

開発環境構築ガイド

RM-240/RM-241

IEEE802.15.4 2.4GHz MAC通信モジュール

RF **LINK**

Ver2.2

目次

1. はじめに
2. 開発環境の準備
 - 2.1 開発に必要な環境
 - 2.2 開発環境構築の流れ
 - 2.3 方法1の環境構築例
 - 2.4 方法2の環境構築例
 - 2.5 ドライバのインストール方法
 - 2.6 シリアル通信ソフトの設定
3. 内蔵FlashROMへの書込み
 - 3.1 書込みに必要なもの
 - 3.2 方法1による書込み ~stm32w_flasherツールを使用する方法
 - 3.3 方法1による書込み ~stm32w_flasherツール + JTAG-ICEを使用する方法
 - 3.4 方法2による書込み ~J-LINK(JTAG-ICE)を使用する方法
4. 統合環境(IAR社・EWARM)を使用した開発方法
 - 4.1 統合開発環境構築までの流れ
 - 4.2 統合開発環境のインストール
 - 4.3 SimpleMACstdプログラムのダウンロード
 - 4.4 RM-240EVとの接続
 - 4.5 RM-24X/9XX_EV Rev.310との接続
 - 4.6 RM-24X/9XX_EV Rev.310 SW2 の設定方法
 - 4.7 RM-24X/9XX Rev.310 J5 の設定方法
 - 4.8 プロジェクトファイルの展開
 - 4.9 プロジェクトのビルド
 - 4.10 プロジェクトのビルド後の確認
 - 4.11 実行モジュールのRM-240へのダウンロード
 - 4.12 デバッグの開始
 - 4.13 プログラムの実行~シリアル通信ソフトによる確認
5. まとめ

1.はじめに

本書では、RM-240開発キット用サンプルソフトウェア(SimpleMACstd)を使用した開発環境の構築方法について説明します。

本書は、弊社で推奨するIAR社の統合環境、及びJ-Link(JTAG-ICE)を使用する事を前提に記述しています。

他メーカーのツールや、GNU環境による開発も可能ですが、ご提供するサンプルプログラムはIAR社のコンパイラに適合する記述になっていますので、

他メーカー、GNU等によるビルドをされる場合は、お客様による修正が必要です。

2.開発の準備

2.1 開発に必要な環境

RM-240/241の開発方法として、2つの方法がありますので、目的と用途に応じて選択して下さい。

(方法1) お客様ご自身で、RM-240/RM-241に書き込むF/Wの開発から着手される場合

(方法2) 弊社でご用意するサンプルソフトウェアを、そのままご使用になる場合

	必要な機材	説明	SDK標準セット	方法1	方法2
1	RM-24X/9XX_EV (開発ボード)	開発ボード本体 (対向通信用として最低2台必要)	●	●	●
2	RM-240/RM-241 通信モジュール	通信モジュール本体 (対向通信用として最低2台必要)	●	●	●
3	USBケーブル(mini/マイクロ)	RM-24X/9XX_EVとパソコンとの接続(最低2本) ※開発ボードのバージョンにより、使用するUSBケーブルの種類が異なります。	●	●	●
4	i-Jet 又は ST-Link	ICEによるトレースデバッグ、F/Wの書き込み	※オプション	●	
5	IAR統合開発環境 (EWARM)	コンパイル、及びデバッグのソフトウェア開発用	※オプション	●	
6	SimpleMACstd ソースコード	RM-240/RM-241用のサンプルプログラム	●	●	
7	シリアル通信ソフト	汎用のフリーソフト等 (teratermなど)	●	●	●

※「RM-240/241開発キット」の(方法1)で使用するJTAG-ICE、EWARM(統合開発環境)はオプションになります。
弊社からもご購入頂けます。

2.開発環境の準備

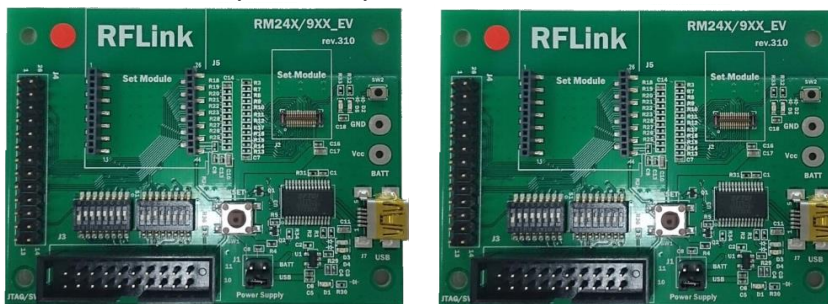
2.2 開発キット同梱内容

◆RM-240 / RM-241



モジュールは、内臓アンテナタイプと外部アンテナタイプの2種類あります。
開発キットをご注文の際、ご指定頂きます。

◆RM-24X/9XX_EV (開発ボード)



◆i-Jet (JTAG-ICE) ※オプション IAR社製品



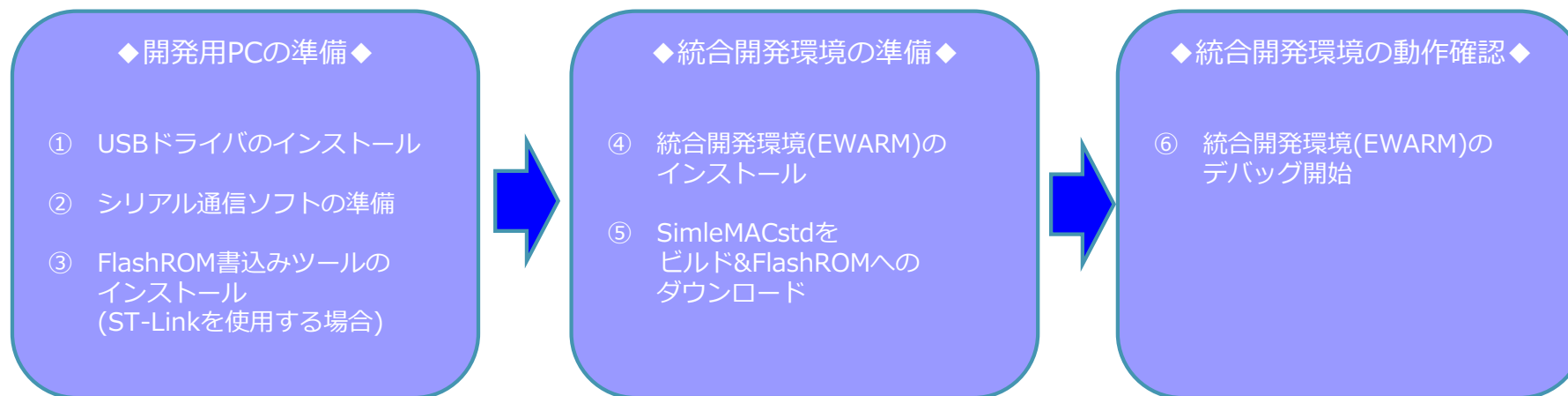
◆コンパイラ・デバッグ統合環境(EWARM) ※オプション IAR社製品



2. 開発環境の準備

2.3 開発環境構築の流れ

標準的な開発環境を構築するまでの流れを以下に説明します。



(方法2)による開発を行い場合は、④～⑥ は不要です。
コンフィグレーションを実行した後、直ぐに使用する事が可能です。

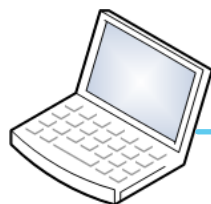
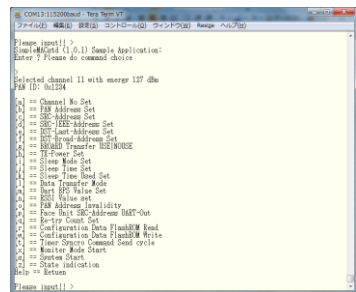
③は、i-jetを使用せず、STマイクロエレクトロニクス社専用のJTAツール(ST-Link)を使用する場合には必要になります。

