1981

A Microprocessor Implementation of a Controller for a Dectape Transporter

Juan Gerardo Alvarado
University of Rhode Island

A MICROPROCESSOR IMPLEMENTATION OF A CONTROLLER FOR A DECTAPE TRANSPORTER

by

Juan Gerardo Alvarado

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

UNIVERSITY OF RHODE ISLAND

1981

MASTER OF SCIENCE THESIS

OF

JUAN GERARDO ALVARADO

Approved:

Thesis Committee

Major Professor D.W. Tufts

W. J. Palm

A A Michel

Dean of the Graduate School

UNIVERSITY OF RHODE ISLAND

1981

ABSTRACT

In the work of this thesis, microprocessor - based software and hardware have been designed to perform the functions of the PDP - 9 Dectape controller. For 8-bit data transfer, seven functions were implemented in software that simulate the move, stop, search, read, write and formating functions of the PDP - 9 Dectape controller. The hardware is designed to interface the Dectape Transport with the microprocessor, as well as to amplify and temporarily store signals. The new controller is implemented with an inexpensive Motorola 6800 microcomputer. Hence, the Dectape transports can be used with any computer or with special, experimental apparatus. Sixteen-bit transfer was also attempted but the resulting system still requires more testing and development.

ACKNOWLEDGEMENT

I would like to acknowledge the support given by Dr. D. W. Tufts who was my major professor and guided me in the development of this thesis. Specially I would like to thank Mr. Tony Zampini who helped me continously during the duration of my work. Also I would like to thank all my friends for their help and encouragement. Without their effort this thesis could not have been done.

TABLE OF CONTENTS

INTRODUCTION

A PDP - 9 minicomputer has been operating in the department of Electrical Engineering of the University of Rhode Island since 1969. This system is becoming old and difficult to maintain. Nevertheless, the tapes and the tape transports can be used with any computer, given a controller and interfaces.

The PDP - 9 TC02 Controller is used to direct the TU55 Transport to read forward or in reverse, to write forward or in reverse, to stop and to go[1,2,3], as well as to write the timing and mark tracks on the tape(formating). These operations have been implemented with a microprocessor - based software and hardware system. The software is divided into seven subprograms. These subprograms can be called up separately, thus, simulating the programming of the original TC02 controller. The functions of some of the original hardware circuits, for example, the up - to - speed delays and the status register, have been implemented in software. The seven subprograms implement the following functions: STOP, MOVE FORWARD and REVERSE, CHANGE DIRECTION, SEARCH, READ, WRITE and FORMATING.

The hardware consists of shift registers to temporarily store the word to be read or to be written, decoders to detect specific codes in the tape, interfaces, as well as amplifiers to match the signals from the tape to the signals which are required for the microprocessor.

The microprocessor - based controller can identify each one of the codes prewritten in the tape, count them, and select the exact location in the tape for reading or writing. The controller can cause the tape to stop at a location specified by the program. It can change the direction of motion of the tape at any desired time or block, make the tape go from one end to the other, or start the read or write functions whether in forward or in reverse. The controller can also, as the PDP - 9 does, format the tape with the number of blocks and words desired.

THE TC02 CONTROLLER[2]

A typical PDP - 9 tape system consists of one TC02 controller and up to eight tape transports.

The controller uses the Manchester phase recording technique rather than an amplitude sensing technique; thus, the tape speed need not be a precisely controlled parameter. Actually, the speed varies ±20 %, depending upon the diameter of the tape pack on the reel.

The controller uses a 10 - track read-write head. Tracks are arranged in five nonadjacent redundant channels: a timing channel, a mark channel, and three information channels.

The timing and mark track channels control the timing of operations within the TC02 controller unit and establish the format of data contained on the information channels. The timing and mark channels are recorded prior to all normal data reading and writing on the information channels. Information read from the mark channel is used during reading and writing of data, to indicate the beginning and end of data blocks and to determine the functions performed by the system in each control function.

The format of the tape is shown in fig. #1.

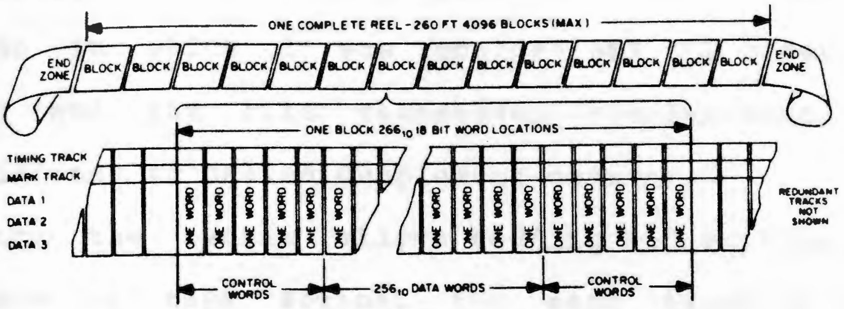


Fig. #1 TC02 tape format

Information is stored on the tape in blocks of flexible length determined by information on the mark track. Each block contains data and control words as shown in fig. #2.

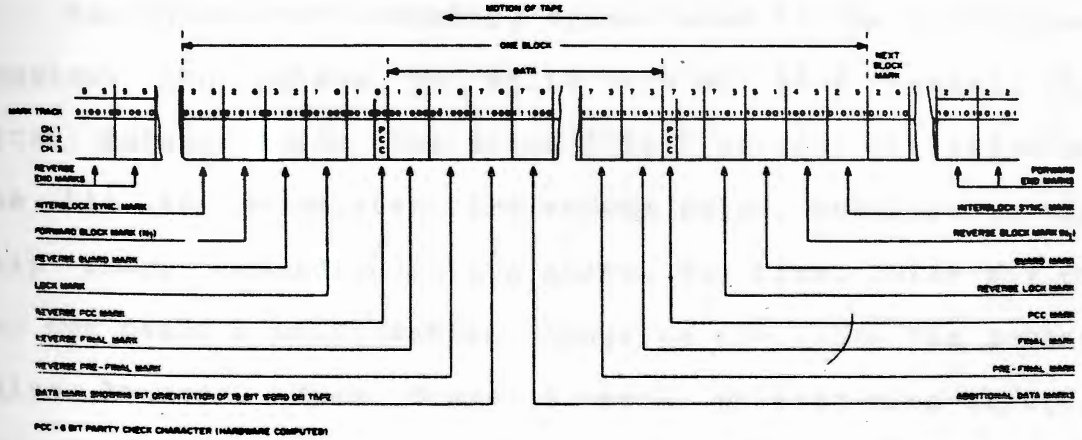


Fig. #2 block format

A given change of polarization on tape read in one direction produces a pulse opposite in polarity to that produced when the tape is read in the opposite direction. Consequently, a mark code read in the reverse of the direction in which it was recorded has the order of bits reversed and the bits themselves complemented. This correspondence is called complement obverse.

Since the system allows reading and writing in both directions of tape motion, the mark track is coded to present the same information when entering a block from either direction. As an example, the following three mark codes are used in the controller.

FORWARD	REVERSE (complement obverse)	FUNCTION
70	70	word mark
26	45	block mark
10	73	lock mark

RECORDING LOGIC

The Manchester recording system used in the controller requires two pulses to write each bit in a channel. The first pulse, loads the write flip flop with the value of the bit to be written. The second pulse, complements the flip flop, depending on its state. The first pulse may or may not cause a polarization change on the tape. The second pulse however, does cause a tape polarization change, because as a complement input, it changes the flip flop state. Fig. #3 shows the Read/Write logic, waveforms and the head connection with the amplifier.

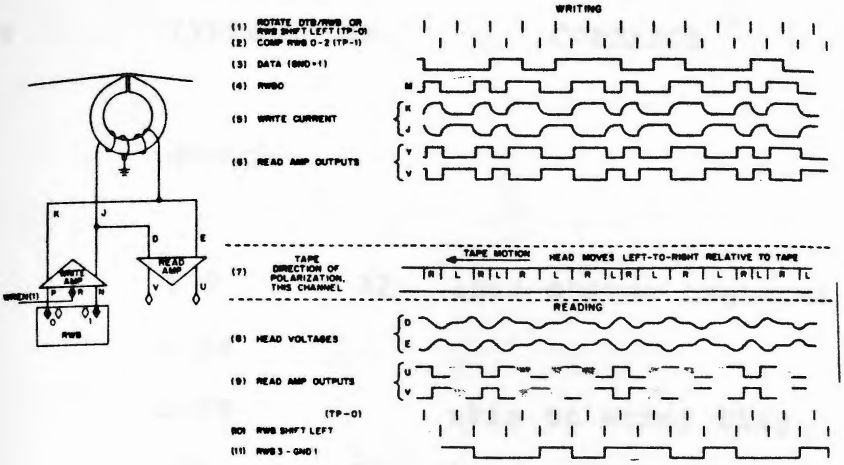


Fig. #3 Read/Write logic.

NOTE: This information has been taken from the TC02 Dectape Control Maintenance Manual and the TU55 Dectape Transport Manual of the PDP - 9.

PROGRAMMING

In programming the TC02 controller, a status register must be loaded with the corresponding information. Seven instructions in the PDP - 9 computer are available for this. With the loading of the status register, seven different functions can be programmed. They are: Move, Search, Read data, Read all, Write data, Write all, and Write timing and Mark track[2].

As an example of how to use the PDP - 9 instructions to program the controller, the following program searches, reads and writes 10 words, from location 100 to 1012 in the PDP - 9 memory.

```
MEMORY        INSTRUCTION        COMMENTS

              SEARCH

00            LAC         32     load status register
01            DTLA
02            DTEF               skip on error flag
03            JMP         02     loop
04            LAC         33     load status register
05            DTLA
06            DTDF               skip on flag
07            JMP         06     loop

              READ

10            LAC         35     load read code
11            DTXA
12            LAC         34
13            DAC         30
14            DTDF               skip on flag
15            JMP         14     loop
16            HLT                stop

              WRITE

10            LAC         36     load write code
11            DTXA
```

```
12        LAC        34    load word counter
13        DAC        30
14        DTDF
15        JMP        14
16        HLT              stop
                MEMORY REFERENCES

30        777634           word counter WC
31        001000           current address CA
32        370000           move reverse
33        331000           forward search
34        777766           10 words
35        003000           mask bit for read
36        005000           mask bit for write
```

The search program starts loading the status register with the move reverse code, then checks for end of tape line 02. When the end of tape is reached, it loads the status register with the search forward code and stops in line 07 when the block #100 is reached.

The read and write programs start in line 10 immediatly after the search program, and load the read (write) code from memory locations 35 (36) into the status register. Then they check for the Dectape flag which is set when the word counter reaches zero and stops the motors.

When programming, there are two hardware delays that have to be considered. The first is the 5 microsec. delay which insures that no direction signal changes occur during a stop operation. This delay cannot be controlled by software, but it is important to take it into consideration, for better understanding of the system function. The second delay is the 140 msec. up-to-speed delay. This is the most important delay and it can be controlled by program. This makes it extremely important because a program can fail if the delay has not been taken into consideration when programming. This delay starts, each time the status register is loaded, and lasts 140 msec. During this time no flags are read. Then if the status register is loaded near the tape end, the tape can go out of the reel without any flags being read. This situation is particularly critical when a controller is programmed to stop at the end of the tape, loads the status register in each loop. Even though the program is logically correct, it will fail because the loading of the status register in each loop, will automatically start the 140 msec. delay each time, thus inhibiting the read out of the end flag.

As an example of the above problem, the following is a move to end and stop program which does not work as expected because the status register is loaded in each loop.

```
                            MOVE

00             LAC        32   load status register
01             DTLA
```

```
02          DTEF            skip on error flag
03          JMP         00  loop
04          HLT             stop
```

In order to make the program work, the jump should be to location 02, rather than 00.

THE 6800 CONTROLLER

All the functions of the TC02 controller can be realized using a 6800 microprocessor and some hardware.

A program that performs the functions of the TC02 controller has been created and the hardware implemented to interface the microprocessor with the transporter.

The program has been divided into seven subprograms which implement the following functions: Stop, Move Forward and Reverse, Change Direction, Search Forward and Reverse, Read Forward and Reverse, Write Forward and Reverse and Formating.

Each function (subprogram) can be called independently of each other. For instance, Move Reverse, Change Direction, Search and Stop can be called to perform a search function. Before calling the search subprogram, it is necessary to load the WC (word counter) memory location, as it is done in the PDP - 9 minicomputer. The WC is in locations 01AF and 01B0 giving a 16 bit capacity or FFFF hexadecimal possible blocks count. Any valid method for loading the WC is allowed, but since it is 16 bit word, the LDA X and STA X (load and store the index register) instructions are preferable. They should be programmed before the call search command.

Each subprogram is called with the following instructions:

STOP	7E0000

MOVE REVERSE	BD0020
MOVE FORWARD	BD0027
CHANGE DIRECTION	BD0049
SEARCH	BD0071
READ	BD00A3
WRITE	BD00EB
FORMATING	0000

the formating subprogram was made separately for two reasons. First, there were not more memory available for more programs and second, its use is not as frequently as is the rest of the programs.

The stop subprogram can jump to a special error subroutine, which shows whether the stop is a correct action or is due to an error in the performance of any of the former programs. This is done, by starting the stop subprogram at two different locations. When there is no error, the first one loads a particular code in a memory location and initializes an optional display subroutine. The second location is started when an error occurs during the operation of any subprogram. A subroutine detects it, by checking the interrupt flag any time an interrupt occurs. Then a particular subprogram is started which loads the code for the optional display subroutine.

The stop subprogram stops the motor movement by placing a zero in the motor move line of the transporter. This signal is converted into a PDP - 9 standard -3V control signal by the circuit shown in fig. # 4 [9]

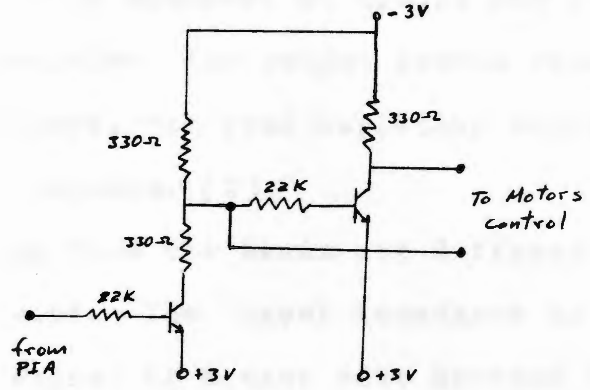


Fig. #4 TTL to PDP - 9 Logic Level Converter

The stop subprogram also loads the direction control line of the transporter with a forward code by placing a zero through a circuit similar to the stop signal circuit.

HARDWARE IMPLEMENTATION

The following is a description of the hardware designed to interface the 6800 Dectape controller with the PDP - 9 Transporter. First there is a brief discussion of the PDP - 9 read/write amplifier and the circuit designed to substitute it, after that, there is a discussion of the interface and registers.

READ AMPLIFIER

The TCO2 amplifier is a high gain differential amplifier with a positive feedback. When a signal of either polarity is sensed by the head, the read amplifier outputs switch immediately (see fig. #3). [2]

The read amplifier outputs U and V, are standard logic levels of -3V and ground. When input E is more positive than input D, V is asserted at ground and U is negative; when D is more positive, the output levels reverse. Because of positive feedback, the read amplifier oscillates in the absence of input signals. [2]

The inputs coming from the heads are differential signals, centered at ground. The input impedance is 400 ohms. A nominal input signal is a sine wave between 5 to 30 KHz., at 20 mV.

The Read/Write amplifier circuit of the TC02 Controller is shown in fig. #5. [2]

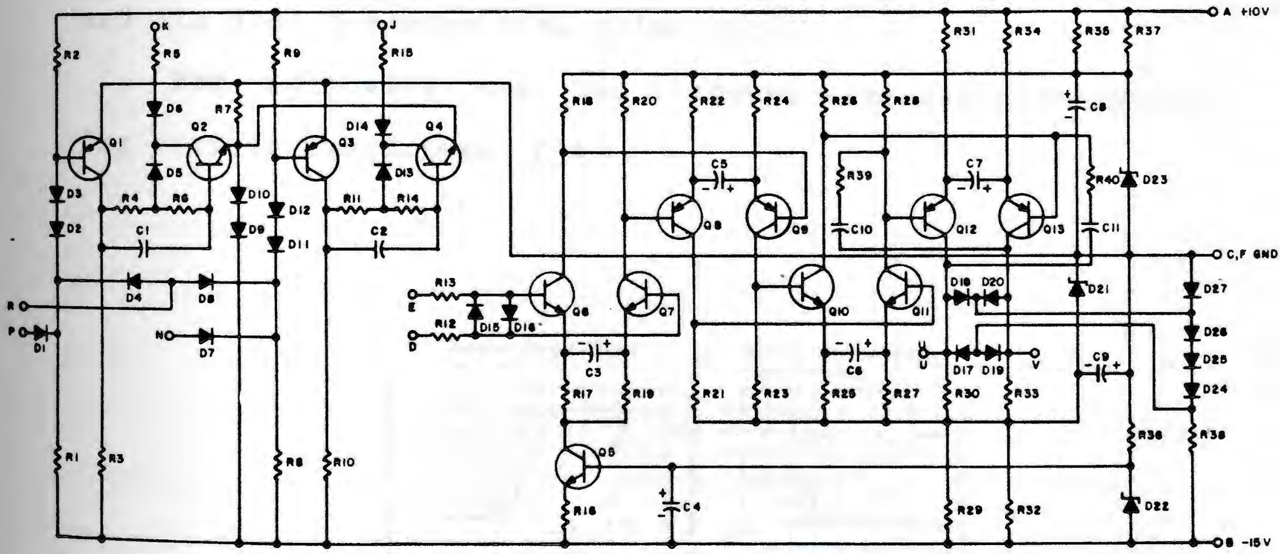


Fig. #5 TC02 read/write amplifier

A LM311 Voltage Comparator was used to perform the read operation in the 6800 controller. The following are its electrical specifications.

