平成 24 年度文部科学省委託 「成長分野等における中核的専門人材養成の栽略的推進事業」

# スマートグリッドエンジニア育成教材

環境・エネルギー分野のスマートグリッドエンジニア育成の調査研究プロジェクト

はじめに

この研究報告書は、平成24年度に実施した、『環境・エネルギー分野のスマートグリッドエンジニア育成の調査研究プロジェクト』の中で、 教材開発を行った研究成果をまとめた報告です。

環境に優しい、再生可能エネルギーを効率的に利用する事は、現代の 大きな命題であり、そこで活躍する I T エンジニアを育成することは、 情報系専門学校にも大変意義ある目標となっています。この研究では、 身近な再生可能エネルギーを利用して、以下の事項に関連づけて、情報 系 I T エンジニアに必要と思われる技術を学ぶことができる教材の開発 を行いました。

◇ 発電(太陽光)

- ◇ 充電
- ◇ 通信
- ◇ 計測
- ◇ 制御(放電)

# 目 次

は	じ	め	に		. 3
目	次				. 4
1		再	生可	能エネルギーの定義と教材の対象	. 7
2		教	材シ	ステムの概要	. 9
	2		1	充電セクション	. 9
	2		2	負荷制御セクション	. 9
	2		3	汎用モニターセクション	10
	2		4	最終仕様	10
3	•	教	材シ	ステム開発(ハードウエア)	13
	3		1	Power $\vec{\pi} - \vec{F}$	14
	3		2	Main ボード	19
	3		3	簡易モニター	23
	3		4	Controller $\vec{\pi} - \vec{F}$	27
	3		5	Sensor $\vec{\pi} - \vec{k}$	29
	3		6	その他のパーツ	33
4		教	材シ	ステムの開発(ソフトウエア)	37
	4		1	簡易モニター	37
	4		2	Main $\vec{\pi} - \vec{F}$	54
	4		3	Android タブレット	70
5	•	テ	スト		75
	5		1	簡易モニター	75
	5		2	Main ボード	77
	5		3	Android タブレット	79
6		稼	働 実	験	82

6.1 充電テスト	82
6.2 放電テスト	82
6.3 PWM 制 御 実 験 8	83
6. 4 停電検出	84
7. 教材としての評価	86
8. 反省点	87
Appendix 1. 簡易モニター プログラムリスト	91
Appendix 2. Main ボード プログラムリスト10	02
Appendix 3. Android タブレットプログラムリスト12	29

# 1. 再生可能エネルギーの定義と教材の対象

#### 経済産業省・自然エネルギー庁では、次の様に説明しています。

http://www.enecho.meti.go.jp/saiene/renewable/index.html

現在わが国の主要なエネルギー源である石油・石炭などの化石燃料は限りがあるエネ ルギー資源です。これに対し、太陽光や太陽熱、水力、風力、バイオマス、地熱などのエネ ルギーは、一度利用しても比較的短期間に再生が可能であり、資源が枯渇しないエネル ギーです。これらは、「再生可能エネルギー」ともいわれます。石油等に代わるクリーンなエネ ルギーとして、政府はさらなる導入・普及を促進します。



#### 自然エネルギー庁 HP

上記で定義される再生可能エネルギーを教材として利用しようとす る場合、次の様な条件を考慮する必要があります。

◇設備・機器の設置が容易

◇小規模導入が可能な事

- ◇実験が容易
- ◇価格が安価
- ◇ 壊れにくい (機械・機構部分が無い)

身近な再生可能エネルギーを考えると、太陽光、水力、風力エネルギー が手の届く材料として有望ですが、機構部分がなく設置が容易で、手軽 に導入でき電気エネルギーに変換しやすい事から、この研究では発電装 置に太陽光発電パネルを採用しました。



採用した太陽光パネル

# 2. 教材システムの概要

教材システム開発にあたり、システム全体像を次の様に構想しました。



教材システムの構想

#### 2.1 充電セクション

太陽光発電パネルで発電した電気エネルギーを、充電コントローラ を通じてバッテリーに充電します。充電コントローラでは、バッテリー の電圧を測り一定以上の電圧になると充電を停止、一定以下の電圧で充 電開始となるよう、マイコンを使用した閾値制御を行います。(充電モニ ター)

# 2.2 負荷制御セクション

バッテリーから電力を供給して、インバータを介して AC100V 家電 品を駆動する制御をおこないます。この際、負荷に流れる交流電流を測 定して、使用電力量を計算しモニターに表示します。このセクションも マイコンを使い通信、センシング、表示等をおこないます。

#### 2.3 汎用モニターセクション

充電セクション・負荷制御セクションでモニタリングしている情報 を Bluetooth 通信で、タブレット PC に送り、手元でモニタリングが可 能とします。

タブレット PC は、OS として Android を使用したものが手元にあり ましたので、それをそのまま利用して開発することとしました。既に開 発済みの Bluetooth 通信プログラムを改造して、今回の目的にかなう汎 用的なモニタリングシステムに適応出来る様に考えました。

#### 2.4 最終仕様

上記構想を草案として、IT 教材としての検討を行い、最終的なシステム仕様を決定しました。



当初構想では、スマートメータを意識してモニタリングを主としたシステム開発の計画でしたが、教材として検討した結果、次に示す様

な基板構成としました。大きな特徴としては、

- ◇ Main ボードは、Power ボードの上段に配置しいくつかの SW により操作ができます。
- ◇ Powerボードは、ACアダプタとバックアップ乾電池の 2Way 電源 とし、AC 電源が途絶える停電を検出する機能を持ち、乾電池バッ クアップに切り替わり、自動的にバッテリーから負荷に通電する。
- ◇ 汎用モニターがない場合や Bluetooth 通信が出来ないときでも、簡 易モニターで操作ができるようにしました。当初の構想では Bluetoothの通信が2系統必要でしたが、タブレット PC の仕様か らこれは難しいので、Main ボードがまとめて1chの Bluetooth 通 信を行う事としました。
- ◇ Main ボードは、Bluetooth 通信と同時に、シリアル通信を行い簡 易モニターとの間で情報のやりとりと、簡易モニターの SW による 操作指示を Main ボードに指示します。IT 教材としては、Bluetooth とシリアル通信の 2 つの技術を学ぶことが出来ます。

※シリアル通信のケーブル(またはコネクタ)は、PC のシリ アルポートに接続して、ターミナルソフトで通信のモニターとテス トが出来る様になっており、今回の開発でも、そのようなコネクタ を作り、PC で通信確認テストをおこないました。

◇ 汎用モニターは、簡易モニターが無い場合でも、全てのモニタリン グと操作が出来るように Android 側のシステムを作り込みます。 組込系マイコンシステムでは、小さな表示器(この研究では 8 文字 ×2 行の液晶表示器) や LED 等による表現方法が採れますが、タ ブレットの表現機能を使う事で、情報システムとしての機能が向上 します。また、一般的に Android タブレットは、JAVA による開発 環境が無料で公開されており、開発環境の構築も容易です。

- ◇ Main ボード CPU は、PIC24FJ64GB002 を使用して、C 言語によ り開発します。これは、Bluetooth 通信で使用出来る様々なライブ ラリ群が CPU の製造メーカ Microchip 社から公開されており、開 発環境も MPLAB IDE として無償公開されている環境が使えるた めです。インターネット上に情報も多く存在するので、システム開 発に不安がありません。
- ◇ 簡易モニターCPU には、PIC16F684A を用います。開発環境は、 Mainボードと同様の環境でも開発出来ますが、情報 IT 系専門学校 の教材として位置づける意味から、手元に既にある mikroBasic Proを使うこととしました。このシステムを開発するにあたり、
  - ①. Basic 言語による組込マイコン開発(簡易モニタ)
  - C 言語による組込マイコン開発(Main ボード)
  - JAVA によるモバイルシステム開発(汎用モニタ) が学べます。

ーからシステムを開発しなくても、部分的にプログラムを改造・ 改善する事で、細かな学習教材として使用する事が出来ます。

◇ PWM 制御の実装

放電制御の部分では、単にバッテリーからインバータに電源供給す る事は、スイッチングだけの制御で容易過ぎておもしろみに掛けま す。スマートグリッドエンジニアに不可欠の技術として、 Main ボ ード CPU のタイマー機能をフルに使った PWM 制御を実装しまし た。DUTY 比 0~95%の範囲で切換が可能で、0~80%は 10%刻み、 80~95%は 1%刻みで制御実験が行えるようプログラムの設計を しました。

# 3. 教材システム開発 (ハードウエア)

このシステムでは、Bluetooth 通信を行う事が、汎用モニタ・Main ボ ード CPU の大きな命題です。システムを開発するにあたり、既に開発 済みのマイコン基板と、手持ちのタブレット PC、Bluetooth ドングルに ついて、Google 社の Android ライブラリ、マイクロチップ社の Bluetooth 通信ライブラリが使えるかどうか実験を行いました。



実験基板(表)



実験基板(裏)



実験用 CPU ボード

実験基板(完成)

完成した実験基板に、簡易プログラムを書き込み、Android タブレットも、実験用プログラムを作り、基板上のSWをクリックすると、Android タブレットの画面表示が変化し、タブレット側をタップすると、基板上のLEDが点灯する事を確認しました。

この実験で、手持ちの Bluetooth ドングルとタブレットが使用出来る 事が分かりましたので、システムの本格的な開発に入りました。 以下、この研究で開発した教材システムのハードウエアについて報告します。

### 3.1 Powerボード

◆設計

このシステムでは、マイコン(電源 3.3V:計測用基準電圧を兼ねる)、 シリアル通信レベル変換 IC(電源 5V)、ソーラーパネル・インバーター・ 12V バッテリーの電源系をスイッチングする FET の制御などが必要な ために、Power ボードでは 5V・3.3V の DC 電源を作ることとし、周辺 基板に電源を供給します。また、停電検出を行う構想があるので、単三 乾電池4個(6V)と 9V1.2Aの AC アダプタという2 WAY 電源としまし た。

停電検出の機構は簡単で、電池電源と 9V のアダプタ電源を、逆流防止用ダイオードを介して並列に接続します。このとき 9V のアダプタ出 力をトランジスタの Baseに接続してトランジスタを駆動しておきます。

停電すると、9V と 6V の電圧比較で低い方に電源が切り替わります。 同時に、トランジスタの Base 電流が途絶えるので、この出力をマイコ ンのデジタル入力で読み取ると、通常時は Low レベル(=0)、停電時は High レベル(=1)と読めます。この信号をマイコン側ソフトで常時監視す れば、停電時を検出して、必要な処理を行いつつ、各通信先に伝達出来 ます。

Main ボードの面積の関係で、シリアル通信用レベル変換 IC も Power ボードに載せることとしました。このために、Power ボードと Main ボ ードの間は、6Pin のヘッダ・ソケットで接続し、信号内訳は

①. GND

2.5V

③. 3.3V

④. 停電検出信号

シリアル通信 Tx (送信側)

⑥. シリアル通信 Rx (受信側)

となりました。

※ GND は、全てのソケットなどで位置を統一し、基板に向かって右側 端の Pin が GND になるように配置しました。



Power ボード回路図

◆製作

Powerボードは、電源を作り出すことが大きな仕事です。基板に中 心となる電源 IC を配置して、AC アダプタやバックアップ電池空の電源 電圧を安定化させた 5V 電源にします。この 5V を使い 3.3V を作り、Main CPU の主電源とします。5V 電源 IC は、乾電池駆動が可能な定損失レギ ュレータを使います。



DC5V 電源 IC



RS232c レベル変換 IC



DC3.3V 電源 IC



16PIN IC ソケット



汎用基板



3ポイントコネクタ



電 解コンデンサ



ノイズ対 策 用 セラミックコンデンサ

Power ボードは、上層の Main ボードに電源供給しながら、シリアル 通信の送受信信号の受け渡しを行いますので、 Pin ヘッダとソケット使 い Main ボードと接続します。また、USB ケーブル接続の出来る乾電池 ケースを入手したので、miniUSB コネクタの受け側コネクタを、基板に 取り付けます。



接続用ピンソケット



miniUSB コネクタとヘッダ



乾 電 池 ケース外 観



USB コネクタと SW が見える



miniUSB コネクタ準 備



ピンソケットの加 エ

写真にあるように、パーツを購入しても、そのまま使える物は少ないの で、準備の為に半田付けや加工が必要になりますが、安価な教材開発に は不可欠な仕事です。これを旨くこなしておくと、後の製作が楽になり ます。



完成した Power ボード

Power ボードのシリアル通信ケーブルは、三つ編みにしてノイズ低減 を図ります。RS232C レベル変換 IC を使用していますので、通信ノイズ は低減されますが、それでも安心の為に GND を含めた線を撚り線にし ました。実験したところ、この先に 3m ほどの LAN ケーブルを繋いでも 問題無く動作しました。

#### 3.2 Main ボード

Main ボードは、このシステムの中枢を担う部分で、16bit マイコン (PIC24FJ64GB002)を用いました。この CPU は、USB コントローラ を内蔵していて、CPU メーカから USB に装着して使用する予定の Bluetooth 用ライブラリなども豊富に提供されています。

◆ 設 計

Main ボードは、外部との接続は、下層の Power ボードとのピンヘッ ダ・ソケットによる接続だけです。主要な部分は CPU なのでこれを中 央に配置し、ソケットで抜き差し出来る様にします。プログラムの書き 込み用に『ICSP』という I/F が内蔵されているので、CPU を基板に装 着したまま内蔵フラッシュメモリのプログラムを書き換えることが出来 ます。これを使う為に、ピンヘッダをつけておきます。(回路図上の PICKit3)

単体でテストが出来る様に、Bluetooth テスト用、Drive テスト(放 電)用、Charge(充電)テスト用それぞれに3つのSWとLEDを配置 します。通常稼働時に、CPUの負荷の具合が分かるように、HB(ハー トブレーク)LEDを点灯します。このLEDが長く点灯していると、CPU の処理が忙しいことを示します。

19



Main ボード回 路 図 ---

◆製作

Main ボードに必要なパーツを写真に示します。基板取り付け用の USB コネクタが市販の Bluetooth ドングルを差し込むソケットになり ます。



LED と電 流 制 限 用 抵 抗



基 板 取 付 用 USB コネクタ



タクト SW



# 基準電圧調整 VR



始めに大間感パーツを基板に載せて、位置決めを行い、マジックなど でマークします。

この基板には、USBコネクタがつきますが、固定用のツメが入るよう に、穴加工を行う必要があります。写真の様なハンドビットと呼ばれる 工具とドリルで、穴空け加工します。



パーツの位 置 決 め



穴加エ



できあがった Main ボード

上の写真の右下の部分に基準電圧を作る回路がありますが、今回の計測には CPU の電源を基準としたので、未使用となっています。

下層の Power ボードとピンヘッダ・ソケットにより重なっています。

#### 3.3 簡易モニター

簡易モニターは、Main ボード周りとは別電源で単独駆動が出来る様 に、006P(9V)乾電池を電源に使います。情報表示器に 8 文字×2 行のバ ックライト付液晶表示器を取り付けます。この表示器は白抜き文字(反 転文字)がバックライトで浮かび上がる仕組みになっていますので、屋 外など明るい場所での文字表示に適していますが、表示文字数が少ない ので、ソフトウエア開発時には、少し工夫が必要です。

またこのユニットで使う CPU は PIC16F486 という 8bit マイコンで、 シリアル通信ポート (UART) を 1ch 持っていますので、通信時の処理 を割り込みで記述することが出来ますので、組込系マイコンシステム開 発の良い教材となります。

◆ 設 計

回路図を示します。



簡易モニター回路図

簡易モニターにも LED×2 個、タクト SW×2 個を取付けて、システム 全体の操作ができるようにしています。 特徴として、この CPU はデータ EEPROM を持っていますので、バッ テリーの上限電圧・下限電圧の閾値を Main ボードからの指示で書き込 み、個別のコンフィギュレーションが出来る様にします。 Main ボード で使用している CPU には、このデータ EEPROM の無い安価なモデルを 使用していますので、 簡易モニター CPU のシリアル通信機能と EEPROM 機能を充分活用した設計になりました。電源は、9V から 5V を作り出す電源 IC を使います。消費電流は CPU が 4MHz 駆動時で 0.9mA、LED1 個で数 mA、液晶表示器で 70mA ですので、シリアル通 信用レベル変換 IC の電源を取っても充分乾電池駆動できます。

◆製作

使用したパーツを示します。



小型汎用基板



006P9V 電 池 スナップ



5V 用 電 源 IC

電 源 IC 用 電 解 コンデンサ





電 源・RS232C 用 セラミックコンデンサ

LED 電流制限用カーボン抵抗



モニタ操 作 用 タクト SW



LED 緑・赤



液晶表示器と構成パーツ

抵抗は、バックライト用



液晶表示器コントラスト調整用 VR

# 金属パーツはコンタクトで、信号線をカシメて使用



通信ケーブル用コネクタ



RS232C 用 IC とソケット



簡易モニター用 CPU 648A

液晶表示器は、裏側にバックライト用の電流制限抵抗と、基板接続用 のピンヘッダを半田付けします。基板側には、付属のピンソケットを取 り付けます。LED は、電流を沢山流ば明るく光りますが、その分寿命が 短くなりますので、電流制限抵抗で必要最小限の明るさにします。

タクト SW で、ON/OFF を判断しますが、使用する CPU の I/O ポー トは、内部プルアップが可能ですので、外部にプルアップ抵抗をつける 必要はありません。これは Main ボードも同様にしています。

完成した簡易モニターを示します。



簡易モニター(表)



簡易モニター(裏)



液晶表示器(裏)

簡易モニター(完成)

#### 3.4 Controllerボード

このユニットは、大きな電流が流れますので、電力系統の配線には 1.25 φ の家庭で使う家電品の電源ケーブルと同じ電線を使います。マイ コンとの信号線は、通常の細い線で構いません。これまでのユニットに 比較するとパーツも少なく作りやすいはずです。

♦ 設 計

電源電圧を基準にして AD 変換してバッテリー電圧を計測するために、 12V を分圧して、最大でも 3.3V 未満にする部分を組み込みます。

Main ボードの指示により、充電 ON/OFF、放電 ON/OFF をスイッチ ングするために、2 つの FET を使います。

設計した回路図は次のとおりです。



Controller ボード

FET の Gate 側に付いている抵抗(22KΩ)は、システムが通電して Main ボードの信号が安定するまでの間に、FET が確実に OFF する様に プルダウンするための抵抗です。このことにより、不安定な状態で不用 意に負荷が駆動される事がなくなります。

♦ 製作

使用したパーツを示します。



逆流防止ダイオード



負荷駆動用 FET



ターミナルブロック



充電制御様FET



Main ボード接 続 用コネクタ



大電流用配線



分圧用抵抗



小信号用配線



事前の配置検討



完成した基板(表)



完成した基板(裏)

#### 3. 5 Sensor ボード

このユニットは、AC 電流センサーで駆動中のインバータの出力電流 を計測します。AC 電流センサーで検出した電流に対応する電圧を Main ボードに送り AD 変換して計測します。また、電流センサからの出力電 流を整流して DC 電流とし、その電流をオペアンプを介して、アナログ 電流計に送り、メータ表示します。

◆設計

回路図を示します。



Sensor ボード

AC 電流センサーにはコイルが内蔵されています。電源線にクランプ して、誘導電流による検出を行います。検出される電流は僅かなので、 抵抗を介して電圧にして、それを Main ボードの AD 変換ポートに送り ます。アナログメータによる表示は、システムとしては特に重要な物で はありません。同じ事が簡易モニターと汎用モニターで表示できます。

◆製作

このユニットで使用するパーツを示します。汎用基板・ターミナル ブロック・整流用ダイオード・電圧調整用 VR・電流調整用 VR・コンデ ンサ・抵抗・オペアンプなどです。



Sensor ボードパーツ



Sensor ボード(完成)

このユニットには、AC 電流センサーとアナログメータが取り付けら れますので、これらのパーツも加工します。 電流センサーは、コネクタが付いていますが、それを取り外し、リード 線を延長して、ターミナルブロックに接続出来ように加工します。 その際、リード線の延長部分には、熱収縮チューブなどで、絶縁加工し ておきます。



AC 電 流 センサー



クランプ部分



コネクタをカット



延長リード用巻き線





コネクタをカット



適当な長さを準備



より線 ハンダメッキ 絶縁 チューブ加工



絶 縁 用 熱 収 縮 チューブ



被覆除去



対にして固定



アナログメータ(表)

# アナログメータ(裏)



メータリードの製作







メータリードの取付

メータリードの製作 メータリード(完成) このアナログメータは、フルスケール 100µA の物ですが、今回フル スケール 50A として使用しますので、調整用ボリュームでスケールに合 せます。メータはカバーが分解できるので、メモリ板を書き換えるなど して、読みやすいようにします。

# 3.6 その他のパーツ

◆バッテリー用接続リード

Controllボードのターミナルブロックに 12V バッテリを接続するた めの、リード線を作成します。鰐口クリップでバッテリに接続出来る様 にします。



鰐 ロクリップを 1.25 ¢ の線 に半 田 付 けして作 成

◆インバータの改造

150W 級の自動車用インバータを、Controll ボードに接続出来 るようにリード線をとりつけます。



放熱ファンの付いたインバータ



裏側のネジをはずし分解します







コネクタ部 分のストッパーを取り外す



シガーライターソケット部分を分解



ソケット部分のコネクタを取り外し、リード線を半田付け



ビニールテープで絶 縁 する



ターミナルブロック側 に+マーク



PWM 制 御 実 験 用 DC モータのリードを延 長 加 エ

◆太陽光発電パネルのリード作成

バッテリリードと同様の物を作成し、太陽光パネルを接続する。

◆12VDC モータの準備



完成した、システム全体(ハードウエア)は、次の様になりました。
### 4. 教材システムの開発(ソフトウエア)

