

RX64M, RX71M グループ

RX Driver Package Ver.1.02

R01AN2606JJ0104 Rev.1.04 2016.5.16

要旨

本書は、RX64M, RX71M グループ用 RX Driver Package Ver.1.02 のユーザーズマニュアルです。

本ユーザーズマニュアルでは、RX Driver Package の構成、特徴、使用方法と RX Driver Package を利用した サンプルアプリケーションプログラムについて説明します。

動作確認デバイス

- RX64M グループ (Renesas Starter Kit+ RX64M)
- RX71M グループ (Renesas Starter Kit+ RX71M)

お客様の製品にてご利用される際は、お客様の環境に合わせて十分に評価してください。

また、本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、 十分評価してください。

関連ドキュメント

RX ファミリ ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685JU)

Firmware Integration Technology ユーザーズマニュアル(R01AN1833JU)

RX ファミリ e² studio に組み込む方法 Firmware Integration Technology (R01AN1723JU)

RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826JJ)

RX Driver Package Application に付属するユーザーズマニュアル



目次

1. 1	既要3
1.1	適用3
1.2	動作環境3
2. I	RX Driver Package とは4
2.1	システム構成4
2.2	RX Driver Package の特徴5
3. I	RX64M, RX71M クルーフ用 RX Driver Package の構成6
3.1	フォルダ構成6
3.2	モジュール構成
3.3	FIT モジュールー覧8
4. 1	史用万法10
4.1	説明で使用する環境10
4.2	e ² studio に RX Driver Package をインストールする10
4.3	アプリケーションの作成11
4.	3.1 ワークスペースとプロジェクトを作成する11
4.	3.2 FIT プラグインで FIT モジュールをインストールする
4.	3.3 LED 点灯プログラムを作成する20
4.	3.4 プログラムをビルドし動作を確認する21
5. I	RX Driver Package Application について24
5.1	RX Driver Package Application の構成24
6.	甫足
6.1	M3S-T4-Tiny(TCP/IP プロトコルスタックライブラリ)について
ホー	ムページとサポート窓口



1. 概要

1.1 適用

本ユーザーズマニュアルは、RX64M, RX71M グループ用 RX Driver Package Ver1.02 に適用します。

1.2 動作環境

本パッケージの動作環境を以下に示します。

表 1-1 動作環境(RX64M)

対応 MCU	RX64M グループ
評価ボード	Renesas Starter Kit+ RX64M
統合開発環境(IDE)	e² studio V4.0.2 以降
	CS+ V3.01.00 以降
クロスツール	RX ファミリ用 C/C++コンパイラパッケージ V2.03.00 以降
エミュレータ	E1, E20

表 1-2 動作環境 (RX71M)

対応 MCU	RX71M グループ
評価ボード	Renesas Starter Kit+ RX71M
統合開発環境(IDE)	e² studio V4.0.2 以降
	CS+ V3.01.00 以降
クロスツール	RX ファミリ用 C/C++コンパイラパッケージ V2.03.00 以降
エミュレータ	E1, E20



2. RX Driver Package とは

RX Driver Package は、開発に必要となる以下のモジュール群を一つのパッケージとしてまとめたソフトウェア・プラットフォーム(フレームワーク)です。複数のモジュールがパッケージングされているため、 個別に入手する必要がなく、開発に直ぐに着手できます。

- ボード・サポート・パッケージ(BSP)モジュール
- FIT 周辺機能モジュール (無償版)
- FIT ミドルウェアモジュール (無償版)
- FIT インタフェースモジュール

また、RX Driver Package を活用したサンプルアプリケーションプログラム(RX Driver Package Application) を用いることで、ユーザアプリケーション層の開発が容易になります。

2.1 システム構成

RX Driver Package のシステム構成を以下に示します。



図 2-1 システム構成

2.2 RX Driver Package の特徴

RX Driver Package の特徴を以下に示します。

(a) 必要なモジュールを選択し、すぐにアプリケーションプログラムを開発可能

システムに必要なモジュールをパッケージから選択するだけで簡単にシステムを構築できます。あとはア プリケーションプログラムを開発するだけです。



図 2-2 構築イメージ

(b) 無償で利用可能

RX Driver Package に入っているモジュールは、全て無償で利用することができます。

なお、TCP/IP、ファイルシステム等のミドルウェアモジュールも無償版が入っています。

(c) 有償版モジュールへの入れ替えが可能

RX Driver Package に入っている無償版モジュールを製品版(有償)モジュールに置き換えることができます。

製品版(有償)を使用することでモジュールの全機能を使用することができ、また製品版に関するサポートも利用することができます。

(d) ユーザアプリケーションを含めた動作確認が可能

RX Driver Package を用いたユーザアプリケーションのサンプルとして、RX Driver Package Application を提供します。RX Driver Package Application は、RX Driver Package の各種モジュールを動作させるプログラムと、そのプログラムをビルドするためのプロジェクトファイルで構成されています。これにより、すぐにユーザアプリケーションを含めた動作確認を開始することができます。

3. RX64M, RX71M グループ用 RX Driver Package の構成

3.1 フォルダ構成

本パッケージのフォルダ構成を以下に示します。

ルネサスの Web サイトからダウンロードした本パッケージの ZIP ファイルを解凍すると、同名のフォルダ があり、その中に「FITModules」フォルダと「reference_documents」フォルダと本ドキュメントが入っていま す。

「FITModules」フォルダ内には、表 3-1に示すような各モジュールの FIT モジュール(ZIP ファイルと XML ファイル)が複数入っています。

「reference_documents」フォルダには、各開発環境へ適用するためのドキュメントが入っています。



図 3-1 RX64M, RX71M グループ用 RX Driver Package フォルダ構成



3.2 モジュール構成

本パッケージに入っている FIT モジュールの種類と構成を以下に示します。



図 3-2 RX64M, RX71M グループ用 RX Driver Package FIT モジュール構成イメージ



3.3 FIT モジュール一覧

本パッケージに入っている FIT モジュール一覧を以下に示します。

表 3-1 RX64M, RX71M グループ用 RX Driver Package FIT モジュールー覧 (O:サポート、-: 未サポート)

