# RX230 マイコン学習セット マニュアル 入門編

第1版 2022.4.27

## 【 製品概要 】

本マニュアルはRX230 R5F52306ADFM(64ピン)マイコンを使ったマイコン学習セットの開発環境構築、ソフトウエアコピー手順、添付CDのサンプルプログラムの動作について解説しています。

●入門編ではマイコンの基本的なハードウエアのアクセス方法、プログラムの書き方をC言語サンプルプ ログラムを参考に学び、習熟度をチェックするために、演習プログラムの課題を自分で考えます。

●マイコンと各デバイスとのやりとりに標準的なI2C、SPIインターフェイスを使用、ライブラリの 使用法を具体例で示し、解説しています。

●ルネサスエレクトロニクス社の統合開発環境CS+ for CC における開発方法について記述してあります。

※本学習セット開発にはルネサスエレクトロニクス社製E2Lite(E2L)が別途必要です。



# 1. 学習環境、事前準備

- 1-1. 学習環境
  - a:学習セット 同梱物
  - b:BCRX230 CPU部の特徴
  - c: E2 Lite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⊐ピー

- f:RX230とRL78、RX71Mの速度比較
  - f-1:ポートアクセス速度の比較
  - f-2:乗除演算速度の比較
- 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: PEOを出力にする初期設定
    - b-2:自動生成されたプログラム
    - b-3:E2Lの設定
    - b-4:コード生成後の初期設定の変更
    - b-5:変数変化を実行中に見る ウオッチ窓の使い方
    - b-6:新しいプログラムを作る簡単な方法

# 2. サンプルプログラム

2 — 1.	Ⅰ/O入出力	sample1
	プログラム	: キー入力でLEDが点滅
	演習プログラム	:sample1_b 押されたキーの LED を点滅
2 – 2.	A/D変換	sample2
	プログラム	:12bitA/D変換器のデータを有機ELに表示
		I2Cインターフェイス
	演習プログラム	:sample2_a A/D変換の値を平均する
2 — 3.	D/A変換	sample 3
	プログラム	:12bit D/A変換器のアナログ出力で作る三角波
	演習プログラム	:A/D値をO-5Vに換算しパソコン側に送信
2 — 4.	FRAM	sample4
	プログラム	:FRAMの読み書き SPIインターフェイス
	演習プログラム	: データを変えて、読み書きできるか確認
2 — 5.	温度センサ	sample 5
	プログラム	:温度センサMCP9801を使う I2Cインターフェイス
	演習プログラム	:MCP9801を10bitデータにして温度表示
2 — 6.	PWM samp	l e 6
	プログラム	:PWMを使ってLEDの明るさを変える
	演習プログラム	:明暗の周期を2.5秒から0.25秒に変える
<b>2</b> — 7 .	USB通信 samp	l e 7
	プログラム	: パソコン側で打ち込んだキーを受信し、返送する
	演習プログラム	: 学習ボードのキーを押してパソコン側にABC他を表示する

# 1-1. 学習環境

a:学習セット同梱物 RX230学習ボード DVD(サンプルプログラム、ドキュメント) マニュアル(本誌) 入門、応用 USBケーブル モーター



※開発に必要なルネサスエレクトロニクス社製デバッカE2Lは同封されておりません。別途必要です。 但し、プログラムの検討、コンパイルは、無料のCS+(後述)で行うことが出来ます。

1

1

各 1

1

1

#### 複数の人間の学習において

A. E2L+本ボード+CS+インストゥール済みパソコンを用意 B. プログラムの検討、コンパイルは他のパソコンで行い、実行だけAのパソコンに席を移る といった使い方で、学習ボードを人数分用意しなくても、効率よく学習することは可能だと思います。も ちろん、各人に各台数あるのが、時間的な効率は一番良いです。

#### b:BCRX230 CPUボード部の特徴

学習ボードのマイコン部分は弊社BCRX230 CPUボードと同じです。

#### ●CPU特徴 R5F52306ADFM RX230

■ 32 ビット RXv2 CPU コア内蔵 最大動作周波数 54MHz / 88.56 DMIPS の性能(54MHz 動作時) DSP 強 化:32 ビット積和、16 ビット積差命令に対応• FPU 搭載:32 ビット単精度浮動小数点(IEEE754 に準 拠)• 除算器(最速 2 クロックで実行)• 高速割り込み• 5 段パイプラインの CISC ハーバードアー キテクチャ• 可変長命令形式:コードを大幅に短縮• オンチップデバッグ回路内蔵• メモリプロテクシ ョンユニット(MPU)対応 ■64 ピン 10×10mm 0.5mmピッチ IC

■消費電力低減機能 1.8V ~ 5.5V 動作の単一電源● バッテリバックアップ専用電源で動作可能な
 RTC● 3 種類の低消費電力モード● ソフトウェアスタンバイ中も動作する LPT (ローパワータイマ)●
 ■データ転送機能 DMAC:4 チャネル内蔵● DTC:4 種類の転送モード●

■ ELC 割り込みを介さず、イベント信号でモジュール動作が可能● CPU スリープ状態において、モジュール間のリンク動作が可能●

■リセットおよび電源電圧制御 パワーオンリセット (POR) など 8 種類のリセットに対応• 低電圧検出 機能 (LVD) の設定可能•

■クロック機能 メインクロック発振子周波数:1 ~ 20MHz• 外部クロック入力周波数:~ 20MHz• サ ブクロック用発振子周波数:32.768kHz• PLL 回路入力 4MHz ~ 12.5MHz• 低速オンチップオシレータ、 高速オンチップオシレータ、IWDT 専• 用オンチップオシレータ内蔵 USB 専用 PLL 回路:4MHz、6MHz、 8MHz、12MHz• システムクロック 54MHz USB クロック 48MHz の設定可能 32.768kHz RTC 専用クロックの 生成• クロック周波数精度測定回路(CAC)内蔵•

■リアルタイムクロック内蔵 補正機能(30 秒、うるう年、誤差)・ カレンダカウントモード / バイナ
 リカウントモードを選択可能・ 時間キャプチャ機能・ 外部端子のイベント入力で時間をキャプチャ・
 ■独立ウォッチドッグタイマ内蔵 15kHz IWDT 専用オンチップオシレータクロック動作●

