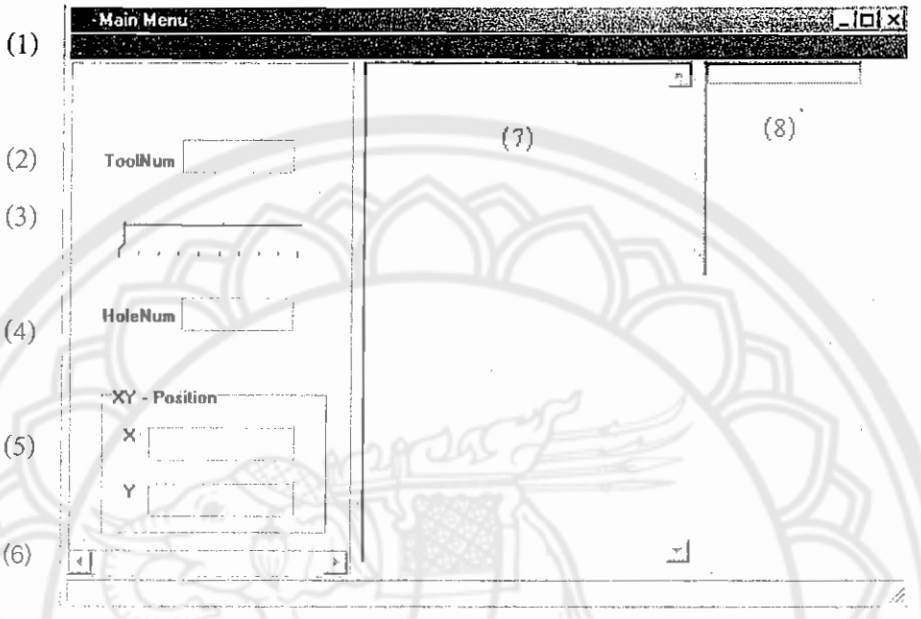
ภาคผนวก

ภาคผนวก ก
ส่วนประกอบและวิธีการใช้งานโปรแกรม

ส่วนประกอบของโปรแกรม

1. เมื่อเปิดโปรแกรมมาจะเจอหน้าจอหลักซึ่งใช้สำหรับเปิดไฟล์และโหลดไฟล์นั้นเข้ามา ซึ่งมีลักษณะดังต่อไปนี้



รูปที่ ก.1 หน้าจอหลักของโปรแกรม

(1) : เป็นเมนูบาร์ประกอบด้วย File และ Machine

(ก) (ข)

รูปที่ ก.2 รายละเอียดในเมนูบาร์ (ก) File (ค) Machine

(2) : ToolNum จะแสดงหมายเลขของดอกสว่าน

(3) : Trackbar จะเป็นส่วนที่ช่วยให้สามารถเลือกหมายเลขของดอกสว่านได้

(4) : HoleNum แสดงลำดับของรูเจาะในไฟล์ NC Drill ของดอกสว่านแต่ละดอก

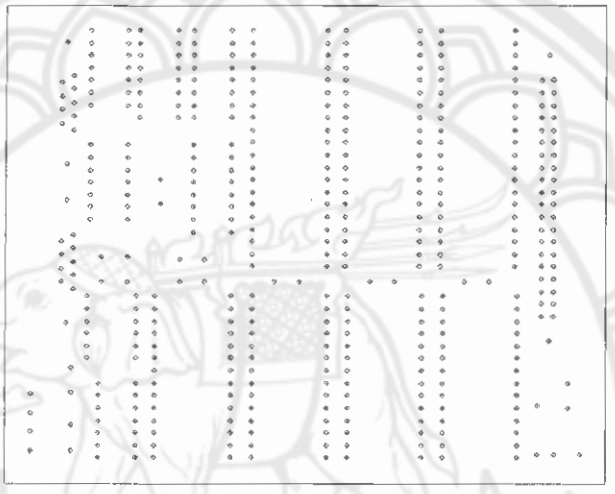
(5) : XY-Position แสดงพิกัด X และ Y ของรูเจาะที่แสดงอยู่ใน HoleNum

(6) : Scrollbar ใช้เลื่อนเพื่อดูพิกัดทั้งหมดของรูเจาะในดอกสว่านดอกนั้น

(7) : Memo1 เป็นตัวแสดงผลการโหลดไฟล์ NC Drill

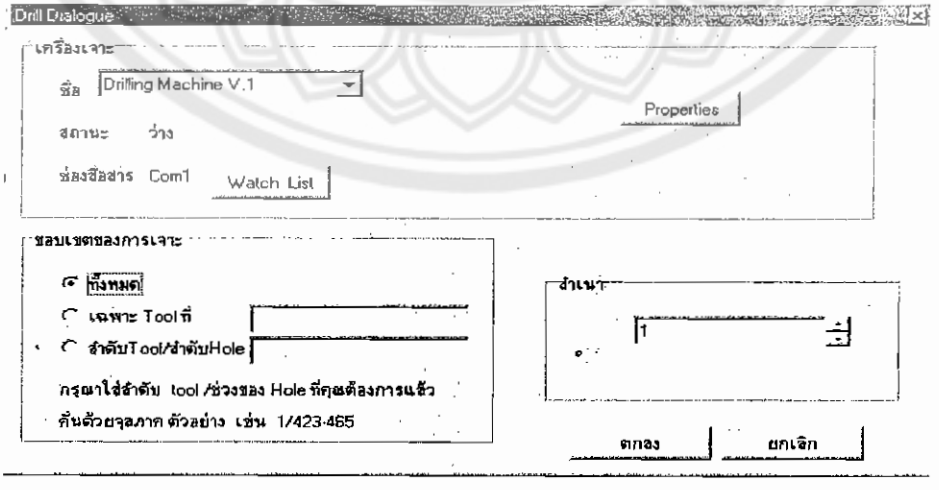
(8) : ListBox เป็นตัวแสดงขนาดของดอกสว่านที่ใช้ในไฟล์ NC Drillที่โหลดเข้ามา

2. Form ที่ 2 จะเป็น Form Preview ซึ่งจะแสดงภาพรูเจาะของไฟล์ที่โหลดเข้ามา ฟอร์มนี้จะไม่แสดงผลหากไม่มีการเปิดไฟล์เข้ามา



รูปที่ ก.3 ภาพรูเจาะที่แสดงในฟอร์ม Preview

3. Form ที่ 3 เป็น Drill Dialogue จะแสดงเมื่อมีการกด Machine | Setup ในเมนูบาร์ของหน้าจอหลัก



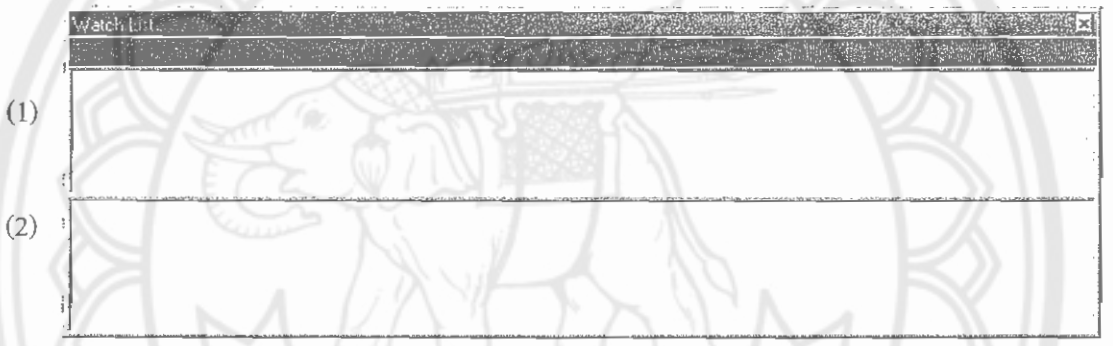
รูปที่ ก.4 Drill Dialogue

(1) : ส่วนแรกจะเป็นรายละเอียดของเครื่องเจาะที่ติดต่ออยู่กับโปรแกรม

(2) : ส่วนที่ 2 เป็นการกำหนดขอบเขตการทำงานของเครื่องเจาะโดยจะระบุว่าต้องการให้ส่งข้อมูลชุดใดไปให้เครื่องเจาะ

(3) : ส่วนที่ 3 เป็นการกำหนดว่าจะให้ส่งข้อมูลแต่ละชุดไปกี่ครั้ง ซึ่งขึ้นอยู่กับความต้องการของผู้ใช้ว่าต้องการงานชิ้นนั้นกี่ชุด

4. Form ที่ 4 ชื่อว่า Watch List เป็น Form ที่แสดงสถานะการทำงานของการส่งข้อมูลระหว่างเครื่องคอมพิวเตอร์กับเครื่องเจาะ จะแสดงผลเมื่อมีการกด Machine | Status จากหน้าจอหลักมีลักษณะดังรูป



รูปที่ ก.5 หน้าจอ Watch List

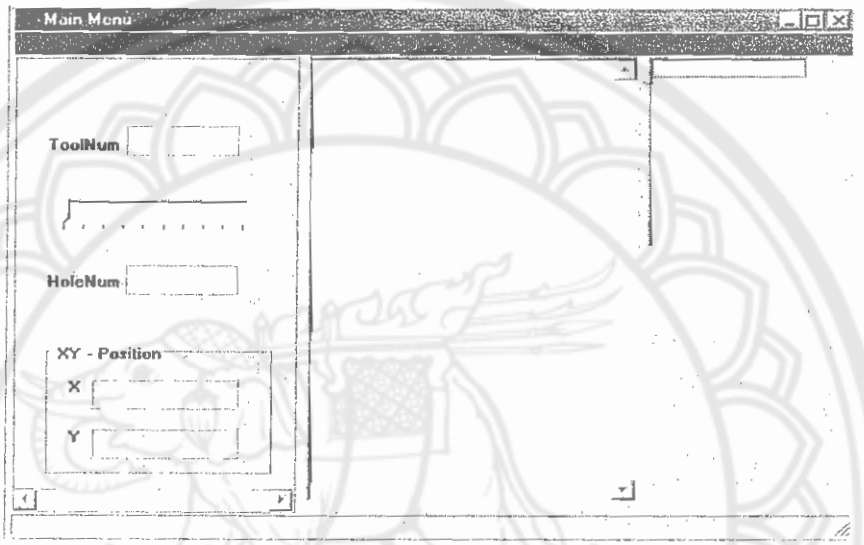
(1) : แสดงสถานะของการส่งข้อมูลไปยังเครื่องเจาะ