※詳細は、「SimpleMACstd取り扱い説明書」を参照下さい。

2. 開発環境の準備

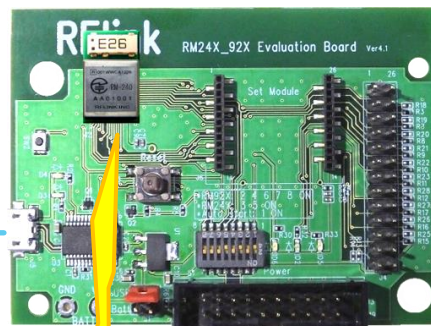
2.4 方法1の環境構築例

汎用シリアル通信ソフト(TeraTerm)



マイクロUSB

RM-24X/9XXEV

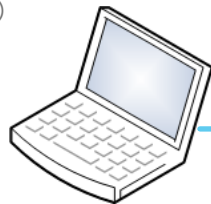
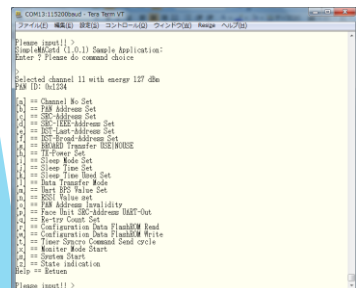


無線

RM-24X/9XXEV



汎用シリアル通信ソフト(TeraTerm)

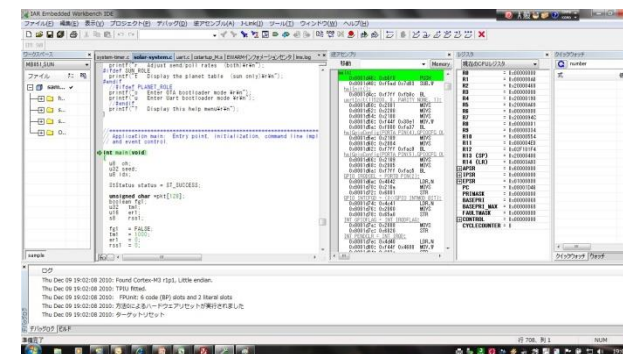


マイクロUSB

i-Jet (JTAG-ICE)



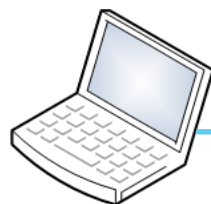
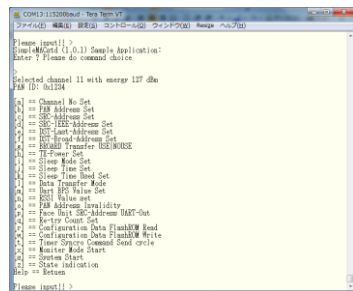
IAR社 統合環境ツール (Embedded Workbench)



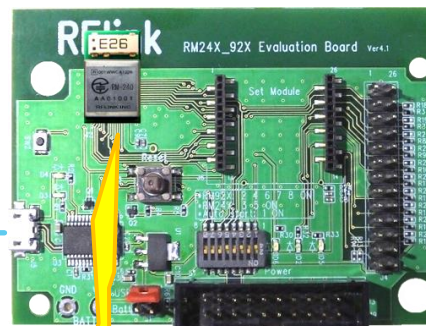
2.開発環境の準備

2.5 方法2の環境構築例

汎用シリアル通信ソフト(TeraTerm)



マイクロUSB

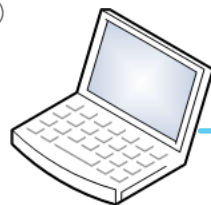
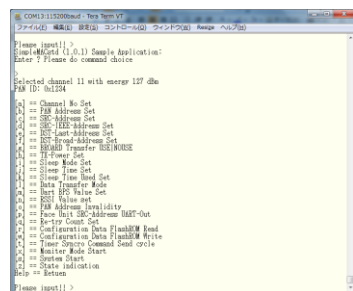


無線

RM-24X/9XXEV



汎用シリアル通信ソフト(TeraTerm)



マイクロUSB

2. 開発環境の準備

2.6 ドライバのインストール

RM-24X/9XX_EVを使用する為に、PCにFT232Cドライバのインストールを行います。

手順1 添付CD、又は弊社HPの「ドキュメントダウンロード」ページから、RM-205シリーズの「FTDI-USBドライバ(FT232)」をダウンロードします。
※この時USB機器はPCに挿入しないで下さい。

手順2 ZIPファイルを任意の場所に解凍します。(システムドライブ(通常はC:ドライブ)のルート上のフォルダを推奨します)

手順3 PCに、RM-24X/9XX_EVをPCとUSB接続します。

手順4 PCのOSのバージョンにより、FT232デバイスをサポートしている場合があります。
※自動インストールされた場合は、手順7に進みます。

手順5 コントロールパネル→デバイスマネージャーを開きます。

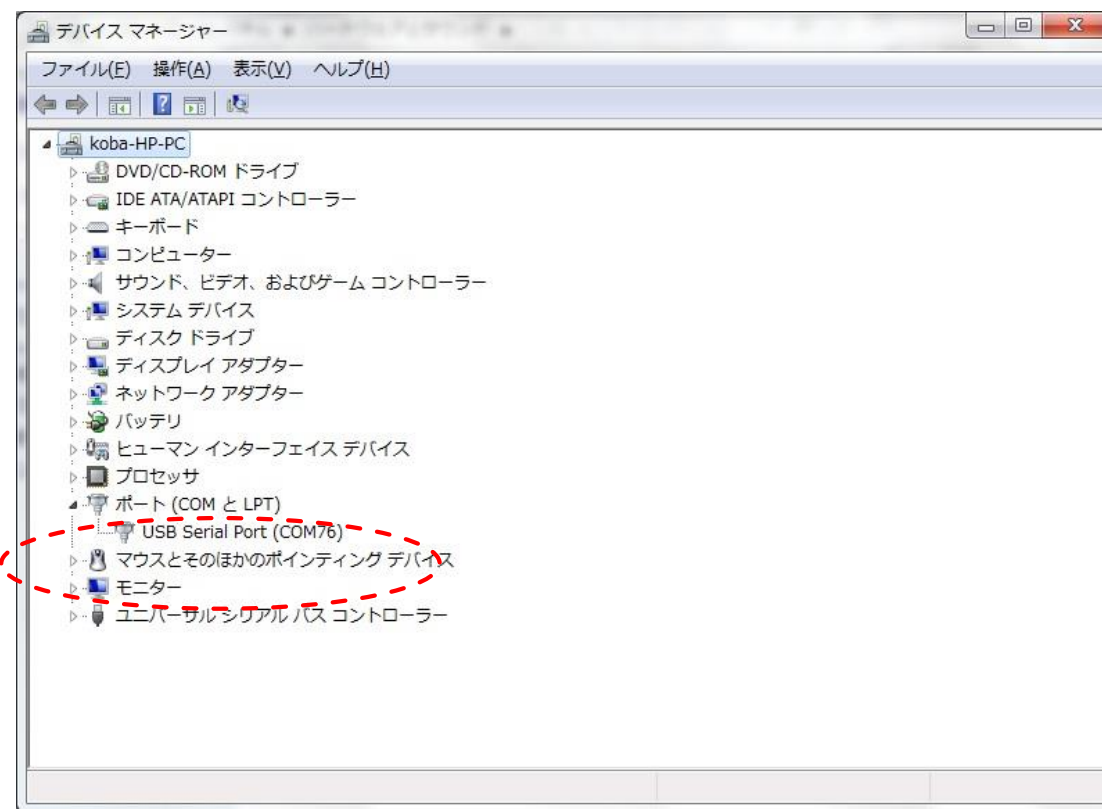
手順6 不明なデバイスとして認識されていた場合、手動設定で、手順2 で指定した場所を指定して、ドライバ設定を完了します。

