CHAPTER - IV

SOFTWARE DEVELOPMENT AND CALIBRATION OF THE SYSTEM

4.1 INTRODUCTION

Chapter III describes the system hardware which is useful for the parameter measurement of induction motor. To perform the particular task there should be some sequence in the operation to be performed, which is called the synchronization in the various operations. In the computer based system this can be achieved through the software. In the present chapter required software will be developed in order to measure the parameters of a motor like stator current, speed and applied voltage to the motor. Before measurement of these parameters the system calibration is very essential for the accurate performance of the data acquisition system. Calibration of the present system is done through the system software.

These parameters are further used to calculate the performance parameters of the motor like torque, efficiency, horse power, power factor and power loss. After completion of calculation these parameters are presented graphically.

4.2 SOME GUIDELINES WHICH WE HAVE USED WHILE DEVELOPING THE SOFTWARE

The greatest difficulties in writing large computer programs are not in deciding what the goals of the program should be or even in finding methods that can be used to reach these goals.

The first major hurdle in solving a large problem is to decide exactly what the problem is. It is necessary to translate vague goals, contradictory requests and perhaps

unstated desires into a precisely formulated project that can be programmed. The maxim that many programmers observed "First make your program work and then make it preety".

Each part of a large program must be well organized, clearly written and thoroughly understood, or else its structure will have been forgotten and it can no longer be tied to the other parts of the project at some much later time by another programmer.

4.2.1 Program Correctness

1. Reduce the number of bugs, making it easier to spot those that remain.
2. Enable us to verify in advance that our algorithms are correct.
3. Provide the ways to test our programs so that we can reasonably be confident that they will not misbehave.

4.2.2. Pin points to be remembered while software development

1. Always name your variables and subprograms with greatest care and explain them thoroughly.
2. Keep your documentation concise but descriptive.
3. The reading time for programs is much more than the writing time. Make reading easy to do.
4. Each subprogram should do only one task but do it well.
5. Each program should hide something.
6. Keep your connections simple, avoid global variables wherever possible.

7. Never cause side effects. If you must use global variables as input document them thoroughly.

8. Keep your input and output as separate modules so that they can be changed easily and can be custom tailored your computing system.

9. The quality of test data is more important than its quantity.

4.3 PROGRAM DEVELOPMENT

The over all working of the system is mainly depends upon synchronization of the individual blocks. This can be achieved through the system software. The developed software is listed at the end of this chapter and can be divided into following modules.

1. Main program
2. Input module
3. Computation module
4. Display module

4.3.1 Main Program

This program is menu driven and has following options.

1. Data access from ADC
2. Data output and graph plotting
3. Delete old data
4. Exit.

The flow chart of main program is given in fig. 4.1. This module performs the first task of initialization of the card. The PCL - 207 I/O driver is written in assembly language

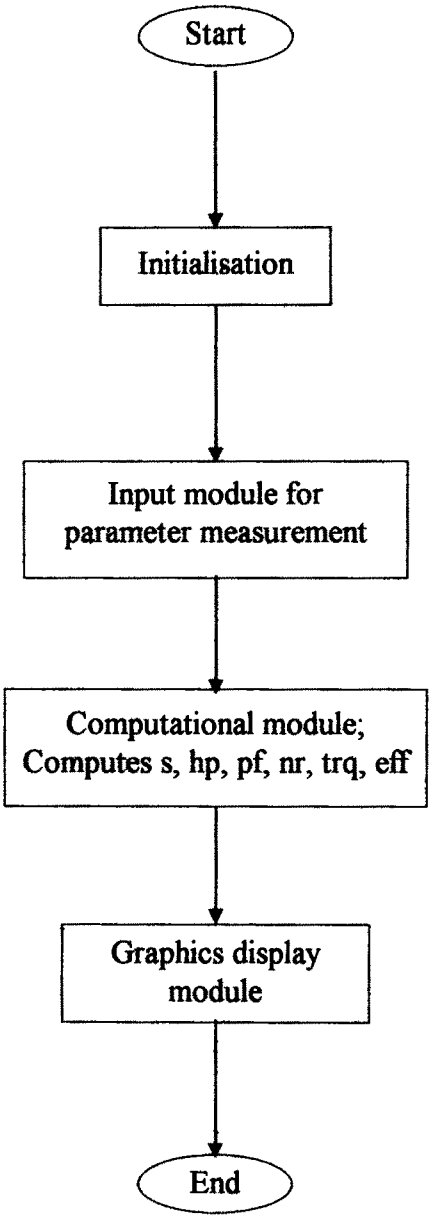

Fig. 4.1 : Flow Chart : Main Program

and compiled into a program 207BAS.BIN. The actual program size is less than 2K bytes. There are two ways to load the driver into the memory and the user must select it depending on the memory size available for the PC.

In the first method the driver is loaded immediately after the normal BASIC workspace area. This is a more difficult method as it requires the correct number of free bytes available in the basic workspace area and calculation of the driver routine starting address. The steps involved are as follows.

1. Check the free bytes available in the BASIC workspace area.
2. If the BASIC free work space does not exceed 60 bytes it is recommended to reduce the workspace by using the BASIC WS (workspace) command and reserve sufficient space for driver routines.
3. Identify the segment address containing BASIC .
4. Calculate the end address of the BASIC workspace from the position the driver program is loaded.
5. Load the binary program 207BAS.BIN using the BLOAD command.

In the second method the I/O driver is loaded to a memory segment which is independent from the BASIC workspace. This approach is easier to use, but it requires more free memory space.

The steps involved in this method are

1. Locate memory segment with at least 2K bytes free at its beginning.
2. Load the binary program 207BAS.BIN with the BLOAD command from the beginning of the memory segment.

Since the driver is located outside the BASIC work area, the driver should be linked by a DEF command immediately before each driver function.

The PCL - 207 requires 16 consecutive address in the I/O space. In order to program the PCL - 207, a good understanding of the 16 registers addressable from the selected I/O port base address is necessary. A summary of the functions of each address and the data format of each register is given below:

I/O Port Address Map

The following table shows the location of each register and driver relative to the base address.

LOCATION	READ	WRITE
BASE + 0	N/U	N/U
+1	N/U	N/U
+2	N/U	N/U
+3	N/U	N/U
+4	A/D low byte	D/A low byte
+5	A/D high byte	D/A high byte
+6	N/U	N/U
+7	N/U	N/U
+8	N/U	N/U
+9	N/U	N/U
+10	N/U	MUX scan channel
+11*	N/U	Software A/D trigger
+12*	N/U	Software A/D trigger
+13	N/U	N/U
+14	N/U	N/U
+15	N/U	N/U

NOTE : N/U = Not used

* BASE +11 and BASE +12 can be selected through SW7 and SW8 of the DIP switch. Their combinations are as given bellow:

SW7	SW8	Location
ON	OFF	BASE +11
OFF	ON	BASE +12

The input module is used to sense the parameters and take the data of individual parameters. Use separate files to store the data read from the ports. This data is further used by the computational module to calculate different performance parameters . The third module is the graphics module which is used to display the results of the motors under testing.

4.3.2 Input Module

This module performs the task of data collection from the motor under control. To do the same following steps are involved.

4.3.2.1 A/D Data Registers

The PCL - 207 performs 12 bit A/D conversion, an 8-bit register is not big enough to accommodate all 12-bits of data. Therefore, A/D data is stored in two registers located at address BASE + 4 and BASE + 5.

Data Format

1. A/D Low byte

BASE + 4	D7	D6	D5	D4	D3	D2	D1	D0
	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0

2. A/D High byte

BASE + 5	D7	D6	D5	D4	D3	D2	D1	D0
	0	0	0	DRDY	AD11	AD10	AD9	AD8

LEGEND

AD11 to AD0 -Analog to digital data. AD0 is the least significant byte (LSB) and AD11 is the most significant byte (MSB) of the A/D data.

