物づくり授業

作成者：松下光次郎

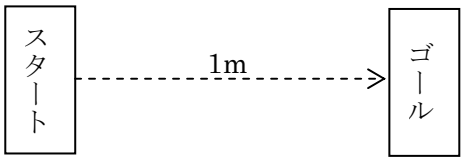
1　ものづくり授業の概要

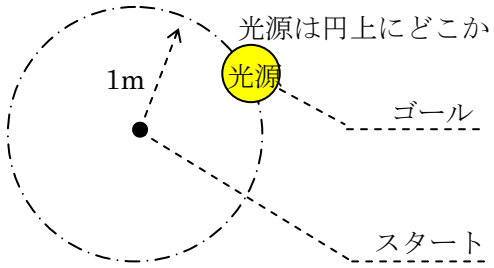
趣旨

１．ロボットの制御プログラムを作成 マイクロチップ制御器におけるＲＣサーボ･光センサ･LED の使い方を理解する（C 言語）.
２．ロボットの機構を製作 家庭用品などを用いて簡単な移動ロボットを製作し，二種類のコンテストを行う．ロボットを設計する上で，如何に身体性が重要かを理解する

コンテスト

コンテスト１：　1m 短距離レース
RC サーボ 1 個または 2 個を使って移動ロボットを製作し，その前進速度を競う．受動機構などの身体性を考慮し，効率の良い移動機構を探求する．
コンテスト２：　光源探索レース
短距離レースを行った移動ロボットに光センサを取り付け，光源探索レースを行う．モータとセンサの組み合わせにより，移動機能を拡張する．





スケジュール

ものづくり授業　～移動ロボットコンテスト～		
7 月 1 日	10:15～10:45	制御器の説明
	10:45～11:45	物づくり時間
7 月 1 日～7 月 7 日（1 週間）		各自，物づくり時間
7 月 8 日	10:15～10:45	ものづくり時間＆準備
	10:45～11:45	コンテスト

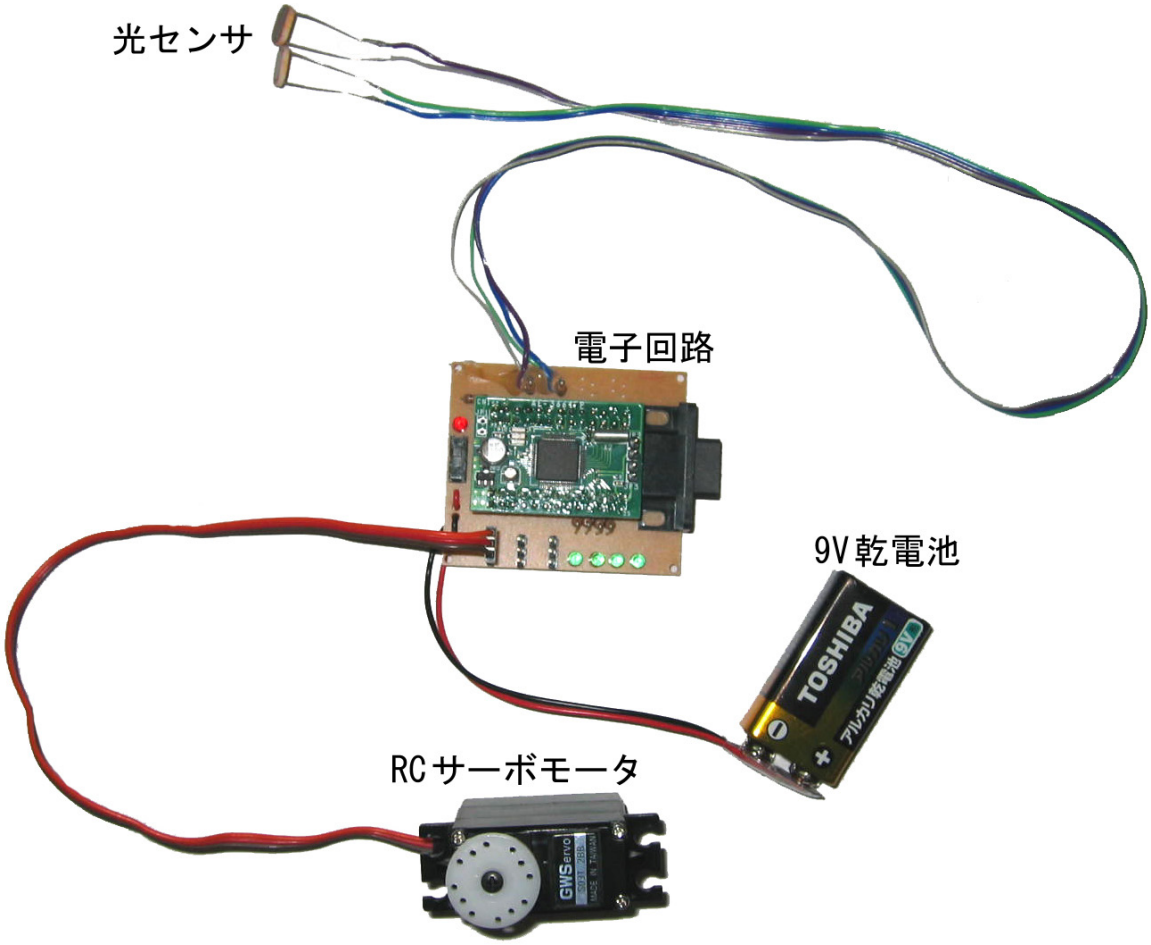
各自，持ってくるもの

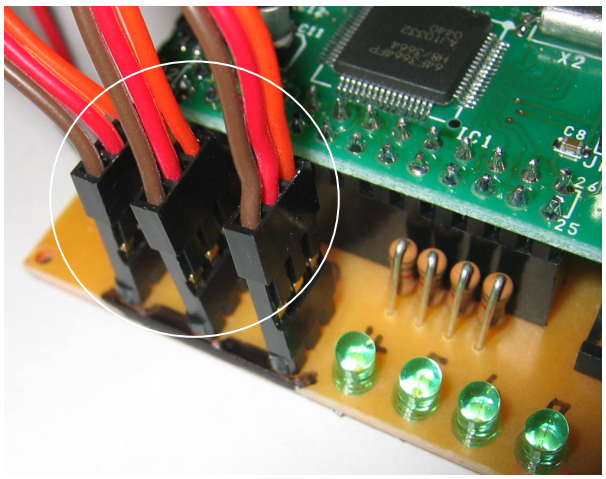
各グループ持ってくるもの（最大 20 グループ）
制御プログラムの書き込み用 ・ノートコンピュータ（シリアルポート付） ロボット製作用 ・家庭用品（ダンボール，ペットボトル，プラスチック製品など） ・道具（はさみ，カッターナイフ，グルーガン，半田ごてなど）

