17. A/Dコンバータ(単発モード)(プロジェクト:ad)

17.1 概要

本章では、0～5V の電圧をマイコンの A/D コンバータで読み込む方法を説明します。A/D 変換した結果は、マイコンボードの LED に出力します。今回の A/D 変換は、単発モードを使います。

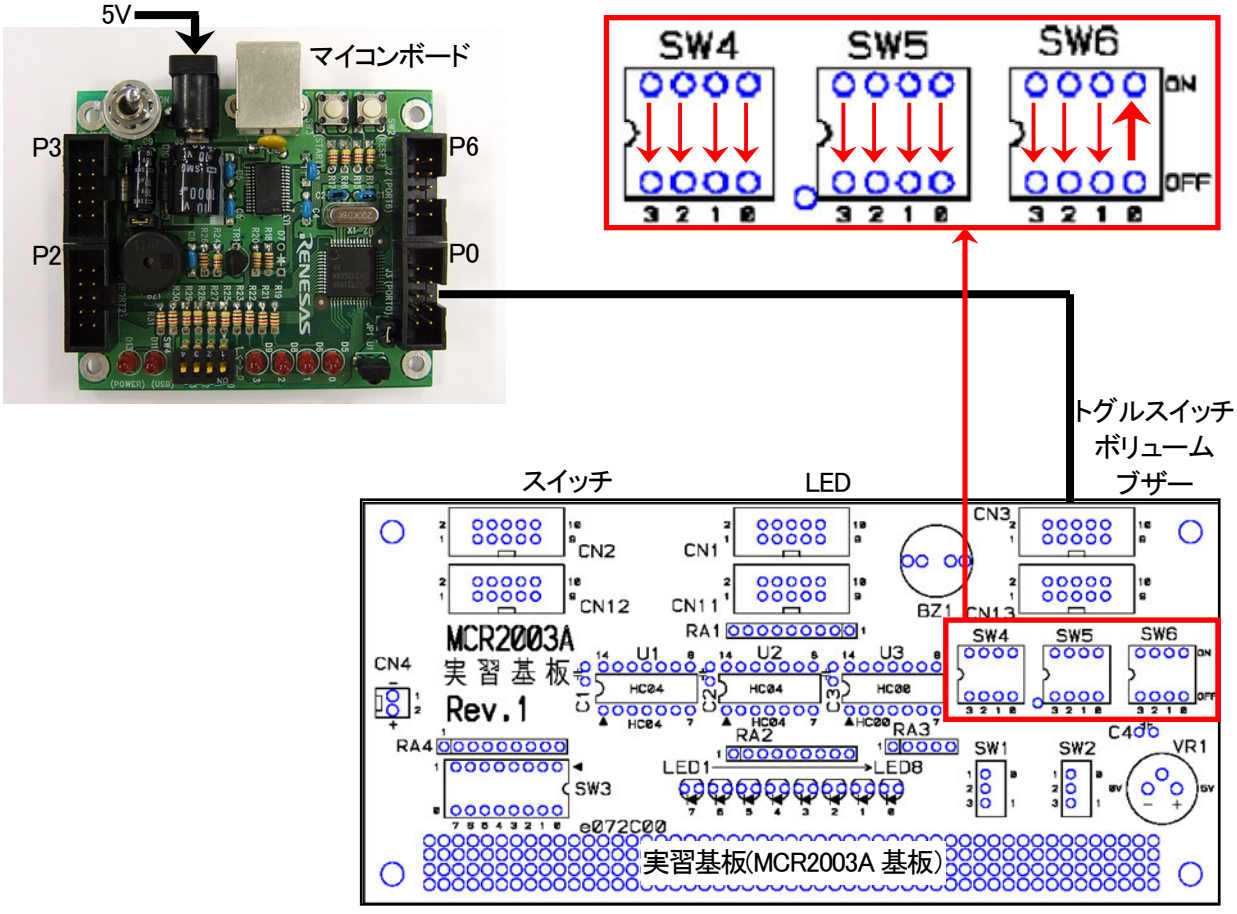
17.2 接続

■使用ポート

マイコンのポート	接続内容
P0_0 (J3)	ボリュームやアナログセンサなど 0～5V を出力する機器を接続します。実習基板またはセンサ部も接続可能です。
P1_3、P1_2、P1_1、P1_0	マイコンボード上の LED です。

■接続例 1

実習基板を使ったときの接続例を次に示します。

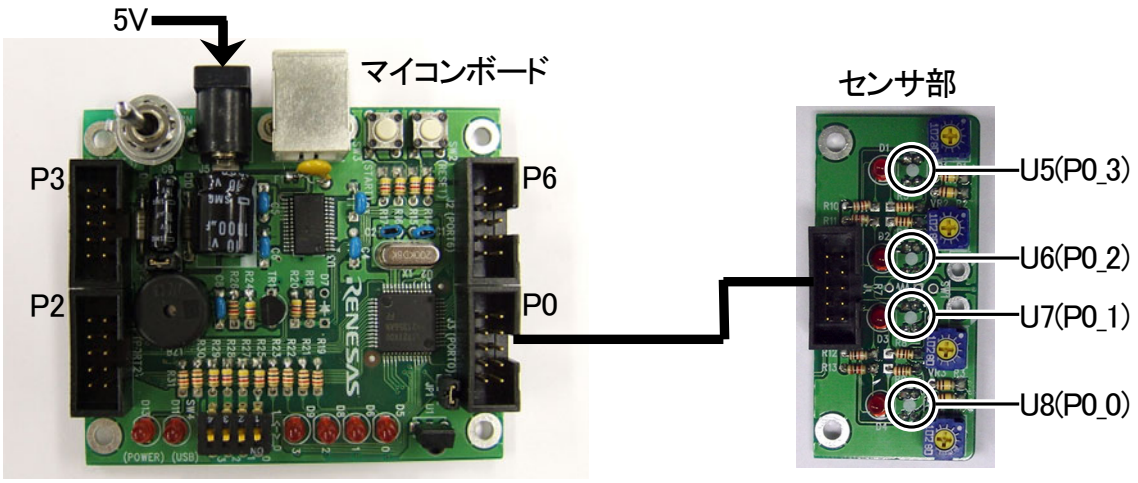


■操作方法(接続例 1 のとき)

実習基板のボリューム VR1 のつまみを左右に回します。ボリュームから出力された 0～5V の電圧をマイコンの P0_0 から読み込み、A/D 変換した値をマイコンボードの LED に出力します。

■接続例 2

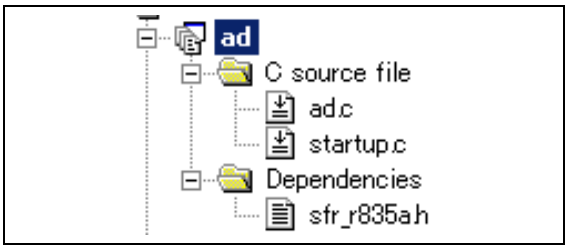
マイコンボードのポート 0(P0)とセンサ部をフラットケーブルで接続します。分離していない場合は、フラットケーブルで接続する必要はありません。センサは 4 個ありますが、今回電圧を読み込むのは、U8(P0_0)のフォトインタラプタです。



■操作方法(接続例 2 のとき)

センサ部の U8 の下部を白色や灰色や黒色に近づけます。フォトインタラプタ U8 から出力された 0～5V の電圧をマイコンの P0_0 から読み込み、A/D 変換した値をマイコンボードの LED に出力します。

17.3 プロジェクトの構成



	ファイル名	内容
1	startup.c	固定割り込みベクタアドレスの設定、スタートアッププログラム、RAM の初期化(初期値のないグローバル変数、初期値のあるグローバル変数の設定)などを行います。このファイルは共通で、どのプロジェクトもこのファイルから実行されます。
2	ad.c	実際に制御するプログラムが書かれています。R8C/35A の内蔵周辺機能(SFR)の初期化も行います。
3	sfr_r835a.h	R8C/35A マイコンの内蔵周辺機能を制御するためのレジスタ(Special Function Registers)を定義したファイルです。

17.4 プログラム「ad.c」

```
/************************************************************************/
/* 対象マイコン R8C/35A                                                 */
/* ﾌｧｲﾙ内容     A/D変換(単発モード)                                     */
/* バージョン   Ver.1.20                                                */
/* Date         2010.04.19                                              */
/* Copyright    ルネサスマイコンカーラリー事務局                        */
/*              日立インターメディックス株式会社                        */
/************************************************************************/
/*
入力：AN7(P0_0)端子 0～5V(ミニマイコンカーの赤外線フォトインタラプタU8)
出力：P1_3-P1_0(マイコンボードのLED)

AN7(P0_0)端子から入力した電圧をA/D変換して、デジタル値をマイコンボードの
LEDへ出力します。
*/

/*====================================*/
/* インクルード                       */
/*====================================*/
#include "sfr_r835a.h"                  /* R8C/35A SFRの定義ファイル     */

/*====================================*/
/* シンボル定義                       */
/*====================================*/

/*====================================*/
/* プロトタイプ宣言                   */
/*====================================*/
void init( void );
int get_ad7( void );
void led_out( unsigned char led );

/************************************************************************/
/* メインプログラム                                                     */
/************************************************************************/
void main( void )
{
    int ad;

    init();                             /* 初期化                       */

    while( 1 ) {
        ad = get_ad7();
        ad = ad >> 6;
        led_out( ad );
    }
}
```

```
 49 :  /************************************************************************/
 50 :  /* R8C/35A スペシャルファンクションレジスタ(SFR)の初期化                */
 51 :  /************************************************************************/
 52 :  void init( void )
 53 :  {
 54 :      int i;
 55 :
 56 :      /* クロックをXINクロック(20MHz)に変更 */
 57 :      prc0  = 1;                          /* プロテクト解除               */
 58 :      cm13  = 1;                          /* P4_6,P4_7をXIN-XOUT端子にする*/
 59 :      cm05  = 0;                          /* XINクロック発振              */
 60 :      for(i=0; i<50; i++ );               /* 安定するまで少し待つ(約10ms) */
 61 :      ocd2  = 0;                          /* システムクロックをXINにする  */
 62 :      prc0  = 0;                          /* プロテクトON                 */
 63 :
 64 :      /* ポートの入出力設定 */
 65 :      prc2 = 1;                           /* PD0のプロテクト解除          */
 66 :      pd0 = 0xe0;                         /* 7-5:LED 4:SW 3-0:アナログ電圧*/
 67 :      p1  = 0x0f;                         /* 3-0:LEDは消灯                */
 68 :      pd1 = 0xdf;                         /* 5:RXD0 4:TXD0 3-0:LED        */
 69 :      pd2 = 0xfe;                         /* 0:PushSW                     */
 70 :      pd3 = 0xfb;                         /* 4:Buzzer 2:IR                */
 71 :      pd4 = 0x83;                         /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
 72 :      pd5 = 0x40;                         /* 7:DIP SW                     */
 73 :      pd6 = 0xff;
 74 :  }
 75 :
 76 :  /************************************************************************/
 77 :  /* A/D値読み込み(AN7)                                                   */
 78 :  /* 引数    なし                                                         */
 79 :  /* 戻り値 A/D値 0～1023                                                 */
 80 :  /************************************************************************/
 81 :  int get_ad7( void )
 82 :  {
 83 :      int i;
 84 :
 85 :      /* A/Dコンバータの設定 */
 86 :      admod   = 0x03;                     /* 単発モードに設定             */
 87 :      adinsel = 0x07;                     /* 入力端子AN7(P0_0)を選択      */
 88 :      adcon1  = 0x30;                     /* A/D動作可能                  */
 89 :      asm( " nop ");                      /* φADの1サイクルウエイト入れる*/
 90 :      adcon0  = 0x01;                     /* A/D変換スタート              */
 91 :
 92 :      while( adcon0 & 0x01 );             /* A/D変換終了待ち              */
 93 :
 94 :      i = ad7;
 95 :
 96 :      return i;
 97 :  }
 98 :
 99 :  /************************************************************************/
100 :  /* マイコン部のLED出力                                                  */
101 :  /* 引数    スイッチ値 0～15                                             */
102 :  /************************************************************************/
103 :  void led_out( unsigned char led )
104 :  {
105 :      unsigned char data;
106 :
107 :      led = ~led;
108 :      led &= 0x0f;
109 :      data = p1 & 0xf0;
110 :      p1 = data | led;
111 :  }
112 :
113 :  /************************************************************************/
114 :  /* end of file                                                          */
115 :  /************************************************************************/
```

