DxLib・mbed を用いた
新入生用春休みの宿題

梶本研究室

Ver.1 2012.1.4

渡すもの：mbed、ブレッドボード、加速度センサ

※※※ 課題を行うにあたって ※※※

1. 特に 1 章の自習は、サンプルソースをコピー&ペーストすると意味がありません。**修行だと思って写経**してください。（一度だけどうしても必要なステップ）

2. わからないことはまず web 検索で調べてください。資料は不完全です。**検索による問題解決能力の向上も本課題全体の重要なテーマです**。英語での検索にも慣れてください。

3. そもそもコンパイルできない等の本質的でないつまづきはメールか研究室に来て質問し、一刻も早く解決してください。**数日止まることは絶対に避けてください。**

4. **自分で作ったバグは宝物**。回避ではなく解決してください。バグを解決したらレベルが 1 上がりますが、回避したら下がります。

5. **最終課題に一ヶ月かけてください**。それ以外をなるべく前倒しで終えてください。

6. 課題によっては何から手をつけて良いか分からず途方に暮れるかもしれません。この宿題はそのような、研究にありがちな状況を疑似体験することも目的にしていますので、**なんとかしてください**。

課題提出日（前倒しで進めてください）

- 1 章　[課題 1] ：2/13(月)締切
- 1 章　[課題 2] ～[課題 5] ：2/20(月)締切
- 2 章　[課題 6] ～[課題 7] ：3/5(月)締切
- 3 章　[課題 8] ～[課題 10] ：3/12(月)締切
- 4 章　[課題 11] ：4/2(月)に研究室にてデモ発表

1　C および C++の学習(目安：2～4 週間)

C or C++を学生実験で少し書いた程度の場合には 2 週間程度かけて学習し、課題に 2 週間かけてください。よく知っている人は課題だけやれば結構です。
[課題 1] は C 言語そのもの、[課題 2] ～[課題 5] は研究用の発展です。[課題 1] までは後学期の授業期間中に行うスケジュールが良いでしょう。

1.1　　環境構築

ここでは Microsoft Visual Studio Express Edition を標準とします。
http://www.microsoft.com/japan/msdn/vstudio/express/
Visual C++ 2010 を選択、インストールしてください。

1.2　　猫でもわかるシリーズで自習

ここでは web 上でまとまっていることから、猫でもわかるシリーズを標準とします。書籍もあります。もし難しいと感じるなら参考文献[1][2]を使ってください。
http://homepage2.nifty.com/c_lang/

C 言語編を 61 章まで、C++編を 11 章までやってください。ただし環境の違い等から、C 言語編の 17、27、36、39、54～57 章は飛ばしてください（コンソール画面の仕様が異なるため。同様のことは例えば 26 章の後半にも言えます。エスケープシーケンスを使う部分を見つけたら省いてください）。さらに先の章のトピックは必要が出てきた時で結構です。

Visual Studio Express Edition の場合、最初のプロジェクト（コンソールアプリ）作成は下記のページなどに従ってください。
http://cvwww.ee.ous.ac.jp/vc10prog.html (岡山理科大学太田研究室のページ)

[課題1] 以下の問題を解け（法政大学佐藤先生の講義資料
http://www.k.hosei.ac.jp/~yuji/C_H16.htm より抜粋させていただきました）

(1) （ビット演算）整数型変数 x, y の値をそれぞれ、x = 0x4567、 y = 0x0f0f とする。x と y の論理積、論理和、排他的論理和、x の補数（否定）を求めるビット演算を行い、それぞれの演算結果（整数）を 16 進数および 10 進数で表示するプログラムを作成せよ。なお x と y の値も同様に表示せよ。

(2) （switch 文による多分岐）月を整数値としてキーボードから入力し、その季節を表示するプログラムを switch 文を用いて作成せよ。

(2)（多重ループ）40 未満の素数を求めるプログラムを作成せよ。

(3)（while 文による反復）1 + 2 + 3 + … + n < 1000 を満たす最も大きな n を求めるプログラムを作成せよ.

(4) (文字関数の作り方)アルファベットの小文字を大文字に変換する関数 cap を定義し、main 関数から小文字を渡し、それを関数 cap で変換された大文字を main 関数で受け取り表示するプログラムを作成せよ。なお元が大文字であればそのままとすること（toupper 関数を自作するということ）

(5)（文字列関数の作り方）(4)のプログラムを改変し、文字列に対して大文字化するようにせよ。関数の引数は文字配列および文字数とせよ。

[課題2]（sprintf 関数、CSV 形式での保存）実験中、データを保存するためのファイルを作成する必要がある場合がある。この練習として、プログラム開始直後に名前の入力を促し、さらに時刻を取得することで、例えば「yamada_2012Jan27_23_15_15.csv」という形式の名前のファイルを自動生成せよ。
さらに C 言語編第 38 章を元に、サイコロ２つを 10000 回振った時の合計値の頻度分布を CSV 形式で記録するようにせよ。
（ヒント）ファイル入出力は 18-20 章､時計は 26 章｡ファイル名の文字列作成は sprintf 関数。CSV 形式は表計算ソフト、例えば Excel で扱うことが出来る。調べること。

