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;
```

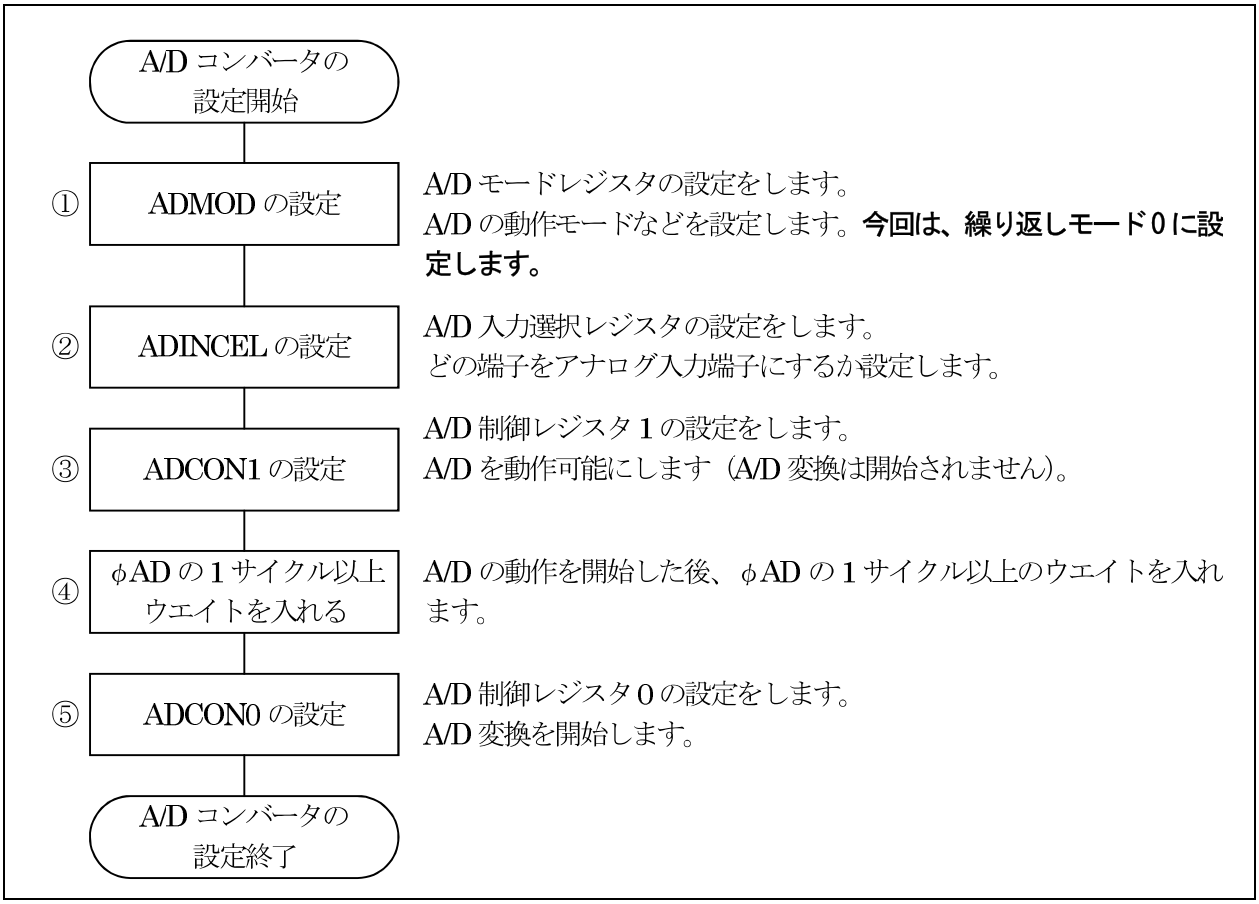
ポート 0 にセンサ部を接続している場合は、P0_4 はマイクロスイッチ、P0_3～P0_1 はセンサが繋がっているので入力にします。実習基板などを使ってこれらの端子が未接続の場合は、出力にしてください。

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。

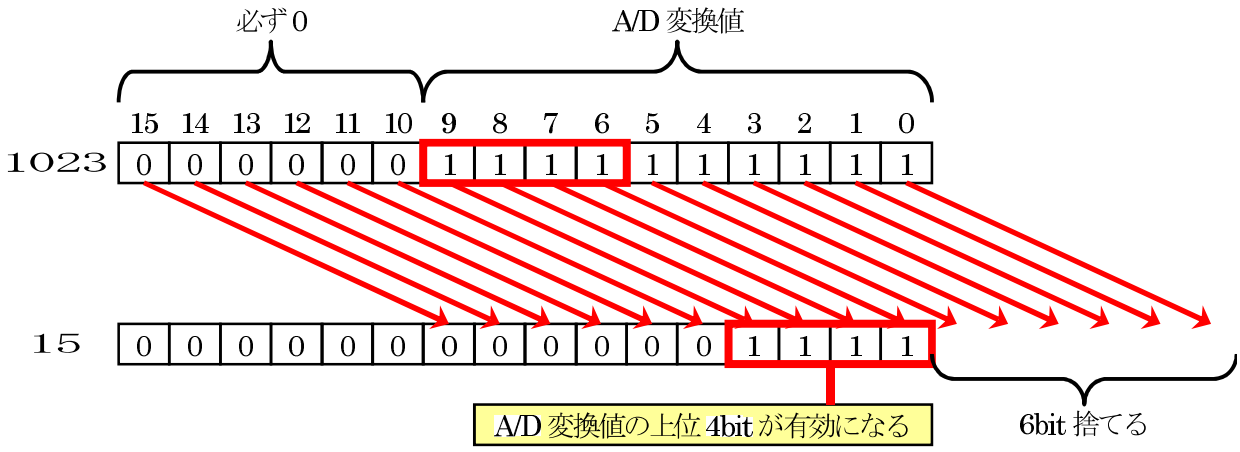
18.5.4 main関数

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。