17.5 プログラムの解説

17.5.1 init関数

P0_0 はアナログ電圧入力端子なので、ポートの入出力設定は入力にします。忘れやすいので、気をつけてください。

```
64 :      /* ポートの入出力設定 */
65 :      prc2 = 1;                    /* PD0のプロテクト解除          */
66 :      pd0 = 0xe0;                  /* 7-5:LED 4:SW 3-0:アナログ電圧*/
67 :      p1  = 0x0f;                  /* 3-0:LEDは消灯                */
68 :      pd1 = 0xdf;                  /* 5:RXD0 4:TXD0 3-0:LED        */
69 :      pd2 = 0xfe;                  /* 0:PushSW                     */
70 :      pd3 = 0xfb;                  /* 4:Buzzer 2:IR                */
71 :      pd4 = 0x83;                  /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
72 :      pd5 = 0x40;                  /* 7:DIP SW                     */
73 :      pd6 = 0xff;
```

ポート0にセンサ部を接続している場合は、P0_4はマイクロスイッチ、P0_3～P0_1はセンサが繋がっているので入力にします。実習基板などを使ってこれらの端子が未接続の場合は、出力にしてください。

17.5.2 get_ad7 関数(A/Dコンバータの設定、A/D値取得)

今回は、AD7からA/D変換値を読み込むので、関数名を「get_ad**7**」にしました。アナログ入力端子を替える場合は、A/D変換する端子名に合わせて関数名を付けてください。

```
76 :  /************************************************************************/
77 :  /* A/D値読み込み(AN7)                                                   */
78 :  /* 引数    なし                                                         */
79 :  /* 戻り値 A/D値 0～1023                                                 */
80 :  /************************************************************************/
81 :  int get_ad7( void )
82 :  {
83 :      int i;
84 :
85 :      /* A/Dコンバータの設定 */
86 :      admod   = 0x03;              /* 単発モードに設定             */
87 :      adinsel = 0x07;              /* 入力端子AN7(P0_0)を選択      */
88 :      adcon1  = 0x30;              /* A/D動作可能                  */
89 :      asm( " nop ");               /* φADの1サイクルウエイト入れる*/
90 :      adcon0  = 0x01;              /* A/D変換スタート              */
91 :
92 :      while( adcon0 & 0x01 );      /* A/D変換終了待ち              */
93 :
94 :      i = ad7;
95 :
96 :      return i;
97 :  }
```

(1) A/Dコンバータとは

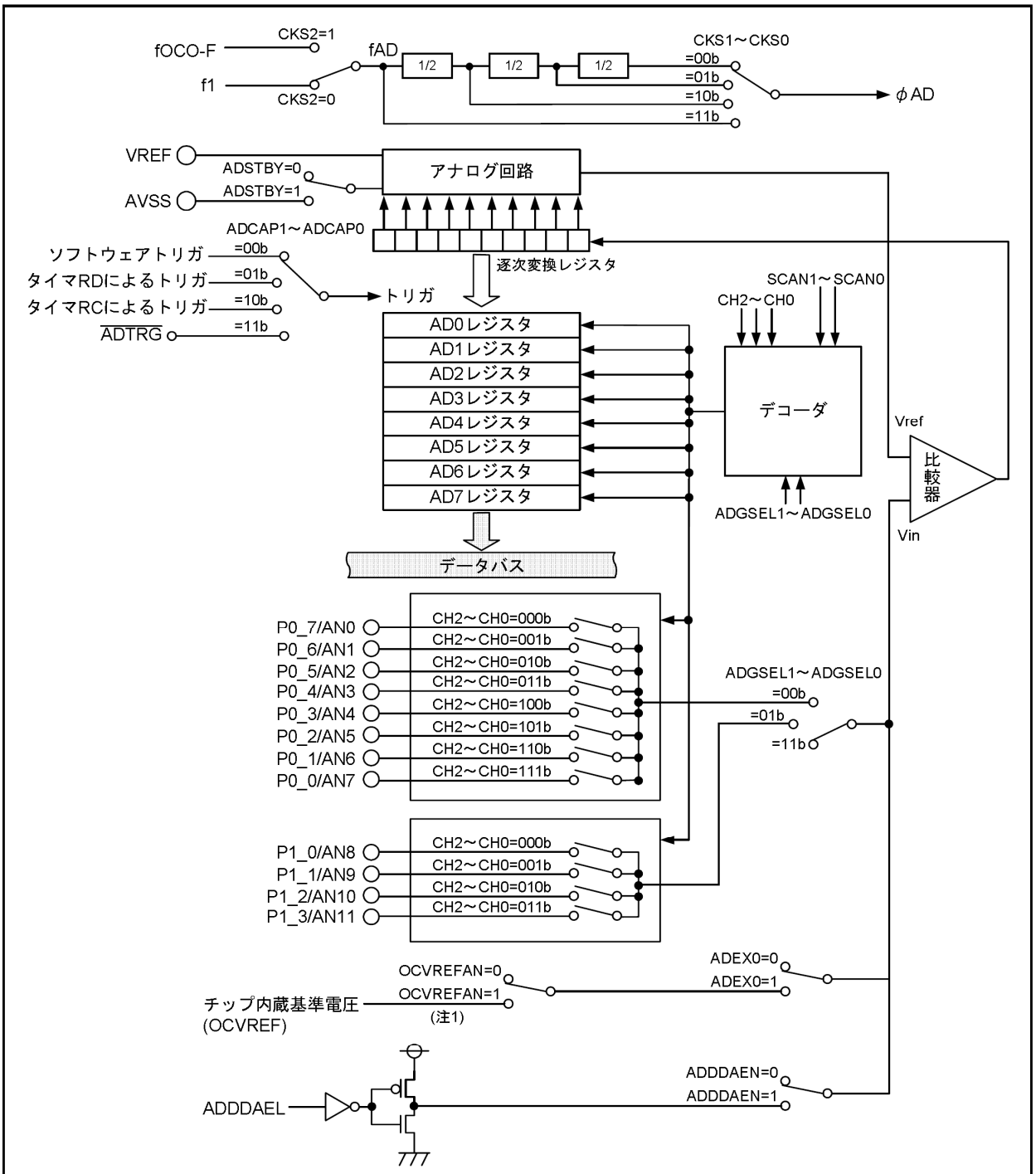
A/Dコンバータは、アナログ／デジタルコンバータの略称です。マイコンは"0"か"1"か("なし"か"あり"か)のデジタルで判断するので、外部からの入力電圧も0V("0")か5V("1")の2通りしか判断することができません。これ以外の電圧が入力されても、ある電圧を境にして"0"または"1"と判断します。

A/Dコンバータを使うと、外部から入力された電圧が何Vか知ることができます。例えば、A/Dコンバータを使わないと4.0Vの電圧が入力されてもマイコンは"1"としか判断できませんが、A/Dコンバータを使うと4.0Vという電圧が入力されている、と判断することができます。

R8C/35A マイコンの A/D コンバータの特徴を下記に示します。

内容	詳細	
A/D 変換方式	容量結合増幅器で構成された、10 ビットの逐次比較変換方式	
アナログ入力端子	P0_0～P0_7、P1_0～P1_3 の 12 本をアナログ入力端子として使用可能	
A/D 変換器の数	1 個 複数の A/D 変換を同時に行うことはできません。複数のアナログ入力端子の電圧を A/D 変換したい場合は、1 端子ずつ順番に行います。どの端子の電圧を A/D 変換するかは、プログラムで設定します。	
分解能	10bit 10bit とは 2 進数で 1 が 10 個ということです。$(11\ 1111\ 1111)_2=(1023)_{10}$となります。 0～5V(電源電圧)を 0～1023 の値に変換することができます。 電圧は次の式で求めることができます。 電圧＝A/D 変換値／1023×5.000(電源電圧) [V] A/D 変換値が 0 なら 0.000V、1023 なら 5.000V、1 なら約 0.004886V が入力されていることになります。A/D 変換値 1 あたり、約 0.004886V(1/1023)です。 ※「 (数値)$_2$ 」は 2 進数、「 (数値)$_{10}$ 」は 10 進数を表します。	
変換時間	最小 43 ϕ fAD fAD は A/D 変換するクロックです。クロックは内蔵クロック、外部クロックなど選択することができ、今回は外部クロックを選択します。ϕ は動作クロックです。ミニマイコンカーVer.2 は、外部クロックとして 20MHz のクリスタルを使っています。よって、変換時間は、次のようになります。 $43\times(1/外部クロック)=43\times(1/20\times10^6)=43\times0.05\times10^{-6}=2.15\times10^{-6}=$**2.15μs** 実際は A/D 変換する端子を設定したり、A/D 変換値を変数に保存したり計算したりするので、すべてを含めると 1 端子あたり数十μs 程度かかります(計算によって変わります)。	
動作モード	単発モード	AN0～AN11 から選択した 1 本の端子の入力電圧を、1 回 A/D 変換するモードです。
	繰り返しモード 0	AN0～AN11 から選択した 1 本の端子の入力電圧を、繰り返し A/D 変換するモードです。
	繰り返しモード 1	AN0～AN11 から選択した 1 本の端子の入力電圧を、繰り返し A/D 変換するモードです。A/D 変換値は、8 個のレジスタに順番に代入します。過去 8 個分の A/D 変換値を保存することができます。
	単掃引モード	AN0～AN11 から選択した 2 本、4 本、6 本、または 8 本の端子の入力電圧を、1 回ずつ A/D 変換するモードです。
	繰り返し掃引モード	AN0～AN11 から選択した 2 本、4 本、6 本、または 8 本の端子の入力電圧を、繰り返し A/D 変換するモードです。
アナログ入力端子	AN0(P0_7)、AN1(P0_6)、AN2(P0_5)、AN3(P0_4)、AN4(P0_3)、AN5(P0_2)、AN6(P0_1)、AN7(P0_0)、AN8(P1_0)、AN9(P1_1)、AN10(P1_2)、AN11(P1_3) の合計 12 本	

