

## RX64M, RX71M グループ

R01AN2608JJ0103

Rev.1.03

TCP/IP プロトコルスタックを用いた産業向けネットワークソリューション 2016.02.29

### RX Driver Package Application

---

#### 要旨

本資料は、TCP/IP プロトコルスタック (M3S-T4-Tiny) を用いた産業向けネットワークソリューションのアプリケーションノートです。

本アプリケーションノートには、ネットワークミドルウェア (DHCP クライアント、DNS クライアント、HTTP サーバ、FTP サーバ) モジュールを使用したメインプログラムのサンプル・コードが含まれており、RX64M, RX71M グループ用 RX Driver Package と組み合わせることで、各種ネットワークシステムを構築できます。

#### 動作確認デバイス

RX64M グループ (Renesas Starter Kit+ RX64M)

RX71M グループ (Renesas Starter Kit+ RX71M)

#### 使用した RX Driver Package 環境

RX64M, RX71M グループ RX Driver Package Ver.1.02 (R01AN2606JJ)

お客様の製品にてご利用される際は、お客様の環境に合わせて十分に評価してください。

また、本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

## 目次

1. 概要	3
1.1 本アプリケーションノートについて	3
1.2 動作環境	4
1.3 モジュール構成	5
1.4 ファイル構成	7
1.5 プロジェクトについて	8
2. 開発環境の入手	9
2.1 e <sup>2</sup> studio の入手方法	9
2.2 コンパイラパッケージの入手方法	9
3. プロジェクトの構築	10
3.1 ワークスペースの作成	10
3.2 プロジェクトの作成	11
3.3 プロジェクトのインポート	13
3.4 変更情報	17
3.4.1 コンフィギュレーションの変更	17
3.4.2 プロジェクト設定の変更	24
4. 動作確認	27
4.1 プロジェクトのビルド	27
4.2 デバッグの準備	28
4.2.1 機器の構成	28
4.2.2 評価ボードの設定	31
4.2.3 クライアント PC の設定	34
4.2.4 USB メモリの準備	38
4.3 プロジェクトのデバッグ	39
4.3.1 HTTP サーバ機能の確認	42
4.3.2 FTP サーバ機能の確認	45
5. ネットワークミドルウェア仕様	46
6. メインプログラム仕様	47
6.1 ファイル構成	47
6.2 モジュール一覧	48
6.3 処理フロー	49
7. ユーザ定義関数について	57
8. 補足	58
8.1 USB ドライバの制限事項	58
8.2 本アプリケーションの制限事項	58
8.3 DHCP クライアント及び FTP サーバの制限事項	58
8.4 無償評価版の「RX ファミリ用 C/C++コンパイラパッケージ」を利用する場合の注意事項	58

## 1. 概要

### 1.1 本アプリケーションノートについて

本アプリケーションノートでは、メインプログラムと RX64M, RX71M グループ用 RX Driver Package 内蔵のネットワークミドルウェア、TCP/IP プロトコルスタック（以下、M3S-T4-Tiny と略す）、Ethernet ドライバ、M3S-TFAT-Tiny FAT ファイルシステム（以下、TFAT と略す）、USB ドライバ（ホストマスストレージクラスドライバ USB HMSC と Basic Firmware）、ボードサポートパッケージ（以下、BSP と略す）の FIT モジュール等と組み合わせて評価するまでの手順について説明します。

本アプリケーションノートは、Renesas Starter Kit+ for RX64M ・ Renesas Starter Kit+ for RX71M（以降、RSK と表記）上で動作します。

## 1.2 動作環境

本アプリケーションノートの動作環境を以下に示します。

表 1-1 動作環境

対応 MCU	RX64M グループ
評価ボード	Renesas Starter Kit+ RX64M <a href="http://japan.renesas.com/products/tools/introductory_tools/renesas_starter_kits/rsk_rx64m/index.jsp">http://japan.renesas.com/products/tools/introductory_tools/renesas_starter_kits/rsk_rx64m/index.jsp</a>
統合開発環境 (IDE)	e <sup>2</sup> studio V4.1.0 以降
クロスツール	RX ファミリ用 C/C++コンパイラパッケージ V2.03.00 以降
エミュレータ	E1 (Renesas Starter Kit+ RX64M に同梱), E20
RX Driver Package	RX64M, RX71M グループ RX Driver Package Ver.1.02 (R01AN2606JJ) 注

表 1-2 動作環境

対応 MCU	RX71M グループ
評価ボード	Renesas Starter Kit+ RX71M <a href="http://japan.renesas.com/products/tools/introductory_tools/renesas_starter_kits/rsk_rx71m/index.jsp">http://japan.renesas.com/products/tools/introductory_tools/renesas_starter_kits/rsk_rx71m/index.jsp</a>
統合開発環境 (IDE)	e <sup>2</sup> studio V4.1.0 以降
クロスツール	RX ファミリ用 C/C++コンパイラパッケージ V2.03.00 以降
エミュレータ	E1 (Renesas Starter Kit+ RX71M に同梱), E20
RX Driver Package	RX64M, RX71M グループ RX Driver Package Ver.1.02 (R01AN2606JJ) 注

【注】本アプリケーションノートは、上記の RX Driver Package に入っているモジュールと組み合わせて動作を確認しています。本アプリケーションノートで使用するモジュールを、別のモジュールと入れかえた場合については、各自で動作を確認してください。

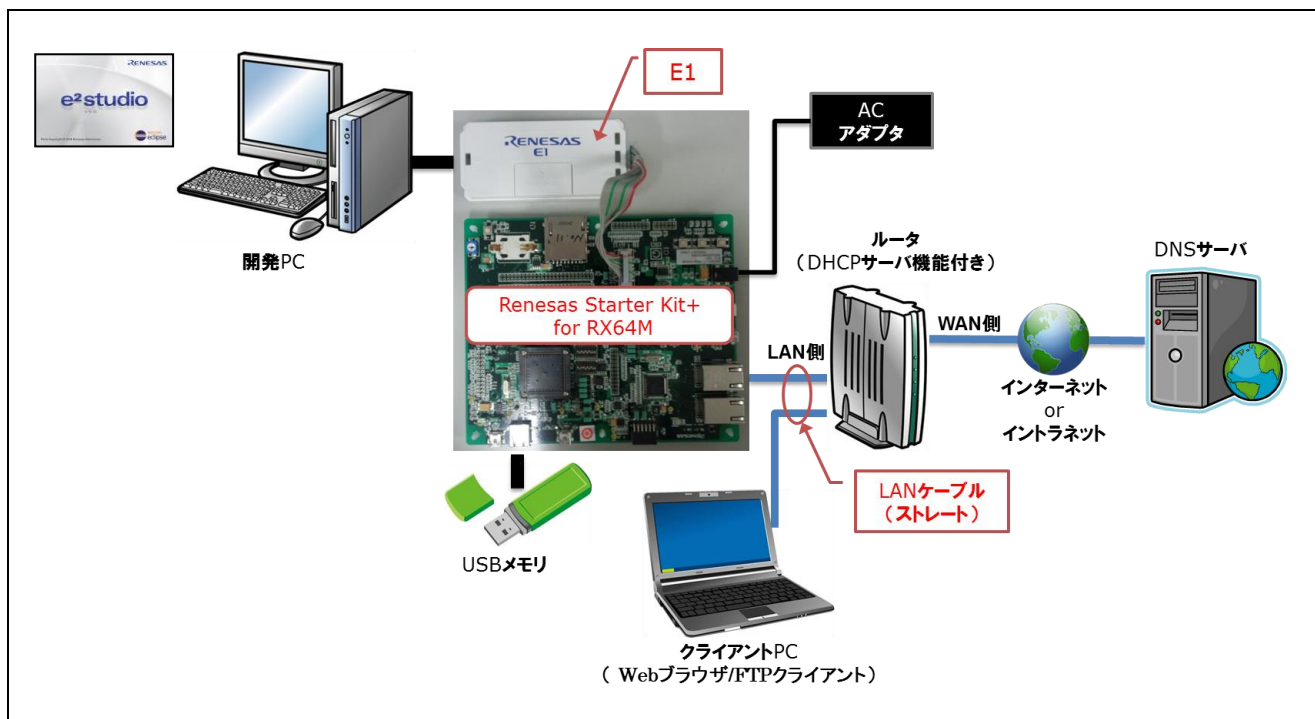


図 1-1 動作環境の例

### 1.3 モジュール構成

本アプリケーションノートでのモジュール構成とモジュール一覧を以下に示します。

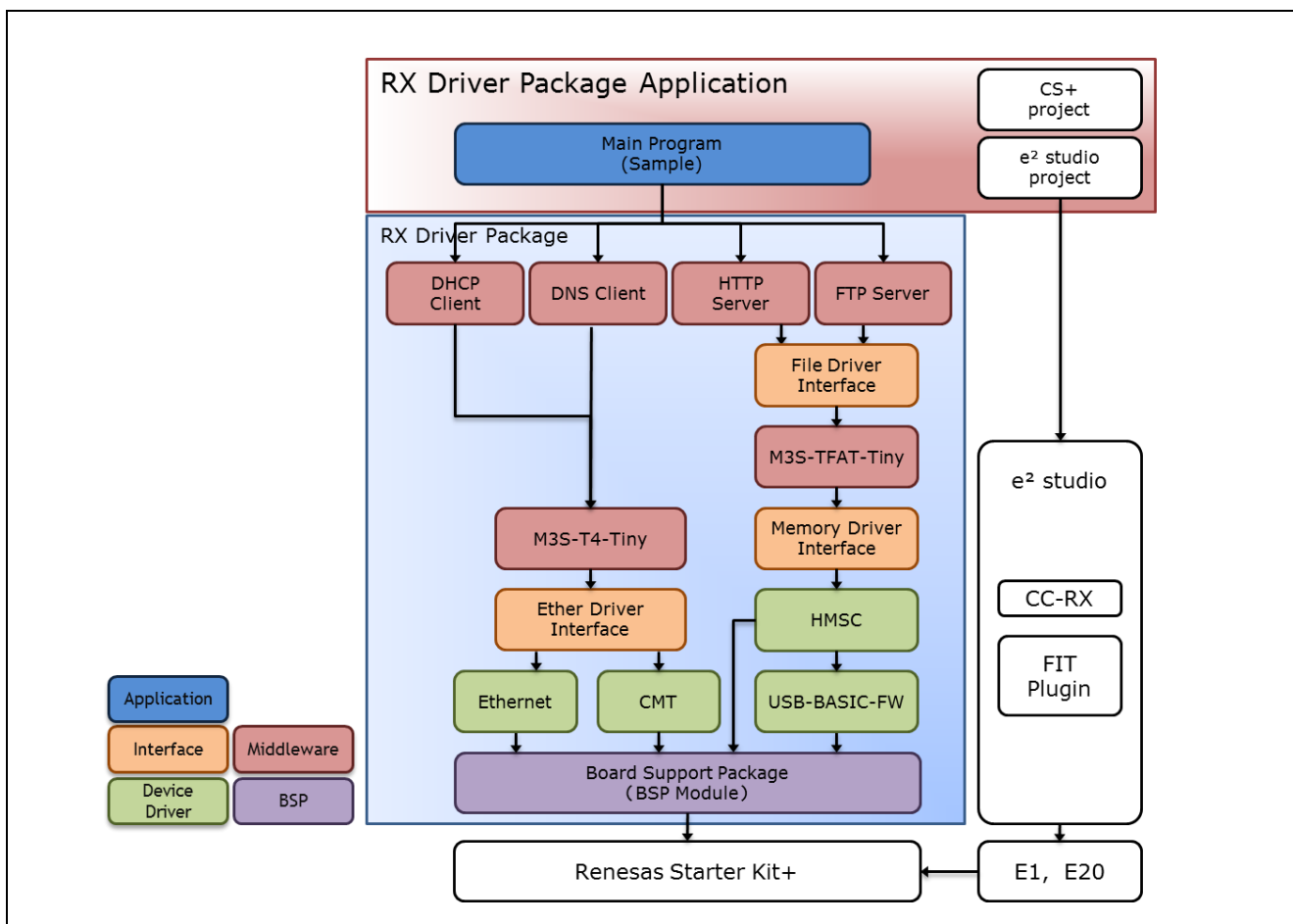


図 1-2 モジュール構成イメージ

表 1-3 モジュール一覧

種類	モジュール名	FIT モジュール名	Rev.
Board Support Package	ボードサポートパッケージ(BSP)	r_bsp	3.00
Device Driver	コンペアマッチタイマ(CMT)	r_cmt_rx	2.60
Device Driver	USB Basic Firmware	r_usb_basic	1.10
Device Driver	USB Host Mass Storage Class	r_usb_hmsc	1.10
Device Driver	イーサネットコントローラ(ETHERC)	r_ether_rx	1.02
Middleware	組み込み用 M3S-T4-Tiny モジュール【注 1】	r_t4_rx	2.02
Interface	Ethernet ドライバと組み込み用 TCP/IP	r_t4_driver_rx	1.02
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた DHCP クライアントモジュール	r_t4_dhcp_client_rx	1.03
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた DNS クライアントモジュール	r_t4_dns_client_rx	1.02
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた FTP サーバモジュール	r_t4_ftp_server_rx	1.03
Middleware	組み込み用 TCP/IP M3S-T4-Tiny を用いた Web サーバモジュール	r_t4_http_server_rx	1.04
Interface	FTP/Web サーバ用ファイルドライバモジュール	r_t4_file_driver_rx	1.01
Middleware	オープンソース FAT ファイルシステム M3S-TFAT-Tiny モジュール	r_tfat_rx	3.02
Interface	M3S-TFAT-Tiny メモリドライバインタフェースモジュール	r_tfat_driver_rx	1.02
Application	メインプログラム	src	1.01

【注 1】本パッケージには、評価版の「M3S-T4-Tiny (TCP/IP プロトコルスタックライブラリ)」が含まれています。製品版については、以下の URL を参照してください。

<http://japan.renesas.com/mw/t4>

## 1.4 ファイル構成

本アプリケーションノートのファイル構成を以下に示します。

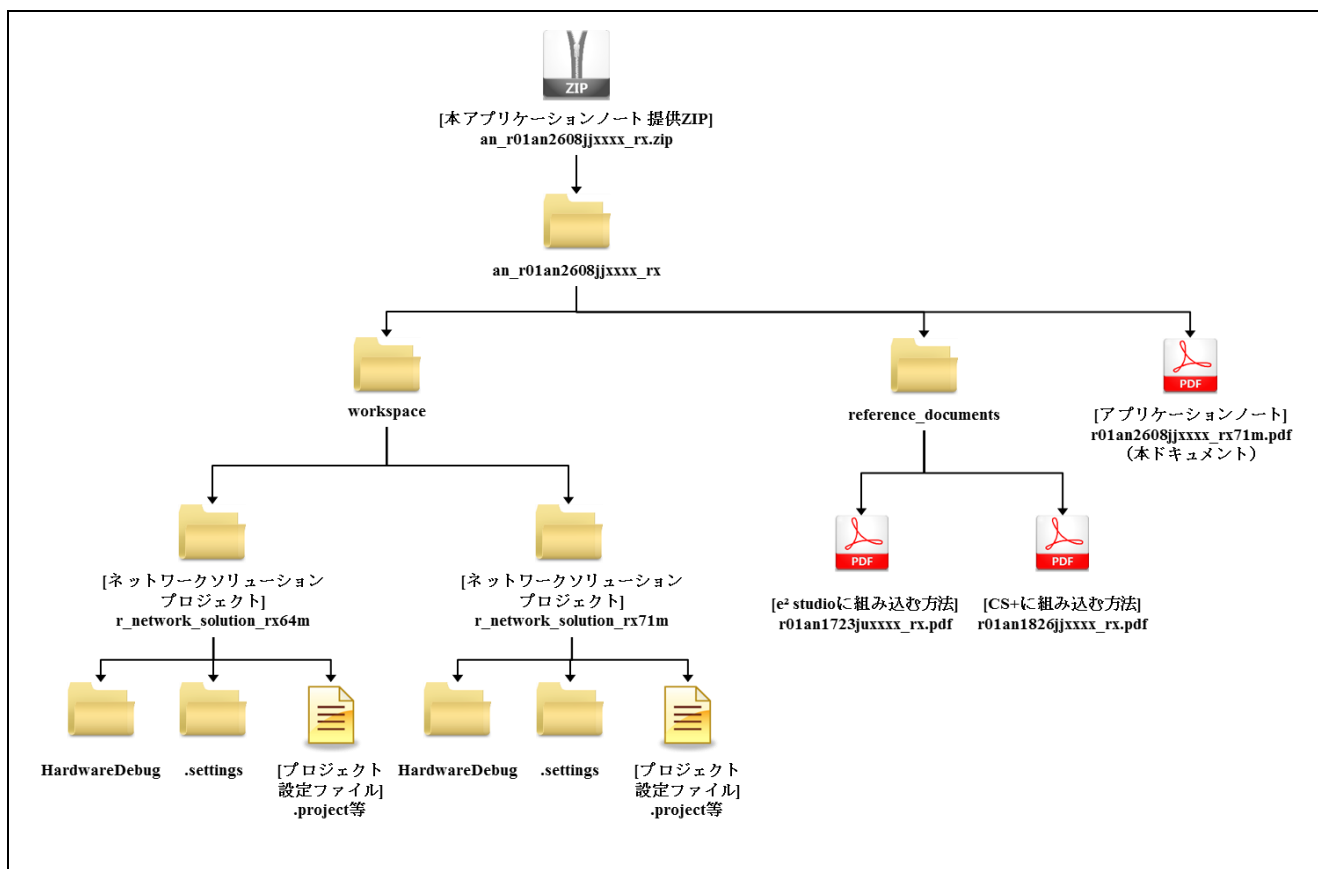


図 1-3 ファイル構成

本アプリケーションノート提供 ZIP ファイルを解凍すると、同名のフォルダが作成され、その中に各フォルダやファイルが入っています。

「workspace」フォルダの下にある「ネットワークソリューションプロジェクト (r\_network\_solution\_rx64m)」又は「ネットワークソリューションプロジェクト (r\_network\_solution\_rx71m)」は、本アプリケーションを構築したプロジェクトです。e<sup>2</sup> studio のワークスペースにインポートすることでアプリケーションの動作確認が可能です。

「reference\_documents」フォルダの中には、FIT モジュールを各開発環境で使用するための説明書が入っています。「e<sup>2</sup> studio に取り込む方法 (R01AN1723JU)」は、FIT プラグインを使用して、FIT モジュールを e<sup>2</sup> studio のプロジェクトに組み込む方法について説明したドキュメントです。「CS+に取り込む方法 (R01AN1826JJ)」は、FIT モジュールを CS+ のプロジェクトに組み込む方法について説明したドキュメントです。

## 1.5 プロジェクトについて

本アプリケーションノートには、本アプリケーションをビルド及び評価するための、e<sup>2</sup> studio 用のプロジェクトが付属しています。プロジェクトには、ビルド設定を保存した「ビルド構成」と、デバッグ設定を保存した「デバッグ構成」を登録しています。

以下に、プロジェクトに登録してあるビルド構成とデバッグ構成の一覧を示します。

表 1-4 プロジェクト設定

	構成名	説明
ビルド構成	HardwareDebug (Debug on hardware)	デバッグ情報付きのロードモジュールを生成するための構成です。
デバッグ構成	HardwareDebug (E1)	「HardwareDebug (Debug on hardware)」で生成したロードモジュールを使用して、E1 エミュレータ経由でのハードウェアデバッグを行います。

また、ターゲット固有の設定は以下のとおりです。

表 1-5 ターゲット固有の設定

項目	設定
ツールチェーン・バージョン	v2.03.00
デバッグ・ハードウェア	E1 (RX)
データ・エンディアン	Little-endian data
ターゲットの選択	R5F564MLCxFC (RX64M LQFP 176pin) 又は R5F571MLCxFC (RX71M LQFP 176pin)
Renesas RTOS サポート (注1)	None

注1：使用するためには、別途 OS 環境をダウンロードする必要があります。



## 2. 開発環境の入手

### 2.1 e<sup>2</sup> studio の入手方法

以下の URL にアクセスし、e2 studio をダウンロードしてください。

[http://japan.renesas.com/e2studio\\_download](http://japan.renesas.com/e2studio_download)

なお、本ドキュメントは e2 studio V4.1.0 以降を使用することを前提としています。V4.1.0 よりも古い Ver. を使用した場合、e<sup>2</sup> studio の一部機能を使用できない可能性があります。ダウンロードする場合、ホームページに掲載されている最新 Ver. の e2 studio を入手してください。

### 2.2 コンパイラパッケージの入手方法

以下の URL にアクセスし、RX ファミリー用 C/C++コンパイラパッケージをダウンロードしてください。

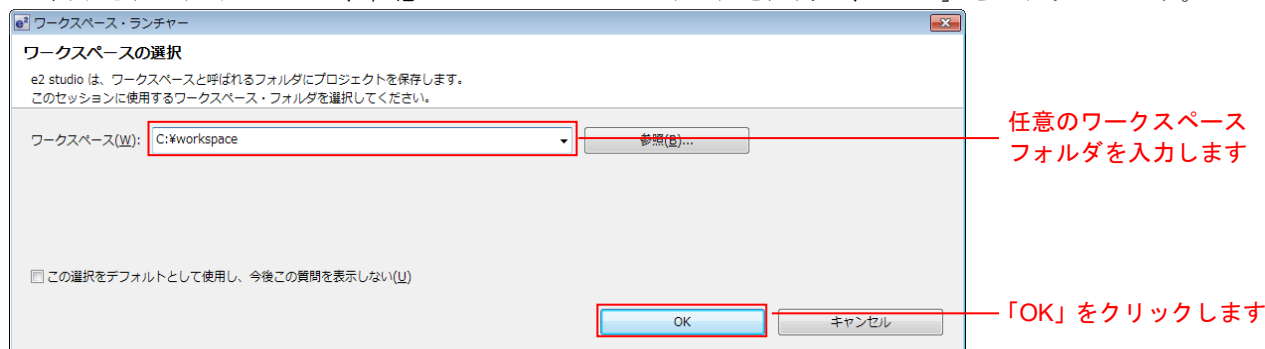
[http://japan.renesas.com/e2studio\\_download](http://japan.renesas.com/e2studio_download)

