Rev 1.10.00

Renesas RZ/A1H 用サンプル(e2studio-GNU RTOS-NORTi 版)の説明

(EV-RX/RZ-xx＋MP-RZA1H-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願います。

・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，いかなる保証も行わない。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（RTOS 版）のプロジェクト名

サンプルプロジェクト名		
EVrxRZ_NORTi	MCU 基板(MP-RZA1H-*) 評価用基板(EV-RX/RZ-*)用サンプル	ソース公開 (有償)
EVrxRZ_NORTi_USB	MCU 基板(MP-RZA1H-*) 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	ツールチェイン
e2 studio(バージョン 5.1.0.022)	GNUARM-NONEv16.01-EABI

C ソースに#ifdef 等のマクロ定義している場合に使用します。	
注＊1 __USED_DEFnano__=x	x = DEFnano を使用[1]する・[0]しない。
RTOS	NORTi 使用時に定義
CH2	NORTi 使用時に定義
ITF_LIB	USB-Function 使用時に定義
GNU	GNU 使用時に定義

ASM ソースに IF 等のマクロ定義している場合に使用します。	
注＊1 --pd="__USED_DEFnano__ EQU x"	x = DEFnano を使用[1]する・[0]しない。

サンプルプロジェクト別に必要なマクロ定義例				
EVrxRZ_NORTi	RTOS	CH2	GNU	
EVrxRZ_NORTi_USB	RTOS	CH2	GNU	ITF_LIB

注＊1

「__USED_DEFnano__=0」と使用しない側に定義しても内蔵 RAM へのダウンロードとシリアルフラッシュ ROM への書き込み操作は可能です。ただし、再操作する場合はターゲット側のリセット操作が必要になります。

２－１．「EVrxRZ_NORTi」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）サンプル CD のフォルダ構成とファイル名

Sample_e2studio¥EV-RXRZ¥EVrxRZ_NORTi				
	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		
	NORTi	空		
		ReadMe.txt	インストール済み NORTi オリジナルからのインポート手順書	
	NORTi_smp	NETSMP	inc	NETSMP のインクルード
			aonettcp.c	TCP/IP ループバックテスト
			noncons.c	コンソール関係
			nondhcp.c	DHCP 処理
			nonedns.c	DNS サーバー
			nonping.c	PING 処理
			nonshel.c	コマンド shell
			nonsntp.c	SNTP 処理
			nontelc.c	TELNET サンプル
			nonteld.c	TELNET デーモン
			sntptime.c	SNTP 処理
			ReadMe.txt	オリジナル NETSMP の変更履歴
		SMP	inc	SMP のインクルード
			ethza1.c	LAN ドライバーの初期化
			n4iza1h.c	NORTi 使用の MCU 依存部
			nonecfg.c	TCP/IP サンプルコンフィグレーション
			nonethw.c	ETHER 割り込み登録
			nosrza1h.c	SIO 用シリアルドライバー
			phyza1.c	PHY コントロール
			mmuA9.s	MMU 処理
			ReadMe.txt	オリジナル SMP の変更履歴
	linker_script_file	Locate.ld	ロケート定義用スクリプトファイル	
	src_app	inc	src_app のインクルード用ディレクトリ	
		main_r.c	メイン処理	
		cevrza1h.c	RZ/A1H の初期化処理	
		board.c	LED・SW 等の処理ソフト	
		bsc.c	BSC 初期化処理	
		ostm_r.c	OS 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	
		spibsc.c	SPIBSC の初期化と read/write 処理	
		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_comm.c	RIIC の初期化と read/write 処理	

		rscan.c	RSCAN の初期化と read/write 処理
		sci_comm.c	SCI の初期化と read/write 処理
		usb_func.c	ITF_USBLib の使用サンプル
	src_gsys	inc	src_sys のインクルード用ディレクトリ
		_vector_table.s	リセットベクターテーブル
		_init_handler.s	割り込みハンドラー処理
		_rst_handler.s	リセット時の ARM 初期化処理
		profile.c	プロファイル処理
		Nmonitor.c	デバッグモニター処理

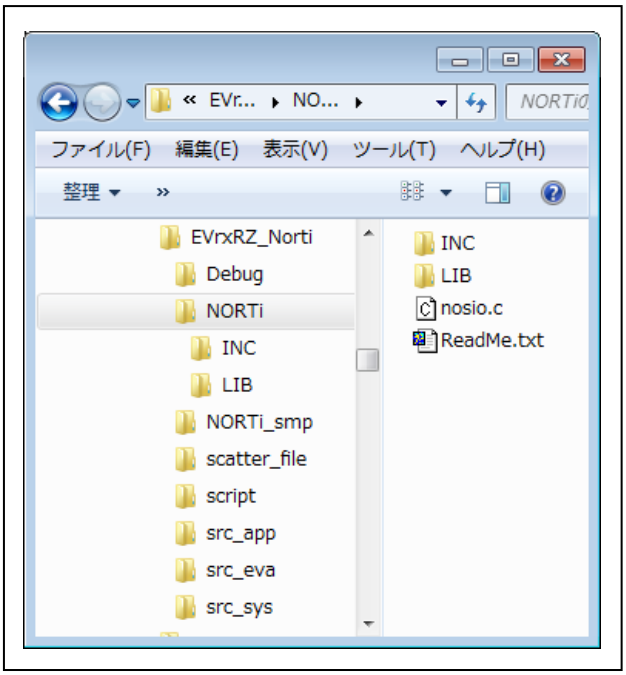
3）インストール済み NORTi オリジナル（有償）からサンプルにインポートする手順

a．サンプル「EVrxRZ_NORTi」を PC 機の適当なフォルダに全 Copy します。または、「Sample_e2studio.zip」の圧縮ファイルを適当なフォルダで解凍します。

b．2）表の黄色部は、空ディレクトリになっていますので、インストール済み NORTi オリジナルから、下記のように Copy します。

NORTi オリジナル		サンプル(EVrxRZ_NORTi)
NORTi¥INC	→	NORTi¥INC
NORTi¥LIB	→	NORTi¥LIB
NORTi¥SRC¥nosio.c	→	NORTi¥nosio.c

