

SmartRA 学習キット スタートアップマニュアル (始めにお読みください)

ルネサス エレクトロニクス社 RA マイコン搭載 HSB シリーズマイコンボード 評価キット

-本書を必ずよく読み、ご理解された上でご利用ください





注意事項	1
安全上のご注意	2
本書で説明する内容	4
1. 最初の注意点	5
2. 電源の投入	6
2.1. 電源供給方法	7
3. プログラムの実行	9
4. プログラムのコンパイル・ビルド	
4.1. プロジェクトのインポート	
4.2. プログラムのビルド	
5. マイコンボードへのプログラムの書き込み	
6. 付録	
6.1. USB シリアル変換 IC のドライバに関して	
6.2. 新規に e2studio のプロジェクトを作成する手順	
6.3. e2studio のパースペクティブ(ウィンドウ配置)	
6.4. エミュレータ(E2Lite)の接続(デバッグ)	
6.5. デバッグに関連する設定等	
6.5.1. 最適化 OFF	
6.5.1. メモリ内容の確認	
6.6. 自動変数のサイズ	
6.6. 自動変数のサイズ 6.7. sprintf を使ったフォーマット	
6.6. 自動変数のサイズ 6.7. sprintf を使ったフォーマット 取扱説明書改定記録	





注意事項

本書を必ずよく読み、ご理解された上でご利用ください

【ご利用にあたって】

- 本製品をご利用になる前には必ず取扱説明書をよく読んで下さい。また、本書は必ず保管し、使用上不明な点がある場合は再読し、よく理解して使用して下さい。
- 2. 本書は株式会社北斗電子製マイコンボードの使用方法について説明するものであり、ユーザシステムは対象ではありません。
- 3. 本書及び製品は著作権及び工業所有権によって保護されており、全ての権利は弊社に帰属します。本書の無断複 写・複製・転載はできません。
- 弊社のマイコンボードの仕様は全て使用しているマイコンの仕様に準じております。マイコンの仕様に関しましては 製造元にお問い合わせ下さい。弊社製品のデザイン・機能・仕様は性能や安全性の向上を目的に、予告無しに変更 することがあります。また価格を変更する場合や本書の図は実物と異なる場合もありますので、御了承下さい。
- 5. 本製品のご使用にあたっては、十分に評価の上ご使用下さい。
- 6. 未実装の部品に関してはサポート対象外です。お客様の責任においてご使用下さい。

【限定保証】

- 1. 弊社は本製品が頒布されているご利用条件に従って製造されたもので、本書に記載された動作を保証致します。
- 2. 本製品の保証期間は購入戴いた日から1年間です。

【保証規定】

保証期間内でも次のような場合は保証対象外となり有料修理となります

- 1. 火災・地震・第三者による行為その他の事故により本製品に不具合が生じた場合
- 2. お客様の故意・過失・誤用・異常な条件でのご利用で本製品に不具合が生じた場合
- 3. 本製品及び付属品のご利用方法に起因した損害が発生した場合
- 4. お客様によって本製品及び付属品へ改造・修理がなされた場合

【免責事項】

弊社は特定の目的・用途に関する保証や特許権侵害に対する保証等、本保証条件以外のものは明示・黙示に拘わらず 一切の保証は致し兼ねます。また、直接的・間接的損害金もしくは欠陥製品や製品の使用方法に起因する損失金・費用 には一切責任を負いません。損害の発生についてあらかじめ知らされていた場合でも保証は致し兼ねます。 ただし、明示的に保証責任または担保責任を負う場合でも、その理由のいかんを問わず、累積的な損害賠償責任は、弊 社が受領した対価を上限とします。本製品は「現状」で販売されているものであり、使用に際してはお客様がその結果に 一切の責任を負うものとします。弊社は使用または使用不能から生ずる損害に関して一切責任を負いません。 保証は最初の購入者であるお客様ご本人にのみ適用され、お客様が転売された第三者には適用されません。よって転 売による第三者またはその為になすお客様からのいかなる請求についても責任を負いません。 本製品を使った二次製品の保証は致し兼ねます。





製品を安全にお使いいただくための項目を次のように記載しています。絵表示の意味をよく理解した上で お読み下さい。

表記の意味



取扱を誤った場合、人が死亡または重傷を負う危険が切迫して生じる可能性が ある事が想定される

取扱を誤った場合、人が軽傷を負う可能性又は、物的損害のみを引き起こすが 可能性がある事が想定される

絵記号の意味

0	一般指示 使用者に対して指示に基づく行為を 強制するものを示します	\bigcirc	一般禁止 一般的な禁止事項を示します
	電源プラグを抜く 使用者に対して電源プラグをコンセ ントから抜くように指示します		一般注意 一般的な注意を示しています















本書では、キットの SmartRA-Base-Board とマイコンボード(HSBRA2L1F100)を最初に動かすところまでを説明し ます。

SmartRA-Base-Board とマイコンボードのハードウェア詳細に関しては、 「SmartRA 取扱説明書」(SMARTRX_REV_x_x_x_x.pdf) 「HSBRA2L1F100 取扱説明書」(HSBRA2L1F100_64_x.x.x.x.pdf)

を参照してください。

マイコンボードに触るのが初めてではないという方は、本書は読み飛ばして頂いても構いません。





1. 最初の注意点



本製品は、ケース等に入っている製品ではなく電子基板がむき出しの製品となっています。

マイコンボード基板には、静電気に弱い部品が搭載されていますので、特に冬場等湿度が低い場合は、金属の手 すりに触れるなどして人体に溜まっている静電気を逃がしてから、マイコンボードに触れるようにしてください。また、マ イコンボードの銀色の部分(電極)には触れないようにしてください。

また、マイコンボードは、絨毯や繊維質のものの上には置かない様にしてください。絨毯や繊維質のものは静電気 が発生しやすいためです。



金属板等電気が流れる素材の上にマイコンボードを置いた場合、ショート(本来電気が流れてはいけない場所に電気が流れる現象)を引き起こす事がありますので、金属板の上では使用しないでください。(なお、マイコンボードの電極部分が金属板と触れないよう、スペーサ等を挿入した場合は問題ありません)





注意

電源の極性及び過電圧には十分にご注意下さい

・ボードに電源を供給する場合は、複数個所からの電源供給を行わないで下さい。製品の破損、故障の原因となります。

・極性を誤ったり、規定以上の電圧がかかると、製品の破損、故障、発煙、火災の原因となります。

・ボード破損を避けるために、電圧を印加する場合には VCC=1.6~5V+0.5V の範囲になるようにご注意下さい。

電源をマイコンボードに接続する際は、

・電圧

・極性

·電流容量

に注意願います。

電圧は、1.6V~5V(最大 5.5V)までとなります。12V の AC アダプタを接続するといった事は、マイコンボードの破壊 につながりますので、手近にある AC アダプタを等の電源を使用する場合は、電圧が適合しているか確認の上、使用 する様にしてください。

電流容量としては、200mA 以上であれば十分です。電源装置から電源を供給する際は、まずは電流の最大値を 200mA 程度に設定してください。ショートが起こった際でも火花が散ったり、過剰な発熱が起こったりする事を抑止で きます。

AC アダプタ等電流の最大値を設定できないタイプの電源で使用することに関しては問題ありませんが、ショートが 起こった際大きな電流が流れる事がありますので、ショートには十分注意願います。

※モータ等消費電力の大きなデバイスを駆動する際は、500mA以上の電流容量が必要になります





2.1. 電源供給方法

電源供給は、以下に示す(1)~(4)の<u>いずれか、一通りの方法</u>で行ってください。

2 箇所以上の電源供給元があると、電源間のショートが起こりますので、電源供給元は 1 箇所となる様にしてください。



図 2-1 電源供給元

(1)USB ケーブル(USB-miniB コネクタ)[B-J4(*1)]

キット付属の USB ケーブルで、ベースボード J4(B-J4)を PC に接続してください。ベースボードの JP1(B-JP1)をショートに設定してください(出荷時状態)。

電圧は 5V(USB 規格による)、電流容量は最大 500mA となります(SmartRA-Base-Board 側に、500mA のポリヒ ューズ(*2)が付いています)。

※USB3.1(900mA max)のポートや USB-TypeA の充電アダプタ(12W, 2.4Amax 等)をつないだ場合でも、電流容 量は最大 500mA に制限されます

(*1)ベースボード(SmartRA-Base-Board)の J4 を B-J4 と表記します (*2)ポリヒューズは過剰な電流が収まった際、自動的に復帰するタイプのヒューズとなります

SmartRA 学習キット スタートアップマニュアル



7



(2)DC 電源コネクタ[M-J4(*1)]

マイコンボード J4(M-J4) DC 電源コネクタから電源供給してください(+1.6~5V)。



(*1)マイコンボード(HSBRA2L1F100)の J4 を M-J4 と表記します

(3)CAN コネクタ[M-J12]

マイコンボード J12(M-J12)の CAN コネクタからの給電も可能です。

CAN ネットワークを構築し、CAN の通信相手側に給電を行っている場合は、CAN ケーブルを挿す事により、本ボードへの給電が可能です。

(4)デバッガコネクタ[M-J7]

