<RCAR/V4H>

V4H

	Created	Reviewed	Approved
Signature	Luis Kim		
Date	2023.08.03		
Version			

Version	Date	Change explanation	Rev. by	Reviewed	Approved
R01	2023.08.03	Initial release	Luis.Kim		

TABLE OF CONTENTS

1 개요

1.1 목적

- RCAR-V4H 의 기능을 설명하고 상세화 하여 이해한다.

1.2 Scope

- IPL 이해
- Real Time Core 의 이해
- ICUMX 의 이해
- MCAL 을 이용한 TRACE32 활용

1.3 Abbreviations, Acronyms and Definitions

문서에서 사용하는 약어와 단어를 설명한다.

Table 1-1 Abbreviations

Abbreviation	Description

Table 1-2 Acronyms

Acronyms	Description

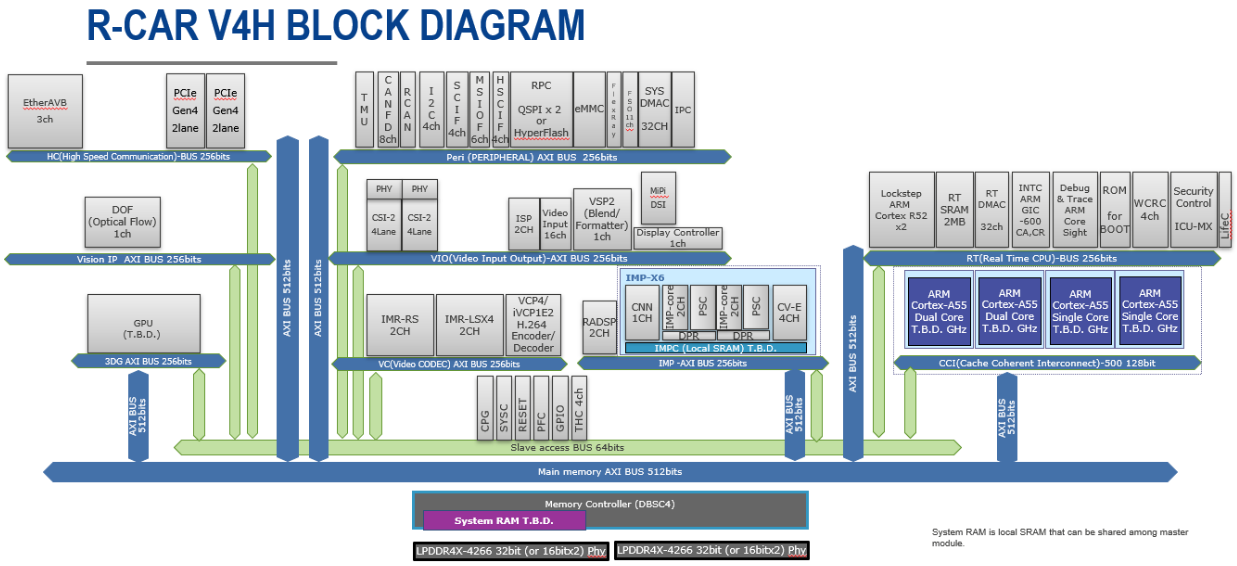
1.4 References

Table 1-3 References

#	Document	Ver	Descriptions

2 V4H ARCHITECTURE

2.1 Block Diagram



System RAM is local SRAM that can be shared among master module.

3 BOOT

3.1 Overview

Cortex-R52 or ICUMX master boot processor executes the instructions in on-chip ROM.
(Datasheet page 890)

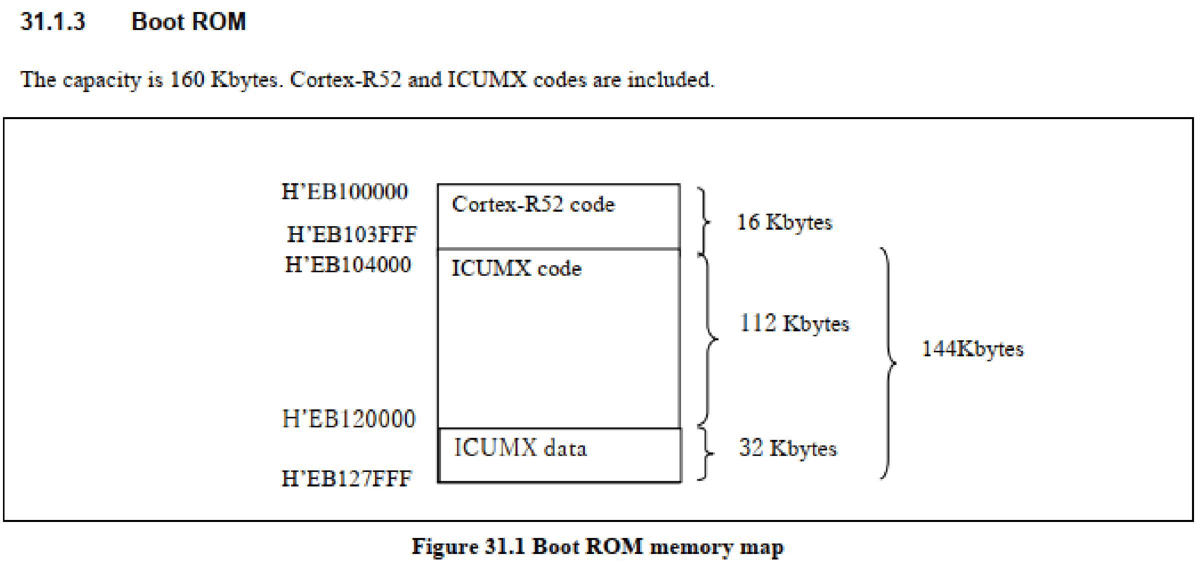
IPL 의 부팅 ROM 매개변수, 주소 정보 및 데이터 크기 정보를 포함한 데이터는 HyperFlash, Octal SPI 플래시, 직렬 플래시 및 eMMC 에서 RT-VRAM 으로 전송됩니다. 그런 다음 데이터 크기 정보에 지정된 데이터 양이 RT-VRAM 으로 전송됩니다. 전송 후 프로세서는 IPL 의 주소 정보에 지정된 주소로 점프합니다.

IPL: Initial Program Loader.

3.1.1 Secure boot supported

- Secure boot is supported if a master boot processor is ICUMX.
- Encrypted boot image is supported.
- Debugging is disabled at booting, and it can be enabled with secure debugging authentication.
- ICUMX can only support one-time authentication by using the Challenge and Response integrity check.

3.1.2 Boot Rom area

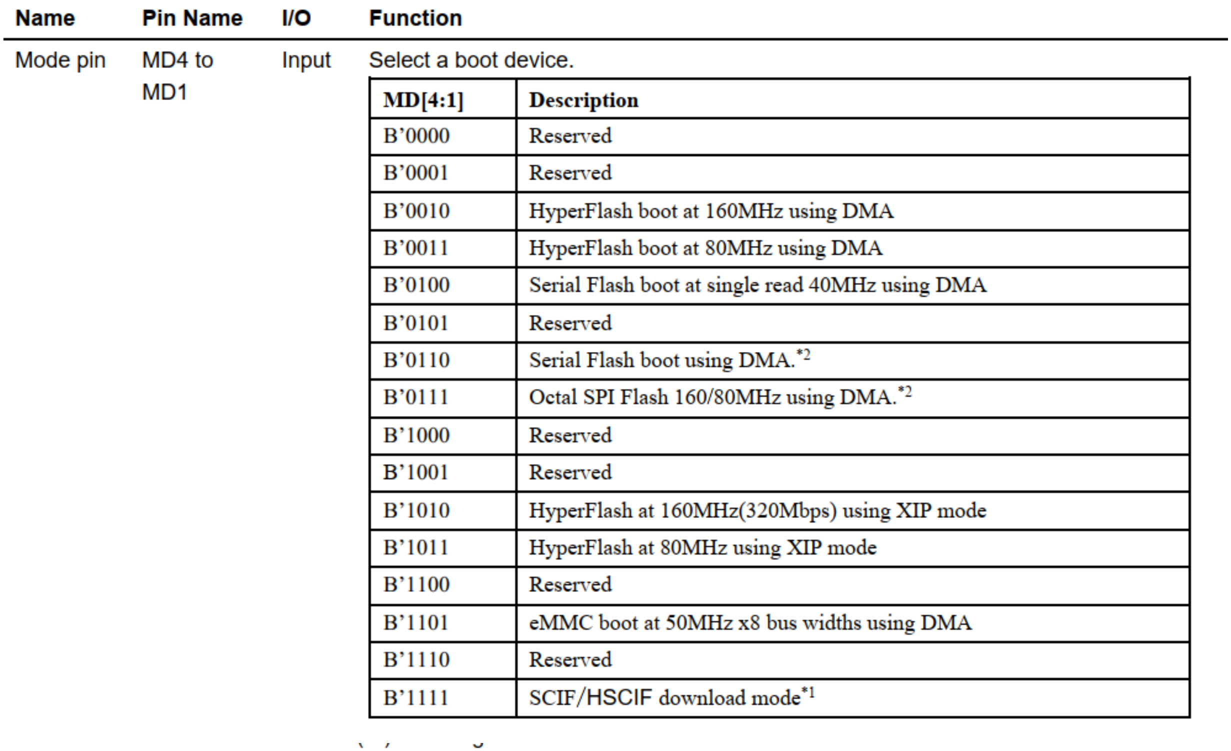


31.1.3 Boot ROM

The capacity is 160 Kbytes. Cortex-R52 and ICUMX codes are included.

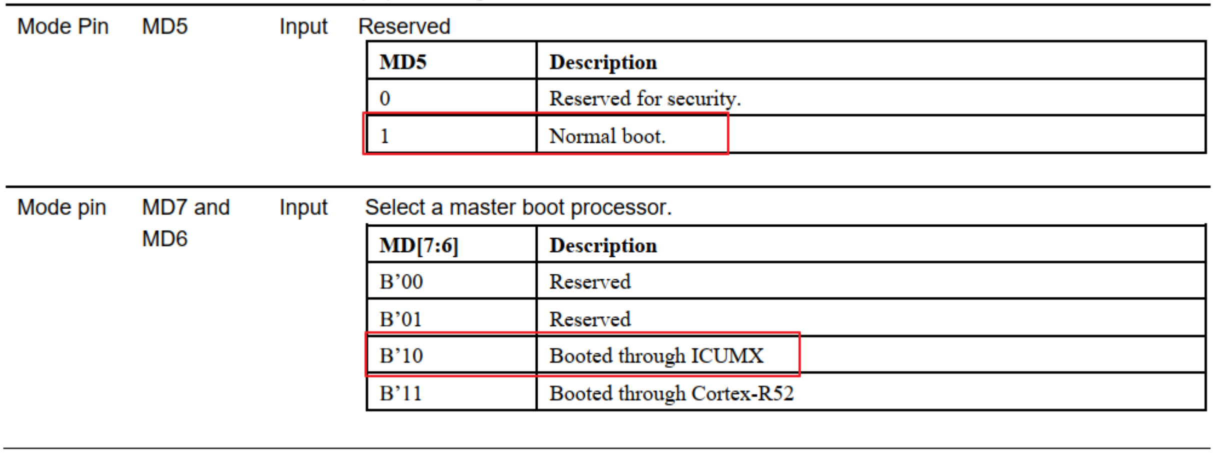
Figure 31.1 Boot ROM memory map

3.1.3 Booting mode



Name	Pin Name	I/O	Function
Mode pin	MD4 to MD1	Input	Select a boot device.

MD[4:1]	Description
B’0000	Reserved
B’0001	Reserved
B’0010	HyperFlash boot at 160MHz using DMA
B’0011	HyperFlash boot at 80MHz using DMA
B’0100	Serial Flash boot at single read 40MHz using DMA
B’0101	Reserved
B’0110	Serial Flash boot using DMA.[*2]
B’0111	Octal SPI Flash 160/80MHz using DMA.[*2]
B’1000	Reserved
B’1001	Reserved
B’1010	HyperFlash at 160MHz(320Mbps) using XIP mode
B’1011	HyperFlash at 80MHz using XIP mode
B’1100	Reserved
B’1101	eMMC boot at 50MHz x8 bus widths using DMA
B’1110	Reserved
B’1111	SCIF/HSCIF download mode[*1]

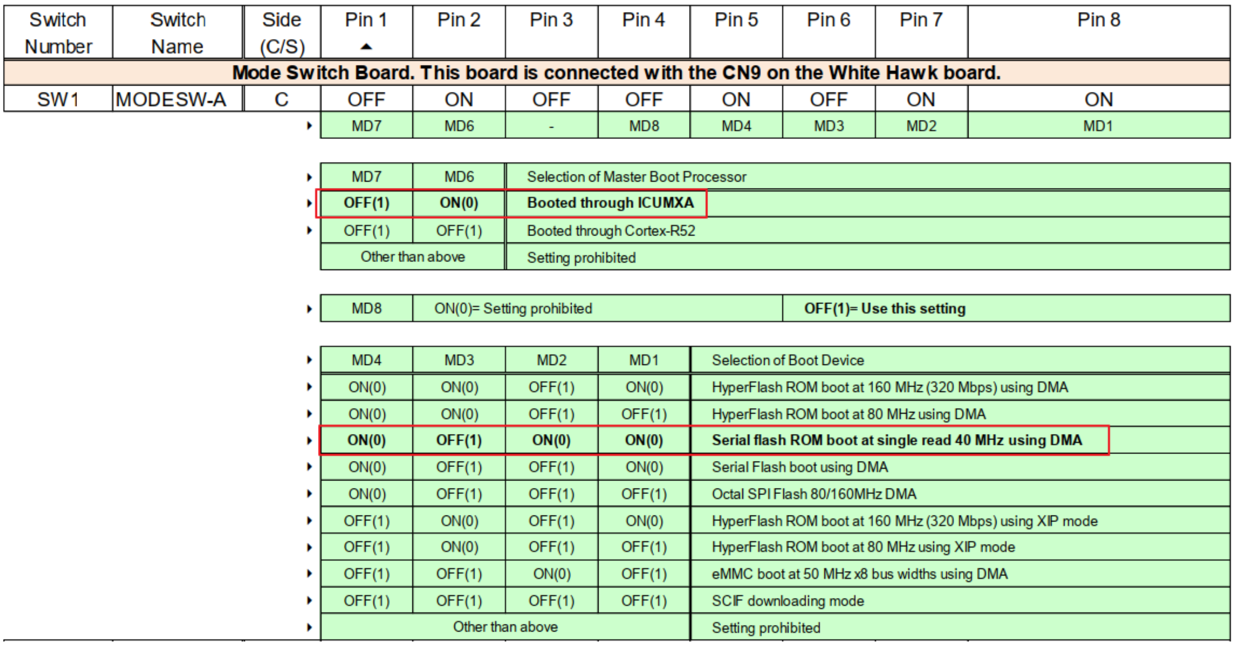


Name	Pin Name	I/O	Function
Mode Pin	MD5	Input	Reserved

MD5	Description
0	Reserved for security.
1	Normal boot.

Name	Pin Name	I/O	Function
Mode pin	MD7 and MD6	Input	Select a master boot processor.

MD[7:6]	Description
B’00	Reserved
B’01	Reserved
B’10	Booted through ICUMX
B’11	Booted through Cortex-R52



Switch Number	Switch Name	Side (C/S)	Pin 1 ▲	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7	Pin 8
Mode Switch Board. This board is connected with the CN9 on the White Hawk board.										
SW1	MODESW-A	C	OFF	ON	OFF	OFF	ON	OFF	ON	ON
			MD7	MD6	-	MD8	MD4	MD3	MD2	MD1

MD7	MD6	Selection of Master Boot Processor
OFF(1)	**ON(0)**	**Booted through ICUMXA**
OFF(1)	OFF(1)	Booted through Cortex-R52
Other than above		Setting prohibited

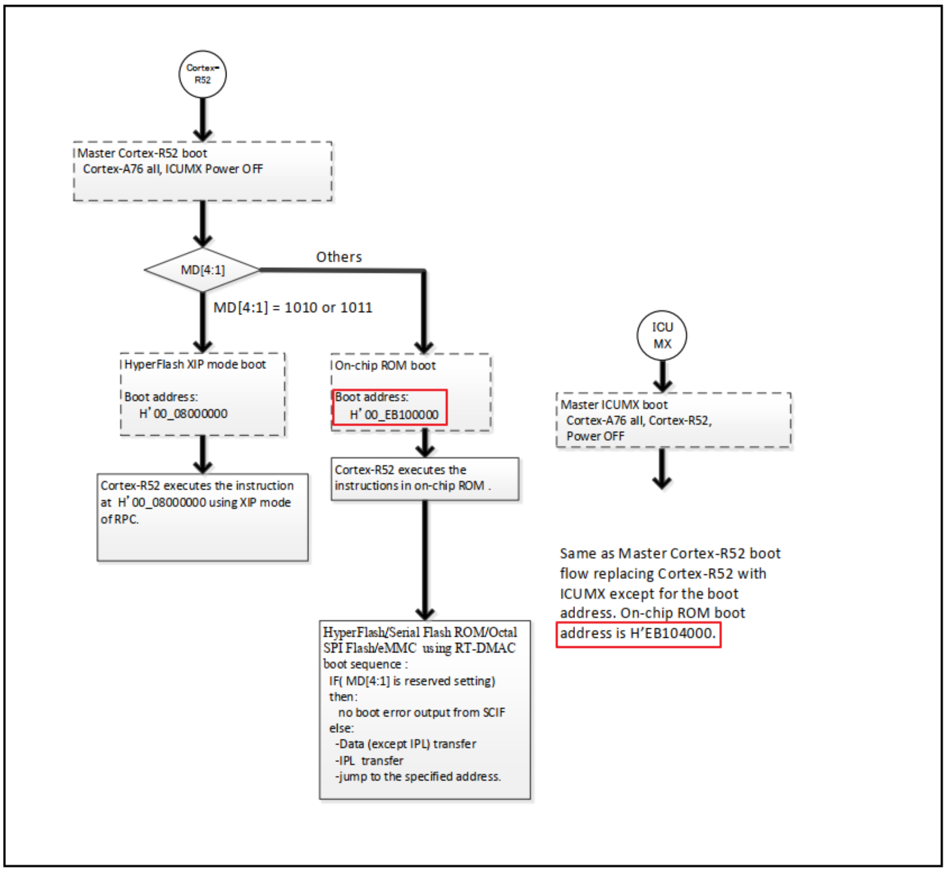
MD8	ON(0)= Setting prohibited	**OFF(1)= Use this setting**

MD4	MD3	MD2	MD1	Selection of Boot Device
ON(0)	ON(0)	OFF(1)	ON(0)	HyperFlash ROM boot at 160 MHz (320 Mbps) using DMA
ON(0)	ON(0)	OFF(1)	OFF(1)	HyperFlash ROM boot at 80 MHz using DMA
ON(0)	**OFF(1)**	**ON(0)**	**ON(0)**	**Serial flash ROM boot at single read 40 MHz using DMA**
ON(0)	OFF(1)	OFF(1)	ON(0)	Serial Flash boot using DMA
ON(0)	OFF(1)	OFF(1)	OFF(1)	Octal SPI Flash 80/160MHz DMA
OFF(1)	ON(0)	OFF(1)	ON(0)	HyperFlash ROM boot at 160 MHz (320 Mbps) using XIP mode
OFF(1)	ON(0)	OFF(1)	OFF(1)	HyperFlash ROM boot at 80 MHz using XIP mode
OFF(1)	OFF(1)	ON(0)	OFF(1)	eMMC boot at 50 MHz x8 bus widths using DMA
OFF(1)	OFF(1)	OFF(1)	OFF(1)	SCIF downloading mode
Other than above				Setting prohibited

(This matrix is in Board manual)

3.1.4 Boot Sequence

HyperFlash Rom 모드가 아닐 경우 boot Rom 영역과 ICUMX 활성화 여부에 따라 Reset Vector 가 0xEB10000(CR52) 또는 0xEB104000(ICUMX) 으로 자동 결정된다.



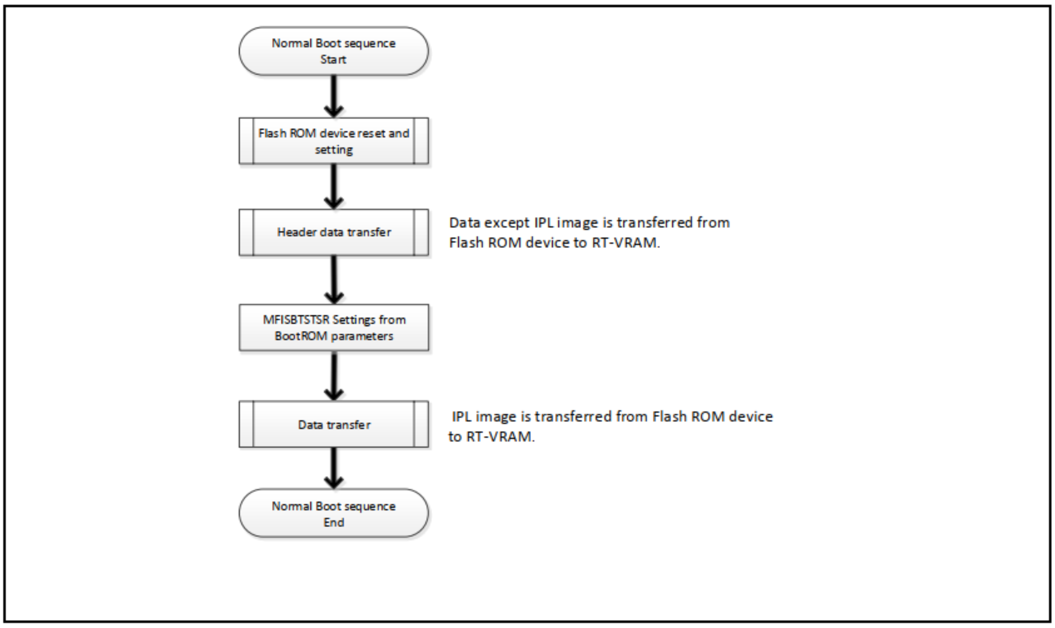
Cortex-R52
Master Cortex-R52 boot
Cortex-A76 all, ICUMX Power OFF
MD[4:1]
Others
MD[4:1] = 1010 or 1011
HyperFlash XIP mode boot
Boot address:
H' 00_08000000
On-chip ROM boot
Boot address:
H' 00_EB100000
Cortex-R52 executes the instruction at H' 00_08000000 using XIP mode of RPC.
Cortex-R52 executes the instructions in on-chip ROM .
HyperFlash/Serial Flash ROM/Octal SPI Flash/eMMC using RT-DMAC boot sequence :
IF(MD[4:1] is reserved setting)
then:
no boot error output from SCIF
else:
-Data (except IPL) transfer
-IPL transfer
-jump to the specified address.
ICU MX
Master ICUMX boot
Cortex-A76 all, Cortex-R52, Power OFF
Same as Master Cortex-R52 boot flow replacing Cortex-R52 with ICUMX except for the boot address. On-chip ROM boot address is H'EB104000.

Figure 31.3 Cortex-R52, ICUMX boot operation overview

3.1.4.1 With Serial Flash

Boot 시퀀스에 따라 부팅이 완료되면 아래와 같이 Boot Rom 에서 header data 및 기타 부팅과 관련된 데이터를 외부 ROM 에서 RT-RAM 으로 복사한다.

기본적인 부팅 설정이 완료되면 IPL 이미지를 RT-VRAM 으로 복사 한 후 IPL 로 분기한다.



Normal Boot sequence Start
Flash ROM device reset and setting
Header data transfer
Data except IPL image is transferred from Flash ROM device to RT-VRAM.
MFISBTSTSR Settings from BootROM parameters
Data transfer
IPL image is transferred from Flash ROM device to RT-VRAM.
Normal Boot sequence End

Figure 31.4 HyperFlash/Octal SPI Flash/Serial Flash using RT-DMAC boot sequence

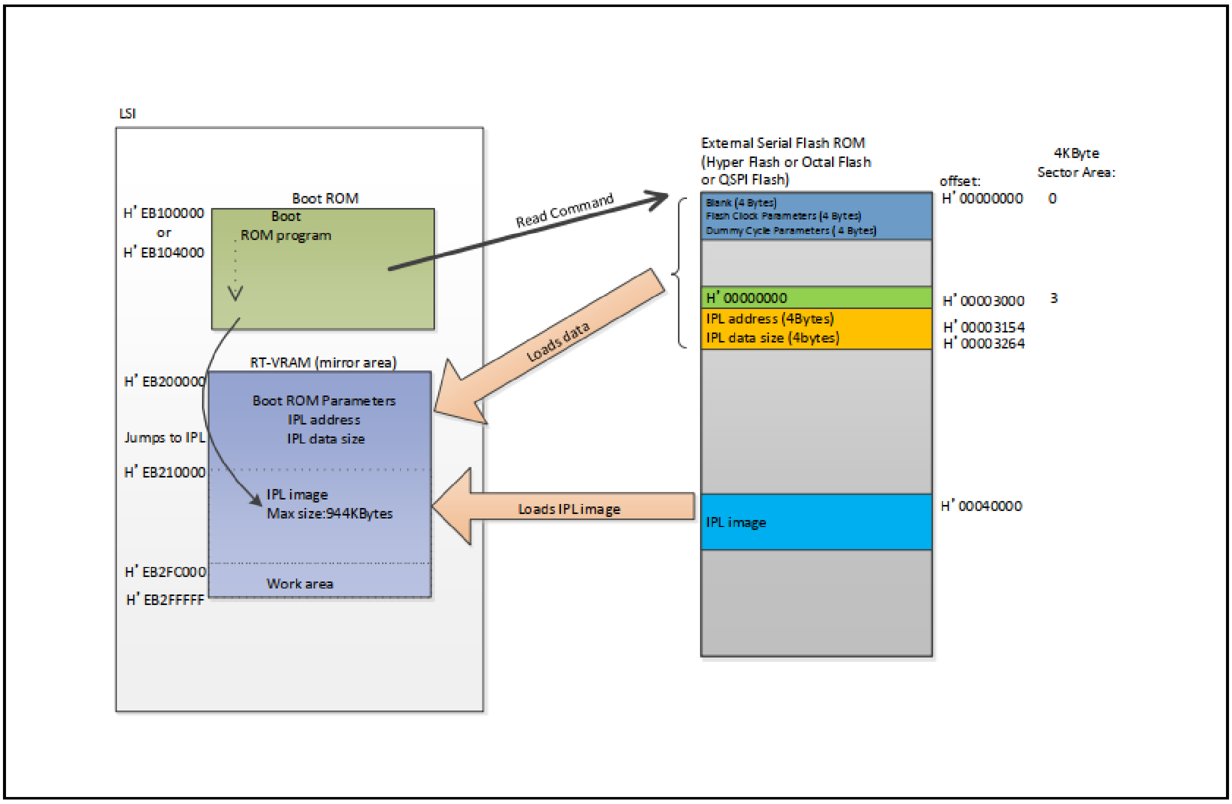


Figure 31.5 IPL transfer and Program image

3.1.4.2 With eMMC

eMMC 의 경우도 Serial Flash 와 동일한 부팅 절차를 갖지만, eMMC 초기화 부분이 추가된다.

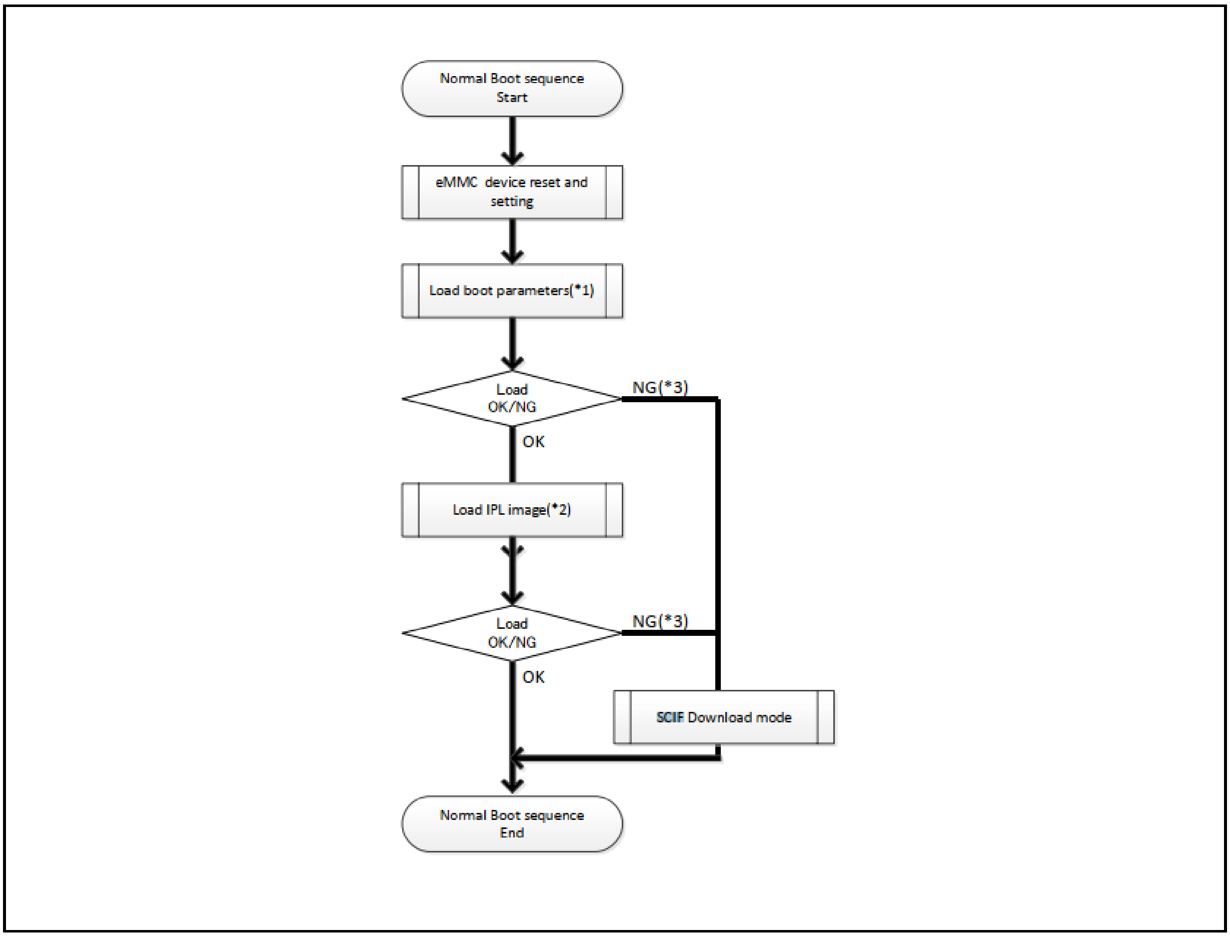
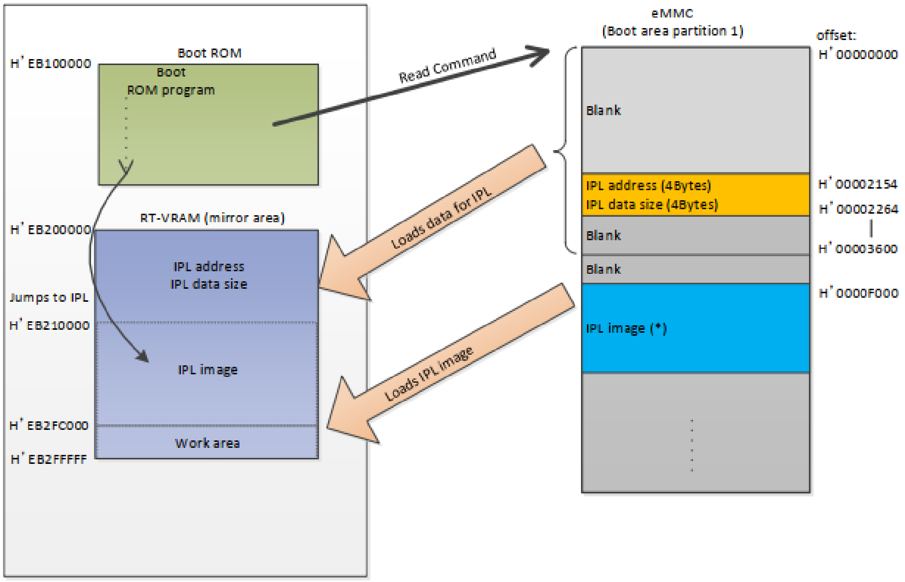


Figure 31.6 eMMC using DMA boot sequence



* : Offset IPL image on eMMC = IPL address on RT-VRAM - H' EB20_1000

Figure 31.7 IPL transfer and Program image(eMMC, Cortex-R52 Boot)

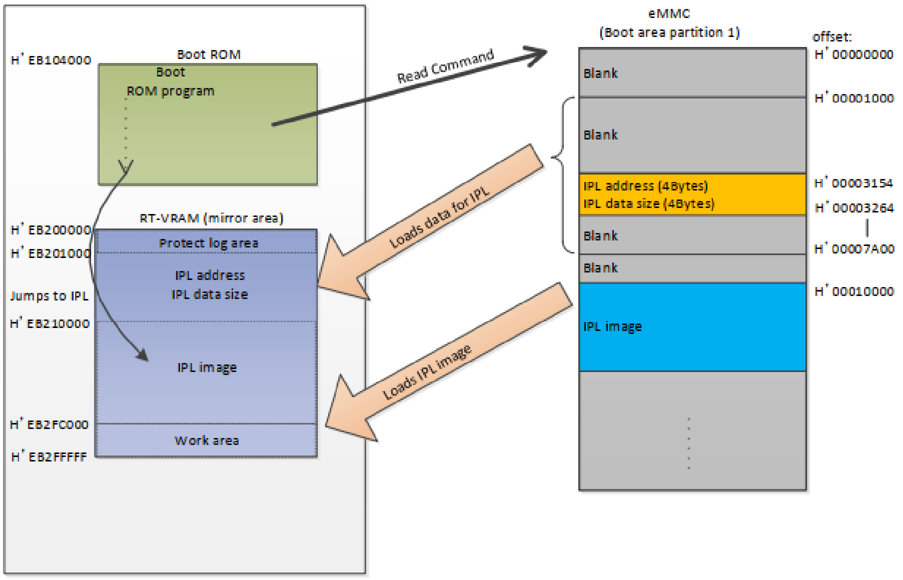
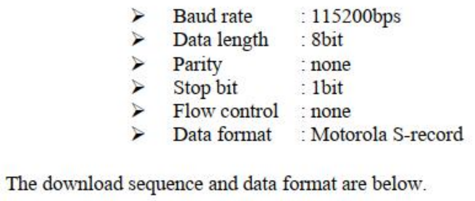


Figure 31.8 IPL transfer and Program image(eMMC, ICUMX Boot)

3.1.5 SCIF Download mode

만약 Booting 이 실패하게 되면, 동작이 멈추는 것이 아니라 SCIF 다운로드 모드로 진입을 하게 된다.

SCIF Download 모드는 PC 에서 RT-VRAM 으로 바로 IPL 및 헤더 정보를 전송 할 수 있다.



➢ Baud rate	: 115200bps
➢ Data length	: 8bit
➢ Parity	: none
➢ Stop bit	: 1bit
➢ Flow control	: none
➢ Data format	: Motorola S-record

The download sequence and data format are below.

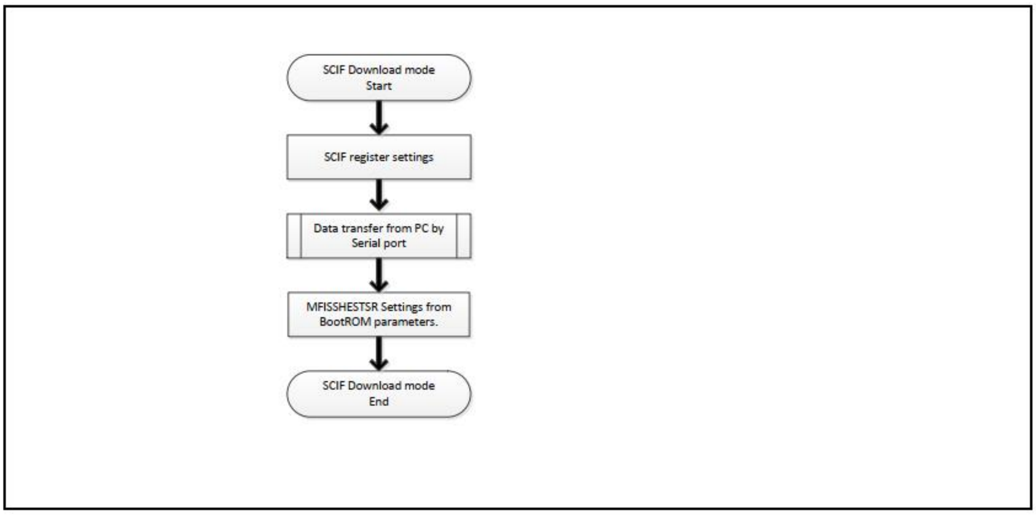

SCIF Download mode Start
SCIF register settings
Data transfer from PC by Serial port
MFISSHESTSR Settings from BootROM parameters.
SCIF Download mode End

Figure 31.9 SCIF download mode sequence

4 SDK 설치

4.1 Main component

SDK 를 설치하기 전에

Windows main installer: rcar-xos_platform-sdk1_v2.18.0_release.exe

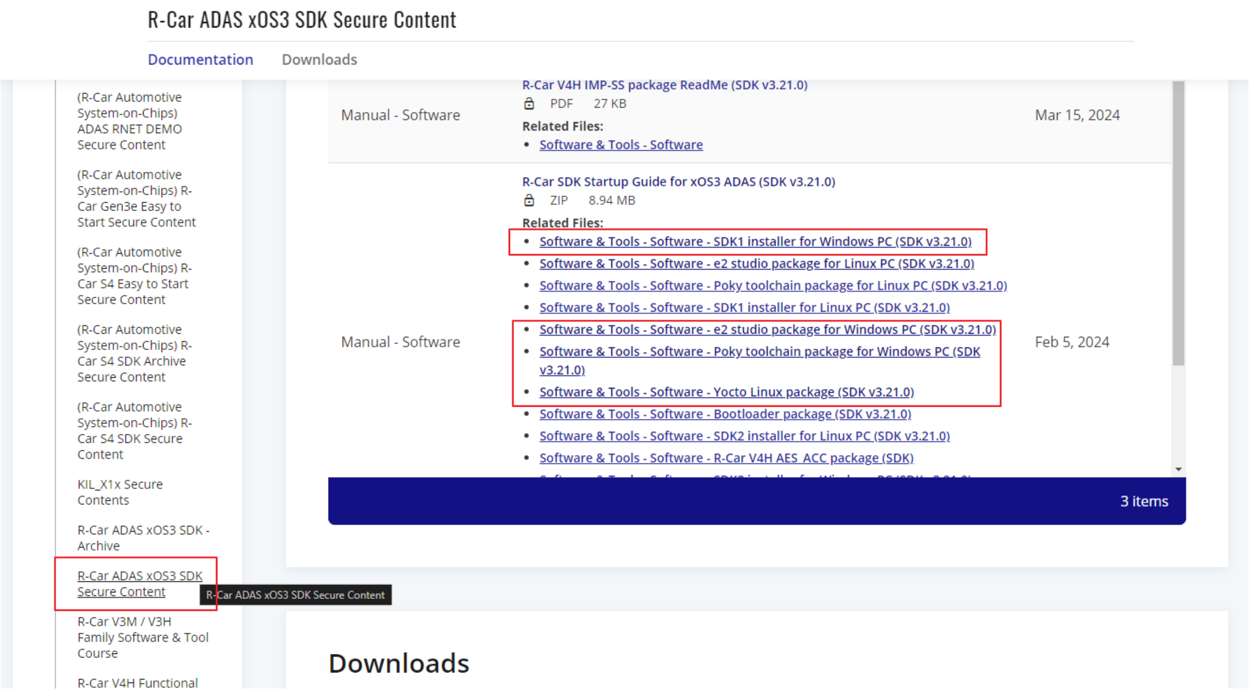
Corresponding tool packages:

e2 studio installer : rcar-xos_tool_e2studio_windows_2022-02.R20220207-2020.tar.gz

Poky toolchain : rcar-xos_tool_poky_toolchain_windows_4.19.2.tar.gz

Corresponding Yocto package: rcar-xos_tool_yocto_linux_4.19.2.tar.gz

URL: https://www.renesas.com/us/en/secure/r-car-adas-xos3-sdk-secure-content#documents



R-Car ADAS xOS3 SDK Secure Content

- (R-Car Automotive System-on-Chips) ADAS RNET DEMO Secure Content
- (R-Car Automotive System-on-Chips) R-Car Gen3e Easy to Start Secure Content
- (R-Car Automotive System-on-Chips) R-Car S4 Easy to Start Secure Content
- (R-Car Automotive System-on-Chips) R-Car S4 SDK Archive Secure Content
- (R-Car Automotive System-on-Chips) R-Car S4 SDK Secure Content
- KIL_X1x Secure Contents
- R-Car ADAS xOS3 SDK - Archive
- R-Car ADAS xOS3 SDK Secure Content
- R-Car V3M / V3H Family Software & Tool Course
- R-Car V4H Functional

R-Car ADAS xOS3 SDK Secure Content

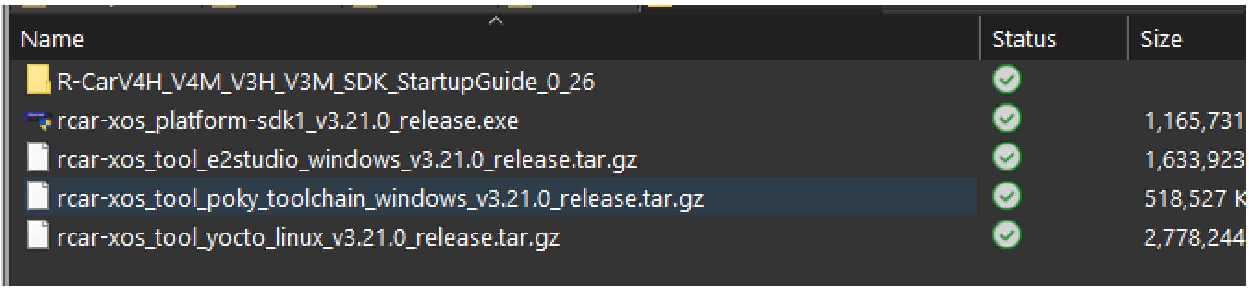
Manual - Software	R-Car V4H IMP-SS package ReadMe (SDK v3.21.0) PDF 27 KB **Related Files:** • Software & Tools - Software	Mar 15, 2024
Manual - Software	R-Car SDK Startup Guide for xOS3 ADAS (SDK v3.21.0) ZIP 8.94 MB **Related Files:** • Software & Tools - Software - SDK1 installer for Windows PC (SDK v3.21.0) • Software & Tools - Software - e2 studio package for Linux PC (SDK v3.21.0) • Software & Tools - Software - Poky toolchain package for Linux PC (SDK v3.21.0) • Software & Tools - Software - SDK1 installer for Linux PC (SDK v3.21.0) • Software & Tools - Software - e2 studio package for Windows PC (SDK v3.21.0) • Software & Tools - Software - Poky toolchain package for Windows PC (SDK v3.21.0) • Software & Tools - Software - Yocto Linux package (SDK v3.21.0) • Software & Tools - Software - Bootloader package (SDK v3.21.0) • Software & Tools - Software - SDK2 installer for Linux PC (SDK v3.21.0) • Software & Tools - Software - R-Car V4H AES_ACC package (SDK)	Feb 5, 2024

3 items

Downloads

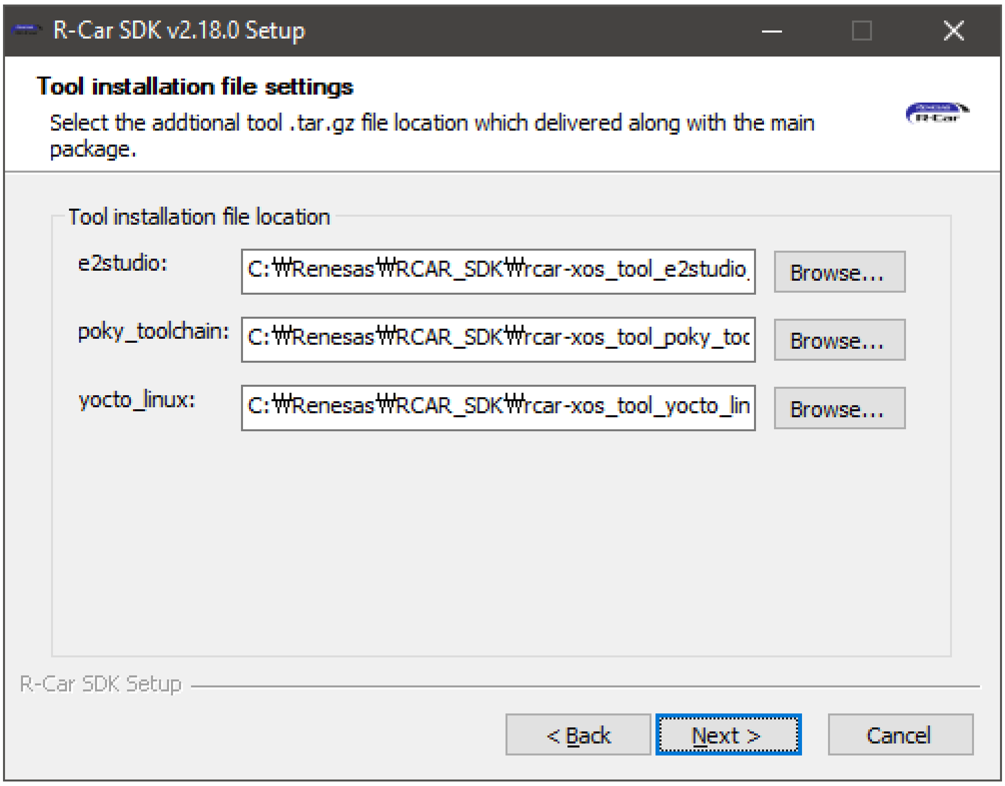
4.2 Installation

모든 패키지를 압축된 상태에서 SDK 설치파일과 동일한 폴더에 복사해서 넣는다.



