F4B9575	3F5E2EC3	3F6CFAD9	3F77D8EE
0000000020000CC0	3F7E60E0	3F8040AB	3F7E60E1	3F77D8F0
0000000020000CD0	3F6CFADA	3F5E2EC2	3F4B9574	3F356064
0000000020000CE0	3F1C295E	3F005393	3EC45211	3E84BB93
0000000020000CF0	3E06215D	B1300000	BE06215C	BE84BB93
0000000020000D00	BEC45210	BF005393	BF1C295E	BF356066
0000000020000D10	BF4B9575	BF5E2EC3	BF6CFAD9	BF77D8EE
0000000020000D20	BF7E60E0	BF8040AB	BF7E60E1	BF77D8F0
0000000020000D30	BF6CFADA	BF5E2EC2	BF4B9574	BF356064
0000000020000D40	BF1C295E	BF005393	BEC45211	BE84BB93
0000000020000D50	BE06215D	31300000	3E06215C	3E84BB93

Figure 7. testInput_f32_1kHz_15kHz Buffer of Lowpass Filter Example

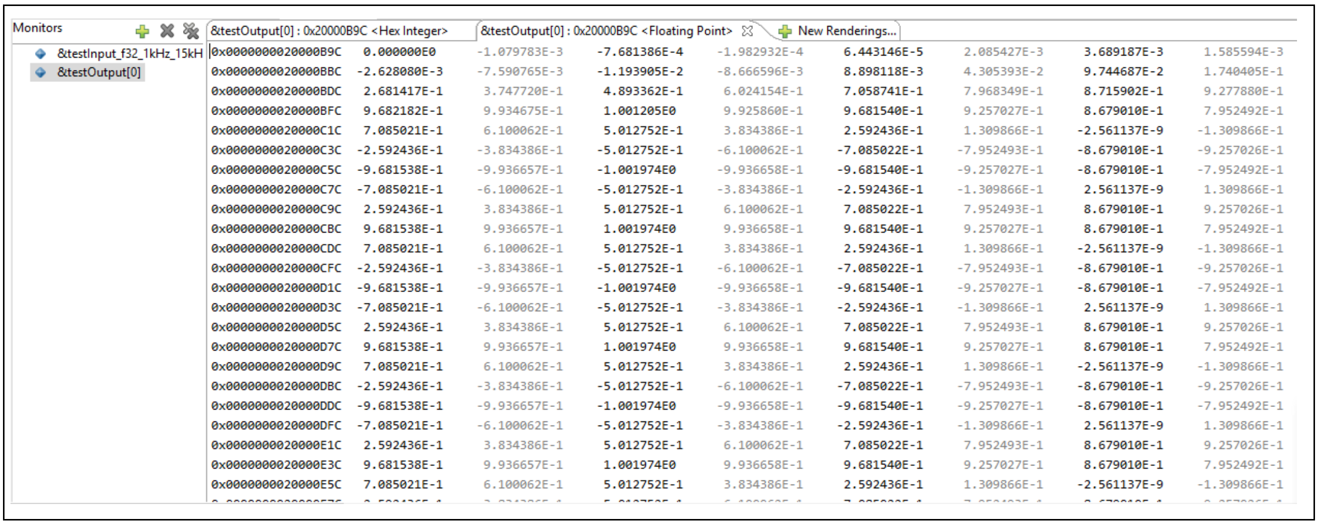


Figure 8. testOutput Buffer in Floating Point format of Lowpass Filter Example

The `testInput` and `testOutput` buffers can be visually shown using Python script and plotting libraries such as MatPlotLib. You can refer to https://matplotlib.org for more details.

You can copy floating point data from **Floating Point Rendering** tab by selecting the memory area and using **Copy to Clipboard** to copy. You need to manually edit the data to be able to use it with plotting tools.

