

統合開発環境 e²studio

R-Car 上で動作する Linux アプリのデバッグ手法 (解析ツール [Bus Monitor トレースウィンドウ] 使用編)

要旨

本アプリケーションノートは、R-Car 上で動作する Linux の CLI (コマンドインタフェース) で R-Car V4H のトレースデータを取得し、R-Car 用 e²studio を用いて、Bus Monitor トレースウィンドウに表示する操作手順を紹介します。

動作確認デバイス

R-Car V4H

目次

1. 使用環境	2
2. 接続図	2
3. Bus Monitor 用のトレースデータ準備	3
3.1 R-Car 上で動作する Linux の起動	3
3.2 トレースデータ取得実行ファイルの準備	4
3.3 Bus Monitor 機能の起動方法	8
3.3.1 elf ファイルを動作させるモジュールのロード	8
3.3.2 Event matching 有り、QoS モニター有りで動作させる場合	8
4. デバッグでの操作	11
4.1 “Tracing” プロジェクト作成	11
4.2 トレースプロファイル設定	12
4.3 トレースデータのインポート	15
4.4 トレースデータの表示	16
改訂記録	18

1. 使用環境

本書の説明に使用した環境の一覧を以下に示します。

表 1-1 使用環境一覧

項目	詳細
R-Car V4H Reference Board	White Hawk [RTP8A779G2ASKB0F10SA001]
HOST PC	・ Windows 10 (terminal display console 用途, また、Virtual Box の仮想環境上で Linux を起動している) ・ Ubuntu 20.04 LTS (e²studio、TFTP server、NFS server)
R-Car 向け e ² studio	Version 2023-03 rcar-xos_tool_e2studio_ubuntu_2023-03.R20230313-1010.tar.gz
R-Car V4H SDK	V3.12.0 rcar-xos_developer-adas-bootloader_v3.12.0_release.tar.gz rcar-xos_platform-sdkX_v3.12.0_release.tar.gz (sample ソース)
R-Car V4H yocto Linux (Kernel version)	V5.10.147 rcar-xos_tool_yocto_linux_6.13.0.tar.gz

本書で説明する手順は、Reference Board 上の R-Car に Linux 環境をセットアップ済みの環境を使用しています。Linux 環境のセットアップ手順は SDK に含まれる User's Manual を参照してください。

2. 接続図

R-Car V4H の Reference Board と Host PC との接続図を以下に示します。Host PC と Reference Board 間の IP アドレスは図のように設定しています。接続設定で設定する IP アドレスは、使用する環境に合わせた IP アドレスを指定してください。

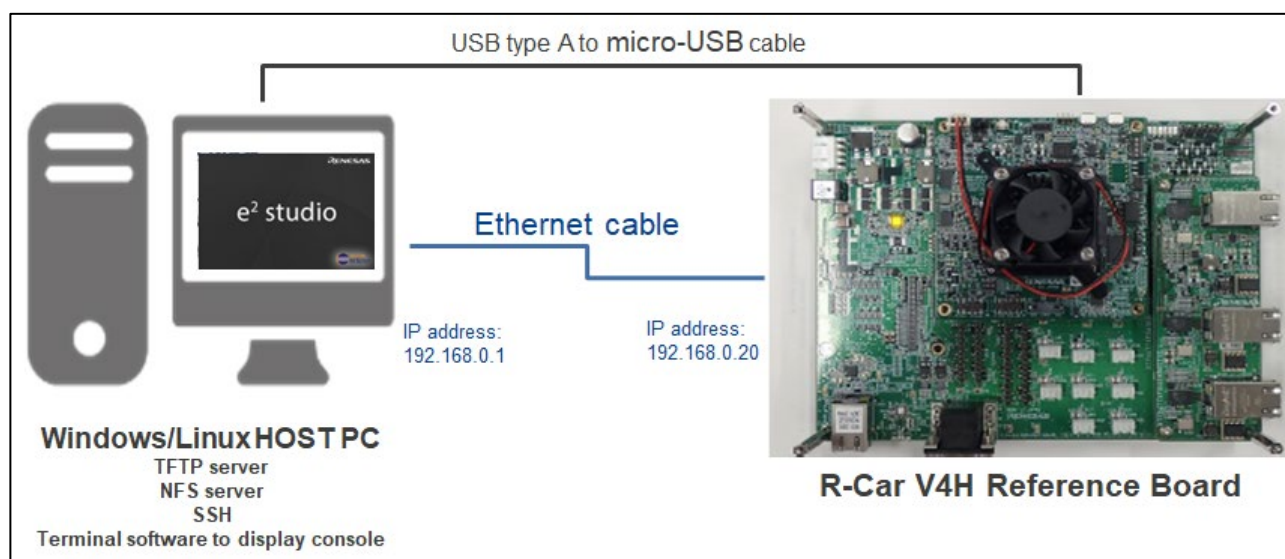


図 2-1 Host PC と Reference Board の接続図

3. Bus Monitor 用のトレースデータ準備

3.1 R-Car 上で動作する Linux の起動

R-Car V4H の電源を入れ、Host PC から R-Car 上で動作する Linux イメージを読み込んで起動します。
シリアル コンソールからのブート ログ メッセージは次のようになります。

```
Using ethernet@e6800000 device
TFTP from server 192.168.0.1; our IP address is 192.168.0.20
Filename 'Image'.
Load address: 0x48080000
Loading: #####
:
#####
6 MiB/s
done
Bytes transferred = 33839616 (2045a00 hex)
Using ethernet@e6800000 device
TFTP from server 192.168.0.1; our IP address is 192.168.0.20
Filename 'r8a779g0-whitehawk.dtb'.
Load address: 0x48000000
Loading: #####
5.5 MiB/s
Done
```

図 3-1 データ準備手順 1

R-Car 上で動作する Linux イメージが起動した後に、ログインして Linux のバージョンを確認します。

```
[ OK ] Started Target Communication Framework agent.
[ OK ] Started Avahi mDNS/DNS-SD Stack.
[ OK ] Reached target Multi-User System.
Starting Update UTMP about System Runlevel Changes...
[ OK ] Started Update UTMP about System Runlevel Changes.

Poky (Yocto Project Reference Distro) 3.1.11 v4x ttyS00
v4x login: root (パスワードを入力して enter 押下)
root@v4x:~#
root@v4x:~# uname -a (コマンド入力して enter 押下)
Linux v4x 5.10.147-yocto-standard #1 SMP PREEMPT Fri Mar 3 11:25:12 UTC 2023 aarch64 aarch64 aarch64 GNU/Linux
root@v4x:~#
```

