NOTE

Interfacing the PET/CBM with Coulbourn's DynaPort for experimental control and data acquisition

JOSEPH D. ALLEN and BARRY NICORA
University of Georgia, Athens, Georgia

Component changes to Coulbourn's DynaPort that are necessary to ensure reliable data exchange with laboratory computers are discussed. Also provided and explained is a data communications package of BASIC and assembly language routines that will permit the PET/CBM to retrieve data from the DynaPort.

In 1980, Coulbourn Instruments, Inc., provided an interface between their solid-state modular logic units and the all-purpose laboratory controller, the microcomputer. This interface, the DynaPort, comes equipped with either an RS-232-C serial (Model S18-12) or an IEEE parallel port (Model S18-13), and may be linked to most laboratory computers in current use (Coulbourn Instruments, Inc., 1982). As stated in the manual (Coulbourn Instruments, Inc., 1982, p. 25), "The S18-13 Model DynaPort employs an IEEE-488-1978 standard interface to the computer." Such a configuration is particularly useful for researchers who use Commodore computers (PET/CBM 2001, 4032, 8032, 9000SP) as laboratory controllers and data collectors. The IEEE-488 port is standard equipment on these computers, and on models equipped with 2.0 BASIC and above, the IEEE bus meets 1978 specifications. Thus, interfacing these units should be a relatively easy task. However, for those who already possess a DynaPort, a few hardware problems must be corrected before reliable interfacing can be achieved with any microcomputer system.

HARDWARE MODIFICATIONS

The National Semiconductor Corporation 8070 CPU used in DynaPort contained a bug such that interrupts arising from its I/O ports were ignored at random times in early units. In general, the frequency with which the CPU "hangs up" appears to be directly proportional to the number of I/O ports in active service at the time (the DynaPort can service up to 10 ports). The interrupt problem is inherent with the DynaPort, and thus no microcomputer is immune to it. Powering down and then up appears to be the only way to reset the chip. In March 1984, National Semiconductor issued the replacement 8070N,

Joseph D. Allen is with the Department of Psychology, and Barry Nicora is with the Electronics Design and Maintenance Shop, both at the University of Georgia, Athens, GA 30602.

which corrects this problem, but which necessitates the addition of a new oscillator and clock circuit to the CPU card and a new EPROM to the memory card. (These cards may be sent to Coulbourn Instruments for upgrade components.)

The IEEE 488 I/O card should contain the new 8291A interface chip by Intel. The system used in our laboratory contained the old 8291 version. For those who already have the 8291A installed, the EPROM, which is located on the DynaPort's memory card with routines to drive this chip, must be upgraded from the S1813E (10/4/83) version to the more recent S1813F (3/5/84). This upgrade is necessitated by the change in CPU.

DATA TRANSFER TO THE PET/CBM

With these hardware alterations on the DynaPort, two-way data communication between the PET/CBM and Coulbourn's modular logic modules via the DynaPort is fairly straightforward, although not thoroughly documented in Coulbourn's manual. Data arriving at each I/O buffer module from Coulbourn's modules (or any 5-12-V modular equivalent) may be latched into its 8-bit input port by pulsing (4 μ sec) its input strobe. A strobe pulse from each active I/O port interrupts the DynaPort's CPU, which transfers the 8-bit data word to a 1K FIFO buffer, where it is stored until accessed by the PET via a cable connecting the PET'S IEEE port to that of the DynaPort.

A workable data-retrieval procedure for the PET is best described by following the BASIC program presented in Figure 1. Line 20 labels the calling addresses of several machine language routines that were written and located in the first cassette buffer of the PET/CBM. The Serial Poll and Device Clear routines to which these labels refer are universal bus commands that are required in the data-polling sequence but are not implemented by PET BASIC.

```
10 : REM *** WORKING DATA FETCH PROGRAM ***
15 :
20 SP=634:DC=731:POLL=237:SRQ=59427:CLRFLG=59426
30 PRINTCHR$(147):TIS="000000": REM CLR SCREEN , ZERO CLOCK
40 OPEN 1,10
50 SYS(DC): REM CLR DYNAPORT
60 A=PEEK(CLRFLG): REM CLR FLAGS
70 IF(PEEK(SRQ)AND128)=0 THEN 70: REM KEEP POLLING SRQ
90 A=PEEK(CLRFLG): REM CLR SRQ FLAG
100 SYS(SP): REM SERIAL POLL
115 FORI=1TO4: REM GET CH#,DA,CR,LF
120 : GET#1,DA$(I):IFDA$(I)=""THENDAS(I)=CHR$(0)
125 NEXTI
130 FORI=1TO2: PRINT ASC(DAS(I)): NEXTI: PRINT
140 GOTO70 REM POLL SRQ AGAIN
READY.
```

Figure 1. BASIC program illustrating the use of Service Request and Serial Polling techniques to access data from the DynaPort and display it on the PET/CBM screen. (Note: Serial Poll assembly listing is in Figure 2.)

