1972

Design of a digital control unit for a seismic digital tape system

Cao-van Chung

Department:

DESIGN OF A DIGITAL CONTROL UNIT FOR A SEISMIC DIGITAL TAPE SYSTEM

by

Cao-van Chung, 1947-

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI - ROLLA

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

1972

Approved by

James R. Tracy (Advisor)

TO MY GRANDPARENTS

ABSTRACT

This paper describes the interface between the NOVA digital computer and the Texas Instrument tape unit which handles field data recorded simultaneously from a maximum of 31 channels. The interface is the means through which the NOVA computer can control the tape unit and recover data. To provide necessary background information a review of the characteristics of the tape unit and the input-output operation of the NOVA computer are given. As the first part of the design, the general interface design is presented and the functions of its units are described. After discussion of functional assignment of each unit, coordination between them is discussed on the transfer level so that the Boolean equations describing the interface operation can be derived.

The interface can run the tape transport in forward or reverse to find a desired record. It can select the data of any specified channel in any desired record from some specified point in the record. The flexibility of an interrupt system is available for the data processing.

ACKNOWLEDGEMENT

The author is deeply indebted to Dr. James H. Tracey for his assistance and guidance during the graduate program and in the preparation of this thesis.

Thanks also to Larry Gardner and Hugh F. Spence for their technical assistance. Appreciation is also extended to Mrs. Eunice French for her typing efforts.

After all the author would like to share this happiness with his wife and the good friends which he happens to know in the research computer lab: Kurt Hambacker and Thomas D. Steury.

TABLE OF CONTENTS

TABLE OF CONTENTS (cont'd) PAGE

TABLE OF CONTENTS (cont'd) PAGE

TABLE OF CONTENTS (cont'd) PAGE

LIST OF FIGURES

I. INTRODUCTION

The nature of this project is to design an interface system between the NOVA computer and the Texas Instrument magnetic tape unit, both of which are located in the Electrical Engineering Building at the University of Missouri - Rolla. The tape unit handles field data recorded simultaneously from a maximum of 31 channels. The purpose of the system design is to provide a mechanism for recovering the data and entering it into the IBM 360/50 for analysis. The function of the interface is to accept tape unit commands from the computer, control the operation of the tape unit, and selectively transfer data recorded on tape to the computer. The interface also contains conventional error detection and recovery hardware to insure reliable operation.

The organization of this paper is divided into three major sections. In the first part, a review of the characteristics of the tape unit and the input-output operation of the NOVA are given. In the second part, the design of the interface system in terms of construction and transfer description is presented. In the third part the design of a software system coordinated with this interface is discussed.

The references listed in the Bibliography would be helpful to the reader in understanding a logic design like

the design presented in this paper. For a basic background in logic design and switching circuit, the reader can refer to references 4, 5, 6, 7, 8, and 9 which give Boolean algebra theory, fundamental logic design and the concepts of the combinational circuits, sequential circuits, and synchronous asynchronous machines. For the advanced digital system design, the computer designs presented in references 10 and 11 are significant. For information about the design languages used to describe the logic design, the reader can refer to references 12, 13, and 14.

Last of all, for the details of this design, the USER'S MANUAL[3] for this interface, which contains logic diagrams and a proposed software system, can be referenced.

II. DESCRIPTION OF THE DFS TAPE UNIT AND THE NOVA COMPUTER

A. DESCRIPTION OF THE DFS IN LOCAL OPERATION

The Texas Instrument Digital Field System (DFS) is a unit used to digitize and record multi-channel analog input data with a high degree of accuracy. It is compatible with the Texas Instrument 9000 series seismic amplifier system.

1. FUNCTIONAL DESCRIPTION

The DFS digitizes 31 analog channels at 1000 samples per second per channel. The binary coded data output is recorded on a 21-track digital magnetic tape. The three primary modes of the operations are:

Record Mode (RCM)

The analog input data is timed multiplexed, sampled, digitized and written on the magnetic tape. For each record the operator enters a record number, by switches, as the address of the record. At present there is no need for the NOVA computer to operate the DFS in record mode.

Search Mode (SRM)

The magnetic tape is read to locate a previously written record. The operator enters the desired record number by switches. This record number will be compared

to the record numbers read from tape by the Timing Address unit (T/A). The operator can run the DFS in either forward or reverse direction in search mode to find his desired record number. When the desired record number is found, the T/A unit generates a "PSS" pulse (Playback, Search, Stop Pulse) to stop the tape transport.

Playback Mode (PBM)

The system reads the magnetic tape, converts the digital data to analog data form and demultiplexes the signal to individual channels. In playback mode an all 1's record number is automatically supplied to the T/A unit, therefore, the "PSS" pulse will be generated at the end of a record to stop the tape transport, because the end-of-data is an all 1's word. The "PSS" pulse will be inhibited in the control from the NOVA.

2. SPECIFICATIONS

Record Number

A record number is an address of a record on tape. It is a 10 bit word and is also called a start-of-data word.

Block Number

A block number is an address of a data block in a record. Each data block contains data of 31 analog channels and a block number. In this design the block number is considered as data of channel 0, so there are a total

of 32 channels. The block number is the time in milliseconds at which samples were taken. In normal seismic use a record contains some 4000 to 5000 blocks.

Speed

The DFS tape transport can operate at two speeds labeled 1MS and 2 MS. At these speeds the time between consecutive data words is approximately 32MS or 64MS. Since there are 32 channels per data block this means each data block takes 1 or 2 milliseconds to read or record.

Tape Format: Fig. 2-1 and Fig. 2-2

The reader should note that four types of data are distinguished by the bits 0, 1, and B:

BIT0	BIT1	BITB		
1	0	1	:	record number
0	0	1	:	block number = data of channel 0
S	S	0	:	data of channels 1-31
1	1	1	:	end-of-data.

In other words

If BIT0.$\overline{\text{BIT1}}$ = 1 then the data is a record number

If $\overline{\text{BIT0}}$.BITB = 1 then the data is a block number

If $\overline{\text{BITB}}$ = I then the data is data of channels 1-31

If BIT0.BIT1.BITB = 1 then the data is an end-of-data word.

Parity Check: "ODD" Parity Check

The parity check bit (P) establishes odd parity over bits 0 through 17. It doesn't check bit (B) and clock bit (C).

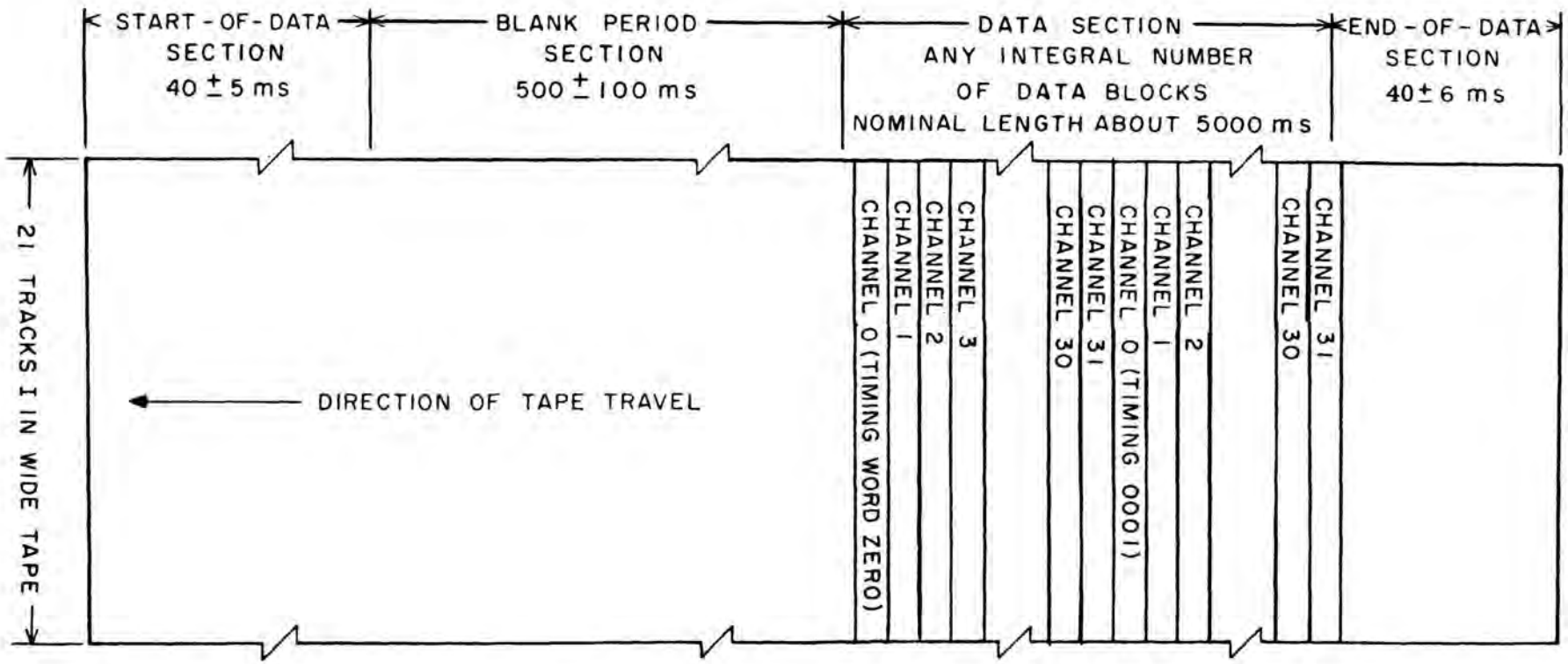
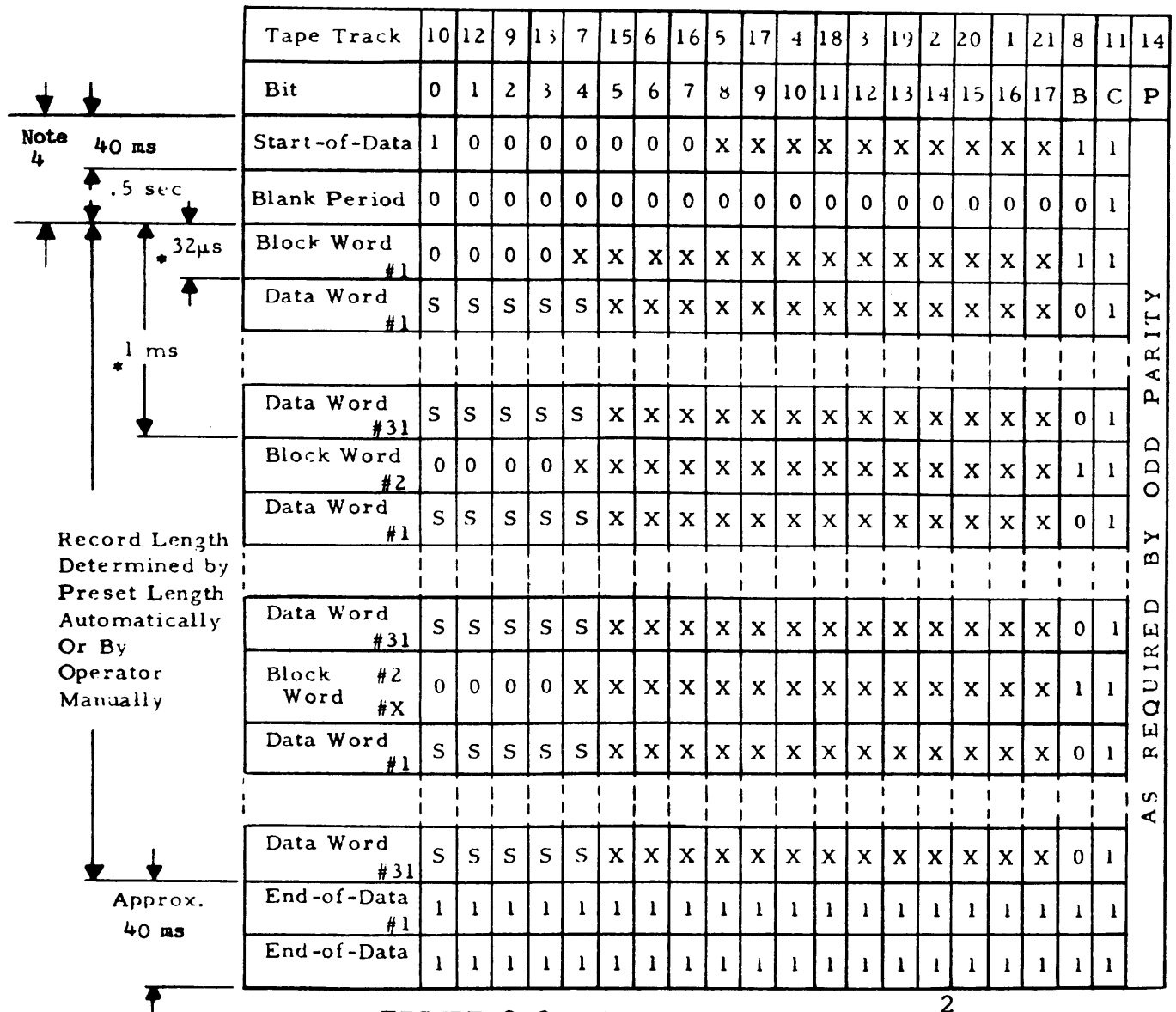


