

RTC-Lite

使用説明書

首都大学東京	高山勇人
	和田一義
東京電機大学	下山直紀
大阪大学	大原賢一

1. はじめに

本マニュアルでは、RTC-Lite を使用するための手順について示す。

2. RTC-Lite の特徴

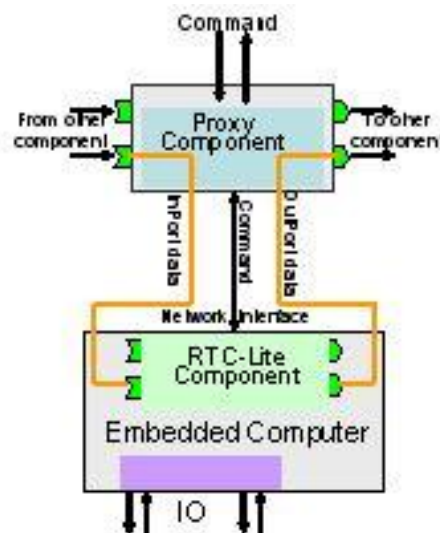
RTC-Lite は、通常の RT コンポーネントを作成するのと同じ感覚で、組み込み MPU 向けの RT コンポーネントを作成、動作させる環境を提供する。作成するハードウェアにより、通信網、コンパイラが異なってくるが、実装形態を工夫することで、コンパイラ依存部と、非依存の共通要素を可能な限り分離し、さらには、通信網に依存する部分も可能な限り分離を行っているため、ユーザサイドで新規デバイスに RTC-Lite を拡張していく場合のことも考慮されている。

RTC-Lite は、通常の RT コンポーネントと同じように利用できるようにするために、以下のような 2 つの要素から構成される。

- Proxy コンポーネント
- RTC-Lite コンポーネント

Proxy コンポーネントは通常の PC 上で利用することを想定しており、RTC-Lite コンポーネントは、組み込み MPU（ここでは PIC, dsPIC 上）で動作する。

このように、他のコンポーネントとの橋渡しとして、Proxy コンポーネントが機能し、コンポーネントの実体は MPU 上に存在するという構造を RTC-Lite では採用している。



3. 制限事項

本バージョンの RTC-Lite では、PIC の入出力、AD 変換に関わるポートは事前に取り決めており、そのポート以外の利用はできないので注意していただきたい。（ユーザは、ソース中で、機能を利用するかしないかを選択するのみ。）

4. 配布物

- RTC-Lite 使用説明書(RTC-Lite 使用説明書.docx)
- Proxy コンポーネントソースファイル(Proxy.zip)
- dsPIC 版 RTC-Lite ソースファイル(dsPIC.zip)
- PIC 版 RTC-Lite ソースファイル(PIC.zip)

5. 動作環境

RTC-Lite を使用するためには，以下のものが必要となる．

- PC 1 台
- PIC, dsPIC と Xport を搭載した回路基板 1 台

なお，PC には，OpenRTM-aist-0.4.2 の動作環境が整っていることを前提としており，本マニュアル作成に当たる検証は以下の環境で行われている．

RTC-Lite 動作環境

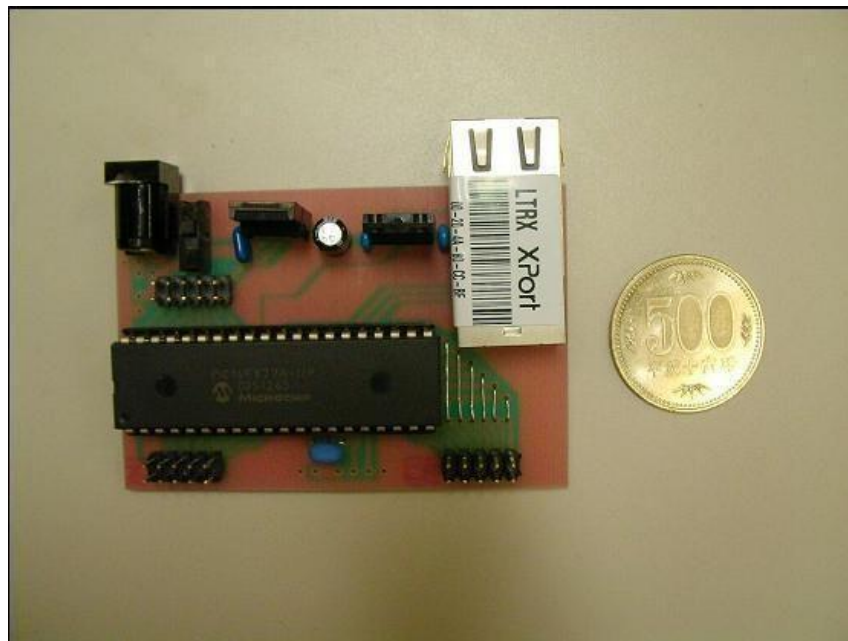
OS	Windows XP
Python Version	2.4.2
OpenRTM Version	0.4.2
Language	Python

6. RT-Unit の作成

RTC-Lite が動作する RT デバイス制御ボードとして、PIC16F877A, dsPIC30F3014 を制御 MPU に採用した RT-Unit を利用する．以降，この RT-Unit について説明を行う．