■最大 14 本の通信機能を内蔵 USB2.0 ホスト / ファンクション /OTG (ON-The-Go) (1 チャネル)、• フルスピード (12Mbps)、ロースピード (1.5Mbps)、アイソクロナ ス転送、BC (バッテリチャージャ) に対応 ISO11898-1 準拠の CAN (1 チャネル)最大 1Mbps 転送• 多彩な機能に対応した SCI (最大 7 チ ャネル) 調歩同期式モード /• クロック同期式モード / スマートカードインタフェースモード ビットモ ジュレーション機能による通信誤差低減 IrDA インタフェース (1 チャネル、SCI5 と連携)・ I2C バス インタフェース 最大 400kbps 転送 SMBus に対応• (1 チャネル) RSPI (1 チャネル) 最大 16Mbps 転送• シリアルサウンドインタフェース (1 チャネル)・ SD Host I/F (オプション :1ch) SD メモリ /SDIO 1 or 4 ビット SD• バスをサポート 注. 48 ピン版は 1 ビットモードのみ

■最大 20 本の拡張タイマ機能 16 ビット MTU:インプットキャプチャ、アウトプットコンペア、• 相補 PWM 出力、位相計数モード(6 チャネル) 16 ビット TPU:インプットキャプチャ、アウトプットコンペ ア、• 位相計数モード(6 チャネル) 8 ビット TMR(4 チャネル)• 16 ビット CMT(4 チャネル)

■ 12 ビット A/D コンバータ内蔵 最小 0.83µs 変換が可能• 24 チャネル• チャネルごとにサンプ リング時間を設定可能• 自己診断機能 / アナログ入力断線検出アシスト機能内蔵

■ 12 ビット D/A コンバータ内蔵 2 チャネル

■汎用入出カポート内蔵 5V トレラント、オープンドレイン、入力プルアップ、駆動能力切● り替え機能
 ■セキュリティ機能(TSIP-Lite) 暗号エンジンへの不正アクセスを禁止し、成りすまし、改ざんを防止
 鍵の安全な管理を提供● AES(鍵長 128/256bit)内蔵。ECB,CBC,GCM 他に対応● 真正乱数発生回路内蔵●
 ■温度センサ内蔵 ■動作周囲温度 -40 ~+ 85 ℃● -40 ~+ 105 ℃

■用途 一般産業、民生機器

#### **CPU部大きさ(部品面)**





#### 概要

E2Liteエミュレータ(以降E2L)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 I C E しか出来なかった機能が、安価に実現されています。変数をウオッチ窓に登録 し、実行中を含めて数値を見ながらデバック出来ます。

また、使い方も比較的短時間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。

対応MPU RL78,RX,RA,RE



E2LをPCのUSBに接続するとwindowsが自動的にデバイスドライバをインスツールします。 続いて、ネットから開発環境CubeSuite+とCコンパイラの最新版ダウンロードを行います。

### d:無償のCS+、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+ for CC 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→「CS+ 無償ダウンロード」の検索で表示されます。

# 以下省略

# 1-2 動作、デバック

a:CS+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+を起動します。ここでは例としてRX230STUDY¥sample1を動作させます。キーを 押すとLEDが点滅するプログラムです。

初めてのときは ファイル → ファイルを開く → sample1.mtpjをダブルクリックしま す。

キュメント > WorkSpace > RX230_STUD	VY → sample1 →	ڻ ~	
			===
名前	状態	更新日時	種類
cg_src	0	2022/04/27 17:11	ファイル フォルダー
	0	2021/11/19 13:08	ファイル フォルダー
SmartManual Docs	0	2021/11/19 11:39	ファイル フォルダー
sample1.mtpj	0	2022/04/27 17:11	MTPJ ファイル

プロジェクトツリーが表示されます。 r \_\_main. cをダブルクリック。

00		
٢	sample1 - RX E2 Lite - CS+ fo	r CC - [r_cg_main.c]
ファ	イル(F) 編集(E) 表示(V) プロ	Iジェクト(P) ビルド(B)
8	🕅 スタート(S) 🔒 🗐	2009
9	▽ < ? 2 < ? < ? < < < < < < < < < < < < <	P 🗣 🔍   8   🦉
	プロジェクト・ツリー	Ψ×
V	2 0 2 2	
マート・マニュアル		マイクロコントローラ) ール) ル) チーツール) ゲーツール) ゲッール) ・ ・ tc c g.c c tb tc g.c c tb tc g.c c tb tc sc t.c tc sc c tb tc c sc tc c c tb tc c c tc c c tb tc c c user.c h h c c user.c h h c user.c h h h h h h h h h h h h h h h h h h h

r \_\_main. cが中央に表示されます。



とりあえず、実行してみます。E2Lのケーブルを基板のCN1に挿入します。電源はE2Lから供給しますので、USB接続は不要です。(写真ご参考)



「デバック・ツールヘプログラムを転送」をクリック。



正常に転送できると、今まで表示されていなかったプログラムの絶対アドレスが表示されます。E2L から電源3.3Vが基板に供給されます。



ここまでいかなかった場合、E2L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)		
₩ ₩ × ₩ ₩ 0 0 0 •	्रिड दुइ टेइ 👗	
	CPUリセット後、	プログラムを実行します。

4つのSW1, 2, 3, 4を押すとLED D1, 2, 3, 4が点滅したら正常に動作しています。CS +の右下部「RUN 実行中」が表示されます。

🗳 実行中 RUN RL78 E2 Lite

ここまで確認できましたら、一度止めます。	
へノレプ(H)	
, 🕼 🗘 🛏 💽 🕟 🛏 I 🖘 Çe 🛎 🔣	
 実行中のプログラムを停止します。	(Shift+F5)

main関数のwaitの数値の2か所、600000の6を2に減らして(3倍速)



セーブして

E)	表示(V)	プロ	ジェク	ト(P)	ビル	レド(	B) デ	バック	"(D)	ツー
•		X 🗈		0	88 <sup>4</sup>	<b>A</b> A	•		-	10
	<mark>ファ-</mark>		指定さ	れたエ	<u>東日</u> を アフカ	保存	マします ゴ r ma	t <mark>. ((</mark>	Ctrl+	s)

プログラムを変更しましたから「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の点灯が先ほどより、早く点滅する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。
	10 1110			ノーノノムとけ止しとのり。

へしプ(H)	
, , , , , , , , , , , , , , , , , , ,	
実行中のプログラムを停止します。	(Shift+F5)

マウスを111番にもって行き、左クリックで設定です。手の形が出ます。解除は設定後、再クリック。



「CPUリセット後、プログラムを実行」をクリック。

先ほど設定した行でSW1を押すとプログラムカウンタが停止します。黄色は現在のプログラムカウンタの位置です。まだコードは実行されていません。



# ステップオーバーで1行実行。



プログラムカウンターは 113 で、111 行のH\_ON が実行されました。LD1の点灯が確認できると思います。引き続きステップオーバー実行で 130 行まで行くと 129 行の命令 H\_OFF が実行されて LED D1 が消灯するのを確認できます。 /



