BIBLIOGRAPHY

Analog Devices, Inc., Data Acquisition Products Catalog, 57-60, 1978.

Bottomley, P.A., "NMR Imaging Techniques and Applications: A Review," Technical Information Series (General Electric), pp. 1-21, 1981.

Burr-Brown Corporation, Integrated Circuits Data Book, pp. 5-13 - 5-71, 1986.

Chaiwat Laowattanakul, "Programmable Pulse Generator for NMR Imaging," Master thesis, Department of Physics Graduate school, Chulalongkorn University, 1989.

Cho, Z.H., "Computerized Tomography," Encyclopedia of Physical Science and Technology, Vol. 3, pp. 507-544, Academic Press, Inc., 1987.

Cho, Z.H., Ahn, C.B., Juh, S.H., Lee, H.K., Jacobs, R.E., Lee, S., Yi, J.H., Jo, J.M., "Nuclear Magnetic Resonance Microscopy with 4-μm Resolution: Theoretical Study and Experimental Results," Med. Phys., 15(6), 815-824, 1988.

Cho, Z.H., Kim, D.J., Kim, Y.K., "Total Inhomogeneity Correction Including Chemical Shifts and Susceptibility by View Angle Tilting," Med. Phys., 15(1), 7-11, 1988.

Gao, J.H., Holland, S.K., Gore, J.C., "Nuclear Magnetic Resonance Signal from Flowing Nuclei in Rapid Imaging Using Gradient Echoes," Med. Phys., 15(6), 809-814, 1988.

Hitachi Denshi, Ltd., Oscilloscope Operation Manual, pp. 34-37.

Hnatek, E.R., A User's Handbook of D/A and A/D Converters, John Wiley & Sons, Inc., New York, 1976.

Holz, D., Jensen, D., Proksa, R., Tochtrop, M., Vollmann, W., "Automatic Shimming for Localized Spectroscopy," Med. Phys., 15(6), 898-903,1988.

Intel Corporation, Embedded Controller Handbook Volume I 8-Bit, pp.5-1 - 10-396, 1988.

Intel Corporation, MCS BASIC-52 User's Manual, 1988.

Lin, K.S., Digital Signal Processing Applications with the TMS320 Family Vol. 1, Prentice-Hall, Inc., New Jersey, 1987.

Morris, P.G., Nuclear Magnetic Resonance Imaging in Medicine and Biology, Clarendon Press, Oxford, 1986.

Motorola Inc., Schottky TTL Data,, 3rd Printing, 1983.

National Semiconductor Corporation, Linear Databook 1, 1988.

Oppenheim, A.V., Schafer, R.W., Digital Signal Processing, pp. 1-34, Prentice-Hall, Inc., New Jersey, 1975.

Precision Monolithics Inc., Precision Analog Integrated Circuits Databook, pp. 11-2 - 11-35, 1988.

Princeton Applied Research Corporation, Model 4202 Signal Averager Operating and Service Manual, 1978.

Shaw, D., Fourier Transform N.M.R. Spectroscopy, Elsevier Science Publishing Company Inc., 2nd ed., 1984.

Taylor, D.G., Inamdar, R., Bushell, M-C, "NMR Imaging in Theory and in Practice," Phys. Med. Biol., 33(6), 635-670, 1988.

Texas Instruments Inc., TMS32010 Assembly Language Programmer's Guide, 1984.

Texas Instruments Inc., TMS32010 Development Support Reference Guide, 1984.

Texas Instruments Inc., TMS32010 User's Guide, 1984.

Wiwat Sidhisoradej, "Image Processing from NMR Signal," Master thesis, Department of Physics Graduate school, Chulalongkorn University, 1989.

Wood, M.L., Runge, V.M., "Artifacts due to Residual Magnetization in Three-dimensional Magnetic Resonance Imaging," Med. Phys., 15(6), 825-831, 1988.

APPENDIX

Appendix A

Circuit Diagrams

Fig. A-1 Circuit diagram of memory part.

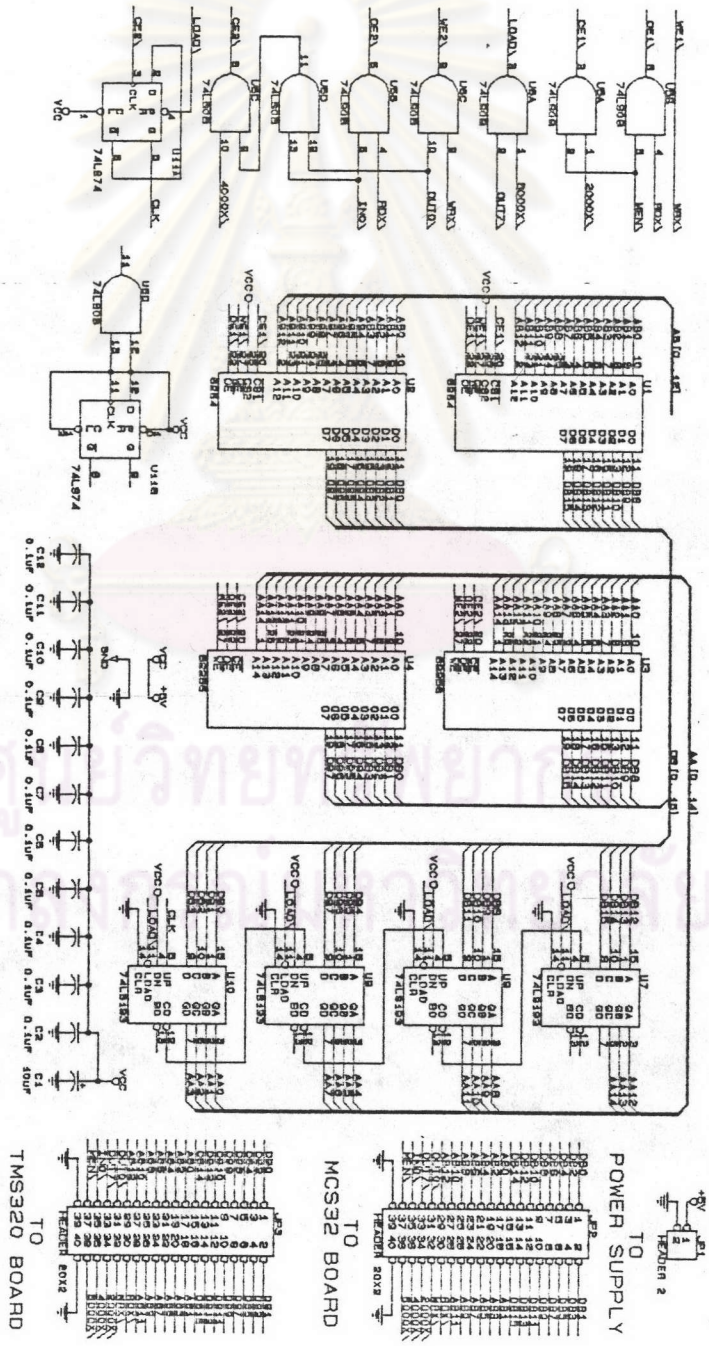


Fig. A-2 Circuit diagram of MCS32 part.

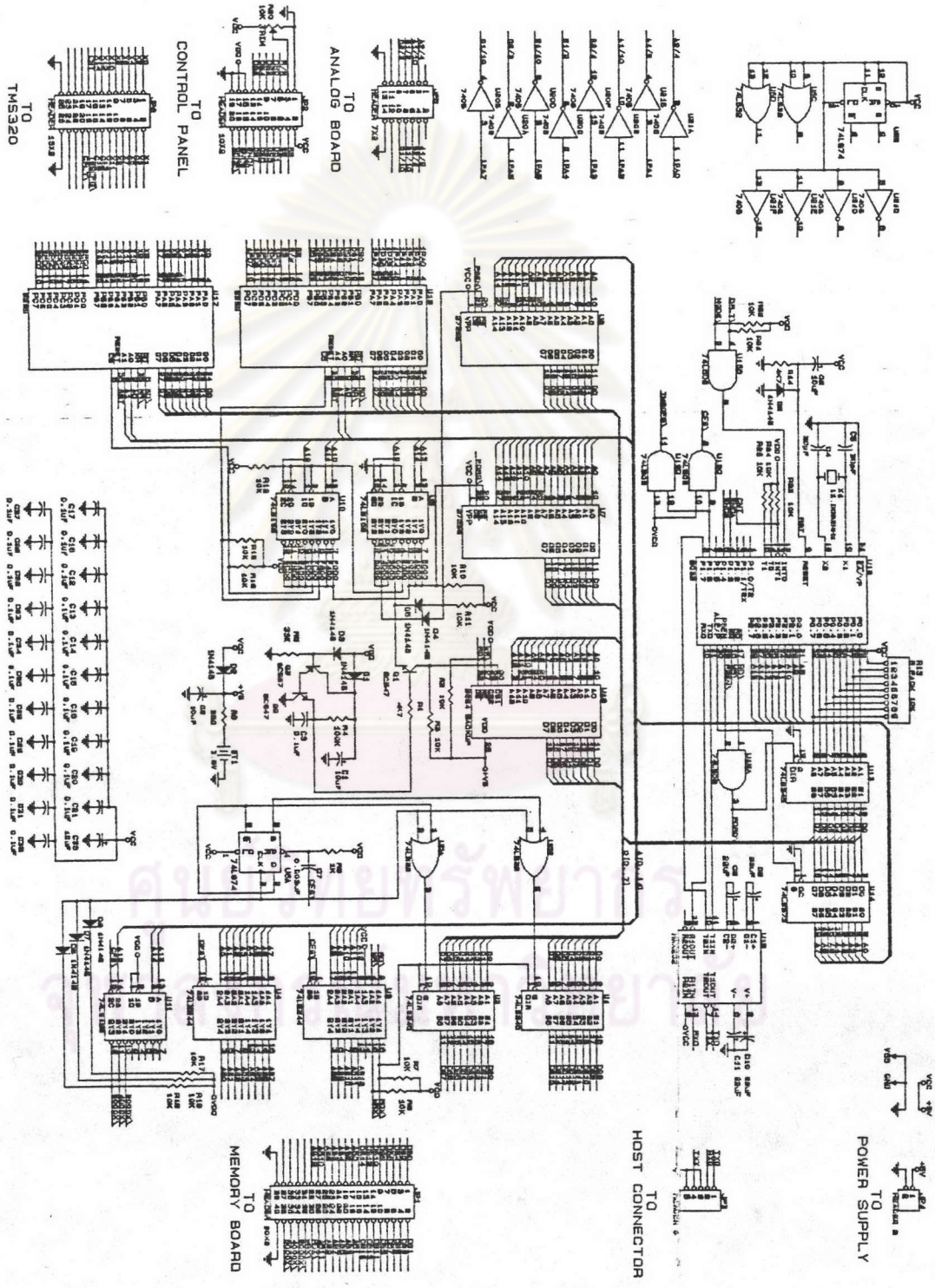


Fig. A-3 Circuit diagram of TMS320 part.

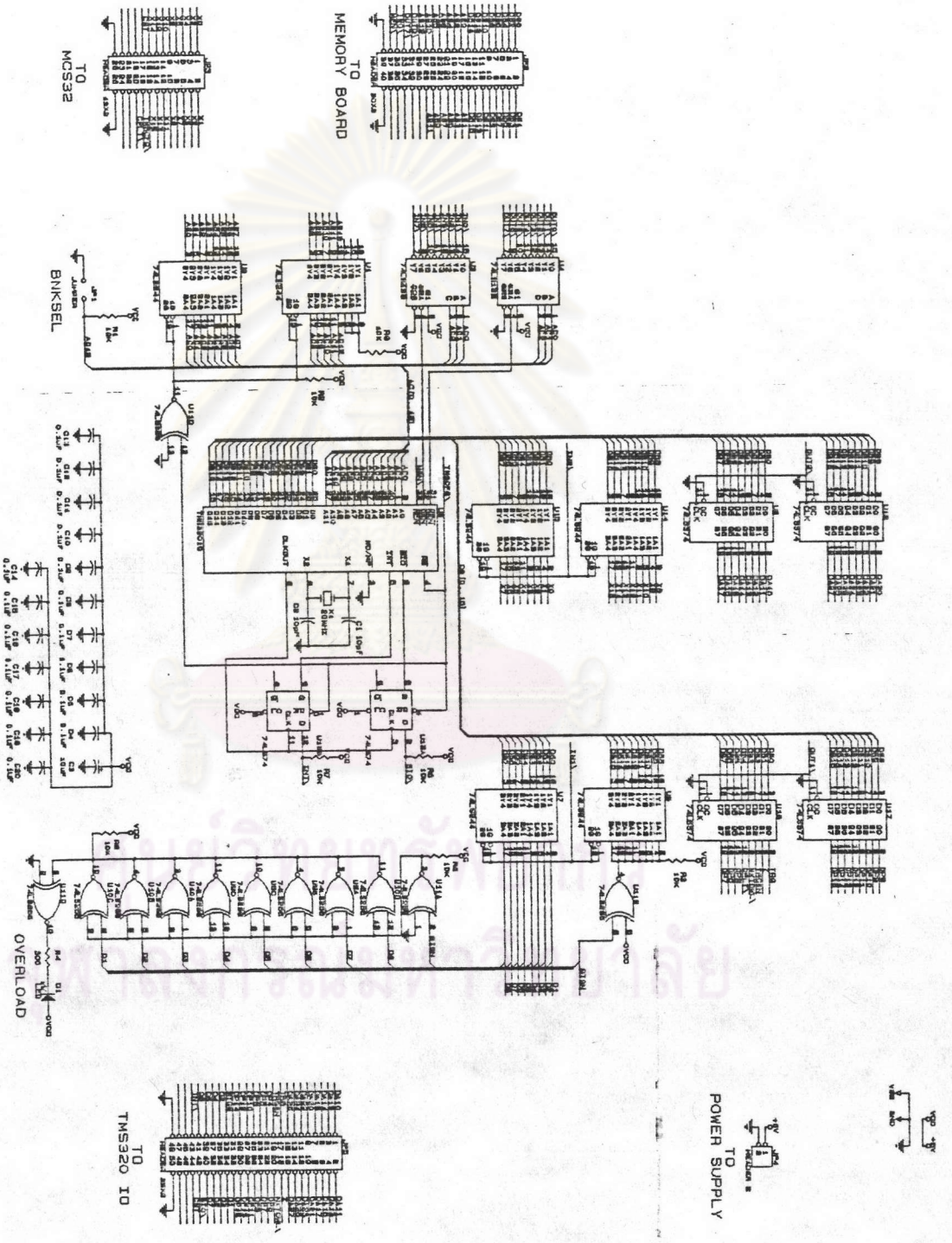


Fig. A-4 Circuit diagram of TMS320 IO part.