# 4.1 簡易モニター

◆開発環境

簡易モニターの CPU は、米 Microchip 社のマイクロコント ローラ【PIC】です。OS は有りませんので、組込システムとし ての OS の役目をする main プログラムを何も無い状態から作 り込む事になります。

開発言語は、MikroElectronika 社の『mikroBasic Pro for PIC』を使用しました。この環境は IDE(Integrated Development Environment)を含み、使用するマイコンの種 類を選択するだけでプログラムの雛形を作ってくれる『フレー ムワーク』機能があります。基本的には、ユーザーが main 関 数を作り込めば良く、言語も理解容易な Basic です。大変多く のライブラリが含まれていて、システム開発の工数削減に大い に役立ちます。筆者の開発デスクには既にこの環境が整った PC がありましたので、これを使いましたが、最新の環境は、 MikroElectronika社の HP からダウンロードできるようになっ

ています。

ダウンロードしたファイルをインストールすれば、プログラ ムサイズの制限はありますが、重敏な機能の開発環境が整い、 Free で使用できます。いろいろな種類の PIC に対応していま すので、開発を多く手掛けるのであれば、しばらく Free で使 い機能を見定めてから購入するとよいでしょう。IT 系専門学校 の教材にはもってこいの環境です。



mikroBasic の HP

HPを開くと、mikroBasic Pro for PIC の画面が表示されています。 これを少しスクロールすると、Download 画面がでてきます。

	Specification	LOF LIN	aries Hy Ford Project	
Your	aic PRO for PIC compiler PIC best friend			Download
mikroBasic PRO for for everyone. Popu syntax and clear or mean a lot when yo ready-to-use librar	PIC is a full-featured Basic compl ar basic programing language is it de. Comfortable IDE with a compr si are making your first steps into functions and examples that will	er which makes Microchip te best choice for beginner ettensive help file, and a fi world of <i>PEC</i> microcentrolit help you in your developm	B PIC development suitable s because of the simple real lifetime technical support set. Not to mention over 500 ent.	compiler version 5.8.0 Download and the softwares -FOR FREE Software in equipand with high functional demo lumena with up to 20 of program works of software code area which can be just anough for simple applications.
Begin	does all the work for you ners are experts	Enjoy v	while working	Save your time!
Compiler is designe you can rely on it to four levels of optim code size up to 20%	d to be smart and efficient, oo o do the hard work. It features sations that can reduce your k.	User is our main conc center of attention, w intuitive, fast and rich comfortable while wo	ern. Having the user as the e developed (he best IDE) i in features. You'll feel very riong.	What's the point of compiler if you have to write (branies from scratch? With our compilers, you' have over 500 library functions, and a head sta in development.
	and a second	State State	Why d	hoose this compiler?
	· Marine Street Land In the Street Street	ALALANA	OneTime**	payment
And			S (2)	Compiler is constantly improved and new functionalities are added with each new release. Owners of compiler licenses are entitled to free upprades, which take just a
American Status         Status <t< td=""><td></td><td>minute of your time. microBasic PRO for PIC license guarantees</td></t<>				minute of your time. microBasic PRO for PIC license guarantees
territori		And plane		you free product Metime technical support, as you can rely on our help when developing.
		An one way to an example of the second	-	your prototype.

#### Dowload 画面

Download ボタンをクリックすると、以下の確認ウインドウが表示さ れますので、保存を選択し保存先のフォルダを適宜選択します。サイズ は 26MB ほどです。無線 LAN 環境でも容易にインストール用のファイ ルを入手できます。



mikroBasic o Download

Download が終了すると、mikrobasic\_pro\_pic\_2012\_v580.zip (この 報告書作成時の最新版)が保存されていますので、このファイルをダブ ルクリックしてインストールします。表示されるメッセージに従い進め れば容易に環境が構築出来ます。

※PCのOSは、Windows VistaとWindows7(64bit)で確認しています。 Free版で使用するとプログラムサイズに制限がありますが、簡易モニ ター程度であれば充分開発出来ますし、他のシステム開発にそれほど困 ることはありません。大きなサイズのプログラムを作成することになっ たときに、ライセンス購入すれば良いでしょう。



今回使用した mikroBasic

今回使用した環境は、過去の研究事業で 実績のある Ver.4.60 の製品版です。使用す る PIC も網羅されていて、申し分ありませ ん。

既に開発済みの簡易モニタプロジェクト表示を示します。



このウインドウの右側に Library Manager というタブがあります。 このタブをクリックするとタブが広がり、含まれている多くのライブラ リが確認できます。ADC、CAN 通信、ボタン、LCD、シリアル通信、 LCD・・・など、数多くのライブラリが含まれていることが分かります。



### 含まれている Library

Bile Ext Verw Project Blud Bun Doos Help         Image: Set Verw Project Blud Bun Doos Help         Image: New Proj	mikroBasic	Pro for PIC v.4.60.0.0 - C:¥Users¥k	en¥Documents¥monka_it¥serial_monitor_1st.mbppi	and the second second second second		×
Weild worker, Linkase         • Main program         • Sold = *1**         • Main program         • Sold = *1**         • Vision program	Eile Edit Vie	ew Project Build Run Tools	Help			
Image: State and State an	i i i i i i i i i i i i i i i i i i i				Library Manager	-m 🔽 🗖
Image: State in the state		unitur_istantibas				
Image:	Toje .					
Main program       # # # # # # # # # # # # # # # # # # #	\$ · E	main:			E Conversions	
Main program Main program Ma	÷ .				E C_Type	9
350       mode = "N#"       # # # # # # # # # # # # # # # # # # #	8 .	' Main program			EEPROM	2
Image: Second	350	mode = "N"			EPSON_S1D13700	
Berger	- ·	rxdata = "			Cied Easts	
ggg       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i stidex = 0       i rirreg = " "       i rirreg = " "         i rirreg = " "       i rirreg = " "       i rirreg = " "         i rirreg = " "       i rirreg = " "       i rirreg = " "         i rirreg = " "       i rirreg = " "       i rirreg = " "         i rirreg = " "       i rirreg = " "       i rirreg = " "         i rirreg = " rirreg = " "       i rirreg = " "       i rirreg = " "         i rirreg = i rirreg = " ri	de •	rxbuf = "			Keynad4y4	000
Image: Second	- E	rxreg = " "			⊞- V Lcd	2
** Clear Port A and B     ** Modeler     *********************************	ore ·	rxindex = 0			Lcd_Constants	
Stee // Class Port A and B     Ore_Wre     Dorta-0     Porta-0     Porta-					😟 🗹 Manchester	
PortBolo	356	'' Clear Port A and B			🕀 💟 One_Wire	
PortDirections     PortDire		porta=0			🕀 🔽 PS2	0,00
360       * PortDirections       * * 8848         CMCON * \$00000111       * * 8548       * * 8548         TRISA * \$0010100       * USART EAUD RATE Setting. \$600 baud       * * 8548         RCSTA * \$1001000       * USART FX Status and Control       * * 8540         OFTION REG. ? • 0       * Pull up Port B for \$%       * * * * * * * * * * * * * * * * * * *		porce-o			PWM	
Policie Lithing     P	260	/ RestDirections				
Image: Status and Control		CMCON = \$00000111			Software SPI	2
IRISB = \$00001111     IRISB = \$00001111       SPBR = 25     ' USART EAUD RATE Setting. \$600 baud       SCSTA = \$1001000     ' USART EAUD RATE Setting. \$600 baud       OTTON REG. 7 = 0     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$0000101     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000001     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000001     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000001     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000001     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000000     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000000     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000000     ' USART EAUD RATE Setting. \$600 baud       IRISB = \$000000     ' USART EAUD RATE Setting. \$600 baud       IRISE = \$000000     ' USART EAUD RATE Setting. \$600 baud       III Metsoges IRISE. *     UWART IRIS       III met     Message Text       III met     Message Text       III Net     UWART IRIS		TRISA = %00111100			Software UART	2
SEBSE - 25 · USART BAUD RATS Setting, 960 baud RCSTA = \$1001000 · USART RX Status and Control V USART RX Status and Control OFTIONERC. 7 = 0 · Pull up Port B for SN Messages Quak Converte Prove Message No. Message Text Unit Message Text Libra ray OF Help		TRISB = %00001111			B Sound	
RCSTA = \$1001000     ' USART Rx Status and Control     TXSTA = \$0010010     ' USART Rx Status and Control     TXSTA = \$0010010     ' USART Tx Status and Control     O' TFT     O' TTT      O' USART Rx Status and Control      O' ' TTT      O' ' ' TTT      O' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '		SPBRG = 25	' USART BAUD RATE Setting. 9600 baud		🗈 💟 String	Ē
TXSTA = \$0010010     'USABT T& Status and Control     OPTION_REG.7 = 0     'Pull up Port B for SW     ''     ''     '''     ''''     ''''''	-	RCSTA = %10010000	' USART Rx Status and Control		ф- 🗹 Т6963С	
OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      OPTION REG. 7 = 0     Pull up Port B for SW      Pull up Port B fo		TXSTA = %00100110	' USART Tx Status and Control		IFT 💟 🕀	3
Insert Modified      C:¥Users¥ken¥Documents¥monka_it¥serial_monko_ist.mbas      LibrarrO Help	•	OPTION_REG.7 = 0	' Pull up Port B for SW		Time	
C:¥Users¥ken¥Documents¥monka_It¥serial_monker_1st.mbas	٠				H- V Irigon	<b></b>
Verros Ve	III Message	es 🔤 Quick Converter			U UART	
Line Message No. Message Text Unt UNT I Read Une Message No. Message Text Unt UNT I Read UMATI J Dide UMATI J DI UMATI J DI DIDA UMATI J DI	Errore	Warnings W Hints			UART1_Init	
Line Message No. Message Text Unit UART Line ad UART Line	Je drois	e warninga e rinna			UADT1 Data Deady	
356: 108 Insert Modified C:¥Users¥ken¥Documents¥monka_it¥serial_montor_1st.mbas	Line	Message No.	Message Text	Unit	VART1_Read	
356: 108 Insert Modified C:¥Users¥ken¥Documents¥monka_it¥serial_moutor_1st.mbas					UART1_TX_Idle	
356: 108 Insert Modified C:¥Users¥ken¥Documents¥monka_It¥serial_monitor_1st.mbas					UART1_Write	
356: 108 Insert Modified C:¥Users¥ken¥Documents¥monka_It¥serial_moutor_1st.mbas					UARTI_Write_Text	
Juise mountes constant and the second	256: 109 T	ncort Modified	C:XI IsorrXkonXDocumontsXmonka, it	Koorial monitor let mbac		
Library O Help	550.100 1	moure mounted	C.+Oseis+Keii+Documents+monka_ic			
library (7) Help						
			Libraryの	dight		

一番下の UART をクリックすると、さらにその関数が表示されます。
 一番上の UART1\_Init 関数をダブルクリックすると、ライブラリ使用の
 Help が表示されます。

mikroBasic PRO for PIC Help	roE Support mikroE Foru	JM		
3次(2) キーワード(2) 検索(2) (▲) ーワードを入力して(ださい(2))		I <b>T Library</b> Basic PRO for PIC Libraries > Hardware Libraries >	end relect end sole sub procedure interrept forms = forced array (portd_index) ports = shafter tingtay	' End of obscorits
	UART1_Init			
DEFINE	Prototype	sub procedure UART1_Init(const baud_rate as longint)		
SE DIF	Returns	Nothing.		
IDEFINE	Description	Configures and initializes the UART module. The internal UART module module is set to: • receiver enabled		
perator		<ul> <li>transmitter enabled</li> <li>frame size 8 bits</li> <li>1 STOP bit</li> <li>parity mode disabled</li> <li>asynchronous operation</li> </ul>		
sment display decoder		Parameters : • baud_rate: requested baud rate Refer to the device data sheet for baud rates allowed for specific	Fosc.	
ssing individual bits e comments Library Get Sample Jinit Read	Requires	You'll need PIC MCU with hardware UART. UARTI_Init needs to be called before using other functions from U/	ART Library.	
les to project a men library so operator ood breakpoints ced edit toolbar metic operators		III Note : Calculation of the UART baud rate value is carried ou large code if performed on the libary level. Therefore, complet needs to know the value of the parameter in to be a constant, and not a variable.	ut by the compiler, as it would produ the compile time. That is why this p	uce a relatively parameter needs
I chart statement mbly view ment statements	Example	This will initialize hardware UART1 module and establish the commu- "This vill initialize hardware UART1 module and establish the UART1_Init(2400)	unication at 2400 bps: communication at 2400 bps	
ic complete o correct o hide	UART1_Data_	Ready		
ic standard issues 2Dec 2Dec16	Prototype	<pre>sub function UART1_Data_Ready() as byte</pre>		
表示( <u>D</u> )	Returns			

Library の内容例

表示された Help を下にスクロールすると、配線例なども示されていますので、大変参考になります。



Library に含まれる配線例

情報系 IT 専門学校の学生は、ハードウエアに関する実践知識が不足 がちですが、このような Help からも、使用している IC やコネクタのピ ン配置・接続などが分かるので、とても有意義です。

プログラムを開発するに当たり、この環境でプロジェクトを作成しま す。以下に手順を示します。

□メニューから Project → New Project と選択します。

『New Project Wizard』 ウインドウが表示されますので、その指示にしたがって Project を作ります。

l n	nikroBasic Pro	for F	IC v.4.60.0.0	and a state of the		×
<u>F</u> ile	<u>E</u> dit <u>V</u> iew	Pro	ject <u>B</u> uild <u>R</u> un <u>T</u> ools <u>H</u> elp			
1 🖻	1 📴 📲 😤	6	New Project Shift+Ctrl+N	😓 i % 🎄 🏡   💩 i 🖳 💹 🕂 🧎 🎊 🔐 i 1024×768 🔹 🔹	- 🖬 🖓 📾 ji 🖉 🚛 ji 🚱 🥔 🍅 j	
3		8	Open Project Shift+Ctrl+O			
Proj		8	Open Project Group			Libra
ect Se			Recent Projects			M Aue
tting		F	Save Project			anage
-		P.	Save Project As			-
0			Close Project			2
ode E			Close Project Group			outin
plore			Add File To Project			e List
¥.			Remove File From Project			=
			Kentove File From Project			P
			Edit Search Paths			oject r
		×.	Edit Project Shift+Ctrl+E			Manaj
			Clean Project Folder			)er
		۵	Import Project Ctrl+I			
		8	Export Project Ctrl+Alt+E			
		Øpen Examples Eolder				
		_				
[	🔠 Messages 👔	i Q	uick Converter			
	Errors		V Warnings V Hints			
[	Line	Mes	sage No.	Message Text	Unit	
						-
					10:56	P
					2013/02/	2

Project 作成開始



Project 作成 ウイザード

□ Wizard が開いたら、Next を選択します。

New Project Wizard		x
	Step 1/6	
Select the device	you want to use.	
Device Name:	P16F648A	
	P16F648A	
	P16F887	
	 P12C671	
	P12C672	
	P12CE673	
	P12CE674	
	P12F1822 •	
l		
	◆ <u>B</u> ack <u>N</u> ext ◆ <u>C</u> ance	-

デバイス選択 □ Device を使用する PIC の型番を選択して Next をクリックします。

New Project Wizard	x
Step 2/6	
Setup the clock, for example 11.0592 MHz.	
Device Clock: 4.000000 MHz	

基本クロック指定

□ Device Clock の指定画面になります。マイコンは PIC に限らず基本 クロックが重要で、この周波数に応じてプログラムの処理速度やタイマ ー基本周波数、シリアル通信の基本スピードなどが変わります。ここで は内部クロック 4MHz を指定します。デフォルトは 8MHz です。

□次に、プロジェクトの名称と保存用 Path を指定します。

※Pathに日本語が含まれないようにしたほうが良いでしょう。

×
6
cel

# Project の path 指定

ロプロジェクトに追加するファイルを指定できますが、ここでは、そのまま Next をクリックして先に進みます。

New Project Wizard	×
Step 4/6	
Add project files if they are available at this point. You can always add project files later using the Project Manager in IDE.	
Add File To Project:	
	Add 😂
File Name	
	<u>R</u> emove
	Re <u>m</u> ove All
	Canad
	<u>C</u> ancel

Projectc にファイル追加 □使用するライブラリの指定です Default で Next をクリックします。

New Project Wizard			×
	Step 5/6		
Select initial Library Manag	jer state.		
Selecting all libraries is recomme Selecting libraries manually usir (recommended for advanced us	Include Libraries Include All (Default) Include None (Advanced) Include None (Advanced) ended for beginners. Ing Library Manager sers) results in faster compilation.	Library Manager Help	
	◆ <u>B</u> ack Next ◆	<u>C</u> ar	ncel

初期の Library 指定

Ne	ew Project Wizard
	Step 6/6
Yo ar	ou have successfully created a new project. Click Finish to save the changes nd to close the wizard.
	<ul> <li>Open Edit Project window to set Configuration bits</li> </ul>
	◆ <u>B</u> ack <u>Finish</u> <u>C</u> ancel

Finish画面

□ Finish で Project が作られて main 関数のひな型が表示されます。

	mikroBasic Pro	for PIC v.4.60.0.0 - C:¥U	sers¥ken¥Docume	ents¥monka_it¥si	implemonitor1.mbr.pr	And a state of the	341-0					x
Eile	e <u>E</u> dit <u>V</u> iew	Project Build Run To	ools <u>H</u> elp									
18	5 🗞 - 🔁 诸	12 😅 🕹 🗳 🗄 🕻	& 🖪 🔚 🗧 ک	s i 🌯 👶 🏡	🛯 🍓 💹 🗛 🦹 .	💐 🗃 🕴 1024x768	- 🖬 🐼 📾	i a 1 📖 i	🕐 🤌 😥			
7	implemonita	r 1.mbas									주 🖾	
P	1 pr	ogram simplemonitor	r1								*	E
ject Settings 1911 Code Explorer	<ul> <li>ma</li> <li>er</li> <li>er</li> <li>ma</li> <li>er</li> </ul>	Declarations sections Main program d.	Hints									rary Manager 📗 Routine List 🛄 Project Manager [1/1] - simplemonitor 1.mbppl
	Line	Message No.		Message Text				Unit				
1: :	1 Inse	rt			C:¥Users¥ken¥Do	ocuments¥monka_it¥s	mplemonitor1.mbas	_			11,21	
K	9 e		🧶 📕						A般警拳會 🕐 🛤 👻	- 📕 🌆 📴 ant 🖲	2013/02/02	2

作成されたmain処理

D r	nikroBa	asic Pro for PIC v.4,60.0.0 - C:¥Users¥ken¥Documents¥monka_It¥serial_monitor_1st.mbppi									
Eile	<u>E</u> dit	View	Pro	oject <u>B</u> uild <u>R</u> un	<u>T</u> ools <u>H</u> elp						
1	• 🔂 •	19 😢	8	New Project	Shift+Ctrl+N	🖹 😓 i 🗞 🚴 🎭 i 🖳 🔣 🖉 i 🗒 🚺 👖 i 1024x768 💿 👻 🖬 i 🖉 🖉 i 1024x768					
3	📄 seria	l_moniti	8	Open Project	Shift+Ctrl+O			주 🔀			
Proj	-			Open Project Gr	oup			*	Libra		
ot Se		₽ma		Recent Projects	,				M Yra		
ttings			æ	Save Project					anage		
Te	350	,  ľ	1	Save Project As							
0			23	<u>C</u> lose Project					2		
ode Ex				Close Project Gr	roup				outine		
plore			a a	Add File To Proj	ect				List		
_		١,	đ	Remove File Fro	om Proiect				=		
			-	Edit Search Dath	, ne				Proje		
			24	Edit Designt	Chifty Chilly F				oct M.		
	360		100	Edi <u>i</u> Project	Shirt+Ctri+E				anage		
			1	Clean Project Fo	older				a [17]		
			1	Import Project.	Ctrl+I			=	ŝ		
			8	Export Project	Ctrl+Alt+E	' USART BAUD RATE Setting. 9600 baud			rial_		
	-		1	Open Examples	Eolder	' USART Rx Status and Control			nonit		
			OPT	ION REG.7 = 0		' USART IX Status and Control ' Pull up Port B for SW			or_1s		
	•		,	-				•	timbp		
	💷 Mes	sages	🖬 Q	uick Converter					Ξ.		
	Err	ors		Warnings	Hints						
	Line		Mes	ssage No.		Message Text Unit					
286	: 58	Inse	rt	-		C:¥Users¥ken¥Documents¥monka_It¥serial_monitor_1st.mbas		11.34			
	9)	e				◎ A般警 Ø 1000 2 × P 1	🗴 📴 lati 🌵	2013/02/03	2		

# Configuration選択

 $\Box \lor = \exists - \mathcal{O} \text{ Project} \rightarrow \text{Edit Project} \ \ \forall \text{ PIC} \ \forall \dashv \exists \lor \mathcal{O} \text{ Configuration}$ 

