BCRX21A マイコン開発セット マニュアル

初版 2015.8.5

【 製品概要 】

本マニュアルはBCRX21A CPUボードのソフトウエア開発を行うために必要なソフトウエアイ ンストゥール手順、添付CDのサンプルプログラムの動作について解説されています。特に新しい統合開 発環境CS+における開発方法について多く記述してあります。

※本CPUボード開発にはルネサスエレクトロニクス社製E1が必要です。



1. 開発環境、事前準備

- 1-1. 開発環境
 - a:開発セット 同梱物
 - b: BCRX21A CPUボードの特徴
 - c: E1エミュレータ (デバッカ)
 - d:無償のCS+、RX用Cコンパイラのダウンロード
 - e:CDコピー、デバイスドライバのインストゥール
- 1-2 動作、デバック
 - a:CS+起動、コンパイル、書き込み、動作
 - b:新しいプログラムを作る CS+ 操作
 - b-1:#include iodefine.hを忘れると何も出来ない
 - b-2:デバッカの設定がデフォルトはエミュレータなので注意
 - b-3:E1から電源供給
 - b-4:WDT動作、リセット電圧選択はOFSO、1で設定する
 - c : その他
 - c-1:動作中に変数の変化を見るには?
 - c-2:サンプルを走らせるときにvect.hの重複するアドレスを削除
 - c-3:三角関数math.hはインクルードもCS+の設定も必要
 - c-4:割り込みが入っているか?周期は?簡単なチェック方法
 - c-5:既存のプログラムを雛形として新しいプログラムを作る
- 2. サンプルプログラム
 - 2-1. sample1 出力ポートのON, OFF
 - 2-2. sample2 SIO (USB) でパソコンとのやりとり

- 2-3. sample3 A/D変換をUSB出力
- 2-4. sample4 割り込み
- 2-5. sample5 PWM出力
- 2-6. sample6 三角、対数、平方根関数を使う
- 2-7. sample7 D/Aにsin、cos演算した正弦波を出力する ポート入力設定注意!

1-1. 開発環境

a:開発セット同梱物

BCRX21A CPUボード CD(サンプルプログラム、デバイスドライバ、ドキュメント) マニュアル(本誌) 電源ケーブル、USB(ミニ)ケーブル

※開発に必要なルネサスエレクトロニクス社製デバッカE1は同封されておりません。別途必要です。



b:BCRX21A CPUボードの特徴

●ルネサス独自のRX CPUコア、内部32ビットデータバス幅マイクロコンピュータ

3.3V 50MHz動作可能。5段パイプラインCISCハーバードアーキテクチャ

●メモリ容量 内蔵フラッシュROM 512Kバイト、内蔵RAM64Kバイト 内蔵データフラッシュ8Kバイト

●A/Dコンバータ :<mark>24ビット分解能</mark>×7 変換速度92µsec、12.2KHz(DSADC

LK=25MHz)、入力範囲±0.5904V(これ以上、以下の電圧を加えないで下さい)

●A/Dコンバータ:10ビット分解能×7 変換速度 2µsec(ADCLK=25MHz) 入力範囲0-3.3V

●D/Aコンバータ:10ビット分解能×2

●外部バス拡張機能:なし(外部にデータバス、アドレスバス等出力できません)

● I /Oポート:入出力66、入力1、

●タイマ:マルチファンクションタイマパルスユニット2(16ビットタイマ×6)、8ビットタイマ(8 ビット×2)×2、コンペアマッチタイマ(16ビットタイマ×2)×2、ウオッチドッグタイマ×1、 独立ウオッチドグタイマ×1、RTCc (リアルタイムクロック)

●シリアルコミュニュケーションインターフェイス×5ch、IICバス×2ch、IrDAバスインタ ーフェイス、シリアルペリフェラルインターフェイス

- ●オンチップデバッキングシステム:(JTAGおよびFINEインターフェイス)
- ●動作周囲温度:-40~+105℃

●USBポート: 1ch(SIO1使用、フォトカプラ絶縁) FTDI社 FT232RL使用

●EEPROM: 25LC256(32Kバイト) 電源OFFでもデータ保持。 ※オプション(実 装品はご相談下さい)

●電源 2.7V~3.6V 単一 30mA(50MHz動作 TYPE)

E1デバッカを使用して動作させる時E1から3.3Vの電源を供給できます。

デバック時など200mA以内の使用であれば他に用意する必要はありません。

- ●クリスタル:メイン 12.5MHz(×4逓倍で50MHz作成)実装済み。
- ●デバックコネクタ: E1用(FINEインターフェイス)デバックコネクタ実装済み。
- ●ΔΣA/D用外部基準電源:1.200V(8ppm/°C)実装済み。
- ●基板サイズ 69×57×13(H)mm
- ●基板仕上げ 金メッキ RoHS指令準拠 基板、部品、半田付け全ての工程でRoHS指令準拠仕様。

基板大きさ(部品面)



c: E1エミュレータ



概要

E1エミュレータは、ルネサス主要マイコンに対応したオンチップデバッギングエミュレータです。基本 的なデバッグ機能を有した低価格の購入しやすい開発ツールで、フラッシュプログラマとしても使用可能 です。

C言語ソースデバックが可能で、1 行実行、ブレークポイント設定、変数、レジスタ、メモリ参照等々、 従来であれば高価なICEしか出来なかった機能が、安価に実現されています。また、使い方もHEW(統 合開発環境)のE8aと同じで、経験があれば半日で、無くても1日で必要な操作を会得することが出来 ると思います。

