DAFTAR PUSTAKA

Gibbs, David and Thomas M. Crandell., 1991, *Dasar-dasar Teknik & Pemrograman CNC,* PT. Rosda Jayaputra, Jakarta.

Hollebrandse, J.J.M., 1988, *Teknik Pemrograman dan Aplikasi CNC,* PT. Rosda Jayaputra, Jakarta.

Kief, Hans B. and Waters, T. Frederick., 1992, *Computer Numerical Control,* McGraw-Hill Book Co., Singapore.

Komputer, Wahana, 2003, *Pemrograman Visual Basic 6.0*, Penerbit Andi, Semarang

Krar, Steve dan Gill, Arthur., 1990, *CNC Technology & Programming,* McGraw-Hill Book Co., Singapore.

Pandey, Dr. P.C. and Singh, Dr. C.K., 2001, *Production Engineering & Science*, Standard Publishers Distributors, Delhi.

Pressman, Roger S. and Williams, John E., 1977, *Numerical Control & Computer Aided Manufacturing*, John Wiley & Sons, USA.

Pusztai, Joseph and Sava, Michael., 1983, *Computer Numerical Control,* Reston Publishing Company Inc., Virginia.

LAMPIRAN

LAMPIRAN 1
DAFTAR DAN KETERANGAN
SOURCE CODE PROGRAM

Form1 (Tampilan Utama.frm)

```
Private Sub Command1_Click()
Form2.Show
Unload Me
End Sub
```

Keterangan:
Prosedur ini akan membuka aplikasi *form2* (Simulasi.form) jika *command1* diklik.

```
Private Sub Command2_Click()
Form3.Show vbModal
End Sub
```

Keterangan:
Prosedur ini akan membuka aplikasi *form2* (Seputar.form) jika *command2* diklik.

```
Private Sub Command3_Click()
Unload Me
End Sub
```

Keterangan:
Prosedur ini akan menutup *form1* (Tampilan Utama.frm) sekaligus mengakhiri keseluruhan program jika *command3* diklik

```
Option Explicit ' deklarasi berasal dari module1
Dim LastX, LastY, LastZ, LastZ1, LastX1, LastY1 As Long ' deklarasi umum di semua sub lastx, last y dan scaleset
Dim scaleSet, PD As Single
```

Keterangan:
Sebelum program dijalankan, variabel utama yang akan digunakan dideklarasikan terlebih dahulu. Perintah *Option Explicit* digunakan untuk menyatakan bahwa variabel tersebut dapat diakses walaupun berada diluar *form* ini. Variabel lainnya yang dideklarasikan dapat digunakan sebagai variabel umum pada *form* ini.

```
Public Sub InitCoord()
  Dim cx, cy, cz, XPos, YPos, ZPos, i As Long 'deklarasi
  Picture1.ScaleMode = 3 'scale mode (3)= pixel
  Picture1.DrawWidth = 1 'ketebalan garis sumbu
  Picture1.DrawStyle = 0 'tipe garis (0)= solid
  Picture1.ForeColor = QBColor(0) 'warna garis sumbu (0) = hitam
  cx = Int(Picture1.ScaleWidth / 2) 'cx didefinisikan
  cy = Int(Picture1.ScaleHeight / 2) 'cy didefinisikan
  Picture1.Cls 'clear screen
  Picture1.Scale (-cx, cy)-(cx, -cy) 'cakupan skala
  Picture1.Line (-cx, 0)-(cx, 0) 'gambar garis horisontal sumbu X
```

```
Picture1.Line (0, -cy)-(0, cy) 'gambar garis vertikal sumbu Y
Picture1.CurrentX = -Picture1.TextWidth(0) - 8 'letak posisi pada sumbu X untuk angka nol
Picture1.CurrentY = Picture1.TextHeight(0) + 5 'letak posisi pada sumbu Y untuk angka nol
Picture1.Print 0 'perintah mencetak angka nol

i = 0
Do While i < cx
   i = i + 10
   If i Mod 50 = 0 Then 'perintah untuk menggambar garis vertikal
      Picture1.ForeColor = &HE0E0E0
      Picture1.Line (-i, -cy)-(-i, cy)
      Picture1.Line (i, -cy)-(i, cy)
      Picture1.ForeColor = &H80000012
      Picture1.DrawWidth = 3
   Else
      Picture1.DrawWidth = 1
   End If
   If i Mod 100 = 0 Then 'perintah manggambar teks/angka
      Picture1.CurrentY = Picture1.TextHeight(i) + 5
      Picture1.CurrentX = i - Picture1.TextWidth(i) / 2
      Picture1.Print i
      Picture1.CurrentY = Picture1.TextHeight(-i) + 5
      Picture1.CurrentX = -Picture1.TextWidth(-i) / 2 - i
      Picture1.Print -i
   End If
   Picture1.Line (-i, -1)-(-i, 1)
   Picture1.Line (i, -1)-(i, 1)
Loop

i = 0
Do While i < cy
   i = i + 10
   If i Mod 50 = 0 Then 'perintah untuk menggambar garis vertikal
      Picture1.ForeColor = &HE0E0E0
      Picture1.Line (-cx, -i)-(cx, -i)
      Picture1.Line (-cx, i)-(cx, i)
      Picture1.ForeColor = &H80000012
      Picture1.DrawWidth = 3
   Else
      Picture1.DrawWidth = 1
   End If
   If i Mod 100 = 0 Then 'perintah manggambar teks/angka
      Picture1.CurrentX = -Picture1.TextWidth(i) - 8
      Picture1.CurrentY = i - Picture1.TextHeight(i) / 2 - 1
      Picture1.Print i
      Picture1.CurrentX = -Picture1.TextWidth(-i) - 8
      Picture1.CurrentY = -i - Picture1.TextHeight(i) / 2 - 1
      Picture1.Print -i
   End If
   Picture1.Line (-1, i)-(1, i) 'mengggambar garis strip pada sumbu koordinat
   Picture1.Line (-1, -i)-(1, -i)
Loop
Picture1.PSet (0, 0), QBColor(11)
LastX = 0
LastY = 0
```

```
Picture2.ScaleMode = 3
Picture2.DrawWidth = 1
Picture2.DrawStyle = 0
Picture2.ForeColor = QBColor(0)
cx = Int(Picture2.ScaleWidth / 2)
cz = Int(Picture2.ScaleHeight / 2)
Picture2.Cls
Picture2.Scale (-cx, cz)-(cx, -cz)
Picture2.Line (-cx, 0)-(cx, 0)
Picture2.Line (0, -cz)-(0, cz)
Picture2.CurrentX = -Picture2.TextWidth(0) - 8
Picture2.CurrentY = Picture2.TextHeight(0) + 5
Picture2.Print 0
i = 0
Do While i < cx
   i = i + 10
   If i Mod 50 = 0 Then
      Picture2.ForeColor = &HE0E0E0
      Picture2.Line (-i, -cy)-(-i, cy)
      Picture2.Line (i, -cy)-(i, cy)
      Picture2.ForeColor = &H80000012
      Picture2.DrawWidth = 3
   Else
      Picture2.DrawWidth = 1
   End If
   If i Mod 100 = 0 Then
      Picture2.CurrentY = Picture2.TextHeight(i) + 5
      Picture2.CurrentX = i - Picture2.TextWidth(i) / 2
      Picture2.Print i
      Picture2.CurrentY = Picture2.TextHeight(-i) + 5
      Picture2.CurrentX = -Picture2.TextWidth(-i) / 2 - i
      Picture2.Print -i
   End If
   Picture2.Line (-i, -1)-(-i, 1)
   Picture2.Line (i, -1)-(i, 1)
Loop
i = 0
Do While i < cz
   i = i + 10
   If i Mod 50 = 0 Then
      Picture2.ForeColor = &HE0E0E0
      Picture2.Line (-cx, -i)-(cx, -i)
      Picture2.Line (-cx, i)-(cx, i)
      Picture2.ForeColor = &H80000012
      Picture2.DrawWidth = 3
   Else
      Picture2.DrawWidth = 1
   End If
   If i Mod 100 = 0 Then
      Picture2.CurrentX = -Picture2.TextWidth(i) - 8
      Picture2.CurrentY = i - Picture2.TextHeight(i) / 2 - 1
      Picture2.Print i
      Picture2.CurrentX = -Picture2.TextWidth(-i) - 8
```

```
        Picture2.CurrentY = -i - Picture2.TextHeight(i) / 2 - 1
        Picture2.Print -i
      End If
      Picture2.Line (-1, i)-(1, i)
      Picture2.Line (-1, -i)-(1, -i)
    Loop
    Picture2.PSet (0, 0), QBColor(11)
    LastX = 0
    LastZ = 0


    Picture3.ScaleMode = 3 'scale mode (3)= pixel
    Picture3.DrawWidth = 1 'ketebalan garis sumbu
    Picture3.DrawStyle = 0 'tipe garis (0)= solid
    Picture3.ForeColor = QBColor(0) 'warna garis sumbu (0) = hitam
    cx = Int(Picture3.ScaleWidth / 2) 'cx didefinisikan
    cy = Int(Picture3.ScaleHeight / 2) 'cy didefinisikan
    Picture3.Cls 'clear screen
    Picture3.Scale (-cx, cy)-(cx, -cy) 'cakupan skala
    Picture3.Line (-cx, 0)-(cx, 0) 'gambar garis horisontal sumbu X
    Picture3.Line (0, -cy)-(0, cy) 'gambar garis vertikal sumbu Y
    Picture3.CurrentX = -Picture1.TextWidth(0) - 8 'letak posisi pada sumbu X untuk angka nol
    Picture3.CurrentY = Picture1.TextHeight(0) + 5 'letak posisi pada sumbu Y untuk angka nol
    Picture3.Print 0 'perintah mencetak angka nol

    i = 0
    Do While i < cx
      i = i + 10
      If i Mod 50 = 0 Then 'perintah untuk menggambar garis vertikal
        Picture3.ForeColor = &HE0E0E0
        Picture3.Line (-i, -cy)-(-i, cy)
        Picture3.Line (i, -cy)-(i, cy)
        Picture3.ForeColor = &H80000012
        Picture3.DrawWidth = 3
      Else
        Picture3.DrawWidth = 1
      End If
      If i Mod 100 = 0 Then 'perintah manggambar teks/angka
        Picture3.CurrentY = Picture1.TextHeight(i) + 5
        Picture3.CurrentX = i - Picture1.TextWidth(i) / 2
        Picture3.Print i
        Picture3.CurrentY = Picture1.TextHeight(-i) + 5
        Picture3.CurrentX = -Picture1.TextWidth(-i) / 2 - i
        Picture3.Print -i
      End If
      Picture3.Line (-i, -1)-(-i, 1)
      Picture3.Line (i, -1)-(i, 1)
    Loop

    i = 0
    Do While i < cy
      i = i + 10
      If i Mod 50 = 0 Then 'perintah untuk menggambar garis vertikal
        Picture3.ForeColor = &HE0E0E0
        Picture3.Line (-cx, -i)-(cx, -i)
```

```
        Picture3.Line (-cx, i)-(cx, i)
        Picture3.ForeColor = &H80000012
        Picture3.DrawWidth = 3
      Else
        Picture3.DrawWidth = 1
      End If
      If i Mod 100 = 0 Then 'perintah manggambar teks/angka
        Picture3.CurrentX = -Picture1.TextWidth(i) - 8
        Picture3.CurrentY = i - Picture1.TextHeight(i) / 2 - 1
        Picture3.Print i
        Picture3.CurrentX = -Picture1.TextWidth(-i) - 8
        Picture3.CurrentY = -i - Picture1.TextHeight(i) / 2 - 1
        Picture3.Print -i
      End If
      Picture3.Line (-1, i)-(1, i) 'mengggambar garis strip pada sumbu koordinat
      Picture3.Line (-1, -i)-(1, -i)
    Loop
    Picture3.PSet (0, 0), QBColor(11)
    LastY = 0
    LastZ = 0
End Sub
```

Keterangan:
Pada prosedur ini dilakukan inisialisasi bidang gambar yang akan digunakan untuk memplot *cutter path*. Prosedur ini ditampilkan dalam bentuk bidang gambar dengan sumbu XY, XZ, dan YZ. Bagian awal prosedur berisi deklarasi bahwa variable cx, cy, cz, Xpos, Ypos, Zpos dan i bertipe *Long*.
Kemudian diset bahwa bidang gambar yang bernama *Picture1* memiliki *ScaleMode* 3 *(pixel)*, *DrawWidth* 1, *DrawStyle* 0 *(solid)*, dan warna bidang gambar QBColour 0 (hitam). Variable cx diisi dengan nilai hasil pembulatan setengah lebar bidang gambar, sedangkan variable cy diisi dengan hasil pembulatan setengah tinggi bidang gambar.
Statement Picture1.cls berarti bidang gambar dibersihkan terlebih dahulu, kemudian diatur bahwa bidang gambar dimulai dari koordinat –cx sampai dengan +cx dan –cy sampai dengan +cy dan digambarkan *axis*nya. Untuk memperjelas ukuran hasil plot, maka pada bidang gambar ditampilkan pula koordinatnya.

```
Private Sub Command2_Click()
Form4.Show vbModal
End Sub
```

Keterangan:
Prosedur ini akan membuka aplikasi *form4* (Penjelasan.form) jika *command2* diklik.

```
Private Sub CETAKKODE_Click()
Dim Msg, PILIH
  If gtext.Text = "" Then
    MsgBox "TIDAK ADA NC PART PROGRAM YANG BISA DICETAK!", , "PERINGATAN!"
  Else
  PILIH = MsgBox("APAKAH ANDA INGIN MENCETAK NC PART PROGRAM?", 4 + 32,
"CETAK FILE")
  If PILIH = 6 Then
    Msg = gtext.Text
    Printer.Print "NC Part Program File", Chr(13)
    Printer.Print commondialog1.FileName, Chr(13)
    Printer.Print Msg  '& Printer.Page & "." ' Print.
```

```
    Printer.NewPage ' Send new page.
    Printer.EndDoc  ' Print done.
  Else
  Print " "
  End If
  End If
End Sub
```

Keterangan:
Prosedur ini akan mencetak *text* yang ada pada layar tampilan *NC Part Program (gtext rich text box)* jika *command* Cetak Kode diklik. Pada awal prosedur dideklarasikan variabel msg dan variable pilih. Dalam prosedur ini ada dua kondisi yang terjadi bila *command* Cetakkode mendapat *eventclick*, yaitu:

1. Kondisi bila dalam pada *gtaxt rich text box* tidak terdapat *text* yang ditampilkan maka program akan menampilkan kotak pesan/ peringatan *(MsgBox)* "TIDAK ADA NC PART PROGRAM YANG BISA DICETAK!".
2. Kondisi bila pada *gtext rich text box* terdapat *text* yang ditampilkan maka akan muncul kotak pesan yang berisi variabel pilih sebagai pertanyaan "Apakah anda ingin mencetak NC Part Program?". Bila variabel pilih bernilai 6 maka akan diset sebagai *command "Yes"* untuk melanjutkan proses mencetak dan bila variabel pilih berisi selain 6 maka akan diset sebagai *command "No"* untuk membatalkan proses mencetak.

```
Private Sub Option1_Click(Index As Integer) 'skala gambar real
scaleSet = 1
InitCoord
Picture1.DrawWidth = 1 'ketebalan garis sumbu
Picture2.DrawWidth = 1
Picture3.DrawWidth = 1
PD = 1
End Sub
```

Keterangan:
Prosedur ini ditampilkan dalam bentuk *option button* untuk menset nilai skala untuk menggambarkan *cutter path* sebesar 1. Besarnya skala disimpan dalam *ScaleSet* yang besarnya ditentukan berdasarkan *index* dari *option button* yang dipilih. Variabel *ScaleSet* ini akan digunakan oleh prosedur Plot XY, XZ, dan YZ. Prosedur *Option1_click* memanggil prosedur *InitCoord* untuk inisialisasi bidang gambar. Dan pengisian variabel PD dengan 1.

```
Private Sub Option2_Click(Index As Integer) 'skala gambar 2 kali besarnya
scaleSet = 0.5
InitCoord
Picture1.DrawWidth = 1.5 'ketebalan garis sumbu
Picture2.DrawWidth = 1.5
Picture3.DrawWidth = 1.5
PD = 1.5
End Sub
```

Keterangan:
Idem dengan prosedur *Private option1_click*. Hanya saja pada prosedur ini nilai *ScaleSet* sebesar 0.5. Pemilihan prosedur ini berarti akan membuat gambar simulasi menjadi 2 kali lebih besar dari ukuran pada *NC Part Program*. Dan juga diikuti pengaturan *DrawWidth* ketebalan garis menjadi 1.5. Dan pengisian variabel PD dengan 1.5.

```
Private Sub Option3_Click(Index As Integer) 'skala gambar 4 kali besarnya
scaleSet = 0.25
```

```
InitCoord
Picture1.DrawWidth = 4  'ketebalan garis sumbu
Picture2.DrawWidth = 4
Picture3.DrawWidth = 4
PD = 4
End Sub
```

