Chapter 10

Sample Trapezoidal, 'S' Curve, and DAC Override Profiles

Features

Using Trapezoidal and 'S' Curve Acceleration Profiles
Sample Trapezoidal and S Curve Algorithms
Trapezoidal Trajectory Generator
Trapezoidal and 'S' Profile Trajectory Generator
A Simple PID System Simulator
An Actual Profile Generator: 'S' to Trapezoidal

Using Trapezoidal and 'S' Curve Acceleration Profiles

It is a never-ending controversy when listening to discussions about *trapezoidal* and *S curve* acceleration profiles. Comparing them is tantamount to comparing apples and oranges. There is no similarity between them; they each have a reason for use. For those who are unfamiliar with these profiles, refer to Figures 10.1 and 10.2.

The **trapezoidal profile** changes velocity in a *linear* fashion until it reaches the move velocity. When decelerating, the velocity again changes in a linear manner until it reaches zero velocity (the stop position). It is a fairly simple exercise to determine at what position the deceleration should begin. The acceleration/deceleration values set in to the profile generator are used to calculate the velocity at any time, and the velocity value set in to the profile generator simply governs the maximum velocity at which to run the profile. Therefore, if the current velocity is greater than the maximum velocity then the current velocity is set equal to the maximum velocity. A software routine for the Trapezoidal profile is given later in this chapter.

The ***S curve*** is a controlled velocity (or acceleration) profile. Basically, there are three profile properties that can be controlled to form the *S*. They are: *acceleration rate, deceleration rate,* and *velocity* parameters.

If your system is using an *S* curve acceleration ramp, the general rule is that the deceleration ramp will match. Because this is usually the case, the formulation of the S curve need only deal with the acceleration rate and velocity parameters (Decel = Accel). In this chapter, you will find a software routine for the *S* curve profile, generated by adjusting the acceleration rate, and the velocity.

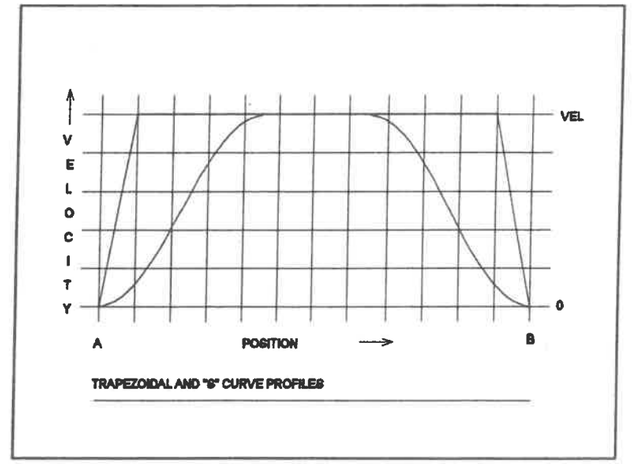

Figure 10.1 Trapezoidal versus *S* profile.

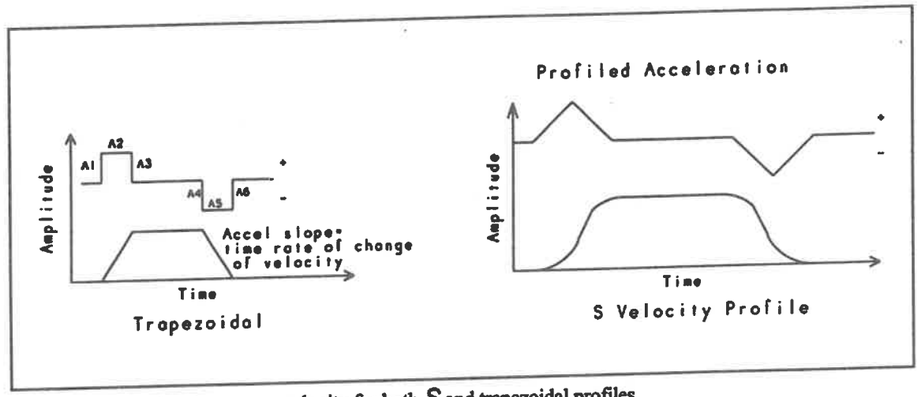

Figure 10.2 Acceleration versus velocity for both *S* and trapezoidal profiles.

The trapezoidal profile is the faster of the two styles when moving equivalent distances. The *S* curve generates a softer takeoff and landing, thus lowering inertia torques. Therefore, the *S* curve would be better suited for an action that does not clamp its parts or is relatively loose in construction (i.e., belts), while the trapezoidal profile would be a better fit for the action of moving quickly. Obviously, it will be necessary to determine when and where either of these profiles would be used. Since many motion control chips such as the *National* LM628 and *Nippon* 3AM generate trapezoidal profiles, a software library would be used to form the *S* curve by manipulation of the velocity and/or the acceleration values.

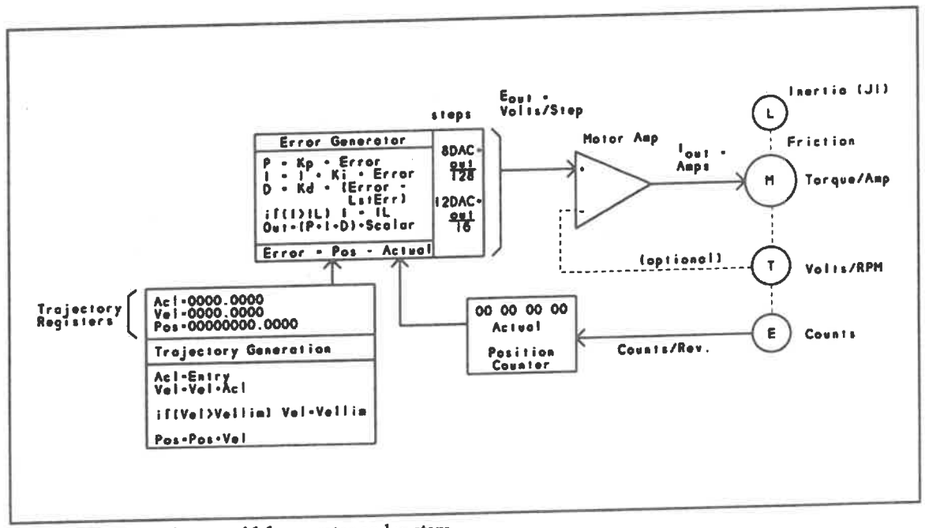

Figure 10.3 Typical trapezoidal generator and system.

Figure 10.3 shows a basic flowchart for a trapezoidal generator. Below are several trajectory algorithms written in C and in BASIC to demonstrate each of the profile styles.

A Trapezoidal Algorithm

A = User Defined Acceleration Rate

V = V + A

P = P + V + 1/2A

An 'S' Curve Algorithm

J = User Defined Acceleration Ramp (Jerk)