[課題3]（ファイル共有による通信、fseek 関数、kbhit 関数）２つのプログラムの間の通信を、ファイルを介したデータ共有によって行う。一つ目のプログラムでは、ファイルを書き込みモードで開き、キーボード入力した文字を常にファイル先頭に書き込み続けるようにせよ。２つ目のプログラムでは、同じファイルを読み出しモードで開き、ファイル先頭の文字を定期的に表示し続けるようにせよ。
（ヒント）ファイルの先頭にアクセスし続けるために fseek 関数を用いる。50～53 章を参照。書き込み側ではキーボード入力の判定に 26 章の kbhit 関数を用いる。
２つのプログラムでファイルを共有するため、ファイルの場所は絶対パスで指定する方法と相対パスで指定する方法がある。例えばファイル名として、C:¥sharedata を使う場合（絶対パス）は、ファイルオープン時には fopen(“C:¥¥sharedata”,…とする必要がある（¥マークがエスケープシーケンスのため２回重ねて書く）。またファイル名として３つ上のフォルダを指定する場合（相対パス）は fopen(“..¥¥..¥¥..¥¥sharedata”…)等とする。

[課題4] [課題 3] のファイルによる通信は、書き込みは「文字」（1 バイト）でなくても構わないし、1 変数でなくても構わない（例えば画像のような巨大なデータすらやり取りできる）。

ここでは double 型の変数を 2 つ書き込み、読みだすことが出来ることを確認せよ。書き込み側は適当な sin 関数および cos 関数の出力を書き込み続ける。読み出し側はその値を読み出して表示し続けるようにせよ。

（ヒント）50～53 章の write、read 関数を用いる。関数の最後の引数（バイト数）はどうなるべきか。

[課題5] (C++クラス作成) C 言語編 58～61 章のコンソール操作の関数群をまとめ、一つのクラスを作成せよ。その上でサンプルとして、コンソール画面に sin 関数を描画せよ

（ヒント）C++編のクラスの入門を参照。コンソール画面操作特有の変数はクラスのメンバ変数とする。指定場所への描画や画面消去、色の変更などのメンバ関数を定義する。コンストラクタ中に必要な初期化処理を書く。

※※※　課題を行うにあたって　※※※

全ての課題はソースコードを他人に読めるように書く訓練も兼ねています。**課題のソースコードは研究室全体で回覧、添削します。**

- とりあえずなるべく沢山関数を定義し、main 関数をコンパクトにしてください。
- 無意味なグローバル変数を減らし、関数の中で閉じた変数を増やしてください。
- インデントを正確に行ってください。
- コメントを多く入れてください。

こうした注意点は、例えば下記にまとめられています。

http://homepage3.nifty.com/mmgames/c_guide/02-02.html

http://www.softech.co.jp/mm_060607_firm.htm

1.3　参考資料

本文中で触れていない書籍。良書を追いきれていませんのであくまで参考です。

初学者向け

[1] C の絵本―C 言語が好きになる 9 つの扉

[2] C++ の絵本

「猫でもわかる」が難しすぎる…という人もけっこう居ます。が、この絵本シリーズで何とかなったようです。

読み物系。今はまだ早いか

[3] C プログラミング専門課程

[4] 改訂新版 C プログラミング診断室
http://www.pro.or.jp/~fuji/mybooks/cdiag/index.html

「ひどいプログラム・バグを通して学ぶ」という趣旨の本。

[5] エキスパート C プログラミング―知られざる C の深層

昔に出た本ですが非常に面白い読み物。

2　DxLib の学習(目安：2 週間)

3 年生の学生実験で少し使った DxLib を、より使いこなす練習です。
http://homepage2.nifty.com/natupaji/DxLib/

2.1　　環境の構築

下記のページから、VisualC++用(Ver3.06c：2011/12/31 現在)をダウンロード、解凍する。
http://homepage2.nifty.com/natupaji/DxLib/dxdload.html

上で解凍した中にある「サンプルプログラム実行用フォルダ¥DxLib_VC2010 用.sln」を開き、実行する。このフォルダには 3D モデルや画像、wave ファイルなど、サンプルプログラムで使われているコンテンツが用意されている。
下記ページの全 34 サンプルを実行し、どのくらいのことが出来るか把握する。ソースを test.cpp にコピペすれば動く。
http://homepage2.nifty.com/natupaji/DxLib/dxprogram.html

続いて Visual Studio Express Edition で最初のプロジェクト構築を行う。
http://homepage2.nifty.com/natupaji/DxLib/dxuse_vc2010express.html

さらに DxLib 講座の基礎編を行う。**特にダブルバッファの辺りは重要**。
http://homepage2.nifty.com/natupaji/DxLib/dxlecture_main.html

以上の準備ができたら以下の課題を行う。はじめは各課題に 1 週間かかるかもしれない。

[課題6]　(2D) ボールがマウスクリックした場所で発生、自由落下運動し、地面で跳ね返り、振幅が減衰していく様子を **2D で**シミュレートせよ。
・ボールの座標情報などをまとめた構造体を作り、使用すること。
・ボールは DxLib の機能で円を描くの**ではなく**、画像として用意し、**画像の透過機能**も用いること。
・地面で跳ね返る瞬間になんらかの変形ないしアニメーションもさせること。これは画像を伸縮させるの**ではなく**、複数の画像を切り替えることで行う。
・地面の映像も画像として用意し、球が衝突する瞬間に振動させること。
・複数のボールに対応すること。つまり大量にクリックすると大量にボールが出現する。
・**可能な限りコミカルに表現すること**