LM311 SPECIFICATIONS [4]

Total voltage supply	36V
Input offset voltage	2mV
Input offset current	6nA
Input bias current	100nA
Voltage gain	200V/mV
Response time	200ns
Saturation voltage	0.75V Vin ≤ -10mV, Iout = 50mA

The main advantages of the LM311 for this project are its compatible output with TTL logic gates, its high gain and its good response time (slew rate).

The following are the response time characteristics for various overdrives. [14]

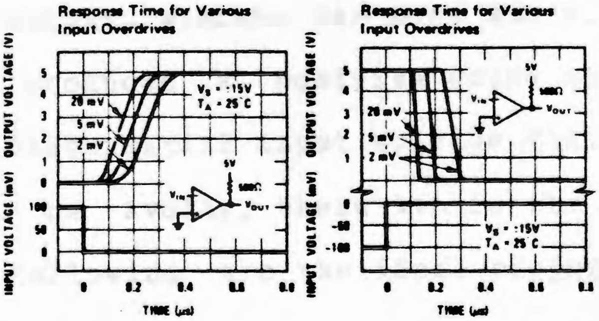


Fig. #6 LM311 response time

The following is a fast view of the operational amplifier and the comparator.

THE OPERATIONAL AMPLIFIER

An operational amplifier as shown in fig. #7 is a direct-coupled device with differential inputs and a single-ended output. [12,13,14] The amplifier responds only to the difference in voltage between the two input terminals, not to their common potential.

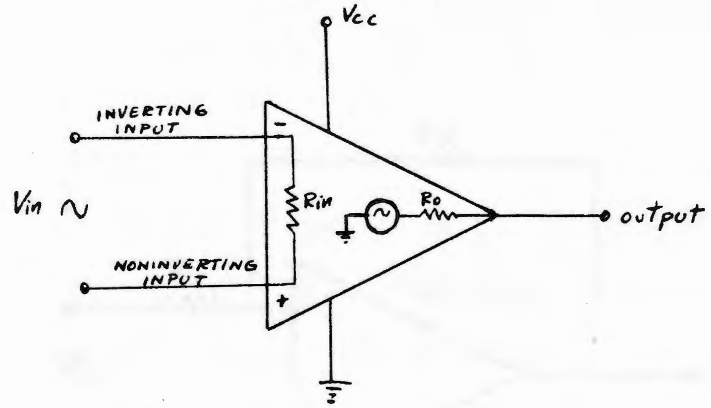


Fig. #7 basic Op Amp.

A positive going signal at the inverting input, while holding the other input at ground produces a negative going signal at the output, whereas the same signal at the non inverting input produces a positive going signal at the output. With a differential input voltage Ein, the output voltage Eo will be AvoEin, where Avo is the gain of the amplifier. The following are the ideal properties of the OpAmp. [12,13,14]

1- infinite voltage gain Avo

2- infinite input resistance

3- zero output resistance

4- infinite bandwidth

5- zero input offset voltage

From these properties a basic non inverting configuration is designed.

The fig. #8 shows a basic non inverting amplifier.

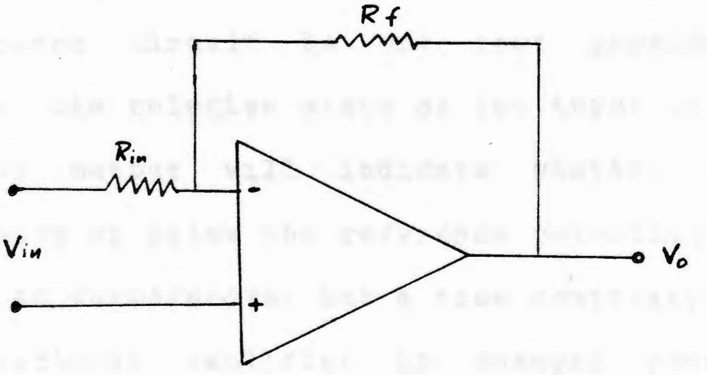


Fig. #8 non inverting OP Amp

Since Es=0 from property 1, the following relationships hold:

Iin = Ein/Rin

If = Iin = Eo/Rf

Ein/Rin = Eo/Rf

Where:

Iin = input current

If = feedback current

Ein = input voltage

Eo = output voltage

Rin = input resistance

Rf = feedback resistance

In terms of gain,

gain = Eo/Ein = Rf/Rin

Then, the gain can be varied by adjusting either Rf or Rin.

THE COMPARATOR [12,13,14]

A comparator circuit is one that provides an indication of the relative state of two input potentials. The comparator output will indicate whether the input signal is above or below the reference potential. Op Amps may be used as comparators, but a true comparator differs from an operational amplifier in several respects. A comparator has a slew rate as much as 100 times faster than that of an Op Amp. It is not frequency compensated and thus would probably be unstable if negative feedback were applied. the CMRR (Common Mode Rejection Ratio) and PSRR (Power Supply Rejection Ratio) are not always specified in comparators.

A basic comparator circuit is shown in fig. #9

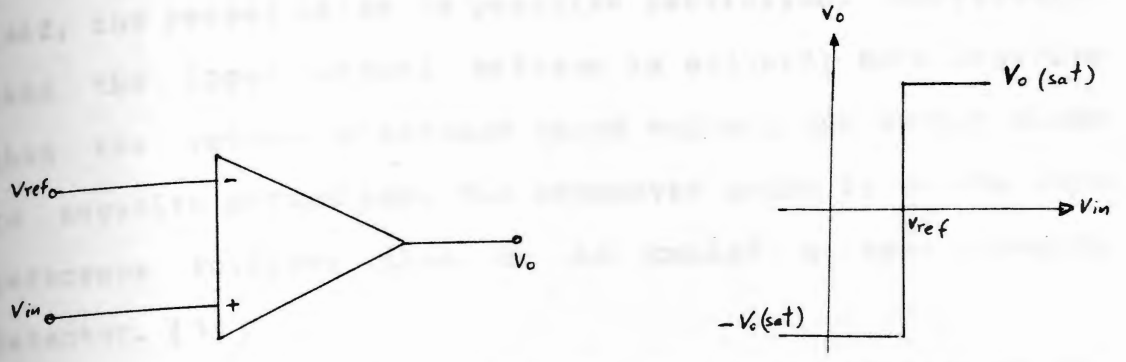


Fig. #9 basic comparator circuit

In this circuit the amplifier is operated in an open loop condition; therefore, the voltage difference required to change the output from one state to the other is quite small, essentially:

$$Eo(sat)-[- Eo\ (sat)]/Avo$$

Since this voltage is but a few hundred microvolts, the dominating factor that determines the exact threshold is the offset voltage of the amplifier, which may be as great as ±10 mV.

For this reason precision comparators should be nulled, so that when the output is zero, the input differential voltage will be as close to zero as practical. Furthermore, any source resistances in the input path should be selected so as to minimize the offset voltage.

ZERO CROSSING DETECTOR

A zero crossing detector is a comparator with the inverting lead grounded and the input signal applied to the noninverting lead. When the input voltage is slightly more positive than the zero reference voltage on the inverting lead, the output slews to positive saturation. Conversely, when the input signal voltage is slightly more negative than the reference voltage (zero volts), the output slews to negative saturation. The crossover point is at the zero reference voltage; thus it is called a zero crossing detector. [14]

Zero crossing detectors are subject to chatter at the crossing point. This usually occurs when a noise voltage is present on the signal. In fig. #10, a chatter situation is illustrated. [14]

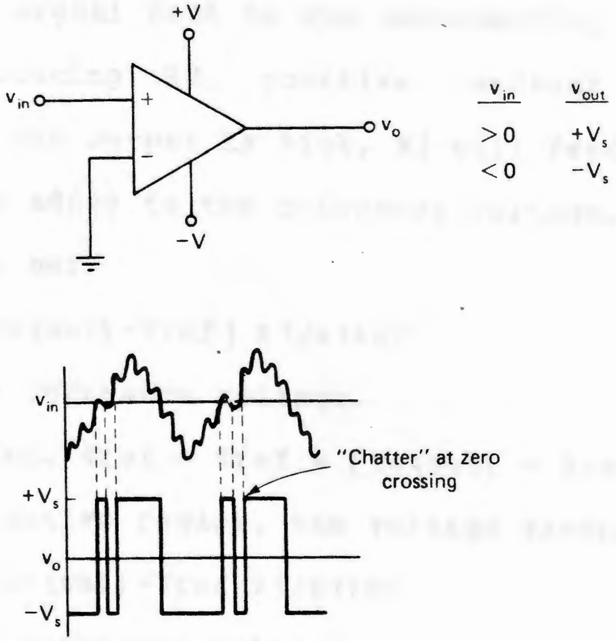


Fig. #10 chattering problem

In order to control the chattering, an hysteresis loop is introduced around the comparator as shown in fig. #11. [14]

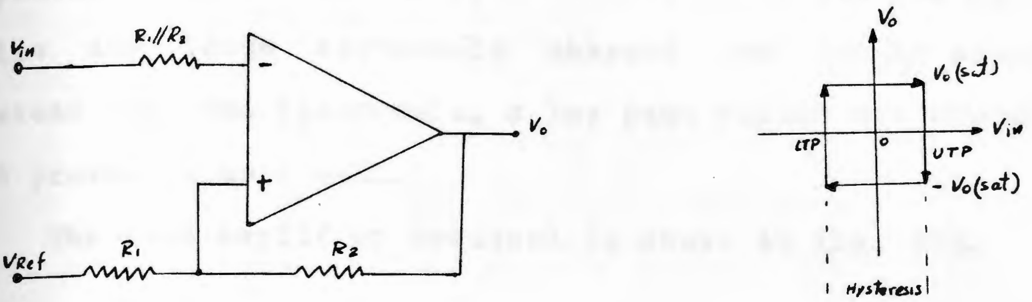


Fig. #11 Hysteresis

Positive feedback is introduced by feeding a portion of the output signal back to the noninverting input.

By introducing R2, positive feedback is developed across R1. If the output is high, R2 will feedback a signal which will be added to the reference voltage. This voltage increment will be:

Inc. Vref = [Eo(sat)-Vref] R1/R1+R2

making the new reference voltage.

UTP = Vref + Inc. Vref = Vref + [Eo(sat) - Vref]R1/R1+R2

In the negative region, the voltage feedback to R1 is:

Inc. Vref = [-Eo(sat)-Vref]R1/R1+R2

making the new reference voltage.

LTP = Vref+Inc.Vref = Vref+[-Eo(sat)-Vref]R1/R1+R2. [12]

Where:

UTP = Upper Threshold Point

LTP = Lower Threshold Point.

With these results, a comparator using the LM311 was designed. Positive feedback was used to perform hysteresis. The circuit was tested, but its performance was not as expected because little hysteresis did not get rid of the noise and large hysteresis changed the output signal. Instead of the hysteresis, a low pass filter was designed and proved to work well.

The read amplifier designed is shown in fig. #12.

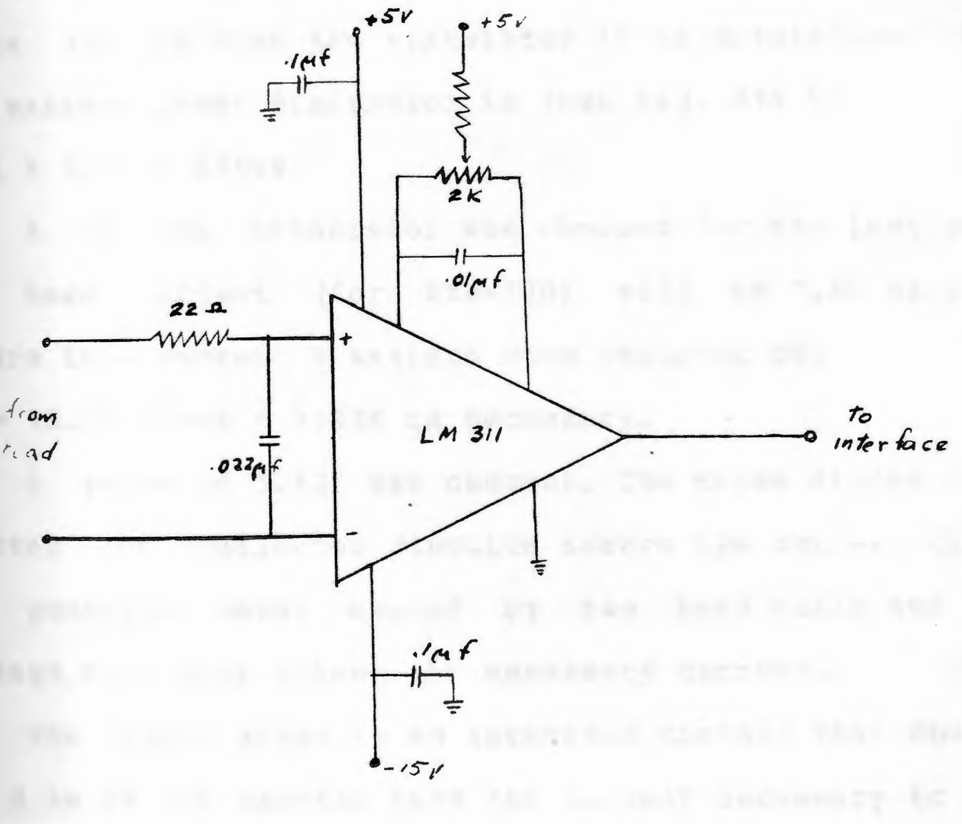


Fig. #12 Read Amplifier

WRITE AMPLIFIER

The write amplifier used in the TCO2 controller, is a high current gain amplifier which has ±3V standard PDP - 9 input voltage and a output of 0 to -15V with a 180 mA current capacity. (2)

A circuit was designed that produces the same output signal as that of the PDP - 9 write amplifier. The first stage works as an interface between the TTL and the PDP - 9 voltages and currents necessary to drive the write amplifier. The second stage is a current amplifier that sinks 180 mA when the transistor is in saturation. Then, the maximum power disipation is (see fig. #14):

90mA • 7.5V = 675mW.

A 1W NPN transistor was choosen for the last stage, the base current (for hfe=100) will be 1.80 mA and to assure this current a maximum base resistor of:

RB = 15.2V/1.8mA = 8.33K is necessary.

A value of 5.12K was choosen. The three diodes in the emitter and collector circuits assure the current flow in the positive sense needed by the head coils and fix a voltage drop that allows the nessesary current.

The first stage is an interface circuit that converts the 0 to 5V TTL signals into the current necessary to drive the write amplifier. A PNP transistor was used as shown in fig. #13. The power and current calculations are:

Ic1 = 3-(-15)V/7.5K = 2.5mA

Ic2 = 3-(-15)-1.4/1K= 16.6mA

Ic=Ic1+Ic2=16.6+2.5=19.1mA.

Ib(sat)=19.1/100=0.19mA.

Power=9•10=90mW.

Rb=3V/0.19mA=15K.

The fig. #13 shows the circuit designed for the first stage.

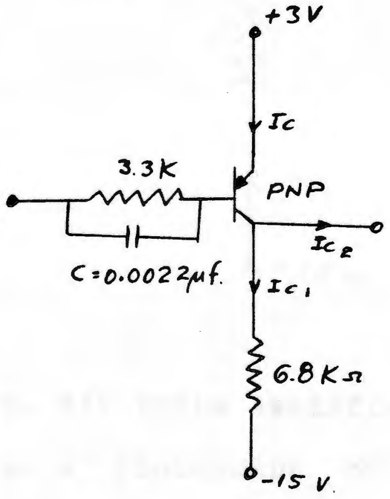


Fig. #13 First stage of the write amplifier

The write amplifier circuit were builded up and tested, working as expected. The circuit for the write amplifier is shown in fig. #14.

