IBM

Customer Engineering Manuals Supplement

System/Unit 7094

Re: Form No. R23-2550-1

This Supplement No. S23-4000

Date October 1, 1962

Previous Supplement Nos. None

This publication is a supplement to the IBM 7094 Data Processing System, Customer Engineering Instruction Reference Manual, (Form R23-2550-1). The inclosed section on Instruction Overlap replaces the corresponding section in the manual.

International Business Machines Corp., Product Publications Dept., Poughkeepsie, N. Y.

PRINTED IN U.S.A.

IBM

Customer Engineering Manuals Supplement

System/Unit 7094

Re: Form No. R23-2550-1

This Supplement No. S23-4000

Date October 1, 1962

Previous Supplement Nos. None

This publication is a supplement to the IBM 7094 Data Processing System, Customer Engineering Instruction Reference Manual, (Form R23-2550-1). The inclosed section on Instruction Overlap replaces the corresponding section in the manual.

International Business Machines Corp., Product Publications Dept., Poughkeepsie, N. Y.

PRINTED IN U.S.A.

INSTRUCTION OVERLAP

Table of Contents

INSTRUCTION OVERLAP

The 7094 overlap feature significantly reduces the number of machine cycles required for computer operation. The number of cycles for certain pairs of instructions is reduced from four to three. Certain other pairs of instructions may be performed in full overlap, i.e., no cycle time is required for the second instruction and the two instructions are performed in the time normally required for the first one. Proper programming, however, is necessary to realize the full benefit of the instruction overlap feature.

Reduction in execution time is accomplished by bringing two sequential instructions to the computer from core storage during the same instruction cycle. The first instruction is placed into the program register in the normal manner while the next instruction is placed into an Instruction Backup Register (IBR) where it is immediately available for testing or use. When the even instruction is completed, the instruction in the IBR is routed to the program register (with no I cycle reference needed to core storage) and the computer proceeds as normal.

Three types of overlap (lookahead) are possible; the general rules for each are:

1. Store Lookahead (SLA) - If the instruction in the even core storage location is a store type instruction not attempting to store in location n+1, the execution time of the next odd location instruction is reduced by one cycle.

2. Data Lookahead (DLA) - If the instruction in the even location requires two or more cycles, (excluding store type instructions), is indexable and the next instruction in the odd location requires a data fetch from core storage, the execution time of the odd instruction is reduced by one cycle.

3. Transfer Lookahead (TLA) - If the instruction in the even core storage location is a two cycle instruction (Op codes 2X, 3X, 4X, or 5X) and the next instruction is a one cycle conditional or unconditional transfer (TIX, TNX, TXH, TXL, TXI, TSX, TRA, or TTR), the transfer instruction is executed during the E cycle of the even instruction and no cycle time is required for the transfer instruction.

The sequence of an overlap operation is shown in Figure The decision as to whether overlapping can be performed is made in a series of steps.

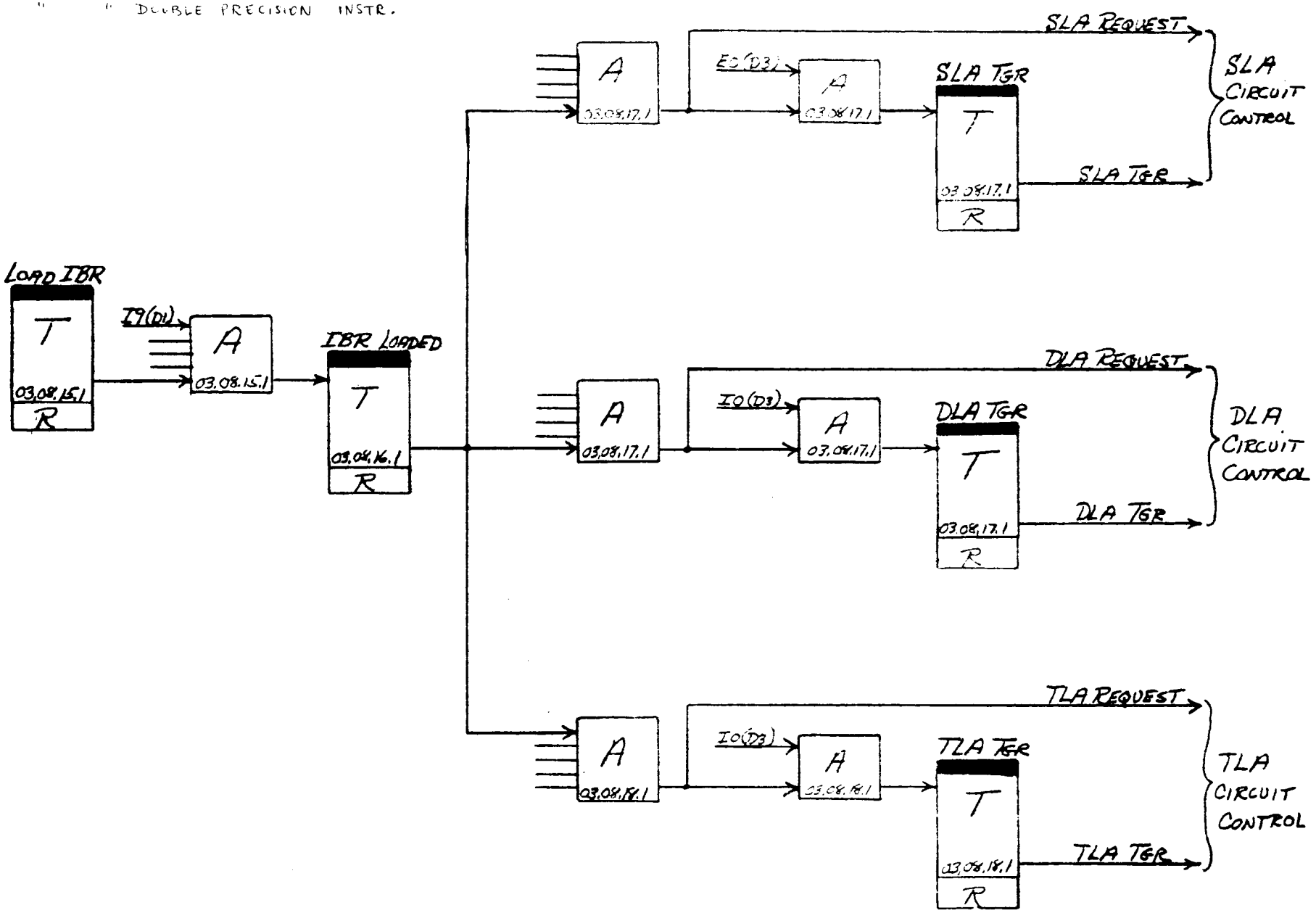
1. A split data-out-gate (DOG) is provided to core storage to enable two words to be sent to the computer. (Even though split DOG provides two words, the second one may be disregarded at the computer.)

2. The "load IBR" trigger is turned ON to signal that it is logically possible to accept and load the second word that is arriving on the storage bus.

3. The IBR loaded trigger being turned ON indicates that the next sequential instruction is in the IBR and that overlapping is possible. In many cases, the IBR is loaded with the second instruction but a last minute decision is made to nullify the overlap condition. In these cases, the IBR is not immediately reset; instead, the IBR loaded trigger is turned OFF signalling that the IBR is not "logically" loaded.

The IBR loaded trigger being ON allows SLA, DLA, or TRA request conditions to become active to initiate the particular circuitry applicable to that type of overlapping.

CAN NOT OVERLAP ON SINGLE CYCLE INSTRUCTIONS
" " " " DOUBLE PRECISION INSTR.



OVERLAP CONTROL SEQUENCE

FIGURE 1

OVLP3

Bo 9/62

Split Data-Out-Gate (Figures &)

Core storage is capable of gating our two 36-bit words on one storage cycle, one from the even location and one from the next higher odd location. This is the result of a modification of the data-out gate circuitry and a "split DOG" signal from the computer.

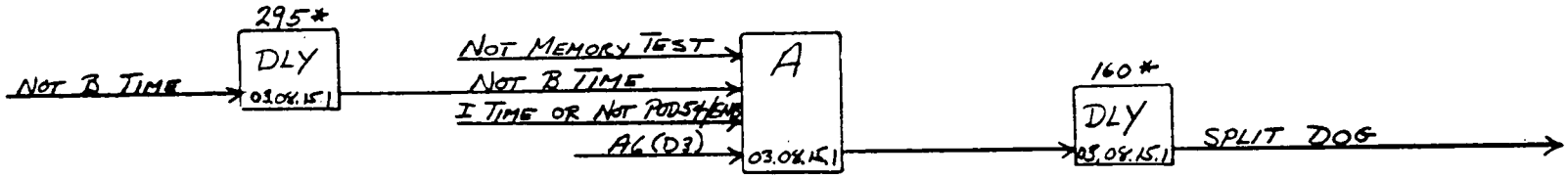
With two instructions in the memory data register (MDR), the even addressed instruction is placed on the storage bus and set into the computer during the period of I6-I9 time. At approximately I9-time the computer "split DOG" signal switches the odd addressed instruction onto the storage bus where it is sampled into the IBR. "Split DOG" originates as an A6 (D3) pulse in the computer but is delayed by an appropriate amount to accomplish the switch at 9-time.

Notice that this "split DOG" signal occurs on cycles other than I cycles. Two data words occurring on the storage bus during E (data fetch) cycles, for example, cause no problem; the desired data is loaded at E6-time in the normal manner and the second word arriving at 9-time is ignored. The only time that the split DOG signal is blocked is when it would be detramental to have two data words on the storage bus during the same cycle; such as:

1. During B cycles when the channel is requesting data from core storage. Data switching during the latter portion of this cycle produces an OR'ing condition at the channel registers.

2. During the E cycles of either ENB or POD 54 instructions. Switching during these cycles, as in the case of B cycles, also produces an OR'ing problem at the channel.

3. During memory test conditions. During testing from the CE panel, only the 36 bits are tested that correspond to the address indicated in MAR. If "split DOG" were not blocked, all 72 bits would be checked during the same memory cycle and the computer could stop as the result of an error not corresponding to the indicated address.

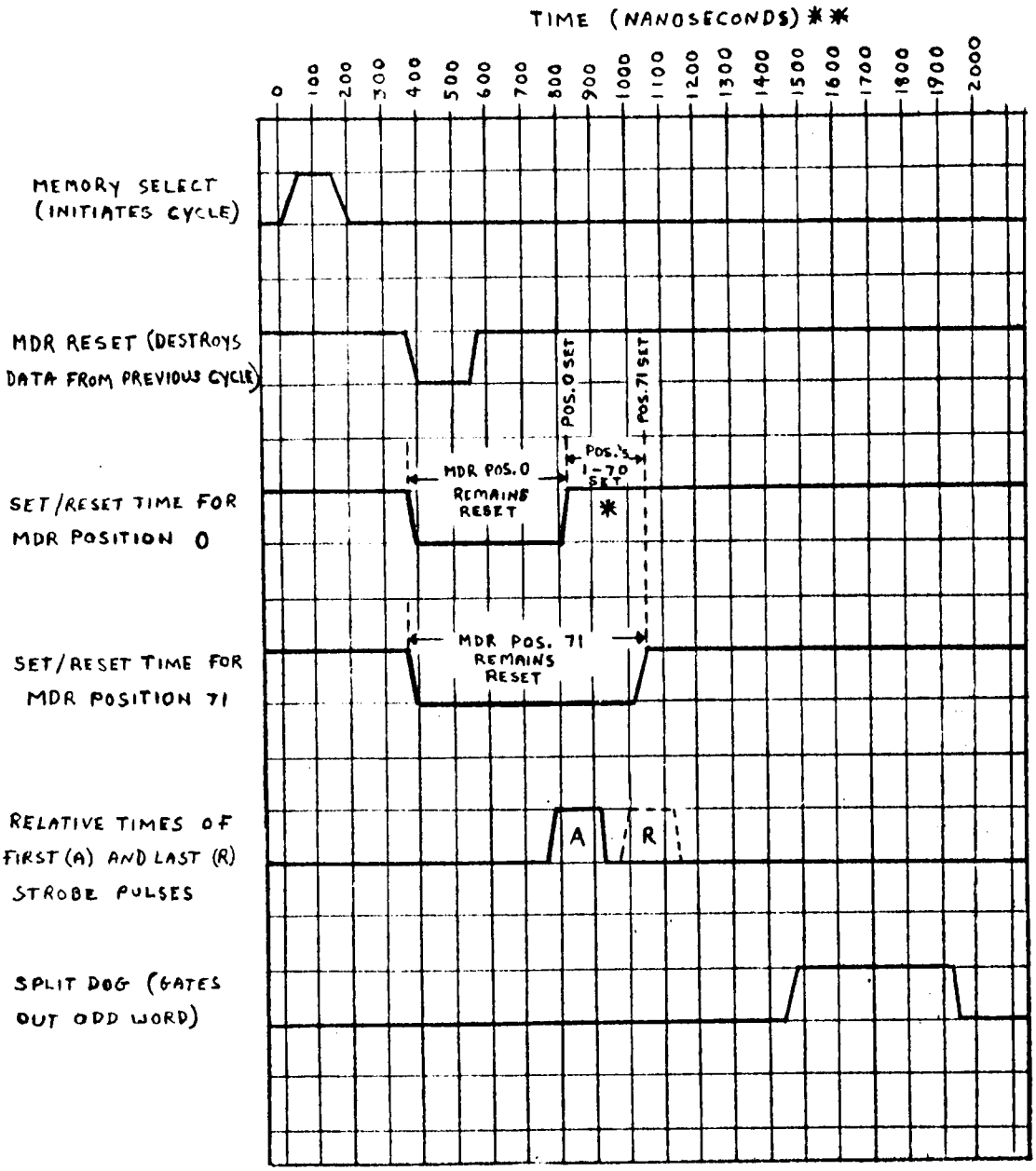


SPLIT DATA OUT GATE

FIGURE _____

OVLP 5

B 9/82



* NOTE: IF MAR 3 IS 0, THE LOWER WORD (0-35) IS PLACED ON MDBO LINES BY 940 ns. SPLIT DOG THEN GATES THE UPPER WORD (36-71) AT 1500 ns. IF MAR 3 IS 1, THE UPPER WORD IS PLACED ON MDBO LINES BY 1050 ns. THE LOWER WORD CAN NOT BE GATED OUT.

** NOTE: ALL TIMES ARE APPROXIMATE.

CORE STORAGE TIMINGS

FIGURE _____

Loading the IBR (Figure)

As stated in the previous section, "split DOG" provides two words on the storage bus almost every reference cycle. The decision now has to be made at the computer whether or not these two instructions are to be accepted. This decision is provided with the "load IBR" trigger. The primary condition at this point is that core storage has referenced an even location; it is only with this condition that a next higher odd instruction can be obtained simultaneously.

Memory diagnostic mode prevents loading of the IBR; MAR 3 & 17 lines to core storage revert the storing and addressing scheme to that of the 7090 making overlap impossible. Manual operations also prevent overlap; therefore, single stepping and machine cycling operate under non-overlap conditions.

At I9-time, "split DOG" has caused the second instruction to be placed on the storage bus. It is at this time, then, that the load IBR trigger allows the IBR to be set with the odd instruction. Computer circuitry is informed of this fact by the "IBR loaded" trigger.

Without the IBR loaded trigger ON, all types of overlap are prevented. In many cases the IBR is loaded on the assumption that overlap is possible only to decide at the last moment that it must be blocked. There are numerous conditions, then, that cause a reset to the IBR loaded trigger. Some of the more important resetting conditions are:

1. If the even instruction has an execution time of one cycle. This includes: the 1-cycle sense indicator instructions; many of the transfer instructions; XEC, XCA, XCL; and shift instructions which can be completed within one cycle because of a shift of six positions or less.

2. If the even instruction is non-indexable. This, as in the previous case, covers many 1-cycle instructions. In addition to that, however, some of the non-indexable

instructions require use of the index adders at a time when they would also be required by the overlapping instruction.

3. If the even instruction requires use of the IBR. Cases of this would be: ERA, ANS and all double precision arithmetic instructions.

4. If the even instruction results in a skip that would bypass the next odd instruction in the IBR. Instructions of this type would be: skip producing SI instructions (RFT, LFT, OFT, ONT); POD 76 type sense instructions such as sense light tests, sense switch tests, LBT, IOT, DCT, PBT; storage tests such asNZT and ZET. POD 34 (CAS and LAS) instructions are special cases where additional circuitry has been added to accommodate their various skips.

STORE OVERLAP (Figure)

If the instruction at the even memory location is a store type instruction and the address in this store instruction is not n+1, the execution time of the next instruction is reduced by one cycle.

In their execution, store instructions require an E cycle to route the required information to the specified core storage location. This information is sent to storage during the early portion of the E cycle and, for all practical purposes, the operation is completed; no useful work is accomplished during either the remaining half of the E cycle or the first half of the following I cycle.

Cyclic makeup of the SLA operation is I - E/I. At 6 time of the E cycle a transition is made to I time; the instruction is routed into the program register from the IBR, and the computer proceeds as normal from that point. Notice that there is no index register or address register modification performed in the overlapped condition as explained for TLA or DLA. Instead, execution time of the even-odd location instructions is reduced by one cycle by eliminating two adjacent half cycles (last half of the store E cycle and first half of the next I cycle).

There are no restrictions on tagging or indirect addressing for either the even or odd addressed instruction; either or both may be tagged and indirectly addressed. If the even addressed store instruction is indirectly addressed, however, overlapping and cycle transition is delayed for one cycle producing I - E(IA) - E/I.

One major restriction on SLA is that the even addressed instruction is not attempting to store its contents in the next sequential location. If this is the case, the instruction, which is now in the IBR, is being modified by the program and, therefore, invalid. A test for this situation is made during the early portion of the E cycle. If a storage reference is being made to n+1, SLA is nullified and the computer proceeds as though no overlap had been initiated.