マイコンとの通信として、シリアル接続方式とJTAG接続方式の2種類に対応しています。使用可能な デバッグインタフェースは、ご使用になるマイコンにより異なります。

また、基本デバッグ機能に加え、ホットプラグイン機能(動作中のユーザシステムに後からE1エミュレ ータを接続して、プログラムの動作確認を行うことが可能)を搭載しているため、プログラムのデバッグ・ 性能評価に大きく貢献できます。

対応MPU

- V850 ファミリ
- RX ファミリ
- RL78 ファミリ
- R8C ファミリ
- 78K ファミリ



E1を購入するとCDが添付されていて、ドライバーのインストールとセルフチェックを行った後に、ネットから開発環境CS+とCコンパイラのダウンロードを行います。

d:無償版RX用Cコンパイラのダウンロード

プログラムの開発はルネサスエレクトロニクス社の統合開発環境CS+でC言語を用い動作させること ができます。CD添付のサンプルプログラムはこの環境下で作成されています。無償版をダウンロードし て使用します。

ネット検索で→「RX コンパイラ」の検索で表示されます。 これを使用します。 RXファミリ用C/C++コンパイラパッケージ 更新通知登録:🕹 Share: 👖 💟 😻 🛨 🖁 祝芸 ドキュメント アプリケーションノート/サンブルコード ダウンロード 設計性報/サポート
 トラ
 特長
 優化に最適化
 既存製品がらの移行支援
 製品・ケケージ内容
 CS+環境用製品情報

 e² studio 環境用製品情報
 High-performance Embedded Workshop 環境用製品情報

 RX7元リ用c/c++ヨンパイ ラパッケージ RXファミリ(32ピット) SuperHファミリ用C/C++コ ンパイラパッケージ 概要 RXファミリ用のC/C++コンバイラは、組み込み用途におけるROM化システムの開発を前提とし、コ 効率やブログラム実 行速度を向上させる強力な最適化機能をはいめ、豊富な根器組み込み向け拡張機能を提供します。統合開発環境ごとご 供方式が異なります。統合開発環境ごとの製品情報は下表をご覧ください。 850用ソフィワーク [SP850] 132Rファミリ用C/C++コ - <u>イラパッケージ</u>[M3T-製品名 統合開発環境 説明 発注型名 統合開発環境、シミュレータ 等を含むエンパイラバッケー 媒体あり^{*1} 14201111 20032R] RL78ファミリ用Cコンパイラ/ RXファミリ用C/C++コンパ イラバッケージ(統合開発環 増つき) RTCRX0000CL02WDF -> 詳細は ◎ こちらをご覧くだ 媒体なし、ライセンスのみ さい 煤(すよし、フイセノノルルか RTCRX0000CL02WNR 使用することもできます コンパイラ、アセンブラ、リン 力を含むコンパイラパッケー 次(統合開発環境、シミュレ ータ) むらッケージに含まれ 統合開発環境はパッケージ RXファジリ用C/C++コンバ に含まれていません。 イラバッケージ(統合開発環 境な」) 使 e² studio(別途インス トール必要です)と組み合わ せて使用できます。 媒体なし、ライセンスのみ^{**} RTCRX0000CC02WNR ません) 詳細は 🛛 こちらをご覧くだ 統合開発環境、シミュレータ 等を含むコンパイラパッケー RXファジリ用C/C++コンバ イラバッケージ(High-performance Embedded Workshopプき) [注1] R0C5RX00XSW01R ン 詳細は 🕲 こちらをご覧くだ

現在(2015.8.7) RX用の開発環境は3種類あります。本開発セットで使用するのは1番上のC S+環境です。

「ダウンロード」をクリックします。CS+用をダウンロードします。

CS+ (I⊟CubeSuite+)	【無償評価版】統合開発 環境 CS+ for CC V3.01.00 (一括ダウン ロード版)	Apr.20.15	CS+ パッケージに含ま れるサブパッケージで す。 デバッガおよび無償評価 版コンパイラを含みま す。 CubeSuite+からのアッ ブデートにも使用できま す。 対応マイコン: RH850フ ァミリ、RXファミリ、RL78 ファミリ
CS+ (BCubeSuite+)	【無償評価版】統合開発 環境 CS+ for CC V3.01.00 (分割ダウン ロード版)	Apr.20.15	CS+ パッケージに含ま れるサブバッケージで す。 デバッガおよび無償評価 版コンパイラを含みま す。 CubeSuite+からのアッ ブデートにも使用できま す。 対応マイコン: RH850フ ァミリ、RXファミリ、RL78 ファミリ

いずれかのCS+ for CCをダウンロードし、指示に従い展開して下さい。統合開発環境とCコン パイラが同時にダウンロードされます。なお、CS+は以前、CubeSuite+という名称でしたが、 2014年にCS+となりました。大きな変更点は

 CS+ for CA, CX、RX、CS+ for CC とCPU別に2つに分割されました。
 設定等も変更されています。但し、上位互換性はあり、CubeSuite+で作成されたソフトは CS+ for CCでコンパイル、実行可能です。 e:開発セット添付CDコピー、デバイスドライバのインストゥール