### 3. プロジェクトの構築

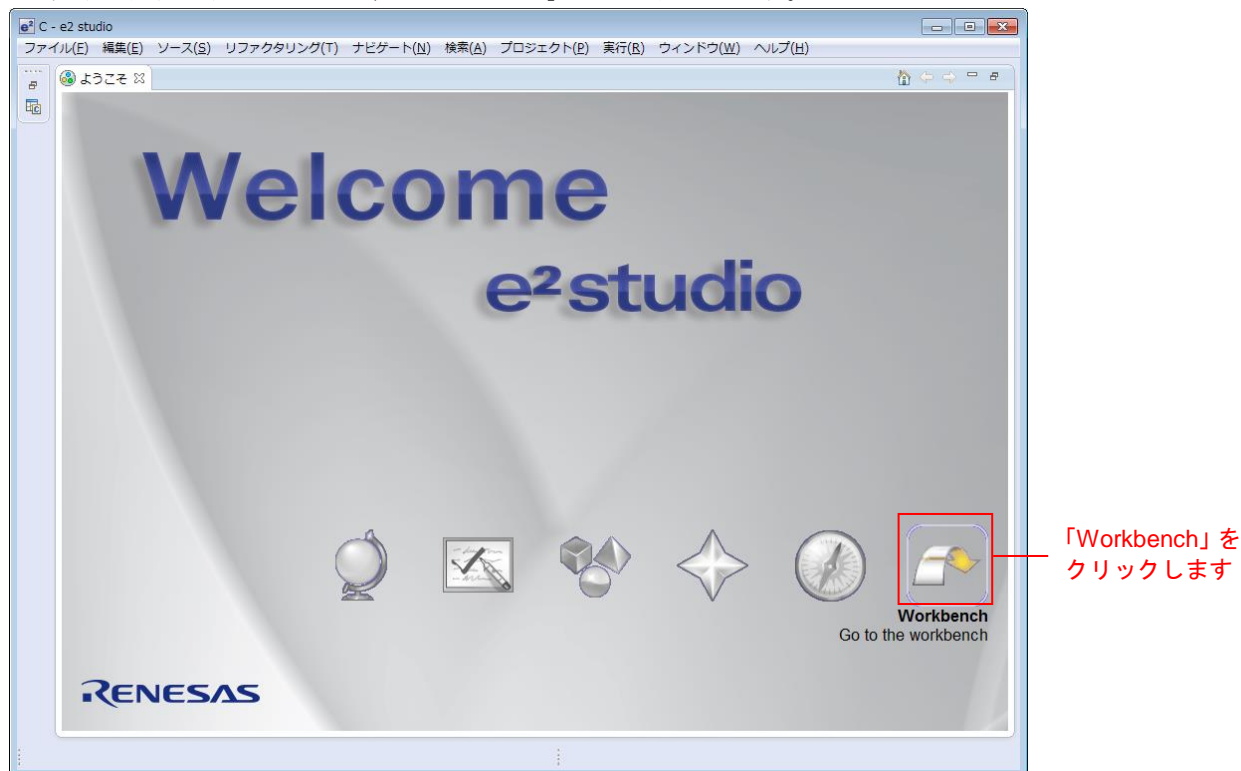
本アプリケーションノートは、環境構築済みのプロジェクトを同梱しています。e2 studio のスマート・ブラウザを用いてプロジェクトをインポートする手順について、以下に説明します。

#### 3.1 ワークスペースの作成

1. e<sup>2</sup> studio を起動します。
2. 表示されたダイアログに、任意のワークスペースフォルダを入力し、「OK」をクリックします。



3. 以下の画面が表示されたら、「Workbench」をクリックします。

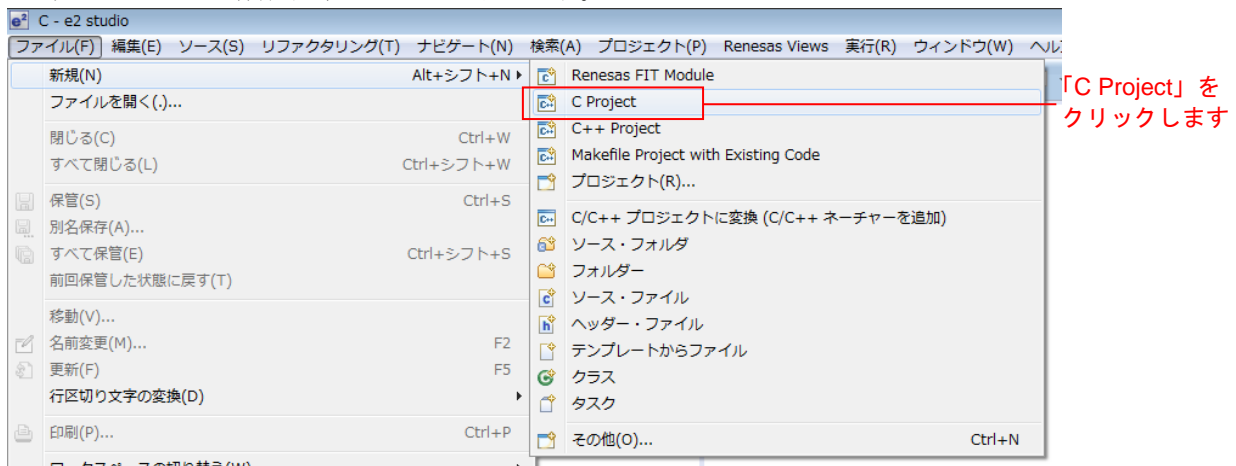


### 3.2 プロジェクトの作成

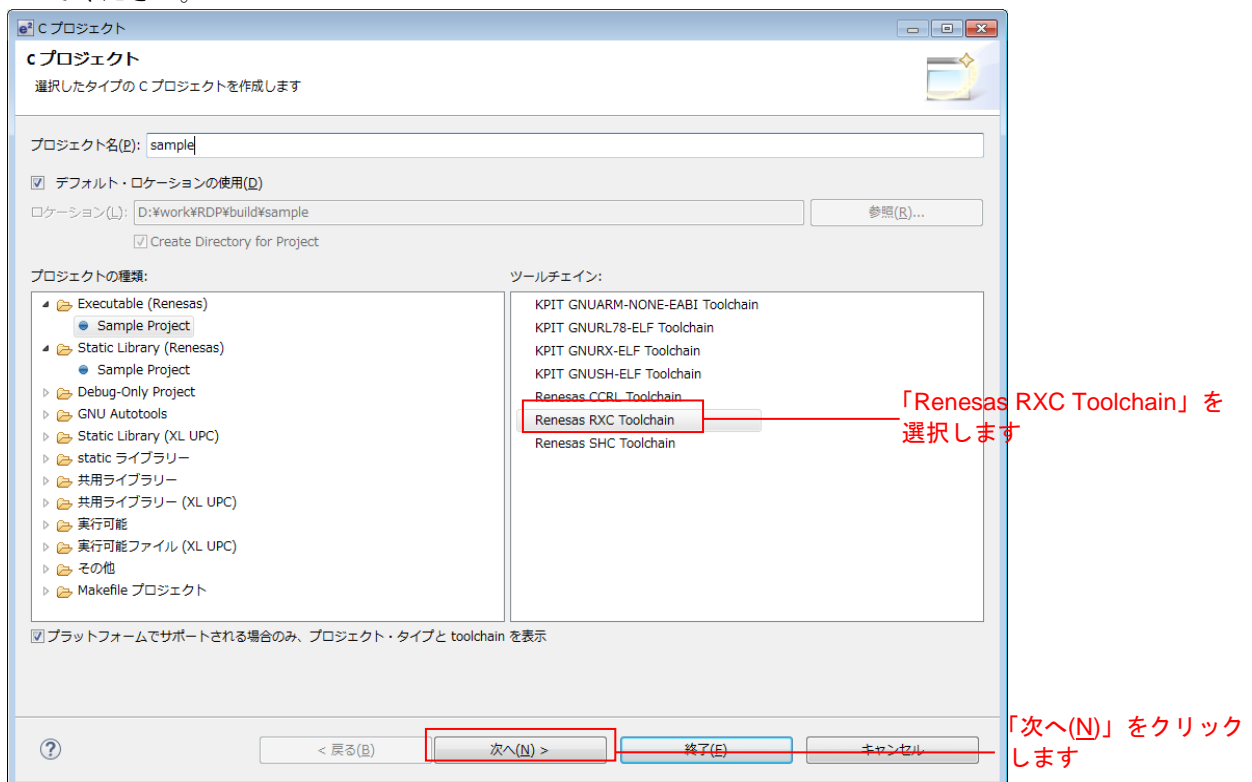
スマート・ブラウザーの機能を使用するときは、対象となるプロジェクトあるいはファイルを選択しておく必要があります。スマート・ブラウザーの機能を使用するために、対象デバイスを指定したプロジェクトを作成します（注1）。

注1：ここで作成するプロジェクトは、スマート・ブラウザーを使用するために作成するためのダミープロジェクトです。インポートするプロジェクトの設定は「表 1-5 ターゲット固有の設定」を参照してください。

1. [ファイル(F)] → [新規(N)] → [C Project]の順にクリックして新しいCプロジェクトを作成します。新規プロジェクト作成ウィザードを起動します。



2. 任意のプロジェクト名を入力し、“Renesas RXC Toolchain”を選択します。[次へ(N)>] をクリックしてください。

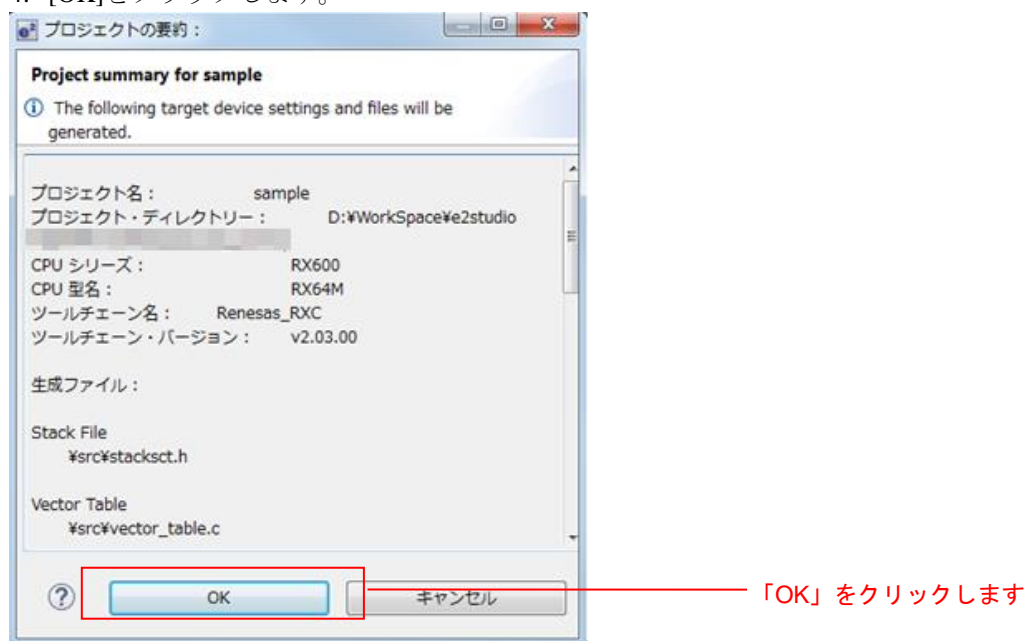


3. 「ターゲットの選択:」を設定します。RX64M の場合「R5F564Mxxxxx」としてください。RX71M の場合「R5F571Mxxxxx」としてください。その他の項目は任意の設定で構いません(注1)。設定が完了したら[終了(F)]をクリックします。

注1: ダミープロジェクト用の設定です。スマート・ブラウザを使用してインポートするプロジェクト環境の設定は「表 1-5 ターゲット固有の設定」を参照ください。



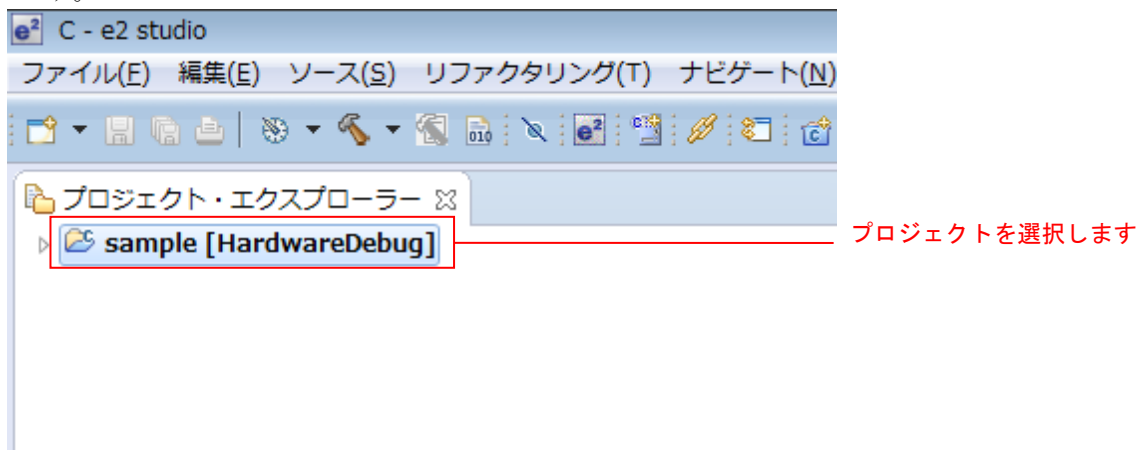
4. [OK]をクリックします。



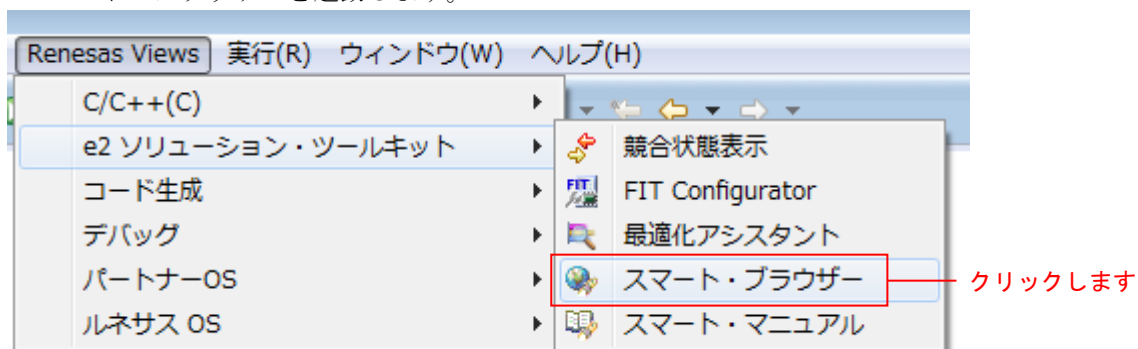
### 3.3 プロジェクトのインポート

メインプログラムのプロジェクトを、作成したワークスペースにインポートします。

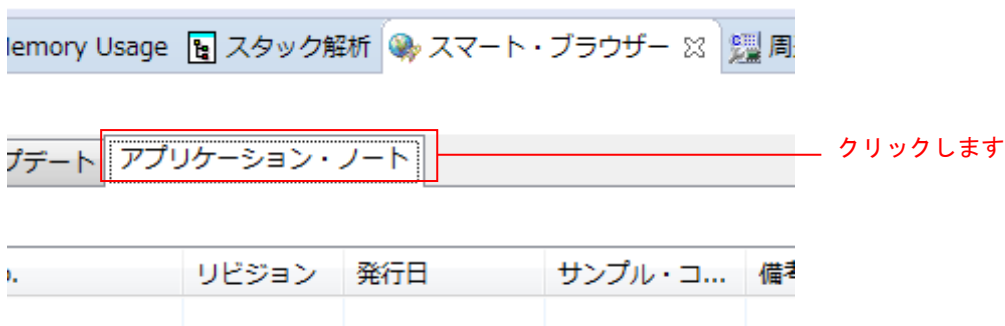
1. プロジェクト・エクスプローラーから「3.2 プロジェクトの作成」で作成したプロジェクトを選択します。



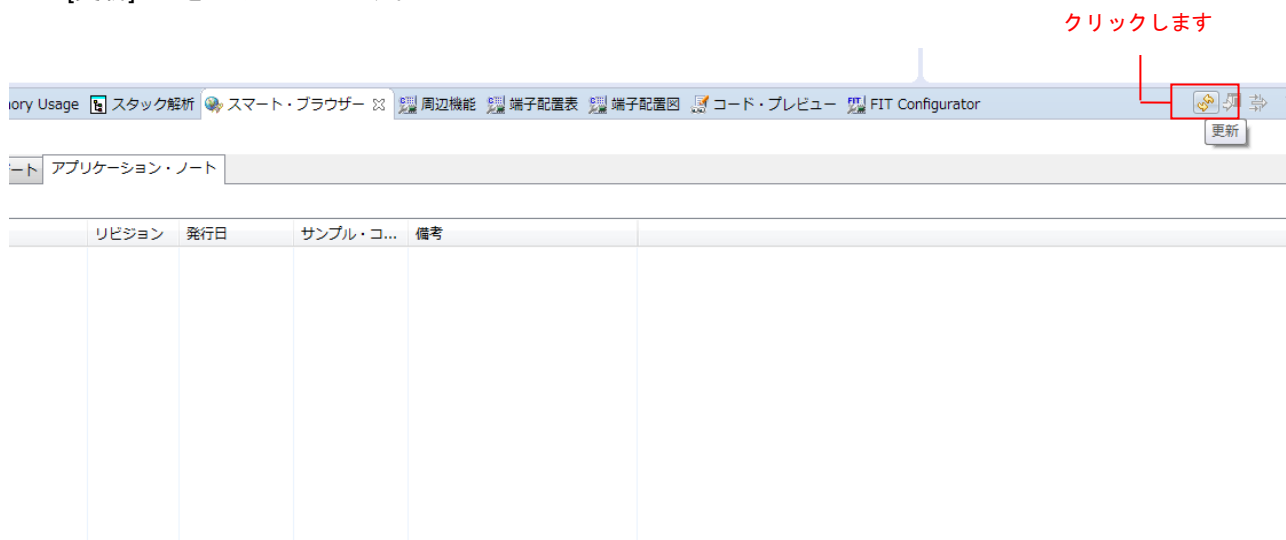
2. [Renesas Views]→[e2 ソリューション・ツールキット]→[スマート・ブラウザー]の順にクリックし、スマート・ブラウザーを起動します。



3. [スマート・ブラウザー]タブの[アプリケーション・ノート]タブをクリックします。

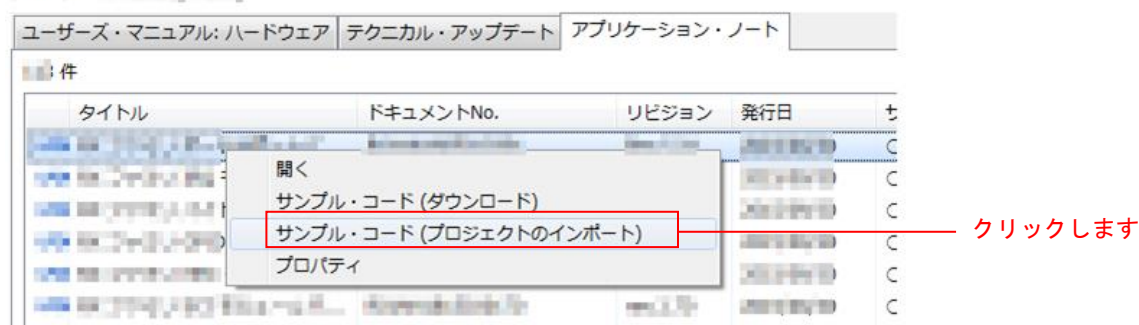


4. [更新]  をクリックします。

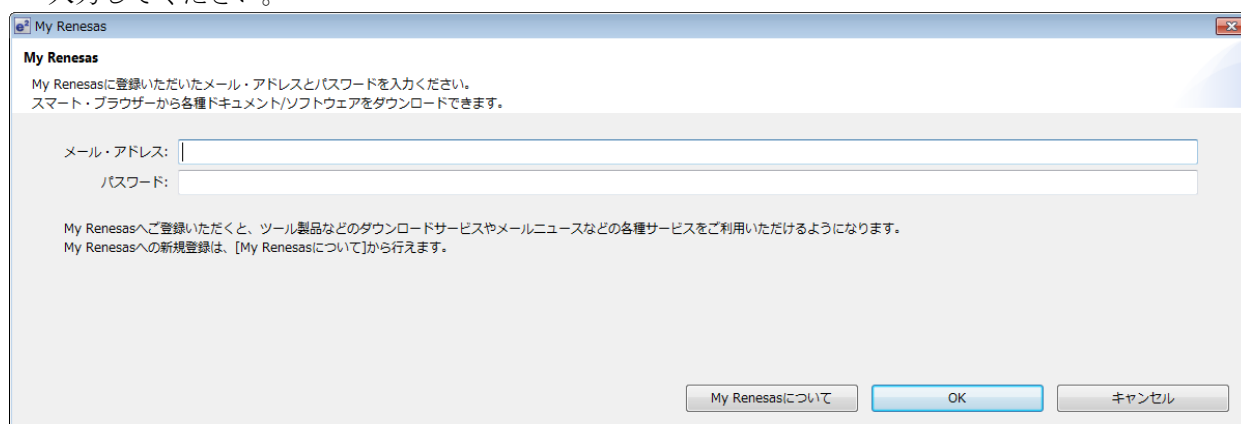


5. 本アプリケーションノートを選択し、右クリックします。コンテキストメニューの[サンプル・コード (プロジェクトのインポート)]をクリックします。(注1)