図 3-2 データ準備手順 2

3.2 トレースデータ取得実行ファイルの準備

Host PC に e²studio をインストールした際に、事前構築されたスタブファイルが、e²studio の DebugComp/RCar ディレクトリに格納されています。

「/home/<ユーザ名>/eclipse/com.renesas.platform_XXXXXXXXXX/DebugComp/RCar」

```

XXXX@XXXXXp:~$ ls -aIF                                (コマンド入力して enter 押下)
合計 XXXXXX
drwxrwxr-x 9 XXXX XXXX   XXXX  MMM DD HH:MM ./
drwxrwxr-x 3 XXXX XXXX   XXXX  MMM DD HH:MM ../
:
-rw-rw-r-- 1 XXXX XXXX  572088  MMM DD HH:MM perfmodule2-5_10_0.ko
-rw-rw-r-- 1 XXXX XXXX  575584  MMM DD HH:MM perfmodule2-5_10_41.ko
:
-rw-rw-r-- 1 XXXX XXXX  494048  MMM DD HH:MM saperfmon.elf
-rw-rw-r-- 1 XXXX XXXX  630592  MMM DD HH:MM saperfmon_vh4.elf
:
XXXX@XXXXXp:~$

```

図 3-3 データ準備手順 3

R-Car 上で動作する Linux のバージョンとターゲットボードに対応する“perfmodule.ko”と“saperfmon.elf”を選択し、R-Car 上で動作する Linux の任意ディレクトリ（WORK ディレクトリ）に格納します。

但し、今回動作させた環境では、以下のファイルを使用しています。

- impsample_core_link (トレースデータに Bus Monitor[QOS]メッセージを追加するためのサンプルアプリケーション)
- perfmodule.ko (elf ファイルを動作させるためのファイル)
- saperfmon.elf (トレースデータを取得するための実行ファイル)

図 3-4 データ準備手順 4

R-Car 上で動作する Linux の任意ディレクトリに格納後、実行させるファイルに実行権を付与します。

```

root@v4x:~/XXXXXXXX# chmod 755 ./saperfmon.elf ./impsample_core_link      (コマンド入力して enter 押下)
root@v4x:~/XXXXXXXX#
root@v4x:~/XXXXXXXX# ls -alF      (コマンド入力して enter 押下)
total XXXX
:
-rwxr-xr-x 1 root root XXXXXX HHH DD YYYY impsample_core_link*
-rw-r--r-- 1 root root XXXXXX HHH DD YYYY perfmoudle.ko
-rwxr-xr-x 1 root root XXXXXX HHH DD YYYY saperfmon.elf*
:
root@v4x:~/XXXXXXXX#

```

図 3-5 データ準備手順 5

注記：

- ・ サンプルアプリケーション (impsample_core_link) は、SDK に含まれるサンプル (rcar-xos/v3.12.0/samples/impsample_core_link/) を使用しています。
- ・ “perfmoudle.ko” と “saperfmon.elf” のファイルは、performance-monitor のサポートディレクトリにある「DTA_guide.pdf」を参考にソースを一部変更の上、リビルドして使用しています。

「/home/<ユーザ名>/eclipse/com.renesas.platform_XXXXXXXXXX/DebugComp/RCar/performance-monitor」

performance-monitor ディレクトリは、Host PC で e2studio を起動し、“R-Car” プロジェクトを作成後にデバッグ開始することで生成されます。

[File] -> [New] -> [Renesas C/C++ Project] -> [Renesas R-Car]を選択し、[New C/C++ Project]ダイアログを開きます。