6-1. PIC16F877A を搭載した RT-Unit

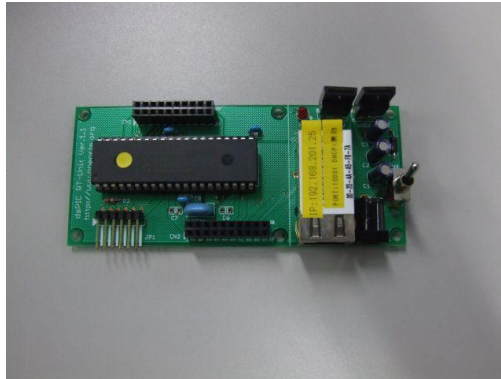
i) 外観



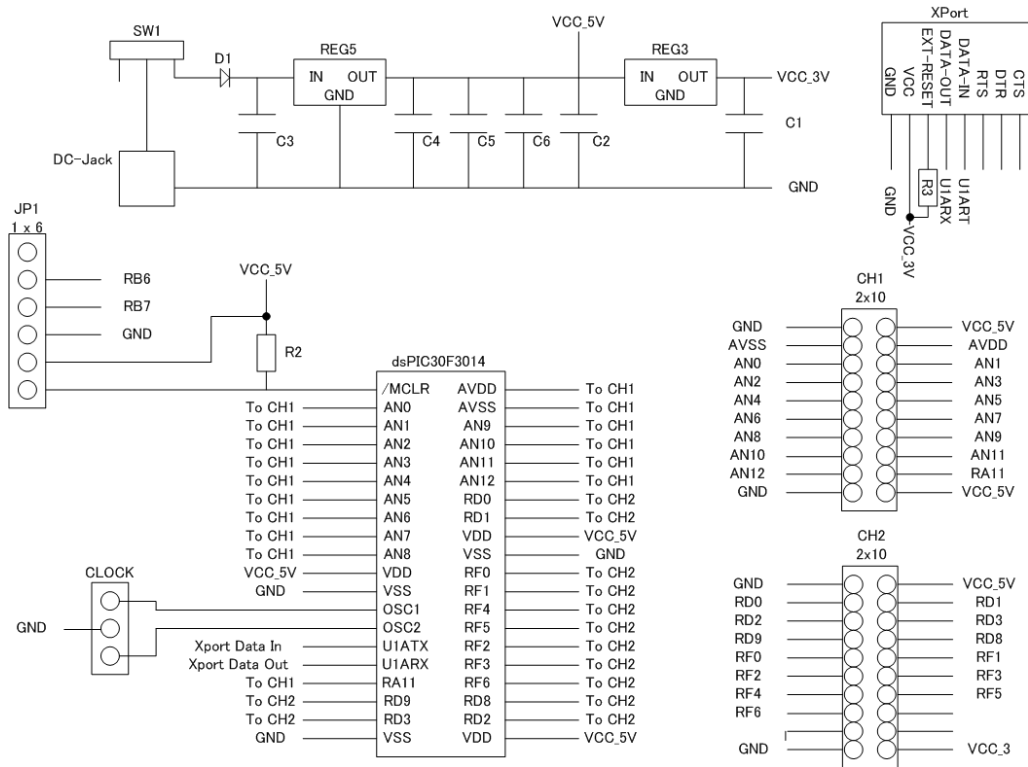
ii) 回路図

6-2. dsPIC30F3014 を搭載した RT-Unit

i) 外観



ii) 回路図



iii) 部品表

部品名	デバイス	型番, 値	個数
C1,C2,C3	電界コンデンサ($\phi 5$)	33uF	3
C4,C5,C6	積層セラミックコンデンサ	0.1uF	3
R1	抵抗	330 Ω	1
R2,R3	抵抗	10k Ω	2
D1	ダイオード	1N400	1
JP1	ピンヘッダ	1 x 6	1
CH1,CH2	ピンヘッダ	2 x 10	2
CLOCK	セラロック	10MHz	1
IC1	dsPIC30F3014	dsPIC30F3014	1
DC-Jack	DC ジャック	MJ-179P	1
SW1	スイッチ	AK-P300	1
LED	LED		1
REG5	3 端子レギュレータ	LM2940CT5	1
REG3	3 端子レギュレータ	TA48033S	1
XPort	XPort	XPort	1

7. RT-Unit のプログラム開発

使用する PIC にかかわらず、本開発では統合開発環境として、Microchip 社から提供されている MPLAB を利用することとする。

http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&dDocName=en019469

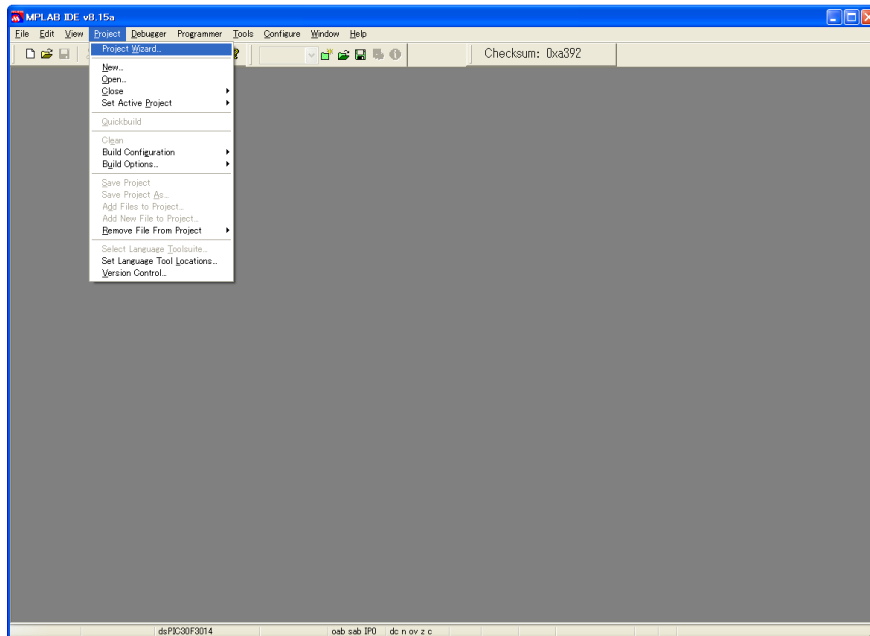
7-1. 16F877A の場合

PIC を C 言語で開発するためには、C コンパイラが不可欠である。RTC-Lite では、16 系の PIC の開発環境として、CCS 社によって販売されている CCS C コンパイラを利用している。そのため、16 系のデバイスを利用するためには、CCS C コンパイラを購入する必要がある。

