```
(*************************************************************************)
(*              A MELMITZLOU PROGRAM FOR THE IEEE-488 BUS                *)
(*************************************************************************)

program IEEE488  (input,output,todisk);

uses
    crt, dos, graph,ieeepas ;

const
     fluke_num1 = 1;                    (* FLUKE ADDRESSES *)
     fluke_num2 = 16;
     fluke_num3 = 18;

var
    x1, x2           : integer;
    y1, y2           : integer;
    status           : integer;
    device           : integer;
    todisk           : text;
    poll             : byte;
    instrument_num   : char;
    ch, ans          : char;
    p                : pointer;
    saving_data      : boolean;

(********************** functions and procedures **********************)

(************************ begin function **************************)

function valid_graphics : boolean;

var
   graphdriver    : integer;
   graphmode      : integer;
   errorcode      : integer;

begin

    valid_graphics       := true;
                          (* AUTODETECT OF GRAPH DRIVERS AND GRAPH MODE *)
    detectgraph(graphdriver,graphmode);
    if (graphdriver = ega) or (graphdriver = vga) then
       begin
            graphdriver := cga;
            graphmode   := cgahi;
       end;
    initgraph(graphdriver, graphmode, '');
    errorcode            := graphresult;
    if errorcode <> grok then                          (* ERROR MESSAGES *)
       begin
            clrscr;
            writeln('graphics error: ',grapherrormsg(errorcode));
            writeln('you probably don''t have a graphics card!');
            writeln('program aborted.....');
            valid_graphics := false ;
       end
    else
       initialize(21,0);                          (* INITIALIZE INSTRUMENT *)
       saving_data := false;

end;
```

```
(************************** end function ******************************)

(************************** begin procedure ***************************)

procedure full_screen;

begin

     setviewport(0,0,639,199,clipon);     (* SET VIEWPORT TO CGA DEMINSIONS *)

end;

(************************** end procedure *****************************)

(************************** begin procedure ***************************)

procedure  main_menu(var instr : char);

var
     valid : boolean;

begin                                                  (* DISPLAY MAIN MENU *)

     full_screen;
     clearviewport;
     rectangle(0,0,639,199);
     setlinestyle(dottedln,0,normwidth);
     line(1,30,640,30);
     line(1,35,640,35);
     line(1,165,640,165);
     line(1,170,640,170);
     setlinestyle(solidln,0,normwidth);

     settextstyle(defaultfont,horizdir,2);
     outtextxy( 160,5,'IEEE488 MAIN MENU ');
     settextstyle(defaultfont,horizdir,1);
     outtextxy( 160,45,' A  ==================>  MULTIMETER #1');
     outtextxy( 160,60,' B  ==================>  MULTIMETER #2');
     outtextxy( 160,75,' C  ==================>  MULTIMETER #3');
     outtextxy( 160,90,' D  ==================>  INSTRUMENT #4');
     outtextxy( 160,105,' E  ==================>  INSTRUMENT #5');
     outtextxy( 160,120,' F  ==================>  INSTRUMENT #6');
     outtextxy( 160,135,' G  ==================>  PLOTTER ');
     outtextxy( 160,150,' Q  ==================>  EXIT PROGRAM ');
     outtextxy(150,180,'SELECT LETTER OF DESIRED INSTRUMENT ');
     repeat
          instr := readkey;                              (* CHECK FOR SELECTION *)
          valid  := (instr in [ 'a', 'A', 'b', 'B', 'c', 'C', 'd', 'D',
                                'e', 'E', 'f', 'F', 'g', 'G', 'q', 'Q']);
     until valid;

end;

(************************** end procedure *****************************)

(************************** begin procedure ***************************)

procedure group_logo;

begin                                                  (* DISPAY GROUP LOGO *)

     full_screen;
     settextstyle(defaultfont,horizdir,3);
     outtextxy(295,125,'GROUP 1 INC.');
     settextstyle(defaultfont,horizdir,1);
```

```
      outtextxy(350,155,'a melmitzlou production...');

end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure clr_msg_line;

begin                                                  (* CLEAR MESSAGE LINE *)

     setviewport(90,190,635,198,clipon);
     clearviewport;
     full_screen;

end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure reverse_vidio(x1,y1,x2,y2 : integer; var p : pointer );

var
   size  : word;

begin                                              (* REVERSES BUTTON IMAGE *)

