COMPANY CONFIDENTIAL

IBM CUSTOMER ENGINEERING

Preliminary Instruction Text

IBM 7030 DATA PROCESSING SYSTEM

LOOKAHEAD UNIT
BOOK A

The information presented in this preliminary instruction text is based on the electrical and mechanical status of the 7030 system as of December 15, 1960. The contents of this text has not been approved by the Engineering or Patent Departments and has not been edited for final printing. Distribution of this text shall be restricted to IBM employees authorized by the nature of their duties to receive such information.

LOOKAHEAD UNIT, BOOK A

CONTENTS

1.0.00 GENERAL INFORMATION

1.1.00 Introduction

The Lookahead unit of the computer is responsible for the proper sequencing and distribution of instructions plus performing all store operations. The need for the Lookahead is apparent when the overlap feature and speed of the computer are considered. The Lookahead unit allows complete utilization of the execution units. In previous computer systems, operations were performed by an instruction time followed by an execution time. The execution units were idle during the instruction time as were the instruction units idle during the execution time. In the 7030 system the instruction time and execution time are independent of each others operation. This means that instructions are being prepared while others are being executed. The diagram in figure 1.1-1 illustrates a comparison between both systems and the Lookahead relationship to the 7030 system.

Previous Systems

I Time

Execute

I Time

Execute

7030 System.

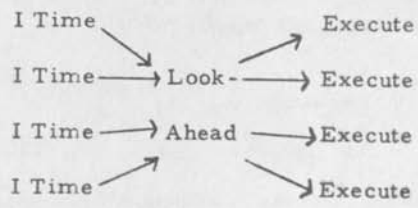


Figure 1.1-1

The function of Lookahead within the 7030 system is evident in the illustration. To further understand the performance of Lookahead, the following example is given.

An instruction has been decoded and is ready to be loaded into Lookahead. As soon as the instruction is loaded in Lookahead, the instruction unit is free to fetch the next instruction and prepare it. As the instruction unit is preparing the next instruction, Lookahead is processing the present instruction. When the next instruction is ready, the instruction unit loads it into Lookahead. In the meantime Lookahead has transferred the first instruction to the execution unit. The chart in figure 1.1-2 indicates the basic operation and communication of the instruction unit and Lookahead and further illustrates the overlap feature of the computer.

I Box	Lookahead	Execution
1- Fetch inst. (N)		
2- Decode inst. (N)		
3- Load inst. (N) to LA	3-1 LA Decode inst. (N)	
4- Fetch inst. (N+1)	4-1 LA Sequencing inst. (N)	
5- Decode inst. (N+1)	5-1 LA Distribution inst. (N)	Execution (N)
6- Load inst. (N+1) to LA	6-1 LA Decode inst. (N+1)	
7- Fetch inst. (N+2)	7-1 LA Sequence inst. (N+1)	
8- Decode inst. (N+2)	8-1 LA Distribution inst. (N+1)	Execute (N+1)
9- Load inst. (N+2)	9-1 Decode inst. (N+2)	
10- Fetch inst. (N+3)	10-1 LA Sequence (N+2)	
	11-1 LA Distribution inst. (N+2)	Execute (N+2)

Figure 1.1-2

By increasing the capabilities of both the instruction unit and Lookahead, as illustrated in figure 1.1-3, the overlap feature is expanded with even a greater increase in speed. Lookahead was designed for the most efficient overlap operation.

The sequencing of instructions through Lookahead is more complex than it may appear. Before any instruction can be executed, the Lookahead must insure that the previous instruction completed without error. A "no error" condition allows the execution of the next instruction. If the execution of

the previous instruction indicated an error, the computer program would be interrupted. This interrupt action is initiated and controlled by Lookahead. The action taken is a recovery of data and a reset of both I Box and Lookahead. Upon completion of an automatic correction routine, the normal program sequence would start with the instruction following the one which caused the error. It should be noted in the diagram in figure 1.1-3 that the Lookahead checks the previous instruction before transferring the data for execution.

The distribution of the instruction consists of the transferring of the operation code and working data (operand) to the proper execution unit. By decoding the instruction by type, Lookahead determines the execution unit and transfers the data. The execution units are considered to be the Parallel Arithmetic Unit (PAU), Serial Arithmetic Unit (SAU) and the Input-Output Units (I/O).

If the result of an operation, other than I/O, is to be stored, Lookahead will perform the operation. The load, data check, data transfer, operation check, and store operations are discussed in detail in the following sections. Figure 1.1-4 illustrates the Lookahead section in an overall computer diagram. Figure 1.1-5 is a more detailed breakdown of the lookahead relationship.

Physically lookahead is contained in frames 18, 19, and 20 of the 7030 System.

1.2.00 Machine Terminology

Before proceeding further, the reader should familiarize himself with the following terms and definitions included in this manual.

1.2.01 Basic Language

All data and control fields in Lookahead are in straight binary. No special coding is used. The decoding of Lookahead looks at the on or off (1 or 0) status of certain bit locations to determine the action necessary.

1.2.02 Operand

The actual working data required by the operation. This data can be an instruction word, index core storage word, internal register, or external memory word. Lookahead does not perform any operation on the data. It only receives, checks and transfers it to an execution unit.

1.2.04 Pre-Execution

The execution, from start to the point of modifying and addressable register is defined as Pre-Execution. This time includes accepting the operation code and operand from Lookahead, and executing the instruction to where the modification of an addressable register is necessary. The changing of the addressable register is accomplished in the execute time for the operation.

1.2.05 Interrupt

During each operation a comparison is made between the pre-set mask register and the indicator register. The indicator registor indicators can be changed at any time during an operation. A no comparison condition is indicative of an error and is called interrupt. Lookahead memorizes this line at the completion of the present instruction. When the execution of the existing operation is completed, the following Lookahead action is determined from the status of the interrupt line. If the interrupt line is active, Lookahead will initiate a corrective action, which caused the normal sequence of the program to be altered (interrupted), and cause both the instruction unit and Lookahead to be reset.

1.2.06 Houseclean

The action taken by Lookahead during an interrupt is called housecleaning. Housecleaning results in nothing more than clearing out all data in the Lookahead after the error was discovered. If some of this data are index core storage words, that were loaded into Lookahead by the I Box for temporary saving, they must be sent back to the index core storage unit. When housecleaning is over, the Lookahead is ready for a reload from I Box.

1.2.07 Pseudo-Store

Some instructions pertain to the instruction unit only and are completely executed there. If any of these instructions result in the changing of any index core storage words, the original words must be saved in case of an interrupt condition which may occur. Any instruction which is completely executed in the I Unit is out of the normal sequence. It is for this reason that any index core storage words that have been changed must be saved. These original words are loaded into Lookahead, with a proper operation code for Lookahead recognition. Any of these types in Lookahead are referred to as pseudo-stores. If an instruction earlier in the normal program sequence causes an interrupt, Lookahead looks for the pseudo stores and returns the original core storage words to I Box to back date the index memory. If no interrupt exists, LA will pass over the pseudo store levels when their normal program sequence time is due. An example of this action is to assume instruction #5 is being executed in an arithmetic unit. At this time the instruction unit is processing instruction #9, and recognizes it to be executed within the instruction unit. Instruction #9 is now being executed at the same time as instruction #5. It is obvious that this execution is well out of sequence. Instruction #9 is loaded into LA as a pseudo store after instruction #8

1.2.08 Recovery Level

This is a special level associated with certain branch instructions (VFL) In a branch, I Box always assumes that there will be a "no branch condition" and continues processing the following sequential instructions. If a branch should occur, the recovery level associated with the branch instruction contains the branch address and will initiate a lookahead and I Box recovery. This will be necessary since all instructions following the branch will be invalid.

in the normal manner. If an interrupt occurs on instruction #7, Lookahead looks for the pseudo stores and sends the index core storage words (instruction #9) back to index memory. The reason is clear that an error occurred before the normal sequence of instruction #9 should have happened. The program will resume with instruction #8 following any corrective routine, because index core storage was restored, the word would be modified again when instruction #9 is reprocessed and the proper result is maintained.

1.3.00 General Machine Logic Figure 1.3-1

The basic logic of the Lookahead can be broken down into five general operations plus error considerations. All instructions are received from the I Unit and loaded into Lookahead. This loading is dependent upon a signal from Lookahead informing the I Unit that it can be loaded. All control information is loaded from the I Unit directly. The necessary operand (data field) can be obtained from one of three distinct places. In some instances the operand may be present in the instruction format itself and is transferred directly into Lookahead from the IUnit. Generally speaking, however, the operand is requested from memory by the IUnit. The return address of the operand is the Lookahead level just loaded.

If the instruction calls for an internal register as the operand, the I Unit codes the information in lookahead. In this

latter case, the Lookahead recognizes this in the decoding of the instruction and causes the internal register involved to be routed to the proper execution unit at a later time. When an internal register is noted as the required operand the data field in Lookahead is not required. Store instructions are of the same type and do not require operands initially. The data field accepts the information at a later time from the execution unit and stores it. All control information is loaded directly from the I Unit as in any type instruction. Here the data field is reserved for the data to be stored.

Once the loading is completed, the operand involved must be checked for any data errors. The action here is dependent upon where the operand was received from. Where the operand was received directly from the IUnit, Lookahead does not require any checking because the operation is checked through the I checker when loaded. In cases of an internal register acting as the operand, Lookahead merely controls the gating out of the internal register to the execution unit. When the internal register is gated out, the data passes through the A checker where it is checked. The only time the Lookahead must initiate an operand check cycle is when the required operand is being received from external memory. Once the operand is received from memory, the Lookahead requests use of the I checker. When the Lookahead priority is satisfied the data is gated through the checker and checked for ECC. The data then returns to

Lookahead with correct parity bits. In case of an error in the I checker, a correct cycle is initiated to correct the error. If the error can not be corrected, the data is still put into Lookahead, but an error tag is also set to identify the error.

Once the loading and operand check cycle (if any) are complete, the Lookahead is ready to go into the instruction execution phase of the operation. Before LA can transfer the data to the execution units, it is necessary to first decode the type instruction involved (also to insure that the previous operation was completed without error). During the loading process, the I Unit tags the data to include pertinent facts about the instruction. The Lookahead uses these tag bit designations along with the operation code to decode each instruction by type. There are four types of instruction that the Lookahead must decode. They are Floating Point (FLP), Variable Field Length (VFL), Input Output (I/O), and I Unit. In addition to these basic four types, Lookahead also uses the op code and tag bit designations for control purposes. The decoded control lines condition the transfer circuits to the proper execution units and are used internally for Lookahead control purposes.

At this time Lookahead is not sure that the previous operation has completed without error. Still the transfer of the operation code and data are allowed to take place to the various

execution register. However, this data will only be acted upon up to the point where an addressable register must be changed. The operation will terminate here and the actual changing of the addressable register is not allowed until Lookahead has checked for error conditions in the previous instruction. This action is called pre execution and allows the execution unit to process the data, as much as possible, before the error or no error condition can be detected. If the execution unit were allowed to change the addressable register and put the result of the present instruction into it and then LA found the previous instruction were in error, the information previously in the addressable register would be lost.

A signal from the execution unit indicating that it has accepted the data and started pre-execution enables the LA to continue to the next step of its operation. In order to allow the execution unit to execute (change an addressable register) the present instruction, tests must be made to determine the error status of the previous instruction. Lookahead must make these tests and inform the execution unit of the results. If the tests indicate no error, the complete execution of the present instruction is allowed to take place. The error condition is a result of the sampling of the indicator register. The indicator register is updated from the lookahead during each operation. Actually the changing of the indicator is not done in the LA, but rather the results of any changes are stored there. During the LA

loading, any indicators changed as a result of instruction preparation are recorded in the IUnit and the result appears in the indicator register field after the load. Any indicators changed as a result of indexing are transferred directly from the updated index result indicator (UXIR) register in the IUnit to Lookahead. Any indicators affected by the previous operation in the I/O or arithmetic operations are entered directly into the main indicator register. Upon completion of the previous instruction, the result of the comparison between the main indicator register and the mask register is sent to Lookahead as interrupt or no interrupt. Assuming no interrupt (previous operation completed without error), the Lookahead signals the execution unit to proceed with the execution of the instruction. Also, LA transfers the indicator field in LA to the main indicator register. Again during the execution of this instruction the execution unit involved sets its indicators directly into the main indicator register. At the end of this instruction the LA must again test for a no error condition to see if the next operation will be allowed to complete.

As mentioned during loading, if the operand required for execution was an internal register, the LA routes the data to the proper execution unit at the proper time. Action is taken during this LA time to complete the transfer of the internal register to the execution unit. Of course, this action is dependent upon the same conditions as before. That is, the

previous instruction completed without error.

If the previous instruction caused an interrupt, the execution unit is not be allowed to complete the operation presently in the pre-execution state. The presence of the interrupt line causes LA to signal the I Unit to do a houseclean because it can be processing instructions beyond the one causing the interrupt. In instructions that are executed in the I Unit, words in index core storage are altered. Because the I Unit is processing these instructions out of sequence - that is the particular instruction can be several program steps behind the one presently being executed in the execution unit - it becomes necessary to retain the old index core storage word in case of an interrupt prior to the normal sequence of the I Unit instruction. I Unit houseclean action will restore these words to index core storage, clear out any other information and reset their control circuits for loading and instruction preparation. When the program interrupt is over and the program is again stepping through the normal routine, all data in index core storage will be in its original form as before the interrupt. Upon completion of the I Unit houseclean, LA is signaled to houseclean. LA housecleaning consists of clearing all data left in LA after the interrupt except any index core storage words that were sent from the I Unit to be saved. These words are routed back to I Unit to restore the index core storage as mentioned above. The

LA control circuits are reset and houseclean is complete. After the interrupt program routine, the program starts on the instruction which followed the one which caused the interrupt and the I Unit and LA are back to normal operation. Assuming no interrupt, LA decodes the instruction just executed to see if a store is required. If a store is not required, LA is ready to load another instruction and repeat. However, if a store is required, LA must determine the location of the store and take the proper action before another load is allowed. There are three types of stores and are only possible through Lookahead. By further decoding of the operation code and tag information, obtained during the load, LA determines the area required for the store. The three possible areas for storing are external storage, the internal registers and index core storage. LA action is different for each type. For storing into external storage, LA receives the required data from the execution and because this is a store to external memory, LA must insure that the data has the proper parity structure and then generate error correction code (ECC) for the data. The parity check and ECC generation is accomplished through the I checker. Lookahead makes a request to the checker and when the priority is satisfied LA routes the data through the checker where the necessary action takes place. The data, with ECC generation returns to LA. Lookahead then makes a request to the memory bus priority system. When the request is satisfied, LA routes the data to the storage bus for storage. When

the store is required to I Unit, Lookahead requests the I checker. When the priority request is honored, the data is routed through the checker with a control line to check the data for proper LA parity. The output of the checker is gated into the I Unit with the proper I Unit parity. Lookahead also initiates the Clear and Write cycle for index core storage. I Unit, using timed lookahead pulses, gates the data to the cores. If an error was detected on the data passing through the I checker the store would complete as mentioned, but the Instruction check indicator in the indicator register would be turned on. In an internal type store, the data is already in the proper parity and Lookahead merely controls the data path to the proper destination. Upon completion of a store operation, Lookahead is ready to load another instruction.

When special type instructions or conditions arise during an operation, Lookahead action is somewhat varied. The procedure just described is the sequence and function performed by Lookahead in the majority of the operations. A list of these functions are:

1- Receive instructions from the I Unit.

2- Receive and check operands.

3- Transfer operation codes and operands to the proper arithmetic unit.

4- Insure that each instruction is completed without error.

   a) Interrupt if error.

   b) Houseclean I Unit and Lookahead.

5- Receive operands to be stored.

6- Initiate all store operations.

To take maximum advantage of the overlap system, Lookahead duplicates four distinct areas to accomplish the logic just described. Each of these areas is called a level and each level has identical functions. Referring to figure 1.1-3 each vertical column in the Lookahead area represents one level. By understanding the operation of one level, all levels are understood. The only restrictions to the operation is that no two levels can be performing the same function simultaneously. Only one level can be loading at a time, one level checking operands at a time, etc. However, when one level is loading, another can be checking operands, another transferring data to execution, etc., as indicated in figure 1.1-3.

2.0.00 FUNCTIONS AND CONTROLS

2.1.00 Level Description

Lookahead contain four separate areas to buffer instructions for execution. Each of these four areas or levels contain identical registers and controls. The levels are named level 1, 2, 3 and 4. Each level contains an operation code field, data field, tag bit field, indicator field and instruction counter field. A brief explanation of each field is listed below. Figure 2.1-1.

Operation Code Field Figure 2.1-2.

This is an 11 bit field specifying the operation to be performed. The final bit is a check bit on the other 10.

Operand (Data) Field Figure 2.1-3

This field contains 64 information bits and 12 check bits. The information in the field can be any of the following:

1. Operand required for execution
2. Data to be stored
3. VFL instruction information
4. Input-Output instruction information
5. Information for computer recovery.

Tag Bit Field

The tag bit field is an eight bit field containing information concerning the status of the instruction stored at the level. The tag bits are used in Lookahead decoding to further control an operation as it sequences through the level. Each of the bits and their function is listed below.

1. Level Filled (LF) tag bit

   Applies to the operand and indicates that the operand has been entered in the level.

2. Level Check (LC) indicates that the operand at the level has been checked and the check bits converted to those required by the operation.

3. Internal Bit indicates the operand required for execution must be fetched from an internal computer register.

4. Instruction Counter (IC) Bit indicates that the value stored in the instruction counter field of the level is valid for interrupt interrogation. It also indicates the final instruction level of multiple level instructions.

5. Lookahead Operation Code (LAOP) indicates the information stored in the Lookahead operation code field is for Lookahead control purposes only.

6. Word Boundry Crossover (WB C) used in conjunction with "not LAOP" indicates the operand associated with a VFL instruction at the level crossed a storage word boundary. When used in conjunction with LAOP, the bit indicates that the operand at the level had ECC bits generated when it was loaded.

7. No Operation (NOOP) Bit indicates that the instruction buffered in the level is to be executed as no operation.

8. Disconnect (Disc) Bit indicates that the lookahead level will not participate in any Lookahead action. This bit is essentially a maintenance feature.

Indicator Field

The indicator field is a 15 bit field which indicates the status of the indicators affected by instruction preparation in the instruction unit.

Instruction Counter Field

The address of the instruction immediately following the instruction at the level is stored in the instruction counter field. The value is stored during instruction execution for re-entry into the ma n program in the event of an interrupt. In multi-level instructions, the value is only valid in the final instruction level.

In addition to the fields just described, the look-ahead also contains one Look-ahead Address Register and one instruction counter buffer. These final two fields serve all four look-ahead levels.

Look-ahead Address Register (LAAR)

The LAAR is a 19 bit register, of which the final bit is a check bit. The register is used to buffer an operand address when necessary for instruction execution. Instructions requiring the LAAR are of the store type or those specifying the contents of an internal register as the operand.

Physically the LAAR is located within the memory bus control unit, but it functions logically within the look-ahead.

Instruction Counter Buffer

The instruction counter buffer is a 21 bit field used to buffer the contents of the IC field of the instruction currently being executed (N+1). The instruction counter value of a level is transferred to the IC buffer as the instruction begins execution and is retained during the interrogation of the program interrupt procedure for re-entry into the program following the correction routine.

2.2.00 Lookahead Level Requirements

Each look-ahead level can accomodate a single half word instruction which requires only one operation. Some instructions, however, may contain more than one operation. When more than one operation is necessary to complete an instruction, additional levels of look-ahead are required. The instruction level requirements are determined by the instruction unit and loaded into look-ahead accordingly. The level requirements for floating point operations range from a minimum of one level to a maximum of two levels. Variable field length instructions, however, range from a minimum of two levels to a maximum of six levels. The level requirements are divided into classes and include fetch only operations, store only operations, fetch and store operations, and recovery levels.

Because all floating point instructions are half-word instructions, all single floating point operations require only one level of look-ahead. Some floating point operations require a fetch and a store operation or two fetch operation and require two levels of look-ahead buffering. In floatingpoint operations, these are the only exceptional cases.

In variable field length instruction, the operation code requires a full word for the instruction format. Because of the added information necessary in VFL operations, part of the data field is required to buffer the additional op code information. With a portion of the data field used for the op code, the operand required cannot be placed at the same level as the instruction op code and thereby an additional level is required for the instruction data. If the required operand cross a word boundry, two operand fetches are required resulting in two additional levels of look-ahead buffering. If the instruction is a fetch and store type instruction,

an additional level is required for each operation. If the instruction crosses a word boundry, two levels are required for the operands, and two for the store levels. When progressive indexing is used, still another level is required. The PX level is a recovery level containing recovery information in the event of an interrupt from an earlier instruction. VFL level requirements consist of an operation code level (always the first level), operand (fetch) levels, store levels and progressive indexing levels. Figure 2.2-1 and 2.2-2 show the level designations including all the possible conditions for FLP, VFL, I/O and I Unit instruction.

When more than one level is required for the execution of an instruction, the operation code is valid only in the first level. All succeeding levels have look-ahead op codes (LAOP), used for look-ahead control purposes only.

When the first level of an instruction is mentioned, it doesn't mean necessarily that level one of look-ahead is used. The first level of any instruction is loaded into the next available level in look-ahead. Any of the four look-ahead levels can be the next available level ready to load. Look-ahead levels, 1, 2, 3, and 4 are only names given to each level and in no way indicate the levels designated for loading multi-level or single level instrutions. The reader should be familiar with this terminology because in the later sections of the manual, the terms instruction levels and look-ahead levels are used fluently.

Even though there may be several look-ahead levels associated with one instruction, each look-ahead level essentially operates independent of any other level. The look-ahead decoding circuits evaluate and determine the correct operation for each level. When an instruction requires more than four levels of look-ahead buffering, the two additional levels are held in the instruction unit until a free level is available

in look-ahead. As an example, using a six level VFL instruction, only four levels can be buffered in look-ahead at a given time. Since the first level is the operation code level, the look-ahead level becomes ready to reload as soon as the op code is transferred. When the level is available, the fifth level of the instruction is loaded. The status of the instruction now shows instruction levels 2, 3, 4, and 5 buffered in look-ahead, instruction level 1 (op code level) is in the execution unit and instruction level 6 in the instruction unit, waiting to load in the next available look-ahead level. Because the second instruction level is the first operand level, that look-ahead level becomes available when the first operand is transferred to the execution unit. When the level is available, the sixth instruction level is loaded into look-ahead. The status of the instruction now shows instruction levels 1 and 2 in the execution unit and levels 3, 4, 5, and 6 buffered in look-ahead. As the next level of look-ahead is transferred (instruction level 3), the look-ahead level then becomes available to load the first level of the next instrution. The process repeats one level at a time until the full instruction has been executed by the execution unit. As each level becomes free, the next instruction is loaded into that level from the instruction unit.

Input/Output instructions always require two levels of look-ahead buffering. The first instruction level is the data transfer level while the second level is a dummy level used to reliably interrogate the interrupt mechanism.

The instruction unit requirements for look-ahead vary from one level to three levels dependent upon the operation being performed and the indexing involved. These instructions are executed within the instruction unit and therefore the necessity for look-ahead requirements are limited. In general, the requirements are necessary for recovery data storage for interrupt purposes, internal operand fetches for

instruction unit execution, look-ahead level clearing for special instruction memory stores, and in some cases just an indicator transfer only is necessary. These areas are covered in detail in the latter sections of this manual.

2.3.00 Lookahead Sequencing

There are five operational steps necessary to sequence an operation through Lookahead. Each of the five operational steps is controlled by a trigger. The associated counter and operation step it controls is listed below.

Indexing Arithmetic Unit Counter (IAUC)

This trigger controls the level loading from the I Unit. With the IAUC trigger on, Lookahead develops "load enable" signals and sends them to the I Unit. These signals inform the I Unit that Lookahead can be loaded. The response to the enable signals are the actual load pulses from the I Unit.

The I Unit load pulses time the data and control information to the Lookahead Registers. The areas loaded from the I Unit in every load cycle are the operation code field, indicator field, IC field and tag bit field. The Lookahead Address Register (LAAR) is set from the I Unit when the operation is a store, internal fetch or if the required operand is from external memory. The LAAR is not affected when the required operand data is loaded directly from the I Unit. The loading process sets the level filled (LF) and level check (LC) tag bits on all type loads except loads requiring the operand from external storage. When the operand is required from external storage, the setting of the LF and LC tag bits is the responsibility of the OCC counter. On all other loads the data check and conversion is accomplished during the loading from the I Unit or is not required.

The completion of the loading operation steps the IAUC counter to the next Lookahead level, providing the next level is not performing a store operation. If the next level is completing a store operation, the stepping of the IAUC counter is delayed until the store is complete. Once the storing is complete, the IAUC counter is allowed to advance to initiate loading in the next Lookahead level.

Anticipation circuits are included in loading to save time. Using the anticipation circuits, the load enable signals can be generated before the actual stepping of the

counter. These anticipation circuits are only active when it is assurred that the counter will step and the next level loading will occur.

Operand Check Counter (OCC)

The function of the OCC is to check operands received from memory and to convert them to the proper execution parity. This action is necessary on operands received from external storage only. If the operand were received directly from the I Unit, it is checked and converted when loaded through the I checker. When the required operand is an internal register, Lookahead controls the gating of the data only. When Lookahead gates the internal register to the execution unit, the data passes through the arithmetic checker where the necessary data check occurs. Parity conversion is not necessary because the internal registers already have the proper parity structure for execution.

The data check and parity conversion for data from external storage is accomplished by the I checker. A level filled (LF) tag bit comes on in Lookahead indicating that the data has been received from external memory. The OCC counter and the LF tag bit form a priority request to the I checker. When priority is satisfied Lookahead gates the data and check bits to the I checker with the proper control signals to check ECC and generate Lookahead parity. After the check and conversion, the data is routed back into Lookahead and a level check (LC) tag bit is set. If an error had occurred during the check, Lookahead automatically starts a correct cycle to correct the error. If an error occurs again, the data is routed back to Lookahead and an appropriate error tag is set.

Upon the completion of the OCC check cycle (or correct if necessary) the OCC counter is allowed to advance to perform the check on the next level. Again, as in the IAUC function, anticipation circuits are included to save time. Because the level filled (LF) and level check (LC) tag bits are set, Lookahead can start the next operational step of this level. The LF and LC tag bits indicate the instruction preparation is complete and execution may begin.

Transfer Bus Counter (TBC)

The function of the TBC is to transfer the data, operation code and other pertinent control data to the proper execution unit. The TBC is allowed to function when Lookahead recognizes that the LF and LC tag bits are on. By decoding of the operation code field and tag bit field, Lookahead determines the execution unit involved and other control information necessary for the TBC function. A DC gate out of the operation code field and data field (if not internal) are accomplished by the TBC output. These DC lines hold up the inputs to the execution unit execution register and the execution data input register (via TOB). The TBC decoding conditions a timer which develops the required signals to tell the execution unit to gate the data and the operation code into their respective registers. The timer (Transfer Bus Timer T) also signals the execution unit to start (pre-execution). With the acceptance of the data and operation code, the execution unit returns a signal informing Lookahead that the information has been accepted. The receipt of the signal from the execution unit allows the TBC to advance to the next level.

If the operand required for execution is an internal register, the internal tag bit in Lookahead is on. During the decoding, Lookahead recognizes the internal tag bit and does not signal the execution register to gate the data and operation code in. The start signal is also blocked. The operation code field output is holding up the input to the execution unit register from the TBC decoding. The actual gate in of the data and operation code is accomplished during ABC time. The TBC is prevented from stepping until the execution unit signals the data has been accepted. These signals are sent during ABC time. All internal fetches are the responsibility of the ABC.

Arithmetic Bus Counter (ABC)

The basic function of the ABC is to perform all necessary functions for the proper operation of the interrupt mechanism. These functions include the transferring of the Lookahead indicator field to the indicator register, transferring the instruction counter

field to the IC Buffer, and in the event of an interrupt, marking the remaining unexecuted instruction for proper handling.

Secondary functions performed by the ABC include internal operand fetches, receiving data to be stored from the arithmetic unit, and performing the store operation if the store address is an internal register.

Before the ABC can perform any operations, Lookahead must be assurred that the previous instruction was completed without a program interrupt. An indication that the previous instruction completed without error is represented by a Modify Addressable Register (MAR) mode condition. At the completion of the execution of each instruction, Lookahead makes a series of tests to check the instruction. The tests insure that all units involved with the present instruction have entered their indicators in the main indicator register. When all the tests have been completed Lookahead will memorize the interrupt line. The memorization is in the form of two triggers. One is interrupt next instruction if the interrupt line is active and the other is the MAR mode trigger if the interrupt line is inactive (no interrupt). Lookahead can memorize the line at this time because the tests indicated that all indicators associated with the instruction have been entered in the main indicator register and therefore the indicator register comparison to the pre-set mask register is valid.

With the MAR mode present in Lookahead, from the previous instruction the ABC is allowed to function for the present instruction. The indicator field in Lookahead is loaded from the I Unit during the loading process. Any indicators altered by the I Unit in preparing an instruction are buffered in the indicator field until the instruction can be executed in its proper sequence. The ABC counter and MAR mode start a transfer indicator timer to transfer the indicator field to the main indicator register. The timer also sends a signal to execute (modify addressable registers) is sent to SAU. The signal is not necessary for PAU because they duplicate the MAR mode test in their circuits and thereby know that they

can execute the instruction. The MAR mode tests are duplicated in PAU circuits for speed purposes only. The transfer indicator timer also transfer the IC field to the IC buffer on all single level instruction. On multi-level instruction the IC field is not transferred to the IC buffer until the last level of the instruction. With the IC field in the IC buffer and containing the address of the present instruction plus 1, an instruction counter value is available to the I Unit in the event that the present instruction should interrupt. On the interrupt operation the IC buffer is returned to the I Unit and is used to restart the normal program sequence following the correction routine for the interrupted instruction.

When an internal operand is required, the ABC makes the fetch to the specified internal register and routes the data to the execution unit. When the data is routed from the internal register to the execution unit data register, it passes through the A checker where the proper parity check is performed. The presence of the internal bit during the ABC decoding allows an A bus timer to be started. The A bus timer times the data from the internal register to the execution unit and also signals the execution unit to gate in the operation code and start. A signal that the data has been accepted is used by Lookahead to step the TBC to the next level. (TBC step is held up on internal fetches.)

If the operation required a store, the ABC dwells after transferring indicators until signalled from the execution unit that the data is ready. The data is routed into the Lookahead data register for storing into external memory or index core storage. The actual storing is accomplished by the SCC. An indication that the SCC has action is accomplished by the ABC resetting the LF tag bit. If the store is to an internal register, Lookahead controls the data routing directly from the arithmetic unit to the required internal register. All store functions to LA and any internal registers are performed with the A bus timer.

The ABC stepping is determined by the type operation performed. A store or internal fetch type instruction delays the ABC advance until the store or fetch is completed. The stepping in these two cases is accomplished with the A bus timer. In operations not involving a store or internal fetch, the transfer indicator timer performs the stepping.

With the stepping of the ABC counter, the ABC is allowed to function for the next level.

Store Check Counter (SCC)

The SCC is responsible for completing all store functions to external storage or index core storage. Completion of the ABC action is generally an indication for the SCC to initiate the storing function. This indication is in the form of logical combinations of the tag bits and operation code field. When no stores are required the SCC counter is stepped simultaneously with the ABC counter. When stores are required the SCC is not stepped until it completes its action.

On stores to external memory, the SCC decoding makes a priority request to the I checker (first priority). When the request is honored, a store check timer is started to gate the data from Lookahead to the I checker. Signals are generated to the I checker for parity check of the data and conversion of the data check bits to ECC. With the data check and conversion complete, the data is routed back into the Lookahead data register. The data is ready for storing into external storage. Lookahead makes a request to the memory bus priority system. When the priority request (4th priority) is honored, a store data timer is conditioned to time the data to the memory bus control unit where it is stored.

Where the store is required to the I Unit, the SCC action is slightly altered. The initial SCC decoding makes a priority request to the I checker. For stores to the I Unit, the Lookahead has fourth priority. When the priority request is honored, a Lookahead to I timer is conditioned to time the data through the I checker to the I Unit X register. Signals are sent to the I checker to check the parity on the data transfer and to convert the check bits to I Unit parity.

Because the instruction unit has no direct access to the contents of the internal registers, requests for their contents must be made through Lookahead. The data is buffered in Lookahead in the usual manner during ABC time. The SCC action is similar to the I Unit index core storage store operation except the data is routed to the appropriate I Unit working register instead of the X register. Data check and conversion are accomp-

lished in the usual manner.

Completion of the store function allows the SCC to advance to the next level. As soon as the SCC is stepped, the level is again available for loading under control of the IAUC counter.

2.4.00 Counter Controls

The five control triggers just described are duplicated in each of the four lookahead levels. Each level can completely sequence an operation through lookahead. Each control trigger of a level is electronically tied to the corresponding control trigger in the other levels. The lookahead system has 5 distinct closed counter rings consisting of four triggers each. Figure 2.4-1. shows the counter layout.

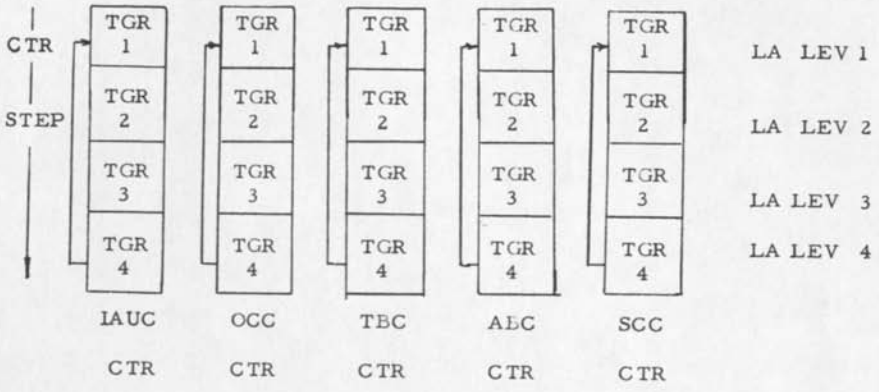


FIGURE 2.4-1.

Only one trigger in each ring can be on at a time. This means that only one level can be loading at a time, only one level operand checking at a time, etc. However, there may be more than one ring counter on in a given level. That is, for example, the IAUC 1, OCC 1, TBC 1, ABC 1 and SCC 1 triggers can all be on in level 1. The same holds true for the other lookahead levels also. In operation however, the counters operate functionally from IAUC, OCC, TBC, ABC to SCC. The indications resulting from one counter operation allow the next counter in the level to function, etc. For example, the OCC 1 counter

cannot function, until the IAUC 1 counter has completed the loading cycle. Indications that the loading has completed allows the OCC action to take plaee. This may occur even before the IAUC counter is stepped to the next level. The restrictions placed upon the counters stepping are such that no counter may pass another and the IAUC cannot step into another level until the SCC has left the level. As an example of the first condition assume that the IAUC counter was prevented from stepping after a loading operation. The IAUC counter then remains in the same level until the stepping is allowed. The OCC counter for that level will function because the loading is complete. However the OCC cannot step to the next level because the IAUC counter has not stepped, so it too must wait. The stepping of the counters can be simultaneous.

An example of the second condition occurs when the IAUC has loaded a level and is ready to step to the next level. If the SCC action is not complete at the level, the IAUC stepping is blocked until the SCC counter steps out of the level. The SCC step from the level allows the IAUC to step into the level. The conditions just explained are the only restrictions placed upon the counter stepping functions. Anticipation circuits are used to save time when stepping the counters. In some instances the conditions within the lookahead are such that the counter will be stepped in the next cycle. The anticipation circuits recognize these conditions and allow an early stepping of the counters. The stepping of the counters is covered in the areas dealing with the function they perform. Figure 2.4-2 shows the overall lookahead logic broken down to counter and functional operation.

2.5.00 DATA FLOW PATHS AND COMMUNICATION

The data flow paths to and from lookahead are shown in Fig. 2.5-1. All data transfer is parallel. By using specific timers started by lookahead decoding the data is routed in and out on the busses shown. All data flow paths shown enter and leave the lookahead data fields in all four lookahead levels.

Fig. 2.5-2 shows the data flow and basic control communications between lookahead and the rest of the 7030 system. The dotted lines within the I checker indicate that the data flow can follow any of the indicated paths. The data flow is conditioned by I Unit or Lookahead controls. For convenience, the communication lines are labelled with the lookahead control causing the transfer (i.e. the transfer bus timer in lookahead causes the data transfer on the transfer out bus 'TOB'). The lookahead action consists of sending and receiving data over the data busses listed and generating specific controls to time the data and control information transfer.

3.0.00 INSTRUCTION PREPARATION

The instruction preparation phase of lookahead operation consists of loading a level from the instruction unit and checking the operand for proper parity structure and errors. The IAUC counter performs the loading function and the OCC controls the operand checking.

3.1.00 GENERAL DESCRIPTION

The IAUC action Figure 3.1-1 consists of sending load enable signals to the instruction unit informing them of a free lookahead level. The instruction unit responses to the enable signals are the actual load pulses. The instruction unit loads the operation code field, indicator field, tag bit field, IC field and, depending upon the type of instruction, the data field is loaded and the LAAR is set. Some instructions do not require the use of the LAAR and other instructions do not contain the required data for execution. In the latter case the data either is not required for execution or the data must be fetched from external storage. The setting of the LAAR is necessary when an internal register is required for the operand or when the instruction involves a store operation. When either of the two conditions are necessary the instruction unit sets the required address into the LAAR.

Where the required operand is set into lookahead from the instruction unit or when the operand is not required, the LF and LC tag bits in lookahead are set. If the required operand is from external memory, the instruction unit makes the fetch request to the memory

bus priority system. When the priority is satisfied, the operand is sent from memory bus to the lookahead data field. The LF tag bit in lookahead is set when the memory bus sends the data to lookahead. The LF tag bit indicates that the operand required for execution has been loaded. The memory word as it comes from the control bus unit has the ECC bits with it. Lookahead then must route the data through the I checker to check the ECC and then convert the ECC bits into the proper parity for execution. This action is the responsibility of the OCC counter. The OCC counter requests the use of the I checker. When the I checker priority is satisfied, the OCC action routes the data to the I checker and sends the proper signals to the I checker to check ECC and generate LA parity. If, during the ECC check, a single bit error occurs, the OCC action automatically starts a correct cycle. Upon completion of the check cycle or correct cycle the data is routed back into the lookahead data field with the proper parity bits. The LC tag bit is also set signifying that the required operand is checked and in the proper parity. If a double error occurs or if the correct cycle cannot correct the error, the data is still routed back into lookahead and the LF and LC tag bits are set. The NOOP tag bit and the instruction reject indicator are also set signifying the error condition. During the execution phase of lookahead operation the NOOP tag bit causes the operation to be executed as a no operation. Also the instruction reject indicator causes an interrupt during

the execution of the instruction. The instruction reject indicator is also set if a storage check indication occurs in the memory bus control unit.

The LF and LC tag bits essentially separate the instruction preparation and instruction execution phases of lookahead operation. The setting of the LF and LC tag bits are dependent upon the source of the operand. When the required operand is loaded directly from the instruction unit, the data path passes through the I checker where the necessary checks and parity conversions are made. The LF and LC tag bits are set thereby indicating that the operand is loaded and checked. This action is accomplished during the IAUC loading operation. If there is a store type instruction or an internal fetch request, the LF and LC tag bits are also set during the loading operation. In this particular case the LF and LC tags signify that the operand is not required for execution or that an internal fetch is necessary to receive the operand. Both of the above actions occur during the loading operation and because the LF and LC tags are set during this time no OCC action is necessary. Only when the operand is requested from external storage is the OCC counter action required. This is the only time the operand is not checked when routed to lookahead. The OCC counter then has to route the data through the checker to perform the check and parity conversion. The absence of LF and LC signifies the necessity of OCC action. In all other cases other than external storage fetches, the operand is routed through the checker during the load or the operand is not required and therefore no check is necessary.

Special action is taken by lookahead when an external memory fetch address compares with the address currently setting in the LAAR. Before proceeding further a more thorough understanding of the LAAR is necessary. Associated with the LAAR is a busy trigger which is set whenever an internal fetch or a store type operation is required. The internal register address or the store "to" address of external memory setting in the LAAR are used by lookahead to address the proper internal register or to address the correct storage location during a store operation. When either of the above conditions prevail, the LAAR busy trigger is set and no further setting of the LAAR can be achieved until the store or internal fetch is realized. In cases where an external fetch is required and the operation is not a store or internal fetch the LAAR is set to the memory address if the busy trigger is off. Remember it is the LAAR busy trigger that signifies the contents of the LAAR are necessary for a store or internal fetch. When the address of an external memory fetch (not a store operation) is set into the LAAR, the busy trigger is not set. Because the busy trigger is not set, any other external fetch requests will change the LAAR. The busy trigger is never set unless the operation involved is a store or internal fetch. When the instruction unit determines that an external fetch is required for execution it sends the fetch request and the fetch address to the bus control unit. The bus control unit compares the fetch address with the setting of the LAAR. If there is a comparison between the two addresses, the bus control unit

cancels the fetch request. The comparison of addresses means that the requested data is already in lookahead at another level. Lookahead has provisions for routing the data from one level to another if an address comparison is made on an external operand fetch. This action is called forwarding. Each load operation that sets the LAAR also sets a "from" tag bit into the level tag bit field. This from tag bit indicates that the data field in that level is associated with the operand address setting in the LAAR. When any operand fetch address compares to the LAAR, the fetch request is cancelled and forwarding is set up. The from bit tag designates the level which the required data is forwarded from. The lookahead procedure consists of requesting the I checker to check the data transfer for any errors. When the I checker priority is satisfied, the data is forwarded from the from bit level to the requesting level. The from bit is also set into the new level and the from bit in the original level is reset. By changing the from bit to a newer level, the time possibilities for further forwarding are increased. If the original level is NOOPed the NOOP tag is also forwarded to the new level to indicate that this level has an error condition associated with it. The NOOP tag bit and instruction reject indicator are also set in the new level if the I checker shows an error during the data transfer. The forwarding controls prevent the forwarding action if the original from bit level is a store level. Forwarding is delayed until the SCC counter

takes the storing action. When the store operation is in progress, forwarding is set up.

Forwarding before this time is blocked because of the possibility that the necessary data may not have been received from the execution unit. Only when the storing function is started is lookahead sure that the required data is at the from bit level. If the instruction unit attempts to fetch an operand from storage when lookahead has a store level associated with that address, the fetch must be cancelled until the system is assured that the required data is in storage. An operand fetch request with an address comparison with the LAAR and with the LAAR busy trigger on, tells the instruction unit it is requesting data that has not yet been stored. Hence the fetch is cancelled and forwarding is set up. The actual forwarding operation is accomplished during the SCC storing action. As an example of this action, assume that a store instruction is being processed in lookahead with the store address of 1000. If the instruction unit attempts to fetch the word at address 1000, there is an address comparison with the LAAR. The comparison condition cancels the fetch request. The reason for the fetch cancellation is apparent when you consider that the instruction unit is fetching operands considerably out of the normal program sequence. The fetch is obviously intended to obtain the resulting word of the store instruction which has not yet been completed by lookahead. To prevent another fetch request and additional memory cycles when the word is stored, lookahead sets up the forwarding condition. During the storing cycle, the word is forwarded to the request-

ing level as well as stored in memory location 1000. In operations where there are not any store levels in lookahead (noted by the LAAR busy trigger being off) forwarding occurs during the IAUC loading cycles. By this action all of the memory cycle time is saved. Only when the LAAR busy trigger is on, will the forwarding to delayed until the SCC action.

3.2.00 LOADING

The lookahead loading operation is divided into three distinct areas. These three areas consist of enabling the load to the instruction unit, the actual loading of data and stepping the IAUC counter to the next level. All of these areas are controlled by the IAUC function and are dis cussed in detail below.

3.2.01 Enabling The Load Figure 3.2-1

Load enable signals are signals developed by lookahead and sent to the instruction unit to inform them that there is a free level available for loading. The instruction unit responses to the load enable signals are the actual load pulses. There are three types of loads each signifying a specific type of operation. The three types of loads are identified as type 1, type 2, and type 3, loads.

A type 1 load signifies that the operand required for execution is available directly from the instruction unit. This type of load is used on instructions with immediate indexing. Before a type 1 load can be loaded into lookahead, a type 1 enable signal must be received by the instruction unit from lookahead.

A type 2 load signifies that the operand required for execution must be fetched from external storage. The instruction unit discovers this necessity in the initial decoding of the instruction. The fetch for the operand is made by the instruction unit and the word, when fetched, is routed to the data field in the appropriate lookahead level. If the LAAR busy trigger is not set, the address of the fetched operand is set into the LAAR. The busy trigger is never set when the LAAR is set with a type 2 load. Because the LAAR busy trigger is not set on type 2 loads, successive type 2 loads change the LAAR setting. The reason for setting the LAAR on type 2 loads is for the convenience for forwarding if the conditions are later satisfied. If the LAAR busy trigger is set, the type 2 load is still allowed to occur with the exception of setting the LAAR to the fetch address. A type 2 load can only be loaded from the instruction unit when the lookahead sends a load enable 2 signal to them.

A type 3 load signifies that the current instruction is an internal fetch or store type instruction. Either of these two conditions demand the use of the LAAR. When the LAAR is set to the address of the storage location or internal register, the busy trigger is also set. The busy trigger remains on until the internal fetch or store is complete. A load enable 3 signal is necessary from lookahead before the type 3 can be loaded from the instruction unit.

The three types of lookahead loads are generally classified into two groups. One group requiring the use of the LAAR (type 3) and the other group not requiring the use of the LAAR (type 1 and type 2). If the LAAR is busy, Lookahead cannot enable a type 3 load. If the next instruction from the instruction unit is a type 3 load, the loading process is delayed until the LAAR becomes not busy. However, type 1's and type 2's can still be loaded

if the LAAR is busy and the next instruction ready for loading from the instruction unit is a type 1 or type 2. Only one type 3 load is processed through lookahead at a time. The reason for this is apparent when you realize that there is only one LAAR for the entire lookahead and each type 3 demands the use of the LAAR. Because the LAAR contains only one address at a time, any following type 3 loads must wait until the LAAR is free (busy trigger off).

The load enable communication with the instruction unit is in the form of two enable lines. The decoding conditions for both lines are basically identical. The two lines are identified as "LALD enable 1 or 2" and "LALD enable 3". An added condition exists for the enable 3 line. That condition is the LAAR busy state. As previously mentioned, the LAAR busy trigger being on blocks lookahead from enabling a type 3 load. The LAAR busy trigger is the only factor differentiating the two enable lines. The LAAR busy reset is anticipated by the store data timer E and M lines (SCC Action). The store data timer is only active when a store is completing. Unconditionally the LAAR trigger is reset by the timer. Instead of waiting for the actual reset, the condition is anticapated and an early enable is realized. The prime considerations for enabling any type of load are concerned specifically with the present mode of lookahead operation. When lookahead is in an operation resulting from an interrupt or if forewarding is currently in process all load enable lines to the instruction unit are blocked. In order to allow the enable signals to be sent to the instruction unit only once during a loading operation, the four IAUC counters share a IAUC advance enable sequence (AES) trigger. The AES trigger is turned on each time the IAUC counter ring steps to a new level (initially it is reset on). The load pulses from the instruction unit resets the AES trigger. The AES trigger remains

off until the next IAUC counter is turned on. The AES trigger is the final condition for enabling the load and it insures that the enable signals are sent to the instruction unit only once during any load cycle. The normal conditions then for enabling the loads signals to the instruction unit are illustrated in Figure 3.2-1.

The other conditions illustrated in the Figure are anticipation circuits included to enable the load at the earliest possible time. When the normal conditions for enabling the load are not yet satisfied, but conditions are such that they are imminent, the anticipation controls, by recognizing these conditions enables the load to the instruction unit earlier than normal , thereby realizing a savings in time. By this method the level is loaded as soon as the IAUC step to the next level. The time saving is realized because the enable signals have already been sent to the instruction unit where loading action is initiated. The actual loading of the data cannot be accomplished until the IAUC steps to the new level indicating that the level is actually ready for loading. The anticipation circuits in Figure 3.2-1 are numbered and explained below.