A = A + J

V = V + A + 1/2J

P = P + V + 1/2A + 1/6J

A Simple Velocity Controlled Trapezoidal Trajectory Generator

Written in 'C' Programming Language:

```
do_update:          samplecount = samplecount + 1
                 sampletime = smplecnt * updtetime

                     /*trapezoidal trajectory generator*/

                 if (traj_position < decl_position){
                        velocity = velocity + traj_accel;
                        if (velocity >= max_velocity)
                               velocity = max_velocity;
                 }
                 else {
                        velocity = velocity-traj_accel;
                        if (velocity < 0) velocity = 0;
                 }
                 traj_position = traj_position + velocity;
                 error_count = traj_position - real_position;

                     /*calculate the new DAC step output*/

                 prop_val = (kp * error_cnt);
                 integ_err_sum = integ_err_sum + error_cnt;
                 integ_scale = (ki * integ_err_sum)/16;

                 if (integ_scale > integ_lim)
                        integ_scale = integ_lim;
                 if (integ_scale < -integ_lim)
                        integ_scale = -integ_lim;

                 diff_current_update_time = diff_current_update_time + 1;
```

```
if (diff_current_update_time >= diff_requested_sample_time){
        if ((error_count >=0)&&(last_error >= 0)){
                error_diff = error_count-last_error;
        }
        else if ((error_count < 0)&&(last_error < 0)){
                error_diff = -(abs(error_count) - abs(last_error));
        }
        else {
                error_diff = error_count-last_error;
        }
        diff_val = kd * eror_diff;
        differ_act_tme = 0;
        last_error = error_count;
}
DAC_step_output = prop_val + integ_scale + diff_val;

if (max_DAC == 8_bit) {
        DAC_step_output = DAC_steps/256;
}
else {
        DAC_steps = DAC_steps/16;
}

if (DAC_step_output > max_DAC) DAC_steps = max_DAC;
else if (DAC_step_output < -max_DAC) DAC_steps = -mac_DAC;

   /*estimate the new mechanical position*/

if (mtr_steps != DAC_setp_output){
        samplecount1 = 1;
}
else {
        samplecount1 = samplecount1 + 1;
}

sampletime1 = samplecount1 * updatetime;
elec_delay = exp(-(smpletme1/mtr_elec_timeconstant));

/*See SMT equation on page 95 for mechanical time constant
  calculation*/


mech_delay = exp(-(smpletme1/mtr_mech_timeconstant));
stepchange =(DAC_steps-mtr_steps)*(1 - elec_delay)*(1-Mech_delay);

mtr_steps = mtr_steps + stepchange;
real_velocity = (mtr_steps/max_DAC) * max_velocity;
real_position = real_position + real_velocity;

if (desired_position != real_position) goto do_update;
```


A Simple Velocity Controlled Trapezoidal and 'S' Profile Trajectory Generator

Written in 'C' Programming Language:

```
Update:

SmpleCnt++;
SmpleTme = (float) SmpleCnt * UpdteTme;

                    A Trajectory Generation "Algorithm"

if ((ProfileStyle == 'S')  /* Do the 'S' curve profile (if req) */
   { Sample_count++;
  if ((S_SampleTime >= (Move_Acl_Time + Move_Slew_Time)) && (Dcl_Flg == 0))
                    { Scount = 5000;      Dcl_Flg = 1;   Acl_Cnt = 0; }
 if (Scount * UpdteTme >= .001)
    { if ((SmpleTme < MVE_ACLTME) && (SVelocity < Max_Velocity)
                                  && (Flg1 == 0))
           { if (SmpleTme <= (MVE_ACLTME / 2))        /* 1/2 slope height */
                    { Acl_Val = Acl_Val + S_Acl; }
             else   { Acl_Val = Acl_Val - S_Acl;
                      if (Acl_Val < 0)  Acl_Val = 0;
                    }
             SVelocity  = SVelocity + Acl_Val;
             if (SVelocity >= Max_Velocity)  { SVelocity = Max_Velocity; }
           }
      else if (SmpleTme > (MVE_ACLTME + MVE_RUNTME))
             { if (SmpleTme <= (MVE_ACLTME + MVE_RUNTME +
                               (MVE_DCLTME / 2)))     /* 1/2 slope height */
                    { Acl_Val = Acl_Val + S_Acl; }
               else   { Acl_Val = Acl_Val - S_Acl; }
               if (Acl_Val < 0)       Acl_Val = 0;
               SVelocity = SVelocity - Acl_Val;
               if (SVelocity <= 0)  { SVelocity = 0; Prof = 0; }
             }
        Scount = 0;
    }
  if (Velocity > SVelocity)
                { Velocity = Velocity - (3 * Traj_Accel);
                  if (Velocity < 0)      Velocity = 0;
                }
  else if (Velocity < SVelocity)
                { Velocity = Velocity + (3 * Traj_Accel);
                  if (Velocity >= SVelocity) Velocity = SVelocity;
                }
  }
 else                    /* Do Trapezoidal */
   {  if (Traj_Position < Decl_Position)
       {  Velocity = Velocity + Traj_Accel;
          if (Velocity >= Max_Velocity)     Velocity = Max_Velocity;
       }
   else {  Velocity = Velocity - Traj_Accel;
           if (Velocity < 0)   Velocity = 0;
        }
   }
Traj_Position = Traj_Position + Velocity;

Error_Cnt = Traj_Position - Real_Position;

Get the NEW Trajectory Generator DAC Step Output PID using the National LM628

/* Proportional Kp */
Prop_Val = Kp * Error_Cnt;

/* Integral Ki */
Integ_Err_Sum = Integ_Err_Sum + Error_Cnt;
Integ_Scale = Ki * Integ_Err_Sum / 16;
if (Integ_Scale > Integ_Lim) Integ_Scale  = Integ_Lim;
if (Integ_Scale < -Integ_Lim) Integ_Scale = -Integ_Lim;

/* Differential Kd and DST */
```

```
Differ_Act_Tme++;
if (Differ_Act_Tme >= Differ_Smple_Tme)
   {   if ((Error_Cnt >= 0 ) && (Last_Eror >= 0))
          { Eror_Diff = Error_Cnt - Last_Eror; }
     else if ((Error_Cnt < 0) && (Last_Eror < 0))
          { if (abs(Error_Cnt) > abs(Last_Eror))
                    { Eror_Diff = abs(Last_Eror) - abs(Error_Cnt); }
            else    { Eror_Diff = abs(Error_Cnt) - abs(Last_Eror); }
          }
     else { if ((Error_Cnt >=0) && (Last_Eror < 0))
               { Eror_Diff = Error_Cnt + abs(Last_Eror); }
            else if ((Error_Cnt < 0) && (Last_Eror >= 0))
               { Eror_Diff = - (abs(Error_Cnt) + abs(Last_Eror)); }
          }
     Diff_Val = Kd * Eror_Diff;
     Differ_Act_Tme = 0;
     Last_Eror = Error_Cnt;
     }

/* 12 Bit DAC Result */
DAC_Steps = Prop_Val + Integ_Scale + Diff_Val;
if (DAC_Steps > Max_DAC) DAC_Steps = Max_DAC;
else if (DAC_Steps < -Max_DAC) DAC_Steps = -Max_DAC;

UpdateExt:
```

Return from the Trajectory handler routine.