SLA Operation (Figure)

The IBR loaded trigger is turned ON at I9 time of the even addressed instruction. 6X decoding from the program register allows "SLA request" to become active at the beginning of the E cycle and the SLA operation continues. One restriction at this point is that the 6X is not a double store operation. This instruction requires two consecutive E cycles to complete its operation and would complicate the SLA circuitry if allowed to proceed.

If the store instruction is indirectly addressed, normal E-time circuitry is blocked and completion of the operation delayed for one cycle. 0-time of the normal E cycle turns ON the SLA trigger.

At the same time that the store instruction is routing the required information to storage, a test is made to determine if that storage location is n+1. During the initial I cycle, the program counter was stepped at 8-time to indicate the next sequential instruction. Also, during the latter portion of the same I cycle, storage register positions 21-35 were routed to the address register and modified to indicate the correct core storage reference address for storing the information. If the program is storing into n+1, both the program counter and address register contain the same value.

Checking is performed by routing both the complement of the program counter and the true address register values to the index adders. The complement output of the index adders is then routed to the storage register input to take advantage of the zero checking circuitry. No set pulse is generated; therefore, the storage register contents remain unchanged. If the program counter and address register both contained the same value, the index adders will contain all 1's and the complement output all 0's. If a zero condition does exist, the SLA trigger is reset at E5 time and no overlapping takes place; instead, the computer proceeds as it would under non-overlap conditions.

Considering that a zero condition does not exist, a cycle transition is made at 6-time by resetting the master E time trigger and turning ON the master I time trigger.

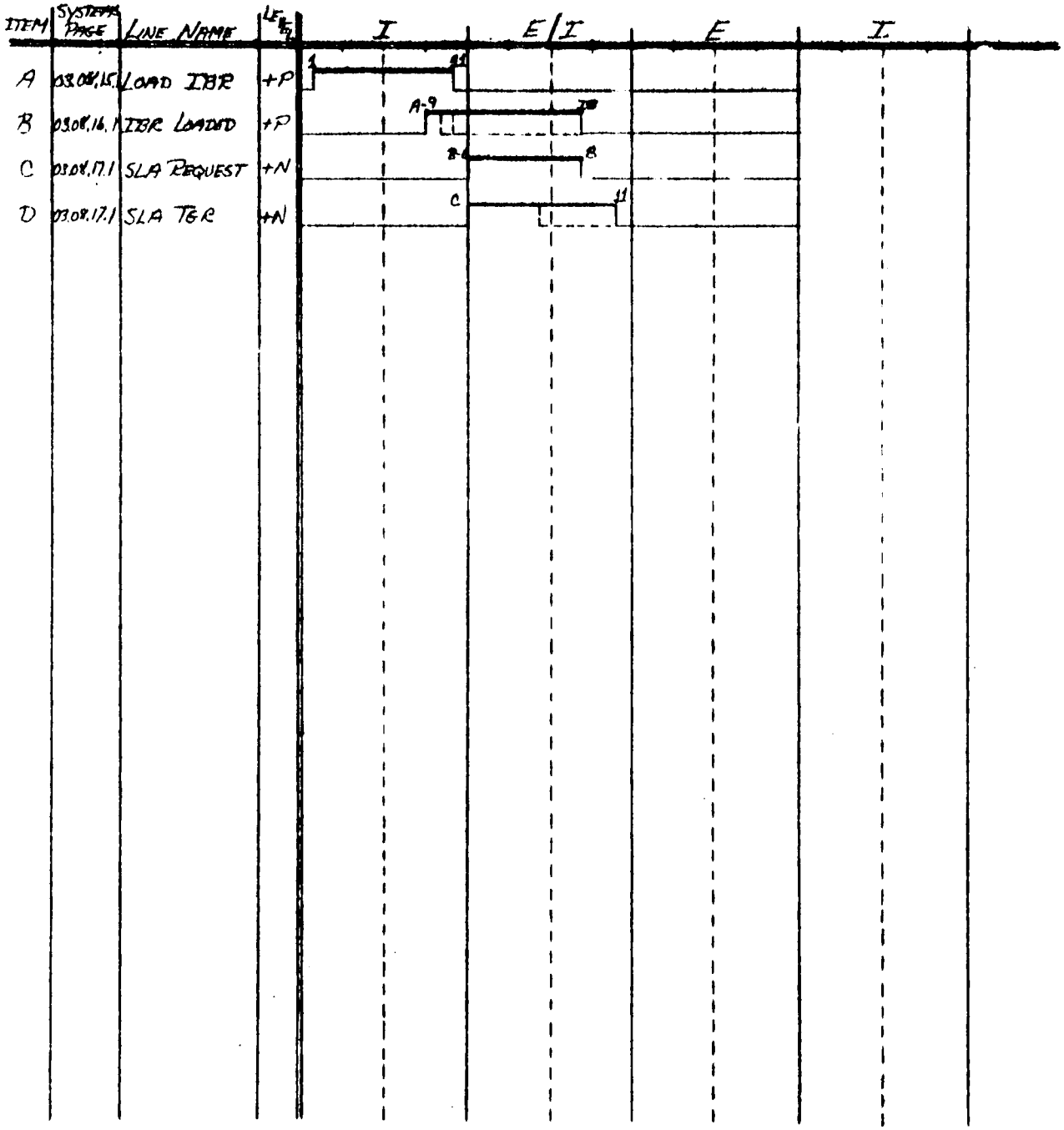
At I6 time the computer normally expects the next instruction from the storage bus. The instruction is supplied; not from the storage bus, however, but from the IBR. Positions 1 and 2 of the IBR are sensed and, if they contain a bit, routed to positions 8 and 9 of the program register. If positions 1 and 2 do not contain information, IBR positions 3-11 are routed to positions 1-9 of the program register. In either case, IBR position S is routed to position S of the program register. Setting of the program register from the storage bus is blocked (03. 04. 00. 1 - 03. 04. 06. 1) under SLA when information is being sent from the IBR.

At I7 time, positions 21-35 of the IBR are routed to the address register and positions 18-20 to the tag register. Normal I7 gating of the storage register to address register is blocked on systems page 03. 06. 10. 1. The line for gating IBR positions 18-20 to the tag register is found on systems page 03. 08. 10. 1 as "gate IBR 18-35 to AR + TR". The tag register is blocked from being set from the storage bus because of the absence of "MF SB gate" (02. 12. 50. 1) at I7 time. Even though the master I time trigger is turned ON at 6-time, the "Itime late" arrives too late to allow storage bus setting into the tag register.

Although all of the essential information has been routed directly from the IBR to the various registers, it is still necessary to set the storage register so that channel decoding can be accomplished for I-O operations. Normal storage bus to storage register gating is blocked by the SLA trigger (02. 12. 51. 1).

At I8 time the IBR loaded trigger is reset to drop "SLA request". At I11 time the SLA trigger is reset and the computer proceeds normally with the odd location instruction.

Critical conditions occur at 6-time when the transition is made from an E to I cycle. SLA trigger outputs are used in some cases to bridge the transition gap and provide good pulses (i.e., 02.11.40.1). E-time is blocked under SLA (i.e., 02.15.40.1 and 02.15.72.1) to prevent sliver conditions from possibly occurring when both the program register and E time lines are in a period of changing state.

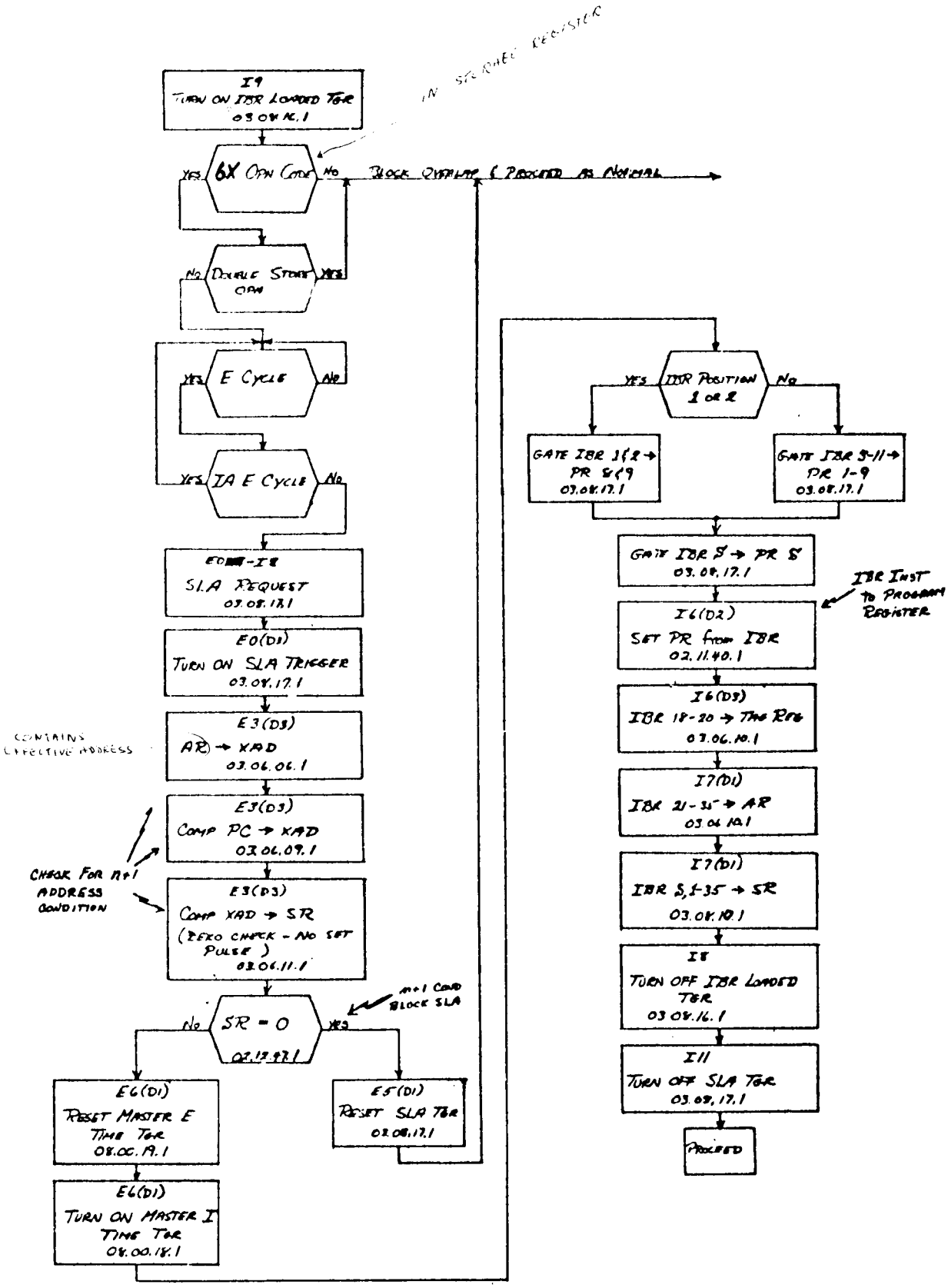


SLA SEQUENCE CHART

FIGURE 4

(SLA 5) OVLP 13

E -- I changes



SLA FLOW CHART

FIGURE _____

(SLA 6) OVLP 14

DATA OVERLAP (Figures &)

Data overlap (data lookahead) allows a one cycle reduction in the execution time of certain combinations of consecutive even-odd location instructions. The basic requirements are that: the even addressed instruction is indexable and composed of at least two cycles; and the odd addressed instruction requires a data fetch from core storage.

During the same I cycle as the even instruction is received from core storage, the instruction in the next higher odd location is set into the IBR. While the even addressed instruction is completing execution, address modification is performed on the odd address data fetch location.

Cyclic makeup of the DLA operation can take on many variations depending on the instruction being overlapped and whether or not indirect addressing is indicated. Two of the common forms are: I - E - I/E - I and I - L - I/E - I. Notice that the even instruction can take the form of either I - E or I - L while the odd instruction must always have an E (data fetch) cycle. Overlapping instructions which require several L cycles such as ARS produce a cyclic makeup of I - L - L - L - I/E - I. Whatever the makeup might be, however, at 6-time of the second I cycle a transition is made to E time. At this point, the instruction in the IBR is routed to the program register, data arriving from core storage is set into the storage register and the computer proceeds as normal. Eliminating the last half of the I cycle and first half of the following E cycle has eliminated one full cycle's worth of time and fulfilled the objective of DLA.

There are no restrictions on tagging or indirect addressing for either the even or odd addressed instruction; either or both may be tagged and indirectly addressed. Cyclic makeup for the various combinations are:

Neither instruction indirectly addressed: I - L - I/E - I

Even instruction indirectly addressed: I - E(IA) - L - I/E - I

Odd instruction indirectly addressed; I - L - I/E(IA) - E - I

Both instructions indirectly addressed: I - E(IA) - L - I/E(IA) - E - I

"DLA request" (03.08.17.1) gates functions during the overlapping E or L cycle and turns ON the DLA trigger at the beginning of the next I cycle. DLA violations prevent "DLA request" from becoming active. Some of these restrictions are that:

1. The IBR loaded trigger has been turned ON and allowed to remain ON. These were discussed in a previous section.

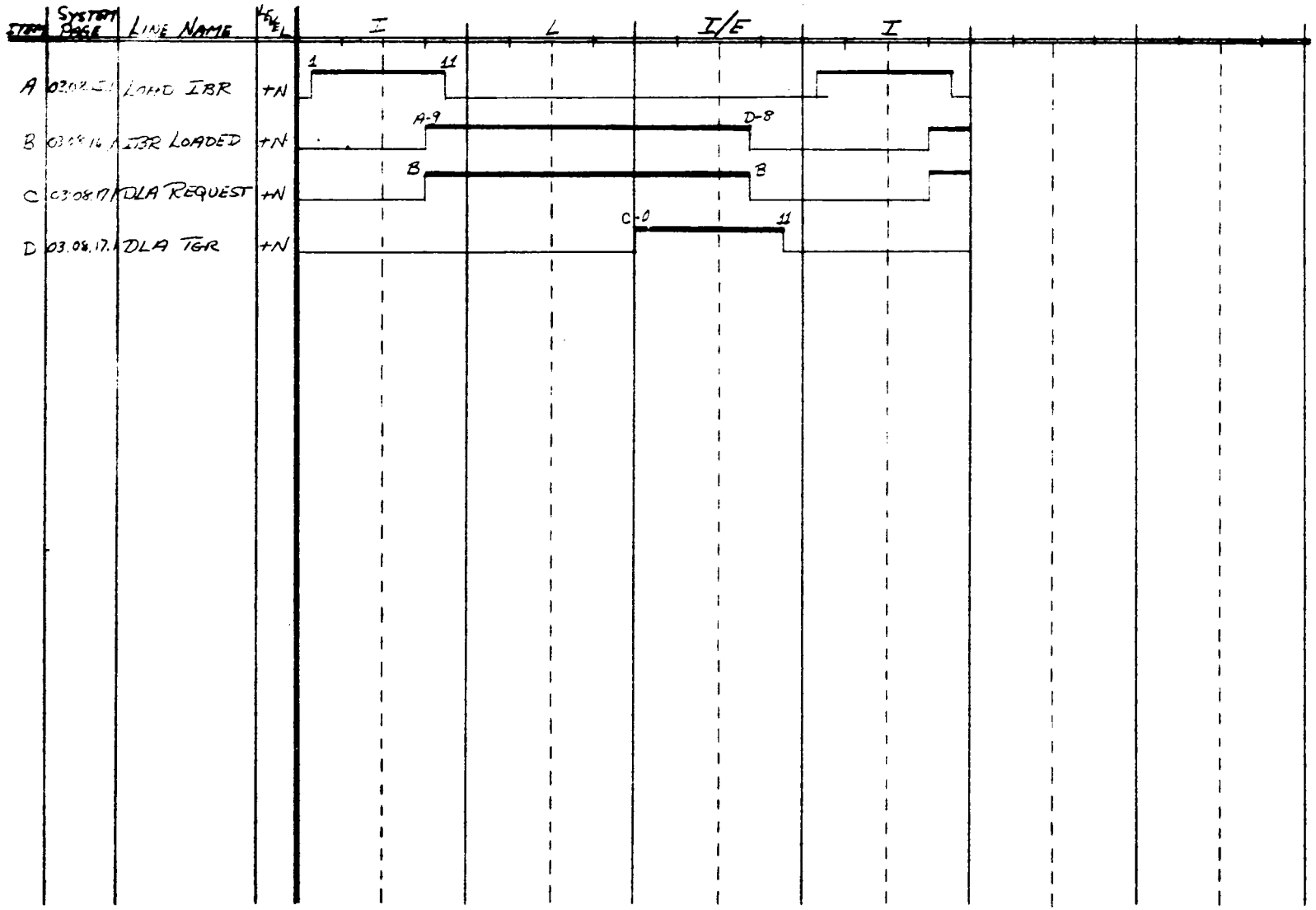
2. The even addressed instruction is not a 6X (store type) operation code. A restriction is placed at this point because the store instruction may be modifying location n+1 (the next sequential instruction). If this is the case, the instruction, which is now in the IBR, is being modified by the program and, therefore, invalid.

3. That the odd addressed instruction requires a data fetch (an E cycle reference to core storage for data). This fact is determined for the most part, by decoding 2X-3X-4X-5X from the IBR.

There are certain restrictions that fall within this group of 2X-3X-4X-5X operation codes, however. POD 54 (RCH and LCH) are 5X instructions concerned with channel operations and may be delayed for many L cycles before going to E time and actually ending operation. Execute (+0520) is a 5X instruction which only requires an I cycle and, therefore, cannot overlap. Enable (+0564) cannot overlap because of the short E cycle which would be presented to the channel. The attached channels require early E pulses for resetting purposes; these pulses would be missed because of the E cycle starting at 6-time. HPR(+0420) in the IBR blocks overlap; the transition from I to E at 6-time prevents the necessary I10 pulse needed to stop the machine.