ポートのデータを見たいときは例えば

#define LED\_ON PORTE.POOR.BIT.BD = 1 #define LED\_OFF PORTE.PODR.BIT.BO = 0 #define M\_ON PORTA.POOR.BIT.B6 = 1 #define M\_OFF PORTA.PODR.BIT.B6 = D #define H\_ON <u>PORTA.POOn DIT Do - 1</u> #define H\_OFF PORTA.PO #define TXD\_ON PORTA.R 🚳 解析グラフに登録(E) #define TXD\_OFF PORTA 2 アクション・イベントの登録(A)... #define S₩1 PORT5.PIDF ★ #define S₩2 PORTC.PIDF Ctrl+X 切り取り(T) #define SW3 PORTC.PIDF 🐚 ⊐ピー(C) Ctrl+C #define SW4 PORTC.PIDF 199 ■占い(付(†(P) Ctrl+V

#define H\_ON の PORTA.PODRの部分をマウスの左ボタンを押したままドラッグ、 離して、右クリックで「ウオッチ1に登録」で現在のPORTA.PODRの内容がウオッチ窓に表示さ れます。

ರೆಕ್ಸುಕ1			д
🗃 🧶 🏭 🗙 🕷	ŧ記(N)- 🕎		
ウォッチ式	値	型情報(バイト数)	アドレス メ
PORTA. PODR	0×10	IOR(1)	0x0008c02a

ブレークポイントを解除し「プログラムを現在の位置から実行」すると /
プログラムを現在の位置から実行します。(F5)

ウオッチ1窓にあるPORTA. PODRの値がSW1を押したときだけ、プログラムの進行に伴い、 0×10、0×11と繰り返すのが確認できると思います。

以上、大急ぎでしたが、CS+ のプログラムの書き換え、コンパイル、E2Lへのダウンロード、実行、 ブレークポイント設定、ウオッチ窓設定、動作の概要でした。 **b**:新しいプログラムを作る

# 以下省略

## b-3:E2Lの設定

出来上がったプログラムを実機にダウンロード、実行させるデバッカにはE2Lを使用します。初めはシ ュミレータになっているのでこれをRX E2 Lite(E)に変更します。



#### E2 Lite のプロパティ→ 接続用設定でクロック、ターゲットボードとの設定を行います。



RX E2 Lite のプロパティ	
✓ አモリ	
> メモリ・マッピング	[17]
メモリ書き込み時にベリファイを行う	はい
実行中のメモリ・アクセス	$\sim$
実行を一瞬停止してアクセスする	
表示更新間隔Lms」	500
V 1928	1.1.5
美行中にPC表示を行う	いいえ
◇ ノレージ 活性からには日子でゴー、万道からしの話覧	D. Porta b
優先に切たして用りるフレークホイントの種類	//~P/JL/*///-/
◇ ンズナム	()()>
(内)取フロクフム ROMで書き 深たるフロクラム です パジクタる	(1)
「11版ナーライノワンエで香きたんのフロクラムでナバックする」 実行時時かざにまた、したまたオス	(1)/2
大口はからかしにほとか リンク大口をつ	(1)/>>
	C ICIAL
トレース継貨の用途	ku-7
トレース・メデリを使いれった後の動作	トレース・メモリを上書きし実行を続ける
トレース・データ種別	分岐
タイム・スタンプルナ	()()ž
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、実行中に一瞬停止してアクセスするかどうかを指定します。 接続用設定、デバッグ・ツール設定、プウンロード・ファイル設定、フック処理設定	

E2 Lite のプロパティ→デバックツール設定 実行を一瞬停止してアクセスする→はい にすると

プログラム実行中のポートのレジスタ値、メモリ、変数の変化等を複数リアルタイムに確認できます。

ここまで設定出来たら、r\_cg\_main. cをクリックし、mainプログラムにプログラムを書き ます。



ポートをON/OFFする命令を書き、



E2Lと実機を接続し、



ビルド後、デバックツールヘプログラムをダウンロード をクリック。

6		мη				H.	•	93	ÇI	¢,	- Ma
Ł	シルド後	デバ	ッグ・ツ	-11/	プロ	グラム	をダ	<del>סעל</del> -	-FL	ŧす.	(F6)

プログラムにエラーが無かった場合、接続、ダウンロードが行われます。

進捗状 *****	況	×
ion.	RX E2 Lite (こ、接続処理中です。	
*****	エミュレータ・ファームウェアの更新が必要な場合、自動的に更新 接続が完了するまではUSBおよび電源は切断しないでください。	を行います。
i here		
	キャンセル	

CPUリセット後、プログラムを実行。

🖓 🞝 🦏 I 🔳 🕑 🕑	🕅 🚳 🖘 🖓 📩
	CPUリセット後、プログラムを実行します。

プログラム実行中は赤いマークと



右下にRUNという緑の文字が移動表示され、その右に実行中と黒文字で表示されます。

5000		出力
FIIIステップ・オーバー	FNステップ・イン	
本語 (シフト/IIS) )RUN	🗼 実行中	
26 € /‡	りのち晴れ へ 夏	

オシロスコープがあればTP17を観測すればPEOポートがO, 1を繰り返すのが確認できますが、目 視でも出来るようにウエイト関数を入れてみます。



## ウエイト関数を入れたプログラム



これをセーブ、コンパイル、ダウンロード、実行させるとLED D4が目で見てわかるほどの点滅になります。

b-4:コード生成後の初期設定の変更

# 以下省略

# b-5:変数変化を実行中に見る ウオッチ窓の使い方

動作中の変数を見る場合、見たい変数を左ドラッグし、

```
void main(void)

{
    R_MAIN_UserInit();
    /* Start user code. Do not edit comment generated here */
    while (1U)
    {
        PCRTE.PCOR.BIT.B0 = 1;
        Iwait(1000000);
        PCRTE.PCOR.BIT.B0 = 0;
        Iwait(1000000);
        Icount++;
    }
    /* End user code. Do not edit comment generated here */
}
```