(2) A/Dコンバータのブロック図

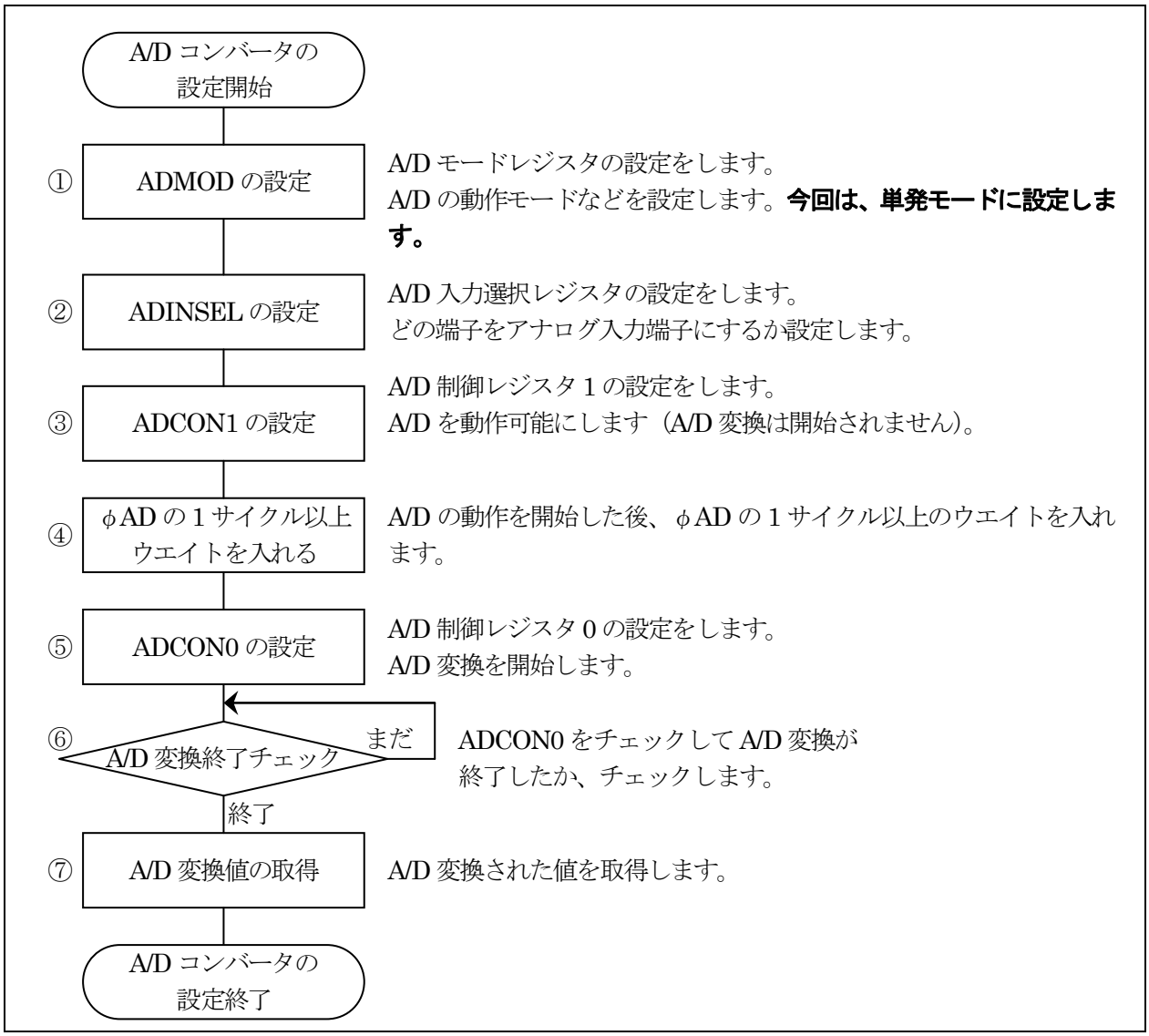


CKS0～CKS2、ADCAP0～ADCAP1：ADMODレジスタのビット
CH0～CH2、SCAN0～SCAN1、ADGSEL0～ADGSEL1：ADINSELレジスタのビット
ADEX0、ADSTBY、ADDDAEN、ADDDAEL：ADCON1レジスタのビット
OCVREFAN：OCVREFCRレジスタのビット

注1. チップ内蔵基準電圧をアナログ入力として使用する場合、ADEX0ビットを“1”(チップ内蔵基準電圧を選択)した後に、OCVREFANビットを“1”(チップ内蔵基準電圧とアナログ入力を接続)にしてください。
また、チップ内蔵基準電圧をアナログ入力として使用しない場合、OCVREFANビットを“0”(チップ内蔵基準電圧とアナログ入力を切断)にした後に、ADEX0ビットを“0”(拡張アナログ入力端子を非選択)にしてください。

(3) A/Dコンバータの設定

今回は、1本の端子からアナログ電圧を読み込み、1回だけA/D変換をする設定(**単発モード**)にします。フォトインタラプタが接続されているP0_0(AN7)の電圧を読み込みます。
レジスタの設定手順を下記に示します。



①A/D モードレジスタ(ADMOD:A-D mode register)の設定

A/D の動作モードを設定します。今回は、単発モードに設定します。

設定 bit	設定内容	内容	今回の内容
bit7,6	A/D 変換トリガ選択ビット bit7: adcap1 bit6: adcap0	00:ソフトウェアトリガ(ADCON0 レジスタの ADST ビット)による A/D 変換開始 01:タイマ RD からの変換トリガによる A/D 変換開始 10:タイマ RC からの変換トリガによる A/D 変換開始 11:外部トリガ(ADTRG)による A/D 変換開始 A/D 変換を開始するきっかけをどれにするか設定します。ソフト的に開始するので、"00"を選択します。	00
bit5～3	A/D 動作モード選択 bit5: md2 bit4: md1 bit3: md0	000:単発モード 001:設定しないでください 010:繰り返しモード 0 011:繰り返しモード 1 100:単掃引モード 101:設定しないでください 110:繰り返し掃引モード 111:設定しないでください 今回は、単発モードを選択します。	000
bit2	クロック源選択ビット cks2	0:f1(20MHz)を選択 1:fOCO-F(高速オンチップオシレータ)を選択 f1 を選択します。	0
bit1,0	分周選択ビット bit1: cks1 bit0: cks0	00:fAD の 8 分周 (8/20MHz=400ns) 01:fAD の 4 分周 (4/20MHz=200ns) 10:fAD の 2 分周 (2/20MHz=100ns) 11:fAD の 1 分周 (1/20MHz=50ns) fAD とは、bit2 で設定したクロック源のことです。このクロックを何分周で使用するか選択します。遅くする必要はないので、いちばん速い 1 分周で使用します。	11

A/D モードレジスタ(ADMOD)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**
16 進数	0				3			

②A/D 入力選択レジスタ(ADINSEL:A-D input select register)

どのアナログ入力端子を A/D 変換するか、設定します。

A/D 入力グループ選択ビット		bit5	bit4	bit3	アナログ入力端子選択ビット			アナログ入力端子
bit7	bit6				bit2	bit1	bit0	
0	0	0	0	0	0	0	0	AN0(P0_7)
0	0	0	0	0	0	0	1	AN1(P0_6)
0	0	0	0	0	0	1	0	AN2(P0_5)
0	0	0	0	0	0	1	1	AN3(P0_4)
0	0	0	0	0	1	0	0	AN4(P0_3)
0	0	0	0	0	1	0	1	AN5(P0_2)
0	0	0	0	0	1	1	0	AN6(P0_1)
0	0	0	0	0	1	1	1	AN7(P0_0)
0	1	0	0	0	0	0	0	AN8(P1_0)
0	1	0	0	0	0	0	1	AN9(P1_1)
0	1	0	0	0	0	1	0	AN10(P1_2)
0	1	0	0	0	0	1	1	AN11(P1_3)

今回は、AN7(P0_0)を選択します。A/D 入力選択レジスタ(ADINSEL)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	0	0	0	**1**	**1**	**1**
16 進数	0				7			

③A/D 制御レジスタ 1(ADCON1:A-D control register1)

A/D を動作可能にします。

設定 bit	上:ビット名 下:シンボル	内容	今回の内容
bit7	A/D 断線検出アシスト方式選択ビット(注 4)	0:変換前ディスチャージ 1:変換前プリチャージ A/D 断線検出アシストしませんのでどちらでも構いませんが、今回は"0"にしておきます。	0
bit6	A/D 断線検出アシスト機能許可ビット(注 4)	0:禁止 1:許可 A/D 断線検出アシストは使いません。	0
bit5	A/D スタンバイビット(注 3) adstby	0:A/D 動作停止(スタンバイ) 1:A/D 動作可能 A/D 動作可能にして A/D 変換できるようにします。この bit を"0"から"1"にしたときは、φA/D の 1 サイクル以上経過した後に A/D 変換を開始します。	1
bit4	8/10 ビットモード選択ビット bits	0:8 ビットモード 1:10 ビットモード A/D 変換を 10bit(0～1023)にするか、8bit(0～255)にするか選択します。今回は、10bit にします。	1
bit3～1		"000"を設定	000
bit0	拡張アナログ入力端子選択ビット(注 1) adex0	0:拡張アナログ入力端子を非選択 1:チップ内蔵基準電圧を選択(注 2) 拡張アナログ入力端子は使いません。	0

注 1. チップ内蔵基準電圧をアナログ入力として使用する場合、ADEX0 ビットを"1"(チップ内蔵基準電圧を選択)にした後に、OCVREFCR レジスタの OCVREFAN ビットを"1"(チップ内蔵基準電圧とアナログ入力を接続)にしてください。また、チップ内蔵基準電圧をアナログ入力として使用しない場合、OCVREFAN ビットを"0"(チップ内蔵基準電圧とアナログ入力を切断)にした後に、ADEX0 ビットを"0"(拡張アナログ入力端子を非選択)にしてください。

注 2. 単掃引モード、繰り返し掃引モードでは設定しないでください。

注 3. ADSTBY ビットを"0"(A/D 動作停止) から"1"(A/D 動作可能) にしたときは、φAD の 1 サイクル以上経過した後に A/D 変換を開始してください。

注 4. A/D 断線検出アシスト機能を許可にするためには、ADDDAEN ビットを"1"(許可)にした後、ADDDAEL ビットで変換開始状態を選択してください。断線時の変換結果は、外付け回路によって変化します。本機能はシステムに合わせた評価を十分に行った上で、使用してください。

