CONTROL DATA 6600

TRAINING MANUAL

CONTROL DATA
CORPORATION

CONTROL DATA 6600

TRAINING MANUAL

CONTROL DATA
CORPORATION

60147400

Record of Revisions

REVISION	NOTES
A	Pages 3-11, E-10, E-11, E-12, E-13 and A-2 through
(6-17-65)	A-14 revised. A-15, A-16 and A-17 added.

CONTROL DATA 6600

TRAINING MANUAL

This manual is intended primarily for training use. It obsoletes Pub. No. 60127400. The information contained herein is subject to correction and change. Corrections and suggestions should be brought to the attention of the supervisor of computer training, Control Data Institute.

60147400
June, 1965
Printed in the United States of America

CONTENTS

SECTION I

Central Processor Address Control

ADDRESS CONTROL

INTRODUCTION

The basic concept of the 6600 computer is parallelism. In the central processor many parallel operations are in progress. One of these operations is accessing central memory. In many cases the peripheral processors will be attempting communications with central memory simultaneously with the central processor. This is a simplification of possibilities but points out that some method of orderly distributing requests for central memory time must be available.

IMPLEMENTATION

Central Memory is divided into 32 independent banks that can be accessed sequentially in an overlapping fashion. This allows memory references every 100 nano-seconds. In order to make full use of central memory and also be able to fulfill the parallelism concept, there is an area of the central processor known as the STUNT BOX. It is to this area of circuitry that all references to central memory must first come. The job, therefore, of the stunt box is to collect requests for central memory access and to distribute these requests in an orderly fashion.

CONDITIONS

The 6600 has ten peripheral processors that can request reading or writing in central memory. The peripheral processors can also execute an exchange jump which requires central memory access. The central processor can reference memory for an instruction word or an operand.

Consequently the peripheral and central processors can send requests to the stunt box simultaneously. This presents the necessity for some means of priority within the stunt box and also some method of naming each of the memory requests in order that the data, once acquired, will know where it must go. Another factor that must be considered is some means of remembering addresses when the first attempted access is rejected because of a memory conflict.

SUMMARY

In general, the stunt box:

a) allows several simultaneous memory requests.

b) establishes a priority for the issue of these addresses to central memory.

c) issues the addresses to memory at a rate that will make maximum use of the 32 independent banks.

d) remembers addresses that have not been accepted by the memory and must be re-issued.

e) adds a tag to the addresses to correctly distribute the data.

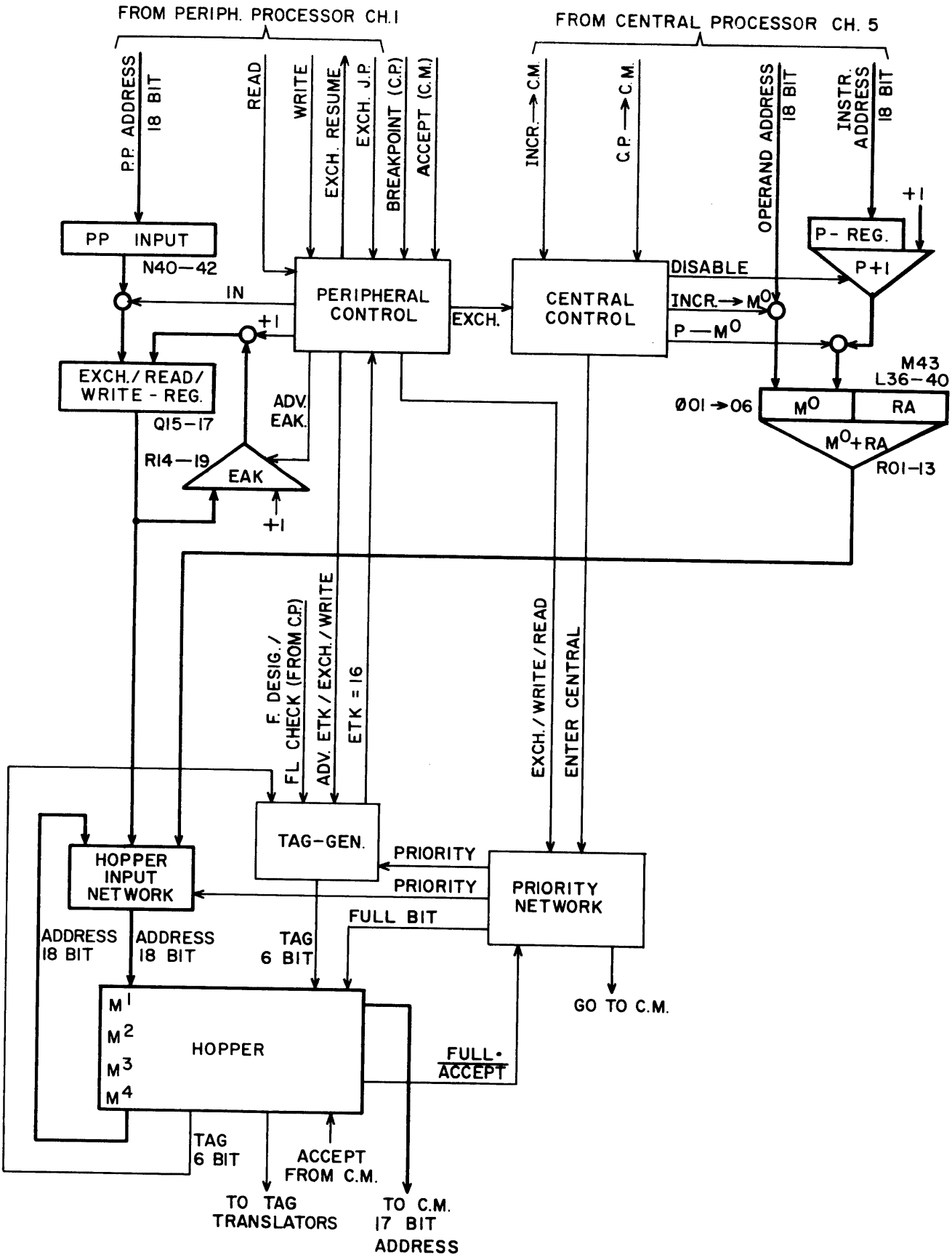


Fig. 1-1. Stunt Box Block Diagram

STUNT BOX

DESCRIPTION:

There are two main address flow paths to the hopper (M_1) and to Central Memory (CM): (Refer to fig. 1-1.)

1) The PPs supply CM with the write and read addresses for the data-exchange between PPs and CP. The exchange jump addresses are also sent over this line.

2) The central processor supplies CM with either operand or instruction addresses. Addresses comming through this path are always added to the content of the Reference Address (RA) Register.

A priority network controls the entry into the hopper (M_1) from which the addresses are sent to CM. These addresses are also stored and circulated in the hopper, from which, in case of bank conflict, they are re-issued to CM after 300 ns.

OPERATION

Peripheral Control

Peripheral Control, for the purposes of this discussion, is a term applied to that section of the central processor that handles addresses and signals from the peripheral processors requesting storage access. (Refer to the block diagram in fig. 1-2.)

When the peripheral processors send an address to the stunt box, an accompanying signal informs Peripheral Control whether it is a Read, a Write, or an Exchange Jump address. Peripheral Control (PC) then transfers this information to the tag generator to enable the IN-path to the EXCH/READ/WRITE Register. In

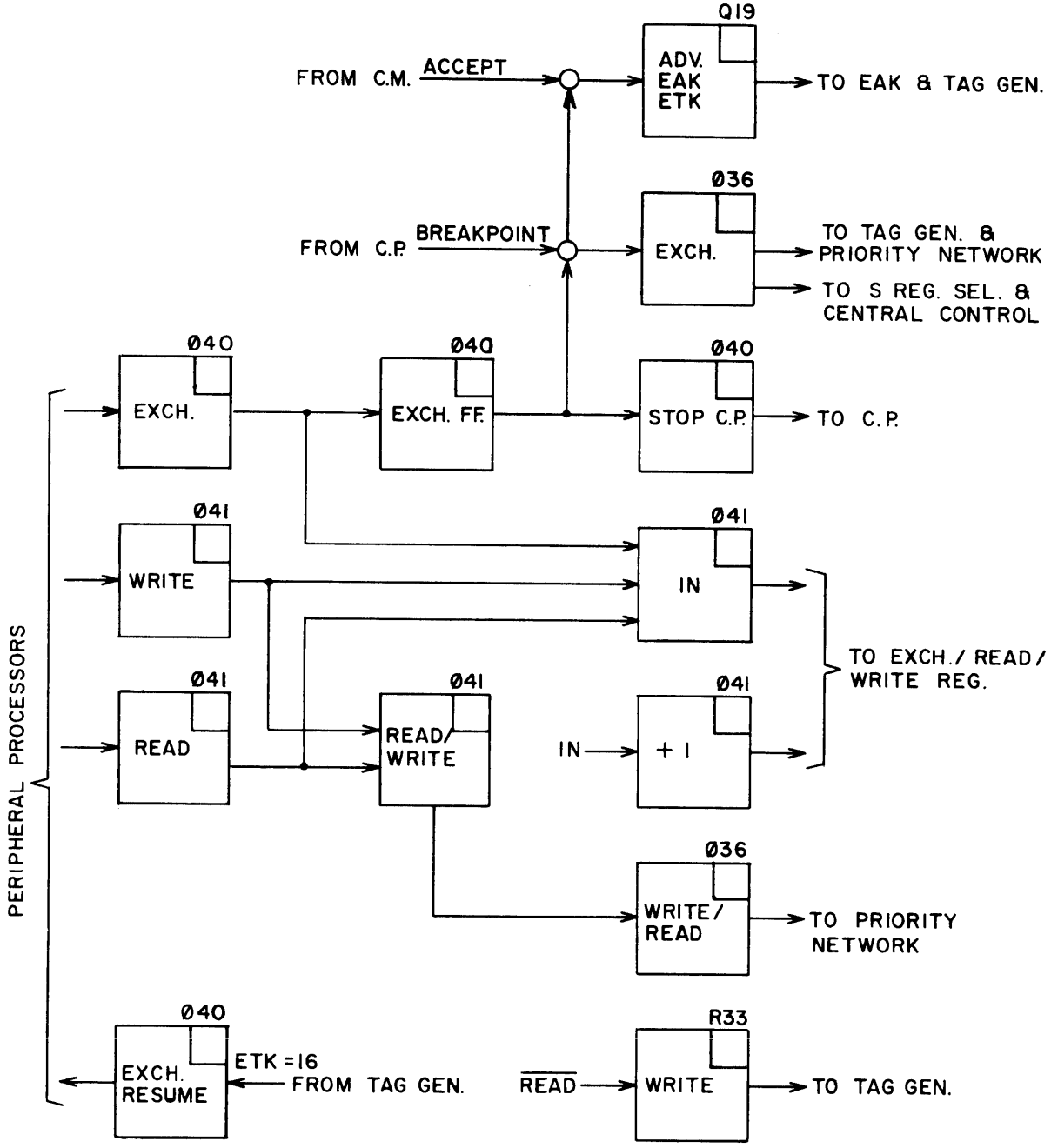


Fig. 1-2. Peripheral Control Block Diagram

an Exchange Jump, this signal also stops the central processor; when the "Breakpoint" signal indicates the central processor and central memory have stopped, it starts Exchange Address (EAK) and Exchange Tag Counters (ETK). The Exchange Address Counter then updates the address in the EXCH/READ/WRITE register for each step of the Exchange process upon receipt of the "accept" from central memory. When the Exchange Tag Counter = 16, Peripheral Control sends an "Exchange Resume" back to the peripheral processors (Write Resume is sent back from central memory chassis 2, and Read Resume from chassis 4).

Central Control

Central Control, for the purposes of this discussion, is a term applied to that section of the central processor that handles addresses from the central processor requesting access to central memory for instructions or operands. Central Control controls entry into M^0 and requests entry (via the priority network) into M^1. (Refer to fig. 1-3.)

When an operand address is sent by one of the two increment units, the INCREMENT ADDRESS FF on P37 will set and the entry signal into M_0 is enabled as soon as M_0 is empty. Simultaneously, the ENTER CENTRAL FF sets indicating to the priority network that a central processor address is waiting to enter central memory.

When a program address is ready in P, the PROGRAM ADDRESS FF sets, enabling the $P \rightarrow M_0$ signal if M_0 is empty and if no operand address is waiting. This signal also sets the ENTER CENTRAL FF.

If a Return Jump instruction or an End Exchange signal occurs, the DISABLE P + 1 FF sets and the program address in P passes through the P-incrementor without being incremented. This does not affect the ADVANCE P gates on N37-N39.

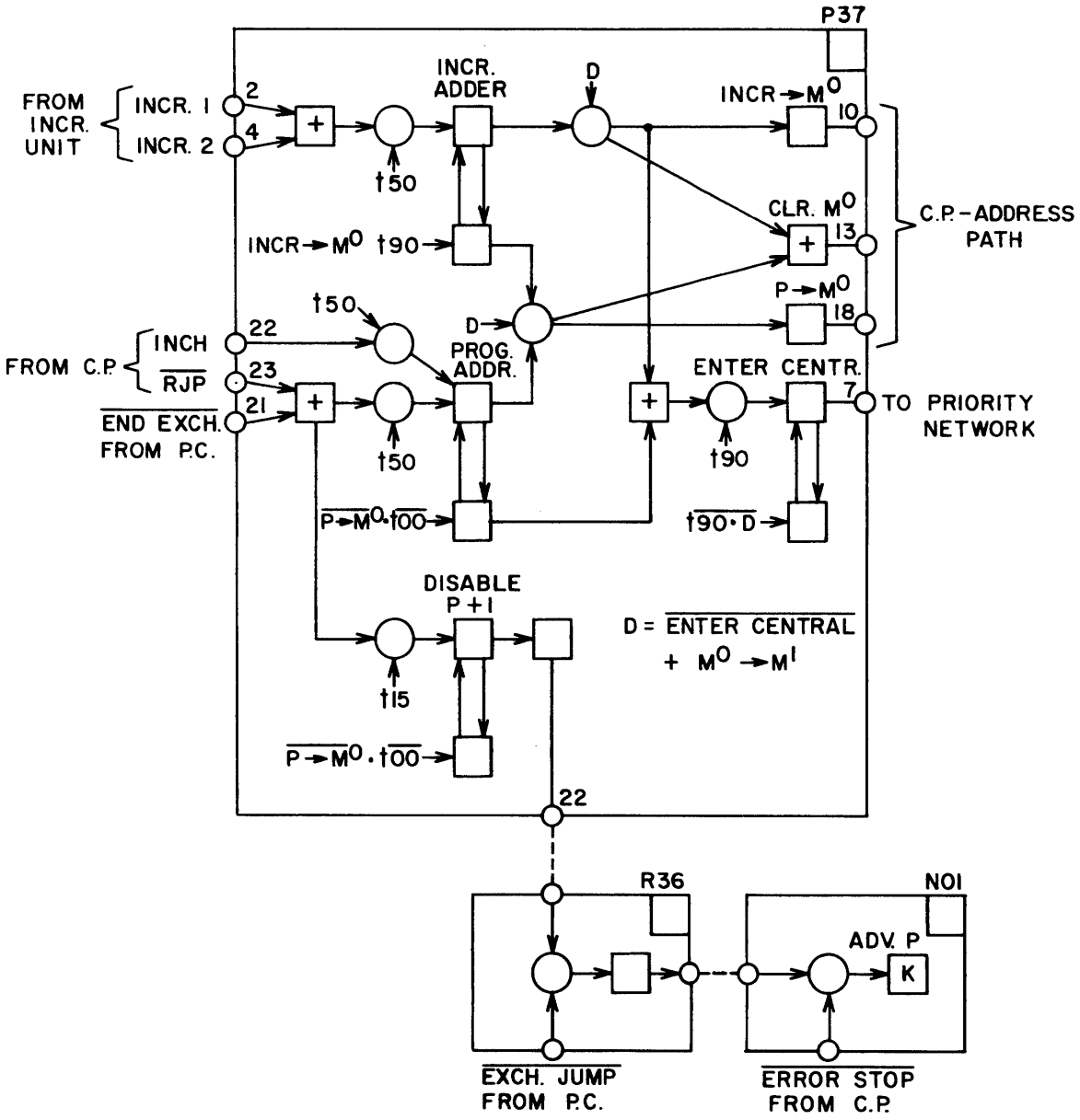


Fig. 1-3. Simplified Central Control Network

Priority Network

The priority network (fig. 1-4) controls inputs to the hopper by sequencing entry to the hopper when more than one address attempts to enter at the same time.

The fixed order of priority is as follows:

1) Address from Hopper

2) Addresses from the Central Processor

3) Addresses from the Peripheral Processors

An address from the hopper is given first priority since it is an un-accepted address due to a central memory bank conflict.

Addresses with second priority are from the central processor (i.e., M^0). Since, for the central processor, storage modes cannot be mixed in the hopper, the Read or Write tags are examined before priority is granted. In attempting a Read, no Write address is allowed in M^1 or M^4. In attempting a Write, no Read address is allowed in M^1 or M^4. If modes are mixed, priority is not granted and entry of the address into the hopper is delayed until central memory has accepted those addresses and modes are no longer mixed.

Addresses from the peripheral processors are assigned lowest priority. Thus, peripheral read and write operations from and to central memory may have to wait for hopper and central processor addresses. An important exception occurs during an Exchange Jump. An Exchange Jump a) stops the central processor, and b) inhibits communications between the peripheral processors and central memory. In this case, therefore, exchange jump addresses are the only addresses entering the hopper.

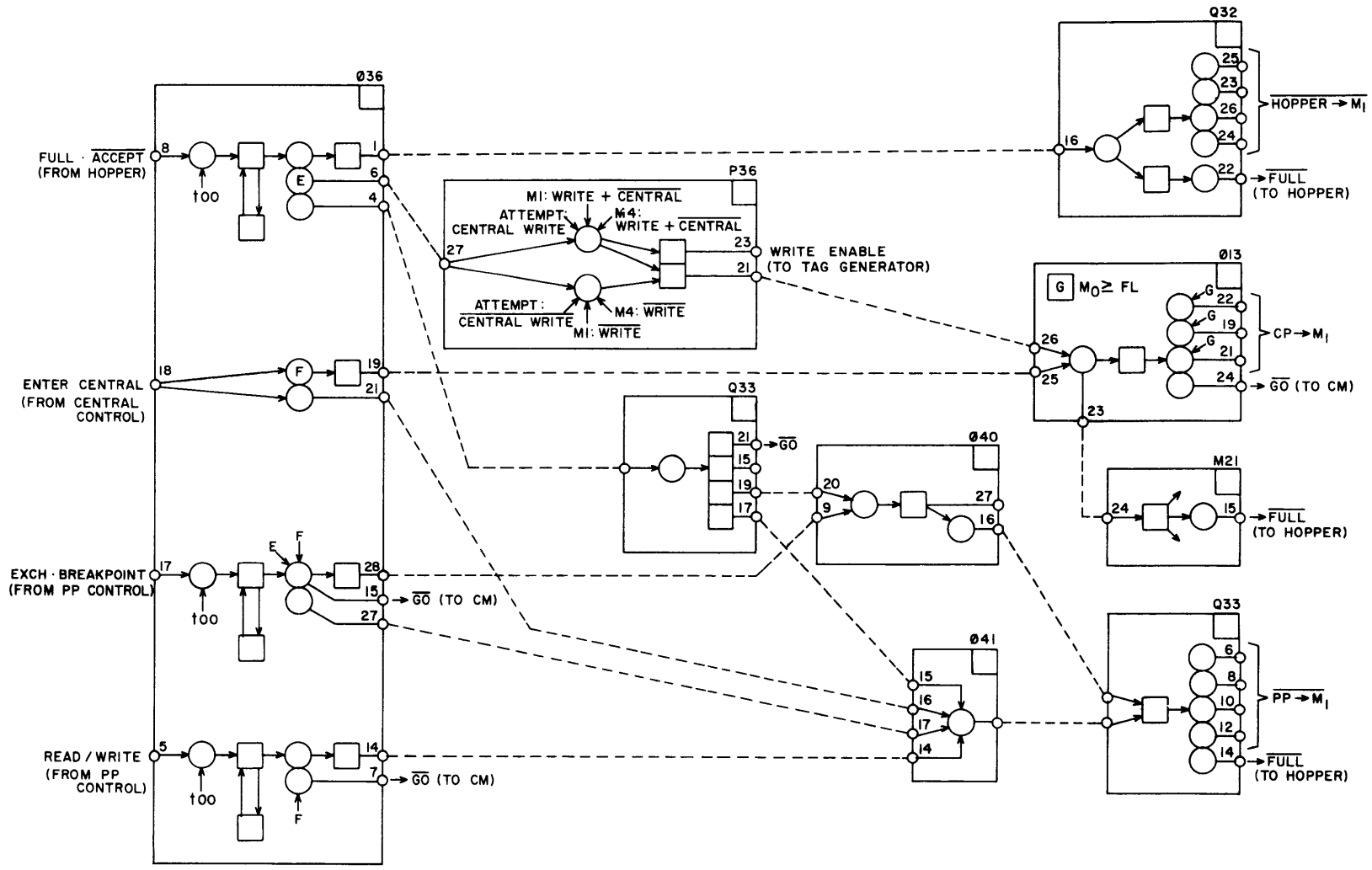


FIG. 1-4 PRIORITY NETWORK

With each address sent to the hopper, a Full bit is generated (to indicate the hopper register contains a usable address) and a Go signal is sent, along with the address, from M^1 to central memory.

The hopper input network is diagrammed in fig. 1-5. Three gates for each of the 18 address bits accommodate the three possible input paths to the hopper. Entry via these gates is controlled by the priority network.

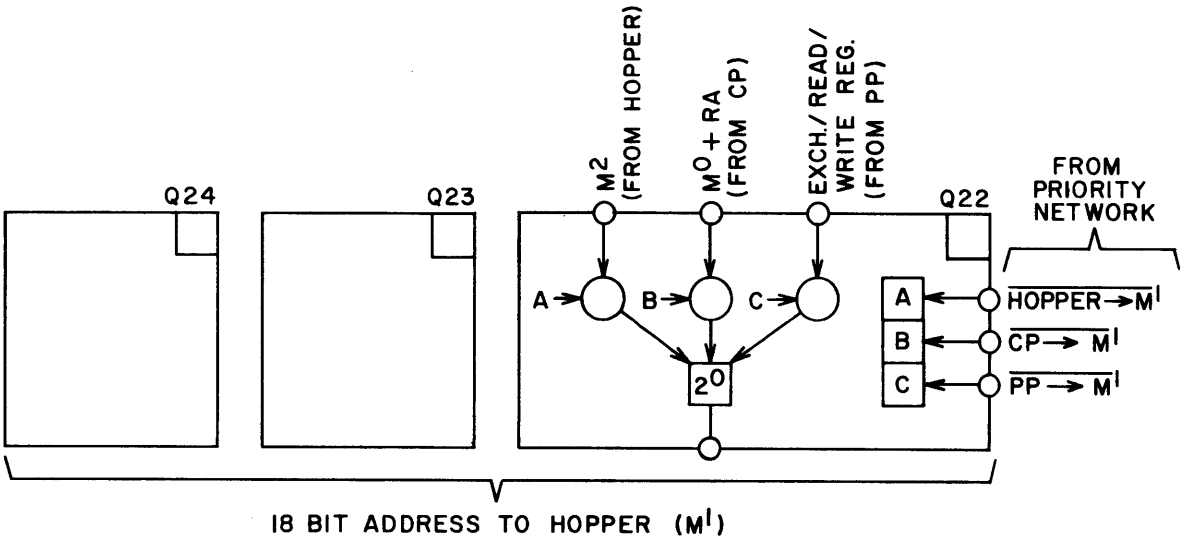


Fig. 1-5. Hopper Input Network

Tag Generator

When an address enters the hopper, a six-bit tag is appended to control the address and data flow. Depending on the source of the addresses, these tags are generated from three sources: (Refer to fig. 1-6.)

1) An un-accepted address due to a bank conflict retains its tag that was generated when the address first entered the hopper.
2) An operand or instruction address from the central processor gets its tag from the translation of the F- designators of the increment units or from the central processor (in case of exit mode stops or return jumps) and from the priority network.
3) Addresses from the peripheral processors obtain their tags from Peripheral Control. The Exchange Tag Counter (ETK), which controls the execution of the exchange jump, generates the tags for all addresses of the exchange jump package.

Hopper

In general, the hopper consists of four registers (M^1, M^2, M^3, and M^4), each capable of holding an 18-bit address, a 6-bit tag, and a single Full bit. (M^2 is an exception and does not have a Full bit). (A block diagram of the hopper is diagrammed in fig. 1-8.)

An address is sent to central memory from hopper register M^1. Hopper registers M^2 - M^4 store the address in case it must be re-issued because of a bank conflict. If the address is accepted by central memory, it drops out of M^2.

In the case of bank conflict, the priority network gates the un-accepted address from M^2 back into M^1 every 300 nanoseconds, until it is accepted by

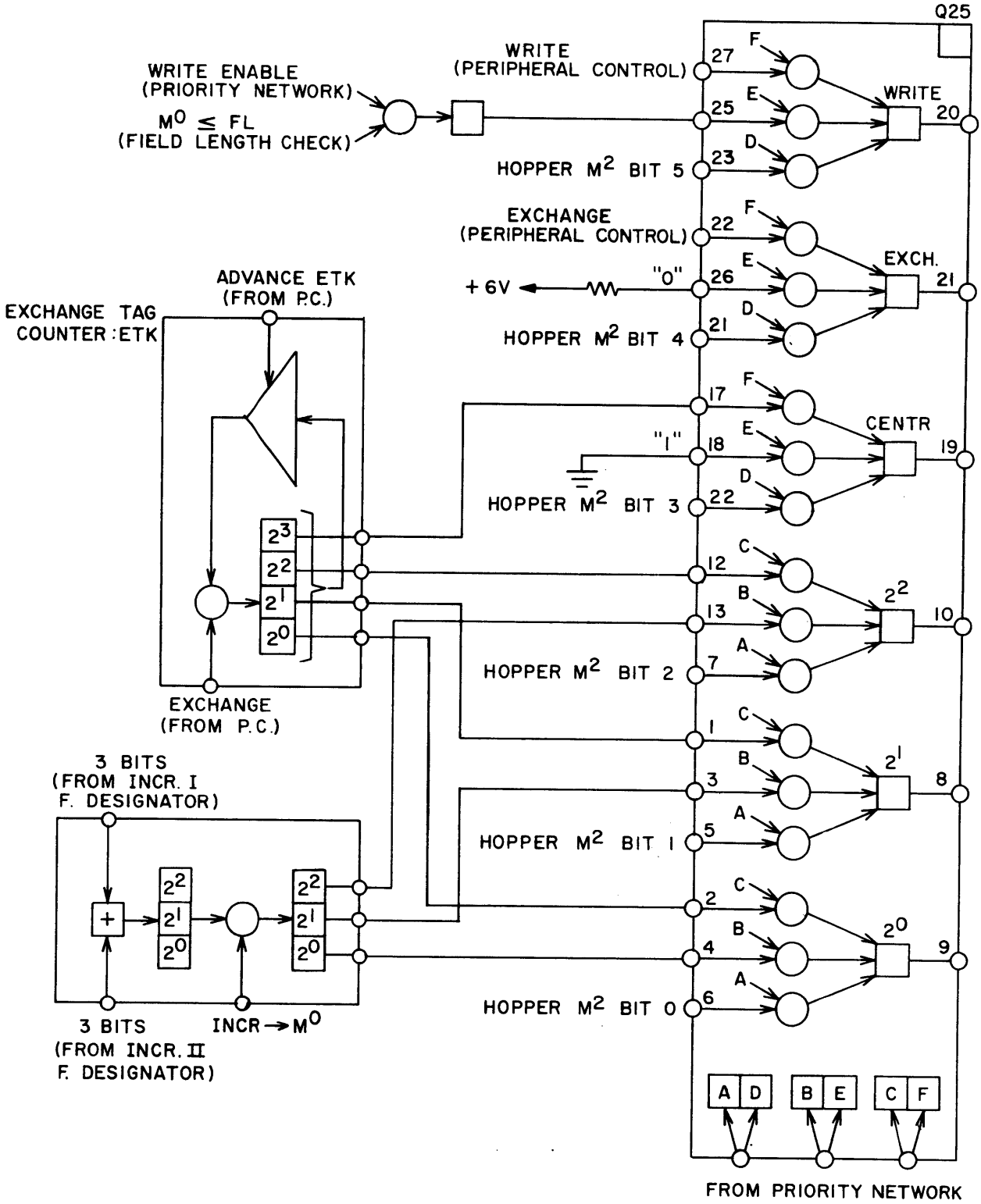


Fig. 1-6. Tag Generator

central memory. An address can be accepted only if the specified bank is free at the time the address is in M^1. Otherwise, it's possible for another address to request access to the same bank and tie it up for a memory cycle. Fig. 1-7 shows possible waiting times in the hopper for worst cases of bank conflict.

The six tag bits travel through the hopper with each address. (The hopper serves as a delay line for the tag.) This line is extended by two additional registers (see fig. 1-8). In each step, the tag controls address and data flow.

Hopper registers M^1, M^4, and M^3 have Full bits associated with the address. The purpose of the Full bit is to indicate to the priority network that the address must be reissued to central memory if no accept is returned. Note that the Full bits also are sampled to stop the central processor (fig. 1-8).

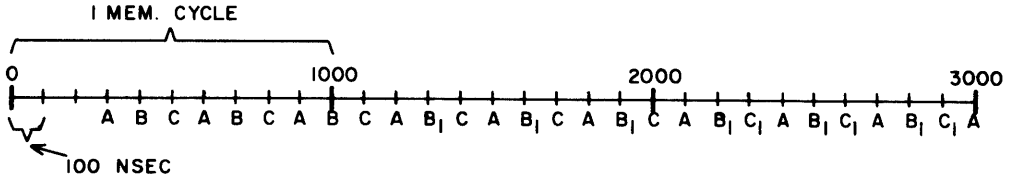


Fig. 1-7. Hopper Waiting Time for Bank Conflicts

The Hopper can have a max. of 3 addresses. An address must be in M1 at the time when the memory cycle for its bank just finished so that it can be accepted by memory, otherwise bank conflict. The longest time an address will have to wait to be accepted in memory is when the address enters M1 300 ns after the memory cycle started.

Example: Let us say all 3 addresses in the Hopper need the same bank, then at t=1000, address B is accepted and a new address B_1 can enter the Hopper, at t=2000 address C and at t=3000 address A will be accepted. (See timing diagram.)

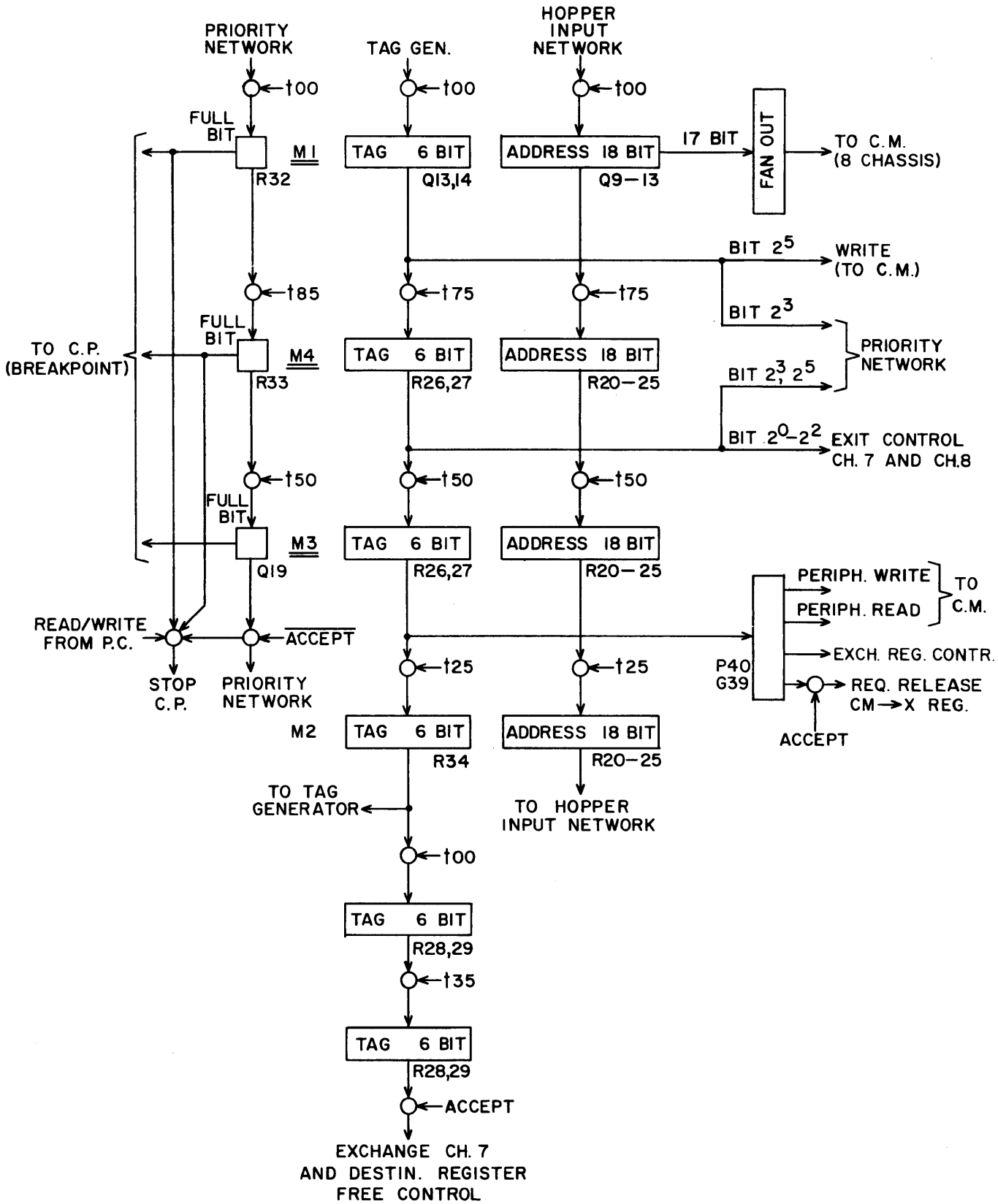


Fig. 1-8. Hopper

Exchange Jump

As an aid in following the somewhat detailed discussion which follows, refer to the central processor diagrams, the timing diagram in fig. 1-9, and the Exchange Jump diagram in fig. 1-10.

The following discussion assumes the Exchange Jump Package (see Reference Manual) is stored in Central Memory starting at address N. To start the Exchange Jump, a Peripheral & Control Processor will send an Exchange Jump pulse together with the 18 bit address N to the Stunt Box.

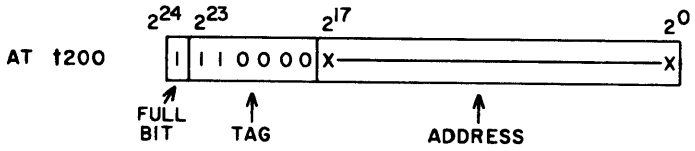
The address will be received by the Input Register (N40, 41, 42) and the Exchange pulse sets FF 040/TP5 at starting time 00. The "0" out of 040/P19 enables the A gates on Q27, 28, 29. At time 50 the contents of the Input Register are transferred to the Exchange/Read/Write Register Q15, 16, 17. At t65, 040/TP2 will be set, setting F37/TP3. If the computer is not running when we start our Exchange Jump, we cannot make the AND gate next to F37/TP3 because we don't have an Issue. However, a "0" from F37/P24 entering I01/P21 and a "1" into I01/P26 are AND'ed giving a "0" into F37/P8 and therefore a "1" from F37/P7. If the computer is running the AND gate at the output of F37/TP3 is not made until an Issue signal occurs and the parcel counter = 1. The output of pin 10 then becomes a "0". This "0" clears the GO FF on G30, stopping the Central Processor. The "1" from F37/P7 will be passed on to F26/P12. If the Hopper is empty and no Branch or REgister reservations are up we will get a "1" into F26/P10 makes the AND gate and sets F26/TP4, placing a "0" into 040/P10. With the t75 pulse, 040/TP1 sets, disabling the A gates on Q27, 28, 29.

This enables a "1" from 040/P13 to set the Initiate Exchange FF on 036/TP6

at t100. A "0" from 036/P15 and a t165 pulse sends a GO signal to central memory to prepare the path for the address being sent. Also at t200 a 60 tag from Q25 of the Exchange Tag Counter is sent to M^1.

The Initiate Exchange FF also sends a "1" to priority control, which, in turn, enables the C gates on Q22, 23, 24, 25. At t200, the contents of the Exchange/Read/Write Register are transferred to M^1 in the Hopper. During the time the address waits to enter M^1, a path from the Exchange/Read/Write Register thru the Exchange Address Counter allows the address to circulate without being incremented until it can be accepted by M^1.

A bit called the Full bit is sent from the priority control to M^1. This bit is only used in the Hopper to indicate that the Hopper contains a usable address. The M^1 Register should now contain the following bits:



At t235, the address is sent from M^1 to all banks of central memory. As the largest possible address in central memory can be specified by 17 bits, the bit position 2^{17} in M^1 is not sent to central memory. At t275, all 25 bits of M^1 start their cycle thru the Hopper (M^1 to M^4 to M^3 to M^2). Once our address has been accepted by central memory, an accept signal is sent back (130 ns after sending out M^1 to central memory) preventing our address from reentering M^1. From M^4, M^3 and M^2 the tag bits are extracted and sent to the Exit control of the Operating Registers and to the Tag Translators.

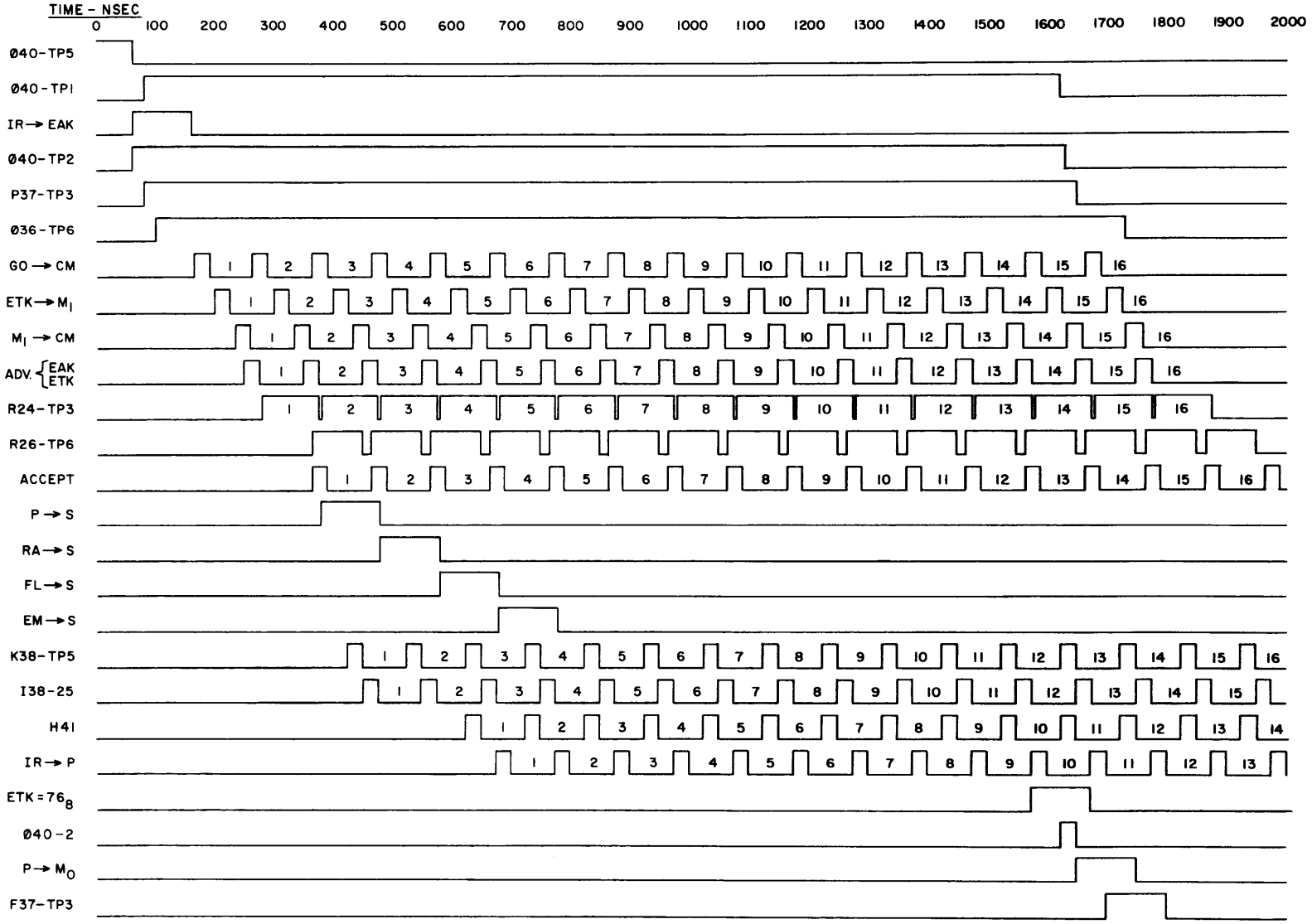


FIG. 1-9 EXCHANGE JUMP CONTROL

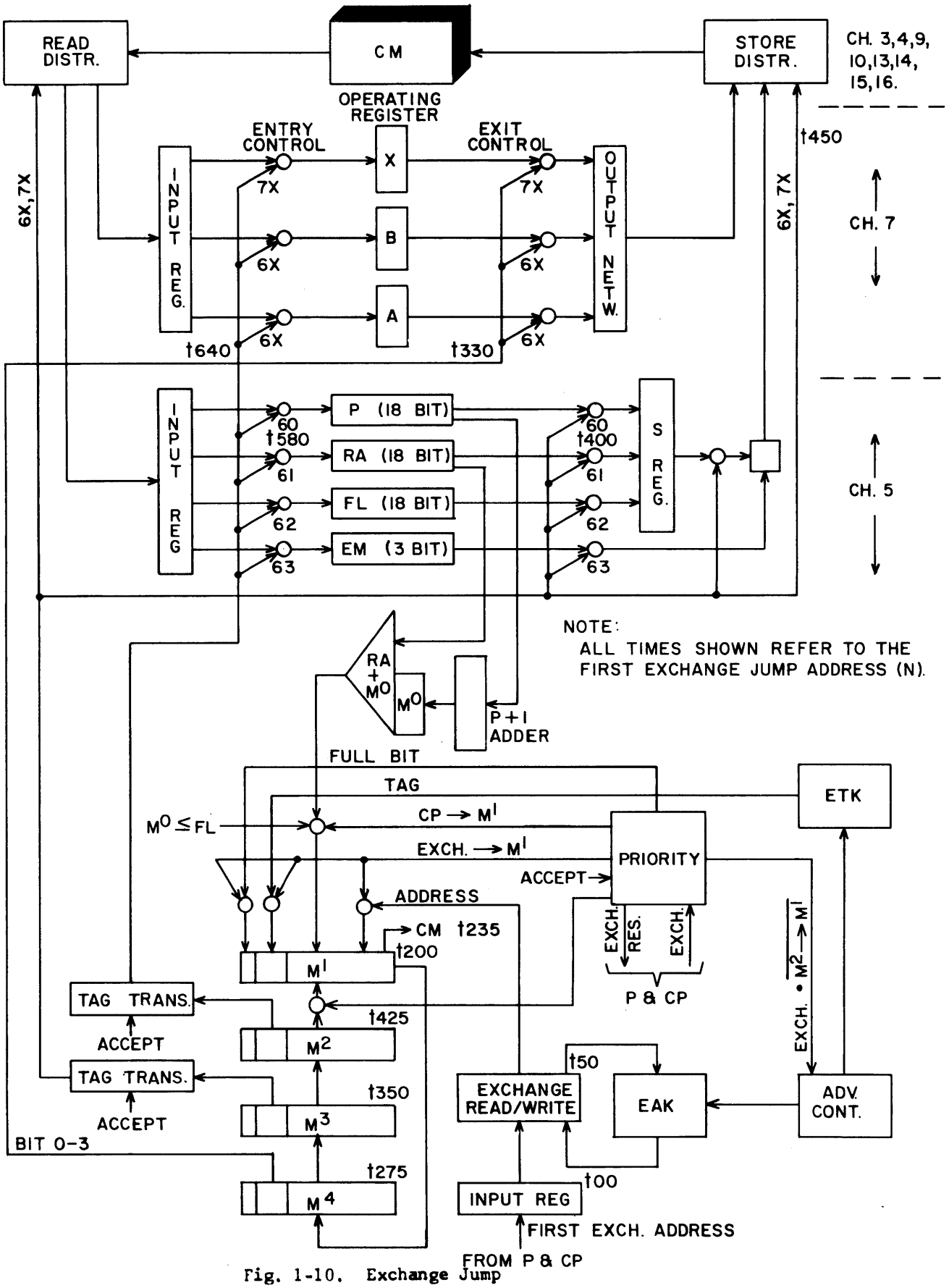


Fig. 1-10. Exchange Jump

This completes the process of sending the first address of the Exchange Jump Package from the Stunt Box to central memory. Since the Exchange Jump Package occupies 16 sequential addresses in central memory, the next 15 addresses must be generated in the central processor. The Exchange Address Counter increments the first address and subsequent addresses to provide these fifteen sequential addresses. From the Exchange Address Counter, they will be sent to central memory at a maximum rate of one every 100 nanoseconds.

The Exchange Address Counter is advanced as follows:

> At t160, Q19/TP6 was set and remains set throughout the Exchange Jump. The "1" from Q19/P26 provides a "1" from "K" on R14 (t200) which advances the counter. The B gates on Q27, 28, 29 were enabled at t115 and also remain enabled throughout the Exchange Jump. At t250, the new address (N+1) enters the Exchange/Read/Write register.

At the same time, the Exchange Tag Counter increments the tag by one and sends it to M^1 along with the new address. When the Exchange Tag Counter reaches 76_8, a "0" into 040/P6 sends an Exchange Resume (t1630) back to the Peripheral & Control Processor and also clears FF TP1 and FF TP2 (t1630). This drops the advance pulses for the Exchange Address and Exchange Tag Counters. However, the counters still have time to reach $N+17_8$ and 17_8 respectively, completing the Exchange Jump in the Stunt Box.

Operation

The previous section described the controls for an Exchange Jump. This section attempts to explain the sequential operation of the exchange jump in the central processor.

The purpose of an exchange jump is to exchange the controls for one program with new controls for a second program. In the 6600 central processor the controls for any program presently running will be found in the 24 operating registers A, X, B and in the control registers P (program address), RA (reference address), FL (field length) and EM (exit mode). Before the exchange jump was executed, values for the above listed registers were loaded into an area of central memory and called the exchange jump package. The starting point of this area in memory was defined by the address N when we executed the exchange jump.

When the first address of the exchange jump entered the hopper, a tag was added. This tag, when translated, provides the gates necessary to exchange the P register and the A_0 register with the contents of address N of the exchange jump package.

P $\rightarrow$ Address N

A tag translation of 60 and an accept signal enables a "0" from P40/P1 at t400, which goes to the S register to enable the contents of the P register into the S register. Then at t465, (P) are sent to the Store Distributor. From M^3, the 6-bit tag is extracted (t380) and transferred to the tag translator, P40. Together with an accept it will provide a "0" out of P40/P20 (6X and accept) which is fed through 041/P26 to K38/P20. At t450 a Central Register

Write signal is sent to the Store Distributor allowing the P register to be written into address N, bit positions 36-53 of central memory.

Address N $\rightarrow$ P

During the same memory cycle, we read P (the contents of P stored in the Exchange Jump Package) from address N and send it via the Read Distributor to the Input Register (B42, A41, 42, C41 on Ch. 5). A 60 tag from P39/P1 and an accept enables the contents of the Input Register into the P register at t580.

A_0 $\rightarrow$ Address N

The lower 4 bits of the 60 tag are extracted from M^4 and sent to G02 (Ch. 8) and H26 (Ch. 7) of the Exit Control. The 2^3 bit (the "GO" bit) is a "0" which indicates that an A or B register can be transferred into memory. The lower 3 bits are also "0"'s, indicating the register number is 0. As the B_0 register is not connected to memory, we can only send the A_0 register to memory. The "1" from H26/P3 (Ch. 7) is AND'ed with t30 on E31 to give a "0" output from E31/P18. This "0" goes into F32, 33, 34, 35/P10 and E26, 27/P10 allowing the 18 bits of the A_0 register to be sent to address N, bit positions 18-35, of central memory.

Address N $\rightarrow$ A_0

From R29 thru P18 and 033 (Ch. 5) a GO bit is sent to Chassis 7, G37/P15. This GO bit passes thru C36,A35 and enables the C gates in the Input Register B40, 41, 42 etc. on Chassis 7. The output of P39/P27 on Ch. 5 (a "0") is fed through H42/P10 into Chassis 7, H27/P11, setting the GO FF. This provides a "0" output from H27/12 thru C20, E28 and into C27/P16. This GO bit enables the transfer of the Input register thru C27, 28 etc. into the A_0 register G15, 16 etc.

This concludes the exchange for P and A_0 of address N. As the second address N+1 is sent to memory, the tags will be set to 61, enabling the exchange of RA, A1 and B1. When the Exchange Tag Counter reaches 76_8 the Enter Central FF on P37/TP3 is set. This forces a 10 tag (RNI) and a full bit into M^1.

After all the 16 addresses have exchanged their contents, the new program address in P will be sent to M^1 together with the 10 tag and full bit to start the execution of the new program.

Peripheral Read/Write

When a peripheral processor requests access to central memory for a read or write, the peripheral processor sends the appropriate signal to the stunt box. The signal is received at time 00 on chassis 5, module 041. The peripheral processor also sends the central memory address from its "A" register. The address is received in the input register in chassis 5 at time 00 on modules N40, N41, N42. The appropriate read or write signal received at 041 generates a 50 nanosecond one-shot pulse on pins 2, 4, & 8, to gate the address from the input register to the Exchange/Read/Write register located on modules Q15, Q16, Q17. (Refer to fig. 1-11.)

At this time, the controls on 041 attempt to gain priority to enter the hopper. (signal on pin 10) Since priority to enter the hopper may not necessarily be available, the address in the exchange register is retained until it can be transferred to the hopper. To accomplish this the address circulates through the exchange address counter located on modules R14 thru R19. The address is not advanced, however, since the advance pulse is only enabled during an exchange jump. (Refer to fig. 1-12.)

When the peripheral address is attempting to gain priority in the stunt box, the"Central Busy"flip-flop in the peripheral processor is set. This prevents any other peripheral processor from attempting a central memory reference. The "Central Busy" flip-flop will only be cleared after the latest address has been accepted by central memory.

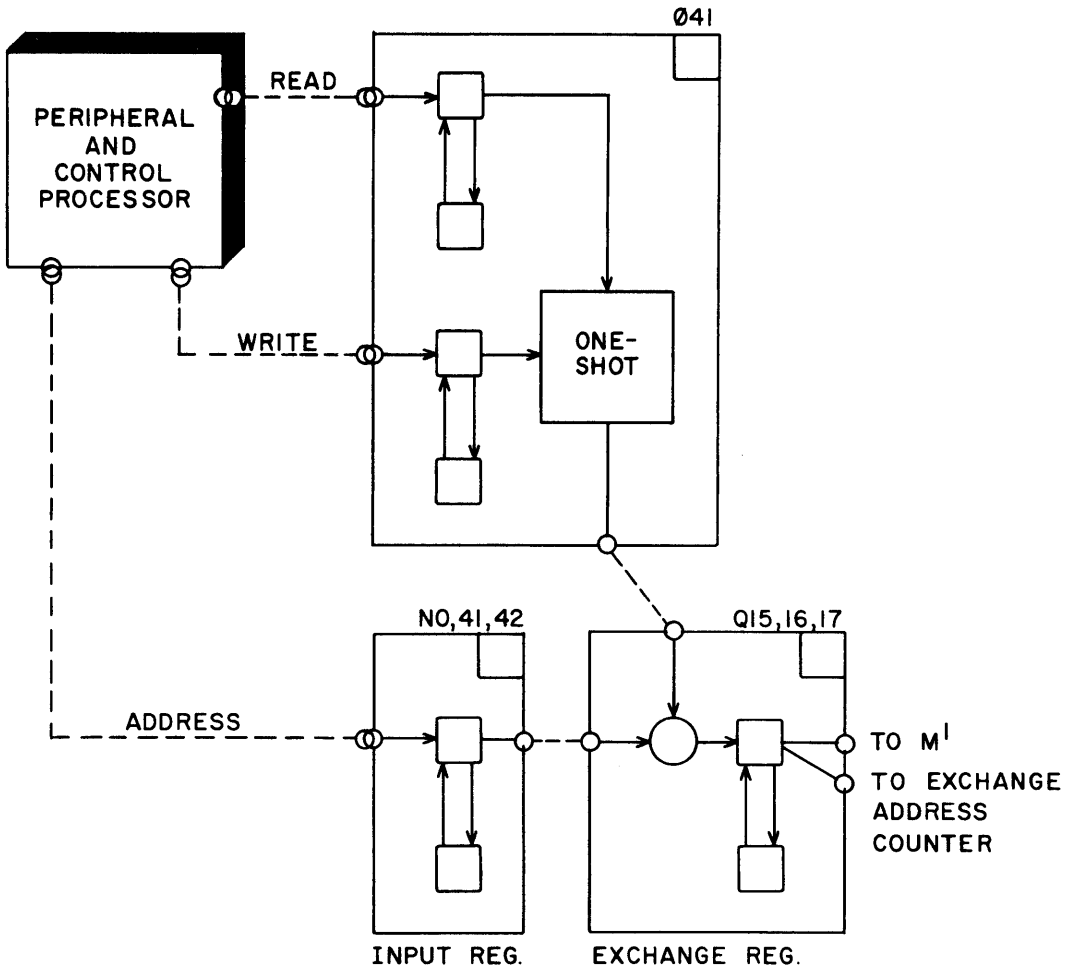

041
PERIPHERAL AND CONTROL PROCESSOR
READ
WRITE
ONE-SHOT
N0,41,42
Q15,16,17
ADDRESS
TO M'
TO EXCHANGE ADDRESS COUNTER
INPUT REG.
EXCHANGE REG.

Fig. 1-11.

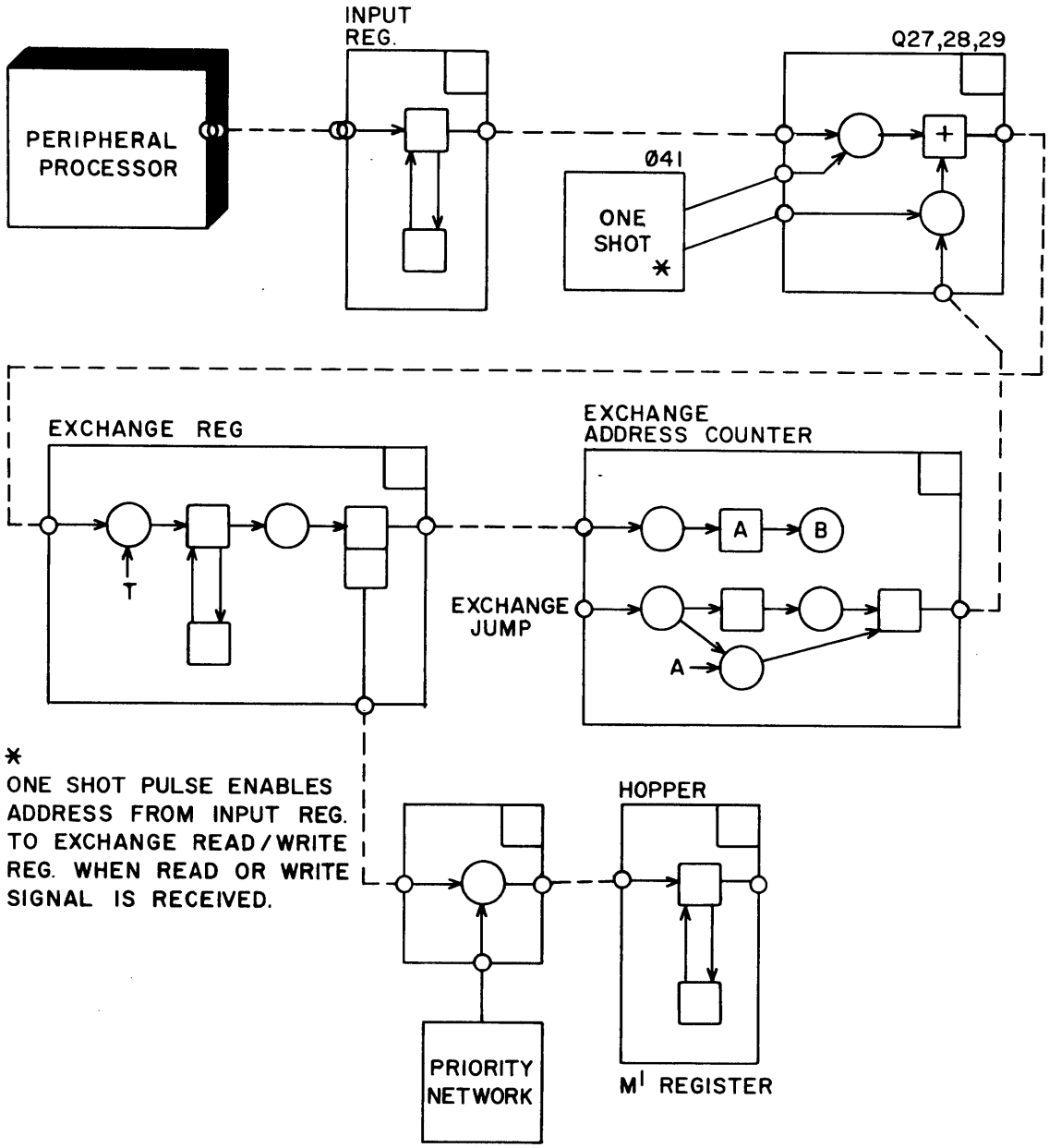

INPUT
REG.
Q27,28,29
PERIPHERAL
PROCESSOR
Ø41
ONE
SHOT
*
EXCHANGE REG
EXCHANGE
ADDRESS COUNTER
A
B
T
EXCHANGE
JUMP
A
*
ONE SHOT PULSE ENABLES
ADDRESS FROM INPUT REG.
TO EXCHANGE READ/WRITE
REG. WHEN READ OR WRITE
SIGNAL IS RECEIVED.
HOPPER
PRIORITY
NETWORK
M' REGISTER

Fig. 1-12.

To avoid the situation where the central processor ties up the hopper such that a peripheral processor request for memory cannot be honored, the circuitry stops the central processor, honors the peripheral request, and then restarts the central processor. Conditions for stopping the central processor are:

1. Address in M^2 not accepted.
2. M^3 full
3. M^4 full
4. Peripheral read or write request.

Full Bits and Tags

When priority for the peripheral read or write address has been established, the priority network enables the gating of the address into the hopper (gates on Q25, Q26, Q27). At this time a tag and Full bit will be appended to the address.

Full Bit

Appending the Full bit is accomplished by setting flip-flop R33/TP1 with the address to M^1 enable. (Refer to fig. 1-14.)

Tag

The tag for a peripheral read is 00; the tag for a peripheral write is 40. One of these two tags must be appended to the address in the hopper when a peripheral read/write is attempted. When priority has been granted for gating the peripheral address to M^1, the "enable peripheral to M^1" term on Q33/TP1 makes the "C" and "F" gates for the fan-in on Q25. This fan-in permits entry of the proper tag into M^1. In a peripheral read, all the inputs to the fan-in will be zero.

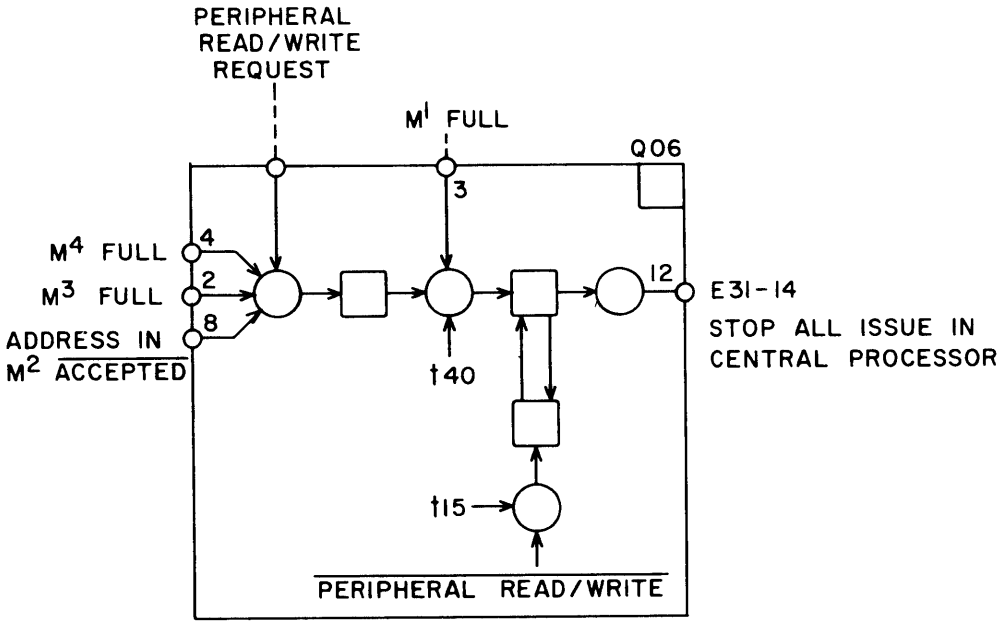

PERIPHERAL
READ/WRITE
REQUEST
M1 FULL
Q06
3
M4 FULL
4
M3 FULL
2
ADDRESS IN
M2 ACCEPTED
8
12
E31-14
STOP ALL ISSUE IN
CENTRAL PROCESSOR
↑40
↑15
PERIPHERAL READ/WRITE

Fig. 1-13.

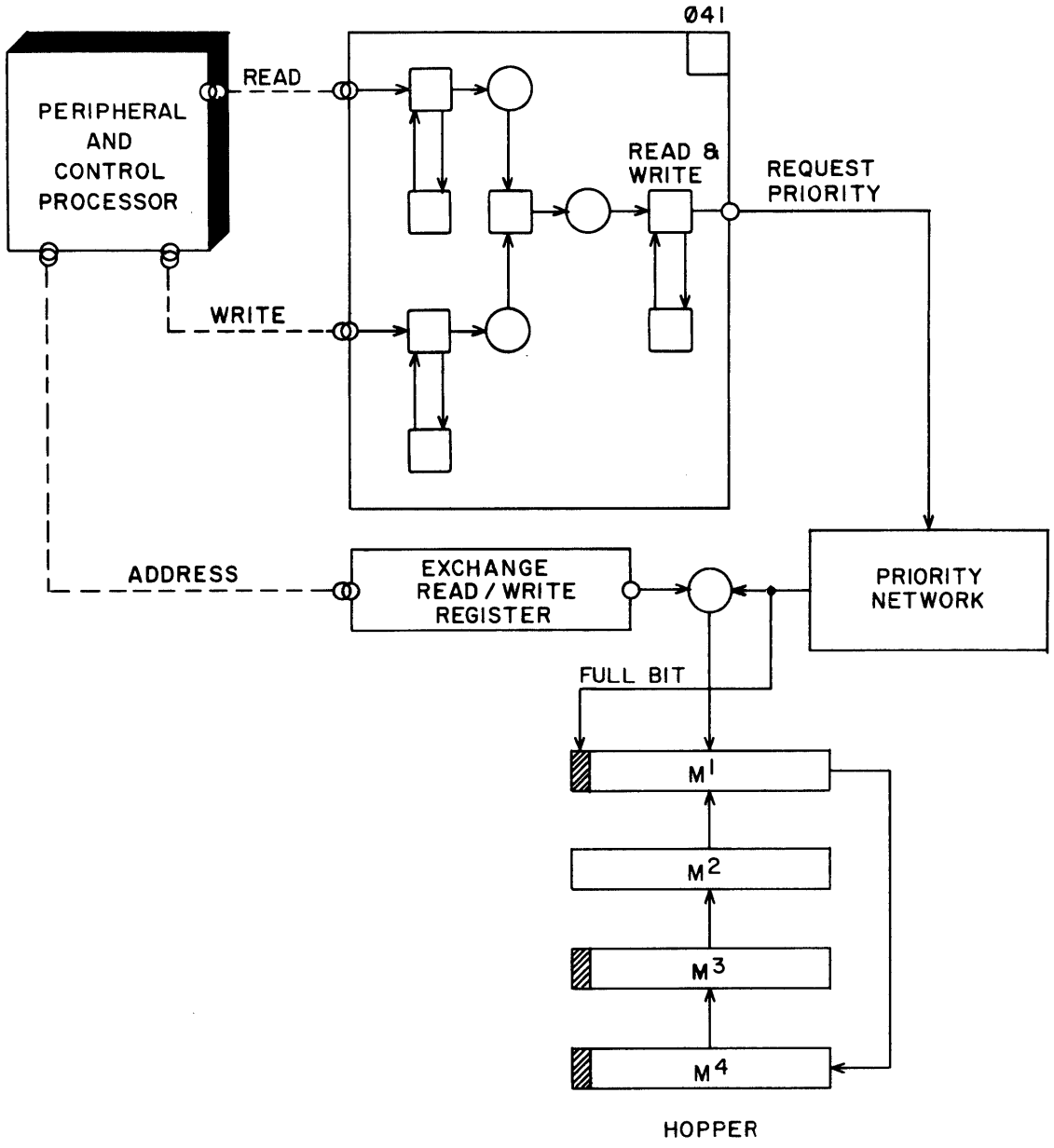

041
PERIPHERAL AND CONTROL PROCESSOR
READ
WRITE
READ & WRITE
REQUEST PRIORITY
ADDRESS
EXCHANGE READ / WRITE REGISTER
PRIORITY NETWORK
FULL BIT
M1
M2
M3
M4
HOPPER

Fig. 1-14

In a peripheral write, the write bit (2^5) will be a "1" since the R33/TP2 FF is set. Once the peripheral address is in the hopper, the address will be issued to central memory until it can be accepted. When the address has been accepted, the tag is translated from the hopper and sent to memory control to allow memory to send the data and resume signals to the peripheral processor.

Central Read/Write

When the central processor attempts a central memory read or write operation, the address originates from one of two basic sources:

1) from the Program Address, P, register when the memory request is for an instruction word, and
2) from one of the Increment functional units when the memory request is for an operand.

As diagrammed in fig. 1-13, either the Increment Address FF or the Program Address FF sets the Enter Central FF. Setting the Enter Central FF initiates action to request priority for the address to enter the hopper.

The address, in either case, is gated to a common register (M^0). The contents of M^0 are added to the reference address (RA) and this address is gated to hopper register M^1 when priority has been granted and $M_0 \leq FL$.

Program Address

The central processor Program Address register, P, and its incrementor network always hold the address of the last instruction word read from memory. When the next instruttion word is needed, an "Inch" signal sets the Program Address FF. This gates the contents of the P register + 1 to the M^0 register, providing M^0 is not holding an address that has not yet entered the hopper. In addition, if there are addresses comming to M^0 simultaneously (i.e., from P and from the Increment units), the Increment address has priority over P+1 for entry into M^0.

Increment Address

When an instruction that causes a change in the A registers one through seven (A1 - A7) is executed in the Increment units, the address developed is sent to the

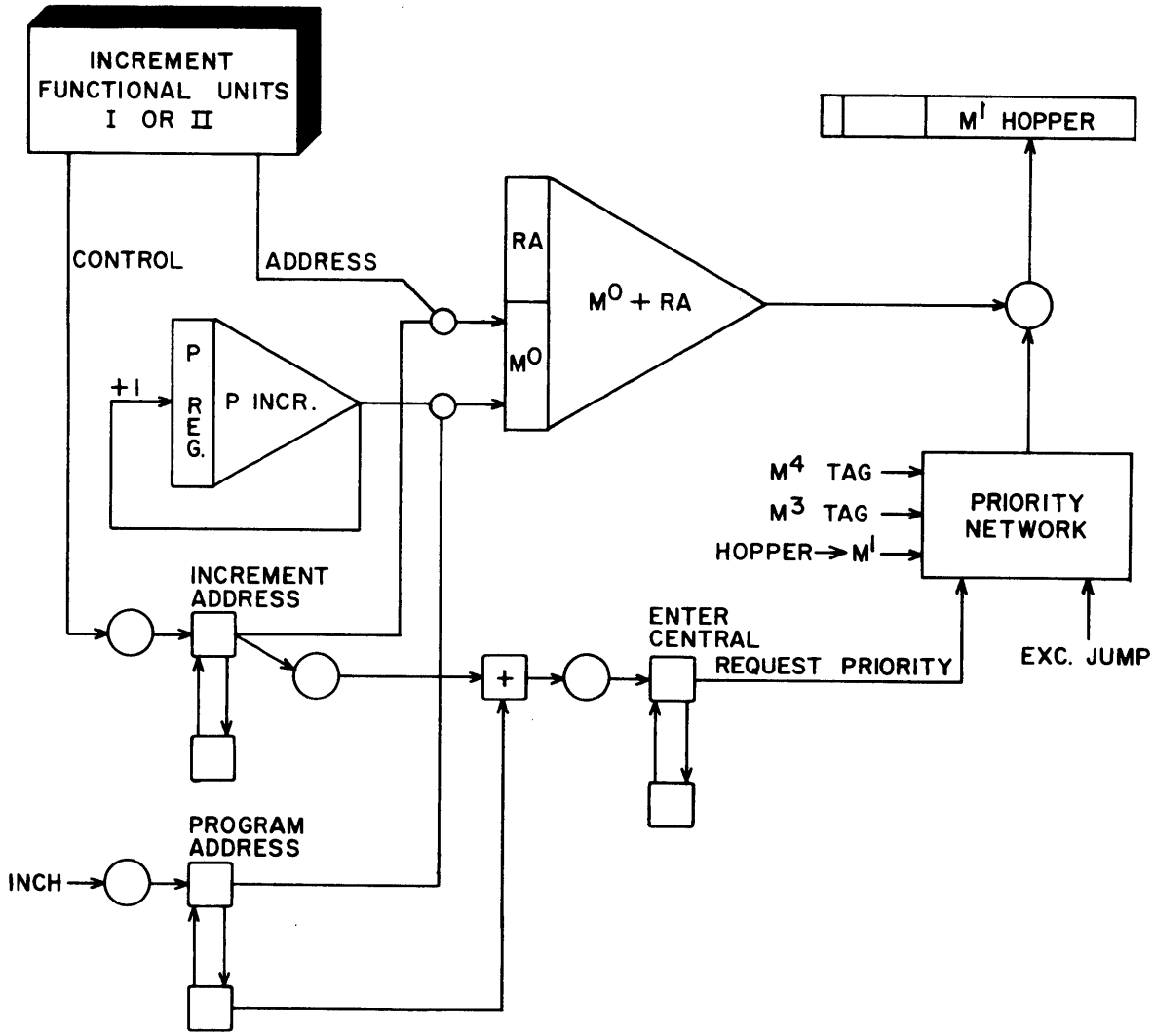
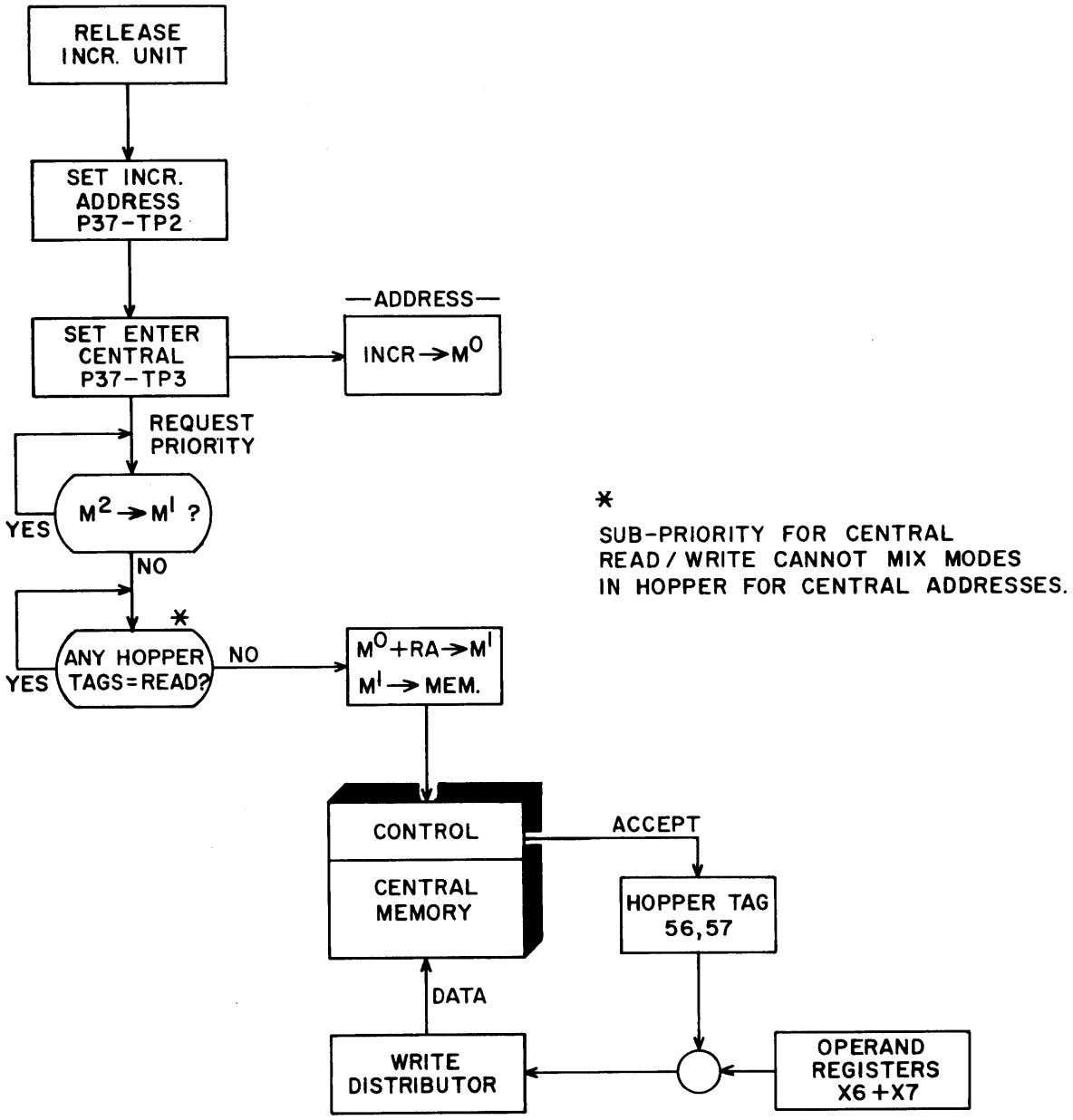


Fig. 1-15. Central Read/Write

Stunt Box to make the necessary memory reference for loading or storing operating registers X^1 - X^7.