A Simple Acceleration Controlled Profile Generator

Reference to which part of profile curve the program code is dealing with is shown in { }. (See Case figures 1,2, and 3 (A,E,B,G,C,F,D) in Figures 10.3, 10.4, and 10.5)

```
10      '
20      MVELFLG=0
30      'SAVE "SCURVE.asc"  ,A
40      CLS  :J=0! :A=0 :V=0 :P=0 :FLG1=0 :CNT=0 :FLG2=0 :FLG3=0
50      UPDTE = .0002 :'fixed per axis
60      'Resolution
70      IF RESO <>0 GOTO 90
80      RESO=10000
90      LOCATE 1,1: PRINT "input the RESOLUTION ("RESO"CNTS/IN)":INPUT RESO1
100     IF RESO1 <>0 THEN RESO=RESO1
110     CLS
120     'Target Position
130     IF TGPOS <>0 GOTO 150
140     TGPOS=1                                 :'user defined INCH
150     LOCATE 1,1: PRINT "input the target position ("TGPOS"IN):INPUT TGPOS1
160     IF TGPOS1 <>0 THEN TGPOS = TGPOS1
170     LOCATE 1,1: TP =TGPOS :PRINT "TP (IN) = "TP"            "TP*RESO" CNTS"
180     'Move Velocity
190     IF MVELFLG<>0 GOTO 210
200     IF MOVVEL <>0 GOTO 220
210     MOVVEL=250: MVELFLG=0                   :'user defined IN/MIN
220     LOCATE 2,1:PRINT "input the move vel ("MOVVEL"IN/MIN)":INPUT MOVVEL1
230     IF MOVVEL1 <>0 THEN MOVVEL=MOVVEL1
240     LOCATE 2,1: PRINT "MOVE VEL = "MOVVEL" IN/MIN           "MOVVEL/60*RESO/500
250     'Max Acceleration
260     IF MXACL <>0 GOTO 280
270     MXACL=20: SACL=MXACL                    :'user defined IN/SEC^2
280     LOCATE 3,1: PRINT "input the MAX ACCEL ("MXACL"IN/SEC@)":INPUT MXACL1
290     IF MXACL1 <>0 THEN MXACL=MXACL1:      sacl=mxacl      :'user defined IN/SEC^2
300     SACL=MXACL
310     LOCATE 3,1: PRINT "MAX ACCEL = "MXACL" IN/SEC^2           "
320     'Input Jerk
330     LOCATE 4,1: PRINT "input the J factor ("J"IN/SEC^3)":INPUT J1
340     'Move Velocity
350     T=(MOVVEL/60)/MXACL                     :'V/MIN/60/ACL = SECS TO GET TO VEL
360     LOCATE 4,1: PRINT"ACL TIME TO VEL = "T" SECS          "T/UPDTE" SAMPLES  "
370     '
380     S = MXACL / 2 * (T * T)                 :'DISTANCE = 1/2 MXACL T^2
390     LOCATE 5,1: PRINT "DIST TO ACL (S=1/2AT^2) = "S" IN          "S*RESO" CNTS"
400     '
410     IF TP>2*S THEN GOTO 500                 :'IS DIST *2 > MOVE DIST
420     'Modify Move Velocity
430     MVELFLG=1
440     MOVVEL= (TP/(2*S))*MOVVEL               :'MVDIST/REQDIST*MXVEL=MODIFIED MOVE VEL
450     LOCATE 2,1: PRINT "MODIFIED MOVE VEL = "MOVVEL" IN/MIN          "MOVVEL/60
460     '
470     T= SQR((2*(TP/2))/MXACL)                :' SECS TO GET TO VEL
480     LOCATE 4,1: PRINT "ACL TIME TO VEL = "T" SECS          "T/UPDTE" SAMPLES
490     '
500     TN=T/UPDTE                              :'NUM OF SMPLS TO FINAL VEL = N SAMPLES
510     N=TN/2                                  :'NUM OF SMPLS TO MXACL = 1/2 N SAMPLES
520     '
530     SMODACL=SACL*RESO*UPDTE*UPDTE
540     LOCATE 6,1: PRINT "MOVE ACE = "SMODACL" CTS/SMP^2"
550     'Determine the Jerk
560     '                                BACK CALC THE J
570     J = SMODACL / N
580     '*******************************
590     IF J1<J THEN GOTO 630
600     IF J1<SMODACL/2 THEN J=J1 : GOTO 630
610     J=SMODACL/2 : GOTO 630
620     J = SMODACL / N
630     LOCATE 7,1: PRINT "JERK = "J
640     '
650     'DETERMINE THE SECTION TO RUN
660     '*******************************
670     '
```

```
680     LOCATE 9,1 : PRINT "J=+J
690     CNT=CNT+1
700     '
710     IF FLG1=6 GOTO 1440
720     IF FLG1=5 GOTO 1340
730     IF FLG1=4 GOTO 1230
740     IF FLG1=3 GOTO 1130
750     IF FLG1=2 GOTO 1020
760     IF FLG1=1 GOTO 920
770     '
780     'DO THE S ACL RAMP-UP                    :{ Ref A part of profile curve}
790     '*******************************
800     '
810     IF (A+J)>=(SACL*RESO*UPDTE*UPDTE) THEN FLG1=1: V1=V: CNT1= CNT: GOTO 900
820     A=A+J
830     V=V+A+J/2
840     P=P+V+A/2J/6
850     LOCATE 10,1: PRINT CNT,A,V,P"       "
860     GOTO 690
870     '
880     'DO THE SACL RAMP-SLOPE                  :{ Ref E part of profile curve}
890     '*******************************
900     LOCATE 11,1 : PRINT "J=0"
910     '
920     IF (CNT+1)>=(T/UPDTE-CNT1) THEN FLG1=2 : GOTO 1000
930     V=V+A
940     P=P+V+A/2
950     LOCATE 12,1: PRINT CNT,A,V,P"       "
960     GOTO 690
970     '
980     'DO THE S ACL RAMP-DN                    :{ Ref B part of profile curve}
990     '*******************************
1000    LOCATE 13,1 : PRINT "J=-J
1010    '
1020    IF (A-2*J<=0) THEN FLG1=3 : P1=TP*TESO-P : GOTO 1110
1030    A=A-J
1040    V=V+A-J/2
1050    P=P+V+A/2-J/6
1060    LOCATE 14,1: PRINT CNT,A,V,P"       "
1070    GOTO 690
1080    '
1090    'DO THE S SLEW                           :{ Ref G part of profile curve}
1100    '*******************************
1110    LOCATE 15,1 : PRINT "J=0
1120    '
1130    IF  P>=P1+V THEN FLG1=4 : A=0 : GOTO 1210
1140    A=0
1150    P=P+V                                    :'+A/2
1160    LOCATE 16,1: PRINT CNT,A,V,P"       "
1170    GOTO 690
1180    '
1190    'DO THE S DCL RAMP-DN                    :{ Ref C part of profile curve}
1200    '*******************************
1210    LOCATE 17,1 : PRINT "J=-J
1220    '
1230    IF ABS(A-J)>=(SACL*RESO*UPDTE*UPDTE) THEN FLG1=5: GOTO 1320
1240    A=A-J
1250    V=V+A-J/2
1260    P=P+V+A/2-J/6
1270    LOCATE 18,1: PRINT CNT,A,V,P"       "
1280    GOTO 690
1290    '
1300    'DO THE S DCL RAMP-SLOPE                 :{ Ref F part of profile curve}
1310    '*******************************
1320    LOCATE 19,1 : PRINT "J=0"
1330    '
1340    IF V+A<=V1 THEN FLG1=6: GOTO 1420
1350    V=V+A
1360    P=P+V-A/2
1370    LOCATE 20,1: PRINT CNT,A,V,P"       "
1380    GOTO 690
1390    '
1400    'DO THE DCL RAMP-UP                      :{ Ref D part of profile curve}
1410    '*******************************
1420    LOCATE 21,1     : PRINT "J=+J
1430    '
1440    IF ((A+J>=0) OR (V<=0)) GOTO 1560
1450    A=A+J
1460    V=V+A+J/2
1470    P=P+V+A/2+J/6
1480    LOCATE 22,1: PRINT CNT,A,V,P"       "
1490    IF ((FLG3<>0) OR (P<TP*RESO)) GOTO 1510
```



```
1500    LOCATE 23,1: PRINT CNT,A,V,P"       " : FLG3=1
1510    GOTO 690
1520    '
1530    'END OF ROUTINE
1540    '************************
1550    '
1560    LOCATE 23,60: PRINT "***DONE***DONE***"
1570    KY$=INKEY$
1580    IF KY$<>"" GOTO 1570
1590    KY$=INKEY$
1600    IF KY$="" GOTO 1570
1610    GOTO 40
```

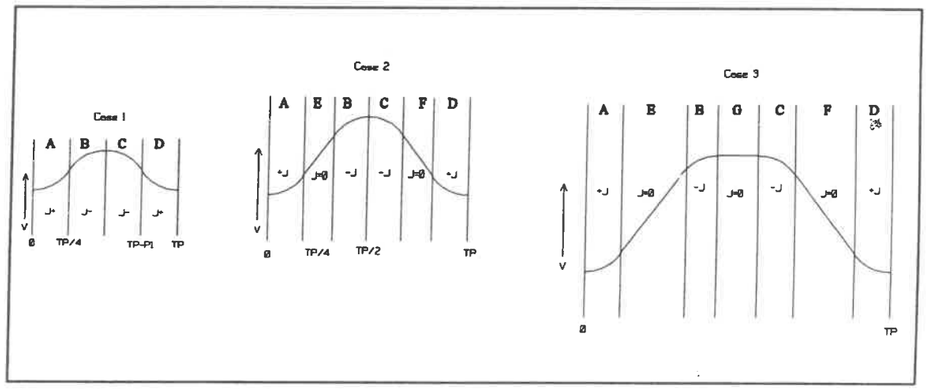

Figure 10.4 Relationship of Jerk to acceleration.

