

Renesas RX63N 用サンプル(ベアメタル版)の説明

(EV-RX/RZ-xx+MP-RX63N/FPGA|CPLD-xx 対応)

1. Sample の免責について

- Sample に関する Tel/Fax でのご質問に関してはお受けできません。ただし、メールでのご質問に関してはお答えするよう努力はしますが、都合によりお答えできない場合もありますので予めご了承ください。
- Sample ソフトの不具合が発見された場合の対応義務はありません。また、この関連ソフトの使用方法に関する質問の回答義務もありませんので承知の上ご利用下さい。
- Sample ソフトは、無保証で提供されているものであり、その適用可能性も含めて、いかなる保証も行いません。また、本ソフトウェアの利用により直接的または間接的に生じたいかなる損害に関しても、その責任を負わないものとします。

2. サンプル (ベアメタル版) のプロジェクト名

サンプルプロジェクト名		
USER_Debug_rx	MCU 基板(MP-RX63N/FPGA CPLD-*) 単体サンプル	ソース公開
EVRXrz_Sample	MCU 基板(MP-RX63N/FPGA CPLD-*) 評価用基板(EV-RX/RZ-*)用サンプル	ソース公開
EVRXrz_Sample_USB	MCU 基板(MP-RX63N/FPGA CPLD-*) 評価用基板(EV-RX/RZ-*) USB-Function 機能を追加したサンプル	実行ファイルのみ添付

統合開発環境	コンパイラー
HEW4(バージョン 4.09.01.007)	RXC(バージョン V.1.02.01.000)
CS+ for CC V3.00.00	CC-RX(バージョン V.2.02.00)

HEW4 の環境をベースにして、CS+は全てサブプロジェクトとしてソースおよびヘッダファイル全てを HEW4 のフォルダから登録しています。

最終生成物(*.mot/*.abs)は、Hew4(Debug)/CS+(DefaultBuild)に作成されます。

C ソースに #fdef 等のマクロ定義している場合に使用します。	
ITF_LIB	USB-Function 使用時に定義
RXC	USB-Function 使用時に定義

サンプルプロジェクト別に必要なマクロ定義例				
EVRXrz_Sample				
EVRXrz_Sample_USB	ITF_LIB	RXC		

2-1. 「USER_Debug_rx」プロジェクトの説明

1) USER_Debug_rxの動作

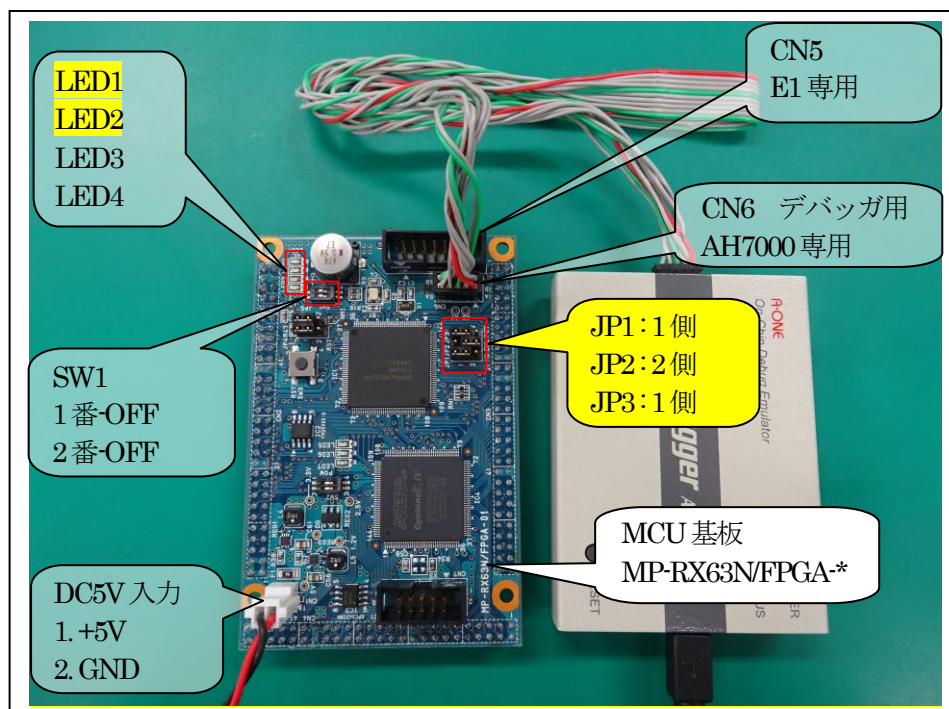
- ・デバッグツール「DEF」等にて「USE_Debug_rx.abs」をダウンロードして実行させる。
- ・基板上的LEDをソフトタイマー10msecを10回ごとにLED1を点滅
- ・基板上的LEDをCMTタイマー割り込みによりLED2を20msec毎に点滅

2) フォルダ構成とファイル名

Sample_RX\EV\RZRZ\Hew4\USER_Debug_rx		【HEW4】	
Debug	ビルドにより生成された実行ファイル等が格納される場所		
src_app	inc	src_appのインクルード用ディレクトリ	
	main.c	メイン処理	
	board.c	LED・SW等の処理ソフト	
	cmt.c	CMTタイマー処理ソフト	
src_sys	inc	src_sysのインクルード用ディレクトリ	
	cpg.c	CPG設定の基本ソフト	
	dbset.c	セクション管理テーブル	
	idcode.c	IDCODE管理テーブル	
	intprg.c	割り込みハンドラー	
	lowsrc.c	低水準I/O関数	
	ofs.c	OFS管理テーブル	
	resetprg.c	リセットプログラム	
	sbrk.c	メモリアロケーション	
	vecttbl.c	リセットベクターテーブル	

