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．

1. 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．

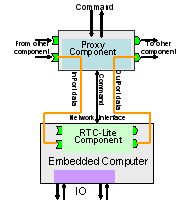
　RTC-Liteは，通常のRTコンポーネントと同じように利用できるようにするために，以下のような2つの要素から構成される．

・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．



1. 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．）

1. 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)

1. 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

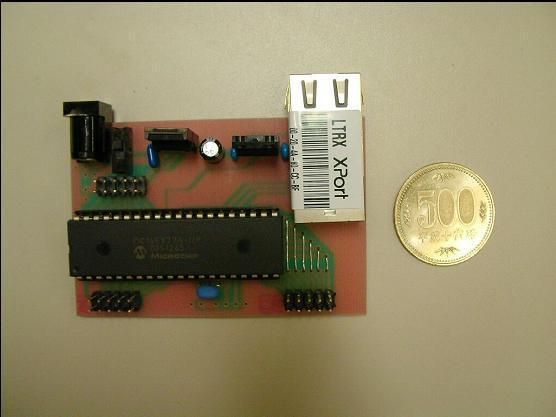
|  |  |
| --- | --- |
| OS | Windows XP |
| Python Version | 2.4.2 |
| OpenRTM Version | 0.4.2 |
| Language | Python |

1. RT-Unitの作成

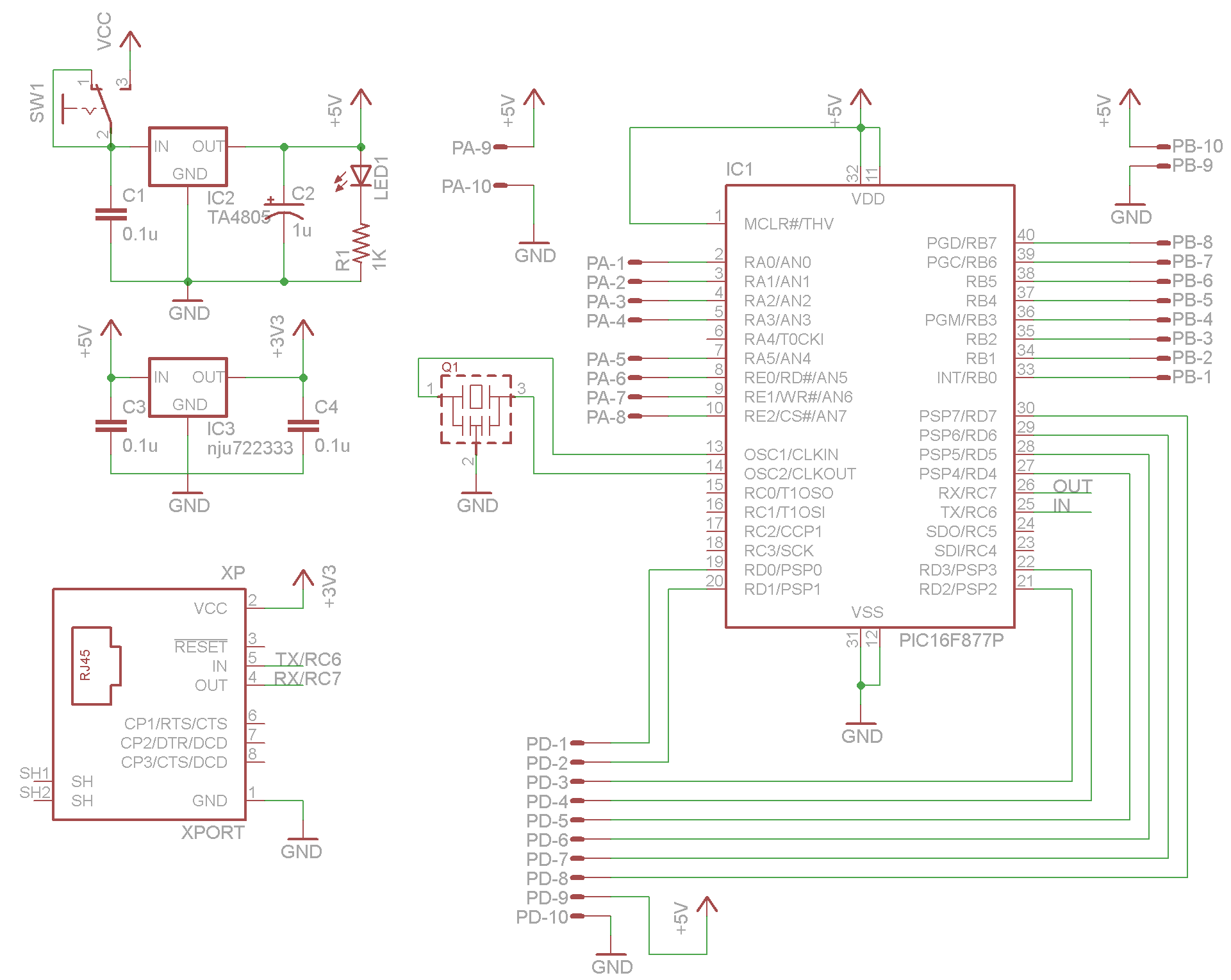
　RTC-Liteが動作するRTデバイス制御ボードとして，PIC16F877A，dsPIC30F3014を制御MPUに採用したRT-Unitを利用する．以降，このRT-Unitについて説明を行う．

