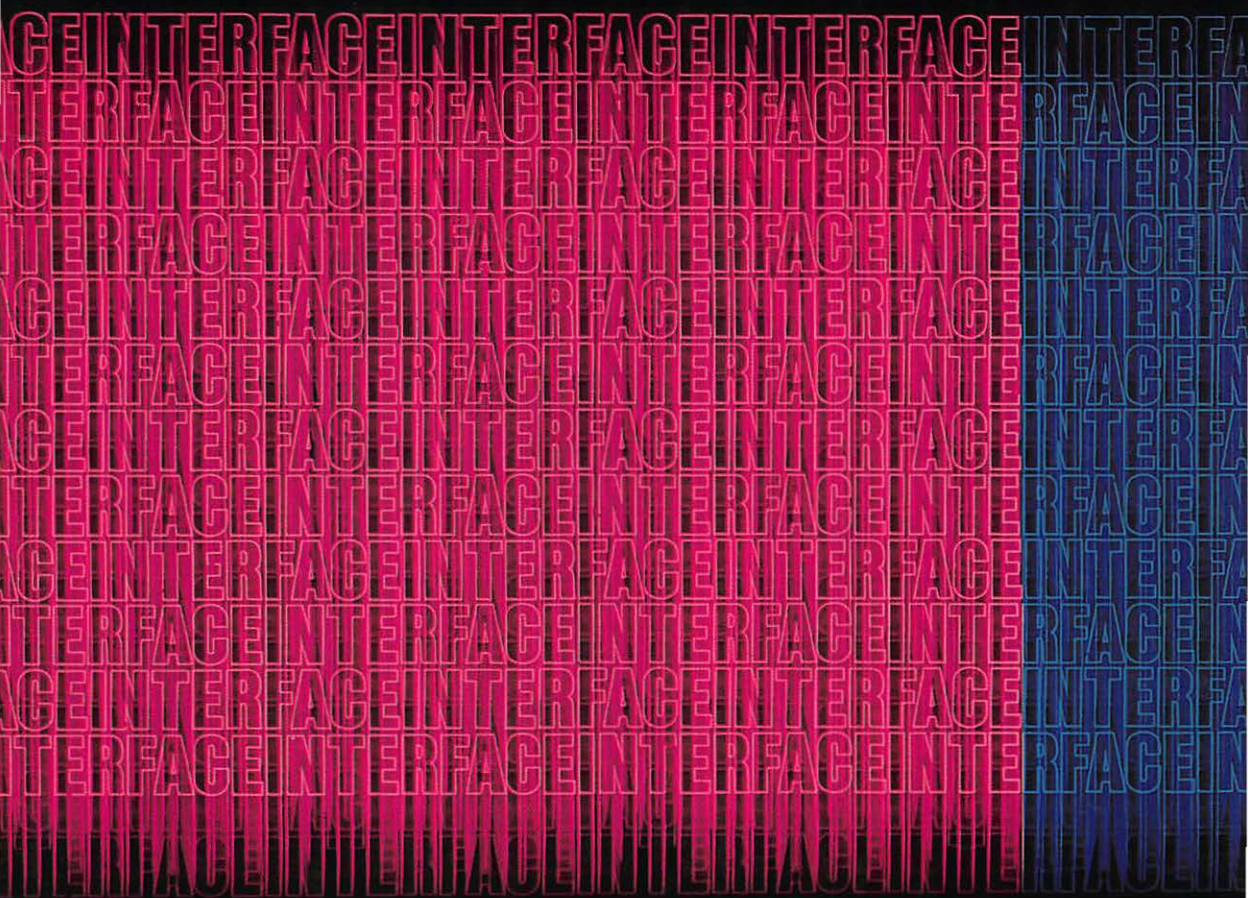


RICHARD C. HALLGREN

interface projects for the apple II



RICHARD C. HALLGREN

INTERFACE PROJECTS FOR THE APPLE II



Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632

Library of Congress Cataloging in Publication Data

Hallgren, Richard C.

Interface projects for the Apple II.

"A Spectrum Book."

Includes index.

1. Computer interfaces. 2. Apple II (Computer)

I. Title.

TK7887.5.H34 621.3819 592 82-3748

ISBN 0-13-469395-7 AACR2

ISBN 0-13-469387-6 (pbk.)

This Spectrum Book is available to business and organizations at a special discount when ordered in large quantities. For information, contact Prentice-Hall, Inc., General Publishing Division, Special Sales, Englewood Cliffs, N.J. 07632.

© 1982 by Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.
All rights reserved. No part of this book may be reproduced in any form or by any means without permission in writing from the publisher.
A SPECTRUM BOOK. Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

Editorial/production supervision by Kimberly Mazur
Manufacturing buyer: Barbara A. Frick

ISBN 0-13-469395-7

ISBN 0-13-469387-6 {PBK.}

Prentice-Hall International, Inc., London

Prentice-Hall of Australia Pty. Limited, Sydney

Prentice-Hall Canada Inc., Toronto

Prentice-Hall of India Private Limited, New Delhi

Prentice-Hall of Japan, Inc., Tokyo

Prentice-Hall of Southeast Asia Pte. Ltd., Singapore

Whitehall Books Limited, Wellington, New Zealand

CONTENTS

1	INTRODUCTION	1
2	REVIEW OF DATA TRANSFER FORMATS	3
3	SAMPLING THE EXTERNAL WORLD	21
4	DIGITAL TO ANALOG CONVERSION	70
5	UTILIZATION OF THE APPLE TWO IN SERIAL APPLICATIONS	78
6	BIOFEEDBACK	101
7	CONTROLLING A VIDEO PLAYBACK DEVICE	117
8	DATA ANALYSIS OF SAMPLED SIGNALS	141
	APPENDIX A: CONSTRUCTION TECHNIQUES	152
	APPENDIX B: OPERATIONAL AMPLIFIER THEORY	158
	APPENDIX C: POWER SUPPLIES	168
	INDEX	171

Richard C. Hallgren, Ph.D., is an assistant professor at Michigan State University. He has written numerous articles and books, including **INTERFACE PROJECTS FOR THE TRS-80**, published by Prentice-Hall.

1

INTRODUCTION

The rapid expansion of the electronics industry and their ability to mass produce reliable, large-scale integrated circuits has led to the reality of scientific calculators and multifunction digital watches for under \$20 and "personal computers" starting from \$200. In 1971 the Intel Corporation had the honor of initiating this revolution by introducing the first commercially available microprocessor. While the performance of this device was limited by its relatively slow central processing unit and its use of a 4-bit data bus, its acceptance was so dramatic that within a two-year period the cost of a single unit had dropped from \$200 to under \$20. Today, it is a rare individual who does not have a microprocessor (if not a personal computer) of some type in his or her home.

As I have talked to owners of personal computers, it has become apparent that there is a certain group of individuals who

would like to use their machines as extensively as is possible, both at home and at their place of employment. This inevitably results in the need for interface circuitry to allow the computer to be connected to external devices allowing users to monitor and control their environment. On the basis of the response of readers to several articles that I have written, I have come to the conclusion that there is a great need for a book that covers not only the theory behind interfacing a computer to external devices but also gives a broad selection of interface circuits that can be built and expected to work. What I have attempted to do is to provide you, the reader, with a number of examples of interface circuits ranging from the very simple to the somewhat complicated. I have included software for each circuit and can assure you that both the hardware and the software have not only been tested together but have been used in some practical area of application. The result is a document that explains the theory behind computer interfacing and gives you the choice of either building the circuit as is or modifying it to meet your specific needs. The hardware for these projects was designed and the components selected so that construction and check-out would be a straightforward matter. The circuits have been chosen to offer something of interest and application to a broad range of individuals: the hobbyist, the experimenter, the manager of a manufacturing plant, and the engineer or scientist in a research facility. The intent being to enable people to more fully use the computational and control capabilities of their personal computer in a practical and interesting way.

The book assumes that readers have a fairly good understanding of the commands in Applesoft BASIC, and have written some of their own programs. Since some of the supporting software will be using 6502 machine code, the reader is encouraged to become familiar with the 6502 instruction set. Programming examples using both Applesoft and 6502 machine language have been provided throughout the book. While construction of the circuits is straightforward, it should be understood that a certain amount of experience, and a certain level of expertise is expected. Previous construction of a commercial kit would probably qualify most individuals. So, with all this in mind, let's get started with the job of connecting your Apple II to the outside world.

2

REVIEW OF DATA TRANSFER FORMATS

BASIC INTERFACE CONCEPTS

Before we get into actual circuit designs, we need to spend some time reviewing basic interface concepts. Even experienced circuit designers should read this section to become familiar with terminology that will be used.

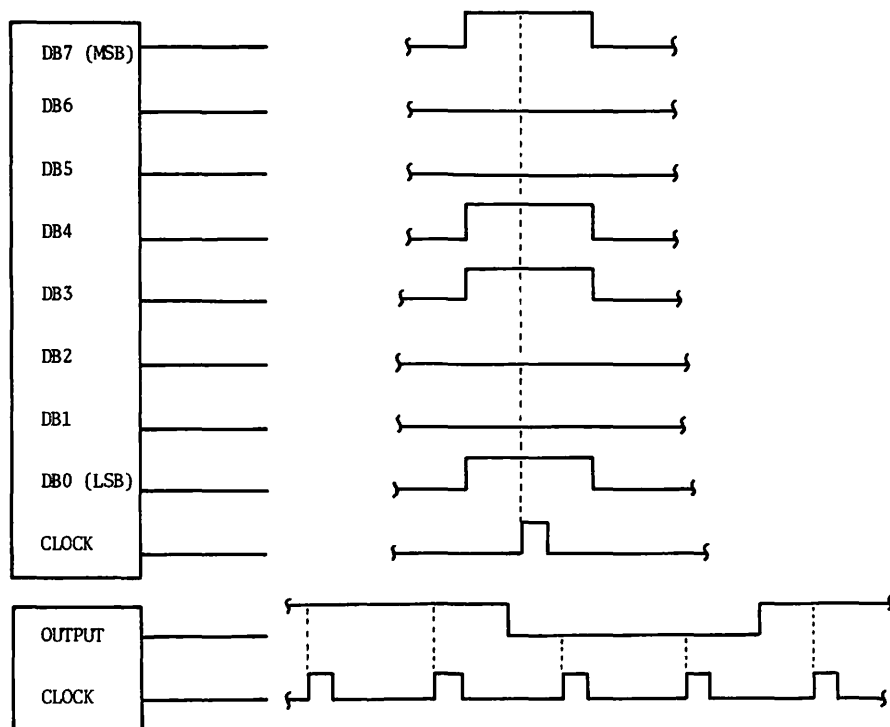
We can define an input/output (I/O) port as a collection of electronic circuits, under the control of the computer, which routes data to and from an external peripheral device. The key words in this definition are *data* and *external*. The route by which the data flow to or from the external device is called the *port*. A line printer is an example of an external device; the computer sends characters to be printed to the printer and, in some cases, the printer sends control signals back to the computer. The pur-

pose of these signals is to regulate the flow of the data. This interaction is called *handshaking*.

Ports can be constructed so that data are handled in either a serial or a parallel format (Figure 2.1).

Parallel transfer of data involves the mass transfer, at a given point in time, of several bits of data, where the number of bits transferred is usually equal to the word size of the computer. For the Apple II this is equal to 8 bits. In general, the number of bits transferred at one time will be equal to the size of the data bus. For example, if we were working with a Z8000, which has a 16-bit data bus, a parallel transfer would involve 16 bits. When high rates of data transfer are desired, such as data transfer between a computer and a floppy disk, the transfer is usually handled with a parallel format. Computers handle parallel communication rela-

FIGURE 2.1 Comparison of parallel and serial format. The clock impulse is used to signal the external device that the signals are stable. For both examples, the data word being transmitted is binary 10011001.



tively easily and, even though multiconductor cables are required, the total expense is not great because the distance between devices is usually quite short.

Serial transfer of data involves transmitting individual bits of data, one bit at a time. Microcomputers do not normally communicate in a serial format, so there is usually not a single machine language command that facilitates this type of operation. Therefore, we need to add either a software subroutine or a new piece of hardware to accomplish a serial transfer of data. Such an addition results in increased system expense, but gives the capability of transferring data over increased distances at relatively high data rates. The Electronics Industry Association (EIA) RS-232C electrical specification for serial transfer of data is the standard for industrial applications. This specification defines the following voltage levels: A logic level 1 is called a *mark* and is considered to be any voltage level more negative than -3 volts. A logic level 0 is called a *space* and is considered to be any voltage level more positive than $+3$ volts. In general, designers use $+12$ and -12 volt levels for the logic 0 and logic 1 states. In addition to specifying voltage levels, the EIA also defines the standard RS-232C connector to be a 25-pin, D subminiature type (commonly referred as a DB-25). The pin assignments and their functions are listed in Table 2.1.

Before we look at specific examples, we should take a look at the architecture of the 6502 to determine how we can communicate in an orderly manner with several peripheral devices that may be connected to the same data bus and to the same address bus. The Apple II computer uses a 6502 type microprocessor to perform the logical, mathematical, and decision-making operations necessary for high-speed operation of the computer. This microprocessor is an 8-bit device which combines the bus structure of the 6501 microprocessor (16-bit address bus, 8-bit bidirectional data bus, and two interrupts) with an instruction set that includes over 150 different commands. The bidirectional data bus is used to transfer information both to and from the central processing unit. The 16-bit address bus has the capability of uniquely defining 65,536 addressable locations. The 6502 generates several control signals which are used both internally and externally to supervise and manage the flow of information. Since data can only flow in one direction at a time, and since the data are usually

TABLE 2.1 EIA standards for DB-25 connector pin assignments when used for communication between RS-232C systems.

Pin 1	PGND - Protective Ground This is chassis or equipment ground. It may also be tied to signal ground.
Pin 2	TD - Transmit Data This is the serial data from the terminal to the remote receiving equipment. When no data is being sent it is in a marking (1) condition.
Pin 3	RD - Receive Data This is the serial data from the remote equipment which is transmitted to the terminal.
Pin 4	RTS - Request to Send Controls the direction of data transmission. In full-duplex operation an "on" sets transmit mode and an "off" sets non-transmit mode. In half-duplex operation an "on" inhibits the receive mode and an "off" enables it.
Pin 5	CTS - Clear to Send Signal from the modem to the terminal indicating ability to transmit data. An "on" is "Ready" and an "off" is "not ready."
Pin 6	DSR - Data Set Ready Signal from the modem to the terminal. An "on" condition indicates that the modem is ready.
Pin 7	SGND - Signal Ground
Pin 8	CD - Carrier Detect An "on" indicates reception of a carrier from the remote data set; "off" indicates no carrier is being received.
Pin 20	DTR - Data Terminal Ready "On" connects the communication equipment to the communications channel; "off" disconnects the communications equipment from the communications channel.
Pin 22	RI - Ring Indicator An "on" indicates that a ringing signal is being received on the communications channel.

directed to a specific device or memory location, these control lines provide an essential coordinating function.

Figure 2.2 shows the pin configuration for the 6502. The 6502 belongs to the one-address architectural class of microprocessors; that is, all memory-related instructions refer to a single memory location. The direction of all data transfers is controlled by the READ/WRITE* (R/W*) line. When this line goes low, all data transfers will occur in the direction from the processor to memory. When it goes high, all data transfers will occur in the direction from memory to the processor. The timing of all data transfers is controlled by the system clock. The clock system used by the 6502 is referred to as *two-phase*; it is most easily thought of as two nonoverlapping square waves. When the microprocessor is reading a data byte from memory, the address of the desired memory location is placed on the address bus during the phase one clock pulse. Information stored in that particular memory location is strobed onto the data bus and into the accumulator

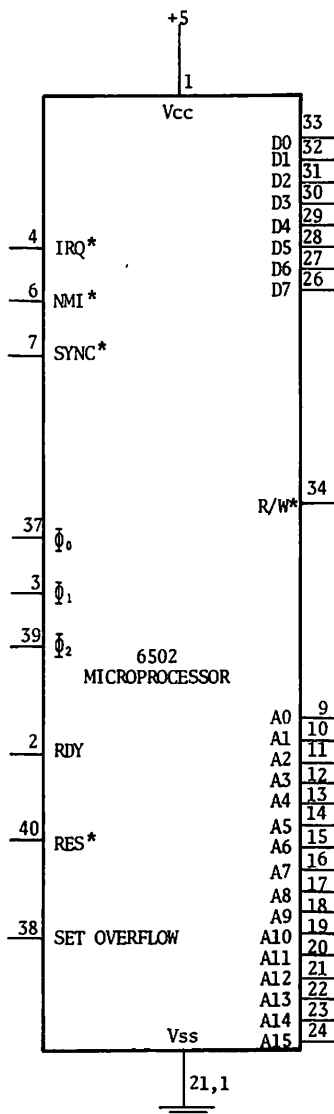


FIGURE 2.2 6502 pin configuration.

during the phase two clock pulse. Data being written to memory is handled by a reversed set of operations.

Fortunately for the individual who wishes to interface the Apple II to external devices, the designer of the computer has provided eight peripheral-board connectors on the mother board. Having these connectors on the mother board allows you to place

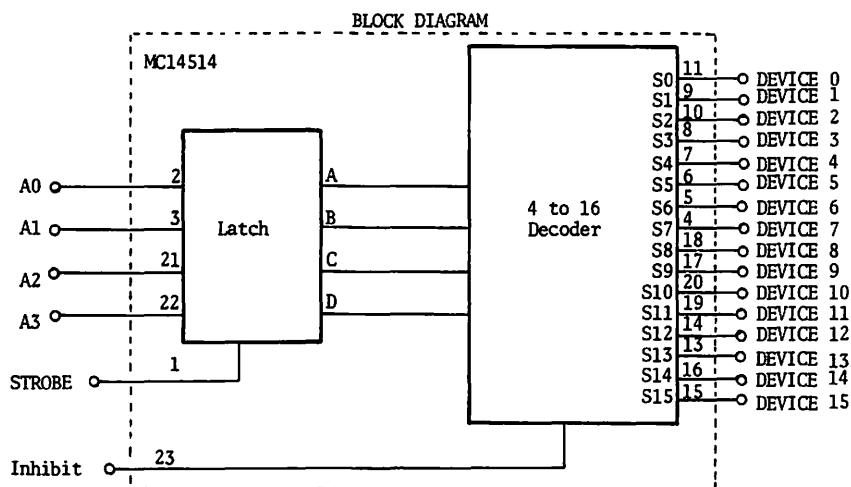
TABLE 2.2 Peripheral I/O location address assignments.

Hex Address	Assigned Function	Comments
C080-C08F	Device Select #0	Pin #41 (DS*) on the selected peripheral connector goes low during phase two of the clock.
C090-C09F	Device Select #1	
C0A0-C0AF	Device Select #2	
C0B0-C0BF	Device Select #3	
C0C0-C0CF	Device Select #4	
C0D0-C0DF	Device Select #5	
C0E0-C0EF	Device Select #6	
C0F0-C0FF	Device Select #7	

any interface boards that you construct inside the computer's case, and permits you to use the Apple's power supply. The peripheral-board connectors give the hardware designer access to all address, data, and control lines. In addition, certain control and address lines have been decoded to provide the designer with a device select (DS*) signal. What this means is that each peripheral connector has a unique range of address locations that will result in a low output on the DS* line. Since DS* goes low during the phase two clock pulse, it is convenient to use this as a data strobe pulse during either a READ or WRITE operation. Table 2.2 shows the address locations that are assigned to each of the eight peripheral connectors.

Since each peripheral connector can address up to sixteen devices, and since we will not exhaust this capability on any one of the projects that you will be building, I will limit my comments here to one peripheral connector, specifically peripheral I/O location 7. In Chapter 5, however, I will design the digital plotter interface so that it resides in I/O location 5. I will do this so that the high speed A/D converter and the digital plotter can be used to digitize and plot data without having to switch the physical location of the circuit boards. Figure 2.3 shows a simple but effective method for decoding the four least significant bits of the address bus into sixteen unique control lines. For normal operation, the INHIBIT control line would remain low. Upon execution of either a READ or WRITE command, the device control signal (DS*), connected to the STROBE control line, would be used to strobe the current contents of the address bus into the decoder. The decoded output would then be used to indicate to a specific external circuit that the data bus was available for its specific use. In the next section we will see how we can use the R/W* control line to direct the flow of data to and from our interface projects.

FIGURE 2.3 4-16 line address decoder. Each device is selected by a unique combination of the address lines A0-A3.



DECODE TRUTH TABLE (Strobe = 1)

INHIBIT	DATA INPUTS				SELECTED OUTPUT
	A3	A2	A1	A0	LOGIC "1"
0	0	0	0	0	Device 0
0	0	0	0	1	Device 1
0	0	0	1	0	Device 2
0	0	0	1	1	Device 3
0	0	1	0	0	Device 4
0	0	1	0	1	Device 5
0	0	1	1	0	Device 6
0	0	1	1	1	Device 7
0	1	0	0	0	Device 8
0	1	0	0	1	Device 9
0	1	0	1	0	Device 10
0	1	0	1	1	Device 11
0	1	1	0	0	Device 12
0	1	1	0	1	Device 13
0	1	1	1	0	Device 14
0	1	1	1	1	Device 15
1	X	X	X	X	All Outputs = 0

X = Don't Care

PARALLEL DATA FORMAT

Transmission and reception of data in a parallel format combines the advantages of high transfer rates and low cost. As mentioned earlier, parallel data transfer involves the mass movement of several bits of data at one time. It becomes a relatively straightforward task to construct both input and output ports since all the control, data, and address lines that we will need are available on the Apple II peripheral I/O connector (see Figure 2.4). Table 2.3 lists the signals appearing on this connector and their functions. Figures 2.5a and 2.5b show examples of practical circuits which could be connected directly to the I/O connector 7, providing one 8-bit input port and one 8-bit latched output port. To transfer data from the accumulator to the latched output port, you need only to perform the BASIC statement POKE-16143,X or the machine language commands LDA #\$X and STA \$C0F1. The microprocessor performs the following bit transfers when you execute either of these commands:

- 1. The device address (C0F1) is strobed onto the address bus.
- 2. The contents of the accumulator (X) are strobed onto the data bus.

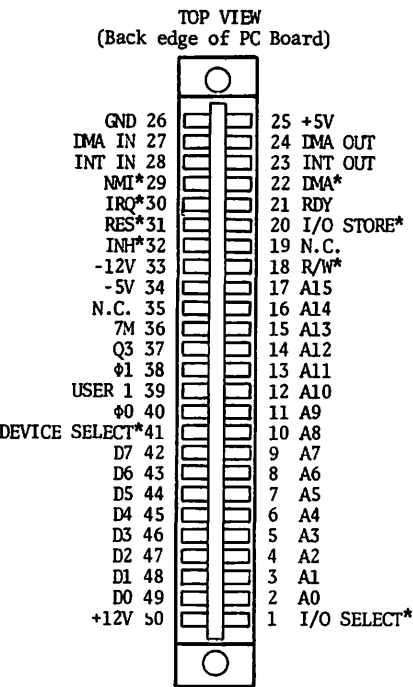


FIGURE 2.4 Output pin configuration for the Apple II peripheral I/O connector. Refer to Table 2.3 for a description of each pin.

TABLE 2.3 Functional description of pins on the Apple II peripheral I/O connector.

P/N	Signal Name	Description
1	I/O SELECT*	Peripheral I/O Select Line
2	A0	Address Output
3	A1	Address Output
4	A2	Address Output
5	A3	Address Output
6	A4	Address Output
7	A5	Address Output
8	A6	Address Output
9	A7	Address Output
10	A8	Address Output
11	A9	Address Output
12	A10	Address Output
13	A11	Address Output
14	A12	Address Output
15	A13	Address Output
16	A14	Address Output
17	A15	Address Output
18	R/W*	System Read/Write Strobe
19	N.C.	No Connection
20	I/O STRBE*	Peripheral I/O Strobe Line
21	RDY	Peripheral Device Ready
22	DMA*	Direct Memory Access Line
23	INT OUT	Interrupt Chain Output
24	DMA OUT	Direct Memory Access Chain Output
25	+5	Positive 5-Volt Supply
26	GND	Power Supply Ground
27	DMA IN	Direct Memory Access Chain Input
28	INT IN	Interrupt Chain Input
29	NMI*	Non Maskable Interrupt
30	IRO*	Interrupt Request Line
31	RES*	Reset Line
32	INH*	Inhibit Line
33	-12	Negative 12-Volt Supply
34	-5	Negative 5-Volt Supply
35	N.C.	No Connection
36	7M	7 MHz Clock
37	Q ₃	1 MHz Timing Signal
38	Φ1	Phase 1 Clock Signal
39	USER 1	Disables Internal I/O Adr Decode
40	Φ0	Phase 0 Clock Signal
41	DS*	Device Select Line
42	D7	Bidirectional Data Bus
43	D6	Bidirectional Data Bus
44	D5	Bidirectional Data Bus
45	D4	Bidirectional Data Bus
46	D3	Bidirectional Data Bus
47	D2	Bidirectional Data Bus
48	D1	Bidirectional Data Bus
49	D0	Bidirectional Data Bus
50	+12	Positive 12-Volt Supply

Note: "*" means logical "0" true input or output.

Since the data are valid for only a few clock cycles (perhaps 500 ns.), a set of clocked flip-flops (IC3 and IC4) are provided so that the data are latched and consequently stable, until the next command to write data to this port is executed. The circuit works in the following manner:

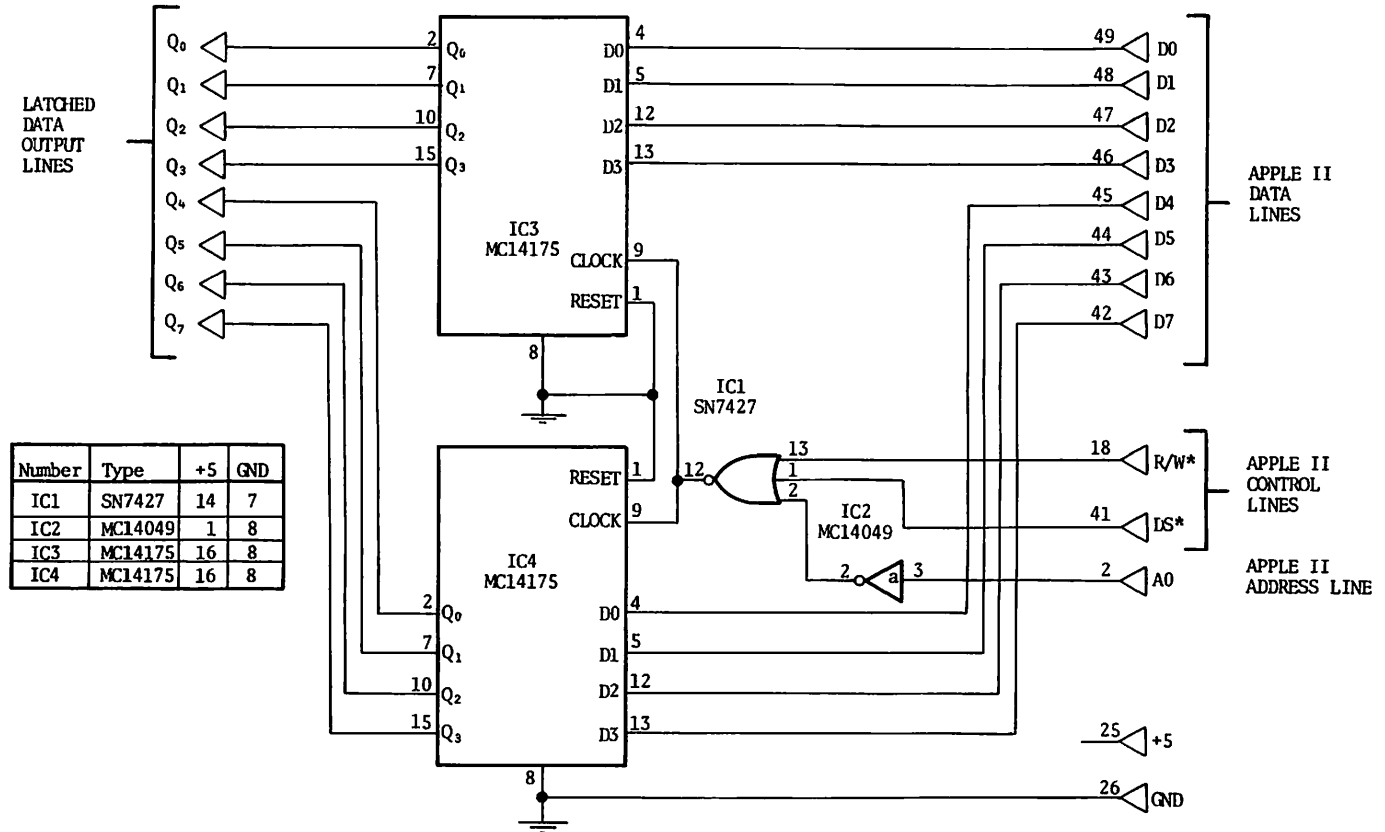


FIGURE 2.5a Circuit diagram of a typical parallel output port. Performing a POKE -16143,16 would cause the DS* and R/W* control lines to go low, address line A0 to go high, and 16₁₀ to be latched into the output register.

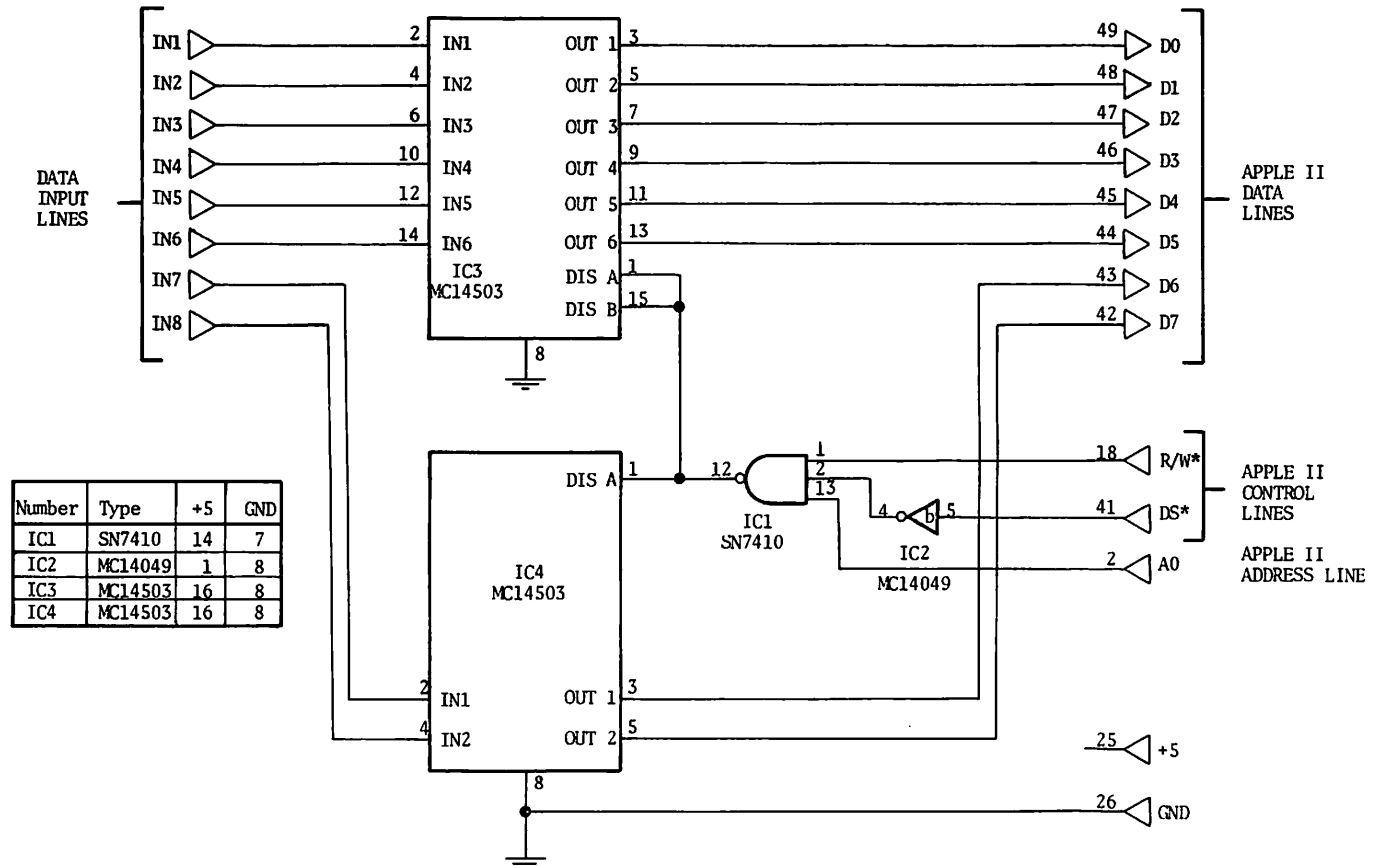


FIGURE 2.5b Circuit diagram of a typical parallel input port. Performing a PEEK (-16143) would cause the DS* control line to go low, the R/W* control line to go high, address line A0 to go high, and the data on the input lines to be transferred into the accumulator.

1. Whenever address line A0 is high and the DS* line and the R/W* line are low, there will appear an output strobe pulse on pin 12 of IC1.
2. Whenever this strobed output pulse appears, the contents of the data bus will be transferred into the clocked flip-flops, and will appear on the latched data output lines.

External devices transferring data into the computer can be connected directly to the data bus, but to add an additional margin of safety (that is, to help keep you from “smoking” your computer), I have included a 3-state buffer. The 3-state buffer is used as a gate which allows signals from the peripheral device to be placed on the data bus at an appropriate time. To transfer data from an external device into the accumulator, you need only to perform the BASIC statement PEEK(-16143), or the assembly language command LDA \$C0F1. The microprocessor performs the following bit transfers when you execute either of these commands:

1. The device address (C0F1) is strobed onto the address bus.
2. The contents of the data bus are strobed into the accumulator (A).

The circuit works in the following manner:

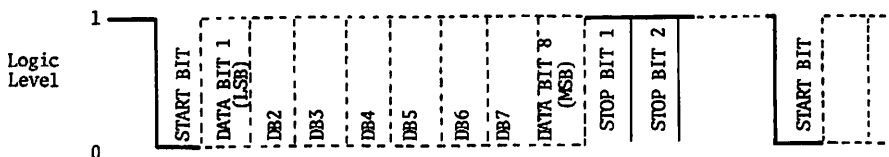
1. Whenever address line A0 is high, control line DS* is low, and control line R/W* is high, there will appear a strobe pulse on pin 12 of IC1.
2. Whenever this strobed input pulse appears, the 3-state buffer will connect the data input lines to the data bus.

SERIAL DATA FORMAT

When considering a computer interface project involving a fairly large distance between the computer and the device to be interfaced, we begin to realize problems in trying to use a parallel data transfer scheme. First, there is the expense and inconvenience of running a multiconductor cable over long distance. There is also the problem of noise which gets introduced into the data, control, and address lines. While this noise can be reduced with filters, or current level signals can be used, it is often more convenient to

use a serial format for the transmission of control and data bits. With serial transmission, individual bits are transmitted over a pair of wires one bit at a time. The receiving device collects the bits and reconstructs them into a single computer word which, for the Apple II, is equal to 8 bits. Since the computer normally processes data in a parallel format, the transmitting device must perform a parallel-to-serial conversion, and the receiving device must perform a serial-to-parallel conversion. Obviously, the transmission speed and the bit format must be well-defined, and there must be some way to synchronize the serial bit flow so that the receiving device knows where each data word begins. The most common format for serial transmission is called *asynchronous-serial*. Because it is self-clocking—that is, it contains a special bit pattern that allows the receiving device to know exactly where a new data word begins—the cable connecting a peripheral device with the computer needs to have only two signal lines and one ground line. Figure 2.6a shows the format for the transmission of a data word in an asynchronous-serial format. The START bit, a logic 1 to logic 0 transition, is used to alert the receiving circuitry that a new data word is going to be sent. The receiver waits for one-half of a bit period and then samples the input again. If the input line is still a logic 0, the receiver assumes that it has detected a “real” START bit and not a fluctuation due to noise on the line. The receiving device then waits one bit period and samples the input line to determine whether data bit 1 is a logic 1 or a logic 0. By periodically waiting one bit period and sampling the input line, the remainder of the data bits are obtained. The two STOP bits then signal the receiving device that it has reached the end of the data word. Having transferred the data word into the computer, the device starts looking for the next START bit. Figure 2.6b shows the actual bit pattern that results when the data word 10011101 is transmitted.

FIGURE 2.6a Typical asynchronous-serial bit pattern for the transmission of an 8-bit data word with 2 stop bits.



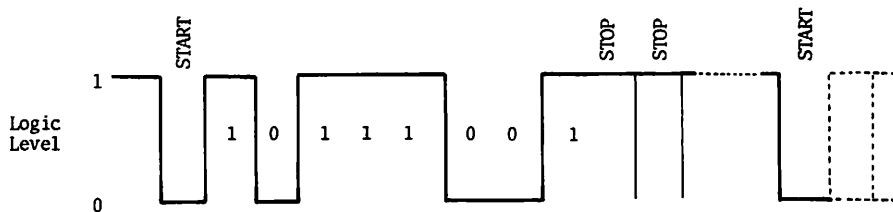
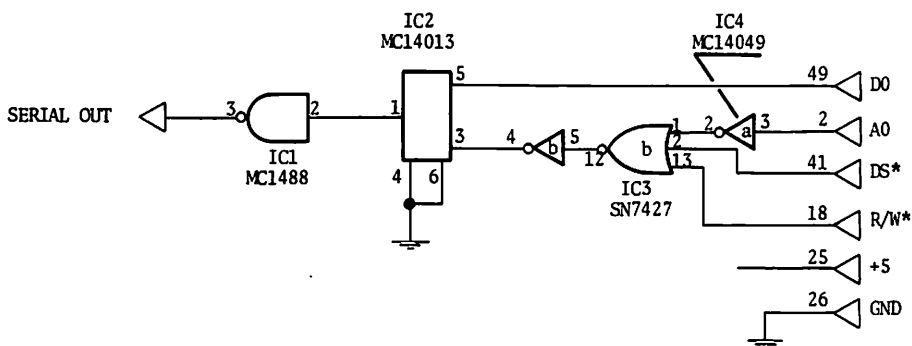


FIGURE 2.6b Logic levels plotted as a function of time during the asynchronous-serial transmission of the 8-bit data word, 10011101_2 .

The asynchronous-serial protocol allows the use of either 5,6,7, or 8 data bits along with either 1 or 2 STOP bits. The examples we will be using will be formatted so as to transmit and receive 1 START bit, 8 data bits, and 2 STOP bits. The bit transfer rate can be set to 110, 150, 300, 600, 1200, 2400, 4800, and 9600 bits per second; the actual value depending upon the application.

Figure 2.7 shows a circuit which converts TTL voltage levels to RS232-C voltage levels, and which can be used with the program shown in Listing 2.1 to accomplish a parallel-to-serial output.

FIGURE 2.7 TTL to RS232-C interface.



Number	Type	+5	+12	-12	GND
IC1	MC1488	--	14	1	7
IC2	MC14013	14	--	--	7
IC3	SN7427	14	--	--	7
IC4	MC14049	1	--	--	8

LISTING 2.1 Subroutine for parallel-to-serial output. Program assumes that the character to be sent will be in the A register.

```

30F 08      PHP      SAVE STATUS
310 A0 B0    LDY      11 BITS
312 18      CLC      CLEAR CARRY
313 48      PHA      SAVE A REGISTER
314 B0 07    BCS      $31D
316 A9 00    LDA      $31D
318 8D F1 C0 STA      $C0F1    SEND A SPACE
31B 90 05    BCC      $322
31D A9 01    LDA      $301
31F 8D F1 C0 STA      $C0F1    SEND A MARK
322 20 40 03 JSR      $340
325 68      PLA      JUMP TO DELAY
326 6A      ROR      RESTORE A REGISTER
327 88      DEY      ROTATE A REGISTER
328 D0 E9    BNE      $313    ALL BITS SENT?
32A 28      PLP      BRANCH IF NOT
32B 60      RTS      RESTORE STATUS
                        RETURN

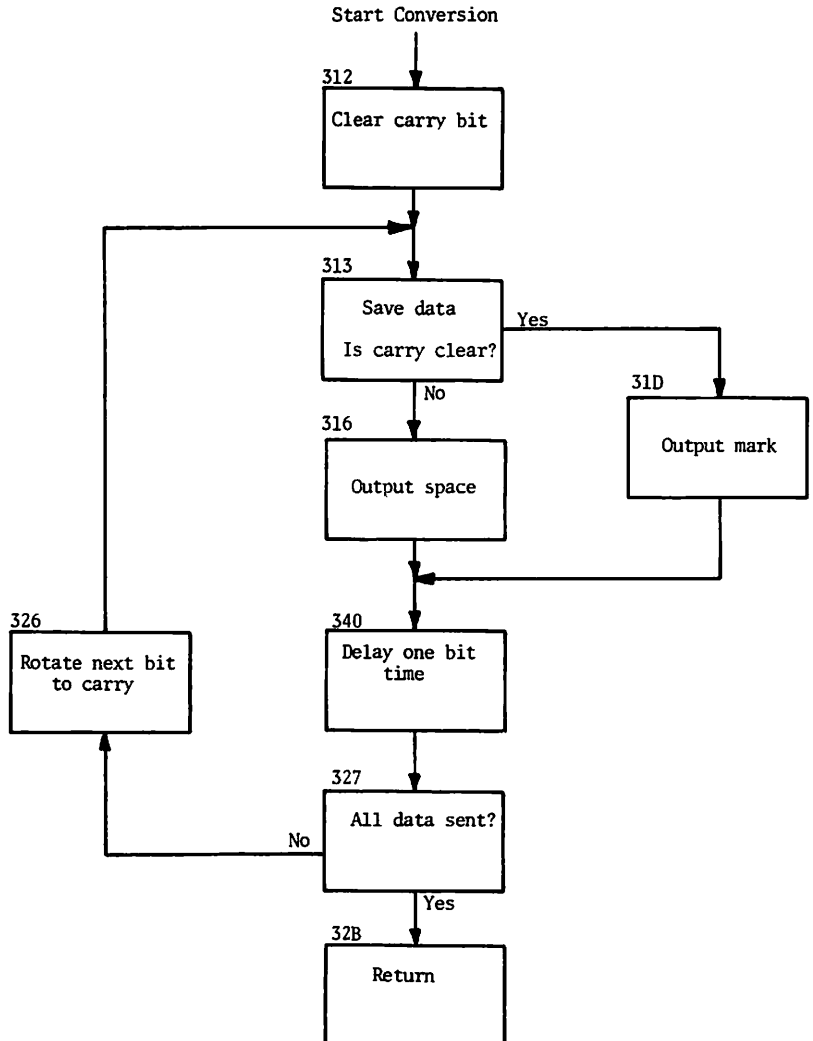
340 A9 4D    LDA      $4D      LOAD A REGISTER
342 48      PHA      SAVE A REGISTER
343 A9 20    LDA      $20      LOAD A REGISTER
345 4A      LSR      SHIFT A REGISTER
346 EA      NOP
347 90 FC    BCC      $345
349 68      PLA      RESTORE A REGISTER
34A E9 01    SBC      $301      SUBTRACT 1
34C D0 F4    BNE      $342
34E 60      RTS      RETURN

```

When a STA \$C0F1 of machine language command is executed, the contents of the least significant bit of the (data bit 0₀) accumulator are latched into the output of the D-type flip-flop (IC2). Figure 2.8 shows a flow chart for the TTL to RS232-C program.

Figure 2.9 shows a circuit which can be used to convert RS232-C voltage levels to TTL levels, and which can be used with the program shown in Listing 2.2 to accomplish a serial-to-parallel conversion. When a LDA \$C0F1 machine language command is executed, the voltage level appearing on the SERIAL IN line will be strobed into data bit D7 of the accumulator. Figure 2.10 shows a flow chart for the RS232-C to TTL program. Timing for both subroutines results in a baud rate of 300 bits per second. The serial output subroutine (Listing 2.1) expects the character which is to be sent to be stored in the accumulator. The serial input subroutine (Listing 2.2) will terminate with the received character displayed on the video monitor. We will put both of these circuits to use in Chapter 5 where we will see how to use the Apple II in serial applications.

FIGURE 2.8 Flow chart for the TTL to RS232-C program.



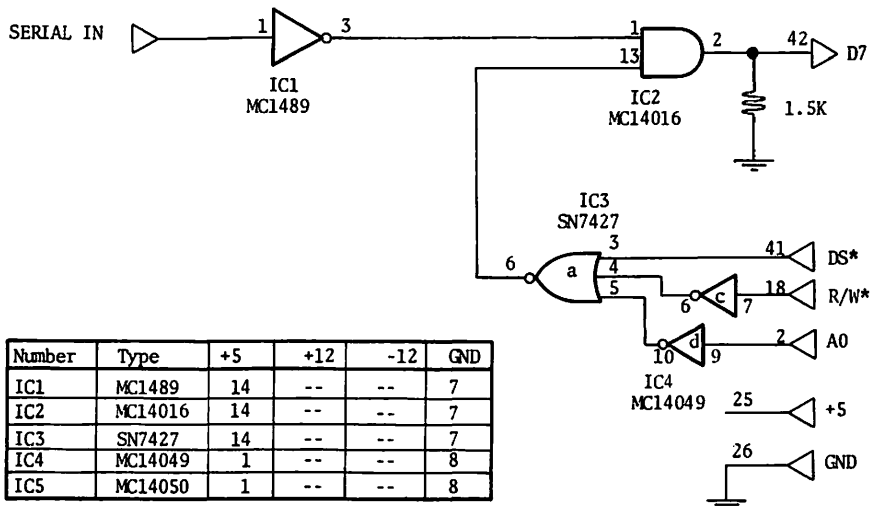


FIGURE 2.9 RS232-C to TTL interface.

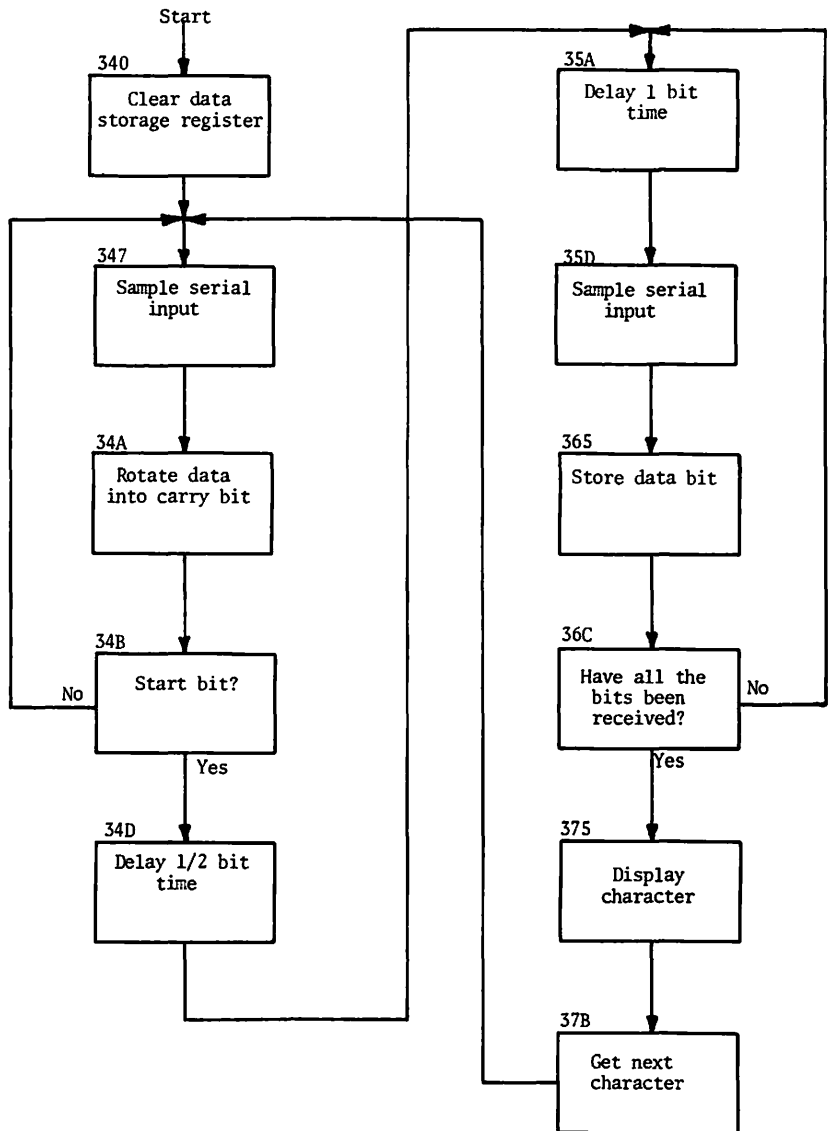
LISTING 2.2 Subroutine for serial-to-parallel conversion. Program displays the received character on the video display.

```

330 A9 4D LDA #$4D LOAD A REGISTER
332 48 PHA SAVE A REGISTER
333 A9 20 LDA #$20 LOAD REGISTER
335 4A LSR SHIFT RIGHT
336 90 FD BCC $335 JUMP IF NOT ZERO
338 68 PLA RESTORE A REGISTER
339 E9 01 SBC #$01 SUBTRACT ONE
33B D0 F5 BNE $332 JUMP IF NOT ZERO
33D 60 RTS RETURN
33E EA NOP
33F EA NOP
340 A9 00 LDA #$00 LOAD A REGISTER
342 8D 80 03 STA $380 TEMP A REGISTER
345 A2 08 LDX #$08 8 DATA BITS
347 AD F1 C0 LDA $C0F1 GET RECEIVED CHARACTER
34A 2A ROL ROTATE LEFT
34B B0 B3 BCS $340 NO CHARACTER RECEIVED
34D A9 26 LDA #$26 1/2 CHARACTER DELAY
34F 48 PHA
350 A9 20 LDA #$20
352 4A LSR
353 90 FD BCC $352
355 68 PLA
356 E9 01 SBC #$01
358 D0 F5 BNE $34F
35A 20 30 03 JSR $330 JUMP TO DELAY
35D AD F1 C0 LDA $C0F1 GET CHARACTER
360 29 80 AND #$80 GET MSB
362 0D 80 03 ORA $380 OR A REGISTER WITH TEMP
365 8D 80 03 STA $380 SAVE IN TEMP
368 18 CLC CLEAR CARRY
369 6E 80 03 ROR $380 ROTATE TEMP RIGHT
36C CA DEX ALL BITS RECEIVED?
36D D0 EB BNE $35A BRANCH IF NOT
36F 09 80 ORA #$80 SET MSB
371 C9 C0 CMP #$C0 CHECK FOR CONTROL CHARAC
373 F0 03 BEQ $378 JUMP IF NOT
375 20 FD FB JSR $FBFD DISPLAY ON VIDEO
378 20 30 03 JSR $330 JUMP TO DELAY
37B 4C 00 03 JMP $340 GET NEXT CHARACTER

```

FIGURE 2.10 Flow chart for the RS232-C to TTL converter.



3

SAMPLING THE EXTERNAL WORLD

Now that we have covered some background material relative to the operation of the Apple II computer, we can proceed to the “fun” part of computer operation, the part that involves the evaluation of a need, and the construction of a circuit to meet that need. We will be progressing from examples that are very simple to examples that are somewhat complicated; with the confidence that once you can build and use simple circuits, you will be able to build and use the more complicated ones. In Appendix A I have made a personal appraisal of several construction techniques which you may want to consider; it may be to your advantage to review this material at this time.

