A MICROCOMPUTER INTERFACE FOR A DIGITAL AUDIO PROCESSOR-BASED DATA RECORDING SYSTEM

Thomas L. Croxton, Stanley J. Stump, and William McD. Armstrong
Department of Physiology and Biophysics, Indiana University School of Medicine, Indianapolis, Indiana 46223

ABSTRACT An inexpensive interface is described that performs direct transfer of digitized data from the digital audio processor and video cassette recorder based data acquisition system designed by Bezanilla (1985, *Biophys. J.*, 47:437–441) to an IBM PC/XT microcomputer. The FORTRAN callable software that drives this interface is capable of controlling the video cassette recorder and starting data collection immediately after recognition of a segment of previously collected data. This permits piecewise analysis of long intervals of data that would otherwise exceed the memory capability of the microcomputer.

The high capacity data recording system designed by F. Bezanilla (1985) is of great value to practitioners of single-channel recording since it allows storage of two channels of very precise data (16 bit resolution), sampled at an acceptably fast rate (44,100 samples/s), for extended periods of time (up to 3 h). We have, however, encountered technical problems which limit the computerized analysis of data recorded in this way. In particular, when the data are played back it is difficult to accurately or reproducibly select a given interval of data for storage in computer memory. This difficulty prompted us to develop an interface between Bezanilla's data recording system and the model PC/XT microcomputer (IBM Instruments Inc., IBM Corp., Danbury, CT). The interface described herein is capable of retrieving the same interval of data repeatedly and of retrieving contiguous intervals. The latter capability allows analysis of single channel records which greatly exceed in duration the data storage capacity of the computer (<6 s data per 512 kbytes memory).

Our interface consists of a switch box that selects the left or right channel for sampling, a plug-in computer board that multiplexes the 16-bit data for transfer on the 8-bit computer data bus, and an assembly language subroutine that controls the video cassette recorder (VCR) and directs storage of data in sequential memory locations. Two functional modes are available. In the interactive mode the VCR is started and the data array is filled repetitively until a key is pressed. The video tape is then rewound to a position ~20 s before the beginning of the collected data. In the automatic mode the calling program provides a marker of 42 sequential, previously sampled data points. The VCR is started and incoming data are compared with a portion of this marker. If a match is found the data array is filled with the data that immediately follow the marker. The video tape is then rewound to a position ~20 s before the beginning of the data and control returns to the calling program. If no match is found within an adjustable period of playback (~1 min to 0.5 h), the video tape is rewound ~60 s and the subroutine returns a flag which indicates that data collection was not successful. If the last 42 data points of each record are used as the marker for the next record, sequential subroutine calls in the automatic mode will yield retrieval of contiguous blocks of data.

Although the interface we describe is hardware specific to our instruments and computer, adaptation to other equipment should be straightforward. However, speed limitations of the 8255 interface chip appear to preclude use of this particular interface with IBM AT-type computers that operate at a 6–8 MHz clock speed. Our interface hardware is depicted in Fig. 1. Total cost for the computer board and additional components is ~$150. The digital output stage designed by Bezanilla provides word clocks for the right and left data channels and 16 parallel data lines. In our digital audio processor (DASS 501; Unitrade Inc., Philadelphia, PA) these signals are available on a 25-pin connector at the rear of the chassis. A ribbon cable extension is used to connect these lines to a similar connector mounted on one side of a 5" × 2.5" × 1.5" plastic box. Point-to-point wiring within the box connects the data lines to a 37-pin connector mounted on the opposite side of the box, connects the clock lines to a channel selector switch, and routes control lines for the VCR from the 37-pin connector to a 5-pin hex connector. A second ribbon cable extension connects the 37-pin connector of the switch box to the matching connector of a plug-in computer board.

The computer board we now employ (model PIO12,

This work was supported by U.S. Public Health Service grants DK36575 and DK07554.