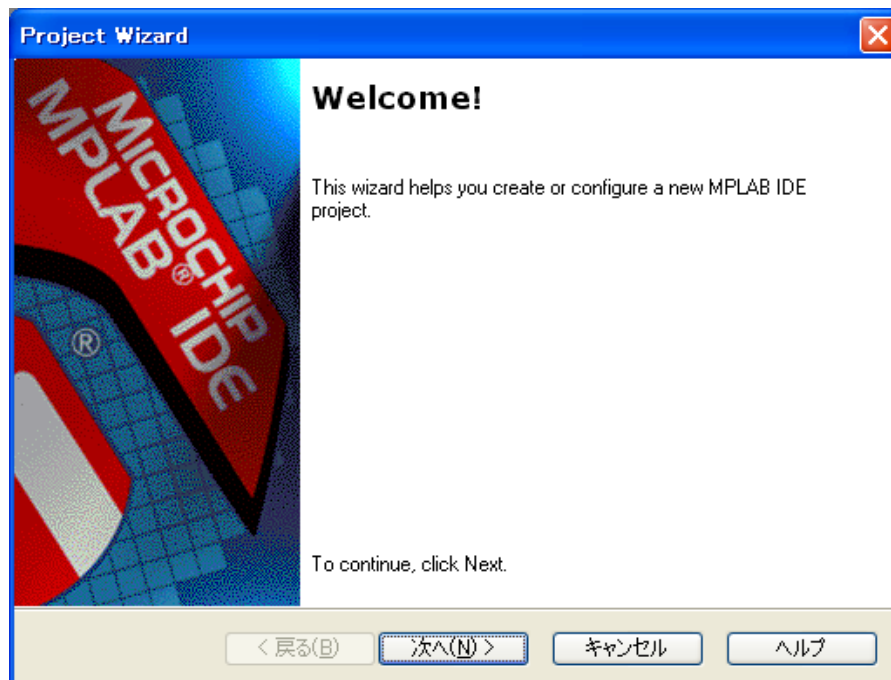
<http://www.ccsinfo.com/content.php?page=compilers>

i) プロジェクトの作成

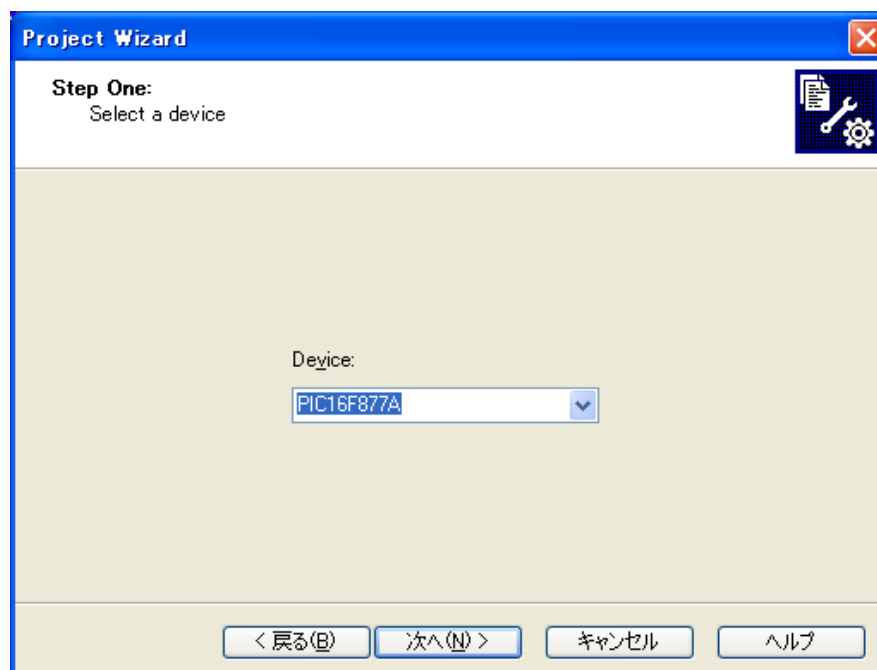
- (1) 前準備として、添付されている PIC フォルダにあるすべてのファイルを任意のフォルダ(例:C¥RTC-Lite_PIC)にコピーする。
- (2) MPLAB を立ち上げる。



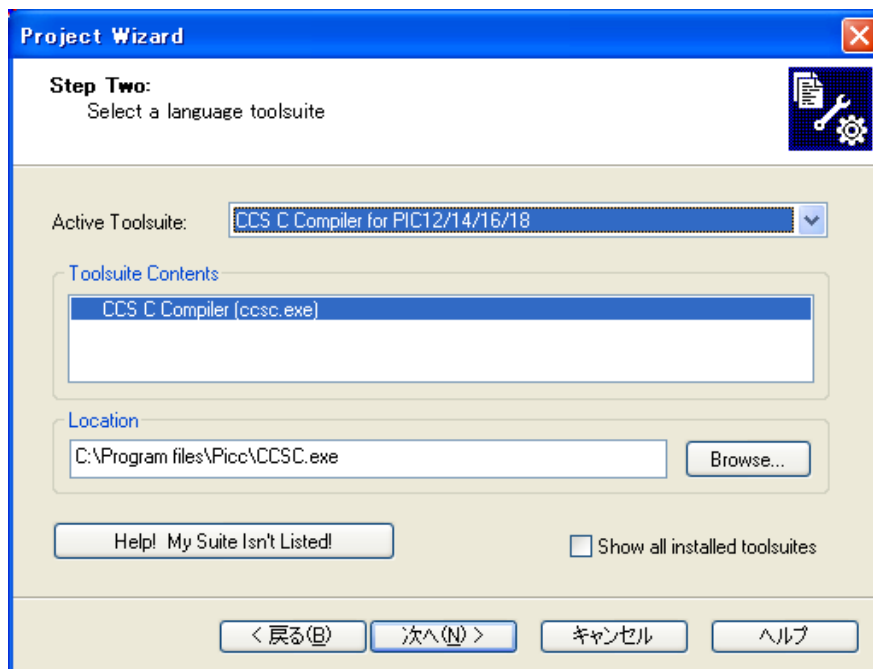
- (3) [Project] → [Project Wizard]をクリックする。