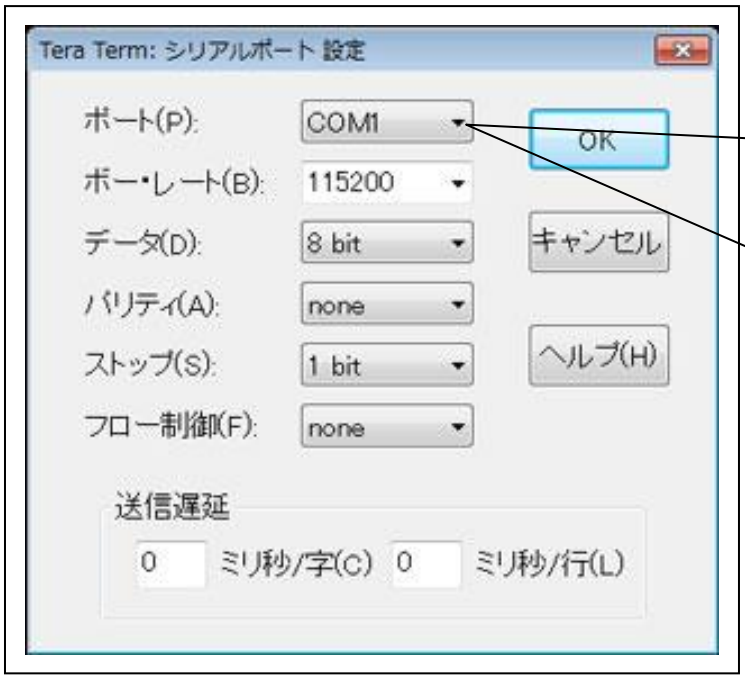
上記のように NORTi オリジナルから、サンプル「EVrxRZ_NORTi¥NORTi」の空ディレクトリに Copy して下さい。



c．ReadMe.txt に Copy 後の変更内容が記述してありますのでソースを変更して下さい。

d．アプリケーションノート「DS5_Inport.pdf」を参考にして e2studio にインポートして下さい。

4）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。
②PC にインストールし実行する
③シリアルポートの設定