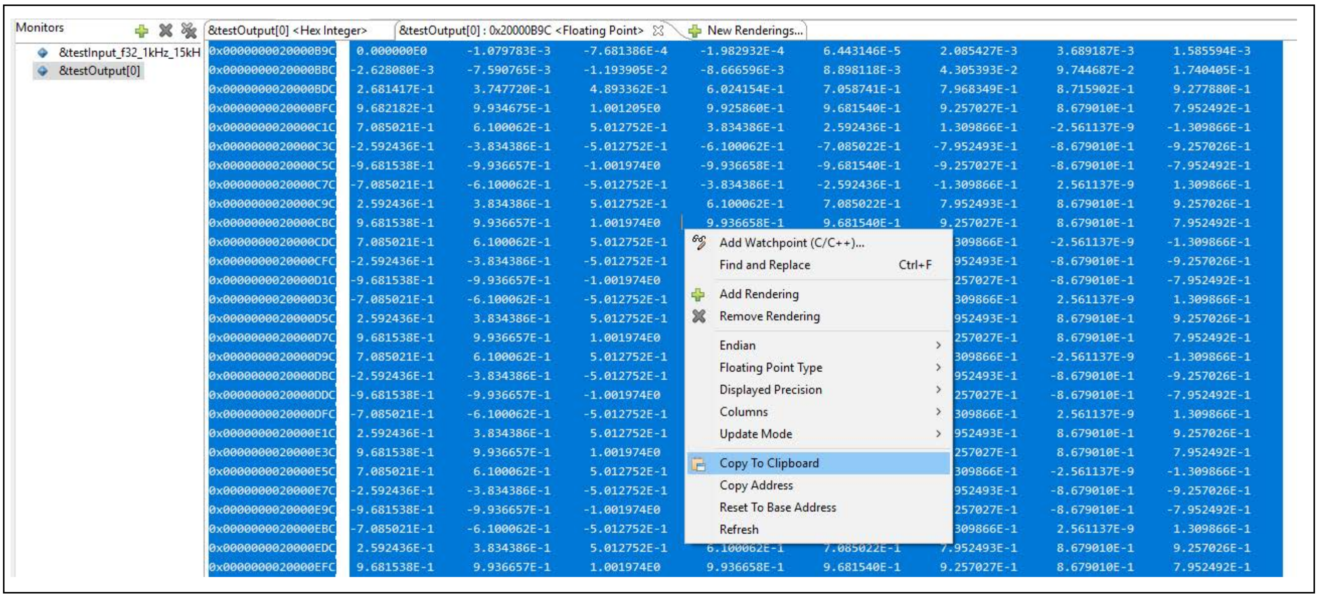


Figure 9. Copy testOutput Buffer to Clipboard

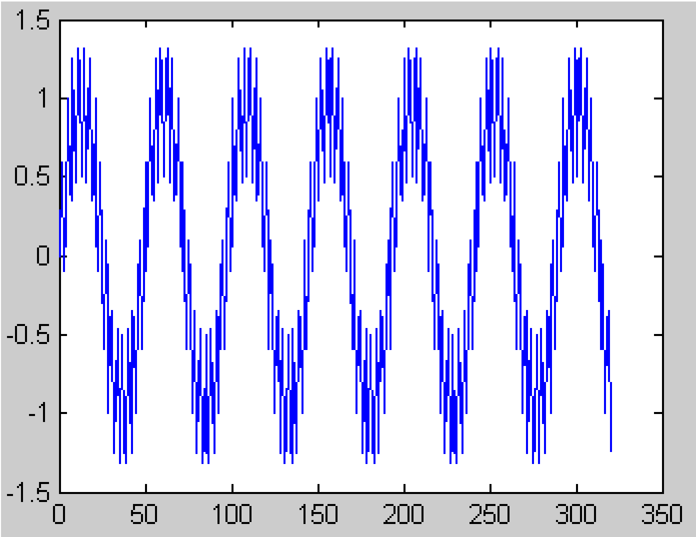


Figure 10. Waveform of Input Buffer of the Lowpass Filter Example

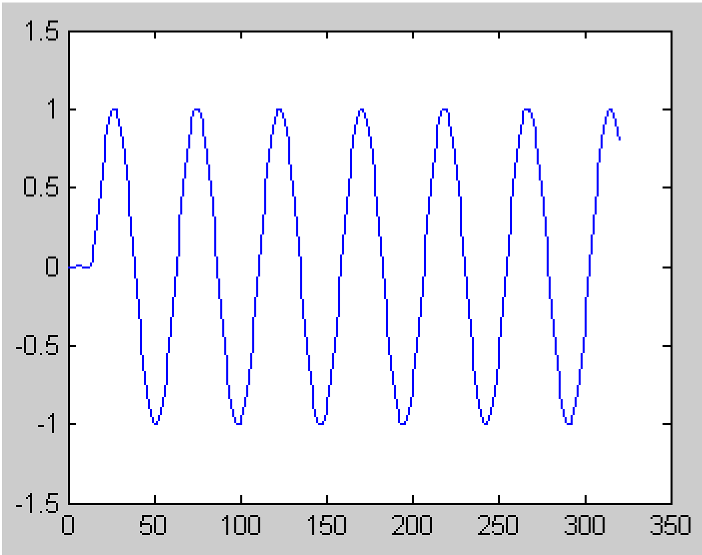


Figure 11. Waveform of Output Buffer of the Lowpass Filter Example

2.2 Project Settings

The following settings are needed to get the best performance from Cortex-M33 core with DSP extension.

Table 1. Project Settings

Parameter	Default	Setting	Notes
-mfloat-abi	hard	hard	MCU has hardware support
-mfpu	fpv5-sp-d16	fpv5-sp-d16	MCU supports this version
semihosting-enable	false	false	Do not use semihosting
-march	Toolchain default	armv8-m.main+dsp	MCU has DSP support
Optimization	-O2	-O3	Most optimized

3. Performance Measurement

3.1 Enable Performance Measurement

The `DSP_INSTRUCTION_BENCH` macro in `arm_dsp_example_config.h` is used to enable example code to measure performance of the `arm_fir_f32` function in the `arm_fir_example_f32` example. Set `DSP_INSTRUCTION_BENCH` to 1 to enable this performance measurement. Set it to 0 to disable the measurement. Refer to section 3.2 for more details.