FIGURE 2-1 GENERAL TAPE FORMAT[2]



Tape Track	10	12	9	13	7	15	6	16	5	17	4	18	3	19	2	20	1	21	8	11	14
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	B	C	P
Start-of-Data	1	0	0	0	0	0	0	0	X	X	X	X	X	X	X	X	X	X	1	1	AS REQUIRED BY ODD PARITY
Blank Period	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
Block Word #1	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	
Data Word #1	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
Data Word #31	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
Block Word #2	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	
Data Word #1	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
Data Word #31	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
Block Word #2 #X	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X	X	X	1	1	
Data Word #1	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
Data Word #31	S	S	S	S	S	X	X	X	X	X	X	X	X	X	X	X	X	X	0	1	
End-of-Data #1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
End-of-Data	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

NOTE:
1. Bits decrease in significance with an increase in number.
2. "1"=Binary One "X"=Binary Variable "0"=Binary Zero "S"= Sign Bit
3. *Period determined by Clock rate
4. Determined by period between start and TB arrival. In automatic operation time as shown.

FIGURE 2-2 DETAILED TAPE FORMAT [2]

B. INPUT-OUTPUT DESCRIPTION OF THE NOVA COMPUTER

The NOVA computer is a general purpose computer system with a 16-bit word length. A memory address is a 15-bit word so the maximum capacity of memory is $2^{15} = 32,768$ words. The arithmetic and logical operations are handled in 4 accumulators ACϕ, AC1, AC2, and AC3. AC2 and AC3 can be used as index registers. The input-output descriptions of the NOVA computer will be summarized as in the following paragraphs.

The NOVA computer assumes that each peripheral device has three registers called A, B, C and four general control flip flops called BUSY, DONE, INTDIS (interrupt disable), INTREQ (interrupt request).

The transfers between any one of the registers A, B, C and any one of the accumulators ACϕ, AC1, AC2, AC3 can be made by executing the instructions DATA OUT A (DOA), DATA IN A (DIA), DATA OUT B (DOB), DATA IN B (DIB), DATA OUT C (DOC), and DATA IN C (DIC)[1].

The BUSY flip flop, which can be set only by the central processor, is the means by which the central processor can control the setting of DONE. When BUSY is off DONE can't be set.

The DONE flip flop, if BUSY = 1, will be set when the device completes an operation and needs to be serviced by the central processor. When DONE is set the signal DONE

will clear BUSY.

The INTDIS flip flop in the off state represents permission given by the central processor to allow the DONE flip flop in the set state to set the INTREQ flip flop to raise an interrupt request to the central processor.

An interrupt request will be honored if the "INTERRUPT ON" flag of the central processor is on. When responding to an interrupt request, the central processor clears the INTERRUPT ON flag to inhibit further requests, puts the address of the instruction just following the last executed instruction into location 0, and then executes an indirect JUMP to location 1 to take the next instruction to execute.

To determine which devices have raised the interrupt requests, the processor can test serially the states of the DONE flip flops of the devices or execute the INTERRUPT ACKNOWLEDGE instruction (INTA) to place into an accumulator the device code of the "first" device on the bus that is requesting an interrupt ("first" means the one that is physically closest to the processor on the bus).

The central processor can specify which devices may raise the interrupt request by executing the MASK OUT instruction (MSKO) to clear or set the INTDIS flip flops of the devices according to an "interrupt mask" contained in an accumulator.

III. HARDWARE DESIGN

A. FUNCTIONAL DESCRIPTION OF THE INTERFACE SYSTEM: REFER TO BLOCK DIAGRAM FIG. 3-1

In this section the functions of every unit in the interface will be generally described. The reader will see how these functions are carried out in Section D, which gives the logic description. The signals which are used in the discussion may be looked up in Appendix A.

1. COMMAND REGISTER

The COMMAND REGISTER, Aϕ, A1, . . . A7, is an 8-bit register. This register contains a command sent from the accumulator. A command has two portions. One portion is sent to the DFS to tell the machine whether to operate in the search mode (reverse), playback mode (forward) or to stop. The second portion is sent to the DONE CIRCUIT to tell this circuit what kind of data should be transferred into the accumulator--record number, block number, or channel data. The second portion also specifies whether all 32 channels or only some of them will have their data transferred into the accumulator.

2. NOVA DFS FLIP FLOP

To enable communication between the NOVA and the DFS

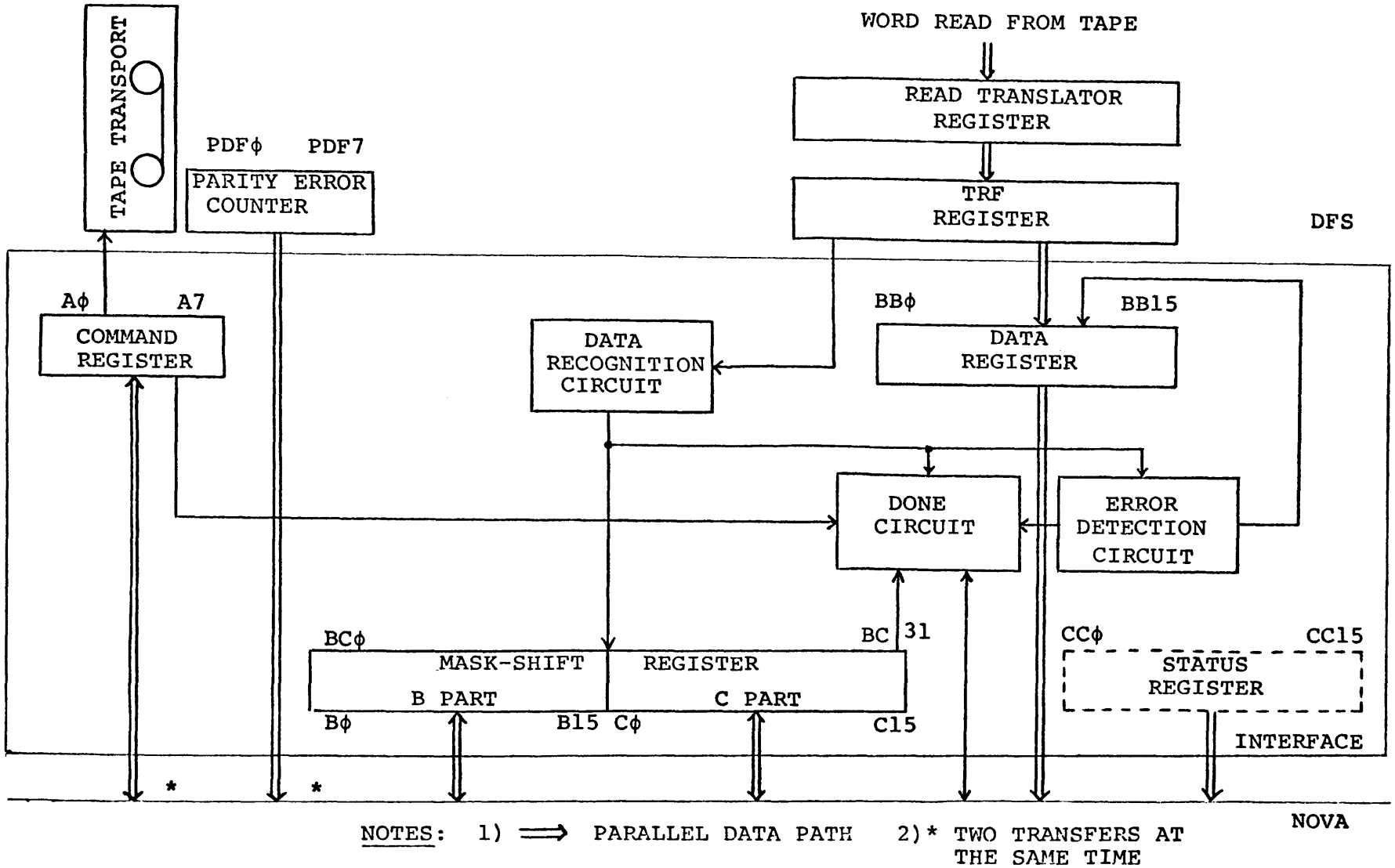


FIG. 3-1. BLOCK DIAGRAM OF THE INTERFACE SYSTEM

operator has to first set a flip flop called "NOVA DFS" by depressing a button on the DFS. The NOVA DFS signal will enable control signals between the NOVA and the DFS and disable any control signals generated by the DFS which can interfere with the control from the NOVA. The NOVA DFS flip flop can be reset manually by the operator with another button on the DFS or reset automatically by an accidental stop of the DFS (e.g., the tape is broken, the tape is over . . .).