Keterangan:
Idem dengan prosedur *Private option1_click*. Hanya saja pada prosedur ini nilai *ScaleSet* sebesar 0.25. Pemilihan prosedur ini berarti akan membuat gambar simulasi menjadi 4 kali lebih besar dari ukuran pada *NC Part Program*. Dan juga diikuti pengaturan *DrawWidth* ketebalan garis menjadi 4. Dan pengisian variabel PD dengan 4.

```
Private Sub cmdclear_Click() 'perintah menghapus G-code pada layar text secara keseluruhan
Dim Tempstr As String 'tempstr dideklarasikan sebagai string
Dim PILIH
  PILIH = MsgBox("SIMPAN FILE NC PROGRAM YANG TELAH ADA PADA LAYAR!", 3 +
32, "PERINGATAN!")
  If PILIH = 6 Then
    With commondialog1
    .DialogTitle = "SAVE NC PROGRAM FILE"
    .Filter = "(*.G;*.txt;*.NC)| *.G;*.txt;*.NC"
    .ShowSave
    Tempstr = .FileName
    End With
    If Tempstr <> "" Then
    gtext.SaveFile Tempstr
    End If
    gtext.Text = ""
    Label1.Caption = ""
    Text1.Text = ""
    ElseIf PILIH = 7 Then
  gtext.Text = ""
  Label1.Caption = ""
  Text1.Text = ""
  Else
  Print ""
  End If
End Sub
```

Keterangan:
Prosedur ini akan membersihkan layar tampilan *NC Part Program* secara keseluruhan jika *commandClear* diklik. Pada awal prosedur ini dideklarasikan variabel *tempstr* sebagai *string* dan variabel pilih. Kemudian dilakukan pengisian bervariabel pilih sebagai nilai masukan kotak pesan/ peringatan *(MsgBox)* "Simpan *file NC Program* yang telah ada pada layar!". Dengan nilai masukan 6, 7, dan 2.
Dalam kotak pesan ini diset untuk 3 kondisi, yaitu:

1. Bila variabel pilih bernilai 6 maka akan diset sebagai *command "Yes"* maka akan dimunculkan kotak dialog *"Save NC Program File"* untuk menyimpan *text* yang ada pada *gtext.richtextbox*. Kemudian diset *gtext.text, label1.caption,* dan *text1.text* dengan kosong/ *clear*.
2. Bila variabel pilih bernilai 7 maka akan diset sebagai *command "No"* maka prosedur akan menset *gtext.text, label1.caption,* dan *text1.text* dengan kosong/ *clear*.

3. Bila variabel pilih bernilai 2 maka akan diset sebagai *command "Cancel"* untuk melakukan perintah *print* dengan nilai kosong sebagai kode untuk membatalkan perintah mencetak kode.

```
Private Sub cmdexit_Click() 'perintah menutup form 1 (millplot)
Dim Tempstr As String 'tempstr dideklarasikan sebagai string
Dim PILIH
If gtext.Text <> "" Then
    PILIH = MsgBox("SIMPAN DULU FILE NC PROGRAM YANG TELAH ADA PADA
LAYAR!", 3 + 32, "PERINGATAN!")
  If PILIH = 6 Then
     With commondialog1
     .DialogTitle = "SAVE NC PROGRAM FILE"
     .Filter = "(*.G;*.txt)| *.G;*.txt"
     .ShowSave
     Tempstr = .FileName
     End With
     If Tempstr <> "" Then
     gtext.SaveFile Tempstr
     End If
     Unload Form2
     Form1.Show
  ElseIf PILIH = 7 Then
     Unload Form2
     Form1.Show
  Else
     Print ""
  End If
Else: gtext.Text = " "
Unload Form2
Form1.Show
End If
End Sub
```

Keterangan:
Prosedur ini akan menutup aplikasi *form2* (Simulasi.*Form*) jika *commandexit* diklik. Pada awal prosedur ini dideklarasikan variabel *tempstr* sebagai *string* dan variabel pilih. Kemudian dilakukan pengecekan kondisi terhadap *gtext.text*. Pada prosedur ini ada 2 kondisi pengecekan, yaitu:
Bila pada *gtext.text* terdapat *text* maka dilakukan pengisian variabel pilih sebagai nilai masukan kotak pesan/ peringatan *(MsgBox)* "Simpan *file NC Program* yang telah ada pada layar!". Dengan nilai masukan 6, 7, dan 2. Dalam kotak pesan ini diset untuk 3 kondisi, yaitu:

1. Bila variabel pilih bernilai 6 maka akan diset sebagai *command "Yes"* maka akan dimunculkan lebih dahulu kotak dialog *"Save NC Program File"* untuk menyimpan *text* yang ada pada *gtext.richtextbox*. Kemudian menutup *form2* (simulasi.frm) dan membuka *form1* (Tampilan Utama.frm).
2. Bila variabel pilih bernilai 7 maka akan diset sebagai *command "No"* maka prosedur akan langsung menutup *form2* (Simulasi.frm) dan membuka *form1* (Tampilan Utama.frm).
3. Bila variabel pilih bernilai 2 maka akan diset sebagai *command "Cancel"* untuk membatalkan perintah menutup *form2* (Simulasi.frm).

Bila pada *gtext.text* tidak terdapat *text* yang aktif maka program dapat langsung menutup *form2* (Simulasi.frm) dan mengaktifkan *form1* (Tampilan Utama.frm).

```
Private Sub Cmdparameter_Click() 'perintah membuka form parameter
Form5.Show vbModal
```

```
End Sub
```

Keterangan:
Prosedur ini akan membuka aplikasi *form5* (Parameter.Form) jika *commandparameter* diklik.

```
Private Sub cmdread_Click() 'perintah membuka file G-CODE
Dim Tempstr As String 'tempstr dideklarasikan sebagai string
Dim PILIH
  If gtext.Text <> "" Then
    PILIH = MsgBox("SIMPAN FILE NC PROGRAM YANG TELAH ADA PADA LAYAR!", 3 +
32, "PERINGATAN!")
      If PILIH = 6 Then
         With commondialog1
         .DialogTitle = "SAVE NC PROGRAM FILE"
         .Filter = "(*.G;*.txt)| *.G;*.txt"
         .ShowSave
         Tempstr = .FileName
         End With
         If Tempstr <> "" Then
         gtext.SaveFile Tempstr
         End If
          With commondialog1
         .DialogTitle = "Open NC Program File" 'judul
         .Filter = "(*.G;*.NC;*.txt)| *.G;*.NC;*.txt" 'filter file
         .ShowOpen
         Tempstr = .FileName
         End With
         If Tempstr <> "" Then
         With gtext
         .LoadFile Tempstr
         .SelStart = 0
         .SelLength = Len(.Text)
         .SelIndent = 15
         .SelStart = 1
         End With
         End If
      ElseIf PILIH = 7 Then
         With commondialog1
         .DialogTitle = "Open NC Program File" 'judul
         .Filter = "(*.G;*.NC;*.txt)| *.G;*.NC;*.txt" 'filter file
         .ShowOpen
         Tempstr = .FileName
         End With
         If Tempstr <> "" Then
         With gtext
         .LoadFile Tempstr
         .SelStart = 0
         .SelLength = Len(.Text)
         .SelIndent = 15
         .SelStart = 1
         End With
         End If
      Else
      Print ""
      End If
```

```
Else
With commondialog1
        .DialogTitle = "Open NC Program File" 'judul
        .Filter = "(*.G;*.NC;*.txt)| *.G;*.NC;*.txt" 'filter file
        .ShowOpen
        Tempstr = .FileName
        End With
        If Tempstr <> "" Then
        With gtext
        .LoadFile Tempstr
        .SelStart = 0
        .SelLength = Len(.Text)
        .SelIndent = 15
        .SelStart = 1
        End With
        End If
End If
End Sub
```

Keterangan:
Prosedur ini akan membuka kotak dialog *"Open NC Program File"* untuk membuka *file* yang berekstensi *.G, *.txt, dan *.NC jika *commandread* diklik. Pada awal prosedur ini dideklarasikan variabel *tempstr* sebagai *string* dan variable pilih. Kemudian dilakukan pengecekan kondisi terhadap *gtext.text*. Pada prosedur ini ada 2 kondisi pengecekan, yaitu:

1. Bila variabel pilih bernilai 6 maka akan diset sebagai *command "Yes"* maka akan dimunculkan lebih dahulu kotak dialog *"Save NC Program File"* untuk menyimpan *text* yang ada pada *gtext.richtextbox*. Kemudian dilakukan prosedur membuka kotak dialog *"Open NC Program File"*.
2. Bila variabel pilih bernilai 7 maka akan diset sebagai *command "No"* maka prosedur akan langsung membuka kotak dialog *"Save NC Program File"*.
3. Bila variabel pilih bernilai 2 maka akan diset sebagai *command "Cancel"* untuk membatalkan perintah membuka kotak dialog *"Open NC Program File"*.

Bila pada *gtext.text* tidak terdapat *text* yang aktif maka program dapat langsung membuka kotak dialog *"Open NC Program File"* untuk membuka *NC Part Program* yang tersimpan.

```
Private Sub cmdsave_Click()
Dim Tempstr As String
If gtext.Text = "" Then
    MsgBox "TIDAK ADA FILE NC PROGRAM YANG BISA DISIMPAN!", , "PERINGATAN!"
Else
  With commondialog1
  .DialogTitle = "SAVE NC PROGRAM FILE"
  .Filter = "(*.G;*.NC;*.txt)| *.G;*.NC;*.txt"
  .ShowSave
  Tempstr = .FileName
  End With
  If Tempstr <> "" Then
    gtext.SaveFile Tempstr
  End If
End If
End Sub
```

Keterangan:
Pada prosedur awal ini *Tempstr* dideklarasikan sebagai *string*. Prosedur ini akan menampilkan kotak dialog untuk menyimpan data yang ada pada layar tampilan *NC Part Program*. Data tersebut

disimpan pada *file* yang berekstensi *.G, *.txt, dan *.NC. Namun bila pada layar tampilan *NC Part Program (gtext)* tidak terdapat program yang aktif, maka akan dimunculkan kotak pesan *(MsgBox)* "Tidak ada *File NC Program* yang bisa disimpan!".

```
Private Sub spicture_Click()
Dim Gambar As String
With CommonDialog2
   .DialogTitle = "SAVE GRAPHIC TO FILE"
   .Filter = "(*.bmp)|*.bmp"
   .DefaultExt = "bmp"
   .ShowSave
   Gambar = .FileName
 If Gambar = "" Then Exit Sub
If CommonDialog2.FileName = "" Then Exit Sub
SavePicture Picture1.Image, CommonDialog2.FileName
End With
End Sub
```

Keterangan:
Pada prosedur awal ini gambar dideklarasikan sebagai *string*. Prosedur ini akan menampilkan kotak dialog untuk menyimpan data yang ada pada *picture1*. Data tersebut disimpan pada *file* yang berekstensi *.bmp.

```
Private Sub cmdSTOP_Click()
RunFlag = False
End Sub
```

Keterangan:
Prosedur ini digunakan untuk menghentikan proses gerakan simulasi alat potong. Dilakukan dengan mengubah status *RunFlag* dari *true* menjadi *false*.

```
Private Sub Command1_Click()
scaleSet = scaleSet
InitCoord
Picture1.DrawWidth = PD   'ketebalan garis sumbu
Picture2.DrawWidth = PD
Picture3.DrawWidth = PD
End Sub
```

Keterangan:
Prosedur ini digunakan untuk membersihkan bidang gambar *(Picture 1, Picture 2,* dan *Picture 3)* setelah digunakan untuk melakukan proses gerakan simulasi alat potong. Prosedur awal adalah melakukan set awal *Initcoord* dilanjutkan pengisian *Picture1.drawwidth, Picture2.drawwidth,* dan *Picture3.drawwidth* dengan variabel PD.

```
Private Sub Form_Load()
On Error Resume Next
Dim instring As String
ChDir App.Path
scaleSet = 1
LastX = 0
LastY = 0
LastX = 0
LastY = 0
End Sub
```

Keterangan:
Prosedur ini yang pertama kali dijalankan saat *window preview* ditampilkan. Pada prosedur ini dilakukan proses inisialisasi dengan pengeturan awal variabel *Instring* bertipe *string*, dan mengisi nilai variabel *ScaleSet* dengan nilai 1, serta variabel *LastX* dan *LastY* dengan nilai 0. Kemudian prosedur tersebut memanggil prosedur *Initcoord* untuk melakukan inisialisasi terhadap bidang gambar sehingga pada saat *window preview* ditampilkan bidang gambar sudah dalam keadaan siap digunakan.

```
Private Sub cmdrun_Click()
Dim i, J, K As Long
Dim Tempstr As String
Dim tokenchr As String
Dim tokenval As Double
Dim tescond As Boolean
Dim isabsolute As Boolean
Dim newcommand, lastcommand, lastmodal As String
Dim valx, valy, valz, valF, vali, valL, valj, valR, valT, vald, valk, ValM As Double
Dim nilai1 As Integer
Dim Radiusmaterial, Tx3, Ty3, Tz3 As Double
Dim Tx1, Tx2, Ty1, Ty2, Tz1 As Double
Dim Tbl1, Tbl2 As Double
Dim PILIH As Single

If gtext.Text = "" Then
   MsgBox "MASUKKAN DULU NC PROGRAM", , "PERINGATAN!"
   With commondialog1
   .DialogTitle = "Open NC Program File" 'judul
   .Filter = "G-code(*.G;*.NC;*.txt)| *.G;*.NC;*.txt" 'filter file
   .ShowOpen
   Tempstr = .FileName
  End With
  If Tempstr <> "" Then
  With gtext
  .LoadFile Tempstr
  .SelStart = 0
  .SelLength = Len(.Text)
  .SelIndent = 15
  .SelStart = 1
  End With
  RunFlag = False
  End If
ElseIf E = 0 And G = 0 And A = 0 And C = 0 Then
    MsgBox "SILAHKAN MELIHAT NILAI PARAMETER UNTUK MASUKAN", , "SET
PARAMETER!"
     Form5.Show vbModal
     'InitCoord
     RunFlag = False
Else
For i = 0 To Form2.Controls.Count - 1
 If TypeOf Form2.Controls(i) Is CommandButton Then Form2.Controls(i).Enabled = False
  Next
  cmdSTOP.Enabled = True
   nilai1 = 1
   newcommand = ""
```

```
lastcommand = ""
lastmodal = ""
isabsolute = True
valx = 0
valy = 0
valz = 0
valF = (C / 60)
valF = 0
vali = 0
valj = 0
valR = 0
valT = 0
vald = 0
RunFlag = True
i = 1
J = Len(gtext.Text)
Do While i <= J And RunFlag
   K = InStr(i, gtext.Text, Chr(13), vbBinaryCompare)
   If K > 0 Then
      Tempstr = Mid(gtext.Text, i, K - i)
      Text1.Text = Tempstr
      Label1.Caption = commondialog1.FileName
       Me.Refresh
       gtext.SelStart = i - 1
       gtext.SelLength = K - i + 2
       i = K + 2
       testcond = True
       newcommand = ""
       Do While testcond And Tempstr <> ""
          tokenchr = ""
          tokenval = 0
          testcond = nexttoken(Tempstr, tokenchr, tokenval)
          If testcond Then
             Select Case tokenchr
                Case "G"
                   newcommand = tokenchr & Trim(Str(tokenval))
                Case "M"
                   newcommand = tokenchr & Trim(Str(tokenval))
                Case "F"
                   valF = tokenval
                Case "T"
                   valT = tokenval
                Case "X"
                   valx = tokenval
                Case "Y"
                   valy = tokenval
                Case "Z"
                   valz = tokenval
                Case "I"
                   vali = tokenval
                Case "J"
                   valj = tokenval
                Case "L"
                   valL = tokenval
                Case "K"
```

```
                valk = tokenval
              Case "D"
                vald = tokenval
              Case "H"
            End Select
          End If
  If silinder = True Then
    Radiusmaterial = (DIAMETERMATERIAL / 2)
    Tx3 = TITIKX2 / scaleSet
    Ty3 = TITIKY2 / scaleSet
    Tbl1 = Tx3 - Radiusmaterial
    Tbl2 = Tx3 + Radiusmaterial
    Tz3 = 0 + (TEBALS / scaleSet)
    Picture1.Circle (Tx3, Ty3), Radiusmaterial, vbBlack
    Picture2.Line (Tbl1, -Tz3)-(Tbl2, 0), vbBlack, B
    Picture3.Line (-Tz3, Tbl1)-(0, Tbl2), vbBlack, B
  Else
    Tx1 = (TITIKX1 / scaleSet) - ((PANJANGX / 2) / scaleSet)
    Tx2 = (TITIKX1 / scaleSet) + ((PANJANGX / 2) / scaleSet)
    Ty1 = (TITIKY1 / scaleSet) - ((PANJANGY / 2) / scaleSet)
    Ty2 = (TITIKY1 / scaleSet) + ((PANJANGY / 2) / scaleSet)
    Tz1 = 0 + (TEBALB / scaleSet)
    Picture1.Line (Tx1, Ty1)-(Tx2, Ty2), vbBlack, B
    Picture2.Line (Tx1, -Tz1)-(Tx2, 0), vbBlack, B
    Picture3.Line (-Tz1, Ty1)-(0, Ty2), vbBlack, B
  End If
        Loop
        If newcommand = "" Then newcommand = lastcommand
        If newcommand <> "" Then
          Select Case UCase(newcommand)
            Case "G80" ' perintah membatalkan program g-code diatasnya
                DrawG80 0, 0, 0, valF, True
                valF = 0
            Case "G0" 'gerakan cepat tanpa pemakanan
                DrawG0 valx, valy, valz, isabsolute
            Case "G1" 'gerakan pemakanan linear
              If valF = 0 Then
                MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
                RunFlag = False
              Else
                DrawG1 valx, valy, valz, valF, isabsolute
              End If
            Case "G2" 'gerakan pemakanan lingkaran searah jarum jam
              If valF = 0 Then
                MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
                RunFlag = False
              Else
                DrawG2 valx, valy, vali, valj, 0.1, valF, isabsolute
              End If
            Case "G3" 'gerakan pemakanan melingkar berlawanan jarum jam
              If valF = 0 Then
                MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
```

```
            RunFlag = False
          Else
            DrawG3 valx, valy, vali, valj, 0.1, valF, isabsolute
          End If
        Case "G54"
          DrawG0 valx, valy, valz, isabsolute
        Case "G81" 'kode pengeboran biasa
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
            DrawG81 valx, valy, valz, vald, isabsolute
          End If
        Case "G82" 'kode pengeboran dengan waktu tinggal
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
            DrawG81 valx, valy, valz, vald, isabsolute
          End If
        Case "G83" 'kode pengeboran penarikan berulang
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
            DrawG81 valx, valy, valz, vald, isabsolute
          End If
        Case "G84" 'kode pembuatan ulir
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
          DrawG84 valx, valy, valz, vald, isabsolute
          End If
        Case "G85" 'kode Reaming
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
          DrawG85 valx, valy, valz, vald, isabsolute
          End If
        Case "G88" 'pocket kotak
          If valF = 0 Then
            MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
            RunFlag = False
          Else
            DrawG88 valx, valy, valz, vali, valj, valk, isabsolute
          End If
        Case "G89" 'pocket silinder
```

```
            If valF = 0 Then
              MsgBox "PROGRAM PADA BLOCK NC PART PROGRAM INI BELUM
LENGKAP. MASUKKAN BESAR KECEPATAN (FEEDING)!", , "PERINGATAN!"
              RunFlag = False
            ElseIf valj = 0 Or valk = 0 Then
              MsgBox "RADIUS(J) DAN JUMLAH STEP(K) PADA PROGRAM G89 TIDAK
BOLEH SAMA DENGAN NOL!", , "PERINGATAN!"
              RunFlag = False
            Else
              DrawG89 valx, valy, valz, valL, valj, valk, isabsolute
            End If

          Case "G90" 'kode absolute
            isabsolute = True
          Case "G91" 'perintah incremental
            isabsolute = False
          Case "M30"
            RunFlag = False
        End Select
        lastcommand = newcommand
      End If
    If newcommand = "G18" Then 'program definisi X dan Y
    MsgBox "PROGRAM INI TIDAK BISA MENJALANKAN G18. SILAHKAN MENGUBAH
G18 MENJADI G17 DENGAN DIIKUTI PERUBAHAN KOORDINAT Z MENJADI KOORDINAT
Y", , "PERINGATAN!"
    RunFlag = False
     If newcommand = "G43" Then newcommand = lastcommand
     End If
    End If
  DoEvents
 Loop
 For i = 0 To Form2.Controls.Count - 1
  If TypeOf Form2.Controls(i) Is CommandButton Then Form2.Controls(i).Enabled = True
 Next
 cmdSTOP.Enabled = False
End If
End Sub
```