     size := imagesize(x1,y1,x2,y2);
     getmem(p,size);
     getimage(x1,y1,x2,y2,p^);
     putimage(x1,y1,p^,notput);

end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure clear_display;

begin                                              (* CLEAR DISPAY ON METER *)

     setviewport(180,20,500,43,clipon);
     clearviewport;
     full_screen;

end;

(************************** end procedure ******************************)

(***********************************************************************)
(**                                                                   **)
(**                  begin multimeter procedures                      **)
(**                                                                   **)
(***********************************************************************)

(************************** begin function *****************************)

function meter_connected : boolean;

var
   ch  : char;

begin                                  (* CHECKS TO SEE IF DEVICE IS CONNECTED *)
```

```
      meter_connected := true;
      send(device,'',status);
         if status = 8 then
            begin
                 meter_connected := false;
                 clear_display;
                 settextstyle(defaultfont,horizdir,2);
                 outtextxy(185,20,'METER NOT CONNECTED');
                 settextstyle(defaultfont,horizdir,1);
                 clr_msg_line;
                 outtextxy(100,190,'HIT ANY KEY TO CONTINUE');
                 repeat
                      send(device,'',status);
                 until (status <> 8) or (keypressed);
                 clr_msg_line;
            end;
end;

(************************ end function ********************************)

(************************ begin procedure *****************************)

procedure multi_sub_menu(var selec : char);

var
   valid  : boolean;

begin                                                (* DISPAY SUBMENU OPTIONS *)

      setviewport(1,101,249,187,clipon);
      outtextxy(24,3, '1...INITIALIZE');
      outtextxy(24,12,'2...SET OPTIONS');
      outtextxy(24,21,'3...SINGLE TRIGGER');
      outtextxy(24,30,'4...CONTINIOUS TRIGGER');
      outtextxy(24,39,'5...DEVICE CLEAR');
      outtextxy(24,48,'6...SELF TEST');
      if not saving_data then
          outtextxy(24,57,'7...SAVE DATA');
      outtextxy(24,66,'8...DISPLAY FILES');
      outtextxy(24,75,'9...QUIT');
      full_screen;
      clr_msg_line;
      outtextxy(100,190,'SELECT OPERATION YOU WISH...');
      repeat
         selec := readkey;                          (* CHECKS FOR VALID SELECTION *)
         valid := (selec in[ '1', '2', '3', '4', '5', '6', '7', '8', '9']);
      until valid;

end;

(************************ end procedure *******************************)

(************************ begin procedure *****************************)

procedure multi_graph;

begin                                                (* DRAW MULTIMETER TO SCREEN *)

      rectangle(0,0,639,199);
      rectangle(5,3,635,98);
      rectangle(150,3,600,45);
      rectangle(150,49,190,68);
      rectangle(150,74,190,93);
      rectangle(200,49,240,68);
      rectangle(200,74,240,93);
```

```
        rectangle(250,49,290,68);
        rectangle(250,74,290,93);
        rectangle(300,49,340,68);
        rectangle(300,74,340,93);
        rectangle(350,49,390,68);
        rectangle(350,74,390,93);
        rectangle(400,49,440,68);
        rectangle(400,74,440,93);
        rectangle(450,74,490,93);
        rectangle(495,50,530,60);
        rectangle(495,65,530,75);
        rectangle(535,50,570,60);
        rectangle(535,65,570,75);
        rectangle(575,50,610,60);
        rectangle(575,65,610,75);
        rectangle(531,81,575,95);
        rectangle(83,68,117,76);

        circle(50,25,15);
        circle(100,25,15);
        circle(50,47,15);
        circle(100,47,15);
        circle(50,72,15);

        outtextxy(160,7,'FLUKE 8840A MULTIMETER');
        outtextxy(25,10,'INPUTS');
        outtextxy(83,10,'SENSE');
        outtextxy(160,57,'VDC');
        outtextxy(210,57,'VAC');
        outtextxy(260,57,'k'+char(234)+'2');
        outtextxy(310,57,'k'+char(234)+'4');
        outtextxy(362,52,'mA');
        outtextxy(362,60,'dc');
        outtextxy(412,52,'mA');
        outtextxy(412,60,'ac');
        outtextxy(161,76,'200');
        outtextxy(155,85,char(234)+'/mv');
        outtextxy(210,80,'2');
        outtextxy(264,80,'20');
        outtextxy(310,80,'200');
        outtextxy(355,80,'2000');
        outtextxy(412,76,'20');