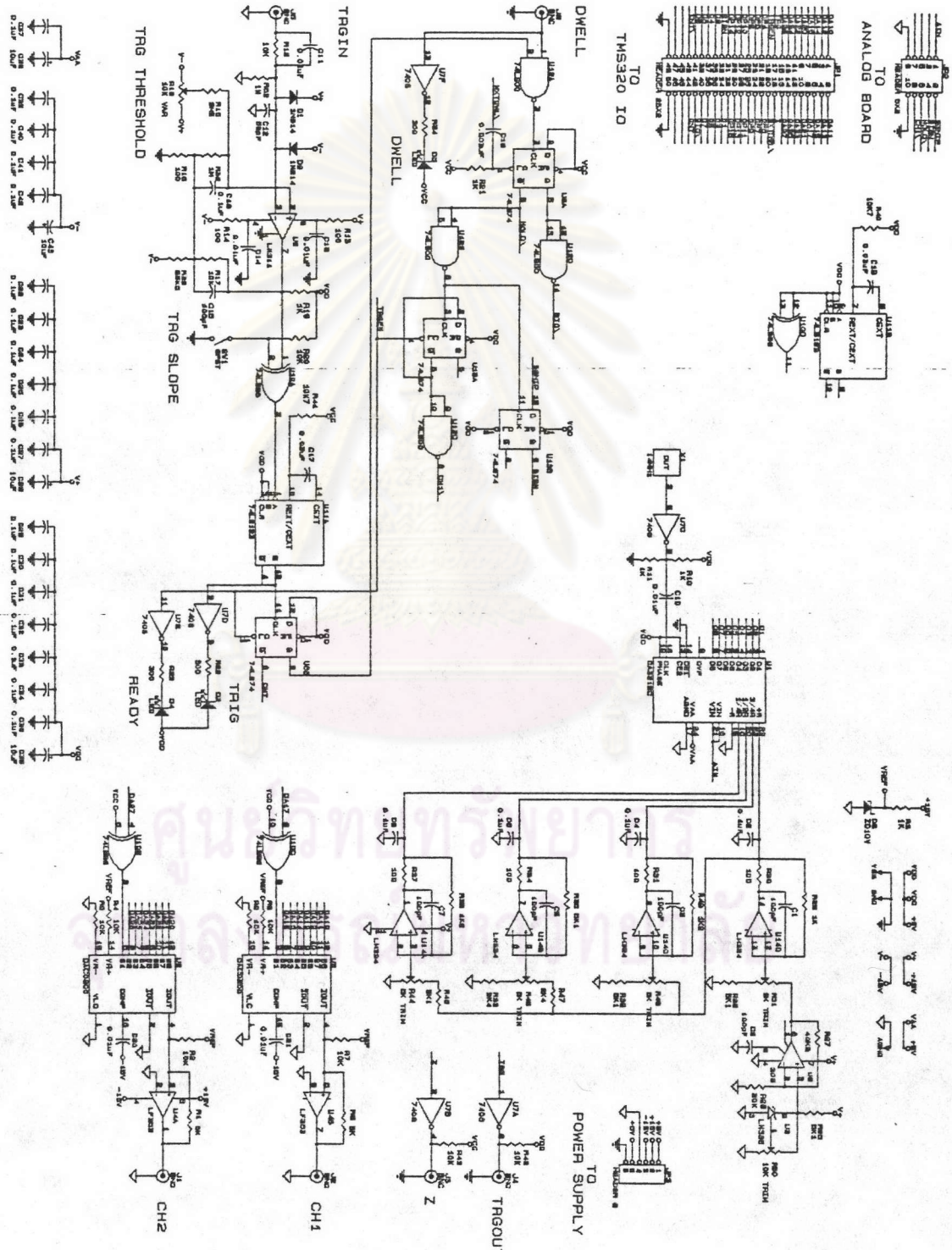
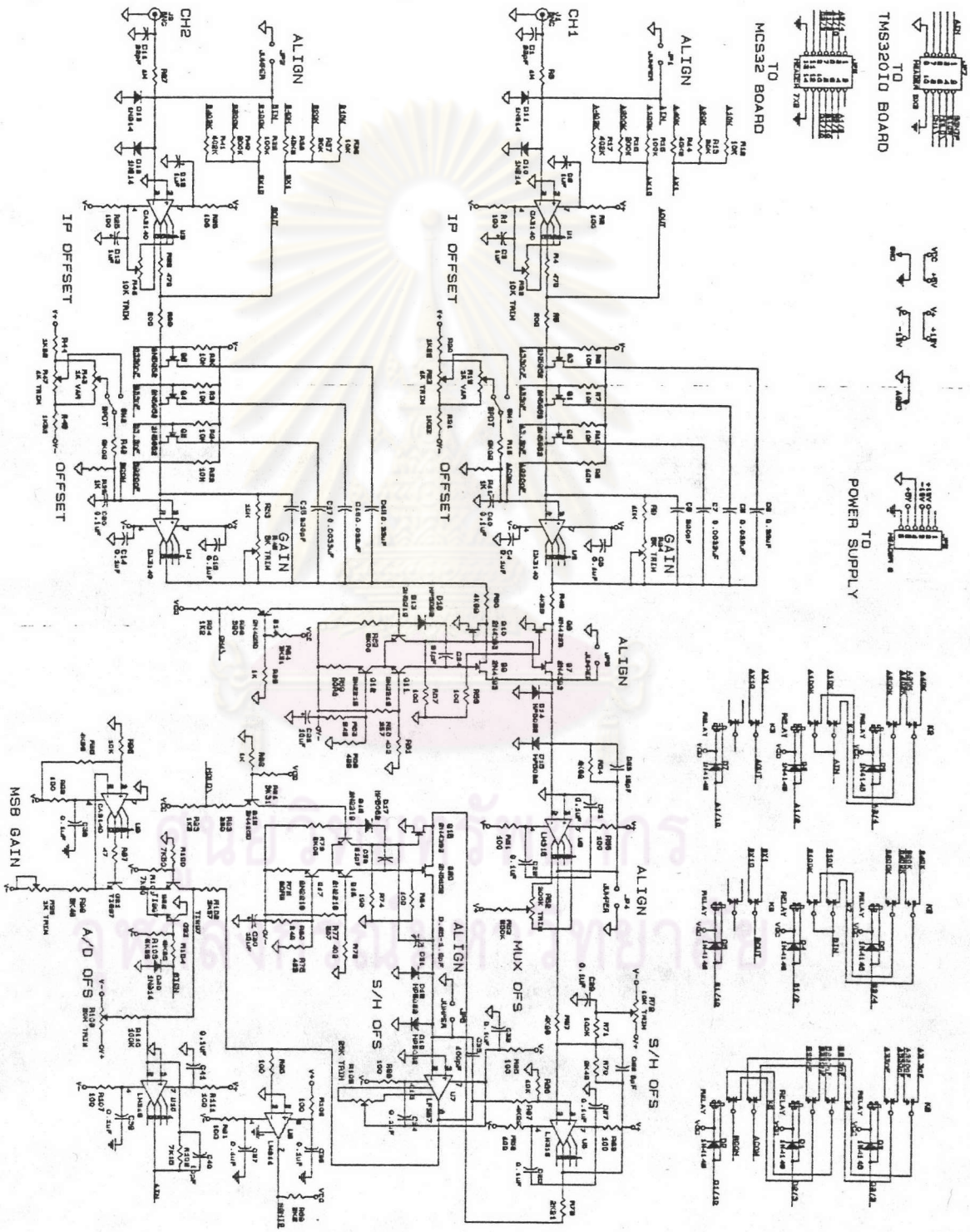


Fig. A-5 Circuit diagram of analog part.



Appendix B

Printed Circuit Board Layouts

Fig. B-1 Component side layout of memory part PCB.

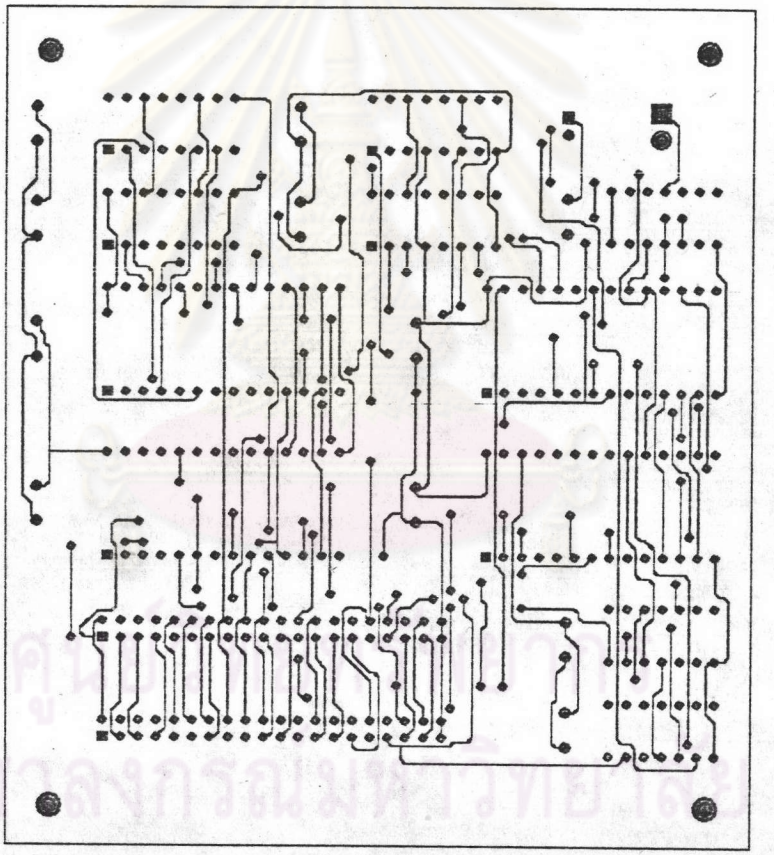


Fig. B-2 Solder side layout of memory part PCB.

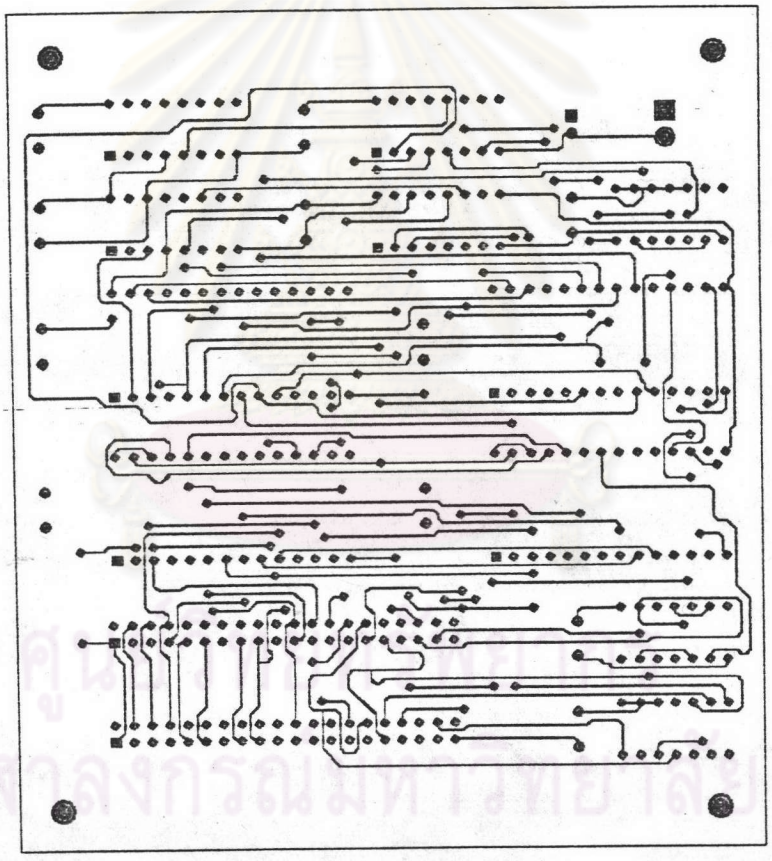


Fig. B-3 Component overlay of memory part PCB.

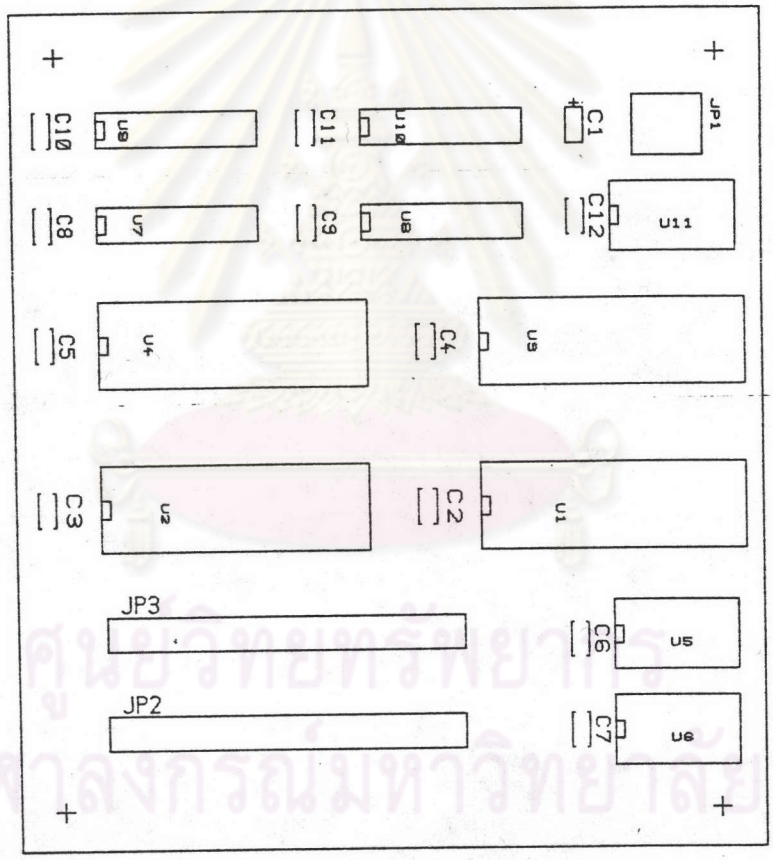


Fig. B-4 Component side layout of MCS32 part PCB.

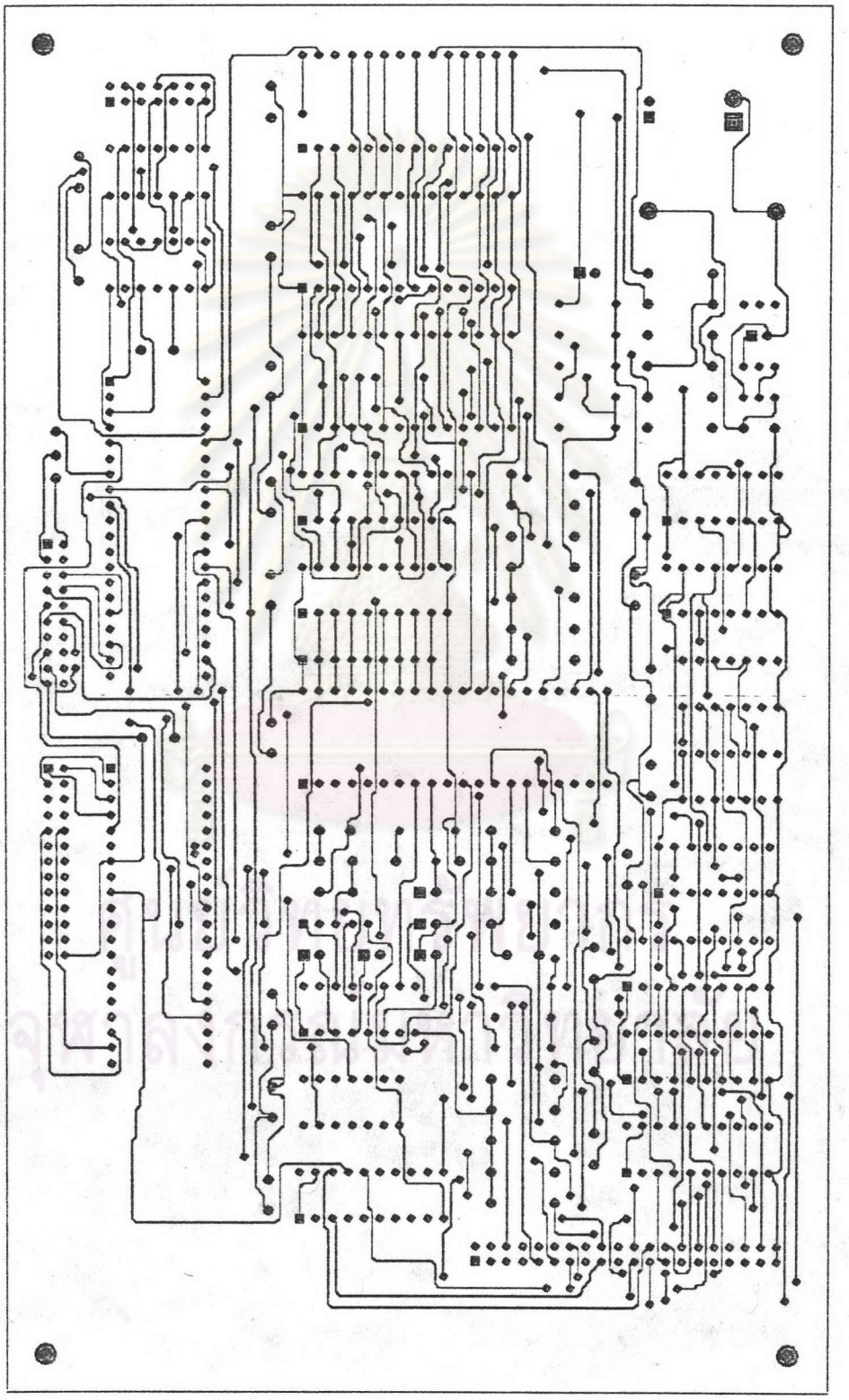


Fig. B-5 Solder side layout of MCS32 part PCB.