Any other instruction which requires the last half I cycle or early half E cycle (which were eliminated by the I/E transition) also blocks DLA overlap from continuing.

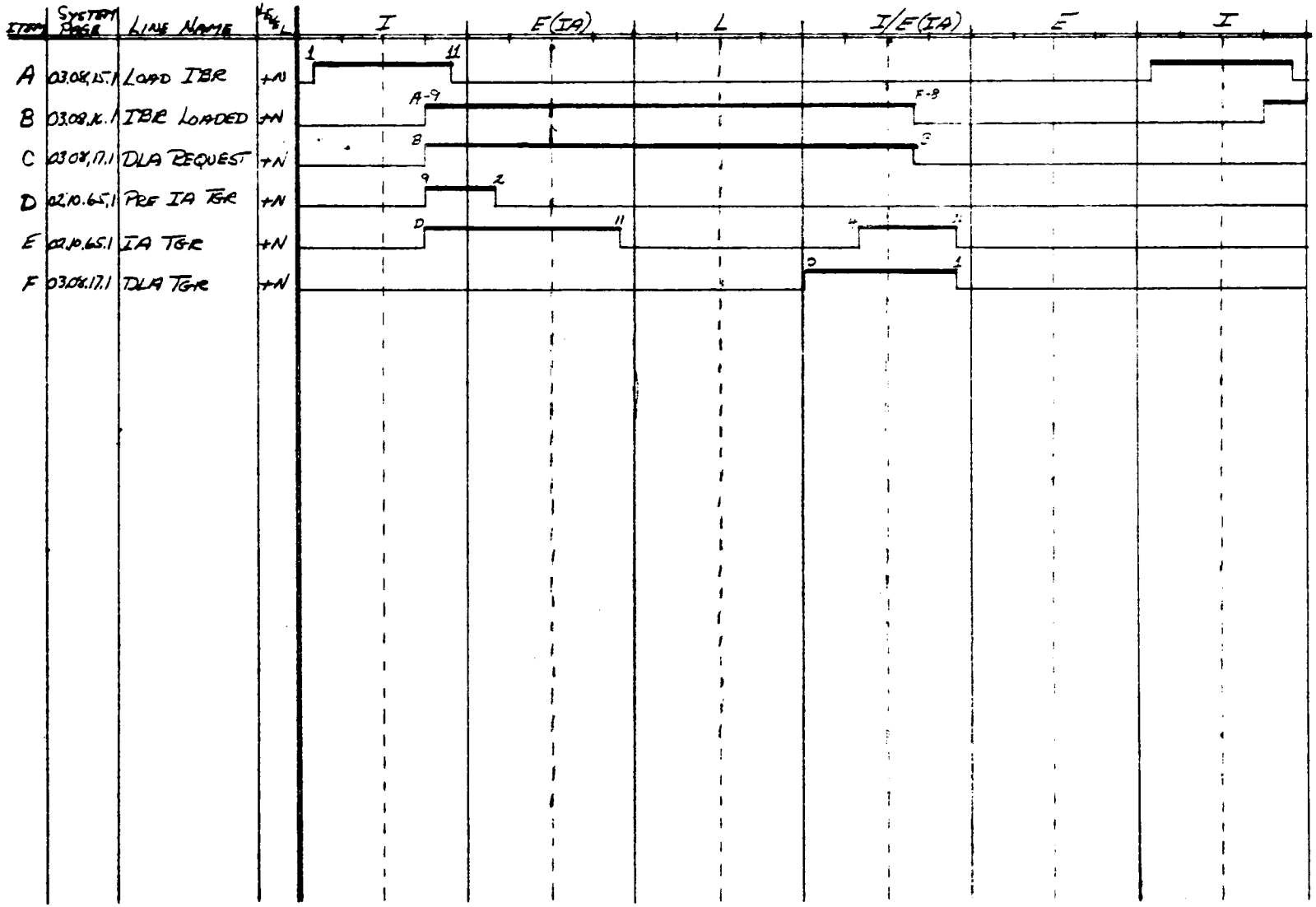
4. The suppress DLA/SLA switch on the 7151 console is in the OFF position. This switch is used as a service aid by the customer engineer when either trouble shooting or servicing the system. With the switch in the "ON" position, all DLA and SLA overlapping is suppressed.



DLA SEQUENCE CHART

FIGURE 6

DA 9/82



DLA SEQUENCE CHART (WITH INDIRECT ADDRESSING)

FIGURE 7

(DLA 0-5;) → AVL

80 9/62

Data Overlap Without POD 34 (Figures , ,)

POD 34 (CAS and LAS) instructions tend to complicate the picture of data overlap. For simplicity, consider an operation without POD 34 restrictions, or indirect addressing in the odd instruction.

Address modification for the odd addressed instruction is accomplished during the E or L cycle in which the even addressed instruction is being executed. As explained earlier, the even addressed instruction can be composed of either E or L cycles; overlap operations are delayed one cycle in the case of indirect addressing. By E/L 2 time, the index registers have served their purpose for the even addressed instruction and the tag register is replaced by positions 18-20 of the IBR. At the same time, IBR positions 21-35 are routed to and replace the contents of the address register.

The even instruction at this time is either awaiting data from core storage (E cycle) or in the process of performing logical functions (L cycle); in either case, the index adders are not being used by the even instruction. It is during this 0-6 time that address modification is accomplished for the odd instruction by routing the address register, index register and a carry to the index adders and back again to the address register. The address register now contains the odd instruction data fetch reference to core storage. This address is sent to MAR at 10 time of the cycle in which the even instruction ends operation. Notice that if the even instruction requires several cycles before ending operation, (for example an ARS with a shift of 35) the complete process of routing the IBR to the tag register and address register and performing the address modification is repeated each cycle. In this way, the correct address is available to be sent to MAR when required.

If an I-O trap, interrupt or channel trap demand occurs during the even instruction, DLA is cancelled, the computer is forced into an STR operation and proceeds as it would under normal conditions. The DLA is suppressed by holding the DLA trigger reset and not allowing it to be turned ON at the normal I0 time. The previous address modification which was accomplished under "DLA request" is ignored.

At 8-time of the initial I cycle the program counter was stepped +1. During the early portion of the second I cycle, the program counter value is incremented through the index adders and set into the address register. The address register value is that of the even instruction +2 (the next sequential instruction to be executed following the DLA). The program counter at this time contains the location of the even instruction +1 (the location of the instruction now in the IBR).

During I4 (D2) time the address register value is incremented through the index adders and set into the program counter provided that the even instruction was not a FP instruction signalling a trap. If a FP trap does occur at this point, the program counter still contains the correct address for storing in location 00000_8. If a trap does not occur, the program counter at this point contains the location of the even instruction +3 which is one location higher than the next sequential instruction to be executed following the DLA.

At I6 time a transition is made from I time to E time by resetting the master I time trigger and turning ON the master E time trigger. Positions 1 and 2 of the IBR are tested to determine if they or positions 3-11 should be routed to the program register at 6 time. At the same time that the odd instruction is set into the program register, its data is arriving from core storage and set into the storage register.

Remember that at transition time, the address register contains the location of the next sequential instruction to be executed, and the program counter is one location higher.

At E9 (D2) time the address register is routed to the index adders; if the odd instruction is not a ZET/NZT requiring a skip of +1, the index adders are set into the program counter to effectively reduce its value by one. If the skip is called for, the program counter is unaltered. In either case the program counter is routed to MAR at E10 time. The DLA trigger is turned OFF at 11 time and the odd instruction completes its operation as it would under normal conditions.

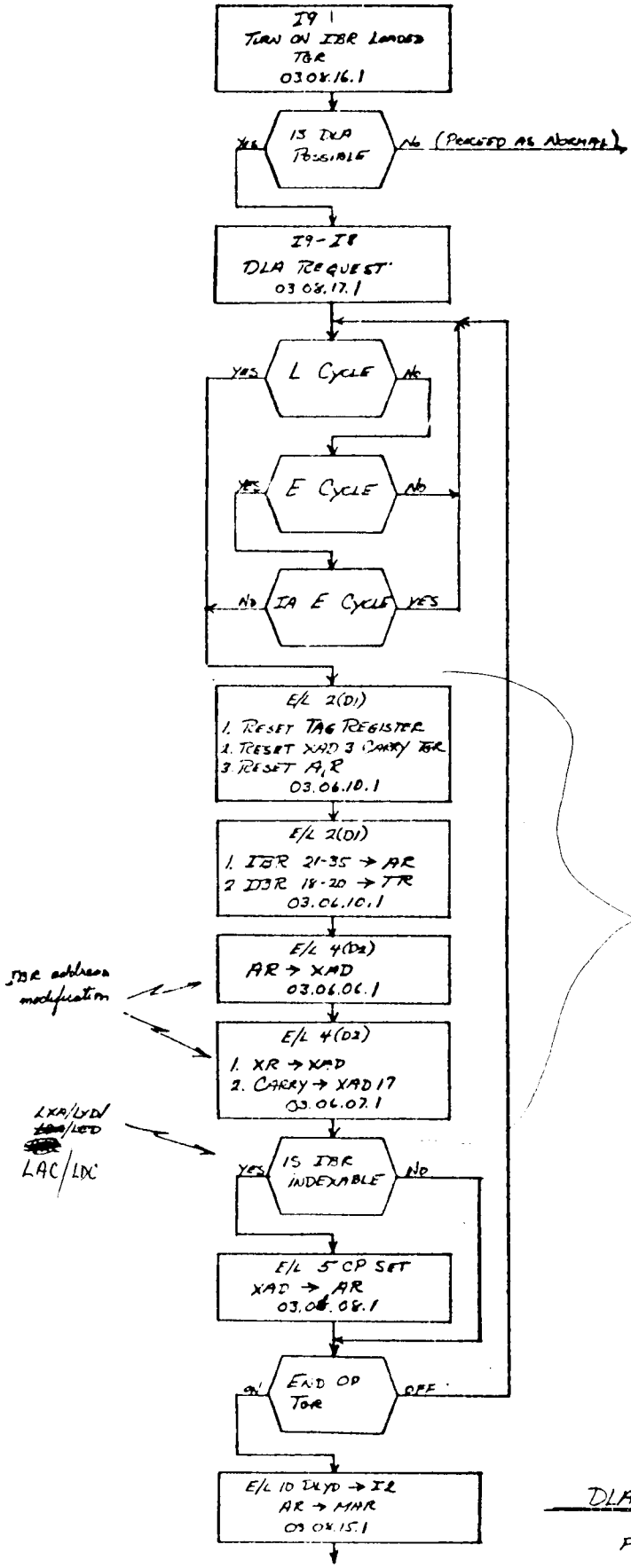
Going back over the operation, take the case of indirect addressing in the odd instruction. Indirect addressing means that the address register cannot be used in conjunction with the program counter to keep track of the program location. This is because the address register is required to perform address modification of the address coming in from storage during the IA cycle.

During the early portion of the split I/E cycle the program counter is not incremented to the address register but transferred directly without a carry to XAD 17. During I4 (D2) time the address register is incremented to the program counter and the program counter now contains the correct address for the next sequential instruction following the DLA.

At the transition point of the I/E cycle, the indirect address core storage location arrives on the storage bus and is set into the storage register and tag register at E6 (D2) and E7 (D1) times respectively. Positions 21-35 of the storage register are immediately routed to the address register and address modification is performed on this address during the period E9 (D2). This address which is the actual data address, is gated to MAR at 10 time and the computer proceeds to another E cycle. At 11-time of the IA cycle both the IA and DLA triggers are turned OFF.

At 6-time of the following E cycle, the data arrives from core storage and is acted upon by the odd instruction. The program counter contains the correct address for the

next sequential instruction following the DLA. However, during E8 (D3) time, the program counter and a carry are gated to the index adders. If the odd instruction is a ZET or NZT with the skip condition met, the index adders are returned to the program counter in time to send the skip address to MAR at 10-time of the cycle.

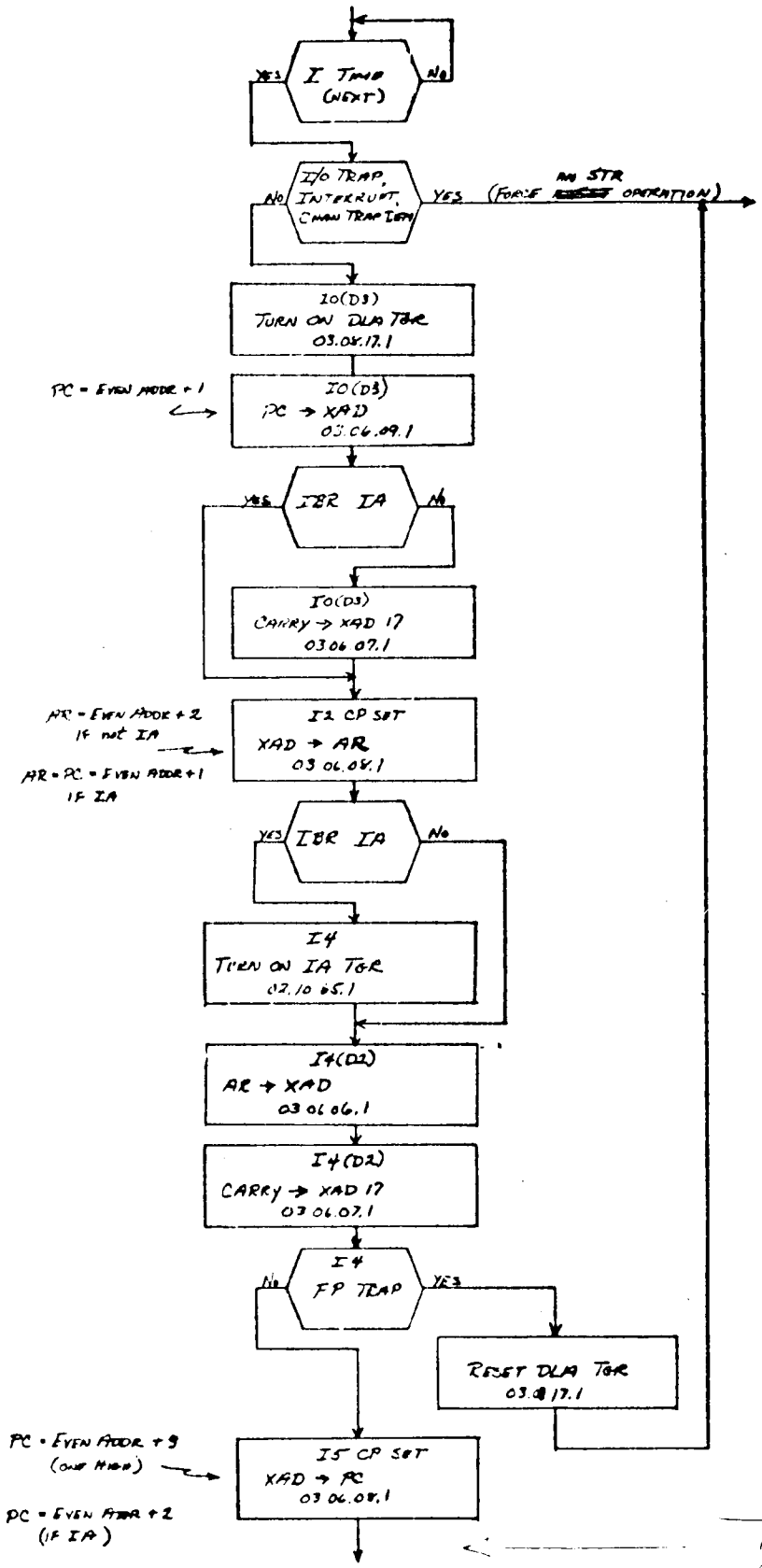


"CAS"

DLA (WITHOUT POD 34)

