AP-RZT-0A (RZ/T1 CPU BOARD) USB ホスト サンプルプログラム(GCC)解説

3.1版 2023年10月02日

1. 札	要	2
1.1	概要	2
1.2	接続概要	2
1.3	本サンプルプログラムについて	3
1.4	開発環境について	3
1.5	ワークスペースについて	4
2. 1	ンプルプログラムの構成	5
2.1	フォルダ構成	5
2.2	ファイル構成	6
3. <i>I</i>	P-RZT-0A サンプルプログラム 9	9
3. <i>A</i>	P-RZT-0A サンプルプログラム9 動作説明	9
 3.1 3.2 	P-RZT-0A サンプルプログラム9 動作説明 メモリマップ1	9 2
3. 4 3.1 3.2 3	P-RZT-0A サンプルプログラム	9 2
3.1 3.2 3. 3.	P-RZT-0A サンプルプログラム	9 2 2 3
 3.1 3.2 3.3 	P-RZT-0A サンプルプログラム 9 動作説明	9 2 3 4
 3.1 3.2 3.3 3.3 3.3 3.3 	P-RZT-0A サンプルプログラム	9 2 3 4 4
 3.1 3.2 3.3 3.3 3.3 3.3 3.3 3.3 	P-RZT-0A サンプルプログラム 9 動作説明. 1 メモリマップ 1 2.1 RAM 動作時のメモリマップ 1 2.2 シリアル FlashROM 動作時のメモリマップ 1 ビルド・デバッグ方法 1 3.1 プロジェクトのインポート 1 3.2 プロジェクトのビルド 1	9 2 2 3 4 8
 3.1 3.2 3.3 3.4 3.4 3.5 	P-RZT-0A サンプルプログラム 9 動作説明 1 メモリマップ 1 2.1 RAM 動作時のメモリマップ 1 2.2 シリアル FlashROM 動作時のメモリマップ 1 ビルド・デバッグ方法 1 3.1 プロジェクトのインポート 1 3.2 プロジェクトのビルド 1 3.3 RAM 動作 1	9 2 2 3 4 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

1. 概要

1.1 概要

本アプリケーションノートでは、AP-RZT-0A に付属するサンプルプログラムについて解説します。 本サンプルプログラムの概要を以下に記します。

サンプルプログラム	動作内容
USB ホスト サンプルプログラム	・USB ホスト ファイル書き込み
	・シリアル通信
	・CAN 通信
	・ネットワーク通信
	・タイマ割り込み

1.2 接続概要

本サンプルプログラムの動作を確認する上で必要な CPU ボードとホスト PC 間の接続例を以下に示します。



LAN クロスケーブル

※AP-RZT-0A と J-Link を直接接続することはできません。

AP-RZT-0A 側(ハーフピッチコネクタ)と J-Link 側(フルピッチコネクタ)を接続するための変換アダプタが必要となります。 変換アダプタについては、J-Link 取扱店へご確認ください。

1.3 本サンプルプログラムについて

本サンプルプログラムは、ルネサス エレクトロニクス株式会社提供のミドルウェア及びドライバを AP-RZT-0A に 移植しています。

各ミドルウェア及びドライバの詳細については、以下の資料を参照してください。

ルネサス エレクトロニクス社 RZ/T1

https://www.renesas.com/jp/ja/products/microcontrollers-microprocessors/rz-cortex-a-mpus/rzt1microprocessors-real-time-control-industrial-equipment-and-networking-same-time#documents

• CMT
・資料名
RZ/T1 グループ コンペアマッチタイマ(CMT)
SCIFA
・資料名
RZ/T1 グループ FIFO 内蔵シリアルコミュニケーションインタフェース(SCIFA)
USB HMSC
・資料名
RZ/T1 グループ USB Host Basic Firmware
RZ/T1 グループ USB Host Mass Storage Class Driver(HMSC)
• RIIC
・資料名
RZ/T1 グループ RIIC サンプルプログラム
• ETHERNET
・資料名
RZ/T1 グループ マルチポート対応 ETHERNET ドライバ

1.4 開発環境について

本サンプルプログラムは統合開発環境 e² studio を用いて開発されています。 サンプルプログラムに対応する開発環境、ツールチェインのバージョン、デバッガは次のようになります。

開発環境	バージョン	ツールチェイン名	バージョン	デバッガ
e ² studio	5.3.0.023	GNUARM-NONE-EABI Toolchain	16.01	J-Link

1.5 ワークスペースについて

本サンプルプログラムの統合開発環境 e² studio ワークスペースは次のフォルダに格納されています。

サンプルプログラム	フォルダ
USB ホスト サンプルプログラム	¥sample¥APRZT0A_sample_usbh
プロジェクトフォルダ	

2. サンプルプログラムの構成

2.1 フォルダ構成

サンプルプログラムは下記のようなフォルダ構成になっています。

sample



アプリケーションノート「AN1613 AP-RZT-0A USB ファンクション サンプルプログラム(GCC)解説」を 参照してください。

2.2 ファイル構成

サンプルプログラムは以下のファイルで構成されています。

<¥sample¥APRZT0A_sample_usbh フォルダ内>

.cproject		CPROJECT ファイル
.Debuglinker		DEBUGLINKER ファイル
.info		INFO ファイル
.project		PROJECT ファイル
.Releaselinker		RELEASELINKER ファイル
.WriteSFlashlinker		WRITESFLASHLINKER ファイル
APRZT0A_sample_usbh Debug.jlin	ık	JLINK ファイル
APRZT0A_sample_usbh Debug.lau	ınch ···	LAUNCH ファイル
APRZT0A_sample_usbh WriteSFlas	sh.jlink ···	JLINK ファイル
APRZT0A_sample_usbh WriteSFlas	sh.launch ···	LAUNCH ファイル
custom.bat		カスタムバッチファイル
makefile.init		メイクファイル

<¥sample¥APRZT0A_sample_usbh¥Debug フォルダ内>

APRZT0A_sample_usbh.x	•••	RAM 動作用実行可能ファイル
-----------------------	-----	-----------------

<¥sample¥APRZT0A_sample_usbh¥Release フォルダ内>

APRZT0A_sample_usbh.bin	•••	シリアル FlashROM 動作用バイナリファイル
-------------------------	-----	---------------------------

<¥sample¥APRZT0A_sample_usbh¥WriteSFlash フォルダ内>

APRZT0A_sample_usbh.x	•••	シリアル FlashROM 書き込み用実行可能ファイル
-----------------------	-----	-----------------------------