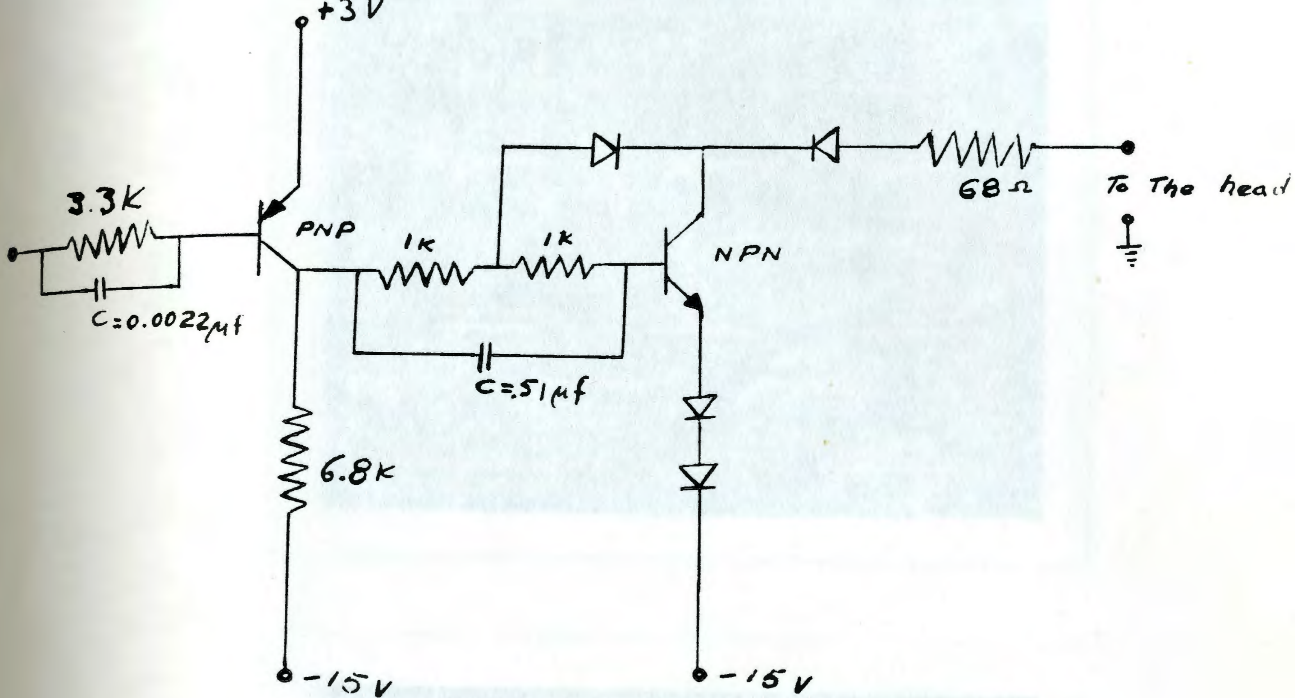


Fig. #14 Write Amplifier

Fig. #15 shows a photograph of the output signals coming from the TCO2 controller write amplifier and the signals coming from the designed circuit.

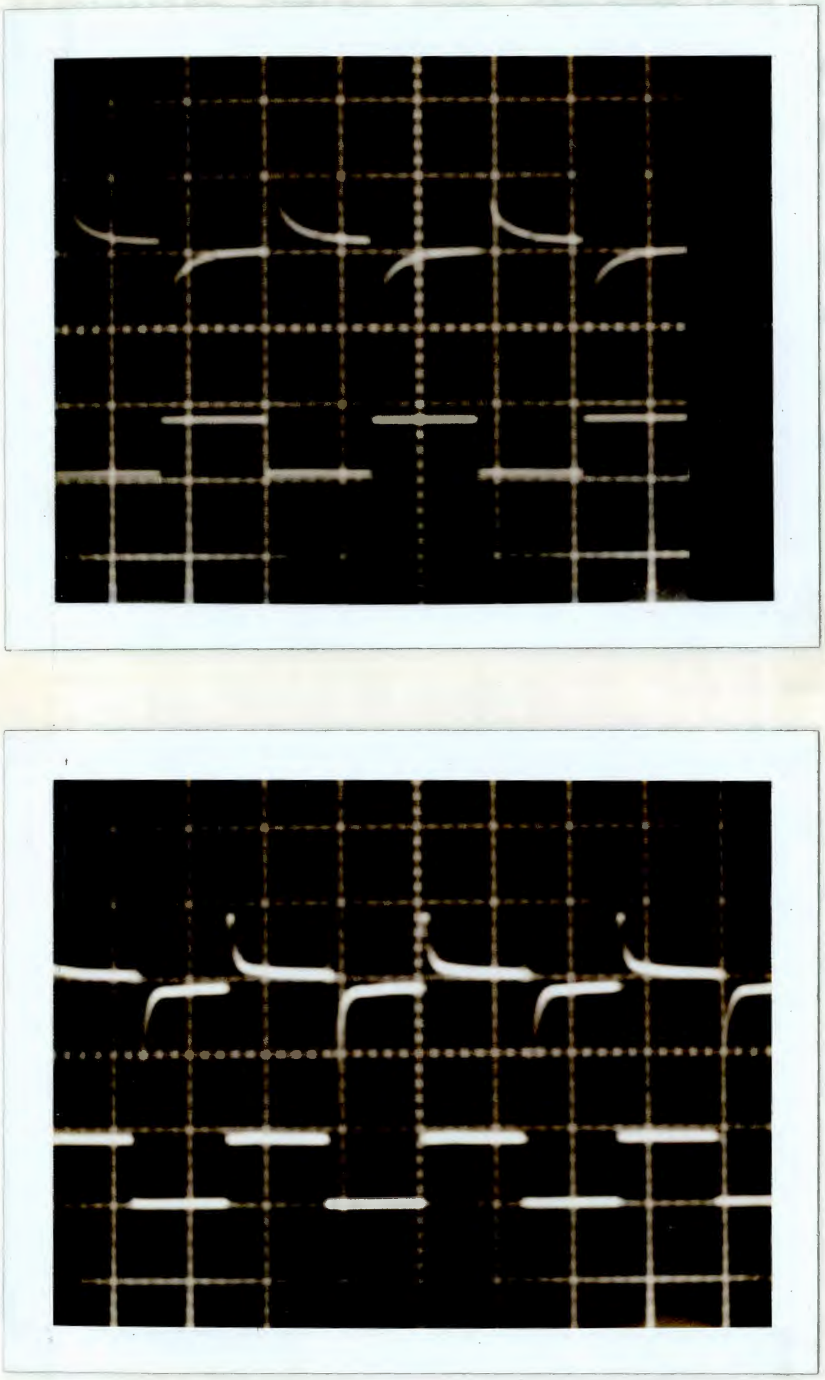


Fig. #15 pulses from the write amplifier

a) TC02 b) 6800

INTERFACE CIRCUIT

The information coming from the tape is a train of pulses that have to be loaded into the microprocessor memory synchronized with the time information and the mark codes also coming from the tape at the same time.

The read out of the timing pulses, is used to synchronize the entire system. Fig. #16 shows photograps of the timing pulse generated by the circuit designed in comparison with the time signal generated in the PDP - 9 controller.

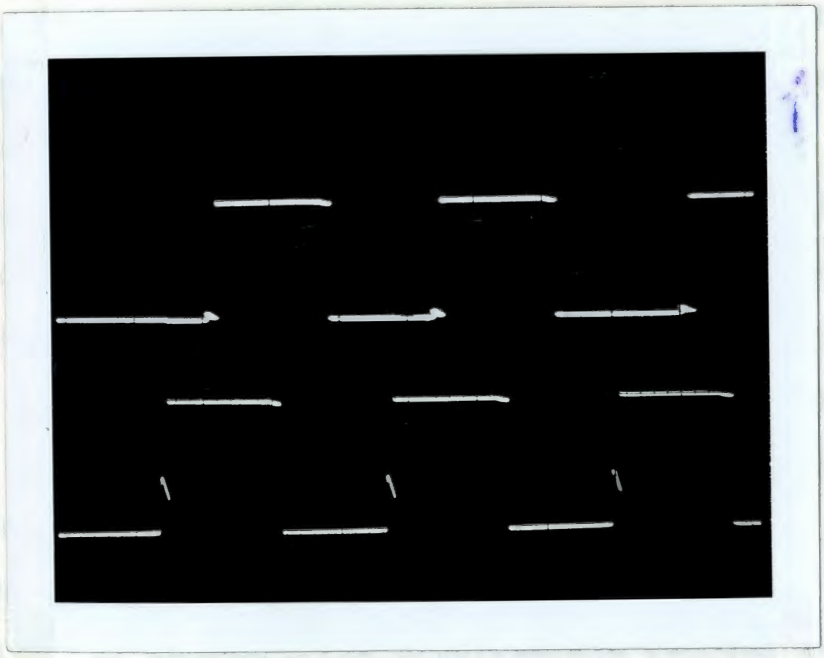


Fig. #16 a) PDP - 9 Timing

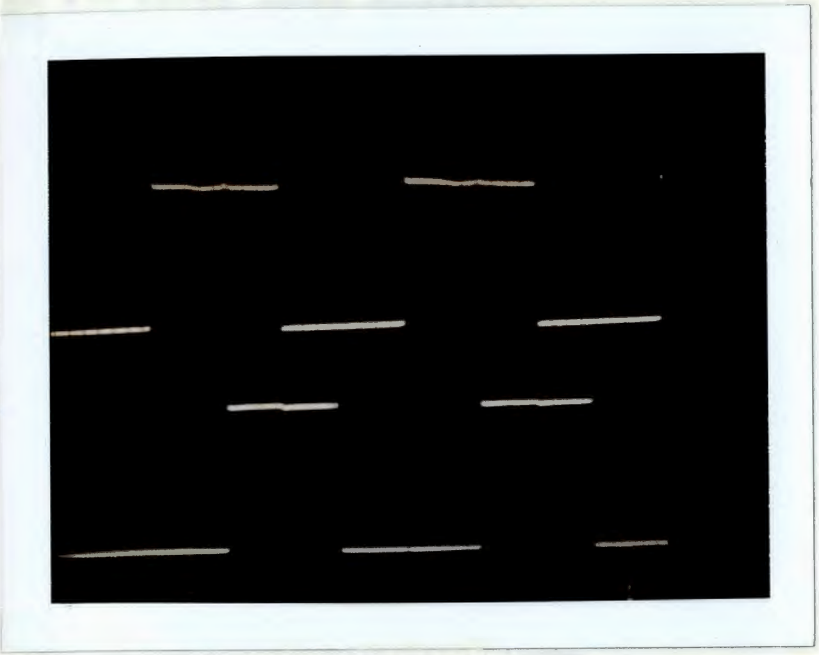


Fig. #16 b) 6800 Timing

The read out of the mark track is used to identify the codes written on the tape (see fig. #2). The mark track signal is connected to the serial input of a shift register (see fig. #21) which is shifted by the timing signal. The shift register is a SN74164 serial input parallel output TTL integrated circuit. Its parallel outputs are connected to two decoders, a fixed 22 decoder and a programmable decoder designed with exclusive or gates. This decoder is programmed by software through the A side of the PIA1 (Peripheral Interface Adapter 1). [9]

NOTE: The 22 decoder, the programmable decoder as well as the interface circuit for driving the brakes and motors of the TU55 were designed in a previous work by Allan Field. [9]

Fig. #17 shows these designs.

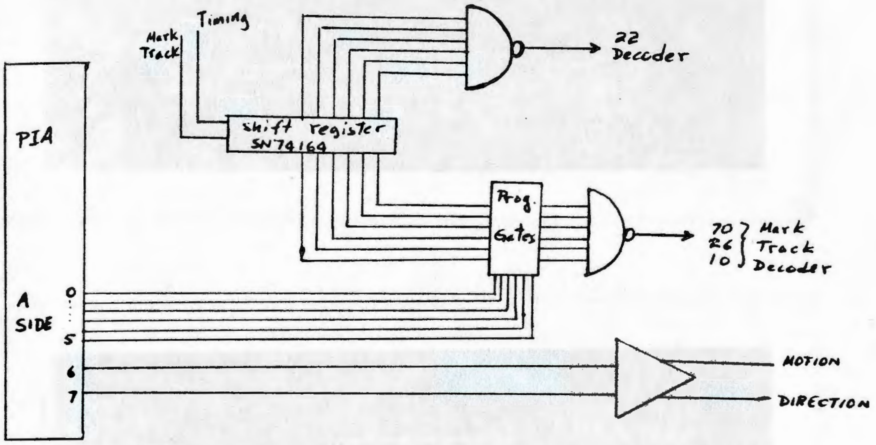


Fig. #17 Decoders and Interface

Fig. #18 shows photographs of the signals from the shift register that go to the programmable decoder. The pulses have been displayed using a Tektronix WR501 Word Recognizer and LA501 Logic Analyzer.

Fig. #18 a) 26 Mark Track Code

b) 70 Mark Track Code

Two PIA's are connected to the interface, the first one (PIA1) is the regular PIA coming with the MEK 6800 kit in address locations 8004 to 8007, the second (PIA2) has been assembled and located in addresses 4004 to 4007 (addresses 9004 to 9007 and 5004 to 5007 respectively, select the same PIA's, due to a partial decodification of address lines in the kit). The B side of both PIA's are used for reading and writing the 16 bit word into the read/write registers.

The interrupt lines of PIA1: CA1, CB1, CA2, and CB2, are connected to the 22 decoder output, the programmable decoder output, the read/write shift register control line and the write enable control respectively.

CA1 is an input and comes directly from the 22 decoder. It goes high each time that the tape reaches the end or beginning. This pulse causes an interrupt in the Move subprogram that makes the system jump to a Change Direction subroutine or to the Stop subprogram. It also stops and initializes the optional display subroutine, any time that the end or beginning ocurrs when the Search, Read or Write subprograms are running.

CB1 is an input and comes directly from the programmable decoder. This decoder is set up by the program, through lines 0 to 5 of the PIA1 side A, and decodes, 26 (block mark), 70 (word mark) and 10 (final mark). The 26 mark, is used to count the number of blocks in the search subprogram, whereas the 70 mark is used in the Read and Write subprograms to decide when to read or write a word.

CA2 is used as an output and goes to the read/write shift registers to control the parallel load.

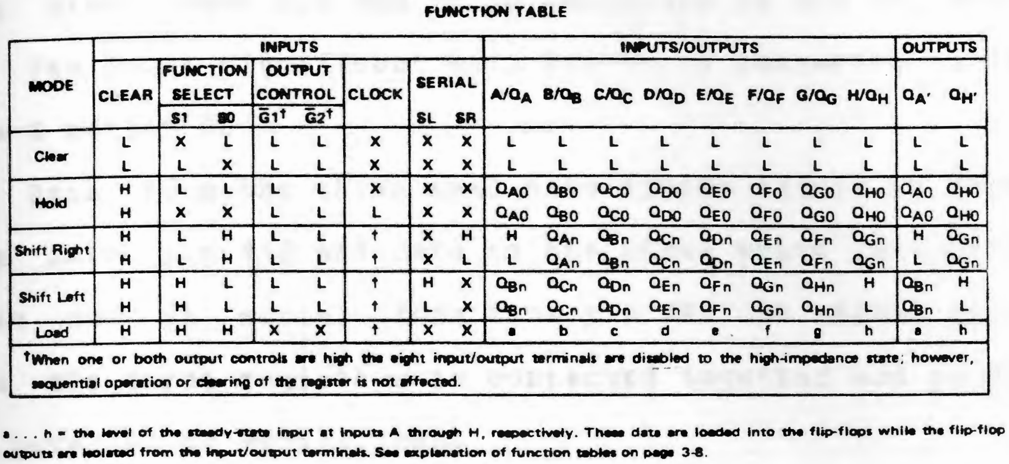
CB2 is used as write enable by the program and is high all the time that the write program is running. CB2 goes low and remains low in the rest of the programs, so inhibiting any posibility of undesired writing.

Output lines 0 to 5 from the PIA1 side A go to the programmable decoder which selects any of the mark track codes to produce an interrupt through the CB1 interrupt line.

Output lines 6 and 7 are connected through an interface to the motion and direction lines of the TU55 Transport respectively.

PIA2 was added to the MEK 6800 Kit and operates at addresses 4004 to 4007. The eight outputs from PIA1 side B and the eight outputs from PIA2 side B have been connected to the read/write shift registers to perform the read/write operations.

The read/write shift register used in the interface is the SN74S299 integrated circuit. Select lines S0 and S1 of this circuit control parallel loading and serial shift. When S1 is high and S0 low, the register shifts to the left, when S1 and S0 are both high, the register loads in parallel, synchronous with the clock pulse (timing from the tape). Then, connecting S1 to Vcc and S0 to CA2, the shift and parallel loading can be controlled by the program. Making the CA2 line high, the register loads in parallel whereas, making CA2 low, shifts left. The SN74S299 shift register and its function table is shown in fig. #19.



