＜リスト１＞ AD コンバータへのデータの出力例

●NEC PC98 用

```
mov al，22h        // CLK -> 1 , CS -> 0 , DI -> 0
out 32h,al         // シリアル・ポートにデータ出力
```

●PC/AT 互換機用

```
mov al,00h         // CLK -> 1
mov dx，3FBh
out dx al          // シリアル・ポートにデータ出力
mov al,03h         // CS -> 0 , DI -> 0
mov dx,3FCh
out dx,al          // シリアル・ポートにデータ出力
```

＜リスト２＞ A-D コンバータからの１ビットの出力データを調べる方法

●NEC PC98 用

```
in 32h,al          // シリアル・ポートからデータの読み取り
test al,80h        // D7 ビットを調べる
```

●PC/AT 互換機

```
mov dx，3FEh
in dx al           // シリアル・ポートからデータの読み取り
test al,20h        // D5 ビットを調べる
```

＜リスト３＞ A-D 変換アダプタ作動関数（DLL タイプ）

●NEC PC98 用（このリストには，後述の説明のため行番号を入れてあります。実際には入りません。）

```
 1: library adcpc98;
 2: uses
 3:    SysUtils,
 4:    Classes;
 5: const                        //CLK   CS   DI
 6: ADSW011              =$08  ;//   0     1    1
 7: ADSW101              =$20  ;//   1     0    1
 8: ADSW001              =$28  ;//   0     0    1
 9: ADSW100              =$22  ;//   1     0    0
10: ADSW000              =$2a  ;//   0     0    0

11: function  Keisoku(Chan:Word;mati1:Word;mati2:Word):Word;stdcall;
12: var
13: bufdata:Word;
14: ChSw1,ChSw2:Byte;
15: label WAIT1,WAIT2,MEJ,DATA0,DATA1,DATA2;
```

```
16:  begin
17:  if Chan=0 then begin ChSw1:=ADSW000;ChSw2:=ADSW100; end
18:              else begin ChSw1:=ADSW001;ChSw2:=ADSW101; end;//CH の選択
19:   asm
20:       mov      al,ADSW011
21:       out      32h,al          //表5の①
22:     mov        al,ADSW001
23:       out      32h,al          //表5の②
24:       mov      cx,mati1        //表5のWait1
25:  WAIT1:
26:       dec      cx
27:       jns      WAIT1
28:       mov      al,ADSW101
29:       out      32h,al          //表5の③
30:       mov      al,ADSW001
31:       out      32h,al          //表5の④
32:       mov      al,ADSW101
33:       out      32h,al          //表5の⑤
34:     mov      al,ChSw1
35:       out      32h,al          //表5の⑥
36:       mov      al,ChSw2
37:       out      32h,al          //表5の⑦
38:       mov      al,ADSW000
39:       out     32h,al           //表5の⑧
40:       mov      al,ADSW100
41:       out      32h,al          //表5の⑨
42:       mov      cx,mati2        //表5のWait2
43:  WAIT2:
44:       dec      cx
45:       jns      WAIT2
46:       mov      dl,07h          //for i=0 to7
47:  MEJ:
48:       mov      al,ADSW000
49:       out     32h,al           //表5の⑩
50:       mov      al,ADSW100
51:       out      32h,al          //表5の⑪
52:       in      al,32h           //表5の⑫
53:       push     ax              //１ビットのデータをスタックへ入れます。
54:       dec      dl
55:       jns      MEJ             //next i,以上８回入力を行います。
//８ビットの計測データをメモリへ入れます。
56:     mov      dx,00ffh         //dx レジスタに計測データを入れます。
```

```
57:       mov     cx,07h          // for i=0 to 7
58:  DATA0:
59:       pop     ax              //スタックから１ビットのデータを取り出します。
60:       test    al,80h          //al レジスタの D7 が0か1かを調べます。
61:       je      DATA1
62:       ror     dl,1h           //al レジスタの D7 が1であれば dl レジスタの D7 に1を入れます。
63:       jmp     DATA2
64:  DATA1:
65        shr     dl,1h           //al レジスタの D7 が0であれば dl レジスタの D7 に0を入れます。
66:  DATA2:
67:       dec     cx              //next i,以上８回繰り返します。
68:       jns     DATA0
69:     mov     bufdata,dx    //dx レジスタの値を変数 bufdata に入れるます。
70:  end;//asm
71:  Keisoku:=bufdata;         //デジタル変換された８ビットのデータをこの関数の戻り値とします。
72:  end; //main

73:  exports Keisoku index 1; //外部からこの関数を参照できるようにします。

74:  begin
75:  end.
```