A/D 制御レジスタ 1(ADCON1)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	**1**	**1**	0	0	0	0
16 進数	**3**				**0**			

④φAD の 1 サイクル以上ウエイトを入れる

③の bit5 の A/D スタンバイビットを"1"にした場合、φA/D の 1 サイクル以上経過した後に A/D 変換を開始しなければいけません。
このウエイトを入れるため、アセンブリ言語の nop 命令を実行します。C 言語ソースファイル内では、アセンブリ言語は実行できないため、asm 命令というアセンブリ言語を実行できる命令を使って nop 命令を実行します。ちなみに、nop は「No Operation（何もしない）」命令で、この命令を実行するのに 1 サイクル分の時間がかかります。
プログラムを下記に示します。

```
asm( " nop ");
```

⑤A/D 制御レジスタ 0（ADCON0：A-D control register0）

A/D 変換を開始します。

設定 bit	上：ビット名 下：シンボル	内容	今回の内容
bit7～1		"0000000"を設定	0000000
bit0	A/D 変換開始フラグ adst	0：A/D 変換停止 1：A/D 変換開始 A/D 変換を開始させるので"1"を設定します。A/D 変換が終了すると自動で"0"になります。	1

A/D 制御レジスタ 0（ADCON0）の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	0	0	0	0	0	**1**
16 進数	**0**				**1**			

⑥A/D 変換の終了チェック

A/D 変換が終了すると、A/D 制御レジスタ 0（ADCON0）の bit0 が"0"になります。プログラムでは、ADCON0 の bit0 が"0"かどうかチェック、"0"なら終了、"1"ならまだ変換中と判断できます。

```
while( adcon0 & 0x01 );          /* A/D 変換終了待ち          */
```

A/D 変換中、ADCON0 の bit0 が"1"なので、「adcon0 & 0x01＝0x01」となります。while 文はカッコの中が 0 以外なら繰り返しますので、この行を繰り返し続けます。
A/D 変換が完了すると、ADCON0 の bit0 が"0"になります。「adcon0 & 0x01＝0x00」となります。while 文はカッコの中が 0 なので、次の行へ進みます。

⑦A/D 変換値の取得

A/D 変換された結果は、A/D レジスタ 0～7(AD0～AD7)に格納されます。AD0～AD7 のどのレジスタに格納されるかは、アナログ入力端子によって変わります。アナログ入力端子と A/D レジスタの関係を下記に示します。

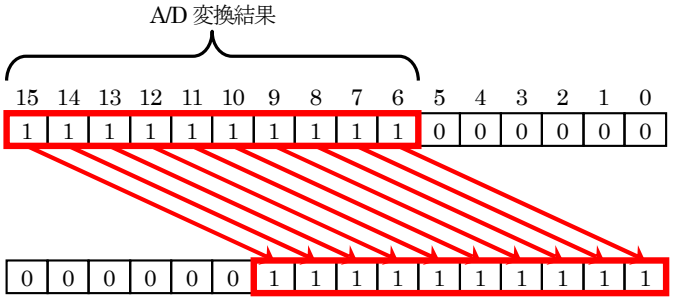
アナログ入力端子	読み込むレジスタ
AN0(P0_7)	AD0
AN1(P0_6)	AD1
AN2(P0_5)	AD2
AN3(P0_4)	AD3
AN4(P0_3)	AD4
AN5(P0_2)	AD5
AN6(P0_1)	AD6
AN7(P0_0)	AD7
AN8(P1_0)	AD0
AN9(P1_1)	AD1
AN10(P1_2)	AD2
AN11(P1_3)	AD3

今回は、AN7(P0_0 端子)を使用しているので、表より AD7 レジスタを読み込みます。

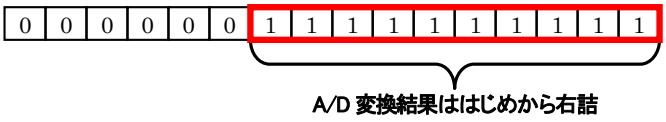
```
i = ad7;
```

※H8/3048F-ONEとR8C/35Aの格納方法の違い

H8/3048F-ONE の A/D 変換結果は、左詰で格納されるためプログラムで 6bit 右シフトしなければいけません。



R8C/35A の A/D 変換結果は、右詰で格納されるためプログラムでシフト処理は必要ありません。



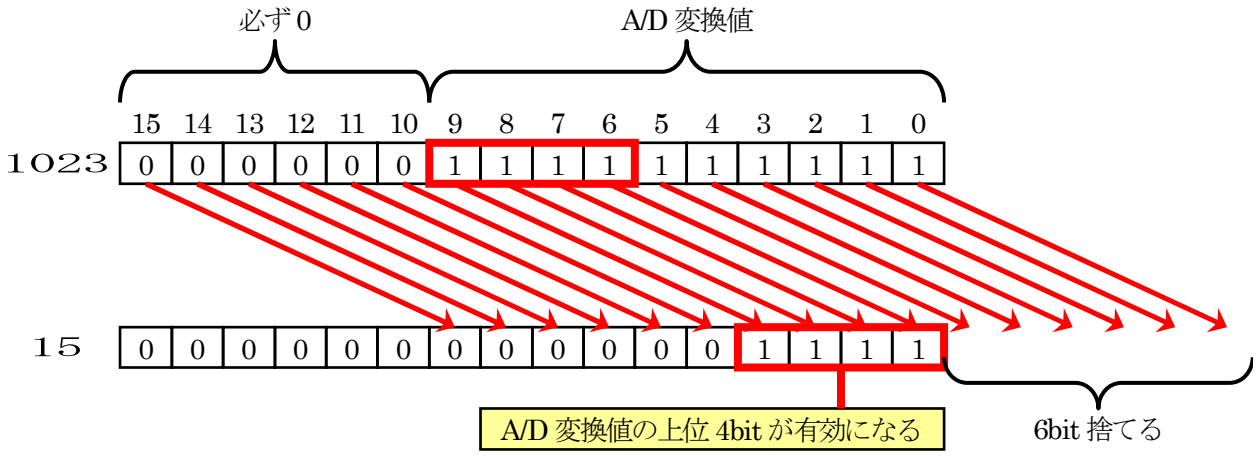
17.5.3 main関数

A/D 変換値を取得、その値をマイコンボード上の LED へ出力します。

```
 36 :  void main( void )
 37 :  {
 38 :      int ad;
 39 :
 40 :      init();                              /* 初期化                          */
 41 :
 42 :      while( 1 ) {
 43 :          ad = get_ad7();
 44 :          ad = ad >> 6;
 45 :          led_out( ad );
 46 :      }
 47 :  }
```

43 行	get_ad7 関数で A/D 変換値を取得し、ad 変数に格納します。
44 行	A/D 変換値は、0～1023(2 進数で 11 1111 1111)の値です。LED は 4 個しかありません。そのため今回は、2 進数で 10 桁の A/D 値を 4 桁に変換します。プログラムは、右シフトを 6 ビット分行い、下位の 6 桁を捨てます。その結果、A/D 値は 0～15 の値になり、ad 変数に代入します。
45 行	0～15 に変換した A/D 値をマイコンボード上の LED に出力します。

変数 ad の値が 1023 のとき、44 行のビットシフトの様子を下記に示します。



17.6 演習

(1) ポート 6 に実習基板の LED 部を接続して、A/D 変換値の上位 8bit をその LED へ出力しなさい。

(2) (1)の状態で、アナログ入力端子を P0_1 端子に変更して、LED へ出力しなさい。

18. A/Dコンバータ(繰り返しモード 0)(プロジェクト:ad_kurikaeshi)

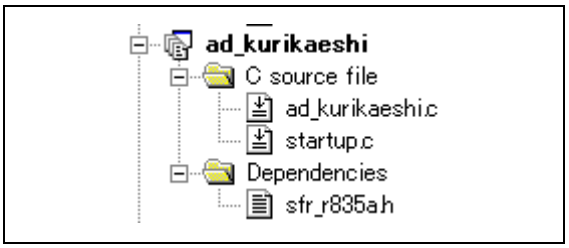
18.1 概要

本章では、0～5V の電圧をマイコンの A/D コンバータで読み込む方法を説明します。A/D 変換した結果は、マイコンボードの LED に出力します。今回の A/D 変換は、繰り返しモード 0 を使います。

18.2 接続

「17. A/Dコンバータ(単発モード)(プロジェクト:ad)」と同じです。

18.3 プロジェクトの構成



	ファイル名	内容
1	startup.c	固定割り込みベクタアドレスの設定、スタートアッププログラム、RAM の初期化(初期値のないグローバル変数、初期値のあるグローバル変数の設定)などを行います。このファイルは共通で、どのプロジェクトもこのファイルから実行されます。
2	ad_kurikaeshi.c	実際に制御するプログラムが書かれています。R8C/35A の内蔵周辺機能(SFR)の初期化も行います。
3	sfr_r835a.h	R8C/35A マイコンの内蔵周辺機能を制御するためのレジスタ(Special Function Registers)を定義したファイルです。

18.4 プログラム「ad_kurikaeshi.c」

```
/************************************************************************/
/* 対象マイコン R8C/35A                                                 */
/* ﾌｧｲﾙ内容     A/D変換(繰り返しモード0)                                 */
/* バージョン   Ver.1.20                                                */
/* Date         2010.04.19                                              */
/* Copyright    ルネサスマイコンカーラリー事務局                        */
/*              日立インターメディックス株式会社                        */
/************************************************************************/
/*
入力：AN7(P0_0)端子 0～5V(ミニマイコンカーの赤外線フォトインタラプタU8)
出力：P1_3-P1_0(マイコンボードのLED)

AN7(P0_0)端子から入力した電圧をA/D変換して、デジタル値をマイコンボードの
LEDへ出力します。
*/

/*====================================*/
/* インクルード                       */
/*====================================*/
#include "sfr_r835a.h"                  /* R8C/35A SFRの定義ファイル    */

/*====================================*/
/* シンボル定義                       */
/*====================================*/

/*====================================*/
/* プロトタイプ宣言                   */
/*====================================*/
void init( void );
int get_ad7( void );
void led_out( unsigned char led );

/************************************************************************/
/* メインプログラム                                                     */
/************************************************************************/
void main( void )
{
    int ad;

    init();                             /* 初期化                       */

    while( 1 ) {
        ad = get_ad7();
        ad = ad >> 6;
        led_out( ad );
    }
}

/************************************************************************/
/* R8C/35A スペシャルファンクションレジスタ(SFR)の初期化                */
/************************************************************************/
void init( void )
{
    int i;

    /* クロックをXINクロック(20MHz)に変更 */
    prc0  = 1;                          /* プロテクト解除               */
    cm13  = 1;                          /* P4_6,P4_7をXIN-XOUT端子にする*/
    cm05  = 0;                          /* XINクロック発振              */
    for(i=0; i<50; i++ );               /* 安定するまで少し待つ(約10ms) */
    ocd2  = 0;                          /* システムクロックをXINにする  */
    prc0  = 0;                          /* プロテクトON                 */

    /* ポートの入出力設定 */
    prc2 = 1;                           /* PD0のプロテクト解除          */
    pd0 = 0xe0;                         /* 7-5:LED 4:SW 3-0:アナログ電圧*/
    p1  = 0x0f;                         /* 3-0:LEDは消灯                */
    pd1 = 0xdf;                         /* 5:RXD0 4:TXD0 3-0:LED        */
    pd2 = 0xfe;                         /* 0:PushSW                     */
    pd3 = 0xfb;                         /* 4:Buzzer 2:IR                */
    pd4 = 0x83;                         /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
    pd5 = 0x40;                         /* 7:DIP SW                     */
    pd6 = 0xff;

    /* A/Dコンバータの設定 */
    admod   = 0x13;                     /* 繰り返しモード0に設定        */
    adinsel = 0x07;                     /* 入力端子AN7(P0_0)を選択      */
    adcon1  = 0x30;                     /* A/D動作可能                  */
    asm( " nop ");                      /* φADの1サイクルウエイト入れる*/
    adcon0  = 0x01;                     /* A/D変換スタート              */
}