マイコンボード J7(M-J7)(14 ピンコネクタ)からの給電も可能です。

接続したデバッガから、給電を行う設定としてください。

E2 エミュレータ Lite を使用した場合は、供給電圧は 3.3V の選択のみとなります。この場合、LCD や CAN 等の機能を使用する事はできません。

E2 エミュレータを使用した場合は、供給電圧は 3.3V と 5V から選択が可能です。

どちらのエミュレータを使用した場合でも、電流容量は最大 200mA となります。





3. プログラムの実行

マイコンボードには、出荷時デモプログラムが書き込まれています。

デモプログラムを動作させる際は、電源電圧は 2~5V の範囲としてください。

※マイコンボード自体は 1.6V~で動作しますが、デモプロプログラムが動作し、LED が視認できるのが、2V~となります

【出荷時デモプログラムの内容】

・ベースボード B-SW1~4 を押すと B-LED0~3 が点灯

・ベースボード B-SW5(DIP-SW)の A~D が ON のとき、B-LED4~7 が点灯

・ベースボード B-K1, B-K2 にタッチしている間は、B-LED9, B-LED8 が点灯

・ベースボード B-K3 にタッチしている間は、マイコンボード M-D1 が点灯

・マイコンボード M-SW2 を押すと、(電源ランプ以外の)全 LED が点滅



図 3-1 ボード図





・UART(SCI9)に起動メッセージ表示

・LCD にメッセージ表示

プログラムを実行する際は、B-SW5を下側(RUN 側)に切り替え、M-J5のジャンパピンを抜いた状態で、電源を投入してください。

電源投入後スイッチとLED が連動しない場合は、以下を確認してください。

・M-D3 が点灯してない場合

M-D3 が点灯していない場合は、ボードの電源供給に問題があるものと考えられます。 極性、電圧、供給方法が適切か、ボードが金属板等でショートしていないかを確認してください。

・M-D3 が点灯している場合

B-SW が下側に切り替えられているかを確認してください。 M-J5 のジャンパを挿していないかを確認してください。

上記問題が無い場合は、ボードにデモプログラムが書き込まれていない可能性が考えられます。チュートリアルや 自作のプログラムを書き込んだり、書き込もうとした際、デモプログラムが消去されているものと考えられます。この場 合は、5章を参考に、出荷時に書き込まれているデモプログラムを書き込んでみてください。





4. プログラムのコンパイル・ビルド

ここで、開発環境を未インストールの場合は、開発環境のインストールを行ってください。

開発環境としては、ルネサスエレクトロニクス製、e2studio+FSP(FSP v2.4.0 以降)が使用できます。

e2studio+FSP は、ルネサスの Web 上でユーザ登録が必要ですが、フリーでダウンロード可能です。

e2studio は、IDE(統合開発環境)というアプリケーションで、RA マイコン向けのプログラムの開発やデバッグを行う ことのできるツールとなります。

FSP は、Flexible Software Package の略で、RA 向けのソフトウェアライブラリです。

e2studio は、単体と RA 向けの FSP がセットになっているものがあります。RA 向けには、FSP がセットになってい る方を選択してインストールしてください。(単体の e2studio は、コンパイラは別途導入が必要です。RA の FSP がセ ットになっているものは、コンパイラ(ARM-GCC)も同梱されています。)

プログラムのコンパイル・ビルド後には、RenesasFlashProgrammer(ver3.08以降)で、マイコンボードにプログラムの書き込みをおこないますので、合わせてダウンロード、インストールを行って頂きたく。

RA マイコン向けのプログラムは、C 言語かアセンブリ言語で開発が可能ですが、ここでは C 言語を使用してプログ ラムの開発を行います。

プログラムのビルドから書き込みまでの一連の手順を、サンプルのチュートリアルで説明します。





4.1. プロジェクトのインポート

e2studio を起動し、付属 CD 内の、

SOURCE¥TUTORIAL¥RA2L1_SMARTKIT_TUTORIAL0.zip

を、e2studioの環境にインポートしてください。

ファイルーインポートを選択

e	works	pace_RA -	e ² studio				
771	(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)	プロシ
۵,	新規(ファイ) ファイ) 最近((N) ルを開く(.). ル・システム のファイル	 からプロジェ	クトを開く	Alt+シフト+N > >	Configur	rations
	閉じる すべて	5(C) 【閉じる(L)		C	Ctrl+W Ctrl+シフト+W		
	保存(別名((S) 保存(A)			Ctrl+S		
R	すべて 前回	【保管(E) 保管した状	態に戻す(T)		Ctrl+シフト+S		
ľ	移動(名前)	(V) を変更(M).			F2		
<i>\$</i>	更新(行区)	(F) 切り文字の	変換(D)		F5 >		
Ð	印刷	(P)			Ctrl+P		
2	インボ	ぱート(I)		インポート			
4	エクス	ポート(O)					
	プロバ	°テイ(R)			Alt+Enter		
	ワーク 再開 終了/	スペースの5 /出口(X)	のり替え(W)		>		





図 インポート	_		×	
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	5	
 インボート・ウィザードの選択(5): フルタ入力 ◇ CMSIS Pack ◇ Rename & Import Existing C/C++ Project into Workspace ◇ Renesas CA78KOR (CS+) プロジェクト ◇ Renesas CC-RX/CC-RL (CS+) プロジェクト ◇ アーカイブ・ファイル ◇ フィル・システム ◇ フィル・システム ◇ フィル・システム ◇ フィル・システム ◇ フィル・システム ◇ B定 > C/C++ > Git > Oomph > Tracing > ML > インストール > テーム > 実行/デバッグ 				
(P) (P) <th (p)<="" <="" td=""><td></td><td>キャンセノ</td><td>ŀ</td></th>	<td></td> <td>キャンセノ</td> <td>ŀ</td>		キャンセノ	ŀ

既存プロジェクトをワークスペースへを選択「次へ」





😰 インポート			×
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(①: ●アーカイブ・ファイルの選択(A): D:¥SOURCE¥TUTORIAL¥RA2L1_SMART プロジェクト(P): ☑ RA2L1_SMARTKIT_TUTORIAL0(RA2L1_SMARTKIT_TUTORIAL0/) 		参照 参照 すべて選打 Rをすべて	(R) (R) R(S) 解除(D)
		更新(]	E)
オプション ✓ ネストしたプロジェクトを検索(<u>H</u>) ✓ プロジェクトをワークスペースにコピー(<u>C</u>) □ 完了次第、新しくインポートしたプロジェクトを閉じる(<u>o</u>) □ ワークスペースに既に存在するプロジェクトを隠す(<u>i</u>)			
ワーキング・セット ロワーキング・セットにプロジェクト を追加(<u>1</u>) ワーキング・セット(<u>0</u>):	/	新規(<u>₩</u> 選択(<u>E</u>))
(N) 次へ(N) > 終了(F)		キャン	セル

アーカイブ・ファイルの選択 側を選択

CD 内の SOURCE¥TUTORIAL¥RA2L1_SMARTRA_TUTORIAL0.zip を指定

(ファイルは CD から、PC のストレージに予めコピーしていた場合、PC のストレージ上のファイルを指定してください) 全て選択 を選択

終了





📴 workspace_RA - e² studio		
ファイル(<u>F</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>) R
🔦 🔯 🔳 🎋 デパッグ(B)	✓ C™ RA2L1_SMART	KIT_TUTORIAL0 Deb
⋬╺┦╺やぺぐっ・		
№ プロジェクト・エクスプローラー 🛛	🖻 😫 โ	7800
> 📂 RA2L1_SMARTKIT_TUTORIAL0 [Debug]		^
RA2L1_SMARTRA_DEMO		
Talan Talah		
RA2L1_SW_LED		

e2studio のプロジェクト・エクスプローラ上に「RA2L1_SMARTKIT_TUTORIAL0」が見えていれば、プロジェクトの インポートは成功しています。

4.2. プログラムのビルド

RA2L1_SMARTKIT_TUTORIAL0のプロジェクトを開いた状態とします。

(上記画面)プロジェクト名が「太字」になっている場合(プロジェクト名の先頭のフォルダが開いているアイコンになっている場合)、そのプロジェクトは開かれています。

混乱を防ぐため、その他のプロジェクトで開いているものがあれば、プロジェクトを閉じてください。

[参考]



左の画面は、プロジェクトが複数開いています。 ビルドしたいプロジェクト 「RA2L1_SMARTRA_TUTORIAL0」以外で 開いた状態となっているプロジェクトがあれば、 プロジェクト名で右クリックープロジェクトを閉じる を選択して、フォルダのアイコンが開いていない状態とし てください。 (もしくは、RA2L1_SMARTRA_TUTORIAL0 を右クリッ クして「無関係なプロジェクトを閉じる」という操作でも構い ません。複数のプロジェクトが開いていても一括で閉じる 事ができます。)

※<u>e2studio の操作に慣れないうちは、開いている状態の</u> プロジェクトは 1 つだけにしておく事を、強く推奨します





C	works	pace_RA -	e ² studio									
77	[,] イル(F)	編集(E)	ソース(S)	リファクタリング(T)) †	ビゲート(N)	検察	索(A)	プロ	ジェク	Ի(P)	F
Q	\$	۹	🏘 デバ	、ッグ(B)	~	C≊ RA2L	1_SM	ARTKI	T_TU	TORI	ALO [Deb
Ł	Build	* > •	* 🖓 🕶	⇒ - 12								
6	プロジェク	クト・エクス:	プロ - ラ- X	3		Į.	_ ₽	57	00			
>	📂 RA2	L1_SMAR	TKIT_TUT	ORIAL0 [Debug]							^	
	📋 RA2	L1_SMAR	TRA_DEMO)								
· ·	📋 RA2	L1_SW										
· ·	📋 RA2	L1 SW LE	D									

