



# LUND UNIVERSITY

## A PC Implementation of a PID Regulator

Åström, Karl; Åström, Karl Johan

1983

*Document Version:*

Publisher's PDF, also known as Version of record

[Link to publication](#)

*Citation for published version (APA):*

Åström, K., & Åström, K. J. (1983). *A PC Implementation of a PID Regulator*. (Technical Reports TFRT-7259). Department of Automatic Control, Lund Institute of Technology (LTH).

*Total number of authors:*

2

### General rights

Unless other specific re-use rights are stated the following general rights apply:

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

Read more about Creative commons licenses: <https://creativecommons.org/licenses/>

### Take down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

LUND UNIVERSITY

PO Box 117  
221 00 Lund  
+46 46-222 00 00

CODEN: LUTFD2 (TFRT-7259) / 1-15/1983

A PC IMPLEMENTATION OF A PID REGULATOR

KALLE ÅSTRÖM AND KARL JOHAN ÅSTRÖM

DEPARTMENT OF AUTOMATIC CONTROL  
LUND INSTITUTE OF TECHNOLOGY

SEPTEMBER 1983

<b>LUND INSTITUTE OF TECHNOLOGY</b> DEPARTMENT OF AUTOMATIC CONTROL Box 725 S 220 07 Lund 7 Sweden		Document name	
		Report	
		Date of issue	
		September 1983	
Author(s)  Kalle Åström and Karl Johan Åström		Document number	
		CODEN: LUTFD2 / (TFRT-7259) / 1-015 / (1983)	
Title and subtitle  A PC implementation of a PID regulator.		Supervisor	
		Sponsoring organization	
Abstract  This report describes an implementation of a PID regulator with graphics display and operator interface on an Apple II personal computer. The program is intended for laboratory experiments with PID control.			
Key words			
Classification system and/or index terms (if any)			
Supplementary bibliographical information			
ISSN and key title		ISBN	
Language	Number of pages	Recipient's notes	
English	15 + appendices		
Security classification			

DOKUMENTATABLAD RT 3/81

Distribution: The report may be ordered from the Department of Automatic Control or borrowed through the University Library 2, Box 1010, S-221 03 Lund, Sweden, Telex: 33248 Lubbis Lund.

## A PC IMPLEMENTATION OF A PID REGULATOR

Kalle Åström and Karl Johan Åström

### Abstract

This report describes an implementation of a PID regulator with graphic display and operator interface based on an Apple II personal computer. The program is intended for laboratory experiments with PID control.

### Contents

1. INTRODUCTION
2. OVERVIEW
3. THE CONTROL ALGORITHM
4. PROGRAM STRUCTURE
5. SUBROUTINES
6. VARIABLES
7. REFERENCES

Appendix A - Program listing  
Appendix B - Cross reference table

## 1. INTRODUCTION

This report describes an implementation of a PID regulator on a personal computer the Apple II [1], [2]. The regulator is designed primarily for educational purposes. Its main use is as a laboratory tool. The PC serves two main functions: as a regulator and as a recorder. Curves representing the set point, the measured variable and the control variable are displayed using the bit mapped graphics. The graphics and the game paddles are also used extensively for the man-machine interaction. The regulator is implemented in Basic.

The report is organized as follows. An overview of the program is given in Chapter 2. The control algorithm is described in Chapter 3. Chapter 4 describes the program structure. Details about the program are given in Chapter 5. The variables are described in Chapter 6. A program listing and a cross-reference table are given in appendices.

## 2. OVERVIEW

The program runs on an Apple II or an Apple IIe with a Mountain Hardware A-D, D-A card [3] and a UTIM timer [4]. It allows manual control and PID control using analog input and output. The important signals i.e. the set point, the measured output and the control variable are displayed in high resolution graphics. It is also possible to make a hard copy of the curves and to store the signals on disc. To make a hard copy it is necessary to have an Epson printer [5] with a Microbuffer [6] parallel interface. The program is written in Basic [2]. There is also a compiled version.

### How to run the program?

To run the program put in the program disc and type:

```
RUN PID
```

for the ordinary Applesoft Basic program or

```
RUN PIDTURBO
```

for the compiled version.

### The main functions

The program which is designed to be self-explanatory is menu controlled. The following options are available

```
configuration           C
parameter changes      P
manual control          M
automatic pid control  A
hardcopy of screen     H
store input and output signals in file S
```

The program goes to menu mode when it is started. It returns to menu mode

when a key is pressed. A copy of the screen for menu selection is shown in Fig. 1.

```

*****
*      PID REGULATOR          *****
*      VERSION I UNIPOLAR SIGNALS  *
*      83-07-04                 *
*      KALLE & KJ ASTRÖM        *
*****
ALTER CONFIGURATION (C)
ALTER PARAMETERS (P)
MANUAL (M)
AUTOMATIC (A)
HARD COPY (H)
STORE (S)
QUIT (Q)

SAMPLING PERIOD = .5 S
K = 5           N = 2
TI = 5         TD = 2.5
TIME UNIT IS 2 S

```

Figure 1 The screen image for menu selection.

Operator interactions can also be made via the game paddles when the program is running in manual or automatic mode. The reference value can be adjusted by paddle 0. The control variable is increased or decreased in manual mode using paddles 5 (decrease) and paddle 6 (increase). The "apple-buttons" on the Apple IIe may also be used [1].

The different functions are described below:

The slot positions of the clock card, the AD/DA card, and the parallel printer card are specified in the configuration mode. The channel numbers for the input and output signals can also be specified in this mode.