6-1. PIC16F877Aを搭載したRT-Unit

1. 外観



1. 回路図



1. 部品表
2. PIC１６F877A版RT-Unitの部品表

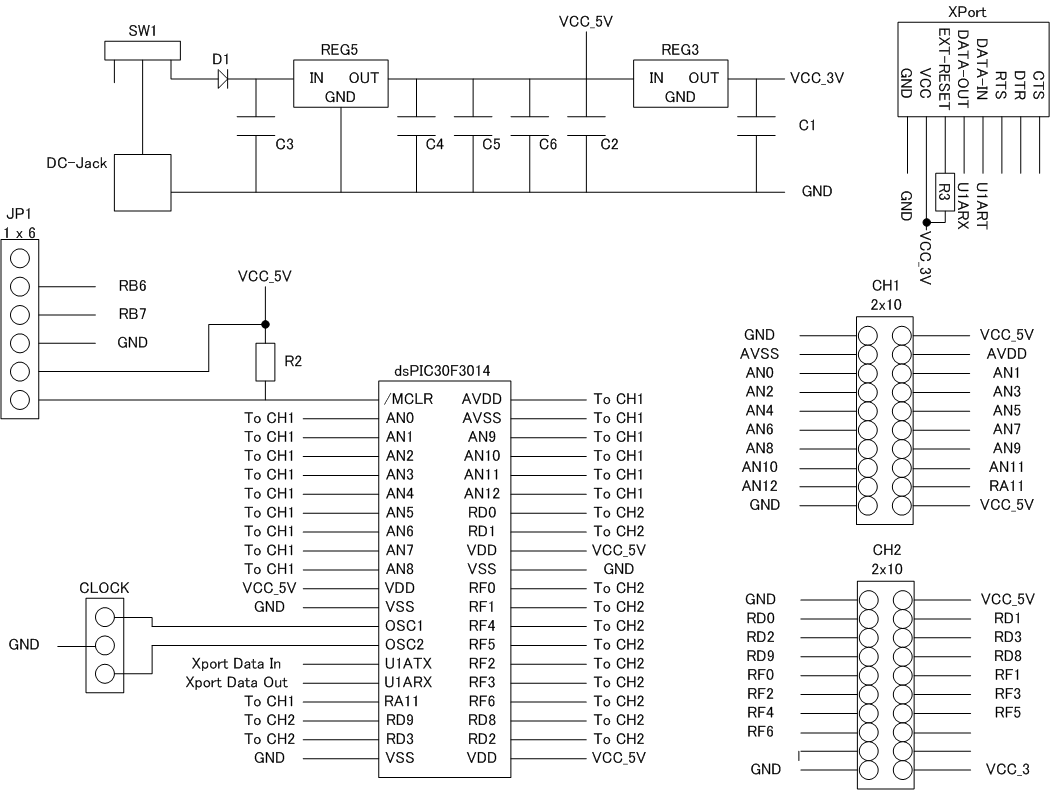
|  |  |  |  |
| --- | --- | --- | --- |
| 部品名 | デバイス | 型番、値 | 個数 |
| C1,C3,C4 | 積層セラミックコンデンサ | 0.1uF | 3 |
| C2 | 電解コンデンサ(φ5） | 1uF | 1 |
| R1 | 抵抗 | 1KΩ | 1 |
| Q1 | セラロック | 20MHz | 1 |
| SW1 | スイッチ |  | 1 |
| IC1 | PIC16F877A | 16F877A | 1 |
| IC2 | 3端子レギュレータ | TA4805 | 1 |
| IC3 | 3端子レギュレータ | nju722333 | 1 |
| PA,PB,PD | ピンヘッダ | 2 x 5 | 3 |
| LED1 | LED |  | 1 |
| XP | Xport | Xport | 1 |