<¥sample¥APRZT0A_sample_usbh¥inc フォルダ内>

common.h	•••	共通ヘッダファイル
eth_hwfnc.h	•••	ETHERNET ハードウェアファンクション定義ヘッダファイル
iodefine.h	•••	IO レジスタ定義ヘッダファイル
platform.h	•••	プラットフォームヘッダファイル
r_atcm_init.h	•••	ATCM 初期化ヘッダファイル
r_cmt.h	•••	CMT ドライバヘッダファイル
r_cpg.h	•••	CPG 設定ヘッダファイル
r_ecm.h	•••	ECM 設定ヘッダファイル
r_eth.h	•••	Ethernet 関連ドライバ公開ヘッダファイル
r_eth_mac.h	•••	EthernetMAC ドライバ定義ヘッダファイル
r_eth_phy.h	•••	EthernetPHY ドライバ定義ヘッダファイル
r_eth_sw.h	•••	EthernetSwitch ドライバ定義ヘッダ
r_icu_init.h	•••	ICU 初期化ヘッダファイル
r_mpc.h	•••	MPC 設定ヘッダファイル
r_port.h	•••	ポート設定ヘッダファイル
r_ram_init.h	•••	RAM 初期化ヘッダファイル
r_reset.h	•••	リセット処理ヘッダファイル
r_riic_rzt1_config.h	•••	RIIC ドライバ設定ヘッダファイル
r_riic_rzt1_if.h	•••	RIIC ドライバ API ヘッダファイル
r_scifa_uart.h	•••	SCIFA ドライバヘッダファイル
r_system.h	•••	システム設定定義ヘッダファイル
r_typedefs.h	•••	基本型定義ヘッダファイル
r_usb_basic_config.h	•••	USB ユーザ定義ヘッダファイル
r_usb_basic_if.h	•••	USB basic API ヘッダファイル
r_usb_hatapi_define.h	•••	USB 共通定義ヘッダファイル
r_usb_hmsc_config.h	•••	USB HMSC 設定ヘッダファイル
r_usb_hmsc_if.h	•••	USB HMSC ドライバヘッダファイル
sio_char.h	•••	シリアル IO 文字制御ヘッダファイル

<¥sample¥APRZT0A_sample_usbh¥src¥common フォルダ内>

ap_rzt_0a_ram.x	•••	RAM 動作用リンカスクリプトファイル
ap_rzt_0a_rom.x	•••	シリアル FlashROM 動作用リンカスクリプトファイル
exit.c	•••	終了処理
loader_init.asm	•••	ローダープログラム 1
loader_init2.c	•••	ローダープログラム 2
loader_param.c	•••	SPI ブートモード用ローダーパラメータ
r_atcm_init.c	•••	ATCM 初期化
r_cpg.c	•••	CPG 設定
r_ecm.c	•••	ECM 設定
r_icu_init.c	•••	ICU 初期化
r_mpc.c	•••	MPC 設定
r_ram_init.c	•••	RAM 初期化
r_reset.c	•••	リセット処理
vector.asm	•••	ベクタテーブルファイル

<¥sample¥APRZT0A_sample_usbh¥src¥sample フォルダ内>

init_main.c	•••	メイン処理
eeprom.c	•••	EEPROM ドライバ
eeprom.h	•••	EEPROM ドライバヘッダファイル
ethernet_app.c	•••	Ethernet アプリケーション処理
r_usb_hmsc_apl.c	•••	USB HMSC ドライバ実行処理
r_usb_hmsc_apl.h	•••	USB HMSC ドライバヘッダファイル
r_usb_main.c	•••	USB メイン処理
sdram.c	•••	SDRAM ドライバ
sdram.h	•••	SDRAM ドライバヘッダファイル
siochar.c	•••	シリアル IO 文字制御

3. AP-RZT-0A サンプルプログラム

3.1 動作説明

サンプルプログラムは、下記の動作を行います。

● USB ホスト

USB ホストポートに USB メモリを挿入すると、FAT ファイルシステムを利用して USB メモリにテキストファ イルの書き込みが行なわれます。 PC に USB メモリを挿入し、USB メモリのルートディレクトリに「HMSCDEM0.TXT」という名前のテキストフ ァイルが作成されていることを確認してください。 「HMSCDEM0.TXT」を開き、「a」が 512 バイト書き込まれていることを確認してください。

シリアル通信

SCIFA2 でエコーバックを行います。(送受信割り込み使用) COM ポートの設定は、115200bps、ビット長 8、パリティなし、ストップビット1、フロー制御なしです。 動作確認は、ホスト PC 上のターミナルソフト(ハイパーターミナルなど)を使用して行ってください。

● CAN 通信

CAN1 でエコーバックを行います。(受信 FIFO 割り込み・送信完了割り込み使用) スタンダードフォーマット、データフレーム、データ長 1~8 バイト、通信速度 500kbps の設定で動作し、受信 したデータを、そのまま送信 ID:B' 10101010100 に対して送信します。

● ネットワーク通信

Ethernet (CN9) で、Ping 応答を行います。 CPU ボードの CN9 と PC を LAN クロスケーブルで接続し、PC から IP アドレス「192.168.1.100」に対して Ping を送信してください。 (本サンプルプログラムでは、CN10、CN11 は動作しません。)

● タイマ割り込み

LD1 を 100msec 間隔で、LD2、LD4、LD5 を 200msec 間隔で ON/OFF します。(CMT 割り込み使用) また、拡張 I/O ポートより方形波を出力します。周期とピン番号を次ページの表に示します。 タイミングの生成は CMT 割り込みを使用します。 方形波出力端子一覧(1)

ピン番号	ピン名	周期	備考
CN1.5	P21/IRQ1/CS0#/MTIC5V/TIOCB1/CTS0#	20msec	
CN1.4	P22/IRQ2/RD#/MTIOC7B/TIOCD0/SCK0	20msec	
CN1.3	P23/A0/MTIC5U/TXD0/DACK1	20msec	
CN1.58	P67/IRQ15/GTIOC3B/CTXD0/TEND0/USB_OVRCUR	20msec	
CN2.26	P11/IRQ9/MTIOC4D/GTIOC2B	40msec	
CN2.25	P12/MTIOC4B/GTIOC2A	40msec	
CN2.24	P13/RAS#/MTIOC4C/GTIOC1B	40msec	
CN2.23	P14/CAS#/MTIOC4A/GTIOC1A	40msec	
CN2.21	P16/CS4#/CS2#/MTIOC3B/GTIOC0A	40msec	
CN2.20	P17/CS5#/ETH1_TXER/PHYRESETOUT#/ADTRG0	40msec	
CN2.58	P70/IRQ0/D16/MTIOC6D/RTS1#/USB_OVRCUR/TRACECLK/ENCIF00	40msec	
CN2.57	P71/D17/POE0#/POE10#/TOC2/SCK1/TRACECTL/ENCIF00	40msec	
CN2.56	P72/D18/MTIOC1A/TIC2/TXD1/SSITXD0/TRACEDATA0/ENCIF02	40msec	
CN2.55	P73/IRQ3/D19/MTCLKB/RXD1/SSIRXD0/TRACEDATA1/ENCIF03	40msec	
CN2.54	P74/D20/MTCLKA/CTS1#/SSL03/SSISCK0/TRACEDATA2	40msec	
CN2.53	P75/IRQ13/D21/MTIOC4D/GTIOC2B/SSL00/TRACEDATA3/ENCIF04	40msec	
CN2.52	P76/D22/MTIOC4B/GTIOC2A/SSL01/SSIWS0/TRACEDATA4	40msec	
CN2.51	P77/D23/MTIOC4C/GTIOC1B/RSPCK0/TRACEDATA5	40msec	
CN2.40	P91/AN101/CAS#/TXD2/ENCIF06	40msec	
CN2.39	P92/AN102/CS5#/TOC3/RXD2	40msec	
CN2.38	P93/AN103/MTIOC1A/TIC3/SCK2/ENCIF07	40msec	
CN2.37	P94/AN104/IRQ4/MTCLKB/RTS2#	40msec	
CN2.36	P95/AN105/IRQ13/MTCLKA/CTS2#	40msec	
CN2.35	P96/AN106/POE0#/POE10#	40msec	
CN2.34	P97/AN107/IRQ7/A25/ADTRG1	40msec	
CN2.43	PA0/D24/MTIOC4A/GTIOC1A/MOSI0_RED/TRACEDATA6/MDAT3	40msec	
CN2.44	PA1/D25/MTIOC3D/GTIOC0B/MISO0/AUDIO_CLK/TRACEDATA7/MCLK3	40msec	
CN2.45	PA2/D26/MTIOC3B/GTIOC0A/SSL02/DREQ2/MDAT2/ENCIF05	40msec	
CN2.46	PA3/D27/ETHSWSECOUT/GTETRG/TIOCA2/SCK2/DACK2/MCLK2	40msec	
CN2.30	PD0/AN108/CS4#	40msec	
CN2.29	PD1/AN109/CS1#	40msec	
CN2.28	PD2/AN110/WAIT#	40msec	
CN2.27	PD4/AN112/ETH2_INT	40msec	
CN2.31	PK1/CS5#	40msec	
CN2.32	PK2/A23	40msec	
CN2.33	PK3/A24	40msec	