Sample_RX\EV\RZRZ\Cube\USER_Debug_rx		【CS+】
DefaultBuild	ビルドにより生成された実行ファイル等が格納される場所	
lnk_app	<Hew4><USER_debug_rx><src_app>にリンク	
lnk_sys	<Hew4><USER_debug_rx><src_sys>にリンク	

3) 動作構成



2-2. 「EVRXrz_Sample」プロジェクトの説明

1) 動作説明

- ・Tera Term からのコマンド指示により各デバイスを動作させる。
- ・各コマンド体系は後記にて説明します。

2) フォルダ構成とファイル名

Sample_RX¥EVRXrz_Sample			【HEW4】
	Debug	ビルドにより生成された実行ファイル等が格納される場所	
	src_app	inc	src_app のインクルード用ディレクトリ
		main_s.c	メイン処理
		board.c	LED・SW 等の処理ソフト
		bsc.c	BSC 初期化処理
		cmt.c	CMT タイマー処理ソフト
		rtc.c	RTC の初期化と処理ソフト
		sfram.c	FRAM の初期化と read/write 処理
		lvc.c	LVC の初期化と LVC 電源断検出のサンプル
		command.c	コマンド処理
	src_eva	inc	src_eva のインクルード用ディレクトリ
		e2p.c	EEPROM の read/write 処理
		riic.c	RIIC の初期化と read/write 処理
		rcan.c	RCAN の初期化と read/write 処理
		sci0.c	SCIO の初期化と read/write 処理
		usb_func.c	ITF_USBLib の使用サンプル
	src_sys	inc	src_sys のインクルード用ディレクトリ
		cpg.c	CPG 設定の基本ソフト
		dbst.c	セクション管理テーブル
		idcode.c	IDCODE 管理テーブル
		intprg.c	割り込みハンドラー
		lowsrc.c	低水準 I/O 関数
		ofs.c	OFS 管理テーブル
		resetprg.c	リセットプログラム
		sbrk.c	メモリアロケーション
		vecttbl.c	リセットベクターテーブル

Sample_RX¥EVRXrz_Sample			【CS+】
	DefaultBuild	ビルドにより生成された実行ファイル等が格納される場所	
	lnk_app	空	Hew4¥ EVRXrz_Sample¥src_app にリンク
	lnk_eva	空	Hew4¥ EVRXrz_Sample¥ src_eva にリンク
	lnk_sys	空	Hew4¥ EVRXrz_Sample ¥src_sys にリンク

- 3) コマンド実行を指示するため「TeraTerm Pro」をインストールする。
- ① 「teraterm-4.80.exe」を検索してダウンロードする。
 - ② PCにインストールし実行する
 - ③ シリアルポートの設定