```

```
 83 :  /************************************************************************/
 84 :  /* A/D値読み込み(AN7)                                                   */
 85 :  /* 引数    なし                                                         */
 86 :  /* 戻り値 A/D値 0～1023                                                 */
 87 :  /************************************************************************/
 88 :  int get_ad7( void )
 89 :  {
 90 :      int i;
 91 :
 92 :      /* 繰り返しモード0は、自動的に繰り返すので、結果を読み込むだけ */
 93 :      i = ad7;
 94 :
 95 :      return i;
 96 :  }
 97 :
 98 :  /************************************************************************/
 99 :  /* マイコン部のLED出力                                                  */
100 :  /* 引数    スイッチ値 0～15                                             */
101 :  /************************************************************************/
102 :  void led_out( unsigned char led )
103 :  {
104 :      unsigned char data;
105 :
106 :      led = ~led;
107 :      led &= 0x0f;
108 :      data = p1 & 0xf0;
109 :      p1 = data | led;
110 :  }
111 :
112 :  /************************************************************************/
113 :  /* end of file                                                          */
114 :  /************************************************************************/
```

18.5 プログラムの解説

18.5.1 init関数(I/Oポートの入出力設定)

P0_0 はアナログ電圧入力端子なので、ポートの入出力設定は入力にします。忘れやすいので、気をつけてください。

```
 64 :      /* ポートの入出力設定 */
 65 :      prc2 = 1;                   /* PD0のプロテクト解除          */
 66 :      pd0 = 0xe0;                 /* 7-5:LED 4:SW 3-0:アナログ電圧*/
 67 :      p1  = 0x0f;                 /* 3-0:LEDは消灯                */
 68 :      pd1 = 0xdf;                 /* 5:RXD0 4:TXD0 3-0:LED        */
 69 :      pd2 = 0xfe;                 /* 0:PushSW                     */
 70 :      pd3 = 0xfb;                 /* 4:Buzzer 2:IR                */
 71 :      pd4 = 0x83;                 /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
 72 :      pd5 = 0x40;                 /* 7:DIP SW                     */
 73 :      pd6 = 0xff;
```

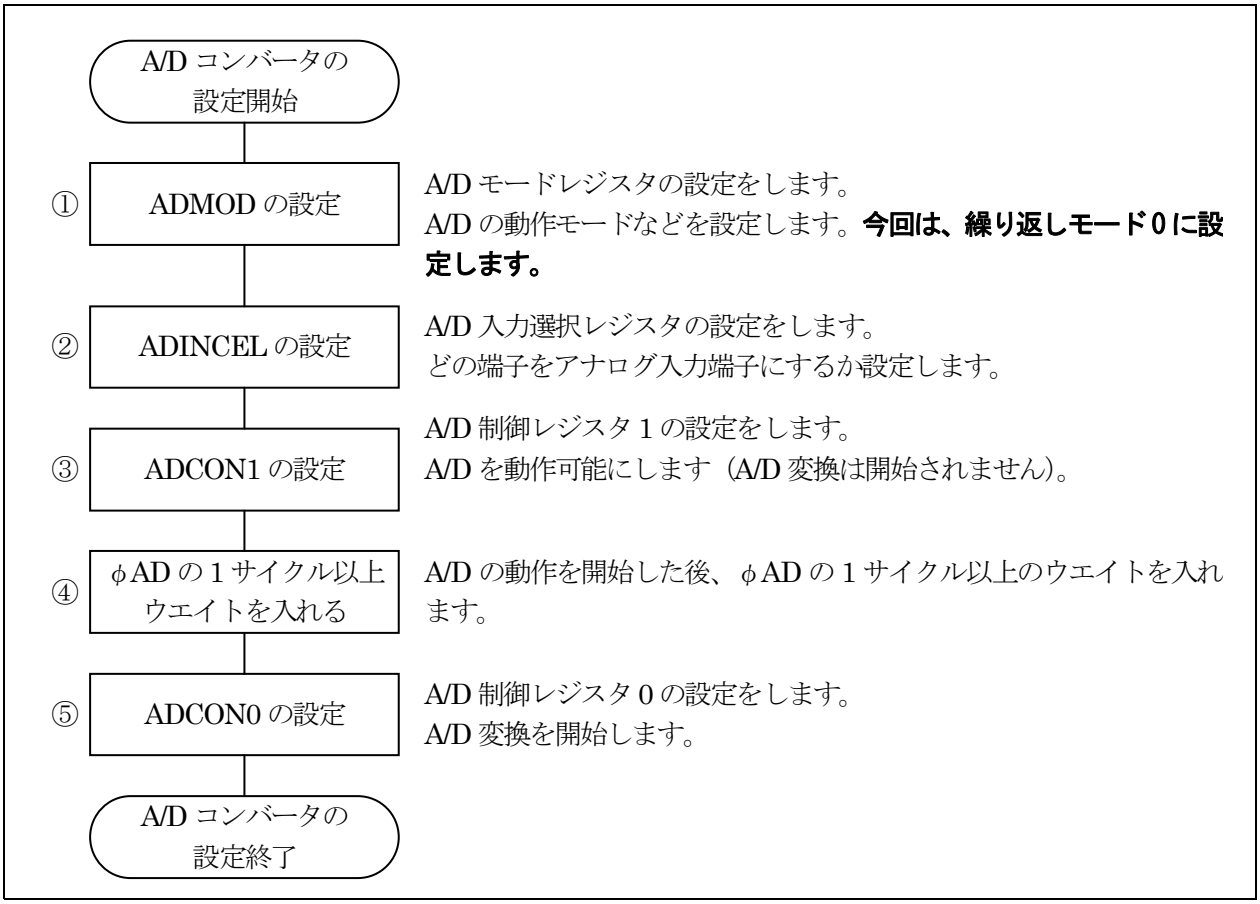
ポート 0 にセンサ部を接続している場合は、P0_4 はマイクロスイッチ、P0_3～P0_1 はセンサが繋がっているので入力にします。実習基板などを使ってこれらの端子が未接続の場合は、出力にしてください。

18.5.2 init関数(A/Dコンバータの設定)

A/D コンバータを設定するプログラムは、次のようになります。

```
75 :      /* A/Dコンバータの設定 */
76 :      admod    = 0x13;                /* 繰り返しモード0に設定          */
77 :      adinsel  = 0x07;                /* 入力端子AN7(p0_0)を選択        */
78 :      adcon1   = 0x30;                /* A/D動作可能                    */
79 :      asm( " nop ");                  /* φADの1サイクルウエイト入れる*/
80 :      adcon0   = 0x01;                /* A/D変換スタート                */
```

今回は、1 本の端子からアナログ電圧を読み込み、繰り返し A/D 変換する設定(**繰り返しモード 0**)にします。フォトインタラプタが接続されている P0_0(AN7)の電圧を読み込みます。
レジスタの設定手順を下記に示します。



①A/D モードレジスタ(ADMOD:A-D mode register)の設定

A/D の動作モードを設定します。今回は繰り返しモード 0 に設定します。

設定 bit	上:ビット名 下:シンボル	内容	今回の内容
bit7,6	A/D 変換トリガ選択ビット bit7: adcap1 bit6: adcap0	00:ソフトウェアトリガ(ADCON0 レジスタの ADST ビット) による A/D 変換開始 01:タイマ RD からの変換トリガによる A/D 変換開始 10:タイマ RC からの変換トリガによる A/D 変換開始 11:外部トリガ(ADTRG)による A/D 変換開始 A/D 変換を開始するきっかけをどれにするか設定します。ソフト的に開始するので、"00"を選択します。	00
bit5～3	A/D 動作モード選択 bit5: md2 bit4: md1 bit3: md0	000:単発モード 001:設定しないでください 010:繰り返しモード 0 011:繰り返しモード 1 100:単掃引モード 101:設定しないでください 110:繰り返し掃引モード 111:設定しないでください 今回は、繰り返しモード 0 を選択します。	010
bit2	クロック源選択ビット cks2	0:f1 (20MHz)を選択 1:fOCO-F(高速オンチップオシレータ)を選択 f1 を選択します。	0
bit1,0	分周選択ビット bit1: cks1 bit0: cks0	00:fAD の 8 分周 (8/20MHz=400ns) 01:fAD の 4 分周 (4/20MHz=200ns) 10:fAD の 2 分周 (2/20MHz=100ns) 11:fAD の 1 分周 (1/20MHz=50ns) fAD とは、bit2 で設定したクロック源のことです。このクロックを何分周で使用するか選択します。遅くする必要はないので、いちばん速い 1 分周で使用します。	11

A/D モードレジスタ(ADMOD)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**1**
16 進数	1				3			

②A/D 入力選択レジスタ(ADINSEL:A-D input select register)

どのアナログ入力端子を A/D 変換するか、設定します。

A/D 入力グループ選択ビット		bit5	bit4	bit3	アナログ入力端子選択ビット			アナログ入力端子
bit7	bit6				bit2	bit1	bit0	
0	0	0	0	0	0	0	0	AN0(P0_7)
0	0	0	0	0	0	0	1	AN1(P0_6)
0	0	0	0	0	0	1	0	AN2(P0_5)
0	0	0	0	0	0	1	1	AN3(P0_4)
0	0	0	0	0	1	0	0	AN4(P0_3)
0	0	0	0	0	1	0	1	AN5(P0_2)
0	0	0	0	0	1	1	0	AN6(P0_1)
0	0	0	0	0	1	1	1	AN7(P0_0)
0	1	0	0	0	0	0	0	AN8(P1_0)
0	1	0	0	0	0	0	1	AN9(P1_1)
0	1	0	0	0	0	1	0	AN10(P1_2)
0	1	0	0	0	0	1	1	AN11(P1_3)

今回は、AN7(P0_0)を選択します。A/D 入力選択レジスタ(ADINSEL)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	0	0	0	**1**	**1**	**1**
16 進数	0				7			