6-2. dsPIC30F3014を搭載したRT-Unit

1. 外観



1. 回路図



1. 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(φ5) | 33uF | 3 |
| C4,C5,C6 | 積層セラミックコンデンサ | 0.1uF | 3 |
| R1 | 抵抗 | 330Ω | 1 |
| R2,R3 | 抵抗 | 10kΩ | 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 |

1. 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

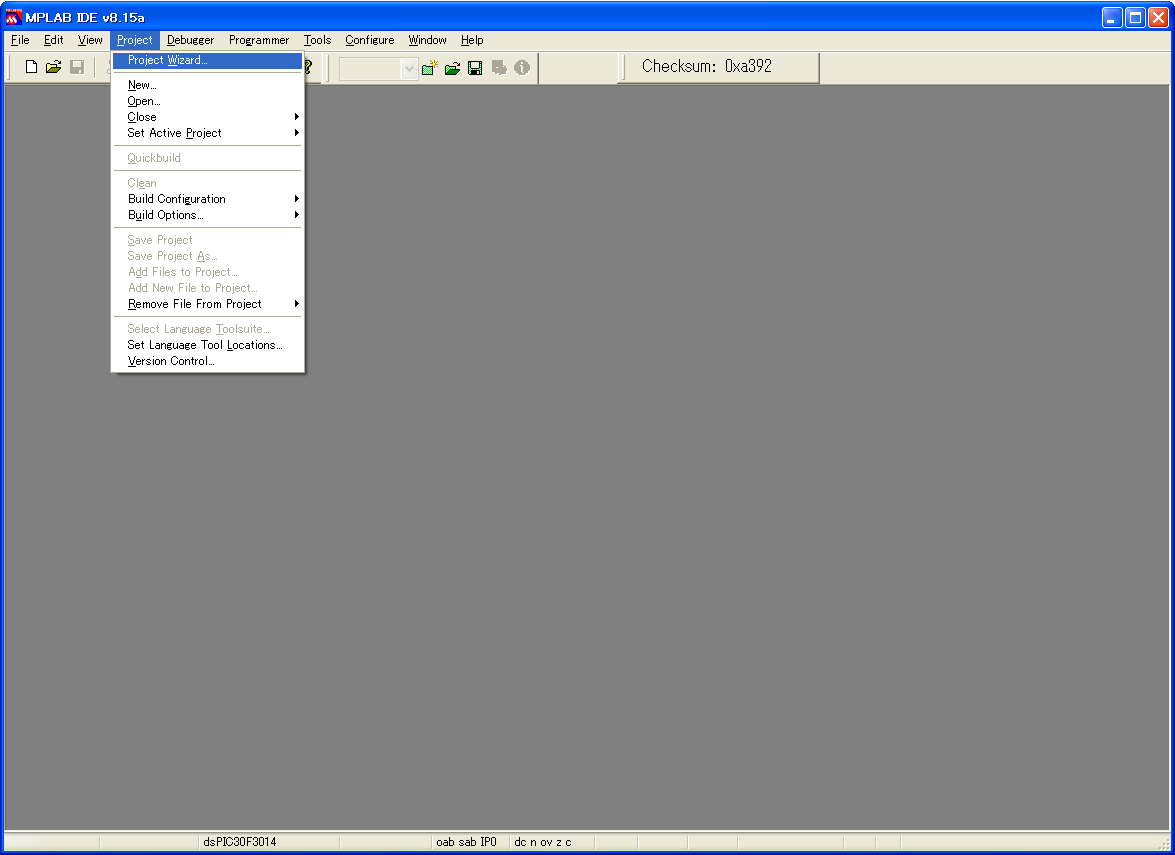
　PICをC言語で開発するためには，Cコンパイラが不可欠である．RTC-Liteでは，16系のPICの開発環境として，CCS社によって販売されているCCS Cコンパイラを利用している．そのため，16系のデバイスを利用するためには，CCS Cコンパイラを購入する必要がある．