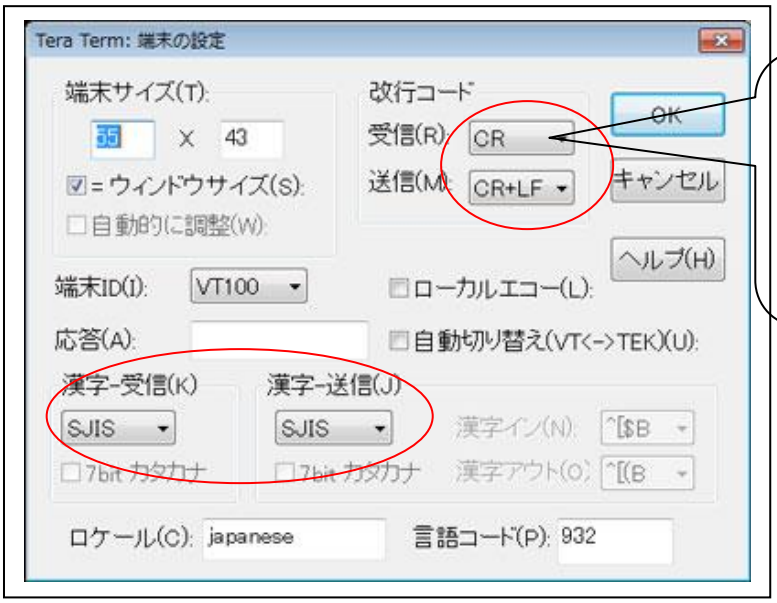


COM 番号は、PC 側でシリアル通信可能な番号を指定する。

115200BPS
8bit
none
1bit
none

の仕様にする。

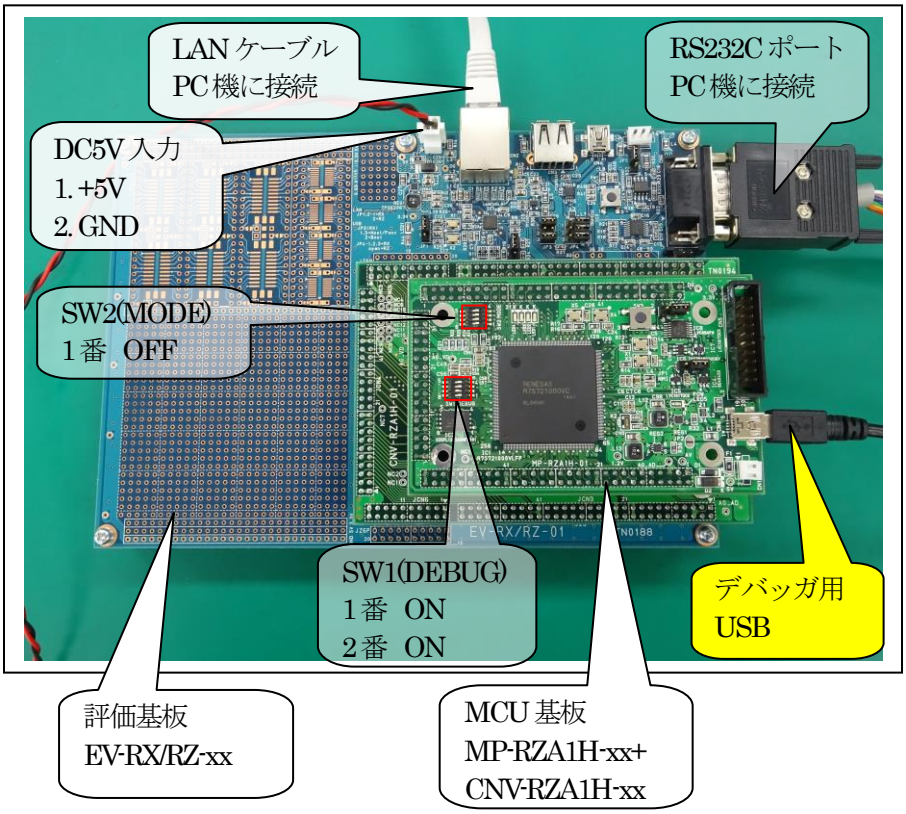
④端末の設定



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。

5）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。

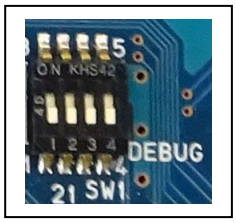
EV-RX/RZ-xx

DSUB 9pin オス	
1	NC
2	RXD
3	TXD
4	NC
5	GND
6	NC
7	RTS
8	CTS
9	NC

PC-Windows

DSUB 9pin オス	
1	NC
2	RXD
3	TXD
4	DTR
5	GND
6	DSR
7	RTS
8	CTS
9	NC

④MCU 基板上の SW1 設定

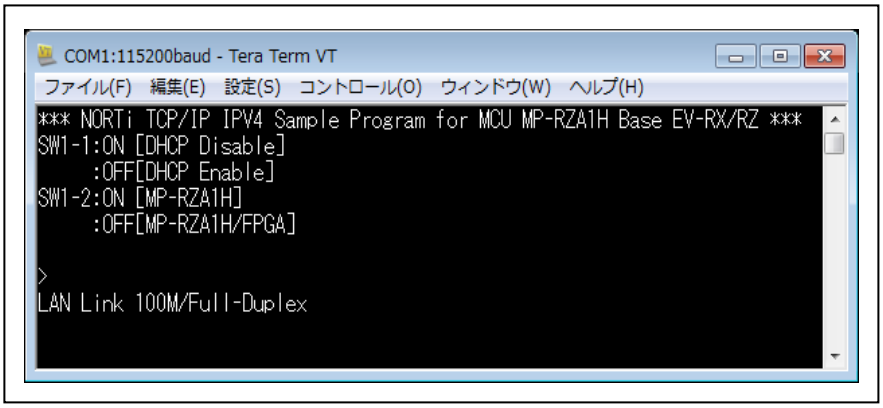


SW1-1　ON　起動時に DHCP サーバーへのアクセスを禁止する。
SW1-2　ON　MP-RZA1H-xx 用サンプルを指定

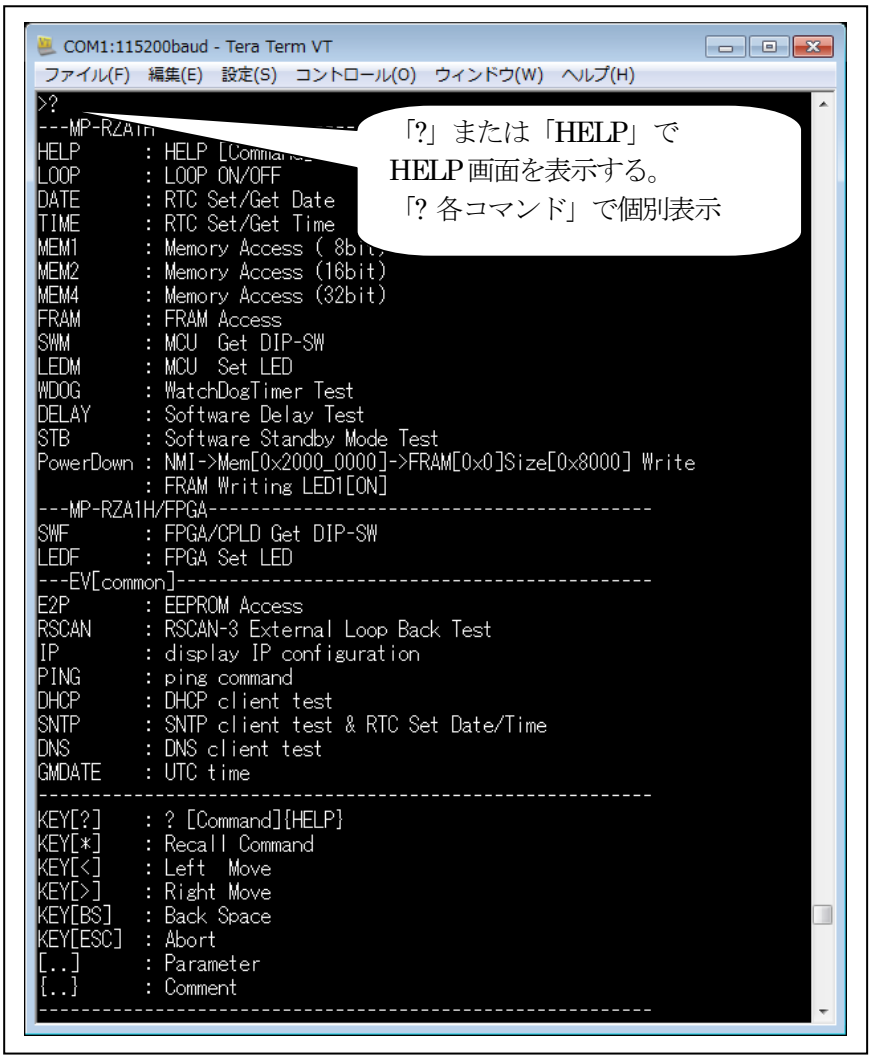
⑤EV-RX/RZ-01 の JP 設定

JP1/JP2　　2 側
JP4　　open(1/2/3)

6）「EVrxRZ_NORTi」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。



TeraTerm pro の画面にオープニングメッセージが表示されます。



２－１－１．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⏎　//LOOP 指示　　␣記述はスペース表現とします。以下省略
LOOP ␣0⏎　//LOOP 解除　　⏎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 ␣年␣月␣日␣曜日⏎　// 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⏎　　//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␣時␣分␣秒⏎　　// TIME␣9␣0␣0　　9 時 0 分 0 秒
TIME⏎　　//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/F/I/W}␣先頭アドレス␣サイズ␣{パターン}⏎

{READ}
MEM1␣R␣0x2000_0000␣0x100⏎　// 0x2000_0000 から 0x100 要素分 8bit ダンプ
MEM2␣R␣0x2000_0000␣0x100⏎　// 0x2000_0000 から 0x100 要素分 16bit ダンプ
MEM4␣R␣0x2000_0000␣0x100⏎　// 0x2000_0000 から 0x100 要素分 32bit ダンプ
{FILL}
MEM1␣F␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0)8bitFILL
MEM2␣F␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0)16bitFILL
MEM4␣F␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0)32bitFILL
{Increment FILL}