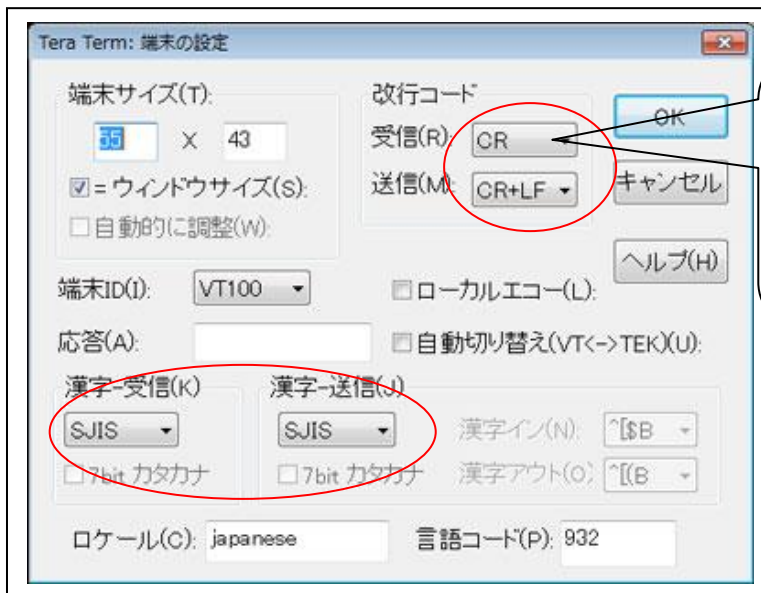


COM 番号は、
PC 側でシリアル
通信可能な番号を
指定する。

115200BPS
8bit
none
1bit
none

の仕様にする。

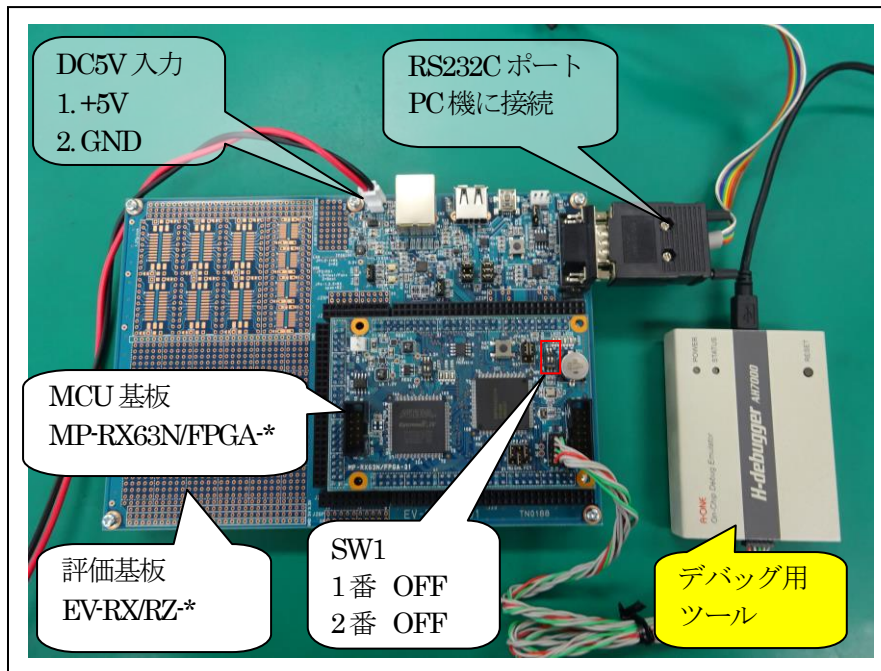
④ 端末の設定



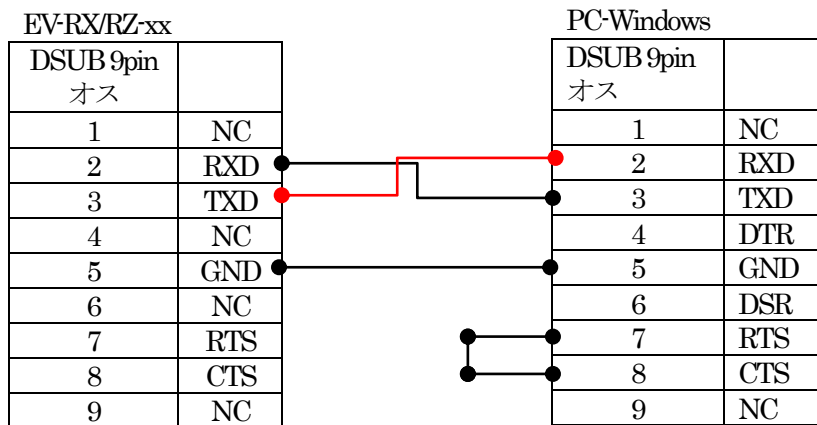
USB シリアルコン
バータ使用時に
CR コードがカット
される設定の場合
は、受信 : LF
にして下さい。

赤丸の設定にする。

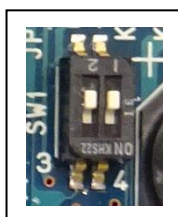
4) 動作構成



- ①PC機と接続するRS232Cケーブルは、市販「クロスケーブル」でも可能です。
- ②USB-シリアル変換ケーブルを使用される場合は、「StarTech.com 社 ICUSB232FTN」を推奨
- ③自作する場合は、下記の配線になります。

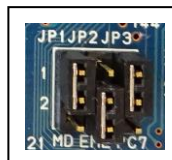


④MCU 基板上的 SW1 設定



SW1-1 **OFF**
SW1-2 **OFF**

⑤MCU 基板上的 JP 設定



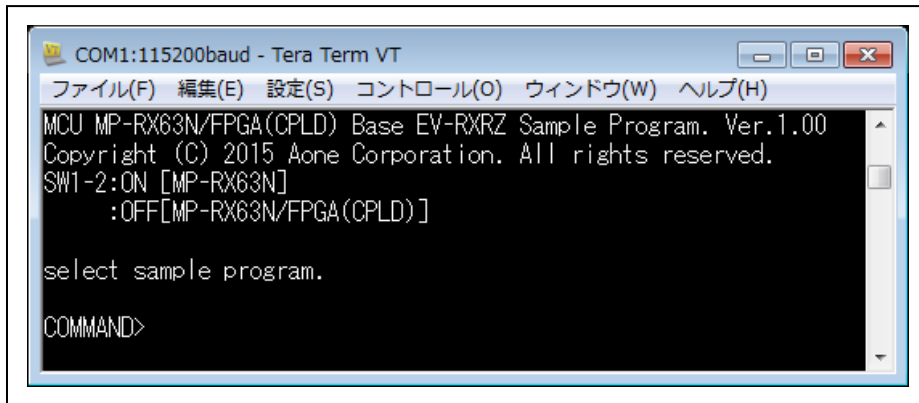
JP1 : 1側 (シングルチップ)
JP2 : 2側 (オンチップエミュレータ)
JP3 : 1側 (PC7 端子 : LOW)