DRDY - Data ready signal, 0 : data ready, 1 : data not ready

4.3.2.2 MUX Scan Register

The MUX scan register is a write only register using address BASE + 10. The low nibble provides the scan channel number. The MUX multiplexer switches to a new channel when written to this register.

Data Format

BASE +10	**D7**	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**
Scan Channel	**X**	**X**	**X**	**X**	**X**	**CL2**	**CL1**	**CL0**
Channel 0	X	X	X	X	X	0	0	0
Channel 1	X	X	X	X	X	0	0	1
Channel 2	X	X	X	X	X	0	1	0
Channel 3	X	X	X	X	X	0	1	1
Channel 4	X	X	X	X	X	1	0	0
Channel 5	X	X	X	X	X	1	0	1
Channel 6	X	X	X	X	X	1	1	0
Channel 7	X	X	X	X	X	1	1	1

LEGEND

CL2 to CL0 - Scan channel number

4.3.2.3 Trigger Mode

PCL - 207 A/D conversion is triggered by software. The software trigger is controlled by the application program issued software command.

4.3.2.4 A/D Data Transfer

The program controls the PCL-207 A/D data transfer using the polling concept. After the A/D converter has been triggered, the application program checks the Data Ready (DRDY) bit of the A/D status register. If the DRDY is detected, the converted data is moved from the A/D data register to computer memory by application program.

4.3.2.5 Execute an A/D Conversion

The user may execute A/D operations by writing a program with I/O instructions directly, or by a program utilizing the PCL-207 driver routines. Given in brief is a step by step implementation procedure for different A/D operations.

1. To perform a software trigger and program control data transfer without the PCL-207 driver.

   Step 1: Set the input channel by specifying the MUX scan range.

   Step 2 : Trigger by writing some value to the I/O port (BASE + 12).

   Step 3 : Wait for the DRDY by reading the A/D high byte register (BASE + 5) DRDY bit.

   Step 4 : Read data from A/D converter by reading the A/D data registers (BASE + 5 and + 4). Always read high byte first.

   Step 5 : Data conversion of binary A/D data to an integer.

The demonstration program (DEMO 01.BAS) in the software diskette performs this function.

2. To perform software trigger and program control data transfer with PCL-207 driver routines.

Step 1: Load 207BAS.BIN driver program to a memory segment.

Step 2: Use FUNCTION 0 to initialize the driver.

Step 3: Use FUNCTION 1 to set input channel range.

Step 4: Use FUNCTION 3 to perform a single A/D conversion.

The demonstration program (DEMO 02.BAS) in the software diskette performs this operation.

3. To perform software trigger and direct to array data transfer with PCL-207 driver.

Step 1 : Load 207BAS.BIN driver program to a memory segment.

Step 2 : Use FUNCTION 0 to initialize the driver.

Step 3 : Use FUNCTION 1 to set input channel range.

Step 4 : Use FUNCTION 4 to perform N A/D conversions and transfer data to array.

The demonstration program (DEMO 03.BAS) in the software diskette performs this operation.

The data read from the different channel is stored in different files, which will be used by the computational module afterwards. The Fig. 4.2 shows the flow chart for the second model.

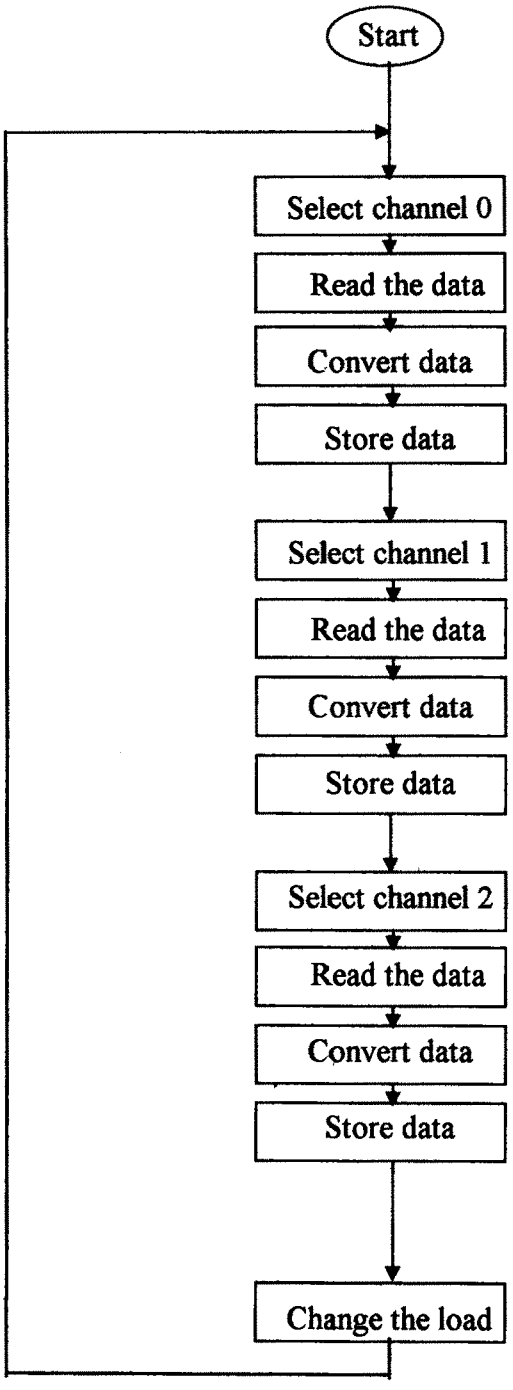

Fig. 4.2 : Flow Chart : Input Module

4.3.3. Computational Module

This module performs the task of computation of performance parameters of the motor by using the data read by input port and stored in individual file. The flow chart of the same is shown in Fig. 4.3.

4.3.4. Display Module

The performance calculation is done in previous section 4.3.3 by using data files from the ports. This calculated data can be displayed on screen or may be represented in the form of graph. To perform the graphical representation this module is written in menu driven form so selection of the graph is easier. The flow chart of the display module is shown in Fig. 4.4.

The menu driven software has the following options.

1. Speed versus voltage.
2. Speed versus slip
3. Speed versus line current
4. Speed versus power factor.
5. Speed versus output power
6. Speed versus power loss
7. Speed versus torque
8. Speed versus horse power
9. Speed versus efficiency
10. Print the data on screen
11. Print the data on printer
12. Exit

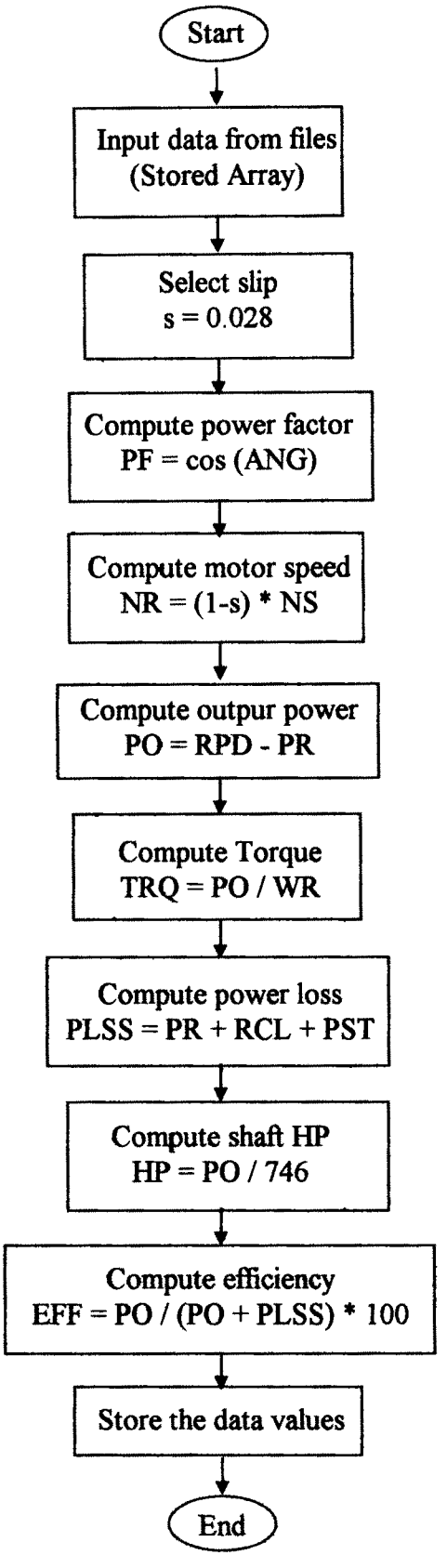

Fig. 4.3 : Flow Chart : Computational Module.