STATUS OF A BINARY SIGNAL

For some applications we will want to be able to sample an input port to determine the status or state of an external device. For example, we might want to know if a device was turned on, a valve opened, or a switch closed. Figure 3.1 shows a schematic of a circuit which will allow you to test the state of six switches. When a `X=PEEK(-16143)` BASIC command is executed, a voltage level, which is a function of whether each individual switch is open or closed, will be strobed into the computer. The variable X can then be decoded to determine the status of each switch. You might use such a circuit to allow external control of a program which is being executed, or to take an action which would be consistent with the state of the switch. The switches might be simple toggle switches mounted on a panel, or they might be relay switches associated with some automated process that you want to monitor. Figure 3.2 shows a flow chart for a BASIC program which will test the status of each switch and display that status on the video display. Listing 3.1 shows the program with comments. This program is intended to be called as a subroutine, and is set up so that if the status has not changed, control will return to the calling routine.

While I have used mechanical toggle switches for this example, the application could be easily extended to the following:

1. Water level sensing switches
2. Displacement sensing microswitches
3. Light sensitive, photoswitches
4. Temperature sensitive, bimetallic switches
5. Motor relay switches

An advantage of using the computer to sense and respond to the status of a switch lies in its ability to be reprogrammed easily when the application changes.

ANALOG-TO-DIGITAL CONVERSION

Background

There are many applications where we will want to have our computer system "measure" some parameter which varies as a function of time. In most cases, we will be using some type of

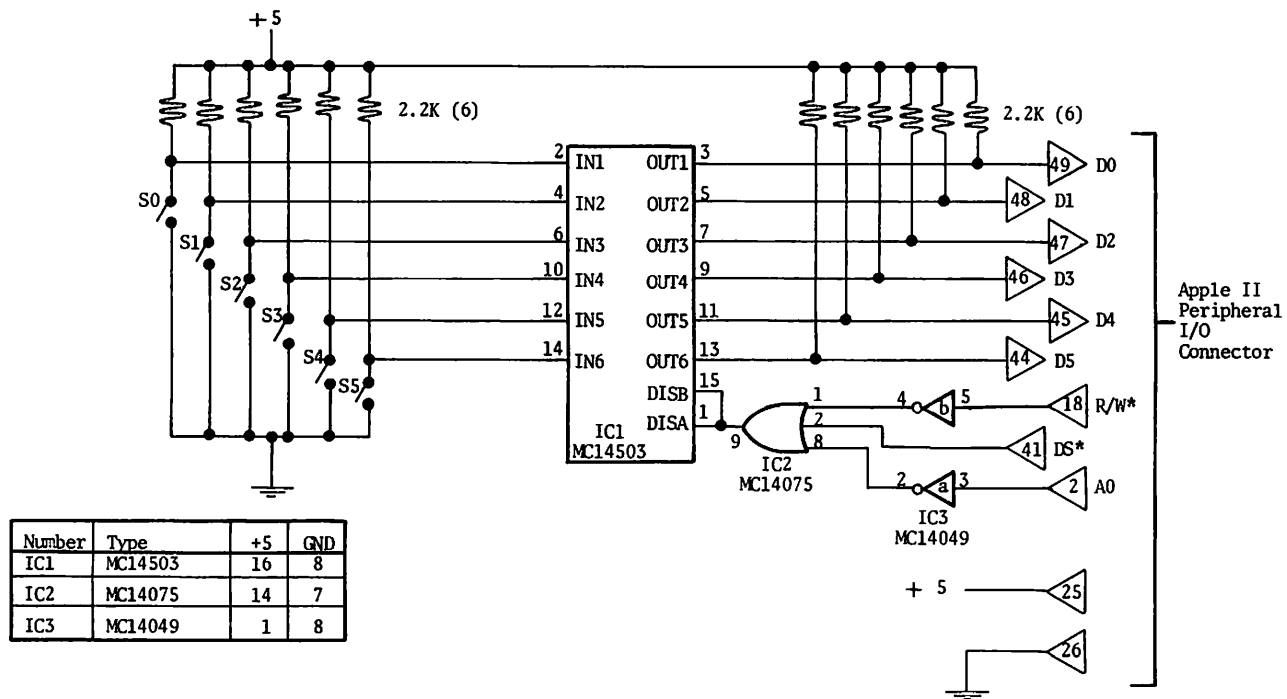


FIGURE 3.1 Schematic of switch status input port.

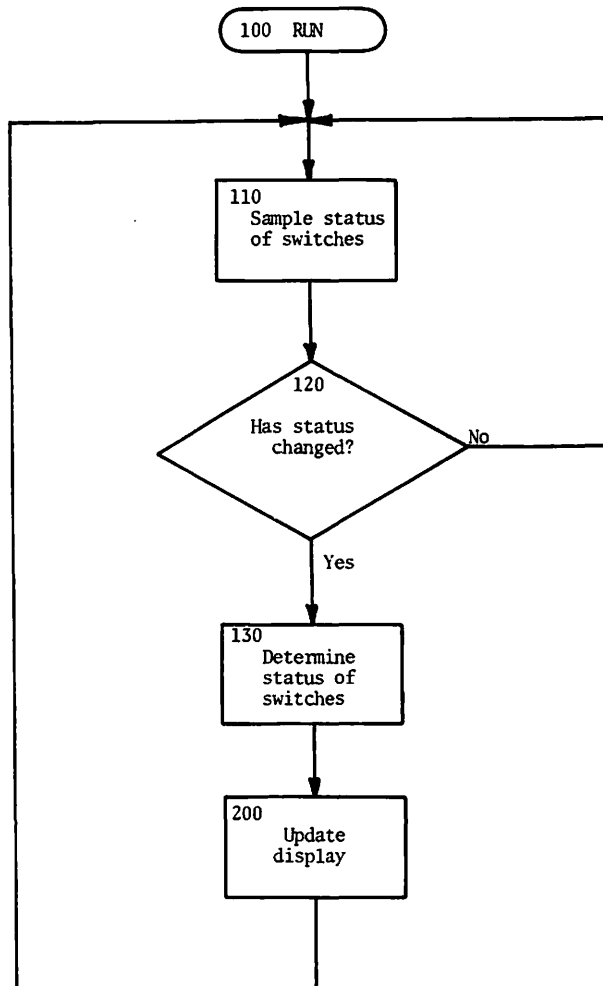


FIGURE 3.2 Flow chart of switch status program.

transducer which takes a physical parameter such as temperature, pressure, strain, or position, and converts it into an electrical voltage or current. While computers are quite proficient when handling binary voltages, they are not directly able to handle the type of analog voltages that will be coming from the transducers we will be using. To resolve this problem we need to build an interface called an analog-to-digital (A/D) converter. Our design criteria will be primarily concerned with determining appropriate values, based on our particular application, for resolution

LISTING 3.1 BASIC program for determining and displaying switch status.

```

100 REM SWITCH STATUS ROUTINE
102 REM APPLE II, R. HALLGREN, 10-20-81
104 X1=0
110 REM SAMPLE SWITCH STATUS
112 X=PEEK(-16143)
120 REM STATUS CHANGED?
122 IF X>127 THEN X=X-128
124 IF X>63 THEN X=X-64
126 IF X=X1 THEN GOTO 110
130 REM DETERMINE SWITCH STATUS
132 X1=X:IF X>31 THEN D5=1:X=X-32:GOTO 140
136 D5=0
140 IF X>15 THEN D4=1:X=X-16:GOTO 150
142 D4=0
150 IF X>7 THEN D3=1:X=X-8:GOTO 160
152 D3=0
160 IF X>3 THEN D2=1:X=X-4:GOTO 170
162 D2=0
170 IF X>1 THEN D1=1:X=X-2:GOTO 180
172 D1=0
180 IF X=1 THEN D0=1:GOTO 184
182 D0=0
184 IF D0=1 THEN D0$="ON"
185 IF D0=0 THEN D0$="OFF"
186 IF D1=1 THEN D1$="ON"
187 IF D1=0 THEN D1$="OFF"
188 IF D2=1 THEN D2$="ON"
189 IF D2=0 THEN D2$="OFF"
190 IF D3=1 THEN D3$="ON"
191 IF D3=0 THEN D3$="OFF"
192 IF D4=1 THEN D4$="ON"
193 IF D4=0 THEN D4$="OFF"
194 IF D5=1 THEN D5$="ON"
195 IF D5=0 THEN D5$="OFF"
200 REM UPDATE DISPLAY
210 HOME
212 PRINT "*****"
222 PRINT "*"
226 PRINT "*" SWITCH S0 IS ";D0$
228 PRINT "*"
230 PRINT "*" SWITCH S1 IS ";D1$
232 PRINT "*"
234 PRINT "*" SWITCH S2 IS ";D2$
236 PRINT "*"
238 PRINT "*" SWITCH S3 IS ";D3$
240 PRINT "*"
242 PRINT "*" SWITCH S4 IS ";D4$
244 PRINT "*"
246 PRINT "*" SWITCH S5 IS ";D5$
248 PRINT "*"
250 PRINT "*****"
299 GOTO 110

```

(dR), relative system error, and sampling frequency (F_s). We will take a look at three examples of A/D converters which are very different in design, using separate and distinct processes to perform the analog-to-digital conversion. I will spend the rest of this section making general comments that will apply to each example, saving specific comments for the individual circuits.

For our purposes, we will define an analog signal as being generally smooth, continuous, and analytical, for example, the output voltage from a pressure transducer, the output voltage from some analytical instrument, or the voltage resulting from some physiological phenomenon such as the electrocardiogram. We shall define a discrete signal as one which is constrained to specific voltage levels, the interval between levels determined by dividing the maximum voltage range ($V_{\max}$) by the number of possible intervals (K). Consequently, any analog voltage which is within the maximum voltage range can be represented as a discrete quantity, the magnitude being equal to some multiple of dR , where dR is defined by Equation 3.1:

$$\frac{dR = V_{\max}}{k} \quad (3.1)$$

Resolution

The function of an A/D converter is to transform an analog quantity into a digital quantity which can be handled by a computer. This transformation usually involves sampling the continuous voltage, representing the voltage as a discrete value, and then forming this quantity so that the computer can process it. An example of a signal represented in both continuous and discrete forms is shown in Figure 3.3. In general, we will assume that Y axis values are dependent upon X axis values such that Y can be expressed as some function of X .

$$Y = F(X) \quad (3.2)$$

Resolution, as we will be using the term, is related to our ability to distinguish between two voltages that have magnitudes which are very close together. The smallest magnitude difference we can detect will define the resolution of our system. For each of the analog-to-digital systems we will discuss, resolution will be equal to the magnitude of a unit change in the least significant digit of the output of the converter.

Relative System Error

Accuracy is related to our ability to determine the actual value of an unknown input voltage. Accuracy will be a function of the

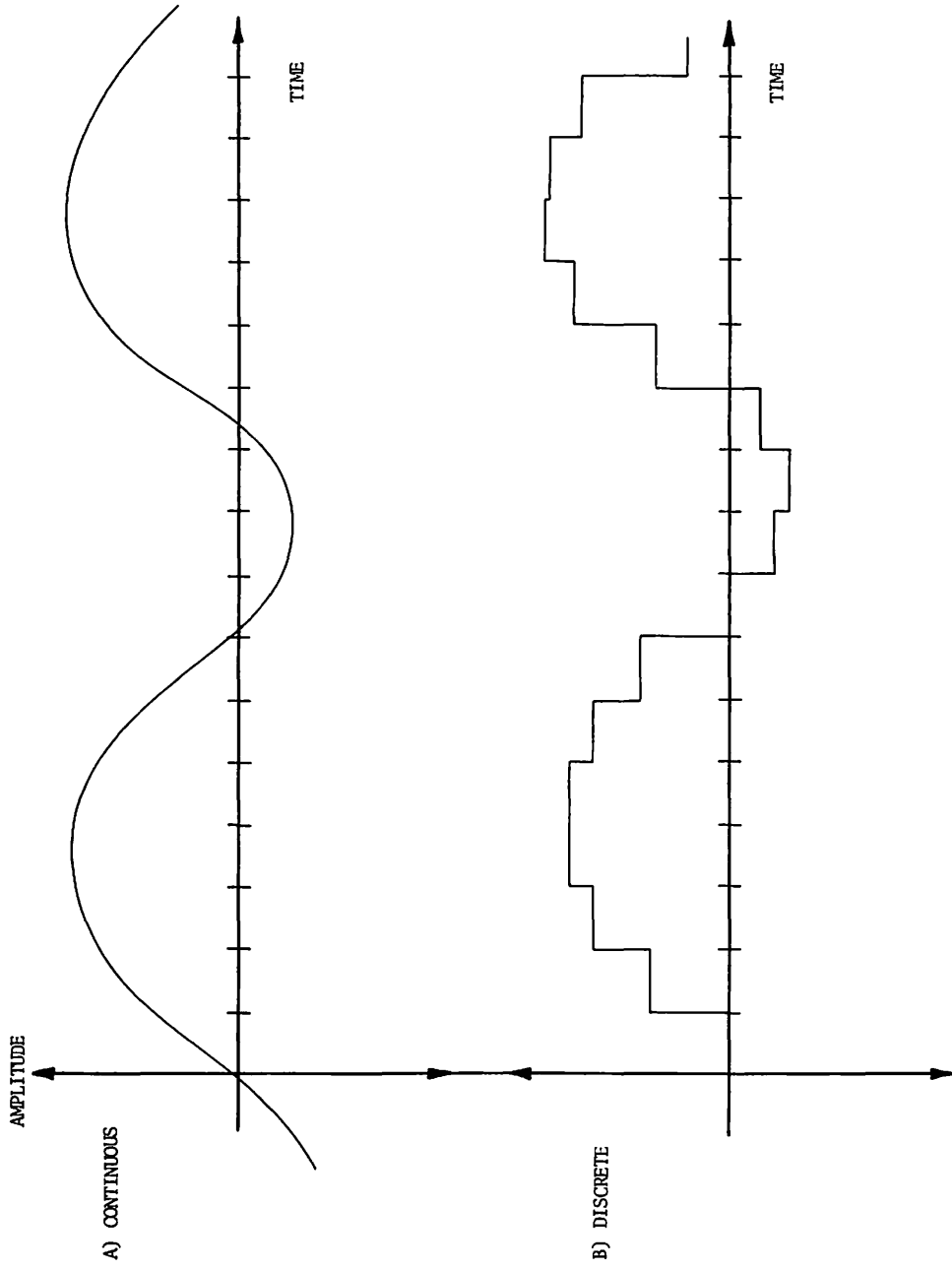


FIGURE 3.3 Continuous and discrete waveforms.

linearity of the system, gain and offset errors, resolution, and magnitude of the signal that we are measuring. In general, the main source of system inaccuracy usually results from the necessity to represent the analog voltage by a discrete quantity. If we assume that this is true for the circuits that we will be building, we can define relative system error as the ratio of system resolution to the magnitude of the voltage being measured. Obviously, for a given system, relative system error will be minimized when the unknown input voltage is equal to the maximum value that the system can handle.

Sampling Theory

Since an A/D converter samples an input signal sequentially with respect to time, the output voltage (Y) can be expressed in the form $Y(1)$, $Y(2)$, and so on. $Y(N)$, where N is equal to the sample number. The rate (F_s) at which we sample the input signal determines the maximum frequency component which can theoretically be detected. The sampling rate can be expressed as a function of the time between samples by the following equation:

$$F_s = 1/dt \text{ samples per second where } dt = [t/(N+1) - t(N)] \quad (3.3)$$

The sampling theorem states that we should sample at a rate which is at least twice the frequency (f_c) of the highest significant frequency component that is present in the waveform which we will be digitizing. In practice, the sampling frequency should be ten times F_c to allow you to reconstruct the complex waveform from the digital data without having to be overimaginative. Remember that you can always throw out data points once you have them if you find that you have sampled at a higher rate than was necessary.

In some applications, you might not be interested in the highest frequency component that was contained in the input signal, but only in lower frequency components. Aliasing, impersonation of a low frequency signal by a high frequency signal, results from choosing a sampling rate that is too low for the frequency components that make up the input signal. Figure 3.4 shows how a relatively high frequency signal and a relatively low frequency signal can share identical sample points, thus making it

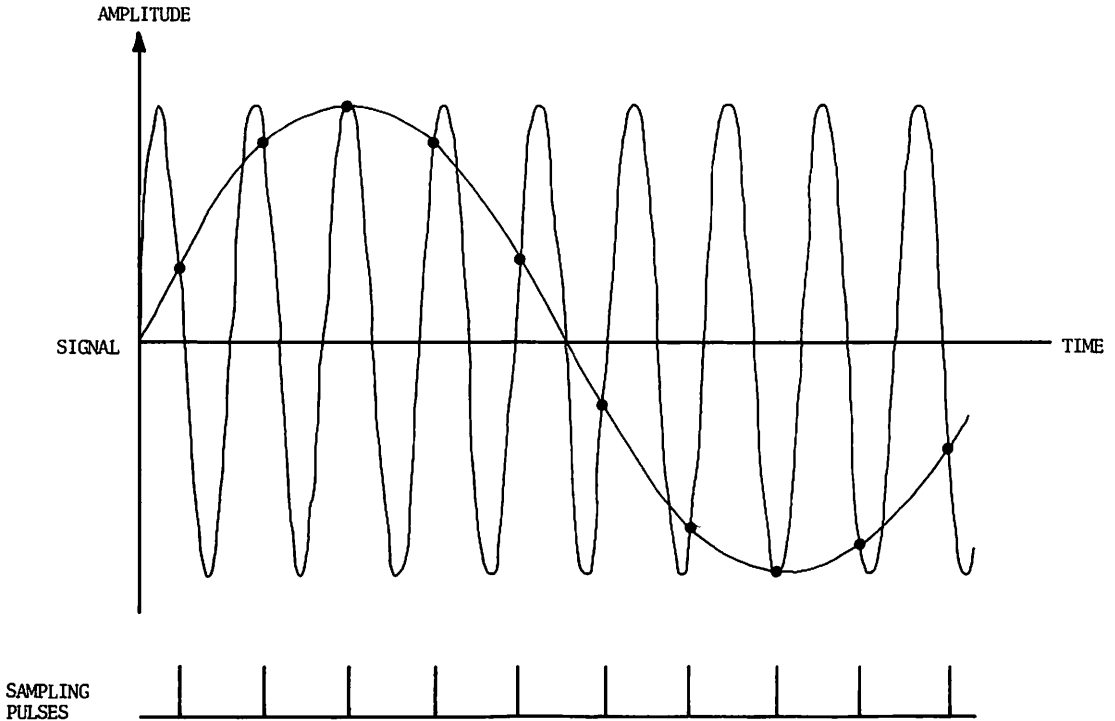


FIGURE 3.4 Aliasing error caused by inadequate sampling rate.

impossible to differentiate between the two signals. For applications where you will be sampling at rates that are less than $2F_c$, you must insert a low pass filter at the input of the A/D converter to limit the frequency content and to ensure faithful reproduction of the input signal.

8-bit, Low-speed A/D Converter

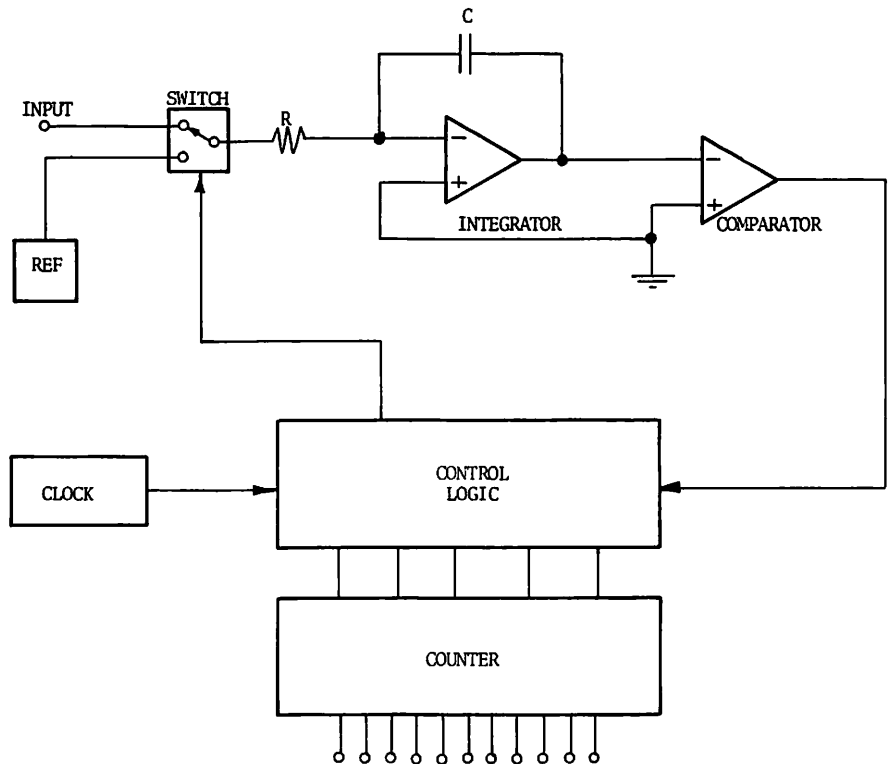
The MC1443 (Motorola Semiconductor Products Inc., Austin, Texas 78721) is a high-performance, low-power, three and one half digit A/D converter combining both linear CMOS and digital CMOS circuitry. This device, when combined with two external resistors and two external capacitors, forms a dual slope A/D converter featuring automatic zero correction and automatic polarity. By using an indirect conversion technique, the dual slope system realizes some of the following positive characteristics:

1. Conversion accuracy is independent of component values and the clock frequency.
2. Differential linearity is excellent.
3. The integration process provides rejection of high-frequency noise and averaging of small voltage changes that may occur during the conversion process.

The main disadvantage to the dual slope technique is that the analog-to-digital conversion process takes a relatively long time, consequently limiting the sampling rate. In spite of this, the device is ideal for digital voltmeters, digital thermometers, and other such devices not requiring sampling rates over 100 samples per second.

Figure 3.5 has been included to help you understand the dual slope conversion process. The actual configuration of the analog circuitry is dependent on the polarity of the input voltage during

FIGURE 3.5 Simplified block diagram for a typical dual slope A/D converter.



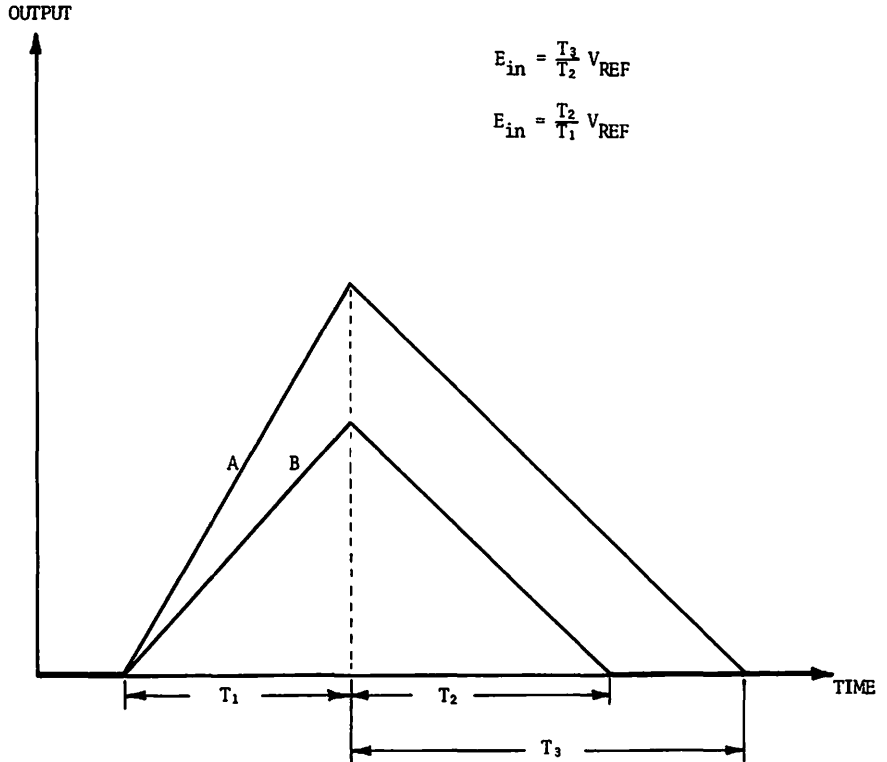


FIGURE 3.6 Integrator output plotted as a function of time for two different input voltages converted by the dual slope process.

the previous conversion cycle, and it is automatically switched by the internal logic circuits. I have chosen to show the analog circuit configuration that would be used when a negative voltage is connected to the input. When performing an A/D conversion, the MC14433 internal control logic first connects the unknown input voltage to the input of an integrator for a fixed period of time determined by an internal clock (See Figure 3.6 for a plot of the integrator output as a function of time.) This results in the voltage across the integrator capacitor increasing to a value which is proportional to the input voltage averaged over the time interval that the input is connected. At the end of this interval, the input to the integrator is switched from the unknown voltage to a reference voltage. The purpose of the reference voltage is to cause the integrator capacitor to discharge at a known rate while a counter

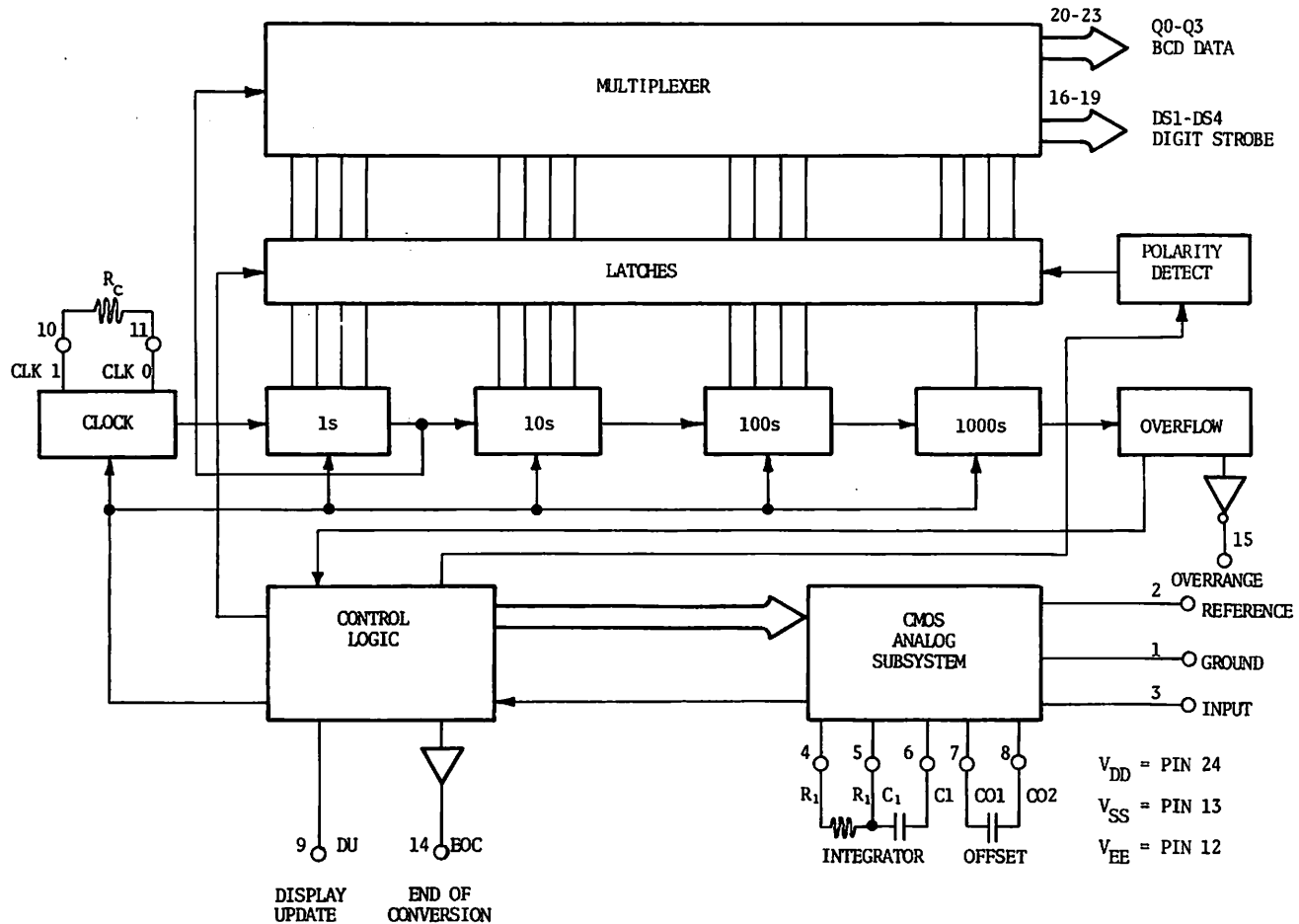
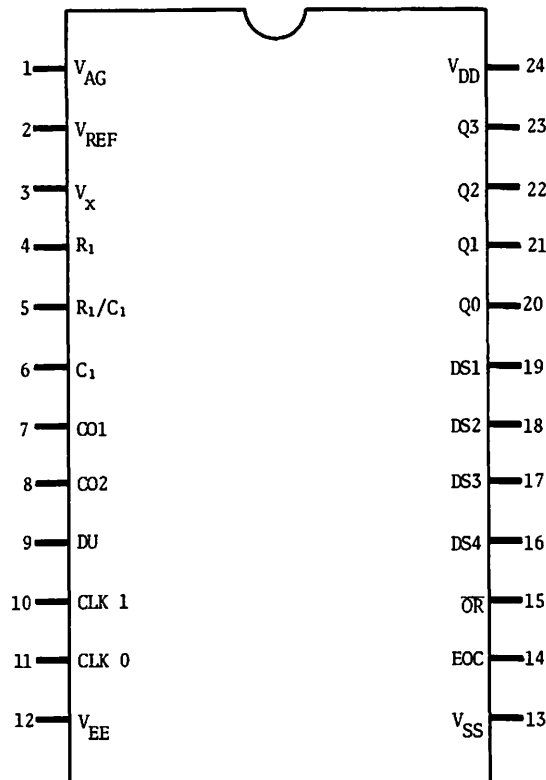


FIGURE 3.7 Block diagram of the Motorola MC 1433 3 1/2 digit A/D converter.

keeps track of the length of time it takes the voltage to return to zero. Note that the count associated with a given input voltage is a function of the reference voltage and the ratio of the time that it takes to charge and discharge the integrator capacitor. In summary, the MC14433 uses a ratiometric measurement scheme that produces an output reading that is a function of the ratio of the unknown input voltage to the reference voltage, where a ratio of 1 equals a count of 1999. It should be obvious to the reader that absolute stability of the reference voltage is necessary to ensure reproducible readings from sample to sample.

Figure 3.7 shows the actual block diagram and Figure 3.8 shows the pin assignment for the MC14433. This integrated circuit operates from standard 5 volt positive (pin 24) and 5 volt negative (pin 12) supplies. (See Appendix C for power supply circuit configurations.) The absolute value of the input voltage

FIGURE 3.8 Pin assignment for the MC14433.



$$V_{\text{out}} = \left[\frac{R_2}{R_1 + R_2} \right] V_{\text{in}}$$

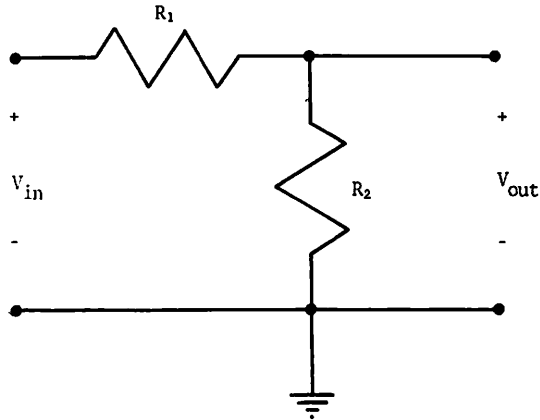
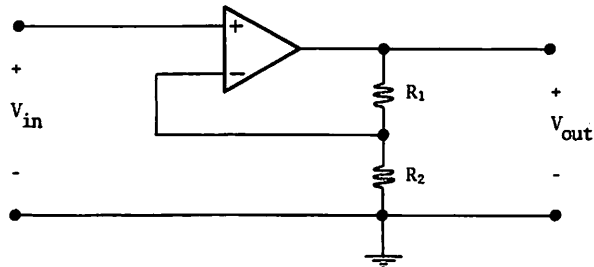


FIGURE 3.9 Simple resistive voltage divider network.

can vary between 0 and 1.999 volts. If the magnitude of the input voltage is greater than the upper limit of 1.999 volts, you can use the voltage divider circuit shown in Figure 3.9 to reduce the voltage to an acceptable value. On the other hand, if the input voltage is less than .2 volts, you will achieve greater accuracy by using a circuit similar to the one shown in Figure 3.10 to increase the magnitude of the input voltage. Appendix B has been in-

FIGURE 3.10 Simple noninverting amplifier.

$$V_{\text{out}} = \left[1 + \frac{R_2}{R_1} \right] V_{\text{in}}$$



cluded to explain the operation of the amplifier circuits such as the one shown in this figure. The frequency of the internal system clock of the MC14433 is controlled by selection of the resistor connected between pins 10 and 11. Figure 3.11 shows typical values for frequency as a function of resistance, and the following equation relates the conversion rate (samples per second) to the clock frequency:

$$\text{Conversion rate} = \text{Clock frequency}/16,400 \quad (3.4)$$

Normal operation of MC14433 permits the conversion of an analog signal to a digital signal on a continuous basis, but the results of each conversion are transferred to the output latches only when an appropriate logic pulse is applied to the DISPLAY UPDATE (DU) (pin 9) control line. We can control this transfer of data by providing a positive going pulse to the DU line prior to the ramp-down segment of the conversion cycle. The data that are strobed into the output latches will then be equal to the digital quantity that was obtained from the last A/D conversion.

During the time that the conversion is being performed, the status line, END OF CONVERSION (EOC) (pin 14), is low indicating that the converter is busy. Upon completion of the conversion cycle, this line produces a pulse whose width is equal to one half of the period of the system clock. When this output is wired directly back into the DU input, the result of every conversion is strobed into the output latches. I will use the EOC status line to indicate to the computer when a conversion has been completed and, consequently, when new data has been strobed into the output latches.

The result of an analog-to-digital conversion is contained in the internal output latches in a binary format. However, the data are sent to the external world over the data lines (pins 20 to 23) in a multiplexed, binary coded decimal (BCD) format. 8 bits of information are used to transfer the results of a conversion. The four most significant bits (pins 16 to 19) contain the code for a particular digit, and the four least significant bits (pins 20 to 23) contain the binary value of the digit that is selected. The digit select output lines (pins 16 to 19) go high when the respective digit is selected, and the data lines then contain the value of the selected digit. The digit select timing sequence is arranged so that the most

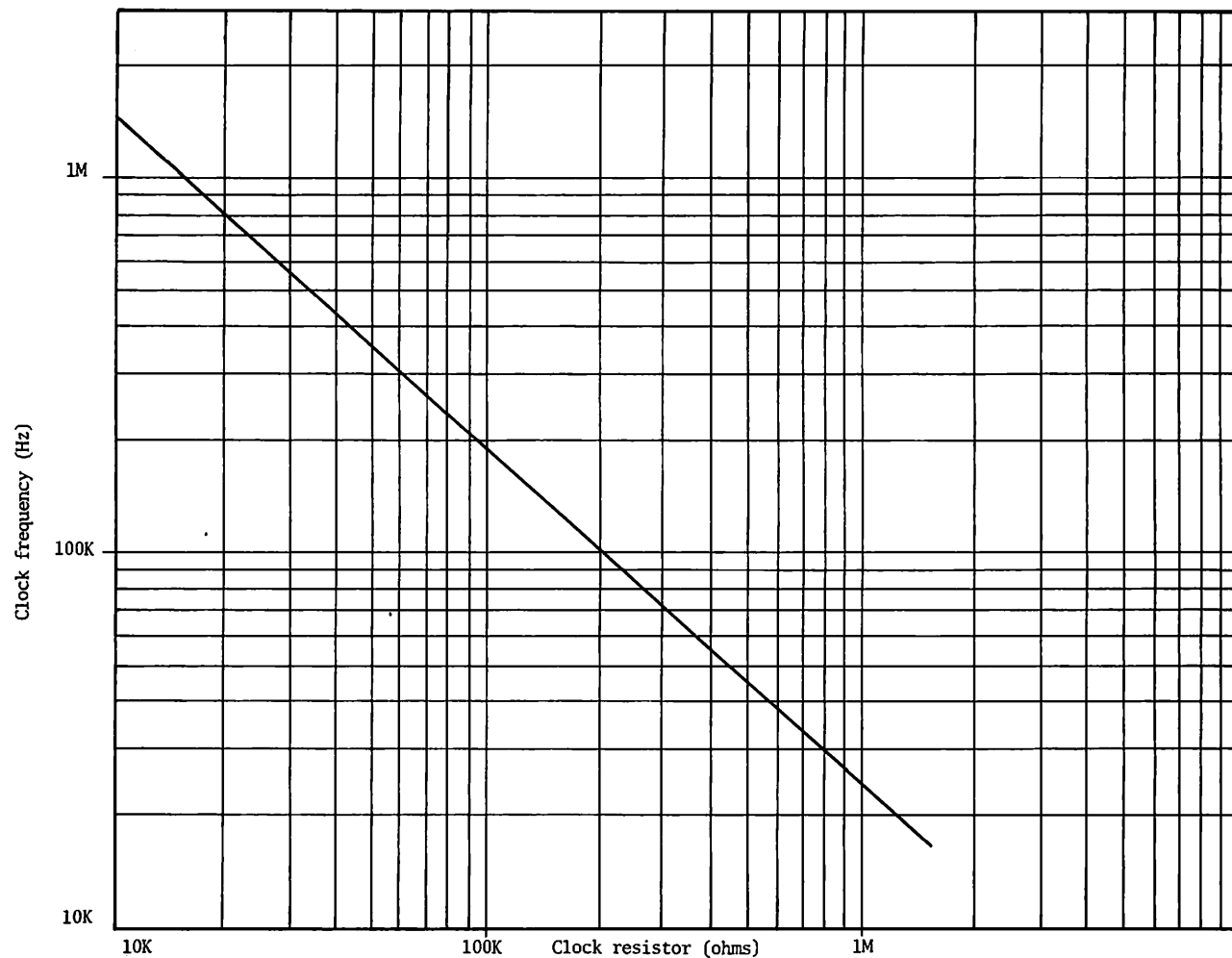


FIGURE 3.11 Typical clock frequency as a function of clock resistor (R_c).

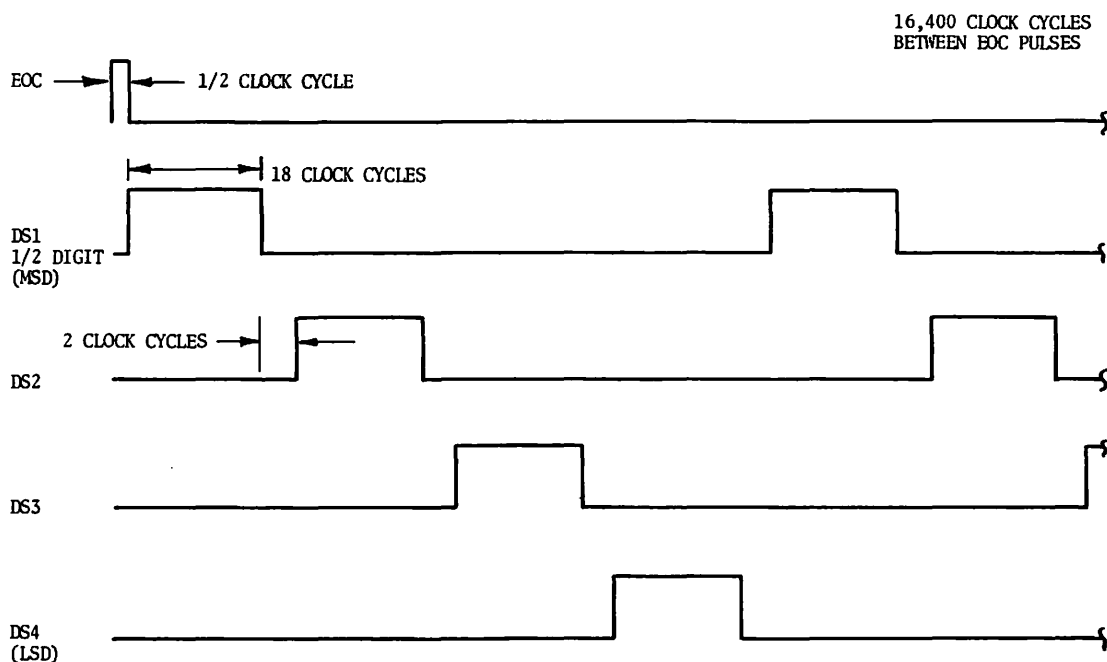


FIGURE 3.12 Digit select timing diagram for the MC14433.

significant digit (MSD) (one half digit) is selected immediately after an EOC pulse. The remaining digits are sequenced as shown in Figure 3.12. When the MSD is multiplexed, the data lines contain not only the value of that digit but also coded information relating to overrange, underrange, and polarity as shown in Table 3.1.

Figure 3.13 shows the low-speed, A/D converter circuit configured for the Apple II. Looking at the upper portion of the figure, you will see the MC14433 A/D converter. The clock resistor (R_c) has been selected and the EOC and DU control lines connected together so that fifteen conversions per second are being performed on a continuous basis. All data and status lines to the Apple II are isolated through the MC14503 three-state buffers (IC7 and IC8). IC4, configured as an R/S flip-flop that is initially reset by the computer, is set by the MC14433 after an analog-to-digital conversion has been completed. When the Apple II senses this change in status, it starts the decoding and data transfer process. The 6502 processor speed is high enough to enable one conversion to be decoded and stored before another conversion begins. A

TABLE 3.1 Truth table showing the information format on the multiplexed data lines during selection of DS1.

Coded Consition of MSD	Q3	Q2	Q1	Q0	BCD to 6 Segment Decoding
+0	1	1	1	0	Blank
-0	1	0	1	0	Blank
+0 UR	1	1	1	1	Blank
-0 UR	1	0	1	1	Blank
+1	0	1	0	0	4 → 1 Hook up
-1	0	0	0	0	0 → 1 only seg b
+1 OR	0	1	1	1	7 → 1 and c to
-1 OR	0	0	1	1	3 → 1 MCD

Notes for Truth Table

Q3 - 1/2 digit, low for "1", high for "0"

Q2 - Polarity: "1" = positive, "0" = negative

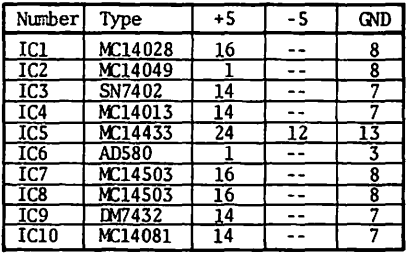
Q0 - Out of range condition exists if Q0 = 1. When used in conjunction with Q3 the type of out of range condition is indicated, i.e., Q3 = 0 → OR or Q3 = 1 → UR.

The overrange indication (Q3 = 0 and Q0 = 1) occurs when the count is greater than 1999, e.g., 1.999V for a reference of 2.000 V. The underrange indication, useful for autoranging circuits, occurs when the count is less than 180, e.g., 0.180 V for a reference of 2.000 V.

stable reference voltage source is obtained from the output of the Analog Devices AD580 voltage regulator (IC6).

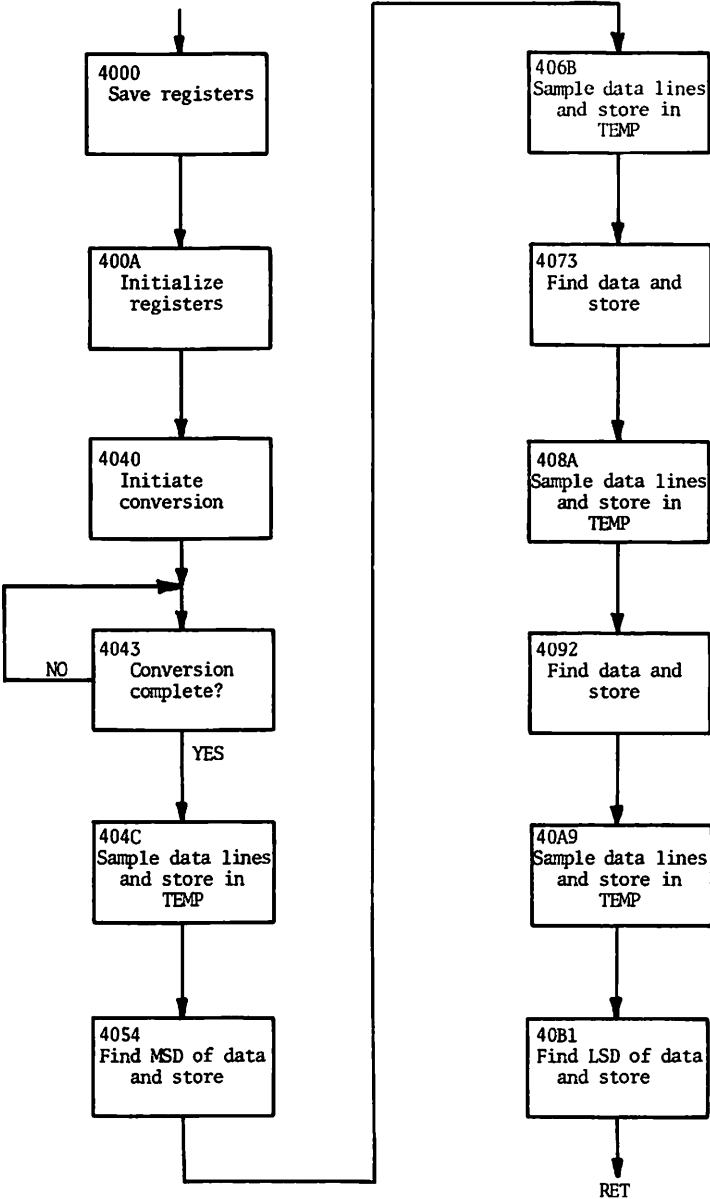
The right lower section of Figure 3.13 shows the control logic that is used to coordinate the transfer of signals to and from the computer. The least significant four bits of the address bus are decoded by IC1 and are used for on-board addressing. By executing an STA \$C0F2 machine language command, the status control line (SC) is forced high, resetting IC4. Performing an LDA \$C0F1 machine language command transfers the end of conversion (EOC) and overrange (OR) status information into the accumulator. Performing an LDA \$C0F1 machine language command transfers the digit select code and the BCD value of the selected digit into the accumulator.

The software portion of the low speed A/D converter has been divided into two parts:



converter.

FIGURE 3.14 Flow chart for the machine language program 1.



1. A machine language routine was written to control the MC14433, and to provide high-speed transfer of data from the MC14433 to a specific location in computer memory.
2. An Applesoft BASIC program was written to take the data in memory and format it into a voltage that can be displayed as a decimal quantity on the video display.

Figure 3.14 shows the flow chart of the machine language program, and Listing 3.2 shows the coded program with comments. Upon entering the subroutine, the end of the conversion (EOC) flip-flop is reset and the program loops until the MC14433 completes the next conversion and sets the flip-flop. The program then samples the data lines and decides whether or not the most significant digit of data was sampled. If it was not, the program continues to sample the data lines until this digit is obtained. Once this digit is found, it is transferred to memory and the program looks for the second digit. After the four digits representing the digitized input voltage have been stored, the program executes an RTS and returns to the BASIC program.

LISTING 3.2 Machine language program for the low-speed A/D converter (single sample).

4000	8D B0 49	STA \$49B0	SAVE REGISTERS
4003	8E B1 49	STX \$49B1	
4006	8C B2 49	STY \$49B2	
4009	08	PHP	SAVE PROC STATUS
400A	A2 00	LDX #\$00	CLEAR X REGISTER
400C	A9 00	LDA #\$00	CLEAR A REGISTER
400E	85 0A	STA \$0A	START LOCATION FOR DATA
4010	A9 4A	LDA #\$4A	
4012	85 0B	STA \$0B	
4014	4C 40 40	JMP \$4040	JUMP TO TAKE DATA
4017	00	BRK	
4018	00	BRK	
4019	00	BRK	
401A	00	BRK	
401B	00	BRK	
401C	00	BRK	
401D	00	BRK	
401E	00	BRK	
401F	00	BRK	
4020	00	BRK	
4021	00	BRK	
4022	00	BRK	
4023	00	BRK	
4024	00	BRK	
4025	AD B0 49	LDA \$49B0	RESTORE REGISTERS
4028	AE B1 49	LDX \$49B1	
402B	AC B2 49	LDY \$49B2	
402E	28	PLP	RESTORE PROC STATUS
402F	60	RTS	RETURN TO BASIC
4030	EA	NOP	

LISTING 3.2 (Continued)

```

4031 EA NOP
4032 EA NOP
4033 EA NOP
4034 EA NOP
4035 EA NOP
4036 EA NOP
4037 EA NOP
4038 EA NOP
4039 EA NOP
403A EA NOP
403B EA NOP
403C EA NOP
403D EA NOP
403E EA NOP
403F EA NOP
4040 8D F2 C0 STA $C0F2 START CONVERSION
4043 AD F1 C0 LDA $C0F1 TEST FOR EOC
4046 29 80 AND #$80
4048 C9 80 CMP #$80
404A D0 F7 BNE $4043 JUMP IF EOC*
404C AD F0 C0 LDA $C0F0 INPUT DATA
404F 8D A2 49 STA $49A2 STORE TEMP
4052 29 80 AND #$80 CHECK FOR MSD
4054 C9 80 CMP #$80
4056 D0 F4 BNE $404C JUMP IF NOT MSD
4058 AD A2 49 LDA $49A2 GET DATA
405B 29 0F AND #$0F SAVE 4 BITS
405D 81 0A STA ($0A,X) STORE DATA
405F 4A 0A LDY $0A
4061 C8 INY INCREMENT STORAGE
4062 84 0A STY $0A
4064 D0 05 BNE $406B
4066 A4 0B LDY $0B INCREMENT STORAGE
4068 C8 INY
4069 84 0B STY $0B
406B AD F0 C0 LDA $C0F0
406E 8D A2 49 STA $49A2
4071 29 40 AND #$40 TEST FOR NEXT BYTE
4073 C9 40 CMP #$40
4075 D0 F4 BNE $406B JUMP IF NOT
4077 AD A2 49 LDA $49A2
407A 29 0F AND #$0F
407C 81 0A STA ($0A,X)
407E A4 0A LDY $0A
4080 C8 INY
4081 84 0A STY $0A
4083 D0 05 BNE $408A
4085 A4 0B LDY $0B
4087 C8 INY
4088 84 0B STY $0B
408A AD F0 C0 LDA $C0F0
408D 8D A2 49 STA $49A2
4090 29 20 AND #$20 TEST FOR NEXT DIGIT
4092 C9 20 CMP #$20
4094 D0 F4 BNE $408A
4096 AD A2 49 LDA $49A2
4099 29 0F AND #$0F
409B 81 0A STA ($0A,X)
409D A4 0A LDY $0A
409F C8 INY
40A0 84 0A STY $0A
40A2 D0 05 BNE $40A9
40A4 A4 0B LDY $0B
40A6 C8 INY

```

40A7	84 0B	STY	\$0B	
40A9	AD F0 C0	LDA	\$C0F0	
40AC	8D A2 49	STA	\$49A2	
40AF	29 10	AND	#\$10	TEST FOR NEXT DIGIT
40B1	C9 10	CMP	#\$10	
40B3	D0 F4	BNE	\$40A9	
40B5	AD A2 49	LDA	\$49A2	
40B8	29 0F	AND	#\$0F	
40BA	81 0A	STA	(\$0A,X)	
40BC	A4 0A	LDY	\$0A	
40BE	C8	INY		
40BF	84 0A	STY	\$0A	
40C1	D0 05	BNE	\$40C8	
40C3	A4 0B	LDY	\$0B	
40C5	C8	INY		
40C6	84 0B	STY	\$0B	
40C8	4C 25 40	JMP	\$4025	

The BASIC routine has the task of calling the machine language routine and assembling the four digits from each conversion back into a single number which is equal to the measured voltage. This program has been written using the CALL command to jump to the machine language routine. A flow chart for the BASIC program is shown in Figure 3.15 and Listing 3.3 shows the coded program. Listing 3.3 has the machine language program included in it as a series of DATA statements that are read in by line 22. You only have to load the BASIC program to get the A/D converter to work. This program performs A/D conversions on a continuous basis, displaying the value of the analog input voltage on the video monitor.

