INSTRUCTION MANUAL

ADL-6408

ANALOG INTERFACE CARD

Technical Hardware Inc.
P.O. Box 9101
Pembroke Pines, Florida 33084

CONTENTS

I. INTRODUCTION

A complete ADL-6408 data logging system results from the integration of a Commodore 64* Computer system (BASIC Computer, television set, and printer) and a Technical Hardware ADL-6408 hardware/software package. The ADL-6408 hardware is an 8 channel, 8 bit analog-to-digital converter (ADC) that provides adequate resolution and speed for practically all monitoring applications. The software package includes software to drive the hardware Real Time Clock as well as an interrupt driven ADC controller and a BASIC interfacing program. The BASIC program can be shortened or deleted entirely if you wish to obtain more memory for data storage or for other programs. The C-64 Computer system provides power, memory, computing ability, and output interfacing capability for the ADL-6408 package. The complete integrated system provides versatile data logging capabilities unmatched by far more expensive systems.

II. INSTALLATION

1. Turn off your C-64 Computer.
2. Jumper pins 29 to 12 and 28 to 11 in order to enable the reference to the ADC.
3. Insert the ADL-6408 with the components upright into the expansion port with the connector making firm contact.
4. Turn on the computer. If it doesn't operate normally, readjust the circuit board in the expansion port connector.
5. Connect a +0v to +4v source to pin 20 (input 1) of a 40 pin mass terminated connector (40 position, .100" spacing edgeboard connector). The low side of the source should be connected to pin 21 (ground). Insert the connector on to the ADL-6408 board.
6. Type in the following program or load "PERFORMANCE TEST" from tape or disk.

```
10 POKE 57342,0
20 POKE 57341,0
30 A=PEEK(57342) AND 128
40 IF A=0 THEN GOTO 30
50 LS=PEEK(57341)
60 LS=255-LS
70 PRINT (LS*2)/100;
80 FOR T=1 TO 500:NEXT T
90 GOTO 10
```

7. RUN the program. The voltage corresponding to your source should be repetitively displayed. If desired, you can calibrate to the precisely correct voltage by adjusting the variable trimpot.

8. Load the BASIC program with a 'LOAD "ADL6408.BAS",8 for disk or

'LOAD "ADL6408.BAS",1 for tape.

9. RUN the program. The machine language program will now be loaded.

10. Try the different menu options. The voltage displayed for channel 0 should be the same as obtained in your test program.

11. Connect each of your desired signals to one of the multiplexer input pins (see Connector Description). The system has a working range of 0 to +5.000 volts. Voltages in excess of 5 volts will damage the A/D converter. Consequently, attenuation similar to that shown in Figure 2 should be used for any signals that may exceed +5 volts. More sensitive voltage ranges can be achieved by using off-board amplifiers shown in Figure 3. Small currents can be measured by using amplifiers shown in Figure 4.

III. HARDWARE

The ADL-6408 board consists of four primary sections; controller, address decoder, input/output ports and A/D converter. The controller consists of a 74LS138 1 of 8 decoder. The address decoder is comprised of a 74LS00 and 74LS85. The address decoder utilizes the 8 low order address lines from the C-64 Computer as well as the $DFXX/ or $DEXX/ select lines (jumper selectable) to establish a single block of 4 addresses for the ADL-6408 to utilize. Address A7 as well as the Phase 2 clock are connected to the enable input for the 74LS85, a 4 bit digital comparator. Additionally, address lines A5, A4, A3 and A2 are connected as inputs to the 74LS85. Other inputs to the comparator are from a four (4) position DIP switch. This switch allows selecting different operating addresses for the ADL-6408. If the Dip Switch and the address sent from the C-64 match, the output line of the comparator will switch high. This high output is connected to an enable line for the 74LS138. Addresses A6 and $DFXX/ or $DEXX/ form a second enable for the 74LS138. Addresses A0 and A1 are connected to the 74LS138 address inputs. These addresses, along with the enables, drive the different I/O devices within the ADL-6408.

The ADC is an eight (8) bit SUCCESSIVE APPROXIMATION Analog to Digital converter. This type of converter was chosen for speed as well as high reliability. The input multiplexer is contained internal to the ADC in order to provide good temperature stability as well as input tracking. The input channel desired is placed on the DATA BUSS of the ADL-6408 and the ALE (Address Latch Enable) is strobed (Address $DFFE or 57342). This stores the desired channel into the ADC. The convert line (Address $DFFE or 57342) is then strobed and approximately 65 microseconds later the ADC will complete the conversion in progress. The data is then read from the ADC and stored in the data memory by executing a read from address $DFFD or 57341. The ADC is now idle until the next load pulse. The data buss of the ADC is not capable of driving the C-64 buss directly, therefore a 74LS368 is used to buffer the ADC from the C-64 data buss. Since these

buffers are inverting, the data received is inverted by the interrupt driven machine code program prior to being stored into the data memory.

IV. INTERRUPT ROUTINE

All the major data acquisition, and data storage functions are performed by a machine language program residing near the top of BASIC RAM memory. This program immediately follows the BASIC "ADL-6408.BAS" program on the disk or tape. It is loaded with the LOAD "ADL6408.OBJ",8,1 command for disk or LOAD "ADL6408.OBJ",1,1 for tape. After the program is loaded and run, the IRQ interrupt is enabled.The microprocessor will jump to the location indicated in memory bytes 788-789 every sixtieth (1/60) of a second. The startup section of the BASIC program initializes the above memory locations to point to the machine code program. This interrupt-driven program normally begins at location $9D00. A complete listing of this program is included. Time keeping is accomplished by utilizing one of the real time clocks in the C-64 hardware. A 16 bit register (DACO) is decremented at one second intervals to determine when a new analysis is required. An analysis cycle is initiated by loading the first input channel number into the ADC multiplexer and toggling the CONVERT pin of the ADC. When the conversion is completed, the one byte answer is stored in the next available memory location and the next required input is loaded into the multiplexer. After all required analog inputs have been digitized, the data available flag (DAFLG) is set. This indicates to the BASIC program that new data is available. Thus, this machine language program performs all the necessary data-logging functions with no need of assistance from the BASIC program. If the memory storage location exceeds the end of the storage area (EDTA), the data pointer will jump to the data wrap-around location (DWRP) and notify the user of the wrap-around condition. This is normally loaded with the data begin location (BDTA) but can be set to any other desired location.

V. BASIC SOFTWARE

A. General

The BASIC software is designed as a convenient interface to the user. It is not essential to data acquisition. The BASIC software consists of a set-up mode to set computer parameters and place the ADL-6408 system into an idle condition. A "menu" is then used to select various options. Each of these options will be discussed.

1. Set Time

This option allows you to set the month, date, hour, minute and second of the C-64 Real Time Clock. Hours should be entered on a 12 hour cycle basis. The time is entered into the appropriate memory locations and hardware registers and updated by the interrupt routine. It will then be displayed in the lower right corner of the screen. Additionally, the clock time will be displayed with an 'AM' or 'PM' indication.

2. Set Parameters

The desired analysis channels may be entered in any order or may even be repeated. Selection of analysis channels is accomplished simply by typing the desired channel number and depressing "RETURN". Entry of any number greater than 8 will terminate this option. After completion of channel entry the user is requested for a 'YES/NO' response for the LIST FILE mode. This mode allows the user to store raw or scaled data on disk or tape as the data is being acquired. This feature allows for transportable data for later analysis. If the user wishes to establish a list file then the prompts need only to be answered. The format of the LIST DATA FILE is found in the APPENDIX under FILE FORMATS. In addition to selecting channels for analysis, this option also includes the ability to define the function of each channel by assigning a name to each channel selected for analysis. The assigned name may be up to 5 characters in length and in any combination of alpha or numeric characters. This allows for ease of data identification at a later time. The function is optional and is selected after all desired analysis channel numbers have been entered. The user is then prompted for an input 'TRIGGER' value. This option allows for the input port to be strobed to start the data acquisition cycle. The input port is checked for a specific bit pattern at the elapse of the data interval. If the value is present at the input port, then data acquisition commences. At the time the port is read, if the bit pattern is not equal to the value set in this option, then data acquisition will not commence until another data interval has elapsed and the port is again read. The next option is analysis interval. The analysis interval entry should be made as "hours (1-18), minutes (0-59), and seconds (0-59) "RETURN". A maximum of 18 hours between analyses is allowed. If the total time interval is five seconds or less, and the number of channels to analyse is less than 5, the program will analyze all the channels entered in option 2 and then remain idle until the time interval elapses, and the desired channels will again be analyzed. If the analysis entry is less than five seconds, data acquisition will proceed on a continuous basis with each data point requiring one (1) interrupt cycle to acquire all data channels. If more than 6 channels are being analyzed, the time interval must be at least 10 seconds. This delay requirement is due in part to the delay of the screen update of new data. The program will now return to the main menu.

3. Set Alarm Limits

This option allows you to set high and low limits for any channel. Enter a value between 0.000 and +5.00 for your highest set point limit. Enter "5.1" for no high limit. For the low set point limit, enter a "-5" for no limit. This procedure will then cause an asterisk to be printed ahead of the value obtained whenever the high or low limit is exceeded. Enter a "0, 1, 2, 3, 4, 5, 6 or 7" for the output alarm. Exceeding a preset value will cause the corresponding digital alarm output line to go low and remain low. A zero (0) will cause an overrange condition to be printed but not output to the digital output lines. Entry of a '8' for the selected alarm will drive all alarms high or off regardless of alarm status. Enter "99" to finish this option and return to the main menu. During operation, all 8 output lines will be reset (go high) when the "R" key is depressed.

4. Start Analysis

Data acquisition will begin one data interval after the START option is selected. Memory locations beginning at BDTA+26 (normally 24602) will then be filled with a one byte data for each input channel with a delay between groups previously determined by option 2. If "Continue" is selected, data acquisition will simply continue from the last storage location. When DAFLG is set by the interrupt cycle, the measured data for each selected channel will be displayed on the screen in tabular format. The memory pointer location and time will also be displayed. The present time will be entered into the memory locations reserved for the start time. This allows the data to be saved on disk and still enable subsequent printouts to indicate the correct time of data acquisition. The menu will reappear when the "Q"uit key is depressed. The printer can be toggled on and off by depressing the "P" key. When the printer is turned on, it will first print an input channel header. It will then print the data collected for each channel beginning with the start of the data collection sequence and ending with the next entry of the "P" key. If the printer routine is interrupted with the "P" key, it will terminate printing and return to the aquisition display. If the "P" key is again depressed, then a new data header and data will be printed untill the entry again of the "P" key. Additionally, if a LIST FILE was selected, then directly to the left of either the RAW or SCALED data headers an asterick(*) will be displayed indicating the type of data being sent to the list device.

5. Save Data on Disk/Tape

The selection of this command will initate the creation of a file

on disk/tape for storage of data. The first portion of the file contains channel names. If channel names were not assigned, then the selected analysis channels will contain asterisks (*) and hyphens(-) (*---*) to indicate that no name was assigned. The second portion of the file created is comprised of the data memory pointer location as well as the actual data. Thus the data is loaded at a later date by a previously assigned data file name. The file format for memory data files is the ".DAT" extension while the list data utilizes a ".LST" extension to the original file name.