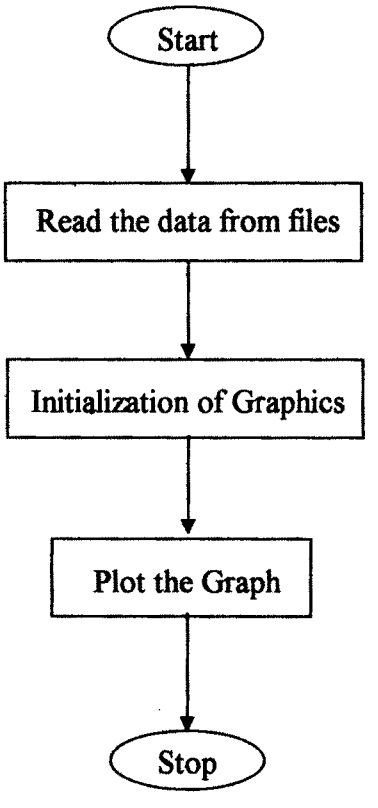

Fig. 4.4 : Flow Chart : Display Module

The desired plot can be seen on the monitor screen by typing the appropriate menu number from the keyboard. The hard copy of this plot is possible from the printer. One can return to main menu by selecting menu choice 12.

4.4 CALIBRATION OF THE SYSTEM

In data acquisition and control system, calibration of the system is very essential to maintain the accuracy of the system. While system calibration it is necessary to have a precision meters and a stable and noise free DC voltage source. In the present data acquisition system there are three variables, namely supply voltage to the motor, stator current and speed of the motor. For the measurement of the speed the digital system is used starting from the speed sensing, so no calibration is required for the speed measurement in the present system.

4.4.1 Voltage Calibration

For the voltage sensing precision rectifier is used. The input voltage to the rectifier is 3 V AC. The variation in the primary voltage of step-down transformer gives respective changes in secondary voltage. The change in secondary voltage gives respective change in output voltage of the rectifier. The Fig. 4.5 shows the variation of rectifier output voltage with input line voltage. From the figure it is seen that output voltage of the rectifier varies linearly with line voltage. To get the actual value of the line voltage, some correction factor must be introduced in the software. The equation for the correction is as follows.

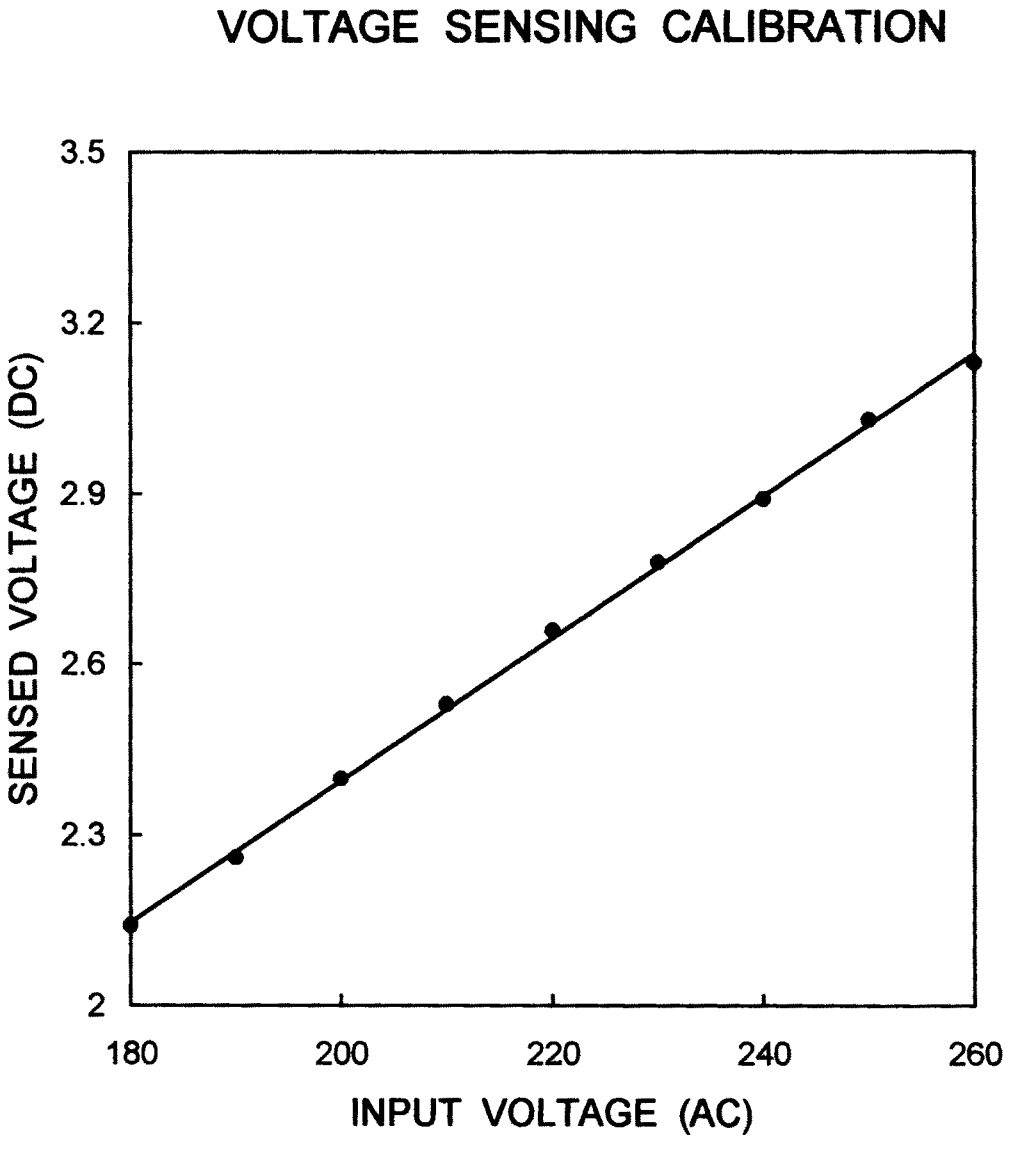

Fig. 4.5 : Input voltage versus sensed voltage

$$\text{Voltage} = (\text{DH \%} * 256 + \text{DL \%} - 2048) * - 66387 \qquad (4.1)$$

The plot of corrected output voltage versus input is shown in Fig. 4.6. This plot shows that output voltage varies linearly with input voltage. The slope of this graph is unity which reveals that input voltage is equal to the output voltage.