The parameters of the PID regulator may be changed in the parameter mode. The following parameters are available:

```

regulator gain      K
integration time    Ti
derivation time     Td
max derivative gain N
low output limit    ulow
high output limit   uhigh
sampling period     h
plotting interval  np

```

The time unit is seconds. Notice that the sampling period must be larger than 0.5 s in the normal Applesoft Basic version and greater than 0.1 s in the compiled version. The variables K, Ti, Td, N, ulow, uhigh and h are explained in detail in Chapter 3. The output level is normalized so that one unit corresponds to full output. The variable np controls the plotting. Every np:th sampling interval is plotted.

The control variable can be manipulated using the apple keys or game buttons 0 and 1 as increase-decrease buttons in the manual mode. Ordinary PID control is executed in the automatic mode.

The set point, the measured variable and the control variable are displayed on the screen when the program is in manual or automat control. A copy of the screen is shown in Fig. 2. A hard copy of the graph on the screen can be obtained using the hard copy mode. There is an option to get either a normal graph as shown below or an enlarged graph. The choice is menu selected.

A record of the setpoint yr, the process output y and the process input u can be stored in a file on the disc using the option store. Two hundred samples of the signals are stored in a text file. The program asks for a file name when this mode is activated.

### 3. THE CONTROL ALGORITHM

The control algorithm is a digital implementation of the PID-algorithm. A brief description is given in this Chapter. More details are found in [7]. The algorithm is in principle described by the operator:

$$G(p) = K \left[ 1 + \frac{1}{T_i p} + \frac{p T_d}{1 + p T_d / N} \right] \quad (3.1)$$

where  $p = d/dt$  is the differential operator. Let  $r$  denote the set point,  $y$  the measured output and  $e$  denote the error. Then

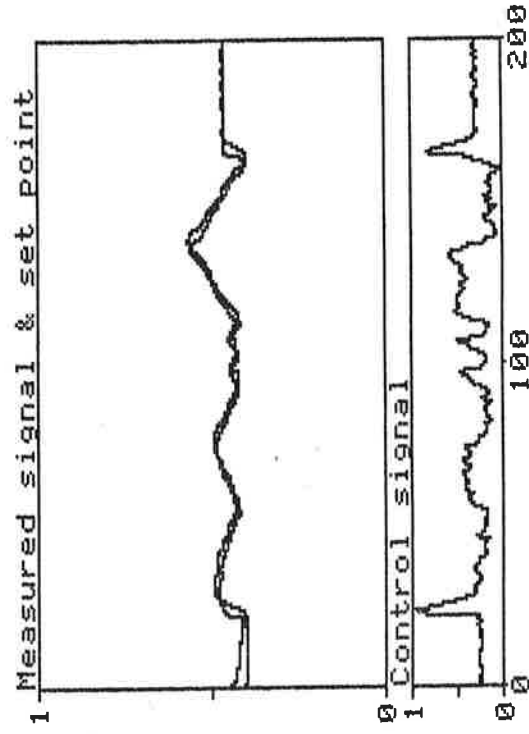


Figure 2 Sample of screen output in manual or automatic control.

$$e = r - y \quad (3.2)$$

The proportional term is then implemented as

$$P = K * e \quad (3.3)$$

The integral term is implemented by the following discrete approximation

$$I(t+h) = I(t) + \frac{K h}{T_i} e(t) \quad (3.4)$$

where  $h$  is the sampling period.

The derivative term is approximated as follows:

$$K \frac{pT_d}{1 + pT_d/N} = K \frac{pT_d + N - N}{1 + pT_d/N} = K N \left[ 1 - \frac{1}{1 + pT_d/N} \right]$$

Notice that the derivative action operates on the output and not on the control error. The derivative term is formed as follows:

$$D = K N (-y - z) \quad (3.5)$$

where

$$z(t) = \frac{1}{1 + pT_d/N} (-y(t)) \quad (3.6)$$

Notice the minus sign because of the definition (3.1) of the error. Rewriting (3.6) as a differential equation we get

$$(T_d/N) \frac{dz(t)}{dt} + z(t) = -y(t)$$

Approximating the derivative by a backward difference we get

$$(T_d/N) \frac{z(t) - z(t-h)}{h} + z(t) = -y(t)$$

Solving for  $z(t)$  we get

$$z(t) = -\frac{T_d}{Nh + T_d} z(t-h) - \frac{Nh}{Nh + T_d} y(t) \quad (3.7)$$

Notice this difference equation is stable for all sampling periods because the number  $T_d/(Nh + T_d)$  is always less than one.

The equation (3.7) can be written as

$$z(t) = z(t-h) - \frac{Nh}{Nh + T_d} [y(t) + z(t-h)] \quad (3.8)$$

Summarizing the control signal is thus given by



$$u(t) = P + I + D \quad (3.9)$$

where P, I and D are given by (3.3), (3.4) and (3.5).

#### 4. PROGRAM STRUCTURE

A description of the main structure of the program is given in this Chapter. The descriptions refers to the program listing in Appendix A.

Subroutines are used to structure the code since this is the only structuring mechanism in Basic. The logic order of the subroutines was sacrificed for speed. The part of the program that is used most frequently is put in the beginning of the program. This speeds up the execution.

##### The main program

The main program is on lines 200 to 370. It loops through the operation test for keypress and execution of menu commands.

The subroutine for initialization is invoked on line 230. Apple II has a cell # -16 384 which indicates if the keyboard has been pressed. The first bit of the contents of this cell is 0 if no key has been pressed and 1 if a key has been pressed. The code on lines 270 and 280 interrogates the first bit of cell # 16 384. The contents of this cell is also used for menu selection. The code on line 280 transfers the contents of the cell to the variable XX. The menu selection is then made on lines 300 to 360.

The code on line 300 is straightforward. In the code for manual control: line 310, automatic control: line 320, and hard copy: line 330, there are several "pokes" to control the switches for controlling the screen modes. These switches are described in the Apple II reference manual [1].