種類	モジュール名	FIT モジュール名	RX64M	RX71M	Rev.
Board Support	ボードサポートパッケージ(BSP)	r_bsp	0	0	3.00
Package					
Device Driver	割り込みコントローラ(IRQ)	r_irq_rx	0	0	1.70
Device Driver	データトランスファコントローラ(DTC)	r_dtc_rx	0	0	2.03
Device Driver	DMA コントローラ(DMAC)	r_dmaca_rx	0	0	1.03
Device Driver	I/O ポート(GPIO)	r_gpio_rx	0	0	1.70
Device Driver	マルチファンクションピンコントローラ(MPC)	r_mpc_rx	0	0	1.70
Device Driver	コンペアマッチタイマ(CMT)	r_cmt_rx	0	0	2.60
Device Driver	コンペアマッチタイマ W(CMTW)	r_cmtw_rx	0	0	1.10
Device Driver	リアルタイムクロック(RTC)	r_rtc_rx	0	0	2.30
Device Driver	シリアルコミュニケーションインタフェース (SCI : 調歩同期式/クロック同期式)	r_sci_rx	0	0	1.70
Device Driver	FIFO 内蔵シリアルコミュニケーションインタ フェース(SCIFA : 調歩同期式/クロック同期式)	r_scif_rx	0	0	1.10
Device Driver	FIFO 内蔵シリアルコミュニケーションインタ フェース(SCIFA:シリアルメモリ制御用デバイ スドライバ)	r_scifa_smstr_rx	0	0	1.08
Device Driver	シリアルコミュニケーションインタフェース (SCI : 簡易 I ² C バス)	r_sci_iic_rx	0	0	1.70
Device Driver	I ^² C バスインタフェース(RIIC)	r_riic_rx	0	0	1.70
Device Driver	シリアルペリフェラルインタフェース	r_rspi_rx	0	0	1.30
Device Driver	シリアルペリフェラルインタフェース(RSPI:シ リアルメモリ制御用デバイスドライバ)	r_rspi_smstr_rx	0	0	1.09
Device Driver	クワッドシリアルペリフェラルインタフェース (QSPI: シリアルメモリ制御用デバイスドライバ)	r_qspi_smstr_rx	0	0	1.08
Device Driver	USB Basic Firmware	r_usb_basic	0	0	1.10
Device Driver	USB Host Mass Storage Class	r_usb_hmsc	0	0	1.10
Device Driver	USB Host Communication Device Class	r_usb_hcdc	0	0	1.10
Device Driver	USB Host Human Interface Device Class	r_usb_hhid	0	0	1.10
Device Driver	USB Peripheral Mass Storage Class	r_usb_pmsc	0	0	1.10
Device Driver	USB Peripheral Communications Device Class	r_usb_pcdc	0	0	1.10
Device Driver	イーサネットコントローラ用 PTP コント ローラ(EPTPC)	r_ptp_rx	0	0	1.11
Device Driver	EPTPC Light モジュール	r_ptp_light_rx	0	0	1.10
Device Driver	イーサネットコントローラ(ETHERC)	r_ether_rx	0	0	1.02
Device Driver	パラレルデータキャプチャユニット(PDC)	r_pdc_rx	0	0	1.02
Device Driver	12 ビット A/D コンバータ(S12AD)	r_s12ad_rx	0	0	2.10
Device Driver	D/A コンバータ(DAC)	r_dac_rx	0	0	2.50
Device Driver	フラッシュメモリ(Flash API)	r_flash_rx	0	0	1.30
Device Driver	サンプリングレートコンバータ(SRC)	r_src_api_rx	0	0	1.11
Device Driver	シリアルサウンドインタフェース(SSI)	r_ssi_api_rx	0	0	1.20
Device Driver	Byte Queue Buffer(データ管理)	r_byteq	0	0	1.50
Device Driver	Long Queue Buffer(データ管理)	r_longq	0	0	1.50
Middleware	組み込み用 M3S-T4-Tiny モジュール【注 1】	r_t4_rx	0	0	2.02
Middleware	Ethernet ドライバと組み込み用 TCP/IP	r_t4_driver_rx	0	0	1.02



	M3S-T4-Tiny のインタフェース変換モジュール				
Middleware	組み込み用 TCP/IP M3S-T4-Tiny ソケット API モ	r_socket_rx	0	0	1.22
	ジュール				
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた DHCP	r_t4_dhcp_client_rx	0	0	1.03
	クライアントモジュール		<u> </u>		
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた DNS	r_t4_dns_client_rx	0	0	1.02
	クライアントモジュール				
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた FTP	r_t4_ftp_server_rx	0	0	1.03
	サーバモジュール				
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた Web	r_t4_http_server_rx	0	0	1.04
	サーバモジュール				
Middleware	FTP/Web サーバ用ファイルドライバモジュール	r_t4_file_driver_rx	0	0	1.01
Middleware	音声録音・再生システム(独自 ADPCM コーデッ	r_s2_rx	0	0	3.03
	ク) M3S-S2-Tiny モジュール				
Middleware	オープンソース FAT ファイルシステム	r_tfat_rx	0	0	3.02
	M3S-TFAT-Tiny モジュール				
Middleware	M3S-TFAT-Tiny メモリドライバインタフェース	r_tfat_driver_rx	0	0	1.02
	モジュール				
Middleware	EEPROM アクセス クロック同期制御モジュー	r_eeprom_spi	0	0	2.32
	ル				
Middleware	Serial Flash memory アクセス クロック同期制	r_flash_spi	0	0	2.33
	御モジュール				

【注 1】本パッケージには、評価版の「M3S-T4-Tiny(TCP/IP プロトコルスタックライブラリ)」が含まれて います。製品版については、以下の URL を参照してください。

http://japan.renesas.com/mw/t4



4. 使用方法

RX Driver Package は、e² studio に入っている FIT プラグインを使用することで、簡単にプログラムを構築 することができます。以下に、e² studio を使用した簡単な使用例を示します。なお、CS+を使用する場合につ いては、本パッケージに付属している「RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826JJ)」を参照してください。

4.1 説明で使用する環境

ターゲット MCU に「RX64M」、ターゲットボードに「Renesas Starter Kit+ RX64M」を使用します。それ 以外の環境で行う場合は、使用する環境に合わせて説明を読み替えてください。

4.2 e² studio に RX Driver Package をインストールする

RX Driver Package に入っている FIT モジュールを、 e^2 studio にインストールします。

- 1. ダウンロードした「an_r01an2606jj****_rx.zip」ファイルを、任意のフォルダに解凍します。
- 2. 解凍してできたフォルダを開き、その中にある「FITModules」フォルダを開きます。
- 3. 「FITModules」フォルダ内ある全てのファイルを選択し、「編集(<u>E</u>)」メニューから「コピー」をクリックします。





- 4. e² studio のインストールフォルダ(通常は、C:¥Renesas¥e2_studio です)を開き、その中にある「FITModules」 フォルダを開きます。
- 5. 「編集(E)」メニューから「貼り付け」をクリックします。 e² studio の「FITModules」フォルダに、FIT モジュールがコピーされます。

→ ・ コンピューター ・ ローカル ディスク (C:) ・ Renesas	e2_studio FITModules		▼ ◆ FITModulesの検索	Q	「FIIModules」フォルタ
ファイル(F) 編集(E) 表示(V) ツール(T) ヘルプ(H)					―― 問え 「炉隹(Г)」ノー」
整理 ▼ ライブラリに追加 ▼ 共有 ▼ 新しいフォルダー			88 •		用こ、「禰未(こ)」 ノーユ
	Chap, v2, 75, and Chap, v2, v2, v2, v2, v2, v2, v2, v2, v2, v2	() ((k), (k), (k), (k), (k), (k), (k	C. J. S. Land, J. Hanser, Hanser, J. Hanser, J. Hanser, J. Hanser, J. Hanser, J. Hans		から「貼り付け」をクリ クし、このフォルダへ: ピーします