Name	Status	Size
R-CarV4H_V4M_V3H_V3M_SDK_StartupGuide_0_26	✓	
rcar-xos_platform-sdk1_v3.21.0_release.exe	✓	1,165,731
rcar-xos_tool_e2studio_windows_v3.21.0_release.tar.gz	✓	1,633,923
rcar-xos_tool_poky_toolchain_windows_v3.21.0_release.tar.gz	✓	518,527 K
rcar-xos_tool_yocto_linux_v3.21.0_release.tar.gz	✓	2,778,244

폴더에 정상적인 설치파일들이 있으면 next 를 누른다.



R-Car SDK v2.18.0 Setup

Tool installation file settings

Select the addtional tool .tar.gz file location which delivered along with the main package.

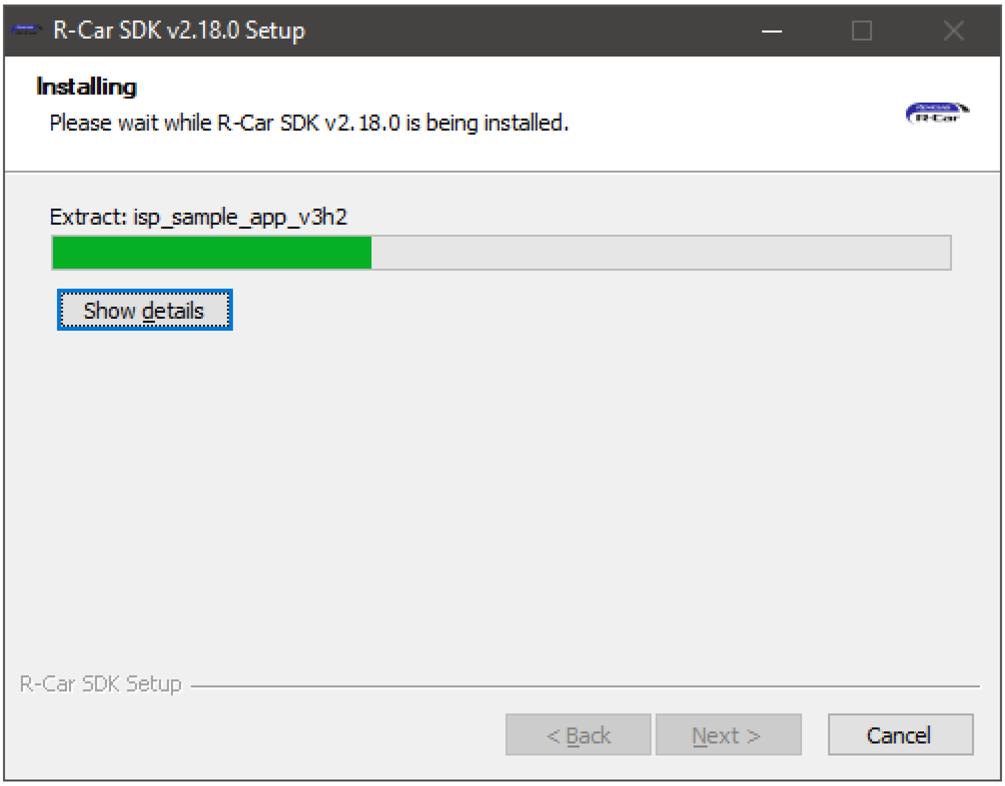
Tool installation file location

e2studio: C:₩Renesas₩RCAR_SDK₩rcar-xos_tool_e2studio_

poky_toolchain: C:₩Renesas₩RCAR_SDK₩rcar-xos_tool_poky_toc

yocto_linux: C:₩Renesas₩RCAR_SDK₩rcar-xos_tool_yocto_lin

설치가 아래와 같이 진행된다.

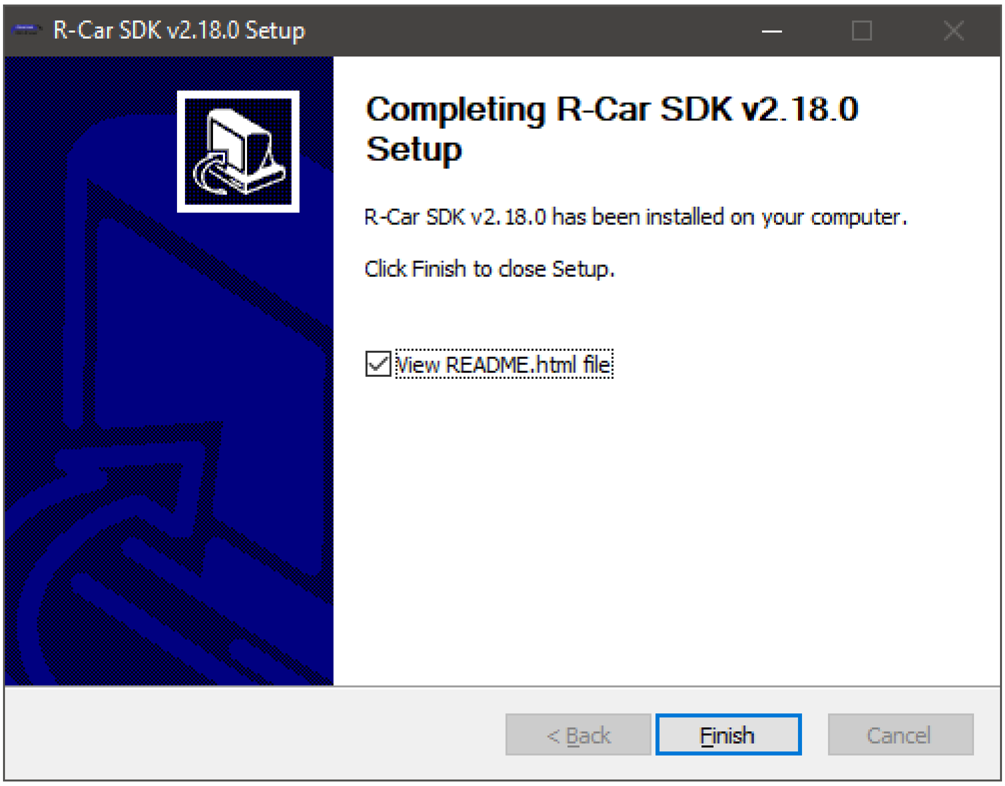


R-Car SDK v2.18.0 Setup

Installing

Please wait while R-Car SDK v2.18.0 is being installed.

Extract: isp_sample_app_v3h2



R-Car SDK v2.18.0 Setup

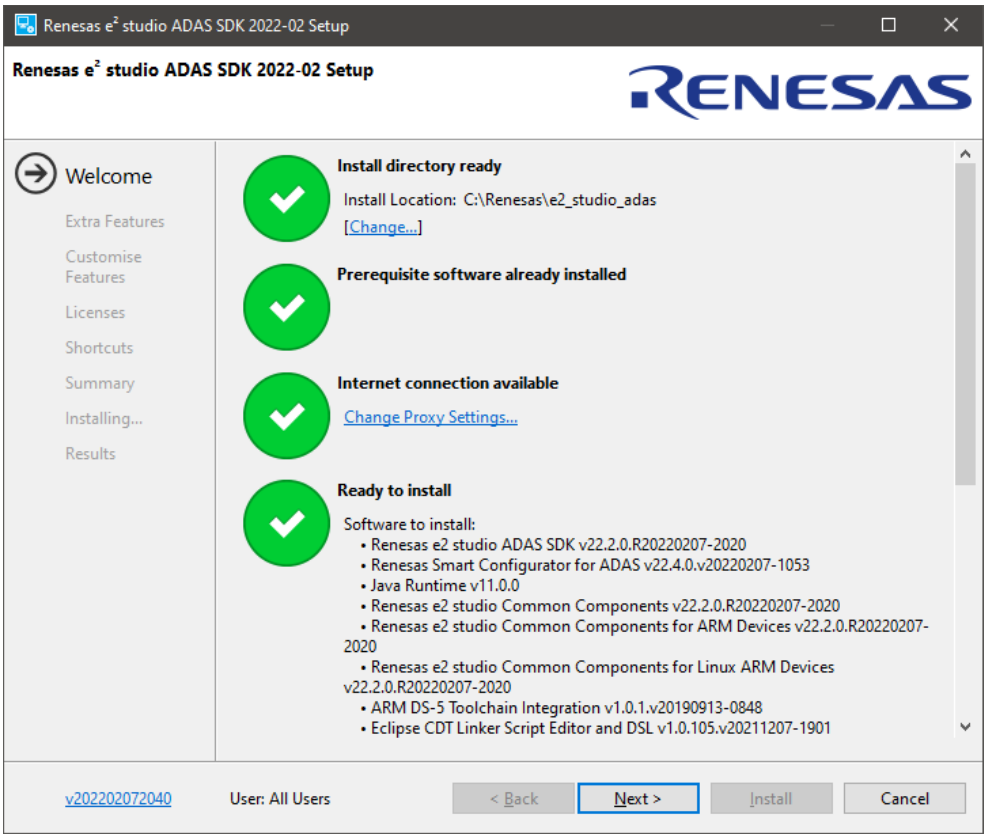
Completing R-Car SDK v2.18.0 Setup

R-Car SDK v2.18.0 has been installed on your computer.

Click Finish to close Setup.

☑ View README.html file

설치가 완료되면 자동으로 e2 Studio 를 설치하여 준다.



Renesas e² studio ADAS SDK 2022-02 Setup

Renesas e² studio ADAS SDK 2022-02 Setup

Install directory ready

Install Location: C:\Renesas\e2_studio_adas

[Change...]

Prerequisite software already installed

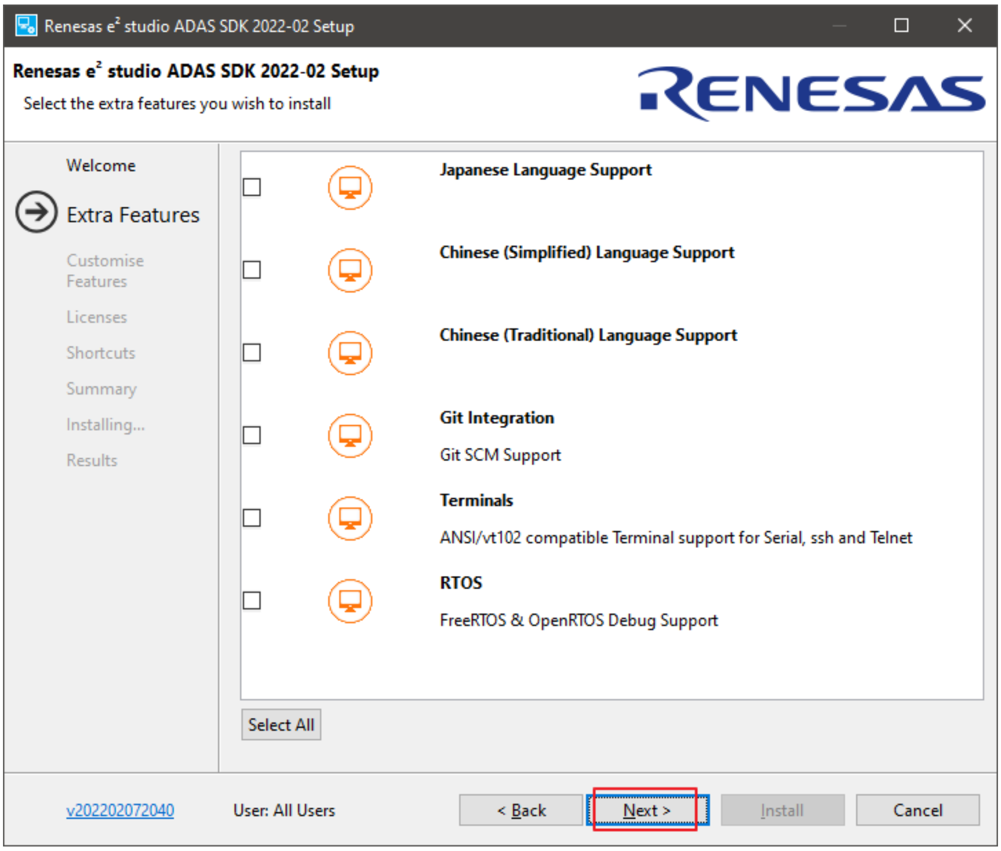
Internet connection available

Change Proxy Settings...

Ready to install

Software to install:
- Renesas e2 studio ADAS SDK v22.2.0.R20220207-2020
- Renesas Smart Configurator for ADAS v22.4.0.v20220207-1053
- Java Runtime v11.0.0
- Renesas e2 studio Common Components v22.2.0.R20220207-2020
- Renesas e2 studio Common Components for ARM Devices v22.2.0.R20220207-2020
- Renesas e2 studio Common Components for Linux ARM Devices v22.2.0.R20220207-2020
- ARM DS-5 Toolchain Integration v1.0.1.v20190913-0848
- Eclipse CDT Linker Script Editor and DSL v1.0.105.v20211207-1901

v202202072040 User: All Users



Renesas e² studio ADAS SDK 2022-02 Setup

Renesas e² studio ADAS SDK 2022-02 Setup

Select the extra features you wish to install

☐ **Japanese Language Support**

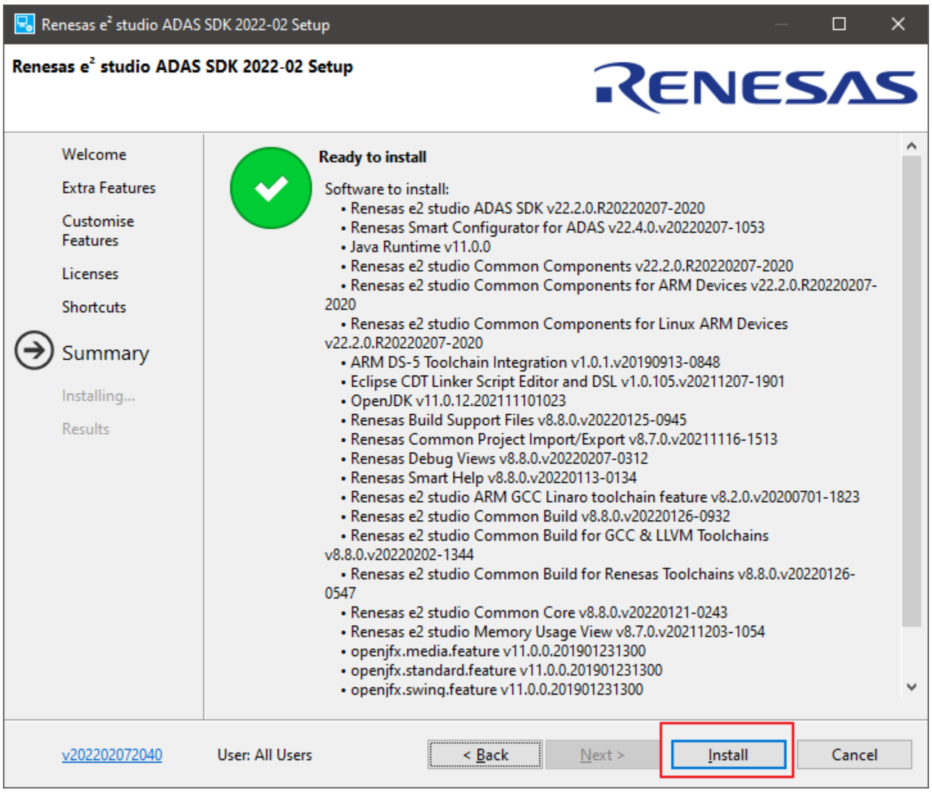
☐ **Chinese (Simplified) Language Support**

☐ **Chinese (Traditional) Language Support**

☐ **Git Integration**
Git SCM Support

☐ **Terminals**
ANSI/vt102 compatible Terminal support for Serial, ssh and Telnet

☐ **RTOS**
FreeRTOS & OpenRTOS Debug Support



Renesas e² studio ADAS SDK 2022-02 Setup

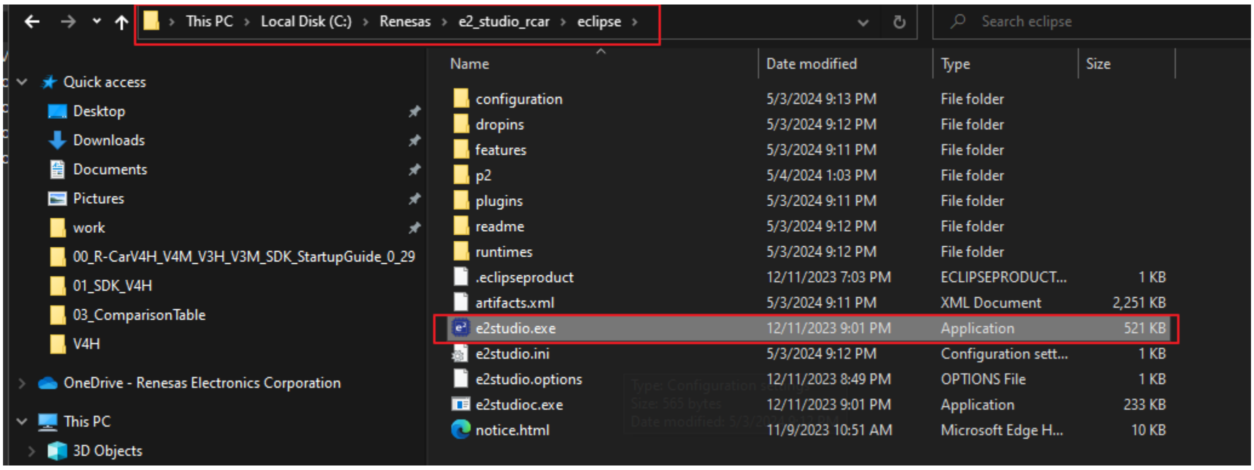
Renesas e² studio ADAS SDK 2022-02 Setup

Ready to install

Software to install:
- Renesas e2 studio ADAS SDK v22.2.0.R20220207-2020
- Renesas Smart Configurator for ADAS v22.4.0.v20220207-1053
- Java Runtime v11.0.0
- Renesas e2 studio Common Components v22.2.0.R20220207-2020
- Renesas e2 studio Common Components for ARM Devices v22.2.0.R20220207-2020
- Renesas e2 studio Common Components for Linux ARM Devices v22.2.0.R20220207-2020
- ARM DS-5 Toolchain Integration v1.0.1.v20190913-0848
- Eclipse CDT Linker Script Editor and DSL v1.0.105.v20211207-1901
- OpenJDK v11.0.12.202111101023
- Renesas Build Support Files v8.8.0.v20220125-0945
- Renesas Common Project Import/Export v8.7.0.v20211116-1513
- Renesas Debug Views v8.8.0.v20220207-0312
- Renesas Smart Help v8.8.0.v20220113-0134
- Renesas e2 studio ARM GCC Linaro toolchain feature v8.2.0.v20200701-1823
- Renesas e2 studio Common Build v8.8.0.v20220126-0932
- Renesas e2 studio Common Build for GCC & LLVM Toolchains v8.8.0.v20220202-1344
- Renesas e2 studio Common Build for Renesas Toolchains v8.8.0.v20220126-0547
- Renesas e2 studio Common Core v8.8.0.v20220121-0243
- Renesas e2 studio Memory Usage View v8.7.0.v20211203-1054
- openjfx.media.feature v11.0.0.201901231300
- openjfx.standard.feature v11.0.0.201901231300
- openjfx.swing.feature v11.0.0.201901231300

5 E2Studio 설정

설치가 완료되면 아래 폴더로 이동하여 e2Studio 를 실행하여 준다.

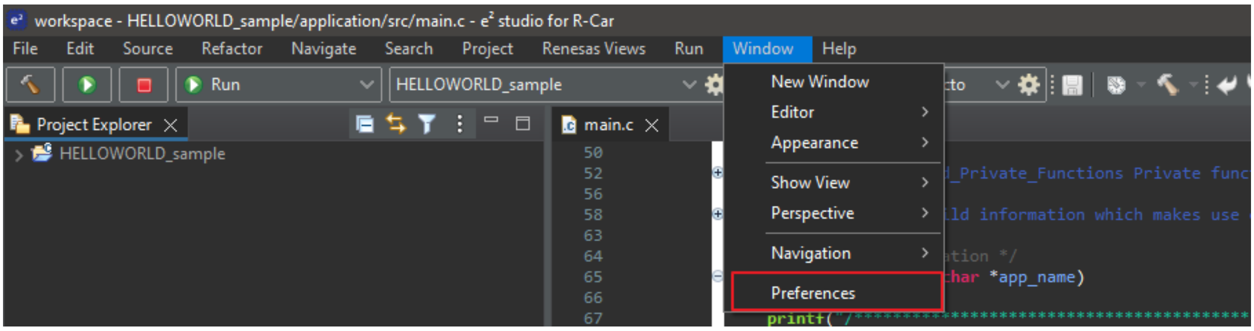


This PC › Local Disk (C:) › Renesas › e2_studio_rcar › eclipse ›

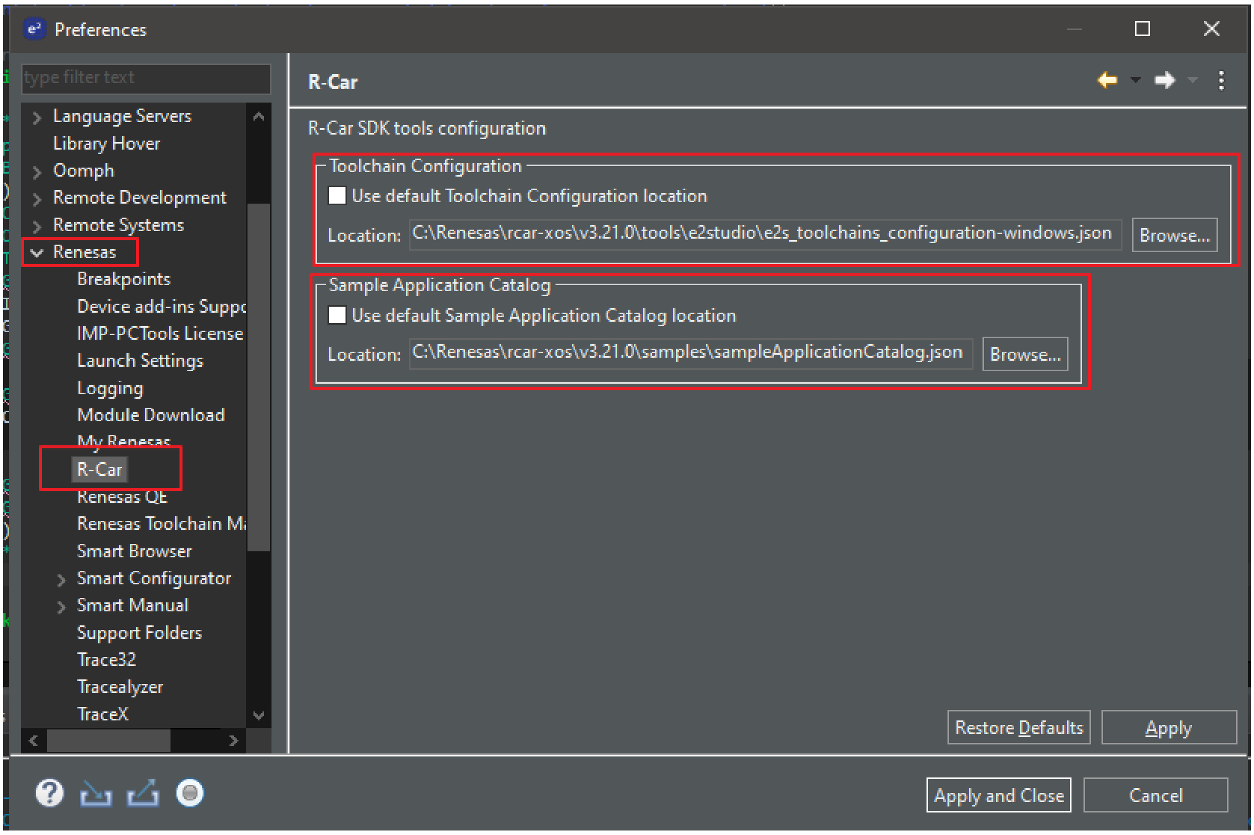
Name	Date modified	Type	Size
configuration	5/3/2024 9:13 PM	File folder	
dropins	5/3/2024 9:12 PM	File folder	
features	5/3/2024 9:11 PM	File folder	
p2	5/4/2024 1:03 PM	File folder	
plugins	5/3/2024 9:11 PM	File folder	
readme	5/3/2024 9:12 PM	File folder	
runtimes	5/3/2024 9:12 PM	File folder	
.eclipseproduct	12/11/2023 7:03 PM	ECLIPSEPRODUCT...	1 KB
artifacts.xml	5/3/2024 9:11 PM	XML Document	2,251 KB
e2studio.exe	12/11/2023 9:01 PM	Application	521 KB
e2studio.ini	5/3/2024 9:12 PM	Configuration sett...	1 KB
e2studio.options	12/11/2023 8:49 PM	OPTIONS File	1 KB
e2studioc.exe	12/11/2023 9:01 PM	Application	233 KB
notice.html	11/9/2023 10:51 AM	Microsoft Edge H...	10 KB

처음 실행이 완료되면 RCAR 의 기본샘플을 불러올 수 없다. 그래서 샘플 프로젝트의 Load 와 컴파일 디버깅을 위하여 아래 설정을 진행한다.

해당 설정은 매뉴얼에 나와 있지 않아, 실수하기가 쉬우니 반드시 주의 바람.



workspace - HELLOWORLD_sample/application/src/main.c - e² studio for R-Car
File Edit Source Refactor Navigate Search Project Renesas Views Run Window Help
Run
HELLOWORLD_sample
Project Explorer
HELLOWORLD_sample
main.c
New Window
Editor
Appearance
Show View
Perspective
Navigation
Preferences



Preferences

R-Car

R-Car SDK tools configuration

Toolchain Configuration

□ Use default Toolchain Configuration location

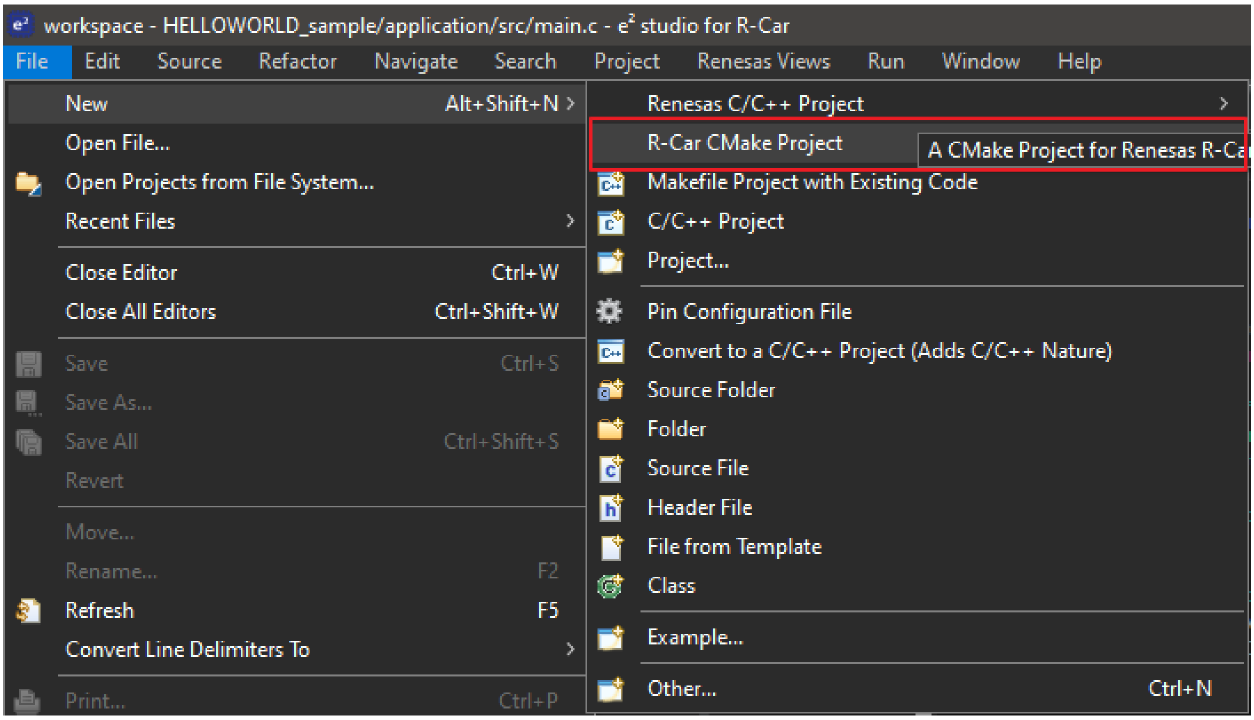
Location: C:\Renesas\rcar-xos\v3.21.0\tools\e2studio\e2s_toolchains_configuration-windows.json

Sample Application Catalog

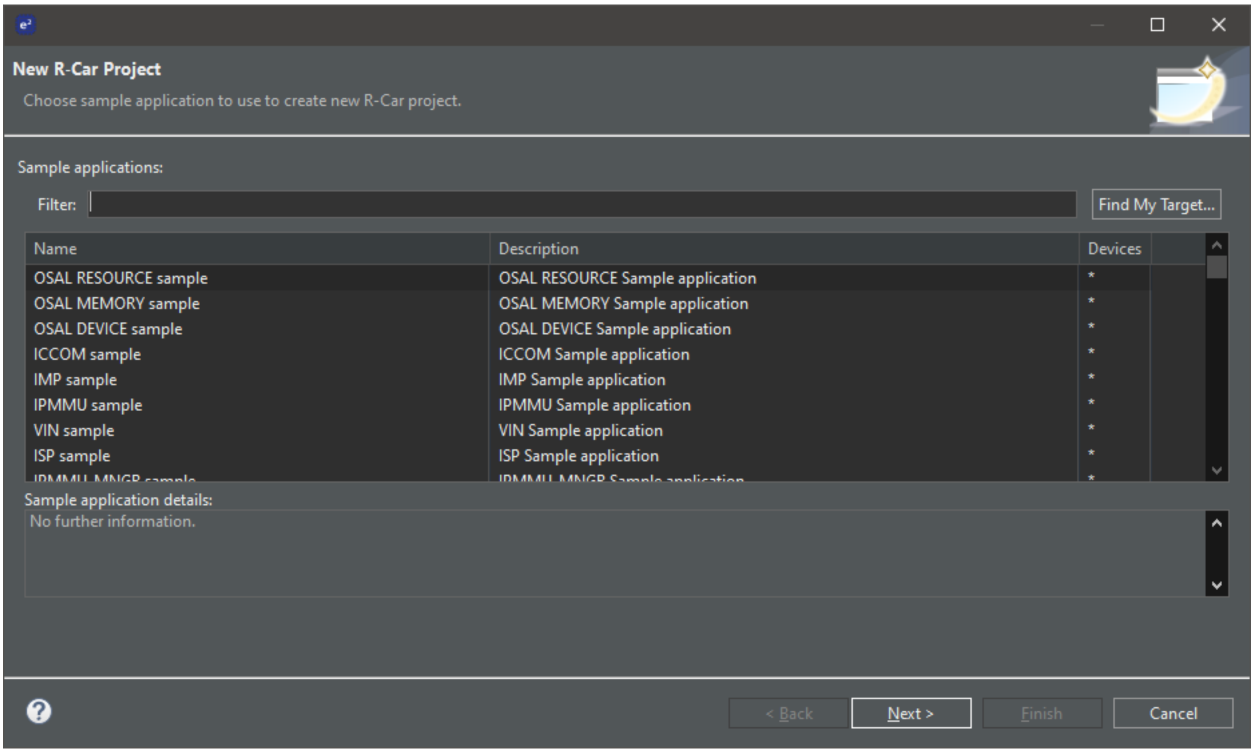
□ Use default Sample Application Catalog location

Location: C:\Renesas\rcar-xos\v3.21.0\samples\sampleApplicationCatalog.json

정상적으로 설정을 완료하면 아래와 같이 New RCAR Cmake 프로젝트 생성시 샘플 프로젝트 목록을 확인할 수 있다.



workspace - HELLOWORLD_sample/application/src/main.c - e² studio for R-Car



New R-Car Project

Choose sample application to use to create new R-Car project.

Sample applications:

Filter:

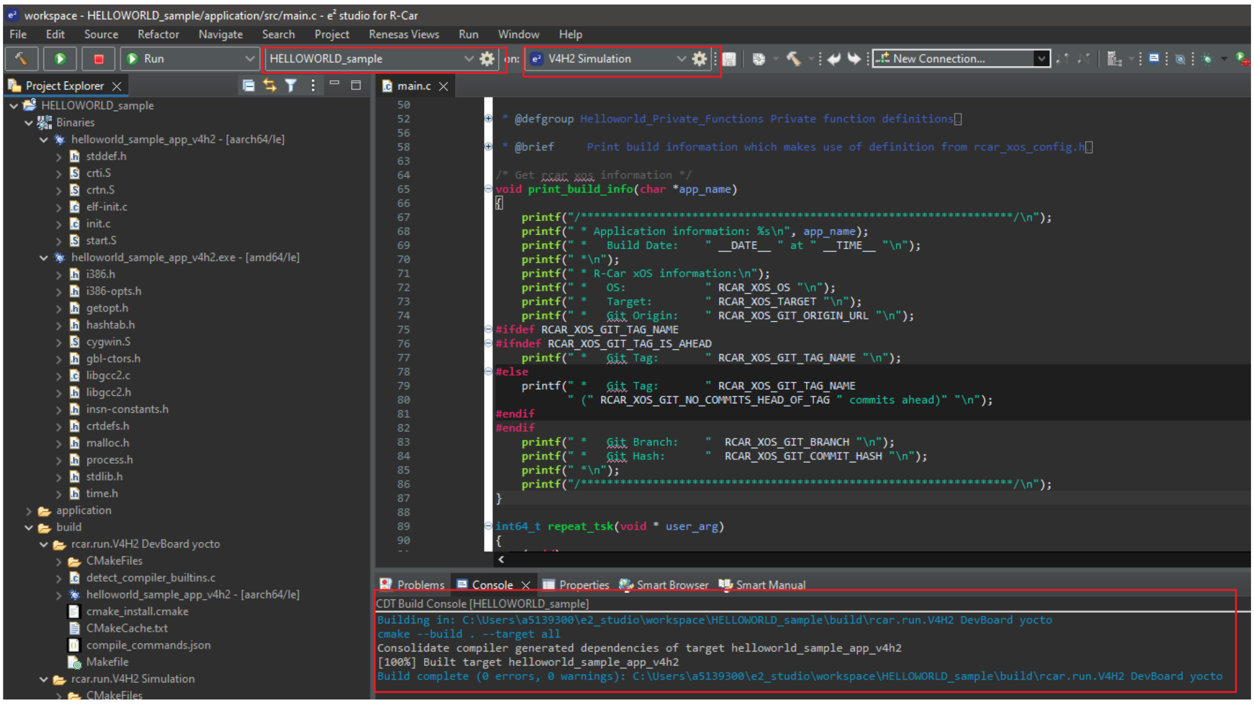
Name	Description	Devices
OSAL RESOURCE sample	OSAL RESOURCE Sample application	*
OSAL MEMORY sample	OSAL MEMORY Sample application	*
OSAL DEVICE sample	OSAL DEVICE Sample application	*
ICCOM sample	ICCOM Sample application	*
IMP sample	IMP Sample application	*
IPMMU sample	IPMMU Sample application	*
VIN sample	VIN Sample application	*
ISP sample	ISP Sample application	*
IPMMU MNGR sample	IPMMU MNGR Sample application	*

Sample application details:

No further information.

처음에는 Hello World 프로젝트를 불러와서 컴파일 및 Software 디버깅을 진행하여 본다.