The instruction POKE UA,0 on line 340 sets the output to zero before quitting. The variable UA is the address to the output signal.

##### Subroutines

The program has the following subroutines:

Automatic	20-	30
Connections	600-	740
First page	100-	190
Hard copy	2000-	2670
Initialize	400-	590
Manual	1300-	1500
New parameters	800-	940
Pid	39-	67
Plot	70-	82
Print parameters	1000-	1090
Store	3000-	3220
Transform parameters	1100-	1280
Wait	31-	38

The structure of the program and the relations between the procedures are given in "pidgeon Pascal" in Listing 1.

```
First page
Main
begin
  Initialize
  Print parameters
  Transform parameters

  Repeat forever
  begin
    Test for keypress

    Case key of
    begin
      C: Connections

      P: New parameters
        Transform parameters
        Print parameters

      M: Manual
        Plot
        Wait

      A: Automatic
        Wait
        Pid
        Plot

      H: Hardcopy

      S: Store

      Q: Quit

    end case
  end repeat
end main
```

Listing 1 - Program structure.

The beginning of a major subroutine is denoted as follows:

```
REM *** name ***
REM *****.
```

The end of a subroutine is denoted as:

RETURN  
:

Detailed descriptions of the subroutines are given in the next Chapter.

## 5. SUBROUTINES

This Chapter describes the different subroutines in alphabetical order.

### Automatic

This subroutine consists of the lines 20-30. It uses three subroutines wait, pid and plot.

Line 23 resets a pointer (TS) by reading the current time (in seconds and tenths of seconds) and adding the sampling period TT. The rest of the subroutine is a loop from line 24 to line 27 which goes on calling the subroutines wait,pid and plot until a key is pressed. (The pokes in line 27 read the keyboard.) When a key is pressed line 28 clears the keyboard strobe before the program returns to the menu.

### Connections

This subroutine which handles all input and output connections begins at line 600 and ends at line 740. It does not use any other subroutines.

Line 630 sets the top of the screen window to line 20 to ensure that any writing done by this subroutine will not disturb the menu.

The lines 640-680 asks for the new values of the variables SCLOC, SAD, YCH, UCH and SPR which define the connections. The subroutine uses a string input so that the program will not crash if the user types a literal by mistake. The string input also makes it possible to keep the default values when the user types <return> instead of a number.

The subroutine changes the base addresses, SCLOC for the clock card, SAD for the A/D - D/A card, and SPR for the printer card at lines 690 through 710. The channel numbers YCH for the analog input and UCH for the analog output are also set on lines 660 and 670. Notice that the Mountain Hardware card has 12 analog inputs and 12 analog outputs.

The subroutine finally clears the four lowest lines of text and restores the normal screen window.

First page

This subroutine consists of the lines 100 to 180. It prints the program name and logo.

Hard copy

This is a subroutine for dumping the graphic screen to the printer. It consists of four parts, an initializing part (2030-2170), a curve adjust to eliminate a possible gap in the curves (2180-2300), a screen dump (2310-2480) and a curve adjust to restore the graph (2490-2670).

The initializing part first requests a reconfirmation by the user on lines 2050-2110. The execution can be aborted by typing <esc>. There is a choice of a small picture suitable for a report or a larger picture suitable for overhead projection. The code for this is on lines 2120-2140. The string variable GC\$ contains the control code for the printer to adjust size, rotation and position of the screen dump print-out. See the manual for the Microbuffer [6] for details. Line 2160 "pokes" to hi-res page two.

The two code sequences for curve adjust are almost identical to the plot routine but they use the arrays RT, YT and UT, where the reference, input and output values are stored.

The screen dump routine begins with an initialization command which gives the slot for the printer (line 2340). It then sends a form feed (chr\$(12)) in line 2350. Lines 2360-2420 prints out a headline and the regulator parameters. The actual screen dump is done at line 2440. <ctrl-l>+"G" is a special command for dumping graphics followed by the other control characters described earlier. The printer is turned off in line 2460, before the second curve adjusts routine

Initialize

This subroutine consists of the lines 400 to 590. It uses the subroutines print parameters: on lines 1000 to 1090, and transform parameters: on lines 1100 to 1230.

The subroutine initialize first gives default values to the parameters on lines 430 to 480. The graphics page which is stored on disc as SAM2.PIC is then loaded to memory on line 510.

Line 520 is a call to the subroutine which prints the parameters. The natural parameters K, Ti, Td, N, ulow and uhigh do not appear directly in the formula for the PID algorithm. The transformed variables KI, KD, AD, UL and UH are used instead to speed up the calculations. The transformation of the parameters is made in the subroutine transform parameters on lines 1100 to 1280. This subroutine is called on line 530.

The code on lines 540 to 557 prints the menu.

The code on line 570 initializes the clock. The variable CADDR contains the address to the clock. Remember that input output on the Apple II is memory mapped. The instruction POKE CCADDR,32 resets the clock and the instruction

POKE CADDR,16 starts the clock [4].

### Manual

This subroutine allows the user to run the process manually. It starts like the automatic routine, with resetting the pointer TS, used by the clock (line 1330). It then reads the two push-buttons (paddle#5 and paddle#6) in line 1340 and 1350. A variable UINC is increased or decreased on lines 1370 and 1380 depending on which button is pushed. The control variable U is then increased by UINC on line 1390. The control signal will thus accelerate if the button is pushed continuously. The acceleration will depend on the sampling period. The code on lines 1400 and 1410 limits the computed output U to the interval (ULOW and UHIGH).

Line 1420 and 1430 generate an analog output by "poking" the output variable to the A/D-D/A card. Line 1440 reads the current input from the process. Line 1450 reads the setpoint from paddle #0 and the line 1460 calls the plot subroutine to draw the new points on the graph. The code in lines 1470 and 1480 reads the keyboard and returns to the menu if a key is pressed.