2　制御器概要

2.1　制御器の接続図

接続は以下の写真を参考にしてください．特に，サーボモータのコネクタの向きには気をつける必要があります．

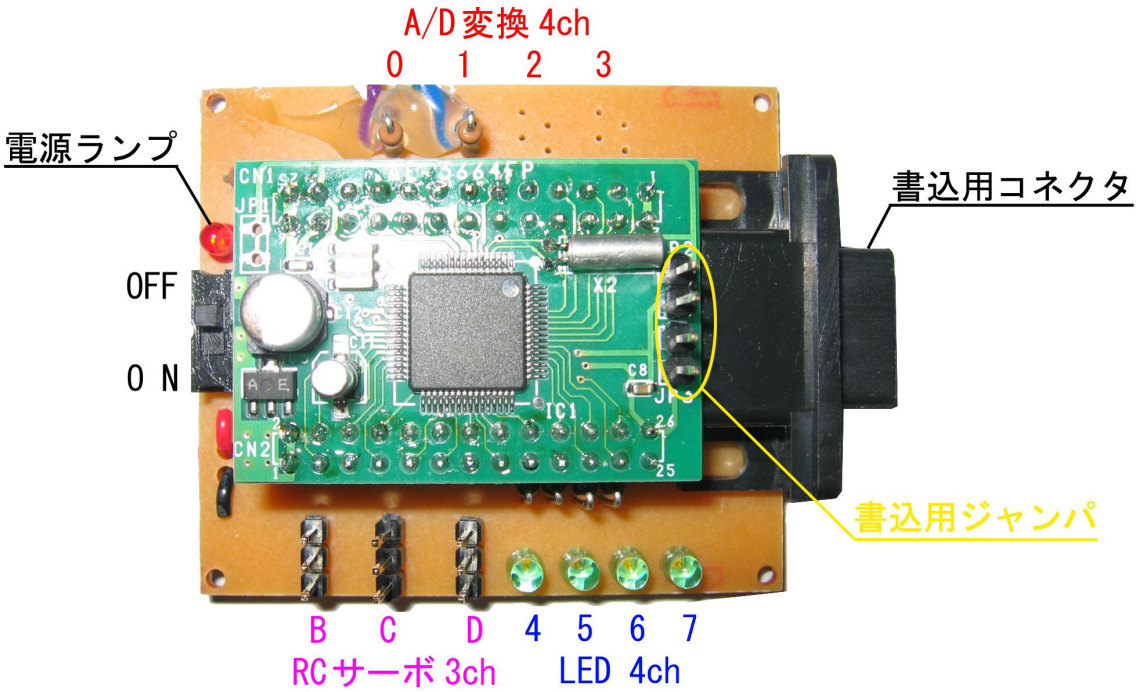




RC サーボを接続するときはコネクタの向きに気をつけること！
茶色の Ground 線が電子回路の外側に！

2.2　電子回路

電子回路は，RC サーボモータ３個を制御でき，光センサ２個の状態を監視でき，LED４個の点灯・消灯を制御できる制御器となっている．マイクロチップには秋葉原にある秋月電商にて購入した H8/3664 を使用している．シリアル通信によりプログラムを書き込むシステムとなっている．



電子回路ポート	内容
PWM 制御信号 3ch	RC サーボモータ制御
A/D 変換入力 2ch	光センサ状態監視
デジタル出力 4ch	LED 点灯・消灯

2.3　初期インストール済プログラム

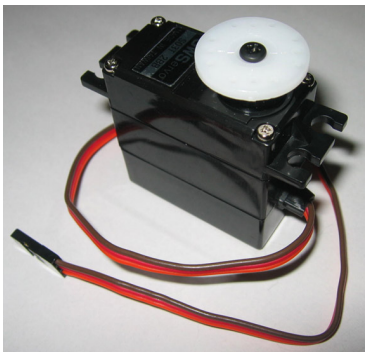
初期プログラムは以下のように動作しているので，授業中にパソコンによるプログラム作成を行わなくとも，制御器を利用してロボット作成を行っても良い．

光センサと LED	
光源探索 モード	光センサ 0ch, 1ch 共に受光している場合，光源探索モードであり，光源に近い方の LED が点灯する．光源がない場合，中央の LED２個が点灯する．
光の強弱表示 モード	光センサ 1ch を黒テープなどで被っている場合， 光センサ 0ch が感知する光の強弱により LED が点灯する個数が決まる．

RC サーボモータ	
B ch	周期 100 ミリ秒，振幅 60°
C ch	周期 500 ミリ秒，振幅 60°
D ch	周期 1 秒，振幅 60°

2.4　RC サーボモータ

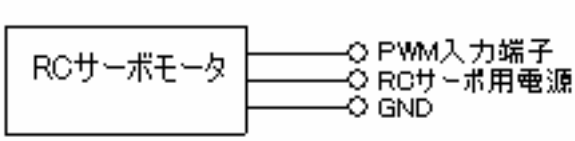
ラジコン用サーボモータ（RC サーボ）は、DC モータ 、減速機構（ギヤボックス)、サーボアンプ（モータを駆動するための回路）が一体になっていて簡単に扱うことのできるアクチュエータといえます。RC サーボには、アナログサーボ（プラスチックギア・メタルギア)・デジタルサーボなどがあります。デジタルサーボはアナログサーボに比べて出力トルクも高く。ラジコンショップや模型店での入手が可能であり、価格も安く、制御が簡単でとても使いやすいため、様々なロボットで RC サーボが使用されています。



標準型 RC サーボ GWS S03T/2B			
性能		トルク	速度
	4.8V	7.2kg-cm	0.33sec/60°
	6.0V	8.0kg-cm	0.27sec/60°
重量	46 g		
寸法	39.5 × 20.0 × 39.6mm		

図　RC サーボモータ

RC サーボは、図のようにケーブル 3 本とコネクタがついています。このケーブル 3 本を制御信号・電源電圧・グラウンド接続して初めて動かすことができます。制御信号とは幅の違う矩形波（PWM）のことであり、幅によりモータの位置を制御しています。PWM については、次の節で説明します。



橙色:　PWM 入力端子
赤色:　RC サーボ用電源
茶色:　グラウンド

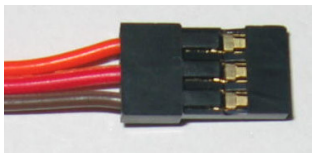
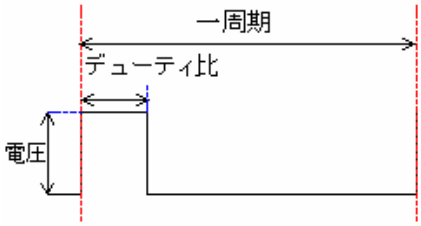


図　RC サーボのコネクタ

2.5　RC サーボモータのモータの制御方法 - PWM（パルス幅変調）

パルス変調幅(PWM)は簡単にいうと、高速でモータ電源の ON/OFF を繰り返すことです。この ON/OFF の繰り返しにはルールがあり、ある一定時間での ON にする時間と OFF にする時間の比を変えてあげるのです。この比をデューティ比といいます。デューティ比が高いほどＯＮになっている期間が長く、デューティ比がゼロではずっとＯＦＦのままである様子を示しています。ここで重要なのは、先ほど述べた“ある一定時間”、つまり、パルスを与える周期です。何秒おきに ON/OFF を繰り返すか最適に設定しなければなりません。



パルス変調幅	
周期	20msec など。
デューティ比	一周期中の High の占める割合 0～100%。
電圧	一般的に 5V。

RC サーボは、マイクロチップから送られる PWM 制御信号のパルス幅を読み取り、モータを決められた角度に動かします。つまり、デューティ比がモータの角度に対応しているのです。一般の RC サーボモータにおいて、パルス：３～５V、周期：20ms、幅：0.9ms～2.1ms(中心 1.5ms)です。RC サーボは一般的に 180° 回転が限界ですので、0.1ms のパルス幅変化が 15° の角度変化に対応しています。

