```
8051 Cross-Assembler (1.3)    (C) 1987, 1989 Binary Technology
D:\8031\PROJ\RESO11.ASM

                        1       ; modified MAY52.ASM ,for one adc,work with de
                        2       ; delete disp1 and bcdsub out..to work with th
    0050=               3       binum   equ 50h
    0051=               4       bcdhi   equ 51h
    0052=               5       bcdlo   equ 52h
    0053=               6       d1_3    equ 53h
    0054=               7       d1_2    equ 54h
    0055=               8       d1_1    equ 55h
    0056=               9       d1      equ 56h
    0057=              10       binum1  equ 57h
    0058=              11       binum2  equ 58h
    0001=              12       rflag bit 20h.1
    0000=              13       sflag bit 20h.0
    0002=              14       aflag bit 20h.2
                       15
    0000               16               org 0000h
    0000 802F          17               sjmp main
    0003               18               org 0003h
    0003 0200F8        19               ljmp adc
    000B               20               org 000bh
    000B 8019          21               sjmp timer
    0023               22               org 0023h
    0023 020114        23               ljmp serial
                       24
    0026               25       timer:
    0026 8E8A          26               mov tl0,r6
    0028 8F8C          27               mov th0,r7
                       28       ;        mov p1,#0ffh
    002A D2B3          29               setb p3.3
    002C C2B3          30               clr p3.3
    002E D2B3          31               setb p3.3
    0030 32            32               reti
                       33
    0031               34       main:
                       35            ;start initialization----------
    0031 758130        36              mov sp,#30h
    0034 758925        37              mov tmod,#25h     ;init timer1 mode 2,ti
    0037 7588DE        38              mov tcon,#0deh     ;init timer control +
    003A 75A881        39              mov ie,#081h       ;enable external int0
    003D 759850        40              mov scon,#50h;
    0040 758700        41              mov pcon,#00h
    0043 7EF0          42              mov r6,#0f0h       ;for t-low
    0045 7FFF          43              mov r7,#0ffh       ;for t-high
    0047 8E8A          44              mov tl0,r6
    0049 8F8C          45              mov th0,r7
    004B 758BFD        46              mov tl1,#0fdh
    004E 758DFD        47              mov th1,#0fdh
    0051 7590FF        48              mov p1,#0FFh      ;set p1 for triggering
    0054 C291          49              clr p1.1          ;ready signal
    0056 D2AF          50              setb ea           ;interrupt enable
    0058 D28E          51              setb tr1          ;start timer1
    005A D2AC          52              setb es           ;enable serial interru
                       53            ;end of initialization----------
    005C 1179          54              acall connect
    005E               55            wait1:
    005E C291          56              clr p1.1          ;set ready signal
    0060 119B          57              acall get_time
    0062 D291          58              setb p1.1            ;clear ready signal
```

```
8051 Cross-Assembler (1.3)    (C) 1987, 1989 Binary Technology
D:\8031\PROJ\RESO11.ASM

0064                    wait2:
0064 C292               60          clr p1.2            ;set hammer signal
0066 11B4               61          acall get_data
0068 D292               62          setb p1.2           ;clear hammer
006A C293               63          clr p1.3            ;send data signal
006C 11D8               64          acall send_data
006E D293               65          setb p1.3           ;finish sending signa
0070 C201               66          clr rflag   ;wait for repeat signal
0072 3001FD             67          jnb rflag,$ ;wait
0075 C201               68          clr rflag   ;get signal to change time
0077 80EB               69          sjmp wait2  ;repeat
                        70
                        71      ;-----------------------------------------
0079                    72      connect:
0079 C0D0               73            push psw
007B C0E0               74            push acc
                        75            ; wait for signal from PC where 255 is
007D                    76          wait0:
007D C201               77            clr rflag
007F 75507B             78            mov binum,#07bh
0082                    79          wait:
0082 3001FD             80            jnb rflag,$  :for 255 from PC
0085 A850               81            mov r0,binum
0087 B8FF0A             82            cjne r0,#0ffh,noteq
008A 75580A             83            mov binum2,#0ah  ;send back A to acknow
008D 312F               84            acall send
008F D0E0               85            pop acc
0091 D0D0               86            pop psw
0093 22                 87            ret  ; sjmp wait1
0094                    88          noteq:
0094 7558FF             89            mov binum2,#0ffh ;send $ff to PC
0097 312F               90            acall send  ;asking for ready seignal (
0099 80E7               91            sjmp wait
                        92      ;-----------------------------------------
009B                    93      get_time:
009B C0D0               94           push psw
009D C0E0               95           push acc
009F C201               96           clr rflag   ;wait for tl0 & th0
00A1 3001FD             97           jnb rflag,$ ;wait !!!