6. Print Old Data

This routine moves all the analysis parameters (starting time, channel names (if assigned), analysis interval, data channels, etc.) into the working registers and then proceeds to print all the data. Additionally, this routine is also used to establish HARD COPY for list mode data files. The type of data printed is chosen by the user by an entry prompt. If the user selects the list data format then the user is asked for tape or disk. The user is then asked for the filename. Upon location of the filename the data is printed directly from the storage device. All previous parameters set are also printed in a tabular form for easy reference. This function is terminated upon file end and control returns to the main menu.

7. Load Old Data

This is the most convenient method of loading old data for a printout. If the previous saved data was assigned channel names then these channel names will be loaded with the previous data. If no channel names were assigned to the saved data then each channel name will be assigned an asterisk and hyphen label (*---*) to indicate that no name was previously assigned.

8. Examine Memory

System memory can be examined by entering the decimal memory address and pressing RETURN. The routine displays the contents of the primary data storage area values. Pressing "M" after entry into this routine, will allow you to examine all the data memory locations. Pressing any other key returns you to the main menu.

9. System Status

This command is a multifaceted system maintenance command which allows three different functions.

A. The alarm lines can be 'STROBED' at 1 second intervals, alarm lines can be assigned names up to 5 characters each and the input port can be read and displayed on the screen. This option is selected by using the 'T'est function.

B. Alarm status, data memory used and values assigned to alarms for HIGH and LOW limits can be examined by selecting the 'S'tatus option. The current level definition is printed on the screen next to the alarm number. Exit to main menu is done by pressing any key.

C. Disk commands as well as disk status and directory for drive 8 can be selected from the sub menu with this option. Entering a 'D' will display the disk (8) directory. Entering a '>' or '.' will set the system to wait for a standard C-64 disk command. The command is executed upon depression of the RETURN key. Entering a 'Q' returns to main menu. Entering an 'S' will return disk status to the screen.

VI. TROUBLESHOOTING

Most problems with the ADL-6408 board arise from applying excessive voltage to the ADC multiplexer inputs. If it is determined that one or more inputs are defective then the ADC must be replaced. Make certain the new device is inserted with the locating dot facing the user connector end. Slight readout errors can be corrected by adjusting the voltage reference trimpot. For other problems, follow the procedure shown below:

1. Turn computer off and then on to erase program.
2. POKE 57342, 0:POKE 57341,0 to activate RUN and open channel 0.
3. Apply a 1 to 4 volt signal to channel 0 (connector pin 20).
4. Do a "PEEK (57341)" command. The output data value should be between 50 and 200. Also, it should change when the input voltage is changed. . Otherwise, the software program is not functioning correctly. Try reloading. If this test doesn't work, return the board for repair

REPAIR FACILITY

If repair of the ADL-6408 is required, the unit should be returned to:

TECHNICAL HARDWARE INC.
P.O. BOX 9101
PEMBROKE PINES, FL. 33084

LIMITED WARRANTY

Technical Hardware Inc. (THinc) warrants the ADL-6408 to be free of defects in workmanship or materials for a period of one (1) year from receipt. Technical Hardware will repair or replace the ADL-6408 (at the option of Technical Hardware Inc.) if found defective during the warranty period when returned to the above address. No other warranties are implied or expressed.

MEMORY MAP

GENERAL

ADDRESS	DESC.
2049 ($801)	Begin of BASIC program
23808 ($5D00)	End of BASIC program
40192 ($9D00)	Start of Machine Code program
40959 ($9FFF)	End of Machine Code program
24602 ($601A)	Start of Data Storage
39936 ($9C00)	End of Data Storage

DATA STORAGE VARIABLES

VARIABLE	ADDRESS	FUNCTION
File Type Flag	24576($6000)	Holds definition of data file type
Data Interval	24577($6001)	Low byte Daint
Data Interval	24578($6002)	High byte Daint
Month	24579($6003)	Month of Data Acquire
Day	24580($6004)	Day of of Data Acquire
Hour	24581($6005)	Hour of Data Acquire setup
Min	24582($6005)	Minute of Data Acquire setup
Sec	24583($6006)	Second of Data Acquire setup
Achan	24584($6007)	Analysis channels
	24600($6018)	future use
Data store	24602($601A)	Begin of Data Storage

FILE FORMATS

The file formats for ADL-6408 are the same for both the list (.LST) and data (.DAT) files. All file parameters are stored as ASCII characters either numeric or alpha. It is recommended that any use of the data files other than in the ADL-6408 program, that the user

adhere to the input constraints of the C-64.

FILE (DATA)

HEADER:

Channel count (K)	number indicating number of channels to be acquired.
EDTA,low	low byte of end of data storage
EDTA,high	high byte of end of data storage
BDTA,low	low byte of start of data storage
BDTA,high	High byte of start of data storage
File Type	0=raw data, 10=list data,RAW 20=list data, SCALED
MONTH	month of data acquire
DAY	day of data acquire
HOUR	hour of data setup
MIN	minute of data setup
SEC	second of data setup
DAINT,low	low byte data interval
DAINT,high	high byte of data interval
CHAN	16 bytes for up to 16 channels
SYNC	4 nulls for syncronization
NAME	maximum of 16-5 char. names for channels.
DATA	Stored quantity based on channel count

DATA FORMAT:

All data is stored as blocks of 1 byte per channel and 3 bytes for hour, min and sec. The time of acquire is stored prior to the new channel data. Channel data will follow immediately after the acquisition time. EDTA is constantly updated to provide a memory pointer for new data storage. Thus recovering EDTA from disk or tape as well as BDTA (begin of data storage) will ensure proper data recovery and storage.

NOTE: All data files created using option five (5) will exhibit an extension of '.DAT', while list files will have a '.LST' extension. Option seven (7) will ONLY load data files with the '.DAT' extension. Files with the '.LST' extension may only be loaded from the print routine (option 6). All files are stored as ASCII equivelent BINARY values with the exception of list files which are stored as ASCII floating point numbers.

ADDRESS SELECTION

The ADL-6408 contains the capability of selecting the operating address for all I/O ports on board via a 4 position DIP switch designated as S-1. This switch, for each position change (16 position

combinations possible), will select a different block of 4 addresses for operation of the ADL-6408. Multiple ADL-6408 systems may be utilized with each having a different DIP switch setting. Additionally, there are 2 sets of 3 jumper pins that allow the user to select 4 discreet blocks of 64 operating addresses (16 blocks of 4 addresses). Thus the ADL-6408 can be placed anywhere in the $DF80 to $DFFF or $DE80 to $DEFF address space. The jumpers are shown in figure 1.

The board, as received, is set for operation at locations $DFFC-$DFFF. This setting will seldom need to be changed. If change is necessary, the following procedure must be followed to ensure proper operation of the ADL-6408

DISK SYSTEMS:

1. Using LOAD "ADDRESS.MOD",8", load the address change program into memory.

2. Run the program. The directions for use are presented first. The machine code program is also loaded into memory.

3. Enter the new address in decimal for the new DIP switch setting. The program will make the necessary changes and rewrite the machine code file back to disk.

4. The modification is now completed and need not be changed again unless the DIP-SWITCH or jumpers are changed.

5. If the ADL-6408 does not operate properly after modification then carefully inspect the DIP switch and jumpers for proper settings. This will usually correct the problem. Go back to step 1 and start over.

TAPE SYSTEMS:

1. Using LOAD "ADL6408.OBJ",1,1 load the machine code program into memory.

2. Using the listing supplied for the machine code program, locate the addresses that reference the I/O ports (i.e. those addresses that contain references to $DFFC,FD,FE,FF). Note these addresses on a piece of paper.

3. Using "POKE" statements, change the memory locations noted above to the new address.

Example: If the new DIP-SWITCH setting is such that the new address is $DF00 and the current location in memory is $9DFE, then POKE 40446, 00 will accomplish the necessary change.

4. After each change, the location should be "PEEK"ed to insure the proper change has occurred. If the change is not proper, then step three (3) should be repeated.

5. Once the changes are done, the file should be saved on a new tape. The new program, as well as a copy of the BASIC operating program should be saved on a new operating system tape.

6. The modification is now completed and need not be changed again unless the DIP-SWITCH or address jumpers are changed.

7. If the ADL-6408 does not operate properly after modification then copy ADL6408.OBJ from the distribution tape to memory and delete the file created earlier. This will replace the original program. Go back to step 1 and start over.

DIP SWITCH SETTINGS

SW-1	SW-2	SW-3	SW-4	ADDRESS, HEX	ADDRESS, DEC
ON	ON	ON	ON	$DFC0	57280
OFF	ON	ON	ON	$DFC4	57284
ON	OFF	ON	ON	$DFC8	57288
OFF	OFF	ON	ON	$DFCC	57292
ON	ON	OFF	ON	$DFD0	57296
OFF	ON	OFF	ON	$DFD4	57300
ON	OFF	OFF	ON	$DFD8	57304
OFF	OFF	OFF	ON	$DFDC	57308
ON	ON	ON	OFF	$DFE0	57312
OFF	ON	ON	OFF	$DFE4	57316
ON	OFF	ON	OFF	$DFE8	57320
OFF	OFF	ON	OFF	$DFEC	57324
ON	ON	OFF	OFF	$DFF0	57328
OFF	ON	OFF	OFF	$DFF4	57332
ON	OFF	OFF	OFF	$DFF8	57336
OFF	OFF	OFF	OFF	$DFFC	57340

NOTE: The DIP switch when in the ON position will cause a LOW at the address comparator. An equivelent LOW must be received from the C-64 address buss to have a positive compare. This must be kept in mind to understand the DIP switch configuration.

JUMPER SETTINGS

JUMPER	BASE ADDRESS RESULT
1-2,5-6	$DFC0 (57280)
1-2,4-5	$DF80 (57216)
2-3,5-6	$DEC0 (57024)
2-3,4-5	$DE80 (56960)

NOTE Jumpers set the significant address for the operating of the ADL-6408. The factory setting is 1-2, and 5-6. This setting addresses the ADL-6408 to operate with the DIP switch settings in the table above. If the jumpers must be changed, then the jumpers will set the BASE address and the DIP switches will set the offset address from the BASE.

EXAMPLE: Jumper setting 1-2,4-5 would set BASE address to $DF80. S1-1 OFF, S1-2 ON, S1-3 ON and S1-4 ON would set OFFSET to 4, thus the operating address would be $DE84 (56964).

USER SCALER ROUTINES

The ADL-6408 incorporates provisions for user data scaling software. The BASIC program contains, in lines 6060 to 6550, locations for scaling equations for each channel. These locations are structured in such a format that all channels set to acquire data will always transition to a given location. This is evidenced by the REM statements heading each scaler routine. The user simply enters his scaler equation in the proper area based on the selected channel. The equation will then scale the data and print the scaled value on the screen during the START ACQUISITION option (4). The user may use any standard Commodore 64 statements and/or functions to accomplish the required scaling. A 'SYS' function may also be used. The variable D(T) is used to carry the RAW data value to the scaler equation. The 'T' is the channel number in a 'FOR-NEXT' loop. If channel 13 is desired then 'T' would equal 13. The variable SC(T) is the scaled variable. An example of a scaler is found below.