Once the circuitry has been built, you still have the job of finding any errors that may have been made during construction. To assist you in this process, I am including the following checks for you to make:

1. The voltage at pin 2 on IC5 should be set to 2.00 volts.
2. Using an oscilloscope attached to pin 10 on IC5, you should see a 150,000 Hz distorted square wave varying between +2 and - 2 volts.
3. The voltage at pin 1 on IC4 should be equal to +5 volts. Run the following program to repeatedly reset the flip-flop.


```

100 POKE -16142,0
110 FOR I=1 TO 20
120 NEXT I
130 GOTO 100
      
```
4. After you have loaded the BASIC program, connect a 1.5 volt battery to the input and execute the program. You should see a

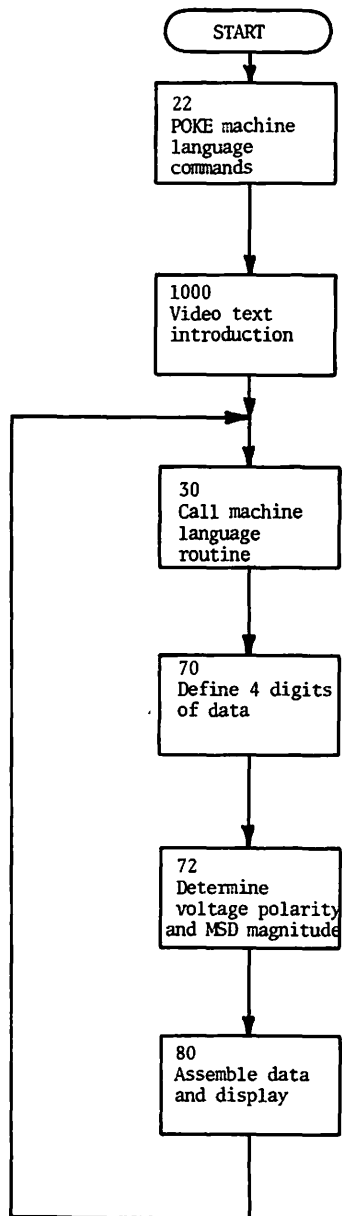


FIGURE 3.15 Flow chart for the BASIC program 1.

onds (fifteen samples per second). Listing 3.4 shows a machine language program that takes a total of 256 samples before returning to the calling routine. Figure 3.16 shows a flow chart for the program. Listing 3.5 shows a BASIC program which calls the machine language routine and then plots the data. Figure 3.17 shows a flow chart for the BASIC program. The BASIC program has the machine language program included in it as a series of DATA statements that are read in by line 24.

Hooking the 1.5 volt battery to the input should give a straight line plot. Figure 3.18 shows a simple resistor, capacitor, and battery combination which can be used to demonstrate the capability of the system. Initially, disconnect the battery and connect the input of the A/D converter across the capacitor. Begin

LISTING 3.4 Machine language program for the low-speed A/D converter (256 samples).

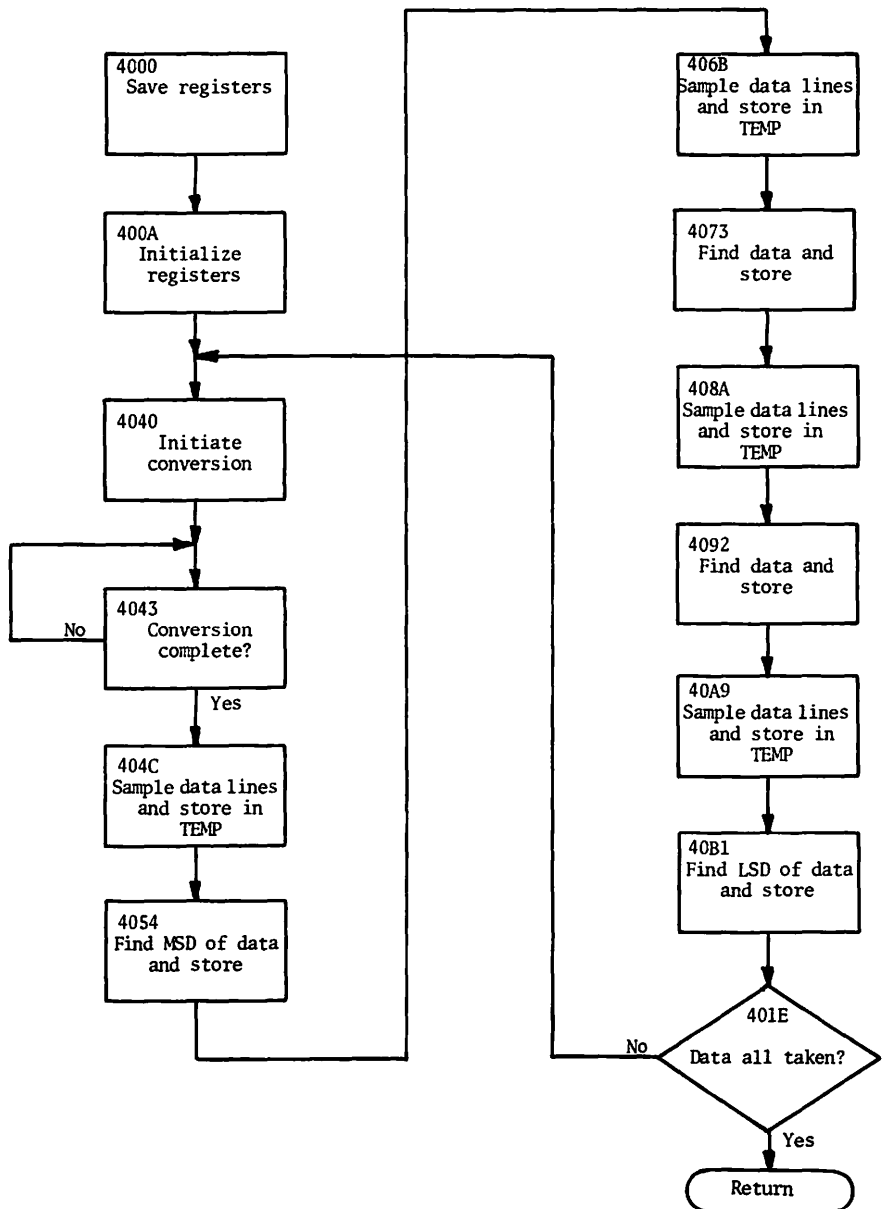
```

4000 8D B0 49   STA  $49B0   SAVE REGISTERS
4003 8E B1 49   STX  $49B1
4006 8C B2 49   STY  $49B2
4009 08         PHP                SAVE PROC STATUS
400A A2 00       LDX  #$00       CLEAR X REGISTER
400C A9 00       LDA  #$00       CLEAR A REGISTER
400E 85 0A       STA  $0A       START LOCATION FOR DATA
4010 A9 4A       LDA  #$4A
4012 85 0B       STA  $0B
4014 A9 00       LDA  #$00       STOP LOCATION FOR DATA
4016 8D A0 49   STA  $49A0
4019 A9 4E       LDA  #$4E
401B 8D A1 49   STA  $49A1
401E AD A1 49   LDA  $49A1
4021 C5 0B       CMP  $0B
4023 D0 1B       BNE  $4040
4025 AD B0 49   LDA  $49B0       RESTORE REGISTERS
4028 AE B1 49   LDX  $49B1
402B AC B2 49   LDY  $49B2
402E 28         PLP                RESTORE PROC STATUS
402F 60         RTS                RETURN TO BASIC
4030 EA         NOP
4031 EA         NOP
4032 EA         NOP
4033 EA         NOP
4034 EA         NOP
4035 EA         NOP
4036 EA         NOP
4037 EA         NOP
4038 EA         NOP
4039 EA         NOP
403A EA         NOP
403B EA         NOP
403C EA         NOP
403D EA         NOP
403E EA         NOP
403F EA         NOP
4040 8D F2 C0   STA  $C0F2       START CONVERSION

```

4043	AD F1 C0	LDA \$C0F1	TEST FOR EOC
4046	29 80	AND #\$80	
4048	C9 80	CMP #\$80	
404A	D0 F7	BNE \$4043	JUMP IF EOC*
404C	AD F0 C0	LDA \$C0F0	INPUT DATA
404F	8D A2 49	STA \$49A2	STORE TEMP
4052	29 80	AND #\$80	CHECK FOR MSD
4054	C9 80	CMP #\$80	
4056	D0 F4	BNE \$404C	JUMP IF NOT MSD
4058	AD A2 49	LDA \$49A2	GET DATA
405B	29 0F	AND #\$0F	SAVE 4 BITS
405D	81 0A	STA (\$0A,X)	STORE DATA
405F	4A 0A	LDY \$0A	
4061	C8	INY	INCREMENT STORAGE
4062	84 0A	STY \$0A	
4064	D0 05	BNE \$406B	
4066	A4 0B	LDY \$0B	INCREMENT STORAGE
4068	C8	INY	
4069	84 0B	STY \$0B	
406B	AD F0 C0	LDA \$C0F0	
406E	8D A2 49	STA \$49A2	
4071	29 40	AND #\$40	TEST FOR NEXT BYTE
4073	C9 40	CMP #\$40	
4075	D0 F4	BNE \$406B	JUMP IF NOT
4077	AD A2 49	LDA \$49A2	
407A	29 0F	AND #\$0F	
407C	81 0A	STA (\$0A,X)	
407E	A4 0A	LDY \$0A	
4080	C8	INY	
4081	84 0A	STY \$0A	
4083	D0 05	BNE \$408A	
4085	A4 0B	LDY \$0B	
4087	C8	INY	
4088	84 0B	STY \$0B	
408A	AD F0 C0	LDA \$C0F0	
408D	8D A2 49	STA \$49A2	
4090	29 20	AND #\$20	TEST FOR NEXT DIGIT
4092	C9 20	CMP #\$20	
4094	D0 F4	BNE \$408A	
4096	AD A2 49	LDA \$49A2	
4099	29 0F	AND #\$0F	
409B	81 0A	STA (\$0A,X)	
409D	A4 0A	LDY \$0A	
409F	C8	INY	
40A0	84 0A	STY \$0A	
40A2	D0 05	BNE \$40A9	
40A4	A4 0B	LDY \$0B	
40A6	C8	INY	
40A7	84 0B	STY \$0B	
40A9	AD F0 C0	LDA \$C0F0	
40AC	8D A2 49	STA \$49A2	
40AF	29 10	AND #\$10	TEST FOR NEXT DIGIT
40B1	C9 10	CMP #\$10	
40B3	D0 F4	BNE \$40A9	
40B5	AD A2 49	LDA \$49A2	
40B8	29 0F	AND #\$0F	
40BA	81 0A	STA (\$0A,X)	
40BC	A4 0A	LDY \$0A	
40BE	C8	INY	
40BF	84 0A	STY \$0A	
40C1	D0 05	BNE \$40C8	
40C3	A4 0B	LDY \$0B	
40C5	C8	INY	
40C6	84 0B	STY \$0B	
40C8	4C 1E 40	JMP \$401E	

FIGURE 3.16 Flow chart for the machine language program 2.



LISTING 3.5 BASIC program for digitizing and plotting data.

```

10 REM LOWSPEED A/D CONVERTER
12 REM APPLE II, R. HALLGREN, 10/28/81
24 FOR I=16384 TO 16586:READ D:POKE I,D:NEXT I
104 DIM Z(300):HOME:GOTO 1000
110 CALL 16384
111 HOME:VTAB 24
112 PRINT "THE DIGITIZED DATA IS BEING FORMATTED"
113 PRINT "FOR PLOTTING."
114 X=18944
115 FOR J=0 TO 255
120 V1=PEEK(X):V2=PEEK(X+1)
126 V3=PEEK(X+2):V4=PEEK(X+3)
128 X=X+4
130 IF V1>7 THEN V1=0
132 IF V1=0 THEN GOTO 140
134 V1=1
140 V$=SRT$(V1)+STR$(V2)+STR$(V3)+STR$(V4)
150 Z(J)=VAL(V$)/1000
160 NEXT J
200 HGR:HCOLOR=3
202 HPLOT 20,0 TO 20,150:HPLOT TO 279,150
208 HPLOT 18,0 TO 22,0
210 HPLOT 18,10 TO 22,10
212 HPLOT 18,20 TO 22,20
214 HPLOT 18,30 TO 22,30
216 HPLOT 18,40 TO 22,40
218 HPLOT 18,50 TO 22,50
220 HPLOT 18,60 TO 22,60
222 HPLOT 18,70 TO 22,70
224 HPLOT 18,80 TO 22,80
226 HPLOT 18,90 TO 22,90
228 HPLOT 18,100 TO 22,100
230 HPLOT 18,110 TO 22,110
232 HPLOT 18,120 TO 22,120
234 HPLOT 18,130 TO 22,130
236 HPLOT 18,140 TO 22,140
238 HPLOT 18,150 TO 22,150
240 HPLOT 4,47 TO 4,53:HPLOT 7,53
246 HPLOT 10,47 TO 10,53
248 HPLOT TO 14,53
250 HPLOT TO 14,47:HPLOT TO 10,47:HPLOT TO 7,103
262 HPLOT 14,97 TO 10,97:HPLOT TO 10,100:HPLOT TO 14,100
264 HPLOT TO 14,103:HPLOT TO 10,103
272 HPLOT 14,147 TO 10,147:HPLOT TO 10,153:HPLOT TO 14,153
274 HPLOT TO 14,147
280 HPLOT 30,148 TO 30,152
281 HPLOT 40,148 TO 40,152:HPLOT 50,148 TO 50,152
282 HPLOT 60,148 TO 60,152:HPLOT 70,148 TO 70,152
283 HPLOT 80,148 TO 80,152:HPLOT 90,148 TO 90,152
284 HPLOT 100,148 TO 100,152:HPLOT 110,148 TO 110,152
285 HPLOT 120,148 TO 120,152:HPLOT 130,148 TO 130,152
286 HPLOT 140,148 TO 140,152:HPLOT 150,148 TO 150,152
287 HPLOT 160,148 TO 160,152:HPLOT 170,148 TO 170,152
288 HPLOT 180,148 TO 180,152:HPLOT 190,148 TO 190,152
289 HPLOT 200,148 TO 200,152:HPLOT 210,148 TO 210,152
290 HPLOT 220,148 TO 220,152:HPLOT 230,148 TO 230,152
291 HPLOT 240,148 TO 240,152:HPLOT 250,148 TO 250,152
292 HPLOT 260,148 TO 260,152:HPLOT 270,148 TO 270,152
300 FOR J=0 TO 255
310 HPLOT J+20,150-(Z(J)*100)
320 NEXT J
999 END
1000 PRINT "PRESS 'RETURN' TO START A/D"
1010 K=PEEK(-16384)

```

LISTING 3.5 (Continued)

```
1012 POKE -16368,0
1014 IF K>127 THEN GOTO 1020
1016 GOTO 1010
1020 TEXT
1022 HOME
1024 VTAB 24
1026 PRINT "256 DATA POINTS ARE BEING DIGITIZED"
1028 GOTO 110
1099 END
2000 DATA 141,176,73,142,177,73,140,178,73,8,162,0,169,0,
          133
2002 DATA 10,169,74,133,11,169,0,141,160,73,169,78,141,
          161,73
2004 DATA 173,161,73,197,11,208,27,173,176,73,174,177,73,
          172
2006 DATA 178,73,40,96,234,234,234,234,234,234,234,234,234
2008 DATA 234,234,234,234,234,234,234,141,242,192,173,241,
          192
2010 DATA 41,128,201,128,208,247,173,240,192,141,162,73,41,
          128
2012 DATA 201,128,208,244,173,162,73,41,15,129,10,164,10,
          200
2014 DATA 132,10,208,5,164,11,200,132,11,173,240,192,141,
          162
2016 DATA 73,41,64,201,64,208,244,173,162,73,41,15,129,10,
          164
2018 DATA 10,200,132,10,208,5,164,11,200,132,11,173,240,
          192,141
2020 DATA 162,73,41,32,201,32,208,244,173,162,73,41,15,
          129,10
2022 DATA 164,10,200,132,10,208,5,164,11,200,132,11,173,
          240
2024 DATA 192,141,162,73,41,201,16,208,244,173,162,73,41
2026 DATA 15,129,10,164,10,200,132,10,208,5,164,11,200
2028 DATA 132,11,76,30,64
```

executing the program. Connect the battery to the circuit. The plot (See Figure 3.19) should show an exponential rise in voltage across the capacitor—exactly what you would expect. Figure 3.20 shows the plot of a .1 Hz sine wave digitized at a sampling rate of fifteen samples per second.

Ten-bit, High-speed A/D Converter

The AD571 (Analog Devices, Norwood, Massachusetts 02062) is a ten-bit, successive-approximation, A/D converter consisting of an internal D/A converter, voltage reference, clock, comparator, approximation register, and output buffers. With this device, an analog voltage can be converted into a quantity represented by 10 binary bits in approximately 25 microseconds, conversion rate not being a function of the magnitude of the input voltage. Figure 3.21 shows the block diagram and pin assignment for the AD571. This integrated circuit can operate from a positive supply voltage

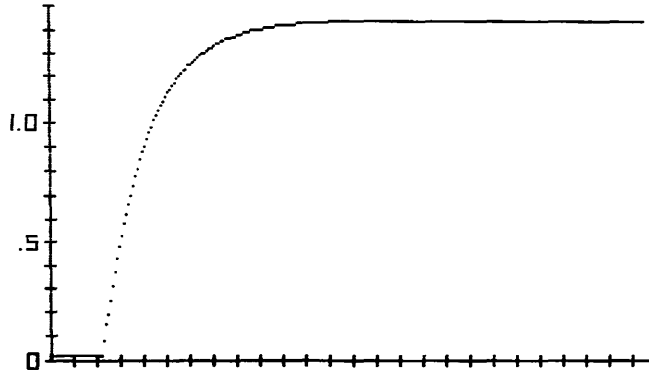


FIGURE 3.19 Voltage across the capacitor measured as a function of time.

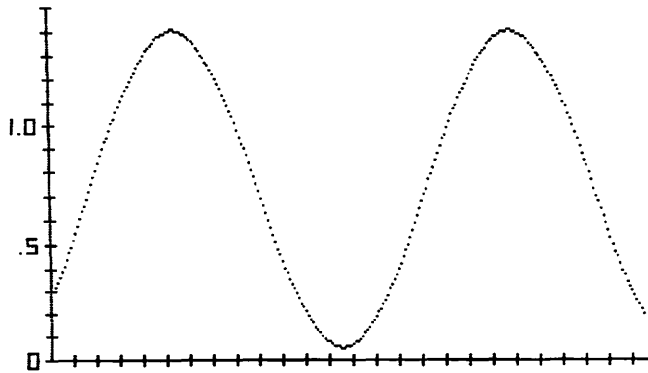
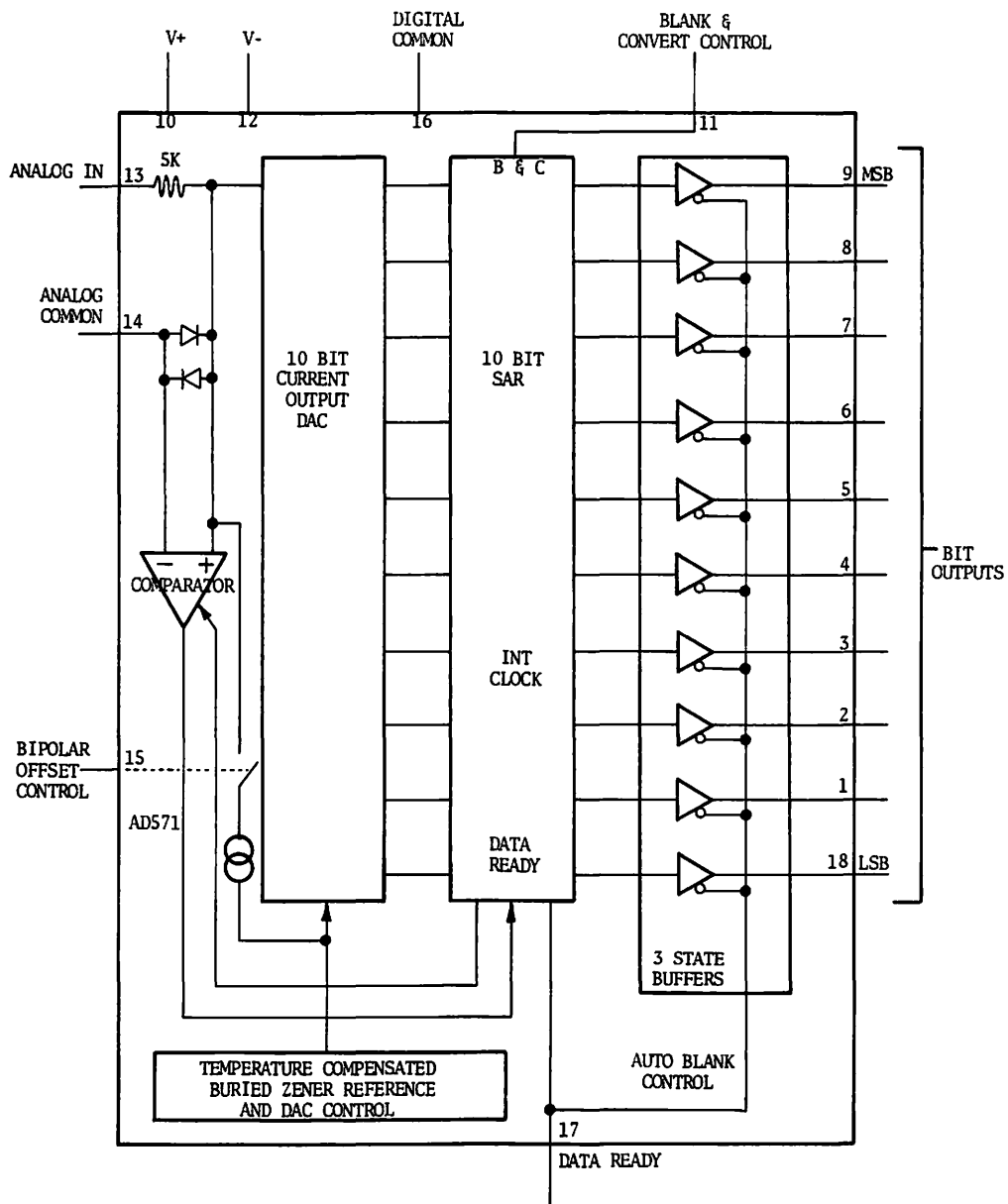


FIGURE 3.20 Sine wave (.1 Hz) digitized at 15 samples per second.

(pin 10) which can be set between + 5 and + 15 volts depending on whether you need TTL or CMOS compatability. The negative supply voltage (pin 12) is intended to be set at - 15 volts, but can be set as low as - 12 volts if necessary. The AD571 gives the designer the option of being able to configure the internal circuitry for either unipolar (0 to + 10 volts) or bipolar (- 5 to + 5 volts) operation. If your application involves only positive signals, then the greatest resolution can be realized by operating the circuit in the unipolar mode. Grounding the BIPOLAR OFFSET CONTROL (pin 15) puts the chip in the unipolar mode, while leaving this pin "open" puts the chip in the bipolar mode. Figure 3.22 shows a way to have logic control of pin 15. When operated in the unipolar mode, 0 volts gives an output code of 0000000000 and + 10 volts gives an output code of 1111111111. When operated in the bipolar

FIGURE 3.21 Block diagram of the Analog Devices AD571 A/D converter.



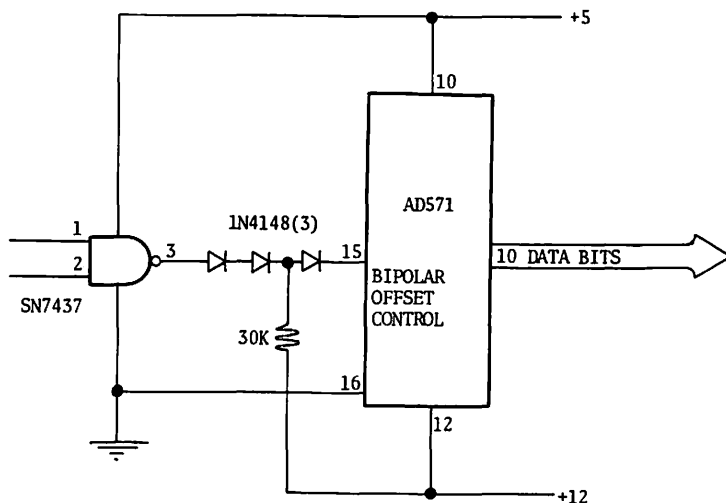


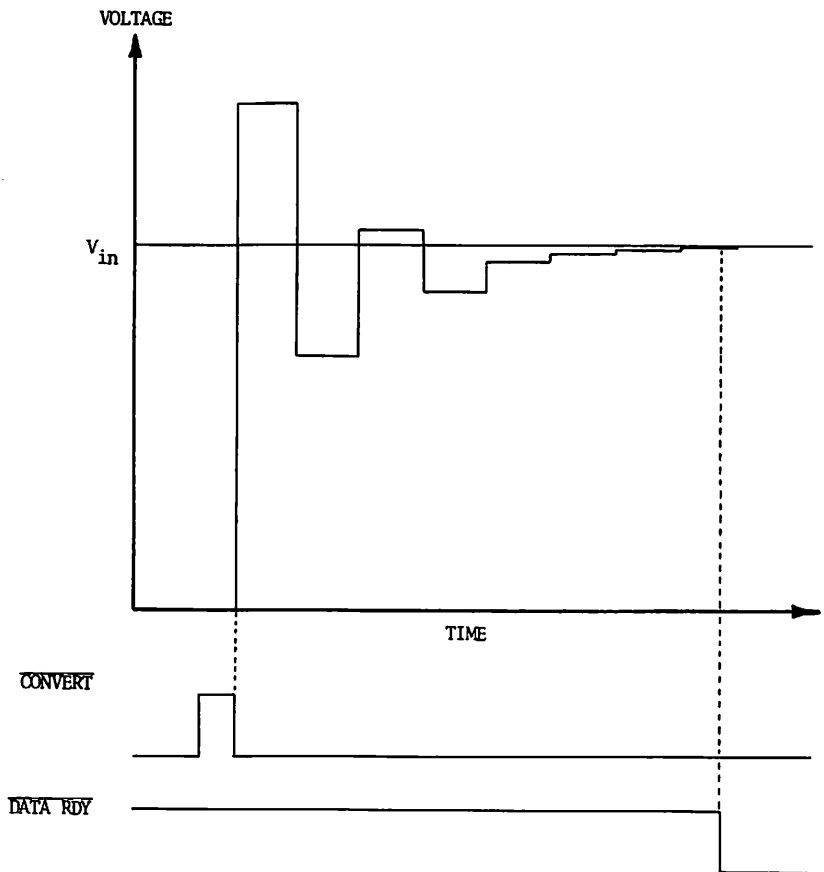
FIGURE 3.22 Bipolar Offset Control controlled by logic gate.

mode, 0 volts gives an output code of 1000000000, - 5 volts gives an output code of 0000000000, and + 4.99 volts gives an output code of 1111111111. For our application, bipolar operation was desired; consequently, pin 15 was left "open."

The AD571 uses a conversion scheme known as successive-approximation to achieve the high resolution and speed necessary for some computer applications. Successive-approximation involves comparing the unknown input voltage to a preset series of voltages that are binary fractions of the maximum voltage that may be measured. Initialization of the conversion process is accomplished by providing a pulse to the control line CONVERT*. Upon receipt of the convert pulse, the internal D/A converter is sequenced from its most significant bit to its least significant bit. The internal comparator determines whether the addition of each successively-weighted bit creates a voltage which is greater or less than the input voltage. If the voltage is greater, the bit is turned OFF; if the voltage is less, the bit is left ON. After the sequence is completed, the conversion is complete and the internal successive approximation register contains the 10-bit binary code which represents the input signal. Thus, for a circuit that measures an input varying between 0 and 10.0 volts, the comparisons would be made between voltage levels that varied in 9.766E-3 volt increments (10 volts divided by 1024 discrete levels). When an unknown input voltage is to be converted, the MSB of the D/A

converter's output ($\frac{1}{2}$ full scale) is turned ON and the input voltage is compared to 5 volts. If the input voltage is less than 5 volts, the MSB is turned OFF, the next bit ($\frac{1}{4}$ full scale) is turned ON, and the input voltage is compared to 2.5 volts. If the unknown voltage is greater than 2.5 volts, the second bit is left ON, the next bit ($\frac{1}{8}$ full scale) is turned ON, and the input voltage is compared to 3.75 volts. If the unknown voltage is less than 3.75 volts, the third bit is turned OFF, the next bit ($\frac{1}{16}$ full scale) is turned ON, and the input is compared to 3.125 volts. This process continues in order of descending bit weight until all the bits have been tried. The conversion process is thus completed and the 10-bit binary number representing the unknown input voltage is ready to be read by the computer. Figure 3.23 shows the timing relationship between the start of a conversion and the test sequence that we have described above.

FIGURE 3.23 Typical test sequence for a successive approximation A/D converter.



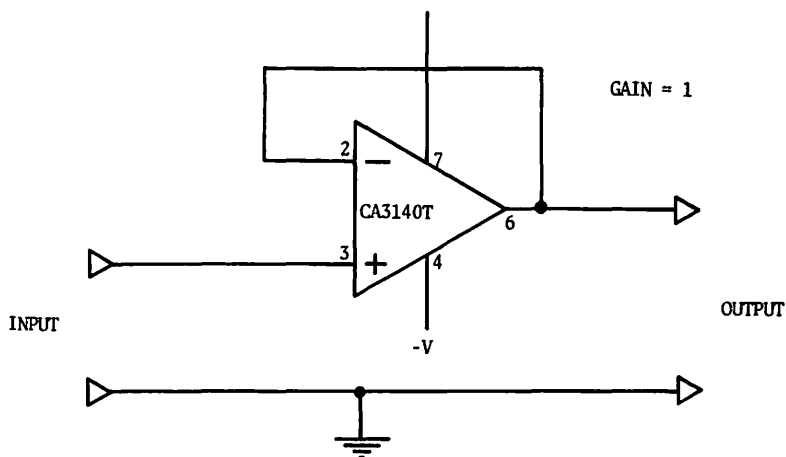


FIGURE 3.24 Noninverting, unity gain, buffer amplifier.

During the time that the conversion is being performed, the status line, DATA READY* (pin 17), goes high indicating that the converter is busy. Upon completion of the conversion, the status line goes low, bringing the three state buffers out of their “open” state and activating the bit output lines.

The typical input resistance of the AD571 is equal to 5000 ohms. Since this is low enough to excessively load some circuits, I would recommend that you use a buffer circuit to provide the necessary impedance matching. If your input voltage approaches the maximum capability of the A/D converter, then the simple unity gain, voltage-follower shown in Figure 3.24 should be adequate. If your input voltage is much less than the maximum the converter is capable of measuring, then the inverting amplifier shown in Figure 3.25 should be used. Notice that the gain of this amplifier configuration is variable, depending upon the ratio of the resistor combinations that are used. Appendix B has been provided to explain the operation of amplifier circuits, and to provide you with enough information so that you can design a specific circuit to satisfy your design requirements, if you should need to do so.

DETERMINING THE RESOLUTION While we have discussed the definition of resolution in an earlier section, I would like to make some additional comments which will help you

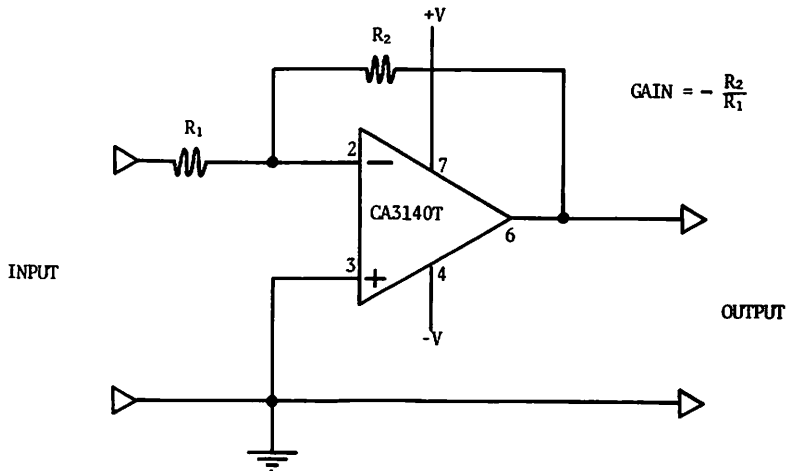


FIGURE 3.25 Inverting, adjustable gain, buffer amplifier.

determine the resolution that your particular application will require. In the process of A/D conversion, we are representing a continuous analog voltage in terms of a certain number of binary bits. We have already stated that the number of binary digits which are available to represent the analog signal is going to determine the resolution of the system, where resolution gives a measure of the converter's ability to distinguish between two voltages separated by some small voltage increment dR . For example, in Figure 3.26 we have the block diagram of a typical 8-bit A/D converter. Functioning correctly, this device will convert an analog input voltage, V_{in} , into a discrete voltage represented by the 8-bit binary number at the output. If a 0 volt signal is represented by a bit configuration equal to 00000000, and a + 10 volt signal is represented by a bit configuration equal to 11111111, then any magnitude of voltage between these extremes will be equal to a multiple of .039 volts (10 volts divided by 256 discrete levels). The converter will by necessity introduce an uncertainty—consequently, an inaccuracy—into the measurement, since the actual input voltage has to be represented by a discrete quantity at the output, and there are only so many bits which are available. It should be easy to see that we should be able to increase the accuracy of the A/D converter by increasing the resolution, that is, by increasing the number of binary output bits which are available to represent the input voltage. For example, if

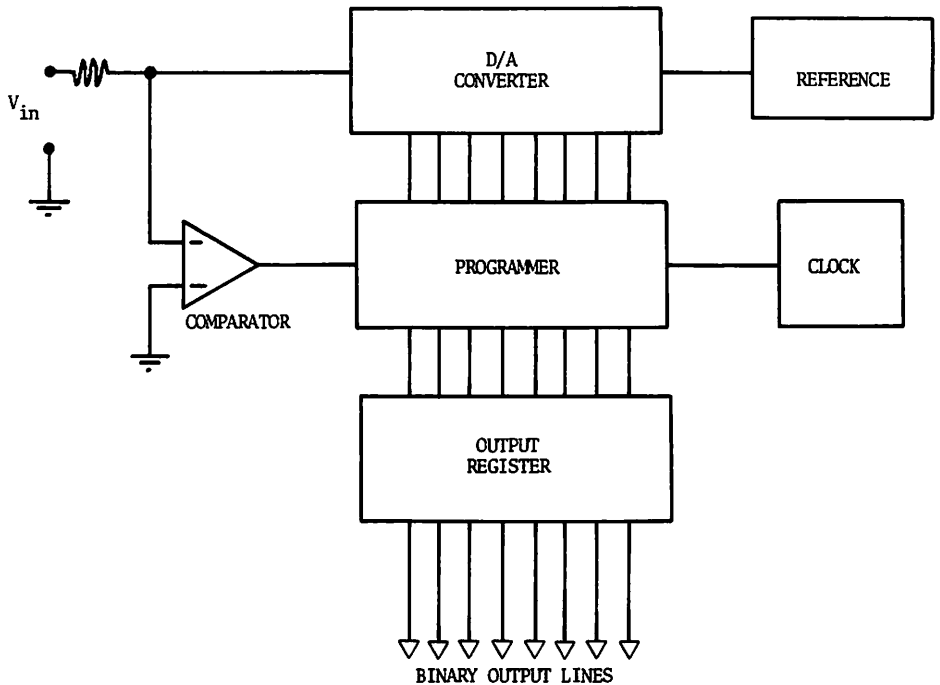


FIGURE 3.26 Block diagram of an 8-bit, successive approximation type A/D converter.

we were using a 10-bit A/D converter, we would expect the device to represent the input voltage as one of 1024 possible values. This would mean that a 0 volt signal would be represented by 0000000000, a +10 volt signal would be represented by 1111111111, and values in between these two extremes would be assigned values in steps of .00977 volts (10 volts divided by 1024 discrete levels). This is obviously an improvement over the 8-bit device.

Since increased resolution usually means increased circuit cost, increased conversion time, and increased memory requirements, we need to wisely choose the number of output bits that will satisfy our particular design requirements without providing excessive capability. How then do we determine the resolution (dR) which we will need to meet the requirements of our application? Suppose we needed to determine the magnitude of the input voltage and be reasonably certain that the apparent magnitude was within 1 percent of the actual magnitude. To achieve this, we would have to have a resolution of at least 1 part in 100. To attain

this resolution, we will need at least 7 binary bits at the output. If both positive and negative quantities are to be represented, the converter will need one extra bit to represent the sign, making a total of 8 bits. While 1 percent sounds pretty good, remember that this is in reference to the maximum voltage that the converter can handle; this means that for a maximum input voltage of 10 volts, you will be able to distinguish signals that are .1 volt apart. If the input voltage is varying between 0 and + 1 volt, you still will have a resolution equal to .1 volt, but your accuracy will have decreased to 10 percent ($1 / .1$). Keep in mind that for this discussion, I am assuming that the primary source of inaccuracy in our system is due to a lack of resolution. To determine a measure of your system's ability to differentiate between the magnitude of two voltages, follow these steps:

1. Determine the magnitude of the signal that you will be measuring.
2. Determine the maximum resolution of your system by dividing the magnitude of the output signal which corresponds to all 1's, by the number of discrete voltage steps which the system is capable of providing. Do not count sign bits.
3. Divide the quantity from Step 2 by the quantity from Step 1 and multiply by 100 to get the accuracy expressed in percent.

You should realize that in Step 1 you were concerned with the magnitude of the signal that you would be measuring, not the magnitude of the signal that you were capable of measuring. The truncating and rounding procedure that the A/D converter performs to get the input signal to fall into predetermined voltage slots has the effect of causing the accuracy to vary as a function of the magnitude of the input signal (even though resolution remains constant). For this reason, you should always be aware of the magnitude of the signal which you will be measuring; often, the system you choose will have to be a compromise between accuracy and dynamic range.

Figure 3.27 shows a schematic diagram of a high-speed A/D converter for the Apple II, and Figure 3.28 shows a schematic diagram of a crystal controlled time base oscillator which we will be using to determine the sampling rate. IC1 is an analog multiplexer which is used to select one of eight possible input signals. Control of input signal selection is accomplished by latching a given bit pattern at the output pins of the quad type D flip-flop

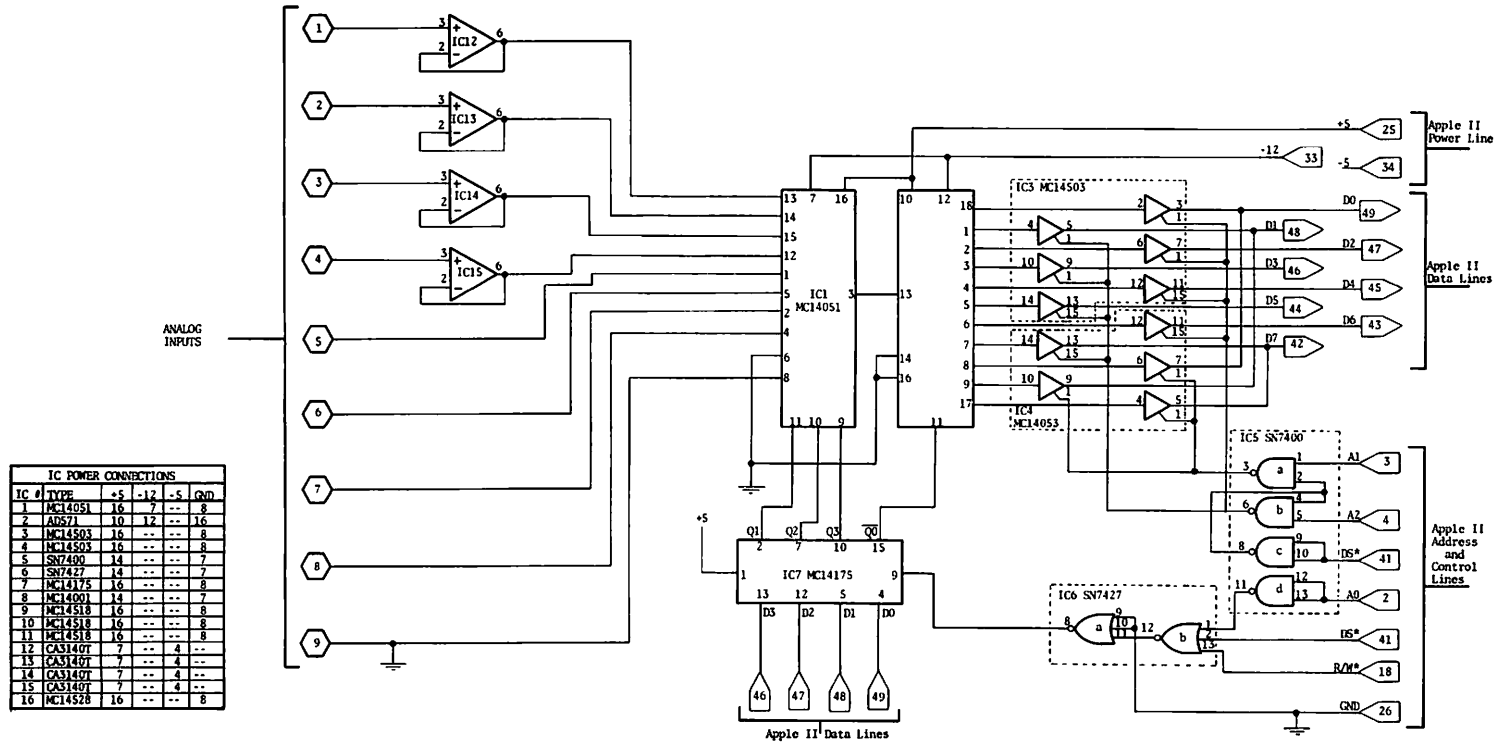


FIGURE 3.27 High-speed A/D converter for the Apple II.

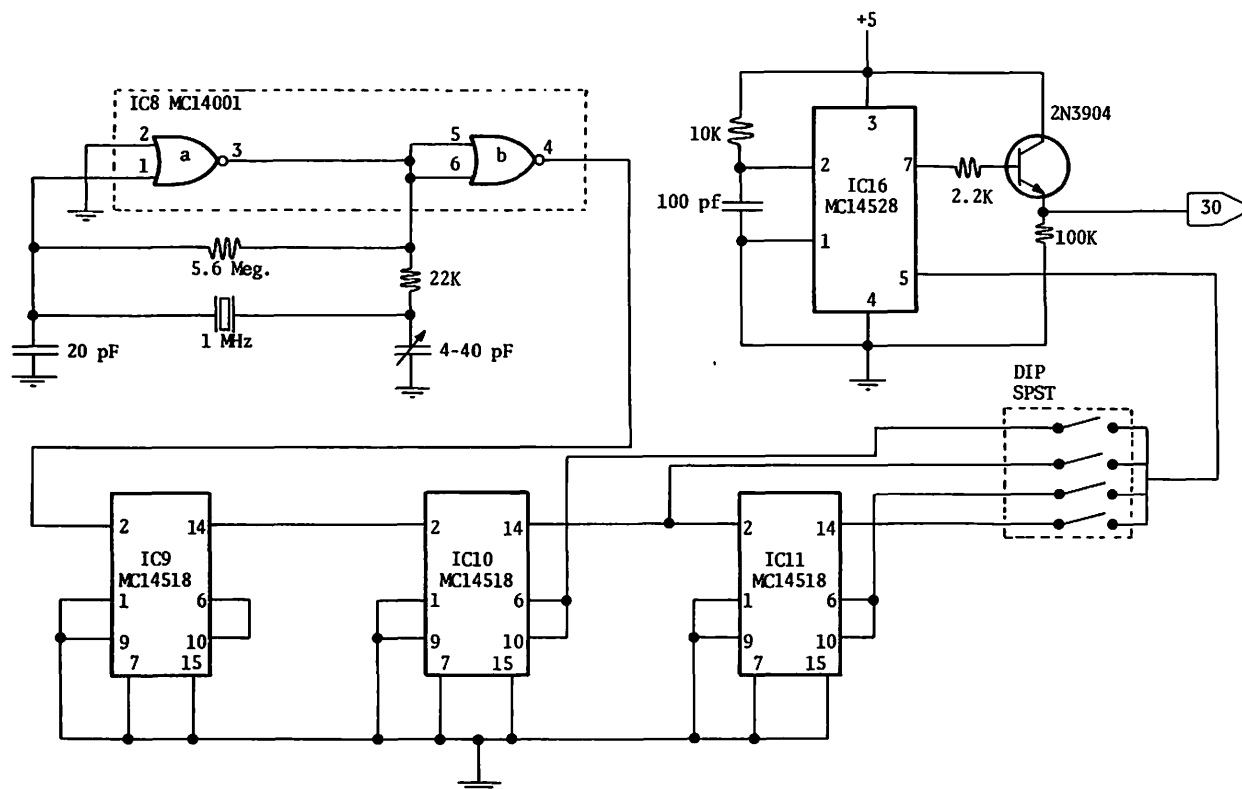


FIGURE 3.28 Crystal-controlled time base oscillator.

(IC7). You should notice that one of the lines from the flip-flop is connected to the A/D converter. This is the control line (CONVERT*), discussed earlier in the overview of the AD571 A/D converter (IC2), which is used to signal that a conversion is to begin. Data lines D0, D1, and D2 then control the selection of the analog input signal, and D3 controls the initiation of the conversion process. If you executed the following BASIC commands, you would convert the analog signal present at input 2 into a digital quantity.

100 POKE - 16143,09

110 POKE - 16143,01

IC12, IC13, IC14, and IC15 are operational amplifiers which are configured as unity gain, noninverting buffers. They are used to isolate an input signal from the input of the AD571 to avoid problems associated with the relatively low input resistance of this device. Four unbuffered inputs are also provided for input signals which do not need to be isolated (having a low value of series output resistance), or which require an additional stage of amplification.

All the data and status lines to the Apple II have been isolated through use of the MC14503 three-state buffers (IC3 and IC4). Since the Apple II is an eight-bit machine, the 10 bits from the AD571 are transferred to memory in a two-step operation. The DATA RDY* line and the two most significant bits are read into the machine through the use of an LDA \$C0F2 machine language command. When the AD571 signals the Apple II that a conversion has been completed (by dropping DATA RDY* to 0 volts), the following sequence of operations should be performed:

1. The most significant two bits of data (appearing on D0 and D1 of the data bus) should be stored in memory.
2. The least significant eight bits should be transferred into the computer through the use of an LDA \$C0F4 machine language command and stored in memory.
3. The Apple II should then either process the data or return to gather more data.

IC8 and the 1 MHz crystal provide a stable pulse train which is used to determine the sampling frequency. IC9, IC10, and IC11 divide the 1 MHz pulse train by multiples of ten so that sampling rates of 1, 10, 100, and 1000 samples per second can be obtained. IC16 is a monostable multivibrator which is used to provide an inverted output pulse, width being independent of the output frequency, to the Apple II I/O interrupt input. The 2N3904 transistor is configured as an emitter follower to provide the current drive requirements necessary to drive the interrupt line.

Once you have constructed the interface, connect it to the Apple II and perform the following tests to make sure that it is working correctly:

1. If you execute the following BASIC statements, pin 7 and pin 2 on IC7 should go to +5 volts. Pin 10 and pin 15 should remain at 0 volts.
100 POKE - 16143,09
2. If you execute the following BASIC statements, pin 2 on IC7 should go to 0 volts. Pin 7 should remain at +5 volts. Pin 10 and pin 15 should remain at 0 volts.
100 POKE - 16143,01
3. If you now apply an analog signal to input IN1, you should see that signal at the output (pin 3) of the multiplexer (IC1).

If you have an oscilloscope, you can perform the following tests and adjustments:

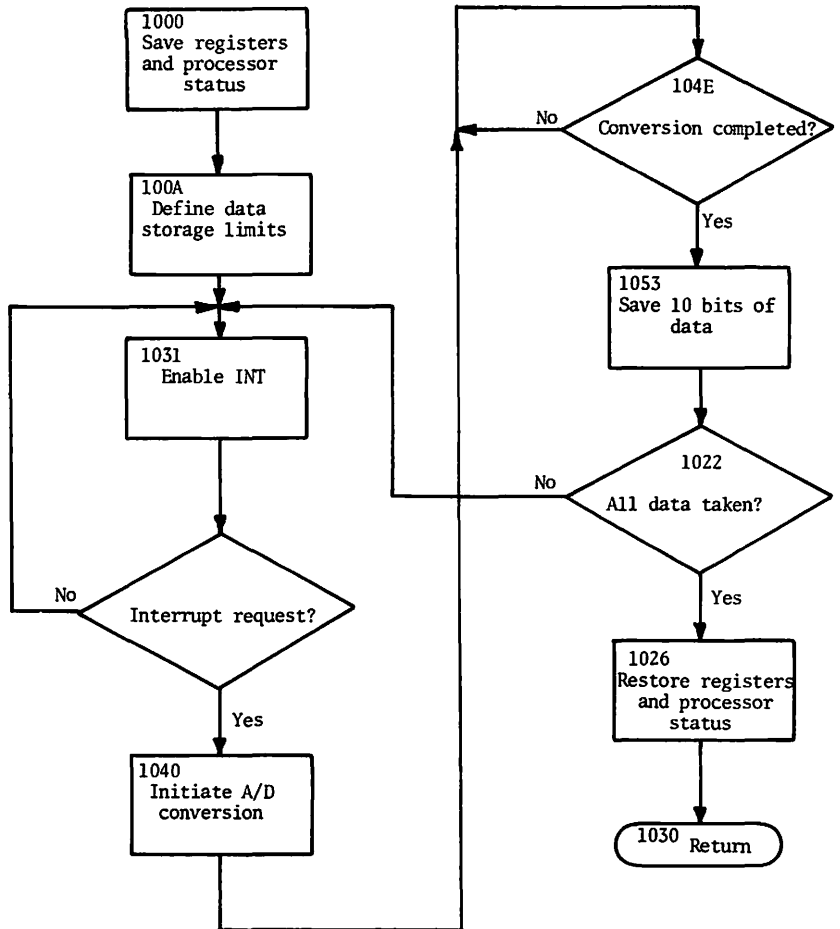
1. If you execute the following BASIC statements, you should see the DATA RDY* line periodically go from 0 to +5 volts.
100 POKE - 16143,09
110 POKE - 16143,01
140 FOR I=1 TO 100
142 NEXT I
150 GOTO 100
2. Adjust the 4-40 pf trimmer capacitor in the crystal oscillator until the output of IC8 is equal to 1 MHz. The IRQ* line should reflect the sampling frequency that you have selected.
3. Measure the pulse width of the output of IC16. It should be approximately equal to .1 milliseconds.

The software portion of the high speed A/D converter interface was handled in two parts:

1. A routine written in Applesoft BASIC takes the binary representation of the analog data and converts it into a decimal number which then can be displayed, or stored for future reference.
2. A machine language routine which was written to control the AD571 and to provide high speed transfer of the binary data into the Apple II.

Figure 3.29 shows a flow chart for the machine language routine, and Listing 3.6 gives the actual program with comments. Upon entering the subroutine, all necessary registers are saved to allow

FIGURE 3.29 Flow chart for the machine language routine for controlling the high-speed A/D converter.