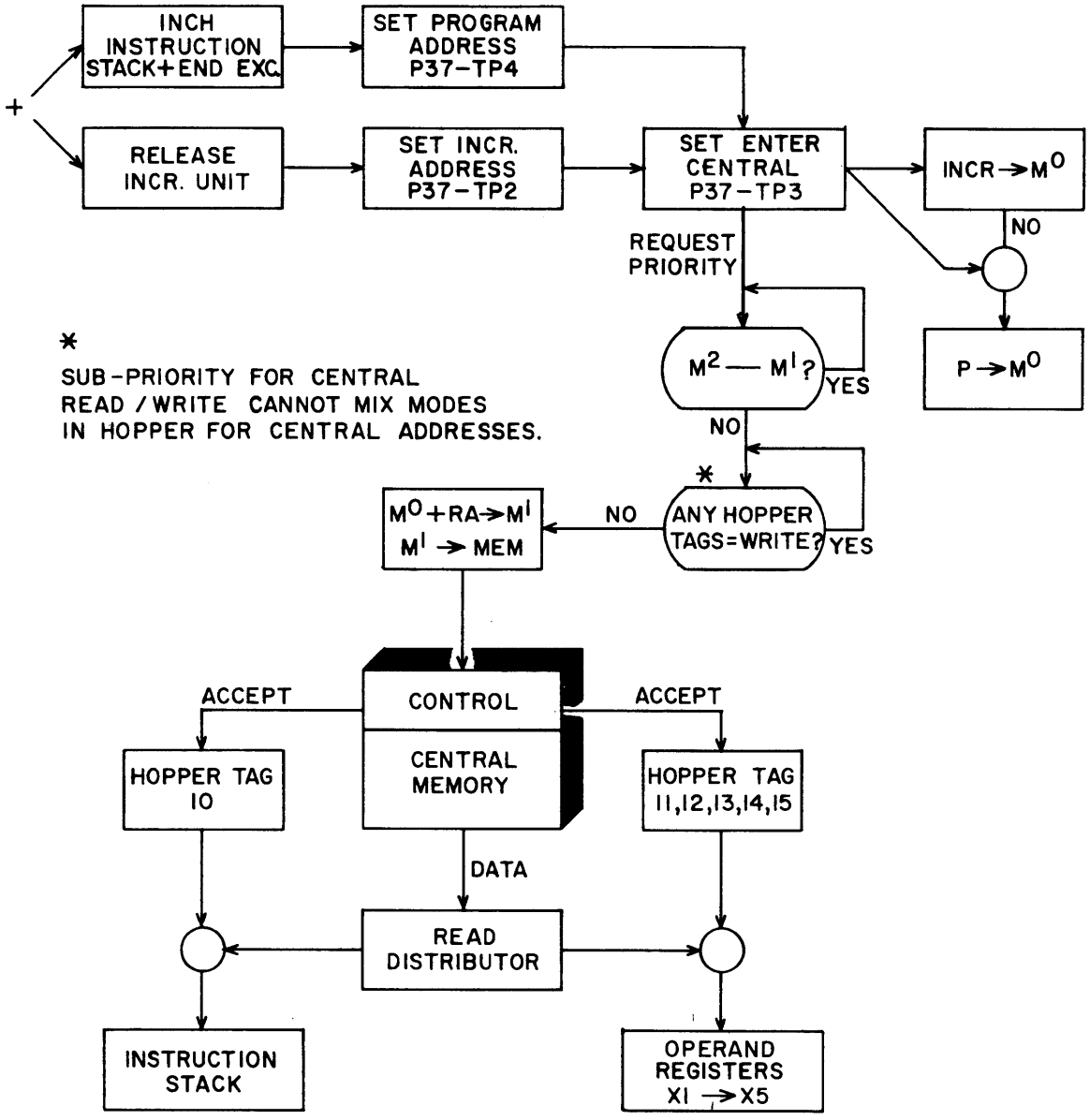
An increment instruction (51 - 57), when nearing completion, sets the Increment Address FF. Subsequent events are as outlined above.

Figs. 1-16 and 1-17 diagram the sequence of events for central memory read and write operations.



CENTRAL WRITE

Fig. 1-16



CENTRAL READ

Fig. 1-17.

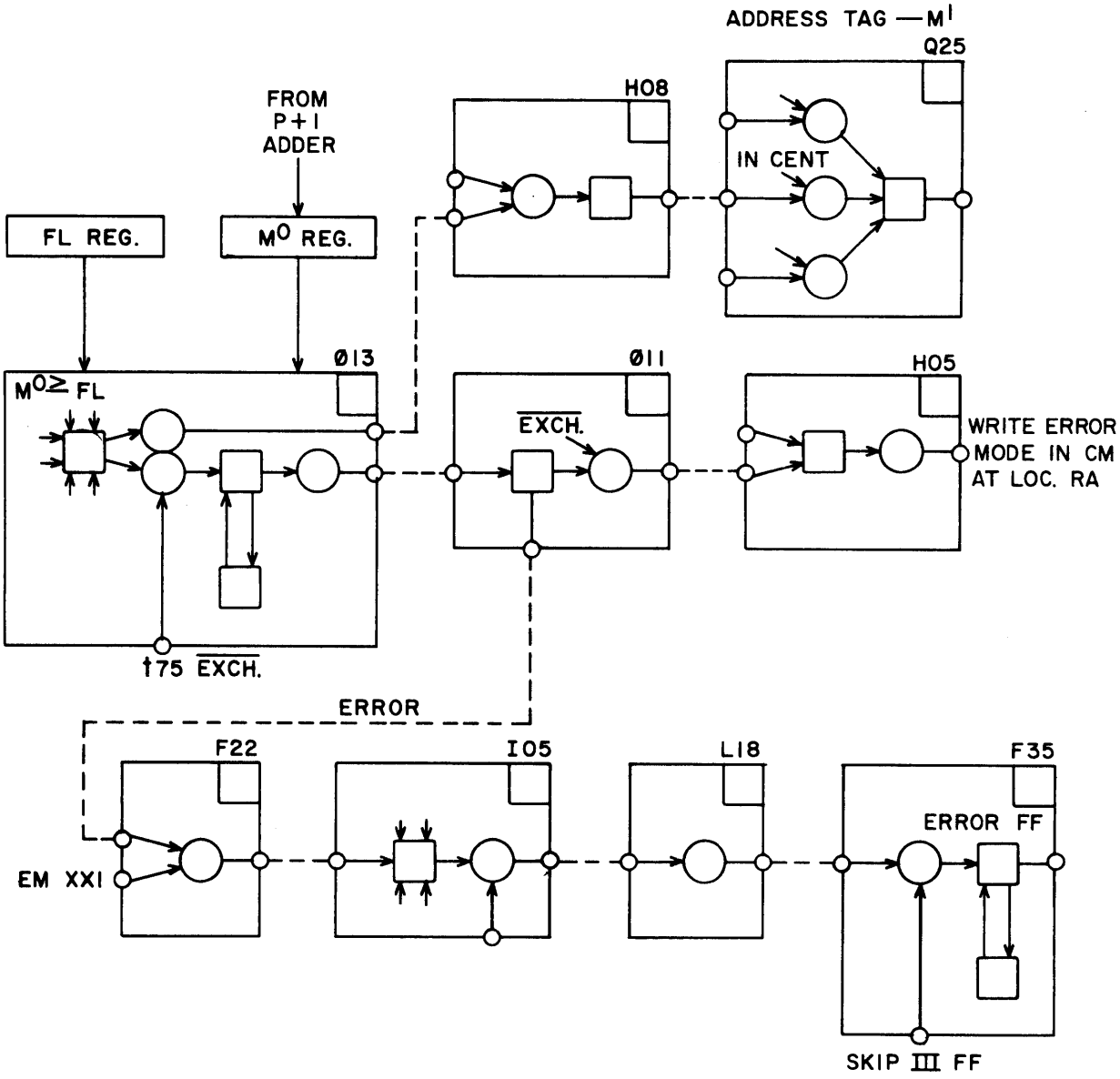

ADDRESS TAG — M1
Q25
H08
FROM
P+1
ADDER
IN CENT
FL REG.
M0 REG.
Ø13
Ø11
H05
M0 ≥ FL
EXCH.
WRITE ERROR
MODE IN CM
AT LOC. RA
↑75 EXCH.
ERROR
F22
I05
L18
F35
ERROR FF
EM XXI
SKIP III FF

Fig. 1-18.

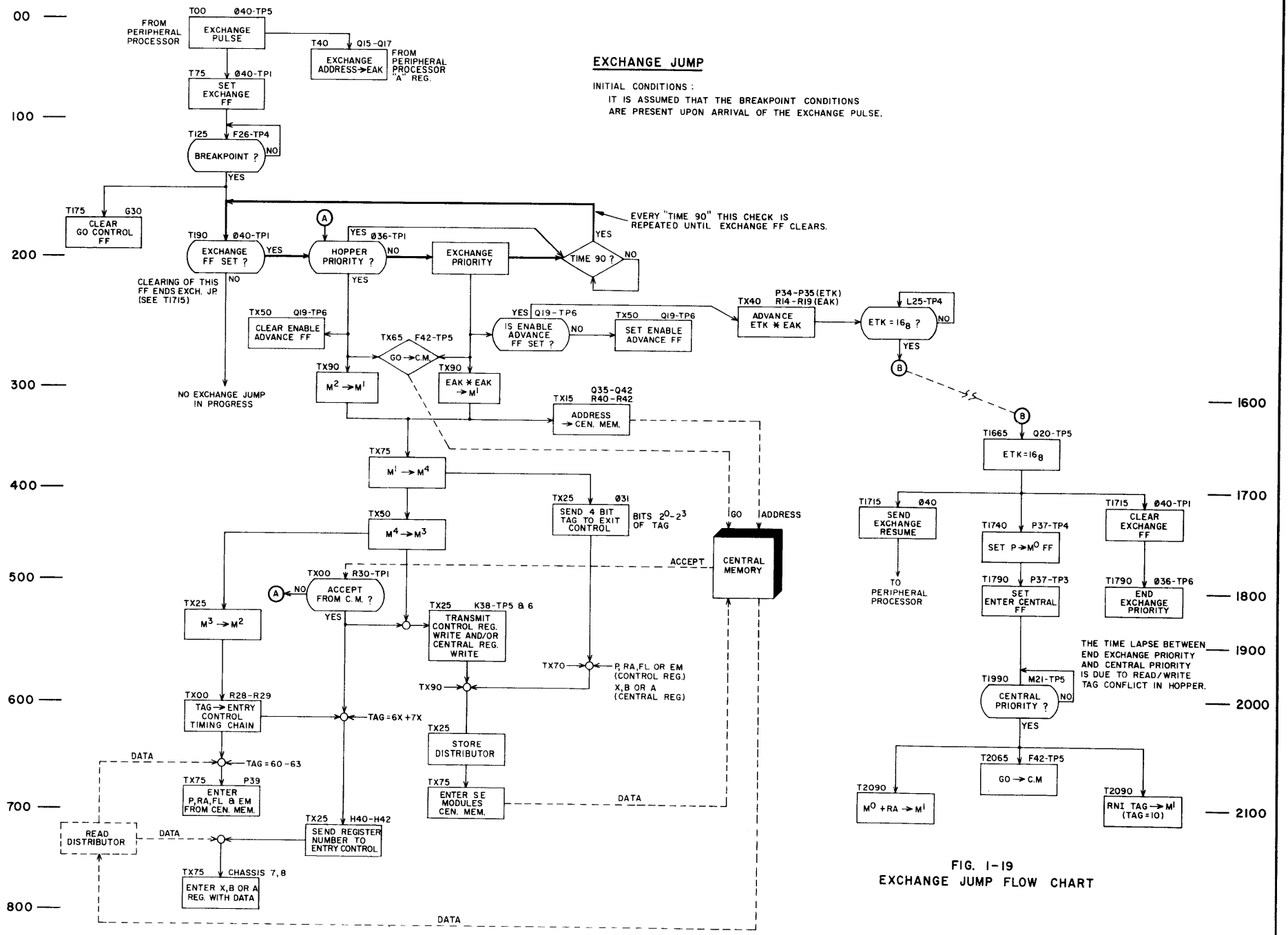


FIG. 1-19
EXCHANGE JUMP FLOW CHART

EXIT MODE

The exit mode feature allows the programmer to choose the exit or stop condition of the central processor. Exit selections are stored in the EM-register, and the exit mode occurs as soon as it is sensed. The various exit conditions are:

1. Normal stop
2. Address out of bounds ($M^{\circ} \geq FL$)
3. Operand out of range (infinite)
4. Indefinite operand

The address RA (and hence P = 0) is reserved for recording program exit conditions.

1. Normal Stop

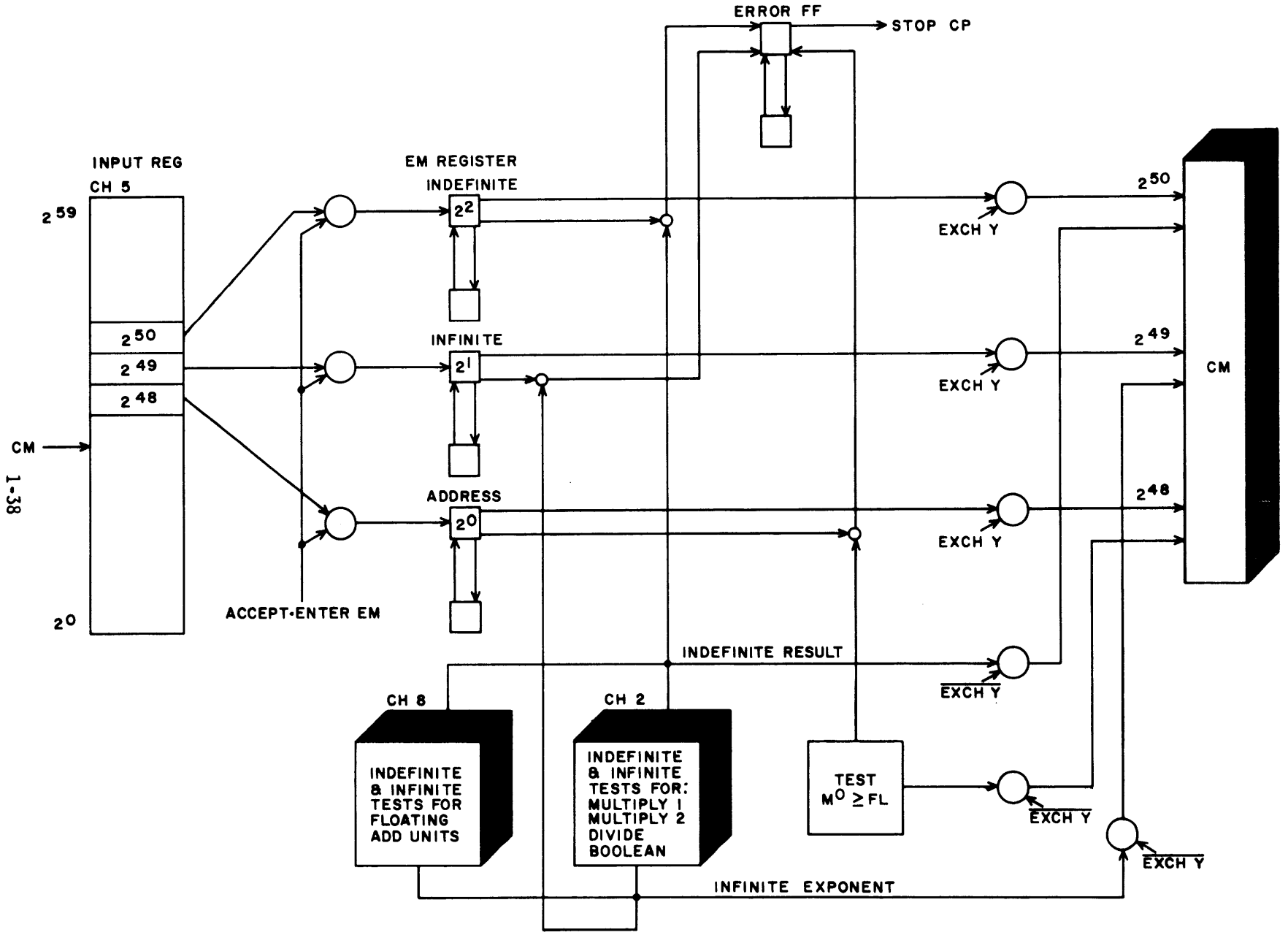
The translation of a normal stop instruction (fm = 00) in the U^2 translator together with a SCBD ISSUE will set the STOP FF(G30 TP1). The P register will be cleared, so that the contents of the P register is 0. The peripheral and control processor searches for a central processor P = 0 conditions (instruction 27) to determine that the CP has stopped.

2. Address out of bounds

$M^{\circ} \geq FL$ test

The contents of the M° register is continously compared with the contents of the FL register as follows:

a. A bit by bit comparison of bits $2^3 \rightarrow 2^{17}$ ($M^{\circ} \geq FL$) and three results of these comparisons are comprised in one group (anded together). If one bit of M° is less than the corresponding bit of FL the group output is "0".


ERROR FF
STOP CP
INPUT REG
CH 5
2^{59}
2^{50}
2^{49}
2^{48}
2^{0}
CM
EM REGISTER
INDEFINITE
2^{2}
INFINITE
2^{1}
ADDRESS
2^{0}
ACCEPT-ENTER EM
EXCH Y
2^{50}
2^{49}
2^{48}
CM
INDEFINITE RESULT
CH 8
INDEFINITE & INFINITE TESTS FOR FLOATING ADD UNITS
CH 2
INDEFINITE & INFINITE TESTS FOR: MULTIPLY 1 MULTIPLY 2 DIVIDE BOOLEAN
TEST $M^{0} \geq FL$
INFINITE EXPONENT

EXIT MODES

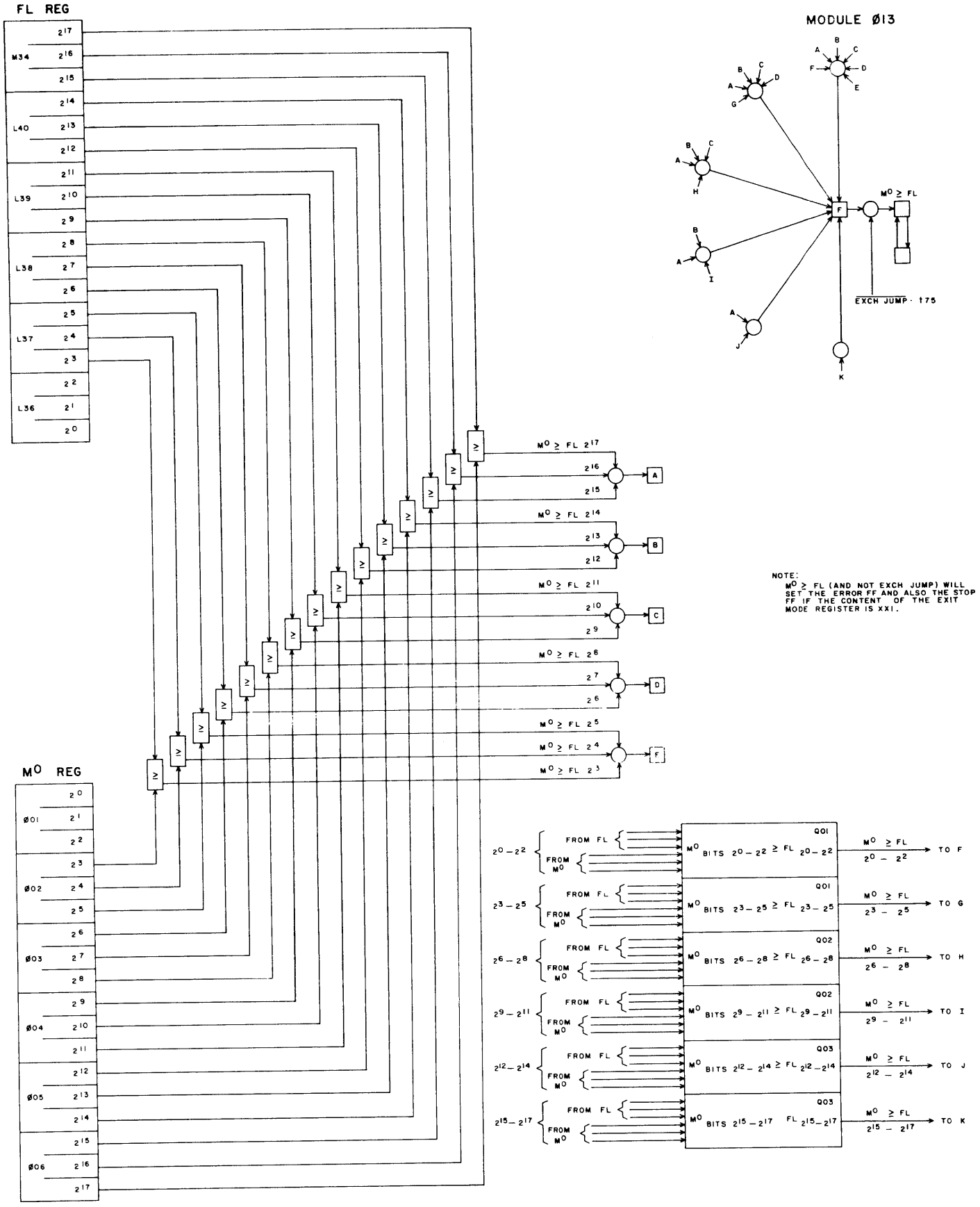


FIG. 1-21
M0 – FL COMPARISON

b. Each of the 6 octal digits of M° is compared with the corresponding octal digit of FL. If $M^\circ \geq FL$ the output will be a "1".

The output of (a) and (b) are anded together to decide if $M^\circ \geq FL$. If $M^\circ \geq FL$ and we are not in an Exchange Jump the $M^\circ \geq FL$ FF (013 TP5) will be set. Supposing the contents of the EM register is XX1 and the Skip III FF is set, the $M^\circ \geq FL$ will set the ERROR FF.

The output of the comparison network will also inhibit sending the write bit of the hopper tag to the hopper and does not allow the value of $M^\circ + RA$ to be sent to M^1 if $M^\circ \geq FL$.

Address out of bounds, exit mode not selected (EM) = XX0

When the central processor attempts to reference a memory location out of bounds, an out of bounds exit mode is not selected. If out of bounds is selected the central processor will stop at the Reference Address.

RNI OUT OF BOUNDS

If the program counter (P register) attempts to send an address that is out of bounds (this would be the RNI situation), the $M^\circ \geq FL$ check circuit will inhibit sending the intended address to the hopper. Instead an address of all zeros and the proper RNI tag will be sent to the hopper. This will result in the reading of memory address absolute zero as well as the data going to the instruction stack and the U register translating networks. If absolute address zero had contained an all zero data word the results would be to stop the central processor, because of the translation of a stop (00) at absolute address zero.

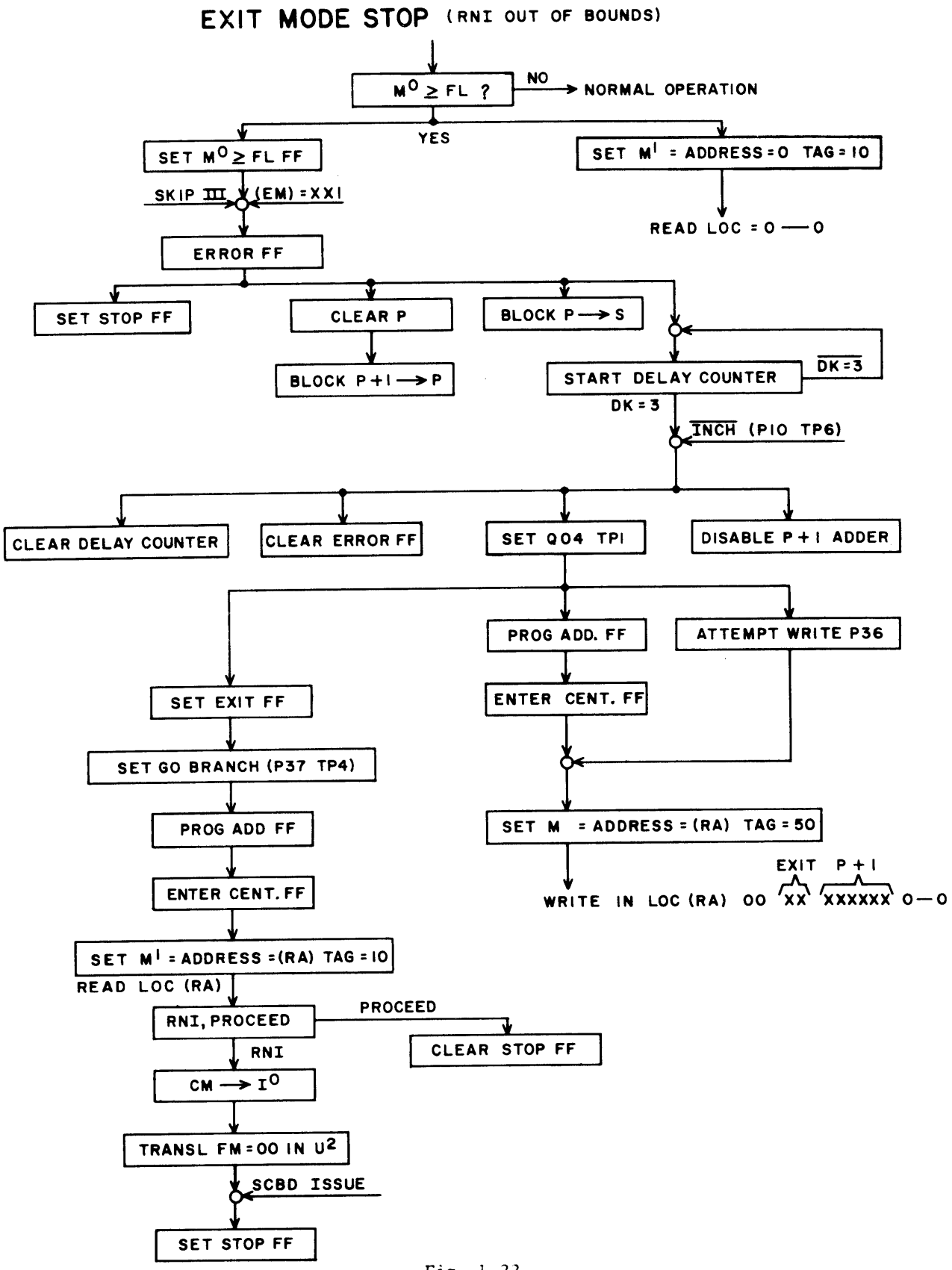

EXIT MODE STOP (RNI OUT OF BOUNDS)
M0 ≥ FL ?
NO
NORMAL OPERATION
YES
SET M0 ≥ FL FF
SET M1 = ADDRESS = 0 TAG = 10
SKIP III
(EM) = XX1
READ LOC = 0 — 0
ERROR FF
SET STOP FF
CLEAR P
BLOCK P → S
BLOCK P+1 → P
START DELAY COUNTER
DK = 3
DK = 3
INCH (P10 TP6)
CLEAR DELAY COUNTER
CLEAR ERROR FF
SET Q04 TP1
DISABLE P+1 ADDER
PROG ADD. FF
ATTEMPT WRITE P36
SET EXIT FF
ENTER CENT. FF
SET GO BRANCH (P37 TP4)
PROG ADD FF
SET M = ADDRESS = (RA) TAG = 50
EXIT P+1
WRITE IN LOC (RA) 00 XX XXXXXX 0 — 0
ENTER CENT. FF
SET M1 = ADDRESS = (RA) TAG = 10
READ LOC (RA)
RNI, PROCEED
PROCEED
CLEAR STOP FF
RNI
CM → I0
TRANSL FM = 00 IN U2
SCBD ISSUE
SET STOP FF

Fig. 1-22

READ OPERAND OUT OF BOUNDS

If the increment unit changes A1-A5 (read memory to X) with an address that is out of bounds, again the M° ≥ FL check circuits will inhibit the sending of the address to the hopper. Instead an all zeros address will be sent to the hopper with a 10 tag. This will result in referencing memory location absolute zero and the contents of this location will go to the indicated X register. No affect on the running program will be seen except that the contents of the X registers will not be what is expected.

WRITE OPERAND OUT OF BOUNDS

If the increment unit changes A6+A7 (store X6+X7) with an address that is out of bounds, the M° ≥ FL check circuit will inhibit the sending of the out of bounds address to the hopper. Instead an address of all zeros will be sent to the hopper. Usually the tag for a store X6+X7 is a 56+57, but if the address is out of bounds the tag will be changed to a 16+17, this tag will inhibit sending the central register write tag to the store distributor. The results of this zero address and 16+17 tag in the hopper is the referencing of address absolute zero. When the data gets to the data distributor however, the lack of any tag will cause the data to be lost. There will be no effect on the running program, nor on the X6+X7 registers. The X6+X7 register would not be stored.

Address out of bounds, exit mode selected (EM)=XX1

(Refer to timing diagram of Exit Mode Stop)

If the M° ≥ FL comparison network detects an address out of bounds it will inhibit the sending of the intended address to the hopper and will also set the M° ≥ FL FF. Instead an address of all zeros and a 10 tag will be sent to

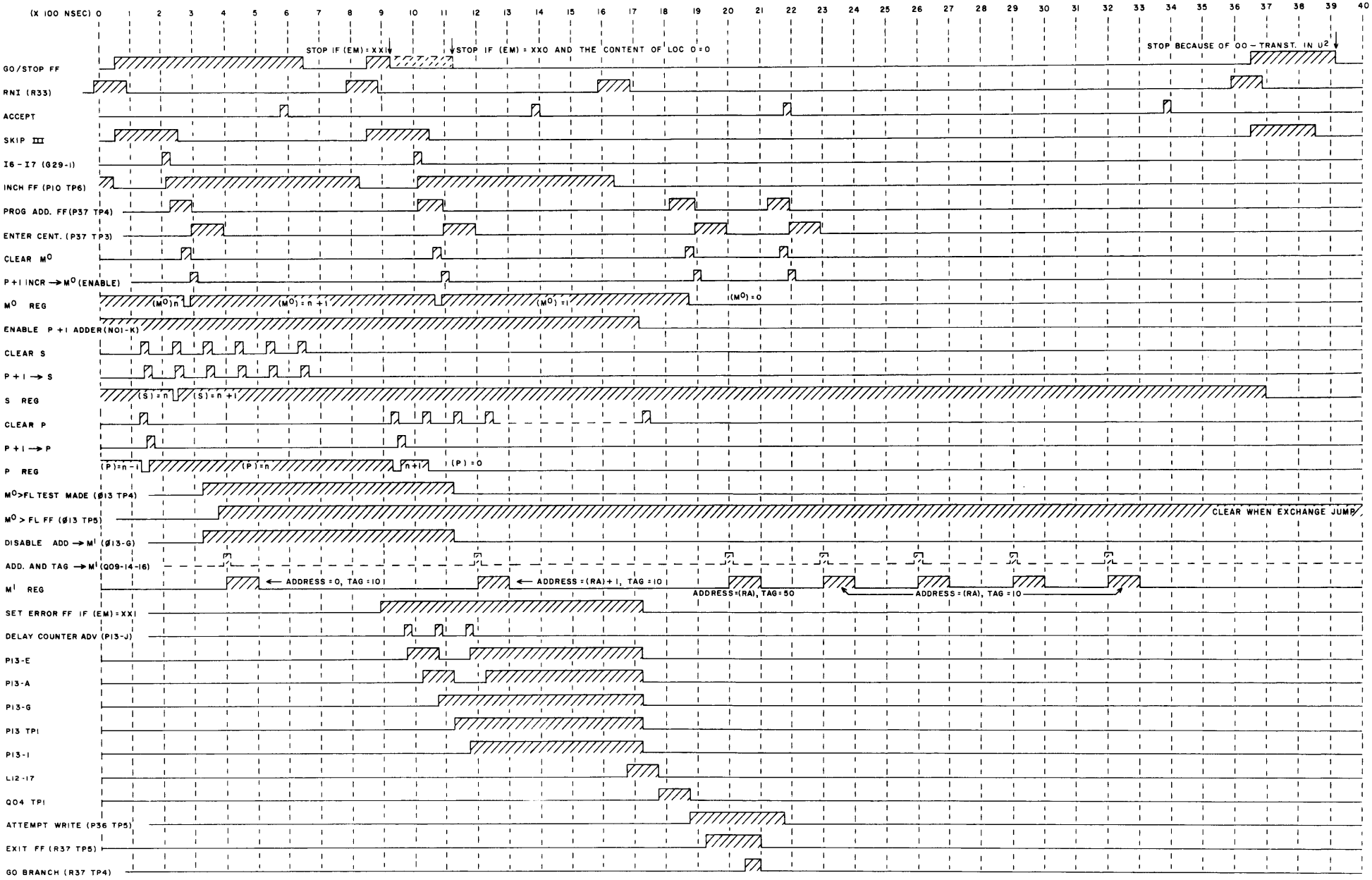
the hopper. A 10 tag in the hopper tag timing chain and an ACCEPT signal will set the RNI FF and a PROCEED signal is sent to the ISSUE CONTROL. As soon as the SKIP III FF is set the ERROR FF is also set. This in turn will start the DELAY COUNTER (P13) and also set the STOP FF. After 200 nsecs. the output of the counter is 1, but the INCH FF (P10 TP6) must be cleared in order that the Delay counter can clear the FF at 004-TP1. This FF will enable the setting of the PROGRAM ADDRESS FF and with a P → M° signal will also set the ATTEMPT WRITE FF. Now the address RA and a 50 tag will be sent to the hopper (M) and the following data will be written into CM at location RA:

Stop	Exit	(P)+1
00	XX	XXXXXX 0 ⟶ 0

At the same time the ATTEMPT WRITE FF and FF at P10 TP1 was set. The latter will set the EXIT FF and this will set the GO BRANCH FF allowing a jump to P = 0 and hence RA. 300 ns after the address RA and a 50 tag was sent to the hopper the same address with a 10 tag will be sent to M^1. When the address is accepted by CM the contents of address RA (now 00 XX XXXXXX 0000000000) will be transferred to the Instruction Stack and to the U registers (because of the RNI and Proceed signals). The translation of fm = 0X in the U^2 translator and a SCBD Issue will set the STOP FF again.

3. Infinite or indefinite operands

The functional units using floating point arithmetic will always test the exponents for infinity (greater than 3777) and for indefinite results (see Appendix D). If any of the above conditions exists, a bit is sent to Chassis 5, anded together with the resp. FF's of the EXIT MODE register and in turn sets the ERROR FF if the particular EM FF was set. As soon as the ERROR FF is set the timing diagram of Exit Mode Stop (RNI out of bounds) can be used.



NOTE:
n = CONTENTS OF P REG AT TIME OF ERROR (P ≥ FL)

FIG. 1-23 EXIT MODE STOP (RNI OUT OF BOUNDS)

(WRITE IN LOC RA : 00 EXIT P+1 XX XXXXXX 0 —— 0)

SECTION II

Central Processor Instruction Issue Control

INSTRUCTION ISSUE CONTROL

INSTRUCTION STACK

INTRODUCTION

The 6600 central processor, designed for very high speed computing, uses parallel functional units and overlapping memory references to reduce program running time. Another method of reducing time consumed during execution of a program eliminates unnecessary memory references. The following example of a portion of a typical scientific program illustrates the desirability of minimizing memory references:

EXAMPLE:

ADDRESS	OPERATION
100	Load operand A modified by B1
101	Multiply by operand C
102	Subtract operand D
103	Jump if sign of results are negative
104	Store results in X modified by B1
105	Update B1
106	Jump to address 100

The above loop example shows that memory references to obtain the same group of instructions are necessary for each pass through the loop. To avoid unnecessary memory waiting time, the central processor uses an instruction stack to hold short loops.

DESCRIPTION

The central processor instruction stack is made up of eight 60-bit instruction registers, labeled I0, I1, I2 . . . . . I7. An additional 60-bit register known as the Input Register serves as an input buffer between central memory and the instruction stack.

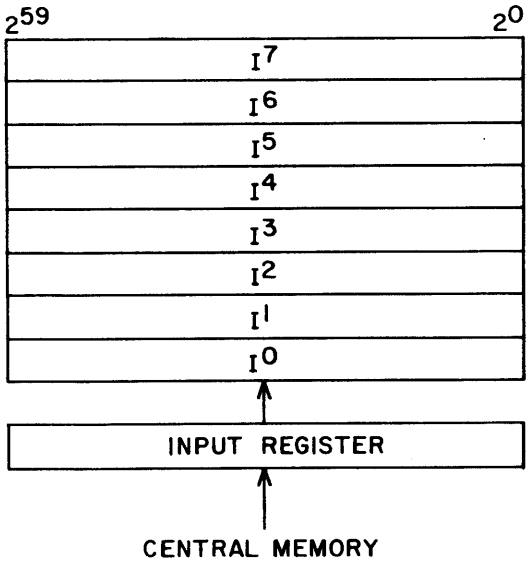


Fig. 2-1. Instruction Stack

Instruction formats for any 60-bit instruction word in the stack may be 15-bit and 30-bit instructions in any of the combinations diagrammed in fig. 2-2.

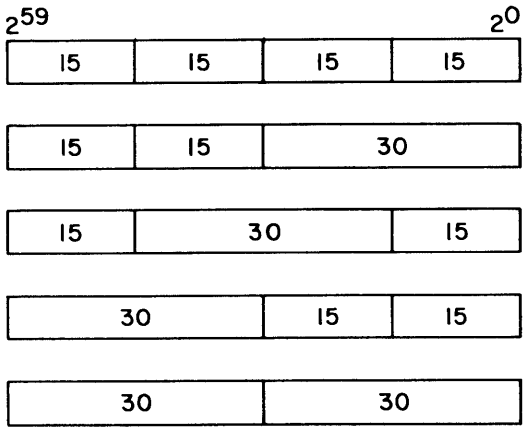


Fig. 2-2. Possible Instruction Word Formats

The maximum capacity of the instruction stack is, then, thirty-two 15-bit instructions. A program loop that can be contained within the stack can be executed as many times as necessary without making any memory references

for instruction words. Note that an iterative loop cannot be 32 instructions and remain in the stack since: a.) the jump instruction comprising part of the loop is a 30-bit instruction, and b.) a jump cannot be executed from I0 (the bottom register of the stack.)

OPERATION

The instruction stack is loaded one instruction word at a time as the instruction words are read from central memory to be executed. Each instruction is loaded into I0, sent to the translating networks, and then moved to I1. As each instruction word enters I0 and is moved upward in the stack, the complete stack contents must move upward to make room for the new instruction word. This upward movement of stack contents is called inching. As the stack contents are inched, the top instruction word (in I7) is discarded.

NOTE

Throughout the descriptive material which follows, references are made to specific FFs, logical networks, etc. To aid in understanding the text, refer to text diagrams and to 6600 Central Processor diagrams.

Inch

The process of inching the instructions in the stack is controlled by the inch counter. This counter is a 2 rank, 2 bit, binary counter initiated by the following conditions:

a) next instruction word is in I0, and

b) first instruction from I0 is being issued to the translating networks (U registers).

Once the inch counter is started, it continues counting until the entire stack has been moved up. During the inching process, the instruction word in I7 is discarded.

Parcelling

Since there are 15 and 30-bit instructions which can be combined in an instruction word in any order, the problem of extracting instructions from the ranks of the stack is encountered. The scheme used to extract instructions is called parcelling and is controlled by a 2 rank, 2 bit, binary counter called the parcel counter (PK). A 60-bit instruction word is partitioned into 4 parcels, designated 0, 1, 2, and 3. (Refer to fig. 2-3.) Each parcel is 30 bits in size to ensure

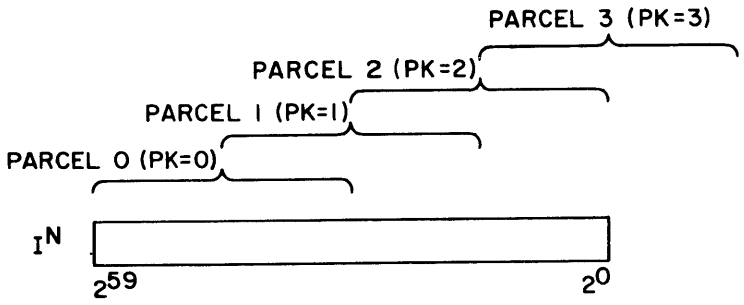


Fig. 2-3. Instruction Word Parcels

getting a complete instruction when extracting a parcel from the word. In many cases (e.g., a 15-bit instruction in a parcel), unusable information is extracted along with the meaningful. In these cases, unusable information is discarded as the instruction is placed in the U registers.

Locator

In most cases, instructions that are sent to the U registers come from the lowest rank of the stack (I0). However, an instruction word inched up into the stack can still be issued to the U registers for execution. Thus, if inching occurs before an entire instruction word is issued (parcelled) to the U registers, the next instructions will come from some stack register other than I0. A Locator network (L) keeps track of the register in the stack holding unissued instructions (i.e., the register from which the next instruction will come).

Issue Control

The movement of instructions from the stack and through the U registers is controlled by a circuit called Issue Control. There are two basic signals associated with Issue Control:

a) U1 issue (i.e., $U^1 \rightarrow U^2$)

b) scoreboard issue (i.e., $U^2 \rightarrow$ Scoreboard)

The Issue Control circuit controls the operations of the parcel and inch counters, the L network and instruction issue and movement from the instruction stack to

the scoreboard. The availability of functional units and operating registers, and information on instruction translations is monitored by Issue Control to assist this circuit in its functions.

INSTRUCTION REGISTERS

From the instruction stack, instructions flow in sequence to two registers to be translated. (Refer to fig. 2-4.)

The first register, U1, consists of 30 FFs. Entry into U1 is controlled by the L counter and by the parcel counter. Instructions which come from even stack registers are transferred via an auxiliary register, U0. (Note that this does not affect the transfer time from the stack to U1; instructions from even and odd ranks of the stack arrive in U1 at the same time (t80).

The fm portion is now translated to set the proper Select, Request and Result FFs. (Refer to fig. 2-5.)

At approximately the same time these FFs are set, the transfer to U2 is performed (t15-25). There are 3 different cases:

a) 0X - Instructions which contain the branch address in the K portion.
b) 30-bit instructions which contain one of the two operands in the K portion.
c) 15-bit instructions which do not use K.

(Refer to fig. 2-6.)

The second register, U2, consists of 53 FFs, which holds:

a) Information about the Functional Unit request, and result and operand register selections gained from the translation in U1.
b) The instruction designators ($\overline{f}$, $\overline{m}$, i, j, k).
c) K portion.

This information is now translated and transmitted into the scoreboard (refer to figs. 2-4 and 2-7).

Parcel Counter

The parcel counter is used to extract 4 parcels of 30 bits each from I_N. The 4 parcels are needed in order to pick out any 30 bit or 15 bit instruction combination which could be contained in I_0 and send these instructions to the Scoreboard. (Refer to fig. 2-8.)

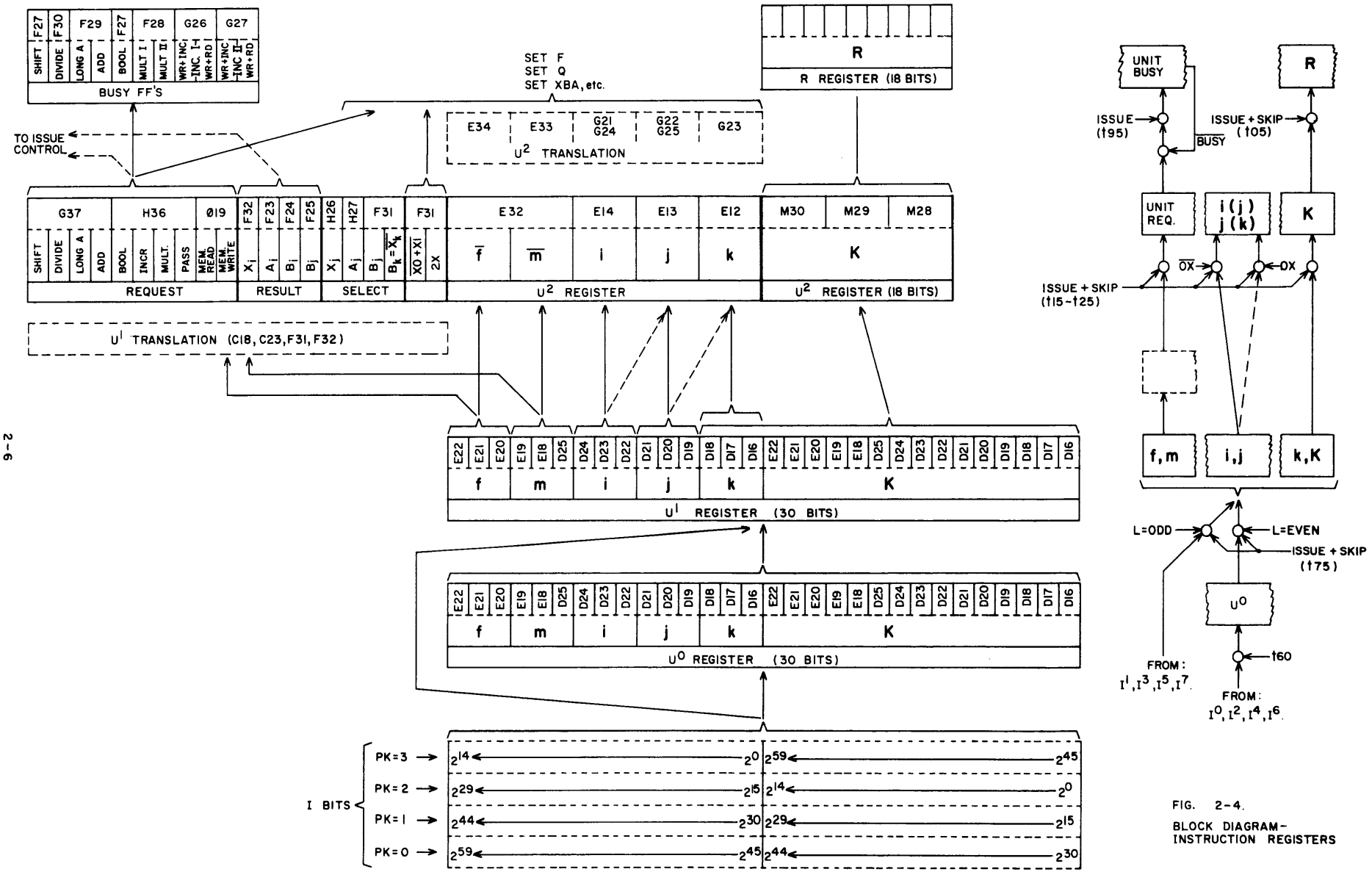


FIG. 2-4.

BLOCK DIAGRAM-
INSTRUCTION REGISTERS

U_2 FF's ↓ \ U_1 fm →		0___ 0 1 2 3 4 5 6 7	1___ 0 1 2 3 4 5 6 7	2___ 0 1 2 3 4 5 6 7	3___ 0 1 2 3 4 5 6 7	4___ 0 1 2 3 4 5 6 7	5___ 0 1 2 3 4 5 6 7	6___ 0 1 2 3 4 5 6 7	7___ 0 1 2 3 4 5 6 7	
UNIT REQUEST	SHIFT			X X X X X X X X		X				G27
	DIVIDE					X X X				
	LONG ADD	X			X X					
	ADD				X X X X X X					
	BOOLEAN		X X X X X X X X							H36
	INCREMENT	X X X X X					X←IF i=0→X	X X X X X X X	X X X X X X X	
	MULTIPLY					X X X				
	PASS					X				
REQU.	MEM. READ						X←IF i=1__5→X			Ø19
	MEM. WRITE						X←IF i=6+7→X			
RESULT	X_i		X X X X X X X X	X X X X X X X X	X X X X X X X	X X X X X X			X X X X X X X	F32
	A_i						X X X X X X X			F23
	B_i							X X X X X X X		F24
	B_j			X X X						F25
SELECT	X_j	X X	X X X X X X X		X X X X X X X	X X X X X	X X	X X	X X	H26
	A_j						X X X	X X X	X X X	H27
	B_j	X X X X X		X X X X X X X		X	X X	X X	X X	F31
	$B_k = \overline{X_k}$	X X X X X					X X X X X X X	X X X X X X X	X X X X X X X	
	$\overline{X0 + X1}$	X X X X X X	X X X X X X	X X X X X X	X X X X X X	X X X X X	X X X X X X	X X X X X X	X X X X X X	F31
	2X			X X X X X X X						

Fig. 2-5. U1 Translation

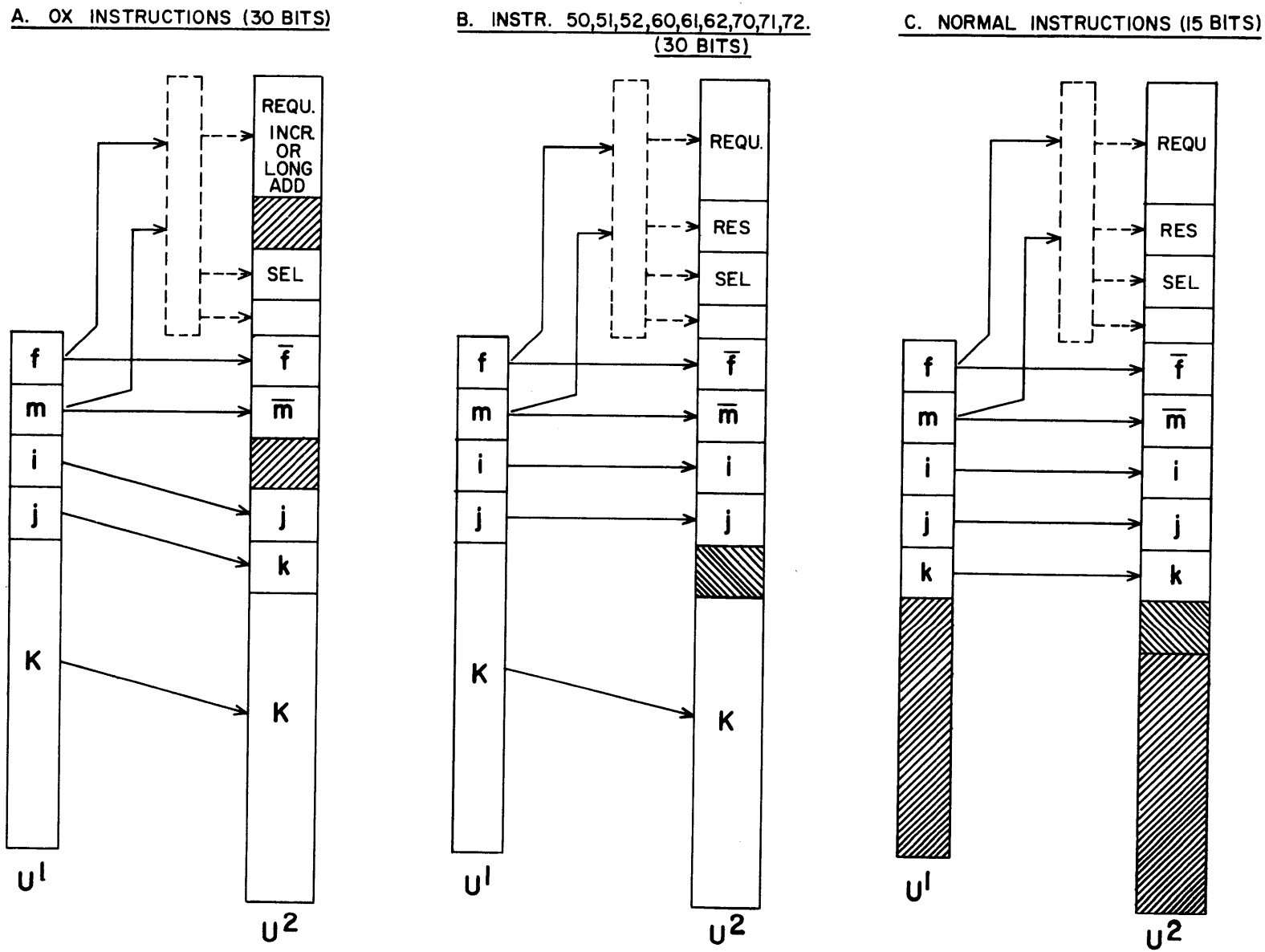


Fig 2-6. $U^1 \rightarrow U^2$ Transfers

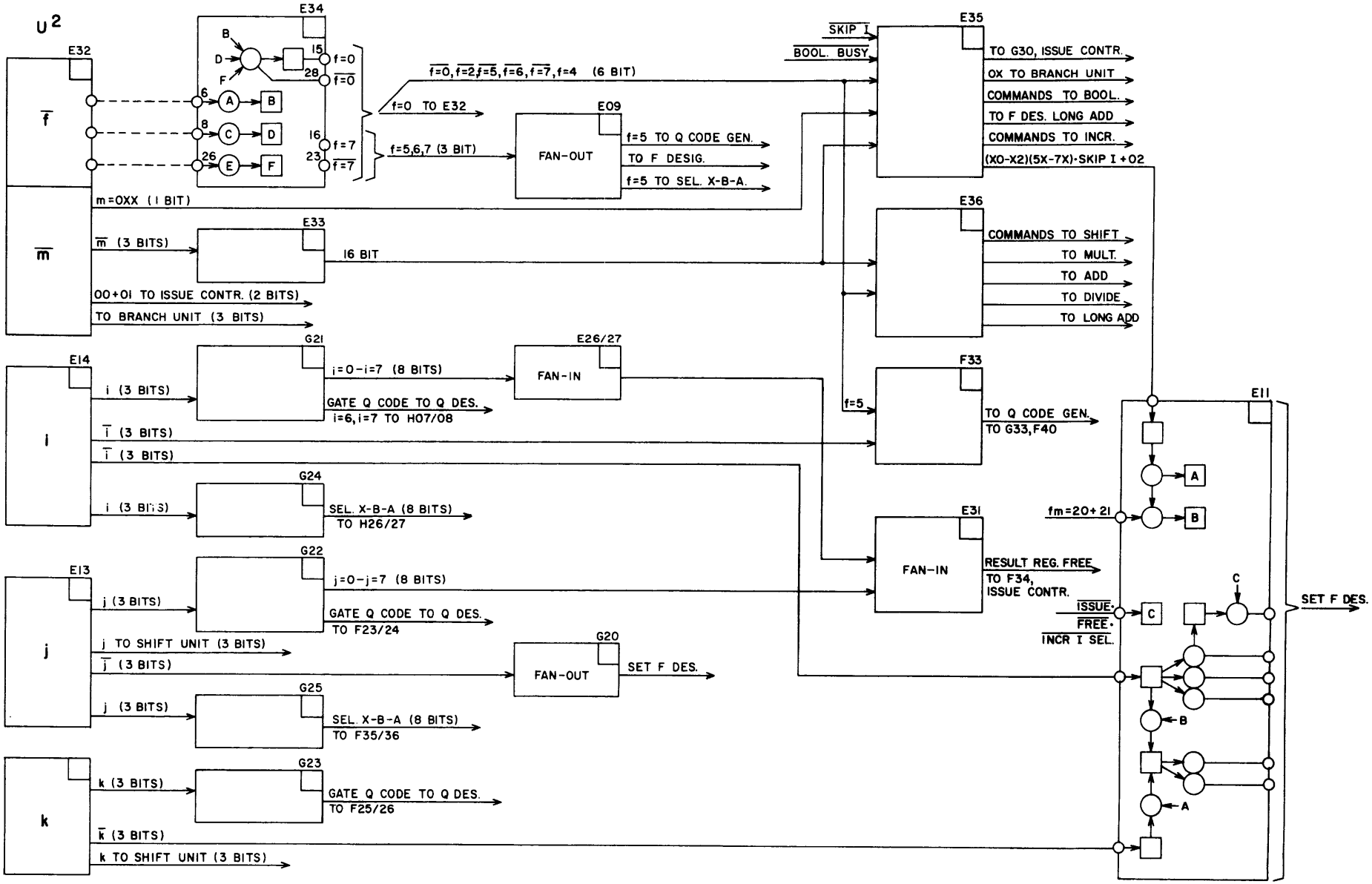


FIG. 2-7. U^2 TRANSLATION

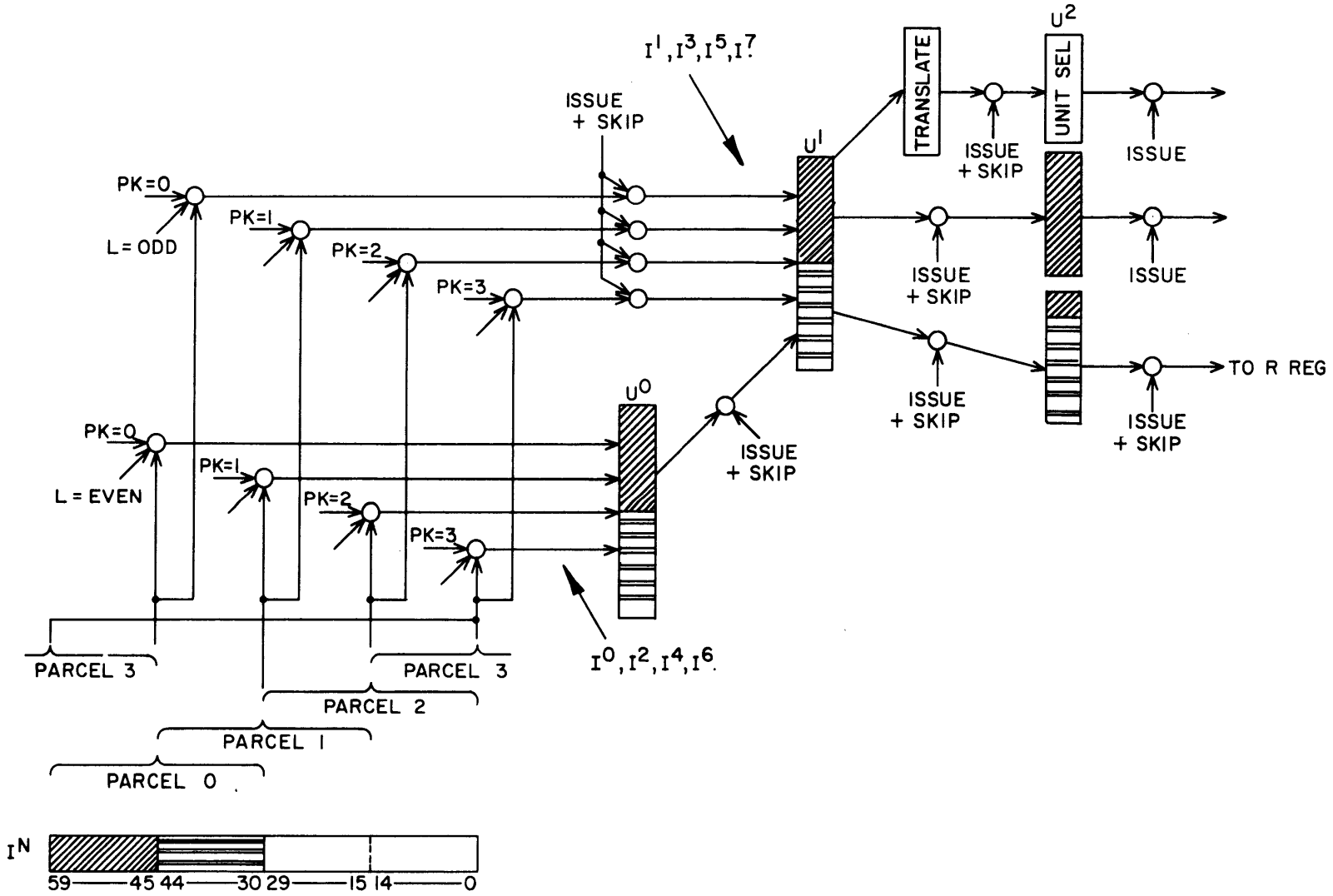


Fig. 2-8. Parcel Counter

15-Bit Instructions

With the first issue pulse, Parcel 0 is sent to U_1 and the parcel counter is advanced to 1. The second Issue Pulse transfers U_1 to U_2; parcel 1 is now in U_0. The ensuing Scoreboard Issue sends the contents of U_2 to the Scoreboard. (Refer to fig. 2-10.)

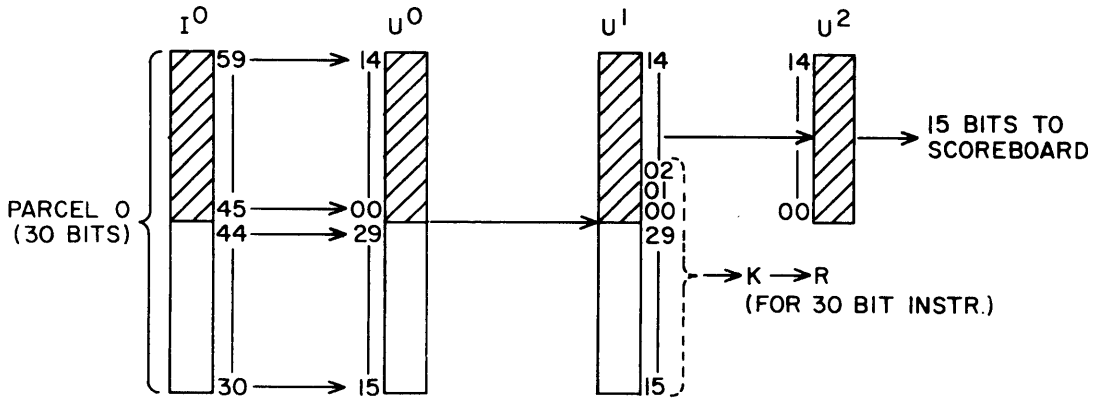
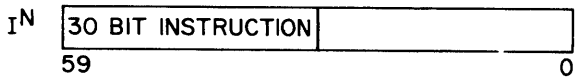
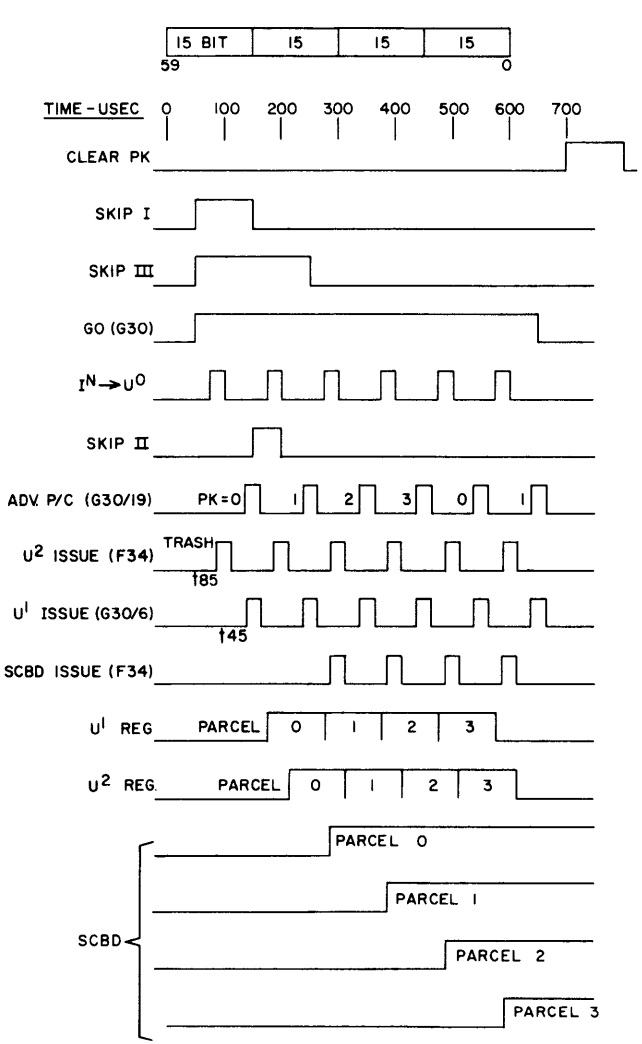


Fig. 2-9. Parcel Path to Scoreboard

After all parcels have been issued to the Scoreboard, the parcel counter will be at count 1. This enables a "0" into F37/P3. Because FFs TP/5 and TP/6 have been set after the inching of I4 to I5, a "0" from F37/P25 goes through H25 and G36 and sets the Stop FF on G30, preventing any more Scoreboard issue pulses. Also, FF A on F37 will be set giving a "1" out of F37/P23, setting the FF TP6 on R32 at t50. One output of this FF is used to send a clear pulse to the Parcel Counter on G31, setting the counter back to 0.

30-bit Instructions (50-52, 60-62, 70-72)





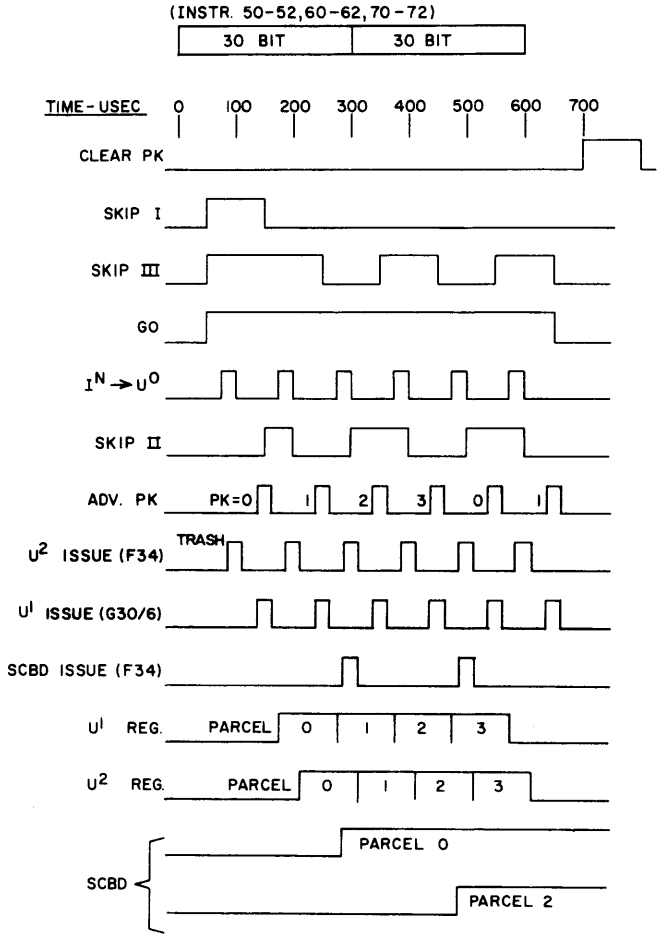


FIG. 2-10. INSTRUCTION ISSUE TIMING.

The proceed signal coming into G30/P11 sets Skip I, III and the GO FF, resulting in a U_2 Issue signal from F34. This enables an Advance PK pulse from G20/P19, and sets the PK to 1. The same Issue pulse also enables parcel 0 into U1. The next Issue pulse (100 ns. later) transfers $U_1 \rightarrow U_2$, parcel 1 into U_1 and advances PK to 2. By this time, the Skip III FF will have been cleared, allowing a Scoreboard Issue pulse.

This pulse transfers the 30-bit parcel 0 from U_2 to the Scoreboard. It also allows the Skip II FF to reset. The 3rd Issue pulse sets PK to 3, transferring parcel 1 to U_2, parcel 2 to U_1. By now, the Skip II FF is reset, disabling any Scoreboard Issue pulses. The 4th Issue pulse transfers parcel 2 to U_2 thereby destroying parcel 1, which was still in U_2. The only difference between a 30-bit and 15-bit instruction is that the Skip II FF is reset to prevent the second Scoreboard Issue pulse and therefore does not send parcel 1 to the Scoreboard.

30-bit Instructions (0X)

The first Issue pulse will again transfer U_0 to U_1 and advance PK to 1. The second Issue will be fed into G30/P23 together with the 0X translation from U_1. This clears FF/E, disabling a second Advance PK pulse and also disabling the P and L incrementers. This Issue also transfers U_1 to U_2 and parcel 1 into U_1. Together with the third Issue, a Scoreboard Issue transfers U_2 to the Scoreboard. This Scoreboard pulse, together with the 0X translation from U_2, is fed into H05/P9 setting the Stop FF. This disables any more Scoreboard Issue pulses. Instruction issue control now waits for an indication from the Scoreboard as to whether the jump condition was met or not. (Refer to figs. 2-11 and 2-12.)