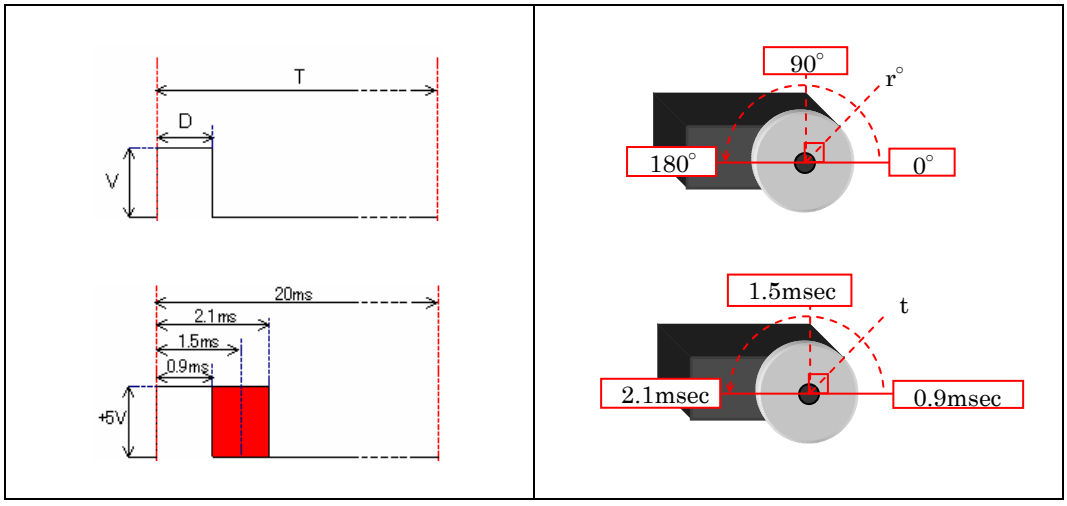
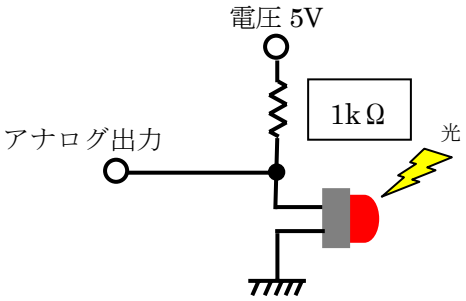


表　RC サーボモータの制御信号

PWM 制御信号				
周期 T	20msec(50Hz) PWM レジスタ GRA = 40000 (16MHz8 周分)			
制御		始点	中央	終点
	デューティ比 d	4.5%	7.5%	10.5%
	時間 t	0.9ms	1.5ms	2.1ms
	角度 r	0°	90°	180°
	PWM レジスタ GRB, GRC, GRD	1200	2600	4000

2.6　光センサ（CdS センサ）

光センサは、光の強弱、物体の有無や距離を検出するセンサとして使用されています。光の強弱を測る場合、受光素子は光が当たると電流が流れるようになる素子なので、光の強弱がアナログ量に比例して出力されます。物体の有無や、物体の距離を検出する場合、このセンサは発光素子と受光素子で構成され、発光素子から照射した光が物体に当たって反射することで反射量に比例した電流が流れて物体の距離を検出することになります。発光素子には赤外線 LED がよく利用されます。受光素子には、フォトトランジスタやフォトダイオード、CdS などが利用されています。



状態	Cds センサ抵抗値 [Ω]	アナログ出力値 [V]	8bit 値 (0-255) x0, x1, x2, x3
光源に向いている状態	0 – 400	0.00 – 1.45	0 – 74
室内環境	400 – 1000	1.45 – 2.50	74 – 128
やや暗い環境	1k – 5k	2.50 – 4.20	128 – 214
真っ暗な環境	5k - ∞	4.20 – 5.00	214 – 255

2　Ｈ８プログラム書き込み環境準備

2.1　必要ファイルの入手

圧縮ファイル「monotuskuri.exe」を手に入れ，解凍してください．このファイルは，自動解凍機能付なので，パスワードを入力すれば解凍できます．解凍すると，２個のフォルダを確認することができます．１つは，USB シリアルケーブルのドライバ．もう１つは，制御器にプログラムを書き込むためのコンパイラおよび書き込みソフトセットです．

ダウンロード：http://www.koj-m.sakura.ne.jp/edutainment/part1/monotsukuri.zip

圧縮ファイル	パスワード
monotsukuri.exe	

ファイル名	内容
arvel_usbtoserial_driver	USB シリアルケーブル用ドライバ
h8	H8/3664 用コンパイラおよび書込ソフト．C ドライブに置く．

2.2　USB シリアルケーブルのインストール

最近のノートパソコンには，図のようなシリアル通信用ポート［Dsub９ピン（オス)］が使用されなくなってきています．ノートパソコンに使用されてない場合は，USB シリアルケーブルを使用してもらうこととなります. このケーブルは USB を利用して仮想的にシリアル通信用ポートとする変換器となっています.

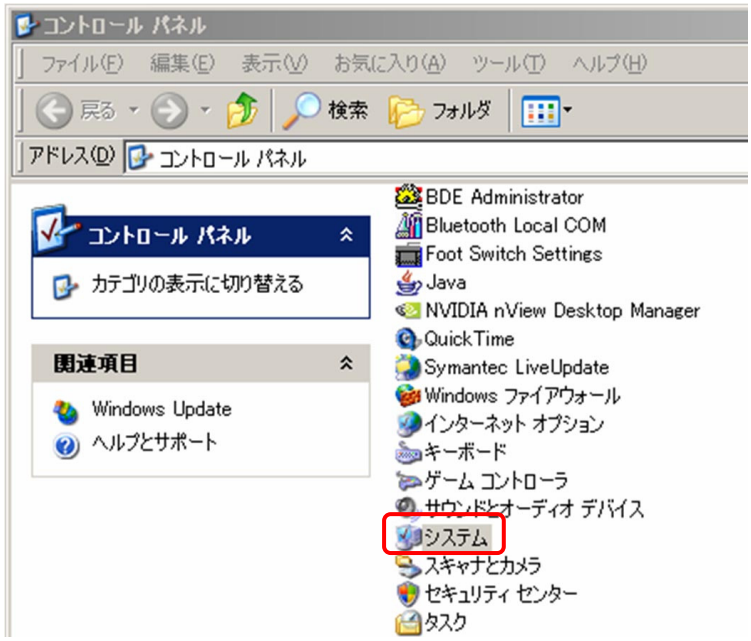
ドライバのインストールには，先に解凍した「arvel_usbtoserial_driver」フォルダを指定して行ってください．WindowsXP に認証されていないと表示されますが，問題なくインストールできます.