(2) : แสดงสถานะของการรับข้อมูลจากเครื่องเจาะกลับมายังโปรแกรมในเครื่องคอมพิวเตอร์

การใช้งานเครื่องงานเครื่องเจาะแผ่น PCB

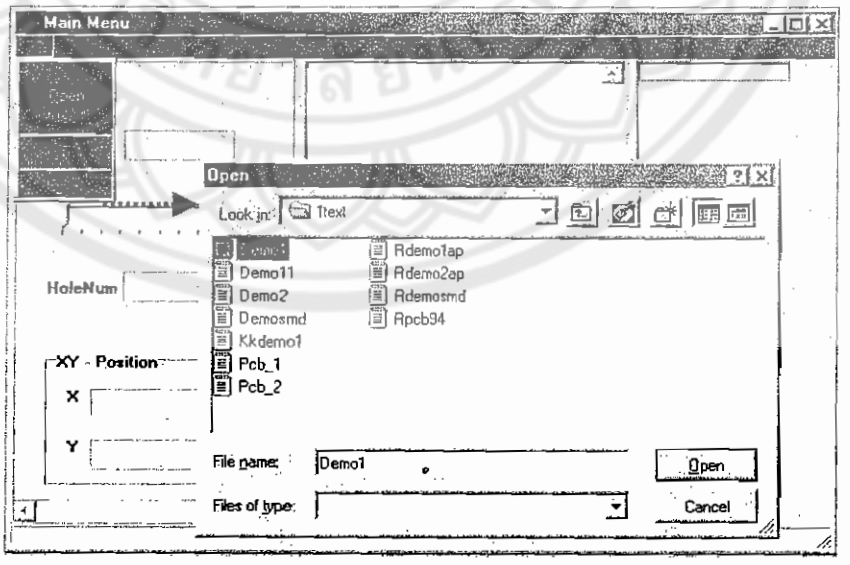
1. ติดตั้งเครื่องเจาะแผ่น PCB ให้เชื่อมต่อกับเครื่องคอมพิวเตอร์โดยการต่อผ่านพอร์ตสื่อสารหรือพอร์ตอนุกรม RS-232

2. เมื่อทำการเชื่อมต่อแล้วให้ทำการ RUN โปรแกรมสำหรับการเจาะขึ้นมา จะได้หน้าจอหลักซึ่งมีลักษณะดังนี้



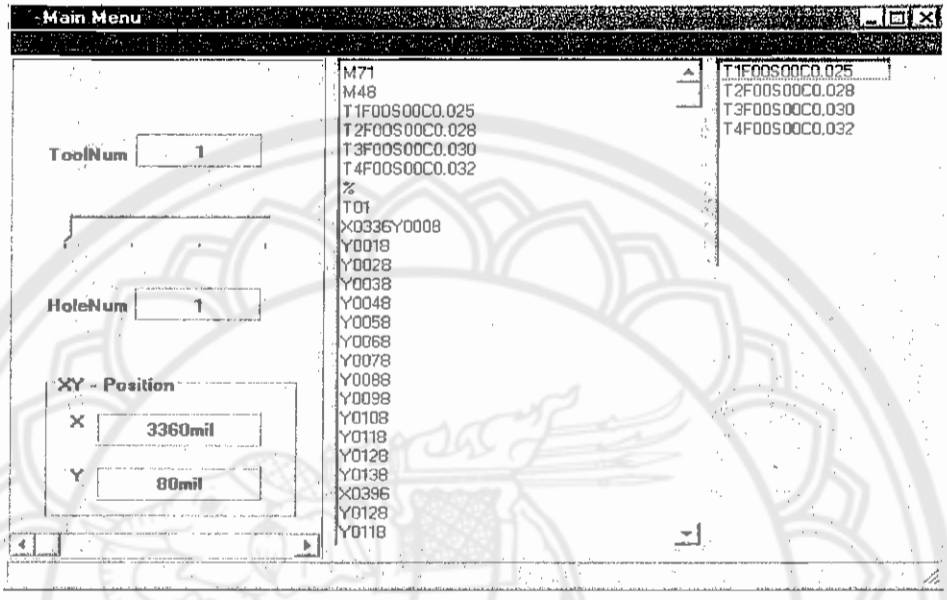
รูปที่ ก.6 หน้าจอหลักของโปรแกรมก่อนที่จะมีการเปิดไฟล์

2. เปิดไฟล์ที่ต้องการเจาะ โดยเลือก File | Open ที่เมนูบาร์ของหน้าจอหลัก จะได้ไดอะล็อกบอ็กซ์ Open เพื่อเลือกไฟล์ที่ต้องการดังรูป

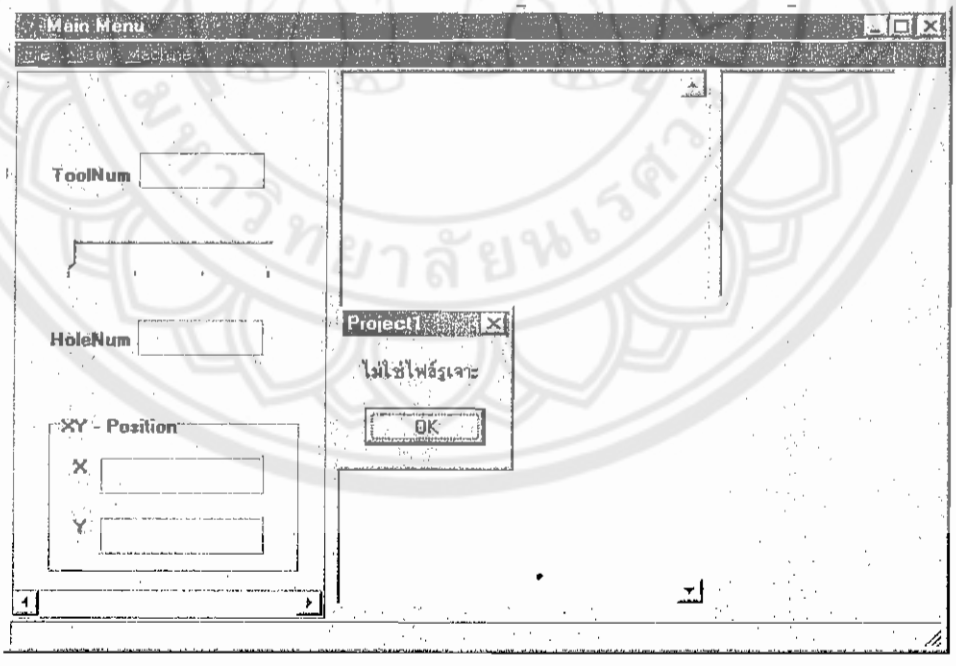


รูปที่ ก.7 แสดงหน้าจอหลักของโปรแกรมเมื่อมีการเลือก เมนู File| Open

3. ถ้าไฟล์ที่เปิดมาเป็นไฟล์ที่เป็น NC Drill ข้อมูลที่อยู่ในไฟล์ก็จะถูกโหลดเข้ามาใน Memo1 และ ใน ListBox ก็จะแสดงชื่อ Tool ทั้งหมดที่ใช้ในการเจาะ แต่ถ้าไฟล์ที่เปิดมาไม่ใช่ไฟล์ NC Drill ก็จะมีการแสดง message error ขึ้นมาตามลำดับ ดังรูป

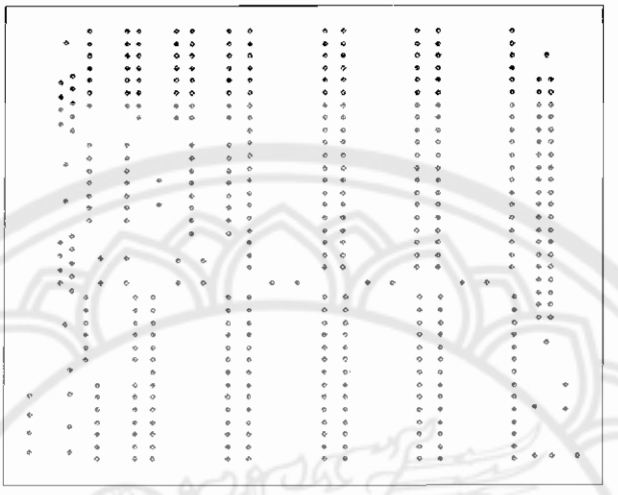


รูปที่ ก.8 หน้าจอหลักของโปรแกรมเมื่อทำการโหลดไฟล์ที่เลือกไว้เข้ามา



รูปที่ ก.9 หน้าจอหลักของโปรแกรมในกรณีที่ไฟล์ที่เปิดมาไม่ใช่ไฟล์ที่ใช้สำหรับเป็นข้อมูลในการเจาะ

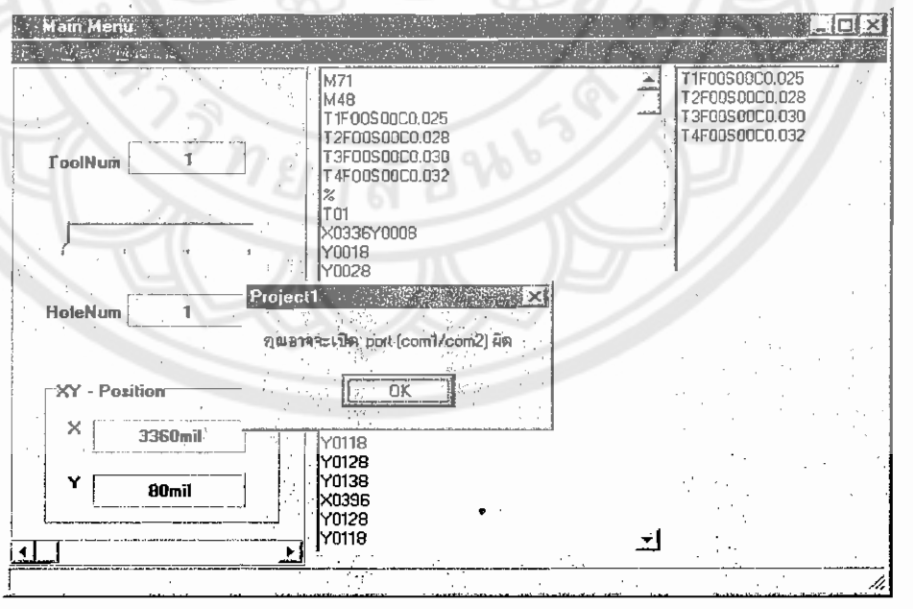
4. เมื่อโหลดไฟล์เข้ามาได้แล้ว เราสามารถดูลักษณะของรูเจาะได้ โดยการเลือกที่ View | Preview บนเมนูบาร์ จะปรากฏภาพของรูเจาะที่ได้จากการ simulate ดังรูป