SW1-1-**OFF** ベアメタル版では影響なし
SW1-2-**OFF** MP-RX63N/FPGA|CPLD-xx 用サンプルを指定

⑥EV-RX/RZ-01 の JP 設定

JP1/JP2 **1側**
JP4 **ショート(1/2/3)**

- 5) 「EVRXrz_Sample」プロジェクトのプログラムを MCU 基板にダウンロードして実行させます。



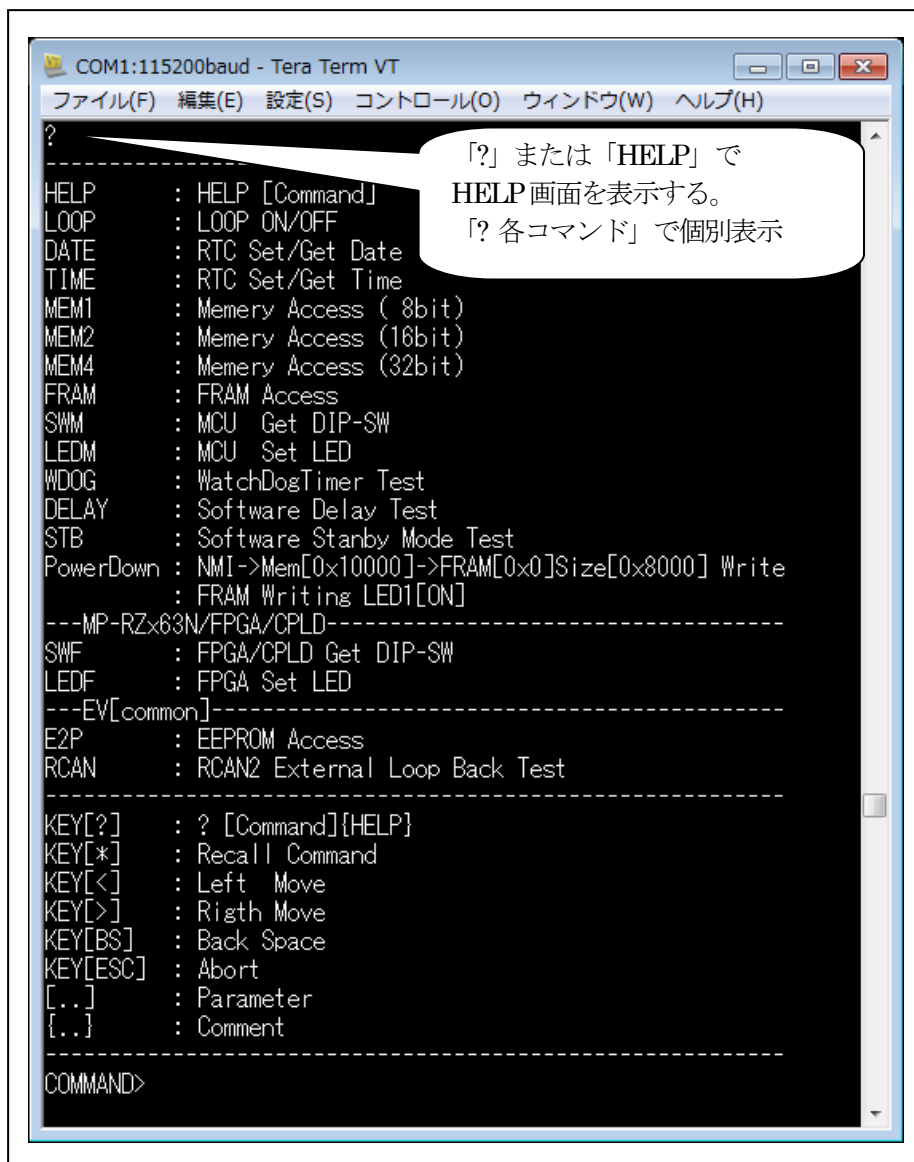
```

COM1:115200baud - 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)
MCU MP-RX63N/FPGA(CPLD) Base EV-RXRZ Sample Program. Ver.1.00
Copyright (C) 2015 Aone Corporation. All rights reserved.
SW1-2:ON [MP-RX63N]
      :OFF[MP-RX63N/FPGA(CPLD)]

select sample program.

COMMAND>
  
```

TeraTerm pro の画面にオープニングメッセージが表示されます。



```

COM1:115200baud - 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)
?
-----
HELP      : HELP [Command]
LOOP      : LOOP ON/OFF
DATE      : RTC Set/Get Date
TIME      : RTC Set/Get Time
MEM1      : Memery Access ( 8bit)
MEM2      : Memery Access (16bit)
MEM4      : Memery Access (32bit)
FRAM      : FRAM Access
SWM       : MCU Get DIP-SW
LEDM      : MCU Set LED
WDOG      : WatchDogTimer Test
DELAY     : Software Delay Test
STB       : Software Standby Mode Test
PowerDown : NMI->Mem[0x10000]->FRAM[0x0]Size[0x8000] Write
          : FRAM Writing LED1[ON]
-----MP-RZx63N/FPGA/CPLD-----
SWF       : FPGA/CPLD Get DIP-SW
LEDF      : FPGA Set LED
-----EV[common]-----
E2P       : EEPROM Access
RCAN      : RCAN2 External Loop Back Test
-----
KEY[?]    : ? [Command][HELP]
KEY[*]    : Recall Command
KEY[<]    : Left Move
KEY[>]    : Righth Move
KEY[BS]   : Back Space
KEY[ESC]  : Abort
[.]       : Parameter
[.]       : Comment
-----
COMMAND>
  