3. DATA RECOGNITION CIRCUIT

This circuit determines the type of all data sent from the DFS to the NOVA to tell the DONE CIRCUIT what kind of data is on the DATA LINES (between the NOVA and the DFS) and to tell the MASK-SHIFT REGISTER when it must shift to stay in synchronism with any channel data coming over the DATA LINES. When shifting of the MASK SHIFT REGISTER is initiated, the DATA RECOGNITION CIRCUIT also initiates the ERROR DETECTION CIRCUIT which checks synchronization between the MASK SHIFT REGISTER and the channel data coming over the DATA LINES.

4. MASK-SHIFT REGISTER

The MASK-SHIFT REGISTER, BCϕ, BC1, . . ., BC31, is a 32-bit rotate-right-shift register. This register is composed of two 16-bit registers B(0-15) and C(0-15). The

MASK-SHIFT REGISTER contains a 32-bit mask sent from the NOVA computer.

When signaled by the DATA RECOGNITION CIRCUIT, the MASK-SHIFT REGISTER will rotate-right shift the mask which it contains synchroniously with the channel data coming over the DATA LINES. Every "mask bit" of the channel will in turn be shown to the DONE CIRCUIT from the rightmost bit of the register, so that the DONE CIRCUIT will know which channels should have their data transferred into the NOVA accumulator. The channel which has mask bit 1 is not masked; its data will be transferred. However, if the second portion of the COMMAND REGISTER has specified that all 32 channels should have their data transferred, then the mask will not have any effect.

5. DONE CIRCUIT

The DONE CIRCUIT consists of the DONE flip flop, the BUSY flip flop, the INTDIS flip flop (interrupt disable), the INTREQ flip flop (interrupt request), and a "combinational circuit"[4]. This combinational circuit uses information supplied by the DATA RECOGNITION CIRCUIT, the COMMAND REGISTER, and the MASK-SHIFT REGISTER to decide whether or not the data on the DATA LINES should be received into the DATA REGISTER of the interface.

The DONE flip flop, if BUSY is on, can be set two ways: by an ERROR signal sent from the ERROR DETECTION

CIRCUIT or by the "combinational circuit" when the data on the DATA LINES is of the type requested by the COMMAND REGISTER and the MASK-SHIFT REGISTER. The DONE flip flop is set to notify the central processor that the device needs to be serviced and if no error occurs the signal DONE will gate the data on the DATA LINES into the DATA REGISTER.

The BUSY flip flop, which can be set only by the computer, is the means by which the central processor can control the setting of DONE. When BUSY is off, DONE can't be set, and when DONE is set, BUSY is cleared[1].

The INTDIS flip flop in the off state represents permission given by the central processor to allow the DONE flip flop in the set state to set the INTREQ flip flop to raise an interrupt request to the central processor. The interrupt request will be honored if the "INTERRUPT ON" flag of the central processor is on[1].

In responding to a DONE setting or an interrupt request the computer is usually programmed to generate a start signal called "STRT"[1] after transferring the data in the DATA REGISTER into the accumulator. The STRT signal sets BUSY, clears DONE, and clears INTREQ to make them ready for the next data transmission. However, when DONE is set by the ERROR signal DONE will be clamped in the on state as long as the ERROR signal is on. The signal STRT[1] won't be able to clear DONE in this case. This is the difference between a DONE setting caused by the data and

a DONE setting caused by the ERROR signal. The computer can test the state of DONE after generating the STRT signal to know whether some system error has occurred.

6. ERROR DETECTION CIRCUIT

The ERROR DETECTION CIRCUIT is used to detect the following errors:

LATE Error

If new data comes to the DATA REGISTER before the old data in the register has been transferred into an accumulator, a LATE flip flop will be set to show that the central processor is late in responding to the data. The new data cannot replace the old data because DONE is still on, and LATE is set at the time the new data should have been gated into the DATA REGISTER.

CUT Error

If the NOVA computer wishes to continue communication with the DFS and if for any reason the NOVA DFS flip flop has been reset to disable the communication, then the CUT flip flop will be set to notify the computer that the communication to the DFS can't be continued. NOVA DFS may have been reset because of the operator or because of a malfunction in the DFS which caused the machine to stop accidentally (see NOVA DFS flip flop).

IMS Error

If by some reason (which will be mentioned in the logic

description of the interface) the MASK-SHIFT REGISTER no longer shifts its mask synchronously with the channel data coming on the DATA LINES from the DFS, then a flip flop, "IMS", will be set to notify the computer that there is a loss of synchronization between the shifting of the MASK-SHIFT REGISTER and the coming of channel data on the DATA LINES.

7. DATA REGISTER

The DATA REGISTER, BBϕ, BB1, . . . BB15 is a 16-bit register which has two functions:

--when no error occurs the DATA REGISTER receives the data selected by the DONE CIRCUIT

--when an error occurs the data on the DATA LINES will be inhibited from entering the DATA REGISTER. If the LATE error occurs then the instruction "JUMP .+1"[1] will be formed in the DATA REGISTER. If the CUT error occurs then the instruction "JUMP .+3" will be formed in the DATA REGISTER. If the IMS error occurs the instruction "JUMP .+7" will be formed in the DATA REGISTER. After sensing an error the computer may transfer JUMP instruction from the DATA REGISTER into the accumulator and execute it to jump to a particular subroutine which initiates recovery from the error.

8. STATUS REGISTER

Except for the DONE and BUSY flip flops, all other control flip flops in the interface are grouped to form the STATUS REGISTER. This 16-bit register, CCϕ, CC1,... CC15, can have its content transferred into an accumulator to supply information necessary to the computer for error diagnosis.

B. SIGNALS BETWEEN THE DFS AND THE INTERFACE (Ref. 2 and Diagram 5[3])

1. CONTROL SIGNALS

NOVA DFS

This signal is the '1' output of the NOVA DFS flip flop of the interface. The function of this signal has previously been described as the function of the NOVA DFS flip flop; however, for the miscellaneous control of the signal, the reader can refer to the USER'S MANUAL.

RUN/STOP

RUN and STOP are respectively the '1' and '0' outputs of the flip flop A1 in the COMMAND REGISTER. RUN is the second condition (after NOVA DFS) to enable the signals SRM and PBM below. STOP, if enabled by NOVA DFS, will stop the tape transport of the DFS[3].

SRM/PBM

SRM and PBM are respectively the '1' and '0' outputs

of the flip flop A2 in the COMMAND REGISTER. PBM is used to run the DFS in playback mode with the tape moving forward. SRM is used to run the DFS in search mode. The direction selected for this mode is reverse only. Search mode in forward was not used because the interface already allows forward search in playback mode.

Each time SRM or PBM is set up to run the DFS in search or playback mode, a stop pulse is generated to stop the DFS first[3]. Therefore, when the interface is commanded to change the DFS from one mode to the other, the DFS will always stop first and then about 30MS later it will run again in the opposite mode[3].

START5 (distinguish from the start signal of the NOVA)

This signal is taken from pin 5 of the start relay K5003[2] of the DFS. It is high when the DFS is running and low when the DFS stops.

STOPDP

The stop delay pulse, STOPDP, is generated about 40MS after the DFS stops[3]. The signals START5 and STOPDP are used to detect an accidental stop of the DFS.

CLOCK-PES (Parity Error Signal)

In the logic description of the interface the reader will see the functions of these two signals; hereafter the generation[2] and the timing of these signals will be described (refer to the timing diagram Fig. 3-2(b) and the tape format for the clock bit (C) Fig. 2-2). Let "t_o" be

the time at which a word is read from tape so at:

t_o : a word including the clock bit (C) is read from tape to the Read Translator Register (RTR).

t_o + 8MS : a pulse, TRR, derived from the clock bit (C) is generated to clear the Transfer Register. This register will be called the "TRF REGISTER" as shown on the logic diagram of the DFS[2].

t_o + 12MS : a pulse, "RTR-TRR", derived from the clock bit is generated to transfer data from the Read Translator Register to the TRF REGISTER.

t_o + 16MS : a pulse, "RTR", derived from "RTR-TRR" is generated to clear the Read Translator Register and make it ready for the next word.

t_o + 28MS : if the word just read into the TRF REGISTER has a parity error, the pulse "PES" is generated.

t_o + 32MS : CLOCK is generated from the RTR pulse[3]. At this time another word is read from tape to the Read Translator Register if the DFS is running at 1MS speed.