RA2L1_SMARTKIT_TUTORIAL0 のプロジェクトが開いている状態で、Build アイコン(トンカチのマーク)をクリック します。

📃 コンソール 🛛 🔝 問題 🁒 スマート・ブラウザー 🎄 デバッグ 🛷 検索 📋 メモリー CDT ビルド・コンソール [RA2L1_SMARTKIT_TUTORIAL0] 'Finished building: ../ra_gen/pin_data.c' . . "make -j6 all" terminated with exit code 2. Build might be incomplete. 14:57:45 Build Failed. 4 errors, 1 warnings. (took 561ms)

Build の操作が実行され、e2studio のコンソールに結果がでます。warning は無視しても良いものもありますが、 error は 0 にする必要があります。

[参考]

コンソールが画面に表示されないときは、



ウィンドウーパースペクティブーパースペクティブのリセット を行ってみてください。

画面の配置構成を変えてしまって、必要なものが表示されていない時に有効です。

HALA

SmartRA 学習キット スタートアップマニュアル

16



ビルドでエラーが出たので、エラーの内容を確認します。



プロジェクトの階層を展開(>アイコンをクリック)すると、

src

Debug

でエラー(×マーク)が出ています。

src の下の hal_entry.c がエラーになっているので、hal_entry.c をダブルクリックして開きます。右側にファイルが開 いてファイルの中身が表示されます。

hal_entry.cの34行目でエラーとなっており、エラー内容は;(セミコロン)がありませんという内容です。

34 行目を、

オリジナル led_num_on(i)

修正後 led_num_on(i);

の様に、行末に;(セミコロン)を追加してください。

エラーを直した後、再度ビルド(トンカチのアイコンをクリック)します。





🗐 コンソール 🛛 🔝 問題 🍓 スマート・ブラウザー 🎄 デバッグ 🛷 検索 📋 メモリー CDT ビルド・コンソール [RA2L1_SMARTKIT_TUTORIAL0] 8 1168 16f4 RA2L1_SMARTKIT_TUTORIAL0.elf 4700 5876 'Finished building: RA2L1_SMARTKIT_TUTORIAL0.srec' 'Finished building: RA2L1_SMARTKIT_TUTORIAL0.siz' 15:24:00 Build Finished. 0 errors, 0 warnings. (took 1s.518ms)

今度は、エラーが0になりました。

今回のケースでは、src と Debug でエラーとなっていましたが、ソース(src)にエラーがあるため、Debug でエラーが 出ていました。ソースのエラーを解消すると、Debug 側も消えました。





※FSP のバージョンアップに伴うエラーに関して

FSP のバージョンが変わると、

🔐 問題 🗙 📮 ユンソール 👒 スマート・ブラウザー 📪 スマート・マニュアル 🎄 デバッグ		7 8	
9 errors, 1 warning, 5 others			
記述/説明 ^	リソース	パス	ロケーシ
▼ ⁶ I5- (9項目)			
8 'BSP_CFG_CLKOUT_DIV' undeclared (first use in this function)	bsp_clocks.c	/RA2L1_SMARTKIT	行 1014
Issp_CFG_CLKOUT_SOURCE' undeclared (first use in this function); did you mean 'BSP_CFG_CLOCKS_SECURE'?	bsp_clocks.c	/RA2L1_SMARTKIT	行 1014
IBSP_CFG_CLOCK_SOURCE' undeclared (first use in this function); did you mean 'BSP_CFG_CLOCKS_SECURE'?	bsp_clocks.c	/RA2L1_SMARTKIT	行 663
8 'BSP_CFG_ICLK_DIV' undeclared (first use in this function)	bsp_clocks.c	/RA2L1_SMARTKIT	行 115
Image: BSP_CFG_PCLKB_DIV undeclared (first use in this function)	bsp_clocks.c	/RA2L1_SMARTKIT	行 127
Image: BSP_CFG_PCLKD_DIV' undeclared (first use in this function)	bsp_clocks.c	/RA2L1_SMARTKIT	行 117
😢 'BSP_CFG_XTAL_HZ' undeclared (first use in this function)	bsp_clocks.c	/RA2L1_SMARTKIT	行 708
@ make: *** [ra/fsp/src/bsp/mcu/all/subdir.mk:55: ra/fsp/src/bsp/mcu/all/bsp_clocks.o] Error 1	RA2L1_SMAR		
😣 make: *** Waiting for unfinished jobs	RA2L1_SMAR		
> & 警告 (1項目)			
▶ i 情報(5項目)			

BSP_CFG で始まるエラーが出る事があります。

この場合は、プロジェクトの FSP バージョンのバージョンアップが必要になります。

	_
V 🔂 RA2L1_SMARTKIT_TUTORIAL0	
> 🔊 Includes	
> 🔂 ra	
> 😕 ra_gen	
> 🚰 src	
> 🗁 Debug	
> 🗁 ra_cfg	
> 🗁 script	
🎡 configuration.xml	
R7FA2L1AB2DFRpincfg	
📄 ra_cfg.txt	
RA2L1_SMARTKIT_TUTORIAL0 Debug_Flat.launch	
⑦ Developer Assistance	

Configuration.xml をダブルクリック

📴 e2 s	itudio	×
<u> </u>	FSP version 2.4.0 is not installed. Version 4.1.0 will be selected.	OK

バージョンが変わる旨のダイアログが出るので OK を押す





*[RA2L1_SMARTKIT_TUTORIAL0] FSP Configuration ×	
Pin Configuration	Generate Project Content
Select Pin Configuration	🖫 Export to CSV file 🛛 🖺 Configure Pin Driver 🖉 ings
R7FA2L1AB2DFP.pincfg Manage configurations	Generate data: g_bsp_pin_cfg

Generate Project Content を押す。

📴 Ger	nerate Project Content	×
\bigcirc	Configuration must be saved before generating	project content.
	Proceed with save and generate?	
Alw	ays save and generate without asking	
		続行(<u>P)</u> キャンセル

続行を押す。

上記で、プロジェクトの FSP バージョンの更新作業が終了しましたので、ビルド操作を続けて行ってください。





5. マイコンボードへのプログラムの書き込み

プロジェクト・エクスプローラ(左側の表示ペイン)で、

RA2L1_SMARTKIT_TUTORIAL0 を右クリックープロパティを開くと

📵 プロパティ: RA2L1_SMARTKIT_T	UTORIAL0 -		×
	リソース	⇔ • ⇔	₩ 00
> <mark>リソース</mark> > C/C++ ビルド > C/C++ 一般 > MCU	パス(<u>P</u>): /RA2L1_SMARTKIT_TUTORIAL0 タイプ(<u>Y</u>): プロジェクト ロケーション(<u>L</u>): C:¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_SMARTKIT_1		2
Renesas QE Task Tags > Validation ビルダー プロジェクト・ネーチャー プロジェクト参照	最終変更日時(<u>M</u>): 2021年4月26日 14:36:48 テキスト・ファイル・エンコード(<u>T</u>) ④ コンテナーから継承(<u>J</u>) (UTF-8) ○ その他(<u>O</u>): UTF-8 □ 派生リソースのエンコードを別途保管(<u>S</u>)	1	
実行/テバック設定	新規テキスト・ファイルの行区切り文字(E) ④ コンテナー (Windows) から継承(E) 〇 その他(<u>H</u>): Windows >		
	デフォルトの復元(]	D 適用()	L)
?	適用して閉じる	キャンセノ	V

プロジェクトが格納されている場所が判ります。

(インストール時に変更していなければ、c:¥Users¥ユーザ名¥e2_studio¥workspace になると思います)





フォルダ名の右のアイコンをクリックすると、Windows のエクスプローラが開きます。

IIII マーム 共有 表示 C¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_SMARTKIT_TUTORIAL0¥Debug ファイル ホーム 共有 表示								
← → × ↑ 🔒 « ドキュメント → e2_studio → wa	orkspace_RA > RA2L1_SMARTKIT_TUTORIAL0	> Debug > V	ට 🔎 Debug(の検索				
🚽 ባለማካ ምሳቱ አ	名前 ^	更新日時	種類	サイズ				
Desktop	ra	2021/04/26 14:36	ファイル フォルダー					
↓ ダウンロード 🦻	src	2021/04/26 15:25	ファイル フォルダー ファイル フォルダー					
E F#1X7F	makefile	2021/04/26 15:23	ファイル	5 KB				
📰 ピクチャ 🔊	makefile.init	2021/04/26 14:57	INIT ファイル	2 KB				
RA2L1	objects.mk	2021/04/26 14:57	Makefile	1 KB				
share	RA2L1_SMARTKIT_TUTORIAL0.elf	2021/04/26 15:24	ELF ファイル	323 KB				
src	RA2L1_SMARTKIT_TUTORIAL0.elf.in	2021/04/26 15:24	IN ファイル	2 KB				
src	RA2L1_SMARTKIT_TUTORIAL0.map	2021/04/26 15:24	Linker Address Map	125 KB				
310	RA2L1_SMARTKIT_TUTORIAL0.srec	2021/04/26 15:24	SREC ファイル	14 KB				
 OneDrive 	B sources.mk	2021/04/26 15:23	Makefile	1 KB				

※このハードコピーの例では、デフォルトとは別な場所にワークスペースを作成しています(フォルダは初期設定や起 動時の設定により異なります)

プロジェクトフォルダの、Debugの下に、

RA2L1_SMARTKIT_TUTORIAL0.srec

というファイルが生成されているはずです。このファイルが今回欲しいものとなります。(プログラムをビルドして得た 生成物)