の設定を行います。下記を参考に指定してください。

※Oscillator の項目で内部発振を指定します。この設定が誤っていて動作しないことがよくあります。

Oscillator INTOSC: I/O on RA6/OSC2/CLKOUT, I/O on RA7/OSC1/CLKIN	MCU and Oscillator	
Watchdog Timer	MCU Name P16F648A 🗸	
Off 🔹		
Power Up Timer	Oscillator Frequency [MHz] 4.000000	
Disabled 🗸		
Master Clear Enable		
Disabled		
Brown Out Detect		
Disabled 🔹	Conferencia Desisters	
Low Voltage Program		
Disabled 🔹	CONFIG :\$2007 : 0x2118	
Data EE Read Protect		Load Scheme
Disabled 🔹		Eave Echamo
Code Protect		Save Scheme
Off 🔹		
		<u>D</u> efault
		<u>0</u> K
		Cancel
	General Output Settings	_

Configuration画面

◆プログラム作成

□ UART によるシリアル通信は、

9600bps、8bit data、1stop bit、non parity とします。

□送信電文は、1文字として、次のような電文とします。

"S":ステータスリクエスト(計測データ要求)

"M":モード変更要求

※ドライブモード(D) 放電テスト

PWM モード (P) PWM 駆動

"D": PWM 駆動時の DUTY 比変更要求

#### □LCD 表示

簡易モニターは、8文字2行のLCDを持っています。効率の良い表示をおこなうために、送信時には、上段先頭に 『@』を付けた送信項目名を表示し、下段にその内容を表示することとします。これに対して受信時は、上段先頭に 『\*』を付けた項目名として、受信したことを示します。

このユニットのプログラムは、main 関数の先頭部分で、PIC の I/O ポート・UART シリアルポートの設定をおこない、Loop に入ります。 Loop では、2 つあるタクト SW の処理と Main ボードから受信した電文 の処理をおこないます。EEPROM に書き込んである、バッテリ電圧の上 限・下限電圧の閾値のうち、下限値を 1 度だけ Main ボードに送信する 処理を加えています。

割り込みは UART の受信割り込みを使用しています。通信の電文は半角 ASCII キャラクターとし、1 文字受信するたびに割り込みが発生しま すので、受信バッファから取り出して、電文バッファに書きくわえるプ

49

ログラムを sub procedure interrupt として記述します。

この環境では、PICの割り込みはすべてこの名前の関数で処理します。 ソースプログラムは、割り込み処理部分を含めて全体で 450 行程度で す。たいへん効率良く開発ができたと思います。

例として、SW2 クリック時のモードセット電文送信時は、次のように 記述します。送信時なので、1 行目先頭に@マークがを表示しています。
Lcd\_Out(1,1, "@MODE ") 'LCD 1 行目 1 カラムから表示(常に 8 文字)
Lcd\_Out(2,1, " SET") 'LCD 2 行目 1 カラムから表示(常に 8 文字)
UART1\_Write\_Text("M") 'モード変更要求電文の送信

受信時の電文は、LCD 表示文字列そのものを、先頭に"U"または"L" をつけて Main ボードが送信します。先頭1文字で上段か下段かの判断 をして受信した電文をそのまま表示します。

プログラムは次のような構成です。

※プログラム全体は、Appendix に掲示します。

1). dim で始まる定義ブロック

PICの I/O ポートの用途ごとの割当て設定を使用するライブラリ用 に設定します。また、ポート名称を何レジスタの何 bit 目、という表現 では、分かりにくいので、用途に応じた名称に定義します。

また、システムで使用するワークエリア用メモリの設定・定数の定義 などもこのブロックで行います。

2). 関数定義ブロック

sub Procedure XXXX()~end sub で囲まれたブロックで、各関数(サ

ブルーチン)を記述した部分です。

sub Procedure interrupt XXXX()~end sub で囲まれたブロックは、 割り込み処理のプログラムです。各 PIC で使用できる割り込みは、すべ てこの記述をした関数内で処理することとなるので、この関数の冒頭は、 どの割り込みなのかを判断することです。簡易モニターでは、UART シ リアル通信の受信割り込みだけ使用しているので、該当する割り込みで あることを示す bit がセットされているかを調べています。

3). main()~end.ブロック

主たる処理のプログラムを記述した部分です。ここまでに宣言、定 義した変数・定数や、記述した関数を使いながら、PIC がリセットされ た直後からの処理を記述しますので、この関数の冒頭には、変数の初期 設定や、I/O ポートの方向設定などを記述します。

#### ♦ Build

コンパイル等一連の実行可能ファイルを生成する手続きを Build と呼んでいます。mikroBasic のウインドウにある該当の ボタンをクリックすれば、コンパイル〜リンク〜hex ファイル 生成までをおこなってくれます。



ウインドウの上部にある、歯車がかみ合っているボ タンをクリックすれば、Build がおこなわれます。

# 上部の Build ボタン

b mikroBasic Pro	o for PIC v.4.60.0.0 - C:¥Users+ken¥Docur	nents¥monka_it¥serial_monitor_1st.mbppi	4.3857.55		
<u>File Edit View</u>	Project Build Run Tools Help				
8 <b>13 1</b> 18	S 68 =⇒   67 67   11 08 • 11 (19 12	2 3 5 8 2 8 5 8 5 8 10 A 1 8 10 11 11 11 11 11 11 11 11 11 11 11 11	-68 🔻 🖬 🔝 💷		
Project Settin	or_ist.mbas ain: Main program	Build Bu Id current project (CTRL+F9)			Elbrary Mana
·R 350	mode = "N"				ĝe
18	rxdata = "				
	rxbuf = "				20
ê ·	rxreg = " "				8
	rxindex = 0				ne
ore -					1
P	' Clear Port A and B				19-1
•	porta=0				
•	portb=0				
•					H M
360 /	PortDirections				anag
	CMCON = %00000111				ger
	TRISA = 400111100				_ X
	PCSTA = \$10010000	I USARI BADD RAIS Secting, 9600 Data			a la
	TXSTA = \$00100110	I USART Ry Status and Control			E S
	OPTION REG 7 = 0	/ Bull up Bort B for SW			
					, 🛄 🚆
			,		
Messages	Quick Converter				
Errors	V warnings V Hints				
Line	Message No.	Message Text		Unit	
70: 0 Inse	ert	C·¥I Isers¥ken¥Documents¥monk	a it¥serial monitor 1st mba	ac	
			actionactionactinue	👝 🗼 A ATL 🔊 🛷 🔿 🤇 🗛 🖉	9:38
<b>1</b>				🔟 A 投 🐸 🧼 🕐 кана 🗸 🔺 🧏	2013/02/03

## ProjectのBuild

□ウインドウ下部に Build の進捗が示され、エラーがあれば赤く表示さ れます。エラー該当部分は、ファイル名と行番号が示されていますので、 そこをダブルクリックすれば、ソースプログラムの該当部分にカーソル が移動します。

エラーがなくコンパイル・リンク・hex ファイル生成まで終了すると ウインドウ下部の最後の行に『Finished successfully』と表示されます。

Message	s 🔜 Quick Converter			
Errors	✓ Warnings ✓ Hints			
ne	Message No.	Message Text	Unit	
3	1016	Warning: Source size (1) does not match destination size (100)	serial_monitor_1st.mbas	
19	1021	Hint: interrupt handler (interrupt at 0x0004)	serial_monitor_1st.mbas	
73	1006	Warning: Generated baud rate is 9615 bps (error =	serial_monitor_1st.mbas	
5	1010	Hint: Unit "serial_monitor_1st.mbas" has been recompiled	serial_monitor_1st.mbas	
	134	Compiled Successfully	C:¥Users¥ken¥Documents¥monka_it¥serial_monitor_1st	
	139	All files Compiled in 62 ms		
	1144	Used RAM (bytes): 90 (38%) Free RAM (bytes): 150 (62%)	Used RAM (bytes): 90 (38%) Free RAM (bytes): 150 (6	( ) ( ) ( ) ( ) ( ) ( ) ( ) ( ) ( ) ( )
	1144	Used ROM (program words): 2307 (56%) Free ROM (program words): 1789 (44%)	Used ROM (program words): 2307 (56%) Free ROM (pr	
	145	Project Linked Successfully	serial_monitor_1st.mbppi	
	140	Linked in 94 ms		
	141	Project 'serial monitor, 1st mbooi' complet d: 203 ms		
	103	Finished successfully: 03 2 2013, 09:43:11	serial monitor 1st.mbppi	

# Build結果表示

◆書き込み

生成された hex ファイルを PIC に書き込みます。この作業には、『PIC プログラマー』と呼ばれるツールを使います。(写真)



この基板を PC にシリアルケ ーブル (RS232C) 経由で接 続して、PIC に hex ファイル を書き込みます。書き込む際 は、専用の書き込みソフトが ありますので、これを使いま す。

プログラマ

基板上のソケットに PIC を差し込んで、書き込みソフトのプログラム ボタンを操作して書き込みを行います。ソケットは 2 つありますが、書 き込みソフトでどちらを使用するか指示が出ますので、それに従い PIC を差し込みます。

PIC가 D가 카지V4 V6.72.17 C:¥Users¥ken¥Documents¥monka_it	×
ファイル(E) オプション( <u>O</u> ) アセンブラ・コンパイラ( <u>C</u> ) 【ReLoad】( <u>L</u> ) 【生成 ReLoad】( <u>A</u> )	【生成 ReLoad 書込】( <u>S</u> ) へルプ( <u>H</u> )
10F/12F50X         1           Hax B-Y         Hax (B-Y)           デンドイス道訳:         12C/F, 16F   18C215   PIC18   12F683           デンドイス道訳:         ワント           チェックサム: F000         ・           コードメジリサイズ* = 4096 word         Dev:	プログラム(P)         拡張機能           リード(R)         ペリファイ(y)           ブランクチェック(B)         CFG5           CFG5         CFG8         CFG7           CFG1         CFG2         CFG3         CFG4
000000: 3FFF 3FFF 3FFF 3FFF 3FFF 3FFF 3F	1         0x3FFF           FOSC         RC_CLKOUT           WDTE         Enable           PWRTE         Disable           MCLRP in         BODEN           BODEN         Enable           LVP         RB4:PGM           CPD         Not_Protect           CP         Not_Protect
0000:FF FF	picpgm6axe: Ver 6.72.17 device6.ini: Ver 6.72.17 COM7 ライター未接続 ?



書き込み操作画面

ソケット使 用指示

このプログラマ以外にも書き込みの方法がありますが、それは Main ボードの項で解説します。

## 4.2 Main ボード

◆開発環境

Main ボードは、PICの開発元『Microchip』社が公開している 無償の開発環境を使用し開発することにしました。

簡易モニターも mikroBasic を使わず、こちらで説明する環境 を使用して開発することも可能です。IT 専門学校の教材としての 選択肢として、この研究では意識して 2 つの環境を使っています が、いずれも無償で使えます。 簡易モニターが Basic 言語を用い たのに対して、Main ボードは、組込マイコンソフトではごく普 通に使われる C 言語で開発します。コンパイラも PIC の種類に応 じて複数公開されていますので、使用する PIC に対応したものを ダウンロードして使います。

以下、環境作成の手順を示します。

【注意事項】この研究に使用した環境は、現行のものより一世代前の ものです。現在は更新されて、旧版は入手できなくなっていますので、 ここでは最新版の IDE について、入手とインストールの解説をしていま す。 最新の環境は統合開発環境として『MPLAB X IDE』という名称で、 Microchip 社の HP から入手します。

□ HP の TOP メニューからダウンロードに移動します。

	The second secon	p.jp/ 前へ 次へ	• ۹   • ۲۶۹۷ •	- 🖒 🗙 🙋 multibrai 🔯 ト	ランジ 🩋 multib	rai <mark> </mark>	
_		IIP 品情報 ニュー	マイク ス・会社案内 ダウンロ・	ロチップ・テクノロジー・ジャパン&	朱式会社 トレーニング	ご購入 動画	٩
	Digitally-Enha Power Analog	anced		"High-efficiency, and highly-flexible power conversion solutions"	Digitally Control	-Enhanced Power Analc	g
	Control	MICADCHIP			PIC32 G	UI with MULTI-TOUCH	
	"Flexibility of a Digital Interface, Power with the	MCP87A	МСР19111		Serial El Reliable	EPROM - World's Most Memory	
	and Performance of an Analog-Based Controller	A CONTRACTOR	MICHOCHI MICHOCHI	ligh-Speed Power MOSFETs			
	お知らせ		>		62 w 7		
	2013/1/29 702399-2	マイクロチップ、柔い る世界初のマイコン 一ラを発表(PDF)	欧かつ高効率な電力変換を実現 内蔵アナログベース電源コン	セミアー・ワー 現す 最新のセミナー情報は とこちらのページをご覧 直接各お同い合わせ先	シンヨッノ いただき、 まで	クリンロート 読者が内容をお読みに なる時の参考のために 日本語に翻訳されています	
	2012/12/26 あ知らせ	<u>アールエスコンポー:</u> <u>にてchipKIT™UNO3</u> 催中(外部リンク)	ネンツ株式会社のDESIGNSP/ 12を使った 電子工作コンテス	ARK 上開 詳細はこちら>	-	詳細はこちら>	7
	2012/12/21	Microchip/SMSC季刊 2012年12月号	リニュースレターConnection:	<u>s</u>			
	2012/12/3 お知らせ	本社移転のご案内					-
		🕒 🎸	×			🧭 🕐 🕬 🛱 🔺 📕 😽 🔐 all	12:01 2013/01/30

□ダウンロードメニューの『ツール/ユーザガイド』を選択します。



Microchip 社の HP

# □表示される画面を下にスクロールします。

MCシフレット       X       X       DLB数       Vリース目         第       MPLAB@XIDEユーザガイド       52027A_JP       1531       2012/07/19         PICkut*3 プログラマ/デバッガ ユーザガイド       51795B_JP       808       2012/03/02         MPLAB REAL ICE** インサーキット TSュレータの使い方       51997A_JP       158       2011/12/26         MPLAB ICD 3インサーキット デバッガの使い方       52010A_JP       359       2011/12/26         PICkut*3 インサーキット デバッガの使い方       52010A_JP       477       2011/12/26         MPLAB ICD 3インサーキット デバッガの使い方       52010A_JP       172       2011/02/19         PICkut*3 インサーキット デバッガの使い方       52010A_JP       172       2011/02/26         MPLAB ICD 3 インサーキットデバッガコーザガイド       52010A_JP       172       2011/02/19         MPLAB ICD 3 インサーキットデバッガコーザガイド       52010A_JP       172       2011/02/19         MPLAB ICD 3 インサーキットデバッガコーザガイド       51766A_JP       172       2010/02/19         MPLAB ICD 3 インサーキットデバッガコーザガイド       51766A_JP       187       2009/05/08         PIC22 入門コーザガイド       51280_JP       613       2010/02/19         MPLAB ICD 3 インサーキットコーザガイド       51280_JP       615       2008/07/29         MPLAB ICD 3 インサーキットブバッガード       51280_JP       615       2008/07/29         MPLAB ICD 3 インサーキットブバッガイド       51288_JP<	ノール/ユーザガイド	19件中1~15件目の条件に合致したダウンロードファイルを表示しています。			
前計算         ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・	製品パンフレット				
MPLAB@         MPLAB@         X IDEユーザガイド         52027A_JP         1531         2012/07/19           PICktt*3 プログラマ/デバッガユーザガイド         51795B_JP         808         2012/03/02           MPLAB REAL ICE** インサーキット TSュレータの使い方         51997A_JP         158         2011/12/26           MPLAB ICD 3インサーキット TSュレータの使い方         5201A_JP         359         2011/12/26           PICkt**3インサーキット デパッガの使い方         5201A_JP         477         2011/12/26           PICkt**3インサーキット デパッガの使い方         5201A_JP         112         2011/02/23           dsPIC@ DSC Speex 首声 エンコーディング ティーディング ライブラリのコーザーガイ         70328A_JP         63         2010/02/19           K         MPLAB@ ICD3 インサーキットデバッガ コーザガイド         51766A_JP         114         2009/05/08           PIC322 入門コーザガイド         51766A_JP         114         2009/05/08         2012/29           MPLAB@ ICD3 インサーキットデバッガ コーザガイド         51288_JP         613         2010/02/19           F         MPLAB@ ICD3 インサーキットデバッガ コーザガイド         51766A_JP         114         2009/05/08           PIC321X スタータ キット コーザガイド         51288_JP         615         2008/07/29           MPLAB@ CG3 C コンパイラ コーザガイド         51288_JP         615         2008/07/29           MPLAB@ CG3 C コンパイラ コーザガイド         51288_JP         615         2009/07	技術記事	名称	文書ID	DL回数	▼ リリース日
Pickat <sup>m</sup> 3 プログラマ/ デバッガユーザガイド         517958_JP         808         2012/03/02           MPLAB REAL ICE <sup>m</sup> インサーキット エミュレータの使い方         51997A_JP         158         2011/12/26           MPLAB REAL ICE <sup>m</sup> インサーキット エミュレータの使い方         52011A_JP         359         2011/12/26           MPLAB ICD 3インサーキット デバッガの使い方         52010A_JP         477         2011/12/26           PICkit <sup>m</sup> 3 インサーキット デバッガの使い方         52010A_JP         172         2011/03/23           dsPICa DSC Speex 音声 エンコーディング ライブラリのユーザーガイ         51347D_JP         172         2011/03/23           dsPICa DSC Speex 音声 エンコーディング ライブラリクユーザーガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         611468_JP         232         2009/04/16           PIC32 レクタ キット ユーザーガイド         611468_JP         114         2009/05/08           PIC32 レパコーザガイド         512881_JP         615         2008/07/29           MPLAB © C18 C コンパイラ ユーザガイド         512881_JP         615         2008/07/29           MPLAB © C18 C コンパイラ ユーザガイド         51381_JP         615         2008/07/29           MPLAB © C18 C コンパイラ ユーザガイド         51535C_JP         272         2007/08/15           Dickit <sup>m</sup> 2 マイクロコントローラ ブログラマムーザガイド         512845_JP         303         2007/05/29           MPLAB © C10 コンパイラ ユーザガイド <t< td=""><td>力画</td><td>MPLAB® X IDEユーザガイド</td><td>52027A_JP</td><td>1531</td><td>2012/07/19</td></t<>	力画	MPLAB® X IDEユーザガイド	52027A_JP	1531	2012/07/19
MPLAB REAL ICE <sup>III</sup> インサーキット Tisle レータの使い方         S1997A_JP         158         2011/12/26           MPLAB REAL ICE <sup>III</sup> インサーキット Tisle レータの使い方         52011A_JP         359         2011/12/26           MPLAB ICD 3インサーキット FJ(ッガの使い方         52010A_JP         477         2011/12/26           PICkit <sup>III</sup> 3インサーキット FJ(ッガの使い方         52010A_JP         172         2011/03/23           dsPICa DSC Speex 音声 エンコーディング FJ コーザオノド         51417D_JP         172         2010/02/19           K         MPLAB@ ICD3 インサーキットデバッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         51146B_JP         232         2009/04/16           PIC32 入門ユーザガイド         61146B_JP         214         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51843_JP         615         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51533C_JP         272         2007/08/12           Inckit <sup>IIII</sup> 2 マイクロコントローラ ブログラマ ユーザガイド         51284J_JP         615         2008/07/29           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29           Inckit <sup>IIII</sup> 2 マイクロコントローラ ブログラマ ユーザガイド         51284E_JP         303         2007/05/29	ድመለዘነ	<u>PICkit™ 3 プログラマ/ デバッガ ユーザガイド</u>	51795B_JP	808	2012/03/02
MPLAB ICD 3インサーキットデパッガの使い方         52011A_JP         359         2011/12/26           PICkit <sup>™</sup> 3インサーキットデパッガの使い方         52010A_JP         477         2011/12/26           不適正ないS8 デパイスドライパのアンインストール         51417D_JP         172         2011/03/23           dsPIC6 DSC Speex 音声 エンコーデイング ライブラリのユーザーガイ         70328A_JP         63         2010/02/19           ビビ 32 入ガーキットデパッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         61146B_JP         232         2009/04/16           PIC32 ハズタータ キット ユーザーガイド         61146B_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51288J_JP         615         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         5153 C_JP         272         2007/08/12           Dickit <sup>™</sup> 2 マイクロコントローラ ブログラマ ユーザガイド         5153 C_JP         272         2007/08/12           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29	-0718	MPLAB REAL ICE™ インサーキット エミュレータの使い方	51997A_JP	158	2011/12/26
PICkkt <sup>m</sup> 3インサーキット デバッガの使い方         52010A_JP         477         2011/12/6           不適正ないS8 デバイスドライバのアンインストール         514170_JP         172         2011/03/23           dsPIC@ DSC Speex 音声 エンコーデイング / デコーディング ライブラリのユーザーガイ         70328A_JP         63         2010/02/19           MPLAB@ ICD3 インサーキットデバッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         61146B_JP         232         2009/04/16           PIC32 ハスタータ キット ユーザーガイド         61144B_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51288J_JP         615         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ ブログラマ ユーザガイド         5153 C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		MPLAB ICD 3インサーキット デバッガの使い方	52011A_JP	359	2011/12/26
不適正ないSB デパイスドライパのアンインストール         514170_JP         172         2011/03/23           dsPIC® DSC Speex 音声 エンコーデイング / デコーディング ライブラリのユーザーガイ         70328A_JP         63         2010/02/19           MPLAB® ICD3 インサーキットデパッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         61146B_JP         232         2009/04/16           PIC32 入門ユーザガイド         61144B_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB® C18 C コンパイラ ユーザガイド         51288J_JP         615         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ ブログラマ ユーザガイド         5153 C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		PICkit™ 3インサーキット デバッガの使い方	52010A_JP	477	2011/12/26
dsPIC@ DSC Speex 音声エンコーディング / デコーディング ライブラリのユーザーガイ         70328A_JP         63         2010/02/19           MPLAB@ ICD3 インサーキットデバッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         61146B_JP         232         2009/04/16           PIC32 入門ユーザガイド         61144B_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51288J_JP         615         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ ブログラマ ユーザガイド         5153 C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		不適正なUSB デバイスドライバのアンインストール	51417D_JP	172	2011/03/23
MPLAB@ ICD3 インサーキットデバッガ ユーザガイド         51766A_JP         187         2009/05/08           PIC32 入門ユーザガイド         61146B_JP         232         2009/04/16           PIC32 入門ユーザガイド         61144B_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラ ユーザガイド         51288J_JP         615         2008/07/29           ImTouch(TM)ユーザガイド         51288J_JP         615         2008/07/29           ImLAB@ C18 C コンパイラ ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		<u>dsPIC® DSC Speex 音声 エンコーディング / デコーディング ライブラリのユーザーガイ ド</u>	70328A_JP	63	2010/02/19
PIC32 入門ユーザガイド         611468_JP         232         2009/04/16           PIC32MX スタータキットユーザーガイド         611448_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラユーザガイド         51288J_JP         615         2008/07/18           dspicDEM** SMPS Buck 開発ポード ユーザガイド         70181A_JP         47         2008/05/22           PICkit** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		MPLAB® ICD3 インサーキットデバッガ ユーザガイド	51766A_JP	187	2009/05/08
PIC32MX スタータキットユーザーガイド         611448_JP         114         2009/03/13           mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@ C18 C コンパイラユーザガイド         51288J_JP         615         2008/07/18           dsPICDEM** SMPS Buck 開発ボード ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		<u>PIC32 入門ユーザガイド</u>	61146B_JP	232	2009/04/16
mTouch(TM)ユーザガイド         41328A_JP         91         2008/07/29           MPLAB@_C18 C コンパイラユーザガイド         51288J_JP         615         2008/07/18           dsPICDEM** SMPS Buck 開発ボード ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@_C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		PIC32MX スタータ キット ユーザー ガイド	61144B_JP	114	2009/03/13
MPLAB@ C18 C コンパイラユーザガイド         51288)_P         615         2008/07/18           dsPICDEM** SMPS Buck 開発ボード ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		mTouch(TM)ユーザガイド	41328A_JP	91	2008/07/29
dsPICDEM** SMPS Buck 開発ボード ユーザガイド         70181A_JP         47         2008/05/22           PICkit*** 2 マイクロコントローラ プログラマ ユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		MPLAB® C18 C コンパイラ ユーザガイド	51288J_JP	615	2008/07/18
PICkit**2マイクロコントローラブログラマユーザガイド         51553C_JP         272         2007/08/15           MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29		dsPICDEM™ SMPS Buck 開発ボード ユーザガイド	70181A_JP	47	2008/05/22
MPLAB@ C30 C コンパイラ ユーザガイド         51284E_JP         303         2007/05/29           bin co in/download ddi         dwnload nbn/ID-a14b041720c5005844255         bin         bin </td <td></td> <td><u>PICkit™ 2 マイクロコントローラ プログラマ ユーザガイド</u></td> <td>51553C_JP</td> <td>272</td> <td>2007/08/15</td>		<u>PICkit™ 2 マイクロコントローラ プログラマ ユーザガイド</u>	51553C_JP	272	2007/08/15
this co. in/download/dl_download_obs/ID-a1db9d1730c500584d255		MPLAB® C30 C コンパイラ ユーザガイド	51284E_JP	303	2007/05/29
ship.co.jp/download/di_download.php/1D=C1db3d1/30c300304d235	rochip.co.jp/download/o	dl_download.php/ID=e1db9d1730c500584d255			
				Ø	