FIGURE _____

(DLA 10) OVLP 24



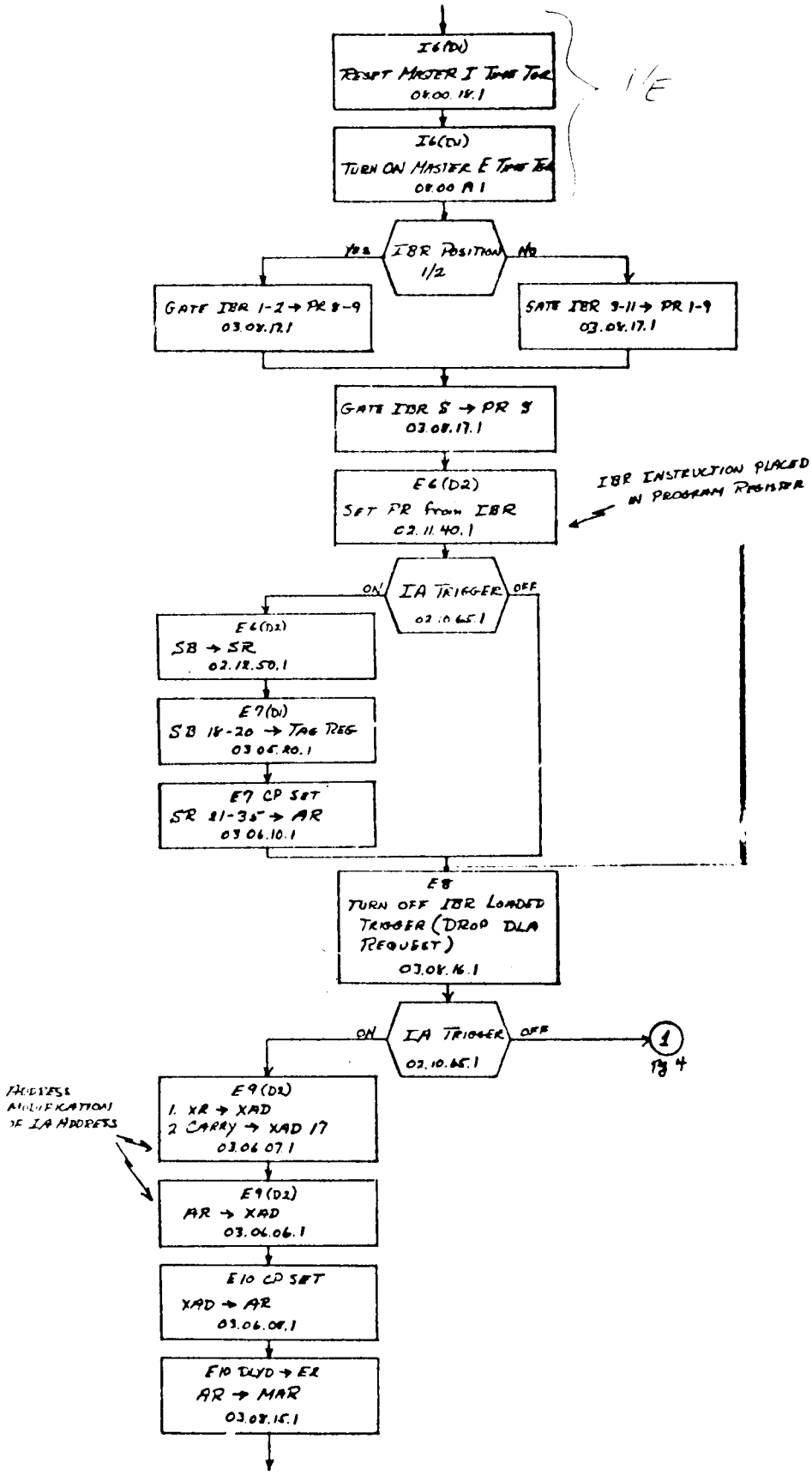
PC - 1
AR - 2 12
PC = 3 15

DLA (WITHOUT POD 34)

FIGURE _____

Pg 2 of 9

(DLA11)
OVLP 25



DLA (WITHOUT POD 34)

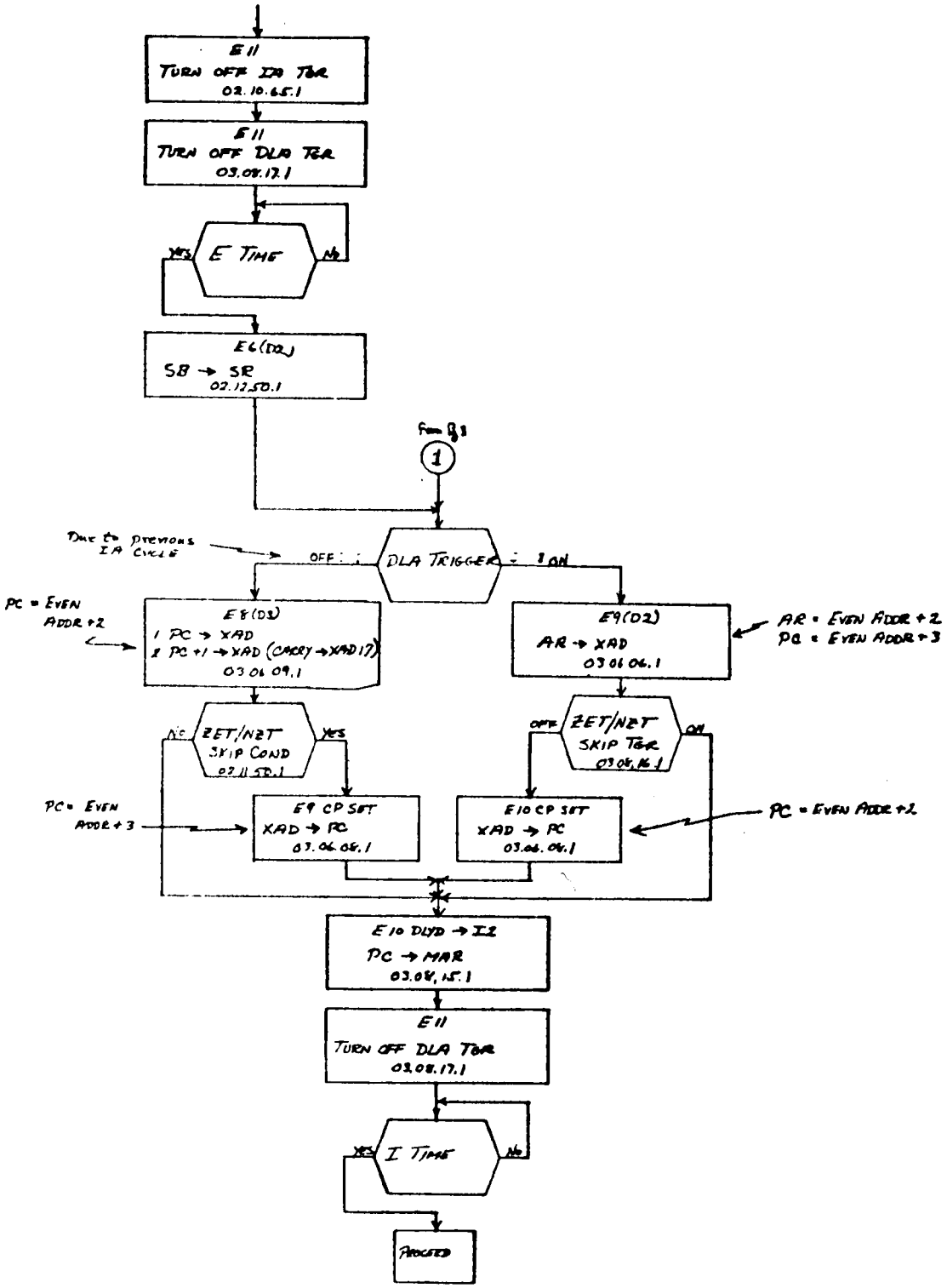
FIGURE

Pg 3 of 4

Pg 9/12

(DLA 12)

OVLP 26



DLA (WITHOUT PMD 34)

FIGURE _____

(DLA 13)
OVLP 27

Data Overlap With POD 34 (Figure)

Data overlap with POD 34 instructions is similar in most respects to the previous explanation. The main area which complicates the operation is that of anticipating the various skip conditions for POD 34's in either or both of the instructions, and maintaining a correspondingly correct address in either the program counter, address register or index adders.

Considering that there is a POD 34 instruction in the even address, the program counter is incremented during the first E6 (D2) time to provide a skip 1 address (even address +2) if needed. In anticipating a skip of 2, the program counter is immediately rerouted to the index adders with a carry to XAD 17 during E8 (D3) time. If, at E10 time, the comparison calls for a skip of 2, the index adders are returned to the program counter in time to be sent to MAR with an E10 delayed gate. If a skip 1 is called for, the skip 2 address dies in the adders and the program counter value of even address +2 is routed to MAR. If a no-skip condition exists, the modified address of the odd instruction is routed from the address register to MAR at E10 delayed time. In this case, as with the other POD 34 conditions, the program counter maintains the skip addresses while the address register provides the no-skip (next sequential instruction) address for MAR.

If either a skip 1 or skip 2 condition exists, the next sequential instruction (which is now in the IBR) is to be bypassed. This situation cancels overlapping by holding the DLA trigger reset and not allowing it to be turned ON during the next I0 (D3) time. Trapping from external devices is blocked from occurring during POD 34 overlapping because of the incorrect value that would be present in the program counter.

If the even addressed POD 34 allows DLA to continue because of a no-skip condition, the computer proceeds to the split I/E cycle. During I0 (D3) time of this split cycle, the

program counter is routed through the index adders and set into the address register. There is no incrementing performed during this transfer from the program counter so the address register contains the value: even address +2. Notice that this address represents the next instruction to be executed following the DLA and also corresponds to the no-skip address of an odd addressed POD 34 instruction.

Assume that a POD 34 instruction is also in the odd address. With the no-skip address in the address register, the skip 1 address is set into the program counter by incrementing the address register through the index adders during I4 (D2) time. Notice, here, that there is a restriction that the odd POD 34 is not indirectly addressed. If indirect addressing is indicated, the no-skip address in the address registers will be destroyed at E7 time of the IA cycle when the indirect address is received from core storage and modified through the index adders. The program counter not being modified at this time, allows the no-skip address (which is now in the program counter) to be maintained and saved for later use.

At 6-time of this split cycle, a transition is made from I to E time and, if indirect addressing is called for, the E cycle becomes an IA E cycle with DLA temporarily suspended. During this IA cycle, the information from core storage is set into the storage register. SR positions 21-35 are immediately routed to the address register and address modification is performed through the index adders and available to MAR at the next E10 time.

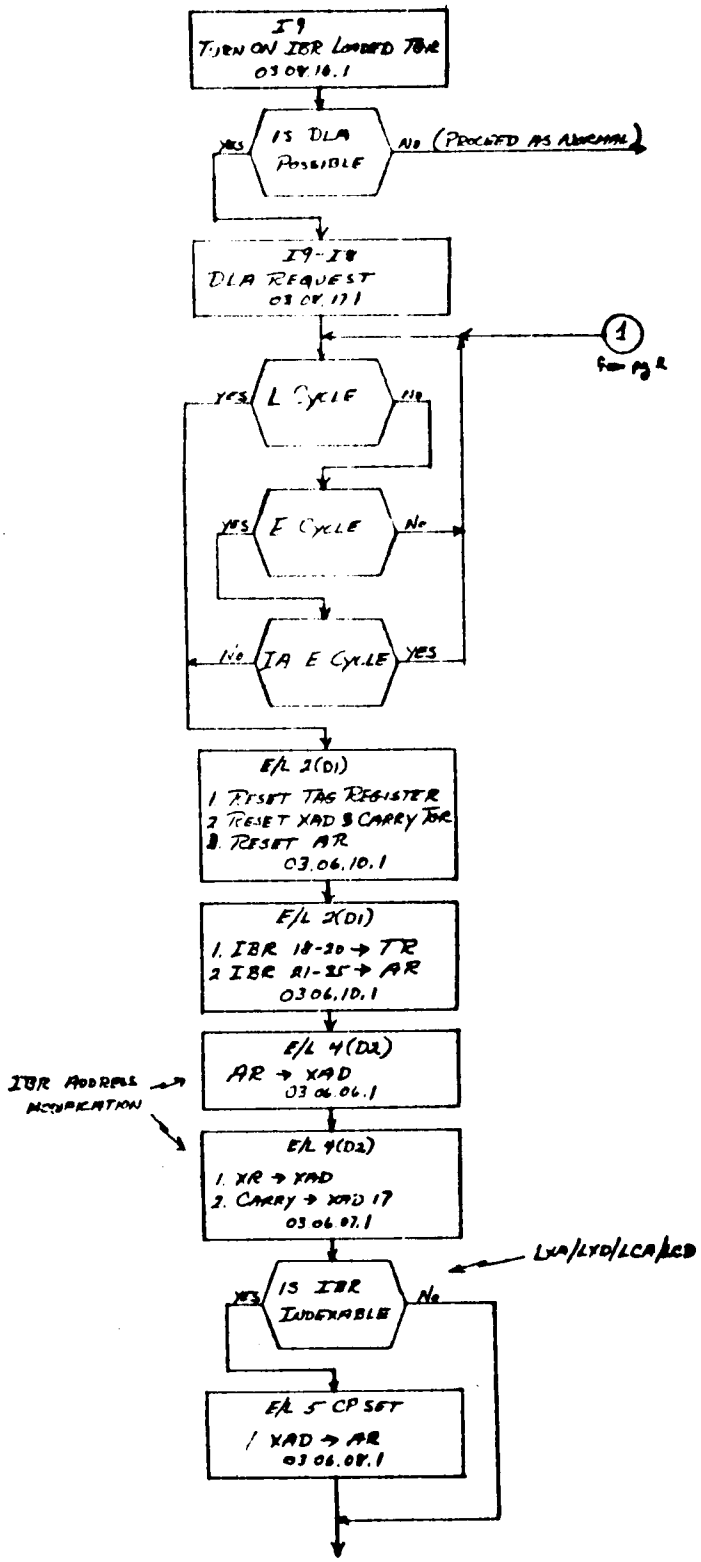
The IA cycle, requiring use of the address register, has prevented the no-skip address from being stored in the address register up to this point. The program counter, because of the IA condition, still contains the no-skip address. During E4 (D2) time this no-skip address is transferred to the address register and during E6 (D2) time the program

(DLA is
OVLP = 1

counter is incremented through the index adders to provide a skip 1 address if needed. At this point the program counter and address register contain the same corresponding values as would have been produced if indirect addressing was not called for in the first place.

A skip 2 address is provided for the odd POD 34 by routing the program counter and a carry to the index adders during E8 (D3) time.

The program counter, as mentioned before, provides either the skip 1 or skip 2 address while the address register contains the no-skip address. Therefore, if a skip condition is indicated at E10 time, the program counter is routed to MAR; if no skip condition exists, the address register is routed to MAR. With either a skip 1 or skip 2 condition, the program counter is correctly set for continuing. With a no-skip condition, however, the address register contains the proper address; therefore, during the following I2 (D2) time the address register is routed to the program counter and the operation is completed.



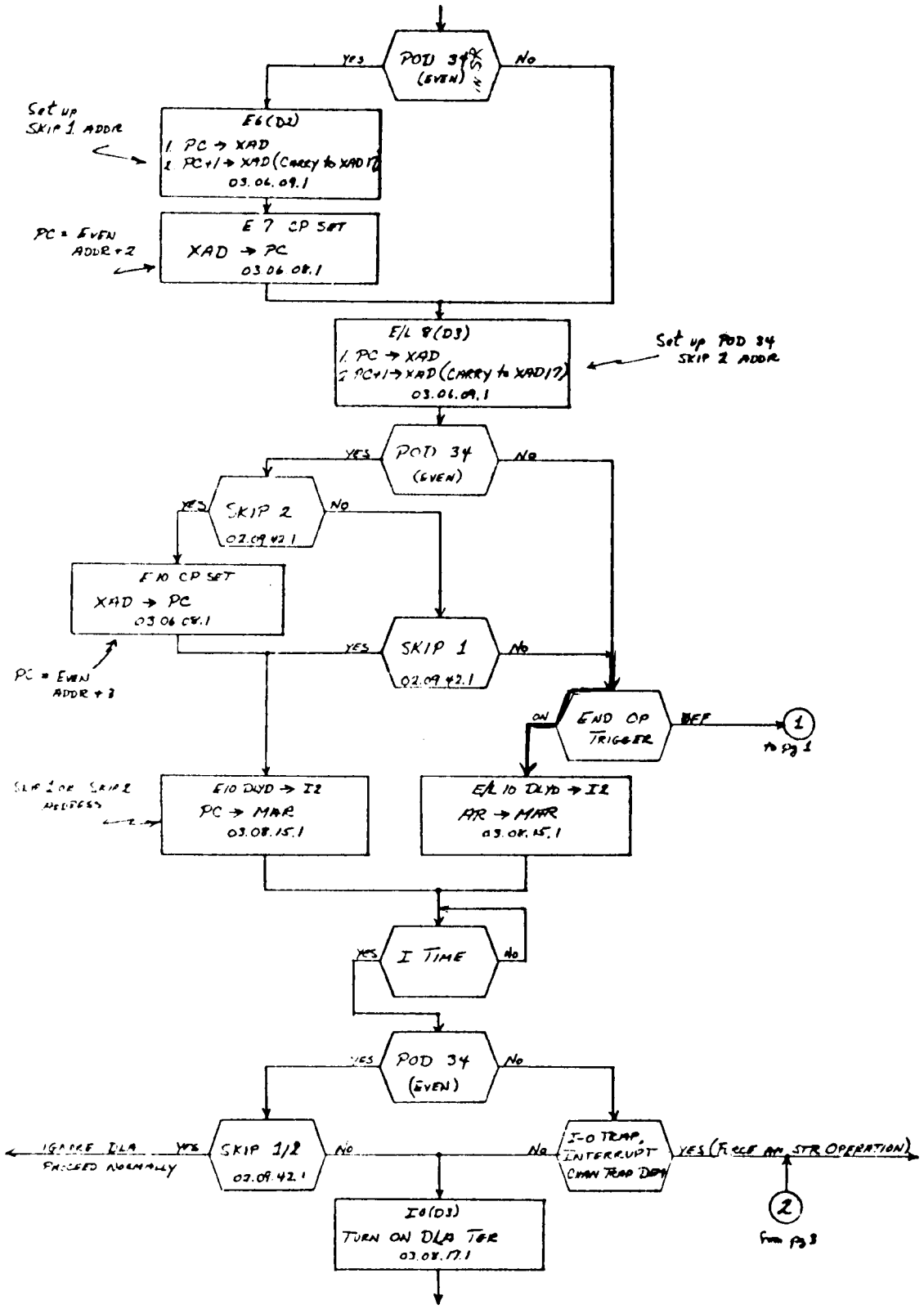
CHS

DLA (WITH PCD 34)

