RZ/T1 Group

CMSIS-DSP library for RZ/T1 Group: Introduction Guide

R20AN0450EJ0100
Rev.1.00
June 23, 2017

Introduction

This document describes the information for introducing the CMSIS-DSP library for RZ/T1 Group.

Features

The CMSIS-DSP library for RZ/T1 Group is based on the CMSIS-DSP Version 1.4.7 (CMSIS V4.5.0) provided by ARM® Ltd, to located in BTCM and to improve performance , the following changes.

- Change to divide the table by number of points. (Each API function in Transform Functions)
- Optimize the memory size (table) to be used by specifying the number of points used by the user. (Each API function in Transform Functions)
- Optimize the library to improve processing speed each compiler.

Target Device

RZ/T1 Group

Target IDE / Compiler / Evaluation Board

e2 studio / Renesas GCC / RENESAS Starter Kit+ for RZ/T1 Board

EWARM / IAR CC / RENESAS Starter Kit+ for RZ/T1 Board

Contents

1. Specification

The API specification of this library has not been changed from the original CMSIS-DSP Version 1.4.7. Refer to the manual of the CMSIS-DSP, which is supplied by ARM Ltd., for the API specification.

- **CMSIS**

  https://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php

- **CMSIS-DSP**

  http://www.keil.com/pack/doc/CMSIS/DSP/html/index.html

2. Operation Verification Conditions

The library is verified under the following conditions.

Table 1. Operation Verification Conditions

Item		Name	Version
Target		CMSIS-DSP Library	CMSIS Version 4.5.0 (DSP Revision: Version1.4.7)
IDE			
	Renesas GCC	Eclipse SDK Version: 4.5.2 e2 studio	Version 5.3.0.023
	IAR CC	IAR Embedded Workbench for ARM	Version 7.80.4.12495
Compiler		Renesas GCC	v16.01
		IAR CC	-
Processor		RZ/T1	-
Evaluation Board		RENESAS Starter Kit+ for RZ/T1 Board (R7S910018)	-
Emulator			
	Renesas GCC	J-Link LITE	-
	IAR CC	I-Jet	-

3. Option of Library Creation

The below table is the option of each compiler used for the library creation.

Table 2. Option of Library Creation

Compiler	Item	Option
Renesas GCC	Compiler	-nostdinc -O2 -mcpu=cortex-r4r -march=armv7-r -mlittle-endian -mthumb -mthumb-interwork -mfloat-abi=softfp -mfpu=vfp -DDOUBLE_HAS_64_BITS -std=c99 -fno-strict-aliasing
	Assembler	-mlittle-endian -mcpu=cortex-r4r -march=armv7-r –mthumb -mfloat-abi=softfp -mfpu=vfp
IAR CC	Compiler	--endian=little --cpu=Cortex-R4 -e --fpu=VFPv3_D16 --cpu_mode thumb -Ohs --no_size_constraints
	Assembler	-s+ -M<> -w+ --cpu Cortex-R4 --fpu VFPv3_D16

4. Software

4.1 Software Structure

The below figure is the example of the software structure for using this library.