方形波出力端子一覧(2)

ピン番号	ピン名	周期	備考
CN2.10	PM2/CATSYNC1/CATLATCH1/TCLKE/RTS4#	40msec	
CN2.9	PM3/CATSYNC0/CATLATCH0/PO16 40msec		
CN2.7	PM5/CATLEDSTER/PO18 40msec		
CN2.6	PM6/IRQ6/CATLINKACT0/PO19	40msec	
CN2.5	PM7/CATLINKACT1/PO20	40msec	
CN2.12	PU7/CATIRQ/RXD4	40msec	
CN3.30	P20/A17/MTCLKD	20msec	
CN3.29	P25/A18/MTCLKC/TEND1	20msec	
CN3.28	P26/A19/MTIOC8D/DREQ1	20msec	
CN3.27	P27/A20/MTIOC8C/TIOCB0/RTS0#	20msec	
CN3.21	P40/MTIOC8A/TXD0	20msec	
CN3.22	P41/BS#/SCK0	20msec	
CN3.23	P42/MTIOC7C/RXD0	20msec	
CN3.24	P43/WE2#/DQMUL/MTIOC8B/USB_VBUSEN	20msec	
CN3.25	P44/IRQ12/WAIT#/TCLKD/ADTRG0/CTS0#	20msec	
CN3.26	P47/WE3#/DQMUU/AH#/MTIOC6C	20msec	
CN3.53	PP0/POE8#/TEND0/MCLK2	20msec	
CN3.54	PP1/MTIOC0D/DACK0/MDAT2	20msec	
CN3.55	PP2/MTIOC0C/TCLKH/MCLK1	20msec	
CN3.56	56 PP3/MTIOC0B/TCLKC/MDAT1 20msec		
CN3.57	PP4/MTIOC0A/MCLK0	20msec	
CN3.58	PP5/PO22/MDAT0	20msec	
CN3.20	PS0/MTIOC7D/AUDIO_CLK	20msec	
CN3.19	PS1/IRQ1/MTIOC7B/SSISCK0	20msec	
CN3.18	PS2/MTIOC7C/SSIWS0	20msec	
CN3.17	PS3/MTIOC7A/SSIRXD0	20msec	
CN3.16	PS4/MTIOC6D/SSITXD0	20msec	
CN3.5	PT0/IRQ0/TIOCA3/TIOCB3/PO25/SCK2/ENCIF07	20msec	
CN3.6	PT1/TIOCA2/TIOCB2/PO26/RTS2#	20msec	
CN3.7	PT2/TIOCA1/TIOCB1/PO27	20msec	
CN3.8	PT3/IRQ11/TIOCA0/TIOCB0/PO28/CTS2#	20msec	
CN3.9	PT4/CS3#/PO29	20msec	
CN3.10	PT5/BS#/PO30/TEND2	20msec	
CN3.11	PT6/A21/DREQ2	20msec	
CN3.12	PT7/A22/DACK2	20msec	
CN4.8	P55/IRQ5/A24/ETHSWSECOUT	40msec	
CN4.18	PF7/IRQ7/A25/ETH0_TXER/RTS3#/SSL30	40msec	

アプリケーションノート AN1614

3.2 メモリマップ

3.2.1 RAM 動作時のメモリマップ

RAM 動作時のメモリマップを以下に示します。

0x0000 0000	ATCM	
0x0007 FFFF	512KB	
	予約領域	N.
0x0080 0000	ВТСМ	
0x0080 7FFF	32КВ	
	予約領域	
0x0400 0000	拡張内蔵 SRAM	
0x0407 FFFF	512KB	
	予約領域	
0x0800 0000	Buffer RAM	
0x0FFF FFFF	128MB	
0x1000 0000	SPI マルチ I/O バス空間	
0x103F FFFF	4MB	
	予約領域	
0x2000 0000	拡張内蔵 SRAM	
0x2007 FFFF	512KB	
	予約領域	
0x2200 0000	拡張内蔵 SRAM のミラー領域 512KB	
0x2207 FFFF	(0x2000 0000~0x2007 FFFF のミラー)	
	予約領域	1
0x2400 0000	拡張内蔵 SRAM のミラー領域 512KB	1
0x2407 FFFF	(0x0400 0000~0x0407 FFFF のミラー)	
	予約領域	1
0x3000 0000	SPI マルチバス I/O 空間のミラー領域	1
0x303F FFFF	4MB	
	予約領域	1
0x4C00 0000	SDRAM のミラー領域	1
0x4CFF FFFF	16MB	
	予約領域	1
0x6C00 0000	SDRAM	1
0x6CFF FFFF	16MB	
	予約領域	1
0xA000 0000	周辺 I/O レジスタ	1
0xA00F FFFF	1MB	
	予約領域	1
0xE800 0000	デバッグ用領域	
0xE800 FFFF	64КВ	
	予約領域]
0xFFFF 0000	ブート専用領域]
0xFFFF 7FFF	32КВ	
	予約領域]