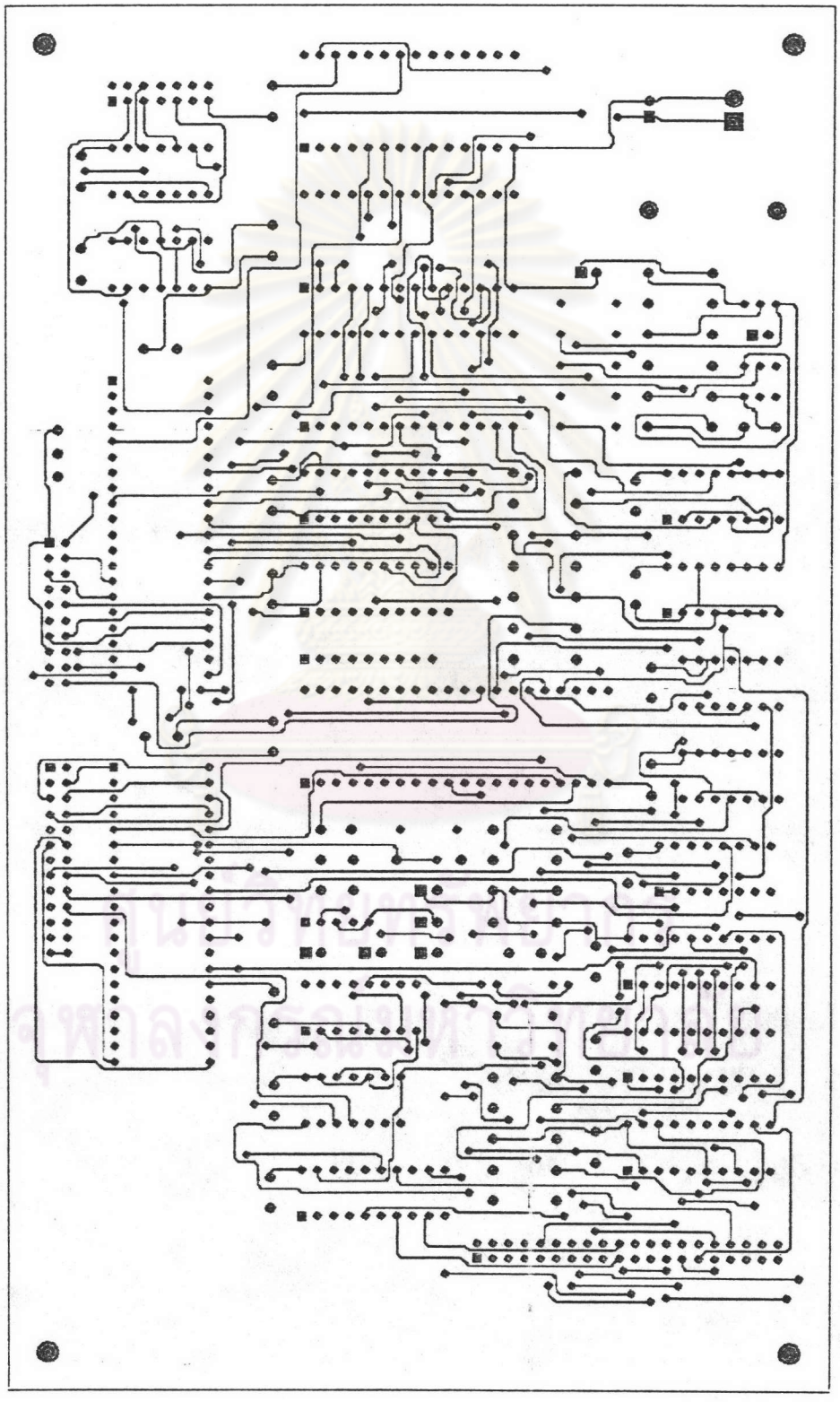


Fig. B-6 Component overlay of MCS32 part PCB.

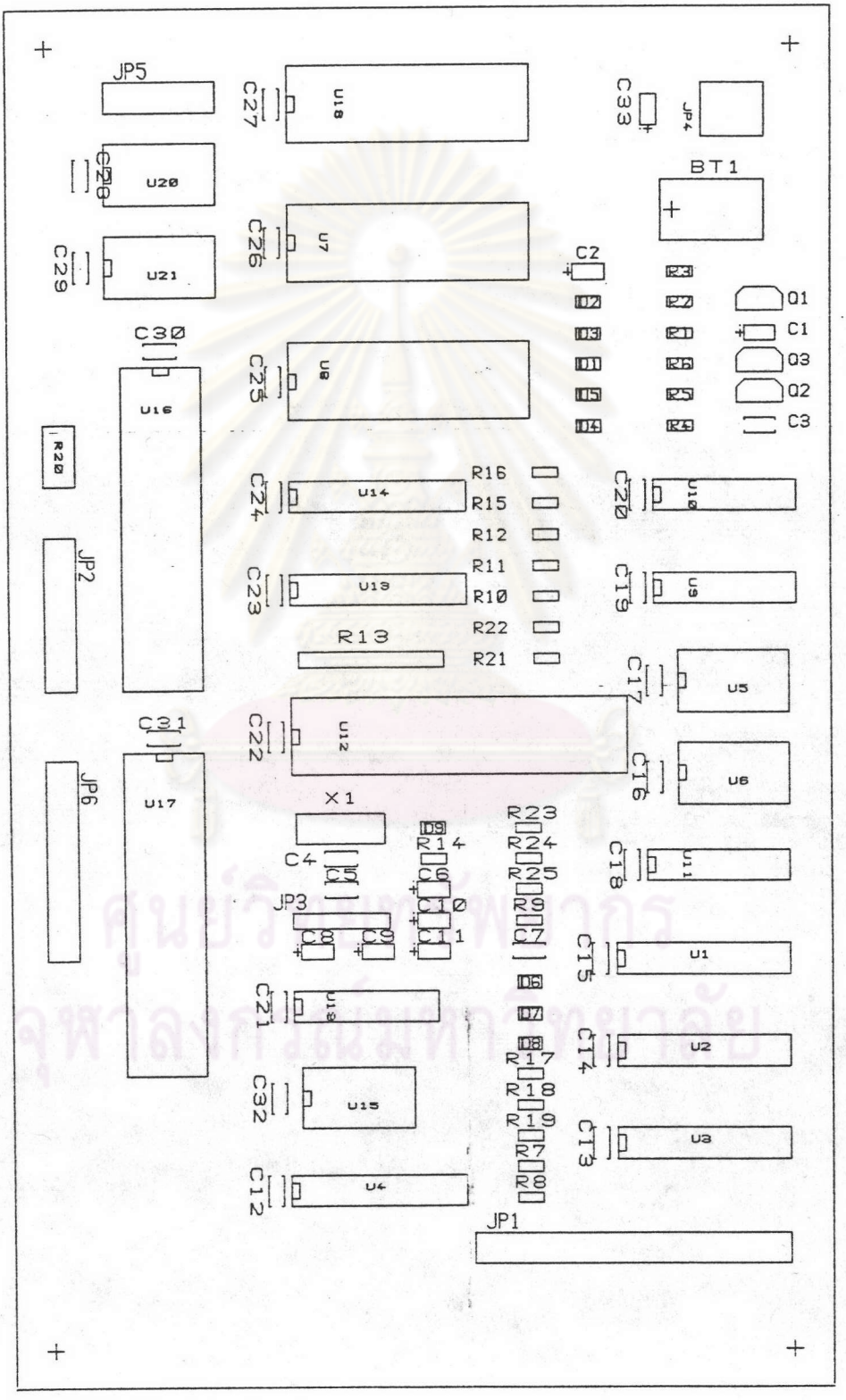


Fig. B-7 Component side layout of TMS320 part PCB.

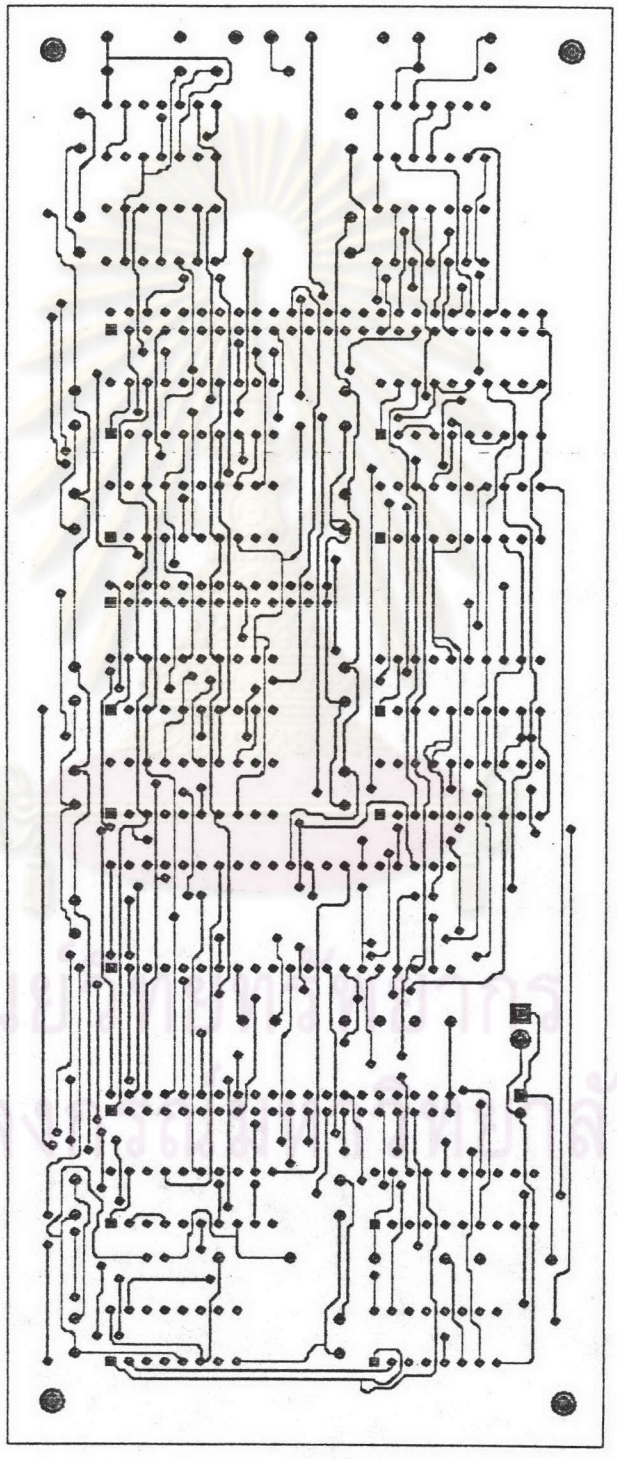


Fig. B-8 Solder side layout of TMS320 part PCB.

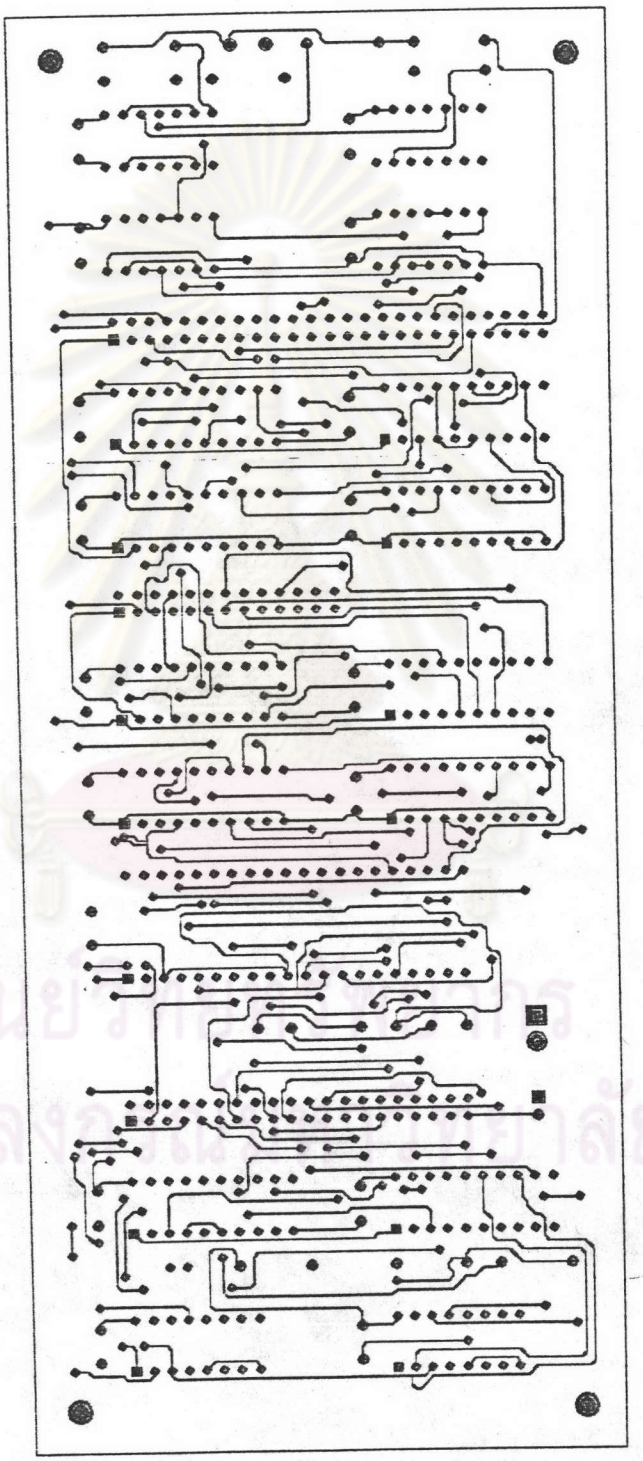


Fig. B-9 Component overlay of TMS320 part PCB.

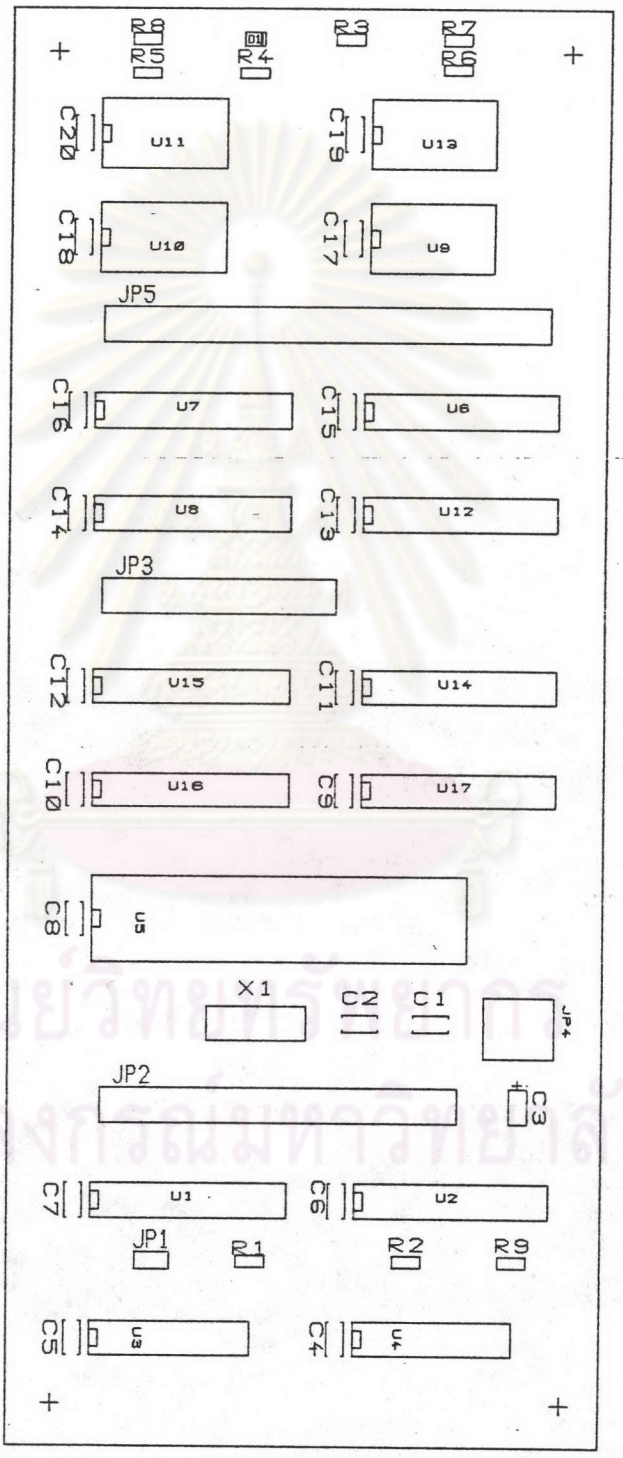


Fig. B-10 Component side layout of TMS320 IO part PCB.