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

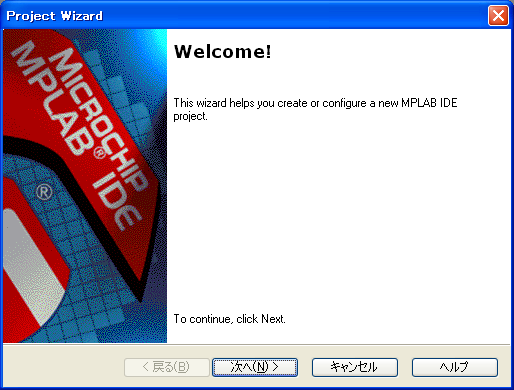
i) プロジェクトの作成

(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．

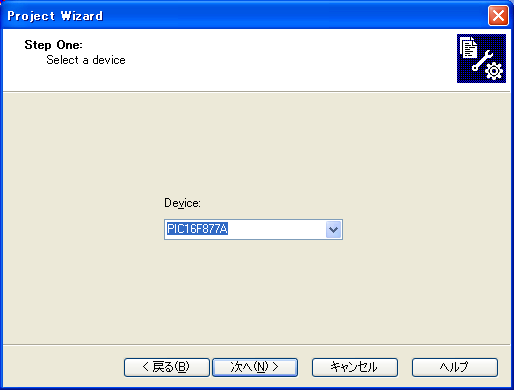
1. MPLABを立ち上げる．



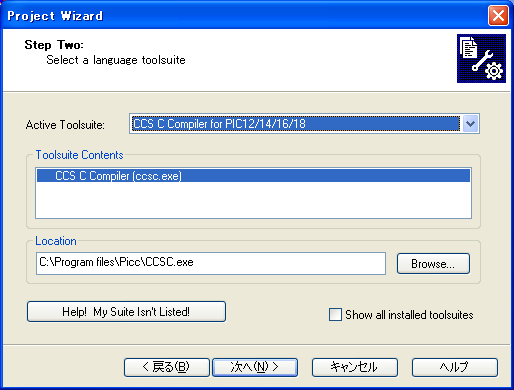
1. [Project] → [Project Wizard]をクリックする．



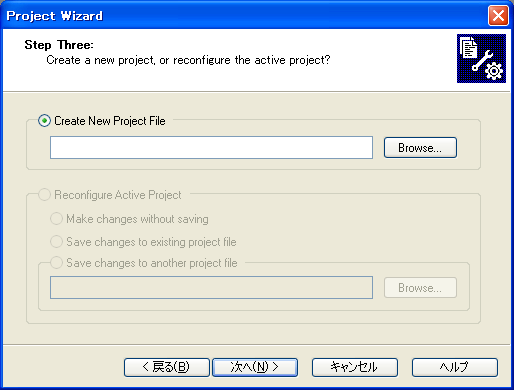
1. [次へ]をクリック．



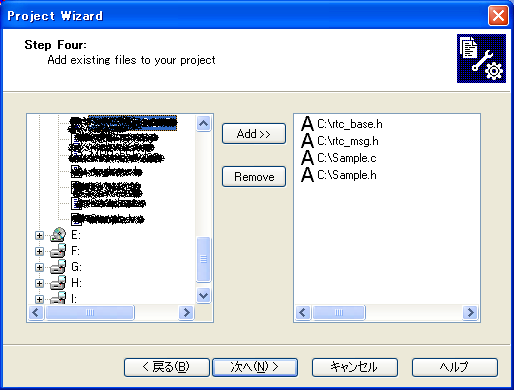
1. 16F877Aを選択し，次へ．



1. CCS C Compilerを選択し，次へ．



1. (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．