Keterangan:
Prosedur ini untuk memerintahkan menggambarkan gerakan simulasi gerakan alat potong pada bidang gambar *Picture1, Picture2,* dan *Picture3* jika *commandrun* diklik. Pada bagian awal dideklarasikan i, J, dan K bertipe *Long*. Variabel *tempstr, tokenchr, newcommand, lastcommand,* dan *lastmodal* bertipe *string*. Variabel *tokenval, valx, valy, valz, valF, vali, valL, valj, valR, valT, vald, valk* bertipe *double*. Sedangkan variabel *TestCond* dan *isabsolute* bertipe *Boolean*.
Kemudian diset apabila *commandrun* diklik maka semua *command* akan diset pasif kecuali *commandstop*. Variabel *newcommand, lastcommand,* dan *lastmodal* dikosongkan terlebih dahulu. Variabel *isabsolute* dan *RunFlag* diset *true*. Variabel *valx, valy, valz, vali, valL, valj, valR, valT, vald, valk* diset 0. Sedangkan *valF* diisi dengan kecepatan pada X dibagi 60. variable i diset bernilai 1, dan nilai j diambilkan dari jumlah karakter pada *gtext.text*.
Dalam prosedur ini diidentifikasi bila nilai resolusi dan kecepatan sama dengan nol (0) maka akan dimunulkan kotak pesan "Silahkan Melihat Nilai Parameter Untuk Masukan!" kemudian diikuti munculnya *form5* (parameter.frm). Kemudian dilakukan proses pembacaan *string* dalam *gtext.text* untuk menangkap karakter perintah (G, F, M, X, Y, Z, I, J, K, L, T, dan D) beserta nilai yang mengikutinya.

Berdasarkan karakter perintah tersebut kemudian dilakukan penggambaran simulasi gerakan alat potong dengan memanggil prosedur *"Draw...."* dengan *input* dari nilai X, Y, Z, I, J, K, L, T, dan D. Namun untuk beberapa perintah gerakan dengan adanya variabel kecepatan *(feeding)* diset bila pada valF masukan nol, maka akan muncul *MsgBox* "Program pada *Block NC Part Program* ini belum lengkap. Masukkan Besar Kecepatan *(Feeding)*!".
Untuk prosedur G18 maka yang akan muncul adalah *MsgBox* "Program ini tidak bisa menjalankan G18. Silahkan mengubah G18 menjadi G17 diikuti perubahan koordinat Z menjadi koordinat Y". Pada saat semua perintah dijalankan, maka semua kontrol akan kembali diaktifkan. Sedangkan *commandstop* akan diset pasif. Untuk prosedur perintah gerakan *DrawG89*, bila pada variabel J *(Radius)* masukan nol, maka akan muncul *MsgBox* "*Radius* (J) pada program G89 Tidak Boleh Sama Dengan Nol!".

```
Public Sub DrawG0(ByVal CoordX, ByVal CoordY, ByVal CoordZ As Double, ByVal Absol As
Boolean)
  Dim stepInt As Long
  Dim StepNum As Long
  Dim StepDir As Boolean
  Dim stepnumX As Long
  Dim StepDirX As Boolean
  Dim stepnumY As Long
  Dim StepDirY As Boolean
  Dim stepnumZ As Long
  Dim StepDirZ As Boolean
  If E = 0 And G = 0 And A = 0 And C = 0 Then
    MsgBox "PROGRAM INI TIDAK BISA MENJALANKAN SILAHKAN MASUKAN NILAI
PARAMETER", , "PERINGATAN!!"
    Form2.Show vbModal
    InitCoord
  End If
  stepnumX = IIf(Absol, Abs(CoordX - LastX), Abs(CoordX))
  stepnumY = IIf(Absol, Abs(CoordY - LastY), Abs(CoordY))
  stepnumZ = IIf(Absol, Abs(CoordZ - LastZ), Abs(CoordZ))
  If stepnumX >= 1 Or stepnumY >= 1 Or stepnumZ >= 1 Then
    StepDirX = IIf((CoordX - LastX) > 0, True, False)
    StepDirY = IIf((CoordY - LastY) > 0, True, False)
    StepDirZ = IIf((CoordZ - LastZ) > 0, True, False)
      If stepnumX > stepnumY Then
        stepInt = A / 10
      ElseIf stepnumX > stepnumZ Then
        stepInt = A / 10
      ElseIf stepnumY > stepnumZ Then
        stepInt = E / 10
      Else
        stepInt = i / 10
      End If
    PlotXY StepDirX, stepnumX, StepDirY, stepnumY, stepInt, vbBlue
    PlotXZ StepDirX, stepnumX, StepDirZ, stepnumZ, stepInt, vbBlue
    PlotYZ StepDirY, stepnumY, StepDirZ, stepnumZ, stepInt, vbBlue
   End If
End Sub
```

Keterangan:
Prosedur DrawG0 berfungsi untuk menggambarkan perintah gerakan G0. Prosedur ini dipanggil oleh *commandrun* dengan *input* berupa nilai koordinat X, Y, dan Z yang bertipe *double* dan koordinat dalam status *absolute*. Pada bagian awal prosedur, dilakukan deklarasi variabel *StepInt*

(yang menyatakan *interval* pengiriman pulsa: kecepatan) dan *StepNum* (yang menyatakan jumlah *step* untuk melakukan gerakan: panjang langkah). *StepNumX, StepNumY,* dan *StepNumZ* bertipe *Long*. Sedangkan untuk variabel *StepDir* (yang menyatakan arah gerakan: positif atau negatif). *StepDirX, StepDirY,* dan *StepDirZ* bertipe *Boolean*.
Proses selanjutnya adalah pengisian variabel *StepNumX, StepNumY* dan *StepNumZ* dengan memeriksa status koordinat gerakan, bila variabel Absol bernilai *true* (yang berarti menggunakan sistem koordinat *absolute*), maka kedua variabel tersebut diisi dengan nilai *absolute* dari selisih nilai *CoordX, CoordY*, dan *CoordZ* dengan *LastX, LastY,* dan *LastZ*. Bila ternyata absol bernilai *false* (berarti menggunakan sistem koordinat relatif), maka kedua variabel tersebut diisi dengan nilai *absolute* dari *CoordX, CoordY,* dan *CoordZ*. Jika nilai dari *StepNumX* atau *StepNumY* atau *StepNumZ* lebih besar atau sama dengan 1 dilakukan pengecekan arah gerakan pada sumbu X, Y, dan Z dengan melihat nilai *(CoordX – LastX), (CoordY – LastY),* dan *(CoordX – LastX)*
Bila memiliki nilai lebih besar dari 0, maka *StepDirX, StepDirY,* dan *StepDirZ* bernilai *true*, dan sebaliknya bernilai *false*. Jika *StepNumX* lebih besar dari *StepNumY*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi X / (Kecepatan X / 60). Jika *StepNumX* lebih besar dari *StepNumZ*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi X / (Kecepatan X / 60). Jika *StepNumY* lebih besar dari *StepNumZ*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi Y / (Kecepatan Y / 60). Kemudian dipanggil prosedur *Plot XY* dengan *input StepDirX, StepNumX, StepDirY, StepNumY, StepInt,* dan *vbBlue, Plot XZ* dengan *input StepDirX, StepNumX, StepDirZ, StepNumZ, StepInt,* dan *vbBlue, Plot YZ* dengan *input StepDirY, StepNumY, StepDirZ, StepNumZ, StepInt,* dan *vbBlue*.

```
Public Sub DrawG1(ByVal CoordX As Double, ByVal CoordY As Double, ByVal CoordZ, ByVal
FRATE As Double, ByVal Absol As Boolean)
  Dim stepInt As Double
  Dim StepNum As Long
  Dim StepDir As Boolean
  Dim stepnumX As Double
  Dim StepDirX As Boolean
  Dim stepnumY As Double
  Dim StepDirY As Boolean
  Dim stepnumZ As Long
  Dim StepDirZ As Boolean
  stepnumX = IIf(Absol, Abs(CoordX - LastX), Abs(CoordX))
  stepnumY = IIf(Absol, Abs(CoordY - LastY), Abs(CoordY))
  stepnumZ = IIf(Absol, Abs(CoordZ - LastZ), Abs(CoordZ))
  If stepnumX >= 1 Or stepnumY >= 1 Or stepnumZ >= 1 Then
     StepDirX = IIf((CoordX - LastX) > 0, True, False)
     StepDirY = IIf((CoordY - LastY) > 0, True, False)
     StepDirZ = IIf((CoordZ - LastZ) > 0, True, False)
        If stepnumX > stepnumY Then
          stepInt = A / 10
        ElseIf stepnumX > stepnumZ Then
          stepInt = A / 10
        ElseIf stepnumY > stepnumZ Then
          stepInt = E / 10
        Else
          stepInt = i / 10
        End If

     PlotXY StepDirX, stepnumX, StepDirY, stepnumY, stepInt, vbRed
     PlotXZ StepDirX, stepnumX, StepDirZ, stepnumZ, stepInt, vbRed
     PlotYZ StepDirY, stepnumY, StepDirZ, stepnumZ, stepInt, vbRed
   End If
End Sub
```