LISTING 3.6 Machine language program for the high-speed A/D converter.

```

1000 8D FD 10      STA $10FD      SAVE REGISTERS
1003 8C FF 10      STX $10FE
1006 8C FF 10      STY $10FF
1009 08           PHP           SAVE PROC STATUS
100A A9 00         LDA #$00      LOAD DATA START ADR
100C 85 0A         STA $0A
100E A9 11         LDA #$11
1010 85 0B         STA $0B
1012 A9 00         LDA #$00      LOAD DATA STOP ADR
1014 8D FB 10      STA $10FB
1017 A9 40         LDA #$40
1019 8D FC 10      STA $10FC
101C A2 00         LDX #$00      CLEAR X REGISTER
101E 78           SEI           DISABLE INTERRUPT
101F AD FC 10      LDA $10FC      END OF DATA STORAGE?
1022 C5 0B         CMP $0B
1024 D0 0B         CNE $1031
1026 AD FD 10      LDA $10FD      RESTORE REGISTERS
1029 AE FE 10      LDX $10FE
102C AC FF 10      LDY $10FF
102F 28           PLP           RESTORE PROC STATUS
1030 60           RTS           RETURN TO BASIC
1031 58           CLI           ENABLE INTERRUPT
1032 EA           NOP
1033 4C 1E 10      JMP $101E
1036 00           BRK
1037 00           BRK
1038 00           BRK
1039 00           BRK
103A 00           BRK
103B 00           BRK
103C 00           BRK
103D 00           BRK
103E 00           BRK
103F 00           BRK
1040 A9 09         LDA #$09      BEGIN CONVERSION
1042 8D F1 C0      STA $C0F1
1045 A9 01         LDA #$01
1047 8D F1 C0      STA $C0F1
104A AD F2 C0      LDA $C0F2      GET DATA
104D 2A           ROL           ROTATE LEFT
104E B0 FA         BCS $104A
1050 6A           ROR           ROTATE RIGHT
1051 29 03         AND #$03      SAVE 3 BITS
1053 81 0A         STA ($0A,X)   SAVE DATA
1055 A4 0A         LDY $0A
1057 C8           INY           INCREMENT STORAGE ADR
1058 84 0A         STY $0A
105A D0 05         BNE $1061
105C A4 0B         LDY $0B
105E C8           INY
105F 84 0B         STY $0B
1061 AD F4 C0      LDA $C0F4      GET DATA
1064 81 0A         STA ($0A,X)   SAVE DATA
1066 A4 0A         LDY $0A
1068 C8           INY
1069 84 0A         STY $0A
106B D0 05         BNE $1072
106D A4 0B         LDY $0B
106F C8           INY
1070 84 0B         STY $0B
1072 A5 45         LDA $45
1074 40           RTI           RETURN FROM INTERRUPT

```

a successful return to the BASIC calling routine. The program then defines the data storage space and initializes the system I/O bus. The interrupt request line is then activated and the program loops until an interrupt request is made. The request for an interrupt comes from the output of the crystal time base oscillator, the frequency being dependent upon the position of switch S1. After

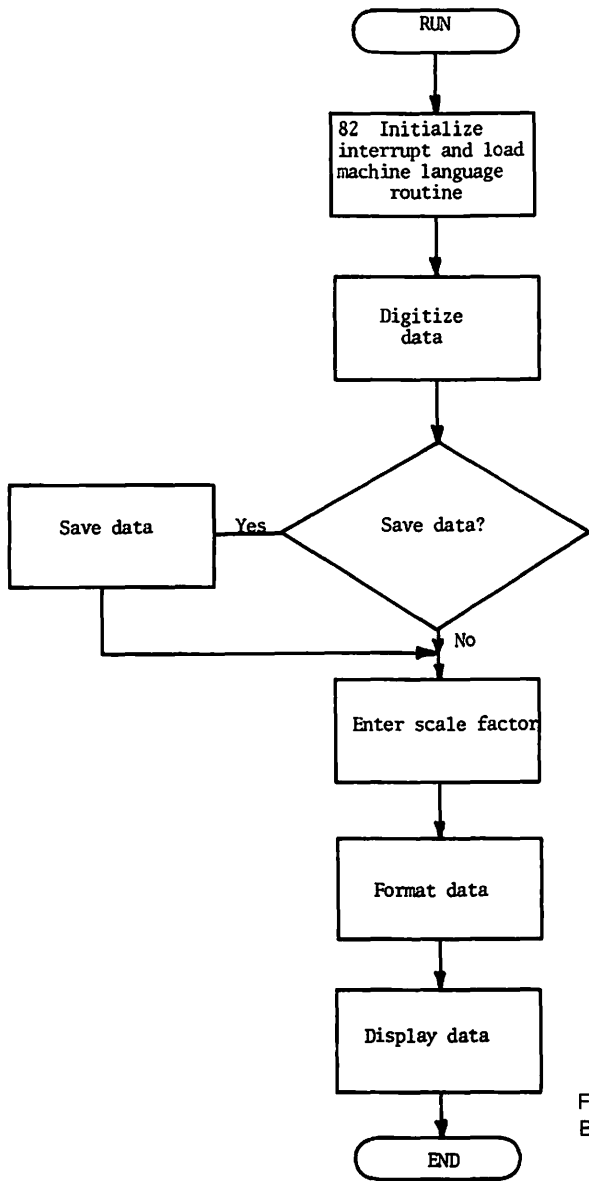


FIGURE 3.30 Flow chart for the BASIC routine.

receiving an interrupt request, the program initiates an A/D conversion, and waits until the AD571 signals that the conversion is completed. The 10 data bits are then stored in memory and the program tests to see if the data storage is filled. If all the data storage has been filled, the program restores all the necessary registers and returns to the BASIC calling routine. If all the data storage has not been filled, the program returns to activate the interrupt system and to take more data.

Figure 3.30 shows a flow chart for the BASIC program, and Listing 3.7 gives the actual program with comments. The BASIC routine first loads the machine language routine by using the READ and POKE statements. Then the machine language routine is called to obtain a block of digitized analog data. When the

LISTING 3.7 BASIC program for the high-speed A/D converter.

```

80 HIMEM:4096
82 POKE 1022,64:POKE 1023,16:POKE 1016,72:POKE 1017,8
83 POKE 1018,120:POKE 1019,40:POKE 1020,104:POKE 1021,96
84 CALL 1016
90 FOR I=4096 TO 4212:READ D:POKE I,D:NEXT I
100 HOME:VTAB 12:PRINT "DATA BEING DIGITIZED"
102 CALL 4096
105 HOME:VTAB 12:PRINT "WHAT IS THE TIME SCALE?"
106 GET K:HOME:VTAB 12
107 PRINT "DO YOU WANT TO SAVE THE DATA?":INPUT K$
108 IF K$="N" THEN GOTO 120
109 IF K$="Y" THEN GOTO 115
110 GOTO 106
115 D$=""
116 PRINT D$;"BSAVE DATA($1100),A$1100,L$2F00
120 HOME:HGR2:HCOLOR=3:X=4352
130 FOR J=0 TO 240/K
135 X=X+2:IF X>16382 THEN GOTO 230
140 V1=PEEK(X):V2=PEEK(X+1)
142 IF V1>1 THEN GOTO 200
150 Z=-((511-(V1*256+V2))/5)
160 H$PLOT K*J,90-Z
170 NEXT J
190 HGR2:HCOLOR=3:GOTO 130
200 Z=((V1-2)*256+V2)/5
210 H$PLOT K*J,90-Z:GOTO 170
230 TEXT:END
1000 DATA 141,253,16,142,254,16,140,255,16,8,169,0,133,
1002 DATA 10,169
1002 DATA 17,133,11,169,0,141,251,16,169,64,141,252,16,
1004 DATA 162,0
1004 DATA 120,173,252,16,197,11,208,11,173,253,16,174,
1006 DATA 254,16
1006 DATA 172,255,16,40,96,88,234,76,30,16,0,0,0,0,0,0,
1008 DATA 0,0,0,0
1008 DATA 169,9,141,241,192,169,1,141,241,192,173,242,
1010 DATA 192,42,176
1010 DATA 250,106,41,3,129,10,164,10,200,132,10,208,5,
1012 DATA 164,11,200
1012 DATA 132,11,173,244,192,129,10,164,10,200,132,10,208
1014 DATA 5,164,11,200,132,11,165,69,64

```

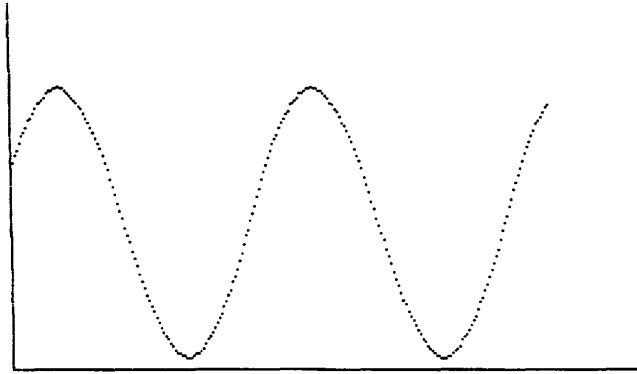


FIGURE 3.31a 10 Hz sine wave digitized at 1000 samples per second.

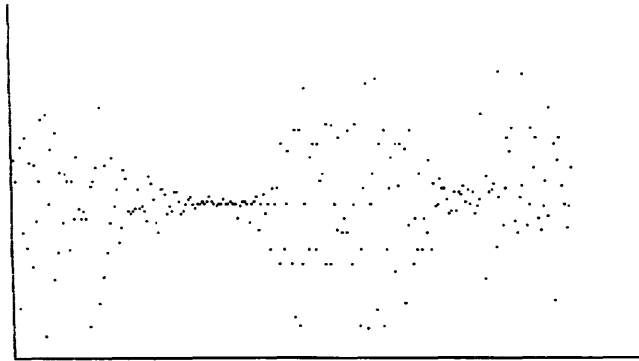


FIGURE 3.31b Music digitized at 1000 samples per second.

program returns from the machine language routine, the BASIC program then reconstructs the data into a voltage and plots the voltage on the video monitor. Figure 3.31a shows the results of digitizing a 10 Hz sine wave at a sampling rate of 1000 samples per second. Figure 3.31b shows the results of digitizing music at a sampling rate of 1000 samples per second.

The program is configured so that you can save the digitized data. This will allow you to recover the data for further analysis. (In Chapter 8 we will use this feature to analyze a complex waveform and determine frequency components.) The program also allows you to expand the displayed data. Figure 3.32a shows a 100 Hz sine wave digitized at a sampling rate of 1000 samples per second and not expanded. Figure 3.32b shows the same block of data displayed with an expansion factor of 5.

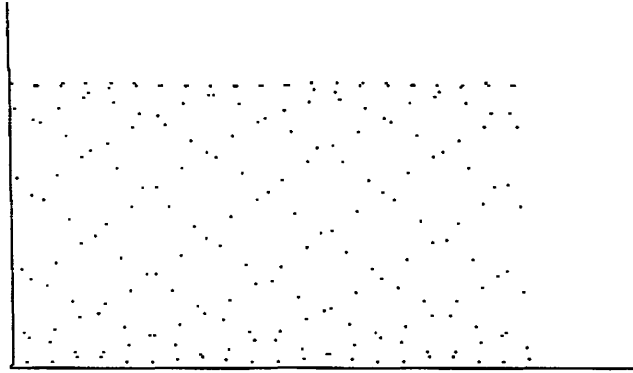


FIGURE 3.32a 100 Hz sine wave digitized at 1000 samples per second ($\times 1$).

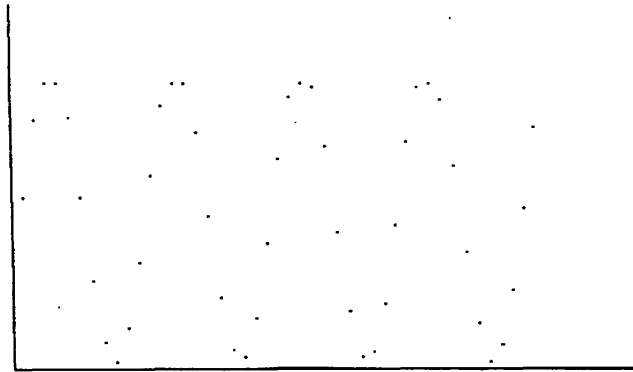


FIGURE 3.32b 100 Hz digitized at 1000 samples per second ($\times 5$).

4

DIGITAL-TO-ANALOG CONVERSION

Digital-to-analog (D/A) converters are devices that take a digital input—usually from the data bus of a computer—and convert it into an analog signal. While the analog signal is often thought of as being a continuous quantity, the output of the D/A converter will actually consist of a series of discrete and easily measurable voltages. Resolution, the voltage difference between steps, will depend upon the number of bits that the D/A converter can handle.

The Analog Devices AD558 DACPORT is an 8-bit digital-to-analog converter that includes a precision voltage reference, a microprocessor interface, a data latch, and an output amplifier—all on a single monolithic chip. Thin film silicon-chromium resistors are used in a current-driven ladder to provide an analog voltage that is stable over a large temperature range. Figure 4.1

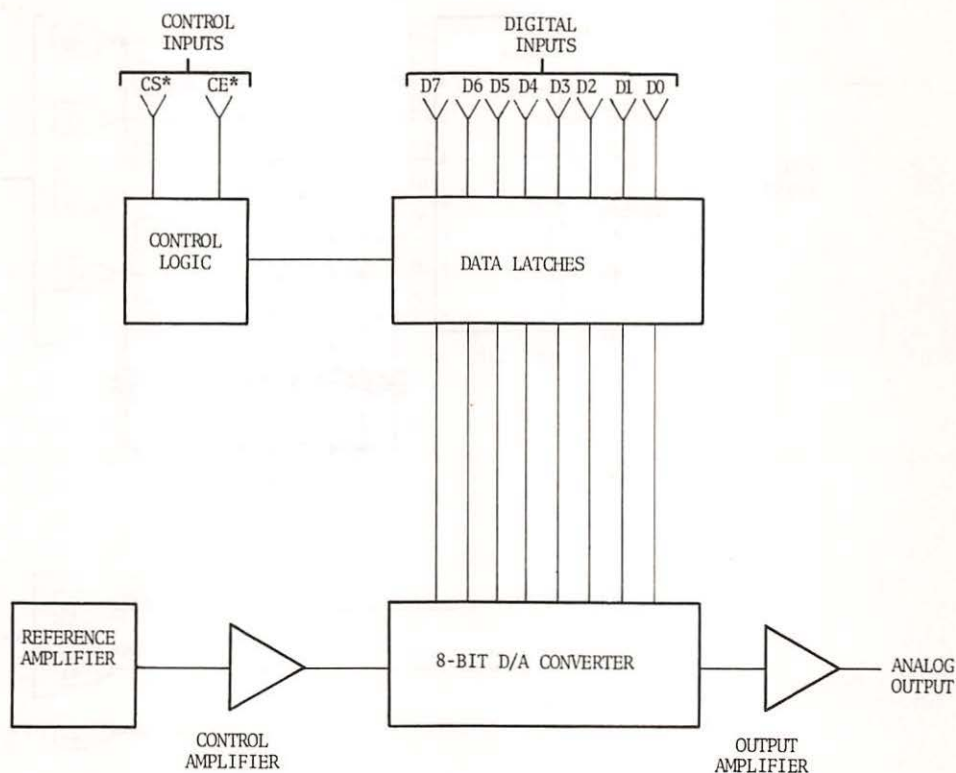


FIGURE 4.1 Functional block diagram of the AD558.

shows a functional block diagram of the AD558. The data latch is connected to the computer data bus, data being accepted when both control inputs, CE^* and CS^* , are low. When either control input returns high, the 8-bit data word is latched and the analog output is unaffected by further activity on the data lines. For a maximum output voltage of 2.56 volts, the resolution of the converter will be equal to 0.1 volts; that is, a change in the least significant digit will result in .01 volt change in the output voltage.

Figure 4.2 shows the schematic diagram for a D/A converter interface for the Apple II. Data is strobed into the converter by writing data to the appropriate I/O slot. The analog output will then remain stable until new data is written.

After you have finished building the interface, perform the following tests to make sure that it is working correctly:

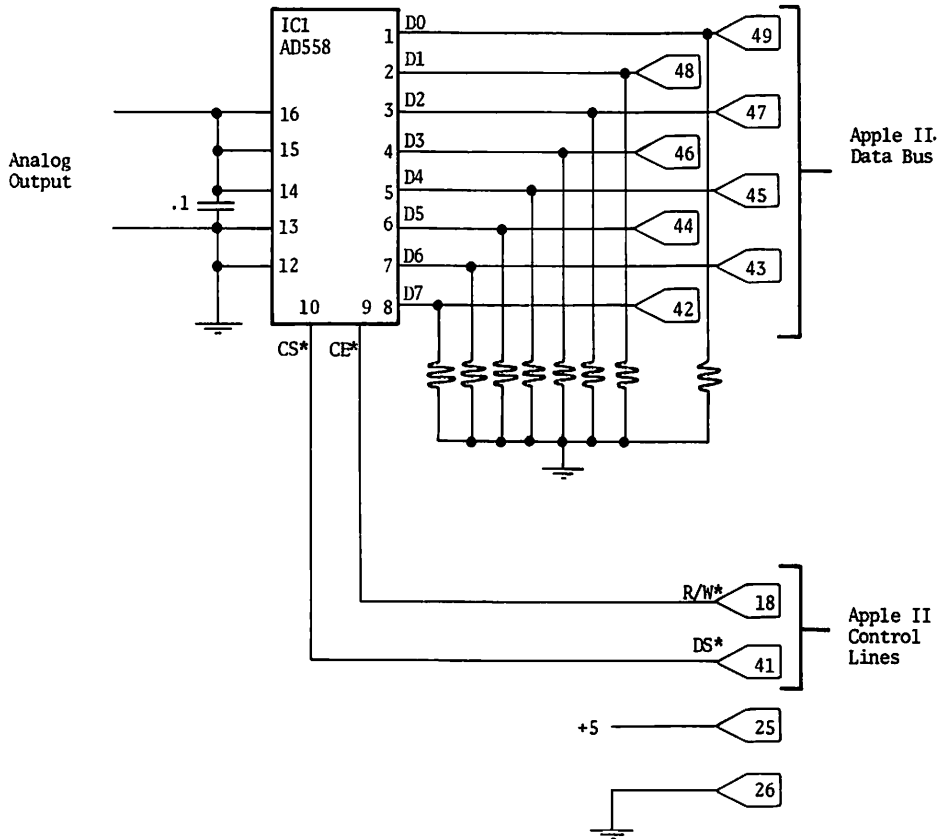


FIGURE 4.2 Interface schematic for the AD558 D/A converter.

If you execute the following BASIC statements, you should see the analog output ramp from 0 volts to 2.55 volts.

```

110 FOR I= 0 TO 255
120 POKE - 16143,I
130 NEXT I
199 END

```

Digital-to-analog converters are commonly used in digitally-controlled power supplies for automated test equipment, digital generation of analog waveforms, and digital control of automatic process control systems. I will present examples of two automatic test equipment systems that might be constructed to perform quality control testing in a manufacturing facility.

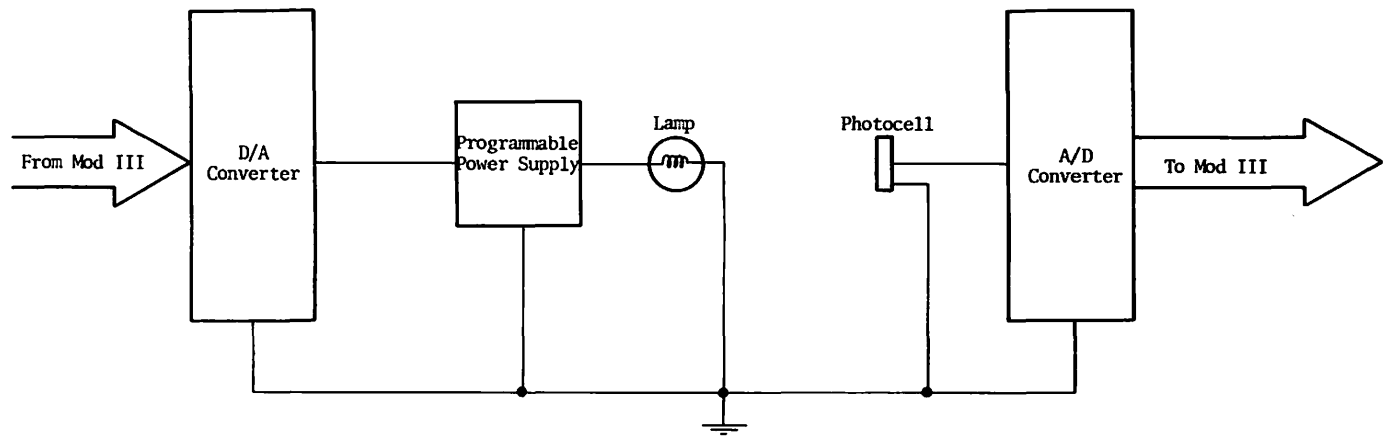
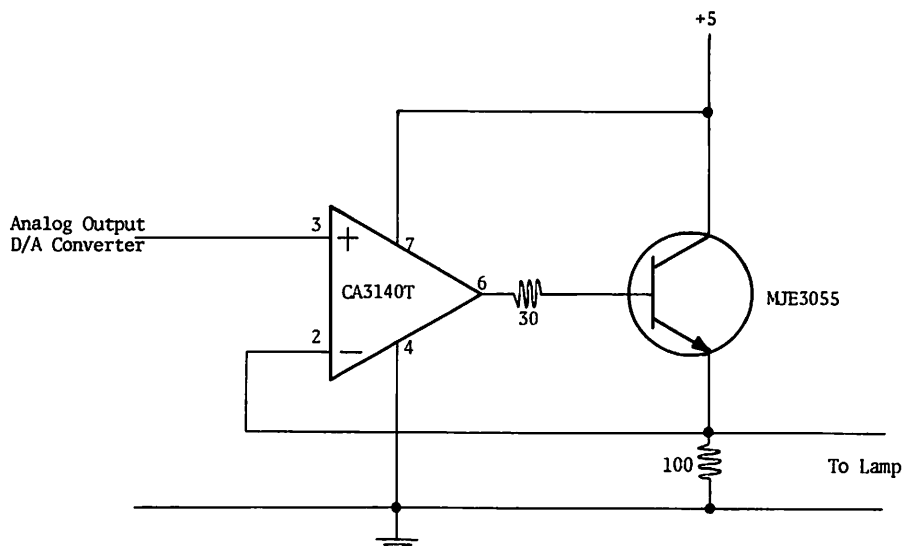


FIGURE 4.3 Block diagram of proposed testing arrangement.

The first example involves cycling an incandescent lamp through a number of voltages to determine if the lamp intensity is acceptably high. The lamp might be used in a simple colorimeter for a chemistry laboratory, and its intensity may need to be above a certain level to ensure reliable operation. Since checking each lamp could be very time consuming, it would be desirable to have the computer automatically increase the lamp voltage and record the lamp intensity at each voltage. At the end of the test, the computer could then make a decision regarding the acceptability of the lamp. The output of the AD558 is already buffered, but by itself cannot provide enough current to drive an incandescent lamp. What we will do is build a voltage programmable power supply, one whose output voltage is some multiple of the input voltage supplied by the A/D converter.

Figure 4.3 shows a block diagram of the proposed testing arrangement. The data bus from the Apple II will supply a binary number that will be converted by the D/A converter into an analog voltage. This voltage will then be used to control the lamp voltage. Lamp intensity will be monitored by a photocell connected to the high speed A/D converter which we discussed in Chapter 3. Figure 4.4 shows a schematic diagram of the programmable power supply. Figure 4.5 shows a flow chart for an Applesoft BASIC program to cycle the lamp voltage from 0 to 2.0 volts. Listing 4.1 shows the actual program with comments.

FIGURE 4.4 Schematic diagram of lamp power supply.



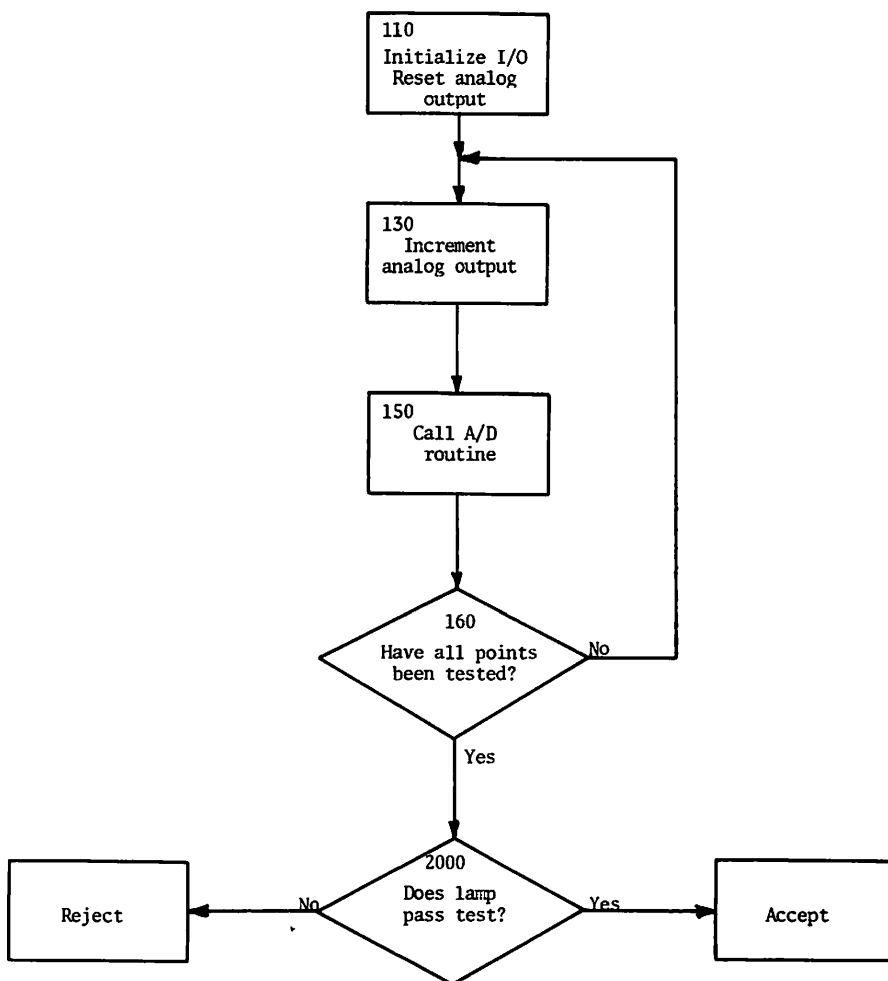


FIGURE 4.5 Flow chart of BASIC program for automatic testing of incandescent lamps.

LISTING 4.1 BASIC routine for lamp test example.

```

100 REM LAMP TEST ROUTINE
102 REM R. HALLGREN, 10-27-81
104 REM APPLE II
120 FOR I=1 TO 10
130 POKE -16143,20*I
140 FOR J=1 TO 500:NEXT J
150 GOSUB 1000
160 NEXT I
170 GOSUB 2000
199 END
1000 REM ANALOG-TO-DIGITAL SUBROUTINE
1999 RETURN
2000 REM LAMP PASS/NOT PASS SUBROUTINE
2999 END
  
```

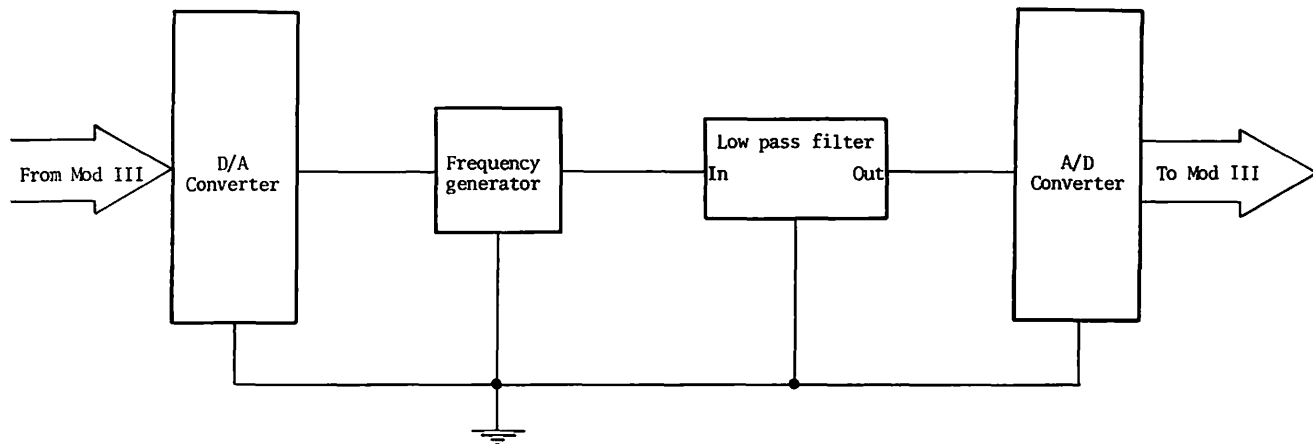


FIGURE 4.6 Block diagram for low-pass filter testing arrangement.

The second example involves testing the frequency response of a low pass filter. We will use the D/A converter to directly control the output frequency of a B&K 3010 Function Generator (Dynascan Corporation, 6460 W. Cortland Street, Chicago, Illinois 60635). This instrument has an input jack which allows an external voltage to directly control its output frequency. Figure 4.6 shows a block diagram of the proposed testing arrangement. Again, the data bus from the Apple II will supply a binary number which will be converted by the D/A converter into an analog voltage. This voltage will then be used to control the frequency of the function generator. The output of the low pass filter could be monitored by connecting it to the high speed A/D converter, and the computer could determine if the 3db cutoff frequency is within acceptable limits.

For both of the examples that we have looked at, the goal was to reduce the amount of personnel time required to perform a routine test operation. This was accomplished through the use of a computer-controlled test protocol.

5

UTILIZATION OF THE APPLE II IN SERIAL APPLICATIONS

HIGH RESOLUTION DIGITAL PLOTTING

Utilization of personal computers, such as the Apple II, in business and research applications has met with considerable success. Not only do they function as computational tools but, when used in conjunction with peripheral devices, they can perform complex functions at reduced expense. The research laboratory often deals with complex, time-dependent waveforms. This type of data can be easily digitized (see Chapter 3) and analyzed (see Chapter 8), but a major problem arises when a permanent copy of the data is required. Printing a column of numbers does little to convey to the researcher the nature of the data, especially several months after the data are taken. Plotting with a terminal such as the Decwriter II is possible, but lacks the necessary resolution for

many applications. The HIPILOT Digital Plotter, manufactured by Houston Instruments, gives the researcher a cost effective means of obtaining quality digital plots. The plotter accepts an 8½ x 11 inch sheet of paper and allows plotting within a 7 x 10 inch boundary. Reversible stepper motors are used to give bidirectional steps of either 200 or 100 steps per inch, amounting to a resolution of .005 or .01 inches per step respectively. The plotter comes with a standard RS-232C serial interface, so interfacing it to the computer is a relatively straightforward task.

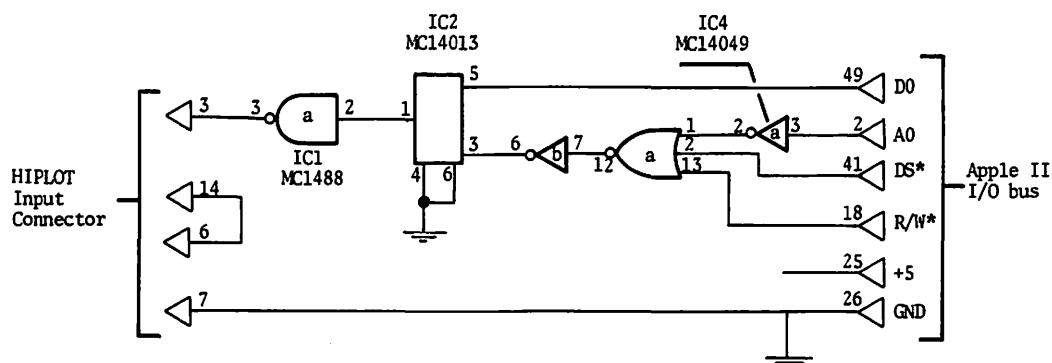


FIGURE 5.1 HIPILOT and Apple II interface connections.

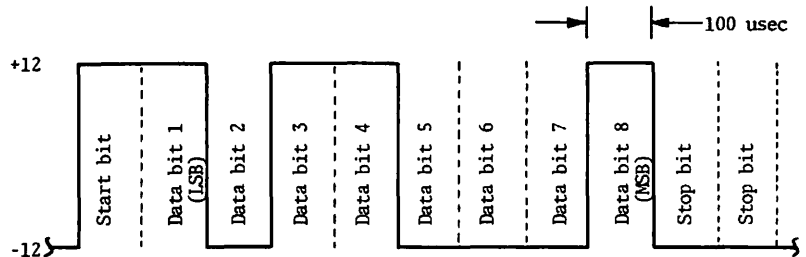
We have already discussed the serial data format in Chapter 2, and we have examined the necessary hardware (see Figure 2.9) and software (see Listing 2.1) to implement a TTL to RS232-C conversion. If you have not already built this circuit, you should do so now. Since we will be using the power supply in the Apple II, you will not need to provide any additional voltages. Once you have constructed the interface, connect it to the plotter as shown in Figure 5.1 and insert the circuit board into I/O slot 5. Perform the following tests to make sure that it is working correctly:

1. If you execute the following BASIC statement the output of IC1 should go to +10 volts.
POKE - 16175,0
2. If you execute the following BASIC statement the output of IC1 should go to -10 volts.
POKE - 16175,1
3. Load and execute the BASIC program shown in Listing 5.1. If everything is working correctly, the plotter pen should begin mov-

[illegible]

Being able to accomplish the transfer of information in an RS232-C format is only the first step towards using the plotter. The HILOT consists of two stepper motors which control the X,Y position of the plotter pen. Control of the pen position, and all other plotter functions, is accomplished by sending specific control characters to the device. Figure 5.3 shows the axis system for the plotter pen, and Table 5.1 shows the character that produces an increment of the pen along the given vector. Also included are

FIGURE 5.2 Bit pattern and spacing of RS232-C output from the demonstration program



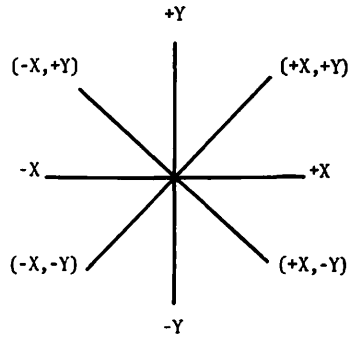


FIGURE 5.3 HILOT vector notations.

TABLE 5.1 HILOT command characters.

Command	Character	BASIC Address
+Y	p	9110
+X,+Y	q	9120
+X	r	9130
+X,-Y	s	9140
-Y	t	9150
-X,-Y	u	9160
-X	v	9170
-X,+Y	w	9180
Pen up	x	9200
Pen down	y	9210

the characters that produce control functions such as PEN UP. Since the plotter comes without any software, it is necessary to write subroutines to generate an axis system, to label the axis, and to plot the data.

Listing 5.2 shows a program which provides some of these capabilities. Figure 5.4 shows a flow chart for the BASIC program. This program draws an X,Y axis system, labels the system with numeric characters, and plots an exponential increase in voltage as a function of time. Figure 5.5 shows a series of plots taken for increasing values of the time constant. The software was written so that the entire plot could be scaled to different dimensions. Figure 5.6 shows the maximum range of plot size. It should be obvious to the reader that the program will most likely have to be modified to meet the requirements of the specific application. The program shown in Listing 5.2 was written in a very general sense

LISTING 5.2 BASIC program for operation of the digital plotter.

```

10 REM DIGITAL PLOTTER
12 REM APPLE II
14 REM R. HALLGREN 9-15-81
20 FOR I = 0 TO 160: READ K
22 POKE ( - 32800 + I),K
24 NEXT I
100 REM MAIN PROGRAM
110 HOME : VTAB 12
112 PRINT "POSITION PEN IN LOWER LEFT HAND"
114 PRINT "CORNER. PRESS ANY KEY TO CONTINUE."
116 GET A$
118 HOME
120 PRINT "INPUT THE SCALE FACTOR(3<K<19)"
122 INPUT K
124 IF K < 4 OR K > 18 THEN GOTO 118
128 HOME
140 GOSUB 1000
150 GOSUB 2000
200 REM PLOT DEMONSTRATION
202 CALL - 32646
210 FOR I = 0 TO .5 * K STEP .005
220 V = 5 * (1 - EXP ( - 10 * I / K))
222 L = INT (10 * K * V)
224 IF L - Z = 0 THEN GOTO 250: REM NO CHANGE IN POTENTIAL
226 IF L - Z < 0 THEN GOTO 240: REM POTENTIAL IS DECREASING
228 FOR J = 1 TO (L - Z): REM POTENTIAL IS INCREASING
230 CALL - 32712
232 NEXT J
234 GOTO 248
240 FOR J = 1 TO (Z - L)
242 CALL - 32681: NEXT J: REM MOVE PEN IN -Y DIRECTION
248 Z = L
250 CALL - 32695: NEXT I: REM MOVE PEN IN +X DIRECTION
299 END
1000 REM X-AXIS
1010 CALL - 32653
1012 FOR I = 1 TO 100 + 3 * K: CALL - 32712: NEXT I
1013 FOR I = 1 TO K + 90: CALL - 32695: NEXT I
1014 CALL - 32646
1016 FOR I = 1 TO 100 * K: CALL - 32695: NEXT I
1020 GOSUB 1900
1030 CALL - 32653
1032 FOR I = 1 TO 3 * K: CALL - 32667: NEXT I
1033 GOSUB 8100
1034 CALL - 32653: FOR I = 1 TO 4 * K: CALL - 32695: NEXT I
1035 GOSUB 8360
1036 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1037 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1040 CALL - 32653
1041 FOR I = 1 TO 10 * K: CALL - 32667: NEXT I: GOSUB 1900: CALL - 32653
1042 FOR I = 1 TO K: CALL - 32667: NEXT I
1043 GOSUB 8330
1044 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1050 CALL - 32653
1052 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1054 GOSUB 1900
1055 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1056 GOSUB 8300
1057 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1060 CALL - 32653
1062 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1064 GOSUB 1900
1065 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1066 GOSUB 8280

```

```

1067 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1070 CALL - 32653
1072 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1074 GOSUB 1900
1075 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1076 GOSUB 8250
1077 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1080 CALL - 32653
1082 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1084 GOSUB 1900
1085 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1086 GOSUB 8220
1087 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1090 CALL - 32653
1092 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1094 GOSUB 1900
1095 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1096 GOSUB 8180
1097 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1100 CALL - 32653
1102 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1104 GOSUB 1900
1105 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1106 GOSUB 8150
1107 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1110 CALL - 32653
1112 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1114 GOSUB 1900
1115 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1116 GOSUB 8120
1117 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1120 CALL - 32653
1122 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1124 GOSUB 1900
1125 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1126 GOSUB 8100
1127 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1130 CALL - 32653: FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1899 RETURN
1900 REM X-AXIS MARKINGS
1910 CALL - 32653
1914 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
1916 CALL - 32646
1918 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
1920 CALL - 32653
1922 FOR I = 1 TO 6 * K: CALL - 32681: NEXT I
1999 RETURN
2000 REM Y-AXIS
2014 CALL - 32646
2016 FOR I = 1 TO 70 * K: CALL - 32712: NEXT I
2020 GOSUB 2900
2030 GOSUB 8280
2035 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2040 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2042 GOSUB 2900
2044 GOSUB 8250
2045 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2050 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2052 GOSUB 2900
2054 GOSUB 8220
2055 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2060 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2062 GOSUB 2900
2064 GOSUB 8180
2065 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2070 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I

```

LISTING 5.2 (Continued)

```

2072 GOSUB 2900
2074 GOSUB 8150
2075 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2080 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2082 GOSUB 2900
2084 GOSUB 8120
2085 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2090 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2092 GOSUB 2900
2094 GOSUB 8100
2095 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2100 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2899 RETURN
2900 REM Y-AXIS MARKINGS
2910 CALL - 32653
2914 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
2916 CALL - 32646
2918 FOR I = 1 TO 4 * K: CALL - 32667: NEXT I
2920 CALL - 32653
2922 FOR I = 1 TO 4 * K: CALL - 32667: NEXT I
2924 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
2999 RETURN
8000 REM NUMERIC SUBROUTINES
8100 REM '1'
8101 CALL - 32646
8102 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8104 FOR I = 1 TO K: CALL - 32667: NEXT I
8106 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8108 FOR I = 1 TO K: CALL - 32674: NEXT I
8110 CALL - 32653: FOR I = 1 TO K: CALL - 32702: NEXT I
8112 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8114 FOR I = 1 TO K: CALL - 32667: NEXT I
8116 RETURN
8120 REM '2'
8122 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8123 CALL - 32646
8124 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8126 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8128 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8130 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8132 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8134 CALL - 32653
8136 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8138 RETURN
8150 REM '3'
8152 CALL - 32653
8154 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8156 CALL - 32646
8158 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8160 CALL - 32653
8162 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8164 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8166 CALL - 32646
8168 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8170 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8172 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8174 RETURN
8180 REM '4'
8182 CALL - 32653
8184 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8186 CALL - 32646
8188 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8190 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8192 CALL - 32646

```

```

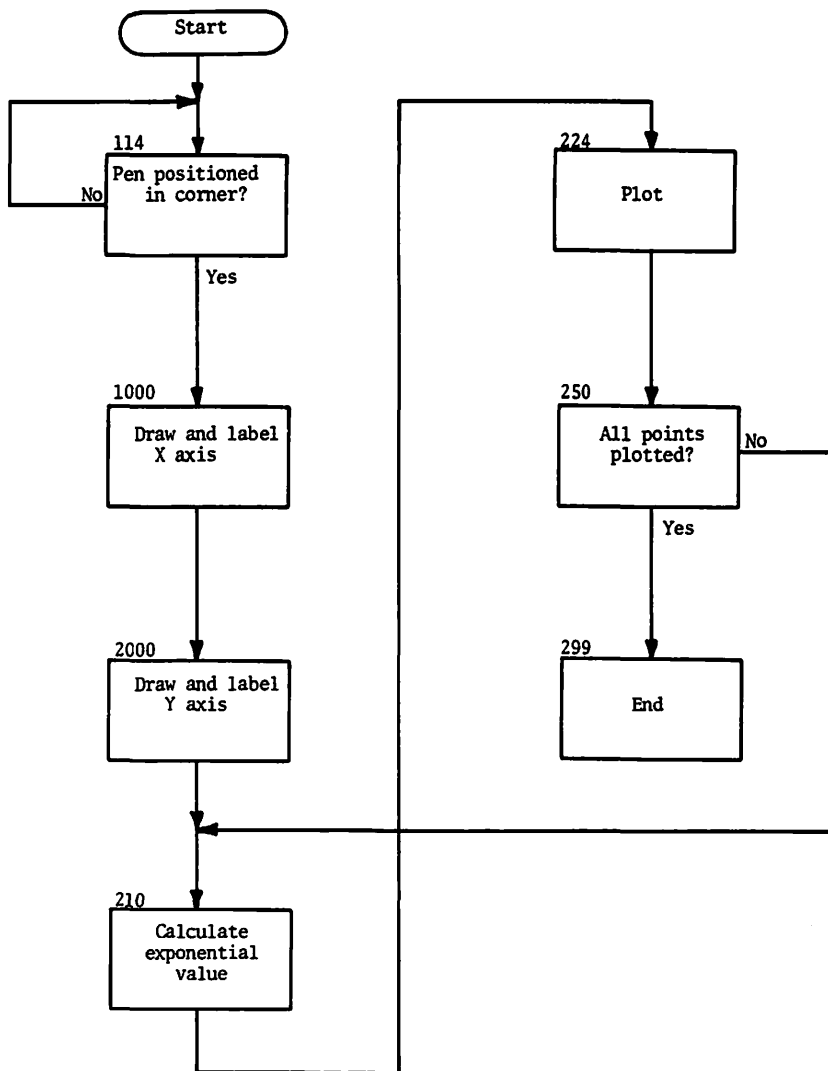
8194 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8196 CALL - 32646
8198 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8200 CALL - 32653
8210 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8212 RETURN
8220 REM '5'
8222 CALL - 32653
8224 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8226 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8228 CALL - 32646
8230 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8232 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8234 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8236 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8238 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8240 RETURN
8250 REM '6'
8252 CALL - 32653
8254 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8256 CALL - 32646
8258 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8260 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8262 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8264 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8266 CALL - 32653
8268 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8270 RETURN
8280 REM '7'
8282 CALL - 32653
8284 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8286 CALL - 32646
8288 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8290 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8292 CALL - 32653
8294 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8296 RETURN
8300 REM '8'
8302 CALL - 32646
8304 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8306 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8308 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8310 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8312 CALL - 32653
8314 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8316 CALL - 32646
8318 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8320 CALL - 32653
8322 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8324 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8326 RETURN
8330 REM '9'
8332 CALL - 32653
8334 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8336 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8338 CALL - 32646
8340 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8342 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8344 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8346 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8348 CALL - 32653
8350 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8352 RETURN
8360 REM '0'
8362 CALL - 32646
8364 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I

```

```
8366 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I  
8368 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I  
8370 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I  
8372 RETURN  
10000 DATA 160,9,24,72,176,7,169,0,141,209,192,144,5,169,1,141,209,192,169,3  
10002 DATA 72,169,4,74,144,253,104,233,1,208,245,104,106,136,208,223,160,2,169  
10004 DATA 1,141,209,192,169,3,72,169,4,74,144,253,104,233,1,208,245,136,208  
10006 DATA 235,96,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,  
10008 DATA 72,8,169,112,32,224,127,40,104,96,72,8,169,113,76,60,128,72,8,169,114  
10010 DATA 76,60,128,72,8,169,115,76,60,128,72,8,169,116,76,60,128,72,8,169,117  
10012 DATA 76,60,128,72,8,169,118,76,60,128,72,8,169,119,76,60,128,72,8,169,121  
10014 DATA 76,60,128,72,8,169,122,76,60,128
```

```

graph TD
    Start([Start]) --> 114{114 Pen positioned in corner?}
    114 -- No --> 114
    114 -- Yes --> 1000[1000 Draw and label X axis]
    1000 --> 2000[2000 Draw and label Y axis]
    2000 --> 210[210 Calculate exponential value]
    210 --> 224[224 Plot]
    224 --> 250{250 All points plotted?}
    250 -- No --> 224
    250 -- Yes --> 299[299 End]
  
```



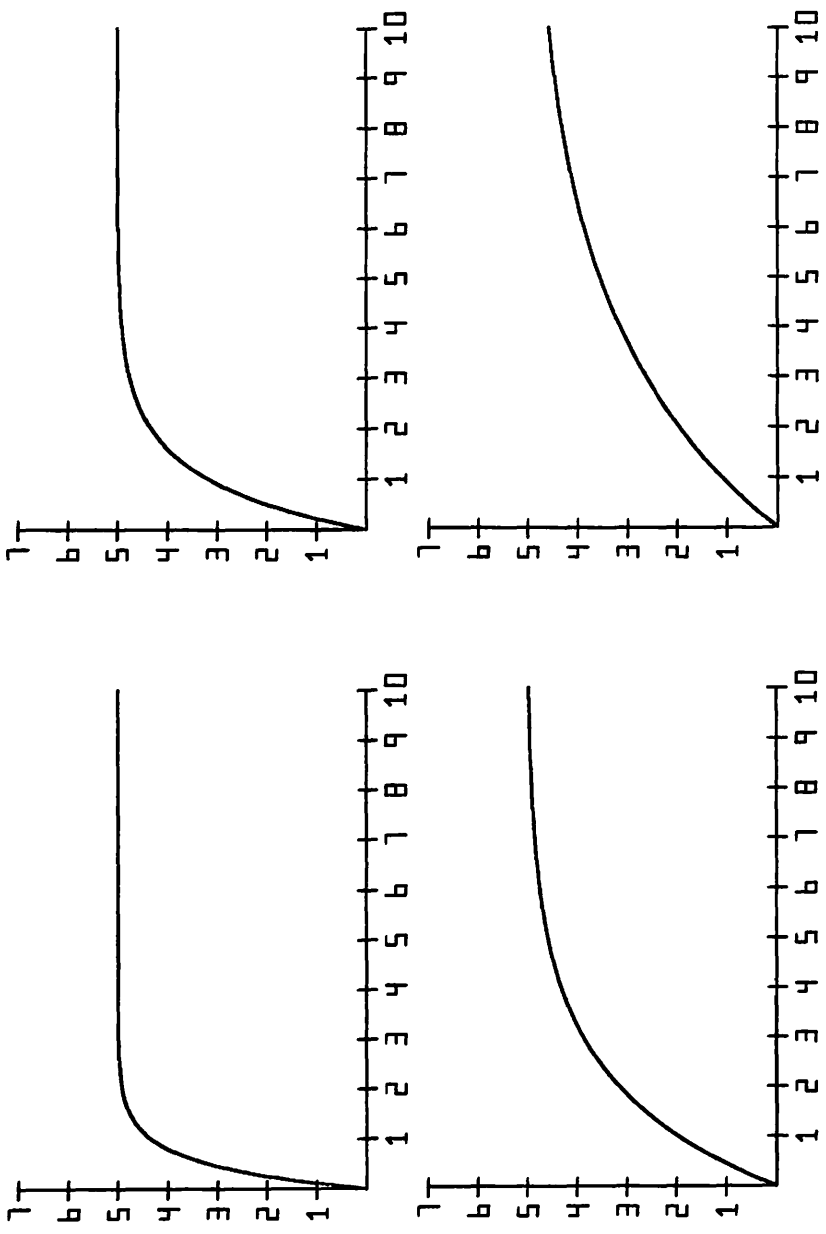


FIGURE 5.5 Series of exponential plots for increasing values of the time constant.