A Simple PID System Simulator

The following BASIC program simulates a simple LM628 *National Semiconductor* trajectory. This program demonstrates the interaction between the LM628, system resolution, and the system mechanics.

This program is by no means perfect. It has been simplified to demonstrate method rather than exactness although the solutions should be within 5 percent.

The PRINT statements and several of the GOTO statements have been slightly modified to maintain clarity for this source listing. It may be necessary to adjust the actual PRINT and GOTO statements to align the curves with the coordinates. However, this should be a relatively simple modification.

```
**********************`

10   'System Simulator ... 9/22/91
20   SCREEN 8: CLS: COLOR 15
30   '
40   'Set the screen pixel dimensions
50   '
60   CRNTOP = 11: CRNBOT = 115: CRNLFT = 44: CRNRGT = 555
70   '
120  COLOR 14: LOCATE 10, 27: PRINT "System Simulation Program"
130  LOCATE 11, 27: PRINT "==========================="
140  COLOR 12: LOCATE 12, 27: PRINT "Chuck Raskin P.E. App.Eng."
150  COLOR 15: LOCATE 17, 16
155  PRINT "Press <ESC> or <S> to graph or abort any function"
160  LOCATE 18, 25: PRINT "Press <F9> to quit the program"
170  LOCATE 20, 19: PRINT "Editing keys are ......"
180  LOCATE 22, 8
185  PRINT "<?><BS><INS><DEL><CR><#s><.> and <UP><DN><LF><RT>"
190  LOCATE 22, 10
195  PRINT "<S> = Stop the move  <A> = Abort the move (E'stop)"
200  '
210  A$ = INKEY$: IF A$ = "" GOTO 210
220  CLS : ROW = 3: COL = 26
240  '
250  'Set the defaults
260  '
270  ERSCALE$ = ".40"  : DIST$ = "1.0"    : UPDT$ = "0.000256"
280  MVEL$ = "500"     : KP$ = "50"       : JSYS$ = ".01"
330  KI$ = "20"        : MXVEL$ = "500"   : KD$ = "0"
360  IL$ = "500"       : KE$ = "30"             : DIFUPDTE$ = "3"
390  KT$ = "0.176"       : PORT$ = "12"         : LA$ = ".0022"
420  LPREV$ = "2000"   : RA$ = "6.0"    : RESO$ = ".0001"
450  VD$ = ".00027"    : ACCTME$ = "0.02"
470  '
480  IF VAL(MXVEL$) < VAL(MVEL$) THEN MXVEL$ = MVEL$
490  '
500  'Traj Variables - Screen Entry
510  '
515  'These print statements have been shortened for easy reading
516  'Re-space and add length as required for your PC
517  '
520  GOSUB 1130: COLOR 15
530  LOCATE 1, 7
535  PRINT " Error Scale (x) =                  <?> = Selection Help"
540  LOCATE 2, 7                                                    "
545  PRINT "
550  LOCATE 3, 7
555  PRINT " Profile move     =              Update Time       =    "
560  LOCATE 4, 7
565  PRINT " Profile Vel.     =              Kp (Gain)         =    "
570  LOCATE 5, 7
575  PRINT " Sys. Iner. (J)   =              Ki (Gain)         =    "
580  LOCATE 6, 7
585  PRINT " Max Sys Vel.     =              Kd (Gain)         =    "
590  LOCATE 7, 7
595  PRINT "                                 Il (Gain)         =    "
600  LOCATE 8, 7
605  PRINT " <<<<  Motor  >>>>               Kd Update Mult.   =    "
```



```
610  LOCATE 9, 7
615  PRINT " Ke  (V/Krpm)     =                                     "
620  LOCATE 10, 7
625  PRINT " Kt (InLb/Amp)    =              DAC Size (8/12)   =    "
630  LOCATE 11, 7
635  PRINT " La (H)           =              Enc. Lines/Revo.  =    "
640  LOCATE 12, 7
645  PRINT " Resis. (ohms)    =              Reso (In./Line)   =    "
650  LOCATE 13, 7
655  PRINT " Visc. Damping    =              Accl. Time (Sec.) =    "
660  LOCATE 14, 7
665  PRINT "                                                       "
670  '
680  'Get - Set - Check the numeric values
690  '
700  GOSUB 3360
710  '
720  'Initialize the values
730  '
740  VL1 = 0: PO1 = 0: DPO1 = 0: DFLG2 = 0: DFLG3 = 0
750  AFLG = 0: DFLG = 0: VSTOR = 0: SV1 = 0
760  APOS = 0: DPOS = 0: X = 0: OLDDAC = 0: POSI = 0: RPOS = 0
770  EROR = 0: R = 0: LASTERR = 0: CNT1 = 0: TME1 = 0: MRPM = 0
780  RPOS1 = 0: DACCNT = 0: DAC = 0: VEL = 0: ERRDIFF = 0: X2 = 0
790  DIFF = 0: PROP = 0: DIFTMEBASE = VAL(DIFUPDTE$): ABRTPOS = 0
810  MXVEL = INT(((MXVEL/60 * UPDATETME * 1 / RESO) + .5) * 2 ^ 16)
820  MVEL = INT(((MVEL / 60 * UPDATETME * 1 / RESO) + .5) * 2 ^ 16)
830  MXVEL = MXVEL / 2 ^ 16
840  MVEL  = MVEL / 2 ^ 16
845  '
850  DEFDBL Q, S                'Q=KI  S=INTEG
870  Q = KI: S = 0!
880  '
890  DIFTMEBASE = DIFUPDTE + 2    'INITIALIZE
900  '
910  ACL = INT(MXVEL / (ACCTME / UPDATETME) * 2 ^ 16)
920  ACL = ACL / 2 ^ 16
930  AC1 = ACL
940  '
950  IF PORT = 12 THEN DACCNT = 2 ^ 11
960  IF PORT = 8 THEN DACCNT = 2 ^ 7
970  ESTP = 10 / DACCNT
980  '
990  XSCALE = (CRNRGT - CRNLFT) / DIST
1000 IF SFLG = 0 GOTO 1020
1010 XSCALE =  (CRNRGT - CRNLFT) / (ACCTME / UPDATETME * 2 + DIST / RESO / VEL)
1020 YTSCALE = (CRNBOT - CRNTOP) / MVEL
1030 YRSCALE = (CRNBOT - CRNTOP) / (1 / UPDATETME * MVEL * 60 / LPREV)
1040 '
1050 GOSUB 1130
1060 GOSUB 1440
1070 GOTO 530
1080 '
1090 '*********************************************************
1100 'Graph the traj. pos. vs. real pos. vs. error
1120 '
1121 'These print statements have been shortened for easy reading
1122 'Re-space and add length as required for your PC
1123 '
1130 COLOR 11