ここで開発ツールグループの MPLAB X IDE を選択します。

製品一覧			素 ページの先頭へ戻る
→ PIC <sup>®</sup> マイクロコントローラ	> 開発ツール	> メモリ	> 安全とセキュリティ
8ビットPIC <sup>®</sup> MCU	コンパイラ	シリアルEEPROM	ホーンドライバ
16ピットPIC® MCU & dsPIC® DSC	エミュレータ	シリアルSRAM	煙&CO検知器
32ビットPIC <sup>®</sup> MCU	MPLAB <sup>®</sup> X IDE	シリアル フラッシュ	
	プログラマ	パラレルフラッシュ	> 熱管埋
> アンプ&リニア			温度センサ
オペアンプ	, 179-717	, =-9-12-17	プラシレスDCファン コントローラ
計装アンプ	CAN	> 電源管理	温度センサ(SMSC)
コンパレータ	Ethernet		リモートダイオードファンコントロー
PGA & SGA	赤外線	パッテリ管理	∋(SMSC)
	LIN	チャージポンプ	> タッチ関連製品
› データコンバータ	シリアル ペリフェラル	CPU/システム スーパパイザ	
A/Dコンパータ	USB	LDOレギュレータ	キー&スライダ
DAC&デジタル ポテンショメータ	、レガシー8051/80C51 MCU	パワーMOSFETドライパ	タッチスクリーン コントローラ
電力計測	, 0,19 3031/30C31 MC0		> 無線
電力/電流センサ(SMSC)		スイッチング レキュレータ	
		电注快出器	赤外線
		> リアルタイム クロック	パワーアンプ
			RF

開 発 ツールグループ

□ MPLAB X FREE DOWNLOAD を選択します。



DOWNLOAD 選択

□対象 OS ごとにファイルが分かれていますので、ここでは Windows 版を選択してダウンロードします。保存フォルダを確認するウインドウ が表示されますので、適宜指定します。

前へ 次へ 📝 オプション 👻			
Features (Extended) Downloads Documentation			
Title	Date Published	Size	D/L
Windows (x86/x64)			
MPLAB <sup>®</sup> X IDE v1.60	12/20/2012	318Mb	<u>(</u> ]]
MPLAB <sup>®</sup> X IDE Release Note : / User' Guide v1.60 (supersedes info in inst	r) 12/20/2012	150KB	<u>(</u> ]
MPLAB <sup>®</sup> XC8 Compiler v1.12	12/4/2012	168Mb	<u>()</u>
MPLAB <sup>®</sup> XC16 Compiler v1.11	12/13/2012	122Mb	<u>()</u>
MPLAB® XC32 Compiler v1.11a	10/04/2012	105Mb	<u>()</u>
Linux 32-Bit and Linux 64-Bit (Requires 32-Bit Compatibility Libraries)			
MPLAB® X IDE v1.60	12/20/2012	268Mb	<u>()</u>
MPLAB® X IDE Release Notes / User' Guide v1.60 (supersedes info in inst	r) 12/20/2012	150KB	<u>(1)</u>
MPLAB <sup>®</sup> XC8 Compiler v1.12	12/4/2012	172Mb	<u>(1)</u>
MPLAB <sup>®</sup> XC16 Compiler v1.11	12/13/2012	120Mb	<u></u>
MPLAB <sup>®</sup> XC32 Compiler v1.11	10/04/2012	104Mb	<u></u>
Mac (10.X)			
MPLAB <sup>®</sup> X IDE v1.60	12/20/2012	238Mb	<u>()</u>
MPLAB® X IDE Release Notes / User' Guide v1.60 (supersedes info in inst	r) 12/20/2012	150KB	<u>()</u>
MPLAB® XC8 Compiler v1.12	12/4/2012	169Mb	<u>(1)</u>
MPLAB <sup>®</sup> XC16 Compiler v1.11	12/13/2012	121Mb	άζ)

MPLAB X IDE 選択

□指定したフォルダに、最新版の IDE ファイルが保存されていますので、 ダブルクリックして解凍します。

						×
🚱 🕞 🚽 🕨 ken wiseman 🕨	ダウンロード 🕨 MPLABX 🕨			- ++ MPLAB>	の検索	2
整理 🔹 🍋 Lhazで解凍 👻	共有 マ 新しいフォルダー				•	0
🚖 お気に入り	名前	更新日時	種類	サイズ		
🍌 ダウンロード	MPLABX-v1.60-windows-installer	2013/01/30 12:08	Archive file	318,472		
■ デスクトップ 気 最近表示した場所	このフォルダに解凍(H) 解凍フォルダを指定(S) 解凍先履歴(R)	•				
10 ライブラリ	閲覧(V)					
<ul> <li>ミドキュメント</li> <li>ミビクチャ</li> <li>ビデオ</li> </ul>	フォルダ自動生成(M) 解凍先を開く(O) フォルダ無視(N)					
🜛 ミュージック						

解凍

□解凍時にインジケータが表示されます。



□解凍されたファイルができますので、そのファイルをダブルクリック
 してインストールを始めます。

Ven wiseman	► タウンロード ► MPLABX ►			▼ 4 <sub>7</sub>
理・ 🗞 Lhazで解凍 🔹	共有▼ 新しいフォルダー			
お気に入り	名前	更新日時	種類	サイズ
ダウンロード	MPLABX-v1.60-windows-installer	2012/12/19 13:43	アプリケーション	321,112
<ul> <li>デスクトップ</li> <li>最近表示した場所</li> </ul>	PIFEADATV1.00-WINdOWS-Installer	2013/01/00 12.00	Archive nic	510,472
ライブラリ				

インストーラー開 始

以下、インストールの手順です。

□ Setup Wizard が開始されます。そのまま Next をクリックします。

👺 Setup	
	Setup - MPLAB X IDE v1.60
	Welcome to the MPLAB X IDE v1.60 Setup Wizard.
MPLAB>	
	< Back Next > Cancel

Setup Wizard 開始

□ライセンス確認画面では、『I accept・・・』を選択して Next ボタン

をクリックします。

Setup
License Agreement
Please read the following License Agreement. You must accept the terms of this agreement before continuing with the installation.
MICROCHIP IS WILLING TO LICENSE THE ACCOMPANYING SOFTWARE AND DOCUMENTATION TO YOU ONLY ON THE CONDITION THAT YOU ACCEPT ALL OF THE FOLLOWING TERMS. TO ACCEPT THE TERMS OF THIS LICENSE, CLICK "I ACCEPT" AND PROCEED WITH THE DOWNLOAD OR INSTALL. IF YOU DO NOT
I accept the agreement         I do not accept the agreement         I do not accept the agreement         BitRock Installer
< Back Next > Cancel

ライセンス確 認

ロインストールするフォルダの指定です。ここではそのまま Next ボタンをクリックします。

ſ	🚰 Setup 📃 🗖 🔲 🗙	
	Installation Directory	
	Installation Directory C:¥Program Files (x86)¥Microchip¥MPLAE	
1		
	BitRock Installer	_
	< Back Next > Cancel	

フォルダ指 定

□ 事前の指定が終わりました。Next ボタンをクリックしてインストール が始まります。

Setup	
Ready to Install	
Setup is now ready to begin installing MPLAB your computer.	3 X IDE v1.60 on
BitRock Installer	
< Back Next	> Cancel

インストールの開始 確認

□インジケータが進みます・・・

👺 Setup	
Installing	
Please wait while Setup installs MPLAB X I computer.	DE v1.60 on your
Installing	
Unpacking C:¥Program []ochip¥MPLAB	K¥gnuBins¥GnuWin32
BitRock Installer	
< Back Nex	t > Cancel

インストールインジケータ

□セキュリティ画面が表示されたら『インストールします』を選択して 先に進みます。

😵 Win	Windows セキュリティ   X									
😧 ドライバー ソフトウェアの発行元を検証できません										
	→ このドライバー ソフトウェアをインストールしない(N) お使いのデバイス用の、更新されたドライバー ソフトウェアが存在するかどうか製造元の Web サイトで確認してください。									
	◆ このドライバー ソフトウェアをインストールします(I) 製造元の Web サイトまたはディスクから取得したドライバー ソフトウェア のみインストールしてください。その他のソースから取得した署名のないソ フトウェアは、コンピューターに危害を及ぼしたり、情報を盗んだりする可 能性があります。									
<b>e</b>	細の表示(D)									

セキュリティ確 認

□さらにインジケータが進みます。

👺 Setup	
Installing	
Please wait while Setup ins computer.	talls MPLAB X IDE v1.60 on your
	Installing
Creating Desktop Sho	ortcut for MPLAB driver switcher
BitRock Installer	
< B	ack Next > Cancel

インストールインジケータ

□しばらくすると、インストール完了です。

Finishボタンをクリックして、インストール完了です。

👺 Setup	
	Completing the MPLAB X IDE v1.60 Setup Wizard
	Setup has finished installing MPLAB X IDE v1.60 on your computer.
	<ul> <li>XC compilers are not installed with the IDE.</li> <li>To download a compiler or</li> </ul>
	assembler please leave this check box checked.
MPLABX	Or uncheck, and download at a later time from
	< Back Finish Cancel

インストール完 了

次に、コンパイラを入手します。

□ Microchip 社 HP の開発ツールグループから、コンパイラを選択します。



コンパイラ選 択

□移動先画面をさらに下にスクロールします。



XC コンパイラ

□ 画 面 左 側 表 示 の XC8 ま た は XC16 を 選 択 し ま す。

P MPLAB <sup>®</sup> XC8 User Guide	All Microchip C language extensions are available for use								
- MPLAB® XC18 User Guide	Microchip is offering a s	pecial edition of our C++ o	compiler software complete	ely free for filling out a sh	nort registration.				
Downloads	Download Free	MPLAB® XC32++	<ul> <li></li></ul>						
E XC8	Which Compiler Is Right For You? Use the table to match your Microchip PIC MCU product with its corresponding compiler. Click on the license type to take you directly to the product in Microchip Direct.								
- 岱 Windows - 岱 Linux									
–nd Read Me	Microchip's PIC to C	Compiler Compatibility	Table						
₽ XC16		PRO	Standard	Free** Click Downloads on Left	C++				
-r∰ Windows	PIC 10/12/16/18 MCUs	MPLAB® XC 8	MPLAB® XC 8	MPLAB® XC 8	MPLAB® XC 8				
- BP OS X - BP Read Me		<ul> <li>Workstation License</li> <li>Network Server License</li> </ul>	<ul> <li>Workstation License</li> <li>Network Server License</li> </ul>	<ul> <li>Workstation License</li> </ul>	Not Available				
B Windows B Linux B OS X	PIC 24 MCUs dsPIC	MPLAB® XC 16	MPLAB® XC 16	MPLAB® XC 16	MPLAB® XC 16				
	DSCs	<ul> <li>Workstation License</li> <li>Network Server License</li> </ul>	<ul> <li>Workstation License</li> <li>Network Server License</li> </ul>	<ul> <li>Workstation License</li> </ul>	Not Available				
- 🗗 Read Me	PIC 32 MCUs	MPLAB® XC 32	MPLAB® XC 32	MPLAB® XC 32	MPLAB® XC 32				
		Workstation License	Workstation License	<ul> <li>Workstation</li> <li>License</li> </ul>	Workstation License-Free				
		License	License		Network Server License-     PRO				
	**NOTE - *Free version includes	a 60-day PRO evaluation that can b	e started at any time in the software.						
	Free Migration from MPLAR® C and HI-TECH Compilers to MPLAR® XC								

XC コンパイラ選択

※XC8は 8bitPIC用、XC16は 16bit用 PIC用です。簡易モニターの PICは XC8で、Mainボードの PICは XC16で開発できます。

インストールは、ダウンロードしたファイルを解凍してできるファイ ルを実行するだけです。XC8、XC16 と操作の違いはありませんので、 ここでは、XC8 を例に Setup 開始からの画面遷移を列記します。



XC コンパイラインストール進行

🐓 Setup	🕼 Setup	
Installation type	License Activation Key	
Select an installation option for MPLAB XC8 C Compiler @ Install MPLAB XC8 C Compiler on this computer You will be prompted to enter your activation key Install a network client	Please enter your license activation key for Standard or PRO compiler. (Leave blank to activate an evaluation license or to use the compiler in Free mode.)	No activation key entered
You will be asked for the host name and port to connect to a license server	Activation Key:	You are about to initial the Free compiler or the PRD compiler with an evaluation license. If this is what you want click "Ves". If you wish to stay on this page, click "No".
BitRock Installer < Back Next > Cancel	BitRock Installer < Back Next > Cancel	はい(X) いいえ(N)
Setup Instal Free of Evaluation Compiler Select an installation option for MPLAB XC8 C Compiler Run the compiler in Free mode No license or server information is required for this option Activate Evaluation Incense Use a one-time 60-day Evaluation of the PRO compiler BEROck Installer	Setup  Installation Directory  Setup  Please specify the directory where MPLAB XCB C Compiler will be installed.  Installation Directory  Program Files (186) MilcrochipVicSV  BtRock Installer  C Back Next > Cancel	Complex Setup Complex Settings Apply settings to all users of this machine Add xeader file path to MCC_INCLUDE enclosed to the PATH environment variable (used by the MPLAB CI B tools) Update MPLAB IDE to use the XCB complex for all existing CLB (mccLB) projects use XLB for the CLB unker, Loranan and Assember BROOK Initiality Editors: Initiality
My Setup	Setup	
Setup is now ready to begin installing MPLAB XCB C Compiler on your computer.	Please wait while Setup installs MPLAB XC8 C Compiler on your computer.	
BitRock Installer	BitRock Installer	・・・Finish で終了

XC コンパイラインストール進行

◆プログラム作成

PIC マイコンを使用して Bluetooth 経由で Android タブレット と通信するシステムのサンプルは、PIC メーカからもライブラリ が公開されていて、それを利用したサンプルが WEB 上に数多く 公開されています。PIC で USB に取り付けた Bluetooth を使う には、2 つの大きなプログラムが必要です。

USB 制御ソフト

②. Bluetooth 制御ソフト

いずれも、何もない状態から開発するとたいへんな工数がかか りこの研究の主題がずれてしまいますので、公開されているシス テムをうまく使い、追加作成する部分の設計をおこないます。

main()関数に次のタスクを作成します。

①. Bluetooth 受信タスク

対向 Android から Bluetooth 経由で電文を受信します。

②. 通信テストタスク

Main ボードの通信テスト SW が操作されたことを対向 Android に通知します。

③. 放電テストタスク

放電テスト SW の操作を処理するタスクです。SW が操作された ことに対する内部処理と、簡易モニター・Android への通知が処 理の内容です。

- ④. 充電テストタスク
   充電テスト SW の操作を処理するタスクです。SW が操作された
   ことに対する内部処理と、簡易モニター・Android への通知が処
   理の内容です。
- ⑤. ハートビート LED タスク

main()関数の処理の進み具合を表示する LED の点滅を制御します。

⑥.シリアル受信タスク

UART 経由の受信処理をおこないます。対向機は簡易モニターです。

⑦. main 制御タスク

バッテリの電圧を閾値と比較しながら充電・放電の制御を、おこ ないます。また、停電検出信号の監視も行います。

⑧. SPP 受信処理

このシステムは、Bluetooth 通信を『SPP プロファイル』(Serial Port Profile) という方式で使用します。この方式は、Bluetooth で接続される2つの機器の通信をシリアル通信でエミュレートし て、無線であることを意識せずに使用することができて大変便利 です。この通信の受信の部分(Android からの受信電文処理)を おこないます。

⑨. 上記プログラムを全体的に制御する main()関数

以上、9つの部分を作成します。

⑧を除く①~⑨は、main()関数と同じファイル main.c に記述しています。 ⑧は、bt spp recieve callback()関数として作成しています。

♦ Make

簡易モニターの開発環境では Build でしたが、こちらの環境で は Make となっています。開発時に使用した MPLAB IDE のウイ ンドウの上部にある Make ボタンをクリックすれば、コンパイル ~リンク~hex ファイル生成まで一連の作業を実行します。

Smart_IT - MPLAB IDE v8.85 - Smart_IT.mcw	And								
File Edit View Project Debugger Programmer Tools Configure Window Help									
D 🚅 🗟   🙏 🐂 🏨   🗟 🚧 🕮 🚚 🌹   Release 🚽 🗃 📾 🗣 🕸 🏥 📾 🖗									
Checksum: Dxa2b	f Mare								
Smart_IT.mcw	C:#Microchip#PICProjects#Smart_IT#main.c	<u> </u>							
	// タイマー4書的込み処理								
B Gormon	//> PWMC1278UGCNS								
Delay.c	<pre>voidattribute((interrupt, no_auto_psv)) _T4Interrupt(void)</pre>								
🖆 main.c	( TES1bite T4TE = 0: // 常的込みフラヴ クリア								
B- DUSB	tm_Sms++;								
usb_bt_driver.c	if(tm_5ms>=20){								
Header Fles									
bt_spp.h									
i btstack	It (mode== 'P')								
	if (tm_Bms==0) (mPORTASetBits(0x000B);)	E							
HardwareProfile.h	<pre>if(tm_Sms ==(duty/5)){     mPOBTAClearBits(0x0008);</pre>								
b usb bt driver.h	}								
usb_config.h									
Chinet Elon		-							
Files V Symbols		► a							
I Watch	L Output								
	Deale Music Council Englis Else BIOLA 2								
Add SFR ADICHS V Add Symbol _C3U_UARI V	Build Version Control Prind in Press Trock 3								
Update Address Symbol Name Value									
packet of Scope	No PICkit 3 Connected								
		-							
	m	- F							
PICkit 3 PIC24FJ64GB002	oab sab IPO dc n ov z c WR								
	🔘 A 般 🔮 🤗 🖓 📩 😕 🦹	13:40 2013/02/03							

