

書いてる途中です

# ADK互換モジュールで遊ぶAndroid

第4回名古屋Android勉強会

2012/2/18 愛知工業大学 本山キャンパス

@magoroku15 最底辺活動家

# このコースの目的と内容

## 目的

AndroidアプリケーションデベロッパがAndroid Open Accessory を理解する。同時にHWを理解する事が広げる世界を体感する。

## 内容

1. 互換モジュールの組み立て
2. Androidとの接続と動作確認
3. 互換モジュールの仕組み
4. Android側の仕組み

# **PARTO**

## **事前準備**

# 最新リソース

- このドキュメントを含む、最新のリソースは以下で管理しています
  - <https://github.com/magoroku15/OpenAccessoryDemo>
    - App Source CodeAndroid アプリのソースコード
    - Doc ドキュメント類
    - Firmware マイコンのソース・バイナリ
- 紹介するMicrochip Technology Inc作成のアンドロイドアプリをAndroid Marketからインストールしておいてください
  - Microchipで検索
    - 基本的なアクセサリデモ3.x
    - 基本的なアクセサリデモ2.3.x以降

# レベルごとの準備

- レベル1 手ぶらコース
  - 特に準備は不要です
- レベル2 実機に接続して動かす
  - ADKに対応したAndroid実機とACアダプタを用意
- レベル3 マイコン実行イメージのビルドまで
  - MPLABXをインストールしたPCを用意
- レベル4 Androidアプリのビルドまで
  - Android SDKをインストールしたPCを用意

# ソースコード・ドキュメント

- Githubで管理

<https://github.com/magoroku15/OpenAccessoryDemo>

- 2つの方法

- A) Githubにアカウントを作つてfork

- Linux/Macで開発する人にお勧め
    - 改変してcommit & push

- B) ZIPアーカイブをダウンロード

- Windowsで開発する人向け

<https://github.com/magoroku15/OpenAccessoryDemo>

Gitに慣れていない人はここからZipをダウンロード

GitHubにアカウントのある人は、ここから。Forkも歓迎

magoroku15 / OpenAccessoryDemo

Code Network Pull Requests 0 Issues 0 Wiki 0 Stats & Graphs

ZIP SSH HTTP Git Read-Only git@github.com:magoroku15/OpenAccessoryDemo.git Read+Write access

branch: master branch

Latest commit: master branch

構成変更

magoroku15 authored about 21 hours ago commit 145fc5fb6e

OpenAccessoryDemo /

name	age	message	history
App Source Code	December 19, 2011	first commit [magoroku15]	
Doc	about 21 hours ago	構成変更 [magoroku15]	
Firmware	December 20, 2011	NA [magoroku15]	

hideo@amdx6: ~ magoroku15/OpenA...

# OpenAccessoryDemoの内容

- App Source Code
  - Androidアプリのソース
- Doc
  - ドキュメント
- Firmware
  - PIC24Fのソース
  - コンパイルは専用のIDE MPLABXで行う

# Android SDKのインストール

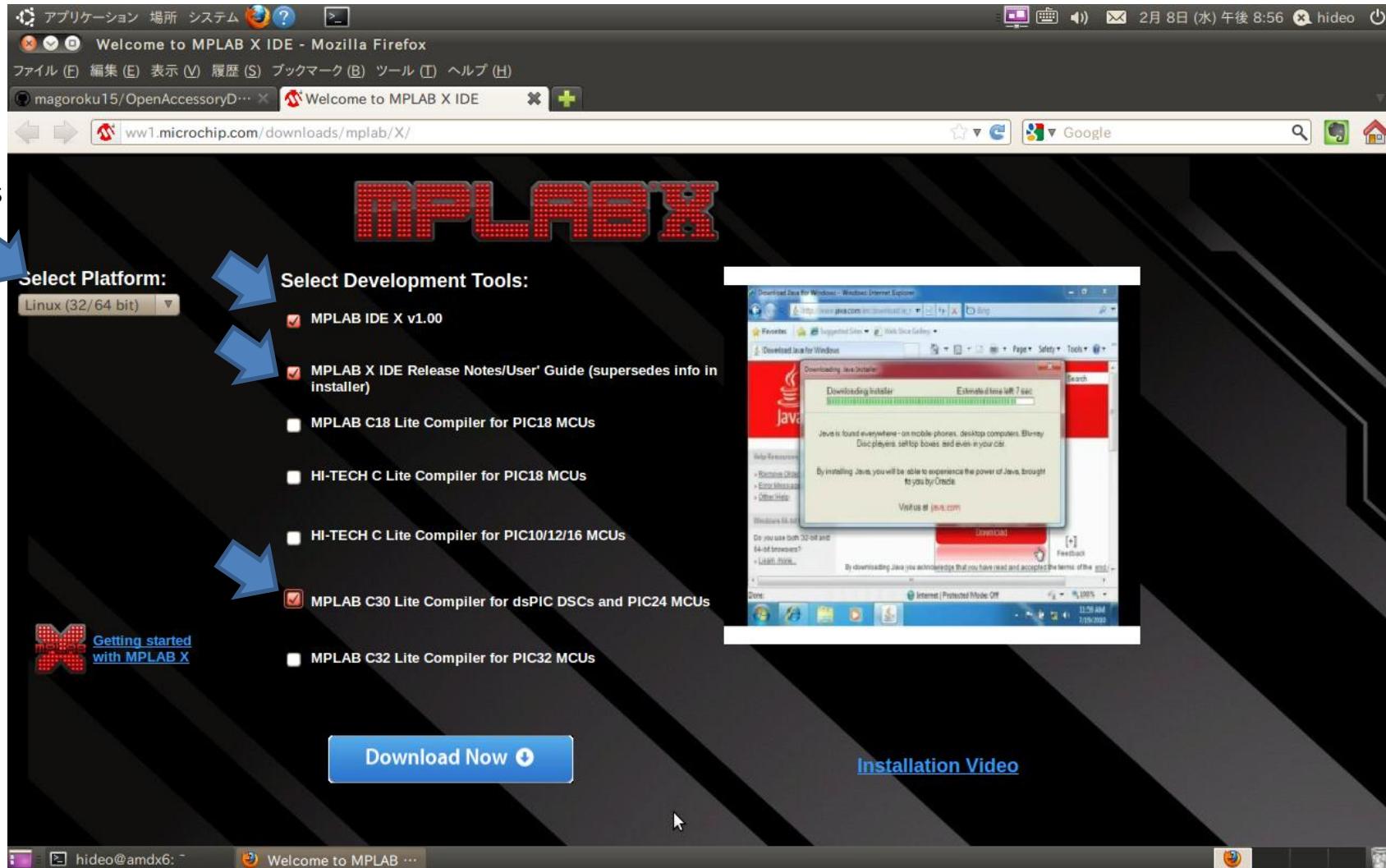
- ・ ハンズオンでAndroidのサンプルプログラムの解説を行い、その中でAppの作成に軽く触れます
- ・ 動作確認はマーケットからダウンロードしたアプリで行います
- ・ サンプルソースのバージョンに合わせて各自で事前にインストールを済ませておいてください

# MPLABXのインストール

- PIC用のIDE
  - 従来のMPLABはWindows専用だった
  - MPLABXはNetBeansベースでマルチプラットフォーム
  - PIC24シリーズ向けはコンパイラも無償/最適化に制限
- <http://ww1.microchip.com/downloads/mplab/X/>から
  1. Platformを選択
  2. 以下をチェックして[Download Now]
    - MPLAB IDE X v1.00
    - **MPLAB X IDE Release Notes/User' Guide (supersedes info in installer)**
    - **MPLAB C30 Lite Compiler for dsPIC DSCs and PIC24 MCUs**
  3. [Download Now]

# MPLABXのインストール

Mac  
Linux  
Windows



# **PART1**

## **互換モジュールの組み立て**

# 配布部品

積層セラミックコンデンサー10 $\mu$ F25V

積層セラミックコンデンサー1 $\mu$ F25V

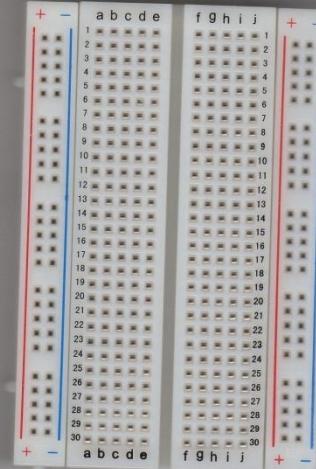
PIC24FJ64GB002

赤色LED 3mm

カーボン抵抗 1/4W10k $\Omega$

ブレッドボード・ジャンパーウイヤ

三端子レギュレーター



ブレッドボード EIC-801

USBケーブル(A-microB)  
Lピンヘッダ

可変抵抗

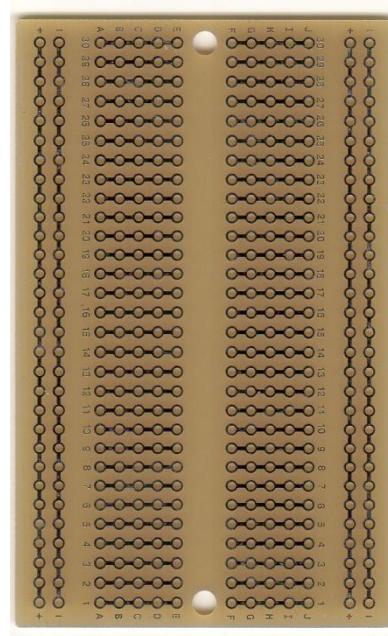
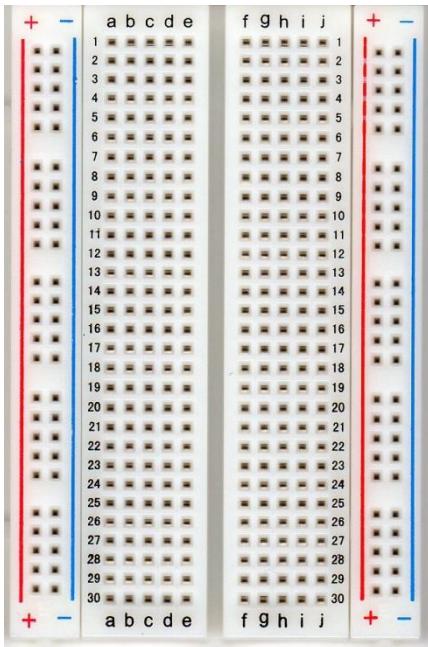
# BOM/購入元

poorman's ADK 30台分の発注部品

購入先	通販型番	発注量	購入単位		1台単位		
			個数	価格	単価	個数	
秋月	C-05223	30	1	130	130	1	130 USBケーブル(A-microB)
秋月	P-02315	30	1	250	250	1	250 ブレッドボード・ジャンパーウイヤ
秋月	P-00315	30	1	250	250	1	250 ブレッドボード EIC-801
秋月	I-03421	15	2	100	50	1	50 三端子レギュレーター[3. 3V] XC6202P332TB
秋月	P-05105	60	1	15	15	2	30 積層セラミックコンデンサー1μ F50V
秋月	P-05103	30	1	30	30	1	30 積層セラミックコンデンサー10μ F25V
秋月	I-00562	1	100	350	3.5	2	7 赤色LED 3mm
秋月	R-25103	2	100	100	1	4	4 カーボン抵抗 1／4W10kΩ
秋月	C-01627	10	5	20	4	1	4 Lピンヘッダ1x20
秋月	P-02470	4	10	200	20	1	20 半固定抵抗10kΩ
秋月	P-04089	1	100	500	5	1	5 チャック袋
秋月	送料2回分	1	30	1000	33	2	67 送料500円x2回
digikey	703-7602	1	30	9821	327	1	327 PIC24FJ64GB002

合計 1174

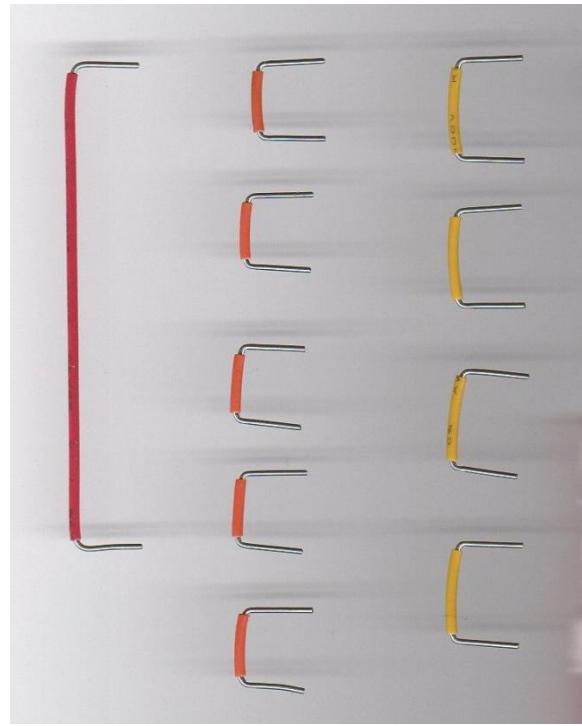
# ブレッドボード



内部の配線

# ジャンパ

- ・ ブレッドボードに刺して回路を構成する線



# マイコン PIC24FJ64GB002

- Microchip社の16bitマイコン max 32MHz
  - 64Kbyte Program Memory (Flash)、64Kbyte RAM
  - I<sup>2</sup>C, IrDA, SPI, UART/USART, USB OTG
- プログラムは書き込み済

The diagram shows the pin configuration for the PIC24FJ64GB002 microcontroller. A blue arrow points from the text "ピン配置" (Pin Configuration) to the table below.

		PIC24FJ64GB002	
PGED3/AN0/C3INC/VREF+/ASDA1 <sup>(2)</sup> /RP5/PMD7/CTED1/V/BUSVLD/CMPST1/CN2/RA0	1	MCLR	VDD
PGECL1/C3IND/VREF-/ASCL1 <sup>(2)</sup> /RP6/PMD6/CTED2/SESSVLD/CMPST2/CN3/RA1	2		VSS
PGED1/AN2/C2INB/DPH/RP0/PMD0/CN4/RB0	3		AN9/C3INA/BUSCHG/RP15/BUSST/CN11/RB15
PGECL1/AN3/C2INA/DMH/RP1/PMD1/CN5/RB1	4		AN10/C3INB/CVREFV/CPCON/BUSON/RP14/CN12/RB14
AN4/C1INB/DPLN/SDA2/RP2/PMD2/CN6/RB2	5		AN11/C1INC/RP13/PMRD/REFO/SESEND/CN13/RB13
AN5/C1INA/DMLN/RTCC/SCL2/RP3/PMWR/CN7/RB3	6		VUSB
VSS	7		PGECL2/D-/MIO/RP11/CN15/RB11
OSCI/CLKI/C1IND/PMCS1/CN30/RA2	8		PGED2/D+/PIO/RP10/CN16/RB10
OSCO/CLKO/PMA0/CN29/RA3	9		VCAP/DDCORE
SOSCI/C2IND/RP4/PMBE/CN1/RB4	10		DISVREG
SOSCO/SCLKI/T1CK/C2INC/PMA1/CN0/RA4	11		TDO/SDA1/RP9/PMD3/RCV/CN21/RB9
VDD	12		TCK/USBOEN/SCL1/RP8/PMD4/CN22/RB8
TMS/USBID/CN27/RB5	13		TDI/RP7/PMD5/INT0/CN23/RB7
	14		VBUS

ピン配置

# 抵抗

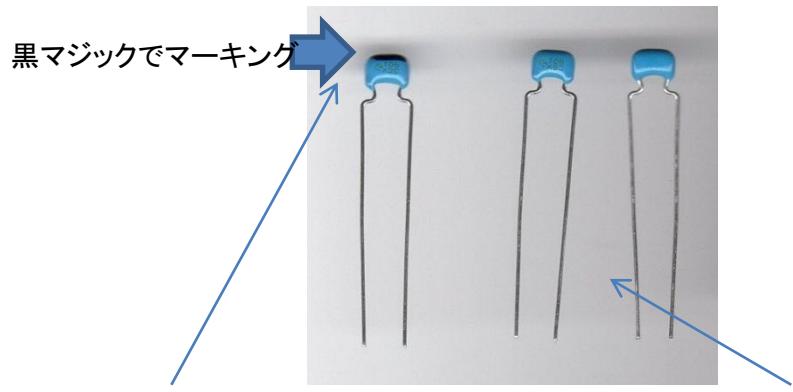
- 電流の流れを抑止する
  - 抵抗の値が小さいと導体
  - 抵抗の値が大きいと絶縁体に近づく
- 極性なし