1140 LOCATE 1, 4: PRINT "  |                                      "
1150 LOCATE 2, 4: PRINT "^ |                                      "
1160 LOCATE 3, 4: PRINT "| |                                      "
1170 LOCATE 4, 4: PRINT "  |                                      "
1180 LOCATE 5, 4: PRINT "V |                                      "
1190 LOCATE 6, 4: PRINT "e |                                      "
1200 LOCATE 7, 4: PRINT "l |                                      "
1210 LOCATE 8, 4: PRINT "o |                                      "
1220 LOCATE 9, 4: PRINT "c |                                      "
1230 LOCATE 10, 4:PRINT "i |                                      "
1240 LOCATE 11, 4:PRINT "t |                                      "
1250 LOCATE 12, 4:PRINT "y |                                      "
1260 LOCATE 13, 4:PRINT "  |                                      "
1270 LOCATE 14, 4:PRINT "  |                                      "
1280 LOCATE 15, 4:PRINT "--+--------------------------------------"
1290 IF SFLG = 1 GOTO 1310
1300 LOCATE 16, 8: PRINT "Position (Encoder Counts) -->" :GOTO 1320
1310 LOCATE 16, 8
1315 PRINT "Time in Updates ("; UPDATETME; "sec. ) -->           "
1320 COLOR 15
1330 LOCATE 17, 29: PRINT "UPDATES =                             "
1340 LOCATE 18, 4: PRINT "                                       "
```

```
1350 LOCATE 19, 7
1355 PRINT "Trajectory Position =          Proportional     ="
1360 LOCATE 20, 7
1365 PRINT " Absolute  Position =          Integral Sum     ="
1370 LOCATE 21, 7
1375 PRINT "  Following  Error  =          Differential     ="
1380 LOCATE 22, 7
1385 PRINT "  DAC output count  =          Estm'd Motor RPM ="
1390 LOCATE 23, 4                                           "
1395 PRINT "
1400 RETURN
1410 '
1420 '1st Update
1430 '
1440 AC1 = ACL: COLOR 15:X = X+1
1450 TME = UPDATETME * X
1460 '
1470 GOSUB 2450                  'DO THE TRAJ GEN CALCS
1480 '
1490 EROR = INT(POSI)            'ERROR = POSITION IN 1ST UPDATE
1500 GOTO 1620
1510 '
1520 'All other updates
1530 '
1540 LOCATE 17, 24: PRINT : X = X + 1
1550 TME = X * UPDATE
1560 '
1570 GOSUB 2450                          'DO THE TRAJ GEN CALCS
1580 GOSUB 3060                          'DO THE MECH POS CALCS
1590 '
1600 'Get the new DAC step output
1610 '
1620 GOSUB 2140                  'GET THE NEW DAC VALUE
1625 '
1630 A$ = "":A$ = INKEY$
1640 IF A$ = CHR$(27) GOTO 1970
1650 IF A$ = "s" OR A$ = "S" THEN DPOS = POSI: APOS = POSI    'STOP
1660 IF A$ = "a" OR A$ = "A" THEN DPOS = POSI: DIST = RPOS: POSI = RPOS 'abort
1670 '
1680 LOCATE 17, 23: PRINT X; " "
1690 IF DFLG3 = 1 OR SFLG = 0 THEN GOTO 1710
1700 LOCATE 17, 11: PRINT "("; X; ")"
1710 LOCATE 17, 39
1711 PRINT INT(UPDATETME * X * 1000000!) / 1000000!; "  Sec.    "
1720 LOCATE 19, 29: PRINT INT(POSI)
1730 COLOR 15
1740 IF SFLG = 1 GOTO 1760
1750 PSET (CRNLFT + POSI * XSCALE, CRNBOT - ABS(VEL * YTSCALE))
1755 GOTO 1780
1760 PSET (CRNLFT+UPDATETME*X*XSCALE , CRNBOT-ABS(VEL*YTSCALE))
1770 PSET (CRNLFT+UPDATETME*X*XSCALE , CRNBOT-ABS(VL1*YTSCALE))
1780 COLOR 15
1790 IF ((POSI >= DIST) AND VEL < ACL) OR VEL < 0 GOTO 1910
1800 IF (ACL < 0) GOTO 1860
1810 IF (POSI > DPOS) THEN ACL = -ABS(ACL)
1820 IF (POSI > DPOS) AND (DPOS <> 0) GOTO 1840
1830 ACL = ABS(ACL)
1840 GOTO 1540
1850 '
1860 IF (POSI > DPOS) THEN ACL = -ABS(ACL)
1870 IF (POSI > DPOS) AND (DPOS <> 0) GOTO 1840
1880 IF (POSI>DPOS) AND (DPOS<>0) AND (POSI>=DIST) AND (VEL<ACL) GOTO 1910
1890 GOTO 1540
1900 '
1910 IF CNT1 > 50 GOTO 2040
1920 IF (RPOS > POSI - 1) AND (RPOS < POSI + 1) THEN CNT1 = CNT1 + 1
1930 IF (RPOS < POSI - 1) OR (RPOS > POSI + 1) THEN CNT1 = 0
1940 X = X + 1
1950 GOTO 1580
1960 '
1970 COLOR 14
1980 LOCATE 1, 17
1985 PRINT "********** Aborted by Operator **********"
1990 COLOR 15: LOCATE 17,23: PRINT X
2000 LOCATE 17, 39
2005 PRINT INT(X * UPDATETME * 1000000!) / 1000000!; "   Sec.    "
2010 A$ = INKEY$:IF A$ = "" GOTO 2010
2020 RETURN
2030 '
2040 COLOR 14
2050 LOCATE 1, 17
2055 PRINT "********** Trajectory Complete **********"
```



```
2060 COLOR 15: LOCATE 17,23: PRINT X - 50
2070 LOCATE 17, 39
2075 PRINT INT((X - 50) * UPDATETME * 1000000!) / 1000000!; " Sec."
2080 A$ = INKEY$: IF A$ = "" GOTO 2080
2090 IF A$ = "?" THEN GOSUB 6060: GOTO 2060
2100 RETURN
2110 '
2120 'Error Count Generator
2130 '
2140 EROR = INT(POSI - RPOS)
2150 '
2160 PROP = KP * EROR
2170 S = S + (Q * EROR)
2180 IF S > IL THEN S = IL
2190 IF S < -IL THEN S = -IL
2200 '
2210 DIFTMEBASE = DIFTMEBASE + 1
2220 IF DIFTMEBASE < DIFUPDTE GOTO 2300
2230 IF EROR >= 0 AND LASTERR >= 0 THEN ERRDIFF = EROR - LASTERR: GOTO 2260
2240 IF EROR < 0 AND LASTERR < 0 THEN ERRDIFF = -(ABS(EROR) - ABS(LASTERR)): GOTO 2260
2250 ERRDIFF = EROR - LASTERR
2260 DIFF = KD * ERRDIFF
2270 DIFTMEBASE = 0
2280 LASTERR = EROR
2290 '
2300 OLDDAC = DAC: DAC = INT((PROP + S + DIFF) / 16)
2310 COLOR 12: LOCATE 21, 29: PRINT -EROR; "    "
2320 IF SFLG = 1 GOTO 2340
2330 PSET (CRNLFT+RPOS*XSCALE, (CRNBOT-EROR/ERSCALE*YRSCALE))
2335 GOTO 2350
2340 PSET (CRNLFT+UPDATETME*X*XSCALE, (CRNBOT-EROR/ERSCALE*YRSCALE))