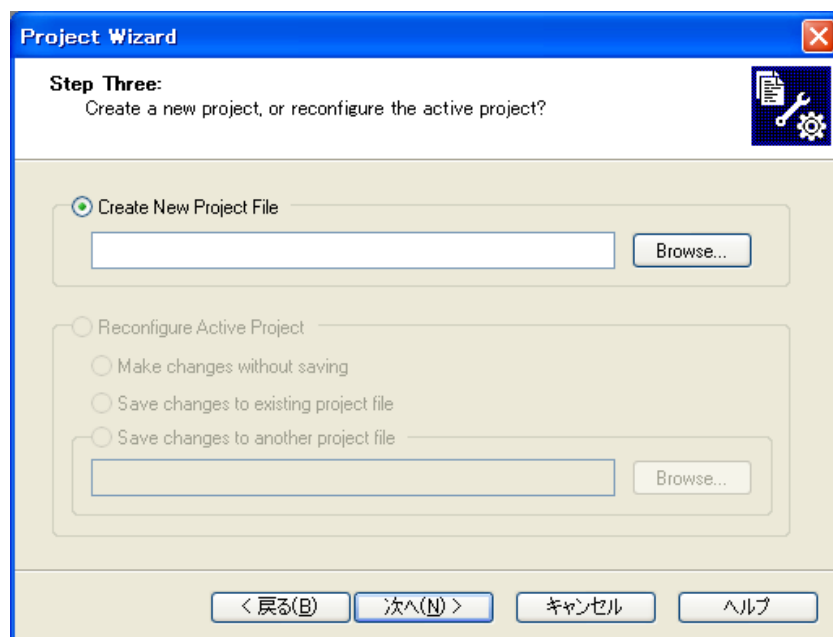
(4) [次へ]をクリック。



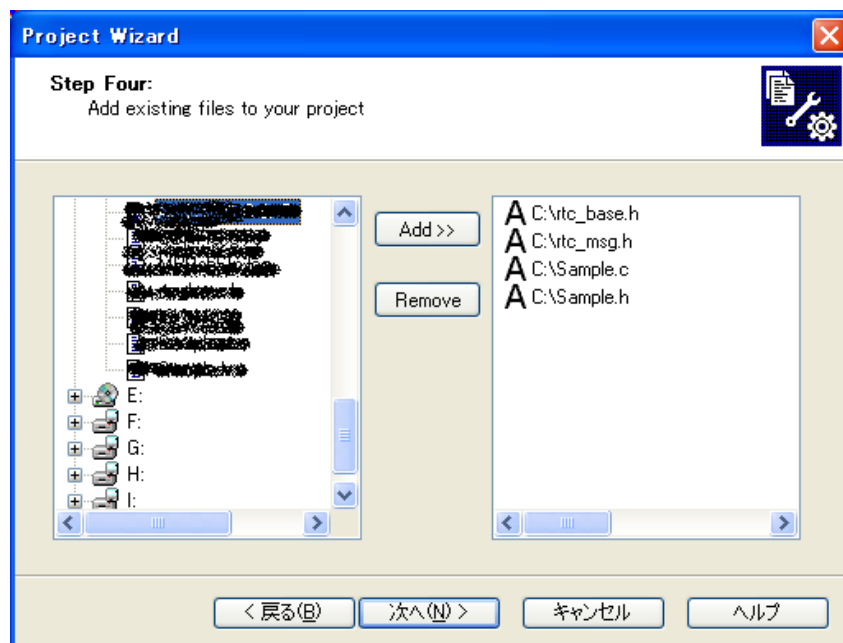
(5) 16F877A を選択し，次へ。



(6) CCS C Compiler を選択し，次へ．



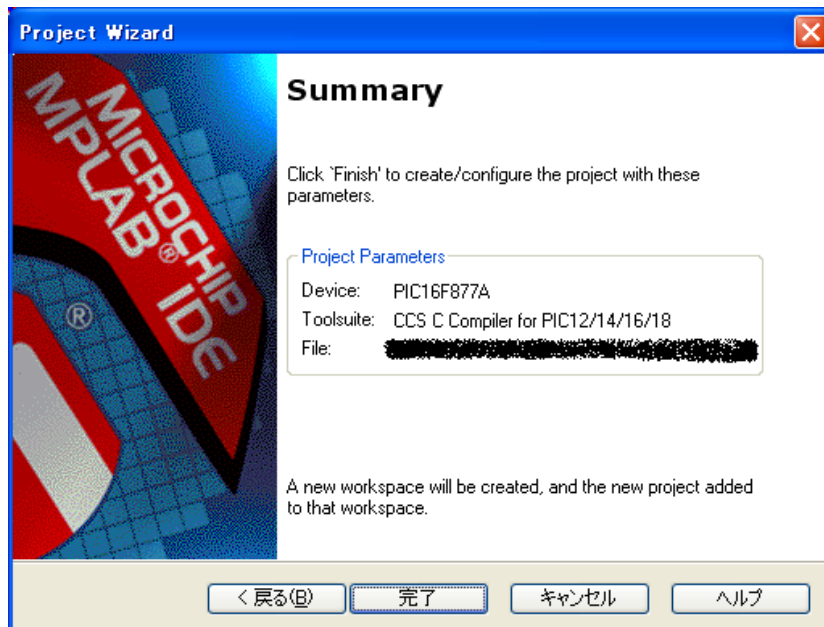
(7) (1)で作成したフォルダを指定しプロジェクト名を代入後，次へ．



(8) (1)で作成したフォルダの下記のファイルを選択して，Add を押し，次へ．

- Sample.c • • • (メインファイル)
- Sample.h • • • (メインファイルのヘッダ)
- rtc_base.h • • • (rtclite の各要素を管理するファイルのヘッダ)
- rtc_msg.h • • • (rtclite の通信関係のヘッダ)