手順7 正常にドライバ認識がされると、仮想COMポートとして認識されます ※次頁参照

2. 開発環境の準備

2.6 ドライバのインストール

正しくインストールが出来た状態のコントロールパネル表示

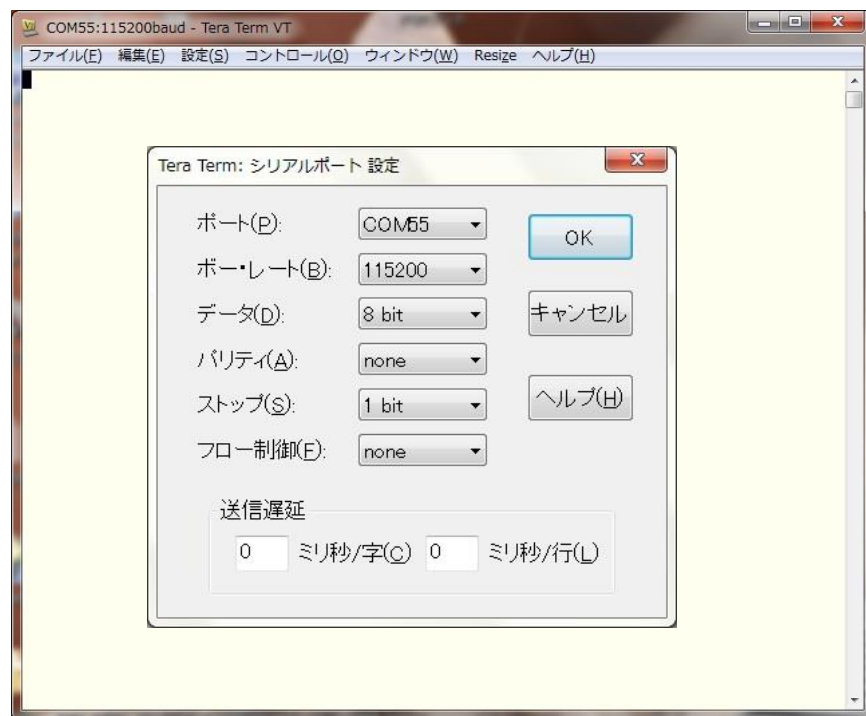


2.開発環境の準備

2.7 シリアル通信ソフトの設定

PCとRM-240/241とはシリアル通信でコンフィグレーションを行います。
通信ソフトウェアは、Windows標準ツール(ハイパーターミナル等)や、LinuxのminiCOMなどの他、フリーソフトなどで前章で設定したCOMポート番号で接続します。

下記例は、フリーソフトのTeratermの画面です。



シリアル通信パラメータは、左図の様に設定して下さい。

※通信速度は、コンフィグレーションで変更可能ですが、工場出荷時は、左図の設定になっています。

3.内蔵FlashROMへの書込み

3.1 書込みに必要なもの

RM-240/241の内蔵FlashROMに、プログラムを書き込む為には、以下のツールが必要になります。

	ツール名	説明
1	RM-24X/9XX (開発ボード)	開発ボード
2	RM-240/241	本体通信モジュール
3	マイクロUSBケーブル	RM-24X/9XXEVとパソコンとを接続するのに必要になります。
4	i-Jet	IAR社製JTAG-ICE
5	IAR統合開発環境 (EWARM)	IAR社製統合環境

3.内蔵FlashROMへの書込み

3.2 書き込み手順

以下の手順で書き込み作業を行います。

手順1 お客様のPCに、IAR社のIAR Embedded Workbench統合環境をインストールします

手順2 SimpleMACstdのプロジェクトファイルをクリックして、統合環境を起動します。
※¥Project¥STM32W108 -SimpleMACstd-Ver5.17¥simplemac¥demos¥sample¥sample.eww

手順3 統合環境により、「リビルド」を実行します。(詳細な説明は、統合環境のマニュアルを参照下さい。)

手順4 統合環境により、「ダウンロードしてデバッグ」を実行します。
※この操作によりRM-240/241のFlashROMへの書込みが実行されます。

4.統合環境(EWARM)を使用した開発方法

4.1 統合開発環境構築までの流れ

RM-240/241のF/W開発を行う場合の開発ツールとして、IAR社の開発環境を推奨しています。本章では、ツールのダウンロードからビルド後のデバッグまでの流れについて説明します。

手順1 弊社HPの「ドキュメントダウンロード」ページから、AM-900シリーズの「ソフトウェアツール」→「EWARM(コンパイラ+統合環境)」をクリックし、IAR社専用サイトに接続します。