INCH COUNTER

The inch counter is used to shift 60-bit instruction words upward in the instruction stack (I_0-I_7). When the parcel counter = 0, L counter = 0 and t = 60, and an Issue or Skip signal occurs, a "0" into G29/11 sets the Inch FF. This FF sends an advance pulse to the inch counter. Before the counter is advanced to 1, gates C and D on G29 are enabled setting FF/TP1. One output of this FF will set the Program Address FF on P37, starting a new RNI. Another output first clears

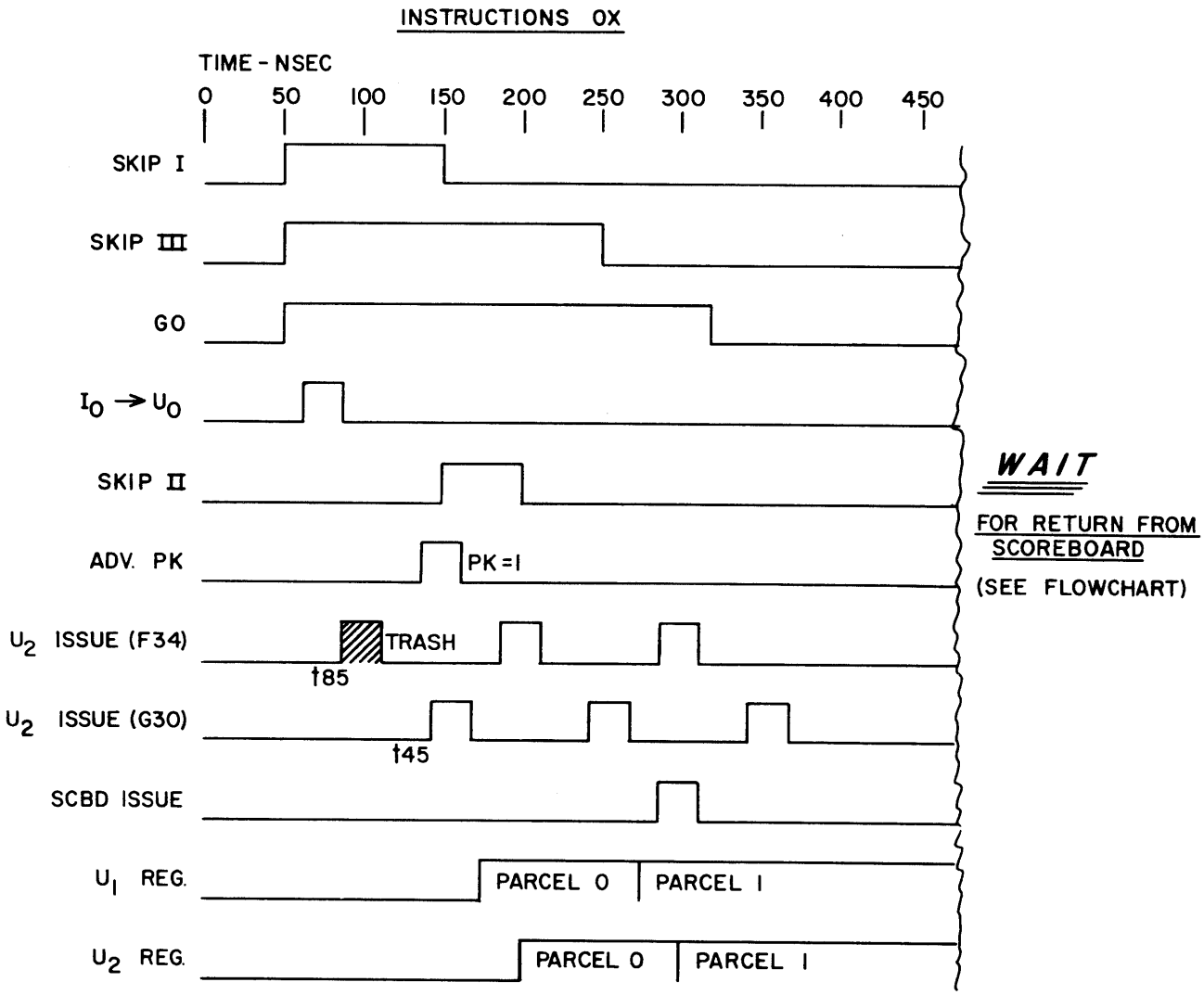


Fig. 2-11. OX Instruction Issue Timing

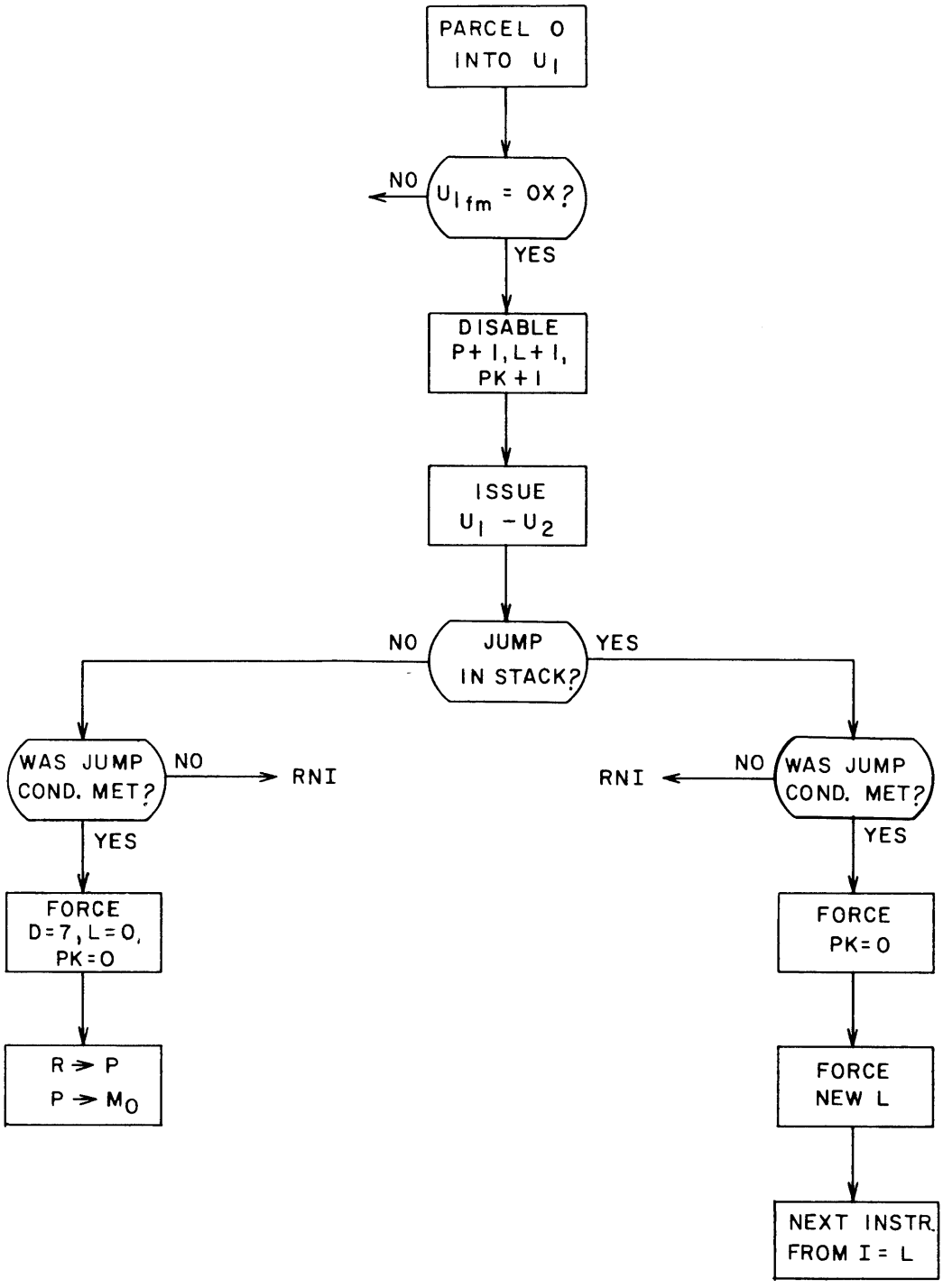


Fig. 2-12. 03-07 Instructions

the I_7 register and then enables the transfer of $I_6 \rightarrow I_7$. Another pulse from this FF, at a slightly later time, clears the I_6 register and transfers $I_5 \rightarrow I_6$. (Refer to fig. 2-13.)

By now, the inch counter is set to 1 enabling the transfer of $I_4 \rightarrow I_5$ and $I_3 \rightarrow I_4$ etc. After the last transfer has been enabled ($I_0 \rightarrow I_1$), the counter is set at 3. The 4th advance pulse now sets the counter back to 0 and the Inch FF is cleared.

The counter is again ready to begin a new transfer sequence. The instruction word now in I_7 is discarded when the next $I_6 \rightarrow I_7$ transfer occurs.

L COUNTER

To avoid confusion in the L counter discussion which follows, the following statements should be noted:

1) The L counter FFs on H28/TP1, 2, 5 contain the complement of the L count. (e.g., all set: L count = 0; all clear: L count = 7) Note the distinction between L counter and L count.

2) Reduce L means the L count goes from $6 \rightarrow 5$ or $3 \rightarrow 2$.

The major function of the L counter is to guide the parcel extraction from the proper register (e.g., I0, I1 etc.) in the instruction stack; an L count of 0 corresponds to I_0, L = 1 to I_1 etc. Assume L = 0 and PK = 0; this means that the upper 30 bits of I_0 would be gated to U_0. When PK advances to 1, the next parcel will be transferred etc. By now, the Inch counter also started the inching of $I_6 \rightarrow I_7$, $I_5 \rightarrow I_6$ etc. During this time the L count was equal to 0.

When the PK reaches 3, and no 0X (Branch) instruction is in the U^1 translator, a "0" into G28/P17 will enable the H gates and disable the G gates.

The function of these H and G gates are to switch the L counter FFs from one loop to another. One loop called the L-T loop goes from H28 through H29, G36, G28 and back to H28. The second loop, called the Reduce L loop, goes from H28 through H30, G28 and back to H28. In a no branch instruction the T FFs on H28 are all set which means that in the L-T loop we are actually subtracting 0 from L. This permits holding L to its same value through the L-T loop as long as the G gates are made. As soon as we switch to the Reduce L loop we will also

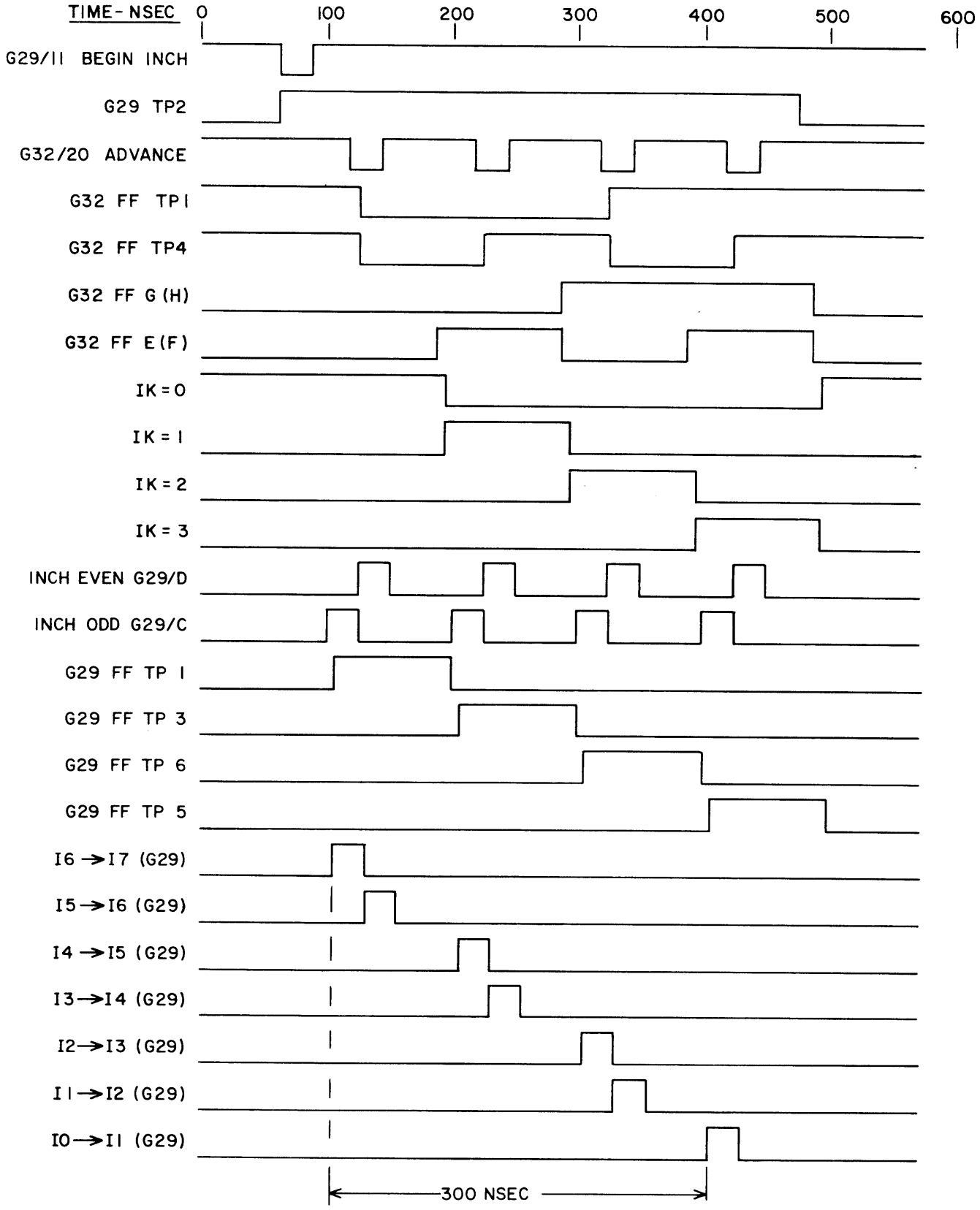


Fig. 2-13. Inch Counter

get a "1" into G28/P12 (from G29/P18) disabling the H and G gates. This prevents FF L_0 from holding its value and it is cleared at the next t50.

This means that we have advanced the L count to 1. However, as soon as the GO FF on G30 is cleared, a "0" out of A on G36 will set all L FFs on H28 forming an L count of 0. Parcelling of the next instruction word from I_0 is now ready to begin.

In case of a conflict (instruction not accepted by the Scoreboard), all Issue pulses are stopped preventing Advance P and Advance PK. However, the Inching process will continue until completion. After the inching of $I_1 \rightarrow I_2$ is completed, the L counter is again advanced setting the count to 1; the count remains set at 1 since the GO FF remains set. This will allow a new instruction to be entered into I_0 while we are waiting for our instruction to be accepted. Once the instruction is accepted by the Scoreboard, the following Issue will advance the Parcel counter again but the remaining parcels are taken from I1. We now have to wait until all instructions out of I_1 are accepted before we can start a new Inching process and therefore a new RNI.

For a branch instruction (if conditions are met and jump is in stack), the L-T loop supplies the new L count and sets the L FFs to the new value. After all the instructions in this new instruction stack register (defined by the L count) have been issued, a count enable K on H30, together with the H gate on G28 allows the L count to be reduced by one, and so on until L is again equal to 0.

At the beginning of the branch, the inching also transferred $I_0 \rightarrow I_1$ but now we cannot start a new inching process because we need L = 0. There is also only one RNI immediately after the inching has started but as we cannot start a new inching we will not get a new RNI.

Remember:

To start Inching: (PK = 0)(L = 0)(Issue pulse)
To start RNI: Inch I_6 - I_7

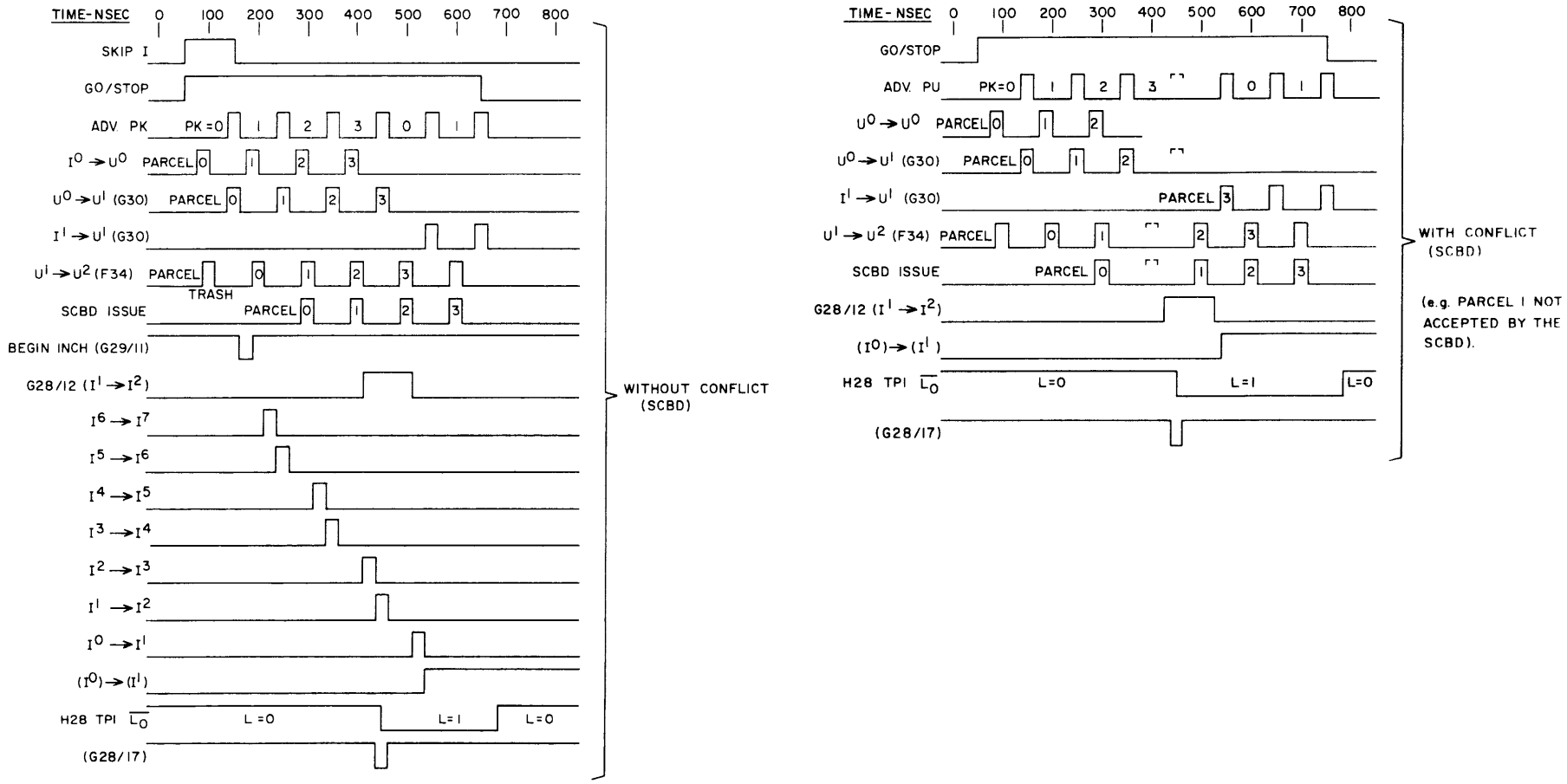


FIG. 2-14. L-COUNTER

ISSUE CONTROL

The Issue Control block diagram in fig. 2-15 depicts the functions performed by the Issue pulses from F34 and G30. As not all of these pulses are produced by the same conditions, we shall give them the following names:

1) Pulses from F34/P9, 11, 13 are called: $\underline{U^2 \text{ Issues}}$
2) Pulses from G30/P6 are called: $\underline{U^1 \text{ Issues}}$
   } also called: Issue+Skip
3) Pulses from F34/P8, 14, 19/26, 28 are called: Scoreboard Issues
   also called: Issues

These pulses are produced if the following conditions exist:

U^1 and U^2 Issues = (Unit Request . $\overline{\text{Unit Busy}}$. Dest. Reg. free . GO) + Skip
Scoreboard Issues = (Unit Request . $\overline{\text{Unit Busy}}$. Dest. Reg. free . GO . $\overline{\text{Skip}}$)

STOP INSTRUCTION ISSUE

The Issue signal, which moves instructions and their translations through U1 and U2 into the scoreboard, and the Skip signal, which moves these instructions up to U2, are controlled by the G0/Stop FF. Conditions necessary for these signals are as follows:

ISSUE: (Go) (Skip) (Unit Request) (Unit Free) (Destination Register Free) (t90)

SKIP: During the 200 ns. after Proceed and during 100 ns. after a 30-bit instruction where K is an operand.

The following section describes the conditions that place the Go/Stop FF in the Stop state. (Refer to fig. 2-16.)

1) PAUSE

When the first parcel is extracted from the 60-bit instruction word in I_0, the inch process begins. The beginning of the inch process, in turn, requests the next instruction word from central memory. When all parcels of the word in I_0 (meanwhile, inching may have moved the word into I_1) have been transferred to the scoreboard, the central processor waits for the next instruction word. This wait period is called Pause.

To get the Pause signal, the inch process sets the Request Pause I FF. When parcel 2 is taken from the stack, the Request Pause II FF is set. The next time PK = 1, parcel 3 is placed in the scoreboard and issue is stopped. (Refer to figure 16.) The Issue and Skip signals are restarted by instruction ready control.

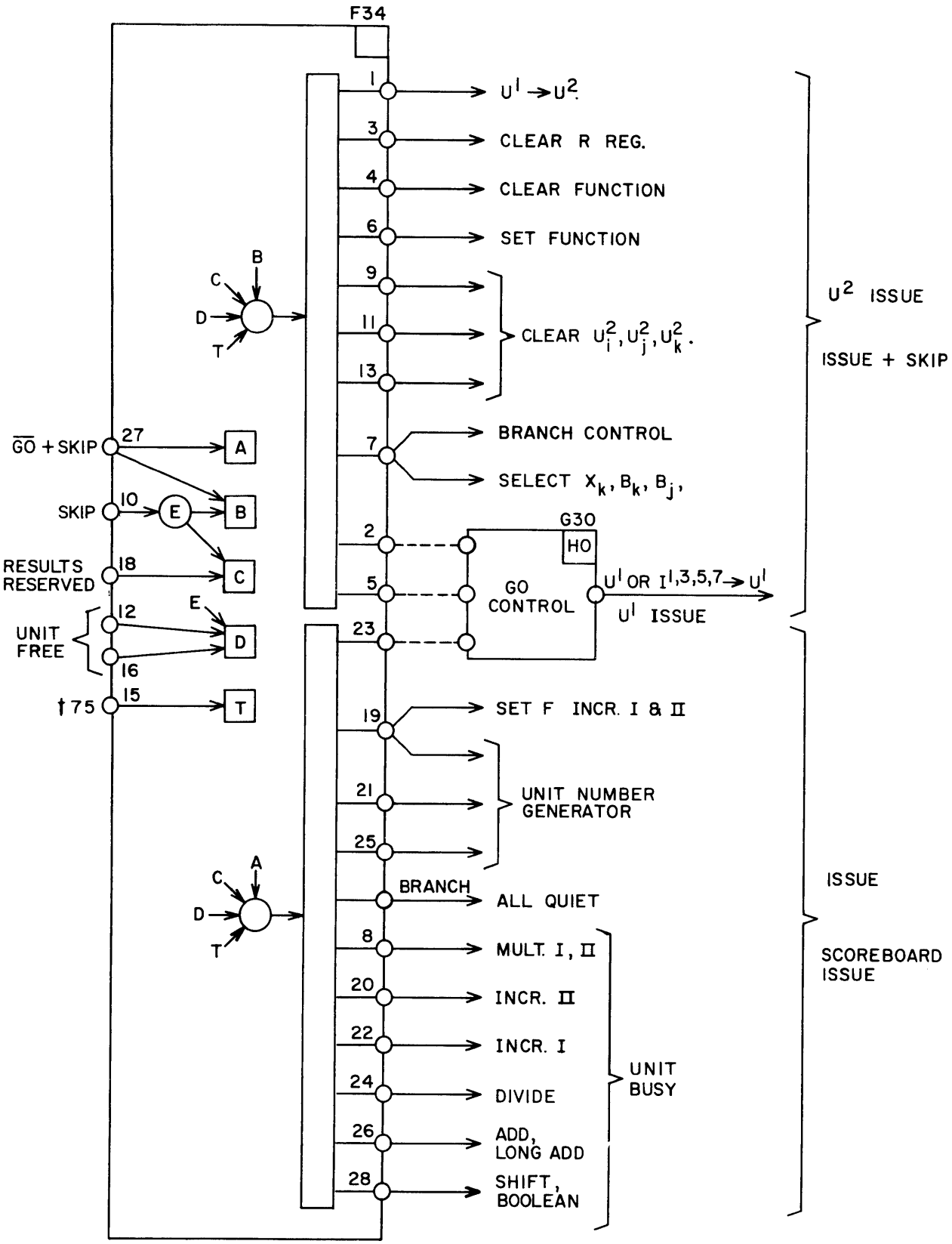


Fig. 2-15. Issue Control

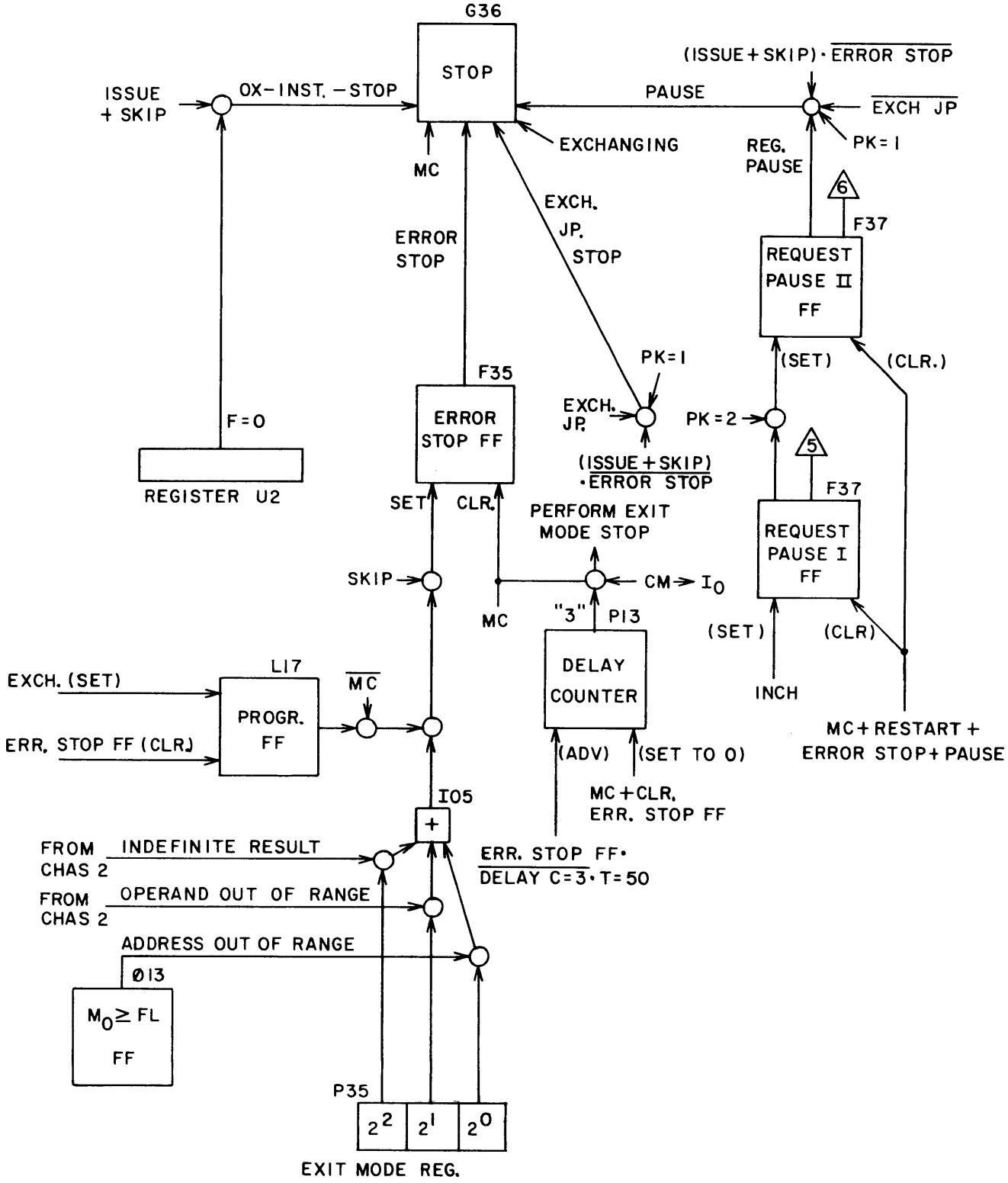


Fig. 2-16. Stop Instruction Issue

2) EXCHANGE JUMP STOP

During the Exchange Jump process, the Go/Stop FF is set to the stop state. The Exchange Jump signal, issued by a peripheral processor, forces the stop when PK = 1. (A PK = 1 count indicates parcel 3 was the last parcel issued to the scoreboard. Thus, an entire 60-bit instruction word is issued before permitting the stop on an Exchange Jump.)

3) ERROR STOP

The contents of bits 2^0 - 2^2 in the Exit mode register specify which errors can stop instruction issue. The Program FF, set at the beginning of a new program by the Exchange Jump, insures that only the first error sets the Error Stop FF. The error signal is also gated by a Skip signal. The Skip signal gate permits finishing a loop in the stack (for branch cases) before stopping. (Only the next Branch instruction performs a Skip.)

The Error Stop FF is cleared by a master clear or by the execution of the Exit mode process.

4) OX - INSTRUCTION STOP

When a Branch instruction is issued into the scoreboard, further instruction issue halts to allow time for branch condition tests; i.e., branch/no branch, jump/loop, etc.

5) MASTER CLEAR

A master clear stops all instruction issues.

PROCEED INSTRUCTION ISSUES

The requirements for proceeding with instruction issues are outlined below:

1) INITIAL START OR RESTART AFTER PAUSE:

When the central processor starts (after an Exchange Jump), or proceeds (after referencing memory for the next instruction word; i.e., pause), two transfer pulses are required to bring the first parcel (parcel 0) from I0 into U2. No information is placed in the scoreboard during this period; these transfer pulses that move the parcel to U2, but not to the scoreboard are called SKIPS.

2) PROCEED AFTER A BRANCH INSTRUCTION:

Three possible "proceed" cases exist after a Branch instruction:

a) BRANCH: When the branch condition is not met, the next instruction may be in the same 60-bit instruction word as the Branch instruction.

b) LOOP: The branch condition is met, and the next instruction is parcel 0 of an instruction word in the stack.

c) JUMP: The branch condition is met; the next instruction must be obtained by a memory reference.

Case (a), if the Branch instruction was in parcel 0 or 1, requires one parcel be skipped. In cases (b) and (c), where a new instruction word is requested, it is necessary to perform two skips to transfer the new instruction into U2.

For simplicity, the logic treats all "proceeds" after a stop condition the same (i.e., as requiring two skips). Case (a) above, the only case requiring only one skip, is reverted to the normal case by disabling one advance pulse of the parcel counter. Thus after every stop condition, two skips are performed before information can be issued to the scoreboard.

3) In cases (1) and(2c), which require a new instruction word from memory, a "proceed" signal occurs only if the new instruction word is available in I0.

OPERATION

A Proceed signal may originate from one of three different sources (cases): (Refer to fig. 2-17.)

1) Restart (Instruction Ready Control)
2) Loop
3) Branch

RESTART:

The Restart signal occurs when a new instruction word is available in I0. This occurs in the following cases:

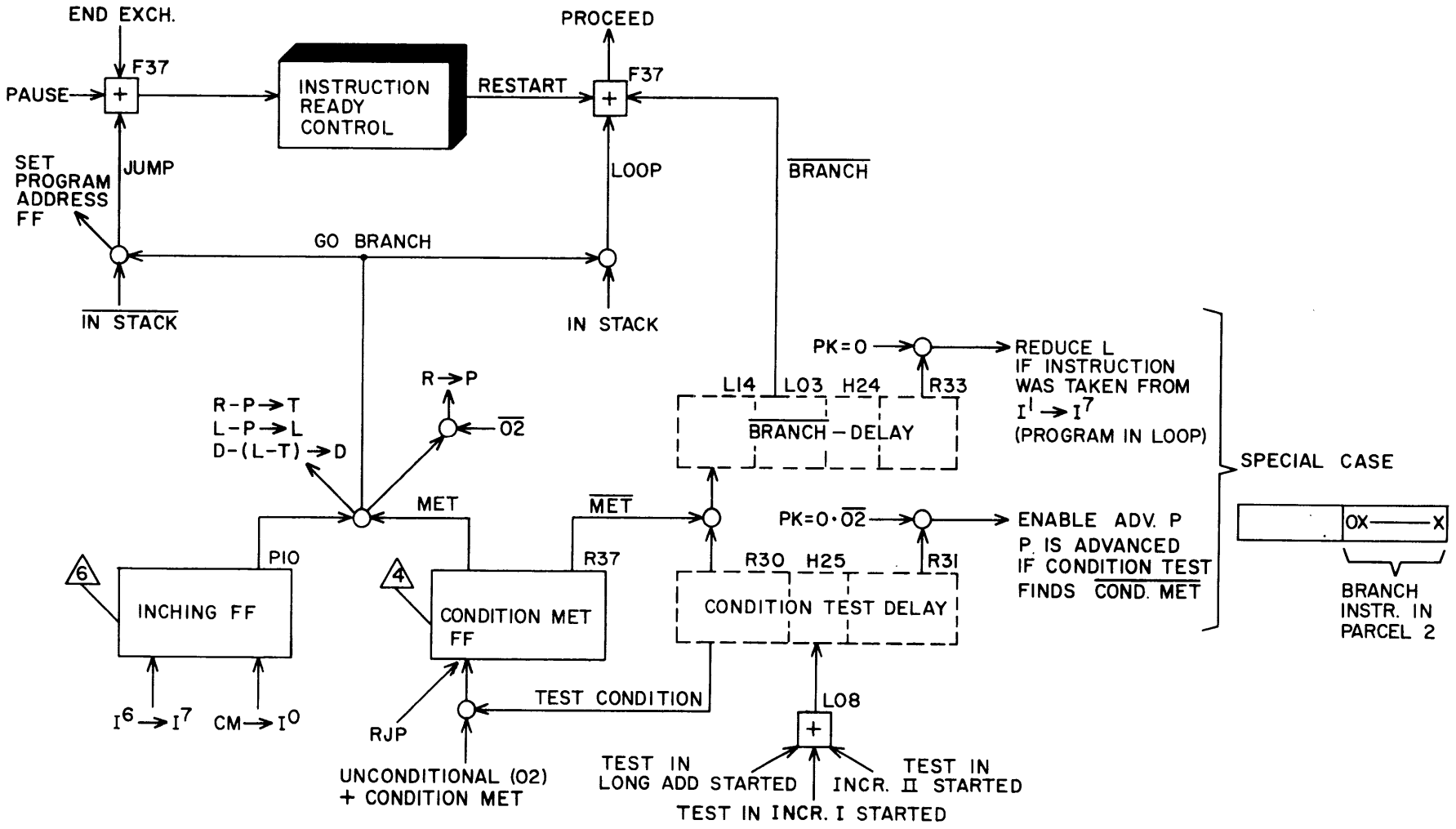


Fig. 2-17. Proceed Instruction Issue

a) End Exchange (start after an Exchange Jump)
b) Pause (stop between instruction words)
c) Jump (Branch is not in the stack)

The corresponding signal sets the CP Stopped FF, clears L, PK, and (in the jump case) the D count, and sets the Enable Restart FF. (Refer to fig. 2-18.)

When the 10-tag (RNI tag) is detected in the hopper and the address is accepted by central memory, the new instruction word is read from memory. The new instruction word is guided to I0, and the Instruction Available FF is set, permitting the Restart. Note that it is possible for an instruction to be available before the Enable Restart FF is set. For example, a Functional Unit conflict before Pause, or (in the jump case) when a requested instruction word must be in the stack before the Go Branch signal occurs.

LOOP:

If the Branch test indicates that the next instruction is located in the stack, the Proceed signal is issued. (In case the inch process is occurring, the Inching FF delays the Go Branch signal until movement in the stack is completed.)

NOT BRANCH:

In a conditional Branch instruction, and the branch condition is not met, the Proceed signal can be issued. (As previously stated, the parcel counter skips one advancement in Branch instructions; this permits two skips following a Proceed signal.)

A special case exists for a $\overline{\text{branch}}$ condition. During a Branch instruction, the P and L counters are disabled. This disabling occurs from the moment a Branch instruction is detected in U1 until the Proceed signal is issued. (Meanwhile, the parcel counter is advanced by one.) If a Branch instruction comes from parcel 2 of an instruction word and enters U2, the parcel count is now 3. Normally on PK = 3, P is advanced and L is reduced. However, in Branch instructions this does not occur. Action proceeds with a PK = 0; PK is counted to 2 during the two Skips, then a stop occurs to wait for the new instruction word. Since this action would not provide opportunity to correct the counts in P and L, special circuits exist to accomplish this before the Proceed signal is given.

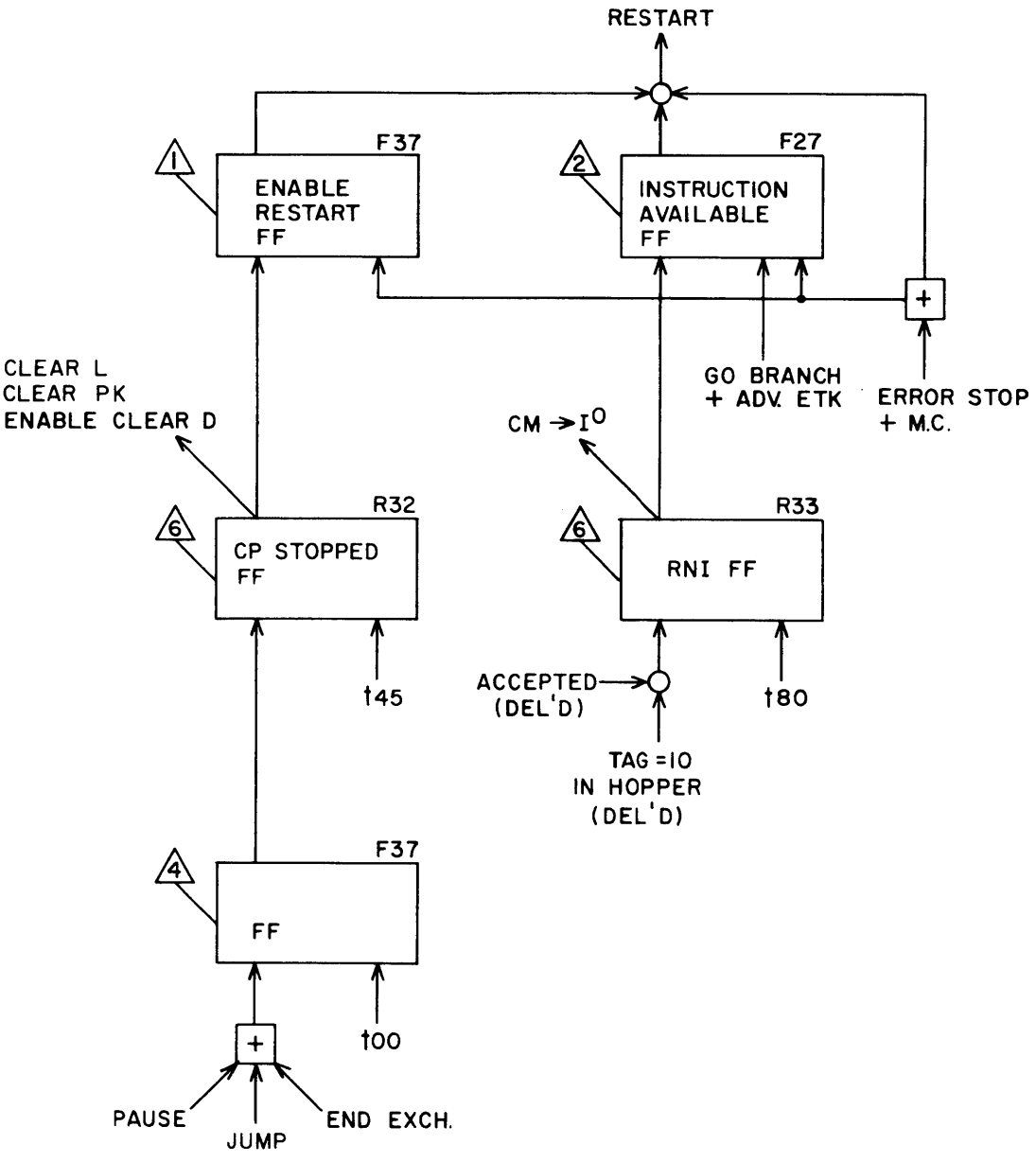


Fig. 2-18. Instruction Ready Control

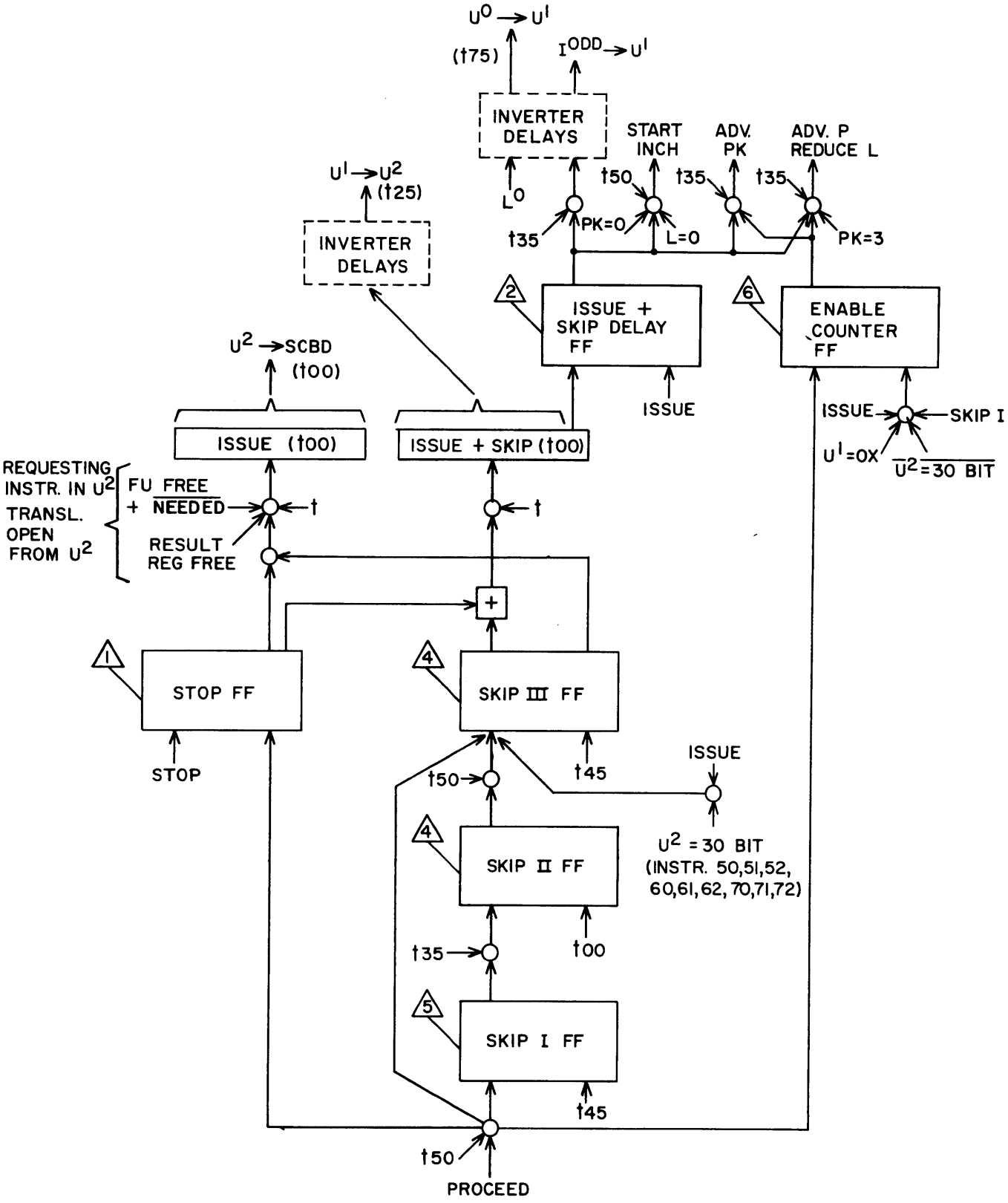


Fig. 2-19. Go Control/Issue/Skip

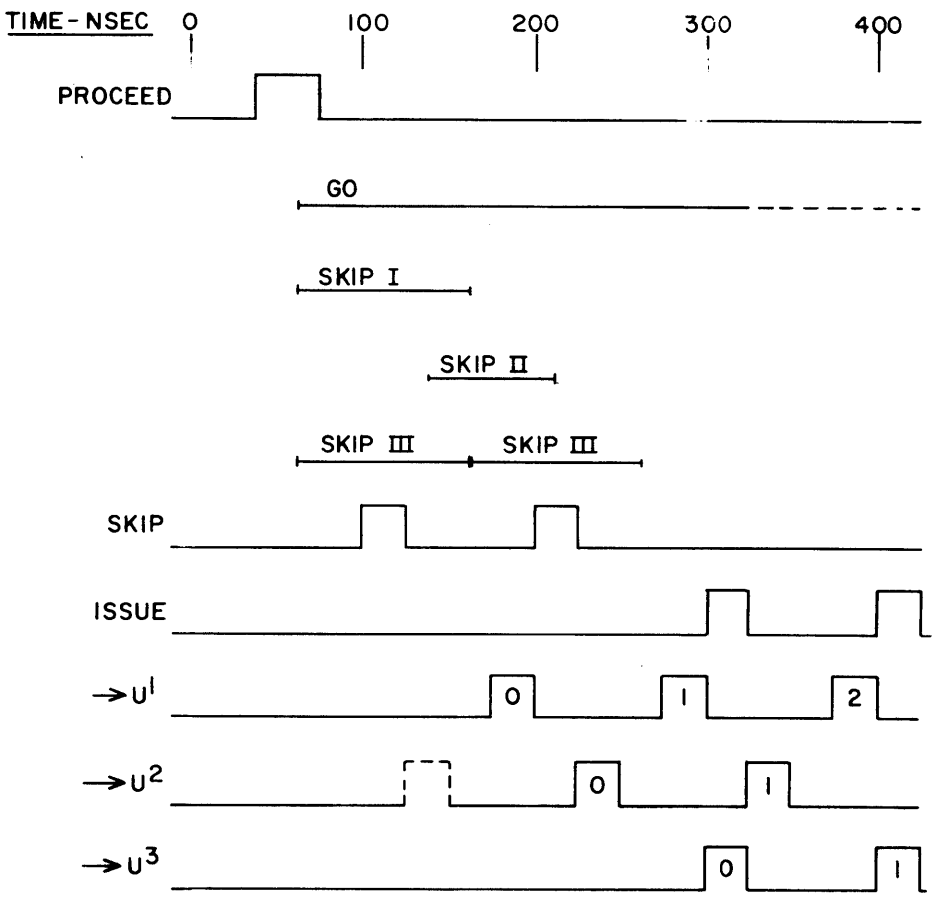

TIME - NSEC
0
100
200
300
400
PROCEED
GO
SKIP I
SKIP II
SKIP III
SKIP III
SKIP
ISSUE
→U1
0
1
2
→U2
0
1
→U3
0
1

Fig. 2-19a

SECTION III

Central Processor Scoreboard

SCOREBOARD

Introduction

The scoreboard directs the exchange of operands and results among the 24 operating registers, central memory, and the 10 functional units. Instructions are issued to the units for execution in the order prescribed by the original program sequence. The scoreboard permits instructions to be executed out of order while retaining the original program sequence.

Scoreboard reservations are made in serial order for one unit at a time and at a maximum rate of one every minor cycle; this corresponds to the maximum instruction issue rate. However, processing of requests to read operands or store results from all units goes on in parallel and coincidently with placing of reservations. Hence, program instructions are executed in a series-parallel arrangement resulting in very high-speed running of a program.

The scoreboard uses a reservation system on the registers and units to direct the instruction issue and execution sequence. The scheme allows all units to be in operation at the same time but prevents one unit from executing more than one instruction at one time or more than one unit from storing in a common register at the same time. Lockouts for the functional units and operating registers and a series of designators which identify registers and units aid the control system.

An instruction is issued to a functional unit only if the unit and the required result register are free; i.e., they are not reserved for another instruction or another result. With both elements free, the instruction is issued and reservations placed in the scoreboard for the necessary operating registers and unit (fig. 3-1).

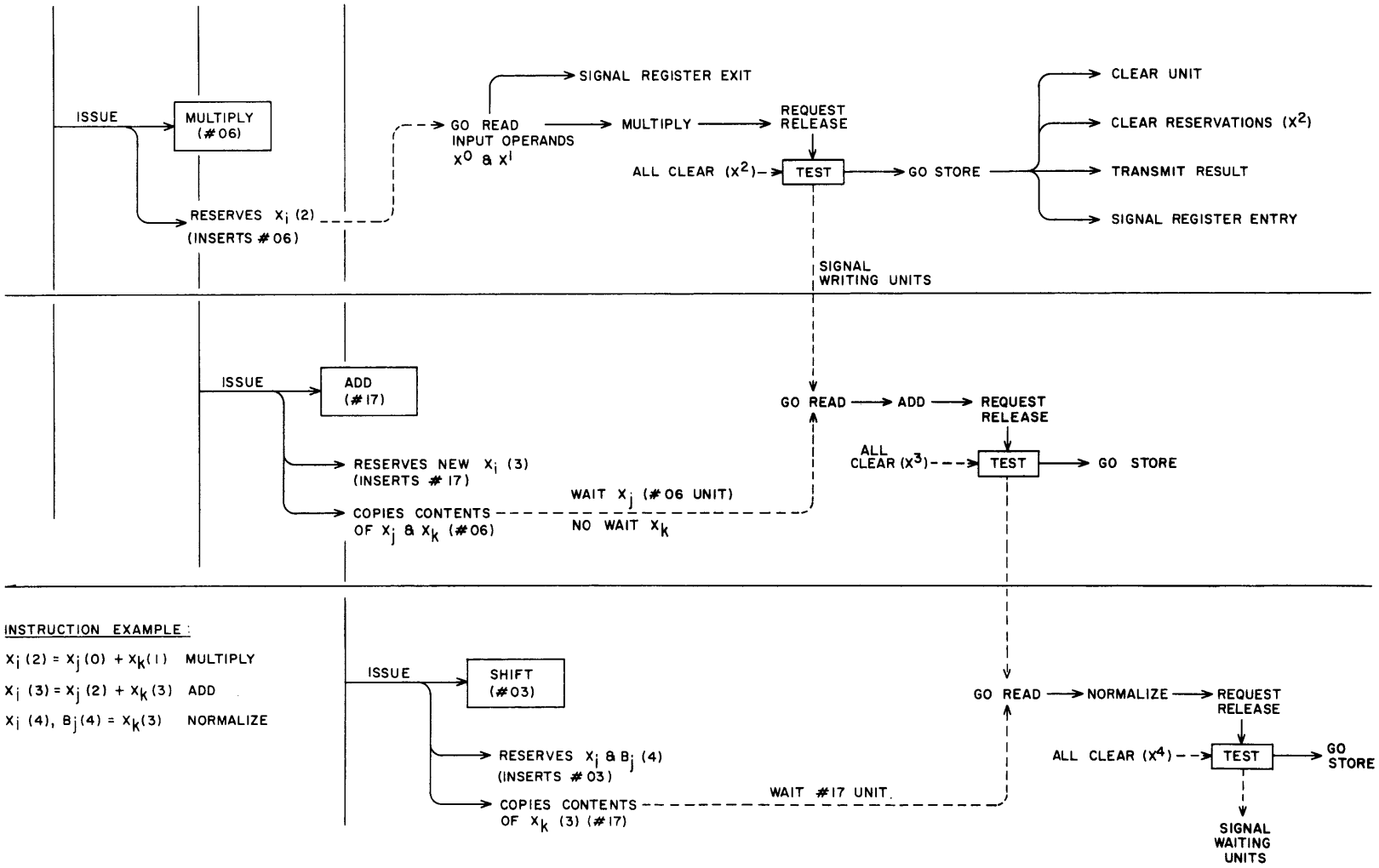


FIG. 3-1.
SCOREBOARD CONTROL

Scoreboard control directs the unit in obtaining its operands and storing its result; computation in the unit proceeds independently. The unit requests permission from the scoreboard to release its result to its result register. The scoreboard determines that the path to the register is clear and signals the requesting unit to release its result. The releasing unit's reservations are then cleared and units waiting for the result are signalled to read the result for their computation.

Special scoreboard action handles the reservation and control scheme for address modification in the increment units. A change to an A^1-A^7 address register changes the corresponding X^1-X^7 operand register so that an operand is read into X^1-X^5 or X^6-X^7 is stored. Hence, for 50-57 instructions, the A result register and associated X register are reserved together to prevent the X register from being pre-empted by a subsequent instruction.

The increment unit computes the address and sends it to the stunt box for issue to CM. Coincidently, the result is released to the A register through normal scoreboard processing. Also, the releasing increment unit's reservations are cleared to free the unit for further computation. However, the X register reservation is held until after the address is accepted by CM (thus spanning any delay time in the stunt box because of bank conflict) and the data word is delivered to the X register (for X^1-X^5 cases) or delivered to the CM store distributor (for X^6-X^7 cases). Additional information is given later for the increment and branch unit action in the scoreboard.

The functional units exchange data with the 24 operating registers over a number of data trunks. A priority system regulates trunk usage, and the scoreboard selects the proper data trunk and honors the priority system in directing the data exchange.

The designators in the reservation system are listed below and shown in fig. 3-2.

X	Operand register	j	First entry operand register designator
B	Increment register	k	Second entry operand register designator
A	Address register	i	Result register designator
F	Functional	Q	Entry operand reservation designator

Note that reservation designators are part of the scoreboard circuits but are shown in fig. 3-2 as related to the entry operand and result registers of a given unit.

Scoreboard operation is described first from the view of placing reservations, and second, the control system which interprets reservations and directs a unit to read its entry operands and store its result.

Placing Reservations

Reservations are placed in three sequential steps following issue to the scoreboard.

1. Assign operating registers to functional unit (set F).
2. Determine previous reservations on entry operand regs. (set Q).
3. Update reservation list to reflect the result register of the latest reserving unit (set XBA).

Set Unit Busy FF

The fm portion of the instruction is translated at the output of U^1 selecting one of the following Unit Request FFs:

a) Shift, Divide, Add, Long Add, Boolean (G37, H36)

The outputs of these FFs are sent to their respective Unit Busy FFs, (F27, 29, 30). The input gates of these cards, however, can only be made if the Unit Busy FF is in the clear state. If this gate is made we will enable the D gate on F34 of the Issue Control.

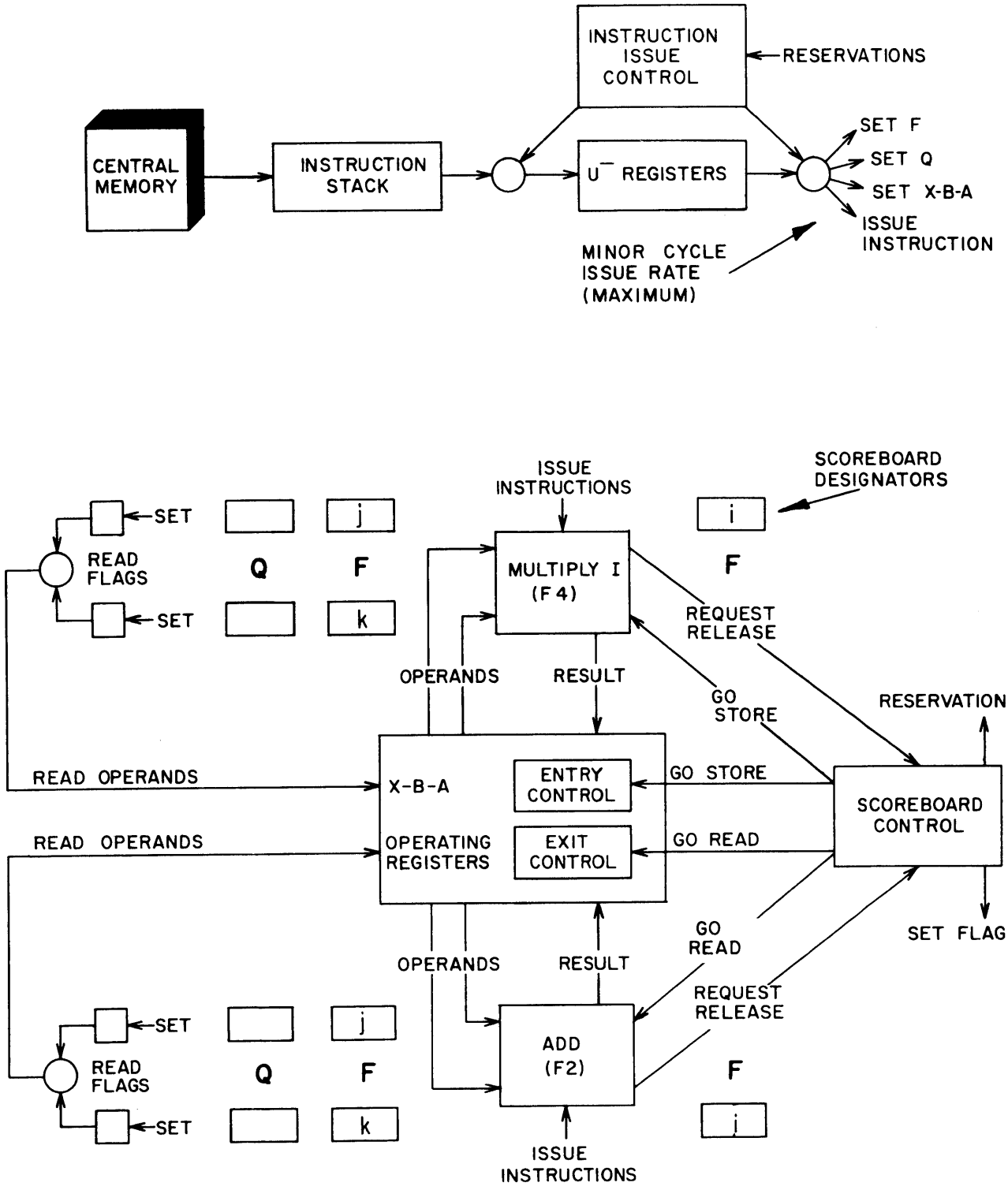


FIG. 3-2. RESERVATION DESIGNATORS

b) Multiply, Increment (H36)

As there are two Multiply and Increment units, these unit request FFs can set one of two Busy FFs under the same conditions described in paragraph a). To set Mult. II or Incr. II Busy FF, their respective unit I Busy FFs must be set.

For 5X instructions when i=0, only FF CD (Write or Increment) will be set on G26 and G27. FF AB (Read or Write) remains cleared, because no memory reference can be made by changing A0.

c) Memory Read, Memory Write (Ø19)

A 5X translation and i=6+7 will set the Memory Write FF on Ø19. A 5X translation and i=1-5 will set the Memory Read FF. These FFs in turn can set the Busy FFs for increment I or II on G26 if all conditions are satisfied.

d) Pass (H36)

This FF on H36/TP6 will send its output directly to the Issue Control to enable gate D on F34.

SET F

The set F step sets the result and entry operand register designators i, j, k for the selected unit. This assigns operating registers to the selected unit. The i designator identifies a result register for most instructions and j and k the entry operands. Each 3 bit designator specifies one of the eight registers in an X, B or A register group. For example, the j designator of the add unit may be set to X^1, which reserves operand register X1 as one of the entry operand registers.

Looking at Table 3-1 we can see that for the Shift and Increment Units, the F designators can specify X, B or A registers. For these units therefore, we must decide which register F designator must be set for a particular instruction.

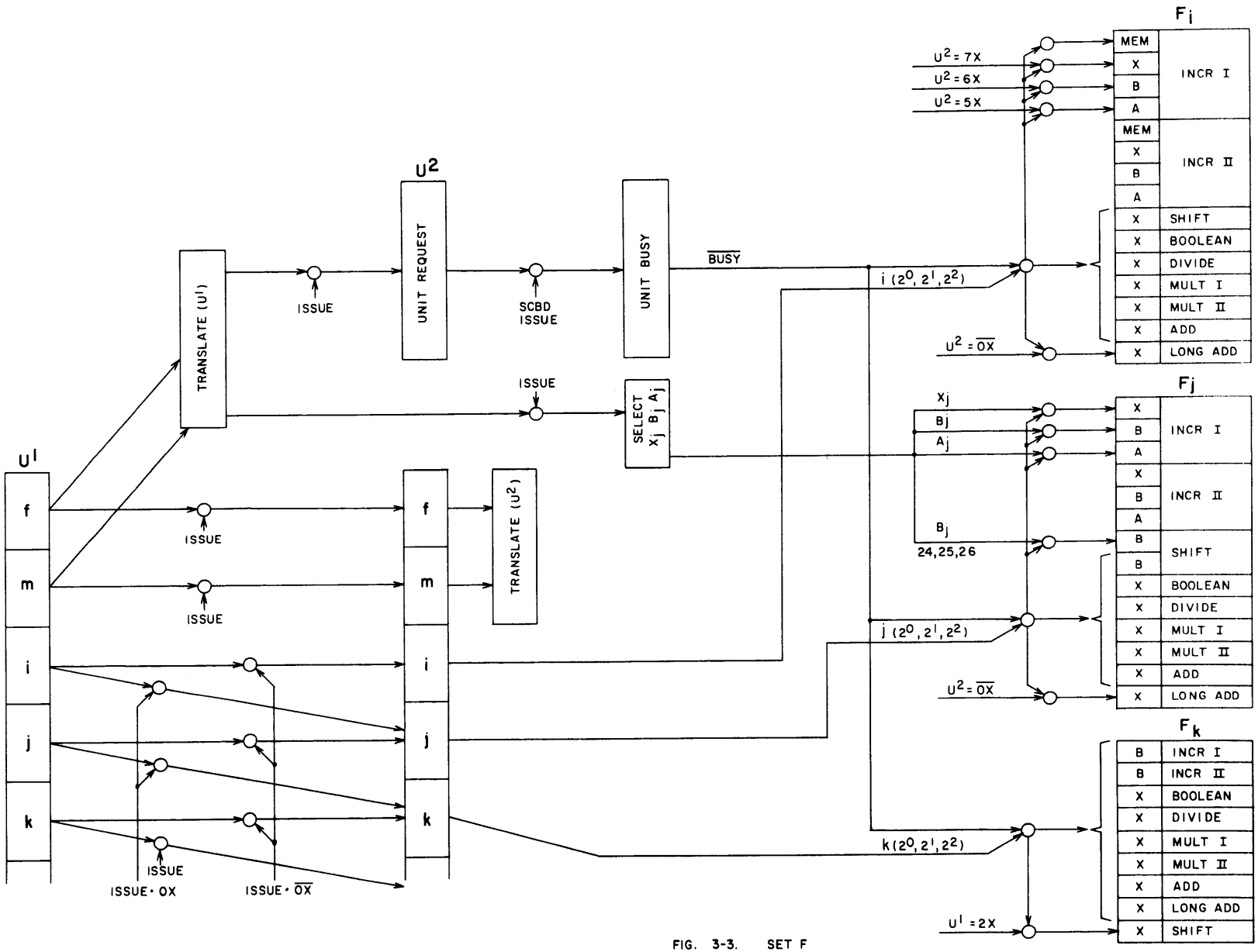

U1
f
m
i
j
k
TRANSLATE (U1)
ISSUE
U2
UNIT REQUEST
SCBD ISSUE
UNIT BUSY
BUSY
ISSUE
SELECT Xj Bj Aj
ISSUE
ISSUE
f
m
i
j
k
TRANSLATE (U2)
ISSUE
ISSUE · OX
ISSUE · OX
Fi
U2 = 7X
U2 = 6X
U2 = 5X
MEM
X
B
A
INCR I
MEM
X
B
A
INCR II
X SHIFT
X BOOLEAN
X DIVIDE
X MULT I
X MULT II
X ADD
X LONG ADD
i (20, 21, 22)
U2 = OX
Fj
Xj
Bj
Aj
X
B
A
INCR I
X
B
A
INCR II
Bj
24,25,26
B
B
SHIFT
X BOOLEAN
X DIVIDE
X MULT I
X MULT II
X ADD
X LONG ADD
j (20, 21, 22)
U2 = OX
Fk
B INCR I
B INCR II
X BOOLEAN
X DIVIDE
X MULT I
X MULT II
X ADD
X LONG ADD
X SHIFT
k(20, 21, 22)
U1 = 2X

FIG. 3-3. SET F

UNIT	F_j			F_k		F_i		
	X	B	A	X	B	X	B	A
ADD	X			X		X		
MULTIPLY I	X			X		X		
MULTIPLY II	X			X		X		
DIVIDE	X			X		X		
LONG ADD	X			X		X		
BOOLEAN	X			X		X		
SHIFT		X		X		X	X	
INCREMENT I	X	X	X		X	X	X	X
INCREMENT II	X	X	X		X	X	X	X

TABLE 3-1.

This selection is done by the U^2 translation and the select Xj, Bj, Aj FFs which will enable the required F designator or designators.

SET Q

The set Q step determines if the entry operand registers are reserved for results of other units already in operation, and which units have them reserved. The fact that the current instruction has been issued to a unit indicates that its own result register is not reserved. The register reservation information is held in 24 separate X-B-A designators. Each unit is assigned a number for reservation purposes, table 3-2, and the number of the last reserving unit is transferred from the proper X-B-A designator to the Q designator of each entry operand of the waiting unit.

As an example, if register X1 was already reserved for the result of multiply 1 unit and the Add unit wants to use X1 as its j entry operand, then the Q_j designator of the add unit will be set to the unit code (06) identifying the multiply 1 unit. (See table 3-2.) If the required register

is not reserved, the respective Q designator will be set to zero and its Read flag will be set to indicate that the register is free and ready to be read. However, a unit will read its operands only when both are flagged.

Q designators are set in the following manner:

After setting the Unit Select FF, the entry operand register numbers for the requesting unit are identified from translation of the i, j, k portions of U^2. Select FFs, corresponding to that instructions entry operands (X_j, X_k, B_j, etc.), are set to identify the XBA register groups. In the case of an add unit selection, X_j and X_k FFs (H26 and F31) are set and their outputs are combined with the k and j translation from U^2 (G22, G23) to identify and read the respective X designators (H01-08). This information is sent to the Q designators (I17, I18) of the add unit under control of the add unit select FF.

Q Designators for Functional Units

Q (octal)	Functional Unit
00	Branch
01	Increment 1
02	Increment 2
03	Shift
04	Boolean
05	Divide
06	Multiply 1
07	Multiply 2
10	---
11	Read Memory Channel 1 (X_1)
12	Read Memory Channel 2 (X_2)
13	Read Memory Channel 3 (X_3)
14	Read Memory Channel 4 (X_4)
15	Read Memory Channel 5 (X_5)
16	Long Add
17	Add

Table 3-2

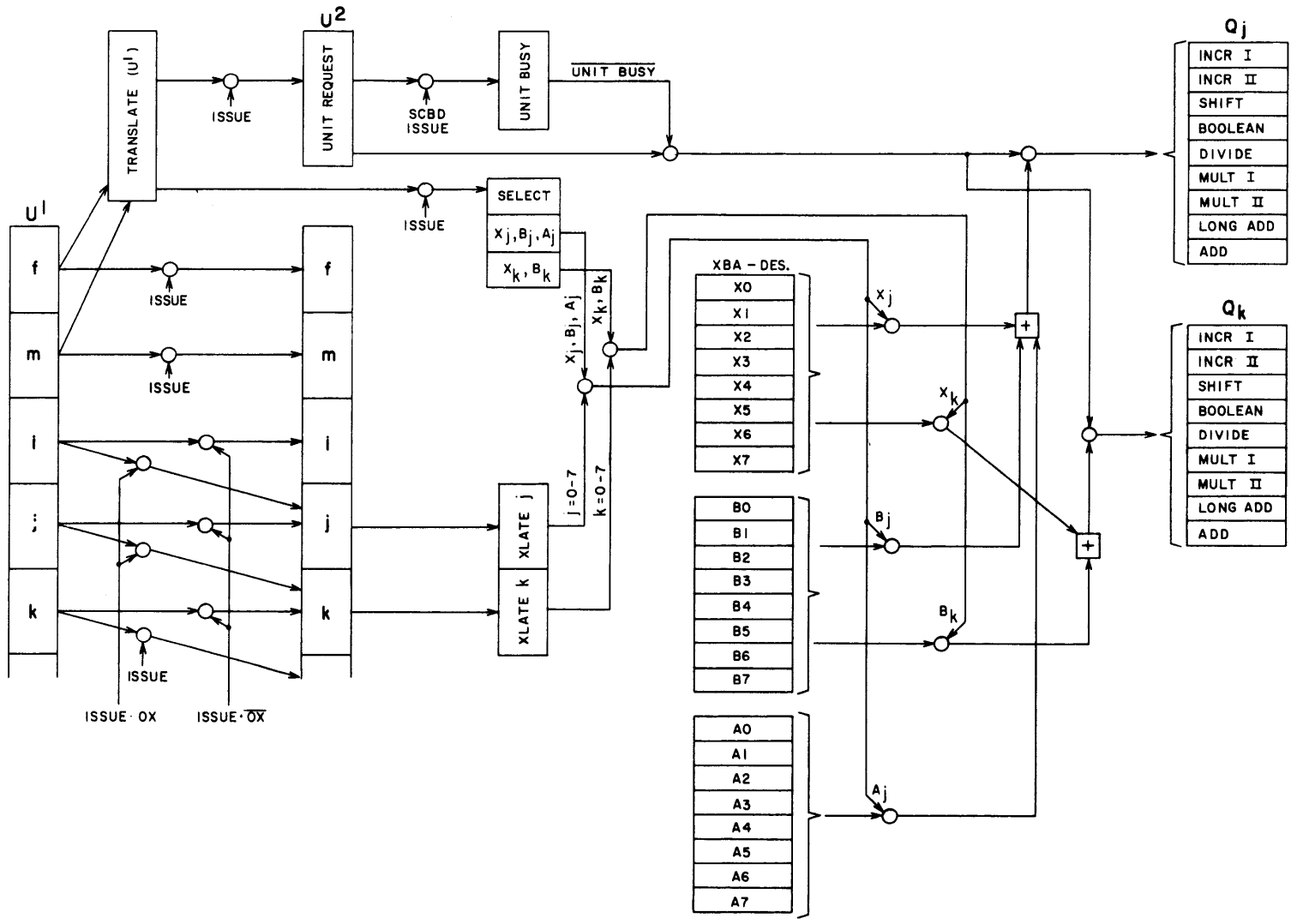

U2
TRANSLATE (U1)
ISSUE
UNIT REQUEST
SCBD ISSUE
UNIT BUSY
UNIT BUSY
Qj
INCR I
INCR II
SHIFT
BOOLEAN
DIVIDE
MULT I
MULT II
LONG ADD
ADD
U1
ISSUE
SELECT
Xj, Bj, Aj
Xk, Bk
f
m
i
j
k
XBA - DES.
X0
X1
X2
X3
X4
X5
X6
X7
B0
B1
B2
B3
B4
B5
B6
B7
A0
A1
A2
A3
A4
A5
A6
A7
Xj
Xk
Bj
Bk
Aj
+
Qk
INCR I
INCR II
SHIFT
BOOLEAN
DIVIDE
MULT I
MULT II
LONG ADD
ADD
XLATE j
XLATE k
j=0-7
k=0-7
ISSUE · OX
ISSUE · $\overline{OX}$

FIG. 3–4. SET Q

SET X-B-A

The set XBA step updates the XBA reservation list to include the result register of the latest or current reserving unit. In the previous example, if the add unit result register is X0, the identifying code for the add unit (octal 17) is entered in the X0 designator as the latest reservation. Should a subsequent instruction require X0 as a result register, it is not issued until the add unit releases X0. Instructions which require X_0 as an entry operand however, are issued and receive the add unit code in their respective Q designator.

Three steps are necessary to place this reservation:

1. The unit is identified
2. The register number is identified
3. The register group (XBA) is identified

The unit is identified by translating the fm portion of U^1; this will set the unit select FF. A second translation of the fm portion of U^1 identifies one of four possible result register groups: Ai, Bi, Xi, or Bj. This information is stored in separate Result FFs.

The register number is derived from translations of the j and k portions of U^2. The register number and group (XBA) are combined to gate the unit code (Table 3-2) into the XBA designator. Each unit selection is converted to a 4 bit binary code by driving four inverters in a combination equalling the binary code. Thus, the add selection drives four inverters which yield binary 1111 or octal 17; a shift selection, for example, drives only bits 0 and 1 to yield binary 0011 or octal 03. (See G33 and F40.) The shift and increment units are the only ones that use the A and B registers for results. The codes for these units use only bit positions 0 and 1. Thus, the B and A designators need be only two bits, whereas the X designators are four.

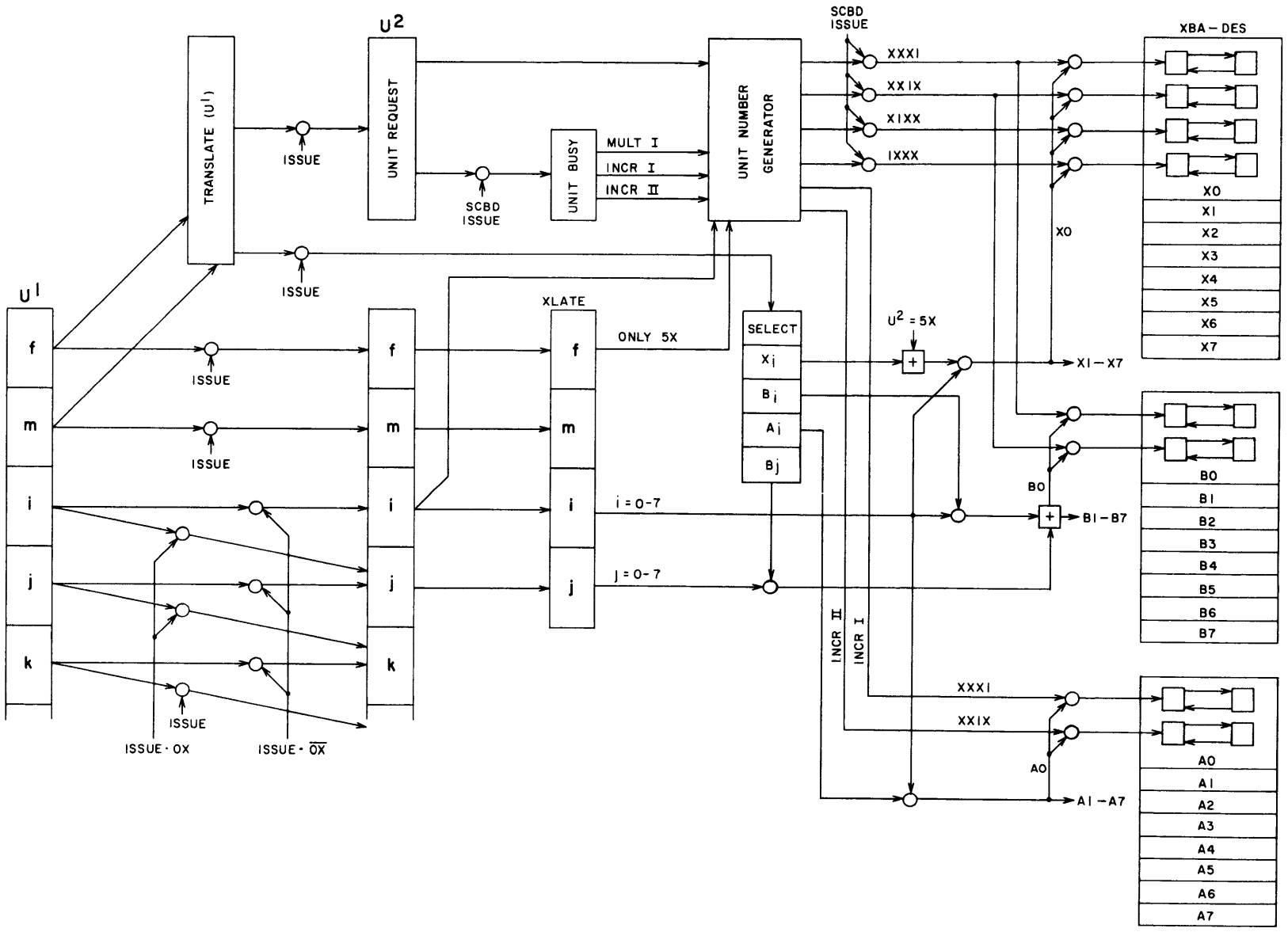


FIG. 3-5. SET X-B-A

5X instructions place their result in an A register and thereby cause a read or write operation in the corresponding X register (except for A0 and X0). In this case, both registers are reserved.

