

R20AN0418JU0100

Rev.1.00

2016.09.23

Renesas Synergy<sup>™</sup>プラットフォーム

IAR C-SPY によるアプリケーションデバッグの実行

本資料は英語版を翻訳した参考資料です。内容に相違がある場合には英語版を優先します。資料によっては 英語版のバージョンが更新され、内容が変わっている場合があります。日本語版は、参考用としてご使用の うえ、最新および正式な内容については英語版のドキュメントを参照ください。

#### 要旨

本アプリケーションノートでは、SK-S7G2 Blinky with ThreadX プロジェクトテンプレートを使用していま す。Renesas Synergy™の IAR Embedded Workbench 入門を参照し、Synergy スタンドアロンコンフィグレータ (SSC)を使用して、プロジェクトを作成してください。



# 目次

1.	初期設定3
2.	デバッグベイシック&ライブウォッチ6
3.	条件付きのデータブレークポイント9
4.	C-SPY Macros12
5.	SWO 経由 PRINTF およびセミホスティング13
6.	ITM イベント15
7.	割り込みログ17
8.	Attach to running target18
9.	ThreadX RTOS プラグイン19
10.	デバッグ中のスタック分析 (View > Stack Usage)21



#### 1. 初期設定

- 1. ワークスペースのプロジェクトの上で右クリックして、[S7G2-SK Blinky with ThreadX]のプロジェクトオ プションを開きます。
- Synergy ボードに実装されている J-Link ドライバが選択されていることを確認します。手順3および4 で、最終的なデバッグプローブ設定を行います。Options ウィンドウを開けたままにしておいてください。

Category:	Factory Settings
General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel CMSIS DAP GDB Server IAR ROM-monitor I-jet/JTAGjet J-Link/J-Trace TI Stellaris	Setup Download Images Extra Options Multicore Plugins Driver URK/J-Trace Run to Main Simulator Angel CMSIS DAP GDB Server IAR ROM-monitor Ust /ITAGet Trace Trace Trace Godebugger\Renesas\R7FS7G27H.d
PE micro RDI	

図1 J-Link ドライバ

3. 全てのプロジェクトのオプションへアクセスできるようにプロジェクトツリーの中で太字のプロジェク トを必ず選択します。



図 2 Debug プロジェクト

4. [Debugger]の下の[J-Link/J-Trace]を選択して、[Setup]のタブで、[CPU clock]を SWO クロックに必要と される 120MHz に設定します。

Category:	Factory Settings	
General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel	Setup Connection Breakpoints Reset Normal JTAG/SWD speed Auto Initial 1000 kHz CPU clock: 120 MHz	
CMSIS DAP GDB Server IAR ROM-monitor L-jet/ITAGjet JHJnk/J=Trace TI Stellaris Macraigor PE micro RDI ST-LINK Third-Party Driver	O Fixed 1000 IkHz   O Adaptive 2000   ETM/ETB    Prefer ETB	

#### 図3 CPU クロックの設定

5. J-Link/J-Traceの[Connection]タブから、本書で全デバックに使用する SWD インターフェースを選 択します。



Category				1. Second Second	
	8			Factory	Settings
General Options Static Analysis					
Runtime Checking					
C/C++ Compiler	Setup Connection	Breakpoints			
Assembler Output Converter	Communication	Device 0	Suddelan.		
Custom Build	© 035.	Device u	o enal no:		
Build Actions	O TCP/IP:	IP address 🗸 🗸			
Debugger	IP address:	aaa.bbb.ccc.ddd	Serial no:		
Simulator	Interface	JTAG scan chain			
Angel	⊖ JTAG	JTAG scan chain	with multiple ta	argets	
CMSIS DAP GDB Server	Own	TAP number:	0		
IAR ROM-monitor	<b>3</b>	Scan chain d	contains non-A	ARM devices	
I-jet/JTAGjet		Preceedin	g bits; 0		
J-Link/J-Trace TI Stellaris	Log communica	ation			
Macraigor	SPROJ DIRK	cspycomm log			
PE micro	er noe_bine	and a community			
Third Darby Driver					

図4 SWD インターフェースの選択

- 6. 全ての変更を確認して[OK]をクリックします。
- 7. [**Project**]メニューから[**Download and Debug**](ショートカットキー:CTRL+D)を選択してアプリケーションをダウンロードしてデバッグします。

guix - IAR Embedded Workbench IDE	low Help
Image: Constraint of the second se	rection ns ect ect Alt+F7 ystem F7 Ctrl+F7 F8 alysis Ctrl+Break ebug Ctrl+Break

図 5 Download and Debug の選択

8. アプリケーションプログラムは main 関数のところで止まります。



022000	8 10 CH V V V V V V V V V V V V V V V V V V	1. 江田口口口 4. 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1				
5-1082583 m	2 ×					
Workspace a	LAR Information Center for AUM Renesas Synergy making	man0 + x	Disessentity			
Delnag -	17	TX USER TRACE BUFFER DECLARE;	Gom	<ul> <li>Mencly</li> </ul>	- D	
Filas to 00	10 - Fendif		Disassembly			
E alabi - Debug y	19		and the second second second	int aginfront		
H-m Synergy	20 void tx_ar	plication_define_internal(wold * first_unused_memory); //				
L-@_Output	21		sein:			
1.12	22 void ts_ar	plication_define_internal(void * first_unused_memory)	0.612.02	Und:580	POSH	(R2. 1
	23 🖯 (	Second Complete a construction of the second sec	(Norscon))	datt:	Ligg(3)	North Street
	24 /* Doe	a nothing. Default implementation of tx_application_define_user().	0x3204	0xb672	CPSID	1
	25 332_24	RAMETER NOT_USED(first_unused_memory);		La heroe	Serrec())	
	20 .		083206	ORIVIT ORITON	. DL .	_(K_1)
	21	All and the first state of the first state of the state o	0x228a	0x2000	M/VPG	14 DI
	20 Void tx Ap	plication_define_internal(void - first_unused_memory);	0x320c	0sbd02	FOF	(R1. F
	20 1010 14 01	hyreacton merine marianta - riske anamed memory, weak wes wyrathou	0x320e	0x0000	HOVS	RU. BL
	71 wold by as	alization define (wold + first unused memory)	_ts_throad_ini	tialize:		
	32	harden and hord and annoth	6x3210	0xb510	FUSH	(R4 I
	33 blinks	thread greate();	Gx3212	0x4c0c	LDR N	R4. [T
	14		0x3214;	0x2000	NOVS	R0. #L
	35 (1)	Aifdef TX USER ENABLE TRACE	0x3216	0x6020	STR	HO. IF
	36	TX USER EMABLE TRACE;	0x3218:	0x6060	STR	R0. []
	37 -	fendif	nc	9e2100	WORD :	24 . er
	38	200000	0x221c	0x1104 0x0020	ATVD II	P0 P/
	39	g_hal_init();	0x3220	0x6001	STR	R1. II
	40	ranna ma Ma Ian Ilan nan an an	return _0			10.000
	41 tx_app	lication_define_user(first_unused_memory);	0x3222	0\$02880	MOVS 1	RG. #1
	42 - )		Ux3224	0x6120	STR	RG. [1
(	43		amabi_ree	HENT C.D. H		
		old)	0x3226:	0x2100	HOVS	H1. X1
	45日 (		0x3228.	Oxf104 0x0024	ADD V	#0, 84
	46disa	ble_irq();	UXJ22C	OXIVIT OXIC48	PL .	-anar
	47 tx_ker	<pre>ust_enter();</pre>	0x3230	0g2000	NOTS	RG . #C
	48		6×3232	0x5BaD	STR	R0 [F
	49 Peture	- 92	0x3234	0x60e0	STR I	R0. [1
	20		0x3236:	0x6168	STR	RG. IF
presentation in the local division of the lo	24		0x3238	0x61a8	STR	R0. [1