FIGURE _____

PG 1 of 1

(DLA17)
OVLP 31

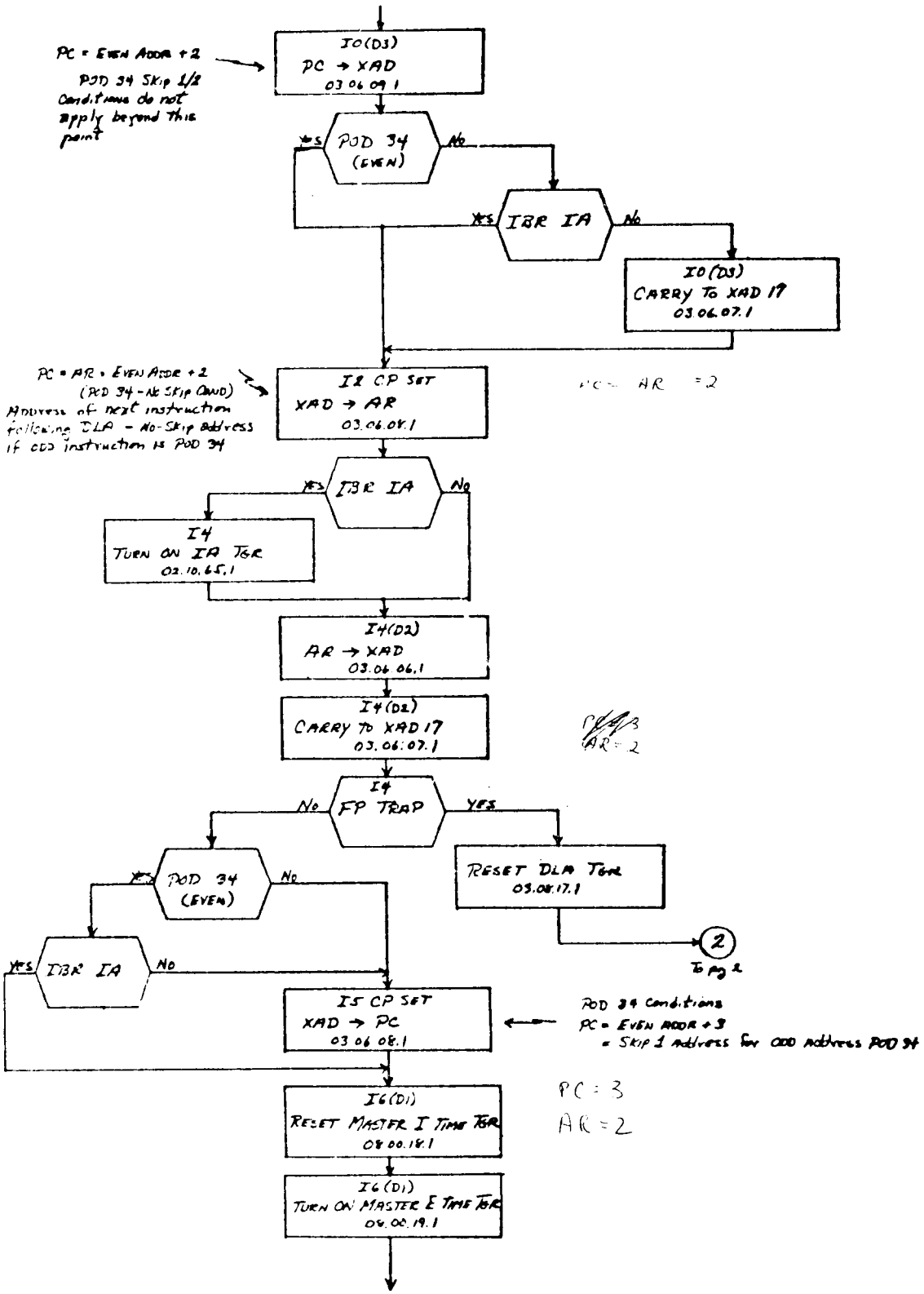


DLA (WITH POD 34)

Pg 2 of 6

FIGURE

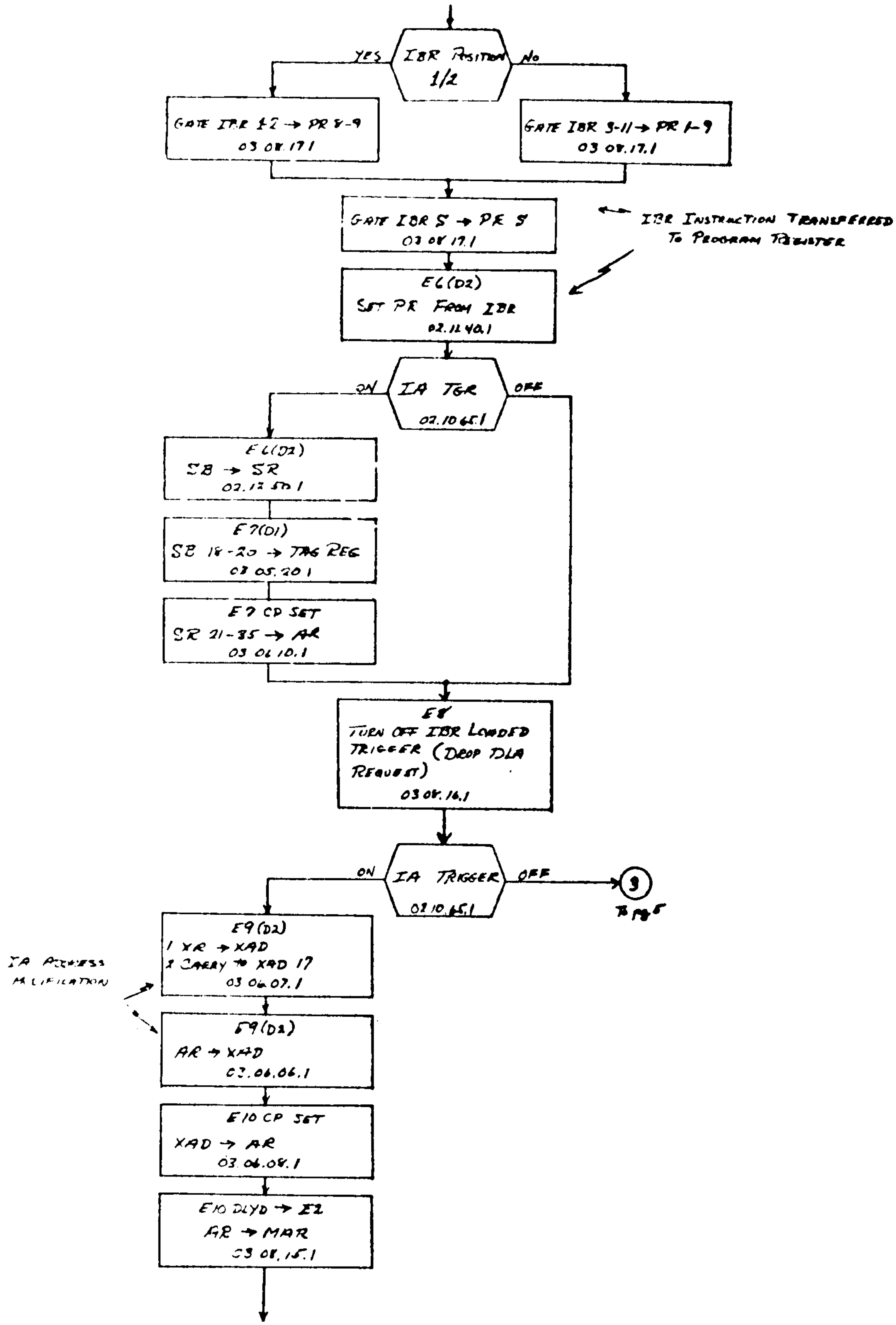
(DLA 18)
OVLP 32



DLA (WITH POD 34)

FIGURE _____

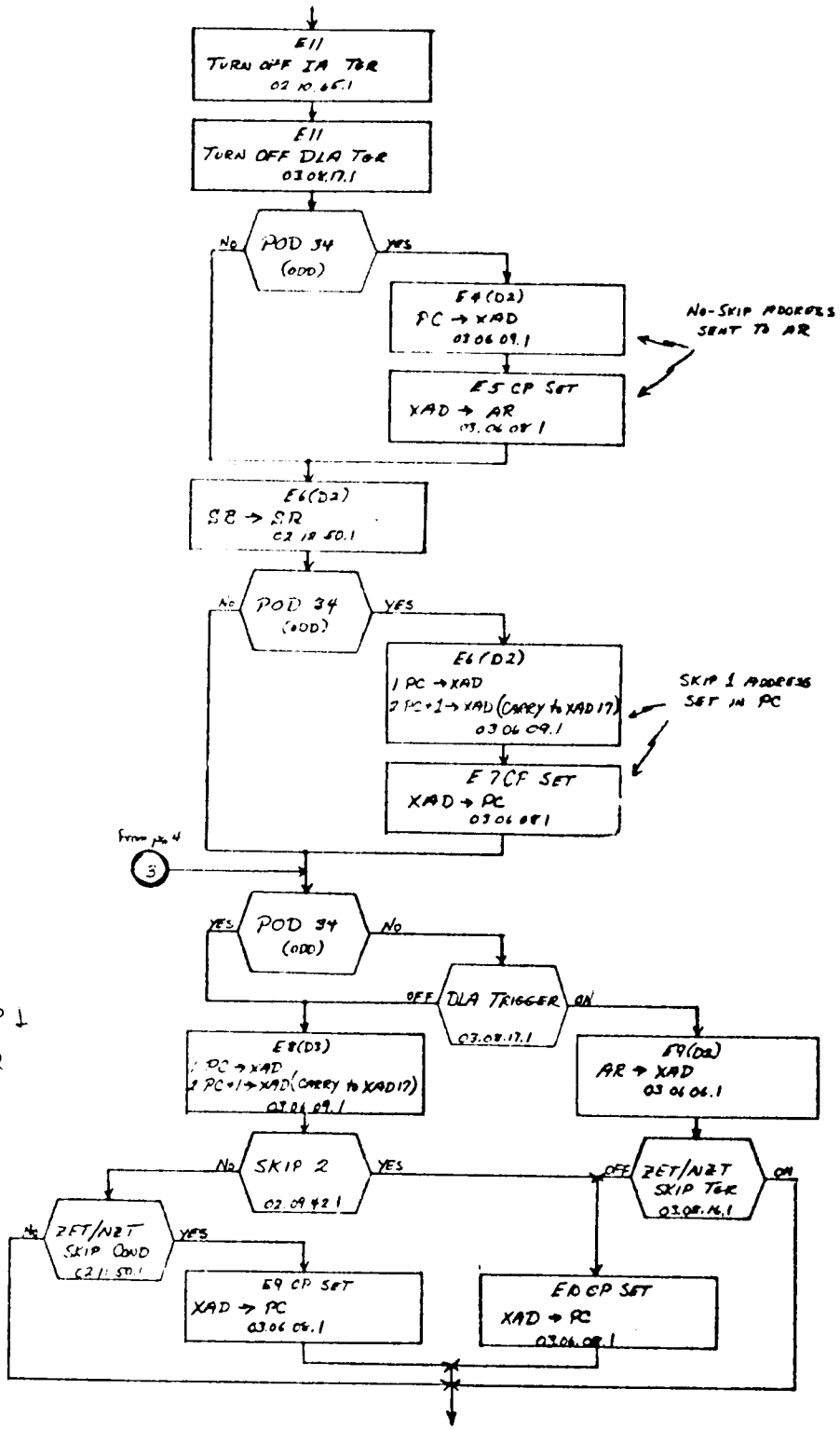
(DLA 19)
OVLP 33



DLA (WITH TRD 34)

FIGURE ________

9.45-6   20 9/62



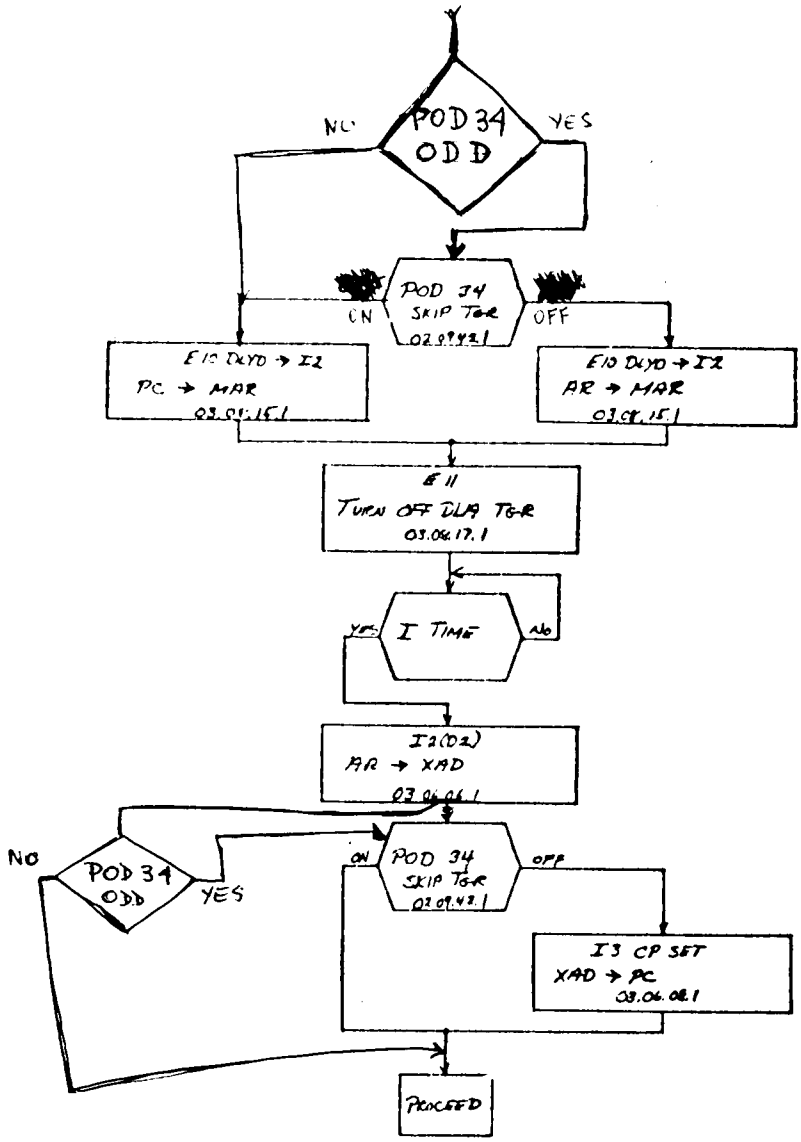
PC = SKIP 1
XAD = SKIP 2

AD 4
PC = 3
AR = 2

DLA (WITH POD 34)
FIGURE _____

pg 5 of 6

2 of 2
(DLA 21)
LP 35



DLA (WITH POD 34)

FIGURE _____

Pg 6/6

(DLA 22)
CVLP 36

TRANSFER OVERLAP (FIGURE)

Transfer overlap (transfer lookahead) allows certain transfer instructions in odd core storage locations to be executed simultaneously with the instruction in the next lower even core location. The transfer is effected completely in overlap and requires no machine cycle time of its own. Cycle time is composed of I - E - I (next); the E cycle is common to the two instructions and is where the overlap occurs.

The allowable transfer overlap instructions are TRA, TTR, TIX, TNX, TXH, TXL, TXI, and TSX. These are one cycle instructions; unconditional transfers or transfers dependent on index register contents.

Restrictions on TLA are:

1. The IBR is allowed to be loaded as explained previously.

2. The IBR LDD trigger is allowed to be turned on and not reset as explained previously.

3. The "suppress TLA" switch is OFF. This is a switch on the 7151 console for customer engineer's use and allows the TLA feature to be disabled during trouble shooting procedures.

4. The computer is not in "trap mode". When in trap mode all successful transfers are caused to trap to a specific location instead of actually transferring.

5. The TRA or TTR is not indirectly addressed. If this is the case, the transfer requires two cycles (I-E) and therefore cannot be completed in the one cycle available for transfer overlapping.

6. The computer is not in multiple tag mode trying to execute a multiple-tagged TIX, TNX or TXI instruction. These instructions cause the specified index registers

to be modified as a result of their execution; multi-tagging causes the index registers to become OR'ed. This OR'ing is completed during the early portion of the overlapped E cycle before the even addressed instruction is completely executed. If this even instruction, for example, is a floating point instruction causing a trap situation, the overlapping must be stopped and the transfer ignored. At this point, however, it is impossible to retrieve and restore the original index register values.

7. The instruction in the even address is not a one cycle instruction. This situation is covered for the most part by decoding 2X, 3X, 4X, or 5X from the program register. ZET, NZT (POD 52 instructions) and ONT, OFT (POD 44 instructions), however, are blocked because of the skip conditions involved with the instruction. CAS and LAS (POD 34) are also skip type instructions but because of their extensive use, additional circuitry has been added to accommodate overlap operations.

LXA, LXD, LAC, LDC (POD 52 instructions) are also not allowed to be overlapped because they make use of the address register and would destroy the overlap transfer address being set up by the overlapped instruction.