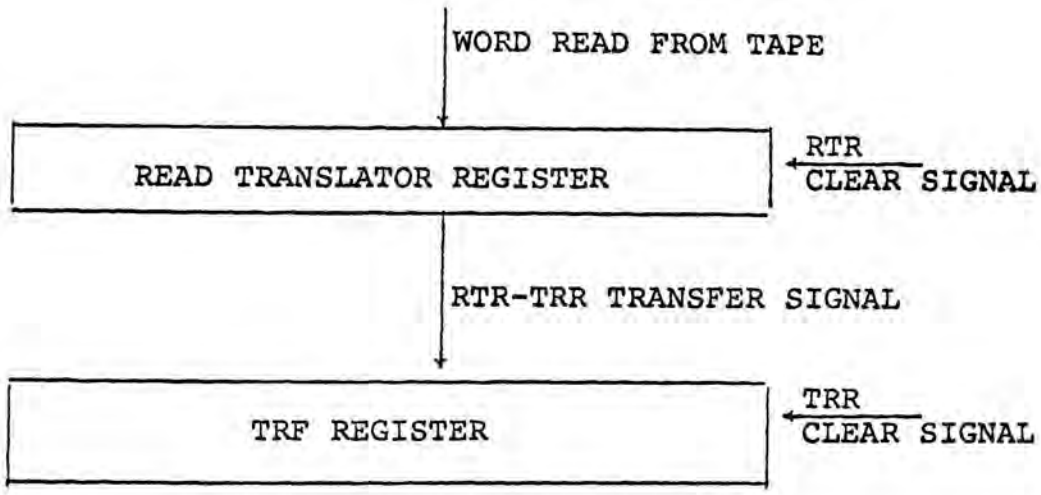
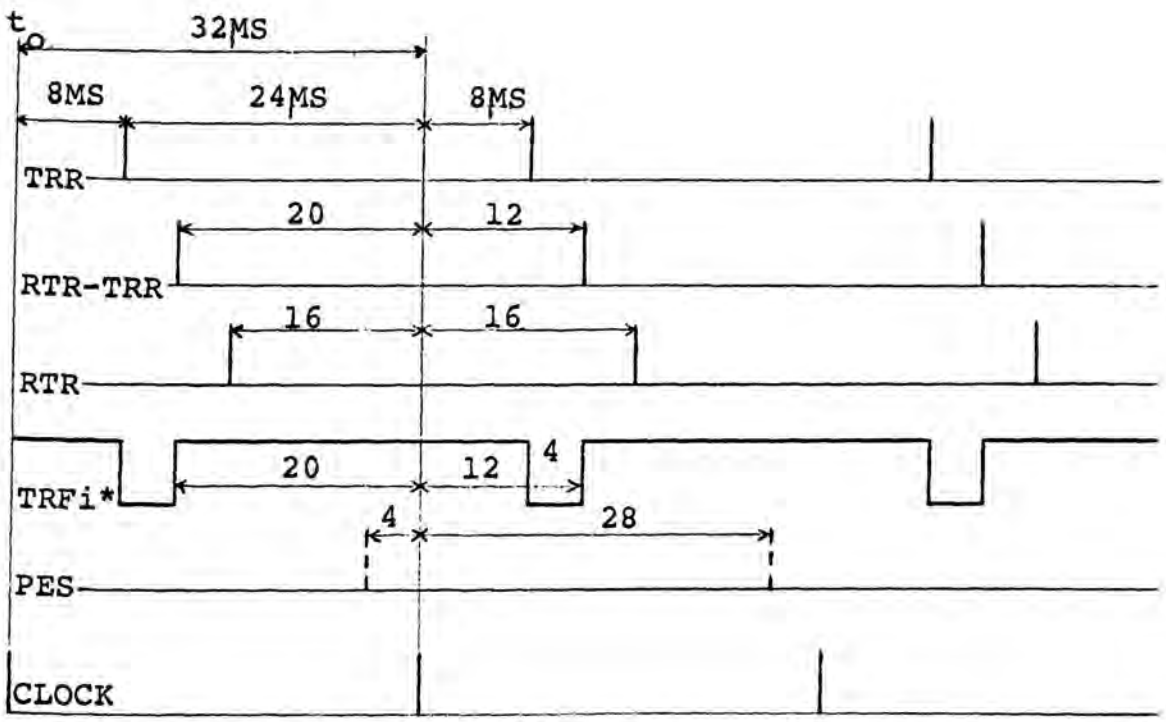


FIG. 3.2(a): READ TRANSLATOR REGISTER AND TRF REGISTER



*TRFi : bit i of TRF REGISTER, supposed in the on state

FIG. 3.2(b): TIMING DIAGRAM OF THE SIGNALS CLOCK,PES,TRFi

2. DATA LINES: TRFϕ, TRF1, TRF4, TRF5, . . ., TRF17, TRFP, TRFB

The data contained in the TRF REGISTER is sent to the interface over 18 lines, TRF0, TRF1, TRF4, TRF5, . . ., TRF17, TRFP, TRFB (see tape format Fig. 2-2). 16 bits TRF4, TRF5, . . ., TRF17, TRFP, TRFB go to the DATA REGISTER of the interface. TRF0, TRF1, TRFB go to the DATA RECOGNITION CIRCUIT to inform the circuit what kind of word is on the DATA LINES. The reader should note the timing relationship between a bit TRFi and the signals CLOCK, PES shown on timing diagram Fig. 3-2(b).

3. PARITY INFORMATION: PDFϕ, . . ., PDF7

On the DFS there are 8 parity display flip flops forming a counter which can be switched by the operator to count either the parity errors in playback mode or the number of overdrive points[2] in playback mode or record mode. Overdrive points are recorded as data which have value of "-0" or 100 . . . 0 in two's complement binary.

C. IN-OUT SIGNALS AND IN-OUT INSTRUCTIONS BETWEEN NOVA AND THE INTERFACE (Diagram 1[3])

For convenience of this discussion (refer to Fig. 3.3):

Let ACI be the accumulator I of the NOVA with I = 0, 1, 2 or 3. ACIn be bit n of ACI (0-15).

With the I/O instructions below, if the instruction affects only the DFS interface, the select device signal, SELDFS, is generated to direct the other control signals to only this interface[1].

1. DATA-OUT-A INSTRUCTION: "DOA I, DFS"

This instruction, which generates signal DATOA[1], is used to transfer a command from an accumulator to the COMMAND REGISTER. Three types of transfer are available. When this instruction is executed

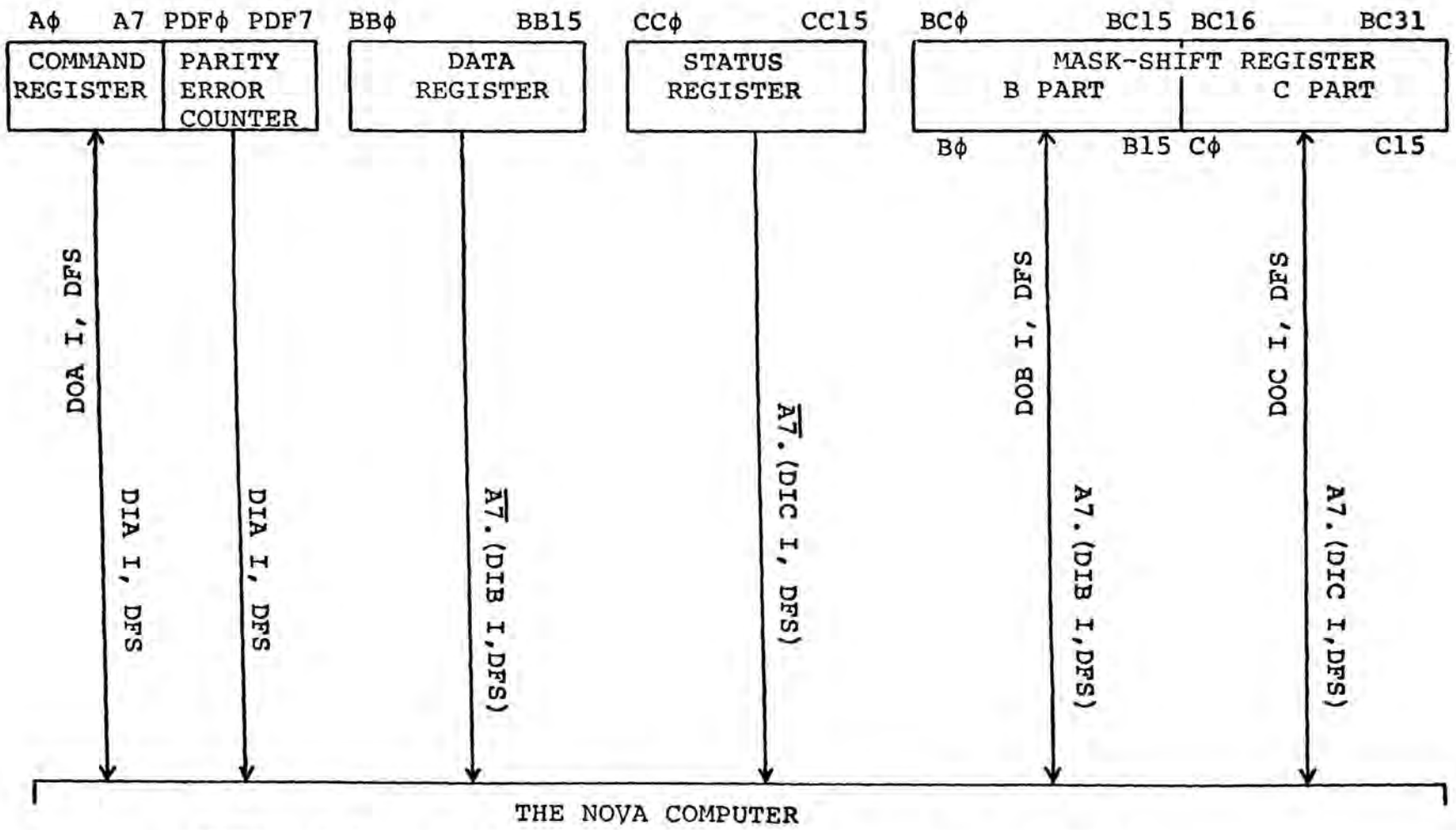
If $\overline{ACI8}*\overline{ACI9}$ = 1 then An←ACIn with n = 0, 1, 2 . . . and 7
or A(0-7)←ACI(0-7)

If ACI8*$\overline{ACI9}$ = 1 then An←An +ACIn

If ACI9 = 1 then An←$\overline{ACIn}$.An

With the second type of transfer a bit, An, of the COMMAND REGISTER will be set if its corresponding bit ACIn = 1 and An will be unchanged if bit ACIn = 0.

With the third type of transfer a bit, An, of the COMMAND REGISTER will be reset if its corresponding bit ACIn = 1. If ACIn = 0, An will be unchanged.



