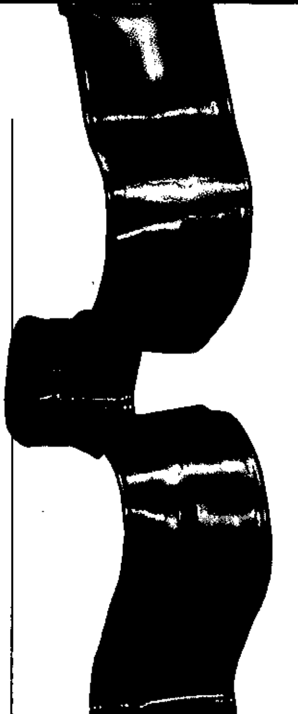
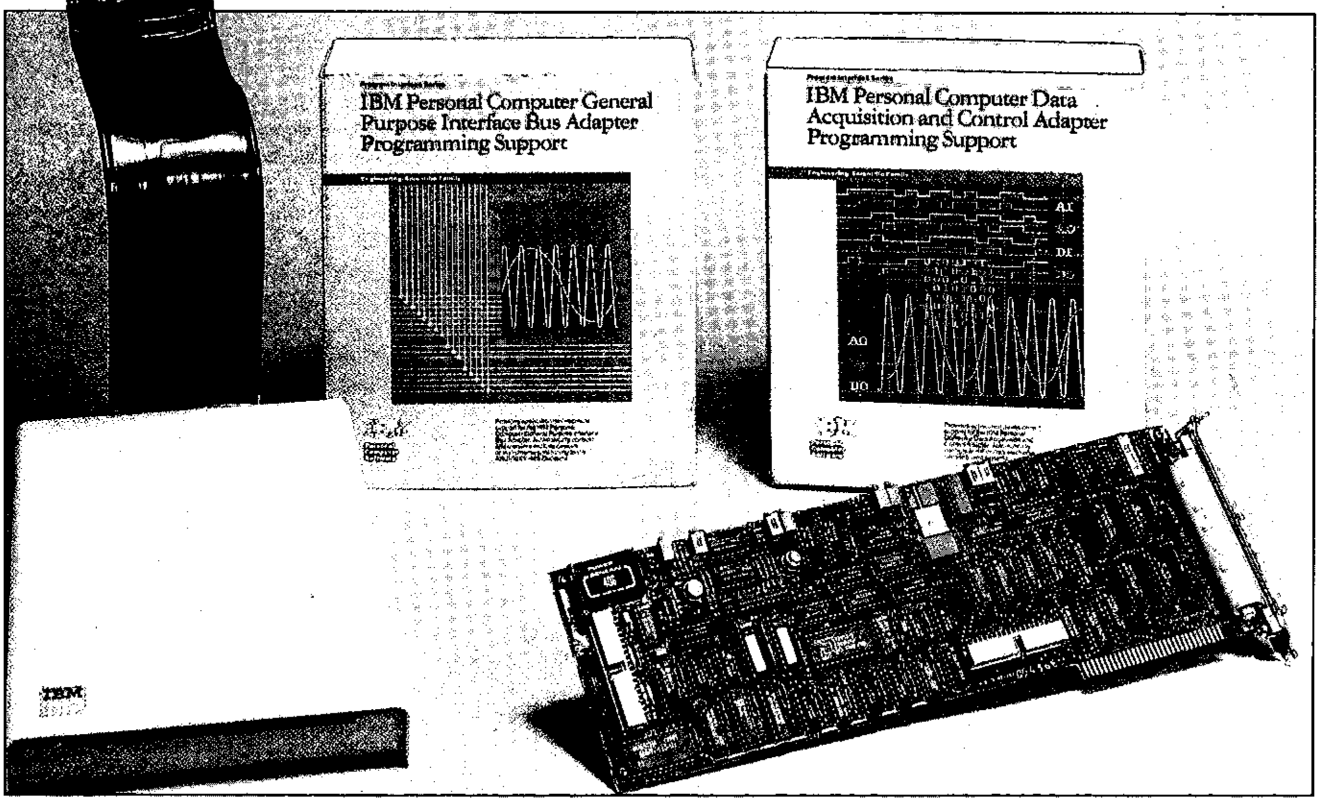


Data Acquisition

IBM's Data Acquisition and Control Adapter is fast, and it lets the user program in BASIC, FORTRAN, or C.