SET READ FLAGS

When the Q designators have been set, the Q translators will provide outputs for Q=0 through 17 (Q=10 not used). These octal codes define the unit that has reserved the particular register. (See Table 3-2.) For example, let Q=01. The increment 1 unit has reserved the particular register referred to by the Q designator. The Q=01 output from the Q translator will be anded with the output of the Increment 1 Release FF setting the operand read flag. In the case of Q=0 the read flag is set directly, since that particular register had not been reserved by another unit. Operands are read only when both read flags are set.

Reservation Control

A unit reads its operands, computes, and stores its result after its reservations have been placed. Scoreboard control directs the read operand and store result action. The example used to describe placing reservations is described further to illustrate control operation.

In the example, an add instruction is issued to the add unit and the necessary register reservations made. Before this, a multiply instruction was issued to the multiply 1 unit, and this unit reserved register X^1 for its result. The add unit needs X^1 for its j operand. The add unit must wait for the multiply 1 result and release of X^1 before it can read X^1. Scoreboard control requires that both operands be available before a unit proceeds to fetch them (discussed later).

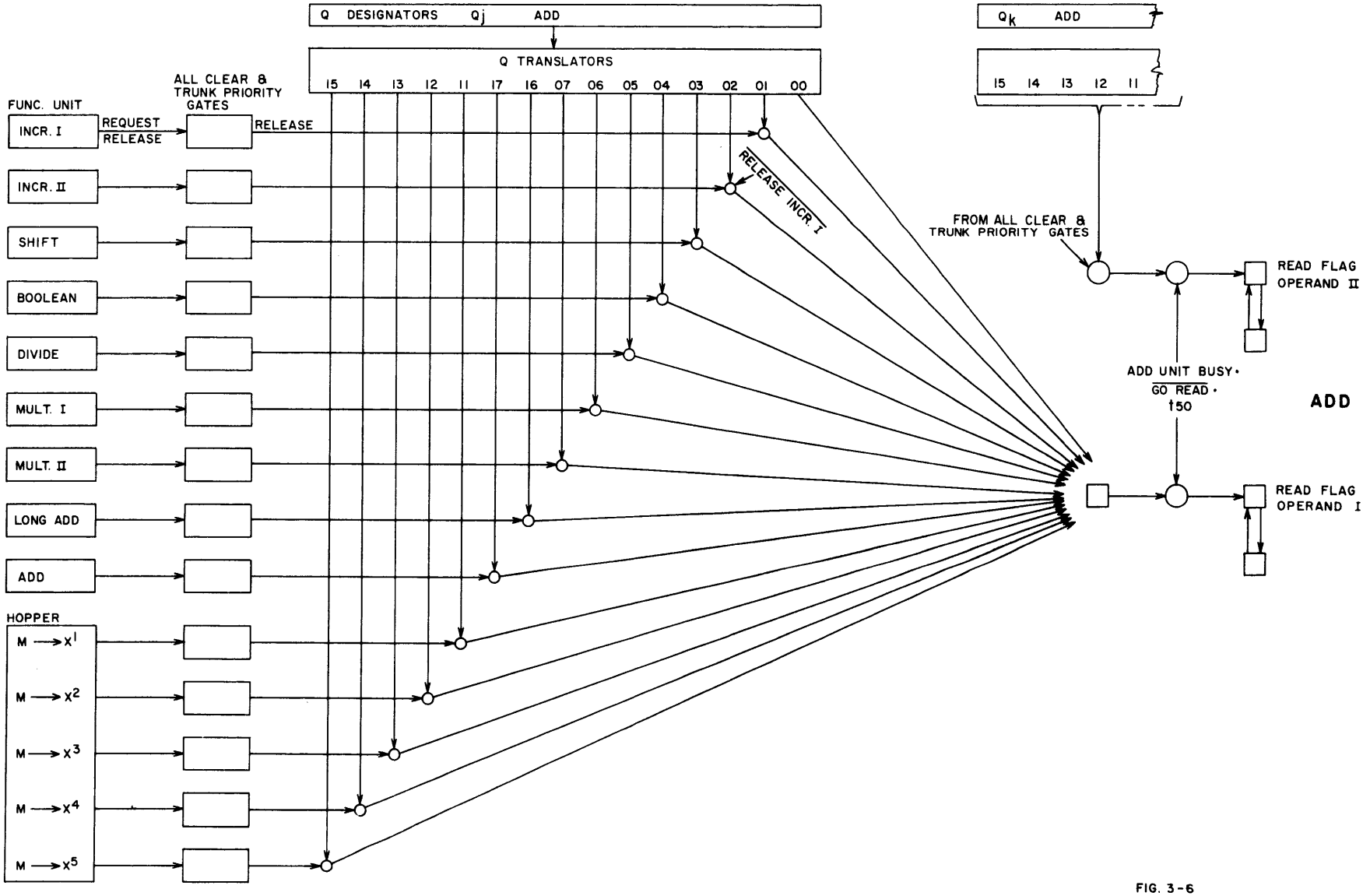


FIG. 3-6
SET READ FLAGS
ADD UNIT

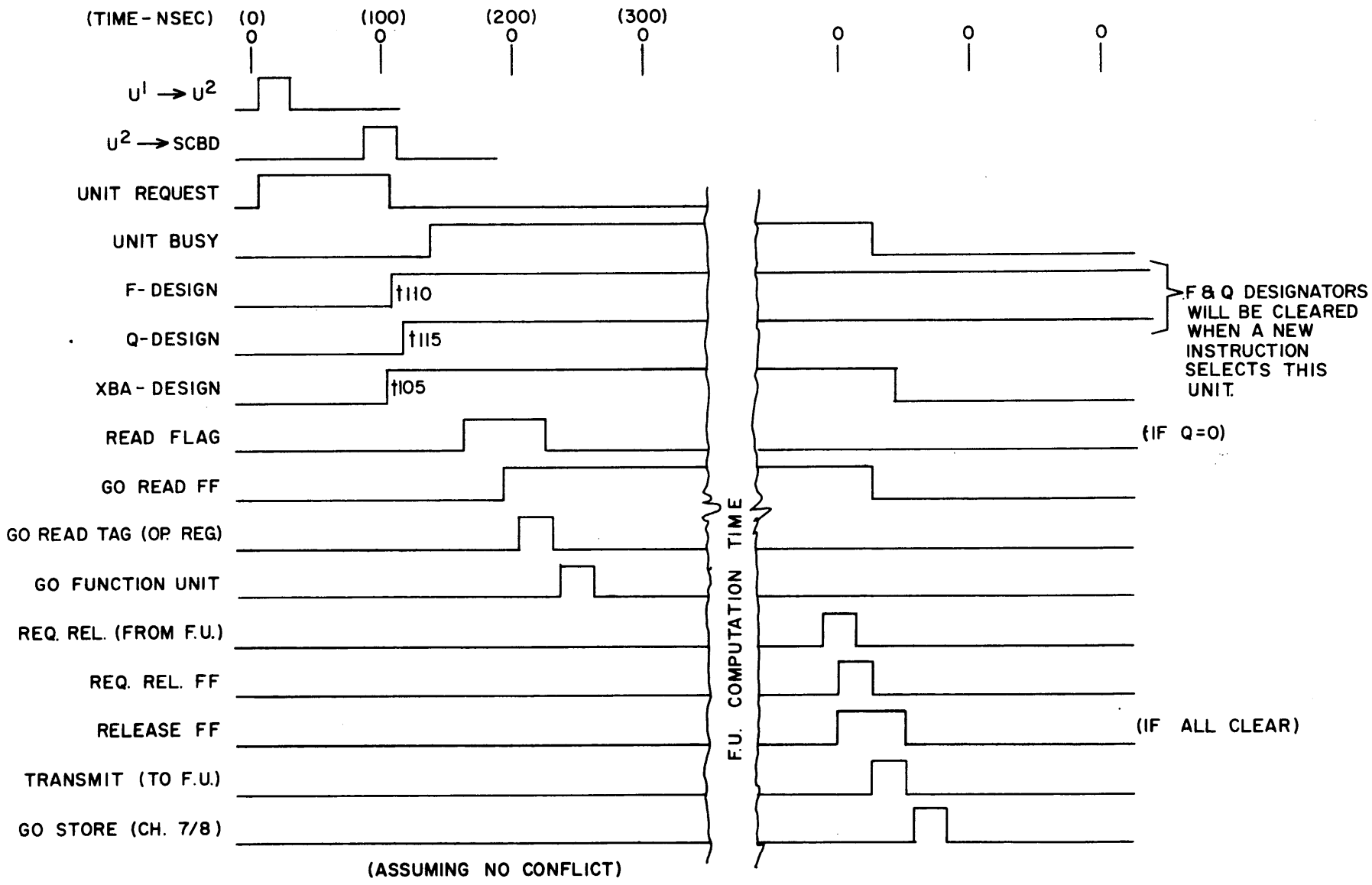


FIG. 3-7.
SCOREBOARD TIMING
(SET F, Q, XBA DESIGNATORS ECT.)

Multiply 1 unit, when ready, requests scoreboard permission to release its result to X^1. Control determines that the data trunk to X^1 is all clear and signals multiply 1 to go store its result in X^1. Control then voids all multiply 1 reservations by clearing the multiply 1 busy FF and the X^1 designator in the X-B-A reservation designators.

At the same time, all Q designators in the scoreboard are tested to determine if any units are waiting for the result being stored in X^1. For our example, the test finds the add unit j operand waiting for X^1 and sets a corresponding read flag to indicate that the j operand is now ready. The k operand has a similar flag. Assuming the k operand read flag set (because its Q=0, thus no reservation conflict), and the data trunk from the add unit to the operating registers clear, control signals the add unit to go read its operands. The add unit reads its operands, computes, and finally requests scoreboard permission to release its result to X^0. The request release sequence described earlier for multiply 1 repeats then. Note that several units may be waiting for X^1 and may read X^1 coincidently. The number of simultaneous reads of a given register relates to the assignment of data trunks from the operating registers to the units (table 3-5.)

Release requests from units arrive at scoreboard control asynchronous to other scoreboard action. Requests are immediately recognized and processed in parallel with each other and other scoreboard action. As noted, units share a number of data trunks connecting them to the 24 operating registers. A priority scheme on the data trunks regulates their use and may produce a small delay in processing go store and go read commands. Otherwise, parallel processing is continuous as long as instructions are issued.

Circuit discussions begin with the unit request release command and continue with the scoreboard go store and go read commands and the circuits which control their production.

Request Release. The request release from a unit indicates the unit wants to release its result to an operating register. The command is sent several minor cycles before the result is actually ready; the premature command allows scoreboard control to process the request and determine the next in-line user(s). Timing permits a unit to store its result and the next user(s) to read the result at a slightly later time. The overall timing scheme from request release, to a store and then a read on a register, is such that no time elapses from the end of one units computation and the start of a unit which is waiting for the result.

Request release signals are stored in separate FFs to allow parallel processing. Each request release is transferred through an all clear gate and a trunk priority gate to a release FF. If these gates can be made, the Release FF will be set, sending a command to the unit to transmit its result. The next user is also determined at this time and told to go read its operands.

All Clear. The all clear gate gives all units (which have been issued instructions) the opportunity to read a register before another unit stores its result in the register. This read-before-store condition arises because one or more instructions which call for reading a given register may be issued before an instruction which calls for storing in the same register. However, varying execution times of the units (and other factors, for example trunk priority) permit the unit executing the last instruction issued in a series to be ready to store its result before previously issued instructions have finished reading the register. The all clear gate resolves this conflict.

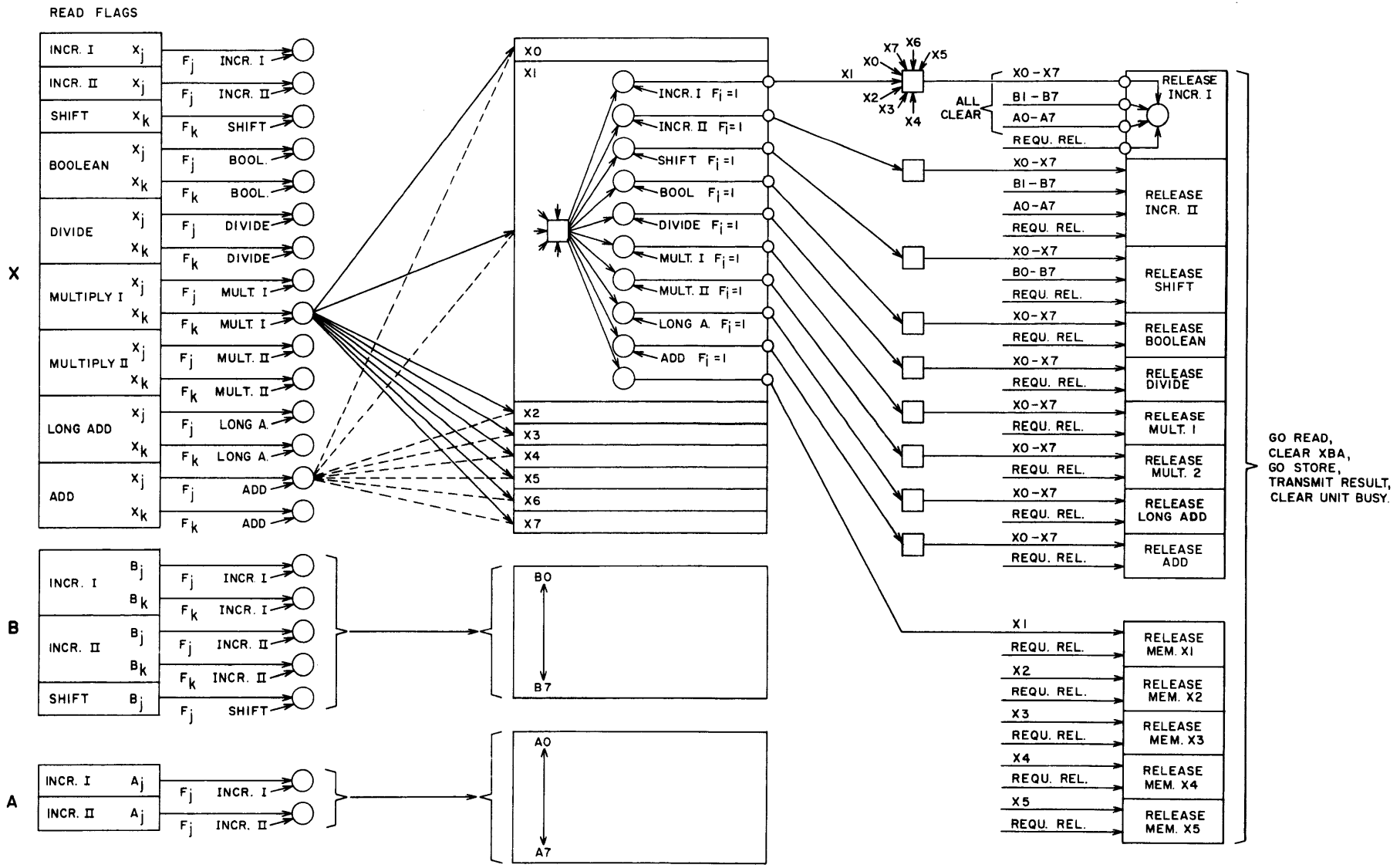


FIG. 3-8. ALL CLEAR

Earlier discussion noted that each entry operand for a unit had a read flag which, when set, indicated that the register associated with the operand was ready to be read. Both entry operand registers must be ready and their flags set before reading starts (flag set conditions described later). Hence, one set flag for a unit indicates that the unit is waiting for its other operand register to become available; both flags set indicate reading in process. Either case voids the all clear condition and delays release.

The all clear condition tests read flags against the register receiving a result. Some units, like the add, communicate only with X registers and in these cases read flags related to A or B registers are not tested. However, the increment units may communicate with X, B, or A registers so that all read flags must be tested. Table 3-4 lists units and read flags tested for all clear conditions.

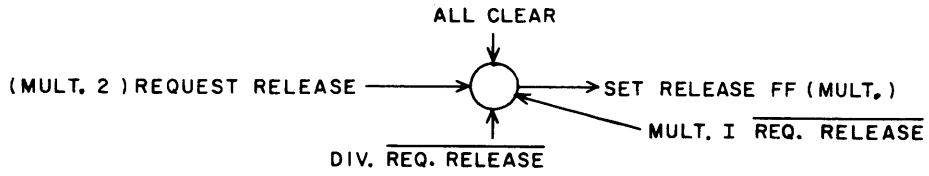
FUNCTION UNIT	READ FLAGS		
	X	B	A
ADD	X		
LONG ADD	X		
MULTIPLY 1	X		
MULTIPLY 2	X		
DIVIDE	X		
BOOLEAN	X		
SHIFT	X	X	
INCREMENT 1	X	X	X
INCREMENT 2	X	X	X

Table 3-4

Data Trunk Priority

a) Go Store Priority

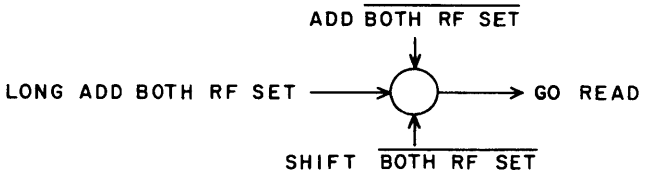
After the Request Release FF of a given unit is set, an AND gate with All Clear, Trunk Priority and Request Release input will allow the Unit Release FF to be set, sending out a GO Store signal. e.g. the AND gate of the Multiply 2 Request Release looks like this:



This means that the Multiply 2 unit (Priority 3) can only be released if Divide unit and Multiply I unit are not requesting release. An exception is the Increment I and II units which have the priority gate at the output of the Release FF (I32); see Set Read Flags block diagram.

b) Go Read Priority

The outputs of the Read Flags of a given unit are anded together with the output of the Read Flags of another unit e.g.



DATA TRUNK PRIORITIES

GO STORE	GO READ	
1. ADD	1. SHIFT	TRUNK 1
2. SHIFT	2. ADD	
3. LONG ADD	3. LONG ADD	
1. DIVIDE	1. BOOLEAN	TRUNK 2
2. MULTIPLY 1	2. DIVIDE	
3. MULTIPLY 2	3. MULTIPLY 1	
4. BOOLEAN	4. MULTIPLY 2	
1. INCREMENT 1	1. INCREMENT 1	TRUNK 3
2. INCREMENT 2	2. INCREMENT 2	

TABLE 3-5

GO Store/GO Read

The following discussion will refer to the attached block diagram. Let us assume that Function Unit X is sending a Request Release setting its Request Release FF. If the All clear and Go Store Priority is made, the Unit Release FF will be set.

The outputs of the Unit Release FF will do the following:

1. Clear Request Release FF
2. Clear Unit Busy FF
3. Enable the F_i (Result) designator to go and clear its respective XBA designator.
4. A go store signal enables the contents of the F_i register into the Entry Control specifying the register into which the result of the releasing unit must be stored.
   Another Go store signal is sent directly to the Entry control to allow it to recognize the F_i input.
5. A transmit signal is sent to the Function Unit telling it to release its result.

Once the releasing unit has stored its result it is free to start a new computation. If there are units waiting to read this result register after the result is stored, their Q designator(s) will contain the unit code of the releasing unit.

These Q designators would then set their respective Read Flags. If both Read Flags are now set for the waiting unit its go read priority is checked. If there is more than one waiting unit with all its read flags set, the priority gate will allow the unit with highest priority to go and read its operands as follows:

a) Clear Read Flags

b) Set Go Read FF to inhibit entrance to the read flags during the read sequence.

c) Send an enable to the outputs of the F_j and Fk designators to transfer the contents to the Exit Control of the Operating Regs.

d) Send a Go Read to the Exit Control to allow it to recognize the register numbers being sent by the F designators.

e) Send a Go F.U. to allow the unit to accept the incoming operands.

Waiting units can therefore be identified by the set state of their Busy FF and a cleared Go Read FF.

Note that the time elapse from a unit's request release to placing the result on the trunk may be within a minor cycle when the all clear is present at request release time and trunk priority favors the requesting unit.

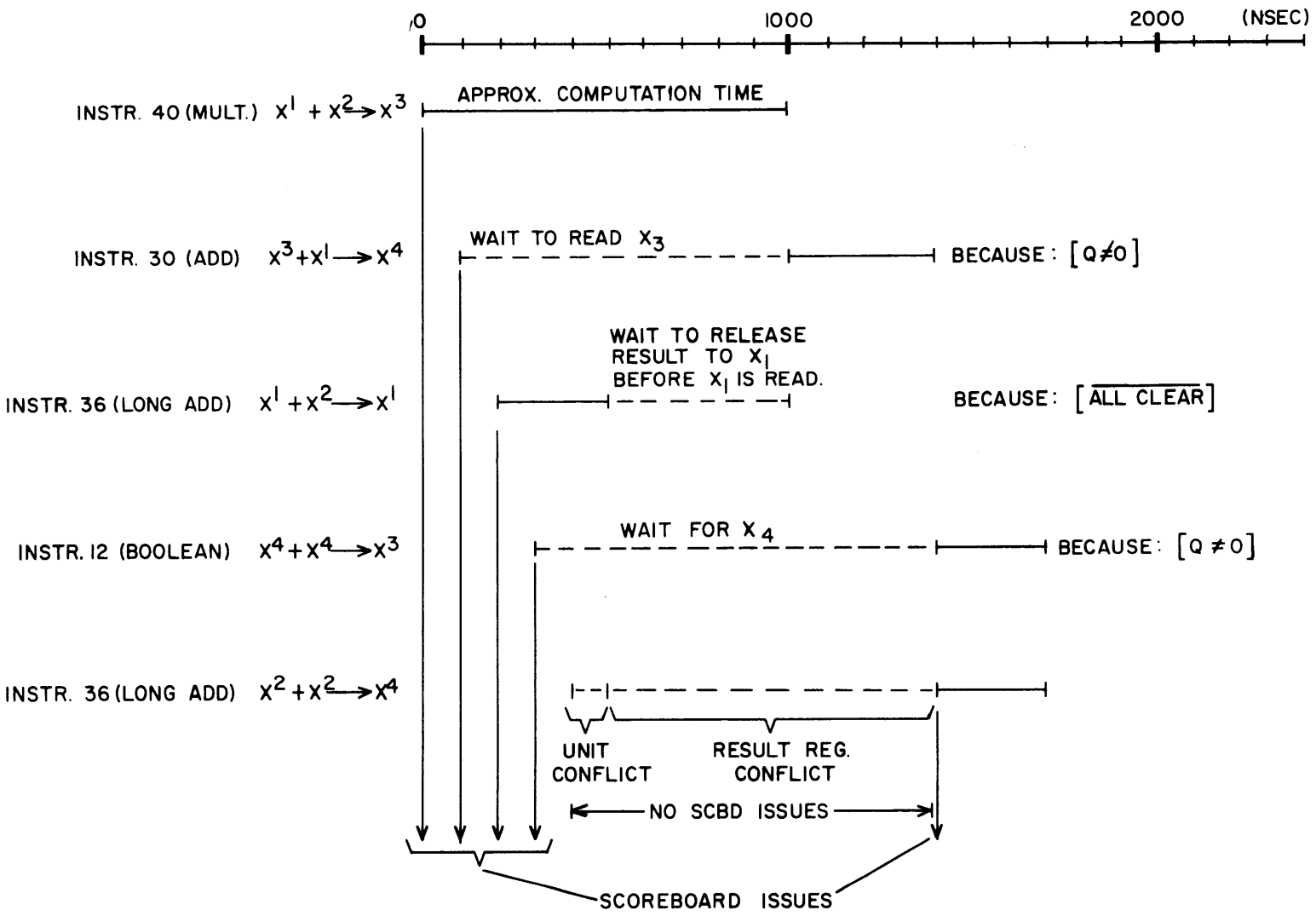


FIG. 3-9.
APPROX. INSTRUCTION EXECUTION TIMING

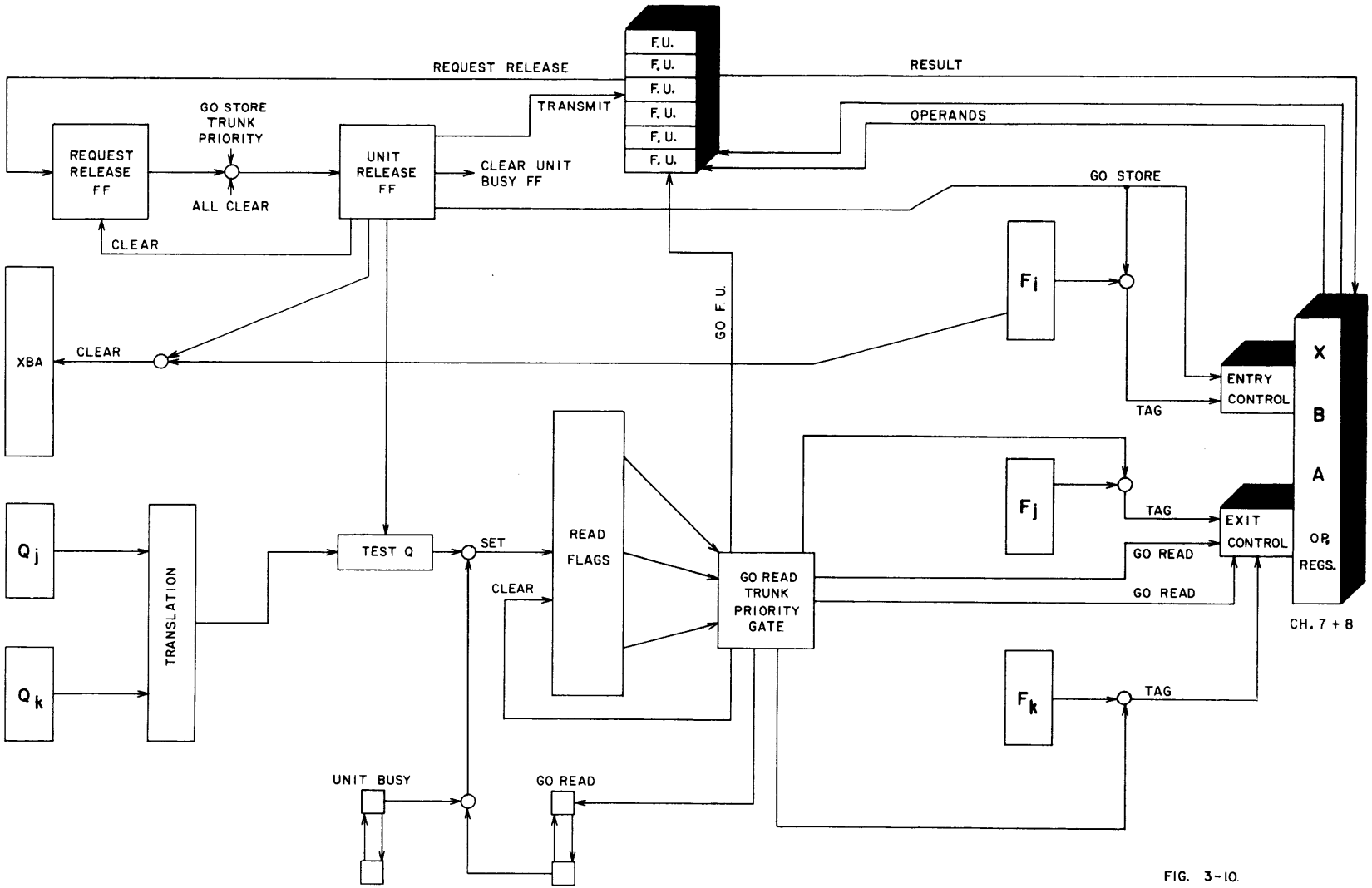


FIG. 3-10.
REQUEST RELEASE
BLOCK DIAGRAM

Entry Control

a) For X Regs:

The data from the different F.U.s and Central Memory are received by the Input Registers on Chassis 7 and 8 (e.g. Ch. 7: A37, B37, C37, H37, etc. Ch. 8: A01-08, B01-05, D01, etc.)

The "Go Store" signal will enable the contents of the F designator into the Translator of the Entry Control to decide into which register the data is to be stored. For the Increment Unit the Input Reg only receives 18 bits, therefore the 2^{17} bit (Sign Bit) is used to fill up the bit locations 2^{18}-2^{59} in the X regs. (Sign Extension). For those instructions that need a memory reference, the address tags are taken from the Hopper and translated to gate the contents of the Input Reg into one of five Buffer Regs D1-D5 corresponding to X_1-X_5. Another translation of these tags, together with an Accept will allow the setting of one of the Reg. Release FF for Memory Ch. 1-5. Once the All Clear gate is made, the Release FF for the selected Memory Channel will be set, allowing the setting of the corresponding D-X FF. The data is now gated from the Buffer Regs into the X operating register. Bits 2^0-2^{35} of the X regs are on Ch. 7, the 2^{36}-2^{59} on Ch. 8.

For an Exch. Jump, the tags from the Exch. Tag Control and the Exch. Tag (1 bit) from the Hopper (M2) together with an accept will gate the data from memory into its resp. X reg.

b) For A, B regs:

The same applies here that has been said about the X regs (see diagram) except that we only have 18 bit operating registers A and B.

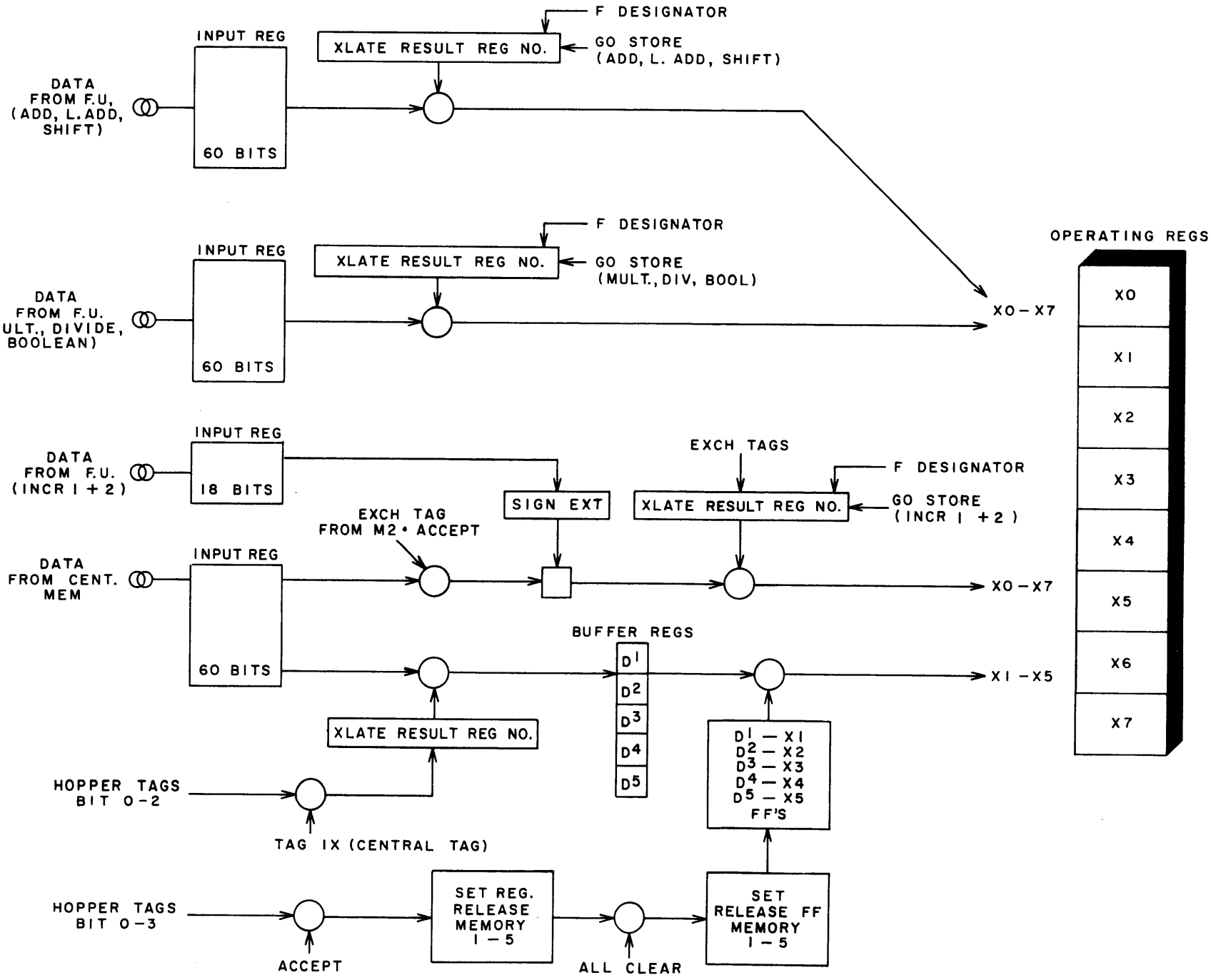


Fig. 3-11. Entry Control - X Regs.

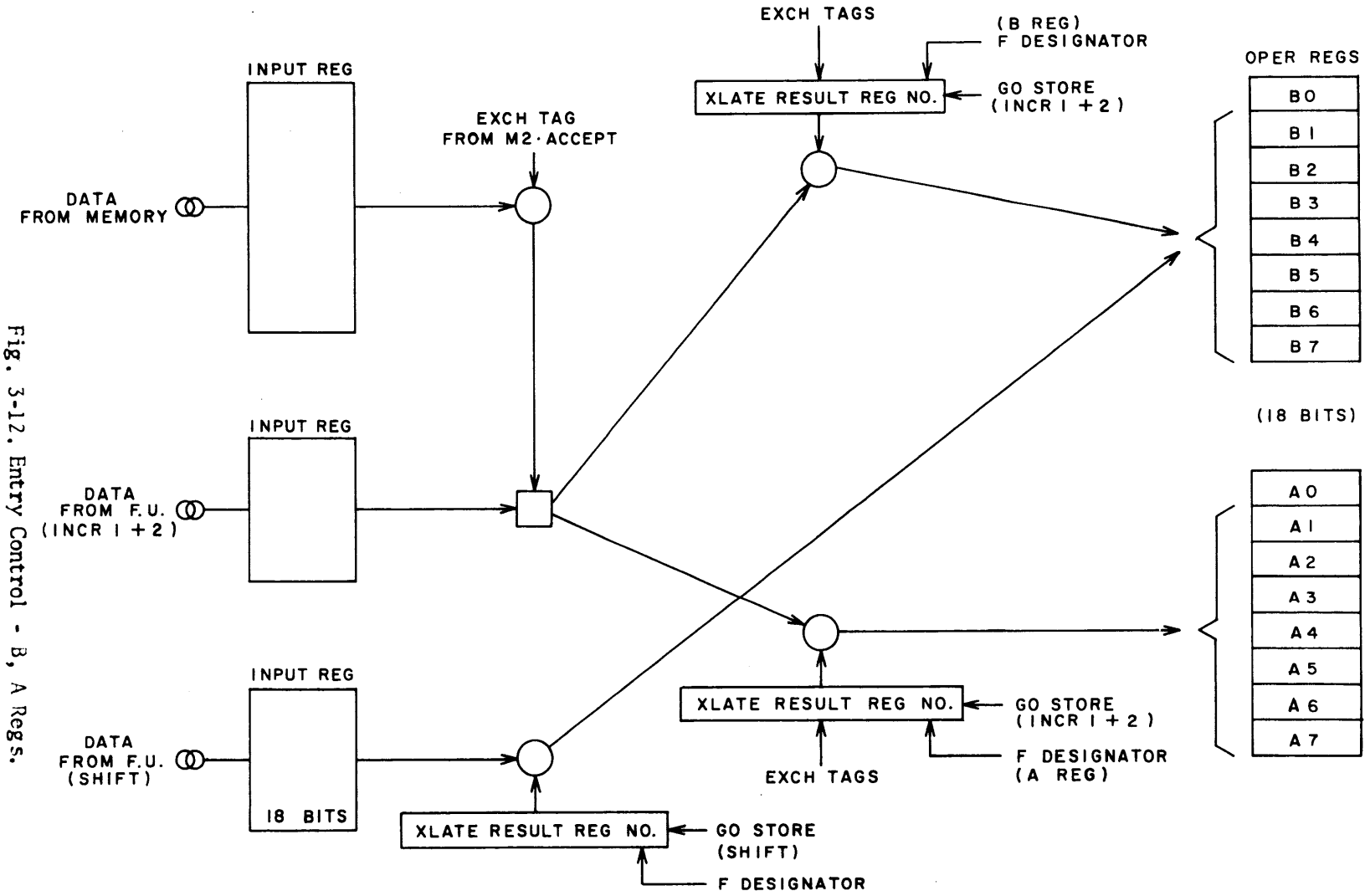


Fig. 3-12. Entry Control - B, A Regs.

EXIT CONTROL

The "Go Read" signal from the Test Trunk Priority network will enable the contents of the F_j- and F_k-designators into the Translators of the Exit Control. This specifies the transfer from a particular X, B or A operating register to a particular functional unit. The $F_{j,k}$-designators are combined into 9 different groups. These can be seen in the block diagram of the Exit Control.

The $F_{j,k}$-designators from the same group will use the same Network to gate the contents of the XBA Operating Register into the F.U. However, in every group there is also a "go" bit from the Test Trunk Priority network. These bits will help gate the data to the proper part of the Output Network.

The Output Network and the data trunks to the functional units are grouped to correspond generally with the $F_{j,k}$ designator groupings (e.g., Operands 1 of Divide, Multiply 1, Multiply 2 and Boolean use the same data trunk).

During an Exchange Jump, the contents of the XBA Registers are transmitted to Central Memory. The tag that controls this transfer comes from the Hopper M4 of the Stunt Box. Only the first 4 bits of the Hopper tag are sent to the Exit Control network.

The 3 lower bits are translated to determine which of the 7 X, B or A Registers will be gated to Central Memory. When the 2^3 bit is zero (6X) the contents of the B-Operating Registers are gated into the lower 18 bits of the Output network going to the Store Distributer. Simultaneously the contents of the A-Operating Registers are gated into bit positions 2^{18}-2^{35} of the Output Network.

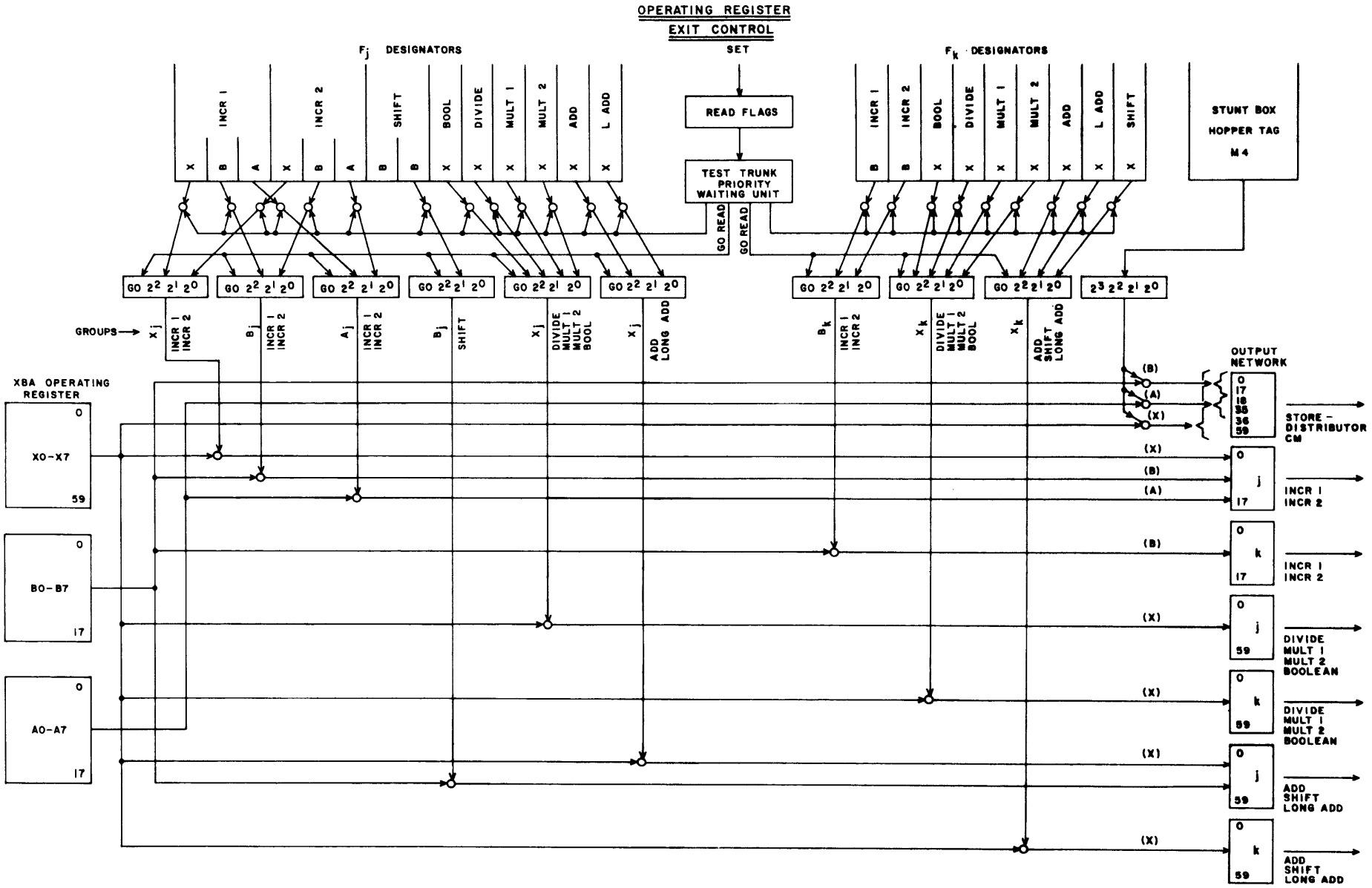

OPERATING REGISTER EXIT CONTROL
SET
READ FLAGS
TEST TRUNK PRIORITY WAITING UNIT
GO READ
GO READ
Fj DESIGNATORS
Fk DESIGNATORS
STUNT BOX HOPPER TAG M4
INCR 1
INCR 2
SHIFT
BOOL
DIVIDE
MULT 1
MULT 2
ADD
L ADD
GROUPS
XBA OPERATING REGISTER
X0-X7
B0-B7
A0-A7
OUTPUT NETWORK
STORE - DISTRIBUTOR CM
INCR 1 INCR 2
DIVIDE MULT 1 MULT 2 BOOLEAN
ADD SHIFT LONG ADD

FIG. 3-13

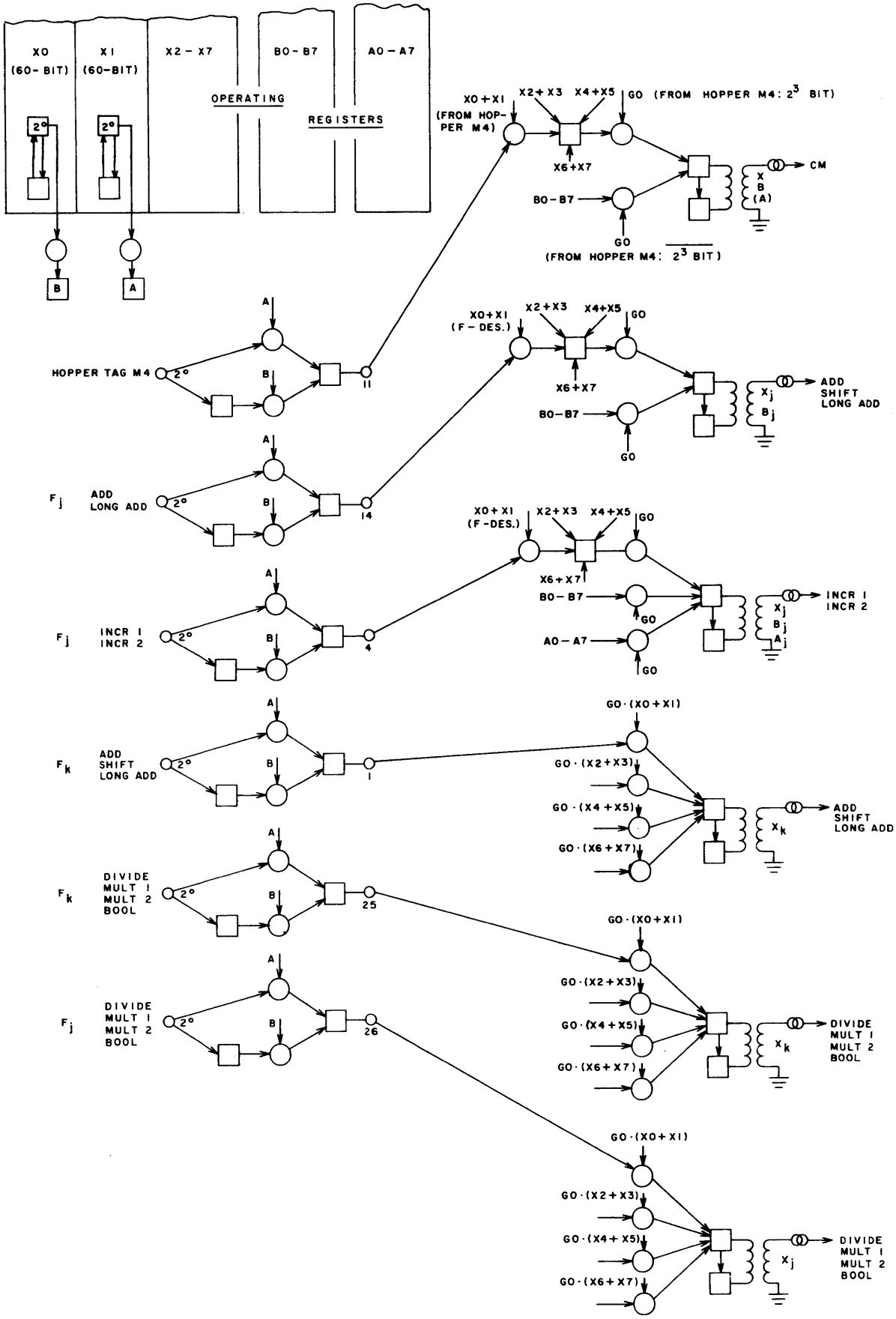


FIG. 3-4
OPERATING REG. EXIT CONTROL

When the 2^3 bit is one (7X) the contents of the X-Registers will be gated to Central Memory via the Output network.

For a Central Processor Write (X6→CM or 7X→CM) a similar situation exits to that described previously for Exchange Jump. The Hopper tag (56 or 57) will gate the contents of Operating Register X6 or X7 to Central Memory via the Output network.

Note: The B and A registers are located on chassis 7 along with bits 0-35 of the X registers. Bits 36-59 of the X registers are located on chassis 8. Therefore, all tags and designators must be sent to both chassis 7 and 8.

DATA TRUNKS

Four Data Trunks (shown in the block diagram) transmit the 2 Operands from the output network of the XBA-Operating Register to the proper functional unit. Data is also transferred on trunks to Central Memory. Four other Data Trunks transmit the result from the functional trunk and data from Central Memory to the Input Register of XBA-Operating register.

Data Trunk 1 connects the output network of the XBA-Register and the Long Add, Add and Shift functional units. There are 60 bits of coax. for transmitting Operand 1 and 60 bits of coax. for Operand 2.

Data Trunk 2 connects the output network of the XBA-Register and the Divide, Boolean, Multiply 1 and Multiply 2 functional units. The operands go via chassis 6 (Multiply 1 and 2) to chassis 2 (Divide and Boolean).

The results coming from the functional units and the data coming from Central Memory will be transmitted by Data Trunks 1', 2', 3', and 4' to the Input Register of the Entry Control network.

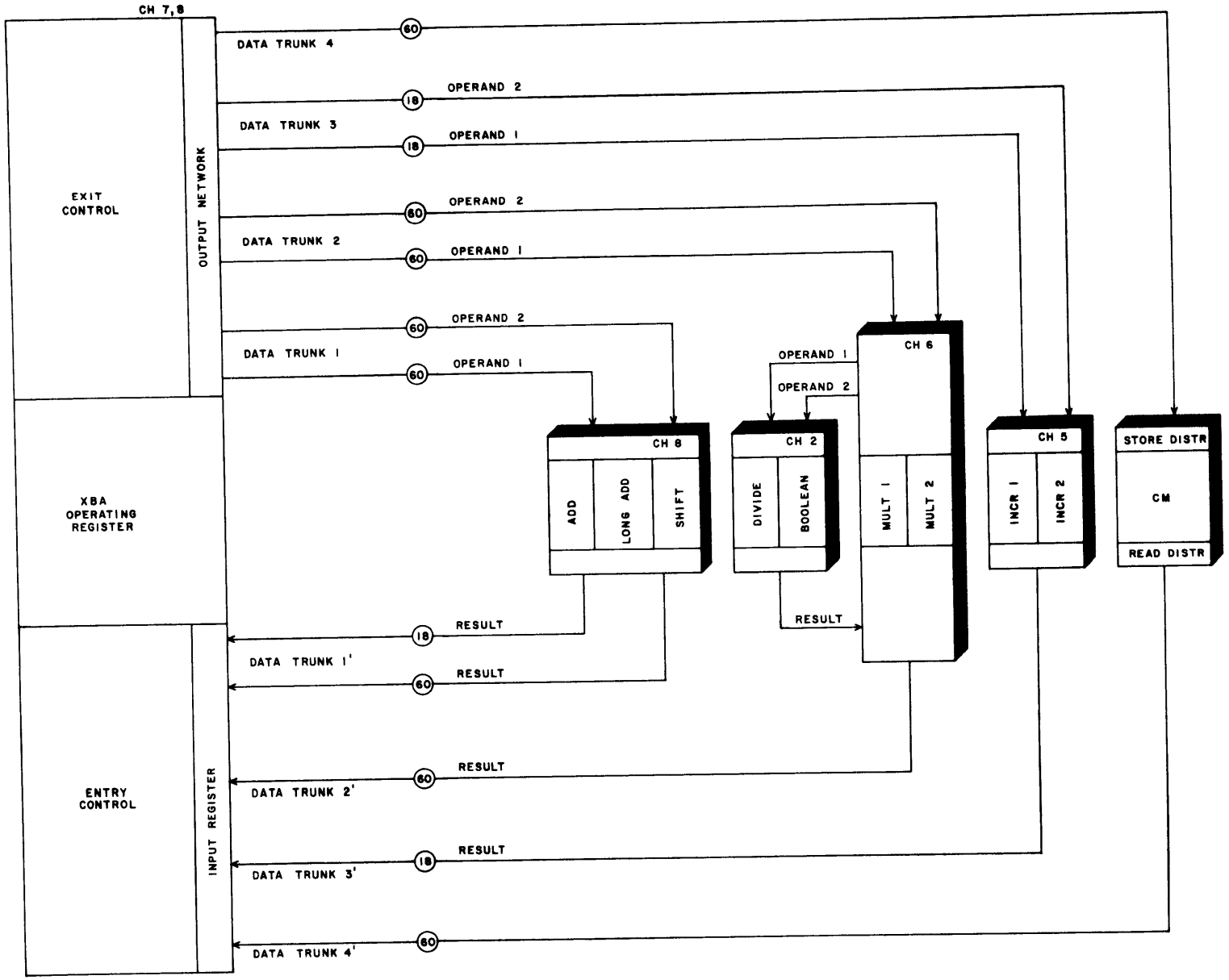


FIG. 3-5
DATA TRUNKS

6600 CPU TIMING NOTES

1. The times given in the reference manual are computational times - the time needed after the execution start until the result is computed and ready to be stored into the result register. Times are given in minor cycles (1 minor cycle = 100 nanoseconds).

2. The functional units are not freed until one minor cycle after the result has been stored into the result register.

3. A result register value may be used as an operand to another instruction as soon as the result has been stored into the register (same minor cycle). This result register will not be freed to be used as a result register of another instruction until one cycle after the result has been stored into that register. (No trunk priority considered)

4. Instructions are issued to the functional units if:

    a. The word containing the instruction is in the stack
    b. The functional unit(s) needed are free, and
    c. The result register(s) needed are free (note table 1)

    If these three conditions are not met, all further instruction issues are held until they are satisfied. Each issued 15-bit instruction requires one minor cycle before the next instruction is available for issue. Each issued 30-bit instruction requires two minor cycles before the next instruction is available for issue.

5. Execution within a functional unit does not start until the operand(s) are available (note table 1). The two operands required are fetched from the registers at the same time (one operand is not loaded while the unit waits for a second operand).

6. In instructions 02-07, where more than one functional unit is used, the instruction is not <u>issued</u> until both functional units involved are free.

7. Times given for instructions (01-07 and 50-57 do not consider any memory conflict conditions.

8. In instructions 50-57, if i = 1, 2...5 (load from memory instructions), the Xi register value is not available until 8 minor cycles after the start of the instruction execution (assuming no memory conflicts). When two load instructions begin execution one minor cycle apart, one extra minor cycle is required for execution of the later instruction. Therefore, the second executed instruction would require 9 cycles for the load, 5 cycles for the increment unit, and 4 cycles for the A register.

9. In instructions 50-57, if i = 6 or 7 (store to memory instructions), the Xi register is not available for a result register until 10 minor cycles after the instruction begins execution (assuming no memory conflicts). When two store instructions begin execution one minor cycle apart, one extra minor cycle is required for execution of the later instruction. Therefore, the second executed instruction would require 11 cycles for the store, 5 cycles for the increment unit, and 4 cycles for the A register. A store instruction does not check the X register before being issued. The X register is available as an entry operand register while the store is taking place.

10. When executing sequential instructions that are not in the stack, the minimum time is one word of instructions every 8 cycles. The time of issue of the last parcel of an instruction word to the time of issue of the first parcel of the next instruction word (while executing sequential instructions that are not in the stack) requires a minimum of 4 cycles.

* 11. All 03 branches made within the stack require 9 minor cycles. An 03 branch to the next sequential word is recognized as a branch within the stack and requires 9 minor cycles.

** 12. When an 04 branch out of the stack is taken, 15 minor cycles are required (considering no memory conflict) from the start of the branch instruction execution to the availability of the branched-to word instruction to a functional unit (instruction ready for issue.)

13. Eleven cycles are required for 03ijK instructions when the branch is not taken (time from branch execution to issue of next instruction). This time is the same for a fall-through branch in the same word, or a fall-through branch to the next word.

14. Ten cycles are required for 04ijK-07ijK instructions when the branch is not taken (time from branch execution to issue of next instruction). This time is the same for a fall-through branch in the same word, or a fall-through branch to the next word.

15. The B0 register is handled as any other Bi register for timing purposes (i.e., B0 will hold up execution of an instruction if it is a result register of a previous non-completed instruction, etc.).

16. Neither increment unit may be involved in a load operation if a store operation is to be issued, and neither increment unit may be involved in a store operation if a load operation is to be issued. The sequential loading of instruction words does not affect the load/store conditions of the increment units.

* 03 instruction requires 8 cycles

** 04 instruction requires 14 cycles

17. The operand registers are available to more than one functional unit in the same minor cycles if the units are in different groups.

Group 1	Group 2	Group 3
Divide	Shift	Branch
Boolean		
Multiply 1	Floating Add	Increment 1
Multiply 2	Long Add	Increment 2

18. The time needed for a functional unit to operate on indefinite, out-of-range, or zero values is the same as for normal, in-range values (i.e., no gain or loss in execution time due to a unit recognizing an indefinite operand and setting an indefinite result).

19. An index jump instruction (02) will always destroy the stack. If an unconditional jump back in the stack is desired, an 0400K instruction should be used (to save memory access time for instructions).

20. A return jump instruction (01) will always destroy the stack.

21. After a result has been computed by a functional unit, the result register is checked to see if it has been reserved as an operand register (for a previously issued instruction). This is done so that a result will not overlay an operand to a previously issued instruction. If a unit (#1) is waiting for an operand to be fetched by another unit (#2) before storing its result, for timing considerations

    a. The result register is available to a third unit (#3) as an operand, the cycle following the fetch, and

    b. The register may be available as a result register two cycles following the fetch, and

    c. The unit is freed two cycles following the fetch.

Table 3-6

6600 Register Reservation Control

INSTRUCTION	Result Register (ISSUE)	Operand Register (EXECUTION)
Branch Unit		
02ijK	-	Bi & Bj
03ijK	-	Xi & Xj
04ijK - 07ijK	-	Bi & Bj
Boolean Unit		
10ijk - 17ijk	Xi	Xj & Xk
Shift Unit		
20ijk - 23ijk	Xi	Bj & Xk
24ijk - 26ijk	Xi & Bj	Bj & Xk
27ijk & 43ijk	Xi	Bj & Xk
Add Unit (Floating)		
30ijk - 35ijk	Xi	Xj & Xk
Long Add (Integer)		
36ijk - 37ijk	Xi	Xj & Xk
Multiply (2 Units)		
40ijk - 42ijk	Xi	Xj & Xk
Divide Unit		
44ijk - 47ijk	Xi	Xj & Xk
Increment (2 units)		
50ijK	Ai & Xi *	Aj & Bo
51ijK	Ai & Xi *	Bj & Bo
52ijK	Ai & Xi *	Xj & Bo
53ijk	Ai & Xi *	Xj & Bk
54ijk & 55ijk	Ai & Xi *	Aj & Bk
56ijk & 57ijk	Ai & Xi *	Bj & Bk
60ijK	Bi	Aj & Bo
61ijK	Bi	Bj & Bo
62ijK	Bi	Xj & Bo
63ijk	Bi	Xj & Bk
64ijk & 65ijk	Bi	Aj & Bk
66ijk & 67ijk	Bi	Bj & Bk
70ijK	Xi	Aj & Bo
71ijK	Xi	Bj & Bo
72ijK	Xi	Xj & Bo
73ijk	Xi	Xj & Bk
74ijk & 75ijk	Xi	Aj & Bk
76ijk & 77ijk	Xi	Bj & Bk

Table 3-7

		Wait Issue For Unit & Result Reg.	Wait Start Of Unit For Operand (READ) Reg & Trunk Priority
01	Return Jump (None)		
02	Go to K+ Bi		Bj & Bi
030	Go to K if Xj = 0		Xj & Xi
031	Go to K if Xj ≠ 0		Xj & Xi
032	Go to K if Xj pos.		Xj & Xi
033	Go to K if Xj neg.		Xj & Xi
034	Go to K if Xj in range		Xj & Xi
035	Go to K if Xj out of range		Xj & Xi
036	Go to K if Xj definite		Xj & Xi
037	Go to K if Xj indefinite		Xj & Xi
04	Go to K if Bi = Bj		Bj & Bi
05	Go to K if Bi ≠ Bj		Bj & Bi
06	Go to K if Bi ≧ Bj		Bj & Bi
07	Go to K if Bi < Bj		Bj & Bi
10	Transmit Xj to Xi	Xi	Xj & Xk
14	Transmit comp. Xk to Xi	Xi	Xj & Xk
20	Shift Xi left jk	Xi	Xk & Bj
21	Shift Xi right jk	Xi	Xk & Bj
43	Mask Xi by jk	Xi	Xk & Bj
47	Sum 1's Xk to Xi	Xi	Xk & Xj
50	Sum Aj & K to Ai	Ai & Xi	Aj & Bo
51	Sum Bj & K to Ai	Ai & Xi	Bj & Bo
52	Sum Xj & K to Ai	Ai & Xi	Xj & Bo
53	Sum Xj & Bk to Ai	Ai & Xi	Xj & Bk
54	Sum Aj & Bk to Ai	Ai & Xi	Aj & Bk
55	Diff. Aj & Bk to Ai	Ai & Xi	Aj & Bk
56	Sum Bj & Bk to Ai	Ai & Xi	Bj & Bk
57	Diff. Bj & Bk to Ai	Ai & Xi	Bj & Bk

SECTION IV

Central Memory

CENTRAL MEMORY

The 131K Central Memory can be accessed by the Central Processor and all peripheral processors at a maximum rate of once every 100 nsec. This is the maximum issue rate of the Stunt Box, which collects and accepts addresses from several sources on a priority basis.

The duration of the memory cycle for each memory reference is 1000 nsec., or one major cycle, and consists of a read and write cycle which is controlled by the Storage Sequence Control circuitry.

In a Central Memory Read, the contents of the specified address are read out of memory and all cores in this location switched to 'C'. The contents are then routed to the Data Distributor and the Restoration Register. The Restoration Register allows data that has been read to be put back into memory during the write part of the cycle. For a Write operation, the current contents of the specified address is read out of memory, but is destroyed in the Data Distributor. This data is not sent to the Restoration Register. Instead, the data to be written is accepted from the Data Distributor and is written into memory, as before, in the write part of the cycle.

The following discussion is divided into four distinct phases:

A. Storage Modules

B. Memory Reference Request

C. Storage Cycle

D. Data Distribution

A. Storage Modules

The storage module stores 4096 12-bit words. The module uses magnetic cores and coincident current techniques in reading and storing data in the cores. The complete time (cycle time) to read or store one word is 1000 nsecs.

PRINCIPLES OF MAGNETIC CORE MEMORY

GENERAL

The memory uses the permanent magnetic properties of ferrite cores to store the bits of computer words. A magnetic core is a bistable device capable of storing a 1 or a 0 depending upon its state of remanent magnetization.

Current carrying wires pass through the cores and magnetize them in one direction or the other. The direction of current flow determines the direction of magnetization. Approximately 400 ma turns of magnetizing force are required to switch a core.

The cores are assembled in a square matrix or memory plane. Five wires pass through each core (fig.4-1).

1. A horizontal drive, or H wire
2. A vertical drive, or V wire
3. A horizontal inhibit, or I wire
4. A vertical inhibit, or I wire
5. A diagonal sense, or S wire

The coincident current switching technique is used. A core is addressed by simultaneously driving half-amplitude (200 ma) current pulses through a selected H wire and a selected V wire.

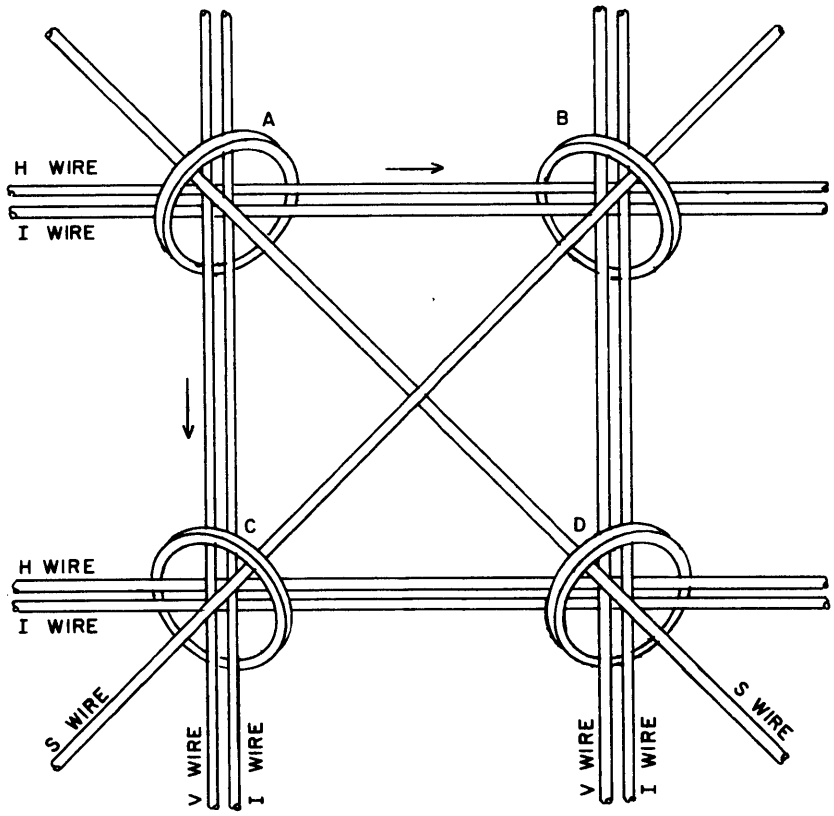


Fig. 4-1. Portion of Magnetic Core Matrix

Only one core among all those in the matrix is subjected to a magnetizing force sufficiently large to switch its magnetic state. This core is at the intersection of the selected H and V wires. All other cores in the same row or column as the selected core receive half-amplitude current pulses and are said to be half-selected. In fig. 4-1, if the left-most V wire and the top H wire carry current pulses, core A is fully selected, cores B and C are half-selected, and core D is unselected.

The polarity of residual magnetization determines the binary information stored in a core. The magnetizing force of a read current pulse on the selected H and V wires stores a 0. The absence of a pulse on the I wires permits a 1 to be stored by the write pulse.

Applying a read pulse to the selected H and V wires extracts or reads information from the selected core. If the core previously stored a 1 the pulse drives it to 0, and this change induces a pulse in the S wire. The pulse in the S wire is interpreted as a 1 bit from the core. If the core previously stored a 0 the read pulse does not affect it, and no pulse is induced in the S wire. This absence of a pulse is interpreted as a 0 bit from the core.

One memory plane holds 4096 cores in the 64 x 64 array. To use the coincident-current technique, each bit of a word is stored in a separate plane. Thus, 12 bits require 12 planes.

MAGNETIC CORES

The magnetic properties of a core are represented by its hysteresis diagram (fig. 4-2) which plots magnetic flux density (B) as a function of the field intensity (H).

If current flow sufficient to cause a field intensity of +Hm is applied to the drive lines, the flux density increases to saturation (+Bs). When the current is removed, the flux density drops to the residual positive value (+Br), which is the 0 state, and remains there. Another pulse of +Hm merely shifts the core to +Bs again, and after the pulse is removed it drops back to +Br.

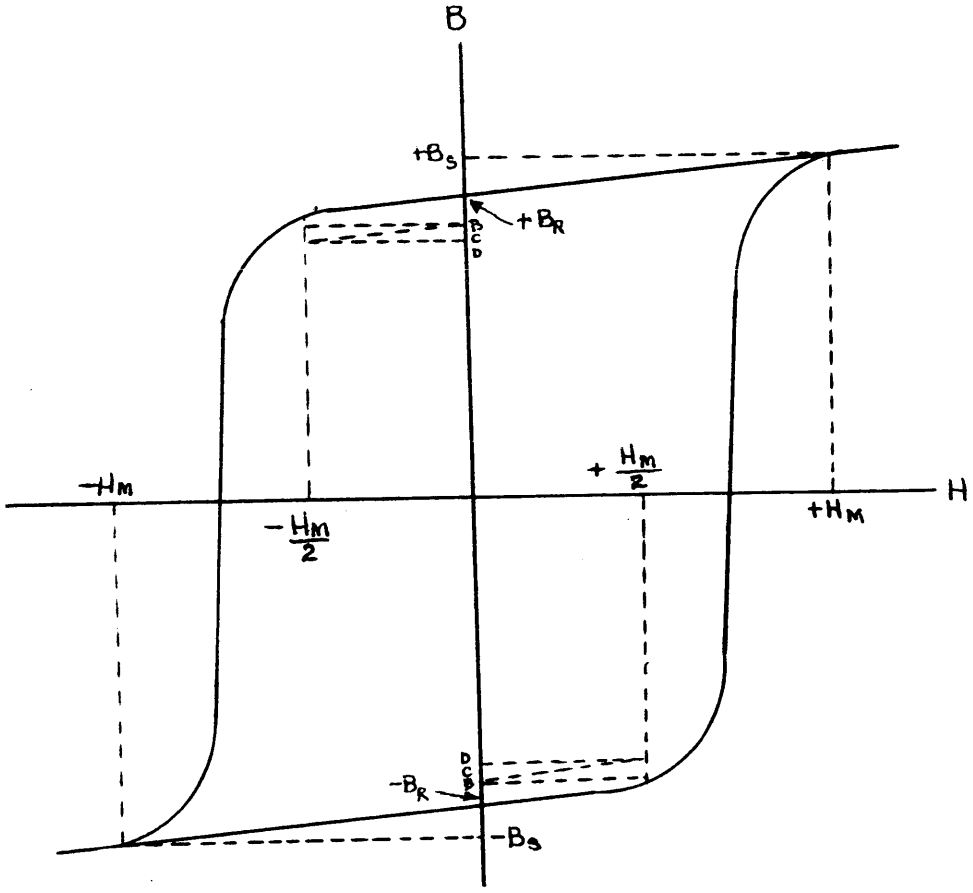


Fig. 4-2. Hysteresis Diagram

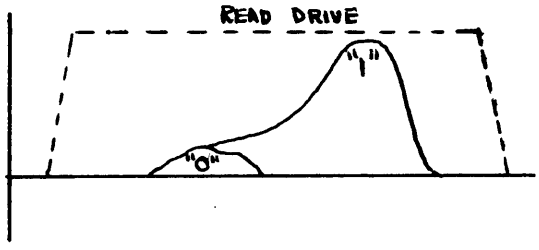


Fig. 4-3. Voltage on Sense Winding as a Result of Read Driver

Application of current flow sufficient to cause a field intensity of -Hm reverses the flux density to -Bs, and, when the current is removed, the flux density drops to the residual negative value (-Br), the 1 state.

The basic memory cycle is composed of half-amplitude pulses, each capable of producing a field intensity of Hm/2. A half-amplitude pulse is insufficient to switch the core; instead, the flux density returns to the residual value, or a slightly lower value, after the pulse is removed. The coincidence of two half-amplitude pulses, one on the H drive line and the other on the V drive line of a core, produces a net field of Hm which is sufficient to switch the core. When a half-amplitude pulse drives the flux density toward the knee of the hysteresis loop, the flux travels up (or down) the knee somewhat and then returns to a slightly lower residual value such as B. Since the core is now operating on a smaller loop, further half-pulses reduce this remanent flux again, but this effect soon reaches a limit, as at point d. Core characteristics are such that the hysteresis loop is nearly square, and half-amplitude pulses produce a small flux change. The smaller the flux change, the smaller the induced voltage on the sense line; this lowers the noise on the sense line due to unselected cores.

Any change in the magnetic state of a core causes a change in the total flux linking the core and any winding passing through it. Such a change produces a voltage output on the sense winding (fig. 4-3). During the period that H is applied, the voltage is sampled to see if the core switches. If a large voltage is sensed, the core was in the 1 state and has switched.

If only a small voltage is sensed, the core was in the 0 state and has merely shifted from +Br to +Bs and back again. All 4096 cores in a plane are strung on a common sense line. When reading, 64 horizontal cores and 64 vertical cores contribute to the sense line voltage. The total voltage induced by the 126 partially selected cores is less than the voltage induced on the line by the one core which has switched. The sense line is strung diagonally through the cores of a plane in such a way that voltages from unselected cores tend to cancel.

STORAGE MODULE CONSTRUCTION

A storage module consists of several assemblies connected mechanically and electrically.

1. Memory plane (total 12) - each holds 4096 magnetic cores
2. Stack - the 12 memory planes assembled
3. Drive deck (2) - contains circuits which supply current to the horizontal and vertical drive lines
4. Inhibit deck (4) - contains circuits which supply current to the horizontal and vertical inhibit lines
5. Connectors (2) - provide electrical access to storage module circuits
6. Miscellaneous mechanical parts for assembly and cooling.

Registers external to the storage module hold the 12-bit address and 12-bit word to be stored. External (to module) amplifiers terminate the output of the 21 double-ended sense lines and distribute the 12-bit read word.

MEMORY PLANE

A memory plane is a 4 x 4-inch printed circuit board which holds 4096 cores in a 64 x 64 matrix. The cores are strung in an alternating pattern, and five wires pass through each core. The wires terminate on the inside edges of the board, and printed wiring carries the connections to the outside edges (top and bottom) for easier access.

The 64 horizontal drive line connections are brought to connections on each of two opposite sides of the board; the 64 vertical drive lines are similarly arranged on the other two edges. The sense line is a single wire threading all cores, and its two ends are brought out at one corner of the board. Inhibit line connections are brought out to the other three corners.

STACK

The 12 memory planes are assembled into a stack about 2 inches high. A thin metal plate between each plane and on the top and bottom of the stack provides shielding and acts as a cold plate for cooling (see cooling discussion).

The drive line connections are made as shown in fig. 4-4. Each drive line threads all 12 planes and returns to its drive deck. Horizontal and vertical drive decks are identical and are located at opposite ends of the stack.

The vertical drive deck mounts on the top of the stack, and the horizontal drive deck mounts at the bottom of the stack and is rotated 90 degrees.

The assembled stack allows extremely short drive (and also inhibit) lines and hence a low inductance. The latter is a contributing factor in the cycle time or rate at which memory can be referenced and respond.