右クリック→ウオッチ1に登録

	ウォッチ1 に登録(R)	>	
count ; 🚵	解析グラフに登録(E)		
5	アクション・イベントの登録	(A)	
(uint8 🐰	切り取り(T)	Ctrl+X	
	」 コピー(C)	Ctrl+C	
	h 貼り付け(P)	Ctrl+V	
8	検索(F)	Ctrl+F	
1	移動(G)	Ctrl+G	
0	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)	
Code	ここまで実行(H)		
R MA	PCをここに設定(P)		
*****	関数または変数ヘジャンフ	f(J) F12	
on i	タグ・ジャンプ(T)	Shift+F12	
alue 🖷	、 逆アセンブル ヘジャンプ(D)	)	
void)	ブックマーク(B)		≁
UserI	高度な設定(V)		≁
rt use (1U)	ブレークの設定(B)		•
PORT	トレース設定(T)		•
Iwai	タイマ設定(1)		
I wai	名前を付けて混合表示を	保存(S)	

ウオッチ1に1 countが表示され、プログラム実行と共に+1された表示を見ることが出来ます。

ウォッチ1		1
🗃 🧠 🛃 🗙 🕲 🗙	表記(N)▼ 🔤	
ウォッチ式	値型情報(バイト数)	アドレス
💊 Icount	13314… uint32_t(4)	0×00000404

同様方法で変数だけでなく、レジスタの値も読み込みできます。下記例では PORTE の出力レジスタ。

ウォッチ1			ų
😰 🔍 🧞 🐧 🗙 表記(N)+	Hey		
ウォッチ式 ● Icount	值 16803…	型情報(バイト数) uint32 t(4)	アドレス 0×00000404
PORTE.PODR	0×01	IOR(1)	0x0008c02e

プログラムでPEOが0,1と変化することが目視出来ます。

# b-6:ウオッチデータのセーブ、ロード

ウオッチ窓に作成したデータはセーブ、ロードできます。プログラムが停止している状態で、カーソルを ウオッチ窓に持っていき、右クリック→ファイル→ウオッチ・データを保存

🚯 test\_RX230 - RX E2 Lite - CS+ for CC - [ウォッチ1

新規作成(N)     ・       ファイルを開く(O)     Ctrl+O       エンコードを指定して開く(W)     ・       遠加(D)     ・       適加(D)     ・       ブロジェクトを閉じる(E)     ・       マークシェクトを閉じる(C)     ・       ブロジェクトを閉じる(C)     ・       ブロジェクトを保存(R)     Ctrl+Shift+S       マロジェクトを保存(R)     Ctrl+Shift+A       マクシッチ・データを保存(S)     Ctrl+Shift+A       ジネの保存設定(V)     ・       ジネの保存設定(V)     ・       ロ刷ブレビュー(N)     ・       シロッガ・デークが(F)     ・       シュンサンデア・クシンジェクトン(F)     ・	771	(ル(F) 編集(E) 表示(V	) プロジェクト(P) ビルド
マアイルを開く(の)…     Ctrl+O       エンコードを指定して開く(W)…     通加(D)       通加(D)     ・       ブロジェクトを閉じる(E)     ・       マーcg_main.cを閉じる(C)     ・       ブロジェクトを保存(R)     Ctrl+Shift+S       オ前を付けてプロジェクトを保存(T)…     ・       ウオッチ・データを保存(S)     Ctrl+S       対象の保存設定(V)…     ・       ゴロジロクトを保存(L)     Ctrl+Shift+A       ロ刷(P)…     Ctrl+Shift+A       印刷ブレビュー(0)     ・       シ近使ったプロジェクト(J)     ・       シ行(X)     ・		新規作成(N)	•
エンコードを指定して開く(W)           追加(D)           適加(D)           ブロジェクトを閉じる(E)           「ロジェクトを閉じる(C)           「ロジェクトを閉じる(C)           「ロジェクトを保閉じる(C)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロジェクトを保存(R)           「ロボを付けてプロジェクトを保存(R)           「ロボでを保存(L)           「ロ刷(P)           「ロ刷「レビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日刷「ブレビュー(R)           「日、「「」「「」」」」」           「日、「」」」」」」           「日、「」」」」」」」」           「日、「」」」」」」」」」」           「日、「」」」」」」」」」」」」」           「日、「」」」」」」」」」」」」」」」」」」」           「日、「」」」」」」」」」」」」」」」」」」」」」」           「日、「」」」」」」」」」」」」」」」」」」」           「」」」」」」」」」」」」」」」」」」」」」」」」」」」」」           「」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」」		ファイルを開く(O)	Ctrl+O
追加(D)     ・       ジロジェクトを閉じる(E)     ・       マロジェクトを閉じる(C)     ・       ジロジェクトを保存(R)     Ctrl+Shift+S       公前を付けてブロジェクトを保存(T)     ・       ・     クオッチ・データを保存(S)     Ctrl+S       ・     クオッチ・データを保存(S)     Ctrl+S       ・     クオッチ・データを保存(S)     Ctrl+Shift+A       ・     オ約をの保存設定(V)     ・       ・     オペでを保存(L)     Ctrl+Shift+A       ・     オペでを保存(L)     Ctrl+Shift+A       ・     オペでを設定(U)     ・       ・     ロ刷(P)     Ctrl+P       ・     印刷ブレビュー(0)     ・       ・     日のリブレビュー(0)     ・       ・     長近使ったプロジェクト(J)     ・       ・     条?(X)     ・		エンコードを指定して開く	.(W)
プロジェクトを閉じる(E)         r_cg_main.c を閉じる(C)         プロジェクトを保存(R)       Ctrl+Shift+S         名前を付けてプロジェクトを保存(T)         プカダ・データを保存(S)       Ctrl+S         ダネの 保存設定(V)         オペてを保存(L)       Ctrl+Shift+A         印刷(P)       Ctrl+Shift+A         印刷(P)       Ctrl+Shift+A         日刷プレビュー(I)       集近使ったファイル(F)         最近使ったプロジェクト(J)       ト         終了(X)       大		追加(D)	•
・.ccg_main.cを閉じる(C)           ブロジェクトを保存(R) Ctrl+Shift+S           な前を付けてブロジェクトを保存(T)           ・	5	プロジェクトを閉じる(E)	1
プロジェクトを保存(R)     Ctrl+Shift+S       名前を付けてプロジェクトを保存(T)       クオッチ・データを保存(S)     Ctrl+S       名前を付けて ウォッチ・データを保存(A)       対象 の保存設定(V)       ジネージ設定(U)       ロ刷(P)       Ctrl+P       自印刷(P)       こ日刷(P)       こ日、       日刷(P)       こ日、       日刷(P)       こ日、       日刷(P)       こ日、        こ日、 <t< td=""><td></td><td>r_cg_main.c を閉じる(C</td><td>(م (ت</td></t<>		r_cg_main.c を閉じる(C	(م (ت
<ul> <li>※ 名前を付けてプロジェクトを保存(T)…</li> <li>ウオッチ・データを保存(S) Ctrl+S</li> <li>名前を付けて ウォッチ・データを保存(A)…</li> <li>対象の保存設定(V)…</li> <li>すべてを保存(L) Ctrl+Shift+A</li> <li>ベージ設定(U)…</li> <li>ロ刷(P)…</li> <li>Ctrl+P</li> <li>印刷ブレビュー(T)</li> <li>最近使ったファイル(F)</li> <li>し</li> <li>※ 行(X)</li> </ul>		プロジェクトを保存(R)	Ctrl+Shift+S
ウオッチ・データを保存(S)       Ctrl+S         緑       名前を付けて ウォッチ・データを保存(A)         ヴ教の保存設定(V)       オ気の保存設定(V)         アペでを保存(L)       Ctrl+Shift+A         ロ刷(P)       Ctrl+Shift+A         印刷(P)       Ctrl+P         回       印刷ブレビュー(I)         最近使ったファイル(F)       ト         最近使ったプロジェクト(J)       ト         後7(X)       その	AB(C)	名前を付けてプロジェクト	、を保存(T)
認     名前を付けて ウォッチ・データを保存(A)       ジ検 の保存設定(V)       ジレ クスでを保存(L)     Ctrl+Shift+A       ロジレ クスの設定(U)       ロジレビュー(I)       シレ クスのよりレビュー(I)       しつ ジレビュー(I)       しつ ジレビュー(I)       シレ クスのよりレビュー(I)       シレ クスのよりレビュー(III)       シレ クスのよりレビュー(IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII		ウォッチ・データ を保存(S	i) Ctrl+S
<ul> <li>対象の保存設定(V)</li> <li>すべてを保存(L) Ctrl+Shift+A</li> <li>ページ設定(U)</li> <li>印刷(P) Ctrl+P</li> <li>印刷ブレビュー(I)</li> <li>登近使ったファイル(F)</li> <li>長近使ったプロジェクト(J)</li> <li>終了(X)</li> </ul>	ABC	名前を付けて ウォッチ・ラ	データ を保存(A)
すべてを保存(L)         Ctrl+Shift+A           パージ設定(U)            印刷(P)         Ctrl+P           日刷ブレビュー(I)            最近使ったファイルレ(F)         ト           最近使ったプロジェクト(J)         ト           後了(X)	R	対象 の保存設定(V)	
ページ設定(U)       印刷(P)       Ctrl+P       日刷プレビュー(I)       最近使ったプロジェクト(J)       後了(X)	9	すべてを保存(L)	Ctrl+Shift+A
印刷(P)     Ctrl+P       印刷ブレビュー()        最近使ったファイル(F)     ト       最近使ったプロジェクト(J)     ト       後了(X)		ページ設定(U)	
印刷プレビュー(I)       最近使ったファイル(F)       最近使ったプロジェクト(J)       ※		印刷(P)	Ctrl+P
最近使ったファイル(F)         ト           最近使ったプロジェクト(J)         ト           総了(X)         ト		印刷プレビュー(I)	
<ul> <li>最近使ったプロジェクト(J) ト</li> <li>※ 終了(X)</li> </ul>		最近使ったファイル(F)	۲
🔞 終了(X)		最近使ったプロジェクト(」	J) •
		終了(X)	