Anticipation of LALD ENABLE 1 & 2

1. This circuit is active during a type 3 load. To allow the fastest loading possible in lookahead, the circuit is conditioned by an actual loading pulse from the instruction unit. During the loading of a type 3 load in one level it becomes desirable, for speed purposes. to enable the load for the next level if possible. The conditions for this circuit illustrates that the normal conditions for enabling any load must prevail plus two additional

conditions indicating that the load may be enabled. These final two conditions show the necessity for no interlock between the SCC and IAUC, indicating that the stepping of the IAUC will not be delayed due to normal counter interlocking. The other condition is that an ECC check on loading is not required for the level currently being loaded. This line is necessary because of the possibility of a correct cycle becoming necessary during the existing ECC load which may delay the IAUC stepping for one cycle in which any correcting action is accomplished. The ECC CK on LD is a special condition necessary for a Transmit/Swap instruction only.

2. The second anticipation circuit is much the same as the first only the conditions for enabling the load were not prevalent when the loading pulses were active. The LD Pulse Men En Adv trigger is substituted for the loading pulse in this case. The LD Pulse Mem En Adv trigger is always turned on during a loading operation to remember that a load operation occurred. Its purpose is to allow the stepping of the IAUC counter and to allow load enable signals to be generated when the proper conditions are met. During the loading operation conditions were such that the IAUC counter was blocked or delayed from stepping. To remember the load, the LD PLS MEM EN ADV Trigger is used to allow the normal stepping and enabling functions to occur after the blocking or delaying conditions no longer prevail. The "NO IAUC AES" line is indicative of the fact that the IAUC

stepping function has not occurred.

3. and 4. These two circuits are essentially identical with the first circuit. The conditions indicate the enabling of 1 and 2 loads during a lookahead 1 or 2 load from the instruction unit. The "NO ECC CK ON LD" line is not necessary because it never occurs during a type 1 or 2 load.

Anticipation of LA LD ENABLE 3

5. and 6. The circuits are identical to circuits 3 and 4 with the additional condition of the LAAR not being busy. If the LAAR is busy, the load enable 3 still may become active. The store data timer E and M indicates that the storing function is almost complete. The LAAR is reset unconditionally from the store data timer so this circuit is anticipating the LAAR busy reset thereby allowing the type 3 to be enabled.

Circuit #7 compares with circuit #2 in that the LD PLS MEM EN ADV trigger is used to remember that a load has occurred. The additional condition placed upon this circuit is the LAAR not busy line or the anticipation of the LAAR not busy as noted by the store data timer E . M condition.

Special enable action is noted in circuits #8 and #9. Under special conditions, the instruction unit in attempting a type 2 load sequence discovers the action to be invalid and terminates the load. However, the IAUC AES trigger is reset and, thus, the load enable signals disappear. The signal type 2 cancel fetch memory allows the level to be reloaded.

Either both of the enable lines or the LD EN 1 or 2 line only are

continually being generated and sent to the instruction unit as new lookahead levels become available. Because the instruction unit is fetching new instructions as fast as possible, the speed at which lookahead is loaded is a prime consideration of the overall speed of instruction preparation.

3.2.02 Loading Figures 3.2-2, 3, 4, 5, 6, 7 and 8

The instruction unit responses to the load enable signals are the actual load pulses. These pulses are identified as LLP1, LLP2, and LLP3. Regardless of the type of load being performed, certain control areas in lookahead are loaded every load cycle. The lookahead operation code field, tag bit field, and indicator field are conditioned for loading during any type of load. The data field in lookahead is loaded directly from the instruction unit during type 1 loads only. The data field on type 2 loads is loaded from external storage or if conditions are right, the data is forwarded from another lookahead level. The data field is not required during a load operation from a type 3 load. The LAAR and its associated busy trigger are set with a type 3 load pulse. The LAAR is also set with each type 2 load if the LAAR busy trigger is off. The IC field in lookahead is set with each single level instruction regardless of the type of load. In multi-level instructions the IC field is set on the final level loaded from the instruction unit. Each of these areas are explained below.

Operation Code Field

With the exception of bit position 8, all operation code bits are set in their respective triggers in the same manner. The data lines from the instruction unit are labeled "set LAOC pos x" (x=0-9 plus parity) and hold up the inputs to their respective triggers. The sample for setting all positions, except position 8, is developed by the load pulse from the instruction unit. The load pulse (LLP1, LLP2 or LLP3) is conditioned by the IAUC counter value designating the loading level. The pulse developed, "GILAi tag bits and op code," samples the data lines (either 1 or 0) into their respective triggers.

The setting of bit position 8 is varied due to special action on some instructions. At times, bit position 8 is made to serve as an auxilliary NOOP bit. The need for an auxiliary NOOP bit is necessary because the NOOP tag bit cannot be set after the last operand level in a VFL instruction. Because the last level of a VFL inst ruction may contain pseudo store information from progressive indexing or recovery information for branch instructions, some method must be used to indicate an error, if the data at the final level failed the I checker test during loading. The "set LAOC position 8" line from the instruction unit conditions the trigger in position 8 of the operation code field. The sample pulse for setting the position 8 trigger on type 2 and type 3 loads is identical to the other bit positions. The IAUC value, designating the level, conditions the type 2 (LLP2) or type 3 (LLP3) load pulse and develops the sample to set bit position 8 direct. When doing type 1 loads that are branch recovery levels or pseudo store levels from progressive indexing, the instruction unit conditions position 8 of the operation code field to act as an auxiliary NOOP bit. The instruction unit conditions the setting circuit of position 8 with the line "No Cond NOOP code on I Parity Error". In all type 1 loads except to two cases mentioned, this line is inactive and allows position 8 to be set in a normal manner. However, when the type 1 load contains progressive indexing or branch recovery information, the normal setting of position 8 is blocked by the instruc - tion unit with the "No Cond NOOP code on I parity error" line. This same line conditions an alternative circuit that is conditioned by the I parity error signal from the I checker. If the data being transferred from the instruction unit fails the I checker test, position 8 of the operation code is set. In later lookahead decoding, position 8 is used as an auxiliary NOOP bit, indicating the data error in the two special recovery levels.

Tag Bit Field

The tag bits that are conditioned only by the instruction unit are the internal (INT), instruction counter (IC), lookahead operation code (LAOP), and word boundary crossover (WBC) tag bits. The level filled (LF), level checked (LC), no operation (NOOP), and the from bit are conditioned by either the instruction unit or lookahead. The disconnect tag is set from the computer maintenance console only. Each of the tag bits is explained below.

Internal Tag Bit - when the instruction unit, in decoding an instruction, finds that the operand required for execution applies to an internal register. The instruction unit conditions the setting of the internal tag bit trigger in lookahead with the line "set internal fetch". The sample to set the trigger is developed by the type of load in progress (LLP1, LLP2, or LLP3). The load pulse, conditioned by the IAUC value designating the level, turns the internal tag trigger either on or off depending upon the status of the instruction unit conditioning line. If the required operand does not apply to the contents of an internal register, the trigger is turned off. If an internal register is required, the conditioning line allows the sample to turn the trigger on. The internal tag is set either on or off (bi-polar) with every load from the instruction unit.

Instruction Counter Tag Bit - indicates that the value stored in the IC field of the lookahead level is valid for interrupt interrogation. The bit is conditioned to set on the final level (this includes single level) of any instruction. When the instruction unit is loading the final or single level of any instruction, the line "set LA IC tag" is active. The sample developed to set the trigger is the load pulse (LLP1, LLP 2, or LLP 3) from the instruction Unit.

The load pulse is conditioned by the IAUC value to insure that the IC bit is set at the correct level. If the level loaded is any level other than the last level or single level the conditioning line for setting the IC bit allows the sample pulse to turn the IC tag bit trigger off.

Lookahead Operation Code (LAOP) - indicates the value store in the lookahead operation code field is for lookahead control purposes only. A good example for setting this bit is in multi-level instructions. The first instruction level contains the real operation code and it is transferred to the execution unit. All succeeding levels pertain to the same instruction, however, the op code field in these levels contains data for lookahead control purposes only. These succeeding levels all have the LAOP tag set. In essence it distinguishes the real op code from lookahead control operation codes. Another example is a pseudo store level. A pseudo store level contains data to be restored to index core storage in the event of an interrupt. There is no real op code in this example as the level is necessary only in the event of an interrupt. The operation code designates the action to be taken by lookahead when an interrupt occurs. The LAOP tag is set to indicate that the Op Code is for lookahead control purposes only. The "set LAOP OP Cod Tag" line from the instruction unit conditions the setting of the LAOP tag bit. The setting sample is developed by the load pulse (LLP1, 2 or 3) and the IAUC value designating the level. The status of the conditioning line determines if the tag bit trigger is turned on or off. If the op code being transferred from the instruction unit is a real op code, the "set LAOP tag bit" line conditions the tag trigger to be reset with the sample pulse.

Word Boundary Crossover Tag (WBC) - the WBC tag is used to signify two separate conditions in lookahead. When used with the LAOP tag being off, it indicates that the operand required for execution of a VFL instruction crosses a work boundary in external storage. The WBC tag is transferred with the other

operation code data to the execution unit. It is always set in the op code level of a VFL instruction when the required operand crosses a word boundary. When the WBC tag is used with LAOP off, it always pertains to a VFL instruction operand. When used with the LAOP tag being on, the WBC tag signifies that the store data buffered at the level had ECC bits generated during the loading process. An example of this decoding is a Transmit/Swap operation where the data being transferred from the instruction unit appears in lookahead in ECC instead of LA parity. To denote this condition to the lookahead controls, the WBC and LAOP tag bits are both set. As in the setting of the other tag bits mentioned previously, the instruction unit conditions the setting of the WBC tag bit when it recognizes any of the two conditions mentioned above. The conditioning line set "LA WBC tag" from the instruction unit conditions the tag bit triggers. The sample developed to set the trigger is the type of load pulse (LLP 1, 2 or 3) and the IAUC value designating the lookahead level. If the WBC is not to be set, the status of the conditioning line is such that the sample pulse turns the trigger off.

Level Filled Tag (LF) - this bit pertains to the data field in lookahead. When on it indicates that the data required at the level is loaded or is not required. The LF tag bit is set from the instruction unit during loading on type 1 and type 3 loads. On type 2 loads the LF setting is a function of the OCC action (refer section 3.3.00). Because the LF setting is so varied, each type of load setting is discussed.

In type 1 loads, the required operand is loaded directly from the instruction unit. The data path for the operand is through the I checker where the I parity check and lookahead parity conversion is made. Unconditionally the type 1 load pulse (LLP 1), conditioned by the IAUC value designating the level, sets the LF tag bit trigger.

On type 2 loads, the required operand is from external storage. In this case, the instruction makes the fetch request to the BCU. Upon honoring the fetch request, the data word is sent from memory and loaded into lookahead. A memory pulse preceding the actual data word sets an LF indication (not the LF tag) in lookahead. The LF indication allows the OCC to function. The LF indication is in the form of an LF E, trigger and the LF tag is the LF M trigger. The next sample pulse following the turn of LF E, turns on the LFM tag trigger. Refer to section 3.3.00 for a detailed description.

There are two variations in setting the LF tag bit on type 3 loads. When a Transmit/Swap instruction is loaded to lookahead, the data is transferred with ECC bits. The lookahead function is to store the words in external storage. Because of this type of transfer the setting condition of the LF tag is dependent upon the ECC error indication from the I checker. If the data transfer through the I checker resulted in an ECC error, the setting of the LF tag is prevented until the correct cycle. As soon as the error indication disappears, the LF tag is set. The conditions then for setting the LF tag bit in this special case are: LLP 3 from the instruction unit, IAUC value designating the level, and NO single I checker error from the I checker. Except for the special case of Transmit/Swap, the LF tag bit is conditioned to set directly from the instruction unit. The conditions for setting the LF tag on a normal type 3 load are: LLP 3 from the instruction unit, IAUC counter value designating the level, and a "NO ECC CK on Ld" signal from the instruction unit. This latter signal signifies that no ECC check is necessary on this type 3 load (no Transmit/Swap). The LF tag, unlike the tag triggers previously mentioned is unipolar. That is, the trigger must have a pulse to set it and another pulse to reset it. The bi-polar settings

are dependent upon the status of the conditioning lines to either turn on or off the tag triggers. Separate conditions are necessary to turn the LF tag trigger on or off. The reset conditions are discussed in section 5.4.00.

Level Checked Tag (LC) - this tag bit indicates that the data at the lookahead level is checked and in the correct parity for execution. The conditions for setting the LC tag bit for type 1 and type 3 loads are identical to the LF tag bit. The same conditioning lines feed both the LF and LC tag bit triggers. On type 2 loads, the LC tag setting is directly related to the OCC function. The LC setting during the OCC function is explained in section 3.3.00. The LC tag and the LF are also set at the completion of a forwarding operation. Forwarding can only occur during type 2 loads. Forwarding is set up when the instruction unit requests an operand from external storage. The fetch request address is compared with the LAAR which contains the latest address of data in lookahead. A comparison signal indicates that the required data is already in lookahead. To save time, lookahead gates the data out of the level containing the required data and forwards it to the level requesting the data. The data at the origin level is not destroyed when it is gated out. The complete forwarding operation is covered in section 3.4.00. The forwarding operation results in setting the LF and LC tag bits at the loading level. When forwarding occurs, no OCC action is necessary and therefore the LF and LC tag are the result of the forwarding operation.

No Operation Tag (NO-OP) - the NO-OP tag bit, when on, indicates that the instruction buffered at the level is to be operated as no operation. In multi-level instructions, the NO-OP bit is not set beyond the last operand level. A NO-OP in any level of a multi-level instruction causes the complete instruction to be operated as a no-operation. A special case is noted for branch recovery levels and recovery level information from progressive indexing by using bit position 8 of the operation code as an auxiliary NOOP bit. This method is required in these two special cases because the NO-OP bit cannot be set after the last operand level. Because the two recovery levels are always the last level, OP position 8 is used for the NOOP indication. Refer to the operation code setting previously explained in this section.

The NO-OP tag bit is set by several different conditions during loading. If the instruction unit, in processing an instruction, discovers something wrong with the instruction it sets the NOOP bit directly during the loading operation. Any data transfers to lookahead through the I checker that result in an error indication, the instruction unit conditions the setting of the NOOP bit. Special conditioning lines are generated by the instruction during special instructions and are used to condition the NOOP bit. The special lines conditioning the NOOP bit are discussed in the instruction unit manual. In this manual the conditioning line is mentioned, but the multiple reasons are not discussed. This section deals with the actual setting of the NOOP bit and only conditions peculiar to lookahead are covered in detail. Also, the NOOP can be set during later lookahead operations, but in this section the setting applies to the loading operation only.

If the instruction unit finds anything wrong about an instruction while they are processing it, a "set LA NOOP" line is sent to lookahead during the loading process. The load pulse (LLP1, 2, or 3) is conditioned by the IAUC value to specify the level at which the NOOP bit is set. This sample, load pulse and IAUC value, is gated by the "set NOOP bit" line from the instruction unit. If the set NOOP line is active from the instruction unit, the NOOP bit trigger is turned on with the sample.

Some instructions require the data transfer from the instruction unit in ECC mode rather than lookahead parity (TSMT/SWAP). In an ECC mode data transfer, a single bit error recognized in the I checker results in an automatic correct cycle. If during the correct cycle, the error proves uncorrectable, the NOOP bit in lookahead is set. In instructions requiring the ECC transfer, the instruction unit indicates the condition to lookahead. An "ECC correct load" signal combines with the IAUC value, designating the level, to condition the NO-OP bit. The actual output condition of the ECC correct load and the IAUC value is called "IB cond NOOP LA i on dbl error". If the ECC correct cycle results in an uncorrectable error, the conditioning line allows the double error indication line to set the NOOP bit.

On instructions where a data fetch (DF) or data store (DS) alarm requires the instruction being loaded to be converted to a no-operation, the instruction unit conditions the lookahead NO-OP setting with the line "condition LA NO-OP from alarm". If the BCU indicates an alarm on this instruction, the alarm signal is gated to lookahead and gated by the conditioning line, turns on the NOOP tag bit.

On normal data loads from the instruction unit, the data is passed through the I checker. The instruction unit signals the checker to check instruction unit parity. If an error occurs during the parity check, the NOOP tag in lookahead is set. On instructions requiring this type of load, the instruction unit conditions the setting of the NOOP bit with "cond NOOP i from ckr par error". This conditioning line gated by the error indication, if any, sets the NOOP tag bit.

There are other methods and conditions that result in setting the NO-OP bit. They are: memory check on type 2 loads, I checker errors from OCC action or forwarding, no-op mode action, houseclean action, forward no-op conversion operations, and divide double stores with a zero divisor. Each of these conditions are explained in the section covering each of the operations.

In general, the NO-OP bit is set anytime before the execution of an instruction when any fault is found in the control information or working data of the instruction.

From Tag Bit - When on in any of the lookahead levels, the from bit signifies that the data stored in the data field is associated with the address in the LAAR. The from bit is used in forwarding only and always designates the level that the data is forwarded from (origin level). Because the from bit applies to the LAAR, only type 2 and type 3 loads are capable of setting it. A type 3 load requires the use of the LAAR and therefore the from bit is always set when a type 3 load occurs. If the LAAR is busy, a type 2 load does not change the contents of the LAAR.

However if the LAAR is not busy, a type 2 load alters the contents of the LAAR and also sets the from bit at the level. The from bit is only set when the LAAR is set in a type 2 load. Successive type 2 loads alter the LAAR and therefore the previous from bit, which applied to the last LAAR setting, is reset and the from bit at the new type 2 level is set (naturally this last action is assuming the LAAR is not busy).

The conditions for setting the from bit on a type 3 load are LLP3 (the actual load pulse) and the IAUC value designating the level. Because there is only one from bit on at a time (only one setting of the LAAR), any from bits at another level are reset. This is accomplished with the LLP3 load pulse gated by no IAUC at the other levels. The IAUC value designates the loading level only and therefore the other 3 lookahead levels are at some other sequence. Because the from bit is set at the IAUC value, the load pulse LLP3 resets the from bit at any of the other levels by gating the reset pulse with not IAUC.

The conditions for setting the from bit during a type 2 load involves more conditions than that for a type 3 load. The from bit is set with a LLP2 load pulse that is conditioned by the IAUC value designating the level, the LAAR busy trigger off, and no address compare. The busy trigger being off is necessary because a type 2 load does not change the LAAR if the busy trigger is on. The no address compare is necessary because an address comparison on a type 2 load results in a forwarding operation. Any from bits on at any other level are reset at the same time the from bit

is set at the IAUC level. The reset conditions are identical to the set condition with the exception of the IAUC value. The from bit set level is designated by the IAUC value and the reset condition is designated by the not IAUC value. This means that any from bit on at levels not being loaded (not IAUC) are reset when the from bit is set in the loading level (IAUC).

The from bit at the IAUC level is reset when a type 1 is loaded from the instruction unit. Since a type 1 load is in no way associated with the LAAR, the from bit is reset during the load.

When the IAUC is conditioned to step into the next level and that level has the NOOP tag and the from tag on, the from bit in that level is reset to prevent the possibility of absolute forwarding. The NOOP bit at the level indicates that the data is in error and therefore the forwarding of error data is prevented.

The from tag set and reset conditions also occur during forwarding operations and they are discussed in section 3.4.00. Logically then, the from bit is set anytime the LAAR is set during a load. When the from bit sets in a level during a load, all other from bits in lookahead are reset. The forwarding operation also alters the from bits during a forwarding operation by resetting the from bit at the origin level and setting the from bit at the destination level.

Indicator Field

The indicator field in lookahead is divided into two groups; the index result indicators and the non-index result indicators. The index result indicators (XF, XCZ, XVLZ, XVC, XVGZ, XL, XE, and XH) are set during every load cycle from the instruction unit. These

indicators are buffered in lookahead and under certain instructions, are transferred, at a later time, to the indicator register. They contain the updated result of any indexing caused by preparation of the current instruction. The indicators are bi-polar; that is their set and reset is conditioned by the status of the input line from the instruction unit. Each indicator is an IRG circuit and requires a (-) minus input and a plus sample to turn the indicator on. With a plus input, the sample pulse turns the indicator off. The status of the input is determined by the instruction unit and the results sent to the input of the individual indicators as "UDI Xi (i indicates individual indicators) to LA IND". The sample is derived from the load pulse from the instruction unit gated by the IAUC value designating the level at which the indicators are set. The sample pulse set each indicator to the status indicated by the input from the instruction unit.

The non-index result indicators (CN1DC, 1R, OP, AD, DS, DF, and IF) are also set into lookahead in every load operation. The setting of the OP, AD, and IF indicators are bi-polar and are conditioned directly from the instruction unit. The sample pulse for setting the OP, AD, and IF indicators is derived from the loading pulse and the IAUC value designating the level. The status of the conditioning line determines the on or off set of the indicators. The other non-index result indicators (CN1DC, 1R, DF, and DS) are unipolar indicators; they require a separate set and reset pulse. Besides being unipolar, they differ from the other indicators in that they are conditioned by

special action discovered during instruction preparation. Because they also differ in their individual setting conditions, each one is considered separately.

The CN1DC indicator is peculiar to the lookahead indicator field. Under special action at a later lookahead time, it ultimately affects setting of the MK indicator in the main indicator register. It is conditioned to set from the instruction unit when any pseudo stores or branch recovery levels are loaded. With the indicator conditioned to set, any I parity error from the I checker during the data transfer sets the indicator on.

The DF indicator is conditioned to set directly from the instruction unit when the data-fetch address is to a permanently protected area of storage (addresses 1, 2, and 3). The conditional input directly from the instruction unit is "set LADF". A sample pulse developed by the load pulse and the IAUC value designating the level, turns the DF indicator on with the "set LADF" line active from the instructionunit. Any data fetches to storage, other than the permanently protected area, results in the instruction unit conditioning the DF indicator set. The sample to set the indicator is a boundary alarm signal from the BCU indicating that the data fetch address is out of bounds.

The DS indicator is conditioned in the same manner as the DF indicator. When the data store address is to a permanently protected area of storage (addresses 1, 2, and 3) the instruction unit conditioning line is "set LADS". A sample developed by the load pulse and the IAUC

value turns the indicator on directly. When the data store address is to any address other than the permanently protected area, the instruction unit conditions the DS indicator to turn on with a boundary alarm signal from BCU. The boundry alarm signal comes only when the data store address fails the boundry comparison set up by the programmer.

The IR indicator is peculiar to the lookahead indicator field. Its output causes the instruction reject indicator (IJ) to be set at a later lookahead time. The setting of the IR indicator in lookahead occurs from several conditions both in the instruction unit and lookahead . Each of these conditions are discussed, but only those peculiar to loading are discussed in detail in this section. References are made to other sections of the manual where the details of the other conditions are noted.

If an instruction fails any instruction unit checks during preparation, the instruction unit conditions the IR indicator bit with the line "set LA IDC" which holds up the set input to the indicator trigger. The sample developed during the loading operation (the load pulse and IAUC value) sets the indicator on. On type 1 loads, the required operand is sent through the I checker during the data transfer. During these loads, the instruction unit conditions the IR indicator to turn on with an I parityerror from the checker. A combination of "cond IR from ckr parity error", from the instruction unit, and "I ckr parity error" from the I checker turns the IR indicator on. The above two conditions signify that there is an error in the instruction itself (first example) or an error occurred during the instruction load to lookahead (second example).

The other conditions that cause the IR indicator to turn on are a memory check during type 2 loads (refer section 3.3.00). forwarding (refer section 3.4.00), operand checking (section 3.3.00), and forward NO-OP conversion (section 3.4.00).

Regardless of the causes for turning on the IR indicator in lookahead, the fact that it is on indicates that the instruction buffered at that level is, in some way, in error. During ABC time the IR indicator is gated out to the main indicator register where, if on, it turns on the instruction reject (IJ) indicator.

The IR indicator in the lookahead indicator field is reset with the same pulse that resets the LC tag. It occurs as the LAUC steps into the level. All other unipolar indicators are reset during ABC time after they are transferred to the main indicator register. The bi-polar indicators are reset by the status of the conditioning line.

Instruction Counter Field

The instruction counter field of lookahead is set during the final level of any instruction load. The address loaded from the instruction unit is one address more than the instruction loaded. For example, if instruction N is being loaded, the IC field in lookahead is set to address N+1,when the final level of the instruction is loaded. The address in the IC field is retained until the final instruction level is acted upon by lookahead. At that time the IC field address is placed in the IC buffer field. It is the address in the IC buffer which is returned to the instruttion unit in the event instruction N interrupts. The IC buffer address is N+1, which

is the instruction that the instruction unit starts processing following any corrective routine for instruction N.

The IC field contains a 19 bit data field and 2 parity bits (bit positions 00-20). The conditioning lines from the instruction unit IC field feed the corresponding bit positions in the lookahead IC field. The bit position trigger are bi-polar and therefore the status of the conditioning lines from the instruction unit determine if the individual triggers are turned on or off. The conditioning line is "1C to LA 1C pos i" and the sample to for the trigger is developed by the load pulse and the IAUC value designating the level.

Data Field

The data field in lookahead is a 76 bit field. Bit positions 00-63 contain the data; positions 64-75 are used for the 10 lookahead parity bits; bit positions 66-73 are used for ECC bits; and bit positions 74 and 75 are reserved for the 2 residue bits. Figure 2.1-3.

The data field is loaded directly from the instruction unit during type 1 loads. The data transfer is from the instruction unit through the I checker into the lookahead data field. The I checker output bus (ICOB) holds up the inputs to the individual triggers in the lookahead data field. Because the checker output consists of lookahead parity and residue, ECC bits, and instruction unit parity bits, the type that is gated into lookahead is dependent upon the communication control from the instruction unit. Generally, the checker output is gated into lookahead in lookahead parity, plus the residue bits. On special instructions, however, the instruction unit signals to lookahead, cause the data to be gated into lookahead in the ECC mode. (TSMT/SWAP).

On type 2 loads, the data field is set from the memory out bus (IMOB). The sample for setting the triggers in this type of load is initiated by a memory select. Type 2 loading is discussed under OCC action in section 3.3.00.

A type 3 load is a store operation or internal fetch. The data field is not set during the loading operation. The data, on type 3 loads is set into lookahead at a later time. The store operations are discussed in section 5.4.00 and the internal fetch operation is discussed in section 5.3.00

The sample developed to set the data register during type 1 loads requiring lookahead parity is conditioned by a load signal from the instruction unit. The sample pulse is sent to bit positions 00-63 where it set the triggers either on or off depending upon the status of the ICOB input line. The setting sample is further conditioned to gate in the parity and residue bits into bits positions 64-73 (parity) and 74 and 75 (residue). If the required data is to be placed in lookahead in ECC mode, the sample pulse conditions bit positions 66-73 to set to the ECC bits also available in ICOB. The setting is either on or off depending upon the status of the ICOB conditioning line. The differentiation to gate in parity or ECC is originated in the instruction unit. and developed in lookahead.

The special operand fields shown in figure 2.1-3 are loaded by type 1 loads. The figure shows the specific data and parity fields used by the special operand fields.

3.2.03 IAUC Counter Control Figure 3.2-9

The stepping of the IAUC counter to the next level occurs after one level is loaded and there is no interlock existing between the IAUC and SCC counters. The IAUC cannot step into a new level until the SCC has left it (refer to section 2.4.00) In some cases, however, provisions are made to anticipate the SCC advance out of a level thereby allowing the IAUC to advance into the level when the interlock is evident.

The actual stepping function is the result of a conditioned advance line, the existing IAUC level value, and the off condition of the IAUC advance enable sequence trigger. The off conditions of the AES signifies that the existing level has sent enable signals to the instruction unit and that the instruction unit has responded by loading the level (the AES trigger is reset with the first half of any load pulse). The existing IAUC value is used to condition the correct stepping circuit to turn on the next IAUC trigger and turn off the existing IAUC trigger. The key to the whole stepping operation is the conditional advance line. The conditional advance line is the result of any interlocks, load pulses and any special cases which may alter the stepping of the IAUC counter. When the advance line is active it signifies that all possible conditions are met and that the IAUC can advance.

In order to remember that a load has occurred, the specific type of load pulse turns on a load pulse memory trigger. The load pulse memory trigger combines with a no IAUC-SCC interlock to produce the conditional advance line. There are variations in turning on the load pulse memory trigger due to the specific type of load being performed. Each type is explained below.

A type 1 load pulse (LLP1) turns on the load pulse memory trigger directly. In this case the load pulse memory trigger remembers that a load to the existing level has occurred. Because a type 2 load requires an external memory fetch, the LLP2 load pulse is prevented from turning on the load pulse memory trigger until lookahead is assured that the BCU has accepted the instruction unit fetch request and that there is no address comparison. The signal that the instruction unit fetch has been honored is the signal "I Accept" from the BCU. The signal indicates that the fetch for the required operand has been initiated in storage and the data, after normal memory cycles, will be sent to lookahead. The no address compare is necessary to indicate that forwarding is not set up (forwarding requires the IAUC value of the existing level to show the origin level of the forwarding operation). The LLP2 load pulse combines with the "I Accept" and the no address compare indications to turn on the load pulse memory trigger. The load pulse memory trigger, conditioned by no IAUC-SCC interlock, allows the conditional advance line to be activated. A load pulse 3 (LLP3) sets the load pulse memory trigger directly if the type 3 load does not require an ECC check of the data load. Normally, data is not loaded during type 3 loads, but when the instruction is TSMT/SWAP, a data load is required in ECC mode. If an ECC check is required, the LLP3 pulse is conditioned by a "no single I checker error line". The turn on of the load pulse memory trigger must be delayed if the ECC check results in an error during the check cycle. The delay is necessary because of the extra two cycles required to accomplish the automatic correct cycle. The LLP3 load pulse turns the load pulse

memory trigger on during the correct cycle, if one is required, or during the check cycle if a single ECC error does not occur. Again the LLP3 turns the load pulse memory trigger on directly if the ECC check is not required. With the load pulse memory trigger on and no IAUC-SCC interlock the conditions are met for the advance line. If during a type 2 load, an address comparison occurred, the LLP2 cannot turn the load pulse memory trigger on. The address compare results in a forwarding operation and the existing IAUC level is necessary for a successful operation. The forwarding operation results in turning the load pulse memory trigger on either during the forward check cycle (no ECC error) or the forward correct cycle (ECC error) .

The load pulse memory trigger is reset after the IAUC steps. The reset is conditioned by the AES trigger being on. The AES is set during the counter advance.

To save time, the LLP1 and the LLP3 (if the LLP3 load does not require an ECC check on load) combines with the NO IAUC-SCC interlock directly to allow the advance line to be active. The LLP2 and the LLP 3(ECC check on load required) can only condition the advance line by turning on the load pulse memory trigger. All of the advance conditions mentioned so far assumed that the IAUC and SCC are not interlocked. Normally, the advance of the IAUC is blocked if the IAUC and SCC are interlocked. The IAUC stepping is blocked to prevent the IAUC from stepping into the same level as the SCC. The IAUC cannot

be at the same level as the SCC because the IAUC is trying to load and the SCC is storing. Both cases utilize the lookahead data field and I checker requests. However, in some cases the SCC advance is anticipated and the IAUC allowed to advance. The SCC anticipation is noted when the LF tag bit remains on during ABC time. The LF tag being on at ABC time indicates that the SCC does not have any action to perform (no store operation) and therefore the IAUC is free to advance. Other anticipations for SCC advance are indicated when the ABC stepping is not delayed (ABC step is delayed only on store operations) and when the ABC and SCC are interlocked (interlock indicates ABC and SCC will step together.) The details of this anticipation conditions are covered in the later sections of the manual. Only a brief description is given here to show the IAUC step is allowed when the IAUC and SCC are interlocked if the SCC advance is anticipated. All of the anticipation circuits mentioned combine with the IAUC-SCC interlock condition to allow the conditional advance line to become active to step the IAUC counter and perform the tag bit and indicator resets.

3.3.00 OPERAND CHECKING Figures 3.3-1, 2, 3, 4, 5

Operand checking is necessary only when the operand required for execution is from external storage (type 2 load). The lookahead action is controlled by the OCC Counter which is allowed to function when the LF and LC indications are not present after a load. As previously mentioned, the LF and LC tag bits are set during the loading of a type 1 and 3 load. A type 2 load requires a fetch from external storage. Because this memory word is received from external storage with ECC bits, the OCC action consists of routing the data to the I Checker along with the necessary control signals to check the ECC bits and generate lookahead parity. These checking and conversion functions are accomplished during a type 1 load when the data passes through the I Checker during the loading process. The action is not necessary during type 3 loads.

A signal, from the bus control unit, arrives just prior to the data and turns on the LF indicator. The LF indication isused to gate the data and check bits to the appropriate lookahead level. If a storage check occurs in the BCU, the NOOP tag bit and instruction reject indicator (LJ) in the level are set. After the data and check bits are loaded, lookahead makes a request to the I Checker priority system (lookahead has 3rd priority for operand checking). When the priority is satisfied an operand check timer is started to gate the data and check bits to the I CHecker. The timer also sends the signals to check ECC and generate LA parity. If no ECC error is encountered during the check cycle, the data and parity bits are routed back into the lookahead data field. If an ECC error occurred during the check cycle, an operand correct timer is started to control the error correct cycle. During the correct cycle the I Checker attempts to correct the error. If the correction is made, the data and parity bits are routed to the lookahead data field with the operand correct timer and no error indication is necessary. However,

if the error proves uncorrectable, the operand correct timer routes the output of the I Checker into the lookahead data field and the NOOP tag bit and instruction reject indicator (IJ) are set. At the completion of the operand check or correct cycle the LC tag bit is also set. With the LF and LC tag bits set, the instruction perparation is complete. Remember that the LF and LC tag bits are set for type 1 and type 3 loads during the loading process. In these two cases, instruction preparation is complete during the IAUC function since no OCC action is necessary. OCC action is only required during type 2 loads to check and convert the operand. OCC action sets the LF and LC tag completing instruction preparation for type 2 loads. A detailed description of the operand check and correct cycles follows.

3.3.01 Data Field Load

The time the lookahead data field is set on a type 2 load is dependent upon the BCU priority system. After the operand fetch request is honored, the word is fetched from storage and set into the memory data bus output register (MDBO). The output of the register is available on IMOB. At the time the data is placed in the MDBO, the BCU sends lookahead one of four memory select pulses selecting the level to receive the operand. The memory select pulse is necessary to give a level fill indication in lookahead. The level fill indication in each of the four levels is in the form of a LFE trigger and its associated LFM trigger. The LFE trigger is turned on with the memory select pulse. The LFE and $\overline{M}$ condition devdlopes the gating in sample for the data from IMOB. All data lines from IMOB feed the inputs to the lookahead data registers. The sampling in pulse, developed from LFE and $\overline{M}$, is only active for the level requiring the data. All 64 data bits and the 8 ECC check bits are gated into lookahead with the gate in sample.

If a storage check indication from the BCU occurs during the operand fetch cycle , the NOOP and instruction reject indicator in the lookahead level are set.

The next clock sample following the turning on of the LFE trigger, allows the LFE trigger to turn on LFM. The LFM trigger is the LF tag bit and indicates that the operand required for execution is loaded in the level. The LF indication is necessary to initiate the operand check cycle.

3.3.02 Operand Check Cycle

After the operand is loaded into lookahead the data and ECC check bits must be routed to the I Checker to check the data transfer for error and to convert the ECC bits to lookahead parity. Before the data can be gated to the I checker, lookahead must request its use and wait until the priority request is honored. The I Checker priority system is listed below

O priority - actual use

1 priority - lookahead store check

2 priority - lookahead forwarding

3 priority - operand checking

4 priority - instruction unit.

The lookahead priority is honored when conditions are such that no 0, no 1, no 2, and 3 are requested. The OCC decoding using the LF indication makes the priority 3 request to the checker. The priority scheme is shown in figure 3.3-4. If the I Checker is not in use or requested by lookahead, the instruction unit priority is always active. The priority to the instruction unit is in the form of an "OF to ICK" trigger which when on gives the I Checker priority to the instruction unit. When another priority request is honored by the I Checker, the "OK to ICK" trigger is reset. The reason the trigger is used is to insure

that the instruction unit is informedof the priority cancellation as quickly as possible. Besides informing the instruction unit of its priority cancellation, the output of the "OK to ICK' trigger is one condition necessary to allow further OCC action. The next OCC action after the priority is satisfied to start the operand check timer. The conditions necessary to start the timer are priority to check granted, a signal that theinstruction unit, priority requests are blocked and a no I checker single error indication from the checker. The reason for the no I checker single error line is to block the turn on of the opd ck timer if the instruction unit is taking a correct cycle. Referring to figure , 3.3-4, , the I checker condition causing the 0 priority to be active is I box check one half, which is an E and not M pulse from an instruction unit control area. This condition prevents the honoring of any other priority requests during the ins truction unit check cycle. If an ECC error occurs during the check cycle the instruction unit must take a correct ccle, at the time the E and not M signal is gone and thepriority system is free to honor any further requests (No 0 priority). There is nothing to block the operand check (3) priority request and the request is granted from the priority system. To prevent any action occurring from lookahead when the instruction unit is taking the correct cycle even though the lookahead priority is satisfied the "No I Ckr single error" line blocks the turn on of the operand check timer. Only after the correct cycle is taken, is the "no I ckr sing error" line active. Gated with the other priority conditions (3 priority granted and "ok to Ick" trigger off) the "NO I ckr single error" line gates the turn on of the operand check timer.

The operand check timer is a standard E and M timer which controls the gating of the data to and from the I checker plus developing the proper signals to send to the checker. The pulse to gate out the data field (bit positions 00-73) is developed with operand check E trigger and the OCC level counter. Besides the gating out function, the operand check timer

E develops a "LA CHK ECC" signal and sends it to the I checker. So far the lookahead data register has been gated out on ICIB to the I checker and the check ECC line generated and sent to the checker.

The next sample pulse, following the setting of the Opd Ck E trigger, turns on the operand check timer M tigger. The M triger develops the sample for gating the data, parity bits, and residue back to lookahead. The output from the I checker (ICOB) holds up the inputs to the lookahead data field in all four levels. The opd ck timer M trigger devlops the pulse for the proper level and samples the data inputs into the data register. A pulse is also developed to gate in the LA parity bits and residue. Assuming no error during the check cycle, the LC tag bit is set at the completion of the operand check cycle. With the LF and LC tag bits on, instruction preparation is complete. However, if an ECC error occurred during the check cycle, the LC tag bit is not set and an operand correct cycle is started.

3.3.03 Operand Correct

The operand correct cycle is controlled by the operand correct timer. The timer consists of an E trigger and an M trigger and provides an additional two clock cycles necessary for the correcting operation. The correct cycle is terminated by setting the LC tag bit and, if the error proves uncorrectable, by setting the NOOP tag and the instruction reject (LJ) indicator at the level.

The operand correct timer duplicates the action of the check timer by routing the data from lookahead to the checker and gating the data from the checker back to lookahead following the correct cycle. Because the correct cycle requires additional I checker time, the timer output duplicates the priority to the I checker by causing the 0 priority to continue

being active. The E trigger gates the data to the checker, E and M keeps the check bits latched on the ICIB input, and the M trigger provides the gate in for the data, parity bits and residue. The M trigger also conditions the setting of the NOOP tag and instruction reject indicator if the checker signals the error is uncorrectable. The LC tag is set by operand correct M. With the LF and LC tag bits set, the instruction preparation is complete.

3.3.04 OCC Stepping.

The OCC Counter stepping is dependent upon the LC tag bit. With the LC tag bit on, the conditions for stepping are allowed providing there is not a counter interlock with the IAUC. Whenever, the interclock is such that both the OCC Counter and IAUC Counter are at the same level, the OCC is blocked from stepping until the IAUC steps. This is necessary to key the counters in the proper sequence (refer to Section 2.4.00).

With a no interlock condition the OCC is allowed to step as soon as the level check (LC) tag bit is on. In type 1 or 3 loads where the LC is set during the loading process, the OCC Counter is conditioned to step during the load. The LC tag being on indicates that the OCC has no action so it steps as soon as the IAUC steps. In type 2 loads, however, the OCC action is necessary and this is evidenced by the absence of the LC tag bit. The OCC step is anticipated during the check cycle and if necessary during the correct cycle. Both anticipation ciruits are controlled by the no interlock conditions. With the interlock condition evident, the stepping is controlled by the normal method.

As with the IAUC, the OCC has a advance Enable Sequence Trigger associated with it. Its purpose is to insure that the OCC functions only once in any particular level.

The AES output is used in starting the OCC action. As soon as the check cycle starts, the AES is reset and does not turn on again until the OCC steps. This insures then that the OCC junctions only once per level. If the OCC does not have any action, the AES remains on and only the counter steps.

Figure 3.3-6 illustrates the stepping action and shows both the normal and anticipation circuits. The normal conditions for stepping are LC, OCC i (i indicating anyone of the four levels), and no interlock. If there is an interlock, the stepping is delayed until the interlock disappears. The operand check anticipation (E and M) is active if there is no ECC error during the check cycle. If the single I checker error is active, the operand check anticipation is blocked. The interlock condition must indicate no interlock to allow any OCC stepping. The operand check can make the anticipation because the end result of the check, assuming no error, is setting the LC tag bit. By this method time can be saved, however, where there is an ECC error, the checking anticipation must be blocked because the OCC requires more action. The operand correct (E and M) anticipation steps the counter, assuming no interlock, during the correct cycle. This action can easily be anticipated because the final action of the correct cycle is setting the LC tag bit. By this time the "No I CKR sing error" is active and the counter stepping occurs. In any of the above anticipation circuits the no interlock line is necessary. If there is a counter interlock during the check or correct cycle the stepping is blocked. As soon as the interlock disappears, the normal stepping is accomplished with the LC tag indication.

The conditioning line causes the existing level trigger (i) to be reset and the next counter to be set (i + 1). The same condition turns on the AES trigger to allow OCC action, if necessary, in the new level.

The disconnect line allows normal stepping of the counter if the level is disconnected from the maintenance console.

3.4.00 FORWARDING

Forwarding, as the name implies, is the ability to move operands from one level to another. It is set up at the time the instruction unit attempts to fetch an operand from external storage. The fetch address is compared to the address setting in the LAAR and any comparison allows forwarding to be initiated. The comparison condition cancels the fetch request and turns on a forwarding required trigger to remember the above conditions.

Forwarding can be accomplished from any of the four lookahead levels depending upon which one has the from bit tag on. The level having the from tag on, implies that the operand located at that level is associated with the address setting in the LAAR. The from tag always identifies the level from which the data is forwarded from. The level that receives the data is identified by the IAUC value. Simply, then, forwarding takes place from the from bit level to the IAUC value level.

The data path for the data being forwarded is through the I checker. Therefore, before forwarding can begin a request must be made to the checker priority system. Once the priority is satisfied the forwarding cycle can be started. The from bit level can have the operand in two forms; lookahead parity or ECC. Ignoring, for the moment, any store levels, the data at the from bit level is always in lookahead parity. In this situation the I checker checks the data for any parity errors. Any error indications result in the IAUC level being NOOPed and the instruction reject indicator being set. When the from bit level is a store level, forwarding is delayed until the storing action is started. This is necessary to insure that the required data is present at the level before allowing forwarding. In a store level where data is to be stored in external storage, it is necessary for the storing action to route the store data through the I checker

to check lookahead parity and convert the parity into the ECC mode. During this store check cycle, the output of the checker is routed back to the store level in ECC and, if forwarding is required, to the IAUC level in lookahead parity. Normally, the from bit is reset in the originating level and set in the destination level to allow further forwarding action when required, However, if the forwarding action occurs during the store check cycle, explained above, the from bit is not reset at the from bit level and the from bit is not set at the destination level. The reason for the procedure is to prevent an error from the I checker if further forwarding occurs before the LAAR is changed. The termination of the store cycle results in a stored executed trigger being set, which participates in the I checker control during forwarding. Whenever the store executed trigger is on, the I checker is conditioned to check ECC during forwarding because the data at the store level is in ECC at this time. If the from bit were transferred to the destination level and forwarding started again after the store check cycle, the data would be routed out of the destination level and sent to the checker in lookahead parity. With the store executed trigger on, however, the checker is controlled to check ECC and an error would ensue during the checker cycle as the parity bits would fail the ECC test. To prevent this possiblity the from bit remains in the store level after forwarding until the LAAR is changed or until another forwarding action occurs. Any forwarding action occurring from the store level, other than the store check cycle, operates in a normal maner.

If the data at any from bit level is NOOP'ed, forwarding can still occur when required. However, the forwarding action results in the NOOP tag and instruction reject (IJ) indicator at the destination level being set. This action is called forward NOOP conversion.

Still another type of forwarding is accomplished by lookahead. Absolute forwarding is when the originating level and the destination level are the same level. An absolute forwarding required trigger is used to remember the condition. Another method is necessary because the priority request to the checker depends upon the LC bit being on in the from bit level. The LC tag is reset when the IAUC advances to a new level and therefore, the normal priority request conditions fail to get priority in this special case. The absolute forwarding required trigger is used as an alternate means of allowing forwarding when the IAUC and the from bit level are identical. An example of absolute forwarding is when the IAUC is loading a level that has just completed an external store. The data still exists in the level after the IAUC advance into the level. If, when loading, the instruction unit makes a fetch address to external storage and it compares to the LAAR, the fetch is cancelled and absolute forwarding is set up. The data is already at the IAUC level in ECC mode (the data is not destroyed when it is gated out to be stored). The forwarding action would consist of routing the data to the I checker with the signal to check ECC. When the check cycle is finished, assuming no error, the data and LA parity bits are routed back into the same level in lookahead. Following is a detailed explanation of the forwarding action.

3.4.01 Forwarding Required Figure 3.4-1

An indication that forwarding is required is in the form of a forwarding required trigger. The forwarding required trigger is turned by an instruction unit LLP2 pulse and an address comparison signal from BCU. The first condition indicates that the instruction unit is loading lookahead with a type 2 load. When the instruction unit makes the fetch for the operand to BCU, the BCU compares the fetch address to the LAAR. If the two address compare, the BCU cancels the fetch request and signals lookahead of the address comparison.

The comparison signal signifies that the requested data is already in lookahead at another level and therefore, the forwarding required trigger is turned on. Before any forwarding can be accomplished, the lookahead must request use of the I checker. In the I checker priority system, forwarding has priority 2. When the forwarding operation is not from a store level containing data in ECC (completed store level), the priority request is decoded by the from bit on, LC tag on, no store execute trigger on, LAAR not busy, and the TBC counter at the level off. These conditions identify the level to be forwarded from (from bit) that the data is available (LC tag), the data is in LA parity (no store execute trigger on) not a store level (LAAR not busy) and the TBC is not functioning at the level (TBC i off). The TBC i trigger off is necessary to insure that the data is not being transferred to two areas simultaneously. Logically, there is no reason why the data could not be transferred to the I checker and to the execution unit at the same time, but because of packaging requirements, the data from a single register is gated to only one destination at a time because there is only a single unit of current available from the gate out circuit on the transistor register trigger used in the operand field. When all conditions are satisfied the 2 priority request is sent to the I checker. If the forwarding action is necessary from a store level that contains the data in ECC mode, the priority request is granted when the LAAR is not busy, the store executed trigger is on, and the forwarding required trigger is on. These conditions identify the level by indicating that the store level data is available)LAAR not busy), the store is completed and data is in ECC (store executed triggeron) and forwarding is required (forwarding required trigger on). These three conditions are sufficient to make the priority request to the checker. Still another type of priority request is necessary when the conditions of forwarding indicate that origin and destination levels for forwarding are the same level. This action is called absolute forwarding and is remembered in an absolute trigger are the same as those necessary for the forwarding required trigger plus the condition that the from bit is at the same level as the IAUC counter. The

reason this trigger is necessary is to make the priority request for the special case because the normal priority circuit fails to qualify. Whenever the IAUC advances into a newlevel, the LC tag bit is reset. With the LC tag off, the normal priority circuit cannot obtain the priority request when absolute forwarding is required. Instead the priority request is made from the absolute forwarding required trigger on and the LAAR not busy.

Any of the conditions just described cause lookahead to make the priority request to the I checker priority system. The conditions for granting priority are similar to that of operand checking only here we ask for second priority instead of the third (refer to Figure 3.3-4). The conditions then are No 0 required, no 1 required and 2 required. When these conditions are satisfied, the priority is granted. The "no ok to I ck trigger" is turned off to block the instruction unit from getting a priority. The output of the I checker priority system is "OK LA 2 Req Fwd cycle". After the priority for forwarding is established, lookahead initiates action to start the data transfer to the checker. The checker cycle is controlledby a forward check timer.