このファイルを、マイコンボード(HSBRA2L1F100)に書き込みます。



(1)接続

(1)ジャンパを確認

B-JP2:ショート M-J5:オープン





<u>図 5-1 書きこみ</u>時の接続

ジャンパ B-JP1(USB から電源を供給する場合)ショートに設定する ジャンパ B-JP2 ピンともショート(横の2端子がジャンパで接続される形となります) マイコンボード上のジャンパ M-J5 はオープン B-SW5 は、WRITE 側(上側)に切り替える ベースボード B-J4 と PC を接続する

K2 R15

(※USB から電源を供給しない場合は、マイコンボードに電源を投入する)



(2)RenesasFlashProgrammerの起動とマイコンボードへの接続

🜠 Renesas Flash Programmer V3.08.00 (無償版)		_		×
<u>ファイル(F)</u> へルプ(H)				
新しいプロジェクトを作成(N)				
プロジェクトを開く(O)				
プロジェクトを保存(S)				
イメージファイルを保存(1)				
ファイルチェックサム(C)				
ファイルパスワード設定(P)				
1 ZZZ.rpj		参照	摇(B)	
2 RA6M4_144_TEST.rpj				
3 RA6M4_DEMO.rpj				
4 RA6M4_144_TEST.rpj				
終了(X)				
スタート(S)				
Renesas Flash Programmer V3.08.00 [1 Oct 2020] (無(賞版)				
	ステータス	とメッセー		P(C)

RenesasFlashProgrammer を起動し、

ファイルー新しいプロジェクトを作成

📕 新しいプロジェクトの作用	艾	_		×
プロジェクト情報				
マイクロコントロ <i>ーラ(<u>M</u>):</i>	RA ~			
プロジェクト名(<u>N</u>):]		
作成場所(<u>F</u>):	C:¥Users¥win64-7¥Documents¥Renesas Flash Pr		参照(<u>B</u>)]
通信 ッール(<u>T</u>): COM port <u>ッール詳細(D</u>)	 ✓ インタフェース(①: 2 wire UART 〜 番号: COM5 			
	接続(_)		キャンセノ	NC)





・マイクロコントローラ

RA

を選択する。

・プロジェクト名

任意の名称を入力(ここでは、TUTORIAL0)

・ツール

COM port

に変更してください

・ツール詳細

🜠 ツール詳細 (COM port)	_		×
ツール選択 リセット設定			
COM5 : Prolific USB-to-Serial Comm	n Port		
	<u>O</u> K	キャンセ	11(<u>C</u>)

PC に複数の COM ポートが存在する場合、マイコンボードに接続している COM ポートを選択してください。 SmartRA-Base-Board では「Prolific USB-to-Serial Comm Port」という表示となります。 (ここでは、COM5を選んでOKを押してください。)



SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



🕻 新しいプロジェクトの作用	Σ.	_		×
プロジェクト情報				
マイクロコントローラ(<u>M</u>):	RA ~			
プロジェクト名(<u>N</u>):	TUTORIAL0]		
作成場所(<u>F</u>):	C:¥Users¥win64-7¥Documents¥Renesas Flash Pr		参照(<u>B</u>)	
通信 ツール(T): COM port ツール詳細(D)	 ✓ インタフェース(①: 2 wire UART ✓ 番号: COM5 接続(②) 		キャンセ	11(C)

接続

を押す

🕻 Renesas Flash Programmer V3.08.00 (無償版)	-	-		×
ファイル(<u>F</u>) デバイス情報(D) ヘルプ(<u>H</u>)				
操作 操作設定 ブロック設定 接続設定 ユニークコード				
プロジェクト情報 現在のプロジェクト: TUTORIALOrpj マイクロコントローラ: RA				
プログラムファイル		参照.	(<u>B</u>)	
フラッシュ操作 消去 >> 書き込み >> ベリファイ スタート(S)				
≓15/7.4 · PA				
アハイス石:CM Device Code:06 Firmware Version:V1.0 Code Flash 1(アドレス:0x00000000、サイズ:256 K、消去サイズ:2 K) Data Flash 1(アドレス:0x40100000、サイズ:8 K、消去サイズ:1 K) Config Area(アドレス:0x01010010、サイズ:36、消去サイズ:0)				~
ツールから切断します。 抹作が成功しました。				~
	ステータスと	メッセージ	ッのクリア	7(<u>C</u>)

操作が成功しました、という表示となれば OK です。





kenesas Flash Programmer V3.08.00 (無償版)	_	×
ファイル(F) ヘルプ(H)		
操作		
プロジェクト情報 現在のプロジェクト: マイクロコントローラ: プログラムファイル	参照(<u>B</u>)	
^{フラッシュ操作} スタート(<u>S</u>)	異常終了	
ツールに接続します。		^
Igen ツール:COM port (COMb), インタフェース:2 wire UART ターゲットデバイスに接続します。 ツールから切断します。 <mark>エラー(E3000105): デバイスから応答がありません。</mark> <mark>操作は失敗しました。</mark>		~

上記の様なエラーとなった場合は、マイコンボード上のリセットスイッチ(M-SW1)を押して、再度「新しいプロジェクト の作成ウィンドウの"接続"ボタン」を押してください。

エラーが解消されない場合は、

•B-SW6 の方向

COM ポート番号の選択

等の設定を確認してください。

なお、「ツールとの接続に失敗しました」というエラーの場合は、当該 COM ポート(上記例では COM5)で別ウィンド ウで端末が開いていないかを確認してください。端末が開いている場合は、その端末を一旦閉じてください。



SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



Nenesas Flash Programmer V3.08.00 (無償版)	-	_		×
ファイル(<u>E</u>) デバイス情報(<u>D</u>) ヘルプ(<u>H</u>)				
操作 操作設定 ブロック設定 接続設定 ユニークコード				
プロジェクト情報 現在のプロジェクト: TUTORIAL0rpj マイクロコントローラ: RA プログラムファイル C:¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_SMARTKIT_ CRC-32:	TUTORIA 5874 1984		<u>R(B)</u>	
フラッシュ抹作				
消去 >> 書き込み >> ベリファイ				
スタート(<u>S</u>)				
デバイス名:RA Device Code:06 Firmware Version:V1.0 Code Flash 1 (アドレス:0×00000000、サイズ:256 K、消去サイズ:2 K) Data Flash 1 (アドレス:0×40100000、サイズ:8 K、消去サイズ:1 K)				^
Config Area (アドレス: 0×01010010、サイズ: 36、消去サイズ: 0)				
ツールから切断します。 操作が成功しました。				~
	ステータス	とメッセー	-ジのクリア	7(<u>C</u>)

参照ボタンを押し、書き込む sreq ファイルを選択してください。(プロジェクトフォルダ¥Debug の中)

マイコンボード上のリセットスイッチ(M-SW1)を押してください。

スタートボタンを押してください。



SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



I Renesas Flash Programmer V3.08.00 (無償版)		-	×
ファイル(<u>F</u>) デバイス情報(<u>D</u>) ヘルプ(<u>H</u>)			
操作 操作設定 ブロック設定 接続設定 ユニークコード			
プロジェクト情報 現在のプロジェクト: TUTORIAL0rpj マイクロコントローラ: RA プログラムファイル C:¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_SMARTKI	T_TUTORIA	参照(<u>B</u>)	
CRC-32	2:58741984		
フラッシュ操作			
消去 >> 書き込み >> ベリファイ			
スタート(<u>S</u>)	Œ	常終了	
[Code Flash 1] 0×00000000 - 0×00001247 サイズ:4.6 K [Config Area] 0×01010018 - 0×01010033 サイズ:28			^
ベリファイを実行します。 [Code Flash 1] 0×00000000 - 0×00001247 サイズ:4.6 K [Config Area] 0×01010018 - 0×01010033 サイズ:28			
ツールから切断します。 操作が成功しました。			
			*

正常終了、「操作が成功しました」と表示されれば OK です。

マイコンボードにサンプルプログラムの書き込みは成功しています。

(エラーとなった場合は、マイコンボードのリセットスイッチ(M-SW1)を押して、再度スタートボタンを押してください)

(4)プログラムの起動

B-SW6を下側(RUN)に切り替えてください。

マイコンボードのリセットスイッチ(M-SW1)を押してください。

LED が B-LEDO 側から点灯を始めた場合は、書き込んだプログラムが実行されています。

※TUTORIALOのプログラムは、出荷時に書き込まれているデモとは別な動作となります





いかがでしょうか。問題なく、プログラムのビルドと書き込みは成功したでしょうか。

具体的なプログラムのソースコード等は、チュートリアル編のマニュアルで説明します。

本マニュアルでは、開封後の電源投入。e2studio 起動から、RenesasFlashProgrammer を使用したプログラムの 書き込みまでを足早に説明しました。

チュートリアルのマニュアルは複数に分かれており、動作させたいプログラムのチュートリアルに従いプログラミング を始める事ができる様になっています。

(必ずしもチュートリアル1から順に進める必要はありません、使用したい機能のチュートリアルを参照してください。)





6. 付録

6.1. USB シリアル変換 IC のドライバに関して

SmartRA-Base-Board を PC に接続した際、ハードウェアの認識が自動的に行われなかった場合は、ドライバのインストールが必要です。

上記のボードには、USB シリアル変換 IC として、prolific 社製、PL2303HXD が搭載されています。

ドライバのダウンロードは、prolific Web

http://www.prolific.com.tw/

から、下記を辿って、ダウンロード願います。

Products Application

SIO(Smart-IO)