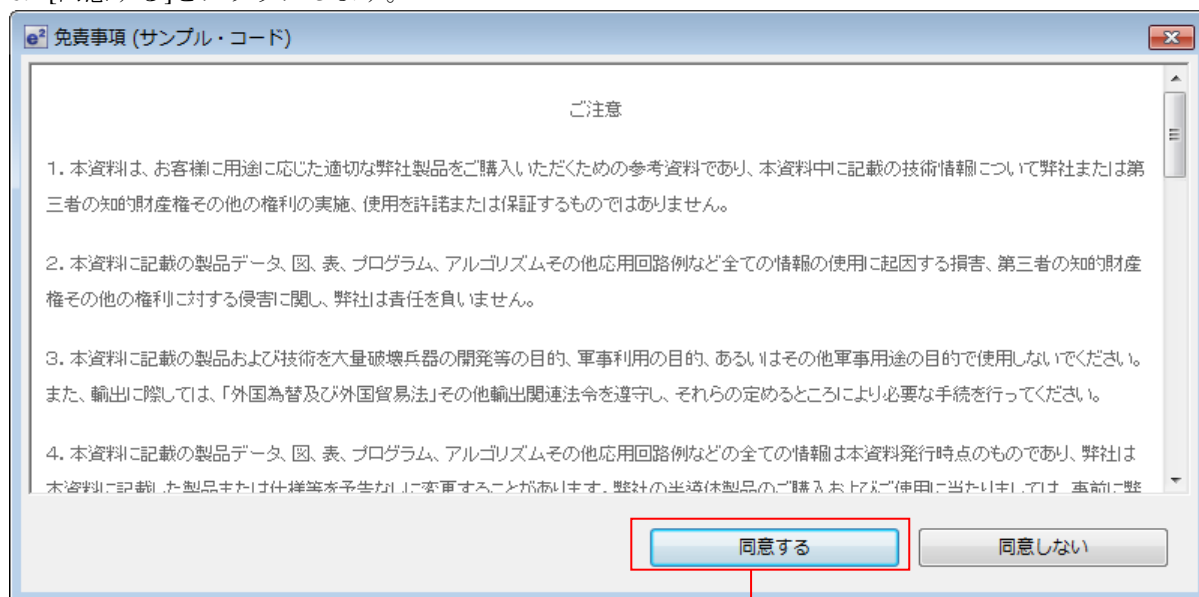
デバイス:        



注1：一度も My Renesas による認証をしていない場合、ファイルをダウンロードする際に「My Renesas」ダイアログがオープンします。ルネサス Web サイトで登録しているメール・アドレスとパスワードを入力してください。

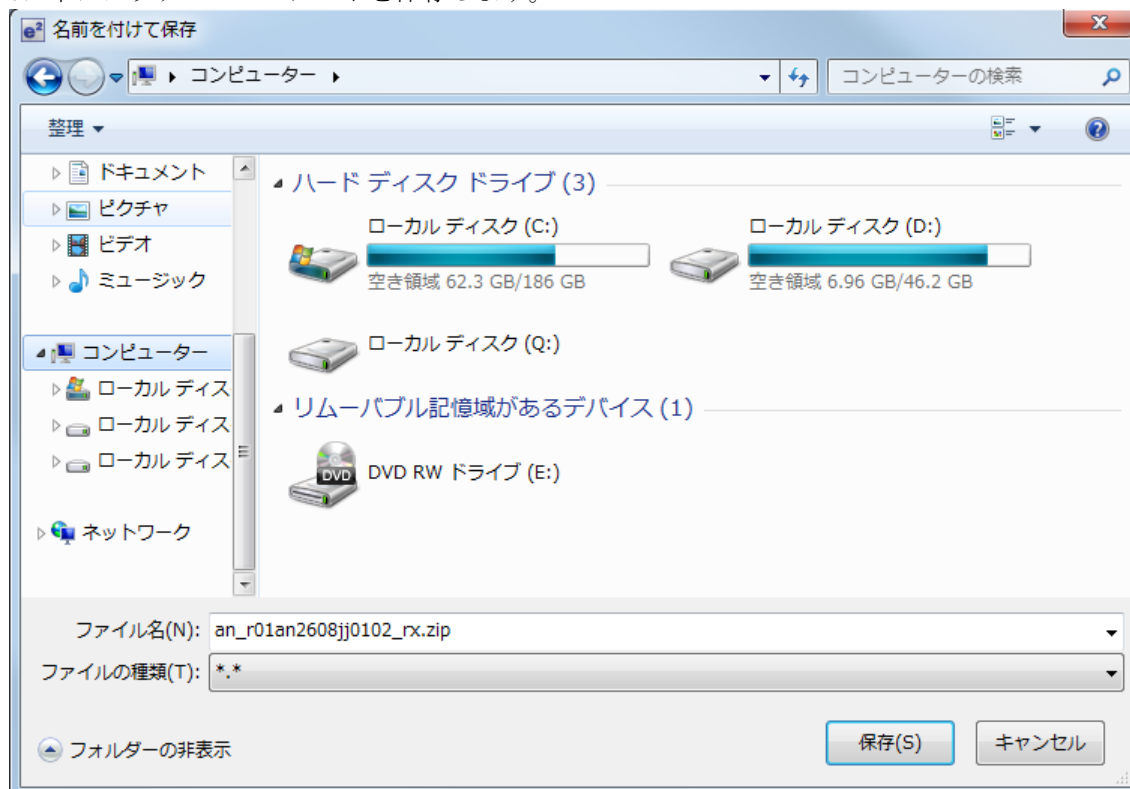


## 6. [同意する]をクリックします。

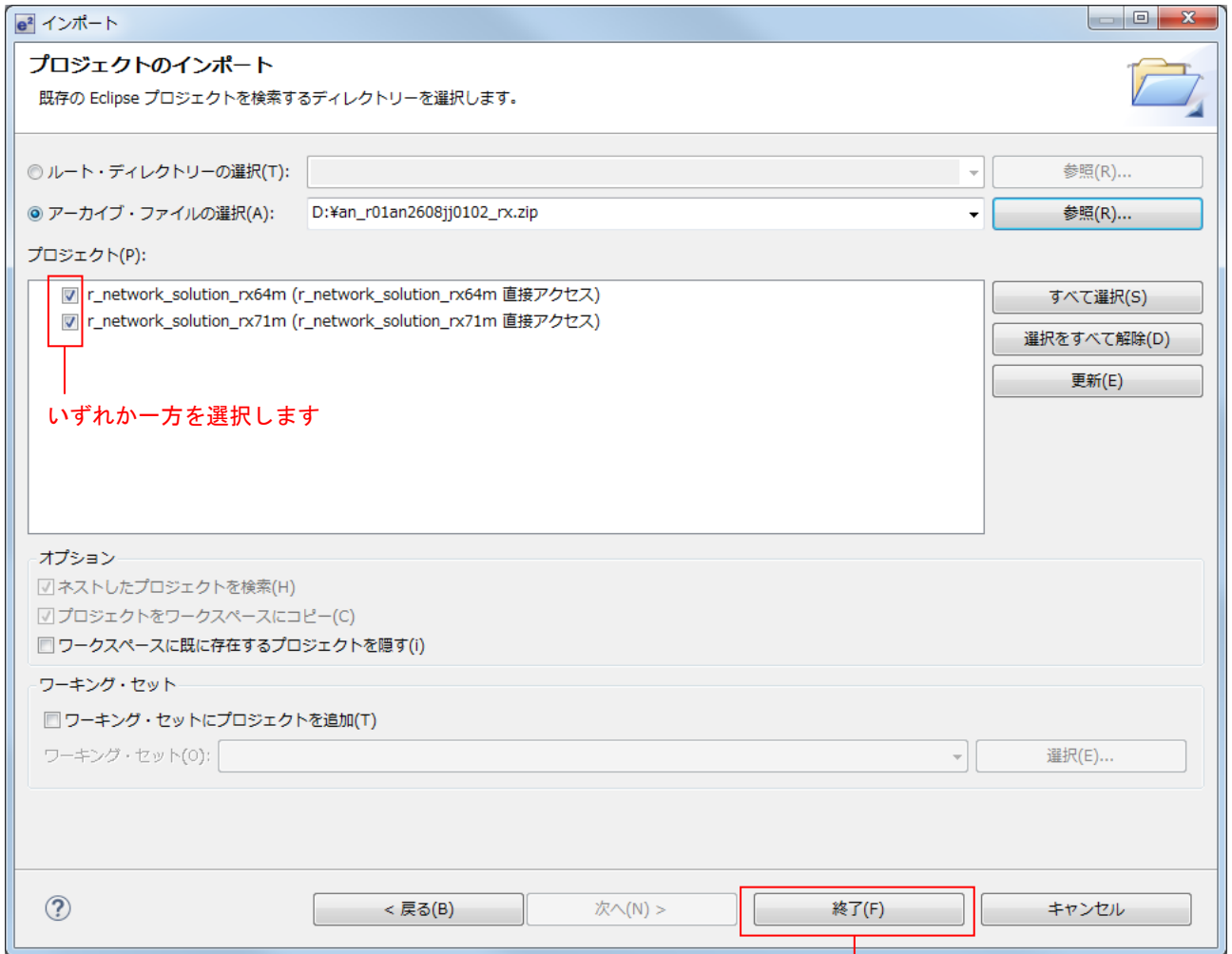


クリックします

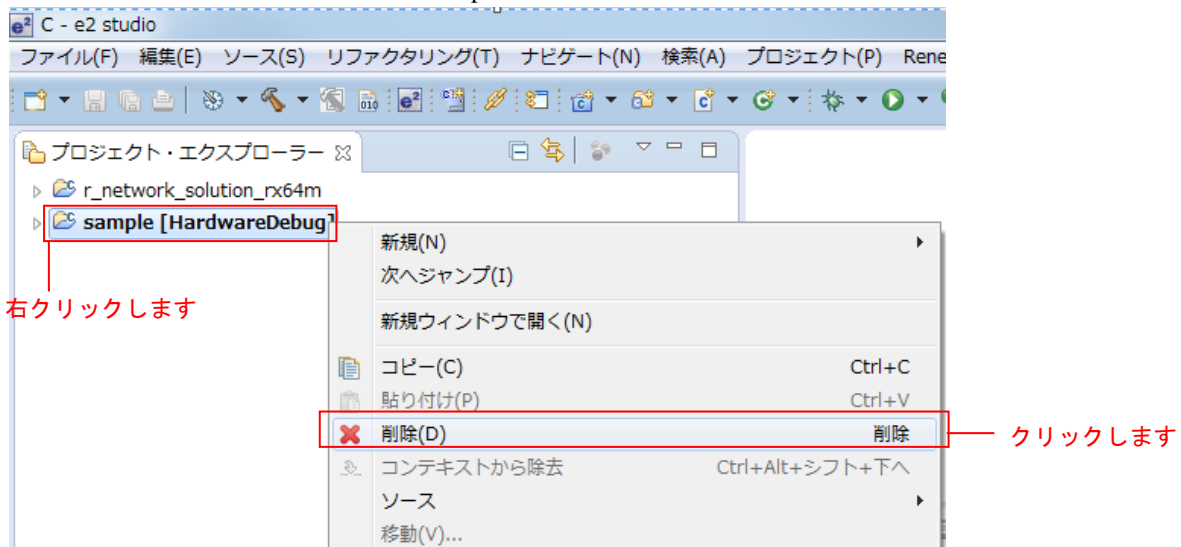
## 7. 本アプリケーションノートを保存します。



8. 「プロジェクト(P):」に表示されている「r\_network\_solution\_rx64m」または「r\_network\_solution\_rx71m」いずれか一方を選択し、「終了(F)」をクリックします。



9. 下図は「r\_network\_solution\_rx64m」を選択した場合です。スマート・ブラウザを使用するために作成したプロジェクト（ここでは sample）は不要ですので削除してください。





### 3.4 変更情報

本プロジェクトでは、本アプリケーションを構成するために各 FIT モジュールのコンフィグレーションファイル設定とプロジェクト設定を変更しています。以下に詳細を示します。

なお、本変更情報は、新規にプロジェクトを構築する場合に参照してください。インポートしたプロジェクトを使用する場合は「4 動作確認」に進んでください。

#### 3.4.1 コンフィギュレーションの変更

本アプリケーションを構成する各 FIT モジュールのコンフィギュレーションファイルを変更します。

コンフィギュレーションファイルの項目と設定内容については、各 FIT モジュールの doc フォルダに入っているマニュアル等を参照してください。

以下にコンフィギュレーションファイルの変更箇所を示します。

##### (1) 割り込みスタックサイズの変更

本アプリケーションは、Ethernet コントローラの割り込みハンドラから、TCP/IP の処理を行っており、約 2.5kbyte の割り込みスタックを必要とします。

r\_bsp のコンフィギュレーションファイルで定義されている割り込みスタックサイズを、以下の様に変更してください。

###### 【r\_config/r\_bsp\_config.h】

```
/* Interrupt Stack size in bytes. The Renesas RX toolchain sets the stack size
using the #pragma stacksize directive.
 * If the interrupt stack is the only stack being used then the user will likely
want to increase the default size
 * below.
 */
#pragma stacksize si=0x1000
```

##### (2) コンペアマッチタイマドライバの設定変更

コンペアマッチタイマの割り込みレベルを、USB ドライバの割り込みレベル (IPR=3) より低く設定します。

###### 【r\_config/r\_cmt\_rx\_config.h】

```
/* The interrupt priority level to be used for CMT interrupts. */
#define CMT_RX_CFG_IPR          (2)
```

## (3) USB ドライバの設定変更

チャンネル0を未使用 (USB\_NOUSE\_PP)、チャンネル1をホスト (USB\_HOST\_PP) にしてください。

**【r\_config/r\_usb\_basic\_config.h】**

```
/*
  Select USB mode to USB IP(USBb)

  USB_HOST_PP : USB Host Mode
  USB_PERI_PP : USB Peripheral Mode
  USB_NOUSE_PP : Not Used (USBb)
*/
#define USB_FUNCSEL_USBIP0_PP  USB_NOUSE_PP

/*
  Select USB mode to USB IP(USBAA/USBA)

  USB_HOST_PP : USB Host Mode
  USB_PERI_PP : USB Peripheral Mode
  USB_NOUSE_PP : Not Used (USBAA/USBA)
*/
#define USB_FUNCSEL_USBIP1_PP  USB_HOST_PP
```

**【r\_config/r\_usb\_hmsc\_config.h】**

```
#define USB_MEDIA_INITIALIZE(data1)  R_tfat_disk_initialize((uint8_t)data1)
```

## (4) T4 の設定変更

T4 の設定を以下のように変更します。

t4\_callback 関数の外部参照宣言をコメントアウトし、新たに http\_callback、ftp\_callback、ftp\_data\_callback、dns\_callback 関数の外部参照宣言を追加してください。

## 【r\_t4\_rx/src/config\_tcpudp.c】

```
#include "r_t4_itcpip.h"
//extern ER t4_callback(ID cepid, FN fncd , VP p_parblk);
extern ER http_callback(ID cepid, FN fncd , VP p_parblk);
extern ER ftp_callback(ID cepid, FN fncd , VP p_parblk);
extern ER ftp_data_callback(ID cepid, FN fncd , VP p_parblk);
extern ER dns_callback(ID cepid, FN fncd , VP p_parblk);
```

ローカルポートの設定を、以下のように変更してください。

## 【r\_t4\_rx/src/config\_tcpudp.c】

```
/** Definition of TCP reception point (only port number needs to be set) **/
T_TCP_CREP tcp_crep[18] =
{
    /* HTTP server use setting below. */
    { 0x0000, { 0, 80 }},
    { 0x0000, { 0, 80 }},
    { 0x0000, { 0, 80 }},
    { 0x0000, { 0, 80 }},
    { 0x0000, { 0, 80 }},
    { 0x0000, { 0, 80 }},
    /* FTP server use setting below. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1024. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1025. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1026. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1024. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1025. */
    { 0x0000, { 0, 21 }},
    { 0x0000, { 0, 20 }}, /* The port number in the active mode is 20, and
the port number in the passive mode is 1026. */
};
```

TCP 通信端点の設定を、以下のように変更してください。

【r\_t4\_rx/src/config\_tcpudp.c】

```

/** Definition of TCP communication end point
    (only receive window size needs to be set) */
T_TCP_CCEP tcp_ccep[18] =
{
    /* { attribute of TCP communication end point,
        top address of transmit window buffer, size of transmit window buffer,
        top address of receive window buffer, size of receive window buffer,
        address of callback routine }
    */
    /* HTTP server use setting below. */
    { 0, 0, 0, 0, 1460, http_callback },
    { 0, 0, 0, 0, 1460, http_callback },
    { 0, 0, 0, 0, 1460, http_callback },
    { 1, 0, 0, 0, 1460, http_callback },
    { 1, 0, 0, 0, 1460, http_callback },
    { 1, 0, 0, 0, 1460, http_callback },
    /* FTP server use setting below. */
    { 0, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 0, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
    { 0, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 0, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
    { 0, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 0, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
    { 1, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 1, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
    { 1, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 1, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
    { 1, 0, 0, 0, 128, ftp_callback },           /* FTP control connection */
    { 1, 0, 0, 0, 1460, ftp_data_callback },    /* FTP data connection */
};

```

2-MSL 待ち時間を、1 分から 10ms に変更してください。

【r\_t4\_rx/src/config\_tcpudp.c】

```

/** 2MSL wait time (unit:10ms) */
const UH _tcp_2msl[] =
{
    (1), /* 10 ms */
    (1), /* 10 ms */
};

```

UDP 通信端点の設定を、以下のように変更してください。

【r\_t4\_rx/src/config\_tcpudp.c】

```

/** Definition of UDP communication end point */
T_UDP_CCEP udp_ccep[1] =
{
    /* only setting port number */
    { 0x0000, { 0, 1365 }, dns_callback },
};

```

## (5) HTTP サーバの設定変更

MAX\_CGI\_FILE を 2 に設定します。

【r\_config/r\_t4\_http\_server\_rx\_config.h】

```
// If user uses CGI function, please specify cgi file name, and number of cgi files.
#define MAX_CGI_FILE      2
```

CGI\_FILE\_NAME\_TABLE\_LIST を以下の様に設定します。

【r\_config/r\_t4\_http\_server\_rx\_config.h】

```
/*#define CGI_FILE_NAME_TABLE_LIST ¥*/
/* {"cgi_smpl.cgi", NULL}, ¥*/
extern ER cgi_sample_function(ID cepid, void *res_info);
extern ER cgi_dns_demo_function(ID cepid, void *res_info);
extern ER cgi_dns_demo_pending_release(ID cepid, void *res_info);
#define CGI_FILE_NAME_TABLE_LIST ¥
    {"cgi_smpl.cgi", cgi_sample_function, NULL},    ¥
    {"dns_demo.cgi", cgi_dns_demo_function, cgi_dns_demo_pending_release}, ¥
```

HTTP サーバで使用する通信端点の数を、r\_t4\_rx/src/config\_tcpudp.c の tcp\_ccep テーブル数に合わせて 6 個に変更します。

【r\_config/r\_t4\_http\_server\_rx\_config.h】

```
// set same value number of CEPID in config_tcpudp.c
#define HTTP_TCP_CEP_NUM 6
```

## (6) DNS クライアントの設定変更

動作確認で使用する DNS サーバのアドレスを設定します。任意のアドレスを設定してください（下記は一例です）。

【r\_config/r\_t4\_dns\_client\_rx\_config.h】

```
/******
Configuration Options
*****/
//#define DNS_IP_ADDR_1    172,30,11,5    /* DNS server IP address (primary) */
//#define DNS_IP_ADDR_2    10,29,72,33    /* DNS server IP address (secondary) */
#define DNS_IP_ADDR_1      10,164,247,5    /* DNS server IP address (primary) */
#define DNS_IP_ADDR_2      10,29,72,33    /* DNS server IP address (secondary) */
```

## (7) FTP サーバの設定変更

T4 の TCP 通信端点の設定に合わせて、FTP サーバで使用する TCP 通信端点の位置を設定します。

【r\_config/r\_t4\_ftp\_server\_rx\_config.h】

```
#include "r_t4_http_server_rx_config.h"
/*****
Configuration Options
*****/
#define FTP_TCP_CEP_NUM 12 // Adapt myself to the communication endpoint number
of config_tcpudp.c

#define FTP_START_TCP_CEP HTTP_TCP_CEP_NUM // starting position of the
communication endpoint in config_tcpudp.c
```

## (8) FTP/Web サーバ用ファイルドライバの設定変更

USB を使用する設定に変更します。

【r\_config/r\_t4\_file\_driver\_rx\_config.h】

```
/* Only one is effective.
(priority is given to SERVER_FILE_DRV_USE_EXTERNAL)

Setting to 1 : used
           0 : unused
*/
#define SERVER_FILE_DRV_USE_EXTERNAL (1) // USB, SD card etc
#define SERVER_FILE_DRV_USE_INTERNAL (0) // internal-RAM / external-SDRAM
```