NOTE: I is an index of an accumulator, I = 0, 1, 2 or 3

FIG. 3.3 IN-OUT DATA TRANSFERS

Ex. Suppose:

	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ACϕ =		1	1	0	1	0	1	0	1	0	0	X	X	X	X	X	X
AC1 =		0	0	1	0	1	0	1	0	1	0	X	X	X	X	X	X
AC2 =		0	0	0	0	1	1	0	1	X	1	X	X	X	X	X	X

After the sequential executions of the instructions DOA 0, DFS; DOA 1, DFS; DOA 2, DFS the COMMAND REGISTER will be respectively:

A = 1 1 0 1 0 1 0 1

A = 1 1 1 1 1 1 1 1

A = 1 1 1 1 0 0 1 0.

NOTE:

X means "don't care"

The second and third type of transfer are convenient to change some bits of the COMMAND REGISTER without effecting the other bits.

2. DATA-IN-A INSTRUCTION: "DIA I, DFS"

This instruction, which generates signal DATIA[1], performs the transfer below:

ACI(0-7)←A(0-7)

ACI(8-15)←PDF(0-7).

PDF is the parity error counter of the DFS (see Section B).

3. DATA-OUT-B INSTRUCTION: "DOB I, DFS"

This instruction, which generates signal DATOB[1], performs the transfer

B(0-15)←ACI(0-15)

B(0-15) is the first part of the MASK-SHIFT REGISTER, B(0-15) = BC(0-15).

4. DATA-OUT-C INSTRUCTION: "DOC I, DFS"

This instruction, which generates signal DATOC[1], performs the transfer

C(0-15)←ACI(0-15)

C(0-15) is the second part of the MASK-SHIFT REGISTER, C(0-15) = BC(16-31).

5. DATA-IN-B INSTRUCTION: "DIB I, DFS"

This instruction, which generates signal DATIB[1], transfers the content of the DATA REGISTER, BB, or the B part of the MASK-SHIFT REGISTER into the accumulator. When this instruction is executed

If A7 = 0 then ACI(0-15)←BB(0-15)

If A7 = 1 then ACI(0-15)← B(0-15).

6. DATA-IN-C INSTRUCTION: "DIC I, DFS"

This instruction, which generates signal DATIC[1], transfers the content of the STATUS REGISTER, CC, or the C part of the MASK-SHIFT REGISTER. When this instruction is executed

If A7 = 0 then ACI(0-15)←CC(0-15)

If A7 = 1 then ACI(0-15)← C(0-15).

NOTE: Case A7 = 1 is used only for testing the MASK-SHIFT REGISTER. During the data processing A7 can always remain zero.

7. INSTRUCTION "NIOS DFS"

When executed, this instruction sends the start signal, STRT[1], to the interface to perform the following functions:

--clear DONE flip flop
--set BUSY flip flop
--clear INTREQ flip flop
--clear SPEF flip flop (will be described in Section D)
--combine with the error signals to form the JUMP instruction in the DATA REGISTER (will be described in logic description of DATA REGISTER).

8. INSTRUCTION "NIOC DFS"

When executed, this instruction sends the clear signal, CLR[1], to the interface to clear all the flip flops and registers except the flip flops Aϕ, A1, A2, A3 of the COMMAND REGISTER.

9. INSTRUCTION "NIOP DFS"

When executed, this instruction sends the special signal, IOPLS[1], to the interface. This signal is used in place of clock for:

--shifting the MASK-SHIFT REGISTER
--incrementing the synchronization counter (will be described

--testing the condition to set DONE except for ERROR condition.

These functions are useful for adjusting "improper synchronization"[3] between the MASK-SHIFT REGISTER and the channel data coming over DATA LINES (between NOVA and DFS). In Section D, which gives logic descriptions, the functions of this signal can be seen clearly.

NOTE: The instructions NOIS, NIOC, NIOP can be combined with the non-skip I/O instructions[1] DOA, DIA, DOB, DID, DOC, DIC. As an example the instruction

"DOBS I,DFS"

will generate the signal STRT "after" the transfer from the accumulator I to the B register.

10. INSTRUCTION "IORST"

This instruction is common for all devices; the select device signal SELDFS is not generated here. The instruction generates the I/O reset signal, IORST[1], to clear all the flip flops and registers of this interface and the control flip flops in all other interfaces connected to the bus[1].

11. INSTRUCTION "MSKO I"

This instruction, which generates signal MSKO[1], sets or clears the INTDIS flip flop of the interface if the bit "ACI 10" is 1 or 0 respectively. Generally this instruction sets or clears the INTDIS flip flops of all devices according to their "interrupt mask bit"[1] in the accumulator I. The device select signal SELDFS is not generated with this instruction.

D. LOGIC DESCRIPTION OF THE INTERFACE SYSTEM

The logic descriptions in this section will show how the functions listed in the previous section of functional description are carried out.

1. COMMAND REGISTER (The reader can refer to Diagram 3[3])

This register has 8 bits; the bit functions are shown below:

Aϕ Named ON/OFF flip flop.

Aϕ = 1: Turns on the ON/OFF indicator on the control pannel of the DFS to show that the NOVA computer still wishes to communicate with the DFS.

$\overline{A\phi}$ = 1: --Turns off the ON/OFF indicator to show that the NOVA no longer needs to communicate with the DFS.
--Clears the error flip flops LATE, CUT, IMS.

A1 Named RUN/STOP flip flop

RUN=A1=1: enables the signals SRM and PBM below to run the tape transport of the DFS

STOP=$\overline{A1}$=1: if enabled by NOVA DFS, stops the tape transport.

A2 Named SRM/PBM flip flop

SRM=A2=1: if enabled by NOVA DFS and RUN the DFS will operate in search mode

PBM=$\overline{A2}$=1 if enabled by NOVA DFS and RUN the DFS will operate in playback mode

A3

A3 = 1: some channels are masked so that their data will not be entered into the interface DATA REGISTER. The NOVA accumulator can therefore never receive these data. A channel whose mask bit contained in the MASK-SHIFT REGISTER is zero is masked

$\overline{A3}$ = 1: no channel is masked. The mask contained in the MASK-SHIFT REGISTER has no effect.

A4

A4 = 1: The computer requests the record numbers

A4 = 0: The computer doesn't request the record numbers.

A5 A5 has no effect if A6 = 1

A5 = 1: if A6 = 0 the computer requests the block numbers

A5 = 0: if A6 = 0 the computer doesn't request the block numbers.

A6

A6 = 1: The computer requests the data of the unmasked channels (including channel 0) and the end of data words (actually the interface will need only one word). A5 has no effect.

A6 = 0: if A5 = 0 the computer doesn't request data of any channel
if A5 = 1 the computer requests only the block numbers (among 32 channels)

A7

A7 = 1: The instructions "DIB I, DFS"
"DIC I, DFS"
are used for the registers B and C of the

32 bit MASK-SHIFT REGISTER, BC [B(0-15) = BC(0-15), C(0-15) = BC(16-31)].

A7 = 0: The instructions "DIB I, DFS"
"DIC I, DFS"
are used for the DATA REGISTER, BB, and the STATUS REGISTER, CC, respectively (see Section C about I/O instructions).

2. NOVA DFS Flip Flop (Diagram 4^3)

This flip flop can be set and reset manually with two buttons on the DFS. However,

if RUN*$\overline{\text{START5}}$*STOPDP = 1 then NOVA DFS←0.

This logic statement shows that the flip flop can also be reset by an accidental stop of the DFS. RUN = 1 shows that the NOVA wishes to run the DFS but STAR5 = 0 indicates that the DFS has stopped. The term STOPDP is used to distinguish an accidental stop from the case when the NOVA changes the tape transport direction and the DFS has to stop for some time before running again with opposite mode. For this case, the condition RUN*$\overline{\text{START5}}$ = 1 also exists but the time for which the DFS stops is shorter than the delay of the STOPDP signal, so the condition RUN*$\overline{\text{START5}}$*STOPDP=1 therefore does not exist (see description of STOPDP and SRM/PBM in Section B).

3. DATA RECOGNITION CIRCUIT (refer to the tape format Fig. 2-1 and Fig. 2-2)

SDF flip flop (Diagram 4^3)

$$SDF \leftarrow SDF \cdot \overline{CLOCK} + TRF1 \cdot \overline{TRF\phi} \cdot CLOCK$$

This start of data flip flop is set in the start of data section to inform the programmer when the record numbers are available on the DATA LINES (between NOVA and DFS).

DATA flip flop (Diagram 4^3)

This flip flop can be set only in playback mode and it is set during the whole DATA SECTION (see Fig. 2-1) to inform the DONE CIRCUIT, the MASK-SHIFT REGISTER and the ERROR DETECTION CIRCUIT when the channel data are available on the DATA LINES.

If $\overline{A2} \cdot \overline{TRF\phi} \cdot TRFB = 1$ then DATA$\leftarrow$1

If $A2 + TRF\phi \cdot TRF1 \cdot TRFB = 1$ then DATA$\leftarrow$0.

EDF flip flop (Diagram 4^3)

This flip flop is set in the end-of-data section to inform the programmer when the record ends.

$$EDF \leftarrow EDF \cdot \overline{CLOCK} + TRF\phi \cdot TRF1 \cdot TRFB \cdot CLOCK.$$

NOTE: The clear signal, CLR, and the I/O reset signal, IORST, can clear the SDF, DATA, EDF flip flops but for the simplicity these two signals were not presented in the logic statements. A complete logic statement should be as follows:

$$EDF \leftarrow (EDF.\overline{CLOCK} + TRF1.TRFB.CLOCK)\,\overline{IORST}.\overline{CLR.SELDFS.}$$

However the reader can refer to the Section C for the functions of these signals, CLR and IORST.

SD Signal (Diagram 4^3)

$$SD = TRF\phi.\overline{TRF1}.$$

This signal is on when a start of data (record number) is available on the DATA LINES.

BL Signal (Diagram 4^3)

$$BL = TRFB.DATA.$$

This signal is on when a block number is available on the DATA LINES (in playback mode only).

ED Signal

$$ED = TRF0 \cdot TRFI \cdot TRFB.$$

This signal is on when an end of data is available on the DATA LINES.