MEM1␣I␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0++)8bitFILL
MEM2␣I␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0++)16bitFILL
MEM4␣I␣0x2000_0000␣0x100␣0⏎　// 0x2000_0000 から 0x100 要素分(0++)32bitFILL
{WRITE}
MEM1␣W␣0x2000_0000␣0x12⏎　　// 0x2000_0000 に 0x12 を Write
MEM2␣W␣0x2000_0000␣0x1234⏎　　// 0x2000_0000 に 0x1234 を Write
MEM4␣W␣0x2000_0000␣012345678⏎ // 0x2000_0000 に 0x12345678 を Write

{Read Only Memory アドレス}
・シリアルフラッシュ ROM エリア　{0x1800_0000 ～0x18FF_FFFF}
・内蔵 RAM エリア　{0x2002_0000 ～0x209F_FFFF}

{Read/Write Memory アドレス}
・MCU 内蔵 RAM エリア　{0x2000_0000 ～0x2001_FFFF}
・FPGA 側 I/O エリア　{0x4800_0000 ～0x4800_7FFF}
・FPGA 内蔵 RAM エリア　{0x4800_8000 ～0x4800_BFFF}
・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␣0x2000_0000␣0x8000
FRAM アドレス(0x0)からサイズ(0x8000)分 Store アドレス(0x2000_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␣0x2000_0000␣0x8000
FRAM アドレス(0x0)に Memory アドレス(0x2000_0000)からサイズ(0x8000)分 Write します。

・FRAM アドレス　{0x0 ～0x7FFF}
・Store アドレス　{0x2000_0000 ～0x2001_FFFF}
・Memory アドレス　{0x2000_0000 ～0x209F_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] SW1_3[ON/OFF] SW1_4[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(0x2000_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。

RSCAN　コマンド
RSCAN-3 の外部ループバックテスト機能を実行します。

RSCAN⏎
ex)
<TX> cnt[0] id[1] dlc[8] 00 01 02 03 04 05 06 07 // 00->07 数字を送信　data++
<RX> cnt[0] id[1] dlc[8] 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␣0x2000_0000␣0x80　　// EEPROM の 0x0 番地に 0x2000_0000 番地の内
　　// 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 6 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　　{0x2002_0000 ～0x209F_FFFF}

{Write Memory アドレス}
・EEPROM エリア　　{0x0 ～0x7F}
・内蔵 RAM エリア　　{0x2000_0000 ～0x2001_FFFF}