END OF MAE PASS:

```

0100 *****
0110 * SERIAL POLL AND DEVICE CLEAR *
0120 * FOR DYNAPORT *
0130 * BY J D ALLEN *
0135 * APRIL 20, 1984 *
0140 *****
0150 ;
0160 *****
0170 * ASSEMBLER FORMAT *
0180 *****
0190 ;
0200 .BA $027A
0210 .CE
0220 .OS
0230 .LS
0240 ;
0250 *****
0260 * SYSTEM ADDRESSES *
0270 *****
0280 ;
0290 HANDVIA .DE $E840
0300 HANDPIA .DE $E821
0310 INPUT .DE $F1C0
0320 OUTPUT .DE $F109
0330 STORE .DE $F1
0340 OUTBUF .DE $A5
0350 STATUS .DE $D0
0360 ;
0370 *****
0380 * SERIAL POLL ROUTINE *
0390 *****
0400 ;
027A- AD 40 E8 0410 LDA HANDVIA ;SET ATN TRUE
027D- 29 FB 0420 AND $F8
027F- 8D 40 E8 0430 STA HANDVIA
0282- A9 4A 0440 LDA $54A ;SEND MY TALK ADDR
0284- 85 A5 0450 STA *OUTBUF
0286- 28 09 F1 0460 JSR OUTPUT
0289- A9 18 0470 LDA $518 ;SEND SERIAL POLL ENABLE
028B- 85 A5 0480 STA *OUTBUF
028D- 28 09 F1 0490 JSR OUTPUT
0290- AD 40 E8 0500 LDA HANDVIA ;SET NRFD TRUE
0293- 29 FD 0510 AND $5FD
0295- 8D 40 E8 0520 STA HANDVIA
0298- AD 21 E8 0530 LDA HANDPIA ;SET NDAC TRUE
029B- 29 F7 0540 AND $5F7
029D- 8D 21 E8 0550 STA HANDPIA
02A0- AD 40 E8 0560 LDA HANDVIA ;SET ATN FALSE
02A3- 09 04 0570 ORA $504
02A5- 8D 40 E8 0580 STA HANDVIA
02A8- 28 C0 F1 0590 JSR INPUT ;GET STATUS FROM DYNA
02AB- 85 ED 0600 STA *STATUS ;STORE IT
02AD- AD 40 E8 0610 LDA HANDVIA ;SET ATN TRUE
02B0- 29 FB 0620 AND $F8
02B2- 8D 40 E8 0630 STA HANDVIA
02B5- AD 40 E8 0640 LDA HANDVIA
02B8- 09 02 0650 ORA $502 ;SET NRFD FALSE
02BA- 8D 40 E8 0660 STA HANDVIA
02BD- AD 21 E8 0670 LDA HANDPIA ;SET NDAC FALSE
02C0- 09 08 0680 ORA $508
02C2- 8D 21 E8 0690 STA HANDPIA
02C5- A9 19 0700 LDA $519 ;SEND SERIAL POLL DISABLE
02C7- 85 A5 0710 STA *OUTBUF
02C9- 28 09 F1 0720 JSR OUTPUT
02CC- A9 5F 0730 LDA $55F ; SEND UNTALK
02CE- 85 A5 0740 STA *OUTBUF
02D0- 28 09 F1 0750 JSR OUTPUT
02D3- A9 3F 0760 LDA $53F ; SEND UNLISTEN
02D5- 85 A5 0770 STA *OUTBUF
02D7- 28 09 F1 0780 JSR OUTPUT
02DA- 60 0790 RTS
0800 ;
0810 *****
0820 * DEVICE CLEAR ROUTINE *
0830 *****
0840 ;
02DB- AD 40 E8 0850 LDA HANDVIA ;SET ATN TRUE
02DE- 29 FB 0860 AND $F8
02E0- 8D 40 E8 0870 STA HANDVIA
02E3- A9 20 0880 LDA $520 ;SEND MY LISTEN ADDR
02E5- 85 A5 0890 STA *OUTBUF
02E7- 28 09 F1 0900 JSR OUTPUT
02EA- A9 14 0910 LDA $514 ;SEND DEVICE CLEAR
02EC- 85 A5 0920 STA *OUTBUF
02EE- 28 09 F1 0930 JSR OUTPUT
02F1- AD 40 E8 0940 LDA HANDVIA ;SET ATN FALSE
02F4- 09 04 0950 ORA $504
02F6- 8D 40 E8 0960 STA HANDVIA
02F9- 60 0970 RTS
0980 .LN

```

END OF MAE PASS:

```

--- LABEL FILE: ---
HANDPIA =E821      HANDVIA =E840      INPUT =F1C0
OUTBUF =00A5      OUTPUT =F109      STATUS =00ED
STORE =00F1
//0000,02FA,02FA
}

```

Figure 2. Assembly listings for Serial Poll and Device Clear routines that load into the first cassette buffer of the PET/CBM. These routines are, however, fully relocatable.