1. (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．

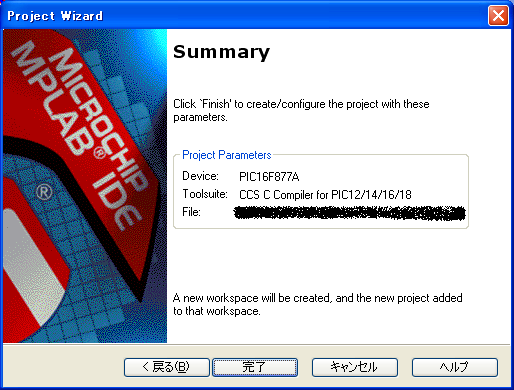
・Sample.c ・・・（メインファイル）

・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．



1. [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

1. 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&dDocName=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．

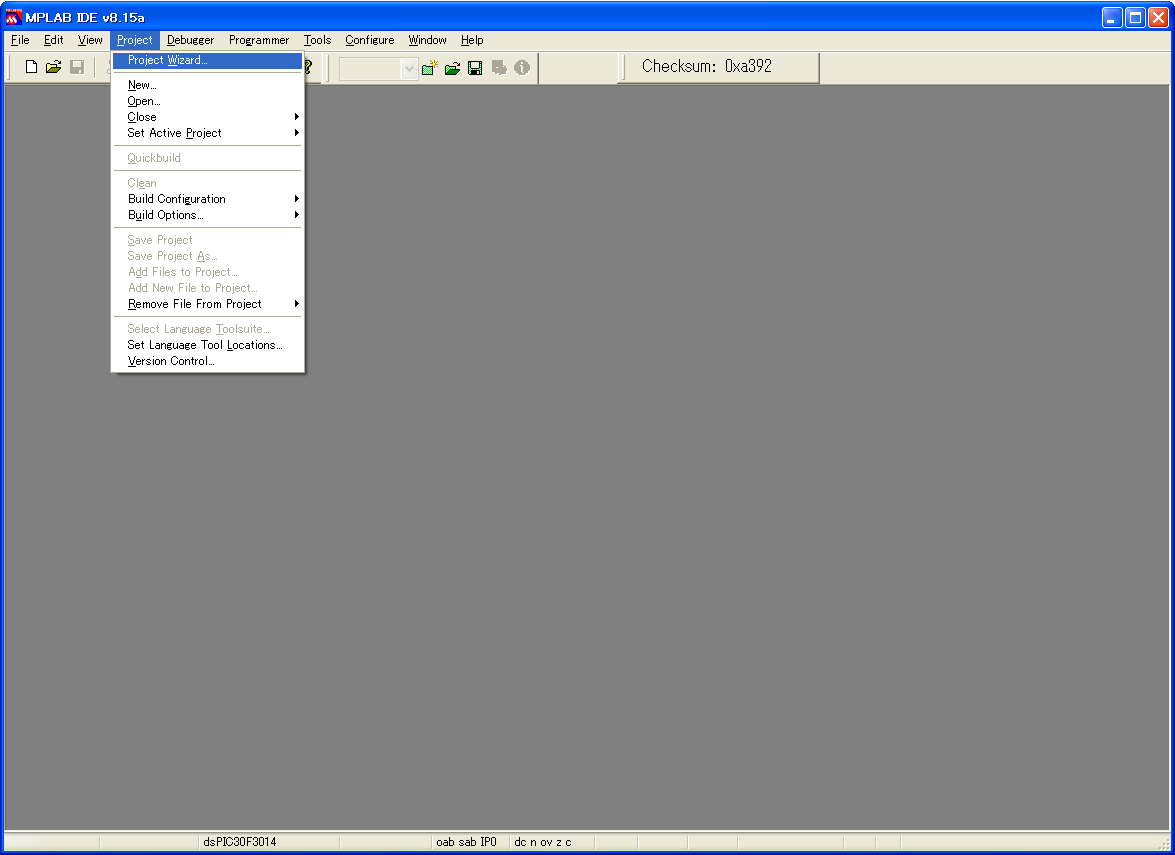
　PICへの書き込みを行うためのライタとして，Microchip社から販売されている，PicKit2を利用する．ただし，MPLABでは，PicKit2を利用しての書き込みが正常に行えないことから，PicKit2用の書き込みソフトウェアであるPicKit2 Programmerを利用する．