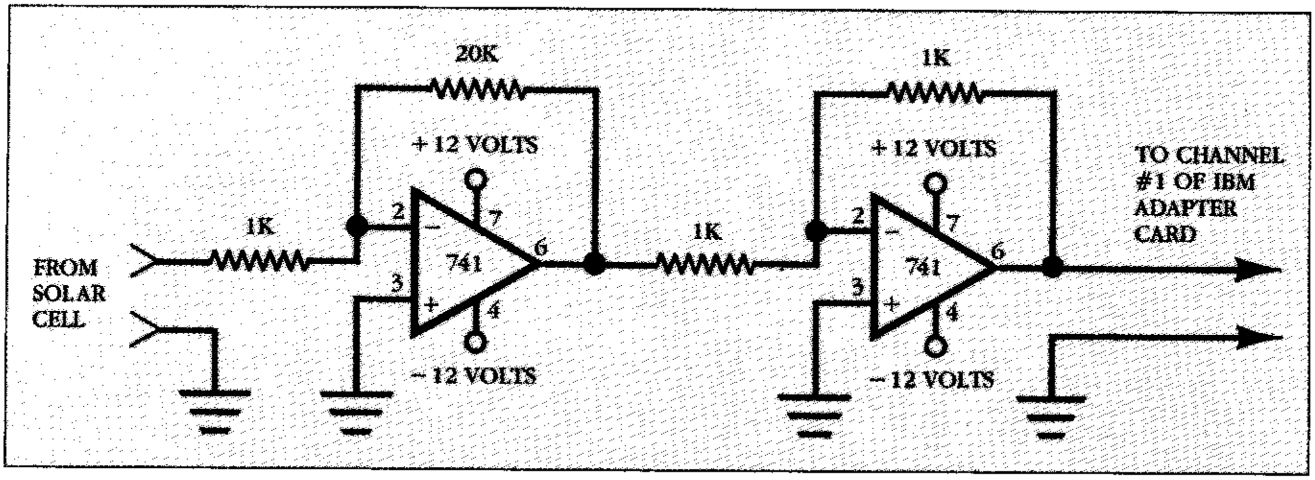
PHOTOGRAPH • STEVE BORNS

WILLIAM MURRAY

The IBM Data Acquisition and Control Adapter with Programming Support (commonly referred to as the IBM Adapter) is one of IBM's entries into the growing market of analog-to-digital (A/D) and digital-to-analog (D/A) data-acquisition boards for the IBM PC/XT and PC/AT family. The IBM Adapter permits the user to have four channels of multiplexed analog input, two channels of analog output, one 16-bit binary input port, one 16-bit binary output port, and one 16-bit addressable counter.

The software support provided with the board allows the user to program in FORTRAN (version 2.0 and Professional), C (Lattice version 2.0), and BASIC (Interpreted or Compiled). The diskette that accompanies the well-written manual contains the "link-time" object modules for interfacing compiled languages with the adapter. For interpreted BASIC, the diskette contains a "header" program that can merge with the user's BASICA program (it becomes lines 1 to 100 of the program) for interfacing to the adapter.

FIGURE 1: *Amplifier Circuit*



Operational amplifiers are necessary to increase the output of the solar cell to a level compatible with the adapter's input.

The IBM Adapter requires an IBM PC/XT or PC/AT with a minimum 64KB of memory (Compiled BASIC, C, and FORTRAN require more memory); one 320KB disk drive; a graphics card, if collected data points are to be plotted; Data Acquisition and Control Adapter (with software support); one of the above-mentioned languages, with DOS 2.0 or later (Professional FORTRAN requires 2.1 or later); and (optional) the IBM Distribution Panel (a cable and wiring interface to the adapter card).

Users who are seriously interested in data acquisition for the XT or AT should read the *PC Tech Journal* reviews of the Tecmar Lab Master ("Digital-to-Analog, Analog-to-Digital," Peter G. Aitken, March 1984, p. 104) and the DASCON-1 ("Poor Richard's Converter: From A to D," William H. Murray, September 1984, p. 30).