* 同フォルダに入っている rtc_base.c, rtc_inport.c, rtc_outport.c は MPLAB のプロジェクト内に入れるとコンパイルエラーを起こします．しかし，各ファイルで書き込んだ内容はコンパイル時に反映されます．



(9) [完了]を押すとプロジェクト作成終了.

ii) ファイル説明

- sample.c
- sample.h
- rtc_base.c
- rtc_base.h
- rtc_msg.h
- inport.c
- outport.c

sample.c はコンポーネントのメインの部分であり、ユーザが書き換える必要のあるファイルの一つである。RT コンポーネントのアクティビティはここに書かれている。

sample.h はコンポーネントのプロファイルと CCS ヘッダを読む部分であり、ユーザが書き換える必要のあるファイルである。書き換える必要がある部分として、プロファイルとプロファイルの文字数の変更とシリアルの設定はここで変更する。

rtc_base.c は RT コンポーネントのアクティビティの管理など、RT コンポーネント共通となる要素の記述が行われている。ユーザはこのファイルを書き換える必要はない。

rtc_base.h は rtc_base.c のヘッダファイルである。ユーザはこのファイルを書き換える必要はない。

rtc_msg.h は Proxy コンポーネントとの間のメッセージのやり取りを管理する機能を提供する。ユーザはこのファイルを書き換える必要はない。

inport.c は inport 関連の動作について記述が行われている．現状では，利用する inport に応じて，適宜修正する必要あり．

Outport.c は outport 関連の動作について記述が行われている．現状では，利用する outport に応じて，適宜修正する必要がある．

7-2. dsPIC30F3014 の場合

i) 開発環境のインストール

dsPIC の C 言語開発環境として、Microchip 社から C-30 コンパイラが販売されており、こちらを利用している。C-30 コンパイラは Student Edition として、期間限定版が公開されており、本開発においてもこちらを利用している。Student Edition の特徴として、60 日間の使用期限があるものの、60 日以後もコンパイラの最適化が上がった状態で使用を続けることができる。

http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&DocName=en535363

MPLAB がインストールされた状態であれば、C-30 コンパイラインストール後、すぐに利用できる状態となっている。

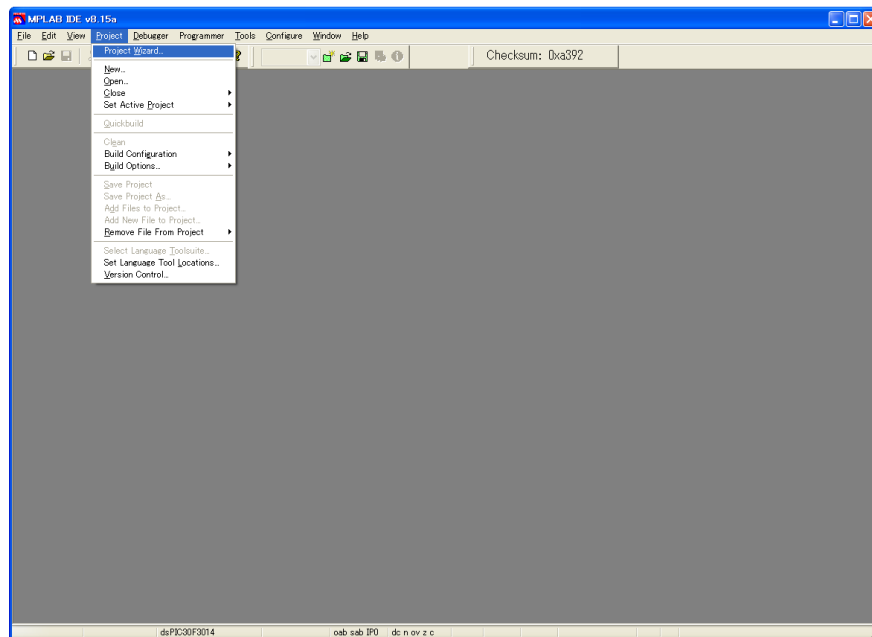
PIC への書き込みを行うためのライターとして、Microchip 社から販売されている、PicKit2 を利用する。ただし、MPLAB では、PicKit2 を利用しての書き込みが正常に行えないことから、PicKit2 用の書き込みソフトウェアである PicKit2 Programmer を利用する。