Keterangan:
Prosedur *DrawG1* berfungsi untuk menggambarkan perintah gerakan G0. Prosedur ini dipanggil oleh *commandrun* dengan *input* berupa nilai koordinat X, Y, dan Z yang bertipe *double* dan koordinat dalam status *absolute*. Pada bagian awal prosedur, dilakukan deklarasi variabel *StepInt* (yang menyatakan *interval* pengiriman pulsa: kecepatan) dan *StepNum* (yang menyatakan jumlah *step* untuk melakukan gerakan: panjang langkah). *StepNumX, StepNumY*, dan *StepNumZ* bertipe *Long*. Sedangkan untuk variabel *StepDir* (yang menyatakan arah gerakan: positif atau negatif). *StepDirX, StepDirY,* dan *StepDirZ* bertipe *Boolean.*
Proses selanjutnya adalah pengisian variabel *StepNumX, StepNumY* dan *StepNumZ* dengan memeriksa status koordinat gerakan, bila variabel Absol bernilai *true* (yang berarti menggunakan sistem koordinat *absolute*), maka kedua variabel tersebut diisi dengan nilai *absolute* dari selisih nilai *CoordX, CoordY*, dan *CoordZ* dengan *LastX, LastY,* dan *LastZ*. Bila ternyata absol bernilai *false* (berarti menggunakan sistem koordinat relatif), maka kedua variabel tersebut diisi dengan nilai *absolute* dari *CoordX, CoordY,* dan *CoordZ*. Jika nilai dari *StepNumX* atau *StepNumY* atau *StepNumZ* lebih besar atau sama dengan 1 dilakukan pengecekan arah gerakan pada sumbu X, Y, dan Z dengan melihat nilai *(CoordX – LastX), (CoordY – LastY),* dan *(CoordX – LastX)*
Bila memiliki nilai lebih besar dari 0, maka *StepDirX, StepDirY,* dan *StepDirZ* bernilai *true*, dan sebaliknya bernilai *false*. Jika *StepNumX* lebih besar dari *StepNumY*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi X / (Kecepatan X / 60). Jika *StepNumX* lebih besar dari *StepNumZ*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi X / (Kecepatan X / 60). Jika *StepNumY* lebih besar dari *StepNumZ*, maka *StepInt* diisi dengan nilai dari persamaan *StepInt* = Resolusi Y / (Kecepatan Y / 60). Kemudian dipanggil prosedur *Plot XY* dengan *input StepDirX, StepNumX, StepDirY, StepNumY, StepInt,* dan *vbRed, Plot XZ* dengan *input StepDirX, StepNumX, StepDirZ, StepNumZ, StepInt,* dan *vbRed, Plot YZ* dengan *input StepDirY, StepNumY, StepDirZ, StepNumZ, StepInt,* dan *vbRed.*

```
Private Sub DrawG2(ByVal CoordX, ByVal CoordY, ByVal CoordI, ByVal CoordJ, ByVal
ArcStep, ByVal FRATE As Double, ByVal Absol As Boolean)
  Dim Rad, DX1, DY1, DX2, DY2, NextX, NextY As Double
  Dim DegStart, DegEnd, DegStep, DegCount As Double
  Dim S, IncX, IncY, SignX, SignY As Integer
  Dim stepInt, stepnumX, stepnumY, stepnumZ As Long
  Dim StepDirX, StepDirY, StepDirZ As Boolean
  Dim Tempstr As String
  Dim Valid As Boolean
  If Absol Then
     CoordI = LastX + CoordI
     CoordJ = LastY + CoordJ
     DX1 = LastX - CoordI
     DY1 = LastY - CoordJ
     DX2 = CoordX - CoordI
     DY2 = CoordY - CoordJ
  Else
     DX1 = 0 - CoordI
     DY1 = 0 - CoordJ
     DX2 = CoordX - CoordI
     DY2 = CoordY - CoordJ
  End If
  Valid = True
  Rad = Sqr(DX1 ^ 2 + DY1 ^ 2)
  If Abs(Rad = Sqr(DX2 ^ 2 + DY2 ^ 2)) > 2 Then
     MsgBox "Data not valid.....!"
     Valid = False
  End If
```

```
If DY1 = 0 Then
   If DX1 = 0 Then          ' no line
      MsgBox "Zero radius.....!"
      Valid = False
   ElseIf DX1 > 0 Then      ' 0 degree
      DegStart = 0
   ElseIf DX1 < 0 Then      ' 180 degree
      DegStart = 180
   End If
ElseIf DY1 > 0 Then
   If DX1 = 0 Then          ' 90 degree
      DegStart = 90
   ElseIf DX1 > 0 Then      ' Quadrant 1
      DegStart = Abs(DegATan(DY1 / DX1))
   ElseIf DX1 < 0 Then      ' Quadrant 2
      DegStart = 180 - Abs(DegATan(DY1 / DX1))
   End If
ElseIf DY1 < 0 Then
   If DX1 = 0 Then          ' 270 degree
      DegStart = 270
   ElseIf DX1 > 0 Then      ' Quadrant 4
      DegStart = 360 - Abs(DegATan(DY1 / DX1))
   ElseIf DX1 < 0 Then      ' Quadrant 3
      DegStart = 180 + Abs(DegATan(DY1 / DX1))
   End If
End If
If DY2 = 0 Then
   If DX2 = 0 Then          ' no line
      MsgBox "Zero radius.....!"
      Valid = False
   ElseIf DX2 > 0 Then      ' 0 degree
      DegEnd = 0
   ElseIf DX2 < 0 Then      ' 180 degree
      DegEnd = 180
   End If
ElseIf DY2 > 0 Then
   If DX2 = 0 Then          ' 90 degree
      DegEnd = 90
   ElseIf DX2 > 0 Then      ' Quadrant 1
      DegEnd = Abs(DegATan(DY2 / DX2))
   ElseIf DX2 < 0 Then      ' Quadrant 2
      DegEnd = 180 - Abs(DegATan(DY2 / DX2))
   End If
ElseIf DY2 < 0 Then
   If DX2 = 0 Then          ' 270 degree
      DegEnd = 270
   ElseIf DX2 > 0 Then      ' Quadrant 4
      DegEnd = 360 - Abs(DegATan(DY2 / DX2))
   ElseIf DX2 < 0 Then      ' Quadrant 3
      DegEnd = 180 + Abs(DegATan(DY2 / DX2))
   End If
End If
If Valid Then
   DegStep = DegATan(ArcStep / Rad)
   If DegStart <= DegEnd Then DegStart = DegStart + 360
```

```
DegCount = DegStart
Do While DegCount >= DegEnd
  Select Case DegCount Mod 360
  Case Is >= 270        'Q4
    SignX = -1
    SignY = -1
  Case Is >= 180        'Q3
    SignX = -1
    SignY = 1
  Case Is >= 90         'Q2
    SignX = 1
    SignY = 1
  Case Is >= 0          'Q1
    SignX = 1
    SignY = -1
  End Select
  NextX = CoordI + Degcos(DegCount) * Rad
  NextY = CoordJ + Degsin(DegCount) * Rad
  IncX = Round(Abs(NextX - LastX))
  IncY = Round(Abs(NextY - LastY))
  stepnumX = Abs(IncX)
  stepnumY = Abs(IncY)
  If stepnumX >= 1 Or stepnumY >= 1 Then
    StepDirX = IIf(SignX > 0, True, False)
    StepDirY = IIf(SignY > 0, True, False)
    If stepnumX > stepnumY Then
      'StepInt = ParamX(0) / FRate * 1000
      stepInt = A / 10
    Else
      'StepInt = ParamY(0) / FRate * 1000
      stepInt = E / 10
    End If
    PlotXY StepDirX, stepnumX, StepDirY, stepnumY, stepInt, vbRed
    PlotXZ StepDirX, stepnumX, StepDirZ, stepnumZ, stepInt, vbRed
    PlotYZ StepDirY, stepnumY, StepDirZ, stepnumZ, stepInt, vbRed

  End If
  DegCount = IIf(DegCount > DegEnd And Abs((DegCount - DegStep)) Mod 90 < DegStep,
DegCount + (DegCount - DegStep) Mod 90, DegCount)
  DegCount = IIf(DegCount - DegEnd > DegStep Or DegCount = DegEnd, DegCount -
DegStep, DegEnd)
  Loop
 End If
End Sub
```

Keterangan:
Prosedur *DrawG2* berfungsi untuk menggambarkan perintah gerakan G2. Prosedur ini dipanggil oleh prosedur *commandrun* dengan *input* berupa nilai koordinat X, Y, Z, I, J, *ArcStep*, dan *Frate* yang bertipe *Double* dan status koordinat dengan nilai *absolute*. Pada bagian awal prosedur, dilakukan deklarasi bahwa variabel *Rad*, DX1, DY1, DX2, DY2, *nextX, nextY, DegStart, DegEnd,DegStep, DegCount* bertipe *Double*. Variabel *S, IncX, IncY, SignX, SignY* bertipe *Integer*. Variabel *StepInt, StepNumX, StepNumY, dan StepNumZ* bertipe *Long*. Variabel *StepDirX, StepDirY,* ,dan *StepDirZ* bertipe *Boolean*. Variabel *TempStr* bertipe *string*.
Proses selanjutnya adalah pengecekan sistem koordinat, dan melakukan inisialisasi terhadap *CoordI, CoordJ,* DX1, DY1, DX2, dan DY2 berdasarkan hasil pengecekan tersebut. Kemudian

dilanjutkan dengan pengesetan variabel *valid* dengan nilai *true* dan melakukan perhitungan terhadap variabel *Rad* yang menyatakan panjang sisi miring antara DX1 dan DY1. Jika nilai *absolute* dari selisih Rad dengan panjang sisi miring antara DX2 dan DY2 lebih besar dari 2, maka dikeluarkan *MsgBox* yang menyatakan bahwa *"Data Not Valid...."* dan variabel *valid* diset *false*.
Jika DY1 = 0 dan DX1 = 0, dikeluarkan *MsgBox "Zero Radius...!"*, sehingga tidak ada garis yang akan digambarkan variabel diset *false*. Jika DY1 = 0 dan DX1 lebih besar dari 0, hal itu menyatakan sudut 0 derajat oleh karena itu variabel *DegStart* diset 0. Jika DY1 = 0 dan DX1 lebih kecil dari 0, hal itu menyatakan sudut 180 derajat, seingga variabel *DegStart* diset 180.
Dalam kondisi DY1 lebih besar dari 0 dan DX1 = 0, hal itu menyatakan sudut 90 derajat oleh karena itu variabel *DegStart* diset 90, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 1), maka variabel *DegStart* diset dengan mengambil nilai *absolute* dari hasil arcTan (DY1 / DX1). Bila dalam kondisi ini DX1 lebih kecil dari 0 (*Quadrant* 2), maka *DegStart* mengambil nilai hasil perhitungan 180 – nilai *absolute* dari arcTan (DY1 / DX1).
Dalam kondisi DY1 lebih besar dari 0 dan DX1 = 0, hal itu menyatakan sudut 270 derajat oleh karena itu variabel *DegStart* diset 270, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 4), maka variabel *DegStart* diset dengan mengambil nilai hasil perhitungan 360 – *absolute* dari hasil arcTan (DY1 / DX1). Bila dalam kondisi ini DX1 lebih kecil dari 0 (*Quadrant* 3), maka *DegStart* mengambil nilai hasil perhitungan 180 + nilai *absolute* dari arcTan (DY1 / DX1).
Jika DY2 = 0 dan DX2 = 0, dikeluarkan *MsgBox "Zero Radius...!"*, sehingga tidak ada garis yang akan digambarkan variabel diset *false*. Jika DY2 = 0 dan DX2 lebih besar dari 0, hal itu menyatakan sudut 0 derajat oleh karena itu variabel *DegStart* diset 0. Jika DY2 = 0 dan DX2 lebih kecil dari 0, hal itu menyatakan sudut 180 derajat, sehingga variable *DegStart* diset 180.
Dalam kondisi DY2 lebih besar dari 0 dan DX2 = 0, hal itu menyatakan sudut 90 derajat oleh karena itu variabel *DegEnd* diset 90, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 1), maka variabel *DegEnd* diset dengan mengambil nilai *absolute* dari hasil arcTan (DY2 / DX2). Bila dalam kondisi ini DX2 lebih kecil dari 0 (*Quadrant* 2), maka *DegStart* mengambil nilai hasil perhitungan 180 – nilai *absolute* dari arcTan (DY2 / DX2).
Dalam kondisi DY2 lebih besar dari 0 dan DX2 = 0, hal itu menyatakan sudut 270 derajat oleh karena itu variabel *DegEnd* diset 270, tetapi jika dalam kondisi ini DX2 lebih besar dari 0 (*Quadrant* 4), maka variabel *DegEnd* diset dengan mengambil nilai hasil perhitungan 360 – nilai *absolute* dari hasil arcTan (DY2 / DX2). Bila dalam kondisi ini DX2 lebih kecil dari 0 (*Quadrant* 3), maka *DegStart* mengambil nilai hasil perhitungan 180 + nilai *absolute* dari arcTan (DY2 / DX2).
Setelah menentukan posisi sudut awal dan sudut akhir, jika variabel *Valid* bernilai *true*, maka diperhitungkan variabel *DegStep* yang nilainya adalah hasil arcTan *(ArcStep / rad)*. Bila *DegStart* lebih kecil atau sama dengan *DegEnd*, maka *DegStart* diisi dengan nilai *DegStart* + 360 (membuka gerakan satu lingkaran dengan awal *DegStart*). Kemudian *DegCount* diset sama dengan *DegStart*.
Kemudian dilakukan proses berulang *(loop)* selama nilai *DegCount* = *DegEnd* dengan tahapan sebagai berikut:

- Dilakukan penandaan *Quadrant* dengan memberikan nilai pada variabel *signX* dan *signY* sesuai dengan *quadrant*nya.
- Perhitungan variabel *NextX* yang nilainya diperoleh dari hasil perhitungan nilai koordinat I ditambah hasil kali Cos *DegCount* dengan panjang *Rad*.
- Perhitungan dengan variabel *NextY* yang nilainya diperoleh dari hasil perhitungan nilai koordinat J ditambah dengan hasil kali Sin *DegCount* dengan panjang *Rad*.
- Perhitungan nilai *IncX* yang diambil dari pembulatan nilai *absolute* selisih *NextX* dan *LastX*. Begitu pula halnya dengan *IncY* diambil dari pembulatan nilai *absolute* selisih *NextY* dan *LastY*.
- Variabel *StepNumX* diambil dari nilai *absolute* dari *IncX* dan *StepNumY* dan *absolute IncY*.
- Jika *StepNumX* dan *StepNumY* lebih besar atau sama dengan 1 maka ditentukan arah gerakan *(StepDirX* dan *StepDirY)* dengan melihat kondisi *SignX* dan *SignY*. Bila nilainya lebih besar dari 1, maka arah gerakan menuju koordinat positif *(true)* dan

sebaliknya menuju koordinat negatif. Jika *StepNumX* lebih besar dari *StepNumY* maka nilai *StepInt* diperoleh dari persamaan Resolusi X / 10, tetapi bila *StepNumX* lebih kecil dari *StepNumY*, maka nilai *StepInt* diperoleh dari Resolusi Y / 10. Kemudian dilakukan penggambaran lintasan alat potong dengan memanggil prosedur PlotXY, PlotXZ, dan PlotYZ, dengan *input* arah gerakan *(StepDirX* dan *StepDirY),* jumlah *step (StepNumX* dan *StepNumY),* interval pengiriman *step (StepInt),* dan warna merah *(vbRed)* sebagai garis lintasan alat potong.

- Dilakukan *setting* terhadap nilai variabel *DegCount* dengan kondisi jika *DegCount* > *DegEnd* dan Abs (*DegCount – DegStep*) Mod 90 < *DegStep* diambil nilai *DegCount* = *DegCount* + (*DegCount – DegStep*) Mod 90. Bila kondisi tersebut tidak terpenuhi, maka *DegCount* = *DegCount*.
- Dilakukan *setting* kembali terhadap nilai variabel *DegCount* dengan kondisi jika *DegCount – DegEnd* > *DegStep* atau *DegCount* = *DegEnd*, diambil nilai *DegCount* = *DegCount – DegStep*. Bila kondisi tersebut tidak terpenuhi, maka *DegCount* = *DegEnd*.

```
Private Sub DrawG3(ByVal CoordX, ByVal CoordY, ByVal CoordI, ByVal CoordJ, ByVal
ArcStep, ByVal FRATE As Double, ByVal Absol As Boolean)
  Dim Rad, DX1, DY1, DX2, DY2, NextX, NextY As Double
  Dim DegStart, DegEnd, DegStep, DegCount As Double
  Dim S, IncX, IncY, SignX, SignY As Integer
  Dim stepInt, stepnumX, stepntumY, stepnumZ As Long
  Dim StepDirX, StepDirY, StepDirZ As Boolean
  Dim Tempstr As String
  Dim Valid As Boolean
  If Absol Then
    CoordI = LastX + CoordI
    CoordJ = LastY + CoordJ
    DX1 = LastX - CoordI
    DY1 = LastY - CoordJ
    DX2 = CoordX - CoordI
    DY2 = CoordY - CoordJ
  Else
    DX1 = 0 - CoordI
    DY1 = 0 - CoordJ
    DX2 = CoordX - CoordI
    DY2 = CoordY - CoordJ
  End If
  Valid = True
  Rad = Sqr(DX1 ^ 2 + DY1 ^ 2)
  If Abs(Rad - Sqr(DX2 ^ 2 + DY2 ^ 2)) > 2 Then
    MsgBox "Data not valid.....!"
    Valid = False
  End If
  If DY1 = 0 Then
    If DX1 = 0 Then          ' no line
      MsgBox "Zero radius.....!"
      Valid = False
    ElseIf DX1 > 0 Then      ' 0 degree
      DegStart = 0
    ElseIf DX1 < 0 Then      ' 180 degree
      DegStart = 180
    End If
  ElseIf DY1 > 0 Then
    If DX1 = 0 Then          ' 90 degree
      DegStart = 90
```

```
    ElseIf DX1 > 0 Then        ' Quadrant 1
      DegStart = Abs(DegATan(DY1 / DX1))
    ElseIf DX1 < 0 Then        ' Quadrant 2
      DegStart = 180 - Abs(DegATan(DY1 / DX1))
    End If
  ElseIf DY1 < 0 Then
    If DX1 = 0 Then            ' 270 degree
      DegStart = 270
    ElseIf DX1 > 0 Then        ' Quadrant 4
      DegStart = 360 - Abs(DegATan(DY1 / DX1))
    ElseIf DX1 < 0 Then        ' Quadrant 3
      DegStart = 180 + Abs(DegATan(DY1 / DX1))
    End If
  End If
  If DY2 = 0 Then
    If DX2 = 0 Then            ' no line
      MsgBox "Zero radius.....!"
      Valid = False
    ElseIf DX2 > 0 Then        ' 0 degree
      DegEnd = 0
    ElseIf DX2 < 0 Then        ' 180 degree
      DegEnd = 180
    End If
  ElseIf DY2 > 0 Then
    If DX2 = 0 Then            ' 90 degree
      DegEnd = 90
    ElseIf DX2 > 0 Then        ' Quadrant 1
      DegEnd = Abs(DegATan(DY2 / DX2))
    ElseIf DX2 < 0 Then        ' Quadrant 2
      DegEnd = 180 - Abs(DegATan(DY2 / DX2))
    End If
  ElseIf DY2 < 0 Then
    If DX2 = 0 Then            ' 270 degree
      DegEnd = 270
    ElseIf DX2 > 0 Then        ' Quadrant 4
      DegEnd = 360 - Abs(DegATan(DY2 / DX2))
    ElseIf DX2 < 0 Then        ' Quadrant 3
      DegEnd = 180 + Abs(DegATan(DY2 / DX2))
    End If
  End If
  If Valid Then
    DegStep = DegATan(ArcStep / Rad)
    If DegEnd <= DegStart Then DegEnd = DegEnd + 360
    DegCount = DegStart
    Do While DegCount <= DegEnd
      Select Case DegCount Mod 360
      Case Is >= 270           ' Q4
        SignX = 1
        SignY = 1
      Case Is >= 180           ' Q3
        SignX = 1
        SignY = -1
      Case Is >= 90            ' Q2
        SignX = -1
        SignY = -1
```

```
        Case Is >= 0          'Qi
          SignX = -1
          SignY = 1
        End Select
        NextX = CoordI + Degcos(DegCount) * Rad
        NextY = CoordJ + Degsin(DegCount) * Rad
        IncX = Round(Abs(NextX - LastX))
        IncY = Round(Abs(NextY - LastY))
        stepnumX = Abs(IncX)
        stepnumY = Abs(IncY)
        If stepnumX >= 1 Or stepnumY >= 1 Then
          StepDirX = IIf(SignX > 0, True, False)
          StepDirY = IIf(SignY > 0, True, False)
          If stepnumX > stepnumY Then
            'StepInt = ParamX(0) / FRate * 1000
            stepInt = A / 10
          Else
            'StepInt = ParamY(0) / FRate * 1000
            stepInt = E / 10
          End If
          PlotXY StepDirX, stepnumX, StepDirY, stepnumY, stepInt, vbRed
          PlotXZ StepDirX, stepnumX, StepDirZ, stepnumZ, stepInt, vbRed
          PlotYZ StepDirY, stepnumY, StepDirZ, stepnumZ, stepInt, vbRed
        End If
        DegCount = IIf(DegCount < DegEnd And Abs((DegCount + DegStep)) Mod 90 <
DegStep, DegCount - (DegCount + DegStep) Mod 90, DegCount)
        DegCount = IIf(DegEnd - DegCount > DegStep Or DegCount = DegEnd, DegCount +
DegStep, DegEnd)
      Loop
    End If
End Sub
```