●PC/AT 互換機用

```
library adcdosv;
uses
  SysUtils,
  Classes;
const
PCLK   =$3FB;
PCSDI =$3FC;
PDO   =$3FE;
CLK0=$40   ;
CLK1=$00   ;
CSDI00=$3  ;
CSDI01=$2  ;
CSDI10=$1  ;
CSDI11=$0  ;
function   Keisoku(Chan:Word;mati1:Word;mati2:Word):Word;stdcall;
var
bufdata:Word;
ChSw:Byte;
label WAIT1,WAIT2,MEJ,DATA0,DATA1,DATA2;
```

```
begin
if Chan=0 then ChSw:=CSDI00 else ChSw:=CSDI01;
 asm
          mov dx,PCLK;mov al,CLK0;out dx,al
        mov dx,PCSDI;mov al,CSDI11;out dx,al   //表5の①
          mov dx,PCSDI;mov al,CSDI01;out dx,al   //表5の②
          mov     cx,mati1
WAIT1:
          dec     cx
          jns     WAIT1
        mov dx,PCLK;mov al,CLK1;out dx,al     //表5の③
        mov dx,PCLK;mov al,CLK0;out dx,al     //表5の④
        mov dx,PCLK;mov al,CLK1;out dx,al     //表5の⑤
        mov dx,PCLK;mov al,CLK0;out dx,al     //表5の⑥
        mov al,ChSw
        mov dx,PCLK;mov al,CLK1;out dx,al     //表5の⑦
        mov dx,PCLK;mov al,CLK0;out dx,al     //表5の⑧
        mov dx,PCSDI;mov al,CSDI00;out dx,al
        mov dx,PCLK;mov al,CLK1;out dx,al     //表5の⑨
          mov     cx,mati2
WAIT2:
          dec     cx
        jns     WAIT2
          mov     ah,07h                  //for i=0 to7
MEJ:
        mov dx,PCLK;mov al,CLK0;out dx,al     //表5の⑩
        mov dx,PCLK;mov al,CLK1;out dx,al     //表5の⑪
        mov dx,PDO
        in al,dx                              //表5の⑫
          push    ax                          //１ビットのデータをスタックへ入れます。
          dec     ah
          jns     MEJ                       //next i,以上８回入力を行います。
//８ビットの計測データをメモリへ入れます。
   ここに，NEC PC98 用の行番号 56～行番号 69 の命令を入れます。
   ただし，60:test al,80h は，test al,20h に変更します。
end;//asm
Keisoku:=bufdata;
end; //main

exports Keisoku index 1;

begin
```

end.

＜リスト４＞　シリアル・ポートのオープン関数（EXE タイプ）

```
unit Ad_swtch;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls;
type
  TForm1 = class(TForm)
    AD_ON: TButton;
    Switch: TLabel;
    procedure AD_ONClick(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private 宣言 }
   FHandle:THandle;
  public
    { Public 宣言 }
  end;
var
  Form1: TForm1;
implementation
{$R *.DFM}

procedure TForm1.AD_ONClick(Sender: TObject);
begin
FHandle:=CreateFile(
'COM1',                                  //address of name of the file
GENERIC_READ or GENERIC_WRITE,   //access (read-write) mode
0,                                          //share mode
nil,                                         // address of security descriptor
OPEN_EXISTING,                          //how to create
FILE_ATTRIBUTE_NORMAL,              //file attributes
0                                           // handle of file with attributes to copy
);//パラメータの詳細はWin32 Programmer's Reference，文献６をご覧ください。
  Switch.caption:='ON';
end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 CloseHandle(FHandle);
end;
```

end.

＜リスト５＞　VBA で A-D 変換アダプタの作動関数 Keisoku を利用するプログラム

●1 チャンネル入力の場合

```
Declare Function Keisoku Lib "adcpc98" (ByVal C As Integer,ByVal N As Integer, ByVal M As Integer) As Integer　'NEC PC 用
'Declare Function Keisoku Lib "adcdosv" (ByVal C As Integer,ByVal N As Integer, ByVal M As Integer) As Integer　'PC/AT 互換機用

Sub 計測()
    Dim MejData(256) As Integer
    Dim counter As Integer
    Dim Mejcounter As Integer

    Mejcounter = 256　'必ず MejData(X)のXより小さい値を指定してください。
    For counter = 1 To Mejcounter
        MejData(counter) = Keisoku(0, 0, 0)　'CH0 からの入力信号を測定
Next counter
    For counter = 1 To Mejcounter
         Worksheets("Sheet1").Cells(counter,1).Value = MejData(counter)
            'Sheet1 の１列目に計測データが表示されます。
    Next counter
End Sub
```