シリアル通信用ポート
Dsub9 ピン（オス）

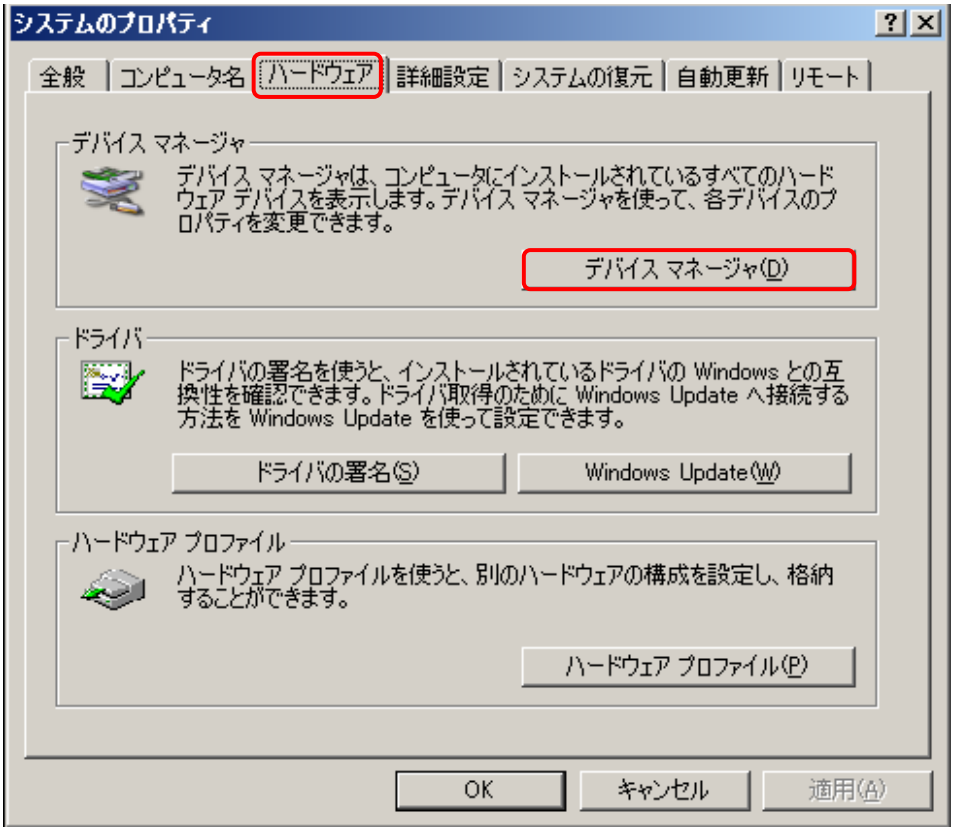
USB シリアルケーブル

次に，USB シリアル通信用ケーブルを「COM1」に設定する必要があります．COM1 とは，第一番目のシリアル通信用ポートに指定するということです. どのメーカの USB シリアル通信用ケーブルを使用する場合でもこの手順を行い，「COM1」であるかどうか確認してください.

まず,「コントロールパネル」を開き,「システム」のアイコンをダブルクリックしてください.

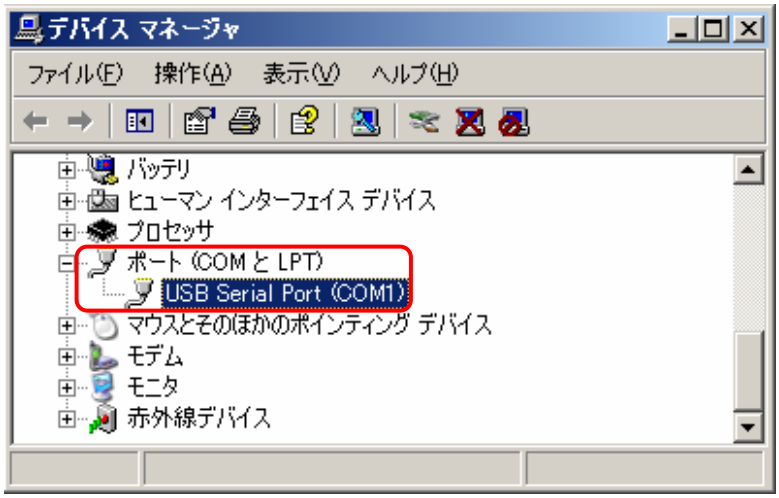


「システムのプロパティ」ウィンドウのタブである「ハードウェア」を選択肢,「デバイスマネージャ」を指定してください.

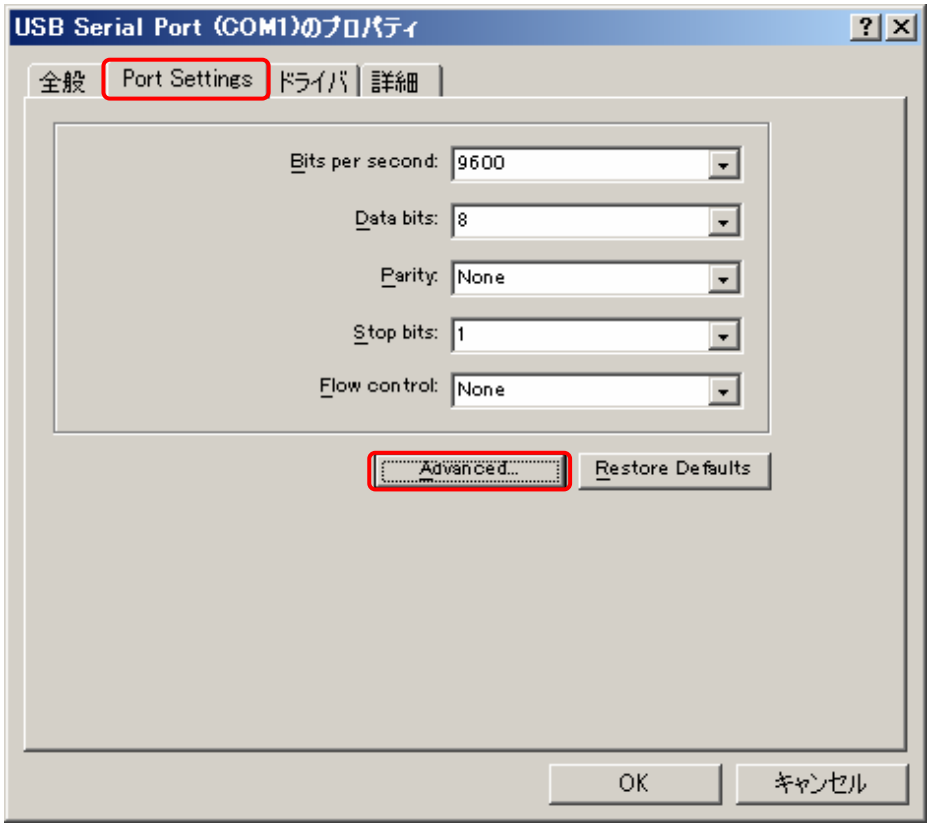


「デバイスマネージャ」のツリー構造中にある「ポート（COM と LPT)」を選択すると，「USB Serial Port (COM1)」と確認することができます．ここで「COM1」と表示されている場合は，問題ないのでこの手順を終了してもらって問題ありません．

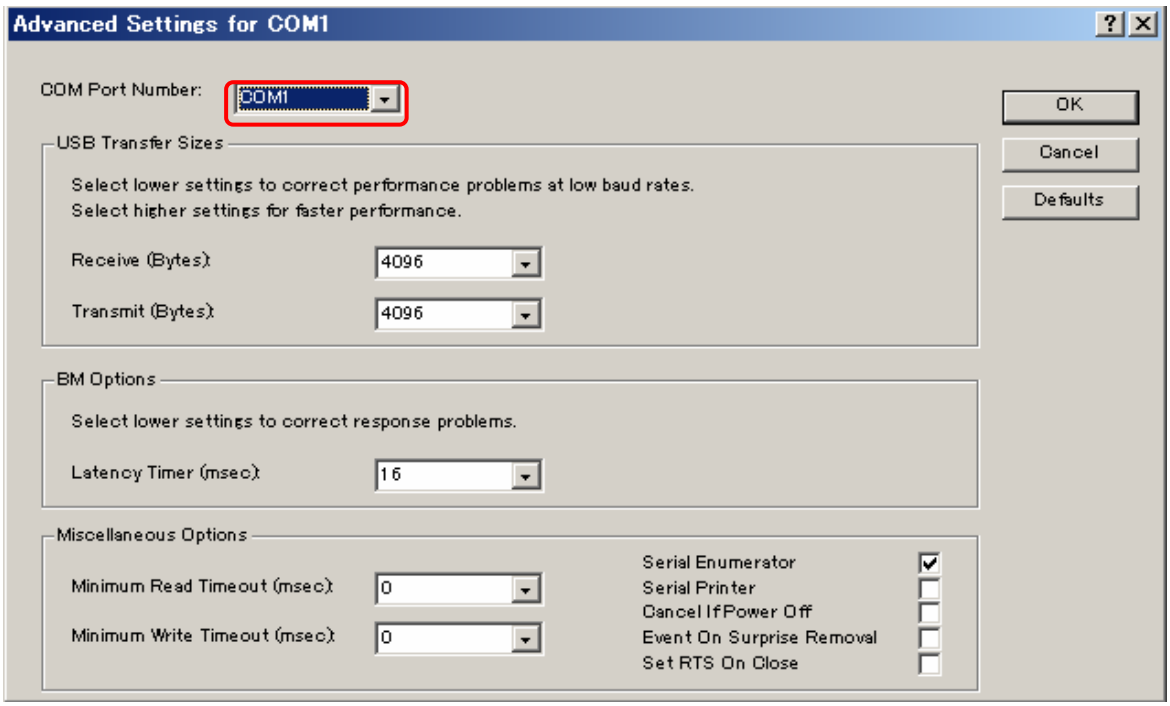
ここで「COM2」や「COM5」と表示されている場合は，「COM1」に直す必要があります．「USB Serial Port (COM2)」をダブルクリックしてください．