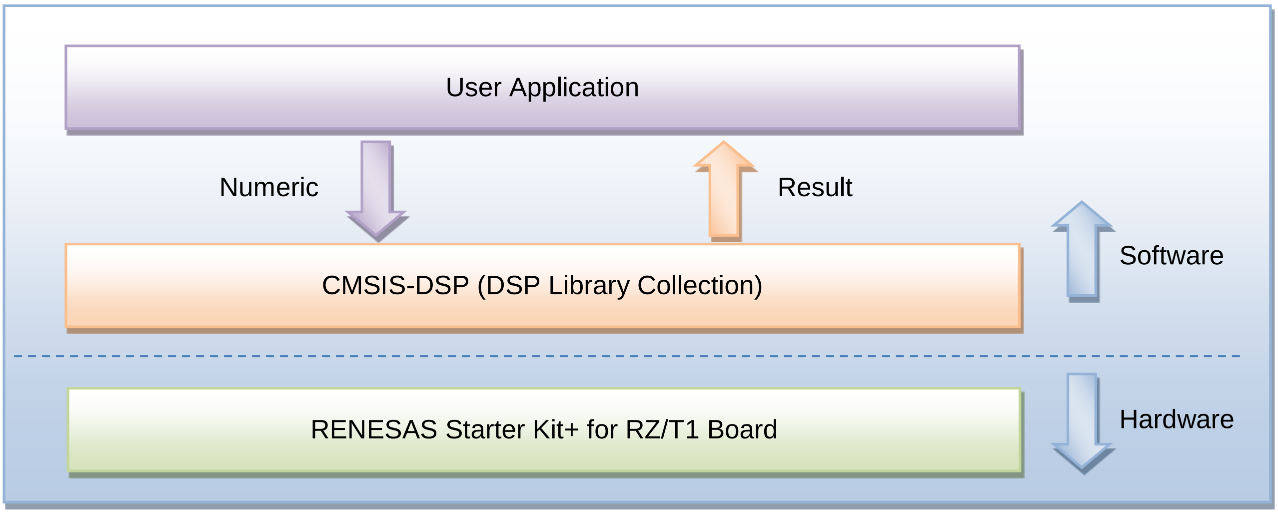

Figure 1. Example of Software Structure

4.2 High-Speed Memory Location

This library can provide maximum performance by being located in memory to meet the following conditions.

- High-Speed Memory Location Conditions (All of the following 1 to 3 are satisfied.)

1. The API functions (code section) and the memory being referenced and updated are located in tightly coupled memory (since TCM).
2. The API functions (code section) and the memory being referenced and updated are located in different TCM.
3. In particular, if the CPU clock frequency is 450MHz or higher, the API functions (code section) are located in ATCM, the memory being referenced and updated are located in BTCM, and set “ATCM Wait Setting” of “ATCM Wait Control Register” to “1-wait with optimization”.

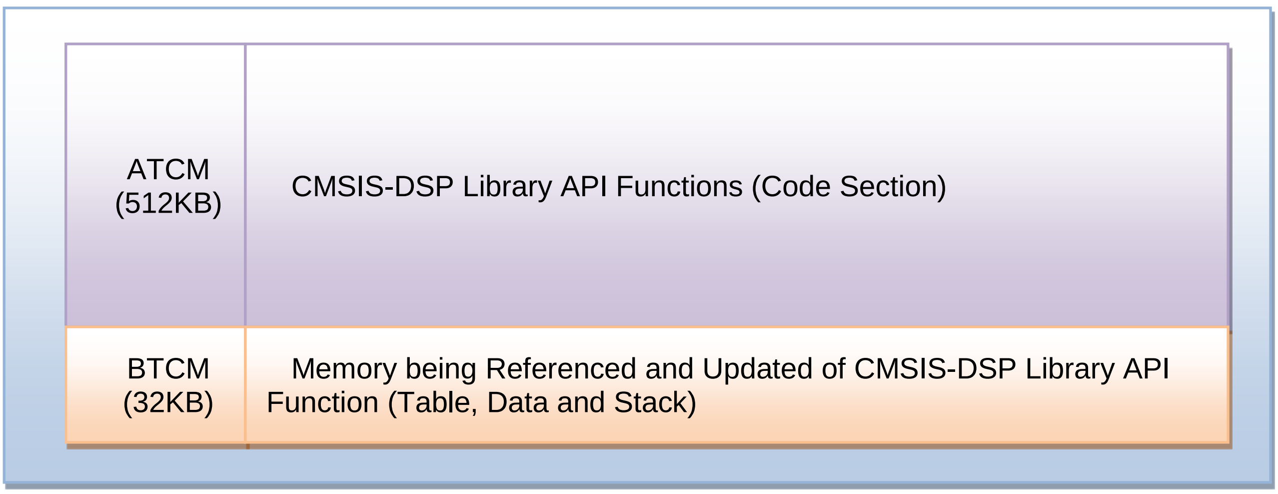

Figure 2 Example of High-Speed Memory Location (CPU clock frequency is 450MHz or higher)

4.3 Folder Structure

The below table is the folder structure of the library.