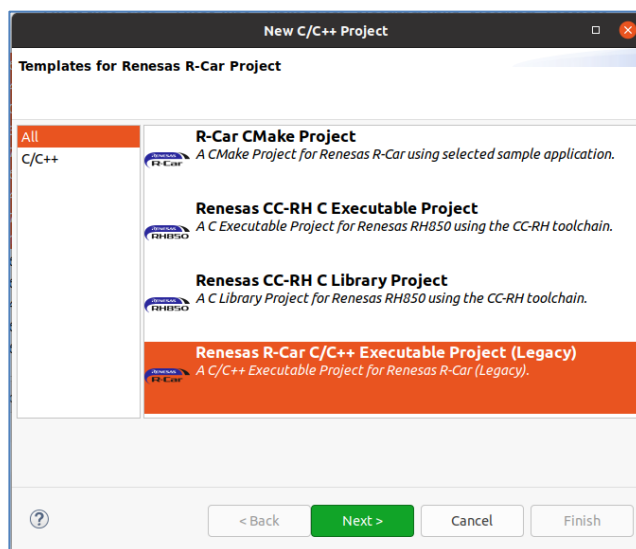


図 3-6 データ準備手順 6

簡易的なプロジェクトとして「Hello World」を選択し、[Next]ボタンを押下します。

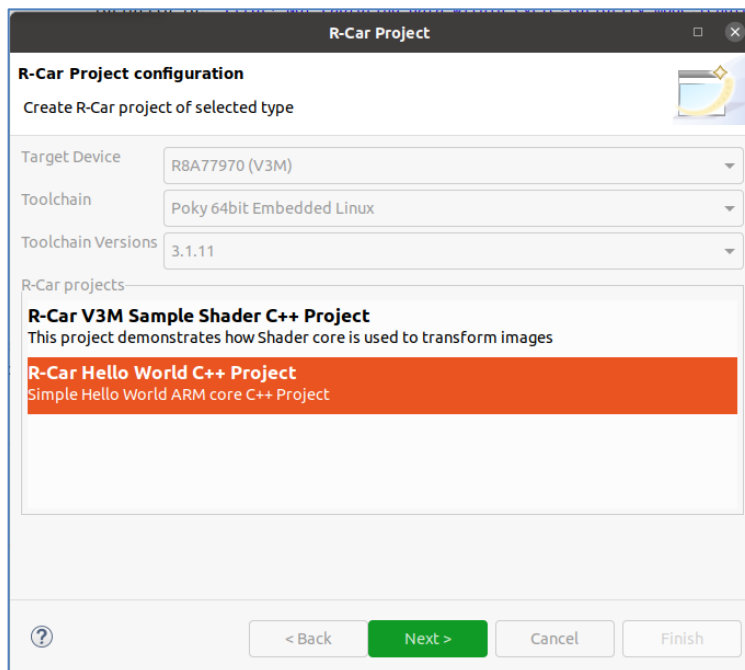


図 3-7 データ準備手順 7

任意プロジェクト名を付けて、[Finish]ボタンを押下します。

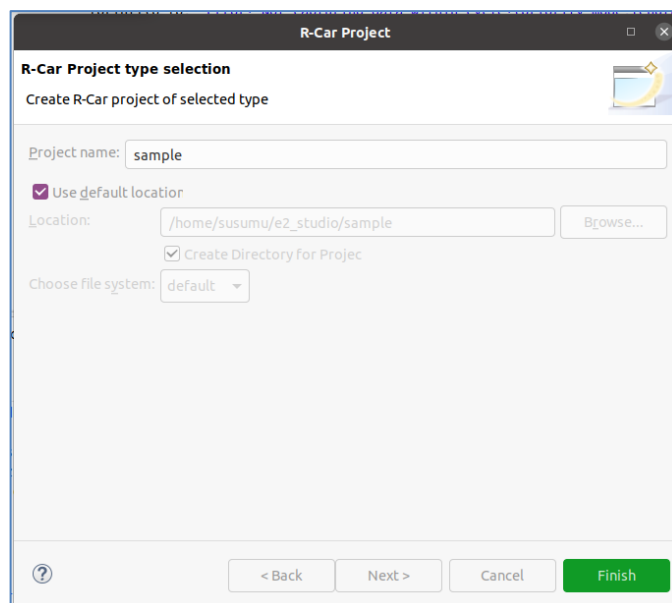


図 3-8 データ準備手順 8

プロジェクト一覧に作成されたプロジェクトが表示されますので、デバッグを開始します。起動に失敗しても「~/DebugComp/RCar」配下に performance-monitor のサポートディレクトリが生成されます。

「DTA_guide.pdf」を参照してください。

サポートディレクトリのパスについては、e²studio でメニュー[Help] → [About e²studio for R-Car] より [About e²studio for R-Car] ダイアログを開き、[Installation Details] ボタンを押下します。

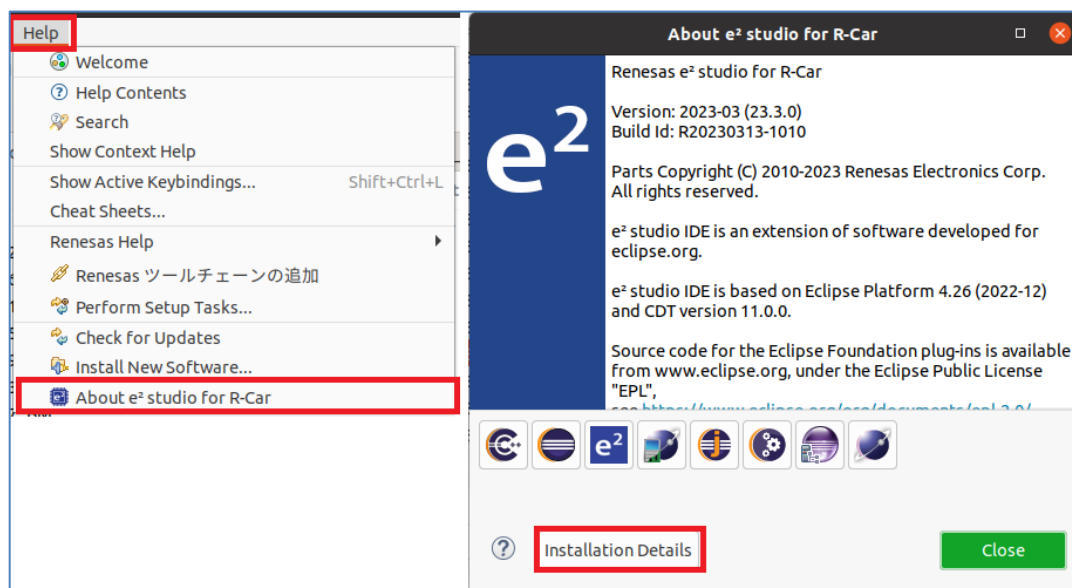


図 3-9 データ準備手順 9

[e²studio for R-Car Installation Details]が開きますので、[Support Folders]タブを選択することで、サポートディレクトリのパスを確認が可能です。

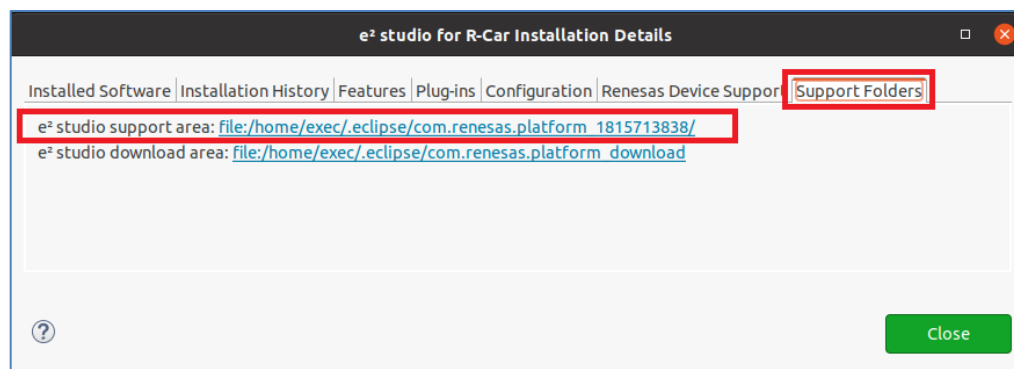


図 3-10 データ準備手順 10

3.3 Bus Monitor 機能の起動方法

R-Car 上で動作する Linux の任意ディレクトリ (WORK ディレクトリ) 内で実行します。

3.3.1 elf ファイルを動作させるモジュールのロード

作業を開始する際、先にモジュールをロードします。

(既にロードしているモジュールをロードしようとした場合はエラーメッセージが表示されます)

※アンロードする場合は、rmmod コマンドを実行します。

```
root@v4x:~/XXXXXXX# insmod ./perfmodule.ko          (コマンド入力して enter 押下)
[ 711.531951] Perfmodule - Board Type V4H detected (0x5c12)
root@v4x:~/XXXXXXX#
root@v4x:~/XXXXXXX# insmod ./perfmodule.ko          (コマンド入力して enter 押下)
insmod: ERROR: could not insert module ./perfmodule.ko: File exists
root@v4x:~/XXXXXXX#
```