USB to UART/Serial/Printer

PL2303HXD

PL2303 Windows Driver





3	workspace_RA -	e² studio							
ファイ	⁽ ル(F) 編集(E)	ナビゲート(N)	検索(A)	プロジェクト(P)	Rene	sas Views	実行(R)	ウィンドウ(W	/) ヘルプ(H)
	新規(N)			Alt+シフト+N		<u> </u>	Project		on:
	ファイルを開く(.)					9 プロジェ	:クト(R)	C またに	よC++ プロジェクトを新規作成
	ファイル・システム	からプロジェクトす	と開く			9 サンプル	/(X)		
	東虹のファイル				- c	9 その他(O)	Ctrl+N	
	閉じる(C)			Ctrl+W	Г				1
	すべて閉じる(L)			Ctrl+シフト+W					
	17								

e2studio の

「ファイル」「新規」「C/C++ Project」 を選択。



「Renesas RA」「Renesas RA C/C++ Project」を選択して、「次へ」。





Renesas RA C/C++ Project			×
Renesas RA C/C++ Project Project Name and Location			2
Project name RA2L1_TEST_PROJECT ジープフォルト・ロケーションの使用(D) ロケーション()): CF¥Users¥win64-7¥Documents¥e2_studio¥workspace_RA¥RA2L1_TEST_PROJECT ファイル・システムを遅択(Y): デフォルト		参照(<u>R</u>)
You can download more Renesas packs here			
(?) (N) > 終了(1))	キャンセ	2.11

Project name の欄に「適当な名称」を設定し「次へ」。

📴 Renesas RA	A C/C++ Project		— 🗆 X]
Renesas RA C Device and To	C/C++ Project		Ď	
Device Selecti FSP Version: Board: Device:	on 2.2.0 ~ Custom User Board (Any Device) ~ R7FA2A1AB3CFM	- Board Description	_	
Language:		RA2 A RA2A1 RA2A	RA2L1 - 100 Pin > RA2L1 - 80 Pin > RA2L1 - 64 Pin > RA2L1 - 64 Pin >	R7FA2L1AB3CFM R7FA2L1A93CFM R7FA2L1AB2DFM R7FA2L1AB2DFM
Toolchains GNU ARM Er 9.2.1.2019102	mbedded 5 ~	Debugger J-Link ARM	v	
?		< 戻る(<u>B</u>) 次へ(<u>N</u>) >	終了(E) キャンセル	-

Device 選択の「…」ボタンを押し、「RA2」「RA2L1」「RA2L1-64Pin」「R7FA2L1AB2DFM」を選択。



🗐 Renesas RA C/C++ Proje	ect		_		×
Renesas RA C/C++ Proje Device and Tools Selection	ct				\$
Device Selection FSP Version: 2.2.0 Board: Custom Use Device: R7FA2L1AB Language: C O C4	r Board (Any Device) 2DFM	Board Description Device Details TrustZone Pins Processor	No 64 cortex-m23		
Toolchains GNU ARM Embedded 9.2.1.20191025	v	Debugger J-Link ARM None E2 (ARM) <u>E2 Lite (ARM)</u> J-Link ARM			~
?		< 戻る(<u>B</u>) 次	∧(<u>N</u>) > 終了(E)	キャンセ	IL

Debuggerを使用するデバッガに合わせて変更(例えば、「E2 Lite(ARM)」)。「次へ」。

Renesas RA C/C++ Project				×
Renesas RA C/C++ Project Build Artifact and RTOS Selection				\$
Build Artifact Selection Executable Project builds to an executable file Static Library Project builds to a static library file Executable Using an RA Static Library Project builds to an executable file Project uses an existing RA static library project 	RTOS Selection FreeRTOS FreeRTOS No RTOS			~
?	< 戻る(<u>B</u>) 次へ(<u>N</u>) > 終了(<u>F</u>)	キャンセ	IL

RTOS を使用しない場合は、「No RTOS」を選択(任意)。「次へ」。



HOHULO



📴 Renesas RA	C/C++ Project			×
Renesas RA C	/C++ Project		_	\diamond
Project Templa	te Selection			2
Project Templ	ate Selection			
0	Bare Metal - Blinky			
	Bare metal FSP project that includes BSP and will blink LEDs if available. This project will initialize clock the C runtime environment.	s, pins, s	tacks, an	Ы
	[Renesas.RA.2.2.0.pack]			
•	Bare Metal - Minimal			
	Bare metal FSP project that includes BSP. This project will initialize clocks, pins, stacks, and the C runtin [Renecad R4 2 2 0 pack]	ne enviro	nment.	
	[nen-sisheren]			
Code Generat	ion Settings			
✓ Ose Kellesa	s coue romatter			
?	< 戻る(<u>B</u>) 次へ(<u>N</u>) > 終了(<u>E</u>)		キャンセ	μ

サンプルコードを空にしたい場合には、「Minimal」側を選択(任意)。「終了」。

※Blinkyを選ぶとLED が点滅する様なコードが生成されます。

workspace_RA - RA2L1_TEST_PROJECT/configuration.xml - e ² st
ファイル(F) 編集(E) ナピゲート(N) 検索(A) プロジェクト(P) Renes
🔦 🏘 🔳 🎋 デバッグ(B) 🗸 🕞 RA2L1_TES
🔁 プロジェクト・エクスプローラー 🛛 🛛 🕒 😫 🍸 🖇
TRA2A1_E2
Tala RA2L1_CANST
✓ [™] RA2L1_TEST_PROJECT
> 🔊 Includes
> 😂 ra
> 🚑 ra_gen
> 📇 src
> 🤁 ra_ctg
> 🔁 script
R7EA2L1AB2DEM pincfa
■ RA2L1 TEST_PROJECT Debug_Elat launch
2 Developer Assistance
BA6T1 100 TEST
RA6T1_BLMKIT_TUTORIAL1
TA6T1_CANST
TRAGT1_E2

プロジェクト・エクスプローラ上に、作成したプロジェクト(ここでは、RA2L1_TEST_PROJECT)が見えれば、プロジ ェクトの作成は成功しています。





プロジェクト作成後に、使用するマイコンの機能に合わせて FSP の設定等を行い、コード生成やプログラムのビル ド、ビルドで生成されたバイナリ(srec ファイル)の書き込みという流れとなります。



SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



6.3. e2studio のパースペクティブ(ウィンドウ配置)

Х | 愛|| 袋||| 袋 マ 💁 マ 🔍 マ 徳 🗰 💷 😭 🖏 🥔 🥖 Q 🛛 😭 🗌 🖬 C/C++ 🎆 FSP Configuration 🎄 デバッグ 🖏 CapTouchモニタ RA, RL 78 (QE)

(e2studio のウィンドウの右上部分)

e2studio(eclipse)では、ウィンドウの配置が ・プログラム作成 ・デバッグ 等で、切り替える機能を持っています。デフォルトでは、 「C/C++」:ソース編集 「FSP Configuration」:FSP の設定 「デバッグ」:デバッグ の3つの切り替えが最初からできるはずです。

QE for CapTouch をインストールすると、 「CapTouch モニタ RA,RL78(QE)」:タッチキーミドルウェア が追加されると思います。

例えば、FSP の設定変更を行う際は、「FSP Configuration」が選択されていないと設定ウィンドウが表示されなかったりしますので、必要なウィンドウが表示されていないという場合は、適切なパースペクティブを選択しているかを確認してください。

また、色々いじっているうちに、必要なウィンドウを閉じてしまい、どこで開くのかが判らなくなる事もあります。その場合は、



パースペクティブのリセットを試してみてください。(非表示となっているウィンドウが復活します)





また、欲しいウィンドウが表示されていないという場合は、

) ウィンドウ(W) ヘルプ(H)		
新規ウィンドウ(N) Tディター >	• 🗐 🕼 🕸 • 🔦 • 📃 🛛 🕸 💸	‡* ▼ 💁 ▼ 🔍 ▼ 🕸 [
	٩	🗄 🗄 🔂 C/C++ 🔅 F
ビューの表示(V) >	症 C/C++ プロジェクト	
パースペクティブ(R) >	CapTouchメイン/センサ・チューナ RA, RL78 (QE)	
ナビゲーション(G) >	Documents	
記(中/D)	アウトライン	Alt+シフト+Q,O
题(P)	E コンソール	Alt+シフト+Q,C
	֎ァ スマート・ブラウザー	
	👒 スマート・マニュアル	
	& ターミナル	
	🙇 ୭スク	
	・ ビルド・ターゲット	
	🔁 プロジェクト・エクスプローラー	
	□ プロパティー	
	🔗 検索	Alt+シフト+Q,S
	🔍 最適化アシスタント	
	▲ 組み込みブラウザー	
	副題	Alt+シフト+Q,X
	顧問題の詳細	
	その他(O)	Alt+シフト+Q,Q

ウィンドウ – ビューの表示

の中、もしくは



Renesas Views

の下にある事が多いですので、選択して表示させるようにしてください。





6.4. エミュレータ(E2Lite)の接続(デバッグ)



マイコンボード J7(M-J7)の 14P コネクタに、ルネサス E2 エミュレータ Lite を接続します。マイコンボード上の J8(M-J8)ジャンパは、右側(2-3)ショートに設定してください。ベースボードの SW6(B-SW6)は、RUN 側である必要が あります。