回路図

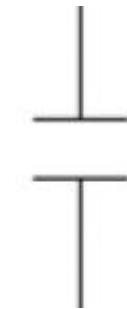
# コンデンサ

- Capacitor(キャパシタ)とも言う
- 微量の電気を蓄える
  - 直流は流れない
  - 交流は流れやすい
- 極性の有無に注意
  - 今回の積層セラミックコンデンサは極性なし



積層セラミックコンデンサー  $10\mu\text{F} 25\text{V}$

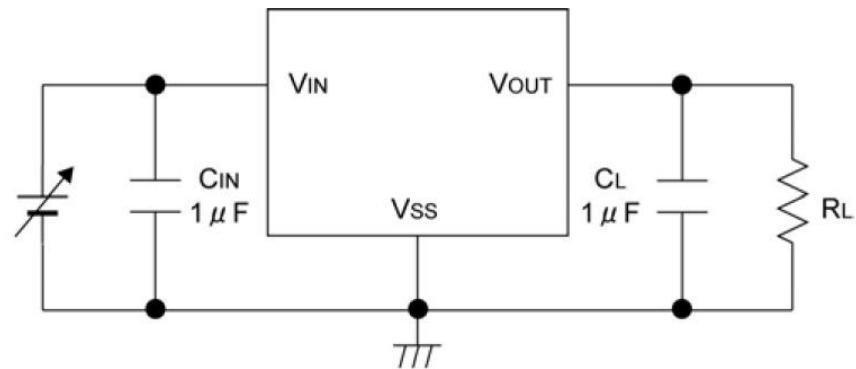
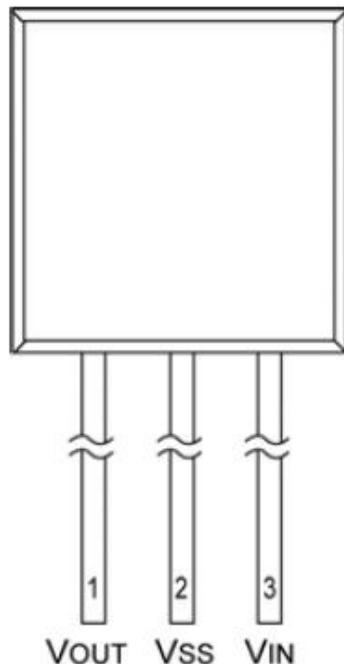
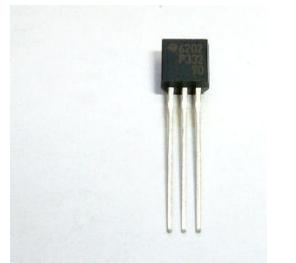
積層セラミックコンデンサー  $1\mu\text{F} 25\text{V}$



回路図

# 3端子レギュレータ

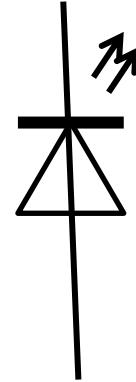
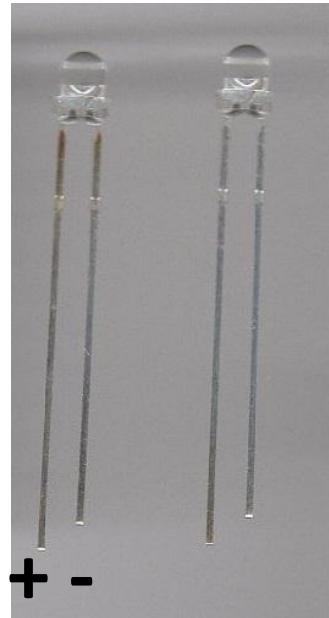
- 電圧を一定に保つ機能を備えたIC
- 5V(充電用ACアダプタ)から3.3Vを生成



回路図(周辺込み)

# LED

- 発光ダイオード
  - LED(エルエイーディー: Light Emitting Diode)
  - 極性あり、一方向にしか電流を流さない

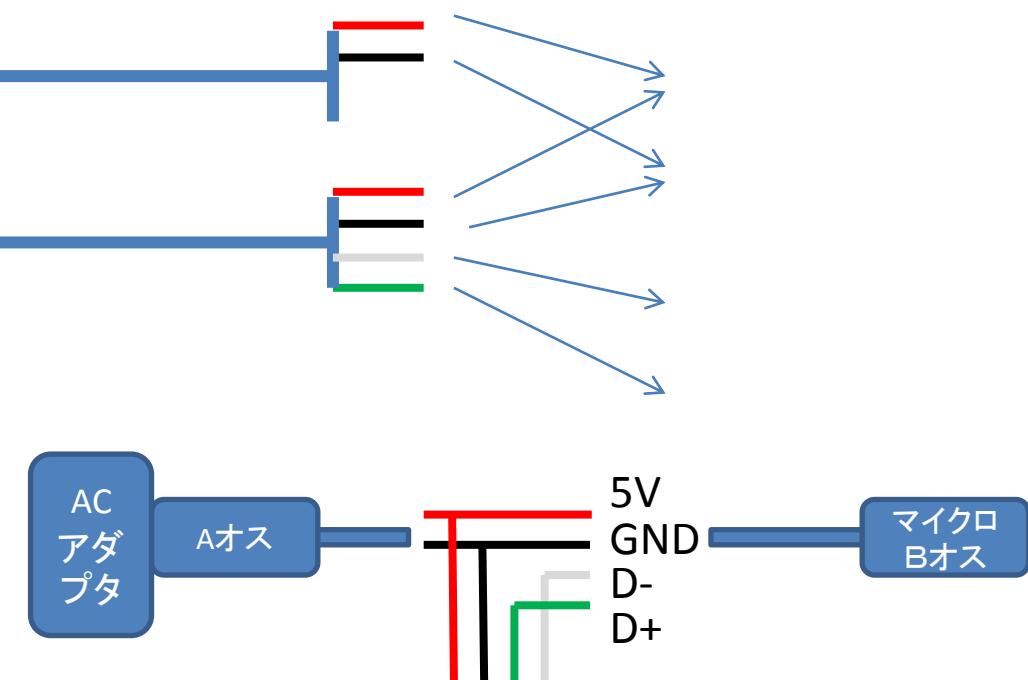


# USBケーブル

- AオスマイクロBオスを改造(改造済で配布)
  - ACアダプタから5Vを取り、マイクロBオスに回す
  - マイクロBのデータ線を取りだす



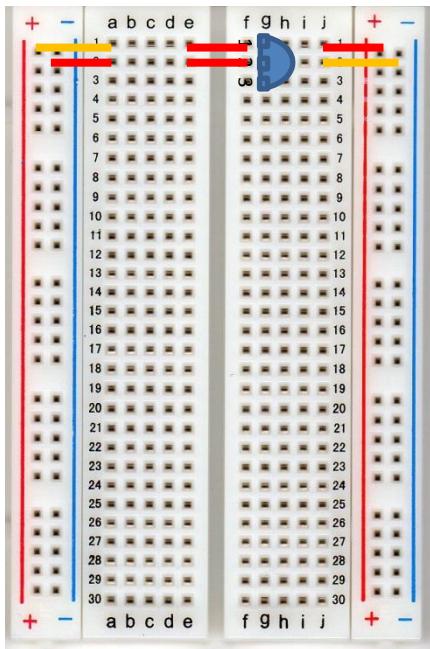
AオスーマイクロBオス



# 互換モジュールの組み立て

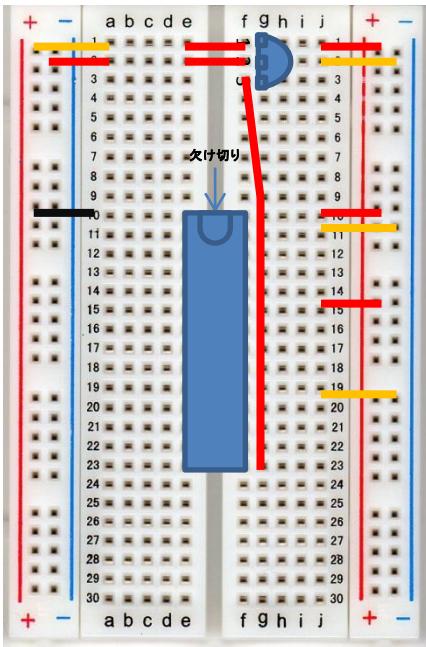
- 利用する資料
  - 回路図(別紙)
  - 実体配線図(別紙)
  - 組み立て手順(本資料)
- 配布部品

# Step1 電源



1. g1にレギュレータの1
  2. g2にレギュレータの2
  3. g3にレギュレータの3
  4. **一**を4か所
  5. **一**を2か所
  6. 積層セラミックコンデンサー  
1μF25V2個
    - h1-h2
    - h2-h3

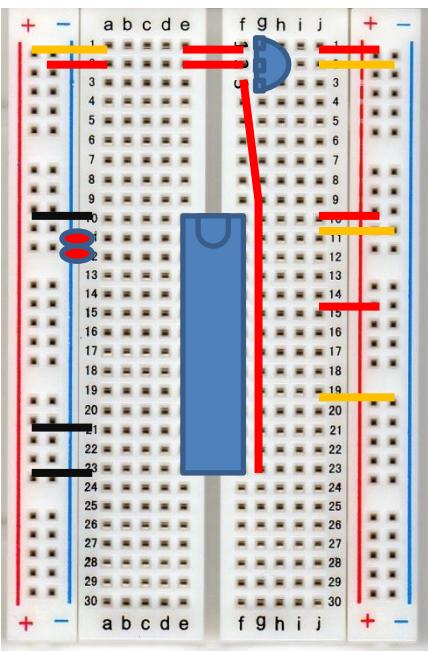
# Step2 マイコン



1. e10-e23-f10-f23'にマイコン
2. ーをf3-g23へ
3. ーをj10-ーへ
4. ーをj11-+へ
5. ーをj15-ーへ
6. ーをj19-+へ
7. 積層セラミックコンデンサー $10\mu\text{F}$ 25をj18-へ
8. 抵抗 $10\text{k}\Omega$ をa10-+へ

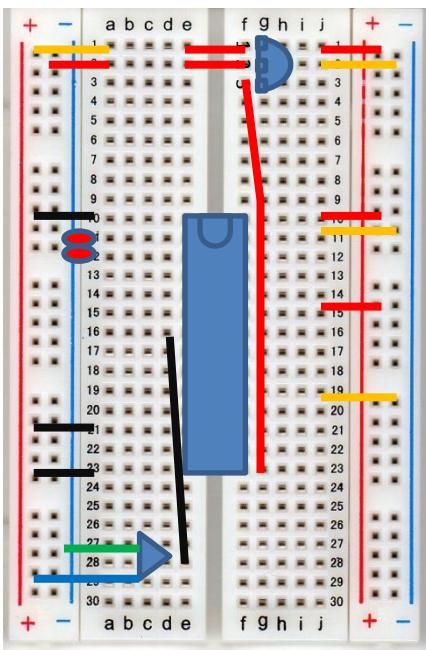
# Step2 LED/SW

- LED、足の長い方をa11,短い方を—へ
- LED、足の長い方をa12,短い方を—へ
- 抵抗 $10k\Omega$ 、a21-+へ
- 抵抗 $10k\Omega$ 、a23-+へ



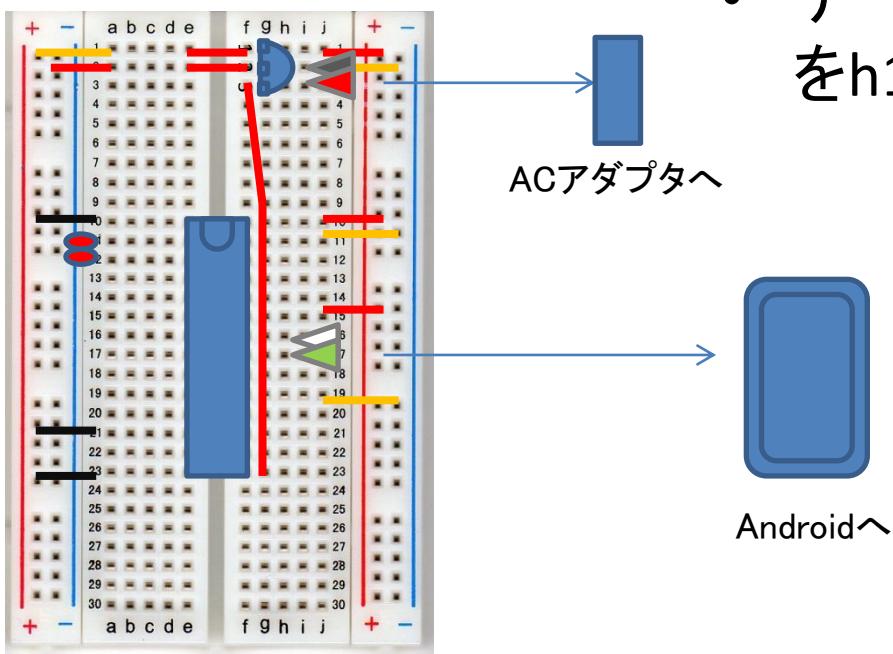
# Step3 可変抵抗

- 組み付け順注意
- d16-e28に抵抗を
- —をc27-—に
- —をc29-+に
- 可変抵抗をc27-c29-e28へ

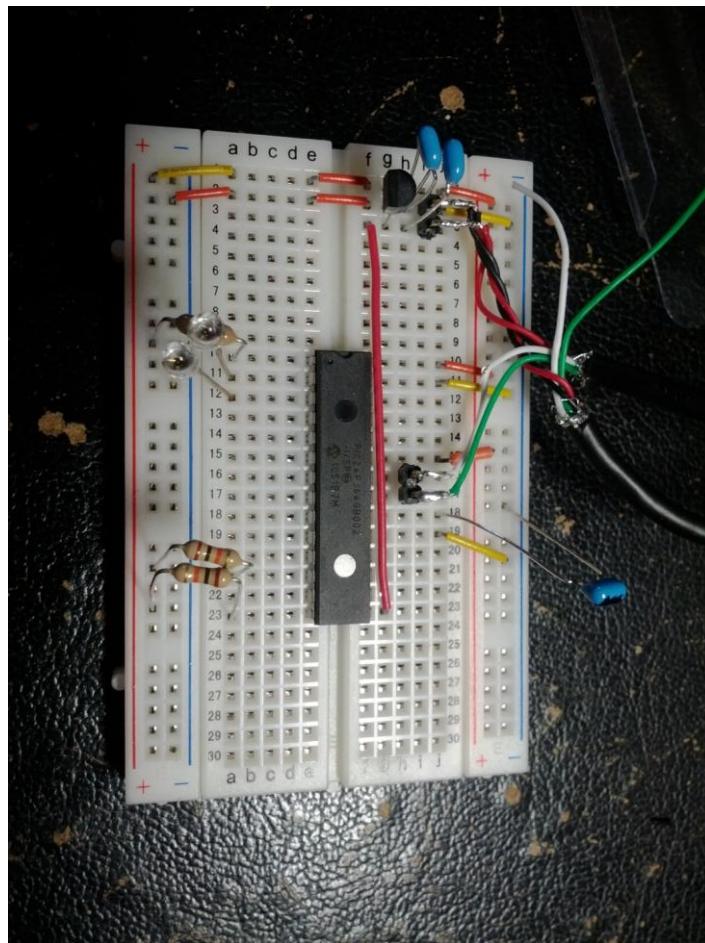
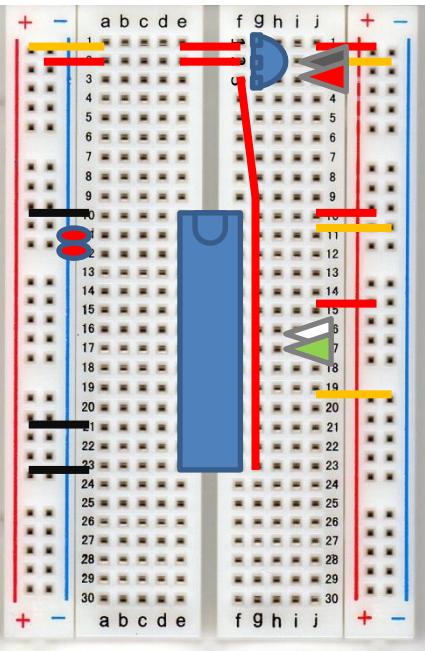