図 3-11 データ準備手順 11

3.3.2 Event matching 有り、QoS モニター有りで動作させる場合

トレースデータ取得実行ファイル (saperfmon.elf) にオプション<-qevt>, <40, 41, 42, 43 (CC10-4 のマスタ)>を指定して実行すると、"trace_YYYYMMDD_HHMMSS"ディレクトリが作成され、トレースが開始されます。その後、サンプルアプリケーション (impsample_core_link) を実行します。

```
root@v4x:~/XXXXXXX# ./saperfmon.elf -qevt 40,41,42,43 &          (コマンド入力して enter 押下)
[1] 499
root@v4x:~/e2studio_BusMon_3# Current working dir: /home/root/XXXXXXX (任意ディレクトリ名)
uio PMU driver 1 auto-detection: /dev/uio369
uio DBSC driver auto-detection: /dev/uio368
uio QoS driver auto-detection: /dev/uio367
Trace directory: /home/root/XXXXXXX/trace_YYYYMMDD_HHMMSS

QoS Flat Event Summary
=====
0 - Merge_int(W)
1 - Secure_engine(W)
:
V4H config end, reading thread started.
Time translation sent back.
Time translation sent back.
Initial sent back.

root@v4x:~/XXXXXXX# ./impsample_core_link          (コマンド入力して enter 押下)
<IMP Framework Demo start>
```

```

RCAR_SOC      V4H
ATMLIB Version 2.20.0
IMPFW Version 3.12.0
Execute IMP with IMP core and PSC core
-- OutPutMemory Image data of SPM --
00 00 00 00 04 00 00 00 08 00 00 00 0C 00 00 00
:
F0 10 F4 10 F8 10 FC 10
[RESULT] PASS
<IMP Framework Demo end>

impsample_core_link return = 0 (OK:0, NG:else)
root@v4x:~/XXXXXXX#

```

図 3-12 データ準備手順 12

注記：

- ・“trace_YYYYMMDD_HHMMSS”ディレクトリ内の trace.txt にて Bus Monitor メッセージが追加されていることが確認できます。

```

:
QOS: CC10 (W):0x0000001E CC10 (R):0x0000002E CC11 (W):0x00000000 CC11 (R):0x00000000, EpochTS: 1677857377620530097
QOS: CC10 (W):0x0000000F CC10 (R):0x0000002E CC11 (W):0x00000000 CC11 (R):0x00000000, EpochTS: 1677857377621569721
QOS: CC10 (W):0x0000000F CC10 (R):0x00000008A CC11 (W):0x00000000 CC11 (R):0x00000000, EpochTS: 1677857377622613065
QOS: CC10 (W):0x0000000F CC10 (R):0x0000000F CC11 (W):0x00000000 CC11 (R):0x00000000, EpochTS: 1677857377623650409
QOS: CC10 (W):0x00000740 CC10 (R):0x000000A6 CC11 (W):0x00000314 CC11 (R):0x000029EF, EpochTS: 1677857377624711213
QOS: CC10 (W):0x0000002E CC10 (R):0x000000A9 CC11 (W):0x0000002E CC11 (R):0x00000000, EpochTS: 1677857377625748678
QOS: CC10 (W):0x0000003D CC10 (R):0x0000002E CC11 (W):0x0000001E CC11 (R):0x00000000, EpochTS: 1677857377626786082
QOS: CC10 (W):0x0000005C CC10 (R):0x0000005B1 CC11 (W):0x0000000F CC11 (R):0x00000000, EpochTS: 1677857377627829966
QOS: CC10 (W):0x000000A8 CC10 (R):0x000000D6 CC11 (W):0x00000000 CC11 (R):0x00000000, EpochTS: 1677857377628874870
QOS: CC10 (W):0x0000007A CC10 (R):0x0000002D CC11 (W):0x0000000F CC11 (R):0x00000000, EpochTS: 1677857377629919054
:

```

図 3-13 データ準備手順 13

- ・ “trace_YYYYMMDD_HHMMSS”ディレクトリ内に stream_XXX ファイルが追加されていることが確認できます。

```
root@v4x:~/XXXXXXXX/trace_YYYYMMDD_HHMMSS# ls -alF                                (コマンド入力して enter 押下)
:
-rw-r--r-- 1 root root   XXX MMM DD  YYYY stream_250
:
root@v4x:~/XXXXXXXX/trace_YYYYMMDD_HHMMSS#
```

図 3-14 データ準備手順 14

- ・ “saperfmon.elf”を実行した後に、サンプルアプリケーション (impsample_core_link) を実行すると NG となる場合があります。NG でエラーになった場合、10 回程度アプリケーションを繰り返し実行してください。

```
root@v4x:~/XXXXXXXX# ./impsample_core_link                                (コマンド入力して enter 押下)
:
[RESULT] FAIL
<IMP Framework Demo end>
<< [ERROR] R_OSAL_Deinitialize : code 1024>>
impsample_core_link return = -1 (OK:0, NG:else)
root@v4x:~/XXXXXXXX# ./impsample_core_link                                (コマンド入力して enter 押下)
:
<< [ERROR] R_OSAL_Deinitialize : code 1024>>
root@v4x:~/XXXXXXXX# ./impsample_core_link                                (コマンド入力して enter 押下)
:
<< [ERROR] R_OSAL_Deinitialize : code 1024>>
root@v4x:~/XXXXXXXX# ./impsample_core_link                                (コマンド入力して enter 押下)
:
[RESULT] PASS
<IMP Framework Demo end>
impsample_core_link return = 0 (OK:0, NG:else)
root@v4x:~/XXXXXXXX#
```