3.4.02 Forward Check Cycle (Figure 3.4-2)

The forwarding check cycle is controlled by the forwarding check timer. This timer consists of an E trigger and an M trigger. The conditions necessary to start the timer are a priority for forwarding from the I checker, and indication that the instruction unit priority requests are blocked, and no ECC error from the checker. If the ECC error line is active from the checker, it indicates that the unit currently using the checker must take a correct cycle. Because of the necessity for a correct cycle, the turning on of the forward check timer is delayed to prevent gating data to the I checker during the correct cycle. When the I checker indicates no single ECC error the forward check timer is turned on.

The forward check timer E trigger and the from bit tag (on in the level the data is to be forwarded from) causd the data reqister bits 00-73 to be gated out to LAICIB. The

forward check E trigger and the status of the store executed trigger determines the signal to the checker for the type of check to perform (parity or ECC) the forward check timer E trigger and the store executed trigger off signal the I checker to checkthe data for parity. If the store executed trigger is on, the forward check timer develops a signal to the I checker to check ECC.

The forward check E trigger conditions the latch input to the M trigger. The forward check M trigger conditions the data gate in from the checker. Because the data is gated into a different level than it was gated out of, the forward check M trigger is conditioned by the IAUC counter value to develop the gate in pulse of the data from the I checker. In all forwarding operations, the data gated into the destination level is in look-ahead parity. Besides bit 00-73 (data and parity), the two residue bits (bits 74 and 75) are gated in the data field.

The next function of the forward check timer is dependent upon the type of check the I checker is performing. Assuming the type of check was a lookahead parity check and assuming no error occurred during the check, the foward check timer M trigger turns on both the LF and LC tag bit in the destination level. Also the from bit at the origin level is reset and the from bit at the destination level is set. All forwarding control triggers are reset and forwarding is complete. If an error occurred during the parity check, the error indication from the checker conditions the turn on of the NOOP tag bit and the instruction reject. The turn on pulse for the NOOP tag and IJ indicator is developed from the foward check timer M trigger and the IAUC counter value. The parity error conditions occur in addition to the other forwarding functions explained above.

If the checker was checking the data for ECC and no error occurred, the data is gated in at the origin level in LA parity, LF and LC tag bits are set, the from bit is reset at the orgin level and set at the destination level, and all forwarding control triggers are reset.

If an ECC eror occurred during the check cycle, lookahead automatically starts a forward correct cycle.

3.4.03 Forward Correct

A forward correct cycle is necessary only during a forwarding operation when the data from the origin level is in ECC mode. The correct cycle is timed by a forward correct timer which consists of an E trigger and an M trigger. The forward correct timer provide the additional two cycles necessary for re-routing the data through the checker. The turn on of the E trigger of the forward correct timer is conditioned by the forward check M trigger. If an ECC error occurrs during the check cycle, the forward correct E trigger is turned' on. The E trigger, conditioned by the from bit, develops the data gate out to the checker. Following the E trigger function, the M trigger is turned on. The M trigger, conditioned by the IAUC counter value, develops the gate in pulse for the data. The data sets into the lookahead data field in lookahead parity. The residue bits are also gated into positions 74 and 75. The completion of the forward correct operation results in the LF and LC tag bits being set, the origin level from bit resetting, the destination level from bit being set and the forwarding control triggers being reset. In addition the NOOP tag bit and IJ indication at the destination level are set if the ECC error proves uncorrectable.

3.4.04 Absolute Forwarding

If the origin level and the destiation level both specify the same lookahead level, the absolute forwarding required is turned on. The absolute forwarding trigger is used to obtain the forwarding priority from the I checker because the normal priority circuit fails in this special case. Refer to section 3.4.01. The normal data path and timers are used. The only difference is in the priority request to the I checker.

3.4.05 Forward NOOP Conversion

An additional complication arises if the data at the origin level is in error or even non-existent as the result of the instruction at the origin level being rejected. The condition is detected if the from tag and the NOOP tag compare at the level (both on). The forwarding action still occurs when forwarding is required, except for any forward correct action. Because a rejected instruction does not result in setting the store executed trigger, it is not possible to request the checker to check ECC; thus no correct cycle is possible. The forwarding action consists of setting the NOOP tag and LJ indicator in the destination level. In addition all from bits are reset. With all the from bits reset, the address comparison system in the BCU is disabled. With the address comparison system disabled, further forwarding action is prevented until the contents of the LAAR change.

4.0.00 INSTRUCTION TRANSFER TO EXECUTION

4.1.00 GENERAL DESCRIPTION

Immediately following the instruction preparation phase of lookahead operation, the lookahead execution phase begins. This section deals with the transfer of the operation code field and operands to a selected execution unit. The selection and transfer of the instruction are accomplished by lookahead. Much of the information in a level is for lookahead decoding or interrupt checking and therefore only the data required for execution is transferred.

The instruction transfer is the responsibility of the Transfer Bus Counter (TBC). The TBC decoding gates out all required fields to the Transfer Out Bus (TOB). All acceptance registers of PAU, SAU, and I/O are conditioned by TOB. The data on TOB are all DC level decoded lines and require a setting pulse to set the information into any particular execution unit. The setting pulse, initiated by a selective pulse from lookahead, is sent to the proper execution unit to place the data into the execution input registers. A transfer bus timer (T timer) times the data transfer operation. An E trigger and M trigger make up the T timer. Combinations of the outputs of these triggers develop the required timing pulses. The key to the transfer operation is the starting of the T timer. TBC decoded lines, determining the required execution unit, start the timer. The timer output signals are sent to the execution unit and effect the setting of the instruction information into the execution unit. A signal from the execution unit informing lookahead that the data is accepted allows the TBC to advance to the next level. Basically, then, the function of the TBC is to decode the instruction, start the T timer, and transfer the required information to the execution unit. Whenever the operation requires an internal fetch or store (internal bit

on at the level) the TBC has no action. With the exception of two internal instruction unit store operations, all internal operations are controlled by the ABC function. With the two exceptions just noted, the TBC function is limited to all operations not requiring an internal fetch or store. The conditions for starting the T timer vary considerably between VFL, FLP, and I/O operations and therefore each type is discussed separately. To insure that the T timer is only started once per level, the TBC has an associated Advance Enable Sequence trigger (AES) to condition the timer turn on. The AES trigger is unconditionally reset by the timer and does not turn on again until the TBC steps to the next level.

All first level FLP instruction, (Figure 4.1-1) not requiring internal operands or store have the same TBC action. Because most FLP instruction are buffered at one level, the TBC transfer action consists of transferring the operation code field and data field (operand) to the PAU. The TBC decoding allows the operation code field in lookahead to be active to the input of the PAU execution register and the lookahead data field to be active to TOB. All of the decoded outputs (operation code field and data field) are dc decoded lines. Further decoding by lookahead determines the instruction to be a floating point instruction not requiring an internal operand. With the instruction identified, the T timer is turned on. The T timer outputs, conditioned by decoded lines identifying the action, send a signal to PAU to gate in the operation code field and start and a signal to gate the operand (data field) into the C register (TOB feeds the C register). If PAU is unable to perform these functions, it must remember the signals and gate the information in when it is able. The TBC advance is conditioned by a signal from PAU indicating to lookahead that it has accepted the information. The arrival of the signal from PAU allow the TBC to advance contingent upon the TBC-OCC interlock. Basically the TBC action for single level floating

point instructions consists of timing the operation code and operand transfer to the PAU execution register and C register respectfully. The maximum number of FLP level requirements are 2. Except for the "multiply and add" (MPYC) instruction, all the second level FLP instructions are store levels and require no TBC action. The decodings is such that the TBC is allowed to step under normal interlock conditions when those level are encountered. The MPYC instruction, however, requires TBC action. The TBC action for the MPYC operand (second operand) consists of a special signal ("FLP continue") from lookahead to PAU and another signal, "GI TOB to C", informing PAU to set the MPYC operand into the C register. A signal from PAU indicating the data has been accepted allows the TBC to step contingent upon the TBC-OCC interlock. With all the information accepted, PAU begins pre-execution. Pre-execution is defined as all data handling possible up to the point where an addressable register must be modified. The modifying of an addressable register is prevented until the system is assured that the previous instruction did not interrupt. Lookahead has a series of tests it performs during ABC time to determine the interrupt or no-interrupt (MAR MODE) condition. For speed purposes, PAU duplicates the lookahead tests to determine the interrupt condition. The PAU test depends upon two lookahead tests to completely interrogate the interrupt mechanism. The lookahead tests for interrupt and no-interrupt (MAR MODE) are covered in detail in the next section of the manual. With the interrupt test duplicated in PAU, the FLP execution does not depend upon lookahead for a signal allowing PAU to modify any addressable register. Any action taken due to an interrupt, however, is completely initiated by lookahead. Interrupt procedures are discussed in section 8.0.00 of the manual.

The function of the TBC for VFL operations, Figures 4.1-2, 3, and 4, are basically the same. The operation code and operand(s) are timed to the execution unit (SAU) by the TBC T timer. However, there is considerable more decoding involved due to the variations in VFL instruction.

Any VFL store levels, internal operand levels, or recovery levels do not require any TBC action. In all other levels, the TBC action consists of transferring the operation code, operands and WBC tag bit to SAU. The first level of any VFL instruction is always the operation code level. Because of the extensive VFL decoding, the TBC employs a TBC "late decode enable" trigger. The function of the LDE trigger is to allow enough time for the decoding circuits to stabilize before starting the T timer. A decoded line identifying the level (operation code level), the LDE trigger and AES trigger start the T timer. Because the VFL operation code level requires additional information, the lookahead data field is used to store the extra data. The full operation code, then, exists between the lookahead operation code field and the lookahead data field. Figure 2.1-3 shows the special operand field used for the VFL operation code level. The T timer times the data transfer such that the lookahead operation code field and the operand field (via TOB) are both gated into the SAU execution register. Because the instruction may indicate the operand involved cross a word boundry, the setting of the WBC tag is also gated to the SAU execution register by the T timer. Besides timing the data transfer to the SAU execution register, the T timer, conditioned by a decoded line identifying the level, turns on a "VFL start" trigger. The VFL start line is sent to SAU and conditions the turn on of the VFL housekeeping trigger. The VFL start trigger remains on until the SAU housekeeping trigger is turned on. As soon as the SAU housekeeping trigger is turned on, a VFL housekeeping line from SAU resets the VFL start trigger in lookahead. If SAU is unable to start at the time lookahead sends the start signal, the VFL housekeeping trigger does not turn on and therefore the VFL start trigger in lookahead remains on keeping the start line active to SAU. The SAU housekeeping mode is the VFL pre-execution. Thus far the first level of a VFL instruction has been transferred. The instruction level requirements

for VFL operation range from a minimum of two to a maximum of six. The first level is always the operation code level and does not include any of the working data (operands) required for execution. The next level of the instruction contains the first (WBC) or only (no WBC) operand. The TBC action at this level consists of gating out the lookahead data field to the C register (via TOB) and timing the data transfer with the T timer. The operand is the only information transferred because the execution unit operation code was transferred during the first level TBC action. The operation code in lookahead at the second level or any succeeding level is for lookahead control purposes only. If the instruction working data crosses a word boundry, the third level of the instruction contains the second operand and the data transfer is identical to that of the second level (except data in gated to the D register via TOB and timed by the T timer). Any succeeding levels are store levels, progressive indexing levels or recovery levels and do not require any TBC action. Basically, the TBC action is identical for both FLP and VFL instructions except that the TBC action required on VFL instructions occurson succeeding levels. After the last operand (only operand with no WBC or second operand with WBC) the T timer also generates a signal for VFL Go. The Go signal to VFL is generated by the T timer, but is conditioned by the fact that the previous instruction did not interrupt (Mar Mode). Unlike PAU, SAU depends entirely upon lookahead for the signal to start execution. When Mar Mode is present in lookahead, the T timer output from the last operand level generates the VFL Go signal and relays it to SAU (Mar Mode is a function of ABC action and is covered in section 5.6.00). The VFL GO signal is the OK signal for SAU to modify addressable registers (execution). The TBC stepping from one instruction level to another, after the operation code level, is conditioned by a decoded line identifying the type of level (only operand, first operand, second operand or MPYC operand) and a T timer output. Special decoded signals are relayed to SAU during a MPYC instruction to allow the special operand to be transferred correctly.

The TBC action on I/O instructions consists of the data field transfer, a decoded operation code line identifying the instruction and a signal selecting the basic exchange or the disc synchronizer. The operation is timed with the T timer. Before the T timer can be started, the counter interlocks are such that a no store operation or interrupt operation is preceeding the I/O instruction. When the TBC and ABC are interlocked it is evident that there is no interrupt from a previous instruction or the ABC could not step into the same level as the TBC. The ABC interlock with the SCC indicates that there is no store preceeding the I/O instruction or the SCC could not have stepped into the same level as the ABC. Once the counters are interlocked, a decoded line identifying the I/O instruction starts the T timer. The T timer outputs conditioned by further decoding cause the select disc synchronizer or exchange pulse to become active. The selective pulse initiates the action in either the exchange or disc synchronizer to gate TOB and the operation code identification into the registers. When the instruction is accepted from lookahead, the I/O unit involved (exchange or disc synchronizer) relays and I/O accept signal back to lookahead. If the I/O unit was busy and unable to accept the instruction an I/O reject signal is relayed back to lookahead. The arrival of either the accept signal or reject signal cause an I/O reaction storage trigger to be turned on. The purpose of the trigger is to establish a timing relationship between lookahead and the I/O unit. The outputs of the trigger resets the T timer and allows the TBC to advance. The next level of the I/O instruction is a dummy level used to interrogate the interrupt mechanism. No TBC action is necessary other than stepping the counter by normal stepping methods.

TBC action is required on instruction unit instructions requiring a store to an internal register. The data path involved in an internal store from the instruction unit is via lookahead (loading process) to the C register (via TOB). The TBC action then is

timing the data transfer from the lookahead data field to the C register. The operation is timed by the T timer. A decoded line identifying the internal instruction unit store plus the MAR mode conditions allows the T timer to be started. The same decoded condition allows the output of the T timer to signal the gate in of the C register from TOB. The TOB lines are dc output lines caused by the instruction unit internal store decoded condition. The TBC advance is allowed by the decoded condition and the T timer output. The data remains in the C register until the ABC starts action. The ABC action causes the data to be routed from the C register to the required internal register. The internal register address is noted by the LAAR.

All the basic TBC actions have just been covered. The remaining areas of this section cover the TBC action in greater detail in respect to PAU, SAU, I/O and instruction unit instructions. Again the important parts to remember are:

1. The TBC function is to select the correct execution unit and relay all the required data to that unit.
2. The key to the TBC action is the T timer and the manner in which it is started.

4.2.00 FLOATING POINT OPERATION Figure 4.2-1

All FLP instructions are executed within PAU. The communication links between lookahead and PAU are the C register for the operands and the PAU execution register for the operation code information. All single or first level FLP instruction not requiring internal operands are transferred by the TBC after instruction preparation is complete. FLP instructions that require internal operands are transferred during ABC time. The FLP instructions which consist of two levels require no TBC action at the second level because they are store levels. The only exception is the MPYC instruction where the second level is a special operand and TBC action is required. Because all FLP instructions are half word instructions, the operation code and operand are both contained within the same lookahead level. The information transfer is accomplished by the transfer bus timer (T timer). The key to the transfer operation is the starting of the T timer. All FLP decoding for TBC action is shown in figure 4.2-2 The TBC decoding causes the dc gate out of the lookahead data field to the C register (via TOB) and the operation code dc gate out to the PAU execution register. Further TBC decoding of the floating point instruction results in starting the T timer to develop the pulses which initiate the setting of the C register and execution register in PAU.

4.2.01 Floating Point Decoding

All floating point instructions are identified by decoding the operation code and tag bit fields in lookahead. The level requirements for floating point instructions are a minimum of one and a maximum of two. With the exception of the second level of a FLP MPYC instruction, all second levels of FLP instructions are store levels. No TBC action is required for FLP store levels so, therefore, a differentiation must be made by the decoding circuits between first level, second level, and MPYC instructions. When internal operands are

required for execution, the TBC action is not required. The decoding circuits then must also differentiate between internal and non-internal operand requirements. There is also the possibility that the information at the level is in error as is noted by the NO-OP tag being on. If that condition is evident the decoders must recognize it and prevent any normal TBC action and cause instruction reject action to be taken. Other considerations necessary for normal decoding are whether lookahead is currently operating in a reject or houseclean mode. If either of the two conditions are prevalent, the normal TBC action cannot be allowed and houseclean or reject action taken instead. The actions taken on reject mode or houseclean mode are covered in sections 7.0.00 and 8.0.00 respectfully. The basic decoding done by the TBC is shown in figure 4.2-2. The second major block shows the basic TBC decoding for floating point instructions. The decoding circuits demand the bit structures as shown and the tag bit designations as shown (1=on, 0 - off). Where a blank position is shown, the bits at those positions do not affect the decoding and can be either 1 or 0.

4.2.02 Data Field and Operation Code Gate Out Figures 4.2-3, 4.2-3A

As soon as the decode line "FLP NOT INT" becomes active in any level of lookahead (normally as soon as LF and LC set) the decoded line gates out the lookahead data field on TOB (bit 00-63 plus parity and residue). The TOB holds up the input to the PAU C register, but does not set the data bits into the register, Because the decoded gate out line is a d.c. line (no pulse) the input to the C register remains active as long as the decoded condition is satisfied. The only way the decoded line can be altered is by changing the TBC value to another level, a signal is required from PAU informing lookahead that the data has been accepted. So until such time as these signals are generated the inputs to the C register remain active. Notice that the gate out is conditioned by the FLP decoding that no

internal operands are required. If an internal operand is required the data field in lookahead is not gated out as it does not contain the required data. The ABC function gates the required operand from the addressed internal register (designated by LAAR) to the C register.

The lookahead operation code field is gated out to the PAU execution register by the decoded line "FLP 1st level". This decoded line conditions the operation code field bits to become active to the input to the PAU execution register (Figure 4.2-8). Again the inputs to the execution register remain active as long as the decoded conditions are satisfied. The execution register is not set by the input from lookahead, but only conditioned to set. As with the C register, a pulse must be generated to set the data into the execution register. Since the gate out of the operation code is not conditioned by a "not internal" condition, the inputs to the execution register are active on any first level floating point instruction regardless of any internal operand requirements. The actual transfer is made during ABC time when internal operands are required, but the inputs to the E register remain active from the TBC decoding during ABC time because the TBC is prevented from stepping until the data is accepted by PAU. In the previous example, the TBC and ABC are at the same level and both are stepped when the data is accepted.

Thus far the lookahead data field is active via TOB to the input to the C register and the operation code field is active to the input to the PAU E register. The next step is to develop the pulse to time the transfer.

4.2.03 Starting the T Timer Figures 4.2-4, 5, 6

The turn on of the T timer is conditioned by a TBC advance enable sequence trigger. The purpose of the AES trigger is to insure that the T timer functions only once during anyone TBC level. The T timer output, once started resets the AES trigger preventing any further conditions from restarting the timer while it remains in the existing level. After the TBC

steps to the next level, the AES is turned back on to allow the timer to function at the new level. Besides the AES conditioning line, the T timer requires a decoded line identifying the need for the timer. In floating point instructions, the decoded line "FLP NOT INT" allows the timer to be started. Again notice that the starting of the timer is prevented if an internal operand is required. The transfer on this type of FLP instruction is a function of the ABC. The T timer is a standard timer consisting of an E trigger and an M trigger. The timer is started by turning on the E trigger. An E trigger output allows the next clock sample to turn the M trigger on. An M trigger output conditions the reset of the E trigger with the sample following the turn on of the M trigger. The M trigger is reset when the timer function is complete. Combinations of the trigger outputs provide the necessary timing pulse to complete the transfer of the instruction to PAU. The T timer generates two signals. One signal tells the PAU to gate in the operation code and start and the other signal is gate in to C. The first signal "FLP ENABLE" is generated by E and not M of the T timer and conditioned by a decoded line (FLP not int) identifying the action. The FP ENABLE signal is relayed to PAU where it initiates the action of setting the operation code information into the C register and also allows PAU to start pre-execution. Because SAU handles the exponent during FLP operation, an identical signal is also sent to SAU to start their action on the FLP instruction. The signal to gate the operand into the C register is conditioned by a decoded line "FLP NOT INT" and generated by T timer E and not M. If PAU is unable to accept the signals, it must remember them and take action as soon as it is able.

With the signals generated and sent to PAU, no further action is taken by lookahead until a signal is received from PAU indicating that it has received the data and started. The signal is a T_0 pulse (PAU clock) and is interpreted in lookahead as FLP first cycle.

The FLP first cycle pulse results in the reset of the T timer and the conditioning of the TBC step to the next level.

4.2.04 Counter Step and T Timer Reset Figure 4.2-7

The actual stepping of the TBC counter to the next level is contingent upon the TBC interlock with the OCC. Assuming that there is no interlock between the OCC and the TBC, the FLP 1st Cycle" pulse, conditioned by no interlock, results in the adv condition for the TBC. The TBC advance condition line is conditioned by the TBC value at the existing level and the AES off condition, (the latter condition ~~ins~~ures that the T timer functioned). The results of the three conditions cause the TBC trigger at the existing level to be reset and the TBC trigger at the next level to be set (~~rest~~ reset TBC i, set TBC i +1). Besides conditioning the counter step circuit, the advance condition line also sets the AES trigger on and resets M of the T timer (E trigger reset with first clock sample following the M trigger turn on). With the TBC stepped to the next level and the AES trigger on, the TBC is set to function at the new level.

If the TBC and OCC are interlocked, the TBC cannot step until the OCC steps (one counter cannot pass another). Because the "FLP 1st cycle" pulse is active from a clock output from PAU, the condition may be passed before the TBC-OCC interlock disappears. To prevent this possiblity, an "Execute First Cycle Mem" trigger is turned on by the "FLP 1st cycle" pulse from PAU. The output of the trigger parallels the "FLP 1st cycle" pulse in the advance condition circuit. If the first cycle pulse disappears before the interlock between the OCC and TBC disappears, the "execute first cycle memory" trigger causes the advance condition to become active after the interlock disappears. The advance condition, once active, accomplishes the functions previously described. The execute first cycle memory trigger is reset when the TBC AES is turned on.

The next lookahead action is the function of the ABC and is covered in the next section of this manual. The area just covered shows the TBC action only for floating point

operations not involving a second level store or internal operands.

4.2.05 Special TBC Action on FLP MPYC

If the FLP instruction being transferred is a multiply and add instruction (MPYC) the TBC has special action. The multiply and add instruction requires 2 lookahead levels to complete the data transfer. The first level of the instruction contains the operation code and first operand. The TBC action for the first level of the instruction is identical to that just described. The MPYC instruction is the only second level FLP instruction that TBC action is required. The second level of the instruction contains the special operand (implied contents of address 14) required by PAU. To transfer the special operand to PAU again involves gating out the data field to the C register and developing the required signals to time the transfer. The timing of the transfer is accomplished by the T timer. The acceptance of the special operand by PAU, causes PAU to relay an acceptance signal to lookahead allowing the normal advance functions to occur.

Because the first level transfer is the same as any first level FLP instruction not requiring internal operands, this section covers the second level transfer only. The TBC action at the second level of the instruction consists of gating the lookahead data field to the C register and timing the transfer with the T timer. Because the operation code at the second level is for lookahead control purposes only, the field is prevented from being gated to PAU. The real operation code necessary for the execution of the instruction is transferred during the first level TBC action. Remember that the operation code gate out at the first level was caused by the decoded line "FLP 1st level". Because the TBC is now functioning at the second level of the FLP instruction the gate out line is not active for the operation code field. The data field in the second level is gated out by a special decoded line "transfer bus Gate Out" which is the result of the MPYC operation code

decoding at the second level. Referring to the TBC decoding diagram figure 4.2-2 a decoded line called "FLP continue" is decoded from the operation code and tag bit fields. The same bit structure allows "cond TB timer" and "cond TB gate out" to be decoded. All three of these decoded lines are used to transfer the second level of the MPYC instruction to PAU.

The data field in lookahead is gated out when the "cond TB gate out" becomes active. The actual gate out condition is a result of the "TB gate out" line, TBC counter value, an a late decode enable trigger (LDE). The TBC value designates the level to be gated out and the LDE trigger is necessary to allow sufficient time for all the decoded lines to become stable before action is taken. The LDE trigger is turned on when instruction preparation is complete (LF and LC) and INSURES that forwarding is not in process or lookahead is not in a houseclean mode.

As soon as the three conditions are satisfied, the data field is gated out to the C register, via TOB, and holds up the inputs to the register . The decode line "cond TB timer", conditioned by the LDE and AES triggers, turns on the T timer to initiate the transfer function. The E and not M output of the timer, conditioned by the decoded line "FL Pt cont" is relayed to PAU as "MPYC cont". The arrival of the signal in PAU causes PAU to set the C register according to the inputs active from TOB. The special operand has now been transferred. After the acceptance PAU generates a signal "cumulative multiplier accepted" and relays it back to lookahead. This signal accomplishes the same function as the "FLP first cycle" pulse did in the first level. That is, allows the TBC advance conditionto become active which allows the TBC to advance, set the AES on, reset the T timer and reset the LDE trigger. If there is an interlock and the acceptance pulse from PAU cannot allow the advance condition immediately, the execute first cycle trigger is turned on to remember the condition. The execute first cycle trigger replaces the acceptance pulse in the advance condition when the TBC-OCC interlock disappears.

4.3.00 VARIABLE FIELD LENGTH (VFL) OPERATION

The restrictions to TBC action for VFL instructions are the same as for FLP instructions plus two additional situations. That is, TBC action does not take place for any VFL instruction levels that require internal operands, store levels, branch recovery levels or progressive indexing levels (pseudo stores). TBC action is required on all VFL operation code levels and all operand levels whose source was xcs or external storage. Referring to section 2.2.00 relating to lookahead and instruction level requirements, it states that VFL instructions require a minimum of two to a maximum of six levels for any one VFL instruction. The only levels that require TBC action on any VFL instruction are the operation code level (always first level) and the operand levels (if not internal). When there is not a word boundry crossover the second level is the only operand level and the last TBC action occurs at that level. If there is a WBC, then the second and third levels require TBC action (first operand INT and second operand INT). If anyone of the two operands requires an internal operand the TBC does not function for that level. Only when the level does not designate an internal operand, a store, a recovery level or progressive indexing level does the TBC function. In the following discussion of VFL TBC action only those levels which require TBC action are discussed. Keep in mind that these levels do not include internal operand requirements, store levels, progressive indexing levels of recovery levels (these levels are a function of ABC action and are discussed in section 5.0.00).

The first level of any VFL instruction is always the operation code level. Because a VFL instruction requires much more operational data than a FLP instruction, the lookahead data field as well as the operation code field is used to buffer the instruction. Because the data field is used to buffer the additional VFL information for the operation code, any operands required by the instruction must be buffered at succeeding levels (1 level for each operand - 2 operands maximum). The TBC action then on a VFL operation code level involves a transfer of both the operation code field and the lookahead data field to the SAU execution (E) register.

Because no operands are available at the first level no data is gated to the C register. A decoded "VFL operation code level" line conditions the gate out of the operation code field, data field (via TOB) and the status of the WBC tag bit to the SAU execution register. Further VFL decoding starts the T timer whose output, conditioned by a decoded line identifying the action, turns on the VFL start trigger. The output of the VFL start trigger is relayed to SAU and attempts to turn on the VFL housekeeping trigger and set the SAU execution register. If SAU is busy, the turn on of the VFL housekeeping trigger and the setting of the execution register is blocked. The reset of the VFL start trigger in lookahead is accomplished by the "on" condition of the VFL housekeeping trigger in SAU. Therefore if SAU is busy and the housekeeping trigger cannot turn on, the VFL start trigger is lookahead is not reset. With the VFL start trigger remaining on, the VFL start line remains active to SAU until such time as SAU becomes not busy and allows the start line to turn the housekeeping trigger on and set the operation code information into the execution register. When the VFL housekeeping indication is relayed to lookahead the VFL start trigger is reset and the TBC is allowed to advance to the next lookahead level. The TBC advance condition also resets the T timer.

The next lookahead level contains either the only operand ($\overline{\text{WBC}}$) or the first of two operands (WBC). The TBC action at the first or only operand level results in decoding the condition (first or only and not internal) and gating the lookahead data field (via TOB) to the inputs of the C register. The operation code at this level is for lookahead purposes only (as noted by the LAOP tag being set) and is not gated out to SAU. Only the data field (operand) is gated to SAU during VFL operand levels. The operation is timed by starting the T timer which develops the necessary pulses to be relayed to SAU to gate the data into the C register. Besides the decoded line, the T timer is conditioned by the fact that the previous instruction did not interrupt (MAR MODE). The MAR MODE is necessary because SAU does not duplicate the interrupt test equipment and relys entirely upon lookahead

for assurance that it may modify an addressable register. If the decoded level indicated that it was the only operand level, the T timer, conditioned by MAR MODE, turns on a VFL GO trigger. The GO Trigger output is sent to SAU and interpreted to mean that the previous instruction completed without error, that all the operands required for execution have been sent by lookahead and OK to execute. If the decoded condition indicated that the level contained the first operand level (implying that another operand is in the next level,) the VFL GO trigger is not turned on. The only TBC action required on the VFL first operand level is the transfer of the data to the C register. The only operand level turns on the VFL GO trigger in addition to the data transfer. A decoded line identifying either first or only operand conditions a T timer output to step the TBC to the next lookahead level.

The action of the TBC at the next level of the instruction is determined by the TBC decoding. The TBC only has action if the level is decoded as having the second operand (WBC) not internal. Any other decoding (store, recovery or progressive indexing level) does not require TBC action other than allowing it to step contingent upon the interlock with the OCC. A decoded level indicating the second operand causes the data field to be gated to the input of the D register (first operand was placed in the C register in the last level). The decoded condition starts the T timer to time the data transfer. The T timer output also turns on the VFL GO trigger. The GO trigger indication is sent to SAU to allow them to start execution. The previous actions are conditioned by MAR mode which remains active until again tested at the end of the present instruction. Again the T timer output allows the TBC advance to occur in the same manner as mentioned previously. All remaining levels of the VFL instruction, if any, do not require any TBC action other than the decoding conditioning the TBC advance directly.

To summarize the action very briefly; all data transfers are timed by the T timer. The VFL start signal is generated at the VFL operation code level and the VFL Go signal is generated at the last operand transfer level. All data transfer other than the operation code level are contingent upon MAR MODE being present within lookahead. All internal operand transfers do not require TBC action - they are transferred during ABC action. The VFL operation code level utilizes the data field at the level and is transferred to the SAU execution register along with the operation code field and WBC tag. The presence of the LAOP tag in all levels other than the operation code level, prevent the lookahead operation code field from being transferred. The first or only operand is transferred to the C register and the second operand, if required (WBC tag on) is transferred to the D register.

The branch on bit (BB) and the branch on indicator (BI), although not VFL instructions, are both investigated by SAU. These two branch instructions are influenced by the status of an addressed bit location in any data word (BB) or indicator address in the indicator register word. Because SAU is the only unit which can inspect individual bits or groups of bits in a word, the branch instructions (BB and BI) are interrogated by that unit. Figure 2.1-2 and 2.1-3 show the operation code field and special operand field layout in the BB and BI operation code levels. Unlike a VFL instruction where the operation code is transferred to SAU to determine the instruction action (add, sub ect), the branch on bit or branch on indicator is entirely decoded by lookahead and one decoded line only (BB or BI) is sent to SAU. Figure 2.1-2 shows the operation code field as it is loaded from the instruction unit. The conditional indicators shown apply to the "branch to" address and are investigated by SAU to determine the reliability of the branch condition. The data field shown in figure 2.1-3 shows the bit address (BB) or indicator address (BI) and the control bits used to determine the status of the bit after interrogation. The TBC action at the operation code level consists

of gating out the operation code field, data field, WBC tag and the decoded line identifying the BB or BI condition. The decoded line identifying the action (BB or BI) starts the T timer to time the transfer to the SAU execution register. The operand involved for the BB instruction is transferred in a normal manner (VFL only operand $\overline{\text{int}}$). If the operand required is internal, the TBC has no action. The operand for the branch on indicator instruction is the indicator register. Because the indicator register is an internal register, the TBC has no action at the operand level of the BI instruction.

4.3.01 VFL Decoding

Figure 4.2-2 shows the TBC decoding for VFL type instructions. Because branch on bit and branch on indicator instructions are also sent to SAU, their decoding is discussed here and is also shown in figure 4.2-2. The VFL decoding all include a bit in position 9 of the operation code field. The VFL operation code is strictly a bit in position 9, no LAOP tag and no instruction reject decoding (NO-OP). All VFL decoding during TBC time, except the operation code, have the LAOP tag bit set indicating that the operation code at that level is for lookahead use only. All VFL decoding is determined by combinations of the lookahead operation code field and tag bits. The decoding involved for TBC VFL decoding consists of operation code level, only operand, first operand, second operand, store level, and multiply and add (MPYC). If the internal bit is on at the level it designates that the required operand is located in an internal register. Internal register fetches are the responsibility of the ABC function. Therefore, the VFL decoding of VFL operands is conditioned by the internal tag bit being off which designates TBC action. The operation code bit structure and tag bit designations cause further decoded control lines to become active to condition the T timer and gate out functions. These other decoded lines are the results of the same bit structure identifying the VFL level. Figure 4.2-2 shows that the VFL identification decoding also causes certain of the special decoded lines (lower part of figure) to become active. For example the VFL op code decoding also cause "cond TB timer" to be active.

The branch on bit and branch on indicator decoding is shown also in figure 4.2-2. Notice that one of the necessary decoding conditions for these two type of instructions is the presence of the LAOP tag bit. The LAOP bit designates that the operation code field is for lookahead purposes only. Lookahead decodes the bits shown to find the branch on bit or indicator condition. Instead of sending the operation code field to SAU for decoding, Lookahead decodes the instruction and sends the decoded results to SAU. The decoded branch condition also pariticpates in the gate out of the operation code field. to SAU as well as the data field. Normally, anytime the LAOP tag is on, the operation code field is not gated out. In the two branch instruction, the operation code field also buffers the conditional indicators which pertain strictly to the "branch to" address. Because of the added data in the operation code field, the decoded branch condition line allows the operation code field to be gated out in order that SAU receives the conditional indicators. SAU looks at the status of the conditional indicators to determine the success of the branch instruction. Figure 2.1-2 shows the operation code field layout for the branch instructions. Notice that the area that the indicators are buffered in no way affect the positions lookahead uses to decode the branch condition. In the operation code field transfer, only the conditional indicators are gated into SAU. The branch on indicator test level, shown in figure 2.1-3, is a special format created by the instruction unit to allow SAU to reset an indicator that caused an interrupt. The level is generated by the instruction unit during interrupt action and is discussed in section 8.0.00. The other two decoded branch levels are special decoding used to allow the TBC to advance over a branch recovery level in the event the branch instruction was not successful. The recovery level is required only when the branch instruction associated with it is successful.

All TBC decoding is conditioned by the TBC counter value, designating the level at which the action occurs and no reject action indicating the data at the level is free from error.

4.3.02 T Timer and Information Transfer

This section discusses the starting of the T timer and data transfer for VFL operation code levels, only operand levels, first operand levels and second operand levels. Included also is the starting of the timer for the branch on bit and branch on indicator instructions. All operand decoding used to start the timer are assumed to be non-internal because any internal operand requirements are discussed in section 5.0.00.

VFL Operation Code Level Figure 4.3-1

The decoded line identifying the VFL operation code also causes "cond operation code gate out" to become active. The "OP CODE GO" line, conditioned by the TBC value and the LDE trigger, gates out the operation code field to the SAU execution register. The bit positions hold up inputs to SAU execution register as long as the decoded condition remains. Figure 4.3-5 shows the relationship between the lookahead operation code field and the SAU execution register positions. Because the lookahead data field also contains instruction control data at the VFL operation code level, it too is gated out to the SAU execution register. The VFL Op code decoding also cause "cond TB (transfer bus) gate out" to become active. This decoded line, conditioned by the LDE trigger and TBC value, causes the data field in lookahead to hold up the inputs to the execution register via TOB. Because the required operand may have crossed a word boundry, the WBC crossover tag bit is also gated out and holds up the input bit position 0 of the SAU execution register. All the inputs to the execution register from lookahead are d.c. gate out lines and they remain active as long as the gate out decoded condition exists (until the TBC step to the next level). To time the operation code level transfer, the T timer is started. Figures 4.2-4 and 4.2-5.

The decoded line "TBC dec cond TB timer" (same decoding as VFL op code level), conditioned by the TBC AES and the TBC LDE trigger allows the T timer to start with the next

sample pulse following the active output of the turn on condition. As soon as the T timer comes on it resets the TBC AES trigger to insure that the TBC will function only once at the level. The T timer output E and not M is conditioned by a "VFL EN START" line to turn on the VFL start trigger. The "VFL EN START" line is conditioned directly from the decoded VFL Op code level" line identifying the level. The output of the VFL start trigger is sent to SAU where it attempts to turn on the VFL housekeeping trigger and affect the setting of the SAU execution register. If SAU is not busy, the VFL start signal from lookahead sets the VFL housekeeping trigger and allows the setting of the execution register (inputs still active from lookahead). If SAU is busy, the VFL start signal from lookahead cannot do anything and remains active until SAU becomes not busy. When the SAU housekeeping trigger is on, it generates a signal "SAU Housekeeping" and relays it to lookahead to inform them that the data has been accepted. The arrival of the SAU housekeeping signal resets the T timer and conditions the TBC to advance to the next level. The SAU housekeeping trigger indicates that pre-execution has started in SAU.

Only Operand Level Figure 4.3-2

The decoding of the only operand level is shown in figure 4.2-2. The TBC action at the level is basically the same as the TBC action at any level. That is, the decoding identifying the level conditions the gate out of the lookahead data field to the C register. With the inputs active to the C register, the decoded condition starts the T timer to time the transfer operation. The presence of the LAOP tag at the level justifies the blocking of the operation code field transfer. Remember the previous level was the operation code level and all control data was transferred at that time. SAU is waiting for the working data(operand). Because this level was decoded as the only operand level, lookahead generates a "Go" signal to SAU at the completion of the data transfer. The Go signal is the OK signal to execute the instruction. The execution is allowed because the "GO" signal in lookahead is conditioned

by MAR MODE indicating the previous instruction completed without interrupting. SAU is already in a housekeeping mode (from previous level) and therefore the full TBC action required at this level is the operand transfer and Go signal generation.

The decoded line identifying the level gates out the data field (via TOB) to the C register in the normal manner (d.c. gate out with decoded line conditioned by LDE trigger and AES. Remember AES was reset at the last level during the transfer but when the TBC stepped into this level it was turned on again). The inputs to the C register remain active as long as the decoded gate condition exists. The "VFL only operand" decoded line causes "cond TB timer on MAR" to become active to condition the turn on of the T timer. (Figures 4-2-4 and 5) Before the timer can be started however, lookahead must be assured that the previous instruction did not interrupt. This assurance is in the form of MAR MODE which is one of the prime requirements for starting the timer at this level. Together with MAR MODE, AES, and LDE the decoded line conditions the timer to start with the first sample pulse following the satisfaction of the turn on requirements. The E and not M timer output together with the decoded line "cond GI to C Reg" (caused from VFL only operand decoding) generate a signal to PAU. The signal sets the C register to the inputs setting on TOB. Another decoded line "Cond SAU MAR" (generated from the only operand decoding) together with the E and not M output of the timer combine to turn on the VFL GO trigger in lookahead. The GO signal is relayed to SAU where it initiates the execution action. As soon as SAU starts executing, the SAU housekeeping trigger is reset. The reset of the housekeeping trigger is relayed back to lookahead where the Go trigger is reset. The decode "VFL only operand" line allows the TBC to advance when, contingent upon the interlock conditions, with the M trigger output of the timer. The advance condition also allows the timer to be reset.

First Operand Figure 4.3-2

The TBC action for the decoding of the first operand is identical to that just described for the only operand decoding except that the Go trigger is not turned on. The decoded name implies that the level contains the first operand thereby indicating that there are two operands involved in the instruction. The GO trigger is a functionof the last operand transfer and therefore it is not turned on at the first operand level. The TBC action is merely a gate out of the data field to the C register and starting the T timer to time the transfer. The decoded line "VFL 1st Operand" replaces the "VFL only operand" participation for the TBC action at this level.

VFL Second Operand Figure 4.3-3

The function of the TBC at the second operand level consists of the lookahead data field transfer to the D register (first operand is in the C register) and the turn on of the VFL GO trigger (last operand transfer) to all SAU to start execution. The TBC action at this level is identical to the only operand level except that the "VFL 2nd operand" decoding replaces the "only operand" decoding in the action. Also the gate out of the data field is conditioned to the D register instead of the C register (first operand is in the C register). The second operand level also sets the Go trigger because this operand is the last operand transfer. All control and functions are the same as the only operand level except the data is gated to the D register. The TBC step in the normal manner by conditioning the advance with the M trigger output of the timer. If the following levels pertain to the same instruction, they are either store levels or recovery levels and thereby do not require any TBC action other than stepping the counter.

(Figure 4.3-4)

The MPYC instruction requires special decoding and signal generation at the MPYC operand level. The MPYC operand level is the last operand transfer and is the only level requiring the special action. The operation code level and the other operand levels are handled in the same manner already described. The TBC action at tht MPYC operand level consists of the data field transfer to the C register. Before lookahead can time the data transfer to the C register, it must be assured that the C register is free. An earlier operand was placed in the C register and the "go" signal generated during the last normal operand transfer. The go signal allows SAU to start executing, but it realizes it hasn't received the special operand yet and therefore must make the C register available to receive it. The SAU action moves the contents of the C register to the F register. When the C register is empty SAU relays a signal to lookahead informing them of the action. The signal "SAU ready for MPYC operand" turns on a "VFL wait for MPYC operand" trigger in lookahead. The output of the trigger, together with a decoded line identifying the level (MPYC), the AES trigger and LDE trigger allow the T timer to start (Figure 4.2-4 and 5). The output of the data field is active from the normal decoded gate out condition and therefore the inputs are active to the C register. The T timer output and the MPYC decoded condition develop the gate in signal for the C register. Besides gating the MPYC operand to the C register, a signal is also developed and sent to the A checker to gate the residue of the MPYC operand into the MPYC initial residue register. Thus the special MPYC operand has been transferred. The T timer resets the VFL wait for MPYC operand" memory trigger. The MPYC decoding conditions the TBC to advance contingent upon the usual advance interlocks.

The branch on bit and branch on indicator operation code levels are basically operated upon in the same manner as any VFL operation code level. The "BB op code" or "BI op code" decoding replaces the "VFL op code" decoding in the TBC action for either

these two branch instructions. Because the operation code field contains the conditional indicators, the normal decoding of an instruction in SAU cannot occur. SAU only has provisions for decoding VFL instruction. To determine the instruction, lookahead decodes the operation code field. The result of the branch decoding is one line (BB op cd or BI op cd). The decoded line is sent to the execution register with the data field and operation code field. The timer is started as in a normal VFL op code level (only using the BB or BI decoding) and the operation to the E register is timed. The TBC advance occurs in the normal manner.

If the operand involved with the BB instruction, is not internal, the resulting decoding and action is identical to the only operand level. The BI operand is the indicator register which is an internal register and thereby no TBC action is required. Figure 4.3-5 shows the bit position relationship in LA and SAU for the BB and BI instruction.

4.3.03 TBC Stepping Controls Figure 4.2-7

The prime condition allowing the TBC to advance is the TBC interlock with the OCC. If the OCC is at the same level as the OCC the TBC cannot advance (one counter cannot pass another). The advance condition for the TBC must be remembered if the interlock condition is present when the advance condition is active. The advance condition for the TBC at the VFL operation code level is from the VFL housekeeping signal. The housekeeping signal signifies that SAU has accepted the operation code data and started pre-execution. The housekeeping signal and the output of the VFL start trigger combine to give the VFL first cycle line. This line is actually a pulse because the VFL start trigger is reset with the first sample following the arrival of the housekeeping signal from SAU. The VFL first cycle, conditioned by NO TBC-OCC interlock allows the TBC advance conditionto become active. All VFL levels that require TBC action cause a control line "adv on TB timer M" to become active. The decoded line combines with the M trigger output of the T timer. The output

of these two conditions are further conditioned by the no interlock indication to provide the TBC advance control line. The TBC advance line when active causes the present level TBC trigger to be reset and the next trigger set. Also the TBC AES is turned on again to provide the necessary condition of allowing the TBC to function at the new level. Finally, the advance also resets the late decode enable trigger. The LDE trigger turn on is conditioned by instruction preparation being complete at the new level. When the AES trigger comes on at the new level the output allows the next sample pulse to reset the T timer. Thus TBC action at one level is fully complete and set up at the new level.

If there is a TBC-OCC counter interlock, the TBC stepping is blocked until the interlock disappears. The stepping condition at the VFL operation code level is the VFL first cycle pulse. If the stepping is blocked because of the interlock, the VFL first cycle pulse is remembered. The pulse is remembered in the execute first cycle memory trigger, which the pulse turns on. The first cycle memory trigger stays on until the stepping of the TBC occurs. The first cycle memory trigger output replaces the VFL first cycle pulse in the stepping activity. As soon as the TBC-OCC interlock disappears (OCC steps) the execute first cycle memory trigger allows the TBC advance condition to become active. The advance condition resets the existing TBC trigger, sets the next TBC trigger on, sets the AES trigger on and resets the LDE trigger. When the AES trigger comes on it allows the next sample, following the setting of the AES, to reset the execute first cycle memory trigger and to reset the T timer. The TBC function is complete for level i and set up for level i + 1. The stepping for the other VFL levels is the same as previously described with no interlock. The stepping for other VFL instruction levels is a function of M of the T timer and the decoded line identifying the level. The M trigger of the timer and the decoding both remain active until the counter steps so therefore if the interlock prevents the TBC stepping initially, the inputs to the stepping function remain active. The M trigger output remains active because the reset of the timer occurs after the

TBC steps. If the TBC does not step the M trigger of the timer remains on. The decoded line is a function of the operation code field and the TBC value. As long as the TBC does not step the bits of the operation code field are decoded and remain active as long as the TBC value remains the same. After the interlock disappears and the counter steps the new TBC value applies to the operation code field at the new level. The stepping of the TBC counter when the TBC at the level had no action is accomplished by the ABC action. When the TBC at a particular level has no action, the ABC starts it action at the level. The TBC and the ABC are then interlocked at the same level. When the ABC action is complete at the level, it attempts to step to the next level, but it cannot pass the TBC counter. Therefore the ABC advance condition together with the TBC-ABC interlock combine to allow the TBC to advance which allows the ABC to advance. However if the TBC is also interlocked with the OCC, both the TBC and ABC are prevented from stepping until the OCC steps. As soon as the OCC steps, the TBC can step which allows the ABC to step. Thus, the counters are kept in synchronism.

4.4.00 NON ARITHMETIC INSTRUCTIONS

The non arithmetic instructions include the input-output (I/O) and instruction unit instructions. All input-output instructions require two levels of lookahead. The first level of the instruction is the operating level while the second level is a dummy level to allow lookahead to reliably interrogate the interrupt mechanism. The TBC action at the first level consists of decoding the instruction for either the exchange or disc synchronizer, gating a decoded line identifying the action, gating the lookahead data field to the input of the selected unit and starting the T timer to time the transfer. The exchange or disc synchronizer must inform lookahead of an acceptance of the data or a rejection. The return indication turns on a trigger in lookahead to establish a timing relationship between the I/O unit and lookahead. The output of the trigger is an I/O response and conditions the TBC advance.

Most of the instruction unit instructions are completely executed by the instruction unit. However, any instruction unit instructionsthat require a store to either an external or internal address are loaded into lookahead for the store operation. An instruction unit store to an external storage address does not require any TBC action. Stores to an internal register however involve a data transfer to the C register. Once the data is in the C register, the data is routed to the internal register designated by the LAAR. The transfer from the C register to the internal register is a function of the ABC and is discussed in section 5.0.00. However, the transfer of the data from lookahead to the C register is a function of the TBC. The only time that TBC action is required for instruction unit instructions is when they require a store to an internal register. The TBC action for internal instruction unit stores consist of a data transfer from the lookahead data field to the C register (via TOB). The operation is timed with the T timer. The TBC decoding of an internal instruction unit store allows the

d.c. gating out of the data field and the starting of the T timer to time the transfer. Further decoding of the instruction unit internal store conditions the TBC advance.