# Step2 USBケーブル

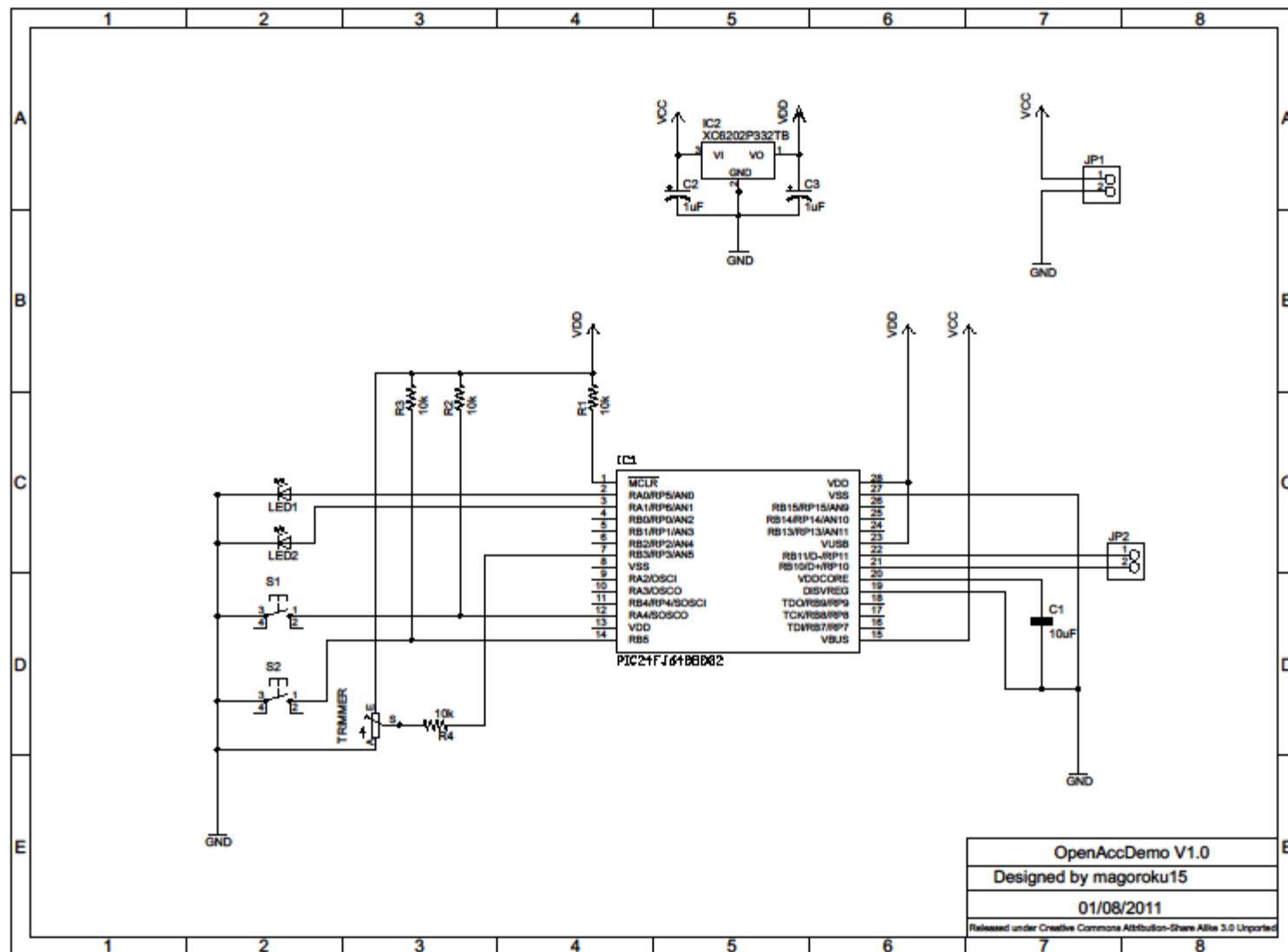
- 電源、黒をj2,赤をj3
- データ 白(D-)をh16, 緑(D+)をh17



# 完成写真



# 回路図



# **PART2**

# **ANDROIDとの接続と動作確認**

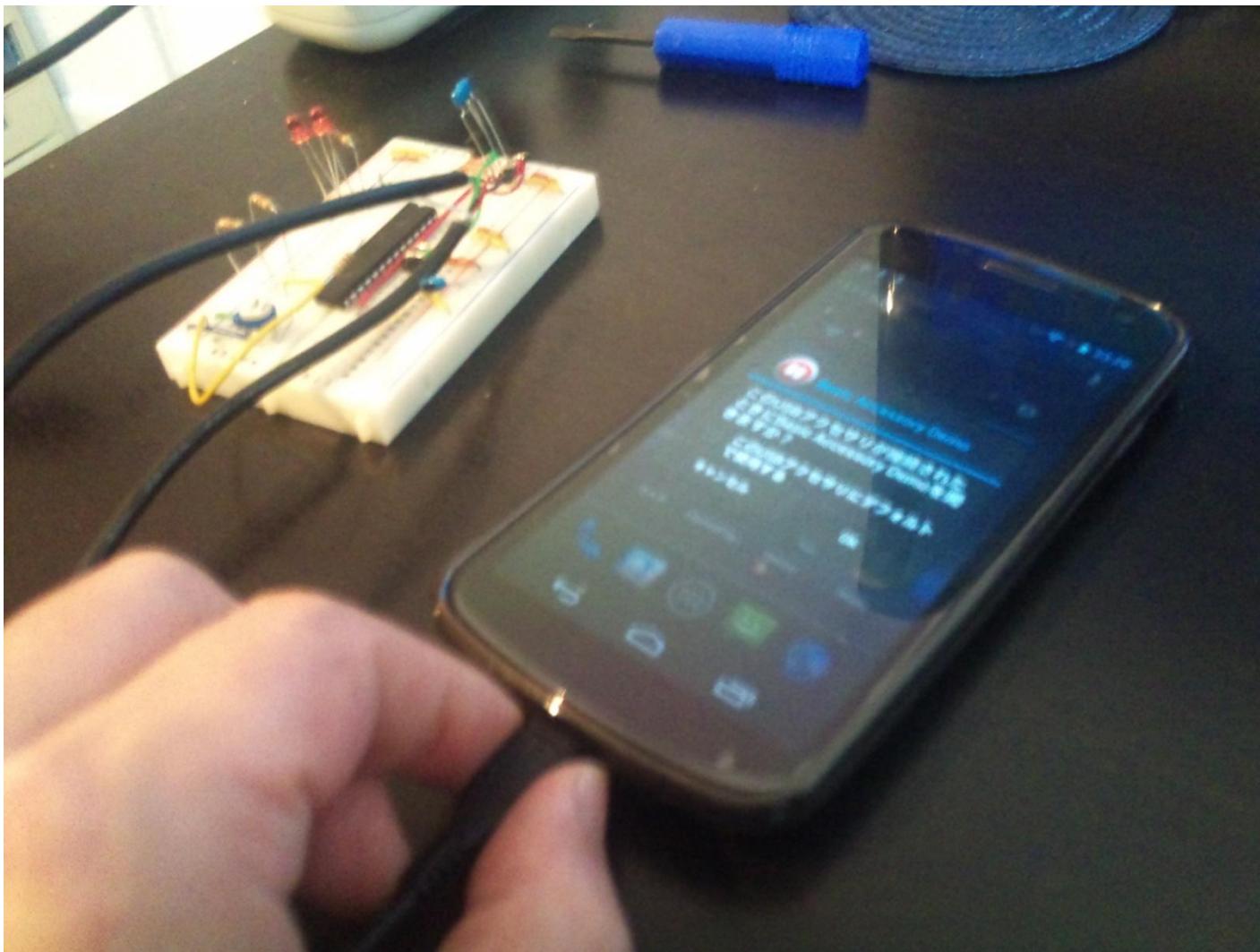
# Step1 アプリのインストール

- Android Marketからインストール
  - Microchipで検索
    - 基本的なアクセサリデモ3.x
    - 基本的なアクセサリデモ2.3.x以降