FUNCTION TABLE

MODE	INPUTS								INPUTS/OUTPUTS								OUTPUTS	
	CLEAR	FUNCTION SELECT		OUTPUT CONTROL		CLOCK	SERIAL		A/Q_A	B/Q_B	C/Q_C	D/Q_D	E/Q_E	F/Q_F	G/Q_G	H/Q_H	$Q_{A'}$	$Q_{H'}$
		S1	S0	$\overline{G}1$†	$\overline{G}2$†		SL	SR										
Clear	L	X	L	L	L	X	X	X	L	L	L	L	L	L	L	L	L	L
	L	L	X	L	L	X	X	X	L	L	L	L	L	L	L	L	L	L
Hold	H	L	L	L	L	X	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}	Q_{E0}	Q_{F0}	Q_{G0}	Q_{H0}	Q_{A0}	Q_{H0}
	H	X	X	L	L	L	X	X	Q_{A0}	Q_{B0}	Q_{C0}	Q_{D0}	Q_{E0}	Q_{F0}	Q_{G0}	Q_{H0}	Q_{A0}	Q_{H0}
Shift Right	H	L	H	L	L	↑	X	H	H	Q_{An}	Q_{Bn}	Q_{Cn}	Q_{Dn}	Q_{En}	Q_{Fn}	Q_{Gn}	H	Q_{Gn}
	H	L	H	L	L	↑	X	L	L	Q_{An}	Q_{Bn}	Q_{Cn}	Q_{Dn}	Q_{En}	Q_{Fn}	Q_{Gn}	L	Q_{Gn}
Shift Left	H	H	L	L	L	↑	H	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	Q_{En}	Q_{Fn}	Q_{Gn}	Q_{Hn}	H	Q_{Bn}	H
	H	H	L	L	L	↑	L	X	Q_{Bn}	Q_{Cn}	Q_{Dn}	Q_{En}	Q_{Fn}	Q_{Gn}	Q_{Hn}	L	Q_{Bn}	L
Load	H	H	H	X	X	↑	X	X	a	b	c	d	e	f	g	h	a	h

†When one or both output controls are high the eight input/output terminals are disabled to the high-impedance state; however, sequential operation or clearing of the register is not affected.

a . . . h = the level of the steady-state input at inputs A through H, respectively. These data are loaded into the flip-flops while the flip-flop outputs are isolated from the input/output terminals. See explanation of function tables on page 3-8.

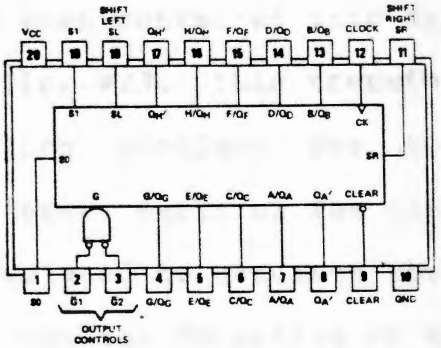


Fig. #19 Shift register SN74S299

In fig. #20 is shown the hardware designed and its connections with the 6800 microprocessor and with the TU55 Transport.

In Fig. #21 is shown the interface alone, while in fig. #22 is shown the Read/Write shift registers. The B side of both PIA's are connected in the order shown, that is, from bit 16 to bit 1. The PIA1 side B output 7 is the most significant bit and it is connected to bit 16, and so on. The least significant bit, bit 1, is connected to PIA2 side B output 0.

Data from the three read data tracks enters in serial form into pin #18 and data to the three write data tracks comes out in serial form from pin #8. SO signal pin #0 from the three registers is connected together and is used for parallel to serial load.

The timing pulses coming from the read out of the timing track, have been converted into narrow pulses by the circuit shown in fig. #23. This transformation is done to avoid synchronization problems due to simultaneous edge transitions in different parts of the circuit.

The circuit designed to generate the timing pulses and the word assembler for the formating of the tape(timing and mark tracks) is shown in fig. #24.

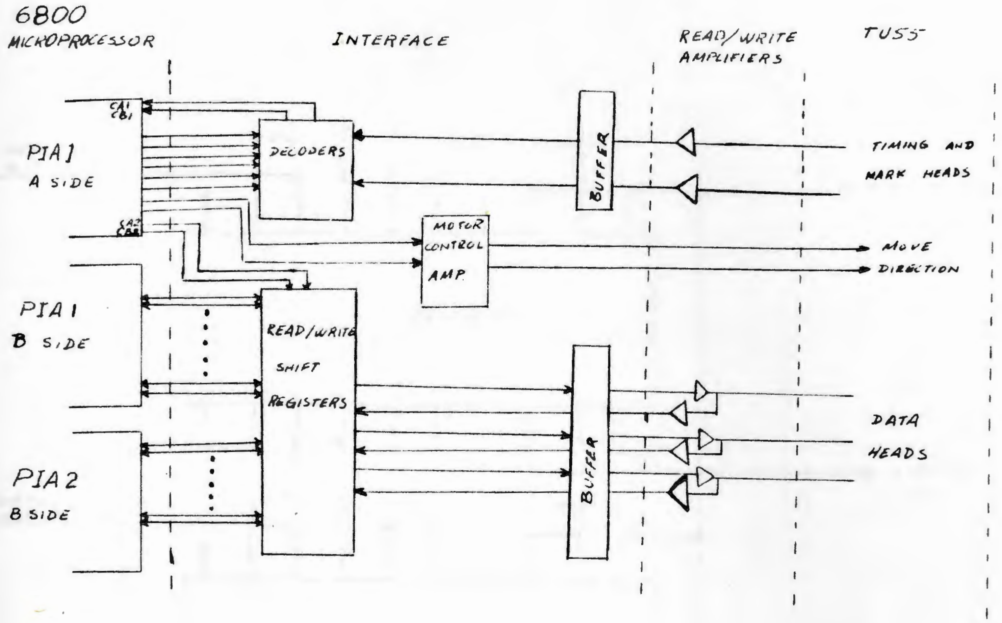


Fig. #20 6800 Controller Hardware

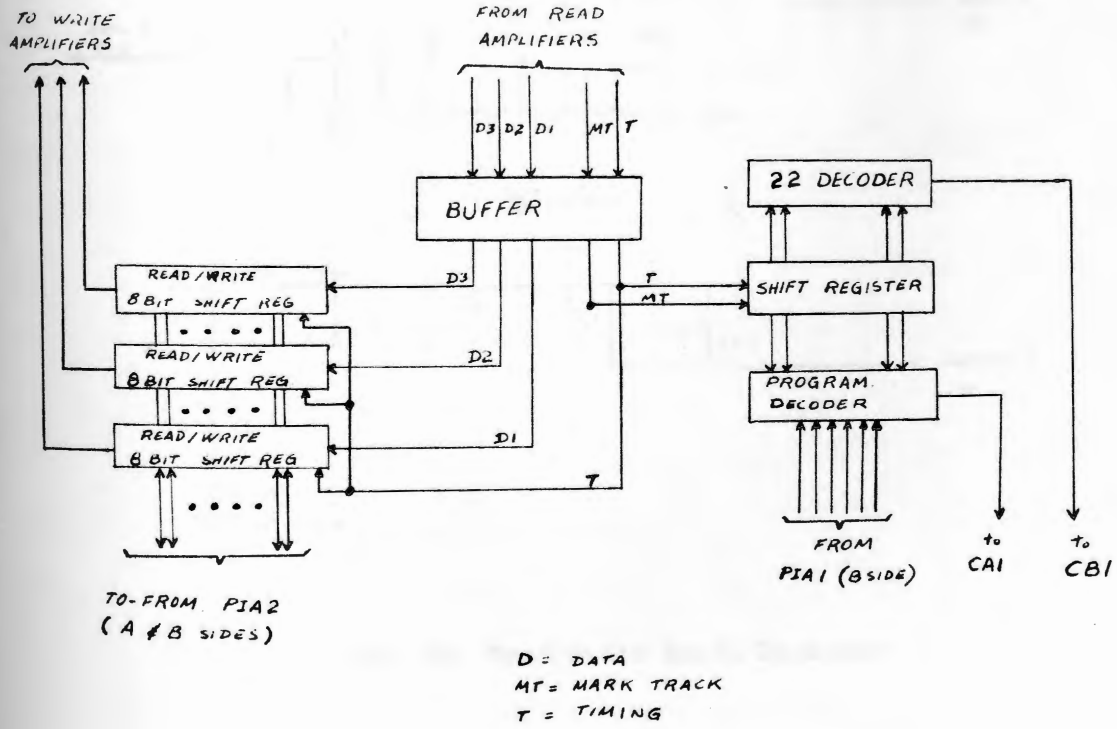


Fig. #21 6800 Interface

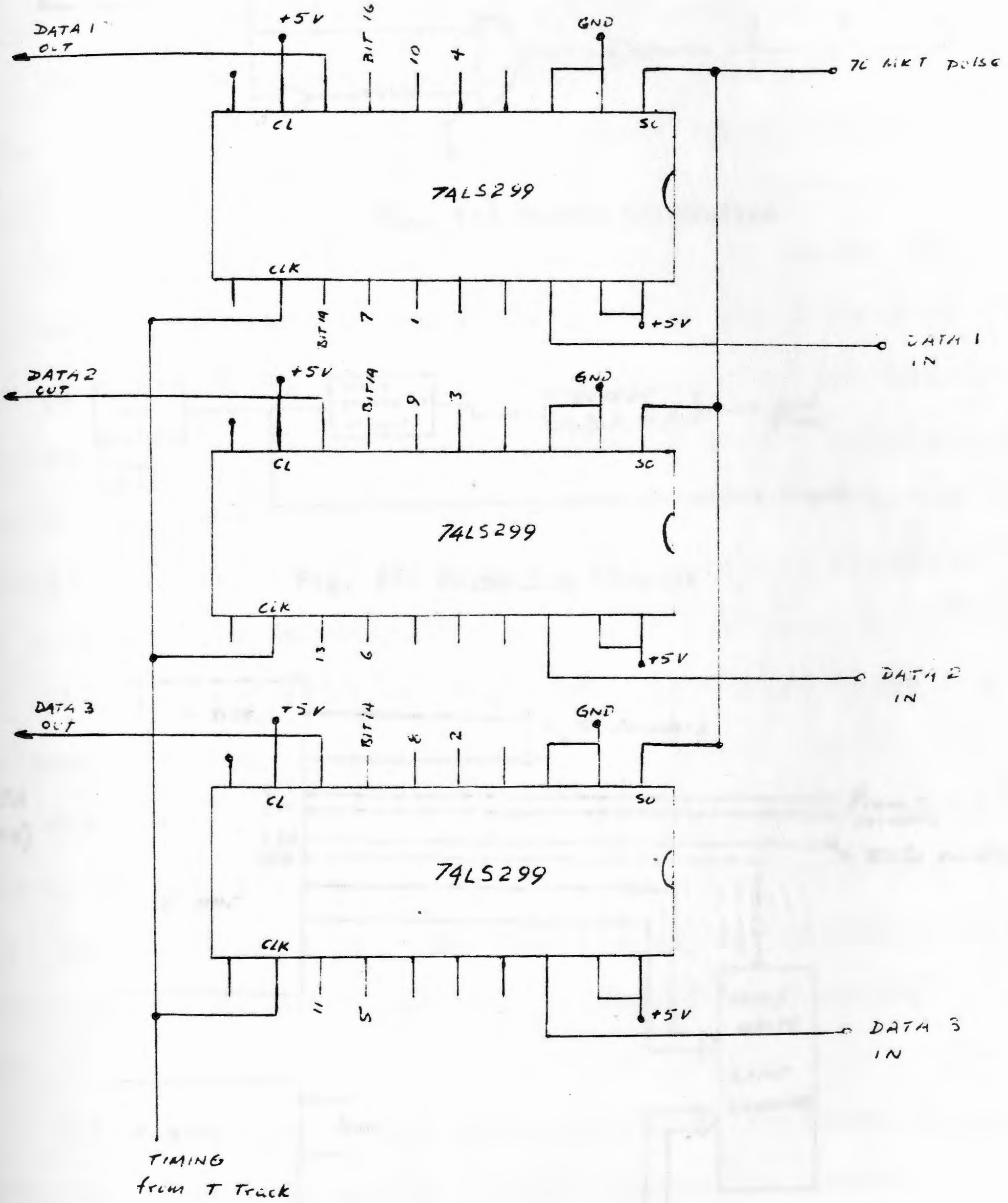


Fig. #22 Read/Write Shift Register

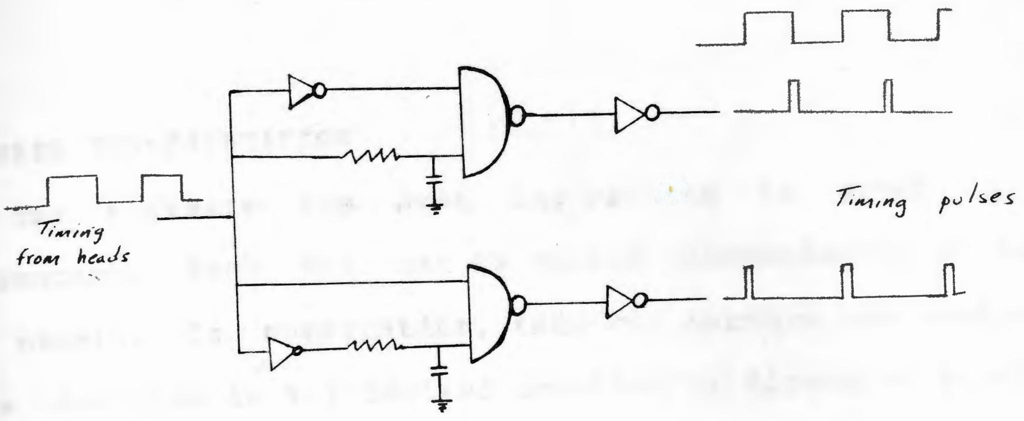


Fig. #23 Pulses Generation

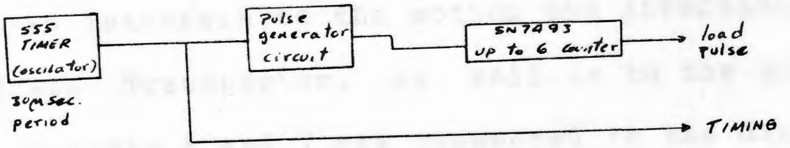


Fig. #24 Formating Circuit

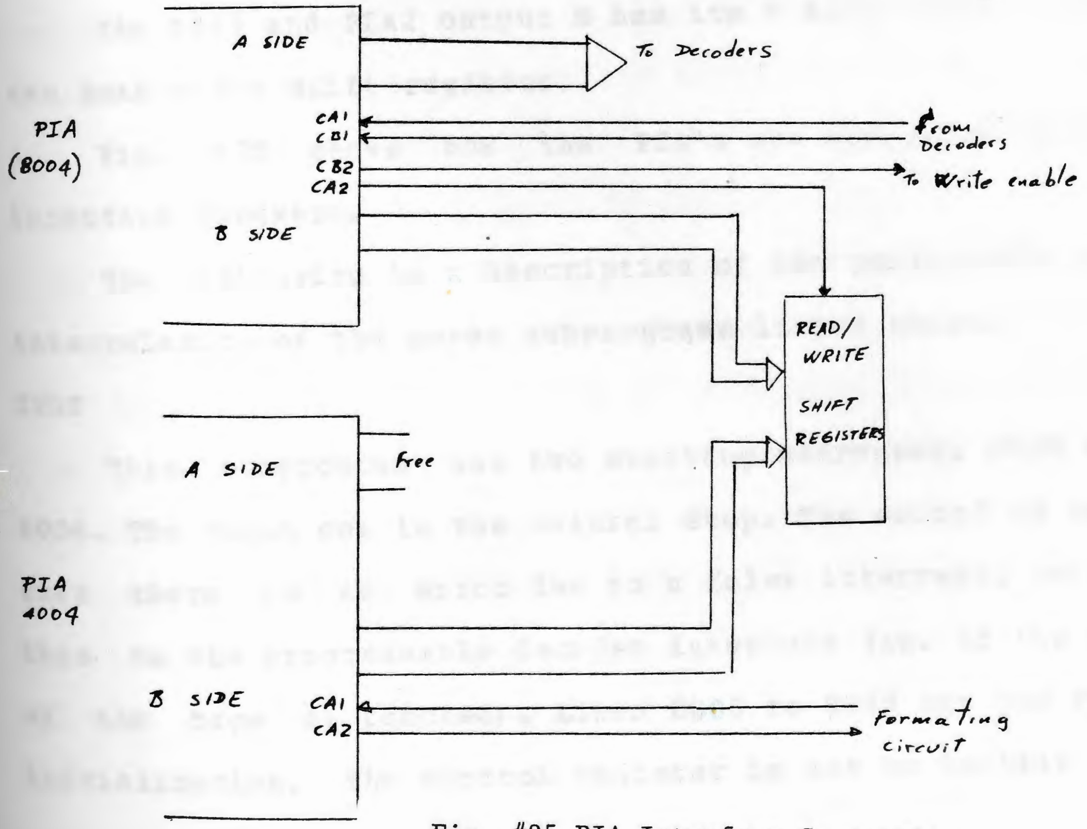


Fig. #25 PIA Interface Connections

SOFTWARE IMPLEMENTATION

The software has been implemented in seven subprograms. Each one can be called independently of all the others. In combination, they can perform the read or write function in any desired location or direction on the tape.

The interface between the software and the hardware is the PIA's, A and B outputs. PIA1 output A is connected through an interface to the motion and direction inputs of the Dectape Transporter, as well as to the programmable decoder. Outputs 6 and 7 are connected to the direction and motion inputs, while outputs 5 through 0 are connected to the programmable decoder.

The PIA1 and PIA2 output B has its 8 bits connected to the Read/Write shift register.

Fig. #25 shows how the PIA's are connected to the interface hardware.

The following is a description of the performance and interrelation of the seven subprograms listed below.