「USB Serial Port (COM2)のプロパティ」ウィンドウを開いき，「Port Settings」のタブを選択し，「Advanced…」をクリックしてください．

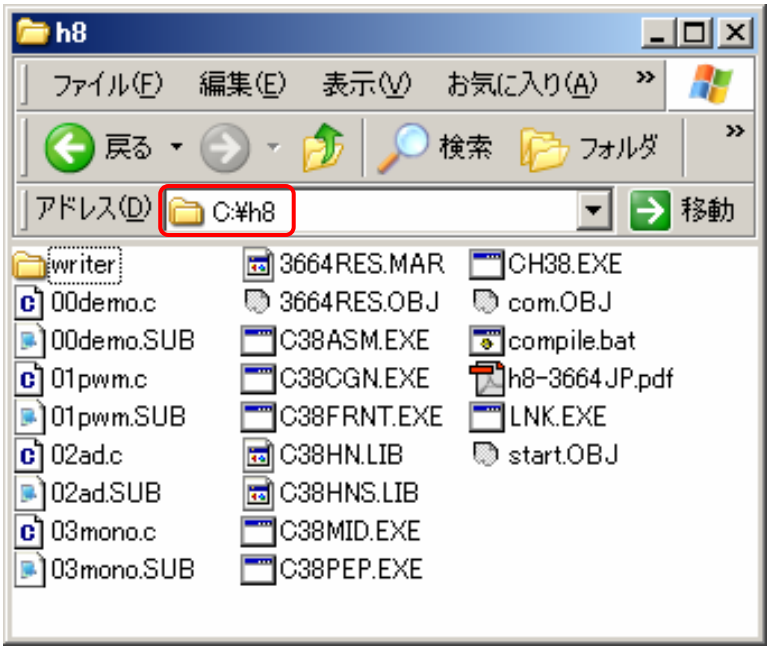


「Advanced Settings for COM2」ウィンドウの最上段に「COM Port Number:」という項目があります．ここの項目で「COM1」をして，「OK」ボタンを押し，この手順を終了してください．



2.3　Ｈ８プログラム書き込み環境準備

先ほど，解凍した「h8」フォルダを「C ドライブ」におき，「h8」のフォルダの内容を見た時そのアドレスバーの表示が「C:¥h8」と表示されているか確認してください．



3　H８プログラム作成方法

3.1　サンプルプログラム概要

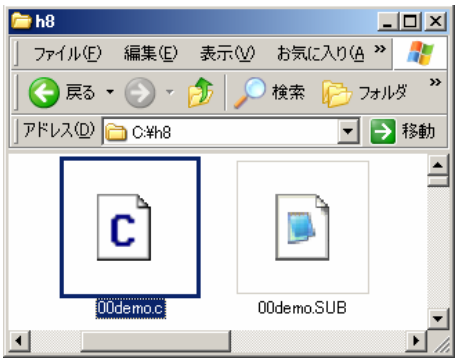
授業では，３個のサンプルファイルを用意しています．H8 のプログラムは C ファイルと Sub ファイルが必要となっており，下表に示すとおりサンプルプログラムにおいて，C と Sub ファイルのセットとなっております．

* 付録にサンプルプログラムがあり，色付部のみを変更することで利用できます．

	ファイル	内容
サンプル１	01pwm.c 01pwm.sub	RC サーボモータ振動制御
サンプル２	02ad.c 02ad.sub	光センサ読取および LED 点灯
サンプル３	03mono.c 03mono.sub	サンプル１＆２の統合プログラム

3.2　新しい H8 プログラムの作成方法手順

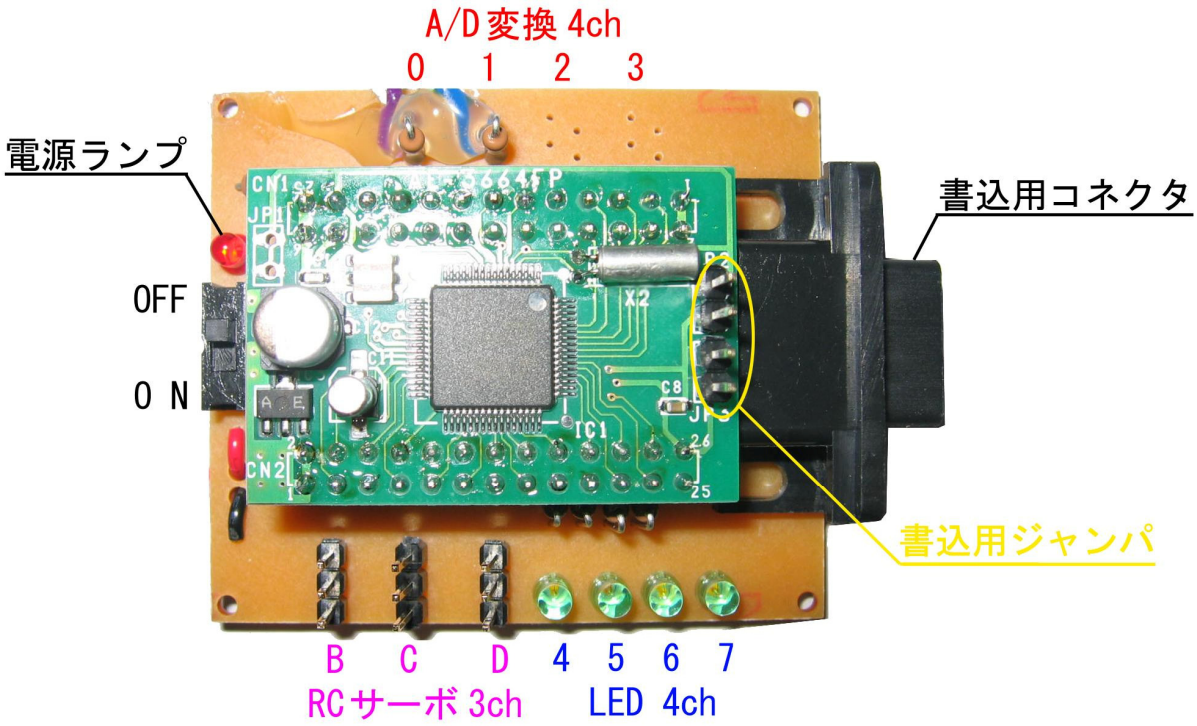
新しいプログラムを作成する場合は，「C:¥h8」フォルダ内でいずれかのサンプルの C ファイルと Sub ファイルをコピーし，それらの名前を変更し，且つ，Sub ファイル内のファイル名部分を変更してください．その後，C ファイルにプログラムを書き込んでください．C ファイル名，Sub ファイル名，Sub ファイル内容のファイル名部分（色付部）は，必ず統一してください．



ファイル名	00demo.sub
OUTPUT 00demo PRINT 00demo INPUT 3664RES, 00demo LIB c38hn START P(100) EXIT	

3.3　電子回路と H8 プログラム記述方式

電子回路の各ポートの H8 プログラムにおける記述方式を下表に示します.