4.3 アプリケーションの作成

LED を光らせる、簡単なプログラムを作成してみます。

4.3.1 ワークスペースとプロジェクトを作成する

まず、ワークスペースとプロジェクトを新規に作成します。

- 1. e² studio を起動します。
- 2. 表示されたダイアログに、任意のワークスペースフォルダを入力し、「OK」をクリックします。





3. 以下の画面が表示されたら、「Workbench」をクリックします。



4. ワークベンチが起動したら、「ファイル(F)」メニューの「新規(N)」の「C Project」をクリックします。





5. プロジェクト名(P)を入力します。「プロジェクトの種類:」は、「Executable (Renesas)」の「Sample Project」 をクリックします。「ツールチェイン:」は、「Renesas RXC Toolchain」をクリックします。設定が終わっ たら「次へ(N)」をクリックします。

e² C プロジェクト			
c プロジェクト			
選択したタイプの C プロジェクトを作成します			
			プロジェクトタ
プロジェクト名(P): led_sample			
			を入力します
ロケーション(L): C:¥sample_workspace¥led_sample		参照(<u>R</u>)	
Create Directory for Project			
プロジェクトの種類・	ツールチェイン		
A (= Everyltable (Renesas)	KRIT CNUARM-RZ-EARI Toolchain		
Sample Project	KPLL GNURL/B-ELE Toolchain		····· クリックします
🔺 👝 Static Library (Renesas)	KPIT GNURX-ELF Toolchain		
 Sample Project 	KPIT GNUSH-ELF Toolchain		
Debug-Only Project	Renesas RXC Toolchain		
Executable (IAR)	Renesas SHC Toolchain		
🕨 👝 static ライブラリー			
▷ 👝 共用ライブラリー			
▷ 👝 実行可能			
V800 Standalone Executable (Green Hills)			
V800 Standalone Static Library (Green Hills)			
V800 ThreadX Executable (Green Hills)			
▷ 🗁 Makefile プロジェクト			
回 ゴニットフィーノ 不井ず トゥかえ根合かれ ゴロジェクト・クノブト keelebai			
■ ノラットノオームビリホートされる場合のみ、ノロシェクト・タイノと toolchall	11 征表小		
(P) < 戻る(B) 次	(N) > 終了(F)	キャンセル	

6. ターゲットを選択します。「ターゲットの選択:」の「・・・」ボタンクリックし、「R5F564MLCxFC」を選 択します。設定が終わったら「次へ(<u>N</u>)」をクリックします。

ターゲット國有の設定の違択	
ツールチェーン・バージョン: v2.01.00 デバッグ・ハードウェア: E1 データ・エンディアン: Little-endian data ターゲットの選択: RSF56107VxFP 編成の選択: マリードウェアを使用してデバッグする ジンミュレーターを使用してデバッグする シミュレーターを使用してデバッグする ジシミュレーターを使用してデバッグする シミュレーターを使用してデバッグする リリース(デバッグしない) :: アン・デバッグする :	クリックして 「R5F564MLCxFC」 を選択します。
ビルド構成は選択したデバッグ・モード・オプションのプロジェクトにのみ作成されます。ただしデフォルトでは、そのプロジェクトはアクティブな構成(グ ループから選択された最初の構成)用にビルドされます。選択したデバイス (RX600)、デバッグ・ハードウェア (E1) 、およびデバッグ・ターゲット (RSF56107VxFP)をもとに、ユーザーのためのデバッグ構成が自動的に作成されます。	
(?) < 戻る(<u>B</u>) 次へ(<u>N</u>) > 終了(<u>E</u>) ギャンセル	クリックします

7. 「FIT モジュールを使用する」にチェックを入れて、「次へ(<u>N</u>)」をクリックします。

C تا ال	
e2 studio · プロジェクト生成 コード生成と FITの設定	
□ コード生成を使用する ■ FITモジュールを使用する ■ FITモジュールのダウンロード	チェックを入れます
	, _ , , 2, (100)
e2 studio コード生成は GUI で設定することにより、自動的に MCU 周辺機能 (クロック、タイマー、シリアル・インターフェース、 A/D コンパーター、 DMA コントローラーなど)のプログラム(デバイス・ドライバー)を生成します。 機能は、 API として提供され、周辺機能の初期化に制限を加えることはありません。 従来、「マイコン初期設定」「ターゲットボードの定美方法」「ファイル構成」「観致名」「コーザアプリケーションとのインタフェースの共通化」などの情報が サンプルコードごとに異なっている場合が多く、ユーザアプリケーションへ組み込む際にサンプルコードの変更が必要でした。 Firmware Integration Technology (以下FIT)では、これら情報のルール化により、ユーザアプリケーションへ名サンプルコードを容易に組み込むことを可能にし ました。また、FITIに対応した周辺機能ドライバやミドルウェアは、ユーザアプリケーションとのインタフェースが共通化されているため、現在使用中のRXマイコ ンから別のRXマイコンへ移行する場合もユーザアプリケーションを容易に移植することができます。	
User Application	
Middleware	
CG RTOS HI	
Device Driver (BSP)	
MCU	
(?) (N) > 検ア(F) キャンセル	―― クリックします

8. 何もせず「次へ(<u>N</u>)」をクリックします。

udio - ノロシェクト生成 1 CPU オプションの選択			
追加 CPU オプションの選択:			
丸めモード:	Nearest	•	
double 型の精度:	Single precision		
Char 型 の符号:	unsigned		
ビットフィールド型の符号:	unsigned		
ビットフィールド型の並び順:	Lower bit		
関数の分岐幅:	24 Bit		
グローバレ・オブションの選択 非正規数を非正規数とし int 型を short 型に置換 一列挙型サイズを自動選択 構造体メンバーの境界録 C++の try, throw, cat C++の dynamic_cast	: て扱う する する 整数を1とする ch を使用する と typeid を使用する		

9. 何もせず「次へ(<u>N</u>)」をクリックします。

e² C プロジェクト			
e2 studio - プロジェクト グローバル・オプション設定	成		
パッチ・コード生成	None	•	
高速割込み専用レジスター:	None		
ROM:	None	•	
RAM:	None		
アドレス (H'):	0000000		
アドレス・レジスター:	None	•	
(?)	< 戻る(B) 次へ(N) > 終了(E)	キャンセル	ーーー クリックします