4.4.01 Input/Output Instructions

The TBC decoding of the I/O instructions is shown in figure 4.2-2. The decoding shows that a bit in position 7 of the operation code field selects the disc synchronizer while the absence of a bit in position 7 selects the exchange The initial decoding shown in figure 4.2-2 decodes the instruction as an I/O instruction only. Because the same decoded lines and fields are gated out and the T timer used for either type of I/O instruction, the initial I/O level decoding serves to gate out the lookahead data field and operation code field. The lookahead data field (via TOB) hold up the inputs to the input registers of both the EXCHANGE and disc sync. The output of the operation code field is further decoded in lookahead to determine the type of operation to be performed. The decoding is such that two decoded operation lines designate the EXCHANGE and disc sync and hold up inputs to both units. To time the transfer the T timer is started. The I/O level decoding is conditioned by MAR MODE to insure that the previous instruction completed without interrupting. No store operations can be preceeding the I/O instruction and this condition is indicated in the initial I/O level decoding by conditioning the decoding with the ABC-SCC interlock. If the SCC is at the same level as the ABC, which in turn is interlocked at the same level as the TBC, then by the interlock conditions it is evident that there is not a store preceeding the I/O instruction and that no interrupt occurred from the previous instruction. If either the interrupt occurred or a store was preceeding the I/O level instruction, it is impossible to interlock the counters at the same level. With the counters interlocked and MAR MODE present, the I/O level decoding allows the T timer to be started to time the transfer. Figures 4.2-4, 5

T-timer M and the TBC decode line identifying the proper Exchange establishes the d.c. "Select Basic Exchange" or "Select High Speed Exchange" signals. No further action occurs until the instruction has been accepted or rejected. When the Exchange accepts the instruction, the accept pulse is remembered in the I/O Reaction Storage trigger and, after proper clock pulse synchronization, the I/O response trigger serves to advance the TBC. When the Exchange rejects the instruction, this signal is sent to the Interrupt Mechanism controls to record the proper I/O reject indicator. This reject signal is then relayed to the lookahead and performs essentially the same action in establishing the TBC advance condition. In either case, the accept or reject signal resets T-timer M directly to terminate the Select signal. The second level provided for I/O instructions is a dummy level tagged with the IC bit such that the interrupt mechanism is interrogated.

4.4.02 Instruction Unit Instructions

The instructions executed within the instruction unit are generally completed within that unit to a point where it is necessary to store and/or enter indicators. These functions are completed by lookahead in proper sequence. Of these, only the instruction unit internal store levels require TBC action. All others are accomplished by the ABC and are explained in section 5.0.00.

The data path existing for transferring the store data from a lookahead level to an internal register exists only through the C register. The TBC action then is only decoding the instruction unit internal store and starting the T timer to time the data transfer. The decoding for the instruction unit internal store is shown in figure 4.2-2. The decoding allows the data field to be gated to the C register in the usual decode manner. The decoded internal store condition, combines with MAR MODE to start the T timer, (figures 4.2-4 and 5). The T timer output combines with

the decoded condition to set the C register. With the data in the C register the next action is to gate out the C register to the internal register specified by the LAAR. This latter action is a function of the AEC and is explained in the next section. The TBC decoded internal store combines with M of the ABC A timer to allow the TBC to advance (contingent upon the counter interlock). The advance, when allowed (no interlock) allows the normal advance control function to occur.

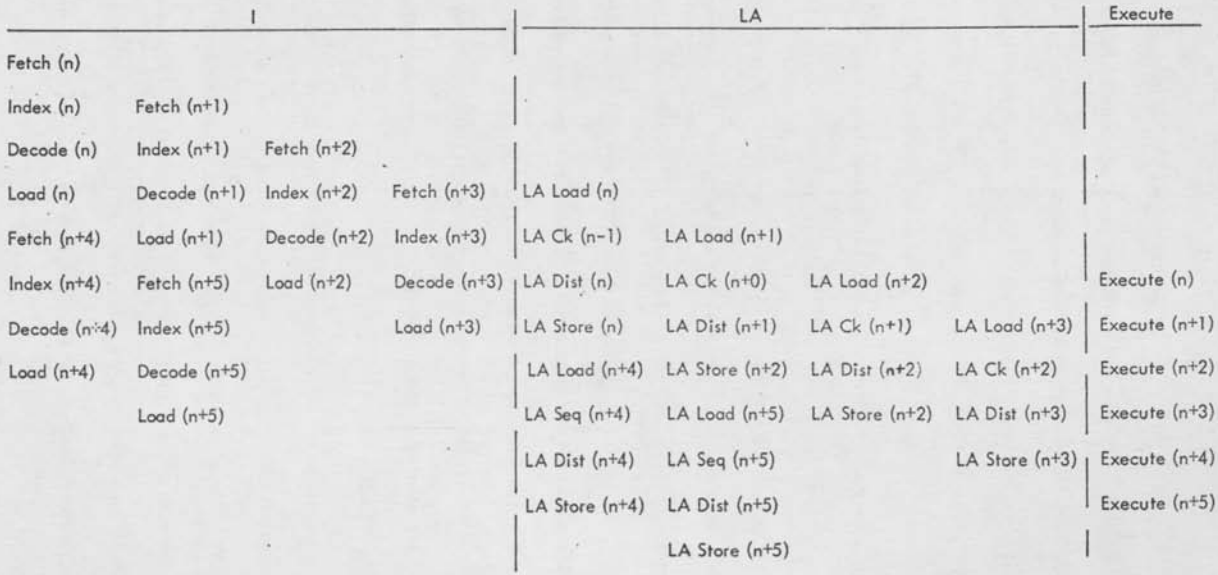


FIGURE 1.1-3. OVERLAP OPERATION

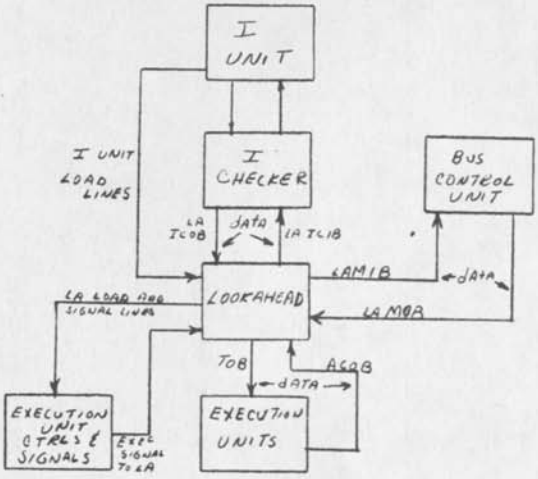


FIGURE 1.1-4. 7030 COMPUTER SYSTEM LOOKAHEAD RELATIONSHIP

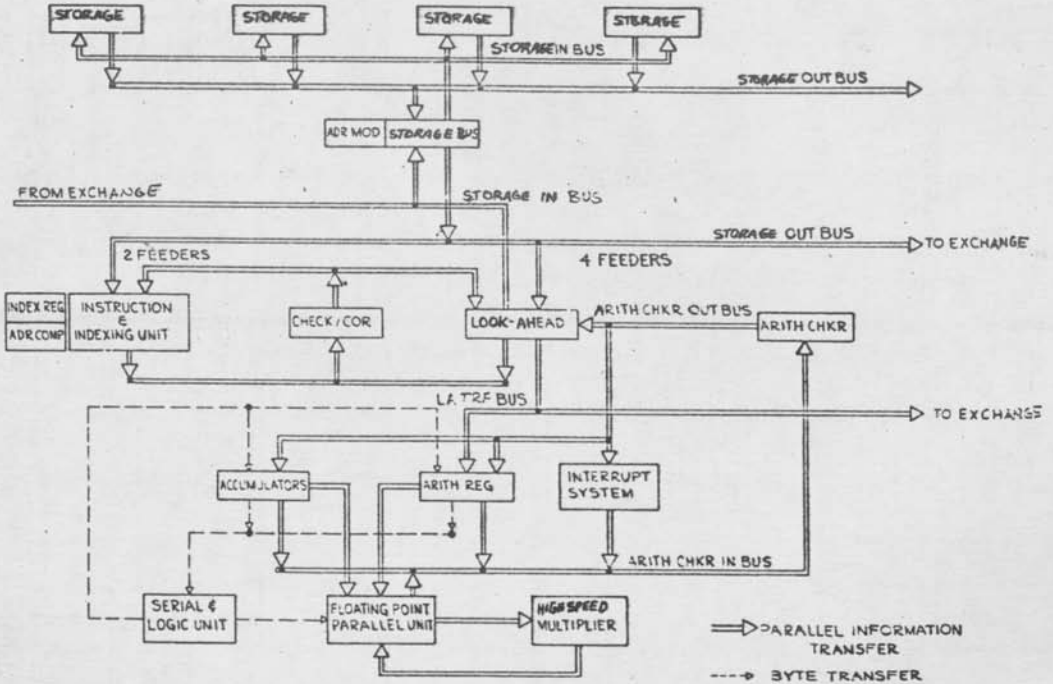


FIGURE 1.1-5. LOOK-AHEAD INPUT-OUTPUT

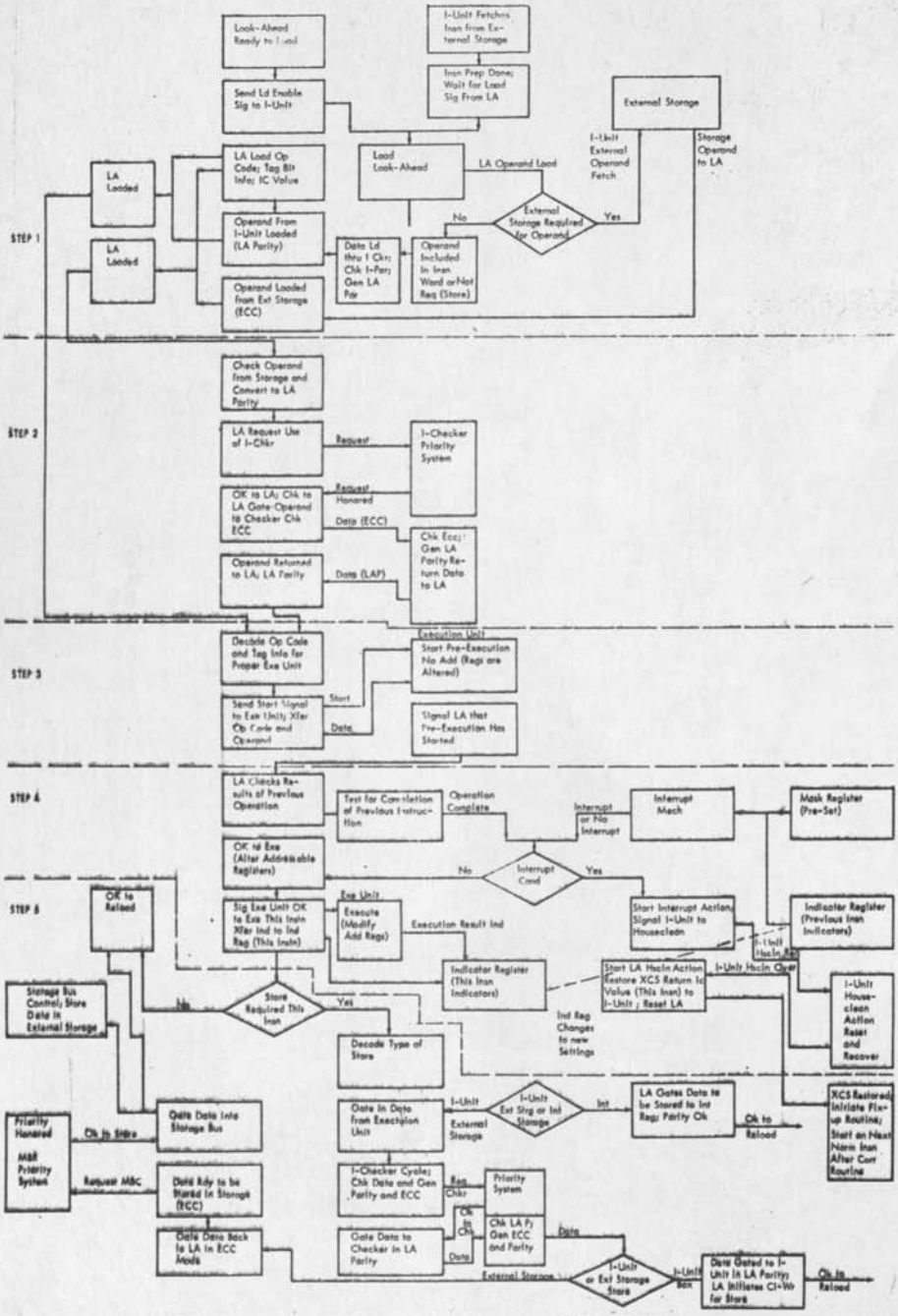


FIGURE 1.3-1. LOOK-AHEAD LOGIC OPERATIONS

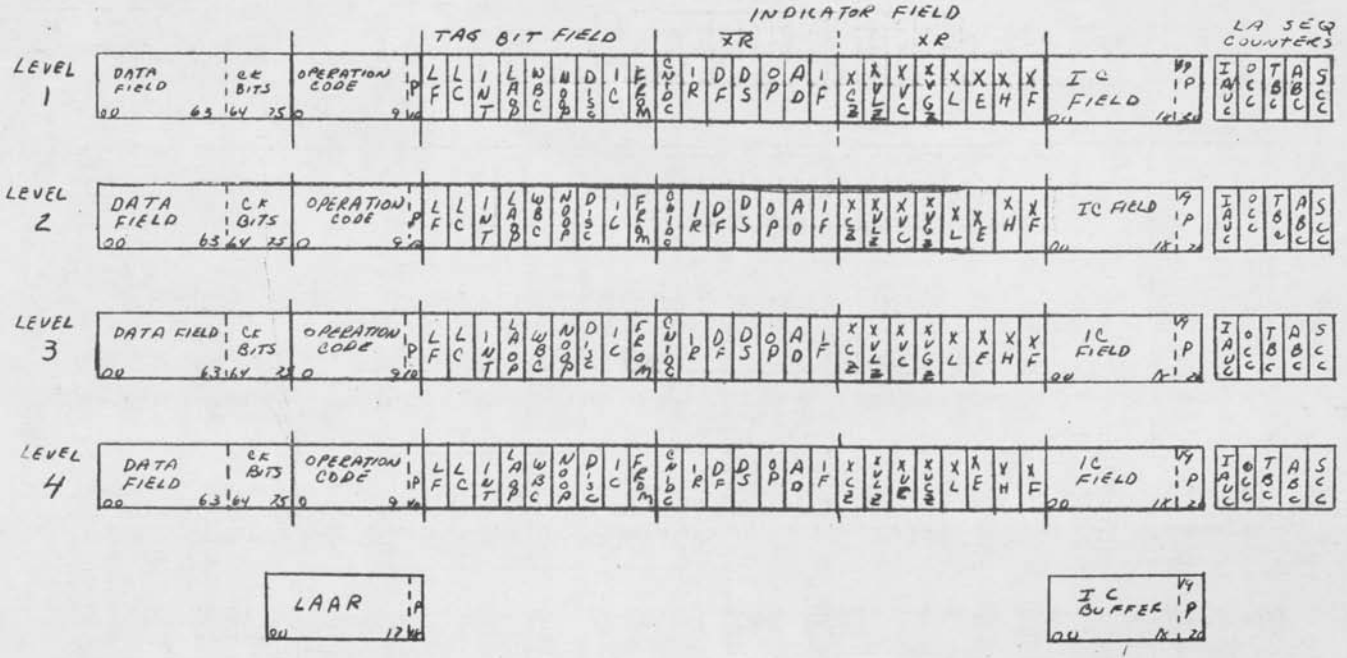


FIGURE 2.1-1. LOOK-AHEAD FIELD LAYOUT

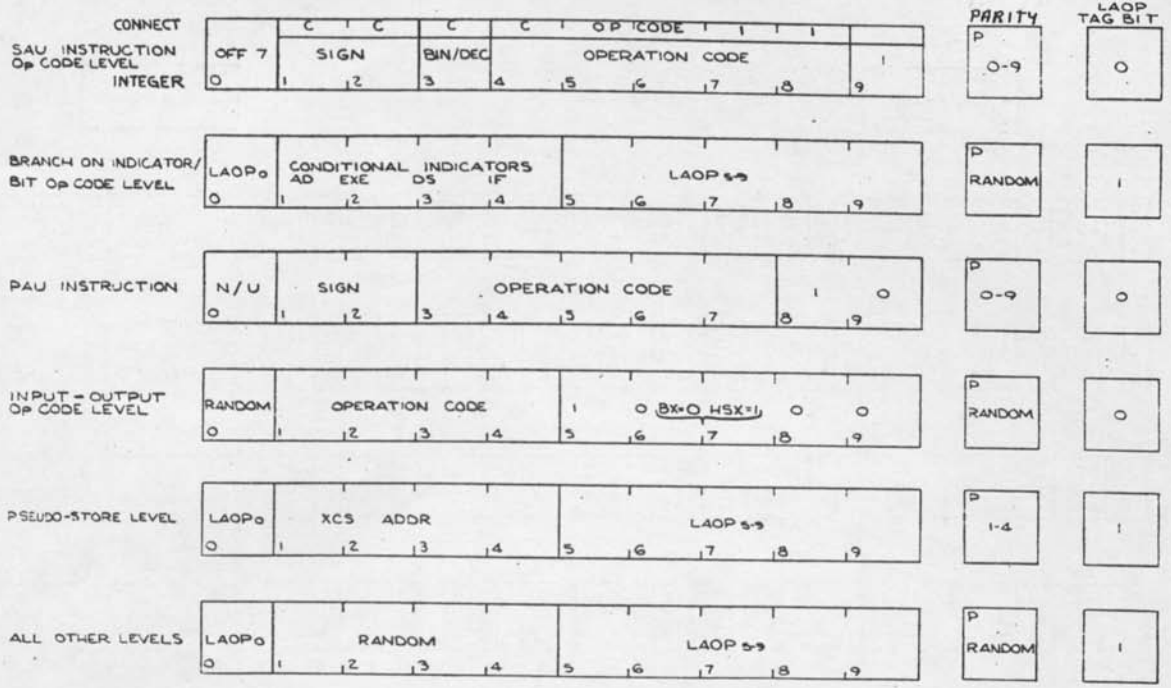


FIGURE 2.1-2. LOOK-AHEAD OPERATION CODE FIELD FORMATS

LA DATA FIELD AND PARITY SCHEME

	0 DATA FIELD 63	64	65	66	67	68	69	70	71	72	73	74	75
LA Parity (Bit Position)	→	P 0-7	P 8-11	P 12-15	P 16-23	P 24-31	P 32-39	P 40-47	P 48-55	P 56-59	P 60-63	X	X
ECC		0	0	C_0	C_1	C_2	C_4	C_8	C_{16}	C_{32}	C_T	0	0
Residue Bits	PAU SAU	X	X	X	X	X	X	X	X	X	X	Res 1	Res 2

SPECIAL OPERAND FIELD FORMATS (DATA FIELD)

	Data Field (0–63)	64	65	66	67	68	69	70	71	72	73	74	75
SAU Instruction Operation Code Level	P_x (0–2) | Length (3–8) | Byte Size (9–11) | Offset (12–18) | 000 ← → 000 (19–49) | Bit Address (50–55) | 0 ← → 0 (56–63)	X	X	X	X	1	1	1	X	1	1	0	0
Branch On Bit Operation Code Level	00 ← → 00 (0–49) | Bit Add (50–55) | 0000 (56–59) | LLF ZN I (60–63)	1	1	1	1	1	1	1	X	1	X	0	0
Branch On Indicator Operation Code Level	00 ← → 00 (0–49) | Indicator Address (50–55) | 0000 (56–59) | 0 LF Z N I (60–63)	1	1	1	1	1	1	1	X	1	X	0	0
Pseudo Branch On Ind for Interrupt-Op Code Level	Random (0–11) | Index Store Add (12–17) | Random (18–23) | 00 ← → 00 (24–59) | 0 LF Z N I (60–63)	X	X	X	X	1	1	1	1	1	X	0	0
Branch Recovery Level	Branch Address (0–18) | 000 ← → 000 (19–63)	X	X	X	X	1	1	1	1	1	1	0	0
SAU Immediate Data	Data (0–24) | 000 ← → 000 (25–63)	X	X	X	X	X	1	1	1	1	1	X	X
PAU Immediate Data	Data (0–11) | Random (12–23) | 000 ← → 000 (24–63)	X	X	X	X	1	1	1	1	1	1	X	X
Input Output	Control Word Address (0–17) | 00 ← → 00 (18–31) | Channel Address (32–38) | 00 ← → 00 (39–63)	X	X	X	X	1	X	1	1	1	1	0	0

← Data Field → (bits 0–63) ← LA Parity → (bits 64–73) Residue (bits 74–75)

X = Can Be 1 or 0

FIGURE 2.1-3. LOOKAHEAD DATA FIELD LAYOUT

FLOATING POINT INSTRUCTIONS

FLP TYPE & MNEUMONIC	FIRST LA LEV	SECOND LA LEVEL	REMARKS
NON STORE OPERATIONS (A)(L)(R)(AR)(*)(/)(AU)(LWR)(RF)(RFR)(R/)(D+)(DL)(F+)(E+)(D-)(D-MG)(DLWR)(SWF)(E-I)	OPERATION CODE & OPERAND	NOT REQUIRED	
ADD TO MEMORY OPERATIONS (M+)(AUM)	OPERATION CODE AND OPERAND	STORE	
LOAD FACTOR OPERATIONS (LFT)(D/)	OPERATION CODE & OPERAND	STORE	CUMULATIVE MULTIPLICAND
MULTIPLY AND ADD (*+)	OPERATION CODE AND OPERAND	SPECIAL OPERAND	
STORE OPERATIONS (ST)(SRD)(SLO)(SRT)	OPERATION CODE & STORE	NOT REQUIRED	

VARIABLE FIELD LENGTH INSTRUCTIONS

VFL TYPE & MNEUMONIC	P_1	N_{B_2}	FIRST LA LEV	SECOND LA LEV	THIRD LA LEV	FOURTH LA LEV	FIFTH LA LEV	SIXTH LA LEV	REMARKS
NON-STORE OPERATION (+)(LCV)(L)(R)(K)(KE)(CT)(M)(CV)(AU)(LWR)(KF)(KFE)(KFR)(CTX)(DCV)	N	N	OPERATION CODE	ONLY OPERAND					NO STORE OPERAND POSSIBLE
	Y	N	OPERATION CODE	ONLY OPERAND	PX PSEUDO STORE				
	N	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND				
	Y	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	PX PSEUDO STORE			
STORE OPERATIONS (M+)(ST)(EM)(AUM)(MI)(SRND)(FM)	N	N	OPERATION CODE	ONLY OPERAND	STORE				TWO STORE FUNCTION POSSIBLE
	Y	N	OPERATION CODE	ONLY OPERAND	STORE	PX PSEUDO STORE			
	N	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	FIRST STORE	SECOND STORE		
	Y	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	FIRST STORE	SECOND STORE	PX PSEUDO STORE	
LOAD ~~STANDARD~~ OPER (*-DEC)(/-DEC)(LTRCV) (*+-DEC)(LTRS)	N	N	OPERATION CODE	ONLY OPERAND	STORE				POSSIBLE TO HAVE 2 RTN OPND BUT ONLY ONE STORE OPN.
	Y	N	OPERATION CODE	ONLY OPERAND	STORE	PX PSEUDO STORE			
	N	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	STORE			
	Y	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	STORE	PX PSEUDO STORE		
MULTIPLY & ADD (BIN) (*+)	N	N	OPERATION CODE	ONLY OPERAND	SPECIAL MPY OPERAND				
	Y	N	OPERATION CODE	ONLY OPERAND	SPECIAL MPY OPERAND	PX PSEUDO STORE			
	N	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	SPECIAL MPY OPERAND			
	Y	Y	OPERATION CODE	FIRST OPERAND	SECOND OPERAND	SPECIAL MPY OPERAND	PX PSEUDO STORE		
BRANCH ON INDICATOR (BX)	N	N	OPERATION CODE	BX OPERAND	STORE	RECOVERY			
BRANCH ON BIT (BB)	N	N	OPERATION CODE	ONLY OPERAND	STORE	RECOVERY			
STR INSTR IF BF IND BIT=1	N	N	OPERATION CODE	BF OPERAND	STORE	RECOVERY	DUMMY LEVEL		

INPUT/OUTPUT INSTRUCTIONS

I/O INSTR MNEUMONIC	FIRST LA LEV	SECOND LA LEV	
EXCHANGE-(ECW)(RD)(W)(RTL)(LOC)(WE)(CTLE)(LWE)(RCE)(RCLE) DISC SYNC-(DCW)	OPERATION CODE	DUMMY	

FIGURE 2.2-1. LA LEVEL REQUIREMENTS

I UNIT LOADING SEQUENCES

Name	MNEMONIC	L. A. LOADING TYPES 1st Level	2nd Level	3rd Level	4th Level
CZRF - M	V / ICR	Pseudo Store	(Pseudo Store)		
Load Index	LX				
Load Value	LV				
Load Count	LC				
Load Refill	LR				
Store Index	SX				
Store Value	SV				
Store Count	SC				
Store Refill	SR				
Increment	V /				
Incr. & Cnt.	V / C				
Incr. Cnt. & Ref. (NM)	V / CR				
Compare Value	KV				
Compare Count	KC				
Load Value IMM	LVI				
Load Count IMM	LCI				
Load Refill IMM	LRI				
Increment IMM	V / I				
Incr. Neg. IMM	V - I				
Incr. & Cnt. IMM	V / IC				
Incr. Neg. & Cnt. IMM	V - IC				
Incr. Cnt. & Ref. IMM (NM)	V / ICR				
Incr. Neg. Cnt. & Ref. (NM)	V - ICR				
Add to Count IMM	C / I				
Add Neg. to Cnt. IMM	C - I				
Refill ZRO Cnt (Cnt-O)	RCZ				
Refill ZRO Cnt (Cnt-O)	RCZ				
Refill	R				
Store Valin Addr	SVA				
Store Zero	Z				
LD Val. Effect	LVL				
LD Val Wi Sum	LVS				
Comp Valve IMM	KVI				
Comp Val. Neg. IMM	KVNI				
Comp. Cnt. IMM	KCI				
Load Val. Neg. IMM	LVNI				
CZRF - M	V / CR				
Ref. Cnt Zro. Cnt - O	SX SV SC SR R RCZ SVA Z	Real Ext Store in ECC	(Real Ext Store in ECC)		
	SX Z	Real Store in P & R			
Cnt - O	SV SC SR R RCZ SVA	(Int. fetch request	Real Store in P & R		
CZRF - NM CZRF - M	LX LV LC LR V / V / C V / CR LVE KV KC V / CR	Int Fetch Reg	Pseudo Store - (May be more than one IRF Level)		
RZC C-O RZC C-O		(Int fetch-Ind Xfer)	(Ind Transfer only)		

I UNIT LOADING SEQUENCES

Name	MNEMONIC	L. A. LOADING TYPES 1st Level	2nd Level	3rd Level	4th Level
Rename	RNX	(clear LA level)	(pseudo store)	Ind Xfer Only	
Rename If Intrpt or Nop	RNX				
Rename Normal	RNX	"			
Rename Normal	RNX	"	Real Ext. Store		
Transmit-Swap	T Swap	Test Level	or (Ext Store or Int Std	(Ind. Xfer only)	
			(I Bx Int Fetch)	Real Store in P & R	
Execute Indirect If No Opd	EXIC	Sp Ext Stm Real Ext St	Any other Instruction Level		
Branch	B				
Branch Rel	BR				
Bran Enable	BE				
Bran Disable	BD	(Ind xfr only)			
Bran Indic. If Not E Box Test	BI				
1st Level Set Wait	BEW				
SIC B	SIC B				
	SIC BR	(pseudo store)			
	SIC BE				
	SIC BD	"			
Count & Bran (If No OPD)	CB	"			
	SIC BR	"			
(If No OPD)	CBR SIC CBR				
(If Not E Box Test)	SIC BIN				
1st Level	SIC BEW				
(If Not E Box Test) 1st Level	SIC B SIC BR SIC BE SIC BD SIC BI SIC BEW	(real ext store in ECC)			
No Opcd No Opcd	SIC CB SIC CBR	(pseudo store)	(pseudo store)		
2nd Level	BEW	(test level)			
3rd Level	BEW	(test level)			

FIGURE 2.2-2.