http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1406&DocName=en023805

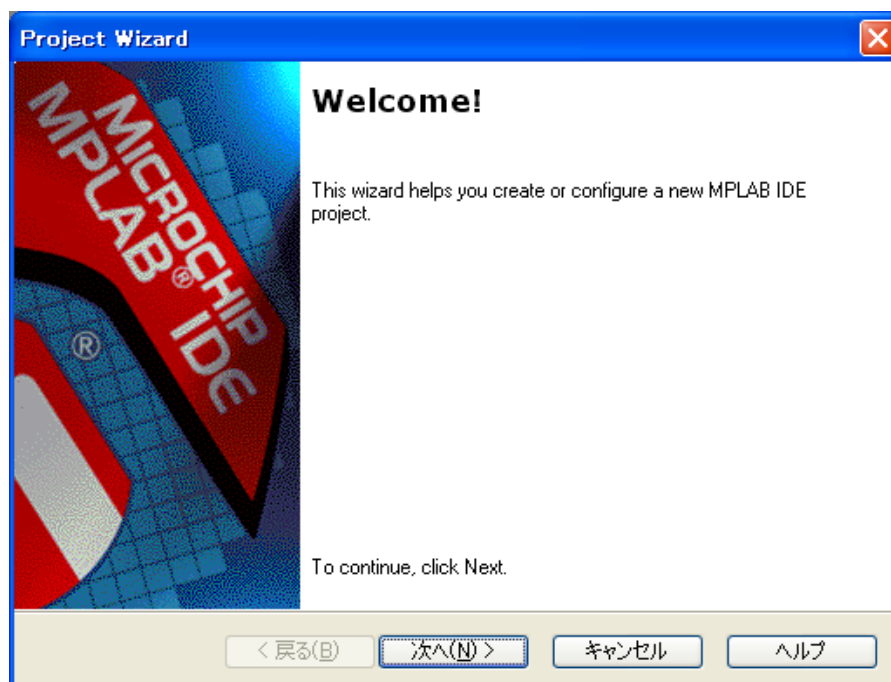
各ソフトウェアの使用方法については本マニュアルでは割愛する。

ii) 開発用プロジェクトの作成

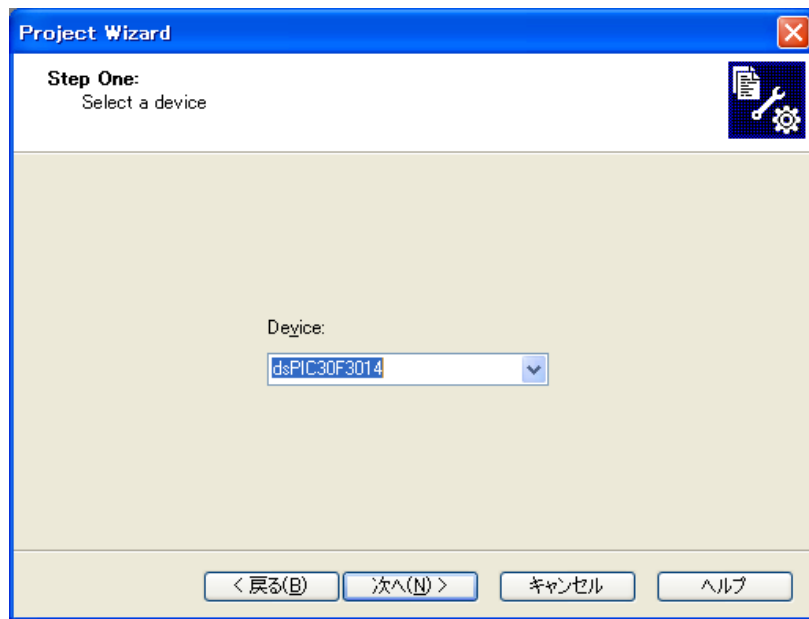
RTC-Lite のプログラム開発を行うために、はじめに開発用のプロジェクトを作成する。以下に手順を示す。



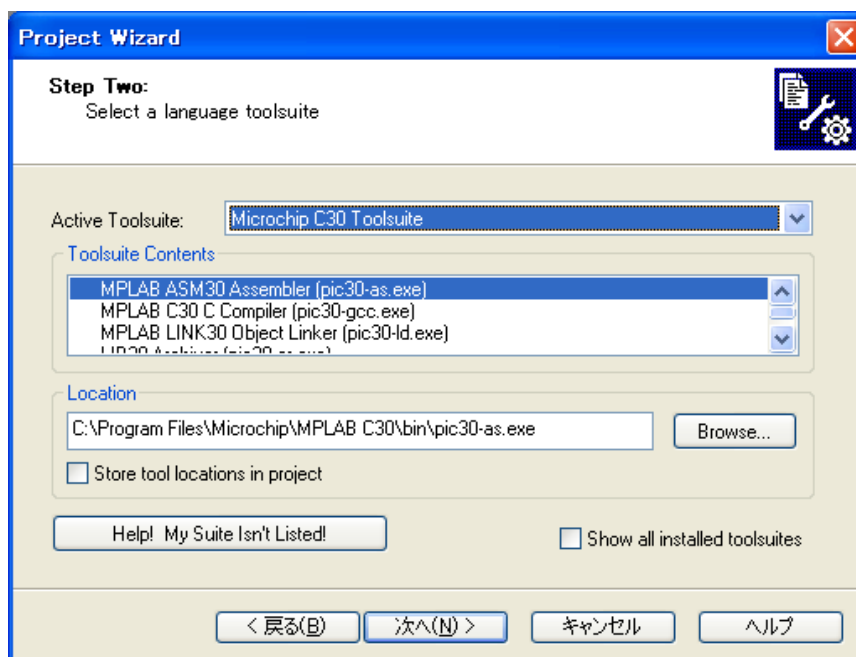
(1) [Project] → [Project Wizard]