컴파일 진행 예시



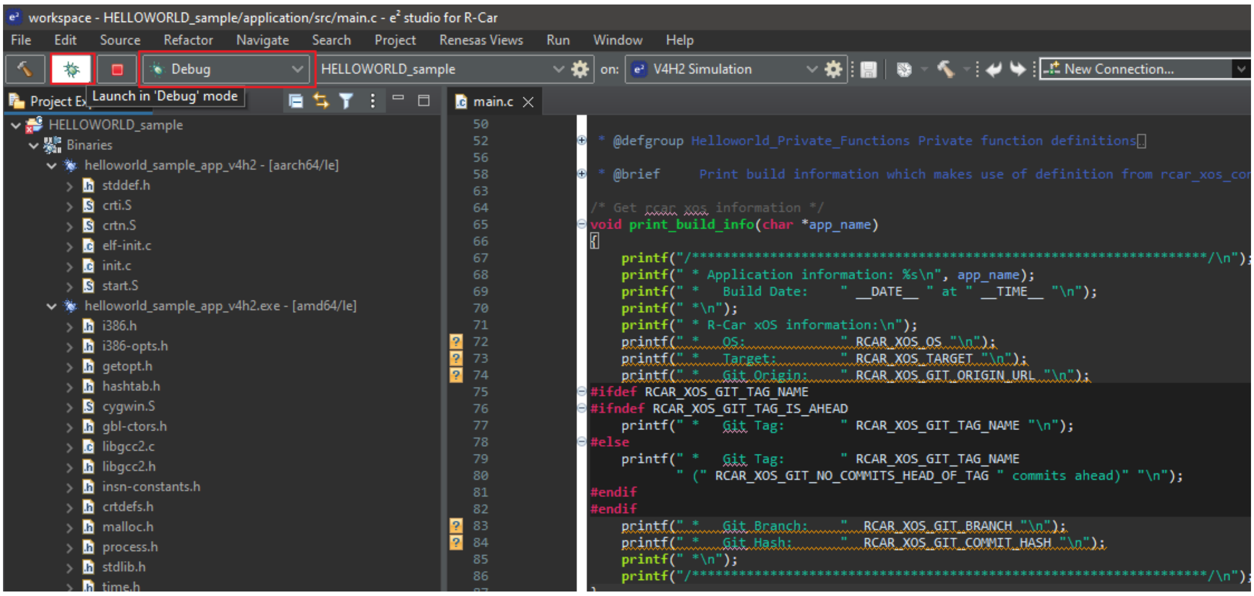
workspace - HELLOWORLD_sample/application/src/main.c - e² studio for R-Car

```
 * @defgroup Helloworld_Private_Functions Private function definitions

 * @brief     Print build information which makes use of definition from rcar_xos_config.h

/* Get rcar xos information */
void print_build_info(char *app_name)
{
    printf("/*****************************************************************/\n");
    printf(" * Application information: %s\n", app_name);
    printf(" *   Build Date:     " __DATE__ " at " __TIME__ "\n");
    printf(" *\n");
    printf(" * R-Car xOS information:\n");
    printf(" *   OS:             " RCAR_XOS_OS "\n");
    printf(" *   Target:         " RCAR_XOS_TARGET "\n");
    printf(" *   Git Origin:     " RCAR_XOS_GIT_ORIGIN_URL "\n");
#ifdef RCAR_XOS_GIT_TAG_NAME
#ifndef RCAR_XOS_GIT_TAG_IS_AHEAD
    printf(" *   Git Tag:        " RCAR_XOS_GIT_TAG_NAME "\n");
#else
    printf(" *   Git Tag:        " RCAR_XOS_GIT_TAG_NAME
           " (" RCAR_XOS_GIT_NO_COMMITS_HEAD_OF_TAG " commits ahead)" "\n");
#endif
#endif
    printf(" *   Git Branch:     "  RCAR_XOS_GIT_BRANCH "\n");
    printf(" *   Git Hash:       "  RCAR_XOS_GIT_COMMIT_HASH "\n");
    printf(" *\n");
    printf("/*****************************************************************/\n");
}

int64_t repeat_tsk(void * user_arg)
{
```

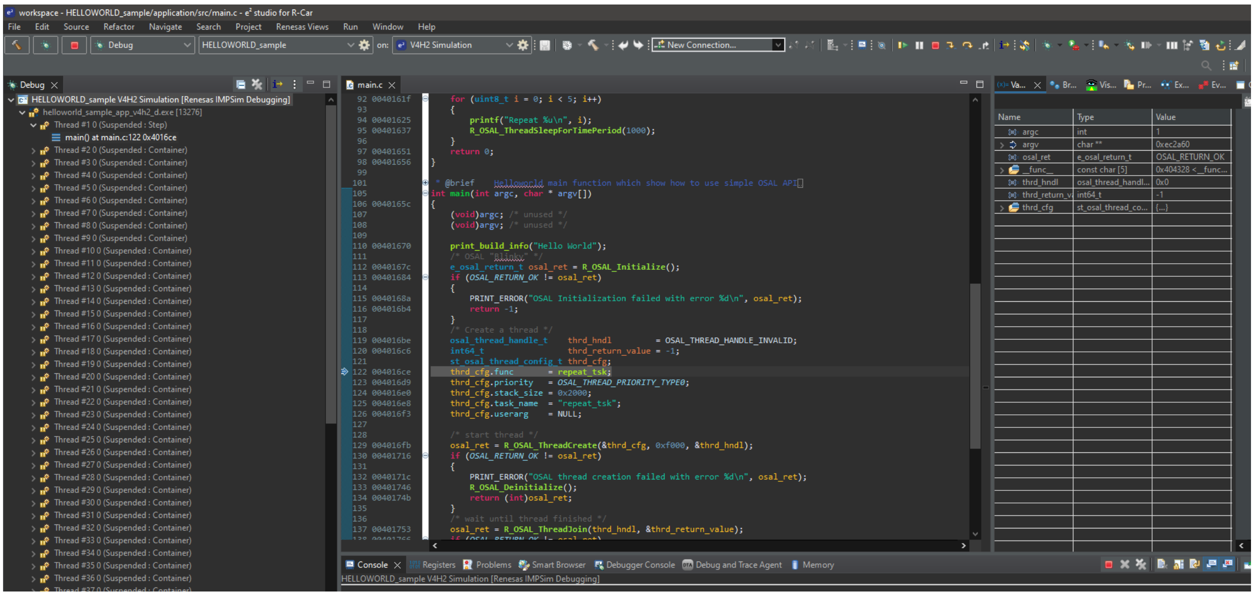
CDT Build Console [HELLOWORLD_sample]

```
Building in: C:\Users\a5139300\e2_studio\workspace\HELLOWORLD_sample\build\rcar.run.V4H2 DevBoard yocto
cmake --build . --target all
Consolidate compiler generated dependencies of target helloworld_sample_app_v4h2
[100%] Built target helloworld_sample_app_v4h2
Build complete (0 errors, 0 warnings): C:\Users\a5139300\e2_studio\workspace\HELLOWORLD_sample\build\rcar.run.V4H2 DevBoard yocto
```

디버깅 진행 예시



```
 * @defgroup Helloworld_Private_Functions Private function definitions
 * @brief     Print build information which makes use of definition from rcar_xos_co
/* Get rcar xos information */
void print_build_info(char *app_name)
{
    printf("/*******************************************************************/\n");
    printf(" * Application information: %s\n", app_name);
    printf(" *   Build Date:    " __DATE__ " at " __TIME__ "\n");
    printf(" *\n");
    printf(" * R-Car xOS information:\n");
    printf(" *   OS:            " RCAR_XOS_OS "\n");
    printf(" *   Target:        " RCAR_XOS_TARGET "\n");
    printf(" *   Git Origin:    " RCAR_XOS_GIT_ORIGIN_URL "\n");
#ifdef RCAR_XOS_GIT_TAG_NAME
#ifndef RCAR_XOS_GIT_TAG_IS_AHEAD
    printf(" *   Git Tag:       " RCAR_XOS_GIT_TAG_NAME "\n");
#else
    printf(" *   Git Tag:       " RCAR_XOS_GIT_TAG_NAME
           " (" RCAR_XOS_GIT_NO_COMMITS_HEAD_OF_TAG " commits ahead)" "\n");
#endif
#endif
    printf(" *   Git Branch:     " RCAR_XOS_GIT_BRANCH "\n");
    printf(" *   Git Hash:       " RCAR_XOS_GIT_COMMIT_HASH "\n");
    printf(" *\n");
    printf("/*******************************************************************/\n");
```



workspace - HELLOWORLD_sample/application/src/main.c - e² studio for R-Car
File Edit Source Refactor Navigate Search Project Renesas Views Run Window Help
Debug
HELLOWORLD_sample
on: V4H2 Simulation
New Connection...
Debug
main.c
Va... Br... Vis... Pr... Ex... Ev...
Console Registers Problems Smart Browser Debugger Console Debug and Trace Agent Memory

```
HELLOWORLD_sample V4H2 Simulation [Renesas IMPSim Debugging]
  helloworld_sample_app_v4h2_d.exe [13276]
    Thread #1 0 (Suspended : Step)
      main() at main.c:122 0x4016ce
    Thread #2 0 (Suspended : Container)
    Thread #3 0 (Suspended : Container)
    Thread #4 0 (Suspended : Container)
    Thread #5 0 (Suspended : Container)
    Thread #6 0 (Suspended : Container)
    Thread #7 0 (Suspended : Container)
    Thread #8 0 (Suspended : Container)
    Thread #9 0 (Suspended : Container)
    Thread #10 0 (Suspended : Container)
    Thread #11 0 (Suspended : Container)
    Thread #12 0 (Suspended : Container)
    Thread #13 0 (Suspended : Container)
    Thread #14 0 (Suspended : Container)
    Thread #15 0 (Suspended : Container)
    Thread #16 0 (Suspended : Container)
    Thread #17 0 (Suspended : Container)
    Thread #18 0 (Suspended : Container)
    Thread #19 0 (Suspended : Container)
    Thread #20 0 (Suspended : Container)
    Thread #21 0 (Suspended : Container)
    Thread #22 0 (Suspended : Container)
    Thread #23 0 (Suspended : Container)
    Thread #24 0 (Suspended : Container)
    Thread #25 0 (Suspended : Container)
    Thread #26 0 (Suspended : Container)
    Thread #27 0 (Suspended : Container)
    Thread #28 0 (Suspended : Container)
    Thread #29 0 (Suspended : Container)
    Thread #30 0 (Suspended : Container)
    Thread #31 0 (Suspended : Container)
    Thread #32 0 (Suspended : Container)
    Thread #33 0 (Suspended : Container)
    Thread #34 0 (Suspended : Container)
    Thread #35 0 (Suspended : Container)
    Thread #36 0 (Suspended : Container)
```

```
    for (uint8_t i = 0; i < 5; i++)
    {
        printf("Repeat %u\n", i);
        R_OSAL_ThreadSleepForTimePeriod(1000);
    }
    return 0;
}

 * @brief    Helloworld main function which show how to use simple OSAL API
int main(int argc, char * argv[])
{
    (void)argc; /* unused */
    (void)argv; /* unused */

    print_build_info("Hello World");
    /* OSAL "Blinky" */
    e_osal_return_t osal_ret = R_OSAL_Initialize();
    if (OSAL_RETURN_OK != osal_ret)
    {
        PRINT_ERROR("OSAL Initialization failed with error %d\n", osal_ret);
        return -1;
    }
    /* Create a thread */
    osal_thread_handle_t    thrd_hndl         = OSAL_THREAD_HANDLE_INVALID;
    int64_t                 thrd_return_value = -1;
    st_osal_thread_config_t thrd_cfg;
    thrd_cfg.func       = repeat_tsk;
    thrd_cfg.priority   = OSAL_THREAD_PRIORITY_TYPE0;
    thrd_cfg.stack_size = 0x2000;
    thrd_cfg.task_name  = "repeat_tsk";
    thrd_cfg.userarg    = NULL;

    /* start thread */
    osal_ret = R_OSAL_ThreadCreate(&thrd_cfg, 0xf000, &thrd_hndl);
    if (OSAL_RETURN_OK != osal_ret)
    {
        PRINT_ERROR("OSAL thread creation failed with error %d\n", osal_ret);
        R_OSAL_Deinitialize();
        return (int)osal_ret;
    }
    /* wait until thread finished */
    osal_ret = R_OSAL_ThreadJoin(thrd_hndl, &thrd_return_value);
```

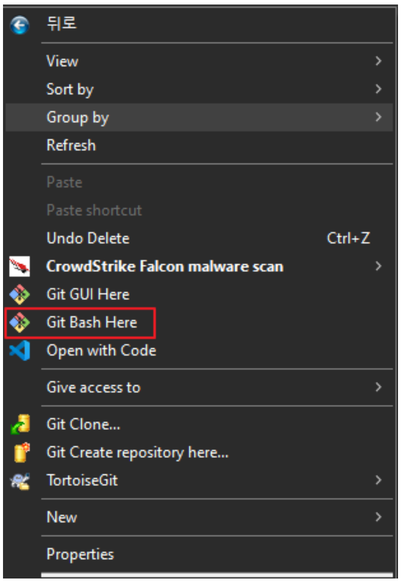
Name	Type	Value
argc	int	1
argv	char **	0xec2a60
osal_ret	e_osal_return_t	OSAL_RETURN_OK
__func__	const char [5]	0x404328 <__func...
thrd_hndl	osal_thread_handl...	0x0
thrd_return_v	int64_t	-1
thrd_cfg	st_osal_thread_co...	{...}

HELLOWORLD_sample V4H2 Simulation [Renesas IMPSim Debugging]

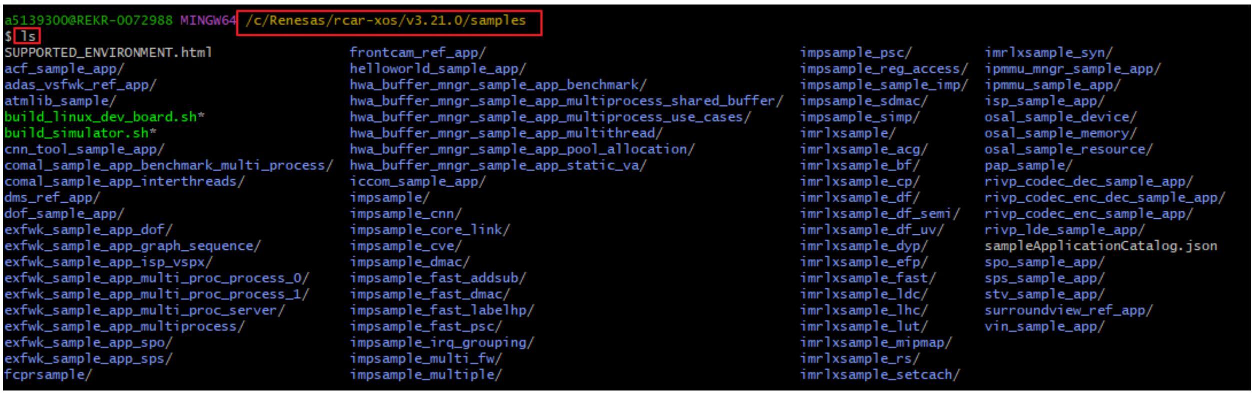
6 GIT Bash 설정

하기의 설정의 GIT 이 설치 되어 있다고 가정하고 진행한다.

.sh 파일에서 마우스 우측을 클릭했을 때 아래와 같이 Git Bash 메뉴가 있어야 한다.



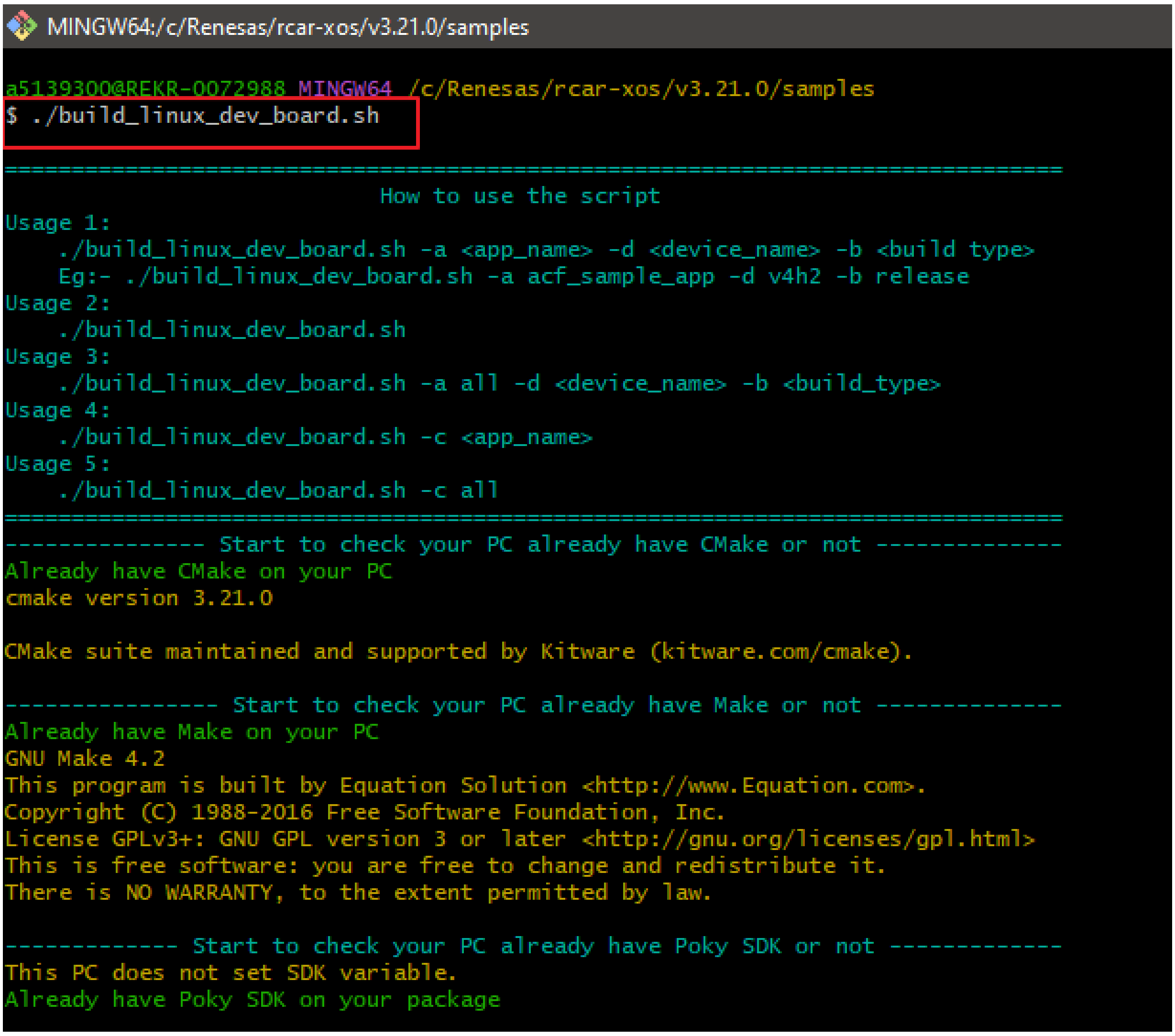
아래와 같이 ls 명령어를 넣었을 때, Samples 프로젝트 폴더에 위치하여야 한다.



```
a5139300@REKR-0072988 MINGW64 /c/Renesas/rcar-xos/v3.21.0/samples
$ ls
SUPPORTED_ENVIRONMENT.html                 frontcam_ref_app/                                      impsample_psc/        imrlxsample_syn/
acf_sample_app/                            helloworld_sample_app/                                 impsample_reg_access/ ipmmu_mngr_sample_app/
adas_vsfwk_ref_app/                        hwa_buffer_mngr_sample_app_benchmark/                  impsample_sample_imp/ ipmmu_sample_app/
atmlib_sample/                             hwa_buffer_mngr_sample_app_multiprocess_shared_buffer/ impsample_sdmac/      isp_sample_app/
build_linux_dev_board.sh*                  hwa_buffer_mngr_sample_app_multiprocess_use_cases/     impsample_simp/       osal_sample_device/
build_simulator.sh*                        hwa_buffer_mngr_sample_app_multithread/                imrlxsample/          osal_sample_memory/
cnn_tool_sample_app/                       hwa_buffer_mngr_sample_app_pool_allocation/            imrlxsample_acg/      osal_sample_resource/
comal_sample_app_benchmark_multi_process/  hwa_buffer_mngr_sample_app_static_va/                  imrlxsample_bf/       pap_sample/
comal_sample_app_interthreads/             iccom_sample_app/                                      imrlxsample_cp/       rivp_codec_dec_sample_app/
dms_ref_app/                               impsample/                                             imrlxsample_df/       rivp_codec_enc_dec_sample_app/
dof_sample_app/                            impsample_cnn/                                         imrlxsample_df_semi/  rivp_codec_enc_sample_app/
exfwk_sample_app_dof/                      impsample_core_link/                                   imrlxsample_df_uv/    rivp_lde_sample_app/
exfwk_sample_app_graph_sequence/           impsample_cve/                                         imrlxsample_dyp/      sampleApplicationCatalog.json
exfwk_sample_app_isp_vspx/                 impsample_dmac/                                        imrlxsample_efp/      spo_sample_app/
exfwk_sample_app_multi_proc_process_0/     impsample_fast_addsub/                                 imrlxsample_fast/     sps_sample_app/
exfwk_sample_app_multi_proc_process_1/     impsample_fast_dmac/                                   imrlxsample_ldc/      stv_sample_app/
exfwk_sample_app_multi_proc_server/        impsample_fast_labelhp/                                imrlxsample_lhc/      surroundview_ref_app/
exfwk_sample_app_multiprocess/             impsample_fast_psc/                                    imrlxsample_lut/      vin_sample_app/
exfwk_sample_app_spo/                      impsample_irq_grouping/                                imrlxsample_mipmap/
exfwk_sample_app_sps/                      impsample_multi_fw/                                    imrlxsample_rs/
fcprsample/                                impsample_multiple/                                    imrlxsample_setcach/
```

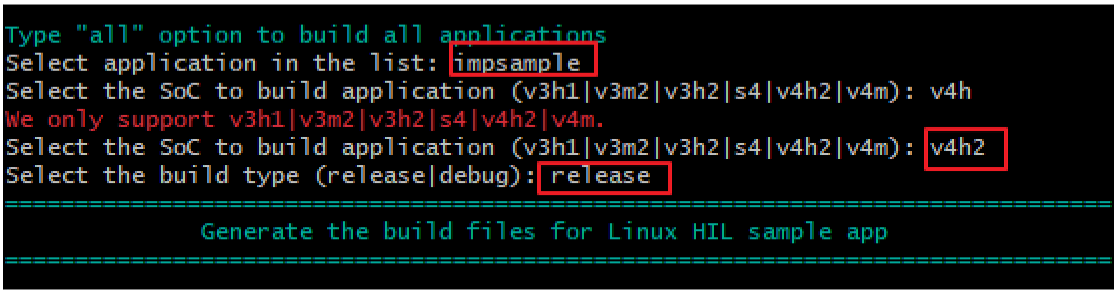
6.1 Build

아래 그림과 같이 “./build_linux_dev_board.sh” 명령어를 입력하면 Build 를 진행할 수 있다.



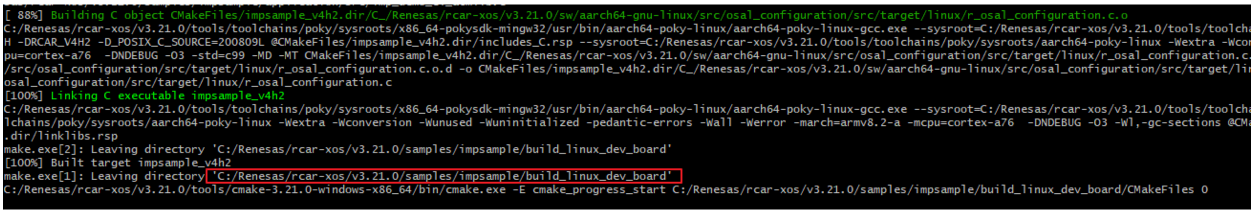
```
MINGW64:/c/Renesas/rcar-xos/v3.21.0/samples

a5139300@REKR-0072988 MINGW64 /c/Renesas/rcar-xos/v3.21.0/samples
$ ./build_linux_dev_board.sh

================================================================================
                            How to use the script
Usage 1:
    ./build_linux_dev_board.sh -a <app_name> -d <device_name> -b <build type>
    Eg:- ./build_linux_dev_board.sh -a acf_sample_app -d v4h2 -b release
Usage 2:
    ./build_linux_dev_board.sh
Usage 3:
    ./build_linux_dev_board.sh -a all -d <device_name> -b <build_type>
Usage 4:
    ./build_linux_dev_board.sh -c <app_name>
Usage 5:
    ./build_linux_dev_board.sh -c all
================================================================================
--------------- Start to check your PC already have CMake or not --------------
Already have CMake on your PC
cmake version 3.21.0

CMake suite maintained and supported by Kitware (kitware.com/cmake).

---------------- Start to check your PC already have Make or not --------------
Already have Make on your PC
GNU Make 4.2
This program is built by Equation Solution <http://www.Equation.com>.
Copyright (C) 1988-2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

------------- Start to check your PC already have Poky SDK or not -------------
This PC does not set SDK variable.
Already have Poky SDK on your package
```

아래와 같이 원하는 샘플 프로젝트 Device 그리고 Build type 을 입력하면 Build 가 진행된다.



```
Type "all" option to build all applications
Select application in the list: impsample
Select the SoC to build application (v3h1|v3m2|v3h2|s4|v4h2|v4m): v4h
We only support v3h1|v3m2|v3h2|s4|v4h2|v4m.
Select the SoC to build application (v3h1|v3m2|v3h2|s4|v4h2|v4m): v4h2
Select the build type (release|debug): release
================================================================================
              Generate the build files for Linux HIL sample app
================================================================================
```

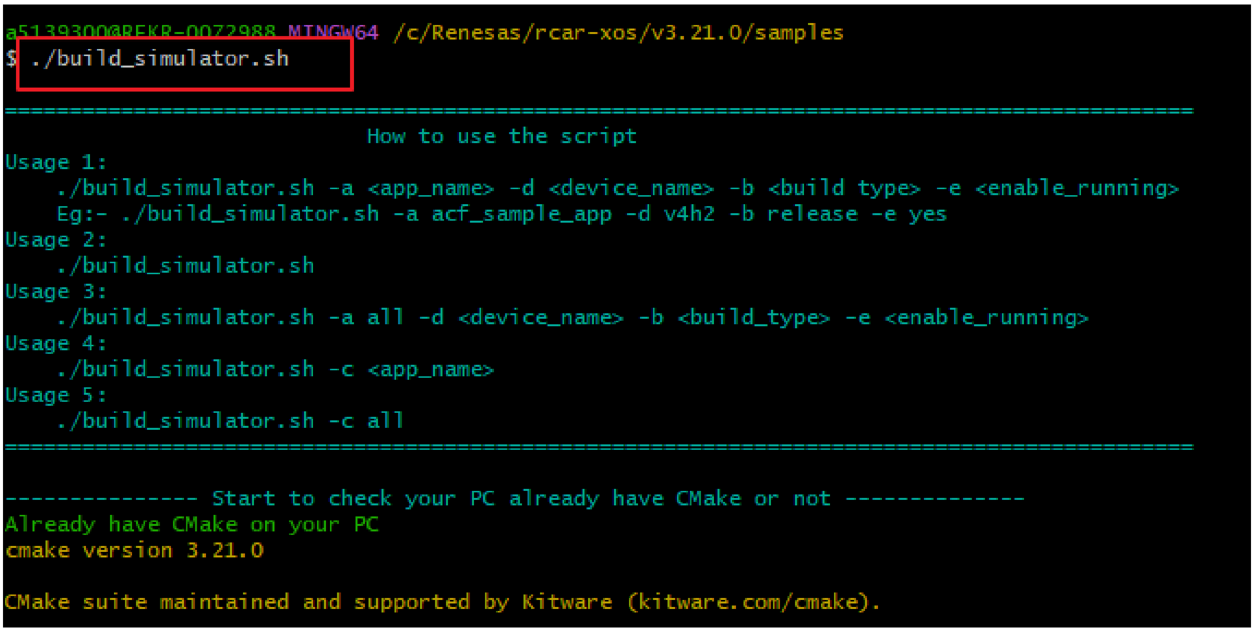
빌드가 완료되면 Sample Project 폴더에 Build 관련 폴더가 생성됨을 확인 할 수 있다.



```
[ 88%] Building C object CMakeFiles/impsample_v4h2.dir/C_/Renesas/rcar-xos/v3.21.0/sw/aarch64-gnu-linux/src/osal_configuration/src/target/linux/r_osal_configuration.c.o
C:/Renesas/rcar-xos/v3.21.0/tools/toolchains/poky/sysroots/x86_64-pokysdk-mingw32/usr/bin/aarch64-poky-linux/aarch64-poky-linux-gcc.exe --sysroot=C:/Renesas/rcar-xos/v3.21.0/tools/toolcha
H -DRCAR_V4H2 -D_POSIX_C_SOURCE=200809L @CMakeFiles/impsample_v4h2.dir/includes_C.rsp --sysroot=C:/Renesas/rcar-xos/v3.21.0/tools/toolchains/poky/sysroots/aarch64-poky-linux -Wextra -Wcon
pu=cortex-a76  -DNDEBUG -O3 -std=c99 -MD -MT CMakeFiles/impsample_v4h2.dir/C_/Renesas/rcar-xos/v3.21.0/sw/aarch64-gnu-linux/src/osal_configuration/src/target/linux/r_osal_configuration.c.
/src/osal_configuration/src/target/linux/r_osal_configuration.c.o.d -o CMakeFiles/impsample_v4h2.dir/C_/Renesas/rcar-xos/v3.21.0/sw/aarch64-gnu-linux/src/osal_configuration/src/target/lin
osal_configuration/src/target/linux/r_osal_configuration.c
[100%] Linking C executable impsample_v4h2
C:/Renesas/rcar-xos/v3.21.0/tools/toolchains/poky/sysroots/x86_64-pokysdk-mingw32/usr/bin/aarch64-poky-linux/aarch64-poky-linux-gcc.exe --sysroot=C:/Renesas/rcar-xos/v3.21.0/tools/toolcha
lchains/poky/sysroots/aarch64-poky-linux -Wextra -Wconversion -Wunused -Wuninitialized -pedantic-errors -Wall -Werror -march=armv8.2-a -mcpu=cortex-a76  -DNDEBUG -O3 -Wl,-gc-sections @CMa
.dir/linklibs.rsp
make.exe[2]: Leaving directory 'C:/Renesas/rcar-xos/v3.21.0/samples/impsample/build_linux_dev_board'
[100%] Built target impsample_v4h2
make.exe[1]: Leaving directory 'C:/Renesas/rcar-xos/v3.21.0/samples/impsample/build_linux_dev_board'
C:/Renesas/rcar-xos/v3.21.0/tools/cmake-3.21.0-windows-x86_64/bin/cmake.exe -E cmake_progress_start C:/Renesas/rcar-xos/v3.21.0/samples/impsample/build_linux_dev_board/CMakeFiles 0
```

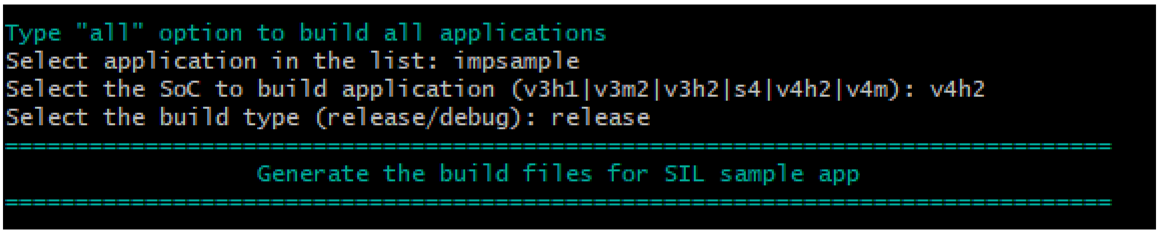
6.2 Simulator (SIL)

아래와 같이 “./build_simulator.sh”를 입력하면 Host PC 에서 시뮬레이션이 가능하다.



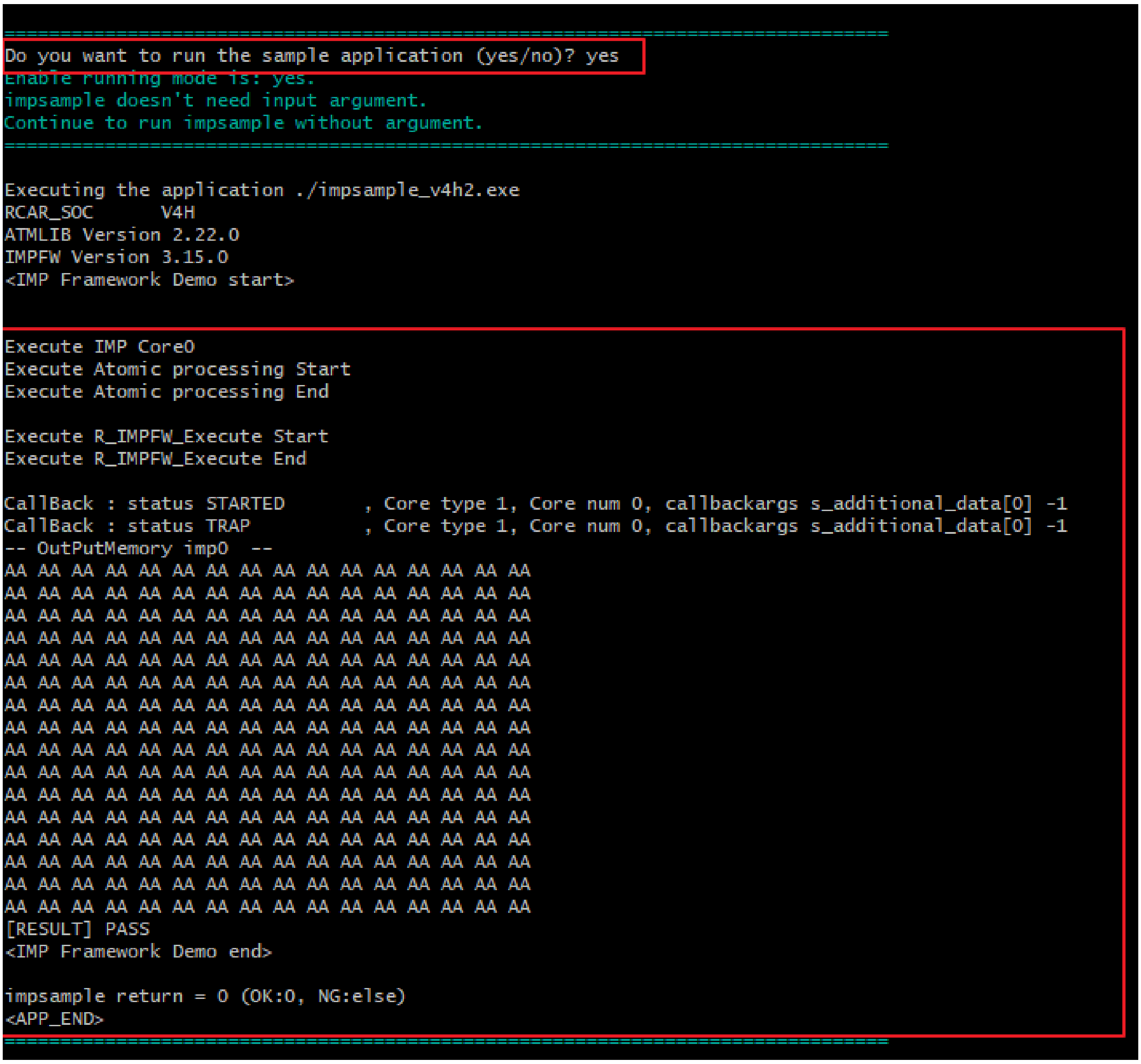
```
a5139300@REKR-0072988 MINGW64 /c/Renesas/rcar-xos/v3.21.0/samples
$ ./build_simulator.sh

============================================================================================
                            How to use the script
Usage 1:
    ./build_simulator.sh -a <app_name> -d <device_name> -b <build type> -e <enable_running>
    Eg:- ./build_simulator.sh -a acf_sample_app -d v4h2 -b release -e yes
Usage 2:
    ./build_simulator.sh
Usage 3:
    ./build_simulator.sh -a all -d <device_name> -b <build_type> -e <enable_running>
Usage 4:
    ./build_simulator.sh -c <app_name>
Usage 5:
    ./build_simulator.sh -c all
============================================================================================

--------------- Start to check your PC already have CMake or not --------------
Already have CMake on your PC
cmake version 3.21.0

CMake suite maintained and supported by Kitware (kitware.com/cmake).
```

Build 와 동일하게 샘플 프로젝트, Device, Build Type 을 정의한다.



```
Type "all" option to build all applications
Select application in the list: impsample
Select the SoC to build application (v3h1|v3m2|v3h2|s4|v4h2|v4m): v4h2
Select the build type (release/debug): release
=====================================================================================
                 Generate the build files for SIL sample app
=====================================================================================
```

Build 가 완료되면 시뮬레이션 여부를 묻는다. Yes 로 대답할 경우 아래와 같이 프로그램이 실행되는 것을 확인할 수 있다.

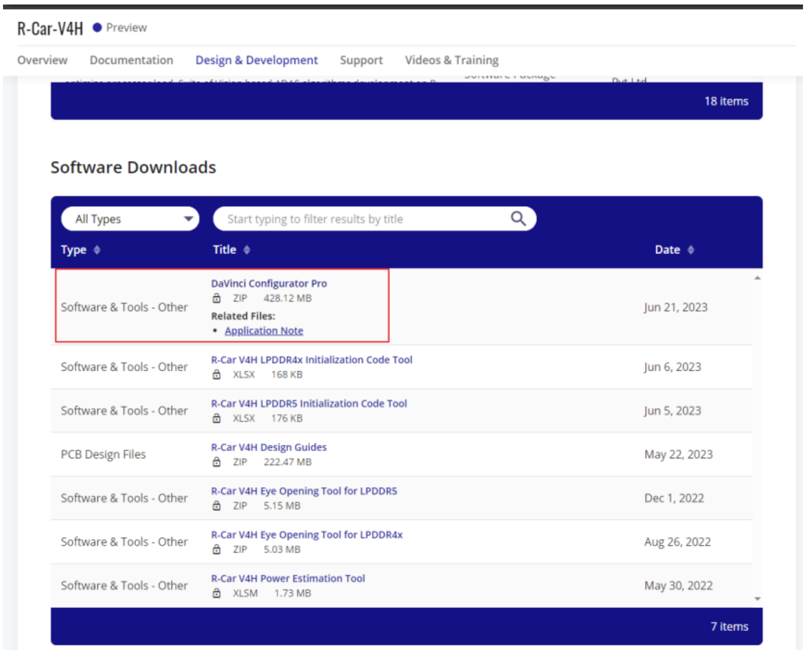


```
===============================================================================
Do you want to run the sample application (yes/no)? yes
Enable running mode is: yes.
impsample doesn't need input argument.
Continue to run impsample without argument.
===============================================================================

Executing the application ./impsample_v4h2.exe
RCAR_SOC      V4H
ATMLIB Version 2.22.0
IMPFW Version 3.15.0
<IMP Framework Demo start>


Execute IMP Core0
Execute Atomic processing Start
Execute Atomic processing End

Execute R_IMPFW_Execute Start
Execute R_IMPFW_Execute End

CallBack : status STARTED       , Core type 1, Core num 0, callbackargs s_additional_data[0] -1
CallBack : status TRAP          , Core type 1, Core num 0, callbackargs s_additional_data[0] -1
-- OutPutMemory imp0  --
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA AA
[RESULT] PASS
<IMP Framework Demo end>

impsample return = 0 (OK:0, NG:else)
<APP_END>
===============================================================================
```

7 MCAL

하기 사이트를 통해 Davinci 를 다운로드 받는다.