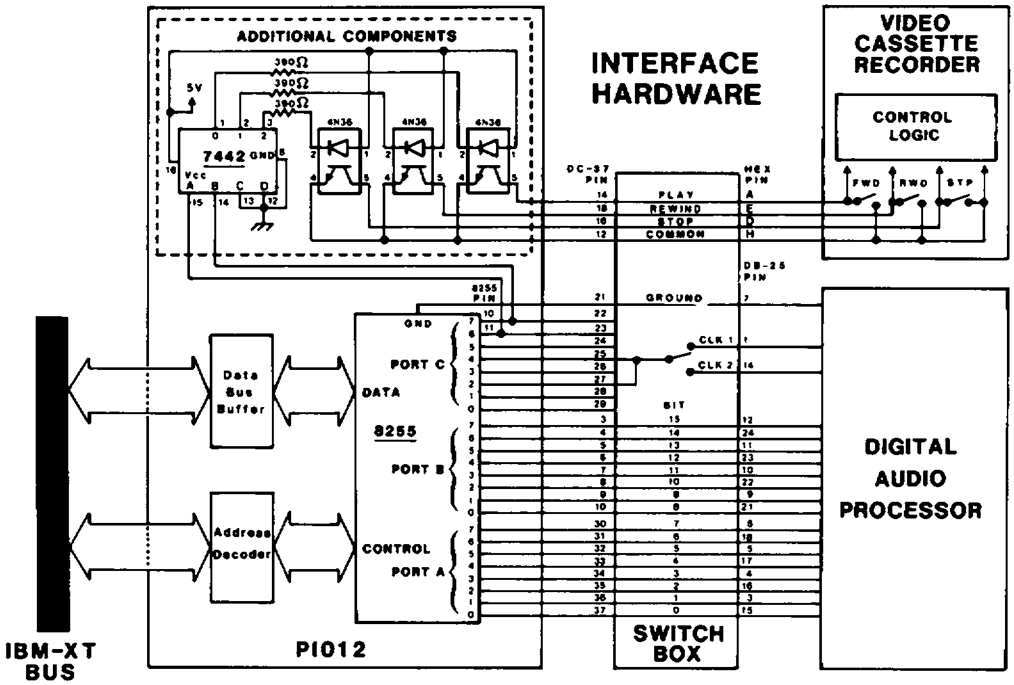

FIGURE 1 Electronic circuitry of digital interface. The parallel digital output of the digital audio processor is routed to ports A and B of an 8255 programmable peripheral interface chip on a PIO12 computer board. The 8255 is programmed to convert each 16-bit data word to two 8-bit words for transfer on the computer bus. Wiring between a DB-25 connector and a DC-37 connector in the switch box facilitates connection of the digital audio processor to the PIO12 computer card. Choice of digital audio processor data channel is made by switch selection of the clock signal to be connected to handshaking inputs of the 8255 port C. Components added to the PIO12 card activate the play, rewind, and stop functions of the video cassette recorder via optoisolators under the control of bits 6 and 7 of port C.

MetraByte Corp., Taunton, MA 02780) was modified to allow computerized control of the VCR. This board contains an 8255 programmable peripheral interface chip (Intel Corp., Santa Clara, CA), a data bus buffer, and an address decoder. An unused area at the top of the circuit board provides ample room for wire wrap integrated circuit sockets which house the additional components shown in Fig. 1. Connections from the added components to the existing board may be made with short jumpers soldered to the circuit side of the board. The only other modification required is the removal of power supply voltages from pins 12, 14, 16, and 18 of the 37-pin connector so that these lines may be used for VCR control. The -5 V, -12 V, and $+12$ V lines are not required and are best interrupted by cutting three circuit board traces near their contacts with the computer bus. Removal of $+5$ V from pin 18 requires cutting of two wide traces which approach the top and bottom of the 37-pin connector on the component side of the board and installation of a jumper to reconnect these two $+5$ V traces.

The play, stop, and rewind functions of our VCR (model SL-HF450; Sony Corp. of America, Long Island City, NY) are normally triggered by depressing SPST switches located on the front panel. Modification of this VCR consists simply of mounting a 5-pin hex connector to the back panel and connecting pins to ground and to the ungrounded sides of the play, stop, and rewind switches. These lines are connected, via the switch box, to three 4N36 optoisolators mounted on the PIO12 circuit board. Activating one of the optoisolators is equivalent to depressing the corresponding switch of the VCR.