[課題7]　（3D）スペースキーを押すごとにボールがランダムな場所に発生、自由落下運動し、地面で跳ね返り、振幅が減衰していく様子を **3D で**シミュレートせよ。

・ボールの座標情報などをまとめた構造体を作り、使用すること。

・ボールは DxLib の球描画機能で描き、テクスチャも貼ること。

・マウスのドラッグで視点を変えられるようにすること。

・衝突音のサウンドファイルを探してくることで、地面に衝突した際に衝突の強さに応じて再生されるようにすること。

・照明も工夫すること。

・複数のボールに対応すること。

・**可能な限りリアルに表現すること**

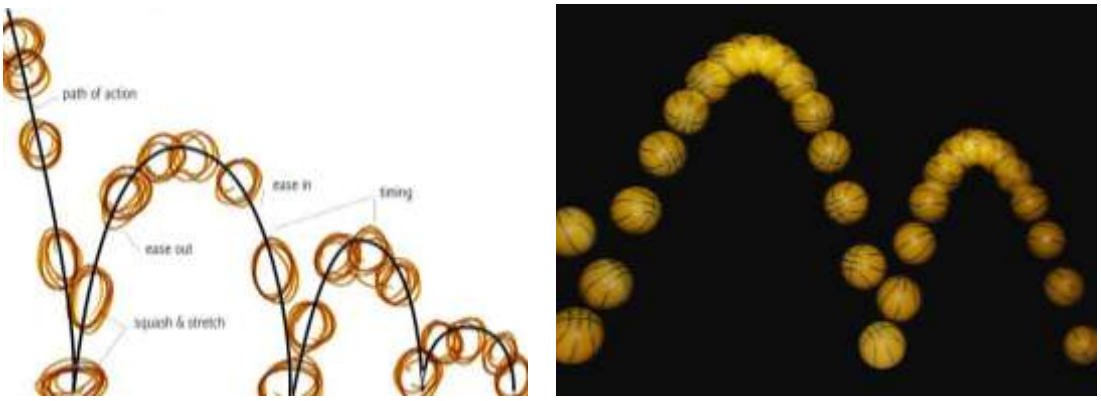


図 1　跳ね返るボールのコミカルな表現とリアルな表現

3　mbed の学習(1～2 日)

mbed を用いた本格的な講習は 4 月以降に行います。ここでは加速度センサを使うまでの最短距離を学習します。

3.1　　導入

エレキジャックの記事が非常に丁寧なので、これを順に追ってください。

- <1>mbed 入門　～mbed を始めましょう!～
  http://www.eleki-jack.com/arm/2010/07/mbed-mbed.html
- <2>mbed 入門　～Hello mbed World!～
  http://www.eleki-jack.com/arm/2010/07/mbed-hello-mbed-world.html
- <3>mbed 入門　コンパイルに挑戦!　～クラウド・コンピューティングの世界～
  http://www.eleki-jack.com/arm/2010/08/mbed.html
- <4>mbed 入門　プログラムを作りましょう　～自分流にカスタマイズ!～
  http://www.eleki-jack.com/arm/2010/08/post-2.html
- <5>mbed 入門　パソコン画面に mbed!　～パソコン画面への文字出力　その 1～
  http://www.eleki-jack.com/arm/2010/08/-mbed1.html
- <6>mbed 入門　パソコン画面に mbed!　～パソコン画面への文字出力　その 2～
  http://www.eleki-jack.com/arm/2010/08/-mbed2.html

シリアル通信用ソフト（ターミナルソフト）としては、Realterm をよく使っています。ただ Realterm は Windows7-64bit に対応していないので、その場合は例えば Rs232c を勧めます。(TeraTerm でも構いませんがバイナリデータの HEX 表示に対応していないようです)
RealTerm http://realterm.sourceforge.net/
Rs232c http://www.vector.co.jp/soft/win95/hardware/se369900.html?ds
TeraTerm http://sourceforge.jp/projects/ttssh2/

<6>の段階で、シリアル通信用クラス Serial を使っています。このクラスのメンバ関数は次のように調べることができます（図 2)。
Program Workspace 内の mbed タブをクリックすると、クラスの一覧が現れます。その中の”Serial”クラスをクリックすると、そのクラスとメンバ関数の説明が出てきます。
メンバ関数をクリックするとさらにその説明が出てきます。
また mbed のハンドブックからも調べることができます。
http://mbed.org/handbook/Homepage

[課題8]　Serial クラスの baud 関数を使い、通信速度をどこまであげられるか実験する（設

定可能なシリアル通信速度の数値は、コントロールパネルで調べる)。また getc 関数を使い、ターミナルソフトからのキーボード入力(1-4)を受付け、4 つの LED を点灯／消灯させる。Switch 文を用いること。

