High Speed Magnetic Tape Interface for a Microcomputer

John C. Scott

DIVISION OF ATMOSPHERIC RESEARCH TECHNICAL PAPER No. 7
COMMONWEALTH SCIENTIFIC AND INDUSTRIAL RESEARCH ORGANIZATION, AUSTRALIA 1984

High Speed Magnetic Tape Interface for a Microcomputer

By John C. Scott

Division of Atmospheric Research Technical Paper No. 7

Commonwealth Scientific and Industrial
Research Organization, Australia
1984

CSIRO Aust. Div. Atmos. Res. Tech. Pap. No. 7, 1-17 (1984)

HIGH SPEED MAGNETIC TAPE INTERFACE FOR A MICROCOMPUTER

JOHN C. SCOTT

CSIRO DIVISION OF ATMOSPHERIC RESEARCH,
PRIVATE MAILBAG NO. 1, MORDIALLOC, 3195

Abstract

A high speed interface for transferring data between a 9 track high performance magnetic tape unit and an Apple type microcomputer is described. The interface uses direct memory access (DMA) which is capable of data transfer rates up to 2×10^6 8 bit bytes per second. After initiation of data transfer, the process is completely transparent to the processor. All functions of the magnetic tape unit are available to the programmer via any level of programming, and the tapes produced are fully compatible with ANSI and IBM specifications.

Introduction

With the increased use of small inexpensive microcomputer systems in the laboratory a method of storing large quantities of data at very high speed has become a necessity.

Expensive high quality magnetic tape units belonging to a past generation of mini and mainframe computers are often available in the laboratory but are not used with microcomputers because of a lack of hardware and software interface facilities. This paper describes an interface which allows the transfer of data between a high volume, high speed, magnetic tape unit (Digi Data model 1640) and a 8 bit microcomputer (Basis Med Fly).

The interface was designed specifically to record data generated by an airborne, spectrally scanning visible-near IR radiometer. The instrument is described in detail elsewhere (Ref 1.) and is capable of generating data at rates up to 6K (8 bit) bytes per second. The controlling processor has considerable overheads. Apart from performing the principal function of data transfer it also displays on a VDU selected real time data in graphical form as well as controlling many aspects of the radiometer operation. The magnetic tape unit is a synchronous device requiring a data transfer rate of 60K bytes per second. In order that the dump to magnetic tape does not interfere with the data transfer, the magnetic tape dump is performed by direct memory access on a cycle steal basis. This results in a magnetic tape transfer which is completely transparent to the processor operation.

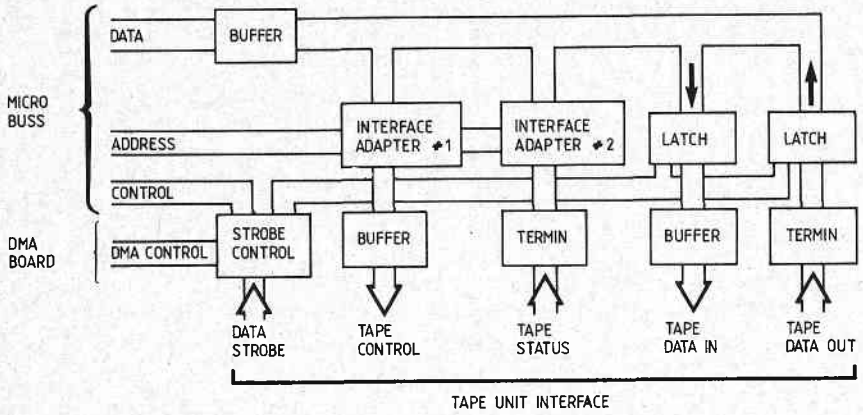
Hardware Description

a) Tape Unit Control

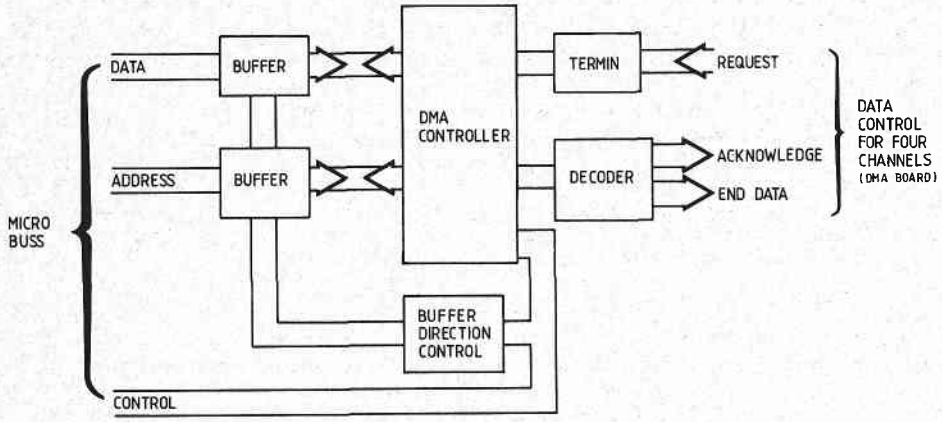
The interface consists of two separate sections, the first is the tape unit control and the second is the DMA control. Each of these is contained on its own double sided printed circuit board and they occupy two of the expansion slots of the Basis.