 	_
0x0000 0000	.fvectors
	.rodata
	.data
	.bss
0x0002 0000	EHCI_PFL
0x0002 0400	EHCI_QTD
0x0003 0400	EHCI_ITD
0x0003 8580	EHCI_QH
0x0003 9080	EHCI_SITD
0x0003 A000	OHCI_HCCA
0x0003 A100	OHCI_TD
0x0003 C100	OHCI_ED
0x0004 0000	.text
0x0007 0000	.cstack
0x0007 2000	.svc_stack
0x0007 4000	.irq_stack
0x0007 6000	.fiq_stack
0x0007 7000	.und_stack
0x0007 8000	.abt_stack
0x0007 9000	.heap
0x0007 A000 0x0007 FFFF	未使用
0x0080 0000	.loader_param

`	0x0080 0000	.loader_param
,	0x0080 2000	.loader_text
1	0x0080 7FFF	.loader_text2

3.2.2 シリアル FlashROM 動作時のメモリマップ

シリアル FlashROM 動作時のメモリマップを以下に示します。

r		0.0000.0000	
0x0000 0000	ATCM	0x0000 0000	.fvectors
0x0007 FFFF	512KB		.rodata
	予約領域	\	.data
0x0080 0000	BTCM		.bss
0x0080 7FFF	32KB	0x0002 0000	EHCI_PFL
	予約領域	0x0002 0400	EHCI_QTD
0x0400 0000	拡張内蔵 SRAM	0x0003 0400	EHCI_ITD
0x0407 FFFF	512KB	0x0003 8580	EHCI_QH
	予約領域	0x0003 9080	EHCI_SITD
0x0800 0000	Buffer RAM	0x0003 A000	OHCI_HCCA
0x0FFF FFFF	128MB	0x0003 A100	OHCI_TD
0x1000 0000	SPI マルチ I/O バス空間	0x0003 C100	OHCI_ED
0x103F FFFF	4MB	0x0004 0000	.text
•	予約領域	0×0007 0000	.cstack
0x2000 0000	拡張内蔵 SRAM	0x0007 2000	.svc_stack
0x2007 FFFF	512KB	0x0007 4000	.irq_stack
	予約領域	0x0007 6000	.fiq_stack
0x2200 0000	拡張内蔵 SRAM のミラー領域 512KB	0x0007 7000	.und_stack
0x2207 FFFF	(0x2000 0000~0x2007 FFFF のミラー)	0x0007 8000	.abt_stack
	予約領域	0x0007 9000	.heap
0x2400 0000	拡張内蔵 SRAM のミラー領域 512KB	0x0007 A000	未使用
0x2407 FFFF	(0x0400 0000~0x0407 FFFF のミラー)		
	予約領域	0x0080 0000	未使用
0x3000 0000	SPI マルチバス I/O 空間のミラー領域	0x0080 2000	.loader_text
0x303F FFFF	4MB	0x0080 7FFF	.loader_text2
	予約領域		
0x4C00 0000	SDRAM のミラー領域	\0x3000_0000	.loader_param
0x4CFF FFFF	16MB		.loader_text
	予約領域	N. N	.loader_text2
0x6C00 0000	SDRAM		.fvectors
0x6CFF FFFF	16MB		.rodata
	予約領域		.data
0xA000 0000	周辺 I/O レジスタ		EHCI_PFL
0xA00F FFFF	1MB		EHCI_QTD
	予約領域	\ \	EHCI_ITD
0xE800 0000	デバッグ用領域	N N	EHCI_QH
0xE800 FFFF	64KB		EHCI_SITD
	予約領域		OHCI_HCCA
0xFFFF 0000	ブート専用領域		OHCI_TD
0xFFFF 7FFF	32KB		OHCI_ED
	予約領域	0x303F FFFF	.text

3.3 ビルド・デバッグ方法

3.3.1 プロジェクトのインポート

① e² studio を起動し、メニューバーの[ファイル]→[インポート]を選択します。



② [既存のプロジェクトをワークスペースへ]を選択し[次へ]を選択します。

■ インボート	- • ×
選択 アーカイブ・ファイルまたはディレクトリーから新規プロジェクトを作成します。	Ľ
インポート・ソースの選択(<u>S</u>):	
 → 一般 ☆ Convert CCRX to GNURX Project ☆ DS-5 KPIT GNUARM-RZ/NONE Project ŵ HEW プロジェクト ☆ Rename & Import Existing C/C++ Project into Workspace ☆ Renesas共通プロジェクト・ファイル ŵ アーカイブ・ファイル ☆ ファイル・システム ☆ 酸皮プロジェクトをワークスペースへ □ 設定 > > C/C 	E
(ア) (アヘ(N) > 終了(E) キャ	ンセル



③ [ルート・ディレクトリーの選択]を選択し、[参照]からサンプルプログラムのフォルダを選択します。

● インボート	
プロジェクトのインポート 既存の Eclipse プロジェクトを検索するディレクトリーを選択します。	
 ◎ ルート・ディレクトリーの選択(I): ◎ アーカイブ・ファイルの選択(Δ): プロジェクト(<u>b</u>): 	▼ 参照(<u>R</u>) ▼ 参照(<u>R</u>)
	すべて選択(<u>5</u>) 選択をすべて解除(<u>0</u>) 更新(<u>5</u>)
オプション 「ネストしたプロジェクトを検索(出) 「プロジェクトをワークスペースにコピー(C) 「ワークスペースに既に存在するプロジェクトを得す(I) ワーキング・セット	
 ワーキング・セットにプロジェクトを追加[]) ワーキング・セット(①): 	邏択(E)
(例) (例)> 終了(E)	キャンセル

④ [APRZTOA_sample_usbf]にチェックを入れ[終了]を選択します。

プロジェクトのインポート	
既存の Eclipse プロジェクトを検索するディレクトリーを選択します。	
	-
● ルート・ディレクトリーの選択(工): C:¥WorkspaceForE2¥ap_rzt_0a_gcc_sample¥sample ・	参照(<u>R</u>)
◎ アーカイブ・ファイルの違択(人):	参照(<u>R</u>)
プロジェクト(度):	
APRZTOA sample usbf(C:¥WorkspaceForE2¥ap rzt 0a gcc sample¥sample¥APRZTOA sample usbf直接アクセス)	すべて選択(<u>S</u>)
☑ APRZT0A_sample_usbh (C:¥WorkspaceForE2¥ap_rzt_0a_gcc_sample¥sample¥APRZT0A_sample_usbh 直接アクセス)	 選択をすべて解除(<u>D</u>)
	更新(E)
オプション	
□ ネストしたプロジェクトを検索(H)	
$\square J \sqcup \Sigma I D h C' J - D A C - A C - I C - (C)$	
□ ワーキング・セットにフロジェクトを追加(I)	
ワーキング・セット(Q):	選択(<u>E</u>)
⑦ < 戻る(B) 次へ(N) > 終了(E)	キャンセル

USB ファンクションのサンプルプログラムを使用される場合は[APRZT0A_sample_usbf]にもチェックを 入れてください。USB ファンクションのサンプルプログラムについてはアプリケーションノート 「AN1613 AP-RZT-0A USB ファンクション サンプルプログラム (GCC) 解説」を参照してください。 ⑤ ツールチェインのパスが変更された場合は、[OK]を選択してください。