        outtextxy(412,85,'M'+char(234));
        outtextxy(455,80,'AUTO');
        outtextxy(497,52,'TRIG');
        outtextxy(537,52,'XTRG');
        outtextxy(577,52,'RATE');
        outtextxy(497,67,'OFFS');
        outtextxy(537,67,'LOCL');
        outtextxy(580,67,'SQR');
        outtextxy(535,84,'RESET');
        outtextxy(5,190,'MESSAGE: ');

        setlinestyle(dottedln,0,normwidth);
        line(1,100,640,100);
        line(250,100,250,188);
        line(1,188,640,188);
        setlinestyle(solidln,0,normwidth);

end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure meter_setings(var func : integer; var rang : integer);
```

```
var
   data        : string;
   length      : word;
   max_len     : word;

begin                                          (* UPDATES BUTTON SELECTION *)

      send(device,'g0',status);
      enter(data,max_len,length,device,status);
      y1 := 50;
      y2 := 67;
      case data[1] of
            '1' : begin
                        x1 := 151;
                        x2 := 189;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '2' : begin
                        x1 := 201;
                        x2 := 239;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '3' : begin
                        x1 := 251;
                        x2 := 289;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '4' : begin
                        x1 := 301;
                        x2 := 339;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '5' : begin
                        x1 := 351;
                        x2 := 389;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '6' : begin
                        x1 := 401;
                        x2 := 439;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
         end;
      func := x1;
      y1 := 75;
      y2 := 92;
      case data[2] of
            '1' : begin
                        x1 := 151;
                        x2 := 189;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '2' : begin
                        x1 := 201;
                        x2 := 239;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '3' : begin
                        x1 := 251;
                        x2 := 289;
                        reverse_vidio(x1,y1,x2,y2,p);
                  end;
            '4' : begin
                        x1 := 301;
                        x2 := 339;
```

```
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
            '5' : begin
                    x1 := 351;
                    x2 := 389;
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
            '6' : begin
                    x1 := 401;
                    x2 := 439;
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
            '0' : begin
                    x1 := 451;
                    x2 := 489;
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
          end;
          rang := x1;

end;

(************************ end procedure ********************************)

(************************ begin procedure ******************************)

procedure clear_buttons;

var
   x1, y1  : integer;
   pixelon : word;

begin                                  (* CLEARS BUTTONS AND RETURNS TO DEFAULT *)

     x1 := 151;
     y1 := 50;
     repeat
     pixelon := getpixel(x1,y1);
     if pixelon <> 0 then
        reverse_vidio(x1,y1,x1+38,y1+17,p);
     x1 := x1 + 50;
     until x1 = 451;
     x1 := 151;
     y1 := 75;
     repeat
     pixelon := getpixel(x1,y1);
     if pixelon <> 0 then
        reverse_vidio(x1,y1,x1+38,y1+17,p);
     x1 := x1 + 50;
     until x1 = 501;

end;

(************************ end procedure ********************************0

(************************ begin procedure ******************************)

procedure display_data;

var
    data, readtime : string;
    length         : word;
    max_len        : word;
    ho,mi,se,se100 : word;
    hour,min,sec   : string[2];
```

```
begin                                    (* DISPLAY DATA TO MULTIMETER ON SCREEN *)
     setviewport(0,0,639,199,clipon);
     settextstyle(defaultfont,horizdir,2);
     enter(data,max_len,length,device,status);
     outtextxy(185,20,data);
     if saving_data then
       begin
            gettime(ho,mi,se,se100);
            str(ho,hour);
            str(mi,min);
            str(se,sec);
            readtime := concat(' TIME = ',hour,'-',min,'-',sec);
            writeln(todisk,data,readtime);
       end;
     settextstyle(defaultfont,horizdir,1);

end;

(************************ end procedure ******************************)