Make ボタン



Make成功

Make によるエラー検出がなければ、右下 Output ウインドウの最後に、 『BUILD SUCCESSED』と表示されます。 ◆書き込み

出来上がった hex ファイルは、やはりプログラマという装置で PIC に書き込みます。MainCPU では、簡易モニターとは違った 装置で書き込みます。

MainCPUは、ICSP(In Circuit Serial Programming)という手法が使えます。書き込みに使用する装置は、Microchip 社の PICKIT3というもので、CPUを基板に装着したまま、hexファイルの書き込みができます。



ICSP による書き込みの様子

書き込み操作は PICKIT3 を USB 接続して、使用している IDE ウインドウで、上部にあるメニューから Programmer→Program と選択します。数秒で書き込みが完了します。

### 4.3 Android タブレット

#### ◆開発環境

Androidの開発環境も、無償で提供されているものが使えます。 次の 4 つのプログラムを順にインストールして環境を作成しま す。

①. Java 開発キット (JDK) のインストール

統合開発環境(Eclipse)のインストール

③. ADT (Android Developer Tools) のインストール

④. Android SDK のインストール

開発環境のバージョンアップが頻繁に行われているので、どんどん使いやすくなっている半面、具体的な手順をここで紹介しても、短時間の間に陳腐化してしまいます。WEBや参考書籍 (Android プログラミングバイブル:ソシム:布留川英一著) などを参考に環境を整えてください。

🥃 Java - SmartMonitor/src/com/hrdapp/android/Bluetooth10/Bluetooth10Activity.java - Eclipse								
File Edit Run Source Navigate Search Project Refactor Window Help								
📑 🕶 🖬 🐨 🔛 📾 📄 🛅 🚔 🔛 🕶 😂	J <sup>0</sup> d 🕸 • O • 🏊	• #	G • 🔌 😂 🖋 •	i 🎤 🖗	••••••••••••••••••••••••••••••••••••••	• 😓 🔶 • 😅 •	😭 🏇 Debug 🐉 Java	
😫 Package Explorer 🛛 📄 🖕 🍃 🍸 🗖 🗖	BluetoothIOActivity.java	23	) main.xml			- 0	🗐 Task List 🛛 🖓 🗖	
i ActivityEx	🖞 🕶 🖫 📽 😵 🗙 🖻 🔞 🏹							
BasicAccessoryDemo	# //import android b	<pre># //import android.bluetooth.BluetoothAdapter;[] # import android.app.Activity;[]</pre>						
CameZonActivity	<pre>mport android.app</pre>							
i EditTextEx								
i GestureEx	public class Blueto	othIOAc	tivity extends Bluetoot	thBaseActi	ivity implements log	gleButton.OnCheckedChan		
GraphicsEx	// Layout Views						① Connect Mylyn X	
HelloWorld	private TextVie	w mTitl	e;	<b>・</b> ノー	- 7 つ — ド		Connect to your task and ALM	
	private TextVie	w ciici w mSWSt	e; atus:				tools or create a local task.	
E KeyEx	private Toggle	Button m	ToggleButtor;					
SampleGideEx	private Button	mButton	1; // 七十支更示 2: // 在一下支更示	0				
SmartMonitor	private Button	mButton	2, // 空電ラストボタ 3; // 充電テストボタ	5				
Street and a stree	private Progres	sBar PB	ar; // プログレスバー				BtnClickListener ^	
gen [Generated Java Files]	private Progres	sBar PB	ar2; // プログレスバー					
Android 4.0.3	private Frogres	w ModeN	ormal: // /-マルモード	表示				
Android Dependencies	private TextVie	private TextView ModeDrive; //ドライグモード表示						
Ge and	private TextVie	private TextView ModePWM; // PW元一作表示						
Services	private TextVie	w Voita w Amper	e: // AC雷流					
AndroidManifest.xmi	•						< III >	
nroquard-project byt	Problems @ lavadoc	Declar	ation 🔲 Console 🗊 LogC	atro	Droperties	) )		
project properties		ep Decidi				·		
TringEx	Saved Filters 💠 😑 🖹	Search	for messages. Accepts Java	reaexes. F	Prefix with pid:, app:, ta	a; or text: to limit scope.	verbose 👻 🔛 📖 🔲	
SurfaceViewEx	All messages (no filters		· · · · · · · · · · · · · · · · · · ·			3		
TextViewEx		Level	Time	PID	. Tag	Text	*	
1 UsbEx		I	02-03 15:09:32.020	369	BlurServiceM	WSBase backOff fired!		
TideoViewEx		I	02-03 15:09:32.020	369	WSBase	mother told us it's okay t	to retry the waiting requests: 1-	
☐ WisemanUsbAccessorySample	D 02-03 15:09:32.020 369 WSBase processing request: clouds/						s/1/newsessioncloud +	
· · ·	< III	•			III		•	
0*	Writable		Smart Insert	265:10				
📀 🧉 📜 o 💰 🕼						📧 A 般 😂 🥔 🕐 📖 🕫	▲ 🧏 隆 🛱 📲 🕕 15:51	

開発に使用した Eclipse ウインドウを示します。

Eclipse による IDE

## ◆プログラム作成

PIC を使用する簡易モニターと Main ボードのプログラムと違い、Android 側のプログラムは、ほぼ出来上がっているものが公開されていますので、画面設計と Bluetooth 通信電文に応じた追加をおこなえば完成できます。

画面は次のような構成としました。配置した Object を示します。 ①. タイトル(固定表示)

- ②. モード表示 (NORMAL)
- ③. モード表示 (DRIVE)
- ④. モード表示 (PWM)

※該当時モード名が青色になります。

- ⑤. バッテリ電圧
- ⑥. 消費電流 (AC): インバータを介して出力している電流のみ
- ⑦. DUTY比 (PWM モード時)
- ⑧.受信メッセージ: Main ボードから受信した Bluetooth 通信
   メッセージを表示します。
- ⑨. モード変更ボタン
- ⑩. 充電テストボタン
- DUTY 変更ボタン
- <sup>12</sup>. Bluetooth 通信テスト

実際の画面は、次のとおりです。



Android タブレットの画 面 設計

プログラムは、主となる Activity クラスに各 Object の部分を追加して作成します。

主な部分は、以下の通りです。

□ BtnClickListener

⑨⑩⑪のボタンがクリックされたときに処理をおこないます。

※Main ボードへの Bluetooth 通信電文は、

⑨ MODE:モード変更=M

① CHARG: 充電テスト=C

⑨ DUTY: DUTY 変 更 = D

の、各1文字の電文とします。

□ onReviceMessage

Bluetooth 通信により受信した Main ボードからのメッセージを処
理する部分です。モード変更メッセージの場合は、②③④のモード表示 を変更します。また、DUTY変更メッセージの場合は、DUTYの%表示 とプログレスバーの割合を DUTY に合わせて変更します。

### ♦ Build

Java - SmartMonitor/src/com/hrdapp/	andro	oid/B	luetoothIO	/BluetoothIOA	tivity.java	- Eclips	CONTRACTOR OF A DESCRIPTION OF A DESCRIP	100.00	State Strength State		
File Edit Run Source Navigate Se	arch	Proj	ect Refa	ctor Window	Help	_					
📑 🛨 🖬 🐨 🔛 🔯 🦉	1.6		Open Pro	ect		₿	ଙ 🔹 🤔 😂 🛷	• 🖗 🌛 🗟	• 🔳 🔳 🕌 👻 😽	• 🖘 💠 •	🔛 🏇 Debug 🐉 Java
🔋 Package Explorer 🛛 🔲 😫	8		Close Pro	lect		X	a main.xml			- 0	Task List 🛛 🖓 🖓
Calify ActivityEx		010	Build All		Ctrl+B	ndroi	.BluetoothIO;			<u> </u>	🕺 🕈 🕶 🕅 🛱 😭 😭 🗶 🖂 🗐 🎽
asicAccessoryDemo			Build Proj	ect			01	1		E	Find O b All b Activate
ameZonActivity			Build Wor	king Set	•	ctivi	v:	4			Tind S F All F Activities
👕 EditTextEx			Clean				,,,,,				
i GestureEx		$\checkmark$	Build Auto	omatically		thIOA	tivity extends Blu	etoothBaseAct	ivity implements Top	ggleButton.OnCheckedChan	
👕 GraphicsEx		@	Generate	lavadoc							Connect Mylyn     S
👕 HelloWorld		-	Generate	3446666		mTit	e;				
👕 ImageEx			Properties	3		cTit	e;				tools or create a local task
👕 KeyEx		_		privat	e ToggleB	w mSWS	atus; ToggleButton:				
👕 SampleGideEx				privat	e Button	mButto	1; // 壬一ド溪	更ポタン			🗄 Outline 🛛 👘 🗖
🕵 SmartMonitor			-	p. i.u.t			2; // 停電元	、トポタン			🖻 📲 😿 💉 🛛 🗙 👘 🏹
src ∰				privat	e Button e Progres	eBac PR	13; // 光電テ。 tar: // プログレ	、トホタン 2 バー			G BtnClickListener ^
🈂 gen [Generated Java Files]				privat	e Progres	sBar PE	lar2; // プログレ	2/5-			onClick(View) : voic
Android 4.0.3				privat	e Progres	sBar PE	lar3; // プログレ	2/1-			onCreateOptionsMenu()
Android Dependencies				privat	e TextVie	w Model w Model	lormal; // ノーマル	モード表示			● _ onOptionsItemSelected
📴 bin				privat	e TextVie	w ModeF	NMM; // PWME-	ド表示			onActivityResult(int, int,)
🐉 res				privat	e TextVie	w Volta	ige; // /เึ่งรูปใ	旺			● ▲ onChangeConnectionSta
AndroidManifest.xml				privat	e TextVie	w Amper	re; // AC電波			· ·	<ul> <li>onCheckedChanged(Con *</li> </ul>
d lint.xml				•						•	۰ III ۲
proguard-project.txt				Problems @	Javadoc 🛛	Declar	ration 📮 Console 🗊	LogCat 🛛 🔪	Properties	h	
project.properties				and fillence							
1 StringEx			2	aved Filters		Search	for messages. Accepts	Java regexes.	Prefix with pid:, app:, t	ag: or text: to limit scope.	verbose 👻 📙 🖳 🛙
SurfaceViewEx				All messages (	no filters	Level	Time	PID	Tag	Text	
TextViewEx						-	02.02.15.00.22.0	20 260	Plunformi coli	MCPage backOff fixed	
1 UsbEx						+	02-03 15:09:32.0	20 369	McB	wabase backorr rifed:	
TideoViewEx						-	02-03 13:09:32.0	20 369	wababe	mother told us it's okay	to fetry the waiting requests: 1
WisemanUsbAccessorySample					_		02-03 15:09:32.0	20 369	wabase	processing request: cloud	s/1/hewsessioncioud
<			• L	< (		•					,
□*				Writable			Smart Insert	265 : 10			
🔗 🧭 🚞 o	<b></b>	3								🔞 A 般 🐸 🥔 💿 🕬 🖉	▲ 📙 📴 🔐 🕕 15:56
											2013/02/03

Auto Build

IDE の上部にある Project メニューから Build Automatically を選択 しておくと、ソースードを入力すると同時に Build が実行されます。 誤った記述をするとその時点で、エラー表示されるので、すぐに訂正す ることができます。

◆書き込み

Android タブレットのプログラム書き込みは、書き込み操作を おこなうというよりも、Debug 操作をおこなうと説明したほうが よく理解できます。 Android タブレットを USB ケーブルで PC に接続します。 Eclipse ウインドウで RUN メニューから Debug Configuration に進み Debug を開始すると、現在のタブレットプログラムがダウ ンロードされて、プログラムが開始されます。

	-	Joodice Navigate Search Proj	ject Nerdetoi Wi	nuow nep						
- 1	8	Run	Ctrl+F11	• • • •	# G ·	• 🙆 🖨 🛷 - 👎 .	/ 🖓 🔳	🔳 🔮 🕶 🂱 🕶 🍤	⇔ • ⇔ •	😭 🕸 Debug 📳
ackag	Ŷ	Debug	F11	IOActivity.jav	/a 🖂 🛛	main.xml			- 0	Task List 🕱
Act		Run History	+	e com.hrdapp	p.android	.BluetoothIO;			*	1 - R & X B
🖥 Bas		Run As	+	et android i	bluetooth	BluetoothAdapter:			E	Find Q > All > Activat
Car		Run Configurations		rt android.h	bluetooth	.BluetoothDevice;				
Edit		Debug History	•	android.app	p.Activit	y;				
Ges		Debug As	+	android.co	.Bundle;	enc;				
Gra		Debug Configurations		rt android.u	util.Log;					① Connect Mylyn
TIM	Θ	Toggle Breakpoint	Ctrl+Shift+B	android.vie	ew.Menu; ew.MenuIn	flater;				Connect to your task and ALI
Kev	0	Toggle Line Breakpoint		android.vie	ew.MenuIt	em;				tools or <u>create</u> a local task.
Sar	۲	Toggle Method Breakpoint	android.vie	ew.View; ew.View.O	nClickListener:				🔠 Outline 🕱	
🖇 Sm	ø	Toggle Watchpoint		android.vie	ew.Window	1				E 1ª R x o x
æ	x	Skip All Breakpoints		android.wig	dget.Comp dget.Text	oundButton; View:				import declarations
2	*	Remove All Breakpoints		android.wid	dget.Toas	t;				<ul> <li>android.app.Activity</li> </ul>
-	٦ŝ	Add Java Exception Breakpoint		android.wig	dget.Togg	leButton;				<ul> <li>android.content.Inte</li> </ul>
-	Θ	Add Class Load Breakpoint		android.wig	dget.Prog	ressBar;				<ul> <li>android.os.Bundle</li> </ul>
e.	G	All References		clarg Blue	toothTOAc	tivity extends Bluetoo	thBaraActi	wity implements Top	aleRutton OnCheckedChap	<ul> <li>android.view.Menu</li> </ul>
6	۵.	All Instances	Ctrl+Shift+N	Class blue	COUCHIORC	civity extends bidetoo	cilbaseAcci	tvity imprements rog	#	<ul> <li>android.view.Menul</li> </ul>
a		Instance Count		-					•	
B		Watch		@ Javadoc	📵 Declar	ation 🗐 Console 😰 Logo	Cat 🛛 🚺	🗉 Properties 🛷 Searc	n	
	Q,	Inspect	Ctrl+Shift+I							
Stri		Display	Ctrl+Shift+D	rs 🕈 = 🖻	Search	for messages. Accepts Java	a regexes. P	Prefix with pid:, app:, ta	ag: or text: to limit scope.	verbose 👻 🔚 🖳 🔲
Sur	40	Execute	Ctrl+U	ges (no filter:	Level	Time	PID	Tag	Text	
Tex		Force Return	AIC+SNITC+F		T	02-03 15:09:32.020	369	BlurServiceM	WSBase backOff fired!	
Ust		Step Into Selection			÷	02-03 15:09:32.020	369	WSBase	mother told us it's okay t	to retry the waiting requests:
Wie	<b>9</b>	External Tools	•		D	02-03 15:09:32.020	369	WSBase	processing request: clouds	s/1/newsessioncloud
		m	<	III	•			m		

#### 5. テスト

テストは、一般的なソフトウエアの Debug 作業になります。詳細な記述はさけますが、基本的には、3つのユニットのなかで、簡易モニターを先にやや完成させ、続けて Main ボードを進めて、Bluetooth で接続 してタブレットをテストしたあと、全体のテストを行うという手順です。



#### 5.1 簡易モニター

完成した簡易モニターは、
 まず電池を接続し、5Vの電圧が
 PIC、RS232C レベル変換 IC、
 LCD 電源ピンに正しく出てい

るかをテスタで確認します。その後、電池を一度外し LCD を接続した 後、改めて電池を接続します。その際、LED が何度か点滅した後に、LCD 簡易モニター表示例 にメッセージが表示されるかどうかを確認します。

次に、SW を操作して該当のメッセージが表示されることを確認しま

す。

簡易モニターは、後の操作で逐次画面を確認しますので、この段階で 完全に出来ていなくても、メッセージの表示と SW 操作に対する反応が できていれば良いでしょう。



写真は、SW1を操作した際の STATUS要求画面が正しく表示 されている様子です。

#### 状態要求画面

以下、簡易モニターの画面表示例を示します。



充 電 テスト

停電検出



PWM モード

DUTY 変更要求

DUTY10%



DUTY30%



DUTY40%



Main Drive

5.2

Main Charge

Monitor Drive



Mainボード

Main ボードは、 Powerボードから電 源が届いて要ること を確認したら、簡易 モニターを接続して、 電源を入れます。

通電開始すると、 LEDが何度か点滅

Main ボード を繰り返し点滅した後、簡易モニターに、Main ボードが処理開始した メッセージが表示されれば、シリアル通信正常であることが確認できま す。

開発時は、一気に全部のプログラムを作らずに通信部分だけ先に作り、 簡易モニターの画面表示を利用しながらさらに作り込んで行きました。 簡易モニターは単なるモニターだけでなく簡易デバッガーの役目も果た してくれます。各々のユニットの SW 操作では、互いに通信を行い、モ ニター画面に表示が行われるので、1つずつプログラムを追加しながら テストします。

Main ボードのシリアル通信は(簡易モニターも)、PC に接続してタ ーミナルソフトでのテストもできます。コネクタケーブルの断線等の時 には、どちらが不具合か切り分けが出来ない場合がありますので、写真 の様に、PC 接続で判断します。



PC のシリアルポ ートにテストコネクタ をつけて接続していま す。シリアル通信の為 の ソ フ ト ウ エ ア は 『TeraTerm』を使用し てテストしました。

PC との接続 テスト

Mian ボードの SW で DRIVE・CHARGE のテストが出来ます。この SWをクリックしたときの簡易モニターの画面表示と、点灯する LED が、 放電回路と充電回路の LED であることが確認出来れば、よいと思いま す。

#### 5.3 Android タブレット

タブレットは、Bluetooth での通信が主体なので、まず、Main ボード とのペアリングを行い、操作してみます。その操作状態で、簡易モニタ ーにメッセージ表示が適切に行われていれば、動作は概ね間違い有りま せん。

今回、簡易モニターの SW と同じ機能をタブレットのボタンに持たせ ましたので、各々のボタンを操作して、Main ボードの動作と簡易モニ ターのメッセージが正しいことを確認します。

システムが完成すると、Main ボードが電圧計測値を常時送信してく るので、このメッセージがタブレットに正しく表示されて、定期的に更 新されることが確認出来れば完成です。

以下、タブレットでシステムを稼働させている様子示します。

















#### 6. 稼働実験

## 6.1 充電テスト

このシステムでは、システムが稼働(通電)している時に、太陽光 パネルからの電流をバッテリに流して充電することとなっています。

Optosupply Optosu	upply So	lar	
Solar Mod	ule : OSSM-	SF0012	
Maximum Power Voltage at Pmax Current at Pmax Open-Circuit Voltage Short-Circuit Current	(Pmax): (Vmp): (Imp): (Voc): (Isc):	12 17.2 0.70 21.6 0.77	WP >A > A
Power Tolerance: +/- 3% Size: Weight: 1.5 Kg Cells: Max.System Operating Voltage: 1000 V,Star	380mm*275m 36pcs,125*17.85pc ndard Test Condition: 1	m*25mm oly-crystalline si 000W/m' , AM1.5	licon , <u>25°C</u> .