00A4 C201               98           clr rflag   ;get tl0
00A6 AE50               99           mov r6,binum ;store at r6
00A8 3001FD            100           jnb rflag,$  ;wait for th0
00AB C201              101           clr rflag    ;get it
00AD AF50              102           mov r7,binum ;store at r7
00AF D0E0              103           pop acc
00B1 D0D0              104           pop psw
00B3 22                105           ret
                       106      ;-----------------------------------------
00B4                   107      get_data:
00B4 C0D0              108           push psw
00B6 C0E0              109           push acc
                       110           ;now for adc ----------
                       111      ;     start:
                       112      ;      mov a,#01h           ; trigger signal( h
                       113      ;      anl a,p1             ;
                       114      ;      cjne a,#00h,start    ;
                       115      ;      clr p1.2             ;set signal
00B8 D2A9              116           setb et0
00BA D28C              117           setb tr0        ;start counter 0 , start
```

```
8051 Cross-Assembler (1.3)    (C) 1987, 1989 Binary Technology
D:\8031\PROJ\RESO11.ASM

    00BC C202       118             clr aflag
    00BE 904000     119             mov dptr,#4000h ;store data on RAM
    00C1            120           loop:
    00C1 3002FD     121             jnb aflag,$     ;wait for data
    00C4 C202       122             clr aflag
    00C6 7960       123             mov r1,#60h     ;point at temp. data
    00C8            124           loopsmall:
    00C8 E7         125             mov a,@r1       ;a <-- r1
    00C9 F0         126             movx @dptr,a    ;a --> dptr
    00CA A3         127             inc dptr ;16 bit-inc
    00CB 09         128             inc r1
                    129     ;        cjne r1,#60h,loopsmall  12 ch.
    00CC A883       130             mov r0,dph
    00CE B842F0     131             cjne r0,#042h,loop ;store 512*12=$1800 b
    00D1 C2A9       132             clr et0          ; disable t0
    00D3 D0E0       133             pop acc
    00D5 D0D0       134             pop psw
    00D7 22         135             ret
                    136     ;-----------------------------------------
    00D8            137     send_data:
                    138
    00D8 C0D0       139             push psw
    00DA C0E0       140             push acc
    00DC 904000     141             mov dptr,#4000h  ;to send and display da
    00DF            142           again1:
    00DF E0         143             movx a,@dptr
    00E0 F558       144             mov binum2,a
                    145     ;        mov binum1,a       ;
                    146      ;       mov binum,binum1
                    147         ;    mov binum2,binum1 ;
    00E2 312F       148             acall send
    00E4 A3         149             inc dptr
    00E5 A983       150             mov r1,dph
    00E7 B942F5     151             cjne r1,#042h,again1 ;512*12 = $1800 poi
    00EA D0E0       152             pop acc
    00EC D0D0       153             pop psw
    00EE 22         154             ret
                    155     ;-----------------------------------------
                    156
    00EF C0E0       157     delay: push acc
    00F1 7D0F       158            mov r5,#0fh
    00F3 DDFE       159     L5:    djnz r5,$
    00F5 D0E0       160            pop acc
    00F7 22         161            ret
                    162     ;-----------------------------------------
                    163
    00F8 C0E0       164     adc:    push acc
    00FA C0D0       165             push psw
    00FC C082       166             push dpl
    00FE C083       167             push dph
    0100 7860       168             mov r0,#60h     ;point at data
    0102 90E000     169             mov dptr,#0e000h ;add. toactivate y7(t
    0105            170           more:
    0105 E0         171             movx a,@dptr    ;read 0804
    0106 F6         172             mov @r0,a       ;store at $60++
    0107 08         173             inc r0
    0108 A3         174             inc dptr
                    175     ;        cjne r0,#60h,more read 12 ch.