E2P コマンドには LAN 用データ設定コマンドが用意されています。

①E2P␣MAC　　{Read Only}
　MAC の表示

②E2P␣NAME　　{Read/Write}
　ネットワークインターフェイス名の表示と設定

　{Read}
　E2P␣NAME⏎

　{Write}
　E2P␣NAME␣{名前}⏎　　//　MAX 7 ASCII 文字 E2P␣NAME␣RZsamp1⏎

③E2P␣PORT　　{Read/Write}
PORT 番号の表示と設定

{Read}
E2P␣PORT⏎

{Write}
E2P␣PORT␣{xxxxx}⏎　　// E2P ␣PORT␣50000⏎

④E2P␣IP　　{Read/Write}
デフォルト IP アドレスの表示と設定

{Read}
E2P␣IP⏎

{Write}
E2P␣IP␣{xx.xx.xx.xx}⏎　　// E2P ␣IP␣192.168.21.12⏎

⑤E2P␣SUB　　{Read/Write}
サブネットマスクの表示と設定

{Read}
E2P␣SUB⏎

{Write}
E2P␣SUB␣{xx.xx.xx.xx}⏎ // E2P ␣SUB␣255.255.255.0⏎

⑥E2P␣GATE　　{Read/Write}
ゲートウェイアドレスの表示と設定

{Read}
E2P␣GATE⏎

{Write}
E2P␣GATE␣{xx.xx.xx.xx}⏎　　// E2P ␣GATE␣192.168.21.126⏎

⑦E2P␣DHCP　　{Read/Write}
DHCP アドレスの表示と設定

{Read}
E2P␣DHCP⏎

{Write}
E2P␣DHCP␣{xx.xx.xx.xx}⏎　　// E2P ␣GATE␣192.168.21.63⏎

⑧E2P␣DNS　　{Read/Write}
DNS アドレスの表示と設定

{Read}
E2P␣DNS⏎

{Write}
E2P␣DNS␣{xx.xx.xx.xx}⏎ // E2P ␣GATE␣192.168.21.126⏎

注記
使用するルータに設定されている IP アドレスを参照して正しいデータを設定して下さい。

IP　コマンド
DHCP 等により内部登録された、ネットワークインターフェイス名、IP アドレス、ポート番号を表示する。

IP⏎

ex)
RZsamp1　:　192.168.21.21　:　50000

PING　コマンド
PING コマンドを発行する。

PING␣xxx.xxx.xxx.xxx⏎

ex)
PING␣192.168.21.22⏎
32 bytes from 192.168.21.22: icmp_seq = 1, time = 2 ms
32 bytes from 192.168.21.22: icmp_seq = 2, time = 1 ms
32 bytes from 192.168.21.22: icmp_seq = 3, time = 1 ms

DHCP　コマンド
DHCP 処理を実行

DHCP⏎

ex)
Successfully assigned by DHCP server (RZsamp1)
[Ethernet Address] : [00-1E-C0-F0-E8-7F]
[My IP Address] : [192.168.21.21]
[Default IP Address] : [192.168.21.12]
[Default Gateway] : [192.168.21.126]
[Subnet Mask] : [255.255.255.0]
[DHCP] : [192.168.21.63]
[DNS] : [192.168.21.126]
[PORT Number] : [50000]

SNTP　コマンド

SNTP サーバーにアクセスして標準時間を取得し、RTC に年月日曜日と時分秒を設定する。

SNTP⏎

ex)
Tue Mar 31 17:49:29 2015
DATE 2015/03/31[Tue]
TIME 17:49:29

DNS　コマンド

DNS サーバーにアクセスして IP アドレスを取得する。

DNS␣{ドメイン名}⏎

ex)
DNS␣www.yahoo.co.jp⏎
www.yahoo.co.jp has address 182.22.59.229

GMDATE　コマンド

SNTP コマンドにより取得したグリニッジ標準時間を表示する。

GMDATE⏎

ex)
Tue Mar 31 08:53:18 2015

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

２－２．「EVrxRZ_NORTi_USB」プロジェクトの説明

１）動作説明

・EVrxRZ_NORTi 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_e2studio¥_			
	EV-RXRZ¥_EVrxRZ_USB	EVrxRZ_NORTi_USB.mot	実行用 Hex ファイル
	_PC_Test	ITF_USB_TEST..EXE	PC 用 USB テストプログラム
		DRIVER¥ITFUSBLib	PC 側 USB ドライバー
		TCP_IP_TEST.EXE	PC 用 TCP/IP テストプログラム

３）フォルダ構成とファイル名(有償用)　ご購入 ITFUSBLib_RZA1H_xx を添付

Sample_e2studio¥EV-RXRZ¥EVrxRZ_NORTi_USB			
	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	
		ReadMe.txt	ITF_LIB オリジナル CD からのインポート手順書
	lnk_NORTi	空	EVrxRZ_NORTi¥NORTi にリンク
	lnk_NORTi_smp	空	EVrxRZ_NORTi¥NORTi_smp にリンク
	linker_script_file	Locate.ld	ロケート定義用スクリプトファイル
	lnk_app	空	EVrxRZ_NORTi¥src_app にリンク
	lnk_eva	空	EVrxRZ_NORTi¥src_eva にリンク
	lnk_gsys	空	EVrxRZ_NORTi¥src_gsys にリンク