The IBM Adapter board fits into a full-size slot in the PC and has a 60-pin interface connector on the back panel. This connector can be attached directly to the circuit under test or wired to the IBM Distribution Panel. The panel, which has a four-foot shielded cable, makes it easy to connect "test circuits" to the adapter with a screwdriver and connecting wires. The panel provides 88 barrier-type screw terminals for access to all 60 lines from the adapter; frequent grounds are also present.

The adapter contains several switches that configure the board:

- The two analog outputs can be set independently for outputs of −5 to +5, −10 to +10, or 0 to +10 volts.
- The four analog inputs are set together for −5 to +5, −10 to +10, or 0 to +10 volts.
- The adapter number (0, 1, 2, 3) allows up to four adapters to be used in one computer.
- IRQ (interrupt level) can be set for levels 3, 4, 5, 6, or 7. The recommended level is 7.

Appendix C of the Programming Support manual implies that IBM supports expansion devices that "increase the capabilities of the internal adapter." These devices are purported to expand the number of analog input and output channels, to allow thermocouple and transducer interfacing, and to improve A/D resolution. No clear description of these devices and no names or IBM product numbers are given, so it remains unclear what IBM offers in the way of instrumentation amplifiers and other interfacing hardware. It may be necessary to construct such hardware or to consult a third-party vendor. Construction, however, is not difficult.

The four A/D inputs provide a resolution of .0025 volts over a 10-volt range with the installed 12-bit converter. If improved resolution is required, a 16-bit A/D converter can be added with the expansion bus interface to provide a resolution of .0001 volts over a 10-volt range. The four A/D inputs are multiplexed. The manual states that 7,999 samples/second are possible in the normal performance mode and that 12,500-16,000 are possible in the extended performance mode, which requires special access to device drivers.

The IBM Adapter provides no onboard instrumentation amplifiers. If the devices that are to be interfaced have outputs that fall outside all of the three selectable A/D input voltages (solar cells, thermocouples, or transducers come to mind), a special amplifier will have to be purchased or built.

The analog outputs and binary I/O are restricted to 7,999 samples/second in the normal mode, with 17,000 samples/second possible with system interrupts disabled. (If some other part of the system requires regular use of system interrupts, the interrupts will have to remain in effect). All binary inputs and outputs are TTL-compatible, providing up to 16 bits of digital information.

The Programming Support software and manual are among the nicest IBM has to offer. The diskette contains several types of programs. Linkable object modules and sample programs are included for each supported language. As might be expected, the only graphics examples are for interpreted BASIC. The sample BASIC program provided on the diskette will *not* work with compiled BASIC, because it uses VIEW and WINDOW, graphics commands that are not supported by compiled BASIC. This should not be a major problem because the program does work nicely with interpreted BASIC.

The manual is divided into seven chapters and three appendices. The first two chapters contain detailed discussions of the converter and data-acquisition techniques, including such topics as A/D and D/A conversion, sampling theory, synchronizing data (handshaking, triggering, and clocking), and event-counting. Chapter 3 introduces the various functions that are available for interfacing software to the adapter (see table 1). Chapter 4 describes the function arguments that must be used when calling the functions. Chapters 5, 6, and 7 describe the calling syntax for each function from each of the three supported languages; abundant example code is included.

IBM ADAPTER

Each language (BASIC, FORTRAN, or C) permits complete control of the IBM Adapter. The execution speed of all three compilers were about equal, because the rate at which the adapter provides samples is the same regardless of how the adapter is accessed.

As soon as a programming language has been selected, an installation program loads a CONFIG.SYS file (containing the invocation of the adapter's device driver) onto a formatted diskette. The system must be booted with that diskette; it could crash if a device driver that is not loaded is called.

Three examples, one in each of the supported languages, will help explain how the adapter is accessed. The C example asks the adapter to do relatively simple tasks, the FORTRAN example asks it to do slightly more complex jobs, and the compiled BASIC example requires the most complicated work.