Table 3. Folder Structure

\an-r20an0450ej0100-rzt1	<DIR>	Folder for CMSIS-DSP Software
r20an0450ej0100-rzt1.pdf		This application Note
\workspace	<DIR>	Folder for Work Space
\build	<DIR>	Folder for Reference File
\gcc	<DIR>	Folder for e2 studio
\RZ_T1_CMSIS_DSP_LIB	<DIR>	
.cproject		Configuration File
.info		Configuration File
.project		Configuration File
\iccarm	<DIR>	Folder for EWARM
RZ_T1_CMSIS_DSP_LIB.dep		Configuration File
RZ_T1_CMSIS_DSP_LIB.ewd		Configuration File
RZ_T1_CMSIS_DSP_LIB.ewp		Configuration File
RZ_T1_CMSIS_DSP_LIB.ewt		Configuration File
RZ_T1_CMSIS_DSP_LIB.eww		Configuration File
\DSP_Lib	<DIR>	Folder for CMSIS-DSP Library Source Code
Source		Folder for CMSIS-DSP Library Source Code
license.txt		License File
\Lib	<DIR>	Folder for CMSIS-DSP Library File
\gcc	<DIR>	Folder for Library File for Renesas GCC
libRZ_T1_CMSIS_DSP_LIB_gcc.a		CMSIS-DSP Library File
\iarcc	<DIR>	Folder for Library File for IAR CC.
libRZ_T1_CMSIS_DSP_LIB_iarcc.a		CMSIS-DSP Library File
license.txt		License File
\Include	<DIR>	Folder for Header File
arm_common_tables.h		Header File
arm_const_structs.h		Header File
arm_math.h		Header File
cmsis_armcc.h		Header File
cmsis_armcc_V6.h		Header File
cmsis_gcc.h		Header File
core_cm0.h		Header File
core_cm0plus.h		Header File
core_cm3.h		Header File
core_cm4.h		Header File
core_cm7.h		Header File
core_cmFunc.h		Header File
core_cmInstr.h		Header File
core_cmSimd.h		Header File
core_sc000.h		Header File
core_sc300.h		Header File

\InitSrc	<DIR>	Folder for Source File
arm_cfft_radix2_init_f32.c		Source File
arm_cfft_radix2_init_q15.c		Source File
arm_cfft_radix2_init_q31.c		Source File
arm_cfft_radix4_init_f32.c		Source File
arm_cfft_radix4_init_q15.c		Source File
arm_cfft_radix4_init_q31.c		Source File
arm_dct4_init_f32.c		Source File
arm_dct4_init_q15.c		Source File
arm_dct4_init_q31.c		Source File
arm_rfft_init_f32.c		Source File
arm_rfft_fast_init_f32.c		Source File
arm_rfft_init_q15.c		Source File
arm_rfft_init_q31.c		Source File

4.4 Required Memory Size

The below table is the required memory size of the library.

Table 4. Required Memory Size

COMPILER	ROM (Kbyte)	RAM (Kbyte)	STACK (byte)
Renesas GCC	309	0	384
IAR CC	336	0	268

5. Configuration

The API functions in Transform Functions of the CMSIS-DSP library for RZ/T1 Group are extended to be able to optimize the tables that are used for the number of points used by user application. With this extension, for example, in the case of API function "arm_cfft_f32", the high-speed memory location of the library can be achieved up to "the number of points 1024 (*1)".

Users can use this extension simply by adding a macro definition to the integrated development environment that corresponds to the number of points of the API functions they use. For more information, refer to the procedure of "5.1 How to Configure" below and the macro definition.

*1 The memory size used by the use application is not considered. For the high-speed memory location, refer to "4.2 High-Speed Memory ".

5.1 How to Configure

This section describes how to configure this library.

- Procedure (Overview)
  1. Select the API function and the number of points that you want to use.
  2. The number of points of the API function in step 1 can be enabled by adding macro obtained from the "6.3 Macro" to the macro definition (symbol definition) of the compiler options for the integrated development environment.
  3. If you want to use multiple the number of points, repeat steps 1 through 2.