```

「?」または「HELP」で
HELP 画面を表示する。
「? 各コマンド」で個別表示

2-2-1. 各コマンドの説明

LOOP コマンド

各コマンドを繰り返し実行させたい時に使用します。

LOOP 1<tab><tab> //LOOP 指示 1記述はスペース表現とします。以下省略

LOOP 1<tab><tab> //LOOP 解除 1記述はリターン表現とします。以下省略

LOOP 1 にてコマンド処理を繰り返し実行している時に「ESC」キー入力で中断します。

DATE コマンド

MCU内蔵の RTC に年月日曜を設定します。

DATE 1年1月1日1曜日 1 //DATE 2015 4 5 0 2015/4/5 日曜日

//曜日 0:日 1:月 2:火 3:水 4:木 5:金 6:土

DATE 1 //現設定データを表示

TIME コマンド

MCU内蔵の RTC に時間を設定します。

TIME 1時1分1秒 1 //TIME 9 0 0 9時0分0秒

TIME 1 //現設定データを表示

MEM1 コマンド

メモリーを 8bit アクセスで Read/Write/FILL/インクリメント FILL します。

MEM2 コマンド

メモリーを 16nit アクセスで Read/Write/FILL/インクリメント FILL します。

MEM4 コマンド

メモリーを 32bit アクセスで Read/Write/FILL/インクリメント FILL します。

MEM{1/2/4} {R/W/FILL} 先頭アドレス サイズ {パターン} 1

{READ}

MEM1 1R 0x1_0000 0x100 1 //0x1_0000 から 0x100 要素分 8bit ダンプ

MEM2 1R 0x1_0000 0x100 1 //0x1_0000 から 0x100 要素分 16bit ダンプ

MEM4 1R 0x1_0000 0x100 1 //0x1_0000 から 0x100 要素分 32bit ダンプ

{FILL}

MEM1 1F 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0)8bitFILL

MEM2 1F 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0)16bitFILL

MEM4 1F 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0)32bitFILL

{Increment FILL}

MEM1 1I 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0++)8bitFILL

MEM2 1I 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0++)16bitFILL

MEM4 1I 0x1_0000 0x100 0 //0x1_0000 から 0x100 要素分(0++)32bitFILL

{WRITE}

MEM1 1W 0x1_0000 0x12 1 //0x1_0000 に 0x12 を Write

MEM2 1W 0x1_0000 0x1234 1 //0x1_0000 に 0x1234 を Write

MEM4 1W 0x1_0000 012345678 1 //0x1_0000 に 0x12345678 を Write

{Read Only Memory アドレス}

- ・内蔵 RAM エリア {0x0 ~ 0xFFFF}

{Read/Write Memory アドレス}

- ・MCU 内蔵 RAM エリア {0x1_0000 ~ 0x1_FFFF}
- ・FPGA 側 I/O エリア {0x700_0000 ~ 0x700_00FF}
- ・FPGA 内蔵 RAM エリア {0x600_0000 ~ 0x600_3FFF}
- ・MCU 内蔵周辺モジュール {周辺モジュールの仕様による}

FRAM コマンド

FRAM の内容を内蔵メモリーに Read します。また、内蔵 RAM の内容を FRAM に Write します。

{READ}

FRAM_ R_ FRAM アドレス_ Store アドレス_ サイズ

ex)

FRAM_ R_ 0x0_ 0x1_0000_ 0x8000

FRAM アドレス(0x0)からサイズ(0x8000)分 Store アドレス(0x1_0000)に Read します。

{WRITE}

FRAM_ W_ FRAM アドレス_ Memory アドレス_ サイズ

ex)

FRAM_ W_ 0x0_ 0x1_0000_ 0x8000

FRAM アドレス(0x0)に Memory アドレス(0x1_0000)からサイズ(0x8000)分 Write します。

- ・FRAM アドレス {0x0 ~ 0x7FFF}
- ・Store アドレス {0x1_0000 ~ 0x1_FFFF}
- ・Memory アドレス {0x0 ~ 0x1_FFFF}

SWM コマンド

MCU 側が制御している DIP-SW1 の状態を表示します。

SWM<↵

ex)

MCU DIP-SW1_1[ON/OFF] SW1_2[ON/OFF]

LEDM コマンド

MCU 側で制御している LED1/2/3 を点灯・消灯します。

LEDM_ {0/1}_ {0/1}_ {0/1}<↵ // LEDM {LED1} {LED2} {LED3} 0:消灯 1:点灯

WDOG コマンド

WDOG タイマーを起動させ MCU リセットさせます。
MCU リセット後は、電源を再立ち上げして下さい。

DELAY コマンド

MCU 内部で利用している 1usec タイマーの精度を計るため LED1 を点滅させます。

```
DELAY_ {Time 値}usec<  // DELAY 10<  10usec の精度
```

- ①LED1{time 値} 点灯
- ②LED1{time 値} 消灯
- ③LED1{time 値} 点灯
- ④LED1{10msec} 消灯

STB コマンド

ソフトウェア・スタンバイ・モードに移行させます。
STB 後は、電源を再立ち上げして下さい。

Power Down(NMI 処理)