The software that drives the data transfer is written for the Microsoft MACRO ASSEMBLER and is intended to be linked with a Microsoft FORTRAN calling program. A brief overview of the algorithm follows. Details of the algorithm and instructions for calling the subroutine are provided in the remarks of the source code listed in Fig. 2. The 8255 is programmed for strobed input of ports A and B. In this configuration, bits 0–5 of port C are used for handshaking. The data word is latched into ports A and B by directing the selected clock signal to bits 2 and 4 of port C. The microcomputer samples bit 0 of port C (INTR B) to determine when the data word has been latched. The word is then input to a microcomputer register and is stored in memory. Memory is addressed in paragraphs of 8 points (16 bytes) each. Between each paragraph the segment register is incremented and the offset register is reduced by 16. Data collection ceases when the required number of paragraphs of data has been stored. Detection of keyboard entry is made via function calls to the disk operating system (Microsoft DOS).

In the automatic mode the program is provided with a marker consisting of 42 previously sampled points. A 32-bit template is constructed from the least significant

```
; 8088 ASSEMBLER SUBROUTINE TO COLLECT DATA FROM AN 8255
; DIGITAL INTERFACE.  FORTRAN CALLING PROGRAM CONTAINS
;
;       COMMON/DAT/ LEADER(8),DATA(XXX)
;       DIMENSION MARKER(42)
;       INTEGER*2 LEADER,FLAG,DATA,N,MARKER,ADDR
;
; AND CALLS THE SUBROUTINE WITH
;
;       CALL VCR(FLAG,DATA(1),N,MARKER(1),ADDR)
;
; FLAG    - DEFINES MODE OF OPERATION:
;             MANUAL - FLAG=0.  DATA COLLECTION REPEATED
;                  UNTIL KEY PRESSED.
;             AUTOMATIC - FLAG=1-27.  DATA COLLECTION BEGINS
;                  WHEN DATA MATCHES MARKER.  IF TOTAL MATCH
;                  NOT FOUND BEFORE 40*FLAG HALF MATCHES,
;                  (ABOUT 1-27 MINUTES) SUBROUTINE ABORTS.
;         - RETURNS WITH STATUS CODE:
;             FAILED - FLAG=0.
;             SUCCESSFUL - FLAG=1.
; DATA    - ARRAY THAT RECEIVES COLLECTED DATA.
; N       - NUMBER OF POINTS TO BE COLLECTED / 16.
; MARKER  - 42 PREVIOUSLY COLLECTED DATA POINTS REQUIRED
;           FOR AUTOMATIC MODE.
; ADDR    - ADDRESS OF 8255 PORT A.
; LEADER  - RETURNS FROM AUTOMATIC CALL WITH 8 DATA POINTS
;           IMMEDIATELY PRECEDING DATA(1).  THESE SHOULD
;           EQUAL MARKER(35)-MARKER(42).
; XXX     - DIMENSION LIMITED BY AVAILABLE RAM.
;
;VERSION 12/8/86 BY TOM CROXTON, DEPARTMENT OF PHYSIOLOGY
;AND BIOPHYSICS, INDIANA UNIVERSITY SCHOOL OF MEDICINE,
;635 BARNHILL DRIVE, INDIANAPOLIS, IN 46223 (317) 274-1443

DATA SEGMENT PUBLIC 'DATA'
DUMMY DW 0
FUNC DW 0
DATA ENDS
DGROUP GROUP DATA

VCODE SEGMENT 'CODE'
ASSUME CS:VCODE,DS:DATA,ES:DATA

TIMER PROC FAR          ;DELAY OF 40*AX USEC
 PUSH BX                ;SAVE REGISTERS
 PUSH CX
 PUSH DX
 MOV CX,AX
STALL: MOV AX,1
 MOV BX,1
 CWD
 IDIV BX
 LOOP STALL
 POP DX                 ;RESTORE ORIGINAL REGISTERS
 POP CX
 POP BX
 RET                    ;RETURN