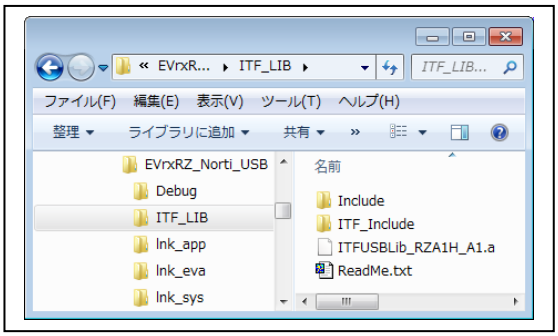
「EVrxRZ_NORTi_USB」に黄色部分(有償)をインポートします。

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

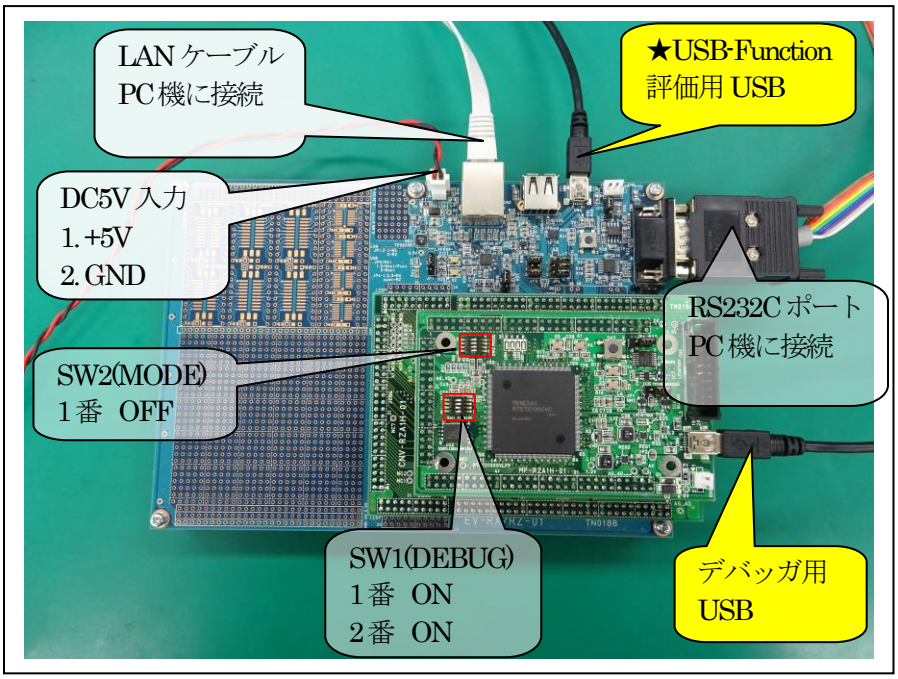
a． 3）表の黄色部は、空ディレクトリになっていますので、ITF_LIB オリジナル CD から必要なファイルを Copy します。

ITF_LIB オリジナル CD		サンプル(EVrxRZ_NORTi_USB)
ITF_LIBi¥Include	→	ITF_LIB¥Include
ITF_LIB¥ITF_Include	→	ITF_LIB¥ITF_Include
ITF_LIB¥ITFUSBLIB_xx.a	→	ITF_LIB¥ITFUSBLIB_xx.a

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

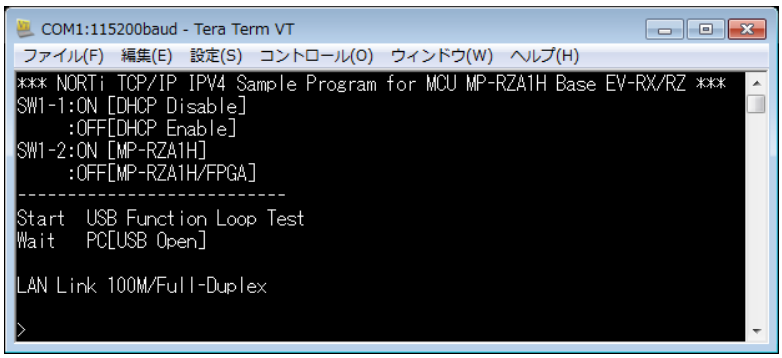


5）動作構成（電源 OFF）



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「DEFnano」を立ち上げる。
d．デバッガ「DEFnano」画面の左下隅の「Start」をクリックする。
e．デバッガ「DEFnano」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_NORTi_USB.mot」をシリアルフラッシュ 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．デバッガ用 USB ケーブルを抜き取る。
i．★USB-Function 評価用 USB ケーブルを PC 機に接続する。
j．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。
k．ターゲット基板側の電源を ON にする。