停電検出回路が有効になっている場合、電源 OFF 時に内蔵 RAM の内容を 32Kbyte 分 FRAM に Write します。

LED1 点灯
FRAM(0x0)から内蔵 RAM(0x1_0000)の内容を 32Kbyte 分 Write する。
LED1 消灯

LED1 の点灯時間を計測することにより書き込み時間を得ることができます。

SWF コマンド

FPGA 側が制御している DIP-SW2 の状態を表示します。

```
SWF<
```

ex)

```
FPGA/CPLD DIP-SW2_1[ON/OFF] SW2_2[ON/OFF]
```

LEDF コマンド

FPGA 側で制御している LED5/6 を点灯・消灯します。

```
LEDF_ {0/1}_ {0/1}<  // LEDF {LED5} {LED6} 0:消灯 1:点灯
```

RCAN コマンド

RCAN-2 の外部ループバックテスト機能を実行します。

```
RCAN<
```

ex)

```
<TX> 00000000 00 01 02 03 04 05 06 07 // 00->07 数字を送信 data++
```

```
<RX> 00000000 00 01 02 03 04 05 06 07 // 00->07 数字を受信
```

E2P コマンド

EEPROM の Read/Write 処理をします。

E2P_{R/W}_EEPROM アドレス_{メモリアドレス}_サイズ

{READ}

E2P_R_ EEPROM アドレス_サイズ

ex)

E2P_0x0_0x100 // EEPROM の 0x0 番地から 0x100 サイズ分ダンプ表示

{WRITE}

E2P_W_ EEPROM アドレス_メモリアドレス_サイズ

ex)

E2P_W_0x0_0x1_0000_0x80 // EEPROM の 0x0 番地に 0x1_0000 番地の内
// 内容を 0x80 サイズ分 Write

この EEPROM は、MAC アドレス内蔵の EEPROM です。

EEPROM の(0x80~0xFF)は、ライトプロテクトになっていますので Write できません。

MAC アドレスは、【0xFA~0xFF】の 8 バイトに格納してあります。

{Read Only Memory アドレス}

- ・ EEPROM エリア {0x80 ~ 0xFF}
- ・ 内蔵 RAM エリア {0x0 ~ 0xFFFF}

{Write Memory アドレス}

- ・ EEPROM エリア {0x0 ~ 0x7F}
- ・ 内蔵 RAM エリア {0x1_0000 ~ 0x1_FFFF}

KEY 操作

簡単な 1 ラインエディタ機能を入れてあります。

- ・ BS バックスペース
- ・ ← 左にカーソル移動
- ・ → 右にカーソル移動
- ・ ↑ 1 回前に入力した内容のリコール
- ・ ESC コマンド処理中の中断

2-3. 「EVRXrz_Sample_USB」プロジェクトの説明

1) 動作説明

- EVRXrz_Sampleに USB-Function 機能を追加したプロジェクトになります。
- 各コマンド体系は、2-2項を参照して下さい。
- USB-Function ライブラリーは、別途有償にて提供しております。ご購入前の評価用として実行用ファイルは添付しております。

2) フォルダ構成とファイル名(評価用)

Sample_RX			
	EV-RXRZ	EVRXrz_Sample.USB.mot	実行用 Hex ファイル
	_PC_Test	ITF_USB_TEST.EXE	PC用テストプログラム
		DRIVER\ITFUSBLib	PC側 USB ドライバー

3) フォルダ構成とファイル名(有償) ご購入 ITFUSBLib_RX63N_xx に添付

Sample_RX\EV-RXRZ\Hew4\EVRXrz_Sample_USB			【Hew4】
	Debug	ビルドにより生成された実行ファイル等が格納される場所	
	ITF_LIB	空	ITF_LIB オリジナル CD からのインポート 手順書
		ReadMe.txt	
	lnk_app	空	EVRXrz_Sample\src_app にリンク
	lnk_eva	空	EVRXrz_Sample\src_eva にリンク
	lnk_sys	空	EVRXrz_Sample\src_eva にリンク

無償評価用「EVRXrz_Sample_USB」に黄色部分(有償)をインポートします。

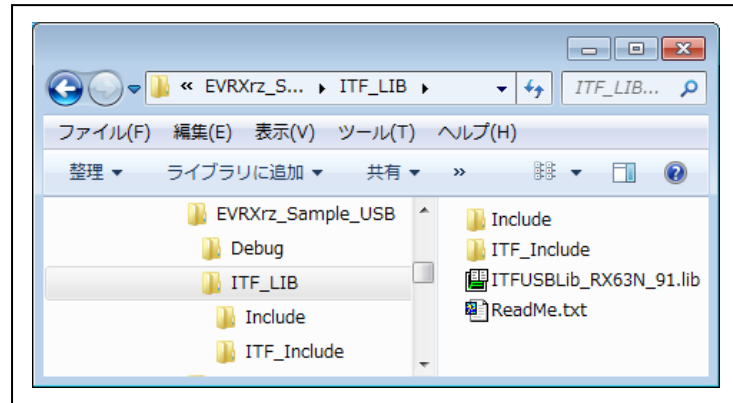
Sample_RX\EV-RXRZ\Cube\EVRXrz_Sample_USB			【CS+】
	DefaultBuild	ビルドにより生成された実行ファイル等が格納される場所	
	ITF_LIB	空	Hew4\EVRXrz_Sample_USB\ITF_LIB にリンク
	lnk_app	空	Hew4\EVRXrz_Sample\src_app にリンク
	lnk_eva	空	Hew4\EVRXrz_Sample\src_eva にリンク
	lnk_sys	空	Hew4\EVRXrz_Sample\src_sys にリンク