EXAMPLE: Scale channel 13 to be the log of the linear value D(T).

```
6480 SC(T)=LOG(D(T))
```

If the user wishes to save the scaled values then a new program should be created on disk or tape. The program should have a name familiar to the user defining the change in the scaler routines.

INPUT/OUTPUT PORTS

The input port of the ADL-6408 is a Low Power Schotkey (LS) TTL input. The input specifications are:

Input HIGH	2.4 Volts minimum @ 40 microamps
Input LOW	0.4 Volts maximum @ 400 microamps

Maximum input voltage is 5.25 Volts. Inputs greater than this value may cause device destruction.

The input port may be used for applications other than the TRIGGER input for the ADL-6408. The user must, however, develop the necessary software to utilize the input data. The input port utilizes 74LS368 devices which are inverting to maintain a high current drive to the Commodore data buss to reduce buss noise. The user must invert any data read from the input port to return to the proper input value. The 6510 microprocessor will accomplish this with the EOR (Exclusive OR) instruction. See the control address table for the read addresses.

OUTPUT PORT

The output port utilizes a 74LS377 8 bit latch to implement the output function. The output specifications are:

Output HIGH 3.75 Volts @ .4 milliamps
Output LOW 0.4 Volts @ 8 milliamps

The latch is controlled by a simple output to the control address shown in the control address table. Care must be taken when driving external devices not to exceed the drive values as device damage or erratic operation may result.

CONTROL ADDRESSES

ADDRESS	FUNCTION	COMMENTS
INPUT MODE		
$DFFC (57340)	Read Input port	Complement of value
$DFFD (57341)	Read ADC Data	Low or High Byte (dependent on Ctrl Latch)
$DFFE (57342)	Read ADC Status	HIGH ADC NOT-BUSY
$DFFF (57343)	No Operation	
OUTPUT MODE		
$DFFC (57340)	Write Output Latch	
$DFFD (57341)	START ADC operation	
$DFFE (57342)	Write ADC MUX ADDR	
$DFFF (57343)	No Operation	

CALIBRATION

The calibration of the ADL-6408 is straight forward. A voltage potential must be connected to channel 0. Once the voltage is

connected to channel 0, type in the test program found in the installation section. Run the program. Your voltage will be displayed on the screen. If the displayed value and the actual value are different, then adjustment of the trim pot is required. Adjust the trim pot SLOWLY until the displayed voltage is the same as the measured value. Calibration is now complete.

ADL-6408 SPECIFICATIONS

ANALOG/DIGITAL CONV.:

TYPE:	Successive Approximation
CONV. RATE:	Approximately 1000 conv/sec
REFERENCE :	External 5.0 Volts, Adjustable
INPUT RANGE:	0- +5.000 Volts
INPUT IMP. :	100K OHMS Min. 10M OHMS Nom.
OUTPUT	8 BITS
ACCURACY:	+/- 1 LSB

INPUT PORT:

TYPE:	74LS368
INPUT RANGE:	.4 Volts max @ 250 microamps for LOW 2.4 VOLTS min @ 20 microamps for HIGH

OUTPUT PORT:

TYPE:	8 BIT latch, 74LS377
OUTPUT RANGE:	Source 3.75 Volts @ .4 milliamperes. Sink .4 Volts @ 8 milliamperes

EXTERNAL POWER:
(USER SENSOR EXCITATION) 5 Volts @ 50 ma. max.

SYSTEM POWER REQ. To Be Determined

* C-64 is a Trademark of Commodore Business Machines

052286

ADL-6408 INTERFACE CONNECTOR

The ADL-6408 interface connector is designed to interface with standard dual 20 pin INSULATION DISPLACEMENT CONNECTOR (IDC) or similiar connector. The spacing is on .100 inch centers. The terminals on the male connector are standard .025" square stake pins.

GROUND	40	1	GROUND
OUT 3	39	2	OUT 2
OUT 1	38	3	OUT 0
OUT 4	37	4	OUT 5
OUT 6	36	5	OUT 7
GROUND	35	6	GROUND
INPUT 0	34	7	INPUT 1
INPUT 2	33	8	INPUT 3
INPUT 4	32	9	INPUT 5
INPUT 6	31	10	INPUT 7
REF OUT	30	11	GROUND
+5 VOLTS	29	12	REF IN
GROUND	28	13	ANALOG CH. 4
GROUND	27	14	ANALOG CH. 5
GROUND	26	15	ANALOG CH. 6
GROUND	25	16	ANALOG CH. 7
GROUND	24	17	ANALOG CH. 8
GROUND	23	18	ANALOG CH. 3
GROUND	22	19	ANALOG CH. 2
GROUND	21	20	ANALOG CH. 1