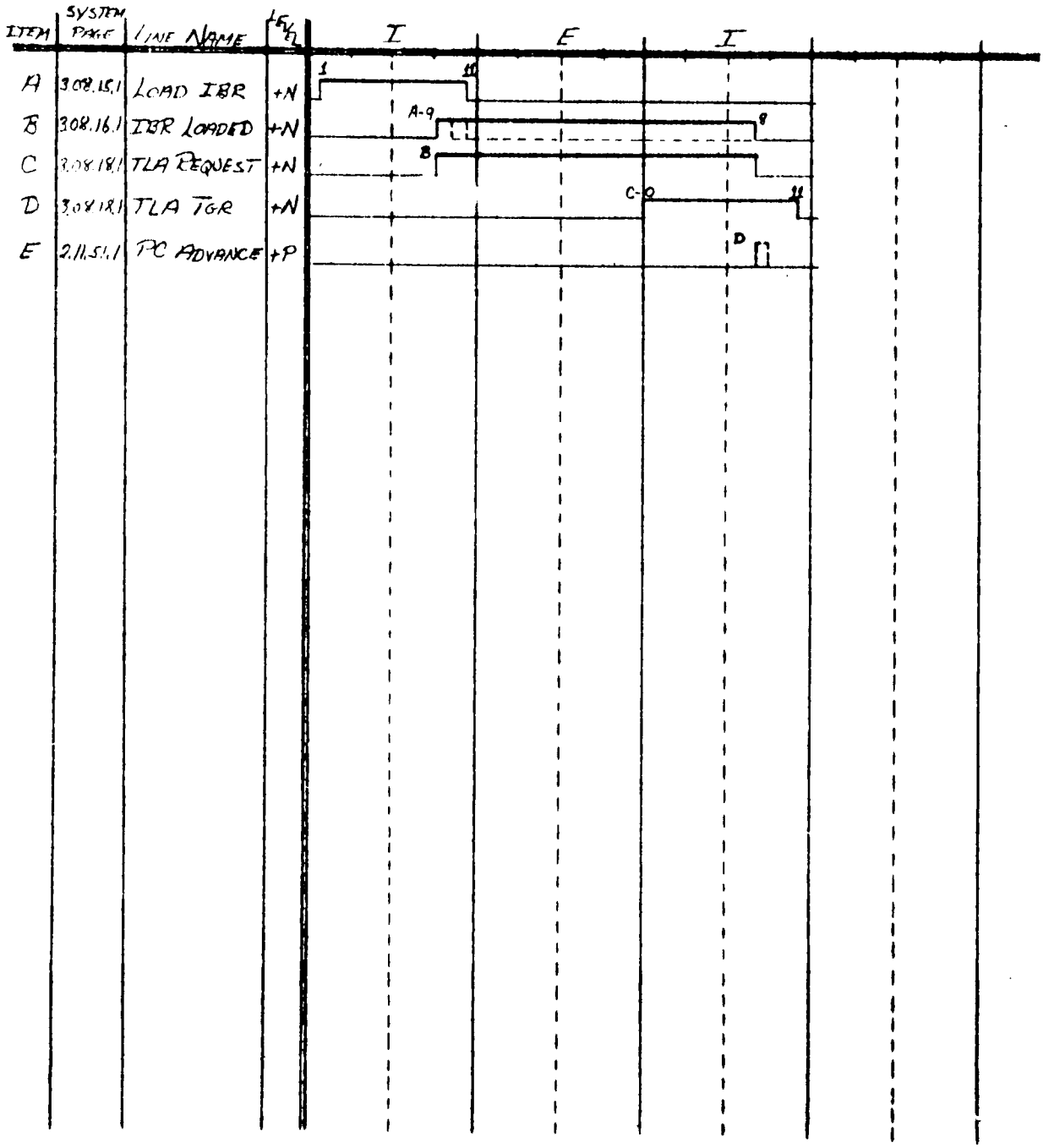
8. The instruction in the even address is not a store type instruction. This restriction is necessary because of the possibility that the store instruction will store in the address represented by the instruction in the IBR. (This is the case of an instruction in location n storing in location n+1.) These store instructions are decoded out by the lack of 6X in the 2X, 3X, 4X, 5X gating. ANS (+0320) is an instruction which involves storing data and is not a 6X operation code. It, therefore, is a special case and blocks transfer overlap by an instruction in the next higher odd location.

Many transfer instructions are prevented from overlapping. Two cycle transfer instructions (TIF, TIO, or TLQ for example) are not allowed because they could not be completed during the one cycle period of overlap. Certain one cycle instructions (TMI, TNO, TOV, or TPL for example) cannot be overlapped because the transfer condition on which the transfer depends has not been fully determined as yet.

DVH, FDH, and VDH instructions also block overlap under TLA conditions. This is necessary because at the time the halt is recognized, the program counter is already advanced beyond the correct value required at the stopping point.

Transfer overlap takes advantage of the time period from E0-E6 when the instruction in the even address is waiting for information to arrive from core storage. It is this "dead time" that the transfer address is routed and set into the address register and made ready to be sent to MAR at E10 time. Two major gating lines control the operation; "TLA request" determines if the transfer overlap is possible and gates circuitry during the overlapped E cycle; "TLA trigger" turns on at the beginning of the last I cycle and gates any circuitry necessary from that point through I11 time.

Indirect addressing is allowable for the instruction in the even address. If the even address instruction is indirectly addressed, TLA E cycle gating is for the normal E cycle and not the IA E cycle. Instructions in both the even or odd address may be tagged.



TLA SEQUENCE CHART

FIGURE 10

(TLA4, OVLP 40

B 6K2

IBR TRA or TTR (FIGURE)

At I9 time of the even address instruction, the IBR is loaded with the overlap transfer Inst. and the IBR loaded trigger turned ON. Considering that there are no conditions warranting its being turned OFF within the next two clock pulses, the IBR loaded trigger remains ON until the next I8 time. If no TLA restrictions are violated, (03.08.18.1), "TLA request" becomes active under control of the IBR loaded trigger.

"TLA request" provides gating for circuit functions during the overlapped E cycle. If the even address instruction is indirectly addressed, normal E time is blocked (02.15.40.1) during the IA cycle. Overlapping is, therefore, delayed for one cycle and begins again when the indirect addressing portion has been completed.

TRA and TTR instructions are unconditional transfers; the logic, then, is to perform any address modification necessary and set the effective transfer address into the address register so that it can be sent to MAR at the next E10 time. This address is routed, modified and set into the AR from E0-E6 time when the even address instruction is awaiting information from its core storage reference. At E2 time IBR positions 18-20 are gated to the tag register while positions 21-35 are gated to the address register. During E4 (D2) time the index register value is subtracted from the transfer address and the modified address set back into the address register. Under normal conditions, the transfer overlapping is completed.

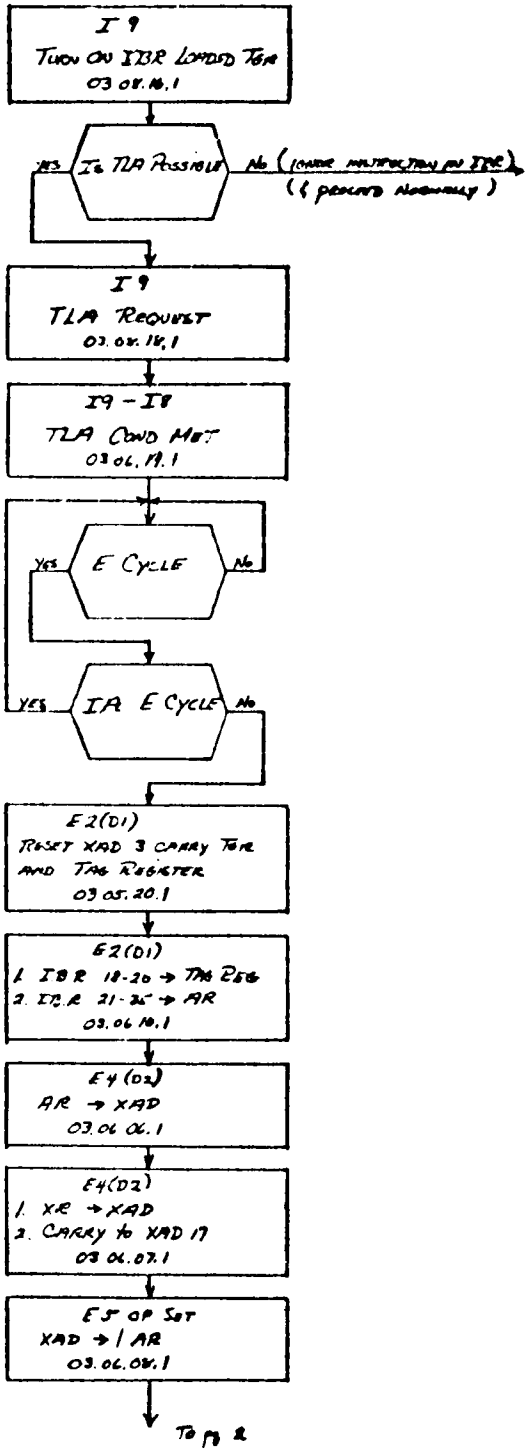
If the transfer is being overlapped on a CAS or LAS (POD 34) instruction, a skip may have to nullify the effect of the transfer. If the POD 34 does not require either a skip 1 or 2, the overlapping transfer (next instruction in sequence) is executed and the address register (transfer-to address) is sent to MAR. If a skip 1 or 2 is required, however, the address register is

not sent to MAR; instead, POD 34 ~~CAS~~ modified addresses from the program counter determine the address of the next instruction. These conditions are shown in the following chart.

Even Address Instruction \ Odd Address Instruction (TRA/TTR)	Transfer Conditions Met
Not POD 34	AR → MAR
POD 34 (No Skip)	AR → MAR
POD 34 (Skip 1)	PC → MAR
POD 34 (Skip 2)	PC → MAR

During the next I cycle, the program counter is updated to indicate the location of the next instruction plus 1. If the TRA is not overlapping a POD 34 ~~CAS~~ or if it is overlapping a POD 34 ~~CAS~~ with a no-skip condition, the address register is incremented through the index adders and set into the program counter. If the TRA is overlapping a POD 34 ~~CAS~~ which requires a skip 1 or 2, the program counter is incremented through the index adders. In either case, the program counter is one step higher than its normal value; therefore, the normal program counter advance is blocked by the TLA trigger during the following I8 time.

A POD 34 ~~CAS~~ in the even address blocks any possible interrupts from occurring during the transfer overlap; an interrupt occurring at this time will be serviced following the next instruction. If the even addressed instruction is not a POD 34, trapping may occur due to and I-O trap, interrupt or channel trap demand.

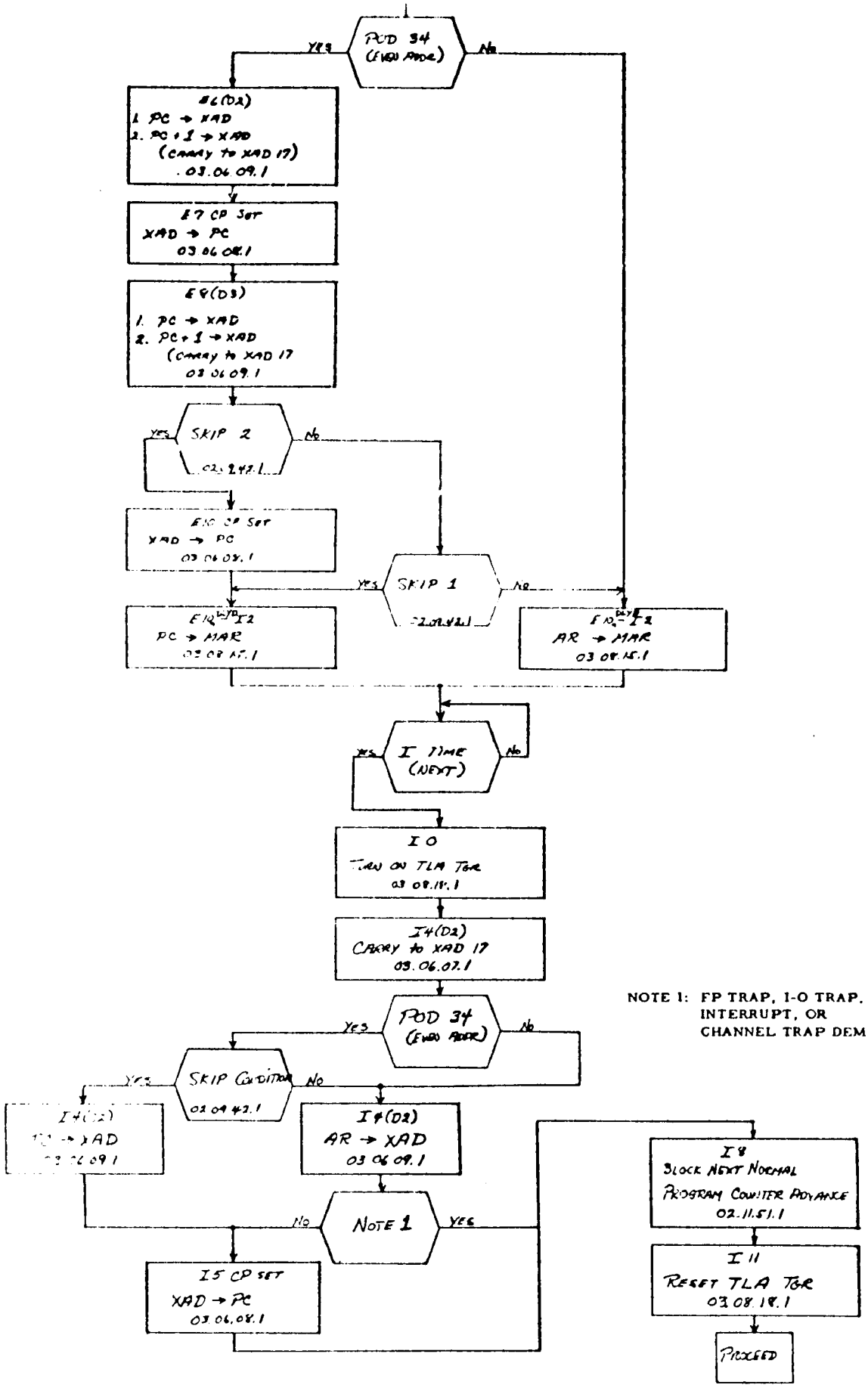


IBR TRA/TTR FLOW CHART

Pg 1 of 2

FIGURE _____

(EXTRA 3)
CVLP 43



NOTE 1: FP TRAP, I-O TRAP, INTERRUPT, OR CHANNEL TRAP DEMAND

IBR TRA/TTR FLOW CHART

FIGURE _____

Pg 2 of 2
Bo 5/12

(EXTRA 4)
OVLP44

IBR, TIX, TNX, TXH, TXL (Figure)

The basic logic that existed for normal execution of these instructions still exists during overlap conditions; the main differences are the times during which the routing and testing is done, where the information is decoded or routed from, and a possible reconstruction of the index modification in the cases where the decision is made to skip around or bypass the instruction in the odd location.

At I9 time of the even address instruction, the TIX, TNX, TXH or TXL instruction is loaded into the instruction backup register and the IBR loaded trigger turned ON. Consider, again, that there are no overlap violations and that the IBR loaded trigger is allowed to remain ON.

If no TLA restrictions are violated (03.08.18.1), "TLA request" becomes active to provide gating for the circuit functions during the overlapped E cycle. One restriction directly concerned with this particular type of overlap is that the TIX or TNX instruction is not multibly tagged with the computer in multiple tag mode. It is possible, when overlapping a CAS or LAS (POD 34 instructions) that a skip 1 or 2 will bypass the index instruction. In this case, any index modification already performed must be reconstructed to its original state before proceeding. If multiple indexing were permitted, the index registers at this point would have been OR'ed and reconstruction of the original contents impossible. Therefore, TLA is blocked under these conditions.

If the even addressed instruction is indirectly addressed, normal E time is blocked (02.15.40.1) during the IA cycle and the overlapping delayed for one cycle until the indirect addressing portion has been completed.

Each of these four index instructions is a conditional, 1-cycle transfer instruction dependent on a comparison between the specified index register and decrement portion of the instruction. This testing requires use of the index adders and possible recognition of an index adder 3 carry. Operation is similar to the normal operation except that it is now done during the period E0-E6 when the even instruction is awaiting its data from core storage.

By E2 time, both the tag register and address register have served their purpose for the even address instruction. Therefore, at this time positions 18-20 of the IBR are set into the tag register and positions 21-35 of the IBR to the address register. The address register at this point contains the transfer-to address in case the TLA transfer conditions are successfully met.

During the period of E0-E6, the decrement portion of the IBR is gated to the index adders. At E4 (D2) time, the specified index register contents are also routed to the adders together with a carry to XAD 17. This operation effectively subtracts (adds the 2's complement of) the index register from the decrement value; an index adder 3 carry occurs if the index register is less than or equal to the decrement. A carry from XAD position 3 turns on an index adder 3 carry trigger and determines the status of "TLA conditions met" (03.06.19.1) dependent on the actual instruction in the IBR. "TLA conditions met" decides subsequent circuit gating.

For a TIX or TNX instruction and no adder 3 carry (the index register was greater than the decrement) the index adders are returned to the index register. At this point, the index register contains the 2's complement of the decremented value and will have to be corrected during the next I time. Bits 1 and 2 of the IBR (03.08.11.1) decode

these instructions as non-indexable. This fact blocks normal gating of the index adders to the address register (03.06.08.1) for address modification.

To simplify the picture, assume that the instruction being overlapped is not a POD 34. If the transfer conditions are successfully met, the transfer-to address is immediately available from the address register which was loaded from positions 21-35 of the IBR at E2 time. If the transfer conditions are not met, the address register must be modified to reflect the next sequential address. During the initial I time, the program counter was stepped +1 (location of the instruction in the IBR). The next sequential address is obtained by routing the program counter and a carry to the index adders during E8 (D3) and setting the incremented value into the address register in time to be sent to MAR at E10 time of the cycle. Notice that whether the transfer was successful or not, the address register contains the correct core storage reference necessary for MAR.

At I0 time the TLA trigger is turned ON to gate additional cleanup functions. Two things remain to be completed: the program counter must be set to the new starting address; and the index register recomplemented to indicate the true decremented value for cases of TIX and TNX.

The first of these is accomplished by routing the address register and a carry through the index adders and to the program counter during I4 (D2) time. Notice that this new address represents one instruction beyond the one to be executed during this next I cycle. Because of this, the normal program counter advance is blocked at the following I8 time. The second item, recomplementing of the index register, is accomplished by routing the specified index register and a carry through the index adders and back again to the index register. (Remember that the index register output is always in 1's complement form.)

Overlapping a POD 34 instruction (CAS or LAS) complicates the circuitry and logic somewhat because the chances are 2 to 1 against the final execution of the odd addressed instruction. The final results of the POD 34 are not fully known until E10 time when the data has been received from core storage and a comparison made with the accumulator. Until these results are known, preparation must be made for all possibilities.