図 3-15 データ準備手順 15

4. デバッグでの操作

4.1 “Tracing” プロジェクト作成

Host PC で e²studio を起動し、“Tracing” プロジェクトを作成します。

[File] -> [New] -> [Project...]を選択し、[New Project]ダイアログを開きます。

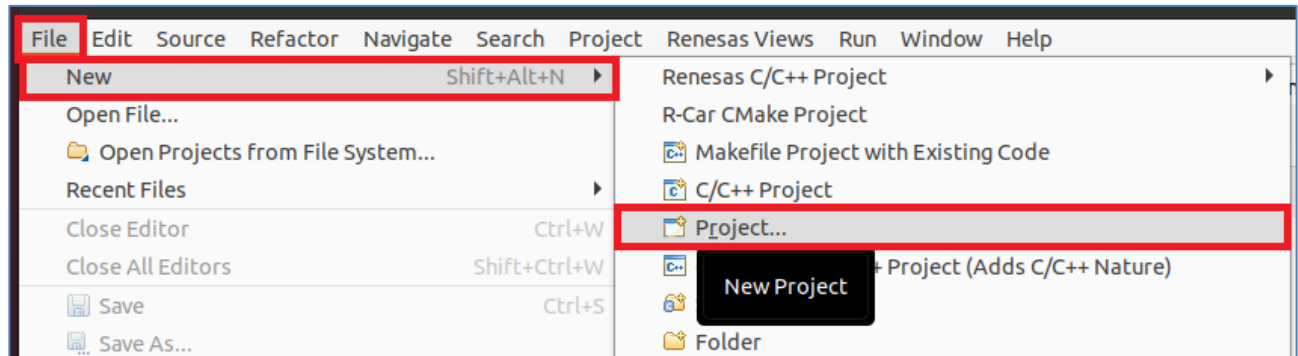


図 4-1 デバッグ操作手順 1

[New Project]ダイアログで“Tracing Project”を選択し、[Next]ボタンを押下します。

[Tracing Project]ダイアログが表示されるので、任意のプロジェクト名を入力後に、[Finish]ボタンを押下します。

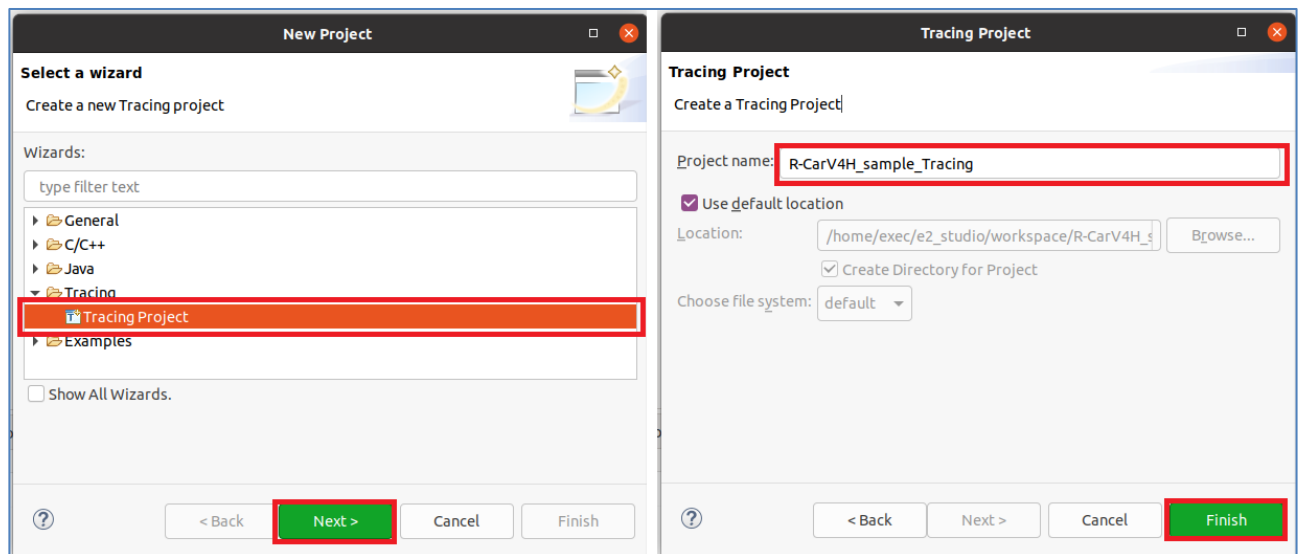


図 4-2 デバッグ操作手順 2

4.2 トレースプロファイル設定

任意のプロジェクト名を付けたトレースプロジェクトから、“Fetch Remote Traces...”を選択し、[Fetch Remote Traces]ダイアログを開きます。

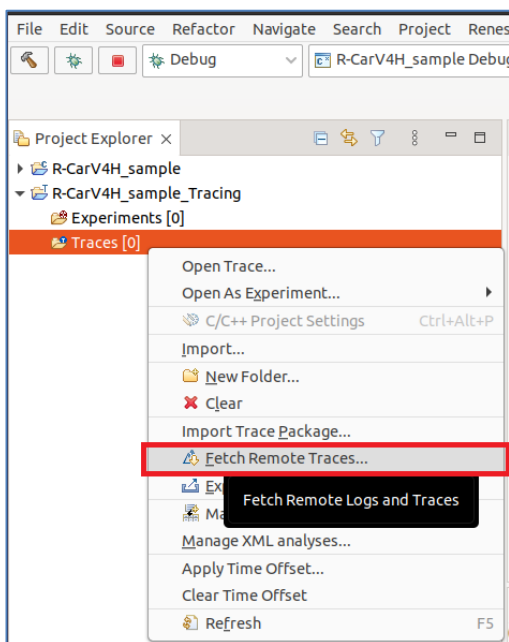


図 4-3 デバッグ操作手順 3

[Manage Profiles]ボタンを押下し、[Preferences (Filtered)]ダイアログを開きます。

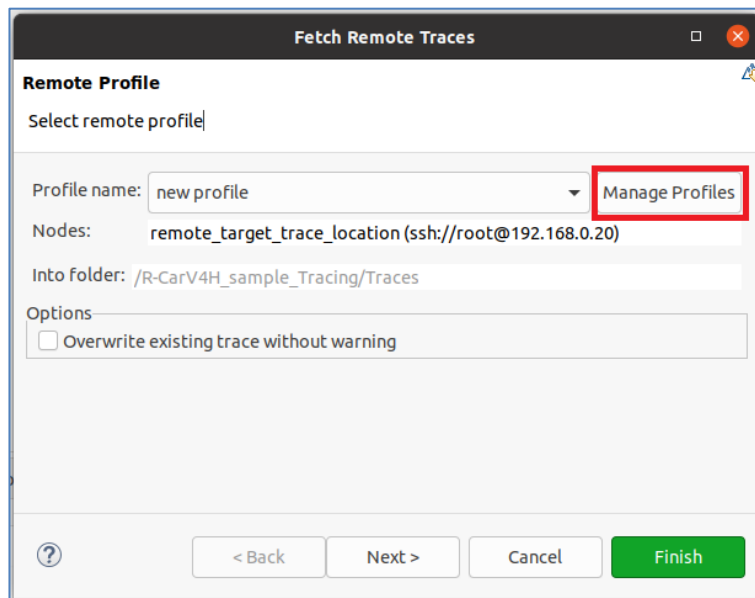


図 4-4 デバッグ操作手順 4

[Preferences (Filtered)] ダイアログでは、SSH 接続のトレースプロファイルを下記画像に従って設定します。

- ・ URI : ルートユーザー名およびターゲット (R-Car V4H Reference board) IP アドレスを指定します。
- ・ Root path : リモートトレースディレクトリ 「R-Car 上で動作する Linux の任意ディレクトリ」を指定します。
- ・ Recursive : チェック欄にチェックを付けます。
- ・ Trace type : 「R-Car Performance Trace」フォーマットを選択します。