図 6 main()関数での停止

9. ソースコード内でライン番号を有効にします。[Tools]メニューから[Options] > [Editor] > [Show line numbers]を選択します。デバッグ中でもいつでもライン番号を有効にできます。

▼ B Fil	ename Extensions	IDE Options	-	)
space cc space cc s filab1 - Deb to ∰ Synergy to Output	nngure viewers nnfigure Custom Argument Variable nnfigure Tools	S Common Fonts - Key Bindings Language Editor Messages Project - Source Code Control - Debugger - Stack - Register Filter - Terminal I/O	Tab size:     8       Indert size:     2       Tab Key Function:     Insert tab       Insert tab     Indert with spaces       Show right margin     Printing edge       O Columns:     80       File Encoding     Default character encoding:       System     Auto-detect character encoding       EOL characters:     PC	Syntax highlighting  Auto indent Configure  Show line numbers Scan for changed files Show bookmarks Show fold margin Enable virtual space Remove trailing blanks Auto code completion and parameter hints Show source browser tootips Show ine break characters
	47 48 49	return	1 0;	OK Cancel Apply Help

図7 ライン番号

## 2. デバッグベイシック&ライブウォッチ

- 1. [Disassembly]ウィンドウが開いていない場合、[View]メニューから開きます。C レベルおよびアセンブ ラーレベルの両方で、コードを使用すればひと手順でできます。
- 2. [Step into]、[Step over]、[Step out]、[Restart debugger]、[Go]、[Break]の機能を使います。





図 8 Debug 機能

- 3. [View]メニューから[Register]ウィンドウを開き、コードを追いながら、いかにレジスタが値を変更して いくかを見ます。
- 4. [View]メニューから[Call Stack]ウィンドウを開き、1つの手順内の call (呼出し)を見ます。
- 5. アプリケーションを中断または停止して、レジスタを見てスタック情報を呼び出します。
- 6. プロジェクトエクスプローラにある blinky\_thread\_entry.c ファイルをダブルクリックします。図9のスク リーンショットのように、while ループ内でシンプルカウンタがインクリメントされます。
  - a. int counter = 0; > 15 行目
  - b. counter++; >40行目、while  $\nu$ -プ内



図 9 インクリメントカウンタ



10. ソースコードに変更を加えた後、[Project]メニューから、[Make & Restart Debugger] (ショートカットキ -: Ctrl+R) または、上部の[Make & Restart Debugger]アイコンをクリックします。

Add Group Import File List Add Project Connection Edit Configurations Remove Create New Project		Make & Restart Debut	igger Disassembly
Import File List Add Project Connection Edit Configurations Remove Create New Project		AR Information Center for ARM Renesas Synergy   man.c. blinky_thread_entry.c.         blinky_thread_entry.c.         blinky_thread_entry.c.           9         * & Bprief         Blinky example application	Disassembly
Add Project Connection Edit Configurations Remove Create New Project		AR Information Center for ARM Renesas Synergy   man.c   blinky_thread_entry.c   blinky_thread_entry() • x 9   * 8 prior 8 linky example application	Disassembly
Edit Configurations Remove Create New Project		AR Driomation Center for ARM Renesas Synergy   man.c blinky_thread_entry.c 000000 +      9     * @brief Blinky example application	Sala V Heren
Remove Create New Project		9 * Chrief Blinky example application	
Remove Create New Project			do to (Menuly
Create New Project		10 *	Disassembly
create treat collector		11 Blinks all leds at a rate of 1 second using the the thr	0x2438: 0xf01e 0x0f
<ul> <li>A CALLER AND ADDRESS OF ADDRESS OF ADDRESS ADDRES ADDRESS</li></ul>		12 • Only references two other modules including the BSP, 10	0x243c: 0xd101
Add bosting Project		13 -	0x243e: 0xecbc 0x8a
Options	Alt+F7	15 int counter = 0:	0x2442: 0xe8bc 0x01
		16 wold blinky thread antry/wold)	0x2446: 0x138C 0x88
Version Control System	×.		0x244a: 0x2001
Make	F7	18 /* Define the units to be used with the thready sleen	0x244c: 0x4010
Course No.	Ca. 17	19 const wint32 t thready tick rate Hz = 100;	0x2450: 0xe7fe
Compac	CUNTRY	20 /* Set the blink frequency (must be de thready tick re	0x2452: 0xb662
Rebuild All		21 const mint 22 t from in ht = 2.	0x2454: 0xbf30
Clean		22 /# Calculate the delay in terms of the thready tick re	0x2456: 0xf3bf 0x8f
Batch build	F8	22 const wint32 t delay = thready tick rate Hz/freg in hz	0x245a: 0xb672
	11.25-	24 /* LED turn change */	0x245c: 0x680a
C-STAT Static Analysis	>	25 has lade t lade:	0x245e: 0x4612
Data Dulla	Carl Buch	25 DSp_reds_t_reds;	0x2460: 0x2a00
Stop enits	CITI+ Break	26 /* DSD state variable */	0x2462: 0x6002
Download and Debug	Ctrl+D	27 IOPORT_IEVEL_C IEVEL - IOPORT_IEVEL_HIGH;	0x2464: 0xd0f5
Debug without Downloading		20 /# Can IFD information for this based #/	0x2466: 0x4a07
Control of the second se	1000	23 / Get heb information for this board /	0x2468: 0x211b
Make & Restart Debugger	Ctrl+R	30 R_BSP_LedsGet (#Leds);	0x246a: 0x6011
Restart Debugger	Ctrl+Shift+R		0x246c: 0xe7cc
Download		32 /* If this board has no leas then trap here */	0x246e: 0x0000
		33 II (U == leas.lea_count)	0x2470: 0x1ffe1194

2 10 Make & Restart Debugger

- 11. [View]メニューから[Live Watch]ウィンドウを開き、"counter"と入力します。
- 12. アプリケーションをリセットして main 関数に戻ります。
- 13. [Go]をクリックして、動作中にカウンタ値がどう変化するかを見ます。アプリケーションを停止せずに 変数値をモニタすることができます。
- 14. [Live Watch]では、短いアップデート間隔が必要になる場合が時々あります。例えば、デフォルトで は、1000ms で設定されていますが、1ms まで短縮できます。アップデート間隔は、[Live Watch]ウィン ドウで右クリックし、[Options]メニューから、 Debugger > Update intervals (milliseconds) > Live watch と選択して、変更可能です。

Image Section     Image Section     S
Update rategy romak HOY R2. R

図 11 Live Watch のアップデート間隔

15. デバッガを中断させて、アプリケーションをリセットし、再度、強制的に"counter == 0"を設定します。

## 3. 条件付きのデータブレークポイント

特定の条件が合った場合に特定の時点で実行を停止するには、コンディショナルブレークポイントを挿入し ます。

1. blinky\_thread\_entry.cの57、58行目に以下の表現があります。

/\* Delay \*/
 tx\_thread\_sleep (delay);

a. 右クリックをして[Toggle Breakpoint (Code)]を選択します。



#### 図 12 Break する位置

- b. 再度右クリックをして [Edit Breakpoint]を選択します。
- c. [**Conditions**]の[**Expression**]ボックス内に[counter>10]とタイプします。
- d. この設定により、カウンタ値が10を超えた場合、実行が停止されます。