⑥ ナビゲータウィンドウにサンプルプログラムのプロジェクトが追加されていることを確認します。

e ² C - e ² studio		
ファイル(F) 編集(E) ソース(S) リファ	フタリング(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	V) ヘルプ(H)
		· · · · · · · · · · · · · · · · · · ·
	21	
🎦 プロジェクト・エクス 🛛 🖓 🗖		Eア 🛛 💿 M 🗖 🗖
<u> </u>		
APRZTOA_sample_usbh		表示するアウトラインはありませ ん。
	※ 問題 🖉 タスク 🖳 コンソール 🕱 🔲 フロバティー 🔰 Memory Usage 🛅 スタック解析 🎕	
	Import without management	🖹 🛃 📑 🗐 🔻 📑 🔻
	Import wizard messages	
	プロジェクト 'APRZTØA_sample_usbh' に対するツールチェーンパスが	
	C:\Program Files (x86)\KPIT\GNUARM-NONEv15.01-EABI\arm-none-eabi\arm-none-eabi	\' から'C:\Program Files\⊦▼ ▶
0 項目が選択されました。	I	

⑦ 文字コードの変更を行います。メニューバーの[ウィンドウ]→[設定]を選択します。

e² 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(W) ヘルプ(H)
C → C → C → C → C → C → C → C → C → C →	● ●
	設定(P)
	😰 問題 🖳 コンソール 🗴 🔲 プロパテ 🔋 メモリー 🖏 進行状況 🧱 周辺機能 🏭 端子配置表 🧱 端子配置図 📓 コード・ 📟 🗖
	Synergy
0 項目が選択されました。	C/C++ インデクサー: (1%) ■ ■

⑧ [一般]→[ワークスペース]を選択し、テキスト・ファイル・エンコードをその他にし、[SJIS]を選択します。
 (SJIS が選択できない場合は直に[SJIS]と打ち込んでください)

e ² 設定	
フィルタ入力	ワークスペース 🗘 🔹 🔿 🔹 🔹
⊿ 一般 ▷ Security	ワークスペースの開始およびシャットダウン設定については、 <u>開始およびシャットダウン</u> を参照してください。
Web ブラウザ ▶ エディタ キー グローバレル コンテンツ・タイプ ▶ ネットワーク接続 パースペクティブ ▶ ワークスペース ▶ 開始およひシヤットダ・ ▶ 外観 検索	 □ 目勤的にビルド(B) □ ネイティブのフックまたはポーリングを使用して更新(B) □ アクセス時に更新(<u>S</u>) □ ビルド前に目勤的に保管(<u>M</u>) □ 無関係なプロジェクトを常にプロンプトなしで閉じる(<u>C</u>) ワークスペース保管間隔(分)(<u>W</u>): 5 ワークスペース名(ウィンドウ・タイトルで表示)(<u>K</u>):
比較/パッチ ▷ C	プロジェクトをオープンした際に、参照するプロジェクトを開く ◎ 常時(Y) ◎ 何もしない(N) ◎ プロンプト(P)
Library Hover > インストール/更新 > チーム > ヘルプ > 寧行/デバッグ	システム・エクスプローラーを起動するコマンド(X): explorer /E,/select=\${selected_resource_loc}
	 デキスト・ブァイル・エンコート(L) 新規デキスト・ブァイルの対応切り文子(E) デフォルト(U) (M5932) デフォルト(E) (Windows) その他(<u>0</u>): SJIS ▼
< >	デフォルトの復元(I) 適用(<u>L)</u>
?	OK キャンセル

3.3.2 プロジェクトのビルド

① ツールバーからビルドアイコンを選択します。

アイコン横のした矢印をクリックすることでコンフィグレーションごとのビルドが可能です。

e ² C - e2 studio		
ファイル(E) 編集(E) ソース(<u>S)</u> リファクタリング(T) ナヒ	ビゲート(<u>N</u>) 検索(<u>A</u>) プロジェクト(<u>P</u>) Renesas <u>V</u> iews 実行(<u>R</u>) ウィンドウ(<u>W</u>) ヘルプ(出)	
] 📸 = 🚳 = 🖻 = @ =] 🕸 = 🖸 = 隆 = 🖉 = 🔍 😕 🖋 = 🗐 🗐 🖢 = 🖗 = 🦛 😓	• 🔶 •
1 Debug	クイック・アクセス	😫 ि 🖬 C
2 Release		
として、「シェクト・エクス…」 ✓ 3 WriteSFlash		
	+	
APRZTOA_sample_usbh	表示するアウトライン	ンはありませ
	708	
	スク 🖳 コンソール 🛙 🔲 プロパティー 🔋 Memory Usage 🐚 スタック解析 🁒 スマート・ブラウ	
		- 🖻 -
Import wizard m	nessages	
		*
709191 APRZT 'C:\Program F	WA_sampie_usbn、レイリタンツールチェーンバスか iles (x86)\KPIT\GNUARH-HONEv15.01-EABI\arm-none-eabi\arm-none-eabi\' から 'C:\Program	Files\+ -
		· · ·
0 頃日か選択されました。		

構成が Debug の場合、¥Debug ワークフォルダ内に RAM 動作用の実行可能ファイルが、

Release の場合、¥Release ワークフォルダ内にシリアル FlashROM 動作用のバイナリファイルが、

WriteSFlash の場合、¥WriteSFlash ワークフォルダ内にシリアル FlashROM 書き込み用の実行可能ファイルが 生成されます。

e² studioの詳細な使用方法に関しては e² studioのマニュアルを参照してください。



3.3.3 RAM 動作

- ① まずは前節[プロジェクトのビルド]で、プロジェクトを Debug 構成にてビルドしてください。
- ② ボード上のディップスイッチを以下に示すように設定してください。







32 ビットバスブートモード 通常動作モード 水晶振動子入力

ボード上の SDRAM を使用する

USB ホストで動作

- ③ ボードに電源を投入してください。
- ④ メニューバーから[実行]→[デバッグの構成]を選択します。

e ² C - e2 studio		
ファイル(E) 編集(E) ソース(S) リファク	7タリング(T) ナビゲート(<u>N</u>) 検索(<u>A</u>) プロジェクト(<u>P</u>) Renesas <u>V</u> iews 実行(<u>R</u>) ウィンドウ(<u>W</u>)	ヘルプ(圧)
📬 🕶 📙 🕒 📎 🕶 🐔 🖷 🔝	💽 🖽 🖉 🛍 🔹 🖄 🕶 🖸 👻 🧭 👻 🎠 🏷 👻 🂁 🖉 🛀 🔲 🕥 🖄	• {} • (+ + + + + +
	(起動ヒストリーなし) クイッ	ク・アクセス 🖻 🔤 С
🎦 プロジェクト・エクス 🛛 🗖 🗖	デバッグ(D) ・ 日 デバッグの構成(B)	■ア X ⑧M □□
□ Image: Second state ▷ Image: Second state ○ Image: Second state ○ Image: Second state ○ Image: Second state ○ <	お求に入りの補成(V)	長示するアウトラインはありませ い。
	💽 問題 🧔 タスク 💷 コンソール 🛿 🔲 プロパティー 🔋 Memory Usage 🐚 スタック解析 🧇	スマート・ブラウ 🖳 🗖
	0 0 S II II I	👔 = 🚉 🚽 💷 🛨 🚽 🚽
	CDT ビルド・コンソール [APRZTOA_sample_usbh]	
	18:22:07 ビルドが完了しました(所要時間 5m:42s.94ms)	*
		-
APR7T0A cample us	hh :	•
a Arteron_sumple_us		