(************************ begin function ******************************)

 procedure error_msg ;

 var
      num       : integer ;
      offset    : integer ;
      data      : string ;
      length    : word ;
      max_len   : word ;
 begin                                                (* DISPLAY ERROR MESSAGES *)
      clr_msg_line ;
      offset := ord('0') ;
      enter(data,max_len,length,device,status) ;
      num := ((ord(data[6])-offset)*10)+(ord(data[7])-offset) ;
      case num of
         1 : outtextxy(100,190,' 200 VAC, Zero ') ;
         2 : outtextxy(100,190,' 700 VAC, Zero ') ;
         3 : outtextxy(100,190,' mA AC, Zero ') ;
         4 : outtextxy(100,190,' mA DC, Zero ') ;
         5 : outtextxy(100,190,' 200 VDC, Zero ') ;
         6 : outtextxy(100,190,' 1000 VDC, Zero ') ;
         7 : outtextxy(100,190,' 1000 VDC + 20 M OHM ') ;
         8 : outtextxy(100,190,' 20 VDC + 20 M OHM ') ;
         9 : outtextxy(100,190,' 20 VDC + 2000 k OHM ') ;
         10 : outtextxy(100,190,' 2 VDC + 2000 k OHM ') ;
         11 : outtextxy(100,190,' 200 OHM, Overrange ') ;
         12 : outtextxy(100,190,' 2 k OHM, Overrange ') ;
         13 : outtextxy(100,190,' 20 k OHM, Overrange ') ;
         14 : outtextxy(100,190,' 200 k OHM, Overrange ') ;
         15 : outtextxy(100,190,' 1000 VDC + X10 T/H + 20 M OHM ') ;
         16 : outtextxy(100,190,' 200 VDC + 200 k OHM ') ;
         17 : outtextxy(100,190,' 200 VDC + 20 k OHM ') ;
         18 : outtextxy(100,190,' 200 VDC + 2 k OHM ') ;
         19 : outtextxy(100,190,' 200 VDC, Filter On ') ;
         20 : outtextxy(100,190,' 200 VDC + 2 k OHM, Filter Off ') ;
         21 : outtextxy(100,190,' 200 VDC, Filter Off ') ;
         25 : outtextxy(100,190,' In-Guard uC Internal RAM ') ;
         26 : outtextxy(100,190,' Display RAM ') ;
         27 : outtextxy(100,190,' In-Guard uC Internal ') ;
         28 : outtextxy(100,190,' External Program Memory ') ;
         29 : outtextxy(100,190,' Calibration memory ') ;
         30 : outtextxy(100,190,' AC availiable only with TRUE AC option ') ;
         31 : outtextxy(100,190,'mA AC or mA DC selected while REAR input select
ed') ;
         32 : outtextxy(100,190,'OFFSET selected with reading unavailbe or overr
```

```
ange') ;
          40 : outtextxy(100,190,'Calibration constant out of range') ;
          41 : outtextxy(100,190,'Calibration input out of range, check input') ;
          42 : outtextxy(100,190,'Calibration memory write error') ;
          50 : outtextxy(100,190,'Guard crossing error detected by In-Guard uC')
;
          51 : outtextxy(100,190,'Calibration command not valid until Calibr. mod
e enabled') ;
          52 : outtextxy(100,190,'Command not valid at this time ') ;
          53 : outtextxy(100,190,'Invalid calibration value in PUT command') ;
          54 : outtextxy(100,190,'Command not valid in Calibration verification')
 ;
          56 : outtextxy(100,190,'No Variable inputs allowed with A/D cal., use p
rompted value') ;
          60 : outtextxy(100,190,'Device-dependant commands not valid during self
-tests') ;
          71 : outtextxy(100,190,'Syntax error in Device-dependant command string
') ;
          72 : outtextxy(100,190,'Guard crossing error detected by Out-Guard uC '
) ;
          77 : outtextxy(100,190,'IEEE-488 Interface self-test error ') ;
       end ;        (** case **)
       delay(2000) ;

 end ;

(************************** end function ******************************)

(************************** begin procedure ***************************)

procedure reset_meter;

begin                                          (* RETURN TO DEFAULT SETTINGS *)

     if meter_connected then
        begin
             reverse_vidio(532,82,574,94,p);
             delay(1000);
             clear_buttons;
             clear_display;
             send(device,'*',status);
             reverse_vidio(532,82,574,94,p);
        end;
end;

(************************** end procedure *****************************)

(************************** begin procedure ***************************)

procedure set_options;

var
    x1,x2,y1,y2 : integer;
    funct       : integer;
    range       : integer;
    ch          : char;

begin                                           (* SELECT FUNCTION AND RANGE *)

     if meter_connected then
     begin
        clear_display;
        clear_buttons;
        clr_msg_line;
        meter_setings(funct,range);
        outtextxy(100,190,'PRESS ARROW KEYS '+CHAR(26)+' '+CHAR(27)+
```

```
    ' TO DESIRED FUNCTION THEN <RETURN>');
    setlinestyle(0,0,3);
    x1 := funct;