省略

1-2 動作、デバック

a:CS+ for CC 起動、コンパイル、書き込み、動作



CDに添付しているサンプルプログラムを使って、コンパイル、書き込み、動作の方法を示します。

CS+ for CCを起動します。ここでは例としてRX21A_sample¥sample1を動 作させます。基板上のLED D1が点滅するプログラムです。

ファイル → ファイルを開く → sample1.mtpjをダブルクリックします。



プロジェクトツリーが表示されます。sample1. cをダブルクリック。 E1は設定済みです。



sample1. cが中央に表示されます。とりあえず、実行してみます。E1のケーブルを基板のCN 1に挿入します。電源はE1から供給しますので、不要です。(写真ご参考)



(E1に貼ってある文字「外部クロック時にE1の立ち上げで外部

クロック入力、周波数を指定しないとフラッシュROMイレーズエラーが出る」場合があります)

「ビルド後、デバック・ツールヘプログラムを転送」をクリック。

インドウ(W) ヘルプ(H)
- Gr Gr 🔨 📩 🐂 🔘 💿 🗠 🖙 🖓 🖕 🖓
ビルド後デバッグ・ツールヘプログラムをダウンロードします。 (F6) 生成/副 端子配置表し

上手く転送できると、今まで表示されていなかったプログラムの絶対番地が表示されます。E1から電源がCPU基板に供給されます。

38 39 40 41 42 43 44 fff805e8	•	<pre>/************************************</pre>
45 46 47 fff805ea	1	∃{ /* System initialization */ system_init();
49 50 51 52 53 54 55 56 57 58 57 58 59 59 50 50 57 58 59 50 51 52 57 58 59 51 50 51 52 53 54 57 58 57 58 57 58 57 58 57 58 59 50 51 52 53 54 55 56 57 58 59		<pre>//port test //P3.5(は入力専用なので波形が出ません。 //PORTJ.PDR.BIT.B1 = 1; PORT0.PDR.BYTE = 0xff; PORT1.PDR.BYTE = 0xff; PORT2.PDR.BYTE = 0xff; PORT3.PDR.BYTE = 0xff; PORT4.PDR.BYTE = 0xff; PORT5.PDR.BYTE = 0xff; PORT5.PDR.BYTE = 0xff; PORT6.PDR.BYTE = 0xff; PORTC.PDR.BYTE = 0xff; PORTE.PDR.BYTE = 0xff; PORTE.PDR.BYTE = 0xff; PORT.PDR.BYTE = 0xff; PORT.PDR.P</pre>

ここまでいかなかった場合、E1のインストゥールをご検証願います。

次に、プログラムを動作させます。「CPUリセット後、プログラムを実行」をクリック。

·ドウ(W) ヘルプ(H)	
Gr Gr 🔨 🧄 🖧 🐂 💷 🗩 🗩 🕨	ريا تي ت≣ (¶ التي تي ت
	CPUリセット後、プログラムを実行します。

E1のRUN(緑LED)が点灯し、基板のD1が点滅したら動作しています。CS+の右下部にも表示 されます。

▶RUN ⇔実行中 ■RX E1(Serial) (3計測中)

ここまで確認できましたら、一度止めます。



main関数のIwaitの数値2箇所を1桁0を増やしてみます。

	N	
while()	/
{		
	PORTO POOR BYTE = 0x55	
	PORT1 PODR BYTE = 0.55	
	POPT2 DOPD DVTE = 0.55	
	PORTZ.FOUR.DITE - 0X00,	
	PURI3.PUUR.BYTE = UX00;	/ /
	PURI4.PUDR.BYIE = 0x66;	
	PORT5.PODR.BYTE = 0x55;	
	PORTA.PODR.BYTE = 0x55;	
	$PORTB.PODR.BYTE = 0 \times 55$:	
	PORTC PODR BYTE = 0×55	
	PORTE PODR BVTE = 0.55	
	DODT I DODD DVTE - 0.65	
	DODTU DODD DVTE - 0.55.	
	PURIH.PUUR.BTTE = 0x00;	▶ /
	Twait(5000000);	/
	PORIO.PODR.BYIE = Oxaa;	/
	PORT1.PODR.BYTE = 0×aa;	/
	PORT2.PODR.BYTE = 0×aa;	/
	PORT3.PODR.BYTE = 0xaa:	/
	PORT4.PODR.BYTE = Oxaa:	/
	PORTS PODR BYTE = Oxaa:	
	PORTA PODR BYTE = 0vaa:	/
	DODTE DODE DVTE - Over	/
	PORTE PODE DVTE - Over	/
	PURIC.PUDR.BYTE - UXaa;	
	PURIE.PUUK.BYIE = Uxaa;	/
	PURIJ.PUUK.BYIE = Uxaa;	/
	PORTH.PODR.BYTE = 0×aa;	*
	lwait(5000000);	

セーブして

E) 表示(V) プロ	コジェクト	・(P) ビ	ルド(B)	デバッグ(D)) ツー
] 🗋 🖉	8	a 🛍 🔊	で 出	å. Å.		↓ 10
7	アイルは	「指定され	た項目	を保存し	,ます。 (Ctrl	+S)

さきほどの、

「ビルド後、デバック・ツールヘプログラムを転送」をクリック。 「CPUリセット後、プログラムを実行」をクリック。 LEDの点滅が先ほどより、遅くなったのが目視できましたでしょうか?