2350 COLOR 15: LOCATE 19, 61: PRINT PROP; "    "
2360 LOCATE 20, 61: PRINT S; "    "
2370 LOCATE 21, 61: PRINT DIFF; "    "
2380 IF DAC > 2048 THEN DAC = 2048
2390 IF DAC < -2048 THEN DAC = -2048
2400 LOCATE 22, 29: PRINT DAC; "    ": RETURN
2420 '
2430 'Trajectory Generator
2440 '
2450 IF DFLG3 = 1 GOTO 2560
2460 IF DFLG2 = 1 AND VL1 < .2 THEN DFLG3 = 1: GOTO 2560
2470 VL1 = VL1 + AC1
2480 IF VL1 > MVEL THEN VL1 = MVEL
2490 PO1 = PO1 + VL1
2500 IF ((PO1 >= DIST/2) AND PO1<>0) OR ((PO1>DPO1) AND DPO1<>0) THEN AC1 = -ABS(AC1)
2510 IF DFLG2 = 1 GOTO 2560
2520 IF VL1 = MVEL THEN DPO1 = DIST - PO1
2530 IF PO1 >= DIST / 2 AND PO1 <> 0 THEN DPO1 = DIST - PO1
2540 IF DPO1 <> 0 THEN DFLG2 = 1
2550 '
2560 IF SFLG = 1 GOTO 2670
2570 VEL = VEL + ACL
2580 IF VEL >= MVEL THEN VEL = MVEL
2590 POSI = POSI + VEL
2600 IF (APOS <> 0) OR (ACL < 0) OR (DPOS <> 0) GOTO 2650
2610 IF (VEL >= MVEL) THEN APOS = POSI
2620 IF (APOS = 0) AND (POSI >= DIST / 2) THEN APOS = POSI
2630 IF (APOS <> 0) THEN DPOS = DIST - APOS
2640 IF (APOS = 0) AND (POSI >= DIST / 2) THEN DPOS = DIST - APOS
2650 RETURN
2660 '
2670 IF AFLG = 1 GOTO 2860
2680 VEL = VEL + ACL
2690 IF VEL >= SVEL THEN VEL = SVEL
2700 POSI = POSI + VEL
2710 IF (APOS <> 0) OR (ACL < 0) OR (DPOS <> 0) GOTO 2770
2720 IF (VEL >= MVEL) THEN APOS = POSI
2730 IF (APOS = 0) AND (POSI >= DIST / 2) THEN APOS = POSI
2740 IF (APOS <> 0) THEN DPOS = DIST - APOS: AFLG = 1: X6 = X
2750 IF (APOS = 0) AND (POSI >= DIST/2) THEN DPOS = DIST - APOS:AFLG = 1:DFLG = 1:X6 = X
2760 '
2770 SV1 = SV1 + 1
2780 IF SV1 < 8 GOTO 2840
2790 SVEL = ACL + (X * ACL * 100 * UPDATETME) ^ 3
2800 IF SVEL >= MVEL THEN SVEL = MVEL
2810 IF ((VEL - VSTOR) / (8 * UPDATETME)) * 2 >= (ACL / UPDATETME) GOTO 2820 VSTOR = VEL
2830 SV1 = 0
2840 RETURN
2850 '
2860 IF DFLG = 1 GOTO 2940
2870 VEL = VEL + ACL
```

```
2880 IF VEL >= MVEL THEN VEL = MVEL
2890 POSI = POSI + VEL
2900 IF POSI >= DIST / 2 AND APOS = 0 THEN DFLG = 1: SVEL = MVEL:SV1 = 0: X6 = X + X6
2910 IF POSI >= DPOS * .997 THEN DFLG = 1: SVEL = MVEL: SV1 = 0: X6 = X + X6
2920 RETURN
2930 '
2940 SV1 = SV1 + 1
2950 IF SV1 < 8 GOTO 3000
2960 SVEL = ((X6 - X) * ABS(ACL) * 100 * UPDATETME) ^ 3
2970 IF SVEL >= MVEL THEN SVEL = MVEL
2980 SV1 = 0
2990 '
3000 IF VEL >= SVEL THEN VEL = VEL - ABS(ACL)
3010 POSI = POSI + VEL
3020 RETURN
3030 '
3040 'Mechanical Position Calculations
3050 '
3051 'Reference Chapter 9, page 95 for mechanical time constant calculation
3052 '
3060 MTRETC = LA / RA
3070 MTRMTC = JSYS * RA / (KE * KT)
3080 '
3090 IF OLDDAC <> DAC THEN X2 = 0
3100 X2 = X2 + 1
3110 '
3120 TME1 = X2 * UPDATETME
3130 '
3140 EDLY = EXP(-TME1 / MTRETC)
3150 MDLY = EXP(-TME1 / MTRMTC)
3160 '
3170 RPOS1 = (ABS(DAC - OLDDAC) * (1 - EDLY) * (1 - MDLY))
3180 '
3190 IF DAC -OLDDAC < 0 THEN MRPM = MRPM -INT(1/UPDATETME * RPOS1 * 60/LPREV * 1000)/1000
3200  GOTO 3220
3210 MRPM = MRPM + INT(1 / UPDATETME * RPOS1 * 60 / LPREV * 1000)/ 1000
3220 RPOS = RPOS + INT(MRPM / 60 * UPDATETME * LPREV * 1000) / 1000
3230 COLOR 10
3240 LOCATE 20, 29: PRINT INT(RPOS); " "
3250 '
3260 IF SFLG = 1 GOTO 3290
3270 PSET (CRNLFT+(RPOS*XSCALE), CRNBOT-(MRPM * YRSCALE))
3280 GOTO 3300
3290 PSET (CRNLFT+(UPDATETME*X*XSCALE), CRNBOT-(MRPM * YRSCALE))
3300 LOCATE 22, 61: PRINT MRPM; "     "
3310 COLOR 15
3320 RETURN
3330 '
3340 'PRINT THE VARIABLES - Get The new Parameters
3350 '
3360 LOCATE 1, 26: PRINT ERSCALE$
3380 LOCATE 3, 26: PRINT DIST$
3390 LOCATE 3, 57: PRINT UPDT$
3410 LOCATE 4, 26: PRINT MVEL$
3420 LOCATE 4, 57: PRINT KP$
3440 LOCATE 5, 26: PRINT JSYS$
3450 LOCATE 5, 57: PRINT KI$
3470 LOCATE 6, 26: PRINT MXVEL$
3480 LOCATE 6, 57: PRINT KD$
3500 LOCATE 7, 57: PRINT IL$
3520 LOCATE 8, 57: PRINT DIFUPDTE$
3540 LOCATE 9, 26: PRINT KE$
3560 LOCATE 10, 26: PRINT KT$
3570 LOCATE 10, 57: PRINT PORT$
3590 LOCATE 11, 26: PRINT LA$
3600 LOCATE 11, 57: PRINT LPREV$
3620 LOCATE 12, 26: PRINT RA$
3630 LOCATE 12, 57: PRINT RESO$
3650 LOCATE 13, 26: PRINT VD$
3660 LOCATE 13, 57: PRINT ACCTME$
3680 A$ = ""
3690 GOSUB 4580
3700 '
3710 'Convert the Entries to Numeric values
3730  '
3740 GOSUB 6190
3750 '
3760 'Check all of the entries
3780 '
3790 IF DIST <= 0 OR DIST > 1500 THEN ROW = 3: COL = 26: GOTO 3380
3800 DIST = DIST / RESO
3810 IF MVEL <= 0 OR MVEL > 1500 THEN ROW = 4: COL = 26: GOTO 3380
```



```
3820 IF JSYS <= 0 OR JSYS > 1000 THEN ROW = 5: COL = 26:  GOTO 3380
3830 IF MXVEL < MVEL THEN MXVEL = MVEL
3840 IF MXVEL = MVEL THEN MXVEL$ = MVEL$
3850 IF MXVEL > 1500 THEN ROW = 7: COL = 26: GOTO 3380
3880 IF KE <= 0 OR KE > 1000 THEN ROW = 9: COL = 26: GOTO 3380
3890 IF KT <= 0 OR KT > 1000 THEN ROW = 10: COL = 26: GOTO 3380
3900 IF LA <= 0 OR LA > 1000 THEN ROW = 11: COL = 26: GOTO 3380
3910 IF RA <= 0 OR RA > 1000 THEN ROW = 12: COL = 26: GOTO 3380
3920 IF VD <= 0 OR VD > 1000 THEN ROW = 13: COL = 26: GOTO 3380
3930 '
3940 IF UPDATETME <= 0 OR UPDATETME > 1 THEN ROW = 1: COL = 57: GOTO 3380
3950 IF KP <= 0 OR KP > 32000 THEN ROW = 4: COL = 57: GOTO 3380
3960 IF KI < 0 OR KI > 32000 THEN ROW = 5: COL = 57: GOTO 3380