รูปที่ ก.10 หน้าจอของการ Simulate ภาพรูเจาะขอวไฟล์ที่โหลดมา

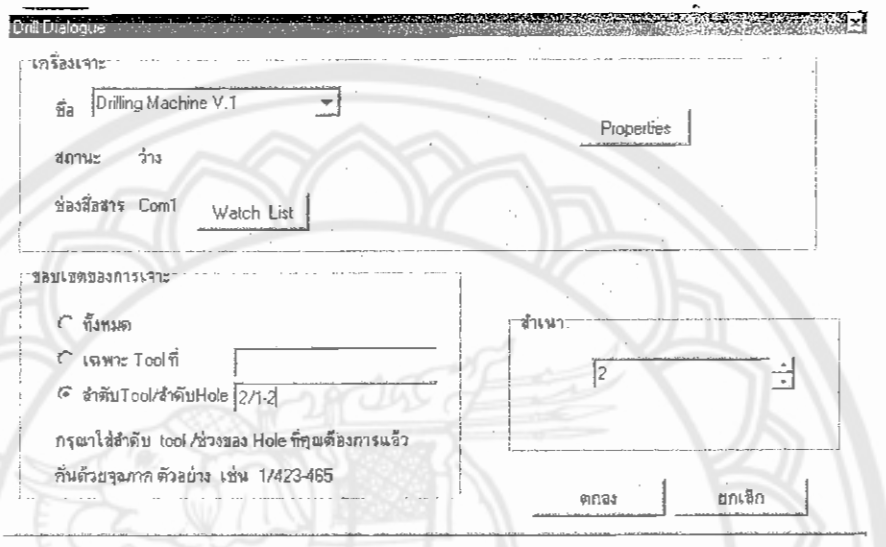
5. เมื่อต้องการออกจากหน้าจอ Preview ทำได้โดยการกด Esc ก็จะกลับมาที่หน้าจอหลัก

6. เมื่อต้องการทำการเจาะให้ไปที่ File | Drill แต่ถ้าการเชื่อมต่อยังไม่เรียบร้อยก็จะมีการแสดง message error ขึ้นมา



รูปที่ ก.11 ผลการผิดพลาดเนื่องมาจากการสื่อสารยังไม่สมบูรณ์

7. ถ้าต้องการ Setup เครื่องเจาะแผ่น PCB สามารถทำได้โดยเลือก Machine | Setup ที่เมนูบาร์ จะปรากฏฟอร์ม Drill Dialogue ซึ่งเราสามารถเซตค่าต่างๆ ได้ เช่น ในตัวอย่างเราต้องการให้เจาะด้วย Tool 2 รูเจาะที่ 1-2 จำนวน 2 สำเนา จะเซตค่าดังรูป

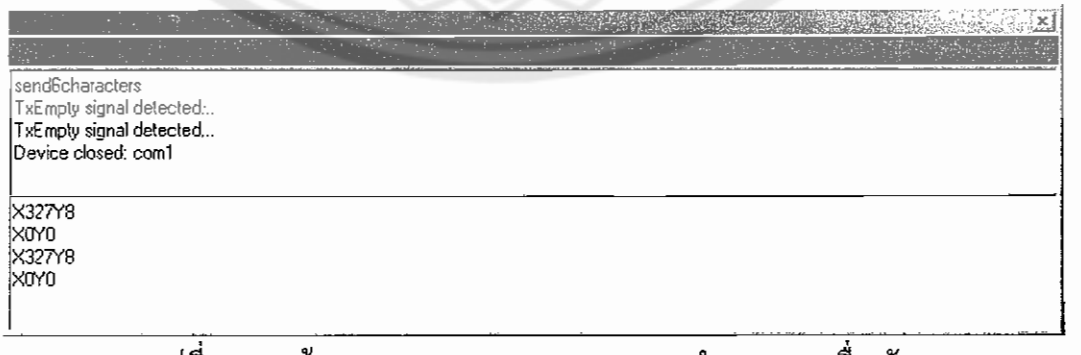


รูปที่ ก.12 Drill Dialog เมื่อมีการสั่งงานเครื่องเจาะ

8. เมื่อกด Button ตกลง ก็จะมีการเปิดพอร์ต ซึ่งค่าเริ่มต้นจะเป็นพอร์ต 1 หากสายสัญญาณ RS-232 ที่เราต่อกับเครื่องพีซีไม่ตรงกับที่เซตไว้จะมีการแสดง message error ดังรูปในข้อ 6 แต่ถ้าไม่เกิดข้อผิดพลาดก็จะทำการเจาะทันที

9. ขณะทำการเจาะ ถ้าต้องการดู status จะต้องเปิด Watch List ก่อนที่จะตอบตกลง เมื่อมีการส่งและรับข้อมูล Watch List จะปรากฏดังรูป

(หมายเหตุ ขณะเปิด Watch List จะต้องไม่ให้ไปซ้อนกับ Drill Dialogue)