4. MASK-SHIFT REGISTER (Diagram 2^3)

This register contains a 32 bit mask sent from the accumulator. If MSKn is the mask bit of channel n then the meaning of each bit in the mask will be determined as follows:

If $A3 \cdot \overline{MSKn} = 1$ then the channel n is masked

If $A3 \cdot MSKn + \overline{A3} = MSKn + \overline{A3} = 1$ then the channel n is not masked

Fig. 3-4(a) shows the initial content of the MASK-SHIFT REGISTER

Fig. 3-4(b) shows the content of the MASK-SHIFT REGISTER after one rotate right shift.

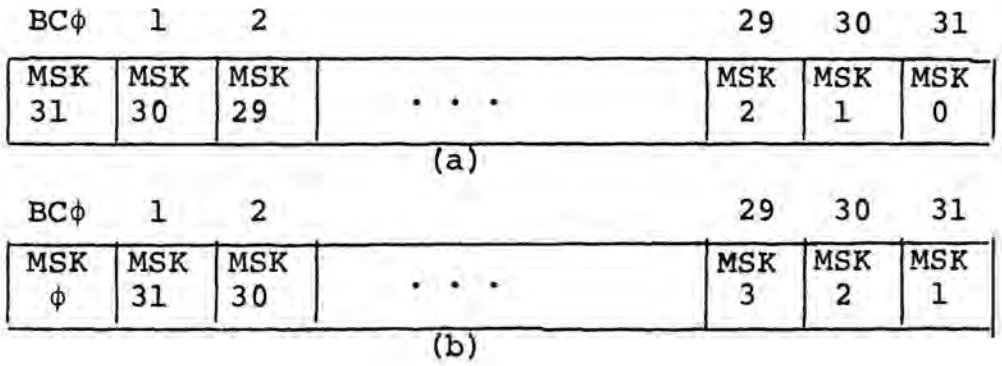


Fig. 3-4 (a): Initial Content of MASK-SHIFT REGISTER (the mask is not yet shifted)

(b): Content of the MASK-SHIFT REGISTER after one shift

The shifting of the MASK-SHIFT REGISTER is shown by the logic statement below:

If DATA.CLOCK+IOPLS.SELDFS = 1 then BC(n+1 modulo 32)←BCn

n=0,1,2,...,30 and 31.

That is, when the "if condition" is satisfied these transfers are performed:

BCϕ←BC31

BC1←BCϕ

BC2←BC1

. . .

. . .

. . .

BC31←BC30.

The following things are noted from the if statement above:

--The shifts are performed only in the DATA SECTION and only during playback mode. When data of channel 0 comes on the DATA LINES mask bit "MSKϕ" is already at the rightmost bit, BC31, of the MASK-SHIFT REGISTER. The DATA flip flop is set at this time to enable the MASK-SHIFT REGISTER. The coming CLOCK will make the first shift to bring the mask bit "MSK1" to BC31, so that when the data of channel 1 comes on the DATA LINES its mask bit, MSK1, is already at the rightmost bit, BC31, of the MASK-SHIFT REGISTER. When the data of the channel 31 comes on the data lines its mask bit, MSK31, is already at BC31 and the next CLOCK will bring mask bit MSKϕ back to BC31. If after channel 31 an end-of-data comes on the DATA LINES, the DATA flip flop will be reset and the MASK-SHIFT REGISTER no longer shifts so it has the original content again.

--The computer, by executing the instruction "NIOP DFS" (to generate signal IOPLS to this device) can also cause the shifts.

5. DONE CIRCUIT (Diagram 1^3)

In this section only the logic description of the

DONE flip flop is presented. The logic descriptions of the BUSY, INTDIS, INTREQ flip flops are just the same as the other peripheral devices and can be looked up in the NOVA manual[1]. The reader can refer to Section III.A.5 for the functions of these flip flops.

The conditions to set and reset DONE are as follows:

Let ND = A4.SD + A5.$\overline{A6}$.BL + A6[($\overline{A3}$+BC31) DATA + ED] (1)

Let CK = CLOCK + IOPLS.SELDFS

Let CLIO = CLR.SELDFS + IORST. Then:

DONE←[(DONE+BUSY.ND.CK)$\overline{STRT}$ + BUSY.ERROR]$\overline{CLIO}$. (2)

The equation (1) defines a Necessary Data signal, ND, which is on when the DATA LINES have data requested by the computer.

The statement (2) shows the following things:

--If BUSY = 1, DONE can be set by the Necessary Data signal or by the ERROR signal. If set by the ND, it will be set at the time the CLOCK comes or the signal IOPLS is generated. The IOPLS signal is used in place of CLOCK in some special case for testing, diagnosis . . . etc. . .

--DONE will be cleared when the IORST signal is generated or when the CLR signal is sent to this device.

--If no error has occurred, DONE will be cleared when the STRT signal is sent to this device. If

some error has occurred, when the STRT signal is sent to this device the BUSY flip flop will be set, the ERROR signal is therefore enables to set DONE. In the last case the STRT signal cannot clear DONE and this provides the means by which a program can check for errors.

6. ERROR DETECTION CIRCUIT (Diagram 4^3)

LATE Flip Flop

This flip flop is set when the LATE error defined at Section III.A is detected.

$$LATE \leftarrow (LATE + ND \cdot DONE \cdot CLOCK) A\phi \cdot \overline{CLR \cdot SELDFS} \cdot \overline{IORST}.$$

The statement shows that the signals $\overline{A\phi}$, IORST and CLR (to this device) can clear this flip flop.

CUT Flip Flop

This flip flop is set when the CUT error defined at Section III.A is detected.

$$CUT \leftarrow [CUT + ON(\overline{NOVA\ DFS}\ \uparrow)] A\phi \cdot \overline{CLR \cdot SELDFS} \cdot \overline{IORST}.$$

NOTE: The notation (NOVA DFS $\uparrow$) denotes the signal $\overline{NOVA\ DFS}$ considered only at the leading edge, after it ($\overline{NOVA\ DFS}\ \uparrow$) = 0 even though $\overline{NOVA\ DFS}$ = 1. In other words whenever the signal $\overline{NOVA\ DFS}$

goes up the logic 1 of ($\overline{\text{NOVA DFS}}$ ↑) exists. Otherwise ($\overline{\text{NOVA DFS}}$ ↑) = 0.

Equally well the trailing edge of a signal can be defined with a notation as (NOVA DFS ↓), and obviously the equation (NOVA DFS ↓) = ($\overline{\text{NOVA DFS}}$ ↑) is true.

IMS Flip Flop

This flip flop is set when an "Improper Synchronization" between the shifting of the MASK-SHIFT REGISTER and the coming of channel data on the DATA LINES is detected.

$$IMS \leftarrow [IMS + (S\phi + S1 + S2 + S3 + S4)(BL\uparrow)]A\phi . \overline{CLR.SELDFS} . \overline{IORST}$$

with:

- (BL↑) is the signal BL considered only at its leading edge
- Sϕ, S1, S2, S3, and S4 are the states of 5 bits of the synchronization counter S defined as below.

Synchronization Counter, S (Diagram 4^3)

The synchronization counter, S, is a 5 bit counter which counts modulo 32.

If DATA = 0	S←0
If DATA.CLOCK = 1	S count 1.

The reader should note these things:

--Any time the content of S is n the mask bit at the

rightmost bit of the MASK-SHIFT REGISTER, BC31, must be MSKn, mask bit of channel n.

--If working properly, whenever a block number comes on the DATA LINES the mask bit MSKϕ must already be at BC31. In other words, whenever the signal BL goes up, the content of the counter S must be zero, or else there is improper synchronization between the shifting of the MASK-SHIFT REGISTER and the coming of channel data on the DATA LINES. Flip flop IMS will be set to indicate this error. The reason for improper synchronization could be parity errors which cause the DATA flip flop to be improperly set, followed by improper shifting of the MASK-SHIFT REGISTER. Another reason might be noise which causes an extra CLOCK signal to be generated on the CLOCK line from the DFS to the NOVA. (A missing CLOCK signal caused by improper reading of the clock bit will also cause an error since the MASK-SHIFT REGISTER will not be shifted for that word.)

ERROR Signal

The signal, ERROR, which was used several times in the previous sections is defined by:

$$\text{ERROR} = \text{LATE} + \text{CUT} + \text{IMS}.$$

7. DATA REGISTER (Diagram 4^3)

When the DONE CIRCUIT recognizes that the data on the DATA LINES is one of the data types requested by the computer, it will set the DONE flip flop at the appropriate CLOCK time (see DONE CIRCUIT). The DONE signal, when going up and if no error occurs, gates the data on the DATA LINES into the DATA REGISTER. So,

If $\overline{\text{ERROR}}$.(DONE↑) = 1 then BB(0-13)←TRF(4-17)
BB14 ←TRFP
BB15 ←TRFB.

When an error has occurred the STRT signal sent to this device will clear the DATA REGISTER (remember the STRT signal also sets BUSY). So,

If ERROR.STRT.SELDFS = 1 then BB(0-15)←0.

However, depending on what error has occurred, some flip flops in this register will be set again, just after they are cleared.

If ERROR.BUSY = 1 then BB7←1 BB15←1 (1)

If (CUT+IMS)BUSY = 1 then BB14←1 (2)

If IMS.BUSY = 1 then BB13←1. (3)

The reader should note these things:

--if the IMS error has occurred then (1), (2), (3) are well satisfied so the content of the DATA

REGISTER, after the generating of STRT, will be 0000000100000111 = JUMP.+7[1]

--If the CUT error has occurred (IMS hasn't) then only (1) and (2) are satisfied, and the content of the DATA REGISTER will be 0000000100000011 = JUMP.+3[1]

--If only the LATE error has occurred then only (1) is satisfied, so the content of the DATA REGISTER will be the instruction JUM.+1 which has the binary value 0000000100000001

--The forming of a JUMP instruction in the DATA REGISTER can't be made if the last necessary data in this register hasn't been responded or hasn't been transferred into the accumulator (because the BUSY flip flop, which was reset by last setting of DONE, is still off. The STRT signal generated in responding to a setting of DONE to clear DONE set BUSY, is usually generated after the transfer of the data from the device to an accumulator[1]).