Fig. 1a shows a block diagram of the tape control section which supplies the command information to the tape unit as well as transferring all status information to the computer. This section also performs the latching and buffering functions for all data transferred to and from the tape unit.

The tape control unit is connected to the buss system of the computer via one of the computers 50 pin expansion slots, registers within the interface adaptor appear as memory locations to the processor. During a tape control function, data is transferred either to interface adaptor number 1 (for tape control) or from interface adaptor number 2 (for status read). Once a command specifying data transfer has been sent, data is clocked out of (or into) computer memory using the strobe lines of the tape unit and the 8 bit data latches. The strobe frequency is controlled directly by the tape unit and for a 1600 bit per inch transfer at 35.5 inches per second tape speed the rate is approximately 60K bytes per second.



(a)



(b)

Fig. 1 System block diagram a) Tape unit control
b) Direct memory access control.

The strobe control consists of logic circuitry which supplies the appropriate commands to the data latches, computer buss control logic, and the DMA controller board, from the tape status and DMA control output.

The complete circuit diagram of the tape unit control is shown in appendix 1.

b) DMA Control

Fig. 1b shows a block diagram of the DMA controller. In contrast to the tape control board, this board is a general purpose DMA controller capable of 4 channel operation at speeds up to $2x10^6$ bytes per second. The heart of the board is a Motorola device type MC6844 DMA controller which is arranged to operate on the buss system of the Basis (which uses a 6502 microprocessor). The MC6844 is a relatively complex device as far as its software requirements are concerned but has the advantage of simple hardware requirements. The MC6844 is a very versatile device, its operation is explained in detail in Ref. 2.

Once the MC6844 has been initialised by defining the starting address of the block of memory to be transferred along with the number of Bytes, an instruction is given to the MC6844 which sets it to a mode where it waits for either a read strobe pulse (RS) or a write strobe pulse (WR) from the tape unit, this becomes the transfer request (TxRO or TxR1) on the DMA controlled board. The DMA controller then issues a buss request (DRQ) to the processor via its DMA line. This buss request informs the processor that the DMA controller requires control over both the data and address busses and that they should be placed in a tri-state (high impedance) mode. The processor acknowledges this request when it is ready via the DMA grant line (DGRNT). The DMA controller then places the relevant address of the memory location to be transferred onto the address buss and latches the corresponding data into the tape interface data latches. The controller then acknowledges the data transfer via the transfer acknowledge line (TxA) and the buss control is passed back to the processor.

The description above refers to data transferred on a cycle steal basis. This has the advantage that the DMA transfer uses only a small percentage of the number of cycles available to the processor, as a result, the only effect of the DMA action is to slow the processor down by this same percentage. Another mode of operation available to the programmer is the halt burst mode, this requires the processor to relinquish the data and address buss for the entire time of the data transfer. This has the advantage that data can be transferred at very high speed (up to $2x10^6$) bytes per second) but has the disadvantage that the processor is idle for long periods of time which in certain applications is unacceptable.

The complete circuit diagram for the DMA controller board is shown in appendix 2. The main control lines (TxA, TxR and DND) are shown numbered 0 to 3-each number refers to the channel being used.

In the software description which follows, two of the four available channels of DMA are used; Channel 0 is for writing to tape and channel 1 is for reading from tape. Appendix 4 shows the wiring between the DMA controller, the tape control and a typical 9 track magnetic unit fitted

with a formatter having a Pertec type interface (Digi-Data 1640 series). Because channels 0 and 1 are used, the transfer acknowledge is generated by 'OR'ing the signals (TxAO,TxA1) from channels 0 and 1. The last word indicator (LWD) for the tape unit is only required on the write channel (Ch.0). The transfer requests are derived from the strobe pulses generated by the tape unit (RS and WS). All other signals are either tape control lines or data lines.

Programming

Using two channels of DMA for the tape interface, as described above, enables the programmer to write a record, read the same record and verify its accuracy with a minimum of programming steps and in a minimum time. Fig. 2 shows flow charts of two programs, one is used to write data at very high speed and is written in 6502 machine code, the second is used to read data in the same format but is written in the Basic language. This second program is far slower in execution than the machine language program but has the advantage of being very versatile, easy to understand, and trivial to modify. Listings of the two programs are given in appendix 5 and the description below should be read in conjuction with this appendix and the block diagrams.

Both programs start by initialising the peripheral interface adaptors, programming information for this device (MC6821) is given in detail in Ref. 2. The initialising sequence specifies adaptor 1 as an output and adaptor 2 as an input and one line of adaptor 2 in program 1 is programmed to generate an interrupt on a positive transition. This is generated from the data source and initiates the transfer to magnetic tape.

Both programs then initialise the DMA controller by specifying the following:

1. Which channels are read and write and whether they should count up or down

2. Transfer to be disabled

3. Four channel operation with no chaining selected

4. On which channel the end of data transfer should occur.

Reference should be made to appendix 3 to establish the correct data to be used for the above initialisation.

In the writing to tape routine (Fig. 2a), program control is transferred to the background program. On an interrupt, control is sent to an assembly language interrupt routine which performs the block specification and transfer command. Control then returns to the background program which waits for further interrupts. In the reading from tape routine (Fig. 2b), program control flows from the DMA initialisation to the block specification and transfer command routines. When transfer is complete either a new block of data is read or an analysis program is run.