รูปที่ ก.13 หน้าจอของการแสดงสถานะของการทำงานของเครื่องจักร

ภาคผนวก ข

Source Code ของโปรแกรม

โค้ดโปรแกรมที่พัฒนาด้วยเดลไฟ 5

```
unit Unit1;

Interface

uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, ComCtrls, ExtCtrls, Menus, Buttons,ShellApi,Math, jpeg, Async32;

Type
TForm1 = class(TForm)
Memo2: TMemo;
Memo3: TMemo;
MainMenu1: TMainMenu;
File1: TMenuItem;
Clear1: TMenuItem;
N1: TMenuItem;
Exit1: TMenuItem;
OpenDialog1: TOpenDialog;
ListBox1: TListBox;
Memo4: TMemo;
Memo5: TMemo;
Memo6: TMemo;
Memo7: TMemo;
Memo8: TMemo;
Panel5: TPanel;
Label3: TLabel;
Label4: TLabel;
GroupBox1: TGroupBox;
Label1: TLabel;
Label2: TLabel;
Panel1: TPanel;
Panel2: TPanel;
```

```
TrackBar1: TTrackBar;
Panel3: TPanel;
Panel4: TPanel;
ScrollBar1: TScrollBar;
Machine1: TMenuItem;
Setup1: TMenuItem;
Status1: TMenuItem;
N2: TMenuItem;
Drill1: TMenuItem;
Memo1: TMemo;
Memo9: TMemo;
Memo10: TMemo;
Memo11: TMemo;
Edit1: TEdit;
Label5: TLabel;
Open1: TMenuItem;
procedure Open1Click(Sender: TObject);
procedure Clear1Click(Sender: TObject);
procedure TrackBar1Change(Sender: TObject);
procedure ScrollBar1Change(Sender: TObject);
procedure Exit1Click(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure ProtelMicrosim1Click(Sender: TObject);
procedure Setup1Click(Sender: TObject);
procedure Status1Click(Sender: TObject);
procedure Drill1Click(Sender: TObject);
procedure FormDestroy(Sender: TObject);
    private
{ Private declarations }
public
{ Public declarations }
end;
```

```
var
Form1: TForm1;
function FiveStr(strings:string):string;
function Tovall(stringl:string):Integer;
function mil_to_step(mess:string):string;
procedure DrawImage(valx,valy,centx,centy:integer);
 implementation

 uses Unit2, Unit4, Unit5, Unit3;

{$R *.DFM}

 procedure TForm1.Open1Click(Sender: TObject);
 var i,posz,posx,posxxx,posyyy,posy,post,posdot,posper,posm30,posm48:integer;
stringline,stringfile,stMem6,strx,stry:string;
toolsize,valxxx,valyyy,lineMem6:integer;
k,n,p,jj:integer;
ss,l,m:string;
holenum,posxx,posyy,posttt,valx,valy:integer;
xx,xxx,yyy,yy,stx,sty,stringx,stringy:string;
lineMem7,lineMem8,minx,miny,maxx,maxy:integer;
scenx,sceny,stMem7,stMem77,stMem8,stMem88:string;
a,b,c,d,e,f,g,h:integer;
Xcenter,Ycenter:extended;
tool_mem4:integer;

begin
  tool_mem4:=0;
 valxxx:=0;
 valyyy:=0;
 toolsize:=0;
 Memo2.clear;
 Memo6.clear;
```

```
Memo9.clear;
Memo10.clear;
Form5.Memo1.Clear;
Form5.Memo2.Clear;
if OpenDialog1.Execute then
   begin
      Memo1.Lines.LoadFromFile(Opendialog1.filename);
      Edit1.Text:=OpenDialog1.FileName;
      stringfile:=Memo1.text;
      posx:=pos('X',stringfile);
      posy:=pos('Y',stringfile);
      post:=pos('T',stringfile);
      posdot:=pos('.',stringfile);
      posper:=pos('%',stringfile);
      posm30:=pos('M30',stringfile);
      posm48:=pos('M48',stringfile);
      if (posx<>0)and(posy<>0)and(post<>0)and(posdot<>0)and(posper<>0)and
      (posm48<>0)and(posm30<>0) then
        begin
          for i:=0 to Memo1.Lines.count-1 do
            begin
              stringline:=Memo1.Lines[i];
              posx:=pos('X',stringline);
              posy:=pos('Y',stringline);
              post:=pos('T',stringline);
              posdot:=pos('.',stringline);
              posm30:=pos('M30',stringline);
              posz:=pos('Z',stringline);
              if(post<>0)and(posdot<>0)then
                 begin
                    ListBox1.Items.Add(copy(stringline,1,20));
                    toolsize:=toolsize+1;
                 end;//endเจอ บรรทัดบอกขนาดของtool
```

```
if((post<>0)and(posdot=0)and(posz=0))then
        begin
          Memo2.Lines.Add(InttoStr(i));
          Memo9.Lines.Add(InttoStr(tool_mem4));
          Memo4.Lines.Add(stringline);
          Memo6.Lines.Add(stringline);
          tool_mem4:=tool_mem4+1;
        end;//เจอบรรทัดบอกtool ลำดับ
      if(posx<>0)or(posy<>0)then
        begin
          if(posx<>0)and(posy<>0)then
            begin
              stx:=FiveStr(copy(stringline,posx+1,posy-posx-1));
              sty:=FiveStr(copy(stringline,posy+1,5));
              valxxx:=StrToInt(stx);
              valyyy:=StrToInt(sty);
              Memo6.Lines.Add('X'+IntToStr(valxxx)+'Y'+IntToStr(valyyy));
              xxx:=mil_to_step(stx);
              yyy:=mil_to_step(sty);
              Memo4.Lines.Add('X'+xxx+'Y'+yyy);
            end;
          if (posx<>0)and(posy=0) then
            begin
              stx:=FiveStr(copy(stringline,posx+1,5));
              valxxx:=StrToInt(stx);
              Memo6.Lines.Add('X'+IntToStr(valxxx)+'Y'+IntToStr(valyyy));
              xxx:=mil_to_step(stx);
              Memo4.Lines.Add('X'+xxx+'Y'+yyy);
             // Memo4.Lines.Add('X'+xxx);
            end;
```

```
if(posx=0)and(posy<>0)then
   begin
      sty:=FiveStr(copy(stringline,posy+1,5));
      valyyy:=StrToInt(sty);
      Memo6.Lines.Add('X'+IntToStr(valxxx)+'Y'+IntToStr(valyyy));
       yyy:=mil_to_step(sty);
       Memo4.Lines.Add('X'+xxx+'Y'+yyy);
       // Memo4.Lines.Add('Y'+yyy);
   end;
       tool_mem4:=tool_mem4+1;
       end;
       if(posm30<>0)then
          begin
             Memo9.Lines.Add(InttoStr(tool_mem4));
             Memo2.Lines.Add(InttoStr(i));
              tool_mem4:=tool_mem4+1;
          end;
          end;//รับบรรทัดใน Memo1

   ScrollBar1.Enabled:=true;
   TrackBar1.Enabled:=true;
   TrackBar1.Max:=toolsize;
   Memo3.Clear;
   k:=TrackBar1.Position-1;
   l:=Memo2.Lines[k];
   m:=Memo2.Lines[k+1];
   n:=StrToInt(l);
   p:=StrToInt(m);
   for jj:=(n+1)to (p-1)do
      begin
         ss:=Memo1.Lines[jj];
         Memo3.Lines.add(ss);
      end;
```

```
ScrollBar1.Max:=Memo3.Lines.Count-1;
holenum:=scrollBar1.Position;
posxx:=pos('X',Memo3.Lines[holenum]);
posyy:=pos('Y',Memo3.Lines[holenum]);
if (posxx<>0)and(posyy<>0)then
   begin
      xx:=copy(Memo3.Lines[holenum],posxx+1,posyy-posxx-1);
      stringx:=FiveStr(xx);
      valx:=Tovall(stringx);
      yy:=copy(Memo3.Lines[holenum],posyy+1,20);
      stringy:=FiveStr(yy);
      valy:=Tovall(stringy);
      Panel1.Caption:=IntToStr(valx)+'mil';
      Panel2.Caption:=IntToStr(valy)+'mil';
   end;
if (posxx<>0)and(posyy=0)then
   begin
      xx:=copy(Memo3.Lines[holenum],posxx+1,20);
      stringx:=FiveStr(xx);
      valx:=Tovall(stringx);
      Panel1.Caption:=IntToStr(valx)+'mil';
   end;
if (posxx=0)and(posyy<>0)then
   begin
      yy:=copy(Memo3.Lines[holenum],posyy+1,20);
      stringy:=FiveStr(yy);
      valy:=Tovall(stringy);
      Panel2.Caption:=IntToStr(valy)+'mil';
   end;
Panel3.Caption:=InttoStr(TrackBar1.Position);
Panel4.Caption:=InttoStr(ScrollBar1.Position+1);
Memo7.Clear;
Memo8.Clear;
```

```
for lineMem6:=0 to Memo6.Lines.Count-1 do
   begin
      stMem6:=Memo6.Lines[lineMem6];
      posxxx:=pos('X',stMem6);
      posyyy:=pos('Y',stMem6);
      posttt:=pos('T',stMem6);
      if ((posxx<>0)or(posyy<>0))and(posttt=0)then
         begin
            strx:=copy(stMem6,posxxx+1,posyyy-posxxx-1);
            stry:=copy(stMem6,posyyy+1,5);
            Memo7.Lines.Add(strx);
            Memo8.Lines.Add(stry);
         end;

   end;
   stMem7:=Memo7.Lines[0];
   a:=StrToInt(stMem7);
   c:=StrToInt(stMem7);
for lineMem7:=1 to Memo7.Lines.Count-1 do
   begin
      stMem77:=Memo7.Lines[lineMem7];
      b:=StrToInt(stMem77);
      d:=StrToInt(stMem77);
      a:=Max(a,b);
      c:=Min(c,d);
   end;
   maxx:=a;
   minx:=c;
   Xcenter:=((maxx+minx)/2)*(96/1000);//(96035/1000000);
   Memo7.clear;
   str(Xcenter:0:0,scenx);
   Memo7.Lines.Add(scenx);
   Memo7.Lines.Add(IntToStr(maxx));
```

```
    Memo7.Lines.Add(IntToStr(minx));
     stMem8:=Memo8.Lines[0];
     e:=StrToInt(stMem8);
     g:=StrToInt(stMem8);
   for lineMem8:=1 to Memo8.Lines.Count-1 do
     begin
        stMem88:=Memo8.Lines[lineMem8];
        f:=StrToInt(stMem88);
        h:=StrToInt(stMem88);
        e:=Max(e,f);
        g:=Min(g,h);
     end;
     maxy:=e;
     miny:=g;
     Ycenter:=((maxy+miny)/2)*(96/1000);//(96035/1000000);
     Memo8.clear;
     str(Ycenter:0:0,sceny);
     Memo8.Lines.Add(sceny);
     Memo8.Lines.Add(IntToStr(maxy));
     Memo8.Lines.Add(IntToStr(miny));
     ////////////////////////
  end //end if -->เป็น file รูเจาะ
else
     begin
        Memo1.Clear;
        beep;
        ShowMessage('ไม่ใช่ไฟล์รูเจาะ');
        Memo2.Clear;
        Memo3.clear;
        ListBox1.Clear;

     end;
```

```
    end; //opendialog
if Memo1.Lines.Count >7 then
begin
    Memo10.Clear;
    Form1.Button1Click(sender);
    Form2.Label2.Font.Color:=clMenu;
    setup1.Enabled:=true;
    status1.Enabled:=true;
    clear1.Enabled:=true;
  end;

end;

  procedure TForm1.Clear1Click(Sender: TObject);
  begin
      Memo1.clear;
      Memo2.clear;
      Memo3.clear;
      Memo4.clear;
      Memo5.clear;
      Memo6.clear;
      Memo7.clear;
      Memo8.clear;
      Memo9.clear;
      Memo10.clear;
      Form5.Memo1.Clear;
      Form5.Memo2.Clear;
      ListBox1.Clear;
      TrackBar1.Position:=1;
      TrackBar1.Enabled:=false;
      ScrollBar1.Enabled:=false;
      ScrollBar1.Position:=0;
```

```
Panel1.caption:='';
 Panel2.caption:='';
Panel3.Caption:='';
Panel4.Caption:='';
refresh;
 end;
 procedure TForm1.TrackBar1Change(Sender: TObject);
 var
      k,n,p,jj:integer;
      ss,l,m,stringx,stringy,xx,yy:string;
      holenum,posxx,posyy,valx,valy:integer;
 begin
      ScrollBar1.Position:=0;
      Memo3.Clear;
      k:=TrackBar1.Position-1;
      l:=Memo2.Lines[k];
      m:=Memo2.Lines[k+1];
      n:=StrToInt(l);
      p:=StrToInt(m);
      for jj:=(n+1)to (p-1)do
      begin
            ss:=Memo1.Lines[jj];
            Memo3.Lines.add(ss);
      end;
      ScrollBar1.Max:=Memo3.Lines.Count-1;
      holenum:=scrollBar1.Position;
      posxx:=pos('X',Memo3.Lines[holenum]);
      posyy:=pos('Y',Memo3.Lines[holenum]);
      if (posxx<>0)and(posyy<>0) then
      begin
            xx:=copy(Memo3.Lines[holenum],posxx+1,posyy-posxx-1);
            stringx:=FiveStr(xx);
```

```
            valx:=Tovall(stringx);
            yy:=copy(Memo3.Lines[holenum],posyy+1,20);
            stringy:=FiveStr(yy);
            valy:=Tovall(stringy);
            Panel1.Caption:=IntToStr(valx)+'mil';
            Panel2.Caption:=IntToStr(valy)+'mil';
        end;
    if (posxx<>0)and(posyy=0)then
        begin
        xx:=copy(Memo3.Lines[holenum],posxx+1,posyy-posxx-1);
        stringx:=FiveStr(xx);
        valx:=Tovali(stringx);
        Panel1.Caption:=IntToStr(valx)+'mil';
        end;
    if (posxx=0)and(posyy<>0)then
        begin
            yy:=copy(Memo3.Lines[holenum],posyy+1,20);
            stringy:=FiveStr(yy);
            valy:=Tovall(stringy);
            Panel2.Caption:=IntToStr(valy)+'mil';
        end;