ADL-6408

INTERFACE CONNECTOR

```
10 REM ADL6408.BAS EDIT 2/9/85
20 POKE 53280,6:POKE53281,06:V$=CHR$(13)
21 IF A=1 THEN GOTO60
22 PRINT "{SC}":GOSUB5960
25 PRINT "  LOAD MACHINE CODE TAPE OR DISK (T/D)":GOSUB5970
26 GET E$:IF E$="" THEN GOTO 26
27 IF E$="T" THEN IF A=0 THEN A=1:LOAD"ADL6408.OBJ",1,1:GOTO 60
28 IF E$="D" THEN IF A=0 THEN A=1:LOAD"ADL6408.OBJ",8,1:GOTO 60
30 REM IF A=1 THEN GOTO60
40 PRINT"{SC}":GOSUB5960 :PRINT "      LOADING MACHINE PROGRAM":GOSUB5970
50 IF A=0 THEN A=1:LOAD"DL6408.OBJ",1,1
60 POKE 51,0:POKE 52,96:POKE 55,0:POKE 56,96:CLR
70 POKE 53280,6:POKE53281,06
80 POKE56334,PEEK(56334) AND 254
90 F=1:G=1:W1=80:GOSUB5780 :W1=80
100 DIM AL(17), CH(17), LD(17), HD(17), L1$(17),C(18),AH(17),AC(17),D$(17)
110 DIM A$(17),D(17),H$(17),SC(17),AA(17),M$(17),SC$(17)
120 Z=40832:POKE(Z+16),255:POKEZ+72,0
130 Z1=40832:Z=Z1:POKEZ+1,60:POKEZ+2,60:POKEZ+4,1:POKEZ+5,0:POKEZ+6,255
140 POKE(Z+3),1:POKEZ+9,255:POKEZ+10,255:POKEZ+11,0:POKE(Z+16),255:POKEZ+7,0
150 FORI=Z+50 TO Z+64:POKEI,0:NEXTI
160 BDTA=Z+65:EDTA=BDTA+2:X=24576:GOSUB710 :POKEBDTA,X2:POKEBDTA+1,X1
170 X=24602:GOSUB710 :POKEBDTA+2,X2:POKEBDTA+3,X1:X=39936:GOSUB710
180 POKEBDTA+4,X2:POKEBDTA+5,X1
190 POKEZ+44,76:REM JUMP (JMP) INSTRUCTION FOR COMPLETION OF IRQ SERVICE ROUTN
195 DATA 31,28,31,30,31,30,31,31,30,31,30,31
200 FORI=1TO8:AA(I)=0:NEXT:FORI=Z+32TOZ+32+11:READ A:POKEI,A:NEXT
210 Z3=PEEK(788):Z4=PEEK(789):POKE788,0:POKE789,157:POKEZ+45,Z3:POKEZ+46,Z4
220 POKEZ+50,00:POKEZ+53,0:POKEZ+71,0:POKEZ+8,255
230 POKE56334,PEEK(56334) OR 1
240 GOTO260
250 GOTO440
260 PRINT "{SC}"
270 FORQ=1TO3:PRINT:NEXTQ
280 PRINT" {YL}UDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDI";"{WH}"
290 PRINT TAB(14);"ADL-6408":PRINT
300 PRINT TAB(13);"DATA LOGGER":PRINT
310 PRINT TAB(13);"DEVELOPED BY"
320 PRINT:PRINT TAB(08); "TECHNICAL HARDWARE INC."
330 PRINT:PRINT TAB(12);"P.O. BOX 9101"
340 PRINT:PRINT TAB(09);"PEMBROKE PINES, FLA."
350 PRINT:PRINT TAB(12);"COPYRIGHT 1984"
360 PRINT" {YL}JFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFK";"{WH}"
370 GOSUB5960 :PRINT"          HIT ANY KEY TO CONTINUE":GOSUB5970
380 GET E$:IF E$="" THEN GOTO380
390 FOR I=1TO08
400 AC(I)=255
410 AL(I)=-5
420 AH(I)=5
430 NEXT I
440 PRINT "{SC}"
450 PRINT "{YL} UDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDI";"{WH}"
460 PRINT"{YL} G";"{WH}";   TAB(02);"SELECT OPTION";TAB(38);"{YL}H";"{WH}"
470 PRINT"{YL} G";"{WH}"; TAB(12);"1. SET TIME";TAB(38);"{YL}H";"{WH}"
```

```
480 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
490 PRINT"{YL} G";"{WH}";TAB(12);"2. SET PARAMETERS";TAB(38);"{YL}H";"{WH}"
500 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
510 PRINT"{YL} G";"{WH}";TAB(12);"3. SET ALARM LIMITS";TAB(38);"{YL}H";"{WH}"
520 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
530 PRINT"{YL} G";"{WH}";TAB(12);"4. START ANALYSIS";TAB(38);"{YL}H";"{WH}"
540 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
550 PRINT"{YL} G";"{WH}";TAB(12);"5. SAVE ON TAPE/DISK";TAB(38);"{YL}H";"{WH}"
560 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
570 PRINT"{YL} G";"{WH}";TAB(12);"6. PRINT OLD DATA";TAB(38);"{YL}H";"{WH}"
580 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
590 PRINT"{YL} G";"{WH}";TAB(12);"7. LOAD OLD DATA";TAB(38);"{YL}H";"{WH}"
600 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
610 PRINT"{YL} G";"{WH}";TAB(12);"8. EXAMINE MEMORY";TAB(38);"{YL}H";"{WH}"
620 PRINT"{YL} G";"{WH}";TAB(38);"{YL}H";"{WH}"
630 PRINT"{YL} G";"{WH}";TAB(12);"9. TEST/STATUS/DIR";TAB(38);"{YL}H";"{WH}"
640 PRINT"{YL} JFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFK";"{WH}"
642 PRINT:PRINT"  ACQUISITION:"
645 A=PEEK(Z+72):IF A=0 THEN PRINT "  INACTIVE":GOTO 650
646 IF A=128 THEN PRINT "  ACTIVE":GOTO 650
650 GET E$
660 IF E$="" THEN GOSUB4680
670 A=VAL(E$)
680 IF A>9 THEN650
690 ON A GOTO760  ,1110,1690,1930,2780,3070,4190,4450,4570
700 GOTO650
710 X1=INT(X/256):X2=X-256*X1:RETURN
720 K=0:T=Z+16
730 P=PEEK(T):IF P>08 THEN RETURN
740 C(K+1)=P:K=K+1:T=T+1:GOTO730
750 POKE Z1+09,PEEK(Z+09) AND AC(T):RETURN
760 GOSUB4740
770 GOSUB5960
780 PRINT"         SET TIME (H,M,S)"
790 GOSUB5970
800 INPUT" AM OR PM ";T$:IF LEFT$(T$,1)<>"A" THEN S=128:GOTO820
810 S=0
820 INPUT"HOUR";A$
830 IF VAL(A$)>23 THEN820
840 IF VAL(A$)=0THEN A$="00":GOTO880
850 IF LEN(A$)<2 THEN870
860 GOTO880
870 IF VAL(A$)<10 THEN A$="0"+A$
880 INPUT"MINUTE";B$
890 IF VAL(B$)>59THEN880
900 IF VAL(B$)=0 THEN B$="00":GOTO940
910 IF LEN(B$)<2 THEN930
920 GOTO940
930 IF VAL(B$)<10 THEN B$="0"+B$
940 INPUT"SECOND";C$
950 IF VAL(C$)>59 THEN940
960 IF VAL(C$)=0 THEN C$="00":GOTO1000
970 IF LEN(C$)<2 THEN990
980 GOTO1000
990 IF VAL(C$)<10 THEN C$="0"+C$
1000 A=VAL(A$):I%=A/10:A=A-(I%*10):J=A+I%*16:POKE56331,J OR S
```

```
1010 B=VAL(B$):I%=B/10:B=B-(I%*10):J=B+I%*16:POKE56330,J
1020 C=VAL(C$):I%=C/10:C=C-(I%*10):J=C+I%*16:POKE56329,J
1030 POKE56328,0
1040 PRINT "DATE"
1050 INPUT "MONTH (1-12): ";F:F$=STR$(F)
1060 IF F)12 THEN1050
1070 POKEZ+4,F:INPUT "DAY:";G:G$=STR$(G)
1080 IF G)31 THEN1070
1090 POKEZ+3,G
1100 GOTO250
1110 PRINT "{SC}{CD}"
1120 I=1144
1130 K=1:T=0:Y=Z+16:GOSUB5610 :R=V+8
1140 PRINT"{SC}": GOSUB5960 :RV=R
1150 PRINT"  INPUT CHANNEL NUMBER (99=END)":I3=0:PRINT
1160 GOSUB5970
1170 DF=10
1180 PRINT:PRINT:PRINT
1190 PRINT"{CU}";:INPUT C(K)
1200 IF C(K))08 THEN PRINT"{SC}":POKEY,255:POKERV,255:GOTO1310
1210 POKEY,C(K):POKERV,C(K):Y=Y+1:RV=RV+1:IF C(K)>9 THEN X=4
1220 A=C(K):B$=STR$(A)
1230 IF X=4 THEN C$=MID$(B$,3,1):D(2)=ASC(C$)
1240 C$=MID$(B$,2,1):D(1)=ASC(C$)
1250 POKEI,D(1)
1260 I=I+1
1270 IF X=4 THEN POKEI,D(2)
1280 I=I+1:K=K+1:IF X=4 THEN I=I+1:X=0
1290 IF K)09 THEN1310
1300 GOTO1190
1310 R=R-8:TT=PEEK(Z+4):TD=PEEK(Z+3):POKE(R+1),TT:POKE(R+2),TD:POKER,0
1320 GOSUB5810  :POKE(R+3),TH:POKE(R+4),TM:POKE(R+5),TS
1330 K=K-1:GOSUB5960 :LX=0
1340 PRINT "     LIST MODE DATA FILE (Y/N)":GOSUB5970
1350 INPUTT$:T$=LEFT$(T$,1):IF T$()"Y" THEN1460
1360 PRINT"FILE TO DISK OR TAPE (D/T)"
1370 GET T$:IF T$="" THEN1370
1380 IF T$="Q"THEN250
1385 IF T$="T" THEN LT=10:GOTO 1410
1390 IF T$()"D" THEN GOTO1460
1400 LT=0
1410 INPUT"LIST MODE FILENAME: ";L$
1412 PRINT"DISK DRIVE NUMBER 8 OR 9?";:INPUT DR:DR$=STR$(DR)+":"
1415 IF LT=10 THEN C=1:OPEN C,1,1,L$+".LST":GOTO 1450
1420 IF LT=0 THEN L$="0:"+L$+".LST,S,W":GOTO1430
1430 OPEN 1,DR,15:C=2:OPEN C,DR,3,L$:INPUT#1,A,B$
1440 IF A()0 THEN PRINTB$:CLOSE 1:CLOSE 2:LX=0:GOTO1410
1450 LX=80:REM HEADER FLAG
1460 GOSUB5960
1470 PRINT"      ASSIGN CHANNEL NAMES(Y/N)"
1480 GOSUB5970
1490 INPUTA$
1500 IF A$()"Y" THEN GOTO1550
1510 PRINT"CHANNEL NAME (5 CHAR MAX.)   "
1520 FOR T=1TOK
1530 PRINT "(";C(T);")";:INPUT L1$(T)
```

```
1540 NEXTT:IF A$="Y" THEN GOTO1560
1550 FORT=1TOK:L1$(T)="*---*":NEXT T
1560 GOSUB5960
1570 PRINT"   SET INPUT PORT TRIGGER VALUE (Y/N)"
1580 GOSUB5970
1590 INPUT A$
1600 IF A$()"Y" THENPOKEZ1+50,0:GOTO1630
1610 INPUT " TRIGGER VALUE TO START AQUIRE= ";XT
1620 POKEZ1+50,128:POKEZ1+54,XT
1630 PRINT"{SC}":GOSUB5960 :PRINT"      INPUT ANALYSIS TIME (H,M,S)":GOSUB5970
1640 INPUTH,M,S:POKEZ+53,0:POKEZ+72,0
1650 AT=3600*H+(60*M)+S:X=INT(AT)
1660 GOSUB710 :POKEZ+5,X2:POKEZ+6,X1:POKEZ+7,X2:POKEZ+8,X1:POKER+6,X2:POKER+7,X1
1670 IF LX=80 OR LX=40 THEN GOSUB6560:REM OPEN UP FILE AND DEVELOP HEADER
1680 GOTO250
1690 GOSUB4740
1700 GOSUB5960
1710 PRINT "   SET ALARM LIMITS":GOSUB5970 :IF I3>7 THEN1910
1720 PRINT"{CD}":PRINT "CHANNEL 99=END":T=0:INPUT"CHANNEL NUMBER";T2
1730 IF T2>08 THEN1920
1740 K=0:T1=Z1+16
1750 P=PEEK(T1):IFP>08 THEN1770
1760 CH(K+1)=P:K=K+1:T1=T1+1:GOTO1750
1770 FORT=1TOK
1780 IF CH(T)=T2 THEN1810
1790 NEXT T
1800 GOTO1720
1810 PRINT "(";L1$(T);")"
1820 IF T2>08 THEN1720
1860 INPUT  "HIGH LIMIT ";AH(T):I3=I3+1
1870 INPUT "LOW LIMIT";AL(T)
1880 INPUT "ALARM LINE (0,1,2,3,4,5,6 OR 7)";I
1890 AA(I+1)=T2:AC(T)=255-(2^I):PRINT AC$(I+1):IF I3>7 THEN1910
1900 GOTO1720
1910 PRINT "{YL}ALL ALARMS ASSIGNED{WH}":FORQ=1TO1000:NEXTQ
1920 GOTO250
1930 GOSUB4740
1940 GOSUB5960
1950 PRINT "          START AQUISITION "
1960 PRINT "      START OR CONTINUE (S/C)"
1970 GOSUB5970
1980 IF DF()0 THEN GOTO2020
1990 PRINT "*ERROR* NO AQUISITION PARAMETERS SET"
2000 PRINT"USE OPTION 2"
2010 FORQ=1TO1000:NEXTQ:GOTO250
2020 GET X$: IF X$="" THEN2020
2030 IF X$="C" THEN CA=1:GOTO 2150
2040 IF X$()"S" THEN250
2050 PRINT"{SC}":Z=Z1:TT=PEEK(Z+4):TD=PEEK(Z+3):GOSUB5610
2060 X=24602:GOSUB710 :POKEZ+67,X2:POKEZ+68,X1:POKE251,X2:POKE252,X1
2070 GOSUB5810 :GOSUB5610 :POKEV,0:POKEV+3,TH:POKEV+4,TM:POKEV+5,TS
2080 TT=((PEEK(Z+63)*10)+PEEK(Z+64)):TD=((PEEK(Z+61)*10)+PEEK(Z+62))
2090 POKEV+1,TT:POKEV+2,TD:POKEZ+72,0
2100 GOSUB720
2110 MP=24602:GOTO2160
2150 PRINT "{SC}"
```

```
2160 PRINT "{YL}CHANNEL";TAB(12);" RAW ";TAB(20);"SCALED";TAB(28);"ALARM(S){WH}"
2170 IF LX=50 OR LX=5  THEN PRINT"{CU}";TAB(11);"*":GOTO2190
2180 IF LX=100 OR LX=15 THEN PRINT"{CU}";TAB(19);"*":GOTO2190
2190 Z=Z1
2200 GOSUB720
2210 FOR T=1TOK
2220 IF L1$(T)="*---*" OR L1$="*" THEN PRINT"*";"(";C(T);")";TAB(8);"=":GOTO2240
2230 PRINT L1$(T);TAB(08);"="
2240 NEXT T:POKEZ+72,128
2250 B=PEEK(Z+10):IF B()0 THEN2640
2260 POKEZ+10,128:PRINT "{HM}":PRINT:IF CA()1 THEN GOTO 2290
2270 PRINT"{HM}":PRINT
2280 MP=ABS((256*PEEK(Z+68)+PEEK(Z+67)-K)-3)
2290 TH=PEEK(MP):TM=PEEK(MP+1):TS=PEEK(MP+2):MP=MP+3
2295 IF LX=50 OR LX=100 THEN GOSUB 6920:GOTO 2297
2296 GOTO 2300
2297 IF LT=10 THEN C=1:GOTO 2299
2298 IF LT=0 THEN C=2
2299 PRINT#C,TH:PRINT#C,TM:PRINT#C,TS
2300 FORT=1TOK
2310 Q=((T*1)-1)
2320 HD=PEEK((MP)+Q)
2340 D(T)=(HD*2)/100
2350 A$(T)=" "
2380 IF D(T))AH(T) THEN A$(T)="*":GOSUB750
2390 IF D(T)(AL(T) THEN A$(T)="*":GOSUB750
2400 IF D(T)=0 THEN D$(T)=A$(T)+" 0.000":GOTO2420
2410 GOSUB5880
2420 IF PR=2 THEN RETURN
2430 IF C(T)=0 THEN GOSUB6080 :GOTO2480
2450 ON C(T) GOSUB6110 ,6140 ,6170 ,6200 ,6230 ,6260 ,6290,6320
2480 IF SC(T)=0 THEN SC$(T)=A$(T)+" 0.000 ":GOTO2500
2490 GOSUB6690
2500 PRINT TAB(10);D$(T);TAB(19);SC$(T);:IF LX=100 THEN PRINT#C,SC(T)
2510 IF LX=50 THEN PRINT#C,D$(T)
2520 IF A$(T)="*" THEN PRINT TAB(29);:GOTO2550