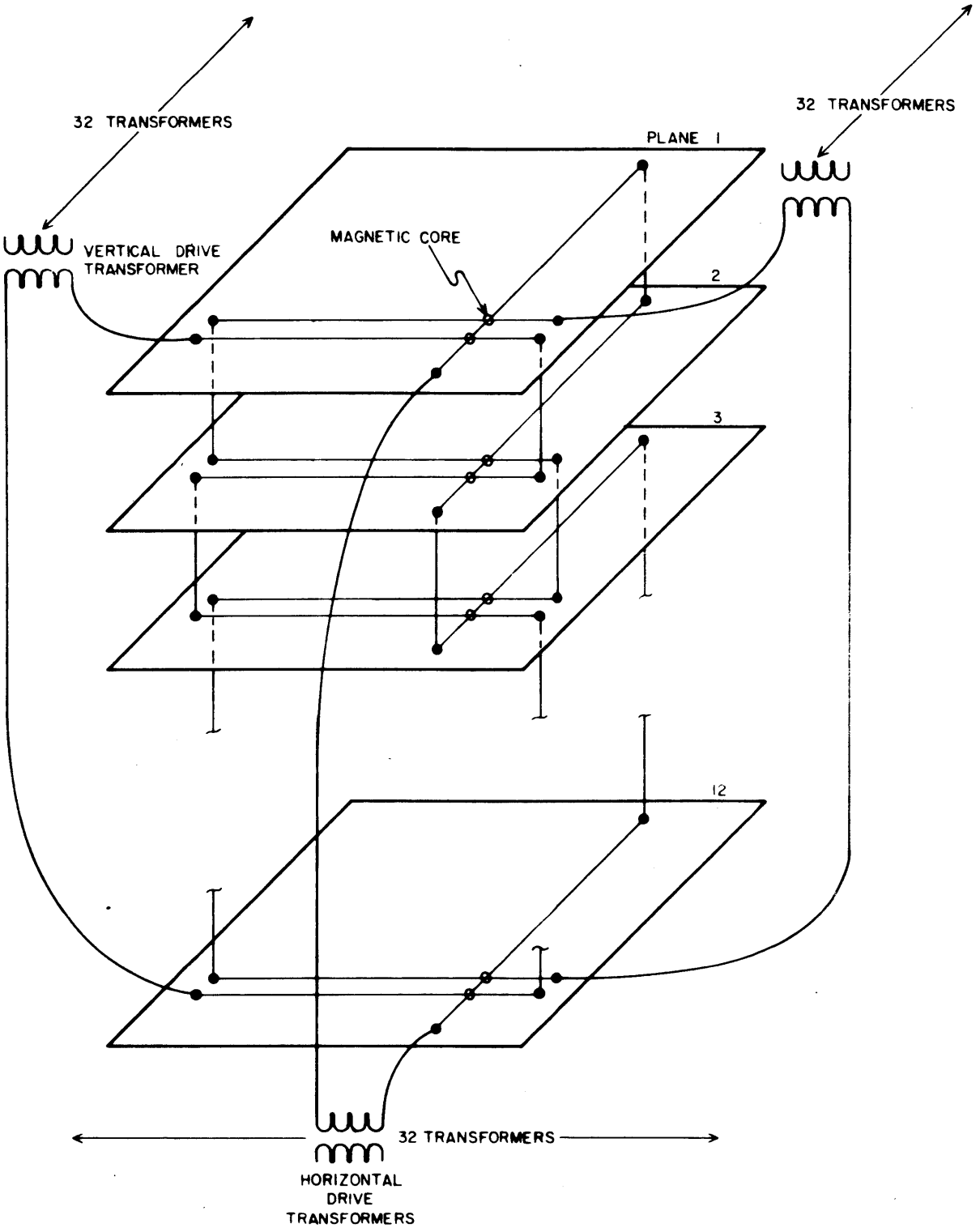


Fig. 4-4. Drive Line Connections

The sense and inhibit lines are distributed evenly at the four corners of the stack for easy connection to the drive and inhibit decks and connectors. The even distribution results from rotating a plane 90 degrees with respect to an adjacent plane.

DRIVE DECK

One drive deck supplies current to the horizontal drive lines, and the other deck supplies current to the vertical drive lines. The two decks are identical.

A deck has two printed circuit boards with printed wiring on both surfaces of a board. Components are mounted on and between the boards.

A deck contains a read and a write current source and circuits to select each of the 64 drive lines.

INHIBIT DECK

The four inhibit decks supply current to the horizontal and vertical inhibit lines. The inhibit decks are of two types, and the types differ only in printed wiring layout.

Inhibit lines on a plane are arranged in four 16-line groups in horizontal and vertical directions. Each deck has six current sources and circuits to select one of four groups on six planes.

CONNECTORS

The two connectors provide electrical access to the storage module. Table 4-1 tabulates signals.

MECHANICAL PARTS

Mechanical parts join the several electrical assemblies. The face plate supports the rest of the module and connects to the cold bars in the chassis. Various metal plates within the module also become cold bars through conduction. Component heat in the module flows by conduction and convection to the chassis cooling system.

The connectors are mounted on the rear or bottom plate and secure to mating connectors in the chassis.

ADDRESS SELECTION

A memory reference provides a 12-bit address and read or write control information, i.e., whether the address is to be read or stored. All memory references cause a complete storage cycle consisting of:

1. Read drive
2. Inhibit drive
3. Write drive

Fig. 4-5 shows memory cycle timing.

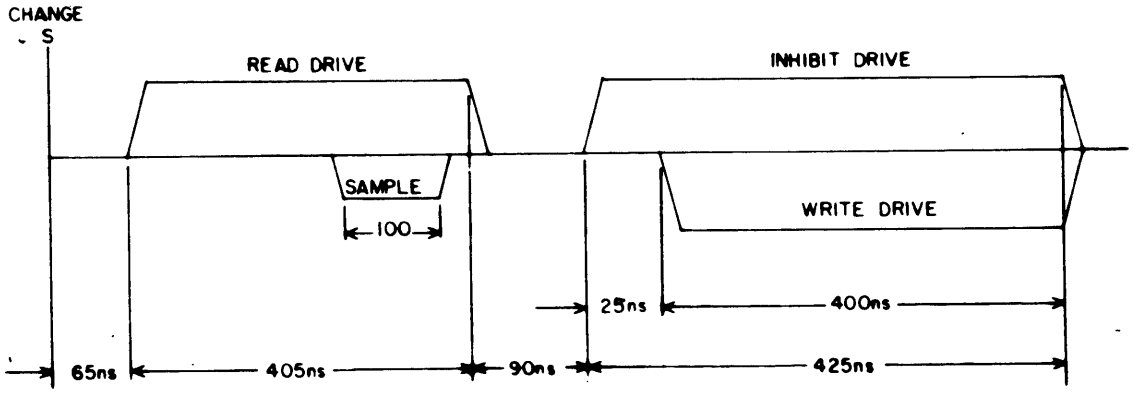


Fig. 4-5. Basic Pulse Sequence for Storage Reference

TABLE 4-1. MODULE CONNECTOR SIGNALS

ADDRESS SELECTION		
Function	Connector	Pin
S^0	2	22
S^1	2	20
S^2	2	8
S^3	2	10
S^4	1	7
S^5	1	9
S^6	1	13
S^7	1	11
S^8	2	12
S^9	2	14
S^{10}	2	18
S^{11}	2	16
WRITE (TO INHIBIT DECKS)		
Z^0	1	29
Z^1	1	5
Z^2	2	6
Z^3	2	26
Z^4	1	27
Z^5	1	1
Z^6	2	28
Z^7	2	4
Z^8	1	3
Z^9	1	25
Z^{10}	2	30
Z^{11}	2	2

READ (FROM SENSE LINES)		
Function	Connector	Pin
Y^0	2	1
Y^0	2	3
Y^1	2	27
Y^1	2	29
Y^2	1	28
Y^2	1	30
Y^3	1	2
Y^3	1	4
Y^4	2	5
Y^4	2	7
Y^5	2	23
Y^5	2	25
Y^6	1	6
Y^6	1	8
Y^7	1	24
Y^7	1	26
Y^8	2	19
Y^8	2	21
Y^9	2	9
Y^9	2	11
Y^{10}	1	10
Y^{10}	1	12
Y^{11}	1	20
Y^{11}	1	22

INHIBIT QUADRANT SELECT		
Function	Connector	Pin
S^{11}	2	24
S^{10}	1	19
S^5	1	17
S^4	1	15

CONTROL

WRITE	1	21
READ	1	23

POWER			
		Pin	Source
+8 (drive)	1	18	CB1
+5 (Inh)	2	13	CB2
+5 (Inh)	2	17	CB3
+6 (Logic)	1	14	Bus
Gnd	2	15	
Gnd	1	16	

Note:
Superscripts identify bit position in a 12-bit word.

The read drive cycle reads the 12 bits at the selected address and produces a 1 on the sense line for stored 1. In doing this, the read drive switches the cores to 0 and destroys the information. A stored 0 produces a 0 on the sense line. The data read out is restored or written in memory during the write drive cycle. The write drive attempts to write all ones for the data word, but the inhibit drive prevents writing a 1 in those bit positions where a 0 should be stored.

The word to be stored is held in an external write or restore register. The word in this register for a read memory reference is the word read out on the sense line, but for a write reference is the word to be stored. External control selects the proper word.

The 12-bit address is available and translated statically during the read, inhibit, and write drive cycles. External control sends read, inhibit, and write drive signals to the module in the proper time sequence, so that drive and inhibit current is applied to the lines as shown in fig 4-5.

DRIVE CIRCUITS

Translations of the 12-bit address provide 4096 unique codes to select any one core on a plane (actually, same core on 12 planes). Fig. 4-7 shows the relation of the address bits to the storage cycle.

The two drive decks are identical and active coincidentally since both receive the address. The drive (assume horizontal) circuits have separate primary read and write current sources, and both feed eight current transformers (fig. 4-6) through a double primary winding arrangement. These transformers thus provide eight current sources, each of which feeds one

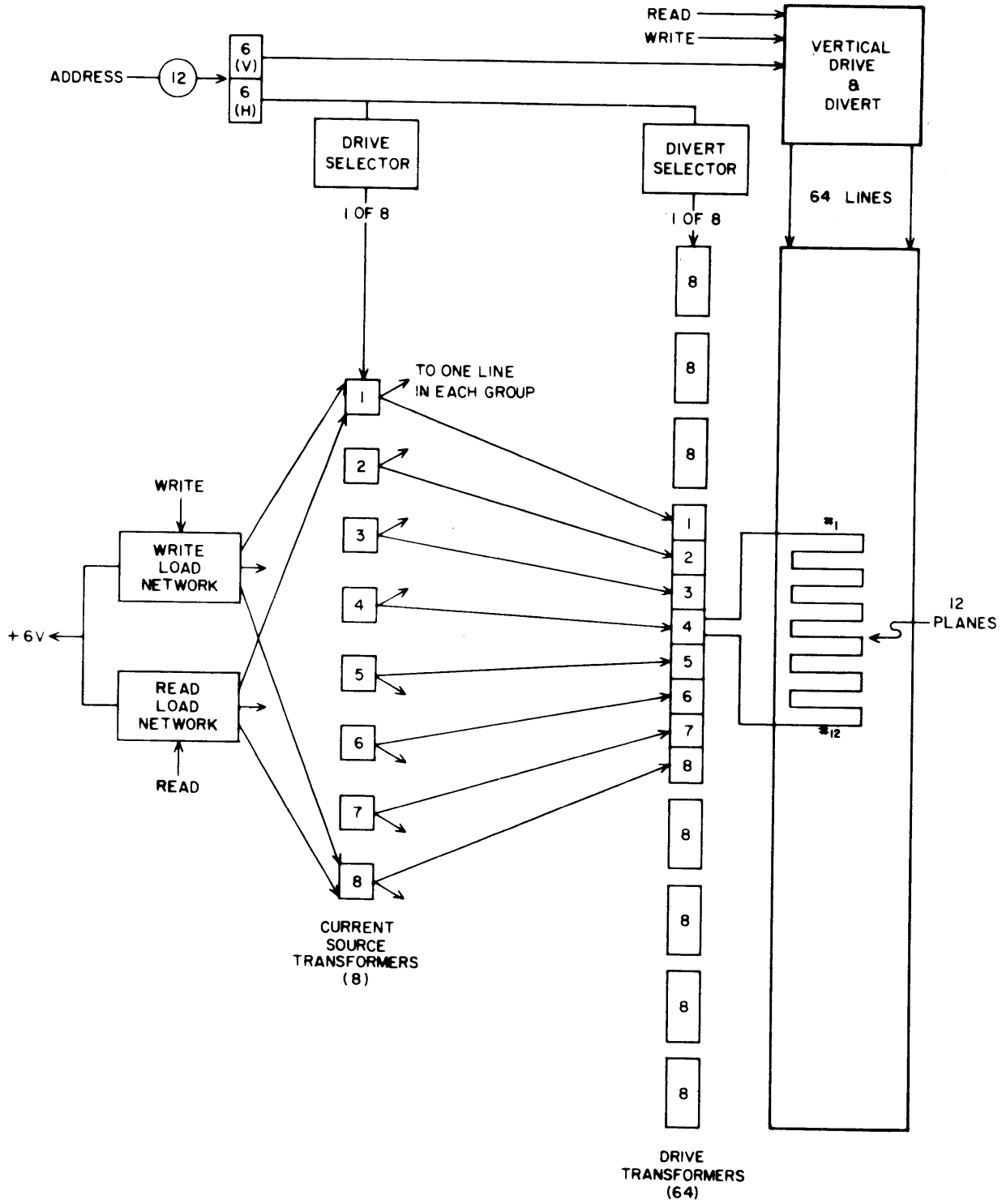


Fig. 4-6. Drive Line Selection

drive line transformer in each of eight groups. The drive transformer secondary winding connects to the ends of the drive line which threads all 12 planes.

There are eight diverter circuits, and each feeds eight successive driver transformers. One of the eight diverters is selected to divert current to the proper drive line transformer and its drive line. Thus a single current source can be diverted to any of 64 drive lines. Separate translations on 3-bit portions of the address select one of eight current source transformers and one of eight diverters to divert current from the source to a drive line. Since both drive decks are turned on at the same time, a coincident current is produced in one core on each plane.

The 64 drive transformers on each deck are so arranged that the 32 odd line transformers are on one edge and the 32 even are on the opposite edge. Thus, successive lines are driven from opposite sides of the deck. The arrangement provides an opposing current relationship with the inhibit drive lines.

Current Source

The primary read or write current source is a large inductance connected between the +6 v supply and the eight current transformers (fig. 4-7). Current of about 250 ma is available, and this is shunted to ground through transistors Q1 (and Q2) and its collector resistor (several in parallel) during non-read or non-write times. This load network is provided so that the power supply load is nearly constant at all times. Losses in the current transformer and drive line transformer allow about 200 ma to be delivered to the drive line during the storage cycle.

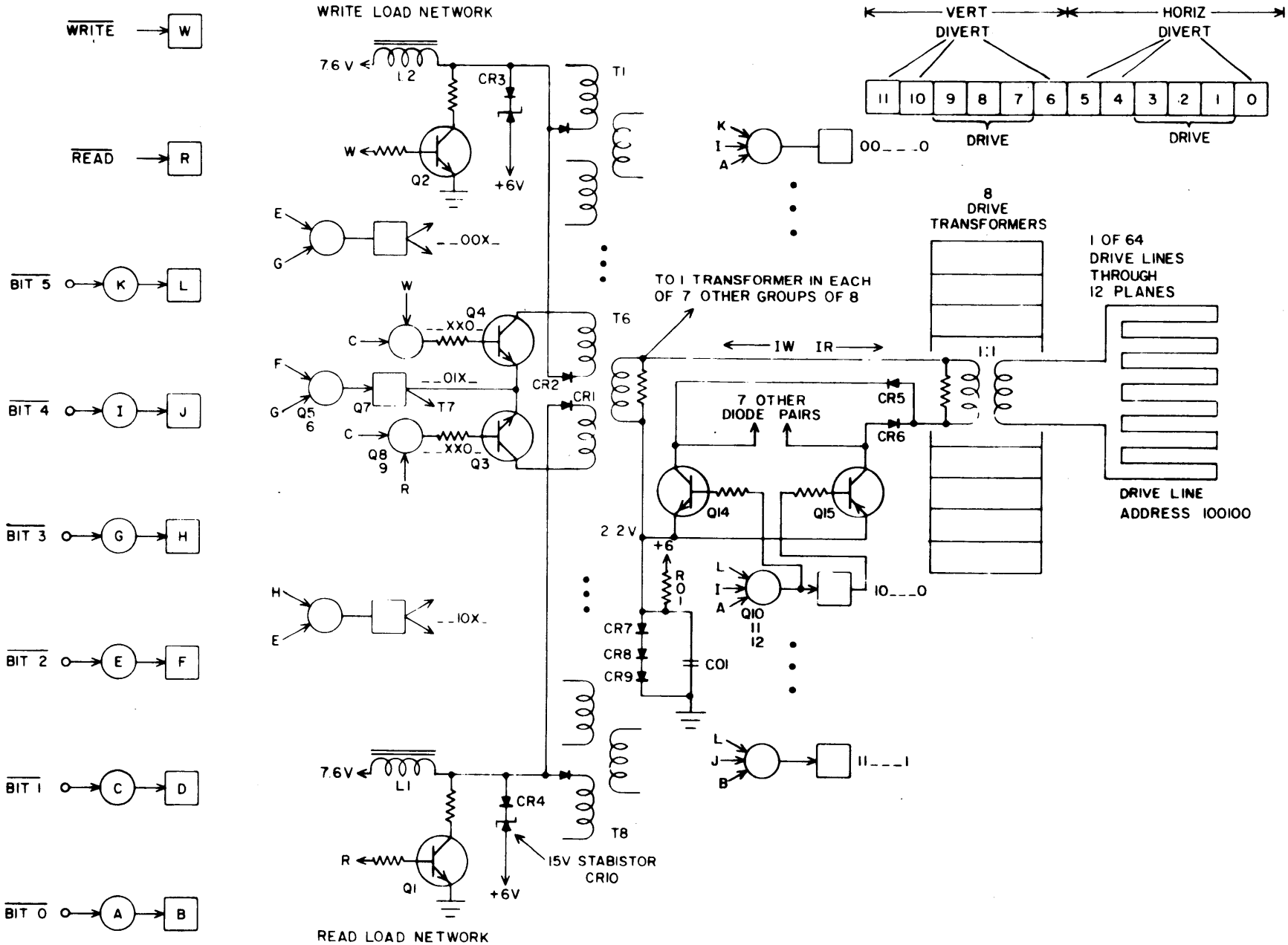


Fig. 4-7. Drive and Divert Circuits

One of eight current transformers is selected from a 2 x 4 transformer matrix. The upper 2 bits of the 3-bit drive translation select one of four transformer groups, and the lower bit selects one of the two in that group. The read or write drive signal is combined with the one of two selection.

Figure 7 assumes a --100- binary code in the 3-bit drive position, and shows that transistors Q5 and Q6 turn off (shown as AND) and Q7 turns on to select transformers T4 and T5 in the one of four group; transistors Q8 and Q9 turn off and Q3 turns on to select the read side of T4. Read load network transistor Q1 turns off, and current flows from ground to the +6 v source via transistors Q7 and Q3, and transformer T4, diode CR1, and inductor L1.

A high impedance is presented to inductor L1 when read load network transistor Q1 turns off, and the voltage level at the junction of diode CR1 and inductor L1 rises rapidly. However, diode CR4 conducts to place stabistor CR10 in the circuit, and the latter limits the peak voltage at the junction to 15 v. Current loss in T4 is about 25 ma, so 225 ma flows in the secondary and is diverted to a drive transformer.

A voltage is coupled to the other primary winding of T4 when Q3 is conducting. The voltage back-biases diode CR2 to prevent the collector of Q4 from becoming negative with respect to its base. This would turn on an unselected circuit and thus defeat the selection mechanism.

The circuit operates in a similar manner in the write cycle except that polarities in the current transformer are reversed.

Diverter Circuits

The eight diverter circuits divert read and write current to the drive transformers. Each circuit feeds eight drive transformers, one for each current source transformer. Thus the selected current source transformer drives eight lines and only one of the eight is diverted.

Fig 4-7 shows one of eight diverter circuits and its selection from the 3 address bits used for divert. The collectors of Q10, Q11, and Q12 feed the base of NPN transistor directly and the base of PNP transistor Q15 indirectly via transistor Q13 which inverts the combined collector signal. The action establishes positive and negative turn-on signals for the bases of Q14 and Q15. Emitter bias on these two transistors is established by the divider network of R01, C01, and diodes CR7, 8, and 9. The diode threshold is about .7 v for a combined drop of about 2.2 v and proper emitter bias.

Diode CR5 is forward biased during read drive to complete the circuit of the selected current source transformer. Diode CR6 is back-biased during read drive and Q15 does not conduct. Current flows from the current transformer through the drive transformer, diode CR-5, transistor Q14, and back to the current transformer.

Some current is lost in the drive transformer, and about 200 ma of the available 225 ma is delivered to the drive line, which appears as a balanced load to the drive transformer secondary. Resistors connected across the current source secondary winding and drive transformer primary winding load the transformers and damp out the transformer ringing produced when the transistors shut off at the end of the cycle.

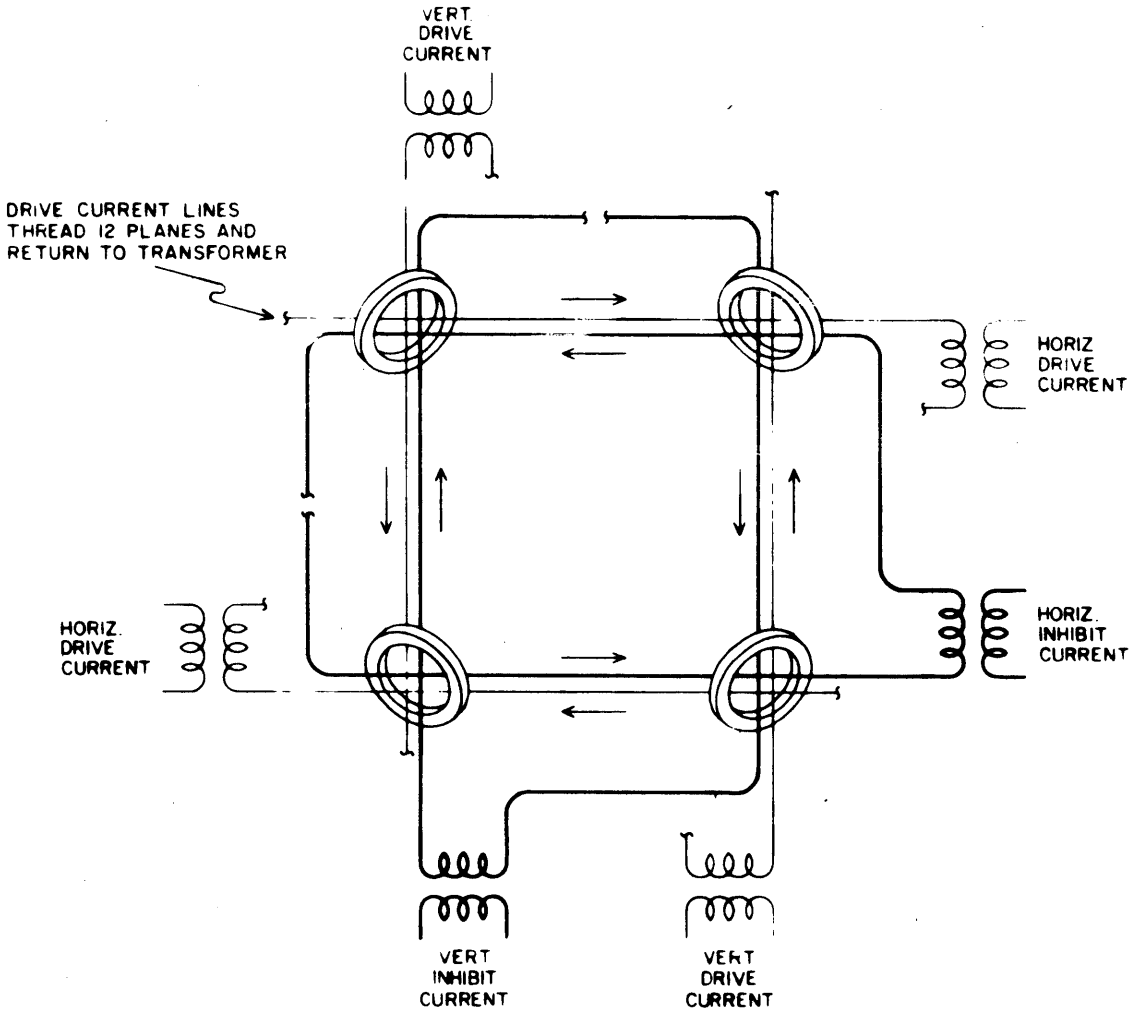


Fig. 4-8. Path of Inhibit Wires Through a Four-Core Matrix

Diode CR6 is forward biased and diode CR5 back-biased during write drive. The circuit operates the same as during read drive except current flow and polarities are reversed. The write drive starts (write load network turned off) 90 nsecs after the read drive stops. The delay period allows the transformers to discharge their stored energy, thereby preventing a flux buildup or transformer bias condition.

INHIBIT CIRCUITS

The inhibit circuits cancel the effect of write drive and thereby allow zeroes to be stored in memory. Inhibiting is done with 100 ma drive current on coincident horizontal and vertical inhibit lines. The resulting 200 ma coincident inhibit current produces a magnetic field in opposition to the field from the 400 ma coincident current write drive, thereby cancelling half of the write drive field. The resultant field, or half-amplitude pulse, is not sufficient to switch a core, and so it remains in the 0 state as produced by the read drive.

Inhibit drive is plane selective and is turned on when a given plane should store a 0 as indicated by the content of the restore register. The register holds the complement of the word to be stored. Fig. 4-8 shows the opposing current relationship of inhibit and write drive signals in a 4-bit plane.

The inhibit lines are arranged in four 16-line groups (fig. 4-9) in the vertical and horizontal directions. Two inhibit decks drive vertical lines, and the other two drive horizontal lines. Each deck handles the four groups in six planes. Only one 16 x 16 core area in each plane receives horizontal and vertical inhibit current when a 0 is to be stored.

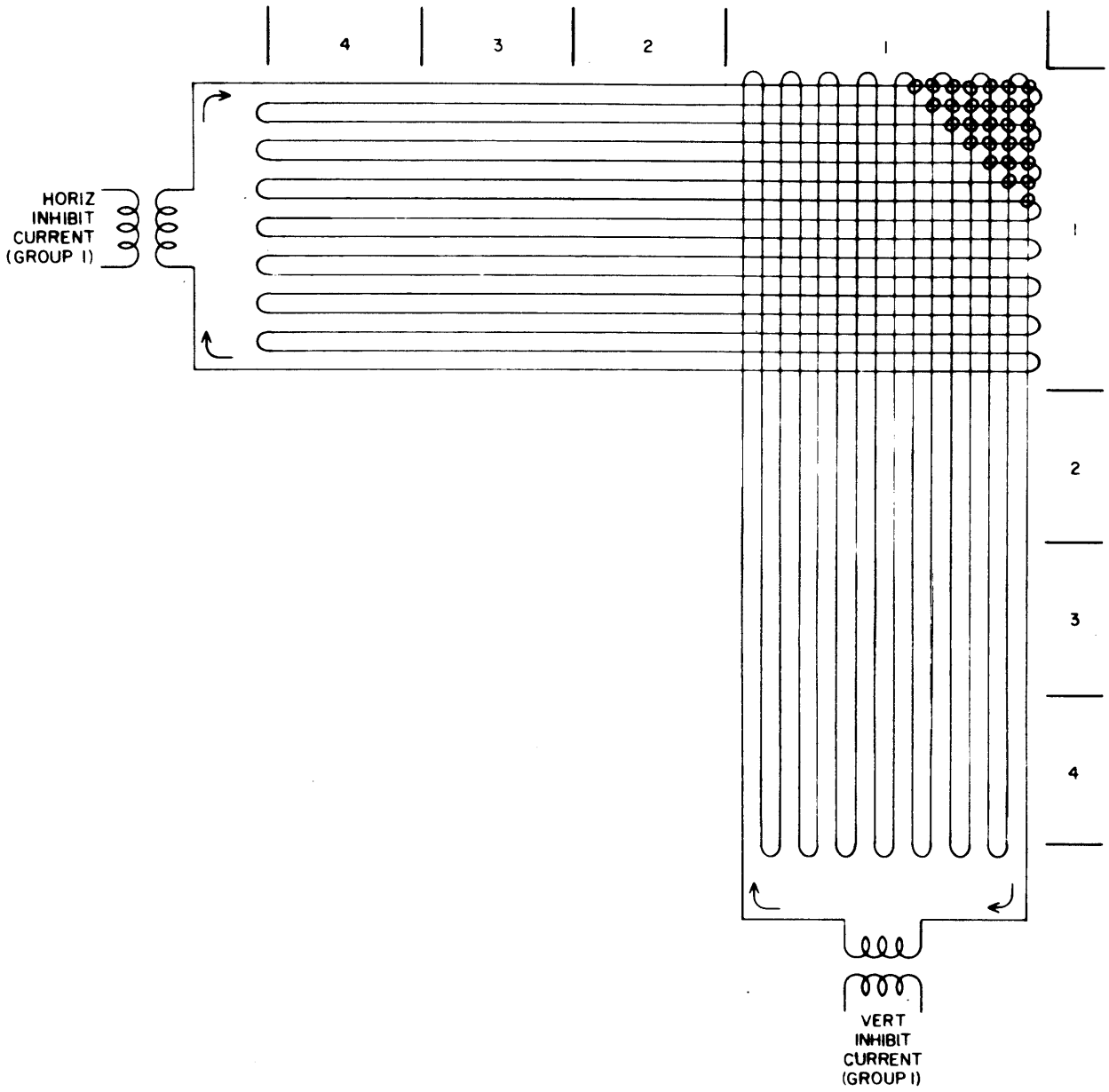


Fig. 4-9. Inhibit Line Grouping

The group chosen is a one of four selection from a translation of a 2-bit address quantity. Separate 2-bit address quantities (figure 10) accommodate horizontal and vertical selections.

Fig. 4-10 shows an inhibit group selection circuit (assume horizontal) activated by two address bits. The circuit forms four unique translations of the bits using diode AND circuits. The output of an AND feeds the base of an emitter follower, which in turn drives one inhibit transformer on each of six planes. An AND circuit turns on its emitter follower when both of its inputs (diode cathodes) are at 1.2 v; the anode junction then rises to this potential to turn on the emitter follower.

A separate current source drives the four inhibit transformers required for each plane. One of these is shown in figure 10 with the circuit to drive a transformer. During read drive, or first portion of the storage cycle, the external register (Z) holding the word to be stored is cleared, and slave inverters from its set output provide a +1.2 v signal to turn on transistor Q1, which, with its collector resistor, is a load network for inductor L1. Inhibit current is shunted to ground through R1 and Q1 during non-inhibit periods.

The complement of the word to be stored is entered in the restore register 600 nsecs after the address is received by the storeage module. Thus, a 0 bit appears in the register as a 1, and vice versa. In the figure, a 0.2 v signal turns off load network transistor Q1, and its collector potential rises positive to turn on the base of transistor Q2. The emitter of the latter transistor is already near ground potential because the group selection turned on transistor Q3 at the beginning of the storage cycle. The Q2 collector potential rises rapidly to about 1-v when Q1 turns off and then drops off as Q2 starts to conduct current through the transformer primary. A peak current of about 50 ma flows during the voltage swing.

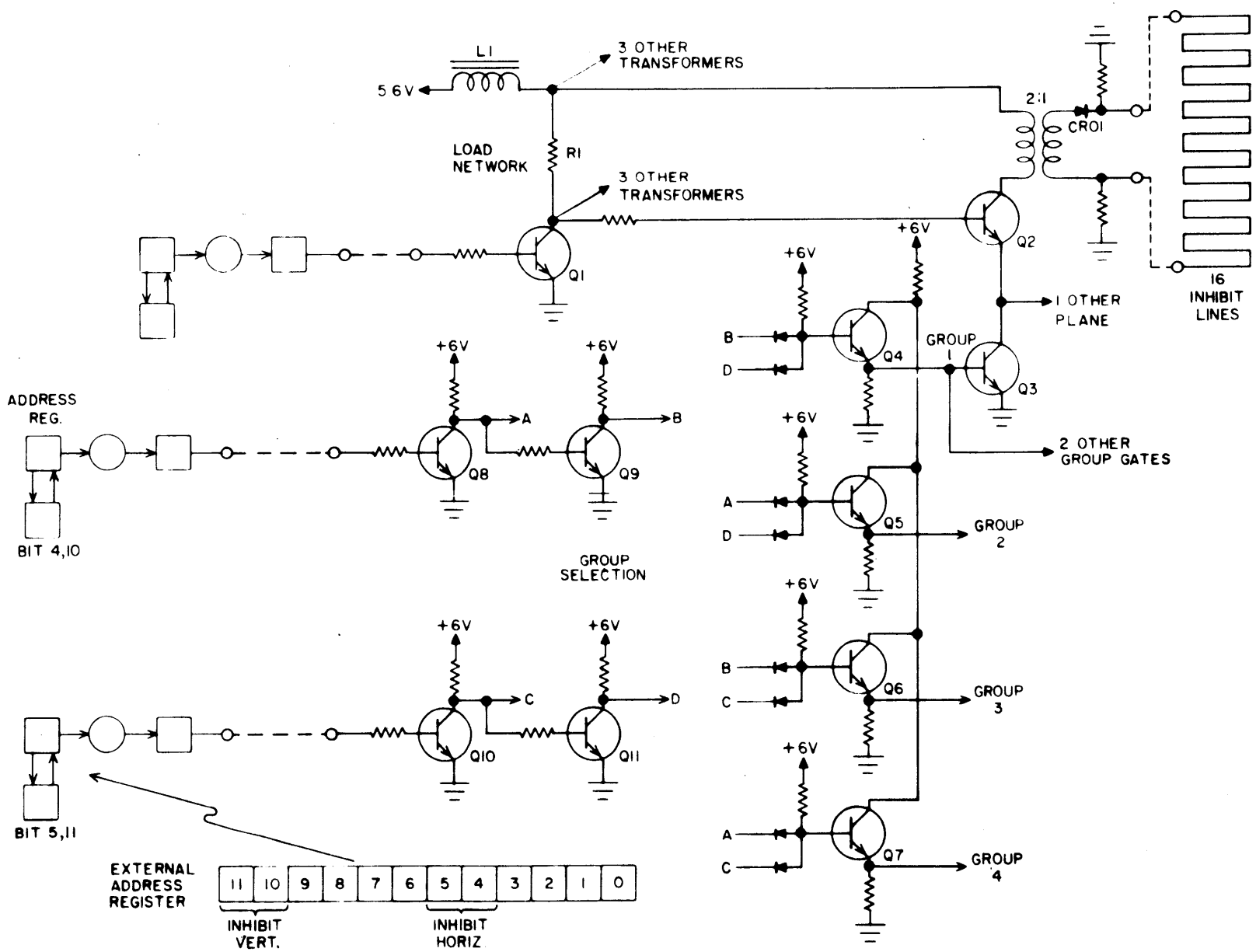


Fig. 4-10. Inhibit Circuits

The transformer turns ratio is about 2:1, resulting in a current step up in the secondary winding to about 100 ma. The secondary voltage turns on diode CR1, and the latter conducts the 100 ma current through the inhibit winding, which appears as a balanced load. A corresponding 100 ma inhibit current in the vertical plane produces the 200 ma half-amplitude current pulse necessary to cancel the effect of the write drive.

The Z register is cleared 425 nsecs after it is set. This action turns on the 12 load network circuits to stop current flow in the transformer primary and turn off inhibit drive. The collapsing primary field of the inhibit transformer produces a large voltage pulse on the collector of Q2. The polarity of this signal is opposite to the desired polarity of the pulse which appeared at the beginning of the cycle. At this time the secondary impedance appears quite large to diode CR1, and it cuts off so that a large secondary voltage is generated, but no secondary current flows. The voltage produces a flux field in opposition to that produced at the beginning of the inhibit drive. Thus, secondary flux is cancelled to avoid bias shift in the transformer from repeated cycling.

Central Memory has 32 banks located on 8 chassis, 4 banks to a chassis.

Address and Tag Format from Stunt Box

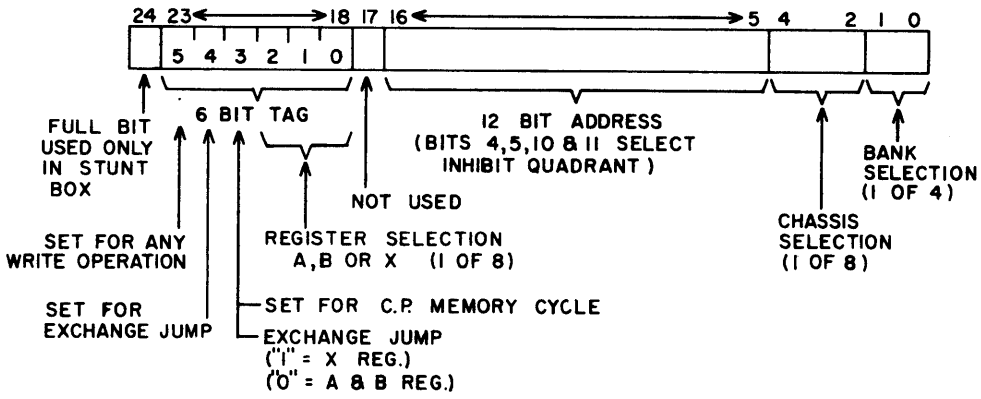


Fig. 4-11a

Translation Table for Bits 0-5

000 00	Chassis 3	Bank 0	Bank No. 1
000 01	Chassis 3	Bank 1	Bank No. 2
000 10	Chassis 3	Bank 2	Bank No. 3
000 11	Chassis 3	Bank 3	Bank No. 4
001 00	Chassis 4	Bank 0	Bank No. 5
001 01	Chassis 4	Bank 1	Bank No. 6
001 10	Chassis 4	Bank 2	Bank No. 7
001 11	Chassis 4	Bank 3	Bank No. 8
010 XX	Chassis 9	Bank 0-3	Bank Nos. 9-12
011 XX	Chassis 10	Bank 0-3	Bank Nos. 13-16
100 XX	Chassis 13	Bank 0-3	Bank Nos. 17-20
101 XX	Chassis 14	Bank 0-3	Bank Nos. 21-24
110 XX	Chassis 15	Bank 0-3	Bank Nos. 25-28
111 XX	Chassis 16	Bank 0-3	Bank Nos. 29-32

eg M1 = 14272116 = Cent. Proc. Read → X^4 Register
Address 5642 of Bank 2 in Chassis 10 [Bank No. 15]

Fig. 4-11b

B. Memory Referency Request

For every Memory reference requested, a GO signal along with the address is sent to all 32 banks of CM. from the M^1 Register in the Stunt Box. If memory can be accessed (no bank conflict), an 'ACCEPT' signal is sent back to the Stunt Box. See Table in Fig. 4-11 for 24 bit M^1 format and CM. Chassis and Bank distribution.

GO CONTROL (Fig. 4-12)

The Go Control circuitry is located on Chassis 4, the GO being sent to Module I20, from where it fans out to all CM. Chassis. Bits 0 to 5 are sent with the GO simultaneously to the 8 Chassis. Bits 2, 3, 4 are for Chassis selection corresponding to FF's B, C and D in G4/G38 (EVEN/ODD CH.) FF. E is for the GO, and FF.A is not used - a possible provision for a larger memory (8 more Chassis). FF's B, C and D are preset by a timing pulse to the complement of the Chassis in which they are located. Bits 2, 3 and 4 are wired in from the hopper appropriately to the set or clear inputs of the FF's. The outputs of B, C, D and the GO are ANDED and the GO transmitted to G5/G39 (EVEN/ODD CH.) if the gate is made. Bank selection is made here with bits 0 and 1 in a similar manner. An additional gating term, a bank free condition which ensures that the selected bank is not in a previous memory cycle, gates the GO to the storage sequence control and sends back an ACCEPT. No ACCEPT is sent to the Stunt Box if the selected bank is executing a storage cycle from a previously issued address. In this case the Address is saved in the Hopper and reissued every 300 nsecs until accepted. In the worst case of bank conflict, an address can remain in the Hopper for 2700 nsecs. (Refer to discussion on Stunt Box).

ACCEPT CONTROL (Fig. 4-12)

The Accept indicates a bank is free and has accepted the address into its Chassis input register. The time at which the GO is issued from the Stunt Box until the time the Accept is received back is an interval of 200 nsecs. The Accept signal is generated at G5/G39 (EVEN/ODD CH.) of the respective Chassis and is routed to Chassis 3I23 which is an OR Circuit (a fan-in to send back the Accept from any of the eight Chassis). Hence an Accept can be sent back to the Stunt Box every 100 nsecs., assuming no bank conflict.

The GO and ACCEPT CONTROL Circuitry does the following:

1. Recognizes an address from the Stunt Box and determines its bank location.
2. Sends an Accept if the address is valid and the bank is free.
3. Starts the 1000 nsec. Storage cycle.

C. STORAGE CYCLE

ADDRESS PATH

The 12 bit address is sent to the eight memory Chassis [G1/G40 (EVEN/ODD CH.)] 50 nsecs. after the GO. In each Chassis the address fans out to all four banks. The address is gated by a Bank FREE signal into the storage address registers of all free banks, but Read Drive is turned on for only one bank - the selected address. Here, each bit fans out to five storage modules of 12 planes each, to select one 60 bit word location (Fig. 4-13). Bits 4, 5, 10 and 11 have an additional five outputs to select the inhibit quadrant in each storage module (Fig . 4-14). The storage address register is statically translated during the whole storage cycle.

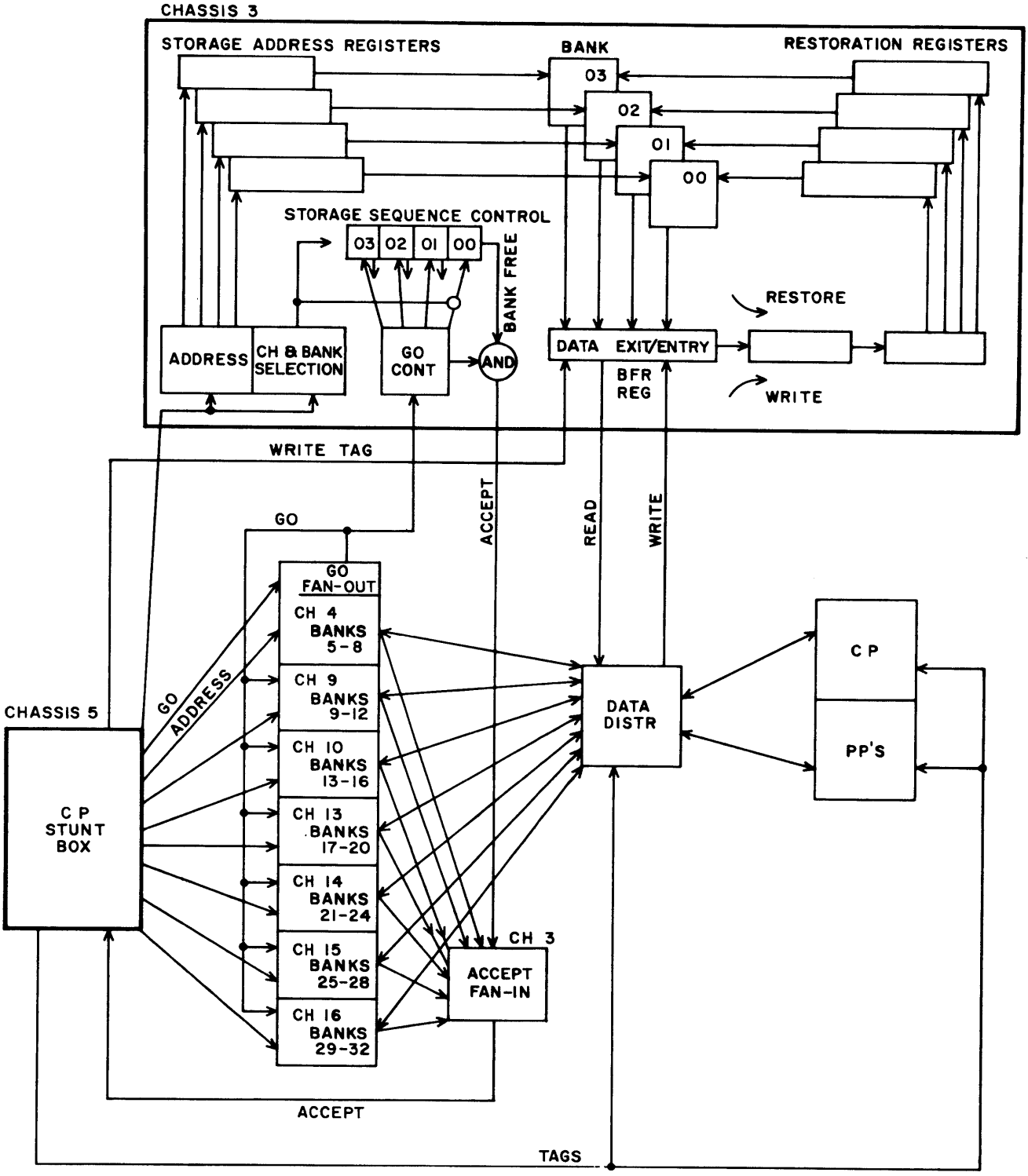

CHASSIS 3
STORAGE ADDRESS REGISTERS
BANK
RESTORATION REGISTERS
03
02
01
00
STORAGE SEQUENCE CONTROL
03 02 01 00
BANK FREE
RESTORE
ADDRESS
CH & BANK SELECTION
GO CONT
AND
DATA EXIT/ENTRY
BFR REG
WRITE
WRITE TAG
GO
ACCEPT
READ
WRITE
GO FAN-OUT
CH 4 BANKS 5-8
CH 9 BANKS 9-12
CH 10 BANKS 13-16
CH 13 BANKS 17-20
CH 14 BANKS 21-24
CH 15 BANKS 25-28
CH 16 BANKS 29-32
CHASSIS 5
GO ADDRESS
C P STUNT BOX
DATA DISTR
C P
PP'S
CH 3
ACCEPT FAN-IN
ACCEPT
TAGS

Fig. 4-12

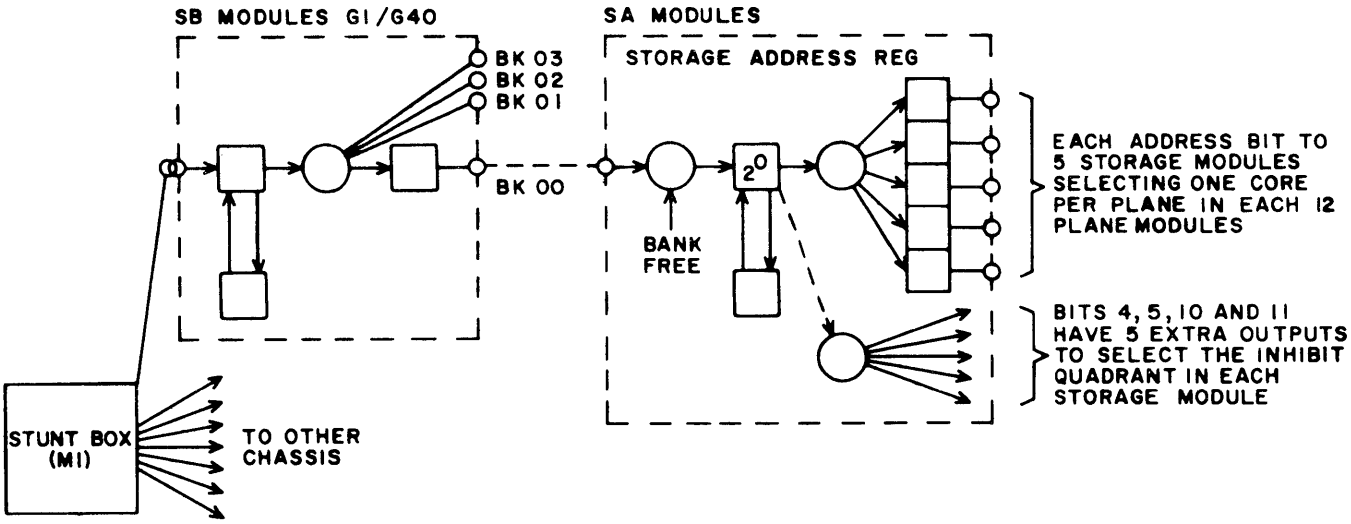


FIG. 4-13 ADDRESS PATH

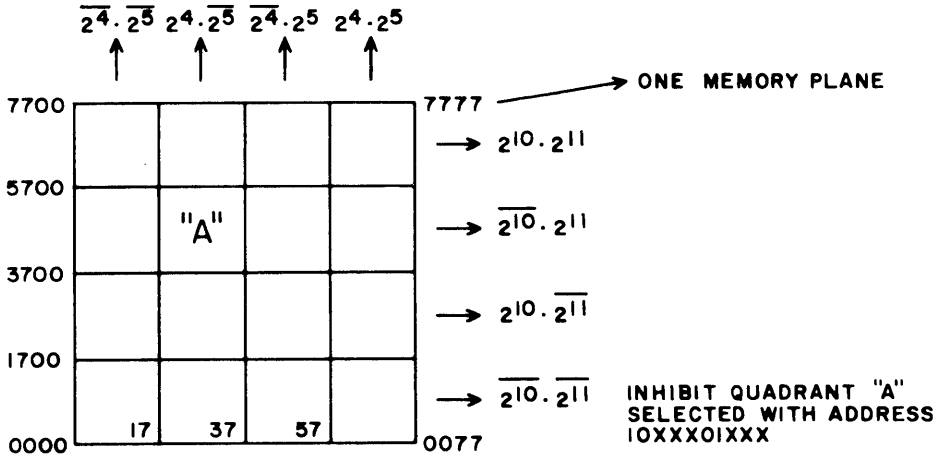


FIG. 4-14 INHIBIT QUADRANT SELECTION

STORAGE SEQUENCE CONTROL (Fig. 4-15 and Timing Chart)

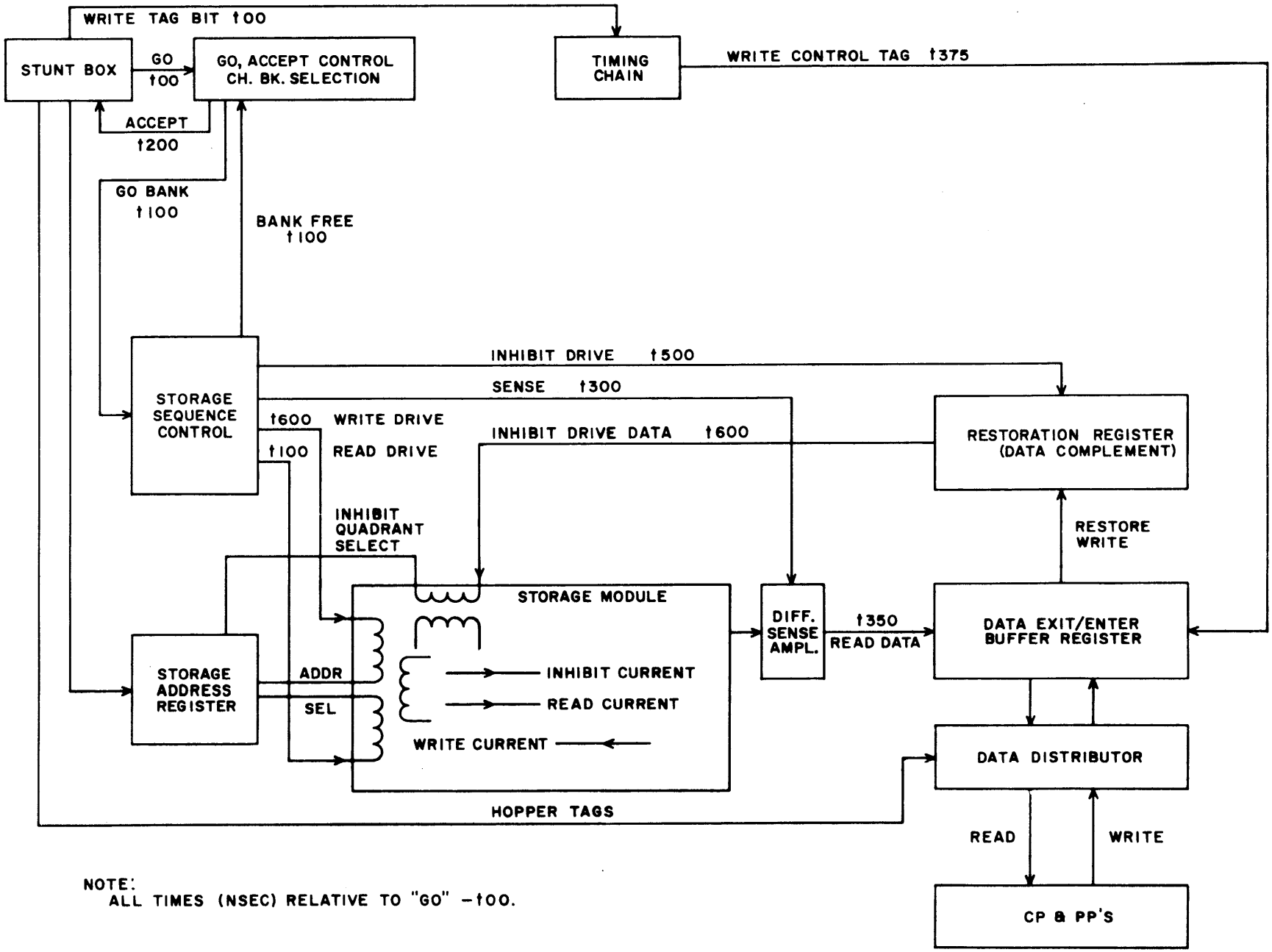
The Storage Sequence Control, located in each memory chassis, is a chain of FF's whose outputs drive slave inverters, generating a series of Timing signals which direct the basic cycle of the storage modules. The Bank GO pulse starts the sequence by setting the Read FF and this pulse is transferred to successive FF's at 50 nsec. intervals. Each FF is set for 400 nsecs. (Fig. 4-15). The Bank Free condition is dropped as soon as the Read FF is set.

a) Read Drive. The clear output of the Read FF starts the 400 nsec. read drive. This is sent to the five storage modules and turns on read current in one core of each plane (H & V wires) already selected by the address. This enables data to be read out of memory.

b) Sense Drive. 200 nsecs. after the read drive, a 100 nsec. sense signal is generated to sample the differential amplifier which receives the data word read out on double-ended sense lines from storage. This sample pulse accomplihses two operations:

1. Samples the data read out of memory.
2. Allows a bank merge at the Data Exit/Entry Buffer Register (SE modules) common to the four banks and gates data to the data distributor and Restoration register.

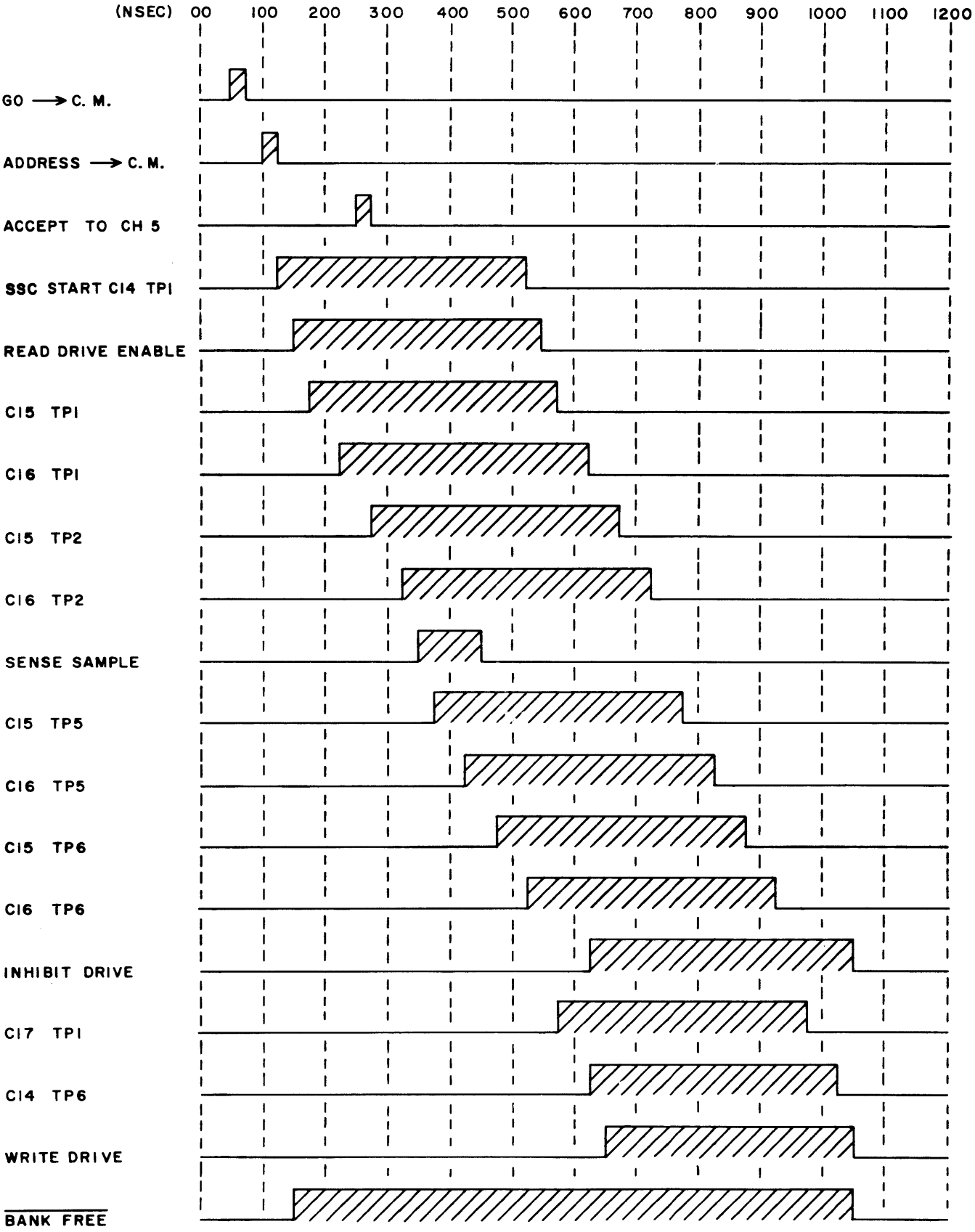
c) Inhibit Drive. The timing chain initiates the inhibit drive 475 nsecs. after the read drive. During this 400 nsecs. before the start of the inhibit drive, data has been read out of memory and transmitted trhough two holding registers common to all banks. The first, H28/H1 (EVEN/ODD CH.) holds the true value of the data word read out and the second G24/G1 the complement.



NOTE:
ALL TIMES (NSEC) RELATIVE TO "GO" –†00.

FIG. 4–15 MEMORY CYCLE BLOCK DIAGRAM

CENTRAL MEMORY BANK CONTROL (BANK 00)



The inhibit gates the complement into the restoration register which also stores the complement value. Depending on the data word, 0 or 1, inhibit current will be turned on or not, respectively. To write a zero into memory the inhibit current controlled by the restoration register content (complement) is turned on and opposes the write current. For a 1 inhibit current is voided. 900 nsecs. after the read drive an end inhibit signal clears the restoration register and ends the inhibit drive.

d) Write Drive. The timing chain transmits the write enable signal 500 nsecs. after the read drive. This is sent to the five storage modules and like then read drive, turns on write current in one core of each plane (H and V wires) already selected by the address. The write current flows in the opposite direction of the read, to switch the core to a 1. If a 0 is to be written, inhibit current opposes the write current and the core does not switch.

MEMORY READ

As outlined above the following is the sequence of operation:

1. GO → Chassis and Bank Selection → Accept to Stunt Box.
2. Address → Storage. Selects parituclar core in each plane, i.e., 60 planes in 5 twelve plane modules.
3. Read Drive. Reads data out of memory.
4. Sense. Samples data and transmits it to the data Distributor and restoration register.
5. Inhibit and Write Drive. Rewrites what has been read out.

MEMORY WRITE

The memory write is similar to the read up to Step 4. For every write operation a write signal (tag) is issued from the Stunt Box 150 nsecs. after the address. This passes through a timing chain and is so timed, that after Step 4 of the memory read has been accomplished, the write control signal clears the Data Exit/Entry Buffer register to allow the word to be written, to be gated in from the data distributor. The word that has been read out already is destroyed in the data distributor due to a lack of destination. Step 5 as outlined in the Memory Read is now executed and the new word written into memory.

A word from the data distributor during a write reference goes to the Data Exit/Entry Buffer register on all chassis and then follows the restore path for writing in memory. Only one of the many banks is in the proper time spot in its storage cycle to store the word received, and this bank is the one associated with the write address.

To prevent writing into an address location before a previous read reference has been accomplished, the hardware in the Stunt Box never allows both a read and write to be in the Hopper at the same time. The exception is that if there is a PP central memory read in the hopper, a CM. write from the Central Processor can gain priority into the hopper. Software should take care of this to prevent a CM write in an address location that has not been already read by a previous PP instruction.

BANK FREE SIGNAL

The bank free condition is established when all FF's in the timing chain are cleared, i.e., no pulse travelling down the chain. The Read FF (first FF in chain), an intermediate FF and the Write FF (last in chain) contribute timing signals to the bank free circuitry (G9, G19/G29, G26 EVEN/ODD CHASSIS) and indicate whether a pulse is in the chain. All these FF's must be clear to signal bank free end of the storage cycle.

D. Data Distributor

The data distributor essentially acts as a post office distributing data to and from CM, the maximum transfer rate being 100 nsec. The distributor routes data to and from proper origins and destinations as directed by control information or tags entered in the stunt box along with each address. The tags serve to identify the address sender, origin or destination of data, and nature of the address, e.g., read, write, or PP exchange jump.

CONTROL TAGS

The six bit tag is sent from the Stunt Box to the tag translator. After translation the tag is sent to the data distributor to identify the origin and destination of data. For any write reference a write tag bit is sent directly from M^1 in the stunt box through a timing chain to the Data Exit/Entry Buffer register, enabling the data from the write distributor into the restoration register.

READ DISTRIBUTOR

The read distributor accepts read words from the 8 CM chassis and routes them to the several destinations.

The distributor is organized on chassis 3, 4, 9, and 10, each of which handles 15 bits of the 60-bit word. Chassis cable limitations dictate the organization. The listing below shows the bits handled by each chassis.

CHASSIS	BITS
3	0-14
4	15-29
9	30-44
10	45-59

Thus, chassis 13-16 (Fig. 4-16) each send the same 16-bit group to chassis 3, 4, 9, and 10. A read word from chassis 3 retains bits 0-14, but sends remaining bits in three groups to chassis 4, 9, and 10. Read words from chassis 4, 9, and 10 are handled similarly. Intra-chassis coaxial cables are used on chassis 3, 4, 9, and 10 for their 15-bit portions so that timing is consistent with the chassis receiving the data.

Each read word is sent unconditionally from chassis 3, 4, 9, and 10 to chassis 5 (CP control) and chassis 7 and 8 (CP registers) as shown in Fig. 4-16. A read peripheral tag from the stunt box is sent to chassis 4 and then on to chassis 3, 9, and 10. The tag gates the read word to the C^5 register in the read pyramid on PP chassis 1.

The read peripheral tag also enters a time delay chain and is returned to the PP as a resume signal. The resume sets the C^5 full FF in the PP (after data word is in C^5) to signal the presence of the read word. The same resume also clears the central busy FF to indicate to PP control that the address has been accepted by the stunt box and CM has delivered the word. This allows the PP to proceed and send another address to the stunt box.

WRITE DISTRIBUTOR

The write distributor accepts words from the several sources and stores them in 1 of 8 memory chassis. The distributor is on chassis 2. The 60-bit word on chassis 2 is split into four 15-bit groups which are sent to chassis 13-16 respectively. Each of these chassis in turn sends (or stores) its 15-bit group to the other 7 chassis (Fig. 4-16) unconditionally. The JH modules (I28-I37) on chassis 13, 14, 15, 16 have 12 outputs, 8 for one bit sent to all 8 CM chassis and 4 for another bit sends to 4 chassis only. The other four chassis are fed with this same bit from another JH module.

Fig. 4-16 shows a detail of the write path. The 3-to-1 fan-in on chassis 2 selects the proper word under control of the store tag from the stunt box which is established ahead of the data. The word is then split and transmitted to chassis 13-16. The chassis 2 data registers and the tag GGs are cleared simultaneously.

One minor cycle after the register clear, a central write resume is sent to the PP to clear the central busy FF. This allows the PPs to send another address to the stunt box.

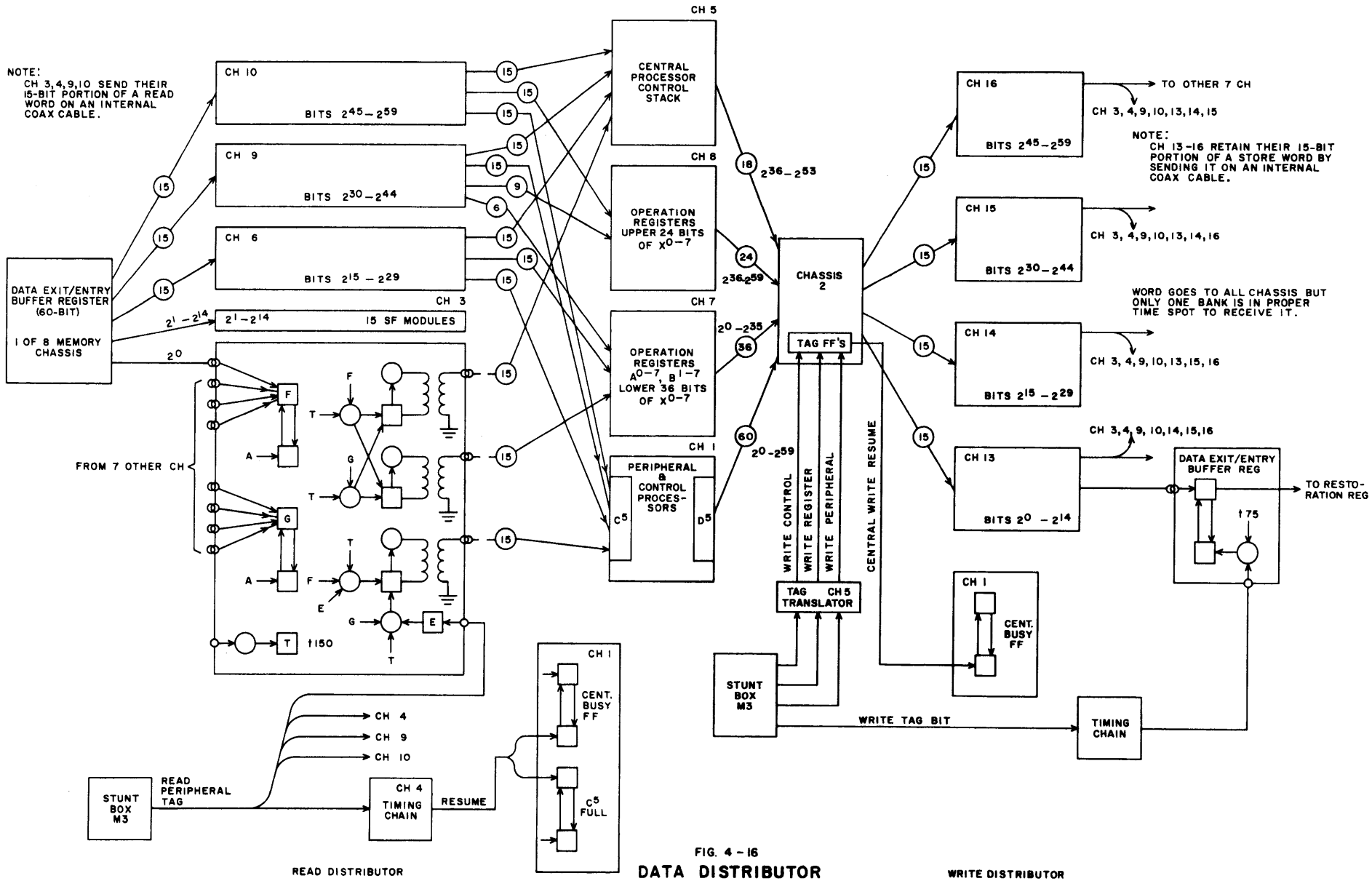


FIG. 4-16

DATA DISTRIBUTOR

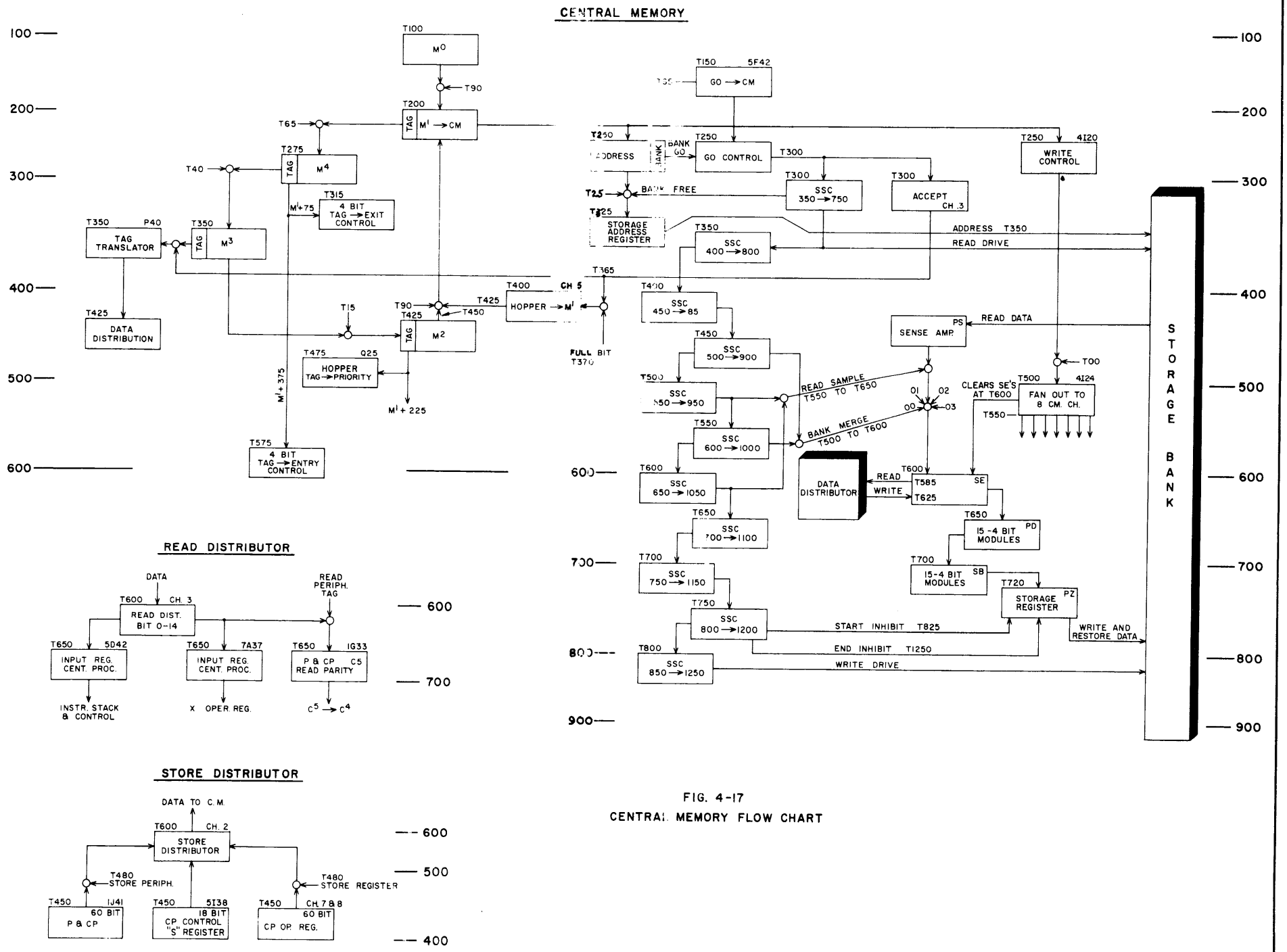


FIG. 4-17
CENTRAL MEMORY FLOW CHART

SECTION V

Peripheral and Control Processors

PERIPHERAL AND CONTROL PROCESSORS

INTRODUCTION

The Control Data® 6600 computer consists of ten peripheral and control processors, a central processor, and central memory plus I/O synchronizers. Each peripheral and control processor is an independent computer with 4096 words of core storage and a repertoire of 64 instructions. The peripheral and control processors share access to central memory (read and write) and to 12 bi-directional input-output channels.

The ten processors are combined in a multiplexing arrangement which allows them to share common hardware for arithmetic, logical, I/O, and other operations without sacrificing speed or independence. This multiplexing arrangement consists of a barrel and slot and common paths to and from storage and I/O circuits (Fig. 5-1).

The barrel is a matrix of FFs used to hold the current instruction and operand for each of the ten processors and to give each a turn to use the execution hardware in the slot (adders, shift network, etc.). The quantities in the barrel are shifted from slot output to slot input (Counter-clockwise in Fig. 1) and each time a processor's data enters the slot a portion of the instruction is executed (one trip around the barrel requires 1000 nsec - one major cycle).

The slot is the execution hardware (adders, incrementors, etc.) which does the actual arithmetic and logical operations, advances the program address, etc.

Each processor has its own independent 4096 word memory which may be referenced once each major cycle (once each trip around the barrel).

There are 12 bi-directional I/O channels which are available to all ten processors.