8. STATUS REGISTER (Diagram 4[3])

The STATUS REGISTER is composed of all the control flip flops of the interface, except DONE and BUSY. The position of every flip flop in this register is shown below:

CCϕ = EDF

CC1 = SDF

CC2 = DATA

CC3 = Sϕ

CC4 = S1

CC5 = S2

CC6 = S3

CC7 = S4

CC8 = NOVADFS

CC9 = IMS

CC10= CUT

CC11= LATE

CC12= INTDIS

CC13= INTREQ

CC14= PEF

CC15= SPEF

where PEF and SPEF are respectively the "parity error flip flop" and the "synchronous parity error flip flop". Their description are shown below:

PEF flip flop

$$PEF \leftarrow (PES+PEF.\overline{CLOCK})\overline{CLR.SELDFS.IORST}$$

This flip flop is used to record the parity error signal, PES, which is generated 16MS after the data word comes on the DATA LINES if this data has a parity error (refer to

Fig. 3-2). 4MS after the PES signal the CLOCK will come to transfer the state of PEF to the SPEF flip flop and clear the PEF flip flop, making it ready for new data.

SPEF flip flop

SPEF←(SPEF+PEF.ND.CLOCK)$\overline{\text{STRT.SELDFS}}.\overline{\text{CLR.SELDFS}}.\overline{\text{IORST}}$.

The reader should note that:

--the transfer from the PEF to the SPEF is conditional, no transfer can be made when the SPEF hasn't been cleared or when the data on the DATA LINES is not data requested by the computer (ND=0)

--with no error, whenever the DATA REGISTER receives data from the DATA LINES, SPEF receives the parity information of this data from PEF. Whenever the DONE flip flop is cleared by a computer program responding to the data (STRT is generated) SPEF is also cleared. These observations show that the parity information of the selected data is always kept with the data at the SPEF flip flop.

IV. SOFTWARE DESIGN

In this section the software system presented in block diagram 4-1 will be described. This system transfers data from the DFS tape unit to the NOVA computer and then transmits stored data to the IBM 360/50. The main body of the system, which consists of the subroutines outlined by the solid line in Fig. 4-1 is contained in the USER MANUAL[3]. The reader can refer to that manual for details. The subroutines outlined by the dashed line are somewhat dependent on the user's exact requirements and are left to the user to accomplish. They will be discussed in Section B however.

A. OPERATIONAL DESCRIPTION (refer to block diagram 4-1)

The operation of the software system will be described block-by-block as follows:

SUB 1. This subroutine performs the miscellaneous functions like: clear the device, start the device (clear DONE set BUSY), send the command word in location COMD to the command register, and send two part of mask B and C in location MSKB, MSKC to the mask shift register (see specification block).

SUB 2: After receiving the command word the command register will decode this word into the control signals to operate the DFS. However these signals

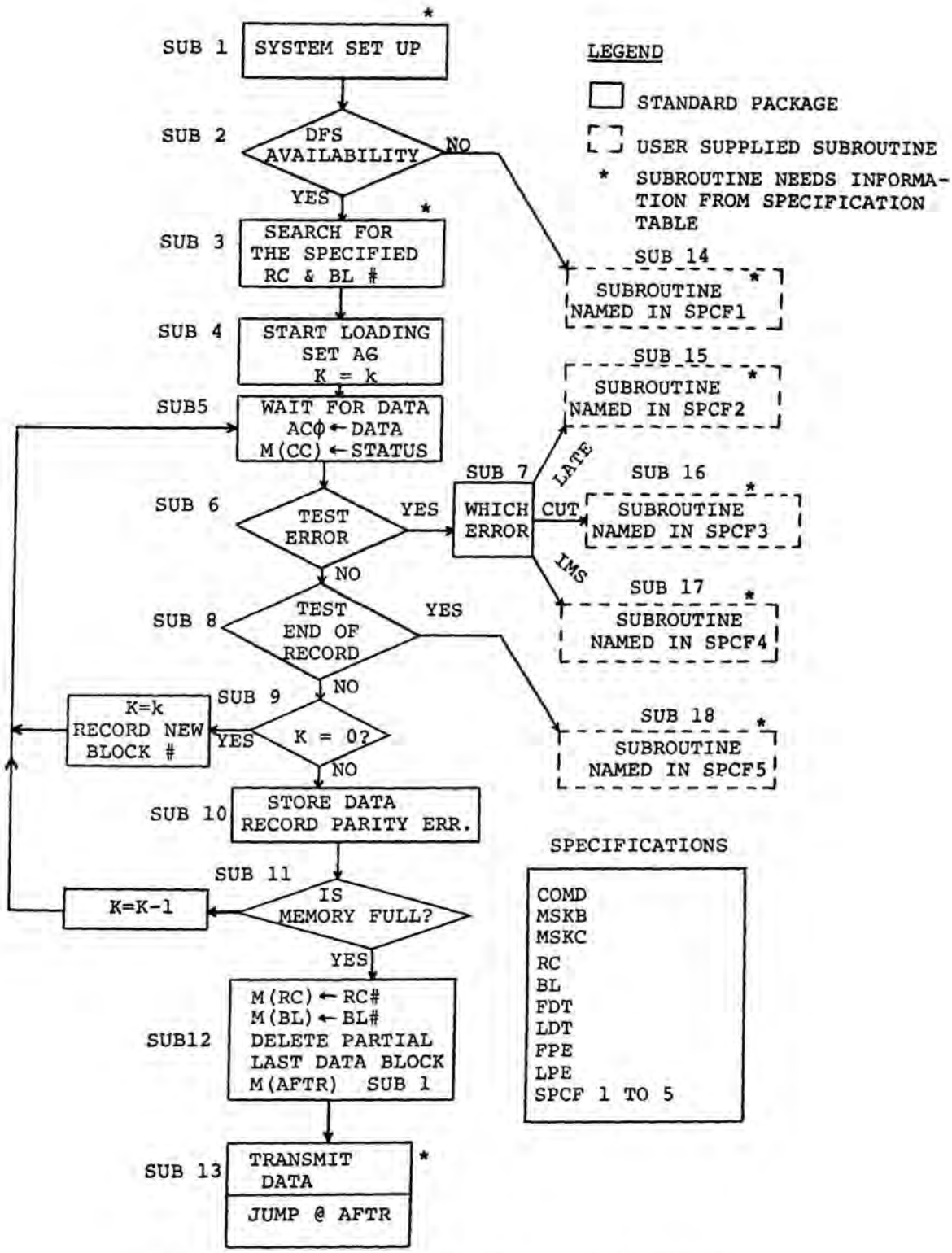


FIG. 4-1 BLOCK DIAGRAM OF SOFTWARE SYSTEM

will be disabled if the NOVA DFS is not set, meaning the DFS is not available for communication. The function of this subroutine is to determine the state of the NOVA DFS flip flop by testing bit 8 of the status word transferred from the status register. If this bit is zero, it means communication to the DFS is not available. In this case the computer will perform the function specified by specification 1, which will be discussed in Section B.

SUB 3: In this subroutine the specified record number, SRC, and the specified block number, SBL, are given in the locations RC and BL. Only data after SBL in the specified record of SRC will be loaded. To start searching for SRC the computer sets A4=1, A5=A6=A7=0 so that only record numbers can set the DONE flip flop to call the processor. The specified record number, SRC, can be zero or non-zero. If zero it means the first record on tape will be identified (there is no zero record number on tape). When SRC = 0 the computer will fill location RC with the first record number encountered, then search for the specified block number, SBL. If SRC $\neq$ 0 the computer, after comparing SRC to the record number first encountered, say FRC, will decide which mode (forward or reverse) to operate the tape transport to find SRC on tape (the record numbers on tape increase in forward direction).

If FRC ≤ SRC it resets A2 to operate the DFS in playback mode (forward).

If FRC > SRC it sets A2 to operate the DFS in search mode (reverse).

All record numbers received are serially compared to SRC until this number is found. After the specified record number has been found the computer begins to search for the specified block number, SBL. It sets A2 = 0 to run the DFS in the playback mode, A4 = 0 A5 = 1 A6=A7=0 so that only the block number can set the DONE flip flop to call the processor. All the block numbers received are compared to SBL until this number is found.

SUB 4: After the specified record and block numbers have been found, the computer sets A6 = 1 to enable unmasked channel data to set the DONE flip flop. Channel 0 is one of the unmasked channels because the computer needs to update the block numbers. A location K, which will be zero every time a block number is received, is set to k at this time. k is the number of unmasked channels excluding channel 0 and will be decremented by one each time a field of data is received.

SUB 5: In this subroutine the computer does nothing more than wait to respond to data. When the requested data comes in, the DONE flip flop is set. In responding to DONE = 1 the processor, after transferring the

data and its corresponding status word in, generates a signal, STRT, to clear DONE making it ready for the next data. The status word is temporarily stored in location CC for further reference.

SUB 6: After generating signal STRT in SUB 5 the computer tests the state of the DONE flip flop again. If it is still on, this means some error has occurred and the processor then jumps to SUB 7. Otherwise it jumps to SUB 8.

SUB 7: To determine which error has occurred the computer transfers in the jump instruction formed in the data register and executes it to jump to the individual subroutine which initiates recovery from the error. If the LATE error has occurred the computer will perform the function specified by specification 2 (all specifications will be mentioned in Section B). A similar operation occurs with the other errors.

SUB 8: No error has occurred and the computer tests bit 0 of the status word, which represents the state of EDF flip flop, to know whether the data just received in SUB 5 is an end of data word or not. If Bit 0 = 1, EDF = 1, the word is an end of data and the record is finished. To know what to do next the computer will refer to specification 5.

SUB 9: This subroutine tests whether the data is a block number or not. If K = 0 the data is a block number

and the computer records this new block number as the number after next block to be loaded. If $K \neq 0$ the data is a field data word and the program proceeds to SUB 10.

SUB 10: In this subroutine the field data are sequentially stored in the memory from the location whose address is specified by location FDT, and the updated address of the latest data is recorded in location LDT. The computer also records the parity error information of the data. The addresses of data having parity errors are sequentially recorded in the locations pointed to by location FPE. Location LPE is used to contain the address of the location recording the address of the latest data having parity error (see Section B description of these locations).

SUB 11: This subroutine tests whether the memory is full or not. If full it goes to SUB 12 and if not it jumps back to SUB 5 to continue loading data, after decrementing location K by 1.