TIMER ENDP

 PUBLIC VCR
VCR PROC FAR            ;CONTROLS VCR AND READS DATA
 PUSH BP
 MOV BP,SP              ;POINT BP TO STACK
 PUSH DS                ;SAVE REGISTERS
 PUSH ES
 CLI                    ;DISABLE INTERRUPTS
 CLD
 LDS BX,DWORD PTR[BP+22] ;GET FUNCTION FLAG
 MOV AX,[BX]
 MOV FUNC,AX
 MOV AX,1
 MOV [BX],AX            ;SET FLAG=1
 CMP FUNC,0             ;TEST FUNCTION FLAG
 JE BEGIN               ;IF 0 JUMP TO BEGIN
 LDS BX,DWORD PTR[BP+10] ;GET MARKER ADDRESS
 MOV CX,20H
MARK: MOV AX,[BX]       ;READ MARKER
 SHR AX,1
 RCL DI,1               ;ROTATE CODE INTO DI,SI
 RCL SI,1
 ADD BX,02H
 LOOP MARK
BEGIN: LDS BX,DWORD PTR[BP+14] ;GET #POINTS/16
 MOV DX,[BX]
 LES BX,DWORD PTR[BP+18] ;GET DATA ADDRESS
 MOV CL,40
 SHR BX,CL
 MOV AX,ES              ;ADJUST ES TO POINT TO
 ADD AX,BX              ;LEADER(1) WITH OFFSET 10H
 SUB AX,02H
 MOV ES,AX
 PUSH ES                ;SAVE SEGMENT REGISTER
 MOV CX,DX
 SHL CX,1
 INC CX                 ;COMPUTE #POINTS/8+1
 PUSH CX                ;SAVE COUNTER
 PUSH FUNC              ;SAVE FUNCTION FLAG
 LDS BX,DWORD PTR[BP+06] ;GET 8255 ADDRESS
 MOV AX,[BX]
 MOV BP,AX              ;POINT BP TO PORT A
 ADD AX,2D
 MOV DS,AX              ;POINT DS TO PORT C
 INC AX
 MOV DX,AX              ;POINT DX TO CONTROL PORT
 MOV AL,0B6H            ;DEFINE MODE 1, A AND B INPUT
 OUT DX,AL              ;BITS C6 AND C7 OUTPUT
 MOV AL,0CH
 OUT DX,AL
 MOV AL,0EH             ;'PRESS' PLAY TAPE BY
 OUT DX,AL              ;RESETTING BITS C6 AND C7
 MOV AX,0600H
 CALL TIMER             ;'HOLD' FOR 60 MSEC
 MOV AL,0FH
 OUT DX,AL
 MOV AL,0DH             ;'RELEASE' BY SETTING
 OUT DX,AL              ;BITS C7 AND C6
 MOV AX,0C000H
 CALL TIMER             ;WAIT 2 SECONDS
 MOV AX,0C000H
 CALL TIMER             ;WAIT 2 SECONDS
 MOV AL,05H
 OUT DX,AL              ;ENABLE PORT B INTR
 MOV DX,DS              ;POINT DX TO PORT C
 POP AX                 ;RECOVER FUNCTION FLAG
 PUSH AX
 CMP AX,0
 JE READY               ;IF 0 JUMP TO READY
 PUSH CX                ;SAVE EXTRA COPY OF CX
 MOV CX,28H
 MUL CL                 ;COMPUTE HALF MATCH COUNTER
 INC AX
 POP CX                 ;RECOVER CX
 PUSH CX
 PUSH AX                ;SAVE HALF MATCH COUNTER
COUNT: POP AX
 DEC AX                 ;DECREMENT HALF MATCH COUNTER
 PUSH AX
 JNE RD1                ;IF NOT ZERO JUMP TO RD1
 POP AX                 ;POP OFF HALF MATCH COUNTER
 JMP FAIL               ;JUMP TO FAIL
;THE FOLLOWING BLOCK IS DUPLICATED 4 TIMES
 IRP READ,<RD1,RD2,RD3,RD4>
READ: IN AL,DX          ;WAIT FOR DATA LATCH
 SHR AL,1
 JNC READ
 MOV DX,BP
 IN AX,DX               ;READ A POINT
 MOV DX,DS
 SHR AX,1
 RCL BX,1               ;ROTATE LEAST SIGNIFICANT
 RCL CX,1               ;BIT INTO BX,CX
 CMP BX,DI              ;COMPARE LOW WORD TO CODE
 JE MATCH               ;IF EQUAL JUMP TO MATCH
ENDM
RD5: IN AL,DX           ;WAIT FOR DATA LATCH
 SHR AL,1
 JNC RD5
 MOV DX,BP
 IN AX,DX               ;READ A POINT
 MOV DX,DS
 SHR AX,1
 RCL BX,1               ;ROTATE LEAST SIGNIFICANT
 RCL CX,1               ;BIT INTO BX,CX
