DR. 927

LA-7894

3/21/80

MASTER

Data Acquisition and Command System for Use with a Microprocessor-Based Control Chassis

University of California

LOS ALAMOS SCIENTIFIC LABORATORY

LA-7884
UC-32
Issued: January 1980

Data Acquisition and Command System for Use with a Microprocessor-Based Control Chassis

J. K. Halbig
S. F. Klosterbuer
V. A. Martinez, Jr.

DATA ACQUISITION AND COMMAND SYSTEM FOR USE WITH A MICROPROCESSOR-BASED CONTROL CHASSIS

by

J. K. Halbig, S. F. Klosterbuer, and V. A. Martinez, Jr.

ABSTRACT

The Pion Generation for Medical Irradiations (PIGMI) program at the Los Alamos Scientific Laboratory is developing the technology to build smaller, less expensive, and more reliable proton linear accelerators for medical applications. We have designed a powerful, simple, inexpensive, and reliable control and data acquisition system that is central to the program development. The system is a NOVA-3D minicomputer interfaced to several outlying microprocessor-based controllers, which accomplish control and data acquisition through data I/O chassis. We describe the equipment interface chassis, which can issue binary commands, read binary data, issue analog commands, and read timed and untimed analog data.

I. INTRODUCTION

The microprocessor is used best as a controller rather than as a mini-minicomputer. Novel examples of an intelligent controller are given in Refs. 1 and 2. To interface with the outside world, the smart controller must be able to send commands in analog or binary form. For example, an analog command can provide a programmable voltage into the summing junction of a power supply controller; a binary command turns the power supply on and off. Also, the controller must be able to acquire analog and digital data. The analog voltage across a shunt in the power supply output is an example of analog data; the on-off status of the supply is an example of binary data. Here, analog data are "untimed" data, inasmuch as the power supply output is considered to be constant in time. In contrast, timed analog data might vary as a function of time after a specified trigger. An example is the response of a current monitor along a linear accelerator (linac), in which a burst of current is periodically transported along the accelerator. The rising edge of the beam gate is the trigger.

The design of the data interface to the microprocessor required dependable, versatile, and easy-to-use building blocks. We designed four basic modules consisting of 11.5- by 16.5-cm printed circuit (PC) boards: analog command, binary command, analog data, and binary data modules.

The analog command module provides on a single board four channels of 0- to +10-V or −10 to +10-V drive signals with 12-bit resolution.

The binary command module also consists of one board that provides eight contact closures and has facility for external interlocks. That is, unless an external interlock is made up, the controller cannot close the command contacts. Also, there are eight data inputs to this board that indicate the command status. They can be jumpered to the on-board relay coil that actuates the contacts or to the device being controlled by the contact closure. In this function,

the binary command module also returns data to the processor.

The analog data module can comprise up to three boards. The 16-channel, 12-bit A/D (analog-to-digital) converter board is required; it converts 0- to +10-V signals or −10- to +10-V signals. Conversion can be triggered by an external begin-convert signal or by the processor. This board can be used with either or both of the other two boards, which are the sample-hold and the analog amplifier boards. The sample-hold board has 16 channels of sample-hold electronics that are triggered by an external timer. A timing board that can run several modules has also been implemented. The module design concept includes an analog amplifier board that would contain 16 channels of amplification; such a card has not been developed for this system; however, a 16-channel amplifier board has been designed at LASL, which fits on this size board. The I/O format is not compatible with the EIC.[3]

The binary data module can read 32 binary channels. The inputs are TTL gates pulled up with 3.3-kΩ resistors and are active low.

The small inexpensive boards can be mounted near the controlled device for one-of-a-kind applications. This is especially desirable for the analog part of a system because the interface to the processor is much less susceptible to noise than are the low-level analog signals. These cards can be mounted in a convenient compact chassis to form a data system. We are using an equipment interface chassis (EIC) with an MC6800-microprocessor-based controller, but use of these boards makes the controller interface virtually processor-independent. That is, the controller may be 6800-, 6502-, 8080-, Z-80-, or any other processor-based, as long as the processor can provide 16 bits of interface and 4 handshake lines.

II. EQUIPMENT INTERFACE CHASSIS

The EIC is shown in Figs. 1a-c. The front view (Fig. 1a) shows the power switch and ac indicator lights. The switch controls the power supply, shown in Fig. 1b, which provides +5 V for the TTL logic on the boards and ±15 V for the analog components.

The boards slide into a 16-slot board cage, which is accessed through the cutout in the rear panel. The controller interface cables also leave through this opening. The data signals enter the EIC via the connectors on the left rear of the chassis (Fig. 1c). Then they are transmitted to the board-edge connectors on the boards in the cage. There is a connector on the back panel for each module.

We wanted a self-contained data crate that is easy to disassemble, move, and reassemble. The cables attached at the rear of the EIC can be "permanently" attached to a junction strip or to the signal source.

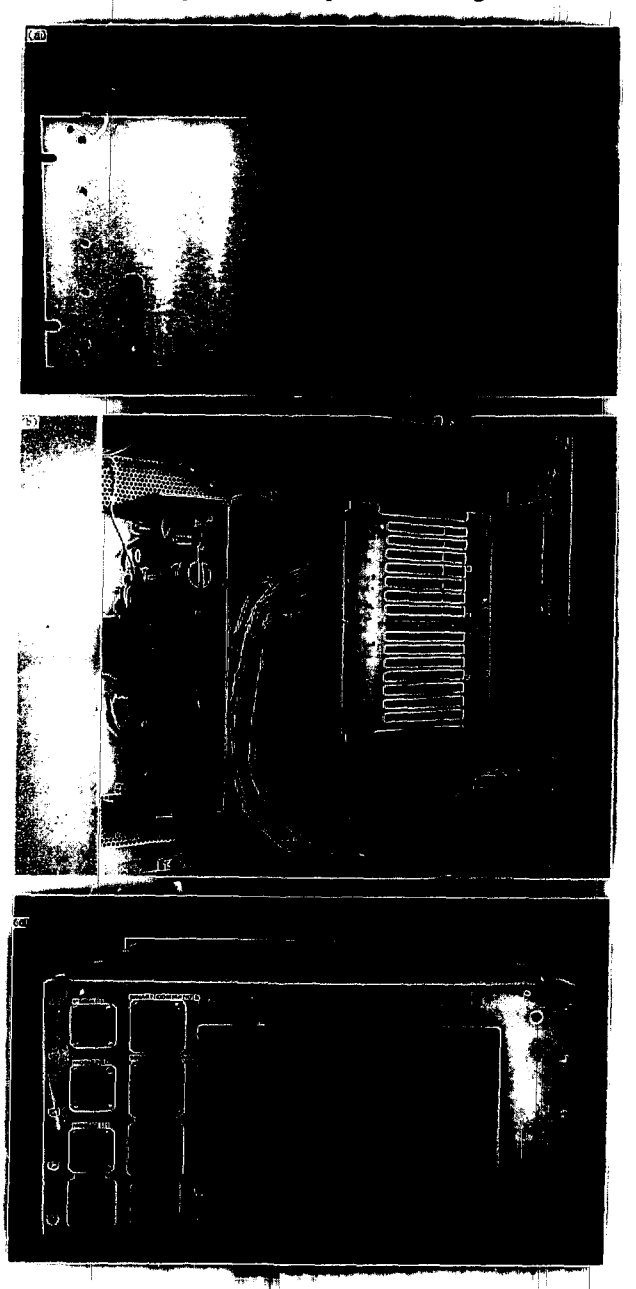


Fig. 1.
The EIC chassis (a) front, (b) top, (c) rear.

A slot guide for the modules is shown in Fig. 2. A board is required in slot A in the analog module because the external data lines are attached to that slot. If only the A/D board is used, it must be in slot A. If the sample-hold board or the amplifier board is used with the A/D board, it must be in slot A, and the A/D board will be in slot B. If the third board is necessary, it will go into slot A in a similar manner. Slot 1 is reserved for a timing board if it is used. Boards for the other modules have specific locations, as shown in Fig. 2.

One disadvantage of the EIC is the large number of cables that pass between the EIC and the microprocessor chassis (MPC) when several EIC boards are used. To overcome this, one could design an intelligent single-board controller with the peripheral interface adapter (PIA) boards located in the EIC. Only 4 lines of serial communication, or 20 lines of parallel communication, would be required between the MPC and the EIC. The cost of this simplification is development of a communication protocol for the MPC and EIC.

III. INTERFACE MODULES

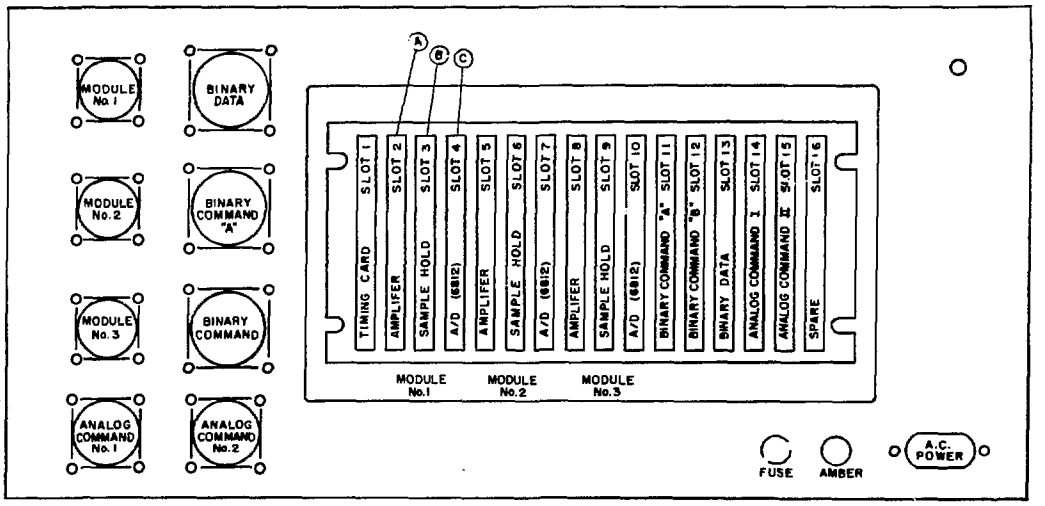
A. Binary Data Module Hardware

Figure 3 is the schematic of the binary data module. There are 32 input channels on the 44-pin board edge. The commons and inputs are wired to the connector on the rear panel. When used with a junction strip, as a minimum, twisted-pair wires should go from the strip to the signal device. The input lines are connected to the inputs of inverting TTL gates. The inputs are pulled up with 3.3-kΩ resistors; this provides logic 0s to the processor when nothing is connected to the card. The input must be pulled low to provide a logic 1 to the processor. The card uses the EIC's +5-V bus, and the MPC common enters through the ribbon cables.

B. Binary Data Module Software

The 32 data lines on this board are interfaced with the EIC by two ribbon cables that connect to two PIAs. The PIA peripheral data lines must be programmed as inputs to the microprocessor. No special handshake pulse is needed on the binary data read; hence, the A- and B-side control registers, CRA and CRB, are initialized to a $04 ($ indicates a hexadecimal number). When the PIAs are properly initialized, data from a specific channel can be read by executing a load from the proper PIA peripheral data register and performing a test on the bit corresponding to the channel being read.

The sample program in listing 1 demonstrates initialization of the PIA and reading and storing data from the 32 channels in a RAM buffer.



NOTE:
① SLOTS 1 THRU 15 PIN Z & ZZ TOWARD TOP OF CRATE
② SLOT 16 PIN A & 1 TOWARD TOP OF CRATE

Fig. 2.
EIC chassis PC board locations.

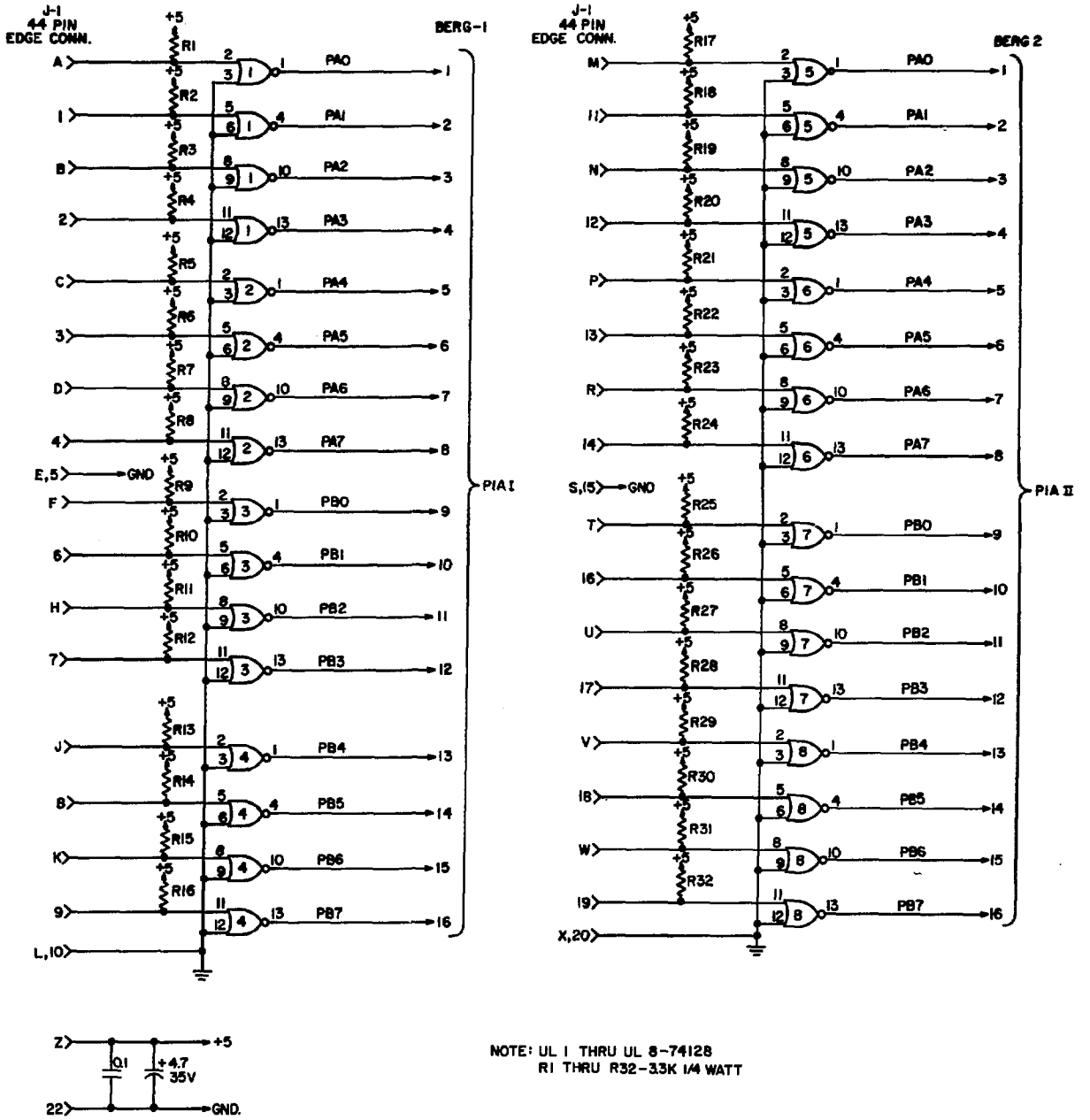


Fig. 3.
Binary data card schematic.