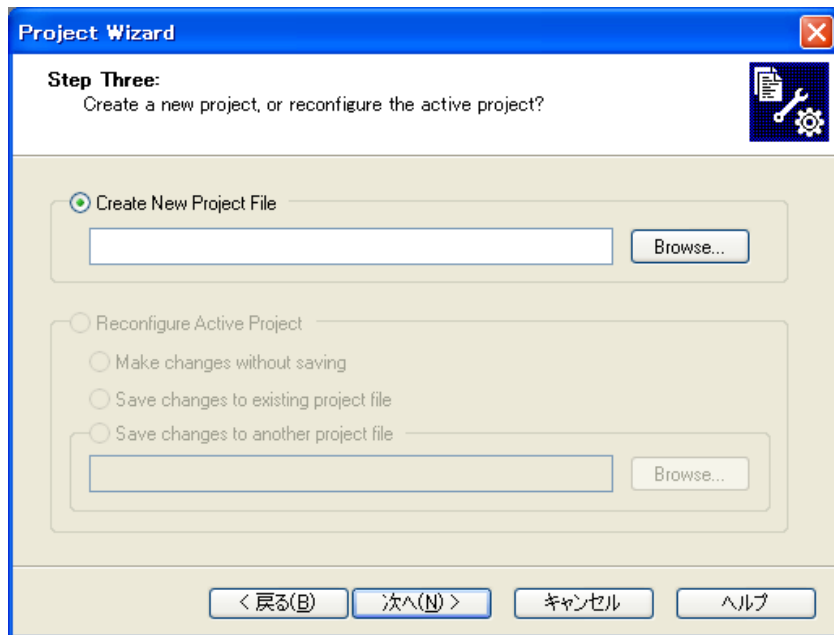
(2) [次へ]をクリック



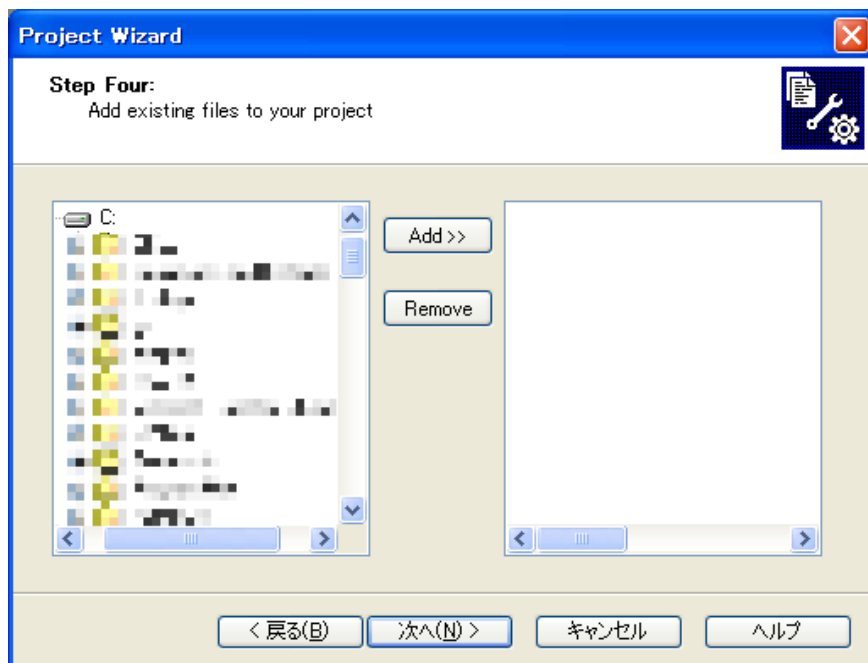
(3) dsPIC30F3014 を指定して次へ



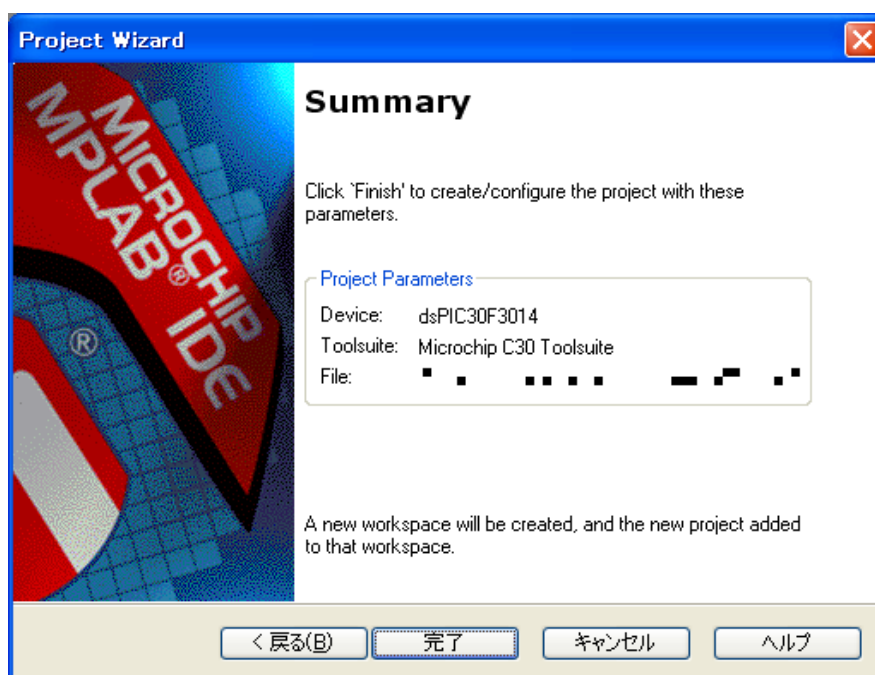
(4) Active Toolsuite を[Microchip C30 Toolsuite]にする



(5) 任意の場所にプロジェクトを作る。(日本語をパスに含まないところにする.)



(6) [次へ]をクリック



(7) [完了]を押すとプロジェクト作成終了

iii) 必要なファイルのプロジェクトへの追加

C-30 では、開発の手間を削減するために、様々なライブラリが組み込まれている。RTC-Lite を利用する上でもこれらのファイルは必要であるため、以下のファイルを読み込んでおく必要がある。

• Header Files

uart.h

<C-30 ディレクトリ>%support%peripheral_30F_24H_33F

adc12.h

• Library Files

Libp30F3014-coff.a

<C-30 ディレクトリ>%lib%dsPIC30F

• Linker Script

P30f3014.gld

<C-30 ディレクトリ>%support%dsPIC30F%gld

iv) RTC-Lite のソースの追加

RTC-Lite 用のソースをプロジェクトに読み込む。現在はコンポーネント作成用のスクリプトが存在しないため、既存のソースの中身を書き換えることで、実装を行う。なお、現在のバージョンでは、入出力用のポート、A/D 変換用のポートなどは固定であり、ユーザ