- Procedure (Detail)

1. Select the API function and the number of points that you want to use. For the selectable combination, refer to "Table 6. List of Supported Number of Points".

For example, if you select the number of points of the API function "arm_cfft_f32", you can select one of 16 to 4096 of supported the number of points from the figure below.

No.	API Function Name	Supported Number of Points									
		16	32	64	128	256	512	1024	2048	4096	8192
1	arm_cfft_f32	○	○	○(*1)	○	○	○(*1)	○	○	○(*1)	—

Figure 3. Number of Points Supported by API Function "arm_cfft_f32" (*2)

*2 Excerpts from "Table 6. List of Supported Number of Points".

2. The number of points of the API function in step 1 can be enabled by adding macro obtained from the "6.3 Macro" to the macro definition (symbol definition) of the compiler options for the integrated development environment.

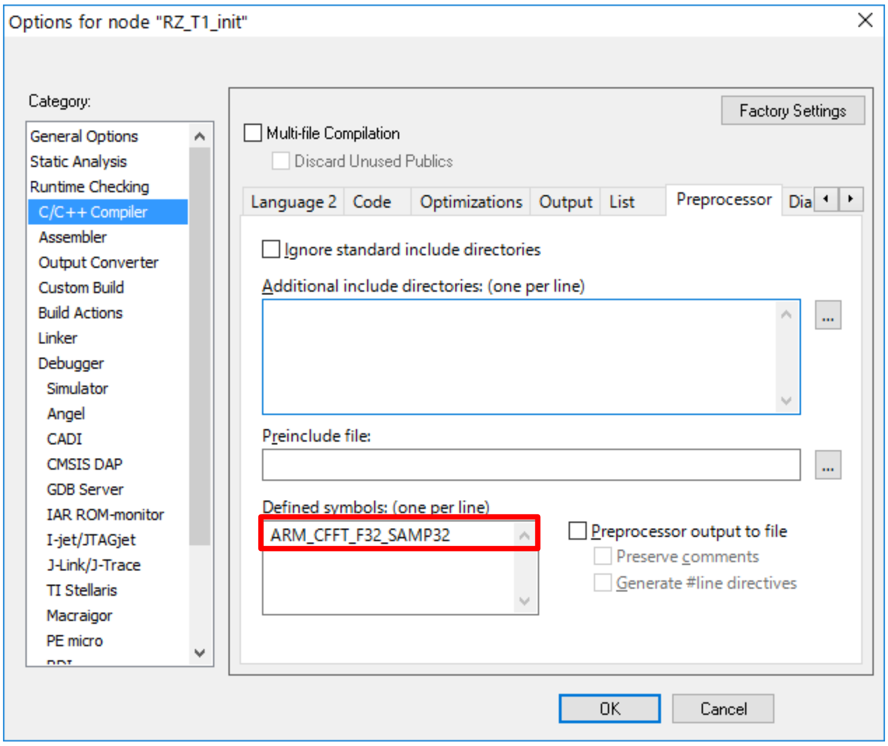

Figure 4. Example of enabling "Number of Points 32" in API Function "arm_cfft_f32" in EWARM

3. If you want to use multiple the number of points, repeat steps 1 through 2.

If you enable multiple the number of points of constants, the macro definition of the compiler options for the integrated development environment adds all the macro definition that you want to enable. The same applies if you use multiple API functions.