Copy Paste	Edit Breakpoint	×
Complete Word Complete Code Parameter Hint Match Brackets Toggle All Folds	// There Code Break At: :\Users\RafaelTa\Synergy\lab1\src\blinky_thread_entry.c}.58.1 Breakpoint type Ovemide default	dit
Insert Template Open Header/Source File	> e the next Software Action	
Go to Definition Go to Declaration Find All References Find All Calls to Functions	= IOPORT_LI	
Find All Calls from Functions Toggle Breakpoint (Code) Toggle Breakpoint (Log)	= IOPORT_LI	ncel
Toggle Breakpoint (Trace Start) Toggle Breakpoint (Trace Stop) Toggle Breakpoint (Trace Filter) Enable/disable Breakpoint	<pre>all board LEDs */ _t 1 = 0; i &lt; leds.led_count; i++) ct.p_api-&gt;pinWrite(leds.p_leds[i], level);</pre>	
Character Encoding	,	

図 13 実行停止: カウンタ値が 10 を超えた場合

- 2. アプリケーションをリセットして、mainへ戻り、強制的に force counter == 0を設定します。
- プログラムを再度走らせます。View > Live Watch からカウンタが変わるのを見ます。条件が真の場合、ブレークポイントは有効になります。
- 4. ここでは表示されていませんが、実行が停止した場合、条件が判定され、その結果が偽の場合、実行は 再開します。複雑なブレークポイントは負荷がかかりますが、とても効力を発揮します。
- 5. 下記種類のシンボルは式に使用できます。
  - a. C/C++ シンボル
  - b. アセンブラシンボル(レジスタ名およびアセンブララベル)
  - c. C-SPY マクロ関数
  - d. C-SPY マクロ変数
- かなり複雑な式も書くことができます。しかし、これらの式はその式の真偽を判定しなくてはならないので、性能が損なわれます。
- 7. ブレークポイントがトリガされるときにアクションボックスが別の式を評価するために使用されます。
- 58 行にある条件付きのブレークポイントを削除します。削除する方法の1つは、[View]メニューから [Breakpoints]ウィンドウを開き、ブレークポイントを全て削除します。このウィンドウから新規でブレ ークポイントを設定することもできます。
- 9. 15 行目のグローバル変数カウンタを右クリックします。メニューで、今回カウンタの[Set Data Breakpoint]を選択します。

Declaration	- 1	winny_uncos_encyte incone
Debug	×	1 ₽ /***********************************
Files 😤 🛤	^	2 * File Name : blinky thread entry.
🗆 🗊 lab1 - Debug 🛛 🗸 🗸		3 * Description : This is a very simpl
		V *******
Source Files		Cut
		Сору
→ ± _ synergy_gen	11	6 #include "b Paste
B hal ontry c	111	7
		8 🖂 / * * * * * * * Complete Word
		9 * @brief Complete Code
- Inc		10 * Parameter Hint
l – – 🖓 🛄 src		11 * Blinks a Match Brackets
→ ⊕ <mark>◯</mark> bsp		12 t On Jur not Toggle All Folds
		13 * insert rempiate /
		14 ******* Open Header/Source File
- Blab1.map		15 int counter
🖵 📮 🗋 lab1.out		16 void blinky
- 🖽 🗀 Output		Go to Declaration of 'counter'
🗋 blinky_thread.o		18 /* Dofi Find All References to 'counter'
		Find All Calls to 'counter'
bsp_clocks.o		Find All Calls from 'counter'
D bop_common.o		20 /* Set
hsp_common_reds.o		21 const u Find in Irace
- Displacing.co		22 /* Calc Toggle Breakpoint (Code)
		23 const U Toggle Breakpoint (Log)
📥 🗋 bsp_irq.o		24 /* LED Toggle Breaknoint (Trace Start)
Disp_leds.o		25 here lod T is the is the set
		2.5 DSP_TEC Toggle Breakpoint (Trace Stop)
Dsp_qspi.o		26 /* LED Toggle Breakpoint (Trace Filter)
		27 ioport Enable/disable Breakpoint
D hsn_shrk o		28 Set Data Breakpoint for 'counter'
- Dbsp_sorto		29 /* Get Set Data Log Breakpoint for 'counter'
		30 R BSP I Set Trace Start Breakpoint for 'counter'
🗋 dl7M_tIn.a		31 Cot Tana Cha Backword ( )
└── D hal data o		Set Trace Stop Breakpoint for 'counter'

#### 図 14 変数用ブレークポイントの設定

10. [Breakpoint]ウィンドウでは、([View]メニューから[Breakpoints]), データブレークポイントを編集しま す。

	— Disp_leas.or	25		
	— 🗋 bsp_qspi.o	26	Edit	i
	— 🗋 bsp_register_protection.o	27	Delete	e
	— [] bsp_rom_registers.o	28	Disable	
	- bsp_sblk.o	29	Enable All	m
	- Dicommon dete o	30	Disable All	e
lab1		<	Delete All	
×	Breakpoint		New Breakpoint	
	Data @ counter [size 4] [Read.	Write] [Ox1FFE]	IOBE - DAIFFEIT	
	para e counter l'arre al lucar			
	para e commer farze el fuedo			
	para e comirca [arze 4] [maan			
	Dara e comirci [3126 4] [4644			
2	Dara e comitor [3126 4] [4644			

#### 図 15 ブレークポイントの編集

11. ブレークポイントの [Access type]を[Read]に設定し、[Match data]の値を 0x05 (5d)に設定します。

Counter Edit   Access type Size   Read/Write Manual   Write Trigger range   Requested: 0x1FFE10BB   Enable Ix1FFE10B8 - 0x1FFE10BB   Value: 0x00000005   Mask: 0xFFFFFFFF	Edit Breakpoint Data	×
Match data       Qx1FFE10B8 - Qx1FFE10BB         Enable       Qx1FFE10B8 - Qx1FFE10BB         Value:       Qx00000005         Mask:       QxFFFFFFFFF	Counter Access type Read/Write Read Write	Size Auto (4) Manual Trigger range Requested:
	Match data Enable Value: 0x0000005 Mask: 0xFFFFFFF	0x1FFE10B8 - 0x1FFE10BB Effective: 0x1FFE10B8 - 0x1FFE10BB

図 16 ブレークポイント範囲の編集

- 12. [Go]をクリックします。
- 13. 変数への読み出しがあり、データが一致した場合にアプリケーションは停止します。
- 14. ライトおよびリード/ライトのように追加のアクセスの設定を試してください。

#### 4. C-SPY Macros

- 1. 古いブレークポイントをすべて削除します。
- 2. blinky\_thread\_entry.cの57、58行目にある式に戻ります。

/\* Delay \*/
tx\_thread\_sleep (delay);

- 3. その行を右クリックして、[Toggle Breakpoint (Log)]を選択します。それから、もう一度右クリックして [Edit Breakpoint]を選択します。
- 4. C-Spy マクロメッセージを有効にして、[Message]ボックスに""data = ", counter"を追加します。



Edit Break	point	×
🕒 Log		
	at:	
IC: No.	aqe:	Edit
"da	a = ", counter	
Conc	itions	
Expr	ession:	
	ondition true	
00	ondition changed	
	ОК	Cancel

図 17 C-SPY メッセージ

- 5. プログラムを実行します。[Debug Log]ウィンドウ内にメッセージが見えます。このブレークポイントに 条件を付けることができます。
- 5. SWO 経由 PRINTF およびセミホスティング
- 1. 単純な printf() 関数コールをアプリケーションの while 文の中に追加します。例えば:

printf("Hello Synergy!SSP 1.10\n");