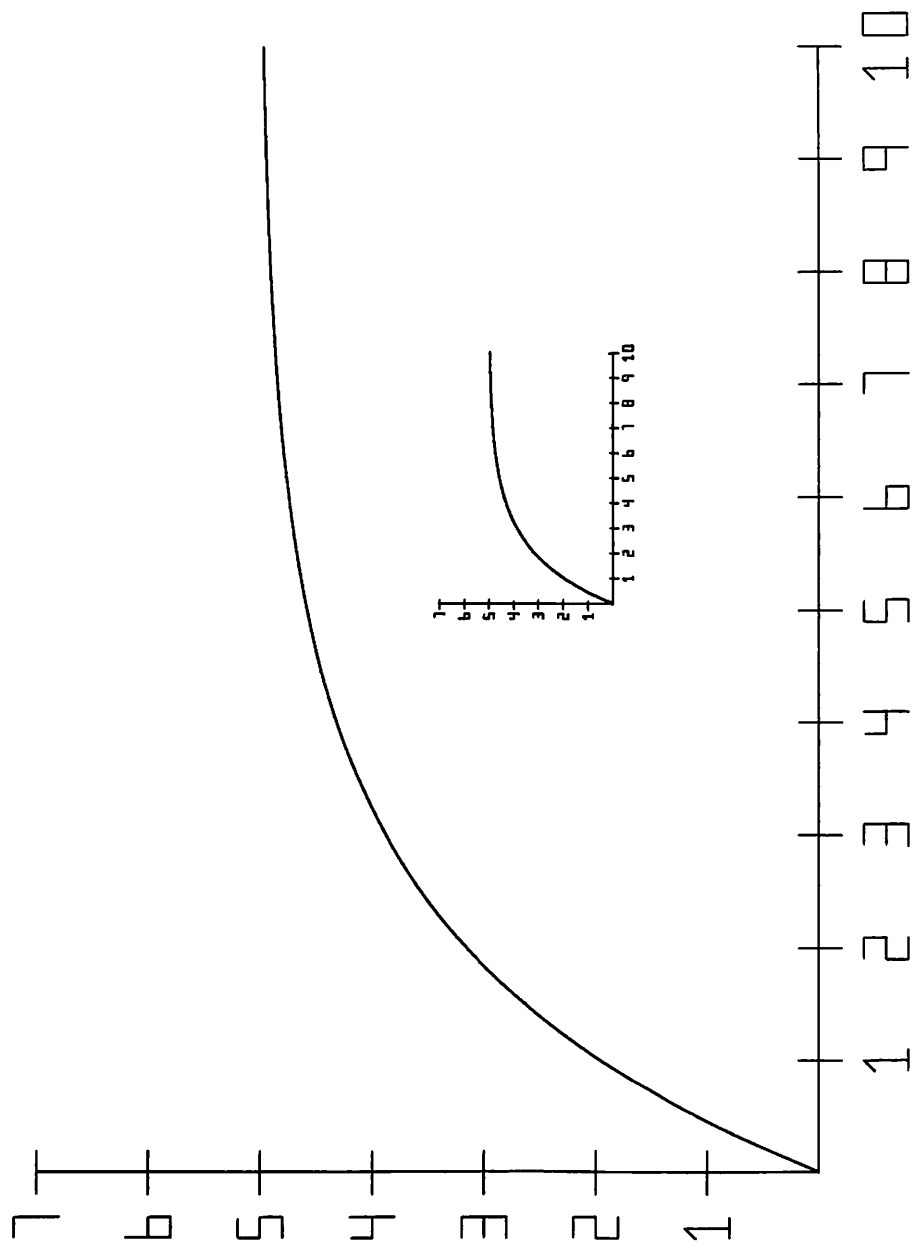


FIGURE 5.6 Maximum and minimum plot dimensions.

to make it easier for the reader to follow. Consequently, a significant increase in plotting speed can be realized by reducing the number of subroutine calls and keeping the software compacted as much as is possible.

Suppose we would like to plot some of the data that we digitized with the high speed A/D converter and stored on disk back in Chapter 3. Listing 5.3 shows a program which will take the stored data and plot it on the digital plotter. Figure 5.7 shows a plot of a sine wave plotted at both the the maximum and minimum plot size. The frequency of the sine wave was approximately 10 Hz. You should be aware that the horizontal scale (for a sampling rate of 1000 samples per second) is such that one period of a 20 Hz sine would occupy one inch. Figure 5.8 shows a plot of digitized music which was playing on the radio while I wrote this. Note that it is very difficult to say much about the frequency content of the music from the plot. When you get to Chapter 8, I will show you how you can analyze data similar to this and come to some conclusions about the frequency spectrum.

LISTING 5.3 BASIC program for plotting data stored on a disc.

```

10 REM DIGITAL PLOTTER
12 REM APPLE II
14 REM R. HALLGREN 9-15-81
16 HIMEM: 30000
20 FOR I = 0 TO 160: READ K
22 POKE ( - 32800 + I),K
24 NEXT I
100 REM MAIN PROGRAM
110 HOME : VTAB 12
112 PRINT "POSITION PEN IN LOWER LEFT HAND"
114 PRINT "CORNER. PRESS ANY KEY TO CONTINUE."
116 GET A$
118 HOME
120 PRINT "INPUT THE SCALE FACTOR (3<K<18)"
122 INPUT K
124 IF K < 4 OR K > 17 THEN GOTO 118
128 HOME
140 GOSUB 1000
150 GOSUB 2000
200 REM PLOT DEMONSTRATION
202 DIM X1(2048):D$ = ""
203 PRINT D$;"BLOAD DATA($1100),A$6100"
205 X = 25136
206 FOR Z = 0 TO 2047:X = X + 2
208 V1 = PEEK (X):V2 = PEEK (X + 1)
209 IF V1 > 1 THEN GOTO 218
210 X1(Z) = - ((511 - (V1 * 256 + V2)) / 100)
211 NEXT Z
212 GOTO 220
218 X1(Z) = ((V1 - 2) * 256 + V2) / 100)
219 GOTO 211
220 RESTORE : FOR I = 0 TO 160: READ R
221 POKE ( - 32800 + I),R: NEXT I

```

LISTING 5.3 (Continued)

```

222 Z = 0: CALL - 32646: FOR I = 0 TO 100 * K: L = INT (10 * K * X1(I))
224 IF L - Z = 0 THEN GOTO 250: REM NO CHANGE IN POTENTIAL
226 IF L - Z < 0 THEN GOTO 240: REM POTENTIAL IS DECREASING
228 FOR J = 1 TO (L - Z): REM POTENTIAL IS INCREASING
230 CALL - 32712
232 NEXT J
234 GOTO 248
240 FOR J = 1 TO (Z - L)
242 CALL - 32681: NEXT J: REM MOVE PEN IN -Y DIRECTION
248 Z = L
250 CALL - 32695: NEXT I: REM MOVE PEN IN +X DIRECTION
299 END
1000 REM X-AXIS
1010 CALL - 32653
1012 FOR I = 1 TO 675 + 3 * K: CALL - 32712: NEXT I
1013 FOR I = 1 TO K + 90: CALL - 32695: NEXT I
1014 CALL - 32646
1016 FOR I = 1 TO 100 * K: CALL - 32695: NEXT I
1020 GOSUB 1900
1030 CALL - 32653
1032 FOR I = 1 TO 3 * K: CALL - 32667: NEXT I
1033 GOSUB 8100
1034 CALL - 32653: FOR I = 1 TO 4 * K: CALL - 32695: NEXT I
1035 GOSUB 8360
1036 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1037 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1040 CALL - 32653
1041 FOR I = 1 TO 10 * K: CALL - 32667: NEXT I: GOSUB 1900: CALL - 32653
1042 FOR I = 1 TO K: CALL - 32667: NEXT I
1043 GOSUB 8330
1044 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1050 CALL - 32653
1052 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1054 GOSUB 1900
1055 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1056 GOSUB 8300
1057 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1060 CALL - 32653
1062 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1064 GOSUB 1900
1065 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1066 GOSUB 8280
1067 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1070 CALL - 32653
1072 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1074 GOSUB 1900
1075 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1076 GOSUB 8250
1077 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1080 CALL - 32653
1082 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1084 GOSUB 1900
1085 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1086 GOSUB 8220
1087 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1090 CALL - 32653
1092 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1094 GOSUB 1900
1095 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1096 GOSUB 8180
1097 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1100 CALL - 32653
1102 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1104 GOSUB 1900

```

```

1105 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1106 GOSUB 8150
1107 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1110 CALL - 32653
1112 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1114 GOSUB 1900
1115 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1116 GOSUB 8120
1117 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1120 CALL - 32653
1122 FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1124 GOSUB 1900
1125 CALL - 32653: FOR I = 1 TO K: CALL - 32667: NEXT I
1126 GOSUB 8100
1127 CALL - 32653: FOR I = 1 TO 8 * K: CALL - 32712: NEXT I
1130 CALL - 32653: FOR I = 1 TO 9 * K: CALL - 32667: NEXT I
1899 RETURN
1900 REM X-AXIS MARKINGS
1910 CALL - 32653
1914 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
1916 CALL - 32646
1918 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
1920 CALL - 32653
1922 FOR I = 1 TO 6 * K: CALL - 32681: NEXT I
1999 RETURN
2000 REM Y-AXIS
2014 CALL - 32646
2016 FOR I = 1 TO 40 * K: CALL - 32712: NEXT I
2020 GOSUB 2900
2030 GOSUB 8180
2035 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2040 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2042 GOSUB 2900
2044 GOSUB 8150
2045 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2050 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2052 GOSUB 2900
2054 GOSUB 8120
2055 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2060 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2062 GOSUB 2900
2064 GOSUB 8100
2065 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2070 FOR I = 1 TO 18 * K: CALL - 32681: NEXT I
2072 GOSUB 2900
2074 GOSUB 8100
2075 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2080 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2082 GOSUB 2900
2084 GOSUB 8120
2085 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2090 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2092 GOSUB 2900
2094 GOSUB 8150
2095 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2100 FOR I = 1 TO 8 * K: CALL - 32681: NEXT I
2110 GOSUB 2900
2112 GOSUB 8180
2114 CALL - 32653: FOR I = 1 TO 6 * K: CALL - 32695: NEXT I
2116 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
2120 CALL - 32646
2122 FOR I = 1 TO 40 * K: CALL - 32712: NEXT I
2899 RETURN
2900 REM Y-AXIS MARKINGS
2910 CALL - 32653
2914 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I

```

LISTING 5.3 (Continued)

```

2916 CALL - 32646
2918 FOR I = 1 TO 4 * K: CALL - 32667: NEXT I
2920 CALL - 32653
2922 FOR I = 1 TO 4 * K: CALL - 32667: NEXT I
2924 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
2999 RETURN
8000 REM NUMERIC SUBROUTINES
8100 REM "1"
8101 CALL - 32646
8102 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8104 FOR I = 1 TO K: CALL - 32667: NEXT I
8106 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8108 FOR I = 1 TO K: CALL - 32674: NEXT I
8110 CALL - 32653: FOR I = 1 TO K: CALL - 32702: NEXT I
8112 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8114 FOR I = 1 TO K: CALL - 32667: NEXT I
8116 RETURN
8120 REM "2"
8122 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8123 CALL - 32646
8124 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8126 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8128 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8130 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8132 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8134 CALL - 32653
8136 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8138 RETURN
8150 REM "3"
8152 CALL - 32653
8154 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8156 CALL - 32646
8158 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8160 CALL - 32653
8162 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8164 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8166 CALL - 32646
8168 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8170 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8172 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8174 RETURN
8180 REM "4"
8182 CALL - 32653
8184 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8186 CALL - 32646
8188 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8190 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8192 CALL - 32646
8194 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8196 CALL - 32646
8198 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8200 CALL - 32653
8210 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8212 RETURN
8220 REM "5"
8222 CALL - 32653
8224 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8226 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8228 CALL - 32646
8230 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8232 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8234 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I

```

```

8236 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8238 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8240 RETURN
8250 REM "6"
8252 CALL - 32653
8254 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8256 CALL - 32646
8258 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8260 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8262 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8264 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8266 CALL - 32653
8268 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8270 RETURN
8280 REM "7"
8282 CALL - 32653
8284 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8286 CALL - 32646
8288 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8290 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8292 CALL - 32653
8294 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8296 RETURN
8300 REM "8"
8302 CALL - 32646
8304 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8306 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8308 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8310 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8312 CALL - 32653
8314 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8316 CALL - 32646
8318 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8320 CALL - 32653
8322 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8324 FOR I = 1 TO 2 * K: CALL - 32681: NEXT I
8326 RETURN
8330 REM "9"
8332 CALL - 32653
8334 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8336 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8338 CALL - 32646
8340 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8342 FOR I = 1 TO 2 * K: CALL - 32712: NEXT I
8344 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8346 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8348 CALL - 32653
8350 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8352 RETURN
8360 REM "0"
8362 CALL - 32646
8364 FOR I = 1 TO 4 * K: CALL - 32712: NEXT I
8366 FOR I = 1 TO 2 * K: CALL - 32695: NEXT I
8368 FOR I = 1 TO 4 * K: CALL - 32681: NEXT I
8370 FOR I = 1 TO 2 * K: CALL - 32667: NEXT I
8372 RETURN
10000 DATA 160,9,24,72,176,7,169,0,141,209,192,144,5,169,1,141,209,192,169,3
10002 DATA 72,169,4,74,144,253,104,233,1,208,245,104,106,136,208,223,160,2,169
10004 DATA 1,141,209,192,169,3,72,169,4,74,144,253,104,233,1,208,245,136,208
10006 DATA 235,96,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10008 DATA 72,8,169,112,32,224,127,40,104,96,72,8,169,113,76,60,128,72,8,169,114
10010 DATA 76,60,128,72,8,169,115,76,60,128,72,8,169,116,76,60,128,72,8,169,117
10012 DATA 76,60,128,72,8,169,118,76,60,128,72,8,169,119,76,60,128,72,8,169,121
10014 DATA 76,60,128,72,8,169,122,76,60,128

```

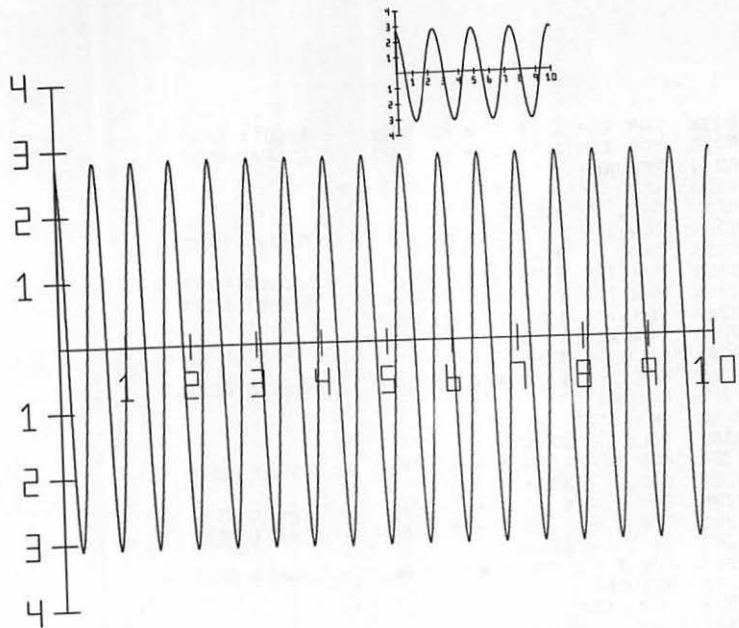
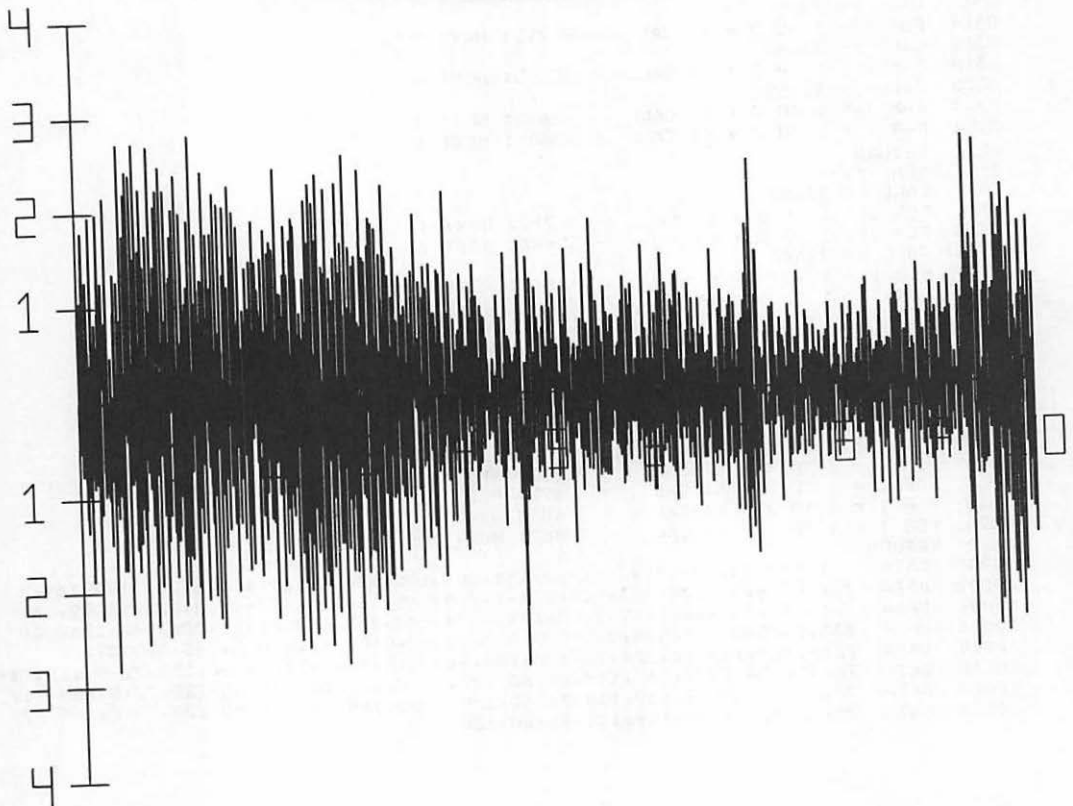


FIGURE 5.7 Sine wave (10 Hz) digitized at 1000 samples per second.

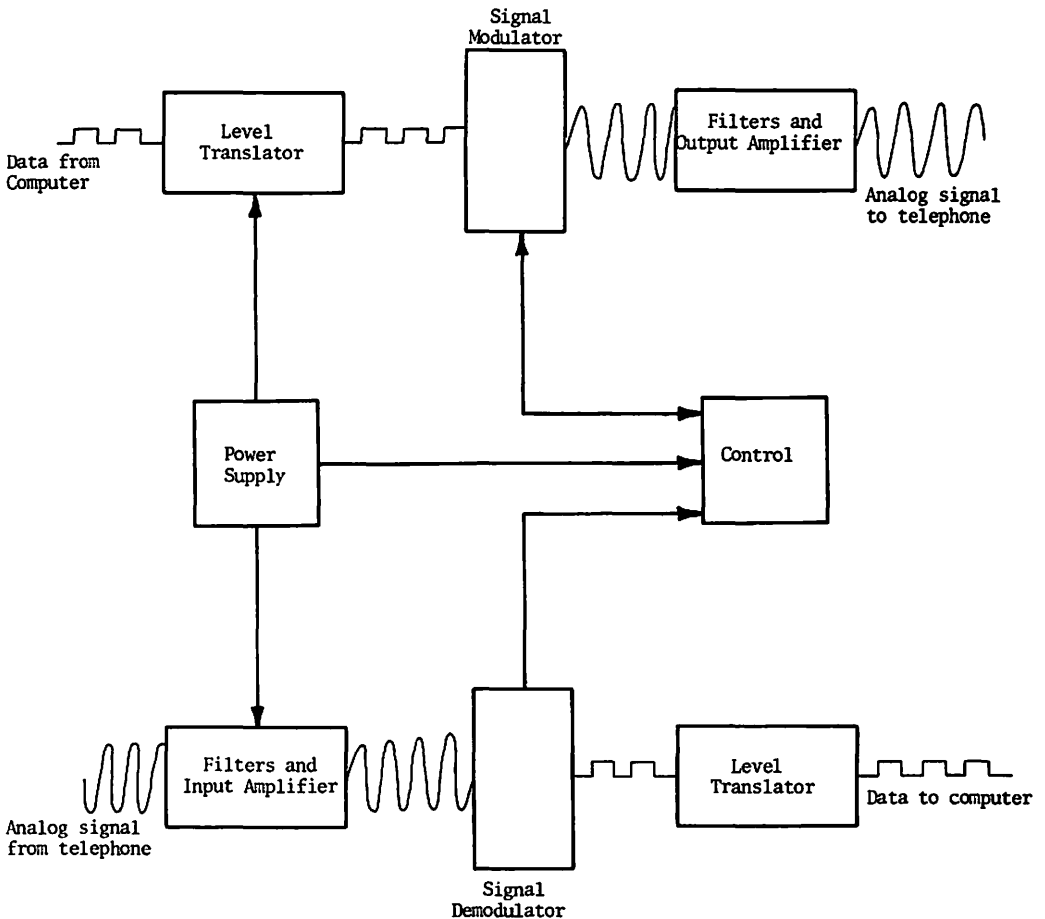
FIGURE 5.8 Music digitized at 1000 samples per second.



VIDEO TERMINAL

There will be times when you will want to use your Apple II as a video terminal to connect into a large, host computer network. Networks that provide a message retrieval system have been around for quite a while, and we are now beginning to see a proliferation of information services which provide local, national, and international news, weather, sports, and investment-related topics. In the future, you will probably have access to information suppliers dealing with banking, car rental, classified ads, and reservation services. Individuals usually have access to one of these networks through the use of a modem connected between a video terminal and their telephone. A modem is a device which performs two basic functions (see Figure 5.9):

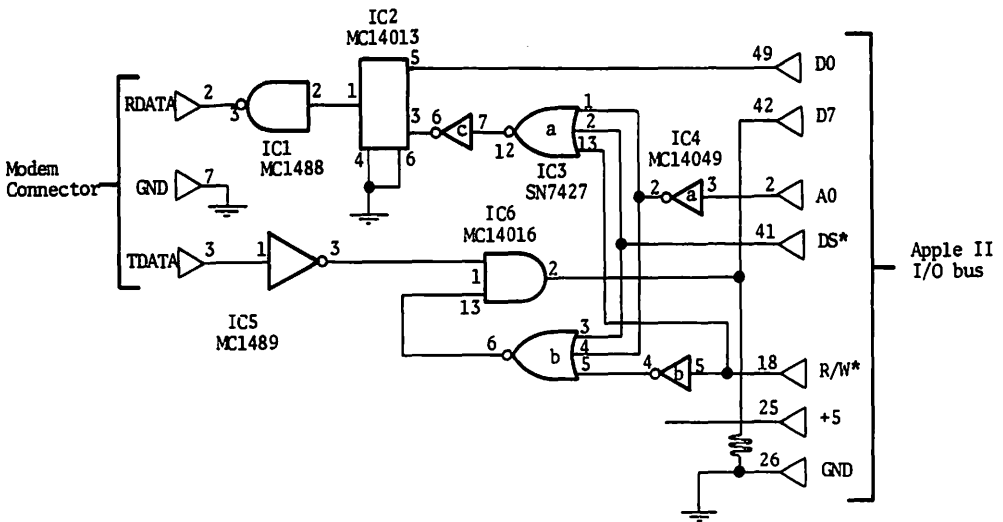
FIGURE 5.9 Block diagram of a modem.



- 1. It accepts digital input from a computer and converts the pulses into analog-frequency signals for transmission over the telephone lines.
- 2. It converts analog-frequency signals from the telephone into digital signals which the computer can process.

Modems that transmit information at 300 bits per second are readily available through a multitude of suppliers for under \$200. You can use the Apple II as a video terminal by building a bidirectional serial interface and writing some software. You could accomplish the same thing by buying a commercial serial interface board, but you can save a few dollars by building your own. You should realize that my approach minimizes hardware and maximizes software, thus saving money at the expense of versatility. Figure 5.10 shows the hardware that will be needed, and the connections that will need to be made to the Apple II and to the

FIGURE 5.10 Video terminal interface circuit and system connections.



Number	Type	+5	+12	-12	GND
IC1	MC1488	--	14	1	7
IC2	MC14013	14	--	--	7
IC3	SN7427	14	--	--	7
IC4	MC14049	1	--	--	8
IC5	MC1489	14	--	--	7
IC6	MC14016	14	--	--	7

modem. You should recognize the circuits as ones that were developed and described in Chapter 2. Once you have constructed the interface and made the interconnections, perform the following tests to make sure that everything is working correctly:

1. If you execute the following BASIC statement the output of IC1 should go to +10 volts.
POKE -16143,0
2. If you execute the following BASIC statement the output of IC1 should go to -10 volts.
POKE -16143,1
3. If you execute the following BASIC statements the number which will be displayed on the video monitor should change from a 1 to a 0 when you apply +5 volts to the SERIAL INPUT line.
100K=PEEK (-16143)
110 IF K<128 THEN PRINT "0"
120 IF K>128 THEN PRINT "1"
130 GOTO 100

Listing 5.4 shows the machine language program (see Figure 5.11 for the program flow chart) which allows the Apple II to transmit and receive serial data at 300 bits per second. Listing 5.5 shows the program converted over to a series of BASIC statements.

LISTING 5.4 Machine language routine for the bi-direction serial interface.

300	AD 00 C0	LDA \$C000	SAMPLE KEYBOARD
303	48	PHA	SAVE A REGISTER
304	C9 80	CMP #\$80	CHECK FOR CHARACTER
306	30 38	BMI \$340	JUMP TO RECEIVE
308	A9 00	LDA #\$00	
30A	8D 10 00	STA \$C010	RESET KEYBOARD
30D	68	PLA	RESTORE A REGISTER
30E	29 7F	AND #\$7F	
310	A0 0B	LDY #\$0B	11 BITS
312	18	CLC	CLEAR CARRY
313	48	PHA	SAVE A REGISTER
314	B0 07	BCS \$31D	MARK
316	A9 00	LDA #\$00	
318	8D F1 C0	STA \$C0F1	SEND SPACE
31B	90 05	BCC \$322	
31D	A0 01	LDA #\$01	
31F	8D F1 C0	STA \$C0F1	SEND MARK
322	20 30 03	JSR \$330	JUMP TO DELAY
325	68	PLA	RESTORE A REGISTER
326	6A	ROR	GET NEXT BIT
327	88	DEY	ALL BITS SENT
328	D0 E9	BNE \$313	JUMP IF NOT
32A	4C 40 03	JMP \$340	JUMP TO RECEIVE
32D	EA	NOP	
32E	EA	NOP	

LISTING 5.4 (Continued)

32F	EA	NOP	
330	A9 4D	LDA #\$4D	LOAD A REGISTER
332	48	PHA	SAVE A REGISTER
333	A9 20	LDA #\$20	LOAD REGISTER
335	4A	LSR	SHIFT RIGHT
336	90 FD	BCC \$335	JUMP IF NOT ZERO
338	68	PLA	RESTORE A REGISTER
339	E9 01	SBC #\$01	SUBTRACT ONE
33B	D0 F5	BNE \$332	JUMP IF NOT ZERO
33D	60	RTS	RETURN
33E	EA	NOP	
33F	EA	NOP	
340	A9 00	LDA #\$00	LOAD A REGISTER
342	8D 80 03	STA \$380	TEMP A REGISTER
345	A2 08	LDX #\$08	8 DATA BITS
347	AD F1 C0	LDA \$C0F1	GET RECEIVED CHARACTER
34A	2A	ROL	ROTATE LEFT
34B	B0 B3	BCS \$300	JUMP TO SEND
34D	A9 26	LDA #\$26	1/2 CHARACTER DELAY
34F	48	PHA	
350	A9 20	LDA #\$20	
352	4A	LSR	
353	90 FD	BCC \$352	
355	68	PLA	
356	E9 01	SBC #\$01	
358	D0 F5	BNE \$34F	
35A	20 30 03	JSR \$330	JUMP TO DELAY
35D	AD F1 C0	LDA \$C0F1	GET CHARACTER
360	29 80	AND #\$80	GET MSB
362	0D 80 03	ORA \$380	OR A REGISTER WITH TEMP
365	8D 80 03	STA \$380	SAVE IN TEMP
368	18	CLC	CLEAR CARRY
369	6E 80 03	ROR \$380	ROTATE TEMP RIGHT
36C	CA	DEX	ALL BITS RECEIVED?
36D	D0 EB	BNE \$35A	BRANCH IF NOT
36F	09 80	ORA #\$80	SET MSB
371	C9 C0	CMP #\$C0	CHECK FOR CONTROL CHARAC
373	F0 03	BEQ \$378	JUMP IF NOT
375	20 FD FB	JSR \$FBFD	DISPLAY ON VIDEO
378	20 30 03	JSR \$330	JUMP TO DELAY
37B	4C 00 03	JMP \$300	JUMP TO SEND

I will describe the startup procedure that I use for interacting with the university computer system. Since operating systems will differ from location to location, this should be used as an example only.

1. Load and execute the BASIC program.
2. Dial up the computer facility, listen for the carrier tone, and insert the handset into the modem (this assumes that you have not purchased a direct-connect modem).
3. Press the RETURN key. If the host does not respond, press the RETURN key again.
4. Once the host responds, sign on in the normal way.

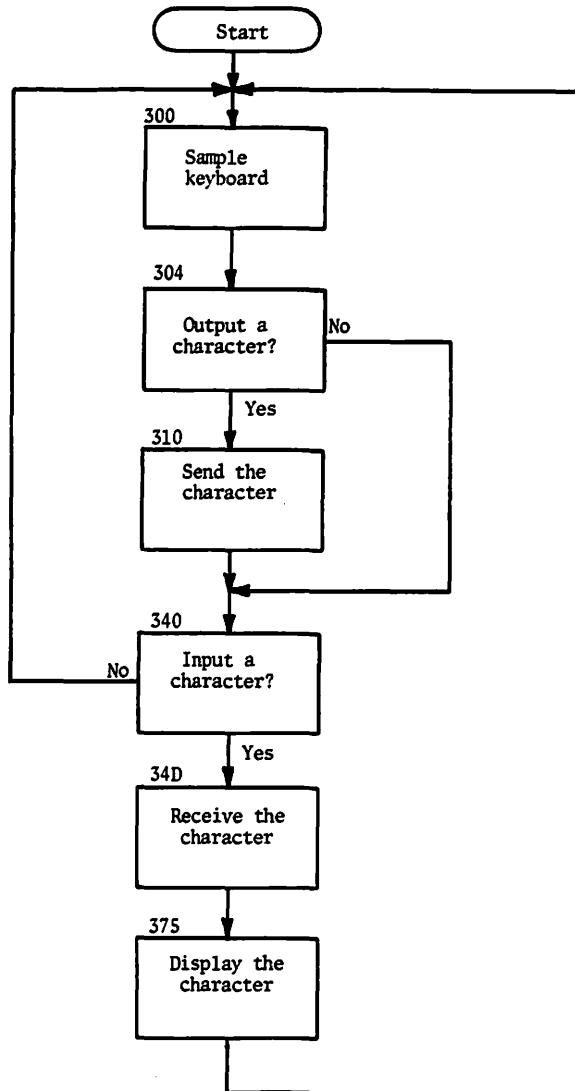


FIGURE 5.11 Video terminal program flow chart.

5. Limit the right hand margin to 38 characters to prevent losing data. Double the delay time before printing is resumed after a line feed and a carriage return to avoid losing characters.
6. To abort an operation, press the ESC key.

LISTING 5.5 The machine language program in Listing 5.4 incorporated into a BASIC program.

```
100 REM APPLE II VIDEO TERMINAL
110 REM R. HALLGREN, 9/17/81
120 FOR I=0 TO 125:READ K:POKE (768+I),K:NEXT I
126 HOME
128 PRINT "DIAL UP THE HOST COMPUTER AND PRESS THE
      'RETURN' KEY"
130 FOR I=1 TO 5:PRINT "":NEXT I
140 PRINT "REMEMBER TO LIMIT THE RIGHT MARGIN"
142 PRINT "TO 36 CHARACTERS, AND TO DOUBLE THE"
144 PRINT "DELAY THAT COMES AFTER A LINE FEED"
146 PRINT "AND A CARRIAGE RETURN."
150 CALL 773
199 END
10000 DATA 173,0,192,72,201,128,48,56,169,0,141,16,192,104,
      41,127,160,11,24,72
10002 DATA 176,7,169,0,141,241,192,144,5,169,1,141,241,192,
      32,48,3,104,106,136
10004 DATA 208,233,76,64,3,234,234,234,169,77,72,169,32,74,
      144,253,104,233,1
10006 DATA 208,245,96,234,234,169,0,141,128,3,162,8,173,
      241,192,42,176,179,169
10008 DATA 38,72,169,32,74,144,253,104,233,1,208,245,32,48,
      3,173,241,192,41,128
10010 DATA 13,128,3,141,128,3,24,110,128,3,202,208,235,9,
      128,201,192,240,3,32
10012 DATA 253,251,32,48,3,76,0,3
```



BIOFEEDBACK

Have you ever thought about the number of complex operations that your body is required to perform in the course of a “normal” day of activity? While we often take for granted our ability to perform complex, coordinated muscular activity, this ability results from years of trial-and-error type training. An infant has very little muscular control over its body, and it develops this control by constantly practicing. Most of us have observed a young child learning how to walk; repetitiously standing, taking a few steps, and falling down.

The human brain is composed of billions of specialized cells called neurons. Neurons are designed to transmit information between peripheral sensory receptors and the brain, to analyze this information, and to transmit commands, if necessary, back to muscles. For example, when you touch an object, there are sen-

sory receptors in the skin that send the brain coded information related to the temperature of the object. The brain then processes the information and decides what course of action should be taken. Coded information is then sent to the muscles in the hand and arm. The possible result is that the hand may be jerked back from a very hot object, or left in place if the object is cool.

The body is equipped with many similar neuromuscular pathways that never get developed because there is no common use for them. It is no big trick to learn how to wiggle your ears; you just spend some time in front of a mirror teaching yourself which nerves are connected to that particular set of muscles. The normal process of learning requires that *feedback* accompany the behavior pattern that is being established. That is, you normally expect to see, hear, or feel the results of a particular action. It is difficult to learn how to play the violin if you are deaf. The normal learning experience requires that auditory feedback accompany manipulation of the bow.

Many functions of the body, such as heart rate and skin temperature—long thought to be under exclusive control of the autonomic nervous system—have been found to be controllable. Training involuntary muscles is normally not possible, because we lack a natural feedback path to tell us how we are affecting the system. However, with the aid of sensitive electronic equipment, physiological changes that a person is normally incapable of sensing can be altered and controlled. Thus, the term *biofeedback* is used when a biological process is measured, and the results of the measurement are fed back to the individual. Through a training experience, the individual can learn to control phenomena such as heart rate and skin temperature.

SKIN TEMPERATURE

Heat is produced in the human body by muscular exercise, assimilation of food, and all the processes that contribute to the basal metabolic rate. Heat is lost to the surroundings through radiation, conduction, and vaporization of water in the nasal passages, and from the skin. Humans—and vertebrates in general—have multiple mechanisms for regulating heat loss so that the body remains at an almost constant temperature under

normal conditions. One of the primary mechanisms for heat regulation is through control of skin temperature. Skin temperature is controlled by regulating the amount of blood that is allowed to flow to the skin. Cold hands and feet are a common phenomena resulting from peripheral vasoconstriction, resulting from a sympathetic nervous system command to contract the smooth muscles lining peripheral blood vessels, thus reducing blood circulation and skin temperature at the extremities and conserving heat.

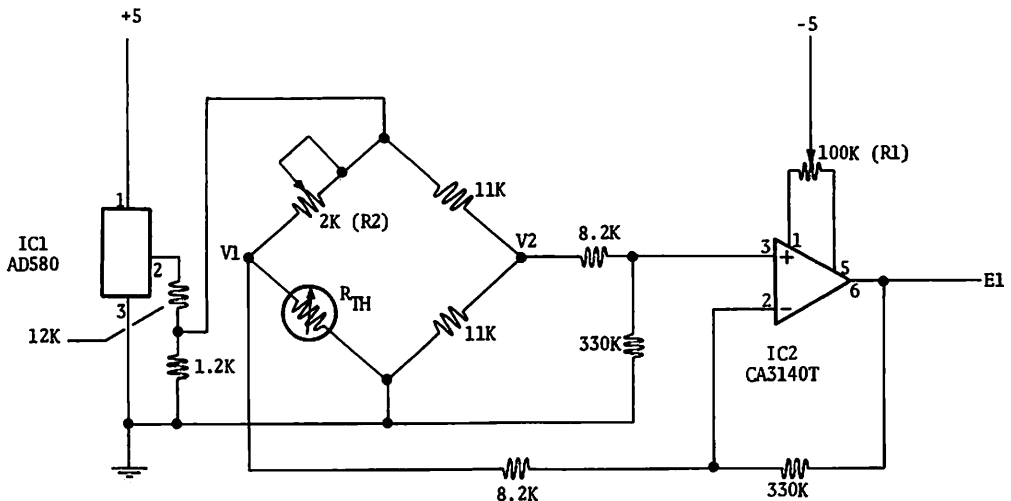
Peripheral skin temperature is influenced by a number of factors.

1. Stress causes fingertip temperature to drop several degrees below room temperature.
2. A variety of drugs, including tobacco, have an influence on vasoconstriction.
3. Vigorous activity increases peripheral circulation, while inactivity decreases circulation.

We will be using the low speed A/D converter that we developed in Chapter 3 along with an analog circuit to interface a thermistor to the Apple II. Figure 6.1 shows the schematic diagram of the

FIGURE 6.1 Schematic diagram of the analog circuitry for the biofeedback temperature monitor.

Number	Type	+5	-5	GND
IC1	AD580	1	--	3
IC2	CA3140T	7	4	--



analog circuitry. This design uses a thermistor, a device whose resistance is a function of its temperature, in one leg of a bridge circuit. Bridge circuits allow us to detect small changes in resistance by measuring imbalance in the bridge with a differential amplifier.

Thermistors are nearly ideal components mechanically. They are usually small in size, rugged, and they demonstrate the reliability and extended life common to semiconductor products. By using appropriate values of resistance in the bridge circuit, small changes in temperature can be made to produce large changes in output voltage. Since the output voltage is amplified by a differential amplifier (see Appendix B), there will be little problem of noise being introduced into the measurements.

Looking at Figure 6.1, IC1 is a voltage regulator which provides a stable reference voltage to the bridge circuitry. The output voltage from IC1 has been reduced to minimize temperature variations that might occur as a result of self-heating in the thermistor. Such variations, appearing as a slow drifting of the output signal, are undesirable. IC2 is a high gain, differential amplifier that is used to measure the difference between voltages V1 and V2.

Once you have built the analog circuit, you should make the following adjustments and voltage checks:

1. The voltage at pin 2 on IC1 should be equal to +2.5 volts.
2. Short the points labeled V1 and V2 together and adjust R1 until the voltage at E1 is equal to 0 volts. This ensures that the output of the differential amplifier will be equal to 0 when the inputs are equipotential.
3. Remove the short from the points labeled V1 and V2, hold the thermistor in your mouth for one minute, and adjust R2 (the temperature calibration adjustment) until the output voltage, E1, is equal to +1.07 volts. You should realize that we are going to be interested in relative changes in temperature and not in absolute values of temperature, so calibration will not be precisely performed.
4. Connect the output, E1, to the input of the A/D converter. Load and execute BASIC program 1 (see Listing 3.3). The video display should indicate that you are measuring a voltage that increases when you hold the thermistor between your fingertips and decreases when you let go of it.

5. Load and execute BASIC program 2 (see Listing 6.1 and Figure 6.2). You should see temperature plotted as each sample is taken. The plot should move toward the top of the screen for increases in temperature and toward the bottom of the screen for decreases in temperature. Figure 6.3 shows a data plot that results from first putting the thermistor into, and then taking the thermistor out of, the author's mouth. Notice how the temperature of the thermistor goes below the room temperature as a result of evaporation.

LISTING 6.1 BASIC program for biofeedback skin temperature experiments.

```

10 REM LOWSPEED A/D CONVERTER #1
20 REM R. HALLGREN,APPLE II,10-22-81
22 FOR I = 16384 TO 16584: READ D: POKE I,D: NEXT I
24 GOSUB 1000
30 REM CONVERT ONE SAMPLE
32 HGR : HCOLOR= 3
34 HPLLOT 0,0 TO 0,150: HPLLOT TO 270,150
40 FOR I = 1 TO 270
50 CALL 16384
70 X4 = PEEK (18944):X3 = PEEK (18945)
72 X2 = PEEK (18946):X1 = PEEK (18947)
73 W4 = X4
74 IF X4 > 7 THEN X4 = 0
75 IF X4 = 0 THEN GOTO 80
76 X4 = 1
80 X$ = STR$ (X4) + STR$ (X3) + STR$ (X2) + STR$ (X1)
81 X = - VAL (X$) / 1000
82 IF X4 = 0 THEN W4 = W4 - 8
83 IF W4 > 3 THEN X = - X
90 HPLLOT I,(150 - 100 * X): FOR J = 0 TO 500: NEXT J: NEXT I
92 GOTO 30
99 END
1000 REM INTRODUCTION
1010 HOME
1030 PRINT "THIS IS DEMONSTRATION PROGRAM FOR"
1032 PRINT "CONTROLLING SKIN TEMPERATURE USING"
1034 PRINT "BIOFEEDBACK IN CONJUNCTION WITH THE"
1036 PRINT "LOW SPEED A/D CONVERTER AND A DATA"
1038 PRINT "PLOT ROUTINE."
1040 PRINT "": PRINT ""
1042 PRINT "THE PROGRAM WILL PLOT A POINT EVERY": PRINT "SECOND"
1044 PRINT "": PRINT ""
1046 PRINT "PRESS THE SPACE BAR TO CONTINUE"
1050 GET K$
1052 IF K$ < > " " THEN GOTO 1050
1099 RETURN
2000 DATA 141,176,73,142,177,73,140,178,73,8,162,0,169,0,133
2002 DATA 10,169,74,133,11,76,64,64,0,0,0,0,0,0,0,0,0,0
2004 DATA 0,0,173,176,73,174,177,73,172,178,73,40,96
2006 DATA 234,234,234,234,234,234,234,234,234,234,234
2008 DATA 234,234,234,234,234,234,141,242,192,173,241,192
2010 DATA 41,128,201,128,208,247,173,240,192,141,162
2011 DATA 73,41,128,201,128,208,244,173
2012 DATA 162,73,41,15,129,10,164,10,200,132,10,208
2014 DATA 5,164,11,200,132,11,173,240,192,141,162,73
2016 DATA 41,64,201,64,208,244,173,162,73,41,15,129
2018 DATA 10,164,10,200,132,10,208,5,164,11,200,132
2020 DATA 11,173,240,192,141,162,73,41,32,201,32
2022 DATA 208,244,173,162,73,41,15,129,10,164,10
2024 DATA 200,132,10,208,5,164,11,200,132,11,173
2026 DATA 240,192,141,162,73,41,16,201,16,208,244
2028 DATA 173,162,73,41,15,129,10,164,10,200,132
2030 DATA 10,208,5,164,11,200,132,11,76,37,64

```

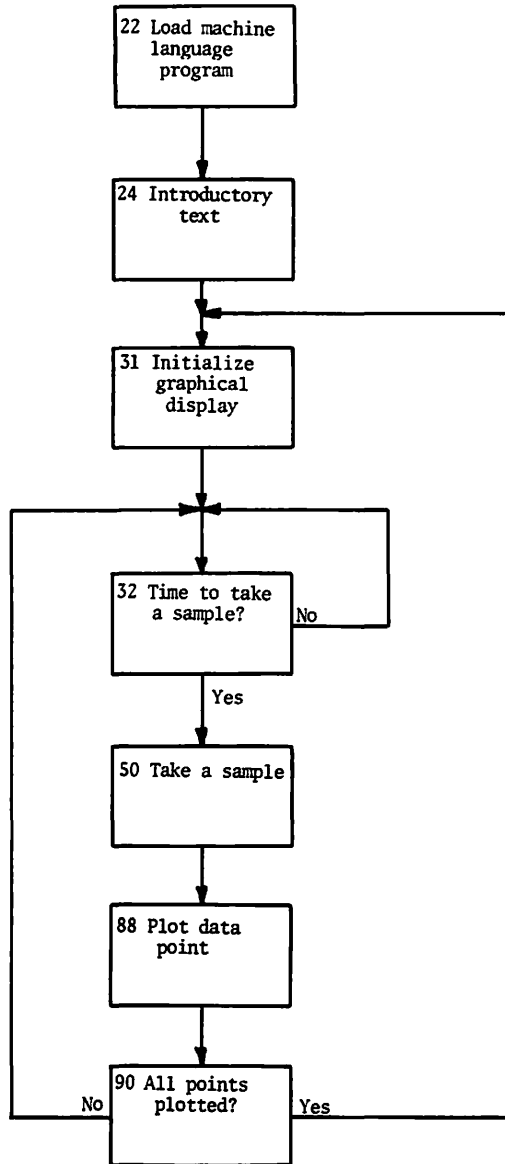


FIGURE 6.2 Flow chart of the BASIC program.

Figure 6.4 shows a diagrammatic representation of the experimental setup. You should tape the thermistor to the index finger on your dominant hand (the hand you write with). You will “see” temperature changes which are too small for you to detect, but

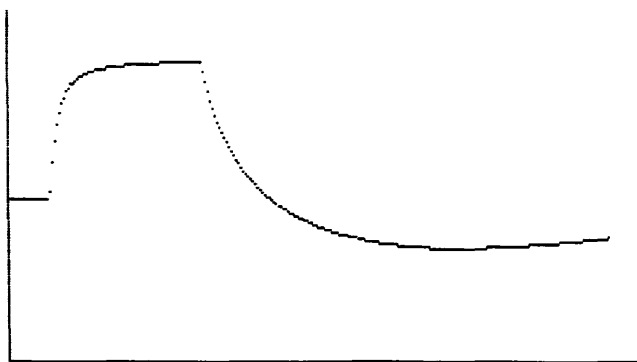
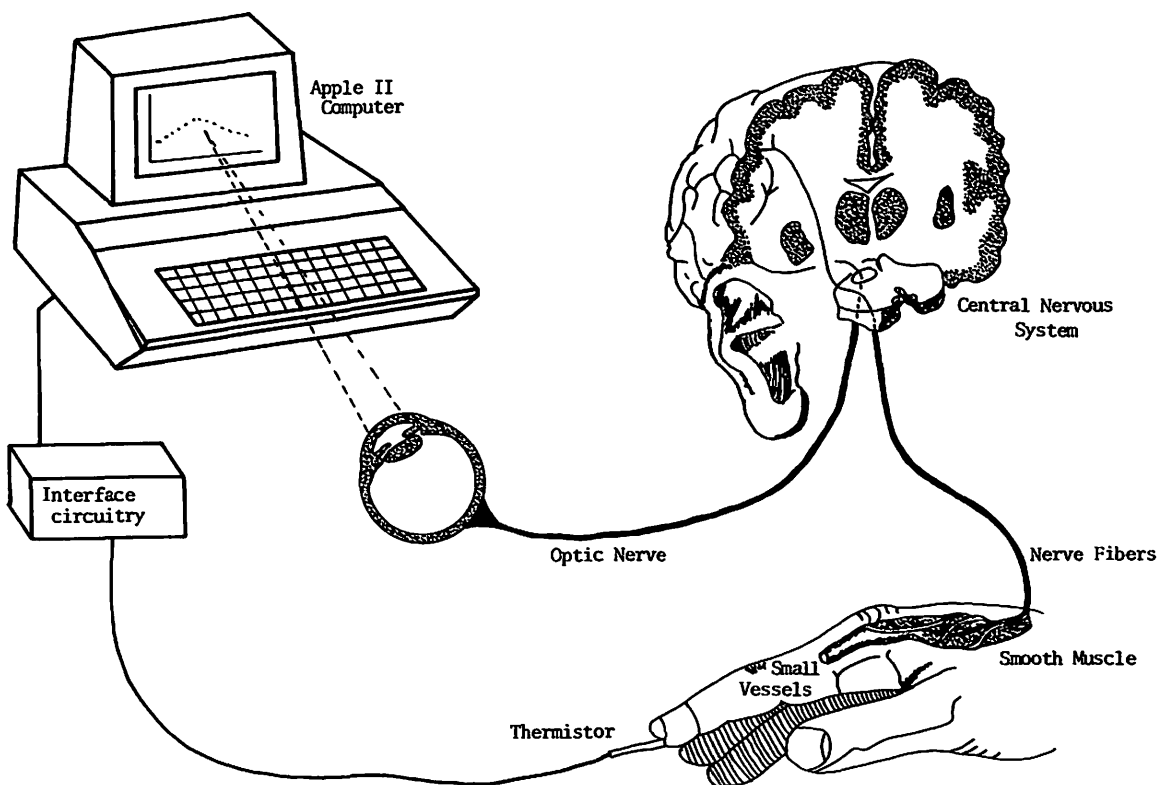


FIGURE 6.3 Graphic display resulting from placing the thermistor in the author's mouth and then removing it.

FIGURE 6.4 Diagrammatic representation of temperature feedback. The electronic measuring system informs the subject of temperature changes so small that they are normally not sensed.



which are easily detected by the thermistor. The graphic display on the CRT will serve as the feedback link to your central nervous system.

First seat yourself in a comfortable chair and, once you are relaxed, imagine that your hand has been placed in hot water. With some practice, you should be able to precisely control an increase or decrease in temperature. Within a few minutes you should be able to raise or lower your temperature one or two degrees merely by “telling” yourself to raise or lower your finger temperature. After some practice, you should be able to increase that variation to ten degrees.

You may want to monitor your temperature while watching a television program, or try taping the thermistor to another part of your body—such as your face—to see how vasoconstriction affects skin temperature in different locations.

HEART RATE

Heart rate changes as a function of many factors. Emotions such as anger and excitement cause an increase in heart rate. Emotions such as fear and grief cause a decrease in heart rate. The body has several regulatory mechanisms which monitor the concentration of carbon dioxide in the blood and the distention of blood vessels, and adjust heart rate to maintain cardiovascular system parameters within acceptable limits. We normally do not think of regulating our heart rate, primarily because there is not an obvious pathway for feedback. However, we can use an electronic measuring system to allow us to monitor our heart rate and to allow us to investigate normal responses to such things as breathing and relaxation.

We can conveniently get an indirect measurement of heart rate by using a system called *photoplethysmography*. Light can easily be transmitted through capillary beds existing in the earlobe or the fingertip. As arterial pulsations cause the blood volume in the capillary bed to increase and decrease, the amount of scattered, absorbed, and reflected light changes. This phenomena can be used to give us a signal which varies as a function of heart rate. Such a system is sensitive to motion artifacts and performs best when the subject is sitting quietly.

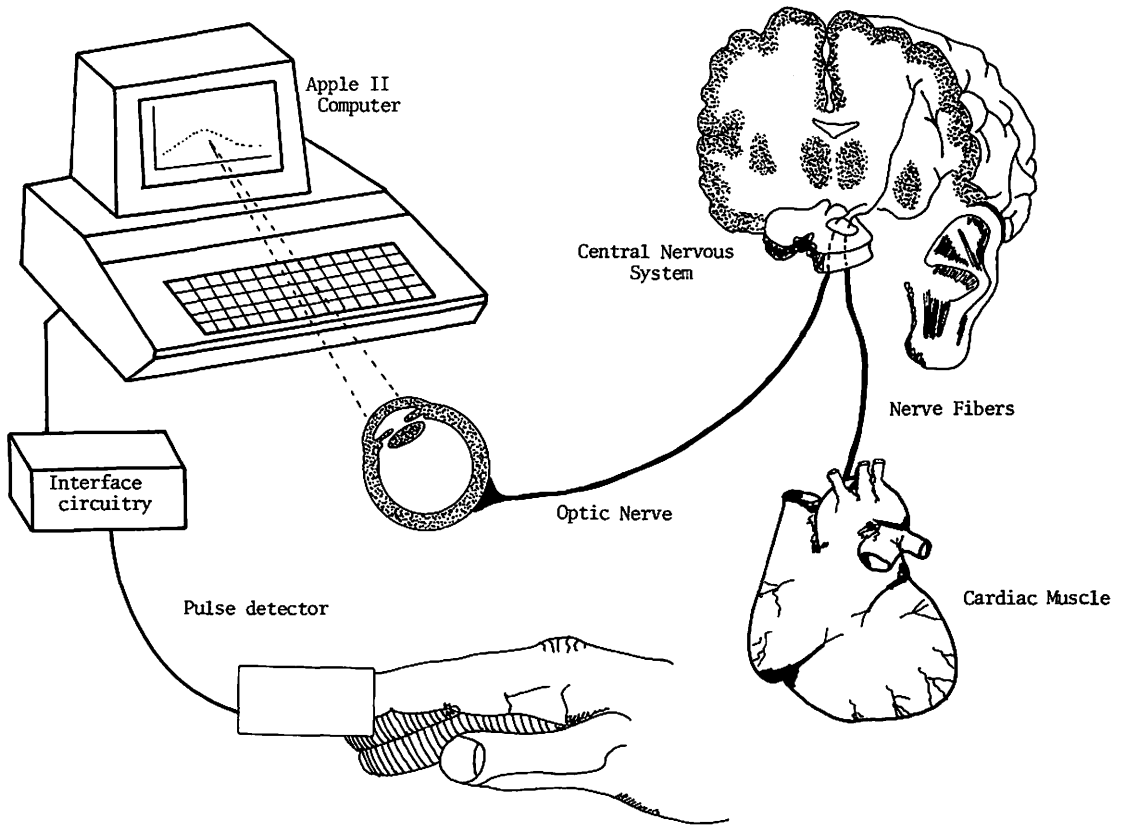


FIGURE 6.5 Diagrammatic representation of heart rate feedback. The electronic measuring system informs the subject of changes too small to be normally sensed.