以下省略

# 2. サンプルプログラム

2-1 sample1 キー入力でLEDが点滅

# 【 概要 】

出力ポート、入力ポートのプログラム

出力ポートとして4つのLED(LD1,LD2,D3,D4)を点滅させるプログラム。入力ポートSW1, 2,3,4各々のスイッチを押している間、各々のLEDが点滅します。電源はE2Lから供給されます。



【 ハードウエア 】

プッシュスイッチ SW1, 2, 3, 4は以下のようにPC4, 5, 6, 7に接続されています。



プッシュスイッチは押すと通電するタイプで、この回路図では押すとGNDと同電位=0Vとなります。

ສັດສິະ <b>ກ</b> ະ	¥	The second s	-						
+	^	🚰 วิติ/วิส	cg_main.c	22 周辺機能	/				
2 ② 3 図 - ⊼ sample1 (プロジェクト) - ■ RSF52306A×FM (マイクロコントローラ)	^	<sup>1</sup> 10 コードを生成する Port0 Port1 Por P54	t2 Port3	8 🖄 📽 🖗	■ \$■ ∰ Za 😂 (C PortA PortB PortC	) 🧐 😗 🧐 🧭 PortE PortH	n 🗔 🗞 🖉 1	<b>5 7 18</b> .	罗뱀朵帆视
□***********************************		- 1 64	◉ 入力	〇出力	☑ 内蔵ブルアップ	CMOS出力	×	□ 1を出力	□ 高駆動出力
		● 使用しない	О∧л	() 出力	□ 内蔵プルアップ			□ 1を出力	□ 高駆動出力

SW1が接続されているP54のI/Oポート設定は入力、内蔵プルアップとなっています。

#### SW2, 3, 4が接続されているPC6, 5, 4も同じです。

PC4							
○ 使用しない	● 入力	〇出力	🗹 内蔵ブルアップ	CMOS出力	~	□ 1を出力	🗌 高駆動出力
_ PC5							
○ 使用しない	● 入力	〇 出力	🗹 内蔵ブルアップ	CMOS出力	~	🗌 1を出力	📃 高駆動出力
_ PC6							
○ 使用しない	● 入力	〇出力	🗹 内蔵プルアップ	CMOS出力	~	□ 1を出力	🗌 高駆動出力



R? が内蔵プルアップを示します。SW1が押されると、SW1は通電しますのでP54電圧レベルは OVになります。離すとP54は電源電圧=3.3Vになります。プログラム的には押されたのを知るた めにはP54がOかどうか?を見ればいいことになります。外部に抵抗を付けなくて済むので、コスト安、 基板を小さく出来るメリットがあります。

内蔵プルアップはシリコン上に抵抗を作っているわけですが、抵抗値の精度は10K~50KΩと5倍 もばらつきます。しかしプルアップという使い方では0または1が検出できればいいので、問題になりま せん。

表50.6 DC特性 (4) 条件:1.8V≦VCC=VCC USB=AVCC0≦5.5V、VSS=AVSS0=VSS USB=0V、T<sub>a</sub>=-40~+105°C