R-Car-V4H - Best-in-Class Deep Learning at Very Low Power, System-on-Chip for Automated Driving Level 2+/Level 3 | Renesas



R-Car-V4H ● Preview

Overview Documentation Design & Development Support Videos & Training

18 items

Software Downloads

All Types ▾ Start typing to filter results by title

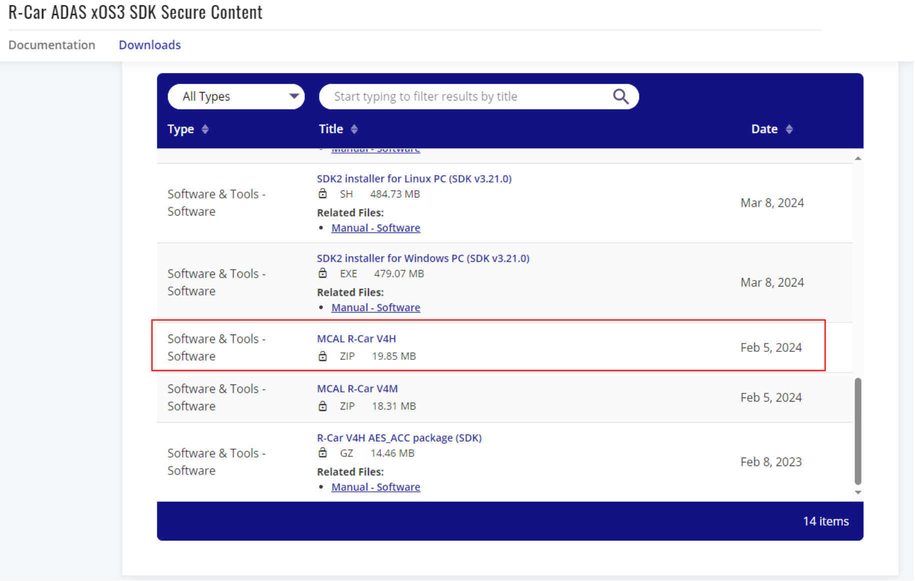
Type	Title	Date
Software & Tools - Other	DaVinci Configurator Pro ZIP 428.12 MB Related Files: • Application Note	Jun 21, 2023
Software & Tools - Other	R-Car V4H LPDDR4x Initialization Code Tool XLSX 168 KB	Jun 6, 2023
Software & Tools - Other	R-Car V4H LPDDR5 Initialization Code Tool XLSX 176 KB	Jun 5, 2023
PCB Design Files	R-Car V4H Design Guides ZIP 222.47 MB	May 22, 2023
Software & Tools - Other	R-Car V4H Eye Opening Tool for LPDDR5 ZIP 5.15 MB	Dec 1, 2022
Software & Tools - Other	R-Car V4H Eye Opening Tool for LPDDR4x ZIP 5.03 MB	Aug 26, 2022
Software & Tools - Other	R-Car V4H Power Estimation Tool XLSM 1.73 MB	May 30, 2022

7 items

하기 사이트를 통해 V4H MCAL 를 다운로드 받는다.

MCAL 은 MyRenesas 를 통해서만 다운로드가 가능하다. (QM Version)

R-Car ADAS xOS3 SDK Secure Content | Renesas



R-Car ADAS xOS3 SDK Secure Content

Documentation Downloads

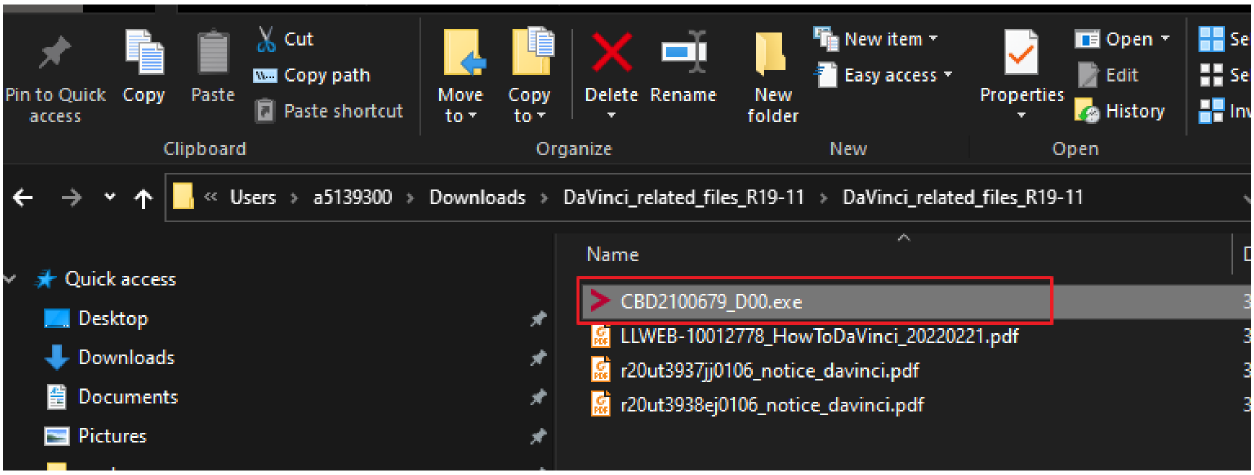
All Types ▾ Start typing to filter results by title

Type	Title	Date
Software & Tools - Software	SDK2 installer for Linux PC (SDK v3.21.0) SH 484.73 MB Related Files: • Manual - Software	Mar 8, 2024
Software & Tools - Software	SDK2 installer for Windows PC (SDK v3.21.0) EXE 479.07 MB Related Files: • Manual - Software	Mar 8, 2024
Software & Tools - Software	MCAL R-Car V4H ZIP 19.85 MB	Feb 5, 2024
Software & Tools - Software	MCAL R-Car V4M ZIP 18.31 MB	Feb 5, 2024
Software & Tools - Software	R-Car V4H AES_ACC package (SDK) GZ 14.46 MB Related Files: • Manual - Software	Feb 8, 2023

14 items

7.1 Installation

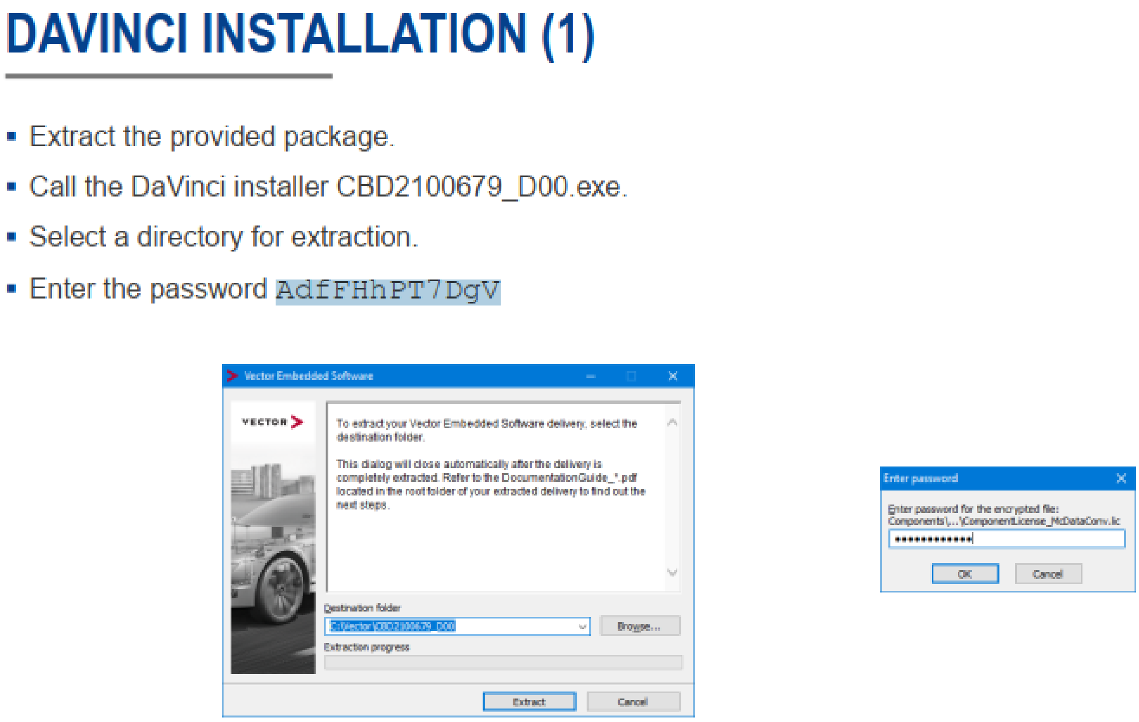
아래 CBD2100679_D00.exe 파일을 실행하여 원하는 폴더에 설치를 진행한다.



« Users › a5139300 › Downloads › DaVinci_related_files_R19-11 › DaVinci_related_files_R19-11

- CBD2100679_D00.exe
- LLWEB-10012778_HowToDaVinci_20220221.pdf
- r20ut3937jj0106_notice_davinci.pdf
- r20ut3938ej0106_notice_davinci.pdf

아래 그림과 같이 암호를 입력하는 순서가 있는데, 이는 "LLWEB-10012778_HowToDaVinci_20220221.pdf" 파일을 참고하면 된다.



DAVINCI INSTALLATION (1)

- Extract the provided package.
- Call the DaVinci installer CBD2100679_D00.exe.
- Select a directory for extraction.
- Enter the password AdfFHhPT7DgV

Vector Embedded Software

To extract your Vector Embedded Software delivery, select the destination folder.

This dialog will close automatically after the delivery is completely extracted. Refer to the DocumentationGuide_*.pdf located in the root folder of your extracted delivery to find out the next steps.

Destination folder: C:\Vector\CBD2100679_D00

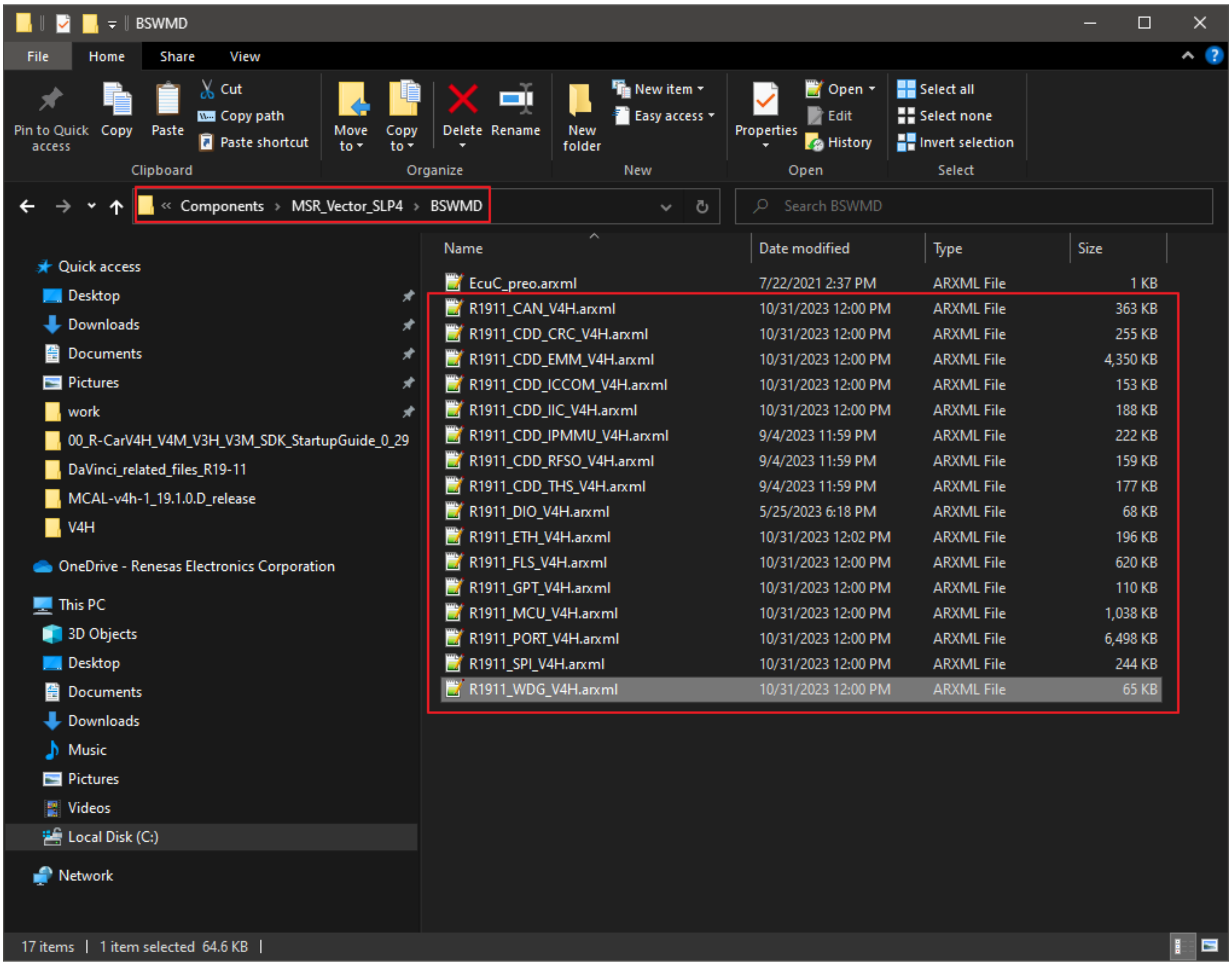
Extraction progress

Enter password

Enter password for the encrypted file:
Components\...\ComponentLicense_McDataConv.lic

•••••••••••••

설치가 완료되면 매뉴얼에 따라 MCAL 에 있는 BSWMDT 파일을 Davinci 설치 폴더로 복사한다.

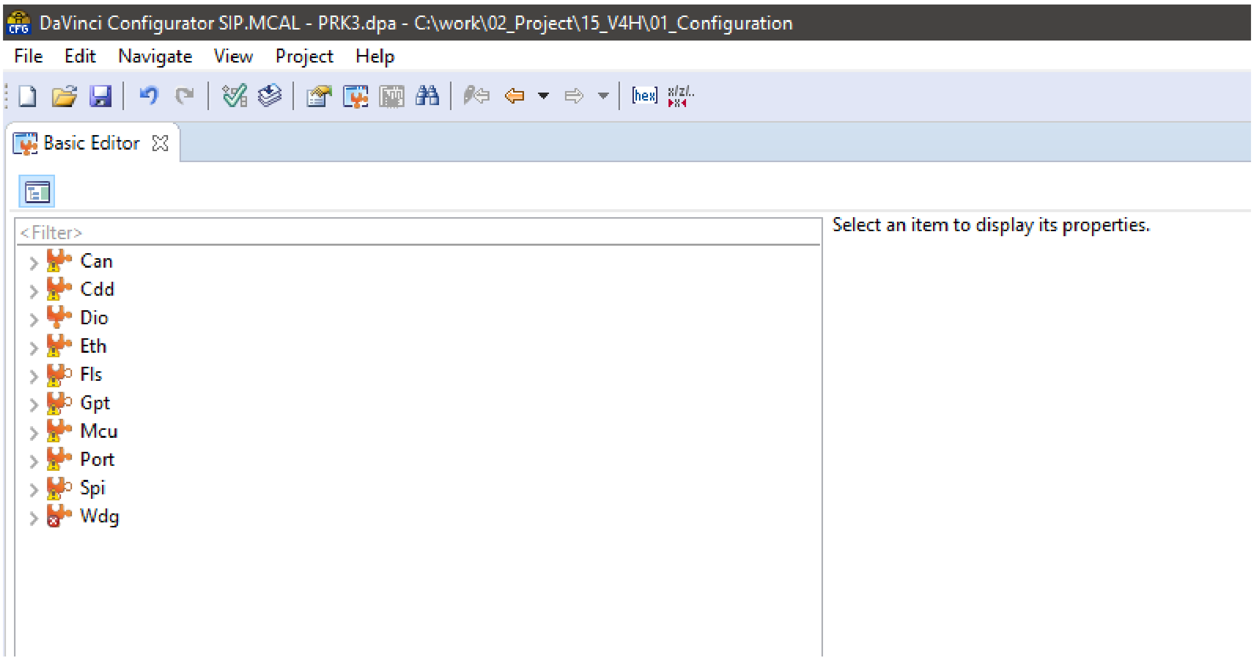


« Components › MSR_Vector_SLP4 › BSWMD

Name	Date modified	Type	Size
EcuC_preo.arxml	7/22/2021 2:37 PM	ARXML File	1 KB
R1911_CAN_V4H.arxml	10/31/2023 12:00 PM	ARXML File	363 KB
R1911_CDD_CRC_V4H.arxml	10/31/2023 12:00 PM	ARXML File	255 KB
R1911_CDD_EMM_V4H.arxml	10/31/2023 12:00 PM	ARXML File	4,350 KB
R1911_CDD_ICCOM_V4H.arxml	10/31/2023 12:00 PM	ARXML File	153 KB
R1911_CDD_IIC_V4H.arxml	10/31/2023 12:00 PM	ARXML File	188 KB
R1911_CDD_IPMMU_V4H.arxml	9/4/2023 11:59 PM	ARXML File	222 KB
R1911_CDD_RFSO_V4H.arxml	9/4/2023 11:59 PM	ARXML File	159 KB
R1911_CDD_THS_V4H.arxml	9/4/2023 11:59 PM	ARXML File	177 KB
R1911_DIO_V4H.arxml	5/25/2023 6:18 PM	ARXML File	68 KB
R1911_ETH_V4H.arxml	10/31/2023 12:02 PM	ARXML File	196 KB
R1911_FLS_V4H.arxml	10/31/2023 12:00 PM	ARXML File	620 KB
R1911_GPT_V4H.arxml	10/31/2023 12:00 PM	ARXML File	110 KB
R1911_MCU_V4H.arxml	10/31/2023 12:00 PM	ARXML File	1,038 KB
R1911_PORT_V4H.arxml	10/31/2023 12:00 PM	ARXML File	6,498 KB
R1911_SPI_V4H.arxml	10/31/2023 12:00 PM	ARXML File	244 KB
R1911_WDG_V4H.arxml	10/31/2023 12:00 PM	ARXML File	65 KB

17 items | 1 item selected 64.6 KB |

정상적으로 BSWMD 파일이 복사되면 아래와 같이 각 모듈을 Import 가능하다.


DaVinci Configurator SIP.MCAL - PRK3.dpa - C:\work\02_Project\15_V4H\01_Configuration
File
Edit
Navigate
View
Project
Help
Basic Editor
<Filter>

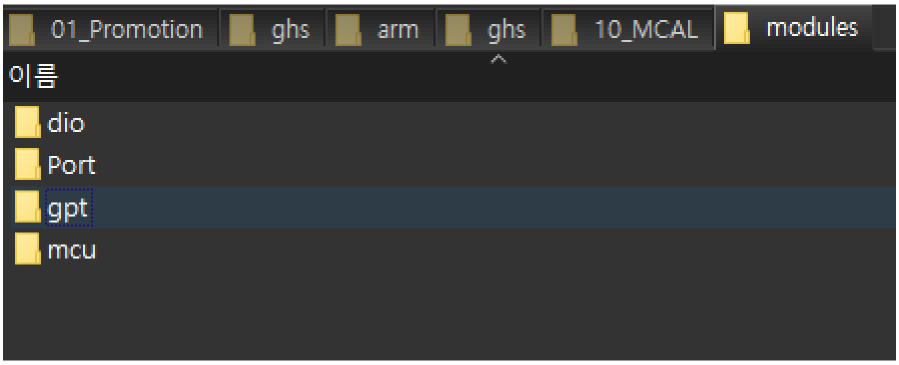
- Can
- Cdd
- Dio
- Eth
- Fls
- Gpt
- Mcu
- Port
- Spi
- Wdg

Select an item to display its properties.

7.2 Building

Please refer to Mcal Module overview

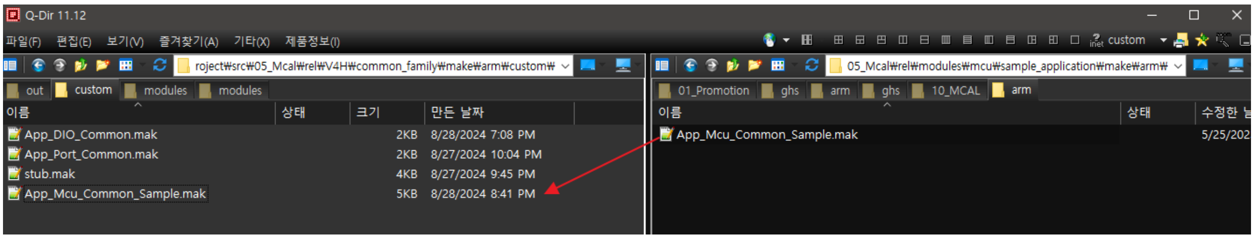
7.3 Make File Integration



이름
dio
Port
gpt
mcu

모듈 폴더에 원하는 모듈을 Copy 하여 붙여넣기 한다.

\\Module\sample_application\make\arm 폴더로 이동하여 make 파일을 rel\V4H\common_family\make\arm\ 위치로 복사 붙여 넣기 한다.



roject₩src₩05_Mcal₩rel₩V4H₩common_family₩make₩arm₩custom₩

이름	상태	크기	만든 날짜
App_DIO_Common.mak		2KB	8/28/2024 7:08 PM
App_Port_Common.mak		2KB	8/27/2024 10:04 PM
stub.mak		4KB	8/27/2024 9:45 PM
App_Mcu_Common_Sample.mak		5KB	8/28/2024 8:41 PM

05_Mcal₩rel₩modules₩mcu₩sample_application₩make₩arm₩

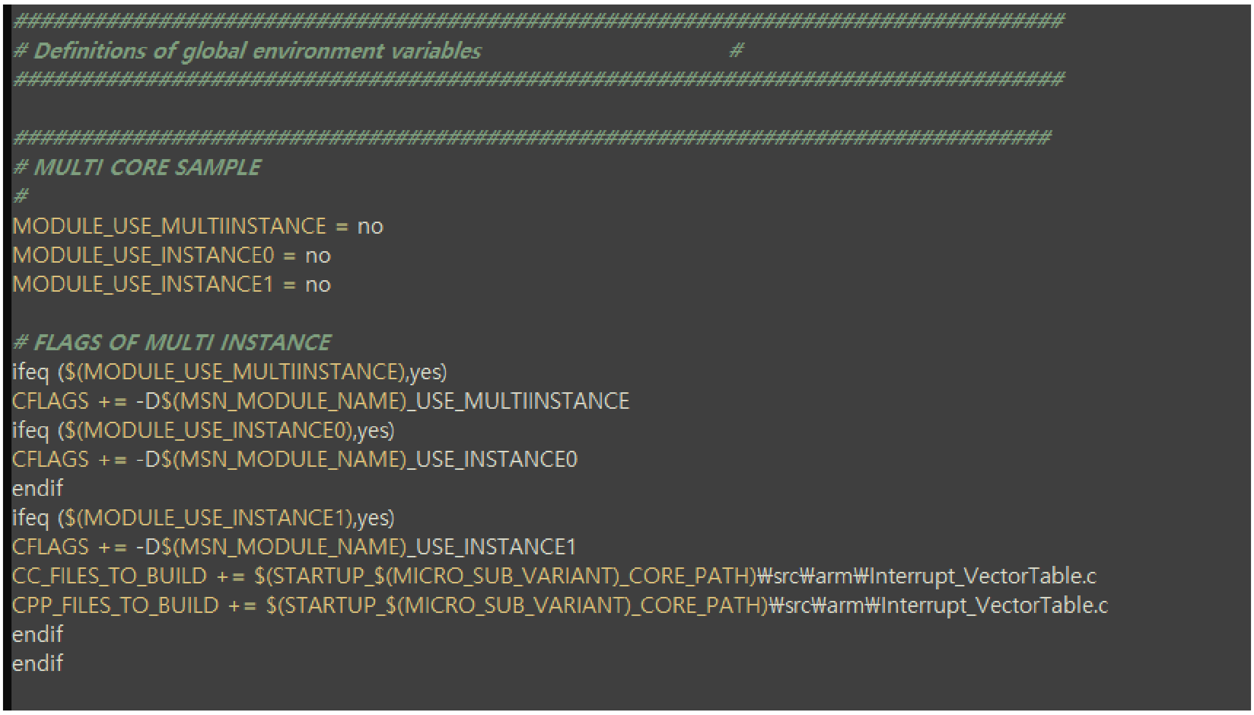
이름	상태	수정한 날
App_Mcu_Common_Sample.mak		5/25/202

이때 파일명이 헷갈리지 않도록 일부 변경하여 준다. (ex: App_Mcu_Common.mak)

7.3.1 Module Make file 수정

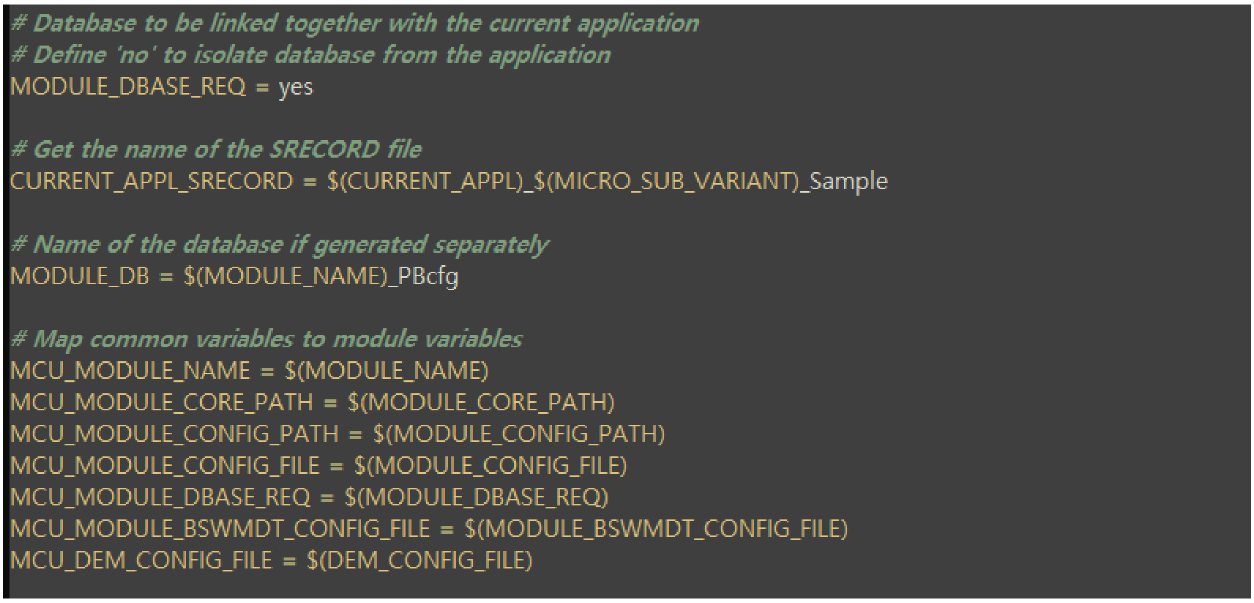
지금부터 Module 별 Make File 을 수정해야 한다.

공통 적인 부분은 삭제하고, 필요한 부분만 남겨두어 Compilation 과정에서 충돌이 없도록 설정해야 한다.



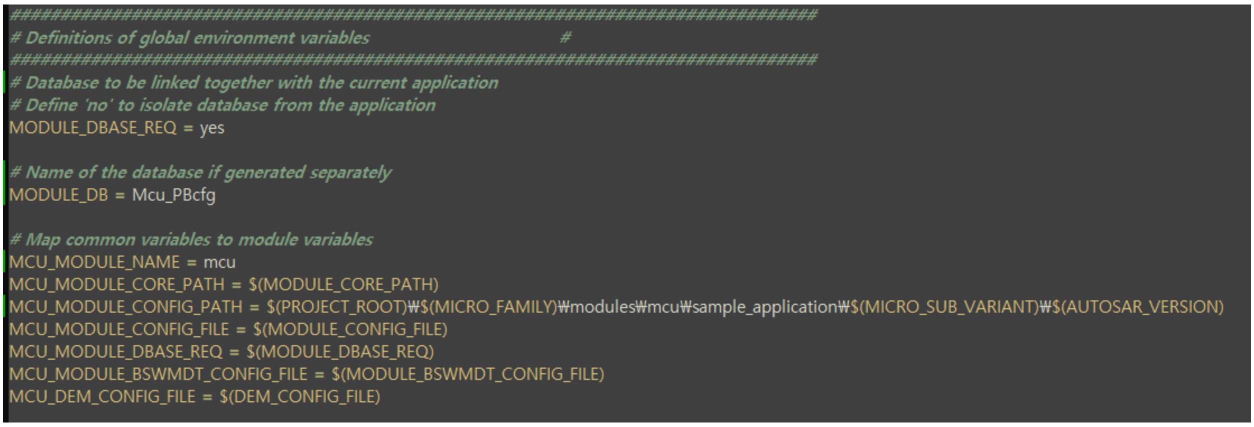
```
##############################################################################
# Definitions of global environment variables                               #
##############################################################################

#############################################################################
# MULTI CORE SAMPLE
#
MODULE_USE_MULTIINSTANCE = no
MODULE_USE_INSTANCE0 = no
MODULE_USE_INSTANCE1 = no

# FLAGS OF MULTI INSTANCE
ifeq ($(MODULE_USE_MULTIINSTANCE),yes)
CFLAGS += -D$(MSN_MODULE_NAME)_USE_MULTIINSTANCE
ifeq ($(MODULE_USE_INSTANCE0),yes)
CFLAGS += -D$(MSN_MODULE_NAME)_USE_INSTANCE0
endif
ifeq ($(MODULE_USE_INSTANCE1),yes)
CFLAGS += -D$(MSN_MODULE_NAME)_USE_INSTANCE1
CC_FILES_TO_BUILD += $(STARTUP_$(MICRO_SUB_VARIANT)_CORE_PATH)₩src₩arm₩Interrupt_VectorTable.c
CPP_FILES_TO_BUILD += $(STARTUP_$(MICRO_SUB_VARIANT)_CORE_PATH)₩src₩arm₩Interrupt_VectorTable.c
endif
endif
```

위 부분은 전부 삭제해도 된다.



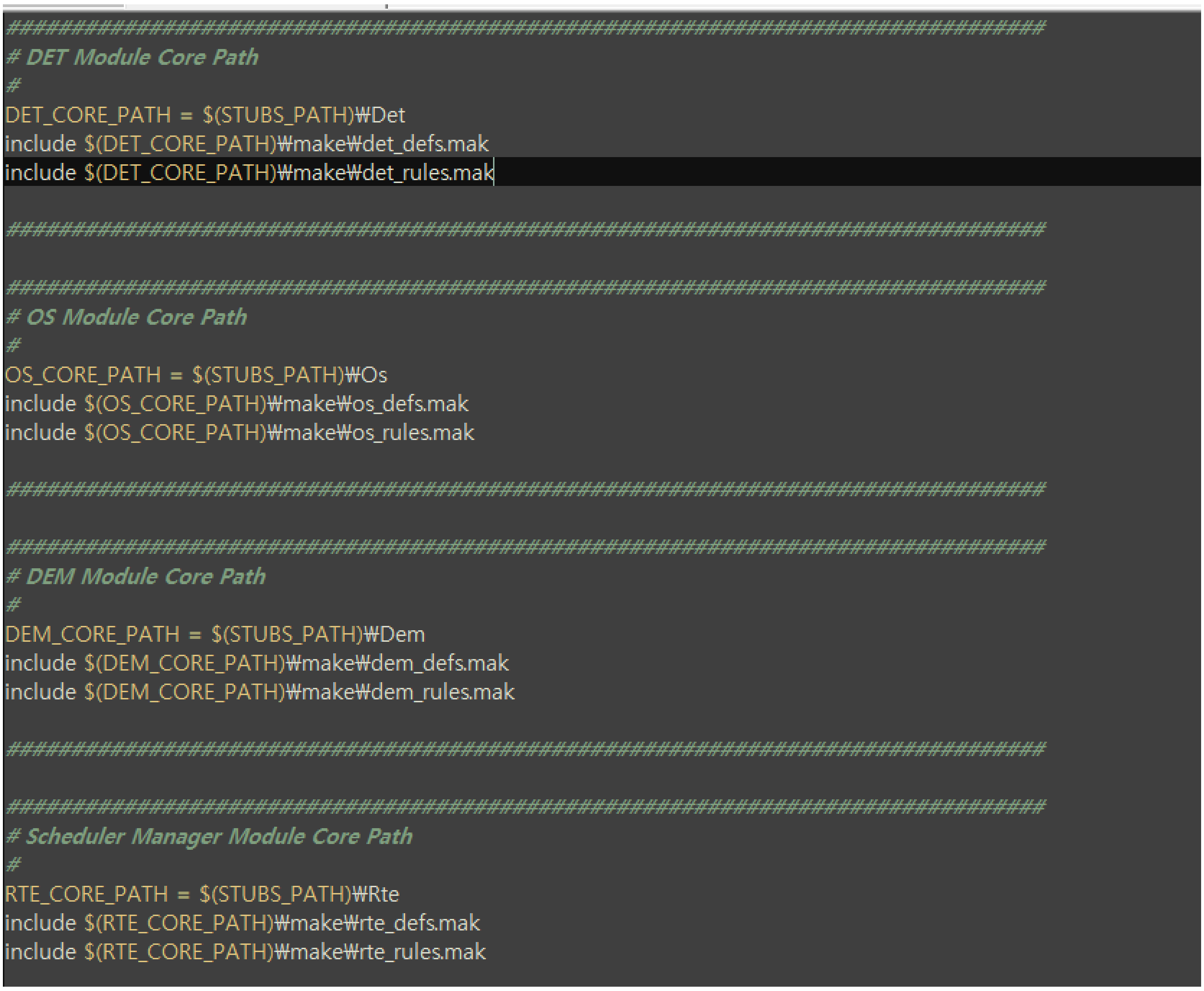
```
# Database to be linked together with the current application
# Define 'no' to isolate database from the application
MODULE_DBASE_REQ = yes

# Get the name of the SRECORD file
CURRENT_APPL_SRECORD = $(CURRENT_APPL)_$(MICRO_SUB_VARIANT)_Sample

# Name of the database if generated separately
MODULE_DB = $(MODULE_NAME)_PBcfg

# Map common variables to module variables
MCU_MODULE_NAME = $(MODULE_NAME)
MCU_MODULE_CORE_PATH = $(MODULE_CORE_PATH)
MCU_MODULE_CONFIG_PATH = $(MODULE_CONFIG_PATH)
MCU_MODULE_CONFIG_FILE = $(MODULE_CONFIG_FILE)
MCU_MODULE_DBASE_REQ = $(MODULE_DBASE_REQ)
MCU_MODULE_BSWMDT_CONFIG_FILE = $(MODULE_BSWMDT_CONFIG_FILE)
MCU_DEM_CONFIG_FILE = $(DEM_CONFIG_FILE)
```

위 부분은 일부 수정을 해야 한다. 충돌을 방지하기 위해서는 Path 설정을 독립적으로 진행해야 한다.



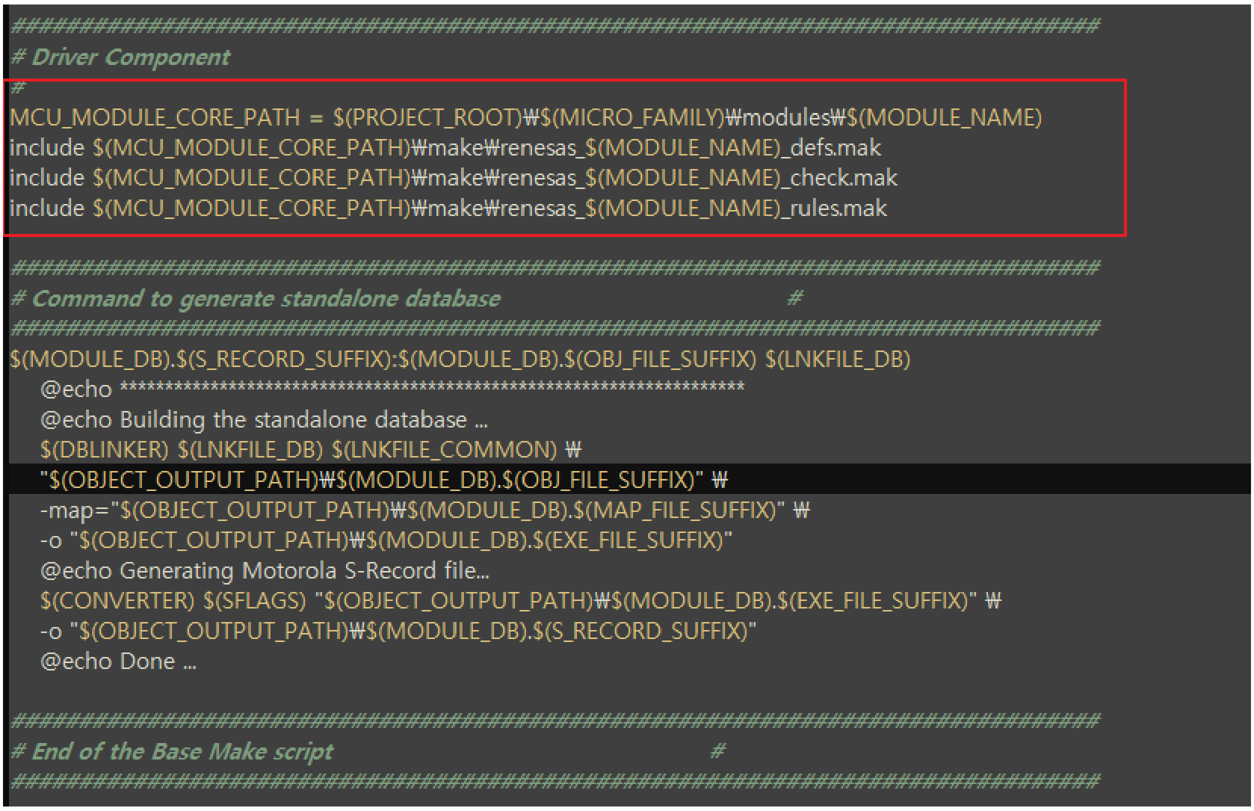
```
###############################################################################
# Definitions of global environment variables                                  #
###############################################################################
# Database to be linked together with the current application
# Define 'no' to isolate database from the application
MODULE_DBASE_REQ = yes

# Name of the database if generated separately
MODULE_DB = Mcu_PBcfg

# Map common variables to module variables
MCU_MODULE_NAME = mcu
MCU_MODULE_CORE_PATH = $(MODULE_CORE_PATH)
MCU_MODULE_CONFIG_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩mcu₩sample_application₩$(MICRO_SUB_VARIANT)₩$(AUTOSAR_VERSION)
MCU_MODULE_CONFIG_FILE = $(MODULE_CONFIG_FILE)
MCU_MODULE_DBASE_REQ = $(MODULE_DBASE_REQ)
MCU_MODULE_BSWMDT_CONFIG_FILE = $(MODULE_BSWMDT_CONFIG_FILE)
MCU_DEM_CONFIG_FILE = $(DEM_CONFIG_FILE)
```

위와 같이 MODULE_DB, MCU_MODULE_NAME, MCU_MODULE_CONFIG_PATH 의 순서로 수정한다.



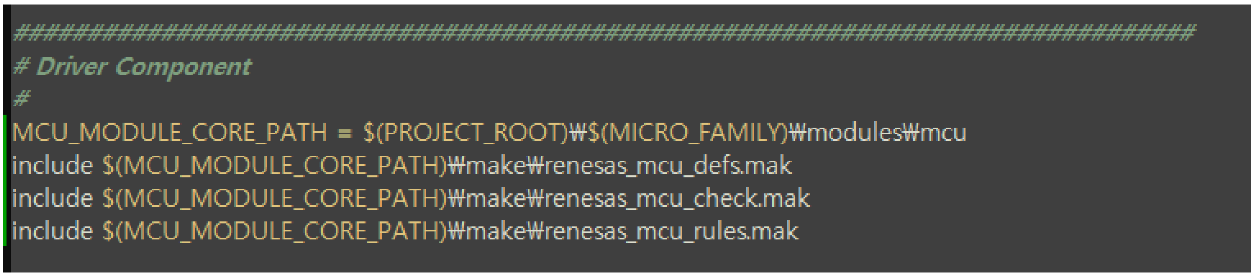
```
###############################################################################
# DET Module Core Path
#
DET_CORE_PATH = $(STUBS_PATH)₩Det
include $(DET_CORE_PATH)₩make₩det_defs.mak
include $(DET_CORE_PATH)₩make₩det_rules.mak

###############################################################################

###############################################################################
# OS Module Core Path
#
OS_CORE_PATH = $(STUBS_PATH)₩Os
include $(OS_CORE_PATH)₩make₩os_defs.mak
include $(OS_CORE_PATH)₩make₩os_rules.mak

###############################################################################

###############################################################################
# DEM Module Core Path
#
DEM_CORE_PATH = $(STUBS_PATH)₩Dem
include $(DEM_CORE_PATH)₩make₩dem_defs.mak
include $(DEM_CORE_PATH)₩make₩dem_rules.mak

###############################################################################

###############################################################################
# Scheduler Manager Module Core Path
#
RTE_CORE_PATH = $(STUBS_PATH)₩Rte
include $(RTE_CORE_PATH)₩make₩rte_defs.mak
include $(RTE_CORE_PATH)₩make₩rte_rules.mak
```

위 부분은 공통 부분에 해당하므로 전부 삭제해도 된다.



```
##############################################################################
# Driver Component
#
MCU_MODULE_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩$(MODULE_NAME)
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_$(MODULE_NAME)_defs.mak
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_$(MODULE_NAME)_check.mak
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_$(MODULE_NAME)_rules.mak

##############################################################################
# Command to generate standalone database                        #
##############################################################################
$(MODULE_DB).$(S_RECORD_SUFFIX):$(MODULE_DB).$(OBJ_FILE_SUFFIX) $(LNKFILE_DB)
   @echo ************************************************************************
   @echo Building the standalone database ...
   $(DBLINKER) $(LNKFILE_DB) $(LNKFILE_COMMON) ₩
   "$(OBJECT_OUTPUT_PATH)₩$(MODULE_DB).$(OBJ_FILE_SUFFIX)" ₩
   -map="$(OBJECT_OUTPUT_PATH)₩$(MODULE_DB).$(MAP_FILE_SUFFIX)" ₩
   -o "$(OBJECT_OUTPUT_PATH)₩$(MODULE_DB).$(EXE_FILE_SUFFIX)"
   @echo Generating Motorola S-Record file...
   $(CONVERTER) $(SFLAGS) "$(OBJECT_OUTPUT_PATH)₩$(MODULE_DB).$(EXE_FILE_SUFFIX)" ₩
   -o "$(OBJECT_OUTPUT_PATH)₩$(MODULE_DB).$(S_RECORD_SUFFIX)"
   @echo Done ...

##############################################################################
# End of the Base Make script                                   #
##############################################################################
```

위 빨간 부분은 일부 수정을 진행해야 한다. 나머지 부분은 삭제한다.