C. Analog Data Module Hardware

The analog data module, shown schematically in Figs. 4a-c, consists of three boards. The primary board is the 16-channel A/D board, which is interfaced to the MPC via a single PIA cable. This module is designed for single-ended input, but the 6812 module can be used for differential input. The multiplexer (mux) address is given to the 6812 on the PB3-PB0 lines. The strobe line that begins the conversion can be jumper-connected (J5) to the CB2 line, which is controlled by the processor, or to an external timing line. When the conversion is complete, a signal is returned to the processor on the CA1 line. The maximum throughput rate on untimed data is ~30k conversions per second.

Listing 1.

LASL Identification
No. LP-1068

```
PAGE  001   BDATA

00001                        NAM    BDATA
00002                  *
00003                  *     BINARY DATA CARD
00004                  *
00005                  *     THIS IS A SIMPLE PROGRAM THAT DEMONSTRATES
00006                  *     INITIALIZATION AND USE OF THE BINARY DATA
00007                  *     CARD.  IT READS VALUES FROM THE 32 CHANNELS
00008                  *     AND STORES THEM IN RAM WHERE THEY COULD BE
00009                  *     MANIPULATED BY THE USER AS DESIRED.
00010                  *
00011       8900     BDATA1 EQU    $8900    FIRST PIA CHANNELS(1-16)
00012       8902     BDATA2 EQU    $8902    SECOND PIA CHANNELS(17-32)
00013       A050     BUFFER EQU    $A050    ADDRESS OF RAM BUFFER
00014                  *
00015 F400                   ORG    $F400
00016 F400 CE 0000           LDX    #$0000   INITIALIZE PIAS
00017 F403 FF 8908           STX    BDATA1+8
00018 F406 FF 890A           STX    BDATA2+8
00019 F409 FF 8900           STX    BDATA1   CONFIGURE AS INPUTS
00020 F40C FF 8902           STX    BDATA2   *
00021 F40F CE 0404           LDX    #$0404   SET CONTROL REGISTERS
00022 F412 FF 8908           STX    BDATA1+8
00023 F415 FF 890A           STX    BDATA2+8
00024                  *
00025 F418 FE 8900           LDX    BDATA1   READ FIRST 16 CHANNELS
00026 F41B FF A050           STX    BUFFER   PUT IN RAM BUFFER
00027 F41E FE 8902           LDX    BDATA2   READ NEXT 16 CHANNELS
00028 F421 FF A052           STX    BUFFER+2
00029 F424 39                RTS
00030                  *
00031                        END

TOTAL ERRORS 00000

!
```

This board can randomly address a mux channel by using the PB3-PB0 lines (the single-channel mode), or by setting the first channel. Then, each time data from the present channel are read, the mux channel is incremented. The last mode is the autosequencing mode.

The board requires ±15 V at 45 mA, and +5 V at 275 mA. The voltages are supplied by the EIC, but in one-of-a-kind applications, the 5 V may be jumper-connected to the 5-V line of the PIA cable and then only the ±15 V need be provided externally by the EIC.

The sample-hold board, shown in Fig. 4b, has 16 sample-hold channels, each with an amplifier and a sample-hold. The on-board amplifier was designed as a buffer amplifier, but it may be used as a voltage amplifier or current-to-voltage converter.

The LF298 sample-hold circuitry has a 10-μs acquisition time to 0.1%. Therefore, if the device is holding a 10-V signal and the device is released while the input is at 0 V, a maximum of 10 μs is required for the output to settle to 0 V within 0.1%. Use of small capacitors with the LF298 enhances a phenomenon called the hold step.[4] We designed potentiometers R1-R16 into the circuit to reduce the hold step introduced into the output when the hold gate has fired.

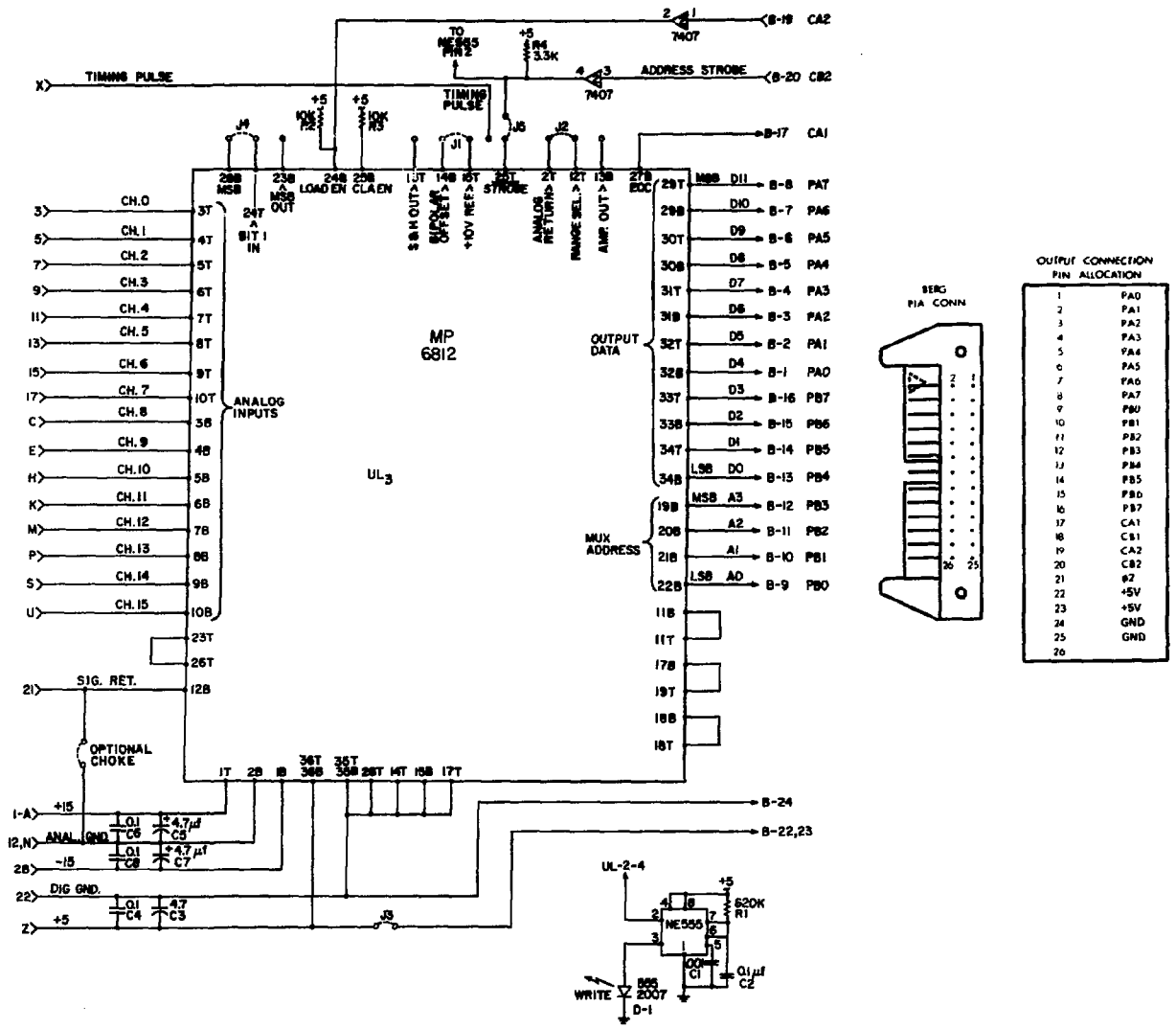


Fig. 4a.
A/D converter schematic.

The droop rate of the sample-hold with the capacitor values given is 1.8 mV/ms. We assume that the processor can address the channel and digitize the information of all the sample-holds within <1 ms after the hold gate has fired.

There is no reset on the sample-hold because it is intended for applications in which the hold is released and the zero level is reacquired before the beam pulse.

The hold gate is connected to the analog timing board (Fig. 4c), which is designed to provide a programmable time delay after an external strobe (rising edge of beam gate) to a timing output. This output can be used to trigger the start-convert gate of the 6812 on the A/D board; its inverted form can trigger the hold gate of the sample-holds. The timing board also provides a second signal on a separate line, 1-, 10-, or 100-μs (jumper-selectable) after the first signal. This line is for cascaded sample-holds, which are used to combat droop in fast-acquisition applications.[4]

Symmetrical signals of 10 MHz and 1 MHz are derived from a 20-MHz crystal controlled timing reference. Either counter may use either signal for its time base. The highest resolution on H1 is 0.1 μs.

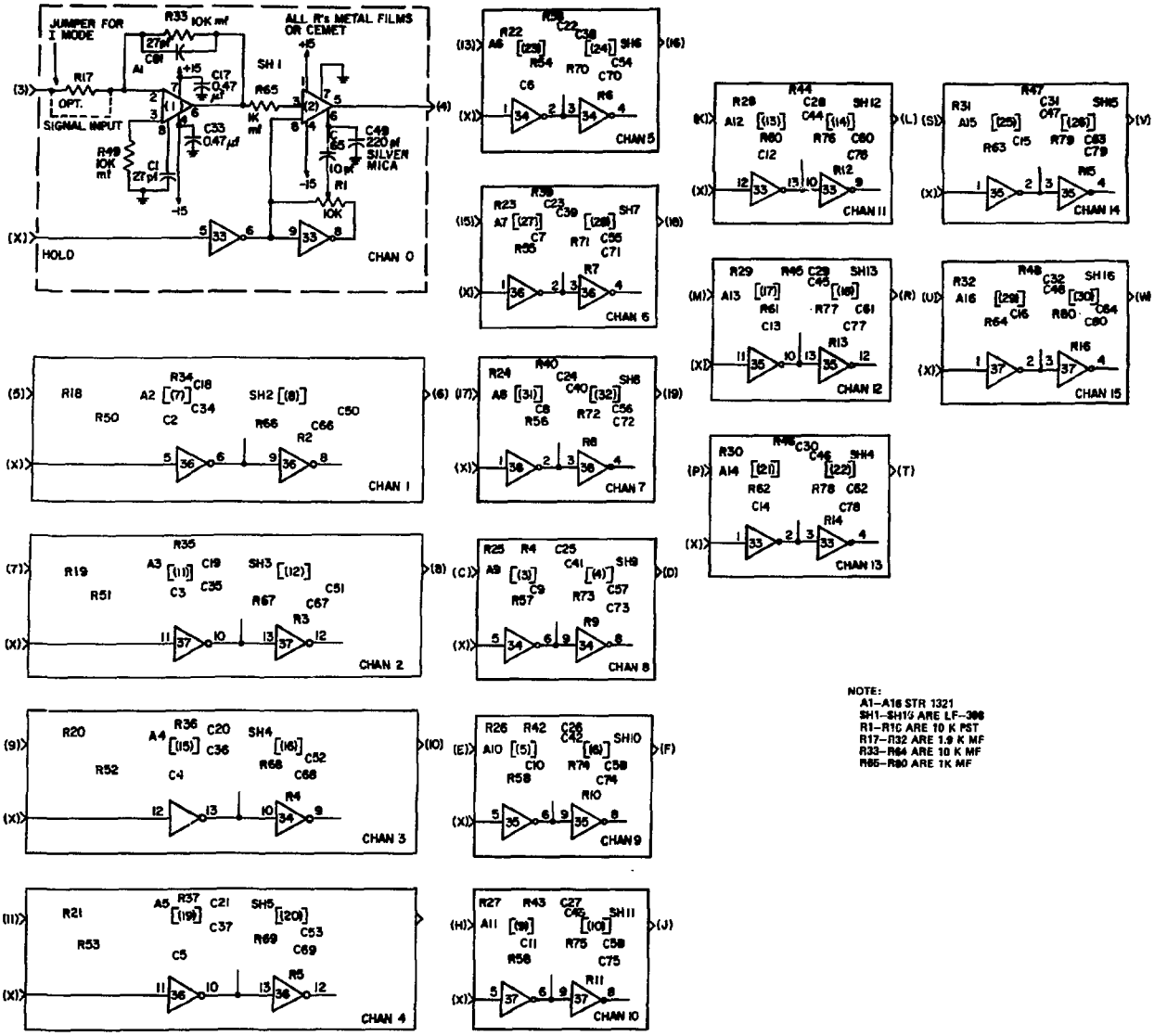


Fig. 4b.
Sample hold schematic.

The timing sequence is initiated only by the rising edge of the strobe or beam-gate signal. To operate the timing sequence, one enters on the PA3-PB0 lines the number of delay units to be held after the beam gate is passed. The lines must remain stable until the beam gate that actuates the timer is passed. A beam gate cannot start a timing sequence until 10 μs after hold-release.

The hold-release is activated by a low pulse, 0.5-9 μs wide, on the hold-release line. As soon as the low pulse is entered on this line, the H1 and H2 lines are reset (made inactive), which allows the sample-hold to acquire the signal. The CA1 line also receives the low pulse at this time, and this can signal the processor that a timing sequence has begun. The beam gate cannot start the timing sequence for 10 μs after the line receives the low pulse. The timing hardware prevents the H1 line from going high within 10 μs after hold-release, which guarantees acquisition of the signal before hold.