STOP

This subprogram has two starting addresses, 0000 and 0006. The first one is the natural stop. The second is used when there is an error due to a false interrupt, rather than to the programmable decoder interrupt (ie. if the end of the tape is reached). Lines 0006 to 0011 are the PIA1 initialization, the control register is set to inhibit the interrupt acknowledge, and the output register A is set to output pins 6 and 7. Pin 6 is connected (see fig. #25) to the motion input of the Dectape Transporter, and pin 7 is connected to the direction input of the Transporter.

Lines 0013 to 0015 perform an AND function to change only the motion bit to zero. This stops the tape, but does not change the direction, thus making the stop smooth in any direction. Lines 0019 and 001B erase the flags from the control register. Line 001D brings the control to an optional display subroutine that can display an error message if desired.

MOVE

This subprogram also has two starting addresses, 0020 and 0027, which correspond to Reverse motion and Forward motion, respectively. First, the accumulators are saved onto the stack; then either the reverse or forward codes are loaded into accumulator B. These codes make pin 6 equal to one and pin 7 equal to one or zero, depending on the direction. Lines 002C to 0037 show the PIA1 initialization. The control register A is set to inhibit the interrupt and to recognize a low to high transition in the interrupt line CA1. This corresponds to the output of the end/begin decoder in fig. #25. The output register A is set to output pins 6 and 7. Line 0039 stores the motion code(reverse or forward) into the output register A which starts the motion. Lines 003B and 003D clear the flags, while lines 003F to 0044 check for flag A set. Whenever the interrupt flag A is set (due to end or begin mark track), the accumulators are retrieved from the stack and the control is passed to the main program.

CHANGE DIRECTION

This subprogram changes the direction bit of the output register A without affecting the rest of the bits. It also allows a delay of 140 ms. in order to give the motors time to change direction.

The subprogram starts at line 0049, inhibits the interrupt acknowledge, then saves the accumulators onto the stack. Lines 004C to 0051 load the output register A and change the direction bit, no matter what the former direction was. Lines 0054 to 0058 cause a 140 ms. delay. Then the accumulators are retrieved from the stack and the control is returned to the main program.

GENERAL SUBROUTINE

This subroutine is common for the search, read and write subprograms. It is called any time an interrupt acknowledge is set. It checks whether the interrupt is due to the programmable decoder (flag B) or to an end/begin mark track (flag A). In case of a end/begin interrupt, a code for the optional display subroutine is set, (lines 0062 to 0064) and the stop subprogram is called. In case of a correct interrupt, the flags are cleared and the return from interrupt is called.

SEARCH

This subprogram looks for a specific block within the tape either in forward or reverse.

It first inhibits the interrupt acknowledge and saves accumulators onto the stack. Then lines 0074 to 0076 load the interrupt address 005D into the interrupt vector in the MC6800 system. Program Lines 0079 to 0086 set the PIA1 to enable the interrupt in case of a transition of the PIA1 input CB1 and set as output all the bits of the output register A and B. Lines 0088 to 0092 check the direction of the tape and load the code to recognize the block marks 26 or 45, depending on whether the motion is forward or reverse. Line 0094 loads the block counter address 01AF (WC word counter in the PDP - 9). Lines 0097 and 0099 clear the flags. Lines 009B to 009D wait for an interrupt and lines 009E to 009F count the interrupts. When the block counter reaches zero, it retrieves the accumulators from the stack and returns the control to the main program.

READ

This subprogram reads a specified number of words from the tape.

It starts by inhibiting the interrupt acknowledge and saving accumulators onto the stack. It then sets the PIA1 so that control register A enables interrupt for a positive transition of CA1 and control register B enables interrupt for a positive transition of CB1. The output registers are set so that output register A is an output register and output register B is an input register. In lines 00BC to 00C2, the mark track code 70 is set and stored in output register A outputs 0 to 5. In lines 00C4 and 00C6, the interrupt vector is loaded with the interrupt address 005D. In line 00C9, the current address is loaded with the current address location in 01B1 (CA in the PDP - 9). Lines 00CC and 00CF clear the flags, and lines 00D2 to 00D4 wait for interrupt. Then a delay is set, to wait for the word to be shifted to the end of the shift register(see fig. #25). Next the output register is loaded into the accumulator A and stored at the address specified by the current address vector. The current address vector is incremented by one and the word counter is decremented by one. Lines 00E0 to 00E3 check whether the last word is reached. If so, it then retrieves the accumulators and gives the control to the main program.

WRITE

This subprogram writes a specified number of words onto the tape in a desired block or blocks. It starts by inhibiting the interrupt acknowledge and saving accumulators. Then it sets the PIA to allow CA1 and CB1 interrupts by low to high transition and CA2 as an output line. In addition, the output registers A and B are all set for outputs. This is done in lines 00EB to 00FC. The interrupt vector is stored, and the current address pointer is loaded in line 0103. In lines 0106 to 010D, the mark track code 70 is set and stored in output register A outputs 0 to 5. In lines 0110 and 0112 a word is loaded into the output register B (see fig. #25). Lines 0117 and

011A pull CA2 and CB2 high. The first one allows the shift register to load in parallel when the next clock pulse goes high. The second is the write enable signal which allows the write circuit to write a word onto the tape. Lines 011D and 0120 clear flags, and lines 0123 to 0125 wait for interrupt. The interrupt is set by the programmable decoder, when the 70 mark track is in the shift register. Immediatly after the interrupt the CA2 line is set low, which allows the shift register to shift the word out in a series. Lines 012B and 012D cause a small delay, in order to allow the word to be shifted out of the shift register. Lines 0130 and 0131 increment the current address and decrement the word counter. Line 0132 checks whether it is the last word. If it is, the accumulators are retrieved from the stack, and the control is returned to the main program.

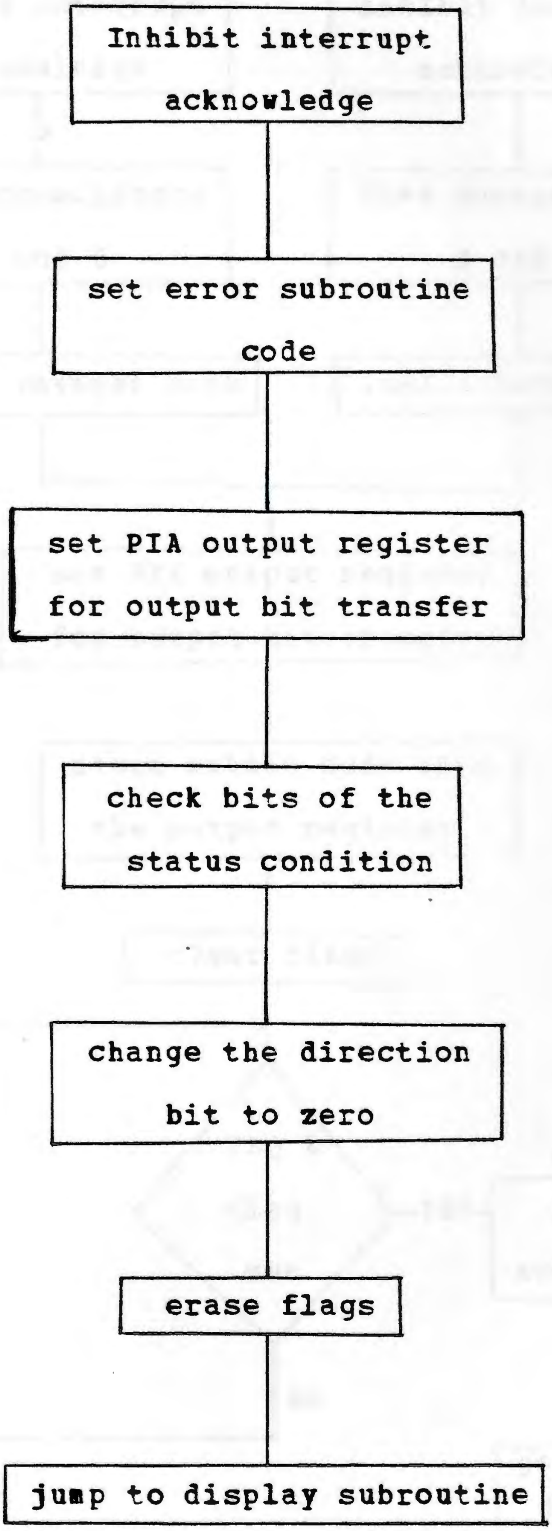
FORMATING

This subprogram formats the tape, writing the timing and mark tracks. It uses flag acknowledge mode instead of interrupts mode for detecting the hardware pulses. The flag is set by a positive pulse in the CA1 input in the PIA2. The CA1 input is driven by the circuit shown in fig. #24.

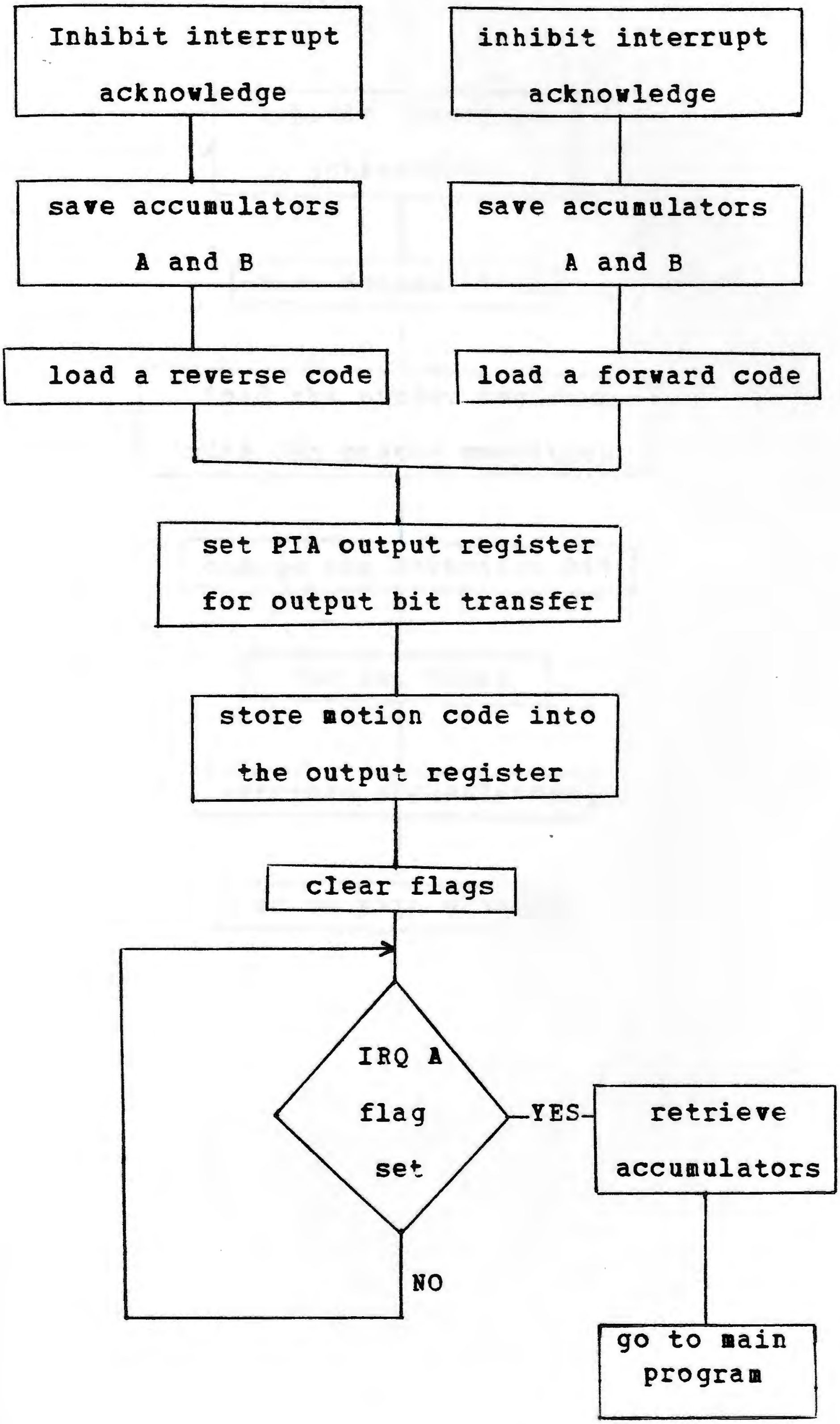
Lines 0017 to 001E set the 22 code and make a loop for ten 22 codes enough to ensure end code detection when reading the mark track. Lines 0036 to 003D set the 26 code and lines 004B to 0052 set the 70 code. Accumulator B is used as a counter for the number of words desired. The number of blocks is specified in memory location 01AD and is decremented in each loop between lines 0036 to 0064. The output line CA2 is used to enable the parallel load due to the circuit shown in fig. #24.

The following are the flow charts of the subprograms, as well as the 6800 machine language and assembler listing of the seven subprograms.

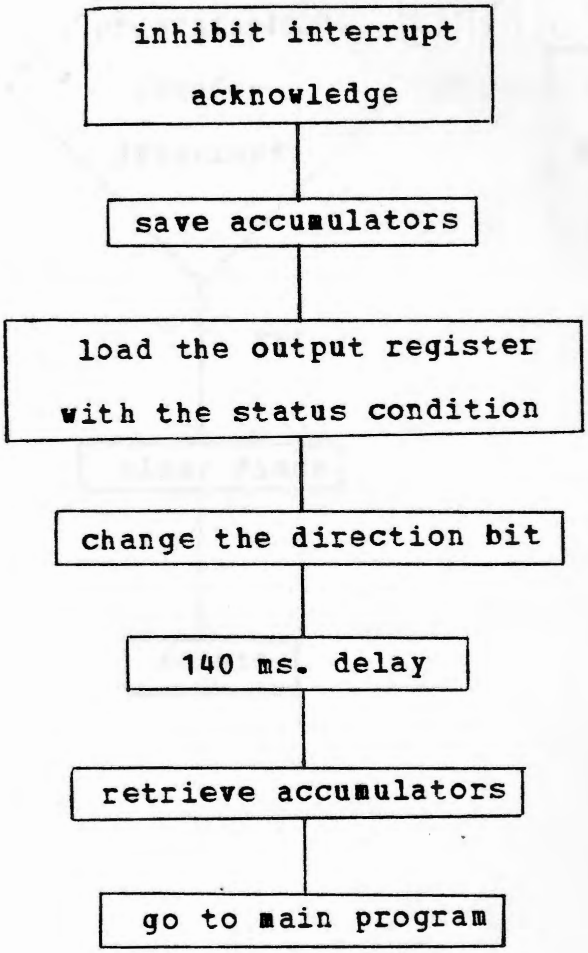
STOP


Inhibit interrupt acknowledge
set error subroutine code
set PIA output register for output bit transfer
check bits of the status condition
change the direction bit to zero
erase flags
jump to display subroutine

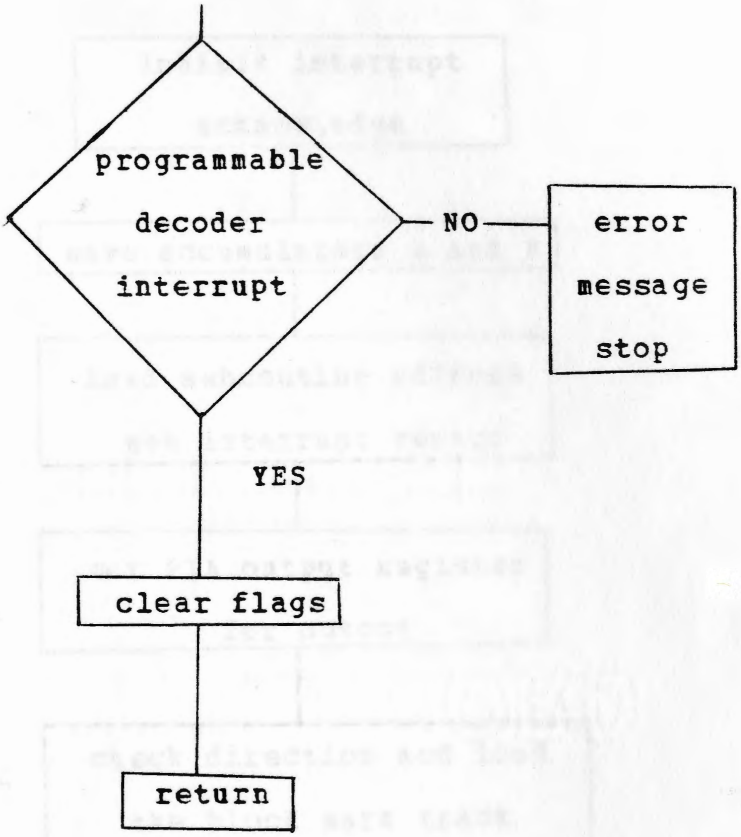
MOVE


Inhibit interrupt acknowledge
save accumulators A and B
load a reverse code
inhibit interrupt acknowledge
save accumulators A and B
load a forward code
set PIA output register for output bit transfer
store motion code into the output register
clear flags
IRQ A flag set
YES
NO
retrieve accumulators
go to main program