太陽光パネルの裏面

太陽光パネルの規格は、パネルの裏面シールで確認できます。

システムを組み上げて、充電テスト SW を操作したとき、パネルか らバッテリに充電電流が流れている事をテスタで確認します。



開放電圧 21.2V

電流 21.26mA

#### 6.2 放電テスト

放電 SW を操作したとき、Main ボードの LED が点灯し、インバー タ接続した AC 電源家電(テストでは、電球型蛍光ランプ)が点灯すれ ば、問題有りません。 5章で解説の通りです。

#### 6.3 PWM 制御実験

モードが NORMAL のとき、簡易モニターの SW を操作して、放電 テストモード (DRIVE モード) にします。このとき放電している事を示 す LED が点灯します。数秒後に NORMAL モードに復帰しますので、復 帰する前にもう一度 SW をクリックすると、PWM モードに入ります。



DRIME モード

PWM モード

PWM モードになったとき、DUTY が0%で、放電 LED は消灯します。 簡易モニターの DUTY 変更 SW を1度クリックすると DUTY が 10%ず つ上昇します。LED は点滅する様になります。

DUTYは10~80%の間は、10%刻みで変化します。80~95%の間は、 1%刻みで変化します。SWを操作して変化する DUTY に対応した信号 が出ていることを、出力端子にオシロスコープを接続して確認します。

LED は、95%で連続点灯しているように見えます。





DUTY 変化をオシロスコープで観測

次に、インバータに AC100V のランプを接続して点灯テストを行いま す。実際に行って見ると、準備したランプは、70%程度で点滅し始めて、 やはり 95%で連続点灯しているように見えます。

この状態で 5%の電流カットを行っていますので、その分が節電となる事が確認できます。

環境エネルギーを利用するうえでは、PWM制御が重要性を持つことを 理解出来る教材となります。

# 6.4 停電検出

システムを稼働させたうえで、乾電池電源の SW も ON して、放電テ ストを行います。その際、インバータに接続した電球が点灯することを 確認します。

次に、システムに供給している AC アダプタ電源を抜きます。電源が 電池電源に切り替わり、インバータに電源供給が始まり、電球が点灯す ることが確認出来れば、停電検出は問題無く稼働していることになりま す。 このシステムでは、太陽光パネルを発電源としていますが、考え方を 変えて、通常 AC 電源からバッテリに通電して充電をしておき、停電時 にバックアップ電源に切り替わることを利用すると、UPS などに代表さ れる無停電電源装置を作ることもできます。

## 7. 教材としての評価

システム全体は、研究を行う際に手作りしていますが、基板を量産し 半田付けを省略して、情報系専門学校の教材として利用し易く出来るで しょう。

簡易モニターと Main ボードは、PIC のプログラムを作り替えることで、PWM 制御だけでなく、MPPT 制御(Maximum Power Point Tracking) などを組み込むことができます。この様にして組み込んだ新たな制御技 術で、どれほどの効果があるかを、Bluetooth 通信の対向機である Android タブレットで、SD カードなどに保存、グラフ表示するなどの プログラムを開発が、さらに進んだ教材となり得ると考えています。

発展的には、この教材を組み込んだ、『太陽光で走り、AC 電源も供給 できる EV (電気自動車)のモデル』を開発して、まさにスマートエネ ルギーの IT 実験教材にまとめ上げたいと考えます。

この研究は、まだ教材テーマとしては模索が続くことと思われますの で、開発者としては次年度以後も継続研究できることを強く望む次第で す。

#### 8. 反省点

この研究では環境エネルギーの利用に関わる IT 技術者の学習教 材を開発して来ましたが、設計・製作・ソフト開発・実験などを通じて 得られた反省点を列記します。

◆教材の仕様について

何を教えるべきか、何が必要とされるのかが分からないままに、 始めた教材開発ですが、初年度としてはまとまりました。この研究で開 発した仕様がベストとは思いませんので、これをたたき台とした、継続 研究が必要です。

◆システムとして、停電検出や PWM 制御など、実用性のあるもの が組み込めることは分かりましたので、電子・電気系学科と情報(ソフ ト)系学科でコラボレーションした学習など、多面的な展開を模索する ことも必要と思われます。

◆製作段階での半田付けなどは必須ですが、これをこなせる情報系 学生は少ないと思いますので、キット化する際には、パターン基板の小 ロット製造を行うか、または、個別パーツをブロックのように加工して、 組合せながら学べる教材とする研究も需要があると思われます。

◆教材を使いこなせる人材育成も必要とおもいます。開発側の意図 に沿った教材の利用と工夫で授業内容が高まると思われますので、モデ ル授業を開催し、教材がどのように使えるかを研究すべきと考えます。

◆ソフト開発については、2種のマイコンに加え Android タブレットのソフト開発(Basic・C・JAVA)と、かなり工数の掛かる教材開発 となりましたが、既存のソフトウエア資産を旨く利用する事で日程内の 工程となりました。この様な研究を行う人材も貴重なので、是非継続的 な研究を行い、Android のグラフィックスに関する開発に取り組み、EV 車のメータ周りや、現在の Status 表示のモバイル対応などを行えば、 さらに魅力有る教材・教材開発研究となるでしょう。

# プログラムリスト

Appendix 1.	簡易モニター プログラムリスト
Appendix <b>2</b> .	Main ボード プログラムリスト
Appendix 3.	Android タブレットプログラムリスト

# Appendix 1. 簡易モニター プログラムリスト

program serial\_monitor\_1st

"2012.12.28 1st version

- " by wiseman Co,.Inc. kenichi.harada
- " CPU: PIC16F648A
- " LCD

" USART with RxIRQ.

" LED

 $"\,\mathrm{SW}$ 

" CopyRight kenichi-harada

'Declarations section

'Define Constants

'LED Port connection

dim

LED1 as sbit at RA6\_bit LED2 as sbit at RA7\_bit LED\_R as sbit at RA6\_bit LED\_G as sbit at RA7\_bit

dim

LED1\_Direction as sbit at TRISA6\_bit LED2\_Direction as sbit at TRISA7\_bit 'End LED Port connection

' SW Port connection dim SW1 as sbit at RB0\_bit SW2 as sbit at RB3\_bit

#### dim

SW1\_Direction as sbit at TRISB0\_bit SW2\_Direction as sbit at TRISB3\_bit 'End SW Port connection

' Lcd module connections dim LCD\_RS as sbit at RA1\_bit LCD\_EN as sbit at RA0\_bit LCD\_D7 as sbit at RB7\_bit LCD\_D6 as sbit at RB6\_bit LCD\_D5 as sbit at RB5\_bit LCD\_D4 as sbit at RB4\_bit

#### dim

LCD\_RS\_Direction as sbit at TRISA1\_bit LCD\_EN\_Direction as sbit at TRISA0\_bit LCD\_D7\_Direction as sbit at TRISB7\_bit LCD\_D6\_Direction as sbit at TRISB6\_bit LCD\_D5\_Direction as sbit at TRISB5\_bit LCD\_D4\_Direction as sbit at TRISB4\_bit ' End Lcd module connections

dim msg as char[8]

i as byte 'Loop variable cbuff as char[9] sbuff as char[9]

dim mode as byte rxbuf as char[10] rxdata as char[10] rxreq as byte rxindex as integer

dim eep\_temp as byte h\_volt as word l\_volt as word f\_volt as byte f\_volt\_l as byte

eep\_check as word

" Selfcheck LED sub Procedure LED\_Blink() LED\_R = 1 Delay\_ms(500) LED\_G = 1 Delay\_ms(500) LED\_R = 0 Delay\_ms(500) LED\_G = 0 Delay\_ms(500) end sub

```
" Blinking Green LED
sub procedure Blink_LED_G()
LED_G = 1
Delay_ms(30)
LED_G = 0
Delay_ms(30)
end sub
```

```
" Blinking Red LED
sub procedure Blink_LED_R0
 LED_R = 1
  Delay_ms(30)
  LED_R = 0
  Delay_ms(30)
end sub
" Wait for SW1 Off
sub procedure SW1_OFF0
  while SW1=0
   asm
     nop
   end asm
  wend
end sub
" Wait for SW2 Off
sub procedure SW2_OFF()
  while SW2=0
   asm
     nop
   end asm
  wend
end sub
```

"

sub Procedure SW1\_Proc0
if SW1 = 0 then
Delay\_ms(5) "Chattering protection.
if SW1 = 0 then
rxindex = 0
Blink\_LED\_G0

#### SW1\_OFF

```
select case mode

case "N"

LED_R = 1

Lcd_Out(1,1,"@STATUS ") 'Write text in first row

Lcd_Out(2,1," REQUEST") 'Write text in second row

UART1_Write_Text("S") 'Send Command for Status request

LED_R = 0

Blink_LED_G0
```

#### case "P"

```
LED_R = 1
Lcd_Out(1,1,"@DUTY ")
Lcd_Out(2,1," CHANGE")
UART1_Write_Text("D")
LED_R = 0
Blink_LED_G0
```

' Write text in first row ' Write text in second row ' Send Command for Duty change

case "D"

```
end select
else
end if
end if
end sub
```

sub procedure SW2_Proc	)
if $SW2 = 0$ then	
$Delay_ms(5)$	" Chattering protection.
if $SW2 = 0$ then	
rxindex = 0	
Blink_LED_G0	
SW2_OFF	
$LED_R = 1$	
Lcd_Out(1,1,"@MO	DE ") 'Write text in first row
Lcd_Out(2,1,"	SET") 'Write text in second row
UART1_Write_Text	t("M") 'Send Command for Status request
$LED_R = 0$	
Blink_LED_G0	
end if	

end if end sub

```
sub procedure RxData_Proc0
 if rxreq = "*" then
   rxreq=""
   select case rxdata[0]
     case "U","u"
       for i = 0 to 7
         msg[i] = rxdata[i+2]
        next i
        Lcd_Cmd(_LCD_CLEAR)
                                            'Clear display
       Lcd_Out(1,1,msg)
                                'Write text in first row
     case "L","l"
       for i = 0 to 7
          msg[i] = rxdata[i+2]
       next i
       Lcd_Out(2,1,msg)
                                'Write text in second row
       if strncmp(msg," CHARGE",8)=0 then
          mode = "C"
       end if
       if strncmp(msg,"
                          DRIVE",8)=0 then
          mode = "D"
       end if
       if strncmp(msg,"
                            PWM",8)=0 then
          mode = "P"
       end if
       if strncmp(msg," NORMAL",8)=0 then
          mode = "N"
       end if
        " Threshold High
       if strncmp(msg,"H",1)=0 then
          EEPROM_Write($02, msg[4])
                                              'High Voltage
          Delay_ms(100)
          EEPROM_Write($03, msg[5])
                                              ,
         Delay_ms(100)
          EEPROM_Write($04, msg[6])
          Delay_ms(100)
          EEPROM_Write($05, msg[7])
          Delay_ms(100)
```

```
if f_volt="I" then
f_volt="L"
end if
```

end if

" Threshold Low
if strncmp(msg,"L",1)=0 then
EEPROM\_Write(\$06, msg[4])
Delay\_ms(100)
EEPROM\_Write(\$07, msg[5])
Delay\_ms(100)
EEPROM\_Write(\$07, msg[6])
Delay\_ms(100)
EEPROM\_Write(\$08, msg[7])
Delay\_ms(100)
end if

'Low Voltage

end select end if end sub

```
' USART Rx Interrupt
sub procedure interrupt
'Check for Rx IRQ.
if PIR1.5 = 1 then
rxbuf[rxindex]=RCREG
rxindex = rxindex + 1
if rxindex = 10 then
rxindex=0
rxreq = "*"
rxdata = rxbuf
rxbuf = """
end if
end if
end sub
```

'Reciev 1 character.

' Rx data length = 10 characters. ' Reset buffer index ' Set Rxdata opereation flug ' Copy Rx data ' Reset Rx data buffer

" EEPROM Initialization	
sub procedure EEP_Init()	
EEPROM_Write(\$00, \$AB)	' Check Digit.
Delay_ms(100)	
EEPROM_Write(\$01, \$CD)	' Check Digit.
Delay_ms(100)	

EEPROM\_Write(\$02, "0") ' High Voltage 0615 (12V) Delay\_ms(100) EEPROM\_Write(\$03, "6") Delay\_ms(100) EEPROM\_Write(\$04, "1") Delay\_ms(100) EEPROM\_Write(\$05, "5") Delay\_ms(100) 'Low Voltage 0563 (11V) EEPROM\_Write(\$06, "0") Delay\_ms(100) EEPROM\_Write(\$07, "5") Delay\_ms(100) EEPROM\_Write(\$08, "6") Delay\_ms(100) EEPROM\_Write(\$09, "3") Delay\_ms(100) end sub

sub procedure send\_high\_threshold()

·-----

"High threshold cbuff="High " cbuff[4]=EEPROM\_Read(\$02) Delay\_ms(100) cbuff[5]=EEPROM\_Read(\$03) Delay\_ms(100) cbuff[6]=EEPROM\_Read(\$04) Delay\_ms(100) cbuff[7]=EEPROM\_Read(\$05) Delay\_ms(100)

Lcd\_Out(1,1,"@THRESH ") Lcd\_Out(2,1,cbuff) 'Write text in first row 'Write text in second row

sbuff="H " sbuff[1]=cbuff[4] sbuff[2]=cbuff[5] sbuff[3]=cbuff[6] sbuff[4]=cbuff[7] UART1\_Write\_Text(sbuff) end sub

sub procedure send\_low\_threshold()

if f\_volt="L" then f\_volt="" ۱\_\_\_\_\_ " Low threshold cbuff="Low " cbuff[4]=EEPROM\_Read(\$06) Delay\_ms(100)  $cbuff[5] = EEPROM\_Read($07)$ Delay\_ms(100) cbuff[6]=EEPROM\_Read(\$08) Delay\_ms(100) cbuff[7]=EEPROM\_Read(\$09) Delay\_ms(100) Lcd\_Out(1,1,"@THRESH ") 'Write text in first row Lcd\_Out(2,1,cbuff) 'Write text in second row sbuff="L " sbuff[1]=cbuff[4] sbuff[2]=cbuff[5] sbuff[3]=cbuff[6] sbuff[4]=cbuff[7]UART1\_Write\_Text(sbuff) end if end sub main: 'Main program mode = "N" rxdata = " " rxbuf=" " rxreq = " " rxindex = 0" Clear Port A and B porta=0 portb=0

' PortDirections CMCON = %00000111 TRISA = %00111100 TRISB = %00001111 SPBRG = 25 RCSTA = %10010000 TXSTA = %00100110 OPTION_REG.7 = 0 " Interrupt Enable PIE1 = %00100000 INTCON = %11000000	' USART BAUD RATE Setting. 9600 baud ' USART Rx Status and Control ' USART Tx Status and Control ' Pull up Port B for SW
UART1_Init(9600)	' USART Initialize with 9600 baud
' Indicates LED Blinking ' LED_Blink() LED_Blink() LED_Blink() '	
" LCD Initialize ' Lcd_Init() Lcd_Cmd(_LCD_CLEAR) Lcd_Cmd(_LCD_CURSOR_OFI	' Initialize Lcd ' Clear display F) ' Cursor off
" Display Startting Message Lcd_Out(1,1,"@MONITOR") Lcd_Out(2,1," START") " Send Power On Command. UART1_Write_Text("P")	' Write text in first row ' Write text in second row
" Wait for 2sec. Delay_ms(2000) " Mode Set Request 'Lcd_Out(1,1,"@MODE ") 'Lcd_Out(2,1," SET")	' Write text in first row ' Write text in second row

" Send Mode Set Command.

#### 'UART1\_Write\_Text("M")

```
" EEPROM Check
Lcd_Out(1,1,"@EEPROM ")
                                 'Write text in first row
Lcd_Out(2,1," CHECK")
                                'Write text in second row
eep_temp = EEPROM_Read($00)
Delay_ms(100)
eep_check = (eep_temp << 8) or EEPROM_Read($01)
Delay_ms(100)
if eep_check=$ABCD then
  'Lcd_Out(1,1,"@EEPROM ")
                                   'Write text in first row
 'Lcd_Out(2,1,"
                                 'Write text in second row
                    OK'')
 Lcd_Out(1,1,"@MONITOR")
                                    'Write text in first row
 Lcd_Out(2,1,"
                START")
                                 'Write text in second row
else
                                   'Write text in first row
 Lcd_Out(1,1,"@EEPROM ")
 Lcd_Out(2,1," Init.")
                              'Write text in second row
  EEP_Init()
                                ' EEPROM Initialize
                                    'Write text in first row
 Lcd_Out(1,1,"@MONITOR")
 Lcd_Out(2,1," START")
                                 'Write text in second row
end if
Delay_ms(1000)
Lcd_Out(1,1,"@MONITOR")
                                  'Write text in first row
Lcd_Out(2,1," START")
                               'Write text in second row
Delay_ms(1000)
" Battery threshold: High
f_volt="I"
send_high_threshold()
Delay_ms(1000)
" Main Loop
while TRUE
                                'Endless loop
 " sw1
 SW1\_Proc0
  "sw2
 SW2_Proc0
  " Rx Data
  RxData_Proc0
```

" Battery threshold: Low send\_low\_threshold0

wend

end.

# Appendix 2. Main ボード プログラムリスト

/\*

Copyright (c) 2012-2013, Kenichi Harada wiseman All rights reserved.  $^{\prime\prime}$ 

/\*

\* main.c \*/

#include <stdlib.h>
#include <stdint.h>

#include "GenericTypeDefs.h"
#include "HardwareProfile.h"
#include "USB/usb.h"
#include "usb\_bt\_driver.h"

#include <btstack/run\_loop.h>
#include <btstack/sdp\_util.h>
#include "btstack/src/hci.h"
#include "btstack/src/l2cap.h"
#include "btstack/src/btstack\_memory.h"
#include "btstack/src/rfcomm.h"
#include "btstack/src/sdp.h"

#include "bt\_spp.h"

#include<ports.h> // C:\Program Files\Microchip\MPLAB C30\Support\peripheral\_24F\ports.h

#include<timer.h>

```
/* Configuration Bits */
```

```
#if defined(__PIC24FJ256GB106__)
```

\_CONFIG1(JTAGEN\_OFF & GCP\_OFF & GWRP\_OFF & COE\_OFF & FWDTEN\_OFF & ICS\_PGx1) \_CONFIG2(PLL\_96MHZ\_ON & IESO\_OFF & FCKSM\_CSDCMD & OSCIOFNC\_OFF & POSCMOD\_HS & FNOSC\_PRIPLL & PLLDIV\_DIV4 & IOL1WAY\_ON)

#elif defined(\_\_PIC24FJ64GB002\_\_)

\_CONFIG1(WDTPS\_PS1 & FWPSA\_PR32 & WINDIS\_OFF & FWDTEN\_OFF & ICS\_PGx1 & GWRP\_OFF & GCP\_OFF & JTAGEN\_OFF)

\_CONFIG2(POSCMOD\_NONE & I2C1SEL\_PRI & IOL1WAY\_OFF & OSCIOFNC\_ON & FCKSM\_CSDCMD & FNOSC\_FRCPLL & PLL96MHZ\_ON & PLLDIV\_DIV2 & IESO\_OFF)

# \_CONFIG3(WPFP\_WPFP0 & SOSCSEL\_IO & WUTSEL\_LEG & WPDIS\_WPDIS & WPCFG\_WPCFGDIS & WPEND\_WPENDMEM) \_CONFIG4(DSWDTPS\_DSWDTPS3 & DSWDTOSC\_LPRC & RTCOSC\_SOSC & DSBOREN\_OFF & DSWDTEN\_OFF)

#endif

// Global Variables

BYTE deviceAddress = 0;

static uint8_t	rfcomm_channel_nr = 1;
static uint8_t	<pre>spp_service_buffer[120];</pre>

// -----

#define true	1
#define false	0

//typedef unsigned char BOOL;
//typedef unsigned int WORD;

WORD BHighVolt; WORD BLowVolt; WORD BVolt;

BOOL BreakNow = false; BOOL ChargeNow = false; BOOL Test1 = false; BOOL Test2 = false;

BOOL TestTime; BOOL SaveChargeSw;

char mode = 'N'; float A\_amp = 0.0; float B\_volt = 0.0; float AD\_val = 0.0; char rx = ''; int ix = 0; char Vol[6] = "01234";

double Volt0, Volt1;

// A/D 変換用データバッファ

int tm\_5ms = 0; // 5ms timer int tm\_50ms = 0; // 50ms timer int tm\_500ms = 0;  $/\!/$  500ms timer int tm\_500msbase = 2; // 500ms base int tm\_1sec = 0; // 1sec timer int tm\_10sec = 10; // 10sec timer int tm\_BT = 10; // 10sec timer int tm\_AC = 20; // 10sec timer // Charge mode timer int tm\_charge; // Drive mode timer int tm\_drive; int tm\_queu\_drive; int tm\_mode; int tm\_status; int tm\_duty; int tm\_volt\_h; int tm\_volt\_l; // 簡易モニター用ステータスインデックス int status = 0; int duty = 0; char duty\_f = 'U'; char buf[11] = "0123456789"; char control\_on = 'N'; // USB Event BOOL USB\_ApplicationEventHandler (BYTE address, USB\_EVENT event, void \*data, DWORD size) { // Handle specific events. switch ((INT)event) { case EVENT\_VBUS\_REQUEST\_POWER: // We'll let anything attach. return TRUE; case EVENT\_VBUS\_RELEASE\_POWER: // We aren't keeping track of power. return TRUE; case EVENT\_HUB\_ATTACH: return TRUE; break;

case EVENT\_UNSUPPORTED\_DEVICE: return TRUE; break; case EVENT\_CANNOT\_ENUMERATE: return TRUE; break; case EVENT\_CLIENT\_INIT\_ERROR: return TRUE; break; case EVENT\_OUT\_OF\_MEMORY: return TRUE; break;  $case \ EVENT\_UNSPECIFIED\_ERROR; \quad \ \ // \ This \ should \ never \ be \ generated.$ return TRUE; break; case EVENT\_SUSPEND: case EVENT\_DETACH: case EVENT\_RESUME: case EVENT\_BUS\_ERROR: return TRUE; break; default: break; } return FALSE; }//USB\_ApplicationEventHandler // USBTask static int usbhost\_process(struct data\_source \*ds) // Maintain USB Host State USBHostTasks(); USBHostUSBBTTask(); return 0;