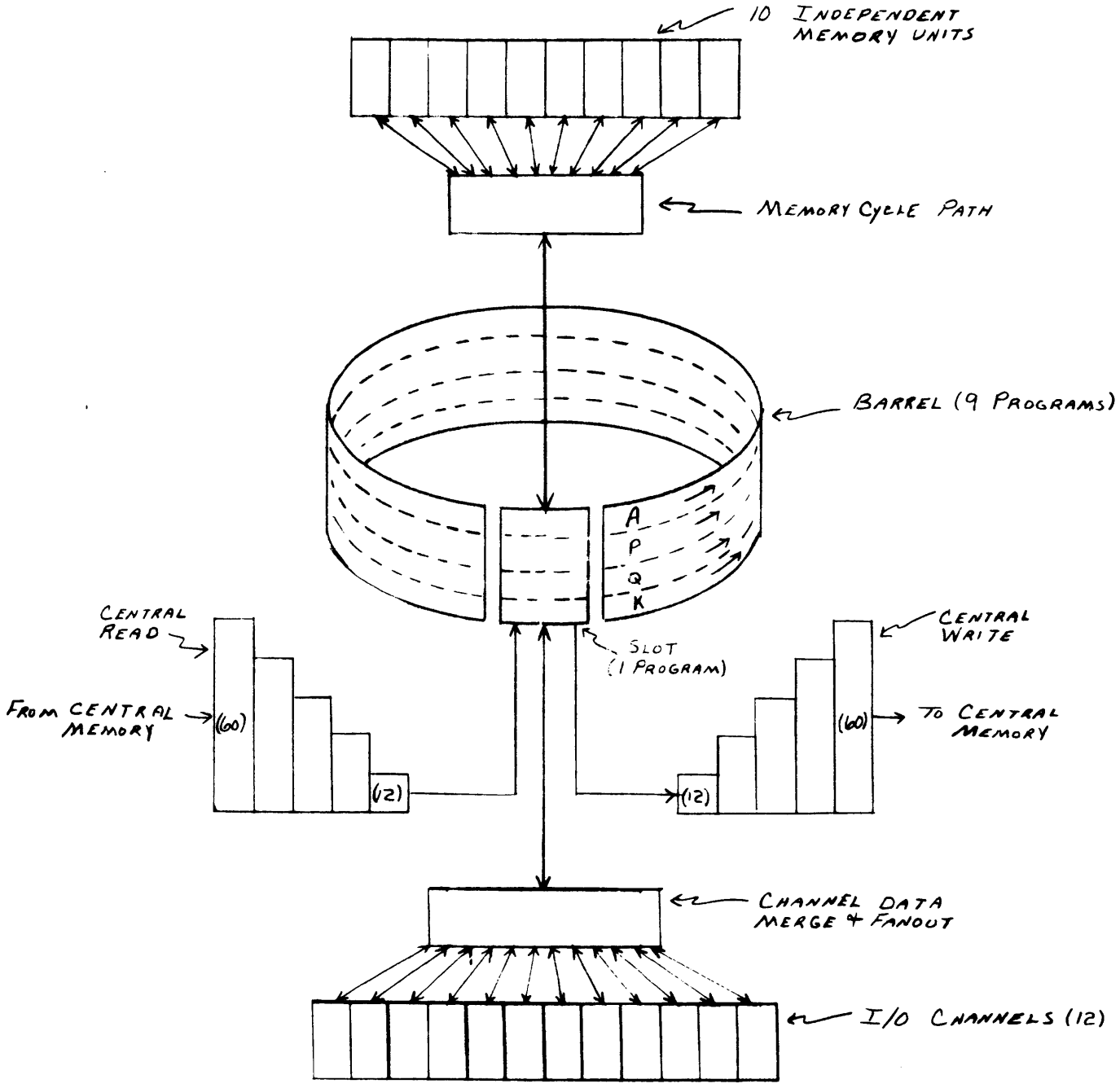


FIG. 5-1 PERIPHERAL AND CONTROL PROCESSORS

The peripheral and control processors read data from input devices, perform preliminary arithmetic and logical operations, send data and programs to central memory, assign tasks to the central processor, read central processor results from central memory, and send results to external storage (magnetic tape, disc file, etc.) or output devices (line printer, display console, etc.).

Characteristics of the peripheral and control processors are:

-- 4096 word magnetic core storage (12-bits)
   Random access, coincident current
   Major cycle - 1000 ns
   Minor cycle - 100 ns

-- 12 bi-directional input-output channels
   All channels available to all processors
   Maximum transfer rate per channel - one word/major cycle

-- Real-time clock (period 4096 major cycles)

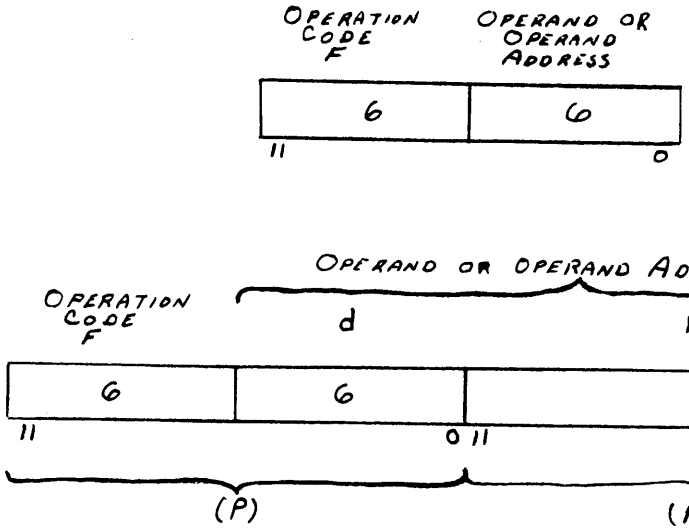
-- Instructions
   Arithmetic
   Logical
   Input-output
   Central memory read/write
   Exchange jump

-- Average instruction execution time - two major cycles

-- Indirect addressing

-- Indexed addressing

A peripheral and control processor instruction may have a 12-bit or a 24-bit format. The 12-bit format has a 6-bit operation code F and a 6-bit operand or operand address d. The 24-bit Format uses the 12-bit quantity m, which is the contents of the next program address (P+1), with d to form an 18-bit operand or operand address.



TIMING

Timing in the 6600 is controlled by a master clock located on the peripheral and control processor chassis (chassis one). Timing of the peripheral and control processors is controlled by a four-phase clock system (Fig. 5-2). Four 25 nsec wide pulses are issued each minor cycle to control movement of data and instructions. A storage sequence control system, timed by the four-phase clock, controls storage references and defines the ten peripheral and control processors.

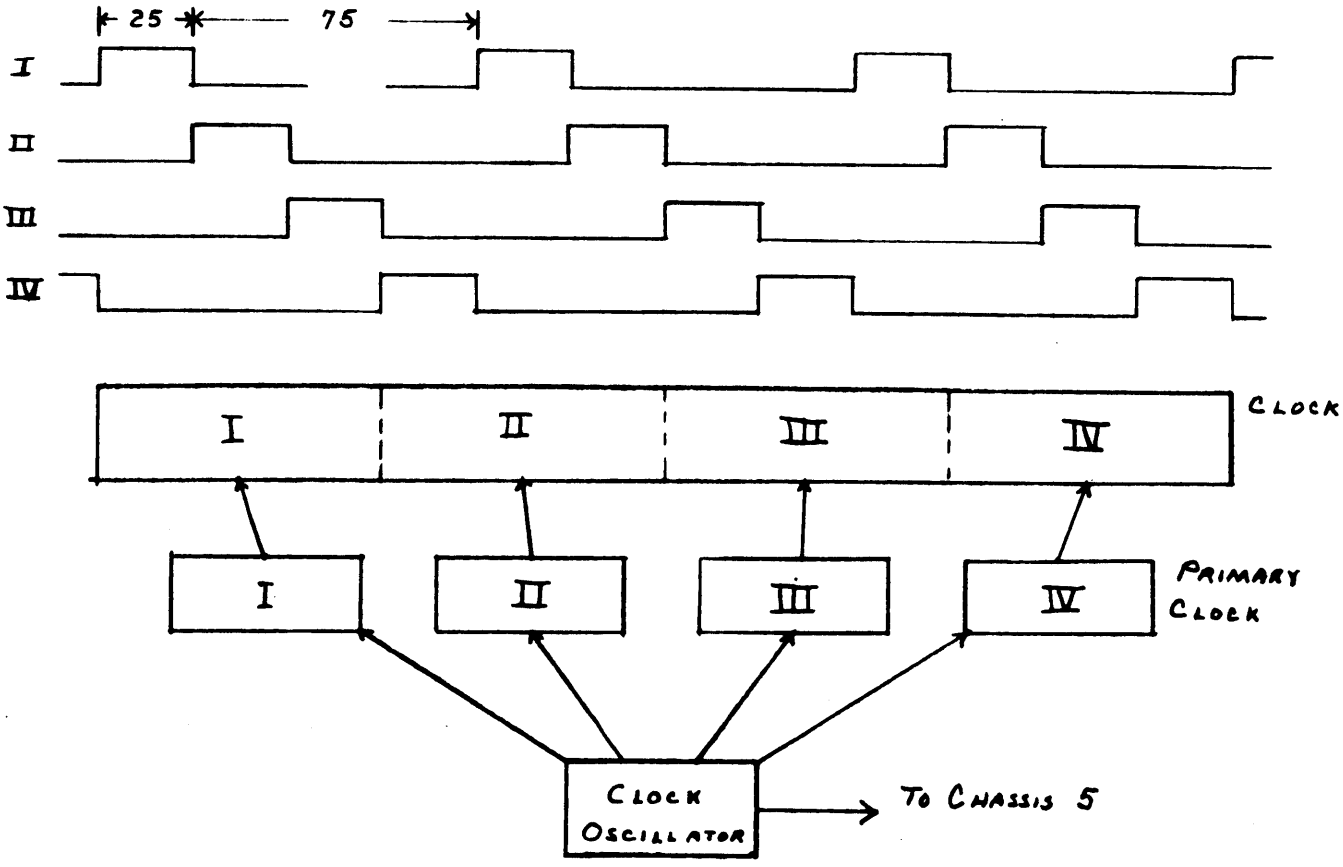


FIG. 5-2 CLOCK

MASTER CLOCK

The master clock oscillator (Fig. 5-3) consists of a TD module and a TI module. To form the 25 nsec clock pulses, a pulse from the TD is "anded" with a similar pulse which has been delayed and inverted by the TI. The result is a series of pulses (primary clock) which are fanned out through TC modules to be used as timing control. In addition to forming the clock pulses on chassis one, the master clock sends pulses to chassis five and from there to all the other chassis. On each chassis, the incoming clock pulses are used to form a clock system similar to chassis one. The clocks on all chassis are synchronized so that time 00 on any chassis is the same as time 00 on any other chassis. Some chassis require more than four discrete clock times and are not restricted to the four phases as used on chassis one.

The storage sequence control circuit, discussed in the storage section, is timed by clock pulses and issues pulses at minor cycle intervals which establish the time separation between processors.

Clock pulses for the I/O channels come from the chassis one clock system and are discussed in the I/O section.

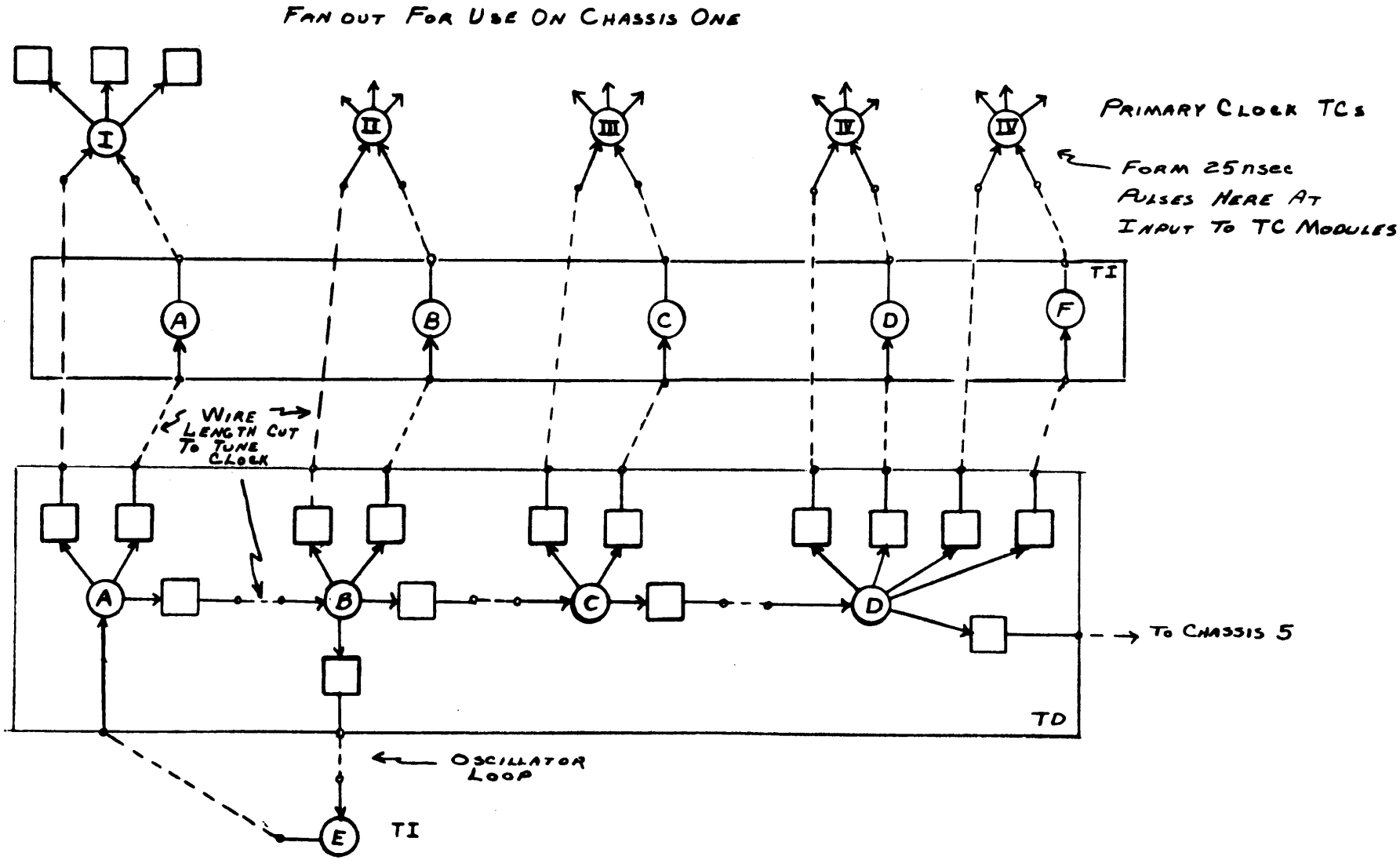


FIG. 5-3 MASTER CLOCK

BARREL

The barrel contains the A, P, Q, and K registers for each of the ten processors. The functions of these four registers in the barrel are:

A (18 bits) A holds one operand for add shift, logical and selective operations. The 18-bit quantity in A may be an arithmetic operand, central memory address, or an I/O function or data word.

P (12 bits) P is the program address register. (P) is also used as a data address in certain I/O and central instructions

Q (12 bits) Q holds the d portion of instructions or may hold a data word when d is an address.

K (9 bits) K holds the F portion of an instruction word and the trip count (The number of times an instruction has been around the barrel).

Figure 4 shows the inputs and outputs of the registers in the barrel and the sequence in which they occur.

A storage address for a particular processor is taken from the barrel six minor cycles before that processor is ready to enter the slot. This allows time for operands to be obtained from storage and entered into the barrel before slot time. Similarly, K is sent to translation network in time to be fully translated before the processor reaches the slot. The dead start inputs shown in Fig. 5-4 are discussed in the section on Dead Start.

A REGISTER

The A register in the barrel receives the result of add, shift, logical or selective operations in the slot. This quantity may be stored, returned to the slot unaltered or used to condition other operations. A is always tested to determine its sign and whether it is zero or non-zero or one. The result of these tests may be used to condition jump or other instructions. The quantity in A may be a full 18 bits as in the case of central instructions or A may contain a 12-bit peripheral word in which case the upper bits are zero.

The connections to A in the barrel are:

Outputs

A→M - (A) may be sent as a data or function word on one of the I/O channels.

A→ Central Address Register - (A) is the central memory address in central read and write and exchange jump instructions.

A→Y - For a store instruction (A) is sent to Y and then to storage.

A→ Translation Networks

Inputs

X→A - The content of the central program address register is sent to the peripheral X register every minor cycle. A 27 instruction sends (X) to A and enables a peripheral and control processor to monitor the progress of the central program.

R→A - An input to A instruction gates a word from an I/O channel into A.

Fd→A - A data word from storage is entered into A by the Fd A path.

A→A - When the quantity in A is to be returned to the slot unaltered the A→A gate is enabled.

P REGISTER

P holds the program address and is not changed in the barrel except in the case of Dead Start. As shown in Fig. 5-4, (P) is sent to a storage unit before slot time. This allows time to read a word from storage and make it available at slot time. (P) is sent to the G register which feeds all ten storage address or S registers (Fig. 5-15). When a jump is called for, P is sent to Q. Q is then altered by the Q adder in the slot and the new address returned to P at the first stage of the barrel.

Q REGISTER

Q holds the d portion of an instruction and has several outputs to translation networks to translate channel selections for I/O instructions.

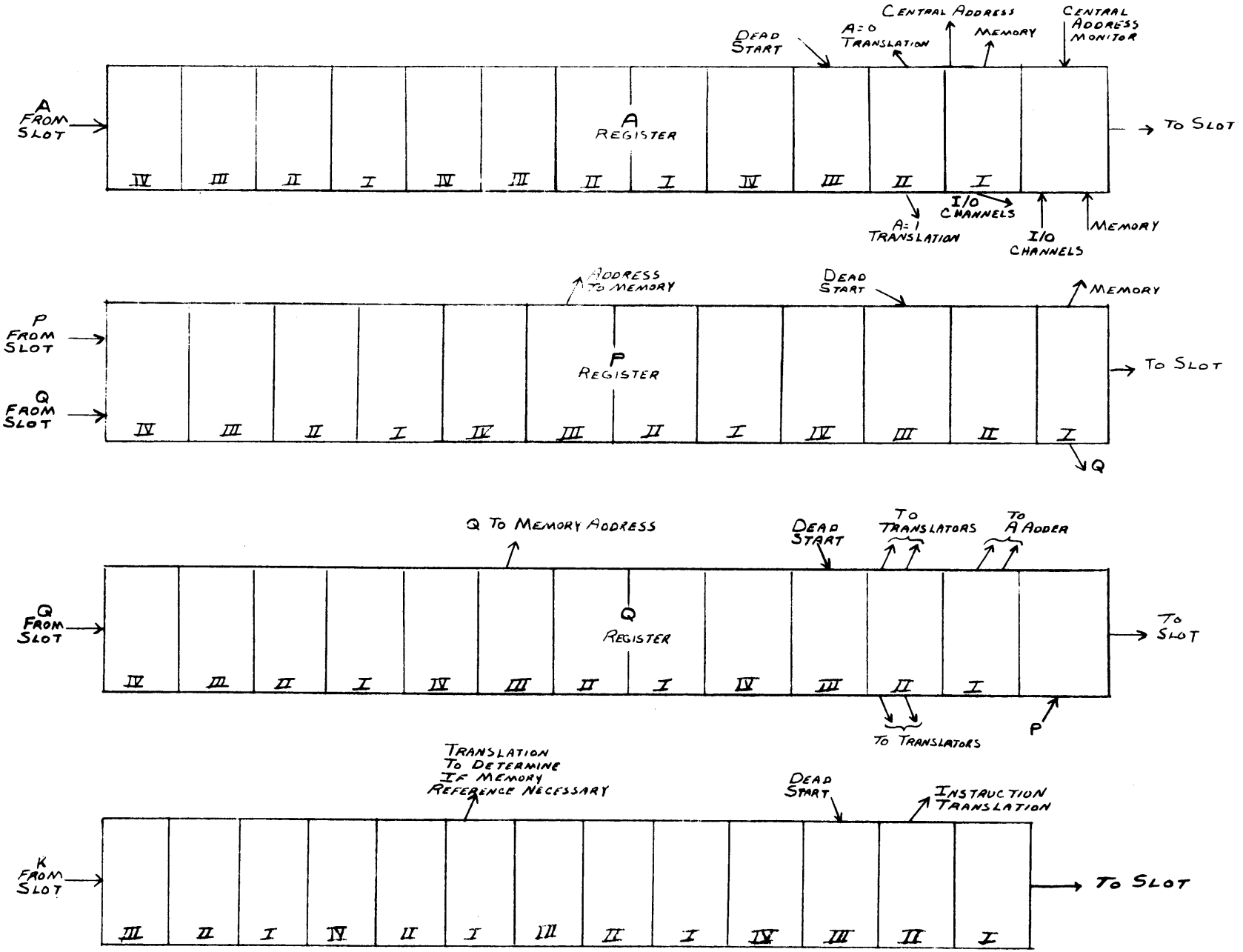


FIG 5-4 BARREL INPUTS AND OUTPUTS

When d is an address, (Q) is sent from the slot to P in the barrel and the word obtained from that address is entered into Q in the slot. When a jump is called for, the quantity in Q is added to or subtracted from (P) in the Q adder and the result sent to P. When an instruction calls for an 18-bit operand, the lower six bits of Q are sent to the upper six bits of A to form the 18-bit quantity dm.

K REGISTER

K holds the F portion of an instruction word and a 3-bit trip count which sequences the execution of an instruction. K is translated at two different times during a trip around the barrel; first to determine if a storage reference is needed and then to provide the proper commands at the slot. During the barrel trip in which a new instruction is being read from storage, a translation of K=00X enables translations from Fd in the storage cycle path to be used in place of K translations. This eliminates the need for a separate "Read Next Instruction" trip through the barrel and allows some instructions to be read from storage and executed all in one trip. The K=00X translation arises from the fact that K is cleared at the end of each instruction.

SLOT

The slot contains the execution hardware for A, P, Q, and K and each processor is allowed one minor cycle in the slot every major cycle. Included in the slot are:

A	Adder
	Shift Network
	Logical Circuits
	Selective Circuits
P	Incrementor
	Inputs from P or Q in the barrel
Q	Adder
	Input Path from Fd

K 3-bit Trip Counter
Input from F
K = 340 Gate

As A, P, Q, and K enter the slot (Fig. 5-5), K translations (started earlier in the barrel) become available and a portion (or all) of an instruction is executed. The results are gated back into the barrel to be stored, used again, or sent to I/O equipment.

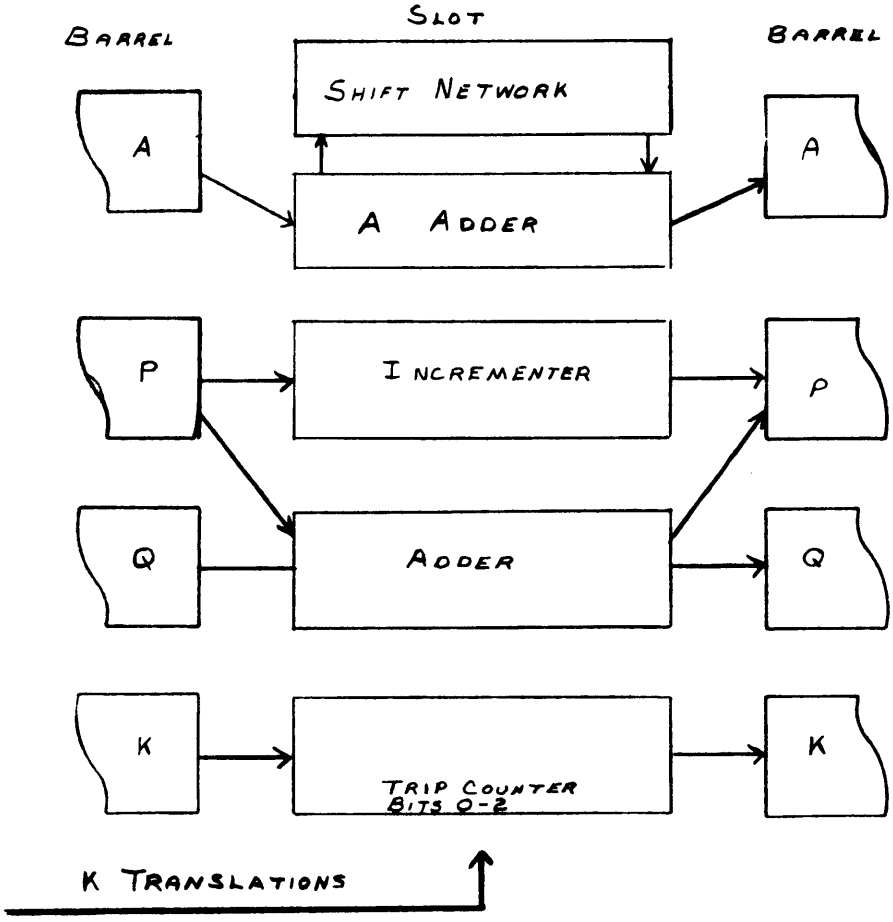


FIG. 5-5 BLOCK DIAGRAM OF SLOT

A ADDER

The A Adder (Fig. 5-6) is used to execute add, subtract, selective clear, logical product, and logical difference instructions. Parts of the A adder are also used to enter a word into the shift network and gate the result back to the barrel. The quantity in A in the barrel is always complemented when it enters the slot. When no operation on A is called for, (A) is complemented, enters the A adder, is added to zero, and the result is re-complemented at the output. The Add gate on the Qd modules (Fig. 5-6) is always enabled except when Selective Clear, Logical Product, or Shift commands are enabled.

Add

For an add instruction (A) is complemented and entered into the A input register. The second operand is also inverted before entering the B input register. The two quantities in the input registers, taken as positive, are added and the sum is re-complemented as it is gated out of the QD modules to the barrel.

Subtract

For subtract instructions, the minuend, (A), is complemented as it enters the adder. The subtrahend is entered into B without being complemented and the two quantities are added as in an add instruction.

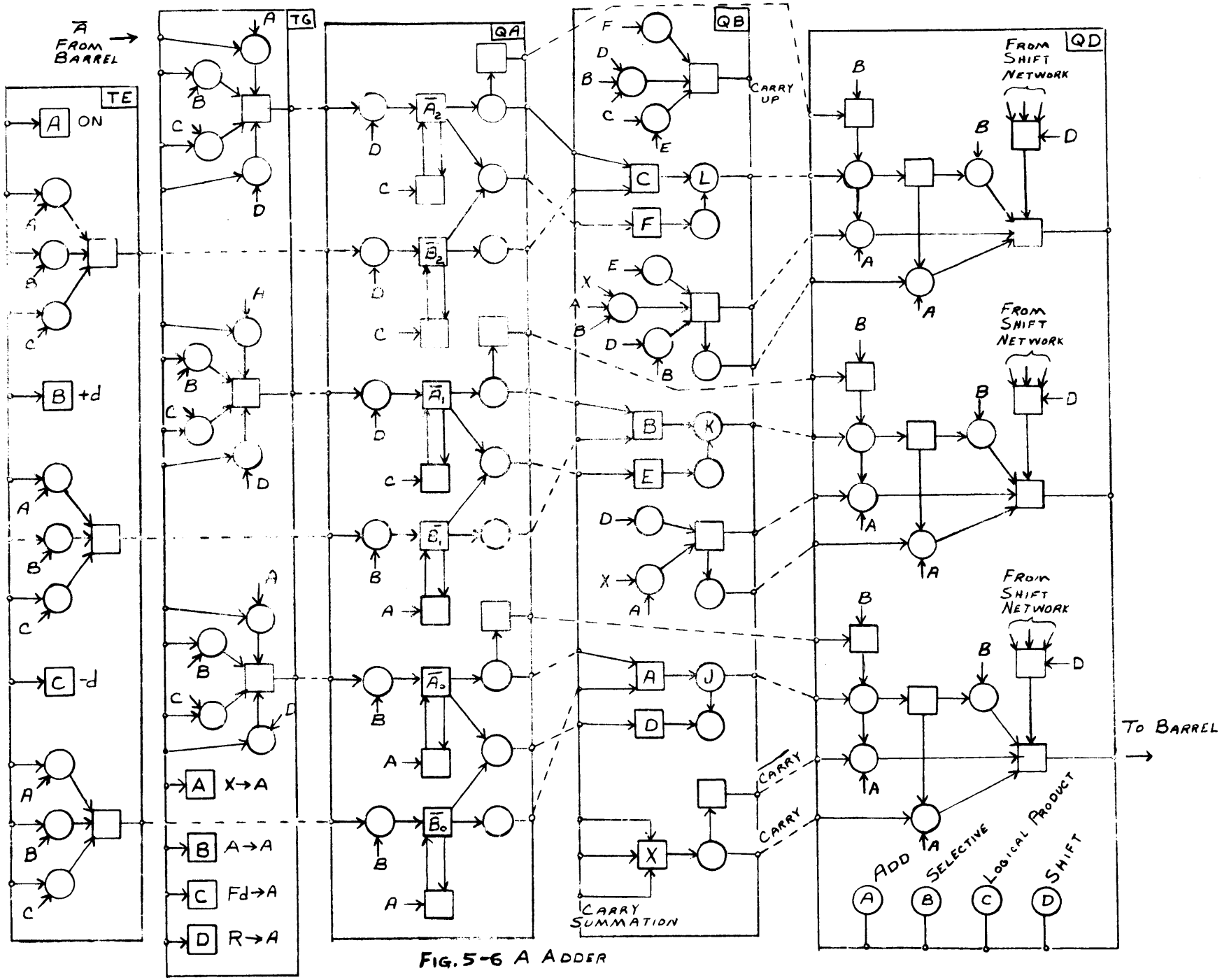


FIG. 5-6 A ADDER

Selective Clear

For selective clear the complement of A and the true value of d are entered into the adder and both the selective and the logical product gates are enabled.

Logical Product

For logical product instructions both A and d (or dm) are complemented before entering the adder and both the logical product and the selective gates are enabled.

Logical Difference

For logical difference instructions the complement of A and the true value of the second operand enter the adder and only the selective gate is enabled.

SHIFT NETWORK

The shift instruction (10) provides for shifting the number in A up to 31 places left or right. Left shift is circular with the high order bits re-entering A at the low-order end. Right shift is end-off with low-order bits discarded as they shift out of the A register and with no sign extension. Thus, a left shift of 18 is equivalent to no shift, and a right shift of 18 clears the A register.

The shift network (Fig. 5-7) is a static network. The content of A enters the register at IV time, each bit follows a path established by static translations of the 6-bit shift count in d, and the result re-enters A in the barrel at the next IV time. The input to the shift network comes from the A input register in the A adder (the content of that register, $\overline{A}$, is complemented before entering shift register). The output of the shift network is gated back to the barrel by way of the output modules (QD) of the A adder. Note that the quantity in A is always shifted but the result is gated to the barrel only when the current instruction is a shift.

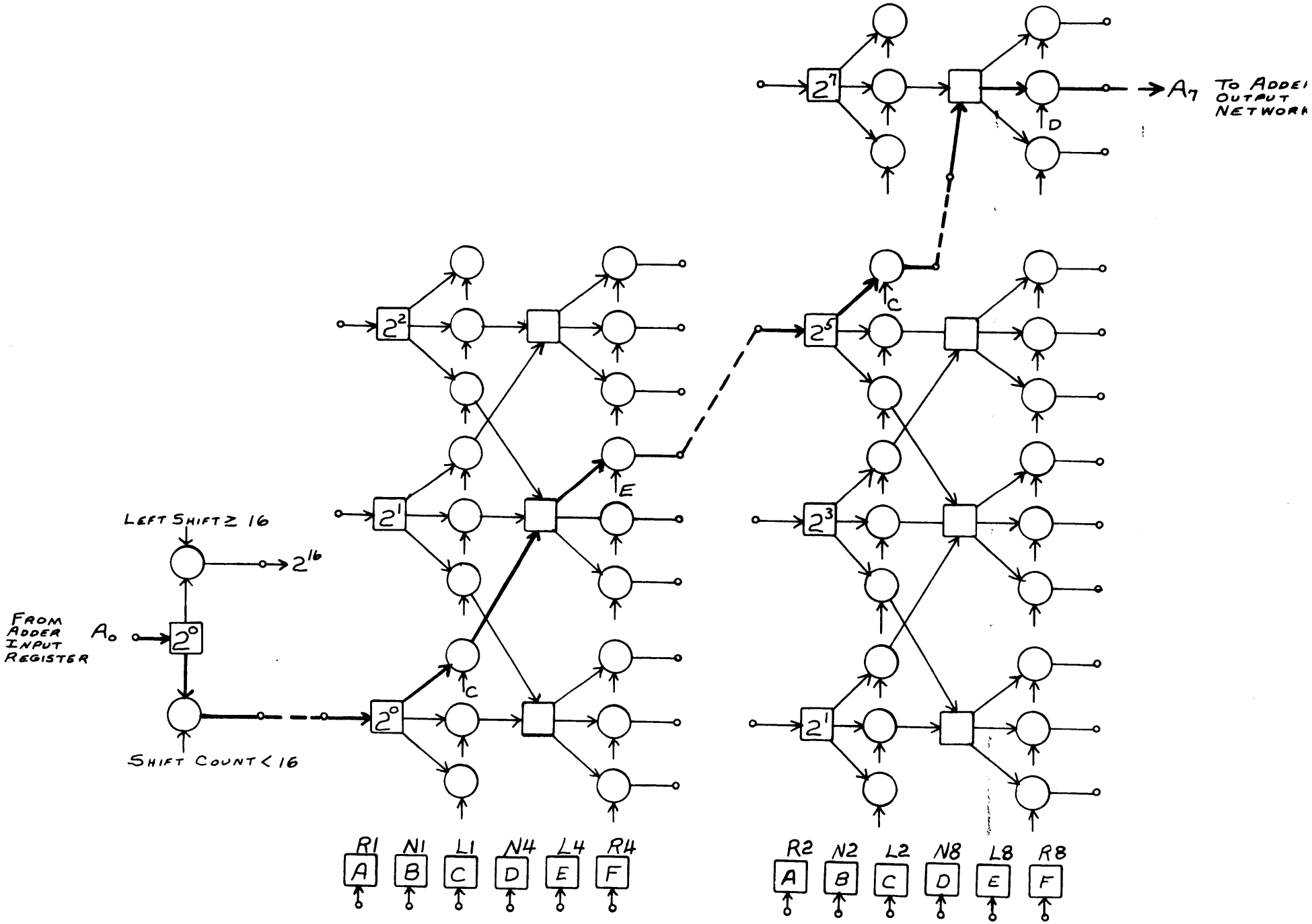


FIG. 5-7 SHIFT NETWORK (LEFT SHIFT 6)

If d is positive ($00\text{-}37_8$) the shift is left and the shift count is the content of d. If d is negative ($40\text{-}77_8$) the shift is right and the shift count is the complement of the number in d.

At the first stage of the shift network d_4 and d_5 are tested to determine whether the shift is greater or less than 16 and whether it is right or left (Fig. 5-8). If the shift is 16 or greater a shift of 16 is made at this point and the result then enters the rest of the shift network.

Bits d_0 - d_3 are tested with d_5 as shown in Fig.5-9 to set up paths through the rest of the network. Fig. 5-7 shows the path of bit 0 for a left shift of six places.

P REGISTER

The P register in the slot is an incrementor (Fig. 5-10) which can advance (P) by one or send (P) back to the barrel unchanged. When P is to be changed by more than one the arithmetic is done in the Q adder and the result entered into the first FF of P in the barrel.

Q REGISTER

Q in the slot is a 12-bit adder (Fig. 5-11) which adds and subtracts in the same way as the A adder. For jump instructions P is sent to Q before the slot. The content of Q (now the program address) is complemented and sent to the Q input register and the complement of d is sent to the lower six bits of the H input register. If d is positive (01-37), the upper six bits of H are set to 77 by the 00 → H gate; if d is negative, (40-77), the upper six bits of H are set to 00 (no input to H gate is enabled).

K REGISTER

K in the slot consists of a 3-bit trip counter for the lower three bits (Fig. 5-12) and a fan-in for the upper six bits. The advance K signal to the trip counter is enabled by instruction translations. For some instructions the advance K signal is controlled by signals which indicate

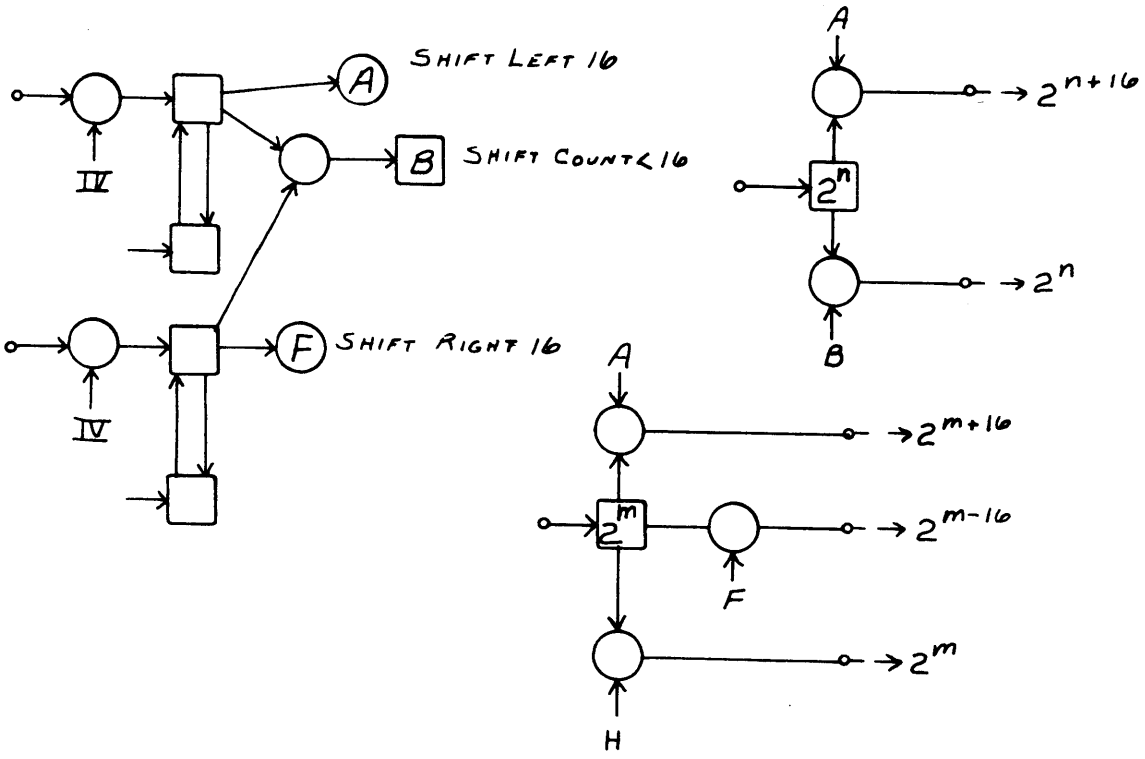


FIG. 5-8 TEST FOR SHIFT COUNT > 16

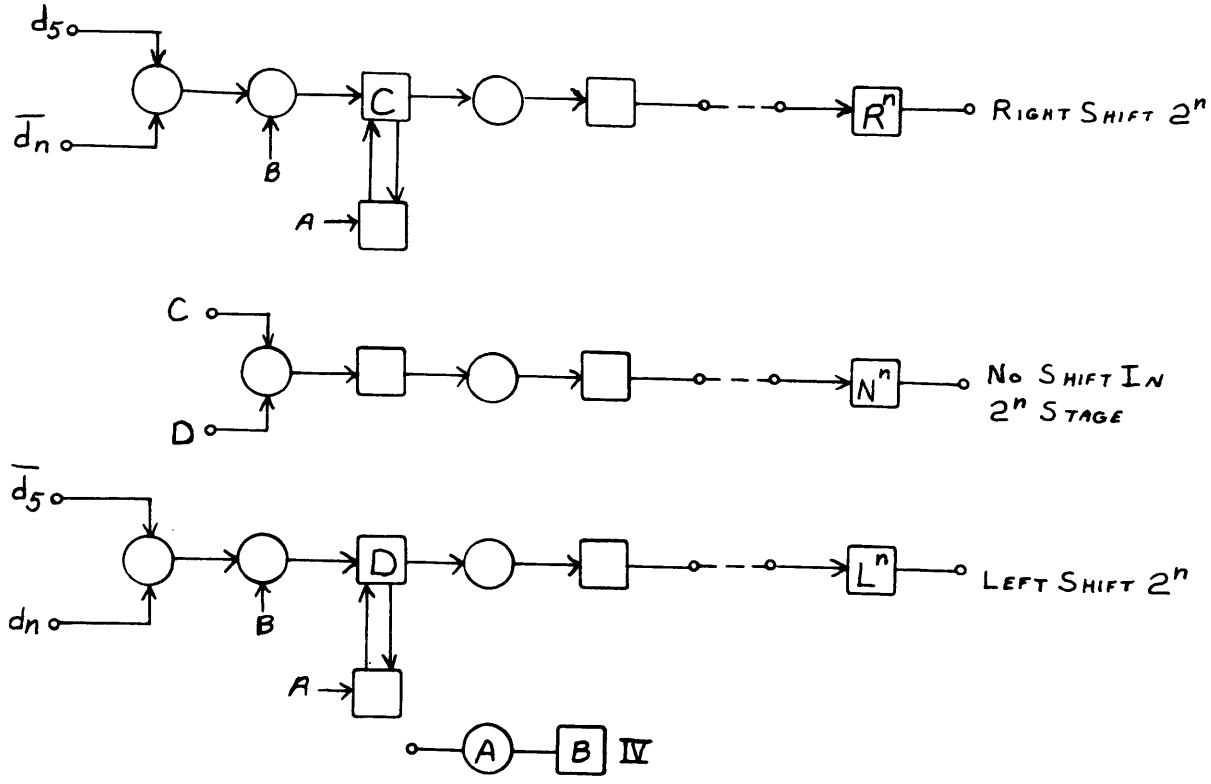


FIG. 5-9. TEST FOR RIGHT OR LEFT SHIFT

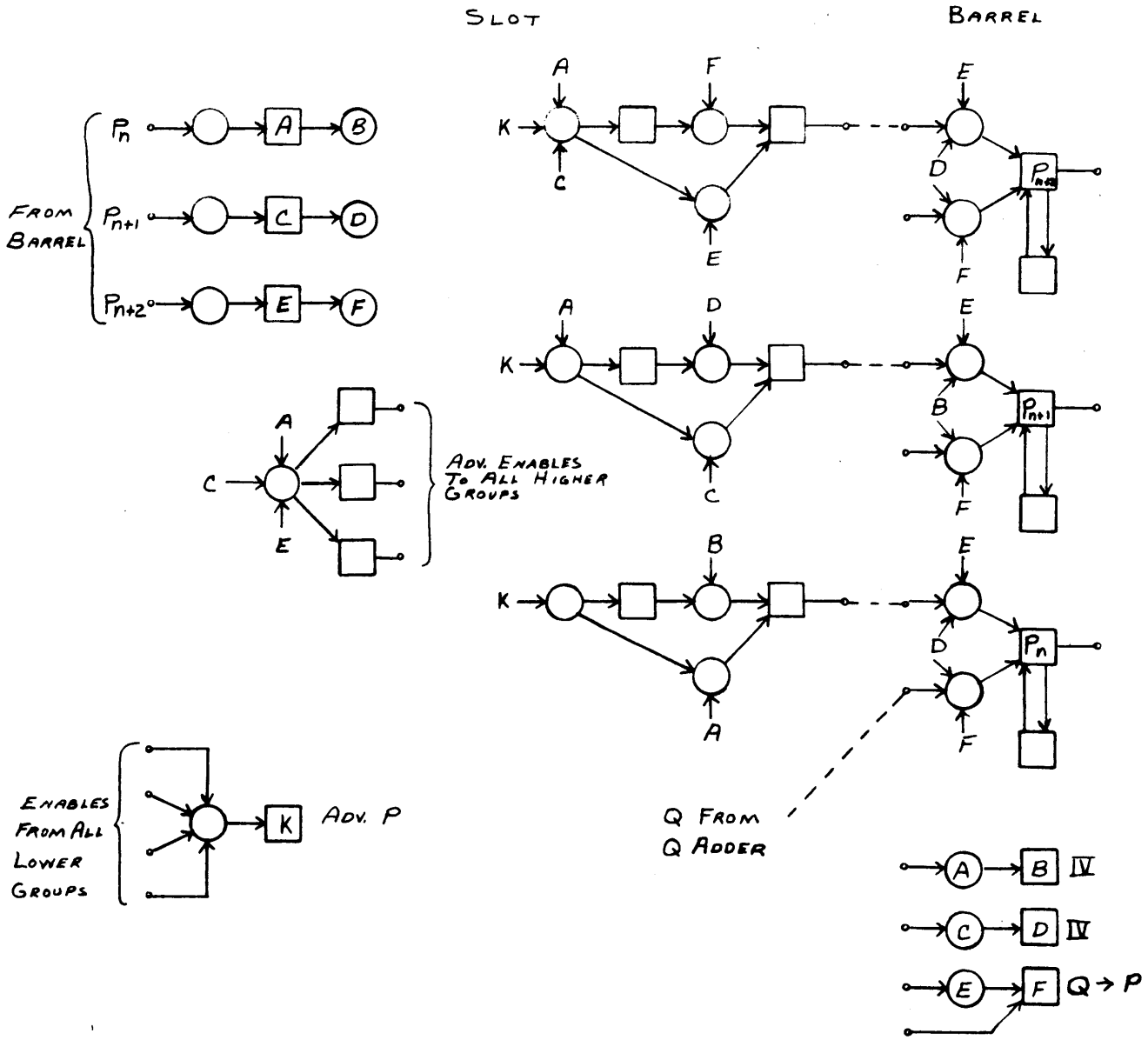


FIG. 5-10. P INCREMENTOR

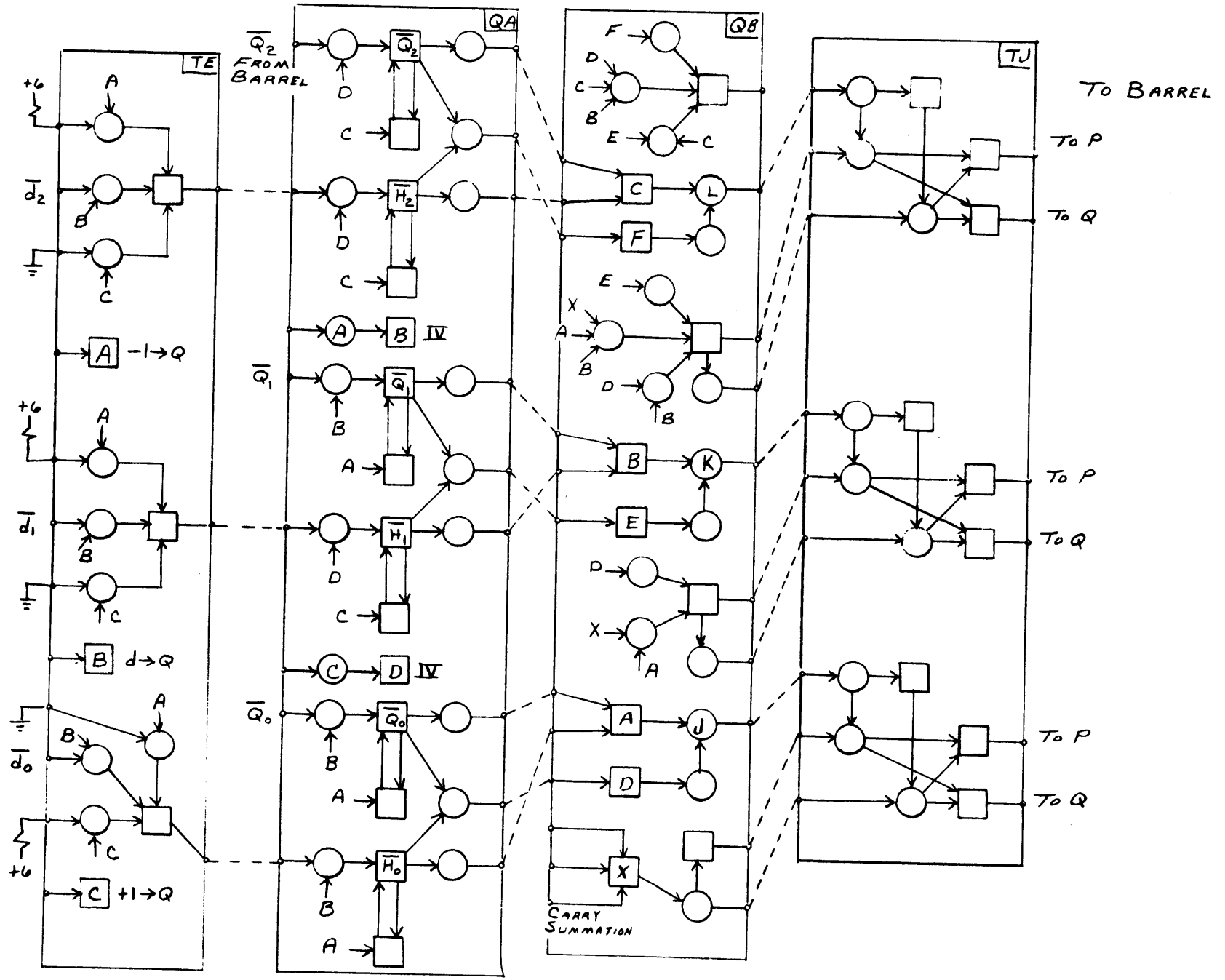


FIG. 5-11. Q ADDER

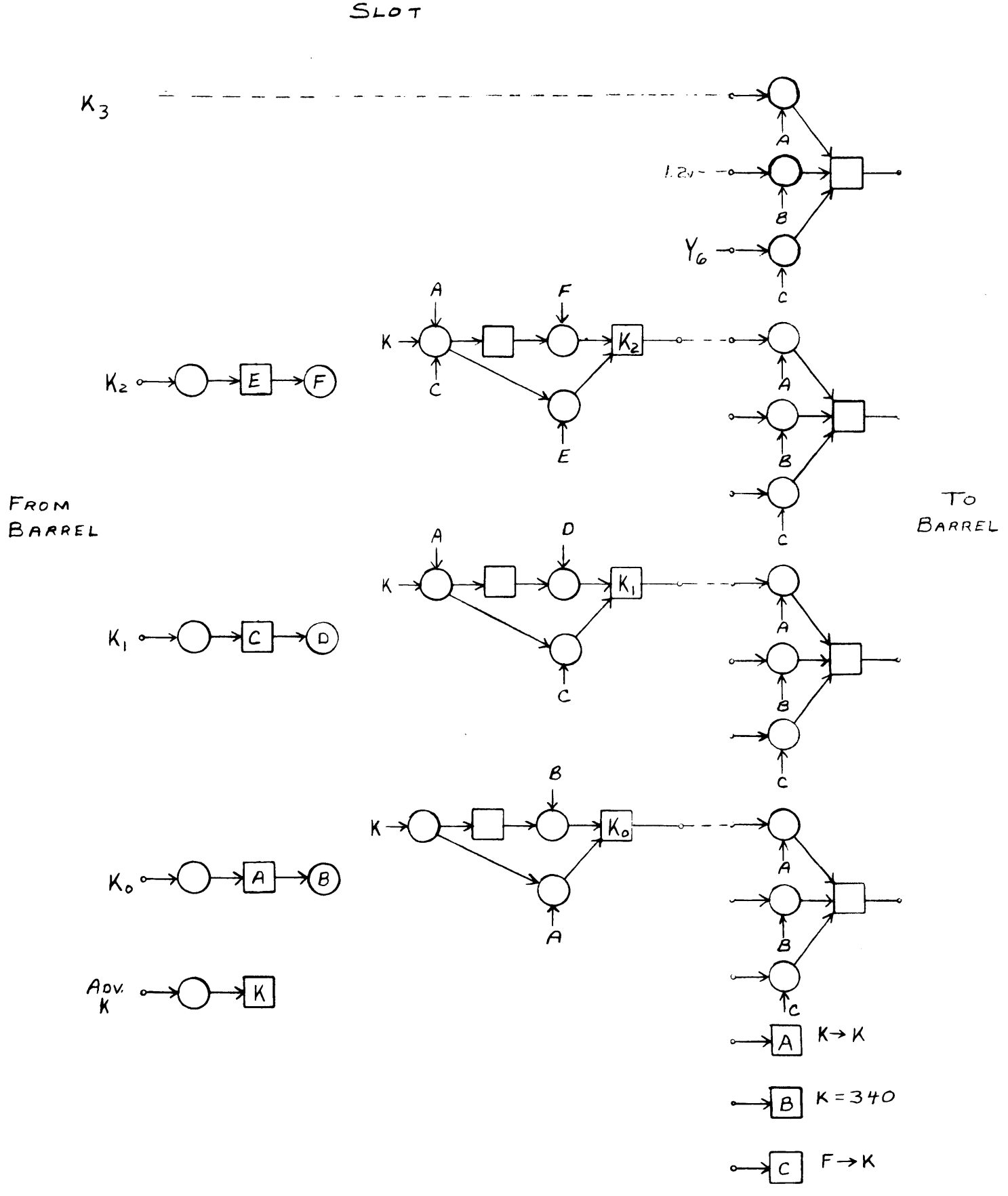


FIG. 5-12 K TRIP COUNTER

status i.e., the 5 X 0 trip is skipped by all 5X instructions if d = 0, and, when K = 732 K is advanced only if the I/O channel is empty and active and A = 1.

The three-bit trip count controls the sequence of operations for each instruction and is sometimes changed by gates other than the trip counter. For instance, for a central write instruction (63) K is changed from 637 to 633 to repeat the sequence of commands and send another word. When a 63 instruction is completed K is changed from 637 to 733 to finalize the instruction and obtain the next instruction from storage.

The fan-in to the upper six bits of K allows the instruction code F, to be entered into K from storage. The K→ K path allows another trip around the barrel for the present instruction. The path K = 340 is used by replace instructions which automatically use the store instruction 34 to accomplish the store portion of the replace instructions.

MEMORY

Each of the ten peripheral and control processors has its own independent core-storage unit with a capacity of 4096 12-bit words. Each has its own address register (S), sense amplifiers, and restore register (Z). However, the ten storage units share a common memory cycle path and common paths to and from the barrel.

Each peripheral and control processor may make one memory reference each major cycle. When no memory reference is called for by the current instruction, address 0000 is read and restored.

STORAGE SEQUENCE CONTROL

Timing for memory references is controlled by storage sequence control. Storage sequence control (Fig. 5-13) consists of a timing chain of FFs gated by clock pulses. As a "one" passes down the chain each FF is set for one minor

cycle during which it issues commands to the storage logic. This chain reinitiates itself after each cycle and runs continuously. One memory reference is initiated each minor cycle. Since a memory reference requires longer than a minor cycle, the references overlap as shown in Fig. 5-14.

The stages of storage sequence control are numbered according to the processor for which they initiate a memory reference. The commands issued by the first half of a typical stage, a (Fig. 5-13) are:

G → S, Storage a
Clear Z, Storage a + 1
Set Z, Storage a + 5
Enable Sense, Storage a + 7

The second half of stage a issues the commands:

Read, a
Write, a + 5
Stop Read, a + 6
Stop Write, a + 1

These commands and other signals from storage sequence control define and separate the peripheral and control processors.

The reset circuit which reinitiates storage sequence control senses whether stages 0-8 are set; if not stage 0 is reinitiated, normally just after stage 9 has issued its commands.

A memory reference is initiated six minor cycles before a processor reaches the slot so that information from memory is available at slot time. Thus a memory reference for processor 0 (storage 0) is initiated at about the same time that processor 4 enters the slot.

MEMORY CYCLE PATH

The common memory-cycle path used by all processors receives data from the memories via the sense merge shown in Fig. 5-15. Inputs to the sense merge from the sense amplifiers are a logical "1" (0.2v) when sense is not enabled. When a processor's sense amplifiers are enabled, the

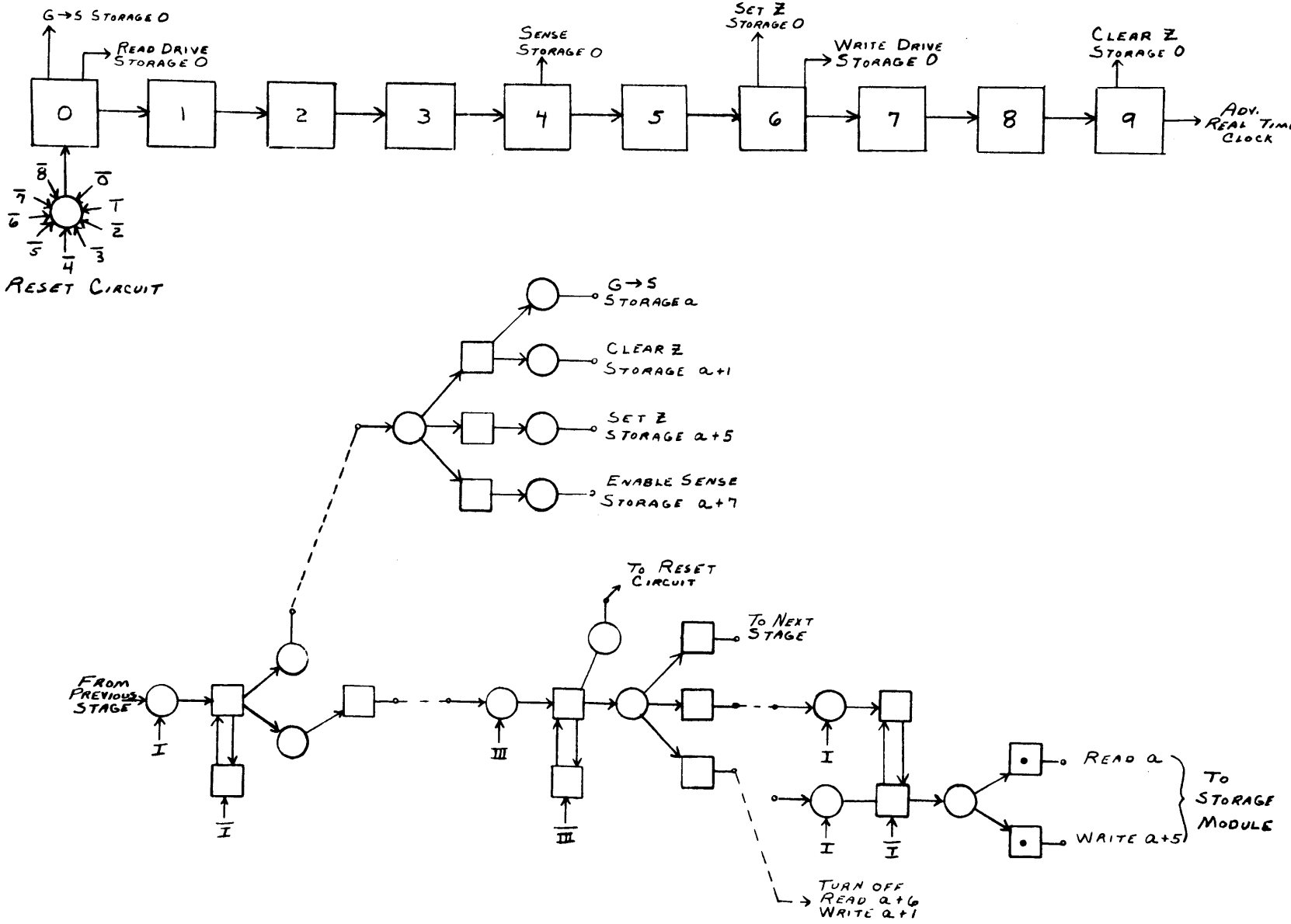


FIG. 5-13 STORAGE SEQUENCE CONTROL WITH TYPICAL STAGE a

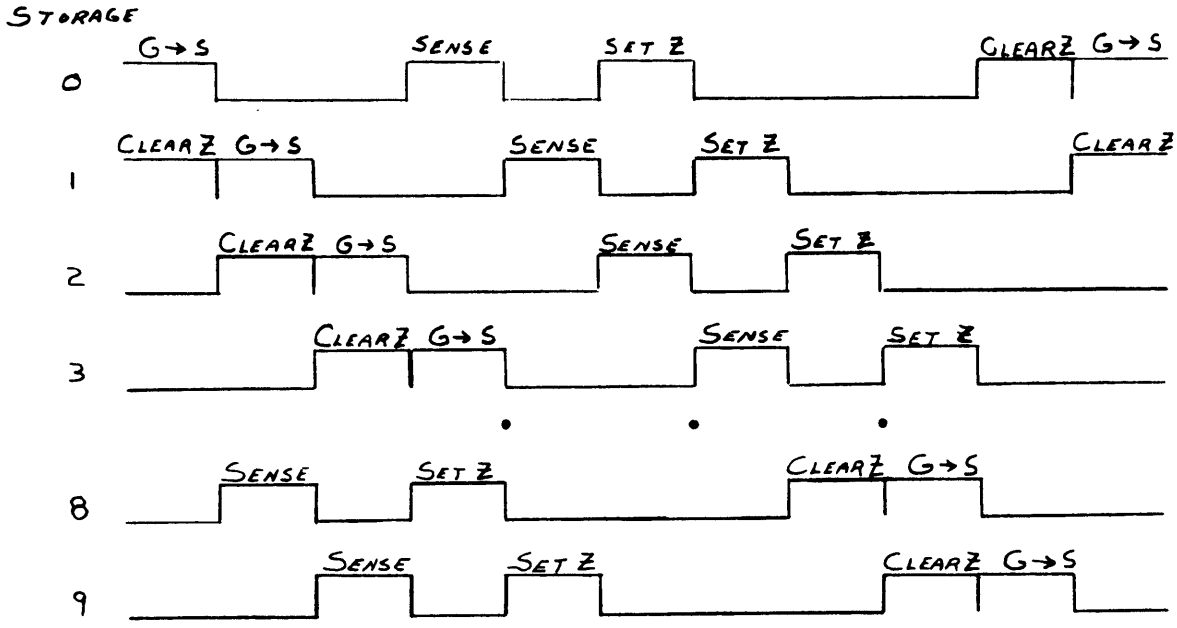


FIG. 5-14 STORAGE REFERENCE TIMING

output is allowed to go to 1.2v for a sensed "0". The "and" combination of logical "1s" from unselected processors, even or odd sense enable, and "1" bits from the selected processor's sense amplifiers sets the word from memory in the Fd register in the memory cycle path.

The memory cycle path sends information to the barrel, I/O channels, translators and central write pyramid and receives information from the barrel, central read pyramid, and I/O channels. Outputs from Fd in the memory cycle path are translated and used to form commands when K = 00X (read next instruction trip).

Information in the memory-cycle path (either the read word or a new word) is fanned out from the Y register to the ten Z registers. The set Z signal from storage sequence control gates the complement of the word to be stored into the proper Z register.

A description of the circuits in the storage modules is contained in Section IV.

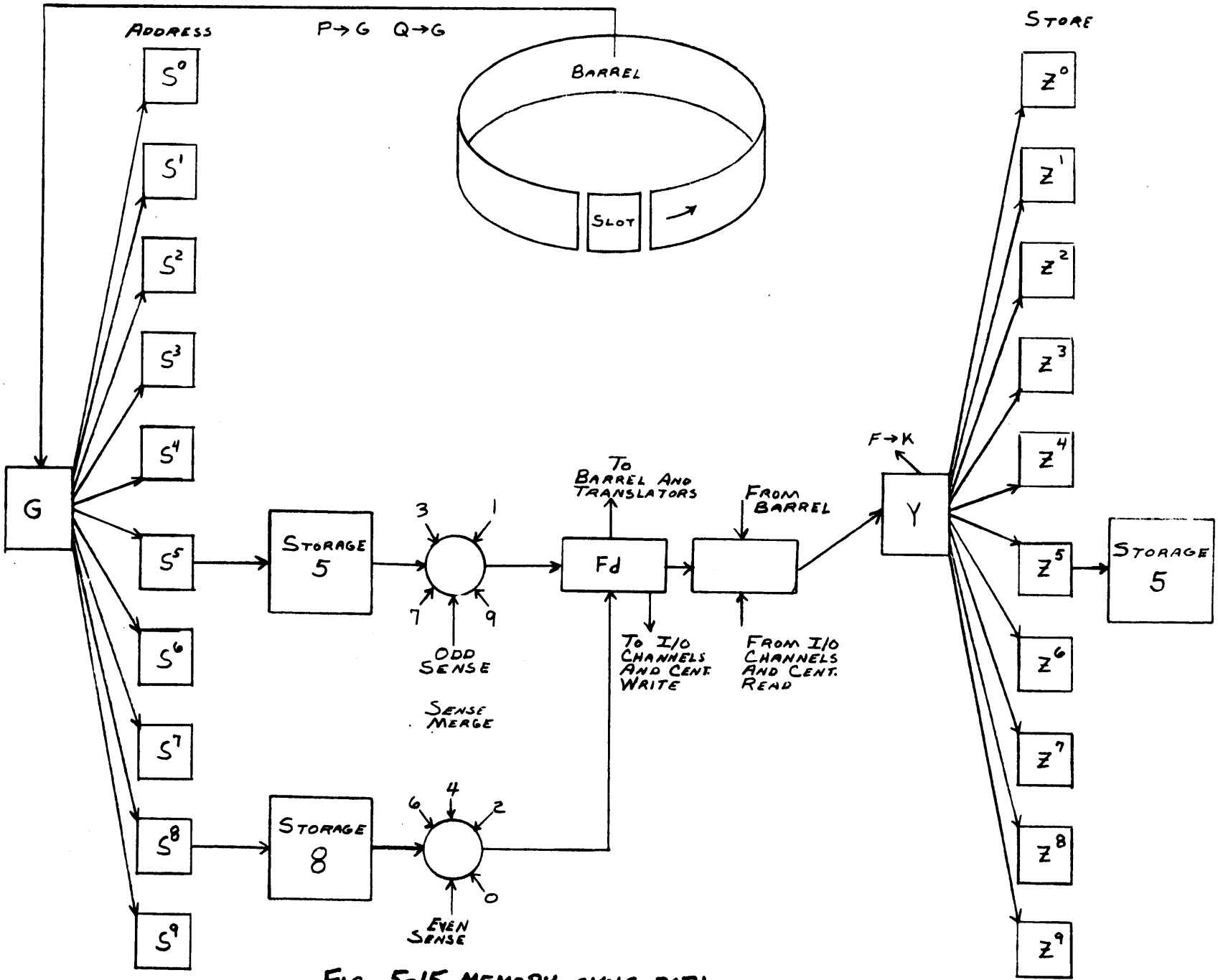


FIG. 5-15 MEMORY CYCLE PATH

INPUT/OUTPUT

Each of the 12 independent I/O channels can handle 12-bit words at a maximum rate of one word every major cycle (1mc rate). Each channel has an Active/Inactive FF and a Full Empty FF (Fig. 5-16) which indicates channel status to the processors. Any channel may be used by an processor but the external equipment assigned to a channel is wired in and may be assigned to another channel only by changing cable connections. Figure 17 shows the channel data fanout and merge for one bit.

The lines of an I/O channel are:

Input	Output
Data or Status Reply (12 bits)	Data or Function word (12 bits)
Active	Active
Inactive (Disoonnect)	Inactive
Full	Full
Empty	Empty
	MC

In addition two clock signals are available to external equipment: a 1 mc clock and a 10 mc clock. The clock pulses are 25 nsec wide as are all data and control signals (except master clear). A synchronizer for each external equipment (or group) changes the 6600 I/O pulse signals into the signals required by I/O devices and changes input signals to the pulses required by the I/O channels. Synchronizers are described in a separate manual.

An I/O channel may be used for communication between processors if it is selected for input by one processor and for output by another.

The status of I/O channels may be sensed by instructions 64-67: jump to m if channel d active, etc.

Instructions and signals common to both input and output are:

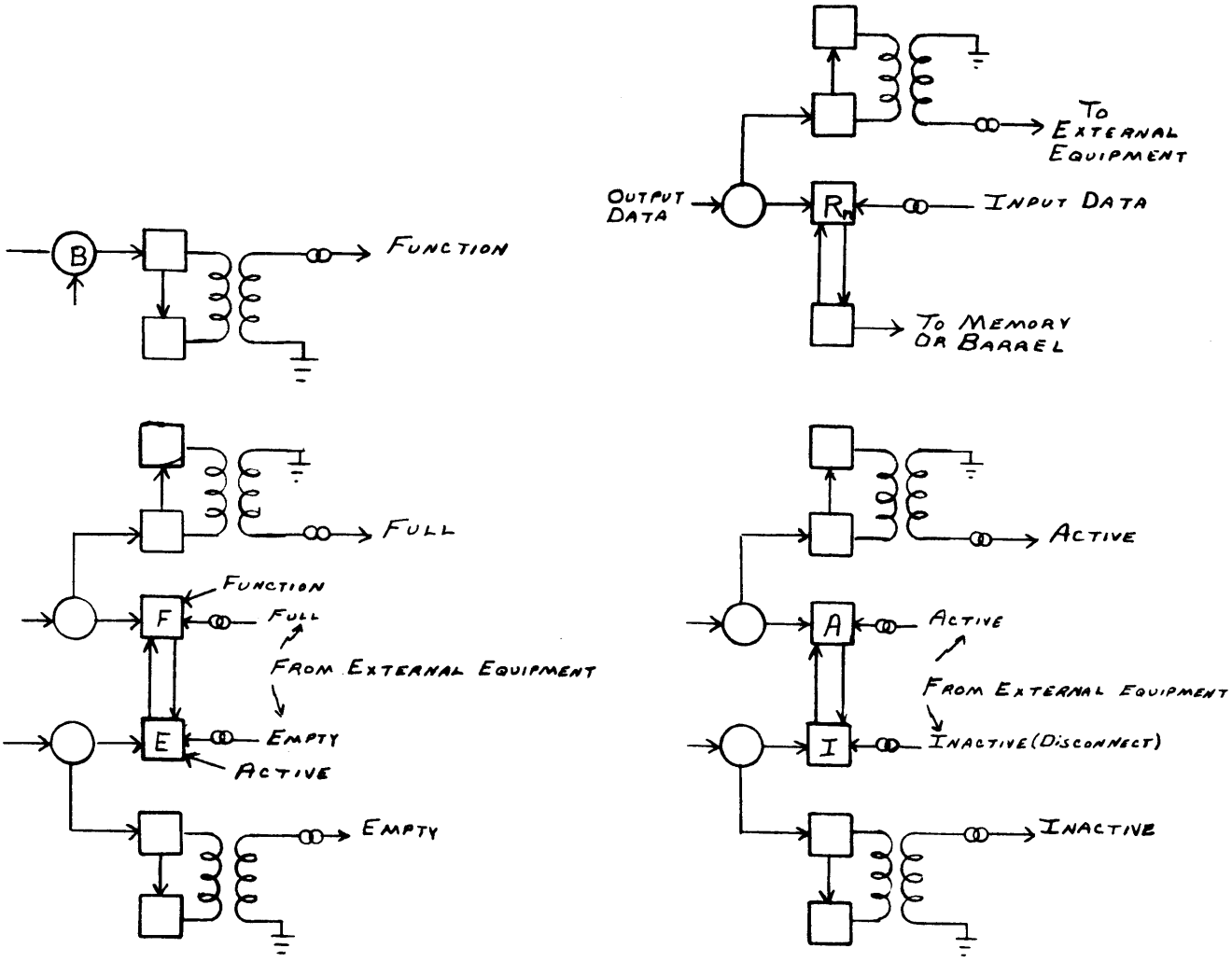


FIG. 5-16 CONTROL FLAGS AND DATA TRANSMISSION

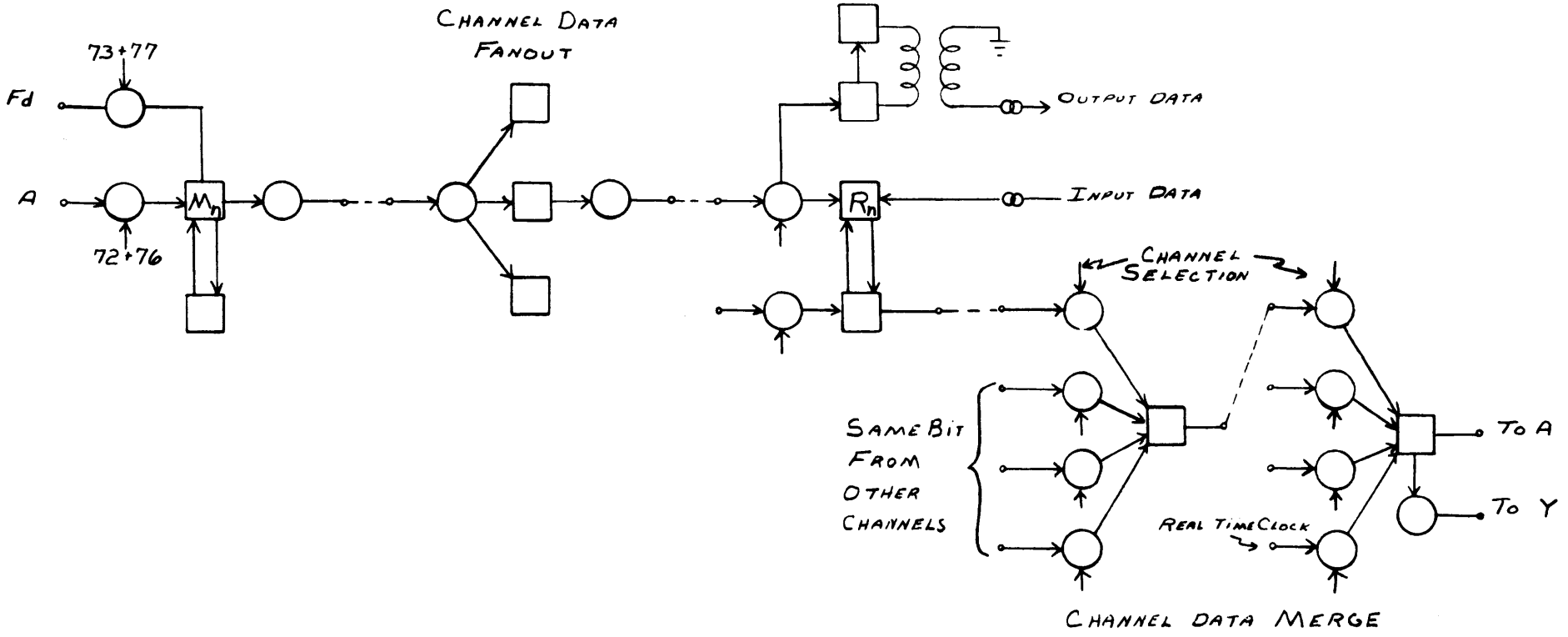


FIG. 5-17 SHARED PATH OF I/O WORDS

Master Clear

A Master Clear (MC) signal is generated only by the Dead Start! circuit. MC removes all equipment selections (except Dead Start synchronizer) and sets all channels to the Active and Empty condition (ready for input). MC is a 1 usec pulse which is repeated every 4,096 ms while the Dead Start switch is on.