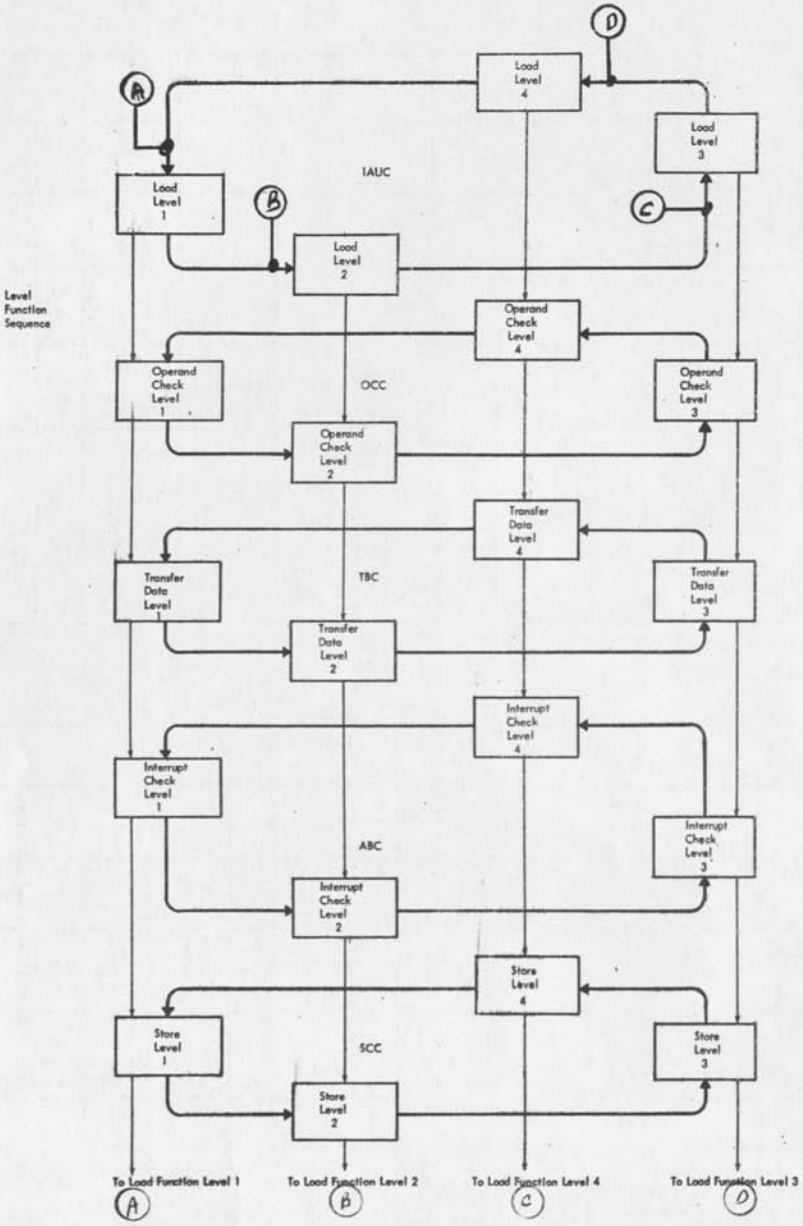


FIGURE 2.4-2. COUNTER CONTROL

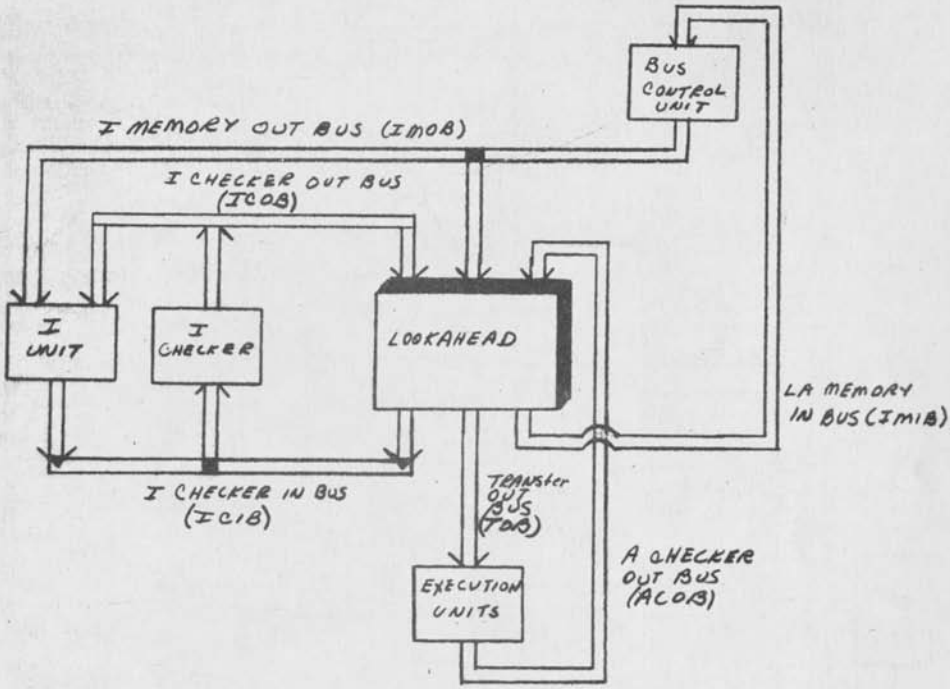


FIGURE 2.5-1. LOOK-AHEAD DATA FLOW

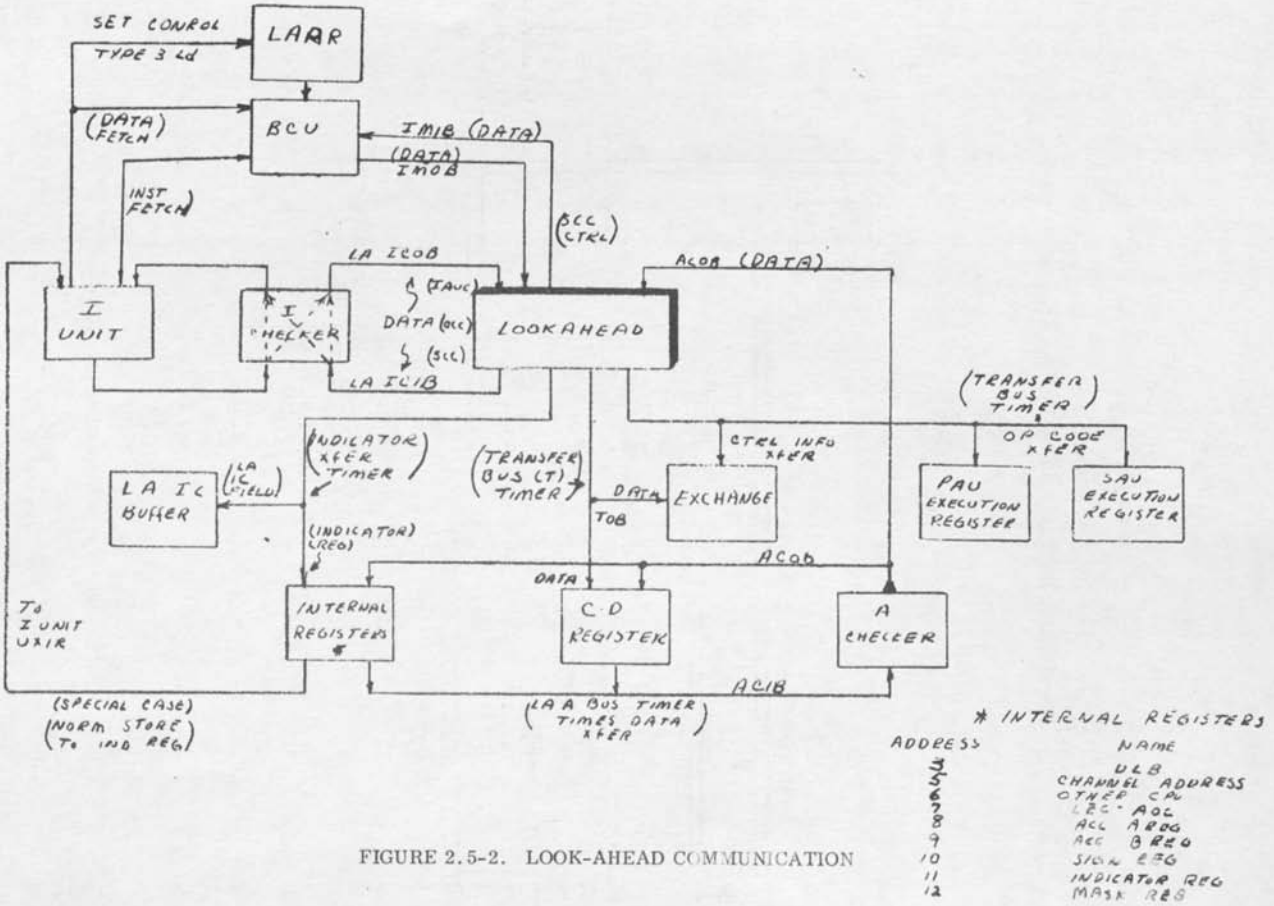


FIGURE 2.5-2. LOOK-AHEAD COMMUNICATION

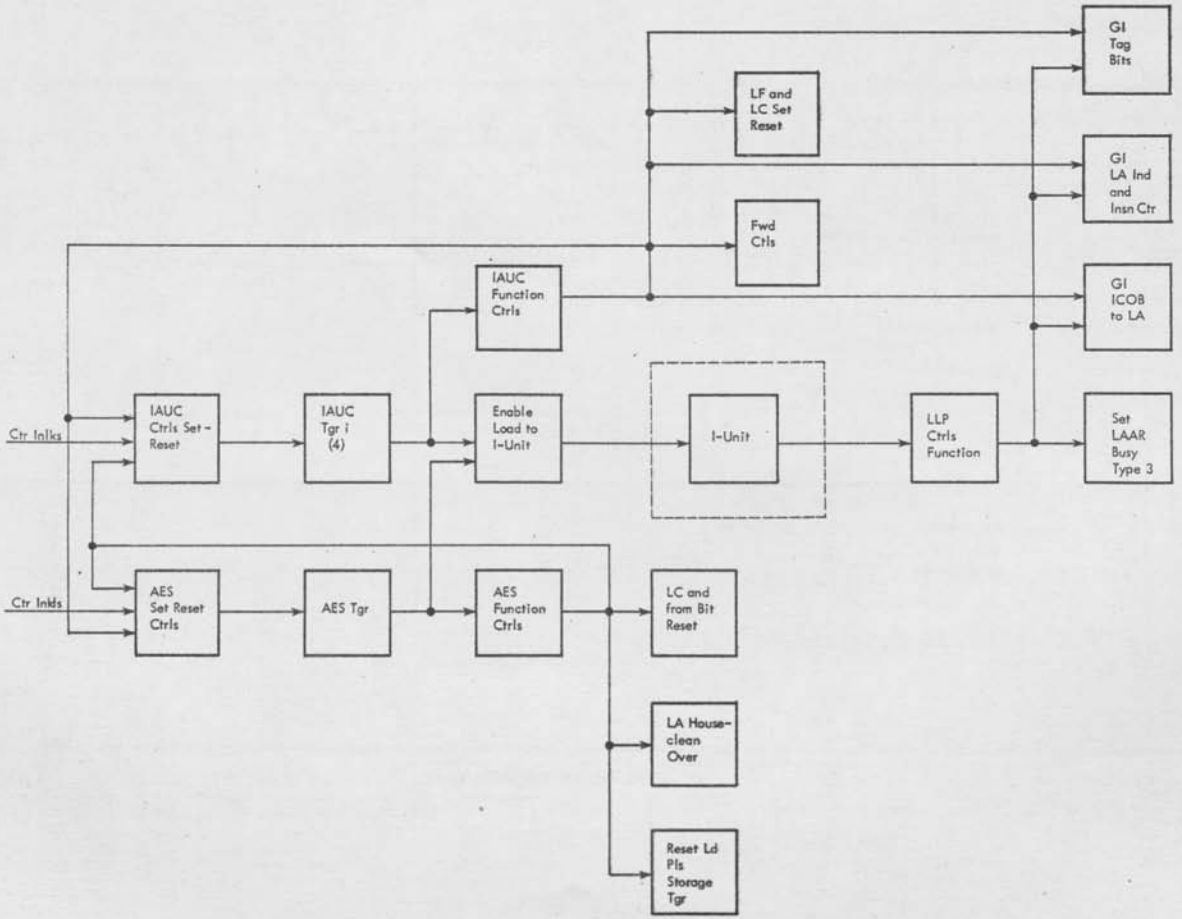


FIGURE 3.1-1. LOOK-AHEAD LOADING OVER-ALL LOGIC

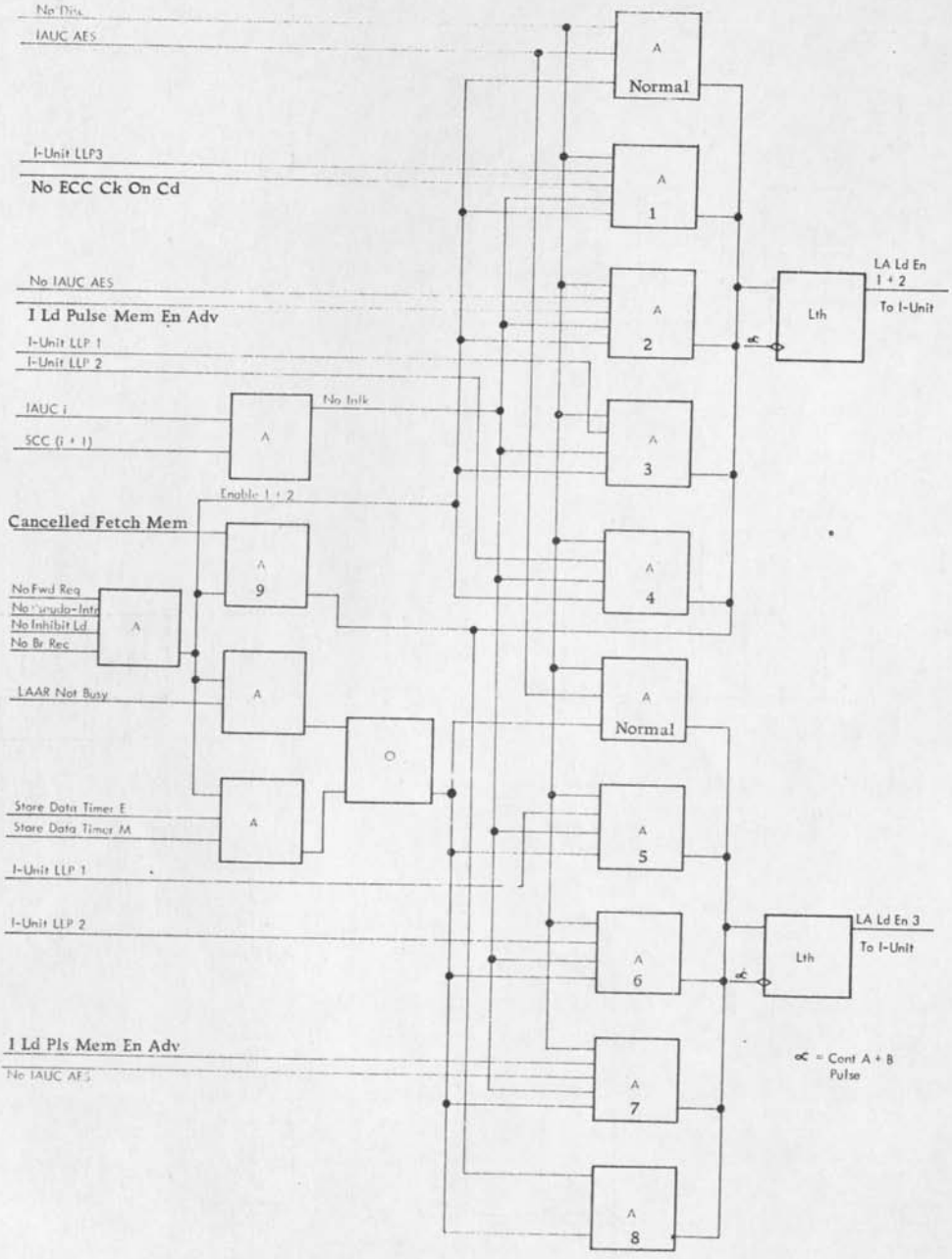


FIGURE 3.2-1. REPRESENTATIVE LOGIC LA ENABLE LOAD TO I-UNIT

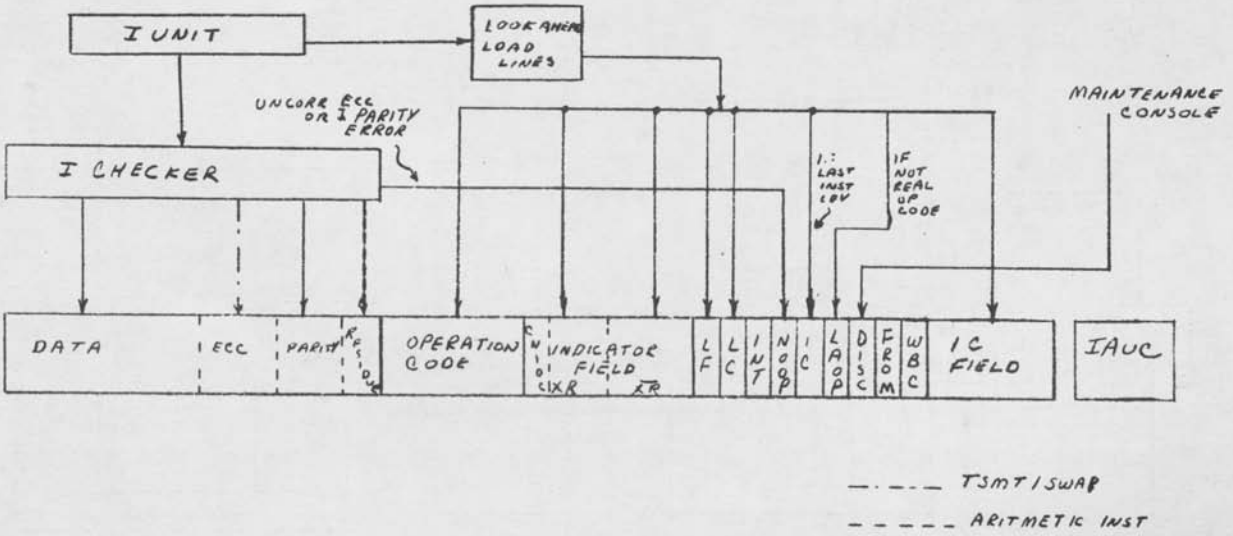


FIGURE 3.2-2. TYPE 1 LOAD LOOK-AHEAD

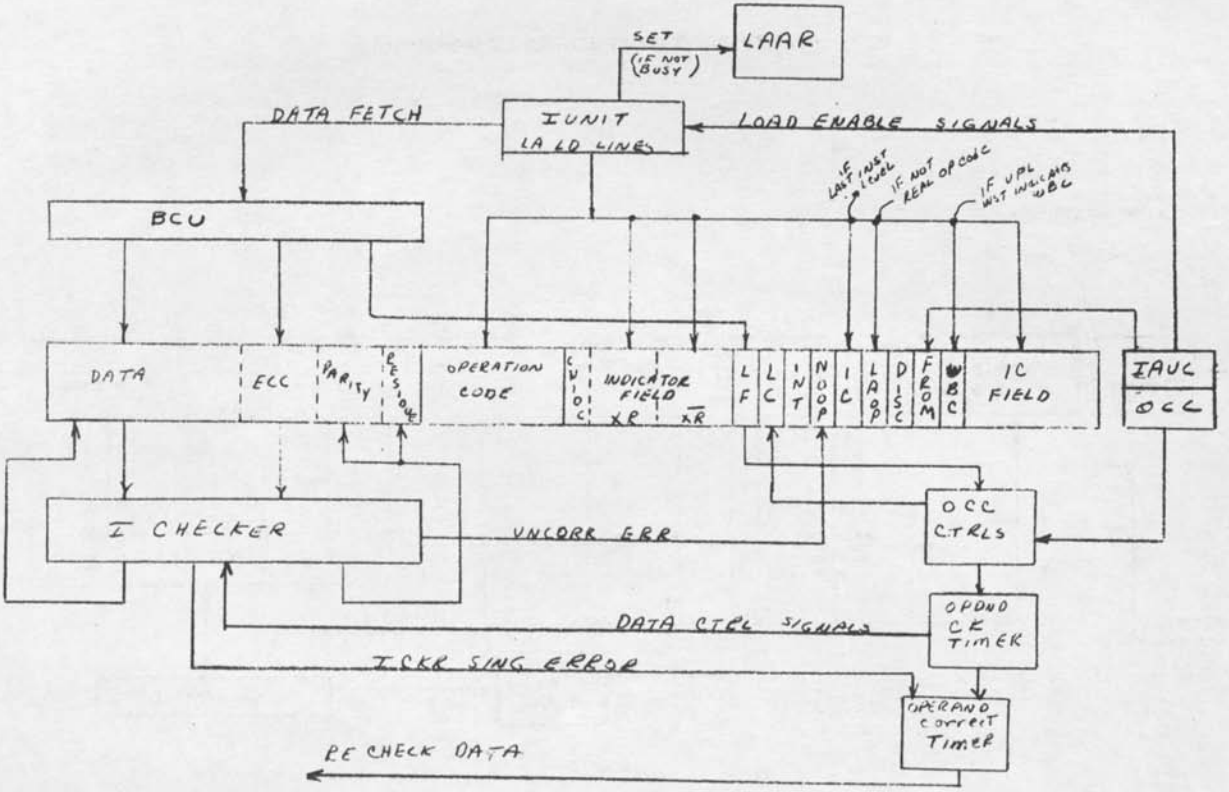


FIGURE 3.2-3. TYPE 2 LOAD AND DATA CHECK AND CONVERSION LOOK-AHEAD

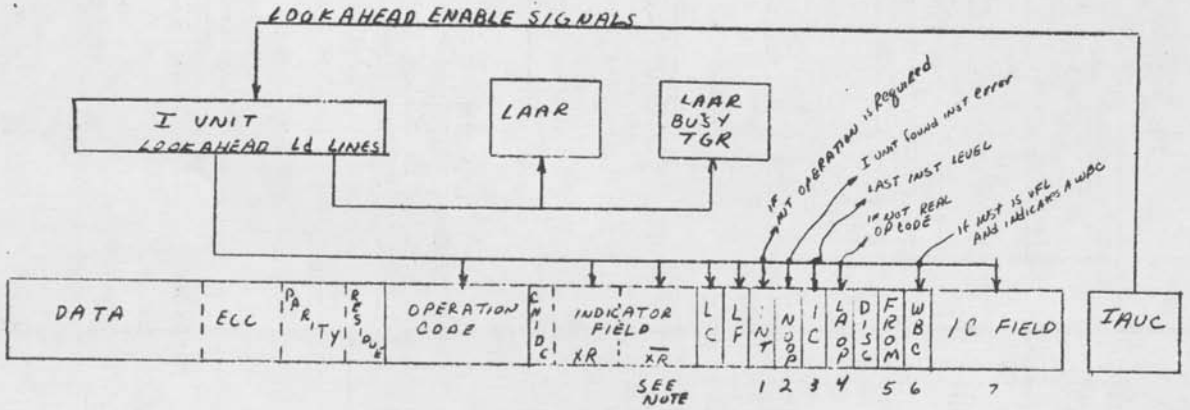


FIGURE 3.2-4. TYPE 3 LOAD LOOK-AHEAD

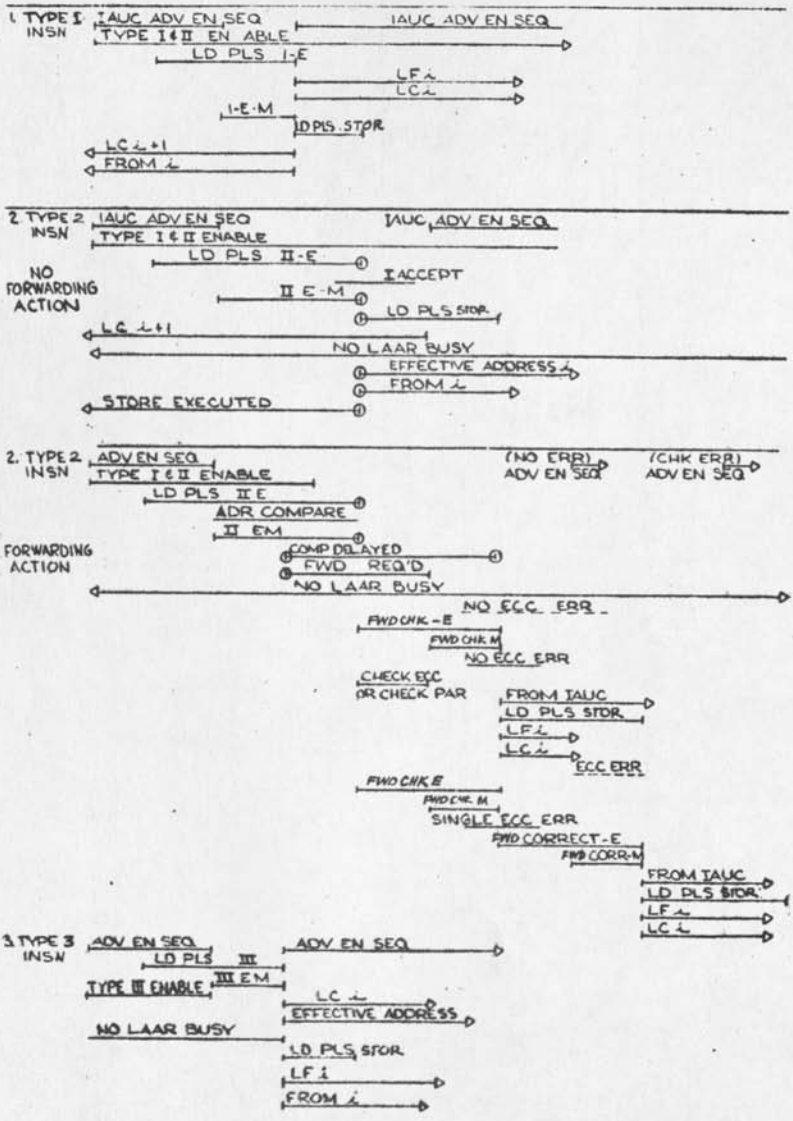
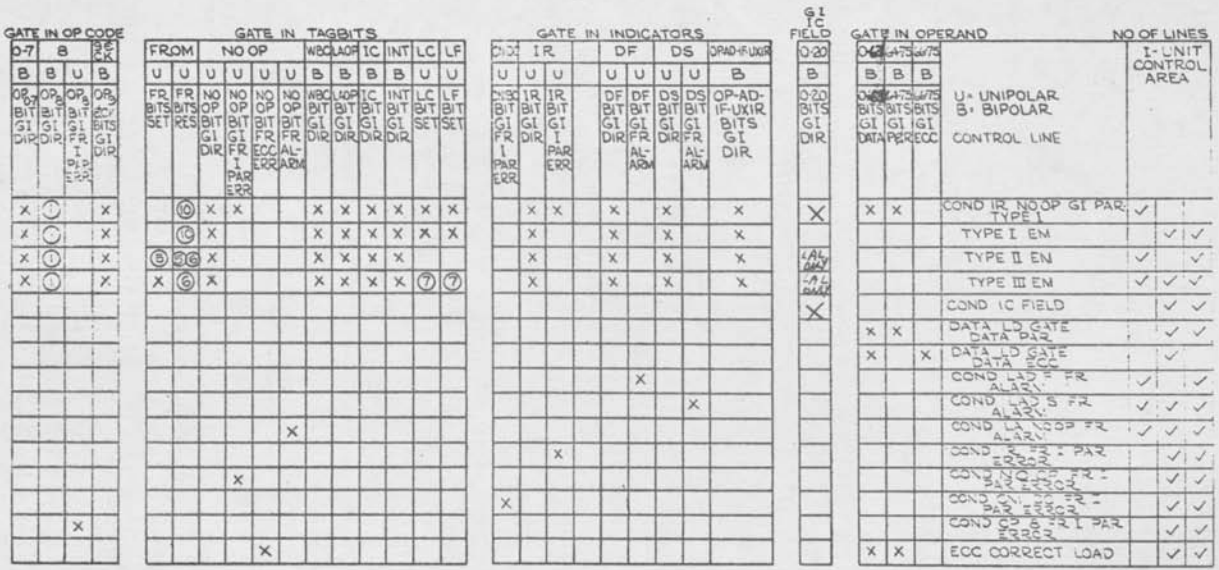


FIGURE 3.2-5. IAUC LEVEL FUNCTION



① IF NO COND OP & FR I PAR ERR
⑤ IF NO COMPARE NO LAAR BUSY
⑥ IF NO IAUC
⑦ IF NO SINGLE ECC ERR + NO ECC·CHK·LD·E1
⑩ IF IAUC

LAL = FLP, VFL, I/O
E1 = I Unit Execute Instructions (Except Branch)
E2 = Branch Instructions

FIGURE 3.2-6. LOOK-AHEAD LOADING LINE DEFINITION -- INFORMATION TRANSFER

	ANTIC IAUC ADV	SET LLP MEM	ANTIC I-II ENABLE	ANTIC III ENABLE	SET LAAR BUSY	RESET STORE EXEC	SET ADS FWD REQ	SET FWD REQ	RESET IAUC AES	DETERMINE CHKR PRIOR	NO. OF LINES (I UNIT CONTROL AREA) LAL	E1	E2
COND IR-COND NOOP G1 PAR-TYPE 1 E.M	X	X	X	9							✓		
TYPE 1 E.M	X	X	X	9								✓	✓
TYPE 2 E.M		2	X	9		5	3	4			✓		✓
TYPE 3 E.M	8	7	8		X	X					✓	✓	✓
LA LOAD E.M									X		✓	✓	✓
CHECK CYCLE E.M										X	✓	✓	✓
MPYC*LD*EM*TO*LA		INHIBIT GATE IN LAAR IF NO LAAR BUSY									✓		
TYPE 2 CANCEL STOR		REPLACE IAUC AES PARTICIPATION IN ENABLE SIGNALS									✓		
ECC CHECK LOAD		INHIBIT ACTION - SEE NOTES 7 AND 8										✓	

2 W STOR ACCEPT
3 IF COMPARE-IAUG$_i$-F$_i$
4 IF COMPARE
5 IF NO COMPARE. NO LAAR BUSY
7 IF NO SINGLE ECC ERR + NO ECC-CHK-LD-E1
8 IF NO ECC-CHK-LD-E1
9 IF NO LAAR BUSY

LAL = FLP, VFL, I/O
E1 = I Unit Execute Instructions (Except Branch)
E2 = Branch Instructions

FIGURE 3.2-7. LOOK-AHEAD LOADING LINE DEFINITION -- CONTROL FUNCTIONS

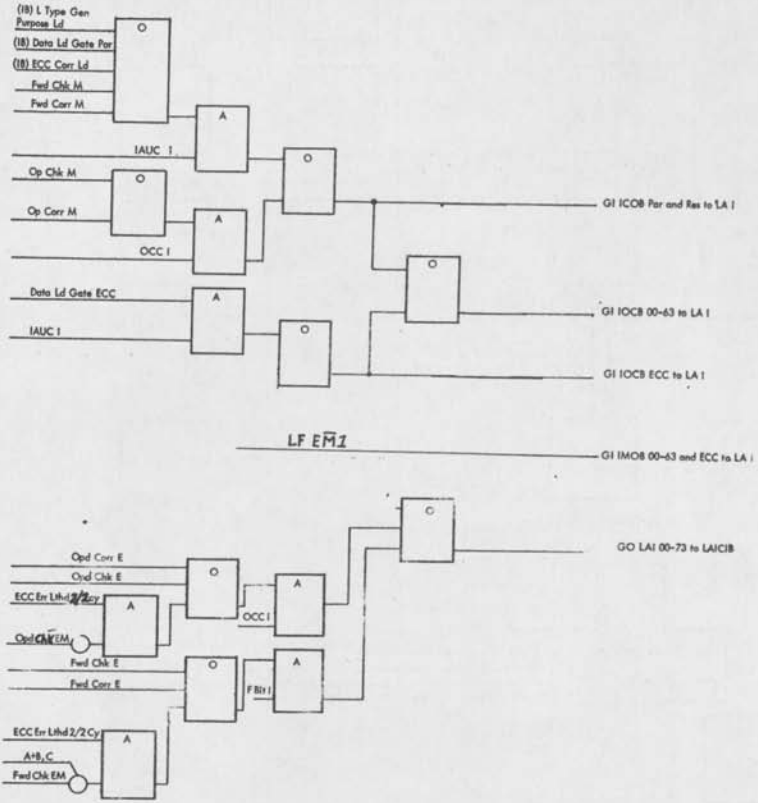


FIGURE 3.2-8. GI AND GO LINES TO LOOK-AHEAD REGISTERS INSTRUCTION PREPARATION

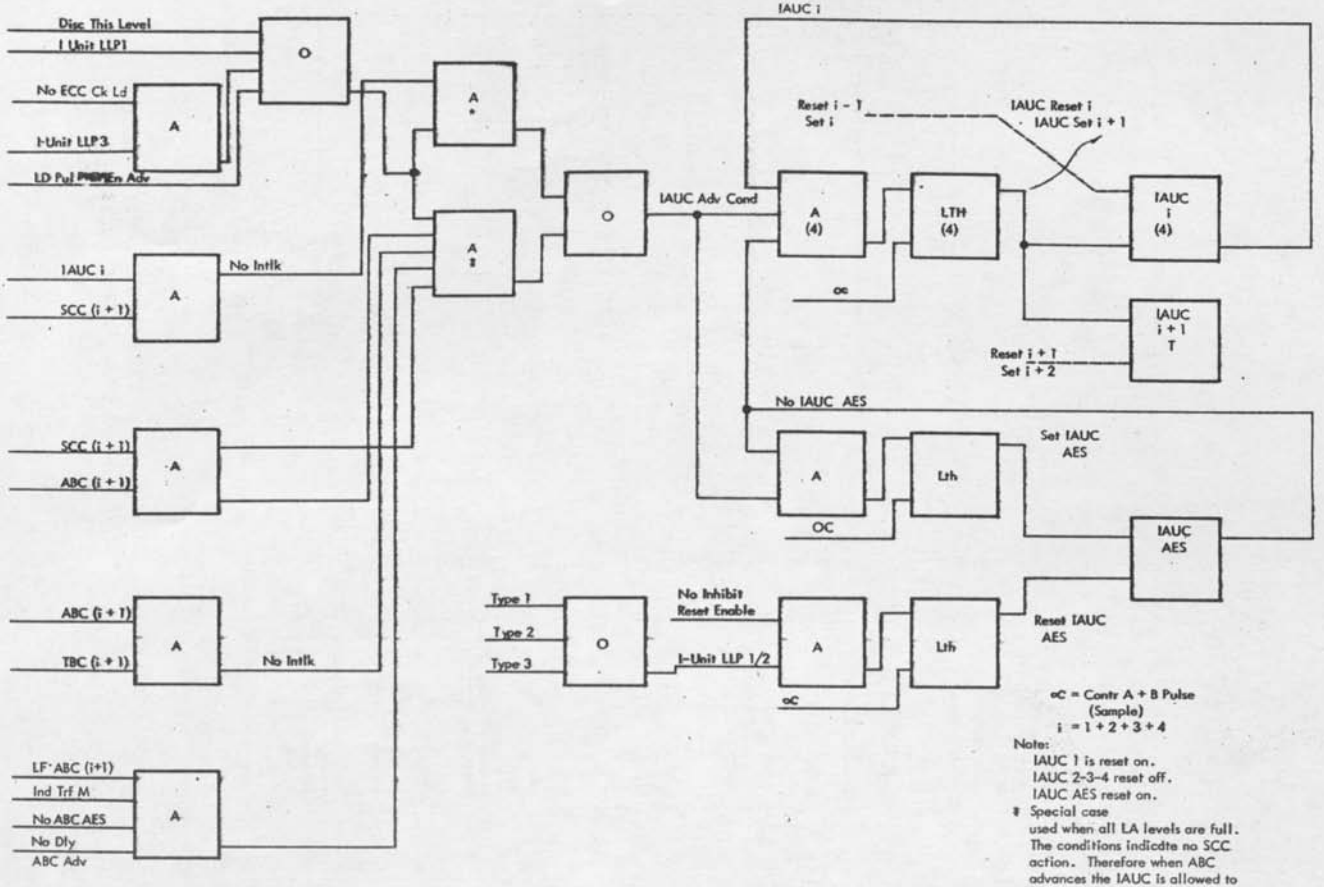


FIGURE 3.2-9. REPRESENTATIVE LOGIC OF IAUC COUNTER ADVANCE AND AES SET AND RESET

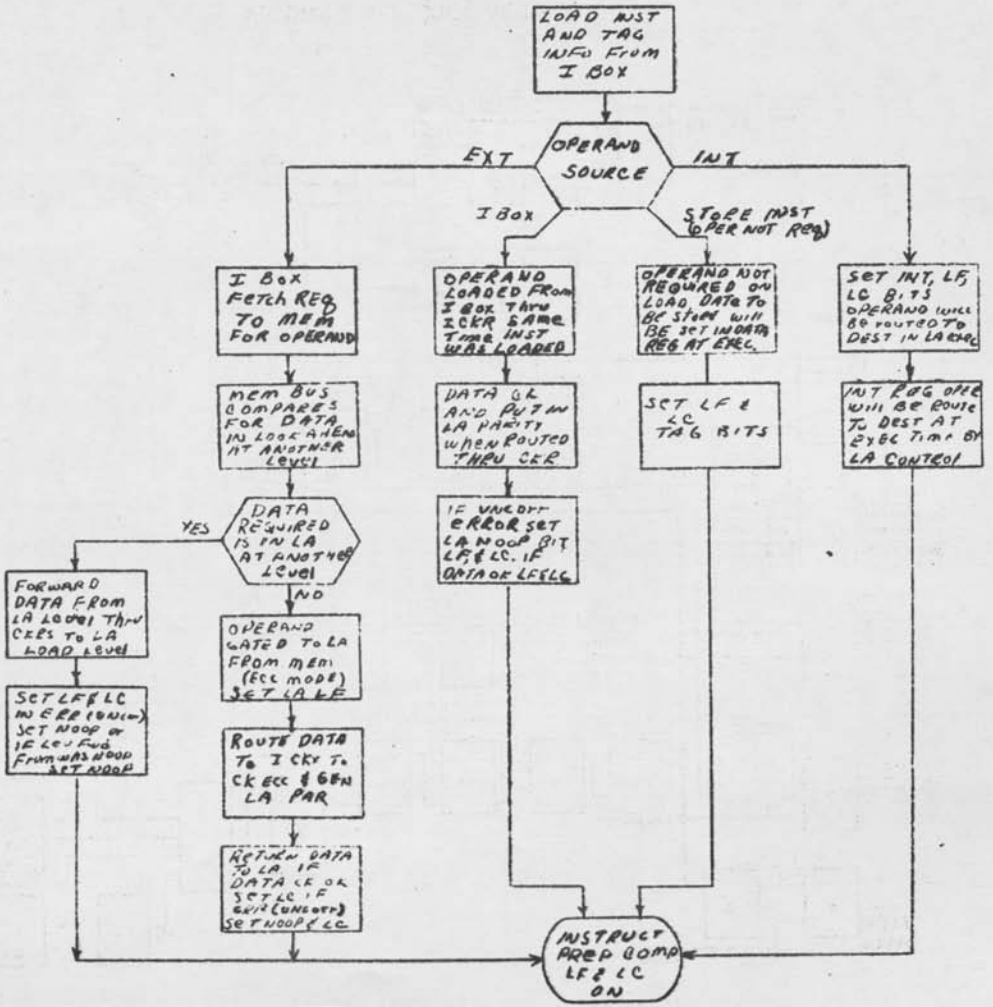
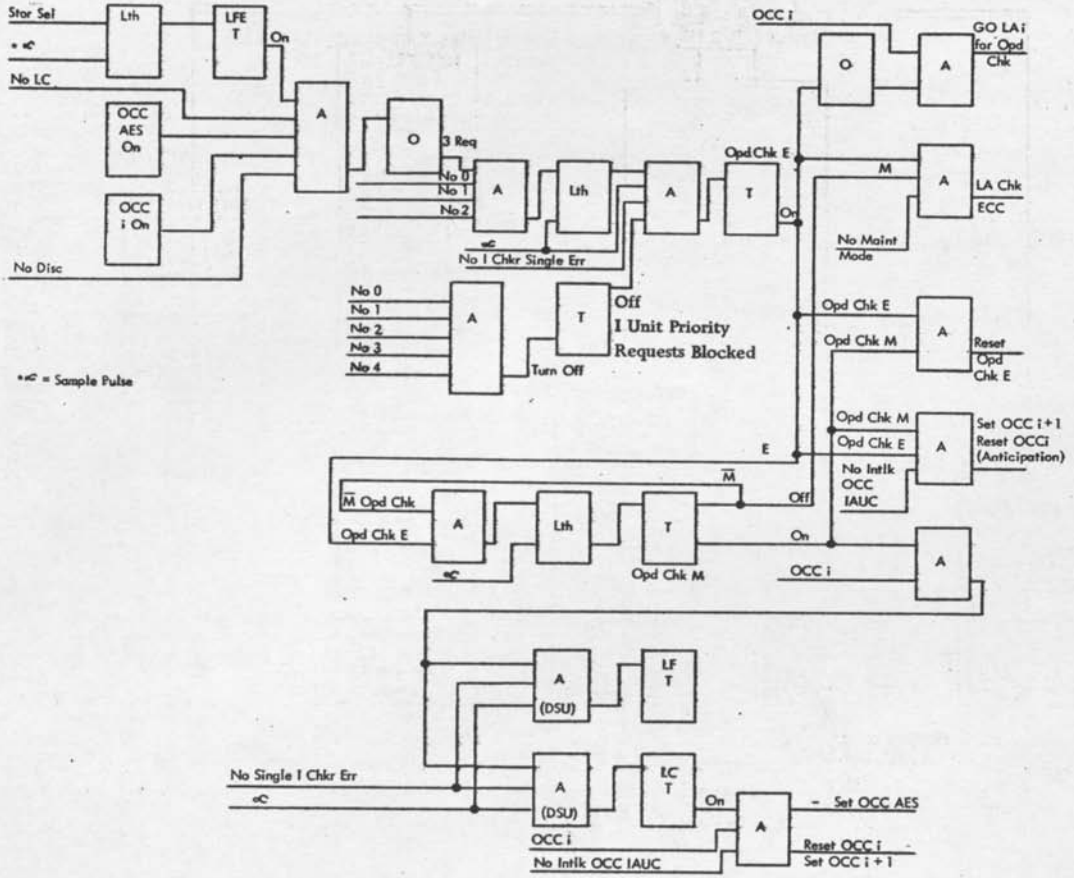

LOAD INST AND TAG INFO FROM I BOX
OPERAND SOURCE
EXT
INT
I BOX
STORE INST (OPER NOT REQ)
I BOX FETCH REQ TO MEM FOR OPERAND
OPERAND LOADED FROM I BOX THRU ICKR SAME TIME INST WAS LOADED
OPERAND NOT REQUIRED ON LOAD, DATA TO BE STORED WILL BE SET IN DATA REG AT EXEC.
SET INT, LF, LC BITS OPERAND WILL BE ROUTED TO DEST IN LA CKT
MEM BUS COMPARES FOR DATA IN LOOK ASIDE AT ANOTHER LEVEL
DATA OK AND PUT IN LA PARITY WHEN ROUTED THRU CKR
SET LF & LC TAG BITS
INT REG OPER WILL BE ROUTED TO DEST AT SYSC TIME BY LA CONTROL
DATA REQUIRED IS IN LA AT ANOTHER LEVEL
YES
NO
IF UNCORR ERROR SET LA NOOP BIT LF & LC. IF DATA OK LF&LC
FORWARD DATA FROM LA LEVEL THAT CKTS TO LA LOAD LEVEL
OPERAND GATED TO LA FROM MEM (ECC MODE) SET LA LF
SET LF & LC IN ERR (UNCORR) SET NOOP OR IF LF=1 FROM MEM NOOP SET NOOP
ROUTE DATA TO I CKT TO CK ECC & GEN LA PAR
RETURN DATA TO LA IF DATA LF OR SET LC IF ERR (UNCORR) SET NOOP & LC
INSTRUCT PREP COMP LF & LC ON

FIGURE 3.3-1



= Sample Pulse

FIGURE 3.3-2. OCC ACTION

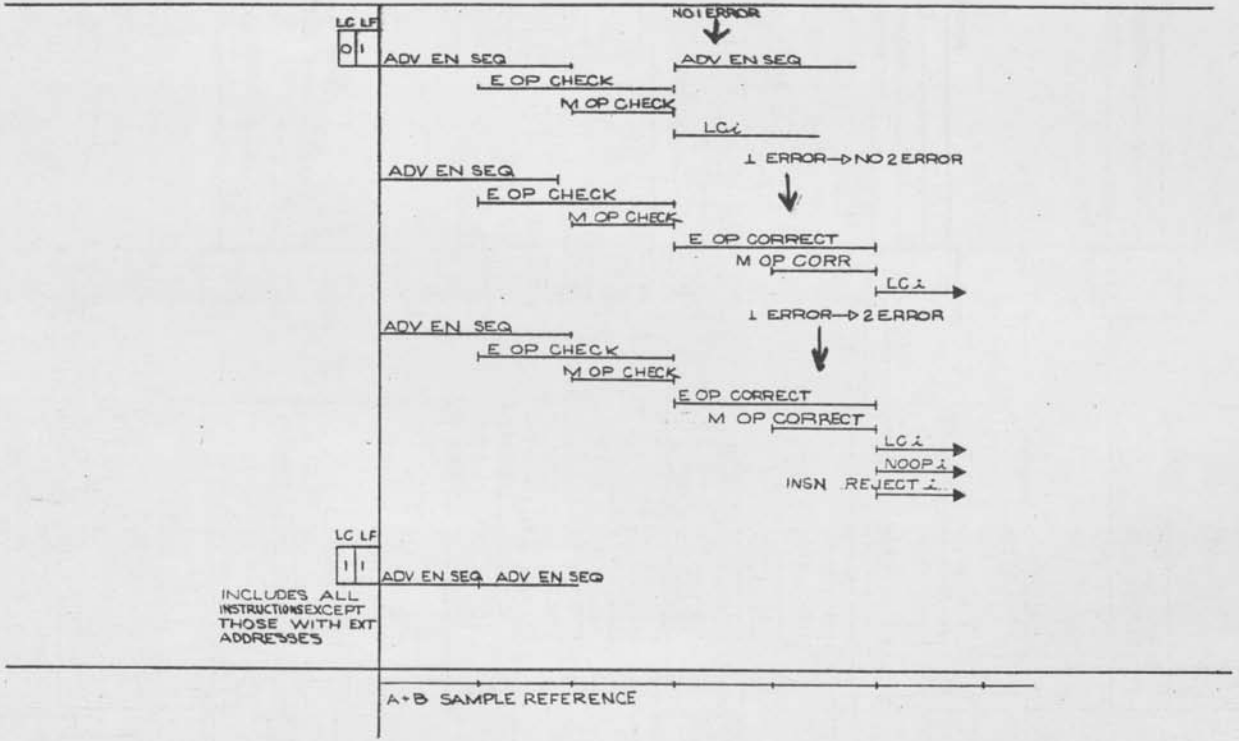


FIGURE 3.3-3. OCC LEVEL FUNCTION

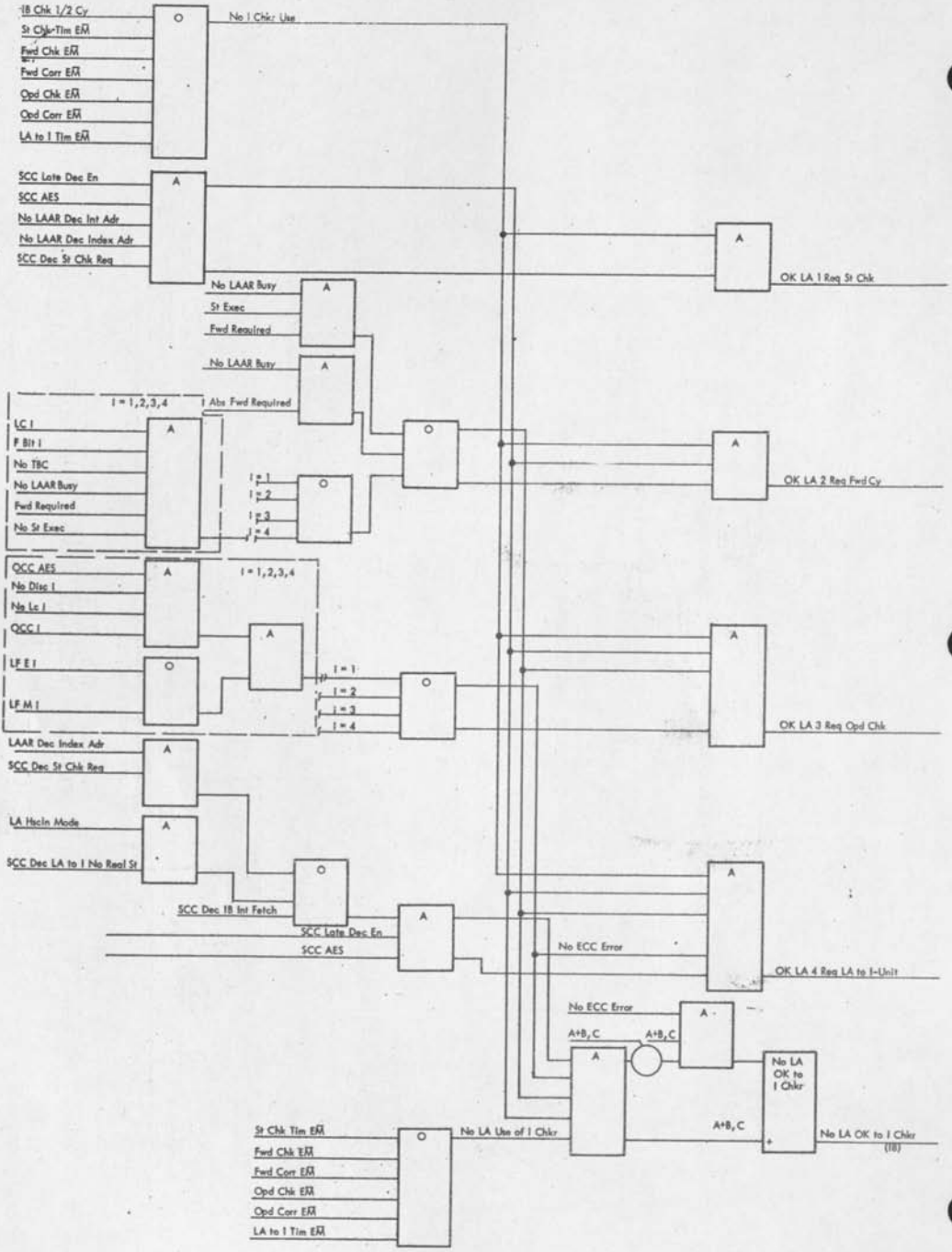


FIGURE 3.3-4. I-CHECKER PRIORITY° OK TO I-CHECKER

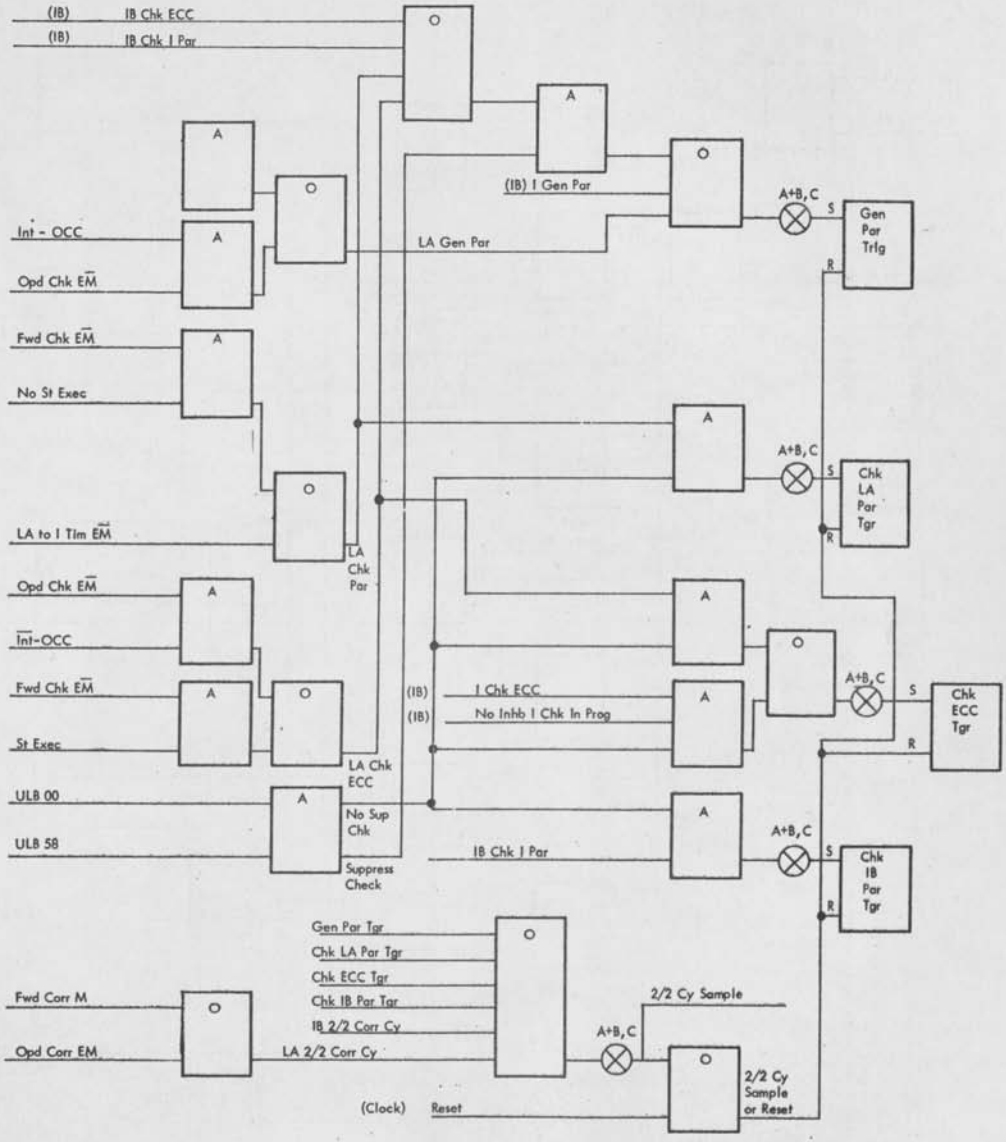


FIGURE 3.3-5. I CHECKER INPUT CONTROLS

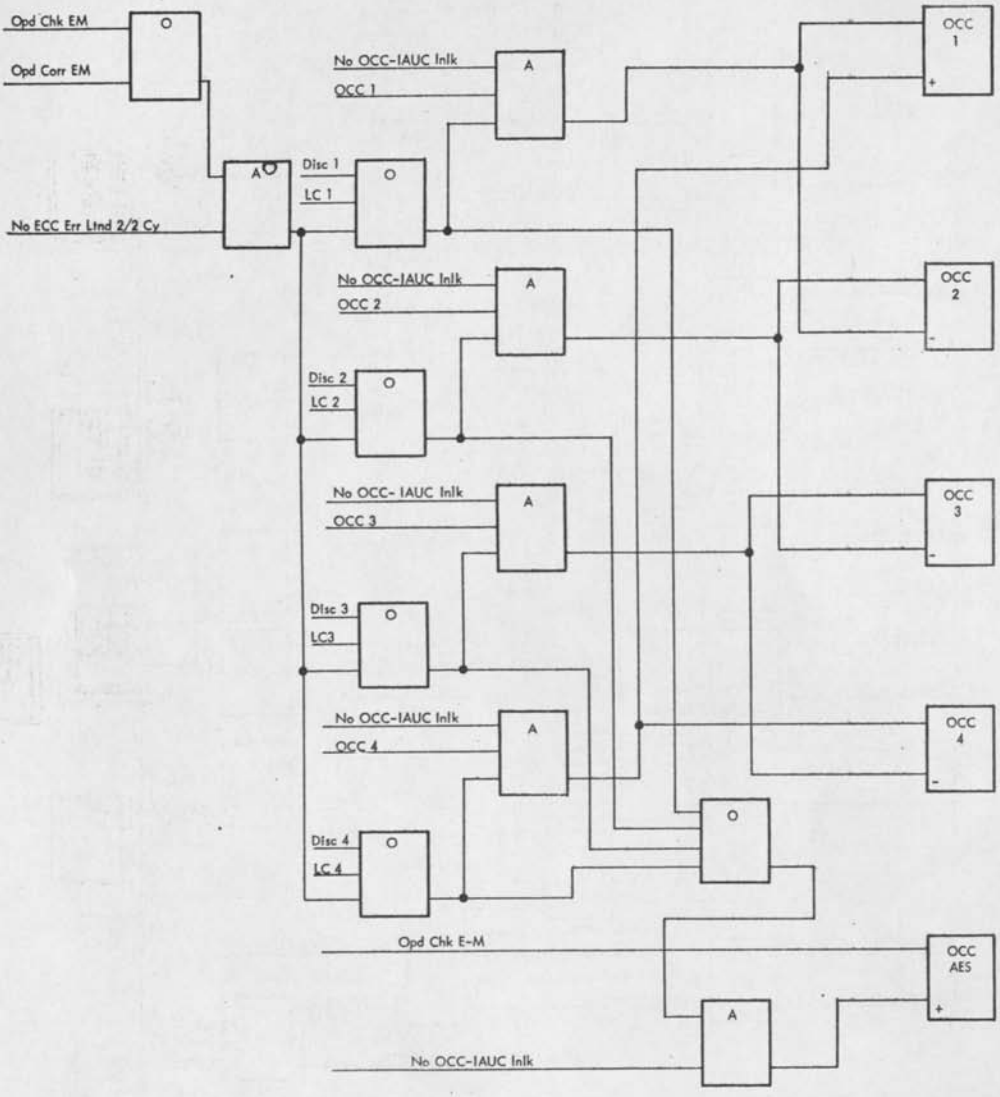


FIGURE 3.3-6. OCC COUNTERS AND OCC ADVANCE ENABLE SEQUENCER

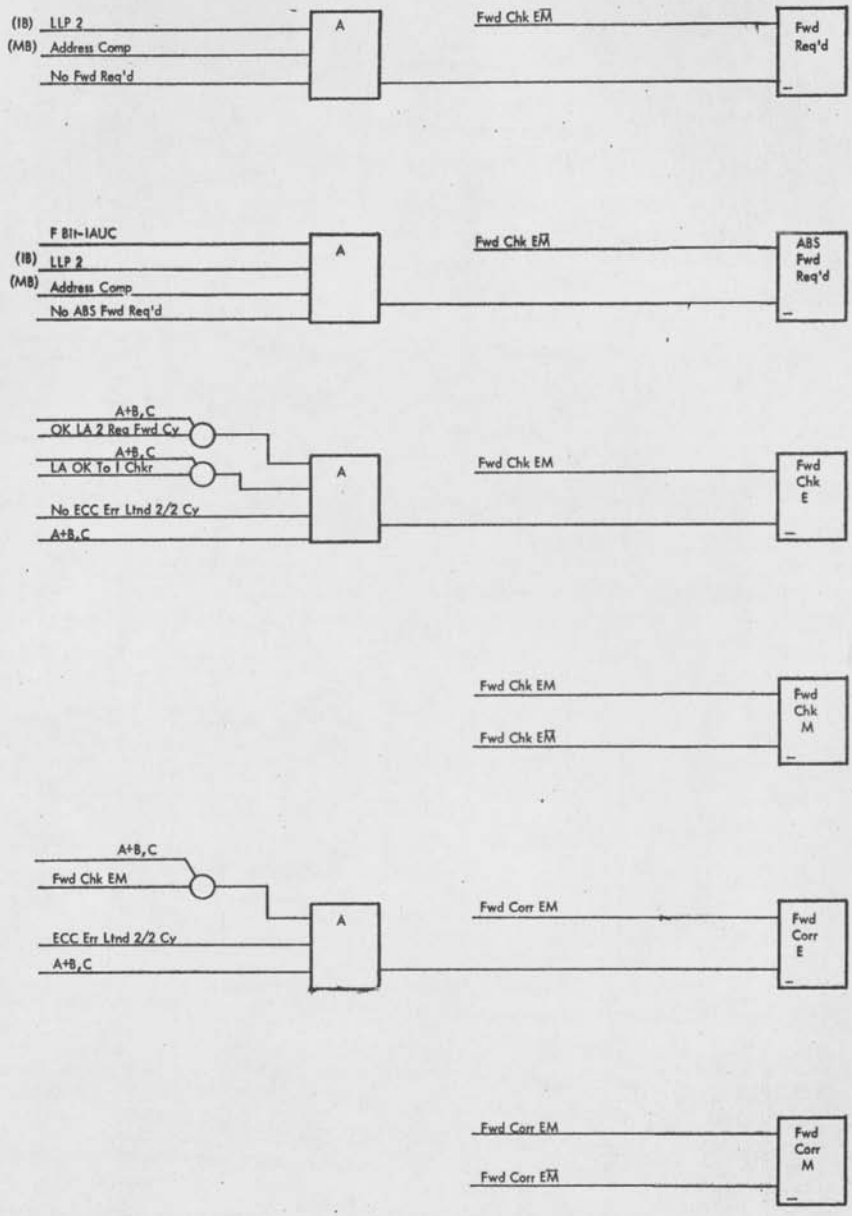
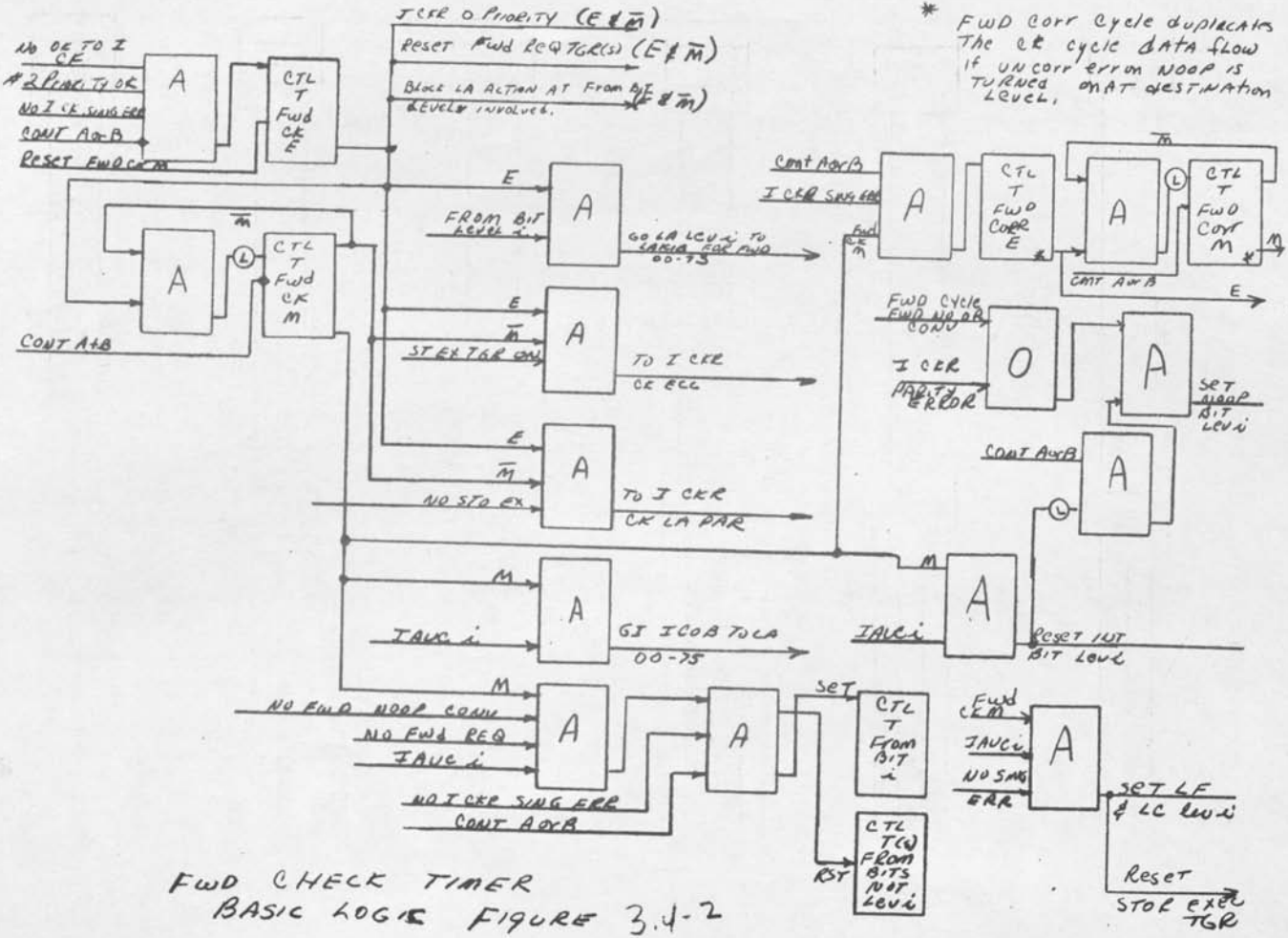
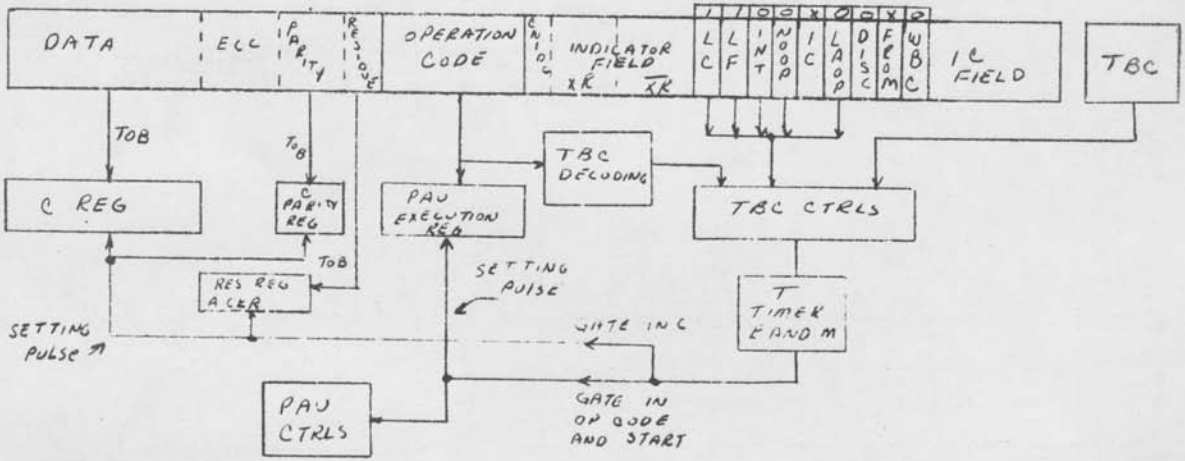


FIGURE 3.4-1. FWD REQUIRED; ABSOLUTE FWD REQUIRED; FWD CHK E, M; FWD CORR E, M



FWD CHECK TIMER
BASIC LOGIC FIGURE 3.4-2



TRANSFER TO PAU (NOT INTERNAL)
(NO STORE)
LOOKAHEAD TBC

FIGURE 4.1-1

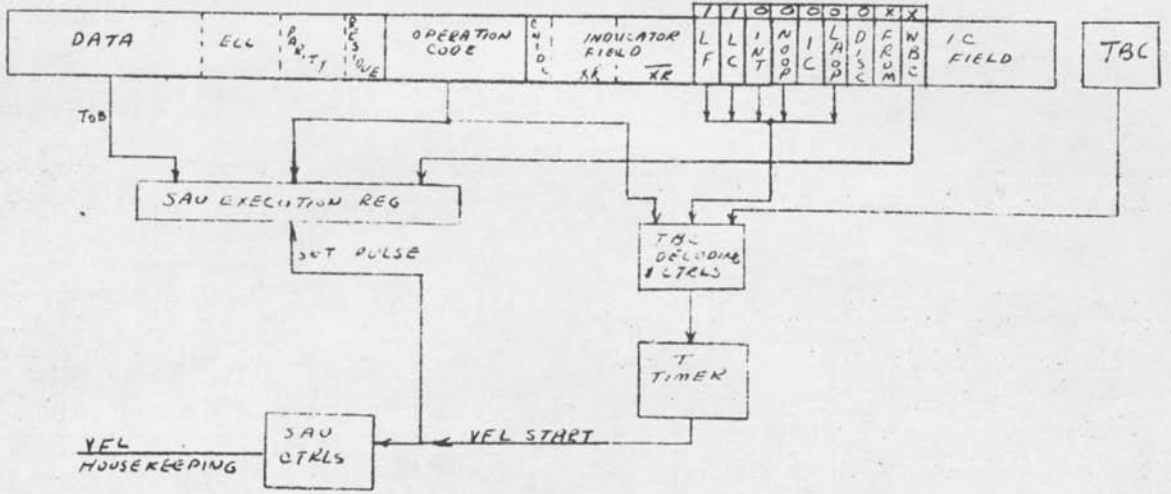
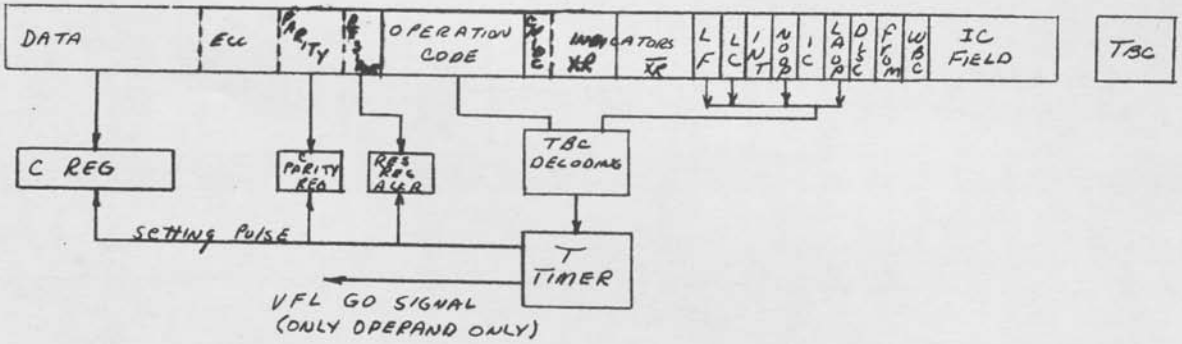
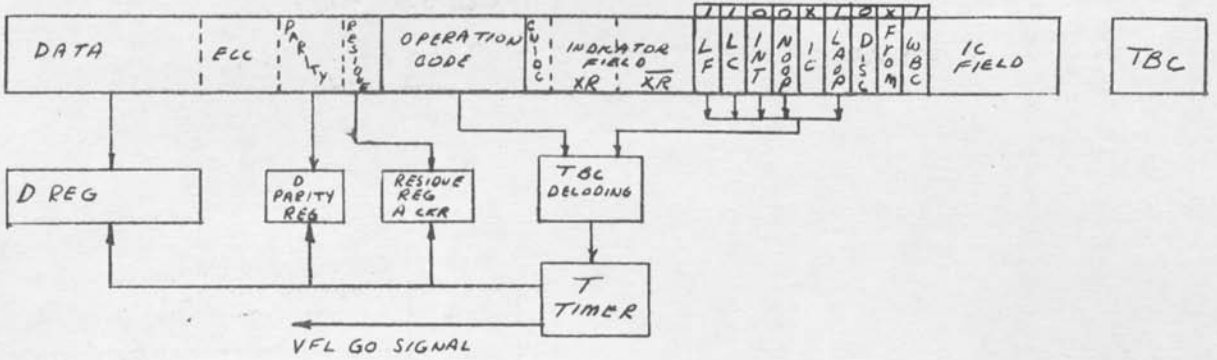


FIGURE 4.1-2. TBC VFL OP CODE LEVEL



FIRST
VFL ONLY OPERAND (NOT INTERNAL)
TBC ACTION

FIGURE 4.1-3.