手順2 IAR社サイトから、「ARM用 30日間期間限定版」を選択して、サイトの指示に従ってユーザー登録を行います。

手順3 ユーザー登録後に、登録したメールアドレスにIAR社からのメールが届きますので、指示に従って環境のダウンロードを行います。

手順4 ダウンロードした実行ファイルを実行します。 ※次頁参照

手順5 SDK添付のCD内から、Projectフォルダ内のプロジェクトサンプルをEWARMにより開きます。

手順6 RM-24X/9XXEV(開発ボード)とJTAG-ICE(i-Jet)を接続します。また、PCとUSB接続を行い、シリアル通信ソフトを起動します。

手順7 ダウンロードしたプロジェクトをビルドとして、エラーが無い事を確認します。

手順8 RM-24X/9XXEVに、実行モジュールをダウンロードしてデバッグできる事を確認します。

4.統合環境(EWARM)を使用した開発方法

4.2 統合開発環境のインストール

<手順4>の説明

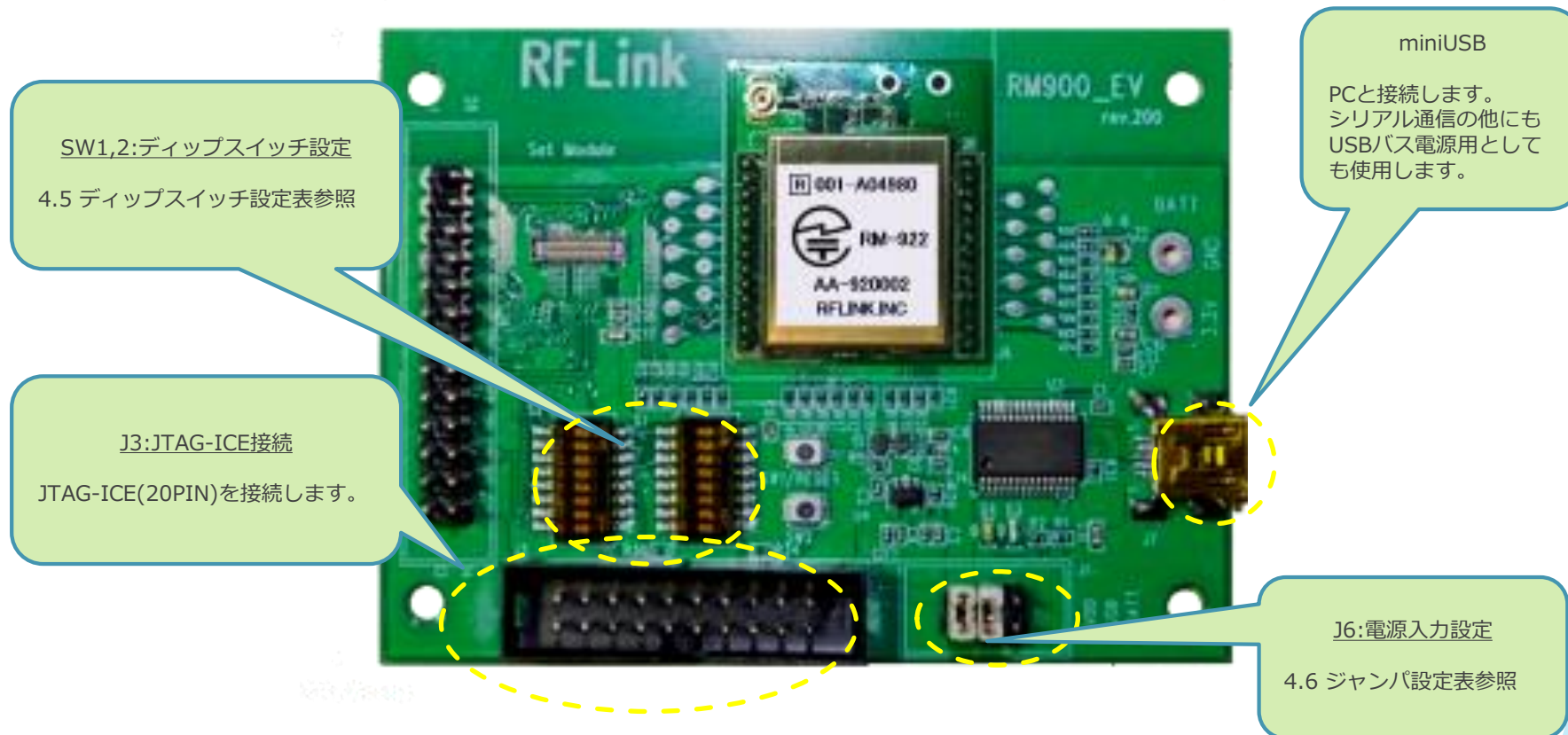
ダウンロードした実行ファイルを実行すると、下記が表示されますので、「IAR Embedde Workbenchのインストール」を選択します。
後は指示に従ってインストール作業を完了して下さい。(GUI画面は随時変更されます)