Keterangan:
Prosedur *DrawG3* berfungsi untuk menggambarkan perintah gerakan G3. Prosedur ini dipanggil oleh prosedur *commandrun* dengan *input* berupa nilai koordinat X, Y, Z, I, J, *ArcStep*, dan *Frate* yang bertipe *Double* dan status koordinat dengan nilai *absolute*. Pada bagian awal prosedur, dilakukan deklarasi bahwa variabel *Rad*, DX1, DY1, DX2, DY2, *nextX, nextY, DegStart, DegEnd,DegStep, DegCount* bertipe Double. Variabel S, *IncX, IncY, SignX, SignY* bertipe *Integer*. Variabel *StepInt, StepNumX, StepNumY,* dan *StepNumZ* bertipe *Long*. Variabel *StepDirX, StepDirY,* dan *StepDirZ* bertipe *Boolean*. Variabel *TempStr* bertipe *string*.
Proses selanjutnya adalah pengecekan sistem koordinat, dan melakukan inisialisasi trhadap *CoordI, CoordJ*, DX1, DY1, DX2, dan DY2 berdasarkan hasil pengecekan tersebut. Kemudian dilanjutkan dengan pengesetan variabel *valid* dengan nilai *true* dan melakukan perhitungan terhadap variabel *Rad* yang menyatakan panjang sisi miring antara DX1 dan DY1. Jika nilai *absolute* dari selisih *Rad* dengan panjang sisi miring antara DX2 dan DY2 lebih besar dari 2, maka dikeluarkan *MsgBox* yang menyatakan bahwa *"Data Not Valid...."* dan variabel *valid* diset *false*.
Jika DY1 = 0 dan DX1 = 0, dikeluarkan *MsgBox "Zero Radius...!"*, sehingga tidak ada garis yang akan digambarkan variabel diset *false*. Jika DY1 = 0 dan DX1 lebih besar dari 0, hal itu menyatakan sudut 0 derajat oleh karena itu variabel *DegStart* diset 0. Jika DY1 = 0 dan DX1 lebih kecil dari 0, hal itu menyatakan sudut 180 derajat, seingga variabel *DegStart* diset 180.
Dalam kondisi DY1 lebih besar dari 0 dan DX1 = 0, hal itu menyatakan sudut 90 derajat oleh karena itu variable *DegStart* diset 90, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 1), maka variabel *DegStart* diset dengan mengambil nilai *absolute* dari hasil arcTan (DY1 / DX1). Bila dalam kondisi ini DX1 lebih kecil dari 0 (*Quadrant* 2), maka *DegStart* mengambil nilai hasil perhitungan 180 – nilai *absolute* dari arcTan (DY1 / DX1).

Dalam kondisi DY1 lebih besar dari 0 dan DX1 = 0, hal itu menyatakan sudut 270 derajat oleh karena itu variabel *DegStart* diset 270, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 4), maka variabel *DegStart* diset dengan mengambil nilai hasil perhitungan 360 – *absolute* dari hasil arcTan (DY1 / DX1). Bila dalam kondisi ini DX1 lebih kecil dari 0 (*Quadrant* 3), maka *DegStart* mengambil nilai hasil perhitungan 180 + nilai *absolute* dari arcTan (DY1 / DX1).
Jika DY2 = 0 dan DX2 = 0, dikeluarkan *MsgBox "Zero Radius...!"*, sehingga tidak ada garis yang akan digambarkan variabel diset *false*. Jika DY2 = 0 dan DX2 lebih besar dari 0, hal itu menyatakan sudut 0 derajat oleh karena itu variabel *DegStart* diset 0. Jika DY2 = 0 dan DX2 lebih kecil dari 0, hal itu menyatakan sudut 180 derajat, sehingga variabel *DegStart* diset 180.
Dalam kondisi DY2 lebih besar dari 0 dan DX2 = 0, hal itu menyatakan sudut 90 derajat oleh karena itu variabel *DegEnd* diset 90, tetapi jika dalam kondisi ini DX1 lebih besar dari 0 (*Quadrant* 1), maka variabel *DegEnd* diset dengan mengambil nilai *absolute* dari hasil arcTan (DY2 / DX2). Bila dalam kondisi ini DX2 lebih kecil dari 0 (*Quadrant* 2), maka *DegStart* mengambil nilai hasil perhitungan 180 – nilai *absolute* dari arcTan (DY2 / DX2).
Dalam kondisi DY2 lebih besar dari 0 dan DX2 = 0, hal itu menyatakan sudut 270 derajat oleh karena itu variabel *DegEnd* diset 270, tetapi jika dalam kondisi ini DX2 lebih besar dari 0 (*Quadrant* 4), maka variabel *DegEnd* diset dengan mengambil nilai hasil perhitungan 360 – nilai *absolute* dari hasil arcTan (DY2 / DX2). Bila dalam kondisi ini DX2 lebih kecil dari 0 (*Quadrant* 3), maka *DegStart* mengambil nilai hasil perhitungan 180 + nilai *absolute* dari arcTan (DY2 / DX2).
Setelah menentukan posisi sudut awal dan sudut akhir, jika variabel *Valid* bernilai *true*, maka diperhitungkan variabel *DegStep* yang nilainya adalah hasil arcTan (*ArcStep* / rad). Bila *DegStart* lebih kecil atau sama dengan *DegEnd*, maka *DegStart* diisi dengan nilai *DegStart* + 360 (membuka gerakan satu lingkaran dengan awal *DegStart*). Kemudian *DegCount* diset sama dengan *DegStart*.
Kemudian dilakukan proses berulang *(loop)* selama nilai *DegCount* = *DegEnd* dengan tahapan sebagai berikut:

- Dilakukan penandaan *Quadrant* dengan memberikan nilai pada variabel *signX* dan *signY* sesuai dengan *quadrant*nya.
- Perhitungan variabel *NextX* yang nilainya diperoleh dari hasil perhitungan nilai koordinat I ditambah hasil kali Cos *DegCount* dengan panjang *Rad*.
- Perhitungan dengan variabel *NextY* yang nilainya diperoleh dari hasil perhitungan nilai koordinat J ditambah dengan hasil kali Sin *DegCount* dengan panjang *Rad*.
- Perhitungan nilai *IncX* yang diambil dari pembulatan nilai *absolute* selisih *NextX* dan *LastX*. Begitu pula halnya dengan *IncY* diambil dari pembulatan nilai *absolute* selisih *NextY* dan *LastY*.
- Variabel *StepNumX* diambil dari nilai *absolute* dari *IncX* dan *StepNumY* dari *absolute IncY*.
- Jika *StepNumX* dan *StepNumY* lebih besar atau sama dengan 1 maka ditentukan arah gerakan *(StepDirX dan StepDirY)* dengan melihat kondisi *SignX* dan *SignY*. Bila nilainya lebih besar dari 1, maka arah gerakan menuju koordinat positif *(true)* dan sebaliknya menuju koordinat negatif. Jika *StepNumX* lebih besar dari *StepNumY* maka nilai *StepInt* diperoleh dari persamaan Resolusi X / 10, tetapi bila *StepNumX* lebih kecil dari *StepNumY*, maka nilai *StepInt* diperoleh dari Resolusi Y / 10. Kemudian dilakukan penggambaran lintasan alat potong dengan memanggil prosedur PlotXY, PlotXZ, dan PlotYZ, dengan *input* arah gerakan *(StepDirX* dan *StepDirY)*, jumlah *step (StepNumX* dan *StepNumY)*, interval pengiriman *step (StepInt)*, dan warna merah *(vbRed)* sebagai garis lintasan alat potong.
- Dilakukan *setting* terhadap nilai variabel *DegCount* dengan kondisi jika *DegCount* > *DegEnd* dan Abs *(DegCount – DegStep)* Mod 90 < *DegStep* diambil nilai *DegCount* = *DegCount* + *(DegCount – DegStep)* Mod 90. Bila kondisi tersebut tidak terpenuhi, maka *DegCount* = *DegCount*.

- Dilakukan *setting* kembali terhadap nilai variabel *DegCount* dengan kondisi jika *DegCount – DegEnd > DegStep* atau *DegCount = DegEnd*, diambil nilai *DegCount = DegCount – DegStep*. Bila kondisi tersebut tidak terpenuhi, maka *DegCount = DegEnd*.

```
Public Sub DrawG81(ByVal CoordX As Double, ByVal CoordY As Double, ByVal CoordZ, ByVal
DIAMETER As Double, ByVal Absol As Boolean)
Dim Radius, x1, y1 As Long
DrawG0 CoordX, CoordY, CoordZ, Absol
Radius = (DIAMETER / 2) / scaleSet
x1 = CoordX / scaleSet
y1 = CoordY / scaleSet
Delay (A * 10)
Picture1.Circle (x1, y1), Radius, vbRed
Picture2.Line (CoordX, 0)-(CoordX, CoordZ), vbRed
Picture3.Line (0, CoordY)-(CoordZ, CoordY), vbRed
End Sub
```

Keterangan:
Prosedur *DrawG81* berfungsi untuk menggambarkan perintah gerakan G81 *(drilling)*. Prosedur ini dipanggil oleh prosedur *commandrun* dengan masukan berupa nilai *CoordX, CoordY, oordZ,* Diameter yang bertipe *Double* dan status koordinat dengan nilai *absolute*.
Pada bagian awal dideklarasikan radius, x1, dan y1 bertipe Long. Dilakukan penggambaran gerakan alat potong menuju *CoordX* dan *CoordY* dengan menggunakan prosedur G0. Nilai *radius* diperoleh dari persamaan (Diameter / 2) / *ScaleSet*. Nilai x1 diperoleh dari persamaan *CoordX / ScaleSet* dan nilai y1 dari persamaan *CoordY / ScaleSet*, selanjutnya x1 dan y1 adalah koordinat titik pusat lingkaran. Sebelum penggambaran dilakukan perintah *delay* dengan nilai (Resolusi X * 10) untuk memberikan waktu tunda. Kemudian dilakukan penggambaran lingkaran dengan titik pusat (x1, y1) dengan *Radius* sebagai nilai jari-jari lingkaran tersebut.

```
Public Sub DrawG84(ByVal CoordX As Double, ByVal CoordY As Double, ByVal CoordZ, ByVal
DIAMETER As Double, Absol As Boolean)
Dim Radius1, Radius2, x1, y1 As Long
DrawG0 CoordX, CoordY, CoordZ, Absol
Radius1 = (DIAMETER / 2) / scaleSet
Radius2 = ((DIAMETER * 1.2) / 2) / scaleSet
x1 = CoordX / scaleSet
y1 = CoordY / scaleSet
Delay (A * 10)
Picture1.Circle (x1, y1), Radius1, vbRed
Picture1.Circle (x1, y1), Radius2, vbRed
Picture2.Line (CoordX, 0)-(CoordX, CoordZ), vbRed
Picture3.Line (0, CoordY)-(CoordZ, CoordY), vbRed
End Sub
```

Keterangan:
Prosedur *DrawG84* hampir sama dengan prosedur *DrawG81* hanya saja pada prosedur ini dilakukan dua kali penggambaran lingkaran. Lingkaran pertama dengan nilai jari-jari sebesar *Radius* merupakan diameter *minor* ulir dalam. Lingkaran kedua dengan nilai jari-jari *Radius2* adalah diameter *mayor*.

```
Public Sub DrawG85(ByVal CoordX As Double, ByVal CoordY As Double, ByVal CoordZ, ByVal
DIAMETER As Double, ByVal Absol As Boolean)
Dim Radius, x1, y1 As Long
DrawG0 CoordX, CoordY, CoordZ, Absol
Radius = (DIAMETER / 2) / scaleSet
```

```
x1 = CoordX / scaleSet
y1 = CoordY / scaleSet
Delay (A * 10)
Picture1.Circle (x1, y1), Radius, vbRed
Picture2.Line (CoordX, 0)-(CoordX, CoordZ), vbRed
Picture3.Line (0, CoordY)-(CoordZ, CoordY), vbRed
End Sub
```

Keterangan:
Prosedur *DrawG85* sama dengan prosedur *DrawG81*.

```
Public Sub DrawG88(ByVal CoordX, ByVal CoordY, ByVal CoordZ, ByVal CoordI, ByVal
CoordJ, ByVal CoordK, ByVal Absol As Boolean)
Dim A, B, C, D As Double
Dim x1, x2, y1, y2 As Double
Dim stepX, stepY As Double
Dim LastX1, LastY1, LastX2, LastY2 As Double
 stepX = (CoordI / CoordK)
 stepY = (CoordJ / CoordK)
 x1 = CoordX + stepX
 y1 = CoordY + stepY
 x2 = CoordX - stepX
 y2 = CoordY - stepY
 DrawG0 CoordX, CoordY, 0, Absol 'MENUJU TITIK PUSAT

  Do While (x1 <= CoordX + CoordI) And (y1 <= CoordY + CoordJ)
    DrawG1 x1, CoordY, CoordZ, 1, Absol    'MENUJU TITIK 1
    LastX1 = x1
    DrawG1 x1, y1, CoordZ, 1, Absol        'MENUJU TITIK 2
    LastY1 = y1
    DrawG1 x2, y1, CoordZ, 1, Absol        'MENUJU TITIK 3
    LastX2 = x2
    DrawG1 x2, y2, CoordZ, 1, Absol        'MENUJU TITIK 4
    LastY2 = y2
    DrawG1 x1, y2, CoordZ, 1, Absol        'MENUJU TITIK 5
    DrawG1 x1, CoordY, CoordZ, 1, Absol    'KEMBALI KETITIK 1
    x1 = LastX1 + stepX
    y1 = LastY1 + stepY
    x2 = LastX2 - stepX
    y2 = LastY2 - stepY
  Loop
A = CoordX + CoordI
B = CoordY + CoordJ
C = CoordX - CoordI
D = CoordY - CoordJ
DrawG1 A, CoordY, CoordZ, 1, Absol    'MENUJU TITIK 1
DrawG1 A, B, CoordZ, 1, Absol         'MENUJU TITIK 2
DrawG1 C, B, CoordZ, 1, Absol         'MENUJU TITIK 3
DrawG1 C, D, CoordZ, 1, Absol         'MENUJU TITIK 4
DrawG1 A, D, CoordZ, 1, Absol         'MENUJU TITIK 5
DrawG1 A, CoordY, CoordZ, 1, Absol    'KEMBALI KETITIK 1
DrawG1 CoordX, CoordY, CoordZ, 1, Absol    'KEMBALI KETITIK 1
End Sub
```

Keterangan:

Prosedur *Draw88* berfungsi untuk menggambarkan perintah gerakan G88. Gerakan ini akan menggambarkan gerakan *pocket rectangular*. Prosedur ini dipanggil oleh prosedur *commandrun* dengan masukan berupa nilai *CoordX, CoordY, CordZ, CoordI, Coordj,* dan *CoordK* yang bertipe *Long* dam status koordinat dengan nilai *absolute*. Pada bagian awal dideklarasikan x1, x2, y1, *stepX,* dan *stepY* bertipe *Integer*. Variabel *LastX1, LastY1, LastX2, LastY2*, A, B, C, D, dan y2 bertipe *Double*.
Proses selanjutnya adalah dilakukan pengisian variabel *stepX* dengan nilai hasil persamaan *CoordI / CoordK* dan variabel *stepY* dengan nilai hasil persamaan *Coord J / CoordK*. Variabel x1 diset dari penjumlahan *CoordX* dengan *stepX*. Sedangkan variabel x2 diset dari selisih *CoordX* dengan *stepX*. Begitu juga dengan y1 diisi dengan nilai dari jumlah *CoordY* dengan *stepX* dan y2 dengan diset dari selisih *CoordY* dengan *stepY*.
Kemudian dilakukan penggambaran gerakan menuju titik pusat dengan menggunakan prosedur G0 dengan masukan berupa *CoordX, CoordY, CoordZ*. Dilanjutkan dengan pengecekan, bila *CoordI* lebih besar atau sama dengan *CoordX + CoordI*. Namun bila sebaliknya, maka diset batas pengulangan dengan parameter Y1 lebih kecil atau sama dengan *CoordY + CoordJ*. Proses pengulangan dilakukan untuk mendapat bentuk kotak melalui kombinasi prosedur *DrawG1* bergerak ke enam titik koordinat dengan nilai masukan *StepX* dan *StepY*. Diakhir prosedur dilakukan sekali lagi gerakan pembentukan kotak namun dengan nilai masukan A, B, C, dan D. Nilai A dan B merupakan jumlah dari *CoordX* dengan *CoordI* dan *CoordY* dengan *CoordJ*. Sedangkan nilai C dan D merupakan selisih dari *CoordX* dengan *CoordI* dan *CoordY* dengan *CoordJ*.

```
Public Sub DrawG89(ByVal CoordX, ByVal CoordY, ByVal CoordZ, ByVal CoordL, ByVal
CoordJ, ByVal CoordK, ByVal Absol As Boolean)
Dim A As Double
Dim Step, Laststep1 As Double
Dim x1. v1 As Double
Dim B As Double
Dim LastX1 As Double
Step = CoordJ / CoordK
A = CoordJ / CoordK
x1 = CoordX + Step
B = CoordJ + CoordX
DrawG0 CoordX, CoordY, 0, Absol

Do While (x1 <= B)
   DrawG1 x1, CoordY, CoordZ, 1, Absol
   LastX1 = x1
   DrawG3 x1, CoordY, -Step, 0, 0.1, 1, Absol
   Laststep1 = Step
   x1 = LastX1 + A
   Step = Laststep1 + A
Loop
If x1 <> B Then
   DrawG1 B, CoordY, CoordZ, 1, Absol
   DrawG3 B, CoordY, -CoordJ, 0, 0.1, 1, Absol
   DrawG1 CoordX, CoordY, CoordZ, 1, Absol    'KEMBALI KETITIK 1
   End If
End Sub
```

Keterangan:
Prosedur *Draw89* berfungsi untuk menggambarkan perintah gerakan G89. Gerakan ini akan menggambarkan gerakan *pocket circular*. Prosedur ini dipanggil oleh prosedur *commandrun* dengan masukan berupa nilai *CoordX, CoordY, CordZ, CoordI, Coordj,* dan *CoordK* yang bertipe

Long dan status koordinat dengan nilai *absolute*. Pada bagian awal dideklarasikan x1, *LastX1*, y1, A, B, *Step* dan *Laststep1* bertipe *Double*.
Proses selanjutnya adalah dilakukan pengisian variabel *step* dengan nilai hasil persamaan *CoordI / CoordK*. Variabel x1 diset dari penjumlahan *CoordX* dengan *step*. Variabel B diset dari penjumlahan *CoordX* dengan *CoordJ*.
Kemudian dilakukan penggambaran gerakan menuju titik pusat dengan menggunakan prosedur G0 dengan masukan berupa *CoordX, CoordY, CoordZ*. Dilanjutkan dengan penggambaran menggunakan prosedur *DrawG1* dengan nilai masukan x1 dan *CoordY* dilanjutkan *DrawG3* dengan nilai masukan x1, *CoordY, -step,* 0, 0.1, 1, dan absol. Dimana prosedur *DrawG1* dan *DrawG3* dilakukan pengulangan dengan batas parameter X1 lebih kecil atau sama dengan B. Diakhir prosedur ini dilakukan sekali lagi gerakan dengan prosedur *DrawG1, DrawG3* dengan masukan B, *CoordY*, dan *–CoordJ*. Ditambah lagi dengan prosedur *DrawG1* dengan masukan *CoordX, CoordY, dan CoordZ* untuk menuju ketitik pusat awal.

```
Public Sub DrawG80(ByVal CoordX, ByVal CoordY, ByVal CoordZ As Double, ByVal FRATE As
Double, ByVal Absol As Boolean)
Dim stepInt As Long
   Dim StepNum As Long
   Dim StepDir As Boolean
   Dim stepnumX As Long
   Dim StepDirX As Boolean
   Dim stepnumY As Long
   Dim StepDirY As Boolean
   Dim stepnumZ As Long
   Dim StepDirZ As Boolean
stepnumX = IIf(Absol, Abs(CoordX - LastX), Abs(CoordX))
stepnumY = IIf(Absol, Abs(CoordY - LastY), Abs(CoordY))
stepnumZ = IIf(Absol, Abs(CoordZ - LastZ), Abs(CoordZ))
   If stepnumX >= 1 Or stepnumY >= 1 Or stepnumZ >= 1 Then
      StepDirX = IIf((CoordX - LastX) > 0, True, False)
      StepDirY = IIf((CoordY - LastY) > 0, True, False)
      StepDirZ = IIf((CoordZ - LastZ) > 0, True, False)
      PlotXY StepDirX, stepnumX, StepDirY, stepnumY, 0.0005, vbBlue '&H8000000F
      Picture2.Line (LastX, LastZ)-(0, LastZ), vbBlue
      Picture3.Line (LastZ, LastY)-(LastZ, 0), vbBlue
   End If
End Sub
```

Keterangan:
Prosedur *DrawG80* hampir mirip dengan prosedur *DrawG0* maupun *DrawG1*, hanya saja pada prosedur ini variabel *stepInt* langsung ditetapkan dengan nilai masukan sebesar dari perhitungan 0.005. Prosedur *DrawG80* berfungsi untuk menghapuskan perintah gerakan G81, G84, G85, G88, dan G89. Melalui perintah ini akan digambarkan gerakan alat potong dari semua titik akan kembali ke titik awal dengan nilai *CoordX, CoordY,* dan *CoordZ* sebesar 0.

```
Private Sub PlotXY(ByVal bDirX As Boolean, ByVal lNumStepX As Long, ByVal bDirY As
Boolean, ByVal lNumStepY As Long, ByVal lPeriode As Double, ByVal DColor As Long)
   Dim i, X, Y, x1, y1, x2, y2, DX, DY As Integer
   Dim XInc, YInc, ErrX, ErrY As Integer
   Dim SaveColor As Long

   Dim DL As Double
   SaveColor = Picture1.ForeColor
   Picture1.ForeColor = DColor
   x1 = LastX
```

```
    y1 = LastY
    DX = lNumStepX * IIf(bDirX, 1, -1)
    DY = lNumStepY * IIf(bDirY, 1, -1)
    x2 = x1 + DX
    y2 = y1 + DY
    DL = lPeriode
    XInc = 1
    YInc = 1
    If DX < 0 Then XInc = -1: DX = -DX
    If DY < 0 Then YInc = -1: DY = -DY
    If DX <> 0 Or DY <> 0 Then
      If DY <= DX Then
        ErrX = 0
        Do Until LastX = x2
          LastX = LastX + XInc
          ErrX = ErrX + 2 * DY
          If ErrX > DX Then LastY = LastY + YInc: ErrX = ErrX - 2 * DX
          Picture1.PSet ((LastX / scaleSet), (LastY / scaleSet)), Picture1.ForeColor
          Delay (DL)
        Loop
      Else
        ErrY = 0
        Do Until LastY = y2
          LastY = LastY + YInc
          ErrY = ErrY + 2 * DX
          If ErrY > DY Then LastX = LastX + XInc: ErrY = ErrY - 2 * DY
          Picture1.PSet ((LastX / scaleSet), (LastY / scaleSet)), Picture1.ForeColor
          Delay (DL)
        Loop
      End If
      If DX <> 0 And DY <> 0 Then
        ErrX = 0
        Do Until LastX = x2 And LastY = y2
          LastX = LastX + XInc
          LastY = LastY + YInc
          Picture1.PSet ((LastX / scaleSet), (LastY / scaleSet)), Picture1.ForeColor
        Loop
      End If
    End If
    Picture1.ForeColor = SaveColor
End Sub
```

Keterangan:
Prosedur *PlotXY* digunakan untuk menggambar gerakan alat potong pada arah sumbu XY. Prosedur ini menerima masukan berupa jumlah *step* gerakan pada arah sumbu X dan Y (*1NumStepX* dan *1NumStepY*), periode (1Periode), arah gerakan pada sumbu X dan Y (*bDirX* dan *bDirY*), dan warna garis (*Dcolor*). Pada bagian awal dideklarasikan bahwa variabel I, X, Y, X1, Y1, X2, Y2, DX, DY, DL, *Xinc, Yinc, ErrX, ErrY* bertipe *Integer*, sedangkan variabel *savecolor* bertipe *Long*.
Data warna asli dari *Picture1.ForeColor* sementara disimpan dahulu dalam variabel *SaveColor*, *Picture1.ForeColor* diganti dengan data dari *Dcolor*. Kemudian variabel X1 diisi dengan nilai *LastX* (nilai X terakhir) dan Y1 diisi dari *LastY* (nilai Y terakhir). Sedangkan variabel DX (jarak perpindahan yang harus ditempuh sejajar sumbu X), diisi dari hasil kali antar jumlah *step* dengan nilai arah pergerakan (*1NumStep* * lif(*bdirX*, 1, -1), begitu pula halnya dengan variabel DY.

Setelah proses diatas, barulah variabel X2 dan Y2 diisi dengan jumlah nilai awal (X1 dan Y1 dengan jarak perpindahan (DX dan DY). Variabel DL berisi besarnya *delay* dalam proses pengiriman *impuls* listrik ke motor *stepper*. Nilai dari variabel ini diambil dari nilai variabel 1Periode. Kemudian variabel *Xinc* dan *Yinc* diset agar bernilai 1.
Jika DX lebih kecil dari 0 maka diset bahwa *Xinc* bernilai -1 dan DX sama dengan –DX. Begitu pula bila Dy lebih kecil dari 0, maka *Yinc* bernilai -1 dan DY sama dengan –DY.
Jika DX tidak sama dengan 0 atau DY tidak sama dengan 0, maka jika DY lebih kecil atau sama dengan DX, maka diset *ErrX* = 0. Nilai *LastX* diisi dengan jumlah dari *LastX* dengan *Xinc*, dan nilai *ErrX* diisi dengan jumlah *ErrX* dengan 2 kali DY. Bila *ErrX* tersebut lebih besar dari DX, kemudian diset bahwa *LastY* merupakan jumlah dari *LastY* dengan *Yinc* dan nilai *ErrX* dikurangi 2 kali DX. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture1.Pset* dengan nilai *LastX* dibagi besarnya *ScaleSet* dan *LastY* dibagi besarnya *ScaleSet* dan *setting* warna garis sesuai dengan *setting Picture1.ForeColor.* Proses tersebut dilanjutkan dengan me*refresh picture1*. Proses ini diulang hingga nilai *LastX* sama dengan X2. Setelah selesai *Picture1. ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor.*
Sebaliknya apabila DY lebih besar dari DX, maka diset *ErrY* = 0, kemudian nilai *LastY* diisi dengan jumlah dari *LastY* dengan *Yinc*, dan nilai *ErrY* diisi dengan jumlah *ErrY* dengan 2 kali DX. Bila *ErrY* tersebut lebih besar dari DY, kemudian diset bahwa *LastX* merupakan jumlah dari *LastX* dengan *Xinc* dan nilai *ErrY* dikurangi 2 kali DY. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture1.Pset* dengan nilai *LastX* dibagi besarnya *ScaleSet* dan *LastY* dibagi besarnya *ScaleSet* dan setting warna garis sesuai dengan *setting Picture1.ForeColor.* Proses tersebut dilanjutkan dengan me*refresh Picture1*. Proses ini diulang hingga nilai *LastY* sama dengan Y2. Setelah selesai *Picture1.ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor.*

```
Private Sub PlotXZ(ByVal bDirX1 As Boolean, ByVal lNumStepX1 As Long, ByVal bDirZ As
Boolean, ByVal lNumStepZ As Long, ByVal lPeriode As Double, ByVal DColor As Long)
  Dim i, X, Z, x1, z1, x2, z2, DX1, DZ As Integer
  Dim X1Inc, ZInc, ErrX1, ErrZ As Integer
  Dim SaveColor As Long

  Dim DL As Double
  SaveColor = Picture2.ForeColor
  Picture2.ForeColor = DColor
  x1 = LastX1
  z1 = LastZ
  DX1 = lNumStepX1 * IIf(bDirX1, 1, -1)
  DZ = lNumStepZ * IIf(bDirZ, 1, -1)
  x2 = x1 + DX1
  z2 = z1 + DZ
  DL = lPeriode
  X1Inc = 1
  ZInc = 1
  If DX1 < 0 Then X1Inc = -1: DX1 = -DX1
  If DZ < 0 Then ZInc = -1: DZ = -DZ
  If DX1 <> 0 Or DZ <> 0 Then
    If DZ <= DX1 Then
      ErrX1 = 0
      Do Until LastX1 = x2
        LastX1 = LastX1 + X1Inc
        ErrX1 = ErrX1 + 2 * DZ
        If ErrX1 > DX1 Then LastZ = LastZ + ZInc: ErrX1 = ErrX1 - 2 * DX1
        Picture2.PSet ((LastX1 / scaleSet), (LastZ / scaleSet)), Picture2.ForeColor
        Delay (DL)
      Loop
```

```
    Else
      ErrZ = 0
      Do Until LastZ = z2
        LastZ = LastZ + ZInc
        ErrZ = ErrZ + 2 * DX1
        If ErrZ > DZ Then LastX1 = LastX1 + X1Inc: ErrZ = ErrZ - 2 * DZ
        Picture2.PSet ((LastX1 / scaleSet), (LastZ / scaleSet)), Picture2.ForeColor
        Delay (DL)
      Loop
    End If
    If DX1 <> 0 And DZ <> 0 Then
      ErrX1 = 0
      Do Until LastX1 = x2 And LastZ = z2
        LastX1 = LastX1 + X1Inc
        LastZ = LastZ + ZInc
        Picture2.PSet ((LastX1 / scaleSet), (LastZ / scaleSet)), Picture2.ForeColor
      Loop
    End If
  End If
  Picture2.ForeColor = SaveColor
End Sub
```

Keterangan:
Prosedur *PlotXZ* digunakan untuk menggambar gerakan alat potong pada arah sumbu XZ. Prosedur ini menerima masukan berupa jumlah *step* gerakan pada arah sumbu X dan Z (*1NumStepX* dan *1NumStepZ*), periode (1Periode), arah gerakan pada sumbu X dan Z (*bDirX* dan *bDirZ*), dan warna garis (*Dcolor*). Pada bagian awal dideklarasikan bahwa variabel I, X, Z, X1, Z1, X2, Z2, DX, DZ, DL, *Xinc, Zinc, ErrX, ErrZ* bertipe *Integer*, sedangkan variabel *savecolor* bertipe *Long*.
Data warna asli dari *Picture2.ForeColor* sementara disimpan dahulu dalam variabel *SaveColor*, *Picture2.ForeColor* diganti dengan data dari *Dcolor*. Kemudian variabel X1 diisi dengan nilai *LastX* (nilai X terakhir) dan Z1 diisi dari *LastZ* (nilai Z terakhir). Sedangkan variabel DX (jarak perpindahan yang harus ditempuh sejajar sumbu X), diisi dari hasil kali antar jumlah *step* dengan nilai arah pergerakan (*1NumStep* * lif(*bdirX*, 1, -1), begitu pula halnya dengan variabel DZ.
Setelah proses diatas, barulah variabel X2 dan Z2 diisi dengan jumlah nilai awal (X1 dan Z1) dengan jarak perpindahan (DX dan DZ). Variabel DL berisi besarnya *delay* dalam proses pengiriman *impuls* listrik ke motor *stepper*. Nilai dari variabel ini diambil dari nilai variabel 1Periode. Kemudian variabel *Xinc* dan *Yinc* diset agar bernilai 1.
Jika DX lebih kecil dari 0 maka diset bahwa *Xinc* bernilai -1 dan DX sama dengan –DX. Begitu pula bila DZ lebih kecil dari 0, maka *Zinc* bernilai -1 dan DZ sama dengan –DZ.
Jika DX tidak sama dengan 0 atau DY tidak sama dengan 0, maka jika DY lebih kecil atau sama dengan DX, maka diset *ErrX* = 0. Nilai *LastX* diisi dengan jumlah dari *LastX* dengan *Xinc*, dan nilai *ErrX* diisi dengan jumlah *ErrX* dengan 2 kali DZ. Bila *ErrX* tersebut lebih besar dari DX, kemudian diset bahwa *LastY* merupakan jumlah dari *LastZ* dengan *Zinc* dan nilai *ErrX* dikurangi 2 kali DX. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture1.Pset* dengan nilai *LastX* dibagi besarnya *ScaleSet* dan *LastZ* dibagi besarnya *ScaleSet* dan *setting* warna garis sesuai dengan *setting Picture1.ForeColor*. Proses tersebut dilanjutkan dengan me*refresh picture1*. Proses ini diulang hingga nilai *LastX* sama dengan X2. Setelah selesai *Picture2. ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor*.
Sebaliknya apabila DZ lebih besar dari DX, maka diset *ErrZ* = 0, kemudian nilai *LastZ* diisi dengan jumlah dari *LastZ* dengan *Zinc*, dan nilai *ErrZ* diisi dengan jumlah *ErrZ* dengan 2 kali DX. Bila *ErrY* tersebut lebih besar dari DY, kemudian diset bahwa *LastX* merupakan jumlah dari *LastX* dengan *Xinc* dan nilai *ErrZ* dikurangi 2 kali DZ. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture2.Pset* dengan nilai *LastX* dibagi besarnya *ScaleSet* dan *LastZ* dibagi besarnya *ScaleSet* dan setting warna garis sesuai dengan *setting Picture2.ForeColor*.