After the 10-μs delay, the next beam gate latches the new delay from PA3-PB0 into the counters. At

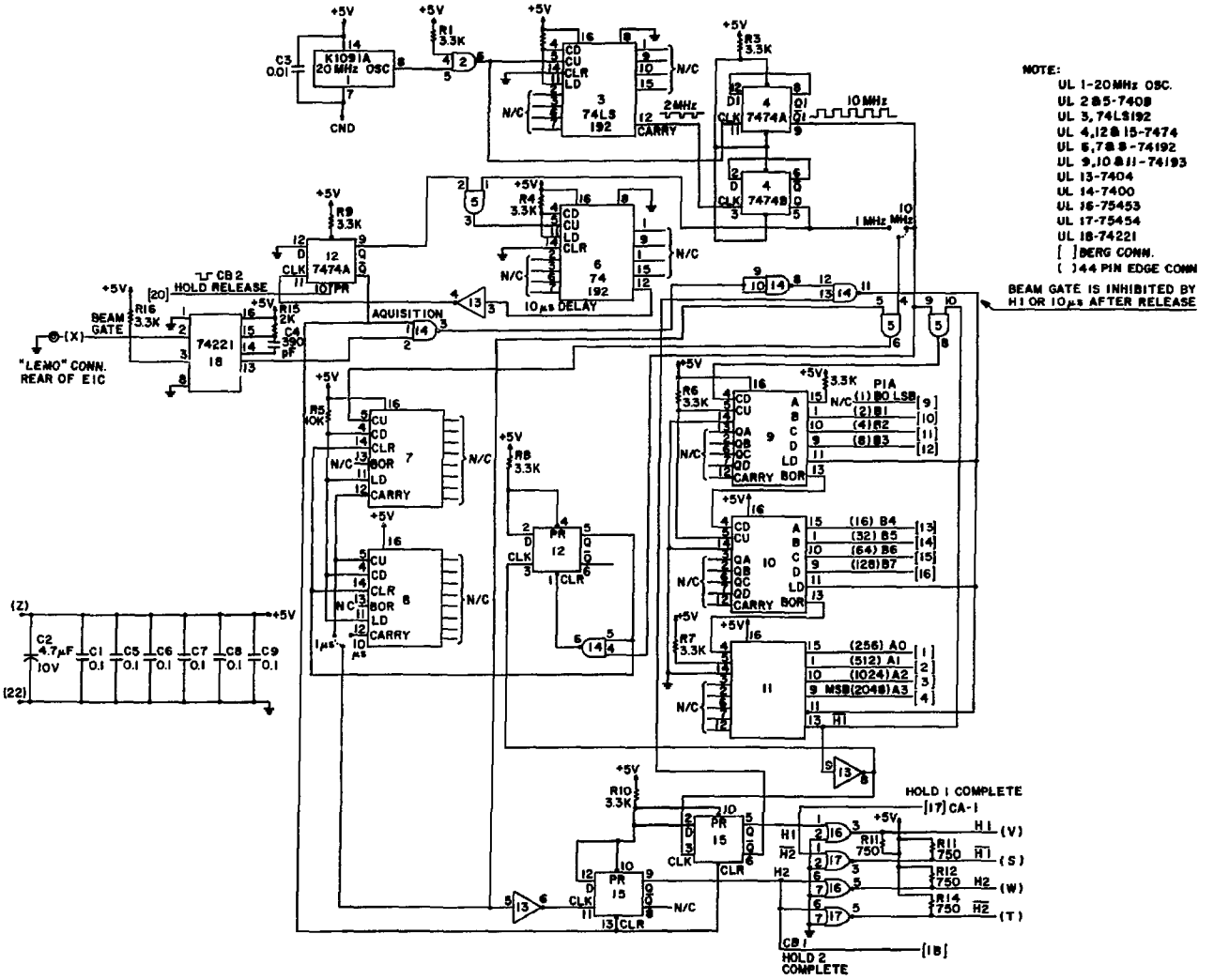


Fig. 4c.
Analog timing module schematic.

the same time, the borrow signal from the last counter is reset and clock pulses are gated into the counter. When the counter reaches zero, the borrow line becomes active; further pulses cannot enter the counter, and the H1 line is set high (active). This information is also available to the MPU through the CA1 line. At the same time, the borrow line clears the H2 counter, which begins to count up. When the carry bit of the selected decade is active, further pulses are inhibited and the H2 line becomes active.

D. Analog Data Module Software

1. A/D Converter Module (Untimed Data). The A/D board interface is made through a PIA. To initialize the PIA, the 12 I/O lines (PA7-PA0, PB7-PB4) that receive the data are programmed as inputs, and the four lines that address the proper channel are programmed as outputs. The role of the handshake lines can vary with specific applications.

The A/D can operate in the single-channel or the autosequencing mode; the selection is made by the software. To select a channel in either mode, the control register is programmed to set the CA2 line low ($34 in the CRA). A channel number (0-15) is entered on the PB3-PB0 lines, and then the CB2 line is pulsed to initiate conversion. (A $2C in the CRB automatically provides this strobe on a store to the B-side peripheral data register PDRB). When the channel number is entered, the single-channel mode is selected by keeping the CA2 line low; autosequencing is selected by bringing the CA2 line high.

After the strobe, there must be a 40-μs delay for the conversion to take place. The software could delay for 40 μs, but the A/D provides a high-to-low transition on the CA1 line to signal end-of-conversion (EOC). The software shown here programs the CRA to be sensitive to this edge, but not to interrupt the MPC. The CA1 flag signals the EOC and the converted number can be read on lines PA7-PA0, PB7-PB4 in 2's complement format. If operating in the autosequencing mode, the data reading causes the mux channel to be incremented. A strobe must still be provided to initiate conversion. The new data can be read in ~40 μs.

The sample program shown in listing 2 demonstrates initialization of the PIAs, selection of a channel and a sequencing mode, and collection of the data.

2. Timed Data Acquisition. The timing board is interfaced through a single PIA. the PIA initialization must specify lines PA3-PA0, PB7-PB0 as outputs through which the 12-bit delay time will be passed to the timing board. The CB2 line provides the hold-release pulse and must normally be high ($3C in CRB). The CA1 line can be used to indicate that the delay has been completed; the control register must be configured to set the IRQA1 flag (bit 7) on the rising edge of the CA1 line ($06 in the CRA).

To set a timing delay, the desired delay time must be entered on lines PA3-PA0, PB7-PB0 [MSB-LSB (most significant bit-least significant bit)]. Then a hold-release pulse is applied by bringing the CB2 from high-to-low-to-high. The CB2 line can be used in the pulse mode if a store precedes each hold-release. At the next beam gate occurring at least 10 μs after the hold-release, the delay entered on the data lines will be latched into the counters, and the countdown will begin. When the counter reaches zero, H1 becomes active. The active H1 can be used to provide the start-convert signal to the A/D board for applications where timed data are acquired without the use of the sample-hold board. The software must wait for an EOC from the A/D board in the manner described earlier.

If the data system consists of a timing board, a sample-hold board, and an A/D board, the software is slightly different. The delay is set in the timing board in the above manner, but when the delay has been completed, the CA1 line will go from low to high, which sets the IRQA1 (bit 7 of CRA), and the sample-holds are triggered. The software uses the signal from the timing board's CA1 line to determine when to initiate the reading of the A/D board. When the hold is in effect, the software provides its own start-convert pulse using the CB2 line on the A/D board, and reads the data as described.

Listing 3 demonstrates timed data in a system consisting of an A/D board, a sample-hold board, and a timing board. The program uses the monitor loop and MPC front panel described in Ref. 1. The program's main task is in the subroutine SETTM, which reads the delay time in decimal units from the hex switch, converts it to a binary number, and stores it in the timer. It then alters the main loop subroutine to DLOOP, which is repeated each time the loop is executed. DLOOP initiates a hold-release

Listing 2.

LASL Identification
No. LP-1068

```
PAGE  001   ADATA
00001                        NAM    ADATA
00002                 *
00003                 *      ANALOG DATA CARD
00004                 *
00005                 *      THIS PROGRAM DEMONSTRATES THE INITIALIZATION
00006                 *      AND USE OF THE 16 CHANNEL 12 BIT ANALOG DATA
00007                 *      CARD.  NUM MUST CONTAIN THE NUMBER OF
00008                 *      CHANNELS TO BE READ AND CHAN MUST CONTAIN
00009                 *      THE CHANNEL NUMBER OF THE STARTING CHANNEL.
00010                 *
00011       8900      ADATA  EQU    $8900
00012                 *
00013 F000                   ORG    $F000
00014 F000 CE 0000           LDX    #$0000    INITIALIZE PIA
00015 F003 FF 8908           STX    ADATA+8
00016 F006 CE 000F           LDX    #$000F    SET DATA DIRECTION REGISTER
00017 F009 FF 8900           STX    ADATA
00018 F00C CE 342C           LDX    #$342C    SET CONTROL REGISTER
00019 F00F FF 8908           STX    ADATA+8
00020 F012 B6 8900           LDA A  ADATA     CLEAR CA1 FLAG
00021 F015 CE 8900           LDX    #ADATA    GET PIA ADDRESS
00022 F018 FF A024           STX    ADPIA     SAVE FOR SUBROUTINE
00023 F01B B6 A000           LDA A  NUM       NUMBER OF CHANNELS TO READ
00024 F01E F6 A001           LDA B  CHAN      STARTING CHANNEL
00025 F021 CE A002           LDX    #BUFFER   RAM STORAGE OF VALUES READ
00026 F024 BD F028           JSR    ADREAD    READ CHANNELS SPECIFIED
00027 F027 39                RTS
00028                 *****************************************
00029                 *
00030                 *      A/D READ ROUTINE
00031                 *
00032                 *      TO ENTER :
00033                 *      ACC A : NUMBER OF CHANNELS TO READ (1-16)
00034                 *      ACC B : STARTING CHANNEL NUMBER (0-15)
00035                 *      X REG : ADDRESS OF STORAGE BUFFER FOR
00036                 *              VALUES READ.
00037                 *
00038                 *      AT EXIT :
00039                 *      VALUES READ SAVED IN BUFFER POINTED TO BY X.
00040                 *
00041                 *
00042 F028 FF A022 ADREAD STX    STRPTR    STORAGE POINTER
00043 F02B 4A              DEC A          ADJUST NUMBER
00044 F02C B7 A026         STA A  NUMBR   SAVE
00045 F02F FE A024         LDX    ADPIA   ADPIA ADDRESS
00046 F032 86 34           LDA A  #$34    SET FOR CHANNEL SELECT
00047 F034 A7 08           STA A  8,X
00048 F036 86 2C           LDA A  #$2C    SET FOR CHANNEL SELECT
00049 F038 A7 09           STA A  9,X
00050 F03A E7 01           STA B  1,X     SELECT INITIAL CHANNEL
00051 F03C 7D A026         TST    NUMBR   HOW MANY CHANNELS TO READ
00052 F03F 27 04           BEQ    READ    ONLY ONE
00053 F041 86 3C           LDA A  #$3C    SET FOR AUTOSEQUENCING
00054 F043 A7 08           STA A  8,X
00055 F045 FE A024 READ    LDX    ADPIA   GET PIA ADDRESS
00056 F048 A6 08   READ1   LDA A  8,X     WAIT FOR EOC
00057 F04A 2A FC           BPL    READ1
00058 F04C E6 01           LDA B  1,X     READ LS BYTE
```

```
PAGE  002   ADATA
00059 F04E A6 00          LDA A  0,X       GET MS BYTE
00060 F050 C4 F0          AND B  #$F0      ONLY 4 DATA BITS
00061 F052 E7 01          STA B  1,X       STROBE CONVERT IN SEQUENTIAL
00062 F054 44             LSR A            RIGHT JUSTIFY DATA
00063 F055 56             ROR B
00064 F056 44             LSR A
00065 F057 56             ROR B
00066 F058 44             LSR A
00067 F059 56             ROR B
00068 F05A 44             LSR A
00069 F05B 56             ROR B
00070 F05C FE A022        LDX    STRPTR    STORE POINTER
00071 F05F A7 00          STA A  0,X       STORE VALUES
00072 F061 E7 01          STA B  1,X
00073 F063 08             INX
00074 F064 08             INX
00075 F065 FF A022        STX    STRPTR    ADVANCE POINTER
00076 F068 7A A026        DEC    NUMBR     ALL CHANNELS READ?
00077 F06B 2A D8          BPL    READ      NO, GO READ
00078 F06D 39             RTS
00079                  *
00080 A000                ORG    $A000
00081 A000 0001    NUM    RMB    1
00082 A001 0001    CHAN   RMB    1
00083 A002 0020    BUFFER RMB    32
00084 A022 0002    STRPTR RMB    2
00085 A024 0002    ADPIA  RMB    2
00086 A026 0001    NUMBR  RMB    1
00087             END

TOTAL ERRORS 00000
```