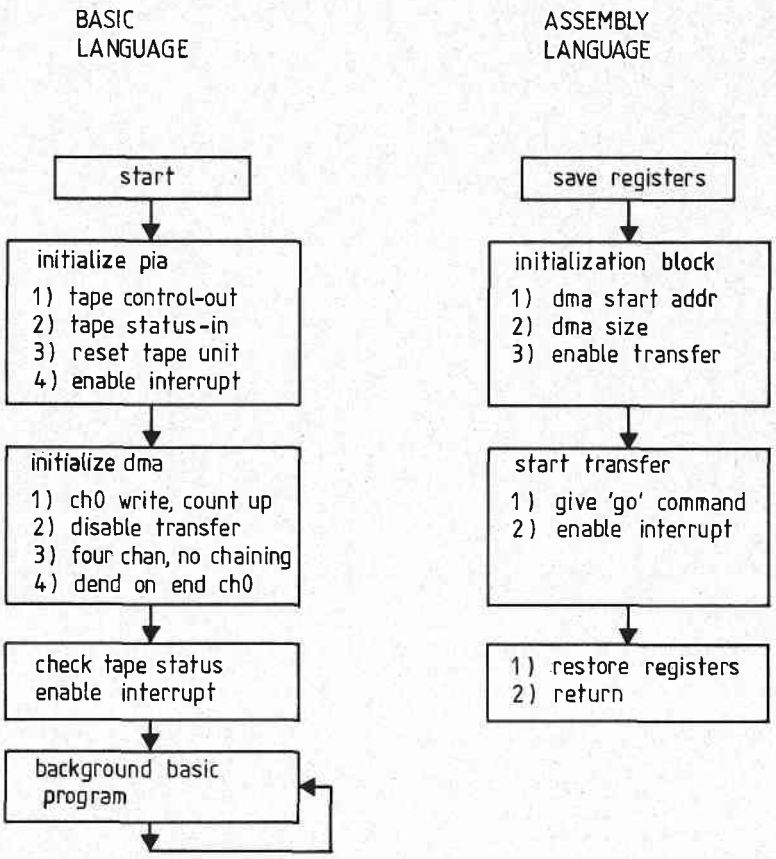


Fig. 2(a) Flow chart of hybrid program written in basic and assembler for writing data to tape.

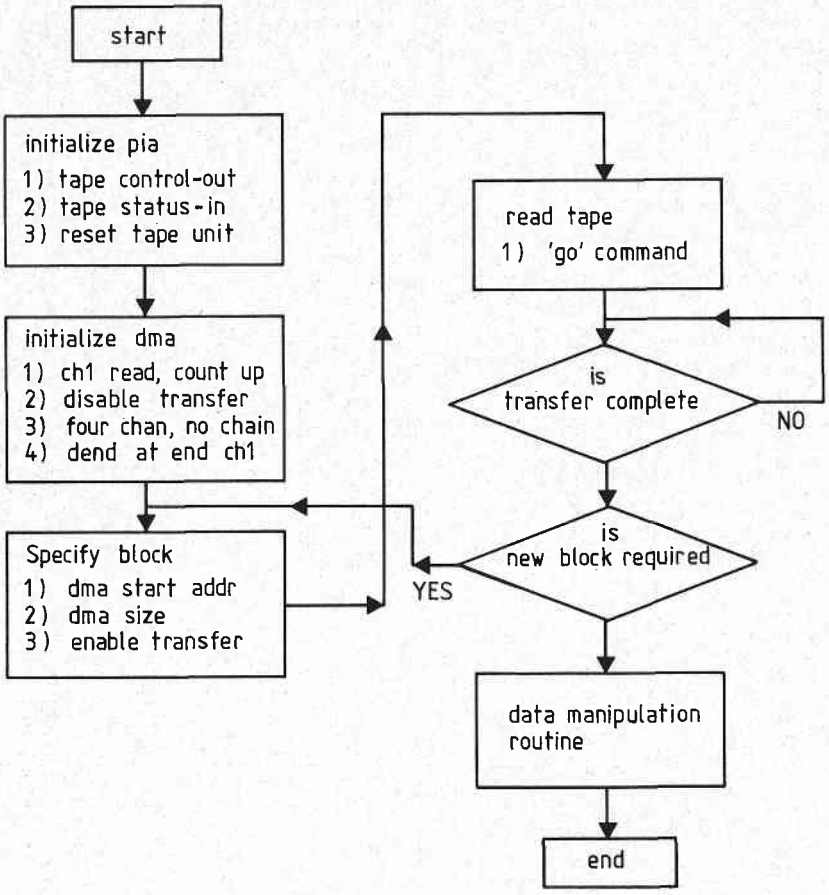


Fig. 2(b) Flow chart of basic program for reading data from tape.