3970 IF KD < 0 OR KD > 32000 THEN ROW = 6: COL = 57: GOTO 3380
3980 IF IL < 0 OR IL > 32000 THEN ROW = 7: COL = 57: GOTO 3380
3990 IF DIFUPDTE <= 0 OR DIFUPDATE > 65000! THEN ROW = 8: COL = 57: GOTO 3380
4000 IF (PORT <> 8) AND (PORT <> 12) THEN ROW = 10: COL = 57: GOTO 3380
4010 IF LPREV <= 0 OR LPREV > 60000! THEN ROW = 11: COL = 57: GOTO 3380
4020 IF RESO <= 0 OR RESO > 2 THEN ROW = 12: COL = 57: GOTO 3380
4030 IF ACCTME <= 0 OR ACCTME > 600 THEN ROW = 13: COL = 57: GOTO 3380
4040 IF ERSCALE < .01 OR ERSCALE > 1000 THEN ROW = 1: COL = 26: GOTO 3380
4050 RETURN
4060 '
4070 'Rewrite the Data in the proper color
4090 '
4100 COLOR 12: LOCATE ROW, COL: PRINT TMP$       'NEW ENTRY
4110 A = LEN(TMP$)
4120 FOR B = COL + A TO COL + 9
4130 LOCATE ROW, B: PRINT " "
4140 NEXT B
4150 CNTFLG = 0
4160 RETURN
4170 '
4180 COLOR 15: LOCATE ROW, COL: PRINT TMP$       '<ESC>
4190 A = LEN(TMP$)
4200 FOR B = COL + A TO COL + 9
4210 LOCATE ROW, B: PRINT " "
4220 NEXT B
4230 RETURN
4240 '
4250 COLOR 12: LOCATE ROW, COL            'NUMBER OR DEC PT ENTRY
4260 IF (CNTFLG + 1 > LEN(TMP$)) GOTO 4280
4270 MID$(TMP$, CNTFLG + 1, 1) = A$: GOTO 4290
4280 IF (CNTFLG + 1 > A) THEN TMP$ = TMP$ + A$
4290 PRINT TMP$
4300 CNTFLG = CNTFLG + 1
4310 RETURN
4320 '
4330 IF LEN(TMP$) <= 1 GOTO 4410                 '<DELETE>
4340 IF CNTFLG + 1 >= LEN(TMP$) GOTO 4430
4350 TMP$ = LEFT$(TMP$,CNTFLG) + RIGHT$(TMP$,LEN(TMP$) - CNTFLG-1)
4360 LOCATE ROW, COL: PRINT TMP$
4370 A = LEN(TMP$)
4380 FOR B = COL + A TO COL + 9
4390 LOCATE ROW, B: PRINT " "
4400 NEXT B
4410 RETURN
4420 '
4430 TMP$ = LEFT$(TMP$, CNTFLG): GOTO 4360
4450 IF LEN(TMP$) = 9 GOTO 4520                  '<INSERT>
4460 TMP$ = LEFT$(TMP$,CNTFLG)+"0"+RIGHT$(TMP$,LEN(TMP$) - CNTFLG)
4470 LOCATE ROW, COL: PRINT TMP$
4480 A = LEN(TMP$)
4490 FOR B = COL + A TO COL + 9
4500 LOCATE ROW, B: PRINT " "
4510 NEXT B
4520 RETURN
4530 '
4540 'Get the In/Out Data
4570 '
4580 LOCATE ROW, COL, 1: IF A$ = CHR$(27) GOTO 4790  'ESC
4590 IF ROW = 1 AND COL = 26 GOTO 4810         'GO TO INP SELECTION
4600 IF ROW = 3 AND COL = 26 GOTO 4820
4610 IF ROW = 3 AND COL = 57 GOTO 4830
4620 IF ROW = 4 AND COL = 26 GOTO 4840
4630 IF ROW = 4 AND COL = 57 GOTO 4850
4640 IF ROW = 5 AND COL = 26 GOTO 4860
4650 IF ROW = 5 AND COL = 57 GOTO 4870
4660 IF ROW = 6 AND COL = 26 GOTO 4880
4670 IF ROW = 6 AND COL = 57 GOTO 4890
4680 IF ROW = 7 AND COL = 57 GOTO 4900
4690 IF ROW = 8 AND COL = 57 GOTO 4910
```

```
4700 IF ROW = 9 AND COL = 26 GOTO 4920
4710 IF ROW = 10 AND COL = 26 GOTO 4930
4720 IF ROW = 10 AND COL = 57 GOTO 4940
4730 IF ROW = 11 AND COL = 26 GOTO 4950
4740 IF ROW = 11 AND COL = 57 GOTO 4960
4750 IF ROW = 12 AND COL = 26 GOTO 4970
4760 IF ROW = 12 AND COL = 57 GOTO 4980
4770 IF ROW = 13 AND COL = 26 GOTO 4990
4780 IF ROW = 13 AND COL = 57 GOTO 5000
4790 LOCATE ROW, COL, 0: RETURN
4800 '
4810 ROW = 1: COL = 26: TMP$ = ERSCALE$: GOSUB 4100: GOSUB 5050: ERSCALE$ = TMP$: GOTO 4580
4820 ROW = 3: COL = 26: TMP$ = DIST$: GOSUB 4100: GOSUB 5050:DIST$ = TMP$: GOTO 4580
4830 ROW = 3: COL = 57: TMP$ = UPDT$: GOSUB 4100: GOSUB 5050:UPDT$ = TMP$: GOTO 4580
4840 ROW = 4: COL = 26: TMP$ = MVEL$: GOSUB 4100: GOSUB 5050:MVEL$ = TMP$: GOTO 4580
4850 ROW = 4: COL = 57: TMP$ = KP$: GOSUB 4100: GOSUB 5050:KP$ = TMP$: GOTO 4580
4860 ROW = 5: COL = 26: TMP$ = JSYS$: GOSUB 4100: GOSUB 5050:JSYS$ = TMP$: GOTO 4580
4870 ROW = 5: COL = 57: TMP$ = KI$: GOSUB 4100: GOSUB 5050:KI$ = TMP$: GOTO 4580
4880 ROW = 6: COL = 26: TMP$ = MXVEL$: GOSUB 4100: GOSUB 5050:MXVEL$ = TMP$: GOTO 4580
4890 ROW = 6: COL = 57: TMP$ = KD$: GOSUB 4100: GOSUB 5050:KD$ = TMP$: GOTO 4580
4900 ROW = 7: COL = 57: TMP$ = IL$: GOSUB 4100: GOSUB 5050:IL$ = TMP$: GOTO 4580
4910 ROW = 8: COL = 57: TMP$ = DIFUPDTE$: GOSUB 4100: GOSUB 5050:DIFUPDTE$ = TMP$: GOTO 4580
4920 ROW = 9: COL = 26: TMP$ = KE$: GOSUB 4100: GOSUB 5050:KE$ = TMP$: GOTO 4580
4930 ROW = 10: COL = 26: TMP$ = KT$: GOSUB 4100: GOSUB 5050:KT$ = TMP$: GOTO 4580
4940 ROW = 10: COL = 57: TMP$ = PORT$: GOSUB 4100: GOSUB 5050:PORT$ = TMP$: GOTO 4580
4950 ROW = 11: COL = 26: TMP$ = LA$: GOSUB 4100: GOSUB 5050:LA$ = TMP$: GOTO 4580
4960 ROW = 11: COL = 57: TMP$ = LPREV$: GOSUB 4100: GOSUB 5050:LPREV$ = TMP$: GOTO 4580
4970 ROW = 12: COL = 26: TMP$ = RA$: GOSUB 4100: GOSUB 5050:RA$ = TMP$: GOTO 4580
4980 ROW = 12: COL = 57: TMP$ = RESO$: GOSUB 4100: GOSUB 5050:RESO$ = TMP$: GOTO 4580
4990 ROW = 13: COL = 26: TMP$ = VD$: GOSUB 4100: GOSUB 5050:VD$ = TMP$: GOTO 4580
5000 ROW = 13: COL = 57: TMP$ = ACCTME$: GOSUB 4100: GOSUB 5050:ACCTME$ = TMP$: GOTO 4580
5010 RETURN
5020 '
5030 GOSUB 4100
5040 '
5050 A$ = "": A$ = INKEY$: IF A$ = "" GOTO 5050          'KEY ?
5060 IF A$ = CHR$(27) THEN GOSUB 4180:  SFLG = 0: GOTO 5240  'ESC
5070 IF A$ = "s" OR A$ = "S" THEN GOSUB 4180: GOTO 5260
                                                          'EXIT
5080 IF A$ = CHR$(0) + "C" THEN SYSTEM
5090 IF A$ = CHR$(0) + CHR$(82) THEN GOSUB 4450: GOTO 5050 'INS
5100 IF A$ = CHR$(0) + CHR$(83) THEN GOSUB 4330: GOTO 5050 'DEL
                                                          'HELP