2530 PRINT TAB(29);"          "
2540 GOTO2600
2550 FOR I=1TO8
2560 IF AA(I)=C(T) THEN2580
2570 NEXT I
2580 PRINT AC$(I);"*";I-1;"*":GOTO2600
2590 PRINT
2600 NEXTT:POKEZ+10,128:IFPEEK(Z+53)()128 THEN GOTO 2620
2602 PRINT:PRINT"* ERROR * MEMORY OVERFLOW":X=24602:GOSUB710:POKEEDTA,X2
2604 POKEEDTA+1,X1
2620 IF PR=1 THEN MD=MP-3:EA=MD+T+T:GOSUB3990
2630 GOSUB5660
2640 GOSUB4680
2650 GET E$:IF E$="Q"THEN2720
2660 IF E$="R" THEN4830
2670 IF E$()"P" THEN GOTO2700
2680 IF PR=1 THEN PR=0:CLOSE 4:GOTO2700
2690 PR=1:GOSUB4820
2700 IF PEEK(Z+10)=0 THEN2270
2710 GOTO2640
```

```
2720 IF LX=50 OR LX=100 THEN CLOSE1:CLOSEC
2730 IF LX=5 OR 15 THEN CLOSE 1:CLOSE C
2740 CLOSE 4:PR=0:LX=0:LF=0:LT=0
2750 GOTO250
2760 PRINT#C,(((A(5)-48)*10)+A(6)-48):PRINT#C,(((A(3)-48)*10)+(A(4)-48))
2770 PRINT#C,(((A(1)-48)*10)+(A(2)-48)):RETURN
2780 GOSUB4740
2790 GOSUB5960
2800 PRINT "     SAVE DATA ON TAPE/DISK"
2810 GOSUB5970
2820 PRINT "{CD}":INPUT "     TAPE OR DISK (T/D/Q)";Z$
2830 IF Z$="Q" THEN250
2840 PRINT"{CD}": INPUT "   {RV}FILENAME:{RO}";T$
2850 GOSUB5610
2860 H=U-V
2870 IFH(=26 THEN PRINT"{RV}NO DATA AVAILABLE TO SAVE{RO}":FORI=1TO1000:NEXT I:G
OTO250
2880 Z$=LEFT$(Z$,1):POKEZ+72,0
2890 IF Z$="T" THEN3010
2900 IF Z$()"D" THEN2810
2905 PRINT"DISK DRIVE NUMBER 8 OR 9?";:INPUT DR:DR$=STR$(DR)+":"
2910 T$=DR$+T$+".DAT,S,W"
2920 PRINT"SAVING ";T$;" TO DISK"
2930 GOSUB720 :OPEN 1,DR,15:C=2:OPEN C,DR,2,T$
2940 INPUT#1,A,B$:IF A()0 THEN PRINT B$:CLOSE 1:CLOSE 2:GOTO2840
2950 GOSUB2960 :INPUT#1,A,B$:PRINT B$:CLOSE 1:CLOSE 2:GOTO250
2960 PRINT#C,K:PRINT#C,U1:PRINT#C,U2:PRINT#C,V1:PRINT#C,V2:PRINT#C,0
2970 FORI=V+1 TO V+26:PRINT#C,PEEK(I):NEXTI
2980 FORI=1TOK:PRINT#C,L1$(I):NEXTI
2990 PRINT#C,H:FORI=V+26 TO (V+26+H):PRINT#C,PEEK(I):NEXTI
3000 RETURN
3010 T$=T$+".DAT":PRINT"REWIND TAPE TO START"
3020 INPUT "TYPE 'GO' WHEN READY";A$:IF A$()"GO" THEN3020
3040 PRINT "OPENING FILE ";T$;" ON TAPE":FORI=1TO1000:NEXTI:C=1:OPEN C,1,1,T$
3050 GOSUB2960 :CLOSE C:GOTO250
3060 GOTO250
3070 GOSUB4740
3080 GOSUB5960
3090 PRINT "     PRINT OLD DATA"
3100 GOSUB5970
3110 PRINT"LIST FILE OR MEMORY DATA (L/M)"
3120 GETX$:IF X$="" THEN3120
3130 IF X$="M" THEN PRINT "DATA FROM MEMORY":PR=2:GOTO3400
3140 IF X$="Q" THEN GOTO250
3150 IF X$="L" THEN PRINT "DATA FROM LIST FILE"
3220 INPUT"FILENAME: ";L$
3221 PRINT"DATA FROM TAPE OR DISK (T/D/Q)"
3222 GET X$:IF X$="" THEN 3222
3223 IF X$="T" THEN GOSUB 12000:OPEN C,1,0,L$+".LST":GOSUB3250:CLOSE C:GOTO 250
3224 IF X$="Q" THEN GOTO 250
3225 PRINT"DISK DRIVE NUMBER 8 OR 9?";:INPUT DR:DR$=STR$(DR)+":"
3229 L$=DR$+L$+".LST,S,R":PRINT"DATA FROM DISK"
3230 OPEN 1,DR,15:C=2:OPEN C,DR,2,L$:INPUT#1,A,B$:IF A=0 THEN GOTO3250
3240 PRINT B$:CLOSE 1:CLOSE 2:GOTO3220
3250 PR=2:OPEN 4,4,1:PRINT#4:PRINT#4,"DATA HEADER"
3260 INPUT#C,K,U1,U2,V1,V2,FG:GOSUB5630
3280 INPUT#C,MO,TD,TH,TM,TS,A1,A2
```

```
3290 FORI=1TO16:INPUT#C,D(I):NEXT:FORI=1TO3:INPUT#C,B$:NEXTI
3300 FORI=1TOK:INPUT#C,L2$(I):NEXT
3310 PRINT#4,"CHANNELS: ";:FORI=1TOK:PRINT#4,D(I);:NEXT:PRINT#4," "
3320 PRINT#4,"DATE:";MO;"/";TD:PRINT#4, "TIME OF AQUIRE: ";TH;"/";TM;"/";TS
3330 PRINT#4,"AQUISITION INTERVAL: ";((A2*256)+A1);" SECONDS"
3340 PRINT#4,"TYPE OF DATA FILE: ";:IF LF=10THENPRINT#4,"SCALED DATA":GOTO3360
3350 PRINT#4,"RAW DATA FILE"
3360 PRINT#4," ":PRINT#4," "
3370 GOSUB3490 :GOSUB3520 :GOSUB3640:GOSUB3670 :GOSUB3770
3380 PRINT#4,MO;"/";TD:GOSUB3860:GOSUB3770:CLOSE4:CLOSE1:CLOSE2:PR=0:GOTO250
3390 REM PRINT MEM DATA
3400 OPEN 4,4,1
3420 GOSUB5610  :MD=V+26:GOSUB720 :GOSUB3490:GOSUB3580:GOSUB3640
3430 GOSUB3720:GOSUB3770:GOSUB3820:IF PR=1 THEN RETURN
3440 GOSUB3980 :IF PR=1 THEN RETURN
3450 GOTO250
3460 GOSUB5610
3470 MD=V+26
3480 GOSUB720
3490 PRINT#4,"CHANNEL          ";
3500 IF Z$="Q" THEN250
3510 RETURN
3520 FORT=1TOK
3530 PRINT#4,D(T);
3540 IF D(T)>9 THEN PRINT#4,SPC(3);:GOTO3550
3550 PRINT#4, SPC(4);
3560 NEXTT
3570 RETURN
3580 FORT=1TOK
3590 PRINT#4,C(T);
3600 IF C(T)>9 THEN PRINT#4,SPC(3);:GOTO3620
3610 PRINT#4, SPC(4);
3620 NEXTT
3630 RETURN
3640 PRINT#4," ":PRINT#4," "
3650 PRINT#4,"NAMES:          ";
3660 RETURN
3670 FORT=1TOK
3680 PRINT#4,L2$(T);SPC(2);
3690 NEXTT
3700 PRINT#4
3710 RETURN
3720 FORT=1TOK
3730 PRINT#4,L1$(T);SPC(2);
3740 NEXTT
3750 PRINT#4
3760 RETURN
3770 FORT=1TO((7*K)+13)
3780 PRINT#4,"-";
3790 NEXTT
3800 PRINT#4,"-"
3810 RETURN
3820 TT=PEEK(Z+4):TD=PEEK(Z+3)
3830 PRINT#4,TT;":";TD
3840 RETURN
3860 INPUT#C,TH,TM,TS:PRINT#4,TH;":";TM;":";TS;:IF TS<10 THEN PRINT#4,SPC(1);
```

```
3870 D$="":A$=" ":FORT=1TOK:A$(T)=" ":D$=""
3880 GET#C,C$
3890 IFC$()CHR$(13) THEN D$=D$+C$:GOTO3880
3900 D$(T)=D$:IF ST=64 THEN CLOSE1,2:PRINT#4:GOTO3930
3910 PRINT#4,D$(T);:NEXTT:PRINT#4," "
3920 GOTO3860
3930 PRINT#4,"END OF DATA FILE":RETURN
3940 MD=MD+T:IF MD=)EA THENCLOSE1:CLOSE2:GOSUB3770 :PRINT#4:CLOSE 4:GOTO250
3950 RETURN
3960 IF PR=1 THEN RETURN
3970 GOSUB5610
3980 GOSUB5610 :MD=V+26:EA=U+2
3990 TH=PEEK(MD):TM=PEEK(MD+1):TS=PEEK(MD+2)
4000 IF MD=)EA THEN GOSUB4160:PRINT#4:CLOSE 4:GOTO250
4010 GOSUB6920
4015 IF PR=2 THEN MD=MD+3
4020 PRINT#4,TH;":";TM;":";TS;:IF TS(10 THEN PRINT#4," ";
4040 IF MD=)EA THEN GOSUB4160:PRINT#4:CLOSE 4:GOTO250
4050 FORT=1TOK
4060 Q=((1*T)-1)
4070 HD=PEEK(MD+Q)
4080 IF PR=1 THEN PRINT#4,D$(T);:GOTO4110
4090 IF PR=2 THEN GOSUB2340 :PRINT#4,D$(T);:GOTO4110
4100 GOSUB5880  :PRINT#4,D$(T);
4110 NEXTT
4120 PRINT#4," "
4130 IF PR=1 THEN RETURN
4140 MD=MD+K:IF MD=)EA THEN GOSUB4160 :PRINT#4:CLOSE 4:PR=0:GOTO250
4150 GOTO3990
4160 PRINT#4:FORT=1TO((7*K)+13):PRINT#4,"-";:NEXTT:PRINT#4," " :PR=0
4170 IF LF()0 THEN CLOSE 1:CLOSE 2:RETURN
4180 RETURN
4190 GOSUB4740
4200 GOSUB5960 :PRINT"   LOAD DATA ":GOSUB5970
4210 INPUT "LOAD DATA FROM TAPE/DISK (T/D/Q)";Z$
4220 Z$=LEFT$(Z$,1):IF Z$="Q" THEN250
4230 INPUT "{RV}FILENAME:{RO}";T$
4240 IF Z$="T" THEN4400
4250 IF Z$()"D" THEN250
4255 PRINT"DISK DRIVE NUMBER 8 OR 9?";:INPUT DR:DR$=STR$(DR)+":"
4260 T$=DR$+T$+".DAT,S,R
4270 PRINT"LOADING ";T$;" FROM DISK"
4280 OPEN 1,DR,15:C=2:OPEN C,DR,2,T$
4290 INPUT#1,A,B$:IF A()0 THEN PRINT B$:CLOSE 1:CLOSE C:GOTO4210
4300 GOSUB4320:POKEZ+72,0:INPUT#1,A,B$:PRINT B$:CLOSE 1:CLOSE C
4310 GOTO250
4320 INPUT#C,K:INPUT#C,U1,U2,V1,V2,FG
4340 GOSUB5980 :GOSUB5630
4350 FORI=V+1 TO V+26:INPUT#C,X:POKEI,X:NEXT I
4360 FORI=1TOK:INPUT#C,L1$(I):NEXTI
4370 INPUT#C,H:FORI=V+26 TO (V+26+H):INPUT#C,X:POKEI,X:NEXTI
4380 GOSUB5990:GOSUB 5980
4390 RETURN
4400 T$=T$+".DAT":PRINT"REWIND TAPE TO START"
4410 INPUT "TYPE 'GO' WHEN READY";A$:IF A$()"GO" THEN4410
4420 IF A$="Q" THEN250
```

```
4430 PRINT "OPENING FILE ";T$;" ON TAPE":FORI=1TO1000:NEXTI:C=1:OPEN C,1,0,T$
4440 POKEZ+72,0:GOSUB4320:CLOSE C:GOTO250
4450 GOSUB4740
4460 GOSUB5960
4470 PRINT "       EXAMINE MEMORY":GOSUB5970
4480 PRINT "INPUT MEMORY LOC TO DISPLAY:":INPUT Z
4490 PRINT"{SC}"
4500 FORI=0TO44 STEP 2
4510 A=PEEK(Z+I):B=PEEK(Z+1+I)
4520 PRINT Z+I,A,B
4530 NEXTI:Z=Z+I
4540 GET E$:IF E$="" THEN4540
4550 IF E$="M" THEN4490
4560 Z=Z1:GOTO250
4570 PRINT"{SC}":GOSUB5960:PRINT" SYSTEM STATUS, TEST OR DIR. (S/T/D)"
4572 GOSUB 5970:X$=" "
4580 INPUTE$
4630 IF E$="S" THEN5310
4635 IF E$="D" THEN 8000
4640 IF E$()"T" THEN GOTO250
4650 GOTO5000
4660 PRINT "TEST"
4670 GOTO250
4680 SYS 40561: SYS 40658
4690 I=1929:B(1)=PEEK(Z+63)+48:B(2)=PEEK(Z+64)+48:B(3)=PEEK(Z+61)+48
4700 B(4)=PEEK(Z+62)+48:POKEI,B(1):POKEI+1,B(2):POKEI+3,47:POKEI+4,B(3)
4710 POKEI+5,B(4)
4720 GOSUB4840
4730 RETURN
4740 PRINT "{SC}{CD}{CD}"
4750 RETURN
4760 FOR I=0 TO 15
4770 PRINT C(I)
4780 NEXT I
4790 GOTO390
4810 IF PR=1 THEN PR=0:PRINT#4:CLOSE 4:RETURN
4820 PR=1:GOSUB3400 :RETURN
4830 POKE(Z+9),255:GOTO2710
4840 FORI=1 TO 6
4850 A(I)=PEEK(I+Z+54)+48
4860 NEXTI
4870 I=1968+10:P=PEEK(Z+71):IF P=128 THEN POKEI,16:POKEI+1,13:GOTO4890
4880 POKEI,1:POKEI+1,13
4890 I=1968
4900 POKEI,A(5)
4910 POKEI+1,A(6)
4920 POKEI+2,47
4930 POKEI+3,A(3)
4940 POKEI+4,A(4)
4950 POKEI+5,47
4960 POKEI+6,A(1)
4970 POKEI+7,A(2)
4980 RETURN
5000 PRINT"{SC}":GOSUB5960 :PRINT"       TEST ROUTINES":GOSUB5970
5010 GOSUB5960 :PRINT"     CYCLE ALARMS (Y/N)":GOSUB5970 :INPUTA$
5020 IF A$()"Y" THEN5080
```

```
5025 ZA=PEEK(40341):ZB=PEEK(40342):ZT=((ZB*256)+ZA)
5030 PRINT "ALARM: "
5040 FORT=0TO7:A%=255:B%=2^T:A%=255-B%
5050 PRINTT;:POKE ZT,A%:PRINT "ON ";:FORQ=1TO500:NEXTQ:PRINT "OFF "
5070 NEXTT:POKE ZT,255
5080 GOSUB5960 :PRINT"      DEFINE ALARM NAMES (Y/N) ":GOSUB5970 :INPUTE$
5090 IF E$()"Y" THEN5150
5100 PRINT"{SC}"
5110 PRINT"ALARM NUMBER ";TAB(15);"NAME (8 CHAR MAX) "
5120 FOR T=0TO7
5130 PRINT T;:INPUT"";AC$(T+1)
5140 NEXTT
5150 GOSUB5960 :PRINT"     READ INPUT PORT (Y/N) ":GOSUB5970 :INPUTE$
5160 IF E$()"Y" THEN GOTO5280
5170 PRINT"{SC}"
5180 PRINT"INPUT PORT STATUS:"
5185 ZA=PEEK(40552):ZB=PEEK(40553):ZT=((ZB*256)+ZA)
5190 FORT=0TO7
5200 PRINTT;TAB(10);
5210 IP=PEEK(ZT)
5220 I1=IP AND (2^T)
5230 IF I1)0 THEN PRINT"ON ": GOTO5250
5240 PRINT "OFF "
5250 NEXTT
5260 PRINT "HIT ANY KEY TO RETURN";
5270 GET E$:IFE$=""THEN5270
5280 IF X$="P" THEN PRINT#1:CLOSE 1
5290 GOTO250
5310 PRINT"{SC}";
5320 PRINT"              {YL}{RV}SYSTEM STATUS{RO}";"{WH}"
5330 GOSUB5610 :PRINT "DATA MEMORY: ";U-V; " BYTES USED "
5340 GOSUB5730  :REM GET PARAMETERS
5350 PRINT"ACQUISITION TIME PERIOD: (SEC'S)   "
5360 PRINT"CURRENT:";CT;SPC(10);"PREVIOUS: ";PT
5370 PRINT "CHANNELS: ";:GOSUB720 :FORT=1TOK:PRINT C(T);" ";:NEXTT
5380 PRINT
5390 PRINT "ALARMS: ";TAB(10);"HIGH SET";TAB(20);"LOW SET"
5400 REM GOSUB 10000
5410 FORT=1TO8:PRINT AA(T);"(";AC$(T);")";TAB(10);
5420 IF AA(T)=0 THEN PRINT 0;TAB(20);0:GOTO5470
5430 FOR I=1TOK
5440 IF C(I)=AA(T) THEN5460
5450 NEXT I
5460 PRINT AH(I);TAB(20);AL(I)
5470 NEXTT
5480 PRINT "ALARM:";TAB(08);"BINARY:";TAB(19);"STATUS:"
5490 FORT=0TO7
5500 AL=PEEK(Z+9):AB=2^T:AL=AL AND AB
5510 PRINT T+1;TAB(10);AL;TAB(20);:IF AL)0 THEN PRINT "OFF ":GOTO5530
5520 PRINT"{YL}{RV}ON {RO}{WH}"
5530 NEXT T
5540 PRINT "HIT ANY KEY TO RETURN";
5550 GET E$:IF E$="" THEN5550
5560 IF X$="P" THEN PRINT#1:CLOSE 1
5570 GOTO250
5580 A$=LEFT$(TI$,2):A(1)=VAL(A$):A$=MID$(TI$,2,2):A(2)=VAL(A$)
```