③A/D 制御レジスタ 1(ADCON1:A-D control register1)

A/D を動作可能にします。

設定 bit	上:ビット名 下:シンボル	内容	今回の内容
bit7	A/D 断線検出アシスト方式選択ビット(注 4)	0:変換前ディスチャージ 1:変換前プリチャージ A/D 断線検出アシストしませんのでどちらでも構いませんが、今回は"0"にしておきます。	0
bit6	A/D 断線検出アシスト機能許可ビット(注 4)	0:禁止 1:許可 A/D 断線検出アシストは使いません。	0
bit5	A/D スタンバイビット(注 3) adstby	0:A/D 動作停止(スタンバイ) 1:A/D 動作可能 A/D 動作可能にして A/D 変換できるようにします。この bit を"0"から"1"にしたときは、φA/D の 1 サイクル以上経過した後に A/D 変換を開始します。	1
bit4	8/10 ビットモード選択ビット bits	0:8 ビットモード 1:10 ビットモード A/D 変換を 10bit(0～1023)にするか、8bit(0～255)にするか選択します。今回は、10bit にします。	1
bit3～1		"000"を設定	000
bit0	拡張アナログ入力端子選択ビット(注 1) adex0	0:拡張アナログ入力端子を非選択 1:チップ内蔵基準電圧を選択(注 2) 拡張アナログ入力端子は使いません。	0

注 1. チップ内蔵基準電圧をアナログ入力として使用する場合、ADEX0 ビットを"1"(チップ内蔵基準電圧を選択)にした後に、OCVREFCR レジスタの OCVREFAN ビットを"1"(チップ内蔵基準電圧とアナログ入力を接続)にしてください。また、チップ内蔵基準電圧をアナログ入力として使用しない場合、OCVREFAN ビットを"0"(チップ内蔵基準電圧とアナログ入力を切断)にした後に、ADEX0 ビットを"0"(拡張アナログ入力端子を非選択)にしてください。

注 2. 単掃引モード、繰り返し掃引モードでは設定しないでください。

注 3. ADSTBY ビットを"0"(A/D 動作停止) から"1"(A/D 動作可能) にしたときは、φAD の 1 サイクル以上経過した後に A/D 変換を開始してください。

注 4. A/D 断線検出アシスト機能を許可にするためには、ADDDAEN ビットを"1"(許可)にした後、ADDDAEL ビットで変換開始状態を選択してください。断線時の変換結果は、外付け回路によって変化します。本機能はシステムに合わせた評価を十分に行った上で、使用してください。

A/D 制御レジスタ 1(ADCON1)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	**1**	**1**	0	0	0	0
16 進数	**3**				**0**			

④φAD の 1 サイクル以上ウエイトを入れる

③の bit5 の A/D スタンバイビットを"1"にした場合、φA/D の 1 サイクル以上経過した後に A/D 変換を開始しなければいけません。
そのウエイトを入れるため、アセンブリ言語の nop 命令を実行します。C 言語ソースファイル内では、アセンブリ言語は実行できないため、asm 命令というアセンブリ言語を実行できる命令を使って nop 命令を実行します。ちなみに、nop は「No Operation（何もしない）」命令で、この命令を実行するのに 1 サイクル分の時間がかかります。
プログラムを下記に示します。

```
asm( " nop ");
```

⑤A/D 制御レジスタ 0（ADCON0：A-D control register0）

A/D 変換を開始します。

設定 bit	上：ビット名 下：シンボル	内容	今回の内容
bit7～1		"0000000"を設定	000 0000
bit0	A/D 変換開始フラグ adst	0：A/D 変換停止 1：A/D 変換開始 A/D 変換を開始させるので"1"を設定します。	1

A/D 制御レジスタ 0（ADCON0)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	0	0	0	0	0	**1**
16 進数	**0**				**1**			

18.5.3 get_ad7 関数

get_ad7 関数は、A/D 変換した結果を取得する関数です。

```
 83 :  /************************************************************************/
 84 :  /* A/D値読み込み(AN7)                                                   */
 85 :  /* 引数    なし                                                         */
 86 :  /* 戻り値 A/D値 0～1023                                                 */
 87 :  /************************************************************************/
 88 :  int get_ad7( void )
 89 :  {
 90 :      int i;
 91 :
 92 :      /* 繰り返しモード0は、自動的に繰り返すので、結果を読み込むだけ */
 93 :      i = ad7;
 94 :
 95 :      return i;
 96 :  }
```

93 行	ad 変換した結果が格納されている ad7 レジスタの値を、変数 i に代入します。

A/D 変換された結果は、A/D レジスタ 0～7(AD0～AD7)に格納されます。AD0～AD7 のどのレジスタに格納されるかは、アナログ入力端子によって変わります。アナログ入力端子と A/D レジスタの関係を下記に示します。

アナログ入力端子	読み込むレジスタ
AN0(P0_7)	AD0
AN1(P0_6)	AD1
AN2(P0_5)	AD2
AN3(P0_4)	AD3
AN4(P0_3)	AD4
AN5(P0_2)	AD5
AN6(P0_1)	AD6
AN7(P0_0)	AD7
AN8(P1_0)	AD0
AN9(P1_1)	AD1
AN10(P1_2)	AD2
AN11(P1_3)	AD3

今回は、AN7(P0_0 端子)を使用しているので、表より AD7 レジスタを読み込みます。

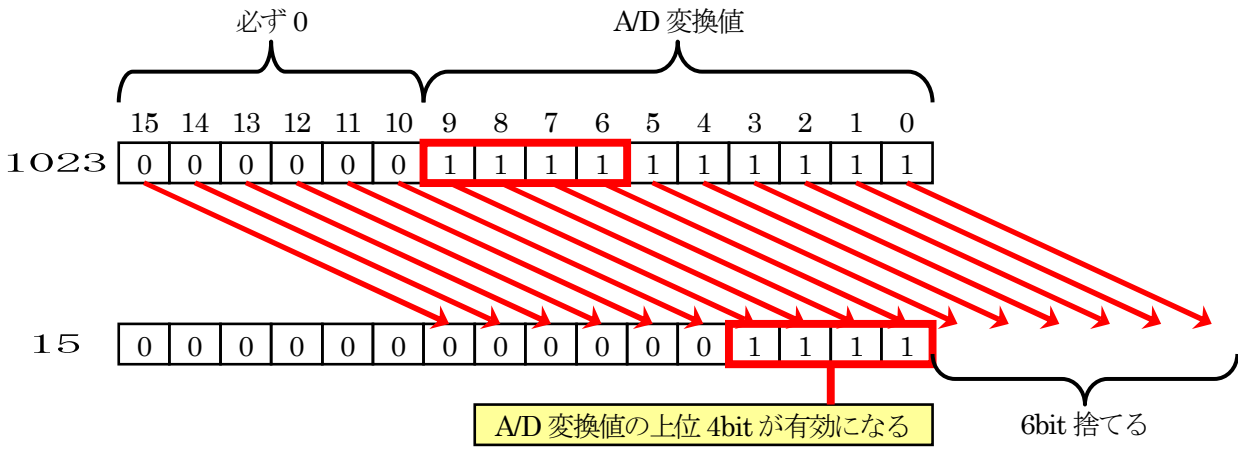
18.5.4 main関数

A/D 変換値を取得、マイコンボード上の LED へ値を出力します。

```
 36 :  void main( void )
 37 :  {
 38 :      int ad;
 39 :
 40 :      init();                              /* 初期化                          */
 41 :
 42 :      while( 1 ) {
 43 :          ad = get_ad7();
 44 :          ad = ad >> 6;
 45 :          led_out( ad );
 46 :      }
 47 :  }
```

43 行	get_ad7 関数で A/D 変換値を取得し、ad 変数に格納します。
44 行	A/D 変換値は、0～1023(2 進数で 11 1111 1111)の値です。LED は 4 個しかありません。そのため今回は、2 進数で 10 桁の A/D 値を 4 桁に変換します。プログラムは、右シフトを 6 ビット分行い、下位の 6 桁を捨てます。その結果、A/D 値は 0～15 の値になり、ad 変数に代入します。
45 行	0～15 に変換した A/D 値をマイコンボード上の LED に出力します。

変数 ad の値が 1023 のとき、44 行のビットシフトのようすを下記に示します。



18.6 演習

(1) ポート 6 に LED 基板(実習基板の LED 部など)を接続して、A/D 変換値の上位 8bit をその LED へ出力しなさい。

(2) (1)の状態で、アナログ入力端子を P0_1 端子に変更して、LED へ出力しなさい。

19. A/Dコンバータ(繰り返し掃引モード)(プロジェクト:ad_kurikaeshi_souin)

19.1 概要

本章では、2 本の 0～5V の電圧信号をマイコンの A/D コンバータで読み込む方法を説明します。A/D 変換した結果は、マイコンボードの LED に出力します。今回の A/D 変換は、繰り返し掃引モードを使います。今回は 2 本分ですが、プログラムを替えることにより 8 本の電圧信号まで読み込むことができます。