### New parameters

This routine consists of the lines 600-740. It is very similar to the connection subroutine. The only difference is the text and the names of the variables. It uses the same input technique as the other routine.

### Pid

This procedure implements the discrete-time approximation of the PID algorithm described in Chapter 3. The equations (3.2), (3.3), (3.5), (3.8) and (3.9) are written as follows:

$$\begin{aligned} E &= R - Y \\ Z &= Z - AD * (Y + Z) \\ V &= K * E + I + KD * (-Y - Z) \\ U &= \text{sat}(V, \text{ulow}, \text{uhigh}) \\ I &= I + KI * E + AW * (U - V) \end{aligned}$$

The saturation function and the term  $U - V$  are introduced to avoid windup of the integrator. If the output saturates the integrator is set to a value which correspond to the an output at the saturation limit. This is explained in more detail in [7]. Notice also that the gain parameters KD and KI and the constant AD have been introduced to speed up the equations. This is explained in more detail in the description of the subroutine transform parameters.

The code which implements these equations is given on lines 50 to 55. Notice that the control algorithm is just a small fraction of the whole code.

Plot

The plotting is implemented so that it mimics a strip recorder. The curves are wrapped around because the plotting window has limited size. Part of the old curves are erased to make a clearer graph. When the curves are plotted using the hard copy option the curves are replotted so that the gap vanishes and the present time at the right most edge of the graph.

The plot routine starts on line 73 by increasing the variable PR. If the variable is less than PN, which is the variable for the plot frequency, then the routine returns without plotting or updating the arrays.

Line 74 increases the plot variable X and checks for a wrap-around. The variable OX denotes the old value of X. Similarly the variable SX contains the value of X to be erased. The variable SX is updated in line 75. Line 76 erases the old curves and lines 77 to 79 plots the three signals. The variables Y1, Y2 and Y3 denote old values of the y-coordinates.

The three arrays UT, YT and RT are updated on line 80.

Print parameters

This subroutine prints the parameters appearing on the menu. Lines 1030 and 1080 take care of the screen window and the lines 1040-1070 does the printing.

Store

This routine makes it possible to store the 200 last samples on disk. It stores the samples as a standard Apple II text-file. The order is:

```
1'st process output
1'st process input
1'st setpoint
2'nd process output
2'nd process input
2'nd setpoint
3'rd process output
etc ...
```

Line 3030 sets the window and line 3040 clears it. Then line 3050 asks for a reconfirmation. Line 3060 and line 3070 take care of the file name.

The file is opened in lines 3090-3120 and written in the loop between 3130 and 3180. It uses the pointer X to start writing from the right places in the circular arrays YT, UT and RT. Line 3190 closes the array and the normal window is restored on line 3200 before the routine ends.

### Transform parameters

This subroutine computes the parameters AD, KD and KI used in the formula for the PID algorithm.

The constant AD is given by

$$AD = \frac{T_d}{N h + T_d} \quad (5.1)$$

Compare with equation (3.8) and AW is given by

$$AW = H/TTI \quad (5.2)$$

The derivative gain KD is given by

$$KD = K N \quad (5.3)$$

See equation (3.5). The integral gain is given by

$$KI = \frac{K h}{T_i} \quad (5.4)$$

Compare with equation (3.4).

The scaled version levels of the control limits are also computed. In algorithm design we always think of the variables as being in the range 0 to 1 for unipolar signals. The limits ulow and uhigh are thus given in this scale. In the program the measured signals are, however, represented as integers in the range 0 to 255. These numbers are obtained directly from the AD conversion. The scale is not changed in order not to lose computational speed. The following formula are used to compute the scaled limits:

$$LS = ulow * 256 \quad (5.5)$$

$$HS = uhigh * 256 \quad (5.6)$$

The variables LS and HS are also constrained to be nonnegative and less than 255 to avoid program crashes for bad input data.

### Wait

When the program is in automatic or manual mode the execution is governed by the clock. A simple loop where the clock reading is compared with the next sampling instant is used. When it is time to sample, the next sampling period is updated and the program is executed. The actual code is given on lines 34-37 in the program listing. Notice that it is very important that the next sampling time is initiated properly. Otherwise the program will never start. The clock is initiated on lines 23 and 1330. See [4] for details.

## 6. VARIABLES

Since all variables in Basic are global it is useful to have a list of them. Such a list is given below. Notice that Applesoft Basic allows long names but only the first two letters are actually used as identifiers. A cross reference table is listed in Appendix B.



List of variables used in the program

A\$ - local variable to select size of graph  
 AD - parameter used to calculate derivative action  
 AW - parameter which adjusts anti-windup computation  
 CA - clock address  
 D\$ - cntrl-D  
 E - control error  
 GC\$ - local string variable used in hard copy  
 H - sampling period, input data  
 HS - scaled upper bound to control variable, HS<256  
 I - integral term in PID algorithm  
 J - loop index  
 K - regulator gain, input data  
 KD - derivative gain  
 KI - integral gain  
 LS - scaled lower bound to control variable, LS≥0  
 N - maximum derivative gain, input data  
 N\$ - string variable used to name the stored file  
 OX - local variable in plot  
 P1 - local variable in manual  
 P2 - local variable in manual  
 PN - every PNth point is plotted, input data  
 PR - local variable in plot  
 RS - set point  
 RT( - array for storing set point  
 SA - slot address for AD card  
 SC - slot address for clock card  
 SP - slot address for printer card  
 SX - local variable in plot and hard copy  
 T - local variable in wait  
 TD - derivative time, input data  
 TI - integration time, input data  
 TS - variable which represents the next sampling instant  
 TT - int(H\*10)  
 U - control variable  
 UA - address to analog output  
 UC - channel number for control variable  
 UH - upper bound of control variable (0,1), input data  
 UI - local variable in manual  
 UL - lower bound of control variable (0,1), input data  
 US - scaled control variable 0≤U<256  
 UT( - array for storing control variable  
 V - local variable in pid  
 X - local variable in plot, hard copy and store  
 XX - local variable automatic and main  
 XX\$ - string variable for input  
 Y - measured variable  
 Y1 - local variable in plot and hard copy  
 Y2 - local variable in plot and hard copy  
 Y3 - local variable in plot and hard copy  
 YA - address to measured value  
 YC - channel number for measured signal  
 YS - scaled measured value 0≤YS<256  
 YT( - array for storing measured signal  
 YY - local variable in plot and hard copy  
 Z - state variable in computation of derivative term