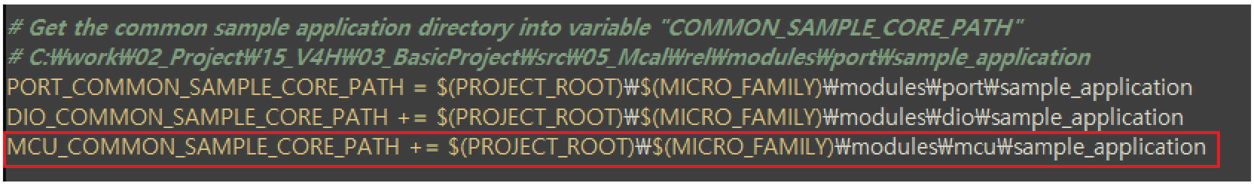
```
##############################################################################
# Driver Component
#
MCU_MODULE_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩mcu
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_mcu_defs.mak
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_mcu_check.mak
include $(MCU_MODULE_CORE_PATH)₩make₩renesas_mcu_rules.mak
```

위와 같이 일부 MODULE NAME 을 수정하였다.

7.3.2 Module Make File 적용

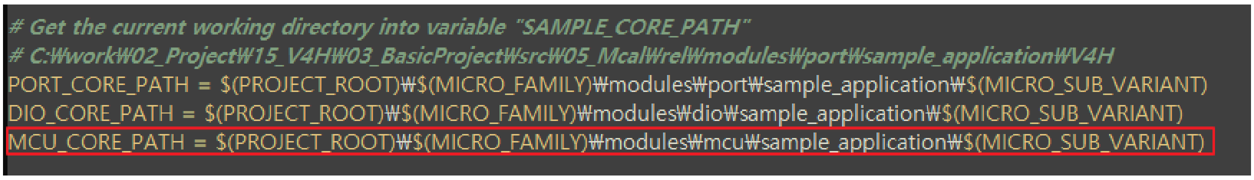
Module Make 파일 작업이 완료되면 CommonCustom.mak 파일을 Open 한다.

어플리케이션 PATH 생성 및 정의



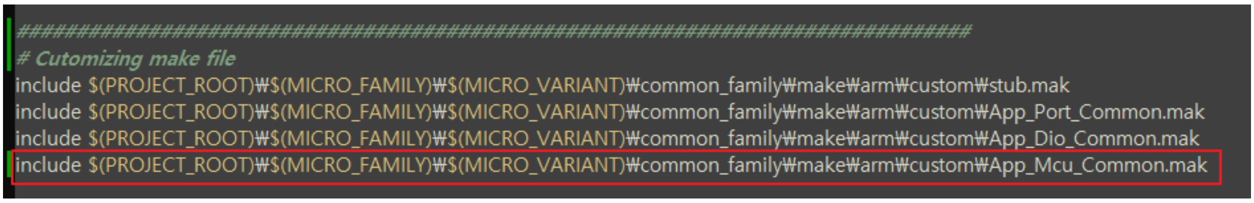
```
# Get the common sample application directory into variable "COMMON_SAMPLE_CORE_PATH"
# C:₩work₩02_Project₩15_V4H₩03_BasicProject₩src₩05_Mcal₩rel₩modules₩port₩sample_application
PORT_COMMON_SAMPLE_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩port₩sample_application
DIO_COMMON_SAMPLE_CORE_PATH += $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩dio₩sample_application
MCU_COMMON_SAMPLE_CORE_PATH += $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩mcu₩sample_application
```

Module Core Path 생성 및 정의



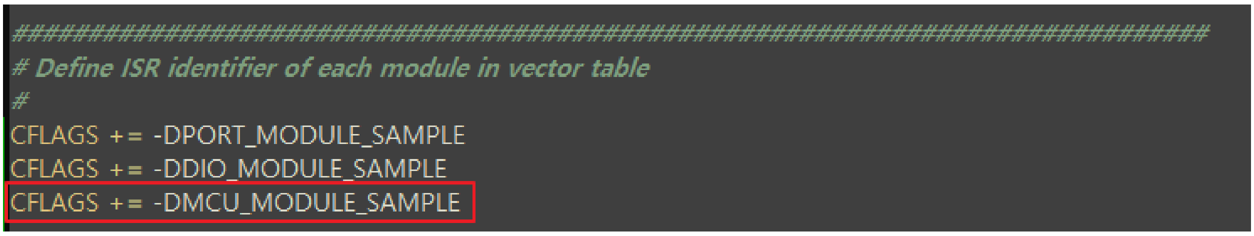
```
# Get the current working directory into variable "SAMPLE_CORE_PATH"
# C:₩work₩02_Project₩15_V4H₩03_BasicProject₩src₩05_Mcal₩rel₩modules₩port₩sample_application₩V4H
PORT_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩port₩sample_application₩$(MICRO_SUB_VARIANT)
DIO_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩dio₩sample_application₩$(MICRO_SUB_VARIANT)
MCU_CORE_PATH = $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩mcu₩sample_application₩$(MICRO_SUB_VARIANT)
```

Module Make File 호출



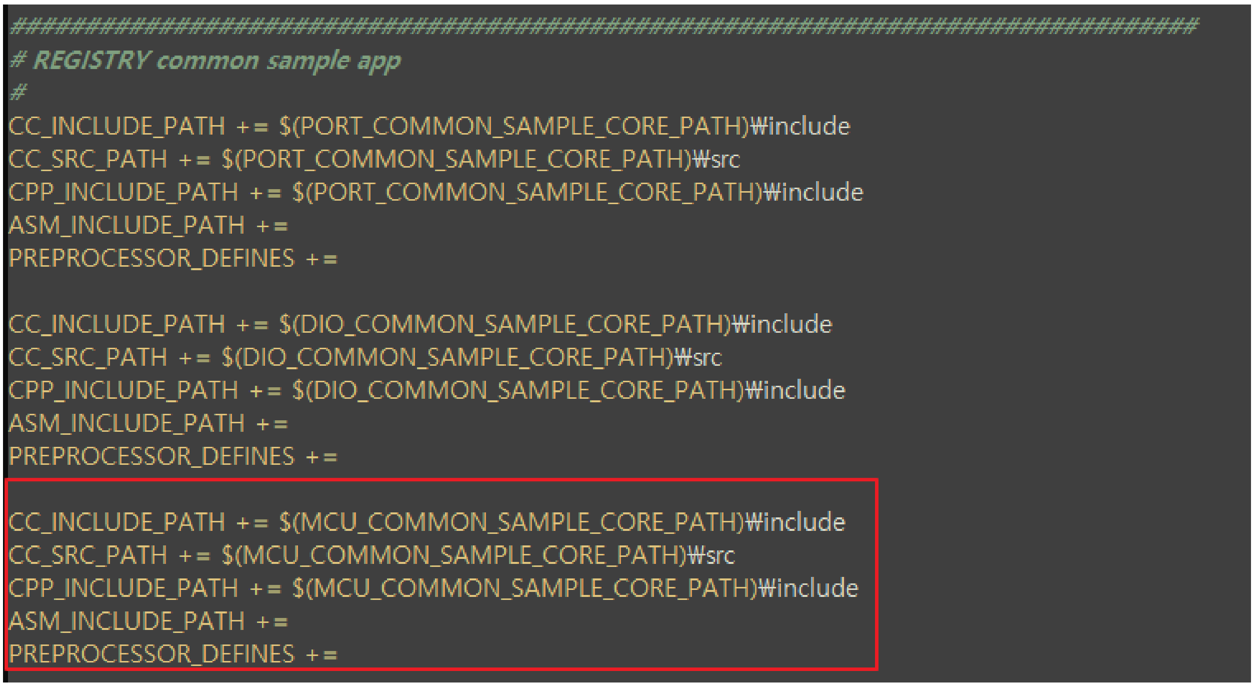
```
################################################################################
# Cutomizing make file
include $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩$(MICRO_VARIANT)₩common_family₩make₩arm₩custom₩stub.mak
include $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩$(MICRO_VARIANT)₩common_family₩make₩arm₩custom₩App_Port_Common.mak
include $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩$(MICRO_VARIANT)₩common_family₩make₩arm₩custom₩App_Dio_Common.mak
include $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩$(MICRO_VARIANT)₩common_family₩make₩arm₩custom₩App_Mcu_Common.mak
```

PreDefine 추가

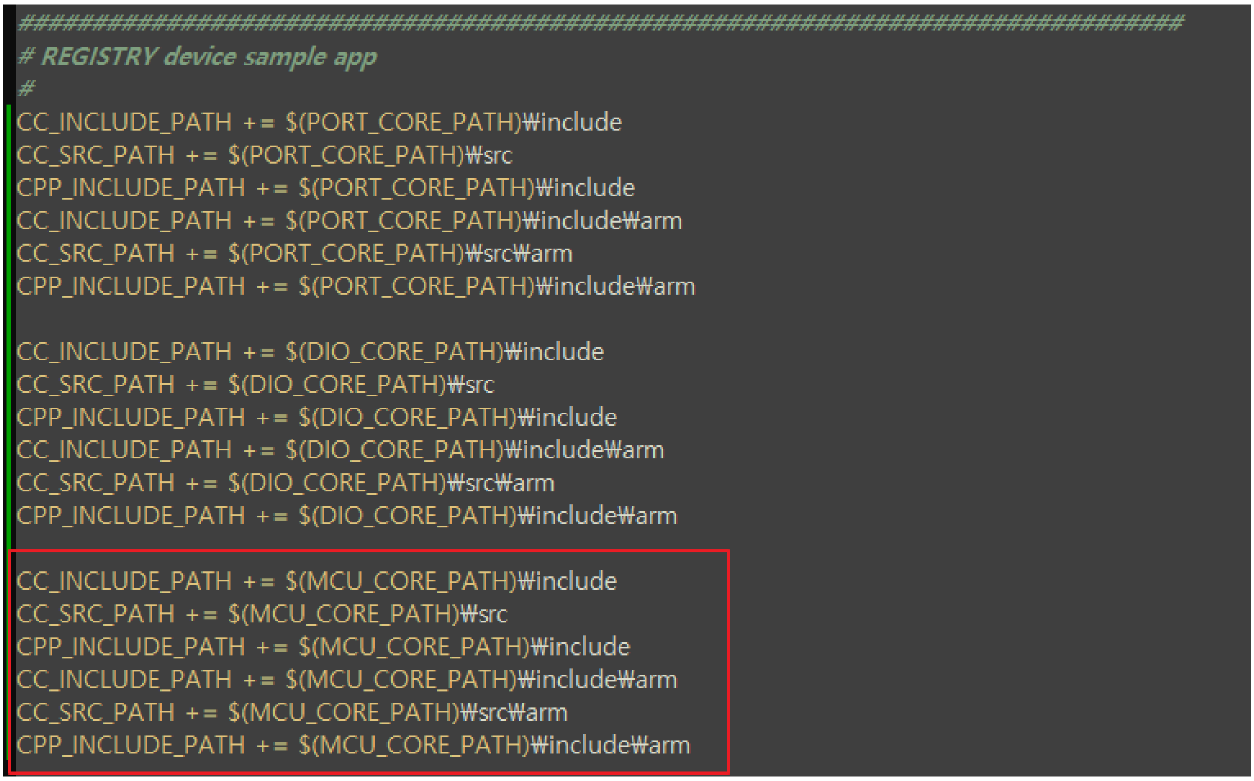


```
################################################################################
# Define ISR identifier of each module in vector table
#
CFLAGS += -DPORT_MODULE_SAMPLE
CFLAGS += -DDIO_MODULE_SAMPLE
CFLAGS += -DMCU_MODULE_SAMPLE
```

Include Path 추가

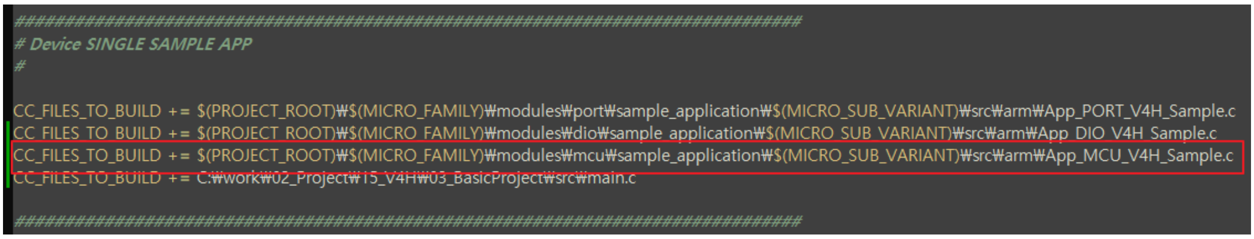


```
################################################################################
# REGISTRY common sample app
#
CC_INCLUDE_PATH += $(PORT_COMMON_SAMPLE_CORE_PATH)₩include
CC_SRC_PATH += $(PORT_COMMON_SAMPLE_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(PORT_COMMON_SAMPLE_CORE_PATH)₩include
ASM_INCLUDE_PATH +=
PREPROCESSOR_DEFINES +=

CC_INCLUDE_PATH += $(DIO_COMMON_SAMPLE_CORE_PATH)₩include
CC_SRC_PATH += $(DIO_COMMON_SAMPLE_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(DIO_COMMON_SAMPLE_CORE_PATH)₩include
ASM_INCLUDE_PATH +=
PREPROCESSOR_DEFINES +=

CC_INCLUDE_PATH += $(MCU_COMMON_SAMPLE_CORE_PATH)₩include
CC_SRC_PATH += $(MCU_COMMON_SAMPLE_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(MCU_COMMON_SAMPLE_CORE_PATH)₩include
ASM_INCLUDE_PATH +=
PREPROCESSOR_DEFINES +=
```



```
##########################################################################################
# REGISTRY device sample app
#
CC_INCLUDE_PATH += $(PORT_CORE_PATH)₩include
CC_SRC_PATH += $(PORT_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(PORT_CORE_PATH)₩include
CC_INCLUDE_PATH += $(PORT_CORE_PATH)₩include₩arm
CC_SRC_PATH += $(PORT_CORE_PATH)₩src₩arm
CPP_INCLUDE_PATH += $(PORT_CORE_PATH)₩include₩arm

CC_INCLUDE_PATH += $(DIO_CORE_PATH)₩include
CC_SRC_PATH += $(DIO_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(DIO_CORE_PATH)₩include
CC_INCLUDE_PATH += $(DIO_CORE_PATH)₩include₩arm
CC_SRC_PATH += $(DIO_CORE_PATH)₩src₩arm
CPP_INCLUDE_PATH += $(DIO_CORE_PATH)₩include₩arm

CC_INCLUDE_PATH += $(MCU_CORE_PATH)₩include
CC_SRC_PATH += $(MCU_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(MCU_CORE_PATH)₩include
CC_INCLUDE_PATH += $(MCU_CORE_PATH)₩include₩arm
CC_SRC_PATH += $(MCU_CORE_PATH)₩src₩arm
CPP_INCLUDE_PATH += $(MCU_CORE_PATH)₩include₩arm
```

Module main 실행파일 추가



```
##########################################################################
# Device SINGLE SAMPLE APP
#

CC_FILES_TO_BUILD += $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩port₩sample_application₩$(MICRO_SUB_VARIANT)₩src₩arm₩App_PORT_V4H_Sample.c
CC_FILES_TO_BUILD += $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩dio₩sample_application₩$(MICRO_SUB_VARIANT)₩src₩arm₩App_DIO_V4H_Sample.c
CC_FILES_TO_BUILD += $(PROJECT_ROOT)₩$(MICRO_FAMILY)₩modules₩mcu₩sample_application₩$(MICRO_SUB_VARIANT)₩src₩arm₩App_MCU_V4H_Sample.c
CC_FILES_TO_BUILD += C:₩work₩02_Project₩15_V4H₩03_BasicProject₩src₩main.c

##########################################################################
```

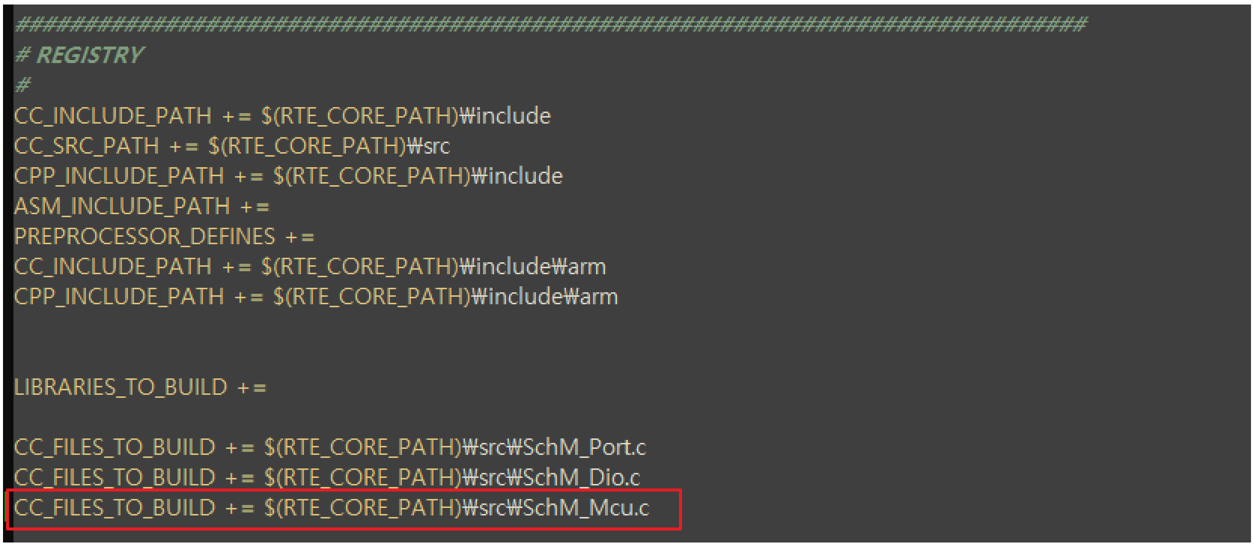
7.3.3 Stub File

Common*.mak 파일 수정이 완료되면 Stub make file 도 일부 변경을 해야 한다.

Stub.mak 파일을 open 하여 SchM 및 RTE 파일을 추가해야 한다.

Path: C:\work\02_Project\15_V4H\03_BasicProject\src\05_Mcal\rel\common\generic\stubs\19_11\Rte\src\

SchM_xxx.c Stub 파일 추가
(C:\work\02_Project\15_V4H\03_BasicProject\src\05_Mcal\rel\V4H\common_family\make\arm\custom\)



```
##########################################################################################
# REGISTRY
#
CC_INCLUDE_PATH += $(RTE_CORE_PATH)₩include
CC_SRC_PATH += $(RTE_CORE_PATH)₩src
CPP_INCLUDE_PATH += $(RTE_CORE_PATH)₩include
ASM_INCLUDE_PATH +=
PREPROCESSOR_DEFINES +=
CC_INCLUDE_PATH += $(RTE_CORE_PATH)₩include₩arm
CPP_INCLUDE_PATH += $(RTE_CORE_PATH)₩include₩arm


LIBRARIES_TO_BUILD +=

CC_FILES_TO_BUILD += $(RTE_CORE_PATH)₩src₩SchM_Port.c
CC_FILES_TO_BUILD += $(RTE_CORE_PATH)₩src₩SchM_Dio.c
CC_FILES_TO_BUILD += $(RTE_CORE_PATH)₩src₩SchM_Mcu.c
```

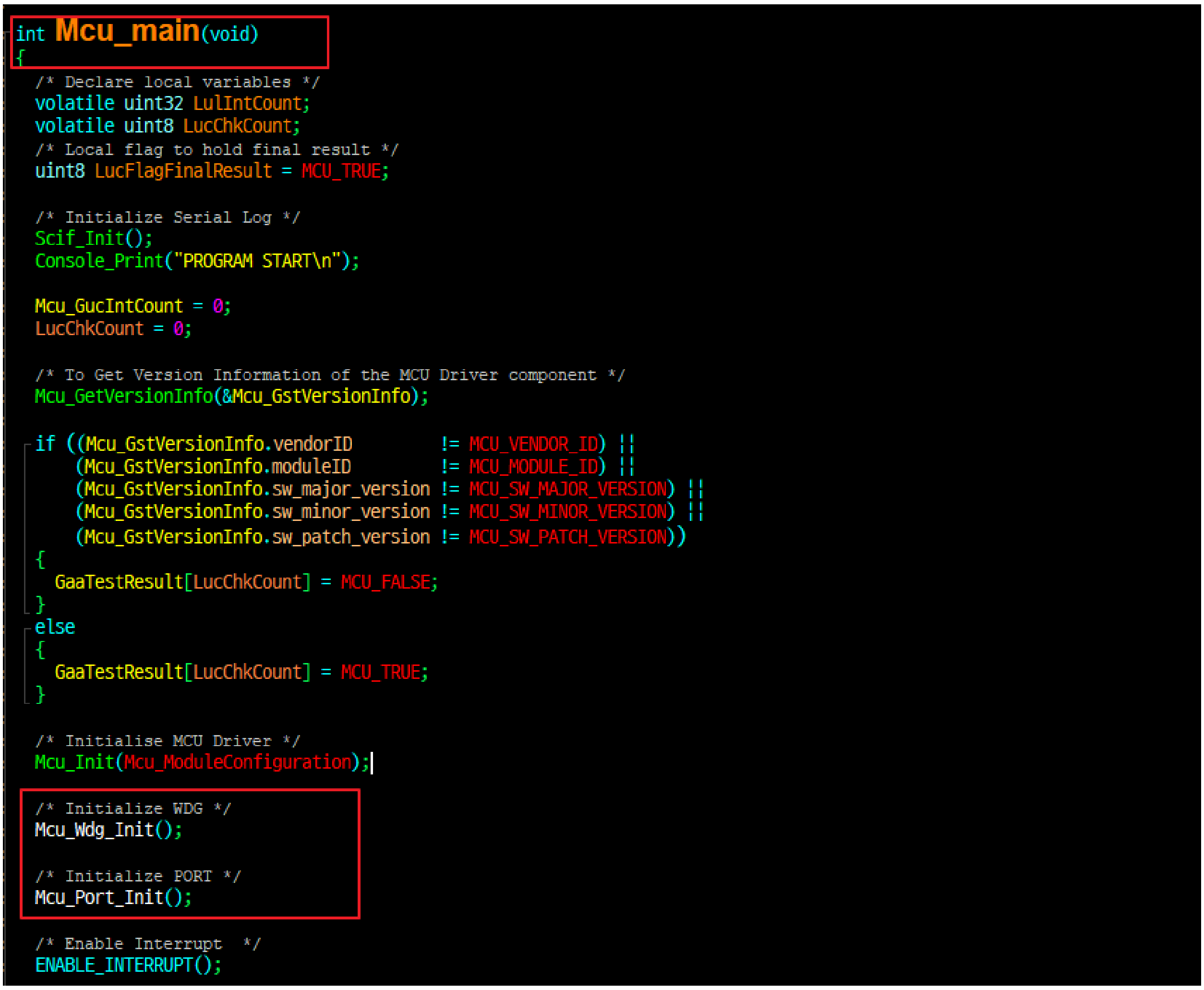
7.4 Sample Application code modification

각 모듈은 같은 Sample Application 테스트를 위하여 같은 함수 및 변수를 사용한다. 예를 들면, Mcu_Init, Wdg_Init, Port_Init 등이 있다.

그래서 prefix 를 추가하여 충돌을 방지한다.

C:\work\02_Project\15_V4H\03_BasicProject\src\05_Mcal\rel\modules\mcu\sample_application\V4H\src\arm\

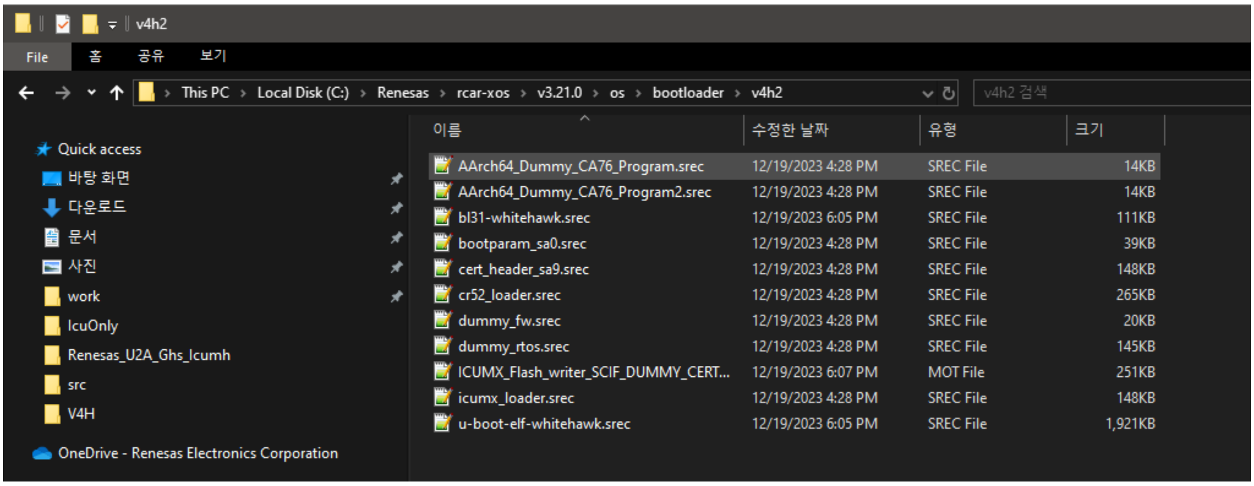
아래 Snap shot 과 같이 “Mcu_” 와 같은 Prefix 를 추가한다.



```
int Mcu_main(void)
{
  /* Declare local variables */
  volatile uint32 LulIntCount;
  volatile uint8 LucChkCount;
  /* Local flag to hold final result */
  uint8 LucFlagFinalResult = MCU_TRUE;

  /* Initialize Serial Log */
  Scif_Init();
  Console_Print("PROGRAM START\n");

  Mcu_GucIntCount = 0;
  LucChkCount = 0;

  /* To Get Version Information of the MCU Driver component */
  Mcu_GetVersionInfo(&Mcu_GstVersionInfo);

  if ((Mcu_GstVersionInfo.vendorID         != MCU_VENDOR_ID) ||
      (Mcu_GstVersionInfo.moduleID         != MCU_MODULE_ID) ||
      (Mcu_GstVersionInfo.sw_major_version != MCU_SW_MAJOR_VERSION) ||
      (Mcu_GstVersionInfo.sw_minor_version != MCU_SW_MINOR_VERSION) ||
      (Mcu_GstVersionInfo.sw_patch_version != MCU_SW_PATCH_VERSION))
  {
    GaaTestResult[LucChkCount] = MCU_FALSE;
  }
  else
  {
    GaaTestResult[LucChkCount] = MCU_TRUE;
  }

  /* Initialise MCU Driver */
  Mcu_Init(Mcu_ModuleConfiguration);

  /* Initialize WDG */
  Mcu_Wdg_Init();

  /* Initialize PORT */
  Mcu_Port_Init();

  /* Enable Interrupt  */
  ENABLE_INTERRUPT();
```

8 Teraterm Download

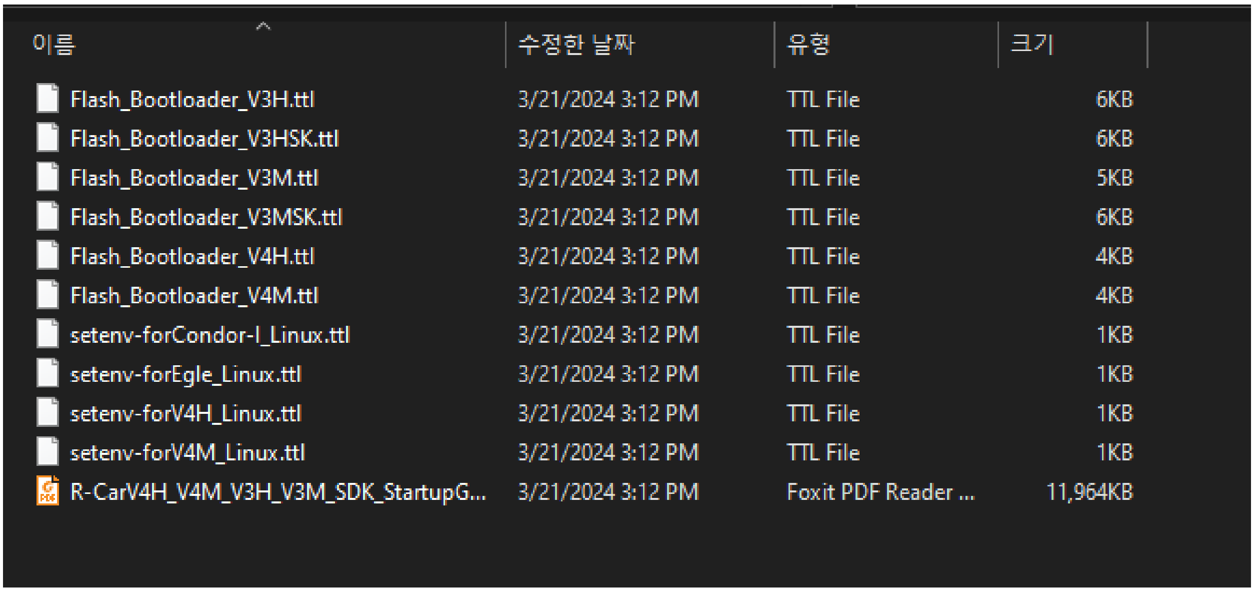
V4H2 폴더에서 아래와 같이 Default Hex 파일들을 확인한다.



v4h2
File
홈
공유
보기
This PC › Local Disk (C:) › Renesas › rcar-xos › v3.21.0 › os › bootloader › v4h2
v4h2 검색
Quick access
바탕 화면
다운로드
문서
사진
work
IcuOnly
Renesas_U2A_Ghs_Icumh
src
V4H
OneDrive - Renesas Electronics Corporation

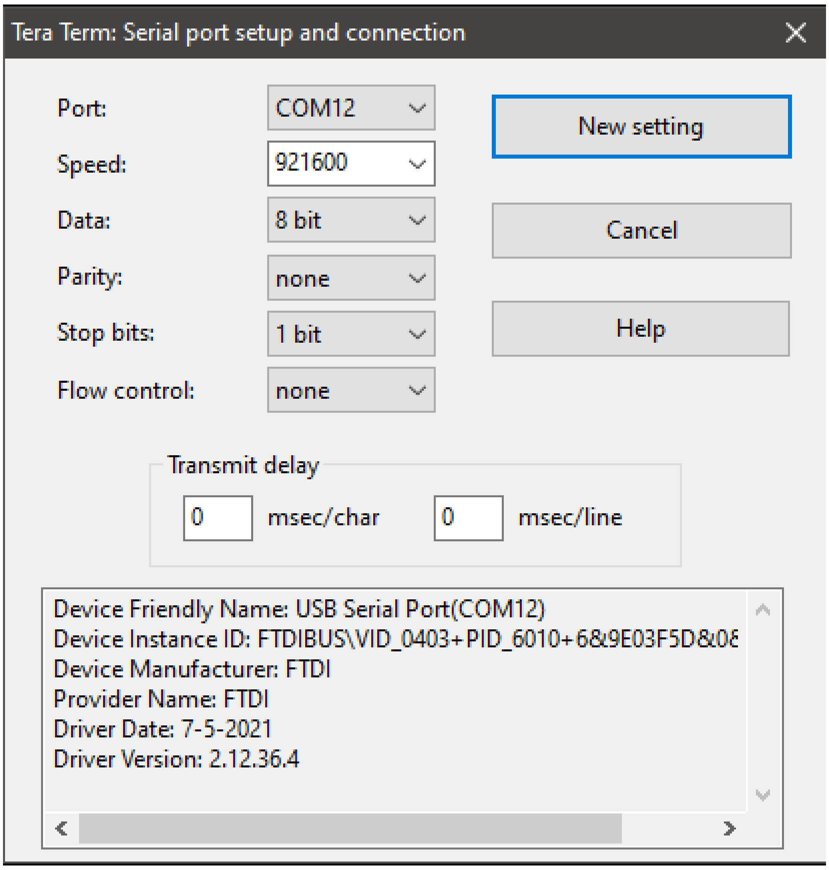
이름	수정한 날짜	유형	크기
AArch64_Dummy_CA76_Program.srec	12/19/2023 4:28 PM	SREC File	14KB
AArch64_Dummy_CA76_Program2.srec	12/19/2023 4:28 PM	SREC File	14KB
bl31-whitehawk.srec	12/19/2023 6:05 PM	SREC File	111KB
bootparam_sa0.srec	12/19/2023 4:28 PM	SREC File	39KB
cert_header_sa9.srec	12/19/2023 4:28 PM	SREC File	148KB
cr52_loader.srec	12/19/2023 4:28 PM	SREC File	265KB
dummy_fw.srec	12/19/2023 4:28 PM	SREC File	20KB
dummy_rtos.srec	12/19/2023 4:28 PM	SREC File	145KB
ICUMX_Flash_writer_SCIF_DUMMY_CERT...	12/19/2023 6:07 PM	MOT File	251KB
icumx_loader.srec	12/19/2023 4:28 PM	SREC File	148KB
u-boot-elf-whitehawk.srec	12/19/2023 6:05 PM	SREC File	1,921KB

SDK 스타트업 가이드를 다운받아 아래와 같이 teraterm 스크립트를 확인한다.



이름	수정한 날짜	유형	크기
Flash_Bootloader_V3H.ttl	3/21/2024 3:12 PM	TTL File	6KB
Flash_Bootloader_V3HSK.ttl	3/21/2024 3:12 PM	TTL File	6KB
Flash_Bootloader_V3M.ttl	3/21/2024 3:12 PM	TTL File	5KB
Flash_Bootloader_V3MSK.ttl	3/21/2024 3:12 PM	TTL File	6KB
Flash_Bootloader_V4H.ttl	3/21/2024 3:12 PM	TTL File	4KB
Flash_Bootloader_V4M.ttl	3/21/2024 3:12 PM	TTL File	4KB
setenv-forCondor-I_Linux.ttl	3/21/2024 3:12 PM	TTL File	1KB
setenv-forEgle_Linux.ttl	3/21/2024 3:12 PM	TTL File	1KB
setenv-forV4H_Linux.ttl	3/21/2024 3:12 PM	TTL File	1KB
setenv-forV4M_Linux.ttl	3/21/2024 3:12 PM	TTL File	1KB
R-CarV4H_V4M_V3H_V3M_SDK_StartupG...	3/21/2024 3:12 PM	Foxit PDF Reader ...	11,964KB

8.1 Teraterm 설정



Tera Term: Serial port setup and connection

Port: COM12

Speed: 921600

Data: 8 bit

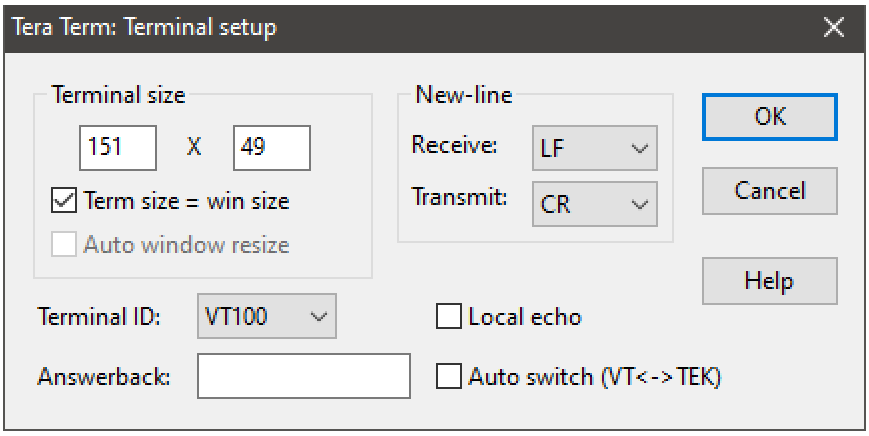
Parity: none

Stop bits: 1 bit

Flow control: none

Transmit delay: 0 msec/char, 0 msec/line

Device Friendly Name: USB Serial Port(COM12)
Device Instance ID: FTDIBUS\VID_0403+PID_6010+6&9E03F5D&0&
Device Manufacturer: FTDI
Provider Name: FTDI
Driver Date: 7-5-2021
Driver Version: 2.12.36.4



Tera Term: Terminal setup

Terminal size: 151 X 49

☑ Term size = win size

□ Auto window resize

New-line: Receive: LF, Transmit: CR

Terminal ID: VT100

Answerback:

□ Local echo

□ Auto switch (VT<->TEK)

8.2 스위치 설정

SCIF 모드로 설정한다.

Step 3 Switch setting (SCIF download mode)
The Switch setting to execute Flash writer is shown below.

Table.5-1-1: White Hawk board(CPU Board)

SW13	Pin 1
SW57	Center

Table.5-1-2: White Hawk board(Mode Switch Board)

SW	Pin1	Pin2	Pin3	Pin4	Pin5	Pin6	Pin7	Pin8
SW1	OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF
SW2	OFF	ON	ON	ON	OFF	ON	OFF	OFF
SW3	ON	ON	ON	ON	OFF	ON	OFF	OFF
SW4	ON	ON	ON	ON	ON	ON	ON	ON
SW5	ON	ON	ON	ON	ON	OFF	OFF	OFF

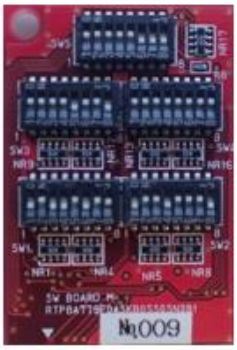


Figure.5-1-4: Mode Switch Board

Step 4 Power ON(SW51) cpu board
Switch SW51 to turn on the power.