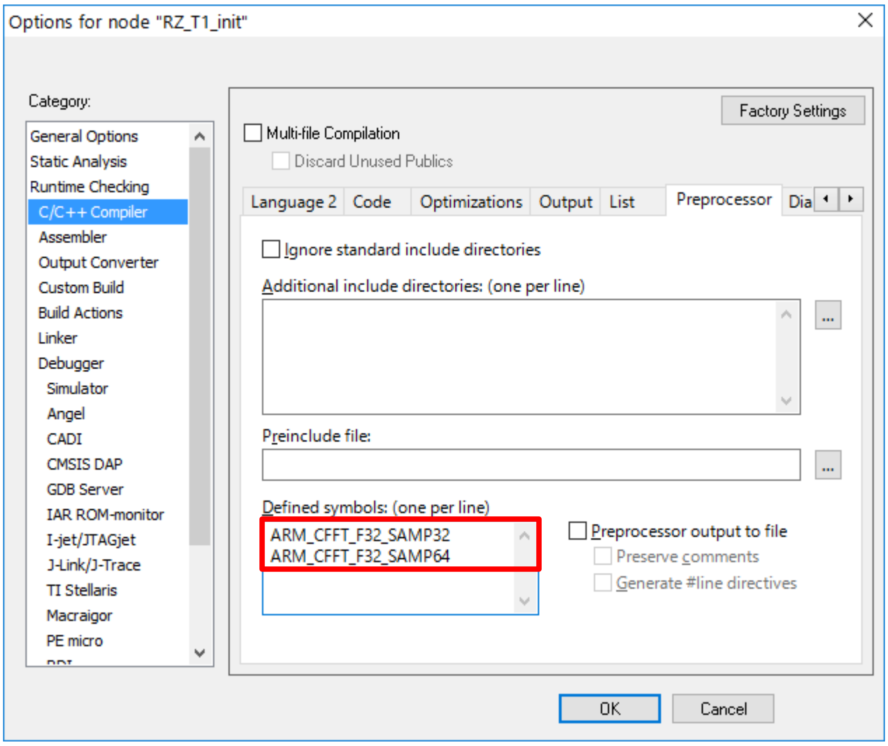

Figure 5. Example of enabling "Number of Points 32 and 64" in API function "arm_cfft_f32" in EWARM

5.2 Memory Usage

The memory size used by the number of points of the selected API function can be checked in "Table 8. List of Memory Usage of Transform Functions".

For example, if you select "number of points 32" in API function "arm_cfft_f32", the memory usage is "0.6KB" from figure 6 (red frame portion).

No.	APIFunction Name	NumberofPoints	Table (KB)	Buffer (KB)	Total (KB)
2	arm_cfft_f32	32	0.3	0.3	0.6

Figure 6. Memory Usage when you selected "Number of Points 32" in API Function "arm_cfft_f32" (*2)

When you select multiple the number of points, the memory usage is the sum of the table size and the maximum buffer size.

For example, if you select "number of points 32 and 64" in API function "arm_cfft_f32", the memory usage is following from figure 7 (red frame portion).

```
Table Size:0.3(KB) + 0.6(KB) = 0.9(KB)
Buffer Size: Maximum Buffer size 0.5(KB)
Total:1.4(KB)
```

No.	APIFunction Name	NumberofPoints	Table (KB)	Buffer (KB)	Total (KB)
2	arm_cfft_f32	32	0.3	0.3	0.6
3		64	0.6	0.5	1.1

Figure 7. Memory Usage when you selected "Number of Points 32 and 64" in API Function "arm_cfft_f32" (*2)

*2 Excerpts from "Table 8. List of Memory Usage of Transform Functions".

5.3 Caution

If you use "5. Configuration", refer to "Table 5. List of Initialization Functions", and call the corresponding initialization function before executing the API function that you want to use. If there is no initialization function, you do not need it.

6. Transform Functions Information

This section describes the information about how to use the API function in Transform Functions.

6.1 Initialization Function

The below table is the list of initialization function that is invoked before each API function in the Transform Functions is executed.