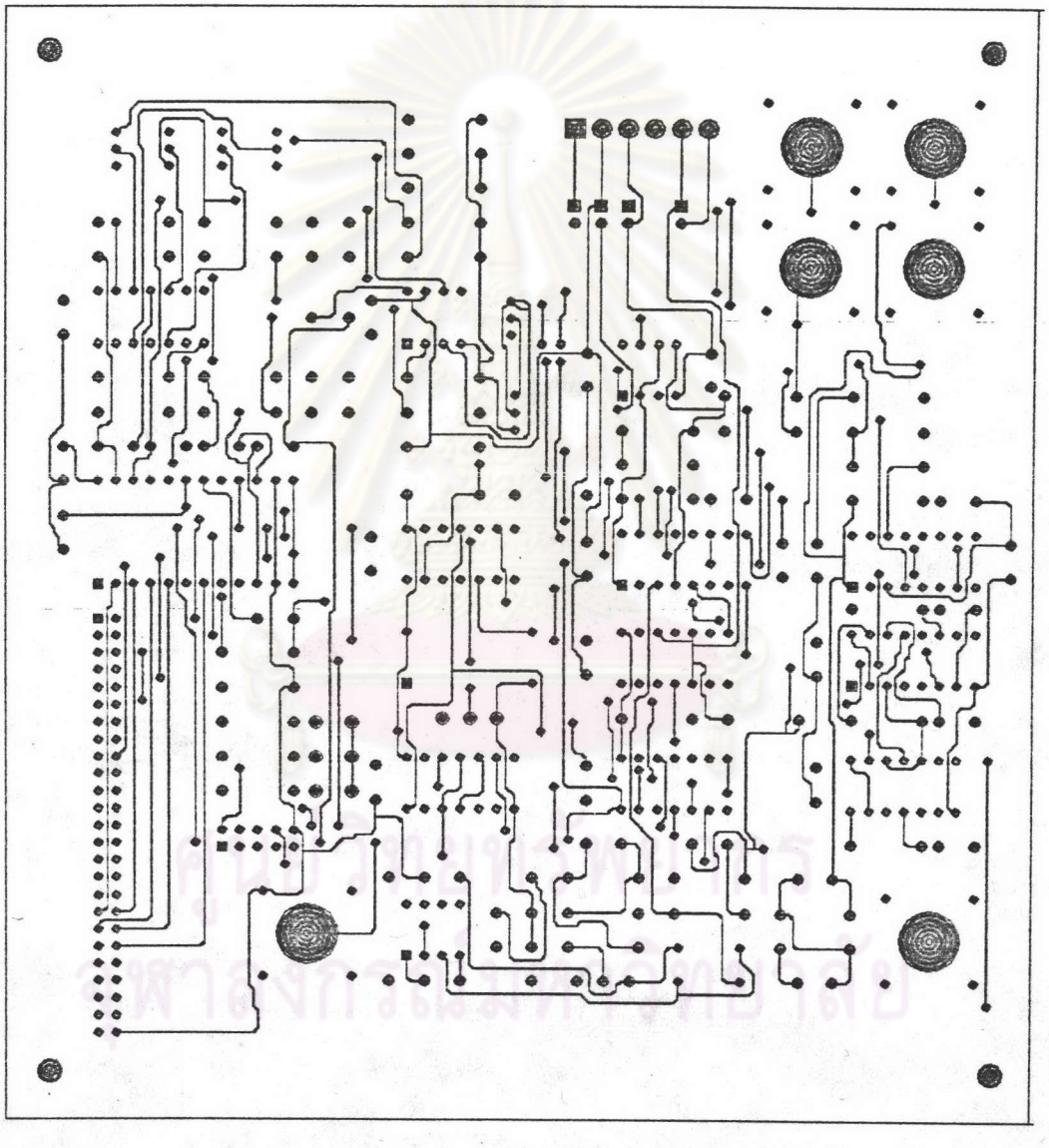


Fig. B-11 Solder side layout of TMS320 IO part PCB.

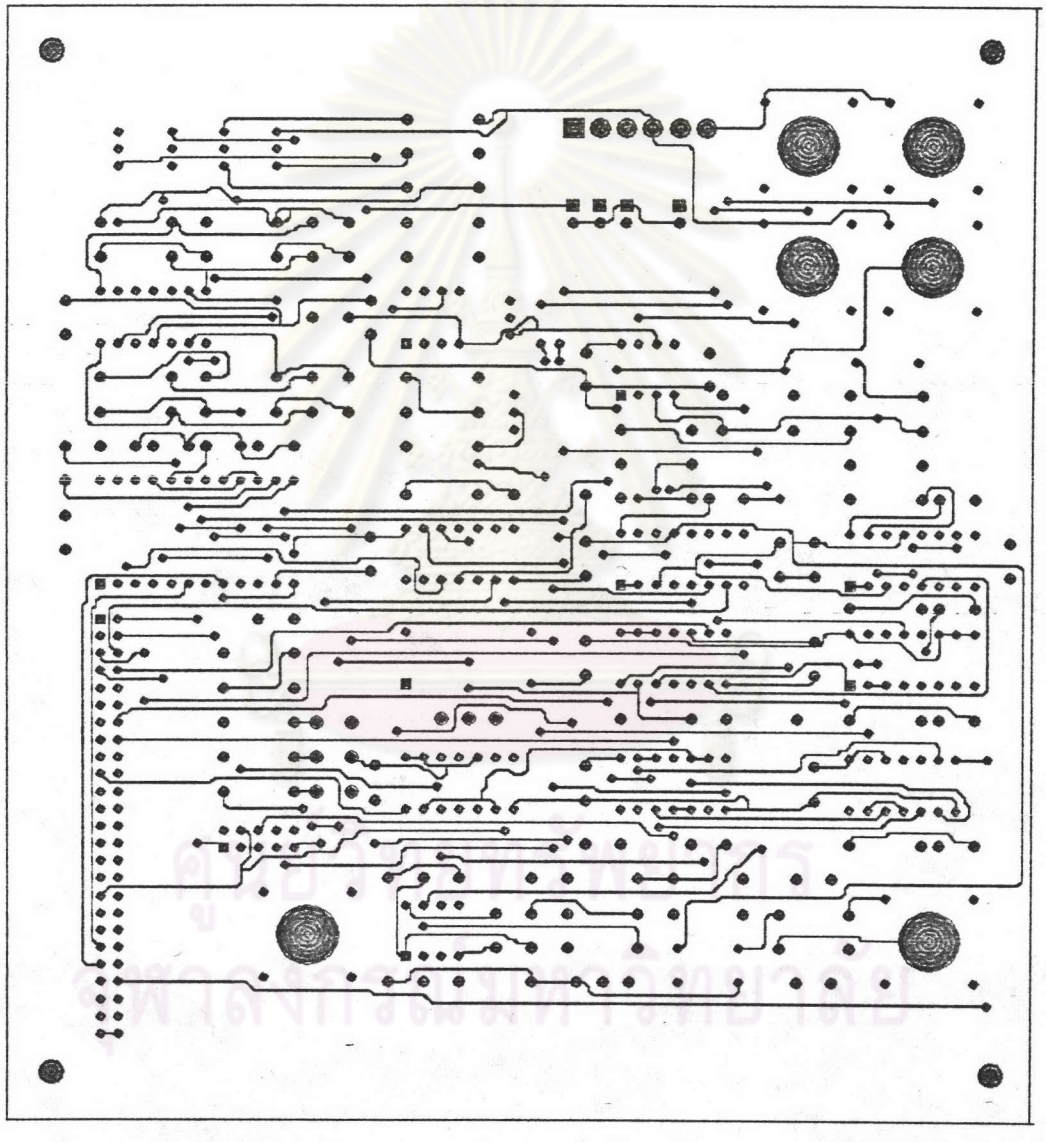


Fig. B-12 Component overlay of TMS320 IO part PCB.

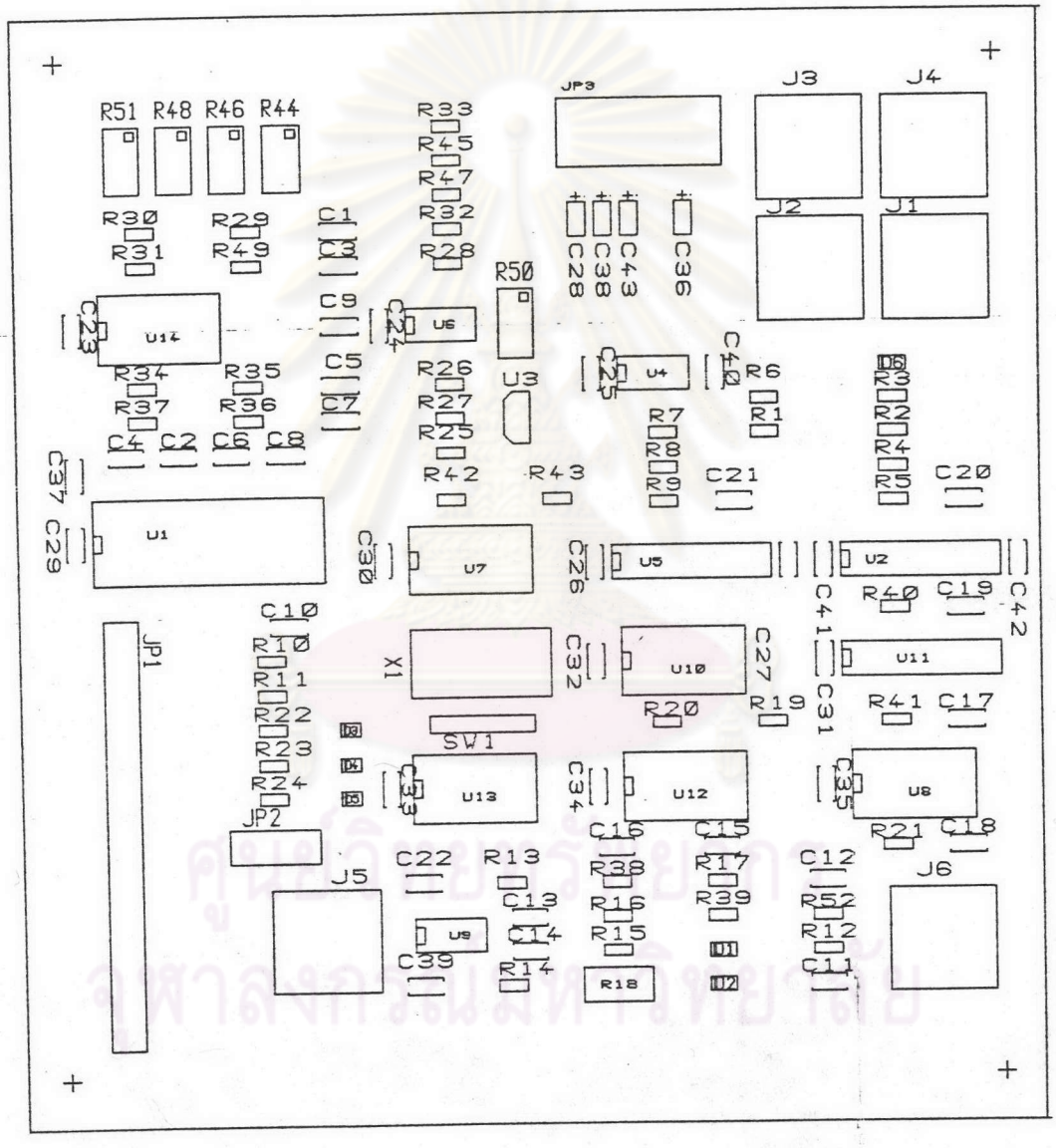


Fig. B-13 Component side layout of analog part PCB.

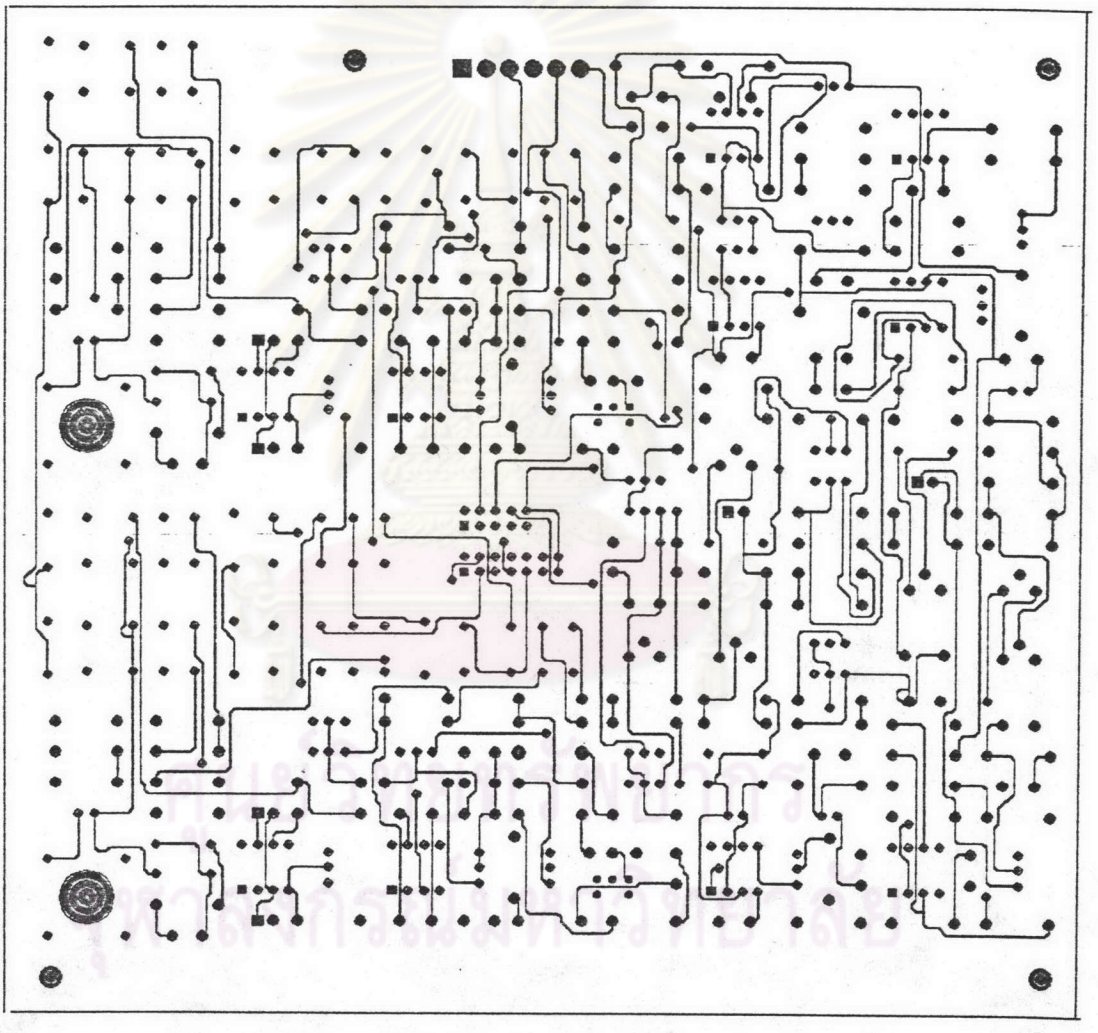


Fig. B-14 Solder side layout of analog part PCB.