## (9) M3S-TFAT-Tiny メモリドライバの設定変更

USB を使用する設定に変更します。

## 【r\_config/r\_tfat\_driver\_rx\_config.h】

```
/* Number of logical drives to be used.
   Setting to 0      : unused memory
   other           : number of logical drives
   (USB and SDHI can be used together.)
*/
#define TFAT_USB_DRIVE_NUM      (1)
#define TFAT_SDHI_DRIVE_NUM    (0)
#define TFAT_USB_MINI_DRIVE_NUM (0)

/* allocate a drive number
   <valid define>
   TFAT_CTRL_USB   : for USB
   TFAT_CTRL_SDHI  : for SDHI
   NULL            : unallocated drive

   MAX 10 drives (TFAT module spec)

   ex.)
   #define TFAT_DRIVE_ALLOC_NUM_0 TFAT_CTRL_USB
   #define TFAT_DRIVE_ALLOC_NUM_1 TFAT_CTRL_SDHI
   #define TFAT_DRIVE_ALLOC_NUM_2 TFAT_CTRL_SDHI
   #define TFAT_DRIVE_ALLOC_NUM_3 TFAT_CTRL_USB
*/
#define TFAT_DRIVE_ALLOC_NUM_0 TFAT_CTRL_USB
#define TFAT_DRIVE_ALLOC_NUM_1 NULL
```

## (10) Ethernet の設定変更

Ethernet コントローラから PHY IC を制御するために使用する PHY IC のチャンネル設定を ch0 に変更します。

## 【r\_config/r\_ether\_rx\_config.h】

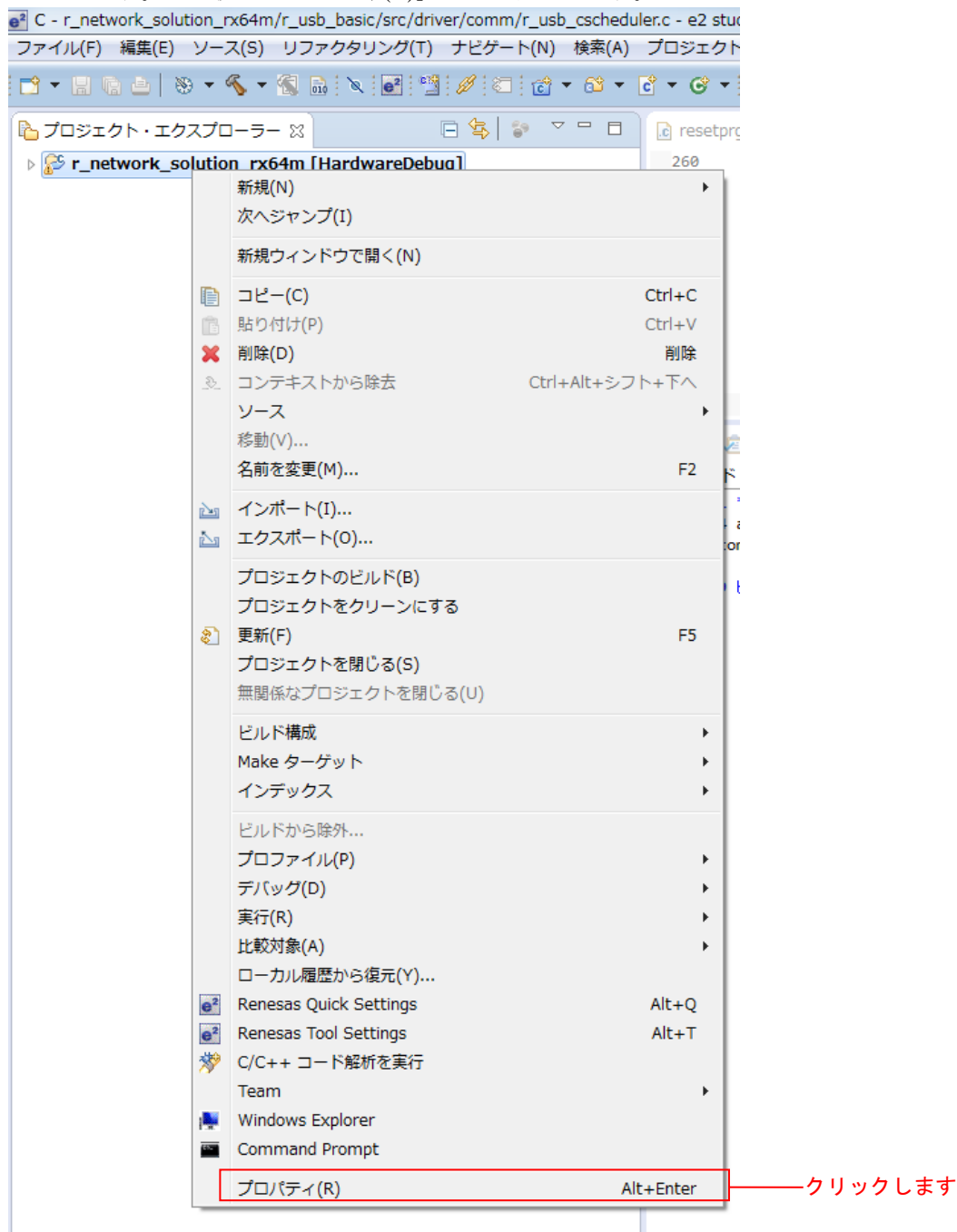
```
/* The register bus of PHY0/1 for ETHER0/1 select
   0 = The access of the register of PHY uses ETHER0.
   1 = The access of the register of PHY uses ETHER1.
*/
#define ETHER_CFG_CH0_PHY_ACCESS (0)
#define ETHER_CFG_CH1_PHY_ACCESS (0)
```

### 3.4.2 プロジェクト設定の変更

ビルド時の設定はデフォルト設定から「表 3-1 変更したビルド設定」に示す内容に変更しています。

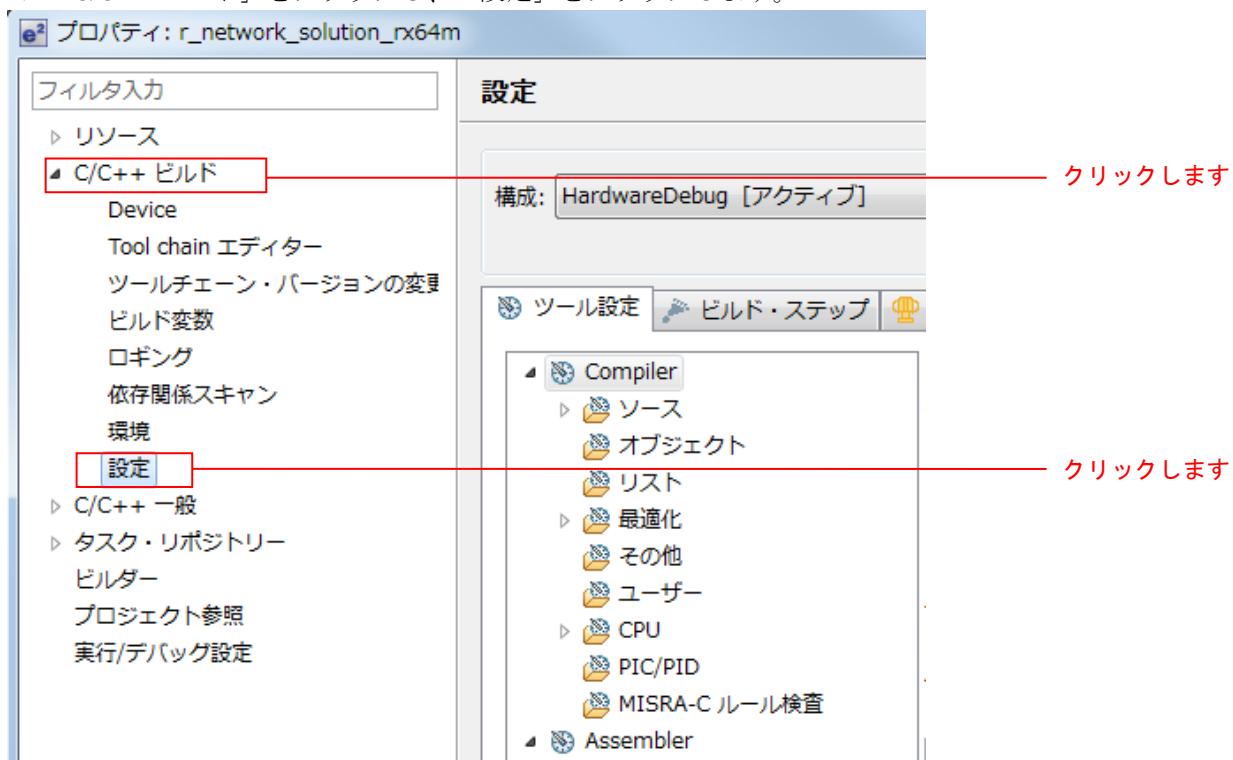
プロジェクト設定を確認する場合は、以下の手順で行ってください。

1. e<sup>2</sup> studio の対象プロジェクト (r\_network\_solution\_rx64m or r\_network\_solution\_rx71m) を選択し右クリックします。その後「プロパティ(R)」をクリックします。



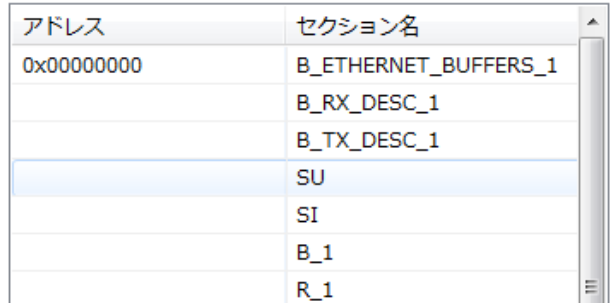


2. 「C/C++ ビルド」をクリックし、「設定」をクリックします。



3. 「ツール設定」タブから「表 3-1 変更したビルド設定」の内容を参照してください。

表 3-1 変更したビルド設定

項目	変更内容	説明																
Compiler -ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加する	各 FIT モジュールで設定が必要なインクルードパスを設定する。  「FIT Configurator」を使用し、各 FIT モジュールを組み込む場合、自動で設定されます。																
Linker -入力	<p>■Little エンディアンの場合 "\${workspace_loc:\${ProjName}/r_tfat_rx/lib/tfat_rx600_little.lib}"を追加する(注 1)</p> <p>■Big エンディアンの場合 "\${workspace_loc:\${ProjName}/r_tfat_rx/lib/tfat_rx600_big.lib}"を追加する(注 1)</p>	<p>TFAT を使用する際には設定する。 (FAT ファイルシステムドライバを使用する際に必須)</p> <p>「FIT Configurator」を使用し場合でも、自動で設定されません。<b>新規にプロジェクトを作成する場合、ユーザが設定してください(注 2)</b></p>																
	<p>■Little エンディアンの場合 "\${workspace_loc:\${ProjName}/r_t4_rx/lib/T4_Library_rxv2_ether_little.lib}"を追加する(注 1)</p> <p>■Big エンディアンの場合 "\${workspace_loc:\${ProjName}/r_t4_rx/lib/T4_Library_rxv2_ether_big.lib}"を追加する(注 1)</p>	<p>Ethernet を使用する際には設定する。</p> <p>「FIT Configurator」を使用し場合でも、自動で設定されません。<b>新規にプロジェクトを作成する場合、ユーザが設定してください(注 2)</b></p>																
Linker -セクション	RAM 領域に B_ETHERNET_BUFFERS_1 セクション、B_RX_DESC_1 セクション、B_TX_DESC_1 セクションを追加する	<p>Ethernet を使用する際には設定する。</p> <p>「FIT Configurator」を使用し場合でも、自動で設定されません。<b>新規にプロジェクトを作成する場合、ユーザが設定してください。</b></p> <p>&lt;設定例&gt; セクション・ビューア:</p>  <table border="1"> <thead> <tr> <th>アドレス</th> <th>セクション名</th> </tr> </thead> <tbody> <tr> <td>0x00000000</td> <td>B_ETHERNET_BUFFERS_1</td> </tr> <tr> <td></td> <td>B_RX_DESC_1</td> </tr> <tr> <td></td> <td>B_TX_DESC_1</td> </tr> <tr> <td></td> <td>SU</td> </tr> <tr> <td></td> <td>SI</td> </tr> <tr> <td></td> <td>B_1</td> </tr> <tr> <td></td> <td>R_1</td> </tr> </tbody> </table>	アドレス	セクション名	0x00000000	B_ETHERNET_BUFFERS_1		B_RX_DESC_1		B_TX_DESC_1		SU		SI		B_1		R_1
アドレス	セクション名																	
0x00000000	B_ETHERNET_BUFFERS_1																	
	B_RX_DESC_1																	
	B_TX_DESC_1																	
	SU																	
	SI																	
	B_1																	
	R_1																	

(注 1) 各 FIT モジュールを組み込むプロジェクトを作成する際に必要な設定変更です。この設定については各 FIT モジュールの doc フォルダに入っているマニュアル等を参照してください。

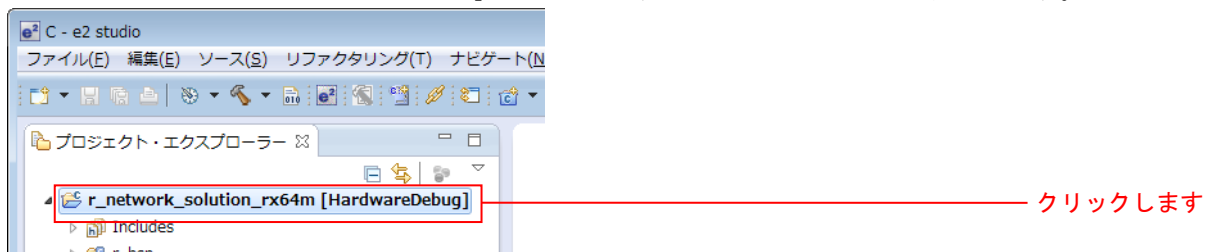
(注 2) e2 studio V4.1 以降では自動設定が可能です。但し、今回使用した FIT モジュールが仕様に対応していないため、自動設定機能を使用できません。最新の FIT モジュールにて対応予定です。

## 4. 動作確認

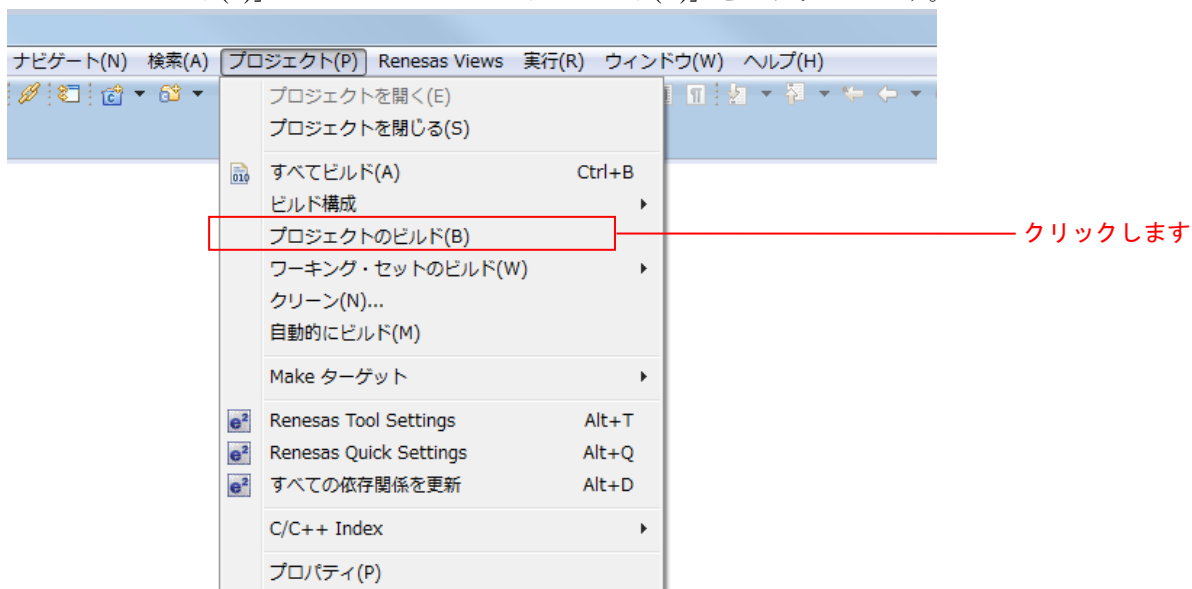
### 4.1 プロジェクトのビルド

以下の手順に従い、プロジェクトをビルドしてロードモジュールを生成します。

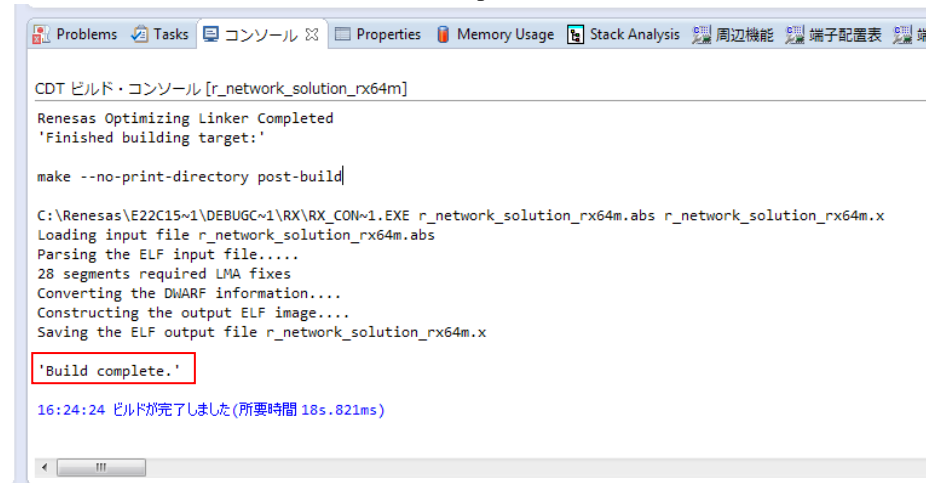
1. 「プロジェクト・エクスプローラー」からビルドするプロジェクトをクリックします。



2. 「プロジェクト(P)」タブの「プロジェクトのビルド(B)」をクリックします。



3. 「コンソール」パネルに「Build complete.」と表示されたらビルド完了です。



## 4.2 デバッグの準備

### 4.2.1 機器の構成

デバッグを開始する前に、評価ボードを準備します。

必要な機器の一覧と構成を以下に示します。

表 4-1 機器構成

No.	機器	補足
1	開発 PC	開発を行う PC です。
2	評価ボード (Renesas Starter Kit+ for RX64M 又は Renesas Starter Kit+ for RX71M)	
3	USB メモリ	FAT、または FAT32 でフォーマットしたもの。
4	クライアント PC <ul style="list-style-type: none"> <li>Web ブラウザ</li> <li>FTP クライアント (FFFTP 等)</li> </ul>	開発 PC で代用可能です。
5	DHCP サーバ、DNS サーバ	DHCP クライアントと DNS クライアントの動作を確認する場合に必要です。
6	ネットワーク環境によって以下のいずれかが必要になります。 <ul style="list-style-type: none"> <li>ルータ 1 台</li> <li>スイッチング・ハブ 1 台</li> <li>LAN ケーブル (ストレート又はクロス) 1~3 本</li> </ul>	ネットワーク環境の例は以下の図を参照してください。