10. 何もせず「終了(F)」をクリックします。

e2 studio - プロジェクト生成 ほ称へッダー・ファイル ライブラリー構成: C(C99) ヘッダー・ファイルの通知: ヘッダー・ファイルの通知: ヘッダー・ファイルの通知: の のの上の の のの上のの の のの上のの の のの上のの の ののしたの の ののしたののののしたの の ののしたのののしたの の のしたの の ののしたの の ののしたの の ののしたの の ののしたの の ののしたの の ののしたの の のしたの の のしたの の のしたの の のしたの の のしたの の のしたの	e² C プロジェクト		
歴年へタイ・ファイル ウイブラリー概: (C(39) ヘッダー・ファイルの避形 - ロ・加れ防m : ジングリール・レーテン(デフォルトではチェック後島効にされます) ・・レナットカ: ・シナットが用ライブラリー 回 math.h :: ジングリーク・レーテン(デフォルトではチェック後島効にされます) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(3) ・・ロック(5) ・・ロック(3) ・・ロック(5) ・・ロック(3) ・・ロック(5) ・・ロック(3) ・・ロック(5) ・・ロック(3) ・・ロック(5) ・・ロック(4) ・・ロック(5) ・ロック(4)	e2 studio - プロジェク	ト生成 ————————————————————————————————————	>
ライブラリー構成: (CC99) ヘッダー・ファイルの選択: 「runtime : ランタイム・ルーチン (デフォルトではチェック後島効にされます) 一、たわれ : 文学操作用ライブラリー 一、mわれ : 茨磁計算用ライブラリー 一、mわれ : 茨磁計算用ライブラリー ● おtdarg.h : 可変価の(実引数アクセス用ライブラリー ● Stdiol.h : ス出カ用ライブラリー ● Stdiol.h : スピン用用ライブラリー ● Stdiol.h : スピン用用ライブラリー ● Stdiol.h : スピン用用ライブラリー ● Stdiol.h : スピン用のラスライブラリー ● Stdiol.h : ジャ列線作用ライブラリー ● Stdiol.h : ジェジ列線作用ライブラリー ● Stdiol.h : ジェジジョン ● Stdiol.h : ジェジジョン ● Stdiol.h : ジェジジョン ● Stdiol.h : ジェジジョン ● Stdiol.h : ジェジョン ● Stdiol.h : ジェジョン ● Stdiol.(C99) : ジェジュシュシー ● Stdiol.(C99) : ジェジュシュシー ● Stdiol.(C99) : ジェジュション ● Stdiol.(S09) : ジ	標準ヘッダー・ファイル		
ウイブラリー構成: (C(59) * ヘッダー・ファイルの選択: * ・ nutime : ランタイム・ルーチン (デフォルトではチェック後無効にされます) ・ Ctype.h :: 文学規準作用ライブラリー ・ math.h : 欲想!算用ライブラリー ・ math.h : 欲想!算用ライブラリー ・ stdarg.h : 可定母の実列数アクセス用ライブラリー ・ Stdin.h : 出数力用ライブラリー ・ Stdin.h : 出数均用ライブラリー ・ Stdin.h : 出数均用ラブブラリー ・ Stdin.h : 出数クライブラリー ・ Stdin.h : 出数のラブラリー ・ Stdin.h : 生り過(相声・ブラリー ・ Complex.(EC++) : 大きり通信用ライブラリー ・ Complex.(EC++) : 生り過(相手・ブラリー ・ Complex.(FC9) : 提表数計算のブラリー ・ Complex.(C9) : 要参判認識問題に ・ Inttypes.h(C9) : アイド文字入出力ライブラリー ・ wchach(C9) : アイド文字入出力ライブラリー ・ アイド文字文法はカライブラリー : 要が認識に ・ Inttype.h(C9) : ワイド文字支換員・ブラリー ・ Woth.perf. : アイド文字支換員・ブラリー ・ Moth.perf. : アイド文字支換員・ブラリー			
マイブラリー構成: (2039) マ ハッダー・ファイルの選択: 「nuttime : ランタイム・ルーチン (デフォルトではチェック後無効にされます) こくたりを.h : 文字操作用ライブラリー math.h : 飲息甘賀用ライブラリー math.h : 飲息甘賀用ライブラリー ** <tr< th=""><th></th><th></th><th></th></tr<>			
ヘッダー・ファイルの選択: 「runtime ジンタイム・ルーチン (デフォルトではチェック後無効にされます) □ ctype.h : 文字操作用ライブラリー math.h : 数砲計算用ライブラリー math.h : 数砲計算用ライブラリー Stdarg.h : 可変個の実別数アクセス用ライブラリー ダ tdio.h : 入出力用ライブラリー ダ stdio.h : 入出力用ライブラリー ダ string.h : 文字列操作用ライブラリー Ios(EC++) : 入出力ストリーム用クラスライブラリー Ios(EC++) : 大出力場中ラスライブラリー Complex(EC++) : 水生列操作用ライブラリー String(EC++) : 水生列操作用ライブラリー Complex(FC++) : 水生列操作用ライブラリー Inttypes.h(C99) : ジャジ別操作用ライブラリー wcharh(C99) : ジャジ判操作用ライブラリー wcharh(C99) : ジャジリ操作用ライブラリー Inttypes.h(C99) : ジャジリ操作用ライブラリー wcharh(C99) : ジャジリ操作用ライブラリー wcharh(C99) : ジャジリ操作用ライブラリー Inttypes.h(C99) : ジャジリ操作用ライブラリー マイドシャンサリ操作用ライブラリー : ジャジリ操作用 ジェント マントレ(C99) : ジャント マント(C99) : ジャント マント(C99) : ジャント マント(C99) : ワント ダント (ジャント ダント : ワント	ライノラリー構成: C((
Puntime ミシタ4ム・ルーチン(デフォルトではチェック後無効にされます) Ctype.h :文字操作用ライブラリー math.h : 数値計算用ライブラリー(float 型眼数) extors : 数回が算用ライブラリー(float 型眼数) extors : 数目が算用ライブラリー ? stdio.h : 入出力用ライブラリー ? stdio.h : 入出力用ライブラリー ? stdio.h : 入出力用ライブラリー ? string.h : 文字列操作用ライブラリー ? string.h : 文字列操作用ライブラリー ? los(EC++) : 入出力ストリーム用クラスライブラリー ? complex.h(C9+) : 提案数計算用イブラリー Complex.h(C9+) : 提案数計算コイブラリー Complex.h(C9) : 提載数計算コイブラリー [nttypes.h(C99) : 逆数型電式支換 [math.h(C99) : 「参助型素支換 [wttype.h(C99) : 「空水正義大力 (wttype.h(C99) : 「空水正義大力 (wttype.h(C99) : 「空水正義大力 (wttype.h(C99) : 「フィド文字支換ライブラリー (wttype.h(C99) : 「フィド文字支換ライブラリー (wttype.h(C99) : 「フィド文字支換ライブラリー (wttype.h(C99) : 「フィド文字支換ライブラリー	ヘッダー・ファイルの	選択:	
はtype.h : 文学操作用ライブラリー math.h : 教健計算用ライブラリー math.h : おび温の実別数アクセス用ライブラリー マまdio.h : 入出力用ライブラリー マまdio.h : 入出力用ライブラリー マまdio.h : 入出力用ライブラリー マまdio.h : 人出力トリーム用クラスライブラリー マまting.h : 文学列操作用ライブラリー このs(EC++) : 大きリ操作用ライブラリー complex.(EC++) : 大きリ操作用ライブラリー complex.(C39) : 複集数計算用クラスライブラリー complex.(C39) : 複集数計算のブラリー complex.(C39) : 複集数計算のブラリー complex.(C39) : 複数型計算のブラリー complex.h(C39) : ジェジ列操作用ライブラリー wcharh(C39) : ジャ列操作用ライブラリー wchwarb.(C39) : ジェジ列操作用ライブラリー complex.h(C39) : ジェジ列操作用ライブラリー complex.h(C39) : ジェジア操作用ライブラリー wcharb.(C39) : ジェジア操作用ライブラリー wcharb.(C39) : ジェジア操作用ライブラリー wcharb.(C39) : ジェジャット wcharb.(C39) : ジェジェシット マイド文字支索換ライブラリー ************************************	✓ runtime	:ランタイム・ルーチン (デフォルトではチェック後無効にされます)	
math.h : 数吨計算用ライブラリー(foot 型関数) mathf.h : 数位計算用ライブラリー(foot 型関数) stdarg.h : 可空偏の突引数アクセス用ライブラリー ダ stdio.h : 入出力用ライブラリー 9 string.h : 文字列操作用ライブラリー 0 st(EC++) : 入出力ストリーム用クラスライブラリー 0 st(EC++) : スレンガ用クラスライブラリー 0 complex(EC++) : マンガ化オーライブラリー 0 complex(EC++) : マンガ化オーライブラリー 0 complex(EC++) : マンガリカノブラリー 0 complex(EC++) : マンガリオフラリー 0 complex(EC++) : マンガリカノブラリー 0 complex(EC++) : マンガリカノブラリー 0 complex(EC++) : マンガリホリガラリー 0 complex(EC++) : マンガリカリブラリー 0 complex(EC++) : マンガリホリカラノブラリー 0 complex(EC++) : マンガリカリカリン 0 complex(EC++) : マンガリカリン 0 complex(EC++) : マンガリカリカリン 0 complex(EC++) : マンガリカリン 0 complex(EC++) : マンガリン 1 fttypes.h(C99) : ジンドシッカリン 1 ftypes.h(C99) : ワンド文字交換ライブラリー 0 woharh(C99) : ワンド文字交換力ブリン 1 wotype.h(C99) : ワンド文字交換力ブリン	Ctype.h	:文字操作用ライブラリー	