項目			min	typ	max	単位	測定条件
入力プルアップ抵抗	全ポート (ポート35以外)	R <sub>U</sub>	10	20	50	kΩ	V <sub>in</sub> = 0V

#define SW1 PORT5.PIDR.BIT.B4

//SW1が押されるとLED D1 ON



出力はLED D3, D4はポート直接、LD1、LD2はトランジスタMJD44H11を介してLE Dがドライブされています。トランジスタは電流増幅素子で、ポートが出力できる電流で足りないので追 加されています。ポートが出力できる電流はハードいウエアマニュアル 50. 電気的特性 によれば通 常で4mA、高駆動出力時で8mAです。

RX230グループ、RX231グループ

50. 電気的特性

	項目		記号	max	単位
出力Lowレベル許容電流 (1端子あたりの平均値)	ボート40~47、ボート	03、05、07、ポート36、37	ポート36、37 loL 4.0		mA
	それ以外のポート	通常出力時		4.0	3
		高駆動出力時		8.0	

表50.16 出力許容電流値(1) 条件:1.8V≤VCC=VCC USB=AVCC0≤5.5V、VSS=AVSS0=VSS USB=0V、T<sub>2</sub>=-40~+85°C

それに対して、Q1にはR13に流す電流Aと、R11に流す電流Bの合計が流れます(重ね合わせの理)。



Q1がONの時のC コレクタ - E エミッタ 間の抵抗を便宜上OΩとすると、電流Aは (3.3V - LD1のVF) /1KΩ です。LD1のVF 順方向降下電圧を1.8Vとすると、 電流A = (3.3-1.8) /1KΩ = 1.5mA となります。 電流Bはバイメタルの抵抗をOΩとすると、3.3V/24Ω = 137.5mA となります。 Q1が制御する電流 = 1.5 + 137.5 = 139mA となります。 139mA >> 4mA で、ポートだけではドライブできないのが明らかで、トランジスタQ1が入 れられています。LED D3, D4は負荷がLEDだけなので、ポートで直接ドライブしています。

### 【 出力ポートの設定、コード生成 】

出力ポートはPAO, PA6, PEO, PA4は出力、CMOS出力に設定されています。 Port0 Port1 Port2 Port3 Port4 Port5 PortA PortB PortC PortE PortH -PA0 〇 使用しない 〇 入力 ④ 出力 □ 内蔵ブルアップ CMOS出力 □ 1を出力 □ 高駆動出力

# 【 プログラム 】

/\* Start user code for global. Do not edit comment generated here \*/

()#define LED\_ON PORTE.PODR.BIT.B0 = 1 #define LED\_OFF PORTE.PODR.BIT.B0 = 0

#define M\_ON PORTA.PODR.BIT.B6 = 1
#define M\_OFF PORTA.PODR.BIT.B6 = 0

#define H\_ON PORTA.PODR.BIT.B0 = 1
#define H\_OFF PORTA.PODR.BIT.B0 = 0

#define TXD\_ON PORTA.PODR.BIT.B4 = 0
#define TXD\_OFF PORTA.PODR.BIT.B4 = 1

```
#define SW1 PORT5.PIDR.BIT.B4
#define SW2 PORTC.PIDR.BIT.B6
#define SW3 PORTC.PIDR.BIT.B5
#define SW4 PORTC.PIDR.BIT.B4
```

```
②void lwait(long time)
```

```
while(time != 0)
{
    time--;
}
```

/\* End user code. Do not edit comment generated here \*/

③void main(void)

# {

{

}

R\_MAIN\_UserInit();

/\* Start user code. Do not edit comment generated here \*/

while (1U)		//この中を無限ループ
{		
4	<b>if</b> (SW1 == 0)	//SW1 が押されると LED D1 ON
	{	
5	H_ON;	
	}	
	if(SW2 == 0)	//SW2 が押されると LED D4 ON
	{	
	M_ON;	
	}	
	if(SW3 == 0)	
	{	
	TXD_ON;	//SW3 が押されると LED D3 ON
	}	
	<b>if</b> (SW4 == 0)	
	{	
	LED_ON;	//SW4が押されるとLED D4 ON
	}	

6	lwait(600000);		1,	/ウエ ⁄	٢ŀ	
	H_OFF;		1,	/LED	D1	OFF
	LED OFF;	//LED	D4	OFF		

M_OFF;	//LED	D2	OFF
TXD_OFF;	//LED	D3	OFF
lwait(600000);	//ウエ~	٢ト	

}

## 【 解説 】

```
    ①#define LED_ON PORTE.PODR.BIT.B0 = 1
    LED_ONをPE0=1と定義しています。 LED D4を点灯するためにPE0ポートに1を出します。
```

```
②void lwait(long time)
```

```
while(time != 0)
{
    time--;
}
```

}

{

ウエイト関数です。timeが0になるまでループします。

③void main(void)

{

R\_MAIN\_UserInit(); /\* Start user code. Do not edit comment generated here \*/

```
while (1U) //この中を無限ループ
{
```

電源がONすると末尾にあるシステム初期設定 R\_MAIN\_UserInit();を行い、while(1U)の中を無限ループします。コード生成で作成したポートの初期設定等はここに来る前のr\_cg\_resetprg.c、r\_cg\_hardware\_setup.c 等で実行されています。

```
④ if(SW1=0) //SW1が押されるとLED D1 ON {
```

SW1 PORT5.PIDR.BIT.B4 ポート54が0 (スイッチ1が押されている) なら

⑤ H\_ON;

}

H\_ON PORTA.PODR.BIT.B0=1 ポートAOに1を出します。トランジスタQ1がONし、LD1が点灯します。

6

lwait(600000);	//ウエ~	イト		
H_OFF;	//LED	D1	OFF	
LED_OFF;	//LED	D4	OFF	
M_OFF;	//LED	D2	OFF	
TXD_OFF;	//LED	D3	OFF	
lwait(600000);	//ウエ -	ſト		

ウエイト時間を置いて、出力をオフ = 消灯しています。その後、ウエイト時間を置いて再びキーを読 みに行きます。

#### 【 演習問題 】

sample1の動作を理解できた方は、演習問題に進んで下さい。課題は SWを押さないとLD2、D3、D4が点滅する。SW1を押すと全LED点滅しない。SW2を押すと D4が消灯、SW3を押すとLD2が消灯、SW4押すとD3が消灯するプロフラム。