        Panel3.Caption:=InttoStr(TrackBar1.Position);
        Panel4.Caption:=InttoStr(ScrollBar1.Position+1);
        end;

procedure TForm1.ScrollBar1Change(Sender: TObject);
var
    xx,yy,stringx,stringy:string;
    valx,valy:integer;
    holenum,posxx,posyy:integer;
begin
    holenum:=scrollBar1.Position;
    posxx:=pos('X',Memo3.Lines[holenum]);
```

```
posyy:=pos('Y',Memo3.Lines[holenum]);
if (posxx<>0)and(posyy<>0)then
   begin
            xx:=copy(Memo3.Lines[holenum],posxx+1,posyy-posxx-1);
            stringx:=FiveStr(xx);
            valx:=Tovall(stringx);
            yy:=copy(Memo3.Lines[holenum],posyy+1,20);
            stringy:=FiveStr(yy);
            valy:=Tovall(stringy);
            Panel1.Caption:=IntToStr(valx)+'mil';
            Panel2.Caption:=IntToStr(valy)+'mil';
   end;
if (posxx<>0)and(posyy=0)then
   begin
            xx:=copy(Memo3.Lines[holenum],posxx+1,20);
            stringx:=FiveStr(xx);
            valx:=Tovall(stringx);
            Panel1.Caption:=IntToStr(valx)+'mil';
   end;
if (posxx=0)and(posyy<>0)then
   begin
            yy:=copy(Memo3.Lines[holenum],posyy+1,20);
            stringy:=FiveStr(yy);
            valy:=Tovall(stringy);
            Panel2.Caption:=IntToStr(valy)+'mil';
   end;
Panel3.Caption:=InttoStr(TrackBar1.Position);
Panel4.Caption:=InttoStr(ScrollBar1.Position+1);
end;

function FiveStr(strings:string):string;
var  len:integer;
begin
```

```
len:=length(strings);
  If len<5 then
     begin
          while len<5 do
          begin
               strings:=strings+'0';
               len:=length(strings);
          end;
     end;
  if (len>=5)then
     begin
          strings:=copy(strings,1,5);
     end;
   result:=strings;
     end;

function Tovall(stringl:string):Integer;
var  a0,a1,a2,a3,a4,suma:integer;
begin
     a0:=StrToInt(copy(stringl,5,1));
     a1:=StrToInt(copy(stringl,4,1));
     a2:=StrToInt(copy(stringl,3,1));
     a3:=StrToInt(copy(stringl,2,1));
     a4:=StrToInt(copy(stringl,1,1));
     suma:=(a0)+(10*a1)+(100*a2)+(1000*a3)+(10000*a4);
     result:=suma;
end;

procedure TForm1.Exit1Click(Sender: TObject);
begin
     Application.OnException:=nil;
     Close;
end;
```

```
function mil_to_step(mess:string):string;
var
    valmil:extended;
    strstep:string;
    mil:integer;
begin
    mil:=StrToInt(mess);
    valmil:=mil*(127/5000)*(1000/261);
    Str(valmil:0:0,strstep);
    result:=strstep;
end;

procedure DrawImage(valx,valy,centx,centy:integer);
var
    xx,yy,xmax,ymax,xmin,ymin:extended;
    X,Y,X1,Y1,X2,Y2,x_max,x_min,y_max,y_min:integer;
    sX1,sX2,sY1,sY2,ax,ay,axmax,axmin,aymax,aymin:string;
begin
    xx:=valx*(96/1000);//(96035/1000000);
    yy:=valy*(96/1000);//(96035/1000000);
    xmax:=(StrToInt(Form1.Memo7.Lines[1]))*(96/1000);//(96035/1000000);
    ymax:=(StrToInt(Form1.Memo8.Lines[1]))*(96/1000);//(96035/1000000);
    xmin:=(StrToInt(Form1.Memo7.Lines[2]))*(96/1000);//(96035/1000000);
    ymin:=(StrToInt(Form1.Memo8.Lines[2]))*(96/1000);//(96035/1000000);
    str(xx:0:0,ax);
    str(yy:0:0,ay);
    X:=StrToInt(ax);
    Y:=StrToInt(ay);

    str(ymax:0:0,aymax);
    str(ymin:0:0,aymin);
    str(xmax:0:0,axmax);
```

```
      str(xmin:0:0,axmin);
      y_max:=600-(StrToInt(aymax)+centy)-20;
      y_min:=600-(StrToInt(aymin)+centy)+20;
      x_max:=StrToInt(axmax)+centx+20;
      x_min:=StrToInt(axmin)+centx-20;
      X:=StrToInt(ax);
      Y:=StrToInt(ay);
      X1:=X-2+centx{+200};X2:=X+2+centx{+200};
      Y1:=600-Y-2-centy{-200};Y2:=600-Y+2-centy{-200};
     Form2.Canvas.pen.color:=clLime;
     Form2.Canvas.MoveTo(x_min,y_min);
     Form2.Canvas.LineTo(x_max,y_min);
     Form2.Canvas.LineTo(x_max,y_max);
     Form2.Canvas.LineTo(x_min,y_max);
     Form2.Canvas.LineTo(x_min,y_min);
     sX1:=IntToStr(X1);
     sX2:=IntToStr(X2);
     sY1:=IntToStr(Y1);
     sY2:=IntToStr(Y2);

   with Form1 do
     begin
            Memo5.Lines.add('X'+ax+'Y'+ay);
     end;
    Form2.canvas.pen.color:=clRed;
    Form2.canvas.arc(X1,Y1,X2,Y2,0,0,0,0);
    Form1.Memo10.Lines.Add('a'+sX1+'b'+sY1+'c'+sX2+'d'+sY2);
end;

procedure TForm1.Button1Click(Sender: TObject);
var  stringline,stx,sty:string;
     post,posx,posy,i,intx,inty:integer;
```

```
    scentx,scenty:string;
    Xcenter,xxmin,xxmax,yymin,yymax,Ycenter:integer;
begin
    Form2.Show;
    Memo5.Clear;
    scentx:=Memo7.Lines[0];
    xxmax:=StrToInt(Memo7.Lines[1]);
    xxmin:=StrToInt(Memo7.Lines[2]);
    scenty:=Memo8.Lines[0];
    yymax:=StrToInt(Memo8.Lines[1]);
    yymin:=StrToInt(Memo8.Lines[2]);
    canvas.Pen.color:=clLime;
    Xcenter:=400-StrToInt(scentx);
    Ycenter:=300-StrToInt(scenty);
  for i:=0 to Memo6.Lines.Count-1 do
    begin
          stringline:=Memo6.Lines[i];
          posx:=pos('X',stringline);
          posy:=pos('Y',stringline);
          post:=pos('T',stringline);
          if (posx<>0)or(posy<>0)then
            begin
                stx:=copy(stringline,posx+1,posy-posx-1);
                sty:=copy(stringline,posy+1,5);
                intx:=StrToInt(stx);
                inty:=StrToInt(sty);
                DrawImage(intx,inty,Xcenter,Ycenter);
            end;
         if (post<>0)then
            begin
                Memo5.Lines.add(stringline);
                Memo10.Lines.Add(IntToStr(i));
```

```
        end;
        end;
        Memo10.Lines.Add(IntToStr(i));
        end;

procedure TForm1.ProtelMicrosim1Click(Sender: TObject);
var  stringline,stx,sty:string;
        post,posx,posy,i,intx,inty:integer;
        scentx,scenty:string;
        Xcenter,xxmin,xxmax,yymin,yymax,Ycenter:integer;
begin
        Form2.Show;
        Form1.Hide;
        Memo5.Clear;
        scentx:=Memo7.Lines[0];
        xxmax:=StrToInt(Memo7.Lines[1]);
        xxmin:=StrToInt(Memo7.Lines[2]);
        scenty:=Memo8.Lines[0];
        yymax:=StrToInt(Memo8.Lines[1]);
        yymin:=StrToInt(Memo8.Lines[2]);
        canvas.Pen.color:=clLime;
        Xcenter:=400-StrToInt(scentx);
        Ycenter:=300-StrToInt(scenty);
        for i:=0 to Memo6.Lines.Count-1 do
        begin
                stringline:=Memo6.Lines[i];
                posx:=pos('X',stringline);
                posy:=pos('Y',stringline);
                post:=pos('T',stringline);
                if (posx<>0)or(posy<>0)then
                begin
                        stx:=copy(stringline,posx+1,posy-posx-1);
                        sty:=copy(stringline,posy+1,5);
```

```
    intx:=StrToInt(stx);
    inty:=StrToInt(sty);
    DrawImage(intx,inty,Xcenter,Ycenter);
      end;
    if (post<>0)then
      begin
        Memo5.Lines.add(stringline);
      end;
    end;
end;

procedure TForm1.Setup1Click(Sender: TObject);
var  Command,Params,WorkDir:string;
begin
    Form1.Hide;
    Form2.Hide;
    Form5.Show;
end;

procedure TForm1.Status1Click(Sender: TObject);
begin
    Form3.Show;
end;
procedure TForm1.Drill1Click(Sender: TObject);
begin
    Form1.Hide;
    Form2.Hide;
    Form5.Show;
end;

procedure TForm1.FormDestroy(Sender: TObject);
begin
end;
```

```
procedure TForm1.FormClose(Sender: TObject);
begin
    Application:=nil;
end;
end.
```

```
unit Unit2;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
ExtCtrls, StdCtrls, ComCtrls;
type
TForm2 = class(TForm)
Label1: TLabel;
Label2: TLabel;
procedure FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
private
{ Private declarations }
public
{ Public declarations }
end;
var
Form2: TForm2;
implementation
uses Unit1;
{$R *.DFM}

procedure TForm2.FormKeyDown(Sender: TObject; var Key: Word;
Shift: TShiftState);
  begin
  If key=VK_ESCAPE then
    begin
            Form2.Hide;
            Form1.Show;
    end;
  end;
end.
```

```
unit Unit3;
interface
uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
Menus, StdCtrls;

type
    TForm3 = class(TForm)
    MainMenu1: TMainMenu;
    Printer1: TMenuItem;
    PauseDrilling1: TMenuItem;
    SetAsDefault1: TMenuItem;
    N1: TMenuItem;
    property1: TMenuItem;
    N2: TMenuItem;
    PauseDrilling2: TMenuItem;
    CancelDrilling1: TMenuItem;
    Help1: TMenuItem;
    About1: TMenuItem;
    Memo1: TMemo;
    Memo2: TMemo;
  procedure PauseDrilling1Click(Sender: TObject);
  private
  { Private declarations }
  public
  { Public declarations }
  end;

var
    Form3: TForm3;
```

```
implementation

uses Unit5;

{$R *.DFM}

procedure TForm3.PauseDrilling1Click(Sender: TObject);
var ss:string;
  count1:integer;
    begin
          ss:='!';
          Count1 := Form5.Comm1.Write(ss[1], Length(ss));
          if Count1 <0 then
                Memo1.Lines.add('Error writing to: ' + Form5.Comm1.DeviceName)
          else Memo1.Lines.add('Transmit'+ IntToStr(Count1) + ' characters');
    end;
end.
```

```
unit Unit4;
interface

uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls;

type
    TForm4 = class(TForm)
    ComboBox1: TComboBox;
    ComboBox2: TComboBox;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    ComboBox3: TComboBox;
    ComboBox4: TComboBox;
    Edit1: TEdit;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
  procedure Button1Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
 private
  { Private declarations }
  public
  { Public declarations }
  end;
var
 Form4: TForm4;
Implementation
```

```
uses Unit5;

{$R *.DFM}

procedure TForm4.Button1Click(Sender: TObject);
begin
    Form4.Hide;
    Form5.Show;
end;

procedure TForm4.FormCreate(Sender: TObject);
begin
    ComboBox1.Items.Text:='cbr9600';
    ComboBox2.Items.Text:='da8';
    comboBox3.Items.Text:='paNone';
    comboBox4.Items.text:='sb10';
end;
end.
```

```
unit Unit5;

Interface

uses
Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
ComCtrls, ExtCtrls, StdCtrls, Async32;

type
    TForm5 = class(TForm)
    Label4: TLabel;
    GroupBox1: TGroupBox;
    Label5: TLabel;
    Label6: TLabel;
    RadioButton2: TRadioButton;
    RadioButton3: TRadioButton;
    Edit1: TEdit;
    Button2: TButton;
    Button3: TButton;
    GroupBox2: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    GroupBox3: TGroupBox;
    Edit2: TEdit;
    UpDown1: TUpDown;
    ComboBox1: TComboBox;
    Comm1: TComm;
    Button1: TButton;
    Edit3: TEdit;
```

```
    Memo1: TMemo;
    Memo2: TMemo;
    Button4: TButton;
  procedure Button3Click(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure RadioButton1Click(Sender: TObject);
  procedure RadioButton2Click(Sender: TObject);
  procedure RadioButton3Click(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure FormDestroy(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure Comm1Break(Sender: TObject);
  procedure Comm1RxFlag(Sender: TObject);
  procedure Comm1TxEmpty(Sender: TObject);
  procedure Comm1CTS(Sender: TObject);
  procedure Comm1DSR(Sender: TObject);