    0109 D202       176             setb aflag        ;data ready status b
```

```
.8051 Cross-Assembler (1.3)    (C) 1987, 1989 Binary Technology
 D:\8031\PROJ\RESO11.ASM

     010B D083        177                pop dph
     010D D082        178                pop dpl
     010F D0D0        179                pop psw
     0111 D0E0        180                pop acc
     0113 32          181                reti
                      182        ;-----------------------------------------
                      183
     0114 C0D0        184        serial: push psw
     0116 C0E0        185                push acc
     0118 309904      186                jnb ti,nxt    ;is bit transmit set
     011B C299        187                clr ti        ;clear it ,clear status of
     011D C200        188                clr sflag      ;(busy) flag
     011F 309808      189        nxt:    jnb ri,bye    ;if it is not a transmit p
     0122 E599        190                mov a,sbuf    ;it should be a recieve pr
     0124 F550        191                mov binum,a   ;get data
     0126 C298        192                clr ri      ;clear recieve flag
     0128 D201        193                setb rflag
     012A D0E0        194        bye:    pop acc
     012C D0D0        195                pop psw
     012E 32          196                reti
                      197
                      198        ;-----------------------------------------
                      199
     012F C0D0        200        send:   push psw
     0131 C0E0        201                push acc      ;save acc
     0133 E558        202                mov a,binum2
     0135 F599        203        again:  mov sbuf,a    ;send what is in a
     0137 D200        204                setb sflag    ;set flag for busy status
     0139 C201        205                clr rflag     ;clr reciev flag
     013B 2000FD      206        slp:    jb sflag,slp   ;wait for interrupt to ch
     013E 3001FD      207        echo:   jnb rflag,$
     0141 E558        208                mov a,binum2
     0143 B550EF      209                cjne a,binum,again ;check with echo
     0146 D0E0        210                pop acc      ;o.k.
     0148 D0D0        211                pop psw
     014A 22          212                ret
                      213        ;-----------------------------------------
     0000=            214                end
```

8051 Cross-Assembler (1.3) (C) 1987, 1989 Binary Technology
D:\8031\PROJ\RESO11.ASM

adc = 00F8	aflag = 0002	again = 0135
bcdlo = 0052	binum = 0050	binum1 = 0057
connect = 0079	d1 = 0056	d1_1 = 0055
delay = 00EF	echo = 013E	get_data = 00B4
loop = 00C1	loopsmall = 00C8	main = 0031
nxt = 011F	rflag = 0001	send = 012F
sflag = 0000	slp = 013B	timer = 0026
wait1 = 005E	wait2 = 0064	

```
A:\>type reso15.c
/* program reso12.c works with reso11.asm,mod. from reso11.c  Apr.20,95*/
#include <dos.h>
#include <stdio.h>
#include <graphics.h>
#include <conio.h>
#include <string.h>

void port_init(int,int);
void sport(int,int);
char rport(int);
int check_stat(int);
void plotline( int , int , int , int, float );
void ploty(int , int ,int ,int ,int);
void filehandle(int *p,char d);
void display1();
void readdata(int p[]);
        int m,nh,nl,t,d,p[512],nb=512; /* nb is no. of sampling points */
        int x,dx,y,maxx,maxy,x1,y1,average;
        float max,min,dy,sum;
        int graphdriver=DETECT,graphmode;
        float n,time = 100e-6,xt,yt,freq=1E6;
        char s[10],ans,move,disp,dis[15],check;
        int port,i,k,j=0,tl,th;
        char *ch,*idx,pcx[13],r,ch1;
        int xnew,xold,ynew,yold,yold1;
        unsigned size ;
        void *buff1;
        int cmode;
main()
{
        clrscr();
        port=0x00;
        port_init(port,0xe3);   /* 1000 0011 */

        detectgraph (&graphdriver,&graphmode);
        initgraph (&graphdriver,&graphmode,"");

        cmode = getgraphmode();
        maxx=getmaxx();
        maxy=getmaxy();
        max = 255.0;
        min  = 0.0;
        dx = maxx/nb;
        dy = (maxy-10)/max;

        restorecrtmode();

 /*send start signal ( 255) to 8031 and must receive 10 back*/
     do {
        clrscr();