RA2L1_NEW_PROJECT RA2L1_PUSHSW_TEST RA2L1_RTC_TIMER RA2L1_SCI RA2L1_SCI RA2L1_SCI	\$	更新(F) プロジェクトを閉じる(S) 無関係なプロジェクトを閉じる(U)	F5			
RA2LT_SCI_TESTT RA2LT_SCI_TESTT RA2LT_SCI_TEST2 RA2LT_SCI_TEST2		ビルド・ターゲット	>			
		ビルド構成	>			
> 端 ハイナリー > 剤 Includes	0	実行(R)	>			
> 🔑 ra	*	デバッグ(D)	>	C×	1 GDB Simulator Debugging (RH850)	1
> 🔑 ra_gen		ローカル履歴から復元(Y)		C ×	2 Renesas GDB Hardware Debugging	
> 🔑 src		MISRA-C	>	C ×	3 Renesas Simulator Debugging (RX, RL78)	
> 👝 Debug		C/C++ Project Settings	Ctrl+Alt+P	C	4 ローカル C/C++ アプリケーション	
> Calcing		Save build settings report			デバッグの進动/P)	諫 (
🔅 configuration.xml		Change Device		-	アパファ の (高)((0)	
R7FA2L1AB2DFP.pincfg	*	C/C++ コ−ド解析を実行				
ia_cfg.txt RA2L1 SMARTKIT TUT		Team	>	L		
> ⑦ Developer Assistance		Compare With	>			
TA2L1_SMARTRA_DEMO		System Explorer				

プロジェクトエクスプローラでプロジェクトを右クリックして、デバッグ-デバッグの構成を選択します。





※この時、複数のプロジェクトが開かれている時は、不要なプロジェクトを閉じ、「開かれているのはプロジェクト1つ だけ」とする事を推奨します。



赤線の「プロジェクト名_Debug_Flat」を選択

※複数のプロジェクトが開いている時は複数の「Debug_Flat」の項目が見えますので注意してください



SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



(型) デバッグ構成	- 0	×
構成の作成、管理、および実行		ñ
 ○ ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	名前(1): RA2L1_SMARTKIT_TUTORIAL0 Debug_Flat ● メイク 登 Debugger ▶ Startup 二 共通(2) 指 ソース プロジェクト(P): RA2L1_SMARTKIT_TUTORIAL0 参照(B) C/C++ アブリケ-ション: Debug/RA2L1_SMARTKIT_TUTORIAL0.elf 変数(U) プロジェクトの検索(H) 参照(R) 起動前に必要に応じてビルド Build Configuration: Use Active ○ 自動ビルドを有効にする ○ 自動ビルドを有効にする ○ 自動ビルドを有効にする	
15 項目のうち 13 項目がフィルターに一致	前回保管した状態に戻す(⊻) 適用(⊻)	
?	デバッグ(<u>D</u>) 閉じる	

Debugger タブを選択

(2) デバッグ構成	- D X
構成の作成、管理、および実行	The second se
 「ア ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	名前(山): RA2L1_SMARTKIT_TUTORIALO Debug_Flat メイン 弥 Debugger ▶ Startup 二 共通(① シ ソース) Debug hardware: E2 Lite (ARM) ▼ Target Device: R7FA2L1A8 … GDB Settings Connection Settings デパッグ・ツール設定 GDB 接続設定: ● ローカル GDB サーパーを自動起動 ホスト名または IP アドレス: localhost ○ リモート GDB サーパーを自動起動 ホスト名または IP アドレス: localhost ○ リモート GDB サーパーへ接続 GDB ポート番号: 61234 GDB GDB GDB GDB Jマンド: arm-none-eabi-gdb ③ Step Mode Additional GDB Server Arguments
15 項目のうち 13 項目がフィルターに一致	川林官した4人間に戻り(⊻) 2月代(⊻)
?	デバッグ(<u>D</u>) 閉じる

E2Lite(ARM)を選択、Connection Settings タブを選択。





(国) デバッグ構成		_	
構成の作成、管理、および実行			Ť.
🗋 🖻 🐼 🗎 🗶 🖻 🏹 🗸	名前(N): RA2L1 SMARTKIT TUTORIAL0 Debug Flat		
7ィルタ入力	■ X/2 微 Debugger ト Startup 同 共通(の) 語 ソース		
 C C/C++ アプリケーション C C/C++ リモート・アプリケーション E ASE Script 	Debug hardware: E2 Lite (ARM) V Target Device: R7FA2L1AB		
GDB OpenOCD Debugging	GDB Settings Connection Settings デバッグ・ツール設定		
C [*] GDB Simulator Debugging (RH850)	✓ 2□ック		
C GDB ハードウェア・デバッギング	メイン・クロック・ソース	内部クロック	~
Java アブリケーション	外部クロック入力周波数 (MHz)		
I Java アフレット	内蔵フラッシュ・メモリー書き換え時にクロック・ソースの変更を許可する	はい	×
C Kenesas GDB Hardware Debugging	✓ ターゲット・ボードとの接続		
C* RA2L1_SMARTKIT_TUTORIAL0 Debug_Flat	エミュレーター	(Auto)	
C [*] Renesas Simulator Debugging (RX, RL78)	タイプ	SWD	×
日 リモート Java アノリケーション	接続速度 (kHz)	Auto	×
□ ■ 起動グループ	✓ 電源		
	エミュレーターから電源を供給する (MAX 200mA)	いいえ	~
	供給電圧 (V)	3.3	\checkmark
	◇ 接続		
	接続時にリセット状態を維持する	はい	¥
	IDコード (バイト単位)	FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF	
	低消費電力モードを使用する	はい	¥
	セキュア領域 /非セキュア領域の境界を設定する	いいえ	\sim
15 項目のうち 13 項目がフィルターに一致		前回保管した状態に戻す(以) 適	1用(⊻)
?		デバッグ(<u>D</u>)	閉じる

エミュレータから電源を供給する「いいえ」を選択

設定だけを行いたい場合は、「適用」「閉じる」 (そのままエミュレータ接続を行う場合は、「デバッグ」を)押す。

※E2Lite からボードに給電を行う事も可能です。その場合は、他から電源を供給しない様にしてください。 (E2Lite からの給電は、3.3V に限られますので、LCD 等 5V が必要なプログラムの動作は確認できません。)

※E2を使用する場合は、5Vの供給も可能です。E2使用時は、マイコンボード上のジャンパ(M-J8)を左側(1-2)ショ ートに設定してください

📴 workspace_RA -	e ² studio		
ファイル(<u>F</u>) 編集(<u>E</u>)	ソース(<u>S</u>)	リファクタリング(T) ;
🐔 🔅 🔳	参 デバ	、ッグ(B)	\sim

デバッグの開始 デバッグの終了

デバッグの開始と終了は、虫のアイコンと、赤四角のアイコンで行います。





エミュレータ接続アイコンをクリックした場合

📴 パーフ	パクティブ切り替えの確認 ×					
\bigcirc	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、アプリケーションのデバッグをサポートするように設計されていま これには、デバッグ・スタック、変数、お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ます。						
	このパースペクティブを開きますか?					
□ 常にこの設定を使用する(<u>R</u>)						
	切り替え(<u>S</u>) いいえ(<u>N</u>)					

上記メッセージが出た場合は、「切り替え」で良いと思います。

🝘 workspace_RA - RA2L1_SMARTKIT_TUTORIAL0/ra_gen/main.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) ウインドウ(W) ヘルプ(H)							
本 デバッグ(B) マ RA2L1_SMARTKIT_TUTORIALO Deb マ 禁 ご マ □ □ □ ▶ ○ ▼ □ □ ■ № 3. つ □ □ i→ 志 ② □ □ i→ 志 ○ □ □ i→ □ □ □ i→ □ □ i→ □ □ □ i→ □ □ □ i→ □ □ i→ □ □ □ □							
* デバッグ 23 ■ 20 10 20 20 20 20 20 20 20 20 20 20 20 20 20	€ startup.c 1 2 3 4 000006590 5 00000694 6 0000698 7 0000069a 8	<pre> @ main.c \lambda /* generated main source file - d #include "hal_data.h" @ int main(void) 0 { hal_entry (); 8 return 0; a } </pre>					

プログラムの実行は、上記のアイコンで進めます。

ボードの電源を落とす前に、デバッグを終了させてください。





6.5. デバッグに関連する設定等

6.5.1. 最適化 OFF

FSP の環境では、デフォルトでは最適化 ON(-O2)です。最適化 ON の場合、変数がメモリに割り当てられなかったりして、確認したい時に値が見られない事があります。