 CMP BX,DI              ;COMPARE LOW WORD TO CODE
 JNE RD1                ;IF NOT EQUAL LOOP
MATCH: CMP CX,SI        ;COMPARE HIGH WORD TO CODE
 JNE COUNT              ;IF NOT EQUAL JUMP TO COUNT
 POP AX                 ;IF EQUAL POP OFF FAIL COUNT
 POP CX                 ;RECOVER COUNTER
READY: MOV BX,ES        ;STORE SEGMENT IN BX
START: MOV DI,10H
WT0: IN AL,DX           ;WAIT FOR DATA LATCH
 SHR AL,1
 JNC WT0
 MOV DX,BP
 IN AX,DX               ;DISCARD FIRST POINT
 MOV DX,DS
;THE FOLLOWING BLOCK IS DUPLICATED 8 TIMES
IRP WAIT,<WT1,WT2,WT3,WT4,WT5,WT6,WT7,WT8>
WAIT: IN AL,DX          ;WAIT FOR DATA LATCH
 SHR AL,1
 JNC WAIT
 MOV DX,BP
 IN AX,DX               ;READ IN DATA
 MOV DX,DS
 STOS DUMMY             ;STORE DATA
ENDM
 SHR DI,1               ;DECREASE OFFSET TO 10H
 INC BX                 ;INCREMENT SEGMENT
 MOV ES,BX
 LOOP WT1               ;DECREMENT COUNT, LOOP IF NOT 0
 POP AX                 ;RECOVER FUNCTION FLAG
 POP CX                 ;RESTORE COUNT AND
 POP ES                 ;SEGMENT REGISTERS
 PUSH ES
 PUSH CX
 PUSH AX
 MOV BX,ES
 CMP AX,0               ;TEST FUNCTION FLAG
 JNE FINISH             ;IF NOT 0 JUMP TO FINISH
 MOV AH,11D
 STI
 INT 21H                ;IF 0 READ KEYBOARD
 CLI
 AND AL,AL
 JNZ FINISH             ;IF KEY PRESSED JUMP TO FINISH
 JMP START              ;IF NOT REPEAT FROM START
FAIL: MOV CX,DS         ;SAVE CONTROL POINTER IN CX
 POP AX                 ;POP OFF TEMPORARY VALUES
 POP AX
 POP AX
 POP AX
 POP ES                 ;RESTORE ES
 POP DS                 ;AND DS
 MOV BP,SP              ;POINT BP TO STACK
 PUSH DS
 PUSH ES
 LDS BX,DWORD PTR[BP+22] ;POINT BX TO FLAG
 MOV AX,0
 MOV [BX],AX            ;SET FLAG=0
 MOV BP,9000H           ;PREPARE TO REWIND 1.5 SEC
 MOV DS,CX              ;POINT DS TO CONTROL PORT
 JMP STOP               ;JUMP TO STOP
FINISH: MOV BP,5800H    ;PREPARE TO REWIND 0.9 SEC
 POP AX                 ;POP OFF FUNCTION FLAG
 POP CX                 ;COUNT REGISTER
 POP ES                 ;AND SEGMENT REGISTER
STOP: MOV DX,DS
 INC DX                 ;POINT DX TO CONTROL PORT
 MOV AL,0CH             ;'PRESS' STOP TAPE BY
 OUT DX,AL              ;RESETTING BIT C6
 MOV AX,0600H
 CALL TIMER             ;'HOLD' FOR 60 MSEC
 MOV AL,0DH             ;'RELEASE' BY SETTING
 OUT DX,AL              ;BIT C6
 MOV AX,6000H
 CALL TIMER             ;WAIT 1 SECOND
 MOV AL,0EH             ;'PRESS' REWIND TAPE BY
 OUT DX,AL              ;RESETTING BIT C7
 MOV AX,0600H
 CALL TIMER             ;'HOLD' FOR 60 MSEC
 MOV AL,0FH             ;'RELEASE' BY SETTING
 OUT DX,AL              ;BIT C7
 MOV AX,BP
 CALL TIMER             ;WAIT 0.9 OR 1.5 SECONDS
 MOV AL,0CH             ;'PRESS' STOP TAPE BY
 OUT DX,AL              ;RESETTING BIT C6
 MOV AX,0600H
 CALL TIMER             ;'HOLD' FOR 60 MSEC
 MOV AL,0DH             ;'RELEASE' BY SETTING
 OUT DX,AL              ;BIT C6
 MOV AX,6000H
 CALL TIMER             ;WAIT 1 SECOND
 STI                    ;ENABLE INTERRUPTS
 POP ES                 ;RESTORE ORIGINAL REGISTERS
 POP DS
 POP BP
 RET 14H                ;RETURN
VCR ENDP
VCODE ENDS
END
```