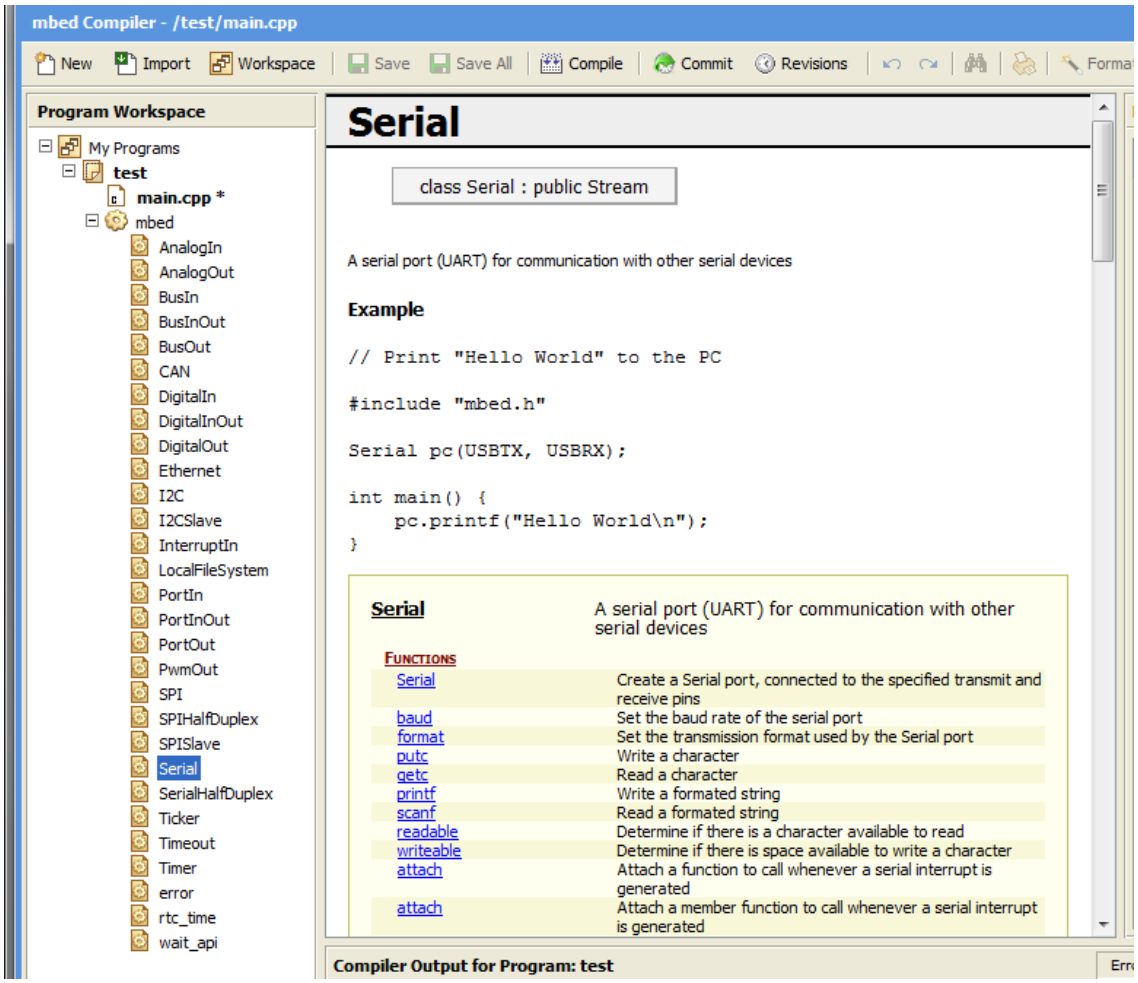


図 2　クラス Serial のメンバ関数の表示

3.2　　ブレッドボード＆加速度センサ

mbed をブレッドボードに挿し、加速度センサと接続します。

ブレッドボードとは何かについては下記を参照してください。ただし配線は今回のものと異なります。

- <7>mbed 入門　～ブレッドボードを使いましょう～（エレキジャック）
  http://www.eleki-jack.com/arm/2010/08/mbed-1.html

mbed をブレッドボードに差した様子を図 3 に示します。mbed は足が少し太いので挿すには意外に強い力が必要です。足が見えなくなるまで挿してください。抜くときはマイナス