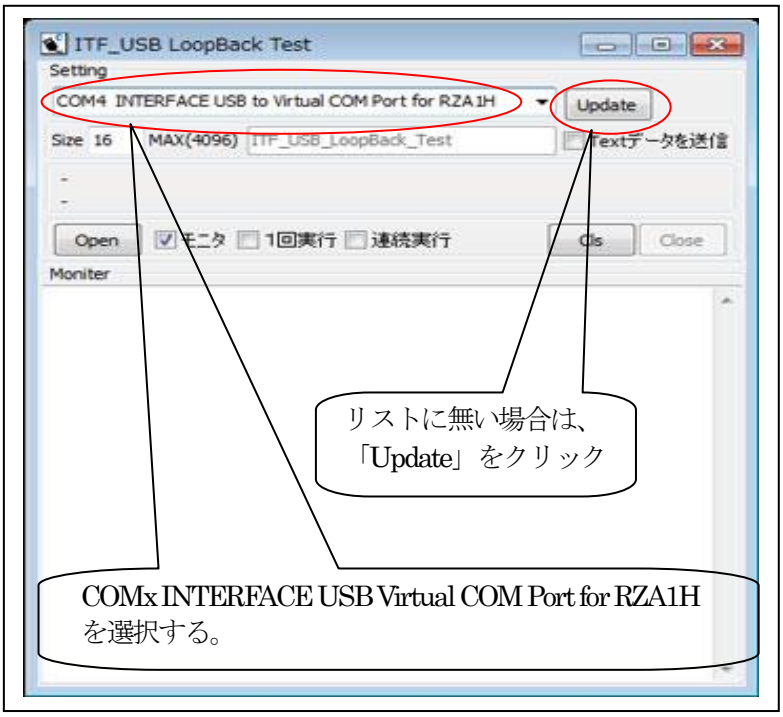


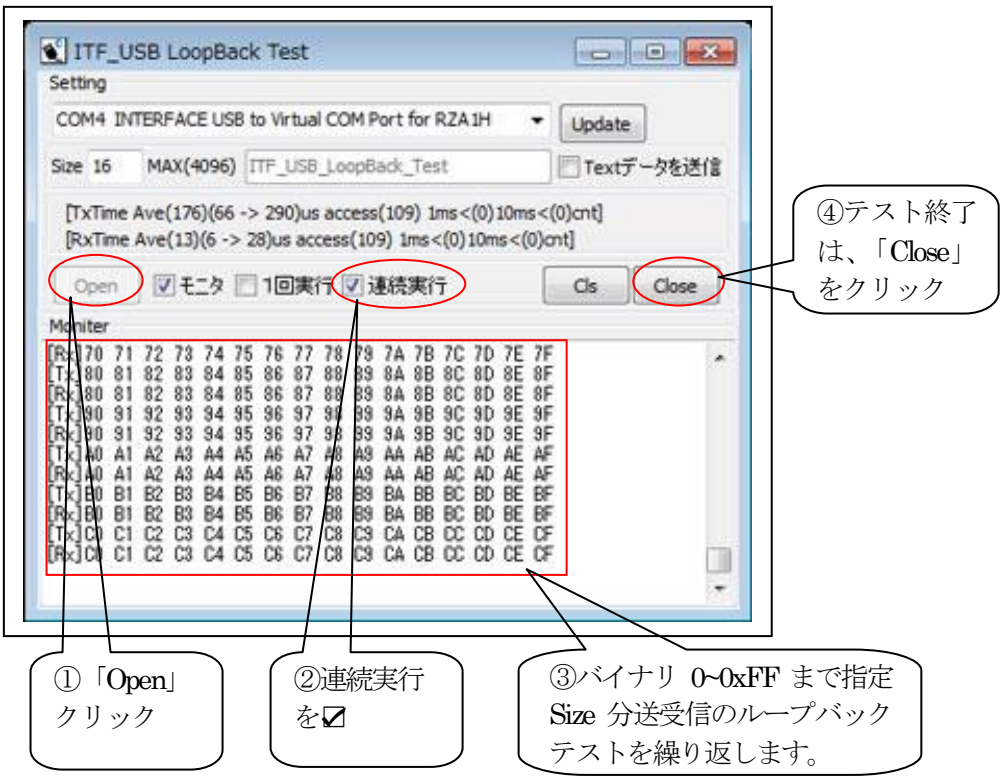
TeraTerm pro の起動画面

7）USB ファンクションの動作確認

a．Windows が、USB ドラーバー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_e2studio¥_PC_Test¥ DRIVER¥ITFUSBLib」
にドライバーがあります。

b．Windows 側のテストプログラム「ITF_USB_TEST. EXE」を起動する。





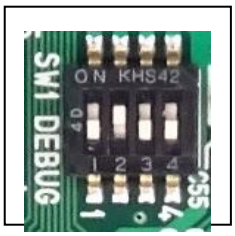
8）TCP/IP プロトコル通信の動作確認

a．TCP/IP 通信に必要な情報を EEPROM 登録する。（ルータの設定データ）
２－１「EVrxRZ_NORTi」プロジェクトの説明
６）各コマンドの説明・E2P　コマンド を参照

ex)
Successfully assigned by DHCP server (RZsamp1)
[Ethernet Address] : [00-1E-C0-F0-E8-7F]
[My IP Address] : [192.168.21.21]
[Default IP Address] : [192.168.21.12]
[Default Gateway] : [192.168.21.126]
[Subnet Mask] : [255.255.255.0]
[DHCP] : [192.168.21.63]
[DNS] : [192.168.21.126]
[PORT Number] : [50000]

上記の全情報を登録しないと TCP/IP 通信は出来ません。

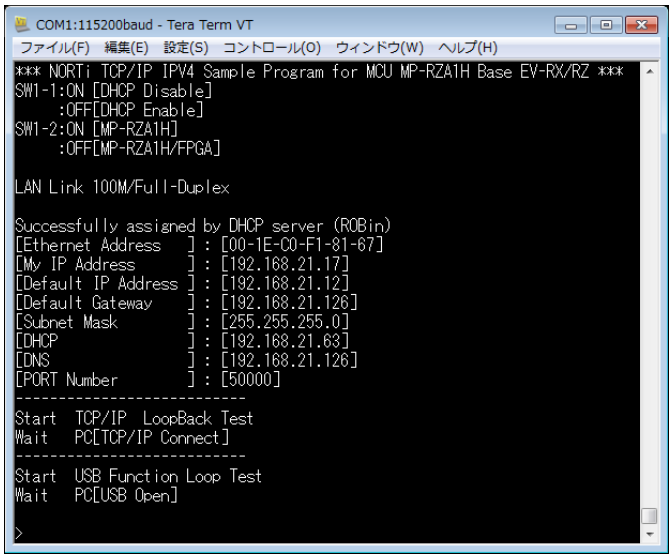
b．MCU 基板上の SW1 設定



SW1-1　OFF
[ON]- 起動時に DHCP サーバーへのアクセスを禁止する。
[OFF] 起動時に DHCP サーバーへのアクセスを許可する。
SW1-2　ON　MP-RZA1H-xx 用サンプルを指定