- Explanation of symbol
  －: Not Applicable

Table 5. List of Initialization Functions

No.	API Function Name	Initialization Function Name
1	arm_cfft_f32	—
2	arm_cfft_q15	—
3	arm_cfft_q31	—
4	arm_cfft_radix2_f32	arm_cfft_radix2_init_f32
5	arm_cfft_radix2_q15	arm_cfft_radix2_init_q15
6	arm_cfft_radix2_q31	arm_cfft_radix2_init_q31
7	arm_cfft_radix4_f32	arm_cfft_radix4_init_f32
8	arm_cfft_radix4_q15	arm_cfft_radix4_init_q15
9	arm_cfft_radix4_q31	arm_cfft_radix4_init_q31
10	arm_cfft_radix8_f32	—
11	arm_dct4_f32	arm_dct4_init_f32
12	arm_dct4_q15	arm_dct4_init_q15
13	arm_dct4_q31	arm_dct4_init_q31
14	arm_rfft_f32	arm_rfft_init_f32
15	arm_rfft_fast_f32	arm_rfft_fast_init_f32
16	arm_rfft_q15	arm_rfft_init_q15
17	arm_rfft_q31	arm_rfft_init_q31

6.2 Number of Points

The below table is the number of points that each API function in the Transform Functions is supported.

- Explanation of symbol
  ○: Supported
  ―: Unsupported

Table 6. List of Supported Number of Points

No.	API Function Name	Supported Number of Points									
		16	32	64	128	256	512	1024	2048	4096	8192
1	arm_cfft_f32	○	○	○(*1)	○	○	○(*1)	○	○	○(*1)	―
2	arm_cfft_q15	○(*2)	○	○(*2)	○	○(*2)	○	○(*2)	○	○(*2)	―
3	arm_cfft_q31	○(*2)	○	○(*2)	○	○(*2)	○	○(*2)	○	○(*2)	―
4	arm_cfft_radix2_f32	○	○	○	○	○	○	○	○	○	―
5	arm_cfft_radix2_q15	○	○	○	○	○	○	○	○	○	―
6	arm_cfft_radix2_q31	○	○	○	○	○	○	○	○	○	―
7	arm_cfft_radix4_f32	○	―	○	―	○	―	○	―	○	―
8	arm_cfft_radix4_q15	○	―	○	―	○	―	○	―	○	―
9	arm_cfft_radix4_q31	○	―	○	―	○	―	○	―	○	―
10	arm_cfft_radix8_f32	―	―	○	―	―	○	―	―	○	―
11	arm_dct4_f32	―	―	―	○	―	○	―	○	―	○
12	arm_dct4_q15	―	―	―	○	―	○	―	○	―	○
13	arm_dct4_q31	―	―	―	○	―	○	―	○	―	○
14	arm_rfft_f32	―	―	―	○	―	○	―	○	―	○
15	arm_rfft_fast_f32	―	○	○	○	○	○	○	○	○	―
16	arm_rfft_q15	―	○	○	○	○	○	○	○	○	○
17	arm_rfft_q31	―	○	○	○	○	○	○	○	○	○

*1 Treat as "arm_cfft_radix8_f32".
*2 Treat "arm_cfft_radix4_q15" and "arm_cfft_radix4_q31" as equivalent.

6.3 Macro

The below table is how to name of the macro name when you enable the number of points of API function in the Transform Function.

Table 7. Macro

Macro Name	Description
[XXX]_SAMP[YYYY]	Specifies number of points of API function to enable. [XXX] = The API function name in the Transform Functions is specified by all uppercase. [YYY] = Specifies number of points. The value is one of the following: 16, 32, 64, 128, 256, 512, 1024, 2048, 4096 or 8192.

Example of macro name specifying "number of points 16"in API function "arm_cfft_f32"

```
ARM_CFFT_F32_SAMP16
```

6.4 Memory Usage

The below tables are table size and buffer size every the number of points of API function in the Transform Function. The buffer size is a reference value for the memory size that the user must minimize for input/output buffers when using API functions.