次に、ブレークポイントの設定を行ってみます。一度、プログラムを停止させます。 ブレークポイントを2点設定しました。手のマークの部分をマウスでダブルクリック。



0x55;命令によりPJ1=0となるので、消灯します。

更に「プログラムを現在の位置から実行」をクリックすると、もう一つのブレークポイントで停止し、P ORTJ. PODR. BYTE=0xaa;命令実行により、LEDは点灯します。



以上が、プログラムのコンパイル、E1へのダウンロード、実行、修正、ブレークポイント設定、動作の 概要です。

b:新しいプログラムを作る

CS+での新規プログラム開発方法と注意点を書きます。

🚺 sample1 - RX E1(Serial) - CS+ fo	for CC - [sample1.c]
ファイル(F) 編集(E) 表示(V) プロミ	ジェクト(P) ビルド(B) デバッグ(D) ツール(T) ウイ
新規作成(N)	▶ 🔀 新しいプロジェクトを作成(N)
J ファイルを開く(0)	Ctrl+O 前しいファイルを作成(F) Ctrl+N

ファイル→新しいプロジェクトを作成。CPUをR5F521A8BxFPを選択。

品(マイクロコントローラを検索	- できます) アッ プデート(U)		
FSF521 A6BxL_(10 FSF521 A7BxFN(64 FSF521 A7BxFN(64 FSF521 A7BxFN(64 FSF521 A7BxFN(64 FSF521 A7BxL_(10 FSF521 A8BxFN(64 FSF521 ABxFN(64 FSF521 ABxFN(64 FSF521 ABxFN(64 FSF521 ABxFN(64 FSF521 ABxFN(64	Opin) 品種名:R5F521 A3BxFP tpin) 内蔵ROMサイズ[K/代十]512 opin) 回 Opin) 道加情報:Package=PLQP0100KB-A Opin) 回 tpin) Opin Opin) 回 Opin) 回	*	
プロジェクト の 種类((K):	アプリケーション(CC-RX)	•	
プロジェクト 名(N):	test_sample		
作成場所(L):	C:¥WorkSpace¥RX21A_sample ▼ 参照(R)		
	☑ プロジェクト 名のフォルダを作成する(A)		
C:¥WorkSpace¥RX21 A_samp	ole¥test_sample¥test_sample.mtpj		
📃 既存のプロジェクトのファイル	構成を流用する(S)		
流用元のプロジェクト(P):	(流用元のプロジェクト・ファイルを入力してください) 💽 参	·昭(W)	

プロジェクト名をtest_sampleとしました。「作成」をクリック。

Image: Section 1 and Secti

ビルド後デバッグ・ツールヘプログラムをダウンロードします。(F6)

ビルド後デバック・ツールへプログラムをダウンロードクリック→エラーなしでダウンロード出来ますが、 当然、何も実行されません。

```
sample1. cを参考にもっとも簡単なプログラムを記入してみます。基板上のLEDを点滅させる
プログラムです。
void lwait(long wdata)
Ð{
        while(wdata != 0)
        {
               wdata--;
        }
 }
 void main(void)
₽{
    /* System initialization */
    system_init();
        PORTJ.PDR.BIT.B1 = 1;
        while(1)
        {
               PORTJ.PODR.BIT.B1 = 1;
               Iwait(500000);
PORTJ.PODR.BIT.B1 = 0;
                lwait(500000);
        }
```

ビルドすると PORTJが見つからないエラーが出ます。

番号 メッセージ 図 E052002(test.sample.c(41):E0520020:Identifier "PORTJ" is undefined

b-1:#include iodefine. hを忘れると何も出来ない

頭のほうに#include "iodefine.h" の1行を追記して下さい。またその下にクロック設定の sys.c をイ ンクルード、sys.cファイルをsample1等からこのtest_sampleにコピーして下さい。

#include ~iodefine.h"
#include ~sys.c"

これでエラーも出ず、ダウンロードできるはずです。ところが、「リセット後、実行」させてもLEDは光りません。

b-2:デバッカの設定がデフォルトはエミュレータなので注意

省略

ļ

b-3:E1から電源供給

省略

b-4:WDT動作、リセット電圧選択はOFS0、1で設定する

省略

c:その他

c-1:動作中に変数の変化を見るには?

例として s a m p l e 3 で使用している a d _ d a t a をウオッチ 1 に登録します。左クリックを押しな がらマウスで文字を囲み、右クリック ウオッチ 1 に登録を選択。

	ウォッチ1 に登録(R)	•	
4	アクション・イベント	の登録(A)	
. %	切り取り(T)	Ctrl+X	• **
•	⊐ピ–(C)	Ctrl+C	
œ	貼り付け(P)	Ctrl+V	vect
88	検索(F)	Ctrl+F	
- 🔁	移動(G)	Ctrl+G	-
3	ジャンプ先の位置へ進	む(U)	
5	ジャンプ前の位置へ戻	る(B)	
3	ここまで実行(H)		
•	PCをここに設定(P)		
~	関数ヘジャンプ(J)	F12	
ſ	タグ・ジャンプ(T)	Shift+F12	
1	逆アセンブルヘジャン	[,] プ(D)	
	ブックマーク(B)	,	-
	高度な設定(V)	,	
ad		ata[U];	
	A		