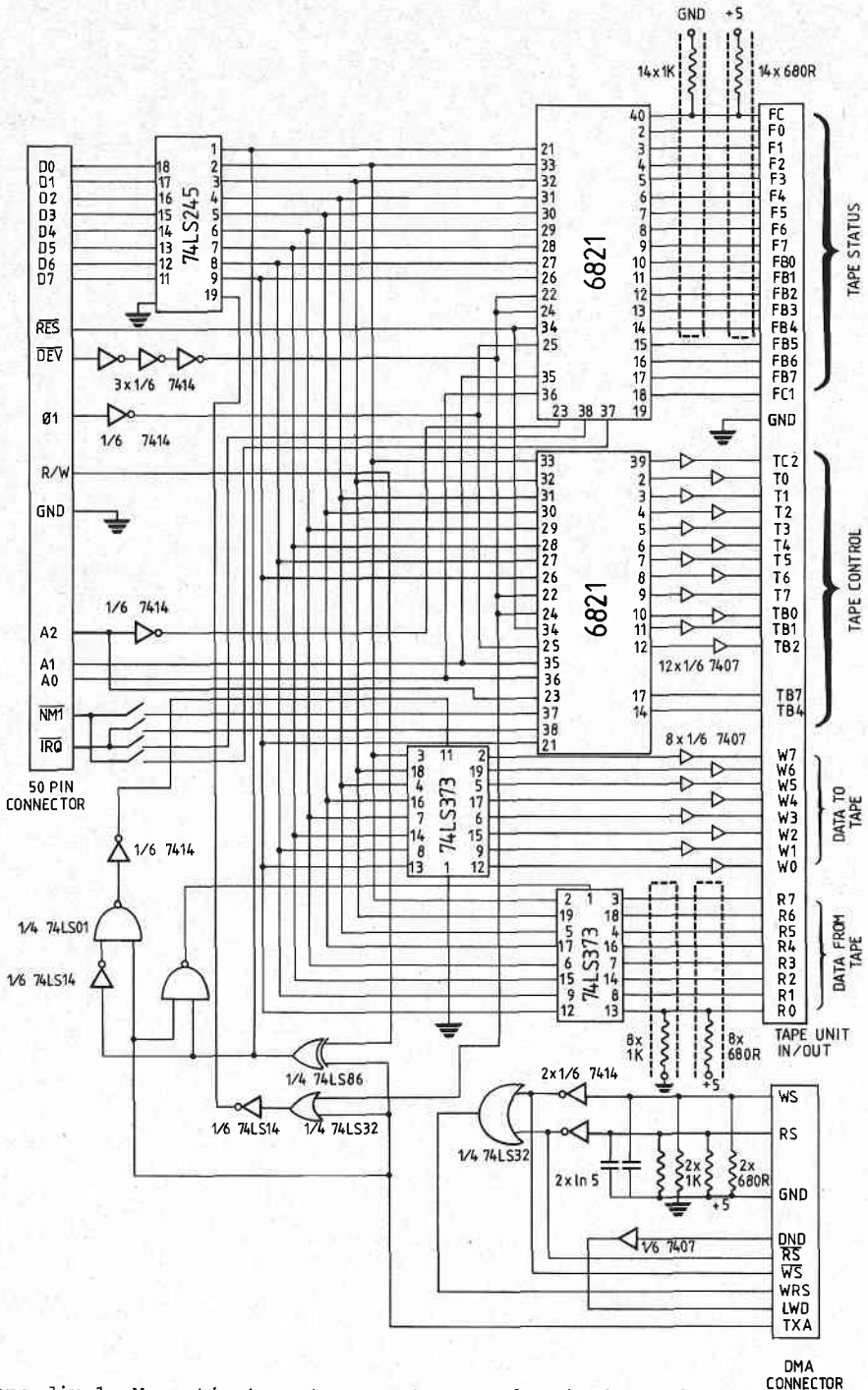
Conclusion

This note describes an inexpensive means of interfacing a small personal computer to a large capacity, high speed magnetic tape transport. The interface is very versatile, enabling complete control of all aspects of the transfer by the programmer. The interface fills the growing need for a means of recording and reading data in a form which is compatible with many mainframe computers.