□ mathf.h : 数値計算用ライブラリー(float 聖観数) □ stdarg.h : 可変個の実引数アクセス用ライブラリー ジ stdio.h : 入出力用ライブラリー ジ stdio.h : 【進処理用ライブラリー ジ stdio.h : 文字列操作用ライブラリー ② stdio.h : 文字列操作用ライブラリー ◎ los(EC++) : ス出力ストリーム用クラスライブラリー ◎ new(EC++) : メモリ操作用ライブラリー ○ complex(EC++) : 文字列操作用ライブラリー ○ complex(EC++) : 支字列操作用ライブラリー ○ complex(EC++) : 支字列操作用ライブラリー ○ complex(EC++) : 支字列操作用ライブラリー ○ complex(EC++) : 支字列操作目ライブラリー ○ complex(FC++) : 支字列操作用ライブラリー ○ complex(EC++) : 支字列操作目ライブラリー ○ complex.h(C99) : ジャ列操作目ライブラリー ● ferv.h(C99) : ジャ列操作目ライブラリー ◎ nttype.h(C99) : ワイド文字支法力ラブラリー ● wchach(C99) : ワイド文字支法人力ラブラリー ● wctype.h(C9) : ワイド文字支援員ライブラリー ● mother : ゴイド文字支援員会 ● mother : ゴイド文字支援員会 ● mother : ゴイド文字支援員会 ● mother : ゴイド文字支援員会 ● mother : ゴイド支援員会	math.h	:数値計算用ライブラリー	
stdarg.h : 可変個の実利数アクセス用ライブラリー ジ stdilo.h : 入出力用ライブラリー ジ stdilo.h : 人能力用ライブラリー ジ stdilo.h : 大学列操作用ライブラリー ご oS(EC++) : スキブリダ作用ライブラリー ○ new(EC++) : メモリ操作用ライブラリー ○ complex(EC++) : 文夫列操作用ライブラリー ○ complex(EC++) : 文夫列操作用ライブラリー ○ complex(EC++) : 支来列操作用ライブラリー ○ complex(CO9) : ジャジル点の環境設定 ○ complex(CO9) : ジャジン 入山カライブラリー ○ complex(CO9) : ジャジン 入山カライブラリー ○ mother : ワイド文字支払カライブラリー ○ wothype.h(CO9) : ワイド文字支換ライブラリー ○ wothype.h(CO9) : ワイド文字支換ライブラリー ○ アイド文字支換ライブラリー : 国北ですべて解除	mathf.h	:数値計算用ライブラリー(float 型関数)	
図 stdio.h : 入出力用ライブラリー ② stdib.h : 標準処理用ライブラリー ③ string.h : 文字列操作用ライブラリー □ los(EC++) : 入出力ストリーム用クラスライブラリー ○ complex.(EC++) : メモリ操作用ライブラリー □ complex.(EC++) : 文字列操作用ライブラリー □ complex.h(C99) : 支穿列操作用ライブラリー □ complex.h(C99) : 支穿列操作用ライブラリー □ complex.h(C99) : 支穿列操作用ライブラリー □ complex.h(C99) : 支穿列換定 □ fenv.h(C99) : ジョン(大文字交換) □ mtypes.h(C99) : ジョン(アド文字文) □ wctype.h(C99) : ワイド文字交換) □ wctype.h(C99) : ワイド文字交換) □ wctype.h(C99) : ワイド文字交換)	🔲 stdarg.h	:可変個の実引数アクセス用ライブラリー	
図 string.h : 標準処理用ライブラリー 図 string.h : 文字列操作用ライブラリー los(EC++) : 入出力入トリーム用クラスライブラリー Ø new(EC++) : 桜東数計算カイブラリー Complex(EC++) : 桜東数計算ライブラリー @ string(EC++) : 文字列操作用ライブラリー @ complex.h(C99) : 文字列操作用ライブラリー @ fenv.h(C99) : 探動小数点数環境設定 [] Inttypes.h(C99) : 受が文字文出カライブラリー wchark(C99) : フイド文字交換ライブラリー wctype.h(C99) : ワイド文字変換ライブラリー	📝 stdio.h	:入出力用ライブラリー	
図 string.h : 文字列操作用ライブラリー [los[EC++) : 入出力入トリーム用クラスライブラリー ⑦ new(EC++) : メモリ操作用ライブラリー ○ complex(EC++) : 又字列操作用ライブラリー □ string(EC++) : 文字列操作用ライブラリー □ complex.h(C99) : 援奏数計算ライブラリー □ ferv.h(C99) : 援数型書式変換 □ wtharh(C99) : ジンド文字入出力ライブラリー □ wthype.h(C99) : ジンド文字交換ライブラリー ■ wthype.h(C99) : ジンド文字変換ライブラリー	🔽 stdlib.h	:標準処理用ライブラリー	
 □ ios(EC++) : 入出力ストリーム用クラスライブラリー ▽ new(EC++) : メモリ操作用ライブラリー □ complex(EC++) : 技業教践計算カクラスライブラリー □ string(EC++) : 技業教員性育ライブラリー □ ferv.h(C99) : 技業教員性育ライブラリー □ ferv.h(C99) : 逆数型書式変換 □ wchach(C99) : ワイド文字交換ライブラリー □ wctype.h(C99) : ワイド文字変換ライブラリー ▼ オペて選択 	🔽 string.h	:文字列操作用ライブラリー	
マmw(EC++) : メモリ操作用ライブラリー Complex(EC++) : 支字列操作用ライブラリー string(EC++) : 文字列操作用ライブラリー Complex.h(C99) : 浸海が物点政環境設定 Inttypes.h(C99) : ジロイド文字入出カライブラリー wchar.h(C99) : ワイド文字入出カライブラリー wctype.h(C99) : ワイド文字支換ライブラリー Type.h(C99) : ワイド文字支換ライブラリー Wctype.h(C99) : ワイド文字支換ライブラリー	ios(EC++)	: 入出力ストリーム用クラスライブラリー	
complex(EC++):接載数計算用クラスライブラリー string(EC++):文字列操作用ライブラリー complex.h(C99):接載数計算ライブラリー ferv.h(C99):採動N数点数環境設定 inttypes.h(C99): 空数型書式変換 wchar.h(C99): ワイド文字入出力ライブラリー wchar.h(C99): ワイド文字変換ライブラリー	V new(EC++)	:メモリ操作用ライブラリー	
・ string(EC++) : 文字列操作用ライブラリー ・ complex.h(C99) : 浸蟲数計算ライブラリー ・ fenv.h(C99) : 浮動小数点数環境設定 ・ inttypes.h(C99) : 整数型書式変換 ・ wchar.h(C99) : ワイド文字入出力ライブラリー ・ wctype.h(C99) : ワイド文字変換ライブラリー	complex(EC++)	:複素数計算用クラスライブラリー	
© complex.h(C99) : 接乗数計算ライブラリー ● fenv.h(C99) : 浮動小数点数環境設定 □ Inttypes.h(C99) : 整数型書式変換 ◎ wchar.h(C99) : ワイド文字文出力ライブラリー ◎ wctype.h(C99) : ワイド文字変換ライブラリー ● wctype.h(C99) : ワイド文字変換ライブラリー	string(EC++)	:文字列操作用ライブラリー	
● ferv.h(C99) : 浮動小数点散環境稳定 ● inttypes.h(C99) : 整数型書式変換 ● wcharh(C99) : ワイド文字入出力ライブラリー ● wctype.h(C99) : ワイド文字変換ライブラリー ● wctype.h(C99) : ワイド文字変換ライブラリー	complex.h(C99)	:複素数計算ライブラリー	
Inttypes.h(C99) : ワイド文字入出力ライブラリー wctype.h(C99) : ワイド文字変換ライブラリー すべて違択 運択をすべて解除	fenv.h(C99)	: 浮動小数点数環境設定	
 □ wcharh(C99) : ワイド文字支入出力ライブラリー □ wctype.h(C99) : ワイド文字変換ライブラリー すべて違択 運択をすべて解除 	inttypes.h(C99)	:整数型書式変換	
□ wctype.h(C99) :ワイド文字変換ライブラリー すべて違択 運択をすべて解除	wchar.h(C99)	: ワイド文字入出力ライブラリー	
すべて道択 通択をすべて解除	wctype.h(C99)	: ワイド文字変換ライブラリー	
		ずべて選択	
⑦ < 戻る(B) 次へ(N) > 検T(F) キャンセル クリックしま	?	< 戻る(B) 次へ(N) > 終了(F) キャンセル	クリックします