また、ソースコードの順番と、実行順が入れ替わっていて、プログラムが追いかけにくいケースがあります。

そこで、最適化を OFF にして、デバッグを行いたいというケースが少なからずあります。

🕲 workspace_RA - R		新提(N)	-
ファイル(F) 編集(E) :		次ヘジャンプ(1)	-
		新現ワイントワで開く(N)	
		表示万法(W)	Alt+シフト+W >
陷 プロジェクト・エクスプロ		ローカル・ターミナルで表示	>
TA2A1_E2	Ð	コピー(C)	Ctrl+C
RA2L1_ADC	Ē	貼り付け(P)	Ctrl+V
RA2L1_CANST	×	削除(D)	削除
BA2L1_CISU		ソース	>
V RAZLI DEBUG		移動(V)	
> 🔊 Includes		名前を変更(M)	F2
> 😕 ra		A will be as	
> 🔑 ra_gen		1ンボート(I)	
> 📇 src		エクスボート(O)	
> 🗁 ra_crg		Export FSP Project	
i configuratio		Export FSP User Pack	
R7FA2L1AB2		プロジェクトのビルド(B)	
RA2L1_DEBU		プロジェクトをクリーンにする	
> ⑦ Developer A	s	更新(F)	F5
BARLI ELOAT	_	プロジェクトを閉じる(S)	
TALLI FSP CAN		無関係なプロジェクトを閉一つ	パティ・ガイアロガを閉く
TRA2L1_FSP_SPI		202 6 6 1	
RA2L1_FSP_SW_		ビルト・ダーケット	×
RA2L1_FULL_CC		インテックス	>
RA2L1_FULL_CC		ビルド構成	>
RA2L1_LCD	0	実行(R)	>
TRAZET_EED	*	デバッグ(D)	>
		ローカル履歴から復元(Y)	
🔲 วือパティー 🛚 💦		MISRA-C	>
RA2L1 DEBUG TE	1	C/C++ Project Settings	Ctrl+Alt+P
		Save build settings report	
リソース プロバティ		Change Device	
● 「「「「」 「「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」	*	C/C++ コード解析を実行	
	~	Team	,
DT		Compare With	
最終		Sustam Evoloror	
派生		Command Desemb	
(E#		Velidete	
		validate	
		構成	>
		Source	>
		プロパティ(R)	Alt+Enter
	_		

プロジェクトを右クリック、プロパティ。







C/C++ビルド – 設定 – ツール設定タブ – Optimization Level「Optimization more(-O2)」

|--|

この部分を「None(-O0)」に変えると、最適化は OFF となりますので、プログラムのデバッグがやり易くなります。

※最適化 OFF でデバッグ完了時は問題のなかったプログラムが、最適化を ON すると期待通りの動作をしなくなる ケースもありますので、最終的に使用するオプションに合わせての動作確認は必要です。





6.5.1. メモリ内容の確認

ルネサス CS+の環境に慣れている方ですと、「表示 – メモリ」でメモリ内容を確認する事があると思います。

(RA2L1_DEBUG)	_TEST] FSP Configuration 🕼 hal_entry.c 🔀 🕼 startup.c 🕼 main.c 🖓		(x)= 変数 🕴 🗣 ブレークポイン	ト 🎬 逆アセンブル 🛛	Ъ プロジェクト・エクスプローラー 🛭 🏘
1	<pre>#include "hal_data.h"</pre>	^ 🗆			
2				Z 1	(+
3	FSP_CPP_HEADER		有削	- <u>+</u>	112
4	<pre>void R_BSP_WarmStart(bsp_warm_start_event_t event);</pre>		> 🏓 a	unsigned long [0x20000450(16 進)
5	FSP_CPP_FOOTER				
6					
8	⊕ * main() is generated by the RA Configuration editor and is used				
11	<pre> ovoid hal_entry(void) </pre>				
12 00000550	{				
13	/* TODO: add your own code here */				
	🙆 unused variable 'a' [-Wunused-variable]				
14 00000556	unsigned long a[10]={1,2,3,4,5,6,7,8,9,10};				
15					
16	⊖ while(1)				
17	{				
♦ 18 00000570	NOP();				
19	}				
20					

実行(R) ウィンドウ(W) ヘルプ(H)				
★ □ 新規ウィンドウ(N)		0 🔳 💦 🥆 🖓 📕 i=	s 🕫 😒 🗸	ا 😫
エディター >			0 :	cic.
外観 >			× : EB 40	
figuration ビューの表示(V) >		Debugger Console	»	
パースペクティブ(R) >	ø	Debug Sources		
IEADER P Warms ナビゲーション(G) >	묾	Peripherals		90
		アウトライン	Alt+シフト+Q,O	Ш
る」 テハック・ビューを更新 ・ この = 10 年(の)	0	エラー・ログ	Alt+シフト+Q,L	90
entry(voice)	₽	コンソール	Alt+シフト+Q,C	96
		シグナル		20
µsed variable 'a' [-Wunused-variab	۰	スマート・ブラウザー		90
ned short a[10]={1,2,3,4,5,6,7,8,	ц.	スマート・マニュアル		100
(1)	\$	ターミナル		96
	栨	デバッグ		90 90
_NOP();	IJ	デバッグ・シェル		
	Ы	テンプレート		90 20
[Z_SECURE_BUILD iter non-secure code */	•	ブレークポイント	Alt+シフト+Q,B	90
_NonSecureEnter();	6	プロジェクト・エクスプローラー		96 - 26
	0	メモリー		
	0	メモリーブラウザー		90
unction is called at various poin	1	モジュール		20
	1010 0101	レジスター		96
SP_WARM_START_RESET == event)	=	逆アセンブル		20
EATURE_FLASH_LP_VERSION != 0	6 €	式		90
* Enable monding from data flack		実行可能ファイル		20
chapte reduting from data flash.	=6	進行状況		
, 🔍 問題 🚇 スマート・ブラウザー 🔲 メエリー	(x)=	変数	Alt+シフト+Q,V	
enesas GDB Hardware Debugging	2	問題	Alt+シフト+Q,X	H
5.01445 V		その他(O)	Alt+シフト+Q,Q	

ウィンドウ – ビューの表示 - メモリー





デフォルトでは、コンソール(コンパイル時のメッセージが出る場所)と同じところにタブでメモリーが追加されます。 +のアイコンが表示領域の追加なので、

モニター・メモリー	X
モニターするアドレスと式を入力してください: 0x20000000 ~	
 OK キャンセル 	

📃 コンソール 🕕 デバッグ・シェル 🔝 問題 🁒 スマート・ブラウザー 🚺	X モリ− 🖾				
E=9- 🕂 🐇 🗞	0x20000000 : 0x200000	000 <16 進数の	整数> 🛛 📭	🏰 新規レンダリン	ッグ]
♦ 0x20000000	アドレス	0 - 3	4 - 7	8 - B	C - F
	000000020000440	00000FE8	00020000	20000450	00000000
	000000020000450	00000001	0000002	0000003	00000004
	000000020000460	00000005	0000006	0000007	8000000
	000000020000470	0000009	A0000000	20000480	0000000
	000000020000480	20000488	000005C5	20000490	00000CED
	000000020000490	00000000	0000000	0000000	0000000
	0000000200004A0	00000000	0000000	0000000	0000000
	0000000200004B0	00000000	0000000	0000000	0000000
	0000000200004C0	00000000	0000000	0000000	0000000
	0000000200004D0	0000000	00000000	00000000	0000000

0x2000 0450~ に、0x1~0xA(1~10)の値が格納されている事が判ります。





6.6. 自動変数のサイズ

long a[300];

上記は、4byte × 300 で、1200bytes の、メモリを消費します。FSP のデフォルトのスタックサイズは 0x400(1024bytes)なので、超えています。

対策としては単純で、

(1)static 変数として確保する

static long a[300];

(2)スタックサイズを増やす

c hal_entry.c	.c startup.c	.c main.c	戀 [RA2L1	_DEBUG_TEST] FSP Configuration 🔀					
Board Supp	Board Support Package Configuration Generate Project Content								
					🔯 Restore Defaults				
Device Selection	on								
FSP version:	2.4.0		~	Board Details					
Board:	Custom User Boar	rd (Any Device) 🚿	1						
Device:	R7FA2L1AB2DFP								
RTOS:	No RTOS		\sim						
Summary BSP C	Clocks Pins Interr	upts Event Links	Stacks Co	mponents					

FSP の BSP の設定を開き、





Custom	User Board (Any Device)			
Settings	プロパティ > OFS0 register settings	值		
	OFS1 register settings MPU			
	> Power	Not Suggested		
	ID Code Mode	Unlocked (Ignore ID)		
	ID Code (32 Hex Characters) V RA Common	FFFFFFFFFFFFFFFFFFFFFFFF		
	Main stack size (bytes) Heap size (bytes)	0x800 0		
	MCU Vcc (mV) Parameter checking	3300 Disabled		
	Assert Failures	Return FSP_ERR_ASSERTION		
	Soft Reset	No Error Log Disabled		

Main stack size

をデフォルトの 0x400(1024)から増やしてください。なお、当該ボードで採用している RA2L1 マイコンは、メインメモリ 32kB です。

6.7. sprintf を使ったフォーマット

プログラムで、数値計算を行った際、sprintf でフォーマットを行いたいというケースがあるかと思います。

```
#include "hal_data.h"
#include <stdio.h>
void hal_entry(void)
{
   /* TODO: add your own code here */
   char buf[80];
   char buf2[80];
   sprintf(buf, "%d:%d", 10,20);
   sprintf(buf2, "%.2f:%f", 1.234f,2.3456f);
   while(1)
   {
       __NOP();
   }
#if BSP_TZ_SECURE_BUILD
   /* Enter non-secure code */
   R_BSP_NonSecureEnter();
#endif
}
```

buf は、"10:20¥0" buf2 は、"1.23:2.3456¥0"





となる事が期待値です。上記コードを実際に実行すると、

(x)= 変数 🙁 💁 ブレーク:	ポイント 🎫 逆アセンフ	ル 陷 プロジェクト・エクスプローラ	
名前	型	値	
🗸 🧀 buf	char [80]	0x200004a0(16 進)	
(×)= buf[0]	char	49 '1'	
(×)= buf[1]	char	48 '0'	
(×)= buf[2]	char	58 ':' 10:20¥	0
(×)= buf[3]	char	50 '2' ^{(¥0:終}	端文字)となっており、期待通り
(×)= buf[4]	char	48 '0'	
(×)= buf[5]	char	0 '¥0'	
(×)= buf[6]	char	0 '¥0'	
(×)= buf[7]	char	0 '¥0'	
(×)= buf[8]	char	-80 '-'	
(×)= buf[9]	char	4 '¥004'	
(×)= buf[10]	char	0 '¥0'	
(×)= buf[11]	char	32'' buf =	未初期化なので内容は不問
(×)= buf[12]	char	101 'e'	
(×)= buf[13]	char	7 '¥a'	
(×)= buf[14]	char	0 '¥0'	
(×)= buf[15]	char	0 '¥0'	
(×)= buf[16]	char	0 '¥0'	