#### 図 18 Printf 関数

2. ソースコードに変更を加えた後、[Project]メニューから、[Make & Restart Debugger] (ショートカットキ ー: Ctrl+R)または、上部の[Make & Restart Debugger]アイコンをクリックします。



Add Files					
Add Group		March B. State Part			
Import File List		Make & Restart Debu	Jgger		
Add Designt Connection			Burgarandaha		
Paul Project Connections.		IAR Information Center for ARM Renesas Synergy main.c blinky_thread_entry.c Dinky_thread_entry() * X	Lasassembly		Thereas
Edit Configurations		9 * Cbrief Blinky example application	Goto	.~	Memory
Remove		10 *	Disass	embly	
		11 * Blinks all leds at a rate of 1 second using the the thr		0x2438: 0xf0	le 0x0f1
Create New Project		12 * Only references two other modules including the BSP, IC		0x243c: 0xd1	01
Add Existing Project		13 *		0x243e: 0xec	bc 0x8a1
Ontinne	AltaF7	14 ************************************		0x2442: 0xe8	bc 0x0ff
opuoni	100111	15 int counter = 0;		0x2446: 0xf3	8c 0x880
Version Control System	¥.	<pre>16 void blinky_thread_entry(void)</pre>		0x244a: 0x20	01
	100	17 早 f in the second		0x244c: 0x80	18
маке	F/	18 /* Define the units to be used with the threadx sleep		0x2446: 0x47	70
Compile	Ctrl+F7	<pre>19</pre>		0x2450: 0xe/	10 60
Rebuild All		20 /* Set the blink frequency (must be <= threadx_tick_ra		0x2452: 0x06	20
Clean		21 donst uint32_t freq_in_hz = 2;		0x2454 0x51	bf BySEG
Batch build	E9	22 /* Calculate the delay in terms of the threadx tick ra		0x245a: 0xb6	72
baten bulut.	10	23 donst uint32_t delay = threadx_tick_rate_Hz/freq_in_hz		0x245c: 0x68	0a
C-STAT Static Analysis	5	24 /* DED type structure */		0x245e: 0x46	12
		25 bsp_leds_t leds;		0x2460: 0x2a	00
stop build	CTU= BLEBK	26 /* LED State Variable */		0x2462: 0x60	02
Download and Debug	Ctrl+D	2/ ioport_level_t level = ioport_level_High;		0x2464: 0xd0	£5
Debug without Downloading				0x2466: 0x4a	07
C		29 /* Get LED information for this board */		0x2468: 0x21	1b
Make & Restart Debugger	Ctrl+R	30 K_BSP_LedsGet(&leds);		0x246a: 0x60	11
Restart Debugger	Ctrl+Shift+R	31 32 (4 TE shite based bar on their sheet bars by 47		0x246c: 0xe7	00
Download	>	32 /* if this board has no leas then trap here */		0x246e: 0x00	00
Victoria and the fi		33 li (U == leds.led_count)		0x24/0: 0x1f	te1194
SFR Setup		3*년 1		0x2474: 0x1t	re1008

図 19 Make & Restart Debugger

- 3. [View]メニューから[Terminal I/O]を選択して、[Terminal I/O]ウィンドウが開いていることを確認します。
- 4. プログラムを走らせ、終了するまでの時間を見ます。
- 5. デバッガを終了します。
- 6. プロジェクトオプションを開き、カテゴリから[General Options]を選択し、[Library Configuration]タブの[stdout/stderr]を[Via SWO]に変更します。

Static Analysis         Runtime Checking         C/C++ Compiler         Assembler         Output Converter         Custom Build         Build Actions         Linker         Debugger         Simulator         Angel         GDB Server         IAR ROM-monitor         I-jet/JTAGjet         J-Link/J-Trace         TI Stellaris         Macraigor         PE micro         RDI	Category: General Options	-			
Output Converter Custom Build Build Actions Linker Debugger       Normal       Use the normal configuration of the C/C++ nutrime library. No locale interface, C locale, no file descriptor support, no multibytes in printf and scanf, and no hex floats in strtod.         Simulator       Configuration file:         Angel       STOOLKIT_DIR\$VINC\c\DLib_Config_Normal.h         GDB Server       Enable thread support in library         IAR ROM-monitor       Library low-level interface implementation         I_jet/JTAGjet       None         J-Link/J-Trace       Semihosted         TI Stellaris       Semihosted         Macraigor       IAR breakpoint         PE micro       RDI	Static Analysis Runtime Checking C/C++ Compiler Assembler	Target Output Library C	onfiguration Library Options	s MISRA-C:200 · ·	
Simulator       Configuration file:         Angel       \$TOOLKIT_DIR\$\INC\c\DLb_Config_Normal.h         GDB Server       Enable thread support in library         IAR ROM-monitor       Library low-level interface implementation         I-jet/JTAGjet       None         J-Link/J-Trace       Semihosted         TI Stellaris       IAR breakpoint         Macraigor       IAR breakpoint         PE micro       RDI	Output Converter Custom Build Build Actions Linker Debugger	Normal V Right Strength Normal V Right Strength Normal Strengt	Description: Ise the normal configuration of untime library. No locale interfi e descriptor support, no multi canf, and no hex floats in strt	of the C/C++ ace, C locale, no bytes in printf and od.	
GDB Server IAR ROM-monitor I-jet/JTAGjet J-Link/J-Trace TI Stellaris Macraigor PE micro RDI	Simulator Angel	Configuration file: STOOLKIT_DIR\$\INC\c\	DLib_Config_Normal.h		
	GINDI DA GDB Server IAR ROM-monitor I-jet/JTAGjet J-Link/J-Trace TI Stellaris Macraigor PE micro RDI	Enable thread support in Library low-level interface None Semihosted IAR breakpoint	n library simplementation stdout/stdem O Via semihosting I Via SWO	CMSIS	

- 7. プログラムをビルドしてダウンロードします。
- 8. プログラムを実行し、終了するまでの時間を見ます。
- 9. 何か違いがわかりましたか?



ご覧の通り、セミホスティングがデータを送信するためにコアを停止する間に、SWOは最大速度で送信さ れ、その後再び実行されます。

#### 6. ITM イベント

時間を計測するためにITMイベントログを設定するには、下記の手順を完了してください。

- 1. [blinky\_thread\_entry.c]から、48 行目で IOPORT レベルが高い間に、while ループ内に ITM イベントを追 加して、7行目に必要なヘッダーファイル#include <arm\_itm.h>を入れます。入れない場合はビルドエラ ーが発生します。大量のデータを転送する必要はないので、8ビットバリアントを選択します。例) ITM EVENT8 WITH PC(1,1)
- 2. 53 行目のインポートレベルが低い場合、2 番目の ITM イベント[ITM\_EVENT8\_WITH\_PC(1,0)]を追加し ます。LED がトグルされた場合、2つの ITM イベント間の時間を測定します。

40	while (1)
41 🖨	{
42	counter++;
43	<pre>printf("Hello Synergy! SSP 1.10\n");</pre>
44	/* Determine the next state of the LEDs */
45	<pre>if(IOPORT LEVEL LOW == level)</pre>
46	{
47	level = IOPORT LEVEL HIGH;
48	ITM EVENT8 WITH PC(1,1);
49 -	}
50	else
51 🖨	{
52	level = IOPORT LEVEL LOW;
53	ITM EVENT8 WITH PC(1,0);
54 -	}
55	