電子回路		プログラム	
RC サーボ 3ch	B	TW.GRB	RC サーボの軸位置 左端 ← 中央 → 右端 1200　2600　3800
	C	TW.GRC	
	D	TW.GRD	
A/D 変換入力 2ch	0	x0	光の強弱 明るい(0) ←→ 暗い(255)
	1	x1	
LED 4ch	7	IO.PDR1.BIT.B7	LED 点灯・消灯 点灯：IO.PDR1.BIT.B7 = 1 消灯：IO.PDR1.BIT.B7 = 0
	6	IO.PDR1.BIT.B6	
	5	IO.PDR1.BIT.B5	
	4	IO.PDR1.BIT.B4	

3.4　H8 における C プログラミングの注意事項

- コメントは「// 」ではなく，「/* */.」で記入すること.
- 二つの条件を含む if 文の場合，「if(x0<10 && x0>50)」とはせず，二つの条件必ず括弧で囲「if((x0<10)&&(x0>50))」とすること
- このコンパイラは，「if(x0=20)」の記述間違いにおいてエラーが表示されないので，気をつけること.（if 文が「if(x0==20)」というように「==」で記述されているのを確認すること.）

3.5　付録：サンプルプログラム

ファイル名	01pwm.c
内容	１秒ごとに，３個のＲＣサーボが左右交互に振動する．

```
#include <3664f.h>
void main(void){
        int t=0;          /* Time           */
        int flag=0;       /* Right/Left     */

        IO.PCR8 = 0xff;
        TW.GRA = 40000;             /* Freq:20msec     */
        TW.TMRW.BYTE = 0xcf;        /* TimerWsetting   */
        TW.TCRW.BYTE = 0xbf;        /* TimerWsetting   */

        TW.GRB = 2600;              /* モータBch位置：中央 */
        TW.GRC = 2600;              /* モータCch位置：中央*/
        TW.GRD = 2600;              /* モータDch位置：中央*/

        while(1){
                if(TW.TSRW.BIT.IMFA==1){    /* GRA:20ミリ秒間隔で実行 */
                        TW.TSRW.BIT.IMFA=0;
                        if(t>50){/* 20msec*50=1000ミリ秒間隔で実行 */
                                /* フラッグの状態により交互に位置が変わる */
                                if(flag==0){
                                        TW.GRB = 2400;     /* モータBch位置：左端 */
                                        TW.GRC = 2200;     /* モータCch位置：左端 */
                                        TW.GRD = 2000;     /* モータDch位置：左端 */
                                        flag=1;
                                }
                                else {
                                        TW.GRB = 2800;     /* モータBch位置：右端 */
                                        TW.GRC = 3000;     /* モータCch位置：右端 */
                                        TW.GRD = 3200;     /* モータDch位置：右端 */
                                        flag=0;
                                }
                                t=0;
                        }
                        t++;
                }
        }
}
```

サーボモータの制御に関してのプログラム．GRB, GRC, GRD の値が各サーボモータの位置に対応しており．フラッグ「flag」の状態により，１秒ごとにモータの位置を切り替えている．

ファイル名	02ad.c
内容	20m 秒ごとに２個の光センサ状態を読み取り，それら値の比較して LED の点灯状態を決定する.

```
#include <3664f.h>
#include "myType.h"
void main(void){
        unsigned int* data;
        UInt8 x0, x1, x2, x3;       /* AD:0ch-3ch      */
        IO.PMR1.BYTE = 0x00;        /* Port1:DIO       */
        IO.PCR1 = 0xff;             /* Port1:Output    */
        IO.PCR8 = 0xff;
        TW.GRA = 40000;             /* Freq:20msec     */
        TW.TMRW.BYTE = 0xcf;        /* TimerWsetting   */
        TW.TCRW.BYTE = 0xbf;        /* TimerWsetting   */

        IO.PDR1.BIT.B7=0;  /* LED1：消灯      */
        IO.PDR1.BIT.B6=1;  /* LED2：点灯      */
        IO.PDR1.BIT.B5=0;  /* LED3：消灯      */
        IO.PDR1.BIT.B4=1;  /* LED4：点灯      */
        while(1){
                if(TW.TSRW.BIT.IMFA==1){    /* 20ミリ秒間隔で実行 */
                        TW.TSRW.BIT.IMFA=0;
                        /* AD4chの読取 */
                        AD.CSR.BIT.ADST=0;
                        AD.CSR.BIT.SCAN=1;
                        AD.CSR.BIT.CKS=1;
                        AD.CSR.BIT.CH=3;
                        AD.CSR.BIT.ADST=1;
                        while(AD.CSR.BIT.ADF==0){};
                        AD.CSR.BIT.ADF=0;
                        AD.CSR.BIT.ADST=0
                        data=&AD.DRA; x0=(int)(*data>>8);
                        data=&AD.DRB; x1=(int)(*data>>8);
                        data=&AD.DRC; x2=(int)(*data>>8);
                        data=&AD.DRD; x3=(int)(*data>>8);
                        /* x0-x3は 0-255(8bit)で表現される  */
                        if(x0<x1){         /* 0ch,1chの比較値によるLED点灯条件分岐 */
                                IO.PDR1.BIT.B7=1; /* 点灯 */
                                IO.PDR1.BIT.B6=1; /* 点灯 */
                                IO.PDR1.BIT.B5=0; /* 消灯 */
                                IO.PDR1.BIT.B4=0; /* 消灯 */
                        }
                        else {
                                IO.PDR1.BIT.B7=0; /* 消灯 */
                                IO.PDR1.BIT.B6=0; /* 消灯 */
                                IO.PDR1.BIT.B5=1; /* 点灯 */
                                IO.PDR1.BIT.B4=1; /* 点灯 */
                        }
                }
        }
}
```

AD4ch を読み取るプログラムだが，実際に使用している値は光センサ 0ch,1ch に対応する AD0ch,AD1ch である．AD 入力で監視される電圧は 8bit(0-255)で表現される．このプログラムは，20 ミリごとにそれら 2ch の電圧値の比較を行い，LED の点灯状態を変化させている.