    x2 := funct + 38;
    y1 := 50;
    y2 := 67;
    reverse_vidio(x1,y1,x2,y2,p);
    reverse_vidio(x1,y1,x2,y2,p);
    repeat
          ch := readkey;
          if ch = char(77) then
          begin
               putimage(x1,y1,p^,normalput);
               if x1 <> 401 then
                  begin
                    x1 := x1 + 50;
                    x2 := x2 + 50;
                  end
               else
                  begin
                    x1 := 151;
                    x2 := 109;
                  end;
               reverse_vidio(x1,y1,x2,y2,p);
          end;
          if ch = char(75) then
          begin
               putimage(x1,y1,p^,normalput);
               if x1 <> 151 then
                  begin
                    x1 := x1 - 50;
                    x2 := x2 - 50;
                  end
               else
                   begin
                    x1 := 401;
                    x2 := 439;
                   end;
               reverse_vidio(x1,y1,x2,y2,p);
          end;
    until ch = char(13);
  case x1 of                                                    (* FUNCTION *)
         151 : send(device, 'f1',status);
         201 : send(device, 'f2',status);
         251 : send(device, 'f3',status);
         301 : send(device, 'f4',status);
         351 : send(device, 'f5',status);
         401 : send(device, 'f6',status);
    end;
       spoll(device,poll,status) ;
       if ((poll and 32) <> 0) then          (* CHECK FOR DEVICE ERRORS *)
          error_msg;

    clr_msg_line;
    outtextxy(100,190,'PRESS ARROW KEYS '+CHAR(26)+' '+CHAR(27)+
    ' TO DESIRED RANGE THEN <RETURN>');
    x1 := range;
    x2 := range + 38;
    y1 := 75;
    y2 := 92;
    reverse_vidio(x1,y1,x2,y2,p);
    reverse_vidio(x1,y1,x2,y2,p);
    repeat
          ch := readkey;
          if ch = char(77) then
          begin
```

```
                    putimage(x1,y1,p^,normalput);
                    if x1 <> 451 then
                      begin
                          x1 := x1 + 50;
                          x2 := x2 + 50;
                      end
                    else
                       begin
                          x1 := 151;
                          x2 := 109;
                       end;
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
                if ch = char(75) then
                begin
                    putimage(x1,y1,p^,normalput);
                    if x1 <> 151 then
                       begin
                          x1 := x1 - 50;
                          x2 := x2 - 50;
                       end
                    else
                       begin
                          x1 := 451;
                          x2 := 409;
                       end;
                    reverse_vidio(x1,y1,x2,y2,p);
                end;
          until ch = char(13);
          clr_msg_line;
          setlinestyle(0,0,1);
      case x1 of                                                    (* RANGE *)
              151 : send(device, 'r7 r1',status);
              201 : send(device, 'r7 r2',status);
              251 : send(device, 'r7 r3',status);
              301 : send(device, 'r7 r4',status);
              351 : send(device, 'r7 r5',status);
              401 : send(device, 'r7 r6',status);
              451 : send(device, 'r0',status);
          end;
             spoll(device,poll,status) ;
             if ((poll and 32) <> 0) then          (* CHECK FOR DEVICE ERRORS *)
                    error_msg;
          end;
end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure single_trigger;

begin                                                  (* TRIGGER A SINGLE READING *)

      if meter_connected then
          begin
             clear_display;
             reverse_vidio(496,51,529,59,p);
             delay(1000);
             putimage(496,51,p^,normalput);
             send(device,'yi ?',status);
             spoll(device,poll,status) ;
             if ((poll and 1) <> 0) then                  (* CHECK FOR OVERFLOW *)
                begin
                  clr_msg_line ;
                  outtextxy(100,190,' OVERFLOW --- SELECT OPTIONS AGAIN ');
```

```
                delay(1000);
              end;
            display_data;
            if ((poll and 32) <> 0) then          (* CHECK FOR DEVICE ERRORS *)
              error_msg;
        end;

end;

(************************** end procedure ********************************)

(************************** begin procedure ******************************)

procedure continious_trigger;

var
   temp          : integer;