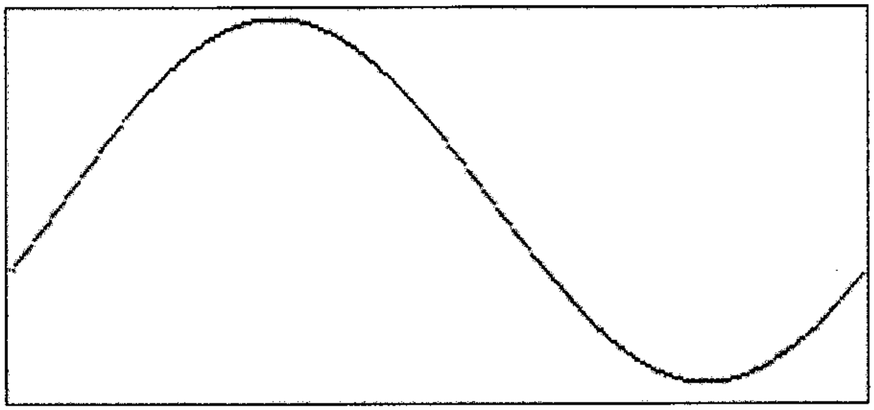
Listing 1 shows a simple C program that makes use of the BINS (Binary Input Simple) call. BINS inputs a 16-bit binary word from the adapter that it stores in a variable called v. This variable is modified slightly to give the correct weight for the particular data bit (for example, bit 0=1, bit 1=2, bit 3=4 . . . and bit 15=32,768). The program is designed to report continuously the current data value to the screen.

The variable **adapt** refers to the DA adapter number. If only one adapter is in place, this number will be 0. The **device** number varies depending upon the functions that are to be accessed. Number 8 permits binary I/O, 9 permits analog I/O, and 10 gains access to the 16-bit counter. The **hndshk** variable is reserved and must be a 0. The data variable, v, must contain an integer or integer array. The variable **stat** is initially set to 0. A nonzero value returned indicates a failure of the function. Good programmers would use this feature to report and recover from conversion problems.

When Lattice C is used, the large C model (CL) must be used when compiling. The small C model is the default, so it is necessary for the user to specify the large model when invoking the C compiler and linker.

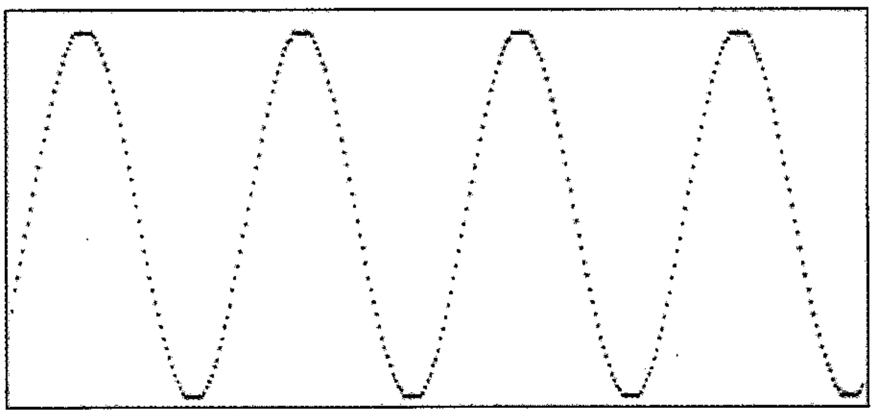
Suppose that a user wants to monitor two of the four available analog inputs. This is called *scanning* on the adapter card. One input will be from a solar cell with a maximum output voltage of +0.5 volts, and the other will be from a variable power supply with a range of 0 to +10 volts. Apart from scanning two inputs, the interface software will not be especially complicated.

FIGURE 2: *60-Hz Sine Wave*



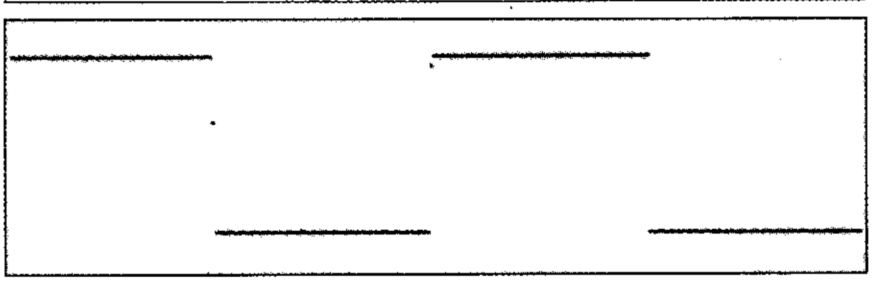
These are the results of using an A/D channel to track a 60-Hz sine wave. The program took 320 samples at 19,200 samples/second.