4.4.2 Current Calibration

The stator current measurement in case of induction motor under working environment is very important. For current measurement we have used very simple techniques as discussed in Chapter III. The output voltage of the amplifier is proportional to the input current. Calibration for the current measurement is done for different current ranges in order to get the correct results. Calibration curves are plotted for the current ranges 0-1 Amp, 0-2 Amp, 0-3 Amp and 0-5 Amp. The plot for calibration in the range of 0-1 Amp is shown in Fig. 4.7 which shows that actual line current is equal to the measured current. The plot of sensed current versus actual current is shown in Fig. 4.8. and Fig. 4.9 for current ranges 0-2 Amp and 0-3 Amp respectively. It can be conclude that actual current and measured current are nearly same. However, from the Fig. 4.10 it is seen that relation between the sensed current and measured current is not quite linear and correction during measurement is required. This non linearity may be because of the resistance of the shunt. By changing the shunt this problem can be solved. Presently the following relation is used to determine the exact current value.

$$\text{Current} = (2046 - \text{DTA \%}) * (- 40 / 10{,}000) \qquad (4.2).$$

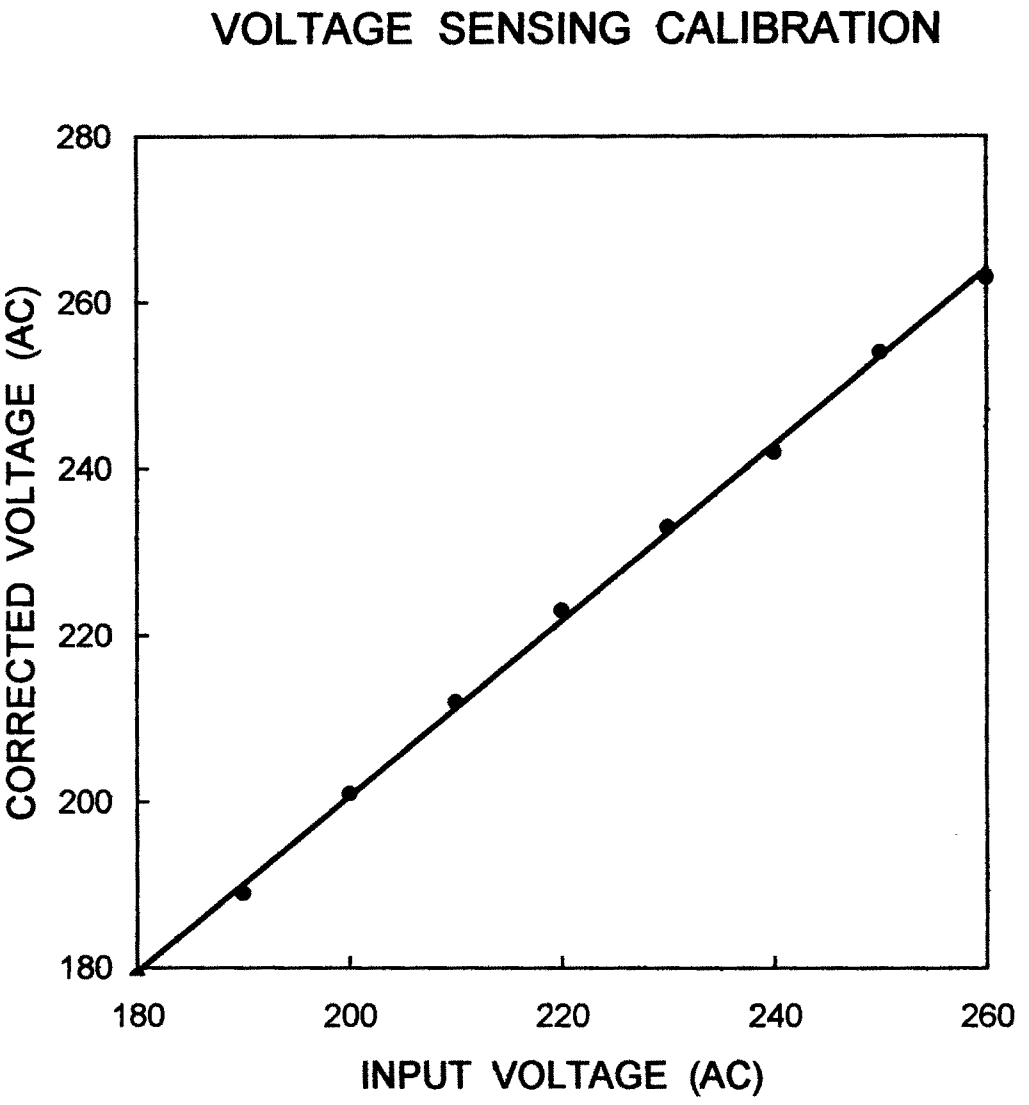

Fig. 4.6 : Input voltage versus corrected voltage

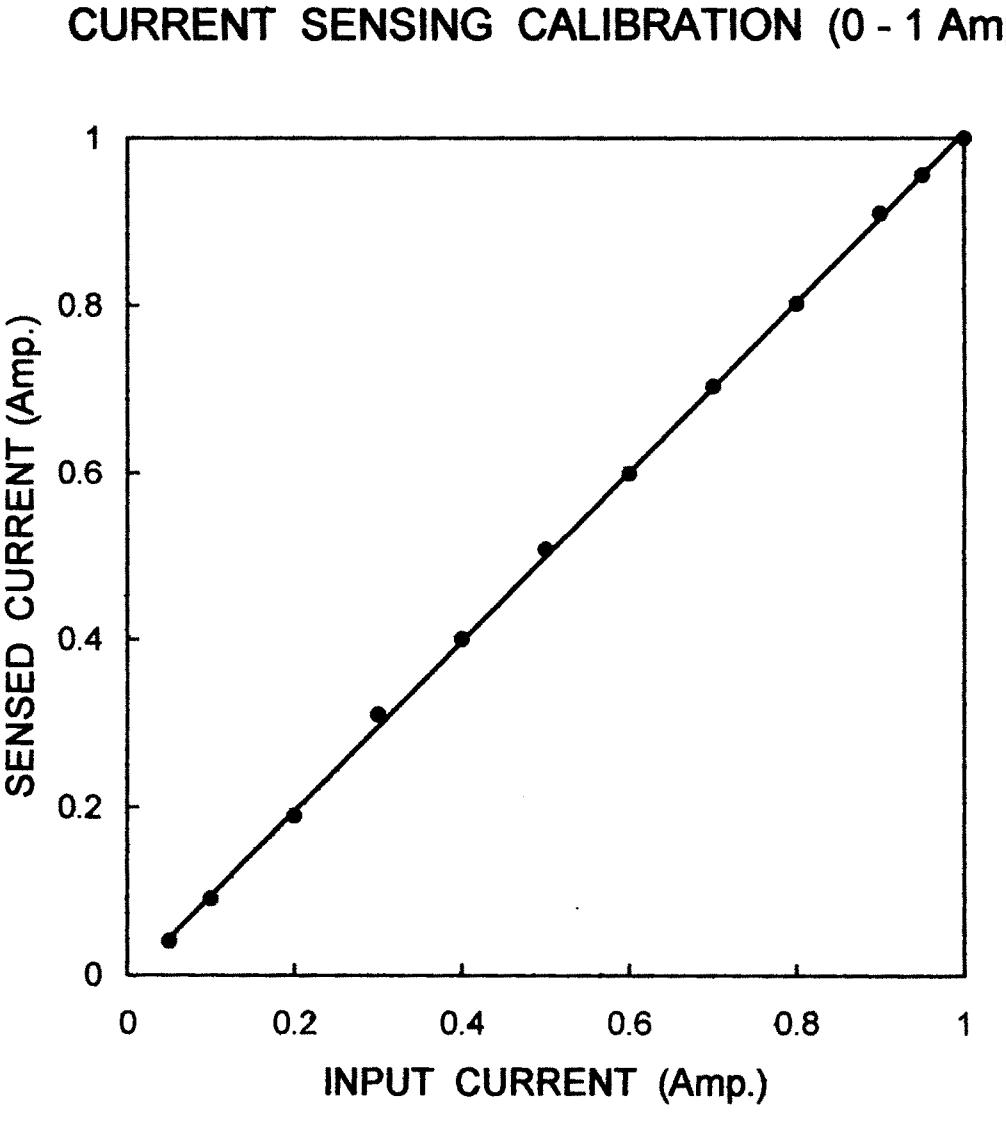

Fig. 4.7 : Input current versus sensed current

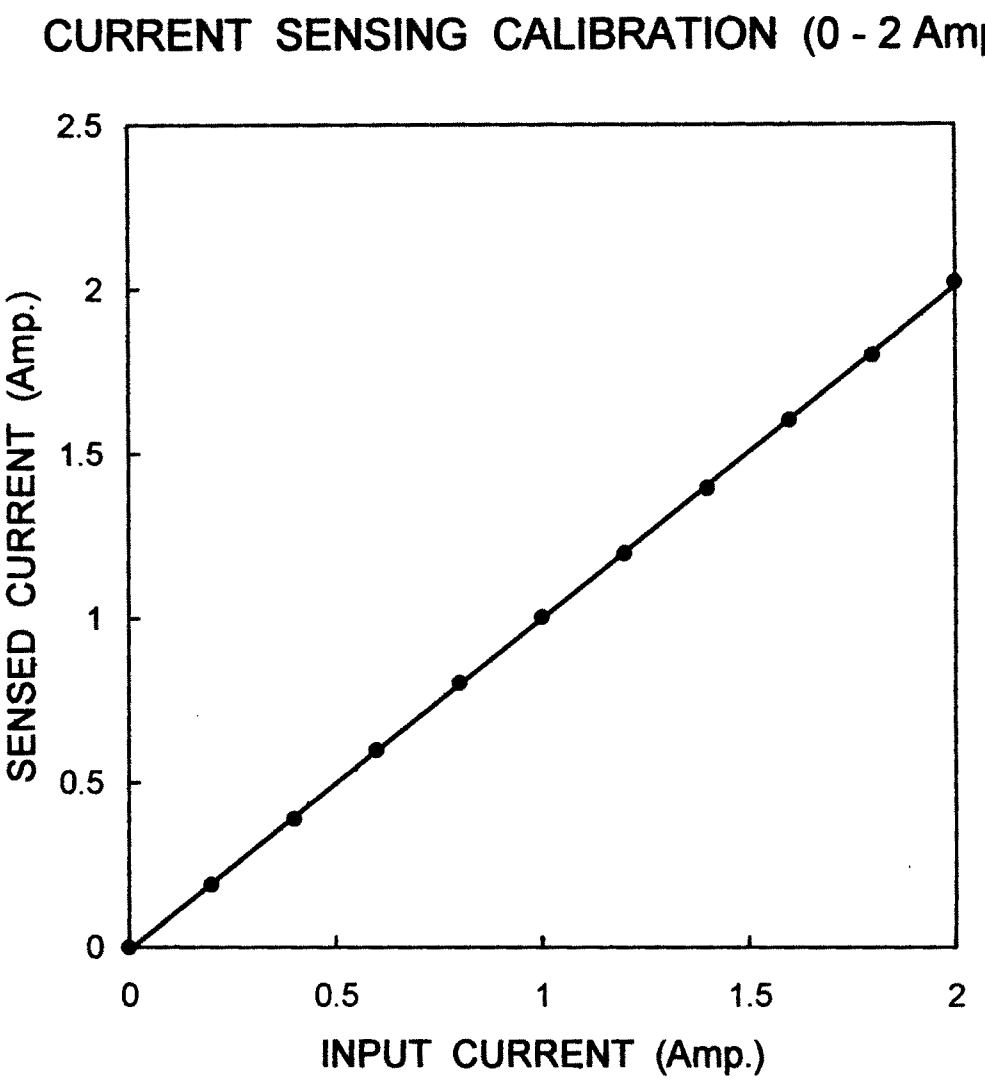

Fig. 4.8 : Input current versus sensed current

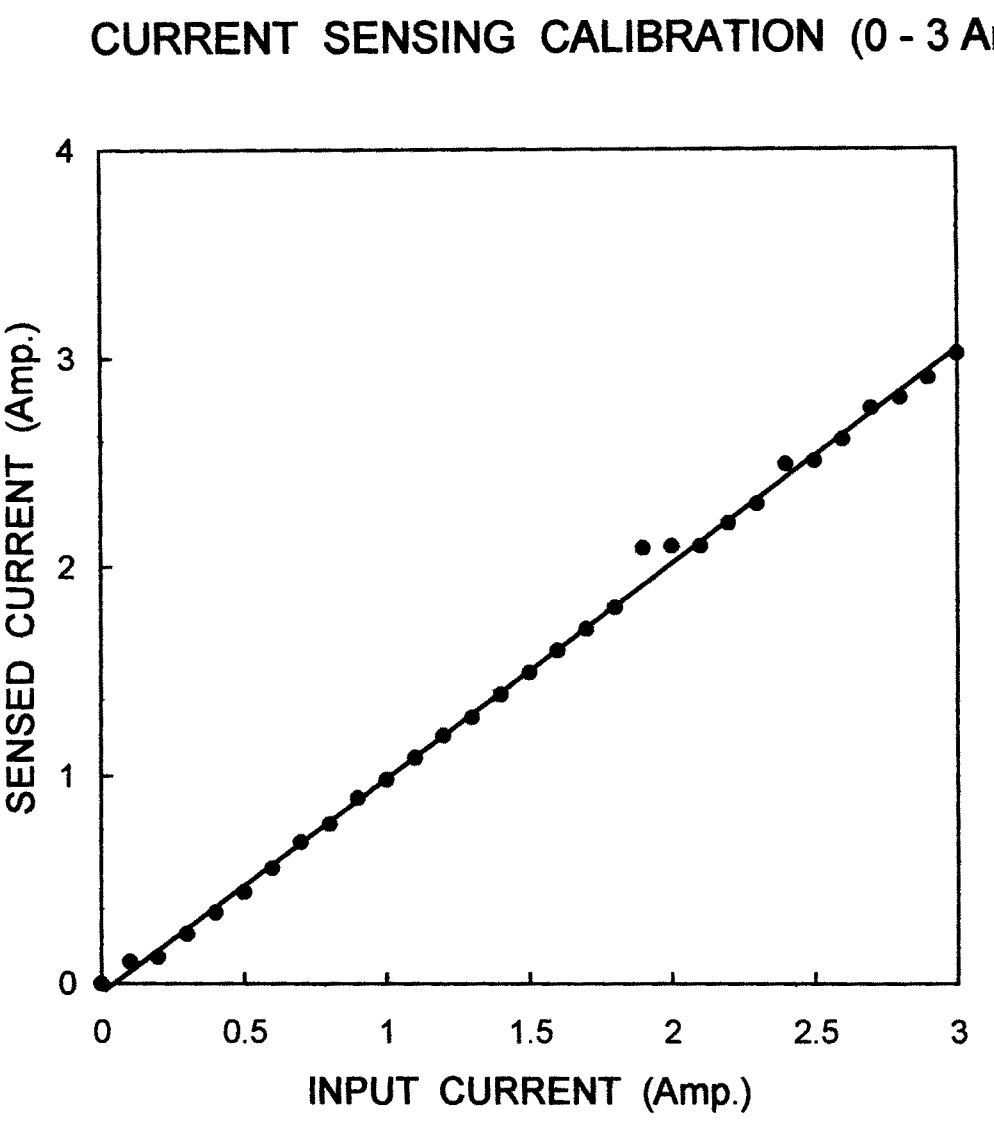

Fig. 4.9 : Input current versus sensed current

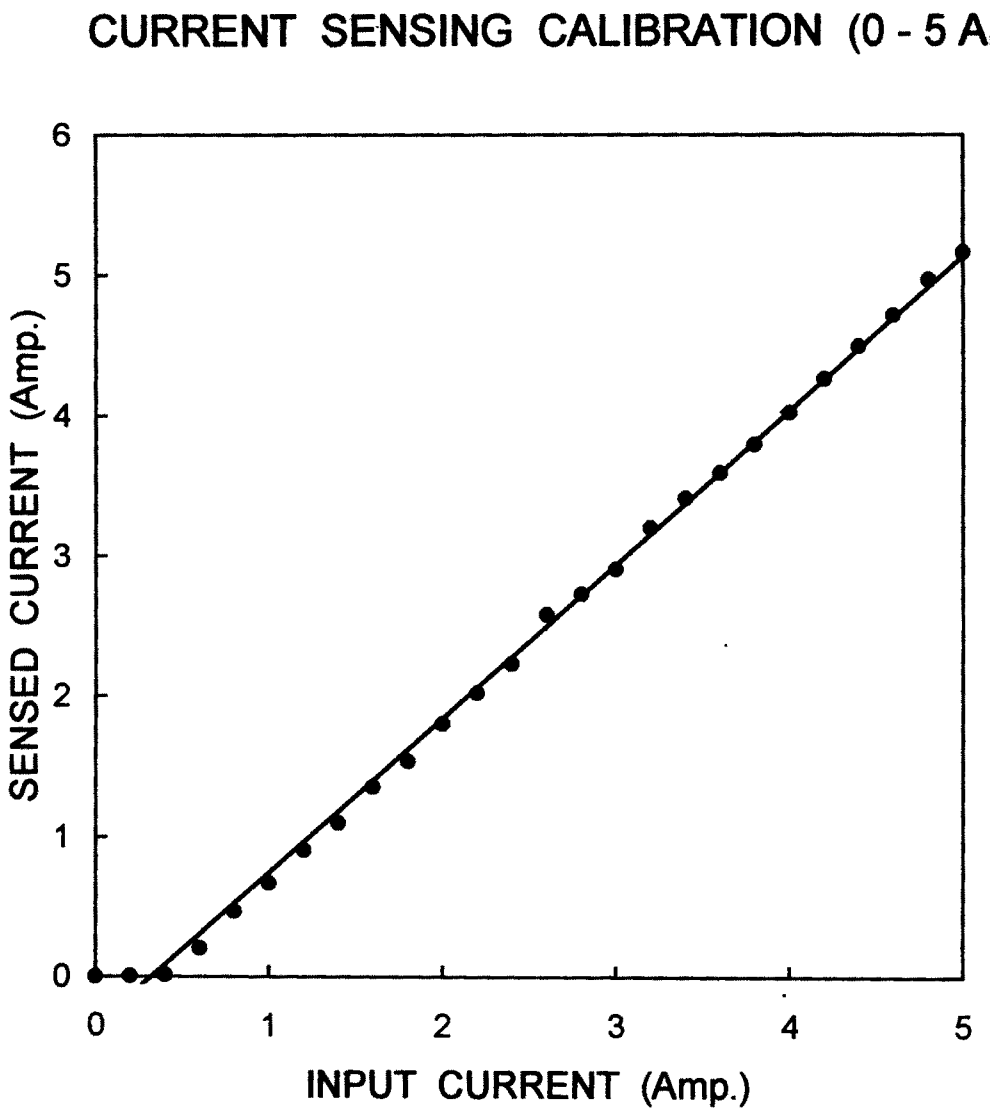

Fig. 4.10 : Input current versus sensed current

4.5 DETERMINATION OF MOTOR PARAMETERS

1) Select A/D channel 0 for speed sensing. DEMO software notes the starting and ending time. Time delay is calculated for 500 samples. Rotation per minute can be calculated by the formula

RPM = COUNT / (YE -YS) * 60

This is value of NS

2. For a fixed value of slip, rotor speed can be calculated as

NR = (1-S) * NS

3. To sense current select A/D channel 1 note the values of line current CL for 500 samples.

4. Power factor can be calculated by writing software assuming magnetizing reactance XM, stator reactance XS, rotor reactance XPR constant by the formula

PF = cos (ANG)

5. Output power = Rotor power developed - Rotational losses

PO = RPD - PR

Here, rotational losses PR, rotor resistance RPR are assumed constant. Then output power varies according to the input current.

Power loss = Rotational loss + Rotor copper loss + Stator copper loss.

PLSS = PR + RCL + PST.

Rotational losses PR are assumed constant.

7. $$\text{Shaft Torque} = \frac{\text{Output power}}{WR} = \frac{PO}{WR}$$

where, WR = 2 * PI * NR / 60

NR - is the rotor speed.

8. $$\text{Shaft Horse Power} = \frac{\text{Power output}}{746}$$

HP = PO / 746

9. Efficiency = Output power / (Output power + power loss) * 100

EFF = PO / (PO + PLSS) * 100

10. The values of NS, CL and V are noted for different load conditions. Values of various parameters are calculated using above relations and these values are then used for plotting the graphs on the monitor.