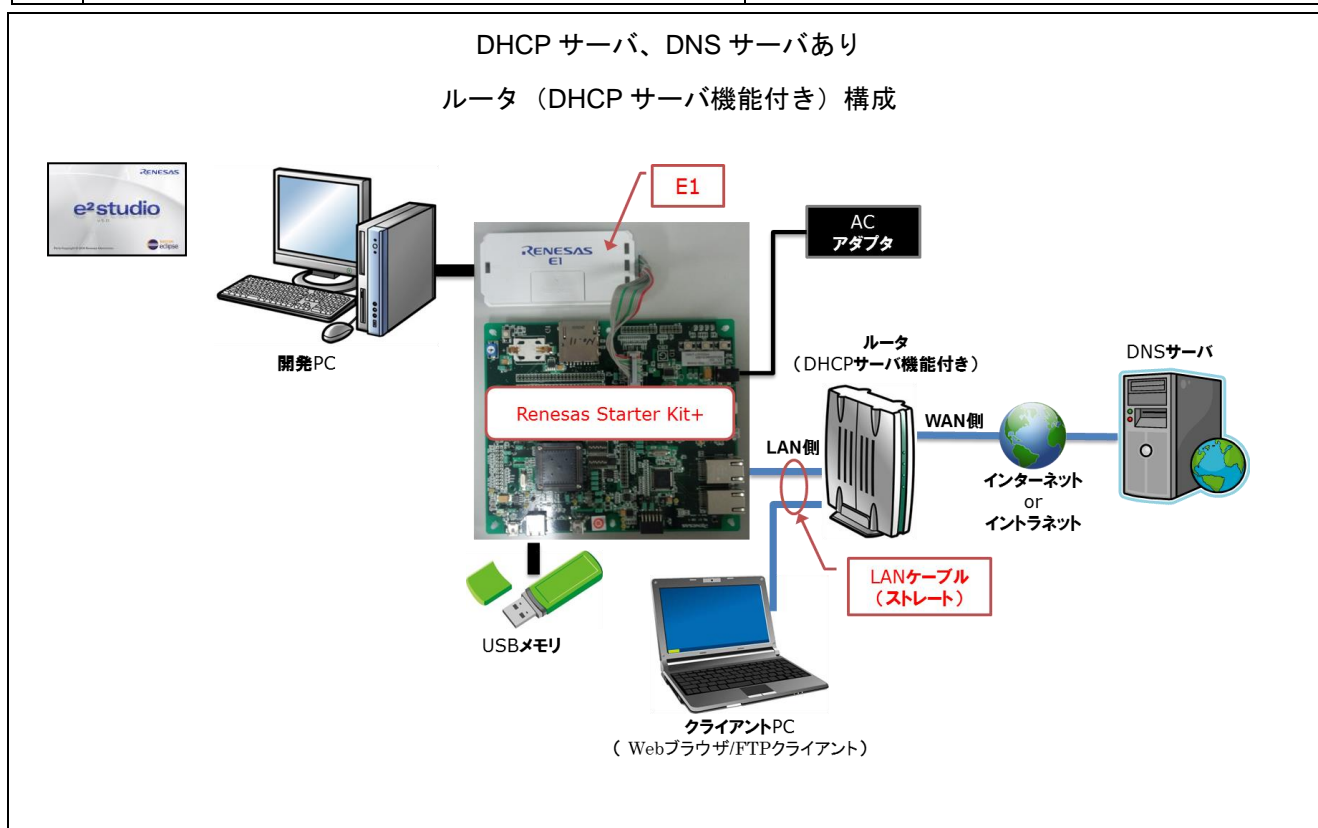


図 4-1 DHCP サーバ、DNS サーバあり ルータ (DHCP サーバ機能付き) 構成



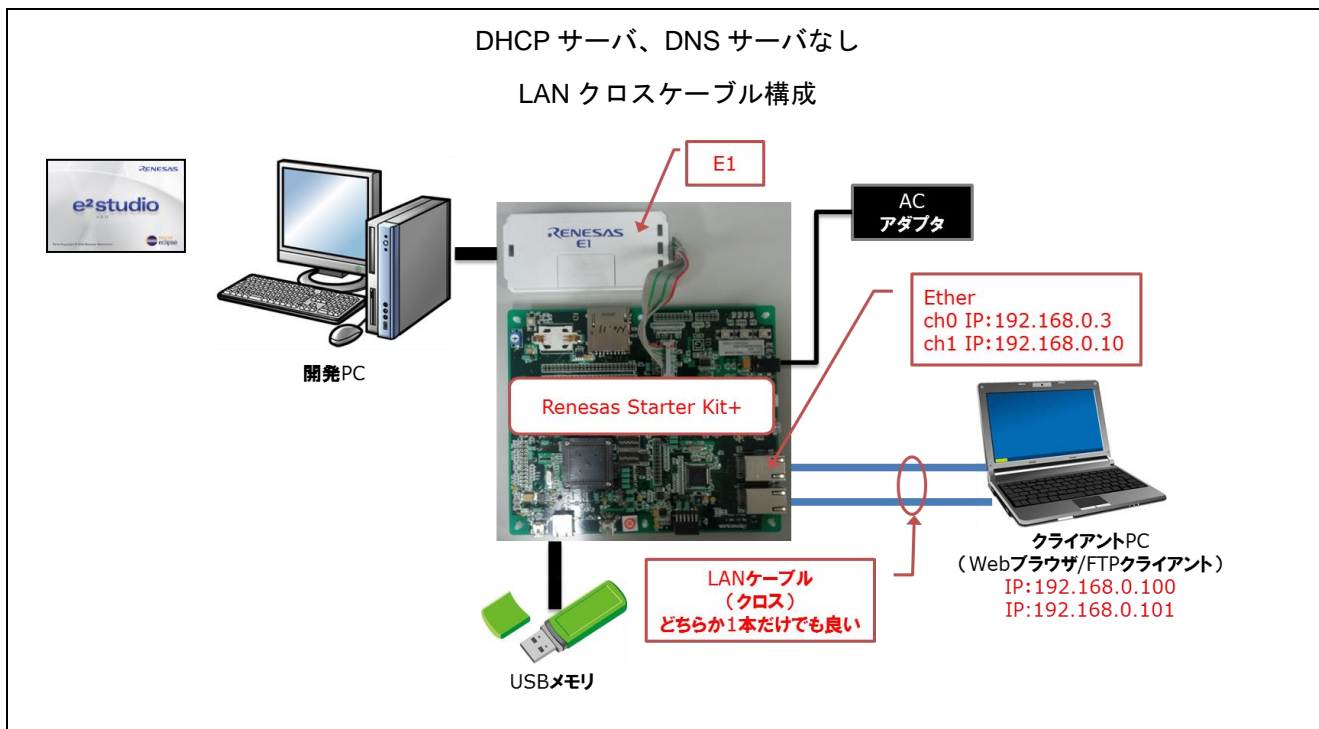


図 4-4 DHCP サーバ、DNS サーバなし –LAN クロスケーブル構成



## 4.2.2 評価ボードの設定

本アプリケーションを動作させるための、評価ボードの設定を以下に示します。

## (1) USB 設定

1. USB ch0 のモード (ホスト/ペリフェラル) を設定します。r\_usb\_config.h の「USB\_FUNCSEL\_USBIP0\_PP」の設定に合わせて、ジャンパ J2 及び J6 を設定します。
2. USB ch1 のモード (ホスト/ペリフェラル) を設定します。r\_usb\_config.h の「USB\_FUNCSEL\_USBIP1\_PP」の設定に合わせて、ジャンパ J7 及び J9 を設定します。

表 4-2 USB 設定一覧

No.	設定内容	ジャンパ	設定内容	動作確認で使用した設定
1	USB0 をホストモードで使用する場合 (USB_FUNCSEL_USBIP0_PP = USB_HOST_PP)	J2	1-2 をショート	未使用
		J6	2-3 をショート	
	USB0 をペリフェラルモードで使用する場合 (USB_FUNCSEL_USBIP0_PP = USB_PERI_PP)	J2	2-3 をショート	未使用
		J6	1-2 をショート	
2	USB1 をホストモードで使用する場合 (USB_FUNCSEL_USBIP1_PP = USB_HOST_PP)	J7	1-2 をショート	使用
		J9	2-3 をショート	
	USB1 をペリフェラルモードで使用する場合 (USB_FUNCSEL_USBIP1_PP = USB_PERI_PP)	J7	2-3 をショート	未使用
		J9	1-2 をショート	

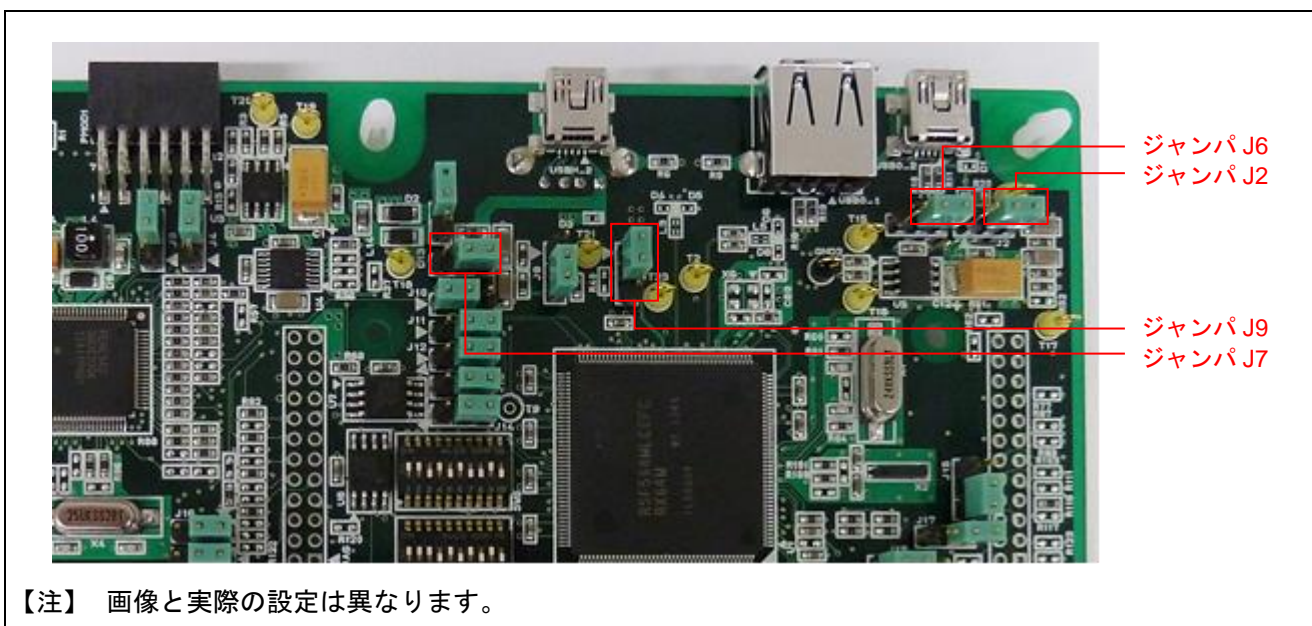


図 4-5 Renesas Starter Kit+ for RX64M/RX71M のジャンパ位置【USB】

## (2) Ethernet 設定

1. Ethernet コントローラから PHY IC を制御するために使用する PHY IC のチャンネルを指定します。  
r\_ether\_rx\_config.h の「ETHER\_CFG\_CH0\_PHY\_ACCESS」と「ETHER\_CFG\_CH1\_PHY\_ACCESS」の  
設定に合わせて、ジャンパ J3 及び J4 を設定します。
2. Ethernet コントローラ (ET0) を使用する場合、スイッチ、ジャンパを設定します。
3. Ethernet コントローラ (ET1) を使用する場合、ジャンパを設定します。

表 4-3 Ethernet 設定一覧

No.	設定内容	ジャンパ & スイッチ		動作確認で 使用した設定
		No.	設定内容	
1	PHY IC を ch0 で制御する	J3	1-2 をショート	使用
		J4	1-2 をショート	
	PHY IC を ch1 で制御する	J3	2-3 をショート	未使用
		J4	2-3 をショート	
2	Ethernet コントローラ (ET0) を使用する場合	J10	1-2 をショート	使用
		J11	2-3 をショート	
		J12	2-3 をショート	
		J13	2-3 をショート	
		J14	2-3 をショート	
		SW5	1 On	
			2 On	
			3 Off	
			4 On	
			5 Off	
			6 On	
			7 Off	
			8 Off	
			9 On	
			10 Off	
		SW6	1 On	
			2 On	
			3 On	
			4 Off	
			5 On	
			6 Off	
			7 On	
			8 Off	
			9 On	
			10 Off	
		SW7	1 On	
			2 Off	
			3 Off	
			4 On	
			5 Off	
			6 Off	
			7 On	
			8 Off	
9 On				
10 Off				



3	Ethernet コントローラ (ET1) を使用する場合	J16	1-2 をショート	未使用
		J18	1-2 をショート	
		J20	1-2 をショート	

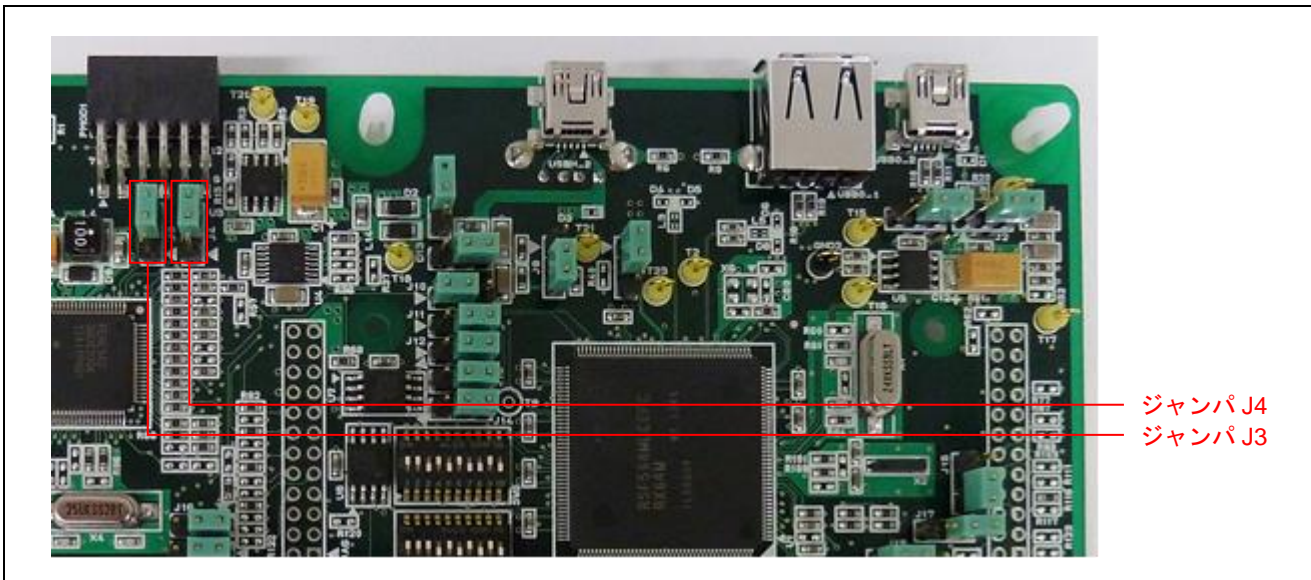


図 4-6 Renesas Starter Kit+ for RX64M/RX71M のジャンパ位置 【Ethernet コントローラ PHY IC】

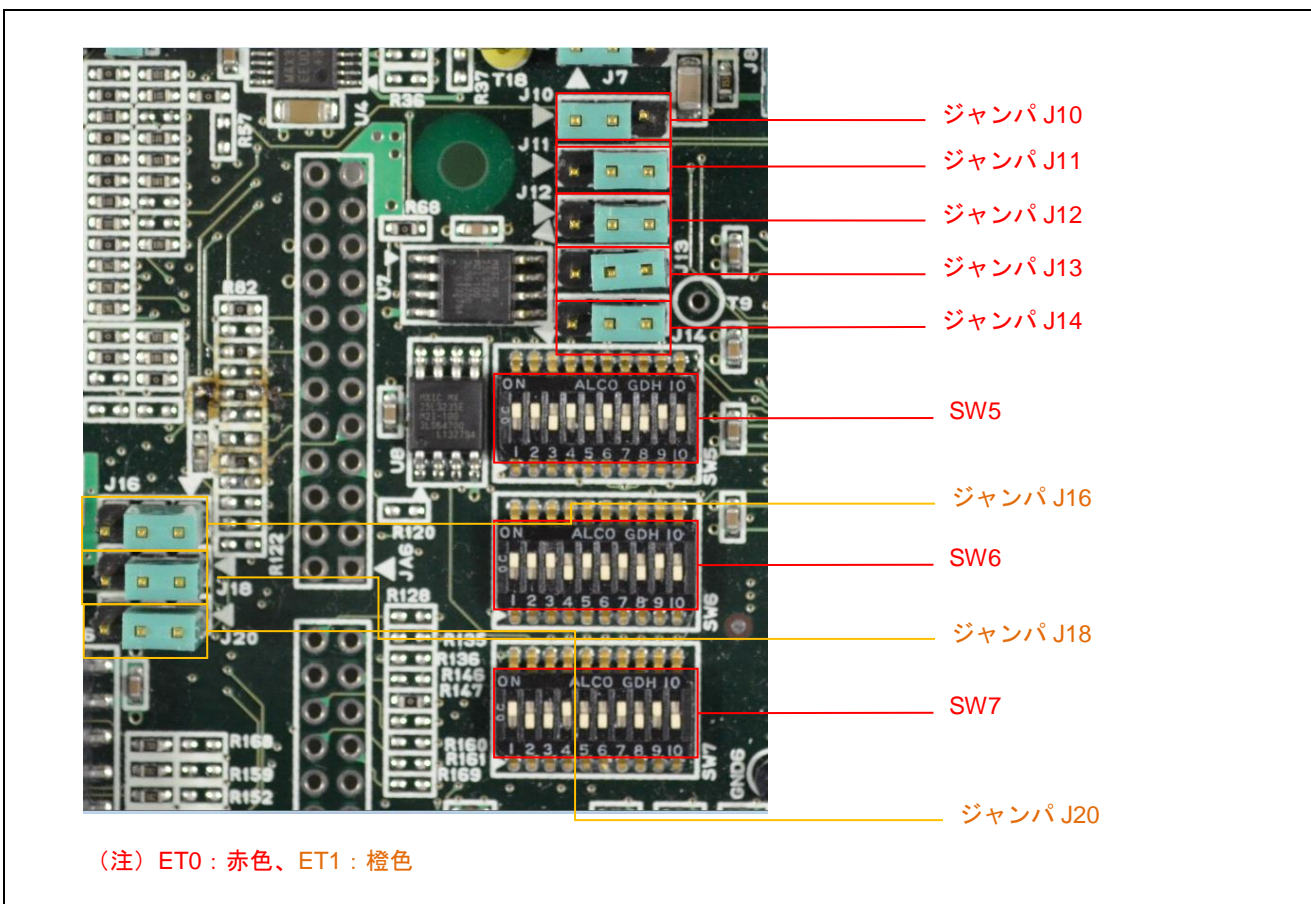
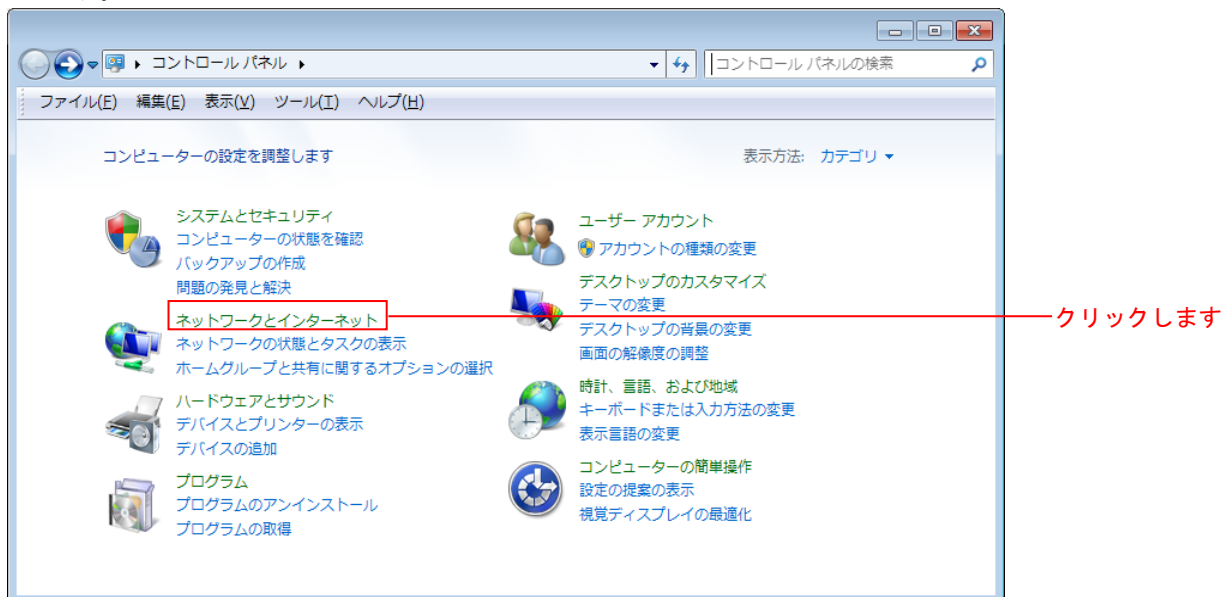