FIGURE 3: *240-Hz Sine Wave*



The tracking rate was 76,800 samples/second. The four wave forms indicate that 19,200 samples/second is about the maximum sampling rate.

FIGURE 4: *120-Hz Square Wave*



These are the results of tracking a 120-Hz square wave. Note the two data points that were captured on the rise and fall of the square wave.

Listing 2 is a FORTRAN program that will print the voltage values of each input channel to the screen.

There is a catch in the interfacing of the input devices. The power supply is well matched to the adapter's input (0 to +10 volts), but the solar cell can output only +0.5 volts. Without an "on-board" instrumentation amplifier, the resolution of the readings will suffer if the solar cell is connected directly to the adapter. Figure 1 shows a simple operational amplifier circuit that will amplify the +0.5 volts of the solar cell to +10.0 volts. In this circuit, the common 741 op-amp was chosen because of noncritical sampling conditions. This same op-amp circuit could be used for expanding the input-voltage swing of other low-voltage transducers.

The program in listing 2 makes use of the AINSC call (Analog Input Scan) for monitoring up to four analog inputs. CHANLO is the first channel to be sampled and CHANHI is the last. Thus channels 0 and 1 are being monitored. The variable CTRL refers to the expansion-device control. If only one adapter is in place, CTRL typically will be 0. MODE and STOR must be 0 for AINSC. COUNT is the number of times the function will execute; for example, the function could be performed 100 times, filling an array with data samples, before the call is exited. The example in listing 2 takes one sample of two channels and places their values in an array, V(I).

Once this pair of values is written to the screen, the program uses a GOTO to get another reading using the call AINSC. Notice that the data in V(I) are conditioned before being reported to the screen as VOLTS. The A/D converter is a 12-bit device capable of 2^{12}=4,096 unique readings. These unique readings or numbers must be scaled to the input parameters. This line in FORTRAN performs the scaling task:

```
VOLTS(I)=(V(i)/204.8)-10
```

In this case, the adapter was set (with switches) for −10 to +10 volts. The input data has a range of 0 to +10.0 volts. To scale the 4,096 possible readings across the adapter's full range of 20 volts, the program must divide 4,096 by some number that yields 20. This number is the scaling factor, 204.8. Subtracting 10 from the result puts the 0-volts point in the middle of the adapter's range rather than at the bottom. The formatting is set to give two digits of decimal precision. Remember that the resolution is .0025 volts over a 10-volt range. This program works with the IBM Adapter to turn the computer into a two-channel voltmeter.

The A/D converter on the IBM Adapter board is fast compared to many for the XT. The specifications say that up to 16,000 samples per second may be made. When one channel of the converter is connected to a 6.3-volt filament transformer, the peak-to-peak voltage for 6.3 volts RMS is −8.91 to +8.91 volts. If the converter were fast, it would have no trouble tracking and sampling a 60-Hz sine wave.

Because one whole sine wave is executed in 1/60=.016667 seconds and 320 points must be plotted on the medium-resolution screen, readings must be taken every .016667/320=.00005208 seconds. The reciprocal of .00005208 yields a sampling rate of 19,200 Hz (samples/second), which should be an interesting test of the 16,000-Hz limit.

The compiled BASIC program in listing 3 allows the user to set the sam-

TABLE 1: *Interfacing Functions*

FUNCTION	PURPOSE
Analog Input Multiple (AINM)	Perform an iterative analog input COUNT times
Analog Input Simple (AINS)	Perform a single analog input operation
Analog Input Scan (AINSC)	Accept input consecutively from adjacent channels
Analog Output Multiple (AOUM)	Perform an iterative analog output COUNT times
Analog Output Simple (AOUS)	Perform a single analog output operation
Binary Input Multiple (BINM)	Perform an iterative 16-bit binary input COUNT times
Binary Input Simple (BINS)	Perform a single 16-bit binary input operation
Binary Bit Input Simple (BITINS)	Input state of bit # BIT from binary input port
Binary Bit Output Simple (BITOUS)	Set the state of bit # BIT in binary output word
Binary Output Multiple (BOUM)	Iteratively output 16-bit binary words COUNT times
Binary Output Simple (BOUS)	Output a single 16-bit binary word
Counter Input Multiple (CINM)	Iteratively input counter value COUNT times
Counter Input Simple (CINS)	Perform a single read of counter value
Counter Set (CSET)	Initialize counter value
Delay Execution (DELAY)	Software delay for sampling interval > 1 second