左クリックを押しながら、マウスで選択。右のウオッチ式に変数名が表示されるとOKです。

ウォッチ式 ⊞ ♥ g_dsad_data ⊮ ♥ ad_buff ♥ ad_data

なお、ad_buffのように頭に+が付く変数は配列で、プラスをクリックすることにより個々の値も 見ることが出来ます。

配列全体の値

オッチ1	
🖲 🏶 🐉 🌯 🗙 表記(N)+ 🔫	
ウォッチ式	値 型情報(バイト数)
🗄 单 g_dsad_data	- long [7](28)
∃ 😜 ad_buff	"ch6 = 🔭-8310" unsigned char…
♥ [0]	'c' (Ox63) unsigned char(1)
🛯 [1]	'h' (Ox68) unsigned char(1)
😜 [2]	'6' (0x36) unsigned char(1)
😜 [3]	''(Ox2O) unsigned char(1)
😜 [4]	'=' (0x3d) unsigned char(1)
😜 [5]	''(Ox2O) unsigned char(1)
😔 [6]	''(Ox2O) unsigned char(1)
☺ [7]	''(Ox2O) unsigned char(1)
😜 [8]	''(Ox2O) unsigned char(1)
😜 [9]	'-' (Ox2d) unsigned char(1)
😜 [10]	'8' (Ox38) unsigned char(1)
😜 [11]	'3' (Ox33) unsigned char(1)
😜 [12]	'1' (Ox31) unsigned char(1)
😔 [13] 🔺	'O' (Ox30) unsigned char(1)
😜 [14]	'' (OxOa) unsigned char(1)
😜 [15]	'' (OxOd) unsigned char(1)
😜 [16] 💊	'' (OxOO) unsigned char(1)
	'' (OxOO) unsigned char(1)
😜 [18]	'' (OxOO) unsigned char(1)
😜 [19]	'' (OxOO) unsigned char(1)
🐳 ad_data 💧	<mark>-8310 (Oxffffdf8a)</mark> long(4)
\mathbf{h}	

更に、デバック・ツール設定→実行中のメモリアクセス→実行を一瞬停止してアクセスする→はいにします。

ראישטר אישטר איש	🚰 プロパティ 🌇 逆アヤンブルイ 📑 test sample.c	
2 🐼 🙎 🔳		
^I test_sample (プロジェクト)* ■ RSF521A8BxFP (マイクロコントローラ) へ cC-RX (ビルド・ツール) ■ RX E1(Serial) (デパッグ・ツール) = ファイル	 ■ K & El(Serial) の JD/374 メモリ メモリ・マッピング メモリ書き込み時にパリファイを行う 単実行中のメモリ・アクセス 実行を一瞬停止してアクセスする 実行中に表示更新を行う 表示更新聞隔[me] 	[23] はい はい はい
 ● 弾 ビルド・ツール生成ファイル - ⑥ dbsct.c - ⑥ intprg.c - ⑨ sbrk.c - ⑨ test_sample.c - ⑨ vecttbl.c - ⑨ iodefine.h - ⑨ sbrk.h - ⑨ stacksct.h 	 ないないでの時間にある レジスタ 実行中にPC表示を行う ブレーク 優先的に使用するブレークポイントの種類 システム 内蔵プログラムをのめを書き換えるプログラムをデバッグする 内蔵データフラッシュを書き換えるプログラムをデバッグする 実行開始前に指定ルーチンを実行する ブレーク後に指定ルーチンを実行する トレース トレース トレース・ボータ種別 	いいえ ハードウ: いいえ いいえ いいえ いいえ いいえ トレース・ トレース・ 分岐
- tipedefine.h	タイム・スタンプ出力 実行を一瞬停止してアクセスする 実行中にはアクセスできないメモリ領域に対して、実行中に一瞬停.	いいえ 止してアクセスする 役定 / フック処3

これにより、動作中に変数が変化する様子を見ることが出来ます。

c-2:サンプルを走らせるときにvect. hの重複するアドレスを削除

省略

c-3:三角関数math.hはインクルードもCS+の設定も必要

sample6は三角、対数、平方根関数を使用しますが、ソースファイルにインクルードを記入するだけでなく、

#include <math.h> //sqr 等演算を行うのに必要 math.h 有効と共にこの表記も必要。

9ロジェクト・ツリー ギ ×	I resetprg.c 習 ブロパティ I sample6.c	
2 0 2 2	▲ CC-RX のプロパティ	
∃ III sample6 (プロジェクト)	4 E-K	
■ R5F521A8BxFP (マイクロコン	標準ライブラリの使用・構築方法	標準ライブラリ・ファイル作成(オプション変更時)
1 CC BX (FILE . W-U)	▲ 標準ライブラリ	
	ライブラリ構成	C(C89)(-lang=c)
	構築対象のライブラリ	カスタム(-head= <suboption>)</suboption>
白 🍱 ファイル	ランタイム・ライブラリを有効にする	(dl)(-head=runtime)
- dbsct.c	ctype.h(C89/C99)を有効にする	いいえ
	math.h(C89/C99)を有効にする	はい(-head=math)
intprg.c	mathf.h(C89/C99)を有効にする	UUX
resetprg.c	stdarg.h(C89/C99)を有効にする	0.012
🚽 sbrk.c	stdie.h(C89/C99)を有効にする	(JU)(-head=stdio)
	stdlib.h(C89/C99)を有効にする	(JU)-head=stdlib)
- vection.c	string.h(UB9/U99)在有XJIL 9 句	(at Ut-head=string)
🔚 iodefine.h	IDSLEUT+1/2 有S/IL 9 合	(i)())() (#()()
📲 sbrk.h	new(EUTT)/2 有X/IL 9 句 annulu (EOLU)を支付(本Z	(alu)=nead=new)
- Stackset h	cumplex(EC++)を有効にする	(1)15
	4 オゴジェカト	0.01%
typedefine.n	キカフォルダ	%BuildMadeName%
🔄 vect.h	出力ファイル名	%ProjectName% lib
sample6.c	機能縮小販入出力開数を生成する	()()Ž
	プロガラ人領域のセクション名	P
	定教領域のセクション名	C
	初期化データ領域のセクション名	D
	未初期化データ領域のセクション名	В
	リテラル 領域のセクション 名	L
	switch 文分岐テーブル領域のセクション 名	W
	初期値なし変数をアライメント4のセクションに配置する	いいえ
	初期値あり変数をアライメント4のセクションに配置する	いいえ
	const修飾変物をマライイントルのわたってに配置する	(31)7
	標準 ライブラリの使用・構築方法 標準ライブラリの使用・構築方法を選択します。	
	↓ 共通オプション / コンパイル・オプション / アセンブル・オプション / リンク・オプション /・	へキサ出力オブション / ライブラリ・ジェネレート・オプション /