図 4-7 Renesas Starter Kit+ for RX64M/RX71M のジャンパ位置 【Ethernet コントローラ ET0/ET1】

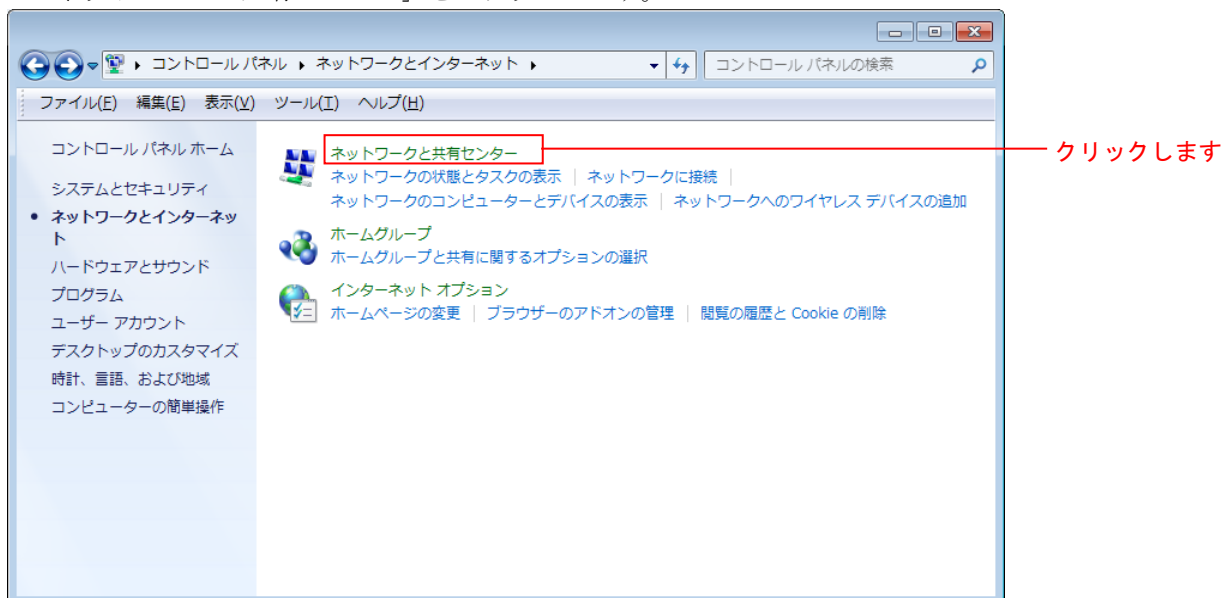
### 4.2.3 クライアント PC の設定

クライアント PC のネットワークを設定します。ここでは Windows7 の場合の例を示します。

1. クライアント PC の「コントロールパネル」を開き、「ネットワークとインターネット」をクリックします。



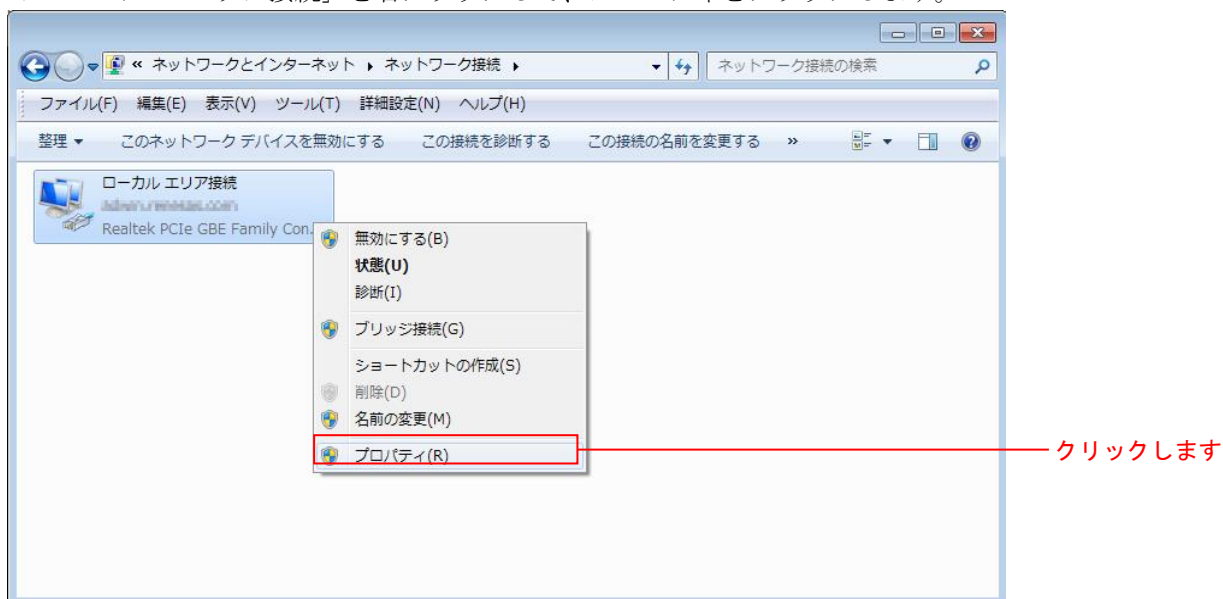
2. 「ネットワークと共有センター」をクリックします。



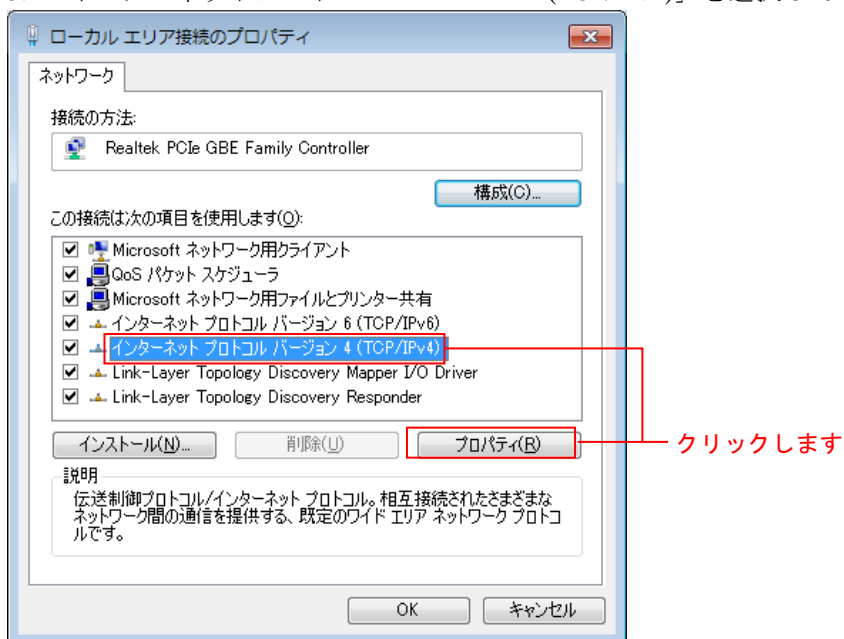
3. 「アダプタの設定の変更」をクリックします。



4. 「ローカル エリア接続」を右クリックして、プロパティをクリックします。

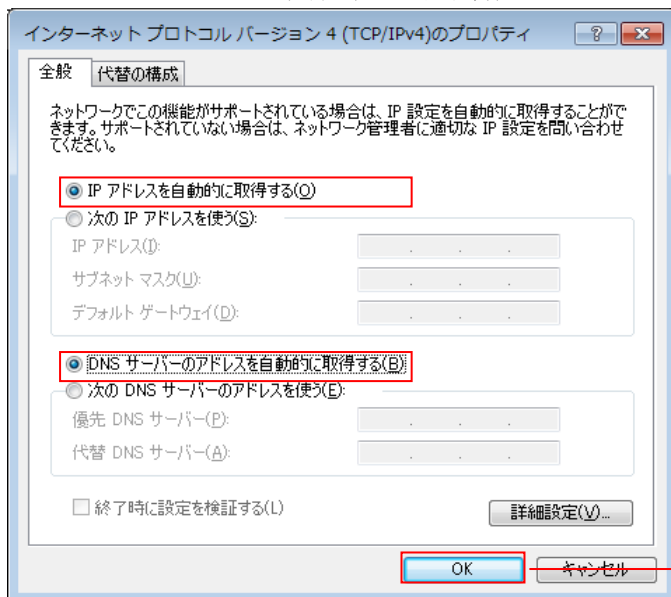


5. 「インターネットプロトコルバージョン 4 (TCP/IPv4)」 を選択して「プロパティ」をクリックします。



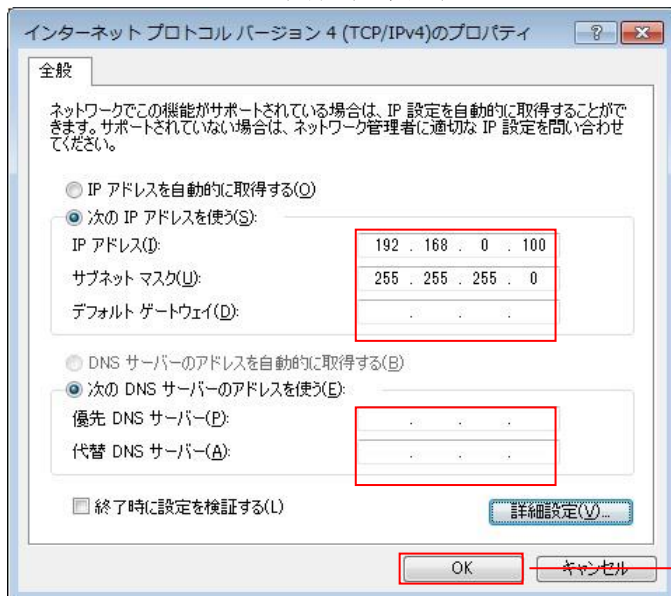
6. IP アドレス等の設定情報が表示されます。DHCP サーバの有無に合わせて、以下のように設定し「OK」をクリックします。

A. DHCP サーバがある場合 (IP 自動取得)



クリックします

B. DHCP サーバがない場合 (IP 固定)



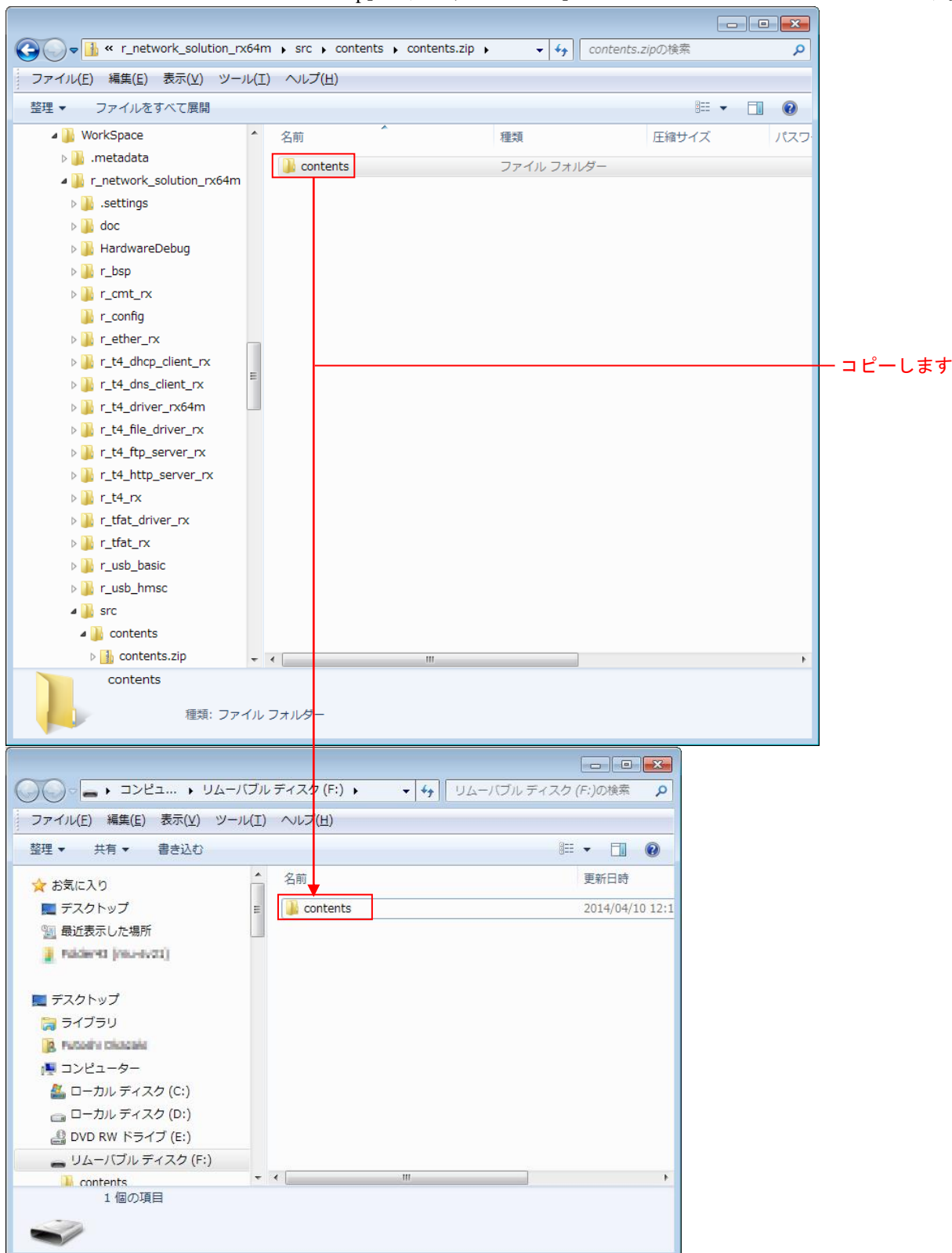
クリックします



## 4.2.4 USB メモリの準備

事前に USB メモリへ HTML コンテンツを格納します。

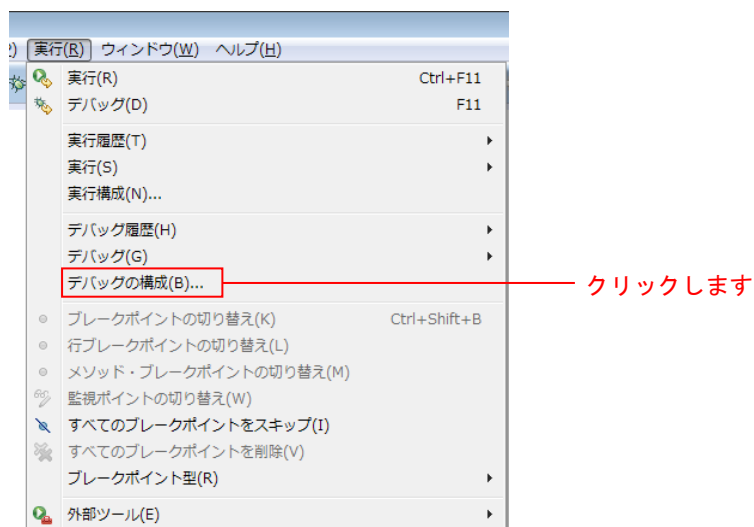
1. プロジェクト内の「src」フォルダを開き、その中にある「contents」フォルダを開きます。「contents」フォルダに入っている「contents.zip」を開き、「contents」フォルダを USB メモリにコピーします。



### 4.3 プロジェクトのデバッグ

以下の手順に従い、プロジェクトのデバッグを開始します。

1. 開発 PC と E1 エミュレータを USB ケーブルで接続します。
2. 評価ボード (Renesas Starter Kit+ for RX64M 又は Renesas Starter Kit+ for RX71M) にアダプタを接続し、電源を入れます。
3. e<sup>2</sup> studio の「実行」メニューの「デバッグ構成」をクリックします。



4. 「Renesas GDB Hardware Debugging」の「r\_network\_solution\_rx64m HardwareDebug 又は r\_network\_solution\_rx71m HardwareDebug」をクリックし、「デバッグ」をクリックします。  
「Debugger」タブをクリックし、「Connection Setting」タブをクリックします。  
「EXTAL 周波数」を「24.0000」に修正し、「エミュレータから電源を供給する」を「いいえ（注1）」に変更します。  
完了したら「デバッグ(D)」をクリックします。

注1：外部電源を使用する場合の設定です。エミュレータから電源を供給する場合は「はい」を選択してください。

The screenshot shows the 'Debug Configuration' dialog box with the following settings:

- 名前(N): r\_network\_solution\_rx64m HardwareDebug
- メイン: Debugger
- Startup: Startup
- ソース: ソース
- 共通(C): 共通(C)
- debug hardware: E1 (RX)
- Target Device: R5F564ML
- GDB Settings:
  - クロック:
    - メイン・クロック・ソース: EXTAL
    - EXTAL 周波数[MHz]: 24.0000
    - 内部フラッシュメモリー書き換え時にクロック・ソ: はい
  - ターゲット・ボードとの接続:
    - エミュレータ: (Auto)
    - 接続タイプ: JTag
    - JTag クロック周波数[MHz]: 16.5
    - Fine ボーレート[Mbps]: 2.00
    - ホット・プラグ: いいえ
  - 電源:
    - エミュレータから電源を供給する (MAX 200mA): いいえ
    - 供給電圧: 3.3V
  - CPU 動作モード:
    - レジスタ設定: シングルチップ

Red boxes and arrows indicate the following actions:

- Click the 'Debugger' button.
- Click the 'Connection Settings' tab.
- Change 'EXTAL 周波数[MHz]' to '24.0000'.
- Change 'エミュレータから電源を供給する (MAX 200mA)' to 'いいえ'.
- Click the 'デバッグ(D)' button.

以下のメッセージが表示されたら、「はい」をクリックします。

The screenshot shows a confirmation dialog box with the following text:

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・デバッグをサポートするために設計されています。これには、デバッグ・スタック、変数、およびブレイクポイント管理を表示するビューが組み込まれています。

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(B)

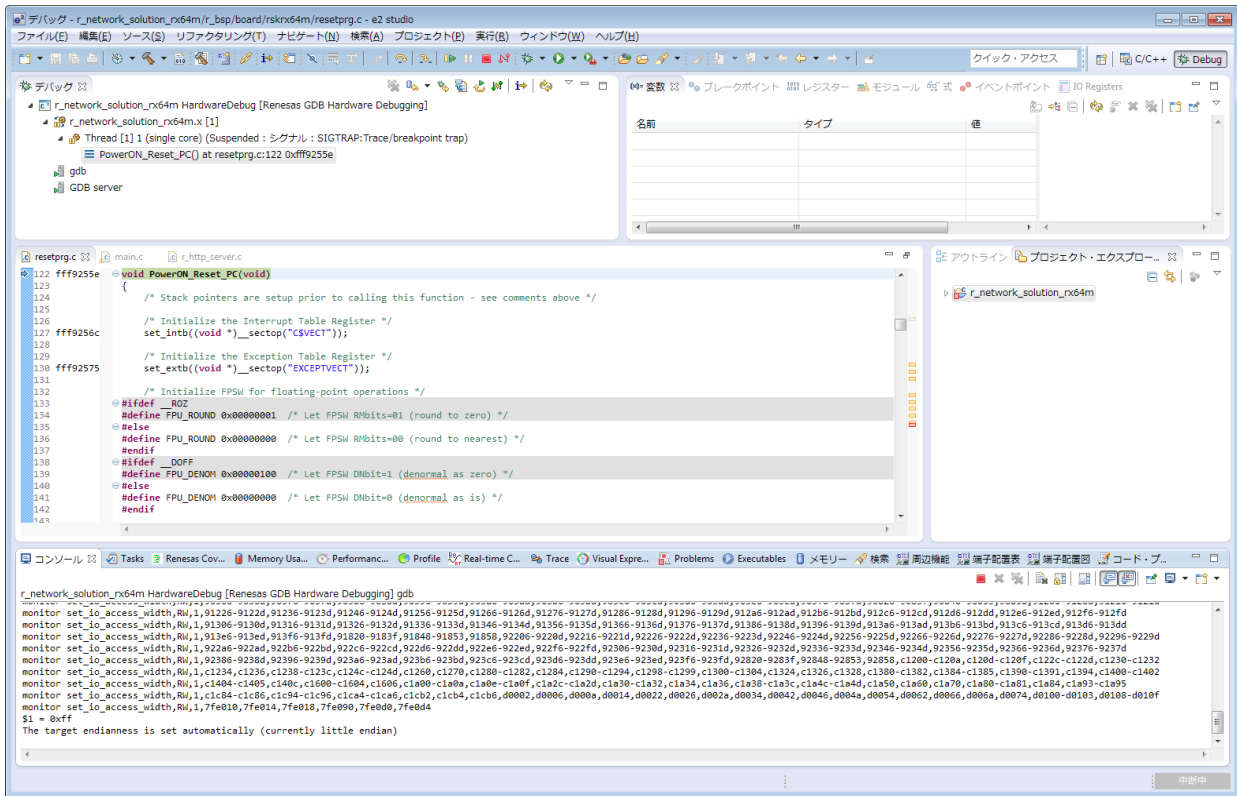
はい(Y)    いいえ(N)

Red boxes and arrows indicate the following action:

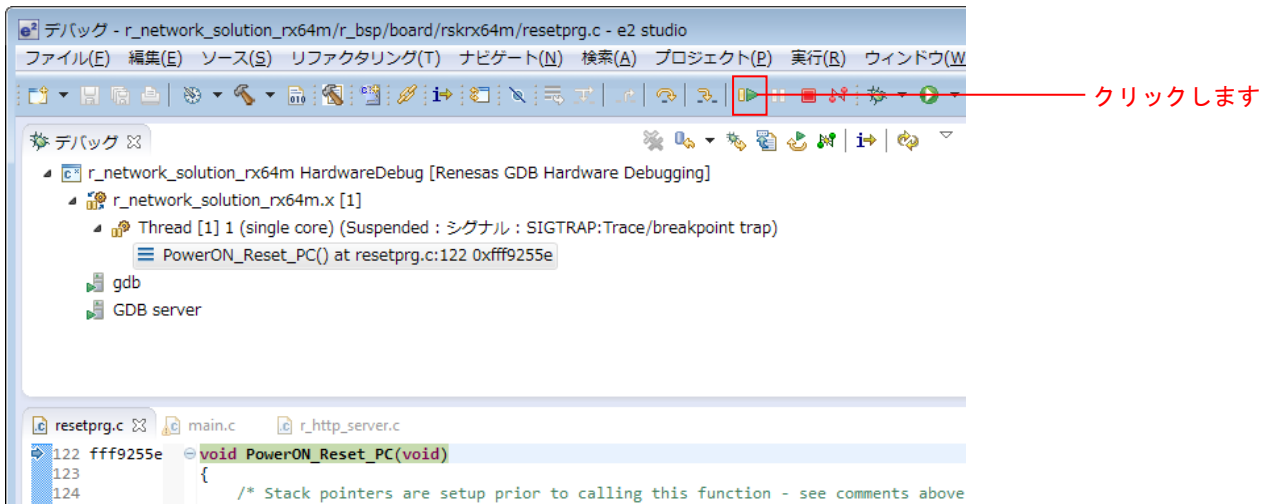
- Click the 'はい(Y)' button.