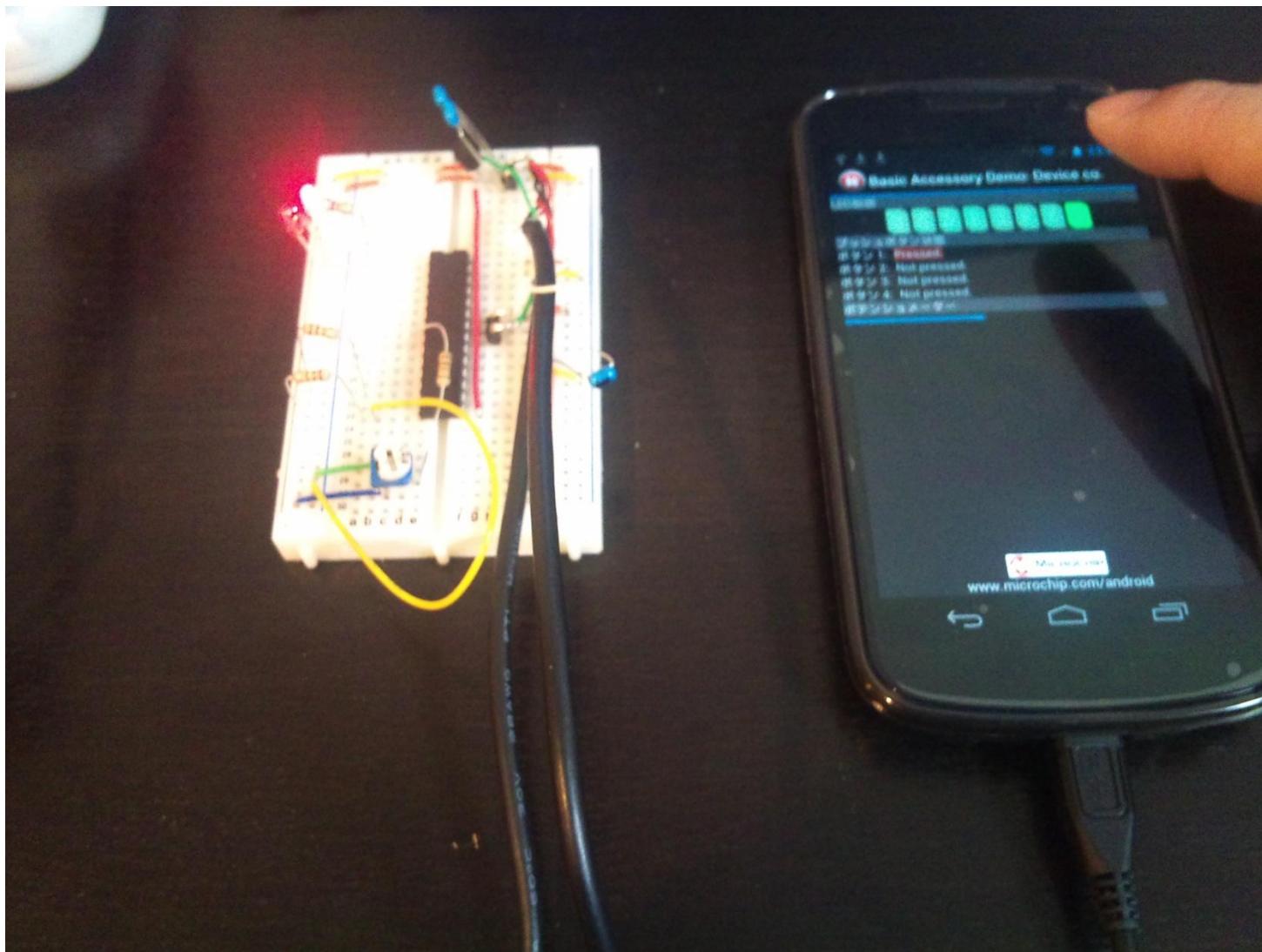
# Step2 ACアダプタに接続



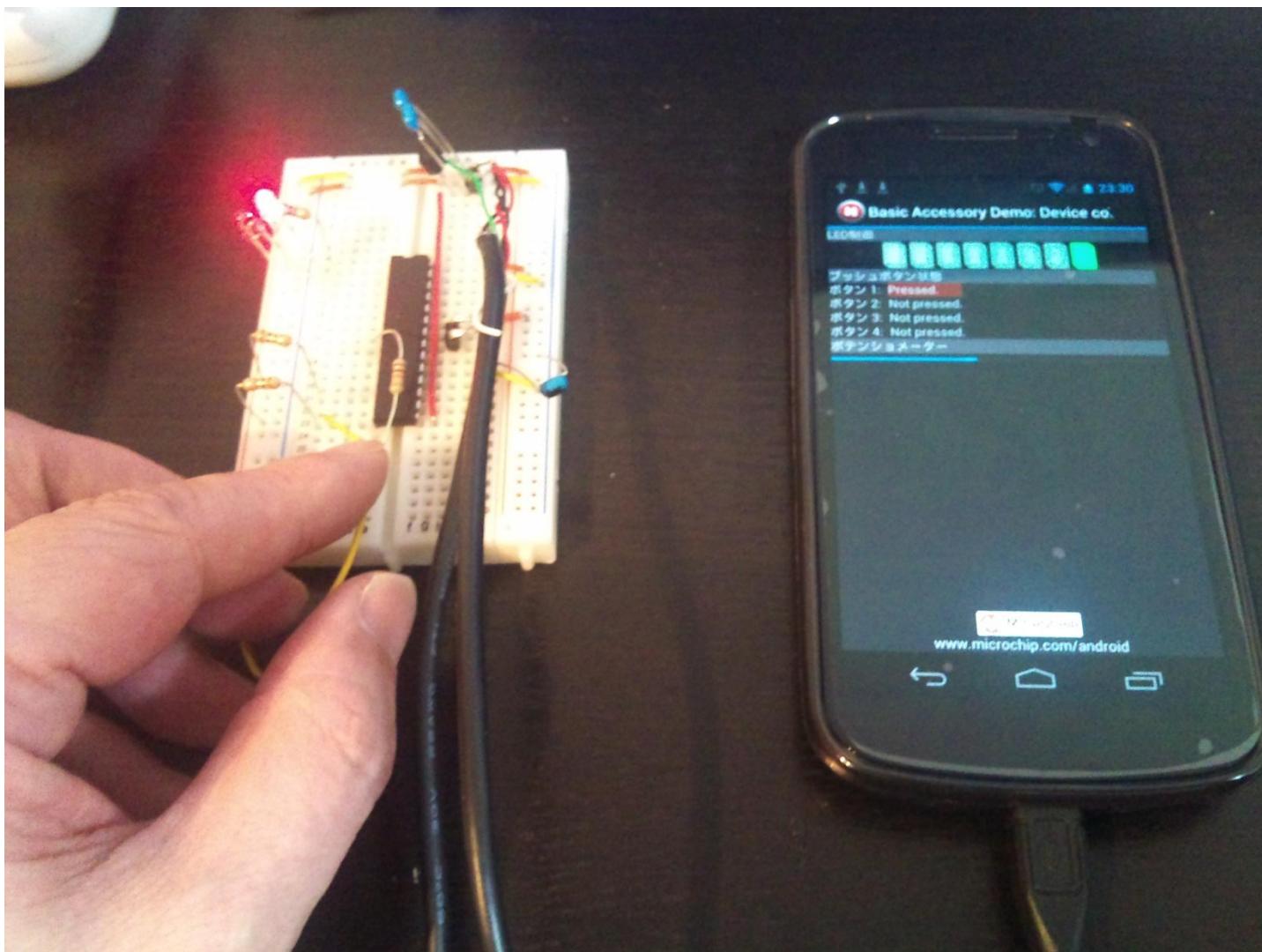
# Step3 Androidに接続



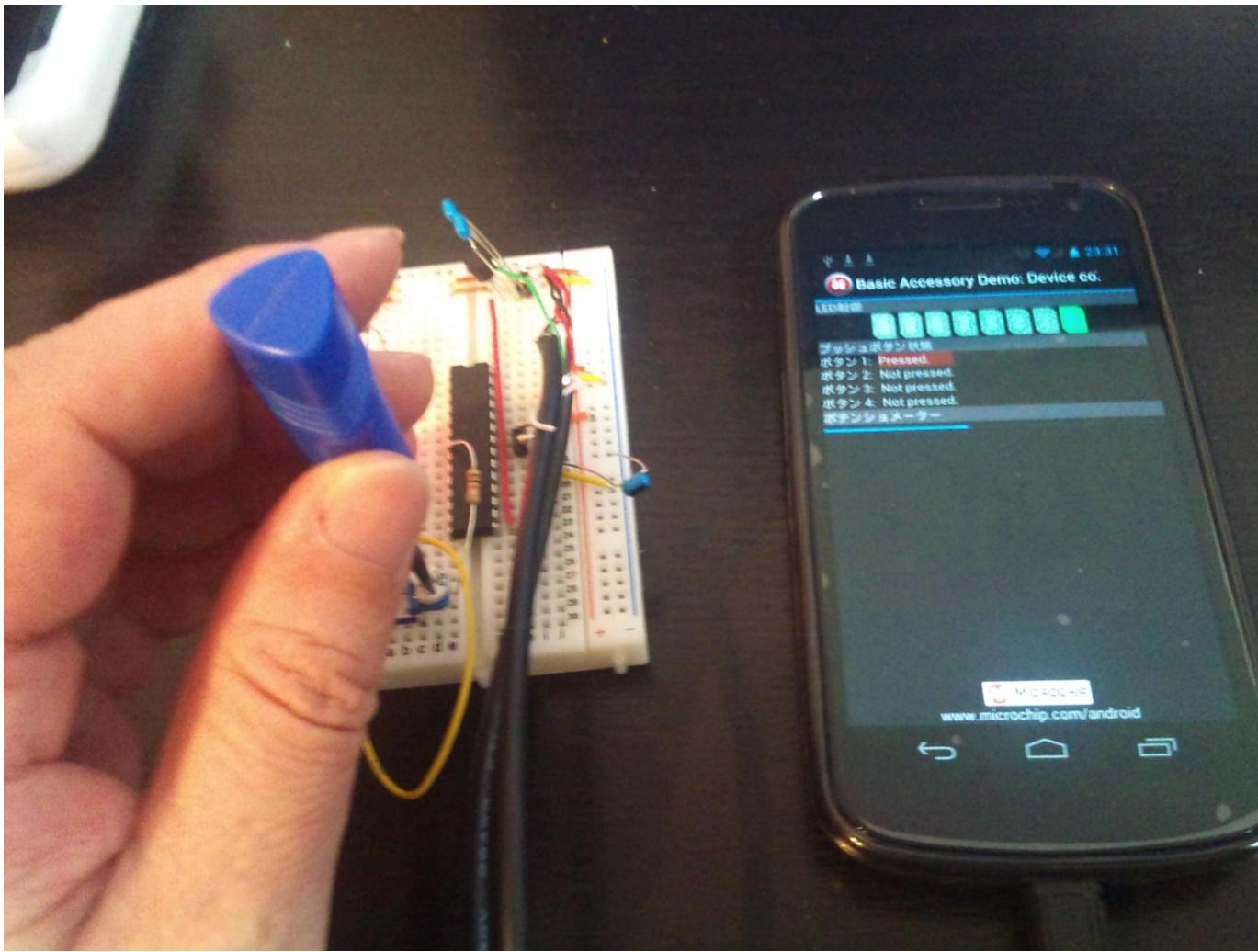
# LEDを点灯



# スイッチを操作



# ポテンションメータを操作



# **PART3**

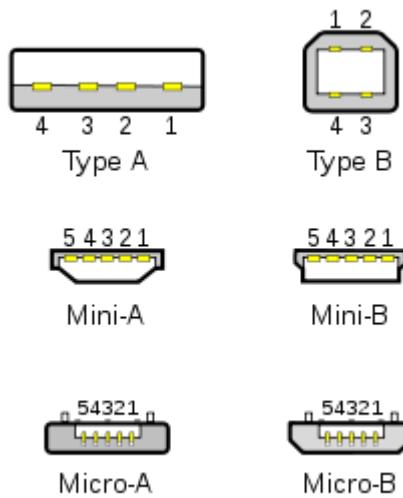
# **USBとOPEN ACCESSORY**

# USB ホスト・デバイス

- 基本は高速のシリアル通信
- 4線式
  - 通信はディファレンシャル D+,D-
  - 納電機能を持つ 5V, GND
- ホスト
  - 一般的にPC側
  - マスタとして動作
  - 複数のデバイスを収容・管理
  - 複雑なソフトウェアスタック
- デバイス
  - 一般的に装置側、マウス、KBD、プリンタ、USBメモリ
  - スレーブとして動作
  - 単純な機能

# USB コネクタ

- 4線式
  - ホスト側 1: + 2: D- 3: D+ 4: -
  - ディバイス側 Type A
  - Type B
- 5線式
  - IDでホスト、ディバイスを判別 1: + 2: D- 3: D+ 4: ID 5: -



Wikipediaより

# Androidでの利用例

- ホスト
  - システム側での対応は限定的、マウス対応(ICS)など
  - アプリ側でUSB manager連携してドライバ相当の動作をアプリとして記述・配布
- ディバイス
  - Android側がディバイス
  - ADB, Mass Storage
- Open Accessory
  - PCがホスト、Android側がディバイス
  - システムのみで利用していたエンドポイントをアプリに開放

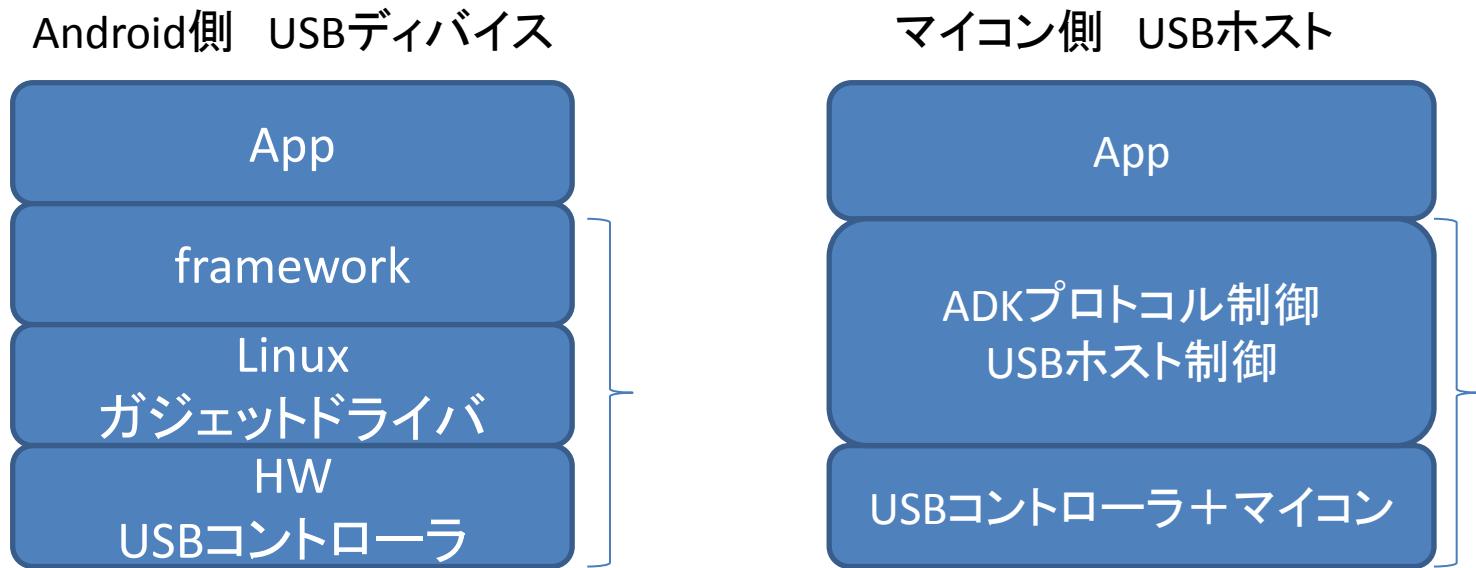
# ADBの仕組み

- ドライバレイヤ
  - ADB用のVid, Pidを登録してエンドポイントを作成
- アプリレイヤ
  - システム起動時にadbdコマンドを起動
- PCに接続すると
  - Vid, PidがXXXを示し
  - adb用のEndpointがみえるのでこれをadbコマンドが握る
  - PC adbコマンド $\leftrightarrow$ PC バルク転送 $\leftrightarrow$ Android バルク転送 $\leftrightarrow$  /dev/adb  $\leftrightarrow$  adbd
- adb (クライアント) $\leftrightarrow$ adbd(サーバ)の関係

# Open Accessoryの仕組み

# Open Accessory modeとは何か

- AdbなどのUSBガジェットの仕組みを流用
- 専用エンドポイントを提供
- フレームワークがアプリを起動し、エンドポイントを接続



# **PART4**

## **マイコン側の仕組み**

# PICマイコン

- マイコンとしては老舗
- ライバルはAtmel AVR
- PIC16シリーズ
- PIC24シリーズ
- PIC32シリーズ
  - MIPS社からライセンスしたMIPS M4K

# 開発環境

- MPLAB.X
  - 統合開発環境
  - 最近1.0に
  - NetBeansベース
    - MAC, Windows, Linuxの各プラットフォームで動作
- プログラマ
  - マイコン上のFlashにプログラムを転送
  - MPLAB.Xから書き込み
  - Pickit3



Pickit3 3900円  
秋月 通販コード M-03608

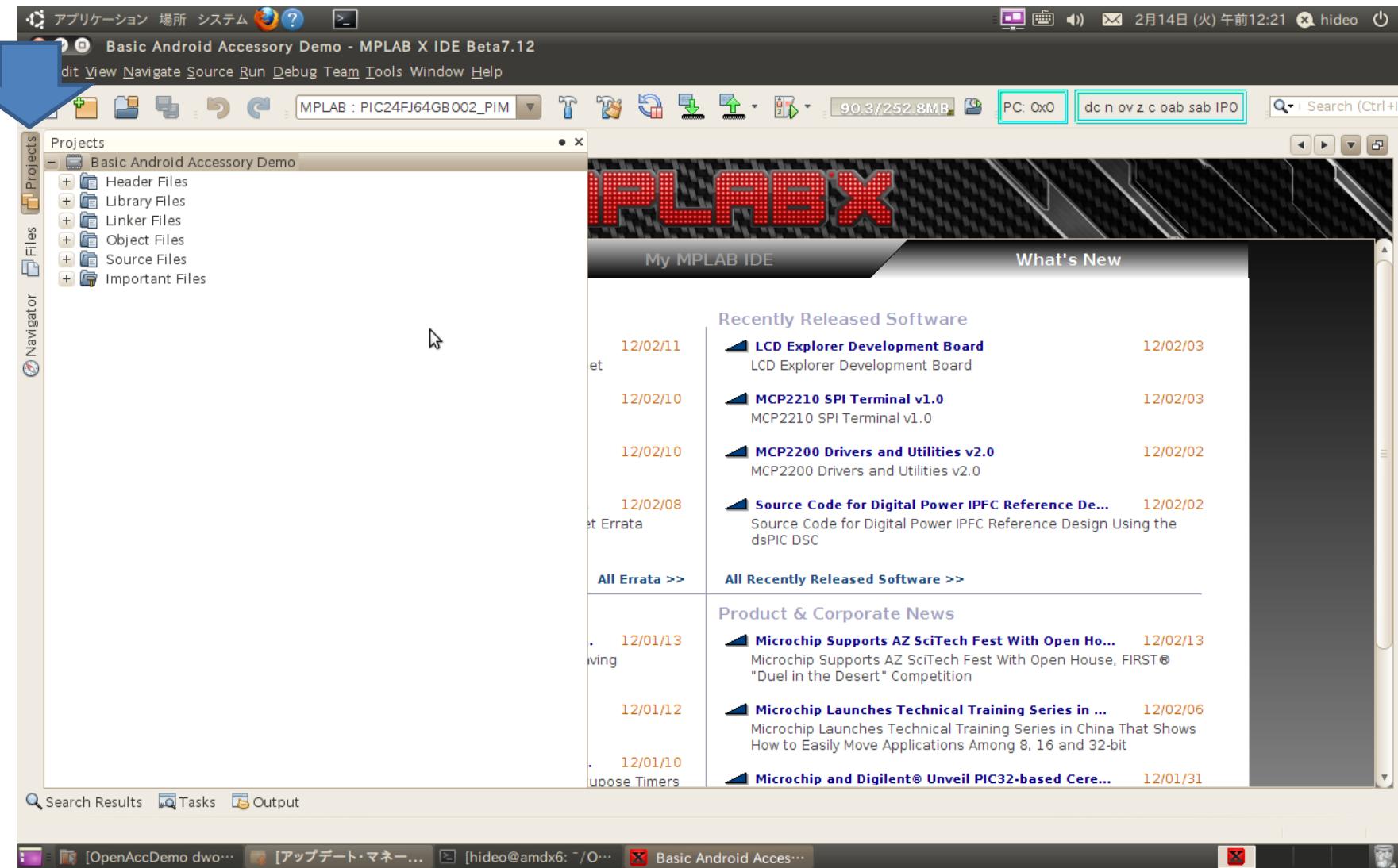
# MPLABXの使い方

- ・プロジェクトの手順
- ・ビルドの手順
- ・プログラムの手順

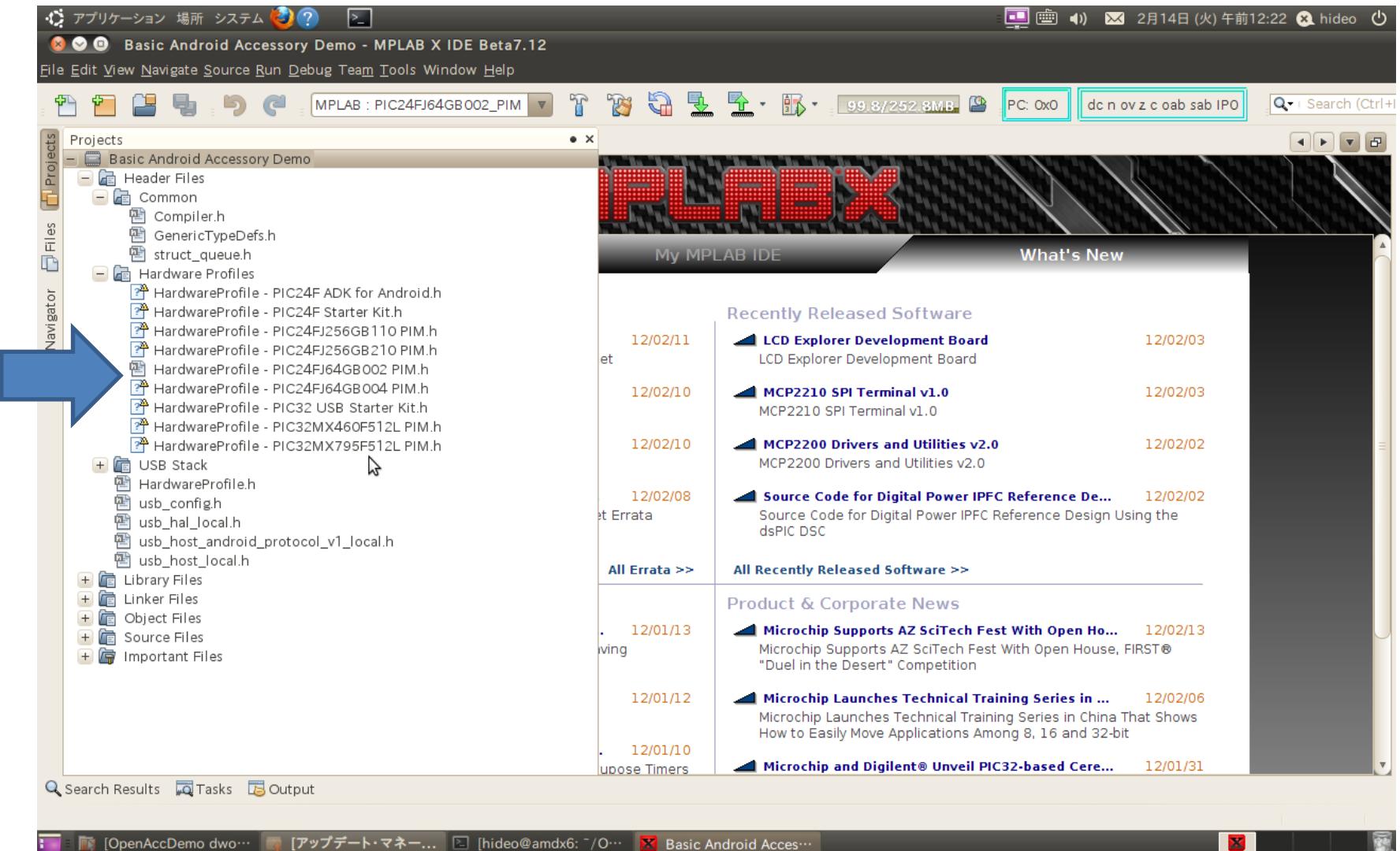
# プロジェクトのOpen

- 左上メニューからFile -> Open Project....
- プロジェクトを選択
  - OpenAccessoryDemo/Firmware/MPXLABを選択