ドライバ等が必要になります。足を曲げないよう&怪我をしないよう気をつけてください。

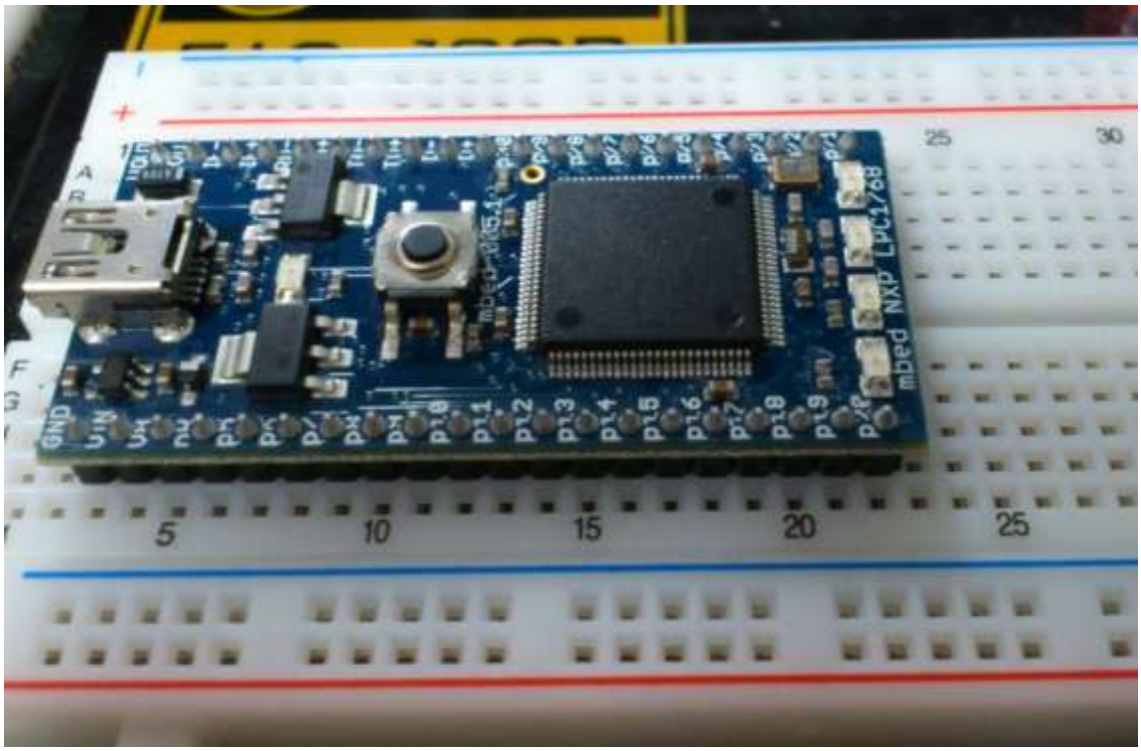


図 3　mbed を挿した様子。足が隠れるまで。

加速度センサ（と今後の追加素子）のために、mbed「から」電源を供給します。40 番ピンから 3.3V(5V ではない)を、1 番ピンから GND を取り出し、ブレッドボードの赤、青ラインに接続します（図 4、図 5）。赤、青ラインは上下にあるので赤を赤に、青を青につなぎ、今後の電源の配線を容易にします。

なおエレキジャックの例のように、mbed「に」電源を供給する場合もあります。主に USB 接続を外して独立動作させる時に使います。この場合の電源端子は異なるので注意。