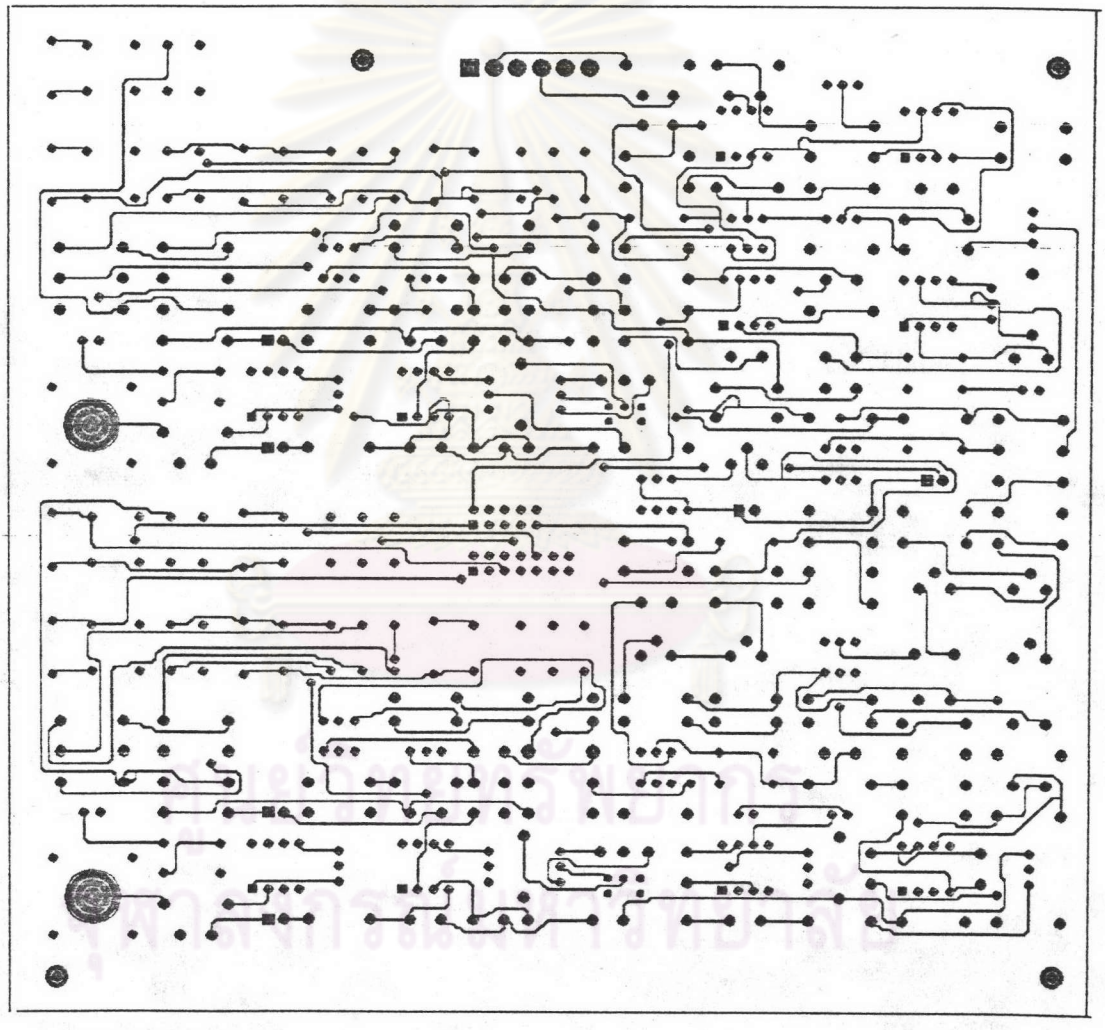
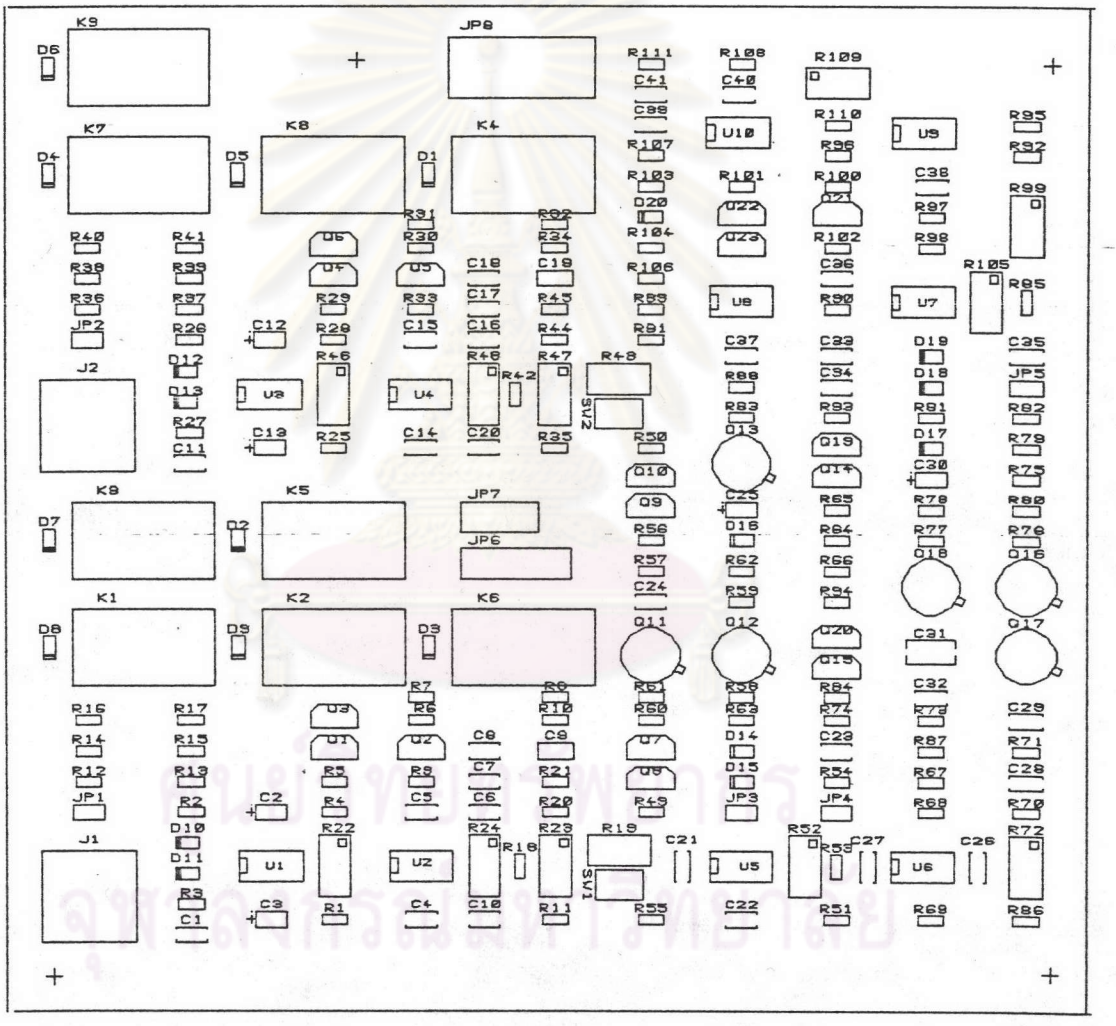


Fig. B-15 Component overlay of analog part PCB.



Appendix C

Software Listing

Source Code for MCS32 Control Program

```
40 REM ******************
50 REM *** INITIALIZ SECTION
60 REM ******************
65 STRING 300,20 :  REM *** Alloc for string
66 $(0)="Hello NMR1":PUSH 10:PUSH 0:GOSUB 9100
67 $(0)="Hello NMR2":PUSH 10:PUSH 1:GOSUB 9100
68 FOR D=0 TO 1000:NEXT
70 PORT1=0BFH : IE=IE.OR.81H : DBY(24H)=DBY(24H).OR.80H :  REM *** ENTER TMS
80 LCDCLS  :  DIM XC(5) :  DIM YC(5) :  REM *** CLEAR LCD
90 REM *** ADDRESS
100 PA=1F01H :  REM *** PARAMETER RAM
110 TI=2000H : RO=1 :  REM *** TMS CODE
120 P1=0F800H : P2=0FC00H :  REM *** I/O PORT
170 VM=5 : VI=1 :  REM *** SENSITIVITY
180 $(0)="  "
200 $(1)="0.2 Volt (FS)"
210 $(2)="0.4 Volt (FS)"
220 $(3)="0.8 Volt (FS)"
230 $(4)="2.0 Volt (FS)"
240 $(5)="4.0 Volt (FS)"
250 $(6)="8.0 Volt (FS)"
280 DM=3 : DI=7 :  REM *** Dwell Time
300 $(7)="  5-99 uS"
310 $(8)="  100-999 uS"
320 $(9)="  1-9 mS"
330 $(10)="  10-99 mS"
650 XS=5 : YS=5
660 DIM XM(XS) :  DIM YM(YS) :  REM *** LENGTH AND DEPTH
670 XM(0)=64 : XM(1)=128 : XM(2)=256 : XM(3)=512 : XM(4)=1024 : XM(5)=2048
680 YM(0)=256 : YM(1)=128 : YM(2)=64 : YM(3)=32 : YM(4)=16 : YM(5)=8
```

```
690 MW=999 :  REM *** MAX SWEEP
700 SC=0 :  REM *** SCALE FACTOR 2 power=(-4,-3,-2,-1,-0,1,2,3,4)
710 REM *** SET DEFAULT PARAMETER
1000 S1=0 : S2=0 :  REM *** SENSITIVITY 0.2
1010 PO=0 :  REM *** POLARITY 0=NORMAL
1020 DT=0 :  REM *** DWELL 5-99 uS
1030 X=64 :  REM *** X =128
1040 Y=64 :  REM *** Y =128
1050 SW=MW
1060 MO=0 :  REM *** MODE (NORM, SUM, EXP)
1070 SC=0
1080 CL=1 :  REM *** CLEAR
1090 RA=1 : RB=X :  REM *** ROI SELECTED
1100 REM *** CHECK PARAMETER RAM
1500 XADDR PA :  XREAD  :  POP C
1510 IF C<>0AA55H THEN  GOSUB 9500 :  REM *** WRITE PARAs
1520 GOSUB 9000 :  REM *** READ PARAMETER
1530 GOSUB 15000 :  REM *** MOVE TMS CODE
1540 REM **********
1550 REM *** MAIN PROGRAM *****1
1560 REM **********
2000 CC=0
2020 GOSUB 10000 :  REM *** SET I/O
2030 CS=1 : XC(0)=0 : YC(0)=1 : XC(1)=13 : YC(1)=1
2040 ENCSET CS,CC,0,64 :  ENCKEY  :  POP K
2050 LCDCLS
2055 PRINT @"Choose Desired Menu" :  PRINT @" Routine     Config"
2060 C=-1
2070 DO
2080 ENCGET  :  POP CC :  IF CC<>C THEN C=CC :  GOSUB 9800
2090 ENCKEY  :  POP K :  UNTIL K<>0
2100 IF CC<>2 THEN  PUSH CC :  ON CC GOSUB 5000,3000 :  POP CC
2110 GOTO 2020
2120 REM **********
2130 REM *** CONFIG
2140 REM **********
```

```
2150 REM *** CH1 SENSITIVITY *****2
3000 ENCSET VM,S1,0,64 : ENCKEY : POP K
3010 C=-1
3020 DO
3030 ENCGET : POP S1 : IF C=S1 THEN GOTO 3040 ELSE C=S1 : GOSUB 10000
3035 LCDCLS : PRINT @"Channel1 Sensitivity" : PRINT @$(0),CHR(3),$(VI+S1),
3040 ENCKEY : POP K : UNTIL K<>0
3050 REM *** CH1 SENSITIVITY *****3
3060 ENCSET VM,S1,0,64 : ENCKEY : POP K
3070 C=-1
3080 DO
3090 ENCGET : POP S2 : IF C=S2 THEN GOTO 3100 ELSE C=S2 : GOSUB 10000
3095 LCDCLS : PRINT @"Channel2 Sensitivity" : PRINT @$(0),CHR(3),$(VI+S2),
3100 ENCKEY : POP K : UNTIL K<>0
3110 REM **********
3120 REM *** POLARITY *****4
3130 CS=1 : XC(0)=1 : YC(0)=1 : XC(1)=11 : YC(1)=1
3140 ENCSET CS,PO,0,64 : ENCKEY : POP K
3150 LCDCLS
3155 PRINT @"Channel1/2 Polarity" : PRINT @" Normal  Invert"
3160 C=-1
3170 DO
3180 ENCGET : POP PO : IF C<>PO THEN C=PO : GOSUB 9800
3190 ENCKEY : POP K : UNTIL K<>0
3200 REM
3210 REM *** DWELL TIME *****5
3220 ENCSET DM,DT,0,64 : ENCKEY : POP K
3230 C=-1
3240 DO
3250 ENCGET : POP DT
3255 IF C=DT THEN GOTO 3265 ELSE C=DT : GOSUB 10000
3260 LCDCLS : PRINT @"Dwell Time" : PRINT @$(DI+DT)
3265 ENCKEY : POP K : UNTIL K<>0
3270 REM *** X LENGTH *****6
3280 XX=XS+1
3290 DO
```

```
3300 XX=XX-1
3310 UNTIL X=XM(XX).OR.XX=0
3320 ENCSET XS,XX,0,64 : ENCKEY : POP K
3330 LCDCLS
3335 PRINT @"Sweep Frame Length" : PRINT @" X= 0000 Columns"
3340 C=-1
3350 DO
3360 ENCGET : POP XX
3370 IF XX<>C THEN C=XX : LCDXY 5,1 : PRINT @USING(####),XM(XX),USING(0), CR ,
3380 ENCKEY : POP K : UNTIL K<>0
3390 X=XM(XX) : IF RB>X THEN RB=X : IF RA>X THEN RA=X
3400 REM *** DEPTH ***** 5
3410 IF Y>YM(XX) THEN Y=YM(XX)
3420 ENCSET YM(XX),Y,0,8 : ENCKEY : POP K
3430 LCDCLS
3435 PRINT @"Sweep Frame Depth" : PRINT @" Y= 000 Rows"
3440 C=-1
3450 DO
3460 ENCGET : POP Y
3470 IF Y<>C THEN C=Y : LCDXY 5,1 : PRINT @USING(###),Y,USING(0), CR ,
3480 ENCKEY : POP K : UNTIL K<>0
3490 REM *** SWEEP *****8
3500 ENCSET MW-1,SW-1,0,8 : ENCKEY : POP K
3510 LCDCLS
3515 PRINT @"Number of Sweeps" : PRINT @" 000000 Counts"
3520 C=-1
3530 DO
3540 ENCGET : POP SW : SW=SW+1
3550 IF C<>SW THEN C=SW : LCDXY 2,1 : PRINT @USING(######),SW,USING(0), CR ,
3560 ENCKEY : POP K : UNTIL K<>0
3570 REM *** SAVE PARAMETERS ***** 9
3580 GOSUB 9600 : IF V=0 THEN GOSUB 9500
3590 RETURN
3600 REM *****************
3610 REM *** ROUTINE SECTION *****10
3620 REM *****************
```

```
5000 CC=0
5010 DO
5020 CS=3 : XC(0)=0 : YC(0)=0 : XC(1)=10 : YC(1)=0
5030 XC(2)=0 : YC(2)=1 : XC(3)=18 : YC(3)=1
5040 ENCSET CS,CC,0,64 :  ENCKEY  :  POP K
5050 LCDCLS  :  PRINT @" Averager  Data Xfer" :  PRINT @" Processor       Q"
5060 C=-1
5070 DO
5080 ENCGET  :  POP CC :  IF C<>CC THEN C=CC :  GOSUB 9800
5090 ENCKEY  :  POP K :  UNTIL K<>0
5100 IF CC<>3 THEN  PUSH CC :  ON CC GOSUB 6000,8000,5000 :  POP CC
5110 UNTIL CC=3
5120 REM ******************
5130 REM *** AVERAGE SECTION *****11
5140 REM ******************
6000 CC=0
6010 DO
6020 CS=2 : XC(0)=0 : YC(0)=1 : XC(1)=8 : YC(1)=1 : XC(2)=18 : YC(2)=1
6030 ENCSET CS,CC,0,64 :  ENCKEY  :  POP K
6040 LCDCLS  :  PRINT @"Averager" :  PRINT @" Start   Config    Q"
6050 C=-1
6060 DO
6070 ENCGET  :  POP CC :  IF C<>CC THEN C=CC :  GOSUB 9800
6080 ENCKEY  :  POP K :  UNTIL K<>0
6090 IF CC<>2 THEN  PUSH CC :  ON CC GOSUB 7000,6500 :  POP CC
6100 UNTIL CC=2
6110 RETURN
6120 REM *** AVG CONFIG *****13
6500 CS=2 : XC(0)=0 : YC(0)=1 : XC(1)=8 : YC(1)=1 : XC(2)=15 : YC(2)=1
6510 ENCSET CS,MO,0,64 :  ENCKEY  :  POP K
6520 LCDCLS  :  PRINT @"MODE" :  PRINT @" Norm    Sum    Exp"
6530 C=-1
6540 DO
6550 ENCGET  :  POP MO :  IF C<>MO THEN C=MO :  GOSUB 9800
6560 ENCKEY  :  POP K :  UNTIL K<>0
6570 REM *** INIT DATA ***** 14
```

```
6580 CS=1 : XC(0)=1 : YC(0)=1 : XC(1)=8 : YC(1)=1
6590 ENCSET CS,CL,0,64 :  ENCKEY  :  POP K
6600 LCDCLS  :  PRINT @"Initialize Data?" :  PRINT @"  Yes   No"
6610 C=-1
6620 DO
6630 ENCGET  :  POP CL :  IF C<>CL THEN C=CL :  GOSUB 9800
6640 ENCKEY  :  POP K :  UNTIL K<>0
6650 RETURN
6660 REM ***  AVG START *****12
7000 LCDCLS  :  PRINT @"Sweep Completed 0000" :  PRINT @"Depth Completed  000"
7010 RO=1 :  GOSUB 16000 :  REM *** MOVE PARAMETER
7020 ENCKEY  :  POP K :  CNTCLR  : CW=0
7030 IE=IE.OR.04H : PORT1=PORT1.OR.40H :  REM *** ENTER TMS RUN MODE
7040 DO
7050 CNTGET  :  POP CN
7060 IF CW=CN THEN  GOTO 7510
7070 CW=CN
7080 CNTCLR  : CW=0 : CY=CY+1
7500 LCDXY 17,1 :  PRINT @USING(###),CY,USING(0),CR,
7510 ENCKEY  :  POP K :  UNTIL K<>0.OR.(PORT1.AND.4)=0 :  REM *** WAIT CPLT
7520 PORT1=PORT1.AND.0BFH : IE=IE.AND.0FBH :  REM *** ENTER TMS CONFIG MODE
7530 LCDXY 17,1 :  PRINT @USING(###),Y,USING(0),CR,
7535 LCDXY 16,0 :  PRINT @USING(####),SW,USING(0),CR,
7540 RETURN
7550 REM **********
7560 REM *** DATAXFER SECTION *****15
7570 REM **********
8000 CC=0
8010 DO
8020 CS=2 : XC(0)=0 : YC(0)=1 : XC(1)=8 : YC(1)=1 : XC(2)=18 : YC(2)=1
8030 ENCSET CS,CC,0,64 :  ENCKEY  :  POP K
8040 LCDCLS  :  PRINT @"Data Xfer" :  PRINT @" Start  Config   Q"
8050 C=-1
8060 DO
8070 ENCGET  :  POP CC :  IF C<>CC THEN C=CC :  GOSUB 9800
8080 ENCKEY  :  POP K :  UNTIL K<>0
```

```
8090 IF CC<>2 THEN  PUSH CC :  ON CC GOSUB 8200,8400 :  POP CC
8100 UNTIL CC=2
8110 REM **********
8120 REM *** DATA XFER WAIT FOR HOST ***** 16
8130 REM **********
8200 LCDCLS  :  PRINT @"Data Xfer" :  PRINT @" Waiting for Host"
8205 ENCKEY : POP K
8210 DO : ENCKEY : POP K : PRINT CHR(6), : H=GET : UNTIL H=6 .OR. K<>0
8220 IF K<>0 THEN  RETURN
8230 LCDCLS  :  PRINT @"Data Xfer" :  PRINT @"    in progress..."
8240 BADDR 0000H : GOSUB 9400
8250 FOR D=1 TO Y
8260 PRINT CHR(2),: DO : ENCGET : POP K: H=GET : UNTIL H=6 .OR. K<>0
8270 IF K<>0 THEN RETURN
8280 FOR IX=1 TO X
8290 BREAD  :  POP DX : IF (IX>=RA) .AND. (IX<=RB) THEN PH1. DX,
8295 BREAD  :  POP DX : IF (IX>=RA) .AND. (IX<=RB) THEN PH1. DX,
8300 NEXT IX
8310 NEXT D
8215 LCDCLS  :  PRINT @"Data Xfer" :  PRINT @" Depth Completed"
8320 PRINT CHR(4),  :  FOR I=0 TO 200 : NEXT : RETURN
8330 REM *** DATAXFER CONFIG *****18
8400 RETURN
8410 REM *** DATAXFER POST PROC *****19
8420 REM ***** 20
8500 RETURN
8510 REM ****************************
8520 REM ******** SUBROUTINE ******
8530 REM ****************************
8540 REM *** READ PARAMETERS
8550 REM ****************************
9000 XADDR PA :  XREAD  :  POP C
9010 XREAD : POP S1 : XREAD : POP S2 : XREAD : POP PO : XREAD : POP DT
9020 XREAD  :  POP X :  XREAD  :  POP Y :  XREAD  :  POP SW
9030 XREAD : POP MO : XREAD : POP SC : XREAD : POP CL : XREAD : POP RA
9035 XREAD  :  POP RB
```

```
9040 RETURN
9050 REM ********************
9060 REM *** TITLE
9070 REM ********************
9100 POP L :  POP Z : C=INT(20-Z)/2
9110 FOR I=19 TO C STEP -1
9120 LCDCLS  :  LCDXY I,L
9130 FOR J=1 TO 20-I
9140 IF J<=Z THEN  PRINT @CHR($(0),J), ELSE  PRINT @" ",
9150 FOR D=0 TO 50 :NEXT:NEXT
9160 NEXT
9200 RETURN
9210 REM ********************
9220 REM *** HEADER PARAMETERS
9400  PRINT  CHR(1),"X",X,",Y",Y,",RA",RA,",RB",RB
9410 PRINT "Channel 1 Sensitivity  = ",$(VI+S1)
9420 PRINT "Channel 2 Sensitivity  = ",$(VI+S2)
9430 PRINT "Channel 1/2 Polarity  = ",
9435 IF PO=0 THEN PRINT "Normal" ELSE PRINT "Invert"
9440 PRINT "Number of Sweeps       =",SW
9450 PRINT "Sweep Frame Length (X) =",USING(####),X
9460 PRINT "Sweep Frame Depth  (Y) =",USING(####),Y
9470 PRINT "ROI A= ",USING(####),RA,"  B= ",USING(####),RB
9480 PRINT CHR(1),: RETURN
9490 REM ********************
9491 REM *** WRITE PARAMETERS
9500 XADDR PA :  XWRITE 0AA55H : PRINT "INIT RAM"
9510 XWRITE S1 :  XWRITE S2 :  XWRITE PO :  XWRITE DT
9520 XWRITE X :  XWRITE Y :  XWRITE SW
9530 XWRITE MO :  XWRITE SC :  XWRITE CL :  XWRITE RA :  XWRITE RB
9540 RETURN
9550 REM ********************
9560 REM *** SAVE DIALOG *****9
9570 REM ********************
9600 V=0
9610 CS=1 : XC(0)=1 : YC(0)=1 : XC(1)=8 : YC(1)=1
```

```
9620 ENCSET CS,V,0,64 : ENCKEY : POP K
9630 LCDCLS : PRINT @"Save All Parameters?" : PRINT @" Yes  No"
9640 C=-1
9650 DO
9660 ENCGET : POP V : IF V<>C THEN C=V : GOSUB 9800
9670 ENCKEY : POP K : UNTIL K<>0
9700 REM *** CURSOR
9710 REM ******************
9800 FOR I=0 TO CS : LCDXY XC(I),YC(I) : PRINT @" ", : NEXT
9810 LCDXY XC(C),YC(C) : PRINT @">", : RETURN
9820 REM ******************
9830 REM *** SET I/O
10000 RL=S1+(S2*8)+((DM-DT)*64) : XBY(P1)=RL
10010 RETURN
10020 REM ******************
10030 REM *** MOVE TMS ROUTINE
10040 REM ******************
15000 XADDR TI : RESTORE
15010 DO : READ T : XWRITE T : UNTIL T=0
15020 RETURN
15030 REM ******************
15040 REM *** PARAMETER
15050 REM *** (ROUTINE,X,Y,SWEEP,POLAR,SCALE,WITHCLEAR:WORD)
15060 REM ******************
16000 XADDR TI+4 : REM *** START OF PARAMETER
16010 XWRITE RO
16020 XWRITE X : XWRITE Y : XWRITE SW : XWRITE PO
16030 XWRITE SC : XWRITE CL
16040 RETURN
16050 REM ******************
16060 REM *** TMS CODE
16070 REM ******************
17000 DATA 0 : REM *** PLACE TMS CODE HERE END WITH ZERO
20000 RETURN
30000 END
```