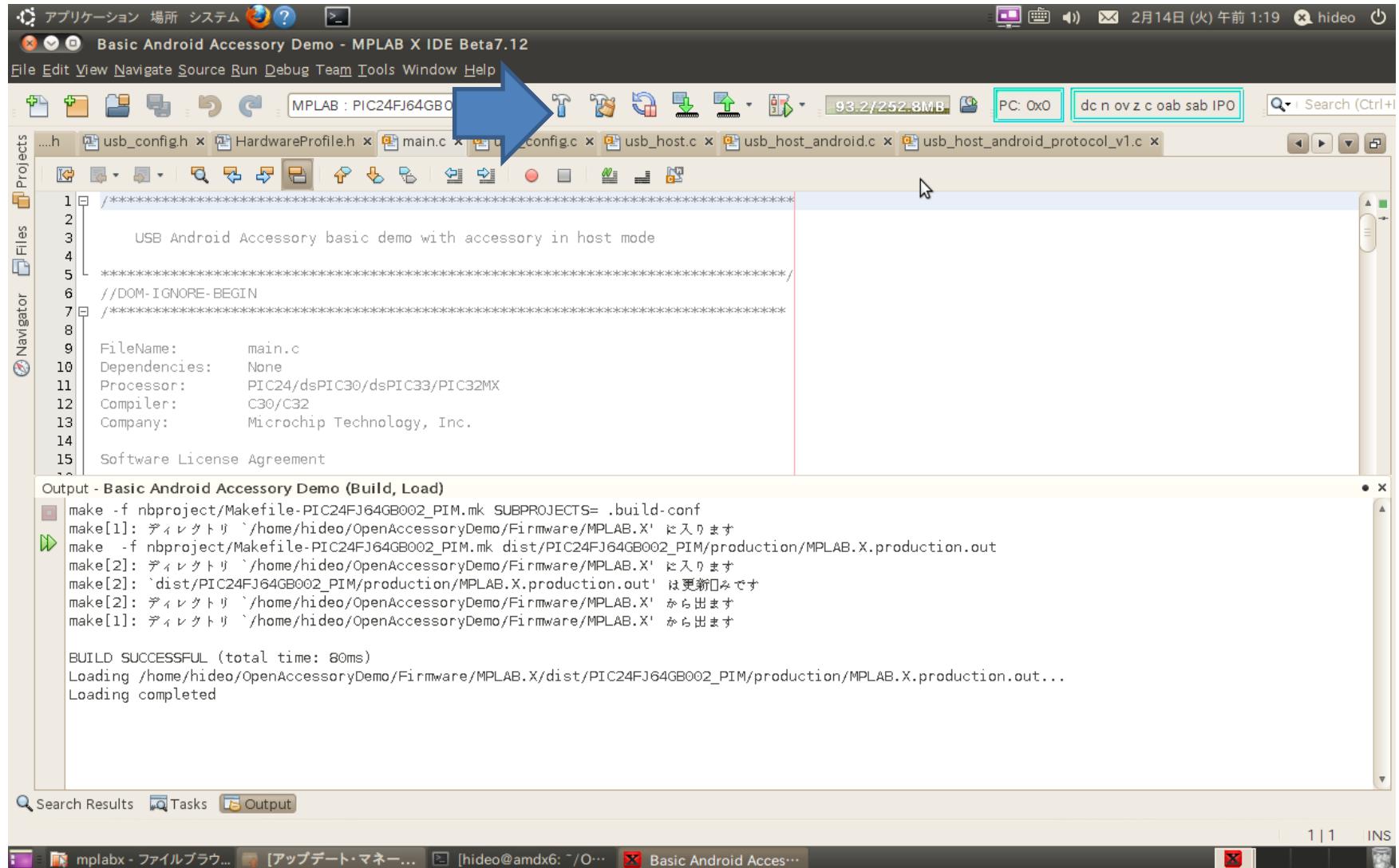
# ファイルの閲覧



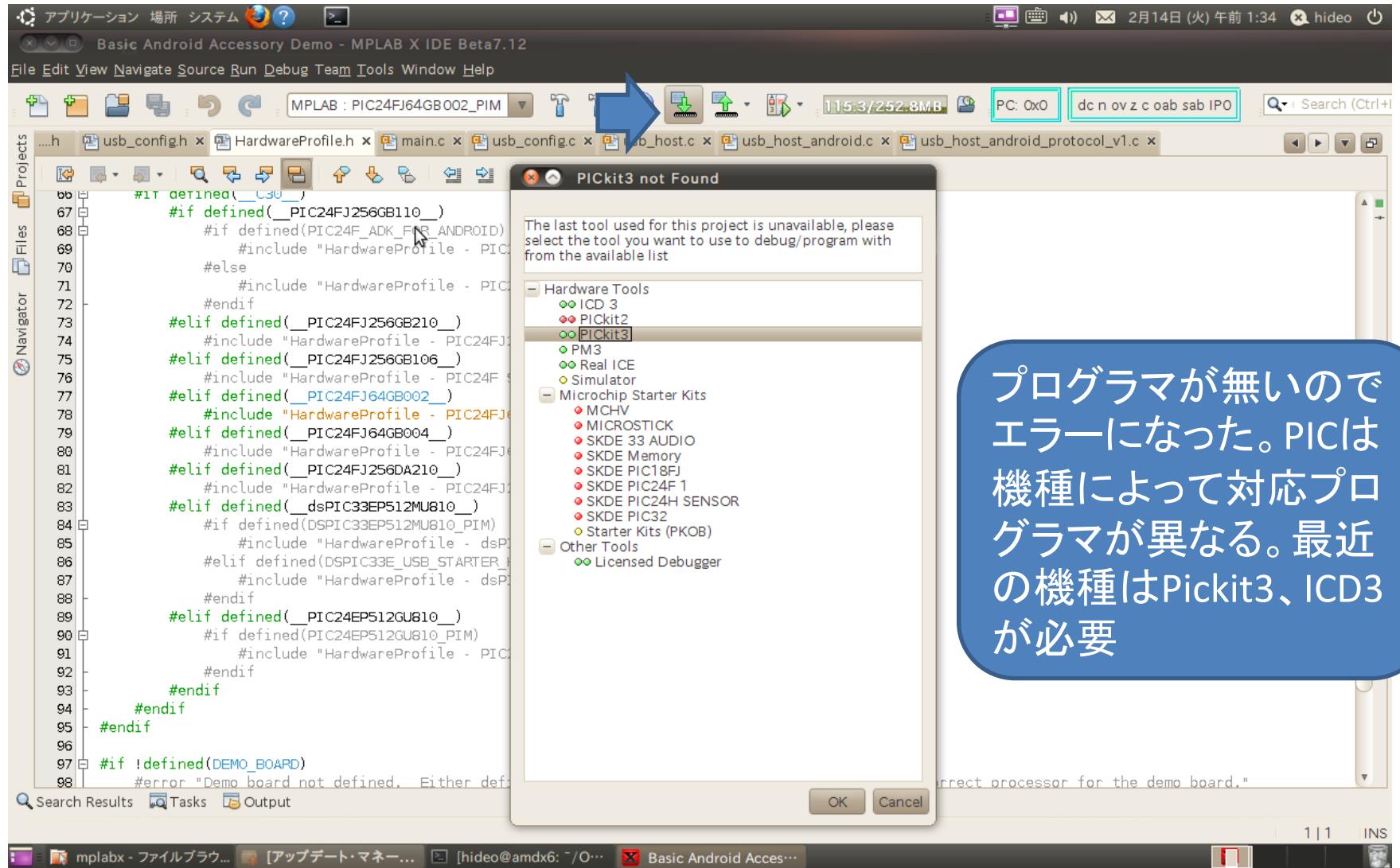
# ファイルを選択



# ビルド



# プログラム

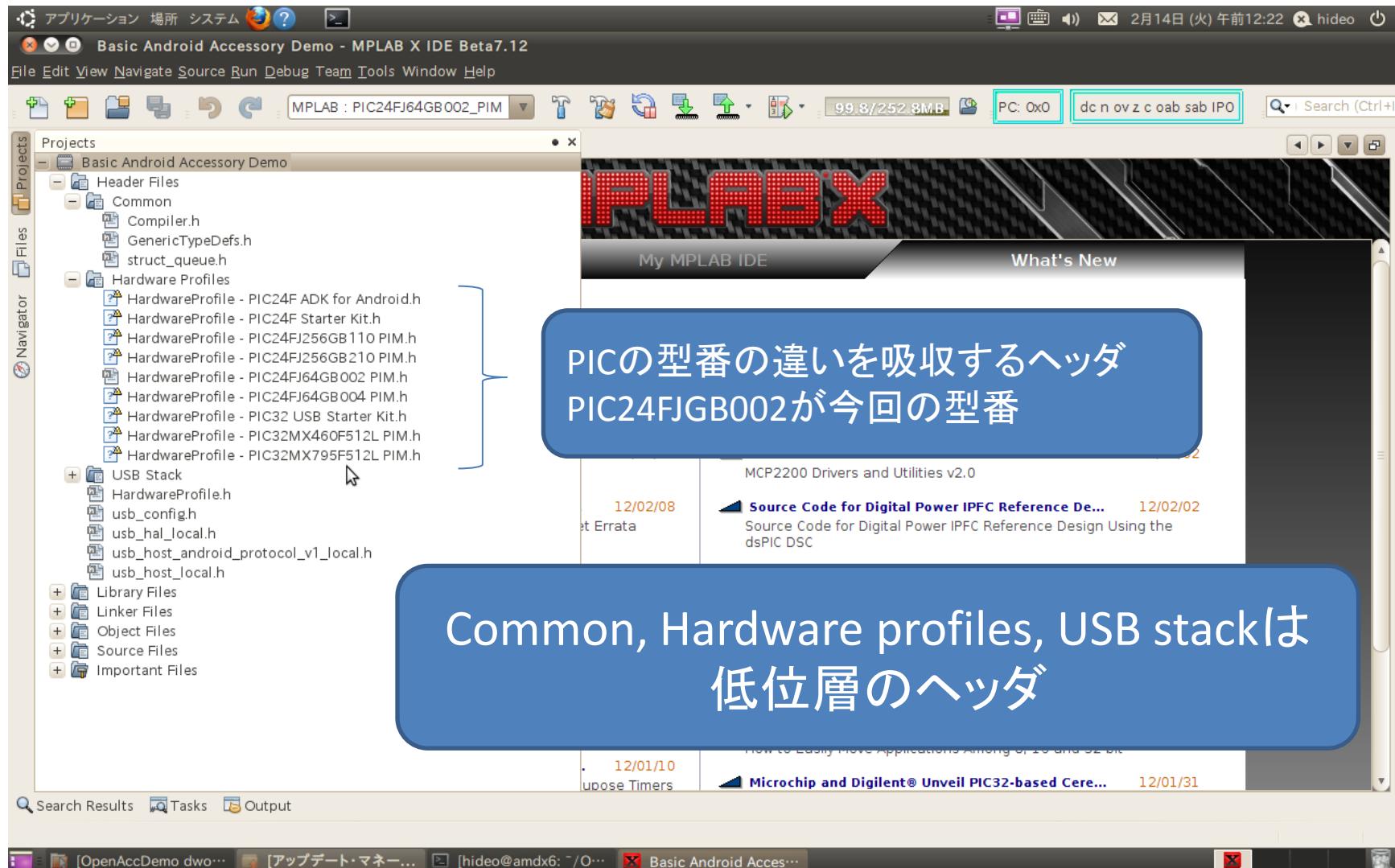


# Firmwareのウォークスルー

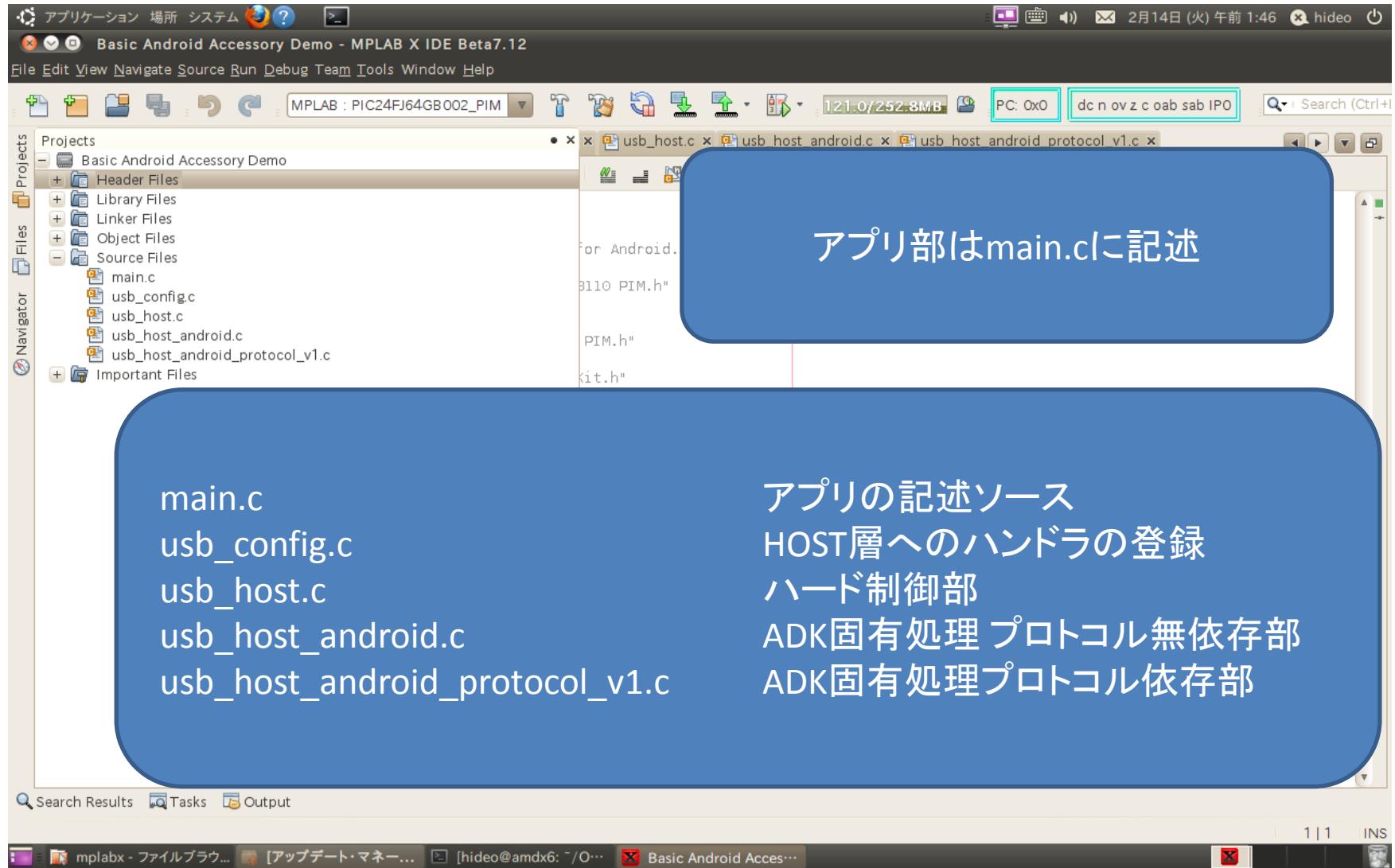
- Firmwareには、USBの低位層からアプリ層までのプログラムがソースで配置
- アプリ層のファイルは