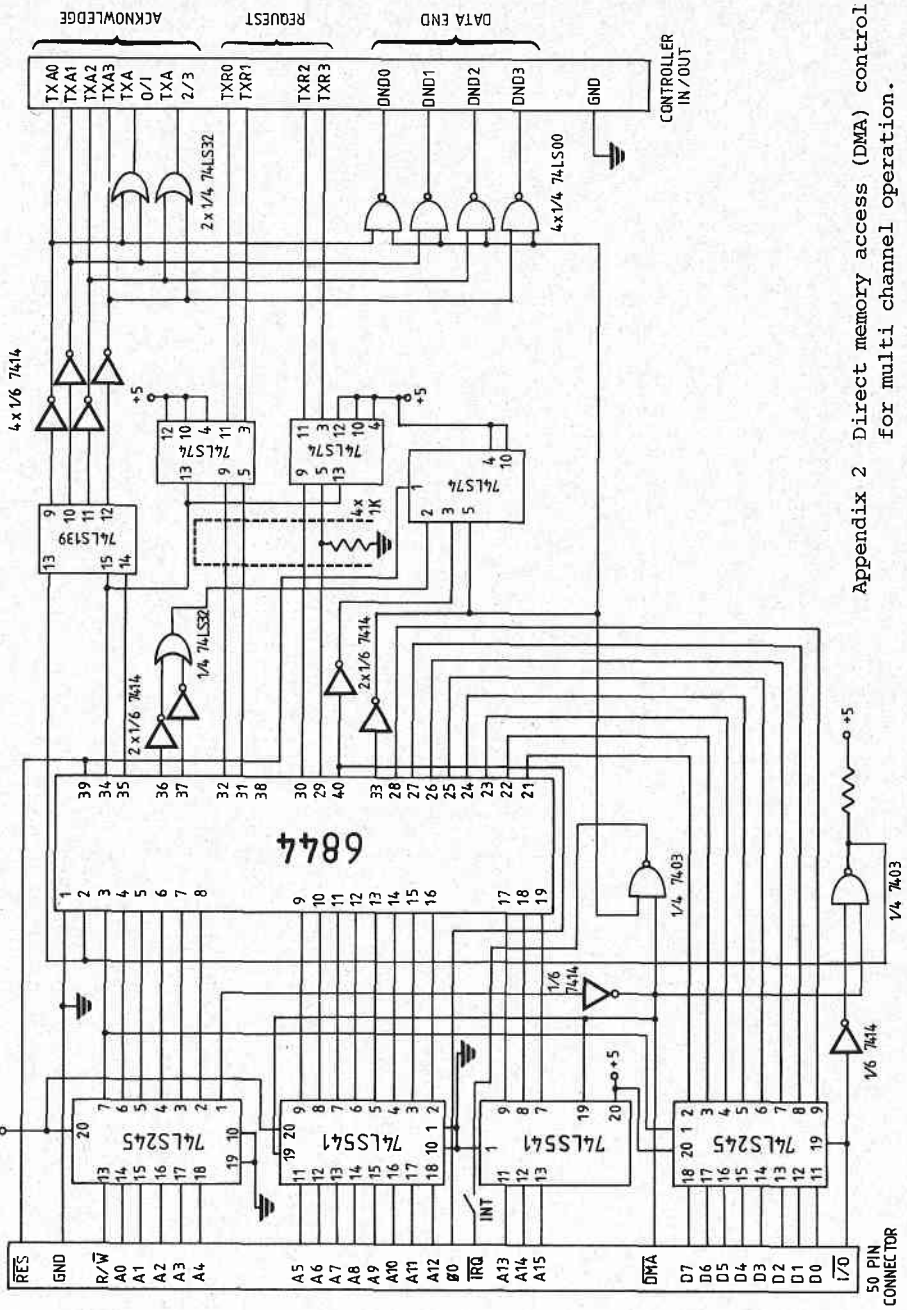
References:

1. Stephens, G.L. and Scott, J.C. A High Speed Spectrally Scanning Radiometer (SPERAD) for Airborne Measurements of Cloud Optical Properties. Journal of Atmospheric and Oceanic Technology (in press).

2. Motorola Microprocessor Data Manual. 1981 Motorola Inc.



Appendix 1 Magnetic tape transport control unit including data latch and buffering.



Appendix 2 Direct memory access (DMA) control unit for multi channel operation.