ファイル名	03mono.c（初期インストール済プログラム）
内容	「01pwm.c」と「02ad.c」の統合ファイルである．20m 秒ごとに２個の光センサ状態を読み取り,それら値の比較して LED の点灯状態を決定する．また各３個のＲＣサーボが，各周期で左右交互に振動する．

```
#include <3664f.h>
#include "myType.h"
void main(void){
        int tb=0;           /* time for pwm b ch        */
        int tc=0;           /* time for pwm c ch        */
        int td=0;           /* time for pwm d ch        */
        int flagb=0;        /* flag for pwm b ch        */
        int flagc=0;        /* flag for pwm c ch        */
        int flagd=0;        /* flag for pwm d ch        */
        int TIMEB=5;        /* freq for pwm b ch        */
        int TIMEC=25;       /* freq for pwm c ch        */
        int TIMED=50;       /* freq for pwm d ch        */
        /* TIME - 100:2sec, 50:1sec, 25:0.5sec, 5,0.1sec */
        int CENTER=2600;    /* RCサーボ中央位置         */
        int GAPB=120;       /* Bchの振幅量              */
        int GAPC=150;       /* Cchの振幅量              */
        int GAPD=600;       /* Dchの振幅量              */
        unsigned int* data;
        UInt8 x0, x1, x2, x3;       /* AD:0ch-3ch*/

        IO.PMR1.BYTE = 0x00;        /* Port1:DIO       */
        IO.PCR1 = 0xff;             /* Port1:Output    */
        IO.PCR8 = 0xff;
        TW.GRA = 40000;             /* Freq:20msec     */
        TW.TMRW.BYTE = 0xcf;        /* TimerW setting  */
        TW.TCRW.BYTE = 0xbf;        /* TimerW setting  */
        TW.GRB = CENTER;            /* モータBch位置：中央 */
        TW.GRC = CENTER;            /* モータCch位置：中央  */
        TW.GRD = CENTER;            /* モータDch位置：中央  */
        IO.PDR1.BIT.B7=0;           /* LED 7ch */
        IO.PDR1.BIT.B6=1;           /* LED 6ch */
        IO.PDR1.BIT.B5=0;           /* LED 5ch */
        IO.PDR1.BIT.B4=1;           /* LED 4ch */

        while(1){
                /* GRA:20msec間隔で実行 */
                if(TW.TSRW.BIT.IMFA==1){
                        TW.TSRW.BIT.IMFA=0;

                        /* 光センサ2ch読取：AD4ch読取内2ch使用 */
                        AD.CSR.BIT.ADST=0;
                        AD.CSR.BIT.SCAN=1;
                        AD.CSR.BIT.CKS=1;
                        AD.CSR.BIT.CH=3;
                        AD.CSR.BIT.ADST=1;
                        while(AD.CSR.BIT.ADF==0){};
                        AD.CSR.BIT.ADF=0;
                        AD.CSR.BIT.ADST=0;
                        data=&AD.DRA; x0=(int)(*data>>8);    /* アナログ入力0ch */
                        data=&AD.DRB; x1=(int)(*data>>8);    /* アナログ入力1ch */
                        data=&AD.DRC; x2=(int)(*data>>8);    /* アナログ入力2ch */
                        data=&AD.DRD; x3=(int)(*data>>8);    /* アナログ入力3ch */
```

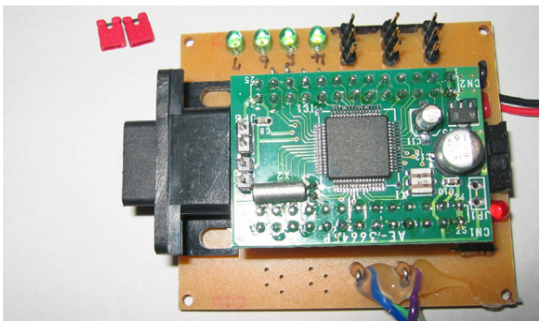
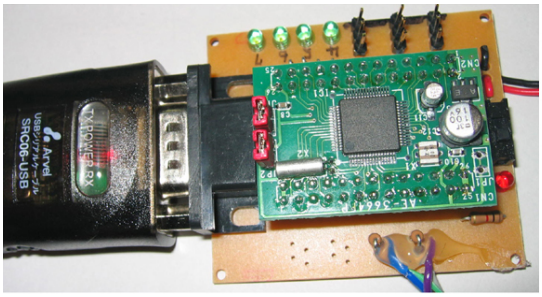
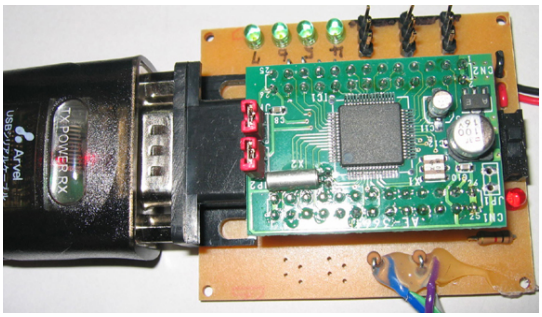
```
if(x1>214){         /* AD1chが光を検知しない場合 */
        /* AD0chの光の強弱によりLED点灯条件が決定される */
        /* 光の強弱は，明るい(0)←→暗い(255)
        if((x0>=0)&&(x0<51)){                   /* 非常に明るい    */
                IO.PDR1.BIT.B7=1;
                IO.PDR1.BIT.B6=1;
                IO.PDR1.BIT.B5=1;
                IO.PDR1.BIT.B4=1;
        }
        else if((x0>=51)&&(x0<102)){            /* やや明るい      */
                IO.PDR1.BIT.B7=0;
                IO.PDR1.BIT.B6=1;
                IO.PDR1.BIT.B5=1;
                IO.PDR1.BIT.B4=1;
        }
        else if((x0>=102)&&(x0<153)){           /* やや暗い        */
                IO.PDR1.BIT.B7=0;
                IO.PDR1.BIT.B6=0;
                IO.PDR1.BIT.B5=1;
                IO.PDR1.BIT.B4=1;
        }
        else if((x0>=153)&&(x0<204)){           /* 非常に暗い      */
                IO.PDR1.BIT.B7=0;
                IO.PDR1.BIT.B6=0;
                IO.PDR1.BIT.B5=0;
                IO.PDR1.BIT.B4=1;
        }
        else if((x0>=204)&&(x0<256){            /* 光を完全遮断    */
                IO.PDR1.BIT.B7=0;
                IO.PDR1.BIT.B6=0;
                IO.PDR1.BIT.B5=0;
                IO.PDR1.BIT.B4=0;
        }
}
else {  /* AD1chが光を検知する場合 */
        /* 2個の光センサのどちらかが光源を感知した場合            */
        /* 対応した左右どちらか二個のLEDが点灯                    */
        if((x0<80)||(x1<80)){           /* 光源を感知した場合      */
                if(x0<x1){              /* x1が光源を感知した場合  */
                        IO.PDR1.BIT.B7=1;
                        IO.PDR1.BIT.B6=1;
                        IO.PDR1.BIT.B5=0;
                        IO.PDR1.BIT.B4=0;
                }
                else {                  /* x0が光源を感知した場合  */
                        IO.PDR1.BIT.B7=0;
                        IO.PDR1.BIT.B6=0;
                        IO.PDR1.BIT.B5=1;
                        IO.PDR1.BIT.B4=1;
                }
        }
        else {  /* どちらも光源を感知しない場合            */
                /* 真ん中二個のLED点灯                    */
                IO.PDR1.BIT.B7=0;
                IO.PDR1.BIT.B6=1;
                IO.PDR1.BIT.B5=1;
                IO.PDR1.BIT.B4=0;
        }
}
```

```
                        /* サーボモータ3chの振動設定          */

                        /* サーボモータBch振動設定  */
                        if(tb>TIMEB){      /* 周期：20msec * TIMEB */
                                if(flagb==0){
                                        TW.GRB = CENTER - GAPB;      /* Bchの左端位置 */
                                        flagb=1;
                                }
                                else {
                                        TW.GRB CENTER - GAPB;        /* Bchの右端位置 */
                                        flagb=0;
                                }
                                tb=0;
                        }
                        tb++;

                        /* サーボモータCch振動設定 */
                        if(tc>TIMEC){      /*周期：20msec * TIMEC */
                                if(flagc==0){
                                        TW.GRC = CENTER - GAPC;      /* Cchの左端位置 */
                                        flagc=1;
                                }
                                else {
                                        TW.GRC = CENTER - GAPC;      /* Cchの右端位置 */
                                        flagc=0;
                                }
                                tc=0;
                        }
                        tc++;

                        /* サーボモータDch振動設定 */
                        if(td>TIMED){      /*周期：20msec * TIMED */
                                if(flagd==0){
                                        TW.GRD = CENTER - GAPD;      /* Dchの左端位置 */
                                        flagd=1;
                                }
                                else {
                                        TW.GRD = CENTER - GAPD;      /* Dchの右端位置 */
                                        flagd=0;
                                }
                                td=0;
                        }
                        td++;

                }
        }
}
```