# 低位層のヘッダ



# ソースファイル



# GPIOポートの出力設定

The screenshot shows the MPLAB X IDE interface with the following details:

- Title Bar:** アプリケーション 場所 システム ? 2月14日(火)午前12:22 hideo
- Project Name:** Basic Android Accessory Demo - MPLAB X IDE Beta7.12
- File List:** Start Page, usb.h, HardwareProfile - PIC24FJ64GB002 PIM.h
- Toolbar:** Standard file operations (New, Open, Save, Print, etc.) and build options.
- Status Bar:** 86.0/252.8MB, PC: 0x0, dc nov zc oab sab IPO, Search (Ctrl+I)
- Code Editor:** Displays the following C code for the PIC24FJ64GB002 PIM.h file:

```
#define EXPLORER_16
#define PIC24FJ64GB002_PIM
#define CLOCK_FREQ 32000000

#define DEMO_BOARD_NAME_STRING "PIC24FJ64GB002 PIM"

/** LED ****
#ifndef InitAllLEDs()    LATA &= 0xFD7F; TRISA &= 0xFD7F; LATB &= 0xFFFF3; TRISB &= 0xFFFF3;
#define InitAllLEDs()    LATA &= 0xFFFFC; TRISA &= 0xFFFFC; LATB &= 0xFFFF; TRISB &= 0xFFFF;

#define mLED_1           LATAbits.LATA0 //      LATAbits.LATA7
#define mLED_2           LATAbits.LATA1 //      LATAbits.LATB3
#define mLED_3           LATBbits.LATB2
#define mLED_4           LATAbits.LATA9

#define LED0_On()         mLED_1 = 1;
#define LED1_On()         mLED_2 = 1;
#define LED2_On()
#define LED3_On()         mLED_1 = 1;
#define LED4_On()
#define LED5_On()
#define LED6_On()
#define LED7_On()

#define LED0_Off()        mLED_1 = 0;
#define LED1_Off()        mLED_2 = 0;
#define LED2_Off()
#define LED3_Off()
#define LED4_Off()
#define LED5_Off()
#define LED6_Off()
#define LED7_Off()
```

The code defines constants for the PIC24FJ64GB002 chip and its board. It includes a macro for initializing all LEDs and individual macros for each LED (mLED\_1 to mLED\_4). The LED functions (On/Off) are implemented using bit manipulation on the LATAbits and LATBbits registers.

# GPIOの入力設定

アプリケーション 場所 システム ヘルプ

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc n ov z c oab sab IPO Search (Ctrl+I)

Start Page x usb.h x HardwareProfile - PIC24FJ64GB002 PIM.h

Projects Files Navigator

```
94  /** SWITCH *****/
95  #define mInitSwitch2()      TRISBbits.TRISB5=1;           //TRISAbits.TRISA10=1;
96  #define mInitSwitch3()      TRISAbits.TRISA4=1;          //TRISAbits.TRISA9=1;
97  #define InitAllSwitches()   mInitSwitch2();                mInitSwitch3();
98  #define sw2                  PORTBbits.RB5               //PORTAbits.RA10
99  #define sw3                  PORTAbits.RA4               //PORTAbits.RA9
100
101 #define Switch1Pressed()    ((PORTBbits.RB5 == 0)? TRUE : FALSE)
102 #define Switch2Pressed()    ((PORTAbits.RA4 == 0)? TRUE : FALSE)
103 #define Switch3Pressed()    FALSE
104 #define Switch4Pressed()    FALSE
105
106 /** POT *****/
107
108 #define mInitPOT()          {AD1PCFG1bits.PCFG5 = 0;        AD1CON2bits.VCFG = 0x0;        AD1CON3bits.ADCS = 0xFF;        AD1CON1bits.SSRC = 0x0;
109
110 /** I/O pin definitions *****/
111 #define INPUT_PIN 1
112 #define OUTPUT_PIN 0
113
114 /** UART *****/
115 #define BAUDRATE2          57600UL
116 #define BRG_DIV2            4
117 #define BRGH2              1
118
119 // #include <p24fxxxx.h>
120 // #include <uart2.h>
121
122 #endif //HARDWARE_PROFILE_PIC24FJ64GB002_PIM_H
123
```

Search Results Tasks Output

1 | 1 INS

[OpenAccDemo dwo... [アップデート・マネー... [hideo@amdx6: ~/O... Basic Android Acces...

# ADCの設定

アプリケーション 場所 システム フォルダ ? ハードウェア

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO Search (Ctrl+I)

Start Page x usb.h x HardwareProfile - PIC24FJ64GB002 PIM.h

Projects Files Navigator

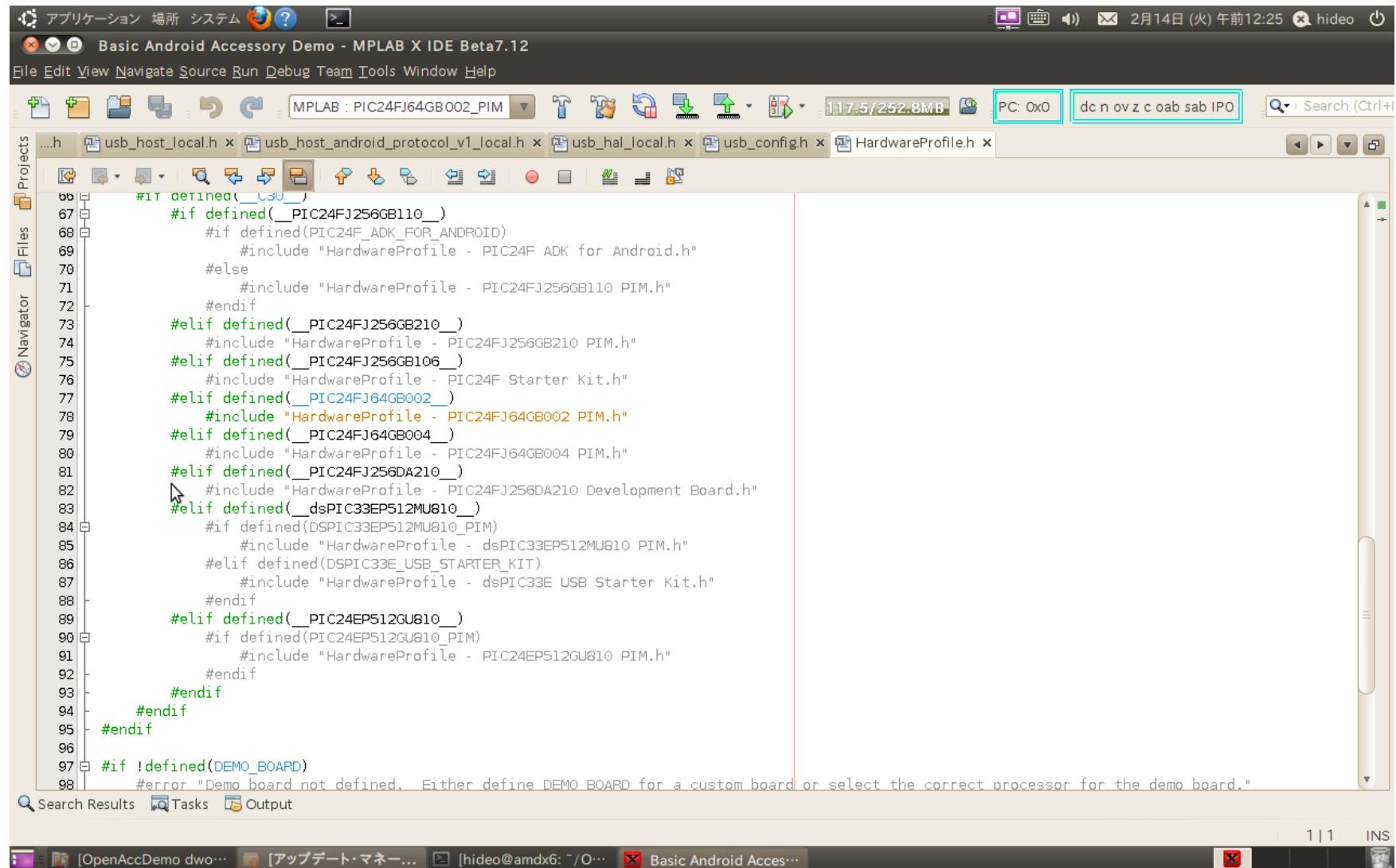
```
94  /** SWITCH *****/
95  #define mInitSwitch2()      TRISBbits.TRISB5=1;           //TRISAbits.TRISA10=1;
96  #define mInitSwitch3()      TRISAbits.TRISA4=1;          //TRISAbits.TRISA9=1;
97  #define InitAllSwitches()   mInitSwitch2();                mInitSwitch3();
98  #define sw2                  PORTBbits.RB5               //PORTAbits.RA10
99  #define sw3                  PORTAbits.RA4               //PORTAbits.RA9
100
101 #define Switch1Pressed()    ((PORTBbits.RB5 == 0)? TRUE : FALSE)
102 #define Switch2Pressed()    ((PORTAbits.RA4 == 0)? TRUE : FALSE)
103 #define Switch3Pressed()    FALSE
104 #define Switch4Pressed()    FALSE
105
106 /** POT *****/
107
108 #define mInitPOT()          {AD1PCFG1bits.PCFG5 = 0;        AD1CON2bits.VCFG = 0x0;        AD1CON3bits.ADCS = 0xFF;        AD1CON1bits.SSRC = 0x0;}
109
110 /** I/O pin definitions *****/
111 #define INPUT_PIN 1
112 #define OUTPUT_PIN 0
113
114 /** UART *****/
115 #define BAUDRATE2          57600UL
116 #define BRG_DIV2            4
117 #define BRGH2              1
118
119 // #include <p24fxxxx.h>
120 // #include <uart2.h>
121
122 #endif //HARDWARE_PROFILE_PIC24FJ64GB002_PIM_H
123
```

Search Results Tasks Output

OpenAccDemo dwo... [アップデート・マネー... [hideo@amdx6: ~/O... Basic Android Acces...

1 | 1 INS

# 型番依存部の切り替え



# 処理対象のVid,Pidの定義

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Window Help

MPLAB : PIC24FJ64GB002\_PIM

121.9/252.8MB PC: 0x0 dc n ov z c oab sab IPO Search (Ctrl+)

```
46 CLIENT_DRIVER_TABLE usbClientDrvTable[NUM_CLIENT_DRIVER_ENTRIES] =
47 {
48     {
49         AndroidAppInitialize,
50         AndroidAppEventHandler,
51         AndroidAppDataEventHandler,
52         0
53     },
54     {
55         AndroidAppInitialize,
56         AndroidAppEventHandler,
57         AndroidAppDataEventHandler,
58         ANDROID_INIT_FLAG_BYPASS_PROTOCOL
59     }
60 };
61
62 // ****
63 // USB Embedded Host Targeted Peripheral List (TPL)
64 // ****
65 USB_TPL usbTPL[NUM_TPL_ENTRIES] =
66 {
67     {
68         /*[1] Device identification information
69          [2] Initial USB configuration to use
70          [3] Client driver table entry
71          [4] Flags (HNP supported, client driver entry, SetConfiguration() commands allowed)
72
73             [1]           [2][3] [4]
74
75             { INIT_VID_PID( 0x18D1ul, 0x2D00ul ), 0, 1, {0} }, // Android accessory
76             { INIT_VID_PID( 0x18D1ul, 0x2D01ul ), 0, 1, {0} }, // Android accessory
77             { INIT_VID_PID( 0xFFFFul, 0xFFFFul ), 0, 0, {0} } // Enumerates everything
78     }
79 }
```

Search Results Tasks Output

OpenAccDemo.dwo... アップデート・マネ... [hideo@amdx6: ~] /O... Basic Android Acces...

1 | 1 INS

アプリケーション 場所 システム ヘルプ

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO Search (Ctrl+I)

usb\_hal\_local.h x usb\_config.h x HardwareProfile.h x main.c x usb\_config.c x usb\_host.c x usb\_host\_android.c x

Projects Files Navigator

```
79     typedef BYTE (*ANDROID_APP_WRITE) (void* handle, BYTE* data, DWORD size);
80     typedef BOOL (*ANDROID_APP_IS_WRITE_COMPLETE) (void* handle, BYTE* errorCode, DWORD* size);
81     typedef BYTE (*ANDROID_APP_READ) (void* handle, BYTE* data, DWORD size);
82     typedef BOOL (*ANDROID_APP_IS_READ_COMPLETE) (void* handle, BYTE* errorCode, DWORD* size);
83     typedef void* (*ANDROID_APP_INITIALIZE) (BYTE address, DWORD flags, BYTE clientDriverID );
84     typedef void (*ANDROID_APP_TASKS) (void);
85
86     typedef struct _ANDROID_PROTOCOL_VERSION
87     {
88         BYTE versionNumber;
89
89         ANDROID_APP_WRITE write;
90         ANDROID_APP_IS_WRITE_COMPLETE isWriteComplete;
91
92         ANDROID_APP_READ read;
93         ANDROID_APP_IS_READ_COMPLETE isReadComplete;
94
95         ANDROID_APP_INITIALIZE init;
96         ANDROID_APP_TASKS tasks;
97
98         USB_CLIENT_EVENT_HANDLER handler;
99         USB_CLIENT_EVENT_HANDLER dataHandler;
100    } ANDROID_PROTOCOL_VERSION;
101
102 /**
103  * Command transfer code */
104 #define ANDROID_ACCESSORY_GET_PROTOCOL 51
105
106 /**
107  * Internal prototypes
108 */
109 BYTE AndroidCommandGetProtocol(ANDROID_DEVICE_DATA* device, WORD *protocol);
110
```

Search Results Tasks Output

1 | 1 INS

# PICのファンクション指定

アプリケーション 場所 システム ヘルプ

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO Search (Ctrl+I)