Listing 3

LASL Identification
No. LP-1068

```
PAGE  001   TIME
00001                          NAM    TIME
00002                  *
00003                  *       TEST OF TIMING CARD
00004                  *       THIS TIMIMG CARD TEST USES BUTTON 1
00005                  *       TO SET THE DESIRED DELAY--WHEN BUTTON 1
00006                  *       IS PUSHED THE VALUE ON THE HEX SWITCH IS
00007                  *       READ AS THE DESIRED DELAY IN DECIMAL.  THE
00008                  *       TIMER IS SET TO THE VALUE AND AT THE
00009                  *       APPROPRIATE TIME THE DESIGNATED A/D
00010                  *       CHANNEL IS READ AND DISPLAYED ON
00011                  *       THE HEX DISPLAY. THIS ACTION IS REPEATED
00012                  *       EACH TIME THROUGH THE MONITOR LOOP.
00013                  *
00014       8902       TIME   EQU    $8902    TIMING CARD PIA
00015       8900       ADATA  EQU    $8900    ANALOG DATA PIA
00016       A1E0       VCTR1  EQU    $A1E0    SOFTWARE BUTTON VECTOR
00017       A1F0       VSUB1  EQU    $A1F0    SOFTWARE LOOP VECTOR
00018       8102       HEXSW  EQU    $8102    4 DIGIT HEX SWITCH
00019       8100       HEXLD  EQU    $8100    4 DIGIT HEX DISPLAY
00020       8104       TOGGL  EQU    $8104    TOGGLE/LED
00021       F01E       RSX    EQU    $F01E    ACTIVITY LIGHTS SUBROUTINE
00022                  *
00023 A200                     ORG    $A200
00024 A200 CE 0000 RESET  LDX    #$0000   INITIALIZE PIAS
00025 A203 FF 890A        STX    TIME+8   SELECT DDR
00026 A206 FF 8908        STX    ADATA+8
00027 A209 CE 000F        LDX    #$000F   SET DATA DIRECTIONS
00028 A20C FF 8900        STX    ADATA
00029 A20F CE 0FFF        LDX    #$0FFF
00030 A212 FF 8902        STX    TIME
00031 A215 CE 342C        LDX    #$342C   SET CONTROL REGISTERS
00032 A218 FF 8908        STX    ADATA+8  CB1 L, CB2 HLH
00033 A21B CE 063C        LDX    #$063C   CA1 LTH, CB2 H
00034 A21E FF 890A        STX    TIME+8
00035 A221 CE A242        LDX    #SETTM   SET BUTTON VECTORS
00036 A224 FF A1E0        STX    VCTR1    1-SET TIMER
00037 A227 CE A236        LDX    #LOOP    DEFAULT MONITOR LOOP
00038 A22A FF A1F0        STX    VSUB1
00039 A22D CE 8900        LDX    #ADATA   SET ADPIA FOR READ
00040 A230 FF A082        STX    ADPIA
00041 A233 A6 00          LDA A  0,X      CLEAR CA1 FLAG
00042 A235 39             RTS
00043                  *
00044 A236 BD F01E LOOP   JSR    RSX      ACTIVITY LIGHTS
00045 A239 FE 8102        LDX    HEXSW
00046 A23C EE 00          LDX    0,X
00047 A23E FF 8100        STX    HEXLD    PEEK AT MEMORY
00048 A241 39             RTS
00049                  *
00050       A242       SETTM  EQU    *        SET NEW VALUE IN TIMER
00051 A242 B6 8102        LDA A  HEXSW    READ DECIMAL VALUE
00052 A245 F6 8103        LDA B  HEXSW+1
00053 A248 BD A2EB        JSR    CV4DB    CONVERT TO BINARY
00054 A24B B7 8902        STA A  TIME     SET TIMER
00055 A24E F7 8903        STA B  TIME+1
00056 A251 CE A258        LDX    #DLOOP   NEW LOOP
00057 A254 FF A1F0        STX    VSUB1
00058 A257 39             RTS
```

```
PAGE  002   TIME
00059                 *
00060                 *
00061       A258      DLOOP   EQU     *
00062 A258  B6 8902           LDA A   TIME        CLEAR CA1 FLAG
00063 A25B  86 34             LDA A   #$34        INITIATE HOLD RELEASE
00064 A25D  B7 890B           STA A   TIME+9
00065 A260  86 3C             LDA A   #$3C
00066 A262  B7 890B           STA A   TIME+9
00067 A265  B6 890A DLOOP1    LDA A   TIME+8      WAIT FOR TIME OUT
00068 A268  2A FB             BPL     DLOOP1
00069 A26A  7F A084           CLR     DNEG        NEGATIVE NUMBER
00070 A26D  F6 8104           LDA B   TOGGL       READ CHANNEL NUMBER
00071 A270  86 01             LDA A   #1          # OF CHANNELS TO READ
00072 A272  CE A085           LDX     #BUFFER     STORAGE LOCATION
00073 A275  BD A2A2           JSR     ADREAD      READ VALUE
00074 A278  CE A085           LDX     #BUFFER     VALUES STORED HERE
00075 A27B  A6 00             LDA A   0,X         GET TWO'S COMP. VALUE
00076 A27D  E6 01             LDA B   1,X
00077 A27F  85 08             BIT A   #$08        CHECK FOR NEG. NUMBER
00078 A281  27 0B             BEQ     ADATA2      POSITIVE
00079 A283  C0 01             SUB B   #$01        UNDO TWO'S COMP.
00080 A285  82 00             SBC A   #$00
00081 A287  43                COM A
00082 A288  53                COM B
00083 A289  84 0F             AND A   #$0F        USE ONLY 12 BITS
00084 A28B  7C A084           INC     DNEG        SIGNAL NEGATVE NUMBER
00085 A28E  BD A32F ADATA2    JSR     CVB4D       CONVERT TO DECIMAL
00086 A291  7D A084           TST     DNEG        NEGATIVE NUMBER?
00087 A294  27 02             BEQ     ADATA3      NO
00088 A296  8B A0             ADD A   #$A0        ADJUST FOR DISPLAY
00089 A298  B7 8100 ADATA3    STA A   HEXLD       DISPLAY
00090 A29B  F7 8101           STA B   HEXLD+1
00091 A29E  BD F01E           JSR     RSX         ACTIVITY LIGHTS
00092 A2A1  39                RTS
```

```
PAGE  003   TIME
00094                ********************************************
00095                *
00096                *       A/D READ ROUTINE
00097                *
00098                *       TO ENTER :
00099                *       ACC A : NUMBER OF CHANNELS TO READ (1-16)
00100                *       ACC B : STARTING CHANNEL NUMBER (0-15)
00101                *       X REG : ADDRESS OF STORAGE BUFFER FOR
00102                *               VALUES READ.
00103                *
00104                *       AT EXIT :
00105                *       VALUES READ SAVED IN BUFFER POINTED TO BY X.
00106                *
00107       F024     DMICRO  EQU     $F024
00108                *
00109 A2A2 FF A0A5 ADREAD  STX     STRPTR    STORAGE POINTER
00110 A2A5 4A              DEC A             ADJUST NUMBER
00111 A2A6 B7 A0A7         STA A   NUMBR     SAVE
00112 A2A9 FE A082         LDX     ADPIA     ADPIA ADDRESS
00113 A2AC 86 34           LDA A   #$34      SET FOR CHANNEL SELECT
00114 A2AE A7 08           STA A   8,X
00115 A2B0 86 2C           LDA A   #$2C      SET FOR CHANNEL SELECT
00116 A2B2 A7 09           STA A   9,X
00117 A2B4 E7 01           STA B   1,X       SELECT INITIAL CHANNEL
00118 A2B6 7D A0A7         TST     NUMBR     HOW MANY CHANNELS TO READ
00119 A2B9 27 04           BEQ     START     ONLY ONE
00120 A2BB 86 3C           LDA A   #$3C      SET FOR AUTOSEQUENCING
00121 A2BD A7 08           STA A   8,X
00122 A2BF FE A082 START   LDX     ADPIA     PIA ADDRESS
00123 A2C2 A6 08    STRT1  LDA A   8,X       EOC?
00124 A2C4 2A FC           BPL     STRT1     WAIT FOR EOC
00125 A2C6 FE A082         LDX     ADPIA     GET PIA ADDRESS
00126 A2C9 E6 01           LDA B   1,X       READ LS BYTE
00127 A2CB A6 00           LDA A   0,X       GET MS BYTE
00128 A2CD C4 F0           AND B   #$F0      ONLY 4 DATA BITS
00129 A2CF E7 01           STA B   1,X       TRIGGER NEXT IF IN SEQUENTIA
00130 A2D1 44              LSR A             RIGHT JUSTIFY DATA
00131 A2D2 56              ROR B
00132 A2D3 44              LSR A
00133 A2D4 56              ROR B
00134 A2D5 44              LSR A
00135 A2D6 56              ROR B
00136 A2D7 44              LSR A
00137 A2D8 56              ROR B
00138 A2D9 FE A0A5         LDX     STRPTR    STORE POINTER
00139 A2DC A7 00           STA A   0,X       STORE VALUES
00140 A2DE E7 01           STA B   1,X
00141 A2E0 08              INX
00142 A2E1 08              INX
00143 A2E2 FF A0A5         STX     STRPTR    ADVANCE POINTER
00144 A2E5 7A A0A7         DEC     NUMBR     ALL CHANNELS READ?
00145 A2E8 2A D5           BPL     START     NO, GO READ
00146 A2EA 39              RTS
00147                *
```

```
PAGE  004   TIME
00149                ********************************************
00150                *
00151                *       CONVERT 4 DIGIT DECIMAL TO BINARY
00152                *
00153                *       12 BIT BINARY--11 BITS+SIGN
00154                *
00155                *       TO ENTER:
00156                *       A = 2 MS DECIMAL DIGITS
00157                *       B = 2 LS DECIMAL DIGITS
00158                *
00159                *       AT EXIT:
00160                *       A = MS BYTE OF CONVERTED NUMBER
00161                *       B = LS BYTE OF CONVERTED NUMBER
00162                *
00163 A2EB 7F A0AC CV4DB  CLR     BINUPR
00164 A2EE B7 A0AB       STA A   SAVEA      SAVE MS DIGITS
00165 A2F1 17             TBA
00166 A2F2 C4 0F          AND B   #$0F       SAVE ONLY ONES VALUE
00167 A2F4 44             LSR A              MOVE TENS VALUE
00168 A2F5 44             LSR A              TO LOWER BITS OF A
00169 A2F6 44             LSR A
00170 A2F7 44             LSR A
00171 A2F8 27 05    TEN   BEQ     DOHUND     DO HUND WHEN TENS=0
00172 A2FA CB 0A          ADD B   #10        ADD 10 TO TOTAL
00173 A2FC 4A             DEC A              DECREMENT 10S DIGIT
00174 A2FD 20 F9          BRA     TEN
00175 A2FF 0C       DOHUND CLC               RESET CARRY
00176 A300 B6 A0AB        LDA A   SAVEA      GET HUND AND THOU DIGIT
00177 A303 84 0F          AND A   #$0F       SAVE 100S DIGITS
00178 A305 27 0A    HUN1  BEQ     DOTHOU     DO 1000S WHEN 100S=0
00179 A307 CB 64          ADD B   #100       ADD 100 TO TOTAL
00180 A309 24 03          BCC     HUN2
00181 A30B 7C A0AC        INC     BINUPR     ADD 256 TO TOTAL
00182 A30E 4A       HUN2  DEC A              DECREMENT 100S DIGIT
00183 A30F 20 F4          BRA     HUN1
00184 A311 B6 A0AB DOTHOU LDA A   SAVEA      GET 100S AND 1000S DIGITS
00185 A314 44             LSR A              MOVE 1000S TO LOWER
00186 A315 44             LSR A              4 BITS
00187 A316 44             LSR A
00188 A317 44             LSR A
00189 A318 B7 A0AB        STA A   SAVEA
00190 A31B 27 0E          BEQ     THOU1      1000S=0?
00191 A31D B6 A0AC        LDA A   BINUPR
00192 A320 0C       THOU  CLC               RESET CARRY
00193 A321 CB E8          ADD B   #$E8       ADD 1000 TO TOTAL
00194 A323 89 03          ADC A   #$03
00195 A325 7A A0AB        DEC     SAVEA      DECREMENT 1000S DIGIT
00196 A328 26 F6          BNE     THOU
00197 A32A 39             RTS
00198 A32B B6 A0AC THOU1  LDA A   BINUPR
00199 A32E 39             RTS
```

```
PAGE  005   TIME
00201                 ********************************************
00202                 *
00203                 *       CONVERT BINARY TO 4 DIGIT DECIMAL
00204                 *       12 BIT BINARY--11 BITS+SIGN
00205                 *
00206                 *       TO ENTER:
00207                 *         A = MS BYTE OF NUMBER TO CONVERT
00208                 *         B = LS BYTE OF NUMBER TO CONVERT
00209                 *
00210                 *       AT EXIT:
00211                 *         A = 2 MS DECIMAL DIGITS
00212                 *         B = 2 LS DECIMAL DIGITS
00213 A32F CE A38B CVB4D  LDX     #CONST    POINT TO 1000
00214 A332 B7 A080       STA A   NUM       SAVE BINARY DIGITS
00215 A335 F7 A081       STA B   NUM+1
00216 A338 7F A0A9       CLR     DIGIT+1
00217 A33B 7F A0AA       CLR     DFLAG     ADD TO UPPER DIGIT
00218 A33E 8D 1E         BSR     SUB       DETERMINE MS (4TH) DIGIT
00219 A340 08            INX               POINT TO 100
00220 A341 08            INX
00221 A342 7C A0AA       INC     DFLAG     ADD TO LOWER DIGIT
00222 A345 8D 17         BSR     SUB       DETERMINE 3RD DIGIT
00223 A347 B6 A0A9       LDA A   DIGIT+1
00224 A34A B7 A0A8       STA A   DIGIT
00225 A34D 08            INX               POINT TO 10
00226 A34E 08            INX
00227 A34F 7F A0A9       CLR     DIGIT+1
00228 A352 7F A0AA       CLR     DFLAG     ADD TO UPPER DIGIT
00229 A355 8D 07         BSR     SUB       DETERMINE 2ND DIGIT
00230 A357 FB A0A9       ADD B   DIGIT+1   DETERMINE 1ST DIGIT
00231 A35A B6 A0A8       LDA A   DIGIT
00232 A35D 39            RTS
00233                 *
00234 A35E B6 A080 SUB   LDA A   NUM       SUBTRACT CONSTANTS
00235 A361 F6 A081       LDA B   NUM+1     GET BINARY DIGITS
00236 A364 E0 01   SUB1  SUB B   1,X       SUBTRACT CONSTANTS
00237 A366 A2 00         SBC A   0,X
00238 A368 25 16         BCS     SUB3      TOO MANY SUBTRACTIONS
00239 A36A 7D A0AA       TST     DFLAG     ADD TO WHICH DIGIT
00240 A36D 27 05         BEQ     SUB2
00241 A36F 7C A0A9       INC     DIGIT+1   ADD TO LOWER DIGIT
00242 A372 20 F0         BRA     SUB1
00243 A374 36       SUB2 PSH A
00244 A375 86 10         LDA A   #$10      ADD TO UPPER DIGIT
00245 A377 BB A0A9       ADD A   DIGIT+1
00246 A37A B7 A0A9       STA A   DIGIT+1
00247 A37D 32            PUL A
00248 A37E 20 E4         BRA     SUB1
00249 A380 EB 01   SUB3  ADD B   1,X       UNDO LAST SUBTRACTION
00250 A382 A9 00         ADC A   0,X
00251 A384 B7 A080       STA A   NUM       SAVE BINARY DIGITS
00252 A387 F7 A081       STA B   NUM+1
00253 A38A 39            RTS
00254                 *
00255 A38B 03E8    CONST FDB     1000
00256 A38D 0064          FDB     100
00257 A38F 000A          FDB     10
0
```

```
PAGE  006   TIME
00259 A080                    ORG     $A080
00260 A080 0002     NUM       RMB     2
00261 A082 0002     ADPIA     RMB     2
00262 A084 0001     DNEG      RMB     1
00263 A085 0020     BUFFER    RMB     32
00264 A0A5 0002     STRPTR    RMB     2
00265 A0A7 0001     NUMBR     RMB     1
00266 A0A8 0002     DIGIT     RMB     2
00267 A0AA 0001     DFLAG     RMB     1
00268 A0AB 0001     SAVEA     RMB     1
00269 A0AC 0001     BINUPR    RMB     1
0027
00271                         END

TOTAL ERRORS 00000

!
```

by pulsing the CB2 line, and this initiates the countdown. At completion of the countdown, the sample-holds are triggered, and the CA1 line on the timer PIA goes high. The routine shows the processor looping until it receives the hold-complete signal. Then it selects and reads the channels by using the A/D routine described above. It displays the value read in decimal format on the hex display in 1/100-V units, e.g., 1024 represents 10.24 V, 528 represents 5.28 V. Negative values are represented by

AXXX = −0XXX, BXXX, = −1XXX, and CXXX = −2XXX .

In addition to the main routine, this software contains the modules used to read the A/D (ADREAD), to convert a four-digit decimal number to binary (CV4DB), and to convert a four-digit binary number to decimal (CVB4D).

E. Binary Command Module Hardware

The binary command board schematic is shown in Fig. 5. It provides for 8 bits of binary command in the form of a relay contact closure. The maximum contact ratings are 0.5 A, 100 Vdc, or 10 VA. The command is made by placing the desired bit pattern on the PB7-PB0 lines and strobing the CB2 line low. The data are latched on all channels when the CB2 line returns high, and are transferred to each channel relay if the relay interlock has been made up. Otherwise, the data are ignored. The interlock is made up by providing a low input to the TTL gate. The gate is held high by a 3.3-kΩ resistor.

Finally, there is a provision for "latch back data" on this board. These data indicate the status of the controlled device. The PA7-PA0 lines can be connected by a jumper to the relay coil voltages or to an external signal. The active signal must pull down a TTL gate that is being pulled high by a 3.3-kΩ resistor.

F. Binary Command Module Software

The interface to the 16 data lines on the binary command board is made through a single PIA. The PIA is configured so that the 8 A-side I/O lines are inputs and the 8 B-side I/O lines are outputs. The binary command board requires a pulse to fire the latches after the data have been entered on the board. This pulse can be generated by programming the CRB with a $2C, which provides a high-low-high pulse for each write to the PDRB.

When the PIA has been initialized, a binary command is issued by storing the desired 8-bit binary word in the PDRB. If the data lines are jumpered to the relay coil voltage, the command status can be read immediately from the A-side peripheral data register PDRA. If the data lines are jumpered to external sources, a delay is necessary to provide for the relay mechanical response and for the controlled device response before the data are read from the PDRA. Listing 4 shows how to initialize the PIAs and how to send and read back the status of a command.