   ans           : char;
   valid         : boolean;

begin                                          (* TRIGGER MULTIPLE READINGS *)

            clr_msg_line;
            reverse_vidio(536,51,569,59,p);
            reverse_vidio(576,51,609,59,p);
                                               (* SELECT SPEED OF READINGS *)
            outtextxy(100,190,'ENTER SPEED YOU WISH READING TAKEN AT'+
            ' 1-SLOW, 2-MED, 3-FAST');
            repeat
                ans    := readkey;
                valid  := (ans in ['1', '2', '3']);
            until valid;
            case ans of
                '1' : begin
                      temp := 3000;
                      outtextxy(510,20,' slow');
                      end;
                '2' : begin
                      temp := 1500;
                      outtextxy(510,20,' medium');
                      end;
                '3' : begin
                      temp := 500;
                      outtextxy(510,20,' fast');
                      end;
            end;  (** end case **)
            clr_msg_line;
            outtextxy(100,190,'HIT ANY KEY TO DISCONTINUE READINGS ');
            x1 := 1;
            repeat
                x1 := x1 + 1;
                outtextxy(100,190,'HIT ANY KEY TO DISCONTINUE READINGS ');
                single_trigger;
                delay(temp);
            until (x1 = 14) or (keypressed);
            reverse_vidio(536,51,569,59,p);
            reverse_vidio(576,51,609,59,p);
            setviewport(500,20,590,40,clipon);
            clearviewport;
            clear_display;

end;

(************************** end procedure ********************************)

(************************** begin procedure ******************************)
```

```
procedure save_data(var saving_data : boolean);

var
   ho,mi,se,se100    : word;
   yr,mo,da,num      : word;
   day,hour,min      : string[2];
   file_name         : string;
   ch                : char;
   test              : word;
   testresult        : string[10];

begin                                                    (* SAVE DATA TO FILE *)

     if not saving_data then
        begin
          getdate(yr,mo,da,num);
          gettime(ho,mi,se,se100);
          str(ho,hour);
          str(mi,min);
          str(da,day);
          file_name := concat('C:\DATA\',day,hour,min,instrument_num,'.flk');
          assign(todisk,file_name);
          {$I-}
          rewrite(todisk);
          {$I+}
          test := ioresult;
          if test <> 0 then
             begin
                clr_msg_line;
                str(test,testresult);
                outtextxy(100,190,'error  '+testresult);
                ch := readkey;
                end
          else
            begin
              saving_data := true;
              setviewport(1,101,249,187,clipon);
              reverse_vidio(10,56,230,64,p);
              full_screen;
            end
        end
      else
         begin
              setviewport(1,101,249,187,clipon);
              reverse_vidio(10,56,230,64,p);
              full_screen;
              close(todisk);
              saving_data := false;
         end;

end;

(************************** end procedure **************************)

(************************ begin procedure ************************)

procedure display_files;

var
   x1, y1   : integer;
   ch       : char;
   dirinfo : searchrec;

begin
```

```
      setviewport(251,101,630,187,clipon);
      clearviewport;
      findfirst('c:\data\*.flk',archive,dirinfo);
      x1 := 1;
      y1 := 10;
      while doserror = 0 do
            begin
                 outtextxy(x1,y1,dirinfo.name);
                 y1 := y1 + 10;
                 findnext(dirinfo);
                 if y1 = 70 then
                  begin
                    x1 := x1 + 150;
                    y1 := 10;
                  end;
            end;
     clr_msg_line;
     outtextxy(100,190,' HIT ANY KEY TO CONTINUE');
     ch := readkey;
     setviewport(251,101,630,187,clipon);
     clearviewport;
     group_logo;

end;

(************************** end procedure ******************************)

(************************** begin procedure ****************************)

procedure multi_one;

var
     x1,x2      : integer;
     selected   : char;
     valid      : boolean;

begin                                        (* WAIT FOR SELECTION FROM SUBMENU *)

      clearviewport;
      multi_graph;
      group_logo;
                                             (* DISPLAY INSTRUMENT READING FROM *)