DMA CONTROLLER REGISTER SUMMARY

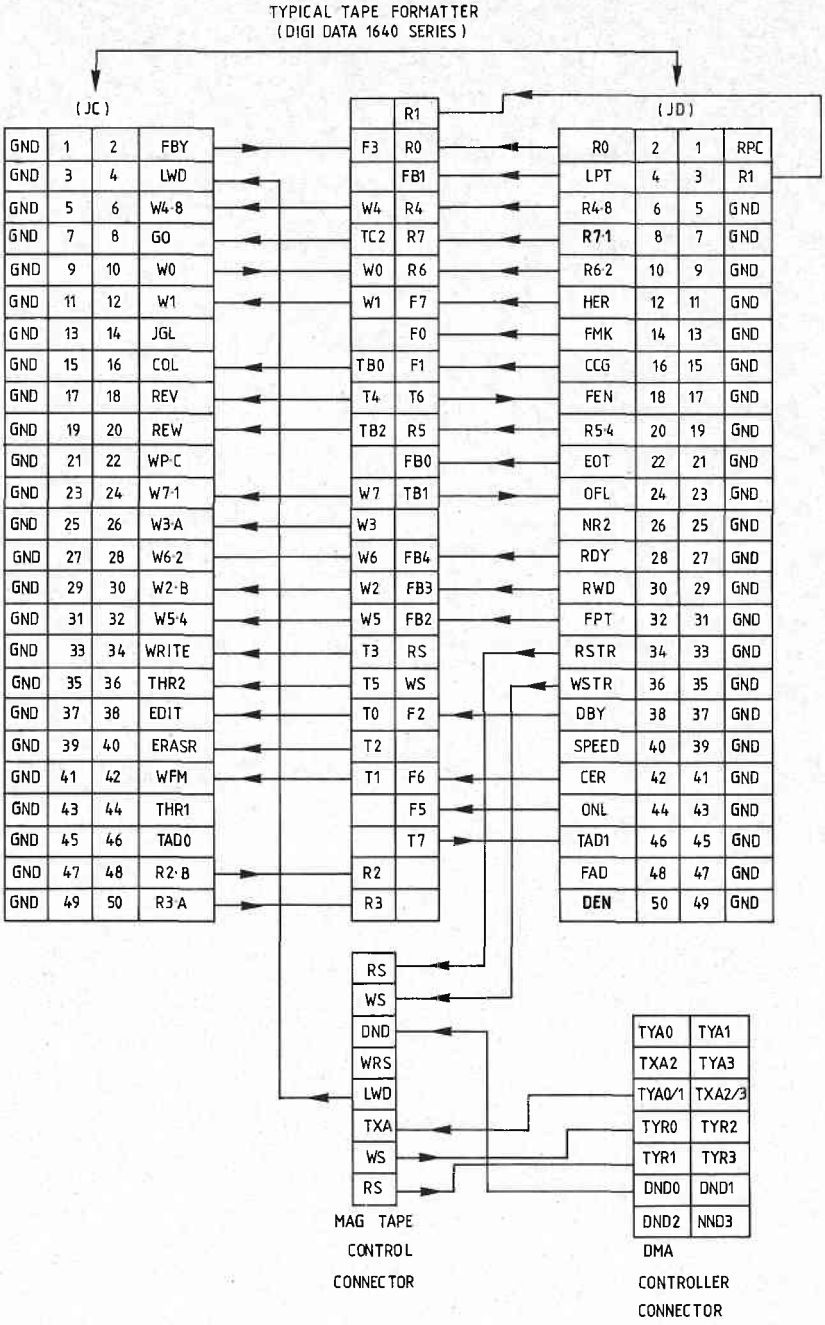
BIT NUMBER

REGISTER	7	6	5	4	3	2	1	0	ADDRESS ($)
CONTROL CH0	end (dend)	ready			address up/down	MCA	MCB	R/W	10
CONTROL CH1	dend	ready			up/down	MCA	MCB	R/W	11
CONTROL CH2	dend	ready			up/down	MCA	MCB	R/W	12
CONTROL CH3	dend	ready			up/down	MCA	MCB	R/W	13
PRIORITY CONT.	rotate				request RE3	RE2	RE1	RE0	14
INTERUPT CONT.	IRQ FLG				dend enable 3	DE2	DE1	DE0	15
DATA CHAIN					2/4 chan	data chain B	data chain A	d. chain enable	16

	ADDRESS ($)			
	0	1	2	3
ADDR HIGH	0	1	2	3
ADDR LOW	4	5	6	7
BYTE COUNT HIGH	8	9	A	B
BYTE COUNT LOW	C	D	E	F

← CHANNEL NUMBER

Appendix 3 6844 direct memory access controller integrated circuit register summary.



Appendix 4 Typical connection showing the interface between the tape control unit, DMA controller and a Digi data magnetic tape unit.

Appendix 5a Basic program to read data from tape.

```
10  REM  BASIC PROGRAM TO SET UP INTERUPT ROUTINE
20  REM  TO STORE DATA ON TAPE FROM MEMORY
30  CALL 37166: REM   INITIALIZE PIA
40  CALL 37230: REM   INITIALIZE DMA
50  CALL 37254: REM   INITIALIZE RAM
60 A = 49392: REM  PIA BASE ADD
70 FBDD = 64477
80  CALL 37120: REM   TAPE STATUS
90 BUF = 832: REM  STATUS BUFFER
100 X =  PEEK (BUF)
110  IF X = 0 GOTO 140
120  IF X = 4 GOTO 210
130  GOTO 170
140  PRINT "NO WRITE ENABLE RING"
150  CALL FBDD: REM   RING BELL
160  GOTO 190
170  PRINT "TAPE UNIT NOT ON LINE"
180  CALL FBDD: REM  RING BELL
190  INPUT "IS TAPE NOW READY? (Y.N) ";Y$
200  GOTO 80
210  CALL 37140: REM   INTERUPT ENABLE & START
220  REM  *** BACKGROUND PROGRAM FOLLOWS ***
230  END
```