5110 IF A$ = "?" THEN GOSUB 5610: GOTO 5050
5120 IF A$ = CHR$(13) THEN GOSUB 4180: ROW = ROW + 1:GOTO 5370      '<CR> THEN SET TO DN
5130 IF A$ = CHR$(0) + CHR$(72) THEN GOSUB 4180: ROW = ROW - 1:GOTO 5290'UP - ERR SCALE
5140 IF A$ = CHR$(0) + CHR$(80) THEN GOSUB 4180: ROW = ROW + 1:GOTO 5400'DN - MVEL
5150 IF A$ = CHR$(0) + CHR$(75) THEN GOSUB 4180: GOTO 5490 'LF - KP
5160 IF A$ = CHR$(0) + CHR$(77) THEN GOSUB 4180: GOTO 5550 'RT - KP
5170 IF (A$ = "0") OR (A$ = "1") OR (A$ = "2") OR (A$ = "3") OR (A$ = "4") THEN GOSUB 4250
5180 IF (A$ = "5") OR (A$ = "6") OR (A$ = "7") OR (A$ = "8") OR (A$ = "9") THEN GOSUB 4250
5190 IF A$ = "." THEN GOSUB 4250
5200 IF CNTFLG > 8 THEN CNTFLG = 0
5210 IF A$ = CHR$(8) AND LEN(TMP$) > 1 THEN GOSUB 4330:CNTFLG = CNTFLG - 1
5220 IF CNTFLG < 0 THEN CNTFLG = 0
                                                          'TRY AGAIN
5230 GOTO 5050
5240 RETURN
5250 '
5260 SFLG = 1: A$ = CHR$(27): AFLG = 0: DFLG = 0: VSTOR = 0
5270 SVEL = ACL: GOTO 5240  'S CURVE
5280 '
5290 'Up key
5300 '
5310 IF ROW = 2 AND COL = 57 THEN ROW = 13
5320 IF ROW = 9 AND COL = 57 THEN ROW = 8
5330 IF ROW = 0 AND COL = 26 THEN ROW = 13
5340 IF ROW = 2 AND COL = 26 THEN ROW = 1
5350 IF ROW = 8 AND COL = 26 THEN ROW = 6
5360 RETURN
5370 '
5380 'Dn Key
5390 '
5400 IF ROW = 14 AND COL = 57 THEN ROW = 3
5410 IF ROW = 9 AND COL = 57 THEN ROW = 10
5420 IF ROW = 2 AND COL = 26 THEN ROW = 3
5430 IF ROW = 7 AND COL = 26 THEN ROW = 9
5440 IF ROW = 14 AND COL = 26 THEN ROW = 1
5450 RETURN
5460 '
5470 'Left Key
5480 '
5490 IF ROW = 2 OR ROW = 7 OR ROW = 8 GOTO 5030
5500 COL = 26 : RETURN
5520 '
```



```
5530 'Right Key
5540 '
5550 IF ROW = 1 OR ROW = 2 OR ROW = 9 GOTO 5030
5560 COL = 57 : RETURN
5580 '
5590 'Help Screen Selections
5600 '
5610 COLOR 14: LOCATE 17, 1
5620 '
5630 IF ROW = 1 AND COL = 26 GOTO 5860  'GO TO INP SELECTION
5640 IF ROW = 3 AND COL = 26 GOTO 5870
5650 IF ROW = 3 AND COL = 57 GOTO 5880
5660 IF ROW = 4 AND COL = 26 GOTO 5890
5670 IF ROW = 4 AND COL = 57 GOTO 5900
5680 IF ROW = 5 AND COL = 26 GOTO 5910
5690 IF ROW = 5 AND COL = 57 GOTO 5920
5700 IF ROW = 6 AND COL = 26 GOTO 5930
5710 IF ROW = 6 AND COL = 57 GOTO 5940
5720 IF ROW = 7 AND COL = 57 GOTO 5950
5730 IF ROW = 8 AND COL = 57 GOTO 5960
5740 IF ROW = 9 AND COL = 26 GOTO 5970
5750 IF ROW = 10 AND COL = 26 GOTO 5980
5760 IF ROW = 10 AND COL = 57 GOTO 5990
5770 IF ROW = 11 AND COL = 26 GOTO 6000
5780 IF ROW = 11 AND COL = 57 GOTO 6010
5790 IF ROW = 12 AND COL = 26 GOTO 6020
5800 IF ROW = 12 AND COL = 57 GOTO 6030
5810 IF ROW = 13 AND COL = 26 GOTO 6040
5820 IF ROW = 13 AND COL = 57 GOTO 6050
5830 '
5860 PRINT "This value varies the height of the following error curve (0.01 - 1000) "
5865  GOTO 6090
5870 PRINT "Enter the profile move Distance in inches (1 TO 1500)":GOTO 6090
5880 PRINT "Enter the Processor motion update time in seconds <= 1:GOTO 6090
5890 PRINT "Enter the desired axis Move Velocity in inches/minute (1 TO 1500)": GOTO 6090
5900 PRINT "Enter the Proportional gain factor (Kp) <= 32000":GOTO 6090
5910 PRINT "Enter the total System Inertia (In.Lb.Sec.^2) <= 1000.0: GOTO 6090
5920 PRINT "Enter the Integral gain factor (Ki<=32000)": GOTO 6090
5930 PRINT "Enter the Maximum axis velocity in inches/minute (1 TO 1500)      ": GOTO 6090
5940 PRINT "Enter the Diff. gain factor (Kd) <= 32000 ": GOTO 6090
5950 PRINT "Enter the Integ Sum Limit value (Il<=32000": GOTO 6090
5960 PRINT "(1 to 65000) times the Update Time = Kd update time period       ": GOTO 6090
5970 PRINT "Enter the motor Kv (V/KRpm) <= 1000     ": GOTO 6090
5980 PRINT "Enter the motor Torque Constant (In.Lb./Amp.) <= 1000 : GOTO 6090
5990 PRINT "Enter the DAC size - 8 or 12 bit     ": GOTO 6090
6000 PRINT "Enter the motor inductance (Henrys) <= 1000": GOTO 6090
6010 PRINT "Enter the number of encoder lines/revolution of the motor <= 60000    ":GOTO 6090
6020 PRINT "Enter the motor resistance (Ohms) <= 1000 ": GOTO 6090
6030 PRINT "Enter the inch value of one encoder line <= 2          ": GOTO 6090
6040 PRINT "Enter the motor Viscous Damping Torque (In.Lb/Krpm/Sec.) <= 1000   ": GOTO 6090
6050 PRINT "Enter the move profile Acceleration Time (Seconds<=600): GOTO 6090
6060 COLOR 14: LOCATE 17, 1
6070 PRINT "Traj. Comp. is at 50 consecutive updates of +/- 1 Fol. Err. count    ": GOTO 6090
6080 '
6090 A$ = INKEY$: IF A$ = "" GOTO 6090           'KEY ?
6100 COLOR 15: LOCATE 17, 30 : PRINT "UPDATES =                "
6120 LOCATE 17, 23: PRINT X : LOCATE 17, 39
6135 PRINT INT(UPDATETME * X * 1000000!) / 1000000!; "  Sec.    "
6140 COLOR 12: A$ = "": RETURN
6150 '
6160 'Convert the Entries to Numeric Values
6180 '
6190 DIST = VAL(DIST$) : UPDATETME = VAL(UPDT$) : MVEL = VAL(MVEL$)
6220 KP = VAL(KP$) : JSYS = VAL(JSYS$) : MXVEL = VAL(MXVEL$)
6250 KI = VAL(KI$) : KD = VAL(KD$) : IL = VAL(IL$)
6280 KE = VAL(KE$) : DIFUPDTE = VAL(DIFUPDTE$) : KT = VAL(KT$)
6310 PORT = VAL(PORT$) : LA = VAL(LA$) : LPREV = VAL(LPREV$)
6340 RA = VAL(RA$) : RESO = VAL(RESO$) : VD = VAL(VD$)
6370 ACCTME = VAL(ACCTME$) : ERSCALE = VAL(ERSCALE$) : RETURN
```