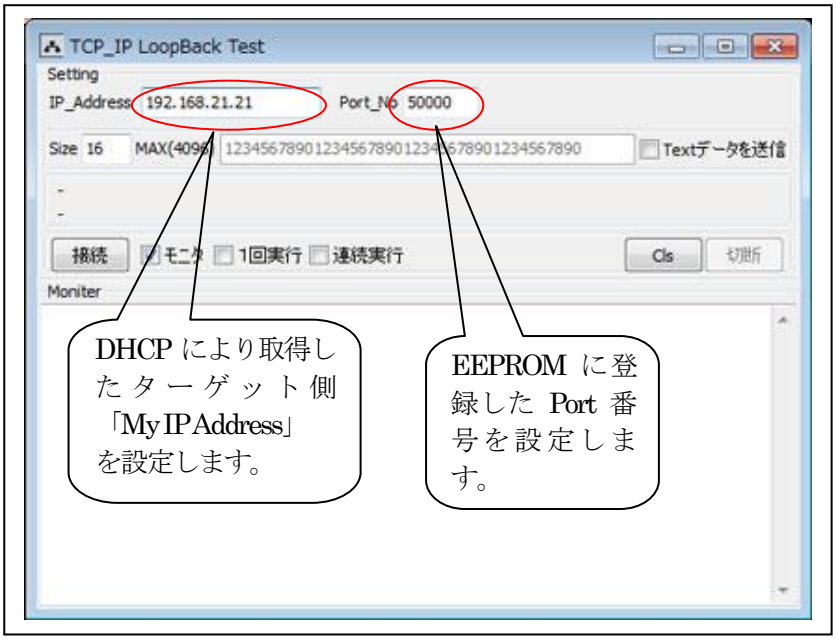
c．ターゲット基板側の電源を入れ直します。
・EEPROM に情報が全て登録されていない場合は、
TeraTerm 画面に「There is no LAN data in EEPROM」と表示します。

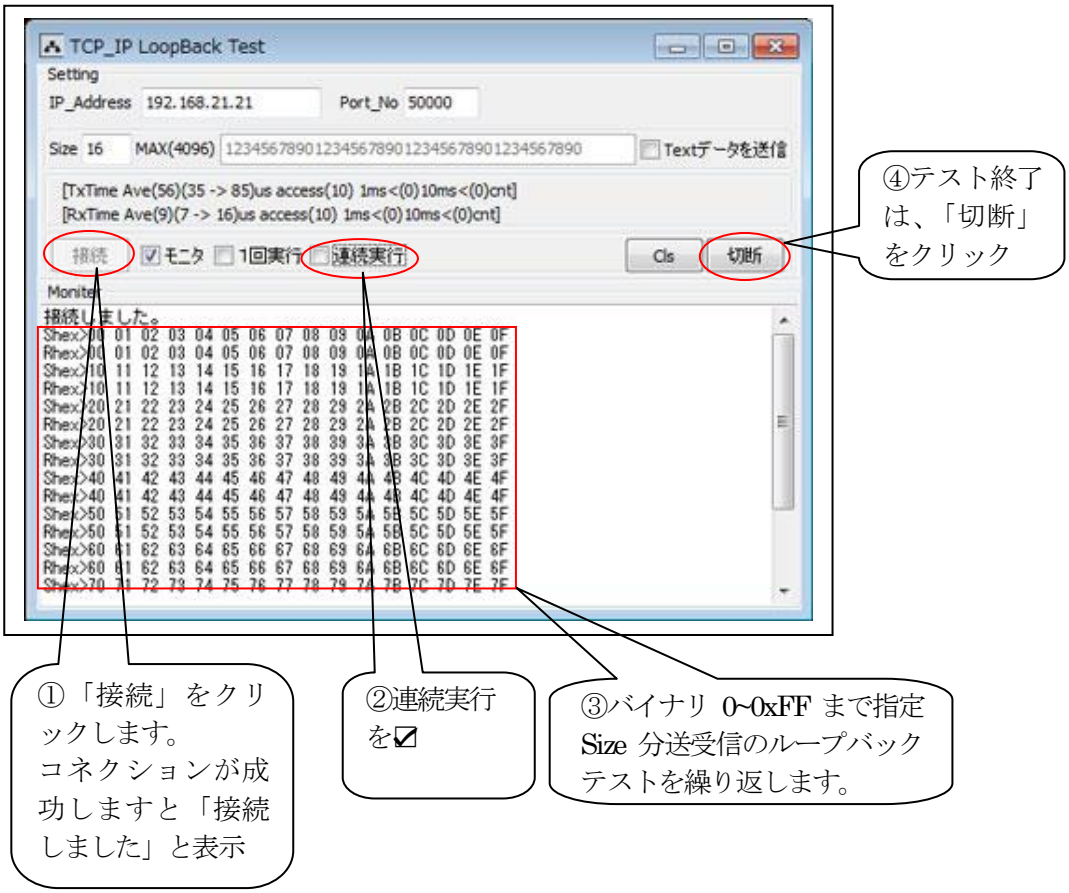
・サンプルソフトは電源 ON 時にルータに対して DHCP 処理(IP アドレスの取得処理)を実行します。



DHCP 処理が成功すると「My IP Address」に取得した IP アドレスを表示します。

d．Windows 側のテストプログラム「TCP_IP_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、ミスポ社、ARM 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は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

・ARM DS-5 は、ARM 社の登録商標、または商品名称です。
・RZ および RZ/A1H 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

・「RZ/A1H 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