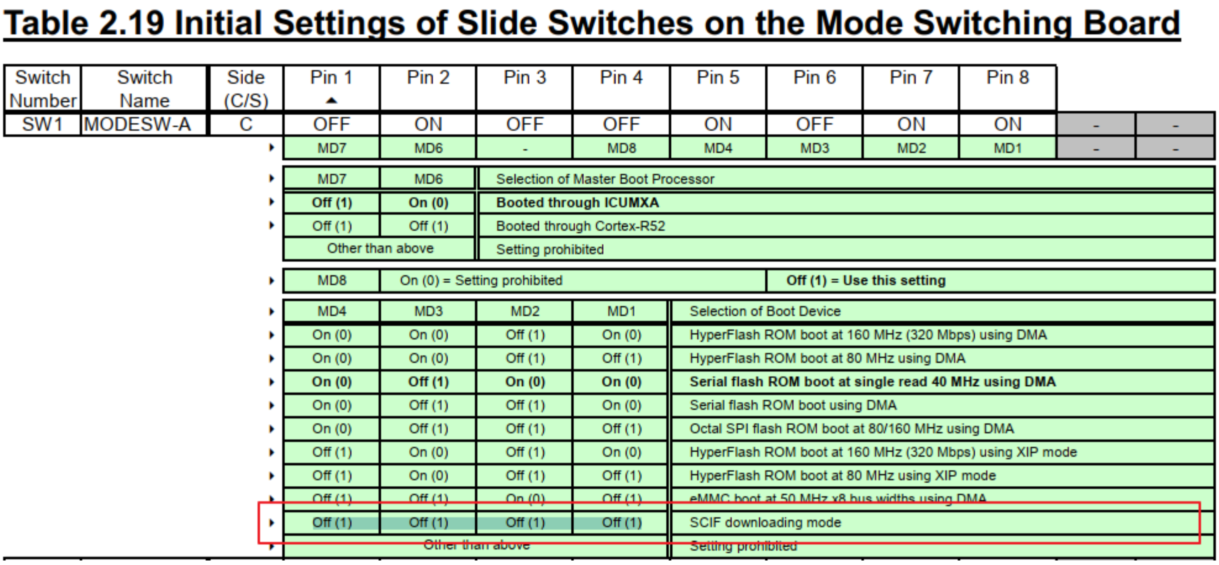


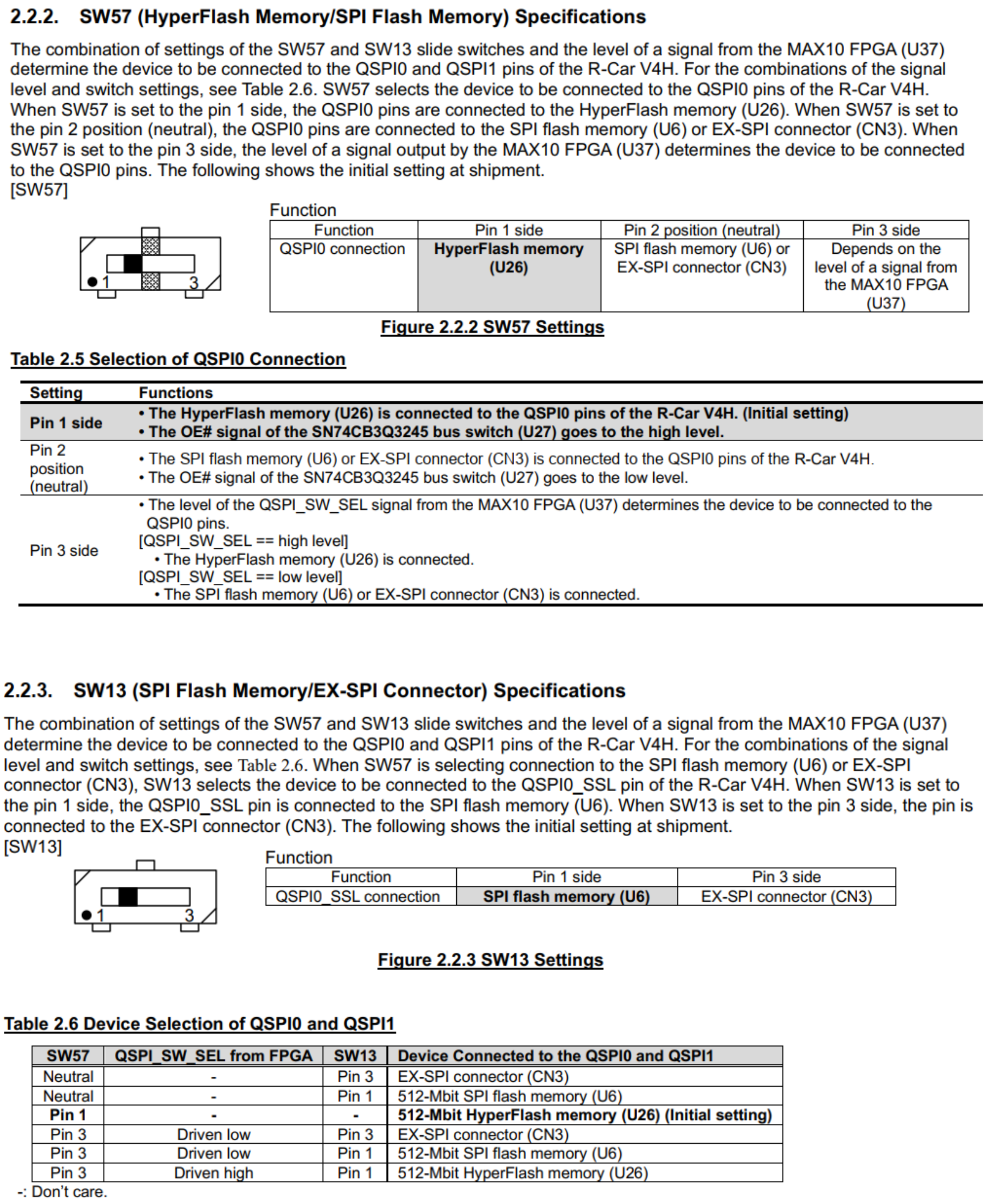
Table 2.19 Initial Settings of Slide Switches on the Mode Switching Board

Switch Number	Switch Name	Side (C/S)	Pin 1 ▲	Pin 2	Pin 3	Pin 4	Pin 5	Pin 6	Pin 7	Pin 8		
SW1	MODESW-A	C	OFF	ON	OFF	OFF	ON	OFF	ON	ON	-	-
			MD7	MD6	-	MD8	MD4	MD3	MD2	MD1	-	-

MD7	MD6	Selection of Master Boot Processor
Off (1)	**On (0)**	**Booted through ICUMXA**
Off (1)	Off (1)	Booted through Cortex-R52
Other than above		Setting prohibited

MD8	On (0) = Setting prohibited	**Off (1) = Use this setting**

MD4	MD3	MD2	MD1	Selection of Boot Device
On (0)	On (0)	Off (1)	On (0)	HyperFlash ROM boot at 160 MHz (320 Mbps) using DMA
On (0)	On (0)	Off (1)	Off (1)	HyperFlash ROM boot at 80 MHz using DMA
On (0)	**Off (1)**	**On (0)**	**On (0)**	**Serial flash ROM boot at single read 40 MHz using DMA**
On (0)	Off (1)	Off (1)	On (0)	Serial flash ROM boot using DMA
On (0)	Off (1)	Off (1)	Off (1)	Octal SPI flash ROM boot at 80/160 MHz using DMA
Off (1)	On (0)	Off (1)	On (0)	HyperFlash ROM boot at 160 MHz (320 Mbps) using XIP mode
Off (1)	On (0)	Off (1)	Off (1)	HyperFlash ROM boot at 80 MHz using XIP mode
Off (1)	Off (1)	On (0)	Off (1)	eMMC boot at 50 MHz x8 bus widths using DMA
Off (1)	Off (1)	Off (1)	Off (1)	SCIF downloading mode
Other than above				Setting prohibited



2.2.2. SW57 (HyperFlash Memory/SPI Flash Memory) Specifications

The combination of settings of the SW57 and SW13 slide switches and the level of a signal from the MAX10 FPGA (U37) determine the device to be connected to the QSPI0 and QSPI1 pins of the R-Car V4H. For the combinations of the signal level and switch settings, see Table 2.6. SW57 selects the device to be connected to the QSPI0 pins of the R-Car V4H. When SW57 is set to the pin 1 side, the QSPI0 pins are connected to the HyperFlash memory (U26). When SW57 is set to the pin 2 position (neutral), the QSPI0 pins are connected to the SPI flash memory (U6) or EX-SPI connector (CN3). When SW57 is set to the pin 3 side, the level of a signal output by the MAX10 FPGA (U37) determines the device to be connected to the QSPI0 pins. The following shows the initial setting at shipment.

[SW57]

Function

Function	Pin 1 side	Pin 2 position (neutral)	Pin 3 side
QSPI0 connection	**HyperFlash memory (U26)**	SPI flash memory (U6) or EX-SPI connector (CN3)	Depends on the level of a signal from the MAX10 FPGA (U37)

Figure 2.2.2 SW57 Settings

Table 2.5 Selection of QSPI0 Connection

Setting	Functions
Pin 1 side	**• The HyperFlash memory (U26) is connected to the QSPI0 pins of the R-Car V4H. (Initial setting)** **• The OE# signal of the SN74CB3Q3245 bus switch (U27) goes to the high level.**
Pin 2 position (neutral)	• The SPI flash memory (U6) or EX-SPI connector (CN3) is connected to the QSPI0 pins of the R-Car V4H. • The OE# signal of the SN74CB3Q3245 bus switch (U27) goes to the low level.
Pin 3 side	• The level of the QSPI_SW_SEL signal from the MAX10 FPGA (U37) determines the device to be connected to the QSPI0 pins. [QSPI_SW_SEL == high level] • The HyperFlash memory (U26) is connected. [QSPI_SW_SEL == low level] • The SPI flash memory (U6) or EX-SPI connector (CN3) is connected.

2.2.3. SW13 (SPI Flash Memory/EX-SPI Connector) Specifications

The combination of settings of the SW57 and SW13 slide switches and the level of a signal from the MAX10 FPGA (U37) determine the device to be connected to the QSPI0 and QSPI1 pins of the R-Car V4H. For the combinations of the signal level and switch settings, see Table 2.6. When SW57 is selecting connection to the SPI flash memory (U6) or EX-SPI connector (CN3), SW13 selects the device to be connected to the QSPI0_SSL pin of the R-Car V4H. When SW13 is set to the pin 1 side, the QSPI0_SSL pin is connected to the SPI flash memory (U6). When SW13 is set to the pin 3 side, the pin is connected to the EX-SPI connector (CN3). The following shows the initial setting at shipment.

[SW13]

Function

Function	Pin 1 side	Pin 3 side
QSPI0_SSL connection	**SPI flash memory (U6)**	EX-SPI connector (CN3)

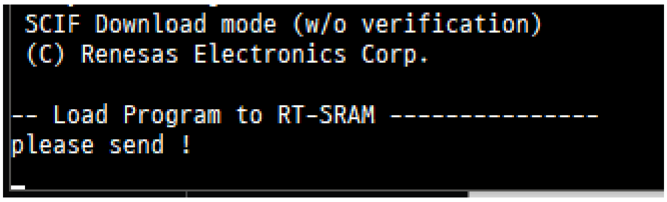
Figure 2.2.3 SW13 Settings

Table 2.6 Device Selection of QSPI0 and QSPI1

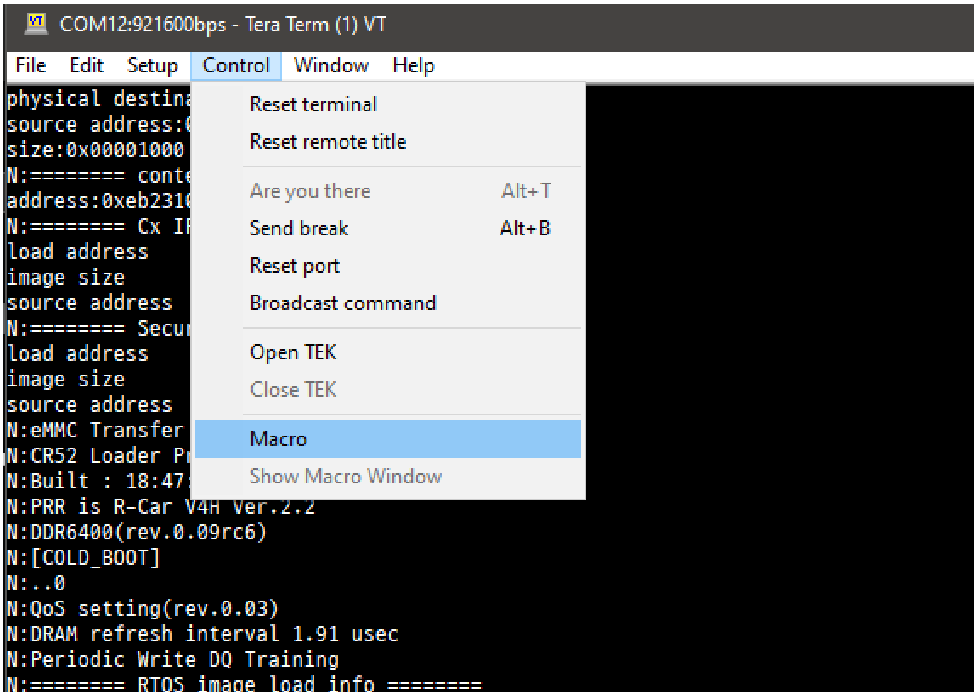
SW57	QSPI_SW_SEL from FPGA	SW13	Device Connected to the QSPI0 and QSPI1
Neutral	-	Pin 3	EX-SPI connector (CN3)
Neutral	-	Pin 1	512-Mbit SPI flash memory (U6)
Pin 1	**-**	**-**	**512-Mbit HyperFlash memory (U26) (Initial setting)**
Pin 3	Driven low	Pin 3	EX-SPI connector (CN3)
Pin 3	Driven low	Pin 1	512-Mbit SPI flash memory (U6)
Pin 3	Driven high	Pin 1	512-Mbit HyperFlash memory (U26)

-: Don't care.

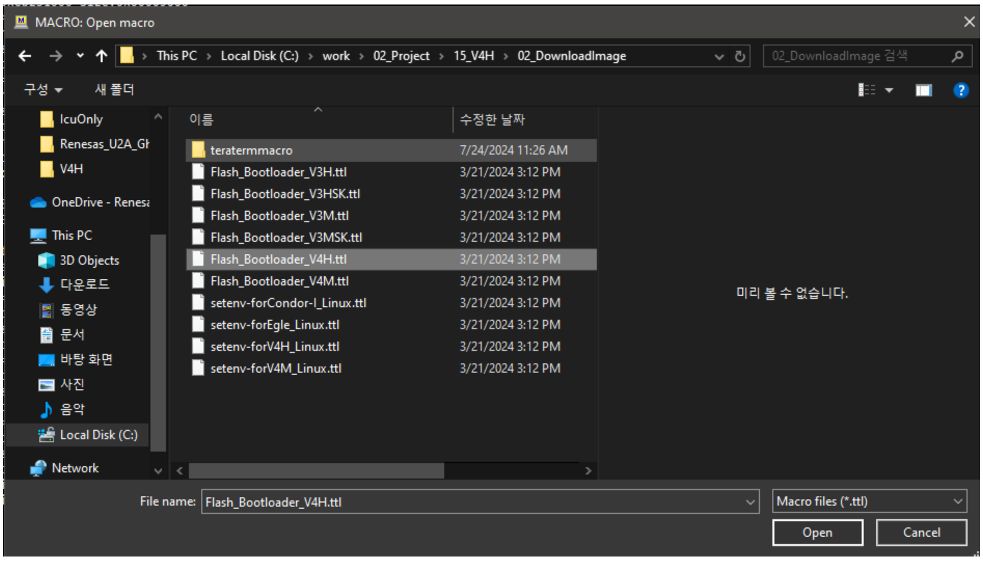
8.3 Download



```
 SCIF Download mode (w/o verification)
 (C) Renesas Electronics Corp.

-- Load Program to RT-SRAM ---------------
please send !
_
```



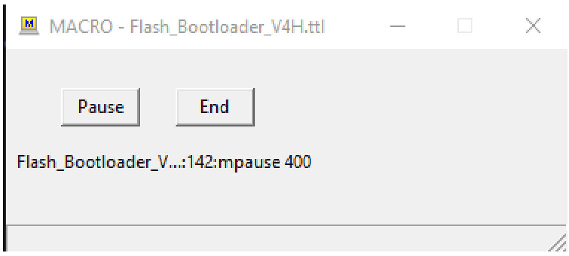
```
physical destina
source address:
size:0x00001000
N:======== conte
address:0xeb231
N:======== Cx IP
load address
image size
source address
N:======== Secu
load address
image size
source address
N:eMMC Transfer
N:CR52 Loader P
N:Built : 18:47:
N:PRR is R-Car V4H Ver.2.2
N:DDR6400(rev.0.09rc6)
N:[COLD_BOOT]
N:..0
N:QoS setting(rev.0.03)
N:DRAM refresh interval 1.91 usec
N:Periodic Write DQ Training
N:======== RTOS image load info ========
```

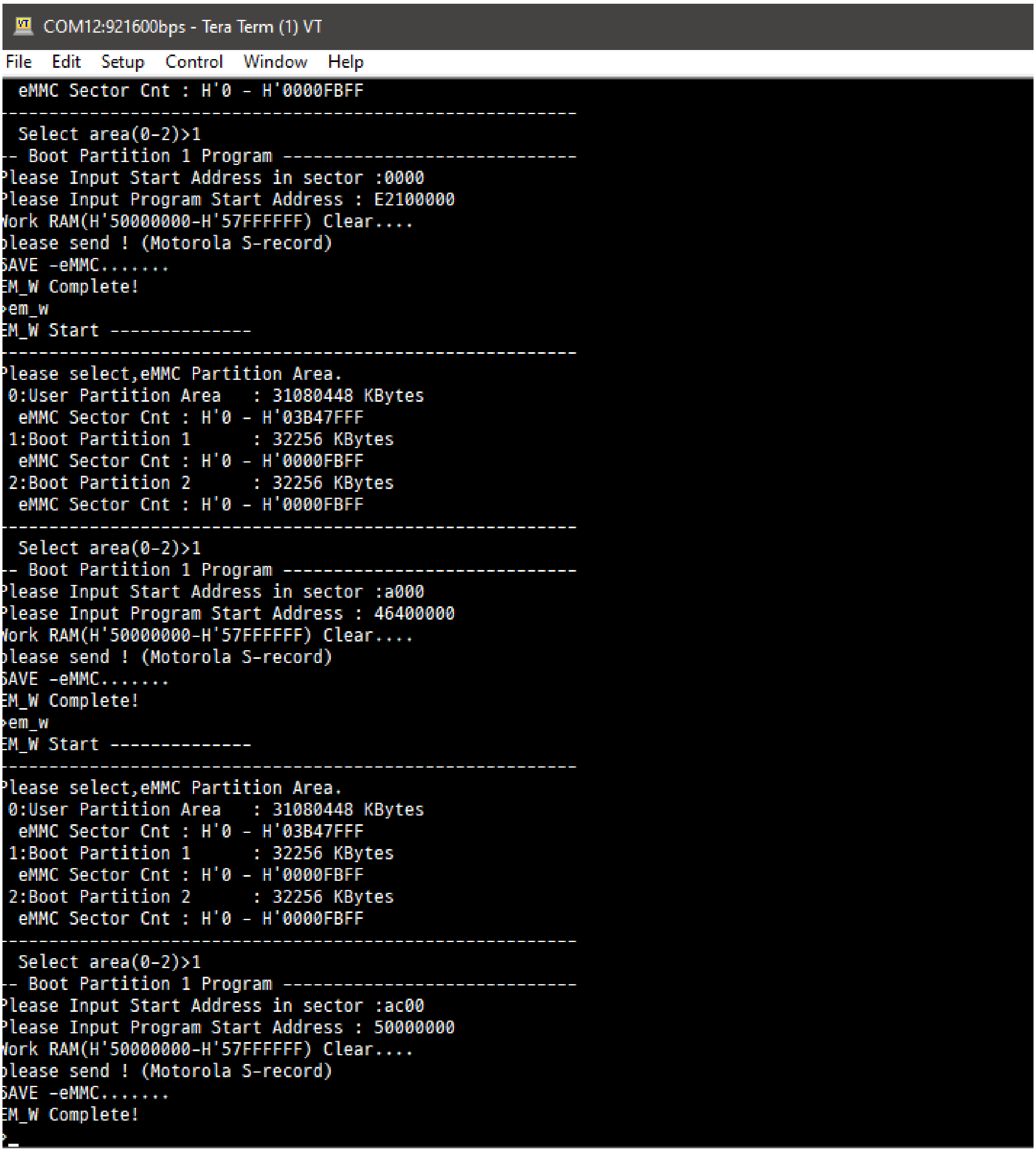


이름	수정한 날짜
teratermmacro	7/24/2024 11:26 AM
Flash_Bootloader_V3H.ttl	3/21/2024 3:12 PM
Flash_Bootloader_V3HSK.ttl	3/21/2024 3:12 PM
Flash_Bootloader_V3M.ttl	3/21/2024 3:12 PM
Flash_Bootloader_V3MSK.ttl	3/21/2024 3:12 PM
Flash_Bootloader_V4H.ttl	3/21/2024 3:12 PM
Flash_Bootloader_V4M.ttl	3/21/2024 3:12 PM
setenv-forCondor-I_Linux.ttl	3/21/2024 3:12 PM
setenv-forEgle_Linux.ttl	3/21/2024 3:12 PM
setenv-forV4H_Linux.ttl	3/21/2024 3:12 PM
setenv-forV4M_Linux.ttl	3/21/2024 3:12 PM

File name: Flash_Bootloader_V4H.ttl

Macro files (*.ttl)

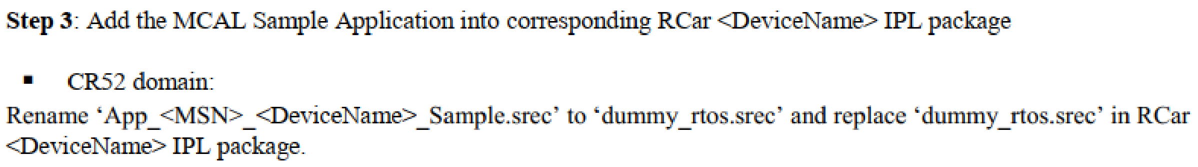

MACRO - Flash_Bootloader_V4H.ttl
Pause
End
Flash_Bootloader_V...:142:mpause 400


COM12:921600bps - Tera Term (1) VT
File Edit Setup Control Window Help

```
  eMMC Sector Cnt : H'0 - H'0000FBFF
---------------------------------------------------------
  Select area(0-2)>1
-- Boot Partition 1 Program -----------------------------
Please Input Start Address in sector :0000
Please Input Program Start Address : E2100000
Work RAM(H'50000000-H'57FFFFFF) Clear....
please send ! (Motorola S-record)
SAVE -eMMC.......
EM_W Complete!
>em_w
EM_W Start --------------
---------------------------------------------------------
Please select,eMMC Partition Area.
 0:User Partition Area    : 31080448 KBytes
  eMMC Sector Cnt : H'0 - H'03B47FFF
 1:Boot Partition 1       : 32256 KBytes
  eMMC Sector Cnt : H'0 - H'0000FBFF
 2:Boot Partition 2       : 32256 KBytes
  eMMC Sector Cnt : H'0 - H'0000FBFF
---------------------------------------------------------
  Select area(0-2)>1
-- Boot Partition 1 Program -----------------------------
Please Input Start Address in sector :a000
Please Input Program Start Address : 46400000
Work RAM(H'50000000-H'57FFFFFF) Clear....
please send ! (Motorola S-record)
SAVE -eMMC.......
EM_W Complete!
>em_w
EM_W Start --------------
---------------------------------------------------------
Please select,eMMC Partition Area.
 0:User Partition Area    : 31080448 KBytes
  eMMC Sector Cnt : H'0 - H'03B47FFF
 1:Boot Partition 1       : 32256 KBytes
  eMMC Sector Cnt : H'0 - H'0000FBFF
 2:Boot Partition 2       : 32256 KBytes
  eMMC Sector Cnt : H'0 - H'0000FBFF
---------------------------------------------------------
  Select area(0-2)>1
-- Boot Partition 1 Program -----------------------------
Please Input Start Address in sector :ac00
Please Input Program Start Address : 50000000
Work RAM(H'50000000-H'57FFFFFF) Clear....
please send ! (Motorola S-record)
SAVE -eMMC.......
EM_W Complete!
>_
```

8.3.1 MCAL Download

만약 MCAL 의 헥사파일을 다운로드 하고싶으면 아래와 같이 헥사 파일 이름을 변경하여 다운로드 하면 된다.



Step 3: Add the MCAL Sample Application into corresponding RCar <DeviceName> IPL package

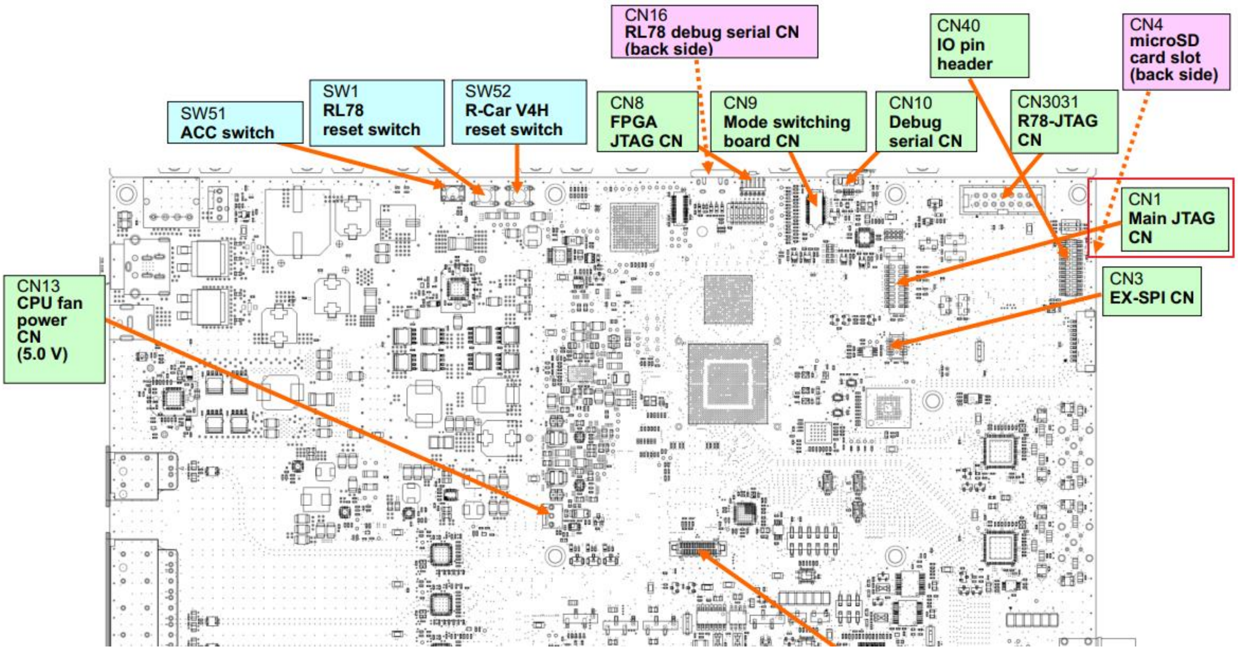
- CR52 domain:

Rename ‘App_<MSN>_<DeviceName>_Sample.srec’ to ‘dummy_rtos.srec’ and replace ‘dummy_rtos.srec’ in RCar <DeviceName> IPL package.

9 TRACE32 사용

9.1 하드웨어 구성

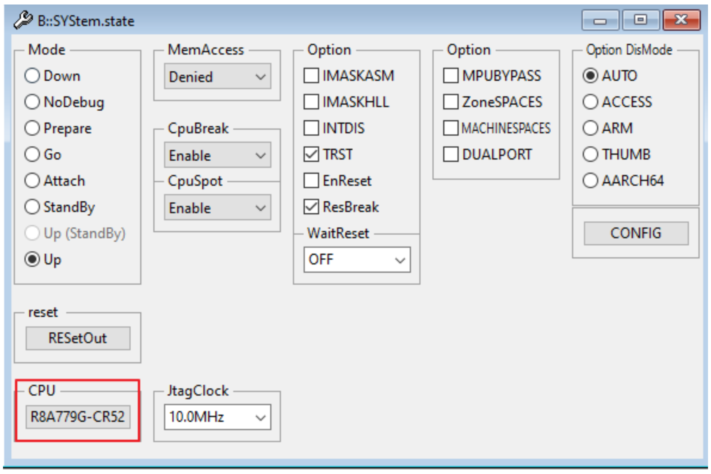
1.1. Locations of Connectors on the White Hawk Board



TRACE32 를 Main JTAG CN 에 연결한다

9.2 Trace 32 Connection method

1. Connection (ex: warm reset script)



B::SYStem.state

Mode: ○ Down ○ NoDebug ○ Prepare ○ Go ○ Attach ○ StandBy ○ Up (StandBy) ● Up

MemAccess: Denied

CpuBreak: Enable

CpuSpot: Enable

Option: □ IMASKASM □ IMASKHLL □ INTDIS ☑ TRST □ EnReset ☑ ResBreak

WaitReset: OFF

Option: □ MPUBYPASS □ ZoneSPACES □ MACHINESPACES □ DUALPORT

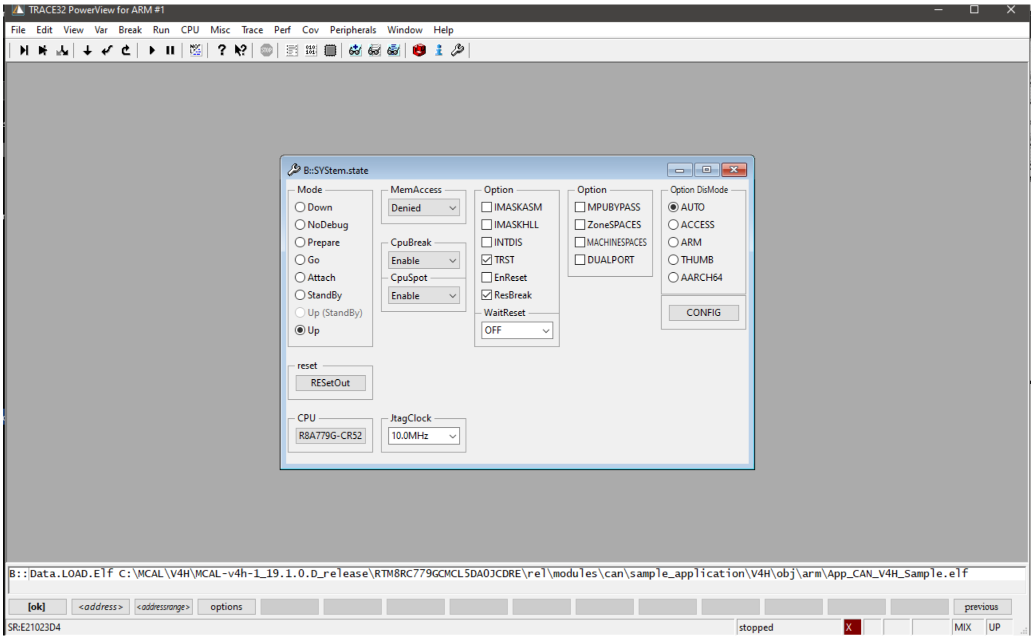
Option DisMode: ● AUTO ○ ACCESS ○ ARM ○ THUMB ○ AARCH64

CPU: R8A779G-CR52

JtagClock: 10.0MHz

2. Attached 후 halt

3. Elf Loading

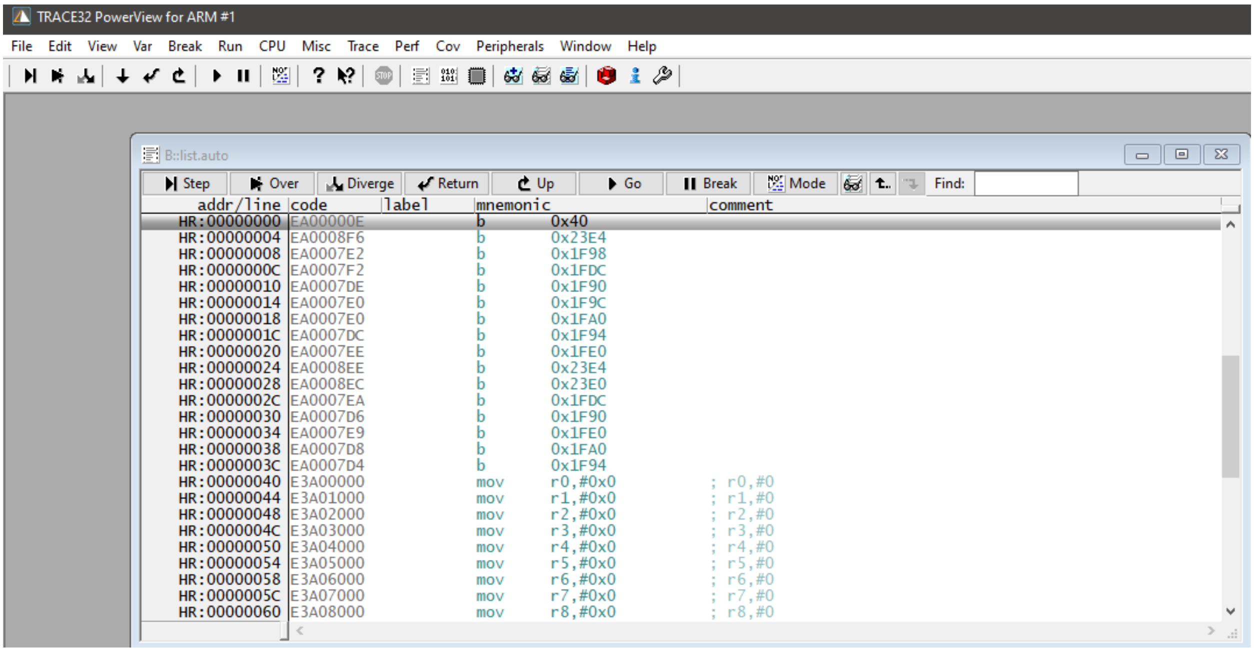


TRACE32 PowerView for ARM #1
File Edit View Var Break Run CPU Misc Trace Perf Cov Peripherals Window Help
B::SYStem.state
[ok]
<address>
<addressrange>
options
previous
stopped
X
MIX
UP

```
B::Data.LOAD.Elf C:\MCAL\V4H\MCAL-v4h-1_19.1.0.D_release\RTM8RC779GCMCL5DA0JCDRE\rel\modules\can\sample_application\V4H\obj\arm\App_CAN_V4H_Sample.elf
```

SR:E21023D4

4. Warm reset script


TRACE32 PowerView for ARM #1
File Edit View Var Break Run CPU Misc Trace Perf Cov Peripherals Window Help
B::list.auto
Step
Over
Diverge
Return
Up
Go
Break
Mode
Find:

```
    addr/line code       label   mnemonic                comment
 HR:00000000 EA00000E            b          0x40
 HR:00000004 EA0008F6            b          0x23E4
 HR:00000008 EA0007E2            b          0x1F98
 HR:0000000C EA0007F2            b          0x1FDC
 HR:00000010 EA0007DE            b          0x1F90
 HR:00000014 EA0007E0            b          0x1F9C
 HR:00000018 EA0007E0            b          0x1FA0
 HR:0000001C EA0007DC            b          0x1F94
 HR:00000020 EA0007EE            b          0x1FE0
 HR:00000024 EA0008EE            b          0x23E4
 HR:00000028 EA0008EC            b          0x23E0
 HR:0000002C EA0007EA            b          0x1FDC
 HR:00000030 EA0007D6            b          0x1F90
 HR:00000034 EA0007E9            b          0x1FE0
 HR:00000038 EA0007D8            b          0x1FA0
 HR:0000003C EA0007D4            b          0x1F94
 HR:00000040 E3A00000            mov        r0,#0x0       ; r0,#0
 HR:00000044 E3A01000            mov        r1,#0x0       ; r1,#0
 HR:00000048 E3A02000            mov        r2,#0x0       ; r2,#0
 HR:0000004C E3A03000            mov        r3,#0x0       ; r3,#0
 HR:00000050 E3A04000            mov        r4,#0x0       ; r4,#0
 HR:00000054 E3A05000            mov        r5,#0x0       ; r5,#0
 HR:00000058 E3A06000            mov        r6,#0x0       ; r6,#0
 HR:0000005C E3A07000            mov        r7,#0x0       ; r7,#0
 HR:00000060 E3A08000            mov        r8,#0x0       ; r8,#0
```

5. Go main (Enter)

10 Booting Sequence

10.1 Boot Rom

RCAR/V4H 의 내부 144Kbytes Secure Rom 을 갖고 있습니다.

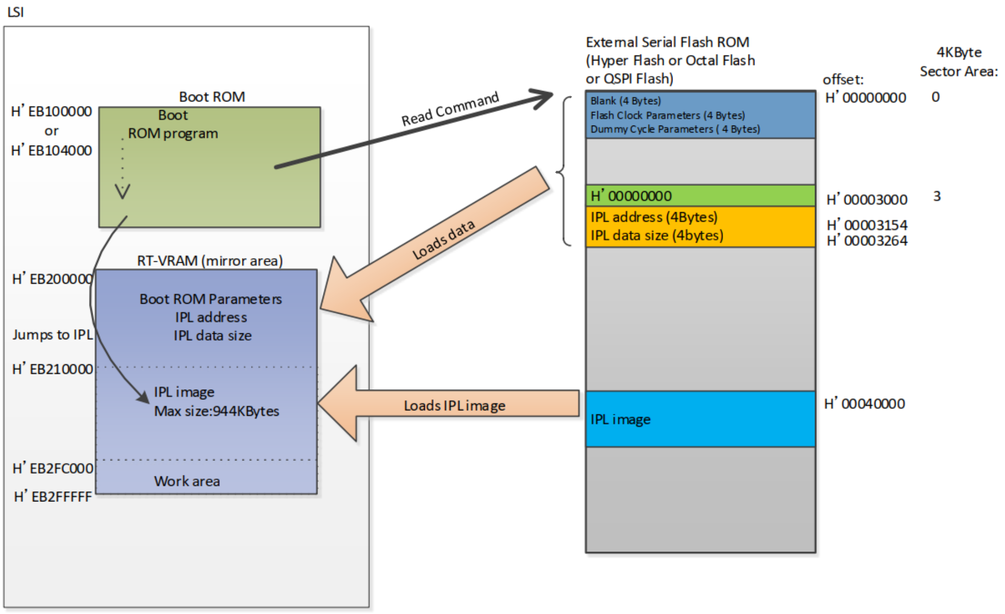
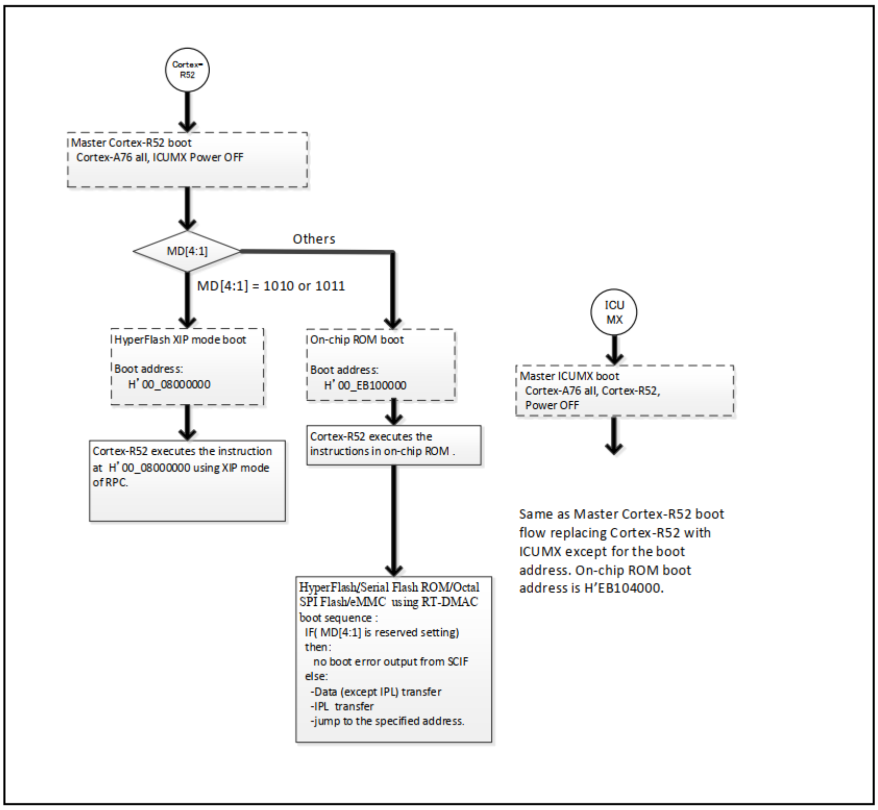


Figure 31.5 IPL transfer and Program image

이 boot Rom 의 프로그램은 MD 핀에 따라 자동으로 부팅이 결정된다.
기본적으로 1st IPL 을 Loading 하는 역할을 한다.



Cortex-R52
Master Cortex-R52 boot
Cortex-A76 all, ICUMX Power OFF
MD[4:1]
Others
MD[4:1] = 1010 or 1011
HyperFlash XIP mode boot
Boot address:
H' 00_08000000
On-chip ROM boot
Boot address:
H' 00_EB100000
Cortex-R52 executes the instruction at H' 00_08000000 using XIP mode of RPC.
Cortex-R52 executes the instructions in on-chip ROM .
HyperFlash/Serial Flash ROM/Octal SPI Flash/eMMC using RT-DMAC boot sequence :
IF(MD[4:1] is reserved setting)
then:
no boot error output from SCIF
else:
-Data (except IPL) transfer
-IPL transfer
-jump to the specified address.
ICU MX
Master ICUMX boot
Cortex-A76 all, Cortex-R52, Power OFF
Same as Master Cortex-R52 boot flow replacing Cortex-R52 with ICUMX except for the boot address. On-chip ROM boot address is H'EB104000.

Figure 31.3 Cortex-R52, ICUMX boot operation overview

10.2 1st IPL (ICUMX IPL)

ICUMX IPL 은 BootROM 프로그램에 의해 로드되고 실행됩니다. ICUMX IPL 은 외부 플래시 메모리에서 바이너리 이미지(G4MH 프로그램, ICUMH 프로그램, RTOS, CX 2nd IPL, CA 프로그램 및 Secure FW)를 로드하고 인증서를 사용하여 로드된 이미지의 무결성 검사를 실행합니다. ICUMX IPL 이 바이너리 이미지의 무결성 검사에 성공하면 각 CPU 에 대한 바이너리 이미지를 부팅합니다.

ICUMX IPL 의 구조는 아래와 같다.