## 7. REFERENCES

- [1] Apple IIe Owners Manual. Manual #A2L2001. Apple Computer, Inc. Cupertino, CA. 1982.
- [2] Apple II Applesoft BASIC Programmers Manual - Volume 1. Manual #A2L2005. Apple Computer, Inc. Cupertino, CA. 1982.
- [3] A/D + D/A CARD Operating Manual. Manual 11-00230-02. Mountain Computer Inc. Scotts Valley, CA. 1982.
- [4] U/TIM User Manual. Manual 01.044. U-Microcomputers Ltd, Warrington. Dec 1981.
- [5] MX-80 Epson dot matrix printer type II. Operation Manual P8190014-1. Shinshu Seiki Co. Ltd. Nagano, Japan.
- [6] MICROBUFFER II Users Manual. Practical Periferals, Inc. Westlake Village, CA. May 1, 1982.
- [7] Aström, K. J. Reglerteknik - En elementär introduktion. Kap 5 PID reglering. Report TFRT-3166. Department of Automatic Control, LTH, Lund 1982.

APPENDIX A - PROGRAM LISTING

```

1 REM *****
2 REM *
3 REM * PID REGULATOR *
4 REM *
5 REM *****
6 REM
7 REM KALLE & KJ ASTROM
8 REM
9 REM 1983-07-04
10 :
15 GOSUB 100: GOTO 200
19 :
20 REM *** AUTOMATIC ***
21 REM *****
22 :
23 TS = PEEK (CA + 3) * 10 + PEEK (CA + 2) + TT: IF TS > 99 THEN TS
= TS - 100
24 GOSUB 31: REM WAIT FOR SAMPLING TIME
25 GOSUB 39: REM PID
26 GOSUB 73: REM PLOT
27 XX = PEEK (- 16384): IF XX < 128 THEN 24
28 POKE - 16368,0
29 RETURN
30 :
31 REM *** WAIT ***
32 REM *****
33 :
34 T = PEEK (CA + 3) * 10 + PEEK (CA + 2)
35 IF T < > TS THEN 34
36 TS = TS + TT: IF TS > 99 THEN TS = TS - 100
37 RETURN
38 :
39 REM *** PID ***
40 REM *****
41 :
42 REM * A/D CONVERSION *
43 YS = PEEK (YADDR):YS = PEEK (YADDR)
44 RS = PDL (0)
45 Y = YS
46 Y = YS
47 :
48 REM * PID ALGORITHM *
49 :
50 E = RS - YS
51 Z = Z - AD * (Z + Y)
52 V = K * E + I + KD * (- Y - Z)
53 U = V
54 IF U < LS THEN U = LS
55 IF U > HS THEN U = HS
56 US = INT (U)
57 :
58 REM * D/A CONVERSION *
59 :
60 POKE UADDR,US
61 :
62 REM * UPDATE INTEGRAL *
63 :
64 I = I + U - V + KI * E
65 IF TI < 0 THEN I = 0: REM NEGATIVE TI TURNS OFF INTEGRAL ACTION
66 RETURN
67 :
70 REM *** PLOT ***
71 REM *****

```

```

72 :
73 PR = PR + 1: IF PR < PN THEN RETURN
74 X = X + 1: IF X > 212 THEN X = 12:OX = 12
75 SX = SX + 1: IF SX > 212 THEN SX = 12
76 HCOLOR= 0: HPLLOT SX,9 TO SX,137: HPLLOT SX,149 TO SX,181: HCOLOR= 3
77 YY = INT (137 - RS / 2): HPLLOT OX,Y1 TO X,YY:Y1 = YY
78 YY = INT (181 - US / 8): HPLLOT OX,Y2 TO X,YY:Y2 = YY
79 YY = INT (137 - YS / 2): HPLLOT OX,Y3 TO X,YY:Y3 = YY:OX = X:PR = 0
80 UT(X, - 12) = US:YT(X - 12) = YS:RT(X - 12) = RS
81 RETURN
82 :
100 REM *** FIRST PAGE ***
110 REM *****
120 :
130 HOME
131 VTAB 5: PRINT "*****";
132 PRINT " "
133 PRINT " "
134 PRINT " "
135 PRINT "*****";
140 VTAB 12
141 HTAB 5: PRINT "THIS PROGRAM IMPLEMENTS A PID"
142 HTAB 5: PRINT "REGULATOR ON THE APPLE IIE"
143 HTAB 5: PRINT "WITH THE MOUNTAIN COMPUTER"
144 HTAB 5: PRINT "A/D+D/A BOARD AND U-TIM CLOCK"
145 HTAB 5: PRINT "BOARD."
150 VTAB 20: PRINT "
151 PRINT "PRESS ANY KEY TO CONTINUE * *"
152 PRINT "
160 VTAB 21: HTAB 36: GET XX
170 HOME
180 RETURN
190 :
200 REM *** MAIN PROGRAM ***
210 REM *****
220 :
230 GOSUB 400: REM INITIALIZE
240 :
250 REM * READ KEYBOARD *
260 :
270 IF PEEK ( - 16384) < 128 THEN 270
280 XX = PEEK ( - 16384): POKE - 16368,0
281 :
282 REM * ON KEYPRESS GOTO *
283 REM * 195="C" CONFIG *
284 REM * 208="P" PARAM *
285 REM * 205="M" MANUAL *
286 REM * 193="A" AUTO *
287 REM * 200="H" HARD COPY*
288 REM * 211="S" STORE *
289 REM * 209="Q" QUIT *
290 :
300 IF XX = 195 THEN GOSUB 600: GOTO 270
310 IF XX = 208 THEN GOSUB 800: GOSUB 1100: GOSUB 1000: GOTO 270
320 IF XX = 205 THEN POKE - 16297,0: POKE - 16304,0: POKE - 16299
,0: POKE - 16302,0: GOSUB 1300: POKE - 16303,0: POKE - 16300,0: GOT
0 270
330 IF XX = 193 THEN POKE - 16297,0: POKE - 16304,0: POKE - 16299
,0: POKE - 16302,0: GOSUB 20: POKE - 16303,0: POKE - 16300,0: GOTO
270
340 IF XX = 200 THEN GOSUB 2000: GOTO 270
350 IF XX = 211 THEN GOSUB 3000: GOTO 270
360 IF XX = 209 THEN TEXT : HOME : POKE UA,0: END
370 GOTO 270
380 :
400 REM *** INITIALIZE ***