Source Code for MCS32 Command/Statement Extension Program

```
SPHOLD    equ    3eh
STACK     equ    7fh              ;Internal
RESVZ     equ    255              ;Reserv for TMS Parameter
RTOP      equ    1fffh-RESVZ      ;1f00h=Start of RAM TOP
MTOP      equ    10ah             ;MCS-BASIC
VTOP      equ    104h
DIMUSE    equ    108h
VARUSE    equ    106h
ARGSTK    equ    9h               ;Argument stack
CTRLSTK   equ    11h              ;Control stack
BASIC     equ    13h              ;Hi, Point to the Start of BASIC program EPROM 8011h
t2048     equ    0ffffh-450+1     ;450 *(1/921.6KHz)=0.5 msec
;
; Internal RAM for Command/Statemant Extensions
XDSTART   equ    4dh
LCDX      equ    XDSTART+0        ;X(1)
LCDY      equ    XDSTART+1        ;Y(1)
ENCF      equ    XDSTART+2        ;Flag(1)
ENCMAX    equ    XDSTART+3        ;Max(2)
ENCMIN    equ    XDSTART+5        ;Min(2)
ENCCUR    equ    XDSTART+7        ;Current(2)
ENCMOD    equ    XDSTART+9        ;Mode(1)
ENCSC     equ    XDSTART+10       ;Scale(1)
ENCK      equ    XDSTART+11       ;Key(1)
ENCSCC    equ    XDSTART+12       ;Scale(1)
COUNT     equ    XDSTART+13       ;Count(2)
XRAMP     equ    XDSTART+15       ;2
KEYDLY    equ    XDSTART+17       ;1
TEMP      equ    XDSTART+18
;
;I/O
PIO1      equ    0f800h
PIO2      equ    0fc00h
BUFAD     equ    06000h
BUFRW     equ    04000h
```

```
;
; OPBYTE
XTALCAL   equ   58          ;XTAL Calculation
DISP      equ   6
CRLF      equ   7           ;Print CR/LF
POPAS     equ   1           ;Fs to I
PUSHAS    equ   154         ;I to Fs
EVAEXP    equ   57
BGETCI    equ   64
GORUN     equ   66
GOCOM     equ   0
;
          org   2001h       ;Tell BASIC That Reset is External
          db    0aah
;
          org   2002h       ;Tell BASIC That Command/Statemant Extensions
          db    5ah
;
          org   2048h       ;Command/Statemant is available
          setb  45
          ret
;
          org   2070h       ;Command/Statemant Vector
          mov   DPTR,#VECXCOM
          ret
;
          org   2078h       ;Command/Statemant Lookup
          mov   DPTR,#XCOM
          ret
;
          org   2090h       ;**** Reset option
          ljmp  XRESET
;
          org   2b00h
XRESET:   mov   r0,#0ffh
          clr   a
```

```
CLRR:   mov     @r0,a
        djnz    r0,CLRR
;
        mov     a,#STACK
        mov     sp,a
        mov     SPHOLD,a
;
        mov     r3,#HIGH RTOP-1
        mov     r1,#LOW RTOP-1
        mov     DPTR,#0ffffh
CLRXR:  inc     DPTR
        clr     a
        movx    @DPTR,a
        mov     a,r3
        cjne    a,DPH,CLRXR
        mov     a,r1
        cjne    a,DPL,CLRXR
;
;       MTOP
        mov     DPTR,#MTOP
        mov     a,#HIGH RTOP
        movx    @DPTR,a
        inc     DPTR
        mov     a,#LOW RTOP
        movx    @DPTR,a
;
;       VARTOP=MEMTOP
        mov     DPTR,#VTOP
        mov     a,#HIGH RTOP
        movx    @DPTR,a
        inc     DPTR
        mov     a,#LOW RTOP
        movx    @DPTR,a
;
;       DIMUSE=528
        mov     DPTR,#DIMUSE
```

```
                mov     a,#HIGH 528
                movx    @DPTR,a
                inc     DPTR
                mov     a,#LOW 528
                movx    @DPTR,a
;
;               VARUSE=VARTOP
                mov     DPTR,#VARUSE
                mov     a,#HIGH RTOP
                movx    @DPTR,a
                inc     DPTR
                mov     a,#LOW RTOP
                movx    @DPTR,a
;
                mov     ARGSTK,#0feh    ;Argument stack
                mov     CTRLSTK,#0feh   ;Control stack
;
                mov     DPTR,#512       ;No program in ram
                mov     a,#01h
                movx    @DPTR,a
;
                mov     BASIC,DPH
                mov     BASIC+1,DPL
;
                mov     DPTR,#XTAL
                mov     a,ARGSTK        ;ARG pointer=ARG pointer-6
                clr     c
                subb    a,#6
                mov     ARGSTK,a
                mov     r0,a
                mov     p2,#1           ;ARG Page address
                mov     r1,#6
;
XTALMV:         clr     a
                movc    a,@a+DPTR
                movx    @r0,a
```

```
                inc     DPTR
                dec     r0
                djnz    r1,XTALMV
;
                mov     a,#XTALCAL      ;OPBYTE XTAL Calculation
                lcall   30h
;
; Setting I/O
; LCD HITACHI 230014 HDTS P/N
                mov     DPTR,#PIO2+3    ;Control port PIO2
                mov     a,#80h          ;Port A/O, B/O, C/O
                movx    @DPTR,a
                mov     DPTR,#PIO1+3    ;Control port PIO1
                movx    @DPTR,a
;
; Init LCD Port B
                mov     a,#00111000b    ;Function set Interface 8 bit,2 Lines, 5x7
                lcall   LCDIW
                mov     a,#00001100b    ;Disp ON,Cursor(bit1=0=OFF),Blink Off
                lcall   LCDIW
                mov     a,#00000110b    ;Entry Modeg,Inc Cursor
                lcall   LCDIW
;
                mov     a,#01000000b    ;Set CG RAM Address Command
                add     a,#00011000b    ;11 000 Address of CHAR(3)
                lcall   LCDIW
;
                mov     r0,#8
CHR3L:          mov     a,#8
                clr     c
                subb    a,r0
                mov     DPTR,#CHR3
                movc    a,@a+DPTR
                lcall   LCDDW
                djnz    r0,CHR3L
;
```

```
                lcall   LCDCLS
;
; Encoder
                mov     TI1,#LOW t2048
                mov     TH1,#HIGH t2048
                orl     TCON,#00000101b         ;INT0, INT1 edge Trigger
;
; Check ROM
CHKROM:         mov     DPTR,#8000h
                movx    a,@DPTR
                cjne    a,#'1',GOTOCOM
                inc     DPTR
                movx    a,@DPTR
                mov     RCAP2H,a        ;Serial set
                inc     DPTR
                movx    a,@DPTR
                mov     RCAP2L,a
                mov     BASIC,#HIGH 8011h
                mov     BASIC+1,#LOW 8011h
                setb    45              ;Command/Statemant is available
                mov     a,#GORUN
                ljmp    30h
;
GOTOCOM:
                mov     r3,#00h         ;Auto BAUD
                mov     r1,#00h
                mov     r0,#04h
                jb      RXD,$
SERL:           djnz    r0,$
                clr     c
                mov     a,r1
                subb    a,#1
                mov     r1,a
                mov     a,r3
                subb    a,#0
                mov     r3,a
```

```
                mov     r0,#3
                jnb     RXD,SERL
                jb      RXD,$
                jnb     RXD,$
                mov     RCAP2H,r3
                mov     RCAP2L,r1
;
                mov     r3,#HIGH SIGNON         ;Sign on
                mov     r1,#LOW SIGNON
                setb    52                      ;print from ROM
                mov     a,#DISP
                lcall   30h
;
                mov     a,#CRLF         ; CRLF
                lcall   30h
;
                clr     a
                ljmp    30h             ;GOTO Command mode
;
SIGNON:
                db      'Averager V1.0"'
XTAL:           db      88h             ;11.0592 MHz    Crystal
                db      00h
                db      00h
                db      92h
                db      05h
                db      11h
;
;               Command/Statemant Lookup
                org     3000h
XCOM:
                db      10h             ;LCD
                db      "LCDCLS"
                db      00h
                db      11h
                db      "LCDXY"
```

```
        db      00h
        db      12h             ;Encoder
        db      "ENCSET"
        db      00h
        db      13h
        db      "ENCGET"
        db      00h
        db      14h
        db      "ENCKEY"
        db      00h
;
        db      15h             ;XRAM
        db      "XADDR"
        db      00h
        db      16h
        db      "XREAD"
        db      00h
        db      17h
        db      "XWRITE"
        db      00h
;
        db      18h             ;COUNT
        db      "CNTCLR"
        db      00h
        db      19h
        db      "CNTGET"
        db      00h
;
        db      1ah
        db      "BADDR"
        db      00h
        db      1bh
        db      "BREAD"
        db      00h
        db      1ch
        db      "BWRITE"
```

```
                db      00h
                db      1dh
                db      "XDOWN"
                db      00h
                db      1eh
                db      "XUP"
                db      00h
                db      1fh
                db      "BUP"
;
                db      0ffh
;
; Command/Statemant Vector
                org     3100h
VECXCOM:
                dw      LCDCLS,LCDXY
                dw      ENCSET,ENCGET,ENCKEY
                dw      XADDR,XREAD,XWRITE
                dw      CNTCLR,CNTGET
                dw      BADDR,BRAED,BWRITE
                dw      2100h,21f0h,234ch
;
                org     3200h
; LCD
LCDCLS:         mov     a,#00000001b    ;Clear Display
                lcall   LCDIW
                clr     a
                mov     LCDX,a
                mov     LCDY,a
                ret
;
; X=0..39, Y=0..1
LCDXY:          mov     a,#EVAEXP
                lcall   30h             ;X
                acall   XNXTEXP         ;Y
                mov     a,#POPAS
```

```
                lcall   30h
                mov     TEMP,r1
                mov     a,#POPAS
                lcall   30h
                mov     LCDX,r1
                mov     LCDY,TEMP
;
LCDDOXY:        mov     TEMP,#0         ;DD Ram Addr of Line 2
                mov     a,LCDY
                jnb     ACC.0,$+6
                mov     TEMP,#40h       ;DD Ram Addr of Line 2
;
                mov     a,LCDX          ;GOTO X
                mov     B,#40           ;40 Chars/Line
                div     AB
                mov     a,B
                add     a,TEMP
                add     a,#10000000b    ;Set DDRAM Address Command
                ljmp    LCDIW
;
LCDCOM:         mov     a,#POPAS
                lcall   30h                     ;in r3:r1
                mov     a,r1
                ljmp    LCDIW
;
LCDDATA:        mov     a,#POPAS
                lcall   30h                     ;in r3:r1
                mov     a,r1
                ljmp    LCDDW
;
; Encoder
; ENCF.0 Encoder A
; ENCF.4 INT0 (CPLT,MODE)
; ENCF.6 MODE Clear= Bound
; ENCF.7 Encoder ON
ENCSET:         clr     IE.3                    ;Disable Interrupt Timer1(ET1)
```

```
        anl     ENCF,#00111110b         ;ENC OFF, Mode=0, ENCA=0
        mov     a,#EVAEXP
        lcall   30h             ;Range
        acall   XNXTEXP         ;Current
        acall   XNXTEXP         ;Mode
        acall   XNXTEXP         ;Scale
        mov     a,#POPAS
        lcall   30h
        mov     a,r1
        jnz     $+3
        inc     a
;
        mov     ENCSC,a         ;Save Scale
        mov     ENCSCC,#0
;
        mov     a,#POPAS        ;Mode
        lcall   30h
        mov     a,r1
        jnb     ACC.0,$+6
        orl     ENCF,#01000000b
;
        mov     a,#POPAS        ;Current
        lcall   30h
        mov     ENCCUR,r1
        mov     ENCCUR+1,r3
        mov     a,#POPAS        ;Max (Range)
        lcall   30h
        mov     ENCMAX,r1
        mov     ENCMAX+1,r3
        clr     c
        mov     a,r3
        subb    a,ENCCUR+1
jc      ESMAXC                  ;M<C then C=M
        jnz     ESMISZ          ;Mh>Ch
        mov     a,r1
        cjne    a,ENCCUR,$+3
```

```
            jnc     ESMISZ
ESMAXC:     mov     ENCCUR,r1
            mov     ENCCUR+1,r3
ESMISZ:     mov     a,r1
            orl     a,r3
            jz      ESRET
            orl     ENCF,#10000000b     ;Enc ON
            orl     P1,#00011000b       ;A=P1.3,B=P1.4
            jnb     P1.3,$+6
            orl     ENCF,#00000001b     ;ENCA=1
;
ESRET:      setb    IE.3                ;Enable Interrupt Timer1
            ret
;
ENCGET:     mov     a,IE
            clr     IE.3
            mov     r0,ENCCUR
            mov     r2,ENCCUR+1
            jnb     ACC.3,$+5
            setb    IE.3
;
            mov     a,#PUSHAS
            lcall   30h
            ret
;
ENCKEY:     mov     r0,ENCK
            clr     a
            mov     ENCK,a
            mov     r2,a
            mov     a,#PUSHAS
            lcall   30h
            ret
;
; XADDR Word: Setup Pointer,XRAMP
; 0-1fffh, 2000h-2fffh read First is LOW next HI
; XREAD Return Word(XRAMP)
```

```
; XWRITE Woed(XRAMP)=Arg
XADDR:  mov     a,#EVAEXP       ;XADDR Word
        lcall   30h
        mov     a,#POPAS
        lcall   30h             ;in r3:r1
        mov     XRAMP,r1
        mov     XRAMP+1,r3
        ret
;
XREAD:  mov     DPL,XRAMP       ;XREAD
        mov     DPH,XRAMP+1
        movx    a,@DPTR
        mov     r0,a
        mov     a,DPH
        clr     c
        subb    a,#20h
        jnc     $+3             ;>1fffh
        inc     DPTR
;
        movx    a,@DPTR
        mov     r2,a
        inc     DPTR            ;inc XRAMP
        mov     XRAMP,DPL
        mov     XRAMP+1,DPH
        mov     a,#PUSHAS
        lcall   30h
        ret
;
XWRITE: mov     a,#EVAEXP
        lcall   30h
        mov     a,#POPAS
        lcall   30h
        mov     DPL,XRAMP
        mov     DPH,XRAMP+1
        mov     a,r1
        movx    @DPTR,a
```

```
                mov     a,DPH
                clr     c
                subb    a,#20h
                jnc     $+3             ;>1fffh
                inc     DPTR
;
                mov     a,r3
                movx    @DPTR,a
                inc     DPTR
                mov     XRAMP,DPL
                mov     XRAMP+1,DPH
                ret
;
; COUNTER
CNTCLR:         mov     TEMP,IE
                clr     IE.2            ;Dis INT1
                clr     a
                mov     COUNT,a
                mov     COUNT+1,a
                mov     a,TEMP
                jnb     ACC.2,$+5
                setb    IE.2            ;Enable INT1
;
                ret
;
CNTGET:         mov     TEMP,IE
                clr     IE.2
                mov     r0,COUNT
                mov     r2,COUNT+1
                mov     a,TEMP
                jnb     ACC.2,$+5
                setb    IE.2
;
                mov     a,#PUSHAS
                lcall   30h
                ret
```

```
;
; 6000h Set Address Counter LOW,HI
; 4000h Read/Write Byte
BADDR:  mov     a,#EVAEXP       ;XADDR Word
        lcall   30h
        mov     a,#POPAS
        lcall   30h             ;in r3:r1
        mov     DPTR,#BUFAD
        mov     a,r1
        movx    @DPTR,a
        mov     a,r3
        movx    @DPTR,a
        ret
;
BRAED:  mov     DPTR,#BUFRW
        movx    a,@DPTR
        mov     r0,a
        movx    a,@DPTR
        mov     r2,a
        mov     a,#PUSHAS
        lcall   30h
        ret
;
BWRITE: mov     a,#EVAEXP
        lcall   30h
        mov     a,#POPAS
        lcall   30h
        mov     DPTR,#BUFRW
        mov     a,r1
        movx    @DPTR,a
        mov     a,r3
        movx    @DPTR,a
        ret
;
XNXTEXP: mov    a,#BGETCI
        lcall   30h
```

```
            cjne    a,#',',XERROR
            mov     a,#EVAEXP
            lcall   30h
            ret
XERROR:     mov     a,#CRLF
            lcall   30h
            mov     r3,#HIGH EMSG
            mov     r1,#LOW EMSG
            setb    52
            mov     a,#DISP
            lcall   30h
            clr     a
            lcall   30h
;
EMSG:       db      'ERROR IN COMMAND/STATEMENT EXTENSIONS"
;
;**************
            org     4003h           ;INT0
            sjmp    INT0X
;
            org     4013h           ;INT1
            sjmp    INT1X
;
            org     401bh           ;Timer1
            sjmp    INTT1
;
;**************
; USER PRINT@
            org     403ch           ;User PRINT@ Command Char in Acc, r5(rb0)
USERP:      cjne    a,#13,UPLF
            mov     LCDX,#0
            ljmp    LCDDOXY
UPLF:       cjne    a,#10,UPASCII
            xrl     LCDY,#1
            ljmp    LCDDOXY
UPASCII:    ljmp    LCDDW
```

```
;
;************** INT0
INT0X:      push    ACC             ;When CPLT\ or MODE\
            mov     a,ENCF
            jb      ACC.4,$+9
            orl     ENCF,#00010000b
            mov     KEYDLY,#40      ;*0.5 mSec Debounce
;
            setb    IE.3            ;Enable Timer1, Disable INT0
            clr     IE.0
            sjmp    INTRET
;
;************** INT1
INT1X:      push    ACC             ;edge Trigger Counter
            mov     a,COUNT
            add     a,#1
            mov     COUNT,a
            jnc     INTRET
            inc     COUNT+1
            sjmp    INTRET
;
;************** TIMER1
INTT1:      push    ACC
            mov     TI1,#LOW t2048
            mov     TH1,#HIGH t2048
            mov     a,ENCF
            jnb     ACC.4,IT1ENC    ;INT0 pending
            djnz    KEYDLY,IT1ENC
            jb      P3.2,$+6
            mov     ENCK,#1
;
            anl     ENCF,#11101111b ;Clear INT0
            setb    IE.0
;
IT1ENC:     jnb     ACC.7,INTRET    ;ENC OFF then return
            orl     P1,#00011000b
```

```
                jnb     P1.3,IT1AB          ;When ENCA=LO
                orl     ENCF,#00000001b     ;ENCA is HI
;
INTRET:         pop     ACC
                pop     PSW
                reti
;
IT1AB:          jnb     ACC.0,INTRET        ;Also LO
                anl     ENCF,#11111110b     ;When ENCA HI to LO
                jnb     P1.4,IT1BH
;
                inc     ENCSCC              ;Inc Current
                mov     a,ENCSCC
                cjne    a,ENCSC,INTRET
                mov     ENCSCC,#0
                mov     a,ENCCUR+1
                cjne    a,ENCMAX+1,IT1INC
                mov     a,ENCCUR
                cjne    a,ENCMAX,IT1INC
                mov     a,ENCF
                jnb     ACC.6,INTRET
                clr     a
                mov     ENCCUR,a
                mov     ENCCUR+1,a
                sjmp    INTRET
IT1INC:         inc     ENCCUR
                mov     a,ENCCUR
                jnz     INTRET
                inc     ENCCUR+1
                sjmp    INTRET
;
IT1BH:          dec     ENCSCC              ;Dec Current
                mov     a,ENCSCC
                add     a,ENCSC             ;Check Scale=Scale count
                jnz     INTRET
                mov     ENCSCC,a            ;If SC=SCC Clear Scale count
```

```
                mov     a,ENCCUR
                orl     a,ENCCUR+1
                jz      IT1CZ           ;When Current=0
                mov     a,ENCCUR
                jz      $+6
                dec     ENCCUR
                sjmp    INTRET
;
                mov     ENCCUR,#0ffh
                dec     ENCCUR+1
                sjmp    INTRET
;
IT1CZ:          mov     a,ENCF
                jnb     ACC.6,INTRET
                mov     ENCCUR,ENCMAX
                mov     ENCCUR+1,ENCMAX+1
                sjmp    INTRET
;
;
LCDIW:          acall   LCDWAIT         ;PortC Can uses Bit Set/Reset Command
                mov     DPTR,#PIO1+1
                movx    @DPTR,a
                mov     c,ACC.7
                mov     P1.5,c
                mov     a,#11111100b    ;E=Hi,RW=Low,I=Low
LCDSTB:         inc     DPTR            ;PortC
                movx    @DPTR,a
                clr     ACC.2           ;E=Low
                movx    @DPTR,a
                ret
LCDDW:          acall   LCDWAIT
                mov     DPTR,#PIO1+1    ;Port B Data
                movx    @DPTR,a
                mov     a,ACC.7
                mov     P1.5,c
                mov     a,#11111101b    ;E=Hi,RW=Low,I=Hi
```

```
            sjmp    LCDSTB
LCDWAIT:    push    ACC
            mov     TEMP,#50
LCDWL:      clr     c
            acall   LCDR
            jnb     ACC.7,LCDOK
            djnz    TEMP,LCDWL
LCDOK:      pop     ACC
            ret
;
; CarryF Set/Clear=Data/Instruction
LCDR:       mov     DPTR,#PIO1+2    ;LCD Control
            mov     a,#11111110b    ;E=Hi,RW=Hi,I=Low
            mov     ACC.0,c         ;Data/Instr
            movx    @DPTR,a
            rlc     a               ;Save carry ACC.0
            setb    P1.5
            mov     c,P1.5          ;Read BF
            rrc     a               ;BF to ACC.7,ACC.0 to Carry
            push    ACC
            mov     a,#11111010b    ;E=Low
            mov     ACC.0,c
            movx    @DPTR,a         ;Send E=Low
            pop     ACC
            ret
CHR3:       db      00000100b       ;Plus/Minus Sign
            db      00000100b
            db      00011111b
            db      00000100b
            db      00000100b
            db      00000000b
            db      00011111b
            db      00000000b
            END
```