These 15 functions, which are used for interfacing software to the adapter, are all available from BASIC, FORTRAN, and C.

pling rate each time the program is run; it displays on the screen the important parameters that are being passed to AINM; it also permits the user to save the screen plot to a diskette in drive B:. Listing 4 is an additional program that lets the screen image be loaded from interpreted BASICA.

Comprehension of this program requires an understanding of the function AINM. Many of the variables that are passed to AINM have already been discussed in connection with listings 1 and 2. MODE controls the system's interrupts and may take on one of two values: if MODE=0, normal system interrupts are possible; if MODE=128, the device driver inhibits system interrupts, thereby increasing I/O performance. (In this example, MODE=128). COUNT is set to 320, because 320 samples must be taken during the pass over the sine wave. The values that are returned will be placed in an array, V%. A unique plot will be obtained each time the program is run, because the sine wave will be "caught" at a different location when the sampling begins. The compiled BASIC program will plot all 320 data points on the medium-resolution screen as white points on a blue background. Figure 2 is a plot of a 60-Hz sine wave from the 6.3-volt filament transformer.

This was the first time I had ever attempted such high sampling rates from an A/D converter on a small system; I hooked up a Heathkit Sine/Square generator and increased the frequency of the sine wave to 240 Hz and the sampling rate to 76,800.

As the results in figure 3 show, something went wrong. Because I increased both the frequency and the sampling rate by a factor of four, there still should have been only one cycle of a sine wave on the screen. Instead, four full sine waves were plotted: apparently, 19,200 samples/second is (or is close to) the upper sampling limit. Increasing the number entered for the sampling rate will not change what is physically impossible. With the sampling rate nearly unchanged and the frequency increasing by a factor of four, the program plotted approximately four complete sine waves.

In another experiment the square-wave generator was hooked to the A/D converter and the frequency was set to 120 Hz and the sampling rate to 19,200. Figure 4 is a plot of the data gathered. These examples show that although the A/D converter is no match for a quality oscilloscope, it does give experimenters a new tool for monitoring and recording relatively low frequency events.

A number of data acquisition boards with many different options are on the market for the IBM PC/XT and PC/AT. The IBM Data Acquisition Adapter does not come loaded with accessories, such as built-in instrument amplifiers, thermometer probes, etc., but it would be a good choice because of its flexibility and quality of workmanship.

The manual is a big plus for the IBM Adapter. It serves not only as a user's guide, but also as an excellent source of sample programs. It takes only minutes to go "from box to data sample" with the IBM Adapter.

The only drawback of the adapter and its supporting software is that assembly language is not supported as a programming option. Furthermore, IBM does not give enough information on the adapter's hardware port assignments or low-level details of the device-driver calling syntax to enable even an experienced assembly language programmer to create his own assembly language calling library. Nonetheless, the three supported languages are versatile and capable enough to make the IBM Adapter a powerful tool for laboratory data acquisition.

IBM Data Acquisition and Control Adapter: $1,275; Distribution Panel: $245
Programming Support: $160
IBM Corporation
5201 Congress
Boca Raton, FL 33431
Contact local IBM dealer
CIRCLE 351 ON READER SERVICE CARD

William Murray teaches computer science at Broome Community College located in Binghamton, New York.