図 20 ITM イベントの設定

- 3. ビルドしてからダウンロードします。
- 4. [J-Link] メニューから[Timeline]を選択します。
- 5. [Timeline]ウィンドウでは、[Event]ログフィールド内でポイントし、右クリックして[Enable]を選択しま す。



lab1	Navigate	>	3.92	
×	Zoom	>		_
	Enable			
	Clear ITM3:			
Detailor	Size	>		_
Interrupts	Style	ues		
Call Stack Power Log	Hexadecimal			
ITM1	Go to Source Select Graphs			
	Time Axis Unit	>		
ІТМЗ	Profile Selection			
Time	0.2s 0.4s	0.6s	0.8s 1.0s	

図 21 イベントログを有効にする

- 6. タイムラインにいくつかのイベントが出てくるまで、数秒間の間プログラムを走らせます。
- [Timeline]ウィンドウにイベントを見つけた場合、重要な部分にどのくらいの時間をかけたか測定できます。拡大 / 縮小するとより簡単にイベントが探しやすくなります。ITM ログフィールドを右クリックするとタイムラインを拡大 / 縮小できますが、ショートカットキー[+]と[]を使ってもできます。

Events       Zoom In       +         Zoom Out       -       F00D, PID: 001BB101 TSG performed         Clear       10 ns       naming performed for 1 range (16384 bytes)         ITM1:       1 us       needed: 0.314s (Prepare: 0.090s, Compare: 0.005s, Erase: 0         Size       10 us       ytes/sec)         Style       100 us       Ta\Synergy\lab1\Debug\Exe\lab1.out         Y       Show Numerical Values       1 ms         Io ms       100 ms       1 s         Select Graphs       1 ns       1 m         Profit Selection       1 m       0 x00000000	×Г	Zoom	3	Zoom to Selectio	n Enter			
Time Axis Unit         1 m           Profile Selection         10 m           1 h         0x0000000	x Debug Log	Events Enable Clear ITM1: Size Style Show Numerical Values Hexadecimal Go to Source Select Graphs	> >	Zoom In Zoom Out 10 ns 100 ns 1 us 10 us 100 us 1 ms 100 ms 100 ms 1 s 10 s	* -	F00D, PID: 001BB101 performed ramming performed fo needed: 0.314s (Prepa ytes/sec) Ta\Synergy\lab1\Deb performed	TSG r 1 range (163) we: 0.090s, Co ug\Exe\lab1.c	84 bytes) mpare: 0.005s, Erase: ( put
	t	Time Axis Unit Profile Selection	>	1 m 10 m 1 h		0×0000000		0x0000001

#### 図 22 タイムラインのズーム

 イベント間の時間を測定するには、タイムライン内で最初のイベントを見つけ、次のイベントへ移行す るのに、[Shift]キーを押して右矢印を押します。印があるフィールドの上にカーソルを持ってきて、イ ベント間の時間を見ます。 9. ITM イベントについてもソースコードナビケーションはわかりやすくなっています。イベントをダブル クリックすると、ソースコードの正しい行に移動します。

Image:     Image: <th>🗅 😂 🖬 🎒 🎒 🐰 🖻 🛍 🗠 🗠</th> <th>~ ~ * <b>%</b> ½ 🖪 🖻 🗇 🗇</th> <th>👙 🍉 🍞 📴 😲 🖄 🗶 🎪</th>	🗅 😂 🖬 🎒 🎒 🐰 🖻 🛍 🗠 🗠	~ ~ * <b>%</b> ½ 🖪 🖻 🗇 🗇	👙 🍉 🍞 📴 😲 🖄 🗶 🎪
IM       Morkspace       IAR Information Center for ARM Revessas Synergy   man.c. blinky_thread_entry.c.       blinky_thread_e	5 B B B B B B B X		
Workspace     ×       Debug     0       Files     12 binky_thread_entry.c       Dispersive     0       Files     12 binky_thread_entry.c       Dispersive     41       Counter++;       Counter++;       SourceFiles       Binky_thread_entry.c       Binky       Count	1 ETH <b>500</b>		
Debug       40       {         Files       f:       0;         files       f:       0	Workspace 3	IAR Information Center for ARM Renesas Synergy   main.c bl	inky thread entry.c blinky_thread
<pre>Files t: Eq Files t: Eq Source Files t: Eq Source Files t: Exp Source Files t: Exp Files t</pre>	Debug ~	40 白 (	
<pre></pre>	Files to Bi	41 counter++;	
<pre>43 if (IOPORT_LEVEL_LOW == level) 44 44 44 44 45 45 45 1TM_EVENT8_WITH_PC(1,1); 1evel = IOPORT_LEVEL_HIGH; 46 47 48 49 49 49 49 49 40 40 40 4000000; 50 51 1evel = IOPORT_LEVEL_LOW; 52 53 54 7* Update all board LEDs */ 55 56 57 58 58 59 60 7* Delay */ 61 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7</pre>	D Clabl - Dobug	42 /* Determine the ne	xt state of the LEDs */
*     Output     44     {       46     ITM_EVENT8_WITH_PC(1,1);       46     Ievel = IOPORT_LEVEL_HIGH;       47     }       48     else       49     ievel = IOPORT_LEVEL_LOW;       50     ITM_EVENT8_WITH_PC(1,0);       51     Ievel = IOPORT_LEVEL_LOW;       52     }       53     ievel = IOPORT_LEVEL_LOW;       54     /* Update all board LEDs */       55     for(uint32_t i = 0; i < leds.led_count; i++)		43 if (IOPORT_LEVEL_LOW	== level)
45     ITM_EVENT8_WITH_PC(1,1);       46     level = IOPORT_LEVEL_HIGH;       47     }       48     else       49     (       50     ITM_EVENT8_WITH_PC(1,0);       51     level = IOPORT_LEVEL_LOW;       52     }       53     /* Update all board LEDs */       56     {       57     g_ioport.p_api->pinWrite(leds.p_leds[i],       58     }       60     /* Delay */       61     tx_thread_sleep (delay);       63     }       64     OFF       70000001     0x0000000       0x0000001     0x0000001	Le Source Files	44 🛱 - {	
46     level = IOPORT_LEVEL_HIGH;       47     }       48     else       49     (       50     ITM_EVENT8_WITH_PC(1,0);       51     level = IOPORT_LEVEL_LOW;       52     ;       53     /* Update all board LEDs */       56     {       57     g_ioport.p_api->pinWrite(leds.p_leds[i],       58     ;       59     /* Delay */       60     /* Delay */       61     tx_thread_sleep (delay);       62     ;       64     OFF       70000000     0x00000001	- G Src	45 ITM_EVENT8_WITH_P	C(1,1);
*     Defailed     0     <	- g _ synergy_gen	46 level = IOPORT_	LEVEL_HIGH;
48     else       49     (       90     (       91     (       92     (       93     (       94     (       95     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       96     (       97	blinky_thread_entry.c	47 }	
49     1       1     1       1     1       1     0       1 </td <td>Hentry.c</td> <td>48 else</td> <td></td>	Hentry.c	48 else	
S0     FTM_EVENTS_WITH_PC(1,0);       S1     level = IOPORT_LEVEL_LOW;       S1     level = IOPORT_LEVEL_LOW;       S2     }       S3     /* Update all board LEDs */       S6     {       S6     {       g_ioport.p_api->pinWrite(leds.p_leds[i],       S8     }       S9     /* Delay */       61     tx_thread_sleep (delay);       62     }       63     }       K     OFF       PowerLog     OFF       TM1     0x00000001       0x00000001     0x00000001	Synergy	49	
31     Ievel = IOPORT_LEVEL_LOW;       33     /* Opdate all board LEDs */       54     /* Opdate all board LEDs */       55     for (uint32_t i = 0; i < leds.led_count; i++)		50 ITM_EVENTS_WITH_P	C(1,0);
>2     }       53     \$4       54     /* Update all board LEDs */       56     {       56     {       58     }       58     }       59     (       60     /* Delay */       61     tx_thread_sleep (delay);       63     }       64     OFF       Power Log     OFF       1     0x00000001       0x00000001     0x00000001		51 IEVEL = IOPORT	LEVEL_LOW;
>3-3     /* Update all board LEDs */ for (uint32_t i = 0; i < leds.led_count; i++)		52 1	
St     // Optice Init Source Datas       St     for (uint32_t i = 0; i < leds.led_count; i++)		54 /* Undate all board	LEDE #/
S6     100 (unlos_c r r r r), r r routrid_count, r), r)       56     {       57     g_ioport.p_api->pinWrite(leds.p_leds[i],       58     }       59     /* Delay */       61     tx_thread_sleep (delay);       62     }       63     }       64     OFF       Cal Stock     OFF       PowerLog     OFF       TM1     0x00000001       0x00000001     0x00000001		55 for (uint 32 t i = 0.	i < leds led count: i++)
\$77     g_ioport.p_api->pinWrite(leds.p_leds[i],       58     }       59     /* Delay */       61     tx_thread_sleep (delay);       62     }       63     }       64     OFF       Cal Stack     OFF       PowerLog     OFF       TtM1     0x00000001       0x00000001     0x00000001		56 - I	1 ( 1000-100_000mo; 1.),
58         3           59         /* Delay */           61         tx_thread_sleep (delay);           62         3           63         3           64         0FF           0FF         0FF		57 g ioport.p api-	>pinWrite(leds.p leds[i].
59         /* Delay */           61         tx_thread_sleep (delay);           62         }           63         }           64         OFF           63         OFF           ColdsLog         OFF		58 - }	
60         /* Delay */           61         tx_thread_sleep (delay);           62         }           63         }           64         0FF           Call Stack         0FF           PowerLog         0FF           TM1         0x00000001           0x00000001         0x00000001		59	
61         tx_thread_sleep (delay);           62         }           63         }           64         tx_thread_sleep (delay);           63         }           64         0FF           0FF         0FF           0x00000001         0x00000001		60 /* Delay */	
62         3           63         3           64         0FF           Call Stack         0FF           Power Log         0FF           TTM1         0x00000001           0x00000001         0x00000001		61 tx_thread_sleep (de	lay);
63         }           64         64           ×         Deta Log         OFF           Interrupt         OFF           Call Stack         OFF           Power Log         OFF           ITM1         0x00000001           0x00000001         0x00000001		62 - }	
Web1         64 L           X         Data Log         OFF           Interrupts         OFF         OFF           Call Stark         OFF         OFF           Power Log         OFF         OFF           ITM1         0x00000001         0x00000001		63 }	
Notes         OFF           Interrupts         OFF           Call Stack         OFF           Power Log         OFF           TTM1         0x00000001           0x00000001         0x00000001	lab1	64 L	
Interrupt         OFF           Call Stock         OFF           Power Log         OFF           ITM1         0x00000001           0x00000001         0x00000001	* Data Log		OFF
Cell Stock         OFF           Power Log         OFF           ITM1         0x00000001           0x00000001         0x00000001	Interrupts		OFF
Power Log         OFF           ITM1         0x00000001         0x00000001	Call Stack		OFF
ITM1 0×00000001 0×00000001	PowerLog	$\frown$	OFF
		a anahanan	0.0000001
TT /2		0×00000000	0×0000001
	The second se		