Table 8. List of Memory Usage of Transform Functions (1/3)

No.	API Function Name	Number of Points	Table (KB)	Buffer (KB)	Total (KB)
1	arm_cfft_f32	16	0.2	0.1	0.3
2		32	0.3	0.3	0.6
3		64	0.6	0.5	1.1
4		128	1.4	1.0	2.4
5		256	2.9	2.0	4.9
6		512	4.9	4.0	8.9
7		1024	11.5	8.0	19.5
8		2048	23.4	16.0	39.4
9		4096	39.9	32.0	71.9
10	arm_cfft_q15	16	0.1	0.1	0.1
11		32	0.1	0.1	0.3
12		64	0.3	0.3	0.5
13		128	0.6	0.5	1.1
14		256	1.2	1.0	2.2
15		512	2.4	2.0	4.4
16		1024	4.9	4.0	8.9
17		2048	9.9	8.0	17.9
18		4096	19.9	16.0	35.9
19	arm_cfft_q31	16	0.1	0.1	0.2
20		32	0.2	0.3	0.5
21		64	0.5	0.5	1.0
22		128	1.0	1.0	2.0
23		256	2.0	2.0	4.0
24		512	3.9	4.0	7.9
25		1024	7.9	8.0	15.9
26		2048	15.9	16.0	31.9
27		4096	31.9	32.0	63.9
28	arm_cfft_radix2_f32	16	0.1	0.1	0.3
29		32	0.3	0.3	0.5
30		64	0.5	0.5	1.0
31		128	1.1	1.0	2.1
32		256	2.1	2.0	4.1
33		512	4.3	4.0	8.3
34		1024	8.5	8.0	16.5
35		2048	17.0	16.0	33.0
36		4096	34.0	32.0	66.0

Table 8. List of Memory Usage of Transform Functions (2/3)

No.	API Function Name	Number of Points	Table (KB)	Buffer (KB)	Total (KB)
37	arm_cfft_radix2_q15	16	0.1	0.1	0.1
38		32	0.1	0.1	0.2
39		64	0.2	0.3	0.5
40		128	0.4	0.5	0.9
41		256	0.9	1.0	1.9
42		512	1.8	2.0	3.8
43		1024	3.5	4.0	7.5
44		2048	7.0	8.0	15.0
45		4096	14.0	16.0	30.0
46	arm_cfft_radix2_q31	16	0.1	0.1	0.2
47		32	0.2	0.3	0.5
48		64	0.4	0.5	0.9
49		128	0.8	1.0	1.8
50		256	1.6	2.0	3.6
51		512	3.3	4.0	7.3
52		1024	6.5	8.0	14.5
53		2048	13.0	16.0	29.0
54		4096	26.0	32.0	58.0
55	arm_cfft_radix4_f32	16	0.1	0.1	0.3
56		64	0.5	0.5	1.0
57		256	2.1	2.0	4.1
58		1024	8.5	8.0	16.5
59		4096	34.0	32.0	66.0
60	arm_cfft_radix4_q15	16	0.1	0.1	0.1
61		64	0.2	0.3	0.5
62		256	0.9	1.0	1.9
63		1024	3.5	4.0	7.5
64		4096	14.0	16.0	30.0
65	arm_cfft_radix4_q31	16	0.1	0.1	0.2
66		64	0.4	0.5	0.9
67		256	1.6	2.0	3.6
68		1024	6.5	8.0	14.5
69		4096	26.0	32.0	58.0
70	arm_cfft_radix8_f32	64	0.6	0.5	1.1
71		512	4.9	4.0	8.9
72		4096	39.9	32.0	71.9

Table 8. List of Memory Usage of Transform Functions (3/3)