Appendix 5b Basic program to supervise the assembler program (App. 5c) used to write data to tape.

```
SOURCE FILE: TECHP
COF1:              1 P2AC     EQU  $COF1      ;FOR 'GO' COMMAND
COF4:              2 P1A      EQU  $COF4      ;FROM TAPE STATUS
COF6:              3 P1B      EQU  $COF6      ;TAPE STATUS + INT-SYNC.
COF7:              4 P1BC     EQU  $COF7      ;INTERUPT CONT.
COF0:              5 P2A      EQU  $COF0      ;TO TAPE FORMATTER CONTROL
COF2:              6 P2B      EQU  $COF2      ;TO TAPE DIRECT CONTROL
6E90:              7 FLG      EQU  $6E90      ;INTERUPT DISABLE FLG
00FE:              8 LUTBL    EQU  $FE        ;LUT ADD BUF LSB
00FF:              9 LUTBH    EQU  $FF        ;LUT ADD BUF MSB
FF4A:             10 SAVE     EQU  $FF4A      ;SAVE REGISTERS
FF3F:             11 REST     EQU  $FF3F      ;RESTORE REGISTERS
C500:             12 DMA      EQU  $C500      ;DMA BASE ADDRESS
0340:             13 BUF      EQU  $340
0000:             14 * DATA TO BE TRANSFERED ARE IN $5000 TO $6D12
----- NEXT OBJECT FILE NAME IS TECHP.OBJ0
8F00:             15          ORG  $8F00
8F00:48           16          PHA              ;SAVE REG
8F01:AD 90 6E     17          LDA  FLG         ;IS THERE A
8F04:D0 02        18          BNE  LP1         ;CURRENT INTERUPT?
8F06:68           19          PLA
8F07:40           20          RTI               ;YES-RETURN
8F08:68           21 LP1      PLA
8F09:20 4A FF     22          JSR  SAVE        ;SAVE REGISTERS
8F0C:A9 00        23          LDA  #0
8F0E:8D 90 6E     24          STA  FLG         ;DISABLE INTERUPT FLG
8F11:             25 * INITIALIZE DMA BLOCK PARAMETERS
8F11:A9 50        26          LDA  #$50
8F13:8D 00 C5     27          STA  DMA         ;MSB START
8F16:A9 00        28          LDA  #$0
8F18:8D 01 C5     29          STA  DMA+1       ;LSB START
8F1B:A9 1D        30          LDA  #$1D
8F1D:8D 02 C5     31          STA  DMA+2       ;MSB LENGTH
8F20:A9 12        32          LDA  #$12
8F22:8D 03 C5     33          STA  DMA+3       ;LSB LENGTH
8F25:             34 *  INITIALIZE DMA CONTROL
8F25:A9 01        35          LDA  #1
8F27:8D 10 C5     36          STA  DMA+$10     ;DATA TO TAPE
8F2A:             37 *  START TRANSFER
8F2A:A9 B7        38          LDA  #$B7
8F2C:8D F0 C0     39          STA  P2A         ;START TAPE
8F2F:A9 34        40          LDA  #$34        ;WRITE ONE REC
8F31:8D F1 C0     41          STA  P2AC        ;'GO' LOW
8F34:A9 3C        42          LDA  #$3C        ;"GO"
8F36:8D F1 C0     43          STA  P2AC        ;'GO' HIGH
8F39:20 A8 91     44          JSR  FFG         ;SET INTERUPT FLAG
8F3C:20 3F FF     45          JSR  REST        ;RESTORE REGISTERS
8F3F:40           46          RTI               ;RETURN
```

```
8F40:             47 * THE SUBROUTINES THAT FOLLOW CAN BE CALLED
8F40:             48 * FROM BASIC AND ASSEMBLY LANGUAGE ROUTINES
----- NEXT OBJECT FILE NAME IS TECHP.OBJ1
9100:             49          ORG  $9100
9100:             50 * SUBROUTINE ERROR- TAPE STATUS
9100:AD F4 C0     51 ERROR    LDA  P1A        ;GET STATUS
9103:29 20        52          AND  #$20
9105:8D 40 03     53          STA  BUF
9108:AD F6 C0     54          LDA  P1B
910B:29 04        55          AND  #$4        ;WRITE RING
910D:0D 40 03     56          ORA  BUF
9110:8D 40 03     57          STA  BUF
9113:60           58          RTS
9114:             59 * SUBROUTINE START- STARTS DMA AND TAPE
9114:A9 50        60 START    LDA  #$50       ;NEW RECORD
9116:8D 00 C5     61          STA  DMA
9119:A9 00        62          LDA  #$0
911B:8D 01 C5     63          STA  DMA+1
911E:A9 1C        64          LDA  #$1C
9120:8D 02 C5     65          STA  DMA+2      ;BLOCK LENGTH
9123:A9 30        66          LDA  #$30
9125:8D 03 C5     67          STA  DMA+3
9128:A9 1C        68          LDA  #$1C       ;ENABLE INTERUPT
912A:8D F7 C0     69          STA  P1BC
912D:60           70          RTS
912E:             71 * SUBROUTINE PINIT- INITIATE PIA'S
912E:A9 00        72 PINIT    LDA  #0
9130:8D F1 C0     73          STA  P2A+1
9133:8D F3 C0     74          STA  P2A+3      ;SELECT COTROL
9136:8D F5 C0     75          STA  P2A+5
9139:8D F7 C0     76          STA  P2A+7
913C:A9 FF        77          LDA  #$FF
913E:8D F0 C0     78          STA  P2A        ;TAPE CONT  OUT
9141:A9 F7        79          LDA  #$F7
9143:8D F2 C0     80          STA  P2B        ;7 OUT 1 IN (MIRROR)
9146:A9 00        81          LDA  #0
9148:8D F4 C0     82          STA  P1A        ; FROM TAPE
914B:8D F6 C0     83          STA  P1B        ; FROM TAPE
914E:A9 3C        84          LDA  #$3C
9150:8D F1 C0     85          STA  P2A+1
9153:8D F3 C0     86          STA  P2A+3
9156:8D F5 C0     87          STA  P2A+5
9159:A9 14        88          LDA  #$14       ;INTERUPT DISABLE
915B:8D F7 C0     89          STA  P1BC       ; $1C FOR ENABLE
915E:A9 FF        90          LDA  #$FF
9160:8D F0 C0     91          STA  P2A
9163:A9 BF        92          LDA  #$BF       ; PULSE FEN LOW
9165:8D F0 C0     93          STA  P2A        ; TAPE RESET
9168:A9 FF        94          LDA  #$FF
916A:8D F2 C0     95          STA  P2B
916D:60           96          RTS
```

```
916E:            97 * SUBROUTINE DINIT- INITIATES DMA CONTROL
916E:A9 00       98 DINIT    LDA  #0
9170:8D 10 C5    99          STA  DMA+$10    ;CH 0 FROM TAPE
9173:8D 14 C5   100          STA  DMA+$14    ;ALL CH'S OFF
9176:A9 08      101          LDA  #$8
9178:8D 16 C5   102          STA  DMA+$16    ;FOUR CHAN OPERATION
917B:A9 03      103          LDA  #3
917D:8D 15 C5   104          STA  DMA+$15    ;DEND ON CH0 & CH1
9180:A9 01      105          LDA  #1
9182:8D 11 C5   106          STA  DMA+$11    ;CH1 TO TAPE
9185:60         107          RTS
9186:           108 * SUBROUTINE RINIT- INITIATE RAM
9186:A9 50      109 RINIT    LDA  #$50       ; CLEAR RAM
9188:85 FF      110          STA  LUTBH
918A:A9 00      111          LDA  #0
918C:85 FE      112          STA  LUTBL      ; START $5000
918E:A0 00      113          LDY  #0
9190:A9 00      114 LP20     LDA  #0
9192:91 FE      115          STA  (LUTBL),Y
9194:20 AE 91   116          JSR  NEXT       ;NEXT IN BLOCK
9197:A5 FF      117          LDA  LUTBH
9199:C9 8F      118          CMP  #$8F       ; IS IT FINISHED
919B:D0 F3      119          BNE  LP20       ; NOT FINISHED
919D:A9 8F      120          LDA  #$8F
919F:8D FD 03   121          STA  $3FD       ; HIGH BYTE
91A2:A9 FF      122          LDA  #$FF       ;INITIAL INT.
91A4:8D 90 6E   123          STA  FLG        ;ENABLE FLG
91A7:60         124          RTS
91A8:           125 * SUBROUTINE FFG -SET INTERUPT FLAG
91A8:A9 FF      126 FFG      LDA  #$FF
91AA:8D 90 6E   127          STA  FLG
91AD:60         128          RTS
91AE:           129 * SUBROUTINE NEXT- INC LUT BY ONE
91AE:A5 FE      130 NEXT     LDA  LUTBL
91B0:18         131          CLC
91B1:69 01      132          ADC  #1
91B3:85 FE      133          STA  LUTBL
91B5:A5 FF      134          LDA  LUTBH
91B7:69 00      135          ADC  #0
91B9:85 FF      136          STA  LUTBH
91BB:60         137          RTS

*** SUCCESSFUL ASSEMBLY: NO ERRORS

 0340 BUF          ?916E DINIT         C500 DMA          ?9100 ERROR
 91A8 FFG           6E90 FLG           8F08 LP1           9190 LP20
   FF LUTBH           FE LUTBL         91AE NEXT          C0F4 P1A
 C0F7 P1BC          C0F6 P1B           C0F1 P2AC          C0F0 P2A
 C0F2 P2B          ?912E PINIT         FF3F REST         ?9186 RINIT
 FF4A SAVE         ?9114 START
```