CC-RX (ビルド・ツール) → ライブラリ・ジェネレート・オプション → math.hを有効にす る→はい としてください。

c-4:割り込みが入っているか?周期は?簡単なチェック方法

省略

c-5:既存のプログラムを雛形として新しいプログラムを作る

省略

2. サンプルプログラム

```
sample1 ポートのON/OFF
```

【 概要 】

ポートのON/OFFを繰り返します。基板上のLD1が点滅します。

【 プログラム 】

```
①void lwait(long wdata)
ł
      while(wdata != 0)
      {
            wdata--;
      }
}
* Function Name: main
* Description : The main loop
* Arguments
         : none
* Return Value : none
**********
②void main(void)
{
  /* System initialization */
3
                                            //外部12.5MH z 内部50MH z 動作
    system_init();
設定
```

//port test

//P3.5は入力専用なので波形が出ません。

//PORTJ.PDR.BIT.B1 = 1;

			4.5
337	hıl	01	1 \
w			
			-,

{

5

5	PORT0.PODR.BYTE = $0x55$;
	PORT1.PODR.BYTE = $0x55$;
	PORT2.PODR.BYTE = $0x55$;
	PORT3.PODR.BYTE = $0x55$;
	PORT4.PODR.BYTE = $0x55$;
	PORT5.PODR.BYTE = $0x55$;
	PORTA.PODR.BYTE = 0x55;
	PORTB.PODR.BYTE = 0x55;
	PORTC.PODR.BYTE = 0x55;
	PORTE.PODR.BYTE = $0x55$;
	PORTJ.PODR.BYTE = $0x55$;
	PORTH.PODR.BYTE = 0x55;
6	lwait(500000);
\bigcirc	PORT0.PODR.BYTE = 0xaa;
	PORT1.PODR.BYTE = 0xaa;
	PORT2.PODR.BYTE = 0xaa;
	PORT3.PODR.BYTE = 0xaa;
	PORT4.PODR.BYTE = 0xaa;
	PORT5.PODR.BYTE = 0xaa;
	PORTA.PODR.BYTE = 0xaa;
	PORTB.PODR.BYTE = 0xaa;
	PORTC.PODR.BYTE = 0xaa;
	PORTE.PODR.BYTE = 0xaa;
	PORTJ.PODR.BYTE = 0xaa;
	PORTH.PODR.BYTE = 0xaa;
	lwait(500000);
}	

}

【 解説 】

```
①void lwait(long wdata)
{
          while(wdata != 0)
          {
                     wdata--;
          }
```

}

時間を作り出している関数です。wdataがOになるまで関数から戻りません。数値が大きいほど時間 も長くなります。

* Function Name: main

* Description : The main loop

* Arguments : none

* Return Value : none

②void main(void)

{ ここからメインです。

/* System initialization */

③ system_init(); //外部12.5MH z 内部50MH z 動作設定 CPUを50MH z で動作させています。sys. cの中に関数はあります。

④ PORT0.PDR.BYTE = 0xff; PORT1.PDR.BYTE = 0xff;

ポートを出力ポートに設定しています。

⑤ PORT0.PODR.BYTE = 0x55; ポートに0 x 5 5 = 0 1 0 1 0 1 0 1 Bを出力しています。

6 lwait(500000);

LEDの点滅を人の目で確認出来る速度にするための待ち時間です。

⑦ PORT0.PODR.BYTE = 0xaa;

ポートに0×aa=10101010Bを出力しています。0×55に比べすべてのビットが反転し、L EDが点滅して見えます。仮に隣のポートと接触しているとレベルの変化がありませんので、例えばLE Dが点滅しません。それによりハードウエアの異常が検出できます。(隣は必ず異なる論理なので0,1で も1,0でも0になります)

2-2 sample2 SIO (USB) でパソコンとのやりとり

【 概要 】

USB出力をパソコンと接続し、データのやり取りを行います。お手数ですが、テラタームやハイパータ ーミナルなどのターミナルプログラムを使用しますので、無い方は、ネットで検索し、インストゥール願 います。例ではテラタームで行います。ボーレートは38400bpsに設定して下さい。