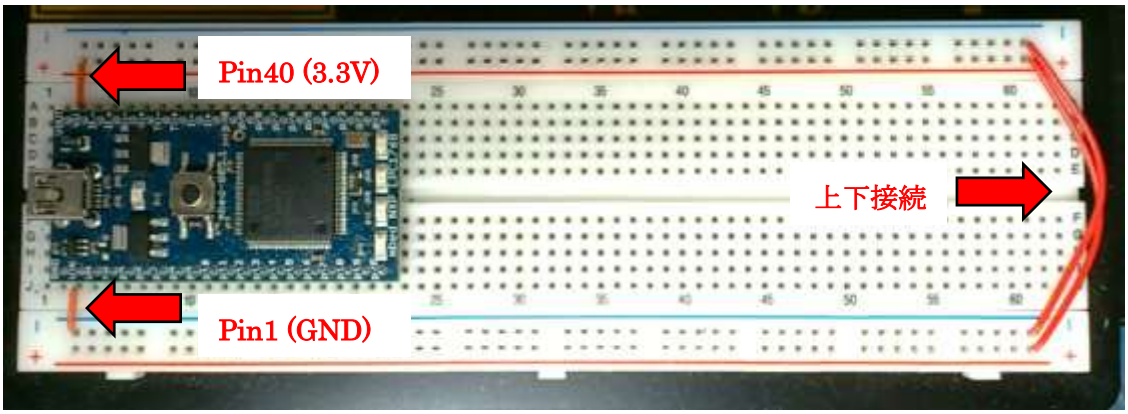


図 4　電源を配線した様子。写真は角度がついているので 2 番、39 番ピンから取り出して

いるように見えてしまっています。写真ではなく、回路図を見て配線してください。

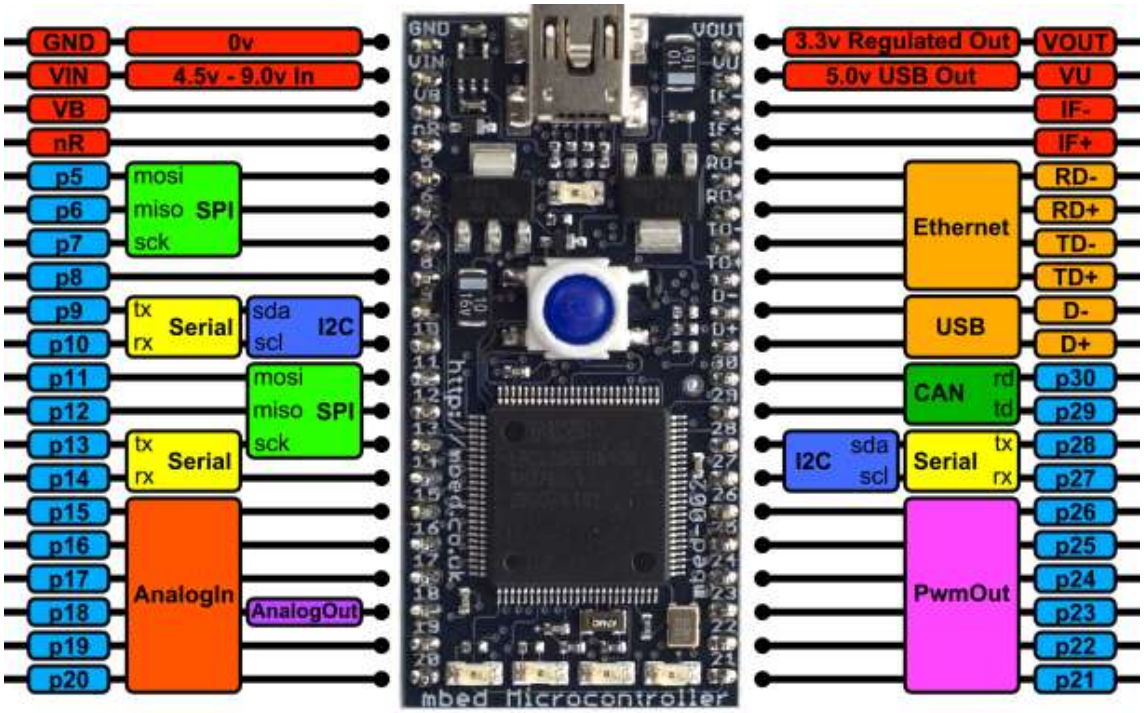


図 5　mbed ピン配置(http://mbed.org/handbook/mbed-NXP-LPC1768)

つぎに加速度センサと mbed を接続します。加速度センサは、秋月の KXM52-1050 を使います。説明書は下記にあります。

http://akizukidenshi.com/download/aki-3axis-module.pdf

説明書によると、この加速度センサは 3.3V 電源に接続した場合、オフセット 1.65V、1G 加わった際に 0.66V の変化があることがわかります。つまり重力加速度の範囲では、大体 0.99V～2.31V 出力が変化することになります。mbed の AD 変換は 0～3.3V までなので、その範囲内であることがわかります。

説明書に従い、電源、GND、PSD、パリティ、セルフテスト、の各ピンを 3.3V または GND に接続します。さらに XYZ 各軸の出力を mbed の AD 入力端子である 15～17 番ピンに接続します（図 6、図 7）。

※※※　重要な注意　※※※

今後すべての場合で、**写真ではなく回路図を見て納得しながら配線してください**。写真をコピーするといつまでも自分で回路を作れなくなります。**あえて写真とは違う場所に挿す**のが望ましいです（本当は写真も載せないほうが良いですが、初めのうちはブレッドボードの使い方が分からないかもしれないので載せています）。