No.	API関数名	サンプリング点数	テーブル(KB)	バッファ(KB)	合計(KB)
73	arm_rfft_f32	128	1.5	2.0	3.5
74		512	6.1	8.0	14.1
75		2048	24.5	32.0	56.5
76		8192	98.0	128.0	226.0
77	arm_rfft_fast_f32	32	0.3	0.5	0.8
78		64	0.6	1.0	1.6
79		128	1.1	2.0	3.1
80		256	2.4	4.0	6.4
81		512	4.9	8.0	12.9
82		1024	8.9	16.0	24.9
83		2048	19.5	32.0	51.5
84		4096	39.4	64.0	103.4
85	arm_rfft_q15	32	0.2	0.3	0.5
86		64	0.4	0.5	0.9
87		128	0.8	1.0	1.8
88		256	1.6	2.0	3.6
89		512	3.2	4.0	7.2
90		1024	6.4	8.0	14.4
91		2048	12.9	16.0	28.9
92		4096	25.9	32.0	57.9
93		8192	51.9	64.0	115.9
94	arm_rfft_q31	32	0.4	0.5	0.9
95		64	0.7	1.0	1.7
96		128	1.5	2.0	3.5
97		256	3.0	4.0	7.0
98		512	6.0	8.0	14.0
99		1024	11.9	16.0	27.9
100		2048	23.9	32.0	55.9
101		4096	47.9	64.0	111.9
102		8192	95.9	128.0	223.9
103	arm_dct4_f32	128	3.0	2.0	5.0
104		512	12.1	8.0	20.1
105		2048	48.5	32.0	80.5
106		8192	194.0	128.0	322.0
107	arm_dct4_q15	128	1.5	1.0	2.5
108		512	6.2	4.0	10.2
109		2048	24.9	16.0	40.9
110		8192	99.9	64.0	163.9
111	arm_dct4_q31	128	3.0	2.0	5.0
112		512	12.0	8.0	20.0
113		2048	47.9	32.0	79.9
114		8192	191.9	128.0	319.9

6.5 Caution

Some number of points in the API functions have a total memory size 32kb or high, and cannot be adjusted to BTCM memory size 32kb or less, so be careful when using them.

7. Pre-processor Macros of CMSIS-DSP

The below table is the pre-processor macros of CMSIS-DSP selected by this library.

The pre-processor macros are defined in “arm_math.h”. If you want to change the macro definition, regenerate the library.

Table 9. Pre-processor Macros of CMSIS-DSP

Macro Name	Value	Description
ARM_MATH_MATRIX_CHECK	(None)	To ensure the robustness of the function, Check on the input and output sizes of matrices.
ARM_MATH_ROUNDING	(None)	Considering calculation error, Round on support functions.
UNALIGNED_SUPPORT_DISABLE	(None)	Disable unaligned memory access, as alignment fault does not occur.
__FPU_PRESENT	1	Because the target supports FPU, Build on FPU supported.
ARM_MATH_CM4	(None)	Select the source code that is the best for the CMSIS-DSP for RZ/T1 Group.

8. How to Use The Library

This section describes about how to use the library.

- Procedure (Overview)

1. Include “arm_math.h” to use the API function of this library.
2. Link the library file used by the target compiler.
3. If you use the API functions in Transform Functions, refer to “5.1 How to Configure” to enable the macro definition for number of points of the API function that you want to use.

- Procedure (Detail)

1. Include “arm_math.h” to use the API function of this library.

2. Link the library file used by the target compiler.

Table 10. Library File Used by Compiler

No	Compiler	Library File
1	Renesas GNU Compiler	libRZ_T1_CMSIS_DSP_LIB_gcc.a
2	IAR Compiler	libRZ_T1_CMSIS_DSP_LIB_iarcc.a

3. If you use the API functions in Transform Functions, refer to “5.1 How to Configure” to enable the macro definition for number of points of the API function that you want to use.

9. Effect of Optimization of CMSIS-DSP Library for RZ/T1 Group

- The API functions in the Transform Functions can be expected to improve performance by an average of more than 40% (*1), compared to the API functions that do not support saving memory.
- About the other API functions, except for the above, there can be expected to improve performance by an average of more than 10% (*1), compared to the API functions that do not support saving memory.

*1 The Compiler is the Renesas GNU Compiler, and the CPU clock frequency is 450MHz or higher.

10. License

All pre-build libraries contained in the folders "ARM", "GCC" and "IAR"
are guided by the following license:

11. Website and Support

Renesas Electronics Website
http://www.renesas.com/

Inquiries
http://www.renesas.com/contact/

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	June 23, 2017	-	First edition issued

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. 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 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. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment 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 moment 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 moment 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 moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has 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. Moreover, 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.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the 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.

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-585-100, Fax: +44-1628-585-900

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709, Quantum Plaza, No.27 ZhiChunLu Haidian District, Beijing 100191, P.R.China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, P. R. China 200333
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 6.0