図 23 タイムラインからソースの検索

10. 同じ方法で tx\_thread\_sleep (delay) がどのくらいの時間がかかっているか測定してみてください。 注: 60 および 62 行に ITM イベントを追加します。



図 24 tx\_thread\_sleep delay のタイミング

### 7. 割り込みログ

割り込みログ機能を使うには、以下の手順を行います。



- 1. ブレークポイントを全て削除します。
- 2. ビルドしてダウンロードします。
- 3. [Timeline]を開き、J-Link メニューから[Interrupt log]および[Interrupt Log Summary]ウィンドウを開き ます。
- 4. タイムラインウィンドウで、[Interrupt Log]を有効にします。



- 5. アプリケーションを実行[Go]します。
- 6. [Timeline]ウィンドウには全ての割り込みが表示されます。
- 7. 選択してズームしながら、SysTick にかかるサイクル数を確認します。

	SysTick <mark>,</mark>						1.64 us			
Ce	all Stack									
Po	wer Log									
	Events									
<	2.0599955s	2.05	99960s	2.0599965s	2.0599970s	2.0599975s	2.0599980s	2.0599985s	2.0599990s	2.0
		Time	Interrupt			Status	Progr	am Cou	Execution	Time
	5s 879991.7	7 us	SysTick			Leave			0.64	us
	5s 899991.0	9 us	SysTick			Enter				
	5s 899991.7	3 us	SysTick			Leave			0.64	us
	5s 919991.0	6 us	SysTick			Enter				
	5s 919991.7	0 us	SysTick			Leave	<u></u>		0.64	us
	5s 939991.0	3 us	SysTick			Enter	<u>242424</u>			
3	5s 939991.6	7 us	SysTick			Leave	101010		0.64	us
	5s 959990.9	9 us	SysTick			Enter	3222			

図 25 SysTick タイミング

#### 8. Attach to running target オプションによる実行中プログラムの追跡

リセットせず、またシステム上で実行中のプログラムに影響することなく対象システムへ接続するには、以 下の手順を行います。

- 1. アプリケーションプログラムのビルド、ダウンロード、実行[Go]をしてください。LED ランプが点滅し ている場合は、アプリケーションが実行中です。
- 2. 変化するカウンタを[View]メニューから[LiveWatch]ウィンドウより見ることができます。
- 3. 実行を停止せずにデバッガを終了してください。
- 4. LED ランプがまだ点滅しているのが確認できます。
- 5. [Options]メニューから [Debugger]を選択して、[Download] タブより[Attach to running target] オプションを 設定します。

Kunnume Unecknig       Setup       Download       Images       Extra Options       Multicore       Plugins         Assembler       Output Converter       Attach to running target       Use flash to ader(s)       Use flash to ader(s)         Simulator       Override default.       StoOLKIT_DIRS\config\flashloader\Renesas\Flashl	load     Images     Extra Options     Multicore     Plugins       unning target     Images     Images     Images       nload     Images     Images     Images       download     Images     Images     Images       ioader(s)     Images     Images     Images       ie default     Images     Images     Images       LKIT_DIR\$\config\flashloader\Renesas\Flashl     Images     Images	eneral Options tatic Analysis untime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Simulator Angel CMSIS DAP GDB Server IAR ROM-monitor IAR ROM-monitor
--	---	--

26 Attach to running target

- 6. [Ctrl+D]または[Download and Debug]ボタンを押してデバッガを再起動します。
- 7. ターゲットがリセットされていないためにカウンタ変数が進行しているのを見てください。ターゲット が実行中ですが、デバッガが完全にコントロールしているので、[Stop]を押すと実行中のプログラムは 停止します。変数およびレジスタを全て可視化することもできます。この機能はアプリケーションのク ラッシュを調べる場合にとても便利です。
- 8. 前の設定に戻るために、デバッガを停止して[Attach to running target]機能を無効にします。全ての機能は **Options** > **Debugger** > **Download** より無効にできます。