ロードモジュールのダウンロードが完了すると、「デバッグ」パースペクティブが開きます。



5. ツールバーの「再開」をクリックします。プログラムが実行され、main 関数の先頭でブレークします。



6. main 関数の先頭でブレークした後に、もう一度ツールバーの「再開」をクリックします。

## 4.3.1 HTTP サーバ機能の確認

1. クライアント PC で Web ブラウザを起動し、LAN ケーブルを接続したポートに合わせて、以下のアドレスを入力します。

Ethernet ポート番号	DHCP サーバ	HTTP サーバアドレス
0	なし	http://192.168.0.3 (コンフィギュレーションで変更可能)
	あり	DHCP サーバから取得
1	未対応	http://192.168.0.10

【注】 HTTP サーバアドレスは、コンフィギュレーションで変更可能です。

DHCP サーバから取得した IP アドレスを確認する手順を以下に示します。

デバッガ接続後、src/main.c を開きます。

関数 main()内の 159 行目にある関数 nop()にブレークポイントを設定します。

```

150
151 ffc024a8  if (!r_dhcp_open(&dhcp, (unsigned char*)tcpudp_work, &myethaddr[0][0]))
152     {
153 ffc024c5      set_tcpudp_env(&dhcp);
154     }
155 ffc024c8  CloseTimer();
156
157 ffc024cc  if (NULL != &dhcp)
158     {
159 ffc024d0      nop();
160     }
161
162     /* Get the size of the work area used by the T4 (RAM size). */
163 ffc024d1  ramsize = tcpudp_get_ramsize();
164 ffc024d5  if (ramsize > (sizeof(tcpudp_work)))
165     {
166     /* Then reserve as much memory array for the work area as the size
167        indicated by the returned value. */
168 ffc024db      for( ;; );
169     }
170
171     /* Initialize the TCP/IP */

```

プログラムを実行し、設定したブレークポイントで停止したのを確認後、変数 dhcp をウォッチウィンドウで確認します。

- 変数 dhcp の各メンバーに設定される情報

ipaddr : IP アドレス

ipaddr : IP アドレス

maskaddr : サブネットマスク

gwaddr : ゲートウェイアドレス

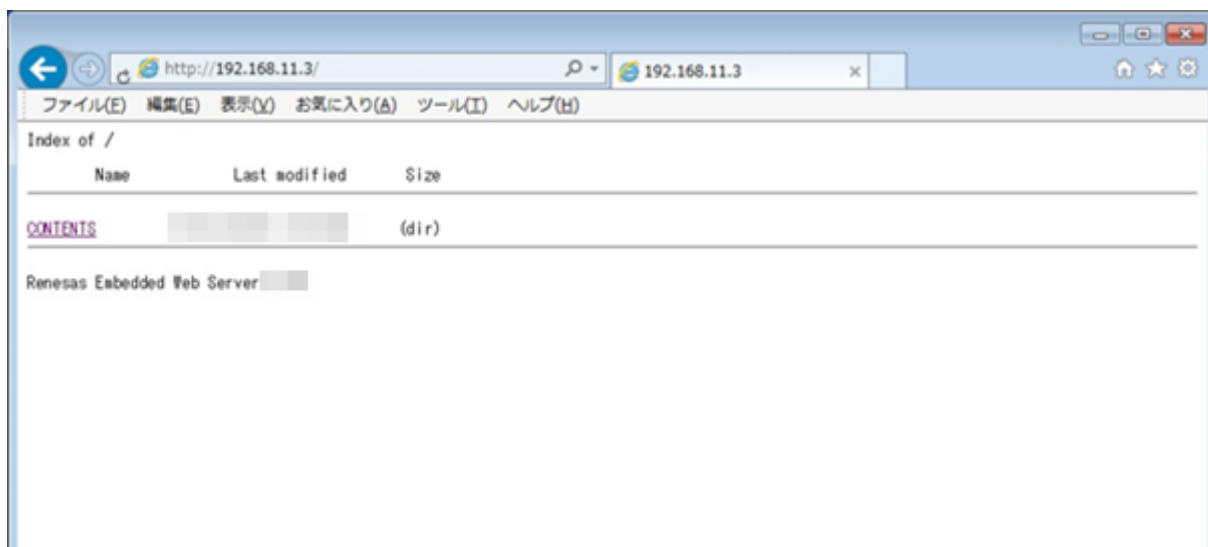
dnsaddr : Primary DNS サーバの IP アドレス

dnsaddr2 : Secondary DNS サーバの IP アドレス

名前	タイプ	値
dhcp	DHCP	{...}
ipaddr	uint8_t [4]	0x2802
(x)= ipaddr[0]	uint8_t	192 'ﾀ'
(x)= ipaddr[1]	uint8_t	168 'ｲ'
(x)= ipaddr[2]	uint8_t	0 '零'
(x)= ipaddr[3]	uint8_t	3 '三'
maskaddr	uint8_t [4]	0x2806
gwaddr	uint8_t [4]	0x280a
dnsaddr	uint8_t [4]	0x280e
dnsaddr2	uint8_t [4]	0x2812
domain	char [20]	0x2816
macaddr	uint8_t [6]	0x282a

Web ブラウザ上に、USB メモリ内のルートディレクトリにあるファイルの一覧が表示されます。

Name はファイル名を表し、Last modified は最終更新日を表し、Size はディレクトリである場合には(dir)と表し、Byte でファイルサイズを表します。Parent Directory をクリックすれば一つ上のディレクトリに移動します。



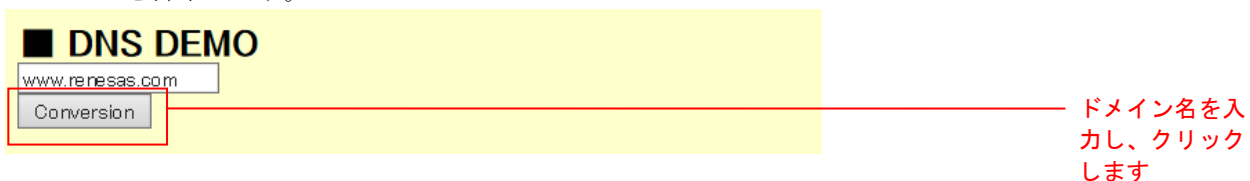
- 「CONTENTS」をクリックし、その中にある「DEMO.HTM」をクリックすると以下のような画面が表示されます。



- CGI 機能を使った LED 制御を確認します。  
「LEDx switch」ボタンを押すと、ボード上の LED を制御(点灯/消灯)することができます。



- CGI 機能を使った DNS クライアント機能を確認します。  
「■DNS DEMO」のテキストボックスに任意のドメイン名(例: www.renesas.com)を入力し、Conversion ボタンを押下します。



以下のような IP アドレス確認画面が表示されます。

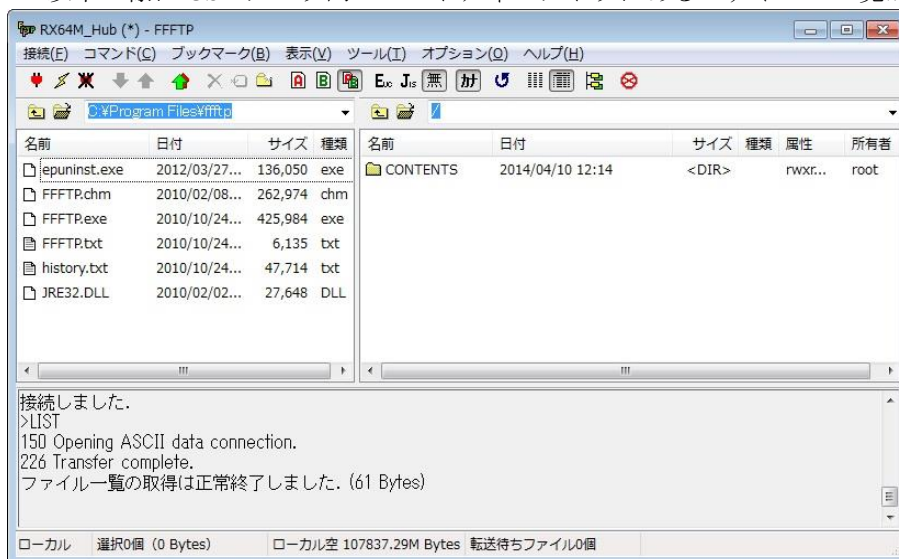


## 4.3.2 FTP サーバ機能の確認

1. クライアント PC で FTP クライアントソフト (FFFTP 等) を起動します。
2. 「r\_t4\_ftp\_server\_rx\_config.h」に設定されているユーザ名とパスワードを使用し、LAN ケーブルを接続したポートに合わせて以下の IP アドレスに接続します。

Ethernet ポート番号	DHCP サーバ	FTP サーバアドレス
0	なし	192.168.0.3 (コンフィギュレーションで変更可能)
	あり	DHCP サーバから取得
1	未対応	未対応

以下の様に USB メモリ内のルートディレクトリにあるファイルの一覧が表示されます。



## 5. ネットワークミドルウェア仕様

本アプリケーションで使用している DHCP クライアント、DNS クライアント、HTTP サーバ、FTP サーバの仕様に関しては、各 FIT モジュールに付属しているドキュメントを参照してください。

## 6. メインプログラム仕様

### 6.1 ファイル構成

メインプログラムはプロジェクトに同梱しています。メインプログラムのソースファイルは src フォルダに入っています。

メインプログラムのファイル一覧を以下に示します。

表 6-1 メインプログラムファイル一覧

フォルダ名	ファイル名	内容
src	main.c	メインソースファイル
	led.c	LED 初期化処理ソースファイル
	led.h	LED 初期化処理ヘッダファイル
	r_http_server_cgi_sample.c	CGI サンプルソースファイル
	r_sys_time.c	HTTP サーバ用システムタイマソースファイル
	r_sys_time.h	HTTP サーバ用システムタイマヘッダファイル
	r_usb_hmsc_api.c	USB ドライバ呼び出し処理ソースファイル
	r_usb_hmsc_api.h	USB ドライバ呼び出し処理ヘッダファイル
src¥contents	contents.zip	USB メモりに格納する HTML コンテンツの ZIP ファイル

## 6.2 モジュール一覧

メインプログラムのモジュール一覧を以下に示します。

表 6-2 メインプログラムモジュール一覧

ファイル名	モジュール名	内容
main.c	main	メインプログラムのメイン処理。 各 FIT モジュールの初期化処理を呼び出し、DNS クライアント、HTTP サーバ、FTP サーバ、USB ドライバ及び Ethernet ドライバのメイン処理を駆動する（無限ループによる周期起動）。
	set_tcpudp_env	DHCP サーバから取得した IP アドレスなどのネットワーク情報を T4 の環境変数に設定する。
	usb_cpu_FunctionUSB0IP	USB ch0 のピン設定及びポート初期化処理
	usb_cpu_FunctionUSB1IP	USB ch1 のピン設定及びポート初期化処理
r_usb_hmsc_api.c	usb_cstd_IdleTaskStart	Low Power Mode で使用するアイドルタスクを起動する。
	usb_cstd_IdleTask	Low Power Mode で使用するアイドルタスク ホスト動作では処理なし。
	usb_hmsc_task_start	HMSC ドライバ起動処理 USB IP の初期化やクラスドライバの登録を行う。
	usb_apl_task_switch	非 OS 環境用の各 USB ドライバタスクのスケジューリング処理
	usb_hapl_task_start	HMSC ドライバアプリケーションタスクを起動する。
	usb_hmsc_DummyFunction	HMSC ドライバ用ダミー関数
	usb_hmsc_DriveOpen	HMSC ドライバオープン処理。
	usb_hapl_registration	HMSC ドライバを登録する。
	usb_hmsc_apl_init	HMSC ドライバアプリケーションタスクの内部変数の初期化を行う。
	usb_hmsc_StrgCommandRes ult	R_usb_hmsc_StrgDriveSearch() のコールバック処理
	usb_hmsc_SampleAplTask	HMSC ドライバアプリケーションタスク処理 USB メモリを検出し、ファイルシステムをマウントする。
led.c	led_init	LED 初期化
r_http_server_cgi_sample.c	—	HTTP サーバモジュールに付属のドキュメント「RX ファミリ 組み込み用 TCP/IP M3S-T4-Tiny を用いた Web サーバモジュール Firmware Integration Technology/R20AN0075JJ」を参照してください。
r_sys_time.c	get_sys_time	HTTP サーバモジュールに付属のドキュメント「RX ファミリ 組み込み用 TCP/IP M3S-T4-Tiny を用いた Web サーバモジュール Firmware Integration Technology/R20AN0075JJ」を参照してください。
	start_system_time	HTTP サーバで使用するタイマの起動
	update_sys_time	HTTP サーバで使用するタイマの割り込みハンドラ
	stop_system_time	HTTP サーバで使用するタイマの停止



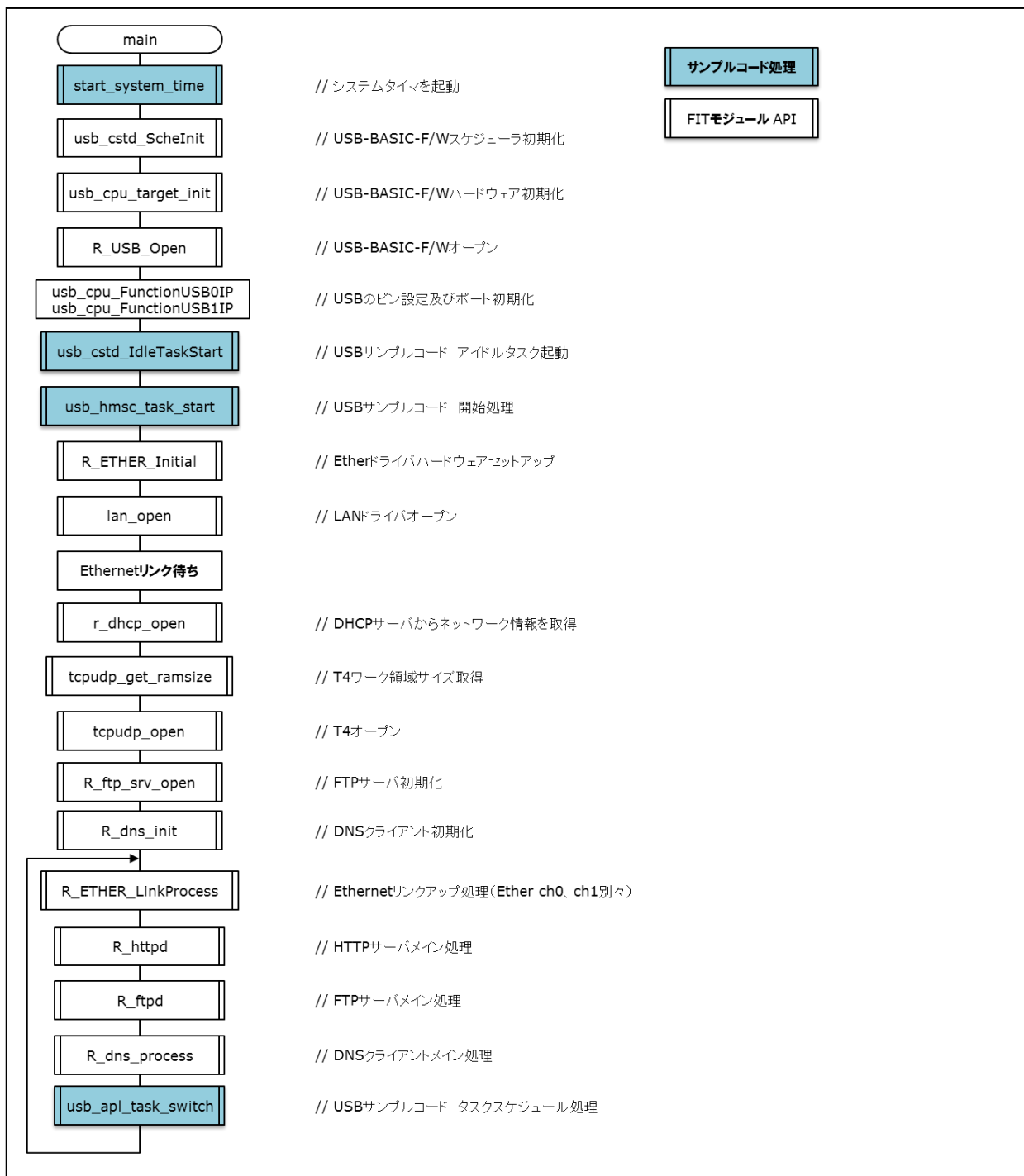
### 6.3 処理フロー

メインプログラムの各モジュールの処理フローを以下に示します。

#### (1) main()

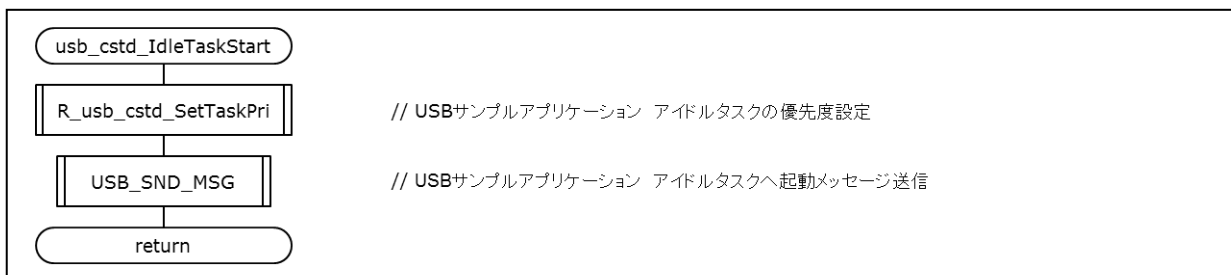
ボードサポートパッケージ (BSP モジュール) のスタートアップルーチンから最初に呼ばれるメイン関数です。

各ドライバと T4 の初期化を行い、無限ループ処理内で Ethernet ドライバのリンクアップと、DNS クライアント、HTTP サーバ、FTP サーバのメイン処理と USB ドライバのスケジュール処理を定期的呼び出します。