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

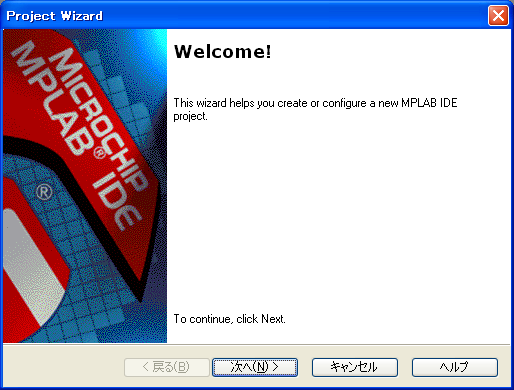
各ソフトウェアの使用方法については本マニュアルでは割愛する．

1. 開発用プロジェクトの作成

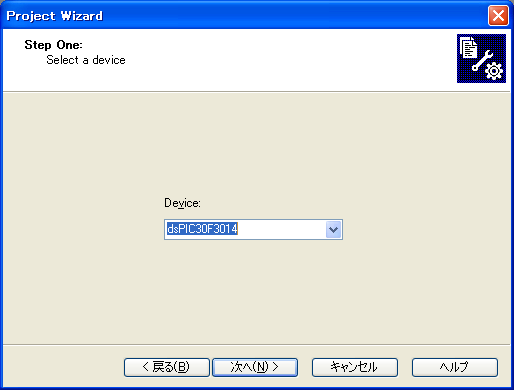
　RTC-Liteのプログラム開発を行うために，はじめに開発用のプロジェクトを作成する．以下に手順を示す．



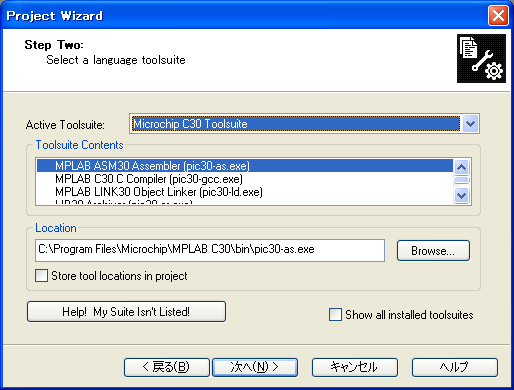
1. [Project] → [Project Wizard]