11.「OK」をクリックすると、プロジェクトが生成されます。

3	王 約
	プロジェクトの要約:
	プロジェクト・ジェネレーター 🔺
	プロジェクト名: led_sample
	プロジェクト・ディレクトリー: D:¥WorkSpace¥
	CPU シリーズ: RX600
	CPU 型名: RX64M
	ツールチェーン名: Renesas RXC Toolchain
	ツールチェーン・バージョン: v2.03.00
	生成ファイル:
	D:¥WorkSpace¥e2studio¥FITModules_Test¥20150930_ Main Program
	< III • • • • • • • • • • • • • • • • •
	'OK' をクリックしてプロジェクトを生成するか、 'キャンセル'
	をクリックして中止してください。
	OK キャンセル

4.3.2 FIT プラグインで FIT モジュールをインストールする

作成したプロジェクトに、FIT プラグインを使って必要なモジュールをインストールします。

ここでは、コンペアマッチタイマドライバ(r_cmt_rx)をインストールします。

1. 「Renesas Views」メニューの「e2 ソリューション・ツールキット」の「FIT Configurator」をクリックしま

e ² C - e2 studio			
ファイル(F) 編集(E) ソース(S) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(P)	Renesas Views 実行(R) ウィンドウ(V	W) ヘルプ(H)	
📑 🗝 📰 🕼 🕒 🛞 🖛 🗞 🖛 🖬 💽 🥞 🥒 🚳 🕈 🚳 🕶 🗳 🕶 🚱 🔹 🚱 🔹 🔅	C/C++(C)	$\bullet \rightarrow \Leftrightarrow \leftarrow \bullet \Rightarrow \bullet \bullet$	
	e2 ソリューション・ツールキット	FIT Configurator	クリック
	コード生成	 最適化アシスタント 	1 = 7
¹	デバッグ	> 🐳 スマート・ブラウザー	649
	パートナーOS	・ 、 スマート・マニュアル	
Ied_sample [HardwareDebug]	ルネサス OS	 るアウトラインはありませ 	
Includes		100	
⊿ 🔑 src			
▷ 🖻 led_sample.c			
🗟 custom.bat			
📄 led_sample Debug.launch			
📄 led_sample HardwareDebug.lat			

2. 画面右下に「FIT Configurator」が表示されます。

「FIT モジュールを追加するプロジェクト」で作成したプロジェクトを選択します。

次に「ターゲットボード」から「RSKRX64M」を選択します。

次に「使用可能なモジュール」から「r_cmt_rx」をクリックし、「モジュールの追加>>」をクリックし ます。



「コンソール」が表示されます。
 「r_cmt_rx」と依存関係にある「r_bsp」も同時に追加されます。(※)
 再度「FIT Configurator」をクリックします。

※:追加する FIT ドライバは、適応可能な「r_bsp」のバージョンを指定します。FIT ドライバのリリース タイミングにより、指定した「r_bsp」のバージョンと追加された「r_bsp」のバージョンが不一致の場合、 コンソール画面に Warning (W0000001) が出力されます。追加された「r_bsp」のバージョンが、指定し た「r_bsp」のバージョン以降の場合、バージョンの新しい「r_bsp」は後方互換を持つため、Warning を無 視することができます。

🖹 問題 🧔 タスク 🖳 コンソール 🛛 🔲 プロパティー 🔋 🛚	emory Usage 🐚 スタック解析 🛞 スマート・ブラウザ	- 灯 *FIT Configurator		_クリック
FITコンフィグレータコンソール M0000001: 以下の依存モジュールが追加されました。 r cmt rx: r bsp version(s) 2.90 M0000001: 以下の依存関係のパージョンが一致していません。 r_cmt_rx: r_bsp version(s) 2.90	 CMT ドライバが BSP via る例です。 追加された r_bsp が ver ンの場合、後方互換のた 無視することができます 	er.2.90 を指定して .2.9 以降のバーシ こめ、この Warnin	■ - 🖻 - こい 3 = g を	します

 「選択したモジュール」に「r_bsp」と「r_cmt_rx」が追加されていることを確認します。尚「r_cmt_rx」 に表示されている [●]は上記 3.で説明した Warning の発生を指し示すものです。無視して構いません。 次に、「コードの生成」をクリックします。

モジュール パージ 概要 ビジュール パージ r_bsp 3.00 Board Support Packages. r_bsp 3.00 r_byteq 1.30 Byte-based circular buffer library. r_bsp 3.00 r_cmtw_rx 1.10 CMTW Driver 0 r_cmt_rx 2.60 r_dac_rx 2.30 Digital-to-Analog Converter driver - - r_dtc_rx 2.03 DTC driver - -	t Packages. ircular buffer library. driver for creating timer tick. alog Converter driver r solution Codel FFFDDOM analysis for Decision Pacification Codel FFFDDOM analysis for
r_bsp 3.00 Board Support Packages. r_bsp 3.00 r_byteq 1.30 Byte-based circular buffer library. 2.60 or_cmt_nx 2.60 r_cmt_nx 2.60 Simple CMT driver for creating timer tick. - - - r_dac_nx 2.30 Digital-to-Analog Converter driver - - - - r_ddac_nx 2.03 DTC driver DTC driver - - -	t Packages. ircular buffer library. ircular buffer library. ircutar for creating timer tick. alog Converter driver r converter driver r converter driver converter
r_byteq 1.30 Byte-based circular buffer library.	ircular buffer library.
r_cmtw_rx 1.10 CMTW Driver r_cmt_rx 2.60 Simple CMT driver for creating timer tick. r_dac_rx 2.30 Digital-to-Analog Converter driver r_dmaca_rx 1.03 DMACA driver r_dtc_rx 2.03 DTC driver	Ariver for creating timer tick.
r_cmt_rx 2.60 Simple CMT driver for creating timer tick. r_dac_rx 2.30 Digital-to-Analog Converter driver r_dmaca_rx 1.03 DMACA driver r_dtc_rx 2.03 DTC driver	driver for creating timer tick. alog Converter driver r
dac_nx 2.30 Digital-to-Analog Converter driver dmaca_nx 1.03 DMACA driver dtc_nx 2.03 DTC driver	alog Converter driver r
r_dmaca_rx 1.03 DMACA driver	r /
r_dtc_rx 2.03 DTC driver	/20
	(25
	<<モジュールの削除
とジュールの追加>>	
+/// 体存エジューリット hon パージョン・2:00	
4004±2771±17710SD71=27317730	