  procedure Comm1Ring(Sender: TObject);
  procedure Comm1RLSD(Sender: TObject);
  procedure Comm1Error(Sender: TObject; Errors: Integer);
  procedure Button1Click(Sender: TObject);
  procedure Button4Click(Sender: TObject);
  procedure FormActivate(Sender: TObject);
  private
  { Private declarations }
   FCurrentLine: Integer;
  procedure HandleException(Sender: TObject; E: Exception);
  public
 { Public declarations }
end;
var
    time25:integer;
    Form5: TForm5;
```

```
  TolOr,HolSta,HolEnd : integer;
  ToolLar:array[1..10] of integer;
  procedure DrawBack(milxy:string);
    Implementation

  uses Unit3, Unit1, Unit4, Unit2;

 {$R *.DFM}
  const
      OnOff: array[0..1] of string = ('Off', 'On');

  procedure TForm5.Button3Click(Sender: TObject);
  begin
      form5.Hide;
      Comm1.Close;
      Form3.Memo1.Lines.Add('Device closed: ' + Comm1.DeviceName);
      Form3.Memo2.Clear;
      FCurrentLine:=0 ;
      Form5.Close;
  end;

  procedure TForm5.FormCreate(Sender: TObject);
  begin
      Application.OnException := HandleException;
      ComboBox1.Text:='Drilling Machine V.1';
      Edit1.Color:=clBtnFace;
      Edit1.Enabled:=false;
      comm1.DeviceName:='com1';
      Comm1.BaudRate := TBaudrate(cbr9600);
      Comm1.Databits := TDataBits(da8);
      Comm1.Parity := TParity(paNone);
      Comm1.StopBits := TStopBits(sb10);
      FCurrentLine := 0;   //prepare receive buffer
```

```
        time25:=0;
end;

procedure TForm5.RadioButton1Click(Sender: TObject);
begin
        Edit1.Color:=clBtnFace;
        Edit1.Enabled:=false;
        Edit3.Enabled:=false;
        Edit3.Color:=clBtnFace;
end;

procedure TForm5.RadioButton2Click(Sender: TObject);
begin
        Edit1.Color:=clBtnFace;
        Edit1.Enabled:=false;
        Edit3.Enabled:=true;
        Edit3.Color:=clWhite;
end;

procedure TForm5.RadioButton3Click(Sender: TObject);
begin
        Edit1.Color:=clWhite;
        Edit1.Enabled:=true;
        Edit3.Enabled:=false;
        Edit3.Color:=clBtnFace;
end;

procedure TForm5.Button2Click(Sender: TObject);
var
        E: Exception;
        rs:word;
        S: string;
```

```
  Count: Integer;
    ifm5,ATool:integer;
    w,v,u,TolN,posSlash,posDat:integer;
    Buffer:array[1..32] of char;
    lines,l2,round,i,h5,Bytes,P,nth: integer;
begin
    u:=0;
    ATool:=0;
    lines:=1;
    Memo1.Clear;
    Memo2.Clear;
    Memo1.text:=Form1.Memo4.Text;
    comm1.DeviceName:=Form4.Edit1.text;
    Atool:=Form1.Memo9.Lines.Count;
 try
    Comm1.Open;
 except
    on E: ECommError do
    showmessage(E.Message);
end;

for ifm5:=1 to Atool do
begin
    ToolLar[ifm5]:=StrToInt(Form1.Memo9.Lines[ifm5 - 1]);
end;

if RadioButton3.Checked then
begin
    posSlash:=pos('/',Edit1.text);
    posDat:=pos('-',Edit1.text);
    TolN:=StrToInt(copy(Edit1.text,1,posSlash-1)); //
    HolSta:=ToolLar[TolN]+StrToInt(copy(Edit1.text,posSlash+1,posDat-posSlash-1));
    HolEnd:=ToolLar[TolN]+StrToInt(copy(Edit1.text,posDat+1,5));
```

```
 end
else if RadioButton2.Checked then
begin
    TolN:=StrToInt(Edit3.Text);
    if (TolN>Atool-1)and(TolN<0)then
      begin
            beep;
            ShowMessage('Please retype new tool');
            Edit3.Clear;
      end;
    if(TolN>=1)and(TolN<Atool) then
      begin
            HolSta:=ToolLar[TolN]+1;
            HolEnd:=ToolLar[TolN+1]-1;
      end;
 end

else
begin
   HolSta:=ToolLar[1]+1;
   HolEnd:=ToolLar[Atool]-1;
end ;
   for w:=HolSta to HolEnd do
   begin
            l2:=(10*Lines)+Lines-1;
            if w=10+HolSta then
                  begin
                        Memo2.Lines.Add('X0Y0');
                        inc(Lines);
                  end;

 if (w=HolSta+l2-Lines+1)and(w>10+HolSta) then
    begin
```

```
            Memo2.Lines.Add('X0Y0');
            inc(Lines);
end
 else;
Memo2.Lines.Add(Memo1.Lines[w]);
end;
Memo2.Lines.Add('X0Y0');
Memo2.Lines.Add('END');
round:=0;
nth:=HolSta;
////////////////////////////////////////////////////////////
if  (Comm1.Enabled and Comm1.DSR)=False then
   begin
          beep;
          ShowMessage('คุณอาจจะเปิด port (com1/com2) ผิด  ');
          Comm1.Close;
          Form3.Memo1.Lines.Add('Device closed: ' + Comm1.DeviceName);
          Exit
    end
else
   begin
         while round < StrToInt(Edit2.text) do
            begin
                  i:=0;
                  While (Memo2.Lines[i]<>'END') do
                        begin
                              s := Memo2.Lines[i];
                              s := s+#13#10;
                              Count := Comm1.Write(S[1], Length(S));
                              Form3.Memo1.Lines.Add('send'+intToStr
                              (Count)+'characters');
```

```
if Count =-1 then
    begin
        ShowMessage('Error writing to: ' + Comm1.DeviceName);
        exit
    end
else S:=' ';
    FillChar(Buffer, Sizeof(Buffer), 0);
    Bytes := Comm1.Read(Buffer, Count);
    //-------------
    Form5.Hide;
    Form2.Show;
    /-------------
    if Bytes = -1 then
      begin
        showmessage('Error reading incoming data...');
        exit
      end
    else
      begin
        for P := 0 to Bytes do
        case Buffer[P] of
            #10:
            begin
                Form3.Memo2.lines.add('');
                Inc(FCurrentLine);
                Inc(nth);
                if (nth<=HolEnd+1)and(nth<=Form1.Memo10.Lines.
  Count) then
                    begin
                        drawback(Form1.Memo10.Lines[nth-1]);
                    end;
            end;
```

```
                    #13:begin
                        end
                          else
                              begin
                                    Form3.Memo2.Lines[FCurrentLine] :=
    Form3.Memo2.Lines[FCurrentLine] + Buffer[P];
                                    Form3.Memo2.Refresh;
                                end;
                      end; //for
                   end;//end else มีข้อมูลมา
                  i:=i+1;
              end; //while <>'END'
              round:=round+1;
              if round < StrToInt(Edit2.text) then
                begin
                   ShowMessage('Please change PCB'+IntToStr(round+1));
                   nth:=HolSta;
                   Form2.refresh;
                end;
          end;
Form2.Label2.Font.Color:=clBlack;
Form2.Label1.Font.Color:=clBlack;
end;
 Comm1.Close;
 Form3.Memo1.Lines.Add('Device closed: ' + Comm1.DeviceName);
 ATool:=0;
end;

procedure TForm5.FormDestroy(Sender: TObject);
begin
    Application.OnException := nil;
end;
```

```
procedure TForm5.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    Comm1.Close;
    Form3.Memo1.Lines.Add('Device closed: ' + Comm1.DeviceName);
    Form3.Memo2.Clear;
    FCurrentLine:=0 ;
    Memo1.Clear;
    Memo2.Clear;
    Form1.Show;
end;

procedure TForm5.Comm1Break(Sender: TObject);
begin
    Form3.Memo1.Lines.add('Break signal detected...');
end;

procedure TForm5.Comm1RxFlag(Sender: TObject);
begin
    Form3.Memo1.Lines.add('RxFlag signal detected...');
end;

procedure TForm5.Comm1TxEmpty(Sender: TObject);
begin
    Form3.Memo1.Lines.add('TxEmpty signal detected...');
end;

procedure TForm5.Comm1CTS(Sender: TObject),
begin
    Form3.Memo1.Lines.add('CTS: ' + OnOff[ord(Comm1.CTS)]);
end;

procedure TForm5.Comm1DSR(Sender: TObject);
```

```
begin
    Form3.Memo1.Lines.add('DSR: ' + OnOff[ord(Comm1.DSR)]);
end;

procedure TForm5.Comm1Ring(Sender: TObject);
begin
    Form3.Memo1.Lines.add('RING: ' + OnOff[ord(Comm1.Ring)]);
end;

procedure TForm5.Comm1RLSD(Sender: TObject);
begin
    Form3.Memo1.Lines.add('RLSD: ' + OnOff[ord(Comm1.RLSD)]);
end;

procedure TForm5.Comm1Error(Sender: TObject; Errors: Integer);
begin
    if (Errors and CE_BREAK > 0) then
        Form3.Memo1.Lines.add('The hardware detected a break condition.');
    if (Errors and CE_DNS > 0) then
        Form3.Memo1.Lines.add('Windows 95 only: A parallel device is not selected.');
    if (Errors and CE_FRAME > 0) then
        Form3.Memo1.Lines.add('The hardware detected a framing error.');
    if (Errors and CE_IOE > 0) then
        Form3.Memo1.Lines.add('An I/O error occurred during communications with
            the device.');
    if (Errors and CE_MODE > 0) then
        begin
            Form3.Memo1.Lines.add('The requested mode is not supported, or
            the hFile parameter');
            Form3.Memo1.Lines.add('is invalid. If this value is specified, it is the only
            valid error.');
        end;
```

```
    If (Errors and CE_OOP > 0) then
            Form3.Memo1.Lines.add('Windows 95 only: A parallel device signaled that it
            is out of paper.');
    if (Errors and CE_OVERRUN > 0) then
            Form3.Memo1.Lines.add('A character-buffer overrun has occurred. The next
            character is lost.');
    if (Errors and CE_PTO > 0) then
            Form3.Memo1.Lines.add('Windows 95 only: A time-out occurred on a parallel
             device.');
    if (Errors and CE_RXOVER > 0) then
            begin
                    Form3.Memo1.Lines.add('An input buffer overflow has occurred.
                     There is either no');
                    Form3.Memo1.Lines.add('room in the input buffer, or a character was
                    received after');
                    Form3.Memo1.Lines.add('the end-of-file (EOF) character.');
            end;
    if (Errors and CE_RXPARITY > 0) then
            Form3.Memo1.Lines.add('The hardware detected a parity error.');
    If (Errors and CE_TXFULL > 0) then
            begin
                    Form3.Memo1.Lines.add('The application tried to transmit a
                    character, but the output');
                    Form3.Memo1.Lines.add('buffer was full.');
            end;
    end;

procedure TForm5.HandleException(Sender: TObject; E: Exception);
begin
    If E is ECommError then
    with E as ECommError do
            ShowMessage('Async32 error: ' + Message);
```

```
    end;

procedure TForm5.Button1Click(Sender: TObject);
begin
    Form3.show;
end;

procedure DrawBack(milxy:string);
var  pa,pb,pc,pd:integer;
    a,b,c,d:integer;
begin
    pa:=pos('a',milxy);
    pb:=pos('b',milxy);
    pc:=pos('c',milxy);
    pd:=pos('d',milxy);
    a:=StrToInt(copy(milxy,pa+1,pb-pa-1));
    b:=StrToInt(copy(milxy,pb+1,pc-pb-1));
    c:=StrToInt(copy(milxy,pc+1,pd-pc-1));
    d:=StrToInt(copy(milxy,pd+1,5));
    Form2.Canvas.Pen.Color:=clRed;
    Form2.Canvas.Arc(a,b,c,d,0,0,0,0);
end;

procedure TForm5.Button4Click(Sender: TObject);
begin
    Form4.show;
end;

procedure TForm5.FormActivate(Sender: TObject);
begin
    Edit1.Text:='';
    Edit3.Text:='';
    Memo1.Clear;
```

```
        Memo2.Clear;
end;
end.
```