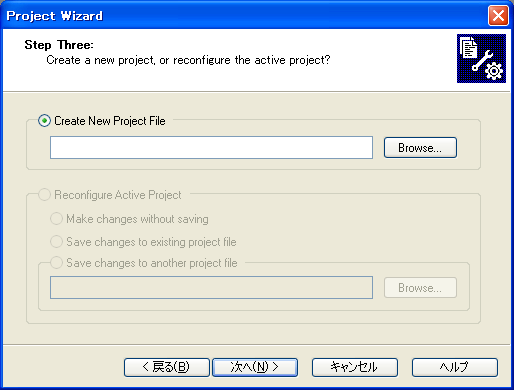
1. [次へ]をクリック



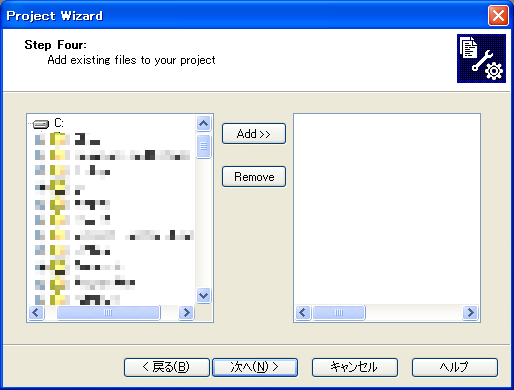
1. dsPIC30F3014を指定して次へ



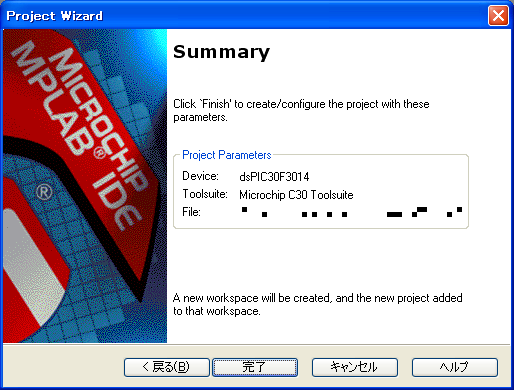
1. Active Toolsuiteを[Microchip C30 Toolsuite]にする



1. 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．）



1. [次へ]をクリック



1. [完了]を押すとプロジェクト作成終了
2. 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

1. 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．

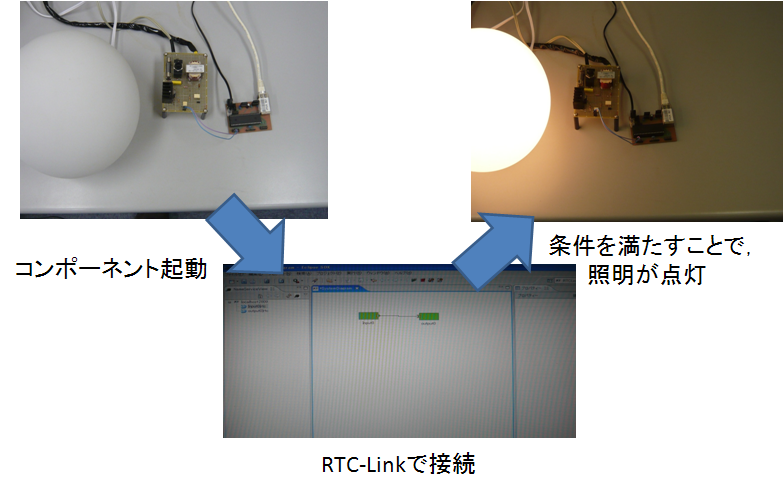
　３．RT-Unitの電源を入れる．

　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．

1. 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．



1. 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

1. 更新履歴

Ver. 1.0　全体的な内容について記述　2008/11/30