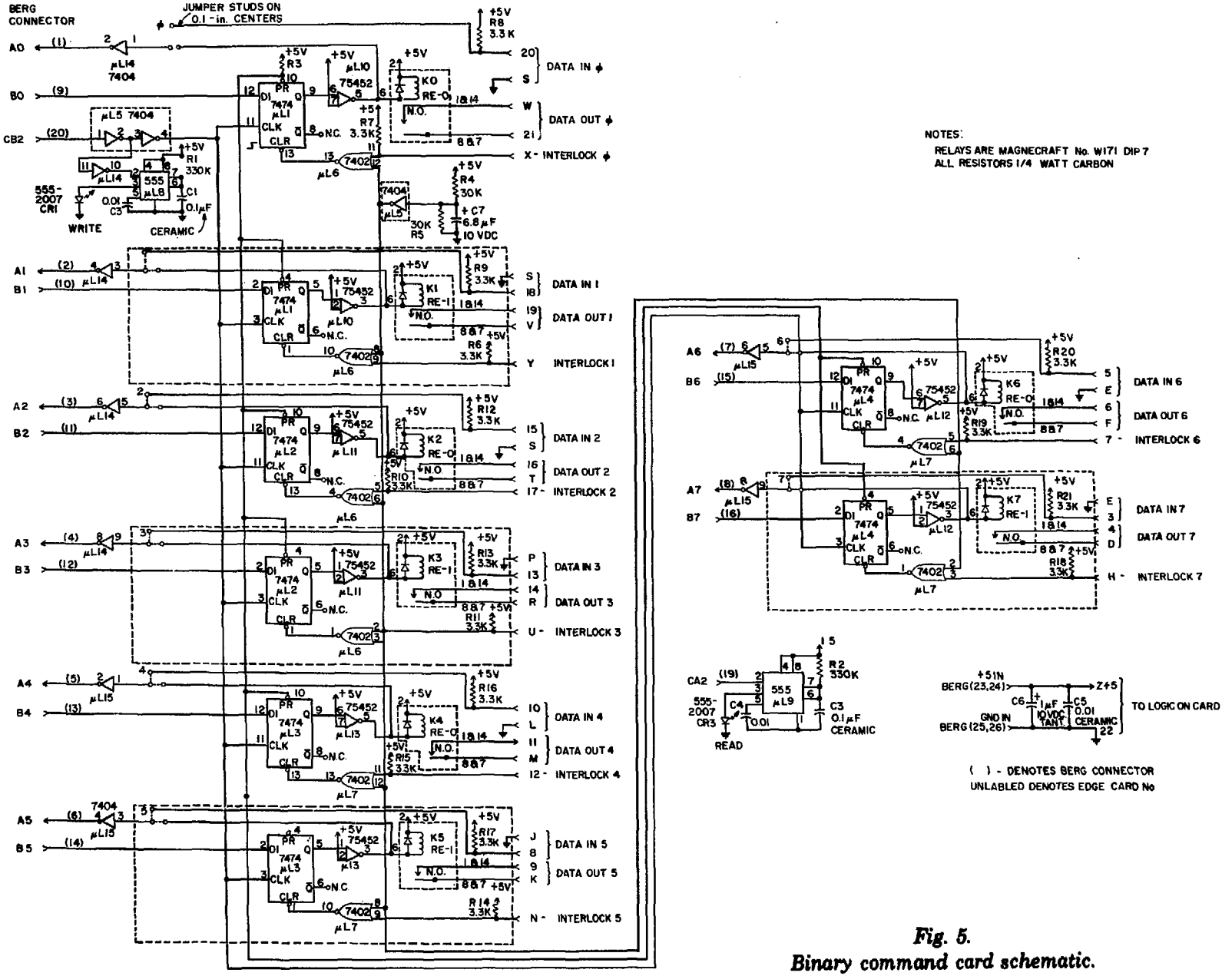


Fig. 5.
Binary command card schematic.

Listing 4.

LASL Identification
No. LP-1068

```
PAGE  001    BCOMD

00001                          NAM     BCOMD
00002                  *
00003                  *       BINARY COMMAND CARD
00004                  *
00005                  *       THIS PROGRAM DEMONSTRATES INITIALIZATION, TH
00006                  *       SENDING OF A BINARY COMMAND, AND
00007                  *       READING BACK THE "LATCH-BACK" DATA.
00008                  *
00009        8900      BCOMD   EQU     $8900     PIA INTERFACED TO BCOMD CARD
00010        F021      DMILLI  EQU     $F021     DELAY MILLISECONDS
00011                  *
00012 F000                     ORG     $F000
00013 F000 CE 0000             LDX     #$0000    INITIALIZE PIA
00014 F003 FF 8908             STX     BCOMD+8
00015 F006 CE 00FF             LDX     #$00FF    SET DATA DIRECTION REGISTER
00016 F009 FF 8900             STX     BCOMD
00017 F00C CE 042C             LDX     #$042C
00018 F00F FF 8908             STX     BCOMD+8
00019 F012 B6 A000             LDA A   COMD      GET COMMAND TO SEND
00020 F015 B7 8901             STA A   BCOMD+1   SEND COMMAND
00021 F018 CE 000A             LDX     #10       DELAY FOR CONTACT CLOSURE
00022 F01B BD F021             JSR     DMILLI
00023 F01E B6 8900             LDA A   BCOMD     READ LATCHBACK DATA
00024 F021 39                  RTS
00025                  *
00026 A000                     ORG     $A000
00027 A000 0001        COMD    RMB     1
00028                  *
00029                          END

TOTAL ERRORS 00000

!
```

G. Analog Command Board Hardware

The schematic for the analog command module is shown in Fig. 6. The board provides four channels of programmable, 12-bit resolution analog output (0 to +10 V or −10 to +10 V). The 12-bit command data are presented on lines PA3-PA0, PB7-PB0 in 2's complement form. Lines PA4-PA7 select the channel to be programmed. A logic 1 on PA4, 5, 6, or 7 will enable output channels 0, 1, 2, or 3, respectively. Any combination of these channels may be enabled at one time. After the data are presented and the channels specified, the data are clocked into the selected channels by pulsing the CB2 line high-low-high. Because all bits are latched at the same time, there will be no transient glitches, which can arise when 12-bit data are generated from two 8-bit words set up sequentially. The data are buffered and inverted before they go to the latches. The buffering is necessary because of the limited drive of the PIA, and the inverting is necessary because the D/A converters use complementary 2's complement coding.

The latches are reset to zeros at power on, which drives the D/As to within one LSB of zero output.

The command word is latched on-board because the control signals should be independent of the state of the controlling processor. That is, if the processor goes down, the interface boards should remain in a well-defined state to prevent runaway. The on-board latches provide this assurance.

Power is supplied to the board on the 44-pin edge connector, and the ribbon cable makes the ground connection to the processor.

H. Analog Command Software

The four-channel analog command board is interfaced through a single PIA. All the PIA lines are used as outputs; a pulse is required after a store to the PDRB, so the CRB is initialized to a $2C. The channels can be selected by writing a 1 in the PA4, 5, 6, or 7 bit corresponding to the desired output channels 0, 1, 2, or 3, respectively. The data on PA3-PA0

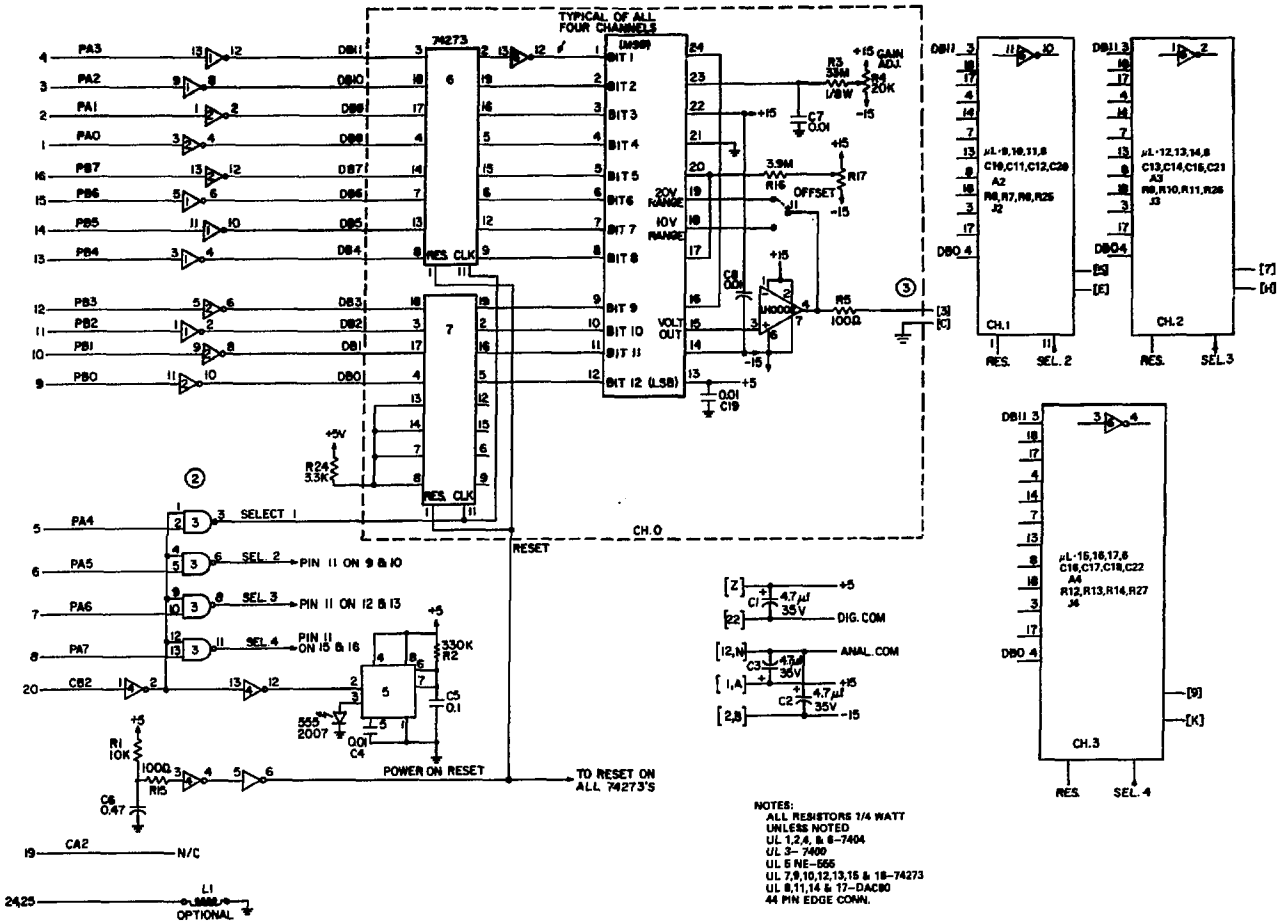


Fig. 6.
Analog command module schematic.

and PB7-PB0 must be a 12-bit 2's complement number. When this is stored in the PDRB, the CB2 line latches the data.

Listing 5 shows an example of software that will initialize the PIAs and send a command to each channel.

IV. DIAGNOSTICS

The modular hardware design greatly reduces down time because the only maintenance required is location and replacement of the bad board. A dependable scheme to check the boards thoroughly is described below. The test box is shown in Fig. 7, and Fig. 8 shows the schematic. The box has a 5- and ±15-V power supply. One of the five board slots is spare, and four are dedicated to specific cards, labeled B.C. for binary command, B.D. for binary data, A.C. for analog command, and A.D. for analog data.

A board is checked by inserting it into the checkout box and connecting it to an MPC or similar unit via a ribbon cable(s). The front panel controls provide the means to check the boards. For example, the binary command board can be made to light the individual LEDs on the test box front panel, subject to the constraint of the interlock switches. External data are provided by the left row of toggle switches. At position 1, the microprocessor should read a 1 and vice versa. The binary data board can be tested by using the toggle switches. Toggle switch N controls binary data bits N, N+8, N+16, and N+24. The analog data board is tested by using a voltage

Listing 5.

LASL Identification
No. LP-1068

```
PAGE  001    ACOMD

00001                          NAM     ACOMD
00002                  *
00003                  *       ANALOG COMMAND CARD
00004                  *
00005                  *       THIS PROGRAM DEMONSTRATES INITIALIZATION
00006                  *       OF THE ANALOG COMMAND CARD PIA AND HOW TO
00007                  *       SEND A COMMAND TO EACH OF THE CHANNELS.
00008                  *       COMD0,COMD1,COMD2,COMD3 ARE 2 BYTE WORDS.
00009                  *       CONTAINING THE COMMAND TO SEND IN ITS LOWER
00010                  *       12 BITS AND THE UPPER 4 BITS SET TO ZERO.
00011                  *       IN THIS SAMPLE PROGRAM THE DATA IS SENT TO
00012                  *       EACH CHANNEL INDIVIDUALLY BUT DATA COULD BE
00013                  *       SENT TO MORE THAN ONE CHANNEL BY SELECTING
00014                  *       THE PROPER CHANNEL NUMBERS.
00015                  *
00016        8900      ACOMD   EQU     $8900
00017                  *
00018 F000                     ORG     $F000
00019 F000 CE 0000             LDX     #$0000    INITIALIZE PIA
00020 F003 FF 8908             STX     ACOMD+8
00021 F006 CE FFFF             LDX     #$FFFF    SET DATA DIR. TO OUTPUTS
00022 F009 FF 8900             STX     ACOMD
00023 F00C CE 042C             LDX     #$042C    SET CONTROL REGISTER
00024 F00F FF 8908             STX     ACOMD+8
00025                  *
00026 F012 B6 A000             LDA A   COMD0     GET 12 BIT COMMAND
00027 F015 F6 A001             LDA B   COMD0+1
00028 F018 8A 10               ORA A   #$10      PICK CHANNEL 0
00029 F01A B7 8900             STA A   ACOMD     SEND COMMAND
00030 F01D F7 8901             STA B   ACOMD+1
00031                  *
00032 F020 B6 A002             LDA A   COMD1     GET 12 BIT COMMAND
00033 F023 F6 A003             LDA B   COMD1+1
00034 F026 8A 20               ORA A   #$20      PICK CHANNEL 1
00035 F028 B7 8900             STA A   ACOMD     SEND COMMAND
00036 F02B F7 8901             STA B   ACOMD+1
00037                  *
00038 F02E B6 A004             LDA A   COMD2     GET 12 BIT COMMAND
00039 F031 F6 A005             LDA B   COMD2+1
00040 F034 8A 40               ORA A   #$40      PICK CHANNEL 2
00041 F036 B7 8900             STA A   ACOMD     SEND COMMAND
00042 F039 F7 8901             STA B   ACOMD+1
00043                  *
00044 F03C B6 A006             LDA A   COMD3     GET 12 BIT COMMAND
00045 F03F F6 A007             LDA B   COMD3+1
00046 F042 8A 80               ORA A   #$80      PICK CHANNEL 3
00047 F044 B7 8900             STA A   ACOMD     SEND COMMAND
00048 F047 F7 8901             STA B   ACOMD+1
00049 F04A 39                  RTS
00050                  *
00051 A000                     ORG     $A000
00052 A000 0002       COMD0    RMB     2
00053 A002 0002       COMD1    RMB     2
00054 A004 0002       COMD2    RMB     2
00055 A006 0002       COMD3    RMB     2
00056                  *
00057                          END

TOTAL ERRORS 00000

!
```

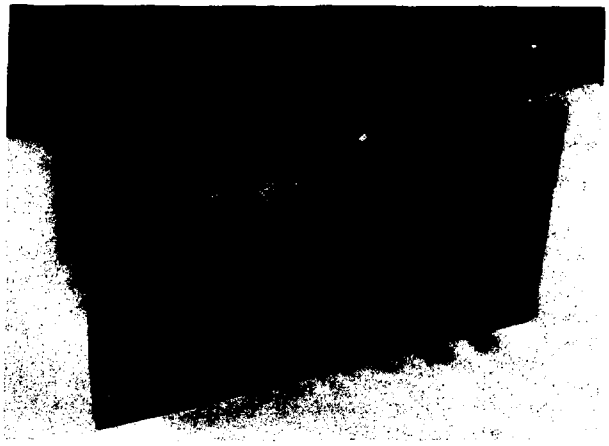


Fig. 7.
Diagnostic test box.