Disconnect (75)

A disconnect instruction clears the channel Active FF if it is set and sends an inactive pulse to the equipment on that channel. If a disconnect instruction is given for a channel which is already inactive, the processor which issued the disconnect will "hang up" and will not be able to continue to another instruction until the channel is activated by another processor (or by MC).

Function (76 or 77)

A function instruction sends a 12-bit function code (from A or Fd) on the data lines and sends a Function signal. It also sets the Active and Full FFs for the channel but does not send Active and Full pulses. Upon receipt of the function code, the external equipment sends an Inactive (disconnect) signal which clears the Active FF which in turn clears the Full FF. If a Function instruction is given for an active channel, the processor will hang up until the channel is deactivated.

Activate (74)

An Activate instruction sends an Active signal on the channel and sets the Active FF if the channel is inactive. If an Activate instruction is given for a channel which is already active, the processor which issued the instruction will "hang up" until the channel is inactivated by another processor or by an Inactive (disconnect) signal from an external equipment on the channel.

INPUT

An Input instruction (70, 71) may call for an input of one word to A (70) or a block of words to memory (71). A block may be as small as one word. For either instruction, the control signals sent to and received from external equipment are similar. A 70X or 712 translation clears the Full/Empty FF and sends an Empty signal on the designated channel. When the input device sends a word on the channel it also sends a Full signal which sets the Full/Empty FF.

Full and Active enables the processor to store the input word or send it to A. A 70 instruction sends the input word to A and exits to the next instruction. A 71 instruction stores the word in memory, reduces the count in A by one, and sends another Empty on the channel. A 71 instruction continues to read words until the word count in A is reduced to zero. The processor sends one more word when it senses that the previous read operation reduced (A) to 1.

OUTPUT

Output instructions (72 and 73) transmit one word from A or a block of words from memory on an I/O channel. A block may be as small as one word.

An instruction translation of 72X or 732 and Empty and Active sets the Full/Empty FF and sends a Full pulse and a data word on the channel. When the external equipment accepts the word it sends an Empty pulse which clears the Full/Empty FF. Empty and Active enables the processor to continue the instruction. A 72 instruction does not wait for the Empty but exits to the next instruction after sending one word.

When the Empty is returned on a channel which is being used for a 73 instruction, the processor reduces the word count in A by one and sends another word. A 73 continues to send words accompanied by Full pulses

until the required number has been sent. When the processor senses that A has been reduced to one, it sends one more word, obtains the address of the next instruction and continues with its program.

COMMUNICATION WITH CENTRAL MEMORY AND CENTRAL PROCESSOR

The peripheral and control processors may communicate with the central processor and central memory in several ways. They may read the central processor's program address, tell the central processor to jump to given central memory address for its next instruction, or read from or write into central memory.

CENTRAL PROGRAM MONITOR

The 18-bit central processor program address is sent to the central program monitor register (Fig. 5-18) on chassis one every minor cycle. A Read Program Address instruction (27) sends the central address to the A register. Thus the progress of a central program may be monitored by any peripheral and control processor.

Exchange Jump, Central Read and Central Write instructions all use the content of A as a central memory address. (A) is unconditionally sent to address control in the central processor every minor cycle (Fig. 5-19). This quantity is recognized and used as a central memory address only if accompanied by a Central Read, Central Write or Exchange Jump signal.

A Central Busy FF (Fig. 5-20) indicates when a reference to central is in progress. A Central Busy condition prevents initiating a central reference until one in progress is completed.

EXCHANGE JUMP

An exchange jump instruction is used to command the central processor to stop the program it is executing and go to a central memory location

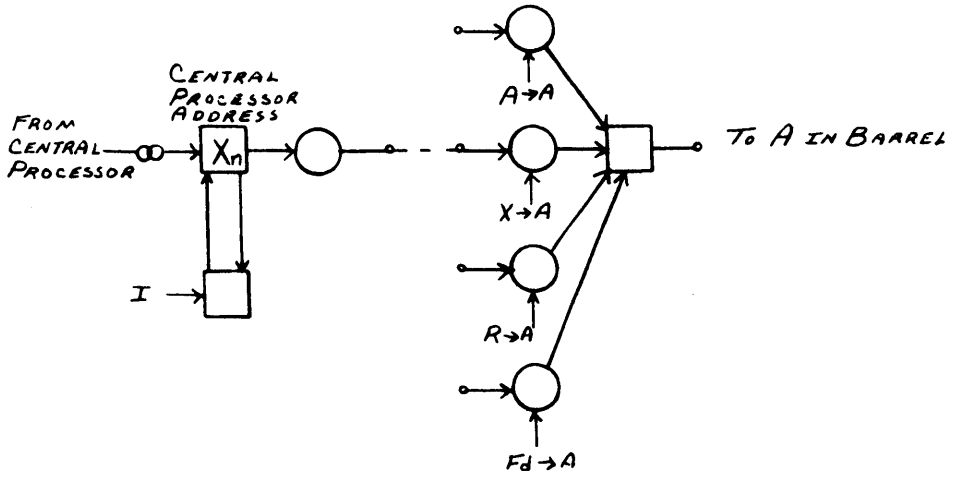


FIG. 5-18 CENTRAL PROGRAM MONITOR

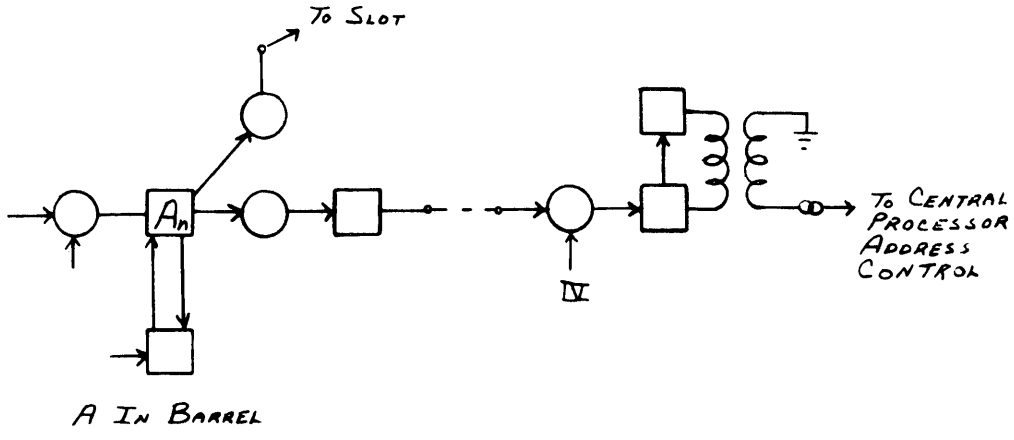


FIG 5-19 PATH OF A AS CENTRAL ADDRESS

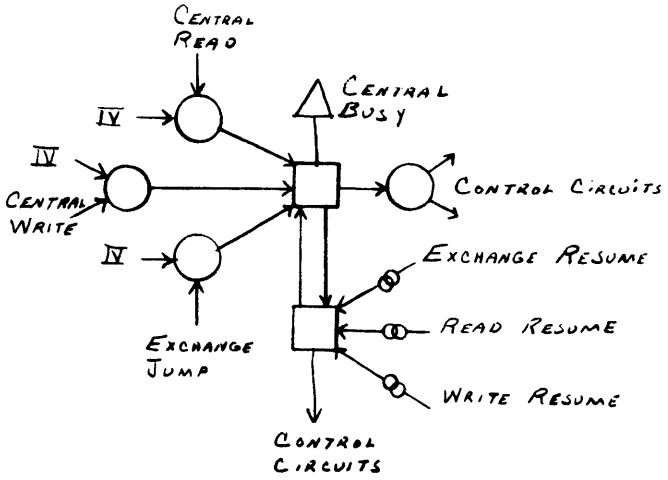


FIG. 5-20 CENTRAL BUSY FF

specified by the instruction. An exchange jump may be issued by any peripheral and control processor so long as the Central Busy FF is clear. The instruction sends an Exchange Jump signal to the central processor and sets the Central Busy FF. The Exchange Jump signal tells the central processor to recognize the 18-bit address sent from the peripheral processor and to perform an exchange jump. After the central processor has performed the exchange jump and started a new program it sends a resume signal which clears the Central Busy FF to allow another central reference. If a peripheral and control processor tries to issue an Exchange Jump instruction while the Central Busy FF is set, the processor must wait until the previous central reference is completed and the Central Busy FF is cleared.

CENTRAL READ

The Central Read instruction allows a peripheral and control processor to obtain one word (60-bits) or a block of words from Central Memory. The instruction sends a Central Read signal to central address control enabling it to use the 18 bit quantity from A as a central memory address. At the same time, the Central Busy FF is set to inhibit other references to central until the read word is received. When a 60-bit word is sent by central to the Central Read Pyramid (Fig. 5-21) it is accompanied by two control signals; an accept which clears the Central Busy FF and a signal which sets the C^5 Full FF. Each rank of the Central Read Pyramid C^1-C^5 has an associated Full/Empty FF used to control the flow of data through the pyramid. C^5 Full and C^4 Empty enables the processor doing the read instruction to send the upper 12 bits of C^5 to memory and the lower 48 bits to C^4. Subsequent steps in the Central Read instruction step the central word down through the pyramid and store the rest of the central word as 12-bit peripheral words. Each step in this storage procedure

requires that the next lower rank in the pyramid be empty before a transfer is made. No Central Read instruction may be issued until C^5 Full and Central Busy FFs are clear. However, as many as five central memory words, in different stages of disassembly, may be in the Central Read Pyramid at one time. A read instruction for which the proper full and empty conditions are not met waits until previous instructions progress further and conditions are met.

A 60 instruction reads only one central memory word and stores it as five peripheral words. A 61 instruction reads a block of words specified by (d). In either instruction the first central memory address is specified by (A). For a 60 instruction d specifies the peripheral address at which the upper 12 bits of the peripheral word are stored; the next lower 12 bits go to d+1, etc. For a 61 instruction (d) gives the number of central words to be read and m is the address for the upper 12 bits of the first central word.

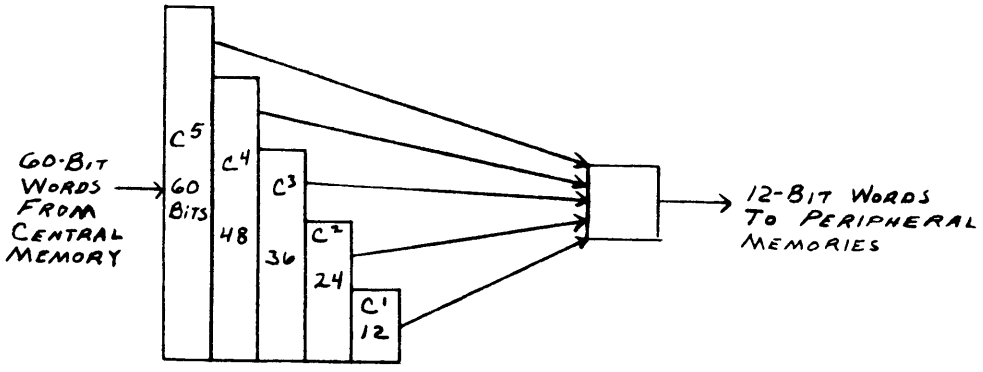


FIG. 5-21. CENTRAL READ PYRAMID

CENTRAL WRITE

Central Write instructions send one 60-bit word or a block of 60-bit words to Central Memory. Each 60-bit word sent to Central Memory is assembled in the Central Write Pyramid (Fig. 5-22) from five 12-bit peripheral words. A Central Write instruction assembles a 60-bit word and sends the word and a Central Write signal to central address control and sets the Central Busy FF. The Central Write signals enables central address control to accept the 60-bit word and store at the address specified by (A). When the word has been stored an accept signal is sent back to clear the Central Busy FF. Up to four Central Write instructions may be in progress at one time with portions of four different words in D^1-D^4. D^5 is an output network only and cannot store a word. The first 12-bit word goes to D^1 and will be the upper 12 bits of the 60-bit word. When

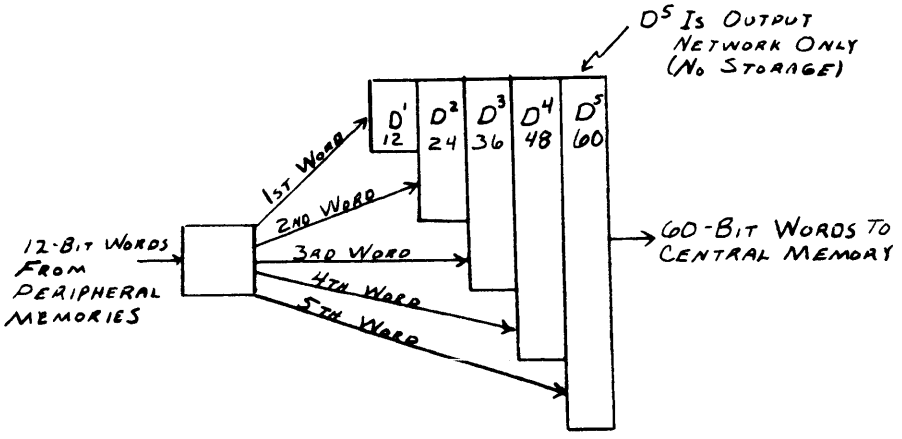


FIG. 5-22. CENTRAL WRITE PYRAMID

a second 12-bit word goes to D^2, D^1 is also sent to D^2. When the fifth word goes to D^5 the 48 bits in D^4 are also sent to D^5 and the 60-bit word is sent to central.

DEAD START

Dead Start is a system used to initially start the computer, dump the contents of the peripheral and control processor memories to a printer or other output device or sweep memory without executing instructions.

The Dead Start panel contains a 12x12 matrix of toggle switches, a Sweep-Load-Dump switch and a Dead Start switch. It also contains memory margin switches which are used for maintenance checks.

DEAD START (LOAD)

To initially load programs and data into the 6600, the Sweep-Load-Dump switch is put in the Load position. The matrix of toggle switches is set to a 12-word program (up = "1", down = "0"). When the Dead Start switch is turned on a 1 usec Dead Start pulse:

1. Assigns to each peripheral and control processor the corresponding I/O channel.
2. Sets all I/O channels to Active and Empty
3. Sets K for all processors to 712 (Input)
4. Sends a MC on all I/O channels.
5. Sets A and P for all processors to zero.

The Dead Start pulse is repeated every 4.096 ms while the Dead Start switch is on. To start the 6600 the DS switch is normally turned on momentarily, then off. Recycling of the DS pulse is controlled by the Real Time Clock; the pulse is formed by anding DS switch on with the 12 bits of Real Time Clock.

When the Dead Start synchronizer on channel 0 receives the MC sent by Dead Start, it sends a Full pulse but no data. When processor 0

receives the Full it stores the content of the channel 0 input register (All zeros) in location 0000 and sends an Empty pulse to the Dead Start synchronizer. The Dead Start synchronizer then acts like an input device and sends 12 words from the switch matrix as 12-bit words and processor 0 stores them in locations $0001\text{-}0014_8$. After the last word the Dead Start synchronizer sends a disconnect which causes processor 0 to exit from the 712 instruction. Processor 0 reads location 0000, adds one to its contents and goes to 0001 for its next instruction. It then executes the 12-word (or less) program which normally is a control program to load information and begin operation. The other processors are still set to 712 and may receive data from processor 0 via their assigned I/O channels.

SWEEP

If the DS switch is operated with the Sweep-Load-Dump switch in the Sweep position, all processors are set to a 505 instruction and P registers set to 0000. Since the 50 instruction doesn't require 5 trips around the barrel there is no logic to clear or advance K from 505. The 50X translation of K, causes all processors to sweep through their memories; reading and restoring without executing instructions. This is a maintenance routine and may be used to check the operation of memory logic.

DUMP

Dead Start with the Sweep-Load-Dump switch in the Dump position:

1. Sets all processors to 732.
2. Sends MC on all channels except channel 0.
3. Holds channel 0 Active and Empty.
4. Assigns each processor to its corresponding I/O channel.
5. Sets all A and P registers to 0.

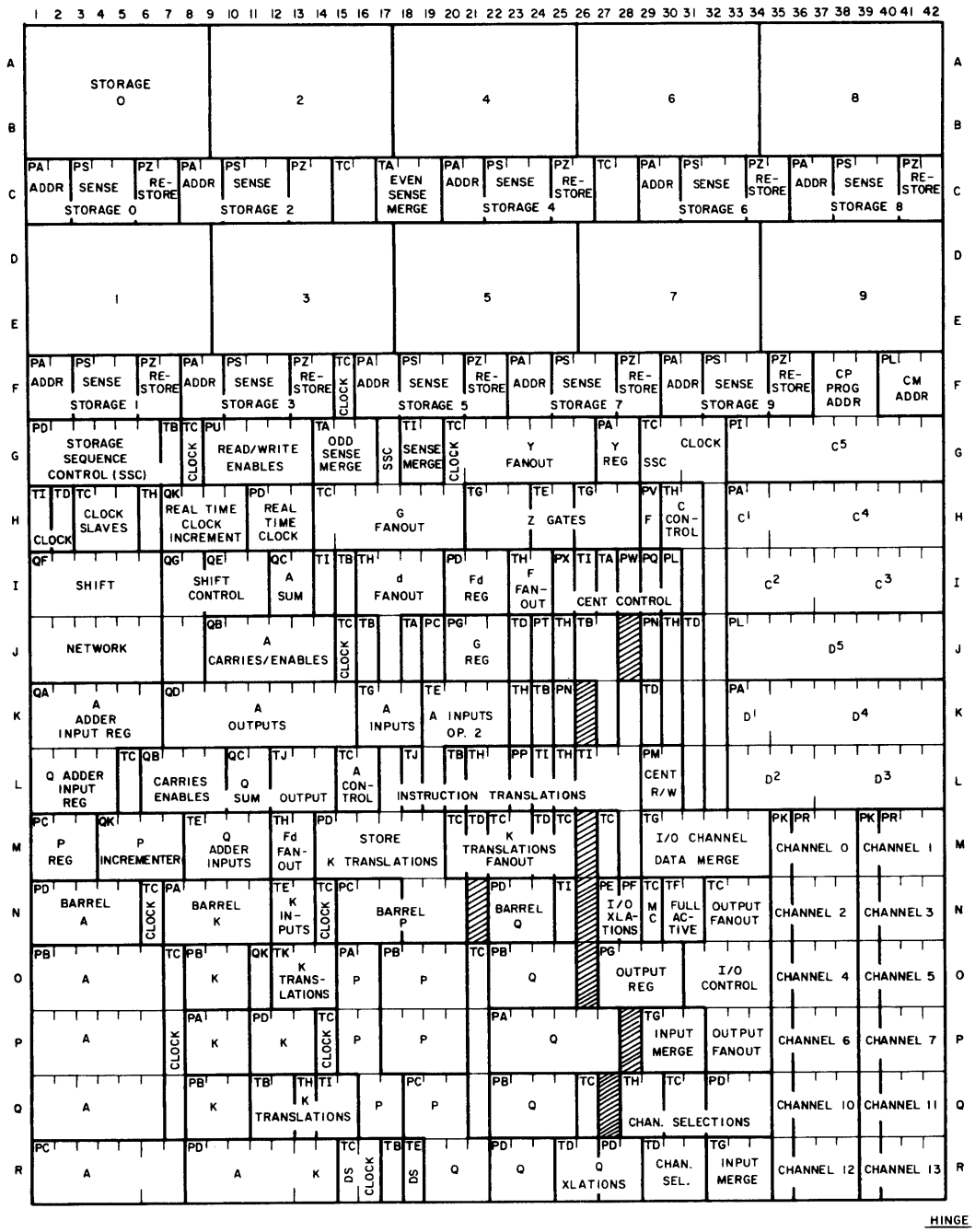
All processors sense the Empty and Active condition of their assigned channels, output the content of their address 0000, set their I/O channels

to Full, and wait for an Empty. All processors advance P by one and reduce A by one ($A = 7776_8$). Channel 0, which is assigned to processor 0, is held to Empty by the Dump switch. Processor 0 therefore cycles through the 732 instruction until A = 1 and then goes to memory location 0001 for its next instruction. Processor 0 has sent its entire memory content on channel 0 although no I/O device was selected to receive it. Processor 0 is now free to execute a dump program which must have been previously stored in memory 0 (beginning at location 0001).

APPENDIX A

Card Placement

CHASSIS 1


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42
STORAGE 0
2
4
6
8
1
3
5
7
9
PA ADDR
PS SENSE
PZ RE-STORE
STORAGE 0
STORAGE 2
STORAGE 4
STORAGE 6
STORAGE 8
STORAGE 1
STORAGE 3
STORAGE 5
STORAGE 7
STORAGE 9
TC CLOCK
TA EVEN SENSE MERGE
CP PROG ADDR
PL CM ADDR
PD STORAGE SEQUENCE CONTROL (SSC)
TB
PU READ/WRITE ENABLES
TI SENSE MERGE
SSC
Y FANOUT
PA Y REG
PI C5
TI CLOCK
TD CLOCK SLAVES
TH
QK REAL TIME CLOCK INCREMENT
PD REAL TIME CLOCK
G FANOUT
TG Z GATES
TE
PV F
TH C CONTROL
C1 C4
QF SHIFT
QG QE SHIFT CONTROL
QC A SUM
d FANOUT
Fd REG
F FAN-OUT
PX TI TA PW PQ PL CENT CONTROL
C2 C3
NETWORK
QB A CARRIES/ENABLES
PC PG G REG
TD PT TH TB PN TH TD
D5
QA A ADDER INPUT REG
QD A OUTPUTS
A INPUTS
A INPUTS OP. 2
D1 D4
Q ADDER INPUT REG
CARRIES ENABLES
Q SUM
OUTPUT
A CONTROL
INSTRUCTION TRANSLATIONS
PP TI TH TI
PM CENT R/W
D2 D3
PC P REG
QK P INCREMENTER
Q ADDER INPUTS
Fd FAN-OUT
STORE K TRANSLATIONS
K TRANSLATIONS FANOUT
I/O CHANNEL DATA MERGE
PK PR CHANNEL 0
CHANNEL 1
BARREL A
BARREL K
K IN-PUTS
BARREL P
BARREL Q
PE PF I/O XLA-TIONS
MC
FULL AC-TIVE
OUTPUT FANOUT
CHANNEL 2
CHANNEL 3
A
K
K TRANS-LATIONS
P
Q
OUTPUT REG
I/O CONTROL
CHANNEL 4
CHANNEL 5
INPUT MERGE
CHANNEL 6
CHANNEL 7
K TRANSLATIONS
CHAN. SELECTIONS
CHANNEL 10
CHANNEL 11
DS
Q XLATIONS
CHAN. SEL.
CHANNEL 12
CHANNEL 13
HINGE

APPENDIX B

Circuits

CIRCUITS

The majority of the circuits in the 6600 use the inversion property of transistors to obtain logical OR and AND configurations. These in turn are connected into flip-flops (FF) which store data temporarily. Special circuits drive twisted pair or coaxial cable transmission lines, control storage or read out of data in the magnetic core memory, amplify and detect magnetic tape or disk file head signals, and control other functions.

PACKAGING

Circuits are packaged in modules (fig. B-1) which use printed circuit wiring. Circuit components are mounted on and between two printed circuit boards in a method sometimes referred to as "cordwood packaging". A 30-pin connector fastened at one end of the module allows electrical access to the circuits. The module connector mates with a chassis connector whose pins are wired to other similar circuits. A plate with captive screws is fastened to the module at the end opposite the connector. This plate connects the module to the chassis mechanically and also conducts component heat to the chassis cooling system. The plate carries an identifying number for the module and six test point terminals. The test points are numbered 1 through 6 from top to bottom and are connected to various parts of a circuit to allow circuit operation to be viewed on an oscilloscope.

COOLING

Modules are mounted vertically in the chassis in separate compartments. Compartment side walls, the connector (back), and module plate (front) have a black finish. Rows of modules are separated and supported by metal plates (cold bars) which form the top and bottom of the compartments. A copper tube is imbedded in each bar and is connected to a freon refrigeration (cooling) system. Component heat from each module is carried to the cold bars and cooling system by convection and by black body conduction.

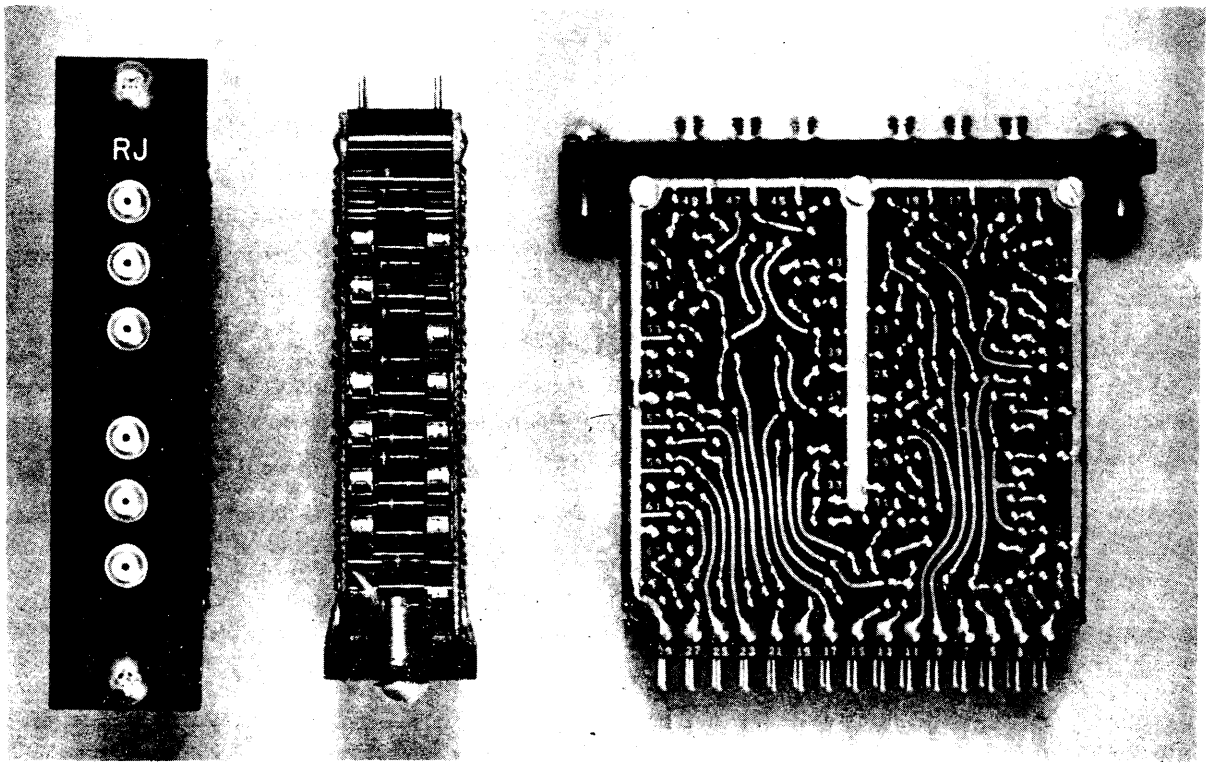


Fig. B-1. 6600 Circuit Module

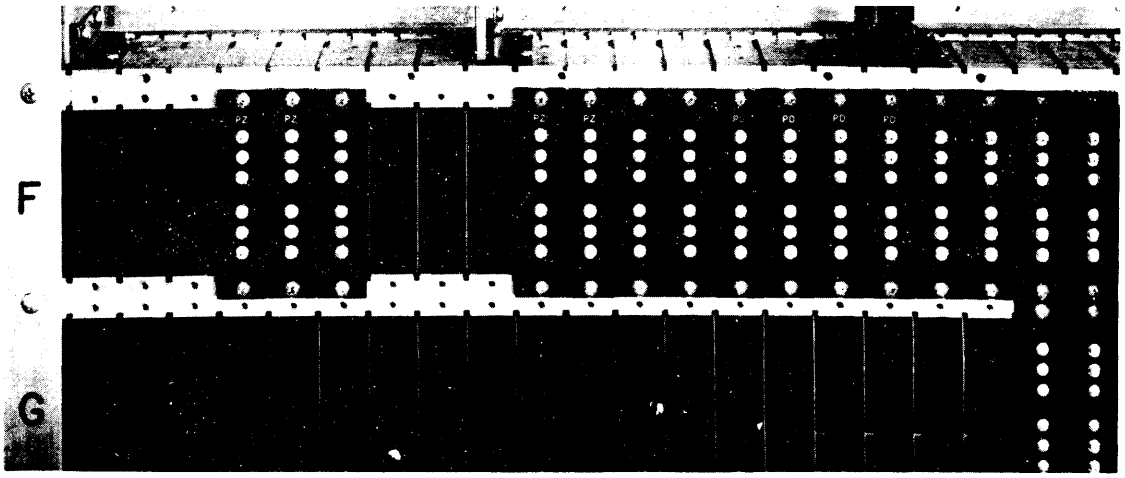


Fig. B-2. Circuit Module Chassis Compartments

BASIC CIRCUIT ANALYSIS

The basic circuit is a single NPN silicon transistor connected in a grounded emitter amplifier circuit (fig. B-3) The transistor operates from a +6 vdc supply. The inherent 180° phase shift of the transistor provides the basic inversion (or NOT) function. The two signal levels and their logical significance are as follows:

+0.2v	"1"
+1.2v	"0"

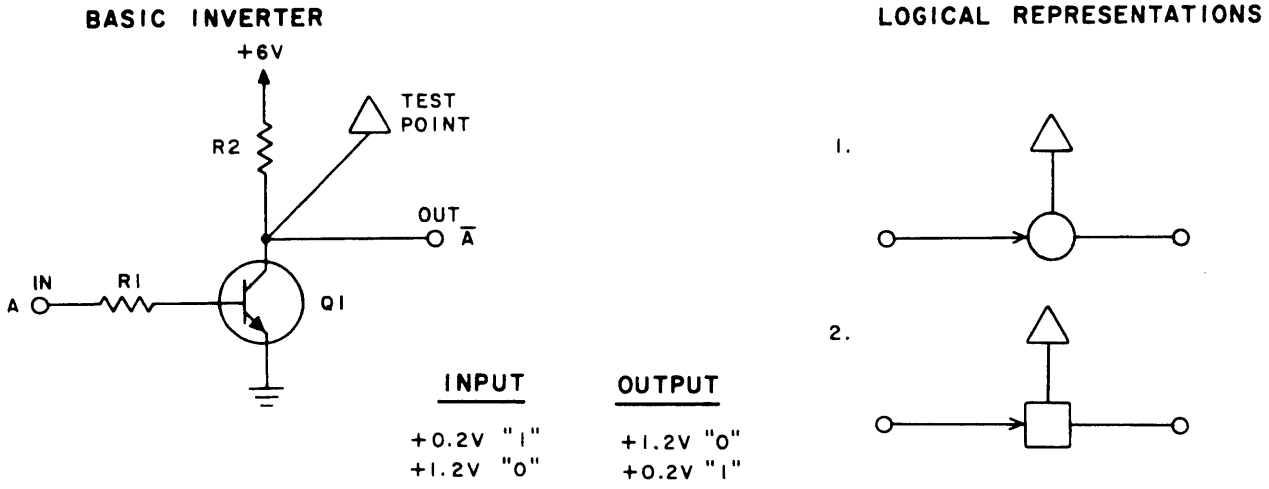


Fig. B-3. Basic Inverter

A +1.2v input turns on the transistor and drives it to saturation. The output signal is then the drop across the transistor, +0.2v. When a +0.2v input cuts off the transistor the output rises toward +6v but is held to +1.2v by the voltage-dividing effect of the collector resistor and the input resistance of the driven load.

The threshold voltage on the base of the transistor is approximately +0.8v (virtually no conduction below +0.7v) which permits the circuit to reject noise signals up to about +0.5v. Circuit parameters for cutoff and saturation are listed below.

	V_B	V_C	I_B (min)	I_C	Average Switching Time
Cutoff	0.2	1.2	0	0	5 nsec
Saturation	0.8	0.2	1 ma	10 ma	

AND & OR COMBINATIONS

To combine signals from two circuits, the collectors are connected to a common load resistor, as in fig. B-4. This provides $\overline{A} + \overline{B} = (\overline{AB})$ as the output signal. This method of combining signals is sometimes called NOR logic. (The name derives from NOT OR since the circuit consists of an OR with phase inversion or negation of the inputs.) Up to six transistors may be connected to a common collector resistor in this way.

The circuit to combine two signals A and B to provide AB as the output appears in figure 5. When either A or B is +1.2v, Q1 or Q2 is biased to saturation and the outer at point C is +0.2v. This in turn cuts Q3 off and the output is +1.2v. But when both A and B are +0.2v, both Q1 and Q2 are cut off, the the potential at point C begins to rise toward +6v. When it reaches +0.8v, Q3 begins to conduct and soon goes to saturation. Base, or leakage, current from Q3 flows through R4 and R3 which form a voltage divider. The base-to-emitter drop of Q3 plus the drop across R4 establish the level at point C at +1.2v. With Q3 conducting the output is +0.2v. Hence the output is +0.2v only when both inputs, A and B, are +0.2v.

The circuit of figure 6 shows the OR combination of two signals A and B, in which a +0.2v input at either A or B results in a +0.2v output.

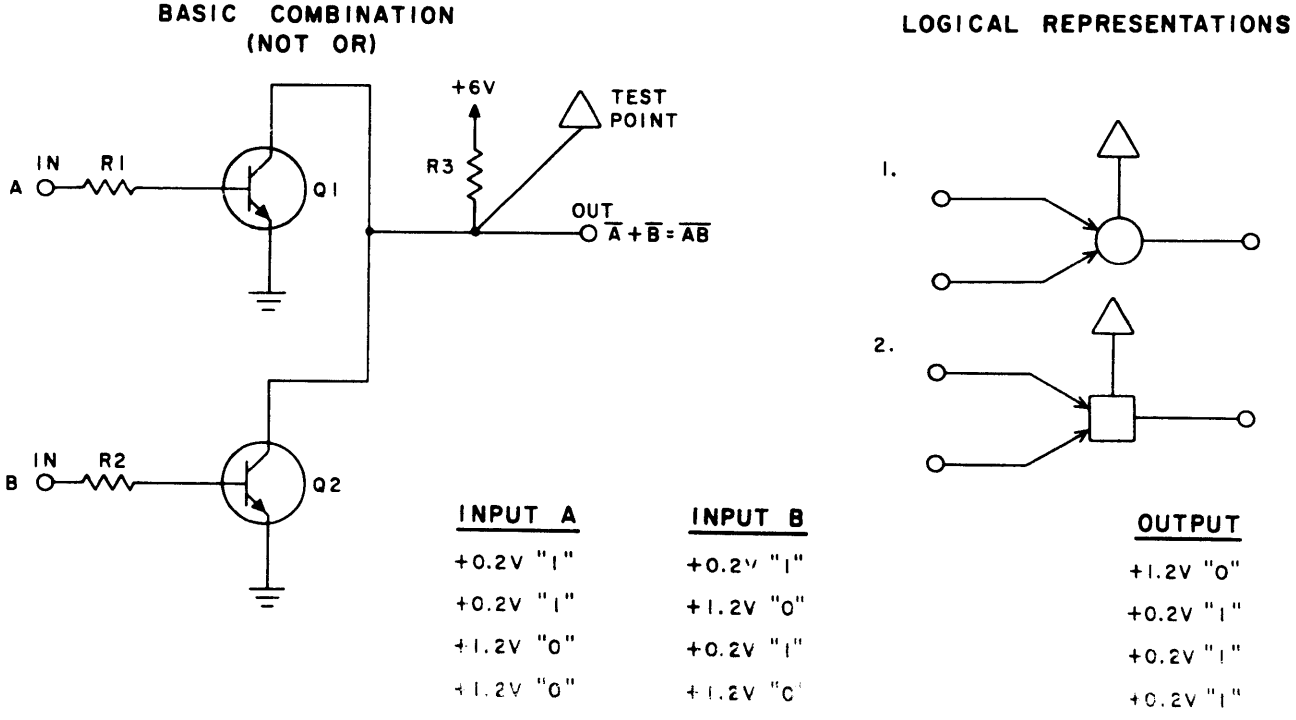


Fig. B-4. Combining Signals

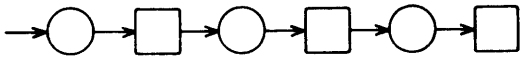
LOGIC DIAGRAM SYMBOLS

The symbols used to represent circuits on logic diagrams are as follows:

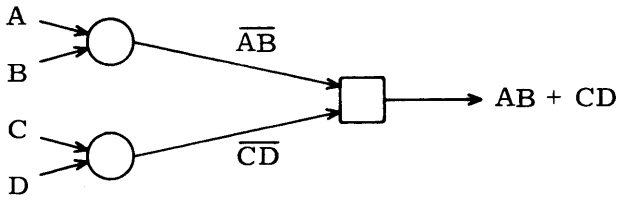
Symbol	Logical Function	Literal Significance
→	inversion	transistor (including its base resistor)
□	OR combination (one or more "0's")	collector load resistor
○	AND combination (all "1's")	collector load resistor

The use of two symbols, circle and square, to represent the combining of signals is a convenient way to indicate alternating logical levels. This alternation results from the inversion accompanying each successive combination. Logic is performed by strings or successions of steps: invert, combine, invert, combine, invert, combine, etc. Thus every other combining step is the reverse of its predecessor.

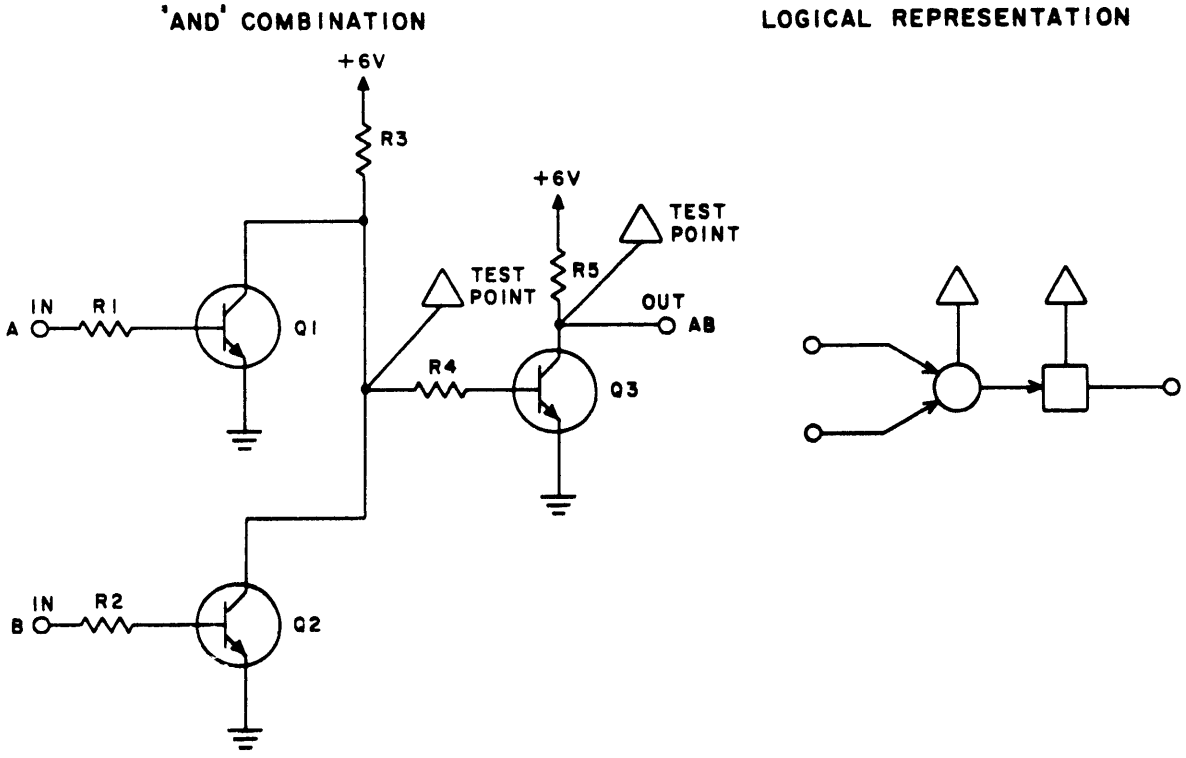
A diagram of such a logical string of steps appears as



Where possible, diagrams have circles and squares appearing in alternation and in such a way as to indicate whether the effect of a combination is an AND or an OR. Consider the logical circuit shown below:

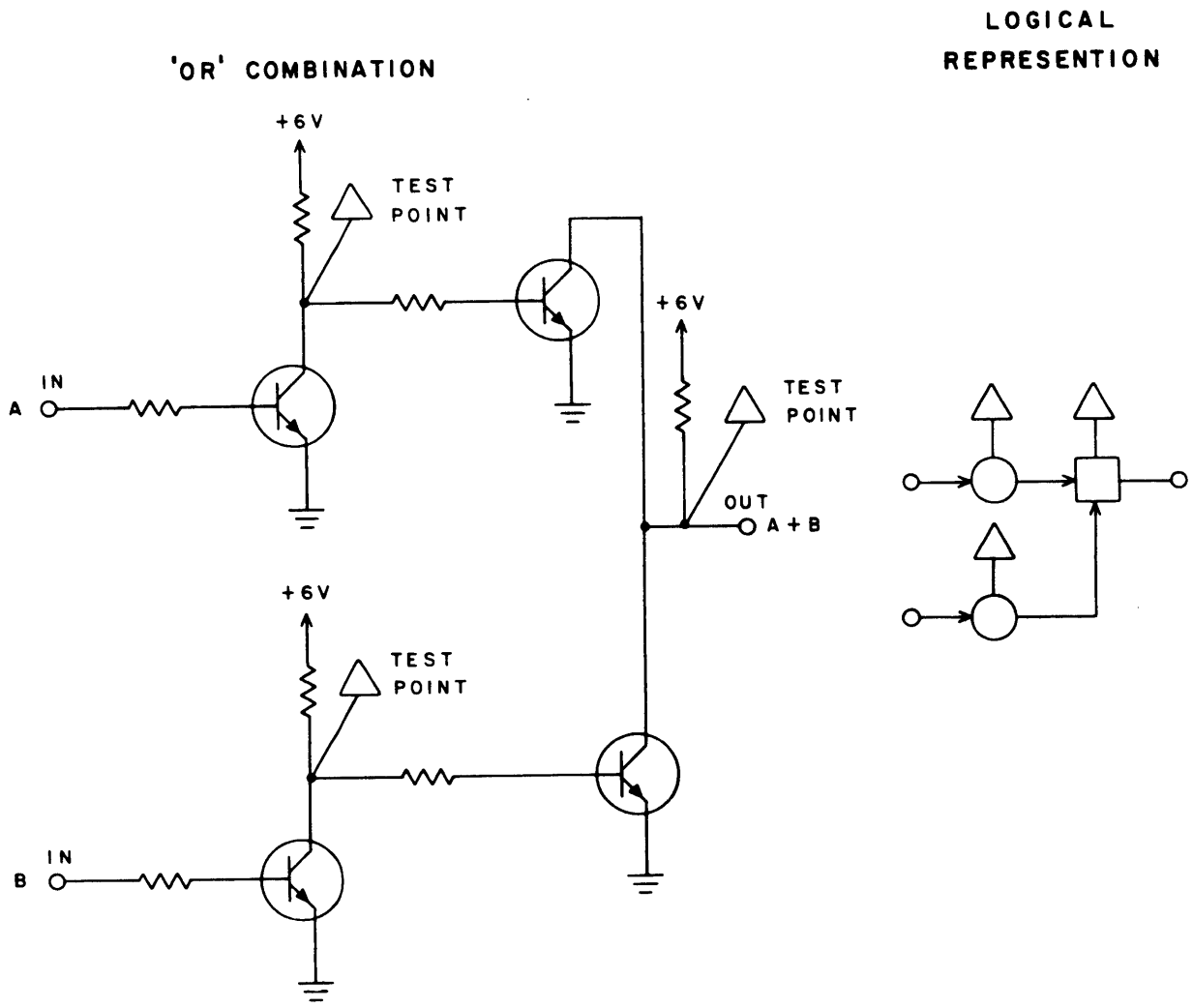


Its output is $\overline{\overline{A} + \overline{B} + \overline{C} + \overline{D}}$ which simplifiers to AB + CD. The square may be thought of as representing an CR.



INPUT A	INPUT B	OUTPUT
+0.2V "1"	+0.2V "1"	+0.2V "1"
+0.2V "1"	+1.2V "0"	+1.2V "0"
+1.2V "0"	+0.2V "1"	+1.2V "0"
+1.2V "0"	+1.2V "0"	+1.2V "0"

Fig. B-3. AND Combination



INPUT A	INPUT B	OUTPUT
+0.2 V "1"	+0.2 V "1"	+0.2 V "1"
+0.2 V "1"	+1.2 V "0"	+0.2 V "1"
+1.2 V "0"	+0.2 V "1"	+0.2 V "1"
+1.2 V "0"	+1.2 V "0"	+1.2 V "0"

Fig. B-6. OR Combination

FLIP-FLOP WITH CLEAR/SET INPUT

The circuit of a flip-flop and its clear/set input is shown in fig. B-7. The basic flip-flop may have up to 5 set inputs and 5 clear inputs; in this example only one of each are shown. The set input is fed by a 3-way AND gate, of which one input comes from the clear/set network.

The clear/set network enables the flip-flop to be cleared and reset by the same gating pulse. In many cases, this will be a 25 nanosecond clock pulse. Figure 8 shows the logical representation of this circuit and a timing chart for the clear/set operation.

A +0.2v "1" clear/set input to A results in +0.2v "1" outputs from B and E. This simultaneously clears the flip-flop while enabling AND gate C to reset it, if "1" inputs are present. Assuming "1" inputs, the +1.2v "0" output of C drives the output of D to a +0.2v "1". In this state, D and E both are producing "1" outputs and the flip-flop is an indeterminate condition. However, when the clear/set pulse drops, the "1" output from A allows E to switch to "0", providing feedback to D which holds the flip-flop in the set condition. Meanwhile, the "0" output of B has forced C to remove the set input from D, but the delay time through B and C will allow the set input to stay up long enough after the clear input drops so that the flip-flop assumes the set state.

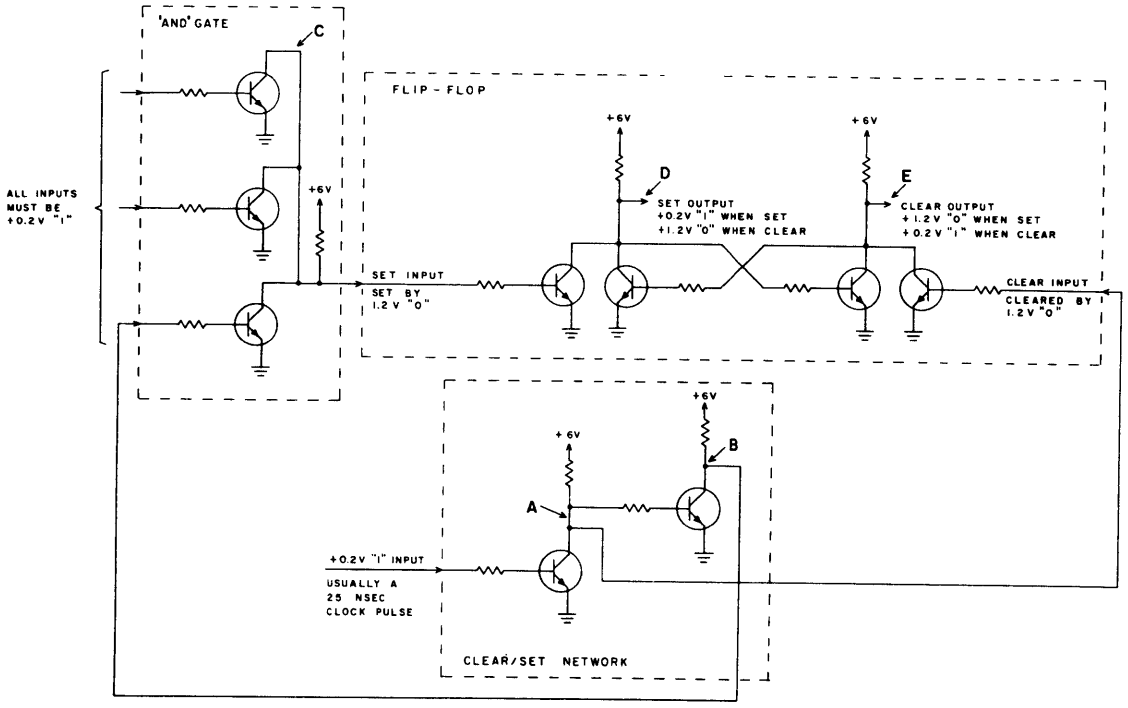


Fig. B-7. Basic Flip-Flop with Clear/Set Input

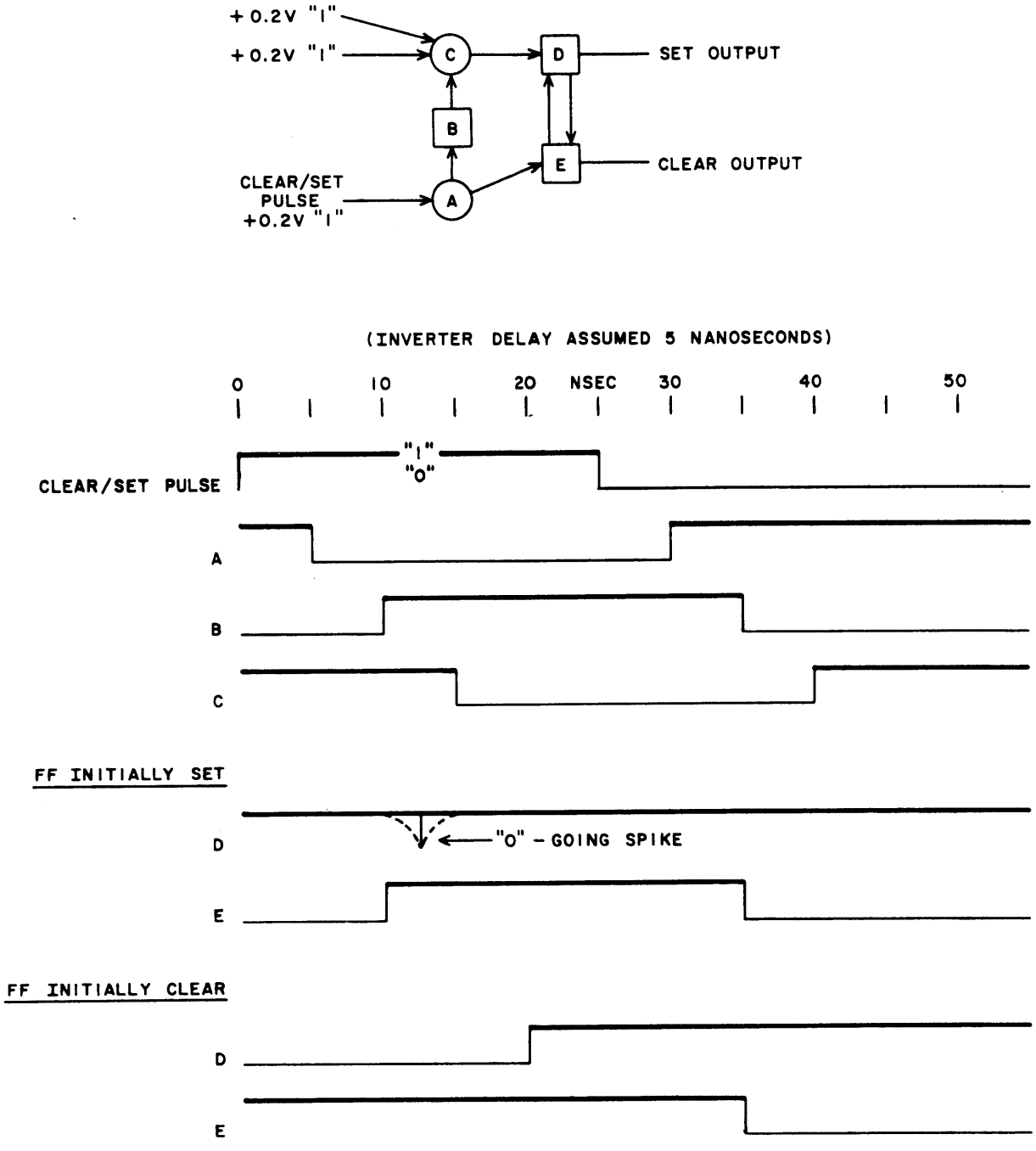


Fig. B-8. Flip-Flop Logical Representation and Timing Chart

CIRCUIT LOADING

One circuit, which may be a common collector connection of several transistors, can drive several other transistors which may be on the same or on a separate module, or a combination of the two. Figure 9 shows examples of loading.

A circuit may drive up to five transistors when all transistors (including driver) are on the same module. This loading limitation is due to variations in base threshold voltage and corresponding base current demands. For each load configuration, the driver collector resistor and base resistors of the driven transistors are adjusted to balance and limit the base currents. This loading arrangement allows ample slave networks to provide multiple outputs.

The following table lists component variations for circuit loading within a module.

	Driver Collector Resistor	Base Resistor of Driven Transistors
1 Load	680	47
2 Loads	680	100
3 Loads	680	150
4 Loads	560	150
5 Loads	470	150

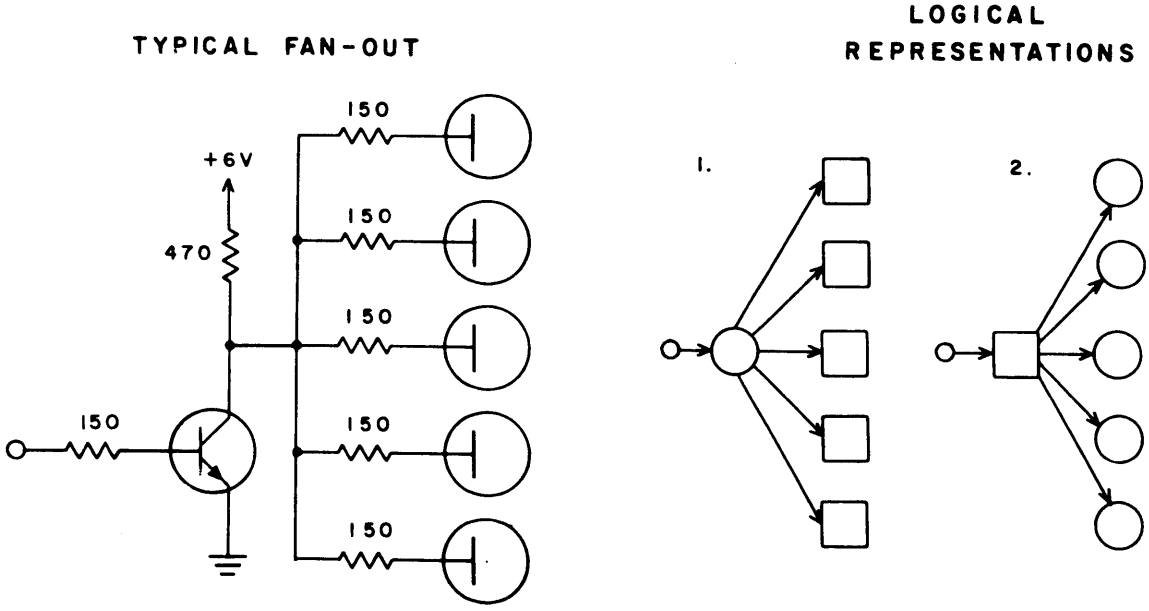


Fig. B-9. Base Resistor Loading

Other load limits apply when the circuit load is on a separate module. A circuit can simultaneously drive up to two transistors on the same module and up to two transistors on a separate module. Loading in this case accounts for the loss on the transmission line connecting the two modules as well as variations in base threshold voltage noted earlier. The significance of this loading scheme is presented in the discussion on line drivers.

The following table lists component variations for a transistor driving loads within the same module, in addition to the twisted-pair line.

Number of Local Loads	Driver Collector Resistor	Output Line Resistor	Base Resistor of Driven Transistors	Output Shunt to Ground
None	470	56	---	120 plus diode
1	470	56	100	---
2	470	56	220	---

LINE DRIVERS

Modules which are located on the same chassis are connected with twisted pair lines; modules located on separate chassis are connected by coaxial cable. Line driver circuits are provided for each type of transmission line.

TWISTED-PAIR DRIVER

The twisted pair driver is shown in fig. B-10a. Initially, consider Q1 conducting so that its collector is at +0.2v. With Q2 off, the transmission line is terminated in an open circuit.

A +0.2v "1" input causes the collector of Q1 to rise to +1.2v. This voltage is greater than the +0.8v diode threshold, and 3.3 ma of the resulting 10 ma current drawn through R2 is diverted through R3 and CR1 to ground. The remaining 6.7 ma passes R4 and is placed on the line with a voltage of +0.9v (1.2v - 6.7 ma x 56).

A finite time later this wavefront reaches the end of the line and appears at the input to R5, the base resistor for Q2. Since, at this instant, the voltage at the base of Q2 is +0.2v, this voltage adds to the incoming wavefront to result in a voltage of +1.1v at the input to R5.

At the next instant, the Q2 threshold is exceeded and base current flows. The base does not draw all of the current which is available in the line since it does not conduct appreciably below the +0.8v threshold. Thus, the source voltage for the base is +0.3v (+1.1v -0.8v) and resistor R5 limits base current to 2 ma.

Transmission line theory shows that power is reflected back to the source when the line is terminated in a load which does not absorb all of the transmitted power. Transmission line formulas for terminal voltage (V_T) and current (I_T) in this case show that the reflected wave has a current component of 4.7 ma and a voltage component of +0.3v. The terminal point is at the output of R4, the input to the line.

$V_T = V_F - V_R$	where:	T = terminal
and		F = forward component
$I_T = I_F - I_R$		R = reflected component
$V_R = +1.1 - (+0.8)$		= +0.3v
$I_R = 6.7 \text{ ma} - 2 \text{ ma}$		= 4.7 ma

The reflected current is shunted to ground by diode CR1. In the absence of CR1, the line voltage would rise to the +6v source.

The low base current, resulting from the line mismatch, retards the build-up of ground noise which occurs as a result of large ground currents.

The shunt diode and series resistor R3 are replaced by a transistor and its base resistor when it is necessary to drive a circuit on the same chassis and a circuit on another chassis. The transistor and diode thresholds are nearly alike so that circuit operation follows the same principle.

The time required by the wavefront to travel down the line is a function of the type of line and its length. The twisted pair line provides a delay of 1.3 nsec per foot. The clock system, for example, uses this feature to establish the 25 nsec duration of clock pulses.

The twisted pair driver is represented on the logic diagrams by the standard square or circle. However, the output of the square or circle is connected to a pin of the module in question and wired from there to a pin on another module. The ground wire of the pair is wired to the connector ground bus of each module.

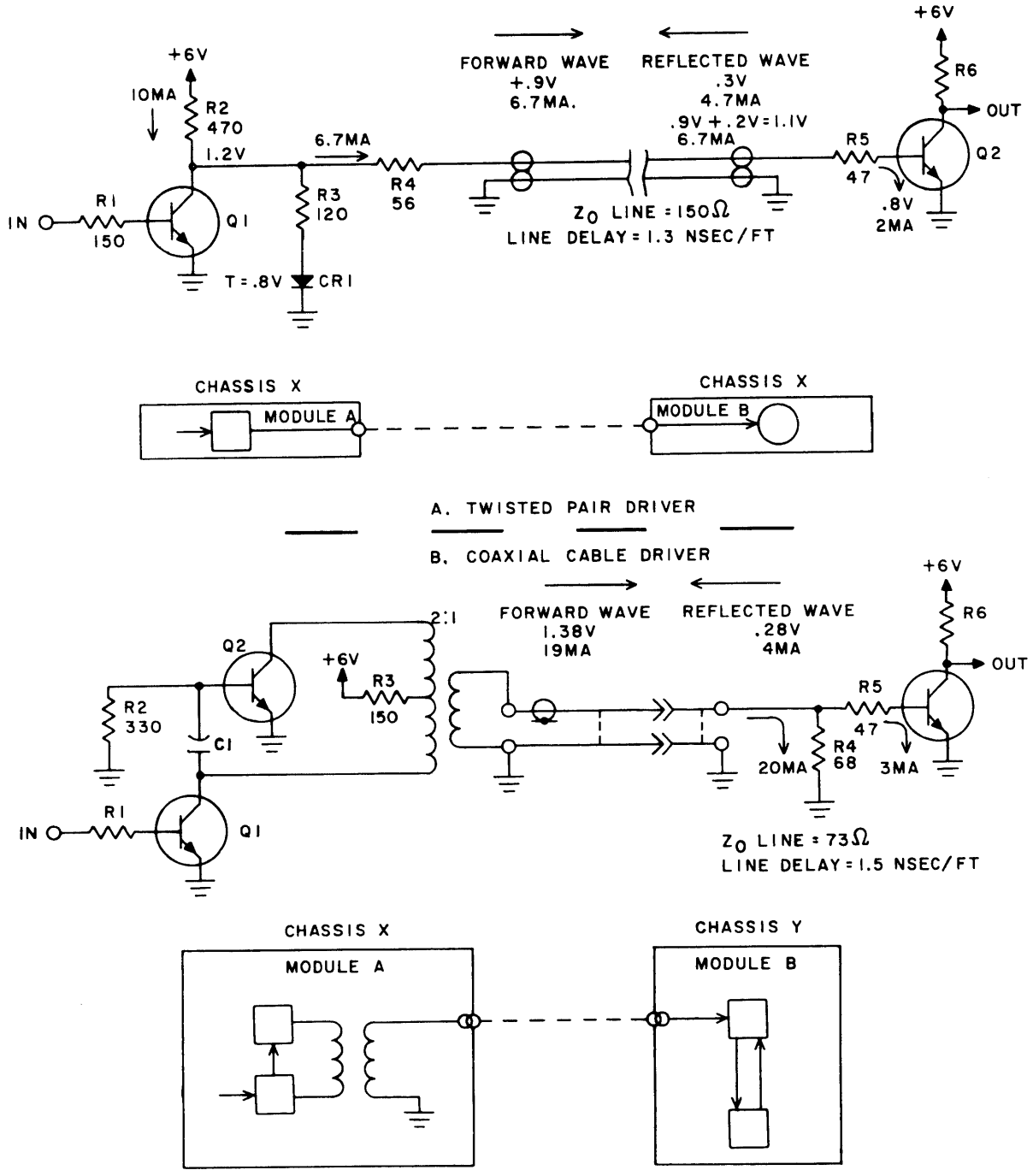


Fig. B-10. Line Driver Circuits

COAXIAL CABLE DRIVER

The coaxial cable driver is shown in fig. B-10b. All transmissions between chassis are 25 nsec pulses on coaxial cables.

Initially, Q1 and Q2 are off so that their collectors are at the source voltage, +6v. No current flows in either primary winding. When Q1 is turned on, a 10 ma current flows in the transformer primary winding to which its collector is connected. The voltage drop across R3 and step-down action of the transformer (and transformer loss) produce a voltage pulse on the secondary winding of approximately +1.38v. The pulse duration on the line follows the 25 nsec pulse on the base of Q1. This pulse travels down the coaxial cable and turns transistor Q3 on. Resistor R4 terminates the line in slightly lower than its characteristic impedance of 75 ohms. Approximately 20 ma of secondary current is shunted to ground through R4; Q3 draws approximately 3 ma base current.

Capacitor C1 couples the change in Q1 collector voltage to the base of Q2. Capacitor C1, R3, R2, and the base/emitter diode of Q1 will cause Q2 to conduct about 25 nsec later than Q1. When Q2 conducts, current flows in the primary winding to which the Q2 collector is connected. The resulting signal in the transformer secondary is equal to but of opposite polarity from the previous signal and lags by 25 nsecs. The net effect of this is to provide an equal and opposite magnetizing force in the transformer for each pulse. This prevents the transformer from building up a bias on the line which could occur under the high switching rates.

The coaxial cable driver circuit is represented on logic diagrams as shown in fig. B-11b. The time delay on the 75 ohm coaxial cables is 1.5 nsec per foot. All coaxial cable lengths are standardized at 10 feet to allow synchronous timing signals on all chassis. Coaxial cables pass directly through a chassis feed-through connector (without soldering) and connect directly to the transmitting or terminating circuit. However, a coaxial connector is provided at the midpoint of each 10-foot cable to give hookup flexibility.

APPENDIX C

Peripheral Command Timing

Mnemonic & Octal Code	Name
PSN 00	Pass
LJM 01	Long jump to m + (d)
RJM 02	Return jump to m + (d)
UJN 03	Unconditional jump d
ZJN 04	Zero jump d
NJN 05	Nonzero jump d
PJN 06	Plus jump d
MJN 07	Minus jump d
SHN 10	Shift d
LMN 11	Logical difference d
LPN 12	Logical product d
SCN 13	Selective clear d
LDN 14	Load d
LCN 15	Load complement d
ADN 16	Add d
SBN 17	Subtract d
LDC 20	Load dm
ADC 21	Add dm
LPC 22	Logical product dm
LMC 23	Logical difference dm
PSN 24	Pass
PSN 25	Pass
EXN 26	Exchange jump
RPN 27	Read program address
LDD 30	Load (d)
ADD 31	Add (d)
SBD 32	Subtract (d)
LMD 33	Logical difference (d)
STD 34	Store (d)
RAD 35	Replace add (d)
AOD 36	Replace add one (d)
SOD 37	Replace subtract one (d)
LDI 40	Load ((d))
ADI 41	Add ((d))
SBI 42	Subtract ((d))

Mnemonic & Octal Code	Name
LMI 43	Logical difference ((d))
STI 44	Store ((d))
RAI 45	Replace add ((d))
AOI 46	Replace add one ((d))
SOI 47	Replace subtract one ((d))
LDM 50	Load (m + (d))
ADM 51	Add (m + (d))
SBM 52	Subtract (m + (d))
LMM 53	Logical Difference (m + (d))
STM 54	Store (m + (d))
RAM 55	Replace add (m + (d))
AOM 56	Replace add one (m + (d))
SOM 57	Replace subtract one (m + (d)
CRD 60	Central read from (A) to d
CRM 61	Central read (d) words from (A) to m
CWD 62	Central write to (A) from d
CWM 63	Central write (d) words to (A) from m
AJM 64	Jump to m if channel d active
IJM 65	Jump to m if channel d inactive
FJM 66	Jump to m if channel d full
EJM 67	Jump to m if channel d empty
IAN 70	Input to A from channel d
IAM 71	Input (A) words to m from channel d
OAN 72	Output from A on channel d
OAM 73	Output (A) words from m on channel d
ACN 74	Activate channel d
DCN 75	Disconnect channel d
FAN 76	Function (A) on channel d
FNC 77	Function m on channel d

PERIPHERAL COMMAND TIMING

(00) read P. P + 1 to P.

(01) read P. P + 1 to P. d to Q. F to K. Adv K if d = 0.

010 read Q. Fd to Q. Adv K.

011 read P. Q + Fd to P. Clear K.

(02) read P. P + 1 to P. d to Q. F to K. Adv K if d = 0.

020 read Q. Fd to Q. Adv K.

021 read P. Q + Fd to Q. P + 1 to P. Adv K.

022 read Q. P to Z. Q + 1 to P. Clear K.

(03) read P. P + d to P.

(04) read P. P + d to P if A = 0. P + 1 to P if A ≠ 0.

(05) read P. P + d to P if A ≠ 0. P + 1 to P if A = 0.

(06) read P. P + d to P if A is +. P + 1 to P if A is -.

(07) read P. P + d to P if A is -. P + 1 to P if A is +.

(10) read P. P + 1 to P. Shift A by d.

(11) read P. A ld d to A. P + 1 to P.

(12) read P. A lp d to A. P + 1 to P

(13) read P. A sc d to A. P + 1 to P.

(14) read P. d to A. P + 1 to P.

(15) read P. -d to A. P + 1 to P.

(16) read P. A + d to A. P + 1 to P.

(17) read P. A - d to A. P + 1 to P.

(20) read P. d to Q. P + 1 to P. F to K.

200 read P. QFd to A. P + 1 to P. Clear K.

(21) read P. d to Q. P + 1 to P. F to K.

210 read P. A + QFd to A. P + 1 to P. Clear K.

(22) read P. d to Q. P + 1 to P. F to K.

220 read P. A lp QFd to A. P + 1 to P. Clear K.

(23) read P. d to Q. P + 1 to P. F to K.

230 read P. A ld QFd to A. P + 1 to P. Clear K.

(24) read P. P + 1 to P.

(25) read P. P + 1 to P.

(26) read P. A to L. If cent. empty (P + 1 to P.)

(27) read P. X to A. P + 1 to P.

(30) read P. d to Q. P + 1 to P. F to K.

300 read Q. Fd to A. Clear K.

(31) read P. d to Q. P + 1 to P. F to K.

310 read Q. A + Fd to A. Clear K.

(32) read P. d to Q. P + 1 to P. F to K.

320 read Q. A - Fd to A. Clear K.

(33) read P. d to Q. P + 1 to P. F to K.

330 read Q. A Ld Fd to A. Clear K.

(34) read P. d to Q. P + 1 to P. F to K.

340 read Q. A to Z. Clear K.

(35) read P. d to Q. P + 1 to P. F to K.

350 read Q. A + Fd to A. Set K = 340.

(36) read P. d to Q. P + 1 to P. F to K.

360 read Q. Fd + 1 to A. Set K = 340.

(37) read P. d to Q. P + 1 to P. F to K.

370 read Q. Fd - 1 to A. Set K = 340.

(40) read P. d to Q. P + 1 to P. F to K.

```
400  read Q.  Fd to Q.  Adv K.
401  read Q.  Fd to A.  Clear K.

(41) read P.  d to Q.  P + 1 to P.  F to K.
410  read Q.  Fd to Q.  Adv K.
411  read Q.  A + Fd to A.  Clear K.

(42) read P.  d to Q.  P + 1 to P.  F to K.
420  read Q.  Fd to Q.  Adv K.
421  read Q.  A - Fd to A.  Clear K.

(43) read P.  d to Q.  P + 1 to P.  F to K.
430  read Q.  Fd to Q.  Adv K.
431  read Q.  A Ld Fd to A.  Clear K.

(44) read P.  d to Q.  P + 1 to P.  F to K.
440  read Q.  Fd to Q.  Adv K.
441  read Q.  A to Z.  Clear K.

(45) read P.  d to Q.  P + 1 to P.  F to K.
450  read Q.  Fd to Q.  Adv K.
451  read Q.  A + Fd to A.  Set K = 340.

(46) read P.  d to Q.  P + 1 to P.  F to K.
460  read Q.  Fd to Q.  Adv K.
461  read Q.  Fd + 1 to A.  Set K = 340.

(47) read P.  d to Q.  P + 1 to P.  F to K.
```

```
470  read Q.  Fd to Q.  Adv K.
471  read Q.  Fd - 1 to A.  Set K = 340.

(50) read P.  d to Q.  P + 1 to P.  F to K.  Adv K if d = 0.
500  read Q.  Fd to Q.  Adv K.
501  read P.  Q + Fd to Q.  P + 1 to P.  Adv K.
502  read Q.  Fd to A.  Clear K.

(51) read P.  d to Q.  P + 1 to P.  F to K.  Adv K if d = 0.
510  read Q.  Fd to Q.  Adv K.
511  read P.  Q + Fd to Q.  P + 1 to P.  Adv K.
512  read Q.  A + Fd to A.  Clear K.

(52) read P.  d to Q.  P + 1 to P.  F to K.  Adv K if d = 0.
520  read Q.  Fd to Q.  Adv K.
521  read P.  Q + Fd to Q.  P + 1 to P.  Adv K.
522  read Q.  A - Fd to A.  Clear K.

(53) read P.  d to Q.  P + 1 to P.  F to K.  Adv K if d = 0.
530  read Q.  Fd to Q.  Adv K.
531  read P.  Q + Fd to Q.  P + 1 to P.  Adv K.
532  read Q.  A Ld Fd to A.  Clear K.

(54) read P.  d to Q.  P + 1 to P.  F to K.  Adv K if d = 0.
540  read Q.  Fd to Q.  Adv K.
541  read P.  Q + Fd to Q.  P + 1 to P.  Adv K.
542  read Q.  A to Z.  Clear K.
```

(55) read P. d to Q. P + 1 to P. F to K. Adv K if d = 0.