{

}

```
// SW[1]Task
static int sw1_process(struct data_source *ds)
{
          static short swcount;
          static short lastsw;
          if(swcount == 0)
          {
                     short sw = mPORTBReadBit(0x4000);
                     if(sw != lastsw)
                     {
                               if(sw == 0x4000)
                               {
                                         bt_spp_send("S0",2);
                               }else{
                                          bt_spp_send("S1",2);
                                          control_on = 'N';
                                                                                              // 通常制御
停止
                                          mPORTAClearBits(0x0008);
                                                                                   // 停電回復·放電停止
                                          BreakNow = false;
                                          mPORTAClearBits(0x0004);
                                                                                   // 充電 OFF
                                          ChargeNow=false;
                                          printf( "U *CONTROL" );
                                         printf("L
                                                        OFF");
                               }
                               swcount = 1500;
                               lastsw = sw;
                     }
          }
          if(swcount != 0)
          {
                     swcount--;
          }
          return 0;
}
//SW[2]Task 強制停電テスト
static int sw2_process(struct data_source *ds)
{
          static short swcount;
          static short lastsw;
          if(swcount == 0)
          {
                     short sw = mPORTBReadBit(0x0200);
```

	if(sw != la	stsw)		
	{			
		if(sw == 0 {	x0200)	
orts	ト> にあります	,	//mPORTAClearBits(0x0008);	//<このマクロも <p< td=""></p<>
01 ( 3.		}else{	//bt_spp_send("M0",2);	
			<pre>short jp1 = mPORTAReadBit(0x0002); short jp2 = mPORTBReadBit(0x0020);</pre>	//JP1 接続か?:low //JP2 接続か?:low
0 <b>00</b>			if(mode=='N' && jp1==0x0002 && jp2==0x	0020) //jp1 jp2
open			{	
	//<このマクロも <j< td=""><td>ports.</td><td>mPORTASetBits(0x0008); h&gt; にあります。</td><td></td></j<>	ports.	mPORTASetBits(0x0008); h> にあります。	
			bt_spp_send("MD",2); tm_drive = 5*2+1;	//
5Sec				
			mode = 'D';	
			<pre>printf("U *MODE ");</pre>	
			$print(^{L} DRIVE^{n});$	
	//jp1 ショート:バッテ	リ電圧上限	yeise nyp10x0000 && jp20x0020パ を EEPROM 書き込み	
			printf( "U *Battery" );	
			printf( "L High%04u",BVolt );	
		n arter	<pre>}else if(jp1==0x0002 &amp;&amp; jp2==0x0000){</pre>	
	//jp2 ショートシップ	リ竜庄下限	を EEPROM 書さ込み	
			printi("U *Battery"),	
			printi( L Low %04u , B voit ),	
		}	3	
		, swcount =	= 500;	
		lastsw = s	w;	
	}			
	}			
	if(swcount != 0)			
	{			
	swcount	;		
	}			
	return 0;			

}

```
// SW[3]Task 強制充電テスト
static int sw3_process(struct data_source *ds)
{
          static short swcount;
          static short lastsw;
          if(swcount == 0)
          {
                    short sw = mPORTBReadBit(0x0100);
                    if(sw != lastsw)
                    {
                              if(sw == 0x0100)
                              {
                                       //mPORTAClearBits(0x0004);
                                                                               //<---このマクロも <p
orts. h > c = b = c
                             }else{
                                       if(mode=='N')
                                       {
                                                 mPORTASetBits(0x0004);
          //<--このマクロも <ports. h> にあります。
                                                 bt_spp_send("MC",2);
                                                 tm_charge = 5*2+1;
                                                                                                   //
5Sec
                                                 mode = 'C';
                                                 printf("U *MODE ");
                                                 printf("L CHARGE");
                                       }
                              }
                              swcount = 500;
                              lastsw = sw;
                   }
          }
          if(swcount != 0)
          {
                    swcount--;
          }
         return 0;
}
```

// HB LED task ---> 忙しいとき、LEDが点いている時間が長くなる
```
static int led_process(struct data_source *ds)
ł
         mPORTAClearBits(0x0010);
                                              //<---HB LED OFF
         return 0;
}
// USART Rx task ---> シリアルポート1から1文字のデータを受信する
static int rx_process(struct data_source *ds)
{
         if(U1STAbits.URXDA == 1)
         {
                  // 受信エラーチェック
                  if(U1STAbits.OERR | | U1STAbits.FERR){
                            // エラーのときは、エラーフラグをキャンセルして、
                            // UARTを一度停止して、再度有効にします。
                            U1STA &= 0xFFF0;
                                                        // Error cancel
                            U1MODEbits.UARTEN = 0;
                                                        // Stop UART1
                            U1MODEbits.UARTEN = 1;
                                                        // Restart UART1
                  }else{
                            rx = getchar(0;
                                                                 // Recive 1 chara.
                            switch(rx)
                            {
                                              // Status Request
                                     case 'S':
                                               tm_status = 2;
                                               control_on = 'G';
                                                                          // 最初の STATUS 要求
//
で、システム制御開始 GO とします。
                                               break;
                                     case 'M': // Mode Change Request
                                               tm_mode = 2;
                                               break;
                                              // Duty Change Request
                                     case 'D':
                                               tm_duty = 2;
                                               break;
                                     case 'H':
                                              // High Threshold
                                               Vol[0]='H';
                                               ix=1;
                                               break;
```

```
// Low Threshold
                                       case 'L':
                                                Vol[0]='L';
                                                ix=1;
                                                break;
                                       default:
                                                Vol[ix]=rx;
                                                if(ix>=4){
                                                          ix=0;
                                                          if(Vol[0]=='H'){
                                                                    tm_volt_h=2;
                                                                    mPORTBSetBits(0x8000);
                                                                                                //
LED ON
                                                          }else{
                                                                    tm_volt_l=2;
                                                                    mPORTBClearBits(0x8000);
                                                                                                //
LED OFF
                                                         }
                                                }else{
                                                          ix++;
                                                }
                             }// end switch
                   }
         }
         return 0;
}
// メインコントロール タスク---->充電・放電制御をおこないます。
static int main_control_process(struct data_source *ds)
{
          // main control flug が GO でかつ、mode が Normal のときだけ制御する
          if(control\_on == 'G' \&\& mode == 'N') \{
                   short ibreak = mPORTBReadBit(0x0080);
                   if(ibreak == 0x0080) // RB7 が1のとき停電検出 通常は0
                   {
                             mPORTASetBits(0x0008);
                                                                    // 放電開始
                             if(BreakNow==false){
                                      // 停電中!
                                       BreakNow = true;
                                                                   // 停電検出メッセージ --->
                                       bt_spp_send("BD",2);
Android
                                       printf("U *BREAK ");
```

```
printf("L DETECT");
                         }
                 }else{
                                                   // 停電回復·放電停止
                         mPORTAClearBits(0x0008);
                         if(BreakNow==true)
                         {
                                  BreakNow = false;
                                  bt_spp_send("BR",2);
                                                           // 停電回復メッセージ --->
Android
                                  printf("U *BREAK ");
                                  printf( "L RECOVERE" );
                         }
                 }
                 // バッテリ電圧を上限値・下限値と比較して、充電制御をおこないます。
                 if(BVolt < BLowVolt){
                         // VLよりバッテリ電圧が低いとき
                         mPORTASetBits(0x0004);
                                                                    // 充電 ON
                         if(ChargeNow==false){
                                  // 充電中になったときだけ、メッセージを送ります
                                  ChargeNow=true;
                                  bt_spp_send("CD",2);
                                                           // 充電開始メッセージ --->
Android
                                  printf("U *CHARGE");
                                  printf("L
                                               ON");
                         }
                 }else if(BVolt > BHighVolt){
                         // VH よりバッテリ電圧が高いとき
                         mPORTAClearBits(0x0004);
                                                           // 充電 OFF
                         if(ChargeNow==true){
                                  // 充電完了になったときだけ、メッセージを送ります
                                  ChargeNow=false;
                                  bt_spp_send("CR",2);
                                                           // 充電終了メッセージ --->
Android
                                  printf("U *CHARGE");
                                  printf("L
                                              OFF");
                         }
                 }
        }
        return 0;
}
```

// main
int main ( void )
{

// Initialize the processor and peripherals.

// Init Clock // CPU 32MHz // Peripheral 8MHz CLKDIV = 0x0000; // クロック 1:1 分周

> unsigned int pll\_startup\_counter = 600; CLKDIVbits.PLLEN = 1; while(pll\_startup\_counter--);

/// ポート割り当て

AD1PCFG = 0xFFCF;	// ポートA4-5をアナログに設定
// Analog IN Disable	// 元の Bluetooth プログラムでは Disable だった。
//AD1PCFG = 0xffff;	

TRISA = 0xFFE3;	// RA2-4 を出力に指定。LED 用
TRISB = 0x7FEF;	//RB15,4 を出力に指定。

// RB15:LED

// RB4:Tx として使う。

<--- RP4 で使用するので、この指定は要らないかも知れない。

//mPORTAOutputConfig(0xFFE3); //<---このマクロも <ports. h> にあります。 //mPORTBOutputConfig(0x7FEF); //<---このマクロも <ports. h> にあります。

// プルアップは、<ports.h> にマクロがあり、ピンごとにプルアップ、プルダウンの指定が可能。
 // 詳細は、main.c の先頭部分で #include している部分に、ports.hの在処があるので
 // そちらを参照してください。
 // SW 用 PIN プルアップ
 EnablePullUpCN12; // SW プルアップ
 EnablePullUpCN21;
 EnablePullUpCN22;
 EnablePullUpCN23; // 停電検出プルアップ

// JP 用 PIN プルアップ EnablePullUpCN3; EnablePullUpCN27;

/// シリアルポートの指定 // U1RX を RP13 (pin 23) に指定 RPINR18bits.U1RXR = 13;

> #U1TX を RP4 (pin 11) に指定 RPOR2bits.RP4R = 3;

// Init UART U1BRG = 103; // 9600bps @ 8MHz U1MODE = 0b10001000000000; // USART1 初期設定 U1STA = 0b000001000000000;

// Timer 2 T2CON = 0b100000000110000; PR2 = 32149; // 500ms IPC1bits.T2IP = 5; // IQR Level 5. IEC0bits.T2IE = 1; // 割り込み許可

// Timer 3 T3CON = 0b100000000110000; PR3 = 3125; // 50ms //IPC2bits.T3IP = 6; // IQR Level 6. //IEC0bits.T3IE = 1; // 割り込み許可

// Timer 4 T4CON = 0b100000000110000; PR4 = 312;// 5ms IPC6bits.T4IP = 6; // IQR Level 6. IEC1bits.T4IE = 1; // 割り込み許可

#ADC
AD1CON1 = 0x8044; #AD オン、整数、タイマー3トリガー、自動サンプリング
#AD1CON2 = 0x2414; #外部 Vref+, AVss,自動スキャン、

AD1CON2 = 0x0414; AD1CON3 = 0x1F05; AD1CHS = 0x0000; AD1PCFG = 0xFFCF; AD1CSSL = 0x0030;IEC0bits.AD1IE = 1; // AVdd, AVss, 自動スキャン、
// 31Tad, ad ,5\*Tcy
// 自動スキャン
// RA4,5 Analog
// AN4,5 自動スキャン
// AD 割り込み許可

BHighVolt=615; BLowVolt=563; // デフォルト上限値 12V// 下限値 11V

//for test Brink LED //for test Brink LED //for test Brink LED mPORTBSetBits(0x8000); //<---このマクロも <ports. h> にあります。 DelayMs(250); mPORTBClearBits(0x8000); //<---このマクロも <ports. h> にあります。 DelayMs(250);mPORTBSetBits(0x8000); DelayMs(250); mPORTBClearBits(0x8000); DelayMs(250); mPORTBSetBits(0x8000); DelayMs(250); mPORTBClearBits(0x8000); DelayMs(250); mPORTBSetBits(0x8000);

DelayMs(250); mPORTBClearBits(0x8000); DelayMs(250); mPORTBSetBits(0x8000); DelayMs(250); mPORTBClearBits(0x8000); //for test Brink LED //for test Brink LED //for test Brink LED

printf( "U MAIN CPU" );
printf( "L >> START" );

```
mPORTASetBits(0x001C);
DelayMs(250);
mPORTAClearBits(0x001C);
DelayMs(250);
mPORTASetBits(0x001C);
DelayMs(250);
mPORTAClearBits(0x001C);
DelayMs(250);
```

/\* while(1) { if(mPORTBReadBit(0x4000) == 0x0000) { mPORTASetBits(0x001C); printf("U 12345678"); while (mPORTBReadBit(0x4000) = 0x0000);}else{ mPORTAClearBits(0x001C); } } \*/ // Init USB if (USBHostInit(0) != TRUE) { mPORTASetBits(0x0010); DelayMs(250);

mPORTAClearBits(0x0010); DelayMs(250); mPORTASetBits(0x0010); DelayMs(250); mPORTAClearBits(0x0010); DelayMs(250); mPORTASetBits(0x0010); DelayMs(250); mPORTAClearBits(0x0010); DelayMs(250);

printf( "U USB HOST" ); printf( "L ERROR" ); //while (1);

}

/// GET STARTED with BTstack ///
btstack\_memory\_init();
run\_loop\_init(RUN\_LOOP\_EMBEDDED);

#### // init HCI

hci\_transport\_t \* transport = hci\_transport\_usb\_instance(); bt\_control\_t \* control = NULL; hci\_uart\_config\_t \* config = NULL; remote\_device\_db\_t \* remote\_db = NULL;//(remote\_device\_db\_t \*) & remote\_device\_db\_memory; hci\_init(transport, config, control, remote\_db);

// ここは、Bluetooth を PIC SPP で使うときの、データパケットの扱いを初期化しています。 // init L2CAP l2cap\_init(); l2cap\_register\_packet\_handler(bt\_packet\_handler);

// init RFCOMM

```
rfcomm_init();
```

rfcomm\_register\_packet\_handler(bt\_packet\_handler);

rfcomm\_register\_service\_internal(NULL, rfcomm\_channel\_nr, 100); // reserved channel, mtu=100

// init SDP, create record for SPP and register with SDP
sdp\_init();

memset(spp\_service\_buffer, 0, sizeof(spp\_service\_buffer));

service\_record\_item\_t \* service\_record\_item = (service\_record\_item\_t \*) spp\_service\_buffer;

sdp\_create\_spp\_service((uint8\_t\*) & service\_record\_item > service\_record, 1, "SPP");

```
\label{eq:spectrum} \textit{"Image: SDP service buffer size: $\uxestimes wuxestimes the size of $\uxestimes t_1 + de_get_len((uint8_t^*) + de_get_len(
```

&service\_record\_item->service\_record)));

sdp\_register\_service\_internal(NULL, service\_record\_item);

// usbhost Bluetooth 受信タスクを登録 data\_source\_t usbhost; usbhost.process = &usbhost\_process; run\_loop\_add\_data\_source(&usbhost);

// Bluetooth 通信テストタスク(SW1 でタブレット側 LED 表示制御)を登録(送信) data\_source\_t sw1task; sw1task.process = &sw1\_process; run\_loop\_add\_data\_source(&sw1task);

// 強制駆動テストタスクを登録 data\_source\_t sw2task; sw2task.process = &sw2\_process; run\_loop\_add\_data\_source(&sw2task);

// 強制充電テストタスクを登録
 data\_source\_t sw3task;
 sw3task.process = &sw3\_process;
 run\_loop\_add\_data\_source(&sw3task);

// ハートビート LED タスクを登録 data\_source\_t ledtask; ledtask.process = &led\_process; run\_loop\_add\_data\_source(&ledtask);

// シリアル受信タスクを登録
 data\_source\_t rxtask;
 rxtask.process = &rx\_process;
 run\_loop\_add\_data\_source(&rxtask);

|| メイン制御タスクを登録

data\_source\_t controltask; controltask.process = &main\_control\_process; run\_loop\_add\_data\_source(&controltask);

// go! run\_loop\_execute();

```
return 0;
}//----- main
/*
 AD 割り込み処理 2ch 3 回 変換後平均を計算
*/
void __attribute__((interrupt, auto_psv)) _ADC1Interrupt(void)
£
                                 //AD割り込みフラグクリア
        IFS0bits.AD1IF = 0;
        Volt0 = ((ADC1BUF0 + ADC1BUF2 + ADC1BUF4)*20.)/3072.0;
                                                               //AD4:バッテリ電圧
        Volt1 = (ADC1BUF1 + ADC1BUF3 + ADC1BUF5)*51.2/3072;
                                                               //AD5:AC電流(1bit=0.05A)
        BVolt = (ADC1BUF0 + ADC1BUF2 + ADC1BUF4)/3;
                                                               // EEPROM 書き込み用のバ
イナリ値をとっておく
                                                  // この値で、バッテリの充電制御もおこなう
        if(BVolt >= 1024){BVolt=1024;}
        #AD 値リミット閾値を設けることができる(任意の値で比較する)
}
\parallel
// タイマー1割り込み処理
  ---> システムで使用している
//
//
//void __attribute__((interrupt, no_auto_psv)) _T1Interrupt(void)
IK
||}
// タイマー2割り込み処理
   --->タイマー2割り込みでHB LEDを点灯し、タスクで消灯する。
//
\parallel
        main loop 処理が忙しいとき、点いている時間が長くなる
\parallel
void __attribute__((interrupt, no_auto_psv)) _T2Interrupt(void)
        IFSObits.T2IF = 0;
                                 // 割り込みフラグ クリア
        mPORTASetBits(0x0010);
                                 // HB LED 点灯
        if(tm_500ms>0){tm_500ms--;}
        if(tm_volt_h>0)
        {
                tm_volt_h--;
```

```
if(tm_volt_h==0)
         {
                   // Recieve high threshold
                   strcpy(buf,"4321");
                   buf[0] = Vol[1];
                                                        // Recive 1 chara.
                   buf[1] = Vol[2];
                                                        // Recive 1 chara.
                   buf[2] = Vol[3];
                                                        // Recive 1 chara.
                   buf[3] = Vol[4];
                                                         // Recive 1 chara.
                   buf[4] = 'Yn';
                   BHighVolt = atoi(buf);
                                               // String to value
// デフォルトで制御する(異常な値にされると、充電制御がおかしくなり
// 見た目では PG エラーか、壊れたのか、設定値が正しくないのかの、判断がつかない)
                                     // デフォルト上限値 12V
BHighVolt=615;
                                     // 下限值 11V
BLowVolt=563;
                   printf("U *Battery");
                   printf("L High%4u",BHighVolt);
         }
}
if(tm_volt_l>0)
{
         tm_volt_l--;
         if(tm_volt_l==0)
         {
                   // Recieve high threshold
                   strcpy(buf,"4321");
                   buf[0] = Vol[1];
                                                        // Recive 1 chara.
                   buf[1] = Vol[2];
                                                        // Recive 1 chara.
                   buf[2] = Vol[3];
                                                         // Recive 1 chara.
                   buf[3] = Vol[4];
                                                        // Recive 1 chara.
                   buf[4] = 'Yn';
                   BLowVolt = atoi(buf);
                                               // String to value
// デフォルトで制御する(異常な値にされると、充電制御がおかしくなり
// 見た目では PG エラーか、壊れたのか、設定値が正しくないのかの、判断がつかない)
BHighVolt=615;
                                     // デフォルト上限値 12V
                                     // 下限值 11V
BLowVolt=563;
```

```
printf("U *Battery");
                    printf("L Low %4u",BLowVolt);
          }
}
if(tm_charge>0)
{
          tm_charge--;
          if(tm_charge==0)
          {
                    // Stop Charge
                    mPORTAClearBits(0x0004);
                    bt_spp_send("MN",2); // Send to Android by Bluetooth
                    mode = 'N';
                    printf("U *MODE ");
                    printf("L NORMAL");
          }
}
if(tm_drive>0)
{
          tm_drive--;
          if(tm_drive==0)
          {
                    // Stop Drive
                    mPORTAClearBits(0x0008);
                    bt_spp_send("MN",2); // Send to Android by Bluetooth
                    mode = 'N';
                    printf("U *MODE ");
                    printf("L NORMAL");
          }
}
if(tm_status > 0)
{
```

```
tm_status--;
if(tm_status==0)
          //
          status++;
          switch(status)
          {
                     case 1:
                                printf("U *Battery");
                                printf("L%7.1fV",Volt0);
                                break;
                     case 2:
                                printf( "U *AC Amp." );
                                printf("L%7.1fA",Volt1);
                                break;
                     case 3:
                                printf("U *Battery");
                                printf( "L %7.1fV",Volt0 );
                                break;
                     case 4:
                                printf( "U *AC Amp." );
                                printf("L%7.1fA",Volt1);
                                break;
                     case 5:
                                printf("U *Battery");
                                printf("L%7.1fV",Volt0);
                                break;
                     case 6:
                                printf( "U *AC Amp." );
                                printf("L%7.1fA",Volt1);
                                break;
                     case 7:
                                printf("U *STATUS");
                                               007");
                                printf("L
                                break;
                     case 8:
                                printf("U *STATUS");
                                printf("L
                                               008");
                                break;
                     case 9:
                                printf("U *STATUS");
                                printf("L
                                               009");
                                break;
                     case 10:
                                control_on = 'G';
                                                                // 最初の10回目
```