[Preferences (Filtered)] ダイアログで [Apply and Close] ボタンを押下し、[Fetch Remote Traces] ダイアログに戻ります。

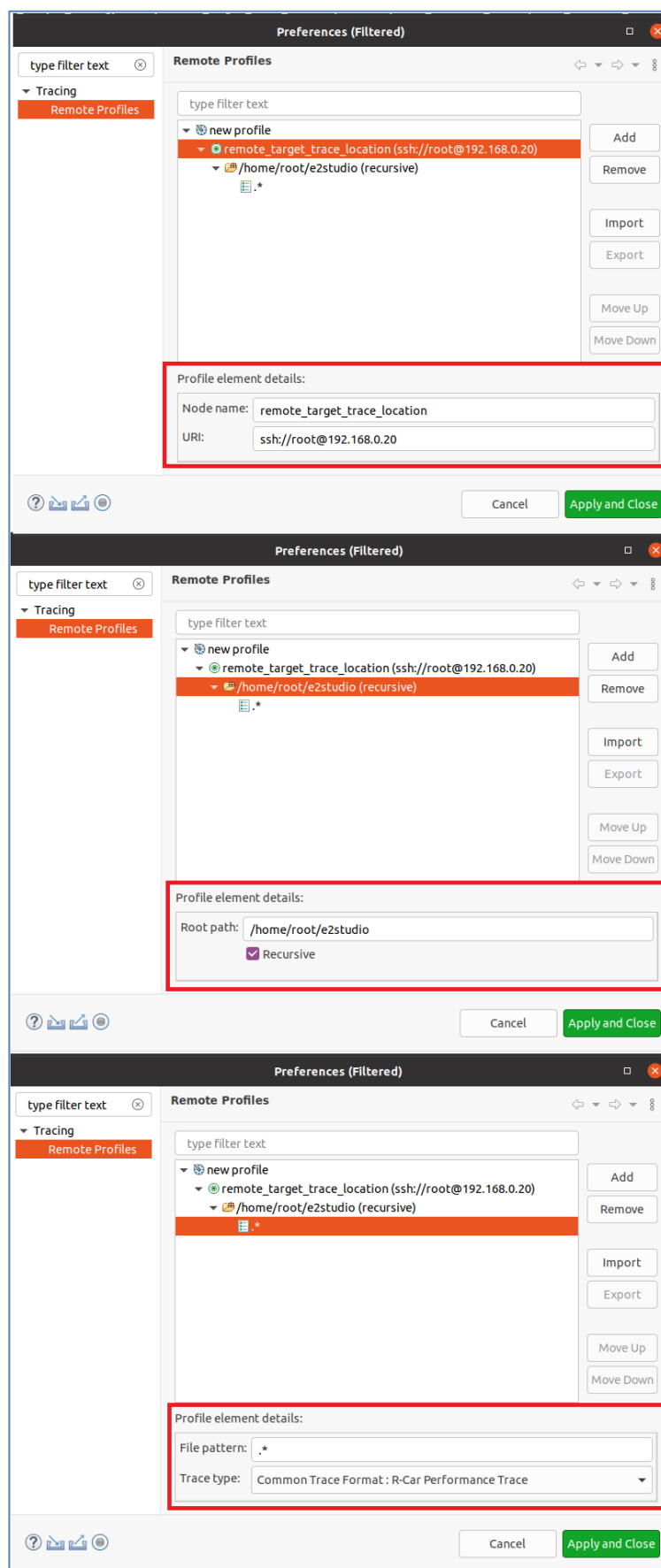


図 4-5 デバッグ操作手順 5

4.3 トレースデータのインポート

[Fetch Remote Traces] ダイアログで [Next] ボタンを押下します。

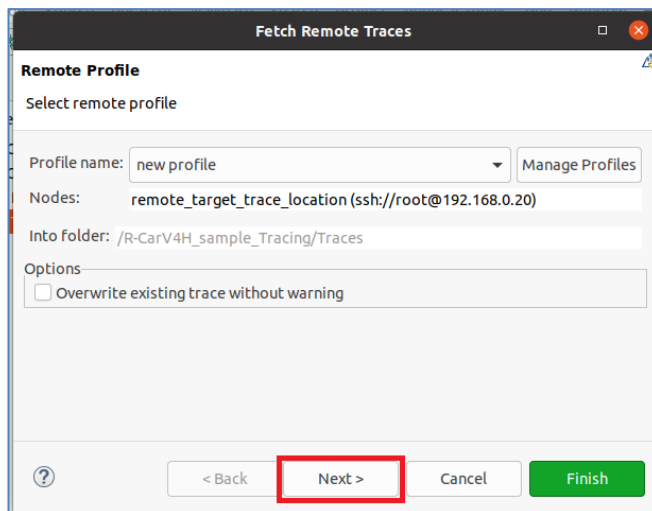


図 4-6 デバッグ操作手順 6

必要なトレースを選択し、テキストログを除くすべてをインポートします。[Finish] ボタンを押下してください。

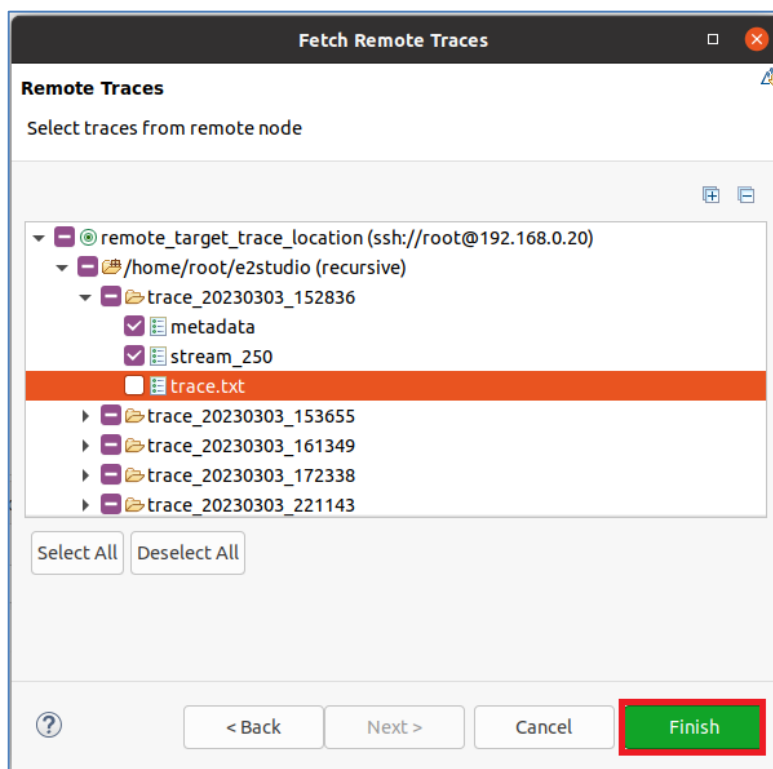


図 4-7 デバッグ操作手順 7