LISTING 1: ONEREAD.C

```
/* Program continuously reads 16 bit binary input */
#include "a:stdio.h"
main()
(
int adapt, device, hndshk, v, stat, answer;
adapt=0;
device=8;
hndshk=0;
stat=0;
repeat: bins (adapt, device, hndshk, &v, &stat);
answer=abs(v)-1;
printf(" %d\n",answer);
goto repeat;
)
```

LISTING 2: TWOREAD.FOR

```
C       FORTRAN PROGRAM WILL SCAN ANALOG INPUTS 0 AND 1
C       THE RATE WILL BE 500 SCANS/SECOND
C       PROGRAM WILL OUTPUT DATA TO SCREEN UNTIL INTERRUPTED
        REAL*4 VOLTS(2)
        INTEGER*2 V(2)
        INTEGER*2 ADAPT,DEVICE,CHANLO,CHANHI,CTRL,
     *  MODE,STOR,STAT
        INTEGER*4 COUNT,RATE
        ADAPT=0
        DEVICE=9
        CHANLO=0
        CHANHI=1
        CTRL=0
        MODE=0
        STOR=0
        COUNT=1
        RATE=1000
        STAT=0
100     CALL AINSC (ADAPT,DEVICE,CHANLO,CHANHI,CTRL,
     *  MODE,STOR,COUNT,RATE,V(1),STAT)
        DO 110 I=1,2
        VOLTS(I)=(V(I)/204.8)-10
110     CONTINUE
        WRITE (*,130)VOLTS
130     FORMAT (1X,F10.2,F10.2)
        GOTO 100
        END
```

LISTING 3: DAPLOT.BAS

```
CLS
PRINT "THIS PROGRAM"
PRINT "PLOTS MULTIPLE DATA POINTS GATHERED BY A/D CONVERTER"
PRINT
PRINT "ENTER THE SAMPLING RATE (samples/second): ":INPUT RATE
CLS
PRINT "THESE ARE THE PARAMETERS WHICH ARE SET FOR THE A/D CARD"
PRINT
ADAPT%=0
DEVICE%=9
MODE%=128
STOR%=0
STAT%=0
CHANLO%=1
CTRL%=0
COUNT=320
DIM V%(319)
PRINT "USING ADAPT NUMBER: ";ADAPT%
PRINT "WITH DEVICE NUMBER: ";DEVICE%
PRINT "EXPANSION DEVICE CONTROL NUMBER: ";CTRL%
PRINT "EXECUTION MODE: ";MODE%
PRINT "CHANNEL ACCESSED: ";CHANLO%
PRINT "NUMBER OF SAMPLES: ";COUNT
PRINT "RATE OF SAMPLING (SAMPLES/SECOND): ";RATE
PRINT
PRINT
PRINT "Shall I save this plot? (type 1 for YES, 2 for NO)"
INPUT TX
IF TX=1 THEN PRINT "ENTER FILE NAME (NO EXTENSIONS)":INPUT P$
PRINT
PRINT "once the plot is complete, hit any key to save"
PRINT
PRINT
PRINT "NOW STRIKE A KEY TO TAKE SAMPLES"
INPUT SS$
CLS
CALL AINM (APAPT%,DEVICE%,CHANLO%,CTRL%,MODE%,
           STOR%,COUNT,RATE,V%(0),STAT%)
SCREEN 1
COLOR 1,7,7
FOR I=0 TO 319
Z=-(V%(I)*6/204.8)+150
PSET(I,Z),14
NEXT I
INPUT SS$
IF TX=2 THEN GOTO 1000
DEF SEG=&HB800
P1$="B:
P2$="
PP$=P1$+P$+P2$
BSAVE PP$,0,&H4000
1000 END
```

LISTING 4: PLOTLOAD.BAS

```
10 CLS
20 PRINT "PROGRAM LOADS A SCREEN IMAGE SAVED BY A PREVIOUS PROGRAM"
30 PRINT
40 PRINT "THIS PROGRAM ASSUMES THAT THE PICTURE IS IN THE B DRIVE"
50 PRINT
60 PRINT "ENTER THE NAME OF THE PICTURE TO BE LOADED (NO EXTENSIONS)
70 INPUT P$
80 CLS
90 SCREEN 1
100 COLOR 1,7,7
110 KEY OFF
120 DEF SEG=&HB800
130 P1$="B:
140 P3$="
150 P2$=P1$+P$+P3$
160 BLOAD P2$,0
170 REM PUSH THE F1 KEY TO RETURN TO BASIC
180 ON KEY(1) GOSUB 210
190 KEY(1) ON
200 GOTO 190
210 END
```