{

```
STATUS 要求で、システム制御開始 GO とします。
                                               status = 0;
                                               printf( "U *STATUS " );
                                               printf("L
                                                            010");
                                               break;
                            }
                  }
         }
         if(tm_mode>0)
         {
                  tm_mode--;
                  if(tm_mode==0)
                  {
                            if(mode=='N'){
                                     mode = 'D';
                                     mPORTASetBits(0x0008);
                                                                                    //<---このマ
クロも <ports. h> にあります。
                                     bt_spp_send("MD",2);
                                     tm_drive = 5*2+1;
                                                                                    /\!/\,5\mathrm{Sec}
                                     printf("U *MODE ");
                                     printf("L DRIVE");
                            }else if(mode=='D'){
                                     mode = 'P';
                                     duty = 0;
                                     duty_f = 'U';
                                     tm_drive=0;
                                     mPORTAClearBits(0x0008);
                                                                                    //<---このマ
クロも <ports. h> にあります。
                                     bt_spp_send("MP",2);
                                     printf("U *MODE ");
                                     printf("L
                                                  PWM");
                            }else if(mode=='P'){
                                     mode = 'N';
                                     mPORTAClearBits(0x0008);
                                                                                    //<---このマ
クロも <ports. h> にあります。
                                     bt_spp_send("MN",2);
```

```
printf("U *MODE ");
                               printf("L NORMAL");
                    }
          }
}
if(tm_duty > 0)
{
          tm_duty--;
          if(tm_duty==0)
          {
                     if(mode=='P'){
                               if(duty<80 && duty_f=='U'){
                                         duty+=10;
                               }else if(duty>=80 && duty<95 && duty_f=='U'){
                                         duty+=1;
                               else if(duty==95 \&\& duty_f=='U') \{
                                         duty=94;
                                         duty_f='D';
                               }else if(duty<=94 && duty>80 && duty_f=='D'){
                                         duty--;
                               }else if(duty<=80 && duty>0 && duty_f=='D'){
                                         duty-=10;
                               }else if(duty==0 && duty_f=='D'){
                                         duty=10;
                                         duty_f='U';
                               }
                               sprintf(buf,"DU%2u%%",duty);
                               bt_spp_send(buf,5);
                               printf("U *DUTY ");
                               printf("L
                                             %3u%%",duty);
                    }
          }
}
// 500ms base
if(tm_500msbase > 0){tm_500msbase -;}
if(tm_500msbase==0)
{
          tm_500msbase = 2; // 1sec
```

```
if(tm_1sec >0){
                             tm_1sec--;
                             if(tm_1sec==0){tm_1sec=1;}
                   }
                   if(tm_10sec >0){
                             tm_10sec--;
                             if(tm_10sec==0){tm_10sec=0;}
                   }
                   if(tm_BT > 0){
                             tm_BT--;
                             if(tm_BT==0){
                                      tm_BT=20;
                                      sprintf(buf,"BA%4.1fV",B_volt);
                                      bt_spp_send(buf,7);
                                      B_volt = Volt0;
         // 送った直後に最新の値にしておく
                             }
                   }
                   if(tm_AC > 0){
                             tm_AC--;
                             if(tm_AC==0){
                                      tm_AC=20;
                                      sprintf(buf,"AC%4.1fA",A_amp);
                                      bt_spp_send(buf,7);
                                      A_amp = Volt1;
         // 送った直後に最新の値にしておく
                            }
                   }
         }
}
//
// タイマー3割り込み処理
    ---> PWMで使用している
//
//
void __attribute__((interrupt, no_auto_psv)) _T3Interrupt(void)
{
                                      // 割り込みフラグ クリア
          IFS0bits.T3IF = 0;
          tm_50ms++;
          if(tm_50ms>=20){
                   tm_50ms=0;
         }
//
         if(mode=='P')
```

```
{
//
                    if(tm_50ms==0){mPORTBSetBits(0x8000);}
//
                    if(tm_50ms ==(duty/5)){
\parallel
                              mPORTBClearBits(0x8000);
//
                    }
\parallel
          }
//
}
\parallel
   タイマー4割り込み処理
//
     ---> PWMで使用している
//
//
void __attribute__((interrupt, no_auto_psv)) _T4Interrupt(void)
{
          IFS1bits.T4IF = 0;
                                        // 割り込みフラグ クリア
          tm_5ms++;
          if(tm_5ms>=20){
                    tm_5ms=0;
          }
          if(mode=='P')
          {
                    if(tm_5ms==0){mPORTASetBits(0x0008);}
                    if(tm_5ms ==(duty/5)){
                              mPORTAClearBits(0x0008);
                    }
          }
}
/*
Copyright(c)2012-2013 kenichi harada Wiseman Corp. All Rights Reserved.
*/
/*
 * bt_spp.c
 */
#include <string.h>
```

#include <stdio.h>

#include <btstack/hci\_cmds.h>
#include <btstack/run\_loop.h>
#include <btstack/sdp\_util.h>

#include "btstack/src/hci.h"
#include "btstack/src/l2cap.h"
#include "btstack/src/btstack\_memory.h"
#include "btstack/src/remote\_device\_db.h"
#include "btstack/src/rfcomm.h"
#include "btstack/src/sdp.h"
#include "btstack/include/config.h"

#include "bt\_spp.h"

#define USE\_AND\_OR /\* To enable AND\_OR mask setting \*/ #include<ports.h>

static uint16\_t rfcomm\_channel\_id;

uint16\_t bt\_localid; char bt\_localname[10] = {'P','I','C','B','T','0','0','0','0',0x00}; char send\_buf[17];

/\* SPP RECIVE DATA \*/ int bt\_spp\_recive\_callback(uint16\_t channel, uint8\_t \*packet, uint16\_t size) {

```
static char a = 0;
//
        packet[size] = 0;
        switch(*(packet+0))
        {
                 case 'l':
                 case 'L':
                         if(*(packet+1) == '1')
                         {
                                  mPORTBSetBits(0x8000);
                         }else{
                                  mPORTBClearBits(0x8000);
                         }
                         control_on = 'N';
                                                                   // 通常制御停止
                         mPORTAClearBits(0x0008);
                                                                   // 停電回復·放電停止
                         BreakNow = false;
                         mPORTAClearBits(0x0004);
                                                                    // 充電 OFF
                         ChargeNow=false;
                         break;
                 case 'S': // Status Request
                         tm_status = 2;
                         break;
                 case 'M': // Mode Change Request
                         tm mode = 2;
                         break;
                 case 'D': // Duty Change Request
                         tm_duty = 2;
                         break;
                 case 'C': // Duty Change Request
                         if(mode=='N')
                         {
                                  mPORTASetBits(0x0004);
                                                                             //<---この
マクロも <ports. h> にあります。
                                  bt_spp_send("MC",2);
                                  tm_charge = 5*2+1;
                                                                             // 5Sec
                                  mode = 'C';
                                  printf("U *MODE ");
                                  printf("L CHARGE");
```

} break; } return 0;

# Appendix 3. Android タブレットプログラムリスト

package com.hrdapp.android.BluetoothIO;

//import android.bluetooth.BluetoothAdapter; //import android.bluetooth.BluetoothDevice; import android.app.Activity; import android.content.Intent; import android.os.Bundle; //import android.util.Log; import android.view.Menu; import android.view.MenuInflater; import android.view.MenuItem; import android.view.View; import android.view.View.OnClickListener; import android.view.Window; import android.widget.CompoundButton; import android.widget.TextView; import android.widget.Toast; import android.widget.ToggleButton; import android.widget.Button; import android.widget.ProgressBar;

public class BluetoothIOActivity extends BluetoothBaseActivity implements ToggleButton.OnCheckedChangeListener{

// Layout Views private TextView mTitle; private TextView cTitle; private TextView mSWStatus; private ToggleButton mToggleButton; private Button mButton1; // モード変更ボタン // 停電テストボタン private Button mButton2; private Button mButton3; // 充電テストボタン private ProgressBar PBar; || プログレスバー || プログレスバー private ProgressBar PBar2; || プログレスバー private ProgressBar PBar3; // ノーマルモード表示 private TextView ModeNormal; // ドライブモード表示 private TextView ModeDrive; private TextView ModePWM; // PWM モード表示 private TextView Voltage; // バッテリ電圧 // AC電流 private TextView Ampere; private TextView Duty; // Duty

//private final static int WC=LinearLayout.LayoutParams.WRAP\_CONTENT;

## @Override

public void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);

// Set up the window layout
requestWindowFeature(Window.FEATURE\_CUSTOM\_TITLE);
setContentView(R.layout.main);
getWindow().setFeatureInt(Window.FEATURE\_CUSTOM\_TITLE, R.layout.custom\_title);

// Set up the custom title
mTitle = (TextView) findViewById(R.id.title\_left\_text);
mTitle.setText(R.string.app\_name);
mTitle = (TextView) findViewById(R.id.title\_right\_text);

cTitle = (TextView) findViewById(R.id.cTitle); cTitle.setHeight(80); cTitle.setTextSize(60);

mSWStatus = (TextView) findViewById(R.id.SWStatus); mSWStatus.setHeight(100); mSWStatus.setTextSize(80); mSWStatus.setBackgroundColor(getResources().getColor(R.color.magenta));

mToggleButton = (ToggleButton)findViewById(R.id.LEDtoggleButton); mToggleButton.setChecked(false); mToggleButton.setOnCheckedChangeListener(this); mToggleButton.setHeight(120); mToggleButton.setTextSize(80);

mButton1 = (Button)findViewById(R.id.button1); mButton1.setTag("button1"); mButton1.setOnClickListener(new BtnClickListener()); mButton1.setHeight(120); mButton1.setWidth(430); mButton1.setTextSize(75); mButton1.setTextColor(getResources().getColor(R.color.blue));

mButton2 = (Button)findViewById(R.id.button2);

mButton2.setTag("button2"); mButton2.setOnClickListener(new BtnClickListener()); mButton2.setHeight(120); mButton2.setWidth(430); mButton2.setTextSize(75); mButton2.setTextColor(getResources().getColor(R.color.blue));

mButton3 = (Button)findViewById(R.id.button3); mButton3.setTag("button3"); mButton3.setOnClickListener(new BtnClickListener()); mButton3.setHeight(120); mButton3.setWidth(430); mButton3.setTextSize(75); mButton3.setTextColor(getResources().getColor(R.color.blue));

PBar = (ProgressBar) findViewById(R.id.progressBar1); PBar.setMax(100); PBar.setProgress(0); PBar.setSecondaryProgress(80); //PBar.setBackgroundColor(getResources().getColor(R.color.lime));

PBar2 = (ProgressBar) findViewById(R.id.progressBar2); PBar2.setMax(100); PBar2.setProgress(0); PBar2.setSecondaryProgress(80);

PBar3 = (ProgressBar) findViewById(R.id.progressBar3); PBar3.setMax(100); PBar3.setProgress(0); PBar3.setSecondaryProgress(80);

ModeNormal = (TextView) findViewById(R.id.textView1); ModeNormal.setHeight(100); ModeNormal.setWidth(430); ModeNormal.setTextSize(80); ModeNormal.setBackgroundColor(getResources().getColor(R.color.magenta)); ModeNormal.setTextColor(getResources().getColor(R.color.blue));

ModeDrive = (TextView) findViewById(R.id.textView2); ModeDrive.setHeight(100); ModeDrive.setWidth(430); ModeDrive.setTextSize(80);  $ModeDrive.setBackgroundColor(getResources0.getColor(R.color.amethyst));\\ModeDrive.setTextColor(getResources0.getColor(R.color.white));\\$ 

ModePWM = (TextView) findViewById(R.id.textView3); ModePWM.setHeight(100); ModePWM.setWidth(430); ModePWM.setTextSize(80); ModePWM.setBackgroundColor(getResources().getColor(R.color.coral)); ModePWM.setTextColor(getResources().getColor(R.color.white));

Voltage = (TextView) findViewById(R.id.textView4); Voltage.setHeight(100); Voltage.setWidth(430); Voltage.setTextSize(80); Voltage.setBackgroundColor(getResources().getColor(R.color.almond)); Voltage.setTextColor(getResources().getColor(R.color.lime));

Ampere = (TextView) findViewById(R.id.textView5); Ampere.setHeight(100); Ampere.setWidth(430); Ampere.setTextSize(80); Ampere.setBackgroundColor(getResources().getColor(R.color.almond)); Ampere.setTextColor(getResources().getColor(R.color.lime));

Duty = (TextView) findViewById(R.id.textView6); Duty.setHeight(100); Duty.setWidth(430); Duty.setTextSize(80); Duty.setBackgroundColor(getResources().getColor(R.color.almond)); Duty.setTextColor(getResources().getColor(R.color.lime));

```
if(!initBT0)
{
    Toast.makeText(this, "Bluetooth is not available", Toast.LENGTH_LONG).show();
    finish();
}
```

```
class BtnClickListener implements OnClickListener {
```

```
//@Override
    public void onClick(View v){
      String tag=(String)v.getTag();
      if (tag=="button1"){
                sendMessage("M");
                //Toast.makeText(v.getContext(),tag,Toast.LENGTH_SHORT).show();
                mSWStatus.setText("モード変更ボタンがおされました");
      }else if (tag=="button2"){
                sendMessage("C");
                //Toast.makeText(v.getContext(),tag,Toast.LENGTH_SHORT).show();
                mSWStatus.setText("充電テストぼたんがおされました");
      }else if (tag=="button3"){
                sendMessage("D");
                //Toast.makeText(v.getContext(),tag,Toast.LENGTH_SHORT).show();
                mSWStatus.setText("DUTY 変更ボタンがおされました");
      }
   }
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.option_menu, menu);
    return true;
@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
    case R.id.connect:
       // Launch the DeviceListActivity to see devices and do scan
       Intent serverIntent = new Intent(this, DeviceListActivity.class);
       startActivityForResult(serverIntent, REQUEST_CONNECT_DEVICE);
       return true;
    case R.id.disconnect:
      stopBTIO0;
        return true;
   }
```

}

return false;

## }

```
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
    case REQUEST_CONNECT_DEVICE:
        // When DeviceListActivity returns with a device to connect
        if (resultCode == Activity.RESULT_OK) {
            // Get the device MAC address
            String address = data.getExtras()
                                  .getString(DeviceListActivity.EXTRA_DEVICE_ADDRESS);
            connectionBT(address);
        }
        break;
    case REQUEST_ENABLE_BT:
        // When the request to enable Bluetooth returns
        if (resultCode == Activity.RESULT_OK) {
            // Bluetooth is now enabled, so set up a io session
            setupBTIO0;
        } else {
            // User did not enable Bluetooth or an error occured
            Toast.makeText(this, R.string.bt_not_enabled_leaving, Toast.LENGTH_SHORT).show();
            finish();
        }
   }
@Override
public void on Change Connection Status (int status) {
          switch (status) {
          case BluetoothBaseService.STATE_CONNECTED:
              mTitle.setText(R.string.title_connected_to);
              mTitle.append(getmConnectedDeviceName());
        Toast.makeText(getApplicationContext(), "Connected to "
                + getmConnectedDeviceName(), Toast.LENGTH_SHORT).show();
              break;
          case BluetoothBaseService.STATE_CONNECTING:
              mTitle.setText(R.string.title_connecting);
              break;
          case BluetoothBaseService.STATE LISTEN:
          case \ Blue to oth Base Service. STATE\_NONE :
              mTitle.setText(R.string.title_not_connected);
```

break; }

}

```
// Tauch Button and chenge status
   public void onCheckedChanged(CompoundButton_tbutton,boolean_checked){
                   // LED ON/OFFボタンが押されました。
                   if( checked){
                             // Send message LED ON Command
         sendMessage("L1");
         Toast.makeText(this,"LED ONです",Toast.LENGTH_SHORT).show();
                   }else{
                             //Send message LED OFF command
         sendMessage("L0");
         Toast.makeText(this,"LED OFFです",Toast.LENGTH_SHORT).show();
                   }
         }
         // with Bluetooth rechive message from PIC
   @Override
   public void onReviceMessage(String message)
   ł
         if(message.equals("S1")){
                   // SW ON Command
                   mSWStatus.setText(R.string.SW_ON);
                   Toast.makeText(this,"SWが押されました",Toast.LENGTH_SHORT).show();
         }else if(message.equals("S0")){
                   // SW OFF command
                   mSWStatus.setText(R.string.SW_OFF);
                   Toast.makeText(this,"SWが離されました",Toast.LENGTH_SHORT).show();
         }else if(message.substring(0,2).equals("MD")){
                   // Mode change to Drive
           ModeNormal.setTextColor(getResources().getColor(R.color.white));
           ModeDrive.setTextColor(getResources().getColor(R.color.blue));
           ModePWM.setTextColor(getResources().getColor(R.color.white));
                   //Toast.makeText(this,"モードが DRIVE になりました
",Toast.LENGTH_SHORT).show();
           mSWStatus.setText("モード DRIVE です");
         }else if(message.substring(0,2).equals("MN")){
                   // Mode change to Normal
           ModeNormal.setTextColor(getResources().getColor(R.color.blue));
           ModeDrive.setTextColor(getResources().getColor(R.color.white));
           ModePWM.setTextColor(getResources().getColor(R.color.white));
                   ModePWM.setText("PWM");
                   Duty.setText("0%");
```

PBar.setProgress(0);
//Toast.makeText(this,"モードが NORMAL になりました
",Toast.LENGTH_SHORT).show();
mSWStatus.setText("モード NORMAL です");
}else if(message.substring(0,2).equals("MP")){
// Mode change to PWD
ModeNormal.setTextColor(getResources().getColor(R.color.white));
ModeDrive.setTextColor(getResources().getColor(R.color.white));
ModePWM.setTextColor(getResources().getColor(R.color.blue));
//Toast.makeText(this,"モードが PWM になりました",Toast.LENGTH_SHORT).show();
mSWStatus.setText("モード PWM です");
}else if(message.substring(0,2).equals("MC")){
// Charge On
ModePWM.setText("CHARG");
ModeNormal.setTextColor(getResources().getColor(R.color.white));
ModeDrive.setTextColor(getResources().getColor(R.color.white));
ModePWM.setTextColor(getResources().getColor(R.color.blue));
//Toast.makeText(this,"充電テスト中です",Toast.LENGTH_SHORT).show();
mSWStatus.setText("充電テスト中です");
}else if(message.substring(0,2).equals("DU")){
// Duty value
Duty.setText(message.substring(2));
PBar.setProgress(Integer.parseInt(message.substring(2,4).replace("", "0"))); //プログレスバ
PBar2.setProgress(Integer.parseInt(message.substring(2,4).replace("", "0"))); //ブログレス
PBar3.setProgress(Integer.parseInt(message.substring(2,4).replace("", "0"))); //フロクレス バーの信亦重
mSWStatus actTaut(massaga):
In Swistatus.set TextInessage/,
massage substring(2) Toast LENGTH SHORT) show():
message.substring(2), hast.Hin (GTT_STONT).snow() mSWStatus.setText("DUTYが変更されました "+ message.substring(2));
}else if(message.substring(0,2).equals("BA")){
// Battery voltage
Voltage.setText(message.substring(2));
//Toast.makeText(this,"バッテリー電圧 "+
message.substring(2),Toast.LENGTH_SHORT).show();
mSWStatus.set'lext("バッテリー電圧 "+ message.substring(2));
}else if(message.substring(0,2).equals("AC")){
// AC current

```
Ampere.setText(message.substring(2));
                 //Toast.makeText(this,"AC電流 "+
message.substring(2),Toast.LENGTH_SHORT).show();
          mSWStatus.setText("AC電流 "+ message.substring(2));
        }else if(message.substring(0,2).equals("BD")){
            // 停雷検出 放電開始
                 Toast.makeText(this,"停電です!!",Toast.LENGTH_SHORT).show();
          mSWStatus.setText("停電検出しました。 放電しています。 "+ message.substring(2));
        }else if(message.substring(0,2).equals("BR")){
            // 停電回復 放電停止
                 Toast.makeText(this,"停電回復!!",Toast.LENGTH_SHORT).show();
          mSWStatus.setText("停電回復しました。 通常制御しています。 "+ message.substring(2));
        }else if(message.substring(0,2).equals("CD")){
            // 停電検出 放電開始
                 Toast.makeText(this,"バッテリ電圧低下!!",Toast.LENGTH_SHORT).show();
          mSWStatus.setText("バッテリの充電を開始しました。 "+ message.substring(2));
        }else if(message.substring(0,2).equals("CR")){
            // 停電回復 放電停止
                 Toast.makeText(this,"電圧回復!!",Toast.LENGTH_SHORT).show();
          mSWStatus.setText("充電終了しました。 通常制御しています。 "+ message.substring(2));
```

}

}

平成 24 年度文部科学省委託 「成長分野等における中核的専門人材養成の戦略的推進事業」 環境・エネルギー分野のスマートグリッドエンジニア育成の調査研究プロジェクト

#### ■実施委員会

◎ 鳥居	高之	船橋情報ビジネス専門学校 校長
古賀	稔邦	日本電子専門学校 校長
岡田	靖志	浜松情報専門学校 教務課長
竹原	伸	近畿大学 工学部 知能機械工学科 教授
大串	卓矢	株式会社 スマートエナジー 代表取締役
池澤	寿弘	コスモライフ株式会社 代表取締役社長
原田	<b>殿</b> —	有限会社ワイズマン 代表取締役
鳥海	豊彦	株式会社コラボレート研究所(代表取締役
渡辺	登	株式会社アフレル エディケーション・プランナー/事業企画室 室長
飯塚	正成	一般社団法人全国専門学校情報教育協会 事務局長

# ■調査

◎ 鳥居	高之	船橋情報ビジネス専門学校 校長	
原田	<b>賢</b> 一	有限会社ワイズマン 代表取締役	
松川	恵美	株式会社グリッド&ファイナンス・アドバイザーズ	取締役副社長
吉岡	正勝	有限会社ザ・ライスマウンド マーケティングマネー	ージャー

# ■開発

◎ 鳥居	高之	船橋情報ビジネス専門学校 校長
原田	賢一	有限会社ワイズマン 代表取締役
松川	恵美	株式会社グリッド&ファイナンス・アドバイザーズ 取締役副社長
吉岡	正勝	有限会社ザ・ライスマウンド マーケティングマネージャー

## 平成 24 年度文部科学省委託

「成長分野等における中核的専門人材養成の戦略的推進事業」 環境・エネルギー分野のスマートグリッドエンジニア育成の調査研究プロジェクト

# スマートグリッドエンジニア育成教材

平成 25 年 3 月

代表校 学校法人三橋学園 船橋

学校法人三橋学園 船橋情報ビジネス専門学校 〒273-0005 千葉県船橋市本町 7-12-16

問合せ先 有限会社ザ・ライスマウンド 〒164-0003 東京都中野区東中野 1-57-8 辻沢ビル 3F 電話:03-5332-5080 FAX 03-5332-5083

●本書の内容を無断で転記、掲載することは禁じます。