4.4 トレースデータの表示

トレースプロジェクトにトレースディレクトリが表示されます。トレースディレクトリを右クリックし、トレースタイプを“R-Car Performance Trace”に設定します。

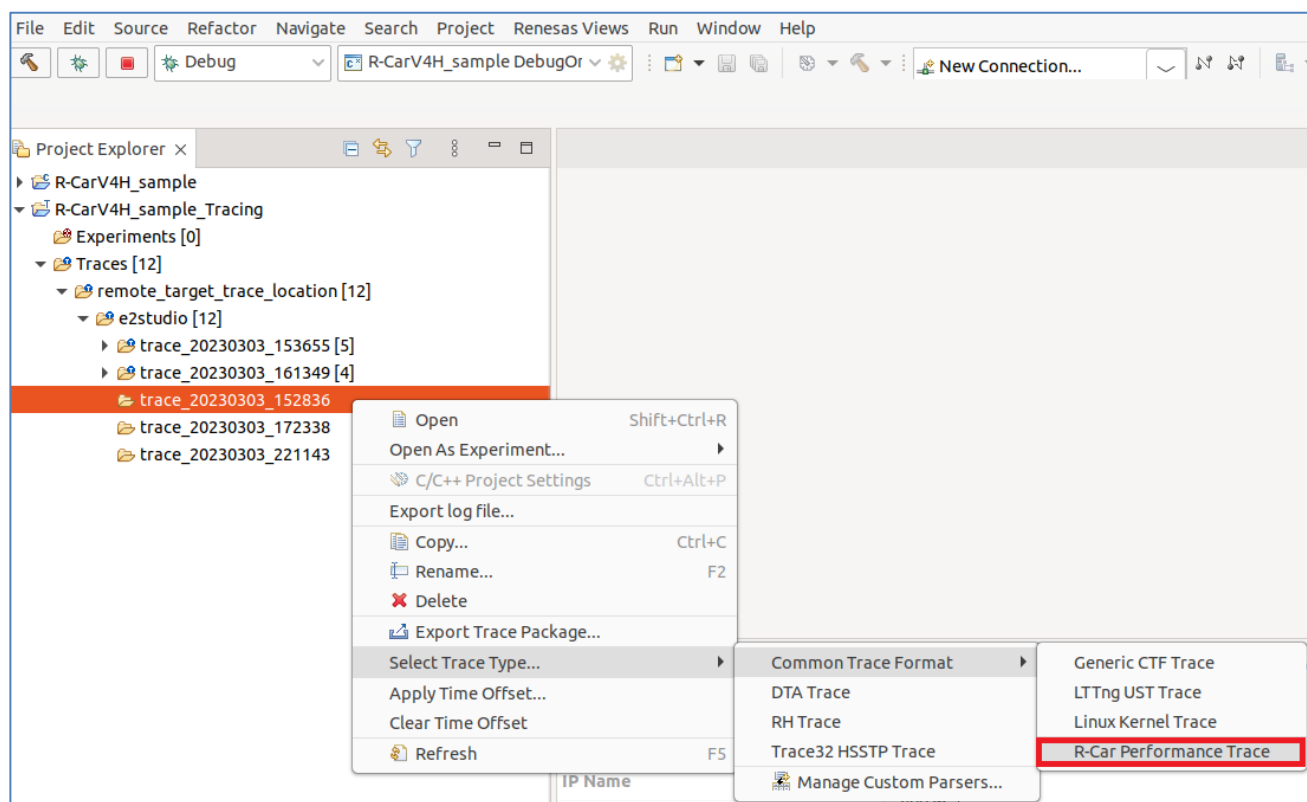


図 4-8 デバッグ操作手順 8

トレースプロジェクトから“trace_YYYYMMDD_HHMMSS”ディレクトリをダブルクリックすると、トレースの一覧が表示されます。

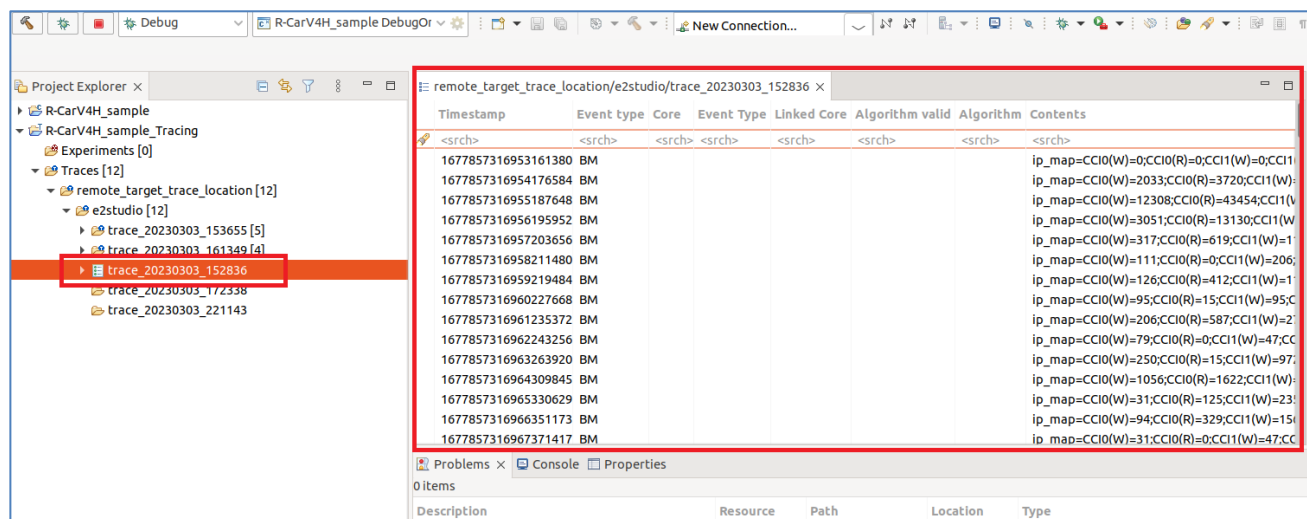


図 4-9 デバッグ操作手順 9

トレースプロジェクトから“System Performance - Bus Monitor (QOS trace)”をダブルクリックすると、Bus Monitor 用 View が開きます。