Source Code for TMS320 Signal Processing Program

```
* Average: Summation
* Parameter(ROUTINE,X,Y,SWEEP,POLAR,SCALE,WITHCLEAR:WORD)
*
          DSEG    Data SEGment
ZERO      BSS     1
ONE       BSS     1
MINUS     BSS     1
*
PARAMD    EQU     $          MOVE PROGRAM MEM TO DATA MEM at >0003
ROUT      BSS     1
X         BSS     1
Y         BSS     1
SWEEP     BSS     1
POLAR     BSS     1          (Not used)
SCALE     BSS     1          (Not used)
CLRBUF    BSS     1
PASIZE    EQU     $-PARAMD
*
CX        BSS     1          Current
CY        BSS     1
CSW       BSS     1
ADDR      BSS     1
SUM1      BSS     1
SUM2      BSS     1
ADC       BSS     1
DAC       BSS     1
*
IOCNTL    BSS     1          I/O
TRGEN     BSS     1
TRGDIS    BSS     1
NXTDTL    BSS     1
NXTDTH    BSS     1
*
TEMP      EQU     127
          DEND
```

```
*
*           PORT:
*           PA0: Buffer R/W (I/O)
*           PA1: Input A/D, Output ANALOG Control CPLT\, NXTDWL\, TRGEN, Z, TRG
*           PA2: Input Dynamic Parameters, Output D/A
*           PA7: Load Buffer Address
CPLTB       EQU     4
NXTDTB      EQU     3
TRGENB      EQU     2
ZB          EQU     1
TRGB        EQU     0
*
            AORG    0           RESET Vector
            B       INIT
            B       INTR
*
PARAM       EQU     $
*           Max Array size: chan1 16k words, chan2 16k words
*           X: (64, 128, 256, 512, 1024, 2048)
*           Y: (256, 128, 64, 32, 16, 8)
*           Sweep: Max=65535 (>0FFFF)
*           Polarity: Normal=0,Invert=1
*           Scale: (1/16, 1/8,....,16) =(-4, -3,...., 4)
*           Clear buffer: Non-Zero Clear,else Not Clear
            DATA    1
            DATA    128         X Default 128x128
            DATA    128         Y
            DATA    9999        Sweep
            DATA    1           Polarity=Normal
            DATA    0           Scale=1
            DATA    1           Clear
*
*           Start of program
*
            AORG    >10
INIT        DINT
```

```
        SOVM            Set overflow mode
        BV      CLROV
CLROV   LDPK    0
        LARP    0
        ZAC
        SACL    ZERO
        LACK    1
        SACL    ONE
        ZAC
        SUB     ONE
        SACH    MINUS
*
*       INIT I/O
*
        LACK    >0018   OFF  CPLT\ NXTDWL\ TRGEN Z TRG
        SACL    IOCNTL
        OUT     IOCNTL,PA1
*
*       Read PARAMETER To Data RAM
*
        LARK    0,PARAMD
        LARK    1,PASIZE-1
        LACK    PARAM-1
LRPARA  LARP    0
        ADD     ONE
        TBLR    *+,1
        BANZ    LRPARA
        LARP    0
*
        LAC     CLRBUF  Clear Buffer RAM ?
        BZ      START
*
*       Clear Buffer RAM
*
        LAC     ONE,14  Count >4000 Words
        OUT     ZERO,PA7 Load address 0
```

```
LCLR    OUT     ZERO,PA0    Chan 1
        OUT     ZERO,PA0
        OUT     ZERO,PA0    Chan 2
        OUT     ZERO,PA0
        SUB     ONE
        BNZ     LCLR
*
*       Start ACQ
*
START   ZAC
        SUB     X
        SACL    ADDR        Buf pointer
        ZALS    Y
        SACL    CY
*
        ZALS    IOCNTL
        SACL    TRGDIS      Trig Disable with TRG Lo
*
        LAC     ONE,TRGENB  Trig Enable and TRG Hi
        SACL    TRGEN
        LAC     ONE,TRGB
        OR      TRGEN
        OR      IOCNTL
        SACL    TRGEN
*
        LAC     ONE,NXTDTB  NXTDWL Lo
        XOR     TRGEN
        SACL    NXTDTL
        LAC     ONE,NXTDTB  NXTDWL Hi
        OR      TRGEN
        SACL    NXTDTH
*
*       Loop for DEPTH
*
LOOPY
        ZALS    ADDR
```

```
                ADD     X
                SACL    ADDR
*
                ZALS    SWEEP
                SACL    CSW
*
*               Loop for SWEEP
*
LOOPSW
                OUT     ADDR,PA7    Load Buf ADDRESS
                ZALS    X
                SACL    CX          Current
*
                OUT     TRGEN,PA1   Enable Trig
*
*               Loop for LENGTH
*
LOOPX
*
WCH1    BIOZ    RDCH1
                B       WCH1        Wait Chan1 Completed
RDCH1   IN      ADC,PA1     Read and ADD/SUB
                IN      SUM1,PA0
                LAC     POLAR
                BZ      NORM1
                ZALH    SUM1
                SUBH    ADC         ***OV Flag?
                B       WRCH1
NORM1   ZALH    SUM1
                ADDH    ADC         ***OV Flag?
WRCH1   SACH    SUM1
                OUT     SUM1,PA0
*
                OUT     NXTDTL,PA1  Send NXTDWL\
                OUT     NXTDTH,PA1
*
```

```
WCH2    BIOZ    RDCH2
        B       WCH2        Wait Chan2 Completed
RDCH2   IN      ADC,PA1
        IN      SUM2,PA0
        LAC     POLAR
        BZ      NORM2
        ZALH    SUM2
        SUBH    ADC         ***OV Flag?
        B       WRCH2
NORM2   ZALH    SUM2
        ADDH    ADC         ***OV Flag?
WRCH2   SACH    SUM2
        OUT     SUM2,PA0
*
        OUT     NXTDTL,PA1  Send NXTDWL\
        OUT     NXTDTH,PA1
*
        LAC     SUM1,8      Concat Chan2,1
        SACH    DAC
        LAC     MINUS,8
        AND     SUM2
        AND     MINUS
        OR      DAC
        SACL    DAC
        OUT     DAC,PA2     Send DAC
*
        ZALS    CX
        SUB     ONE
        SACL    CX
        BNZ     LOOPX       Loop for LENGTH
*
        OUT     TRGDIS,PA1  Disable Trig
        ZALS    CSW
        SUB     ONE
        SACL    CSW
        BNZ     LOOPSW      Loop for SWEEP
```

```
        ZALS    CY
        SUB     ONE
        SACL    CY
        BNZ     LOOPY       Loop for DEPTH

        LAC     ONE,CPLTB   Send Completed Signal
        XOR     IOCNTL
        SACL    IOCNTL
        OUT     IOCNTL,PA1

        B       $
INTR    B       $
        END
```

Appendix D

Operating Instructions

Introduction

Operation of the signal averager is simple and straightforward because the operating procedures are of menu-driven style. A rotary encoder is used to select the desired function similar to the mouse used on the desktop PC. A liquid crystal display shows what operation is going on. For any analysis, signals are connected to the front and rear panel connectors. The required signals are input signal, trigger signal and dwell signal. Input sensitivity and other related parameters are set by choosing available options on the menu.