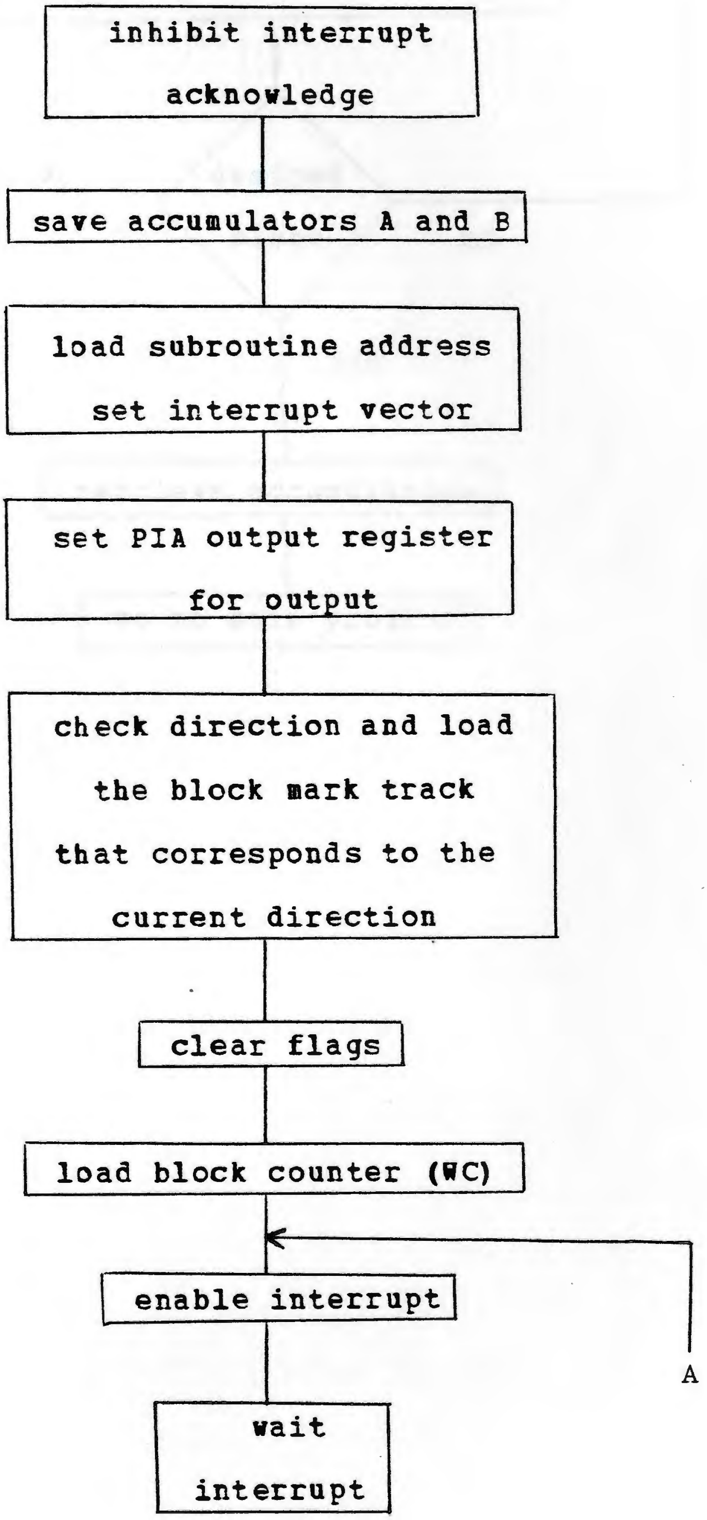
CHANGE

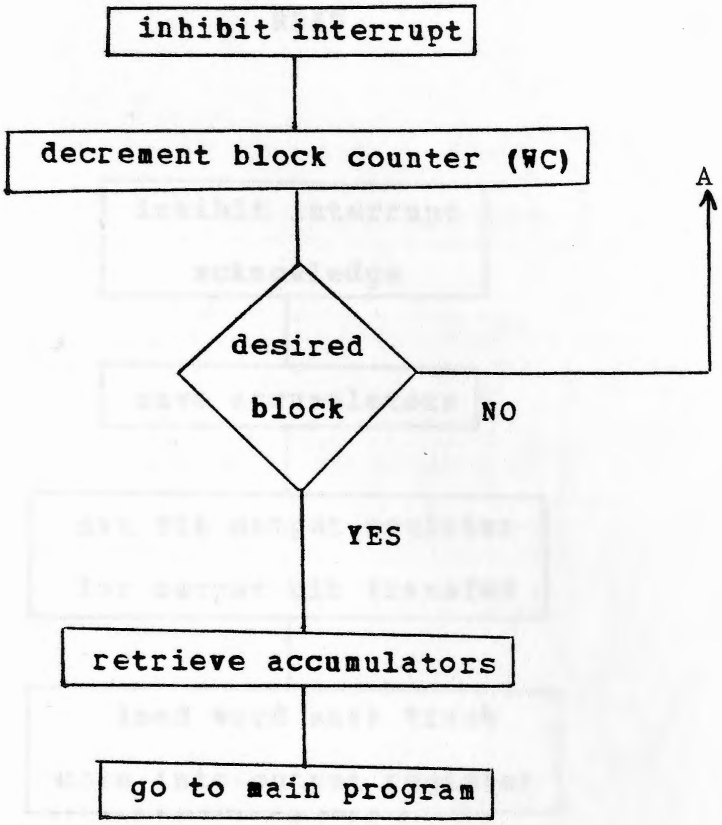

inhibit interrupt acknowledge
save accumulators
load the output register with the status condition
change the direction bit
140 ms. delay
retrieve accumulators
go to main program

GENERAL SUBROUTINE

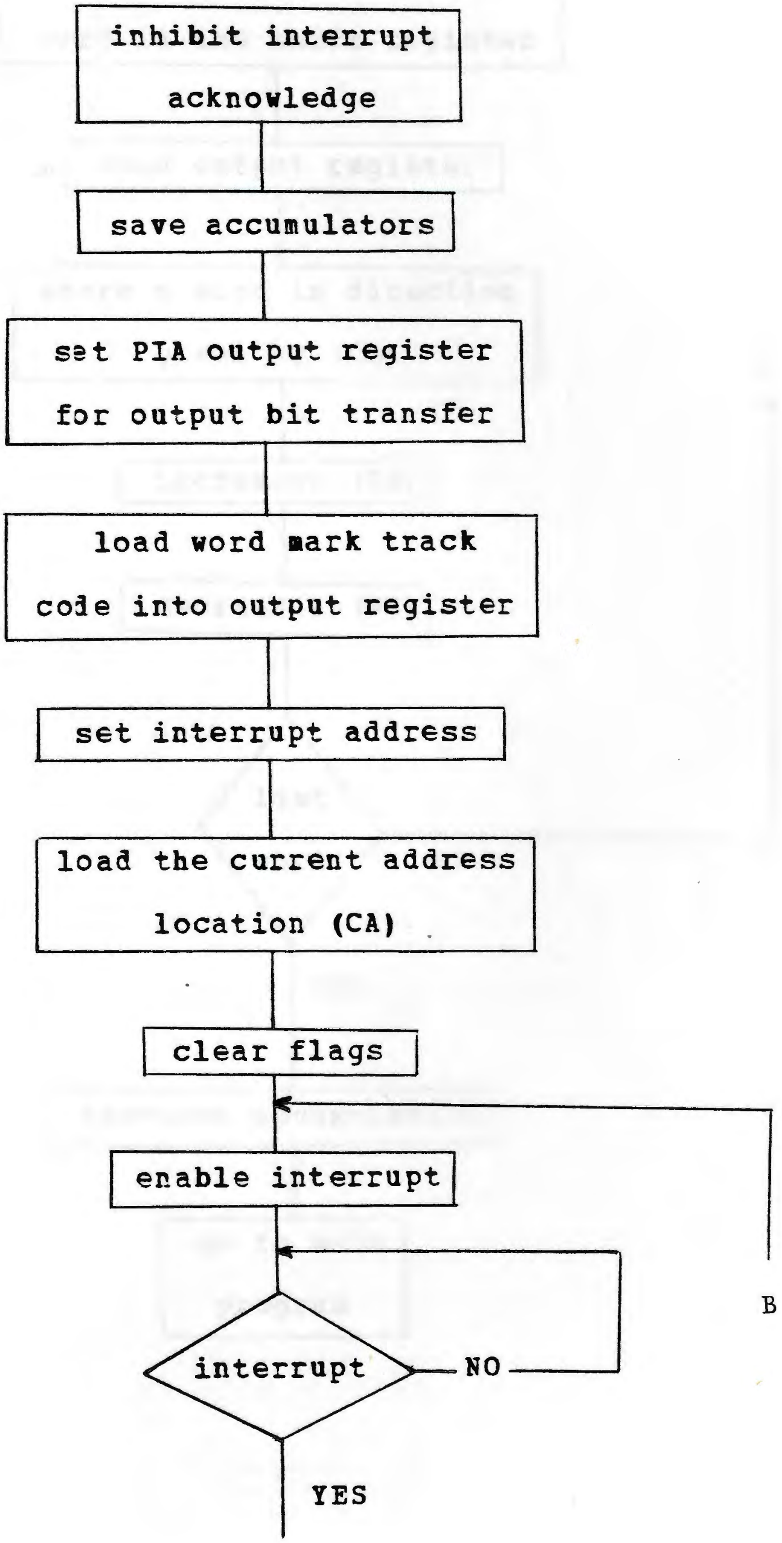


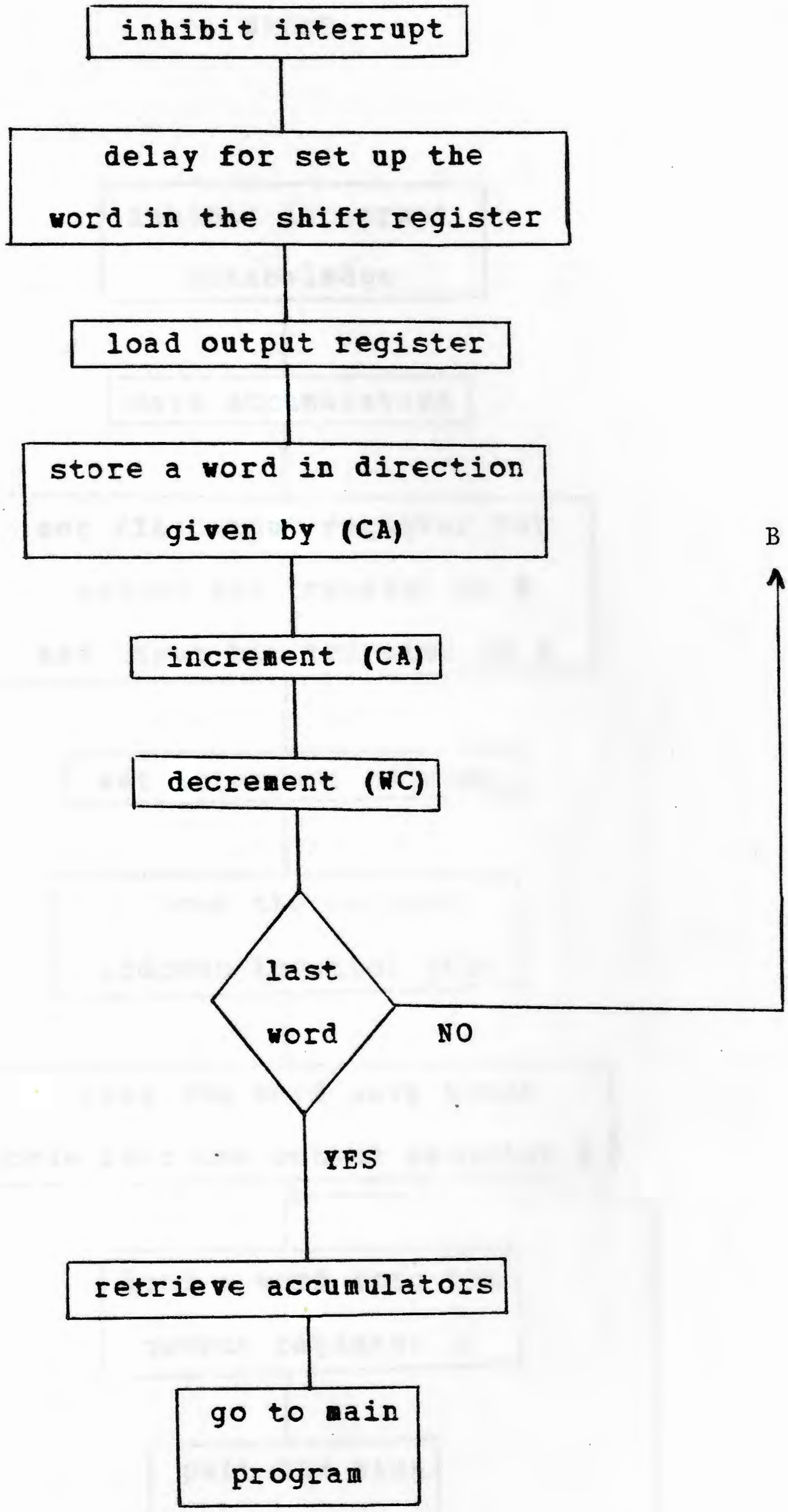
SEARCH


inhibit interrupt
acknowledge
save accumulators A and B
load subroutine address
set interrupt vector
set PIA output register
for output
check direction and load
the block mark track
that corresponds to the
current direction
clear flags
load block counter (WC)
enable interrupt
A
wait
interrupt

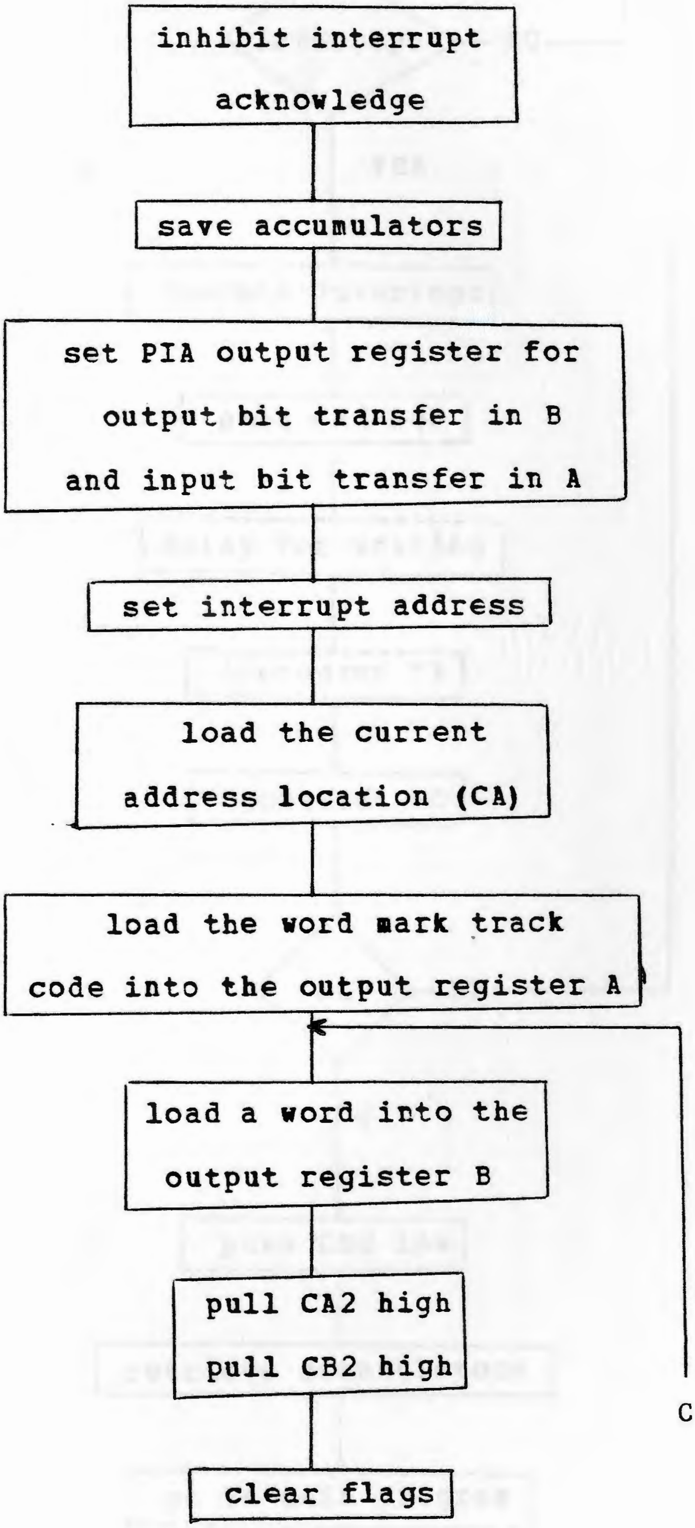

inhibit interrupt
decrement block counter (WC)
A
desired block
NO
YES
retrieve accumulators
go to main program