IP Name 欄にある[CCI0(R)]、[CCI0(W)]、[CCI1(R)]、[CCI1(W)]の各チェック欄を有効にし、トレースの一覧から選択した時間帯のトレースデータが View に表示されます。

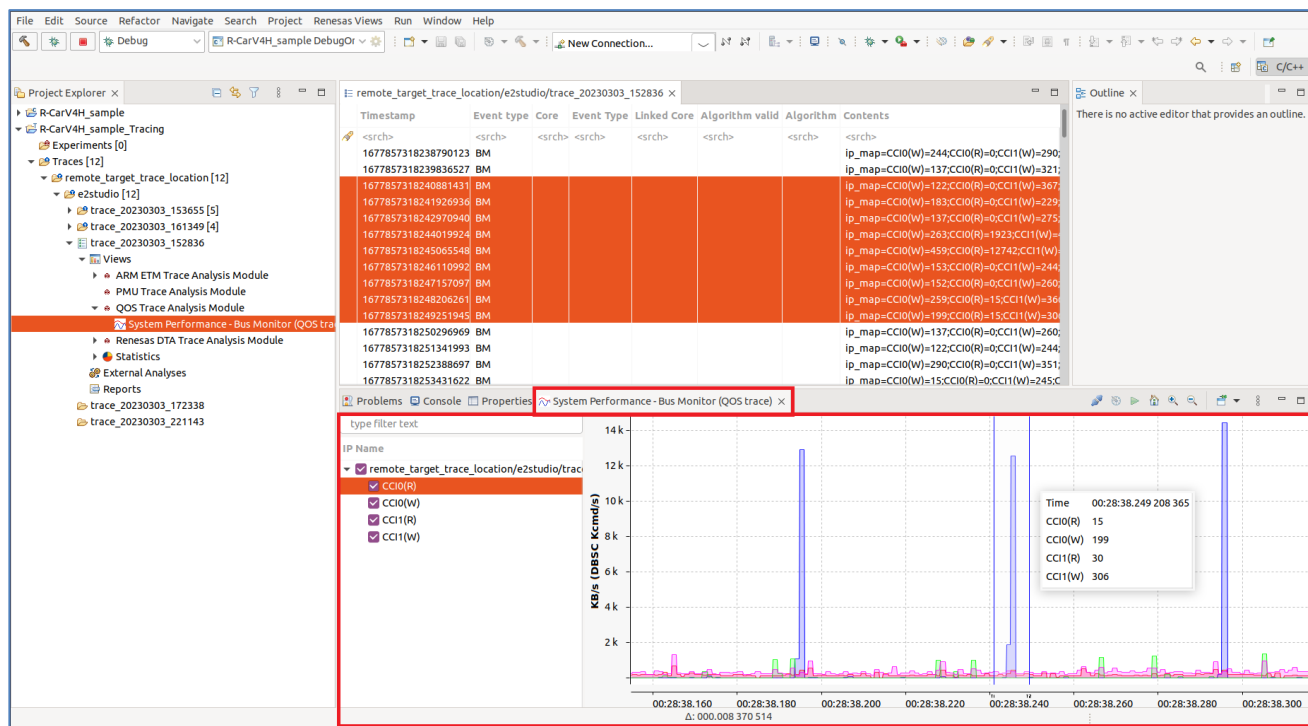


図 4-10 デバッグ操作手順 10

[Window] -> [Show View] -> [Other...] で [Show View] ダイアログが開きますので、[Renesas R-Car Trace] -> [System Performance - Bus Monitor (QOS trace)]を選択し、[Open] ボタン押下することでも Bus Monitor 用 View を開くことができます。

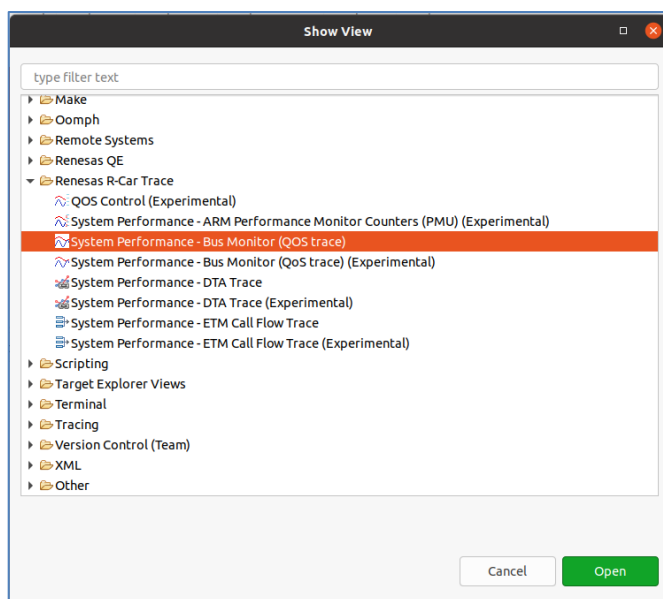


図 4-11 デバッグ操作手順 11

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1. 00	2023. 10. 31	－	新規作成

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策

CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、 $V_{IL}(\text{Max.})$ から $V_{IH}(\text{Min.})$ までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、 $V_{IL}(\text{Max.})$ から $V_{IH}(\text{Min.})$ までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違くと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含まれます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等

高品質水準：輸送機器（自動車、電車、船舶等）、交通管制（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。

7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
 8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
 9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
 10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
 11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
 12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものとしします。
 13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
 14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。
- 注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。
- 注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev. 5.0-1 2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/