        printf("    type   n    to collect new data \n");
        printf("\n         r    to read data from file \n");
        printf("\n         s    to store data          \n");
        printf("\n         d    to display             \n");
        printf("\n         q    to quit                \n");
        ch1 = getch();
        switch (ch1) {
            case 'n' : clrscr();
                       do {
                         printf(" hit enter to start linking with 8031 \n");
                         getch();
                         sport(port,255);
```

```
                        r = rport(port);
                        sport(port,r);
                        printf(" r = %d \n",r);
                      } while ( r != 10 );
                      clrscr();
                      yt  = time*100000;
                      xt = (65536-yt);
                      th =(int)(xt/256);
                      tl = xt-th*256;
                      sport(port,tl);
                      sport(port,th);
                      readdata(p);
                      display1();
                      break;
            case 'r' : clrscr();
                      filehandle(p,'r');
                /*      printf("before display /n");
                      getch();*/
                      display1();
/*                    printf("after /n");
                      getch();*/
                      break;
            case 's' : clrscr();
                      filehandle(p,'w');
                      break;
            case 'd' : clrscr();
                      display1();
                      break;
            }   /* switch */
        } while (ch1 != 'q');
    closegraph();
}

void readdata (int p[])
{
    int i,k;
    char r;
          printf("\n\n Please wait for seconds !!\n");
          i=0;
             do
               {
                  r = rport(port);
                  sport(port,r);
                  k=r;
                  if(r<0)
                     k=256+r;
                  p[i] = k;
                  i = i+1;
               }
             while ( i <512 );
/*           for(i=0;i<512;i++)
                printf(" %d     %d  \n",i,p[i]);
             getch();*/
}

void display1()
{
   do {
             sum = 0.;
             for(i=0;i<512;i++)
                 sum = sum + p[i];
             average = sum/512;
/*      printf(" average = %d \n",average);
        getch();*/
```

```
setgraphmode(cmode);
cleardevice();
for (i=0,x=0;i<nb;i++,x += dx)
    {
   y= (5.+dy*(max-p[i]));
   printf("i= %d ,x= %d, y= %d,p[i]=%d ,max= %f,dy =%f\n",i,x,y,p
   putpixel(x,y,63);
     }
   setviewport(0,0,maxx,20,0);
   size = imagesize(0,0,maxx,20);
   buff1 = malloc(size);
   getimage(0,0,maxx,20,buff1);
   clearviewport();
   outtext("x-detail ,y-detail y, r-repeat, q-quit");
   disp = getch();
   check = disp;
   if (disp =='x') {
      putimage(0,0,buff1,COPY_PUT);
      setlinestyle(0,0,1);  /*solidln,0,norwidth */
      setwritemode(1);      /* xorput */
      xold = 0;
      setviewport(0,0,maxx,maxy,0);

      moveto(maxx*0.4,0);
      outtext("r-right,l-left,q-quit, time( m-sec) = ");
      x1 = getx();
      y1 = gety();
      outtext(dis);*/
      line(0,0,0,maxy);
      move = getch();
      do {
         switch (move) {
           case 'l' :xnew = xold-1;
                 plotline(xold,xnew,x1,y1,time);
                 xold = xnew;
                 break;
            case 'r' :xnew = xold+1;
                 plotline(xold,xnew,x1,y1,time);
                 xold = xnew;
                 break;
            case 'R'  :xnew = xold + 5;
                 plotline(xold,xnew,x1,y1,time);
                 xold = xnew;
                 break;
            case 'L'  :xnew = xold - 5;
                 plotline(xold,xnew,x1,y1,time);
                 xold = xnew;
                 break;

       }  /* switch */
        move = getch();
       } while (move != 'q');
   } /* if */

   if (disp =='y') {
      putimage(0,0,buff1,COPY_PUT);
      setlinestyle(0,0,1);  /*solidln,0,norwidth */
      setwritemode(1);      /* xorput */
      yold = (5.+dy*(max-average));
      yold1 = yold;
      printf("yold1 = %d , average = %d \n",yold1,average);
      getch();*/