4.統合環境(EWARM)を使用した開発方法

4.4 RM-900EV Rev.200との接続

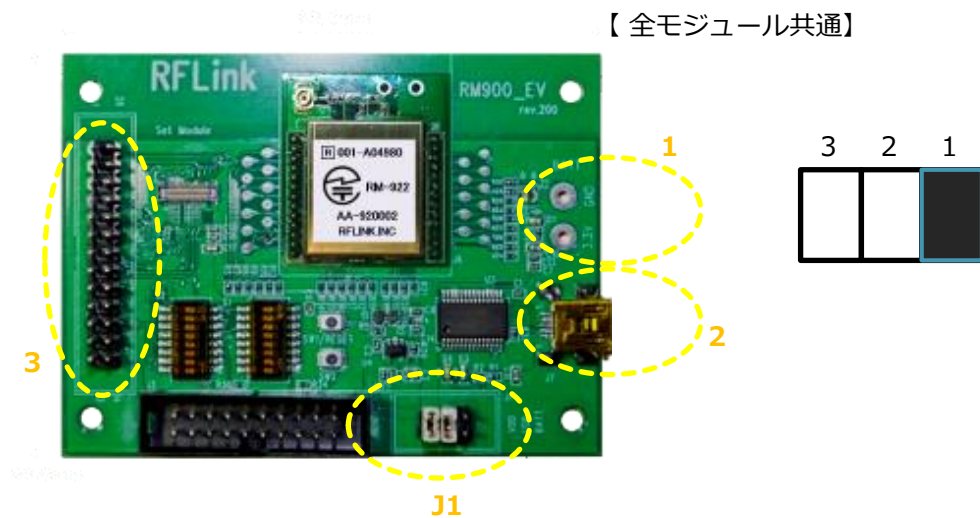
<手順6>の説明
RM-900EVとIAR社のi-Jetを接続します。



4.統合環境(EWARM)を使用した開発方法

4.6 RM-900EV Rev.200 J1 の設定方法

◆RM-900EVは、電源の供給元に応じて、J5の設定を切り替えて使用します。

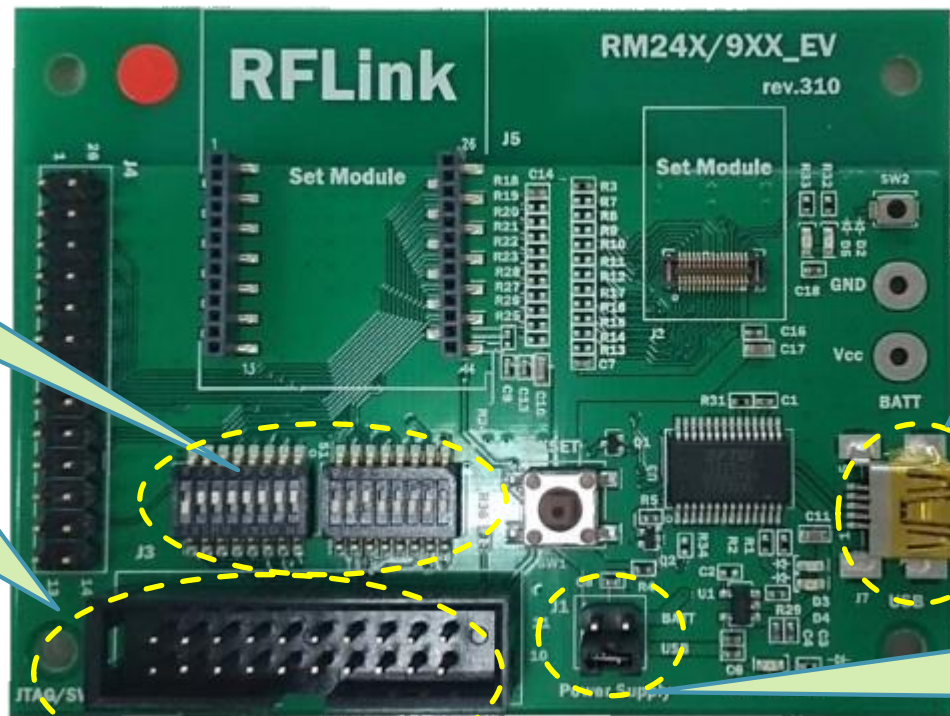


4.統合環境(EWARM)を使用した開発方法

4.7 RM-24X/9XX EV Rev.310との接続

<手順6>の説明

RM-24X/9XX_EV Ver310とIAR社のi-Jetを接続します。



SW1,2:ディップスイッチ設定
4.8 ディップスイッチ設定表参照

J3:JTAG-ICE接続
JTAG-ICE(20PIN)を接続します。

miniUSB
PCと接続します。
シリアル通信の他にも
USBバス電源用として
も使用します。

J6:電源入力設定
4.9 ジャンパ設定表参照

4.統合環境(EWARM)を使用した開発方法

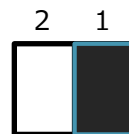
4.9 RM-24X/9XX EV Rev.310 J5 の設定方法

◆RM-24X/9XX_EV Ver310は、電源の供給元に応じて、J1の設定を切り替えて使用します。

【全モジュール共通】



J1



1:BATT
2:USB

4.統合環境(EWARM)を使用した開発方法

4.10 RM-24X/92X Rev.4.0との接続

<手順6>の説明

RM-24X/92X_EV Ver4.0 とIAR社のi-Jetを接続します。

マイクロUSB

PCと接続します。
シリアル通信の他にも
USBバス電源用として
も使用します。



SW1,2:ディップスイッチ設定

4.11 ディップスイッチ設定表参照

J3:JTAG-ICE接続

JTAG-ICE(20PIN)を接続します。

J6:電源入力設定

4.12 ジャンパ設定表参照

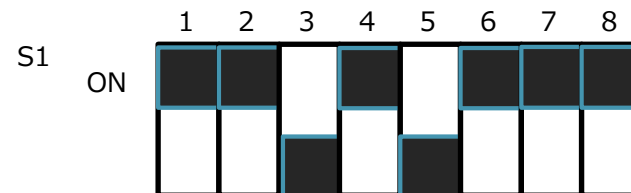
4.統合環境(EWARM)を使用した開発方法

4.11 RM-24X/92X Ver Rev.4.0 SW2 の設定方法

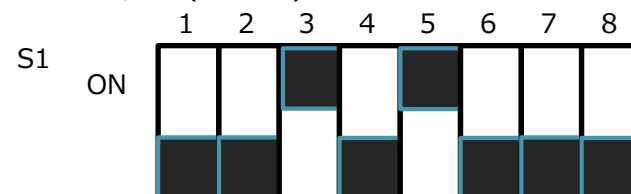
- ◆RM-24X/92X_EV Ver4.0は、920MHz通信モジュール(RM-922/RM-92A)と、2.4GHz通信モジュール(RM-240/241)と共通に使用する事が出来ます。使用するモジュールに応じて、SW2のディップスイッチを切り替えて使用します。



- ◆RM-922/92A(920MHz)モジュールで使用する場合



- ◆RM-240/241(2.4GHz)モジュールで使用する場合

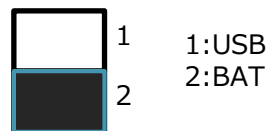


4.統合環境(EWARM)を使用した開発方法

4.12 RM-24X/92X EV Rev.4.0 J5 の設定方法

◆RM-24X/92X_EV Ver4.0 は、電源の供給元に応じて、J1の設定を切り替えて使用します。

【全モジュール共通】



4.統合環境(EWARM)を使用した開発方法

4.13 RM-24X/92X Rev.4.1との接続

<手順6>の説明

RM-24X/92X_EV Ver4.1 とIAR社のi-Jetを接続します。

マイクロUSB

PCと接続します。
シリアル通信の他にも
USBバス電源用として
も使用します。

SW1,2:ディップスイッチ設定

4.11 ディップスイッチ設定表参照



J3:JTAG-ICE接続

JTAG-ICE(20PIN)を接続します。

J6:電源入力設定

4.12 ジャンパ設定表参照

4.統合環境(EWARM)を使用した開発方法

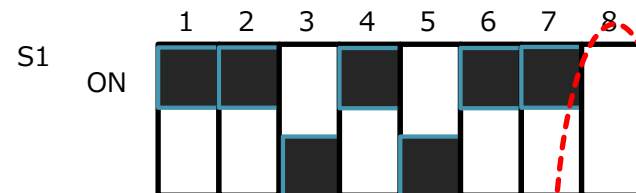
4.14 RM-24X/92X Ver Rev.4.1 SW2 の設定方法

◆RM-24X/92X_EV Ver4.0は、920MHz通信モジュール(RM-922/RM-92A)と、2.4GHz通信モジュール(RM-240/241)と共通に使用する事が出来ます。使用するモジュールに応じて、SW2のディップスイッチを切り替えて使用します。

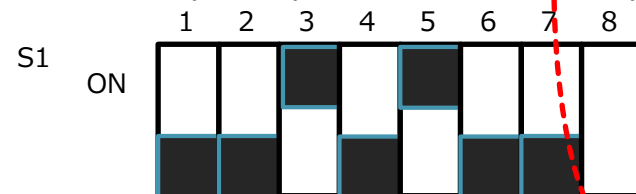
※RM-240/241は、S1の8番SW操作による自動スタート機能は使用できません。



◆RM-922/92A(920MHz)モジュールで使用する場合(有効ビット1~7)



◆RM-240/241(2.4GHz)モジュールで使用する場合(有効ビット1~7)



自動スタート設定

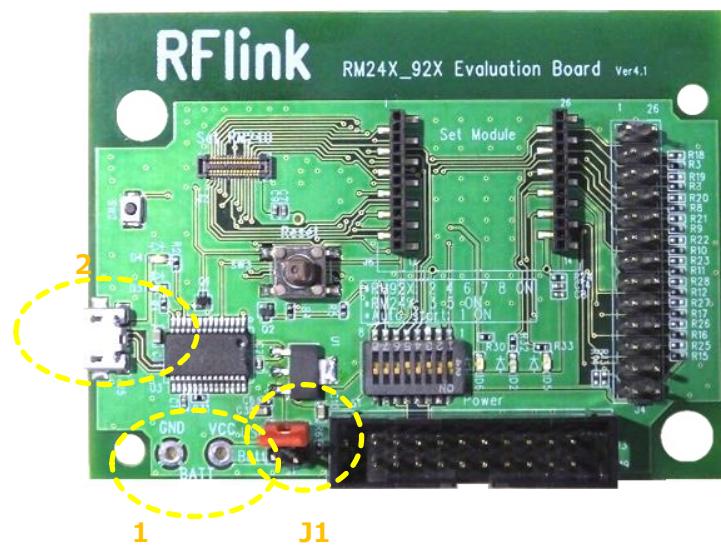
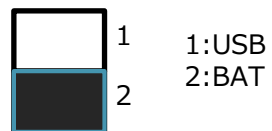
ON : 電源投入後・自動スタート
OFF : 電源投入後・手動スタート