⑤ [APRZT0A_sample_usbh Debug]をクリックし、[デバッグ]を選択してください。



⑥ ボードとの接続が完了したらプログラムを実行し、サンプルプログラムを動作させてください。

er テバック - APK210A_sample_usbh/src/common/loader_init.asm - e2 studio					
		ク	イック・アクセス	😰 🔤 C 🎄 デバッグ	
* デバッグ 🛛 🦓 📞 マ 🍬 🍇 🕹 🖋 🖬 😓 マ 🗖 🗖	(x)= 変数 💥	• ブレ… 淵淵 レジ… 👔	🛚 モジ 🐼 式 🥊	イベ 🔲 IO 🖳 🗖	
APRZTOA sample usbh Debug [Renesas GDB Hardware Debugging]			8 at 🗖 🖄	. # x % rt r ⊽	
APRZTOA_sample_usbh.x [1]	名前	タイプ	値		
▲ 🧬 Thread #1 1 (single core) (Suspended : シグナル : SIGINT:Interrupt)	Hus				
<pre>stack_init() at loader_init.asm:73 0x802000</pre>					
C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)					
🔓 GDB server					
s loader_init.asm 🐹					
73 00802000 cps #17 /* FIQ mode */ 74 00802004 ldr sp. = fig stack e			^	□ 🗳 🗸	
75 00802008 cps #18 /* IRQ mode */			⊳ 🚰 APRZT	^r 0A_sample_usbh	
76 0080200c					
78 00802014 ldr sp, =_abt_stack_e					
79 00802018 cps #27 /* Undef mode */					
oo oooocci iii aa ah					
82 00802024					
84 0080202c ldr sp. = svc stack e			•		
•		÷		►	
🔁 コン 🗴 🧟 タスク 💿 Rene 🔋 Mem 📀 Perfo 🔮 Profile 💱 Real 🎭 Trace 🕥 Visua 🎭 スマ 🦹 問題 🕖 実行 📋 メモリー 🖓 🗖					
APRZT0A_sample_usbh Debug [Renesas GDB Hardware Debugging] C:/Renesas/e2_studio/DebugComp/arm-none-eabi-gdb (7.8.2)					
monitor set_io_access_width,RW,1,a00d0934-a00d0935,a00d0980-a00d0981,a00d0984,a00d098e-a00d098f,a00d09a8-a00d09a9,a00d09ae-a00d09af					
				•)	
中断中					

3.3.4 シリアル FlashROM 動作

- ① まずは前節[プロジェクトのビルド]で、プロジェクトを Release 構成にてビルドしてください。
- ② [init_main.c]ファイルの 65 行目にある[USER_PRG_SIZE]を Release 構成にてビルドした バイナリファイルのサイズに変更してください。 なお、弊社のサンプルプログラムをそのまま動作させる場合は変更する必要はありません。 お客様がサンプルプログラムを修正された場合は、生成されたバイナリファイルおよびマップファイルを参考に、 [USER_PRG_SIZE]を変更してください。
- ③ 続いて前節[プロジェクトのビルド]で、プロジェクトを WriteSFlash 構成にてビルドしてください。
- ④ ボード上のディップスイッチを以下に示すように設定してください。