●2チャンネル入力の場合

```
Declare Function Keisoku Lib "adcpc98" (ByVal C As Integer,ByVal N As Integer, ByVal M As Integer) As Integer　'NEC PC 用
'Declare Function Keisoku Lib "adcdosv" (ByVal C As Integer,ByVal N As Integer, ByVal M As Integer) As Integer　'PC/AT 互換機用

Sub 計測()
    Dim MejData0(256) As Integer
    Dim MejData1(256) As Integer
    Dim counter As Integer
    Dim Mejcounter As Integer

    Mejcounter = 256　'必ず，MejData0(X)，MejData1(X)のXより小さい値を指定してください。
    For counter = 1 To Mejcounter
        MejData0(counter) = Keisoku(0, 0, 0)　'CH0 からの入力信号を測定
        MejData1(counter) = Keisoku(1, 0, 0)　'CH1 からの入力信号を測定
```

```
    Next counter
    For counter = 1 To Mejcounter
        Worksheets("Sheet1").Cells(counter, 1).Value = MejData0(counter)
        Worksheets("Sheet1").Cells(counter, 2).Value = MejData1(counter)
          ' CH0 からの計測データを Sheet1 の１列目に表示
          ' CH1 からの計測データを Sheet1 の２列目に表示
    Next counter
End Sub
```

＜リスト６＞　タイムスタンパを利用する方法

```
function Sampling:Double;          //戻り値は，経過時間を ms の単位で返します。
var
TimeS,TimeE:Word;
TimeMej:Integer;
begin
 asm
 in ax,005ch                       //タイムスタンパのカウントを読みます。
 mov TimeS,ax
{時間計測を行いたい部分}
 in ax,005ch                       //もう一度タイムスタンパのカウントを読みます。
 mov TimeE,ax
 end;

 TimeMej:=TimeE-TimeS;             //カウントの差を出します。
   if TimeMej<0 then
     TimeMej:=TimeMej+ 65536;    //カウントが ffffh から 0000h に変化した場合
 Sampling:=TimeMej*0.00326;      //１カウント当たり 0.00326ms の時間経過です。
 end;
```

＜リスト７＞　サンプリング時間の計測を含んだ A-D 変換アダプタ作動関数（DLL タイプ）

```
library adcpc98t;
uses
  SysUtils,
  Classes;
const
ADSW011                 =$08  ;//  CLK=0 CS=1 DI=1
ADSW101                 =$20  ;//       1     0     1
ADSW001                 =$28  ;//       0     0     1
ADSW100                 =$22  ;//       1     0     0
ADSW000                 =$2a  ;//       0     0     0
type
```

```
    PDouble = ^double;
    PWord    = ^Word;
procedure   Keisoku_time(Data:PWord;Time:PDouble;count:Word;Chan:Word;mati1:Word;mati2:Word);stdcall;
var
i:Word;
bufdata:Word;
timecount:Word;
TimeMej:Integer;
ChSw1,ChSw2:Byte;
TimeData : array[0..1024] of Word;
label WAIT1,WAIT2,MEJ,DATA0,DATA1,DATA2;
begin
if Chan=0 then begin ChSw1:=ADSW000;ChSw2:=ADSW100; end
          else begin ChSw1:=ADSW001;ChSw2:=ADSW101; end;
//ここに,  asm cli end;を入れると取りこぼしなく計測データが取り込めますが大変危険な方法です。
for i:=1 to count do
begin
asm
ここに，リスト２の NEC PC98 用の行番号 20～行番号 69 の命令を入れます。
  in ax,005ch                 //タイムスタンパのカウントを読み出します。
  mov timecount,ax
end;//asm
Data^:=bufdata;               //得られた８ビットの計測データを変数 Data に入れます。
inc(Data);
TimeData[i]:=timecount;    //タイムスタンパのカウントを変数 TimeData に入れます。
end;//for next

//もし，割り込み禁止を入れた場合は，ここに asm sti end;を入れます。

TimeData[0]:=timeData[1];
for i:=1 to count do          //以下，計測データ一つ一つのサンプリング時間を求めます。
begin
 TimeMej:=TimeData[i]-TimeData[i-1];
 if TimeMej<0 then
  TimeMej:=TimeMej+ 65536;
 Time^:= TimeMej*0.00326;   //サンプリング時間を変数 Time に入れます。
 inc(Time);
end;

end; //main

exports Keisoku_time index 1;
```

```
begin
end.
```

＜リスト８＞　VBA で A-D 変換アダプタの作動関数 Keisoku を利用するプログラム

```
Declare Sub Keisoku_time Lib "adcpc98t" (ByRef D As Integer, ByRef T As Double, ByVal C As Integer, ByVal
H As Integer, ByVal N As Integer, ByVal M As Integer)
Sub 計測()
    Dim Data(1024) As Integer
    Dim time(1024) As Double
    Dim count As Integer
    Dim Mejcount As Integer
    Mejcount=1024                    ’Data(X)，time(X)のXより小さい値を指定してください。
    Call Keisoku_time(Data(1), time(1), Mejcount, 0, 0, 0)’CH0 からの入力信号を 1024 個測定します。

    For count = 1 To Mejcount
    Worksheets("Sheet1").Cells(count + 1, 1).Value = time(count)
    Worksheets("Sheet1").Cells(count + 1, 2).Value = Data(count)
    ’サンプリング時間を Sheet1 の１列目，計測データを Sheet1 の２列目に表示します。
    Next count
End Sub
```