(2) usb\_cstd\_IdleTaskStart

USB ドライバ処理のアイドルタスクを起動します。



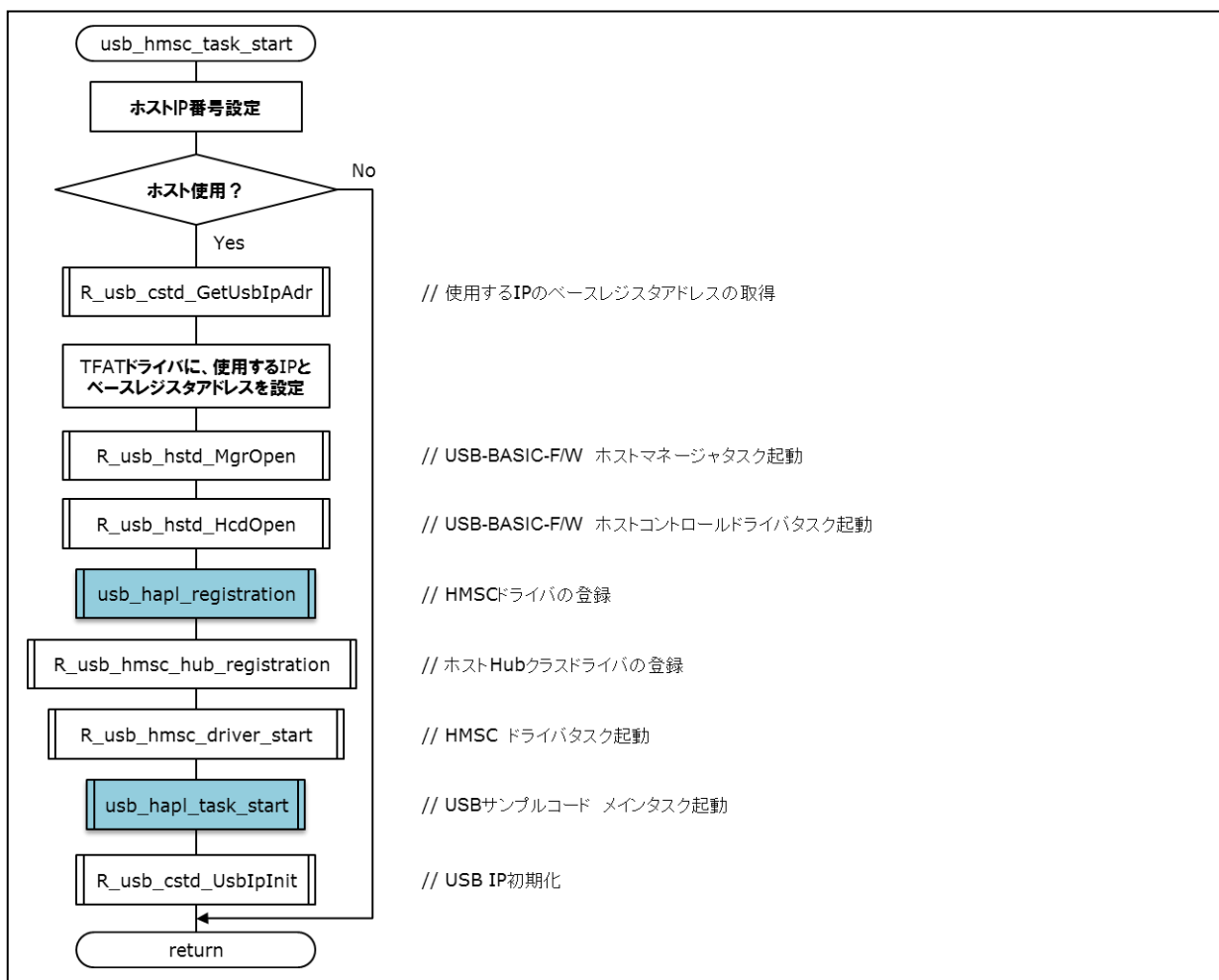
(3) usb\_cstd\_IdleTask

USB ドライバ処理のアイドルタスクです。



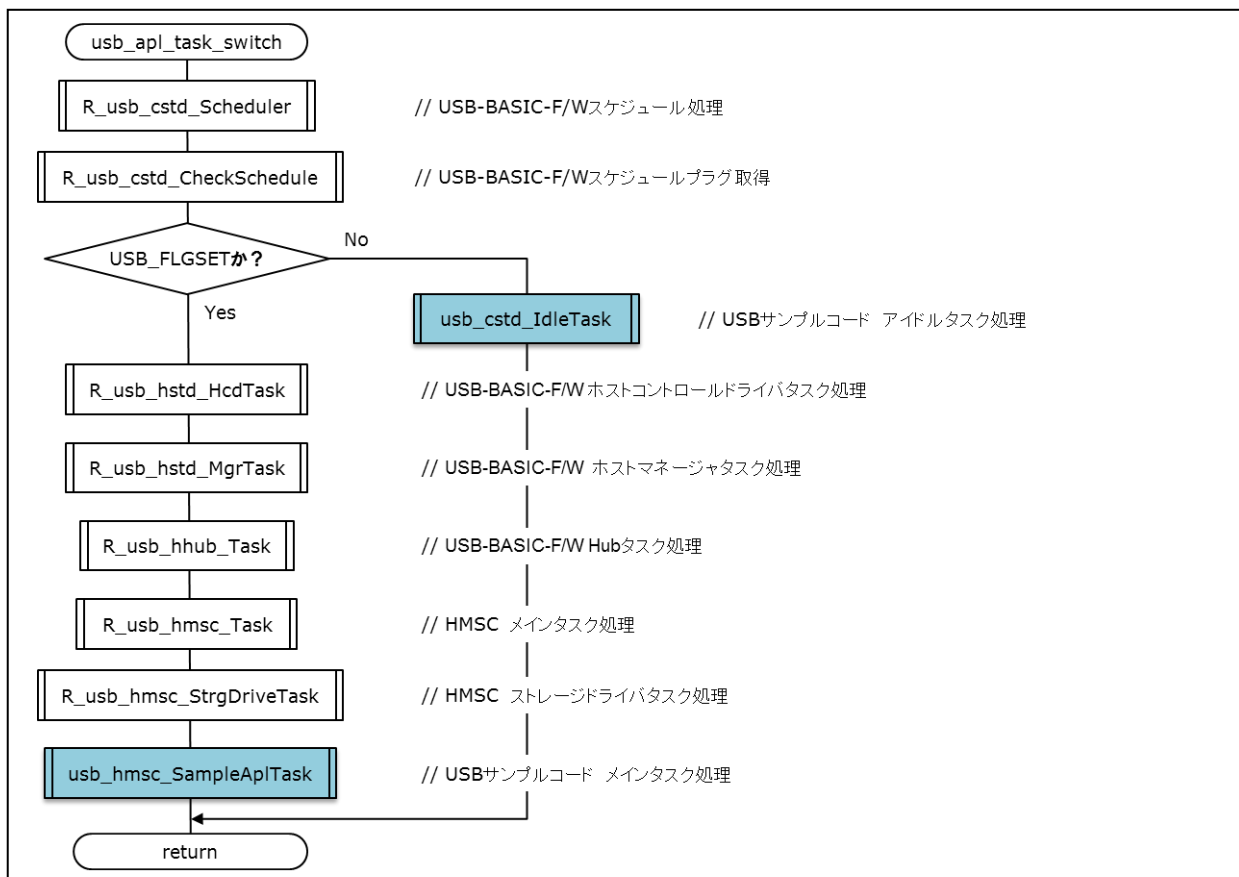
## (4) usb\_hmsc\_task\_start

USB ドライバ内の各タスクの起動と、クラスドライバの登録を行い、USB メモリのマウント処理用タスクを起動します。



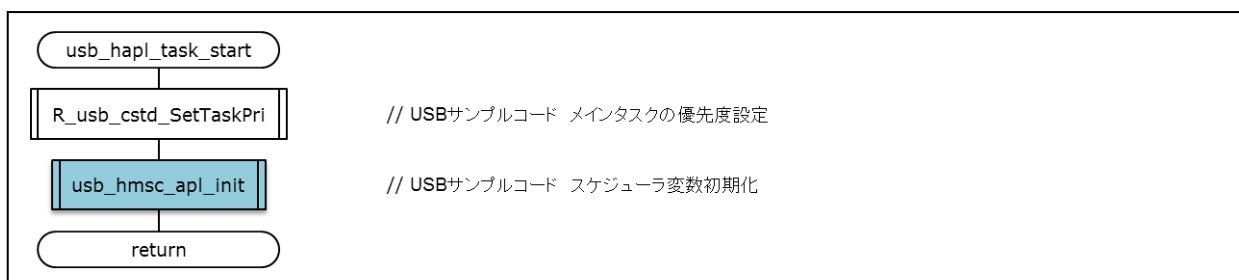
(5) usb\_apl\_task\_switch

USB ドライバのスケジュール処理です。



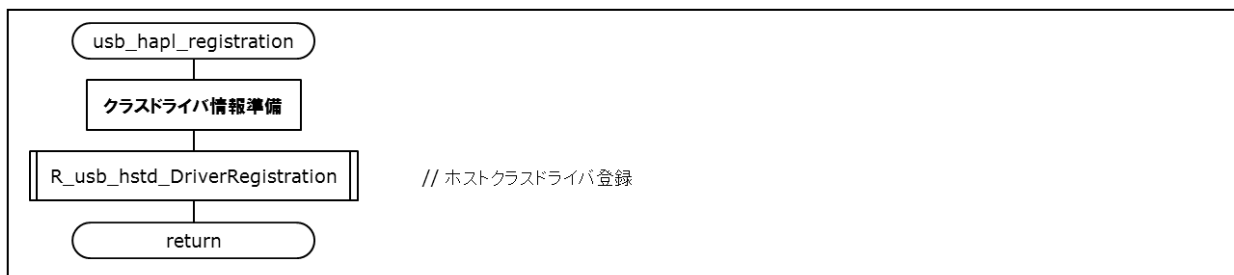
(6) usb\_hapl\_task\_start

USB メモリのマウント処理用タスクの初期化を行います。



(7) usb\_hapl\_registration

クラスドライバの登録処理です。



(8) usb\_hmsc\_apl\_init

USB メモリのマウント処理用タスクのシーケンス処理変数を初期化します。



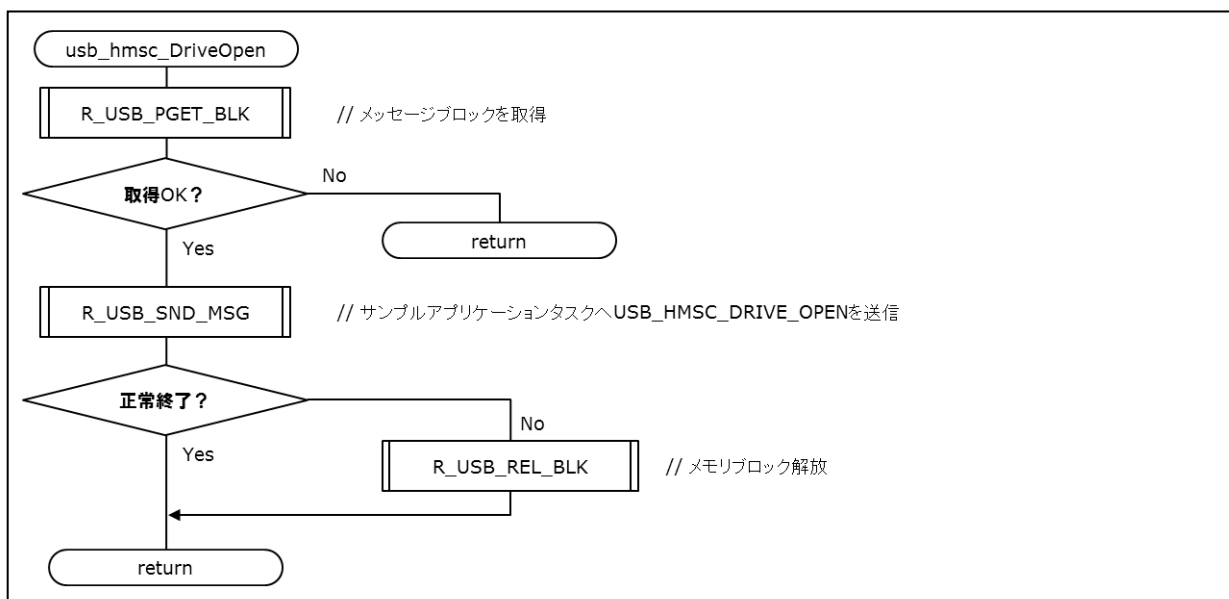
(9) usb\_hmsc\_DummyFunction

クラスドライバ登録時に指定する suspend と resume 用のダミー関数です。



(10) usb\_hmsc\_DriveOpen

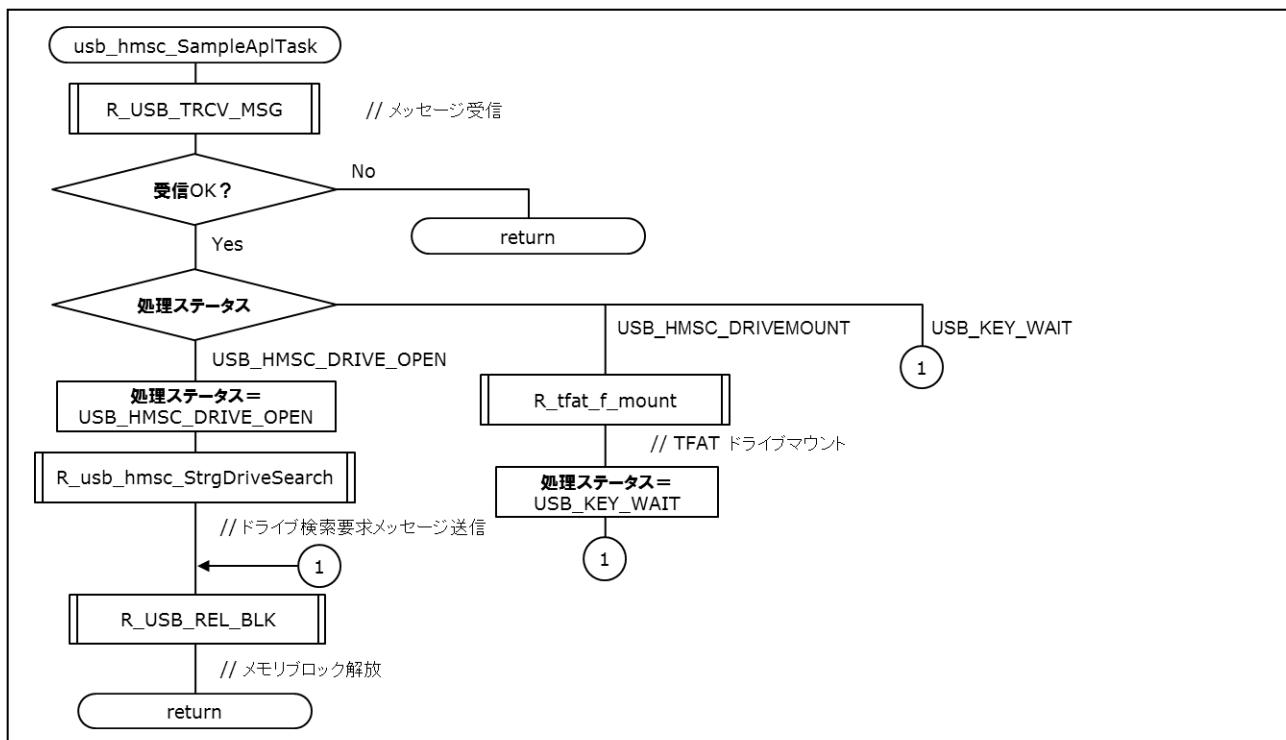
USB メモリが挿入されたときに、USB ドライバから呼び出されるコールバック関数です。サンプルアプリケーションタスクに対し、USB\_HMSC\_DRIVE\_OPEN メッセージを送信します。



(11) usb\_hmsc\_SampleApiTask

サンプルアプリケーションタスク処理です。usb\_hmsc\_DriveOpen 関数から送信された USB\_HMSC\_DRIVE\_OPEN メッセージを受信し、マウント可能なドライブの検索を行います。

また、usb\_hmsc\_StrgCommandResult 関数から送信された USB\_HMSC\_DRIVEMOUNT メッセージを受信し、ファイルシステムへのマウントを行います。



## (12) led\_init

Renesas Starter Kit+ for RX64M or Renesas Starter Kit+ for RX71M 上の LED を使用するための初期化処理を行います。



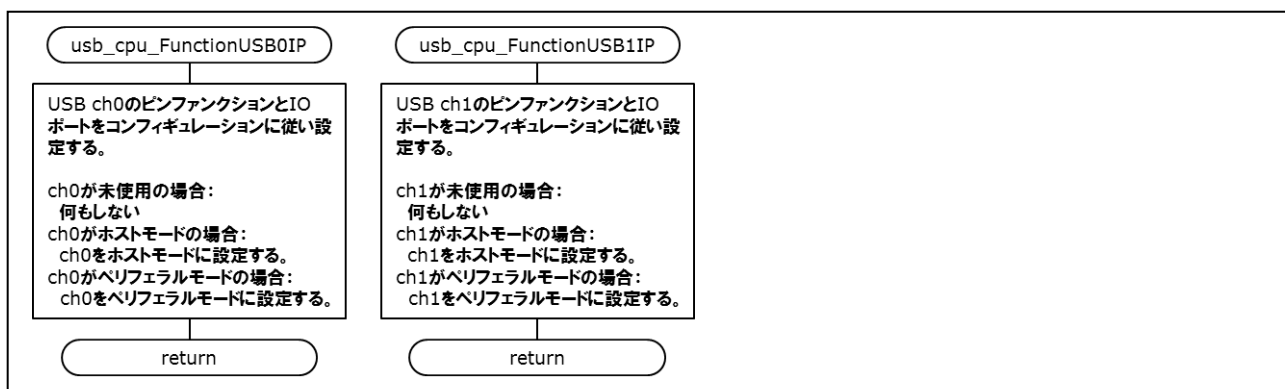
## (13) set\_tcpudp\_env

DHCP サーバから取得した IP アドレスなどのネットワーク情報を T4 の環境変数 `tcpudp_env` に設定します。



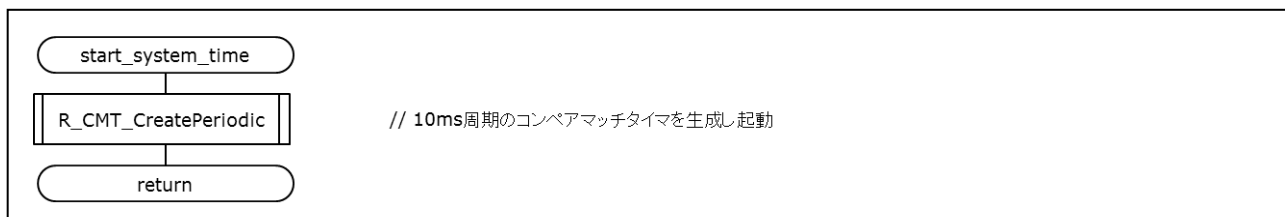
## (14) usb\_cpu\_FunctionUSB0IP、usb\_cpu\_FunctionUSB1IP

USB の ch0 と ch1 を使用するためのピンファンクションと IO ポートの初期化処理を行います。



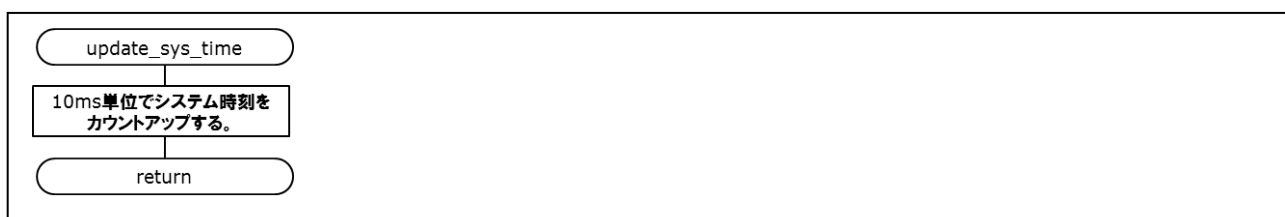
## (15) start\_system\_time

HTTP サーバのシステム時刻用のコンペアマッチタイマ（10ms 周期）を起動します。



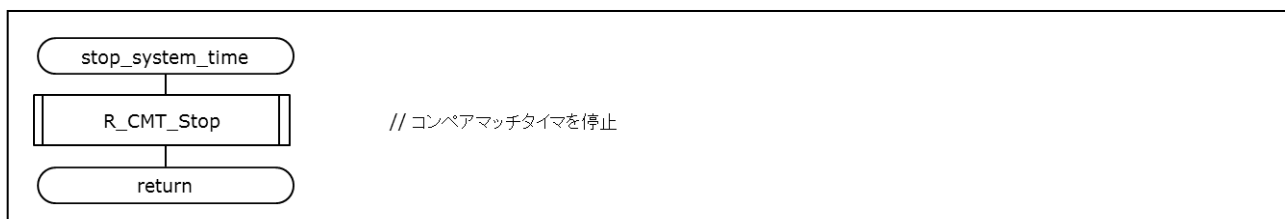
## (16) update\_sys\_time

HTTP サーバのシステム時刻をカウントアップする割り込みハンドラです。



## (17) stop\_system\_time

HTTP サーバのシステム時刻用のコンペアマッチタイマを停止します。





## 7. ユーザ定義関数について

ユーザ定義関数は、ユーザのシステム環境に合わせて、ユーザが記述する必要があるコードです。FIT モジュールの中には、ユーザ定義関数を必要とするものがあります。

本パッケージでは、以下のユーザ定義関数のサンプルを収録しています。ユーザ定義関数の仕様については、対応する FIT モジュールのマニュアル等を参照してください。

表 7-1 ユーザ定義関数一覧

ユーザ定義関数	ファイル名	FIT モジュール名	ドキュメント名/型番
システムタイミンタフェース	r_sys_time.c	r_t4_http_server_rx	RX ファミリ 組み込み用 TCP/IP M3S-T4-Tiny を用いた Web サーバモジュール Firmware Integration Technology/R20AN0075JJ 「4.2 システムタイマ関連」

## 8. 補足

### 8.1 USB ドライバの制限事項

USB の ch0 と ch1 の両方をホストモードに設定した場合、ch0 側でしか USB メモリを認識しません。ch1 側をホストモードで使いたい場合は、ch0 側を未使用かペリフェラルモードに設定してください。設定例は「3.4.1(3) USB ドライバの設定変更」を参照してください。

### 8.2 本アプリケーションの制限事項

プログラム動作後に USB メモリを抜いて、再度挿入した場合、USB メモリが認識されません。プログラムを再起動してください。

### 8.3 DHCP クライアント及び FTP サーバの制限事項

DHCP クライアント及び FTP サーバは Ethernet の ch0 側でのみ動作します。

### 8.4 無償評価版の「RX ファミリ用 C/C++コンパイラパッケージ」を利用する場合の注意事項

無償評価版の「RX ファミリ用 C/C++コンパイラパッケージ」には、使用期限と使用制限があります。使用期限が過ぎた場合、使用制限によりロードモジュールが正しく生成されなくなる場合があります。

詳しくは、ルネサスのホームページにある、評価版ソフトウェアツールのページを参照してください。

URL : [http://japan.renesas.com/products/tools/evaluation\\_software/index.jsp](http://japan.renesas.com/products/tools/evaluation_software/index.jsp)

## ホームページとサポート窓口

ルネサス エレクトロニクスホームページ

<http://japan.renesas.com/>

お問い合わせ先

<http://japan.renesas.com/contact/>

すべての商標および登録商標は、それぞれの所有者に帰属します。

## 改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2014.09.01	－	初版発行 アプリケーションノート番号：R01AN2153JJ0100
1.01	未発行	－	R01AN2153JJ0100 をベースに、DHCP クライアント、DNS クライアント、FTP サーバに対応
1.02	2015.10.30	－	R01AN2153JJ0100 をベースに RX71M を追加 アプリケーションノート番号を R01AN2608JJ0102 に変更
1.03	2016.02.29	1	使用した RX Driver Package 環境 にて、 ドキュメント番号の rev 番号を削除
		4	1.2 動作環境 にて、 ドキュメント番号の rev 番号を削除
		6	1.3 モジュール構成 表 1-3 モジュール一覧 にて、 「Rev.」 元は、「バージョン」であった。 「メインプログラム」 元は、「メインプログラム FIT モジュール」であった。 メインプログラムの Rev. 元は、1.02 であった。
		－	アプリケーションノート英語版の同梱と FIT モジュールの更新

## 製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

### 1. 未使用端子の処理

【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。

CMOS製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI周辺のノイズが印加され、LSI内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

### 2. 電源投入時の処置

【注意】電源投入時は、製品の状態は不定です。

電源投入時には、LSIの内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。

外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。

同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

### 3. リザーブアドレス（予約領域）のアクセス禁止

【注意】リザーブアドレス（予約領域）のアクセスを禁止します。

アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

### 4. クロックについて

【注意】リセット時は、クロックが安定した後、リセットを解除してください。

プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。

リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

### 5. 製品間の相違について

【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。

同じグループのマイコンでも型名が違っていると、内部ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

## ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
2. 本資料に記載されている情報は、正確を期すため慎重に作成したのですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。  
標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、  
家電、工作機械、パーソナル機器、産業用ロボット等  
高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、  
防災・防犯装置、各種安全装置等  
当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図しない用途に当社製品を使用したことによりお客様または第三者に損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。お客様がかかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途に使用しないでください。当社製品または技術を輸出する場合は、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

注1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社その総株主の議決権の過半数を直接または間接に保有する会社をいいます。

注2. 本資料において使用されている「当社製品」とは、注1において定義された当社の開発、製造製品をいいます。



ルネサス エレクトロニクス株式会社

■営業お問合せ窓口

<http://www.renesas.com>

※営業お問合せ窓口の住所は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス株式会社 〒135-0061 東京都江東区豊洲3-2-24（豊洲フォレストシア）

■技術的なお問合せおよび資料のご請求は下記へどうぞ。  
総合お問合せ窓口：<http://japan.renesas.com/contact/>