32 ビットバスブートモード 通常動作モード 水晶振動子入力



ボード上の SDRAM を使用する



USB ホストで動作

- 5 ボードに電源を投入してください。
- ⑥ メニューバーから[実行]→[デバッグの構成]を選択します。

Image: Standard	C ADD7T0A cample uchł (szc/cample/init maji c. o? ctudio						
> Control		7月117月(T) ナビゲート(N) 絵素(A) プロジェクト(D) Renacae Viewe 宇行(R) ウィンドウ(W) ヘルプ(H)	<u> </u>				
TD3:r0h·rr078 ■ # # ■ ■ ■ #			~ ~				
I AP2TOA_sample_usbh Debug TJv2/Q(D) TJv2/Q(~ *				
[→] / □ □ □ □ / □ · □ / □ / □ / □ / □ / □ /		C* 1 APRZTOA_sample_usbh Debug ス : 昭 幅 C 株 デバッ	ッグ				
> July 10 × 10 × X > ass > ass > f(x) / 0 / 0 / 0 / 0 / 0 / 0 / 0 / 0 / 0 /		デバッグ(D)					
Image: Solution of the second sec		S loader_init_asm E init_main.c X デバッグの構成(B) 語 ア X @ M					
<pre>> P APRZTOA_sample_usbh [Write:</pre>	E 🔄 ▽	105 port_init(); → → → → → → → → → → → → → → → → → → →	\bigtriangledown				
for the formation of the formati	APRZTOA_sample_usbh [Writes	106 object Objec	~				
109 /* InitializEDR_DATALDR_DATAE the ICU settings */ * r_system.h 111 icu_init(); 112 /* Release the CHT module stop status. */ 113 /* Release the CHT module stop status. */ 114 cet_standby(); 115 /* Initialize SDRAM */ 116 /* Initialize SDRAM */ 117 sifash_prg_write(); 118 sflash_prg_write(); 121 /* Initialize CMT */ 122 /* CMT Setting */ 124 /* Initialize CMT */ 125 R_CMT_Init(CMT_CH_g, CMT_CKS_DIVISION_512); 126 R_CMT_Init(CMT_CH_g, CMT_CKS_DIVISION_512); 127 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ 128 Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø		108 ecm_init(); I iodefine.h					
111 /* InitializeDATADR_DATABe the ICU settings */ 111 icu_init/(); 112 icu_init/(); 113 /* Release the CMT module stop status. */ 114 cent_standby(); 115 /* Initialize SDRAM */ 116 /* Initialize SDRAM */ 117 sdram_init(); 118 ● #ifdef NNITE_SFLASH 120 #sflash_prg_write(); 121 #endif 122 /* CMT Setting */ 123 /* Initialize CMT_CKS_DIVISION_512); 124 /* Initialize (CMT_CH_1, CMT_CKS_DIVISION_512); 125 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); 126 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); 127 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ 128 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ 129 # Q GX 0 ICD/F 120 CDT E/LK · ICV/JUL (APRZTOA_sample_usbh] 121 Initialize CMT Memory Usage L Z990/98/ff @ X2= h · J790 128 /* Initialize STS Memory Usage L Z990/98/ff @ X2= h · J790 129 INIT // * Light(MB Sm:375.443ms) * <		109 r system.h					
		110 /* InitializLDR_DATALDR_DATAe the ICU settings */					
113 /* Release the CMT module stop status. */ * <t< th=""><th></th><th></th><th>-</th></t<>			-				
114 cet_standby(); 115 /* Initialize SDRAM */ 116 /* Initialize SDRAM */ 117 sdram_init(); 118 ##ifdef wRITE_SFLASH 120 #sflash prg_write(); 121 #endif 122 /* CMT Setting */ 123 /* CMT Setting */ 124 /* Initialize CMT */ 125 R_CMT_Init(CMT_CH_9, CMT_CKS_DIVISION_512); 126 R_CMT_Init(CMT_CH_9, CMT_CKS_DIVISION_512); 127 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ 128 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * <		113 /* Release the CMT module stop status. */	=				
		114 cmt_standby();					
In the set of		116 /* Initialize SDRAM */					
III8 III IIII IIII IIII IIII IIII IIII III I		117 sdram_init();					
121 #endif # SDRAM_S_ADDR 122 /* CMT Setting */ # USER_PRG_SIZE 123 /* Initialize CMT # USER_PRG_SIZE 124 /* Initialize CMT # USER_PRG_SIZE 125 R_CMT_Init(CMT_CH_0, CMT_CKS_DIVISION_512); # SDRAM_S_ADDR 126 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); # SDRAM_S_ADDR 127 /* Enable IRQ Interrupt (Clear CPSR.I bit to 0) */ + ecm_Init(void) : vv 128 /* Enable IRQ Interrupt (Clear CPSR.I bit to 0) */ + ecm_Init(void) : vv * 101 * 101 /* 101 * 101/574- 128 Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø Ø		120 sflash prg write();	к				
122 /* CMT Setting */ 123 /* Initialize (III */* CMT_CKS_DIVISION_512); 124 /* Initialize (III */* CMT_CKS_DIVISION_512); 125 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); 126 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); 127 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */ 128 /* III */* IIII */* IIIII */* IIII */* IIIII */* IIII */* IIIII */* IIII */* IIIII */* IIII */* III		121 #endif SDRAM_S_ADDF	ζ				
		122 # USER_PRG_SIZE	-				
125 R_CMT_Init(CMT_CH_0, CMT_CKS_DIVISION_512); R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); 127 + ⁶ cmt_standby(vold) :vd + ⁶ port_Init(vold) :vd + ⁶ port_Init(vold) :vd + ⁶ ecm_init(vold) :vd + ⁷ e		124 /* Initialize (MT */					
Image: Second Seco		125 R_CMT_Init(CMT_CH_Ø, CMT_CKS_DIVISION_512); ↔ ^S cmt_standby(vo	id)				
		126 R_CMT_Init(CMT_CH_1, CMT_CKS_DIVISION_512); ⊕ ⁸ port_init(void):	VC				
		127 128 /* Enable IRO interrunt (Clear CPSR.I bit to 0) */ ++ ecm_init(void):	vc				
▲ 印題 ④ タスク ■ コンソール 図 ■ プロパティー ● Memory Usage ● スタック解析 ● スマート・ブラウ ■ □ CDT ビルド・コンソール [APRZTOA_sample_usbh] 18:44:05 ビルドが売了しました (所要時間 5m: 37s. 443ms) ▲ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●			-				
		🔝 問題 鹵 タスク 📮 コンソール 🛛 🔲 プロパティー 🔋 Memory Usage 😼 スタック解析 🦓 スマート・ブラウ 🕒					
CDT ビルド・コンソール [APRZTOA_sample_usbh] 18:44:05 ビルドが完了しました (所要時間 5m: 37s. 443ms) く ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・ ・			<u> </u>				
18:44:05 ビルドが完了しました(所要時間 5m:37s.443ms) < m APRZTOA_sample_usbh		CDT ビルド・コンソール [APRZTOA_sample_usbh]					
APRZTOA_sample_usbh		18:44:05 ビルドが完了しました(所要時間 5m:37s.443ms)	~				
APRZTOA_sample_usbh							
APRZTOA_sample_usbh							
APRZT0A_sample_usbh			r				
	APRZT0A_sample_usbh						

⑦ [APRZT0A_sample_usbh WriteSFlash]をクリックし、[デバッグ]を選択してください。

●* デバッグ構成	AND REAL PROPERTY AND	×
構成の作成、管理、および実行		- Ale
 ・ ・ ・	名前(N): APRZTOA_sample_usbh Debug メイン 珍 Debugger ♪ Startup ↓ ソース □ 共通(C) プロジェクト(P): APRZTOA_sample_usbh C/C++ アプリケーション: Debug/APRZTOA_sample_usbh.x 変数(V) プロジェクトの検案(H) [起動前に必要に応じてビルド ビルド構成: Use Active ● 自動ビルドを有効にする ● 自動ビルドを無効にする ● ワークスペース設定の使用 <u>ワークスペース設定の構成</u>	参照(E) 参照(E) ▼
< 111 ・ 111 ・ 111 ・ フィルターー致: 12 / 14 項目	「前回保管した状態に戻す(⊻)」	適用(⊻)
0	デバッグ(<u>D</u>)	閉じる