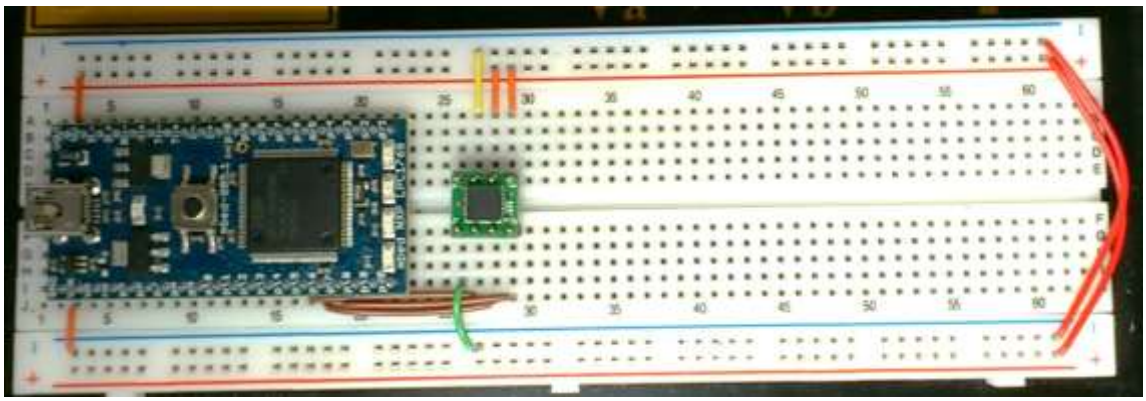


図 6　加速度センサの配置。加速度センサの向きに注意。写真ではなく、回路図を見て配線してください。

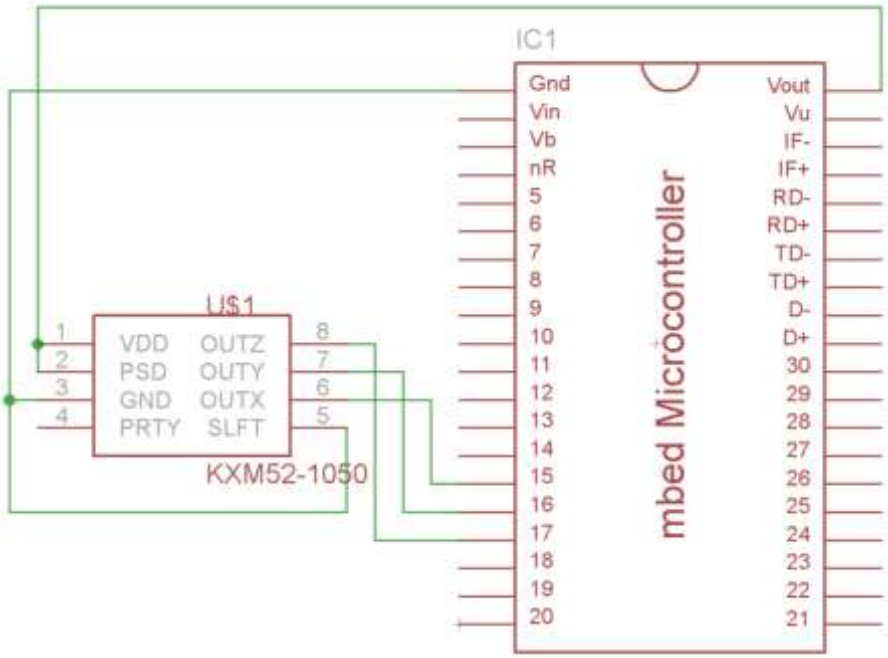


図 7　加速度センサと mbed の配線

[課題9]　以上の接続を行った上で、mbed の 15、16、17 番ピンの電圧を AD 変換で読取り、シリアル通信で出力するプログラムを書く。AnalogIn クラスおよび read 関数を用いる。データは 1/30 秒ごとに取得する。ターミナルソフトで確認する。

https://mbed.org/compiler/#AnalogIn.read

3.3　　参考資料

http://www.eleki-jack.com/arm/mbed/

サンプルソースやハードウェア等の解説が豊富に載っているため mbed を使ってどのようなことができるのかわかるためにも一度目を通すと良い.

http://jksoft.cocolog-nifty.com/blog/mbed/index.html
上のサイトと同じようにサンプルが豊富．mbed の導入についての解説もあるためまずはこちらから目を通すべきか

3.4　　シリアル通信

[課題 9] の回路を使い、mbed からの加速度センサ出力を PC 側のプログラム(DxLib プログラム中)で読み取る。

- mbed 側：図 8

X、Y、Z 軸のデータを読み出す際、文字列長さが可変だと扱いにくいため、mbed でのデータ読み出し関数を read()から read_u16()に変更する。

https://mbed.org/compiler/#AnalogIn.read_u16

- PC 側：図 8

Serial クラスを使いシリアル通信を行い、出力を読み出す。使っているのは下記のもの。

http://www.codeproject.com/KB/system/serial.aspx

”COM3”の部分の値はコントロールパネルで確認すること。COM 番号が大きい場合（8 以上？）上手く動かないことがある。その場合はコントロールパネルの方で COM 番号を設定する必要がある。

read_u16 関数によって AD 変換されたデータは非負 16bit、すなわち 2 バイトのデータとなる。これを 1 バイトごとに区切り、また PC 側では送られてきた 2x3=6 バイトのデータを元に戻している。これらはビット演算の例でもあるので猫の 40、49 章で確認する。

[課題10]　　Read_u16 関数は、AD 変換された 12bit のデータ(0xFFF=4095)を 16bit のデータ(0xFFFF=65535)にするために、65535/4095≒16.0037 倍している。これは 16 倍(=4bit シフト)とは厳密には異なる。この計算は結果として、最上位 4bit を最下位 4bit にコピーする処理と等しくなる。受信データを 16 進数で表示することでこのことを確認せよ。

```
#include "mbed.h"

Serial pc(USBTX, USBRX); // tx, rx

AnalogIn gX(p15);
AnalogIn gY(p16);
AnalogIn gZ(p17);

int main() {
    unsigned short sX,sY,sZ;

    while (1) {
        //read AD values
        sX=gX.read_u16();
        sY=gY.read_u16();
        sZ=gZ.read_u16();

        //prepare for sending. pack to 6 bytes
        pc.putc(sX>>8); //upper 8bit
        pc.putc(sX&0xFF); //lower 8bit
        pc.putc(sY>>8); //upper 8bit
        pc.putc(sY&0xFF); //lower 8bit
        pc.putc(sZ>>8); //upper 8bit
        pc.putc(sZ&0xFF); //lower 8bit

        wait(1.0); //wait for 1 sec
    }
}
```