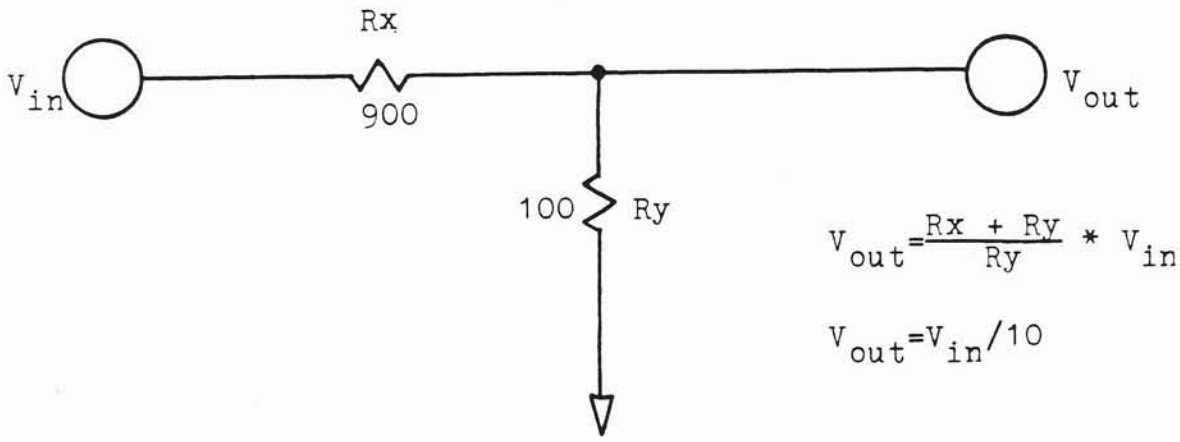
```
5590 A$=RIGHT$(TI$,2):A(3)=VAL(A$)
5600 RETURN
5610 V1=PEEK(BDTA):V2=PEEK(BDTA+1)
5620 U1=PEEK(EDTA):U2=PEEK(EDTA+1)
5630 V=((V2*256)+V1):U=((U2*256)+U1)
5640 RETURN
5650 L=((V-U)/K):RETURN
5660 A$=STR$(MP):B=LEN(A$)
5670 FORQ=1TOB
5680 B$=MID$(A$,Q,1):B(Q)=ASC(B$)
5690 NEXT Q:I=1888
5700 POKEI,13:POKEI+1,05:POKEI+2,13:POKEI+4,58
5710 FORI=1TOB
5720 POKE(I+1893),B(I):NEXTI:RETURN
5730 Z=Z1
5740 A=PEEK(Z+6):B=PEEK(Z+5)
5750 PT=CT
5760 CT=(A*256)+B
5770 RETURN
5780 FORJ=1TO8:AC$(J)="(***)":NEXT J
5790 IF W1=80 THEN RETURN
5800 GOTO5150
5810 TH=(PEEK(Z+59)*10)+PEEK(Z+60):TS=(PEEK(Z+55)*10)+PEEK(Z+56)
5820 TM=(PEEK(Z+57)*10)+PEEK(Z+58)
5830 RETURN
5840 FORT=1TO08
5850 PRINT AC(T),C(T),AL(T),AH(T)
5860 NEXT
5870 STOP
5880 IF ABS(D(T))<9.999 THEN M$=STR$(D(T)):S$="0":GOTO5900
5890 D$(T)=A$(T)+STR$(D(T)):RETURN
5900 IF D(T)=0 THEN D$(T)=A$(T)+" 0.000":RETURN
5905 IF LEN(M$)<3 THEN M$=M$+"."
5910 IF MID$(M$,2,1)<>"." THEN5950
5920 Q=LEN(M$)-1
5930 W$=LEFT$(M$,1)+S$+RIGHT$(M$,Q)
5940 M$=W$
5950 D$(T)=LEFT$(A$(T)+M$+"000",7):RETURN
5960 PRINT"{YL}UDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDDI";"{WH}":RETURN
5970 PRINT"{YL}JFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFK";"{WH}":RETURN
5980 POKEBDTA,V1:POKEBDTA+1,V2:POKEEDTA,U1:POKEEDTA+1,U2:RETURN
5990 K=0:GOSUB5630
6000 T=V+8:T1=Z+16
6010 P=PEEK(T):IF P>08THEN6050
6020 C(K+1)=P:K=K+1:T=T+1
6030 POKET1,P:T1=T1+1
6040 GOTO6010
6050 POKEZ+72,0:RETURN
6060 REM THIS IS THE SCALER ROUTINES
6070 REM USER ENTERS SCALER FUNCTIONS
6080 REM CH#0
6090 SC(T)=D(T)*1
6100 RETURN
6110 REM CH#1
6120 SC(T)=D(T)*1
6130 RETURN
```

```
6140 REM CH#2
6150 SC(T)=D(T)*1
6160 RETURN
6170 REM CH#3
6180 SC(T)=D(T)*1
6190 RETURN
6200 REM CH#4
6210 SC(T)=D(T)*1
6220 RETURN
6230 REM CH#5
6240 SC(T)=D(T)*1
6250 RETURN
6260 REM CH#6
6270 SC(T)=D(T)*1
6280 RETURN
6290 REM CH#7
6300 SC(T)=D(T)*1
6310 RETURN
6320 REM CH#8
6330 SC(T)=D(T)*1
6340 RETURN
6560 INPUT"LIST FILE TO CONTAIN SCALED OR RAW DATA (S/R) ";T$:T$=LEFT$(T$,1)
6570 IF T$="S" THEN X=10:LX=100:GOTO6600
6580 IF T$<>"R" THEN GOTO6560
6590 X=20:LX=50:REM FLAG FOR RAW DATA
6600 GOSUB5610 :PRINT#C,K:PRINT#C,U1:PRINT#C,U2:PRINT#C,V1:PRINT#C,V2
6610 PRINT#C,X
6620 FORI=V+1 TO V+26:PRINT#C,PEEK(I):NEXTI
6630 FORI=1TOK:PRINT#C,L1$(I):NEXTI
6640 IF LT=10 THEN GOTO6680
6650 INPUT#1,A,B$:IFA=0 THEN GOTO6680
6660 PRINT B$:CLOSE1:CLOSEC:LX=0:RETURN
6680 RETURN
6690 IF ABS(SC(T))<9.999 THEN M$=STR$(SC(T)):S$="0":GOTO6710
6700 SC$(T)=A$(T)+STR$(SC(T)):RETURN
6710 IF SC(T)=0 THEN SC$(T)=A$(T)+" 0.000 ":RETURN
6715 IF LEN(M$)<3 THEN M$=M$+"."
6720 IF MID$(M$,2,1)<>"." THEN6760
6730 W=LEN(M$)-1
6740 W$=LEFT$(M$,1)+S$+RIGHT$(M$,W)
6750 M$=W$
6760 SC$(T)=LEFT$(A$(T)+M$+"000",7):RETURN
6920 R%=TH/16:A=TH-(R%*16):TH=A+(R%*10)
6930 R%=TM/16:A=TM-(R%*16):TM=A+(R%*10)
6940 R%=TS/16:A=TS-(R%*16):TS=A+(R%*10)
6950 RETURN
8000 OPEN 2,8,15:PRINT"{SC}":GOTO8500
8010 OPEN 1,8,0,"$0"
8020 GET#1,A$,B$
8030 GET#1,A$,B$
8040 GET#1,A$,B$
8050 C=0
8060 IF A$<>"" THEN C=ASC(A$)
8070 IF B$<>"" THEN C=C+ASC(B$)*256
8080 PRINT MID$(STR$(C),2);TAB(3);
8090 GET#1,B$:IF ST<>0 THEN 8200
```