4) ITF_LIB オリジナル CD (有償) からサンプル CD にインポートする手順

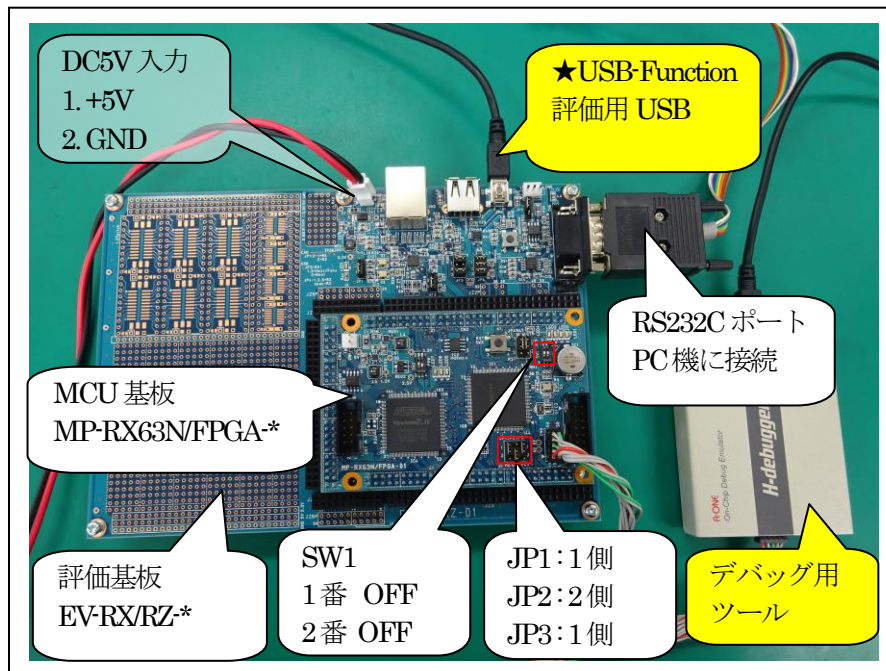
- a. 3) 表の黄色部は、空ディレクトリになっていますので、ITF_LIB オリジナル CD から必要なファイルを Copy します。

ITF_LIB オリジナル CD		サンプル(EVRXrz_Sample_USB)
ITF_LIB\Include	→	ITF_LIB\Include
ITF_LIB\ITF_Include	→	ITF_LIB\ITF_Include
ITF_LIB\ITFUSBLIB_xx.lib	→	ITF_LIB\ITFUSBLIB_xx.lib

上記のように ITF_LIB オリジナル CD から、サンプル「EVRXrz_Sample_USB\ITF_LIB」の空ディレクトリに Copy して下さい。

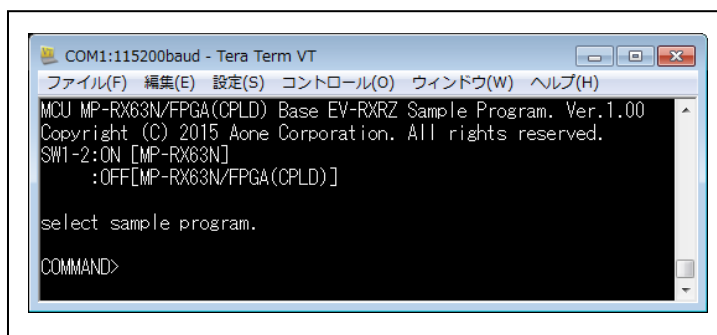


5) 動作構成 (電源 OFF)