Figure 6.5 shows a diagrammatic representation of the experimental setup. We will be using the capillary bed in your fingertip to monitor pulsations in blood volume due to contractions of the cardiac muscle. A pulse detector and an analog circuit will be used to detect, filter, and amplify the pulses so that they can be measured by the low speed A/D converter. The computer will then count the pulses and present your heart rate in the form of a graphic display.

Figure 6.6 shows a schematic diagram of the analog circuitry that we will use to interface the pulse detector to the low-speed A/D converter. You should notice that it is a modification of the circuit that we used for measuring skin temperature. This is primarily because a photocell, a device whose resistance changes

Number	Type	+5	-5	GND
IC1	AD580	1	--	3
IC2	CA3140T	7	4	--

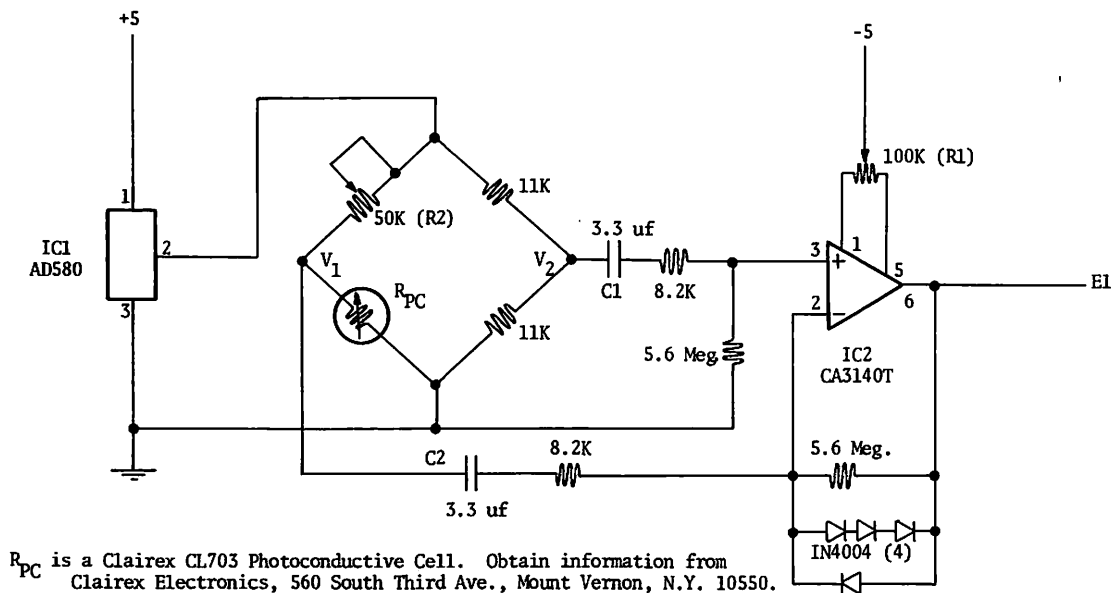


FIGURE 6.6 Schematic diagram of the analog circuitry for biofeedback heart rate monitor.

as a function of the intensity of light falling upon its surface, is being used to detect changes in light passing through the capillary bed in the finger. By putting the photocell into a bridge circuit, small changes in resistance that occur due to small changes in light intensity can be made to produce large changes in output voltage.

Looking at Figure 6.6, IC1 is a voltage regulator which provides a stable reference voltage to the bridge circuitry. Since we are counting pulses and not really measuring a value of resistance, we do not have to be overly concerned about output voltage drift due to internal heating of the photocell. Consequently, we do not have to reduce the output of the regulator, and can thus obtain increased sensitivity from the output of the bridge. IC2 is a high-gain, differential amplifier that is used to measure the difference between voltages V1 and V2. Capacitors C1 and C2 are included to

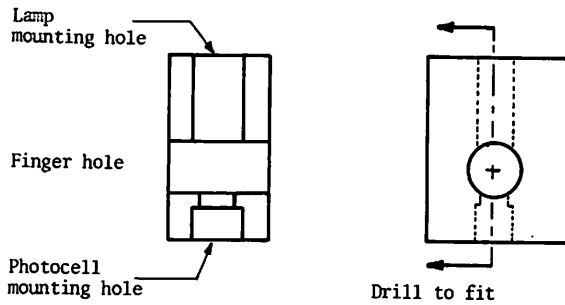


FIGURE 6.7a Position of the photocell relative to the lamp.

block the d.c. component of the amplifier to limit the output voltage, increasing the sharpness of the voltage change that occurs when the heart beats.

You should first concern yourself with the construction of the pulse detector. Figure 6.7a shows a drawing indicating the position of the photocell relative to the light source. The distance between the photocell and the light source will depend on the size of your fingers. To keep this distance to a minimum, make the hole just large enough to fit the smallest finger on your hand, usually the little finger. To get enough intensity from the lamp you will need a 3-volt d.c. supply that can provide about 250 milliamperes. You can use two D-cell batteries connected in series, or you can build a 5-volt supply from Appendix C and use a 10 ohm, 1 watt resistor in series with the lamp. I have shown the latter. To simplify construction as much as possible, I mounted the photocell and the lamp in a small block of wood.

Once you have built the pulse detector and the analog circuit, you should make the following adjustments and voltage checks:

1. The voltage at pin 2 on IC1 should be equal to + 2 volts.
2. Short the points labeled V1 and V2 together and adjust R1 until the voltage at E1 is equal to 0 volts.
3. Remove the short from the points labeled V1 and V2, and connect the output, E1, to the input of the A/D converter. Load and execute BASIC program 3 (See Listing 6.2). With the lamp disconnected from the lamp supply, you should see the graphics display periodically flashing. When the lamp is connected to its power supply, the flashing should stop.
4. Because of the simplicity of the design, you are going to have to develop some skill in positioning your finger in the pulse detector

LISTING 6.2 BASIC program for biofeedback heart rate system set-up.

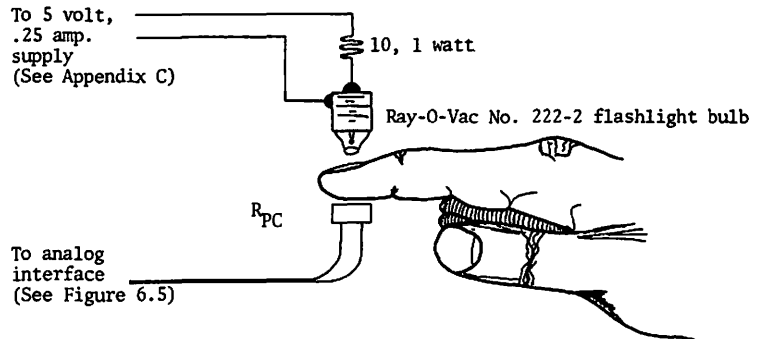
```

10 REM BIOFEEDBACK; HEART RATE SYSTEM SET UP
20 REM R. HALLGREN, APPLE II, 11-05-81
22 FOR I = 16384 TO 16458: READ D: POKE I,D: NEXT I
24 GOSUB 1000
200 REM SAMPLE INTERFACE
210 HOME : VTAB 12
220 CALL 16384
230 PRINT "THROB"
232 FOR I = 1 TO 100: NEXT I
240 GOTO 200
1000 REM DEMONSTRATION
1010 HOME
1020 PRINT "THIS PROGRAM WILL RESPOND TO THE PULSE"
1022 PRINT "DETECTOR OUTPUT.": PRINT ""
1024 PRINT "YOU WILL START THE TEST WITH THE LAMP": PRINT "TURNED OFF.":
PRINT "": PRINT ""
1026 PRINT "AFTER DETERMINING THAT THE INTERFACE IS"
1028 PRINT "PERFORMING O.K., YOU WILL TURN ON THE"
1030 PRINT "LAMP AND SEE IF THE SOFTWARE IS O.K.": PRINT "": PRINT ""
1032 PRINT "PRESS THE SPACE BAR TO CONTINUE"
1034 GET K$: IF K$ < " " THEN GOTO 1034
1099 RETURN
10000 DATA 141,176,73,142,177,73,140,178,73,8,141,242,192,173,241,192,4
1,128
10002 DATA 201,128,208,247,173,240,192,141,162,73,41,128,201,128,208,24
4,173
10004 DATA 162,73,41,4,201,0,208,223,173,240,192,141,162,73,41,32,201
,32,208,244
10006 DATA 173,162,73,41,15,201,2,48,202,173,176,73,174,177,73,172
,178,73,40,96

```

to ensure consistent operation. We learned in the previous section that cold hands are a good indication of poor circulation. Poor circulation means that blood flow is reduced. This means that the signals from the pulse detector will be small, resulting in erratic operation. If your hands are cold, hold them under warm water for a few minutes. Now, insert your little finger into the pulse detector. Figure 6.7b shows the relative position of the finger with respect to the photocell. As you insert your finger, you should see the display

FIGURE 6.7b Position of the finger relative to the photocell.



flash in response to the motion of your finger. After you have settled down, you should see the graphics display periodically start to pulse in response to the beat of your heart. Your finger should rest lightly upon the face of the photocell. Too much pressure will stop circulation and cause the display to skip beats.

5. Load and execute the BASIC program 4 (See Listing 6.3, Listing 6.4, and Figure 6.8). The program should flash in response to each heart beat, and should display a graphical representation of pulse rate in beats per minute. Figure 6.9a shows a graphical display of the author's resting heart rate.

LISTING 6.3 BASIC program for biofeedback heart rate experiments.

```

10 REM BIOFEEDBACK; HEART RATE
20 REM R. HALLGREN, APPLE II, 11-05-81
21 DIM V(300):V(1) = 12:V(2) = 12
22 FOR I = 16384 TO 16459: READ D: POKE I,D: NEXT I
23 POKE 1022,00: POKE 1023,03: POKE 1016,72: POKE 1017,8
24 POKE 1018,88: POKE 1019,40: POKE 1020,104: POKE 1021,96
26 POKE 768,8: POKE 769,238: POKE 770,15: POKE 771,3: POKE 772,40: POKE
  773,165: POKE 774,69: POKE 775,64
27 CALL 1016
29 GOSUB 1000
30 HGR : HCOLOR= 3
32 HPLLOT 0,0 TO 0,150: HPLLOT TO 270,150
50 REM AVERAGE HEART RATE
52 FOR I = 3 TO 270
53 Q = 0: POKE 783,0
56 GOSUB 1100
58 Q = Q + 1
60 X = PEEK (783): IF X < 30 THEN GOTO 56
70 V(I) = (V(I - 1) + V(I - 2) + 2 * Q) / 3
80 HPLLOT I,(150 - 6 * V(I))
90 NEXT I
200 REM PLOT AVERAGE
1000 REM INTRODUCTION
1010 HOME
1020 PRINT "THIS IS A DEMONSTRATION PROGRAM FOR"
1022 PRINT "CONTROLLING HEART RATE USING"
1024 PRINT "BIOFEEDBACK IN CONJUNCTION WITH THE LOW"
1026 PRINT "SPEED A/D CONVERTER AND A DATA PLOT": PRINT "ROUTINE.": PRINT
  "": PRINT ""
1028 PRINT "THE PROGRAM WILL PLOT A POINT": PRINT "10 SECONDS.": PRINT "
  ": PRINT ""
1030 PRINT "PRESS THE SPACE BAR TO CONTINUE."
1032 GET K$: IF K$ < > " " THEN GOTO 1032
1099 RETURN
1100 REM SAMPLE PULSE DETECTOR
1110 CALL 16384
1120 FOR R = 1 TO 50: NEXT R
1199 RETURN
10000 DATA 141,176,73,142,177,73,140,178,73,88,141,242,192,173,241,
  192,41,128
10002 DATA 201,128,208,247,173,240,192,141,162,73,41,128,201,128,208,24
  4,173
10004 DATA 162,73,41,4,201,0,208,223,173,240,192,141,162,73,41,32,201,3
  2,208,244
10006 DATA 173,162,73,41,15,201,2,48,202,173,176,73,174,177,73,172,178,
  73,40,96

```

```

4000- 8D B0 49 STA $49B0
4003- 8E B1 49 STX $49B1
4006- 8C B2 49 STY $49B2
4009- 08 PHP
400A- 8D F2 C0 STA $C0F2
400D- AD F1 C0 LDA $C0F1
4010- 29 80 AND #$80
4012- C9 80 CMP #$80
4014- D0 F7 BNE $400D
4016- AD F0 C0 LDA $C0F0
4019- 8D A2 49 STA $49A2
401C- 29 80 AND #$80
401E- C9 80 CMP #$80
4020- D0 F4 BNE $4016
4022- AD A2 49 LDA $49A2
4025- 29 04 AND #$04
4027- C9 00 CMP #$00
4029- D0 DF BNE $400A
402B- AD F0 C0 LDA $C0F0
402E- 8D A2 49 STA $49A2
4031- 29 20 AND #$20
4033- C9 20 CMP #$20
4035- D0 F4 BNE $402B
4037- AD A2 49 LDA $49A2
403A- 29 0F AND #$0F
403C- C9 02 CMP #$02
403E- 30 CA BMI $400A
4040- AD B0 49 LDA $49B0
4043- AE B1 49 LDX $49B1
4046- AC B2 49 LDY $49B2
4049- 28 PLP
404A- 60 RTS
404B- 00 BRK
404C- 00 BRK
404D- 00 BRK
404E- 00 BRK
404F- 77 ???
4050- 9F ???
4051- FF ???
4052- 80 ???

```

LISTING 6.4 Machine language routine contained within the BASIC program.

Once you have confirmed that the system is working correctly, insert your finger into the pulse detector and we will learn some things about your cardiovascular system. Remember that the detector is motion-sensitive, so you should assume a quiet sitting position with your arm and hand supported on some surface. The first thing you will notice is that your heart rate is a function of respiration. When you inhale your heart rate decreases. Also, if you hold your breath for several seconds, you should expect to see a decrease followed by an increase in heart rate. The graphical display on the CRT will be used as the feedback path to your central nervous system. You can alter your heart rate by practicing progressive relaxation exercises. The idea behind these exercises is to learn how to relax by sensing the difference between tension and relaxation. Start by assuming a comfortable position. Starting

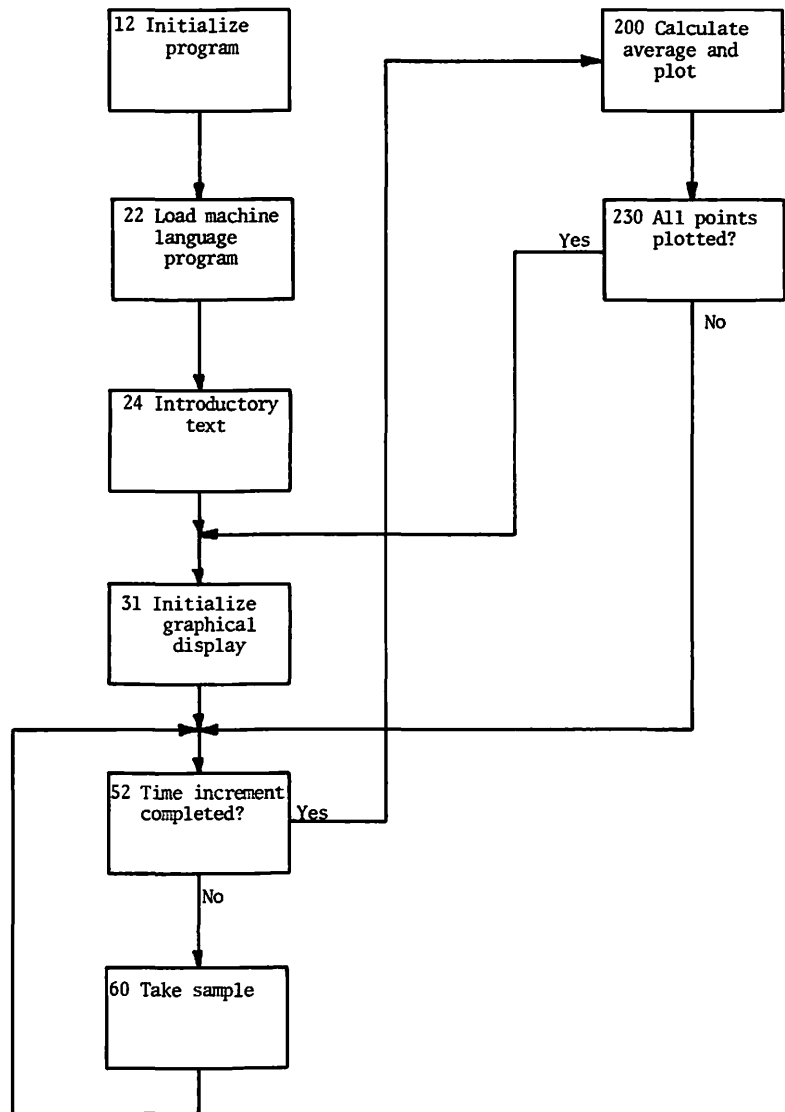
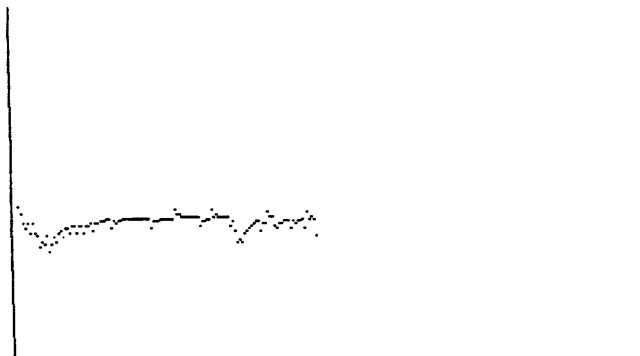


FIGURE 6.8 Flow chart for the BASIC program.

FIGURE 6.9 Graphical display of resting level heart rate.



with the feet, alternately tense and relax your muscles. Suppose you have tensed the muscles in your left leg. When you relax that group of muscles you should feel the leg become limp and heavy. As you progress through this type of relaxation process, your heart rate should decrease, indicating a state of relaxation.

You may want to monitor your heart rate while watching your favorite television program. Sporting events or mystery shows are good candidates for causing your heart rate to significantly increase and decrease. Monitor your pulse rate while you drink a cup of coffee or as you smoke a cigarette. For the more daring individual, you might want to monitor your heart rate as you balance your checkbook at the end of the month.

7

CONTROLLING A VIDEO PLAYBACK DEVICE

The phrase “one picture is worth a thousand words” illustrates the importance of visual representation of material in the learning process. Educators have effectively used snapshots, slides, film-strips, and motion pictures to aid students while avoiding some of the problems that might occur from direct contact with the subject being studied. While this media serves a purpose, its effectiveness is limited since the viewing speed and the content of the material is fixed, and there is no simple means of allowing individual students to interact with the viewing device.

In an attempt to increase the personal interaction of teaching systems with the student, slide/audio cassette systems have been developed. These systems have been used successfully for many years as a low-cost supplement to textbook instruction at all educational levels. Using individual viewing screens and head-

phones, the student can view and review such things as sequential anatomical dissections, while listening to a detailed description of what they are seeing.

Television has expanded on slide/audio tape systems by allowing the low-cost presentation of prepared lecture material and permitting live demonstrations of such things as surgical procedures to large audiences. The introduction of the videotape recorder has increased the impact of television by allowing lecture and demonstration materials to be recorded and then played back any number of times. Recently, the less expensive videocassette player has made it possible for students to individually view and review prerecorded lecture material. The intimacy of videocassette presentations is surpassed only by the live classroom lecture where the important ingredient of student-teacher interaction is present.

The availability of low-cost personal computing systems has provided another (and more sophisticated) tool for educators. Computer-assisted instruction (CAI) is a term used for a system in which the computer carries on a predetermined instructional strategy, and permits conversational interaction with the student. Computers are most recognized for their high-speed computational ability. Because of this, computers can be used in real-time simulation of complex physical systems and, when interfaced with a graphic display, the computer can be very effective in assisting students in their understanding of these systems.

While the use of the small personal computer in an educational environment is not a new concept, the use of a personal computer to interactively control a videocassette player is. Videotaped material is abundant and readily available. Often the student will not need to review a complete lecture, but will only be concerned with a specific segment. In a case such as this, the computer can be used to provide a menu from which the student can choose the desired segment. The computer then is responsible for finding the starting point of the desired segment, initiating the PLAY command, and stopping the presentation when the end of the segment is reached. The interaction of the computer and the video device can also be used as a teaching machine where the computer is used to control the presentation of videotaped lectures. At appropriate times, the lecture could be interrupted and questions asked of the viewer. Depending on the response, the

viewer would be allowed to continue, would be supplied with remedial material, or would just be returned to the start of the most recently viewed segment. This provision of immediate feedback is an important part of the system concept and is a particularly strong feature of computer-assisted instruction. Theoretically, this system should have the following advantages over conventional methods of instruction such as a group lecture:

1. Students can proceed through material at a rate which is consistent with their ability.
2. Correct responses to questions allow additional material to be presented. Incorrect responses cause material to be repeated, or supplementary material to be presented.
3. Material presentation can be organized so that special guidance can be given to correct deficiencies in individual educational backgrounds.
4. Teaching material can be organized for an optimal sequence of presentation.
5. The student is forced to play an active part in the learning process.

The design of the Sony Betamax SL0-320 videocassette recorder makes interfacing the Apple II computer to it a relatively easy task. Table 7.1 shows the control, status, and timing lines that are available from the interface connector on the back of the Betamax. During the recording process, a timing signal is recorded onto the videotape. This timing signal, available on pin 15 of the Auto Search control unit interface connector, can be sampled by the computer and used to index the tape to any relative location. There are two status signals that are available at the interface connector: one signal (CASSETTE IN*) indicates when the video-

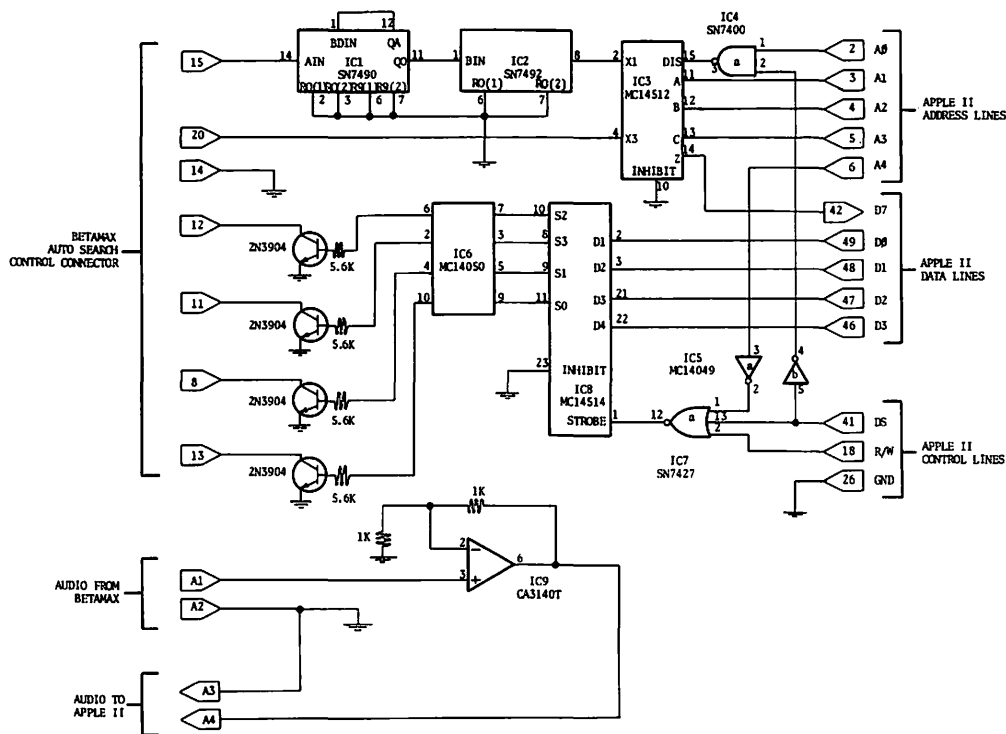
TABLE 7.1 Betamax status and control signals available through the pins that connect the recorder to the RM-300 Auto Search control unit.

Betamax Connector Pin	Signal	Source/ Destination
CN1-20	BEGINNING OF TAPE*	Status signal from Betamax
CN1-7	CASSETTE IN*	Status signal from Betamax
CN1-11	REWIND*	Command signal to Betamax
CN1-8	STOP*	Command signal to Betamax
CN1-13	PLAY*	Command signal to Betamax
CN1-12	FAST FORWARD*	Command signal to Betamax
CN1-15	COUNT*	Timing signal from Betamax

cassette has been inserted into the player and the other signal (BEGINNING OF TAPE*) indicates when the videocassette tape has been fully rewound. These signals were not used in the system that is being described, but certainly could be used if a modification of the system concept makes it desirable. Four control lines (STOP*, PLAY*, FAST FORWARD*, and REWIND*) are also available at the interface connector, and are used to indicate to the Betamax the desired mode of operation.

Figure 7.1 shows the Betamax to Apple II interface in schematic form. The left side of the schematic shows connections

FIGURE 7.1 Betamax/Apple II interface schematic.



Number	Type	+5	-5	GND
IC1	SN7490	5	---	10
IC2	SN7492	5	---	10
IC3	MC14512	10	---	8
IC4	SN7400	14	---	7
IC5	MC14049	1	---	8
IC6	MC14050	1	---	8
IC7	SN7427	14	---	7
IC8	MC14514	24	---	12
IC9	CA3140T	7	4	---

made to the Betamax through the RM-300 Auto Search control unit connector, while the right side of the schematic shows the connections made to the Apple II through the I/O bus connector. Line 15 (COUNT*) from the Betamax carries the timing signal formatted onto the videotape. This signal is divided by a factor of sixty by the combined action of integrated circuits IC1 (SN7490) and IC2 (SN7492). IC3 (MC14512) is an eight-channel data selector which is used to connect the divided timing signal to the Apple II data line seven (D7). Increased channel sampling capability was incorporated so that the two status lines from the Betamax, or other user-defined status lines, could be sampled if so desired. Data line D7 was chosen because its state is easily tested by rotating its contents into the carry bit. Integrated circuit IC7 (MC14514) is a 4-16 line decoder latch which is used to selectively turn on one of the four transistors, thus causing the Betamax to enter either the PLAY, the REWIND, the FAST FORWARD, or the STOP mode. Table 7.2 shows the function that will be accessed for a given BASIC statement or a given machine language command. Provision was made for storing the computer program on a beginning segment of the videotape. To record the program on the videotape you should amplify the output of the computer going to the cassette recorder during a SAVE operation by a factor of ten. Figure 7.2 shows a simple circuit for doing this (refer to Appendix B for a description of the circuit). In order for the audio to be recorded, you will have to provide both the audio signal and a video signal. IC12 (MC14528) is a dual monostable multivibrator configured so that it switches the Betamax into the PLAY mode

TABLE 7.2 Software commands, in both BASIC and 6502 machine language, necessary to activate the Betamax control lines.

Control Signal	BASIC Command	Machine Language Command
REWIND*	POKE-16142,3	LDA #03 STA \$C0F2
STOP*	POKE-16142,0	LDA #00 STA \$C0F2
PLAY*	POKE-16142,1	LDA #01 STA \$C0F2
FAST FORWARD*	POKE-16142,2	LDA #02 STA \$C0F2

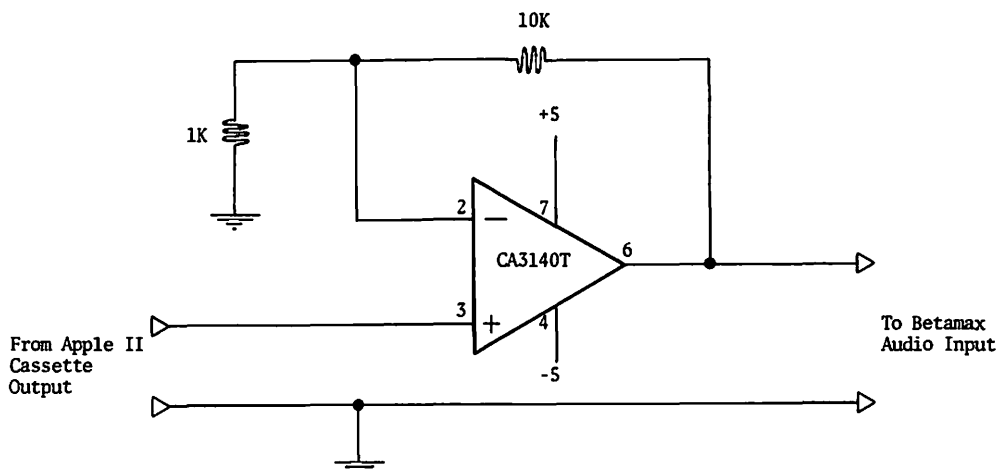


FIGURE 7.2 Operational amplifier circuit for recording Apple II cassette output into the video tape.

when LOAD is entered from the Apple II keyboard, and switches it into the STOP mode when the program has been loaded. IC11 (CA3140T) buffers the audio output from the Betamax to the Apple II cassette audio input during the program load operation.

Once you have completed construction of the interface, you should perform the following tests to make sure that it is working correctly:

1. For each of the Betamax control functions listed in Table 7.2, execute the two BASIC commands associated with the control function. The Betamax should respond in an appropriate manner.
2. Load the Apple II with the test program that is shown in Listing 7.1. Using a prerecorded video tape placed in the cassette player, execute the program. The program should exercise the following player functions:
FAST FORWARD
PLAY
STOP
REWIND
STOP
3. If everything functions correctly up to this point, try storing the test program on the beginning segment of a blank videocassette. Remember to connect the audio cables from the Apple II to the Betamax and back again. Remember also to provide some type of video

LISTING 7.1 Betamax/Apple II test program.

```

10 REM TEST PROGRAM FOR THE BETAMAX/APPLE II INTERFACE
50 REM START
51 GOSUB 12000: REM LOAD MACHINE LANGUAGE ROUTINE
52 POKE 771,0: POKE 772,0
53 POKE - 16142,14: HOME
70 X = PEEK ( - 16137)
72 IF X < 127 THEN GOTO 80
74 POKE - 16142,3
76 GOTO 70
80 POKE - 16142,14
1000 REM EXERCISE BETAMAX CONTROL FUNCTIONS
1010 SR = 100:SP = 150
1011 HOME
1012 GOTO 10000
1014 FOR I = 1 TO 2000: NEXT I
1020 SR = 100:SP = 100
10000 REM CONTROLLER ROUTINE
10002 R1 = 0:R2 = 0:P1 = 0:P2 = 0
10010 X = PEEK (771) + 256 * PEEK (772)
10020 R1 = SR
10021 R1 = R1 - 256
10022 IF R1 = 0 THEN GOTO 10030
10023 IF R1 > 0 THEN GOTO 10032
10024 IF R1 < 0 THEN GOTO 10035
10030 R2 = R2 + 1: GOTO 10090
10032 R2 = R2 + 1: GOTO 10021
10035 R1 = R1 + 256
10050 P1 = SP
10051 P1 = P1 - 256
10052 IF P1 = 0 THEN GOTO 10080
10053 IF P1 > 0 THEN GOTO 10082
10054 IF P1 < 0 THEN GOTO 10085
10080 P2 = P2 + 1: GOTO 10090
10082 P2 = P2 + 1: GOTO 10051
10085 P1 = P1 + 256
10090 IF X < SR THEN GOTO 10100
10092 IF X > SR THEN GOTO 10200
10094 IF X = SR THEN GOTO 10300
10100 REM FAST FORWARD
10110 POKE 773,R1: POKE 774,R2
10112 POKE - 16142,2: CALL 784
10114 POKE - 16142,0
10116 GOTO 10300
10200 REM REWIND
10202 R3 = 0:R4 = 0
10210 R3 = SR + 2
10211 R3 = R3 - 256
10212 IF R3 = 0 THEN GOTO 10220
10213 IF R3 > 0 THEN GOTO 10230
10214 IF R3 < 0 THEN GOTO 10240
10220 R4 = R4 + 1: GOTO 10250
10230 R4 = R4 + 1: GOTO 10211
10240 R3 = R3 + 256
10250 POKE 773,R3: POKE 774,R4
10252 POKE - 16142,3: CALL 864
10254 POKE - 16142,0
10256 POKE 771,R1: POKE 772,R2
10300 REM PLAY
10310 POKE 773,P1: POKE 774,P2
10320 POKE - 16142,1: CALL 784
10400 REM STOP
10410 POKE - 16142,0
10412 POKE - 16142,14

```

LISTING 7.1 (Continued)

```

10420 RETURN
11000 REM GRAPHIC INTRODUCTION
11012 HOME : GR : COLOR= 2
11020 VLIN 0,11 AT 0: VLIN 0,11 AT 12: PLOT 1,0: PLOT 2,2: PLOT 3,4: PLOT
      4,6: PLOT 5,8: PLOT 6,10: PLOT 7,8: PLOT 8,6: PLOT 9,4: PLOT 10,2: PLOT
      11,0: PLOT 6,11: REM M
11025 VLIN 0,11 AT 0: VLIN 0,5 AT 15: HLIN 15,25 AT 5: VLIN 6,11 AT 25: HLIN
      25,15 AT 11: HLIN 15,25 AT 0: REM S
11030 VLIN 0,11 AT 29: VLIN 0,11 AT 39: HLIN 29,39 AT 11: REM U
11035 VLIN 28,39 AT 0: HLIN 1,10 AT 28: HLIN 0,10 AT 39: REM C
11040 HLIN 13,23 AT 28: HLIN 13,23 AT 39: VLIN 28,39 AT 13: VLIN 28,39 AT
      23: REM O
11045 VLIN 28,39 AT 27: VLIN 28,39 AT 39: PLOT 28,28: PLOT 29,30: PLOT 3
      0,32: PLOT 31,34: PLOT 32,36: PLOT 33,38: PLOT 33,39: PLOT 34,36: PLOT
      35,34: PLOT 36,32: PLOT 37,30: PLOT 38,28: REM M
11090 GET A$
11099 RETURN
12000 REM MACHINE LANGUAGE ROUTINE
12010 POKE 768,234: POKE 769,234: POKE 770,234: POKE 771,234: POKE 772,2
      34: POKE 773,234: POKE 774,234: POKE 775,234: POKE 776,234: POKE 77
      ,234: POKE 778,234: POKE 779,234
12012 POKE 780,234: POKE 781,234: POKE 782,234: POKE 783,234
12020 POKE 784,008: POKE 785,072: POKE 786,173: POKE 787,243: POKE 788,1
      92: POKE 789,042: POKE 790,176: POKE 791,250
12025 POKE 792,173: POKE 793,243: POKE 794,192: POKE 795,042: POKE 796,1
      44: POKE 797,250: POKE 798,238: POKE 799,003
12030 POKE 800,003: POKE 801,208: POKE 802,003: POKE 803,238: POKE 804,0
      04: POKE 805,003: POKE 806,173: POKE 807,003
12035 POKE 808,003: POKE 809,205: POKE 810,005: POKE 811,003: POKE 812,2
      08: POKE 813,228: POKE 814,173: POKE 815,004
12040 POKE 816,003: POKE 817,205: POKE 818,006: POKE 819,003: POKE 820,2
      08: POKE 821,220: POKE 822,104: POKE 823,040: POKE 824,096
12045 POKE 864,008: POKE 865,072: POKE 866,173: POKE 867,243: POKE 868,1
      92: POKE 869,042: POKE 870,176: POKE 871,250
12050 POKE 872,173: POKE 873,243: POKE 874,192: POKE 875,042: POKE 876,1
      44: POKE 877,250: POKE 878,056: POKE 879,173
12055 POKE 880,003: POKE 881,003: POKE 882,233: POKE 883,001: POKE 884,1
      41: POKE 885,003: POKE 886,003: POKE 887,173
12060 POKE 888,004: POKE 889,003: POKE 890,233: POKE 891,000: POKE 892,1
      41: POKE 893,004: POKE 894,003: POKE 895,173
12065 POKE 896,003: POKE 897,003: POKE 898,205: POKE 899,005: POKE 900,0
      03: POKE 901,208: POKE 902,219: POKE 903,173
12070 POKE 904,004: POKE 905,003: POKE 906,205: POKE 907,006: POKE 908,0
      03: POKE 909,208: POKE 910,211: POKE 911,104
12075 POKE 912,040: POKE 913,096
12999 RETURN

```

signal to the Betamax during the recording process. If you now rewind the tape and enter LOAD, you should see the Betamax begin to play, and the computer should sound a tone, indicating that a normal load is taking place.

The software portion of the controller was written in two parts:

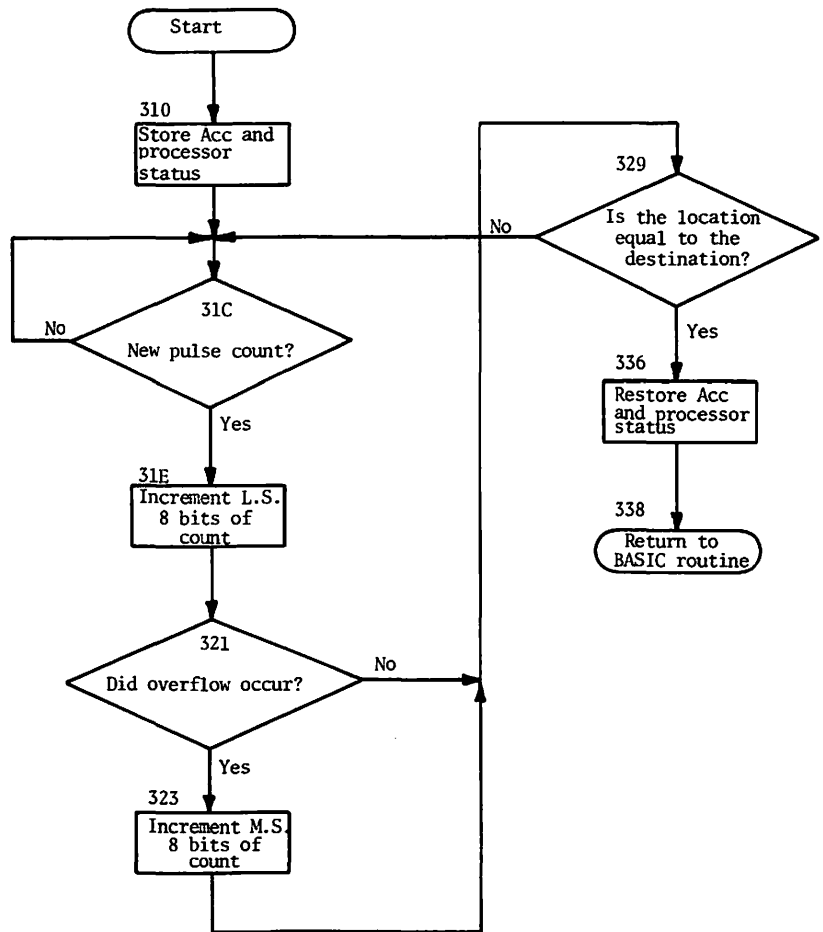
1. Two machine language routines were written to count the pulses coming from the timing track stored on the videotape, and to determine when the desired destination on the videotape had been

reached. One routine is used when the tape is moving forward and the other routine is used when the tape is moving backward.

2. A routine was written in Applesoft BASIC to load the desired tape destination, and to control the operational mode of the cassette player.

Figure 7.3 shows a flow chart of the machine language routine which is used when the tape is moving forward, and Listing 7.2 shows the actual program with comments. Figure 7.4 shows a flow chart of the machine language routine which is used when

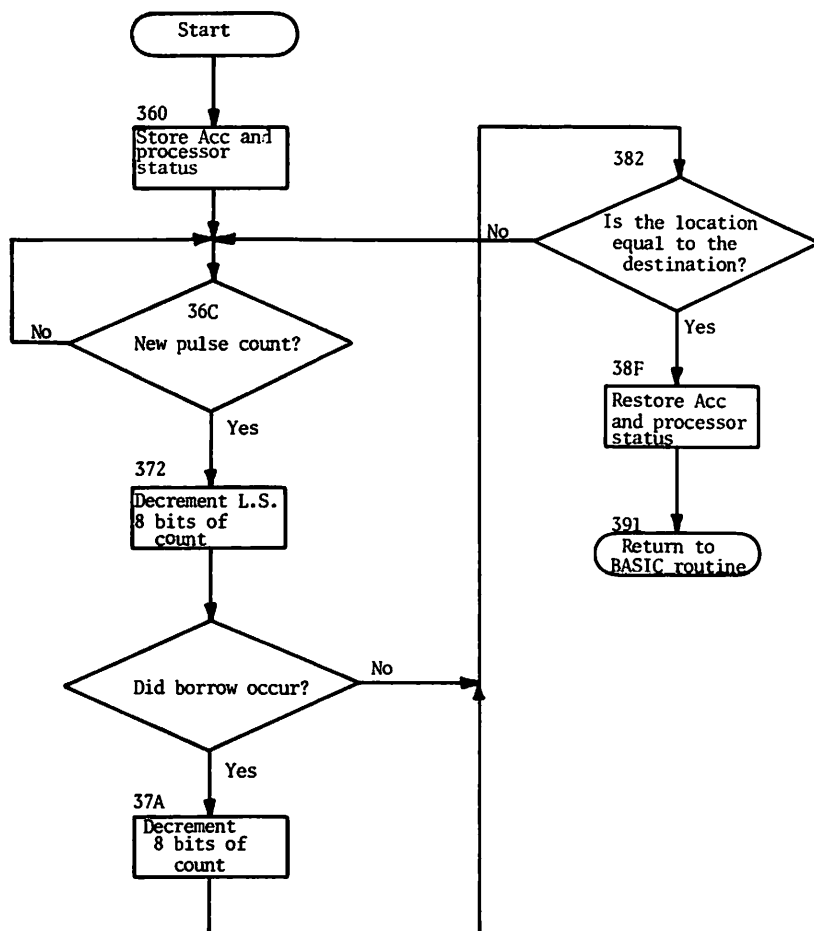
FIGURE 7.3 Flow chart of the machine language routine for incrementing and comparing the videotape location count.



LISTING 7.2 Machine language program for incrementing and comparing the videotape location count.

310	08	PHP		Save processor status
311	48	PHA		Save Accumulator
312	ADF3C0	LDA	\$C0F3	Save processor status
315	2A	ROL		Rotate bit 7 into carry
316	B0FA	BCS	\$312	Branch if carry set
318	ADF3C0	LDA	\$C0F3	Sample count line
31B	2A	ROL		Rotate bit 7 into carry
31C	90FA	BCC	\$318	Branch if carry clear
31E	EE0303	INC	\$303	Increment least sig bit
321	D003	BNE	\$326	Branch if no overflow
323	EE0403	INC	\$304	Increment most sig 8 bits
326	AD0303	LDA	\$303	Load least sig 8 bits
329	CD0503	CMP	\$305	Compare with stop count
32C	D0E4	BNE	\$312	Branch if not equal
32E	AD0403	LDA	\$304	Load most sig 8 bits
331	CD0603	CMP	\$306	Compare with stop count
334	D0DC	BNE	\$312	Branch if not equal
336	68	PLA		Restore accumulator
337	28	PLP		Restore processor status
338	60	RTS		Return form subroutine

FIGURE 7.4 Flow chart of the machine language routine for decrementing and comparing the videotape location count.



the tape is moving backward, and Listing 7.3 shows the actual program with comments. Upon entering either routine, the processor status and the accumulator are pushed onto the stack. The line containing the pulses from the timing track on the video tape is then sampled until it has been determined that the tape has moved a distance equal to one pulse width. A register containing two 8-bit words is then either incremented or decremented depending on whether the tape is being moved forward or in reverse. The contents of this register is then compared to the contents of a register containing the two 8-bit words representing the destination. If the two registers are equal, the tape has reached its destination and the computer returns to the BASIC routine. If the two registers are not equal, the program goes back to wait for the next timing pulse.

Figure 7.5 shows a flow chart of a BASIC routine which has been used for demonstration purposes, and Listing 7.4 shows the actual program with comments. There is an initial decision point at line 232 where a student decides whether he or she needs to watch the whole program or just selected parts. After viewing a segment of instructional material, the student is asked a series of

LISTING 7.3 Machine language program for decrementing and comparing the videotape location count.

Hex Adr	Code	Mnemonic	Comments
360	08	PHP	Save processor status
361	48	PHA	Save accumulator
362	ADF3C0	LDA \$C0F3	Sample count line
365	2A	ROL	Rotate bit 7 into carry
366	B0FA	BCS \$362	Branch if carry set
368	ADF3C0	LDA \$C0F3	Sample count line
36B	2A	ROL	Rotate bit 7 into carry
36C	90FA	BCC \$368	Branch if carry clear
36E	38	SEC	Set carry
36F	AD0303	LDA \$303	Load least sig 8 bits
372	E901	SBC #\$01	Decrement least sig 8 bits
374	8D0303	STA \$303	Save
377	AD0403	LDA \$304	Load most sig 8 bits
37A	E900	SBC #\$00	Decrement if borrow
37C	8D0403	STA \$304	Save
37F	AD0303	LDA \$303	Load least sig 8 bits
382	CD0503	CMP \$305	Compare with stop count
385	D0DB	BNE \$362	Branch if not equal
387	AD0403	LDA \$304	Load most sig 8 bits
38A	CD0603	CMP \$306	Compare with stop count
38D	D0D3	BNE \$362	Branch if not equal
38F	68	PLA	Restore accumulator
390	28	PLP	Restore processor status
391	60	RTS	Return from subroutine

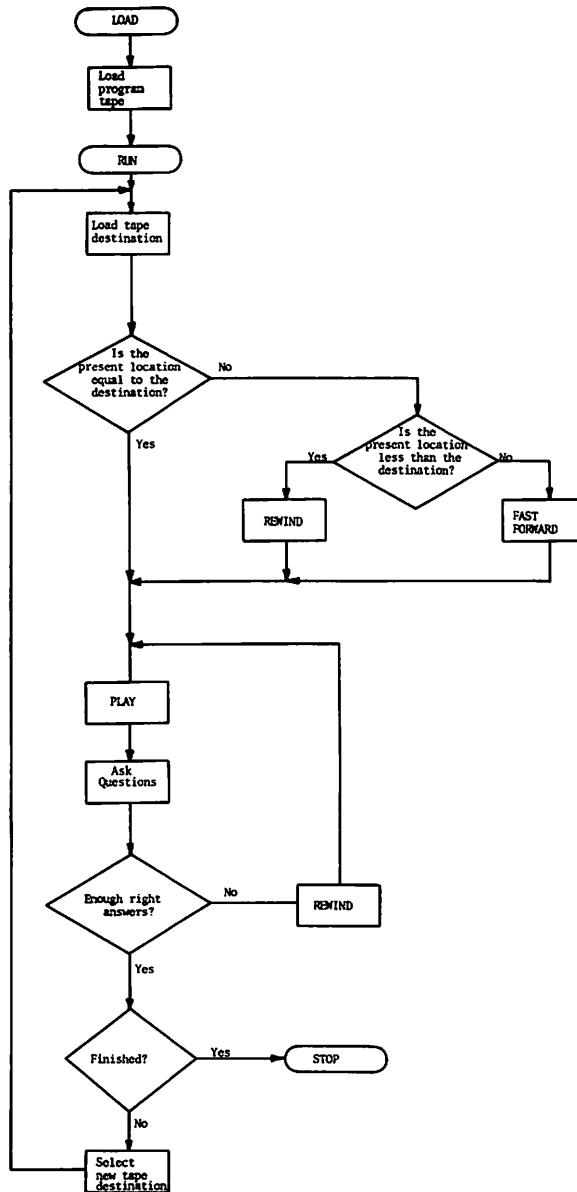


FIGURE 7.5 Flow chart for a demonstration program written in BASIC.

questions designed to test retention and comprehension of the material which has been viewed. If retention and/or comprehension is below a specified level, the videotape is rewound to the segment start point and the material is re-presented to the student. Once comprehension and retention are demonstrated, the student is allowed to continue on to new material.