USBミニケーブルでパソコンとつなげると、USB側に電源が入ります。コントロールパネル→全ての コントロールパネル項目→デバイスマネージャー → ポート(COMとLPT)でUSB Seria I Port(COMxx)があることを確認して下さい。例ではCOM80となっています。



Tera Tremをシリアルポート COM80 →OKとします。



設定→シリアルポート→ボーレート38400として下さい。



CS+でsample2を開き、デバック・ツールヘプログラムダウンロード→CPUリセット後、プロ

グラム実行。



sample2 2015.07 と表示され、PCのキーボードを何か押すたびに、押した文字が表示 されると動作としてはOKです。

パソコンのキーボードを押した文字がCPU基板に送信され、それを返信(エコーバック)し、表示され るようになっています。

【 プログラム 】

void main(void)

{

//System initialization

system_init(); 50MH z 動作設定 //外部12.5MH z 内部

//SIO initial

① init_sio();

(2) char_out1('s'); char_out1('a'); char_out1('m'); char_out1('p'); char_out1('p'); char_out1('e'); char_out1('e'); char_out1('2'); char_out1('2'); char_out1('2'); char_out1('0'); char_out1('1'); char_out1('.'); char_out1('.');

省略		
[解説	3
}		
		}
4		<pre>while(1) { char_out1(char_in1());</pre>
3		<pre>char_out1('7'); char_out1(CR); char_out1(LF);</pre>

2-3 sample3 A/D変換をUSB出力

【 動作概要 】

ANDSON(CN7 23) ANDSOP(CN7 22)を入力とし、A/D変換した値をUSB からパソコンに送ります。±0.5904Vがフルスケールです。これ以上の電圧を入力しないで下さい。

🚇 COM80:38400baud - Tera Term VT	×
ファイル(F) 編集(E) 設定(S) コントロール(O) ウィンドウ(W) ヘルプ(H)	
sample3 2015.07	
ch0 = 0	
ch0 = 10487774	
ch0 = 10486933	
ch0 = 10489086	
ch0 = 10487527	
ch0 = 10487474	
ch0 = 10490753	
ch0 = 10487544	
ch0 = 10488253	
ch0 = 10488201	
chU = 1048/689	
chU = 10488153	
chu = 10488623	
chU = 10486988	
ICHU = 10488692	
IChU = 104688852	
chu = 10487883	
ICNU = 10491862	
chU = 10489770	
ICNU - 10487838	
ICNU = 10491800	
ICHU = 10486138	_
ICHU = 10487858	+

【 プログラム 】

void main(void)

{
unsigned char cf,loop;

//System initialization

① clrpsw_i();

//System initialization

	<pre>system_init();</pre>	//外部12.5MH z 内部50MH z 動作設定	
2	<pre>//DSAD initialization dsad_init();</pre>	//ΔΣ24bit A/Dイニシャライズ	
3	<pre>//MTU2a-channel0 (MTU0) initialization mtu0_init();</pre>	n //	
4	<pre>//ELC initialization elc_init();</pre>	//イベントリンクコントローラ ΔΣ24bitデータ変換-	\rightarrow

//SIO initial

init_sio();

```
char_out1('s');
char_out1('a');
char_out1('m');
char_out1('p');
char_out1('l');
char_out1('e');
char_out1('3');
char_out1(' ');
char_out1('2');
char_out1('0');
char_out1('1');
char_out1('5');
char_out1('.');
char_out1('0');
char_out1('7');
char_out1(CR);
char_out1(LF);
```

5

//

setpsw_i();

//Starts MTU0 and DSAD conversion MTU.TSTR.BIT.CST0 = 1;

while(1) {

```
6
         ad_data = g_dsad_data[0];
```

sprintf(ad_buff,"ch0 = %8d¥n¥r",ad_data);

```
cf = 1;loop = 0;
while(cf != 0)
{
           cf = ad_buff[loop];
           char_out1(cf);
           loop++;
}
```

lwait(100000);

```
}
```

{

}

⑦#pragma interrupt dsadi0_isr(vect = VECT(DSAD, DSADI0))
static void dsadi0_isr(void)

/* Read conversion data of DSAD channel0
DSADDR0 Delta-Sigma Data Register 0
b31-b0 Holding A/D results. Read only register. */
g_dsad_data[0] = (int32_t)DSAD.DSADDR0;

```
/* Clear Interrupt Request Register assigned DSADI0. */
IR(DSAD, DSADI0) = 0;
```

```
}
```