COMPUTER BASED ON LINE MONITORING OF INDUCTION MOTOR

SOFTWARE DEVELOPED BY : V.P.PAWAR

GUIDE:Dr.M.D.UPLANE

PLACE:DEPT.OF PHYSICS (SUK)

```
10 DIM DTA%(600),A(100,10),X(100),Y(100)
20 REM interfacing,calculation,data storage in #1 & #2
30 REM "DATA.DAT" TO STORE THE ADC DATA
40 REM "RPM.DAT" TO STORE RPM
50 REM "CAL.DAT" TO STORE CALCULATED VALUES
60 OPEN "data.dat" FOR APPEND AS #1
70 OPEN "RVC.DAT" FOR APPEND AS #2
80 OPEN "CAL.DAT" FOR APPEND AS #3
90 'GOTO 210
100 CLS:KEY OFF
110 SCREEN 1
120 FOR U=0 TO 11
130 'SOUND 1500,3:SOUND 1700,4
140 LOCATE 1+U,15:PRINT "graph plotter"
150 LOCATE 23-U,15:PRINT "graph plotter"
160 FOR I=1 TO 40:NEXT I : REM time loop
170 LOCATE 1+U,15:PRINT "              "
180 LOCATE 23-U,15:PRINT "              "
190 NEXT U
200 LOCATE 12,15:PRINT "GRAPH PLOTTER "
210 LINE (100,75)-(220,75)
220 LINE (100,105)-(220,105)
230 LINE (100,75)-(100,105):LINE (220,75)-(220,105)
240 LOCATE 23,15:PRINT " BY PAWAR    "
250 FOR T=1 TO 500
260 NEXT T : REM time loop
270 REM main program
280 REM generating graph sheet and offering options
290 CLS
300 SCREEN 2
```

```
310 LOCATE 7,15:PRINT "*****  MAIN MENU *****"
320 LOCATE 9,15:PRINT "---------------------"
330 LOCATE 11,15:PRINT "1. DATA ACCES FROM ADC"
340 LOCATE 12,15:PRINT "2. DATA OUTPUT & GRAPH PLOTTING"
350 LOCATE 13,15:PRINT "3. DELETE OLD DATA"
360 LOCATE 14,15:PRINT "4. EXIT"
370 LOCATE 16,15:INPUT "enter your choice:";C
380 IF C<1 OR C>4 THEN GOTO 400
390 ON (C) GOTO 1330,1760,3230,3200
400 BEEP :CLS:GOTO 330
410 DIM DTA%(1000)
420 DIM DTV%(1000)
430 PORT%=&H220                    'GET I/O PORT BASE ADDRESSS
440 CLS : KEY OFF: SCREEN 3
450 'VIEW (0,0)-(319,199),3,1
460 'VIEW (56,8)-(265,35),1,1
470 LOCATE 3,10 : PRINT " SENSING CONTROL MENU"
480 'VIEW (56,55)-(265,150),1,1
490 LOCATE 9,8:  PRINT "   1 - SPEED SENSING"
500 LOCATE 12,8 :PRINT "   2 - CURRENT SENSING"
510 LOCATE 15,8 :PRINT "   3 - VOLTAGE SENSING"
520 LOCATE 18,8 :PRINT "   4 - QUIT"
530 PRINT :PRINT :PRINT :PRINT
540 'VIEW (40,162)-(280,185),1,1
550 LOCATE 22,8 :INPUT "PLEASE CHOOSE YOUR OPTION ",N$
560 IF N$ = "1" THEN BEEP : GOTO 610
570 IF N$ = "2" THEN BEEP : GOTO 960
580 IF N$ = "3" THEN BEEP : GOTO 1100
590 IF N$ = "4" THEN BEEP : BEEP
600 BEEP : GOTO 440
610 OPEN "SPEED.DAT" FOR APPEND AS #1
620 'STEP1:SET SCAN CHANNEL
630 OUT PORT%+10,2                 'SELECT A/D CHANNEL 0
640 'STEP2:IMPLEMENT SOFTWARE TRIG & PROGRAM TRANSFER
650 Y1$=TIME$
660 FOR LP1=0 TO 500              'GET 500 DATAS
670 OUT PORT%+11,0                 'SOFTWARE TRIG
680 DH%=INP(PORT%+5)               'READ HIGH BYTE DATA
690 IF DH%>15 GOTO 680             'CHECK DRDY READY ?
700 DL%=INP(PORT%+4)               'READ LOW BYTE DATA
710 DTA%(LP1)=DH%*256+DL%-2048   'STORE  A/D DATA INTO ARRAY
720 NEXT LP1
730 Y2$ = TIME$
740 YS$=RIGHT$(Y1$,2)
750 YE$=RIGHT$(Y2$,2)
760 YS=VAL(YS$)
770 YE = VAL(YE$)
```

```
780 COUNT = 0
790 Y = SGN(DTA(0))
800 Y1= Y
810 FOR I=1 TO 500
820 Y = SGN(DTA%(I))
830 IF Y<>Y1 THEN 850
840 GOTO 870
850 COUNT = COUNT+1
860 Y1 = Y
870 NEXT I
880 COUNT=COUNT/2
890 RPM = COUNT/(YE-YS)*60
900 'VIEW (0,0)-(319,199),1,1
910 CLS: LOCATE 12,7 : PRINT "Rotation Per Minute :";RPM
920 A$ = INPUT$(1)
930 PRINT #1,RPM
940 CLOSE #1
950 GOTO 440
960 OPEN "CUR.DAT" FOR APPEND AS #2
970 OUT PORT%+10,4
980 OUT PORT%+11,0
990 FOR I = 1 TO 5000:NEXT I
1000 DH%=INP(PORT%+5)
1010 IF DH%>15 GOTO 1000
1020 DL%=INP(PORT%+4)
1030 DTA%=DH%*256+DL%
1040 PRINT #2,DTA%/1
1050 'VIEW (0,0)-(319,199),1,1
1060 LOCATE 12,12 : PRINT "CURRENT = ";(2046-DTA%)*(-40/10000);"
     Amp"
1070 A$ = INPUT$(1)
1080 CLOSE #2
1090 GOTO 440
1100 OPEN "VOL.DAT" FOR APPEND AS #3
1110 OUT PORT%+10,6
1120 C = 0
1130 FOR LP1=0 TO 500            'GET 500 DATAS
1140 OUT PORT%+11,0               'SOFTWARE TRIG
1150 DH%=INP(PORT%+5)             'READ HIGH BYTE DATA
1160 IF DH%>15 GOTO 1150           'CHECK DRDY READY ?
1170 DL%=INP(PORT%+4)             'READ LOW BYTE DATA
1180 DTV%(LP1)=(DH%*256+DL%-2048) * (-.66389) 'STORE
     A/D  DATA INTO ARRAY
1190 C = C + DTV%(LP1)
1200 NEXT LP1
1210 A = C / 500
1220 PRINT #3,DTV%
```

```
1230 'VIEW (0,0)-(319,199),1,1
1240 LOCATE 12,12 : PRINT "VOLTAGE = ";A;" V"
1250 CLOSE #3
1260 A$ = INPUT$(1)
1270 GOTO 440
1280 PRINT "Rotation Per Minute :";RPM
1290 REM Input Motor data on a per phase basis
1300 REM V=voltage, RS=stator R, RPR=R prime rotor
1310 REM XS=stator X, XPR=X prime rotor, XM=X magnetizing
1320 REM PR=rotational loss,S=slip value
1330 FOR I=1 TO 3
1340 INPUT "INPUT  RPM,VOL,CUR",RPM,V,CL
1350 'V=230
1360 RS=15.2
1370 RPR=4.22
1380 XS=10.2
1390 XPR=5.4
1400 XM=160
1410 PR=10.8
1420 S=.028
1430 A=RPR/S +RS
1440 XE=XS+XPR
1450 B=SQR(A*A +XE*XE)
1460 TH=-ATN(XE/A)
1470 T=V/B
1480 CRE=T*COS(TH)
1490 CIM=T*SIN(TH)
1500 AIM=-V/XM
1510 CLIM=CIM+AIM
1520 'CL=SQR(CRE*CRE +CLIM*CLIM)
1530 ANG=ATN(CLIM/CRE)
1540 PF=COS(ANG)
1550 NS=RPM
1560 NR=(1-S)*NS
1570 PI=3.141593
1580 WR=2*PI*NR/60
1590 RPI=T*T*RPR/S
1600 RPD=RPI*(1-S)
1610 PO=RPD-PR