5 そのまま「OK」をクリックします。





4.3.3 LED 点灯プログラムを作成する

コンペアマッチタイマを使用し、0.5 秒間隔で LED0 を点滅させるプログラムを作成します。

src/(プロジェクト名).cを開き、以下のように修正します。

【src/(プロジェクト名).c】

```
#include "platform.h"
#include "r cmt rx if.h"
/* LED Currently status */
uint32_t ledstatus = LED_OFF;
void call_back(void *pdata)
{
  if (ledstatus == LED_OFF)
  {
    /* Turn ON the LED0 If the status is LED_OFF */
   LED0 = LED ON;
   ledstatus = LED_ON;
  }
  else
  {
    /* Turn OFF the LED0 If the status is LED_ON */
   LED0 = LED_OFF;
   ledstatus = LED_OFF;
  }
}
void main(void)
Ş
  uint32_t cmt_ch;
  /* LED0 off */
  LED0 =LED_OFF;
  /* Create of 0.5 second(2Hz) cyclic timer. */
  R_CMT_CreatePeriodic(2, &call_back, &cmt_ch);
  while(1);
}
```

4.3.4 プログラムをビルドし動作を確認する

作成したプログラムをビルドして、動作を確認します。

1. 「プロジェクト(<u>P</u>)」メニューの「プロジェクトをビルド(B)」をクリックします。



2. ビルドが完了すると、「コンソール」ビューに以下のように表示されます。



3. 「実行(<u>R</u>)」メニューの「デバッグ構成(B)...」をクリックします。





4. 画面左側「Renesas GDB Hardware Debugging」の [▲] をクリックし、「(プロジェクト名) HardwareDebug」 をクリックします。

「Debugger」タブをクリックし、「Connection Setting」タブをクリックします。 「EXTAL 周波数」を「24.0000」に修正し、「エミュレータから電源を供給する」を「いいえ(※)」に 変更します。 完了したら「デバッグ(<u>D</u>)」をクリックします。

※:外部電源を使用する場合の設定です。エミュレータから電源を供給する場合は「はい」を選択してください。

e ² デバッグ構成			×	
構成の作成、管理、および実	行		The second	
「 ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・	名前(N): led_sample HardwareDebug シイン 参 Debugger Startup キッソース Debug hardware: E1 Targe	□ 共通(C) at Device: R5F564ML		クリックします
 Construction of C++ Educion Renesas GDB Hardware At Renesas GDB Hardware Di Renesas Simulator Debugç 	GDB Setting Connection Settings デバック メイン・クロック・ソース EXTAL 周波数[MHz] 内部フラッシュメモリー書き換え時にクロ ターケット・ボードとの接続 エミュレーター 接続タイプ JTag クロック周波数[MHz] Fine ボーレート[Mbps] ホット・プラグ	EXTAL [24.0000] ック・はい (Auto) JTag 16.5 2.00 いいえ		「24.0000」に修 正します
< III) フィルターー初・0 / 12 頂白	 ▲ 電源 エミュレーターから電源を供給する (MAX 供給電圧 ▲ CPU 動作モード レジスター設定 	200 <mark>m^{いい)え} 3.3V シングルチップ 違用(<u>Y</u>) 前回保管し</mark>		「いいえ」に変更 します
? ? ?		デバッグ(<u>D</u>)	閉じる	クリックします

5. 以下のメッセージが表示されたら、「はい(Y)」をクリックします。

● ² パース/	クティブ切り替えの確認	—
?	この種類の起動は、中断時に デバッグ パースペクティブが開くように構成されています。	
	このデバッグ・パースペクティブは、アプリケーション・デバッグをサポートするために設計されています。これには、デバッグ・スタック、変数、 よびブレークポイント管理を表示するビューが組み込まれています。	ŧ
	このパースペクティブを開きますか?	
常に	の設定を使用する(<u>R</u>)	
	(まい)(<u>)</u> いいえ(<u>N</u>)	



6. ロードモジュールのダウンロードが完了すると、「デバッグ」パースペクティブが開きます。

■デブ(ツグ・led_sample/r_bsp/board/rskn/64m/resetprg.c・e2 studio ファイル(E) 編集(E) ソース(E) リファクタリング(T) ナビゲート(N) 検索(A) プロジェクト(E) 実行(B) さ			
■▼□□□ ●▼ % ▼□ ● ■#2.◆↓#==.7. % ¹ # % % * * *			
黎 テバック ☆	(※)- 変数 🙁 💊 ブレ	ークボ 🔠 レジスター 🛋	モジュール 🦋 式 🥐 イベントボ 📄 IO Registers 🖳 🗖
C led_sample HardwareDebug [Renesas GDB Hardware Debugging]		5/7	
▲ 🔐 Thread [1] 1 (No thread info available) (Suspended : シヴナル : SIGTRAP:Trace/breakpoint tra	石刷	812	12
PowerON_Reset_PC() at resetprg.c:122 0xffc017e2			
gdb			
B GOD Server			
(III)			٨
<pre>118 * Once this initialization is complete, the user's main() 1 129 120 120 121 122 122 122 122 123 124 125 125 126 127 125 126 127 127 128 129 139 129 130 129 129 130 129 129 129 129 129 129 129 129 129 129</pre>) function is called. It should not return.		Lh_c,lib.h BSP_DECLARE_STACK platform.h # BSP_DECLARE_STACK platform.h # PSW_init # PSW_init # PSW_init # CLOSEALL(void) : void ftCLOSEALL(void) : void ftCLOSEALL(void) : void ftWarmstart_prec_function(void) ftWarmstart_prec_function(void)
■ コンソール ※ 必方スク 第 Renetas Coverage Memory Usage Performance Analysis Performance Analysi			
	1		中断中

7. ツールバーの「再開」をクリックします。プログラムが実行され、main 関数の先頭でブレークします。



8. main 関数の先頭でブレークした後に、もう一度ツールバーの「再開」をクリックします。 プログラムが実行され、LED0 が 0.5 秒間隔で点灯と消灯を繰り返します。

5. RX Driver Package Application について

5.1 RX Driver Package Application の構成

RX Driver Package Application は、RX Driver Package を簡単に使って頂くためのサンプルアプリケーション プログラムです。RX Driver Package Application には、RX Driver Package に入っているデバイスドライバやミ ドルウェアを使って動作するアプリケーションプログラムと、そのアプリケーションをビルドするためのプ ロジェクトファイルが入っているので、すぐに評価を開始することができます。



図 5-1 RX Driver Package Application の構成

RX Driver Package Application には、複数のドライバやミドルウェアを組み合わせて動作するシステムプロ グラムや、RX Driver Package に入っているモジュール単体の評価プログラムなど、さまざまな種類を順次公 開していく予定です。