If the results of the POD 34 indicate a no-skip condition, the odd addressed instruction is executed and the next instruction address is dependent on whether or not the transfer conditions are met. If a successful transfer is indicated, the address register contains the correct transfer-to address. All other addresses are supplied by the program counter.

During the first I time, the program counter was stepped +1 (location POD 34 +1). During the period E6 (D2), the program counter is incremented through the index adders and contains the location POD 34 +2 (location TIX +1). Notice that this address can fulfill two conditions: A POD 34 skip of 1 or; a POD 34 no-skip condition and a transfer condition which was not met.

In order to supply a skip 2 address (location POD 34 +3), the program counter is gated to the index adders together with a carry to XAD 17 during E8 (D3). If, at E10, a skip 2 decision is made, the index adders are immediately returned to the program counter in time to be further gated to MAR.

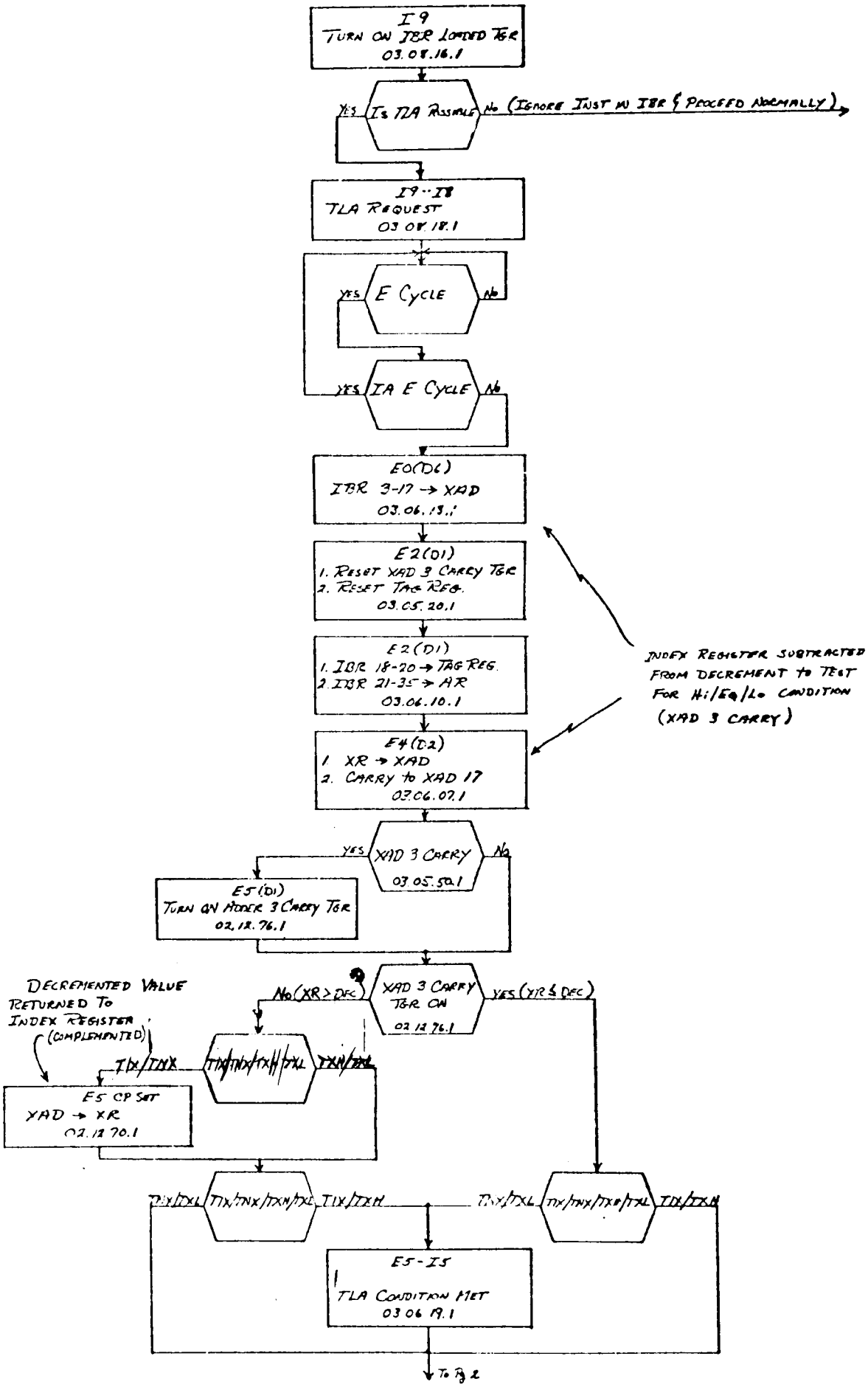
During the next I cycle the program counter must be set to a correct value. This is accomplished by incrementing the particular component (address register or program counter) that supplied the last address to MAR and returning the new value to the program counter. Because this is the address of one instruction beyond the next instruction being executed, normal stepping of the program counter is suppressed at I8 time.

During the early portion of the previous E cycle index register modification was performed for TIX or TNX on the assumption that odd addressed instruction would be executed. Recognition of either a skip 1 or 2 condition of the POD 34 must now nullify this execution and restore the index register to the original value. To do this, the decrement portion of the IBR instruction (positions 3-17) is gated to the index adders. The previous decremented 2's complement answer in the index register, when added to the decrement with a carry to XAD 17 produces the original value. This value set back into the index register completes the operation.

When overlapping a POD 34 instruction, interrupts are prevented from being serviced until after execution of one instruction following the overlap. This is necessary because with the modifications of the program counter and address register, it would be difficult, if not impossible, to provide the proper address in the program counter for storing in location 00000_8. If the even addressed instruction causes a trap (such as a floating point instruction), execution of the odd addressed instruction is nullified and the index registers restored as with a POD 34 skip condition.

The following table shows MAR gating for the various combinations of instructions and conditions.

Even Address Instruction \ Odd Address Instruction	Transfer Conditions Not Met	Transfer Conditions Met
Not POD 34	AR → MAR	AR → MAR
POD 34 (No Skip)	PC → MAR	AR → MAR
POD 34 (Skip 1)	PC → MAR	PC → MAR
POD 34 (Skip 2)	PC → MAR	PC → MAR



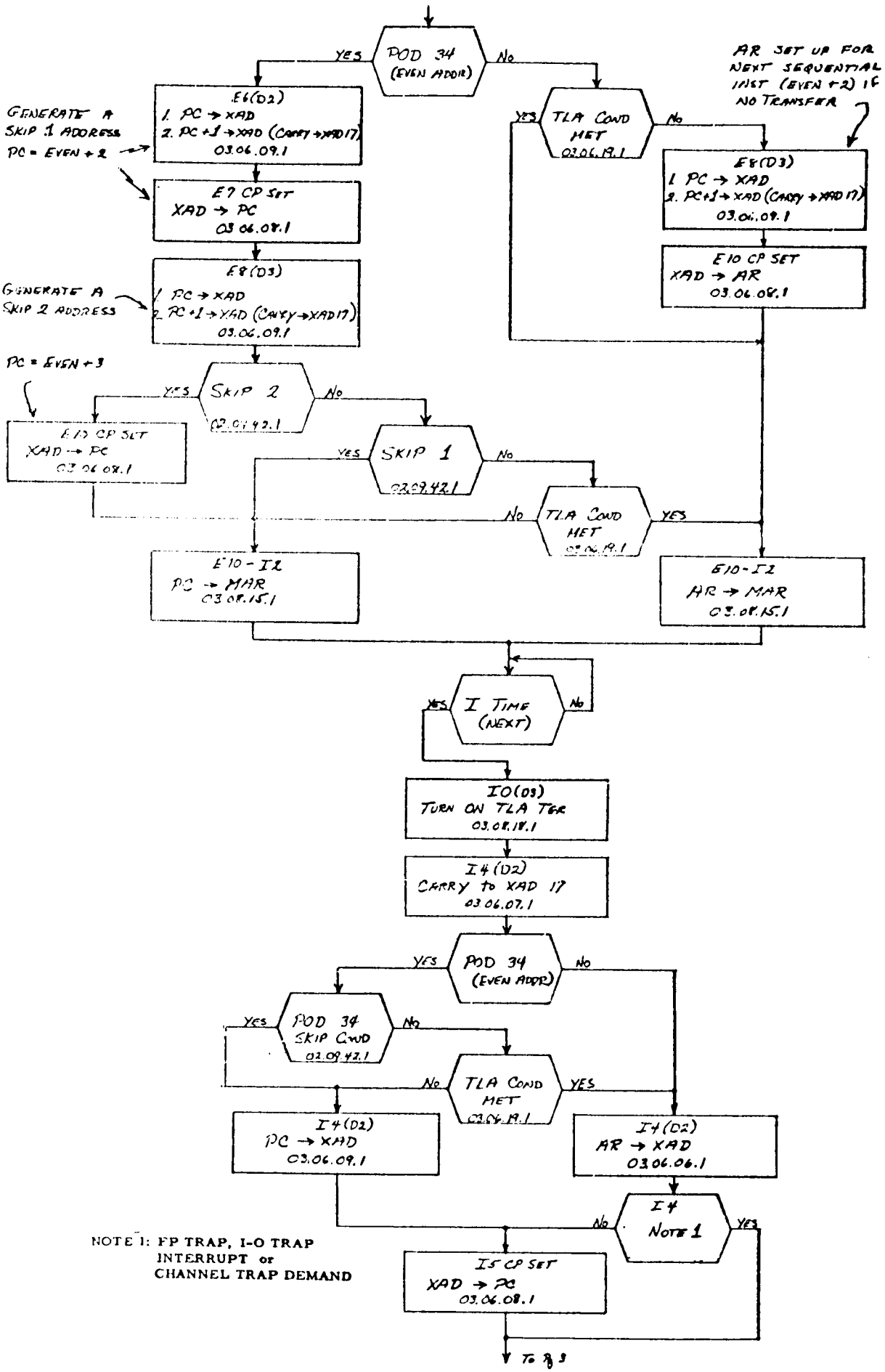
IPR TIX/TINX/TXH/TXL FLOW CHART

Pg 1 of 3

FIGURE _____

Bo 9/62

(TSKTIX 6)



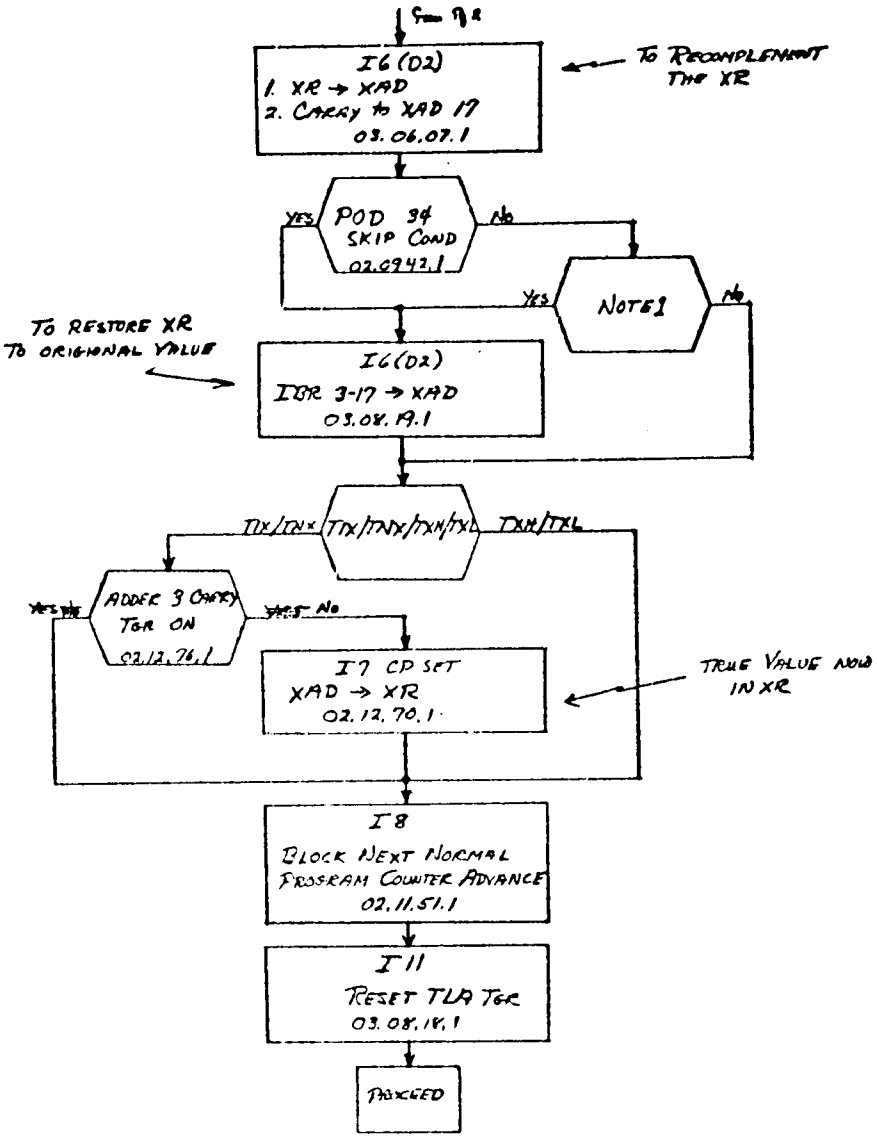
IXR TIX/TNX/TXH/TXL FLOW CHART

FIGURE _____

Pg 2 of 3

(IXR TXI 4)

(IXR TXI 7) OVLP 51



IOP TIX/TINX/TXH/TXL FLOW CHART

FIGURE _____

Pg 393

30 9/62

(IRTNX 8) OVLP 52

IBR TXI (Figure)

This instruction is an unconditional transfer which increments the specified index register by the amount indicated in positions 3-17 of the instruction. The basic logic is similar to that of the non-overlapped condition. One restriction, however, is that the TXI is not multi-tagged with the computer in multiple tag mode. This restriction is necessary in the case of trapping or POD 34 overlapping where it becomes necessary to restore the index registers to their original state. Multi-tagging would have already OR'ed the index registers and, therefore, made restoring impossible.

At I9, DELAYED time of the even addresses instruction, the IBR is loaded with the TXI and the IBR loaded trigger turned ON. Considering no TLA violations, "TLA request" becomes active and, with TXI decoding from the IBR, brings up "TLA conditions met" (03.06.19.1).

If the even addressed instruction is indirectly addressed, overlapping is suspended during the IA E cycle. During the early portion of the next E cycle IBR positions 18-20 are gated to the index register and positions 21-35 to the address register. The address register now contains the transfer-to-address.

Bits 1 and 2 of the IBR (03.08.11.1) decode this instruction as non-indexable and block the normal gating of the index adders to the address register (03.06.08.1) for address modification.

To increment the index register by the decrement value requires adding the decrement value to index register in the index adders and returning this value again to the particular index register. Taking the present index register value (in 1's complement form) and adding it to the decrement would effectively produce subtraction; therefore, during E4(D2)

time the index register is complemen by routing its contents and carry through the index adders. The actual incrementing is not completed until the following I cycle when decisions have been fully made as to whether or not to actually execute this TXI instruction.

Considering that the TXI is not overlapping a POD 34 instruction, the transfer is effected by gating the address register to MAR at the next E10 time. During the next I cycle, when the TLA trigger is turned ON, two functions remain to be completed; taking the incremented address register to the program counter and; incrementing the index register.

The first of these, setting the program counter, is accomplished during I4 (D2) time. This function is conditioned, however, by a possible floating point trap. If a trap is signalled, the program counter is forced to retain its present value (location of the even address +1).

Incrementing of the index register is accomplished during T6(D2) time when IBR positions 3-17 are routed to the index adders together with the index register contents and a carry to XAD 17. Taking the adders back to the index register completed the operation.

If a floating point trap has been signalled, IBR positions 3-17 are not gated to the adders; the output of the index register and carry to adder 17 effectively restores the index register to its original value and the TXI is nullified and bypassed.

When overlapping a POD 34 instruction, steps are taken to provide skip addresses if necessary. During E6(D2) the program counter is incremented through the index adders to provide a skip 1 address. During E8(D3) time the program counter is routed to the index adders again with a carry to XAD 17 to provide a skip 2 address if needed. Either

a skip 1 or 2 condition causes the appropriate program counter address to be sent to MAR; a no-skip condition allows the TXI to be executed and the address register sent to MAR.

During the next I time, the program counter or address register (whichever one supplied the last address to MAR) is incremented through the index adders and set into the program counter. In either case the program counter contains the address of one instruction beyond the next instruction to be executed; therefore, the next normal I8 stepping of the program counter is blocked by the TLA trigger.