Proses tersebut dilanjutkan dengan me*refresh Picture2*. Proses ini diulang hingga nilai *LastZ* sama dengan Z2. Setelah selesai *Picture2.ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor*.

```
Private Sub PlotYZ(ByVal bDirY1 As Boolean, ByVal lNumStepY1 As Long, ByVal bDirZ1 As
Boolean, ByVal lNumStepZ1 As Long, ByVal lPeriode As Double, ByVal DColor As Long)
  Dim i, Y, Z, y1, z1, y2, z2, DY1, DZ1 As Integer
  Dim Y1Inc, Z1Inc, ErrY1, ErrZ1 As Integer
  Dim SaveColor As Long

  Dim DL As Double
  SaveColor = Picture3.ForeColor
  Picture3.ForeColor = DColor
  y1 = LastY1
  z1 = LastZ1
  DY1 = lNumStepY1 * lIf(bDirY1, 1, -1)
  DZ1 = lNumStepZ1 * lIf(bDirZ1, 1, -1)
  y2 = y1 + DY1
  z2 = z1 + DZ1
  DL = lPeriode
  Y1Inc = 1
  Z1Inc = 1
  If DY1 < 0 Then Y1Inc = -1: DY1 = -DY1
  If DZ1 < 0 Then Z1Inc = -1: DZ1 = -DZ1
  If DY1 <> 0 Or DZ1 <> 0 Then
    If DZ1 <= DY1 Then
      ErrY1 = 0
      Do Until LastY1 = y2
        LastY1 = LastY1 + Y1Inc
        ErrY1 = ErrY1 + 2 * DZ1
        If ErrY1 > DY1 Then LastZ1 = LastZ1 + Z1Inc: ErrY1 = ErrY1 - 2 * DY1
        Picture3.PSet ((LastZ1 / scaleSet), (LastY1 / scaleSet)), Picture3.ForeColor
        Delay (DL)
      Loop
    Else
      ErrZ1 = 0
      Do Until LastZ1 = z2
        LastZ1 = LastZ1 + Z1Inc
        ErrZ1 = ErrZ1 + 2 * DY1
        If ErrZ1 > DZ1 Then LastY1 = LastY1 + Y1Inc: ErrZ1 = ErrZ1 - 2 * DZ1
        Picture3.PSet ((LastZ1 / scaleSet), (LastY1 / scaleSet)), Picture3.ForeColor
        Delay (DL)
      Loop
    End If
    If DY1 <> 0 And DZ1 <> 0 Then
      ErrY1 = 0
      Do Until LastY1 = v2 And LastZ1 = z2
        LastY1 = LastY1 + Y1Inc
        LastZ1 = LastZ1 + Z1Inc
        Picture3.PSet ((LastZ1 / scaleSet), (LastY1 / scaleSet)), Picture3.ForeColor
      Loop
    End If
  End If
  Picture3.ForeColor = SaveColor
End Sub
```

Keterangan:
Prosedur *PlotYZ* digunakan untuk menggambar gerakan alat potong pada arah sumbu YZ. Prosedur ini menerima masukan berupa jumlah *step* gerakan pada arah sumbu Y dan Z (*1NumStepY* dan *1NumStepZ*), periode (1Periode), arah gerakan pada sumbu Y dan Z (*bDirY* dan *bDirZ*), dan warna garis (*Dcolor*). Pada bagian awal dideklarasikan bahwa variabel I, Y, Z, Y1, Z1, Y2, Z2, DY, DZ, DL, *Yinc, Zinc, ErrY, ErrZ* bertipe *Integer*, sedangkan variabel *savecolor* bertipe *Long*.
Data warna asli dari *Picture3.ForeColor* sementara disimpan dahulu dalam variabel *SaveColor*, *Picture3.ForeColor* diganti dengan data dari *Dcolor*. Kemudian variabel Y1 diisi dengan nilai *LastY* (nilai Y terakhir) dan Z1 diisi dari *LastZ* (nilai Z terakhir). Sedangkan variabel DY (jarak perpindahan yang harus ditempuh sejajar sumbu Y), diisi dari hasil kali antar jumlah *step* dengan nilai arah pergerakan (*1NumStep* * lif(*bdirY*, 1, -1), begitu pula halnya dengan variabel DZ.
Setelah proses diatas, barulah variabel Y2 dan Z2 diisi dengan jumlah nilai awal (Y1 dan Z1) dengan jarak perpindahan (DY dan DZ). Variabel DL berisi besarnya *delay* dalam proses pengiriman *impuls* listrik ke motor *stepper*. Nilai dari variabel ini diambil dari nilai variabel 1Periode. Kemudian variabel *Yinc* dan *Zinc* diset agar bernilai 1.
Jika DY lebih kecil dari 0 maka diset bahwa *Yinc* bernilai -1 dan DY sama dengan –DY. Begitu pula bila DZ lebih kecil dari 0, maka *Zinc* bernilai -1 dan DZ sama dengan –DZ.
Jika DY tidak sama dengan 0 atau DZ tidak sama dengan 0, maka jika DZ lebih kecil atau sama dengan DY, maka diset *ErrY* = 0. Nilai *LastY* diisi dengan jumlah dari *LastY* dengan *Yinc*, dan nilai *ErrX* diisi dengan jumlah *ErrX* dengan 2 kali DY. Bila *ErrX* tersebut lebih besar dari DX, kemudian diset bahwa *LastZ* merupakan jumlah dari *LastZ*dengan *Zinc* dan nilai *ErrY* dikurangi 2 kali DY. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture3.Pset* dengan nilai *LastY* dibagi besarnya *ScaleSet* dan *LastZ*dibagi besarnya *ScaleSet* dan *setting* warna garis sesuai dengan *setting Picture1.ForeColor*. Proses tersebut dilanjutkan dengan me*refresh picture3*. Proses ini diulang hingga nilai *LastY* sama dengan Y2. Setelah selesai *Picture3.ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor*.
Sebaliknya apabila DZ lebih besar dari DY, maka diset *ErrZ* = 0, kemudian nilai *LastZ* diisi dengan jumlah dari *LastZ* dengan *Zinc*, dan nilai *ErrZ*diisi dengan jumlah *ErrZ*dengan 2 kali DY. Bila *ErrZ* tersebut lebih besar dari DZ, kemudian diset bahwa *LastY* merupakan jumlah dari *LastY* dengan *Yinc* dan nilai *ErrZ* dikurangi 2 kali DZ. Proses ini dilanjutkan dengan proses penggambaran dengan mengatur *Picture3.Pset* dengan nilai *LastY* dibagi besarnya *ScaleSet* dan *LastZ* dibagi besarnya *ScaleSet* dan setting warna garis sesuai dengan *setting Picture3.ForeColor*. Proses tersebut dilanjutkan dengan me*refresh Picture3*. Proses ini diulang hingga nilai *LastZ* sama dengan Z2. Setelah selesai *Picture3.ForeColor* dikembalikan seperti *setting* semula dengan memberikan masukan dari *SaveColor*.

Form3 (Seputar Program.frm)

```
Private Sub Command1_Click()
Unload Me
End Sub
```

Keterangan:
Prosedur ini akan menutup *form3* (Seputar Program.frm) jika *command1* pada *form2* diklik.

Form4 (Penjelasan.frm)

```
Private Sub Command1_Click()
Unload Me
End Sub
```

Keterangan:
Prosedur ini akan menutup *form4* (Penjelasan.frm) jika *command1* pada *form4* diklik.

Form5 (Parameter.frm)

Option Explicit

Keterangan:
Perintah *Option Explicit* digunakan untuk menyatakan bahwa variabel tersebut dapat diakses walaupun berada diluar *form1*.

```
Private Sub Form_Load()
text1(0).Text = A
Text2(0).Text = B
Text3(0).Text = C
Text4(0).Text = D
text5.Text = E
Text6.Text = F
Text7.Text = G
Text8.Text = H
Text9.Text = i
Text10.Text = J
Text11.Text = K
Text12.Text = L
End Sub
```

Keterangan:
Pada saat *form* ini ditampilkan prosedur ini akan melakukan proses pengisian variabel dari *text1(0).text* hingga *text12.text* dengan nilai yang telah terisi pada variabel A, B, C, D, E, F, G, H, I, J, K, dan L.

```
Private Sub exit_Click()
A = text1(0).Text
B = Text2(0).Text
C = Text3(0).Text
D = Text4(0).Text
E = text5.Text
F = Text6.Text
G = Text7.Text
H = Text8.Text
i = Text9.Text
J = Text10.Text
K = Text11.Text
L = Text12.Text
If A = 0 Or C = 0 Or E = 0 Or G = 0 Then
MsgBox "NILAI PARAMETER MASUKAN RESOLUSI, KECEPATAN, AKSELERASI TIDAK
BOLEH BERNILAI 0! ATAU CLICK TOMBOL RESET ", , "PERINGATAN!!"
RunFlag = False
Else
Unload Me
End If
End Sub
```

Keterangan:
Prosedur ini akan melakukan pengisian terhadap variabel A, B, C, D, E, F, G, H, I, J, K, dan L dengan *text1(0).text* hingga *text12.text* jika *commandexit (Caption OK)* diklik. Kemudian dilanjutkan pengecekan bila variabel A dan C atau E dan G mendapat nilai masukan nol maka

akan muncul *MsgBox* "Nilai parameter masukan Resolusi, Kecepatan, Akselerasi tidak boleh bernilai 0!" dilanjutkan dengan mengeset *RunFlag* bernilai *false*. Namun bila A, C, E, dan G tidak bernilai nol maka dilanjutkan prosedur *unload form*.

```
Private Sub RESET_Click()
text1(0).Text = 0.05
A = 0.05
Text2(0).Text = 0
B = 0
Text3(0).Text = 1200
C = 1200
Text4(0).Text = 20
D = 20
text5.Text = 0.05
E = 0.05
Text6.Text = 0
F = 0
Text7.Text = 1200
G = 1200
Text8.Text = 20
H = 20
Text9.Text = 0.05
i = 0.05
Text10.Text = 0
J = 0
Text11.Text = 1200
K = 1200
Text12.Text = 20
L = 20
End Sub
```

Keterangan:
Prosedur ini akan kembali melakukan pengisian terhadap variabel A, B, C, D, E, F, G, H, I, J, K, dan L pada saat *commandRESET* diklik.

Formmaterial (Material.frm)

```
Option Explicit
```

Keterangan:
Perintah *Option Explicit* digunakan untuk menyatakan bahwa variabel tersebut dapat diakses walaupun berada diluar *form* ini.

```
Private Sub Command1_Click()
Dim Pilihan As Boolean
PANJANGX = Text1.Text
PANJANGY = Text2.Text
TEBALB = Text3.Text
TITIKX1 = Text4.Text
TITIKY1 = Text5.Text
DIAMETERMATERIAL = Text6.Text
TEBALS = Text7.Text
TITIKX2 = Text8.Text
TITIKY2 = Text9.Text
Unload formmaterial
```

```
End Sub
```

Keterangan:
Prosedur ini diawali pengisian terhadap variabel PanjangX, PanjangY, TebalB, TitikX1, TitikY1, Diametermaterial, TebalS, TitikX2, dan TitikY2 dengan *text1.text* hingga *text9.text* jika *command1* diklik. Prosedur terakhir yang dilakukan *unload formmaterial*.

```
Private Sub Command2_Click()
PANJANGX = 0
PANJANGY = 0
TEBALB = 0
TITIKX1 = 0
TITIKY1 = 0
DIAMETERMATERIAL = 0
TEBALS = 0
TITIKX2 = 0
TITIKY2 = 0
Unload formmaterial
End Sub
```

Keterangan:
Prosedur ini diawali pengisian terhadap variabel PanjangX, PanjangY, TebalB, TitikX1, TitikY1, Diametermaterial, TebalS, TitikX2, dan TitikY2 dengan nilai 0 (nol) jika *command2* diklik. Prosedur terakhir yang dilakukan *unload formmaterial*.

```
Private Sub Form_Load()
Text6.BackColor = &H8000000F
Text7.BackColor = &H8000000F
Text8.BackColor = &H8000000F
Text9.BackColor = &H8000000F
Text1.Text = PANJANGX
Text2.Text = PANJANGY
Text3.Text = TEBALB
Text4.Text = TITIKX1
Text5.Text = TITIKY1
Text6.Text = DIAMETERMATERIAL
Text7.Text = TEBALS
Text8.Text = TITIKX2
Text9.Text = TITIKY2
End Sub
```

Keterangan:
Prosedur ini untuk melakukan identifikasi variabel *text* dan pengaturan awal warna.

```
Private Sub Option1_Click()
silinder = False
Text6.Enabled = False
Text7.Enabled = False
Text8.Enabled = False
Text9.Enabled = False
Text6.BackColor = &H8000000F
Text7.BackColor = &H8000000F
Text8.BackColor = &H8000000F
Text9.BackColor = &H8000000F
Text1.Enabled = True
```

```
Text2.Enabled = True
Text3.Enabled = True
Text4.Enabled = True
Text5.Enabled = True
Text1.BackColor = &H80000005
Text2.BackColor = &H80000005
Text3.BackColor = &H80000005
Text4.BackColor = &H80000005
Text5.BackColor = &H80000005
End Sub
```

Keterangan:
Prosedur ini ditampilkan dalam bentuk *option button* untuk melakukan inisialisasi pada *formmaterial* dengan melakukan beberapa pengaturan warna batasan *property* tertentu.

```
Private Sub Option2_Click()
silinder = True
Text1.Enabled = False
Text2.Enabled = False
Text3.Enabled = False
Text4.Enabled = False
Text5.Enabled = False
Text1.BackColor = &H8000000F
Text2.BackColor = &H8000000F
Text3.BackColor = &H8000000F
Text4.BackColor = &H8000000F
Text5.BackColor = &H8000000F
Text6.Enabled = True
Text7.Enabled = True
Text8.Enabled = True
Text9.Enabled = True
Text6.BackColor = &H80000005
Text7.BackColor = &H80000005
Text8.BackColor = &H80000005
Text9.BackColor = &H80000005
End Sub
```

Keterangan:
Prosedur ini ditampilkan dalam bentuk *option button* untuk melakukan inisialisasi pada *formmaterial* dengan melakukan beberapa pengaturan warna batasan *property* tertentu.