4.統合環境(EWARM)を使用した開発方法

4.15 RM-24X/92X EV Rev.4.1 J5 の設定方法

◆RM-24X/92X_EV Ver4.0 は、電源の供給元に応じて、J1の設定を切り替えて使用します。

【全モジュール共通】

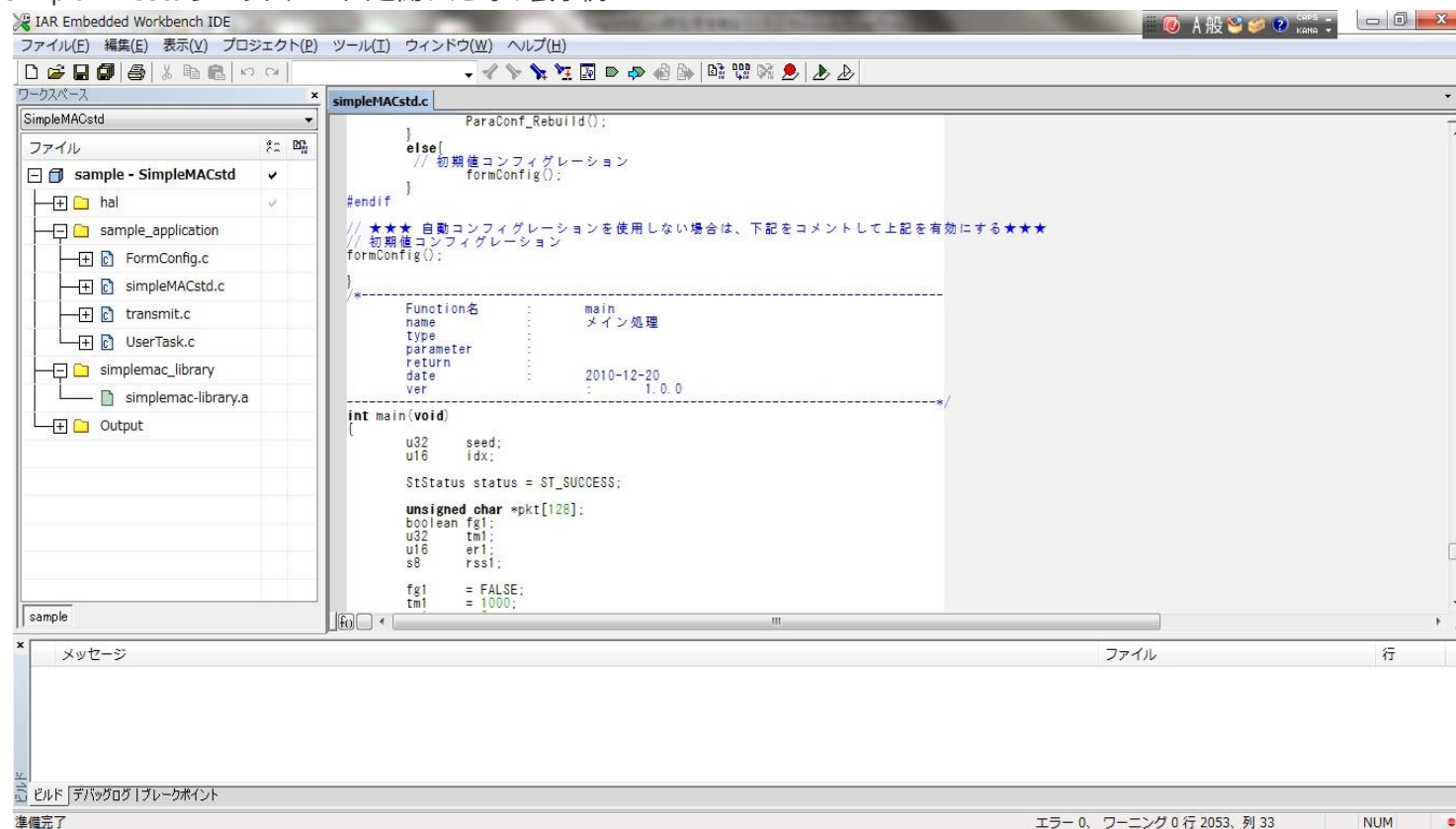


4. 統合環境(EWARM)を使用した開発方法

4.16 プロジェクトファイルの展開

<手順7>の説明

SimpleMACStdワークスペースを開いた時の表示例



ビルド | デバッガログ | ブレークポイント

準備完了

エラー-0、ワーニング0行 2053、列 33

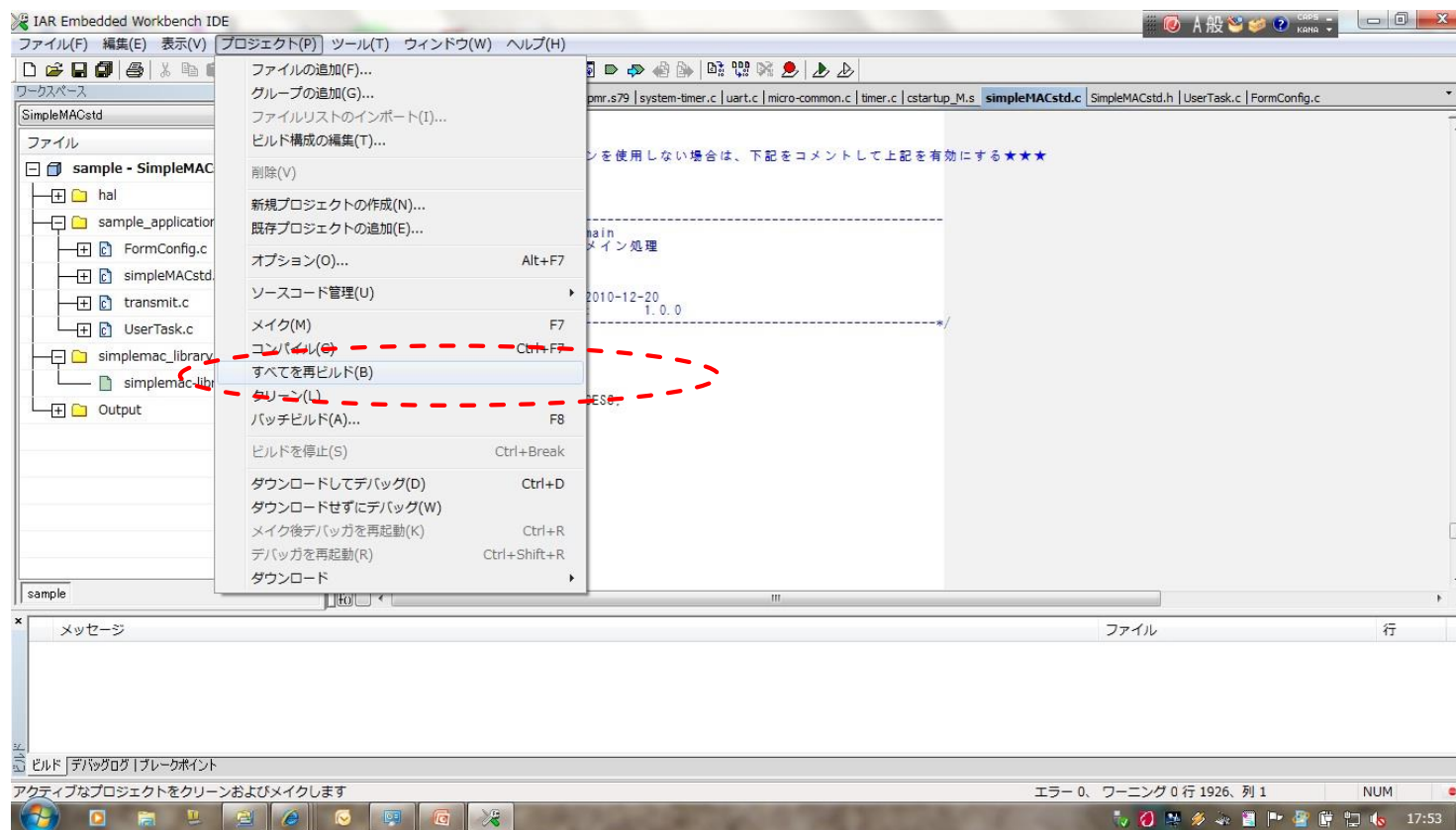
NUM

4. 統合環境(EWARM)を使用した開発方法

4.17 プロジェクトのビルド

<手順7>の説明

プロジェクトのビルドを実行します。下図の通り、「プロジェクト」→「全てを再ビルド」を実行します。

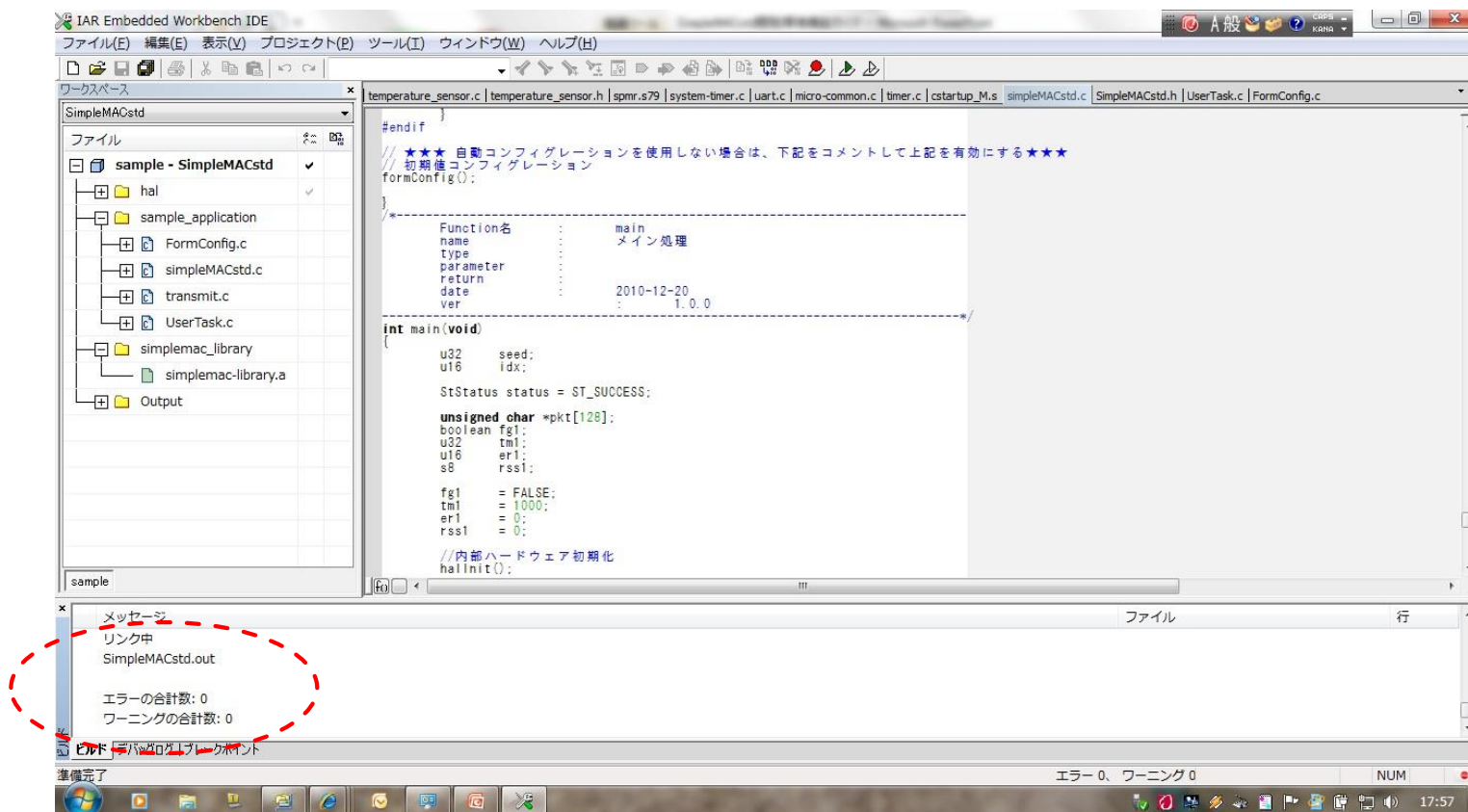


4. 統合環境(EWARM)を使用した開発方法

4.18 プロジェクトのビルド

<手順7>の説明

ビルド終了後に、エラー、ワーニング、が無い事を確認します。

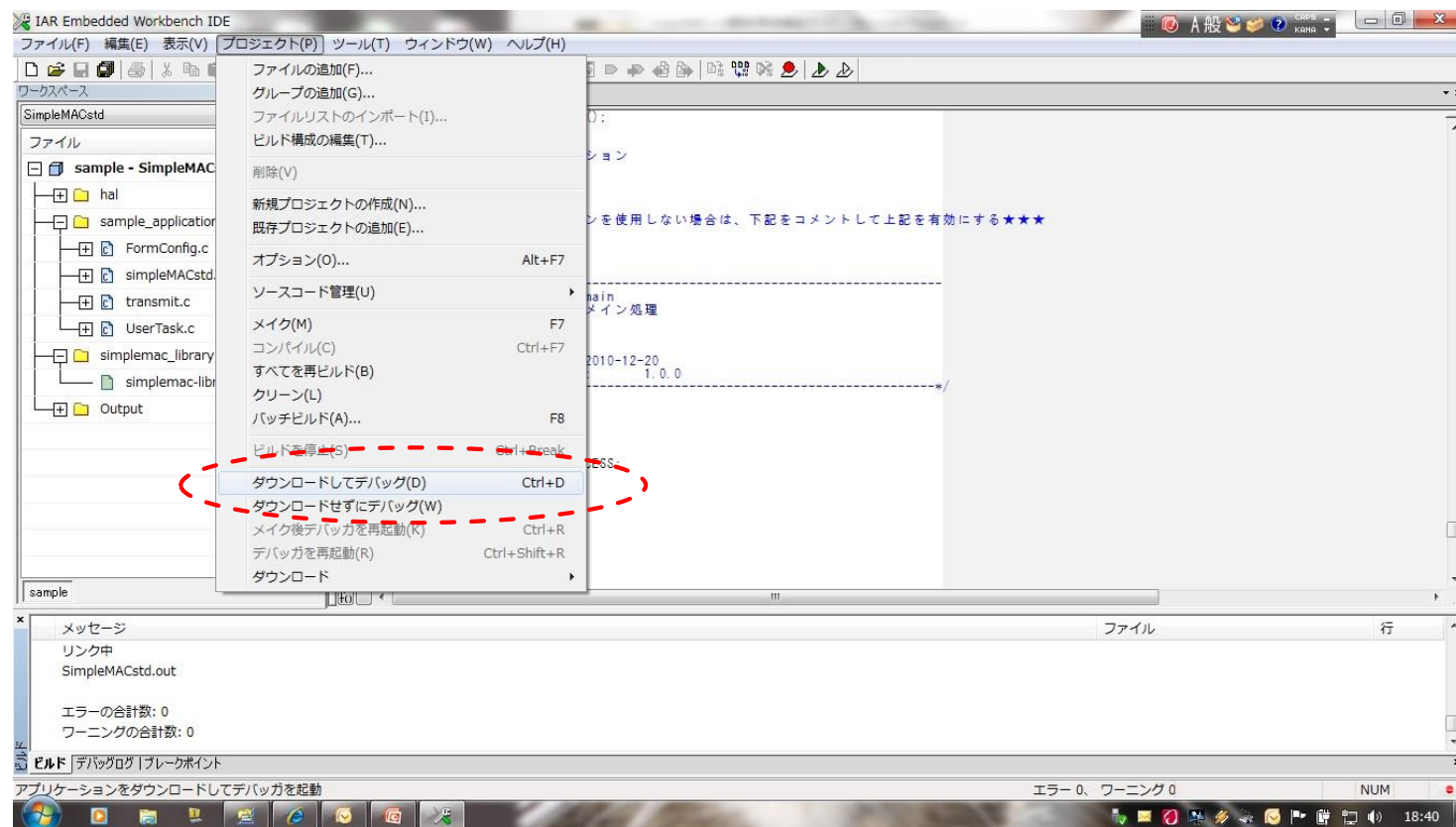


4. 統合環境(EWARM)を使用した開発方法

4.19 実行モジュールのRM-922/92A/92Cへのダウンロード

<手順8>の説明

「プロジェクト」→「ダウンロードしてデバッグ」を選択します。→RM-922/92Aの内蔵FROMに実行ファイルがダウンロードがされます。



4. 統合環境(EWARM)を使用した開発方法

4.20 実行モジュールのRM-922/92A/92Cへのダウンロード

<手順8>の説明

「プロジェクト」→「ダウンロードしてデバッグ」を選択します。→RM-920の内蔵FROMに実行ファイルがダウンロードがされます。

The screenshot shows the IAR Embedded Workbench IDE interface. The main window displays the source code for `simpleMACstd.c`. The code includes a `ParaConf_Rebuild()` function and a `main` function. A red dashed circle highlights the `int main(void)` function signature. Below the code, a table lists the function's details:

Function名	main
name	メイン処理
type	
parameter	
return	
date	2010-12-20
ver	1.0.0

To the right of the code editor, the 'レジスタ' (Registers) window is open, displaying the CPU register list. The registers are listed with their values and names:

Register	Value	Register	Value
R0	0x00000000	IAPSR	0x60000000
R1	0x080000B0	EAPSR	0x61000000
R2	0x080000B0	IEPSR	0x01000000
R3	0x00000000	CYCLECOUNTER	0
R4	0x00000000	CCTIMER1	0
R5	0x08003798	CCTIMER2	0
R6	0x08003740	CGSTEP	0
R7	0xFFFFFFFF		
R8	0x200001B0		
R9	0x46A54870		
R10	0x3EE0CACE		
R11	0x2C03485		
R12	0x08006D06		
R13 (SP)	0x20000400		
R14 (LR)	0x08006503		
xPSR	0x61000000		
APSR	0x60000000		
IPSR	0x00000000		
EPSR	0x01000000		
PC	0x08003EC0		
R13_main (MSP)	0x20000400		
R13_proc (PSP)	0x4743C738		
PRIMASK	0x00000000		
BASEPRI	0x00000060		
BASEPRI_MAX	0x00000060		
FAULTMASK	0x00000000		
CONTROL	0x00000000		

The bottom window shows the 'ログ' (Log) window with the following output:

```
Wed Apr 20 18:23:36 2011: TotalIRLen = 8, IRPrint = 0x00E1
Wed Apr 20 18:23:36 2011: Found Cortex-M3 r1p1, Little endian.
Wed Apr 20 18:23:36 2011: TPIU fitted.
Wed Apr 20 18:23:36 2011: FPUUnit: 6 code (BP) slots and 2 literal slots
Wed Apr 20 18:23:36 2011: 方法0によるハードウェアリセットが実行されました
Wed Apr 20 18:23:36 2011: ターゲットリセット
```

4. 統合環境(EWARM)を使用した開発方法

4.21 デバッグ開始

<手順8>の説明

下図の赤丸部をクリックして実行します。

The screenshot shows the IAR Embedded Workbench IDE interface. The main window displays a C source file named 'simpleMACStd.c'. The code includes a function 'ParaConf_Rebuild()' and a 'main' function. A red circle highlights the 'Run' button (a play icon) in the toolbar. The 'Registers' window on the right shows the CPU register list with values for R0 through R14, xPSR, and other system registers. The 'Log' window at the bottom shows system initialization messages, including 'TotalIRLen = 8, IRPrint = 0x00E1', 'Found Cortex-M3 r1p1, Little endian.', and 'TPIU fitted.'.

```
ParaConf_Rebuild():
else{
// 初期値コンフィグレーション
formConfig();
}
#endif
// ★★★ 自動コンフィグレーションを使用しない場合は、下記をコメントして上記を有効にする★★★
// 初期値コンフィグレーション
formConfig();
-----
Function名      :      main
name           :      メイン処理
type          :
parameter     :
return        :
date          :      2010-12-20
ver           :      1.0.0
-----
int main(void)
{
    u32  seed;
    u16  idx;

    StStatus status = ST_SUCCESS;

    unsigned char *pkt[128];
    boolean fgl;
    u32  tml;
    u16  erl;
    s8   rssl;
```

Register	Value	Register	Value
R0	0x00000000	IAPSR	0x60000000
R1	0x080000B0	EAPSR	0x61000000
R2	0x080000B0	IEPSR	0x01000000
R3	0x00000000	CYCLECOUNTER	0
R4	0x00000000	CCTIMER1	0
R5	0x08003798	CCTIMER2	0
R6	0x08003740	CCSTEP	0
R7	0xFFFFFFFF		
R8	0x200001B0		
R9	0x46A54870		
R10	0x3EE0CACE		
R11	0x2C03485		
R12	0x08006D06		
R13 (SP)	0x20000400		
R14 (LR)	0x08006503		
xPSR	0x61000000		
APSR	0x60000000		
IPSR	0x00000000		
EPSR	0x01000000		
PC	0x08003EC0		
R13_main (MSP)	0x20000400		
R13_proc (PSP)	0x4743C738		
PRIMASK	0x00000000		
BASEPRI	0x00000060		
BASEPRI_MAX	0x00000060		
FAULTMASK	0x00000000		
CONTROL	0x00000000		

Log
Wed Apr 20 18:23:36 2011: TotalIRLen = 8, IRPrint = 0x00E1
Wed Apr 20 18:23:36 2011: Found Cortex-M3 r1p1, Little endian.
Wed Apr 20 18:23:36 2011: TPIU fitted.
Wed Apr 20 18:23:36 2011: FPUUnit: 6 code (BP) slots and 2 literal slots
Wed Apr 20 18:23:36 2011: 方法0によるハードウェアリセットが実行されました
Wed Apr 20 18:23:36 2011: ターゲットリセット

4.統合環境(EWARM)を使用した開発方法


4.22 プログラムの実行～シリアル通信ソフトによる確認

<手順8>の説明

前頁の「実行」操作により、正常に実行されると、PCのシリアル通信ソフト(以下はTeraterm)に、SimpleMACstdから起動メッセージが表示されます。

“0” を入力すると

最初に表示される画面



次に表示される画面 (コンフィグレーションの基本メニュー)



```
COM25 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウインドウ(W) 漢字コード(C) ヘルプ(H)
Command I/F MODE [0:Message Mode 1:Simple Mode] ?= []

COM25 - Tera Term VT
ファイル(F) 編集(E) 設定(S) コントロール(O) ウインドウ(W) 漢字コード(C) ヘルプ(H)
Command I/F MODE [0:Message Mode 1:Simple Mode] ?= 0
SimpleMACstd (1.0.1) Sample Application:
Enter ? Please do command choice
>
*****
* SimpleMACstd Command List                               *
* Version 5.1.7                                           *
*****
[a] : Channel No Set           [11 - 26 ]
[b] : PAN Address Set          [1 - 65534 ]
[c] : SRC-Address Set          [1 - 65534 ]
[d] : HOP Counter Set          [1 - 255 ]
[e] : DST-Address Set          [1 - 65535 ]
[f] : BROADCAST Transfer Mode [0 or 1 or 2]
[j] : Timer Handler Cycle Set  [1(1ms) to 65535(65.535s)]
[k] : Timer Handler Use        [0:Not Use 1:Use]
[l] : Sleep Mode Set           [0: Sleep Not Use 1: Timer Wake Up 2: GPIO Wake Up]
[m] : Sleep Time Set           [0 - 65530(per 250msec)]
[n] : Sleep Mode Used Set      [0:Not Use 1:Use]
[o] : Uart BPS Value Set       [1 to 10]
1:4800 2:9600 3:14400 4:19200 5:38400 6:57600 7:115200 8:230400 9:460800
10:921600
[p] : RSSI Value set           [0: Not Use 1:Use]
[q] : PAN Address Invalidity    [0: Not Use 1:Use]
[r] : Configuration Data FlashROM Read
[t] : Re-try Count Set         [1 - 255 ]
[u] : RF TX-Power Value Set     [0:+3dbm to 46:-43dbm]
[v] : RF TX-Boost(+4dbm) Set    [0:Not Use 1:Use]
[w] : Configuration Data FlashROM Write
[x] : Data Transfer Mode        [0: Discharge 1:flag+Discharge 2:Frame]
[y] : Trans of SRC-Address Out  [0: Not output 1:Output]

[s] : System Start
[?] : State indication
Help : Return

Please input!! >[]
```

開発環境構築ガイド

◆ Release version

Version 2.2.0 2017-03-28