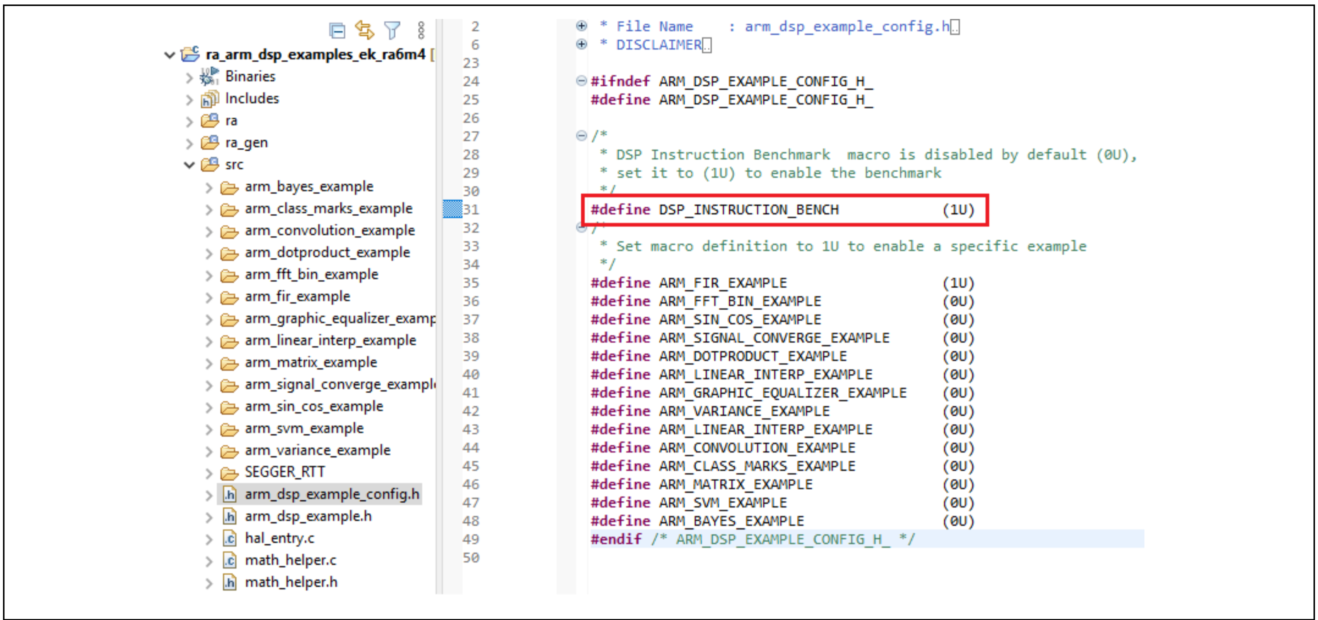


Figure 12. Enable Performance Measurement in arm_dsp_example_config.h

3.2 Adding Supporting Code to Measure DWT Cycle

Store the DWT counter (`DWT->CYCCNT`) before and after calling the DSP function.

```
#if   DSP_INSTRUCTION_BENCH   //Performance Benchmark
  uint32_t ts_0      = 0;
  uint32_t ts_1      = 0;
  uint32_t dwt_cycle = 0;
  ts_0 = DWT->CYCCNT;
#endif
  /* ------------------------------------------------------------------
  ** Call the FIR process function for every blockSize samples
  ** --------------------------------------------------------------- */
  for(i=0; i < numBlocks; i++)
  {
    arm_fir_f32(&S, inputF32 + (i * blockSize), outputF32 + (i * blockSize), blockSize);
  }
#if   DSP_INSTRUCTION_BENCH   //Performance Benchmark
  ts_1 = DWT->CYCCNT;
#endif
```

Figure 13. Store DWT Before and After Calling DSP Functions

The execution time of a specific DSP function call can be calculated basing on DWT cycle and MCU system clock frequency (ICLK) using the following formula:

Time = DWT cycle x (1/ICLK)

Note: DWT cycles is the count of MCU execution cycles. ICLK is the system clock of the MCU, which is set to 200 MHz in this application project for EK-RA6M4.

```
#if  DSP_INSTRUCTION_BENCH  //Performance Benchmark
    /* Calculate DW cycles */
    dwt_cycle = ts_1 > ts_0? ts_1 - ts_0 : ~(ts_0 - ts_1);
    APP_PRINT("DWT Clock Cycle: %d\n", dwt_cycle);
#endif
```

Figure 14. Calculate DWT Cycle

3.3 Printing Example Status and DWT Cycle

Example status and DWT cycle will be sent to SEGGER J-Link RTT Viewer. You can connect to SEGGER RTT Viewer using the following steps.

Launch SEGGER RTT Viewer V7.20a or greater from Windows Start Menu.

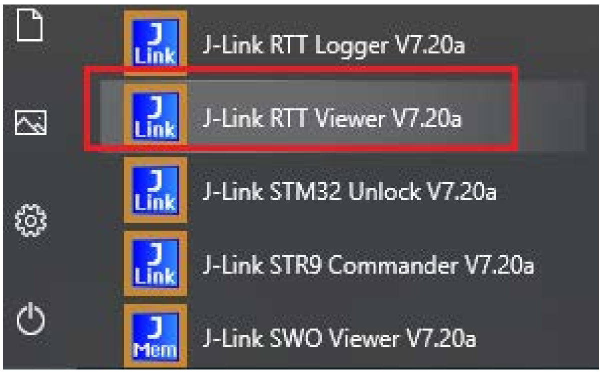


Figure 15. Launch SEGGER RTT Viewer V7.20a

Configure the SEGGER RTT Viewer.