Module1 (Module1.bas)

```
Option Explicit
Global Const pi = 3.14159265358979
```

Keterangan:
Pendeklarasian dan pengisian variabel pi dengan nilai 3.14159265358979 dan sekaligus variabel ini bisa digunakan untuk semua pengoperasian di semua *form*.

```
Public testcond As Boolean
Public RunFlag As Boolean
Public A As Double
Public B As Double
Public C As Double
```

```
Public D As Double
Public E As Double
Public F As Double
Public G As Double
Public H As Double
Public i As Double
Public J As Double
Public K As Double
Public L As Double
Public PANJANGX As Double
Public PANJANGY As Double
Public TEBALB As Double
Public DIAMETERMATERIAL As Double
Public TEBALS As Double
Public TITIKX1 As Double
Public TITIKY1 As Double
Public TITIKX2 As Double
Public TITIKY2 As Double
Public silinder As Boolean
Public balok As Boolean
```

Keterangan:
Pendeklarasian variabel A, B, C, D, E, F, G, H, I, J, K, L, PANJANGX, PANJANGY, TEBALB, DIAMETERMATERIAL, TEBALS, TITIKX1, TITIKY1, TITIKX2, TITIKY2 bertipe *double*. Variabel *testcond, Runflag,* silinder, dan balok bertipe *Boolean*.

```
Public Sub Delay(pauseTime As Double)
Dim Start, Finish As Double
 Start = Timer
 Do While (Timer < Start + pauseTime)
  DoEvents
Loop
 Finish = Timer
End Sub
```

Keterangan:
Prosedur ini dimulai dengan pendeklarasin masukan prosedur *Delay* berupa *pauseTime* yang bertipe *Double*. Variabel *Start* dan *Finish* bertipe *Double*. Proses selanjutnya dilakukan pengisian variabel *start* dengan prosedur *timer*. Dibuat pembatasan untuk pengulangan prosedur *DoEvent* dengan *timer* lebih kecil dari jumlah nilai *start* dengan *pauseTime*.

```
Public Function nexttoken(thestring As String, thetoken As String, theval As Double) As Boolean
   Dim Tempstr As String
   Dim i As Long
   nexttoken = False
   thetoken = Left(thestring, 1)
   If thetoken <> "" Then
      i = InStr(2, thestring, " ", vbBinaryCompare)
      If i = 0 Then
         Tempstr = Mid(thestring, 2)
         thestring = ""
      Else
         Tempstr = Mid(thestring, 2, i - 2)
         thestring = Mid(thestring, i + 1)
      End If
```

```
    If IsNumeric(Tempstr) Then
      theval = Val(Tempstr)
      nexttoken = True
    End If
  End If
End Function
```

Keterangan:
Prosedur *nexttoken* berfungsi untuk memilah dan membaca karakter dari Tempstr. Variabel *nexttoken* dideklarasikan sebagai *string*. Prosedur ini dipanggil oleh prosedur *commandrun* pada *form* simulasi dengan nilai masukan berupa karakter *TempStr* dan *TokenChr* bertipe *string* serta nilai variabel (val) yang bertipe *Double*.
Pada awal prosedur *nexttoken* diset *false* dan *thetoken* merupakan karakter dari *thestring*. Kemudian dilakukan pengisian terhadap *TempStr* dan *thestring* dengan memperhatikan kondisi dari nilai i. Kemudian dilakukan pengecekan pada karakter *Tempstr*, jika terdapat angka numerik maka *Theval* diisi nilai numerik dari *tempstr* tersebut dan *nexttoken* diset *true*.

```
Public Function DegAcos(ByVal X As Double) As Double
  DegAcos = (Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)) * 180 / pi
End Function
```

Keterangan:
Prosedur ini mendeklarasikan fungsi DegAcos yang didapat dari persamaan DegAcos = (Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)) * 180 / pi

```
Public Function DegAsin(ByVal X As Double) As Double
  DegAsin = Atn(X / Sqr(-X * X + 1)) + 2 * 180 / pi
End Function
```

Keterangan:
Prosedur ini mendeklarasikan fungsi DegAsin yang didapat dari persamaan DegAsin = Atn(X / Sqr(-X * X + 1)) + 2 * 180 / pi

```
Public Function DegATan(ByVal X As Double) As Double
  DegATan = Atn(X) * 180 / pi
End Function
```

Keterangan:
Prosedur ini mendeklarasikan fungsi DegATan yang didapat dari persamaan DegATan = Atn(X) * 180 / pi

```
Public Function Degcos(ByVal X As Double) As Double
  Degcos = Cos(X * pi / 180)
End Function
```

Keterangan:
Prosedur ini mendeklarasikan fungsi Degcos yang didapat dari persamaan Degcos = Cos(X * pi / 180)

```
Public Function Degsin(ByVal X As Double) As Double
  Degsin = Sin(X * pi / 180)
End Function
```

Keterangan:

Prosedur ini mendeklarasikan fungsi Degsin yang didapat dari persamaan Degsin = Sin(X * pi / 180)

```
Public Function DegTan(ByVal X As Double) As Double
DegTan = Tan(X * pi / 180)
End Function
```

Keterangan:
Prosedur ini mendeklarasikan fungsi DegTan yang didapat dari persamaan DegTan = Tan(X * pi / 180)

LAMPIRAN 2

NC CODE UNTUK BLOCK CYLINDER

```
N1 G17 S400 T1 M66
N2 G54
N3 G1 Z50 M3
N4 G0 Y-30
N5 G1 Z-28 F100
N6 G1 X40.3
N7 G1 Y30
N8 G1 X-40.3
N9 G1 Y-30
N10 G1 X20.3
N11 G1 Z-12.4
N12 G1 Y30
N13 G1 X-20.3
N14 G1 Y-30
N15 G1 X0
N16 G1 Z50
N17 G0 Y0
N18 G81 Z-24 D7.3
N19 G1 Z10
N20 G0 Y-20
N21 G81 Z-24 D9
N22 G1 Z10
N23 G0 Y20
N24 G81 Z-24 D9
N25 G1 Z10
N26 G81 Z-5 D14
N27 G1 Z10
N28 G0 Y-20
N29 G81 Z-5 D14
N30 G1 Z5
N31 G0 Z50
N32 M30
```

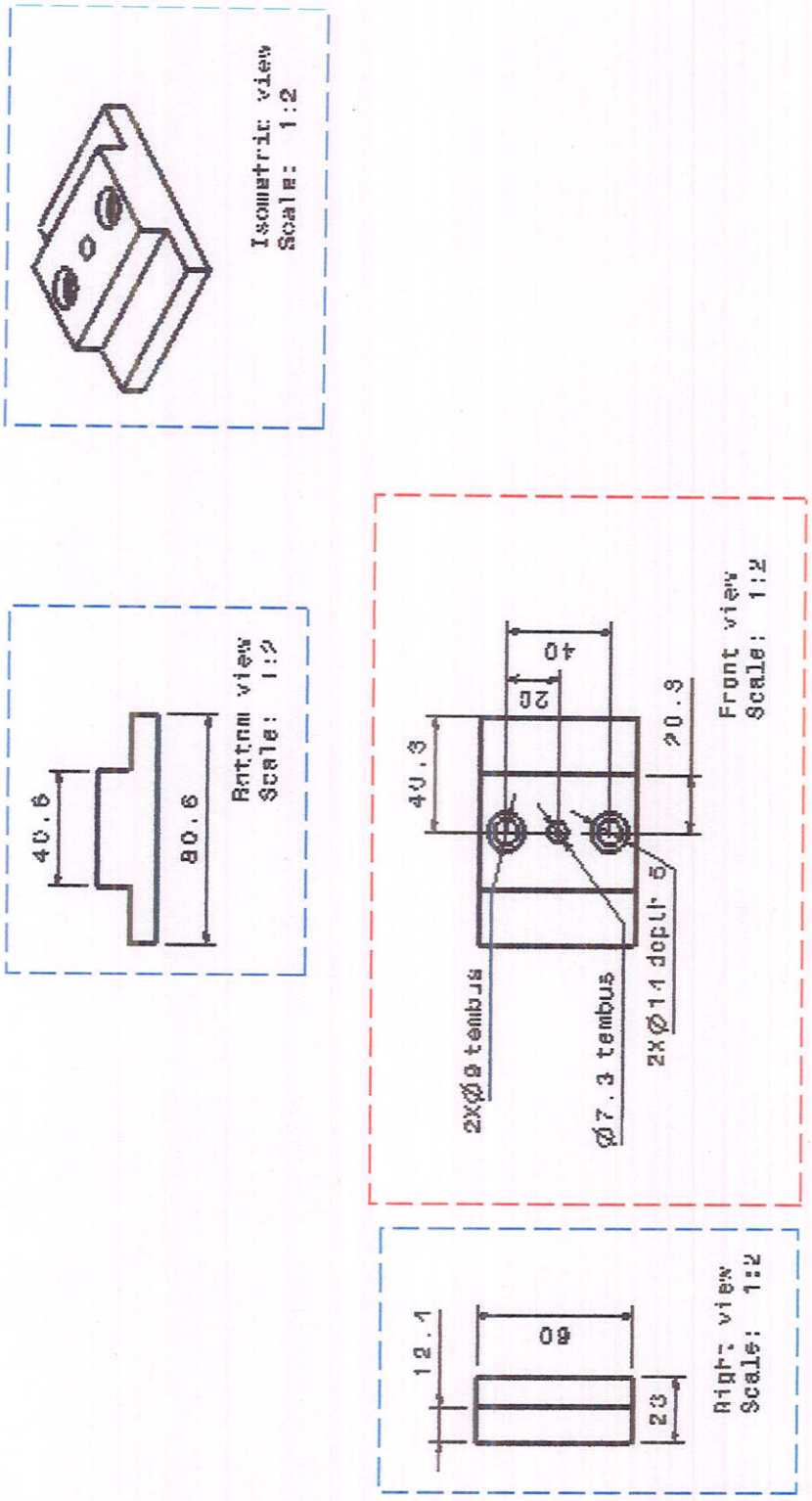
LAMPIRAN 3

NC CODE UNTUK BLOCK SUPPORT

```
N1 G17 S800 T1 M66
N2 G54
N3 G0 X5 Y5 M3
N4 G1 Z-5 F100
N5 G1 Y-150
N6 G1 X130
N7 G3 X150 Y-130 I10 J10
N8 G1 Y-5
N9 G1 X0
N10 G1 Z20
N11 G0 X50 Y-30
N12 G1 Z-5
N13 G88 I25 J15 K10 F100
N14 G89 Z-10 I12.5 J12.5 K10
N15 G1 Z20
N16 G0 X136 Y-19
N17 G1 Z-2.7 F100
N18 G1 X107.72 Y-47.28
N19 G1 X114.79 Y-54.35
N20 G1 X143.07 Y-26.07
N21 G1 X136 Y-19
N22 G1 Z11.5
N23 G0 X23.75 Y-62.5
N24 G1 Z-5.2
N25 G1 Y-130
N26 G1 X36.25
N27 G1 Y-62.5
N28 G1 X23.75
N29 G1 Z5
N30 G0 Z30
N31 G0 X70 Y-130
N32 S1000 T2 M66
N33 G81 Z-20 D6
N34 G1 Z10
N35 G0 X86.7 Y-112.5
N36 G81 Z-20 D6
N37 G1 Z10
N38 G0 X102.62 Y-112.5
N39 G81 Z-20 D6
N40 G1 Z10
N41 G0 X115.42
N42 G81 Z-20 D6
N43 G1 Z10
N44 G0 X133.13 Y-130
N45 G81 Z-20 D6
N46 G1 Z10
N47 G90
N48 G0 Z50
N49 M30
```

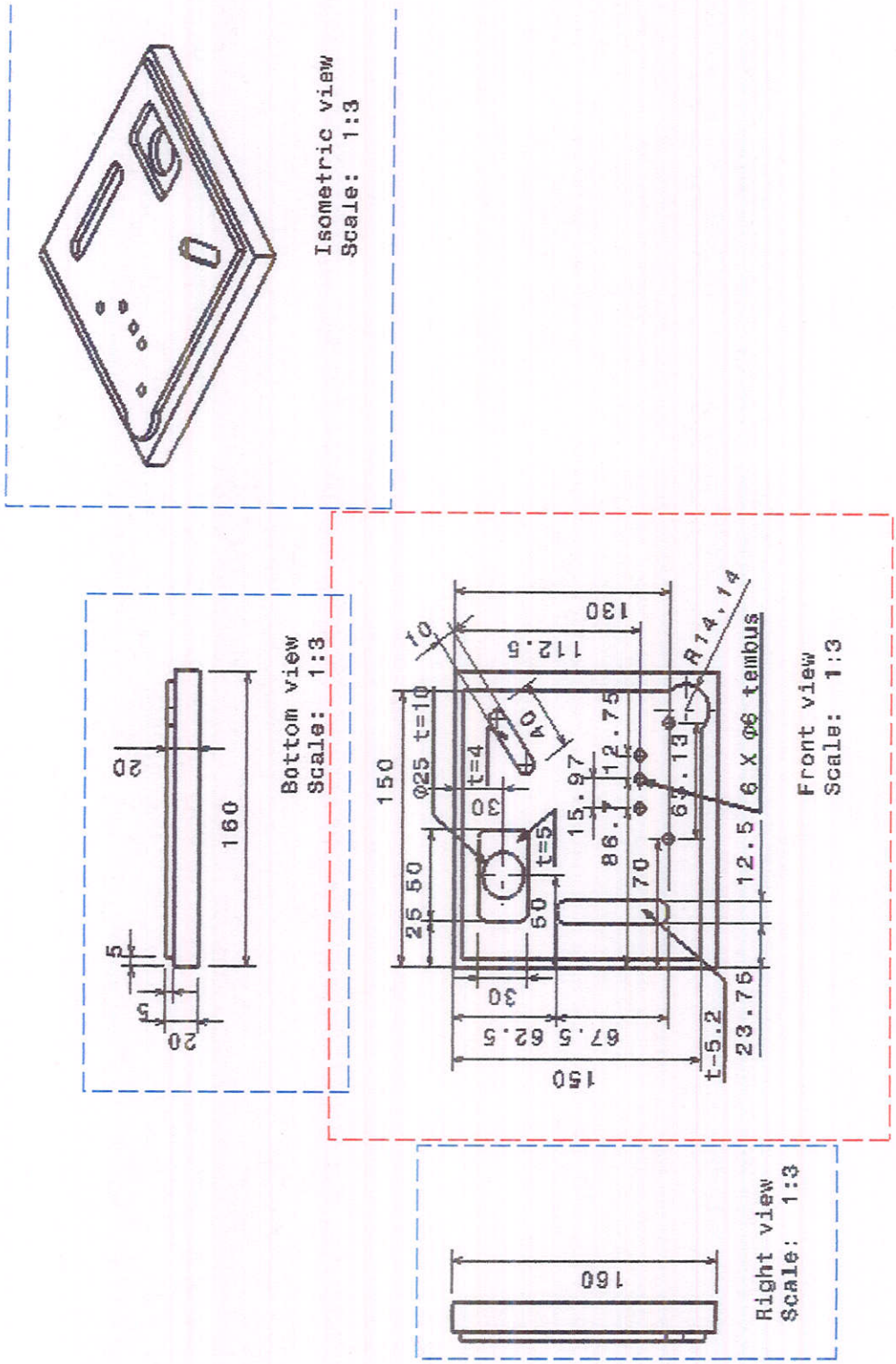
LAMPIRAN 4

GAMBAR PART BLOCK CYLINDER



LAMPIRAN 5

GAMBAR PART BLOCK SUPPORT



UNIVERSITAS
GADJAH MADA

G-functions

MAHO

LAMPIRAN 6

G FUNCTION

	Meaning
G0	Rapid traverse
G1	Linear interpolation
G2	Circular interpolation, clockwise
G3	Circular interpolation, anti-clockwise
G4**	Dwell time (0,1 to 983 sec.)
G11**	Polar coordinates, corner rounding, chamfer shifting
G14**	Jump command and repeat function
G17*	Plane selection XY, horizontal
G18	Plane selection XZ, vertical
G19	Plane selection ZY, horizontal rotated through 90°
G22**	Subroutine call
G25*	Feed override effective
G26	Feed 100%
G27*	Feed motion with smooth transition
G28	Feed motion with precision stop
G29**	Conditional jump command
G40*	No radius offset
G41	Radius offset, left
G42	Radius offset, right
G43	Radius offset, up to
G44	Radius offset, past
G51	Erasing G52
G52	Activating shift value of Reset AXIS
G53*	No stored ZP-offset
G54	Stored ZP-offset 1
G55	Stored ZP-offset 2
G56	Stored ZP-offset 3
G57	Stored ZP-offset 4
G58	Stored ZP-offset 5
G59	Stored ZP-offset 6
G63	Switching off the geometry-data processing system
G64	Switching on the geometry-data processing system

	Meaning
G70	Inch-input system
G71*	Metric input system
G72*	No mirror-image machining
G73	Mirror-image machining
G77**	Hole circle definition
G78**	Point definition
G79**	Cycle call
G81	Drilling cycle
G83	Deep-hole drilling cycle
G84	Tapping cycle
G85	Reaming cycle
G86	Boring cycle
G87	Pocket milling cycle
G88	Groove milling cycle
G89	Circular pocket milling cycle
G90*	Programming absolute dimensions
G91	Programming incremental dimensions
G92**	ZP-offset incremental
G93**	ZP-offset absolute
G94*	Feed rate mm/min, unit 0,001 mm/min.
G95	Feed rate mm/r, unit 0,001 mm/r
G98	Display window
G99	Blank contour

Explanation of symbols: * = Switch-on position
** = effective for blocks only

UNIVERSITAS
GADJAH MADA

MAHO

M-functions

LAMPIRAN 7

M FUNCTION

	Meaning
M0**	Program stop
M3	Workspindle-clockwise rotation
M4	Workspindle anti-clockwise rotation
M5	Workspindle stop
M6**	Tool change with automatic retraction
M7	Coolant No. 2 on
M8	Coolant No. 1 on
M9	Coolant off
M10	NC-circular table clamped
M11	NC-circular table released
M13	Workspindle-clockwise rotation and coolant on
M14	Workspindle anti-clockwise rotation and coolant on
M16	Erasing M17 and M18
M17	Chip rinsing
M18	Workpiece cleaning
M19	Oriented spindle stop
M20**	Additional M-function
M21	2nd change speed at M6, M46
M30**	Program end
M46**	Tool change at any position
M60**	Pallet change
M61**	Pallet change left pallet
M62**	Pallet change right pallet
M66**	Tool change at actual axis position
M67**	Tool correction change

Explanation of symbols: * = Switch-on position
** = effective for blocks only