550 read Q. Fd to Q. Adv K.

551 read P. Q + Fd to Q. P + 1 to P. Adv K.

552 read Q. A + Fd to A. Set K = 340.

(56) read P. d to Q. P + 1 to P. F to K. Adv K if d = 0.

560 read Q. Fd to Q. Adv K.

561 read P. Q + Fd to Q. P + 1 to P. Adv K.

562 read Q. Fd + 1 to A. Set K = 340.

(57) read P. d to Q. P + 1 to P. F to K. Adv K if d = 0.

570 read Q. Fd to Q. Adv K.

571 read P. Q + Fd to Q. P + 1 to P. Adv K.

572 read Q. Fd = 1 to A. Set K = 340.

(60) read P. A to L. d to Q.

If C^1 + C^2 + C^3 + C^4 Empty and C^5 Empty and Cent. $\overline{\text{Busy}}$ (P + 1 to P. F to K.)

600 read Q. If C^5 full and C^4 Empty (C^5 to Z. C^5 to C^4. Q + 1 to Q. Adv K.)

601 read Q. If C^3 Empty (C^4 to Z. C^4 to C^3. Q + 1 to Q. Adv K.)

602 read Q. If C^2 Empty (C^3 to Z. C^3 to C^2. Q + 1 to Q. Adv K.)

603 read Q. If C^1 Empty (C^2 to Z. C^2 to C^1. Q + 1 to Q. Adv K.)

604 read Q. C^1 to Z. Clear K.

(61) read P. d to Q. P + 1 to P. F to K.

610 read Q. Fd to Q. Adv K.

611 read O. P to Z. Adv K.

612 read P. A to L. If C^5 Empty and Cent. $\overline{\text{Busy}}$ and C^1 or C^2 or C^3 or C^4 Empty (Fd to P. Adv K.)

613 read P. If C^5 full and C^4 empty (C^5 to Z. C^5 to C^4. P + 1 to P. Adv K.)

614 read P. If C^3 empty (C^4 to Z. C^4 to C^3. P + 1 to P. A + 1 to A. Adv K.)

615 read P. If C^2 empty (C^3 to Z. C^3 to C^2. P + 1 to P. Q - 1 to Q. Adv K.)

616 read P. If C^1 empty (C^2 to Z. C^2 to C^1. P + 1 to P. Adv K.)

617 read P. If Q = 0 (C^1 to Z. K = 713.)

If Q ≠ 0 and cent. $\overline{\text{busy}}$ and C^5 empty (C^1 to Z. Ato L. P + 1 to P. K = 613.)

(62) read P. d to Q. P + 1 to P. F to K.

620 read Q. If D^1 empty (Fd to D^1. Q + 1 to Q. Adv K.)

621 read Q. If D^2 empty (Fd to D^2. D^1 to D^2. Q + 1 to Q. Adv K.)

622 read Q. If D^3 empty (Fd to D^3. D^2 to D^3. Q + 1 to Q. Adv K.)

623 read Q. If D^4 empty (Fd to D^4. D^3 to D^4. Q + 1 to Q. Adv K.)

624 read Q. A to L. IF cent. $\overline{\text{busy}}$ (Fd to D^5. D4 to D^5. A to L. Clear K.)

(63) read P. d to Q. P + 1 to P. F to K.

630 read Q. Fd to Q. Adv K.

631 read O. P to Z. Adv K.

632 read P. Fd to P. Adv K.

633 read P. If D^1 empty (Fd to D^1. P + 1 to P. Adv K.)

634 read P. If D^2 empty (Fd to D^2. D^1 to D^2. P + 1 to P. Adv K.)

635 read P. If D^3 empty (Fd to D^3. D^2 to D^3. P + 1 to P. Q - 1 to Q. Adv K.)

636 read P. If D^4 empty (Fd to D^4. D^3 to D^4. P + 1 to P. Adv K.)

637 read P. A to L. If cent. $\overline{\text{busy}}$ (Fd to D^5. D^4 to D5. A + 1 to A. P + 1 to P)

IF cent. $\overline{\text{busy}}$ and Q = 0 (K = 733.)

If cent. $\overline{\text{busy}}$ and Q ≠ 0 (K = 633.)

(64) read P. d to Q. P + 1 to P. F to K.

640 read P. Fd to P if ch active. P + 1 to P if ch inactive. Clear K.

(65) read P. d to Q. P + 1 to P. F to K.

650 read P. Fd to P if ch inactive. P + 1 to P if ch active. Clear K.

(66) read P. d to Q. P + 1 to P. F to K.

660 read P. Fd to P if ch full. P + 1 to P if ch empty. Clear K.

(67) read P. d to Q. P + 1 to P. F to K.

670 read P. Fd to P if ch empty. P + 1 to P if ch full. Clear K.

(70) read P. d to q. P + 1 to P. F to K.

700 read P. R to A. If ch full (Clear K.)

(71) read P. d to Q. P + 1 to P. F to K.

710 read O. P to Z. Adv K.

711 read P. Fd to P. Adv K.

712 read P. If ch disconnect (Adv K.)

If ch full (R to Z. P + 1 to P. A - 1 to A. Adv K if A = 1.)

713 read O. Fd + 1 to P. Clear K.

(72) read P. d to Q. P + 1 to P. F to K.

720 read P. If ch empty and active (A to R. Clear K.)

(73) read P. d to Q. P + 1 to P. F to K.

730 read O. P to Z. Adv K.

731 read P. Fd to P. Adv K.

732 read P. If ch disconnect (Adv K.)

If ch empty active (Fd to R. P + 1 to P. A - 1 to A. Adv K if A = 1.)

```
733  read O.  Fd + 1 to P.  Clear K.

(74) read P.  d to Q.  P + 1 to P.  F to K.
740  read P.  If ch inactive (Activate ch.  Clear K.)

(75) read P.  d to Q.  P + 1 to P.  F to K.
750  read P.  If ch active (Disconnect ch.  Clear K.)

(76) read P.  d to Q.  P + 1 to P.  F to K.
760  read P.  If ch inactive (A to R.  Function.  Clear K.)

(77) read P.  d to Q.  P + 1 to P.  F to K.
770  read P.  If ch inactive (Fd to R.  Function.  P + 1 to P.  Clear K.)
```

APPENDIX D

Infinite/Indefinite Forms

6600
INFINITE/INDEFINITE FORMS

FLOATING ADD

(+ ∞) + (+ ∞) = 377700...00

(+ ∞) + (- ∞) = 177700...00

(- ∞) + (- ∞) = 400000...00

(- ∞) + (+ ∞) = 177700...00

(+ ∞) - (+ ∞) = 177700...00

(+ ∞) - (- ∞) = 377700...00

(- ∞) - (+ ∞) = 400000...00

(- ∞) - (- ∞) = 177700...00

(+ ∞) ± (± N) = 377700...00

(- ∞) ± (± N) = 400000...00

(± Indef.) ± (± N) = 177700...00

(± Indef.) ± (± ∞) = 177700...00

(± Indef.) ± (± 0) = 177700...00

Underflow = 0000 (coefficient = coefficient X_j ± coefficient X_k)

Overflow on right shift one = 3777XXX...XX (coefficient positive)

4000XXX...XX (coefficient negative)

MULTIPLY

(+ ∞) · (- N) = 400000...00

(+ ∞) · (+ N) = 377700...00

(- ∞) · (+ N) = 400000...00

(- ∞) · (- N) = 377700...00

(+ ∞) · (+ ∞) = 377700...00

(+ ∞) · (- ∞) = 400000...00

(- ∞) · (± 0) = 177700...00

MULTIPLY (cont.)

$(\pm\ 0) \cdot (\pm\ 0) = 000000...00$

$(\pm\ 0) \cdot (\pm\ N) = 000000...00$

$(\pm$ Indef.$) \cdot (\pm\ N) = 177700...00$

$(\pm$ Indef.$) \cdot (\pm\ \infty) = 177700...00$

$(\pm$ Indef.$) \cdot (\pm\ 0) = 177700...00$

Underflow: (no left shift one) = 000000...00

(left shift one & sign record) = 7777 (coefficient = coefficient X_j • coefficientX_k

(left shift one & no sign record) = 0000 (coefficient = coefficient X_j • coefficientX_k,

Overflow: #(sign record) = 400000...00

(no sign record) = 377700...00

\# Left shift one does not take the exponent out of overflow

DIVIDE

$(\pm\ 0\ ) / (\pm\ 0\ ) = 177700...00$

$(\pm\ \infty\ ) / (\pm\ \infty\ ) = 177700...00$

$(\ \infty\ ) / (\ N\ ) = 377700...00$

$(\ \infty\ ) / (-\ N\ ) = 400000...00$

$(-\ \infty\ ) / (\ N\ ) = 400000...00$

$(\pm\ 0\ ) / (\pm\ \infty\ ) = 000000...00$

$(\pm\ 0\ ) / (\pm\ N\ ) = 000000...00$

$(\pm\ N\ ) / (\pm\ \infty\ ) = 000000...00$

$(N\ \ ) / (\ 0\ ) = 377700...00$

$(-\ N\ ) / (\ 0\ ) = 400000...00$

$(\ N\ ) / (-\ 0\ ) = 400000...00$

$(-\ N\ ) / (-\ 0\ ) = 377700...00$

$(\pm$ Indef.$) / (\pm\ N\ ) = 177700...00$

DIVIDE (cont.)

($\pm$ Indef.) / ($\pm$ ∞) = 177700...00

($\pm$ Indef.) / ($\pm$ 0) = 177700...00

Underflow: # = 000000...00

Overflow: (right shift & sign record) = 4000 (coefficient = coefficient X_j/coefficient X_k

(right shift & no sign record) = 3777 (coefficient = coefficient X_j/coefficient X_k

\# Right shift one does not take the exponent out of underflow

NORMALIZE

(+ ∞) = 3777XX...XX B_j = 000000

(- ∞) = 4000XX...XX B_j = 000000

($\pm$ Indef.) = 1777XX...XX B_j = 000000

Underflow = 0000...00 B_j = Shift count

SUPPLEMENT TO TABLE OF INFINITE/INDEFINITE FORMS
Coefficient Fields for Indefinite Operands
In X_j And/Or X_k May be Any Value In Any Flt. Pt. Unit

FLOATING ADD UNIT USING 30, 31, 34 or 35 INSTRUCTION

X_j		X_k		X_i
3777000000000000000	+	3777000000000000000	=	3777000000000000000
3777000000000000000	+	4000000000000000000	=	1777000000000000000
4000000000000000000	+	4000000000000000000	=	4000000000000000000
4000000000000000000	+	3777000000000000000	=	1777000000000000000
3777000000000000000	-	3777000000000000000	=	1777000000000000000
3777000000000000000	-	4000000000000000000	=	3777000000000000000
4000000000000000000	-	3777000000000000000	=	4000000000000000000
4000000000000000000	-	4000000000000000000	=	1777000000000000000
3777000000000000000	+	1720600000000000000	=	3777000000000000000
3777000000000000000	+	6057177777777777777	=	3777000000000000000
3777000000000000000	-	1720600000000000000	=	3777000000000000000
3777000000000000000	-	6057177777777777777	=	3777000000000000000
4000000000000000000	+	1725700000000000000	=	4000000000000000000
4000000000000000000	+	6052077777777777777	=	4000000000000000000
4000000000000000000	-	1725700000000000000	=	4000000000000000000
4000000000000000000	-	6052077777777777777	=	4000000000000000000
1777000000000000000	+	1620450000000000000	=	1777000000000000000
1777000000000000000	+	6157327777777777777	=	
6000000000000000000	+	1620450000000000000	=	
6000000000000000000	+	6157327777777777777	=	
1777000000000000000	-	1620450000000000000	=	
1777000000000000000	-	6157327777777777777	=	
6000000000000000000	-	1620450000000000000	=	
6000000000000000000	-	6157327777777777777	=	1777000000000000000
1777000000000000000	+	3777000000000000000	=	1777000000000000000

MULTIPLY UNIT USING 40 OR 41 INSTRUCTION

X_j		X_k		X_i
3777000000000000000	•	57773177777777777777	=	40000000000000000000
3777000000000000000	•	20004600000000000000	=	37770000000000000000
40000000000000000000	•	20004600000000000000	=	40000000000000000000
40000000000000000000	•	57773177777777777777	=	37770000000000000000
37770000000000000000	•	37770000000000000000	=	37770000000000000000
37770000000000000000	•	40000000000000000000	=	40000000000000000000
37770000000000000000	•	00000000000000000000	=	17770000000000000000
37770000000000000000	•	77777777777777777777	=	17770000000000000000
40000000000000000000	•	00000000000000000000	=	17770000000000000000
40000000000000000000	•	77777777777777777777	=	17770000000000000000
00000000000000000000	•	17154370000000000000	=	00000000000000000000
77777777777777777777	•	17154370000000000000	=	00000000000000000000
00000000000000000000	•	60623407777777777777	=	00000000000000000000
77777777777777777777	•	60623407777777777777	=	00000000000000000000
17770000000000000000	•	20606543000000000000	=	17770000000000000000
17770000000000000000	•	57171234777777777777	=	17770000000000000000
60000000000000000000	•	20606543000000000000	=	17770000000000000000
60000000000000000000	•	57171234777777777777	=	17770000000000000000
17770000000000000000	•	37770000000000000000	=	17770000000000000000
17770000000000000000	•	40000000000000000000	=	17770000000000000000
60000000000000000000	•	37770000000000000000	=	17770000000000000000
60000000000000000000	•	40000000000000000000	=	17770000000000000000
00305000000000000000	•	16277000000000000000	=	00000000000000000000
00305000000000000000	•	61500777777777777777	=	00000000000000000000
77472777777777777777	•	16277000000000000000	=	00000000000000000000
77472777777777777777	•	61500777777777777777	=	00000000000000000000
07214000000000000000	•	07777000000000000000	=	00007000000000000000

MULTIPLY (Contd.)

X_j		X_k		X_i
70563777777777777777	•	07777000000000000000	=	77770777777777777777
30007000000000000000	•	27174000000000000000	=	37770000000000000000
30007000000000000000	•	50603777777777777777	=	40000000000000000000

DIVIDE UNIT USING 44 OR 45 INSTRUCTION

X_j		X_k		X_i
0000000000000000000	/	0000000000000000000	=	1777000000000000000
0000000000000000000	/	7777777777777777777	=	1777000000000000000
7777777777777777777	/	0000000000000000000	=	1777000000000000000
7777777777777777777	/	7777777777777777777	=	1777000000000000000
3777000000000000000	/	3777000000000000000	=	1777000000000000000
3777000000000000000	/	4000000000000000000	=	1777000000000000000
4000000000000000000	/	3777000000000000000	=	1777000000000000000
4000000000000000000	/	4000000000000000000	=	1777000000000000000
3777000000000000000	/	20424321000000000000	=	3777000000000000000
3777000000000000000	/	57353456000000000000	=	4000000000000000000
4000000000000000000	/	20424321000000000000	=	4000000000000000000
4000000000000000000	/	57353456777777777777	=	3777000000000000000
0000000000000000000	/	3777000000000000000	=	0000000000000000000
0000000000000000000	/	4000000000000000000	=	0000000000000000000
7777777777777777777	/	3777000000000000000	=	0000000000000000000
7777777777777777777	/	4000000000000000000	=	0000000000000000000
0000000000000000000	/	17347560000000000000	=	0000000000000000000
0000000000000000000	/	60430217777777777777	=	0000000000000000000
7777777777777777777	/	17347560000000000000	=	0000000000000000000
7777777777777777777	/	60430217777777777777	=	0000000000000000000
16717400000000000000	/	3777000000000000000	=	0000000000000000000
16717400000000000000	/	4000000000000000000	=	0000000000000000000
61060377777777777777	/	3777000000000000000	=	0000000000000000000
61060377777777777777	/	4000000000000000000	=	0000000000000000000
32044540000000000000	/	0000000000000000000	=	3777000000000000000
45733237777777777777	/	0000000000000000000	=	4000000000000000000
20615567000000000000	/	7777777777777777777	=	4000000000000000000
57162210777777777777	/	7777777777777777777	=	3777000000000000000

DIVIDE (Cont.)

X_j		X_k		X_i
1777000000000000000	/	1736754000000000000	=	1777000000000000000
1777000000000000000	/	6041023700000000000	=	1777000000000000000
6000000000000000000	/	1775667700000000000	=	1777000000000000000
6000000000000000000	/	6002110077777777777	=	1777000000000000000
1777000000000000000	/	3777000000000000000	=	1777000000000000000
1777000000000000000	/	4000000000000000000	=	1777000000000000000
6000000000000000000	/	3777000000000000000	=	1777000000000000000
6000000000000000000	/	4000000000000000000	=	1777000000000000000
0777600000000000000	/	2720400000000000000	=	0000000000000000000
3000600000000000000	/	0721400000000000000	=	3777600000000000000
4777177777777777777	/	0721400000000000000	=	4000177777777777777

NORMALIZE

	X_k	B_j	X_i
	37770043200000000000	000000	37770043200000000000
	40007734577777777777	000000	40007734577777777777
	17770002100000000000	000000	17770002100000000000
	60007775677777777777	000000	60007775677777777777
	00000000000000000000	000060	00000000000000000000
*	00000000000000000000	000060	00000000000000000000
	00040006000000000000	000011	00000000000000000000
	77777777777777777777	000060	00000000000000000000
*	77777777777777777777	000060	00000000000000000000
	77737777777777777777	000011	00000000000000000000
	20000000000000000000	000060	00000000000000000000
*	20000000000000000000	000060	17174000000000000000
	57777777777777777777	000060	00000000000000000000
*	57777777777777777777	000060	60603777777777777777

* Results due to rounded normalize

APPENDIX E

Refrigeration System

REFRIGERATION SYSTEM

The cooling system uses the principles of common mechanical refrigeration systems to dissipate component heat from the logic and storage modules. Chassis refrigerant temperature is maintained at 55° to 57° F.

The cooling system is discussed following some introductory comments on principles of refrigeration.

PRINCIPLES OF REFRIGERATION

Nearly all refrigeration systems are based on the principle that heat is absorbed during evaporation of a liquid. The device which gives up heat to the liquid is thus cooled. The change in state of a substance from liquid to gas or vice versa varies with temperature and pressure. For example, water boils at 100° C at normal atmospheric pressure but boils at a lower temperature under reduced pressures. Similarly, the boiling point of refrigerant 12 varies with pressure.

Fig. E-1 shows a simple refrigeration scheme whose main parts are:

1. Cooling unit or evaporator
2. Compressor
3. Condenser

The evaporator and condenser are partly filled with liquid refrigerant and the remaining parts with vapor. The compressor, a pump, runs continuously and removes vapors from the top of the evaporator and pumps them into the condenser. The low pressure in the evaporator and heat absorption cause the liquid to boil, thereby cooling the unit through evaporation.

The compressor action increases the pressure and thus temperature of the vapor. The hot vapor is condensed into a liquid by transferring heat to cold water running through coils in the condenser.

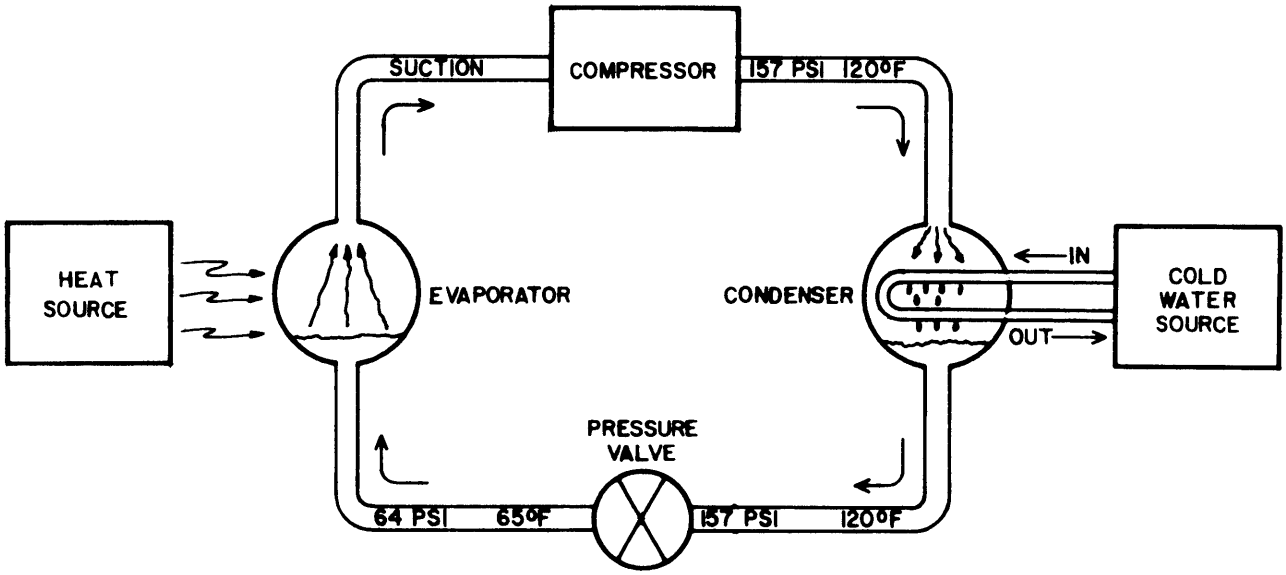


Fig. E-1. Simple Refrigeration Scheme

A pressure valve in the condenser to evaporator line regulates the pressure at which the liquid is returned to the evaporator. The pressure adjustment relates to the boiling point in the evaporator and hence to the temperature to be maintained. Thus, the liquid bouls to cool the evaporator when the heat load raises the temperature of the liquid above its boiling point as determined by the pressure. The compressor pumps the vapors from the boiling liquid into the condenser (and increases the pressure), where they are converted to the liquid state and returned to the evaporator under controlled pressure and temperature. The evaporator and an area surrounding it are thus maintained at a constant temperature.

SYSTEM DESCRIPTION

The cooling system contains the principal elements (evaporator, compressor, condenser) of common refrigeration systems plus additional regulating valves and gauges for visual monitoring of performance. The refrigerant is refrigerant 12; table E-1 gives its characteristics for saturated conditions. **The condenser is water cooled.**

Table E-1. Refrigerant 12 Characteristics

Temp. °F	Gauge Pressure
20	21 **PSI**
25	24.6
30	28.5
40	37
50	46.7
55	52
60	57.7
65	63.7
70	70.1
75	76.9
80	84.1
85	91.7
90	99.7
100	117.1
110	136.4
120	157.6
130	181

Physical Description

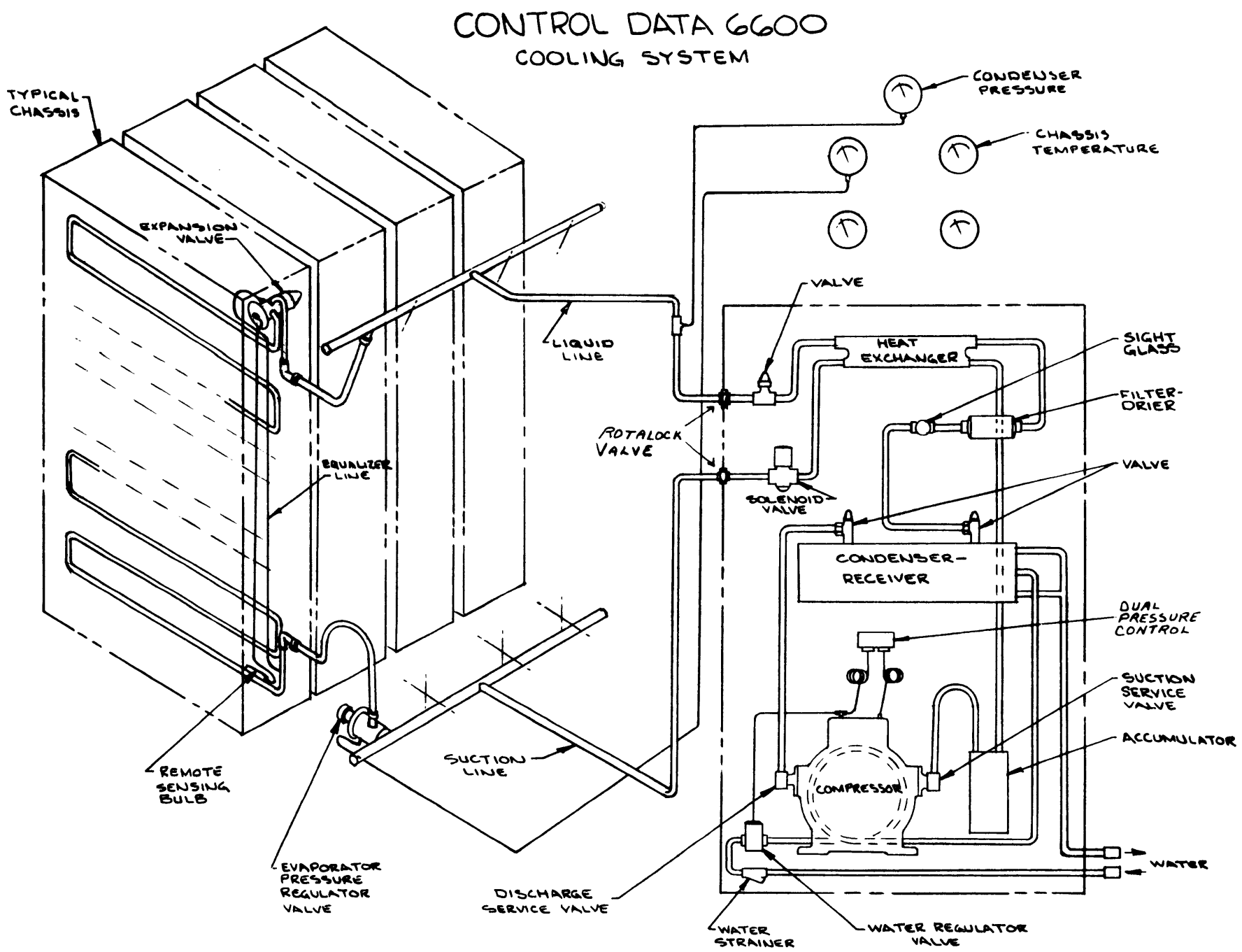
Fig. E-2 shows the various parts of the system. The evaporator and expansion valve are on each chassis as shown in fig. E-2 and E-3. Four chassis evaporators are connected in parallel to a common compressor, condenser, and related parts which are mounted in a frame assembly in the end of a wing. Each wing is a separate refrigeration system for four chassis with each chassis regulating its cooling with the varying logic and storage module heat load.

A control panel (fig. E-4) in each wing has separate temperature-pressure gauges for each chassis.

The evaporator coil is a copper tube imbedded in the aluminum bar that separates each row of modules in a chassis. Each module is secured to the aluminum bar and is separated from adjacent modules by black-finished metal spacers termed heat-sink plates. Each module thus fits in a separate compartment. Component heat from the modules flows by conduction and convection to the aluminum bar via the heat sink plates and the metal module cap.

Operation

Liquid refrigerant from the condenser enters the top of the chassis through an expansion valve. The expansion valve causes a sharp pressure drop and controls the refrigerant flow by reading the outgoing refrigerant temperature and pressure. The chassis refrigerant temperature is held at 55° F by means of a pressure regulator valve on the chassis outlet. Heat from the aluminum bar raises the refrigerant temperature above the boiling point, thereby producing evaporation and hence cooling of the aluminum bars. Each bar is termed a cold bar and is cool to the touch. The aluminum face plate on the storage module and the logic module, the thin heat sink plates between logic modules, the thick plates between storage modules, the air



11-25-63

Fig. E-2. 6600 Cooling System

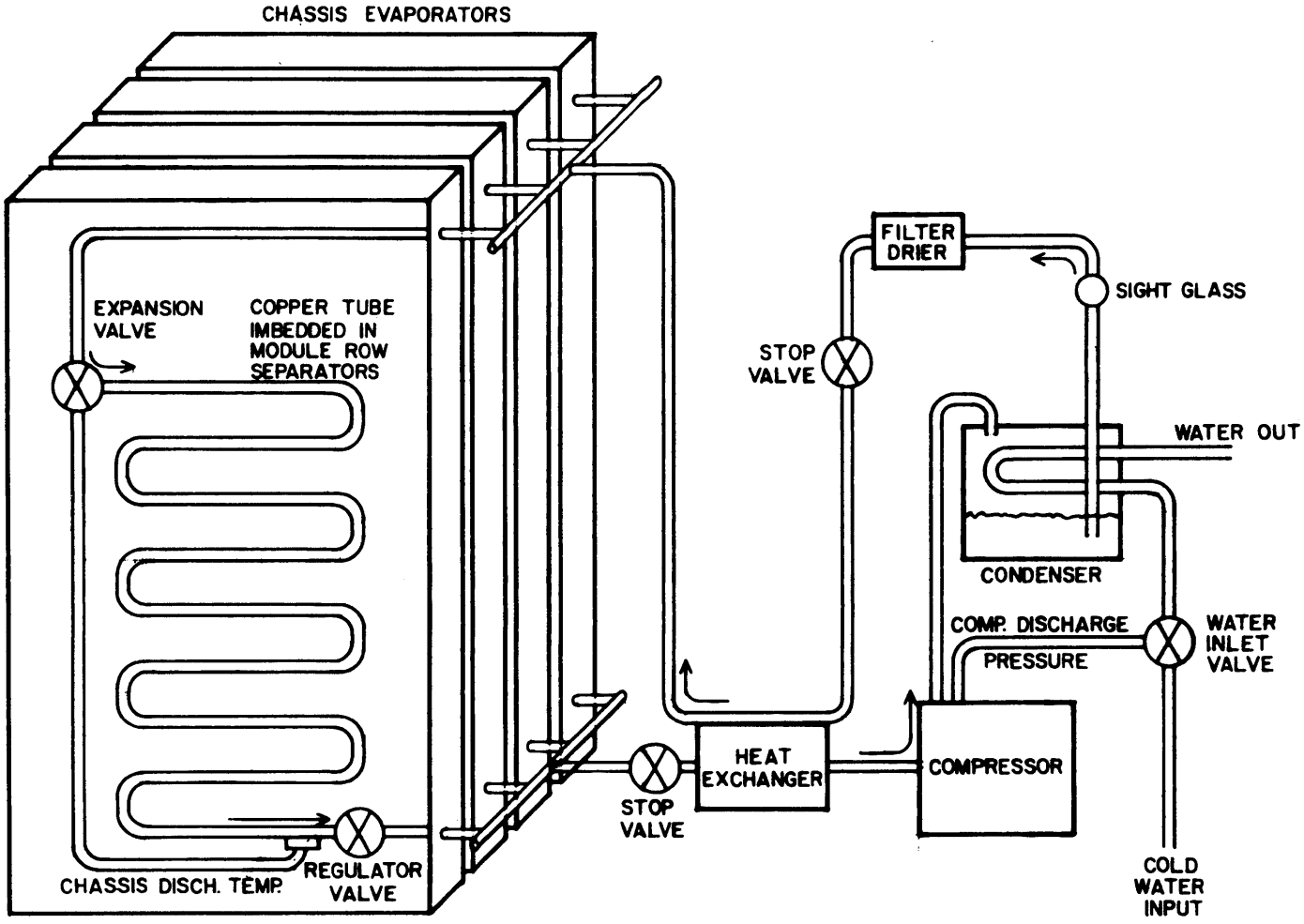


Fig. E-3. 6600 Cooling System - Simplified Diagram

between modules and plates, the soldered component leads and the printed circuit boards all form part of the heat conducting path from the discreet electronic component to the cold bar and then to the refrigerant.

The gaseous refrigerant returns to the compressor via the chassis pressure regulator valve at the bottom of the chassis (valve set to 55 psi) and a heat exchanger. The latter improves system efficiency and insures that no liquid refrigerant returns to the compressor during operation. The heat exchanger derives its heat from the hot liquid refrigerant produced by condensing the hot vapors from the high pressure (157 psi, 120° F) output of the compressor. The cold suction line which feeds the compressor passes through the heat exchanger jacket, resulting in a warming of the refrigerant vapor as it passes to the compressor and cooling of the liquid refrigerant.

ELEMENT DESCRIPTIONS

Accumulator. Prevents compressor slugging when certain room conditions cause liquid refrigerant to collect behind suction solenoid valve during a long shut-down period. Unit has built-in oil pick-up tube.

Compressor. High temperature, water cooled, semi-hermetic unit. 208/230 volt, 60 cycle, 3 phase motor. Unit has own current and temperature protection devices.

Condenser. Water cooled unit for 40° F to 90° F (maximum) water. Piped for cooling tower operation but satisfactory for well operation. Unit equipped with a 275 psi pressure relief valve or 162° F fusible plug.

Dual pressure control. Automatically shuts down compressor by opening the starter if the compressor discharge pressure rises above the set point or if the suction pressure falls below the set point. The high pressure cut-out is normally set at 190 psi (fixed 40 psi differential) and the low pressure at zero psi with a cut-in of 15 psi.

Evaporator pressure regulator. Valves keep a constant set pressure (normally 55° F) on the chassis evaporator it is controlling. The

chassis pressure/temperature gauges are connected to the valve.

Expansion valve. Valve controls the amount of refrigerant into its chassis evaporator regardless of load on the chassis by means of a thermostatic element attached to the bottom evaporator tube. Normal superheat adjustment of 4° to 8° F.

Filter-drier. Removes moisture from the refrigerant which enters the system by leaks, by opening the system for certain service operations, or by a less than adequate evacuation process. It also removes foreign particles introduced during the assembly process.

Rotalock valves. Valves of the cabinet liquid and suction lines. Used in the event of a condensing unit replacement for keeping the cabinet and chassis refrigerant lines from being opened to the atmosphere.

Sight glass. Gives a visual check on whether there is sufficient refrigerant in the system and also whether the system is dry. If the dot is green, the system is dry; if the dot is yellow, the system is wet.

Solenoid valve. Closed during shut-down to prevent refrigerant from migrating to the compressor. Solenoid is 115 volt, 60 cycle, and is connected to the incoming power side of the compressor starter. Unit is equipped with manual operating valve stem. Unit will hold pressure against chassis side only (not condensing unit side).

Starter. 3 phase compressor starter with 220 volt, 60 cycle coil. Unit does not have protection device for compressor.

Water regulating valve. An automatically regulating valve to keep the condensing temperature at a set value (normally 120° F). It also closes off water flow during unit shut-down. Valve is operated off the compressor head pressure.

WARNING SYSTEM

A warning system is built into each wing of the 6600 to protect the

modules against high temperature and condensation in case of a cooling system malfunction or adverse room air conditions.

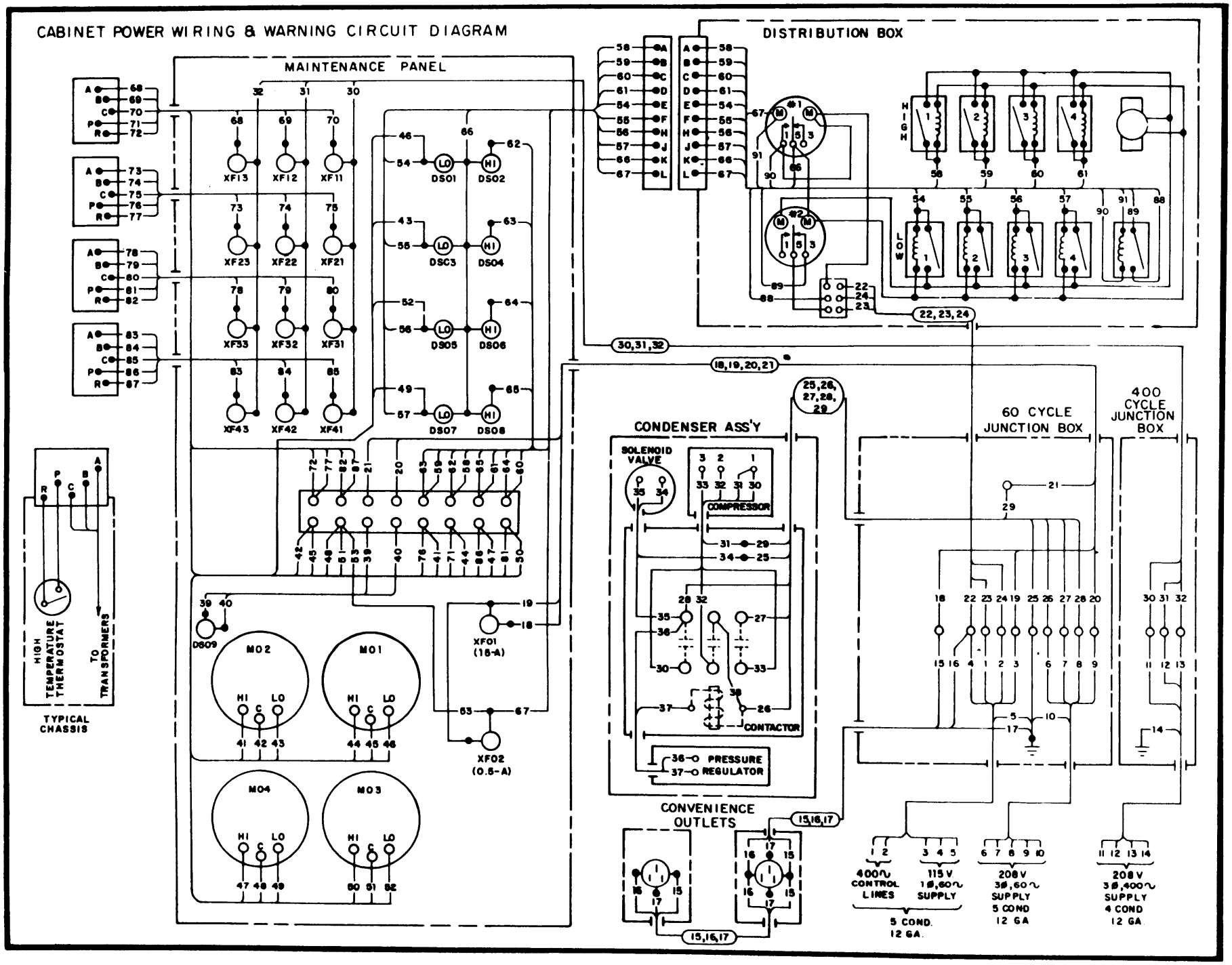
In the event of a malfunction a relay activates a warning buzzer and indicator light, and after a short period of time (approximately 2 minutes) the 400 cycle and 60 cycle power is automatically turned off.

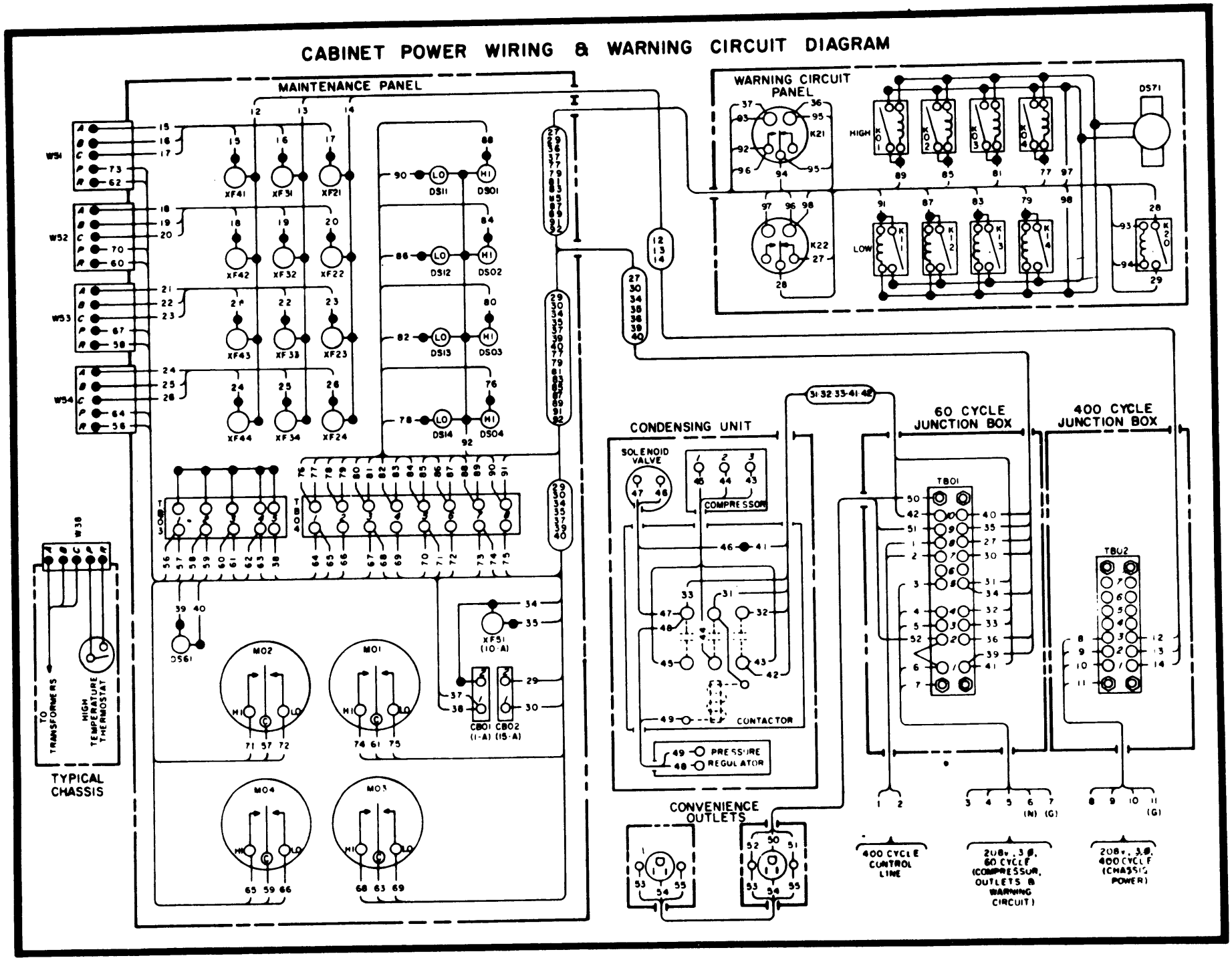
If the gauge pressure for any one chassis drops below the minimum setting of the red hand (53°), the gauge switch will start the sequence for operating the warning buzzer, the low temperature indicator for that chassis, and the time delay relay for shutting down the power. A low gauge pressure indicates a low temperature condition which could cause condensation on the chassis. See trouble K on the trouble-shooting chart.

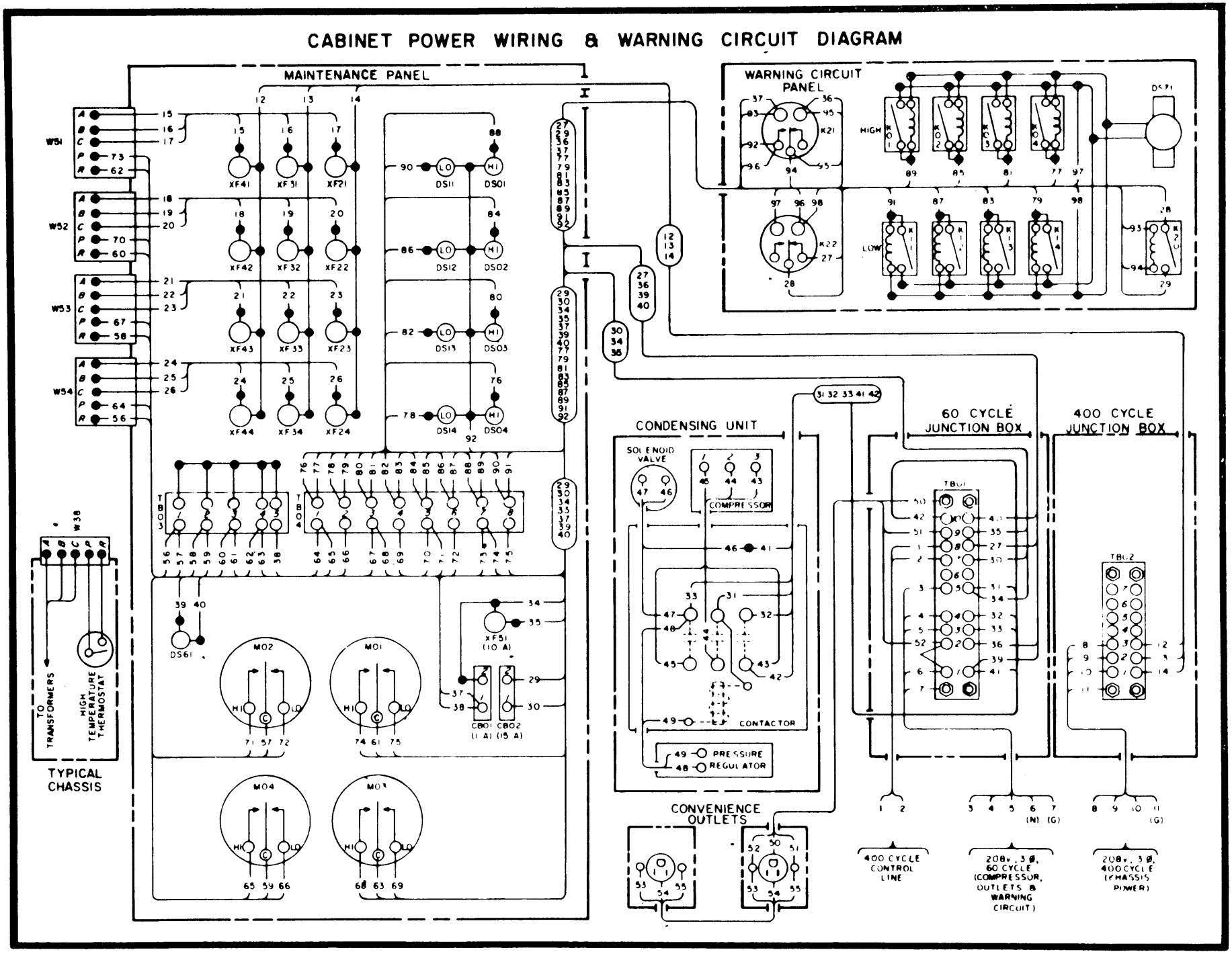
If the gauge pressure on any one chassis arises above the maximum setting of the red hand, or the thermostat on the bottom cold bar closes (at approximately 85° F), the buzzer, the high temperature indicator for that chassis, and the time delay relay are turned on. See trouble J on the trouble-shooting chart.

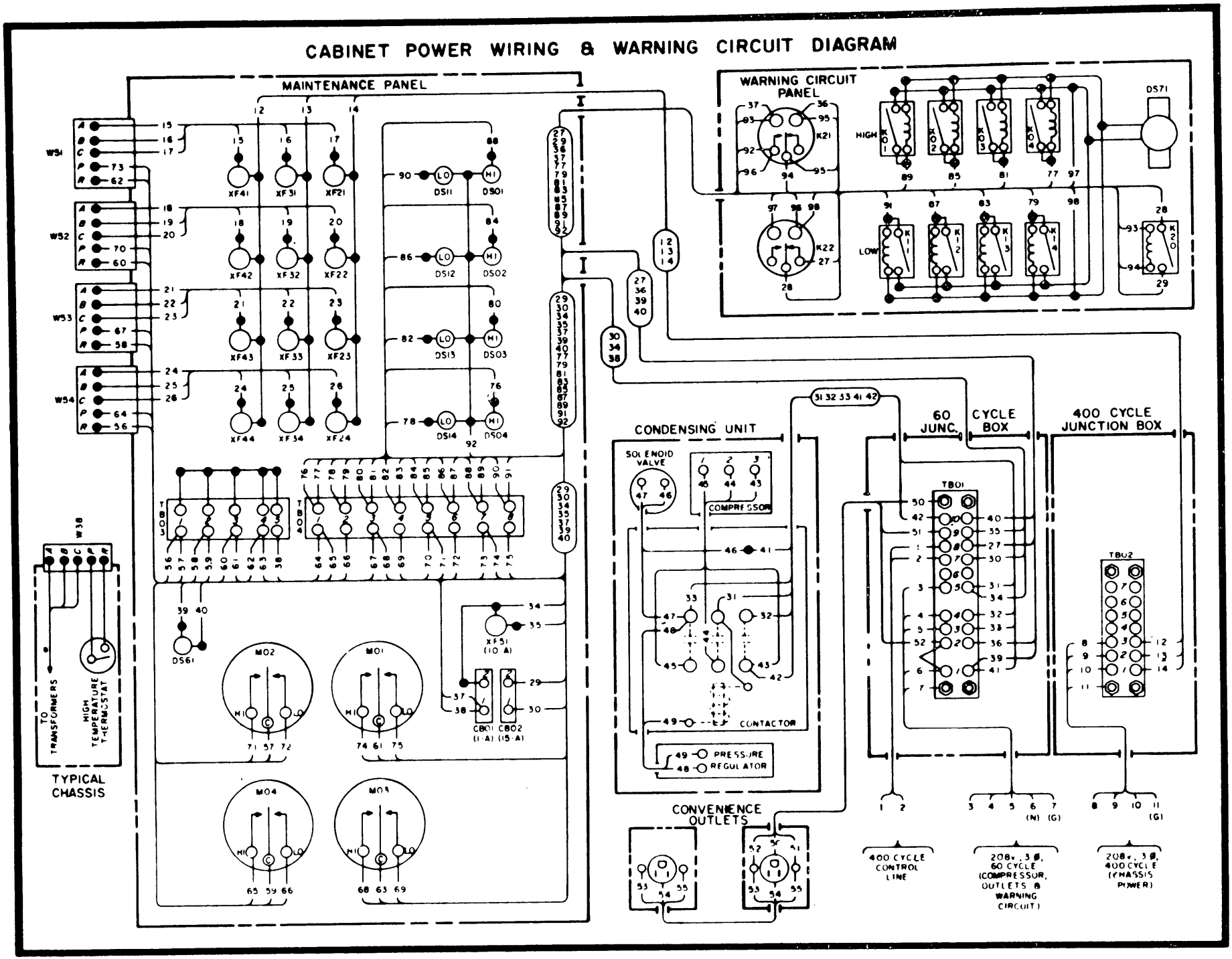
If trouble should be indicated it is very important that the settings on the gauges or time delay relays not be altered, as this could result in serious damage to the modules in the affected chassis.

When trouble is indicated the customer engineer should immediately look at the refrigeration maintenance panel and observe which chassis the cooling has failed on, whether the high or low temperature light is on, and whether the gauge for the chassis is the activating source. If more than one chassis temperature indicator is lit, observe the condenser gauge (it should be approximately 120°) and the compressor indicator light to see if the compressor is running or short cycling. If this information can be obtained before the machine shuts down, the source of trouble can probably be tracked down by use of the trouble-shooting chart.


CABINET POWER WIRING & WARNING CIRCUIT DIAGRAM
MAINTENANCE PANEL
DISTRIBUTION BOX
CONDENSER ASS'Y
SOLENOID VALVE
COMPRESSOR
CONTACTOR
PRESSURE REGULATOR
60 CYCLE JUNCTION BOX
400 CYCLE JUNCTION BOX
CONVENIENCE OUTLETS
HIGH TEMPERATURE THERMOSTAT
TO TRANSFORMERS
TYPICAL CHASSIS
XF13
XF12
XF11
XF23
XF22
XF21
XF33
XF32
XF31
XF43
XF42
XF41
DS01
DS02
DS03
DS04
DS05
DS06
DS07
DS08
DS09
XF01 (18-A)
XF02 (0.5-A)
M01
M02
M03
M04
HIGH
LOW
400∿ CONTROL LINES
115 V 1ø,60∿ SUPPLY
208 V 3ø,60∿ SUPPLY 5 COND 12 GA
208 V 3ø,400∿ SUPPLY 4 COND 12 GA
5 COND. 12 GA.
30,31,32
18,19,20,21
22,23,24
25,26,27,28,29
15,16,17


CABINET POWER WIRING & WARNING CIRCUIT DIAGRAM
MAINTENANCE PANEL
WARNING CIRCUIT PANEL
CONDENSING UNIT
SOLENOID VALVE
COMPRESSOR
CONTACTOR
PRESSURE REGULATOR
60 CYCLE JUNCTION BOX
400 CYCLE JUNCTION BOX
CONVENIENCE OUTLETS
TYPICAL CHASSIS
TO TRANSFORMERS
HIGH TEMPERATURE THERMOSTAT
400 CYCLE CONTROL LINE
208v, 3 Ø, 60 CYCLE (COMPRESSOR, OUTLETS & WARNING CIRCUIT)
208v, 3 Ø, 400 CYCLE (CHASSIS POWER)


CABINET POWER WIRING & WARNING CIRCUIT DIAGRAM
MAINTENANCE PANEL
WARNING CIRCUIT PANEL
CONDENSING UNIT
SOLENOID VALVE
COMPRESSOR
CONTACTOR
PRESSURE REGULATOR
60 CYCLE JUNCTION BOX
400 CYCLE JUNCTION BOX
CONVENIENCE OUTLETS
TYPICAL CHASSIS
TO TRANSFORMERS
HIGH TEMPERATURE THERMOSTAT
400 CYCLE CONTROL LINE
208v., 3 Ø, 60 CYCLE (COMPRESSOR, OUTLETS & WARNING CIRCUIT)
208v., 3 Ø, 400 CYCLE (CHASSIS POWER)


CABINET POWER WIRING & WARNING CIRCUIT DIAGRAM
MAINTENANCE PANEL
WARNING CIRCUIT PANEL
CONDENSING UNIT
SOLENOID VALVE
COMPRESSOR
CONTACTOR
PRESSURE REGULATOR
60 CYCLE JUNC. BOX
400 CYCLE JUNCTION BOX
CONVENIENCE OUTLETS
TYPICAL CHASSIS
TO TRANSFORMERS
HIGH TEMPERATURE T-STAT
400 CYCLE CONTROL LINE
208v, 3 Ø, 60 CYCLE (COMPRESSOR, OUTLETS & WARNING CIRCUIT)
208v, 3 Ø, 400 CYCLE (CHASSIS POWER)

Another part of the warning system, but not an inherent part of the computer, is a chart recorder assembly for the room temperature and dew point. This instrument is for a daily record of the room conditions and also has an audible (buzzer) warning device if the dew point goes above 50° F.

PREVENTIVE MAINTENANCE

Daily

Whenever the computer is started, listen for short cycling of any of the compressors.

Check the room conditions for the dew point and temperature. Do not run the computer if the dew point goes over the maximum of 50° F dew point. Serious computer damage can result if the room air dew point rises to the chassis temperature.

After the machine has run for approximately ten minutes, check the following on each maintenance panel:

1. Condenser pressure. Should read approximately 157 psig or 120° F and steady.

2. All the chassis gauges. They should be indicating well within the limit hands. They should read 55° to 57° F with the limit hands set at 53° F and 65° F.

Weekly

The daily routine plus the following:

1. The compressor oil level before startup and after running for an hour. If the oil level lowers to the bottom of the glass right after startup, it probably indicates one or more chassis are flooding (see troubles C-3 and G on trouble-shooting chart). The glass should always be between 1/4 and 3/4 full.

2. The liquid line sight glass should be checked for bubbles after the system runs for 30 minutes. If bubbles are present, or it is not running full, see trouble B-5. Also check the color of the dot in the center of the sight glass. If the dot is not green, but yellow or with a yellowish tinge, moisture has entered the system beyond the capacity of the drier. The leak must be found, repaired, and a new filter-drier installed. **The compressor will burn out if the system is allowed to run long in this condition.**

Yearly

1. Unless rust inhibited **and filtered water is used, the water should be** drained from the condensing system and a careful inspection made of the condenser tubes by removing the condenser end plate(s). Fouled tubes should be cleaned and scale removed.

2. The water pipe strainer on each condensing unit should be cleaned.

3. Check the condition of the contacts in the pressure control and starter on each condensing unit.

GENERAL REFRIGERATION SERVICE

Pump Down Procedure

1. Attach pressure gauge to compressor suction valve port and crack the valve off the back seat so the gauge will function.

2. Close the liquid line valve on the condenser.

3. Open the evaporator pressure regulator valves to the minimum setting (40 psi). This step is not necessary unless the liquid lines on the chassis end of the system are to be opened.

4. Compressor may be started and allowed to run until the suction gauge indicates 2 psig. At this point the compressor must be stopped and the compressor discharge line quickly closed. The valve on the condenser inlet should also be closed. The small pressure remaining in the lines must be left so that air and moisture will not enter when the system is opened.

Note: If the refrigerant lines to be opened are colder than the ambient dew point temperature, sweating will occur on the inside as well as the outside surface. Allow the piping to warm up to room temperature before opening it up to the air.

Filter-Drier Replacement

In the event that a filter-drier requires replacement, the condensing unit must be pumped down per the instructions, except that it is not necessary to open the pressure regulator valves. When the unit has been pumped down, close the valve in the liquid line, loosen the drier on the unit so it may be removed quickly, remove the seals on the new drier, and, as quickly as possible, remove the old drier and install the new. Put system back into operation and leak check joints around drier.

Adding refrigerant

Refrigerant should be added through the compressor suction valve port.

1. Connect refrigerant drum to the valve port with a charging line. Purge air from the line before tightening the hose connection onto the compressor valve.

2. Hold the refrigerant drum in a vertical position with the charging line at the top so that only gas can enter the system.

3. The compressor can be operated in the normal manner while charging. Crack the suction valve off the back seat and open the refrigerant drum valve. Gas can now enter the system.

4. Observe the sight glass. When the charge is adequate no bubbles will be seen and the bottom of the condenser will be cool. Shut off the drum valve and back seat the compressor valve.

5. Check the compressor oil level.

Purging

If air or other noncondensable gases are present in the system, they will tend to collect in the condenser. Noncondensables may enter when part of the system has been opened for servicing, such as drier replacement or addition of oil. When noncondensables are present, the head pressure will rise to a point above the pressure corresponding to the temperature at which the vapor is condensing.

To determine whether the system needs purging, the entire system must stand idle long enough to cool down to room temperature. This could best be done overnight, with the condensing unit door open and the panels off. When the system has attained room temperature the reading of the condenser gauge on the panel should not be more than 10 psi above the saturation pressure at room temperature.

To purge-pump down the system and close the compressor discharge valve after stopping the compressor. Allow the condenser water to run for approximately five minutes in order to condense as much vapor as possible. The water regulating valve will have to be opened all the way in order for the water to run. After the water has run for five minutes, purge through the compressor discharge port by opening the valve for approximately five seconds. Repeat the purging step several times at about three-minute intervals and with the condenser water running continuously. After purging, the system may be placed back into service.

Condensing Unit Replacement

1. Use the recommended pump down procedure.

2. After the unit has been pumped down, close the compressor suction valve, cabinet suction valve, cabinet liquid valve, and the condensing unit liquid line valve.

3. On Serial No. 1 only--disconnect condenser gauge line and insert plug.

4. Shut down the main power switch and disconnect the power cable between the cabinet distribution box and the condensing unit.

5. Remove condensing unit hold-down bolts in the bottom of the frame.

6. Disconnect the liquid and suction lines at the cabinet shut-off valves.

7. Slide the condensing unit out, taking care not to hook any tubing on either the cabinet or condensing unit.

8. Slide in replacement unit after removing the liquid and suction line caps. Place caps on the old unit.

9. Reverse Steps One through Six.

10. Place unit into operation, leak check, observe sight glass for sufficient charge, and check compressor oil level.

Servicing the Chassis Pressure Regulator Valve

If this valve will not hold its set pressure it may be obstructed (resulting in a high gauge pressure), stuck open (resulting in a low gauge pressure), or oil could be collected in the chassis (resulting in a low gauge pressure) due to a very light load on that chassis. In the first two cases the problem might be remedied by opening the valve to the maximum and then resetting. In the case of no load, this cannot be **permanently remedied without adding load to that chassis.**

If opening and resetting the valve does not appear to help, the gauge, gauge line, and actual chassis temperature should be checked before resorting to the valve replacement.

First, check the actual chassis temperature against the indicated gauge temperature. Do this on any cold bar except the bottom one. Temperature of

the bar should be not more than 5° F above the indicated gauge temperature. If there is a wide disparity, the gauge or gauge line is the problem, or the chassis is loaded with oil resulting in a much higher actual temperature than the gauge indicates.

Check the gauge line out by loosening the line at the gauge enough to see if a flow of refrigerant is obtained.

If the line is not plugged, shut off the gauge line valve on the regulator by removing the cap on the underside and close the gauge line valve with a screwdriver. Attach a service gauge in place of the panel gauge, open the gauge line valve, and observe the pressures obtained while the system is in operation. If satisfactory readings are obtained, reconnect to the panel gauge and recheck. If panel gauge is still erroneous, replace.

If the gauge line seems to be plugged it may be possible to blow it clear by disconnecting from the gauge and operating the system. If this is done the serviceman must be ready to close off the gauge line by the gauge valve very quickly or oil will be blown all over the unit. If this does not clear the line, both ends of the line must be disconnected and dry nitrogen blown through the line.

If it has been determined that valve replacement is required, proceed as follows:

1. Open the pressure regulator valve to be removed to the minimum setting.

2. Pump down system and close off the cabinet liquid and suction lines.

3. Remove the suction manifold cover, gauge line, hose connector, and the valve connector.

4. Replace valve with new one and make all connections. Open gauge line valve.

5. Open all valves and start system.

6. Set valve to 55-57° F, leak check, check sight glass and oil sump.

Expansion Valve Service

Superheat Setting

If it is suspected that the superheat setting is incorrect (see trouble shooting chart, troubles C-3, D-4, G-1 and I-1), it can easily be checked out by getting the temperature of the chassis lower hose fitting (place thermocouple on the plated tube of the fitting) and getting the temperature difference between it and the indicated gauge temperature. The actual temperature should be 4° to 8° F above the indicated gauge temperature. As the chassis evaporator is quite high, the temperature at the hose fitting may not be constant--take readings over a period of a few minutes.

If the superheat is incorrect (a large variation of 10 or 15° F indicates too high a superheat setting), remove the cap on the back of the valve (wired side of the chassis), being sure to place a wrench on the valve body to keep it from twisting. Turn in (clockwise) the valve stem to increase superheat; turn out to decrease superheat. When set, replace valve cap snugly.

Power (Thermostat) Element

The power element of the expansion valve consists of a remote bulb, capillary tube, and the diaphragm. If this power element has lost its charge, the valve will maintain a closed, or nearly closed, position (see trouble chart, troubles H-3 and D-3). If this is suspected, due to a high chassis temperature not gauge indicated or because of a very high superheat which is not adjustable, the power element can be checked as follows: Remove the bottom transformer cover and remove the bulb from the tube on the chassis. Hold the bulb in your hand and observe, by listening at the valve by stethoscope or by temperature at the lower hose connection, whether the valve opens up or not.

If the temperature does not drop down to a reasonable superheat, the power element must be replaced by the following method:

1. Pump down the system after screwing out all the valves.

2. Remove the top transformer cover, the hinge side cover, and remove the bulb capillary from its clamps.

3. Unscrew the valve diaphragm. Hold the valve body with a wrench to keep it from turning.

4. Replace with the new element. Coil the capillary and clamp as before. Clamp bulb to top of tube and insulate. Capillary coil must be coiled and taped so it will not rub itself or other parts. Capillary must also be drawn tightly through the clamps.

5. Start up the system, reset the pressure regulator valve, and leak check around the diaphragm.

6. Check and set superheat setting. Replace hardware.

Inspection and Servicing of Inner Valve Parts

If an expansion valve indicates that it is obstructed (see trouble D-3), the following procedure is required:

1. Pump down the system with all the pressure regulator valves opened up.

2. Slowly remove the bottom cap assembly as the pressure is relieved. Hold the valve body when loosening the bottom cap to keep the valve from twisting.

3. Inspect the inner parts, and especially the valve seat, for foreign matter. Clean out with a lintless rag.

4. Inspect the valve seat and pin for wear or cuts and replace if necessary. If the seat has to be replaced, the diaphragm must be removed to be able to get the push rods out of the way.

5. Reassemble valve, start system, and reset pressure regulator valves.

6. Leak check valve and adjust the superheat.

7. Check refrigerant charge.

External Equalizer Line

If the external equalizer becomes plugged or restricted (see troubles C-3 and G-1) the expansion valve will feed too much, resulting in a low superheat (possibly not adjustable) or none at all, with the possibility of flood-back to the compressor. If this condition is suspected, it can be checked out and possibly remedied by the following method:

1. Pump down the system after opening the affected chassis pressure regulator valve. Close off the cabinet liquid and suction lines.

2. Remove the equalizer line at the expansion valve and cap the valve connection.

3. Crack the liquid line valves momentarily to allow refrigerant to flow through the chassis and blow out the equalizer line.

4. If sufficient pressure cannot be applied in this manner, attach dry nitrogen to equalizer line and apply 250 psi. Do so in short timed spurts, as this will have to be purged off later.

5. Reconnect line, start system, and reset pressure regulator.

6. Check superheat setting, leaks, and observe sight glass.

TROUBLE SHOOTING CHART - 6600 COOLING SYSTEM

Complaint	Possible Cause	Symptoms	Recommended Action
A. Compressor does not start.	1. Open breaker	1. No current to line side of contactor	1. Determine why breaker was open--whether due to manual trip or overload on compressor.
	2. Burned-out motor	2. Full voltage on motor terminals but motor will not run.	2. Replace condensing unit assembly. Use standard cleaning procedures for chassis evaporators if required.
	3. Open motor protector	3. Full voltage to motor protector but not to motor terminals	3. Replace protector.
	4. Dual pressure control a. Faulty control b. Suction pressure cut-in setting c. Discharge pressure cut-in setting	4. Motor contactor coil not energized a. Pressures correct for settings but contact remains open. b. Suction pressure below cut-in setting c. Discharge pressure above cut-in setting	4.a. Repair or replace. b. Check for loss of refrigerant. Repair leak or replace condensing unit. See Complaint H. c. See Complaint E.
B. Compressor short cycles.	1. Low pressure control differential set too close	1. Normal operation except frequent stops and starts	1. Set differential to job conditions.
	2. High pressure control cut-out set too close	2. Normal operation except frequent stops and starts	2. Set to job conditions.
	3. Faulty condensing	3. Excessively high discharge pressure	3. Check for water failure. See Complaint E.
	4. Overcharge of refrigerant or noncondensable gas	4. High discharge pressure	4. Remove excess refrigerant or purge noncondensables.
	5. Low refrigerant charge	5. Normal operation except frequent stops and starts. Sight glass partially full or contains bubbles.	5. Repair refrigerant leak and recharge.
	6. Overcharge of compressor oil	6. Low suction and discharge pressure	6. Remove excess oil.

TROUBLE SHOOTING CHART - 6600 COOLING SYSTEM

Complaint	Possible Cause	Symptoms	Recommended Action
B. (cont.)	7. Water regulating valve inoperative or restricted or water temperature is too high.	7. High discharge pressure	7. Clean or repair valve or correct temperature of water.
	8. Water piping restricted or supply pressure too low	8. High discharge pressure	8. Correct cause.
	9. Restricted drier-strainer	9. Low suction pressure and frosting at strainer	9. Replace strainer.
	10. Fouled condenser	10. High discharge pressure, cool outlet water	10. Clean condenser tubes.
C. Compressor is noisy.	1. Lack of oil	1. Oil level cannot be seen in glass.	1. Add oil.
	2. Internal compressor parts broken	2. Compressor knocks.	2. Replace condensing unit assembly.
	3. Liquid flood-back	3. Abnormally cold suction line. Compressor knocks.	3. Check expansion valve superheats. Check exp. valve bulbs for being loose. Check chassis gauges for low pressure. Check for restricted exp. valve external equalizer.
	4. Dirty water regulating valve. Too high water pressure or intermittent water pressure.	4. Water valve hammers or chatters.	4. Clean water regulating valve. Install air chamber ahead of valve.
D. System short of capacity (one or more chassis evaporators running too warm)	1. Flash gas in liquid line	1. Expansion valve hisses. Shows in sight glass.	1. Add refrigerant or lower condensing temperature.
	2. Clogged Drier-strainer	2. Temperature drop through strainer	2. Replace strainer
	3. Expansion valve stuck or obstructed or bad power element	3. Chassis gauge pressure correct but chassis is several degrees warmer.	3. Clean or repair expansion valve.
	4. Improper superheat adjustment	4. Chassis gauge pressure correct but bottom part of chassis is several degrees warmer than top part.	4. Check expansion valve bulb for looseness. Reset superheat setting of valve.

TROUBLE SHOOTING CHART - 6600 COOLING SYSTEM

Complaint	Possible Cause	Symptoms	Recommended Action
D. (cont.)	5. Worn compressor valves	5. High suction pressure, nearly equal to discharge	5. Install new compressor valve & valve plate.
	6. Chassis pressure regulator valve stuck or obstructed.	6. Chassis gauge pressure too high and chassis too warm	6. If valve won't regulate, replace valve.
E. Condenser pressure too high	1. Too little or too warm condenser water	1. Excessively warm water leaving condenser	1. Provide adequate cool water. Adjust water valve.
	2. Fouled tubes in condenser	2. Excessively cool water leaving condenser	2. Clean tubes.
	3. Air or noncondensable gas in system	3. Exceptionally hot condenser and excessive discharge pressure	3. Purge.
	4. Overcharge of refrigerant	4. Exc. disch. press. Cond. excessively hot. (Bottom 3d or less should be cool.)	4. Purge.
F. Condenser pressure too low	1. Too much condenser water	1. Cold water leaving condenser	1. Adjust water regulating valve.
	2. Lack of refrigerant	2. Bubbles in sight glass	2. Repair leak and charge or replace condensing unit.
	3. Broken or leaking compressor discharge valve	3. Suction pressure rises faster than 5 lbs./min. after shutdown (can also happen with leaking suction solenoid valve).	3. Repair compressor or replace condensing unit.
G. Suction pressure too high	1. Overfeeding of expansion valve	1. Abnormally cold suction line. Liquid flood-back	1. Check exp. valve remote bulb for being properly attached. Adjust superheat setting. Check for restricted exp. valve ext. equalizer.
	2. Expansion valve stuck in open position	2. Abnormally cold suction line. Liquid flood-back	2. Repair or replace valve.
	3. Pressure regulator valve stuck in open position	3. Abnormally cold suction line. Liquid flood-back. Low chassis gauge pressure	3. Tap valve lightly or replace.

TROUBLE SHOOTING CHART - 6600 COOLING SYSTEM

Complaint	Possible Cause	Symptoms	Recommended Action
C. (cont.)	4. Broken or leaking valves in compressor	4. Noisy compressor or rapid pressure change after shutdown	4. Replace valves on compressor or replace condensing unit assembly
H. Suction pressure too low	1. Lack of refrigerant	1. Bubbles in sight glass	1. Repair leak or replace condensing unit assembly.
	2. Clogged drier-strainer	2. Temp change of refrigerant line through strainer	2. Replace drier-strainer.
	3. Loss of charge in expansion valve power assembly	3. No refrig. flow through valve. Chassis exc. warm	3. Replace expansion valve power assembly.
	4. Obstructed expansion valve	4. Loss of capacity. Chassis is excessively warm.	4. Clean or repair valve.
	5. Obstructed chassis pressure regulator valve	5. Loss of capacity. Chassis gauge pressure is high.	5. Replace valve.
I. Compressor knock on start-up and frosting suction manifold	1. Expansion valve stuck in open position	1. Gauge pressure correct. Frost on suction line. Rattling knock of compressor on start-up	1. Check expansion valves for superheat. Clean or replace internal parts.
J. Buzzer and high temperature light for one chassis	1. Expansion valve obstructed or stuck in closed position	1. Chassis gauge pressure correct but chassis has abnormally high temperature. (High temp thermostat is controlling.)	1. Clean out or replace internal parts of expansion valve.
	2. Pressure regulator valve obstructed or stuck in closed position	2. Chassis gauge pressure abnorm. high & chassis temp abnorm. high. (Gauge or thermostat is controlling.)	2. Open valve to max. and reset or replace valve.
K. Buzzer and low temperature light for one chassis	1. Pressure regulator valve stuck in open position	1. Chassis gauge pressure abnormally low and chassis temperature abnormally cold. (Gauge is controlling.)	1. Open and close valve to maximum settings and reset or replace valve.
L. Noisy pressure regulator valve	1. Adjustment spring	1. Noisy valve(s) with noticeable vibrations	1. Remove adjusting cap; lightly grease spring and brass washer (end for end spring).

CUT ALONG LINE

COMMENT SHEET

CONTROL DATA 6600 TRAINING MANUAL
Pub. No. 60147400

FROM NAME : _____

BUSINESS ADDRESS : _____

COMMENTS: (DESCRIBE ERRORS, SUGGESTED ADDITION OR DELETION AND INCLUDE PAGE NUMBER, ETC.)

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
8100 34TH AVENUE SOUTH
MINNEAPOLIS 20, MINNESOTA

ATTN: TECHNICAL PUBLICATIONS DEPT.
COMPUTER DIVISION
PLANT TWO

FOLD

FOLD

CUT ALONG LINE

STAPLE

STAPLE

CONTROL DATA
CORPORATION
8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440

PRINTED IN U.S.A.