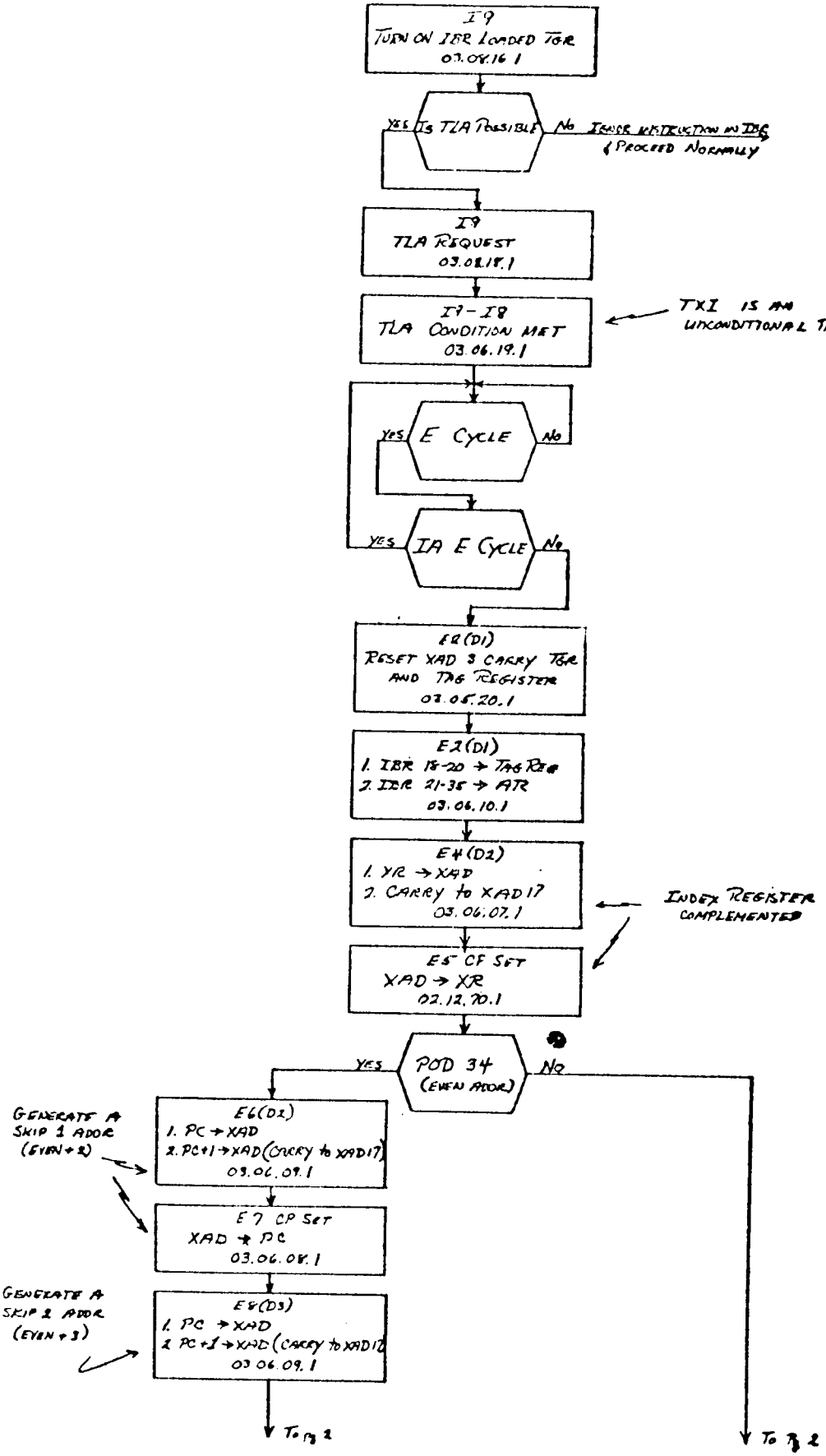
A skip 1 or 2 also nullified the final execution of the TXI instruction and casues the index register to be returned to its original status. This is accomplished, as explained above, by not gating the TXI decrement to the index adders. The value in the index register passing through the adders with a carry to XAD 17 effectively restores the original contents and completes the operation.

Interrupts are prevented from being serviced when overlapping a POD 34. Servicing occurs after the first instruction following the overlap.

The various combinations of gating to MAR are shown in the following chart.

Even Address Instruction / Odd Address Instruction TXI	Transfer condtions met
No POD 34	AR → MAR
POD 34 (No Skip)	AR → MAR
POD 34 (Skip 1)	PC → MAR
POD 34 (Skip 2)	PC → MAR

(IBTXI 3) OVLP 55



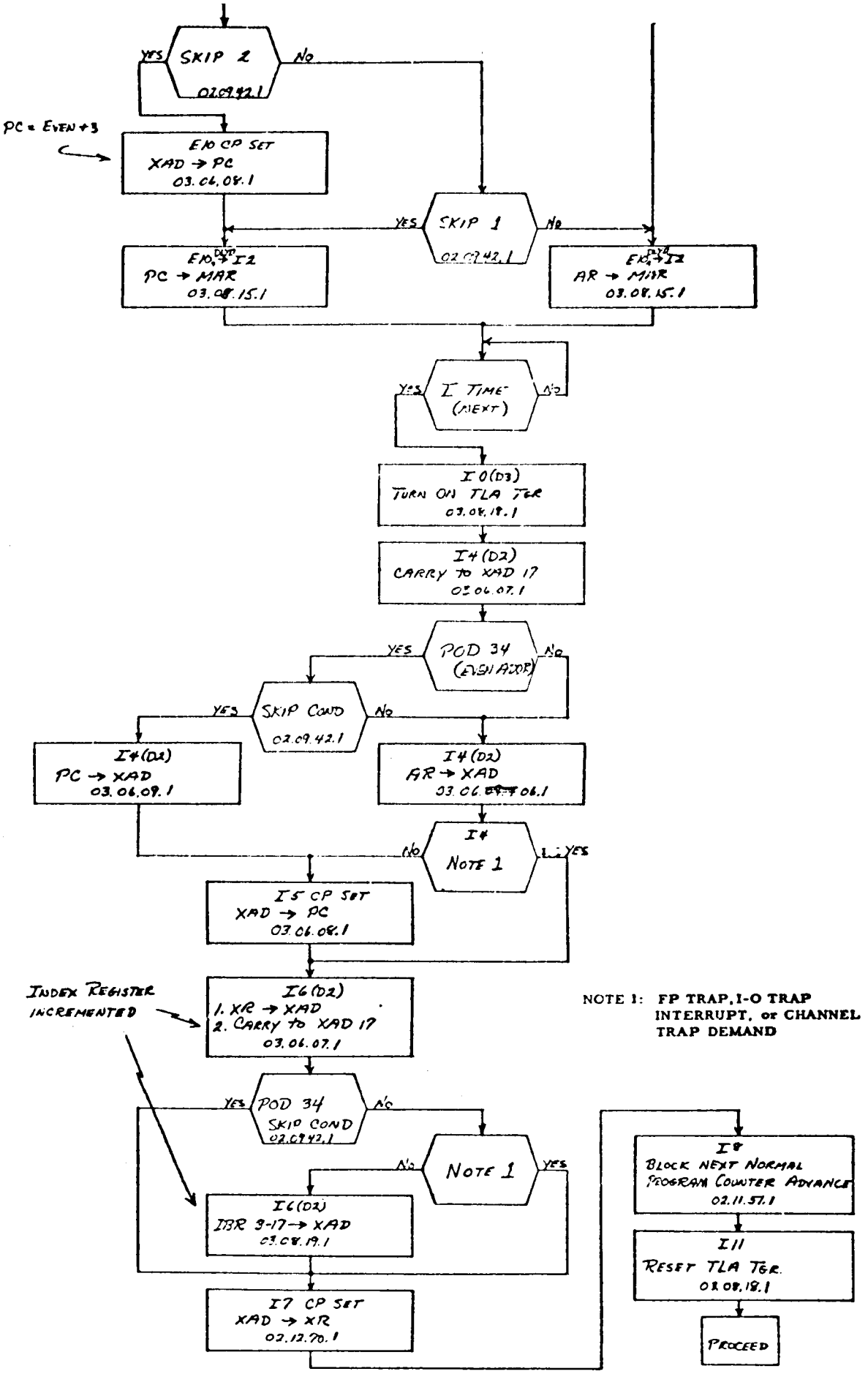
(IEXTXI 4)
OVLP 5C

IBR TXI FLOW CHART

FIGURE _____

Pg 1 of 2

Bo 5/12



NOTE 1: FP TRAP, I-O TRAP INTERRUPT, or CHANNEL TRAP DEMAND

IBR TXI FLOW CHART

FIGURE _____

Pg 2 of 2

2 of 2

(IBTXI 5)

OVLP 57

IBR TSX (Figure)

The logic of the TSX instruction is to store the 2's complement of the program counter into the specified index register.

Considering that there are no TLA violations, the early portions of the overlapping E cycle are concerned with routing IBR positions 18-20 to the tag register and positions 21-35 to the address register. The address register now contains the transfer-to address which is routed to MAR at E 10 time. The program counter at this time contains the address of the TSX instruction (even instruction + 1).

During the following I cycle, two functions remain to be performed: the complement of the program counter must be set into the specified index register/registers and; the address register incremented and set into the program counter.

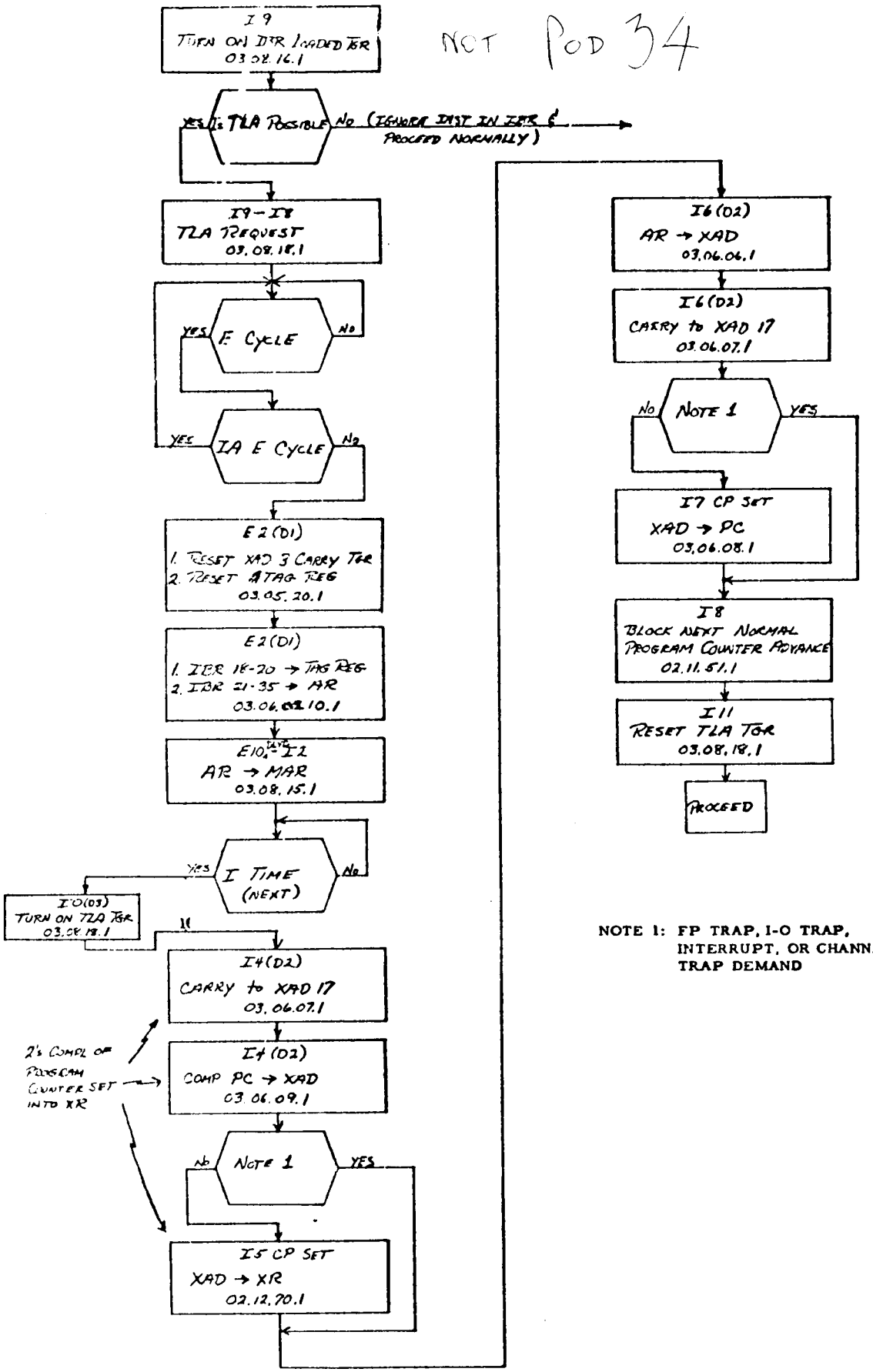
Many operations have the program counter modified during 14(D2) time. During the TSX operation, however, this must be delayed until after the program counter has been properly stored in the index register. It is during the 14(D2) period, then, that the complement of the program counter and a carry to XAD 17 are routed through the adders and to the index register. During the next two clock pulses, the transfer-to address is incremented through the index adders and set into the program counter. Because of the incrementing at this time, the following normal I8 program counter advance is blocked.

A trap situation, which occurs during or as a result of the even address instruction, blocks execution of the TSX. A floating point trap is recognized by I4 time; channel traps or interrupts are recognized by I0 time of the overlapping E cycle. In these cases, the program counter is not routed to the index register but remains at its setting of the even address instruction +1.

Multiple tagging is permitted; index register modification is performed only after it is known that there are no trapping conditions which will interfer with the TSX operation.

Because the TSX operation involves storing the program counter, it is not possible to overlap a POD 34 instruction. Recognition of the POD 34—IBR TSX combination immediately resets the IBR loaded trigger at I11 time and, therefore, blocks "TLA request" and the TLA functions which would normally be performed.

NOT POD 34



NOTE 1: FP TRAP, I-O TRAP, INTERRUPT, OR CHANNEL TRAP DEMAND

IBR TSX FLOW CHART

FIGURE _____

(TSX 3)

OVLP 60

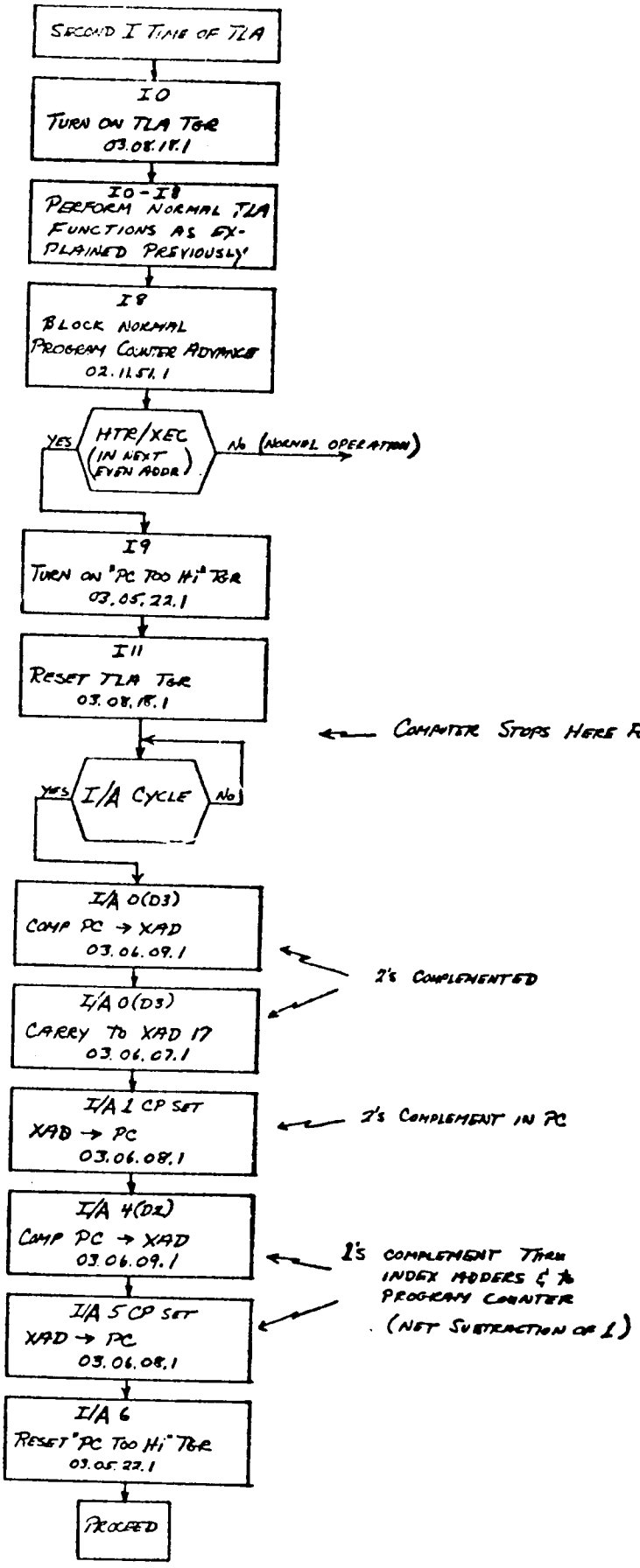
Rev 4/62

TLA Special Considerations - HTR & XEC (Figure)

During TLA operations, the program counter is stepped prematurely. The following I8 program counter advance is then blocked to bring the counter back to the proper value with respect to the program. There are two cases where the program counter remains too high with respect to the next instruction to be executed and must, therefore, be decremented to reflect the correct address.

The first of these is an HTR instruction. When the computer stops at the end of the I cycle, the program counter must indicate the address of the HTR instruction. During the first cycle following the halt (A-clock pulses), the program counter is decremented by: first, routing the 2's complement through the index adders and back into the counter during A0(D3) time and; second, routing the 1's complement through the index adders and back into the counter during A4(D2) time. The net effect is a subtraction of one and the computer stops with the address of the HTR in the program counter.

The second case is an XEC instruction. This instruction is composed of two consecutive I cycles; one to execute the XEC and one to execute the specified instruction. During normal operation, the program counter advance of the first I cycle is blocked (02. 11. 51. 1) and the second I8 pulse used to the program counter. Under TLA conditions, at I8 time of the following XEC instruction, the program counter already indicates the address of the next instruction. By decrementing the program counter during the second I0(D6) time (as explained for HTR), the next I8 pulse will step the program counter and bring it back in line with the program.



TLA - HTR/XEC PROGRAM COUNTER DECREMENTING

FIGURE _____

(TBDR)
OVLP 62