#### 9. ThreadX RTOS プラグイン

IDEのThreadXで、組み込み式 RTOS awareness プラグインを使用するには、以下の手順を行います。

1. **Options** > **Debugger** > **Plugins** から、RTOS awareness プラグインを有効にします。



Options for node "lab1"		×
Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel CMSIS DAP GDB Server IAR ROM-monitor I-jet/JTAGjet J-Link/J-Trace TI Stellaris Macraigor PE micro RDI ST-LINK Thicd-Party Driver	Setup       Download       Images       Extra Options       Multicore       Plugins         Select plugins to load:	Factory Settings
TI XDS	ОК	Cancel

図 27 ThreadX awareness プラグイン

- 2. [Ctrl+D]または[Download and Debug]ボタンを押してデバッガを再起動します。
- 3. デバッグ中に ThreadX に新しいメニューが追加されているのがわかります。

D 🚅 🖬 🕼	Thread List		🙀 🍡 🛐 🔍 🐢 🦛 🚳	🖌 💱 🛯 🖓 👘	色 🕁 🛃
5-12	Message Queues			i bi	
ETM SWO	Semaphores				
Workspace	Mutexes	ad c main c	main() 🔫 🗙	Disassembly	
Debug	Byte Pools		1	Go to	✓ Memory
Files	Block Pools		void g_hal_	Disassembly	
□ 🗇 lab1 - D 	Timers Event Flag Groups		#if defined	4	int main(v {
	Execution Profile Communication Performance Metrice		<pre>#doiind #in #pragma wea #elif defin</pre>	main: 0x3	462: 0xb580 disa
-⊞ [      └⊞ [    └── [	Memory Performance Metrics Synchronization Performance Metric		#define WEA #endif	0x3	464: 0xb672 tx_ker 466: 0xf7ff 0xff
	Thread Performance Metrics		#ifdef TX_U	0x3	return 46a: 0x2000
	a sinc Fi 🗋 bsp	18	#endif	0x3 0x3	46c: 0xbd02 46e: 0x0000
	E Cramework	20	<pre>void tx_app</pre>	_tx_thread	_initialize: 470 0xb510

図 28 ThreadX メニュー

4. [ThreadX] メニューから[ThreadX List]ウィンドウを開きます。



Renesas Synergy<sup>™</sup> プラットフォーム

File Edit View	Project Debug Disassembly J-Link ThreadX
0 🛩 🖬 🕼	Thread List
5-12	Message Queues
ETM SWO	Semaphores
Workspace	Mutexes
Debug	Byte Pools
	Black Pools

図 29 ThreadX List ウィンドウの表示

5. [ThreadX List]ウィンドウの active タスクの一覧で、スタック、最大スタック使用、スレッド状態、およびスレッドの呼び出し回数等の追加情報も表示されます。

ID		Name	Priority	State	Run Count	Stack Ptr	Stack Start	Stack End	Stack Size	Max Stack Usage
	0x1ffe1008	Blinky Thread	1	Sleep	21	0x1ffe2688	0x1ffe2318	0x1ffe2717	1024	144
		No Task		1 Lana 22						

#### 図 30 ThreadX List ウィンドウ

- タスクがない場合は、しばらくアプリケーションを動作させるか、もしくは、少なくともマルチタスク が始まるまでアプリケーションを実行させておかなければなりません。
- スレッド用の最大使用スタックや使用可能なスタックも見てください。アプリケーションプログラムに スレッドのサイズを設定する時に役に立ちます。
- 8. デバッガを停止して終了してください。

#### 10. デバッグ中のスタック分析 (View > Stack Usage)

EWSYN の SSP パッケージでデバッグ中の画像スタック分析を有効にするには、下記よりデバッガの追加オプションを有効にする必要があります。Project > Options > Debugger > Extra Options > Use command line options:

--proc\_stack\_main=g\_main\_stack, g\_main\_stack+sizeof(g\_main\_stack)



Runtime Checking   C/C++ Compiler   Assembler   Output Converter   Custom Build   Build Actions   Linker   Debugger   Simulator   Angel   CMSIS DAP   GDB Server   IAR ROM-monitor   I-jet/JTAGjet   J-Link/J-Trace   TI Stellaris   Macraigor   PE micro   RDI   STI-LINK   Setup Download Images Extra Options Multicore Plugins Multicore Plugins Maraigor
---

図 31 スタック分析用の Command line options

2. Tools > Options >Stack から[Enable Graphical Stack display]を選択して、追加で画像スタックディスプレイを有効にします。

D I II	
The Configure Vieway Configure Vieway Configur	po tarcking e a bytwa SK Corcel Aury Help

図 32 画像スタックディスプレイの有効化

- 3. [Ctrl+D]または[Download and Debug]ボタンを押してデバッガを再起動します。
- 4. main()に来たら、 View > Stack > Stack 1 から Stack ウィンドウを開いてください。



Renesas Synergy<sup>™</sup> プラットフォーム

Image: Source Browser     Image: Source Browser       Image: Disasembly     Image: Source Brow	The control vi	lew Project Debug	Disasse	anbiy 7-bitk roots window help
T         Workspace           Source Browser         *           Source Browser         *           Source Browser         *           Breakpoints         *           Disassembly         *           Her for Reveise Syvery [solines WEAK SEP_ATTRIBUTE           Program weak tx, application, define_user         -           Disassembly         *           Her for Reveise Syvery [solines WEAK SEP_ATTRIBUTE         -           Her for Reveise Syvery [solines meak tx, application, define_user         -           Her for Reveise Syvery [solines meak tx, application, define_user         -           Her for Reveise Syvery [solines meak tx, application, define_user         -           Her for Reveise Syvery [solines meak tx, application, define_internal (woid * first_unused_nemory);         *           Statics         woid tx_application_define_internal (woid * first_unused_nemory);           Auto         /         * Does nothing. Defult implementation of tx_application_define           Uck Watch         ;         * Does nothing. Defult implementation of tx_application_define           Macros         ;		Messages	•	+
1 AVI	5-6	Workspace		
Auto     C-STAT     * mte for Revenue Symegy [bit.mage immine]       Breakpoints     #define NEAK SEP_ATRIBUTE       Disassembly     #define NEAK SEP_ATRIBUTE       Disassembly     #define NEAK SEP_ATRIBUTE       Breakpoints     #define NEAK SEP_ATRIBUTE       Disassembly     #define NEAK SEP_ATRIBUTE       Breakpoints     #define NEAK SEP_ATRIBUTE       Symbolic Memory     #define NEAK SEP_ATRIBUTE       Register     Tr_USER_TRACE_BUTFER_DECLARE       Watch     #endif       Locals     void tr_application_define_incernal(void * first_inused_memory);       Statics     void tr_application_define_incernal(void * first_inused_memory);       Auto     /* Does actains_Default implementation of tr_application_define       Quick Watch     ;	ERM 580	Source Browser	- e [	
ebug     Breakpoints       Breakpoints     #define NEXX SEP_ATIBIOTE       Disassembly     #define NEXX SEP_ATIBIOTE       Books     #define NEXX Sep_Atibiotion_define_Incernal (Note + first_unued_memory);       Watch     #define NEXX Sep_Calars       Live Watch     /* Does notains_ Default implementation of tx explication_define       Quick Watch     ;	Workspace	C-STAT		nter for Renesas Synergy  lab1.map   main.c
<pre>void tx application define internal(void * first unused memory);</pre>	Debug Files B Class He Cost He Cost	Breakpoints Disassembly Memory Symbolic Memory Register Watch Locals Statics Auto Live Watch Quick Watch Macros	ж Э	<pre>#define WEAK_REF_ATTRIBUTE #program weak tr_application_define_user = tr_applicat telif define#(_GNUC) #define WEAK_BEF_ATTRIBUTEsturibute_ ( weak, alias "tr_appl #endif #ifdef Tr_USER_TRACE_BUFFER_DECLARE; #endif void tr_application_define_internal(void * first_unused_memory); void tr_application_define_internal(void * first_unused_memory); /* Dees nothing. Default implementation of tr_epplication_defin SSF_PARAMETER_NOT_USED(first_unused_memory); void tr_application_define_internal(void * first_unused_memory); </pre>
	C	Stack		Stack 1
Stack + Stack 1		Terminal I/O Images Cores	Ī	Stack 2 processing access (vold * 1135_Didded_Diddedy) blinky_thread_create(); #inder TX_USER_ENABLE_TRACE
Stack         Stack 1         pplication_define(wold * first_unused_memory)           Terminal I/O         Stack 2         Dilinky_thread_create();           Images         Cores         different fix_USEA_RUBBLE_TRACE		Code Coverage C-RUN Symbols	•	IX USER ENABLE TRACE; #endif g_bsl_init();
Stack     Stack 1       Terminal I/O     Stack 2       Images     Dilinky_thread_create();       Cores     #inder IX_USER_NUBLE_TRACE       Code Coverage     TUSER_NUBLE_TRACE;       C-RUN     #statif       Symbols     g_hal_init();		Toolbars Status Bar	•	<pre>tx_application_define_user(first_unused_memory); }</pre>