3ch の 2byte AD 変換データを 6byte で送信している

図 8 mbed 側サンプル

```
#include "DxLib.h"
#include "Serial.h"

#define SERIAL_PORT "COM3"  //シリアルポートの番号．コントロールパネル等で確認
CSerial serial;             //シリアル通信用オブジェクト(詳しくはSerial.h参照)

int serial_init(void)
{
    //シリアルポートのオープン
    if(serial.Open(_T(SERIAL_PORT),0,0,false) != 0){
        return -1;
    }
    //通信速度を9600bpsに設定
    serial.Setup(CSerial::EBaud9600);
    //全データが揃うまで待ち続けるよう設定
    serial.SetupReadTimeouts(CSerial::EReadTimeoutBlocking);
    return 0;
}

int WINAPI WinMain( HINSTANCE hInstance, HINSTANCE hPrevInstance,
                    LPSTR lpCmdLine, int nCmdShow )
{
    int Color ;
    unsigned short sX,sY,sZ;
    unsigned char rcvBuffer[6];

    if( DxLib_Init() == -1 ){
         return -1;
    }

    if(serial_init()!=0){
        Color= GetColor( 255, 0,0) ;
        DrawFormatString( 0, 0, Color, "シリアルポートを開けません") ;
        WaitTimer(1000); //1s wait
        return -1;
    }

    while(1){
        WaitTimer(20); //20ms wait

        //read 6 bytes from serial
        serial.Read(rcvBuffer,6);

        //converts to 2 bytes data x 3
        sX=(rcvBuffer[0]<<8)|rcvBuffer[1];
        sY=(rcvBuffer[2]<<8)|rcvBuffer[3];
        sZ=(rcvBuffer[4]<<8)|rcvBuffer[5];

        //print out
        Color= GetColor( 255, 255,255) ;
        ClearDrawScreen();
        DrawFormatString(0,0,Color,"X軸:%d Y軸:%d Z軸:%d¥n¥n",sX,sY,sZ);

        //フリーズしないためのまじない。Windowsメッセージを処理
        if( ProcessMessage() == -1 ) break ;

        //ESCキーが押されたらループから抜ける
        if(CheckHitKey( KEY_INPUT_ESCAPE) == 1 ) break ;
    }

    DxLib_End() ;               // ＤＸライブラリ使用の終了処理

    return 0 ;                  // ソフトの終了
}
```

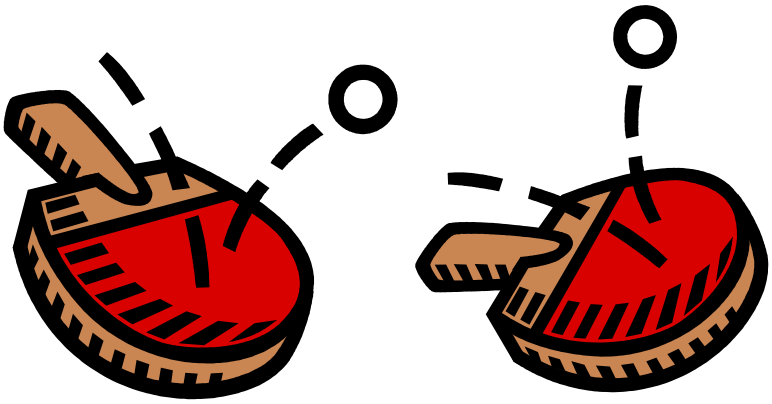
シリアルはデータがそろうまで待ち続ける設定にしている。serial_init 参照

受け取った 6byte のデータを 3 つの unsigned short 型に変換している。

図 9 DxLib 側サンプル

4　最終課題(1 ヶ月)

[課題11]　　自由落下運動するボールをリフティングしつづけるゲームを作ってください。卓球ボールをラケットでコンコンするアレです。コントローラとして mbed＋加速度センサを、描画、音響に DxLib を使います。



※※※　心得　※※※

ここで学んで欲しいのは**「作りこみと」いう体験**です。**一般に技術力は完成度への執着によって伸びます**。最低限で済ませようとせず、ぜひ執着心を持って、市販のゲームと同レベルを狙ってください。道具立てとしては十分に可能です。

ここまでの講習はほとんどテキストをなぞっただけですので、**あまり実力は付いていません**。この最終課題に**出来れば 1 ヶ月**かけ、望ましくは「寝ても覚めても」状態を体験してください。

この課題を解いていく順番は、例えば次のようになるでしょう。

(1) [課題 7] を改変し、跳ね返る地面を動かせるようにする。
(2) 加速度センサの値を地面の傾きに反映させる。X、Y の値に応じて地面が傾くようにする。
(3) 地面が傾いた時のボールの挙動をシミュレートする。地面とボールの反射は物理学の基本ですし、検索すればたくさん出てきます。
(4) 加速度センサの値を地面の並進運動に反映させる。Z の値に応じて動くには加速度を 2 回積分する必要。しかしそれだけでは、傾けただけでも Z の値は変わってしまうはず。どうするか。
(5) 地面の速度によって、ボールの跳ね返り量は変化するはず。どうするか。
(6) 地面を「ラケット」に置き換える。見た目の良いラケットを 3D で作成する。ラケットにボールがぶつかるかどうかの判定はどうするか。

(7) さらに全体をより美しく、楽しくしていく。市販のゲームレベルの完成度を目指す。スコアの表示、落ちてくるものをゴムボール等に変える、ラケットにボールの「影」を映す、落ちてくるボールの個数を複数にしてみる、ボールの回転、ビジュアルエフェクト、サウンドエフェクト、etc…