โค้ดโปรแกรมควบคุมสเต็ปเปอร์มอเตอร์

```
#include <reg51.h>
#include <string.h>
#include <absacc.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>

#define Pa    XBYTE [0xE0E0]   /* //ship 255 port A              */
#define Pb    XBYTE [0xE0E1]   /* //ship 255 port B              */
#define Pc    XBYTE [0xE0E2]   /* //ship 255 port C              */
#define Pcon  XBYTE [0XE0E3]   /*          //ship 255 register control */
#define LCD_CONTROL   XBYTE[0xE0C0]
#define LCD_BUSY_FLAG XBYTE[0xE0C1]
#define LCD_WRITEDATA XBYTE[0xE0C2]
#define LCD_READDATA  XBYTE[0xE0C3]
#define StartStep 10
#define ZSpeed 400
#define ZstepDrill 50
#define milpermm 39.37007874
#define mmperstep 0.261
#define milperstep 10.27559055
#define ZstepDrill 50
void delay(unsigned int time)
        {
                unsigned int i;
                for(i=0;i < time;i++)
                {
                }
        }
```

```
void wait_for_lcd_ready (void)
        {
while ((LCD_BUSY_FLAG & 0x80) != 0)
   {
   }
}

void specified_lcd(void)
{
        LCD_CONTROL=0x38;
         wait_for_lcd_ready ();
}

void clear_lcd(void)
{
        LCD_CONTROL=0x01;
        //bit0=1 clear lcd
        //and cursor goto top left
         wait_for_lcd_ready ();
}

void cursor_on(void)
{
 LCD_CONTROL = 0x0f;
 wait_for_lcd_ready ();
}

void cursor_off(void)
{
        LCD_CONTROL = 0x0c;
        wait_for_lcd_ready ();
}
```

```
void entry_mode_set(void)
{
LCD_CONTROL = 0x02;
//bit0=0 after put data shift cursor to right
//bit1=1 after read/write dd ram address increase1
wait_for_lcd_ready ();
}

void init_lcd(void)
{
        specified_lcd();
        cursor_on();
        clear_lcd();
        entry_mode_set();
}

void WChar(char ch)
{
        LCD_WRITEDATA = ch;
        wait_for_lcd_ready();
}

void gotoxy(unsigned char row, unsigned char col)
{
if (row == 1) LCD_CONTROL = 0x80+col-1; else
if (row == 2) LCD_CONTROL = 0xc0+col-1; else
if (row == 3) LCD_CONTROL = 0x90+col-1; else
if (row == 4) LCD_CONTROL = 0xd0+col-1;
wait_for_lcd_ready ();
}
```

```
void WCharxy(unsigned char row,unsigned char col,unsigned char ch)
{
gotoxy(row,col);
WChar(ch);
}

void string_to_lcd (unsigned char row,unsigned char col,char *st)
{
data unsigned char mcol,colfinish,len;
//clear_lcd();
len = strlen(st);
colfinish=col+len-1;
if (colfinish > 16)  colfinish=16;
for (mcol=col; mcol<=colfinish; mcol++)
   {
   WCharxy(row,mcol,st[mcol-col]);
   }
}

void string_to_lcd32 (char *st)
{
data unsigned char mcol,colfinish,len,i;
xdata char display_lcd[32];
for (i=0;i<=31;i++)
   {
   display_lcd[i]=' ';
   }
len = strlen(st);
if (len>32) len=32;
for (i=0;i<=len-1;i++)
   {
```

```
    display_lcd[i]=st[i];
    }
for (mcol=1; mcol<=16; mcol++)
    {
    WCharxy(1,mcol,display_lcd[mcol-1]);
    }
for (mcol=1; mcol<=16; mcol++)
    {
    WCharxy(2,mcol,display_lcd[(mcol+16)-1]);
    }
}

void string_to_lcd32RotateLeft (char *st)
{
    data unsigned char mcol,colfinish,len,i,temp;
    xdata char display_lcd[32];
    for (i=0;i<=31;i++)
            {
                    display_lcd[i]=' ';
            }
len = strlen(st);
if (len>32) len=32;
for (i=0;i<=len-1;i++)
    {
    display_lcd[i]=st[i];
    }
while (1)
{
string_to_lcd32(display_lcd);
temp=display_lcd[0];
for (i=1;i<=31;i++)
```

```
{
display_lcd[i-1]=display_lcd[i];
}
display_lcd[31]=temp;
delay(10000);
}
}

void XStop(void)
{
while(1)
        {
        Pa=0x00;
        }
}

void YStop(void)
        {
        while(1)
                {
                Pb=0x00;
                }
        }

void ZStop(void)
{
        while(1)
                {
                        Pc=0x00;
                }
}
```

```
void Xbackward(int XStepNo)/*Move forward on x axis*/
        {
        unsigned int Tround,XSpeed,i=0;
        unsigned int round;
        char stepx[2];
         round=1;
         Tround=XStepNo/4;
        XSpeed=800;
        Pa=0x30;
        stepx[0]='D';
        while(i<XStepNo)
                {
                        if(round<StartStep)
                        XSpeed=XSpeed-50;
         else if((round>=StartStep)&&(round<=(Tround-StartStep))) //rotation
                XSpeed=XSpeed+0;
        Pa=0x31;
        delay(XSpeed);
        i++;
        if(i<XStepNo)
          {
                 Pa=0x32;
                delay(XSpeed);
                i++;
                if(i<XStepNo)
                   {
                        Pa=0x34;
                       delay(XSpeed);
                        i++;
                        if (i<XStepNo)
        {
            Pa=0x38;
```

```
        delay(XSpeed);
        i++;
        round=round+1;
                    }else
                            {
                                    Pa=0x30;
                                    delay(XSpeed);
                                    stepx[0]='C';
                            }/*end if step4*/
                             }else
                                    {
                                    Pa=0x30;
                                    delay(XSpeed);
                                    stepx[0]='B';
                                    }/*end if step3*/
                }else
                        {
                            Pa=0x30;
                            delay(XSpeed);
                            stepx[0]='A';
                            }/*end if step2*/
    }/*end while*/
    Pa=0x30;
    //delay(XSpeed);
    //string_to_lcd(1,1,stepx);
    }/*end of Xbackward function */

  void Xforward(int XStepNo)/*Move backward on x axis*/
    {
unsigned int XSpeed,round,Tround,i=0;
      char stepx[2];
      round=1;
```

```
Tround=XStepNo/4;
XSpeed=800;
Pa=0x30;
stepx[0]='D';
while(i<XStepNo)
 {
 if(round<StartStep)
        XSpeed=XSpeed-50;
        else if((round>=StartStep)&&(round<=(Tround-StartStep)))
                XSpeed=XSpeed+0;
                else XSpeed=XSpeed+50;
          Pa=0x38;
          delay(XSpeed);
          i++;
          if(i<XStepNo)
                {
                 Pa=0x34;
                delay(XSpeed);
                     i++;
                     if(i<XStepNo)
                        {
                              Pa=0x32;
                            delay(XSpeed);
                              i++;
                           if (j<XStepNo)
                            {
                            Pa=0x31;
                            delay(XSpeed);
                            i++;
                                    round=round+1;
                                    }else
                                                {
```

```
                Pa=0x30;
                delay(XSpeed);
                stepx[0]='C';
                }/*end if step4*/
                }else
                {
                  Pa=0x30;
                  delay(XSpeed);
                  stepx[0]='B';
                }/*end if step3*/
                }else
                        {
                        Pa=0x30;
                        delay(XSpeed);
                        stepx[0]='A';
                        }/*end if step2*/
}/*end of while*/
Pa=0x30;
//delay(XSpeed);
//string_to_lcd(2,1,stepx);
}/*end of Xforward function*/

 void Ybackward(int YStepNo)/*Move forward on y axis*/
{
unsigned int YSpeed,round,Tround,i=0;
char stepy[2];
Pb=0x30;
round=1;
Tround=YStepNo/4;
YSpeed=800;
stepy[0]='D';
while(i<YStepNo)
```

```
{
if(round<StartStep)
        YSpeed=YSpeed-50;
        else if((round>=StartStep)&&(round<=(Tround-StartStep)))
                YSpeed=YSpeed+0;
                else YSpeed=YSpeed+50;
               Pb=0x31;
        delay(YSpeed);
               i++;
               if(i<YStepNo)
                {
                        Pb=0x32;
                  delay(YSpeed);
                        i++;
                       if(i<YStepNo)
                         {
                                Pb=0x34;
                    delay(YSpeed);
                                i++;
                                if (i<YStepNo)
                       {
                         Pb=0x38;
                      delay(YSpeed);
                                        i++;
                round=round+1;
                                }else
                                                {
                                                Pb=0x30;
                                                stepy[0]='C';
                                                }/*end if stepf4*/
                        }else
                                {
```

```
                                Pb=0x30;
                                stepy[0]='B';
                                    } /*end if step3*/
                        }else
                        {
                         Pb=0x30;
                         stepy[0]='A';
                        }/*end if step2*/
  }/*end while*/
Pb=0x30;
  }/*end of Ybackward function */

void Yforward(int YStepNo)/*Move backward on y axis*/
  {
          unsigned int YSpeed,round,Tround,i=0;
          char stepy[2];
          Pb=0x30;
          round=1;
          Tround=YStepNo/4;
          YSpeed=800;
          stepy[0]='D';
          while(i<YStepNo)
          {
          if(round<StartStep)
                          YSpeed=YSpeed-50;
          else if((round>=StartStep)&&(round<=(Tround-StartStep)))
                   YSpeed=YSpeed+0;
                  else YSpeed=YSpeed+50;
    Pb=0x38;
    delay(YSpeed);
    i++;