```

```

410 REM *****
420 :
430 REM *DEFAULT PARAMETERS*
440 :
445 DIM UT(200),YT(200),RT(200)
450 H = 1:K = 10:TI = 10:TD = .1:N = 0:ULOW = 0:UHIGH = 1:PN = 1
460 SCLOC = 4:SAD = 5:YCH = 0:UCH = 0:SPR = 2:UA = 49360:YA = 49360:CA
    = 16192
470 X = 11:DX = 11:Y1 = 50:Y3 = 50:Y2 = 150:YX = 21:DX = CHR$(4)
480 :
490 REM * INIT PROCESS *
500 :
510 HGR2 : TEXT : HOME : INVERSE : VTAB 5: HTAB 5: PRINT "LOADING DAT
A FROM DISK PLEASE WAIT": NORMAL : PRINT DX;"BLOAD SAM2.PIC": HOME
520 GOSUB 1000: REM PRINT PARAMETERS
530 GOSUB 1100: REM TRANSFER PARAMETERS
540 VTAB 1: HTAB 1
541 PRINT "*****"
542 PRINT "*" PID REGULATOR *****"
543 PRINT "*" VERSION I UNIPOLAR SIGNALS *****"
544 PRINT "*" 83-07-04 *****"
545 PRINT "*" KALLE & KJ ASTROM *****"
546 PRINT "*****"
550 VTAB 8
551 HTAB 5: PRINT "ALTER CONFIGURATION (C)"
552 HTAB 5: PRINT "ALTER PARAMETERS (P)"
553 HTAB 5: PRINT "MANUAL (M)"
554 HTAB 5: PRINT "AUTOMATIC (A)"
555 HTAB 5: PRINT "HARD COPY (H)"
556 HTAB 5: PRINT "STORE (S)"
557 HTAB 5: PRINT "QUIT (Q)"
560 HCOLOR= 3
570 POKE CA,32: POKE CA,16
580 RETURN
590 :
600 REM *** CONECTIONS ***
610 REM *****
620 :
630 POKE 34,20: HOME
640 PRINT "ENTER SLOT NUMBER FOR CLOCK CARD ";SCLOC;" ";: INPUT XX$:
IF LEN (XX$) THEN SCLOC = VAL (XX$)
650 PRINT "ENTER SLOT NUMBER FOR A/D D/A CARD ";SAD;" ";: INPUT XX$:
IF LEN (XX$) THEN SAD = VAL (XX$)
660 PRINT "ENTER CHANNEL # FOR MEASURE SIGNAL ";YCH;" ";: INPUT XX$:
IF LEN (XX$) THEN YCH = VAL (XX$)
670 PRINT "ENTER CHANNEL # FOR ANALOG OUTPUT ";UCH;" ";: INPUT XX$:
IF LEN (XX$) THEN UCH = VAL (XX$)
680 PRINT "ENTER SLOT NUMBER FOR PARALELL PRINTER INTERFACE ";SFR;"
";: INPUT XX$: IF LEN (XX$) THEN SFR = VAL (XX$)
690 YADDR = 49280 + SAD * 16 + YCH
700 UADDR = 49280 + SAD * 16 + UCH
710 CADDR = - 16256 + 16 * SCLOC
720 HOME : POKE 34,0
730 RETURN
740 :
800 REM *** NEW PARAMETERS ***
810 REM *****
820 :
830 POKE 34,20: HOME
840 VTAB 21: PRINT "SAMPLING PERIOD ";H;" ";: INPUT XX$: IF LEN (XX$
) THEN H = VAL (XX$)
850 PRINT "K ";K;" ";: INPUT XX$: IF LEN (XX$) THEN K = VAL (XX$)
860 PRINT "TI ";TI;" ";: INPUT XX$: IF LEN (XX$) THEN TI = VAL (XX$
)
870 PRINT "TD ";TD;" ";: INPUT XX$: IF LEN (XX$) THEN TD = VAL (XX$
)