は利用したいポートに RT デバイスの信号線を接続する必要がある。
プロジェクトに追加するソースファイルは、以下のとおりである。

- sample.c
- rtc_base.h
- rtc_message.h
- rtc_inport.h
- rtc_outport.h
- rtc_global.h
- C30_dsPIC30F3014Def.h

sample.c はコンポーネントのメインの部分であり、ユーザが書き換える必要のあるファイルの一つである。RT コンポーネントのアクティビティはここに書かれている。

rtc_base.h は RT コンポーネントのアクティビティの管理など、RT コンポーネント共通となる要素の記述が行われている。ユーザはこのファイルを書き換える必要はない。

rtc_message.h は Proxy コンポーネントとの間のメッセージのやり取りを管理する機能を提供する。ユーザはこのファイルを書き換える必要はない。

rtc_inport.h は inport 関連の動作について記述が行われている。現状では、利用する inport に応じて、適宜修正する必要あり。

rtc_outport.h は outport 関連の動作について記述が行われている。現状では、利用する outport に応じて、適宜修正する必要がある。

rtc_global.h は RTC-Lite 全般で利用するグローバル変数等の定義が書かれている。ユーザはこのファイルを書き換える必要はない。

C30_dsPIC30F3014Def.h は使用する MPU、コンパイラに依存する宣言をまとめて記述するファイルである。RT-Unit に従って作成した回路の場合は修正の必要がないが、それ以外の場合は、使用クロックの設定など、適宜修正する必要がある。なお、初期状態では、AD0(RB0)からの A/D データの取得と、RD0-3 の出力を制御することができるようになっている。

以上のソースファイルをプロジェクトに組み込み、コンパイルすることで、dsPIC 側の動作準備が整う。

7-3. XPort の設定

RT-Unit で利用している XPort の設定を行う必要がある。後述する Proxy コンポーネントは、サーバとして動作することから XPort をクライアントとして設定する必要がある。設定方法の詳細は、XPort の資料を参照していただきたい。
ここでの設定では、PC の IP アドレスを接続先アドレスとして設定し、ポートを 10001 に設定する。

7-4 Proxy コンポーネントの設定

Proxy コンポーネントのポート設定は Proxy.py である。このファイルの先頭にサーバとして待ち受けるポートの番号が書かれているので、ポート番号が 10001 となっているか確認する。

7-5. PC 側での準備

1. ネーミングサービスと RTC-Link などを起動する。起動方法などは、OpenRTM-aist のサイトを参照。

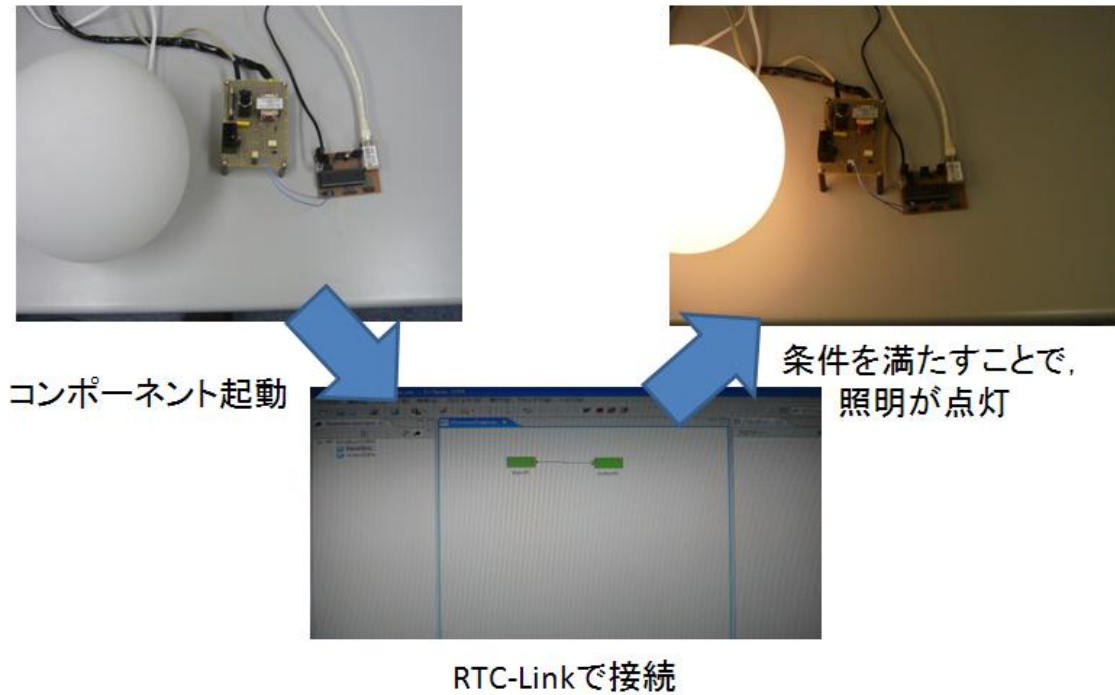
<http://www.is.aist.go.jp/rt/OpenRTM-aist/html/>

2. Proxy 側のプログラムを立ち上げる
メインファイル（サンプルは Proxy.py）をダブルクリック。
3. RT-Unit の電源を入れる。
4. RT-Unit が Proxy コンポーネントに接続できると、コンポーネントが起動する。

以上が RT Unit を動作させるための準備となる。次章において、動作の一例を示す。

8. コンポーネントの動作確認

動作の一例として、ポートから出力し、ライトを点灯させる RTC-Lite コンポーネント (Output0) を載せる。ライトを点灯させるハード要素の概要図は下記になる。Output0 コンポーネントは、onExecute 時に他のコンポーネント(この場合、Input0)からの入力データが敷居値を超えると、PIC のピンが high になり、ライトが点灯する。



9. 今後の課題

- コンポーネント作成スクリプトのリリース
- 16F877A 版のソース構造変更（dsPIC 版に合わせる）
- 新規 MPU 追加のための方法についての説明書作成
- 対応 MPU を増やす

10. 更新履歴

Ver. 1.0 全体的な内容について記述 2008/11/30