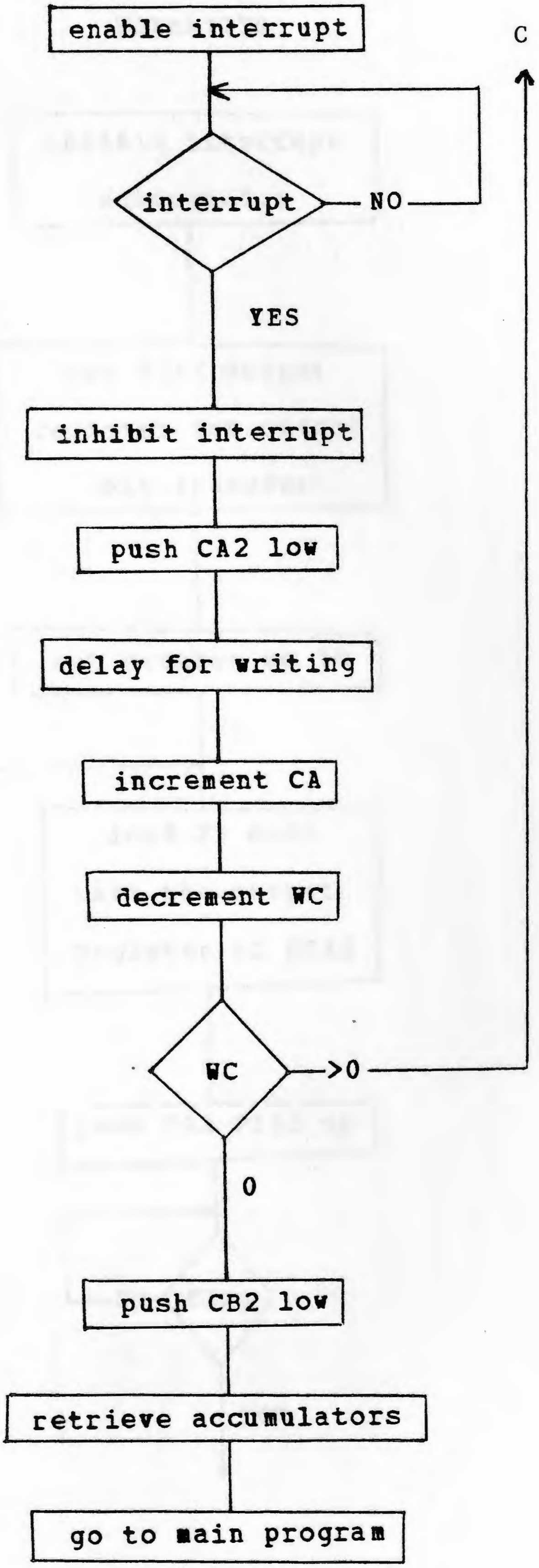
READ


inhibit interrupt acknowledge
save accumulators
set PIA output register for output bit transfer
load word mark track code into output register
set interrupt address
load the current address location (CA)
clear flags
enable interrupt
interrupt
NO
YES
B

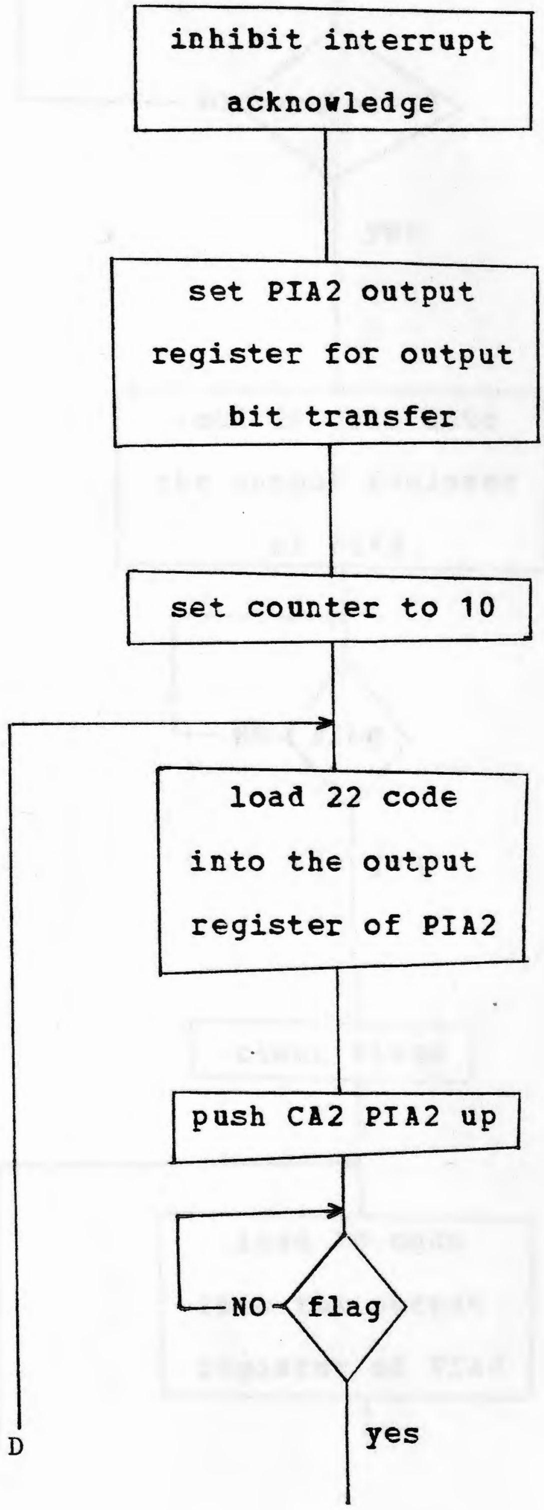

inhibit interrupt
delay for set up the word in the shift register
load output register
store a word in direction given by (CA)
increment (CA)
decrement (WC)
last word
NO
B
YES
retrieve accumulators
go to main program

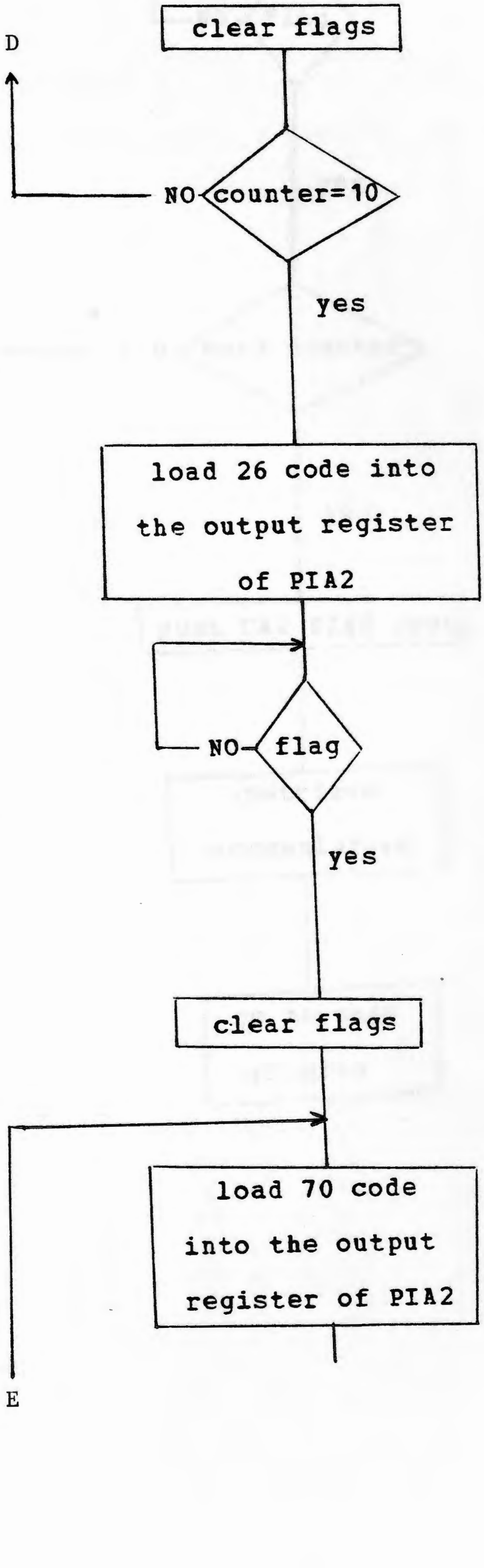
WRITE

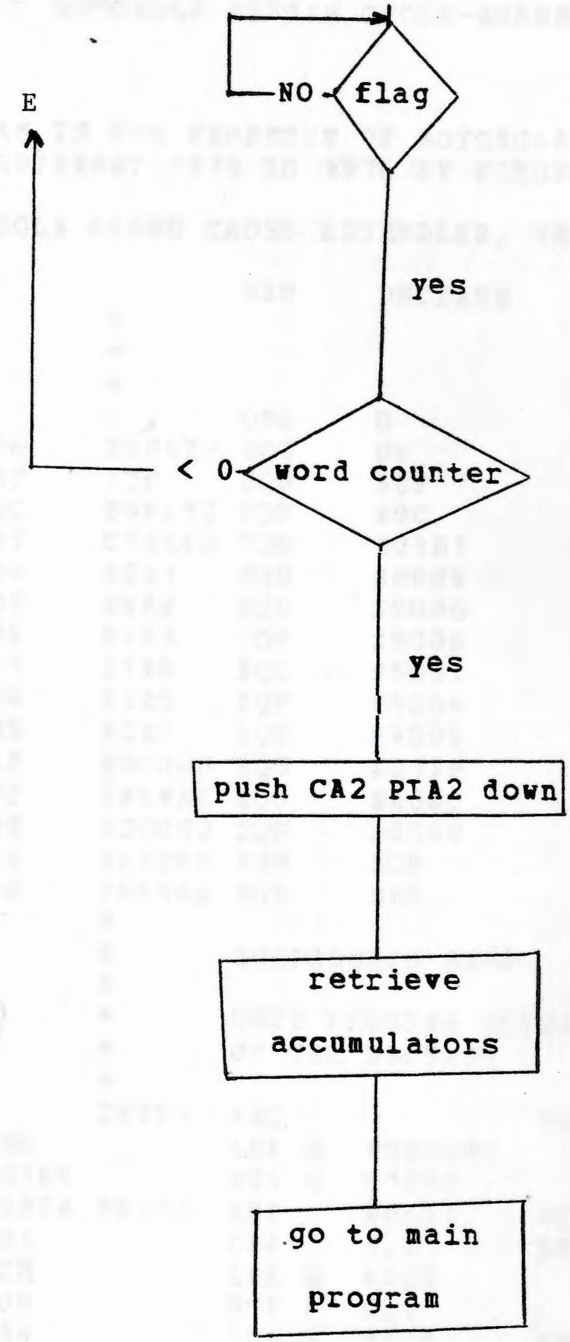

inhibit interrupt acknowledge
save accumulators
set PIA output register for output bit transfer in B and input bit transfer in A
set interrupt address
load the current address location (CA)
load the word mark track code into the output register A
load a word into the output register B
pull CA2 high
pull CB2 high
clear flags
C


enable interrupt
C
interrupt
NO
YES
inhibit interrupt
push CA2 low
delay for writing
increment CA
decrement WC
WC
>0
0
push CB2 low
retrieve accumulators
go to main program

FORMATING


inhibit interrupt acknowledge
set PIA2 output register for output bit transfer
set counter to 10
load 22 code into the output register of PIA2
push CA2 PIA2 up
NO
flag
yes
D


clear flags
D
NO
counter=10
yes
load 26 code into the output register of PIA2
NO
flag
yes
clear flags
load 70 code into the output register of PIA2
E