Documented assembler listings for these routines appear in Figure 2. Line 40 opens a communication file (#1) to the DynaPort, which is set at the factory to device #10. The unit's Device No. may be altered by setting the five lines, s4-s8, of an 8-bit DIP switch on the interface card. The switch is read in binary, with s8 as its least significant bit. S3 switches between binary and ASCII data for-

mat, and must be set to "0" for binary data. Lines 50-60 do some preliminary housecleaning. Line 50 SYSs to the Device Clear routine (Figure 2), which resets the DynaPort's data pointers and effectively clears its buffers to "power up" state. Line 60 reads the output buffer for the IEEE bus on the PET's PIA #2 and effectively clears any flags set prior to waiting for the DynaPort to request service. The main routine begins with a polling loop in line 70, which continuously reads bit 6 of the PIA's control register for a service request (SRQ) generated by the DynaPort whenever there is data in its buffer. (SRQ polling was chosen in preference to the Parallel Poll command, which was found to be difficult to implement with the DynaPort.) An SRQ sets bit 6 high on the control register, where it remains until cleared by reading the PIA's output port. This is again done in line 90. After an SRQ is polled and cleared, the PET must next issue a Serial Poll command to the DynaPort. This is accomplished in line 100. The Serial Poll routine (Figure 2) returns the status of the poll to location 237, where it may be peeked and tested if error checking is desired. A proper polling sequence, however, inevitably returns a value of 65, signifying that the DynaPort's SRQ bit is set and there is a single data sample to be read. The data is then obtained with a sequence of four GET#s in the loop from lines 115 to 125. In the single-sample mode, all data are stored in the DynaPort buffer as the following 2-byte series: (1) I/O Channel #, (2) data. Following a serial poll, each data sample is transmitted on the bus as a 4-byte sequence with carriage-return and line-feed characters appended. Each byte must be tested for the illegal null value and substituted with CHR\$(0) if found. The CR and LF characters must be obtained but, of course, are not printed to the screen along with the ASCII values of the Channel # and data sample in line 130. A more useful routine, of course, would assign these values to subscripted variable locations for later storage. Line 140 returns to the SRQ polling routine and awaits more data from the DynaPort.

A word of caution if a Model 4040 disk drive also resides on the bus: The drive should be turned off when the PET and DynaPort are "talking." An undocumented bug in the DOS (Southern Bell Telephone Co., personal communication, December 1983) may cause it to interact periodically with communications on the bus and send spurious commands that can hang up the DynaPort. No problems are experienced when the drive is turned on only for file-storage purposes after data collection is completed.

Using a program written in BASIC, such as that described above, it is possible to poll four I/O buffers at a time and read data at rates between 10 and 12 samples per second. If desired, much higher data-sampling rates are obtainable with assembly language routines using interrupts. The PET's handshaking routines are fully adequate for data exchange and may be accessed directly from the following locations in 4.0 ROM models: \$F1C0 = SHKIN, \$F109 = SHKOUT, (2.0 ROMs: \$F18C, \$F0EE). Also, the DynaPort may be configured to send

a continuous stream of data from one channel at a time using the High Speed Sample Mode. This method eliminates the need for requesting each data sample with an SRQ and Serial Poll routine, and the data from the selected channel are read in a burst without appended CR and LF characters.

DATA TRANSFER TO THE DYNAPORT

Sending data to the DynaPort is very straightforward. After a file# to the bus (i.e., 3) is opened, data may be sent to the DynaPort with the following instruction: `PRINT#3,CHR$(2);CHR$(23);`. Notice that the data are converted to character strings (binary) and sent as a couplet representing first the destination I/O port# (2) and second the data byte (23). Semicolons are used after each character to suppress CRs. Data placed on the bus are latched onto the 8-bit output port of the specified I/O module and a 4- μ sec pulse is simultaneously generated by the port's output strobe, which may be used to operate TTL logic modules.

SUMMARY

This article provides interfacing solutions that are peculiar to the PET/CBM's communication with the DynaPort. For a fuller description of interfacing the PET/CBM's IEEE-488 and parallel ports to laboratory equipment, the reader is referred to Downey and Rogers (1981), Fisher and Jensen (1982), Hampshire (1980), and West (1982).

REFERENCES

- COULBOURN INSTRUMENTS, INC. (1982). *DynaPort: Computer interface*. Lehigh Valley, PA: Author.
- DOWNEY, J. M., & ROGERS, S. M. (1981). *PET interfacing*. Indianapolis: Howard Sams.
- FISHER, E., & JENSEN, C. W. (1982). *PET/CBM and the IEEE-488 bus (GPIB)* (2nd ed.). Berkeley, CA: Osborne/McGraw Hill.
- HAMPSHIRE, N. (1980). *The PET revealed* (2nd ed.). Somerset, England: Computabits Ltd.
- WEST, R. C. (1982). *Programming the PET/CBM*. Greensboro, NC: COMPUTE! Books.

(Revision accepted for publication September 19, 1984.)