6) 動作手順

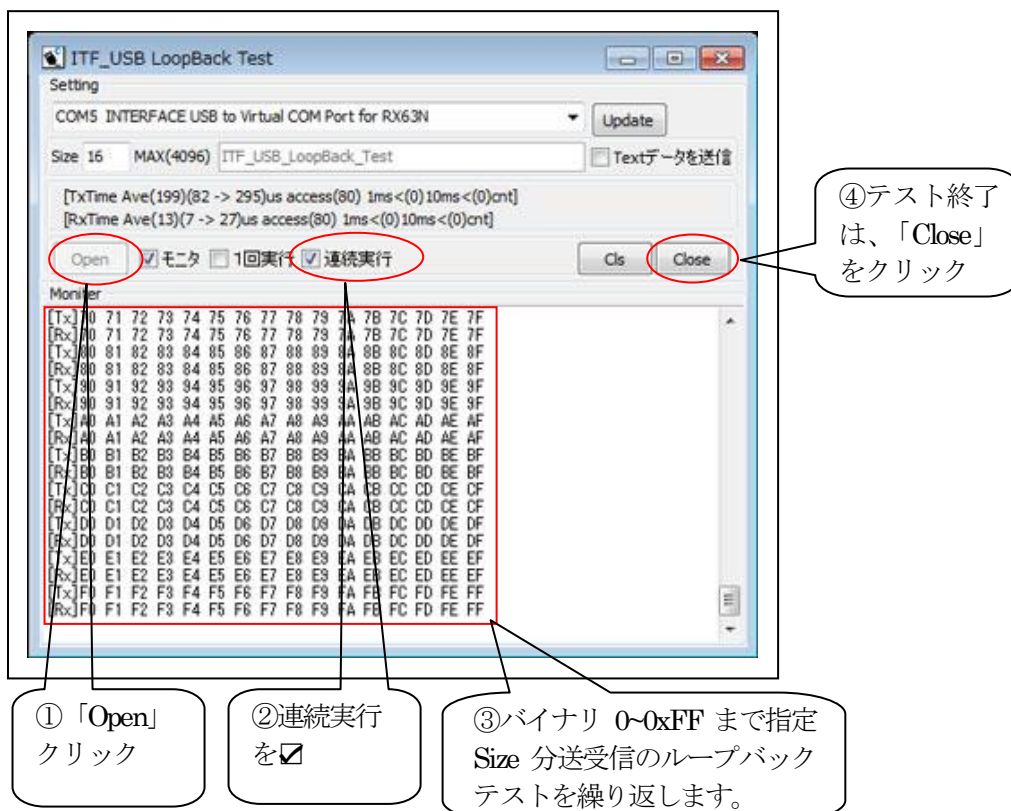
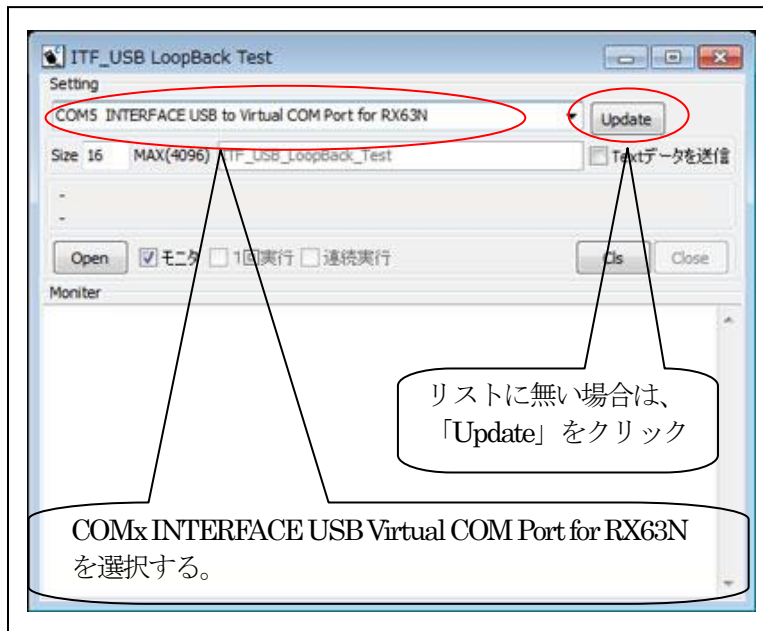
- a. ターゲット基板側の電源 OFF の状態で上図の★USB-Function 評価用 USB ケーブル以外を接続する。
- b. ターゲット基板側の電源を ON にする。
- c. デバッガ「DEF」を立ち上げる。
- d. デバッガ「DEF」画面の左下隅の「Start」をクリックする。
- e. デバッガ「DEF」の【オプション】－【フラッシュ ROM ライター】を起動する。
- f. 無償評価用 Hex ファイル「EVRXrz_Sample_USB.mot」を内蔵 ROM へ書き込みをする。
- g. ターゲット側の電源を OFF にする。
- h. ジャンパー(JP2)を 1 側にする。
- i. デバッガツールを外す。
- j. ★USB-Function 評価用 USB ケーブルを PC 機に接続する。
- k. RS232C ケーブルが PC 機に接続されているのを確認後、「TeraTerm pro」を起動する。
- l. ターゲット基板側の電源を ON にする。



TeraTerm pro の起動画面

7) USB ファンクションの動作確認

- a. Windows が、USB ドライバのインストールを要求しますので USB-Driver をインストールする。
「Sample_RX¥_PC_Test¥_DRIVER¥ITFUSBLib」にドライバーがあります。
- b. TeraTerm pro 画面でコマンド「**USBF**」を入力する。
- c. Windows 側のテストプログラム「ITF_USB_TEST.EXE」を起動する。



以上です。

3. 注意事項

- ・本文書の著作権は、エーワン（株）が保有します。
- ・本文書を無断での転載は一切禁止します。
- ・本文書に記載されている内容についての質問やサポートはお受けすることが出来ません。
- ・本サンプルプログラムに関して、インターフェイス社、ルネサス エレクトロニクス社への問い合わせは御遠慮願います。
- ・本文書の内容に従い、サンプルソフトを使用した結果、不具合が発生しても、弊社では一切の責任を負わないものとします。
- ・本文書の内容に関して、万全を期して作成しましたが、ご不審な点、誤りなどの点がありましたら弊社までご連絡くだされば幸いです。
- ・本文書の内容は、予告なしに変更されることがあります。

4. 商標

- ・RX および RX63N は、ルネサス エレクトロニクス株式会社の登録商標、または商品名です。
- ・その他の会社名、製品名は、各社の登録商標または商標です。

5. 参考文献

- ・「RX63N グループ ユーザーズマニュアル ハードウェア編」
ルネサス エレクトロニクス株式会社
- ・ルネサス エレクトロニクス株式会社提供のサンプル集
- ・その他

〒486-0852
愛知県春日井市下市場町 6-9-20
エーワン株式会社
<http://www.robin-w.com>