SECOND VFL OPERAND (NOT INTERNAL)
TBC ACTION

FIGURE 4.1-4.

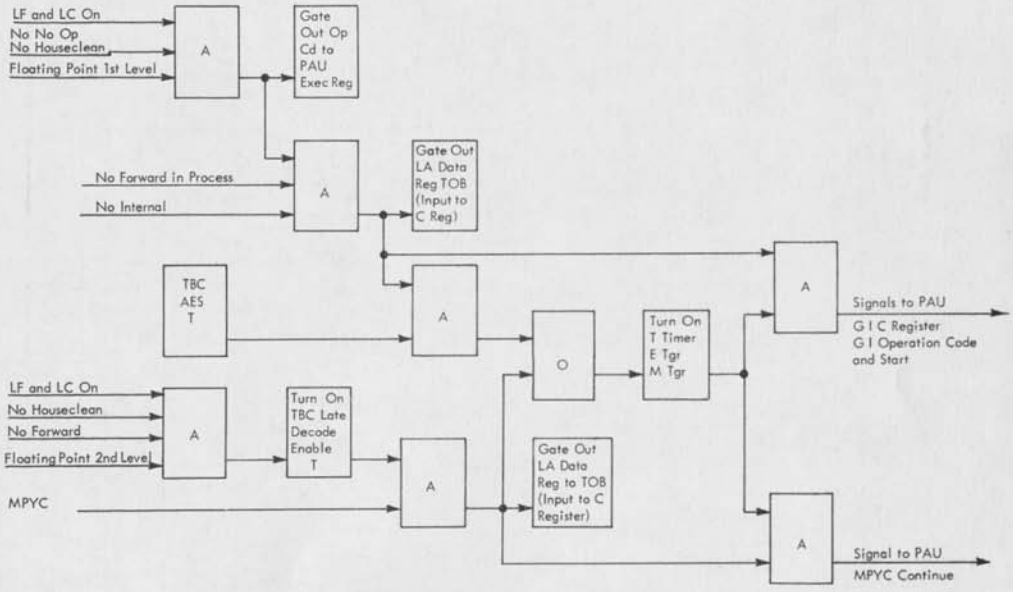


FIGURE 4.2-1. FLOATING POINT TBC ACTION

TBC DECODE

TBC DECODE LINES	OP CODE & TAG BIT CONDITIONS 0	5	6	7	8	9	LOOP	NOOP	I/O	LC	LF	DIS	HLT	BRC	IC	OTHER	REMARKS
																	NOTE! IN ALL CASES OP CODE BIT POS 1-4 ARE NOT USED
VFL OP CODE						1	0	0	0								ACTIVATES: TBC DEC COND-OP CO G A & VFL EN START - OP CO G B
VFL ONLY OPER	0	0	0	1	0	1	1	0	0								ACTIVATES: TBC DEC COND - TB TIMER ON MAR - ADV ON TB TIMER M GI TO C REG - SAV MAR
VFL 1ST OPERAT	0	0	0	0	1	1	1	0		0							ACTIVATES: TBC DEC COND - TB TIMER ON MAR - ADV ON TB TIMER M GI TO C REG - SAV MAR
VFL 2ND OPERAT	0	0	0	1	1	1	1	0		0							ACTIVATES: TBC DEC ADV ON TB TIMER M - SAV MAR - GI D REG
VFL MPYC OP CO	0	0	1	1	0	1	1	0	0	0							ACTIVATES: TBC DEC ADV ON TB TIMER M - GI TO C REG
VFL STORE BUS	0	1	0	0	0	1	1		0							TBC LATE DECODE EN	
FLP 1ST LEV					1	0	0	0	0		1	1	0	0			
FLP INT					1	0	0	0	0	0	1	1	0	0		NO FROM BIT OR FROM BIT FWD OR OP CODE	
FLP STORE 2ND LEV	0	0	1	0	1	1	1		0							TBC LATE DECODE EN	
COND FLP CONT	0	0	1	0	0	1	1	0	0								SPECIAL SIG USED ON MPYC INST
FLP INST REJECT	0	0	1	0	0	1	1	1								INTLK TBC-ABC	
BB OP CO LEV	1	0	0	0	1	1	1	0	0								ACTIVATES: TBC DEC COND - OP CO G A - VFL EN START - OP CO G B
BIN 1ST LEV		1	1	1	0	1	1	0	0								ACTIVATES: TBC DEC COND - OP CO G A - VFL EN START - OP EA G A
BIN INTERPT LEV	1	1	1	1	0	1	1	0	0								ACTIVATES: TBC DEC INT BIN LEV
ADV BR REC LEVA	1	0	0	1			1	1		0						NO BE SUCC NOT COND NO F TBC LOOP MAR REG INH LD	
ADV BR REC LEVB	1	0	0	1	1	1	1			0						TBC LOOP & NO BR REC INH LD	
I/O LEVEL ATA						0	0	0	0	0						INTLK SCC-ABC	ACTIVATES: TBC DEC COND - OP CO G A - OP CO G B - TB TIMER ON MAR
I/O BX					0	0	0	0	0	0						NOTE: BIT 7 = 0 = BX	
I/O HX					1	0	0	0	0	0						NOTE: BIT 7 = 1 = HX	
INT I BOX STORE	0	1	1	0	1	1	1	0	0						1		ACTIVATES: TBC DEC COND - TB TIMER ON MAR; GI TO C REG
SING ADV I BOX STORE INT	0	1	1	0	1	1	1	0	0							A BUS TIMER & INTLK TBC ABC	
COND TB TIMER	0	0	1	0	0	1	1	0	0								MPYC DECODING FL PT VFL OP CODE DECODING
" " "						1	0	0	0								
" " "	TBC DECODE VFL 2ND OPER INT OR BB OP CO LEV OR BIN FIRST LEV																
COND TB GO	0	0	1	0	0	1	1	0	0								MPYC DECODING FL PT VFL OP CODE DECODING
" " "						1	0	0	0								
" " "	0	0	0	1	0	1	1	0	0							TBC DEC I/O LEV	
" " "	TBC DECODE VFL 1ST OPER INT OR VFL 2ND OPER INT OR VFL MPYC OP CO																
" " "	TBC DECODE BB OP CO LEVOR BIN FIRST LEV																
" " "	TBC DECODE INT I BOX STORE																
" " "	TBC DECODE I/O LEVEL																
COND GI RESTORE COND TB TIM MPYC ADV NOOP TC	0	0	1	1	0	1	1	0	0	0							
								1							0		
ADV PX	0	0	0	1	0	0	1		0							TBC LATE DEC EN	
COND TBG DEC LEV 1					1	0										TBC LATE DEC EN	

FIGURE 4.2-2.

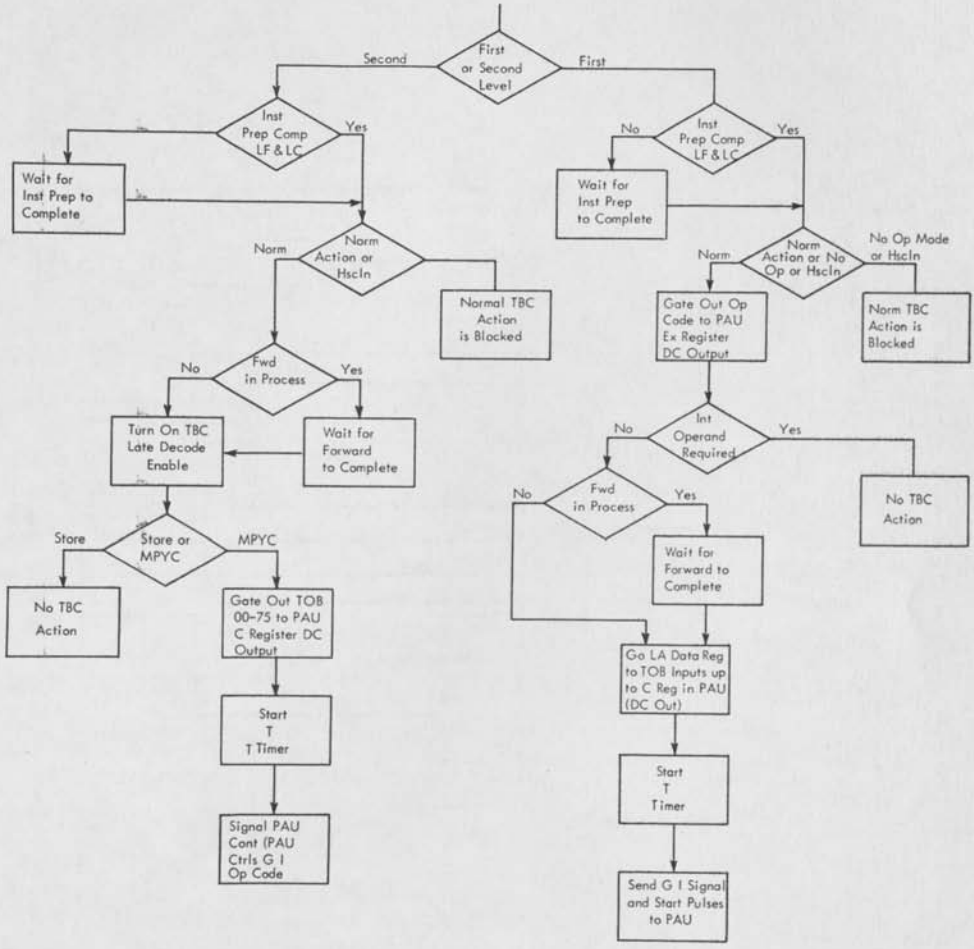


FIGURE 4.2-3. FLOATING POINT TBC ACTION; TBC DECODE FLOATING POINT (OP 8 NO OP 9)

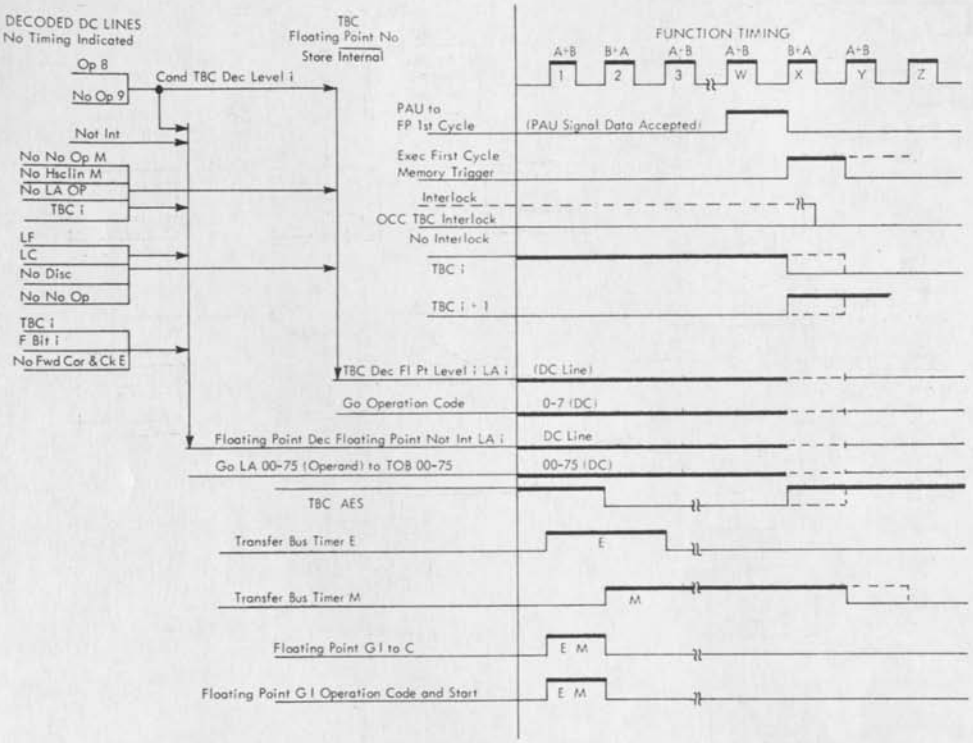
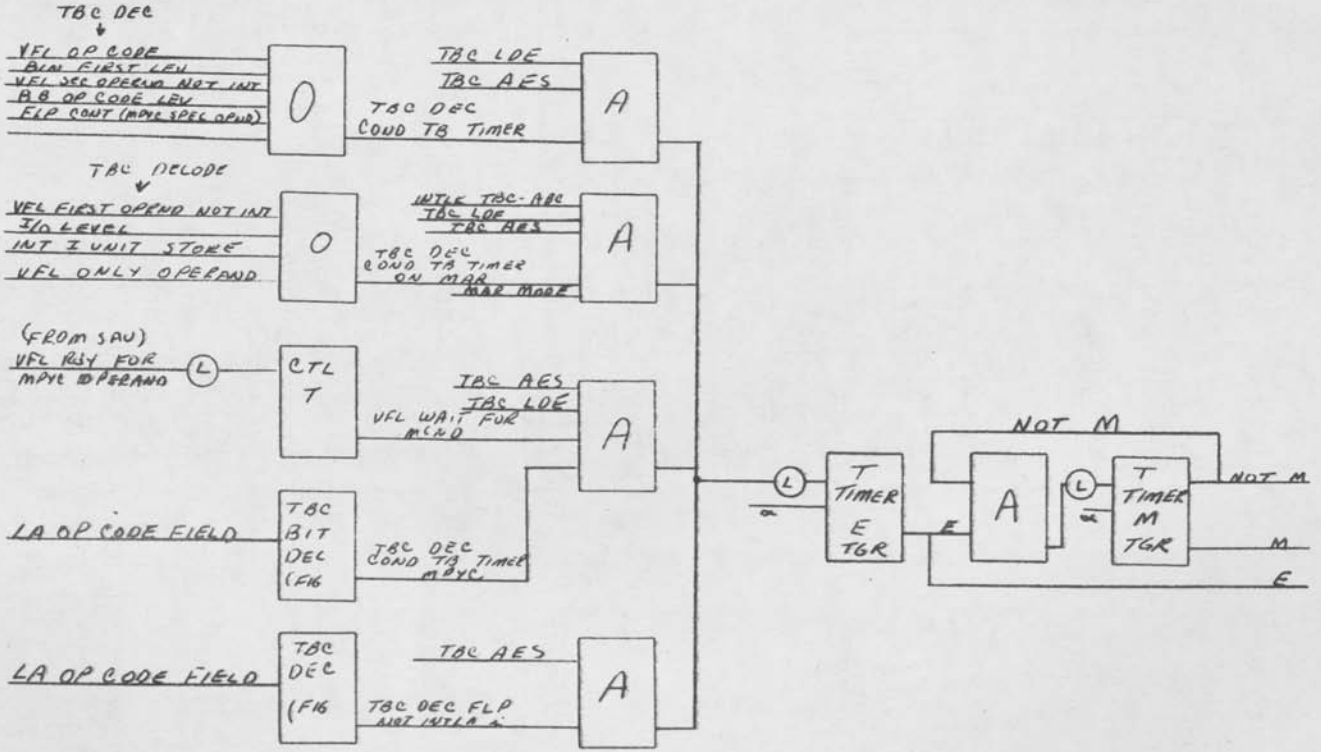

DECODED DC LINES
No Timing Indicated
Op 8
No Op 9
Cond TBC Dec Level i
TBC
Floating Point No
Store Internal
FUNCTION TIMING
A+B
B+A
A+B
A+B
B+A
A+B
1
2
3
W
X
Y
Z
Not Int
No No Op M
No Hsclin M
No LA OP
TBC i
LF
LC
No Disc
No No Op
TBC i
F Bit i
No Fwd Cor & Ck E
PAU to
FP 1st Cycle
(PAU Signal Data Accepted)
Exec First Cycle
Memory Trigger
Interlock
OCC TBC Interlock
No Interlock
TBC i
TBC i + 1
TBC Dec Fl Pt Level i LA i
(DC Line)
Go Operation Code
0-7 (DC)
Floating Point Dec Floating Point Not Int LA i
DC Line
Go LA 00-75 (Operand) to TOB 00-75
00-75 (DC)
TBC AES
Transfer Bus Timer E
E
Transfer Bus Timer M
M
Floating Point G I to C
E M
Floating Point G I Operation Code and Start
E M

FIGURE 4.2-3A.



TBC
STARTING THE T TIMER

FIGURE 4.2-4.

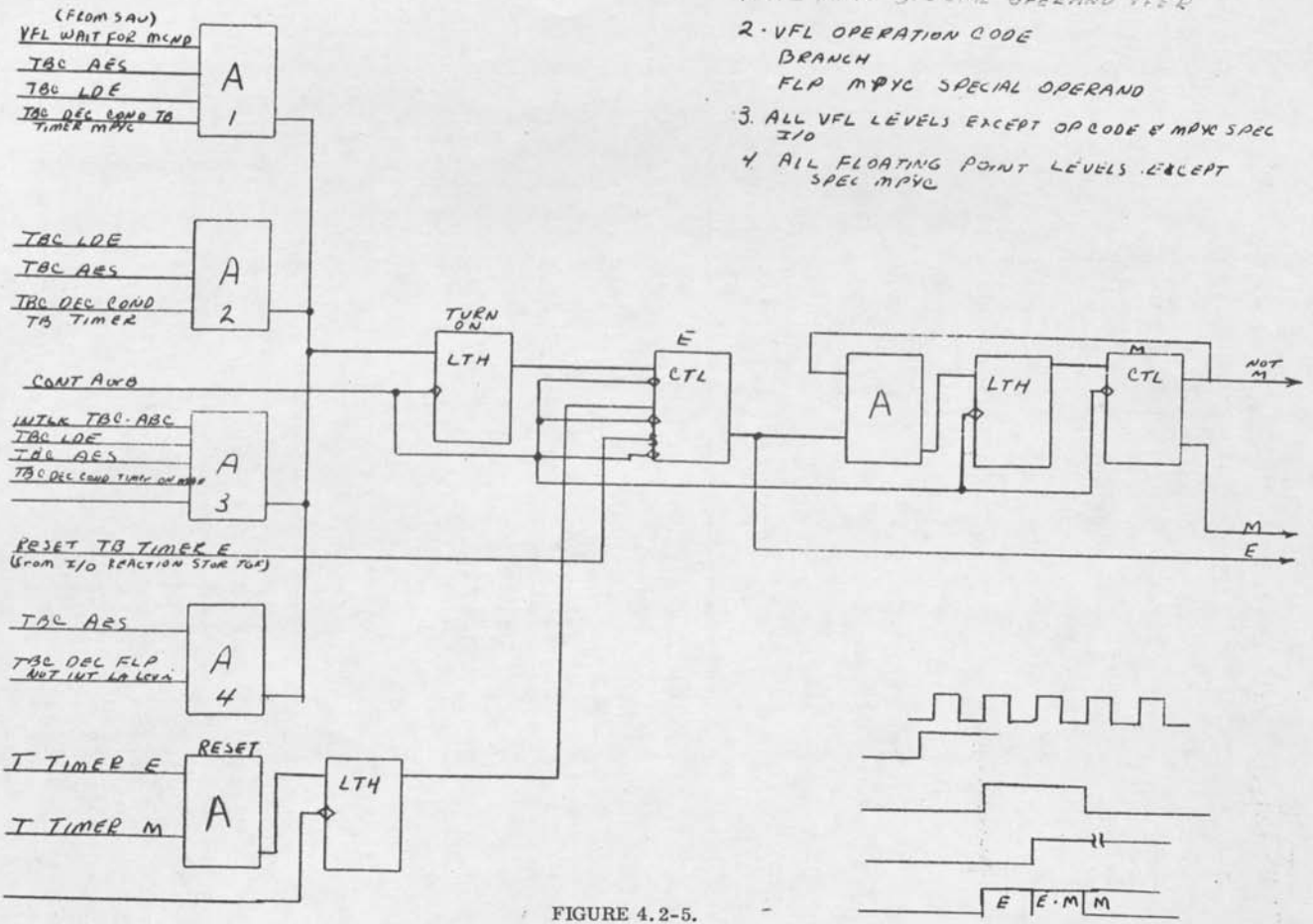

1. VFL MPYC SPECIAL OPERAND VFER
2. VFL OPERATION CODE
BRANCH
FLP MPYC SPECIAL OPERAND
3. ALL VFL LEVELS EXCEPT OP CODE & MPYC SPEC I/O
4. ALL FLOATING POINT LEVELS EXCEPT SPEC MPYC
(FROM SAU)
VFL WAIT FOR MCND
TBC AES
TBC LDE
TBC DEC COND TB TIMER MPYC
A 1
TBC LDE
TBC AES
TBC DEC COND TB TIMER
A 2
CONT AWB
WILK TBC-ABC
TBC LDE
TBC AES
A 3
RESET TB TIMER E
(From I/O REACTION STOR FLP)
TBC AES
TBC DEC FLP
A 4
T TIMER E
T TIMER M
RESET
A
LTH
TURN ON
LTH
E
CTL
A
LTH
M
CTL
NOT M
M
E
E
E·M
M

FIGURE 4.2-5.

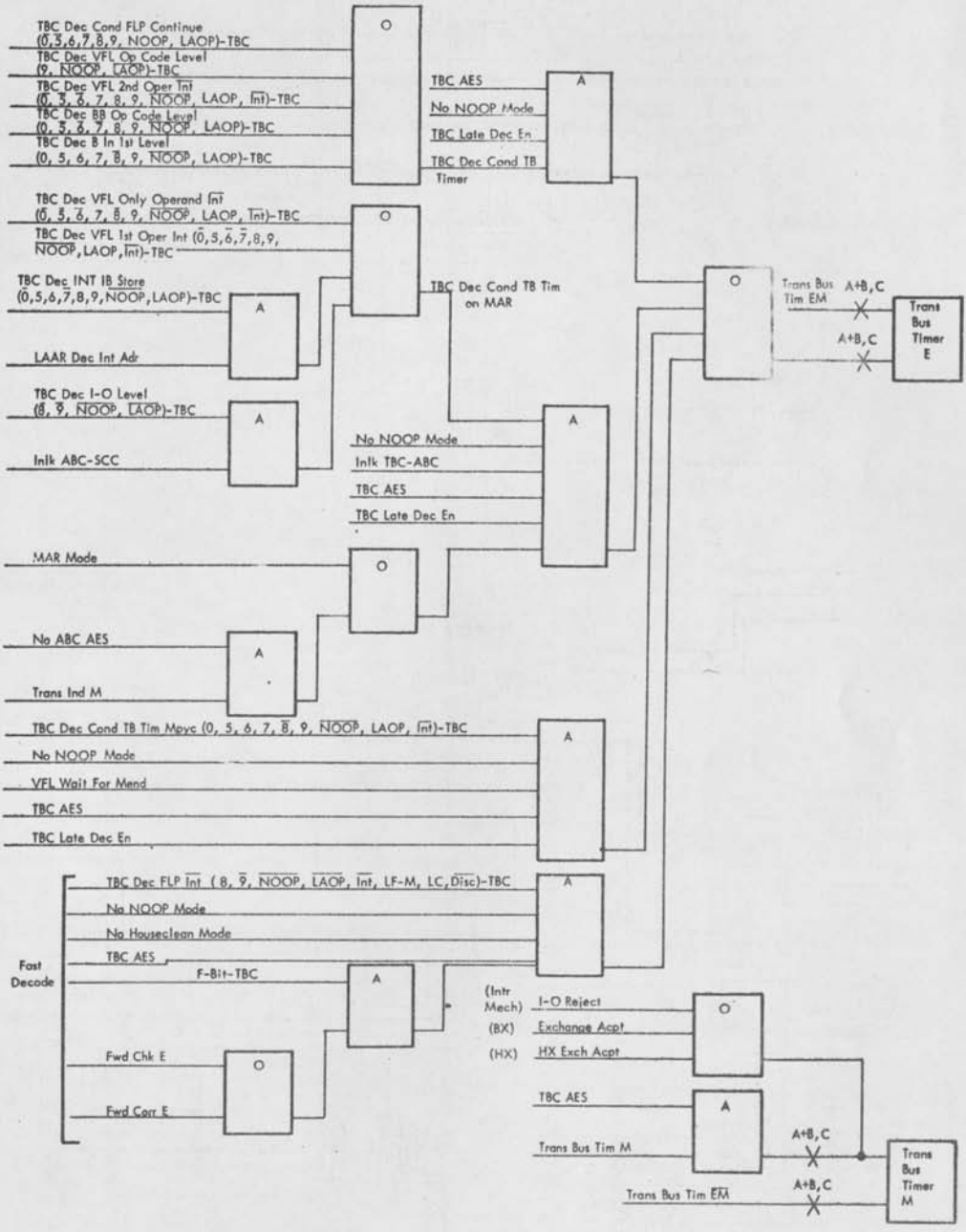


FIGURE 4.2-6. TRANSFER BUS TIMER E AND M

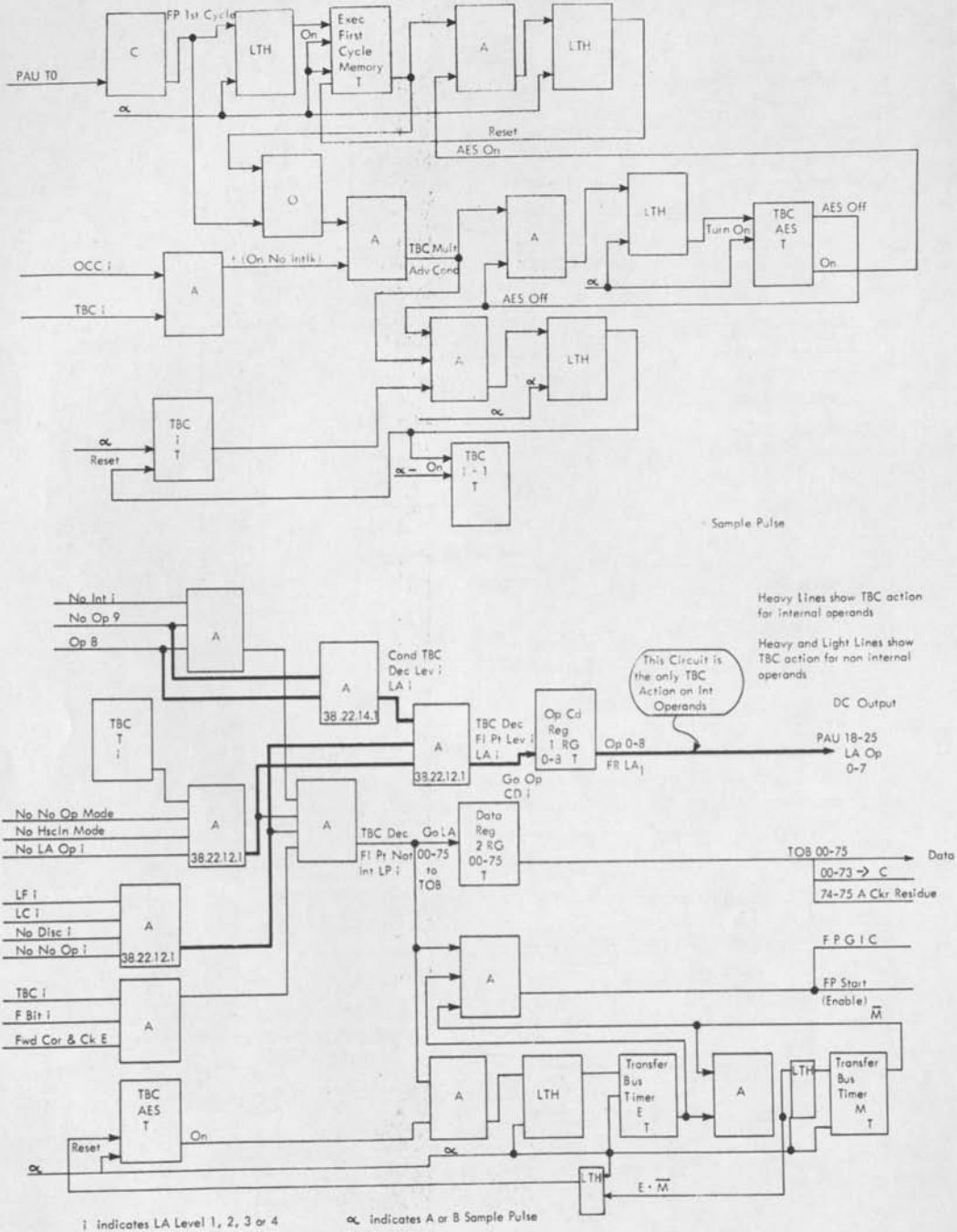
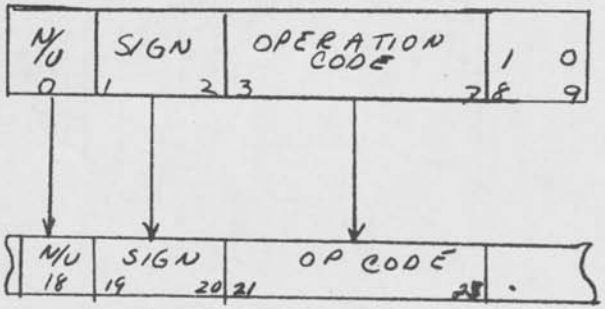


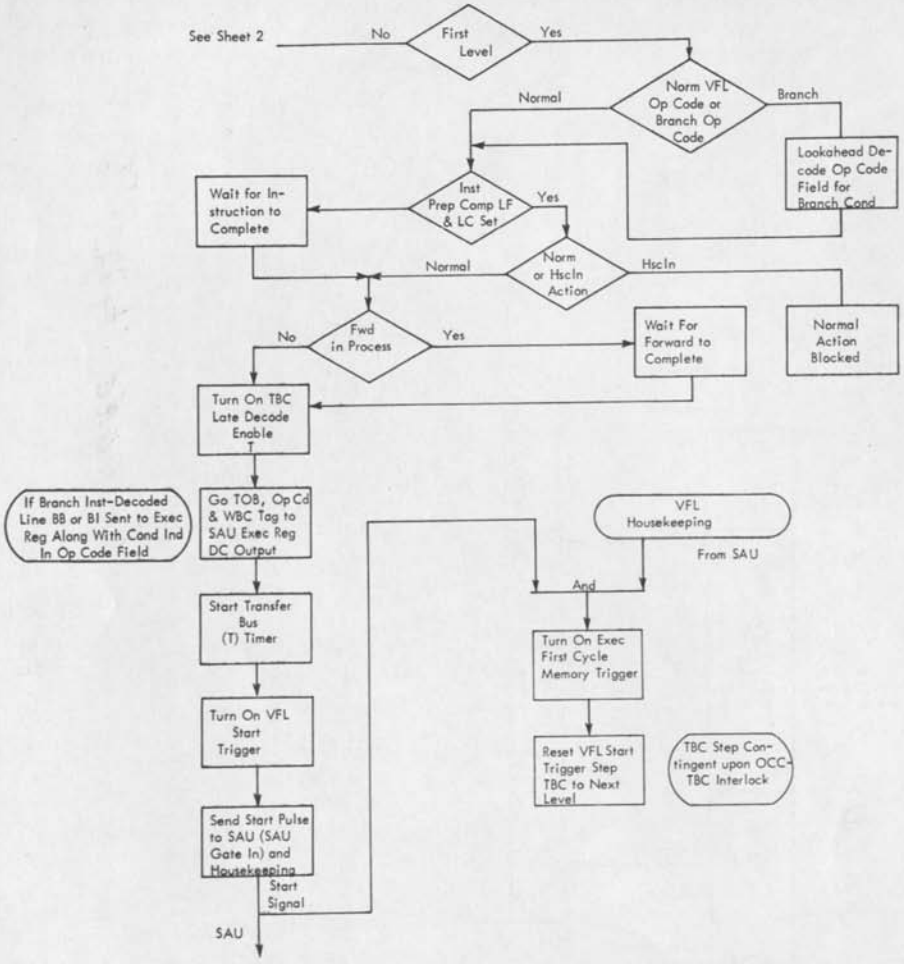
FIGURE 4.2-7. FLP OPERATIONS, TBC COUNTER CONTROL

LA OP CODE FIELD



PAU EXEC REG

FIGURE 4.2-8



Sheet 1

FIGURE 4.3-1. VARIABLE FIELD LENGTH (VFL)
TBC ACTION
TBC DEC VFL OP 9

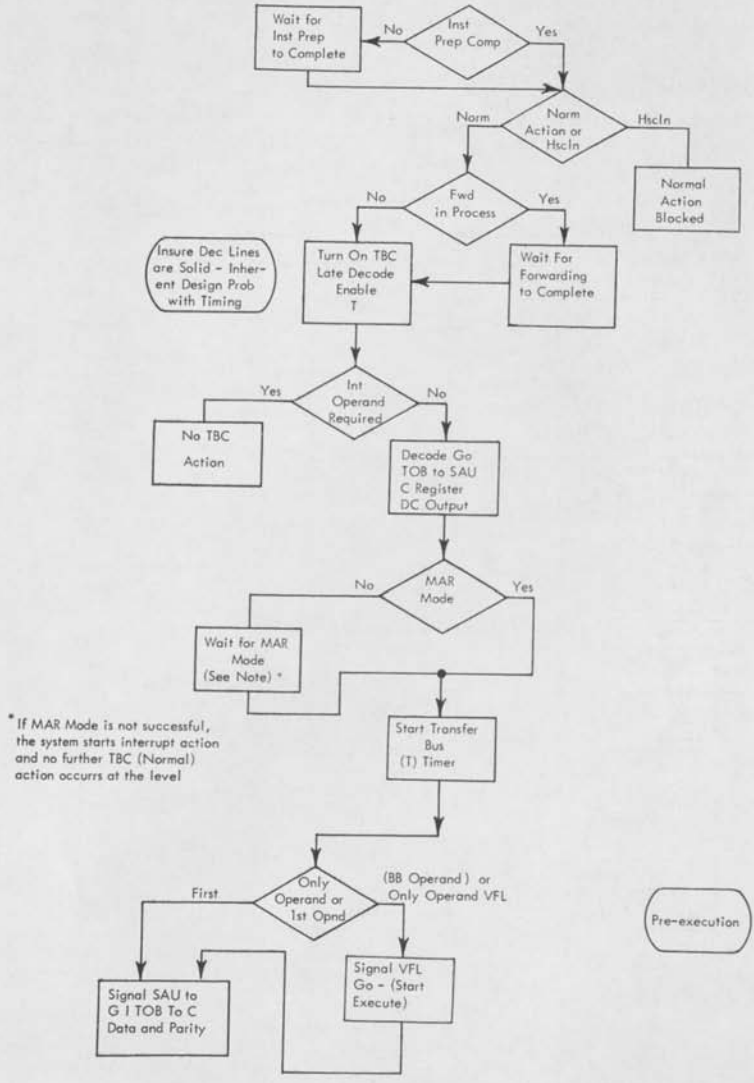


FIGURE 4.3-2. VFL TBC ACTION; SECOND LEVEL

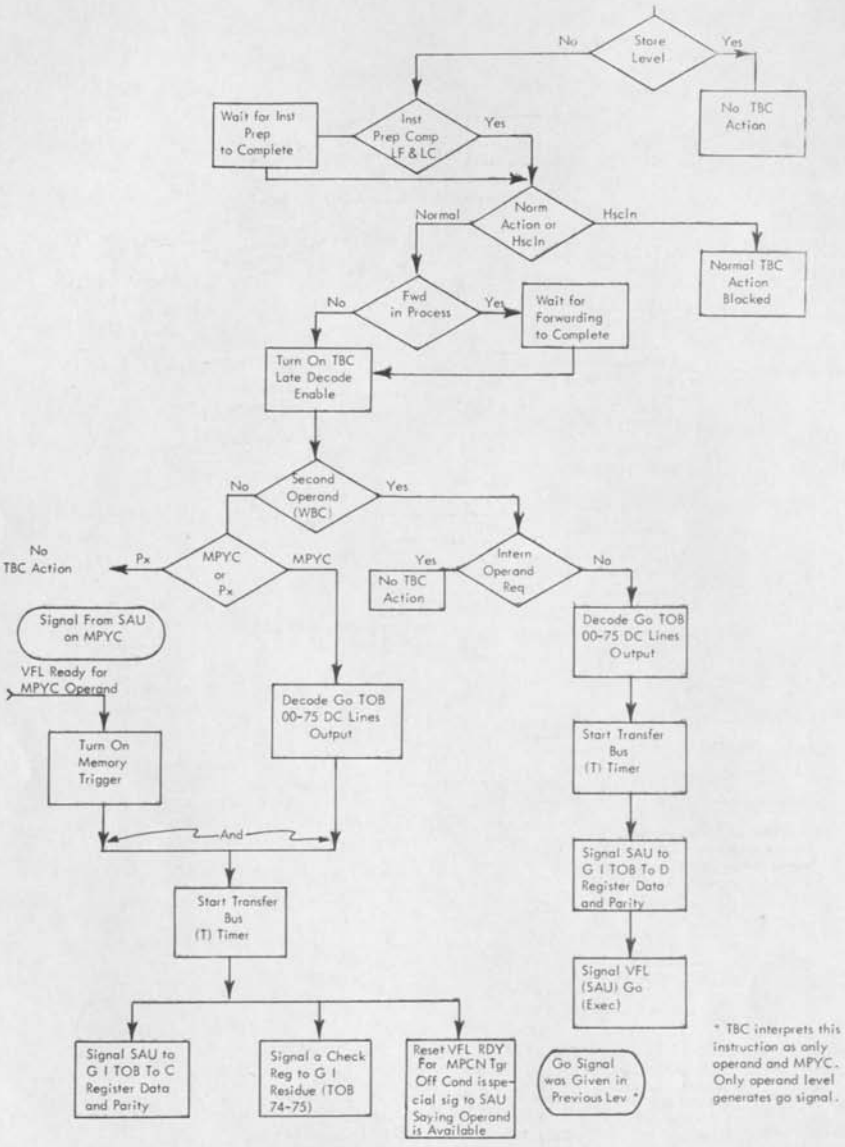


FIGURE 4.3-3. VFL TBC ACTION, 3rd LEVEL

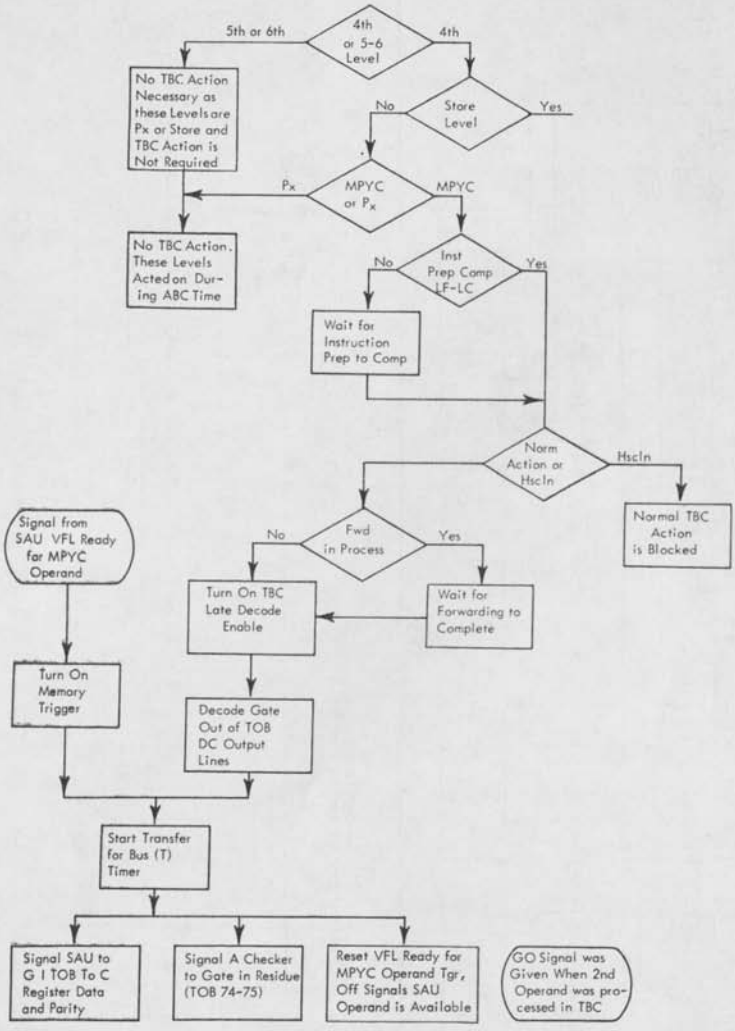


FIGURE 4.3-4. VFL TBC ACTION 4th - 5th - 6th LEVEL

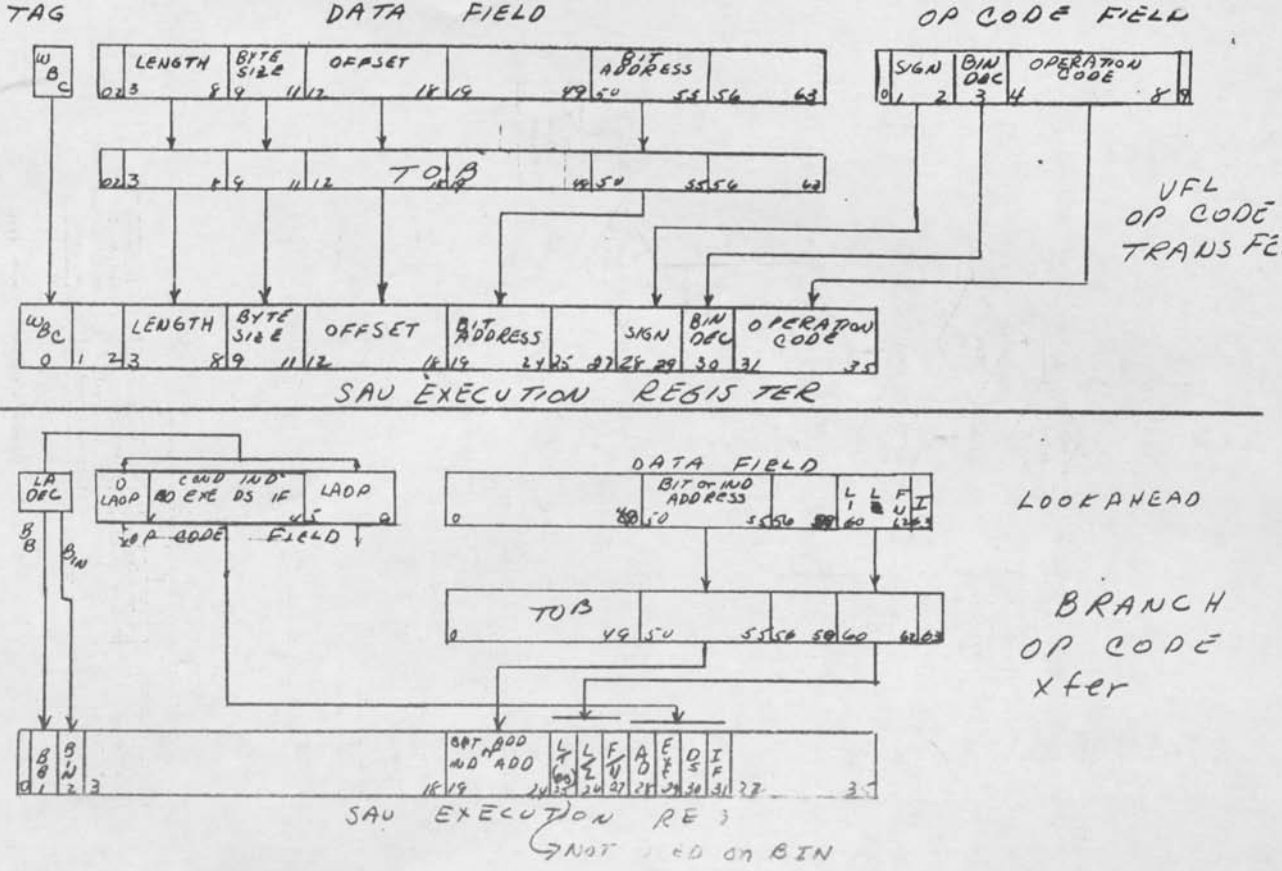

TAG
DATA FIELD
OP CODE FIELD
WBC
LENGTH
BYTE SIZE
OFFSET
BIT ADDRESS
SIGN
BIN DEC
OPERATION CODE
TOB
UFL OP CODE TRANS FER
SAU EXECUTION REGISTER
LA OPC
COND MOD EXE DS IF
LADP
OP CODE FIELD
DATA FIELD
BIT or IND ADDRESS
LOOKAHEAD
BRANCH OP CODE xfer
SAU EXECUTION REG
NOT USED ON BIN

FIGURE 4.3-5

COMPANY CONFIDENTIAL

IBM CUSTOMER ENGINEERING

Preliminary Instruction Text

IBM 7030 DATA PROCESSING SYSTEM

LOOKAHEAD UNIT
BOOK B

The information presented in this preliminary instruction text is based on the electrical and mechanical status of the 7030 system as of December 15, 1960. The contents of this text has not been approved by the Engineering or Patent Departments and has not been edited for final printing. Distribution of this text shall be restricted to IBM employees authorized by the nature of their duties to receive such information.