回答例 sample1\_b

回答例は例であり、課題に合致していれば、書き方がこの通りである必要はありません。

# 2-2 sample2 A/D変換データを有機ELディスプレイに表示する I2Cインターフェイス

# 【 概要 】

P40/AN000入力に入力される電圧をA/D変換器でアナログ→デジタル変換し、VR1を回すことにより0-3.3VのスパンでI2Cインターフェイスで有機ELディプレイに表示します。

電源はE2Liteから供給されます。



【 ハードウエア 】



VR1とA/D変換器 AN0000間にはオペアンプMC33202によるバッファ(インピーダンス 変換器 電圧増幅しない)が1段入っています。この理由はRX230マイコンのA/Dコンバータが0. 83μsの高速変換を実現するために、信号源インピーダンスが0.5KΩ以下であること、という条件 があるためです。

#### 43.7 許容信号源インピーダンスについて

本 MCU のアナログ入力は、高速変換 0.83µs を実現するために、信号源インピーダンスが 0.5kΩ 以下の入 力信号に対し、変換精度が保証される設計となっています。シングルスキャンモードで1端子のみ変換を行 ハードウエアマニュアル 12ビットA/Dコンバータ 許容信号源インピーダンス の項参照 174 6ページ

オペアンプは入力インピーダンスは大きく、出力インピーダンスは低くという特性を持っているので、このようなバッファでインピーダンス変換する場合にも使われます。

ところで、プログラムを動作させて見ると、VR1 TP10を0Vにしても、TP11ががゼロにならず、有機EL表示器もゼロが表示されないという現象があります。



測定してみるとオペアンプ入力のTP10は0.000Vですが、オペアンプ出力のTP11は0.04 V前後でした。理想的なオペアンプでは入力0Vの場合、出力も0になるのですが、実際は入力オフセッ ト電圧と入力バイアス電流、レイルツーレイル構造の影響で0になりません。





MC33202 オペアンプは入出力レイルツーレイルオペアンプなのでゲイン1の場合、3.3V入力 に対して出力3.3V出て欲しい所ですが、実際は出ません。アウトプットボルテージスイング (出力電圧スイング)によれば電源電圧5.0Vに対して4.95V、負荷電流が増えると4.85V

Characteristic	Figure	Symbol	Min	Тур	Max	Unit
Input Offset Current (V <sub>CM</sub> = 0 V to 0.5 V, V <sub>CM</sub> = 1.0 V to 5.0 V)	 	110				nA
$T_A = + 25^{\circ}C$			() <u>-</u> (	5.0	50	
$T_A = -40^{\circ} \text{ to } +105^{\circ}\text{C}$				10	100	
T <sub>A</sub> = - 55° to +125°C			-	() <del>,</del>	200	
Common Mode Input Voltage Range	-	VICR	VEE	-	Vcc	V
Large Signal Voltage Gain (V <sub>CC</sub> = + 5.0 V, V <sub>EE</sub> = - 5.0 V)	7	Avol				kV/V
$R_L = 10 k\Omega$		1922	50	300	1.7	
$R_L = 600 \Omega$			25	250	170	
Output Voltage Swing ( $V_{ID} = \pm 0.2 V$ )	8, 9, 10			0.00		V
$R_L = 10 k\Omega$		VOH	4.85	4.95	-	
$R_1 = 10 k\Omega$	1	Vol		0.05	0.15	
$R_1 = 600 \Omega$		Voh	4.75	4.85	-	
$R_1 = 600 \Omega$		Voi		0.15	0.25	

DC ELECTRICAL CHARACTERISTICS (cont.) (V<sub>CC</sub> = + 5.0 V, V<sub>FE</sub> = Ground, T<sub>A</sub> = 25°C, unless otherwise noted.)

など、電源電圧5.0Vに対して出力は電源電圧まで出力されません。 また、出力の最小値も0.05V~0.15Vと0Vにはならないようです。

A/Dコンバータの入力特性上、入出力レイルツーレイルオペアンプを単電源で使用していますが、上 下限近辺のデータには注意が必要になります。

表示に使われている有機EL SO1602AWWB ですが、表示素子自身が光りますので、液晶に比べ視認性が良い、応答速度が高速などの特徴があります。インターフェイスはI2Cで、信号線は2本しか使いません。本機はマイコンから有機ELにデータを出力するだけの結線となっています。

# 【 コード生成 】

コード生成は12ビットA/Dコンバータの項を以下のように設定されています。本サンプルではソフト ウエアでA/D変換を開始しています。

ファイル(F) 環集(E) 表示(V) プロジェクト(P) ビルド(B)	デバッグ(D) ツール(T) ウインドウ(W) ヘルプ(I					9
🚳 27-HS 🔒 🗃 🍠 🐰 🖄 🖄 🤊	6 日本 1	0% 👻 👦 👦 DefaultBuild 🔹 🔩 🖓 🕒 🦏 🖉	9 🖍 🛞   ea Ca 🖓   👯			
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	▲ ソリューション一覧(S)					
JDジェクト・ツリー # ×	四フロバティ 予道アセンブル1 Mr.se.	ano 🔛 Millitite	* >	08051		
2 3 2 3	C D UL C C C C C		A & 28 51	- 🗷 🔫 🐉 🚺	× 表記(N)• ◎	
4 B- 3 sample2(70917h) ^	0 3-16 T WAA TO 0 00 D		A * 1 10 4	ウォッチ式	値 聖情報(パイト数)	アドレス
7	8年1 ウィンドウA 設定 ウィンドウB 該	1	·	ibe 🤪	4068 (0x0fe5) uint16_t(2)	0x0001006e
- 1 - ド生成(設計ツール)	-S12AD 動作設定			e fdatal	8.278242E+000 float(4)	0x00010470
8 8-2 端子図	○使用しない	<ul> <li>使用する</li> </ul>		e sdata	2 1	1
日本語用型機能	約代モードの設定					
□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	C TORGE	○ /// / / / / / / / / / / / / / /				
クロック周波教緒度測定回路		0.000-20000-0000-0000000-0				
→● 消費電力低減機能		0.07				
	© #E	0 (#4)				
KZ	- A/D实换動作選択E9					
	④ 高速支流動作	①電流実換動作				
⊕ ♥ DMA32/E0-5	自己的账 稳定					
	-Bussin and Euck	NOT A SEC.				
21.479/03/PU-5						
■ ポートアウトブットイネーブル2	使用電圧	■W0/電圧を使って自己診断を行う ~				
⊕ ♥ 8ビットタイマ	- 新線検出 アウスト 設定					
● ● コンペアマッチタイマ	チャージ 時家	未使用				
⊕ 📦 16ビットタイマバルスユニット						
	5-4	2 AUGLK				
	- グループスキャン優先 設定					
	グループA優先設定	グループAの優先刺導動作を行わない				
	All - TOTALIN'S	A (1)(生)(株計)(2)(1)(1)(2)(1)(1)(2)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)				
	500 7075745	Pri 626 (Prilo) ( PHIOTOCOPHISARO 2, 0357				
ロード シリアルコミュニケーションインタフ	-A / D212(030) 19/2					
- Contra 7777	O mWH-NB	Emi-k				
12E9FA/D32/(-9)	C mile i	e inc i				
12/15/D/A 12/16-D	-高電位側基準電圧選択ビット					
⊕ ♥ コンパレ-98	AVCC0	○ VREFH0				
テータ演算回路	任要位用基准要任遵护的。			<		
B-B D-N-70Ea-	AVSSI	O VREEL0		(1-カル実験 (1))	SPU1/929 🗔 ウォッチ1	
S PV Faller (Elfedition)		0.000		出力		
(1000000000000000000000000000000000000	-ワインドワ機能設定			BFIE2LRX.s.J	1 40 40 4041	
B-1 7711	(U) #L	C Heal		LEVEL1	1.07.00.000	
T 20 de de la section de	- ワインドワ A動作用来明 統定			Comuni.dll	3.02.00.000.	