Projects Files Navigator

...h usb\_config.h x HardwareProfile.h x main.c x usb\_config.c x usb\_host.c x usb\_host\_android.c x usb\_host\_android\_protocol\_v1.c x

```
#if defined(__PIC24FJ256GB110__)
    _CONFIG2(FNOSC_PRIPLL & POSCMOD_HS & PLL_96MHZ_ON & PLLDIV_DIV2) // Primary HS OSC with PLL, USBPLL /2
    _CONFIG1(JTAGEN_OFF & FWDTEN_OFF & ICS_PGx2) // JTAG off, watchdog timer off
#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_WPFF0 & SOSCSEL_IO & WUTSEL_LEG & WPDIS_WPDIS & WPCFG_WPCFGDIS & WPEND_WPENDMEM)
    _CONFIG4(DSWDTPS_DSWDTPS3 & DSWDTCOSC_LPRC & RTCOSC_SOSC & DSBOREN_OFF & DSWDTEN_OFF)
#elif defined(__PIC24FJ64GB004__)
    _CONFIG1(WDTPS_PS1 & FWPSA_PR32 & WINDIS_OFF & FWDTEN_OFF & ICS_PGx1 & GWRP_OFF & GCP_OFF & JTAGEN_OFF)
    _CONFIG2(POSCMOD_HS & I2C1SEL_PRI & IOL1WAY_OFF & OSCIOFNC_ON & FCKSM_CSDCMD & FNOSC_PRIPLL & PLL96MHZ_ON & PLLDIV_DIV2 & IESO_ON)
    _CONFIG3(WPFP_WPFF0 & SOSCSEL_SOSC & WUTSEL_LEG & WPDIS_WPDIS & WPCFG_WPCFGDIS & WPEND_WPENDMEM)
    _CONFIG4(DSWDTPS_DSWDTPS3 & DSWDTCOSC_LPRC & RTCOSC_SOSC & DSBOREN_OFF & DSWDTEN_OFF)
#elif defined(__PIC24FJ256GB106__)
    _CONFIG1( JTAGEN_OFF & GCP_OFF & GWRP_OFF & COE_OFF & FWDTEN_OFF & ICS_PGx2)
    _CONFIG2( 0xFF & IESO_OFF & FCKSM_CSDCMD & OSCIOFNC_OFF & POSCMOD_HS & FNOSC_PRIPLL & PLLDIV_DIV3 & IOL1WAY_ON)
#elif defined(__PIC24FJ256DA210__) || defined(__PIC24FJ256GB210__)
    _CONFIG1(FWDTEN_OFF & ICS_PGx2 & GWRP_OFF & GCP_OFF & JTGEN_OFF)
    _CONFIG2(POSCMOD_HS & IOL1WAY_ON & OSCIOFNC_ON & FCKSM_CSDCMD & FNOSC_PRIPLL & PLL96MHZ_ON & PLLDIV_DIV2 & IESO_OFF)
#elif defined(IOTIO)
    _CONFIG1(FWDTEN_OFF & ICS_PGx2 & GWRP_OFF & GCP_OFF & JTGEN_OFF)
    _CONFIG2(POSCMOD_NONE & IOL1WAY_ON & OSCIOFNC_ON & FCKSM_CSDCMD & FNOSC_FRCPLL & PLL96MHZ_ON & PLLDIV_NODIV & IESO_OFF)
#elif defined(_dsPIC33EP512MU810_) || defined(PIC24EP512GU810_PIM)
    _FOSCSEL(FNOSC_FRC);
    _FOSC(FCKSM_CSECMD & OSCIOFNC_OFF & POSCMD_XT);
    _FWDT(FWDTEN_OFF);
#endif
#elif defined( __PIC32MX__ )
// #pragma config UPLLLEN = ON          // USB PLL Enabled
// #pragma config FPLL_MUL = MUL_15      // PLL Multiplier
// #pragma config UPLLIDIV = DIV_2        // USB PLL Input Divider
// #pragma config FPLLIDIV = DIV_2        // PLL Input Divider
// #pragma config FPLL_ODIV = DIV_1       // PLL Output Divider

```

Search Results Tasks Output

[OpenAccDemo dwo... [アップデート・マネー... [hideo@amdx6: ~/O... Basic Android Acces...

1 | 1 INS

# Appとのプロトコル定義

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar indicates the project is "Basic Android Accessory Demo - MPLAB X IDE Beta 7.12". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, and Help. The toolbar contains various icons for file operations like Open, Save, and Build. The status bar at the bottom shows "PC: 0x0", "dc nov z c oab sab IPO", the date "2月14日(火)午前12:42", and the user "hideo".

The main workspace displays the "main.c" source code. The code defines enum constants for application commands and a struct for data packets. It also includes local prototypes and variables.

```
...h  usb_config.h x  HardwareProfile.h x  main.c x  usb_config.c x  usb_host.c x  usb_host_android.c x  usb_host_android_protocol_v1.c x
Projects  Files  Navigator
154 //Definitions of the various application commands that can be sent
155 typedef enum _ACCESSORY_DEMO_COMMANDS
156 {
157     COMMAND_SET_LEDS          = 0x01,
158     COMMAND_UPDATE_PUSHBUTTONS = 0x02,
159     COMMAND_UPDATE_POT        = 0x03,
160     COMMAND_APP_CONNECT       = 0xFE,
161     COMMAND_APP_DISCONNECT    = 0xFF
162 } ACCESSORY_DEMO_COMMANDS;
163
164 //Creation of the data packet that is going to be sent. In this example
165 // there is just a command code and a one byte data.
166 typedef struct __attribute__((packed))
167 {
168     BYTE command;
169     BYTE data;
170 } ACCESSORY_APP_PACKET;
171
172 //Local prototypes
173 #if defined(__C32__)
174 static void InitPIC32(void);
175 #endif
176
177 static void SetLEDs(BYTE setting);
178 static BYTE GetPushbuttons(void);
179 static BYTE ReadPOT(void);
180
181 //local variables
182 static BYTE read_buffer[64];
183 static ACCESSORY_APP_PACKET outgoing_packet;
184 static void* device_handle = NULL;
185 static BOOL device_attached = FALSE;
186
```

At the bottom, there are tabs for "Search Results", "Tasks", and "Output". The status bar at the very bottom shows "[OpenAccDemo dwo...]" and "[hideo@amdx6: ~/O...]".

# アクセサリの識別情報

The screenshot shows the MPLAB X IDE Beta 7.12 interface with the following details:

- Title Bar:** Basic Android Accessory Demo - MPLAB X IDE Beta 7.12
- Toolbar:** Includes icons for Application, Places, System, File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help.
- Status Bar:** 2月14日(火) 午前12:42 hideo 86.0/252.8MB
- Project Explorer:** Shows files like usb\_config.h, HardwareProfile.h, main.c, etc.
- Code Editor:** Displays the main.c file content. The code defines static variables for device information and an `ANDROID_ACCESSORY_INFORMATION` structure. The `myDeviceInfo` variable is initialized with values for manufacturer, model, description, version, uri, and serial.

```
175 L #endif
176
177 static void SetLEDs(BYTE setting);
178 static BYTE GetPushbuttons(void);
179 static BYTE ReadPOT(void);
180
181 //local variables
182 static BYTE read_buffer[64];
183 static ACCESSORY_APP_PACKET outgoing_packet;
184 static void* device_handle = NULL;
185 static BOOL device_attached = FALSE;
186
187 static char manufacturer[] = "Microchip Technology Inc.";
188 static char model[] = "Basic Accessory Demo";
189 static char description[] = DEMO_BOARD_NAME_STRING;
190 static char version[] = "2.0";
191 static char uri[] = "http://www.microchip.com/android";
192 static char serial[] = "N/A";
193
194 ANDROID_ACCESSORY_INFORMATION myDeviceInfo =
195 {
196     manufacturer,
197     sizeof(manufacturer),
198     model,
199     sizeof(model),
200     description,
201     sizeof(description),
202     version,
203     sizeof(version),
204     uri,
205     sizeof(uri),
206     serial,
207     ...
```

- Search Bar:** PC: 0x0 dc nov zc oab sab IPO
- Bottom Status:** Search Results, Tasks, Output
- Bottom Navigation:** OpenAccDemo dwo..., アップデート・マネ..., [hideo@amdx6: ~] /0... Basic Android Acces..., 1 | 1 INS

# 46

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar reads "Basic Android Accessory Demo - MPLAB X IDE Beta7.12". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help. The toolbar has various icons for file operations like Open, Save, Print, and Build. The status bar shows "90.0/252.8MB" and "PC: 0x0". The main window displays a C source code editor for "main.c". The code is as follows:

```
232     *****/
233     int main(void)
234     {
235         DWORD size = 0;
236         BOOL responseNeeded;
237         BYTE pushButtonValues = 0xFF;
238         BYTE potPercentage = 0xFF;
239         BOOL buttonsNeedUpdate = FALSE;
240         BOOL potNeedsUpdate = FALSE;
241         BOOL readyToRead = TRUE;
242         BOOL writeInProgress = FALSE;
243         BYTE tempValue = 0xFF;
244         BYTE errorCode;
245         ACCESSORY_APP_PACKET* command_packet = NULL;
246
247         BOOL connected_to_app = FALSE;
248         BOOL need_to_disconnect_from_app = FALSE;
249
250 #if defined(__PIC32MX__)
251
252 #if defined(__dsPIC33EP512MU810__) || defined (__PIC24EP512GU810__)
253
254 // futaba
255 #if defined(__PIC24FJ64GB002__)
256     //On the PIC24FJ64GB004 Family of USB microcontrollers, the PLL will not power up and be enabled
257     //by default, even if a PLL enabled oscillator configuration is selected (such as HS+PLL).
258     //This allows the device to power up at a lower initial operating frequency, which can be
259     //advantageous when powered from a source which is not gauranteed to be adequate for 32MHz
260     //operation. On these devices, user firmware needs to manually set the CLKDIV<PLLEN> bit to
261     //power up the PLL.
262     {
263         unsigned int pll_startup_counter = 600;
264     }
265
266 #endif
267
268 #endif
269
270 }
```

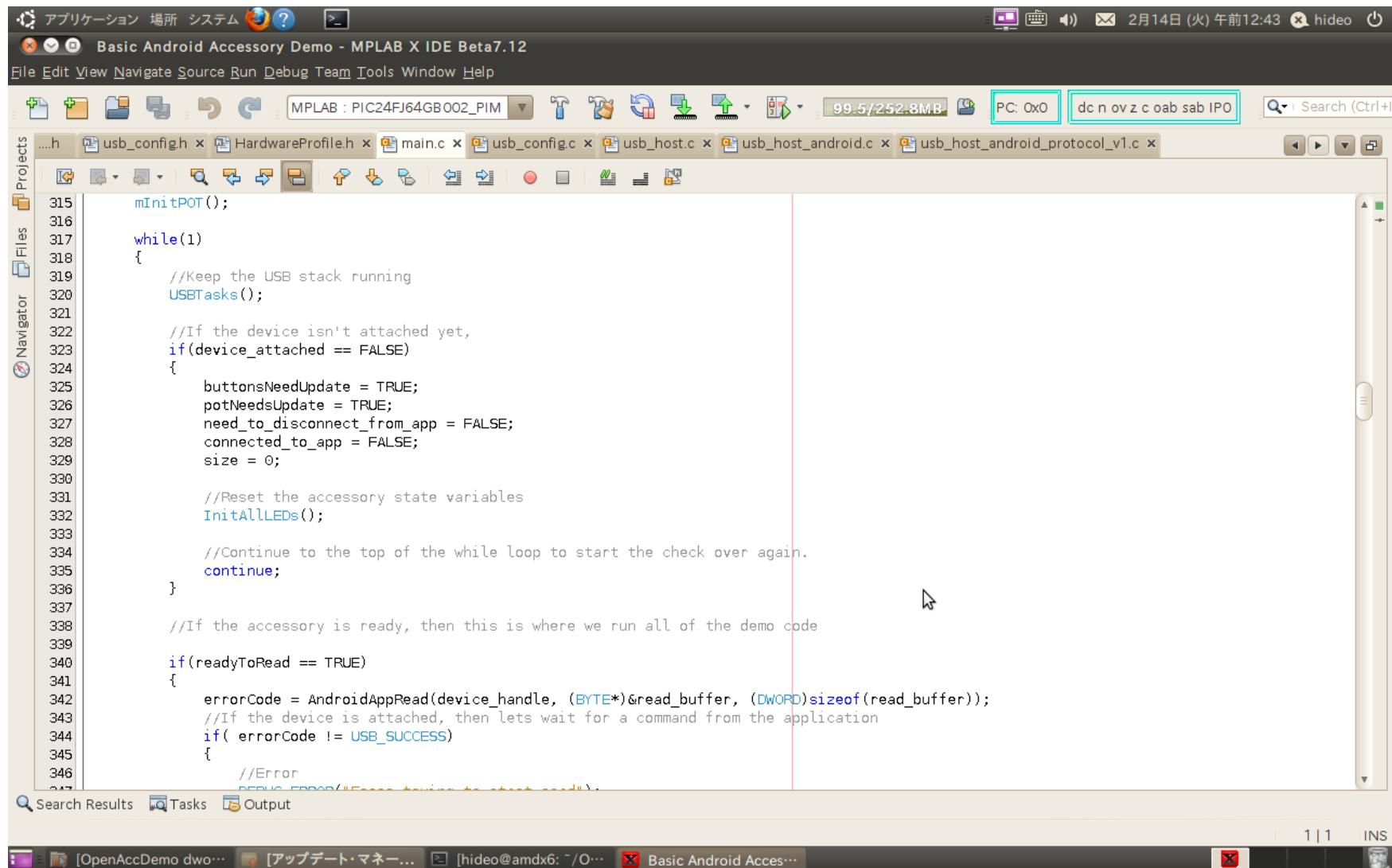
The code handles USB communication, manages button and potentiometer values, and manages connections to an Android application. It includes conditional compilation for different PIC32MX and dsPIC33EP families, specifically targeting the PIC24FJ64GB002 device.

# PLLモードの設定/POTの初期化

The screenshot shows the MPLAB X IDE Beta 7.12 interface with the project "Basic Android Accessory Demo". The main window displays the file "main.c" containing C code for setting up the PLL and initializing a POT. A red vertical line highlights the code from line 291 to 322. The status bar at the bottom shows tabs for "OpenAccDemo dwo...", "アップデート・マネー...", "[hideo@amdx6: ~] /O...", and "Basic Android Acces...".

```
291 // futaba
292 #if defined(__PIC24FJ64GB002__)
293 //On the PIC24FJ64GB004 Family of USB microcontrollers, the PLL will not power up and be enabled
294 //by default, even if a PLL enabled oscillator configuration is selected (such as HS+PLL).
295 //This allows the device to power up at a lower initial operating frequency, which can be
296 //advantageous when powered from a source which is not gauranteed to be adequate for 32MHz
297 //operation. On these devices, user firmware needs to manually set the CLKDIV<PLLEN> bit to
298 //power up the PLL.
299 {
300     unsigned int pll_startup_counter = 600;
301     CLKDIVbits.PLLEN = 1;
302     while(pll_startup_counter--);
303 }
304
305 AD1PCFG = 0xffff;
306 CLKDIV = 0x0000;      // Set PLL prescaler (1:1)
307
308 #endif
309
310 USBInitialize();
311 AndroidAppStart(&myDeviceInfo);
312
313 responseNeeded = FALSE;
314 mInitPOT();
315
316 while(1)
317 {
318     //Keep the USB stack running
319     USBTasks();
320
321     //If the device isn't attached yet,
322     //then don't do anything
323 }
```

# メインループの入り口



The screenshot shows the MPLAB X IDE Beta 7.12 interface with the project "Basic Android Accessory Demo". The main window displays the "main.c" source code. The code implements a main loop that initializes the USB stack and enters a continuous loop to handle USB tasks. It includes logic to check if the device is attached and to read data from the application. The code is annotated with line numbers from 315 to 347.

```
315 mInitPOT();
316
317 while(1)
318 {
319     //Keep the USB stack running
320     USBTasks();
321
322     //If the device isn't attached yet,
323     if(device_attached == FALSE)
324     {
325         buttonsNeedUpdate = TRUE;
326         potNeedsUpdate = TRUE;
327         need_to_disconnect_from_app = FALSE;
328         connected_to_app = FALSE;
329         size = 0;
330
331         //Reset the accessory state variables
332         InitAllLEDs();
333
334         //Continue to the top of the while loop to start the check over again.
335         continue;
336     }
337
338     //If the accessory is ready, then this is where we run all of the demo code
339
340     if(readyToRead == TRUE)
341     {
342         errorCode = AndroidAppRead(device_handle, (BYTE*)&read_buffer, (DWORD)sizeof(read_buffer));
343         //If the device is attached, then lets wait for a command from the application
344         if( errorCode != USB_SUCCESS)
345         {
346             //Error
347             //PC: 0x0 dc nov z c oab sab IPO
348         }
349     }
350 }
```

# 50

The screenshot shows the MPLAB X IDE Beta 7.12 interface with the following details:

- Title Bar:** Basic Android Accessory Demo - MPLAB X IDE Beta 7.12
- Toolbar:** Includes standard icons for file operations, project management, and debugging.
- Status Bar:** Shows memory usage (80.8/252.8MB), PC address (PC: 0x0), and a search bar.
- Project Explorer:** Lists files: ...h, usb\_config.h, HardwareProfile.h, main.c, usb\_config.c, usb\_host.c, usb\_host\_android.c, and usb\_host\_android\_protocol\_v1.c.
- Code Editor:** Displays the main.c file content. The code handles USB reads from an Android device, checking for errors and processing commands if successful.

```
339     if(readyToRead == TRUE)
340     {
341         errorCode = AndroidAppRead(device_handle, (BYTE*)&read_buffer, (DWORD)sizeof(read_buffer));
342         //If the device is attached, then lets wait for a command from the application
343         if( errorCode != USB_SUCCESS)
344         {
345             //Error
346             DEBUG_ERROR("Error trying to start read");
347         }
348         else
349         {
350             readyToRead = FALSE;
351         }
352     }
353
354     size = 0;
355
356     if(AndroidAppIsReadComplete(device_handle, &errorCode, &size) == TRUE)
357     {
358         //We've received a command over the USB from the Android device.
359         if(errorCode == USB_SUCCESS)
360         {
361             //Maybe process the data here. Maybe process it somewhere else.
362             command_packet = (ACCESSORY_APP_PACKET*)&read_buffer[0];
363         }
364         else
365         {
366             //Error
367             DEBUG_ERROR("Error trying to complete read request");
368         }
369     }
370 }
```

- Bottom Navigation:** Search Results, Tasks, Output.
- Bottom Status:** Shows multiple tabs: [OpenAccDemo dwo...], [アップデート・マネ...], [hideo@amdx6: ~ /O...], and Basic Android Acces... with a red 'X' icon.
- Bottom Right:** Page number (1 | 1) and INS indicator.