```

```

880 PRINT "N";N;" ";; INPUT XX0: IF LEN (XX0) THEN N = VAL (XX0)
890 PRINT "LOW OUTPUT LIMIT ";ULOW;" ";; INPUT XX0: IF LEN (XX0) THE
N ULOW = VAL (XX0)
900 PRINT "HIGH OUTPUT LIMIT ";UHIGH;" ";; INPUT XX0: IF LEN (XX0) T
HEN UHIGH = VAL (XX0)
910 PRINT "PLOT EVERY ";FN;" TIME(S) ";; INPUT XX0: IF LEN (XX0) THE
N FN = VAL (XX0)
920 HOME : POKE 34,0
930 RETURN
940 :
1000 REM ** PRINT PARAMETERS **
1010 REM *****
1020 :
1030 POKE 34,16: HOME
1040 VTAB 16: HTAB 5: PRINT "SAMPLING PERIOD = ";H;" S "
1050 HTAB 5: PRINT "K = ";K TAB( 20)"N = ";N
1060 HTAB 5: PRINT "TI = ";TI TAB( 20)"TD = ";TD: IF TI < 0 THEN VTA
B 18: HTAB 10: PRINT "OFF";: VTAB 19
1070 HTAB 5: PRINT "TIME UNIT IS ";H * FN;" S "
1080 POKE 34,0: RETURN
1090 :
1100 REM *TRANSFORM PARAMETERS*
1110 REM *****
1120 :
1130 H = INT (10 * H) / 10: IF H < .5 THEN H = .5
1140 IF TI = 0 THEN TI = 1
1150 KI = K * H / TI
1160 AD = N * H / (TD + N * H)
1170 KD = K * N
1180 LS = INT (256 * ULOW)
1190 IF LS < 0 THEN LS = 0
1200 IF LS > 255 THEN LS = 255
1210 HS = INT (256 * UHIGH)
1220 IF HS > 255 THEN HS = 255
1230 IF HS < LS THEN HS = LS
1240 TT = INT (10 * H)
1250 IF TT > 99 THEN TT = 99
1260 FN = INT (FN): IF FN < 1 THEN FN = 1
1270 RETURN
1280 :
1300 REM *** MANUAL ***
1310 REM *****
1320 :
1330 TS = PEEK (CA + 3) * 10 + PEEK (CA + 2) + TT: IF TS > 99 THEN T
S = TS - 100
1340 P1 = PDL (6)
1350 P2 = PDL (5)
1360 IF NOT (P1 OR P2) THEN UINC = 0
1370 IF P1 THEN UINC = UINC + 1
1380 IF P2 THEN UINC = UINC - 1
1390 U = U + UINC
1400 IF U > HS THEN U = HS
1410 IF U < LS THEN U = LS
1420 US = INT (U):I = US
1430 POKE UADDR,US
1440 YS = PEEK (YADDR):YS = PEEK (YADDR)
1450 RS = PDL (0)
1460 GOSUB 73
1470 IF PEEK (- 16384) < 128 THEN GOSUB 31: GOTO 1340
1480 POKE - 16368,0
1490 RETURN
1500 :
2000 REM *** HARD COPY ***
2010 REM *****
2020 :
2030 POKE 34,20

```

```

2040 HOME
2050 VTAB 21: PRINT "AN EPSON PRINTER WITH PARALLEL INTERFACE"
2060 VTAB 22: PRINT "IS REQUIRED TO EXECUTE THIS COMMAND"
2070 PRINT "PRESS <RETURN> TO CONTINUE "
2080 PRINT "OR <ESC> TO GET BACK TO MENU";
2090 GET A$
2100 IF A$ = CHR$(27) THEN HOME : POKE 34,0: RETURN
2110 IF A$ < > CHR$(13) THEN 2040
2120 HOME : PRINT "DO YOU WANT A LARGE PICTURE (Y/N) ? " : GET XX$
2130 IF XX$ = "Y" THEN GC$ = "G2DELR": GOTO 2160
2140 GC$ = "G2E"
2150 REM GC$ GRAPHIC CODE TO BE SENT TO PRINTER
2160 POKE - 16304,0: POKE - 16299,0: POKE - 16302,0: REM SWITCH 1
0 HI-RES GRAPHIC PAGE TWO
2170 :
2180 REM * CURVE ADJUST *
2190 :
2200 HCOLOR= 0: HPLOT 12,9 TO 12,137: HPLOT 12,149 TO 12,181
2210 FOR J = 0 TO 200
2220 X = X + 1: IF X > 212 THEN X = 12
2230 HCOLOR= 0
2240 IF J < 200 THEN HPLOT J + 13,9 TO J + 13,137: HPLOT J + 13,149
TO J + 13,181
2250 HCOLOR= 3
2260 YY = INT (137 - RT(X - 12) / 2): HPLOT J + 11, Y1 TO J + 12, YY: Y1
= YY
2270 YY = INT (181 - UT(X - 12) / 8): HPLOT J + 11, Y2 TO J + 12, YY: Y2
= YY
2280 YY = INT (137 - YT(X - 12) / 2): HPLOT J + 11, Y3 TO J + 12, YY: Y3
= YY
2290 NEXT J
2300 :
2310 REM * SCREEN DUMP *
2320 :
2330 PRINT
2340 PRINT D$;"PR#";SPR
2350 PRINT CHR$(12)
2360 PRINT "Results from experiments with PID control."
2370 PRINT
2380 PRINT "Parameters:"
2390 PRINT "Sampling period: ";H;" s"
2400 PRINT "K : ";K TAB(20)"N : " ;N
2410 IF TI < 0 THEN PRINT "TI : OFF" TAB(20)"TD : " ;TD: GOTO 2430
2420 PRINT "TI : " ;TI TAB(20)"TD : " ;TD
2430 PRINT "A time unit corresponds to " ;FN * H;" s"
2440 PRINT : PRINT
2450 PRINT CHR$(9);GC$: REM <CTRL-I>+GRAPHIC CODE TO PRINTER.
2460 PRINT
2470 PRINT D$;"PR#0"
2480 :
2490 REM * CURVE ADJUST *
2500 :
2510 HCOLOR= 0: HPLOT 12,9 TO 12,137: HPLOT 12,149 TO 12,181
2520 FOR J = 12 TO 212
2530 HCOLOR= 0
2540 IF J < 212 THEN HPLOT J + 1,9 TO J + 1,137: HPLOT J + 1,149 TO
J + 1,181
2550 HCOLOR= 3
2560 YY = INT (137 - RT(J - 12) / 2): HPLOT J, Y1 TO J, YY: Y1 = YY
2570 YY = INT (181 - UT(J - 12) / 8): HPLOT J, Y2 TO J, YY: Y2 = YY
2580 YY = INT (137 - YT(J - 12) / 2): HPLOT J, Y3 TO J, YY: Y3 = YY
2590 NEXT J
2600 HCOLOR= 0
2610 X = X + 1: IF X = 213 THEN X = 12
2620 HPLLOT X,9 TO X,137: HPLLOT X,149 TO X,181: IF X < > SX THEN 2610
2630 X = X - 10: IF X < 13 THEN X = X + 20