LOOKAHEAD UNIT, BOOK B

CONTENTS

5.0.00 INTERRUPT TEST AND INSTRUCTION EXECUTION (ABC)

5.1.00 General Description

The execution phase of an instruction is defined as the modification of an addressable register. Before any instruction is allowed to alter the contents of any addressable register, the system must be assurred that the preceding instruction completed without an error. The assurance is in the form of MAR MODE which is a result of a reliable interrogation of the interrupt mechanism at the completion of the preceding instruction. At the completion of each instruction it is the responsibility of lookahead to interrogate the status of the interrupt mechanism. For lookahead purposes, the instruction is not considered complete until lookahead is certain that all units associated with the instruction have entered all abnormal results into the main indicator register. This action insures that any indicators affected by the preparation and execution of the instruction are set in the main indicator register. To make sure all units have reported their results to the indicator register, lookahead has a group of test triggers whose status is determined by the units involved in the instruction. The test triggers and their functions are listed below.

1. Lookahead Normal Indicator Test

This trigger records the fact that the lookahead has entered all indicators associated with the instruction. Specifically these are the indicators affected during instruction preparation.

2. Execution Unit Test

This trigger records the fact that the execution unit involved has entered all indicators associated with the execution of the instruction. Any indicators affected by the execution of the instruction in the execution unit are entered in the in-

dicator register directly from the execution unit.

3. A Checker Indicator Test

This trigger records the fact that the possible error detected during an SCC (store) check and check bit conversion cycle for a store type instruction has been entered into the indicator register. Unlike the other test triggers, the store check trigger is normally on and is reset as an ABC function only when it is evident that a store check cycle follows.

Each trigger must be set following each arithmetic unit instruction regardless of whether any error condition actually occurs.

The triggers just discussed are the test triggers for all arithmetic operations and are not applicable to non-arithmetic instructions. Because the interrupt mechanism must be interrogated following any instruction, an alternate method isused to determine the final indicator setting for non arithmetic instructions or No OP'ed arithmetic instructions. A lookahead No-Op Indicator Test Trigger records the last indicator setting for those instructions not requiring the arithmetic units. Because lookahead is the only unit involved, the interrupt interrogation is allowed on the basis of the Lookahead No-Op Test and the Store Check Indicator Test Triggers only.

It is important to remember that the memory action of the interrupt line due to the coincidence of the test triggers resulting from instruction N define the action to be taken on instruction (N+1). For speed purposes PAU has a duplicate set of test triggers and depends upon lookahead for assurance that it has transferred indicators (La Normal indicator test trigger) and that the instruction does not require a store (store check indicator test trigger). In certain cases

it is the responsibility of lookahead to properly sequence the PAU testing as well as the lookahead tests. These special actions are discussed as the situations are discussed later in the manual.

The coincidence of the arithmetic test triggers or the non arithmetic test triggers allows lookahead to determine the status of the interrupt mechanism. The result of the comparison of the indicator register and the mask register comes to lookahead as interrupt or no interrupt. The interrupt line conditioned by the coincidence of the test triggers, turns on the interrupt next instruction trigger or if the no interrupt line is active it turns on the MAR MODE trigger. Either trigger defines the action for the following instruction. The interrupt next instruction causes lookahead to initiate recovery action and the MAR MODE trigger allows the modification of addressable registers (execution) by the following instruction.

The action of the Arithmetic Bus Counter (ABC) depends upon instruction preparation being complete (LF and LC) and MAR MODE. With both conditions present the ABC action is started. The primary function of the ABC at any level is to transfer the lookahead indicator field to the main indicator register and to transfer the lookahead IC field to the IC buffer. These two functions are timed with the Indicator Transfer Timer. The timer is started by proper ABC decoding, MAR MODE and instruction preparation being complete. The indicators being transferred pertain to the instruction (N) presently being executed. (The MAR MODE is a result of instruction N -1).

The IC field is placed in the IC buffer to be saved in the event the current instruction causes an interrupt. If an interrupt does occur, the IC buffer contents is sent to the instruction unit to provide them with an address to re-start with

following any correction routines resulting from the interrupt. Remember the IC address located in the buffer is one more than the instruction being executed. For example assume instruction N is being executed. The following facts are known at this time.

1. MAR MODE is present from the result of testing instruction (N-1)
2. The address is the IC field is (N+1) as loaded from the instruction unit. During the ABC function the indicator transfer timer is started. The timer function cause three primary occurrences.

    1. Indicator field of the ABC level is transferred to the main indicator register.
    2. IC field (N+1) is tranferred to the IC buffer. The IC buffer now contain address (N+1).
    3. MAR MODE is reset.

As the instruction is being executed the MAR MODE test triggers start being affected. When all units have reported (execution of instruction N is complete), the test triggers allow the interrogation of the interrupt system. The interrupt system at this time indicates either no interrupt or interrupt. With the test triggers indicating instruction completion the interrupt line, if active, causes the interrupt next instruction to come on. Before the execution of the next instruction can occur, however, the interrupt next instruction trigger starts a recovery operation. Basically the recovery action is a houseclean request to the instruction unit. When the instruction unit finishes its recovery cycle they inform lookahead and lookahead proceeds to recover . One of the lookahead recovery actions is to send pseudo stores back to the instruction unit to back

date the index core storage. Also the IC buffer contents is returned to give an instruction address for the main program restart following any corrective routines. Remember the IC buffer during the interrupt procedures contains the address (N+1) and the interrupt was caused by the result of instruction (N). Any corrective routines apply to instruction N and at their completion, the (N+1) address is used to restart the main program, thereby maintaining the proper program sequence. If the interrupt line resulting from instruction N is not active, then the MAR MODE trigger is turned on. With the MAR MODE trigger on the execution of instruction N+1 occurs. Druing the execution of instruction N+1, the IC field (containing address (N+2)) is transferred to the IC buffer. The buffer now contains address (N+2) which is the restart address for the instruction unit if instruction (N+1) interrupts. At the time the IC field is transferred to the buffer, the lookahead indicator field is transferred to the main indicator field register. Both of these transfer functions are controlled by the indicator transfer timer during ABC time. The transfer of indicators and the IC field transfer are the primary functions of the ABC counter. Secondary functions of the ABC counter are control of internal operations (fetches and stores) and receiving data from the execution units in preparation for an external or index core storage store. Any internal operation is so noted by the internal tag bit being set at the level being processed. The ABC decoding determines whether the internal operation is a fetch or a store. Because any internal fetch or store operation was loaded as a type 3 load, the LF and LC tag bits were set unconditionally thereby allowing the normal ABC function of transfer of indicators and transfer of IC field to buffer (IC tag on). When the required operand for

an arithmetic operation is requested from an internal register, the ABC times the data transfer on the arithmetic bus to the C or D register. All internal registers are tied to the arithmetic bus and therefore the internal register word required by SAU or PAU does not appear in the lookahead data field.

The ABC decoding for an internal fetch requests starts an A Bus timer to time the data transfer. The address of the required internal register is contained in the LAAR which was set during the lookahead load. Referring to the TBC action, when internal operations were called for at a level, the TBC did not start the T timer to transfer the data. This is true at any level tagged with the internal tag bit. Also in this situation the TBC did not step because the stepping depends upon a signal from the execution unit informing lookahead that they have received the data. The A bus timer, when started, times the data transfer and generates the required signals to gate the data in. In FLP operation, the gate in signal also includes the operation code field which is still being decoded by the TBC.

In a VFL operation the operation code is gated in normally because it is never concerned with internal operations. All operand requirements for VFL follow in succeeding levels and never appear at the operation code level. So in a VFL operation, the operation code is transferred normally by the TBC, but any succeeding levels requiring internal operations are dependent upon the ABC. The ABC starts the A bus timer and times the data flow to the C or D register and generates the necessary signals to gate the data in.

In a FLP instruction, (Figure 5.1-1) an operand appears at the same level as the operation

code. If the level is tagged with an internal bit, the transfer of the operation code and data to the execution unit automatically became a function of the ABC. Because the TBC was prevented from stepping the d. c., gate out of the operation code field is still active from the TBC. When the A bus timer is started the data from the internal register, designated by the LAAR, is timed to the C register and the necessary signals are sent to PAU to gate in the operation code field, C register and start. The execution register input was a result of the TBC setting and the signal to set the execution register was in the form of the A bus timer. When the data has been accepted by PAU or SAU a return signal from SAU or PAU allows the TBC to advance to the next level.

If the internal fetch is requested by the instruction unit, the specified internal register contents are gated out via the A checker bus in a normal manner, but instead of placing the data in the C-D register, the data is placed in the lookahead data field where, during SCC action, it is sent to the Y register in the instruction unit. The complete operation is defined by the internal tag bit and the ABC decoding. The LAAR contains the address of the required internal register. By starting the A bus timer, the necessary signals are generated to route the data from the internal register to the lookahead data field via the arithmetic bus.

The ABC function for store levels consists of completing all internal store operations while in stores to external storage and index core storage the ABC only has partial action. When an internal store is required during an arithmetic operation, the ABC times the data transfer via ~~to~~ the arithmetic bus to the internal register designated by the LAAR. The A bus timer is started by the ABC when the decoding is such that a store is required and the internal tag is on at this level.

Because the result of an arithmetic operation, as far as lookahead is concerned, appears in the C-D register, the operation consists of routing the data from the C-D register to the internal register. The data does not appear in lookahead, but all data control functions are derived in lookahead. After the TBC function at a level, lookahead waits for a signal from the execution unit saying that the data to be stored is latched on the arithmetic bus (in the C-D register). With the arrival of that signal the A bus timer is started to time the data transfer from the C-D register to the A checker to the internal register designated by the LAAR. The operation also resets the LAAR busy trigger because the store is complete. With the LAAR busy trigger off, forwarding can occur if the forwarding conditions are met.

When the store operation is to external storage or index core storage from an arithmetic operation, the lookahead action is similar to an internal store with the end result being the data being placed in the lookahead data field. The initial lookahead action is the same (lookahead waits for a signal from the arithmetic unit that the data is latched on the arithmetic bus). The A bus timer is started to time the data transfer from the C-D register to the A checker to the lookahead data field. Also the A bus timer resets the LF tag bit and the store check test trigger. Because the data is in the lookahead data field, the store is not complete until the data is placed in the location designated by the LAAR. The ABC action however is complete. The final storing of the data is a function of the SCC counter and is described in section 6.0.00. The signal to allow to SCC to function is the fact that the LF tag bit is reset.

code. If the level is tagged with an internal bit, the transfer of the operation code and data to the execution unit automatically became a function of the ABC. Because the TBC was prevented from stepping the d.c., gate out of the operation code field is still active from the TBC. When the A bus timer is started the data from the internal register, designated by the LAAR, is timed to the C register and the necessary signals are sent to PAU to gate in the operation code field, C register and start. The execution register input was a result of the TBC setting and the signal to set the execution register was in the form of the A bus timer. When the data has been accepted by PAU or SAU a return signal from SAU or PAU allows the TBC to advance to the next level.

If the internal fetch is requested by the instruction unit, the specified internal register contents are gated out via the A checker bus in a normal manner, but instead of placing the data in the C-D register, the data is placed in the lookahead data field where, during SCC action, it is sent to the Y register in the instruction unit. The complete operation is defined by the internal tag bit and the ABC decoding. The LAAR contains the address of the required internal register. By starting the A bus timer, the necessary signals are generated to route the data from the internal register to the lookahead data field via the arithmetic bus.

The ABC function for store levels consists of completing all internal store operations while In stores to external storage and index core storage the ABC only has partial action. When an internal store is required during an arithmetic operation, the ABC times the data transfer via the arithmetic bus to the internal register designated by the LAAR. The A bus timer is started by the ABC when the decoding is such that a store is required and the internal tag is on at this level.

Because the result of an arithmetic operation, as far as lookahead is concerned, appears in the C-D register, the operation consists of routing the data from the C-D register to the internal register. The data does not appear in lookahead, but all data control functions are derived in lookahead. After the TBC function at a level, lookahead waits for a signal from the execution unit saying that the data to be stored is latched on the arithmetic bus (in the C-D register). With the arrival of that signal the A bus timer is started to time the data transfer from the C-D register to the A checker to the internal register designated by the LAAR. The operation also resets the LAAR busy trigger because the store is complete. With the LAAR busy trigger off, forwarding can occur if the forwarding conditions are met.

When the store operation is to external storage or index core storage from an arithmetic operation, the lookahead action is similar to an internal store with the end result being the data being placed in the lookahead data field. The initial lookahead action is the same (lookahead waits for a signal from the arithmetic unit that the data is latched on the arithmetic bus). The A bus timer is started to time the data transfer from the C-D register to the A checker to the lookahead data field. Also the A bus timer resets the LF tag bit and the store check test trigger. Because the data is in the lookahead data field, the store is not complete until the data is placed in the location designated by the LAAR. The ABC action however is complete. The final storing of the data is a function of the SCC counter and is described in section 6.0.00. The signal to allow to SCC to function is the fact that the LF tag bit is reset.

Whenever the SCC enters a level and the LF tag bit is off, the SCC has a store function to perform. If the LF tag is on when the SCC steps into a level, it signifies no storing action and the SCC merely steps to the next level. The store check test trigger is reset to prevent a MAR Mode test until the SCC finishes the storing action at which time the store check trigger is turned on to allow the MAR Mode test to be completed.

The ABC action taken at branch recovery levels, other than the normal transfer of indicators, varies depending upon the branch response from SAU. There are three distinct responses from SAU about the branch success or failure test. They are:

1. Branch unsuccessful. There is no action necessary when this response is received from SAU. No action is required because SAU found the branch condition was not satisfied. Because the instruction unit assumes that a BB and a BI will not branch, they continue processing instructions sequentially. When SAU tests for the branch and proves the branch condition is not satisfied it means the instruction unit assumption was correct and therefore the ABC steps to the next level and performs no action at the recovery level.

2. Branch successful — Conditional Indicators. This response means that the branch condition was satisfied, but one of the conditional indicators that apply to the "branch to" address is on thereby causing an unsuccessful branch condition. The conditional indicators are buffered in the operation code field of the operation code level of the instruction and during the TBC action are placed in the SAU execution register. It is important to remember that these indicators (AD, IF, DS, EXE) apply to the "branch to" address.

Therefore, if SAU finds the branch condition is satisfied it looks at the indicators to see if the "branch to" area is reliable . If any one of the indicators are on it indicates something is wrong with the "branch to" address and causes the branch to be unsuccessful, The ABC action upon receiving the response generates a gate in signal to set the indicators in the indicator register. The indicators in SAU hold up the inputs to the corresponding indicators in the main indicator register and if the response "branch successful-conditional indicators" is given, the ABC generates the gate in signal to set the indicators in the main indicator register to the status of the inputs from the conditional indicator in SAU.

3. Branch successful - no conditional indicators . This is the response given when the SAU test show the branch condition is satisfied and the branch to address is reliable. Because the instruction unit assumed the branch to be unsuccessful, it continued processing the instruction sequentially. With SAU saying the branch must occur it means that all instructions following the branch instruction are invalid because they are no longer in the correct sequence. Because of this, a branch successful response causes lookahead to enter a recovery mode of operation in order to recover.

A variation of the Branch on Indicator instruction is encountered when prefixed with the Store Instruction Counter If function. In this case an additional level of lookahead is involved. This additional level requires different action depending upon the branch conditions.

No assumption is made concerning the success of the bit test. Instead after loading the recovery level into the lookahead, the Instruction Unit waits for the SAU to complete the test before completing the instruction. (Since the Instruction Unit retains the branch address in this case, no recovery is

necessary and the "recovery level" exists only for control simplicity).

The special action of the ABC at the recovery level is modified to the following: (The coding of this level differs from the recovery load for Branch on Indicator/Bit only by the absence of the IC bit.)

1. Branch Unsuccessful

   In addition to the transfer of indicators, the control pulse, Resume-No Stica is sent to the Instruction Unit. The additional level in this case is a "dummy" level with the IC tag bit to effectively close out the instruction and cause the interrupt mechanism to be memorized.

2. Branch Successful-conditional indicators

   Same as 1) above with these indicators entered into the Indicator Register.

3. Branch Successful - no conditional indicators

   In this case, the control pulse Resume-Stica is given the Instruction Unit. This unit proceeds to effect the store instruction counter action and places the result in the lookahead for eventual storing.

Certain instruction unit instructions while being processed in lookahead require certain ABC action. This action results in a transfer of the index result indicators when necessary. Each of the special ABC actions is discussed in the following sections.

As can be seen from the previous discussions, the ABC action is quite diversified. Actions which occur every time regardless of any other conditons are the transfer of indicators at every level and the transfer of the IC field to the IC buffer at every ABC level tagged with an IC bit. The internal

tag bit and ABC decoding determine additional and special ABC action and each is discussed in the following sections of Section 5.0.00. An important point to consider is the fact that for normal ABC functions, the ABC is not dependent upon TBC action being complete. The normal ABC action requires only instruction preparation being complete and MAR Mode, With these conditions satisfied, the ABC can function and depending upon the time the conditions are met, both the TBC and ABC can be functioning together. The MAR Mode timing is the deterimining factor as far as the time the ABC can function assumming of course that instruction preparation is complete. The main thing to remember about ABC action is the normal indicator and IC field transfer. An internal tag causes specific internal register routing and stores require still other action. By realizing the conditions at the ABC level, the action can be easily followed. Specific details are given for normal ABC action, internal operations, stores, and all special actions.

5.2.00 Indicator Transfer - Figure 5.2-2

The indicator field at each lookahead level is classified into three main categories. The distinction is necessary because of the variation in handling the indicators during three major modes of lookahead operation (normal, reject, and houseclean). Each group is explained below:

Instruction Exception ($\overline{XR}$) Indicators

These indicators are the IR, OP, AD, DS, DF, and IF. They are gated into the main indicator register from each level during normal operation and instruction reject action. These indicators are the indicators affected during instruction preparation from either the instruction unit or lookahead (refer Section 3.0.00).

Index Result (XR) Indicators

The index result indicators are the XF, SCZ, XVLZ, XVZ, XVGZ, XL, XE, and XH. These indicators pertain strictly to the status of index core storage and are loaded from the instruction unit during the loading process. These indicators are gated to the main indicator register from distinctly coded instruction unit instruction levels during normal operation only.

Conditional Machine Check (CNIDC) Indicator

This single indicator is gated to the main indicator register from each level during instruction reject and houseclean action. When gated to the main indicator register it effects the status of the MK indicator. The setting of the CNIDC indicator in lookahead is conditioned by the instruction unit during the loading process. The CNIDC indicator is set during the loading process when an error occurs during the data transfer of any recovery information (psuedo stores, progressive indexing pseudo stores and branch recovery data). Only

if the recovery data is to be used is the error significant.

All the indicators in the indicator field are gated out (dc gate out) to the main indicator register simply by the ABC value. The ABC value designates the level and the indicator field at the level hold up inputs to the corresponding indicator positions in the main indicator register. The transfer indicator timer is started as soon as instruction preparation (LF and LC) 15 complete and that MAR MODE is present. The ABC decoding selects the timed outputs of the transfer indicator timer to set the appropirate indicator in the main indicator register. The indicators affected (XR, $\overline{\text{XR}}$, CNIDC) is determined by the ABC decoding and the setting pulse, selected by the decoding, is derived from the transfer indicator timer.

Another function of the transfer indicator timer is the transfer of the lookahead IC field to the IC buffer. This transfer however does not occur at each level. An addtional requirement for the IC field transfer is the presence of the IC tag bit. The IC tag bit indicates the final levd of any instruction and only at that time is the IC field transferred to the IC buffer. The transfer is timed with the transfer indicator timer.

5.2.01 Transfer Indicator Timer - Figure 5.2-2

The timing pulses necessary for the indicator and IC field transfer are derived from the transfer indicator timer. The starting of the timer requires only two specific conditions. First the instruction preparation must be complete, this is decoded by the ABC with a line called ABC decode level loaded. Referring to Figure 5.2-1 all ABC decoding is shown. Item 35 shows the required conditions are LF, LC, NO DISC and the ABC value designating the level at which the action is to occur. The second requirement for starting the timer is the presence of MAR MODE. The MAR MODE codition is necessary to insure that the previous instruction did not cause an interrupt. It signifies that the main indicator register is available to receive indicators for the following instruction test. For speed purpose the MAR Mode anticipation is used if MAR MODE is not yet present. The turn on of the indicator transfer timer is further conditioned by "no houseclean mode" and the ABC Advance Enable Sequence (AES) trigger being on. The no houseclean mode signifies normal ABC action and the AES trigger is used to insure that the timer functions only once for any given level. The AES trigger is reset when the timer functions and is not turned on again until the ABC steps to the next level. By this method the transfer indicator timer can function only once at any level. With all conditions satisfied the timer is turned on.

5.2.02 Indicator Transfer - Figure 5.2-2

The ABC decoding determines which category of indicators to be transferred. All indicators as well as the IC field are holding up inputs to their respective triggers simply by the ABC Counter allowing the DC gate out. The ABC decoding gates the timer output to allow the selected indicators to be gated into the main

indicator register.

Instruction Exception ($\overline{XR}$) Indicators -- These indicators are transferred at every level regardless of the decoding. The timer output E and M is sent to the indicator register where the input lines from the look-ahead indicator field are already active (IJ, AD, OP, DS, DF and IF). The arrival of the setting pulse sets the indicators in the main indicator register to the status of the input lines. The transfer of the conditional indicators only requires the transfer indicator timer and the transfer occurs once in every ABC level.

Index Result Indicators -- The transfer of these indicators is also accomplished with E and M of the indicator transfer timer. However, the timer output is conditioned by an ABC decoded line identifying the transfer condition. The decoded condition allowing the transfer of the index result indicators is called "ABC decode NOR PX NO NOOP MODE". The conditions causing this line to be active are shown in the ABC decoding chart in Figure 5.2-1 (Item 40). Besides the bit structure shown the line is also active when the ABC decodes:

1. Instruction unit external store (21)
2. Instruction unit pseudo store (24)
3. I Ex indicator transfer only (26)
4. Internal instruction unit store (32)
5. NOOP instruction unit pseudo store (38)

Those instructions which the instruction unit terminates by loading look-ahead with any of the levels listed above may cause the index result indicators to change, thus, the decoding of these levels allows the indicator transfer to the main indicator register. With these levels decoded by the ABC, the decoded result gates the timer output to the main indicator register where the indicators are set to the status of the input lines from the index result indicators in look-ahead.

Conditional Machine Check (CNIDC) -- This indicator is transferred to the MK indicator in the main indicator register by the transfer indicator timer E and M output only when the ABC decodes:

1. Houseclean mode
2. NOOP coded recovery level after a successful branch condition.
3. NOOP Mode
4. NOOP coded PX level with no NOOP tag.

This last condition is an example where op code position 8 is used as an auxillary NOOP bit. Remember the NOOP tag cannot be set after the last operand level of a VFL instruction. Because the PX level of a VFL instruction is the last level of a VFL instruction, op code position 8 serves as an auxillary NOOP bit if a data error occurs during the data load from the instruction unit. The decoding of this level also initiates a pseudo interrupt and therefore the CNI, DC indicator is transferred because the recovery information is required. Because the recovery data is required and is in error, the error condition is so noted in the main indicator register by transferring the CNIDC indicator from look-ahead to the MK indicator in the main indicator register.

The only time the CNIDC indicator is transferred to the MK indicator in the main indicator register is during instruction reject, branch recovery or houseclean action. Notice that each one of the decoded condition indicate one of these conditions. The decoded condition gates the timer output to the MK indicator in the main indicator register where it is set to the status of the CNIDC input line from look-ahead.

5.2.03 IC Field to IC Buffer

Another function of the transfer indicator timer is to time the transfer of the look-ahead IC field from the designated ABC level to the look-ahead IC buffer. Unlike the indicator transfer, the IC field is transferred to the IC buffer only when the level is tagged with the IC tag bit designating the final level of the instruction. The input to the IC buffer is active from the ABC level counter designating the physical lookahead level. These inputs are d.c. level inputs derived from the output of the IC field. The ABC value allows the IC field output at the counter value to become active to the input of the IC buffer. Because the inputs are only d.c. lines, the buffer triggers require a setting pulse to set. The setting pulse is derived from the indicator transfer timer E and not M and is conditioned by the IC tag bit at the level. Regardless then of the IC field input to the buffer, the only time the buffer triggers set is when the IC tag is on at the level. The IC tag is set at the last level of any instruction during the level load from the instruction unit. In a single level FLP instruction the buffer is set during the ABC time of the level. In a six level VFL instruction the IC buffer is set during the ABC action at the sixth level because that is the only level at which the IC tag is on. Remember the IC

buffer always contain the instruction value plus 1 (N + 1) and is used to give the instruction unit an instruction restart address if instruction N should interrupt.

The indicator transfer timer also participates in the MAR MODE test for the instruction at the level. This action is discussed in Section 5.6.00. The transfer of indicators occurs at every level of every instruction regardless of any conditions. The IC field is transferred to the IC buffer at every level that contains an IC tag bit and, besides the IC tag, is an unconditional transfer. These two operations however do not necessarily complete the ABC action at a particular level. The ABC has control of all internal functions and these functions are realized at any level tagged with the internal tag bit. Also if the level is a store level the ABC has additional action is preparing the data for the store operation. The internal operations and the store operations are covered in sections 5.3.00 and 5.4.00 respectfully. Assuming no store level or internal requirements the ABC is allowed to stop, contingent upon the TBC-ABC interlock at the completion of the transfer indicator function. Section 5.5.00 explains the counter stepping in detail.

5.3.00 Internal Fetch Request - Figure 5.3-1

Whenever an operand required for execution is located in an internal register (address 3 and 5 thru 12), the fetch request is made by lookahead during ABC time. The operands are required either by an arithmetic execution unit or the I unit. When the operand is required by an arithmetic unit (SAU or PAU) lookahead controls the data flow from the internal register to the C-D register. All the data flow is on the arithmetic bus and the requested data word does not physically appear in lookahead.

It is not necessary to place the word in lookahead because the internal registers and the C-D register are both tied directly to the arithmetic bus. The lookahead ABC action then consists of controlling the data flow and initiating the proper signals to gate the word from the internal register and gating the word into the C-D register.

If the operand is requested from the I unit, the action is somewhat different in that the word is placed in lookahead. The lookahead ABC action consists of gating the word from the internal register to the arithmetic bus and then initiating the proper control signals to gate the data into the lookahead data field at the ABC level. The arrival of the data into the lookahead level is the extent of the internal fetch action for the ABC. The following SCC action routes the data field to the I unit Y register to complete the internal register fetch for the I Unit.

It must be remembered that the above internal fetch actions are in addition to the normal ABC action of transferring indicators, and if the level is tagged with an IC bit, the transferring of the IC field to the IC buffer. The transfer of indicators is accomplished with the indicator transfer timer and

the internal register fetching is achieved with the A bus timer; both are controlled by the ABC.

5.3.01 A Bus Timer - Figure 5.3-1

The A bus timer differs slightly from the other timers used in lookahead in that it has 3 steps instead of 2. The timer consists of an EE trigger, E trigger and an M trigger. The basic conditions for starting the timer are the internal tag bit on, no noop bit, no noop mode and the pulse E and M from the transfer indicator timer. The timer outputs developed the gate out and gate in pulses from the internal register to the destination register (C-D register or lookahead) via the arithmetic bus. The timer also develops special signals to the A checker to pass the data through. If the level is a first level floating point instruction, the A bus timer also develops a signal to PAU to gate in the operation code field from lookahead and start. Remember that a FLP instruction buffers the operand at the same level as the operation code field. If the operand is required from an internal register, the T time is not started during the TBC time and consequently the operation code is not transferred or the start signal given during the TBC action. TBC action at the level consists only of a d.c. gate out of the operation code field to the PAU execution register. Because nothing is tranferred at TBC time, PAU does not develop the signal that it has gated in; hence, the TBC does not step. The conditions for the A bus timer are not dependent upon any TBC action so as soon as the ABC is at the same level as the TBC, the A bus timer starts. The TBC is still gating out the operation code field to the PAU execution register. The A bus timer along with the LAAR decoding determining the required internal register gates the data to the C-D register and develops the signals to gate the data into the

C register plus sending PAU the signal to gate in the operation code field and start. As soon as PAU accepts the data and start signal, it returns a (to) pulse to lookahead telling that they have accepted the data. The (to) pulse allows the TBC to advance contingent upon the TBC-OCC interlock Also the A bus timer M trigger provides for the ABC advance contingent upon the ABC-TBC interlock.

To insure that the A bus timer functions only once at any given ABC level, the ABC has an advance enable sequence (AES) trigger that is on when the ABC starts action. When the transfer indicator timer functions the AES trigger is reset for the balance of the ABC action. After the ABC action is complete, the AES trigger is turned on when the counter steps. The turn on of the AES allows the reset of the M trigger of the A bus timer. The EE trigger and the E trigger are reset with the first sample following the E and M output from the timer. The M trigger reset is the first sample following the turn on of the ABC AES trigger which is a function of the ABC advance.

5.3.02 Internal Register Gate Out

The gate out of the internal register is accomplished by EE of the A bus timer. The address of the internal register is contained in the LAAR. Remember an internal function is a type 3 load and requires the use of the LAAR. The address of the internal register that an operand is required from is set into the LAAR by the I unit during the load. When the A bus timer starts, the EE trigger of the timer combines with the LAAR decoding and the internal bit and gates the addressed internal register contents out on ACIB.

The

LAAR decoding is shown in figure _____. At the same time the internal register is gated out on ACIB, the A bus timer EE trigger signals the A checker to "pass the data". This means that the A checker accepts the data in the first level latches (ACIB) and transfers it to the second level latches (ACOB). The following sample turns on the A bus timer E trigger which combines with not M of the timer to provide the following ABC action. Because the data is passing from one register to another, parity checks are made to insure an error free transfer. However, if the addressed internal register was address 7 or 11, the A checker checking circuits are blocked. All other internal register addresses are parity checked. Address 7 and 11 do not contain any check bits on their data contents and it is for this reason that the checking is blocked. All that remains to be done is the gating in of the word from ACOB to the C-D register. With the sample following the turn on of the E trigger, the M trigger is turned on to complete the operation. With the data latched on ACOB the internal register has effectively been gated out. The destination of the word is dependent upon the ABC decoding. If the operation is arithmetic the decoding is such that the word is gated into the C-D register. If the fetch was requested for the I unit, the word is gated into the lookahead data field for subsequent storing into the Y register in the I unit.

5.3.03 Arithmetic Internal Fetch

If the internal fetch was required for an arithmetic operation, the final destination of the word is the C-D register. The A-bus timer function for gating the word to the C or D register is determined by the ABC decoding. The only time an operand is placed in the D register is when it is the second operand

for a VFL instruction with WBC. When the ABC decoding is such that the word is a VFL second operand, the word is routed to the D register, otherwise it is gated to the C register. All FLP internal operands are routed to the C register. The timer E and M pulse gated by the internal tag and the "C register required" line sets the C register triggers to the ACOB lines feeding the register. The ACOB output are a d.c. level lines holding up the inputs to all registers tied to it. With the gated E and M pulse, just described, the C register is the only register which receives the gate in pulse and therefore the result on ACOB is gated into the C register. If the ABC decoding indicates the D register, the setting pulse (E and M of the A bus timer) is routed to the D register instead of the C register. The result is then that the ACOB data lines are set into the D register. Because the internal register fetch is now complete, the LAAR is not required and therefore the LAAR busy trigger is reset. The LAAR busy trigger protected the LAAR from being altered because it contained the address of the internal register that was required by the fetch. Since the fetch is complete the LAAR can be altered by a type 3 or type 2 load. The reset pulse for the LAAR busy trigger is the E and M output of the A bus timer conditioned by the LAAR decoding internal address and the ABC decoding no branch on indicator. The latter decoding is necessary because the branch on indicator requires a normal internal fetch but the word must be re-stored at a later level (after execution) and therefore the LAAR remains busy to protect the LAAR address until the store is complete.

Another signal is developed by the E and M output of the A bus timer if the ABC decoding specifies a FLP internal operand fetch. The TBC action at this level results only in the TBC gating the operation code field to the input

of the PAU execution register. The inputs from the TBC to the execution register are d.c. lines resulting from TBC decoding. To complete the transfer then the ABC must generate the gate in signal for the execution register and also send the start signal to PAU. The start and gate in signal is the A bus timer E and M pulse gated by the ABC decoding the FLP internal condition. The acceptance of the signal by PAU results in a return signal to lookahead allowing the stepping if the TBC. This action is never necessary on VFL instructions because the entire first level of a VFL instruction contains operation code information only and no operands. Therefore a VFL first level is always tranferred by the TBC normally and SAU starts its housekeeping. Any operand requirements at succeeding levels are also transferred by the TBC unless they are required of an internal register in which case they are transferred by the ABC action already explained.

5.3.04 Non Arithmetic Internal Fetch

If the internal fetch was made to obtain an operand for the I unit, the word must be placed in the lookahead data field. The ABC decoding specifies an internal I unit fetch. This decoding conditions E and M of the A bus timer to gate the latched data on ACOB to the lookahead data field designated by the ABC value included in the decoding. With the data located in the lookahead data field, subsequent action is taken by lookahead at SCC time to place the data into the Y register of the I unit. To signify that the SCC requires action, the ABC action

on internal fetches resets the LF tag bit. The LF tag bit being off signifies SCC action. The LF tag bit is reset by E and M of the A bus timer conditioned by the I unit internal fetch decoding and the ABC value designating the level.

Following is a brief summary of the internal fetch action. The internal fetch is specified by the internal tag bit being on at the level. By starting the A bus timer the internal register contents designated by the LAAR are gated out on ACIB to the A checker. The A checker latches the data on ACOB. The data is gated into the C-D register or lookahead data field depending upon the ABC decoding. If the data is routed to lookahead, the LF tag is reset to specify SCC action. The compeltion of the ABC action for the internal fetch resets the LAAR busy trigger. The internal fetch is complete and the ABC allowed to advance, ABC advance conditions are discussed in section 5.6.00.

5.4.00 STORE OPERATIONS

There are three possible areas to which lookahead can store data. They are the internal registers, external storage and the I-unit. The ABC functions on all types of stores regardless of the area at which the data is to be placed. The ABC action differs between internal register stores and non-internal stores. The difference is that the ABC entirely completes all stores to an internal register, while the ABC action is limited when non-internal stores are specified. The ABC action for non-internal stores is limited to resetting the LF tag bit as an indication for SCC to function to complete the store and if the store involves an arithmetic result, the ABC has the additional function of placing the data into lookahead from the arithmetic unit. When data is received from the I-unit for storing, there are two possible locations it can be stored at. The LAAR decoding of the store "to" address specifies either an internal or non-internal location. If the address specifies an internal register, the data is routed to the C register during TBC time. When the ABC functions at the level, the data is routed from C register to the internal register specified by the LAAR. If the LAAR decoding indicates an external location, the ABC action is limited only to a reset of the LF tag bit. The LF tag bit being off allows the SCC to function to complete the store. In this latter example, the data is already in lookahead from the load from the I-unit.

If the store operation is a VFL or FLP instruction, the data to be stored is either latched on the arithmetic checker out bus (ACOB) or in the CD register. In all FLP instructions the store data is latched on the arithmetic out bus (ACOB). In VFL instructions, however, the data is either latched on ACOB or in the CD register. All VFL fetch and store instructions have the data placed in the CD register. If a WBC exists, the first word is in the C register and the second word

is in the D register. If no WBC crossover exists the single word is located in the C register. This type of store is accomplished in two lookahead levels, but the ABC action is identical for both levels except at one level the ABC gates the data from C and the other level is gated from D. In the special operand fetch and store instructions (MPYC, Load factor, etc.), where only one store level exists regardless of the WBC condition, the data is latched on ACOB and the action is identical to an FLP store operation.

The ABC action for internal stores when the data is latched on the arithmetic bus consists of routing the data to the internal register specified by the LAAR. The ACOB data outputs are available to the inputs of all internal registers. The ABC action then consists of developing a setting pulse which becomes selective to only the internal register designated by the LAAR decoding. When the action is finished the store is complete and the LF tag is not reset. No SCC action results if the LF tag bit is on when the SCC steps into the level. On non-internal stores, the ABC action for data latched on ACOB consists of routing the data to the lookahead data field specified by ABC value. The LAAR decoding of not internal gates an ABC generated setting pulse to sample the data into lookahead. At the same time, the LF tag bit is reset. When the SCC steps into the level with the LF tag off, it starts the action of storing the data at the location specified by the LAAR. The ABC action is complete with the data transfer to lookahead and the LF tag reset. In the VFL type instructions where the data is in the CD register

register to the A checker in bus (ACIB) and then to the A checker. The A checker action latches the data on ACOB and from this point the ABC action is identical to that already described.

All store operations with the exception of an I unit store to an external address are accomplished with the A bus timer. Because all store instructions are type 3 loads, the LAAR busy trigger is set during the load. The LAAR contains the address of the location where the data is to be stored. This address must be protected until the store is complete; hence the busy trigger. The busy trigger is reset during ABC time on internal register stores, but the LAAR remains busy on non internal stores. The LAAR reset is a function of the SCC on non internal stores because it is the SCC that completes the storing action.

The ABC action for store levels involved with BB or BI instructions are similar to that just described. The branch functions of the ABC are discussed in section 5.5.00. This section discusses the ABC action for all store levels other than those of conditional branching.

Again it is important to remember that ABC action just described is in addition to the normal ABC action of transferring indicators and, if the level is tagged with an IC bit, the IC field transfer to the buffer also occurs. In addition to the transferring of the non index result indicators ($\overline{XR}$), the index result (XR) indicators are also transferred if the level is an instruction unit store level.

5.4.01 Starting The A Bus Timer - Figure 5.4-1

The starting of the A bus timer for a ABC arithmetic store function is determined by the ABC decoding and an indication from PAU or SAU that the data is available. The ABC decoding of an I unit internal store is sufficient to condition the start the timer for that type of store. An I unit external store does not condition the start of the timer because it is not required for non-internal I unit stores.

On all arithmetic type stores where the data is latched on ACOB, the A bus timer E trigger is conditioned to turn on. The EE trigger of the timer is not necessary because a two cycle transfer of the data from ACOB to lookahead or the specified internal register is all that is required. When the store data is located in the C-D register, however, an additional timer cycle is required. The additional timer cycle is obtained by starting the A bus timer with the EE trigger instead of the E trigger. The additional cycle is required to transfer the data from the C-D register to the A checker output (ACOB). The variable starting point of the timer is determined by the ABC decoding.

When the store level is decoded as an internal I unit store, the decode line establishes the turn on condition of the A bus timer with the EE trigger. Here again a three cycle transfer of the store data is required because the TBC action placed the store data in the C register. The additional cycle is necessary for the C register to ACOB transfer. The E and M triggers accomplish the final transfer of data.

5.4.02 Arithmetic Store Operations - Figures 5.4-2 and 3

This section discusses all arithmetic store instructions both internal and non internal. Because of the variable starting point of the A bus timer, each specific store type is discussed separately.

In stores from the C-D Register in the VFL stores, where the store data is located in the C-D register, the decoding is such that the A bus timer is started by turning on the EE trigger of the timer. The ABC decoding of the lookahead operand field results in a decoded line "ABC decode store C or D". This line establishes the starting point of the A bus timer at the EE trigger. Further conditioning for starting the timer is a signal from SAU that the data to be stored is available. This signal from SAU is "VFL last cycle store" and results in a last cycle store memory trigger being turned on in lookahead. This trigger remembers that the data is available. The final condition for starting the timer is a transfer indicator E or M line from the transfer indicator timer. This line is necessary as it proves the right for the ABC to function (MAR MODE and instruction preparation complete). The M trigger of the indicator transfer timer does not reset following the indicator transfer action. The reset conditions for the reset of the M trigger of the indicator transfer timer is a A bus timer output or an ABC advance condition. When the ABC is in a store level, all ABC advance conditions are delayed and become a function of the A bus timer. Of course if the level is NO-OPed, no ABC store action occurs. The combination of the signals just mentioned (Xfer IND E or M, last cycle store, ABC decode store C or D and no noop mode) allows a clock sample pulse to turn on the EE trigger of the A bus timer. The A bus timer combines with "ABC decode store C" or "ABC decode store D" to gate the

C or D register to ACIB. The C-D register output feeds the input (ACIB) to the A checker. The EE trigger output also sends a "pass data LA" signal to the A checker. This signal allows the A checker to accept the data from ACIB and latch it on the arithmetic checker out bus (ACOB). The data gate out of the C or D register and the "pass data" signal to the I checker is the extent of the A bus timer EE function and with the following sample pulse the EE trigger turns on the E trigger of the timer. At the E trigger turn on, the store data is latched on ACOB. The E and not M triggers of the timer send two signals to the checker to control the A checking. The E and M output of the timer, gated by the decoded conditions resets the LAAR busy trigger. The reset conditions for the LAAR reset are the LAAR decoding an internal address, and no branch on indicator fetch level. These decoded lines condition the E and M pulse from the A timer to reset the LAAR. Because the store is to an internal register, the LAAR use is now complete and the busy trigger can be reset. The no branch on indicator condition is necessary because the fetch level on a branch on indicator instruction is followed by a store level to return the word after testing. Because the word is replaced, the LAAR must remain busy to protect the store address.

With the data latched on ACOB, the next ABC action is determined by the LAAR decoding. From ACOB the data is routed either to the internal register specified by the LAAR or to lookahead if the decoding recognizes a non internal store. If the store is to an internal register, the A bus timer E and M triggers control the gate in of the specified internal register. Because the ACOB output is available to all internal registers, the setting pulse (E and M) is conditioned by the LAAR decoding thereby selecting the proper internal register. The

conditioning for the internal register gate in results from the LAAR decoding an internal address, the ABC decoding a store, and the last cycle store memory trigger on. These conditions result in a "cond internal register gate in" line which combines with the E and M timer output and the decoded LAAR address to provide the sample pulse to gate the data off ACOB into the register. Following is a listing of the bits gated and the internal register address for VFL stores from the C-D register.

LAAR Address	LOOKAHEAD OUTPUT
1. Address 3	GI ACOB To ULB 00-63
2. Address 6	CND ACOB To CPU S00-19
3. Address 7	ACOB COUNTS
4. Address 8	CND A 00-59 & PARITY
	CND A 60-63 & PARITY
5. Address 9	CND B 00-63 & PARITY
6. Address 10	G7 ACOB 00-07 SRAB 0-7
7. Address 11	CND ACOB To N 20-63
8. Address 12	CND ACOB To M 20-49

Notice that all the internal registers can be stored into except address 5. The loading of address 5 is a function of the exchange operation. Lookahead cannot store into address 5, but it can fetch it on an internal fetch. If address 11 is the specified internal register, the ABC has additional action to perform. Because the store is to the indicator register (address 11) the index result indicators may be modified. In this special case, an additional timer is started by lookahead to transfer the index result indicators in the main indicator register to the updated indicator register in the I unit. The timer is started with E and M of the ABC timer and a decoded ABC line identifying the store to the indicator register. If it is the branch on indicator store level, however, the action is not taken because the lookahead is only returning the index word after bit modification by SAU (no new results are stored). The indicator transfer timer consists of an E and M

trigger. The E and M trigger outputs provide the transfer pulse for the index result indicators in the main indicator register to the up-dated indicator register in the I unit. The E trigger is reset with the first sample following the turning on of the M trigger. The M trigger is reset when the ABC advances.

Special signals are also generated to the A checker when the internal store is to address 8 or 9. These signals are "LA STORE TO A" (address 8) or "LA STORE TO B" (address 9) and they are sent to the arithmetic checker to update the residue.

If the store operation from the C-D register is designated to a non internal address, the ABC action is simplified. The A bus timer EE function is identical to that already described. The signals to control the A checker during the data flow are also identical. The E and M functions of the A bus timer differ for non internal stores. The data, latched on ACOB following the EE function, is gated into lookahead by the E and M triggers of the A bus timer. In addition, the LF tag bit is reset to signify SCC action. The output of ACOB conditions the data field of all the lookahead levels. The data field that accepts the E and M setting pulse is determined by the ABC value. The data path is specified as ACOB to lookahead by the LAAR decoding a non internal address. These conditions combined with ABC decoding store and the last cycle store memory trigger on combine with E and M of the A bus timer to set the data into the lookahead data field, designated by the ABC value. The same conditions are used to gate E and M of the timer to reset the LF tag bit. With the LF tag bit reset, the SCC recognizes that is has action and it completes the store to the address specified by the LAAR. When the store is not internal, the LAAR busy trigger is not reset.

The busy trigger reset for non internal stores is a function of the SCC action. The stores just discussed are identical to an ABC decoding of an I unit internal store. The data paths and A checker controls are the same.

If the decoded store operation is a FLP or a special store associated with the load factor type of VFL instructions, the ABC action is similar but not identical. The major difference is the operation of the A bus timer. In the previous discussion of stores, the A bus timer was started with the EE trigger to obtain the required 3 cycle data transfer. This was necessary because the store data was located in the C-D register. In the FLP stores and the special VFL store levels, the data is already latched on ACOB. This eliminates the use of the EE trigger of the A bus timer and therefore when the ABC decodes this type of store the A bus timer is conditioned to start with the E trigger. The turn on the E trigger of the A bus timer is transfer indicator E or M, last cycle store (indicates data is available) and ABC decode arithmetic bus. The latter condition identifies the FLP or special VFL store level. The timer operation controls the data flow to the address specified by the LAAR. The data flow paths are the same as those already described. The only difference is the signals to the A checker to control the data check. The signal to the A checker is "store bus except" which tells the checker it can perform parity and residue checks on the data. The signal is a result of the ABC decoding, A bus timer E and M, and last cycle store memory. It effectively means that lookahead has routed the data. Again if the store is to address 11, the indicator register timer transfers the index result indicators to the updated indicator register in the I unit. If the address is 8 or 9, special signals are generated and sent to the A

checker to update the residue. If the store is internal, the LAAR busy trigger is reset, or, if not internal the busy trigger remains on and the LF tag reset to signify SCC action. The difference between the store operations from the C-D register versus those from ACOB is the extra cycle (EE) needed to transfer the data from C-D to ACOB on the C-D stores. The controls to the checker for C-D stores are signals to pass the data from ACIB to ACOB and check parity while the signal from ACOB stores is a signal to check parity and residue. Data flow and data controls are identical from ACOB to the destination.

5.5.00 Conditional Branching, Progressive Indexing and Non Arithmetic ABC Action

This first part of this section explains the ABC action for conditional branching. The branch instructions considered are the branch on bit (BB) and the branch on indicator (BI) instructions. Both of these instructions are tested by SAU because SAU is the only unit capable of examining any bit and modifying any bit in the computer system. These instructions require action very similar to normal VFL instructions.

When the I unit process a BB or BI instruction, it assumes the branch will fail and processes instructions accordingly. In the event that SAU finds a successful branch condition, lookahead must have some means of restoring the computer because instructions following the branch instruction are out of sequence. To allow for the possible recovery action, an additional level is associated with the BB and BI instructions. This level is called a branch recovery level and contains the "branch to" address which lookahead sends back to the I unit if the branch proves successful. When the SAU proves a branch successful condition a modified housecleaning procedure occurs and the "branch to" address is returned to the I unit.

The instruction preparation of the BB and BI instructions include testing for those conditions, which, if the branch conditions are met, cause the instruction to be rejected. These tests include the "branch to" address failing the address invalid (AD) or instruction fetch (IF) tests, certain store address failing the data store (DS) test and the instruction itself failing the execute exception

(EXE) test. These conditional indicators are buffered in the operation code field of the first level associated with the BB or BI instructions. The indicators must be interrogated by SAU such that the presence of any one causes any bit modification to be blocked if the branch conditions are met. If any one of the indicators are on, regardless of the branch being successful, the system does not branch. The system branches only when SAU indicates the branch conditions are met and no conditional indicator is on.

The last part of this section covers ABC action on progressive indexing levels (PX), I/O levels, and certain I unit instruction levels. The ABC action is generally simplified for these levels consisting of an indicator transfer and the MAR Mode test.