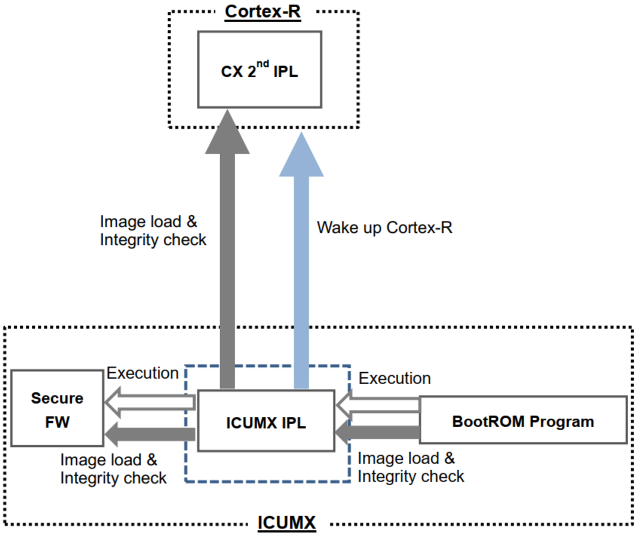


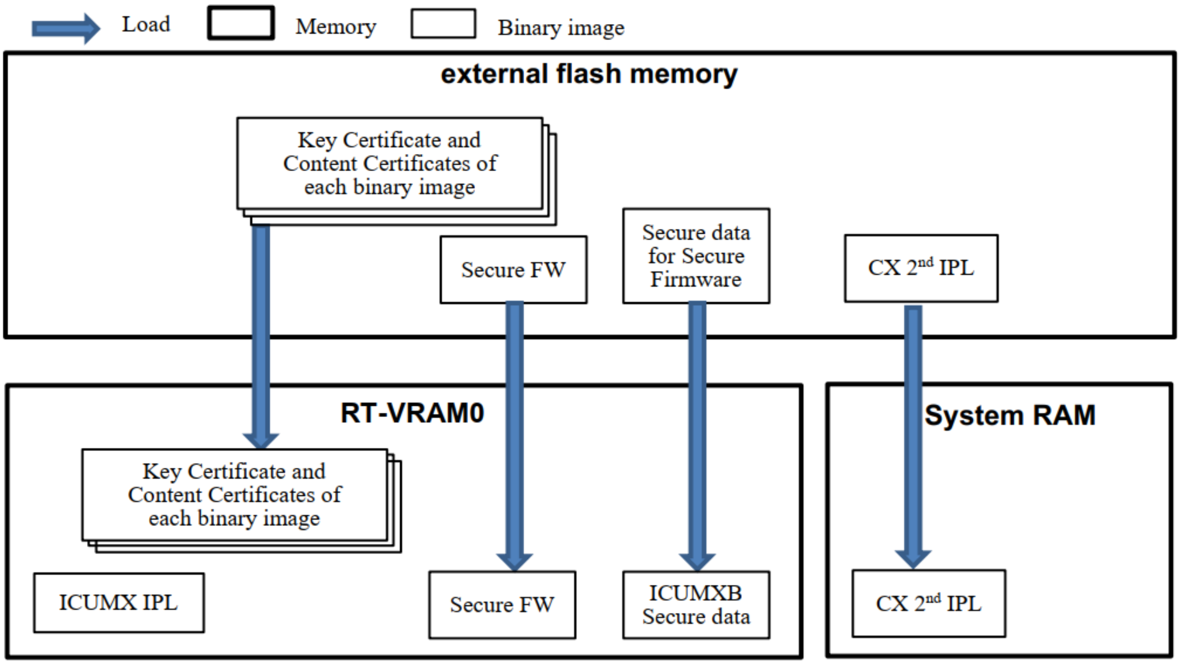
Figure 1-2 Scope of the Internal Program of ICUMX IPL (for V4H/V4M)

ICUMX IPL loads a binary image to RAM (RT-SRAM, RT-VRAM *1, System RAM, SDRAM or Code SRAM) from external flash memory.

Binary images loaded by ICUMX IPL are shown below.

- Key certificate
- Cert header
- G4MH program image *2
- ICUMH program image
- RTOS
- CX 2nd IPL *3
- CA Program #[X] *4
- Secure FW
- Secure data *5
- Certificates of each binary images. *6

R-Car V4H R-Car V4M



Load
Memory
Binary image
external flash memory
Key Certificate and Content Certificates of each binary image
Secure FW
Secure data for Secure Firmware
CX 2nd IPL
RT-VRAM0
Key Certificate and Content Certificates of each binary image
ICUMX IPL
Secure FW
ICUMXB Secure data
System RAM
CX 2nd IPL

Figure 1-6 Data flow of a binary image (for V4H/V4M)

기본적인 시퀀스는 아래와 같다. Boot Rom 부팅이 완료되고 ICUMX 의 프로그램을 RT-VRAM0 에 로딩하면 아래와 같은 시퀀스가 시작된다.

R-Car V4H R-Car V4M

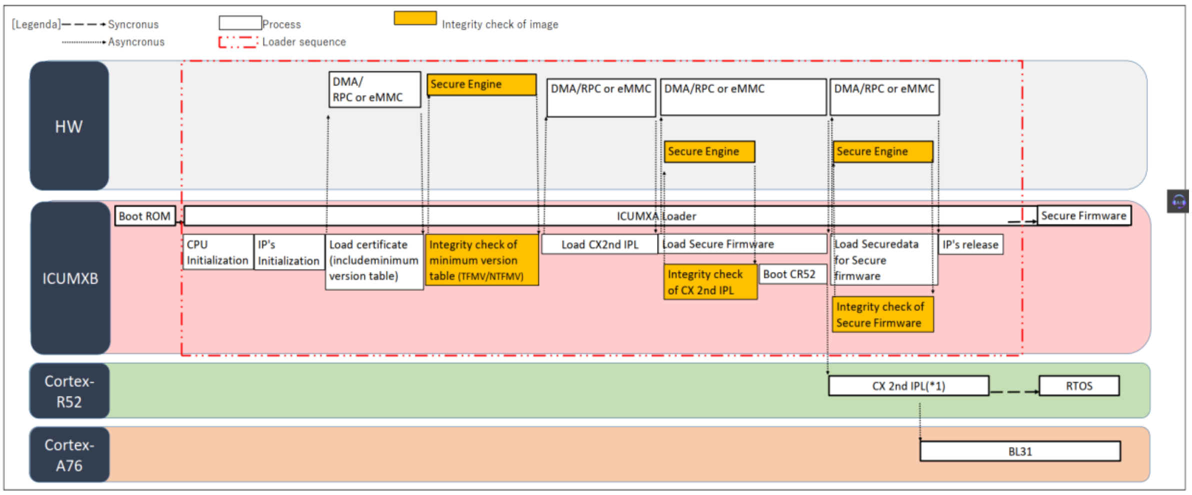


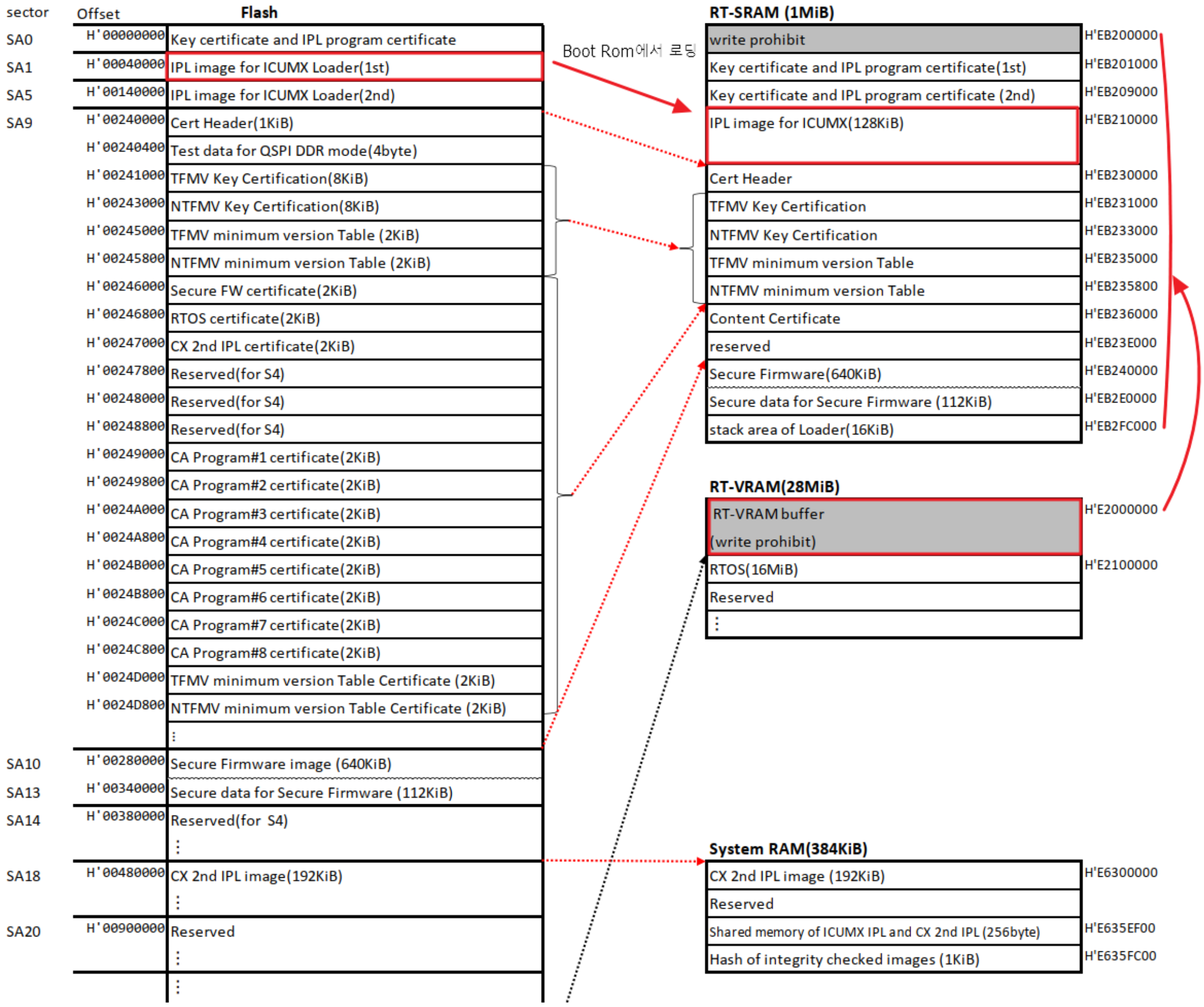
Figure 3-12 Sequence of ICUMX IPL for V4H / V4M (load target is CX 2nd IPL)

기본적인 Memory Area 는 아래와 같다.
Boot Rom 에서 로딩이 완료된 IPL Image 가 실행되는 과정이다.

모든 초기화 루틴 및 로딩이 완료되면 CR 코어를 깨우고 Secure Firmware 를 수행한다.

- After transferring the binary image of RTOS to RT-VRAM, ICUMX IPL boots up the Cortex-R and executes RTOS on Cortex-R.
- After transferring the binary image 'Secure FW' 'Secure data for Secure Firmware' and all ICUMX IPL processing is finished, ICUMX IPL executes Secure FW.
- ICUMX IPL jumps the program counter to Secure FW entry point. The entry point is the top address of the binary image of Secure FW.

Memory map when after loading



sector	Offset	Flash
SA0	H'00000000	Key certificate and IPL program certificate
SA1	H'00040000	IPL image for ICUMX Loader(1st)
SA5	H'00140000	IPL image for ICUMX Loader(2nd)
SA9	H'00240000	Cert Header(1KiB)
	H'00240400	Test data for QSPI DDR mode(4byte)
	H'00241000	TFMV Key Certification(8KiB)
	H'00243000	NTFMV Key Certification(8KiB)
	H'00245000	TFMV minimum version Table (2KiB)
	H'00245800	NTFMV minimum version Table (2KiB)
	H'00246000	Secure FW certificate(2KiB)
	H'00246800	RTOS certificate(2KiB)
	H'00247000	CX 2nd IPL certificate(2KiB)
	H'00247800	Reserved(for S4)
	H'00248000	Reserved(for S4)
	H'00248800	Reserved(for S4)
	H'00249000	CA Program#1 certificate(2KiB)
	H'00249800	CA Program#2 certificate(2KiB)
	H'0024A000	CA Program#3 certificate(2KiB)
	H'0024A800	CA Program#4 certificate(2KiB)
	H'0024B000	CA Program#5 certificate(2KiB)
	H'0024B800	CA Program#6 certificate(2KiB)
	H'0024C000	CA Program#7 certificate(2KiB)
	H'0024C800	CA Program#8 certificate(2KiB)
	H'0024D000	TFMV minimum version Table Certificate (2KiB)
	H'0024D800	NTFMV minimum version Table Certificate (2KiB)
		⋮
SA10	H'00280000	Secure Firmware image (640KiB)
SA13	H'00340000	Secure data for Secure Firmware (112KiB)
SA14	H'00380000	Reserved(for S4) ⋮
SA18	H'00480000	CX 2nd IPL image(192KiB) ⋮
SA20	H'00900000	Reserved ⋮
		⋮

RT-SRAM (1MiB)	
write prohibit	H'EB200000
Key certificate and IPL program certificate(1st)	H'EB201000
Key certificate and IPL program certificate (2nd)	H'EB209000
IPL image for ICUMX(128KiB)	H'EB210000
Cert Header	H'EB230000
TFMV Key Certification	H'EB231000
NTFMV Key Certification	H'EB233000
TFMV minimum version Table	H'EB235000
NTFMV minimum version Table	H'EB235800
Content Certificate	H'EB236000
reserved	H'EB23E000
Secure Firmware(640KiB)	H'EB240000
Secure data for Secure Firmware (112KiB)	H'EB2E0000
stack area of Loader(16KiB)	H'EB2FC000

RT-VRAM(28MiB)	
RT-VRAM buffer (write prohibit)	H'E2000000
RTOS(16MiB)	H'E2100000
Reserved	
⋮	

System RAM(384KiB)	
CX 2nd IPL image (192KiB)	H'E6300000
Reserved	
Shared memory of ICUMX IPL and CX 2nd IPL (256byte)	H'E635EF00
Hash of integrity checked images (1KiB)	H'E635FC00

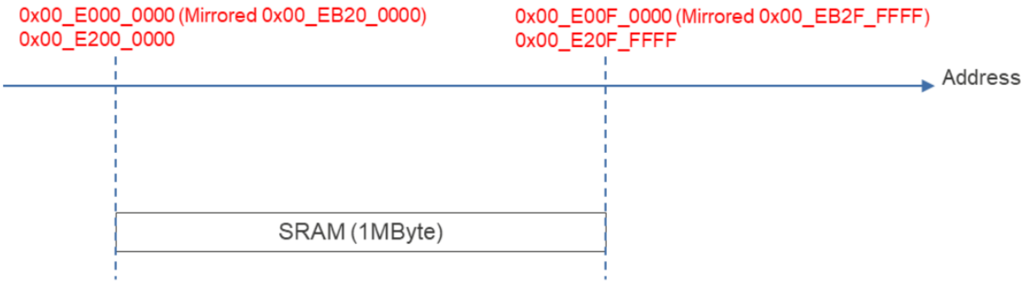
RT-VRAM 은 내부에 있는 SRAM 으로서 아래와 같은 Address 를 갖는다.

Address of RT-VRAMs are below:

RT-VRAM0: 0xE000_0000 to 0xE1BF_FFFF (up to 28MB)

RT-VRAM1: 0xE200_0000 to 0xE3BF_FFFF (up to 28MB)

RT-VRAM0(Only) 은 0xEB20_0000 주소로 Mirrored 되어 있다.
Only in RT-VRAM0, 1MB of SRAM area (0xE000_0000 to 0xE00F_FFFF) are mirrored other area (0xEB20_0000 to 0xEB2F_FFFF).



0x00_E000_0000 (Mirrored 0x00_EB20_0000)
0x00_E200_0000
0x00_E00F_0000 (Mirrored 0x00_EB2F_FFFF)
0x00_E20F_FFFF
Address
SRAM (1MByte)

Figure 36.2 Address area of RT-VRAM in Compatible mode

10.3 2nd IPL (CR52 Core1)

CX 2nd IPL 은 eMMC(이하 "외부 플래시 메모리"라고 함)에서 바이너리 이미지 로딩 기능을 제공합니다.

Table 1-1 Function list of the CX 2nd IPL

Functions	Explanation
Hardware initialization	Initialize the configuration and the hardware used by the system.
Display the booting message	Output the message (software version, etc.) to SCIF or HSCIF.
Loading the image	(S4) Load binary images (Secure FW, Trusted OS and Non-Secure OS).
	(V4H/V4M) Load binary images (RTOS, Secure FW and Non-Secure OS).
Starting Secure Monitor	Execute a Secure Monitor binary image after CX 2nd IPL processing was finish.

2nd IPL 은 아래와 같은 기본적인 구조를 갖는다.

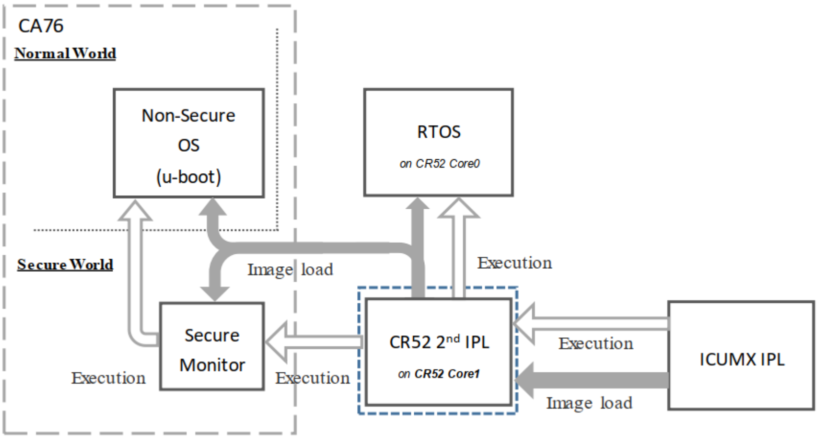


Figure 1-2 Scope of the Internal Program of CX 2nd IPL (V4H/V4M)

10.3.1 Memory Structure

2nd IPL 를 이해하기위해서는 RAM 의 Mirror Area 에 대한 이해가 필요하다.
RT-VRAM 은 내부에 있는 SRAM 으로서 아래와 같은 Address 를 갖는다.

Address of RT-VRAMs are below:

RT-VRAM0: 0xE000_0000 to 0xE1BF_FFFF (up to 28MB)

RT-VRAM1: 0xE200_0000 to 0xE3BF_FFFF (up to 28MB)

RT-VRAM0(Only) 은 0xEB20_0000 주소로 Mirrored 되어 있다.
Only in RT-VRAM0, 1MB of SRAM area (0xE000_0000 to 0xE00F_FFFF) are mirrored other area (0xEB20_0000 to 0xEB2F_FFFF).

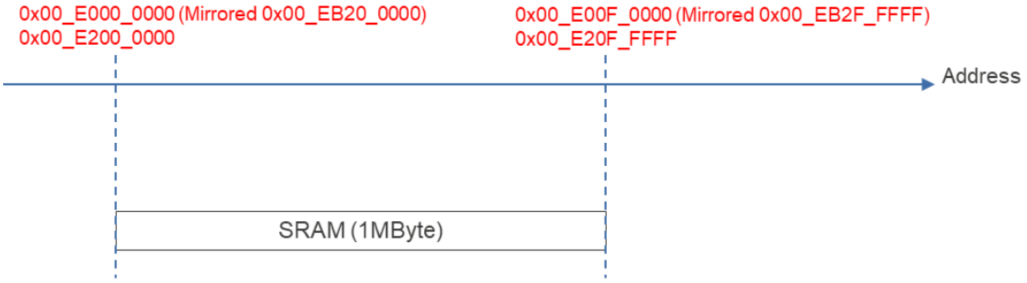

0x00_E000_0000 (Mirrored 0x00_EB20_0000)
0x00_E200_0000
0x00_E00F_0000 (Mirrored 0x00_EB2F_FFFF)
0x00_E20F_FFFF
Address
SRAM (1MByte)

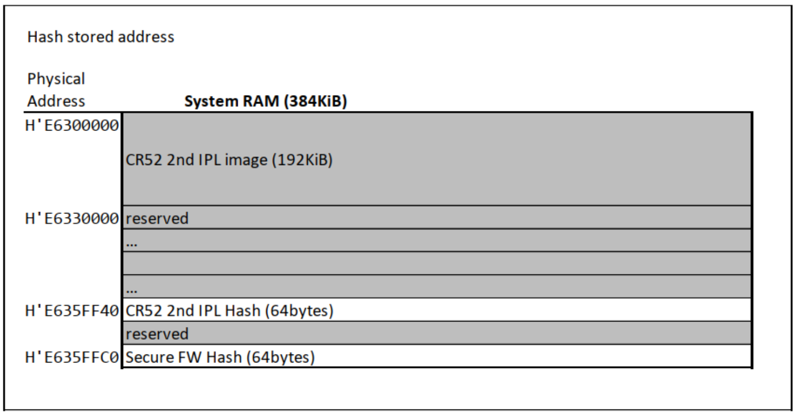
Figure 36.2 Address area of RT-VRAM in Compatible mode

2nd IPL 은 System RAM 에 위치한다.

Table 3-3 Hardware resource (V4H/V4M)

Component	Channel	Purpose	Initialize	Release / Finalize
RT-VRAM1	-	Executing the RTOS.	-	-
SCIF	ch0	Displaying starting message and log message.	-	-
HSCIF	ch0	Displaying starting message and log message.	-	-
SDHI	-	Reading the eMMC.	Yes	Yes
System RAM	-	CX 2nd IPL runs on System RAM. (Top address: H'E6300000, Size: 384KiB)	-	-
LPDDR5-SDRAM	-	Loading programs from eMMC.	-	-

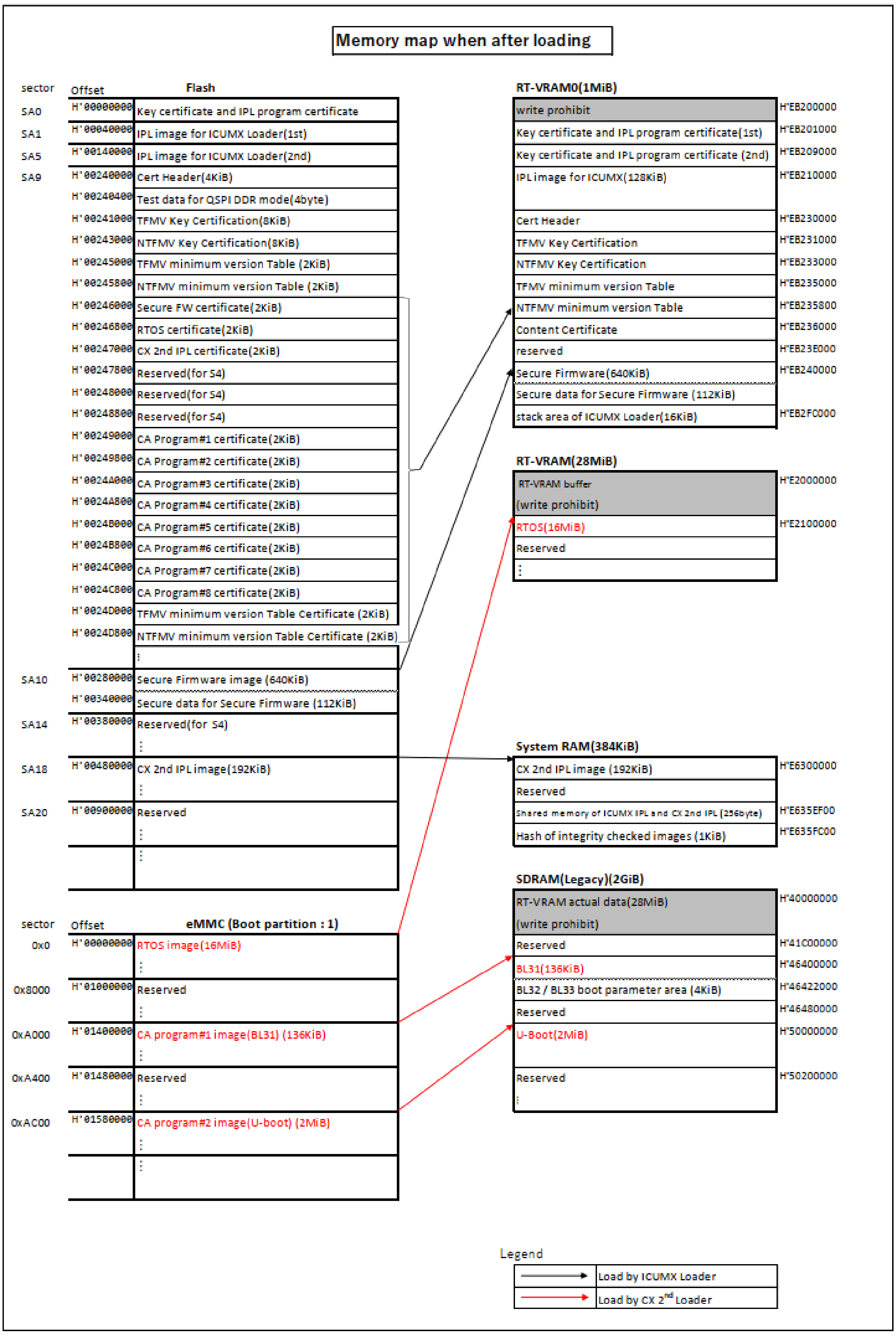
R-Car V4H R-Car V4M



Hash stored address

Physical Address	System RAM (384KiB)
H'E6300000	CR52 2nd IPL image (192KiB)
H'E6330000	reserved
	...
	...
H'E635FF40	CR52 2nd IPL Hash (64bytes)
	reserved
H'E635FFC0	Secure FW Hash (64bytes)

Figure 3-9 Hash storage memory map for V4H/V4M



Memory map when after loading

sector	Offset	Flash
SA0	H'00000000	Key certificate and IPL program certificate
SA1	H'00040000	IPL image for ICUMX Loader(1st)
SA5	H'00140000	IPL image for ICUMX Loader(2nd)
SA9	H'00240000	Cert Header(4KiB)
	H'00240400	Test data for QSPI DDR mode(4byte)
	H'00241000	TFMV Key Certification(8KiB)
	H'00243000	NTFMV Key Certification(8KiB)
	H'00245000	TFMV minimum version Table (2KiB)
	H'00245800	NTFMV minimum version Table (2KiB)
	H'00246000	Secure FW certificate(2KiB)
	H'00246800	RTOS certificate(2KiB)
	H'00247000	CX 2nd IPL certificate(2KiB)
	H'00247800	Reserved(for S4)
	H'00248000	Reserved(for S4)
	H'00248800	Reserved(for S4)
	H'00249000	CA Program#1 certificate(2KiB)
	H'00249800	CA Program#2 certificate(2KiB)
	H'0024A000	CA Program#3 certificate(2KiB)
	H'0024A800	CA Program#4 certificate(2KiB)
	H'0024B000	CA Program#5 certificate(2KiB)
	H'0024B800	CA Program#6 certificate(2KiB)
	H'0024C000	CA Program#7 certificate(2KiB)
	H'0024C800	CA Program#8 certificate(2KiB)
	H'0024D000	TFMV minimum version Table Certificate (2KiB)
	H'0024D800	NTFMV minimum version Table Certificate (2KiB)
		⋮
SA10	H'00280000	Secure Firmware image (640KiB)
	H'00340000	Secure data for Secure Firmware (112KiB)
SA14	H'00380000	Reserved(for S4) ⋮
SA18	H'00480000	CX 2nd IPL image(192KiB) ⋮
SA20	H'00900000	Reserved ⋮
		⋮

sector	Offset	eMMC (Boot partition : 1)
0x0	H'00000000	RTOS image(16MiB) ⋮
0x8000	H'01000000	Reserved ⋮
0xA000	H'01400000	CA program#1 image(BL31) (136KiB) ⋮
0xA400	H'01480000	Reserved ⋮
0xAC00	H'01580000	CA program#2 image(U-boot) (2MiB) ⋮
		⋮

RT-VRAM0(1MiB)	
write prohibit	H'EB200000
Key certificate and IPL program certificate(1st)	H'EB201000
Key certificate and IPL program certificate (2nd)	H'EB209000
IPL image for ICUMX(128KiB)	H'EB210000
Cert Header	H'EB230000
TFMV Key Certification	H'EB231000
NTFMV Key Certification	H'EB233000
TFMV minimum version Table	H'EB235000
NTFMV minimum version Table	H'EB235800
Content Certificate	H'EB236000
reserved	H'EB23E000
Secure Firmware(640KiB)	H'EB240000
Secure data for Secure Firmware (112KiB)	
stack area of ICUMX Loader(16KiB)	H'EB2FC000

RT-VRAM(28MiB)	
RT-VRAM buffer (write prohibit)	H'E2000000
RTOS(16MiB)	H'E2100000
Reserved	
⋮	

System RAM(384KiB)	
CX 2nd IPL image (192KiB)	H'E6300000
Reserved	
Shared memory of ICUMX IPL and CX 2nd IPL (256byte)	H'E635EF00
Hash of integrity checked images (1KiB)	H'E635FC00

SDRAM(Legacy)(2GiB)	
RT-VRAM actual data(28MiB) (write prohibit)	H'40000000
Reserved	H'41C00000
BL31(136KiB)	H'46400000
BL32 / BL33 boot parameter area (4KiB)	H'46422000
Reserved	H'46480000
U-Boot(2MiB)	H'50000000
Reserved ⋮	H'50200000

Legend

⟶	Load by ICUMX Loader
⟶	Load by CX 2^{nd} Loader

Figure 4-2 Memory layout (QSPI Flash/HyperFlash and eMMC) of V4H/V4M

10.3.2 Operation sequence

Figure 3-11 shows sequence of ICUMX IPL in case of load target is CX 2nd IPL of V4H/V4M.

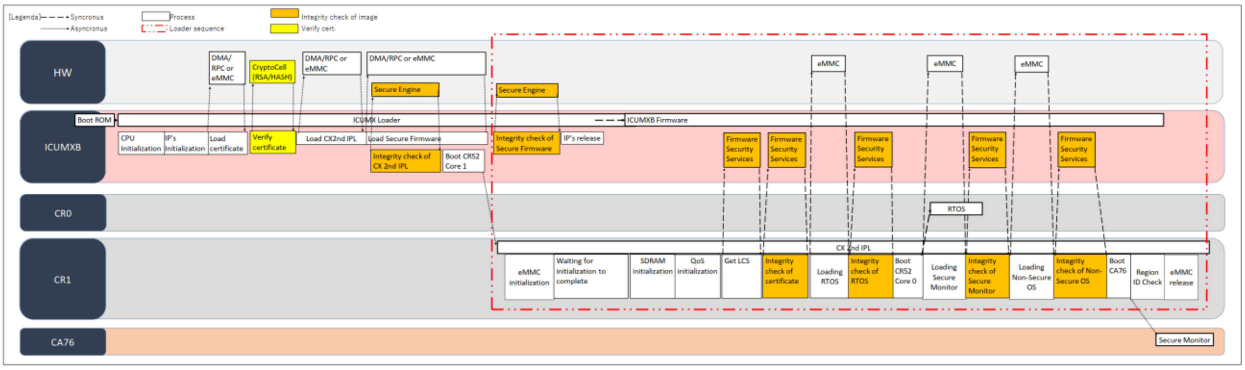


Figure 3-11 Sequence of ICUMX IPL and CX 2nd IPL of V4H/V4M

10.3.3 Download image address mapping

Table 4-3 Information list of release image when ICUMX IPL loads CX 2nd IPL of V4H/V4M (QSPI Flash / HyperFlash)

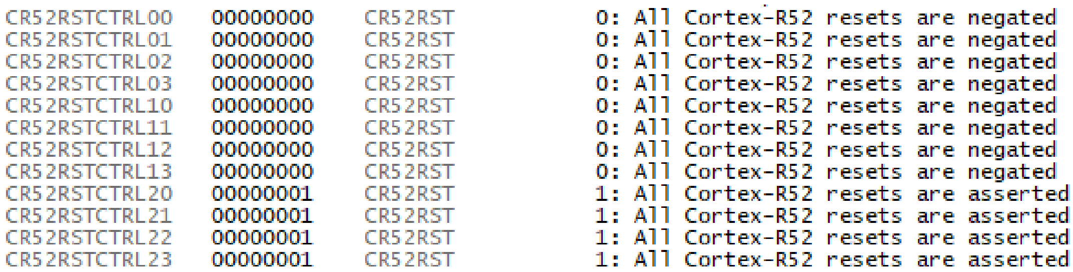
Filename	Program Top Address	Flash Save Address	Description
bootparam_sa0.srec	H’EB200000 *1	H’000000	ICUMX IPL (Boot parameter)
icumx_loader.srec	H’EB210000	H’040000	ICUMX IPL
cert_header_sa9.srec	H’EB230000	H’240000	ICUMX IPL (Certificate)
dummy_fw.srec	H’EB240000	H’280000	Dummy firmware (Instead of secure firmware)
cr52_loader.srec	H’E6300000	H’480000	CX 2nd IPL (CR)

Filename	Program Top Address*2	eMMC Save Partition	eMMC Save Sectors *1	Description
dummy_rtos.srec	H’E2100000	boot partition1	H’0000	Dummy RTOS (Instead of CR main OS)
AArch64_Dummy_CA76_Program.srec	H’46400000*3	boot partition1	H’A000	Dummy Secure monitor (Instead of Secure monitor or BL31)
AArch64_Dummy_CA76_Program2.srec	H’50000000*3	boot partition1	H’AC00	Dummy U-Boot (Instead of U-Boot)

Note *1) 1 sector is 512 bytes.
Note *2) Program Top Address must be 512 bytes boundary.
Note *3) In case of build option ‘ACC_PROT_ENABLE=1’, Program Top Address is offset H’20_00000000.

10.3.4 CR Core Booting

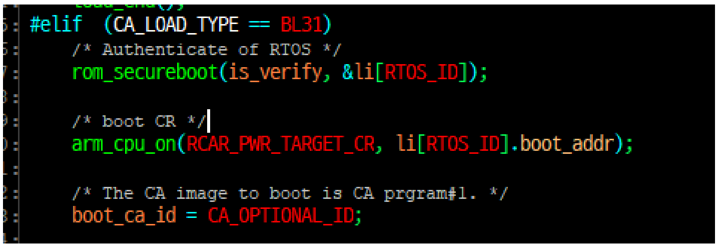
기본적으로 SDK 에서는 아래와 같이 CR0, CR1 이 부팅되고 있다.



```
CR52RSTCTRL00   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL01   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL02   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL03   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL10   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL11   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL12   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL13   00000000   CR52RST        0: All Cortex-R52 resets are negated
CR52RSTCTRL20   00000001   CR52RST        1: All Cortex-R52 resets are asserted
CR52RSTCTRL21   00000001   CR52RST        1: All Cortex-R52 resets are asserted
CR52RSTCTRL22   00000001   CR52RST        1: All Cortex-R52 resets are asserted
CR52RSTCTRL23   00000001   CR52RST        1: All Cortex-R52 resets are asserted
```

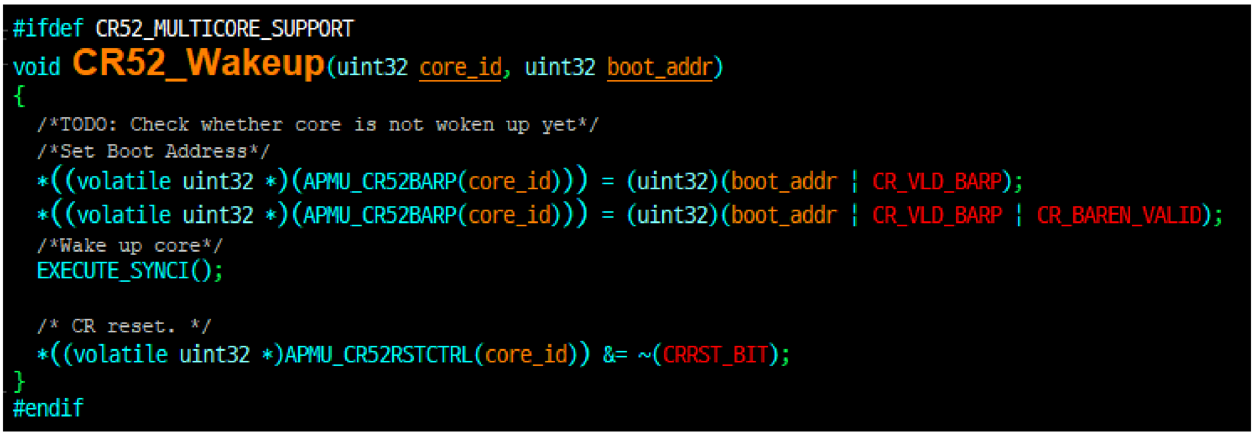
이는 IPL(ICUMX)에서 부팅 여부를 결정할 수 있습니다.

uint32_t loader_main(void) 함수 내부에 아래와 같이 arm_cpu_on 함수를 통해 Reset Register 를 컨트롤 한다.



```
#elif  (CA_LOAD_TYPE == BL31)
    /* Authenticate of RTOS */
    rom_secureboot(is_verify, &li[RTOS_ID]);

    /* boot CR */
    arm_cpu_on(RCAR_PWR_TARGET_CR, li[RTOS_ID].boot_addr);

    /* The CA image to boot is CA prgram#1. */
    boot_ca_id = CA_OPTIONAL_ID;
```

MCAL 패키지에도 아래와 같이 C 코드로 Wakeup 코드를 제공한다.



```
#ifdef CR52_MULTICORE_SUPPORT
void CR52_Wakeup(uint32 core_id, uint32 boot_addr)
{
  /*TODO: Check whether core is not woken up yet*/
  /*Set Boot Address*/
  *((volatile uint32 *)(APMU_CR52BARP(core_id))) = (uint32)(boot_addr ¦ CR_VLD_BARP);
  *((volatile uint32 *)(APMU_CR52BARP(core_id))) = (uint32)(boot_addr ¦ CR_VLD_BARP ¦ CR_BAREN_VALID);
  /*Wake up core*/
  EXECUTE_SYNCI();

  /* CR reset. */
  *((volatile uint32 *)APMU_CR52RSTCTRL(core_id)) &= ~(CRRST_BIT);
}
#endif
```

해당 레지스터는 아래와 같이 APMU 에서 제어 한다.

10. Advanced Power Management Unit

10.1 Overview

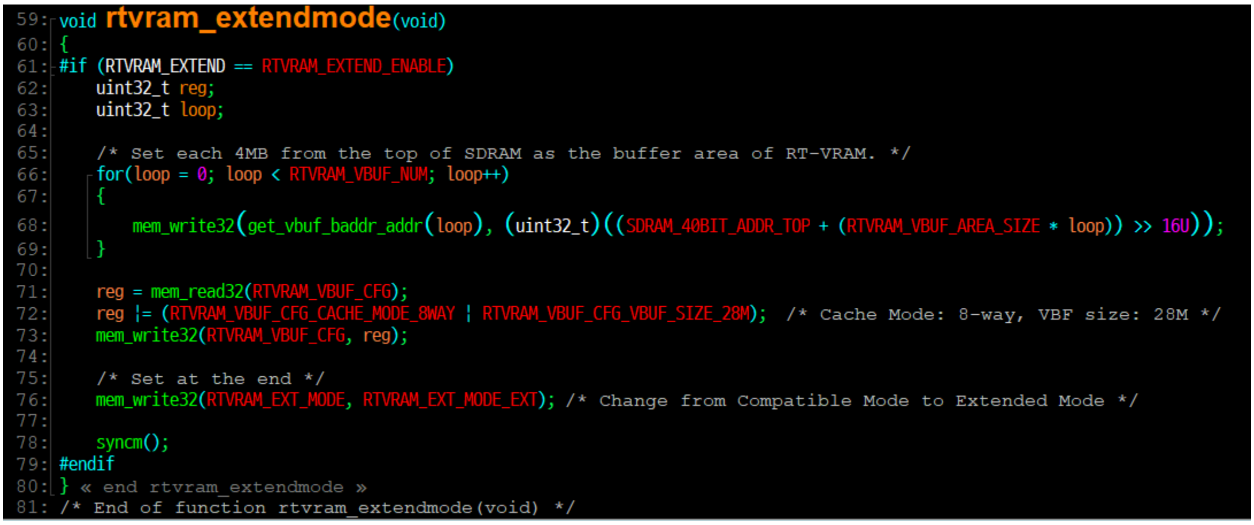
The Advanced Power Management Unit (APMU) is a module to control power supply and clock supply to the AP-System Core. It also provides the reset vector base address for AP-System Core and Realtime Core.

10.4 RT-VRAM Configuration

RT-VRAM 은 compatible Mode 와 Extended Mode 를 제공한다.