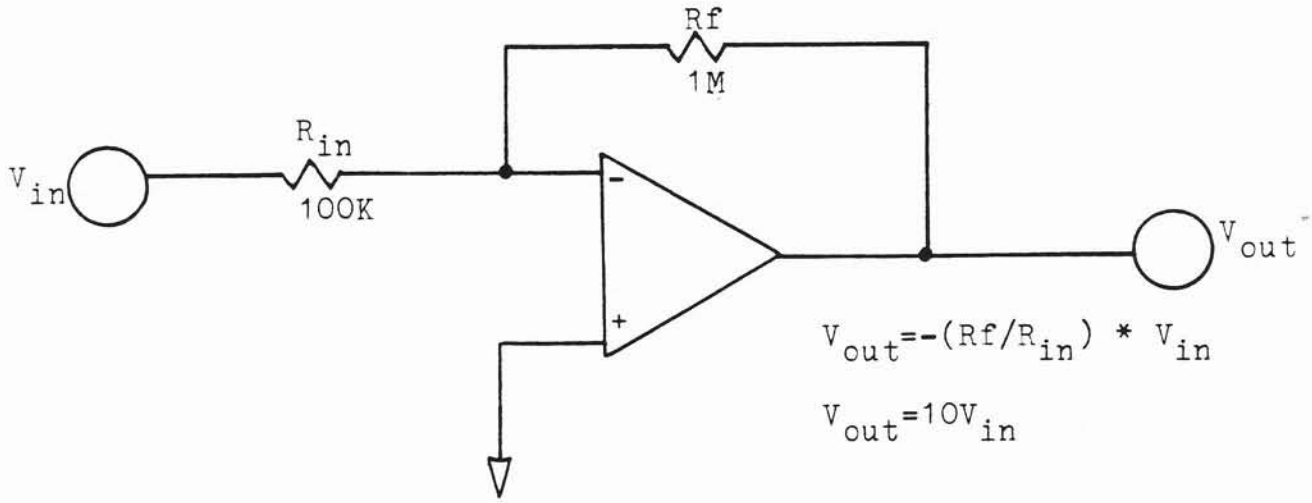
```
8100 IF B$<>CHR$(34) THEN 8090
8110 GET#1,B$:IF B$<>CHR$(34) THEN PRINTB$;:GOTO 8110
8120 GET#1,B$:IF B$=CHR$(32) THEN 8120
8130 PRINT TAB(20);:C$=""
8140 C$=C$+B$:GET#1,B$:IF B$<>"" THEN 8140
8150 PRINT LEFT$(C$,3)
8160 GET T$:IF T$<>"" THEN GOSUB8300
8170 IF ST=0 THEN 8030
8200 PRINT" BLOCKS FREE":PRINT
8205 PRINT "HIT ANY KEY TO EXIT"
8207 GET T$:IF T$="" THEN 8207
8208 PRINT"{SC}"
8210 CLOSE1:GOTO 9000
8300 IF T$="Q" THEN CLOSE 1:GOTO250
8310 GET T$:IF T$="" THEN 8310
8320 RETURN
8400 REM DISK COMMANDS
8410 C$="":PRINT")";
8420 GETB$:IFB$="" THEN 8420
8430 PRINTB$;:IFB$=CHR$(13) THEN 8450
8440 C$=C$+B$:GOTO 8420
8450 PRINT#2,C$
8500 PRINT"{RV}";
8510 GET#2,A$:PRINTA$;:IF A$<>CHR$(13)GOTO8510
8520 PRINT"{RO}"
9000 PRINT"{SC}":GOSUB 5960:PRINT:PRINT TAB(13);"D-DIRECTORY":PRINT
9010 PRINT TAB(13);")-DISK COMMAND"
9015 PRINT
9020 PRINT TAB(13);"Q-QUIT PROGRAM"
9025 PRINT
9030 PRINT TAB(13);"S-DISK STATUS"
9033 PRINT:PRINT
9035 GOSUB 5970
9040 GETA$:IFA$=""THEN9040
9050 IF A$="D" THEN PRINT"{SC}":GOTO 8010
9060 IF A$="." OR A$=")" THEN 8400
9070 IF A$="Q" THEN CLOSE 1:CLOSE 2:GOTO 250
9080 IF A$="S" THEN 8500
9090 GOTO 9040
10000 OPEN2,1,0,"TESTRUN"
10007 GET#2,A$:PRINT A$
10010 IF(ST AND 64)=64 THEN PRINT "END OF FILE":CLOSE 2
10020 GOTO 10007
11000 OPEN2,1,1,"TESTRUN"
11010 FOR I=1TO 1000:PRINT#2,I:PRINT I;
11020 NEXT I:CLOSE 2
12000 PRINT"DATA FROM TAPE CASSETTE":C=1:RETURN

READY.
```

```
LINE# LOC   CODE        LINE

00001  0000             ;
00002  0000             ;
00003  0000             ;
00004  0000             ; LAST EDIT DATE 2/10/85
00005  0000             ;
00006  0000             ;
00007  0000             ;
00008  0000             *       =       $9D00   ;BEGINNING PROG ADDR
00009  9D00             Z       =       $9F80   ;TEMP VAR STORAGE
00010  9D00             JIFF    =       Z+1     ;JIFFY CTR
00011  9D00             JIFFT   =       Z+2
00012  9D00             DAY     =       Z+3     ;CURRENT DAY
00013  9D00             MON     =       Z+4     ;CURRENT MONTH
00014  9D00             DAINT   =       Z+5     ;DATA INTERVAL
00015  9D00             DACO    =       Z+7     ;TEMP DAINT CTR
00016  9D00             ALOPT   =       Z+9     ;ALARM REG
00017  9D00             DAFLG   =       Z+10    ;DATA AVAIL FLG
00018  9D00             BUSY    =       Z+11    ;BUSY FLG
00019  9D00             XTEMP   =       Z+12    ;TEMP STORE X
00020  9D00             YTEMP   =       Z+13    ;TEMP STORE Y
00021  9D00             TEMP    =       Z+14
00022  9D00             MUX     =       Z+15    ;CURRENT CHAN
00023  9D00             CHAN    =       Z+16    ;CHAN # TO AQUIRE HERE
00024  9D00             MONL    =       Z+32    ;MONTH LENGTHS (12)
00025  9D00             VECTR   =       Z+44    ;JMP TO IRQ ROUTINE
00026  9D00             SECT    =       Z+47     ;SECOND STORAGE
00027  9D00             MIN     =       Z+48     ;MIN
00028  9D00             HOUR    =       Z+49     ;HOUR
00029  9D00             TRIG    =       Z+50     ;I/O PORT TRIGGER FLAG
00030  9D00             XTEM1   =       Z+51
00031  9D00             XTEM0   =       Z+52
00032  9D00             OVFL    =       Z+53
00033  9D00             TVAL    =       Z+54    ;TRIGGER MASK VALUE
00034  9D00             SEC1    =       Z+55    ;HIGH NIB SEC
00035  9D00             SEC2    =        Z+56    ;LOW  NIB SEC
00036  9D00             MIN1    =        Z+57    ;HIGH NIB MIN
00037  9D00             MIN2    =        Z+58    ;LOW NIB MIN
00038  9D00             HOUR1   =       Z+59    ;HIGH NIB HOUR
00039  9D00             HOUR2   =       Z+60    ;LOW NIB  HOUR
00040  9D00             DAY1    =       Z+61    ;HIGH ORDER
00041  9D00             DAY2    =       Z+62    ;LOW ORDER
00042  9D00             MON1    =       Z+63    ;HIGH MONTH
00043  9D00             MON2    =       Z+64    ;LOW MONTH
00044  9D00             BDTA    =       Z+65    ;BEGIN OF DATA STORAGE ADDR
00045  9D00             EDTA    =       Z+67    ;CURRENT END OF DATA STORAGE
00046  9D00             DWRP    =       Z+69    ;DATA WRAP AROUND LOCATION
00047  9D00             DAV     =       Z+71    ;DATA AVAIL FLG
00048  9D00             RFLG    =       Z+72    ;RUN FLAG
00049  9D00             LFLG    =       Z+73    ;DAY/MON UPDATE FLG
00050  9D00             ;
00051  9D00             ;
00052  9D00             ; I/O REGISTERS/ ADDRESSES
00053  9D00             ;
00054  9D00             ;
00055  9D00             STATU   =       $DFFE   ;STATUS REG ADDR
```