SUB 12: When the memory is full the computer resets A1=0 to stop the DFS, puts the record number of the present record into location RC and puts the latest block number into location BL. These numbers will be the specified record and block number for the next loading. All the data stored after this block number are deleted by correcting the address of the latest data contained in location

LDT. Finally it puts address "SUB 1" into location AFTR. This location is used to specify where the processor will jump to after transmitting data to IBM 360/50.

SUB 13: This subroutine does nothing more than transmit data stored in the NOVA to the IBM 360/50. It refers to the locations FDT and LDT for the range of data stored and locations FPE and LPE for the range of parity error information of these data. After transmitting all data to the IBM 360/50, the NOVA executes the instruction JUMP @ AFTR to know what to do next. If the loading has not been finished, address "SUB 1" was put in location AFTR by SUB 12, then the computer will jump back to continue loading data with the necessary record and block numbers already available in the locations RC and BL.

B. SPECIFICATIONS

All the specifications are contained in the locations described below:

1. LOCATION RC

To load data of any record the user should give the computer the record number of this record in LOCATION RC. The record number specified can be zero or non-zero. If

non-zero the computer will search for the record number on tape matching the specified number. If zero it takes the record number first encountered on tape as the desired record. This specification is permitted because there is never a zero record on tape.

2. LOCATION BL

This location contains the specified block number, SBL, which the computer has to search for in the specified record. A zero SBL has a similar meaning to the non-zero SBL and is the first block number of the record (see tape format).

3. LOCATION COMD

This location contains the command sent to the COMMAND REGISTER at the beginning of operation. The first command given by the user should have the format below:

1	1	U	U	0	0	0	0	0	0	X	X	X	X	X	X
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Bit 2 Bit 3 depends on the application (see meaning of A2, A3)

Bit 10 to 15 are "don't care"

Bit 8 = Bit 9 = 0 for normal transfer (see description of DATA-OUT-A instruction)

Bit 4 = Bit 5 = Bit 6 = Bit 7 = 0 so that no type of data has been requested, the computer will decide which type to request later on.

Bit 0 = 1 to indicate "ON"

Bit 1 = 1 to run the DFS.

4. LOCATION MSKB AND MSKC

LOCATION MSKB contains part B of mask (Channel 0 to 15) and LOCATION MSKC contains part C of mask (Channel 16 to 31). The unmasked channels must have their mask bits 1. For instance suppose all channels are masked except channel 0, 13 and 15 then

M(MSKB) = 1000000000000101

M(MSKC) = 0000000000000000

NOTE: M(X) is read "the content of location X".

5. LOCATIONS FDT, LDT, FPE, LPE

These locations supply the necessary information to the transmitting subroutine, SUB 13, with:

M(FDT) = address of the first location of the table containing data

M(LDT) = address of the last location of the table containing data

M(FPE) = address of the first location of the table

containing address of data with parity errors

M(LPE) = address of the last location of the table containing address of data with parity errors.

EXAMPLE: M(FDT) = 1000 M(100) = 1500

M(LDT) = 3000 M(101) = 1700

M(FPE) = 100 M(102) = 1701

M(LPE) = 103 M(103) = 1702.

This means:

a--2000_8 data words have been stored from location 1000 to location 3000

b--4 data words have parity errors, their addresses are stored from location 100 to location 103 and their addresses are 1500, 1700, 1701, 1702.

6. LOCATION SPCF1TO5

This location contains the specifications 1, 2, 3, 4, 5 combined together into a single word. The format of this word will be shown below. The way to set the specifications 1, 2, 3, 4, 5 presented here is just an approach proposed to the user. The user should improve these things for his specific application.

The specifications 1 to 5 are nothing but the orders given by the user to tell the computer what to do if the corresponding case shown on the diagram happens.

Specification 1

When the communication between the NOVA and the DFS is not available, the user might want to tell the computer to wait, halt or do other tasks. These desires might be

varied from one time to another. To save the user from changing his program for each case, all subroutines used for all possible applications must always be available in the computer. Each time he changes his mind he need only give a specific code to tell the computer what he wants when this case, the communication to the DFS is not available, happens. The computer will test his code to jump to the proper subroutines.

Specification 2, 3, 4

Similary to specification 1, the possible orders of the user in the case of each error might be: start the operation over, continue loading data, recover from the error, etc.

Specification 5

Similarly to specification 1, the possible user's orders when the end of a record is reached might be: stop loading, continue loading data of the next record on tape or the next record in a given sequence of records, etc.

A code with the format below can be used to specify all the specifications above:

	SPCF1		SPCF2			SPCF3		SPCF4				SPCF5			
Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

This format gives 2^2 = 4 codes to specify specification 1,

2^4 = 16 codes to specify specification 2,

2^2 = 4 codes to specify specification 3,

2^4 = 16 codes to specify specification 4,

2^4 = 16 codes to specify specification 5.

V. SUMMARY

It is worthwhile to review some main characteristics of the interface system and consider some problems which the user must face in the data processing.

The hardware system handles the major part of the data recovery. With an 8 bit COMMAND REGISTER the computer can give 2^8 possible commands to the interface. The interface controls the tape transport, recognizes the data coming in, selects the data requested by the computer, detects errors, and calls the central processor when the requested data comes in or when an error has occurred. The computer usually does nothing more than transfer the data in, store it, or make a decision for recovery from some error. Since the hardware system performs most of the tasks, this shortens the length of the program and saves time of program execution. The computer, therefore, can reserve more locations for data buffers and always get ready in time to respond to new data.

The Interface system can be diagnosed easily by the computer. The computer at any time can acknowledge the states of all the flip flops (including the registers) of the interface. A 16 bit STATUS REGISTER can always report to the computer the necessary information about the current status of the data and the interface. This would help the computer to make the precise decisions for correcting some problem.

Two main problems in this data processing which the user must consider are the execution time of subroutines in the software system and parity errors. The execution time of several subroutines should not be so long that the computer will be late in responding to the data. After sensing incoming data, the computer can be programmed to transfer the data in, test for errors, test end of data, record parity error information of the data, store the data into the memory and then come back to wait for more data. The total time for this should be less than 32MS if two consecutive channels can have their data stored, and if the tape transport selects the HI SPEED. (In LOW SPEED time between two consecutive words is 64MS.) This condition is satisfied with the subroutine proposed in the USER'S MANUAL[3] which takes care of the things just listed.

Regarding parity errors, normally the DFS tape unit can supply data with very few parity errors, typically 5 or 6 errors per record. Therefore, it is usually not a big problem. However, if in some unexpected cases there are so many parity errors that the interface can misinterpret the data and control information, the programmer may need a subroutine to check the data received by the interface to be sure that the data received is one of the type requested by the computer. In any case, a good software system coordinated with the hardware system will handle the data processing efficiently. The software system presented in the USER'S MANUAL[3] is available to support

the hardware system for a continuous loading and transmitting data from the DFS, through the NOVA, to the IBM 360/50. All what the user usually needs to do is to supply the NOVA enough information about data which he wants to transfer. This information was mentioned as the specifications in Section IV.B.

BIBLIOGRAPHY

1. Data General Corporation (1970). How to Use the NOVA and the SUPERNOVA. Southboro, Massachusetts.

2. Texas Instruments, Incorporated. Digital Field System Instruction Manual. No. 180138 (1965. Book 1 and 2). Houston, Texas.

3. Chung, Cao-van. User's Manual of the DFS Interface, Technical Report CRL 72.7 (September 1972) Rolla, Missouri.

4. Whitesitt, J. E., Boolean Algebra and Its Application. Addison-Wesley, Reading, Massachusetts. 1961.

5. McCluskey, E., "Minimization of Boolean Functions," Bell System Tech. J., 1956, 35, pp. 1417-1444.

6. Lewin, Douglas, Logical Design of Switching Circuits, American Elsevier, Inc., New York, 1969.

7. Moore, E. F. (Ed)., Sequential Machine: Selected Papers, Addison-Wesley, Reading, Massachusetts, 1964.

8. Huffman, D. A., "The Synthesis of Sequential Switching Circuit," J. Franklin Institute, 1954, 257, 161-190 (March), 275-303 (April).

9. Unger, S. H., Asynchronous Sequential Switching Circuits, Wiley-Interscience, New York, 1969.

10. Bartee, T. C., Theory and Design of Digital Machines, McGraw-Hill, New York, 1962.

11. Sobel, Herbert S., Introduction to Digital Computer Design, Addison-Wesley, Reading, Massachusetts, 1970.

12. Pottinger, H. J. and Tracey, J. H., Formal Description of the SCC-650 Computer, Technical Report CRL68.1 (August 1968) Rolla, Missouri.

13. Chu, Y., "An Algol-like Computer Design Language," Communications of the ACM, Vol. 8, No. 10, pp. 607-615, 1965.

14. Schlaeppi, H. P., "A Formal Language for Describing Machine Logic Timing and Sequencing (Lotis)," IEEE Trans. Electron. Comput., 1964, EC13, 439-448.

VITA

Chung, Cao-van was born on April 15, 1947 in a remote village named The-Chi-dong-VIETNAM. He graduated as an electric power engineer from Saigon Electrical Engineering College in 1969. He worked for Vietnam Power Comapny from July 1969 to December 1970.

He has been enrolled in UMR since January 1971. During this time he has been a Graduate Assistant in the Electrical Engineering Department.

He is married to Nguyen-thi-ngoe-Bich in SAIGON.

APPENDIX. SIGNALS (ALPHABETICALLY LISTED)

SIGNAL	SECTION
Aϕ-A7	III.D.1
Bϕ-B15	III.A.4,III.C.3
BBϕ-BB15	III.7,III.D.7
BCϕ-BC31	III.A.4,III.D.4
BL	III.D.3
BUSY	II.B,III.A.5,REF (1)
Cϕ-C15	III.A.4,III.C.4
CCϕ-CC15	III.D.8
CLOCK	III.B.1
CLR	III.C.8,REF (1)
CUT	III.D.6
DATA	III.D.3
DATA ϕ-DATA15	REF (1)
DATIA	III.C.2,REF (1)
DATIB	III.C.5,REF (1)
DATIC	III.C.6,REF (1)
DATOA	III.C.1,REF (1)
DATOB	III.C.3,REF (1)
DATOC	III.C.4,REF (1)
DONE	III.D.5
(DONE $\uparrow$)	SEE DEFINITION OF ($\overline{\text{NOVA DFS}}$ $\uparrow$)
DSϕ-DS5	REF (1)