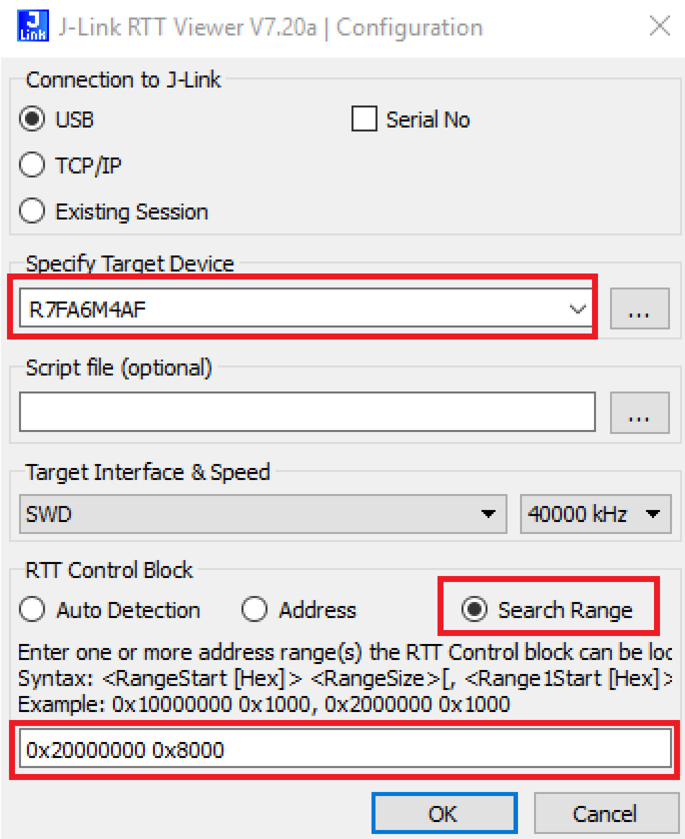


Figure 16. RTT Viewer Settings for EK-RA6M4

A typical example status and DWT cycle are shown below:

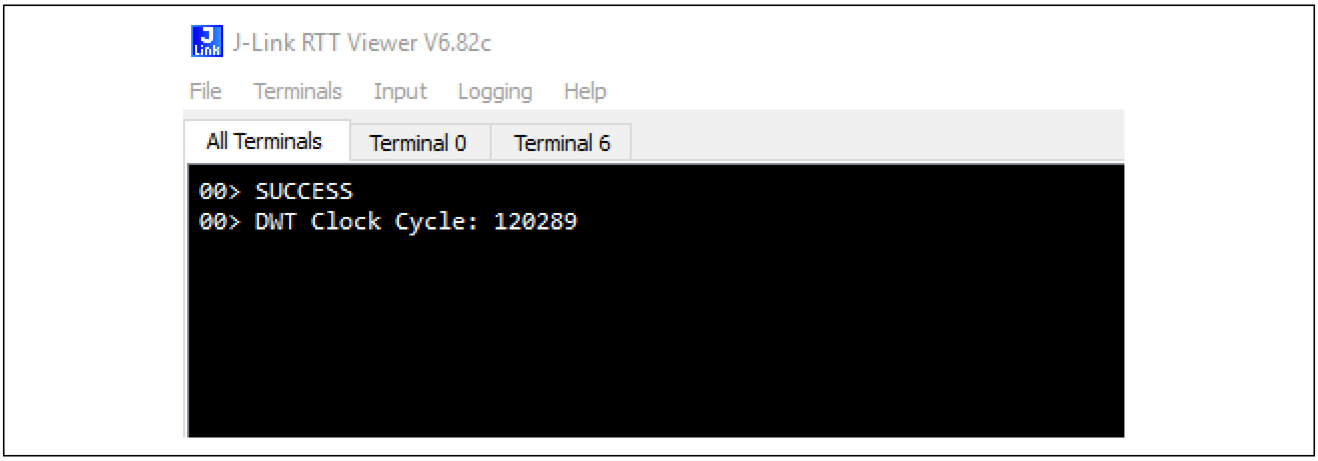


Figure 17. Enable Performance Benchmark

Website and Support

Visit the following vanity URLs to learn about key elements of the RA family, download components and related documentation, and get support.

RA Product Information	www.renesas.com/ra
RA Product Support Forum	www.renesas.com/ra/forum
RA Flexible Software Package	www.renesas.com/FSP
Renesas Support	www.renesas.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	May.17.21	—	First release document

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.
5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.
6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit: www.renesas.com/contact/.