【 解説 】

省略

🗿 🧠 🛃 🏹 🗙 表記(N) - Heg	
ウォッチ式	値 型情報(バ	イト数) アドレ 🖌
🖃 👽 g_dsad_data	- long [7](28) 0x000004
😜 [O]	10467438 (0x009fb86e) long(4)	0×000004
€ [1]	3806 (0x00000ede) long(4)	0×000004
😜 [2]	-9953 (Oxffffd91f) long(4)	0×000004
😜 [3]	-21149 (Oxffffad63) long(4)	0×000004
€ [4]	15644636 (0x00eeb7dc) long(4)	0×000004
😜 [5]	15807746 (0x00f13502) long(4)	0×000004
😜 [6]	15565676 (0x00ed836c) long(4)	0x000004
🖃 👽 ad_buff	"ch0 = 10468080" unsigned	char 0x000000
● [0]	'c' (0x63) unsigned	char(1) 0x000000
😜 [1]	'h' (Ox68) unsigned	char(1) 0x000000
😜 [2]	'O' (Ox3O) unsigned	char(1) 0x000000
😜 [3]	''(Ox2O) unsigned	char(1) 0x000000
😻 [4]	'=' (0x3d) unsigned	char(1) 0x000000
😜 [5]	''(Ox2O) unsigned	char(1) 0x000000
😜 [6]	'1' (Ox31) unsigned	char(1) 0x000000
😜 [7]	'O' (Ox3O) unsigned	char(1) 0x000000
😻 [8]	'4' (0x34) unsigned	char(1) 0x000000
😜 [9]	'6' (0x36) unsigned	char(1) 0x000000
🐳 [10]	'8' (Ox38) unsigned	char(1) 0x000000
😜 [11]	'O' (Ox3O) unsigned	char(1) 0x000000
🔍 [12]	'8' (0x38) unsigned	char(1) 0x000000
😜 [13]	'O' (Ox3O) unsigned	char(1) 0x000000
😻 [14]	'' (OxOa) unsigned	char(1) 0x000000
😜 [15]	'' (OxOd) unsigned	char(1) 0x000000 *

2-4 sample4 割り込み

【 動作概要 】

sample4を動作させます。オシロスコープがあればPJ1 CN7 1番を観測すると、以下のような1msec毎の波形が観測できます。



TBS 1064 - 10:46:07 2015/08/18

 $\Delta \Sigma 2 4 b i t A / D のデータをUSBで出力する動作はsample3と同じです。$ 違いは

①sample3のウエイト

/ lwait(100000);

wtime = 1000; while(wtime != 0)

,

を止め、割り込みで時間を作成してあります。1msec×1000=1S毎にデータを出力します。

②1msec毎にデータを加算し、平均値を出力しています。

【 プログラム 】

省略

【解説】

省略

100回の平均化で上位5桁まで、1digitのちらつきで測定出来ました。

📒 COM80:38400baud - Tera Term VT	
ファイル(F) 編集(E) 設定(S) コントロール(O)) ウィンドウ(W) ヘルプ(H)
ch0 = 10479021	A
$h_{\rm ch0} = 104/9142$	
ch0 = 10479324 ch0 = 10479231	
ch0 = 10479182	
ch0 = 10479846	
ch0 = 10479118	
chU = 104/8865	
ch0 = 10479227 ch0 = 10478963	
ch0 = 10478800	
ch0 = 10478877	
ch0 = 10478917	
ch0 = 10478718	
ch0 = 10478900	
ch0 = 10479095	
ch0 = 10479449	
ch0 = 10479398	
ch0 = 10479238	
ch0 = 10479117	
ch0 = 10479232	E
]	•

- 2-5 sample5 PWM出力
- 2-6 三角、対数、平方根関数を使う

【 概要 】

| og、sin、√ 演算を行い、演算結果の確認とその速度を測定します。

【 プログラム 】

省略

【解説】

省略

ウォッチ1		
🖻 🦃 🐉 🖏 🗙 表記(N)- 🔤	
ウォッチ式	値 型情報(バイト数)	値 型情報(バイト数)
👽 d1	4.000000E+000 float (4)	00E+000 float (4)
👽 d2	7.071068E-001 float(4)	<mark>68E-001</mark> float (4)
😜 d3	1.414214E+000 float(4)	14E+000 float (4)
👽 s1	<mark>4 (0x0004)</mark> short(2)	0x0004) short(2)
🐳 s2	0 (0x0000) short(2)	0×0000) short(2)
🐳 s3	1 (0x0001) short(2)	0x0001) short(2)

演算速度ですが、

| og10 (10000) が約33μsec、sin(45°) が2. 5μsec、√2が3μsec程度 かかるようでした。例えばRL78(32MHz)では約220µsec、sin(45°)が130µs ec、√2が100µsec程度ですので、それぞれ6.6倍、52倍、33倍も速いことになります。 マイコンに必要な能力が演算処理速度の場合、RXを使用するのが圧倒的に有利であることが分かります。

RX21A (50MHz)



RL78 (32MHz)



2-7 D/Aにsin, cos演算した正弦波を出力する

【 概要 】

RX21Aがもつ、2ch 10ビットD/Aにsin, cos演算結果(最大±1)を0-3.3Vに 変換し出力します。正弦波オシレーターになります。



【 プログラム 】

省略

【解説】

省略

Oから359度まで、先に演算したデータをD/Aコンバータにセットしています。12.5KHz程度 の正弦波が得られています。毎回、演算をしていると速い波形は出せません。



それぞれはそれぞれの会社の登録商標です。 フォース®は弊社の登録商標です。

- 1. 本文章に記載された内容は弊社有限会社ビーリバーエレクトロニクスの調査結果です。
- 2. 本文章に記載された情報の内容、使用結果に対して弊社はいかなる責任も負いません。

3. 本文章に記載された情報に誤記等問題がありましたらご一報いただけますと幸いです。

4. 本文章は許可なく転載、複製することを堅くお断りいたします。

お問い合わせ先:

〒350-1213 埼玉県日高市高萩1141-1
TEL 042(985)6982
FAX 042(985)6720
Homepage: http://beriver.co.jp
e-mail: info@beriver.co.jp
有限会社ビーリバーエレクトロニクス ©Beyond the river Inc. 20150819