FIGURE 2 MACRO ASSEMBLER source code for control of interface. This subroutine programs the 8255 for data transfer, stores data in microcomputer memory, and controls the video cassette recorder. Instructions for calling the subroutine and a description of its algorithm are given in the comments.

bits of the first 32 points of the marker. The least significant bit of each sampled point is rotated into a 32-bit pair of registers and is compared with the template. This comparison is done in two stages. Only if the first 16 bits match, are the other 16 bits compared. Since, for real data, the least significant bits are essentially random, a 16-bit match occurs with a frequency of ~1 in 2^{16} or about once every 1.5 s. If a specified number of half matches occur before a total match is found the subroutine aborts and returns an error code. If a total match is found data collection is started. The first two points (corresponding to the 33rd and 34th points of the marker) are missed. The next eight points are stored in an array that immediately preceeds the data array in memory. We recommend that the calling program verify that these 8 points are identical to the last 8 points of the 42-point marker. Thereafter, points are stored in the data array.

Control of the VCR is accomplished through use of bits 6 and 7 of port C of the 8255. These are not required for handshaking and are programmed for output. A 7442 BCD-to-Decimal converter is used to demultiplex these two bits. Codes C6 C7 = 00, 01, and 10 activate the play, stop, and rewind functions, respectively. Bits C6 and C7 are normally set, resulting in no action. When either or both bits are reset, the appropriate output of the 7442 is driven low, activating the corresponding 4N36 optoisolator. A software timer is used to control the duration of optoisolator activation and the period of time between VCR commands.

The only defect we have observed in testing this interface is occasional failure of the subroutine to detect the marker. This occurs <1% of the time. Often, a second subroutine call with the same marker will execute correctly. We believe this failure to result from misreading of the video tape.

The authors will be pleased to provide interested readers with diskette copies of the subroutine listed in Fig. 2 and its assembled object code. Such requests should include a formatted diskette in a stamped, self-addressed diskette mailer.

Received for publication 23 February 1987 and in final form 17 June 1987.

REFERENCE

Bezanilla, F. 1985. A high capacity data recording device based on a digital audio processor and a video cassette recorder. *Biophys. J.* 47:437–441.