```

```

2640 :
2650 POKE - 16303,0: POKE - 16300,0: REM SWITCH BACK TO TEXT (I.E.
THE MENU)
2660 Y1 = INT (137 - RT(X - 12) / 2):Y2 = INT (181 - UT(X - 12) / 8)
:Y3 = INT (137 - YT(X - 12) / 2)
2670 HOME
2680 POKE 34,0
2690 RETURN
2700 :
3000 REM *** STORE ***
3010 REM *****
3020 :
3030 POKE 34,20
3040 HOME
3050 PRINT "ARE YOU SURE (Y/N) ? " : GET XX$: IF XX$ < > "Y" THEN G
OTO 3200
3060 HOME : INPUT "FILENAME :";N$
3070 HOME : INVERSE : PRINT "STORING AS ";N$:".TEXT": NORMAL
3080 N$ = N$ + ".TEXT"
3090 PRINT D$:"OPEN ";N$
3100 PRINT D$:"DELETE ";N$
3110 PRINT D$:"OPEN ";N$
3120 PRINT D$:"WRITE ";N$
3130 FOR J = 0 TO 200
3140 X = X + 1: IF X = 213 THEN X = 12
3150 PRINT STR$(YT(X - 12))
3160 PRINT STR$(UT(X - 12))
3170 PRINT STR$(RT(X - 12))
3180 NEXT J
3190 PRINT D$:"CLOSE ";N$
3200 HOME : POKE 34,0
3210 RETURN
3220 :
9999 PRINT "PR#2"
10000 PRINT CHR$(27);"Q"; CHR$(70)
10001 LIST 320
10002 PRINT "PR#0"

```



APPENDIX B - CROSS REFERENCE TABLE

A&X	
AX	2090, 2100, 2110
AD	51, 1160
CA	23, 34, 460, 570, 710, 1330
DX	470, 510, 2340, 2470, 3090, 3100, 3110, 3120, 3190
E	50, 52, 64
GCX	2130, 2140, 2450
H	450, 840, 1040, 1070, 1130, 1150, 1160, 1240, 2390, 2430
HS	55, 1210, 1220, 1230, 1400
I	52, 64, 65, 1420
J	2210, 2240, 2260, 2270, 2280, 2290, 2520, 2540, 2560, 2570, 2590
K	2590, 3130, 3180
KD	52, 450, 850, 1050, 1150, 1170, 2400
KI	52, 1170
LI	64, 1150
LS	54, 1180, 1190, 1200, 1230, 1410
N	450, 880, 1050, 1160, 1170, 2400
NX	3060, 3070, 3080, 3090, 3100, 3110, 3120, 3190
OX	74, 77, 78, 79, 470
F1	1340, 1360, 1370
F2	1350, 1360, 1380
FN	73, 450, 910, 1070, 1260, 2430
PR	73, 79
RS	45, 50, 77, 80, 1450
RT	80, 445, 2260, 2560, 2660, 3170
SA	460, 650, 690, 700
SC	460, 640, 710
SP	460, 680, 2340
SX	75, 76, 470, 2620
T	34, 35
TD	450, 870, 1060, 1160, 2410, 2420
TI	65, 450, 860, 1060, 1140, 1150, 2410, 2420
TS	23, 35, 36, 1330
TT	23, 36, 1240, 1250, 1330
U	53, 54, 55, 56, 64, 1390, 1400, 1410, 1420
UA	60, 360, 460, 700, 1430
UC	460, 670, 700
UH	450, 900, 1210
UI	1360, 1370, 1380, 1390
UL	450, 890, 1180
US	56, 60, 78, 80, 1420, 1430
UT	80, 445, 2270, 2570, 2660, 3160
V	52, 53, 64
X	74, 77, 78, 79, 80, 470, 2220, 2260, 2270, 2280, 2610, 2620, 263
O,	2660, 3140, 3150, 3160, 3170
XX	27, 280, 300, 310, 320, 330, 340, 350, 360
XXX	160, 640, 650, 660, 670, 680, 840, 850, 860, 870, 880, 890, 900,
910,	2120, 2130, 3050
Y	46, 51, 52
Y1	77, 470, 2260, 2560, 2660
Y2	78, 470, 2270, 2570, 2660
Y3	79, 470, 2280, 2580, 2660
YA	44, 460, 690, 1440
YC	460, 660, 690
YS	44, 46, 50, 79, 80, 1440
YT	80, 445, 2280, 2580, 2660, 3150
YV	77, 78, 79, 2260, 2270, 2280, 2560, 2570, 2580
Z	51, 52