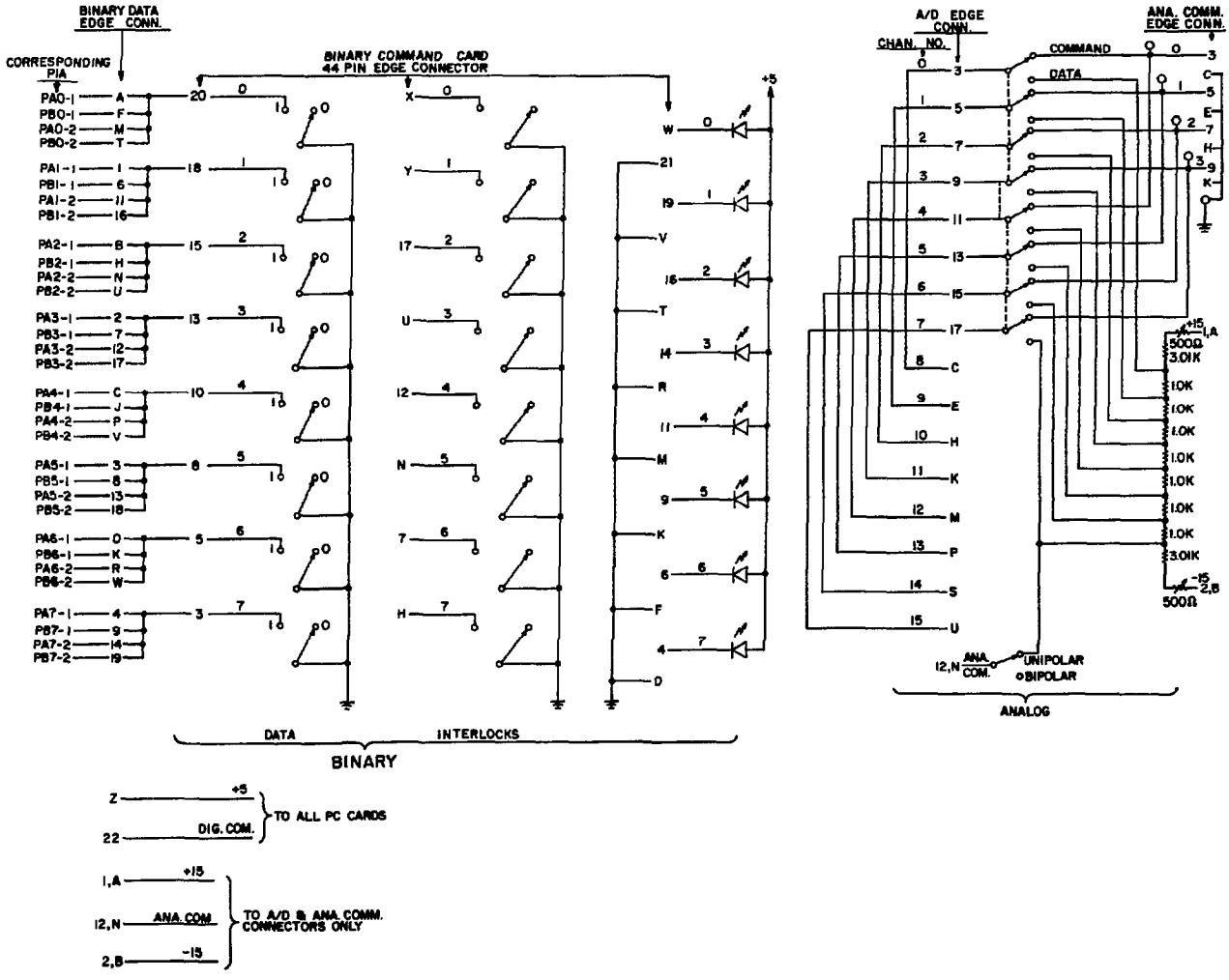


Fig. 8.
Diagnostic test box schematic.

divider to provide inputs to 8 pairs of channels. That is, the same voltage is fed to channels 0 and 8, 1 and 9, etc. The span of the voltage divider is controlled by the bipolar-unipolar switch. In the unipolar position, all taps are positive. In the bipolar position, half the taps are positive and half are negative. When used with a known working analog command board, a dynamic test may be run in which the A/D board reads a dynamic voltage pattern generated by the analog command board.

Diagnostic Software

The software in listing 6 uses panel monitor software described in Ref. 2. This software changes the monitor loop function to that defined by the subroutine SUB, which just displays memory. The PIA board slot to which the test boards are to be connected is defaulted to slot 7; button 8 can be used to change this address if desired.

The button tasks are defined as follows.

Button 1 tests the binary command board, which must be connected to the first PIA on the PIA board. Button 1 will initialize the PIA, read the value on the front panel toggle switches, and send it to the binary command board. If the interlock switches on the test box are made up, the front panel LEDs will reflect the data sent, and they will also be displayed on the LS two digits of the MPC front panel display. The data being read back will be displayed in the MS byte. The source of the data read back for each channel depends on the position of an on-board jumper. For this test, the jumper should be positioned so the state of the relays is the data.

Button 2 provides a test of all binary commands. As with the button 1 test, the board must be connected to the first PIA of the PIA board. The routine initializes the PIA and sends a $00 to the binary command board, using the same read-back features as step 1; when the data are read back, the program compares the value read (data) to the value sent (command). If the values are the same, the program will send the next value, $01, and the procedure will continue until the value $FF is processed. If the command and the data are not equal, the routine will abort and the hex display will show the noncompare condition.

Button 3 reads specified analog data channels from the analog data board, which must be connected to the second PIA on the PIA board. This routine reads the number of channels to read (binary format) from the LS four bits of the toggle switches (all switches down = 16). The starting channel is read from the MS 4 bits of the toggle switches; the channels 0-15 are in binary format. The voltages read are represented in terms of counts −2048 to +2047(−2048 is −full scale and +2047 is +full scale). Negative numbers are represented by

AXXX = −0XXX, BXXX = −1XXX, and CXXX = −2XXX

The data read from the first channel specified are displayed in decimal format in the hex display on the MPC front panel. All values read are stored in a buffer and other values may be displayed by using button 4.

Button 4 displays successive values from the channels read when button 3 was pushed. Each push of button 4 causes data from the next channel to be displayed on the hex display. The LEDs above the toggle switches represent the channel (binary format) currently being displayed. When all valid data collected by a button 3 push have been displayed, further pushes will not result in a new display.

Button 5 tests the binary data board, which must be connected to the second PIA on the PIA board. When button 5 is pushed, the toggle switches on the test box front panel provide the data for the binary data board. The data from one set of 16 channels are displayed on the MPC hex display in the hexadecimal format, and the data from the first 8 channels are also displayed on the lights above the toggle switches. To check the other 16 data channels on the binary data board, the ribbon cable must be moved to the other connector on the board.

Button 6 tests the analog command board, which must be connected to the first PIA on the PIA board. The desired voltages are represented by counts from −2048 to +2047. The desired voltage is dialed in the hex switch with numbers represented by the same scheme as described for the analog data board. The LS four toggle switches represent the analog command channels (toggle bit 0 is channel 0, toggle bit 1 is channel 1, etc.), which allows a command to be sent to any combination of channels simultaneously. When button 6 is pushed, the voltage will be sent to the designated channels. The command voltage can be monitored by reading the voltages at the test

Listing 6.