 ⑧ ボードとの接続が完了したら[init_main.c]の129行目の左隅をダブルクリックすることでブレークポイントを設定し、 プログラムを実行してください。

e² デバッグ - APRZTC	A_sample_usbh/src/sam	nple/init_main.c - e2	studio) 🗆 🗙
ファイル(<u>E</u>) 編集(<u>E</u>)	ソース(<u>S</u>) リファクタ	リング(T) ナビゲート	(<u>N</u>) 検索(<u>A)</u> プロ]ジェクト(<u>P</u>) Rei	nesas <u>V</u> iews 実行	(<u>R</u>) ウィ	rンドウ(<u>W</u>)	ヘルプ(圧)	
1 🖬 🕶 🔡 🔞 📎	• 🔦 • 🐔 🗟 ≌ 🎉	🖉 🖏 🕶 🚺 🕶	9a 🔸 🕅 💵 🗉	📕 💦 🕄 🖓 .	in 🖬 🗟 🖂 🏟	8 🙋 🔬	? 🕶 🗾 🐓	💌 🖓 💌 🌤	$ \diamondsuit \bullet \bullet \bullet \bullet$
						クイック	・アクセス	🖹 🖻 🖥 C 🕻	な デバッグ
☆ デバッグ ☆	🎉 🔩 🔻 🎭 🕪 💷	😭 🖏 🕹 🛤 i+	▽ - □	(x)= 変数 💥 💁	ブ 🔐 レ 🛋	€ %	`式 👴イ	🛅 I 😤 P	8
▲ C APRZT0A_sar	mple_usbh WriteSFlash [i	Renesas GDB Hardwa	e Debugging]			X) 📲 🖻 🖄) if 🗙 🔆	📫 🖻 🝸
▲ 2 APRZTOA_	sample_usbh.x [1]			名前	タイプ	値			^
4 🧬 Thread	#1 1 (single core) (Susp () at init manin av122.0	pended:フレークホイン	×F)						
	h() at init_main.c:122.03	www.arm.none.e	abi-adb (7.8.2)						
GDB serve	er	agoonip, ann none ei	ioi geo (rioiz)						
~									
			•					4	
Denden init een	District service of D		Dinit main a M						
Is loader_init.asm	adaan init():	5 loader_inic.asm	init_main.c 23				BE // 9 N	10 JU2	
120	suram_init();					^		04 cample us	le 🔄 🎽
	#ifdef WRITE_SFLASH	- () -				_		UA_sample_us	ushh [Dahua
123	#endif	=();					P J AFRZ	TUA_sample_t	JSDII [Debug
124	/* CNT C-++	*/							
125	/* Initialize CM	ung "/ Ит */							
127 00040da0	R_CMT_Init(CMT_C	CH_0, CMT_CKS_DIVIS	ION_512);						
128 000400aC	K_CHI_INIC(CHI_C	LH_I, CHI_CKS_DIVIS	10N_512);						
130	/* Enable IRQ in	nterrupt (Clear CPS	R.I bit to 0) *	/					
132 0004046-	4 4 4 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	// Clear Cr5k.1			ł	Ψ	4		
	Per -								
「レージー」とW (Wetforman 次 Real-time W Irace () Visual Exp 図 ARM Core W スマート・ W 問題 O 美行利能フ ① メモリー ロ									
📕 👻 🐘 🔛 🐨 🔄 🕶 🔂 🕹 🔂 👘 🛄 🕼 🗐 📰 🕞 🐨 📑 👻 🖬 🚱 👘									
APRZ I UA_sample_us	on writesHash [Renesa:	s GDB Hardware Debu	gging] C:/workFoi	der/e2_studio_v5	.3/DebugComp/arr	m-none-	eabl-gab (7.8	.2)	
Breakpoint 1, mai	n () at/src/sample	e/init_main.c:122							
122 s	flash_prg_write();								-
•		III							•
中断中	<u></u>	書き込み可能	ス	- M					

 「レークポイントで停止したらボード上の SDRAM に Release 構成で生成したバイナリファイルを展開します。
 [コンソール] (e² studio V6.0.0 以降では[Debugger Console]) から
 [restore <u>filename</u> binary <u>bias start end]</u>と入力しエンターを押します。

(下線が引かれた項目については以下を参考に、ユーザの環境に合わせて入力してください。)

filename	Release 構成で生成したバイナリファイルのパスを指定してください。		
bias	SDRAM の先頭アドレス[0x4C000000]を指定してください。		
start	[0]を指定してください。		
end	[init_main.c]ファイルの 65 行目にある[USER_PRG_SIZE]を指定してください。		



S = If w/d _ ADP7T0A_cample_usbb/crc/cample/init_main.c., e2 ctudie					
フィスパック・AFR2100_Sample_List/isid/per/mic_mainter/ez/subult ファスパック、編集(E)、レスパン、ロースイン、アイトン・ション・ション・ション・ション・ション・ション・ション・ション・ション・ショ					
· · · · · · · · · · · · · · · · · · ·	1 32 93 La 1→ = 3 (() (2) / → / / / / → () → · · · · · ·				
	クイック・アクセス 酔 励 C な デバッグ				
APRZIVA_sample_usbn writesHasn [Renesas GDB Hardware Debugging] APRZIVA_sample_usbh.x [1]					
Am Phread #11 (single core) (Suspended : ブレークポイント)					
main() at init_main.c:122 0x40d9c					
C:/WorkFolder/e2_studio_v5.3/DebugComp/arm-none-eabi-gdb (7.8.2)					
GDB server					
	· · · ·				
🔝 loader_init.asm 🛛 🔓 init_main.c 🕼 loader_init.asm 🔂 init_main.c 🔀	🗄 🏦 アウト 🔥 プロジ 🕺				
118 /* Initialize SDRAM */ 119 00040d98 sdram init();					
	► ▷ ▷ APRZIOA_sample_usbt				
<pre>sil22 00040d9c sflash_prg_write();</pre>					
123 #endif 124					
125 /* CMT Setting */ 126 /* Initialize CMT */					
127 00040da0 R_CMT_Init(CMT_CH_0, CMT_CKS_DIVISION_512);					
120 000400ac R_CHT_INIC(CHT_CH_I, CHT_CKS_DIVISION_512); 129					
130 /* Enable IRQ interrupt (Clear CPSR.I bit to 0) */	-				
	• <u> </u>				
🖳 コンソール 🛛 🕐 Performan 🖏 Real-time 🗞 Trace 🚫 Visual Exp 🖄 ARM (Core 🦦 スマート・ શ 問題 🕐 実行可能フ 🚺 メモリー 🙄 🗖				
🕒 🗶 🔆 🖟 🔐 🔛 📰 💆 📰 🗸 🔝 🖬 💷 📰 🗸 🔝 🗸 💷					
122 sflash_prg_write();	3coulo_v5.5/DebugC011p/a111-101ie-eabi-yob (7.6.2)				
restore C:\WorkspaceForE2\ap_rzt_0a_gcc_sample\sample\APRZT0A_sample_usbh\Rel	ease\APRZT0A_sample_usbh.bin binary 0x4C000000 0 0x00040000				
rescoring of the Alice of the space of the all the all the all the sample (sample (sam	ampre_uson (rerease (or relion_sumpre_uson of in theo memory (or recover				
	,				
中断中 合口 書き込み可能 スマート					

LD4 と LD5 が点灯したら[終了]アイコンをクリックし、ボードとの接続を切り、電源を切ってください。
 電源が切れたらボード上のディップスイッチを以下に示すように設定してください。







ボード上の SDRAM を使用する



USB ホストで動作

 の
 の
 ボードへ電源を投入することで、シリアル FlashROM に書き込まれたサンプルプログラムが動作することを
 確認してください。

ご注意

- ・本文書の著作権は株式会社アルファプロジェクトが保有します。
- ・本文書の内容を無断で転載することは一切禁止します。
- ・本文書に記載されている内容についての質問等のサポートは一切受け付けておりませんのでご了承ください。
- ・本文書の内容については、万全を期して作成いたしましたが、万一ご不審な点、誤りなどお気付きの点がありましたら弊社までご連絡 ください。
- ・本サンプルプログラムに関して、ルネサス エレクトロニクス株式会社への問い合わせはご遠慮ください。
- ・本文書の内容に基づき、アプリケーションを運用した結果、万一損害が発生しても、弊社では一切責任を負いませんのでご了承ください。
- ・本文書の内容は、将来予告なしに変更されることがあります。

商標について

・e² studio 、RZ および RZ/T1 は、ルネサス エレクトロニクス株式会社の登録商標、商標または商品名称です。

・その他の会社名、製品名は、各社の登録商標または商標です。

ALPHAPROJECT

株式会社アルファプロジェクト

〒431-3114 静岡県浜松市中央区積志町 834 https://www.apnet.co.jp E-Mail: query@apnet.co.jp