# 51

アプリケーション 場所 システム ヘルプ

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO

Search (Ctrl+I)

Projects

...h usb\_config.h x HardwareProfile.h x main.c x usb\_config.c x usb\_host.c x usb\_host\_android.c x usb\_host\_android\_protocol\_v1.c x

Files

Navigator

```
372     while(size > 0)
373     {
374         if(connecting_to_app == FALSE)
375         {
376             if(command_packet->command == COMMAND_APP_CONNECT)
377             {
378                 connecting_to_app = TRUE;
379                 need_to_disconnect_from_app = FALSE;
380             }
381         }
382         else
383         {
384             switch(command_packet->command)
385             {
386                 case COMMAND_SET_LEDS:
387                     SetLEDs(command_packet->data);
388                     break;
389
390                 case COMMAND_APP_DISCONNECT:
391                     need_to_disconnect_from_app = TRUE;
392                     break;
393
394                 default:
395                     //Error, unknown command
396                     DEBUG_ERROR("Error: unknown command received");
397                     break;
398             }
399         }
400     }
401     //All commands in this example are two bytes, so remove that from the queue
402     size -= 2;
403     //And move the pointer to the next packet (this works because
```

Search Results Tasks Output

1 | 1 INS

[OpenAccDemo dwo... [アップデート・マネー... [hideo@amdx6: ~/O... Basic Android Acces...

# 52

アプリケーション 場所 システム ヘルプ

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO

Search (Ctrl+I)

Projects

Files

Navigator

```
...h usb_config.h x HardwareProfile.h x main.c x usb_config.c x usb_host.c x usb_host_android.c x usb_host_android_protocol_v1.c x
```

402 size -= 2;  
403 //And move the pointer to the next packet (this works because  
404 // all command packets are 2 bytes. If variable packet size  
405 // then need to handle moving the pointer by the size of the  
406 // command type that arrived.  
407 command\_packet++;  
408  
409 if(need\_to\_disconnect\_from\_app == TRUE)  
410 {  
411 break;  
412 }  
413  
414  
415 if(size == 0)  
416 {  
417 readyToRead = TRUE;  
418 }  
419  
420 //Get the current pushbutton settings  
421 tempValue = GetPushbuttons();  
422  
423 //If the current button settings are different than the last time  
424 // we read the button values, then we need to send an update to the  
425 // attached Android device  
426 if(tempValue != pushButtonValues)  
427 {  
428 buttonsNeedUpdate = TRUE;  
429 pushButtonValues = tempValue;  
430 }  
431  
432 //Get the current potentiometer setting  
433 tempValue = ReadPOT();

Search Results Tasks Output

[OpenAccDemo dwo... [アップデート・マネー... [hideo@amdx6: ~/O... Basic Android Acces...

1 | 1 INS

# 53

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar indicates the project is "Basic Android Accessory Demo". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, and Help. The toolbar contains various icons for file operations like Open, Save, and Build. The status bar at the bottom shows the date and time as "2月14日 (火) 午前12:44" and the user name as "hideo". The main workspace displays a portion of the "main.c" source code. The code handles USB communication, specifically reading potentiometer values and managing write operations to an Android device. A red vertical line highlights a section of the code from line 432 to line 463. The code editor has syntax highlighting for C/C++ and includes comments explaining its functionality. The right side of the interface features a scroll bar and a vertical toolbar with icons for navigation and search.

```
432 //Get the current potentiometer setting
433 tempValue = ReadPOT();
434
435 //If it is different than the last time we read the pot, then we need
436 // to send it to the Android device
437 if(tempValue != potPercentage)
438 {
439     potNeedsUpdate = TRUE;
440     potPercentage = tempValue;
441 }
442
443 //If there is a write already in progress, we need to check its status
444 if( writeInProgress == TRUE )
445 {
446     if(AndroidAppIsWriteComplete(device_handle, &errorCode, &size) == TRUE)
447     {
448         writeInProgress = FALSE;
449         if(need_to_disconnect_from_app == TRUE)
450         {
451             connected_to_app = FALSE;
452             need_to_disconnect_from_app = FALSE;
453         }
454
455         if(errorCode != USB_SUCCESS)
456         {
457             //Error
458             DEBUG_ERROR("Error trying to complete write");
459         }
460     }
461 }
462
463 if((need_to_disconnect_from_app == TRUE) && (writeInProgress == FALSE))
```

# 54

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar reads "Basic Android Accessory Demo - MPLAB X IDE Beta7.12". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help. The toolbar has various icons for file operations like Open, Save, Print, and Build. The status bar at the bottom shows "OpenAccDemo dwo..." and "[hideo@amdx6: ~/O... Basic Android Acces...]" along with a power icon and the date/time "2月14日 (火) 午前12:45".

The main window displays the code for "main.c". A red vertical line highlights a section of the code from line 462 to line 493. The code handles USB communication and button updates:

```
462     if((need_to_disconnect_from_app == TRUE) && (writeInProgress == FALSE))
463     {
464         outgoing_packet.command = COMMAND_APP_DISCONNECT;
465         outgoing_packet.data = 0;
466         writeInProgress = TRUE;
467
468         errorCode = AndroidAppWrite(device_handle,(BYTE*)&outgoing_packet, 2);
469         if( errorCode != USB_SUCCESS )
470         {
471             DEBUG_ERROR("Error trying to send button update");
472         }
473     }
474
475     if(connecting_to_app == FALSE)
476     {
477         //If the app hasn't told us to start sending data, let's not do anything else.
478         continue;
479     }
480
481
482     //If we need up update the button status on the Android device and we aren't
483     // already busy in a write, then we can send the new button data.
484     if((buttonsNeedUpdate == TRUE) && (writeInProgress == FALSE))
485     {
486         outgoing_packet.command = COMMAND_UPDATE_PUSHBUTTONS;
487         outgoing_packet.data = pushButtonValues;
488
489         errorCode = AndroidAppWrite(device_handle,(BYTE*)&outgoing_packet, 2);
490         if( errorCode != USB_SUCCESS )
491         {
492             DEBUG_ERROR("Error trying to send button update");
493         }
494     }
495 }
```

Below the code editor, there are tabs for "Search Results", "Tasks", and "Output". The status bar at the bottom right shows "1 | 1 INS".

# 55

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar reads "Basic Android Accessory Demo - MPLAB X IDE Beta7.12". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help. The toolbar has various icons for file operations like Open, Save, Print, and Build. The status bar shows memory usage "88.0/252.8MB" and a connection status "PC: 0x0 dc nov z c oab sab IPO". The main window displays a C code editor for "main.c". The code handles button and potentiometer updates to an Android device via USB. A red vertical line highlights a section of the code from line 483 to line 514. The code editor also features a search bar at the top right.

```
483 // already busy in a write, then we can send the new button data.
484 if((buttonsNeedUpdate == TRUE) && (writeInProgress == FALSE))
485 {
486     outgoing_packet.command = COMMAND_UPDATE_PUSHBUTTONS;
487     outgoing_packet.data = pushButtonValues;
488
489     errorCode = AndroidAppWrite(device_handle,(BYTE*)&outgoing_packet, 2);
490     if( errorCode != USB_SUCCESS )
491     {
492         DEBUG_ERROR("Error trying to send button update");
493     }
494
495     buttonsNeedUpdate = FALSE;
496     writeInProgress = TRUE;
497 }
498
499 //If we need up update the pot status on the Android device and we aren't
500 // already busy in a write, then we can send the new pot data.
501 if((potNeedsUpdate == TRUE) && (writeInProgress == FALSE))
502 {
503     outgoing_packet.command = COMMAND_UPDATE_POT;
504     outgoing_packet.data = potPercentage;
505
506     errorCode = AndroidAppWrite(device_handle,(BYTE*)&outgoing_packet, 2);
507     if( errorCode != USB_SUCCESS )
508     {
509         DEBUG_ERROR("Error trying to send pot update");
510     }
511
512     potNeedsUpdate = FALSE;
513     writeInProgress = TRUE;
514 }
```

Search Results Tasks Output

1 | 1 INS

# SetLEDs 56

アプリケーション 場所 システム ホーム ? フォルダ

2月14日(火) 午前12:46 hideo

Basic Android Accessory Demo - MPLAB X IDE Beta7.12

File Edit View Navigate Source Run Debug Team Tools Window Help

MPLAB : PIC24FJ64GB002\_PIM

PC: 0x0 dc nov zc oab sab IPO Search (Ctrl+I)

Projects

Files

Navigator

...h usb\_config.h x HardwareProfile.h x main.c x usb\_config.c x usb\_host.c x usb\_host\_android.c x usb\_host\_android\_protocol\_v1.c x

```
708 static void SetLEDs(BYTE setting)
709 {
710     if((setting & 0x01) == 0x01) { LED0_On(); } else { LED0_Off(); }
711     if((setting & 0x02) == 0x02) { LED1_On(); } else { LED1_Off(); }
712     if((setting & 0x04) == 0x04) { LED2_On(); } else { LED2_Off(); }
713     if((setting & 0x08) == 0x08) { LED3_On(); } else { LED3_Off(); }
714     if((setting & 0x10) == 0x10) { LED4_On(); } else { LED4_Off(); }
715     if((setting & 0x20) == 0x20) { LED5_On(); } else { LED5_Off(); }
716     if((setting & 0x40) == 0x40) { LED6_On(); } else { LED6_Off(); }
717     if((setting & 0x80) == 0x80) { LED7_On(); } else { LED7_Off(); }
718 }
```

\*\*\*\*\*

Function:  
BYTE GetPushbuttons(void)

Summary:  
Reads the current push button status.

Description:  
Reads the current push button status.

Precondition:  
None

Parameters:  
None

Return Values:  
BYTE - bitmap for button representations (1 = pressed, 0 = not pressed)  
bit 0 = button 1  
bit 1 = button 2  
bit 2 = button 3

Search Results Tasks Output

1 | 1 INS

[OpenAccDemo dwo... [アップデート・マネ... [hideo@amdx6: ~/O... Basic Android Acces...

# GetPushbuttons 57

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The main window displays the 'main.c' file with the following code:

```
744     None
745     ****
746     static BYTE GetPushbuttons(void)
747     {
748         BYTE toReturn;
749
750         InitAllSwitches();
751
752         toReturn = 0;
753
754         if(Switch1Pressed()){toReturn |= 0x1;}
755         if(Switch2Pressed()){toReturn |= 0x2;}
756         if(Switch3Pressed()){toReturn |= 0x4;}
757         if(Switch4Pressed()){toReturn |= 0x8;}
758
759         return toReturn;
760     }
761
762
763     ****
764     Function:
765     BYTE ReadPOT(void)
766
767     Summary:
768     Reads the pot value and returns the percentage of full scale (0-100)
769
770     Description:
771     Reads the pot value and returns the percentage of full scale (0-100)
772
773     Precondition:
774     A/D is initialized properly
775
776
```

The code implements a function 'GetPushbuttons' that reads four pushbutton states and returns a byte value where each bit corresponds to a button's state. It also includes a placeholder function 'ReadPOT' with its documentation.

The IDE interface includes a toolbar, a menu bar (File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help), and various tool buttons. The status bar at the bottom shows the date and time (2月14日 (火) 午前12:46), the user (hideo), and the project name (Basic Android Access...).

# POTの読み取り

The screenshot shows the MPLAB X IDE Beta 7.12 interface. The title bar reads "Basic Android Accessory Demo - MPLAB X IDE Beta 7.12". The menu bar includes File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help. The toolbar has various icons for file operations. The top status bar shows "103.3/252.8MB" and "PC: 0x0 dc nov z c oab sab IPO". The search bar says "Search (Ctrl+I)". The left sidebar has "Projects", "Files", and "Navigator" sections. The main editor area displays the following C code:

```
...h  usb_config.h x  HardwareProfile.h x  main.c x  usb_config.c x  usb_host.c x  usb_host_android.c x  usb_host_android_protocol_v1.c x
783 None
784 ****
785 static BYTE ReadPOT(void)
786 {
787     WORD_VAL w;
788     DWORD temp;
789
790 #if defined(__18CXX)
791     mInitPOT();
792
793     ADCON0bits.GO = 1;          // Start AD conversion
794     while(ADCON0bits.NOT_DONE); // Wait for conversion
795
796     w.v[0] = ADRESL;
797     w.v[1] = ADRESH;
798
799 #elif defined(__C30__) || defined(__C32__)
800 #if defined(PIC24FJ256GB110_PIM) || \
801     defined(PIC24FJ256DA210_DEV_BOARD) || \
802     defined(PIC24FJ256GB210_PIM)
803
804     AD1CHS = 0x5;             //MUXA uses AN5
805
806     // Get an ADC sample
807     AD1CON1bits.SAMP = 1;      //Start sampling
808     for(w.Val=0;w.Val<1000;w.Val++); //Sample delay, conversion start automatically
809     AD1CON1bits.SAMP = 0;      //Start sampling
810     for(w.Val=0;w.Val<1000;w.Val++); //Sample delay, conversion start automatically
811     while(!AD1CON1bits.DONE);   //Wait for conversion to complete
812
813     temp = (DWORD)ADC1BUFO;
814     temp = temp * 100;
815 }
```

The code implements a function to read from a POT using the ADC module. It handles different PIC families based on compiler definitions. The code uses bit manipulation to control the ADC and read the result.

# ADCの起動と読み込み 59

The screenshot shows the MPLAB X IDE interface with the following details:

- Title Bar:** Basic Android Accessory Demo - MPLAB X IDE Beta7.12
- Toolbar:** Includes icons for File, Edit, View, Navigate, Source, Run, Debug, Team, Tools, Window, Help.
- Status Bar:** Shows memory usage (98.2/252.8MB), PC: 0x0, and search terms (dc n ov z c oab sab IPO).
- Project Explorer (Projects):** Shows files like ...h, usb\_config.h, HardwareProfile.h, main.c, usb\_config.c, usb\_host.c, usb\_host\_android.c, and usb\_host\_android\_protocol\_v1.c.
- Code Editor:** Displays C code for an ADC sampling routine. The code handles three different PIC variants based on defines: PIC24FJ64GB002\_PIM, PIC24FJ64GB004\_PIM, and PIC24F\_STarter\_KIT. It includes comments explaining the sampling process and conversion logic.
- Search Bar:** Located at the bottom left, it contains "Search Results", "Tasks", and "Output".
- Bottom Status Bar:** Shows tabs for OpenAccDemo.dwo, [アップデート・マネー...], [hideo@amdx6: ~ / O...], and Basic Android Acces... along with a progress bar and status indicators.

# **PART5**

# **ANDROID側の仕組み**

おわり