LASL Identification
No. LP-1068

```
PAGE  001    BOX
00001                          NAM     BOX
00002                    *
00003                    *     TESTS FOR DATA CARDS
00004                    *       DATA CARDS ARE CONTAINED IN THE TEXT BOX.
00005                    *       CARDS ARE CONNECTED TO PIA SPECIFIED AT
00006                    *       EACH TEST DESCRIPTION.
00007                    *
00008                    *
00009 A200                     ORG     $A200
00010                    *
00011                    *     REASSIGN BUTTON VECTORS
00012 A200 CE A290             LDX     #BCONE   1-TEST 1 BIN. COM.
00013 A203 FF A1E0             STX     VCTR1
00014 A206 CE A2B9             LDX     #BCALL   2-TEST ALL BIN. COM.
00015 A209 FF A1E2             STX     VCTR1+2
00016 A20C CE A307             LDX     #RDAD    3-READ AD CHANNELS
00017 A20F FF A1E4             STX     VCTR1+4
00018 A212 CE A367             LDX     #DSPLA   4-DISPLAY CHANNELS READ
00019 A215 FF A1E6             STX     VCTR1+6
00020 A218 CE A252             LDX     #BDATA   5-READ BIN. DATA
00021 A21B FF A1E8             STX     VCTR1+8
00022 A21E CE A3DA             LDX     #ACONE   6-SET 1 ANALOG COMD.
00023 A221 FF A1EA             STX     VCTR1+10
00024 A224 CE A426             LDX     #ACALL   7-DYNAMIC AC/AD TEST
00025 A227 FF A1EC             STX     VCTR1+12
00026 A22A CE A240             LDX     #PIADD   8-ESTABLISH PIA ADDRESS
00027 A22D FF A1EE             STX     VCTR1+14
00028 A230 CE A249             LDX     #PEEK
00029 A233 FF A1F0             STX     VSUB1    PUT PEEK IN LOOP
00030 A236 7F 8105             CLR     TOGGL+1  CLEAR TOGGLE LIGHTS
00031 A239 CE 8700             LDX     #DPIA    SET DEFAULT ADDRESS
00032 A23C FF A0AE             STX     PIA
00033 A23F 39          NOSUB   RTS
00034                    *
00035 A240 FE 8102     PIADD   LDX     HEXSW    ESTABLISH PIA ADDRESS
00036 A243 FF A0AE             STX     PIA      SAVE IT
00037 A246 8D 01               BSR     PEEK     PEEK AT MEMORY
00038 A248 39                  RTS
00039                    *
00040 A249 FE 8102     PEEK    LDX     HEXSW    DISPLAY MEMORY
00041 A24C EE 00               LDX     0,X
00042 A24E FF 8100             STX     HEXLD
00043 A251 39                  RTS
00044                    *
00045       8700       DPIA    EQU     $8700    DEFAULT PIA ADDRESS
```

```
PAGE  002   BOX
00047                ********************************************
00048                *
00049                *       BINARY DATA TEST ROUTINE
00050                *       BUTTON 5
00051                *
00052                *       BINARY DATA CARD IS CONNECTED TO
00053                *       THE SECOND PIA.
00054                *       ROUTINE INITIALIZES PIA AND DISPLAYS
00055                *       VALUES READ ON LIGHTS ABOVE TOGGLE SWITCHES.
00056                *       ALSO VALUE AND VALUE+1 AT ADDRESS OF
00057                *       PIA IS DISPLAYED ON HEXLED.
00058                *       CAN ONLY READ 16 OF THE CHANNELS: MUST
00059                *       CHANGE RIBBON CABLE FOR THE OTHER 16.
00060                *
00061 A252 CE A26B BDATA  LDX    #BDATA1   ALTER LOOP SUBROUTINES
00062 A255 FF A1F0        STX    VSUB1
00063 A258 FE A0AE        LDX    PIA       GET PIA ADDRESS
00064 A25B 4F             CLR A
00065 A25C A7 0A          STA A  $A,X      INITIALIZE BDATA PIA
00066 A25E A7 0B          STA A  $B,X
00067 A260 A7 02          STA A  2,X
00068 A262 A7 03          STA A  3,X
00069 A264 86 04          LDA A  #$04
00070 A266 A7 0A          STA A  $A,X
00071 A268 A7 0B          STA A  $B,X
00072 A26A 39             RTS
00073                *
00074 A26B FE A0AE BDATA1 LDX    PIA       PIA ADDRESS
00075 A26E A6 02          LDA A  2,X       READ BINARY DATA
00076 A270 B7 8105        STA A  TOGGL+1   DISPLAY ON TOGGLE LIGHTS
00077 A273 EE 02          LDX    2,X       READ DATA
00078 A275 FF 8100        STX    HEXLD     DISPLAY IT
00079 A278 39             RTS
00080                *
```

```
PAGE  003    BOX
00082                ********************************************
00083                *
00084                *       BINARY COMMAND TEST
00085                *       BUTTONS 1 AND 2
00086                *
00087                *       BINARY DATA CARD CONNECTED TO
00088                *       FIRST PIA.
00089                *       ROUTINE INITIALIZES PIA.
00090                *
00091                *       BUTTON 1 -BCONE -- VALUE ON
00092                *       TOGGLE SWITCHES SENT TO BINARY COMMAND
00093                *       CARD.  HEXLED THEN DISPLAYS VALUE
00094                *       READ BACK ON LEFT 2 DIGITS AND VALUE
00095                *       SENT ON RIGHT 2 DIGITS.
00096                *
00097                *       BUTTON 2- BCALL -- ALL VALUES FROM 0 TO
00098                *       256 SENT TO BINARY COMMAND CARD:
00099                *       DATA THEN READ BACK.  IF VALUES
00100                *       DIFFER ROUTINE ABORTS AT THE
00101                *       DISAGREEING VALUES.
00102                *
00103      A279      BCIN    EQU     *           INIT. BINARY COMMAND PIA
00104 A279 FE A0AE           LDX     PIA
00105 A27C 4F                CLR A
00106 A27D A7 08             STA A   8,X
00107 A27F A7 09             STA A   9,X
00108 A281 A7 00             STA A   0,X
00109 A283 B7 8105           STA A   TOGGL+1
00110 A286 4A                DEC A
00111 A287 A7 01             STA A   1,X
00112 A289 86 2C             LDA A   #$2C
00113 A28B A7 08             STA A   8,X
00114 A28D A7 09             STA A   9,X
00115 A28F 39                RTS
00116                *
00117 A290 CE A2AA BCONE     LDX     #BCOMD1     ALTER LOOP
00118 A293 FF A1F0           STX     VSUB1
00119 A296 8D E1             BSR     BCIN        INIT. BIN. COMD. PIA
00120 A298 F6 8104           LDA B   TOGGL       READ VALUE TO SEND
00121 A29B FE A0AE           LDX     PIA         ADDRESS OF PIA
00122 A29E E7 01             STA B   1,X         SEND VALUE
00123 A2A0 F7 A084           STA B   BSAVE
00124 A2A3 CE 000A           LDX     #10
00125 A2A6 BD F021           JSR     DMILLI      DELAY FOR RELAYS
00126 A2A9 39                RTS
00127                *
00128 A2AA FE A0AE BCOMD1    LDX     PIA
00129 A2AD A6 00             LDA A   0,X         READ VALUE RETURNED
00130 A2AF F6 A084           LDA B   BSAVE       GET VALUE SENT
00131 A2B2 B7 8100           STA A   HEXLD       DISPLAY READ VALUE
00132 A2B5 F7 8101           STA B   HEXLD+1     DISPLAY SENT VALUE
00133 A2B8 39                RTS
00134                *
00135      A2B9      BCALL   EQU     *           BUTTON 2
00136 A2B9 CE A366           LDX     #DUM        DEFAULT LOOP
00137 A2BC FF A1F0           STX     VSUB1
00138 A2BF 8D B8             BSR     BCIN        INITIALIZE BINARY CARD
00139 A2C1 C6 00             LDA B   #0          START WITH 0
```

```
PAGE  004   BOX
00140 A2C3 FE A0AE BCALL1 LDX .   PIA       GET PIA ADDRESS
00141 A2C6 E7 01          STA B   1,X       SEND VALUE
00142 A2C8 F7 A084        STA B   BSAVE     SAVE SENT VALUE
00143 A2CB CE 000A        LDX     #10
00144 A2CE BD F021        JSR     DMILLI    DELAY FOR RELAYS
00145 A2D1 FE A0AE        LDX     PIA       GET PIA ADDRESS
00146 A2D4 A6 00          LDA A   0,X       READ RETURN VALUE
00147 A2D6 F6 A084        LDA B   BSAVE     GET VALUE SENT
00148 A2D9 B7 8100        STA A   HEXLD     DISPLAY VALUE READ
00149 A2DC F7 8101        STA B   HEXLD+1   DISPLAY VALUE SENT
00150 A2DF 11             CBA               VALUES THE SAME?
00151 A2E0 26 0B          BNE     BEND      ABORT IF NOT
00152 A2E2 37             PSH B
00153 A2E3 CE 0014        LDX     #20       WAIT FOR DISPLAY TIME
00154 A2E6 BD F021        JSR     DMILLI
00155 A2E9 33             PUL B
00156 A2EA 5C             INC B
00157 A2EB 26 D6          BNE     BCALL1    DO NEXT VALUE UNTIL AT 256
00158 A2ED 39       BEND  RTS
00159                     *
00160       A1E0    VCTP1 EQU     $A1E0
00161       8104    TOGGL EQU     $8104
00162       8102    HEXSW EQU     $8102
00163       8100    HEXLD EQU     $8100
00164       A1F0    VSUB1 EQU     $A1F0
00165       F021    DMILLI EQU    $F021
```

```
PAGE  005   BOX
00167                ********************************************
00168                *
00169                *       ANALOG DATA CARD TEST
00170                *       BUTTONS 3 AND 4
00171                *
00172                *       ANALOG DATA CARD CONNECTED TO
00173                *       SECOND PIA.
00174                *
00175                *       BUTTON 3 - RDAD --
00176                *       ROUTINE INITIALIZES PIA, READS CHANNEL
00177                *       NUMBER (0-15 ENTERED IN 4 MS BITS OF
00178                *       TOGGLE SWITCHES) FROM TOGGLE, READS
00179                *       DATA AT THAT CHANNEL AND DISPLAYS
00180                *       DATA READ IN HEXLED (DECIMAL FORMAT
00181                *       RANGE OF -2048 TO +2047). NEGATIVE
00182                *       NUMBER DISPLAY SCHEME :
00183                *       AXXX = -0XXX
00184                *       BXXX = -1XXX
00185                *       CXXX = -2XXX
00186                *
00187                *       BUTTON 4 -DSPLA --
00188                *       DISPLAYS VALUE READ AT THE NEXT CHANNEL.
00189                *       VALUE DISPLAYED IN HEXLED IN ABOVE
00190                *       FORMAT; CHANNEL DISPLAYED IN LIGHTS
00191                *       ABOVE TOGGLE.  WHEN ALL 16 CHANNELS
00192                *       HAVE BEEN DISPLAYED NO FURTHER READINGS
00193                *       ARE DISPLAYED UNTIL THE CHANNELS ARE READ
00194                *       AGAIN WITH BUTTON 3.
00195                *
00196       A2EE     ADIN    EQU     *           INIT. AD PIA
00197 A2EE FE A0AE           LDX     PIA
00198 A2F1 4F                CLR A
00199 A2F2 A7 0A             STA A   $A,X
00200 A2F4 A7 0B             STA A   $B,X
00201 A2F6 A7 02             STA A   2,X
00202 A2F8 86 0F             LDA A   #$0F
00203 A2FA A7 03             STA A   3,X
00204 A2FC 86 34             LDA A   #$34
00205 A2FE A7 0A             STA A   $A,X
00206 A300 86 2C             LDA A   #$2C
00207 A302 A7 0B             STA A   $B,X
00208 A304 A6 02             LDA A   2,X         CLEAR CA1 FLAG
00209 A306 39                RTS
00210                *
00211       A307     RDAD    EQU     *           READ AD PIA
00212 A307 CE A366           LDX     #DUM        DEFAULT LOOP
00213 A30A FF A1F0           STX     VSUB1
00214 A30D 8D DF             BSR     ADIN        INIT. AD PIA
00215 A30F B6 8104           LDA A   TOGGL
00216 A312 4A                DEC A
00217 A313 84 0F             AND A   #$0F        GET # OF CHANNELS TO READ
00218 A315 4C                INC A
00219 A316 B7 A083           STA A   CNUM        # OF CHANNELS
00220 A319 F6 8104 ADATA5    LDA B   TOGGL       READ CHANNEL NUMBER
00221 A31C 54                LSR B               SHIFT TO LS 4 BITS
00222 A31D 54                LSR B
00223 A31E 54                LSR B
00224 A31F 54                LSR B
```

```
PAGE  006   BOX
00225 A320 F7 A0AB          STA B  CHAN      SAVE CHANNEL NUMBER
00226 A323 CE A088          LDX    #BUFFER   STORAGE LOCATION
00227 A326 BD A382          JSR    ADREAD    READ VALUE
00228 A329 CE A088          LDX    #BUFFER   VALUES STORED HERE
00229 A32C A6 00    ADATA4  LDA A  0,X       GET TWO'S COMP. VALUE
00230 A32E E6 01            LDA B  1,X
00231 A330 08               INX
00232 A331 08               INX
00233 A332 FF A0AC          STX    BUFPTR    SAVE BUFFER SPOT
00234 A335 7F A087          CLR    DNEG
00235 A338 85 08            BIT A  #$08      CHECK FOR NEG. NUMBER
00236 A33A 27 0B            BEQ    ADATA2    POSITIVE
00237 A33C C0 01            SUB B  #$01      UNDO TWO'S COMP.
00238 A33E 82 00            SBC A  #$00
00239 A340 43               COM A
00240 A341 53               COM B
00241 A342 84 0F            AND A  #$0F      USE ONLY 12 BITS
00242 A344 7C A087          INC    DNEG      SIGNAL NEGATIVE NUMBER
00243 A347 BD A4B5 ADATA2   JSR    CVB4D     CONVERT TO DECIMAL
00244 A34A 7D A087          TST    DNEG      NEGATIVE NUMBER?
00245 A34D 27 02            BEQ    ADATA3    NO
00246 A34F 8B A0            ADD A  #$A0      ADJUST FOR DISPLAY
00247 A351 B7 8100 ADATA3   STA A  HEXLD     DISPLAY VALUE READ
00248 A354 F7 8101          STA B  HEXLD+1
00249 A357 B6 A0AB          LDA A  CHAN      DISPLAY CHANNEL
00250 A35A 81 10            CMP A  #$10      ADJUST FOR DISPLAY
00251 A35C 26 01            BNE    ADAT31
00252 A35E 4F               CLR A
00253 A35F B7 8105 ADAT31   STA A  TO6SL+1   DISPLAY CHANNEL
00254 A362 4C               INC A
00255 A363 B7 A0AB          STA A  CHAN      NEXT CHANNEL
00256 A366 39       DUM     RTS
00257                 *
00258 A367 CE A366  DSPLA   LDX    #DUM
00259 A36A FF A1F0          STX    VSUB1
00260 A36D 7A A083          DEC    CNUM      # OF CHANNELS READ
00261 A370 27 09            BEQ    DSPLA1
00262 A372 FE A0AC          LDX    BUFPTR    GET PRESENT POINTER
00263 A375 8C A0AA          CPX    #BUFEND+2  AT END OF VALID DATA?
00264 A378 26 B2            BNE    ADATA4    DISPLAY DATA READ
00265 A37A 39               RTS
00266 A37B CE A0AA  DSPLA1  LDX    #BUFEND+2
00267 A37E FF A0AC          STX    BUFPTR
00268 A381 39               RTS
00269                 *
00270                 *******
00271                 *
00272                 *       A/D READ ROUTINE
00273                 *
00274                 *       TO ENTER :
00275                 *       ACC A : NUMBER OF CHANNELS TO READ
00276                 *               (1-16)
00277                 *       ACC B : STARTING CHANNEL NUMBER (0-15)
00278                 *       X REG : ADDRESS OF STORAGE BUFFER FOR
00279                 *               VALUES READ.
00280                 *
00281                 *       AT EXIT :
00282                 *       VALUES READ SAVED IN BUFFER POINTED TO BY X.
```

```
PAGE  007   BOX
00283                 *
00284       F024      DMICRO EQU    $F024
00285                 *
00286 A382 FF A0A8 ADREAD STX     STRPTR   STORAGE POINTER
00287 A385 4A            DEC A             ADJUST NUMBER
00288 A386 B7 A0AA       STA A  NUMBR      SAVE NUMBER TO READ
00289 A389 FE A0AE       LDX    PIA        PIA ADDRESS
00290 A38C 86 2C         LDA A  #$2C       SET FOR CHANNEL SELECT
00291 A38E A7 0B         STA A  $B,X
00292 A390 E7 03         STA B  3,X        SELECT INITIAL CHANNEL
00293 A392 7D A0AA       TST    NUMBR      HOW MANY CHANNELS TO READ
00294 A395 27 04         BEQ    START      ONLY ONE
00295 A397 86 3C         LDA A  #$3C       SET FOR SEQUENTIAL TRIGGER
00296 A399 A7 0A         STA A  $A,X
00297 A39B FE A0AE START  LDX    PIA        GET PIA ADDRESS
00298 A39E A6 0A   START1 LDA A  $A,X
00299 A3A0 2A FC         BPL    START1
00300 A3A2 E6 03         LDA B  3,X        READ LS BYTE
00301 A3A4 A6 02         LDA A  2,X        GET MS BYTE
00302 A3A6 C4 F0         AND B  #$F0       ONLY 4 DATA BITS
00303 A3A8 E7 03         STA B  3,X        STROBE FOR SEQUENTIAL
00304 A3AA 44            LSR A             SHIFT TO LOWER 12 BITS
00305 A3AB 56            ROR B
00306 A3AC 44            LSR A
00307 A3AD 56            ROR B
00308 A3AE 44            LSR A
00309 A3AF 56            ROR B
00310 A3B0 44            LSR A
00311 A3B1 56            ROR B
00312 A3B2 FE A0A8       LDX    STRPTR     STORE POINTER
00313 A3B5 A7 00         STA A  0,X        STORE VALUES
00314 A3B7 E7 01         STA B  1,X
00315 A3B9 08            INX
00316 A3BA 08            INX
00317 A3BB FF A0A8       STX    STRPTR     ADVANCE POINTER
00318 A3BE 7A A0AA       DEC    NUMBR      ALL CHANNELS READ?
00319 A3C1 2A D8         BPL    START      NO, GO READ
00320 A3C3 39            RTS
```

```
PAGE  008   BOX
00322                ****************************************
00323                *
00324                *       ANALOG COMMAND CARD TEST
00325                *       BUTTONS 6 AND 7
00326                *
00327                *       ANALOG COMMAND CARD CONNECTED TO THE
00328                *       FIRST PIA.
00329                *
00330                *       BUTTON 6 - ACONE --
00331                *       ROUTINE READS DATA TO BE SENT FROM
00332                *       HEXSWITCH (DECIMAL FORMAT RANGE
00333                *       -2048 TO +2047.  NEGATIVE NUMBERS
00334                *       REPRESENTED BY THIS SCHEME :
00335                *       AXXX = -0XXX
00336                *       BXXX = -1XXX
00337                *       CXXX = -2XXX
00338                *       CHANNEL TO SEND DATA TO REPRESENTED
00339                *       IN LS 4 BITS OF TOGGLE SWITCHES
00340                *       (TOGGLE 0 = 0, TOGGLE 1 = 1, ETC)
00341                *       WILL READ AD CHANNEL DIALED IN
00342                *       MS BYTE OF TOGGLE AND DISPLAY
00343                *       THAT VALUE ON THE HEXLED IF THE RTS
00344                *       AT ACOMD6 IS CHANGED TO A NOP.
00345                *
00346                *       BUTTON 7 - ACALL --
00347                *       THIS IS A DYNAMIC TEST OF THE
00348                *       ANALOG COMMAND AND ANALOG DATA
00349                *       CARDS. IT REQUIRES THAT BOTH CARDS ARE
00350                *       PRESENT. VALUES ARE SENT FROM NEGATIVE
00351                *       FULL SCALE TO POSITIVE FULL SCALE
00352                *       WITH A DELAY BETWEEN NEW VALUES.
00353                *       FOR DISPLAY PURPOSES.
00354 A3C4 FE A0AE ACIN    LDX     PIA       GET PIA ADDRESS
00355 A3C7 4F              CLR A
00356 A3C8 A7 08           STA A   8,X       INIT. PIA
00357 A3CA A7 09           STA A   9,X
00358 A3CC 4A              DEC A
00359 A3CD A7 00           STA A   0,X
00360 A3CF A7 01           STA A   1,X
00361 A3D1 86 04           LDA A   #$04
00362 A3D3 A7 08           STA A   8,X
00363 A3D5 86 2C           LDA A   #$2C
00364 A3D7 A7 09           STA A   9,X
00365 A3D9 39              RTS
00366                *
00367 A3DA 8D E8    ACONE  BSR     ACIN      INIT. AC CARD
00368 A3DC 7F A086  ACOMD  CLR     ANEG      INDICATES NEGATIVE NUMBER
00369 A3DF B6 8102         LDA A   HEXSW     READ DATA FROM HEX SWITCH
00370 A3E2 2A 05           BPL     ACOMD1    POSITIVE NUMBER
00371 A3E4 7C A086         INC     ANEG      NEGATIVE NUMBER
00372 A3E7 80 A0           SUB A   #$A0      ADJUST DISPLAY VALUE
00373 A3E9 F6 8103 ACOMD1  LDA B   HEXSW+1   READ LS BYTE
00374 A3EC BD A471         JSR     CV4DB     CONVERT TO BINARY
00375 A3EF 85 F0           BIT A   #$F0      WITHIN RANGE
00376 A3F1 27 02           BEQ     ACOMD2    YES
00377 A3F3 4F              CLR A             NO, SET TO ZERO
00378 A3F4 5F              CLR B
00379 A3F5 7D A086 ACOMD2  TST     ANEG      NEGATIVE NUMBER?
```

```
PAGE   009    BOX
00380 A3F8 27 08          BEQ     ACOMD3   NO
00381 A3FA 43             COM A            TAKE TWO'S COMPLEMENT
00382 A3FB 53             COM B
00383 A3FC CB 01          ADD B   #$01
00384 A3FE 89 00          ADC A   #0
00385 A400 84 0F          AND A   #$0F     ONLY USE 12 BITS
00386 A402 B7 A085 ACOMD3 STA A   ASAVE    SAVE VALUE
00387 A405 B6 8104        LDA A   TOGGL    GET CHANNEL NUMBER(S)
00388 A408 48       ACOMD5 ASL A           MOVE TO MS BITS
00389 A409 48             ASL A
00390 A40A 48             ASL A
00391 A40B 48             ASL A
00392 A40C BA A085        ORA A   ASAVE    CHANNEL AND VOLTAGE
00393 A40F FE A0AE        LDX     PIA      GET PIA ADDRESS
00394 A412 A7 00          STA A   0,X      SEND VALUE
00395 A414 E7 01          STA B   1,X
00396 A416 39       ACOMD6 RTS             REMOVE IF CONNECTED TO AD
00397 A417 BD A2EE        JSR     ADIN     INIT. AD PIA
00398 A41A CE A421        LDX     #READ
00399 A41D FF A1F0        STX     VSUB1
00400 A420 39             RTS
00401 A421 86 01    READ  LDA A   #1       AD CHANNELS TO READ
00402 A423 7E A319        JMP     ADATA5   READ AND DISPLAY
00403                     *
00404      A426     ACALL EQU     *        DYNAMIC AC/AD TEST
00405 A426 BD A3C4        JSR     ACIN     INIT. ACOMD PIA
00406 A429 7F A082        CLR     FLAG
00407 A42C CE 0800        LDX     #$0800   MOST NEGATIVE NUMBER
00408 A42F FF A080        STX     COMD
00409 A432 B6 A080 ACALL1 LDA A   COMD     COMMAND
00410 A435 84 0F          AND A   #$0F     ONLY 12 DATA BITS
00411 A437 F6 A081        LDA B   COMD+1
00412 A43A B7 A085        STA A   ASAVE
00413 A43D B6 8104        LDA A   TOGGL    CHANNELS TO SEND TO
00414 A440 48             ASL A
00415 A441 48             ASL A
00416 A442 48             ASL A
00417 A443 48             ASL A
00418 A444 BA A085        ORA A   ASAVE    MERGE CHANNEL AND DATA
00419 A447 FE A0AE        LDX     PIA      PIA ADDRESS
00420 A44A A7 00          STA A   0,X      SEND VALUE
00421 A44C E7 01          STA B   1,X
00422 A44E BD A2EE        JSR     ADIN     INIT. A/D PIA
00423 A451 CE A470        LDX     #ACALL3  DEFAULT LOOP SUBROUTINE
00424 A454 FF A1F0        STX     VSUB1
00425 A457 86 01          LDA A   #1       READ ONE A/D CHANNEL
00426 A459 BD A319        JSR     ADATA5   AND DISPLAY
00427 A45C FE A080 ACALL2 LDX     COMD     CURRENT COMMAND
00428 A45F 8C 1800        CPX     #$1800   AT PLUS FULL SCALE?
00429 A462 27 0C          BEQ     ACALL3   YES, RETURN
00430 A464 08             INX              GET NEXT COMMAND
00431 A465 FF A080        STX     COMD
00432 A468 CE 0100        LDX     #$0100
00433 A46B BD F021        JSR     DMILLI   DELAY FOR DISPLAY
00434 A46E 20 C2          BRA     ACALL1   SEND THIS COMMAND
00435 A470 39       ACALL3 RTS
00436      FC0F     OUT4HS EQU     $FC0F
00437      FC1E     PCRLF  EQU     $FC1E
```

```
PAGE  010    BOX
00439                ********************************************
00440                *
00441                *       CONVERT 4 DIGIT DECIMAL TO BINARY
00442                *
00443                *       12 BIT BINARY--11 BITS+SIGN
00444                *
00445                *       TO ENTER:
00446                *       A = 2 MS DECIMAL DIGITS
00447                *       B = 2 LS DECIMAL DIGITS
00448                *       AT EXIT:
00449                *       A = MS BYTE OF CONVERTED NUMBER
00450                *       B = LS BYTE OF CONVERTED NUMBER
00451                *
00452 A471 7F A0B6 CV4DB   CLR     BINUPR
00453 A474 B7 A0B5         STA A  SAVEA     SAVE MS DIGITS
00454 A477 17              TBA
00455 A478 C4 0F           AND B  #$0F      SAVE ONLY ONES VALUE
00456 A47A 44              LSR A            MOVE TENS VALUE
00457 A47B 44              LSR A            TO LOWER BITS OF A
00458 A47C 44              LSR A
00459 A47D 44              LSR A
00460 A47E 27 05    TEN    BEQ     DOHUND   DO HUND WHEN TENS=0
00461 A480 CB 0A           ADD B  #10       ADD 10 TO TOTAL
00462 A482 4A              DEC A            DECREMENT 10S DIGIT
00463 A483 20 F9           BRA     TEN
00464 A485 0C       DOHUND CLC              RESET CARRY
00465 A486 B6 A0B5         LDA A  SAVEA     GET HUND AND THOU DIGIT
00466 A489 84 0F           AND A  #$0F      SAVE 100S DIGITS
00467 A48B 27 0A    HUN1   BEQ     DOTHOU   DO 1000S WHEN 100S=0
00468 A48D CB 64           ADD B  #100      ADD 100 TO TOTAL
00469 A48F 24 03           BCC     HUN2
00470 A491 7C A0B6         INC     BINUPR   ADD 256 TO TOTAL
00471 A494 4A       HUN2   DEC A            DECREMENT 100S DIGIT
00472 A495 20 F4           BRA     HUN1
00473 A497 B6 A0B5 DOTHOU  LDA A  SAVEA     GET 100S AND 1000S DIGITS
00474 A49A 44              LSR A            MOVE 1000S TO LOWER
00475 A49B 44              LSR A            4 BITS
00476 A49C 44              LSR A
00477 A49D 44              LSR A
00478 A49E B7 A0B5         STA A  SAVEA
00479 A4A1 27 0E           BEQ     THOU1    1000S=0?
00480 A4A3 B6 A0B6         LDA A  BINUPR
00481 A4A6 0C       THOU   CLC              RESET CARRY
00482 A4A7 CB E8           ADD B  #$E8      ADD 1000 TO TOTAL
00483 A4A9 89 03           ADC A  #$03
00484 A4AB 7A A0B5         DEC     SAVEA    DECREMENT 1000S DIGIT
00485 A4AE 26 F6           BNE     THOU
00486 A4B0 39              RTS
00487 A4B1 B6 A0B6 THOU1   LDA A  BINUPR
00488 A4B4 39              RTS
00489                *
```

```
PAGE  011   BOX
00491                 ********************************************
00492                 *
00493                 *       CONVERT BINARY TO 4 DIGIT DECIMAL
00494                 *
00495                 *       12 BIT BINARY--11 BITS+SIGN
00496                 *
00497                 *       TO ENTER:
00498                 *         A = MS BYTE OF NUMBER TO CONVERT
00499                 *         B = LS BYTE OF NUMBER TO CONVERT
00500                 *       AT EXIT:
00501                 *         A = 2 MS DECIMAL DIGITS
00502                 *         B = 2 LS DECIMAL DIGITS
00503                 *
00504 A4B5 CE A511 CVB4D  LDX    #CONST    POINT TO 1000
00505 A4B8 B7 A0B0       STA A  NUM       SAVE BINARY DIGITS
00506 A4BB F7 A0B1       STA B  NUM+1
00507 A4BE 7F A0B3       CLR    DIGIT+1
00508 A4C1 7F A0B4       CLR    DFLAG     ADD TO UPPER DIGIT
00509 A4C4 8D 1E         BSR    SUBT      DETERMINE MS (4TH) DIGIT
00510 A4C6 08            INX              POINT TO 100
00511 A4C7 08            INX
00512 A4C8 7C A0B4       INC    DFLAG     ADD TO LOWER DIGIT
00513 A4CB 8D 17         BSR    SUBT      DETERMINE 3RD DIGIT
00514 A4CD B6 A0B3       LDA A  DIGIT+1
00515 A4D0 B7 A0B2       STA A  DIGIT
00516 A4D3 08            INX              POINT TO 10
00517 A4D4 08            INX
00518 A4D5 7F A0B3       CLR    DIGIT+1
00519 A4D8 7F A0B4       CLR    DFLAG     ADD TO UPPER DIGIT
00520 A4DB 8D 07         BSR    SUBT      DETERMINE 2ND DIGIT
00521 A4DD FB A0B3       ADD B  DIGIT+1   DETERMINE 1ST DIGIT
00522 A4E0 B6 A0B2       LDA A  DIGIT
00523 A4E3 39            RTS
00524                 *
00525 A4E4 B6 A0B0 SUBT  LDA A  NUM       SUBTRACT CONSTANTS
00526 A4E7 F6 A0B1       LDA B  NUM+1     GET BINARY DIGITS
00527 A4EA E0 01   SUB1  SUB B  1,X       SUBTRACT CONSTANTS
00528 A4EC A2 00         SBC A  0,X
00529 A4EE 25 16         BCS    SUB3      TOO MANY SUBTRACTIONS
00530 A4F0 7D A0B4       TST    DFLAG     ADD TO WHICH DIGIT
00531 A4F3 27 05         BEQ    SUB2
00532 A4F5 7C A0B3       INC    DIGIT+1   ADD TO LOWER DIGIT
00533 A4F8 20 F0         BRA    SUB1
00534 A4FA 36       SUB2 PSH A
00535 A4FB 86 10         LDA A  #$10      ADD TO UPPER DIGIT
00536 A4FD BB A0B3       ADD A  DIGIT+1
00537 A500 B7 A0B3       STA A  DIGIT+1
00538 A503 32            PUL A
00539 A504 20 E4         BRA    SUB1
00540 A506 EB 01   SUB3  ADD B  1,X       UNDO LAST SUBTRACTION
00541 A508 A9 00         ADC A  0,X
00542 A50A B7 A0B0       STA A  NUM       SAVE BINARY DIGITS
00543 A50D F7 A0B1       STA B  NUM+1
00544 A510 39            RTS
00545                 *
00546 A511 03E8   CONST  FDB    1000
00547 A513 0064          FDB    100
00548 A515 000A          FDB    10
```

```
PAGE  012   BOX
00550 A080                    ORG     $A080
00551 A080 0002     COMD      RMB     2          COMMAND FOR DYNAMIC TEST
00552 A082 0001     FLAG      RMB     1          ADD OR SUBTRACT
00553 A083 0001     CNUM      RMB     1          # OF AD CHANNELS TO READ
00554 A084 0001     BSAVE     RMB     1          VALUE SENT TO BINARY COMMAND
00555 A085 0001     ASAVE     RMB     1          TEMP. STORAGE
00556 A086 0001     ANEG      RMB     1          NEGATIVE ANALOG COMMAND
00557 A087 0001     DNEG      RMB     1          NEGATIVE AD READING
00558 A088 0020     BUFFER    RMB     32         STORAGE FOR AD READINGS
00559      A0A8     BUFEND    EQU     *
00560 A0A8 0002     STRPTR    RMB     2          POINTER TO STORAGE BUFFER
00561 A0AA 0001     NUMBR     RMB     1          NUMBER OF AD CHANNELS TO REA
00562 A0AB 0001     CHAN      RMB     1          AD CHANNEL
00563 A0AC 0002     BUFPTR    RMB     2          POINTER TO AD STORAGE BUFFER
00564 A0AE 0002     PIA       RMB     2          PIA ADDRESS
00565               * :       CONVERT ROUTINE SCRATCH RAM
00566 A0B0 0002     NUM       RMB     2
00567 A0B2 0002     DIGIT     RMB     2
00568 A0B4 0001     DFLAG     RMB     1
00569 A0B5 0001     SAVEA     RMB     1
00570 A0B6 0001     BINUPP    RMB     1
00571               *
00572                         END

TOTAL ERRORS 00000

!
```