1620 TRQ=PO/WR
1630 HP=PO/746
1640 REM calculate the efficiency,EFF
1650 RCL=S*RPI
1660 PST=CL*CL*RS
1670 PLSS=PR+RCL+PST
1680 EFF=PO/(PO+PLSS)*100
1690 PRINT
1700 PRINT #2,RPM,V,CL
1710 PRINT #3,NR,V,S,CL,PF,PO,PLSS,TRQ,HP,EFF
1720 PRINT NR,V,S,CL,PF,PO,PLSS,TRQ,HP,EFF
1730 NEXT I
```

```
1740 CLOSE ALL:GOTO 290
1750 'CLS :
1760  CLOSE  #1,#2,#3:'LOCATE 18,20:PRINT  "select  one      (1)
     small Circle around the point"
1770 'LOCATE 19,20:PRINT "                (2) Small cross on  the
     point     "
1780  'LOCATE 20,20:INPUT "                (3) or only the  point
     ",N
1790 'IF N>3 THEN GOTO 1460
1800 'IF N<1 THEN GOTO 1460
1810 'GOTO 1470
1820 'BEEP:LOCATE  21,20:PRINT  "enter any one out  of  1,2
    or 3":GOTO 1420
1830 CLS
1840 'DIM A(20,9)
1850 OPEN "CAL.DAT" FOR INPUT AS #2
1860 I1=0
1870 IF EOF(2) THEN 1910
1880 I1=I1+1
1890 INPUT #2,A(I1,1),A(I1,2),A(I1,3),A(I1,4),A(I1,5),A(I1,6),
    A(I1,7),A(I1,8),A(I1,9),A(I1,10):GOTO 1870
1900 PRINT A(I1,1),A(I1,2),A(I1,3),A(I1,4),A(I1,5),A(I1,6),
    A(I1,7),A(I1,8),A(I1,9),A(I,10):GOTO 1870
1910 CLOSE #2:PRINT I:INPUT "VVK",LL
1920 LOCATE 5,20:PRINT "DATA OUTPUT & GRAPH MENU"
1930 LOCATE 6,20:PRINT "================================="
1940 LOCATE 7,20:PRINT "1. VOLTAGE VS SPEED"
1950 LOCATE 8,20:PRINT "2. speed vs slip"
1960 LOCATE 9,20:PRINT "3. speed vs line current"
1970 LOCATE 10,20:PRINT "4. speed vs power factor"
1980 LOCATE 11,20:PRINT "5: speed vs output power"
1990 LOCATE 12,20:PRINT "6. speed vs power loss"
2000 LOCATE 13,20:PRINT "7: speed vs torque
2010 LOCATE 14,20:PRINT "8: speed vs horse power"
2020 LOCATE 15,20:PRINT "9.SPEED VS EFFICIENCY"
2030 LOCATE 16,20:PRINT "10.PRINT THE DATA ON SCREEN"
2040 LOCATE 17,20:PRINT "11.PRINT  THE DATA ON PRINTER"
2050 LOCATE 18,20:PRINT "12.EXIT"
2060 LOCATE 20,20:INPUT "enter your choice :",C
2070 IF C>12 OR C<1 THEN 1920
2080 B$="speed"
2090 FOR I=1 TO I1
2100 X(I)=A(I,1)
2110 NEXT I
2120 'FOR I=1 TO I1
2130 'PRINT X(I)
2140 'NEXT I
```

```
2150 'INPUT "ANY KEY";YY
2160 IF C=1 THEN GOSUB 2940
2170 IF C=2 THEN GOSUB 2950
2180 IF C=3 THEN GOSUB 2960
2190 IF C=4 THEN GOSUB 2970
2200 IF C=5 THEN GOSUB 2980
2210 IF C=6 THEN GOSUB 2990
2220 IF C=7 THEN GOSUB 3000
2230 IF C=8 THEN GOSUB 3010
2240 IF C=9 THEN GOSUB 3020
2250 CLS : PRINT "wait" :FOR I=1 TO 1000 :NEXT I:CLS
2260 ON (C) GOTO 2280,2290,2300,2310,2320,2330,2340,
     2350,2360,3030,3110,2270
2270 I1=0:GOTO 290
2280 VV=1 :GOTO 2370
2290 VV=2 :GOTO 2370
2300 VV=3 :GOTO 2370
2310 VV=4 :GOTO 2370
2320 VV=5 :GOTO 2370
2330 VV=6 :GOTO 2370
2340 VV=7 :GOTO 2370
2350 VV=8 :GOTO 2370
2360 V=9:GOTO 2370
2370 FOR I=1 TO I1
2380 Y(I)=A(I,VV)
2390 NEXT I
2400 CLS
2410 L1=X(1):L2=Y(1)
2420 FOR I=2 TO I1
2430 IF L1<X(I) THEN L1=X(I)
2440 IF L2<Y(I) THEN L2=Y(I)
2450 NEXT I
2460 L=LEN(B$):P=(80-L)/2
2470 'FOR I=1 TO L
2480 LOCATE 25,P :PRINT B$
2490 'PRINT MID$(B$,I,1)
2500 'P=P+1
2510 'NEXT I
2520 'L=LEN(A$):P=(25-L)/2
2530 'FOR I=1 TO L
2540 'LOCATE P,1 :PRINT MID$(A$,I,1)
2550 'P=P+1
2560 'NEXT I
2570 SCREEN 2
2580 LOCATE 25,60:PRINT "ANY KEY..."
2590 'LINE(35,7)-(639,172),1,B
2600 'SCREEN 2
```

```
2610 CX=0 :CY=185
2620 C=L1-SL
2630 SX=590/L1:SY=170/L2
2640 X1=CX+SX*X(1):Y1=CY-SY*Y(1):CIRCLE(X1,Y1),4,1
     :PAINT(X1,Y1),1
2650 FOR I=2 TO I1
2660 X=CX+SX*X(I) :Y=CY-SY*Y(I)
2670 CIRCLE(X,Y),4,1:PAINT(X,Y),1
2680 LINE(X1,Y1)-(X,Y)
2690 X1=X:Y1=Y
2700 NEXT I
2710 'SCREEN 2
2720 X=L2:Y=L1
2730 LINE(22,8)-(620,8):LINE(620,8)-(620,171)
2740 'LINE(50,1)-(50,160):LINE-(624,164)
2750 C=X/10
2760 J=22
2770 FOR I=0 TO 10
2780 A=C*I
2790 LOCATE J,1
2800 PRINT A:LOCATE J,7:PRINT "-"
2810 J=J-2
2820 NEXT I
2830 G=Y/10
2840 FOR I=1 TO 10
2850 B=G*I
2860 LOCATE 23,(I+1)*6.4:PRINT B
2870 LOCATE 22,(I+1)*6.6  :PRINT "+"
2880 LINE(50,9)-(50,171):LINE-(622,171)
2890 NEXT I
2900 LOCATE 1,30:PRINT"GRAPH OF  "A$"  vs  " B$
2910 LOCATE 1,5:PRINT A$
2920 IF INKEY$="" THEN 2920
2930 CLS:GOTO 1920
2940 A$="VOLTAGE":RETURN
2950 A$="SLIP": RETURN
2960 A$="LINE CURRENT" :RETURN
2970 A$="POWER FACTOR" :RETURN
2980 A$="OUTPUT POWER" :RETURN
2990 A$="POWER LOSS" :RETURN
3000 A$="TORQUE" :RETURN
3010 A$="HORSE POWER":RETURN
3020 A$="EFFICIENCY" :RETURN
3030 CLS:I=1
3040 PRINT "NR S CL PF PO PLSS TRQ HP EFF"
3050 WHILE I <=I1
```

```
3060 PRINT A(I,1);A(I,2);A(I,3);A(I,4);A(I,5);A(I,6);
     A(I,7);A(I,8);A(I,9),A(I,10)
3070 I=I+1
3080 WEND
3090 INPUT "PRESS ANY KEY ..";XX:CLS
3100 GOTO 1920
3110 CLS:I=0
3120 LPRINT "NR S CL PF PO PLSS TRQ HP EFF"
3130 WHILE I <I1
3140 CLS:LOCATE 12,40:PRINT "WAIT WORKING":I=I+1
3150 LPRINT A(I,1),A(I,2),A(I,3),A(I,4),A(I,5),A(I,6),
     A(I,7),A(I,8),A(I,9),A(I,10)
3160 PRINT
3170 WEND
3180 'INPUT "PRESS ANY KEY ..";XX
3190 CLS:GOTO 1920
3200 CLS
3210 LOCATE 12,30:PRINT "*** THANK YOU ***
3220 END
3230 CLS:LOCATE 12,40:PRINT "Are you sure ?(Y?N):"
3240 X$=INKEY$
3250 IF X$="" THEN 3230
3260 IF (X$="Y") OR (X$="y") THEN 3280
3270 GOTO 290
3280 CLOSE #1,#2,#3:KILL"DATA.DAT"
3290 KILL"RVC.DAT"
3300 KILL"CAL.DAT"
3310 GOTO 290
```