図 33 Stack ウィンドウを開く

- 5. アプリケーションを実行したままにしてしばらくの間(5~10秒)おきます。
- 6. デバッガを停止して、[Stack]ウィンドウでスタック使用情況を見てください。スタックウィンドウの上部にグラフで最大の使用量のログが表示されます。

n 🚅 🖬 🎒 🛁 🗴 🗞 I	8 00 - 455XB000	A 64 @ 193 1	1 🔆 🌲 ab. ab.
t . I sieta. deta			
	2014		
III See		1	
venspace	5003 1	AR Information Center	for Renesas Synergy lab1.map_main.c
Debug	MAIN_STACK 🔻	13	<pre>#define WEAK_REF_ATTRIBUTEattribute ( weak,</pre>
Files to B	216 bytes used out of 1280 (16%)	14	4end12
E Clabi - Debug -	Stark range: 0x1EEE0008 - 0x1EEE0508	15	AIGAS TV HOTE TELOF BUFFER DECIME
La Constray	URINELIA URI	17	TV HEED TOACE REFERE DECIMAL
- Opharty	UXISEE0494 UXEDUUEDU4	18 -	tendi F
	0x12220436 0x00001000	19	
	DATESTAAL DATESTO	20	<pre>void tx_application_define_internal(void * first_unuse</pre>
	0+155504A4 0+0000220B	21	
	Ow1FFFD436 Ow000022B0	22	<pre>void tx_application_define_internal(void * first_unuae</pre>
	0x1FFE04AC 0x01000000	23日	
	0x15550480 0x0000000	24	/* Does nothing. Default implementation of tx_app.
	0s1FFE04B4 0s1FFE1818	20	SSP_PARAMETER_NOT_USED(FIFSt_Unused_memory);
	0x1FFED4B8 0x0000000	27	1
	Ow1FFE04DC Ow000030ED	28	wold tx application define internal (wold * first unuse
	0x1FFE04C0 0x1FFE1818	29	wold th application define user (wold * first unused me
	0s1FFE04C4 0s00000400	30	
	0s1FFE04C8 0s0000001	31	<pre>void tx_application_define(void * first_unused_memory)</pre>
	0x1FFE04CC 0x00000001	32日	1
	0x1FFE04D0 0x00000001	33	blinky_thread_create();
	0x1FFE04D4 0x00000001	25	ACTION TV. DEED FULSTE TRACE
	0s1FFE04D8 0s0000002	36	TY HEED FEARLY TOLOR.
	OSIFFE04DC 08000031FB	37 -	tendif
	URIFFEDAEU URFOFOFOFO	38	2223/22
	ORIFFEDER ORDEDIAL	39	<pre>g_hal_init();</pre>
	DATEXTORN DATEXTERS	40	
	001FFF04P0 001FFP0508	41	<pre>tx_application_define_user(first_unused_memory);</pre>
	0x1FFF04F4 0x0000120B	42 -	1
	0x1FFE04F6 0x00000002	40	and manufactured
	0x1FFE04FC 0x00003363	45 1	
	0s1FFE0500 0s0000002	46	disable irg():
	0s1FFE0504 0sFFFFFFF	47	ts wernel_enter();
		48	
		49	return 0;
		50	2
		51	
leb1		101	***

図 34 Stack ウィンドウ

# ホームページとサポート窓口

Renesas Synergy<sup>™</sup> ギャラリー:

https://synergygallery.renesas.com/support

テクニカルサポート窓口:

- 米国:
  - 国: <u>https://renesas.zendesk.com/anonymous\_requests/new</u>
- ・ヨーロッパ: <u>https://www.renesas.com/en-eu/support/contact.html</u>
- 日本: <u>https://www.renesas.com/ja-jp/support/contact.html</u>

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
Rev.1.00	2016.09.23	-	初版

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.	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。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。未使用端子は、本文「未使
	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、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.	リザーブアドレス(予約領域)のアクセス禁止
	【注意】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、将来の機能拡張用に割り付けられているリザープアドレス(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
	l I.
	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。
5.	製品間の相違について
	【注意】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
	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。

	こ汪意書を
	<ol> <li>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みます。以下同じです。)に関し、当社は、一切その責任を負いません。</li> <li>当社製品、本資料に記載された製品データ、図、表、ブログラム、アルブリズム、応用回路例等の情報の使用に起因して発生した第二者の特許権、薬作権その他の</li> </ol>
	2. 当社後間、年夏村に記載された後間が、「人間、な、「レージス、「ルージスス、加州自由が多の情報の使用に起因して光上した第二目のも前種、有作権での他の 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. 当社制品を、全部または一部を問わず、改造、改変、復制、その他の不適切に使用したいでください、かかる改造、改変、復制等に上り生じた損害に問し、当社
	4. 当社表面を、主部または、部を同わり、以近、以及、後表、その他の小週時に使用のないでくたとい。がから以近、以及、後表寺により主のた損害に用り、当社 は、一切その責任を負いません。
	<ol> <li>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。</li> <li>標準水準: コンピュータ、OA機器、通信機器、計測機器、AV機器、</li> </ol>
	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)、交通制御(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、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、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損害が生じても、当社は一切その責任を負い ません。
	6. 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。
	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。
	8. 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。
	9. 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、(1)核兵器、化学兵器、生物兵器等の大量破壊兵器およびこれらを運搬することができるミサイル(無人航空機を含みます。)の開発、設計、製造、使用もし くは貯蔵等の目的、(2)通常兵器の開発、設計、製造または使用の目的、または(3)その他の国際的な平和および安全の維持の妨げとなる目的で、自ら使用せず、か つ、第三者に使用、販売、譲渡、輸出、賃貸もしくは使用許諾しないでください。
	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。
	10. お客様の転売、貸与等により、本書(本ご注意書きを含みます。)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、当社は一切その 責任を負わず、お客様にかかる使用に基づく当社への請求につき当社を免責いただきます。
	11. 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。
	12. 本資料に記載された情報または当社製品に関し、ご不明点がある場合には、当社営業にお問い合わせください。
	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において定義された当社の開発、製造製品をいいます。
1	

(Rev.3.0-1 2016.11)

# RENESAS

ルネサスエレクトロニクス株式会社

http://www.renesas.com

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。

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:https://www.renesas.com/contact/

■営業お問合せ窓口

© 2017 Renesas Electronics Corporation. All rights reserved. Colophon 5.0