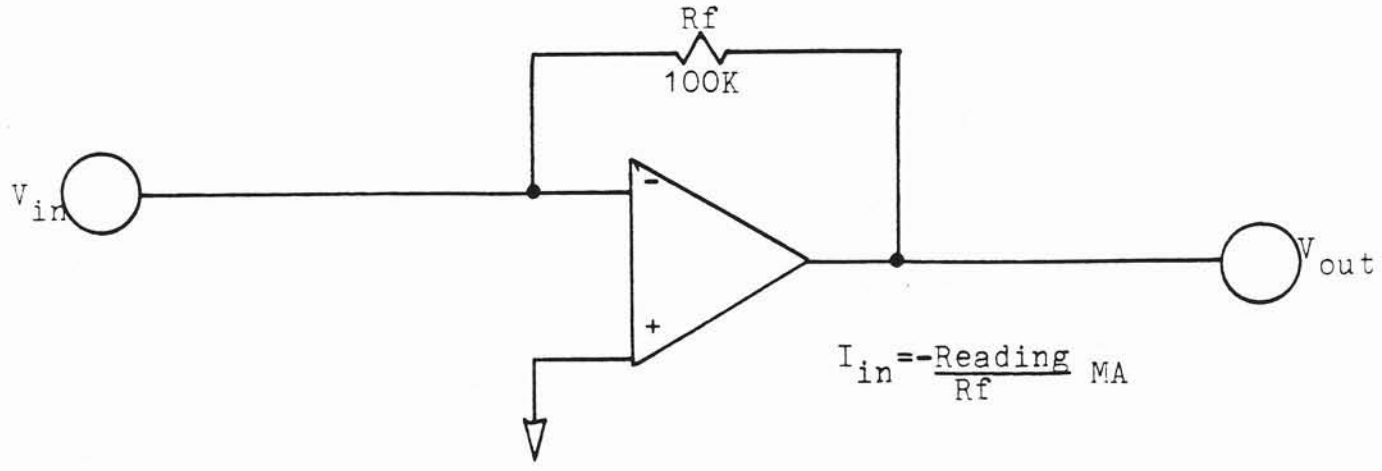
```
LINE# LOC   CODE          LINE

00056  9D00               READ   =          STATU-1 ;READ ADC DATA
00057  9D00               INRD   =          READ-1 ;READ INPUT PORT
00058  9D00               CNVT   =          READ   ;CONVERT REG ADDR
00059  9D00               OUTWR  =          INRD   ;OUTPUT PORT
00060  9D00               LMUX   =           STATU  ;LOAD MUX ADDR
00061  9D00               ;
00062  9D00               ;
00063  9D00               ;
00064  9D00               ; REAL TIME CLOCK REG'S
00065  9D00               ;
00066  9D00               ;
00067  9D00               ;
00068  9D00               TSEC1  =          $DC08  ;.1 SEC LOC
00069  9D00               TSEC   =          TSEC1+1 ;SEC REG
00070  9D00               TMIN   =          TSEC+1  ;MIN REG
00071  9D00               THOUR  =          TMIN+1  ;HOUR REG
00072  9D00               PGZO   =          $00FB   ;PAGE ZERO ADDRESS FOR INDIREC
T
00073  9D00               ;
00074  9D00               ;
00075  9D00               ;
00076  9D00               ;
00077  9D00               ;
00078  9D00               ;
00079  9D00               ;
00080  9D00               ;
00081  9D00  CE 81 9F     INT    DEC        JIFF   ;JIFFY CTR-1
00082  9D03  D0 27               BNE        INTZ
00083  9D05  AD 82 9F            LDA        JIFFT  ;WAS 0
00084  9D08  8D 81 9F            STA        JIFF   ;REPLACE
00085  9D0B  CE 87 9F            DEC        DACO   ;INTERVAL CNT-1
00086  9D0E  D0 1C               BNE        INTZ    ;NOT READY TO AQUIRE
00087  9D10  AD 87 9F            LDA        DACO   ;CHK FOR 0
00088  9D13  0D 88 9F            ORA        DACO+1
00089  9D16  F0 05               BEQ        INZ1
00090  9D18  CE 88 9F            DEC        DACO+1 ;DACO WAS =0
00091  9D1B  D0 0F               BNE        INTZ    ;NOT READY TO AQUIRE
00092  9D1D  AD 85 9F     INZ1   LDA        DAINT  ;TO REPLACE DACO
00093  9D20  8D 87 9F            STA        DACO
00094  9D23  AD 86 9F            LDA        DAINT+1
00095  9D26  8D 88 9F            STA        DACO+1
00096  9D29  4C 36 9D            JMP        INZ2
00097  9D2C  AD 8B 9F     INTZ   LDA        BUSY   ;GET FLAG
00098  9D2F  C9 80               CMP        #$80
00099  9D31  F0 2C               BEQ        INTX
00100  9D33  4C B1 9D            JMP        INTY
00101  9D36  AD C8 9F     INZ2   LDA        RFLG   ;GET RUN FLAG
00102  9D39  C9 80               CMP        #$80
00103  9D3B  D0 74               BNE        INTY
00104  9D3D  AD B2 9F            LDA        TRIG   ;GET INPUT MASK
00105  9D40  C9 80               CMP        #$80    ;SEE IF FLG SET
00106  9D42  D0 05               BNE        INZ3   ;IF FLG SET
00107  9D44  20 67 9E            JSR        IORD
I00109  9D49               IG
```



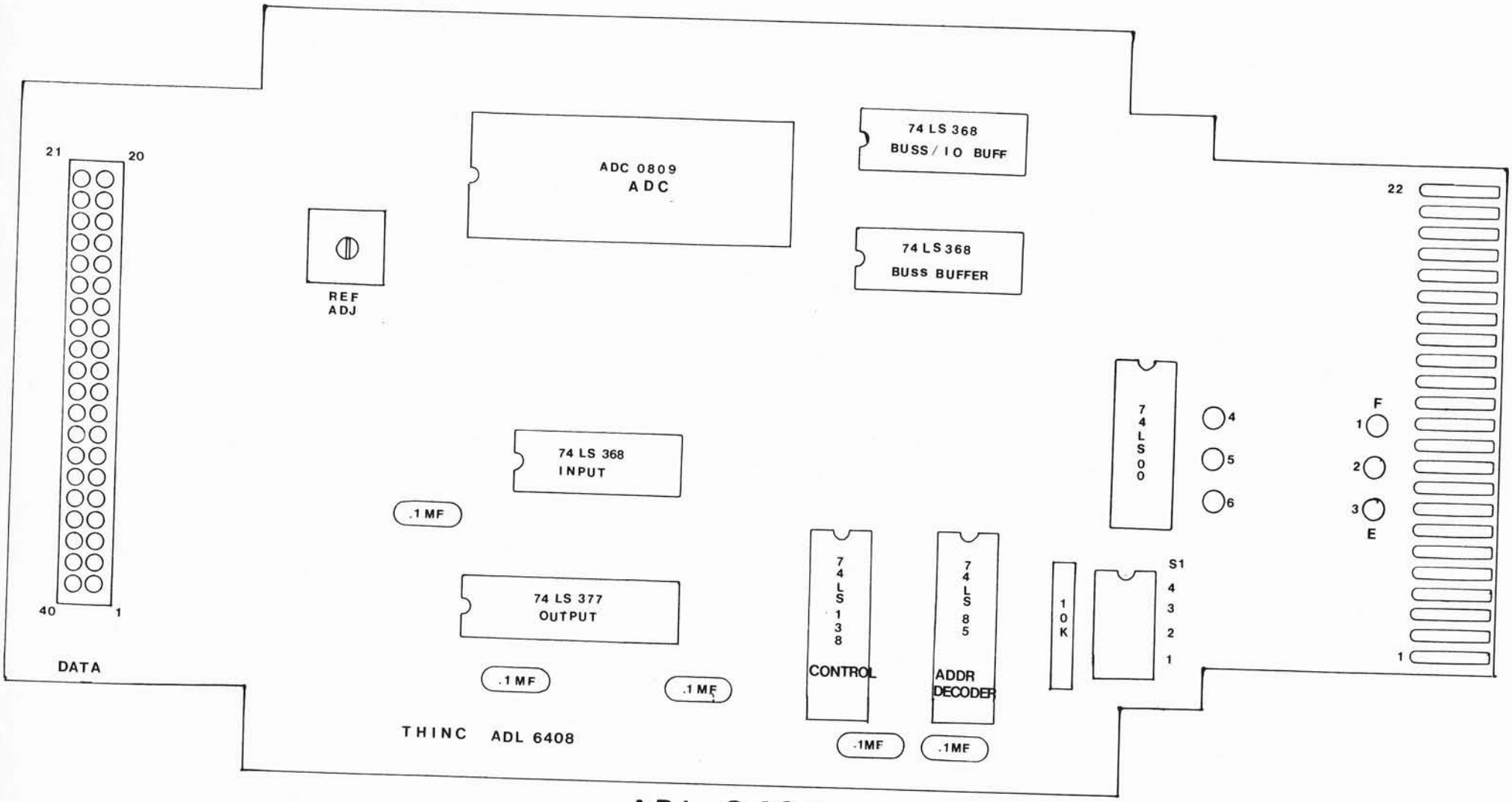
SIGNAL ATTENUATION
FIGURE 2



SIGNAL AMPLIFICATION
FIGURE 3



CURRENT MEASUREMENT
FIGURE 4



ADL 6408

Component Placement