Appendix 5c Assembly language program used for writing data to tape.

```
10  REM   *** DATA FROM TAPE TO MEMORY VIA DMA CH1 ***
20  REM   **** INITIALIZE PIA'S
30 A = 49392: REM   BASE PIA ADD.
40  POKE A + 1,0: REM  SEL CONTROL REG
50  POKE A + 3,0: REM  SEL CONTROL REG
60  POKE A + 5,0: REM  SEL CONTROL REG
70  POKE A + 7,0: REM  SEL CONTROL REG
80  POKE A,255: REM    OUTPUT
90  POKE A + 2,255: REM    OUTPUT
100  POKE A + 4,0: REM    INPUT
110  POKE A + 6,0: REM    INPUT
120  POKE A + 1,60: REM   CA2-GO COMMAND(3C-34)
130  POKE A + 3,60: REM   INT DISABLE
140  POKE A + 5,60: REM   INT DISABLE
150  POKE A + 7,20: REM     INTERUPT ON L-H TRANS. OF CB2
160  POKE A,255
170  POKE A,191: REM   FEN LOW
180  POKE A + 2,255
190  REM   **** INITIALIZE DMA CONTROLLER
200  POKE B + 16,0: REM  CH 0 DATA FROM TAPE
210  POKE B + 17,1: REM  CH 1 DATA TO TAPE
220  POKE B + 22,8: REM    4 CHANNEL OPERATION
230  POKE B + 21,3: REM  DENT SET ON READ AND WRITE
240  GOTO 430
250  REM   **** SPECIFY BLOCK
260  POKE B,79: REM  CH 0 ADD HI
270  POKE B + 1,255: REM  CH 0 ADD LO
280  POKE B + 2,30: REM  CH 0 COUNT HI
290  POKE B + 3,20: REM  CH 0 COUNT LO
300  POKE B + 4,79: REM  CH 1 ADD HI
310  POKE B + 5,255: REM  CH 1 ADD LO
320  POKE B + 6,30: REM  CH 1 COUNT HI
330  POKE B + 7,20: REM  CH 1 COUNT LO
340  POKE B + 20,0: REM  BOTH CHANNELS OFF
350  RETURN
360  REM    **** START TRANSFER
370  POKE B + 17,0: REM  CH 1 DATA FROM TAPE
380  POKE B + 20,2: REM    CHAN 1 ON
390  POKE A,191: REM  FEN LOW
400  POKE A + 1,52: REM  GO LOW
410  POKE A + 1,60: REM  GO HIGH
420  RETURN
430 B = 50432: REM  DMA BASE ADD.
440  REM  **** READING DATA
450  GOSUB 250: REM  SPECIFY BLOCK
460  GOSUB 370: REM  START
470  REM   **** WAIT FOR DATA TRANSFER
480  FOR I = 1 TO 100: NEXT I
490  REM   **** NEW DATA?
500  INPUT "IS NEW DATA REQUIRED? (Y/N)       ";Y$
510  IF Y$ = "Y" THEN  GOTO 440
520  REM   **** DATA MANIPULATION ROUTINE FOLLOWS
530  END
```