In this section, a description of the controls, connectors, and switches is firstly presented. After that, operating procedures are described based on the sequential illustrations of the LCD.

Front Panel

1. Liquid Crystal Display (LCD)

20 columns x 2 lines LCD is used to display the selectable menu items and alert messages. The discussion of each messages is presented in operating procedures section.

2. Rotary Encoder

An optical encoder is used as a manual input device. Menu selection and cursor movement are controlled by user via this encoder.

3. Mode Switch

This push-button switch is used to confirm any item selection chosen by user and to interrupt the processing sequence.

4. Signal Input Connectors

Signals to be processed are applied to these connectors.

5. Signal Overload

This indicator flashes whenever the applied signal exceeds the full scale of ADC input.

6. Signal Input Offset

When the offset switch is activated, the offset function is enabled. An offset range of ± 1.5 full scale is obtained by the offset control knob.

7. Trigger Input Connector

A trigger signal synchronized with the input signal is applied to this connector to initiate data sampling sequence.

8. Trigger Threshold

The trigger threshold can be set any point over a range of ± 4 V.

9. Trigger Slope

This toggle switch allows triggering on either the negative-going or positive-going edge of an applied trigger.

10. Trigger Indicator

This indicator flashes to indicate an incoming of trigger signal which is effected by trigger threshold and slope.

11. Ready Indicator

This indicator lights to indicate that the signal averager is ready to process the signals. In other word, trigger signal is allowed to activate data sampling sequence.

Rear Panel

1. Dwell Input connector

External dwell signal is applied to this connector. Dwell time is controlled by this signal.

2. Dwell Indicator

This indicator flashes to indicate an incoming of dwell signal.

3. Signal Output Connectors

Two output signals of processed signals are provided at this connector. These signals can used to drive a typical dual channel oscilloscope.

4. Z Output Connector

This output signal is provided to give brightness modulation of processed signals on an oscilloscope equipped with blanking capability.

5. Trigger Output Connector

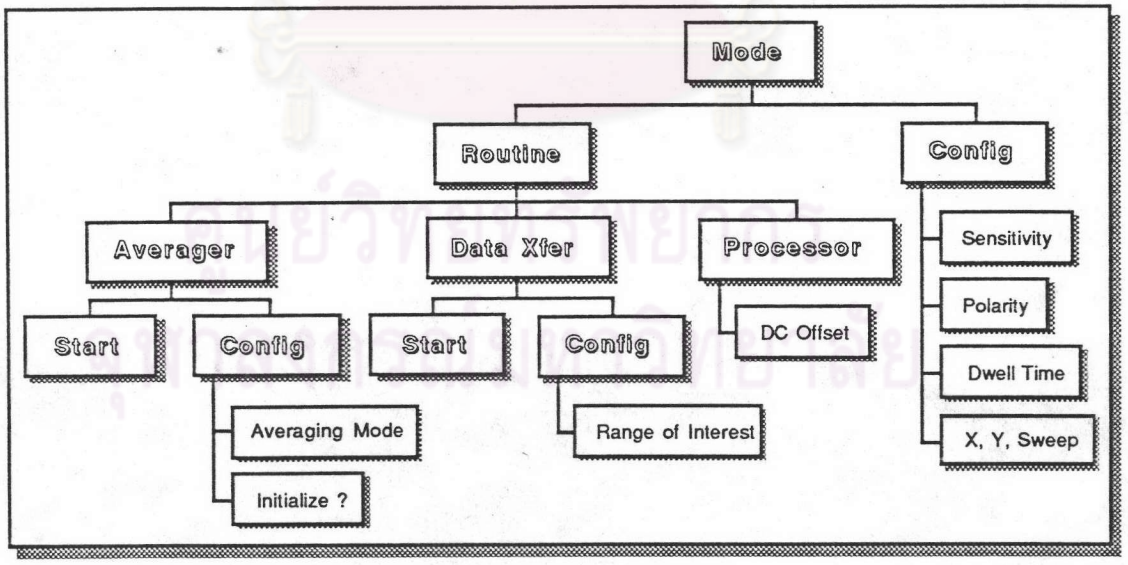
A trigger signal synchronized with the output signals is provided to this connector to initiate the horizontal sweep of a display oscilloscope.

6. RS232 Interface Connector

The processed data can be transferred to a host computer by connecting this connector to a standard RS232 connector equipped with the computer.

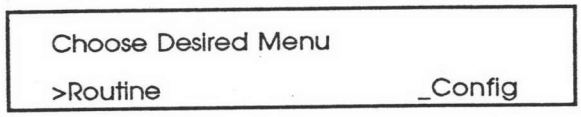
Operating Procedures

The operating procedures of the signal averager are mainly controlled by a menu-driven program. Most of control panel devices are replaced by an LCD panel display and a optical encoder. Operator simply changes any parameters by selecting desired item and then confirming by a click of button. The sequence of operations can be divided in menu sections by determining their functions. Block diagram of operating menu is shown in Fig. D-1. The description of each unit is presented in the following sections.



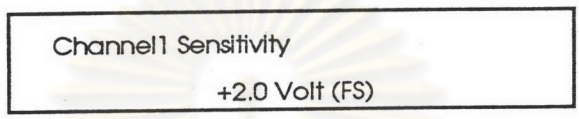
Two notations are used in this section to indicate the current status on the display: ">" indicates the default item and "_" shows other selectable options.

1. Mode Menu

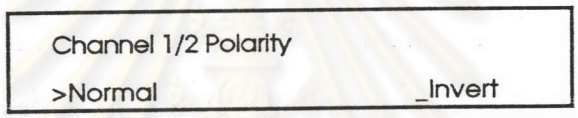


This section is initiated by the power-up operation. Two options are provided. Choosing "Routine" to start the routine menu or "Config" to change the system parameters.

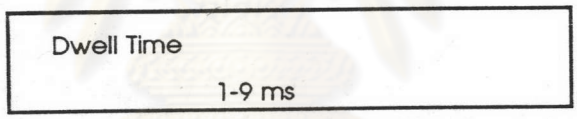
2. Configuration Menu



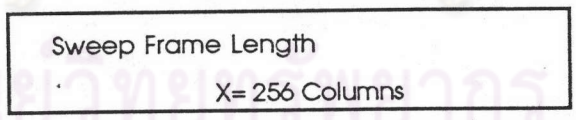
Choose the input sensitivity of channel 1 and channel 2 respectively. The options are ±0.2, ±0.4, ±0.8, ±2, ±4, and ±8 volt full-scale.



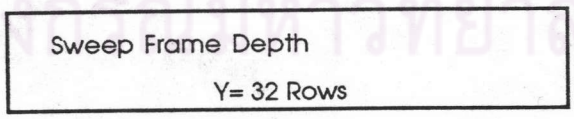
Choose the input polarity of channel 1 and channel 2 .



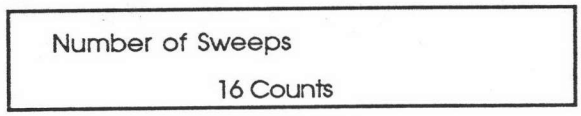
Choose the range of the external dwell time from 5-99µs, 100-999µs, 1-9ms, 10-99ms, and 100-999ms. This parameter is used to select the suitable cut-off frequency of the low-pass filter.



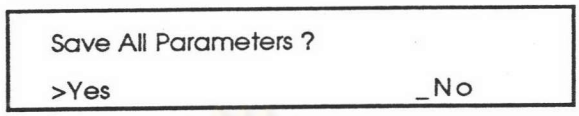
Choose the sweep frame length from 64, 128, 256, 512, 1024, and 2048.



Choose the sweep frame depth from 1 to the maximum value allowed which depends on the previously selected sweep frame length. The options are ranged from 256, 128, 64, 32, 16, and 8 respectively.



Choose the number of sweep from 1 to 256.

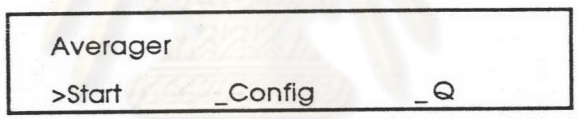


Confirm to use these parameters.

3. Routine Menu

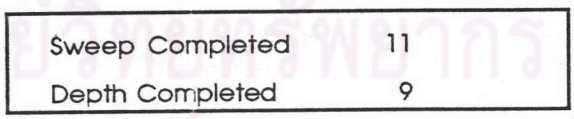
Choose the desired routine to average, process or transfer the data. Choose "Q" to return to the previous menu.

4. Averager Menu

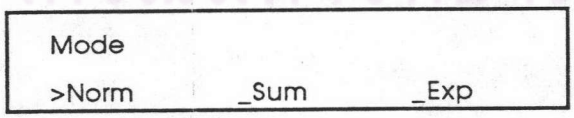


Choose "Start" to start the averaging process , "Config" to change the averaging parameters, or "Q" to return.

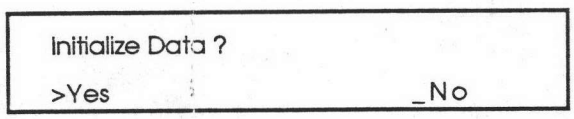
Choosing "Start", the display shows number of elapsed sweeps and number of completed rows.



Choosing "Config", the display presents with some averaging parameters.

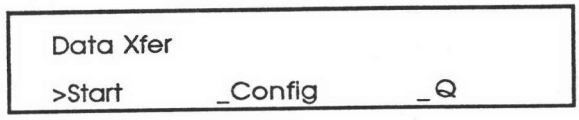


Choose the desired averaging mode: normalized summation, linear or exponent.



Choose either to clear data memory before starting the acquisition or not.

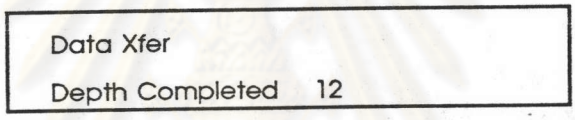
5. Data Transfer Menu



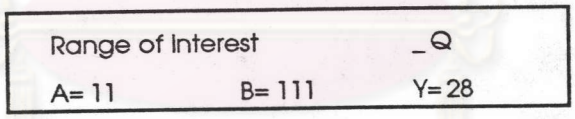
Choose "Start" to start the transferred process , "Config" to change the parameters, or "Q" to return.

Choosing "Start", the display shows that it is waiting for the connection from the host computer.

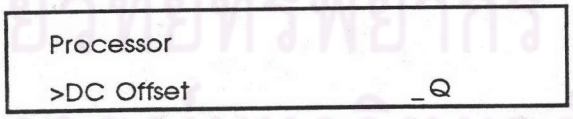
After the connection is established, the number of rows that are successfully transferred is shown.



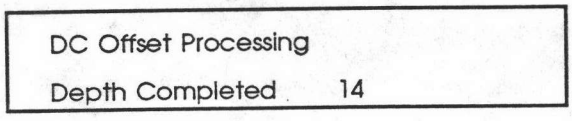
Choosing "Config", the display presents the range of interest defined by the minimum address (A) and the maximum address (B). Press mode button to select the desired parameter and use the encoder to modify its value.



6. Post Processor Menu



Choose "DC Offset" to start the DC offset process or "Q" to return. The number of processed rows is displayed as the process is in progress.



Appendix E

TMS32010 Instruction Summary

Table E-1 Accumulator Instructions

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
ADD	Add to accumulator with shift	1	1
SUB	Subtract from accumulator with shift	1	1
LAC	Load accumulator with shift	1	1
SACL	Store low-order accumulator bits	1	1
SACH	Store high-order accumulator bits with shift	1	1
ADDH	Add to high-order accumulator bits	1	1
ADDS	Add to accumulator with no sign extension	1	1
SUBH	Subtract from high-order accumulator bits	1	1
SUBS	Subtract from accumulator with no sign extension	1	1
SUBC	Conditional subtract (for divide)	1	1
ZALH	Zero accumulator and load high-order bits	1	1
ZALS	Zero accumulator and load low-order bits	1	1
LACK	Load accumulator immediate	1	1
ABS	Absolute value of accumulator	1	1
ZAC	Zero accumulator	1	1
XOR	Exclusive OR with accumulator	1	1
AND	AND with accumulator	1	1
OR	OR with accumulator	1	1

Table E-2 Auxiliary Registers and Data Page Pointer Instructions

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
SAR	Store auxiliary register	1	1
LAR	Load auxiliary register	1	1
MAR	Modify auxiliary register and pointer	1	1
LDPK	Load data memory page pointer immediate	1	1
LDP	Load data memory page pointer	1	1
LARK	Load auxiliary register immediate	1	1
LARP	Load auxiliary register pointer immediate	1	1

Table E-3 T Register, P Register, and Multiply Instructions

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
LT	Load T Register	1	1
LTA	Load T Register and accumulate product	1	1
LTD	Load T Register, accumulate product, and move data in memory forward one address	1	1
MPY	Multiply with T Register, store product in P Register	1	1
PAC	Load accumulator from P Register	1	1
APAC	Add P Register to accumulator	1	1
SPAC	Subtract P Register from accumulator	1	1
MPYK	Multiply T Register with immediate operand, store product in P Register	1	1

Table E-4 I/O and Data Memory Operations

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
IN	Input data, from port	2	2
OUT	Output data to port	2	2
TBLR	Table read from program memory to data RAM	3	1
TBLW	Table write from data RAM to program memory	3	1
DMOV	Shift contents of data memory forward	1	1

Table E-5 Control Instructions

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
LST	Load status register	1	1
SST	Store status register	1	1
NOP	No operation	1	1
DINT	Disable interrupt	1	1
EINT	Enable interrupt	1	1
ROVM	Reset overflow mode	1	1
SOVM	Set overflow mode	1	1
POP	Pop stack to accumulator	2	1
PUSH	Push stack from accumulator	2	1

Table E-6 Branch Instructions

MNEMONIC	DESCRIPTION	NO. OF CYCLES	NO. OF WORDS
BANZ	Branch on auxiliary register not zero	2	2
BV	Branch on overflow	2	2
BIOZ	Branch on BIO = 0	2	2
B	Branch unconditionally	2	2
BLZ	Branch if accumulator < 0	2	2
BLEZ	Branch if accumulator <= 0	2	2
BGZ	Branch if accumulator > 0	2	2
BGEZ	Branch if accumulator >= 0	2	2
BNZ	Branch if accumulator <> 0	2	2
BZ	Branch if accumulator = 0	2	2
CALL	Call subroutine immediate	2	2
CALA	Call subroutine from accumulator	2	1
RET	Return from subroutine	2	1

VITA

Phisit Damrongkijkarn was born in Bangkok on November 10, 1965. He graduated on 1986 from Department of Physics, Chulalongkorn University. His previous project was "Finite-Difference Eigenvalue Calculation" which he used microcomputer for calculating the energy of 1-dimensional quantum mechanics problems. His general interests are in the applications of microcomputer with particularly emphasis in Physics experiments.