    if(i<YStepNo)
```

```
{
    Pb=0x34;
    delay(YSpeed);

        i++;
         if(i<YStepNo)
          {
                Pb=0x32;
        delay(YSpeed);
                i++;
                if(i<YStepNo)
        {
         Pb=0x31;
         delay(YSpeed);
         i++;
                    round=round+1;
                      }else
                                {
                           Pb=0x30;
                                stepy[0]='C';
                               }/*end if step4*/
        }else
                {
                 Pb=0x30;
                          stepy[0]='B';
                 } /*end if step3*/
   }else
        {
        Pb=0x30;
        stepy[0]='A';
        }/*end if step2*/
}/*end while*/
```

```
Pb=0x30;
}/*end of Yforward function */
void Zup(int ZStepNo)/*Move down on z axis*/
{
 unsigned int i=0;
 char stepz[2];
 Pc=0x00;
 stepz[0]='D';
 while(i<ZStepNo)
        {
        Pc=0x01;
delay(ZSpeed);
        i++;
        if(i<ZStepNo)
          {
                Pc=0x02;
            delay(ZSpeed);
                i++;
                if(i<ZStepNo)
                  {
                    Pc=0x04;
              delay(ZSpeed);
                        i++;
                        if (i<ZStepNo)
                      {
                      Pc=0x08;
                      delay(ZSpeed);
                      i++;
                            }else
                                  {
                                   Pc=0x00;
                                   stepz[0]='C';
```

```
}/*end if stepf4*/
                }else
                        {
                         Pc=0x00;
                                stepz[0]='B';
                        } /*end if step3*/
                }else
                 {
                        Pc=0x00;
                        stepz[0]='A';
                 }/*end if step2*/
        }/*end while*/
        Pc=0x00;
  //string_to_lcd(1,1,stepz);
}/*end of Zup function */

void Zdown(int ZStepNo)/*Move up on z axis*/
{
unsigned int i=0;
 char stepz[2];
 Pc=0x00;
 stepz[0]='D';
 while(i<ZStepNo)
        {
                Pc=0x08;
   delay(ZSpeed);
                i++;
          if(i<ZStepNo)
                {
                 Pc=0x04;
                 delay(ZSpeed);
                 i++;
```

```
                if(i<ZStepNo)
                        {
                         Pc=0x02;
                delay(ZSpeed);
                        i++;
                        if (i<ZStepNo)
                {
                 Pc=0x01;
                 delay(ZSpeed);
                 i++;
                        }else
                                {
                                Pc=0x00;
                                stepz[0]='C';
                                }/*end if stepf4*/
                }else
                 {
                  Pc=0x00;
                                stepz[0]='B';
                 } /*end if step3*/
               }else
                {
                Pc=0x00;
                stepz[0]='A';
                }/*end if step2*/
        }/*end while*/
        Pc=0x00;
        //string_to_lcd(2,1,stepz);
}/*end of Zdown function */

void HomeX(void)
{
```

```
unsigned int XSpeed=200;
Pa=0x30;
while(Pa & 0x40 != 0x40)/*check bit for x axis sensor not high */
  {
        Pa=0x38;
        delay(XSpeed);
         if(Pa & 0x40 != 0x40)/*check bit for x axis sensor not high */
                {
                 Pa=0x34;
                 delay(XSpeed);
                  if(Pa & 0x40 != 0x40)/*check bit for x axis sensor not high */
                        {
                 Pa=0x32;
                 delay(XSpeed);
                f(Pa & 0x40 != 0x40)/*check bit for x axis sensor not high */
                        {
                          Pa=0x31;
                          delay(XSpeed);
                        }else Pa=0x30;  /*end if 3*/
                }else Pa=0x30;  /*end if 2 */
        }else Pa=0x30;  /*end if 1*/
}/*end while*/
Pa=0x30;
} /*end HomeX*/

void HomeY(void)
unsigned int YSpeed=200;
Pb=0x30;
while(Pb & 0x40 != 0x40)/*check bit for y axis sensor not high */
        {
 Pb=0x38;
```

```
            delay(YSpeed);
            if(Pb & 0x40 != 0x40)/*check bit for y axis sensor not high */
                    {
                     Pb=0x34;
                     delay(YSpeed);
                      if(Pb & 0x40 != 0x40)/*check bit for y axis sensor not high */
                             {
                              Pb=0x32;
                              delay(YSpeed);
                             if(Pb & 0x40 != 0x40)/*check bit for y axis sensor not high */
                             {
                                      Pb=0x31;
                                      delay(YSpeed);
                                     }else Pb=0x30;  /*end if 3*/
                             }else Pb=0x30;  /*end if 2 */
                    }else Pb=0x30;  /*end if 1*/
            }/*end while*/
 Pb=0x30;
}/*end HomeY*/

void HomeZ(void)
{
 Pc=0x30;
while(Pc & 0x40 != 0x40)/*check bit for z axis sensor not high */
  {
          Pc=0x38;
          delay(ZSpeed);
           if(Pc & 0x40 != 0x40)/*check bit for z axis sensor not high */
                    {
                     Pc=0x34;
                     delay(ZSpeed);
```

```
if(Pc & 0x40 != 0x40)/*check bit for z axis sensor not high */
{
        Pc=0x32;
        delay(ZSpeed);
        if(Pc & 0x40 != 0x40)/*check bit for z axis sensor not high */
                {
                 Pc=0x31;
                 delay(ZSpeed);
                }else Pc=0x03;  /*end if 3*/
}else Pc=0x30;  /*end if 2 */
}else Pc=0x30;  /*end if 1*/
 }/*end while*/
Pc=0x30;
}/*end HomeZ*/

void HomeAll(void)
{
 HomeX();
HomeY();
HomeZ();
}

void Drill(void)
{
Zdown(ZstepDrill);
Zup(ZstepDrill);
}

void Move(int Xstep,int Ystep,int stepcase)
{
switch(stepcase)
   {
```

```
        case 1: Xforward(Xstep); Yforward(Ystep);break;
        case 2: Xforward(Xstep); Ybackward(Ystep);break;
        case 3: Xbackward(Xstep); Yforward(Ystep);break;
        case 4: Xbackward(Xstep); Ybackward(Ystep);break;
        }
      }

void init_serial(void)
        {
        SCON = 0x52;
        PCON = 0x00;
        TMOD = 0x20;
        TH1 = 0xFD;
        TR1 = 1;
        }

 void main(void)
        {
        unsigned char mBUF,mess[10],Strdat[20],*pmess,Xvalue[15],*pXvalue;
        unsigned char Xstr[10],Ystr[10];
        unsigned int i,j,k,ix,iy;
        unsigned int Xint,Yint; //integer value
        unsigned int Xprev,Yprev,Xact,Yact; //motor step value
        delay(2000);
         Pcon=0x80;
        init_lcd();
        init_serial();
        Pa=0x00;//Preset Port A
        Pb=0x00;//Preset Port B
```

```
Pc=0x00;//Preset Port C
strcpy(Strdat," ");//Clear Array
strcpy(Xstr," ");
strcpy(Ystr," ");
Xprev=0;
Yprev=0;
Xact=0;
Yact=0;
i=0;
j=0;
k=0;
ix=0;
iy=0;
clear_lcd();
while(1)
{
  while(RI!=1)  //have data?
{
 }
RI = 0; //get data
mess[0]=SBUF; //transfer data from sbuf register
if(mess[0]==13)
 WChar('X');
 if(mess[0]!=10)//is it 's not end of Line data?
        {
        /*check only X,Y and 0-9*/
        if(mess[0]>='0' && mess[0]<='9'||mess[0]=='X'||mess[0]=='Y')//|| mess[0]=='E')
         {
          Strdat[k]=mess[0]; //get a string of a point
          k=k+1;
         }else //if(mess[0]!='E')
                {
```

```
Strdat[k]='';
pmess=Strdat; //Pointer point to the string
            if(*pmess=='X')//check X
                    pmess=pmess+1;
          while(*pmess>='0' && *pmess<='9')
                    {
                    Xstr[ix]=*pmess; //get X value in string
                    pmess=pmess+1;
                    ix=ix+1;
                    }
      if(*pmess=='Y')//check Y
              pmess=pmess+1;
  while(*pmess>='0'&& *pmess<='9')
          {
                    Ystr[iy]=*pmess;//check Y value in string
                    pmess=pmess+1;
                    iy=iy+1;
          }
          Xstr[ix]=0;
          Ystr[iy]=0;
          Xint=atoi(Xstr);//get X value in integer
          Yint=atoi(Ystr);//get Y value in integer
          if(Xint>Xprev)
                    {
                   Xact=Xint-Xprev;
                   if(Yint>Yprev)
                    {
                    Yact=Yint-Yprev;
                    Move(Xact,Yact,1);
                    }else{
        Yact=Yprev-Yint;
     Move(Xact,Yact,2);
```

```
}
                    }else
                {
                        Xact=Xprev-Xint;
                        if(Yint>Yprev)
                            {
                             Yact=Yint-Yprev;
                             Move(Xact,Yact,3);
                            }else {
                             Yact=Yprev-Yint;
                             Move(Xact,Yact,4);
                                }
                            }
                        delay(500);
                       if (Xact != Xprev && Yact != Yprev)
                        {
                        Drill();
                        delay(500);
                        printf("X%dY%d\n",Xint,Yint); //Sent to Interface Program
                         }else printf("X%dY%d\n",Xint,Yint);
                        Xprev=Xint;
                        Yprev=Yint;
                       k=0;
                       ix=0;
                       iy=0;
                      }
                }
            }
}
```