# 【 プログラム 】

#### void main(void)

# {

1

R\_MAIN\_UserInit();

/\* Start user code. Do not edit comment generated here \*/

//有機 EL イニシャル init SO1602(); while(1) { ② S12AD.ADCSR.BIT.ADST = 1; //AD スタート while(S12AD.ADCSR.BIT.ADST) //変換時間待ち ; 3 ad0 = S12AD.ADDR0;//データ読み込み fdata1 = ad0/(4095/3.3);//AD データ→電圧換算 3.3V で 4095 sprintf(ad\_buff,"AD0= %.2fV",fdata1); //表示バッファにデータ設定 LED ON; //マーカー (4) lcd\_puts(0,ad\_buff); //1 行目 数值表示 //2 行目 バー表示 lcd\_bar\_disp(1,ad0); LED\_OFF; //マーカー }

/\* End user code. Do not edit comment generated here \*/

}

【 解説 】

(1)	init	SO1602()	;
$\sim$	_		~

//有機 EL イニシャル

**有機 E L のイニシャルを行っています。詳細は#include** <lcd\_so1602\_rx230.c> //有機 EL ライブラリ をご参照ください。

while	(1)
{	

;

2	S12AD.ADCSR.BIT.ADST = 1;	//AD スタート
	while(S12AD.ADCSR.BIT.ADST)	//変換時間待ち

ADSTビットを1にしてA/D変換開始。1になると変換終了です。

3	ad0 = S12AD.ADDR0;	//データ読み込み
	fdata1 = ad0/(4095/3.3);	//ADデータ→電圧換算 3.3V で 4095
	<pre>sprintf(ad_buff,"AD0= %.2fV",fdata1);</pre>	//表示バッファにデータ設定

データを読み込み、3.3V に換算し、表示のためのアスキーコードに変換してます。内容は右手のウオッチ 窓でリアルタイムに確認できます。表示されていない場合、ウオッチ式をインポートして表示させてくだ さい。**b-5:変数変化を実行中に見る ウオッチ窓の使い方**参照

ರೆಕ್ಸಲ್ರಕ1			4 ×
2 🛞 🛃 🥲	× 表記(N) - 世感		
ウォッチ式	値	型情報(バイト数)	アドレス メモ
😜 ad0	4068 (0×0fe4)	uint16_t(2)	0×0000006e
😜 fdatal	3.278242E+000	float(4)	0×00000470
🖂 😜 ad_buff	"AD0 = 3.28 V"	uint8_t [20](20)	0×0000008
😜 [0]	'A' (0×41)	uint8_t(1)	0×0000008
😜 [1]	'D' (0×44)	uint8_t(1)	0×0000009
😜 [2]	'0' (0×30)	uint8_t(1)	0×000000a
😜 [3]	'=' (0×3d)	uint8_t(1)	0×000000Ь
😜 [4]	''(0×20)	uint8_t(1)	0×000000c
😜 [5]	'3' (0×33)	uint8_t(1)	Dx000000d
😜 [6]	'.' (0x2e)	uint8_t(1)	0×000000e
😜 [7]	'2' (0×32)	uint8_t(1)	0×000000f
😜 [8]	'8' (0×38)	uint8_t(1)	0×00000010
😜 [9]	'V' (0×56)	uint8_t(1)	0×00000011
😜 [10]	'' (0x00)	uint8_t(1)	0×00000012
😜 [11]	'' (0x00)	uint8_t(1)	0×00000013
😜 [12]	'' (0x00)	uint8_t(1)	0×00000014
😜 [13]	'' (0x00)	uint8_t(1)	0×00000015
😜 [14]	'' (0x00)	uint8_t(1)	0×00000016
😜 [15]	'' (0x00)	uint8 t(1)	0×00000017
😜 [16]	'' (0x00)	uint8_t(1)	0×00000018
😜 [17]	'' (0x00)	uint8_t(1)	0×00000019
😜 [18]	'' (0x00)	uint8_t(1)	0×0000001a
💊 [19]	'' (0x00)	uint8_t(1)	0×0000001b
and the second second second		a de la companya <del>de</del> segundos	2002/00/00/00/20/20/20/20/20

	LED_ON;	//マーカー
4	<pre>lcd_puts(0,ad_buff);</pre>	//1 行目 数值表示
	<pre>lcd_bar_disp(1,ad0);</pre>	//2 行目 バー表示
	LED_OFF;	//マーカー

I c d \_ p u t s () は有機 E L に数値を表示する関数です。 I c d \_ b a r () は数値をバーグラフで表示する関数です。 いずれも詳細はライブラリ ご参照下さい。

【 演習 】

A/Dデータを1回ではなく、何回か平均して表示するプログラムを作成してみてください。平均回数を 変えると数値のバラツキ、安定性が変化することを確認できると思います。

参考プログラム sample2\_b

2-3 sample3 D/Aコンバータで三角波を作る

#### 【 動作概要 】

DAコンバータ、定周期割り込みの使い方を学習します。

1msec毎にDAOの電圧レベルを変えて三角波形を作っています。 PO3/DAO CN8の1番、TP7で波形観測できます。





それぞれはそれぞれの会社の登録商標です。

フォース及びFORCE®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. 20190204