LISTING 7.4 Betamax/Apple II demonstration routine: Hypercalcemia.

```

50 REM START
51 GOSUB 12000: REM LOAD MACHINE LANGUAGE ROUTINE
52 POKE 771,0: POKE 772,0
53 POKE - 16142,14: HOME
54 X = PEEK ( - 16133)
56 IF X < 127 THEN GOTO 70
58 VTAB 10
60 PRINT "!!!!!!!!!!LOAD TAPE!!!!!!!!!!"
62 GOTO 54
70 X = PEEK ( - 16137)
72 IF X < 127 THEN GOTO 80
74 POKE - 16142,3
76 GOTO 70
80 POKE - 16142,14
100 REM PROGRAM START
102 TEXT : HOME
106 GOSUB 11000: REM GRAPHIC INTRODUCTION
200 REM WHOLE PROGRAM OR PARTS?
201 TEXT : HOME : VTAB 8
204 PRINT "*****"
205 PRINT "*"
206 PRINT "* THE FOLLOWING IS A DEMONSTRATION *"
208 PRINT "*"
210 PRINT "* OF THE USE OF A VIDEO TAPE *"
212 PRINT "*"
214 PRINT "* RECORDER IN INTERACTIVE COMPUTER *"
216 PRINT "*"
218 PRINT "* INSTRUCTION. *"
220 PRINT "*"
222 PRINT "*****"
223 CALL - 198
224 VTAB 24: PRINT "PRESS ANY KEY TO CONTINUE.": GET A$
230 HOME : VTAB 6
232 PRINT "*****"
234 PRINT "*"
236 PRINT "* YOU HAVE THE CHOICE OF LOOKING *"
238 PRINT "*"
240 PRINT "* AT THE WHOLE VIDEO TAPE OR AT *"
242 PRINT "*"
244 PRINT "* SELECTED PARTS. *"
246 PRINT "*"
248 PRINT "* WOULD YOU LIKE TO VIEW THE WHOLE *"
250 PRINT "*"
252 PRINT "* TAPE? *"
256 PRINT "*"
258 PRINT "*****"
260 VTAB 24: PRINT "PRESS Y FOR YES OR N FOR NO.": CALL - 198: GET A$
261 IF A$ = "*" THEN GOTO 50
262 IF A$ = "Y" THEN GOTO 1000
264 IF A$ = "N" THEN GOSUB 2000
265 GOTO 230
266 VTAB 24: PRINT "PRESS Y FOR YES OR N FOR NO!": CALL - 198: GET A$: GOTO
262
867 ,3: POKE 868,233: POKE 869,1
1000 REM WHOLE TAPE PRESENTATION
1002 HOME
1010 SR = 0:SP = 85
1012 GOSUB 10000
1014 GOSUB 2200
1020 SR = 92:SP = 114
1022 GOSUB 10000
1024 GOSUB 3100
1030 SR = 122:SP = 172
1032 GOSUB 10000

```

LISTING 7.4 (Continued)

```

1034 GOSUB 4100
1040 SR = 179:SP = 236
1042 GOSUB 10000
1044 GOSUB 5100
1050 SR = 241:SP = 308
1052 GOSUB 10000
1054 GOSUB 6100
1060 SR = 314:SP = 440
1062 GOSUB 10000
1064 GOSUB 7100
1070 SR = 446:SP = 822
1072 GOSUB 10000
1074 GOSUB 8100
1080 SR = 829:SP = 1100
1082 GOSUB 10000
1084 GOSUB 9100
1090 GOTO 230
1999 END
2000 REM SELECTED TAPE PRESENTATION
2002 HOME
2010 PRINT "*****"
2012 PRINT "*"
2014 PRINT "*" CHOSE FROM THE FOLLOWING TOPICS "*"
2020 PRINT "*"
2022 PRINT "*" 1. CALCIUM REGULATION "*"
2024 PRINT "*"
2026 PRINT "*" 2. HYPERCALCEMIA SYMPTOMS "*"
2028 PRINT "*"
2030 PRINT "*" 3. CAUSES FOR HYPERCALCEMIA "*"
2032 PRINT "*"
2034 PRINT "*" 4. TESTING FOR HYPERCALCENIA "*"
2036 PRINT "*"
2038 PRINT "*" 5. RANGE OF NORMAL CALCIUM LEVELS*"
2040 PRINT "*"
2042 PRINT "*" 6. CORRECTION FACTORS "*"
2044 PRINT "*"
2048 PRINT "*" 7. CASE HISTORY---MILD CASE "*"
2049 PRINT "*"
2050 PRINT "*" 8. CASE HISTORY---SEVERE CASE "*"
2051 PRINT "*"
2052 PRINT "*****"
2054 VTAB 22: CALL - 198
2056 PRINT "PRESS THE KEY CORRESPONDING TO YOUR CHOICE.": GET A$
2060 HOME
2070 IF A$ = "1" THEN GOSUB 2100: GOTO 2000
2072 IF A$ = "2" THEN GOSUB 3000: GOTO 2000
2074 IF A$ = "3" THEN GOSUB 4000: GOTO 2000
2076 IF A$ = "4" THEN GOSUB 5000: GOTO 2000
2078 IF A$ = "5" THEN GOSUB 6000: GOTO 2000
2080 IF A$ = "6" THEN GOSUB 7000: GOTO 2000
2082 IF A$ = "7" THEN GOSUB 8000: GOTO 2000
2084 IF A$ = "8" THEN GOSUB 9000: GOTO 2000
2085 IF A$ = "*" THEN GOTO 50
2086 CALL - 198: VTAB 22
2088 PRINT "PRESS THE KEY CORRESPONDING TO YOUR CHOICE.": GET A$: GOTO
2070
2100 SR = 0:SP = 85
2110 SR = 0:SP = 85
2112 GOSUB 10000
2200 REM FIRST QUESTION
2201 R = 0: HOME
2202 PRINT "*****"
2204 PRINT "*"
2206 PRINT "*" HYPERCALCEMIA IS BEING DIAGNOSED "*"

```

```

2208 PRINT "*"
2210 PRINT "*" MORE BECAUSE OF:
2212 PRINT "*"
2214 PRINT "*" 1. THE INCREASE IN STRESS IN OUR
2216 PRINT "*" DAILY LIVES
2218 PRINT "*"
2220 PRINT "*" 2. THE ABILITY OF CLINICAL
2222 PRINT "*" LABORATORIES TO ROUTINELY TEST
2224 PRINT "*" CALCIUM LEVELS
2226 PRINT "*"
2228 PRINT "*" 3. THE INCREASE IN SUGAR IN OUR
2230 PRINT "*" DIET
2232 PRINT "*"
2234 PRINT "*****"
2236 VTAB 20: CALL - 198
2237 POKE - 16368,0
2238 PRINT "PRESS THE KEY CORRESPONDING TO YOUR CHOICE.": GET A$
2240 IF A$ = "2" THEN GOTO 2250
2242 IF A$ = "*" THEN GOTO 50
2244 PRINT "INCORRECT"
2246 GOTO 2255
2250 PRINT "CORRECT"
2254 R = R + 1
2255 FOR I = 0 TO 1000: NEXT I
2300 REM SECOND QUESTION
2302 HOME
2303 PRINT "*****"
2304 PRINT "*"
2305 PRINT "*" WHAT THREE HORMONES ARE PRIMARILY
2306 PRINT "*"
2307 PRINT "*" RESPONSIBLE FOR CONTROLLING THE
2308 PRINT "*"
2310 PRINT "*" MOVEMENT OF CALCIUM BETWEEN THE
2311 PRINT "*"
2312 PRINT "*" EXTRACELLAR FLUID AND BONE,KIDNEY
2313 PRINT "*"
2314 PRINT "*" AND GASTROINTESTINAL TRACT?
2316 PRINT "*"
2317 PRINT "*****"
2330 VTAB 17
2331 POKE - 16368,0
2332 CALL - 198: INPUT "1.":A1$
2334 GOSUB 2900
2335 VTAB 19: POKE - 16368,0
2336 CALL - 198: INPUT "2.":A2$
2337 GOSUB 2920
2338 VTAB 21: POKE - 16368,0
2340 CALL - 198: INPUT "3.":A3$
2341 GOSUB 2930
2342 FOR I = 0 TO 1000: NEXT I
2346 IF R < 2 THEN GOTO 2382
2350 HOME
2351 PRINT "YOU HAVE SCORED WELL ENOUGH TO"
2352 CALL - 922
2353 PRINT "PROCEED ON TO NEW MATERIAL."
2354 FOR I = 0 TO 1000: NEXT I
2355 RETURN
2382 HOME
2384 PRINT "YOU HAVE NOT SCORED WELL ENOUGH"
2385 CALL - 922
2386 PRINT "TO CONTINUE ON TO NEW MATERIAL!"
2387 CALL - 922
2388 PRINT "PAY CLOSE ATTENTION TO THE LECTURE"
2389 CALL - 922
2390 PRINT "MATERIAL!"

```

LISTING 7.4 (Continued)

```

2392 FOR I = 0 TO 1000
2393 NEXT I
2394 GOTO 2100
2900 B1$ = "D":B2$ = "VITAMIN D":B3$ = "PTH":B4$ = "PARATHYROID":B5$ = "P
    ARATHYROID HORMONE":B6$ = "CALCITONIN"
2905 IF A1$ = "*" THEN GOTO 50
2910 IF A1$ = B1$ THEN GOTO 2960
2911 IF A1$ = B2$ THEN GOTO 2960
2912 IF A1$ = B3$ THEN GOTO 2960
2913 IF A1$ = B4$ THEN GOTO 2960
2914 IF A1$ = B5$ THEN GOTO 2960
2915 IF A1$ = B6$ THEN GOTO 2960
2916 GOTO 2950
2920 IF A2$ = B1$ THEN GOTO 2960
2921 IF A2$ = B2$ THEN GOTO 2960
2922 IF A2$ = B3$ THEN GOTO 2960
2923 IF A2$ = B4$ THEN GOTO 2960
2924 IF A2$ = B5$ THEN GOTO 2960
2925 IF A2$ = B6$ THEN GOTO 2960
2926 IF A2$ = "*" THEN GOTO 50
2927 GOTO 2950
2930 IF A3$ = B1$ THEN GOTO 2960
2931 IF A3$ = B2$ THEN GOTO 2960
2932 IF A3$ = B3$ THEN GOTO 2960
2933 IF A3$ = B4$ THEN GOTO 2960
2934 IF A3$ = B5$ THEN GOTO 2960
2935 IF A3$ = B6$ THEN GOTO 2960
2936 IF A3$ = "*" THEN GOTO 50
2950 CV = PEEK (37): VTAB CV
2952 HTAB 20: PRINT "INCORRECT"
2954 RETURN
2960 CV = PEEK (37): VTAB CV
2962 HTAB 20: PRINT "CORRECT"
2963 R = R + 1
2964 RETURN
2999 RETURN
3000 REM HYPERCALCEMIA SYMPTOMS
3010 SR = 92:SP = 114
3012 GOSUB 10000
3100 REM FIRST QUESTION
3102 R = 0: HOME
3104 VTAB 8
3105 PRINT "*****"
3106 PRINT "*"
3108 PRINT "* PATIENTS HAVING MILD"
3110 PRINT "*"
3112 PRINT "* HYPERCALCEMIA CAN BE DESCRIBED"
3114 PRINT "*"
3116 PRINT "* AS BEING":CV = PEEK (37)
3118 VTAB CV: HTAB 37: PRINT "*"
3120 PRINT "*"
3122 PRINT "*****"
3123 POKE - 16368,0
3124 VTAB CV: HTAB 12: INPUT A$
3130 VTAB CV: HTAB 37: PRINT "*"
3132 IF A$ = "ASYMPTOMATIC" THEN GOTO 3150
3134 IF A$ = "*" THEN GOTO 50
3140 VTAB 22: PRINT "INCORRECT"
3142 GOTO 3154
3150 VTAB 22: PRINT "CORRECT"
3152 R = R + 1
3154 FOR I = 0 TO 1000: NEXT I
3200 REM SECOND QUESTION
3202 HOME

```

```

3210 PRINT "*****"
3212 PRINT "*"
3214 PRINT "* LIST TWO SYMPTOMS OF HYPER-"
3216 PRINT "*"
3218 PRINT "* CALCEMIA WHICH ARE ASSOCIATED"
3219 PRINT "*"
3220 PRINT "* WITH DEHYDRATION AND IMPAIRED"
3222 PRINT "*"
3224 PRINT "* RENAL FUNCTION."
3226 PRINT "*"
3228 PRINT "*****"
3230 VTAB 17
3231 POKE - 16368,0
3232 CALL - 198: INPUT "1.":A1$
3233 GOSUB 3300
3234 POKE - 16368,0: CALL - 198: INPUT "2.":A2$
3235 GOSUB 3320
3236 FOR I = 0 TO 1000: NEXT I
3237 IF R < 2 THEN GOTO 3252
3240 HOME
3242 PRINT "YOU HAVE SCORED WELL ENOUGH TO"
3243 CALL - 922
3244 PRINT "CONTINUE ON TO NEW MATERIAL."
3245 FOR I = 0 TO 1000: NEXT I
3246 RETURN
3252 HOME
3254 PRINT "YOU HAVE NOT SCORED WELL ENOUGH"
3256 CALL - 922
3258 PRINT "TO CONTINUE ON TO NEW MATERIAL!"
3260 CALL - 922
3262 PRINT "PAY CLOSE ATTENTION TO THE LECTURE"
3264 CALL - 922
3266 PRINT "MATERIAL!"
3268 FOR I = 0 TO 1000: NEXT I
3270 GOTO 3000
3300 B1$ = "VOMITING":B2$ = "POLYURIA"
3310 IF A1$ = B1$ THEN GOTO 3360
3312 IF A1$ = B2$ THEN GOTO 3360
3313 IF A1$ = "*" THEN GOTO 50
3314 GOTO 3350
3320 IF A2$ = B1$ THEN GOTO 3360
3322 IF A2$ = B2$ THEN GOTO 3360
3324 IF A2$ = "*" THEN GOTO 50
3350 CV = PEEK (37): VTAB CV
3352 HTAB 20: PRINT "INCORRECT"
3354 RETURN
3360 CV = PEEK (37): VTAB CV
3362 HTAB 20: PRINT "CORRECT"
3364 R = R + 1
3366 RETURN
4000 REM CAUSES FOR HYPERCALCEMIA
4010 SR = 122:SP = 172
4012 GOSUB 10000
4100 REM FIRST QUESTION
4102 R = 0: HOME
4108 VTAB 6
4110 PRINT "*****"
4112 PRINT "*"
4114 PRINT "* THE MOST IMPORTANT PATHOGENETIC"
4116 PRINT "*"
4118 PRINT "* MECHANISM OF HYPERCALCEMIA IS:"
4120 PRINT "*"
4122 PRINT "*"
4124 PRINT "* 1. INCREASED BONE ABSORPTION"
4126 PRINT "*"
4128 PRINT "* 2. INCREASED BONE RESORPTION"

```

LISTING 7.4 (Continued)

```

4130 PRINT "*"
4132 PRINT "*" 3. DECREASED BONE ABSORPTION "*"
4134 PRINT "*"
4136 PRINT "*****"
4138 VTAB 22
4140 CALL - 198
4142 PRINT "PRESS THE KEY CORRESPONDING TO YOUR CHOICE.": GET A$
4143 POKE - 16368,0
4144 IF A$ = "2" THEN GOTO 4150
4145 IF A$ = "*" THEN GOTO 50
4146 PRINT "INCORRECT"
4148 GOTO 4155
4150 PRINT "CORRECT"
4152 R = R + 1
4155 FOR I = 0 TO 1000: NEXT I
4200 REM SECOND QUESTION
4202 HOME
4208 VTAB 6
4210 PRINT "*****"
4212 PRINT "*"
4214 PRINT "*" INCREASED BONE RESORPTION RESULTS "*"
4216 PRINT "*"
4218 PRINT "*" FROM THREE CAUSES, LIST TWO "*"
4220 PRINT "*"
4222 PRINT "*" OF THEM. "*"
4224 PRINT "*"
4226 PRINT "*****"
4228 VTAB 17
4229 POKE - 16368,0
4230 CALL - 198: INPUT "1.":A1$
4231 GOSUB 4280: VTAB 19
4232 POKE - 16368,0: CALL - 198: INPUT "2.":A2$
4233 GOSUB 4285
4234 FOR I = 0 TO 1000: NEXT I
4236 GOTO 4300
4280 B1$ = "HYPERTHYROIDISM":B2$ = "METASTATIC CANCER":B3$ = "BONE-RESORB
    ING HORMONE"
4281 IF A1$ = B1$ THEN GOTO 4295
4282 IF A1$ = B2$ THEN GOTO 4295
4283 IF A1$ = B3$ THEN GOTO 4295
4284 IF A1$ = "*" THEN GOTO 50
4285 GOTO 4290
4286 IF A2$ = B1$ THEN GOTO 4295
4287 IF A2$ = B2$ THEN GOTO 4295
4288 IF A2$ = B3$ THEN GOTO 4295
4289 IF A2$ = "*" THEN GOTO 50
4290 CV = PEEK (37): VTAB CV: HTAB 30
4291 PRINT "INCORRECT"
4292 RETURN
4295 CV = PEEK (37): VTAB CV: HTAB 30
4296 PRINT "CORRECT"
4297 R = R + 1
4298 RETURN
4300 REM THIRD QUESTION
4302 HOME
4308 VTAB 6
4310 PRINT "*****"
4312 PRINT "*"
4314 PRINT "*" INCREASED GASTROINTESTINAL "*"
4316 PRINT "*"
4318 PRINT "*" ABSORPTION OF CALCIUM IS DUE TO: "*"
4320 PRINT "*"
4322 PRINT "*"
4324 PRINT "*" 1. VITAMIN D INTOXICATION "*"

```

```

4326 PRINT "*"
4328 PRINT "*" 2. THYROTOXICOSIS "*"
4330 PRINT "*" "*"
4332 PRINT "*" 3. PAGET'S DISEASE "*"
4334 PRINT "*" "*"
4336 PRINT "*****"
4338 CALL - 198: VTAB 22
4339 POKE - 16368,0
4340 PRINT "PRESS THE KEY CORRESPONDING TO YOUR CHOICE.": GET A$
4341 IF A$ = "1" THEN GOTO 4348
4342 IF A$ = "*" THEN GOTO 50
4343 PRINT "INCORRECT": GOTO 4350
4348 PRINT "CORRECT"
4349 R = R + 1
4350 FOR I = 0 TO 1000: NEXT I
4351 IF R > 2 THEN GOTO 4400
4352 HOME
4354 PRINT "YOU HAVE NOT SCORED WELL ENOUGH"
4356 CALL - 922
4358 PRINT "TO CONTINUE ON TO NEW MATERIAL."
4360 CALL - 922
4362 PRINT "PAY CLOSE ATTENTION TO THE LECTURE"
4364 CALL - 922
4366 PRINT "MATERIAL."
4368 FOR I = 0 TO 1000: NEXT I
4370 GOTO 4000
4400 HOME
4402 PRINT "YOU HAVE SCORED WELL ENOUGH TO"
4403 CALL - 922
4404 PRINT "CONTINUE ON TO NEW MATERIAL."
4406 FOR I = 0 TO 1000: NEXT I
4408 RETURN
5000 REM TESTING FOR HYPERCALCEMIA
5010 SR = 179:SP = 236
5012 GOSUB 10000
5100 REM FIRST QUESTION
5102 R = 0: HOME
5108 VTAB 6
5110 PRINT "*****"
5112 PRINT "*"
5114 PRINT "*" WHAT ARE THREE METHODS CURRENTLY "*"
5116 PRINT "*" "*"
5118 PRINT "*" USED TO MEASURE THE TOTAL SERUM "*"
5120 PRINT "*" "*"
5122 PRINT "*" CALCIUM CONCENTRATION? "*"
5124 PRINT "*" "*"
5126 PRINT "*****"
5128 CALL - 922
5130 POKE - 16368,0
5132 CALL - 198: INPUT "1.":A1$
5134 GOSUB 5900
5136 POKE - 16368,0
5138 CALL - 198: INPUT "2.":A2$
5140 GOSUB 5920
5142 POKE - 16368,0
5144 CALL - 198: INPUT "3.":A3$
5146 GOSUB 5930
5148 FOR I = 0 TO 1000: NEXT I
5150 IF R < 2 THEN GOTO 5191
5152 HOME
5154 PRINT "YOU HAVE SCORED WELL ENOUGH TO"
5156 CALL - 922
5158 PRINT "CONTINUE ON TO NEW MATERIAL."
5160 FOR I = 0 TO 1000: NEXT I
5162 RETURN
5191 HOME

```

LISTING 7.4 (Continued)

```

5192 PRINT "YOU HAVE NOT SCORED WELL ENOUGH": CALL - 922
5193 PRINT "TO CONTINUE ON TO NEW MATERIAL.": CALL - 922
5194 PRINT "PAY CLOSE ATTENTION TO THE LECTURE": CALL - 922
5196 PRINT "MATERIAL."
5197 FOR I = 0 TO 1000: NEXT I
5198 GOTO 5000
5900 B1$ = "ATOMIC ABSORPTION":B2$ = "ATOMIC ABSORPTION SPECTROPHOTOMETER"
      B3$ = "SPECTROPHOTOMETER"
5902 B4$ = "COLORIMETER":B5$ = "COLORIMETRIC METHODS":B6$ = "AUTOMATIC TI
      TRATION":B7$ = "TITRATION"
5906 IF A1$ = "*" THEN GOTO 50
5908 IF A1$ = B1$ THEN GOTO 5960
5910 IF A1$ = B2$ THEN GOTO 5960
5912 IF A1$ = B3$ THEN GOTO 5960
5914 IF A1$ = B4$ THEN GOTO 5960
5916 IF A1$ = B5$ THEN GOTO 5960
5917 IF A1$ = B6$ THEN GOTO 5960
5918 IF A1$ = B7$ THEN GOTO 5960
5919 GOTO 5950
5920 IF A2$ = B1$ THEN GOTO 5960
5922 IF A2$ = B2$ THEN GOTO 5960
5923 IF A2$ = B3$ THEN GOTO 5960
5924 IF A2$ = B4$ THEN GOTO 5960
5925 IF A2$ = B5$ THEN GOTO 5960
5926 IF A2$ = B6$ THEN GOTO 5960
5927 IF A2$ = B7$ THEN GOTO 5960
5928 IF A2$ = "*" THEN GOTO 50
5929 GOTO 5950
5930 IF A3$ = B1$ THEN GOTO 5960
5932 IF A3$ = B2$ THEN GOTO 5960
5933 IF A3$ = B6$ THEN GOTO 5960
5934 IF A3$ = B3$ THEN GOTO 5960
5935 IF A3$ = B7$ THEN GOTO 5960
5936 IF A3$ = B4$ THEN GOTO 5960
5937 IF A3$ = B5$ THEN GOTO 5960
5950 CV = PEEK (37): VTAB CV: HTAB 30
5952 PRINT "INCORRECT"
5955 RETURN
5960 CV = PEEK (37): VTAB CV: HTAB 30
5962 PRINT "CORRECT"
5964 R = R + 1
5966 RETURN
6000 REM RANGE OF NORMAL CALCIUM LEVELS
6010 SR = 241:SP = 308
6012 GOSUB 10000
6100 REM FIRST QUESTION
6102 R = 0: HOME
6108 VTAB 6
6110 PRINT "*****"
6112 PRINT "*"
6114 PRINT "    THE NORMAL SERUM CALCIUM    *"
6116 PRINT "    *"
6118 PRINT "    CONCENTRATION RANGES FROM:  *"
6120 PRINT "    *"
6122 PRINT "    *"
6124 PRINT "    1.  88-104 MG/100ML          *"
6126 PRINT "    *"
6128 PRINT "    2.  8.8-10.4 MG/100ML       *"
6130 PRINT "    *"
6132 PRINT "    3.  .88-1.04 MG/100ML      *"
6134 PRINT "    *"
6136 PRINT "*****"
6138 VTAB 22: CALL - 198
6139 POKE - 16368,0

```

```

6140 PRINT "PRESS THE KEY CORRESPONDING TO YOUR      CHOICE.": GET A$
6141 IF A$ = "2" THEN GOTO 6145
6142 IF A$ = "*" THEN GOTO 50
6143 PRINT "INCORRECT"
6144 GOTO 6147
6145 PRINT "CORRECT"
6146 R = R + 1
6147 FOR I = 0 TO 1000: NEXT I
6148 IF R = 1 THEN GOTO 6170
6150 CALL - 922
6151 HOME : PRINT "YOU HAVE NOT SCORED WELL ENOUGH": CALL - 922
6152 PRINT "TO CONTINUE ONTO NEW MATERIAL.": CALL - 922
6154 PRINT "PAY CLOSE ATTENTION TO THE LECTURE": CALL - 922
6155 PRINT "MATERIAL."
6158 FOR I = 0 TO 1000: NEXT I
6160 GOTO 6000
6170 HOME
6172 PRINT "YOU HAVE SCORED WELL ENOUGH TO": CALL - 922
6174 PRINT "CONTINUE ON TO NEW MATERIAL."
6175 FOR I = 0 TO 1000: NEXT I
6176 RETURN
7000 REM CORRECTION FACTORS
7010 SR = 314:SP = 440
7012 GOSUB 10000
7100 REM FIRST QUESTION
7102 R = 0: HOME
7110 PRINT "*****"
7112 PRINT "*"
7114 PRINT "* GIVEN THE FOLLOWING RESULTS FORM *"
7116 PRINT "*"
7118 PRINT "* THE LABORATORY ANALYSIS, WHAT *"
7120 PRINT "*"
7122 PRINT "* WOULD THE CORRECTED CALCIUM *"
7124 PRINT "*"
7126 PRINT "* CONCENTRATION BE EQUAL TO? *"
7127 PRINT "*"
7128 PRINT "*      CALCIUM----10.2 MG/100ML *"
7129 PRINT "*      ALBUMIN-----3.5 MG/100ML *"
7130 PRINT "*      NORMAL ALBUMIN--4.5 MG/100ML *"
7134 PRINT "*"
7136 PRINT "* 1.  11 MG/100ML *"
7138 PRINT "*"
7140 PRINT "* 2.  9.6 MG/100ML *"
7142 PRINT "*"
7144 PRINT "* 3. 13.8 MG/100ML *"
7146 PRINT "*"
7148 PRINT "*****"
7150 VTAB 22: CALL - 198
7151 POKE - 16368,0
7152 PRINT "PRESS THE KEY CORRESPONDING TO YOUR      CHOICE.": GET A$
7153 IF A$ = "*" THEN GOTO 50
7154 IF A$ = "1" THEN GOTO 7160
7156 PRINT "INCORRECT"
7158 GOTO 7165
7160 PRINT "CORRECT"
7162 R = R + 1
7165 FOR I = 0 TO 1000: NEXT I
7200 REM SECOND QUESTION
7202 HOME
7210 PRINT "*****"
7212 PRINT "*"
7213 PRINT "* GIVEN THE FOLLOWING RESULTS FROM *"
7214 PRINT "*"
7218 PRINT "* THE LABORATORY ANALYSIS, WHAT *"
7220 PRINT "*"

```

LISTING 7.4 (Continued)

```

7223 PRINT '* WOULD THE CORRECTED CALCIUM      *'
7224 PRINT '*                                  *'
7225 PRINT '* CONCENTRATION BE EQUAL TO?      *'
7226 PRINT '*                                  *'
7230 PRINT '*          CALCIUM----10.2 MG/100ML *'
7232 PRINT '*          ALBUMIN-----5.5 MG/100ML *'
7234 PRINT '*          NORMAL ALBUMIN-4.5 MG/100ML*'
7236 PRINT '*                                  *'
7238 PRINT '*                                  *'
7240 PRINT '* 1.  11 MG/100ML                    *'
7241 PRINT '*                                  *'
7242 PRINT '* 2.   9.6 MG/100ML                  *'
7244 PRINT '*                                  *'
7245 PRINT '* 3.  13.8 MG/100ML                  *'
7250 PRINT '*                                  *'
7252 PRINT '*****'
7254 VTAB 22: CALL - 198
7255 POKE - 16368,0
7256 PRINT 'PRESS THE KEY CORRESPONDING TO YOUR    CHOICE,': GET A$
7257 IF A$ = "*" THEN GOTO 50
7258 IF A$ = "2" THEN GOTO 7290
7259 PRINT 'INCORRECT': GOTO 7295
7262 HOME
7264 PRINT 'YOU HAVE NOT SCORED WELL ENOUGH': CALL - 922
7266 PRINT 'TO CONTINUE ON TO NEW MATERIAL.': CALL - 922
7270 PRINT 'PAY CLOSE ATTENTION TO THE LECTURE': CALL - 922
7272 PRINT 'MATERIAL.'
7274 FOR I = 0 TO 1000: NEXT I
7276 GOTO 7000
7280 HOME
7281 PRINT 'YOU HAVE SCORED WELL ENOUGH TO': CALL - 922
7282 PRINT 'CONTINUE ON TO NEW MATERIAL.'
7284 FOR I = 0 TO 1000: NEXT I
7286 RETURN
7290 PRINT 'CORRECT'
7292 R = R + 1
7295 FOR I = 0 TO 1000: NEXT I
7296 IF R = 2 THEN GOTO 7280
7297 GOTO 7262
8000 REM CASE HISTORY(MILD)
8010 SR = 446:SP = 822
8012 GOSUB 10000
8100 REM FIRST QUESTION
8102 RETURN
9000 REM CASE HISTORY(SEVERE)
9010 SR = 831:SP = 1100
9012 GOSUB 10000
9100 REM FIRST QUESTION
9102 RETURN
10000 REM CONTROLLER ROUTINE
10002 R1 = 0:R2 = 0:P1 = 0:P2 = 0
10010 X = PEEK (771) + 256 * PEEK (772)
10020 R1 = SR
10021 R1 = R1 - 256
10022 IF R1 = 0 THEN GOTO 10030
10023 IF R1 > 0 THEN GOTO 10032
10024 IF R1 < 0 THEN GOTO 10035
10030 R2 = R2 + 1: GOTO 10090
10032 R2 = R2 + 1: GOTO 10021
10035 R1 = R1 + 256
10050 P1 = SP
10051 P1 = P1 - 256
10052 IF P1 = 0 THEN GOTO 10080
10053 IF P1 > 0 THEN GOTO 10082

```

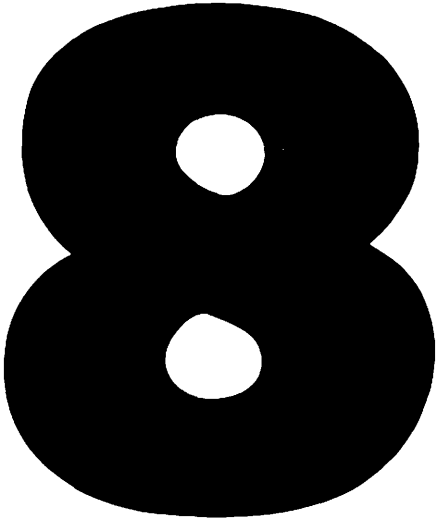
```

10054 IF P1 < 0 THEN GOTO 10085
10080 P2 = P2 + 1: GOTO 10090
10082 P2 = P2 + 1: GOTO 10051
10085 P1 = P1 + 256
10090 IF X < SR THEN GOTO 10100
10092 IF X > SR THEN GOTO 10200
10094 IF X = SR THEN GOTO 10300
10100 REM FAST FORWARD
10110 POKE 773,R1: POKE 774,R2
10112 POKE - 16142,2: CALL 784
10114 POKE - 16142,0
10116 GOTO 10300
10200 REM REWIND
10202 R3 = 0: R4 = 0
10210 R3 = SR + 2
10211 R3 = R3 - 256
10212 IF R3 = 0 THEN GOTO 10220
10213 IF R3 > 0 THEN GOTO 10230
10214 IF R3 < 0 THEN GOTO 10240
10220 R4 = R4 + 1: GOTO 10250
10230 R4 = R4 + 1: GOTO 10211
10240 R3 = R3 + 256
10250 POKE 773,R3: POKE 774,R4
10252 POKE - 16142,3: CALL 864
10254 POKE - 16142,0
10256 POKE 771,R1: POKE 772,R2
10300 REM PLAY
10310 POKE 773,P1: POKE 774,P2
10320 POKE - 16142,1: CALL 784
10400 REM STOP
10410 POKE - 16142,0
10412 POKE - 16142,14
10420 RETURN
11000 REM GRAPHIC INTRODUCTION
11012 HOME : GR : COLOR= 2
11020 VLIN 0,11 AT 0: VLIN 0,11 AT 12: PLOT 1,0: PLOT 2,2: PLOT 3,4: PLOT
4,6: PLOT 5,8: PLOT 6,10: PLOT 7,8: PLOT 8,6: PLOT 9,4: PLOT 10,2: PLOT
11,0: PLOT 6,11: REM M
11025 VLIN 0,11 AT 0: VLIN 0,5 AT 15: HLIN 15,25 AT 5: VLIN 6,11 AT 25: HLIN
25,15 AT 11: HLIN 15,25 AT 0: REM S
11030 VLIN 0,11 AT 29: VLIN 0,11 AT 39: HLIN 29,39 AT 11: REM U
11035 VLIN 28,39 AT 0: HLIN 1,10 AT 28: HLIN 0,10 AT 39: REM C
11040 HLIN 13,23 AT 28: HLIN 13,23 AT 39: VLIN 28,39 AT 13: VLIN 28,39 AT
23: REM O
11045 VLIN 28,39 AT 27: VLIN 28,39 AT 39: PLOT 28,28: PLOT 29,30: PLOT 3
0,32: PLOT 31,34: PLOT 32,36: PLOT 33,38: PLOT 33,39: PLOT 34,36: PLOT
35,34: PLOT 36,32: PLOT 37,30: PLOT 38,28: REM M
11090 GET A$
11099 RETURN
12000 REM MACHINE LANGUAGE ROUTINE
12010 POKE 768,234: POKE 769,234: POKE 770,234: POKE 771,234: POKE 772,2
34: POKE 773,234: POKE 774,234: POKE 775,234: POKE 776,234: POKE 777
,234: POKE 778,234: POKE 779,234
12012 POKE 780,234: POKE 781,234: POKE 782,234: POKE 783,234
12020 POKE 784,008: POKE 785,072: POKE 786,173: POKE 787,243: POKE 788,1
92: POKE 789,042: POKE 790,176: POKE 791,250
12025 POKE 792,173: POKE 793,243: POKE 794,192: POKE 795,042: POKE 796,1
44: POKE 797,250: POKE 798,238: POKE 799,003
12030 POKE 800,003: POKE 801,208: POKE 802,003: POKE 803,238: POKE 804,0
04: POKE 805,003: POKE 806,173: POKE 807,003
12035 POKE 808,003: POKE 809,205: POKE 810,005: POKE 811,003: POKE 812,2
08: POKE 813,228: POKE 814,173: POKE 815,004
12040 POKE 816,003: POKE 817,205: POKE 818,006: POKE 819,003: POKE 820,2
08: POKE 821,220: POKE 822,104: POKE 823,040: POKE 824,096
12045 POKE 864,008: POKE 865,072: POKE 866,173: POKE 867,243: POKE 868,1
92: POKE 869,042: POKE 870,176: POKE 871,250

```

LISTING 7.4 (Continued)

```
12050 POKE 872,173: POKE 873,243: POKE 874,192: POKE 875,042: POKE 876,1
      44: POKE 877,250: POKE 878,056: POKE 879,173
12055 POKE 880,003: POKE 881,003: POKE 882,233: POKE 883,001: POKE 884,1
      41: POKE 885,003: POKE 886,003: POKE 887,173
12060 POKE 888,004: POKE 889,003: POKE 890,233: POKE 891,000: POKE 892,1
      41: POKE 893,004: POKE 894,003: POKE 895,173
12065 POKE 896,003: POKE 897,003: POKE 898,205: POKE 899,005: POKE 900,0
      03: POKE 901,208: POKE 902,219: POKE 903,173
12070 POKE 904,004: POKE 905,003: POKE 906,205: POKE 907,006: POKE 908,0
      03: POKE 909,208: POKE 910,211: POKE 911,104
12075 POKE 912,040: POKE 913,096
12999 RETURN
```



DATA ANALYSIS TECHNIQUES

CURVE FITTING

The observation of naturally-occurring phenomena is often the basic approach of a scientific researcher. Data are recorded as a function of some dependent variable (a parameter that is not a function of the phenomena that is being observed) and the researcher then draws conclusions based on the relationship between the two variables. The first step that usually occurs after the data have been gathered is the posing of question: What mathematical formula best describes the relationship between the two variables?

Linear Regression

The simplest relationship that two variables could have is a linear relationship, that is, the dependent variable and the independent variable would increase or decrease with a one-to-one relationship. When a linear relationship occurs, the dependent variable can be expressed as a function of the independent variable by the following equation:

$$y = A x + B \quad (8.1)$$

where A and B are constants

The following program will find values for A and B which will give the closest agreement between the experimental data and Equation 8.1. The technique that will be used is called *linear regression by the method of least squares*.

Listing 8.1 shows the program with comments. The program expects to receive x and y pairs of data. When all the values have been entered, the program computes the coefficient for Equation 8.1 and calculates a coefficient of determination to give a relative indication of the goodness of fit. Once the equation has been generated, you may predict values of y for given values of x.

EXAMPLE A forgetful electrical engineering professor wakes up one morning and thinks that he might have a fever. A search through his medicine cabinet produces an oral thermometer, but it slips through his fingers and falls to the floor, breaking into a thousand pieces. Disgusted with himself, he suddenly realizes that he has an outdoor thermometer which is calibrated in degrees Fahrenheit, but it is much too big to fit in his mouth. He remembers that the skin-temperature biofeedback project he built has a thermistor probe, but that it is not calibrated. He suspects that the probe is linear and that the relationship between the output voltage from the interface and temperature might be equal to Equation 8.2. He knows that if he could get a few data points he could run the linear regression program and solve for the coefficients of the equation:

$$T = aV + b \quad (8.2)$$

LISTING 8.1 BASIC linear regression routine.

```

100 REM LINEAR REGRESSION
102 REM R. HALLGREN 3-5-81
110 HOME
130 PRINT "*****"
132 PRINT "*"
134 PRINT "* YOU WILL INPUT THE PAIRED *"
138 PRINT "* VALUES OF DATA THAT YOU *"
142 PRINT "* HAVE GATHERED, *"
144 PRINT "*"
146 PRINT "* WHEN ALL DATA PAIRS HAVE *"
150 PRINT "* BEEN ENTERED, PRESS * TO *"
154 PRINT "* CALCULATE THE REGRESSION *"
158 PRINT "* COEFFECIENTS. *"
159 PRINT "*"
160 PRINT "* PRESS THE SPACE BAR *"
162 PRINT "* TO CONTINUE. *"
164 PRINT "*****"
180 GET K$
182 IF K$ < > " " GOTO 180
200 REM INPUT DATA PAIRS
210 J = 0:K = 0:M = 0:R2 = 0
212 N = 1
230 HOME
240 PRINT "DATA PAIR #";N
242 INPUT "X VALUE ";X(N)
244 INPUT "Y VALUE ";Y(N)
246 N = N + 1
250 PRINT " "; PRINT " "; PRINT " "
252 PRINT "PRESS * TO CALCULATE"
254 PRINT "COEFFICIENTS."
255 PRINT
256 PRINT "PRESS THE SPACE BAR TO ENTER"
258 PRINT "MORE DATA PAIRS."
260 GET K$
262 IF K$ = "*" GOTO 300
264 IF K$ < > " " GOTO 260
268 GOTO 230
300 REM CALCULATE REGRESSION COEFFICIENTS
302 N = N - 1
310 FOR I = 0 TO N
312 J = J + X(I)
314 K = K + Y(I)
316 L = L + (X(I)) ^ 2
318 M = M + (Y(I)) ^ 2
320 R2 = R2 + X(I) * Y(I)
330 NEXT I
340 A = (N * R2 - K * J) / (N * L - J ^ 2)
342 B = (K - A * J) / N
350 HOME
354 PRINT "Y=";A;"* X +";B
400 REM COMPUTE COEFFICIENT OF DETERMINATION
410 J = A * (R2 = J * K / N)
412 M = M - K ^ 2 / N
414 K = M - J
416 R2 = J / M
420 PRINT
424 PRINT "COEFFICIENT OF DETERMINATION = ";R2
430 PRINT
432 PRINT "DO YOU NEED TO PREDICT VALUES"
434 PRINT "OF Y FOR GIVEN VALUES OF X?"
440 PRINT
442 PRINT "PRESS Y OR N"
444 GET K$
446 IF K$ = "N" THEN END
448 IF K$ < > "Y" THEN 444
450 REM CALCULATE Y FROM VALUES OF X
452 HOME
456 INPUT "X=";X
458 PRINT "Y=";A * X + B
460 GOTO 430

```

He places both the thermometer and the thermistor probe into a sink of warm water and records the following data points:

V(volts)	1.055	1.044	1.014	.984	.974	.943
T(F)	104	103	100	97	96	93

Find the values for a and b ? What is the coefficient of determination? If the professor's temperature is equal to 1.006 volts, should he be worried?

Exponential Regression

Another common relationship between data pairs is described by an exponential curve. When such a relationship exists, the dependent variable can be expressed as a function of the independent variable by an equation having the following form:

$$y = A e^{Bx} \quad (8.3)$$

The following program will find values for A and B which will give the closest agreement between the experimental data and Equation 8.2. The technique that will be used is called *exponential regression by the method of least squares*.

Listing 8.2 shows the program with comments. The program expects to receive x and y pairs of data. When all the values have been entered, the program computes the coefficients for Equation 8.2 and calculates a coefficient of determination to give a relative indication of the goodness of fit. Once the equation has been determined, you may predict values of y for given values of x .

EXAMPLE Realizing a bargain when he saw one, Fred bought several large capacitors at the local flea market for a very low price. Unfortunately, these capacitors were unmarked. Fred knows that a capacitor, being charged through a series resistor from a fixed voltage source, will exponentially increase in voltage as time increases. By knowing the value of the series resistor, and measuring the voltage across the resistor as a function of time, he should be able to use the exponential regression program to determine the value of the capacitance. Using his high school electronics text, Fred determines that the voltage across the series resistor will be equal to the following equation:

LISTING 8.2 BASIC exponential routine.

```

100 REM EXPONENTIAL REGRESSION
102 REM R. HALLGREN 3-5-81
110 HOME
130 PRINT "*****"
132 PRINT "*"
134 PRINT "* YOU WILL INPUT THE PAIRED *"
138 PRINT "* VALUES OF DATA THAT YOU *"
142 PRINT "* HAVE GATHERED. *"
144 PRINT "*"
146 PRINT "* WHEN ALL DATA PAIRS HAVE *"
150 PRINT "* BEEN ENTERED, PRESS * TO *"
154 PRINT "* CALCULATE THE REGRESSION *"
158 PRINT "* COEFFECIENTS. *"
159 PRINT "*"
160 PRINT "* PRESS THE SPACE BAR *"
162 PRINT "* TO CONTINUE. *"
164 PRINT "*****"
180 GET K$
182 IF K$ < > " " GOTO 180
200 REM INPUT DATA PAIRS
210 J = 0:K = 0:M = 0:R2 = 0
212 N = 1
230 HOME
240 PRINT "DATA PAIR #";N
242 INPUT "X VALUE ";X(N)
244 INPUT "Y VALUE ";Y(N)
246 N = N + 1
250 PRINT
252 PRINT "PRESS * TO CALCULATE"
254 PRINT "COEFFICIENTS."
255 PRINT
256 PRINT "PRESS THE SPACE BAR TO ENTER"
258 PRINT "MORE DATA PAIRS."
260 GET K$
262 IF K$ = "*" GOTO 300
264 IF K$ < > " " GOTO 260
268 GOTO 230
300 REM CALCULATE REGRESSION COEFFICIENTS
302 N = N - 1
310 FOR I = 1 TO N
311 Y(I) = LOG (Y(I))
312 J = J + X(I)
314 K = K + Y(I)
316 L = L + (X(I)) ^ 2
318 M = M + (Y(I)) ^ 2
320 R2 = R2 + X(I) * Y(I)
330 NEXT I
340 B = (N * R2 - K * J) / (N * L - J ^ 2)
342 A = (K - B * J) / N
350 HOME
354 PRINT "B=";B
356 PRINT "A=";EXP (A)
400 REM COMPUTE COEFFICIENT OF DETERMINATION
410 J = B * (R2 - J * K / N)
412 M = M - K ^ 2 / N
414 K = M - J
416 R2 = J / M
420 PRINT
424 PRINT "COEFFICIENT OF DETERMINATION = ";R2
430 PRINT
432 PRINT "DO YOU NEED TO PREDICT VALUES"
434 PRINT "OF Y FOR GIVEN VALUES OF X?"
440 PRINT
442 PRINT "PRESS Y OR N"
444 GET K$

```

LISTING 8.2 (Continued)

```

446 IF K$ = "N" THEN END
448 IF K$ < > "Y" THEN 444
450 REM CALCULATE Y FROM VALUES OF X
452 HOME
456 INPUT "T=";X
458 PRINT "V="; EXP (A) * EXP (B * X)
460 GOTO 430

```

$$V = Ee^{-t/RC}$$

where V = resistor voltage (volts)
 E = battery voltage (volts) = 5
 R = resistance (ohms) = 1000
 C = capacitance (farads)
 t = time (seconds)

(8.4)

Getting out the high speed A/D converter that he built in Chapter 3, Fred gets ready to prove to himself that the capacitors really were a bargain. Fred sets the sampling rate at one sample per second and obtains the following set of data points:

t	0	1	2	3	4	5
V	5	1.9	.7	.25	.109	.042

Find the values for a and b in Equation 8.4 and calculate the capacitance of Fred's capacitor. If he needs ten farads for his plasma discharge generator, will he be able to use this particular capacitor?

Fast Fourier Analysis

The sounds that your ears receive are comprised of many individual frequencies which are summed together to form a complex waveform which gives the sound its characteristic quality. It is possible to take an arbitrary continuous signal, expressed in the time domain, and break it up into a series of individual frequencies. The magnitude of this spectrum at different frequencies is a measure of the frequency content that comprises the original signal. The concept of spectrum finds many practical applications in many varied disciplines including waveform analysis of music, frequency content of muscle voltages, vibrational characteristics of mechanical systems, and so on. Figure 8.1 shows a plot, in the

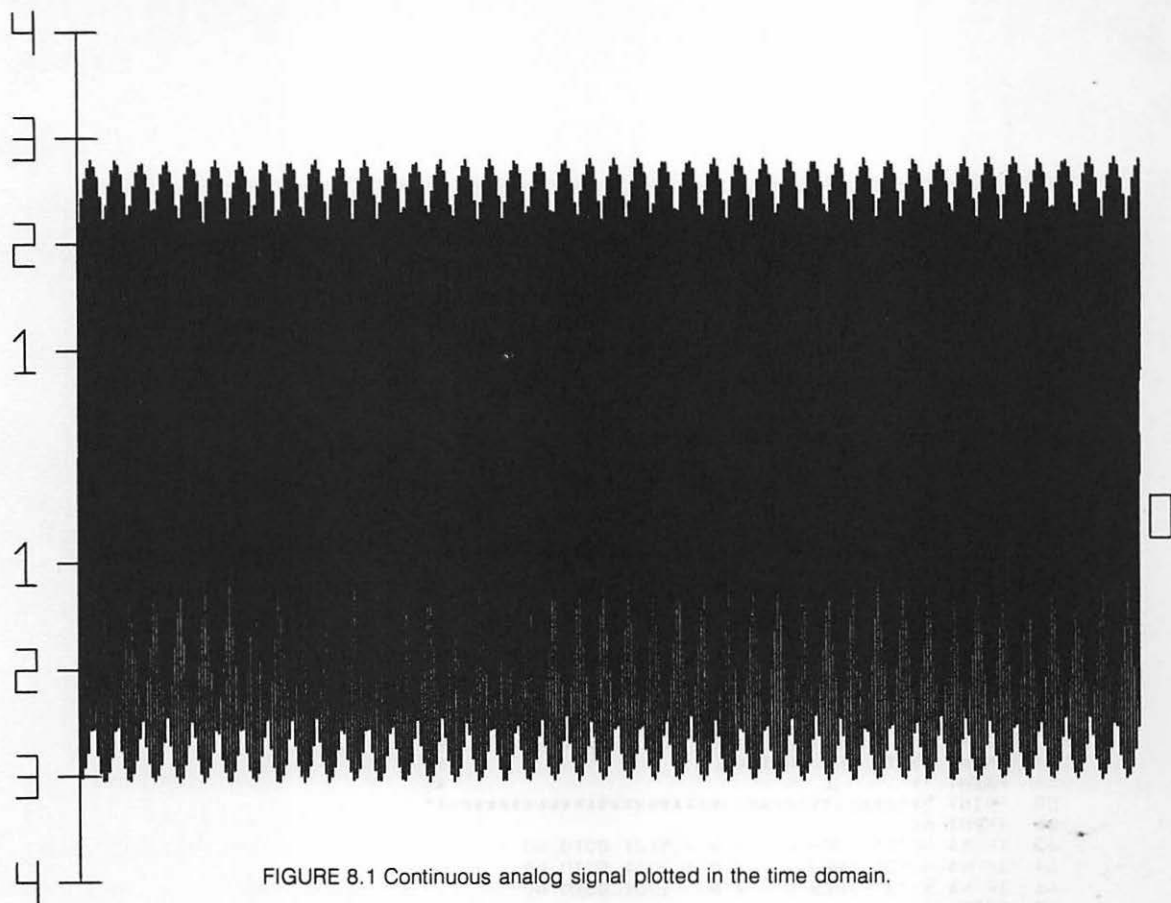


FIGURE 8.1 Continuous analog signal plotted in the time domain.

time domain, of a complex analog waveform. It would be very difficult to come to any conclusions regarding the spectral content of the signal by just visually analyzing it. Figure 8.2 shows the same waveform transformed into the frequency domain by a fast Fourier transform (FFT). You can easily observe a small d.c. component at the far left of the plot and make the statement that the complex waveform contains a single high-frequency signal. In fact, Figure 8.1 is just the plot of a 200 Hz sine wave.

Listing 8.3 shows a program written in Applesoft which performs the fast Fourier routine. The program first gets the data file that we created with the high speed A/D converter that we described in Chapter 3. The FFT routine then mathematically separates the complex waveform into the sum of a number of

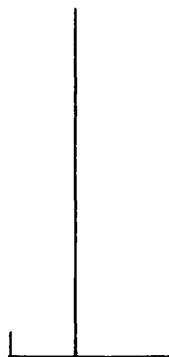


FIGURE 8.2 Signal from Figure 8.1 analyzed by the fast Fourier transform and plotted as a function of frequency.

LISTING 8.3 Fast Fourier routine written in BASIC.

```

10 REM FFT ROUTINE
12 REM R. HALLGREN, 11-2-81
14 REM APPLE II
20 HOME : VTAB 5
34 PRINT "*****"
36 PRINT "*"
38 PRINT "*" HOW MANY POINTS DO YOU WANT "*"
40 PRINT "*" TO PROCESS? "*"
42 PRINT "*"
44 PRINT "*" 1. 512 "*"
46 PRINT "*" 2. 256 "*"
48 PRINT "*" 3. 128 "*"
50 PRINT "*"
52 PRINT "*" PRESS THE NUMBER CORRESPONDING "*"
54 PRINT "*" TO THE CHOICE. "*"
56 PRINT "*"
58 PRINT "*****"
60 INPUT K$
62 IF K$ = "1" THEN L = 9:N = 512: GOTO 80
64 IF K$ = "2" THEN L = 8:N = 256: GOTO 80
66 IF K$ = "3" THEN L = 7:N = 128: GOTO 80
68 GOTO 60
80 GOTO 200
200 REM DATA FORMATTER SUBROUTINE
201 T = 0
202 DIM X1(N): DIM X2(N): P = 3.14159: X = 24830
203 D$ = "": PRINT D$; "LOAD DATA($1100),A$6100"
204 HOME : VTAB 10
206 PRINT "DATA IS BEING FORMATTED"
220 FOR Z = 0 TO N - 1
222 X = X + 2
224 V1 = PEEK (X): V2 = PEEK (X + 1)
230 IF V1 > 1 THEN GOTO 300
250 X1(Z) = - (511 - (V1 * 256 + V2)) / 100
270 NEXT Z
280 GOTO 640
300 X1(Z) = ((V1 - 2) * 256 + V2) / 100
320 GOTO 270
640 REM SCALE INPUT TIME FUNCTION
642 FOR Z = 0 TO N - 1
650 X1(Z) = X1(Z) / N
660 NEXT Z
670 REM FFT IN-PLACE ALGORITHM
675 HOME : VTAB 10

```

```

676 PRINT "      FFT CALCULATION IN PROCESS"
680 I1 = N / 2:I2 = 1:V = 2 * F / N
690 FOR I = 1 TO L
700 I3 = 0:I4 = I1
710 FOR K = 1 TO I2
720 X = INT (I3 / I1)
730 GOSUB 1300
740 I5 = Y
750 Z1 = COS (V * I5)
760 Z2 = - SIN (V * I5)
770 FOR M = I3 TO I4 - 1
780 A1 = X1(M):A2 = X2(M)
790 B1 = Z1 * X1(M + I1) - Z2 * X2(M + I1)
800 B2 = Z2 * X1(M + I1) + Z1 * X2(M + I1)
810 X1(M) = A1 + B1:X2(M) = A2 + B2
820 X1(M + I1) = A1 - B1:X2(M + I1) = A2 - B2
830 NEXT M
840 I3 = I3 + 2 * I1:I4 = I4 + 2 * I1
850 NEXT K
860 I1 = I1 / 2:I2 = 2 * I2
870 NEXT I
880 REM OUTPUT RESULTS
960 HOME : VTAB 10: PRINT "PLOT ON DIGITAL?": INPUT K$
962 IF K$ = "N" THEN GOTO 970
963 IF K$ < > "Y" THEN GOTO 962
964 FOR I = 0 TO 160: READ K: POKE ( - 32800 + I),K: NEXT I
970 B = 0
975 HOME : VTAB 10: PRINT "      CALCULATION IN PROGRESS"
980 FOR Z = 0 TO N / 2
985 X = Z
990 GOSUB 1390
1000 IF X3 > B THEN B = X3
1005 C = 0
1010 NEXT Z
1012 IF K$ = "N" THEN GOTO 1020
1015 HOME : VTAB 10: PRINT "LOAD PEN AND PRESS ANY KEY": GET A$
1020 HGR2 : HCOLOR= 3
1021 FOR Z = 0 TO N / 2
1022 IF Z > 900 THEN GOTO 1110
1025 X = Z
1030 GOSUB 1390
1040 X4 = INT (56 * X3 / B)
1050 K = INT (10 * X4)
1051 IF K > 600 THEN K = 600
1052 IF K$ = "N" THEN GOTO 1093
1055 CALL - 32646: FOR I = 0 TO 10: NEXT I
1056 FOR J = 1 TO K: CALL - 32712: NEXT J
1057 CALL - 32653
1058 FOR J = 1 TO K: CALL - 32681: NEXT J
1092 CALL - 32695
1093 GOTO 1500
1100 NEXT Z
1105 FOR I = 0 TO N:X2(I) = 0: NEXT I
1110 END
1290 REM SCRAMBLER SUBROUTINE
1300 Y = 0:N1 = N
1310 FOR W = 1 TO L
1320 N1 = N1 / 2
1330 IF X < N1 THEN GOTO 1360
1340 Y = Y + 2 ^ (W - 1)
1350 X = X - N1
1360 NEXT W
1370 RETURN
1380 REM MAGNITUDE(X3) SUBROUTINE
1390 GOSUB 1300
1400 X3 = SQR (X1(Y) ^ 2 + X2(Y) ^ 2)

```

LISTING 8.3 (Continued)

```

1410 RETURN
1500 REM VIDEO PLOT
1502 FOR R = 0 TO (K / 4)
1504 HPLLOT T,(155 - R)
1506 NEXT R
1508 T = T + 1
1599 GOTO 1100
10000 DATA 160,9,24,72,176,7,169,0,141,209,192,144,5,169,1,141,209,192,169,3
10002 DATA 72,169,4,74,144,253,104,233,1,208,245,104,106,136,208,223,160,2,169
10004 DATA 1,141,209,192,169,3,72,169,4,74,144,253,104,233,1,208,245,136,208
10006 DATA 235,96,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
10008 DATA 72,8,169,112,32,224,127,40,104,96,72,8,169,113,76,60,128,72,8,169,114
10010 DATA 76,60,128,72,8,169,115,76,60,128,72,8,169,116,76,60,128,72,8,169,117
10012 DATA 76,60,128,72,8,169,118,76,60,128,72,8,169,119,76,60,128,72,8,169,121
10014 DATA 76,60,128,72,8,169,122,76,60,128

```

discrete frequencies, and plots the magnitude of each frequency component. The frequency spacing between each component is a function of the digitization frequency and the number of points that you have in the routine process. You then have to decide how many of those points you wish to process with the FFT. Using a larger number of points results in greater resolution but requires more processing time. For example, 512 points requires 16 minutes, 256 points requires 8 minutes, and 128 points requires 4 minutes. You can calculate the frequency spacing between each component with Equation 8.5.

$$F = 1/(t*N) \text{ Hz}$$

$$\begin{aligned} \text{where } F &= \text{the distance between each component} \\ t &= \text{the time increment between samples} \\ N &= \text{the number of processed points} \end{aligned} \quad (8.5)$$

If we were sampling an input signal at a rate of 1000 Hz, then t would be equal to .001 seconds. If we then decided to process 512 points, we would expect the distance between components to be equal to 1.95 Hz. This means that if we had digitized a 19.5 Hz sine wave, we would expect to see a peak in the magnitude of the output at the 11th point (the magnitude of the d.c. component occupying the first point).