AD1ch の光センサ値の判別で，光センサは二つのモードに切り替わる．1ch の光センサの受光部を黒いテープなどで被い，光を完全に遮断すると「光の強弱表示モード」になり，受光部を覆わない状態は「光源探索モード」となる．また，各サーボモータは各々振幅値・周期値を設定できるようになっている．

4　Ｈ８プログラムのコンパイルおよび書き込み方法

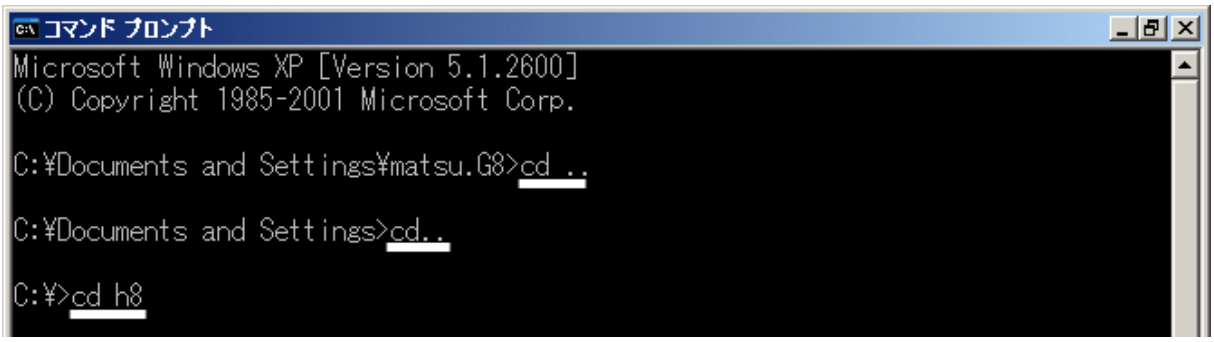
4.1　H8（ハードウェア）の書き込み準備

内容	写真
書込手順 1 書き込み用ジャンパを用意します.	
書込手順 2 電源を OFF にして，ジャンパを書き込み用ジャンパピンに取り付け，また Dsub 9 ピンをシリアル通信ポートに接続してください.	
書込手順 3 ジャンパを取り付けた状態で，電源をＯＮにして，書き込み準備ができました.「コマンドプロンプト」を立ち上げ，作成したプログラムのコンパイルおよび「hterm」による H8 への書き込みを行ってください.	

4.2　H8 プログラムのコンパイル

次に，ノートコンピュータから H8 のプログラムの書き込みをします.「スタート」＞「すべてのプログラム」＞「アクセサリ」＞「コマンドプロンプト」を起動してください.

コマンドプロンプト起動後，まず「C:¥h8」フォルダに移動してください．「cd ..」で上層のフォルダに移動することが出来ます．



「C:¥h8」に移動できたならば，プログラムのコンパイルを行います．ここでは「00demo.c」と「00demo.sub」のコンパイルを行うこととします．コンパイルは「compile 00demo」というように，「compile」をタイプし，スペース後に C ファイルおよび Sub ファイルの名前のみをタイプし，実行することです．無事に「LINKAGE EDITOR COMPLETED」が表示されれば，コンパイル終了です．表示されない場合，間違いのあるプログラムの行番号が表示されますので，改善してください．

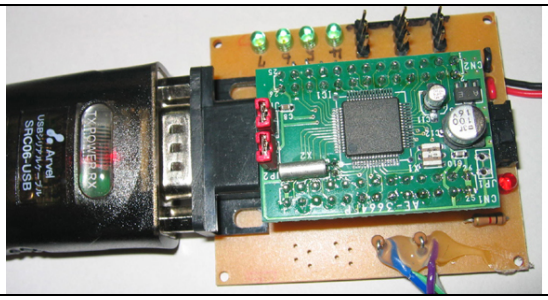
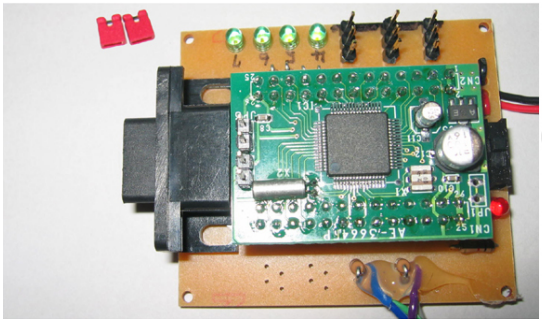
```
C:¥h8>compile 00demo

C:¥h8>ECHO OFF

H8S,H8/300 SERIES C Compiler Ver. 2.0D Evaluation software
Copyright (C) 1994,1996 Hitachi,Ltd.
Licensed Material of Hitachi,Ltd.
Licensed Material of Hitachi Engineering Co.,Ltd.

H SERIES LINKAGE EDITOR Ver. 5.3B Evaluation software
Copyright (C) Hitachi, Ltd.1989,1998
Copyright (C) HITACHI MICROCOMPUTER SYSTEM LTD. 1990,1998
Licensed Material of Hitachi, Ltd.

: OUTPUT 00demo
: PRINT 00demo
: INPUT 3664RES, 00demo
: ;INPUT 3664RES, start, com, 00demo
: LIB c38hn
: ;ROM     (D,X)
: ;START        VECT(0FC40),P,C,D,X,B(0F780),PASM,BASM(0FC74)
: ;START        VECT(0FC40),P,C,D,X,B(0F780),STK(0FF80),PASM,BASM(0FC74)
: ;START        VECT(00000),P,C,D,ASM_P(00034),X,B,ASM_B(0F780),STK(0FF80)
: START P(100)
: EXIT

LINKAGE EDITOR COMPLETED
```

4.3　H8 プログラムの書き込み方法

次に「C:¥h8」フォルダ内にある「writer」フォルダに移動するため,「cd writer」とタイプし,リターンボタンを押してください．そして書き込み用ソフト「hterm」を起動するため,「hterm」タイプし，リターンボタンを押してください.

「Ctrl」ボタンを押しながら「f」を押して「Set Boot Mode and Hit Any key」が表示された後,「Return」ボタンを押してください．そして,「Input Control Program Name:」が表示されたら,「3664.mot」と入力し，リターンを押してください．「Input Program Name:」の表示がありましたら，作成したプログラム名，ここでは「00demo」を入力し，リターンを押してください．プログラムが無事に書き込まれると「`Program Completed.」と表示されますので,「Esc」ボタンを押して，シリアル通信を終了し，プログラムの書き込み終了となります.

4.4　H8（ハードウェア）のプログラム実行準備

内容	写真
実行手順１ 書き込み完了後，まず電源を OFF して，取り付けられている書き込み用ジャンパを取り外し，かつ，シリアル通信用ポートからはずして下さい.	 OFF
実行手順２ 全て取り外した後，電源をＯＮにすれば，書き込みしたプログラムが実行されます.	 ＯＮ