      outtextxy(275,175,'READING INSTRUMENT NUMBER '+instrument_num);
      valid        := false;
      if meter_connected then
        begin
         clear_buttons;
         meter_setings(x1,x2);
         repeat
             multi_sub_menu(selected);
             case selected of
                '1'   :  initialize(21,0) ;
                '2'   :  set_options;
                '3'   :  single_trigger;
                '4'   :  continious_trigger;
                '5'   :  reset_meter;
                '6'   :  send(device,'z0',status);
                '7'   :  save_data(saving_data);
                '8'   :  display_files;
                '9'   :  valid := true;
           end (** case **);
         until valid;
         if saving_data then
            begin
                 saving_data := false;
                 close(todisk);
```

```
            end;
        end;

end;

(************************** end procedure ********************************)

(*************************************************************************)
(**                                                                     **)
(**                    end multimeter procedures                        **)
(**                                                                     **)
(*************************************************************************)

(************************** begin procedure ******************************)

procedure no_instrument;

var
   ch     : char;

begin                                          (* NO INSTRUMENT ERROR MESSAGE *)

     clearviewport;
     outtextxy(150,100,'NO INSTRUMENT ATTACHED AT THIS TIME!');
     outtextxy(150,150,' PRESS ANY KEY TO CONTINUE.....');
     ch := readkey;

end;

(************************** end procedure ********************************)

(************************** begin procedure ******************************)

procedure instr_plotter;

var
   ch  : char;

begin                                    (* PLOTTER UNDER DEVELOPMENT MESSAGE *)

     clearviewport;
     outtextxy(150,10,'  PROGRAM MODULE UNDER DEVELOPMENT...');
     outtextxy(150,25,'        CONTACT:');
     outtextxy(150,40,'             LES KINSLER');
     outtextxy(150,55,'    KANSAS COLLEGE OF TECHNOLOGY');
     outtextxy(150,70,'       2409 SCANLAN AVE.');
     outtextxy(150,85,'      SALINA, KANSAS   67401');
     outtextxy(150,100,'          913-825-0275');
     outtextxy(200,115,'For completion date');
     outtextxy(200,150,'PRESS ANY KEY TO CONTINUE');
     ch := readkey;

end;

(************************** end procedure ********************************)

(************************** begin procedure ******************************)

procedure main_processing;

var
    instrument     : char;
    valid      : boolean;

begin                                        (* SELECT FROM MAIN MENU CHOICES *)
```

```
      valid  := false;
      repeat
            main_menu(instrument);
            case instrument of
                  'A', 'a'    : begin
                                    device := fluke_num1;
                                    instrument_num := '1';
                                    multi_one;
                                    end;
                  'B', 'b'    : begin
                                    device := fluke_num2;
                                    instrument_num := '2';
                                    multi_one;
                                    end;
                  'C', 'c'    : begin
                                    device := fluke_num3;
                                    instrument_num := '3';
                                    multi_one;
                                    end;
                  'D', 'd'    :  no_instrument;
                  'E', 'e'    :  no_instrument;
                  'F', 'f'    :  no_instrument;
                  'G', 'g'    :  instr_plotter;
                  'Q', 'q'    :  valid  :=  true;
            end;
      until valid;
      closegraph;

end;

(************************** end procedure ******************************)

(*************************************************************************)
(**                                                                     **)
(**                          MAIN PROGRAM                               **)
(**                                                                     **)
(*************************************************************************)

begin                                          (* MAIN PROGRAM BEGINNING *)

      if valid_graphics  then
            main_processing;

end.
```