図 5-2 RX Driver Package Application の種類

6. 補足

6.1 M3S-T4-Tiny(TCP/IP プロトコルスタックライブラリ)について

本パッケージには、評価版の「M3S-T4-Tiny(TCP/IP プロトコルスタックライブラリ)」が含まれていま す。製品版については、以下の URL を参照してください。

http://japan.renesas.com/mw/t4



ホームページとサポート窓口

ルネサス エレクトロニクスホームページ http://japan.renesas.com/

お問合せ先

http://japan.renesas.com/contact/

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。

改訂記録

			改訂内容
Rev.	発行日	ページ	ポイント
1.00	2014.09.01	—	初版発行
1.01	2015.01.05	_	既存モジュールを最新バージョンに更新。
			新規公開されたモジュールを追加。
1.02	2015.09.30	—	RX71M グループを追加し、既存モジュールを最新バージョンに
			更新。
			e2studio のバージョンを更新。併せて設定手順を修正。
1.03	2016.02.05	5	2.2 RX Driver Package の特徴 図 2-2 構築イメージ にて、
			「*1 Japanese edition only」を削除
		7	3.2 モジュール構成 図 3.2 RX64M, RX71M グループ用 RX
			Driver Package FIT モジュール構成イメージ にて、
			「*1 Japanese edition only」を削除
		8	3.3 FIT モジュールー覧 表 3-1 RX64M, RX71M グループ用 RX
			Driver Package FIT モジュール一覧 にて、
			FIFO 内蔵シリアルコミュニケーションインタフェースのモジュール
			名 元は、「r_scif_smster_rx」であった。
			「イーサネットコントローフ用 PTP コントローフ(EPTPC)」を 実际
		0	「EPTPOLIght モンユール」を迫加
		9	3.3 FII モジュール一覧 衣 3-1 RX64M, RX71M グルーノ用 RX
			Driver Package Fit モジュール一見 にて、 細れ込む田 TCD/ID M2S T4 Tipy ソケット ADI エジュールのエジュー
			地外区が用TCF/F MSS-14-Tilly クリットAFIモンユールのモンユー ル名 元は、「r socket」であった。
			Serial Flash memory アクセス クロック同期制御モジュール
			のRev. 元は、「2.32」であった。
		—	イーサネットコントローラ用 PTP コントローラ(EPTPC)モ
			ジュールを削除
			EPTPC Light モジュールを追加
1.04	2016.05.16	4	2.1 システム構成 にて、図中の POSIX Wrapper に「※」を追
			加し、「開発中」に変更した。
		5	2.2 RX Driver Package の特徴 にて、図中から「POSIX API
			/F」を削除した。
		6	
		(3.2 モジュール構成 にて、図中から「POSIX API I/F」を削除
		0	
		8	
			ューッネットコノトローフ册 PTP コノトローフ(EPTPC)モ ジュールを追加
			EPTPC Light モジュールを更新
			POSIX ラッパーモジュールを削除
		18	4.3.2 FIT プラグインで FIT モジュールをインストールする
			3 「コンソール」が表示されます にて、「r bsp / バージョン
			の依存説明を変更

製品ご使用上の注意事項

E.

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.	未使用端子の処理
	【注意】未使用端子は、本文の「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 表体田
	動士を開放仏感で動作でせると、誘導現象により、L31周辺のノイスが印加され、L31内部で負進電
	流か流れたり、人力信号と認識されて誤動作を起こす恐れかあります。木使用 <u></u> ふ本文「木使用
	端子の処理」で説明する指示に従い処理してください。
2.	電源投入時の処置
	【注意】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。
	同様にの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
	へいたい、「「「「」」、「」、「」、「」、「」、「」、「」、「」、「」、「」、「」、「」
2	ががる 定電圧に圧するよくの対向、減了の状态は休証ととよどが。 リザーゴマドレフ(圣約領域)のマクセス林山
з.	
	【注息】リサーノアトレス(予約領域)のアクセスを禁止します。
	アドレス領域には、将来の機能拡張用に割り付けられているリザーフアドレス(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。
4.	クロックについて
	【注意】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、リャ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
	(またけ从部発振回路)を用いたクロックに切り替える場合け、切り替え先のクロックが十分安定)
	(よたは外部元派自時)を用いたアロックに切り自たの場合は、切り自た九のアロックが十万文だし てから切し 妹うてください
_	
э.	
	【注意】型名の異なる製品に変更する場合は、製品型名ことにシステム評価試験を実施してくたさ
	同じグループのマイコンでも型名が違うと、内部ROM、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。お客様の機器・システムの設計におい
て、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には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三 * #字に開し、2014は、「脚子のまださ合いさせ」
石に生し 2 本姿料に	に損害に関し、当社は、一切ての員社を見いません。 記載されている結果は、正確を期またか持重に休成したたのですが、追いがたいことを保証するたのではなりませく、正一、古姿料に記載されている結果。
2. 本資料にの誤りに	記載されている情報は、正確を効うため展生に作成されるのですが、読みがないことを体証するものではのみなどの。カー、本資料に記載されている情報 起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 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	侵害に関し、当社は、何らの責任を負うものではありません。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	を改造、改変、複製等しないでください。かかる改造、改変、複製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。
5. 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: コンビュータ、OA機器、通信機器、計測機器、AV機器、
	家电、工作仮似、ハーンナル仮器、性楽用ロホット寺 百日気水準,給光準里(白動車)・電車(約)約(年)・六通田信日準里
	同如貝小华·辅达饭硷(日到半、电半、加加守)、又进用信亏饭硷。 防災·防抑生者 各種安全生者等
当社製品	は、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(原子力制御システム、軍事機器等)に使用されることを意図しておらず、使用することはできません。 たとえ、意図しない用
途に当社	製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。 なお、ご不明点がある場合は、当社営業にお問い
合わせく	ださい。
6. 当社製品	をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製
品をご使	用された場合の故障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。
7. 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	、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせ
ないよう た行って	、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シンク処理等、お客様の機器・システムとしての出荷保証 / ださい、時に、フィコンパコトウェアは、単独での検証は困難なため、お客様の機器・システノトレスの中全検証をお客様の表にで行って/ ださい
21100	へださい。特に、マイコンフノアウエアは、単独での使証は凶難なだめ、お各体の成益・フヘチムとしての女主使証をお各体の真正で行うてください。 の環境適合性笑の詳細につきましてけ、 赳具個別に必ず当社営業変日までお問合せください。 ご使用に際してけ、 特定の物質の今方・使用を掲制する
0. 当任表面 RoHS指	の深光過日にすめ評判につきましては、表面に加水しがす当社日来心口をでお尚日とくたとい。ことが高いしては、特定の物質の日本「使加速税制する」 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。お客様が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。
 4. 本資料に 	記載されている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。ま
た、当社	製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外
国為替及	び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の	転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負
担して頂	きますのでご了承ください。
11.本資料の 	全部または一部を当社の又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。
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その総株主の議決権の過半数
を直	接または間接に保有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	いて定義された当社の開発	、製造製品をいいます。
-----	--------	----------	--------	-----	-------	--------------	-------------

RENESAS

ルネサスエレクトロニクス株式会社

http://www.renesas.com

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24(豊洲フォレシア)

■技術的なお問合せおよび資料のご請求は下記へどうぞ。 総合お問合せ窓口:http://japan.renesas.com/contact/

■営業お問合せ窓口

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。