points on the test box front panel. The analog command voltage can be read by the analog data board by changing an RTS software instruction to an NOP at the statement labeled ACOMD6. This test is made by dialing "command" on the switch on the upper right corner of the test box. The test box is designed so that the voltage sent by channel 0 of the analog command board can be read from channels 0, 4, 8, and 12 on the analog data board, and channel 1, from 1, 5, 9, 13, etc. The routine reads the analog data channel selected by the MS bit of the toggle switches and displays the value read on the MPC hex display. If proper channels are selected, the data on the hex display should match that on the hex switch.

Button 7 is a dynamic test of the analog command and data boards. The analog command board must be connected to the first PIA and the analog data board must be connected to the second PIA. The program sends all values from negative full scale to positive full scale to the analog command board; these values are then read by the analog data board and displayed on the hex display.

Button 8 assumes the data and command boards are connected to the PIA board in slot 7 of the MPC. If another slot is used, dial the PIA board first word address in the hex switch and push button 8. Any of the tests for buttons 1-7 can then be executed.

ACKNOWLEDGMENTS

The authors acknowledge helpful discussions with D. A. Swenson and J. E. Stovall about the requirements of the data system, and the drafting efforts of Felix Martinez.

REFERENCES

1. R. W. Goodwin, R. F. Kocanda, and M. F. Shea, "A Method for Implementing Microprocessor Controlled Systems," IEEE Trans. Nucl. Sci. **NS-23**, No. 1, 297-300 (1976).

2. J. K. Halbig, S. F. Klosterbuer, and D. A. Swenson, "A General-Purpose Microprocessor-Based Control System," Los Alamos Scientific Laboratory report LA-7896.

3. J. E. Swansen, private communication, Los Alamos Scientific Laboratory, 1979.

4. National Semiconductor Specification sheet on LF198/LF298/LF398 monolithic sample and hold circuits.

☆ U.S. Government Printing Office: 1980 — 677 — 115/8