Virtual buffer configuration Register for RTVRAM[k](VBUF_CFG_[k])	RT_VRAM[k]	H'6504	8	8	CACHE_MODE	B'0	R/W	Specifies the number of way, 0: 4-way 1: 8-way This parameter shall be set in conpatible mode. Changing this parameter in extended mode is prohibited.
Virtual buffer configuration Register for RTVRAM[k](VBUF_CFG_[k])	RT_VRAM[k]	H'6504	7	3	-	All 0	R	Reserved These bits are always read as 0. The write value should always be 0.
Virtual buffer configuration Register for RTVRAM[k](VBUF_CFG_[k])	RT_VRAM[k]	H'6504	2	0	VBUF_SIZE[2:0]	0	R/W	Specifies size of the virtual buffer, 0: 4M 1: 8M 2: 12M 3: 16M 4: 20M 5: 24M 6: 28M 7: setting prohibited This parameter shall be set in conpatible mode. Changing this parameter in extended mode is prohibited.

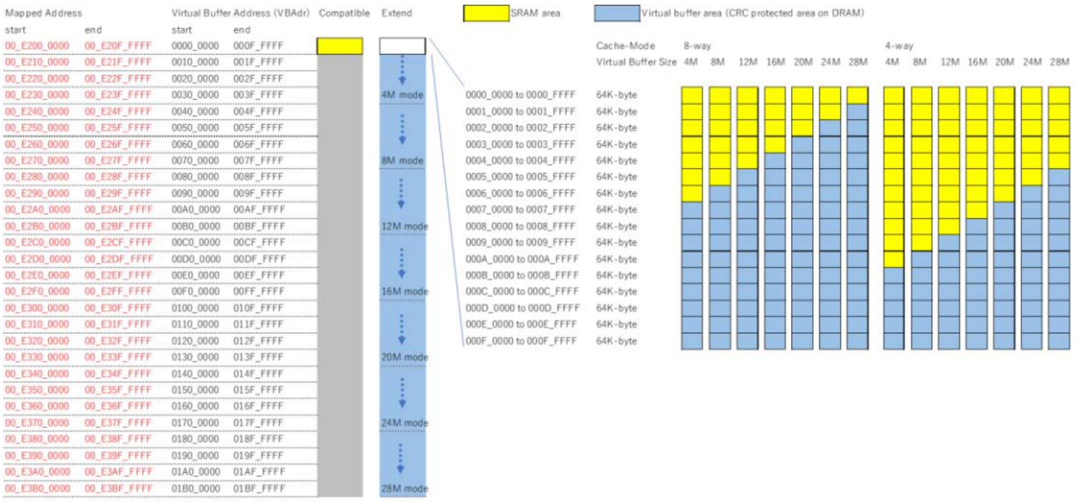
기본적으로 RT-VRAM1 은 Extended Mode 로 설정되어 있다.



```
void rtvram_extendmode(void)
{
#if (RTVRAM_EXTEND == RTVRAM_EXTEND_ENABLE)
    uint32_t reg;
    uint32_t loop;

    /* Set each 4MB from the top of SDRAM as the buffer area of RT-VRAM. */
    for(loop = 0; loop < RTVRAM_VBUF_NUM; loop++)
    {
        mem_write32(get_vbuf_baddr_addr(loop), (uint32_t)((SDRAM_40BIT_ADDR_TOP + (RTVRAM_VBUF_AREA_SIZE * loop)) >> 16U));
    }

    reg = mem_read32(RTVRAM_VBUF_CFG);
    reg |= (RTVRAM_VBUF_CFG_CACHE_MODE_8WAY | RTVRAM_VBUF_CFG_VBUF_SIZE_28M);  /* Cache Mode: 8-way, VBF size: 28M */
    mem_write32(RTVRAM_VBUF_CFG, reg);

    /* Set at the end */
    mem_write32(RTVRAM_EXT_MODE, RTVRAM_EXT_MODE_EXT); /* Change from Compatible Mode to Extended Mode */

    syncm();
#endif
} « end rtvram_extendmode »
/* End of function rtvram_extendmode(void) */
```

Base Address Name	Base Address
RT_VRAM0	H'FFE90000
RT_VRAM1	H'FFEC0000

RT-VRAM 의 구조는 아래와 같다.



Mapped Address start	end	Virtual Buffer Address (VBAdr) start	end	Compatible	Extend
00_E200_0000	00_E20F_FFFF	0000_0000	000F_FFFF	🟨	
00_E210_0000	00_E21F_FFFF	0010_0000	001F_FFFF		
00_E220_0000	00_E22F_FFFF	0020_0000	002F_FFFF		
00_E230_0000	00_E23F_FFFF	0030_0000	003F_FFFF		4M mode
00_E240_0000	00_E24F_FFFF	0040_0000	004F_FFFF		
00_E250_0000	00_E25F_FFFF	0050_0000	005F_FFFF		
00_E260_0000	00_E26F_FFFF	0060_0000	006F_FFFF		
00_E270_0000	00_E27F_FFFF	0070_0000	007F_FFFF		8M mode
00_E280_0000	00_E28F_FFFF	0080_0000	008F_FFFF		
00_E290_0000	00_E29F_FFFF	0090_0000	009F_FFFF		
00_E2A0_0000	00_E2AF_FFFF	00A0_0000	00AF_FFFF		
00_E2B0_0000	00_E2BF_FFFF	00B0_0000	00BF_FFFF		12M mode
00_E2C0_0000	00_E2CF_FFFF	00C0_0000	00CF_FFFF		
00_E2D0_0000	00_E2DF_FFFF	00D0_0000	00DF_FFFF		
00_E2E0_0000	00_E2EF_FFFF	00E0_0000	00EF_FFFF		
00_E2F0_0000	00_E2FF_FFFF	00F0_0000	00FF_FFFF		16M mode
00_E300_0000	00_E30F_FFFF	0100_0000	010F_FFFF		
00_E310_0000	00_E31F_FFFF	0110_0000	011F_FFFF		
00_E320_0000	00_E32F_FFFF	0120_0000	012F_FFFF		
00_E330_0000	00_E33F_FFFF	0130_0000	013F_FFFF		20M mode
00_E340_0000	00_E34F_FFFF	0140_0000	014F_FFFF		
00_E350_0000	00_E35F_FFFF	0150_0000	015F_FFFF		
00_E360_0000	00_E36F_FFFF	0160_0000	016F_FFFF		
00_E370_0000	00_E37F_FFFF	0170_0000	017F_FFFF		24M mode
00_E380_0000	00_E38F_FFFF	0180_0000	018F_FFFF		
00_E390_0000	00_E39F_FFFF	0190_0000	019F_FFFF		
00_E3A0_0000	00_E3AF_FFFF	01A0_0000	01AF_FFFF		
00_E3B0_0000	00_E3BF_FFFF	01B0_0000	01BF_FFFF		28M mode

🟨 SRAM area
🟦 Virtual buffer area (CRC protected area on DRAM)

	Cache-Mode	8-way							4-way						
	Virtual Buffer Size	4M	8M	12M	16M	20M	24M	28M	4M	8M	12M	16M	20M	24M	28M
0000_0000 to 0000_FFFF	64K-byte	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨
0001_0000 to 0001_FFFF	64K-byte	🟨	🟨	🟨	🟨	🟨	🟨	🟦	🟨	🟨	🟨	🟨	🟨	🟨	🟨
0002_0000 to 0002_FFFF	64K-byte	🟨	🟨	🟨	🟨	🟨	🟦	🟦	🟨	🟨	🟨	🟨	🟨	🟨	🟨
0003_0000 to 0003_FFFF	64K-byte	🟨	🟨	🟨	🟨	🟦	🟦	🟦	🟨	🟨	🟨	🟨	🟨	🟨	🟨
0004_0000 to 0004_FFFF	64K-byte	🟨	🟨	🟨	🟦	🟦	🟦	🟦	🟨	🟨	🟨	🟨	🟨	🟨	🟨
0005_0000 to 0005_FFFF	64K-byte	🟨	🟨	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟨	🟨	🟨	🟨	🟦
0006_0000 to 0006_FFFF	64K-byte	🟨	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟨	🟨	🟨	🟦	🟦
0007_0000 to 0007_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟨	🟨	🟦	🟦	🟦
0008_0000 to 0008_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟨	🟦	🟦	🟦	🟦
0009_0000 to 0009_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟨	🟦	🟦	🟦	🟦	🟦
000A_0000 to 000A_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟨	🟦	🟦	🟦	🟦	🟦	🟦
000B_0000 to 000B_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
000C_0000 to 000C_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
000D_0000 to 000D_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
000E_0000 to 000E_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦
000F_0000 to 000F_FFFF	64K-byte	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦	🟦

Figure 36.4 The size of SRAM area and Virtual buffer area in the first 1MB area in Extended Mode

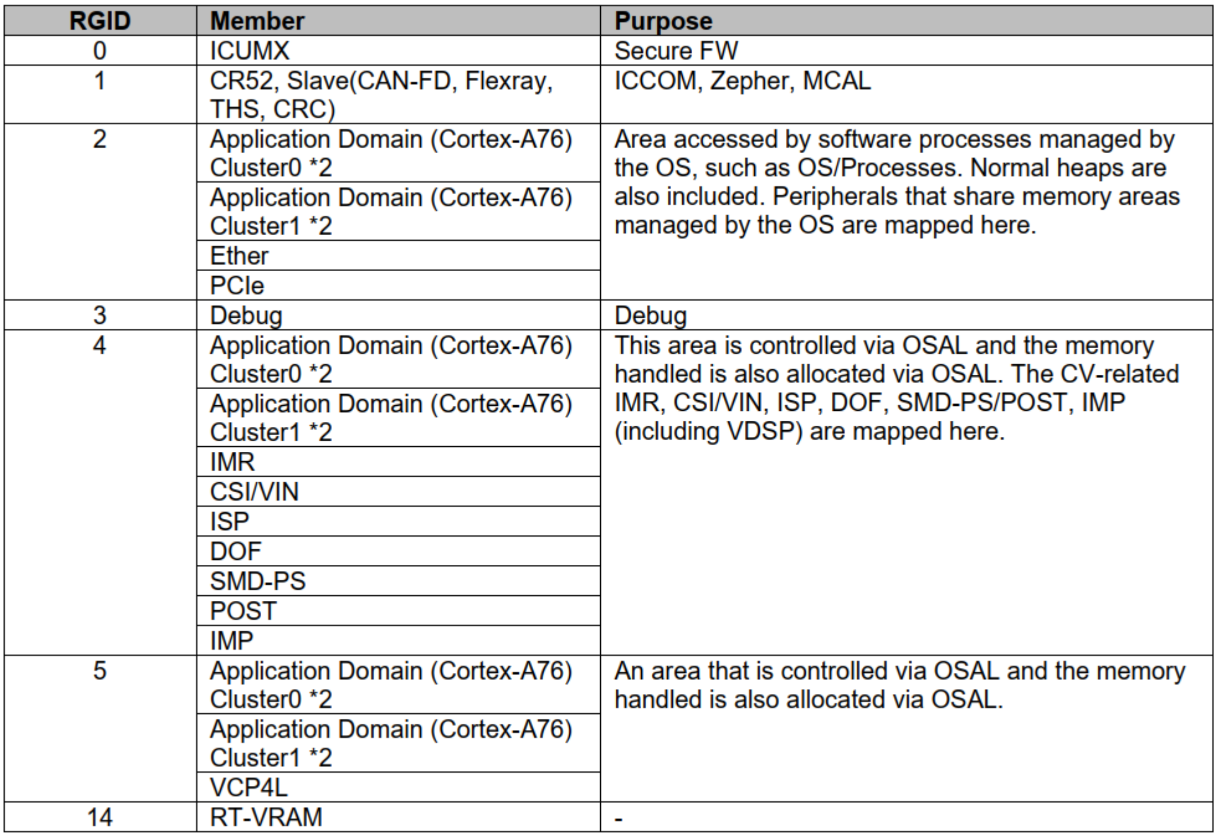
Extend RT-VRAM from 1MiB to 28MiB. Valid address of RT-VRAM is 0xE2100000 – 0xE3BFFFFF.
The top 1MiB of RT-VRAM (0xE2000000 – 0xE20FFFFF) become reserved area (cache area of RTVRAM).
Extended RT-VRAM area’s actual area is 0x40000000 – 0x41BFFFFF on SDRAM. So, this area on SDRAM become reserved area.

10.5 RGID

모든 메모리는 접근권한이 있는 Core 에 한해서만 접근을 허용한다.
V4H 의 경우 아래와 같은 Region ID 를 갖고 있다.

R-Car V4H

Table 1-9 An example RGID value for V4H Master I/F



RGID	Member	Purpose
0	ICUMX	Secure FW
1	CR52, Slave(CAN-FD, Flexray, THS, CRC)	ICCOM, Zepher, MCAL
2	Application Domain (Cortex-A76) Cluster0 *2	Area accessed by software processes managed by the OS, such as OS/Processes. Normal heaps are also included. Peripherals that share memory areas managed by the OS are mapped here.
	Application Domain (Cortex-A76) Cluster1 *2	
	Ether	
	PCIe	
3	Debug	Debug
4	Application Domain (Cortex-A76) Cluster0 *2	This area is controlled via OSAL and the memory handled is also allocated via OSAL. The CV-related IMR, CSI/VIN, ISP, DOF, SMD-PS/POST, IMP (including VDSP) are mapped here.
	Application Domain (Cortex-A76) Cluster1 *2	
	IMR	
	CSI/VIN	
	ISP	
	DOF	
	SMD-PS	
	POST	
	IMP	
5	Application Domain (Cortex-A76) Cluster0 *2	An area that is controlled via OSAL and the memory handled is also allocated via OSAL.
	Application Domain (Cortex-A76) Cluster1 *2	
	VCP4L	
14	RT-VRAM	-

부팅 파트에서는 ICUMX, CR52 에 관련된 부분을 집중적으로 기술한다.

10.5.1 Memory Matrix for RGID

R-Car V4H

Table 1-12 Memory access protection setting for R-Car V4H

Memory	Top address (physical)	Bottom address (physical)	Attribute	Region ID													
				0		1		2		3		4		5		14*5	
				R	W	R	W	R	W	R	R	R	W	R	W	R	W
RT-SRAM (Area0) *1	0xE0000000	0xE003FFFF	N	✓	✓	✓	-	-	-	-	-	-	-	-	-	-	-
RT-SRAM (Area1) *1	0xE0040000	0xE00FFFFF	S	✓	✓	-	-	-	-	-	-	-	-	-	-	-	-
RT-VRAM1 (Area0)	0xE2000000	0xE200FFFF	N	✓	-	✓	✓	-	-	-	-	-	-	-	-	-	-
RT-VRAM1 (Area1)	0xE2010000	0xE20FFFFF	N	-	-	-	-	-	-	-	✓	✓	-	-	-	✓	✓
RT-VRAM1 (Area2)	0xE2100000	0xE3BFFFFF	N	✓	- *6	✓	✓	-	✓	-	-	-	-	-	-	-	-
System RAM(Area0)	0xE6300000	0xE635DFFF	N	✓	- *4	✓	✓	-	-	-	-	-	✓	✓	✓	-	-
System RAM(Area1)	0xE635E000	0xE635FFFF	N	✓	✓	✓	✓	-	-	-	-	-	✓	✓	✓	-	-
System RAM(Area2)	0xE6360000	0xE63FFFFF	N	✓	✓	-	-	-	-	-	-	-	✓	✓	✓	-	-
SDRAM (Area0) *2	0x0400000000	0x0401BFFFFF	N	-	-	-	-	-	-	-	✓	✓	-	-	-	✓	✓
SDRAM (Area1) *2	0x0401C00000	0x0401CFFFFF	N	✓	✓	✓	✓	-	-	-	-	-	-	-	-	-	-
SDRAM (Area2) *2	0x0401D00000	0x04063FFFFF	N	✓	✓	✓	✓	✓	✓	✓	-	-	✓	✓	✓	-	-
SDRAM (Area3) *2*3	0x0406400000	0x040643FFFF	S	- *6	- *6	-	-	✓	✓	-	-	-	-	-	-	-	-
SDRAM (Area4) *2	0x0406440000	0x0407FBFFFF	N	✓	✓	✓	✓	✓	✓	✓	-	-	✓	✓	✓	-	-
SDRAM (Area5) *2	0x0407FC0000	0x0407FFFFFF	N	✓	✓	✓	✓	-	-	-	-	-	-	-	-	-	-
SDRAM (Area6) *2	0x0408000000	0x041DBFFFFF	N	- *6	- *6	-	-	✓	✓	-	-	-	-	-	-	-	-
SDRAM (Area7) *2	0x041DC00000	0x041FFFFFFF	N	-	-	-	-	-	-	-	-	-	✓	-	-	-	-
SDRAM (Area8) *2	0x0420000000	0x043FFFFFFF	N	-	-	✓	✓	-	-	-	-	-	-	-	-	-	-
SDRAM (Area9) *2	0x0440000000	0x045FFFFFFF	N	-	-	-	-	-	-	-	-	-	-	✓	✓	-	-
SDRAM (Area10) *2	0x0460000000	0x047FFFFFFF	N	-	-	-	-	-	-	-	-	-	✓	-	-	-	-
SDRAM (Area11)	0x0480000000	0x04FFFFFFFF	N	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
SDRAM (Area12)	0x0500000000	0x05FFFFFFFF	N	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SDRAM (Area13)	0x0600000000	0x06FFFFFFFF	N	-	-	-	-	✓	✓	-	-	-	-	-	-	-	-

legend … 'S': Secure 'N': Non-Secure 'R': Read 'W': Write '✓': permit '-': prohibit

Note*1) RT-VRAM0 is mirroring to 0xEB200000. RT-SRAM is same as RT-VRAM0.
Note*2) SDRAM 0x0400000000-0x047FFFFFFF is mirroring to 0x40000000-0xBFFFFFFF.
Note*3) Attribute setting is done by CX 2nd IPL.
Note*4) Give RGID0 writing privilege only when loading CX 2nd IPL.(Refer to 1.4.1 Related Document[11] chapter 6.23.5)
Note*5) SDRAM(Area0)
Note*6) It is required privilege when secure boot on CX 2nd IPL. Give privilege on ICUMX IPL and remove privilege on CX 2nd IPL.

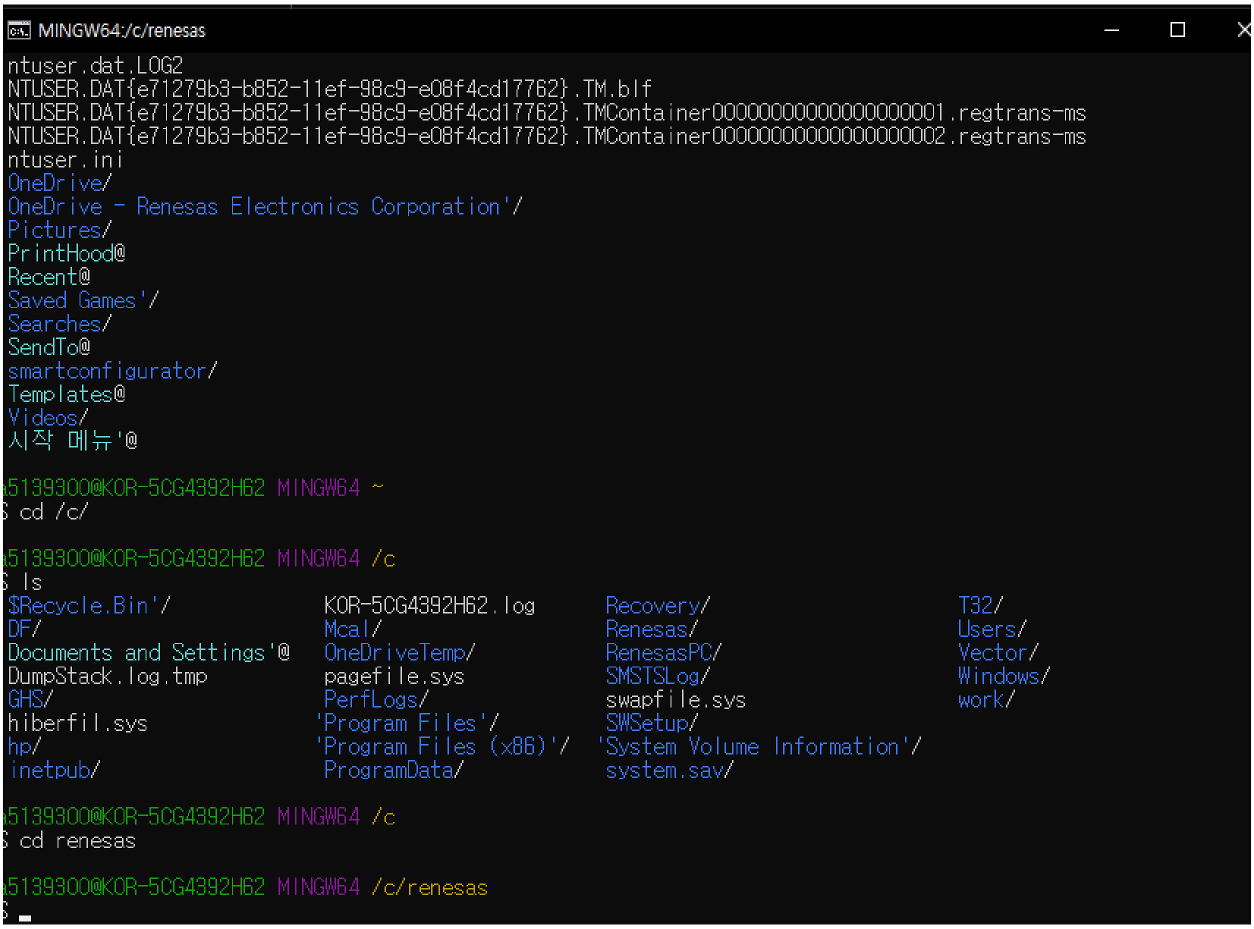
11 인터럽트 GIC(Global Interrupt Controller)

GICD_IGROUPR1-30 레지스터는 ARM 의 GIC (Generic Interrupt Controller)에서 사용되며, 주로 SPI (Shared Peripheral Interrupts)에 대한 그룹을 설정하는 역할을 합니다. 이 레지스터들은 각 SPI 가 **Group 0** 에 속할지, **Group 1** 에 속할지를 결정합니다.

- **SPI (Shared Peripheral Interrupts)**: 외부 장치에서 발생하는 인터럽트입니다. 각 레지스터는 32 개의 SPI 에 해당하는 비트들을 포함합니다.
- **Group 0**: 이 그룹에 속한 인터럽트는 FIQ (Fast Interrupt Request)로 처리될 수 있으며, 일반적으로 중요한 인터럽트로 취급됩니다.
- **Group 1**: Group 1 에 속한 인터럽트는 IRQ (Interrupt Request)로 처리됩니다. 이 그룹의 인터럽트는 보통 덜 중요한 용도로 사용됩니다.

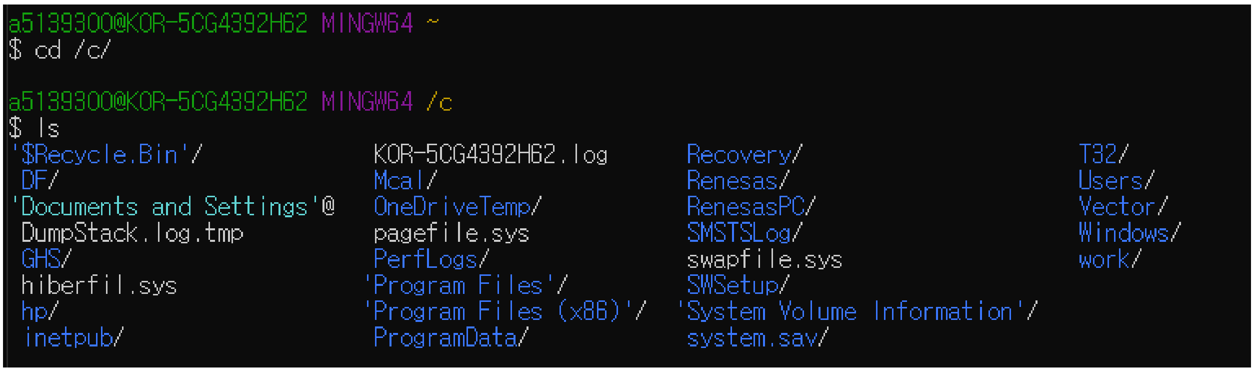
12 Flash Writer Build

Git Bash 를 실행하여 아래와 같은 화면 출력



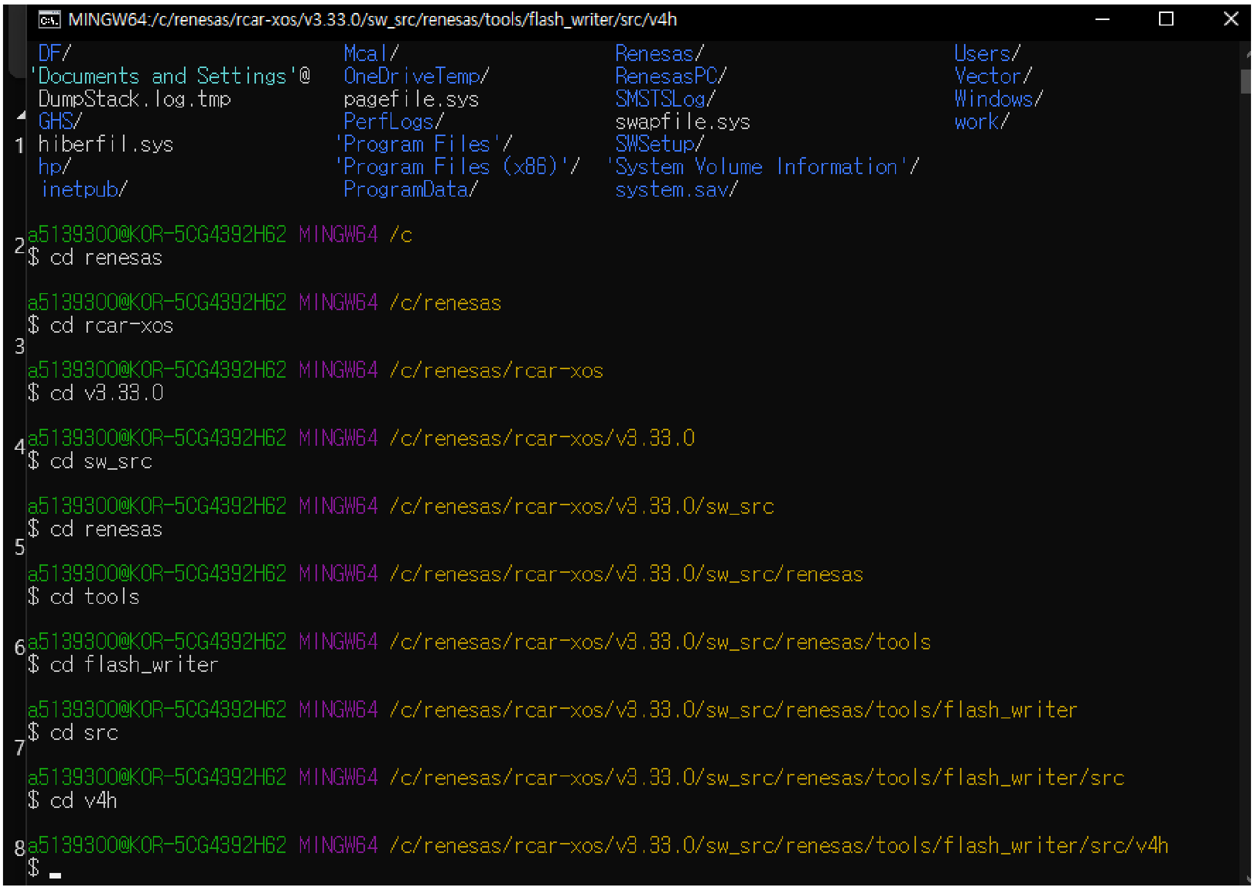
```
ntuser.dat.LOG2
NTUSER.DAT{e71279b3-b852-11ef-98c9-e08f4cd17762}.TM.blf
NTUSER.DAT{e71279b3-b852-11ef-98c9-e08f4cd17762}.TMContainer00000000000000000001.regtrans-ms
NTUSER.DAT{e71279b3-b852-11ef-98c9-e08f4cd17762}.TMContainer00000000000000000002.regtrans-ms
ntuser.ini
OneDrive/
OneDrive - Renesas Electronics Corporation'/
Pictures/
PrintHood@
Recent@
Saved Games'/
Searches/
SendTo@
smartconfigurator/
Templates@
Videos/
시작 메뉴'@

5139300@KOR-5CG4392H62 MINGW64 ~
$ cd /c/

5139300@KOR-5CG4392H62 MINGW64 /c
$ ls
$Recycle.Bin'/            KOR-5CG4392H62.log      Recovery/                  T32/
DF/                       Mcal/                   Renesas/                   Users/
Documents and Settings'@  OneDriveTemp/           RenesasPC/                 Vector/
DumpStack.log.tmp         pagefile.sys            SMSTSLog/                  Windows/
GHS/                      PerfLogs/               swapfile.sys               work/
hiberfil.sys             'Program Files'/         SWSetup/
hp/                      'Program Files (x86)'/  'System Volume Information'/
inetpub/                  ProgramData/            system.sav/

5139300@KOR-5CG4392H62 MINGW64 /c
$ cd renesas

5139300@KOR-5CG4392H62 MINGW64 /c/renesas
$ 
```

Mingw 가 실행되면 C 드라이버로 이동



```
a5139300@KOR-5CG4392H62 MINGW64 ~
$ cd /c/

a5139300@KOR-5CG4392H62 MINGW64 /c
$ ls
'$Recycle.Bin'/            KOR-5CG4392H62.log      Recovery/                  T32/
 DF/                       Mcal/                   Renesas/                   Users/
'Documents and Settings'@  OneDriveTemp/           RenesasPC/                 Vector/
 DumpStack.log.tmp         pagefile.sys            SMSTSLog/                  Windows/
 GHS/                      PerfLogs/               swapfile.sys               work/
 hiberfil.sys             'Program Files'/         SWSetup/
 hp/                      'Program Files (x86)'/  'System Volume Information'/
 inetpub/                  ProgramData/            system.sav/
```

이후 Make 파일이 있는 폴더로 이동



MINGW64:/c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools/flash_writer/src/v4h

```
 DF/                           Mcal/                   Renesas/                     Users/
'Documents and Settings'@      OneDriveTemp/           RenesasPC/                   Vector/
 DumpStack.log.tmp             pagefile.sys            SMSTSLog/                    Windows/
 GHS/                          PerfLogs/               swapfile.sys                 work/
 hiberfil.sys                 'Program Files'/         SWSetup/
 hp/                          'Program Files (x86)'/  'System Volume Information'/
 inetpub/                      ProgramData/            system.sav/

a5139300@KOR-5CG4392H62 MINGW64 /c
$ cd renesas

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas
$ cd rcar-xos

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos
$ cd v3.33.0

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0
$ cd sw_src

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src
$ cd renesas

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src/renesas
$ cd tools

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools
$ cd flash_writer

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools/flash_writer
$ cd src

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools/flash_writer/src
$ cd v4h

a5139300@KOR-5CG4392H62 MINGW64 /c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools/flash_writer/src/v4h
$ _
```

매뉴얼에서 가이드한 내용을 참고하여 아래 커맨드 입력

- Make clean
- Make LSI=V4H

R-Car V4H

```
$ make clean
$ make LSI=V4H
```

Figure 4-3 example of building Flash writer for V4H

폴더의 형식은 아래와 같다.

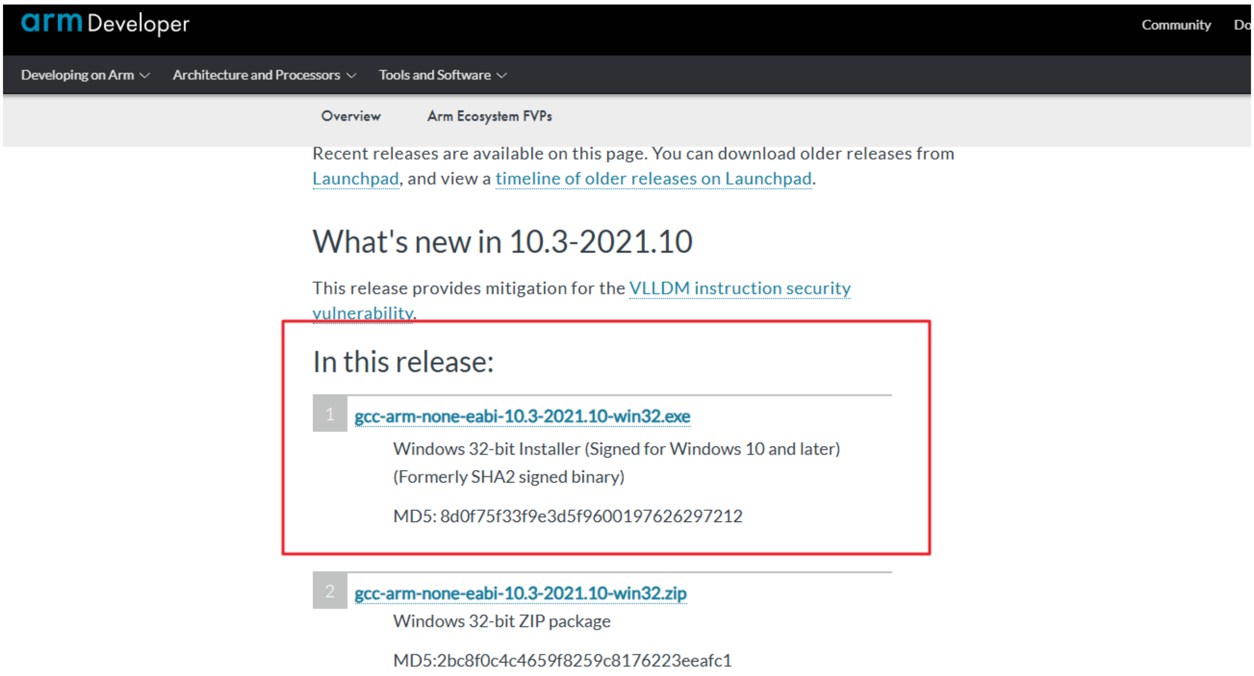
Cd /c/ [enter]
Cd /c/renesas/rcar-xos/v3.33.0/sw_src/renesas/tools/flash_writer/src/v4h [enter]

13 ICUMX Loader Build

Mingw 를 실행하여 make 파일이 있는 폴더로 이동

14 Cx Loader build

Gcc 를 윈도우 버전으로 설치한다.
https://developer.arm.com/downloads/-/gnu-rm/10-3-2021-10



Recent releases are available on this page. You can download older releases from Launchpad, and view a timeline of older releases on Launchpad.

What's new in 10.3-2021.10

This release provides mitigation for the VLLDM instruction security vulnerability.

In this release:

1 gcc-arm-none-eabi-10.3-2021.10-win32.exe

Windows 32-bit Installer (Signed for Windows 10 and later) (Formerly SHA2 signed binary)

MD5: 8d0f75f33f9e3d5f9600197626297212

2 gcc-arm-none-eabi-10.3-2021.10-win32.zip

Windows 32-bit ZIP package

MD5:2bc8f0c4c4659f8259c8176223eeafc1

설치시 공백이나 기호가 들어가지 않도록 주의한다.

설치가 완료되면 매뉴얼의 가이드와 같이 컴파일을 진행한다.

(1) Prepare the GCC compiler

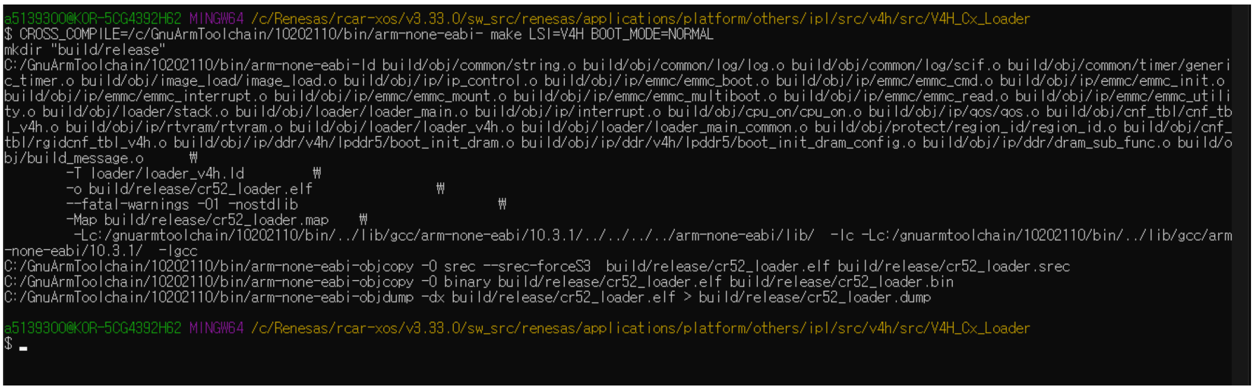
Create a WORK directory. Prepare the GCC compiler according to the following procedure.
(S4)

```
$ cd $WORK
$ wget https://developer.arm.com/-/media/Files/downloads/gnu-a/\
  10.3-2021.07/binrel/gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf.tar.xz
$ tar xvf gcc-arm-10.3-2021.07-x86_64-aarch64-none-elf.tar.xz
```

(V4H/V4M)

```
$ cd $WORK
$ wget https://developer.arm.com/-/media/Files/downloads/gnu-rm/\
  10.3-2021.10/gcc-arm-none-eabi-10.3-2021.10-x86_64-linux.tar.bz2
$ tar xvf gcc-arm-none-eabi-10.3-2021.10-x86_64-linux.tar.bz2
```

- 소스코드가 있는 폴더로 이동한다.
  - cd /c/Renesas/rcar-xos/v3.33.0/sw_src/renesas/applications/platform/others/ipl/src/v4h/src/V4H_Cx_Loader/
- Clean 을 진행한다.
  - CROSS_COMPILE=/c/GnuArmToolchain/10202110/bin/arm-none-eabi- make clean
- 컴파일을 진행한다.
  - CROSS_COMPILE=/c/GnuArmToolchain/10202110/bin/arm-none-eabi- make LSI=V4H BOOT_MODE=NORMAL



```
a5139300@KOR-5CG4392H62 MINGW64 /c/Renesas/rcar-xos/v3.33.0/sw_src/renesas/applications/platform/others/ipl/src/v4h/src/V4H_Cx_Loader
$ CROSS_COMPILE=/c/GnuArmToolchain/10202110/bin/arm-none-eabi- make LSI=V4H BOOT_MODE=NORMAL
mkdir "build/release"
C:/GnuArmToolchain/10202110/bin/arm-none-eabi-ld build/obj/common/string.o build/obj/common/log/log.o build/obj/common/log/scif.o build/obj/common/timer/generi
c_timer.o build/obj/image_load/image_load.o build/obj/ip/ip_control.o build/obj/ip/emmc/emmc_boot.o build/obj/ip/emmc/emmc_cmd.o build/obj/ip/emmc/emmc_init.o
build/obj/ip/emmc/emmc_interrupt.o build/obj/ip/emmc/emmc_mount.o build/obj/ip/emmc/emmc_multiboot.o build/obj/ip/emmc/emmc_read.o build/obj/ip/emmc/emmc_utili
ty.o build/obj/loader/stack.o build/obj/loader/loader_main.o build/obj/ip/interrupt.o build/obj/cpu_on/cpu_on.o build/obj/ip/qos/qos.o build/obj/cnf_tbl/cnf_tb
l_v4h.o build/obj/ip/rtvram/rtvram.o build/obj/loader/loader_v4h.o build/obj/loader/loader_main_common.o build/obj/protect/region_id/region_id.o build/obj/cnf_
tbl/rgidcnf_tbl_v4h.o build/obj/ip/ddr/v4h/lpddr5/boot_init_dram.o build/obj/ip/ddr/v4h/lpddr5/boot_init_dram_config.o build/obj/ip/ddr/dram_sub_func.o build/o
bj/build_message.o      ₩
        -T loader/loader_v4h.ld         ₩
        -o build/release/cr52_loader.elf                ₩
        --fatal-warnings -O1 -nostdlib                          ₩
        -Map build/release/cr52_loader.map    ₩
         -Lc:/gnuarmtoolchain/10202110/bin/../lib/gcc/arm-none-eabi/10.3.1/../../../../arm-none-eabi/lib/  -lc -Lc:/gnuarmtoolchain/10202110/bin/../lib/gcc/arm
-none-eabi/10.3.1/  -lgcc
C:/GnuArmToolchain/10202110/bin/arm-none-eabi-objcopy -O srec --srec-forceS3  build/release/cr52_loader.elf build/release/cr52_loader.srec
C:/GnuArmToolchain/10202110/bin/arm-none-eabi-objcopy -O binary build/release/cr52_loader.elf build/release/cr52_loader.bin
C:/GnuArmToolchain/10202110/bin/arm-none-eabi-objdump -dx build/release/cr52_loader.elf > build/release/cr52_loader.dump

a5139300@KOR-5CG4392H62 MINGW64 /c/Renesas/rcar-xos/v3.33.0/sw_src/renesas/applications/platform/others/ipl/src/v4h/src/V4H_Cx_Loader
$ _
```

컴파일이 완료되면 Build 폴더를 참조하여 이미지와 헥사 파일을 확인한다.