You can do several things to enhance the utility of these routines:

1. Write programs to plot both the complex waveform and the Fourier transform with the digital plotter developed in Chapter 5.

2. Convert the BASIC program shown in Listing 8.3 into a machine language routine to greatly decrease the time spent processing the data.

To help assure yourself that you have entered the program correctly, I have included a simple program which generates a perfect 100 Hz sine wave and stores it in memory. After loading the main program, key in the following BASIC statements.

```

80 REM
90 Y=24830:T=0
91 FOR Z=0 TO N-1
92 Y=Y+2:X=1+SIN(2*3.14159*100*T)
94 T=T+7.8125E-4
95 POKE Y, 3:POKE (Y+1),100*X
96 NEXT Z
203 REM

```

If everything is working correctly, you should observe a peak in magnitude at the eleventh output.

The following four figures show the results of taking the FFT of 512 points of data from audio data digitized at a rate of 1000 samples per second. Figure 8.3 shows the spectrum of a violin solo. Notice how there are very distinct harmonics displayed. Figure 8.4 shows the spectrum of the speaking voice of a male radio announcer. Notice how the spectrum is rich in low frequency content. Figure 8.5 shows the spectrum of the singing voice of a popular singer who was holding a high clear note when the data was collected.

FIGURE 8.3 Fourier transform of violin solo.

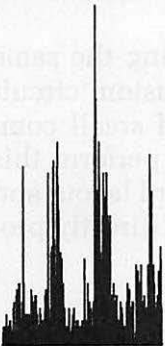


FIGURE 8.4 Fourier transform of the voice of a male radio announcer.

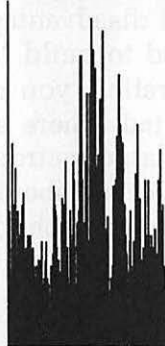
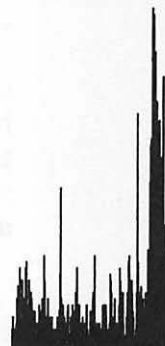


FIGURE 8.5 Fourier transform of the voice of a female singer.



APPENDIX A: CONSTRUCTION TECHNIQUES

Before you start to build, you should have some feeling for the construction options that are available to you. This will allow you to use appropriate construction techniques for specific circuits that you will be building. I am not going into a lot of detail concerning the actual construction process involved in each option, since this has been covered in detail in other publications. However, based on my personal experience, I will summarize the advantages and disadvantages of each process.

If you need to build ten or more boards having the same circuit configuration, you might want to have a custom circuit board constructed. There are usually a number of small companies in any large metropolitan area which can perform this service. Costs will run about \$500 for the circuit board layout and about \$20 apiece for each circuit board. The cost is directly pro-

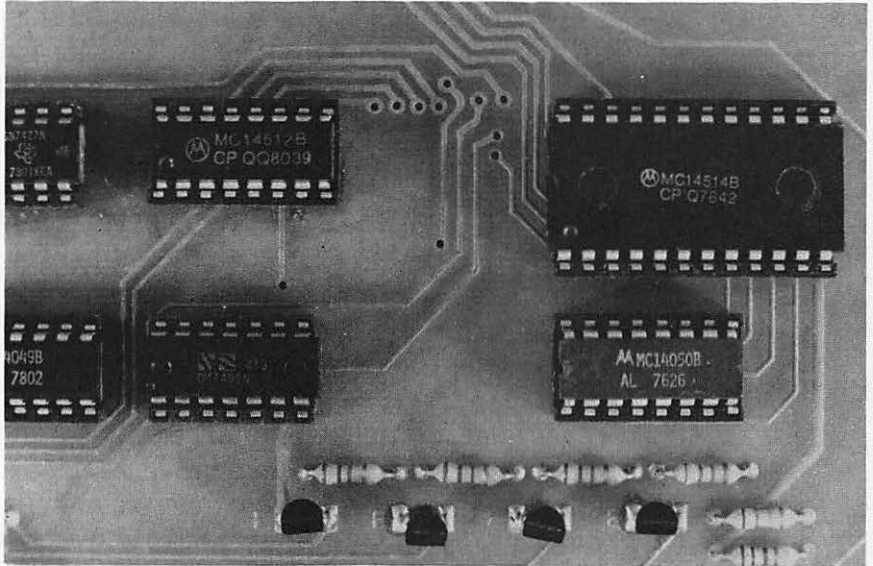
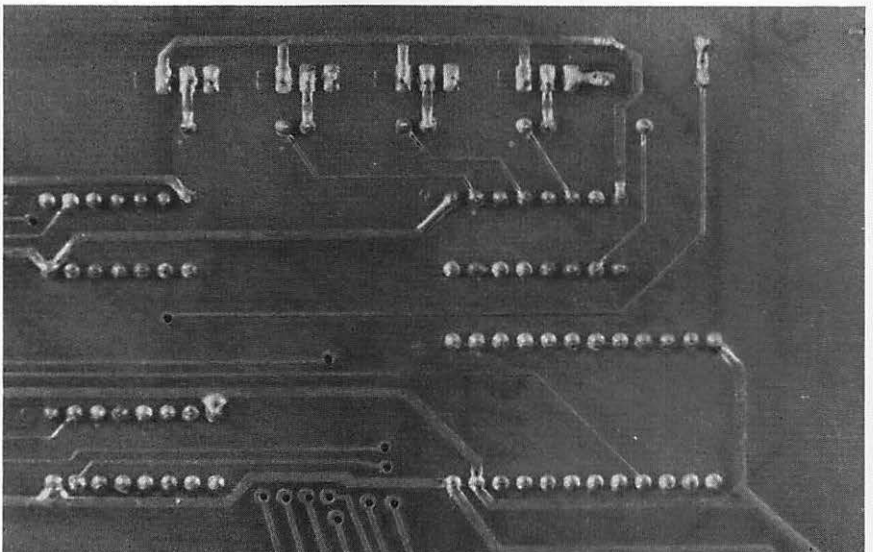


FIGURE A.1a Example of custom circuit board construction, front view.

portional to the complexity of the circuit design. Disadvantages include waiting about four weeks for the boards to be fabricated, the cost of fabrication, and the loss of modification flexibility. Advantages include decreased construction time for each circuit, increased reliability of operation, and an obviously professional appearance. Figures A.1a and A.1b show examples of this type of construction.

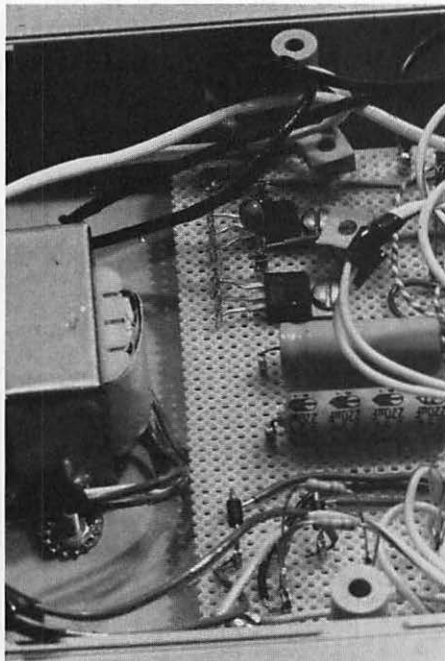
FIGURE A.1b Example of circuit board construction, rear view.



If you need to construct a single circuit that is very simple, I would recommend Vector board and push-in, solderable terminal type construction. Figure A.2 shows an example of this type of construction. Vector board (Vector Electronics Company, Inc., Sylmar, California 91342), Vector pins, and a tool to insert the pins can be purchased at your local electronics store. It is the least expensive method of construction and I have used it with success. However, this type of construction sometimes results in undesirable oscillations caused by output signal wires being in close proximity with input signal wires. Another possible problem is the introduction of high frequency noise, generated by the digital circuitry, into power supply lines. I would discourage its use with DIP packaged integrated circuits unless you are quite proficient with a soldering iron.

If you are adventurous, and have an abundance of time, you should consider laying out the foil pattern and then etching your own circuit boards (*Circuit Board Etching*, M. W. Houser, Davis Publications, New York, New York 10017, January 1981). This

FIGURE A.2 Example of Vector board construction.



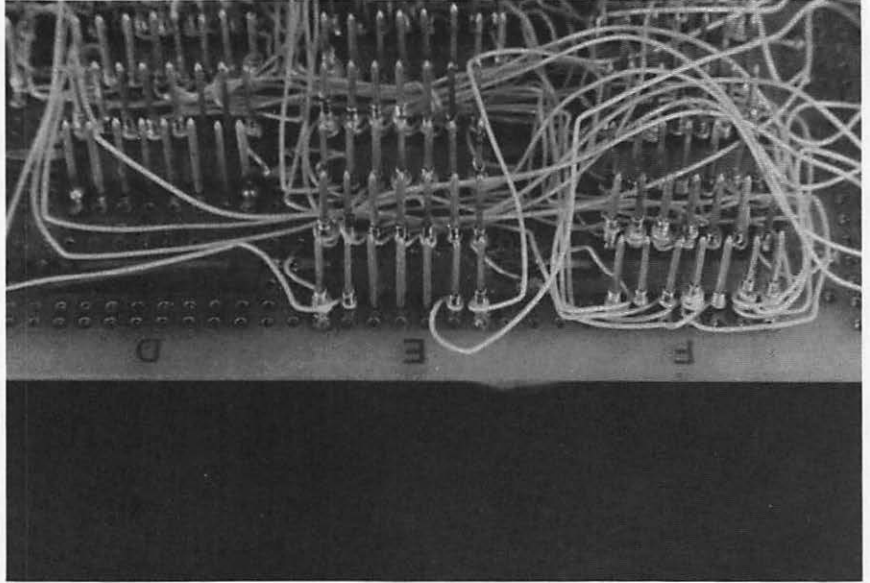


FIGURE A.3b Example of wire-wrap construction, rear view.

the project. Plastic boxes are very easy to work with and are quite adequate for the types of projects that I will be describing. Figures A.4a and A.4b show examples of this type of finished construction.

FIGURE A.4a Example of finished product.



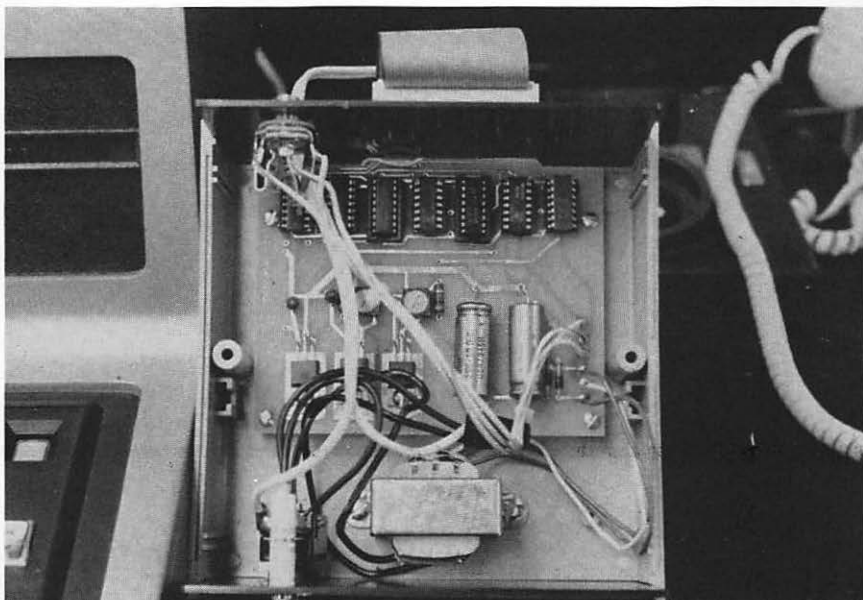


FIGURE A.4b Example of finished product.

APPENDIX B: OPERATIONAL AMPLIFIER THEORY

The operational amplifier (op amp) will be one of the most important tools you will have at your disposal when it comes time to interface some type of transducer to an A/D converter. Since most operational amplifiers have one output and two input ports, we can consider them to be a general three-port device. Their basic function is to amplify the difference between the two signals which are applied to the input ports. The amplified difference then appears at the output port. The relationship between the output voltage and the input voltages is dependent on the sign and magnitude of feedback which is supplied by the output to one or more of the inputs. As a result of feedback, we can configure our circuit to add, subtract, multiply, integrate, differentiate, compare, rectify, filter, and perform current-voltage conversion.

There are three general characteristics of the operational amplifier that make it a useful tool:

1. It has very high gain from either of the two input ports to the output port.
2. The input impedance at each input port is very high and the output impedance is very low.
3. It has very low gain when the same signal is applied to both input ports at once. Consequently, the amplifier is most sensitive to the voltage difference between the two inputs rather than the voltage difference between either input and ground. This allows the amplifier circuit to disregard many signals which would be classified as noise and to respond only to the signal coming from the transducer.

Since many of our projects deal with small magnitude signals, we will need to provide amplification, and possibly filtering, to increase our ability to resolve and accurately display these signals. The operational amplifier will allow us to combine a fifty-cent integrated circuit with a knowledge of Ohm's law, and become proficient at designing small signal amplifiers for our interface projects.

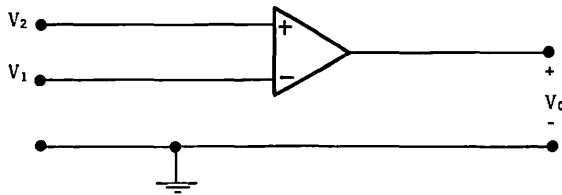


FIGURE B.1 Op-amp circuit symbol. A voltage at V_1 , the inverting input, is greatly amplified and inverted to yield an out-of-phase output at V_0 . A voltage at V_2 , the noninverting input, is greatly amplified to yield an in-phase output at V_0 .

Figure B.1 shows the operational amplifier circuit symbol. You should notice the two input terminals and the single output terminal. One of the input terminals has a negative sign associated with it (the inverting input) and one of the inputs has a positive sign associated with it (the noninverting input). There are two basic rules which we will be using to design and analyze op amp circuits:

1. When the op amp is operating in a linear region, the two input terminals are at the same voltage.
2. There can be no flow of current into either of the input terminals.

Armed with these two constraints we can begin our discussion of specific amplifier circuits.

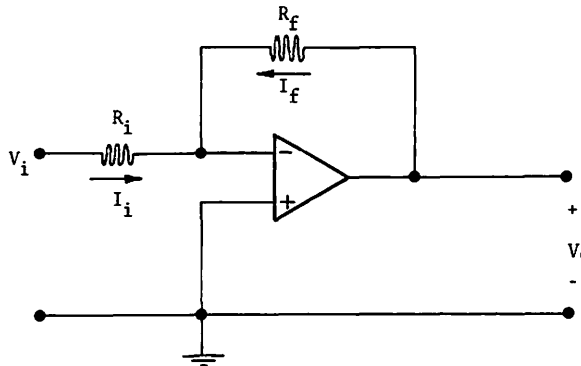
INVERTING AMPLIFIER

Figure B.2 shows a basic inverter circuit which is widely used for instrumentation purposes. The input impedance for this configuration is approximately equal to the magnitude of resistor R_i . Note that a portion of the output signal is fed back through resistor R_f to the negative input of the op amp. Reviewing the two rules which we have previously discussed, we know that both the positive and the negative inputs are at the same potential, which for this circuit is equal to 0 volts. We also know that the current flowing into or out of each of the two input terminals has to be equal to 0. Using Kirchhoff's current law we can write the following equation:

$$I_i + I_f = 0 \quad (\text{B.2a})$$

FIGURE B.2 Simple resistive inverting amplifier. A voltage at V_i , the inverting input, produces a voltage at the output, V_o , which is a function of the values R_i and R_f .

$$V_o = - \left[\frac{R_f}{R_i} \right] V_i$$



Using Ohm's law we can write the following two equations:

$$V_i - R_i I_i = 0 \quad (\text{B.2b})$$

$$V_o - R_f I_f = 0 \quad (\text{B.2c})$$

Solving equation B.2.2b for I_i , solving equation B.2.3a for I_f , and then substituting these two values back into equation B.2.2a, we obtain equation B.2d:

$$(V_i/R_i) + (V_o/R_f) = 0 \quad (\text{B.2d})$$

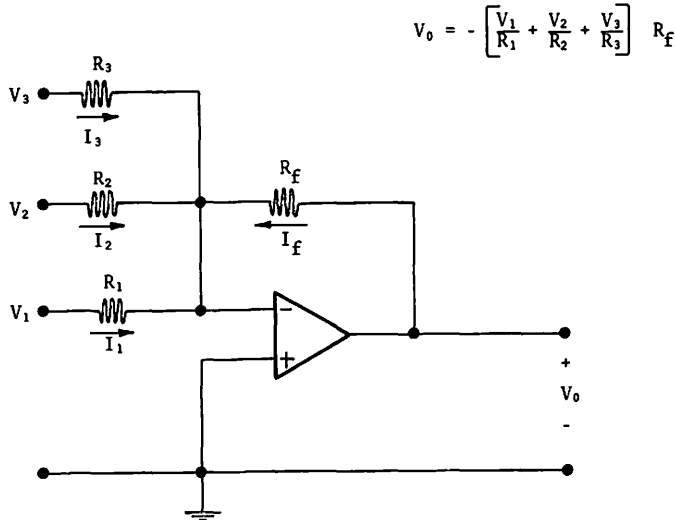
The voltage gain (K) for the circuit can then be written as follows:

$$K = V_o/V_i = -(R_f/R_i) \quad (\text{B.2e})$$

Figure B.3 shows a modification to the basic inverter circuit which allows us to add several input voltages together. Remembering our two basic rules, we know that the inverting and the noninverting input terminals are at the same potential and that there can be no current flowing into or out of either of these terminals. Summing currents at the inverting terminal we obtain the following equation:

$$I_1 + I_2 + I_3 + I_f = 0 \quad (\text{B.3a})$$

FIGURE B.3 Simple resistive inverting summing amplifier.



Using Ohm's law we can write equations for these currents in terms of the input and output voltages.

$$I_1 = V_1/R_1 \quad (\text{B.3b})$$

$$I_2 = V_2/R_2 \quad (\text{B.3c})$$

$$I_3 = V_3/R_3 \quad (\text{B.3d})$$

$$I_f = V_o/R_f \quad (\text{B.3e})$$

Substituting these equations back into equation B.3a we get equation B.3f:

$$(V_1/R_1) + (V_2/R_2) + (V_3/R_3) + (V_o/R_f) = 0 \quad (\text{B.3f})$$

Solving for the output voltage, expressed as a function of the input voltages and the resistance values, we get the following equation:

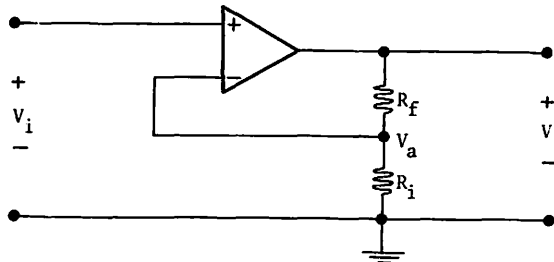
$$+ V_o = (R_f/R_1)V_1 + (R_f/R_2)V_2 + (R_f/R_3)V_3 \quad (\text{B.3g})$$

NONINVERTING AMPLIFIER

Figure B.4 shows an amplifier configuration which is sometimes called a *voltage follower* because the output voltage is the same polarity as the input voltage. This circuit is useful because the

FIGURE B.4 Simple resistive noninverting amplifier. A voltage at V_i , the noninverting input, produces at the output V_o , which is a function of the values R_i and R_f .

$$V_o = \left[1 + \frac{R_f}{R_i} \right] V_i$$



input impedance is extremely large; therefore, the circuit does not draw much current from the circuit that is attached to its input. Again we have a circuit that has a portion of the output signal fed back to the inverting input through the feedback resistor R_f . The two rules which we have discussed state that the voltage at the inverting input will be equal to the voltage at the noninverting input, and that there will be no current flow into or out of either input terminal. Using Ohm's law we can write an expression which will relate the magnitude of the voltage at point a to the output voltage V_o .

$$V_a = V_o R_i / (R_i + R_f) \quad (\text{B.4a})$$

We know that $V_a = V_i$, and if we substitute this back into equation B.4b we get the following equation:

$$V_i = V_o R_i / (R_i + R_f) \quad (\text{B.4b})$$

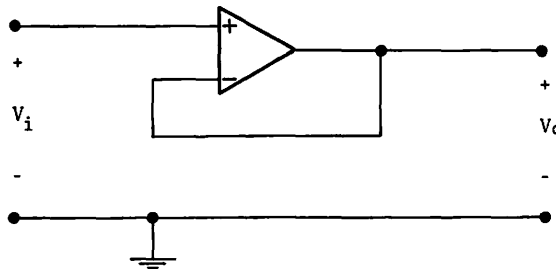
By rearranging this equation we can obtain an expression which gives the voltage gain of the circuit as a function of the resistors R_i and R_f .

$$K = V_o / V_i = 1 + R_f / R_i \quad (\text{B.4c})$$

By looking at equation B.4c, you should be able to predict what happens to the voltage gain as a resistor R_i becomes very large. Figure B.5 shows a circuit diagram where this resistance has been

FIGURE B.5 Simple unity gain voltage follower.

$$V_o = V_i$$

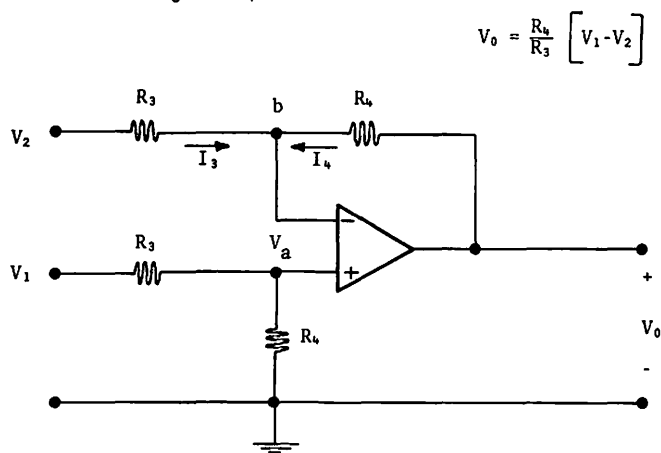


set equal to infinity (open circuit). This is a special circuit called the unity-gain voltage follower. Remembering that the voltages at the two input terminals have to be equal to one another, we can see that the output voltage is forced to be equal to the input voltage. This circuit is often used to provide impedance transformation from very high at the input to very low at the output.

DIFFERENTIAL AMPLIFIER

We found that we could add two or more signals together by using the inverting summing amplifier configuration. We will now take a look at a circuit which will allow us to subtract one signal from another. Figure B.6 shows an operational amplifier used as a differential amplifier. One of the advantages to using this circuit is that the circuit subtracts voltages applied to both inputs therefore producing an output only when the input voltages are not equal. This is useful when we have a balanced input circuit and need to reduce noise which is common to both of the inputs. Remembering that the voltage at the inverting input has to equal the voltage at the noninverting input and that no current can flow into or out of either terminal, and using Ohm's law, we can write

FIGURE B.6 Simple resistive differential amplifier. This circuit is configured so that the output voltage, V_0 , is equal to the difference of the input voltages, V_1 and V_2 , multiplied by a constant which is a function of the resistors R_3 and R_4 .



an equation for the voltage at point a in terms of the input voltage V_1 :

$$V_a = V_1 R_4 / (R_3 + R_4) \quad (\text{B.6a})$$

This voltage has to be equal to the voltage at point b. We also know that the following relationship exists between current I_3 and current I_4 :

$$I_3 + I_4 = 0 \quad (\text{B.6b})$$

We can write the following equations to relate the output voltage to the input voltages:

$$I_3 = (V_2 - V_b) / R_3 \quad (\text{B.6c})$$

$$I_4 = (V_o - V_b) / R_4 \quad (\text{B.6d})$$

Substituting equation B.6a into equations B.6c and B.6d and then substituting these equations back into equation B.6b gives the following result:

$$V_o / R_4 = (V_1 - V_2) (1 / R_3) \quad (\text{B.6e})$$

Rearranging this equation gives us an equation which relates the output voltage to the input voltages as a function of the resistance values.

$$V_o = (R_4 / R_3) (V_1 - V_2) \quad (\text{B.6f})$$

INTEGRATING AMPLIFIER

The integrating amplifier (see Figure B.7) is a circuit whose gain varies as a function of frequency making it different from the circuits we have previously considered. Detailed analysis would show that the output voltage will be related to the magnitude and the frequency of the input voltage by the following equation:

$$V_o / V_i = -1 / (wRC) \quad (\text{B.7a})$$

$$V_o = - \frac{1}{RC} \int_{t_1}^{t_2} V_i dt$$

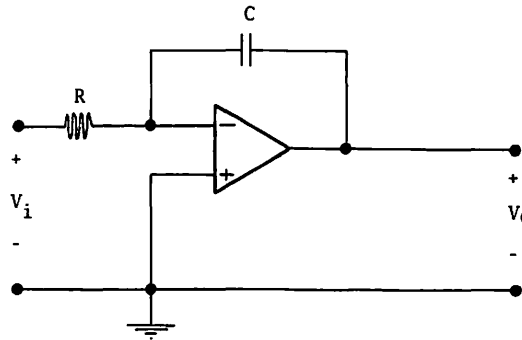


FIGURE B.7 Simple integrating amplifier. The time constant of this circuit is directly related to the product of resistor R and capacitor C.

This equation can also be written as a function of time; this may actually be a more familiar representation for the majority of readers:

$$V_o = - \frac{1}{RC} \int_{t_1}^{t_2} V_i dt \quad (\text{B.7b})$$

Figure B.8 shows an integrator which has had a resistor added in parallel with the capacitor. This provides us with a circuit, called a *low pass filter*, which is useful for attenuating high frequency noise. As with the integrator, the frequency response is given by the ratio of feedback to input impedance. However, at low frequencies the circuit behaves like an inverter because the impedance of capacitor C is large when compared to the resistor R_2 . At higher frequencies, the circuit behaves like an integrator because the impedance of C becomes smaller than R_2 . Equation B.8a expresses the gain of this circuit as a function of frequency:

$$V_o/V_i = (R_2/R_1)(\omega/R_2C)(1/1 + \omega/R_2C) \quad (\text{B.8a})$$

At low frequencies this equation reduces to the following equation which will be recognized as the equation for the simple inverting amplifier:

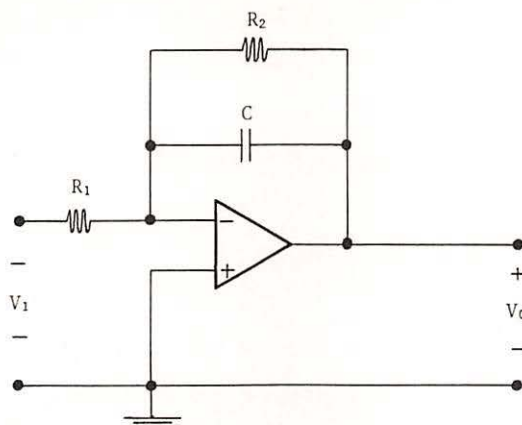


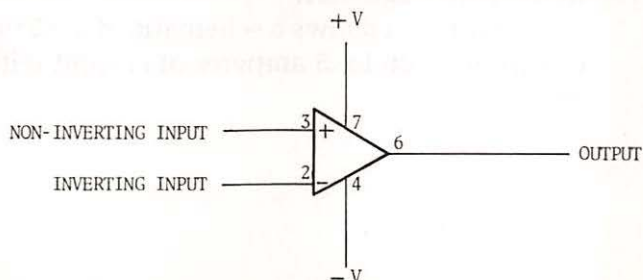
FIGURE B.8 Simple low-pass filter. The low frequency gain of the circuit is a function of R_1 and R_2 . The 3_{dB} corner frequency is determined by the values of capacitor C and resistor R_2 .

$$V_o/V_i = -(R_f/R_i) \quad (\text{B.8b})$$

At high frequencies equation B.8a will reduce to equation B.7a. When a circuit designer needs to limit the frequency response of a wide-band amplifier, there is no need to add a separate stage but only to add the correct size capacitor to the existing circuit.

There are many operational amplifiers which you could use in the circuits which we have discussed. An op-amp which is commonly available, low in cost, and easy to use is the 741. This is a general purpose device which provides output short circuit protection, internal phase compensation, and latch free operation. Another device which is readily available is the CA3140. This operational amplifier differs from the 741 by having MOS/FET input stages which provide high input impedance. The pin configuration for both of these devices is shown in Figure B.9.

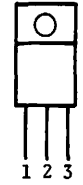
FIGURE B.9 Pin assignment for the 741 and CA3140T operational amplifiers.



APPENDIX C: POWER SUPPLIES

The widespread availability of three terminal regulators makes the design and construction of power supplies for digital and analog circuits almost a simple task. In the past, power supplies were generally made from a number of discrete components. Today, advancements in fabrication techniques related to monolithic semiconductors have enabled a number of discrete components to be incorporated into a single chip. This has resulted in a series of low-cost positive and negative regulators which are extremely easy to use, and difficult to abuse. Monolithic voltage regulators usually have a series pass transistor power control element, a reference source, and current overload protection all on the same chip. As a result of this, no adjustments are required to set up the output voltage, and it is virtually impossible to destroy the regulator.

Figure C.1 shows a schematic of a + 5 volt power supply that can provide up to .5 amperes of current with adequate heatsinking.



MC7805T

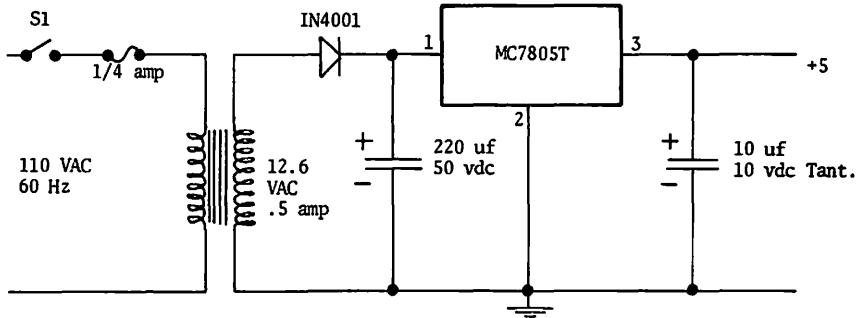


FIGURE C.1 Positive 5-volt power supply.

Figure C.2 shows a schematic of a +12 and -12 power supply that can provide up to .5 amperes of current with adequate heatsinking.

Once you have constructed either of the supplies, you should check to make sure that the output voltage is what you were expecting it to be. You should be concerned about the following:

1. Treat the 110-volt line voltage with care.
2. Make sure that the polarity of the filter capacitors is correct.
3. Make sure that the polarity of the diodes is correct.
4. The positive and negative regulators do not use the same pin configuration. Be sure that you have used the correct pins for input, output, and ground.

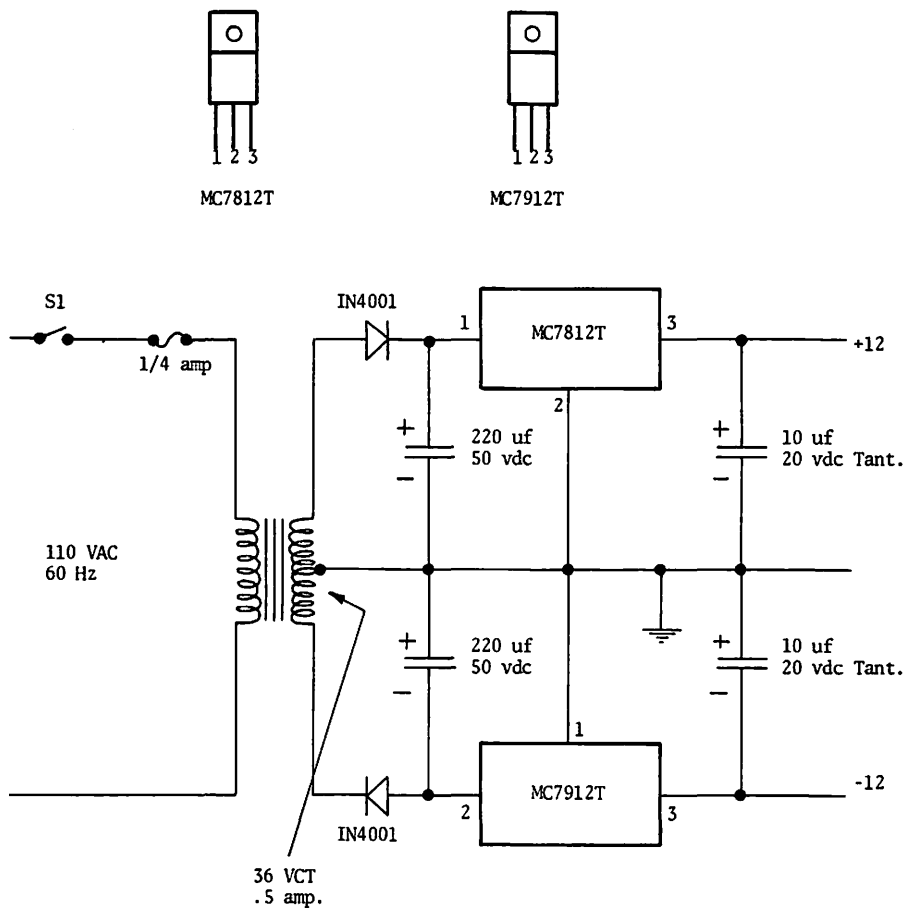


FIGURE C.2 Positive 12, negative 12 volt power supply.

Accumulator in parallel data transfer, 10, 14
A/D (analog-to-digital) conversion, 8, 22-69
 in biofeedback temperature monitor, 103-4
 dual slope system of, 29-31
 eight-bit, low speed, see MC14433 A/D converter
 eight-bit, successive approximation type (block figure), 58
 general considerations on, 22-29
 in heart rate biofeedback monitor, 109
 number of binary output bits and resolution in, 56-59
 relative system error in, 24-28
 resolution (dR) in, 24-28, 56-57
 sampling frequency (F_s) in, 24-25, 28-29
 successive-approximation scheme of, 54-55
 ten-bit, high speed, see AD571 A/D converter
AD558 DACPORT D/A converter, 70-77
 in automatic test equipment systems, 72-77
 functional block diagram of, 71
AD571 A/D converter, 50-69
 block diagram for, 53
 isolation of input signals, 62
 pin assignment for, 53
 software for, 64-69
 successive-approximation conversion scheme of, 54-55
 transfer of data to Apple II memory, 62, 67
 typical test sequence, 54-55
AD580 voltage regulator, 37-39
Address bus of 6502, 5, 8-10
 in parallel data transfer, 10, 14
Address decoder, 4-16 line, 8, 9
Aliasing error, 28-29
Amplifiers, see Operational amplifiers
Analogue Devices, 50
 See also AD558 DACPORT D/A converter; AD571 A/D converter; AD580 voltage regulator
Analog signal, definition of, 26, 27
Analysis, see Data analysis techniques
AO address line in parallel data transfer, 12-14
Applesoft BASIC, 1, 2
 See also BASIC programs: BASIC statements
Asynchronous-serial format, definition of, 15
Auto Search control unit interface connector, 119-21
Automatic test equipment systems, 72-77

BASIC program(s):
 for Betamax/Apple II interface, 121, 123-24, 127-40
 for bidirectional serial interface, 98-100
 for biofeedback heart rate monitor, 112-15
 for biofeedback skin temperature monitor, 104-6
 demonstration, for HILOT, 79-80
 for digitized data stored on disc, 89-93
 exponential regression, 144-46
 fast Fourier analysis, 147-51
 for high speed A/D converter interface, 64-68
 for HILOT Digital Plotter, 81-86
 linear regression, 142, 143
 for low speed A/D converter interface, 41, 43-46, 49-51, 104
 switch status, 22, 25
 for testing of incandescent lamps, 74-75
BASIC statements/commands, 67
 PEEK(-16143), 14
 POKE-16143.0, 97
 POKE-16143.1, 97
 POKE-16143.X, 10-14
 POKE-16175.0, 79
 POKE-16175.1, 79
 X=PEEK(-16143), 22
 100K=PEEK(-16143), 97
 100 POKE-16143.01, 62, 63
 100 POKE-16143.09, 62, 63
 110 FOR I=0 TO 255, 72

110 IF K<128 THEN PRINT "O", 97
 120 IF K>128 THEN PRINT "1", 97
 120 POKE-16143.I, 72
 130 GOTO 100, 97
 130 NEXT I, 72
 140 FOR I=1 TO 100, 63
 142 NEXT I, 63
 150 GOTO 100, 63
 199 END, 72
BEGINNING OF TAPE* status signal, 119-20
Betamax, see Sony Betamax S10-320
Bidirectional data bus, see Data bus of 6502
Bidirectional serial interface, 96-100
Binary coded decimal (BCD) format, 35
Binary signal, determining status of, 22-25
Biofeedback, 101-16
 definition of, 101-2
 heart rate monitor, 108-16
 skin temperature monitor, 102-8
BIPOLAR OFFSET CONTROL, 52-54
Bipolar vs. unipolar mode with AD571, 52-54
Bit pattern:
 in asynchronous-serial transmission, 15-16
 from HILOT demonstration program, 80
B&K 3010 Function Generator, 76, 77
Buffers:
 in high speed A/D converter interface, 62
 three-state (MC14503), 11-14, 37-39, 62

CA3140 operational amplifier, 187
Capacitors, determining capacitance of, 144-46
CASSETTE IN* status line, 119-20
CE* control input, 71
 Circuit board construction techniques, 152-57
Circuit Board Etching (Houser), 154-55
Clock frequency:
 and conversion rate, 35
 as function of clock resistor, 35, 38
Clock resistor (R_c):
 clock frequency as function of, 35
 in low speed A/D converter configuration, 37, 39
Clock system of 6502, 6-7
Clocked flip-flops, 11-14
Coefficient of determination, 142-44
Comparator, internal, in AD571, 54
Computer-assisted instruction (CAI), 118-19
Construction techniques, 21, 152-57
Continuous signal, definition of, 26, 27
Control lines:
 in Betamax/Apple II interface, 119-22
 of 6502 microprocessor, 5-8, 8
 See also specific lines
Control systems, 72-77
Conversion:
 parallel-to-serial, 15-18
 serial-to-parallel, 15, 17-20
 See also A/D conversion; D/A conversion
 Conversion rate and clock frequency, 35
CONVERT* control line, 54, 62
COUNT* timing line, 119-21
Crystal controlled time base oscillator, 59, 61, 63
CS* control input, 71
Curve fitting, see Data analysis techniques

D/A (digital-to-analog) conversion, 70-77
 in automatic test equipment systems, 72-77
Data analysis techniques, 141-51
 exponential regression, 144-46
 fast Fourier transform (FFT), 146-51
 linear regression, 142-44
Data bus of 6502 microprocessor, 4-8
 in automatic test equipment systems, 74, 77
 in parallel data transfer, 10, 14
Data lines:
 multiplexed, with MC14433 A/D converter, 35-38
 See also specific lines
DATA READY* status line, 56, 62

Davis Publications, 154-55
DB-25 connector, 5, 6
Decwriter II terminal, 78-79
Device address in parallel data transfer, 14
Differential amplifiers, 104, 110, 164-65
Digit select timing, 35-37
Digital plotting, 8, 78-84
 See also HILOT Digital Plotter
Digitized data, 78
 fast Fourier analysis, 146-51
 from high speed A/D converter interface, 68-69
 from low speed A/D converter interface, 50, 52
 plotting of, 89-94
Discrete signal, definition of, 26, 27
DISPLAY UPDATE (DU) control line, 35, 37, 39
dR, see Resolution
DS* (device select) line, 8
 in parallel data transfer, 12-14
Dual slope conversion process, 29-31
Dynascan Corporation, 77

Electronics Industry Association (EIA):
 electrical specifications of, 5
 See also RS-232-C
END OF CONVERSION (EOC) status line, 35-39, 41
ESC key, 99
Exponential regression, 144-46

FAST FORWARD* control line, 119-22
Fast Fourier transform (FFT), 146-51
Feedback:
 definition of, 102
 See also Biofeedback
Fourier transform, 146-51
Frequency (f_c), 28
 See also Sampling frequency (F_s)

Handshaking, definition of, 3-4
Heart rate biofeedback monitor, 108-16
 adjustments and voltage checks, 111-13
 analog circuitry for, 109-11
 software, 112-15
HILOT Digital Plotter, 79-94
BASIC program for, 81-86
 command characters, 81
 control of pen position, 80-81
 demonstration program for, 79-80
 increasing values of time constant, 81, 87
 interface connections with Apple II, 79
 maximum range of plot size, 81, 88
 program for data stored on disc, 89-93
 TTL to RS-232-C conversion, 79-80
 vector notations, 80-81
Houser, M. W., 154
Houston Instruments, 79
 See also HILOT Digital Plotter

INHIBIT control line, 8
Input port, parallel (figure), 13
I/O (Input/Output) port:
 definition of, 3-4
 interrupt input, 83
 location 5, 8, 79
 location 7, 8, 10
Input voltage:
 in dual slope process, 30-35
 number of bits and resolution, 57-59
 with AD571 A/D converter, 50-58
Integrating amplifiers, 165-67
Intel Corporation, 1
Internal clock of MC14433, 35, 36
Internal output latches, 35
Inverting amplifiers, 56, 57, 160-62
IRQ* line, 63
Isolation of input signal, 62

Lamp intensity testing, 74-75

Least squares method:

exponential, 144-46

linear, 142-44

Linear regression, 142-44

Logic level 1, 5

Logic level 0, 5

Low-pass filter, 166-67

Machine language commands:

LDASC0F1, 14, 17, 38

LDASC0F2, 62

LDASC0F4, 62

LDA#\$X, 10

STASC0F1, 10, 17

STASC0F2, 38

Machine language routine(s):

for Apple II/Betamax interface, 121, 124-27

for bidirectional serial interface, 98-100

for heart rate biofeedback monitor, 114

for high speed A/D converter interface, 64-65,

67-68

for low speed A/D converter interface, 40-42,

46-48

Mark, definition of, 5

Maximum voltage range (V_{max}), 26

MC14433 A/D converter, 29-32

BASIC programs for, 41, 43-46, 49-51

block diagram of, 32

in configuration for Apple II, 37-39

digit select timing, 35-37

dual slope conversion process, 29-31

integrator output, 31

internal clock, 35, 36

machine language routines for, 40-42, 46-48

pin assignment, 33

resistor, capacitor, and battery combination,

46-52

three-state buffers, 37, 39

transfer of data to output latches, 35

MC14503 three-state buffers, 11-14, 37-39, 62

Modems, 95-97

Most significant digit (MSD), 35-37

Motorola Semiconductor Products, 29

See also MC14433 A/D converter; MC14503

three-state buffers

MSB of AD571 A/D converter, 54-55

Multiplexed data lines, 35-38

Multiplexer, 59, 60

Neurons, 101-2

Noninverting amplifiers, 34, 56, 162-64

Operational amplifiers (op amps), 34-35, 56,

57, 158-67

for Apple II/Betamax interface, 121-22

basic rules with, 159-60

circuit symbol, 159

differential, 104, 110, 164-65

general characteristics of, 159

integrating, 165-67

inverting, 56, 57, 160-62

in isolation of AD571 input signal, 82

noninverting, 34, 56, 162-64

Oscillator, crystal controlled time base, 59, 61,

63

Output port, parallel (figure), 12

Output voltage (Y), 26

Overrange (OR), 37, 38

Parallel data transfer format, 10-14

definition of, 4-5, 10

typical input port circuit diagram, 13

typical output port circuit diagram, 12

Parallel-to-serial conversion, 15-18

PEN UP control function, 81

Peripheral-board connectors, 5, 7-9

Photocell in heart rate monitor, 109-12

Photoplethysmography, 108

Plot dimensions, maximum and minimum, 81,

88

Plotting, see HILOT Digital Plotter

PLAY* control line, 119-22

POKE, see BASIC statements

Polarity and low speed A/D converter, 29, 37, 38

Port:

definition of, 3-4

See also I/O port

Power supply, design and construction of,

168-70

Programs, see BASIC programs; Machine

language routines; Software

Pulse detector in heart rate monitor, 109-11

Quality control testing, 72-77

R_c, see Clock resistor

READ BASIC statements, 67

Reference voltage, 31-33, 104

Relative system error, 24-28

Resistive amplifiers, see Operational amplifiers

Resistive voltage divider network, 34

Resolution (dR), 24-28, 56-60

definition of, 26

number of binary output bits and, 57-60

REWIND* control line, 119-22

RM-300 Auto Search control unit interface

connector, 119-21

RS-232-C specification, 5

connector (DB-25), 5, 6

RS-232-C to TTL interface, 17-20, 79-80

R/W* (READ/WRITE*) control line, 6, 8

in parallel data transfer, 12-14

Sampling frequency (F_s), 24-25, 28-29

crystal controlled oscillator for measurement

of, 59, 61, 63

fast Fourier analysis, 146-51

See also Digitized data

Serial data transfer format, 14-20

asynchronous, 15-16

definition of, 4, 5, 15

SERIAL INPUT line, 97

Serial-to-parallel conversion, 15, 17-20

741 operational amplifier, 167

Sikonowitz, W., 155

6501 microprocessor, 5

6502 machine code, 2

See also Machine language commands;

Machine language routines

6502 microprocessor:

architecture of, 5-7

clock system of, 6-7

peripheral-board connectors of, 5, 7-9

pin configuration of, 6-7

speed of, and conversion decoding and

storage, 37-38

Skin temperature biofeedback monitor, 102-8

analog circuitry for, 103-4

diagrammatic representation of, 106-8

software for, 104-6

SLO-320, see Sony Betamax SLO-320

Software:

BASIC program for HILOT, 81-88

Betamax/Appie II demonstration, 127-40

Betamax/Appie II testing, 122-24

for bidirectional serial interface, 98-100

for D/A converter, 74-75

demonstration for HILOT, 79-80

exponential regression, 144-46

for fast Fourier analysis, 147-51

for heart rate biofeedback monitor, 112-15

for high speed A/D converter interface, 64-69

for incandescent lamp test, 74-75

linear regression, 142-44

for low speed A/D converter interface, 38-51

for plotting digitized data on disc, 69-93

for RS-232-C to TTL conversion, 17-20

for skin temperature biofeedback:

monitor, 104-6

switch status, 22, 25

videotape location count, 124-27

Sony Betamax SLO-320 videocassette recorder,

119-40

BASIC demonstration routine, 127-40

machine language routines, 121, 124-27

op amp for, 121-22

schematic for, 120-21

storage of program on videotape, 121-22

testing procedure, 122-24

Space, definition of, 5

Spectrum and Fourier transform, 146-51

START bits, 15, 16

Status lines:

for Apple II/Betamax interface, 119-20

See also specific lines

STOP bits, 15, 16

STOP* control lines, 119-20

STROBE control line, 8

Successive-approximation conversion scheme,

54-55

See also AD571 A/D converter

Switch status, 22-25

Temperature monitor, see Skin temperature

biofeedback monitor

Test equipment systems, 72-77

Thermistor:

in biofeedback skin temperature:

monitor, 104-8

definition of, 104

in linear regression example, 142-44

Three-state buffers, 11-14, 37-39, 62

Time base oscillator, crystal controlled, 59,

61, 63

Time constant in plotting, increasing values of,

81, 87

Timing signal in Apple II/Betamax interface,

119-21

Transducers, definition of, 22-24

Truth table on multiplexed data lines, 38

TTL to RS-232-C interface, 16-18, 79-80

2N3904 transistor, 63

Two-phase clock system, definition of, 6

Underrange, 37

Unipolar vs. bipolar mode with AD571, 52-54

Variables, see Data analysis techniques

Vector board, construction with, 154

Vector notations for HILOT Digital Plotter,

80-81

Video playback device, 117-40

in computer-assisted instruction, 117-19

interface with Apple II, see Sony Betamax

SLO-320

Video terminal, Apple II as, 95-100

Voltage:

analog, 26

EIA standards, 5

maximum range (V_{max}), 26

reference, 31-33, 104

RS-232-C interface, 16-18

in sampling, 28-29

See also Input voltage; Output voltage

Voltage divider circuit, 34

Voltage followers, 162-64

Voltage regulator, 168

AD580, 37-39

in heart rate biofeedback monitor, 110

in skin temperature biofeedback monitor,

103, 104

Waveforms:

analog, digital generation of, 72

continuous vs. discrete (figure), 27

in fast Fourier analysis, 146-51

sampling and, 28

See also Sampling frequency

Wire-wrapping Techniques (Sikonowitz), 155

Z6000, 4

RICHARD C. HALLGREN

interface projects for the apple II

This practical manual explains how computer users can better utilize the computational and control capabilities of their personal computer.

Written for the Apple II user with some previous computing experience, INTERFACE PROJECTS FOR THE APPLE II provides a series of complete interface projects that are easily built and enable you to discover the computer's capabilities as the project is constructed.

Inside these pages you'll find dozens of fully tested hardware projects—plus all the necessary interfacing software—that you can use as is or simply modified to meet your specific needs. It includes important information on:

- review of data transfer formats
- digital and analog conversions with the Apple II
- serial applications on the Apple II
- biofeedback
- how to control a video playback device
- data analysis of sampled signals
- construction techniques
- power supplies
- and more.

PRENTICE-HALL, Inc.,
Englewood Cliffs, New Jersey 07632