      setviewport(0,0,maxx,maxy,0);
```

```
                    moveto(maxx*0.4,0);
                    outtext("u-up ,d-down ,q-quit , y = ");
                    x1 = getx();
                    y1 = gety();
                    line(0,yold1,getmaxx(),yold1);
                    move = getch();
                    do {
                       switch (move) {
                         case 'u' :ynew = yold-1;
                                  ploty(yold,ynew,x1,y1,yold1);
                                  yold = ynew;
                                  break;
                          case 'd' :ynew = yold+1;
                                  ploty(yold,ynew,x1,y1,yold1);
                                  yold = ynew;
                                  break;
                          case 'U'  :ynew = yold - 10;
                                  ploty(yold,ynew,x1,y1,yold1);
                                  yold = ynew;
                                  break;
                          case 'D'  :ynew = yold + 10;
                                  ploty(yold,ynew,x1,y1,yold1);
                                  yold = ynew;
                                  break;

                       }  /* switch */
                        move = getch();
                       } while (move != 'q');
        }   /*  end if */

        free(buff1);
        restorecrtmode();
        if ( disp == 'r' )
        {
            sport(port,0); /* signal 8031 for repeat */
            readdata(p);
        }

 }while (check == 'r') ;

}

void  plotline(int old, int new, int xx1, int yy1, float time)
{
    char dis[25];
    int sig = 5;
    float num;
    if ( new < 0 || new > getmaxx())
        return(1);
    moveto(xx1,yy1);
    setcolor(0);
    num = old*time*1000;
    gcvt(num,sig,dis);
    outtext(dis);
    setcolor(63);
    line(old,0,old,getmaxy());
    line(new,0,new,getmaxy());
    moveto(xx1,yy1);
    num = new*time*1000;
    gcvt(num,sig,dis);
    outtext(dis);
}

void  ploty(int old, int new, int xx1, int yy1, int old1)
```

```
{
    char dis[25];
    int sig = 5;
    float num;
    if ( new < 0 || new > getmaxx())
        return(1);
    moveto(xx1,yy1);
    setcolor(0);
    num = old1-old;
/*    printf(" num = %f \n",num);
    getch();*/
    gcvt(num,sig,dis);
    outtext(dis);
    setcolor(63);
    line(0,old,getmaxx(),old);
    line(0,new,getmaxx(),new);
    moveto(xx1,yy1);
    num = old1-new;
/*    printf(" num = %f \n",num);
    getch();*/
    gcvt(num,sig,dis);
    outtext(dis);
}

void port_init(int port,int code)
{
        union REGS r;

        r.x.dx=port;
        r.h.ah=0;
        r.h.al=code;
        int86(0x14,&r,&r);
}


void filehandle(int *p,char d)
{
    int i,j;
    char name[10];
    FILE *outfile,*infile;
    if (d== 'w'){
        printf("\n put in your filename to store  ");
        scanf("%s",&name);
        while((outfile = fopen(name,"w"))==0) {
            printf("  give  other filename \n");
            scanf("%s",&name);
        }
        for (i=0;i<nb;i++)
            fprintf(outfile,"%d\n",p[i]);
        fclose(outfile);
        return;
    }

/* read file */

    printf("\n put in filename to read from   ");
    scanf("%s",&name);
    while((infile=fopen(name,"r"))==0) {
        printf(" wrong name ! put in new name  ");
        scanf("%s",&name);
    }
    for (i=0;i<nb;i++)
        fscanf(infile,"%d",&p[i]);
    fclose(infile);
```

```
}

void sport(int port,int c)
{
        union REGS r;

        r.x.dx=port;
        r.h.al=c;
        r.h.ah=1;
        int86(0x14,&r,&r);
        if (r.h.ah & 128)
        {
                printf("\nSend error detected in serial port.\n");
                exit(1);
        }
}

char rport(int port)
{
        union REGS r;

        while (!(check_stat(port)&256))
                if (kbhit())
        {
            getch();
            exit(1);
        }
        r.x.dx=port;
        r.h.ah=2;
        int86(0x14,&r,&r);
        if (r.h.ah & 128)
        printf("Read error detected in serial port");
        return r.h.al;
}

int check_stat(int port)
{
        union REGS r;

        r.x.dx=port;
        r.h.ah=3;
        int86(0x14,&r,&r);
        return r.x.ax;
}
```

A:\>