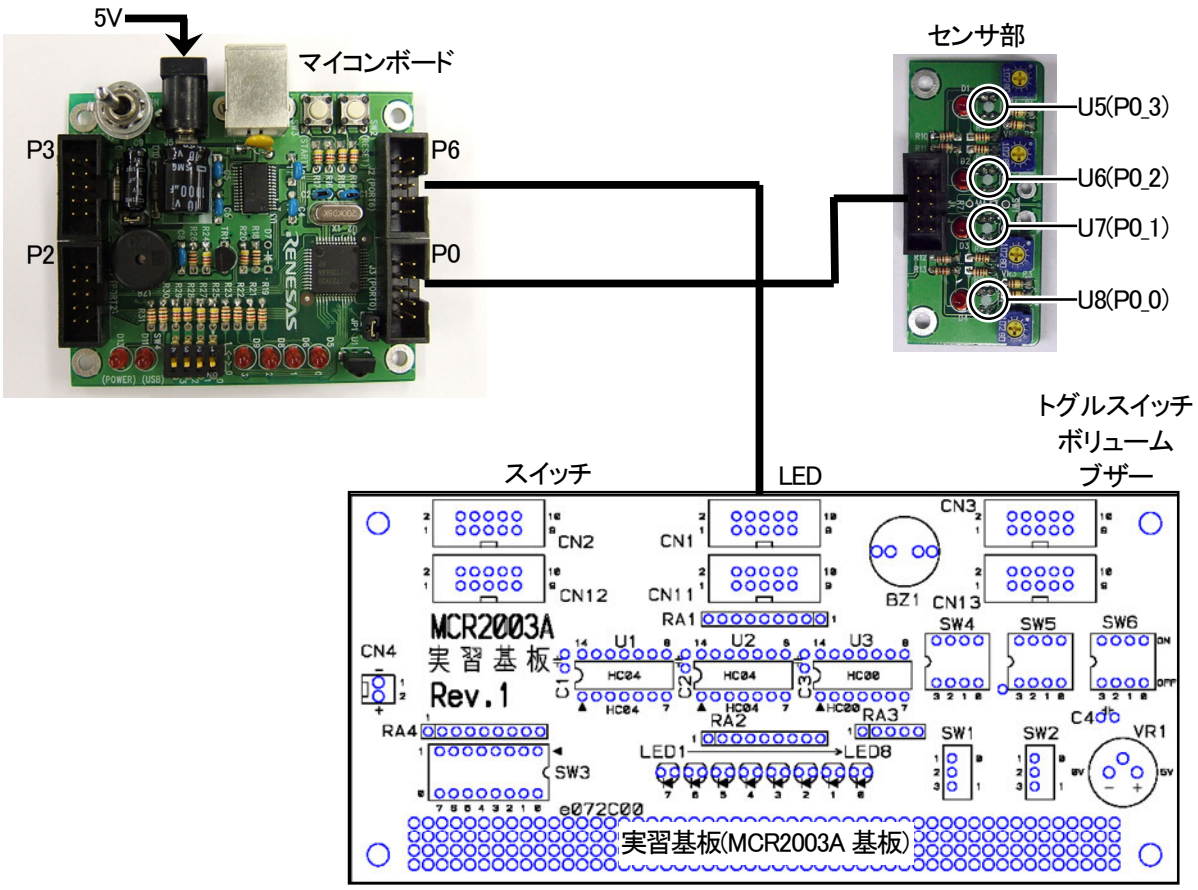
19.2 接続

■使用ポート

マイコンのポート	接続内容
P0_0、P0_1 (J3)	センサ部を接続します。U8、U7 の A/D 値を読み込みます。
P1_3、P1_2、P1_1、P1_0	マイコンボード上の LED です。
P6 (J2)	実習基板の LED 部など、出力機器を接続します。

■接続

マイコンボードのポート 0 とセンサ部をフラットケーブルで接続します。センサは 4 個ありますが、今回 LED に出力するのは、U8(P0_0)と U7(P0_1)のセンサです。また、マイコンボードのポート 6 と実習基板の LED 部を接続します。

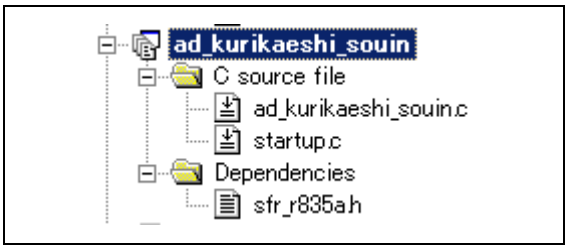


■操作方法

センサ部の U8 の下部を白色や灰色や黒色に近づけます。フォトインタラプタ U8 から出力された 0～5V の電圧をマイコンの P0_0 から読み込み、A/D 変換した値をマイコンボードの LED に出力します。

センサ部の U7 の下部を白色や灰色や黒色に近づけます。フォトインタラプタ U7 から出力された 0～5V の電圧をマイコンの P0_1 から読み込み、A/D 変換した値を実習基板の LED 部に出力します。

19.3 プロジェクトの構成



	ファイル名	内容
1	startup.c	固定割り込みベクタアドレスの設定、スタートアッププログラム、RAM の初期化(初期値のないグローバル変数、初期値のあるグローバル変数の設定)などを行います。このファイルは共通で、どのプロジェクトもこのファイルから実行されます。
2	ad_kurikaeshi_souin.c	実際に制御するプログラムが書かれています。R8C/35A の内蔵周辺機能(SFR)の初期化も行います。
3	sfr_r835a.h	R8C/35A マイコンの内蔵周辺機能を制御するためのレジスタ(Special Function Registers)を定義したファイルです。

19.4 プログラム「ad_kurikaeshi_souin.c」

```
 1 :  /***********************************************************************/
 2 :  /* 対象マイコン R8C/35A                                                */
 3 :  /* ﾌｧｲﾙ内容     A/D変換(繰り返し掃引モード)                            */
 4 :  /* バージョン   Ver.1.20                                               */
 5 :  /* Date         2010.04.19                                             */
 6 :  /* Copyright    ルネサスマイコンカーラリー事務局                       */
 7 :  /*              日立インターメディックス株式会社                       */
 8 :  /***********************************************************************/
 9 :  /*
10 :  入力：AN0(P0_7)～AN7(P0_0)端子
11 :        0～5V(ミニマイコンカーの赤外線フォトインタラプタ(U5,U6,U7,U8)
12 :  出力：P1_3-P1_0(マイコンボードのLED)
13 :        P6_7-P6_0(実習基板のLED部など)
14 :
15 :  AN0(P0_7)～AN7(P0_0)端子の8端子から入力した電圧をA/D変換して、
16 :  デジタル値をマイコンボードのLEDと実習ボードのLED部へ出力します。
17 :  8端子分のA/D変換しますが、入出力設定を出力しているAD端子の値は不定です。
18 :  */
19 :
20 :  /*====================================*/
21 :  /* インクルード                       */
22 :  /*====================================*/
23 :  #include "sfr_r835a.h"                 /* R8C/35A SFRの定義ファイル     */
24 :
25 :  /*====================================*/
26 :  /* シンボル定義                       */
27 :  /*====================================*/
28 :
29 :  /*====================================*/
30 :  /* プロトタイプ宣言                   */
31 :  /*====================================*/
32 :  void init( void );
33 :  void led_out( unsigned char led );
34 :
```

```
 35 :  /************************************************************************/
 36 :  /* メインプログラム                                                     */
 37 :  /************************************************************************/
 38 :  void main( void )
 39 :  {
 40 :      int ad7data, ad6data;
 41 :
 42 :      init();                             /* 初期化                       */
 43 :
 44 :      while( 1 ) {
 45 :          // AD7を出力
 46 :          ad7data = ad7;                  /* AD7取得                      */
 47 :          ad7data = ad7data >> 6;
 48 :          led_out( ad7data );             /* マイコンボードのLEDへ出力    */
 49 :
 50 :          // AD6を出力
 51 :          ad6data = ad6;                  /* AD6取得                      */
 52 :          ad6data = ad6data >> 6;
 53 :          p6 = ad6data;                   /* P6のbit3～0のLEDへ出力       */
 54 :      }
 55 :  }
 56 :
 57 :  /************************************************************************/
 58 :  /* R8C/35A スペシャルファンクションレジスタ(SFR)の初期化                */
 59 :  /************************************************************************/
 60 :  void init( void )
 61 :  {
 62 :      int i;
 63 :
 64 :      /* クロックをXINクロック(20MHz)に変更 */
 65 :      prc0  = 1;                          /* プロテクト解除               */
 66 :      cm13  = 1;                          /* P4_6,P4_7をXIN-XOUT端子にする*/
 67 :      cm05  = 0;                          /* XINクロック発振              */
 68 :      for(i=0; i<50; i++ );               /* 安定するまで少し待つ(約10ms) */
 69 :      ocd2  = 0;                          /* システムクロックをXINにする  */
 70 :      prc0  = 0;                          /* プロテクトON                 */
 71 :
 72 :      /* ポートの入出力設定 */
 73 :      prc2 = 1;                           /* PD0のプロテクト解除          */
 74 :      pd0 = 0xe0;                         /* 7-5:LED 4:SW 3-0:アナログ電圧*/
 75 :      p1  = 0x0f;                         /* 3-0:LEDは消灯                */
 76 :      pd1 = 0xdf;                         /* 5:RXD0 4:TXD0 3-0:LED        */
 77 :      pd2 = 0xfe;                         /* 0:PushSW                     */
 78 :      pd3 = 0xfb;                         /* 4:Buzzer 2:IR                */
 79 :      pd4 = 0x83;                         /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
 80 :      pd5 = 0x40;                         /* 7:DIP SW                     */
 81 :      pd6 = 0xff;                         /* LEDなど出力                  */
 82 :
 83 :      /* A/Dコンバータの設定 */
 84 :      admod   = 0x33;                     /* 繰り返し掃引モードに設定     */
 85 :      adinsel = 0x30;                     /* 入力端子P0の8端子を選択      */
 86 :      adcon1  = 0x30;                     /* A/D動作可能                  */
 87 :      asm( " nop ");                      /* φADの1サイクルウエイト入れる*/
 88 :      adcon0  = 0x01;                     /* A/D変換スタート              */
 89 :  }
 90 :
 91 :  /************************************************************************/
 92 :  /* マイコン部のLED出力                                                  */
 93 :  /* 引数    スイッチ値 0～15                                             */
 94 :  /************************************************************************/
 95 :  void led_out( unsigned char led )
 96 :  {
 97 :      unsigned char data;
 98 :
 99 :      led = ~led;
100 :      led &= 0x0f;
101 :      data = p1 & 0xf0;
102 :      p1 = data | led;
103 :  }
104 :
105 :  /************************************************************************/
106 :  /* end of file                                                          */
107 :  /************************************************************************/
```

19.5 プログラムの解説

19.5.1 init関数(I/Oポートの入出力設定)

ポート 0 にはセンサ部が接続されています。bit7～5 は LED で出力、bit4 はマイクロスイッチで入力、bit3～0 はフォトインタラプタで入力に設定します。

```
 72 :      /* ポートの入出力設定 */
 73 :      prc2 = 1;                    /* PD0のプロテクト解除          */
 74 :      pd0 = 0xe0;                  /* 7-5:LED 4:SW 3-0:アナログ電圧*/
 75 :      p1  = 0x0f;                  /* 3-0:LEDは消灯                */
 76 :      pd1 = 0xdf;                  /* 5:RXD0 4:TXD0 3-0:LED        */
 77 :      pd2 = 0xfe;                  /* 0:PushSW                     */
 78 :      pd3 = 0xfb;                  /* 4:Buzzer 2:IR                */
 79 :      pd4 = 0x83;                  /* 7:XOUT 6:XIN 5-3:DIP SW 2:VREF*/
 80 :      pd5 = 0x40;                  /* 7:DIP SW                     */
 81 :      pd6 = 0xff;                  /* LEDなど出力                  */
```

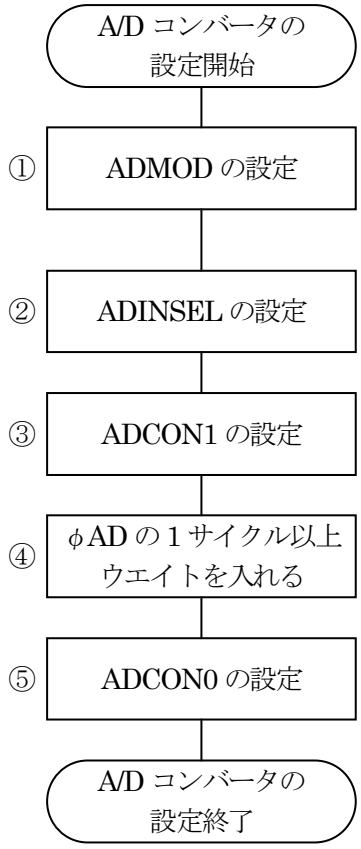
19.5.2 init関数(A/Dコンバータの設定)