(*)= 変数 🕴 🗣 ブレークポイン	ト 🏧 逆アセンブル 🛛	Ъ プロジェクト・エクスプローラー 💠
名前	型	值 ^
> 🥭 buf	char [80]	0x200004a0(16 進)
🗸 🥭 buf2	char [80]	0x20000450 <g_main_sta< th=""></g_main_sta<>
(×)= buf2[0]	char	58 ':' :¥0 になっている
(×)= buf2[1]	char	0 '¥0'
(×)= buf2[2]	char	0 '¥0'
(×)= buf2[3]	char	0 '¥0'
(×)= buf2[4]	char	-75 'オ'
(×)= buf2[5]	char	45'-' タハキ知期化初
(×)= buf2[6]	char	0 '¥0' 多万木初期化部
(x)= buf2[7]	char	0 '¥0'
(x)= buf2[8]	char	0 '¥0'
(×)= buf2[9]	char	0 '¥0'
(×)= buf2[10]	char	0 '¥0'

フォーマットとして、%dを指定した buf は期待通りの値が入っていますが、%fを指定した buf2 は、%f の部分が抜 け落ちている感じです。





プロジェクトの設定で、リンカ(GNU ARM Cross C Linker)の Use float with nano printf の設定を「チェック」してください。(デフォルトは未チェックです)

※デフォルトでは、printf ライブラリの nano 版(サイズの小さな版)が使われており、float の値を扱わない様になって いるためです

上記チェックした状態で、ビルド、実行を行うと、



エラールチンに飛んできてしまいました…。(プログラムの実行が sprintf の先へは進みません) Detault_Handrer は FSP のエラーハンドラーです。メモリ関係のエラーの時に飛んでくる事が多いです。

SmartRA 学習キット スタートアップマニュアル 株式会社 北手電子



.c hal_entry.c	c startup.c	.c main.c	戀 [RA2L1	I_DEBUG_TEST] FSP Configuration 🔀	
Board Support Package Configuration Generate Project Content					
					Restore Defaults
Device Selection	on				
FSP version:	2.4.0		~	Board Details	
Board:	Custom User Boar	d (Any Device)	~ 🔤		
Device:	R7FA2L1AB2DFP				
RTOS:	No RTOS		\sim		
Summary BSP C	Clocks Pins Intern	upts Event Links	Stacks Co	mponents	

FSP の BSP の設定を開き、

□ プロパティー ☆ 問題 Q スマート・ブラウザー					
Custom	User Board (Any Device)				
Settings プロパティ ID Code (32 Hex Characters)		値 FFFFFFFFFFFFFFFFFFFFFFFFFFFF			^
	Main stack size (bytes) Heap size (bytes)	0x400			
	MCU Vcc (mV) Parameter checking	3300 Disabled		-	
	Assert Failures Error Log	Return FSP_ERR_ASSERTION No Error Log			
	Soft Reset Main Oscillator Populated	Disabled Populated			
	PFS Protect C Runtime Initialization	Enabled Enabled			
	Main Oscillator Wait Time Main Oscillator Clock Source	32768 us Crystal or Resonator			
	Subclock Populated Subclock Drive (Drive capacitance availability varies	Populated Standard/Normal mode			~

Heap size を適当な値(例えば 0x200 等)を入力してください。(デフォルトは 0)





※FSPの設定を変えた場合は



のボタンを押さないと設定が反映されませんので注意ください

(x)= 変数 🕴 🎱 ブレークポイン	ト 🎫 逆アセンブル	Ъ プロジェク	ト・エクスプローラー 🙀 🤋
名前	型	値	
> 🥟 buf	char [80]	0x20000a10	0(16進)
🗸 🏓 buf2	char [80]	0x200009c	0 <g_main_stack+< td=""></g_main_stack+<>
(×)= buf2[0]	char	49 '1'	
(×)= buf2[1]	char	46 🖓	
(×)= buf2[2]	char	50 '2'	
(×)= buf2[3]	char	51 '3'	
(x)= buf2[4]	char	58 't'	
(×)= buf2[5]	char	50 '2'	
(×)= buf2[6]	char	46 '.'	
(×)= buf2[7]	char	51 '3'	
(×)= buf2[8]	char	52 '4'	1.23:2.345600¥
(×)= buf2[9]	char	53 '5'	<mark>になっている</mark>
(×)= buf2[10]	char	54 '6'	→期待通り
(×)= buf2[11]	char	48 '0'	
(x)= buf2[12]	char	48 '0'	
(x)= buf2[13]	char	0 '¥0'	
(x)= buf2[14]	char	0 '¥0'	
(×)= buf2[15]	char	0 '¥0'	多分未初期化部
(×)= buf2[16]	char	0 '¥0'	

-Heap size に関して-

size	size(10 進)	举動
0 ~ 0x10	0 ~ 16	DefaultHandrer トラップ(実行時エラー)
0x20 ~ 0xe0	32 ~ 224	実行はされるが結果(buf2 の中身)が正しくない
0xf0 ~	240 ~	正常に動作

sprintf で float の値をフォーマットするのに、どの程度のヒープサイズが必要か(標準ライブラリの仕様書があれば、 記載があるとは思うのですが)を実験してみた結果が上記です。ヒープサイズは、16byte 単位(0 の次は 0x10)での 設定となる様です。確保した容量が少ない時は、正常に動作しませんでした。上記は、1 回の sprintf 呼び出しで 2 個 の float 型の変数を処理した場合です。(もっと複雑なフォーマットを指定した場合、より多くのメモリが必要になる可能 性は考えられます。)ここでは、正常に動作した 0xf0 の倍程度の 0x200(512 バイト)としましたが、実行結果がおか しいと思われる場合は、ヒープサイズが足りないというケースもあるという事を認識して頂ければと思います。





-sprintf(標準ライブラリ関数)とfloat2str(自作関数)の比較-

(a)sprintf を使用

sprintf(buf2, "%.2f", 1.234f);

(b)float2str(自作関数)を使用 ※sci.c に含まれています

float2str(1.234f, 2, buf2);

(1)実行時間

上記の実行時間を見てみました。

	最適化デフォルト(-O2)	最適化 OFF(-O0)
(a)sprintf	238[us]	247[us]
(b)float2str	41.6[us]	57.6[us]

実行時間は、自作関数の方が有利という結果となりました。

※実は、RX231(SmartRX 学習キット)の時も同様の比較を行っているのですが、その際は

sprintf 4.4us

float2str 17.9us

と、自作関数の方が遅く、これだったら、わざわざ自作関数を作る理由もなく、標準ライブラリ関数を使う方が良いと思いました。RA では、sprintf が大幅に遅く、かなり改善の余地があるのではないかと思います。(sprintf は、ARM-gcc に含まれている部分であるかと思うのですが。(*1))

(*1)RX231 の sprintf は、ルネサスの CC-RX に含まれるものですので、マイコンで動かす事を前提にチューニングさ れているのかも知れません。それに対し、ARM-gcc の sprintf は元々は、x86 のコードから来ていて、マイコンで動か す事があまり意識されていないのかも知れません。あくまで憶測です。

(2) プログラムサイズ (トータルのサイズ)

	最適化デフォルト(-O2)	最適化 OFF(-O0)
(a)sprintf	0x4e94(20,116bytes)	0x534c(21,324bytes)
(b)float2str	0x55e4(21,988bytes)	0x5bf4(23,540bytes)
	内 float2str 0x1b0(432bytes)	内 float2str 0x2fc(764bytes)
差分	1,872bytes	2,216bytes

プログラムのサイズは、自作関数を使った方が増える形となりました。

※float2strを使った場合、浮動小数点の四則演算ルーチンが追加されている分、float2str 関数での増加分以上に、 サイズが増加しています。



SmartRA 学習キット スタートアップマニュアル

54



取扱説明書改定記録

バージョン	発行日	ページ	改定内容
REV.1.0.0.0	2021.6.17	_	初版発行
REV.1.1.0.0	2023.1.12	P19,20	FSP のバージョンアップに伴う操作を追記

お問合せ窓口

最新情報については弊社ホームページをご活用ください。 ご不明点は弊社サポート窓口までお問合せください。

_{株式会社} 北斗電子

〒060-0042 札幌市中央区大通西 16 丁目 3 番地 7 TEL 011-640-8800 FAX 011-640-8801 e-mail:support@hokutodenshi.co.jp (サポート用)、order@hokutodenshi.co.jp (ご注文用) URL:http://www.hokutodenshi.co.jp

商標等の表記について

- ・ 全ての商標及び登録商標はそれぞれの所有者に帰属します。
- ・ パーソナルコンピュータを PC と称します。



ルネサス エレクトロニクス RA マイコン搭載 HSB シリーズマイコンボード 評価キット

SmartRA 学習キット スタートアップマニュアル (始めにお読みください)

_{株式会社} 上手電子

©2021-2023 北斗電子 Printed in Japan 2023 年 1 月 12 日改訂 REV.1.1.0.0 (230112)