NO
flag
E
yes
< 0
word counter
yes
push CA2 PIA2 down
retrieve accumulators
go to main program

```
        MOTOROLA M68SAM CROSS-ASSMBLER


    M68SAM IS THE PROPERTY OF MOTOROLA SPD, INC.
       COPYRIGHT 1974 TO 1976 BY MOTOROLA INC

    MOTOROLA M6800 CROSS ASSEMBLER, RELEASE 1.3

                          NAM    DECTAPE
                   *
                   *
                   *
0000                      ORG    0
     0006  DSPLY1 EQU    06
     00BF  STP    EQU    $BF
     000C  DSPLY2 EQU    $0C
     01B1  CURADD EQU    $01B1
     8004  PIA1   EQU    $8004
     8005  PIA2   EQU    $8005
     8006  PIA3   EQU    $8006
     8007  PIA4   EQU    $8007
     4004  PIA5   EQU    $4004
     4005  PIA6   EQU    $4005
     01AF  WDCOUN EQU    $01AF
     A000  INTVEC EQU    $A000
     01B0  WDCOU2 EQU    $01B0
     00C0  REVERS EQU    $C0
     0040  FORWAR EQU    $40
                   *
                   *      SUBPROGRAM STOP
                   *
                   *      THIS PROGRAM STOPS THE MOTION
                   *      OF THE DECTAPE
                   *
0000 0F        INIT1  SEI                DISABLE IRQ
0001 86 06            LDA A  #DSPLY1
0003 B7 0184          STA A  0184H
0006 CE 8004   INIT2  LDX    #PIA1       PIA A SIDE  ADDRESS
0009 6F 01            CLR    1,X         ENABLE ACCESS TO DDRA
000B 86 C0            LDA A  #$C0
000D A7 00            STA A  0,X
000F 86 34            LDA A  #$34        ENABLE ACCESS TO
0011 A7 01            STA A  1,X         OUTPUT REGISTER
0013 86 BF            LDA A  #STP
0015 A4 00            AND A  0,X         STOP MOTION
0017 A7 00            STA A  0,X
0019 A6 00            LDA A  0,X         ERASE FLAGS
001B A6 02            LDA A  2,X         ERASE FLAGS
001D 7E 0179          JMP    0179H       JUMP TO DISPLAY SUBR.
```

```
             *
             *      SUBPROGRAM MOVE
             *
             *       THIS PROGRAM STARTS THE MOTION
             *        OF THE DECTAPE
             *
0020 0F             SEI               DISABLE INTERRUPT
0021 36             PSH A             SAVE ACCUMULATOR A
0022 37             PSH B             SAVE ACCUMULATOR A
0023 C6 C0          LDA B  #REVERS     LOAD REVERSE CODE
0025 20 05          BRA    PROG
0027 0F             SEI               DISABLE INTERRUPT
0028 36             PSH A             SAVE ACCUMULATOR A
0029 37             PSH B             SAVE ACCUMULATOR B
002A C6 40          LDA B  #FORWAR     LOAD FORWARD CODE
002C CE 8004 PROG   LDX    #PIA1      PIA INITIALIZATION
002F 6F 01          CLR    1,X
0031 86 C0          LDA A  #$C0
0033 A7 00          STA A  0,X
0035 86 34          LDA A  #$34
0037 A7 01          STA A  1,X
0039 E7 00          STA B  0,X        START MOTION
003B A6 00          LDA A  0,X        CLEAR FLAGS
003D A6 02          LDA A  2,X        CLEAR FLAGS
003F B6 8005 FLAG   LDA A  PIA2       LOAD CONTROL REGISTER
0042 84 80          AND A  #$80       OF PIA
0044 27 F9          BEQ    FLAG       CHECK FOR IRQ FLAG
0046 33             PUL B             RETRIEVE ACCUMULATOR B
0047 32             PUL A             RETRIEVE ACCUMULATOR A
0048 39             RTS               GO TO MAIN PROG.
             *
             *       SUBPROGRAM CHANGE
             *
             *       THIS PROGRAM CHANGE THE DIRECTION
             *        OF MOVEMENT OF THE TAPE
             *
0049 0F             SEI               DISABLE INTERRUPT
004A 36             PSH A             SAVE ACCUMULATOR A
004B 37             PSH B             SAVE ACCUMULATOR B
004C B6 8004        LDA A  PIA1       LOAD OUTPUT REGISTER
004F 88 80          EOR A  #$80       CHANGE DIRECTION BIT
0051 B7 8004        STA A  PIA1       SET THE NEW DIRECTION
0054 CE 4000        LDX    #$4000     LOAD DELAY
0057 09      DEL2   DEX               DECREMENT DELAY
0058 26 FD          BNE    DEL2
005A 33             PUL B             RETRIEVE ACCUMULATOR B
005B 32             PUL A             RETRIEVE ACCUMULATOR A
005C 39             RTS               GO TO MAIN PROG.
```

```
                 *
                 *     GENERAL SUBRROUTINE
                 *
005D 7D 8007 SUBRTN TST     PIA4      CHECK CB1 FLAG
0060 2B 08          BMI     OK        NO ERR.? GO TO END
0062 86 0C          LDA A   #DSPLY2    INITIALIZE DISPLY SUB.
0064 B7 01AC        STA A   $01AC
0067 7E 0006        JMP     INIT2     GO TO STOP SUBPROGRAM
006A B6 8004 OK     LDA A   PIA1      CLEAR FLAGS
006D B6 8006        LDA A   PIA3      CLEAR FLAGS
0070 3B             RTI               END OF SUBRROUTINE
                 *
                 *        SUBPROGRAM SEARCH
                 *
                 *    THIS PROGRAM SEARCHS  AN SPECIFIC BLOCK
                 *
0071 0F             SEI               DISABLE INTERRUPT
0072 36             PSH A             SAVE ACCUMULATOR A
0073 37             PSH B             SAVE ACCUMULATOR B
0074 CE 005D        LDX     #SUBRTN    LOAD INTERRUPT ADDR.
0077 FF A000        STX     INTVEC    STORE INTERRUPT VECTOR
007A CE 8004        LDX     #PIA1     LOAD PIA ADDRESS
007D 6F 01          CLR     1,X
007F 86 FF          LDA A   #$FF      ALL BITS OF DDRA ARE
0081 A7 00          STA A   0,X       OUTPUT
0083 86 35          LDA A   #$35
0085 A7 01          STA A   1,X       SELECT OUTPUT REG. A
0087 A7 03          STA A   3,X       ENABLE INTERRUPT (CB1)
0089 C6 69          LDA B   #$69      CODE FOR 26 MK. TRCK.
008B A6 00          LDA A   0,X
008D 84 80          AND A   #$80
008F 27 02          BEQ     GO
0091 C6 DA          LDA B   #$DA      CODE FOR 45 MK. TRCK.
0093 E7 00   GO     STA B   0,X
0095 FE 01AF        LDX     WDCOUN    LOAD WORD COUNTER
0098 A6 02   BLKCON LDA A   2,X       CLEAR FLAG
009A 0E             CLI               ENABLE INTERRUPT
009B 3E             WAI               WAIT FOR INTERRUPT
009C 0F             SEI               DISABLE INTERRUPT
009D 09             DEX               DECREMENT BLOCK COUNT.
009E 26 F8          BNE     BLKCON
00A0 33             PUL B             RETRIEVE ACCUMULATOR B
00A1 32             PUL A             RETRIEVE ACCUMULATOR A
00A2 39             RTS               GO TO MAIN PROG.
```

```
             *
             *      SUBPROGRAM  READ
             *
             *      THIS PROGRAM READS WORDS FROM THE TAPE
             *
00A3 0F             SEI                 DESABLE INTERRUPT
00A4 36             PSH A               SAVE ACCUMULATOR A
00A5 37             PSH B               SAVE ACCUMULATOR B
00A6 CE 4004        LDX    #PIA5        PIA STARTING ADDRESS
00A9 6F 02          CLR    2,X
00AB 6F 03          CLR    3,X
00AD 86 00          LDA A  #$0
00AF A7 01          STA A  1,X
00B1 A7 00          STA A  0,X
00B3 86 04          LDA A  #$04         OUTPUT REGISTER ENABLE
00B5 A7 03          STA A  3,X
00B7 A7 02          STA A  2,X
00B9 B6 8004        LDA A  PIA1
00BC 84 C0          AND A  #$C0         CODE FOR 70 MK. TRCK.
00BE 8A 07          ORA A  #$07
00C0 B7 8004        STA A  PIA1
00C3 CE 005D        LDX    #SUBRTN       LOAD INTERRUPT ADDR.
00C6 FF A000        STX    INTVEC       STORE INTERRUPT VECTOR
00C9 FE 01B1        LDX    CURADD       COURRENT ADDRESS
00CC B6 8006        LDA A  PIA3         CLEAR FLAG
00CF 0E      READ   CLI                 ENABLE INTERRUPT
00D0 3E             WAI                 WAIT FOR INTERRUPT
00D1 0F             SEI                 DESABLE INTERRUPT
00D2 86 01          LDA A  #$01         DELAY FOR READ
00D4 4A      DEL3   DEC A
00D5 26 FD          BNE    DEL3
00D7 B6 4004        LDA A  PIA5         READ A WORD
00DA A7 00          STA A  0,X          STORE A WORD
00DC B6 4005        LDA A  PIA6         READ SECOND WORD
00DF A7 01          STA A  1,X          STORE SECOND WORD
00E1 08             INX                 INC. CURRENT ADDRESS
00E2 08             INX
00E3 7A 01B0        DEC    WDCOU2       DECREMENT WORD COUNTER
00E6 26 E7          BNE    READ         READ ANOTHER WORD
00E8 33             PUL B               RETRIEVE ACCUMULATOR B
00E9 32             PUL A               RETRIEVE ACCUMULATOR A
00EA 39             RTS                 GO TO MAIN PROG.
```

```
              *
              *      SUBPROGRAM WRITE
              *
              *      THIS PROGRAM WRITES WORDS INTO THE TAPE
              *
00EB 0F              SEI                DESABLE INTERRUPT
00EC 36              PSH A              SAVE ACCUMULATOR A
00ED 37              PSH B              SAVE ACCUMULATOR B
00EE CE 4004         LDX    #PIA5       LOAD PIA START ADDRESS
00F1 6F 01           CLR    1,X
00F3 6F 03           CLR    3,X
00F5 86 FF           LDA A  #$FF        OUTPUT ALL DDR BITS
00F7 A7 00           STA A  0,X
00F9 A7 01           STA A  1,X
00FB 86 04           LDA A  #$04
00FD A7 02           STA A  2,X
00FF A7 03           STA A  3,X
0101 B6 8004         LDA A  PIA1
0104 84 C0           AND A  #$C0        CODE FOR 70 MK. TRCK.
0106 8A 07           ORA A  #$07
0108 B7 8004         STA A  PIA1
010B B6 8006         LDA A  PIA3        CLEAR FLAGS
010E CE 005D         LDX    #SUBRTN      LOAD INTERRUPT ADDR.
0111 FF A000         STX    INTVEC      STORE INTERRUPT VECTOR
0114 FE 01B1         LDX    CURADD      CURRENT ADDRESS POINT.
0117 B7 8007         STA A  PIA4        PULL CB2 HIGH (R SIG.)
011A A6 00    WRITE  LDA A  0,X         WRITE THE WORDS
011C B7 4004         STA A  PIA5
011F A6 01           LDA A  1,X
0121 B7 4005         STA A  PIA6
0124 86 3D           LDA A  #$3D
0126 B7 8005         STA A  PIA2        PULL CA2 HIGH
0129 B6 8006         LDA A  PIA3        CLEAR FLAG
012C 0E              CLI                ENABLE INTERRUPT
012D 3E              WAI                WAIT FOR INTERRUPT
012E 0F              SEI                DESABLE INTERRUPT
012F 86 35           LDA A  #$35
0131 B7 8005         STA A  PIA2        PUSH CA2 LOW
0134 86 01           LDA A  #$01
0136 4A       DEL4   DEC A              DELAY FOR WRITING
0137 26 FD           BNE    DEL4
0139 08              INX                INCREMENT CURRENT ADDR.
013A 08              INX
013B 7A 01B0         DEC    WDCOU2      DECREMENT WORD COUNTER
013E 26 DA           BNE    WRITE       CHECK WORD COUNTER
0140 86 35           LDA A  #$35
0142 B7 8007         STA A  PIA4        PUSH CB2 LOW
0145 33              PUL B              RETRIEVE ACCUMULATOR B
0146 32              PUL A              RETRIEVE ACCUMULATOR A
0147 39              RTS                GO TO MAIN PROGRAM
                     END
```

```
        MOTOROLA M68SAM CROSS-ASSMBLER


      M68SAM IS THE PROPERTY OF MOTOROLA SPD, INC.
         COPYRIGHT 1974 TO 1976 BY MOTOROLA INC

      MOTOROLA M6800 CROSS ASSEMBLER, RELEASE 1.3

                              NAM   FORMAT
                   *
                   *
                   *
0000                          ORG   $0
     4004         PIA5        EQU   $4004
     01AD         BLOKS       EQU   $01AD
     8006         PIA3        EQU   $8006
0000 36                       PSH A
0001 37                       PSH B
0002 CE 4004                  LDX   #PIA5
0005 6F 02                    CLR   2,X
0007 6F 03                    CLR   3,X
0009 86 FF                    LDA A #$FF
000B A7 00                    STA A 0,X
000D A7 01                    STA A 1,X
000F 86 34                    LDA A #$34     DIRECTION REG. CA2 LOW
0011 A7 02                    STA A 2,X
0013 A7 03                    STA A 3,X
0015 C6 0A        END1        LDA B #$0A
0017 86 10        OTRA        LDA A #$10     22 MK. TRK. CODE
0019 B7 8006                  STA A PIA3
001C 86 08                    LDA A #$08
001E A7 01                    STA A 1,X
0020 A6 00                    LDA A 0,X      CLEAR FLAGS
0022 A6 01                    LDA A 1,X
0024 A6 02        FLAG2       LDA A 2,X
0026 84 80                    AND A #$80     FLAG A
0028 27 FA                    BEQ   FLAG2
002A 86 3D                    LDA A #$3D     PUSH CA2 UP
002C A7 02                    STA A 2,X
002E 5A                       DEC B
002F 26 E6                    BNE   OTRA
0031 7D 01AD                  TST   BLOKS
0034 27 33                    BEQ   FIN
0036 86 10        BLK         LDA A #$10     26 MK. TRK. CODE
0038 B7 8006                  STA A PIA3
003B 86 48                    LDA A #$48
003D A7 01                    STA A 1,X
003F A6 02        FLAG3       LDA A 2,X
0041 84 80                    AND A #$80
0043 27 FA                    BEQ   FLAG3
0045 A6 00                    LDA A 0,X      CLEAR FLAGS
0047 A6 01                    LDA A 1,X
0049 C6 FF                    LDA B #$FF
004B 86 92        OTRA2       LDA A #$92     70 MK. TRK. CODE
004D B7 8006                  STA A PIA3
```

```
0050 86 00              LDA  A   #$00
0052 A7 01              STA  A   1,X
0054 A6 02     FLAG4    LDA  A   2,X
0056 84 80              AND  A   #$80
0058 27 FA              BEQ      FLAG4
005A A6 00              LDA  A   0,X       CLEAR FLAGS
005C A6 01              LDA  A   1,X
005E 5A                 DEC  B
005F 26 EA              BNE      OTRA2
0061 7A 01AD            DEC      BLOKS
0064 26 D0              BNE      BLK
0066 7E 0015            JMP      END1
0069 86 34     FIN      LDA  A   #$34      PUSH CA2 DOWN
006B A7 02              STA  A   2,X
006D 33                 PUL  B
006E 32                 PUL  A
006F 39                 RTS
                        END
```

DISCUSSION

In the present thesis several difficulties were encountered. The solution of the most of them made possible the accomplishment of the work in its present state.

The first big difficulty was related with the noise in the system. The signal from the tape is a 20mV wave with a very low signal to noise ratio. This caused a false triggering of the read-comparator, resulting in an erroneous reading from the tape. After several tries, using hysteresis and filters, a final design was developed which increased the signal to noise ratio considerably and gave a clear read out wave. The hysteresis circuit was finally replaced by a better design using an input low pass filter, which worked fine when experimentally tested.

The interference problem resulted because of the physical proximity of the wires coming from the read/write heads. In the original PDP - 9 read/write amplifier, a ground is provided all around the board and precautions had been taken to shield the conductors to the heads. A good and a stable ground solved this problem.

The second major problem was an apparent interference between the timing and mark track that distorts the mark track in each positive transition of the time track. This problem, that probably produced the false readings, was solved later using a pull up resistor of 1K ohms in the output of the read buffers MC8T97P. The lack of specification data about this chip did not allow an early solution to the problem.

Tape degradation due to excessive use and frequent handling caused a delay in the work. As a matter of fact, several reformatings were necessary to achieve the correct read out.

When a second 6821 PIA was connected, the A and B sides (8 bit/each) of this new PIA supposed to be used for reading and writing the 16 bit word, using the two consecutive memory locations 4004 and 4005 (memory location for the output register of the new PIA). Nevertheless, when the word loaded into the read/write registers were checked with the logic analyzer, it showed an incorrect word being loaded from the A side and a correct one being loaded from the B side. This behaivor is due to the difference in the hardware of the two sides of the PIA. The A side is a TTL compatible input/output peripheral line, while the B side has a three state buffering between the output register and the peripheral lines such that the MPU will read the current contents of the output register for those bit-positions programmed as outputs. The word from the A side was distorted because of the loading into the read/write registers. This problem was solved using the output register B of the previous PIA already connected in the microprocessor kit.

Finally, when the information loaded into the read/write registers were checked, it was found that the pulse that loads in parallel into the read/write registers, inhibits two timing pulses, making the register lose information. This problem was solved, making the parallel load pulse respond to a negative clock transition instead of the positive one and changing the clock pulse from a square wave to a train of impulses at each positive transition of the previous pulse clock so that only one impulse is present when the load pulse is high. This was needed, because the 74299 Universal Shift Register requires a clock pulse synchronous with the parallel load pulse in the input S0. Whenever this input is low, the clock pulses make the shift register shift in serial form.

CONCLUSIONS

The following conclusions aim to help the further continuation of the present work.

Due to the difficulties encountered in the present work, the final circuit does not perform the complete set of functions.

The read out of the mark and timing tracks have been achieved, making the Stop, Move, Search, and Change of Direction subprograms work as expected. The read and write subprograms, were tested in the 8 bit case and proved to work well. For incorporating the 16 bit word into the existing system, several changes in the hardware as well as in the software had to be made which affected the performance of the circuit.

The words that were written and read have been checked carefully and proved to be correct in most of the cases. A formating subprogram was designed that writes the codes 22, 26 and 70, with the feature (same as in the PDP - 9) of variable word size (number of 70 mark track. Tested initially with FF hexadecimal blocks) and variable block size (number of 26 mark track. Tested initially with FF hexadecimal blocks).

The tape was formated and the different mark codes were successfully read with the only problem that unexpected 22 marks were found all over the tape. To solve these problems, a careful check of all timing signals must be done to ensure an appropiate synchronization of the timing and mark track signals.

Finally, the 6800 microprocessor proved to be powerful enough to accomplish this project, but a few disadvantages are worth pointing out. First, the low frequency system clock of 614.4 KHz make the operations of read and write very critical. For instance the read and store operations have to be done twice while the information is simultaneously serially shifted into the read/write registers. Second, the lack of more 16 bit work registers in the CPU make difficult operations like using the index register as a pointer for reading or writing words into the microprocessor memory. This makes it impossible (without a big amount of software complication) to use as an intermediate storage register for loading both PIA sides at the same time, an operation that would be desirable if it had been possible.

REFERENCES

1. PDP - 9 MAINTENANCE MANUAL, "Digital Equipment Corporation", Maynard, Mass. 1969
2. TC02 Dectape CONTROL MAINTENANCE MANUAL, "Digital Equipment Co". Maynard Mass. 1971
3. TU55 DECTAPE TRANSPORT MAINTENANCE MANUAL, "Digital Equipment Co". Maynard Mass. 1971
4. THE TTL DATA BOOK second edition. "Texas Instruments Inc". 1976
5. MICROPROCESSOR APPLICATIONS MANUAL, "Motorola Semiconductor Products Inc". McGraw hill, 1975
6. M6800 PROGRAMMING REFERENCE MANUAL "Motorola", 1976
7. INTRODUCTION TO SWITCHING LOGIC C. L. Sheng "Intext Educational Publishers" 1972
8. Kraft G. D, Toy W. N. MINI/MICROCOMPUTER HARDWARE DESIGN "Prentice-Hall". 1979
9. Field A. DECTAPE CONTROLLER WITH THE MKE6800, "Electrical Engineering Dep. U.R.I." Rhode Island, 1979
10. Lee, S. C. DIGITAL circuits AND LOGIC DESIGN "Prentice Hall", 1976
11. McNamara, B. M. COMPUTER ENGINEERING "Digital Equipment Co". 1978
12. Jung, W. G. IC OP-AMP Cookbook, "Sams Publications". 1974
13. LINEAR DATA BOOK "National Semiconductor", USA. 1980.

14. Irvine, R. G. "OPERATIONAL AMPLIFIERS" "Prentice Hall". 1981.

BIBLIOGRAPHY

1. Field A. DECTAPE CONTROLLER WITH THE MKE6800, "Electrical Engineering Dep. U.R.I." Rhode Island, 1979

2. Irvine, R. G. OPERATIONAL AMPLIFIERS, "Prentice Hall". 1981.

3. Jung, W. G. IC OP-AMP COOKBOOK, "Sams Publications". 1974.

4. Kraft, G. D. Toy, W. N. MINI/MICROCOMPUTER HARDWARE DESIGN "Prentice-Ha-l". 1979.

5. Lee, S. C. DIGITAL CIRCUITS AND LOGIC DESIGN "Prentice Hall", 1976.

6. LINEAR DATA BOOK "National Semiconductor", National Semiconductor Publications . 1980.

7. MICROPROCESSOR APPLICATIONS MANUAL, "Motorola Semiconductor Products Inc". McGraw Hill, 1975.

8. McNamara, B. M. COMPUTER ENGINEERING "Digital Equipment Co". 1978.

9. M6800 PROGRAMMING REFERENCE MANUAL "Motorola", 1976.

10. Sheng, C. L. INTRODUCTION TO SWITCHING LOGIC "Intext Educational Publishers", 1972

11. TCO2 Dectape CONTROL MAINTENANCE MANUAL, "Digital Equipment Co". Maynard Mass. 1971

12. THE TTL DATA BOOK Second edition. "Texas Instruments Inc. 1976.

13. TU55 DECTAPE TRANSPORT MAINTENANCE MANUAL, "Digital Equipment Co". Maynard Mass. 1971.

14. PDP-9 MAINTENANCE MANUAL, "Digital Equipment Co", Maynard Mass. 1969.