A/D コンバータを設定するプログラムは、次のようになります。

```
83 :      /* A/Dコンバータの設定 */
84 :      admod    = 0x33;                /* 繰り返し掃引モードに設定      */
85 :      adinsel = 0x30;                 /* 入力端子P0の8端子を選択       */
86 :      adcon1   = 0x30;                /* A/D動作可能                   */
87 :      asm( " nop ");                  /* φADの1サイクルウエイト入れる*/
88 :      adcon0   = 0x01;                /* A/D変換スタート               */
```

今回は、ポート 0 の 8 本の端子からアナログ電圧を読み込み、繰り返し A/D 変換する設定(**繰り返し掃引モード**)にします。ポート 0 から AD 値を読み込めるのは、フォトインタラプタが接続されている P0_0(AN7)～P0_3(AN4)の 4 端子です。残り 4 端子は読み込んでも値は不定です。

レジスタの設定手順を下記に示します。



A/D モードレジスタの設定をします。
A/D の動作モードなどを設定します。**今回は繰り返し掃引モードに設定します。**

A/D 入力選択レジスタの設定をします。
どの端子をアナログ入力端子にするか設定します。

A/D 制御レジスタ 1 の設定をします。
A/D を動作可能にします（A/D 変換は開始されません)。

A/D の動作を開始した後、φAD の 1 サイクル以上のウエイトを入れます。

A/D 制御レジスタ 0 の設定をします。
A/D 変換を開始します。

①A/D モードレジスタ(ADMOD:A-D mode register)の設定

設定 bit	上:ビット名 下:シンボル	内容	今回の内容
bit7,6	A/D 変換トリガ選択ビット bit7: adcap1 bit6: adcap0	00:ソフトウェアトリガ(ADCON0 レジスタの ADST ビット)による A/D 変換開始 01:タイマ RD からの変換トリガによる A/D 変換開始 10:タイマ RC からの変換トリガによる A/D 変換開始 11:外部トリガ(ADTRG)による A/D 変換開始 A/D 変換を開始するきっかけをどれにするか設定します。ソフト的に開始するので、"00"を選択します。	00
bit5～3	A/D 動作モード選択 bit5: md2 bit4: md1 bit3: md0	000:単発モード 001:設定しないでください 010:繰り返しモード 0 011:繰り返しモード 1 100:単掃引モード 101:設定しないでください 110:繰り返し掃引モード 111:設定しないでください 今回は、繰り返し掃引モードを選択します。	110
bit2	クロック源選択ビット cks2	0:f1(20MHz)を選択 1:fOCO-F(高速オンチップオシレータ)を選択 f1 を選択します。	0
bit1,0	分周選択ビット bit1: cks1 bit0: cks0	00:fAD の 8 分周 (8/20MHz=400ns) 01:fAD の 4 分周 (4/20MHz=200ns) 10:fAD の 2 分周 (2/20MHz=100ns) 11:fAD の 1 分周 (1/20MHz=50ns) fAD とは、bit2 で設定したクロック源のことです。このクロックを何分周で使用するか選択します。遅くする必要はないので、いちばん速い 1 分周で使用します。	11

A/D モードレジスタ(ADMOD)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**1**
16 進数	3				3			

②A/D 入力選択レジスタ(ADINSEL:A-D input select register)
どのアナログ入力端子を A/D 変換するか、設定します。

A/D 入力グループ選択ビット		A/D 掃引端子数選択ビット		bit3	bit2	bit1	bit0	アナログ入力端子
bit7	bit6	bit5	bit4					
0	0	0	0	0	0	0	0	AN0(P0_7)～AN1(P0_6)の 2 端子
0	0	0	1	0	0	0	0	AN0(P0_7)～AN3(P0_4)の 4 端子
0	0	1	0	0	0	0	0	AN0(P0_7)～AN5(P0_2)の 6 端子
0	0	1	1	0	0	0	0	AN0(P0_7)～AN7(P0_0)の 8 端子
0	1	0	0	0	0	0	0	AN8(P1_0)～AN9(P1_1)の 2 端子
0	1	0	1	0	0	0	0	AN8(P1_0)～AN11(P1_3)の 4 端子

※それ以外は設定禁止

今回は、AN0(P0_7)～AN7(P0_0)の 8 端子を選択します。A/D 入力選択レジスタ(ADINSEL)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	**0**	**0**	**1**	**1**	0	0	0	0
16 進数	3				0			

③A/D 制御レジスタ 1(ADCON1:A-D control register1)

A/D を動作可能にします。

設定 bit	上:ビット名 下:シンボル	内容	今回の内容
bit7	A/D 断線検出アシスト方式選択ビット(注 4)	0:変換前ディスチャージ 1:変換前プリチャージ A/D 断線検出アシストしませんのでどちらでも構いませんが、今回は"0"にしておきます。	0
bit6	A/D 断線検出アシスト機能許可ビット(注 4)	0:禁止 1:許可 A/D 断線検出アシストは使いません。	0
bit5	A/D スタンバイビット(注 3) adstby	0:A/D 動作停止(スタンバイ) 1:A/D 動作可能 A/D 動作可能にして A/D 変換できるようにします。この bit を"0"から"1"にしたときは、φA/D の 1 サイクル以上経過した後に A/D 変換を開始します。	1
bit4	8/10 ビットモード選択ビット bits	0:8 ビットモード 1:10 ビットモード A/D 変換を 10bit(0～1023)にするか、8bit(0～255)にするか選択します。今回は、10bit にします。	1
bit3～1		"000"を設定	000
bit0	拡張アナログ入力端子選択ビット(注 1) adex0	0:拡張アナログ入力端子を非選択 1:チップ内蔵基準電圧を選択(注 2) 拡張アナログ入力端子は使いません。	0

注 1. チップ内蔵基準電圧をアナログ入力として使用する場合、ADEX0 ビットを"1"(チップ内蔵基準電圧を選択)にした後に、OCVREFCR レジスタの OCVREFAN ビットを"1"(チップ内蔵基準電圧とアナログ入力を接続)にしてください。また、チップ内蔵基準電圧をアナログ入力として使用しない場合、OCVREFAN ビットを"0"(チップ内蔵基準電圧とアナログ入力を切断)にした後に、ADEX0 ビットを"0"(拡張アナログ入力端子を非選択)にしてください。

注 2. 単掃引モード、繰り返し掃引モードでは設定しないでください。

注 3. ADSTBY ビットを"0"(A/D 動作停止) から"1"(A/D 動作可能) にしたときは、φAD の 1 サイクル以上経過した後に A/D 変換を開始してください。

注 4. A/D 断線検出アシスト機能を許可にするためには、ADDDAEN ビットを"1"(許可)にした後、ADDDAEL ビットで変換開始状態を選択してください。断線時の変換結果は、外付け回路によって変化します。本機能はシステムに合わせた評価を十分に行った上で、使用してください。

A/D 制御レジスタ 1(ADCON1)の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	**1**	**1**	0	0	0	0
16 進数	**3**				**0**			

④φAD の 1 サイクル以上ウエイトを入れる

③の bit5 の A/D スタンバイビットを"1"にした場合、φA/D の 1 サイクル以上経過した後に A/D 変換を開始しなければいけません。
そのウエイトを入れるため、アセンブリ言語の nop 命令を実行します。「ad.c」内では、アセンブリ言語は実行できないため、asm 命令というアセンブリ言語を実行できる命令を使って nop 命令を実行します。ちなみに、nop は「No Operation（何もしない）」命令で、この命令を実行するのに 1 サイクル分の時間がかかります。
プログラムを下記に示します。

```
asm( " nop ");
```

⑤A/D 制御レジスタ 0（ADCON0：A-D control register0）

A/D 変換を開始します。

設定 bit	上：ビット名 下：シンボル	内容	今回の内容
bit7～1		"0000000"を設定	0000000
bit0	A/D 変換開始フラグ adst	0：A/D 変換停止 1：A/D 変換開始 A/D 変換を開始させるので"1"を設定します。	1

A/D 制御レジスタ 0（ADCON0）の設定値を下記に示します。

bit	7	6	5	4	3	2	1	0
設定値	0	0	0	0	0	0	0	**1**
16 進数	**0**				**1**			

19.5.3 main関数

AN7 端子、AN6 端子から A/D 値を取得、マイコンボード上の LED と実習基板の LED へ値を出力します。

```
 38 :  void main( void )
 39 :  {
 40 :      int ad7data, ad6data;
 41 :
 42 :      init();                             /* 初期化                          */
 43 :
 44 :      while( 1 ) {
 45 :          // AD7を出力
 46 :          ad7data = ad7;                  /* AD7取得                         */
 47 :          ad7data = ad7data >> 6;
 48 :          led_out( ad7data );             /* マイコンボードのLEDへ出力       */
 49 :
 50 :          // AD6を出力
 51 :          ad6data = ad6;                  /* AD6取得                         */
 52 :          ad6data = ad6data >> 6;
 53 :          p6 = ad6data;                   /* P6のbit3～0のLEDへ出力          */
 54 :      }
 55 :  }
```

46 行	AD7 端子(P0_0)の A/D 変換値を取得し、ad7data 変数に格納します。
47 行	A/D 変換値は、0～1023(2 進数で 11 1111 1111)の値です。右シフトを 6 ビット分行い、下位の 6 桁を捨てます。その結果、A/D 値は 0～15 の値になり、ad7data 変数に代入します。
48 行	0～15 に変換した A/D 値をマイコンボード上の LED に出力します。
50 行	AD6 端子(P0_1)の A/D 変換値を取得し、ad6data 変数に格納します。
51 行	A/D 変換値は、0～1023(2 進数で 11 1111 1111)の値です。右シフトを 6 ビット分行い、下位の 6 桁を捨てます。その結果、A/D 値は 0～15 の値になり、ad6data 変数に代入します。
52 行	0～15 に変換した A/D 値を実習基板の LED に出力します。

※A/D 変換値を取得するレジスタ

A/D 変換された結果は、A/D レジスタ 0～7(AD0～AD7)に格納されます。AD0～AD7 のどのレジスタに格納されるかは、アナログ入力端子によって変わります。アナログ入力端子と A/D レジスタの関係を次に示します。

アナログ入力端子	読み込むレジスタ
AN0(P0_7)	AD0
AN1(P0_6)	AD1
AN2(P0_5)	AD2
AN3(P0_4)	AD3
AN4(P0_3)	AD4
AN5(P0_2)	AD5
AN6(P0_1)	AD6
AN7(P0_0)	AD7
AN8(P1_0)	AD0
AN9(P1_1)	AD1
AN10(P1_2)	AD2
AN11(P1_3)	AD3

19.6 演習

(1) AN5 の A/D 変換値をマイコンボードの LED へ、AN4 の A/D 変換値を実習基板の LED 部へ出力しなさい。

(2) マイコンボードのディップスイッチの値 0～7 によって、実習基板の LED 部へ AN0～AN7 の A/D 変換値を出力するようにしなさい。