5.5.01 Branch on Bit (BB) and Branch on Indicator

The BB and BI instructions consists of four levels. They are:

1. Operation Code Level
2. Fetch level (operand)
3. Store level
4. Recovery level

In addition the BI instruction, if specified, with the store instruction counter instruction, (STICA) has an additional level. This additional level is a dummy level tagged with an IC bit so to effectively allow lookahead to close out the instruction. The difference between the normal BI instruction and the one pre-

fixed with the STICA instruction is that no assumption is made concerning the success of the bit test. Instead after loading the recovery level into the look-ahead, the Instruction Unit waits for the SAU to complete the test before completing the instruction. (Since the Instruction Unit retains the branch address in this case, no recovery is necessary and the "recovery level" exists only for control simplicity.)

The special action of the ABC at the recovery level is modified to the following: (The coding of this level differs from the recovery load for Branch on Indicator/Bit only the absence of the IC bit.)

The operation code level is handled similar to the normal VFL operation code. The only exception is that the branch condition is decoded in look-ahead and relayed to SAU as BB or BI. The reason for this action is because SAU does not have provisions for a decoding the branch instruction and also the operation code field in look-ahead contains the conditional indicators which are relayed to the execution register SAU for their test of the branch conditions. With these exceptions the level is handled identically as any SAU operation code level from TBC through SCC. The operand level is identical to any VFL operand level. The BB store level is identical to a normal VFL store level plus it receives an indication from SAU specifying the results of the branch test. The BI store level is the same as a VFL store with address 11 except that the indicator register timer is not necessary to transfer the main indicator register to the I unit updated indicator register. This latter action is not necessary because the word was not altered and is only being returned. The same signals from SAU are received during ABC to indicate the status of the branch conditions.

Coincident with the Last Cycle Store pulse, responses indicating the results of the bit test and the status of the conditional indicators are memorized by the look-ahead in the Branch Test Result triggers to define the action to be taken at the branch recovery level.

The ABC action at the recovery level, in addition to the normal transfer of indicators, etc., varies depending upon the response.

1. Branch Unsuccessful

   No special action since Instruction Unit assumption was correct and the instruction was not rejected.

2. Branch Successful-conditional indicators

   Instruction Unit assumption was correct (since instruction is rejected) and the conditional indicators are entered into the Indicator Register. This is a direct path from the SAU execution register to the input gates of the Indicator Register. Look-ahead develops the sample to set the indicators into the main indicator register.

3. Branch Successful-no conditional indicators

   This case indicates the Instruction Unit assumption was incorrect and any later look-ahead levels must be considered invalid. Thus, the housecleaning Mode of operation is initiated in order to recover. Housecleaning caused by recovery action is discussed in Section 8.0.00.

The branch on indicator instruction action is slightly different than a normal store . The difference is:

The associated fetch and store levels of the Branch on Indicator instruction are distinctly coded due to special handling of bit positions 0-19 of the Indicator Register. These bit positions are normally unaffected by the storing function; however, to accomplish the bit resetting function of this instruction, coincident with the transfer of the register contents to the C register, bits 0-19 are reset. There is then a unipolar gate provided and activated by the ABC during the store transfer. With this exception, the Branch on Indicator and Branch on Bit instructions are handled by the look-ahead.

If the BI instruction is prefixed with the STICA instruction, the ABC action is different than that of the normal BI instruction. Listed below is the action taken by the ABC at the recovery level.

1. Branch Unsuccessful

   In addition to the transfer of indicators, the control pulse, Resume-No Stica is sent to the Instruction Unit. The additional level in this case is a "dummy" level with the IC tag bit to effectively close out the instruction and cause the interrupt mechanism to be memorized.[2]

2. Branch Successful-conditional indicators

   Same as 1) above with these indicators entered into the Indicator Register.

3. Branch Successful-no conditional indicators

   In this case, the control pulse Resume-Stica is given the Instruction Unit. This unit proceeds to effect the store instruction counter action and places the result in the look-ahead for eventual storing.

5.5.02 SAU Progressive Indexing (PX) levels.

The final level of all SAU instructions in which the progressive mode of indexing is specified is used to buffer the original contents of the affected index core storage location. During normal operation, the only look-ahead functions required are the transfer of indicators and instruction counter fields at the ABC level. Since the progressive indexing function allows the index result indicator to change, this category of indicators is included in the transfer.

5.5.03 Non Arithmetic ABC Action

This section discusses the ABC action for all I/O instruction and those I I Unit instructions that are not internal stores or external fetches. The internal stores and fetches are explained earlier in Section 5.0.00.

I/O Instructions

The ABC action at the first level of an I/O instruction consists of the non index result indicator transfer only. It is accomplished in the usual manner by starting the indicator transfer timer with an ABC decoded line identifying the I/O first level condition. The ABC action at the second (dummy) level of an I/O instruction transfers the non index result indicators and transfers the IC field to the IC buffer (second level I/O is tagged with IC bit). Besides the normal ABC transfer function the ABC resets the PAU master test complete trigger and the TBC AES. The PAU master test complete trigger is PAU's counterpart of the look-ahead MAR MODE trigger and therefore must be reset between each instruction. Because the I/O instruction is non arithmetic, the look-ahead must accomplish the reset. If MAR MODE is present following the I/O instruction the PAU master test complete trigger is turned on.

Although the TBC has no action to perform at the level, the advance must be delayed until the PAU master test complete trigger is reset to prevent the possibility of PAU falsely interrogating the trigger.

I Unit Instructions

The I Unit internal fetch and store instructions have already been discussed earlier in Section 5.4.00. This section discusses the ABC action for all other I Unit instructions and a review for the internal fetch and store instructions.

The instructions executed within the Instruction Unit are generally completed within that unit to the point where it is necessary to store and/or enter indicators. These functions are completed by the look-ahead in proper sequence. These levels are referred to as:

(1) Instruction Unit Store levels

(2) Indicator Transfer Only levels

Two other levels exist to process the cases where Index Core Storage is modified and where the look-ahead must furnish the contents of an internal register for proper execution by the Instruction Unit. These levels are referred to as:

(3) Pseudo Store level

(4) Internal Fetch level

Proper combinations of these four types of levels provide the look-ahead action necessary for completion of all Instruction Unit instructions.

It should be noted that those instructions which the Instruction Unit terminates by loading the look-ahead with any of the first three levels above may cause the Index Result indicators to change; thus, this category of indicators is included in the transfer.

(1) Instruction Unit Store Levels

In general, these levels may specify either external memory locations or an internal computer register as indicated by the contents of the LAAR. When external memory is specified, the necessary ECC bits, if possible, are generated during the loading process, thus eliminating the necessity of a later SCC check and check bit conversion cycle. The WBC tag bit is set by the Instruction Unit and used in conjunction with this particular Look-ahead Operation Code to identify this condition.

a. External Store in ECC

The ABC at this level, coincident with Indicator Transfer timer E-$\overline{M}$ resets the LF tag bit to initiate SCC action. This latter action provides the store request to the BCU directly.

b. External Store in Look-ahead Parity

It is not always possible, for checking purposes, to include the generation of ECC during the loading process. Thus, the necessity may exist for the normal SCC check cycle. The absence of the WBC tag bit indicates this condition. The SCC action is identical to that described in 6.0.00 with the Interrupt Mechanism interrogated after the check and check bit conversion cycle.

(2) Indicator Transfer Only Level

Processing the Indicator Transfer Only Level entails only the indicator transfer function and interrupt sequencing action.

(3) Pseudo Store Level

During normal operations, this level is identical to 2) above. Only in the event of rejecting the instruction or detecting a program interrupt do the Operand field contents become useful information.

(4) Internal Fetch Level

The look-ahead action of supplying the contents of an internal register to be used as an Instruction Unit operand is accomplished in two steps. The transfer from the specified internal register to the Operand field is accomplished at the ABC level. This takes place subsequent to the transfer of indicators and is timed by the A-timer in the usual manner. The sample selected by A-timer E-$\overline{M}$ resets LF to allow completion of the transfer by the SCC.

The ABC decoding identifies each level to allow the timer to be started.

5.6.00 ABC COUNTER STEPPING

The controls for stepping the ABC Counter are relatively simple. The basic stepping condition is the TBC-ABC interlock. If the TBC is at the same level as the ABC, the ABC cannot step until the TBC steps (one counter cannot pass another). The other ABC stepping conditions are dependent upon the type of ABC action at the level. If the ABC decoding indicates any internal or store operation, the ABC advance is delayed until the internal or store action has been completed. Also, if lookahead is in a houseclean mode of operation, the houseclean timer steps the ABC.

5.6.01 Normal Stepping (No stores or external operation).

The ABC action at every level, regardless of the decoding, is the transfer of indicators at the level to the main indicator register. By decoding that the ABC has no internal action or store action, the transfer indicator timer M trigger cause the "cond ABC advance" to become active. The conditional advance line combines with the TBC-ABC interlock status. If no interlock, the advance is allowed. If there is an interlock with the TBC, the advance cannot occur until the TBC steps. In the latter case, the transfer indicator timer M trigger remains active because the reset of the trigger is initiated by the ABC advancing. With no interlock signified the "cond advance cond" line causes the "ABC advance" to become active. The advance line combines with the present ABC counter value and the AES off to reset the present level ABC trigger and set the next level trigger. Also the advance line combines with the AES off, to turn the AES trigger back on. The AES is reset as soon as ABC action starts at a level and remains off until the ABC steps. This insures that the ABC functions occur only once at any given level. The advance condition

combines with the transfer indicator timer M trigger to provide the reset for the timer. Besides causing the "cond ABC advance cond" line, the transfer indicator timer M trigger, conditioned by no stores and no internal operations, causes an "anticipate ABC advance" line to be active. This line is sent to the SCC counter stepping controls when it participates in the SCC advance. This can be anticipated because the SCC only has action on non internal stores and since the ABC decoded condition, allowing the anticipation, has signified no stores, the SCC step can be anticipated. The only time it can be anticipated is when the ABC action consists of a transfer of indicators only. If the ABC has any other action at all, the SCC advance is not conditioned by the ABC.

5.6.02 ABC Stepping at Store and Internal Levels

The stepping again is dependent upon the TBC-ABC interlock. The conditional advance line becomes active dependent upon the ABC decoding. The transfer indicator M trigger is blocked from causing the conditional advance to become active by the ABC decoded line identifying the store or internal condition. Because the ABC decoding results in an internal or store operation, the A bus timer M trigger is used to obtain the conditional advance line. Normally, the A bus timer M trigger is sufficient to allow the conditional advance line to become active, but if the ABC decoding signifies that a store is to the indicator register and is not part of a branch on indicator, the ABC advance is delayed further. The reason for the added delay is because stores to the indicator register result in another timer (indicator transfer timer) being started to transfer the main indicator register contents to the I unit updated indicator register after the store. Because the additional data transfer requires more ABC time, the ABC stepping is delayed and becomes a function of

the indicator transfer timer M trigger. The indicator transfer timer is used in the special case because it signifies that the ABC action is complete. If the store to the indicator register was a result of a branch on indicator store level, the added delay is not necessary because the word is not altered but is only being replaced after testing by SAU. The indicator register timer is not used and therefore the ABC stepping on a branch on indicator store to the indicator register is a function of the A bus timer. Briefly then, ABC stepping for internal or store operations is a function of the A bus timer M trigger for all internal operations and stores except a normal store to the indicator register, in which case, the stepping becomes a function of the indicator register timer M trigger. When the conditional advance line becomes active, it is further conditioned by the TBC-ABC interlock. The reset of the M triggers of the A bus timer and indicator register timer are a function of the ABC stepping so therefore if an interlock prevails and stepping is blocked, the conditional advance line remains active until the interlock disappears. When the interlock disappears, the conditional advance line becomes active to allow the ABC to step. The conditions from the point are identical with those described in section 5.6.01. The ABC trigger at the existing level is reset and the trigger at the next level is set. The AES trigger is turned back on and resets are sent to the indicator transfer timer, the A bus timer and indicator register timer. The ABC action at level N is complete and all necessary resets are performed. The ABC is ready to function at level N+1. The SCC stepping is not anticipated.

If lookahead is in a houseclean operation, the ABC stepping is provided by a houseclean mode and houseclean timer. Housecleaning is discussed in section 8.0.00.

5.7.00 Modify Addressable Registers (MAR MODE)

The ABC functions are not only contingent upon instruction preparation being complete, but that the interrupt mechanism has been reliably interrogated. The course of normal ABC action entails those functions which require the modification of an addressable register. Before any addressable register can be modified, the computer system must be assurred that everything functioned correctly up to that point. The assurrance is the responsibility of lookahead because lookahead performs the instruction sequencing for execution. The indication that everything functioned correctly through the execution of any given instruction is in the form of MAR MODE and is the result of a reliable interrogation of the interrupt mechanism. Before the interrupt mechanism can be reliably interrogated, lookahead must test to see that all units have set into the main indicator register a record of any abnormal conditions arising from the execution of an instruction. Because such indicator settings arise from any number of non synchronous units, each units participation must be recorded and only when all units have reported can the interrupt line be reliably interrogated.

For the purpose of testing to see if all units have reported their indicator settings, lookahead contains a set of test triggers. In each case the sample pulse to set the trigger is timed identically with the sample selected to gate the information into the main indicator register. The test triggers and their functions were described in section 5.1.00 and are also included here for review.

Execution Unit Indicator Test

This trigger records the fact that the execution unit involved has entered all indicators associated with execution of instruction n.

Lookahead Normal Indicator Test

This trigger records the fact that the lookahead has entered all indicators associated with executing instruction n.

A-Checker Indicator Test

This trigger records the fact that the possible error detected during the normal arithmetic result check of instruction n has been entered.

Store Check Indicator Test

This trigger records the fact that the possible error detected during the SCC check and check bit conversion cycle for store instruction n has been entered into the Indicator Register. This trigger, unlike the others, is normally on and is reset as an ABC function only when it is evident such a check cycle will follow.

Each trigger must be set following each arithmetic unit instruction regardless of whether any abnormal condition actually occurs. The sample pulse selected by the coincidence of all the test triggers is used to memorize the state of the interrupt mechanism. For speed purposes, when all other units have reported, and no A checker error occurs, the interrupt mechanism is memorized immediately. The A checker test trigger is not set. If all other units have reported, then the A checker test is the last one to complete and if it indicates no error, the interrupt mechanism is memorized immediately rather than waiting until the test trigger sets and then waiting for another sample to memorize to interrupt state. The result of the interrupt memorization is MAR-MODE for the normal conditions or Interrupt Next Instruction for the interrupt condition.

The test triggers are also reset at the same time to allow them to record the

indicator settings for the next instruction (N+1).

Because the interrupt mechanism must be interrogated between each instruction, an alternate method must be used to determine the final indicator setting of the main indicator register for non-arithmetic instructions or no-oped arithmetic instructions because the execution units are not involved. The lookahead NO-OP indicator test trigger records the last lookahead indicator setting for these instructions. Because lookahead is the only unit involved, the interrupt line is memorized by the sample selected by the coincidence of the lookahead NO-OP test and store check indicator test trigger.

5.7.01 MAR MODE - Arithmetic Instructions - Figure 5.7-2

The units concerned with the execution of any arithmetic instruction are lookahead, the A checker, and SAU or PAU. The test triggers used for recording the indicator settings are:

In Lookahead:

1. Lookahead Normal Test
2. Store Check Indicator Test

In SAU or PAU

1. Execution Indicator Test

In A-checker

1. Check Indicator Test

The turn-on of the lookahead normal indicator test trigger is determined by the ABC decoding of the level, no no-op mode, and an indication that lookahead has transferred indicators. The fact that lookahead has taken the transfer indicator action is recognized in an indicator test E trigger. In multi-level

instructions, the final indicator setting cannot be realized until the indicators at each level are transferred; therefore a prime pre-requisite for turning on the indicator test E trigger is the IC tag being on. Only at the last level can the indicator test E trigger be turned on. If the ABC action at the IC level does not specify a store, the transfer indicator E and not M selects the sample to turn on the indicator test E trigger. If the ABC decoding specifies a store level, the turn on of the indicator test E trigger is accomplished with E and not M of the A bus timer. The turn on of indicator test E for ABC stores is delayed to catch a possible A checker error in the data transfer. In effect, the possible A checker error is associated with the lookahead indicators during ABC stores. Therefore the A bus timer E and not M participate in the indicator test E trigger turn on when the ABC decodes a store at the IC level. The indicator test E trigger, signifying lookahead indicator transfer, conditions the turn on of the lookahead normal indicator test trigger. Other conditions, besides indicator test E, that must be met before the lookahead normal indicator test can be turned on are not NOOP mode (signified by the NOOP mode trigger being off), the ABC decoding of an arithmetic type instruction (specified by the decoded line "no ABC LA only MAR next instruction) and the normal test indicator test being off. When all conditions are met, the lookahead normal indicator trigger is turned on.

The execution unit test trigger is turned on from a signal from either SAU or PAU. Essentially the signal received from either of the execution units is their end operation signal. The signal from PAU is "FLP END OP"; the one from SAU is "VFL SIR TO LA" (SIR=set indicators and reset). Both signals turn on the execution unit indicator test trigger directly.

The store check indicator test trigger is normally reset on. If the ABC decodes any non internal store, the trigger is turned off. This action delays the interrupt memorization until the SCC storing action is complete (SCC storing action is discussed in section 6.0.00). At the completion of the storing action the store check indicator test is turned back on. The turn on of the trigger indicates that any abnormal conditions arising as a result of the SCC storing action have been entered in the main indicator register. Because the trigger is initially reset on, no test is necessary if the ABC decoding specifies no SCC action (no stores to external storage or the I unit).

The test triggers just described are combined to form a line called, "all other test triggers." What this line means is that all tests, except the check indicator test are complete. If anyone of the preceding tests is not complete, the line "not all other tests complete" is active. The following lookahead test action is determined by the status of the line.

The check indicator test trigger is turned on by two signals from the A checker. One A checker signal "check complete" signifies that the error testing is complete. The other line no "A checker error" indicates there is no error in the checker at that time. Both lines are necessary to turn on the check indicator test trigger. If the A checker discovered an error during the check, the A checker signals to lookahead are delayed until the error is recorded, with the final result being both the check complete and no error lines becoming active to lookahead. If the "not all other test triggers" line is active (all other lookahead tests are not yet complete)

the next sample following the receiving of the signals from the A checker turns on the check indicator test complete trigger to record the completion of the A checker test. If the line "all other test triggers" (all other lookahead test triggers on) is active, the two signal lines from the A checker are combined with it to memorize the interrupt or no interrupt line and the check indicator test trigger is not set. By this latter method a savings in time is realized because the interrupt status is memorized immediately by setting the MAR MODE (no interrupt) or interrupt next instruction triggers directly. If all other tests are not complete, the check indicator test trigger is turned on. Eventually all test triggers come on and all test triggers combine to form a sample ("cond interrupt and MAR triggers) for the interrupt line.

The status of the interrupt line is the result of a comparison between the pre-set mask register and the main indicator register. The coincidence of all test triggers gates the "interrupt" line to turn on the interrupt next instruction tirgger. The coincidence of all test triggers gate the "no interrupt" line to turn on the MAR MODE trigger. Either the interrupt or no interrupt line is active from the comparison so either MAR MODE (no interrupt) or Interrupt next instruction (interrupt) is the final result.

If all other test triggers are on before the check indicator test trigger, the A checker signals combine with the coincidence of the other test triggers to gate the no interrupt line to turn on the MAR MODE trigger directly. The interrupt next instruction trigger is turned on directly if the interrupt line is active. The MAR MODE trigger defines lookahead action for the next instruction. The interrupt next instruction causes lookahead to go into a

housecleaning mode. Housecleaning is discussed in section 8.0.00. The MAR MODE trigger is reset at the IC level of every instruction by the transfer indicator E and not M output of the transfer indicator timer (ABC). This means then that the tests must be made between each instruction. The test triggers are reset at the same time MAR MODE is set.

5.7.02 MAR MODE - Non Arithmetic and NO-OPed Instructions- Figure 5.7-2

If the instruction being processed through lookahead is non arithmetic or NO-OPed arithmetic, the test for MAR MODE just described can not be accomplished because the arithmetic units are not involved. An alternate method is used for memorizing the interrupt status for those instructions not utilizing the arithmetic units. The alternative method uses a NOOP indicator test trigger in combination with the store check indicator test trigger. There are two ways in which the NOOP indicator test trigger can be turned on. A NOOPed level conditions one turn on and the ABC decoding of a non arithmetic instruction provides the second turn on condition for the NOOP indicator test trigger. There is a third turn on condition provided, but it applies to the houseclean mode of operation. Whenever houseclean is over MAR MODE must be turned on to allow the ABC to function for the first instruction following the houseclean recovery action.

The ABC decoding of any non arithmetic instruction results in the line "ABC LA Only MAR NEXT INST." (lookahead is the only unit involved with the MAR test). This ABC decoded line is one condition for turning on the NOOP indicator test trigger. Other conditions are indicator test E (same as turn conditions discussed in 5.7.01) and the lookahead normal indicator test being off (the ABC decoding blocks the turn on of the normal

test trigger - "no ABCLA MAR NEXT INST"). When all conditions are met the NOOP indicator test trigger is turned on. The NOOP indicator test trigger combines with the store check indicator test trigger to memorize the state of the interrupt mechanism.

If any level of any instruction has the NOOP bit on, the transfer indicator timer turns on the NOOP MODE trigger. The NOOP mode trigger blocks all normal tests (arithmetic and non arithmetic) for MAR MODE. The NOOP mode trigger combines with the indicator test E trigger and the not normal test condition to turn on the NOOP indicator test trigger. The NOOP indicator test trigger combines with the store check indicator test trigger to memorize the state of the interrupt mechanism. The MAR MODE trigger is turned on with the no interrupt line or interrupt next instruction is turned on with the interrupt line being active. Even though the instruction is NOOPed the tests must still be made. The result of no-oping the instruction may or may not cause the system to interrupt (function of programming on some indicators). If no interrupt is indicated as a result of no-oping an instruction MAR MODE must still be realized to allow ABC action for the next level. If interrupt is active, the turning on of the interrupt next instruction trigger causes lookahead to enter a houseclean mode for recovery (refer section 8.0.00). At the same time the MAR MODE trigger is turned on, the check triggers are reset. MAR MODE is reset between every linstruction at the IC level by the indicator transfer timer. The test must be made for MAR MODE in order to continue program sequencing.

If the system interrupted, lookahead enters a houseclean mode of operation which consists of returning pseudo store information back to

index core storage and resetting all levels following the one causing the interrupt. At the conclusion of housecleaning, the houseclean timer turns on the NOOP indicator test trigger. This trigger combines with the store check indicator test trigger to memorize the interrupt mechanism to obtain MAR MODE operation for the possible fix up routine. The houseclean action is discussed in section 8.0.00.

6.0.00 LOOKAHEAD STORE OPERATIONS

6.1.00 GENERAL DESCRIPTION

This section discusses all lookahead storing operations to external storage and the I unit. Lookahead store operations to internal registers is discussed in section 5.4.00. The storing operations to external storage and the I unit are accomplished by the Store Check Counter (SCC).

The SCC action is similar to the lookahead OCC action discussed in section 3.0.00. The operation code field of a store level becomes meaningful to the SCC counter when the LF tag is reset. The ABC action at the store level resets the LF tag bit when an external or I unit store is indicated.

When the data to be stored originated from an arithmetic unit, it is always in lookahead parity. The data is already in lookahead at SCC time because the ABC function routed the data from the C or D register to lookahead. The SCC action at this type of level results in first a priority request to the I checker. When the priority (first priority) is satisfied, the SCC routed the data to the I checker in lookahead parity along with the necessary signals for the I checker to check lookahead parity. If the LAAR decoding (figure 5.3-2) indicates a store to external storage, the data is routed back to lookahead in ECC mode. Further SCC action results in a store requests signal being sent to the BCU. When the BCU sends an accept signal to lookahead, the data is routed out on LAM1B for storing. At the same time, the LAAR busy trigger is reset and the store execute trigger is set. The store execute trigger defines the checking conditions for any subsequent forwarding of the data .

An exception to the above procedure exists when address 0 or 4 are specified. Because these are external storage locations, and sources of zeros, the data is converted to zeros with ECC code and the store performed. This action is accomplished by inhibiting the data transfer and check control signal to the I checker. This effectively puts zeros on the output of the I checker and, along with the ECC generated during the check cycle, the data is gated back into lookahead.

The SCC action if the store address is index core storage (XCS) consists of a priority request to the I checker (fourth priority) for the lookahead to the I unit transfer. The data is gated to the I checker with the signal to check lookahead parity. Signals are generated from lookahead to transfer the data from the I checker output to the X register of the I unit. The gate in of the word in the I unit is in I unit parity.

When the I unit requests the contents of an internal register as the operand, the SCC action is almost identical to normal storing action to the I unit. Remember the internal fetch was accomplished during ABC time and the word placed in lookahead. The SCC action is identical except that lookahead generates signals to gate the data to the I unit Y register instead of the X register to complete the fetch for the I unit.

6.2.00 EXTERNAL STORAGE STORES Figure 6.2-2

When stores to external memory are specified, the first SCC Action consists of routing the data through the I checker to check the word for errors. The data transfers is timed with a Store Check timer which is started

by the SCC decoding. Besides the data transfer, signals must be sent to the checker to designate the type of check to perform (ECC or parity). In most of the storing operations, the data word is in lookahead parity and therefore a signal to check lookahead parity is sent to the checker. When the store level is part of a TSMT/SWAP instruction, the data is loaded into lookahead in ECC mode and thus the need for a store check cycle is eliminated. This condition is noted in the SCC decoding by the WBC tag and the LAOP tag bits being present at the same level. In this case, the store request is made directly to the BCU.

If the word is in lookahead parity, a store check cycle is necessary. This action is denoted by the WBC tag or LAOP tag being off at the store level. A priority request must be made to the I checker to accomplish to parity check and check bit conversion before the store request is sent to the BCU.

6.2.01 Priority Request

The operation code field of the store level becomes meaningful to the SCC decoding when the LF tag bit is off. The operation code decoding is shown in figure 6.2-1. A decoded line identifying a store check request combines with the LAAR decoding (identifying the external requirement), an SCC AES trigger (allows one SCC function per level) and an SCC late decode enable (LDE) trigger (allows decoded lines to stabilize) to form the one request to the checker. The checker priority scheme is the same as explained in section 3.0.00. When the one priority is granted, it conditions the turn on of the store check timer E trigger. Other conditions necessary to allow the E trigger to turn on are "no I checker single error" and "no OK to I check". The "no I

single error" insures that the I checker does not require an automatic correct operation from a previous check cycle from another unit and the "no ok to I ck" insures that any I unit checker requests are blocked. These conditions, combining with the #1 priority granted from the I checker priority scheme allows the E trigger of the store check timer to be turned on. The E trigger turns off the SCC AES trigger to block any further timer functions for the level.

6.2.02 Store Check Timer

The store check timer E trigger gates out the lookahead data field to LA ICIB along with a control signal to "ck LAPAR". The sample following the data transfer to the checker allows the store check M trigger to be turned on. With store check timer E and M both on, the IK indicator in the main indicator register is conditioned to turn on if a LA parity error occurs in the I checker test. The M trigger of the store check timer gates the data from ICOB (I check output) back into lookahead and also gates ECC bits in with it. Following the checker test the data is back in lookahead in ECC and the result of the test is recorded in the main indicator register by the IK indicator (on if an error occurred). The M trigger also turns on the store check indicator test trigger to allow the MAR MODE test to be made (store check indicator test trigger was reset during ABC time when the store was realized). PAU is also signalled that the store check has completed by a store check trigger turned on by store check timer M. This allows PAU to make their interrupt test. At the same time the store check timer M trigger turns on the LA store request trigger is also turned on.

An exception to the data transfer and checking occurs when address 0 or 4 are specified as the "store to" location. Because addresses 0 and 4 are external storage locations and a source of zeros, the data in lookahead is converted to zeros with ECC code and the store performed. This is accomplished by inhibiting both the gate out of the data to LAICIB and the request to check parity during the check cycle. To prevent the possibility of an error during the check cycle, the "LA gen PAR" line is sent to the I checker to prevent any comparison check from being made (all inputs to checker from lookahead are inactive). The "LA gen PAR" line is a result of LAAR decoding Address 0 or 4 and the store check timer E and not M triggers. The M trigger functions identically as described previously to complete the store. The data gated in from ICOB is all zeros with ECC code. Because all inputs to the I checker were inactive, the output is effectively all zeros and the check cycle generates the ECC code. Store check timer M gates in the results in ICOB (all zeros) and the ECC code into the lookahead data field.

Another point to be made here is that the store check E and not M triggers participate in setting the forward check M trigger if forward required is ON. This means that the data is routed into the SCC level of lookahead with ECC bits during store check M time and the same data is routed into the IAUC level in lookahead parity during forward check M time. This action is called early forwarding.

6.2.03 Store Request to BCU

The output of the store request trigger is sent to the BCU to request priority for the store. The sample setting the store request trigger also sets the Gate Out LAMIB trigger necessary to gate the lookahead data

field to LAMIB. The store request trigger is a self-resetting trigger and therefore its output gates the following sample pulse to reset the trigger. The Gate Out LAMIB trigger output combines with the SCC value designating the level and puts the data field results on the LAMIB. No further action occurs until the arrival of a store accept pulse from the BCU. The store accept pulse from the BCU turns on the E trigger of the store data timer to time the data transfer. The timing of the data transfer is a matter of keeping the inputs to LAMIB active long enough to allow the storage unit to sample the MIB lines into the cores. This time is allowed for by the store data timer. The E and not M of the timer condition the anticipation circuits for a type 3 load for the level because it is evident that the store is nearly completed. The type 3 load then can be anticipated to allow fast loading. When the store data timer M trigger comes on, the E and M output of the timer provide the reset for the LAAR busy trigger, turn on the store execute trigger to define checking conditions for any subsequent forwarding from the level and E and M also resets the GO LAMIB trigger dropping the lookahead inputs to the LAMIB. Sufficient time has been allotted to store the data, the M trigger also allows the SCC advance to occur conditioned of course by the interlock between the SCC and ABC. If the interlock is active, the M trigger of the timer remains on until the interlock disappears, thereby keeping the advance circuits conditioned until the interlock disappears. The advance of the SCC counter also sets the AES trigger back on. The AES on condition allows the reset of the M trigger of the store data timer. The external store is complete.

In the case of a TSMT/SWAP instruction which specifies an external location, the data is already in ECC mode when loaded. Because of this, the necessity for a store check cycle is eliminated. The store check cycle is prevented by the LAOP and WBC tag bits being on at the level. In this example the store request trigger is turned on directly by the SCC decoding to complete the store. The turn on of the store request trigger is the SCC AES trigger, SCC LDE trigger, and the WBC tag identifying the condition. When the store request trigger comes on, the action is the same as already described.

6.3.00 STORE TO THE I UNIT Figures 6.3-1 and 2

When the SCC decoding designates a store or internal fetch to the I unit, the SCC Action varies considerably from that already described. The first action is a priority request for the I check. Data transfers to the I Unit have fourth priority in the I checker scheme. The SCC decoding of an I Unit store or I unit internal fetch causes a 4th priority request to the checker. As soon as the priority is granted, an LA to I Timer is conditioned to start. Other conditions necessary to start the timer are indications that any I unit priority requests are blocked (no OK to I Ck LA") and that I checker is not going into an automatic correct cycle from another units check cycle (no sing I ckr error). With these conditions satisfied and the priority granted, the E trigger of the LA to I timer is turned on.

6.3.01 LA To I Timer

The purpose of the timer is to time the data transfer to the I unit. The E trigger combines with the SCC value (designates level) to gate out the lookahead data field to the LAICIB. The E and not M combination signal the I checker to check the data for lookahead parity errors. Because the I

checker output feeds both the I unit and lookahead, it is not necessary to gate the data back into lookahead following the check cycle. Instead, the LA to I timer can time the transfer from the checker out bus (ICOB) directly to the I Unit. Before this can occur however, further SCC decoded lines must be available to designate the area in the I Unit that is to receive the data. There are two possible storage locations in the I Unit. One is index core storage where the normal I Unit store data is sent to and the other is the Y register, which receives the external fetch word. To facilitate the transfer to two different I unit locations, the LA to I timer has two M triggers. One M trigger times the transfer to the X register for index core storage stores and the other M trigger times the data transfer to the Y register for internal fetch type instructions. The E trigger thus far has timed the data transfer to the I checker and sent the appropriate signal to check the LA parity. All that remains is the turning on of the appropriate M trigger to time the data from the checker to the I Unit.

6.3.02 Internal Fetch Requests Figure 6.3-1

The SCC decoding of not an index core storage store combines with LA to I E trigger to turn on "I timer no trans to index" M trigger. Because the I timer is started and the SCC decoding classified no index core storage store, the only other available destination is the Y register. With the M trigger on, it combines with the E trigger to provide the timing for the data transfer. All lookahead to I unit transfers other than index core storage are accomplished with this particular M trigger.

The data is gated into the Y register with I timer E and an SCC decoded line identifying the no index core storage store. E and M of the timer condition the MK indicator in the indicator register to set if a lookahead parity error is encountered during the I checker test. Also if the SCC decoding indicated an I unit internal fetch, the IDC trigger in the I Unit is conditioned to set if a lookahead parity error is encountered in the checker test. The IDC trigger in the I unit results in NOOP ing the instruction requesting the internal register word because the parity error indicates errors in the required operand. The M trigger allows the SCC to advance contingent upon the SCC-ABC interlock (normal stepping conditions and resets are performed). Since this is not a real store the store execute trigger is not set and the LAAR busy trigger is reset as a function of ABC time.

6.3.03 Store To Index Core Storage Figure 6.3-2

If the SCC decoding designated a store to index core storage, the LA to I timer index transfer M trigger is set. The data is sampled into the X register with the E trigger and an SCC decoded line identifying the index core storage store. The E and M trigger outputs combine to set the store check test trigger to inform PAU of the store and turns on a Clear Index E trigger. Two timers, clear index and write index time the transfer from the X register to the cores. Clear index E is essentially used as a store request signal to the index core storage controls. Clear index E and M is used to reset the LAAR busy trigger, and turn on the store check indicator test trigger and Set the store executed trigger. The write index M and no ABC-SCC interlock allow normal SCC advancing and timer resets. The timers are necessary to

allow sufficient time for the index core storage controls to sample the X register data into the cores. The store is complete.

7.0.00 INSTRUCTION REJECT ACTION

7.1.00 Conditions for NOOP

The action described to this point allows processing the entire computer instruction set under normal operating conditions. It is possible, upon detection of certain conditions during preparation, to reject the instruction. These conditions cause the No Op tag bit to be set in the level involved and include:

1) Abnormalties detected during the Instruction Unit preparation, prior to loading the lookahead, such as addresses failing the boundary comparison circuits, invalid operation codes, errors, etc.

2) An operand address failing the Memory checking circuits.

3) The operand itself failing the I-Checker circuits.

7.2.00 Lookahead Action - Figure 7.2-1

Under control of the No Op bit, normal functions are inhibited and only the necessary interrupt system sequencing functions are performed. The presence of this bit at any level through the last (fetch) operand level associated with a given instruction causes the entire instruction to be rejected. Beyond this point, the No Op bit will not appear since the execution units are allowed to modify addressable registers.

During the ABC action (Transfer Indicators $E \cdot \overline{M}$) at a No Oped level, the No Op MODE trigger is set and remains on to define the action through the last level associated with the instruction.

TBC action consists of merely advancing over a rejected instruction on the basis of No Op MODE, $\overline{IC_i} \cdot TBC_i$. At the last level, no action occurs until the ABC has completed action at this same level, at which time the TBC advance into the next instruction and the No Op MODE trigger reset occur simultaneously.

The ABC action during instruction rejection includes the transfer of the Instruction Exception and the CNIDC indicators, with the No Op Indicator Test trigger recording the last transfer. In addition, upon detecting a store level during this mode of operation, the LAAR B trigger is reset and the No Op tag bit set. Thus, if necessary, the Forward No Op Conversion cycle may take place to allow the com to continue.

7.2.01 PAU Instruction Reject

In general, the PAU is not allowed to begin pre-execution of an instruction to be rejected. The exception occurs when the special operand associated with the PAU Multiply and ADD instruction is No Oped. In this case, the PAU MPYC Reject pulse is given in lieu of the PAU Continue signal, and the instruction terminated prior to modifying an addressable register.

7.2.02 SAU Instruction Reject

The d.c. output of the SAU Reject trigger serves to cause rejection of SAU instruction. This signal is given only when housekeeping has actually begun and is given only at the first No Oped operand level. The SAU Enabled Memory trigger set by the ABC (Indicator Transfer E·M) at the first SAU level (if not rejected) remembers the first condition above. The logic for setting SAU Reject is, then, SAU Enabled Memory, No Op_i, ABC_i, No No Op MODE, Indicator Transfer $E \cdot \overline{M}$. The trigger is reset by the SAU NOT OPERATING condition indicating instruction rejection prior to the modification of an addressable register. The SAU Enabled Memory trigger is reset at the last level with the sample selected by Indicator Transfer $E \cdot M$, ABC_i, IC_i.

7.2.03 Pseudo Interrupt

Four levels exist which normally cause the Index Result indicators to be changed. These are:

1) Instruction Unit Store level
2) Indicator Transfer Only level
3) Instruction Unit Pseudo Store level
4) SAU-PX Pseudo Store level

When an instruction resulting in any of the above levels is rejected, the transfer of these indicators is inhibited at the ABC level; thus, the Instruction Unit record of these indicators is in error. The validity of any subsequent levels appearing in the lookahead is in question and House-cleaning action is initiated to recover. This recovery is referred to as a "pseudo interrupt".

Following this action, the applicable bit positions of the Indicator Register are transferred "back" to correct the Instruction Unit Updated Indicator Register, the contents of the IC Buffer are transferred to the Instruction Counter and the Instruction Unit resumes by re-preparing instructions from this point.

7.2.04 No Op Code

The interpretation placed on the No Op tag bit prohibits the bit setting beyond the last fetch operand level of SAU instructions. However, errors detected during the loading of 1) pseudo store data for the progressive

indexing level and 2) the branch address into the branch recovery level must be recorded and appropriate action taken. Only if this recovery data is to be used is this error significant. The error line is gated into Operation Code position 8 and the CNIDC indicator. The former action converts these particular Lookahead Operation Codes to "No Op Codes" and the level is handled consistent with instruction reject action. When it is determined a recovery is necessary, the CNIDC indicator is transferred to the MK position of the Indicator Register.

8.0.00 LOOKAHEAD HOUSECLEAN ACTION

8.1.00 Conditions for Houseclean

The Housecleaning mode of operation may be necessitated by any of three conditions:

1. Detection of program interrupt,
2. Detection of the need for "pseudo interrupt",
3. Detection of the need for "branch recovery".

In each case, the levels beyond that at which the ABC initiates the Housecleaning mode are considered invalid. Lookahead loading is terminated immediately and, at any level containing pseudo store data, the operand field is transferred to Index Core Storage to "back date" this memory area. This action continues until all counters are interlocked with the IAUC and the IAUC Advance Enables Sequence trigger is on indicating all levels have been processed. The necessary lookahead status triggers are then reset to allow resumption of normal action before allowing the loading to begin.

8.2.00 Housecleaning due to an Interrupt – Figure 8.2-1

This action is initiated due to the Interrupt Next Instruction trigger indicating the latest memorization of the interrupt line. The sample selected by this trigger, ABC Advance Enables Sequence, ABC_i, $LF_i \cdot LC_i$ sets E trigger of the Houseclean Timer which is used (in place of the Indicator Transfer timer) throughout the remaining levels during this mode of operation. At this time, synchronization with the Instruction and Execution Units occurs to define the status of the computer.

The sample selected by Houseclean Timer E·M, No LA Houseclean Mode. Interrupt Next Instruction: 1) sets the Interrupt Inhibits Load trigger the output of which inhibits all Load Enable signals to the Instruction Unit. -2) sets the LA Disable Interrupt trigger which insures the memorization of no new interrupt until this recovery is complete. -3) sets the I Houseclean Request trigger informing the Instruction Unit of the recovery. This trigger is reset by that unit upon initiation of that action necessary to insure availability of the registers necessary to receive recovery data from the lookahead. Completion of this action is indicated by the d.c. signal LA Houseclean Request from the Instruction Unit.

When the instruction at this level-i,e., the instruction being interrupted-is the first level of a PAU instruction (specifying a non-internal operand and is not rejected), it is evident that the PAU will begin preexecution. This units action upon detection of the interrupt condition, is to terminate pre-execution and enter an "Idle" state such that no further instructions may begin. (This is necessary since it cannot be guaranteed that the TBC has not given the GI Op Code & Start signal which applies to the next instruction. Neither the PAU nor the SAU may honor a Start signal from the lookahead while the PAU is in this "Idle" state.)

The pulse, Set Execution Units Idle from the PAU indicates this condition and is sampled into the Execution Unit Idle trigger. Coincidence of this trigger and Lookahead Houseclean Request sets the LA Houseclean Mode trigger which controls the action at the remaining levels of lookahead.

When the interrupted instruction is the first level of an SAU instruction (not rejected), it is evident the SAU will begin its Housekeeping activity. This ABC decode line, Houseclean Timer E·M, and No LA Houseclean Mode sets SAU Instruction Interrupt as temporary storage. This trigger and SAU Housekeeping sets the SAU Reject trigger which again informs that unit to terminate prior to modification. of an addressable register. SAU Not Operating, SAU Reject, Interrupt Inhibits Load·SAU Instruction Interrupt is used to set Execution Units Idle.

When the ABC decodes any level other than the first level of PAU or SAU instructions, the Execution Units Idle trigger is set directly (Houseclean Timer E·M) by the lookahead such that the LA Houseclean Mode trigger will be set in the usual manner after the Instruction Unit completes the necessary recovery.

By this method, then, once LA Houseclean Mode is present, the state of all other units involved has been established and the lookahead may proceed to restore any necessary data. (It should be noted that LA Houseclean Mode inhibits normal TBC action, so until this trigger comes on, the TBC continues to function normally. This should justify the concept of the necessary execution units "idle" state.

The LA Houseclean Mode provides a continuous advance condition for the TBC such that this counter will advance through to the IAUC level contingent only upon the TBC-OCC interlock. At each level to be housecleaned, the ABC action is restricted to: Figure 8.2-2.

1) Transfer only the CNIDC indicator to the Indicator Register to record errors in recovery information.

2) Reset LAAR Busy and set the No Op tag bit at levels requiring use of the LAAR. (This again allows the Forward No Op Conversion Cycle if forwarding has been specified.) This action is performed by the sample selected by LAAR Busy, ABC_i, $From_i$, Houseclean Timer E·M.

3) Resetting the LF tag bit in pseudo store levels to allow the SCC to return the index word. This is done with the ABC decode line and Houseclean Timer E·$\overline{M}$.

The advance condition of the ABC during housecleaning is Houseclean Timer M, LA Houseclean Mode and no interlock ABC-TBC.

The SCC action of returning the index word is contingent upon LA Houseclean Mode and is very similar to effecting a valid store to Index Core Storage. The address involved, however, is buffered in positions 1-4 of the Operation Code Field.

The process of returning this recovery information must be accomplished in "reverse" order--i.e.--the "earliest" original contents of any one Index Core Storage location is the only information to be restored. Thus, during the return of data, the address involved is compared with all other pseudo store addresses in lookahead and the comparison signal gated into Operation Code position O of the level compared with. This new lookahead Operation Code is then interpreted by the SCC as merely an advance condition.

The ABC and SCC action continues until all counters indicate the same (IAUC) value indicating the lookahead is empty. With the sample selected by $IAUC_i$, OCC_i, TBC_i, ABC_i, SCC_i, IAUC Advance Enables Sequence, and LA Houseclean Mode, the Houseclean Over Timer is used to reset the various computer status triggers to allow resumption of normal action.

Houseclean Over E $\overline{M}$ selects a sample to: Figure 8.2-3

1) Gate the contents of the IC Buffer to the Instruction Counter. This is a direct path.

2) Gate the Index Result indicator position of the Indicator Register to the Instruction Unit Updated Indicator Register. This corrects the latter information for local interrogation.

3) Reset the PAU and SAU Start triggers. This removes the false start which may have been set up by the TBC before LA Houseclean Mode terminated this counters normal action.

4) Reset lookahead MAR MODE and the PAU Master Tests Complete triggers in preparation for re-memorizing the interrupt line.

5) Reset LA Houseclean Mode and Interrupt Inhibit Load triggers to allow further lookahead loading. (This sample also resets Pseudo Interrupt Inhibits Load and Branch Recovery Inhibits Load used during these types of housecleaning action.)

Houseclean Over E·M selects a sample to:

1) Set the PAU Wait trigger. This essentially terminates the PAU "Idles" state allowing this unit to accept new instructions.

2) Set the No Op Indicator Test trigger which allows re-memorization of the interrupt line in the usual manner. Since the LA Disable Interrupt trigger is on, this memorization results in MAR MODE and MAR Next Instruction.

Houseclean Over Timer-M forms a d.c. signal to the Instruction Unit to resume. The first action includes "generating" a special Branch on Indicator instruction (IRPT-BIN) to accomplish resetting the bit which caused the interrupt. The conditions, branch if bit is off (F/N) and set to zero (L/Z) are generated. Thus the SAU accomplishes the bit resetting and the lookahead action is identical to that of the Branch on Indicator instruction with the Branch Unsuccessful response. (The TBC decode line at the first level is memorized by the SAU in lieu of any SAU op code.)

The next instruction seen by the lookahead is the "free" instruction stored at the address specified by the sum of the Interrupt Address and the bit address of the bit causing the interrupt.

The LA Disable Interrupt trigger is reset by MAR Next Instruction, Indicator Transfer timer E·M and inhibited by the ABC decode line identifying the first level of the IRPT-BIN instruction. In this manner, it is insured that both the IRPT-BIN instruction and the free instruction will be executed by the lookahead before a new interrupt may be honored.

(An additional Interrupt Test level exists for the special purpose of forcing interrogation of the interrupt mechanism. In certain cases, (notably the Transmit or Swap instructions) it is desirable for the Instruction Unit to complete stores to the Index Core Storage area without recourse to loading successive pseudo store levels since the depth of lookahead limits the degree of recovery possible. This may be accomplished only upon proving logically that the instruction is not to be interrupted.

Prior to taking this action, an Interrupt Test level is loaded into the lookahead, after which the Instruction Unit waits for the Lookahead

Empty signa. This signal is formed by MAR MODE, or IRPT NEXT INSTRUCTION, No Forward Required, the absence of all Inhibit Load triggers, all counters interlocked, IAUC Advance Enables Sequence, and the fact that the level designated by the IAUC is not disconnected.

The normal action taken at this level is seen to be similar to that of the Indicator Transfer Only level with the Index Result indicators unaffected.)

8.3.00 Housecleaning due to Pseudo Interrupt – Figure 8.3-1

This action is necessary upon detection of a rejected instruction affecting the Index Result indicators. The action is initiated during ABC action at the last (IC) level of the rejected instruction, thus insuring the address of the next instruction appears in the IC Buffer.

The sample selected by the ABC decode line, and Transfer Indicators $E \cdot \overline{M}$ sets the Pseudo Interrupt Inhibits Load and the I Houseclean Request triggers. Following the Instruction Unit recovery, the sample selected by LA Houseclean Request, Pseudo Interrupt Inhibits Load and No LA Houseclean Mode sets LA Houseclean Mode.

The lookahead action once the Mode trigger is on is identical to that previously described. The Instruction Unit resumes, however, by merely re-preparing the instruction specified by the contents of the IC Buffer. (The Inhibit Load triggers serve as memory of the particular type of housecleaning action and define the method of resumption necessary for the Instruction Unit.)

8.4.00 Housecleaning due to a Branch Recovery- Figure 8.3-1

This action is necessary upon detection of the Instruction Unit making a false assumption in attempting to process the Branch on Bit/Indicator instructions. The action is initiated by the ABC at the branch recovery level associated with these instructions.

The Branch Recovery Inhibits Load trigger terminates the loading process, and the I Houseclean Request signal is given. LA Houseclean Request from the Instruction Unit, Branch Recovery Inhibits Load and No LA Houseclean Mode sets the LA Houseclean Mode trigger. SCC action of returning the branch address[2] is contingent upon this Mode trigger (since LF was reset by the ABC "early".) The LA to I timer is used to time the transfer with the data destination of a preselected Y register.

The recovery at the remaining levels is identical to that for housecleaning due to a "real" interrupt or pseudo interrupt. However, upon completion of this recovery, the Instruction Unit resumes by preparing the instruction specified by the Branch Address and continues from this point.

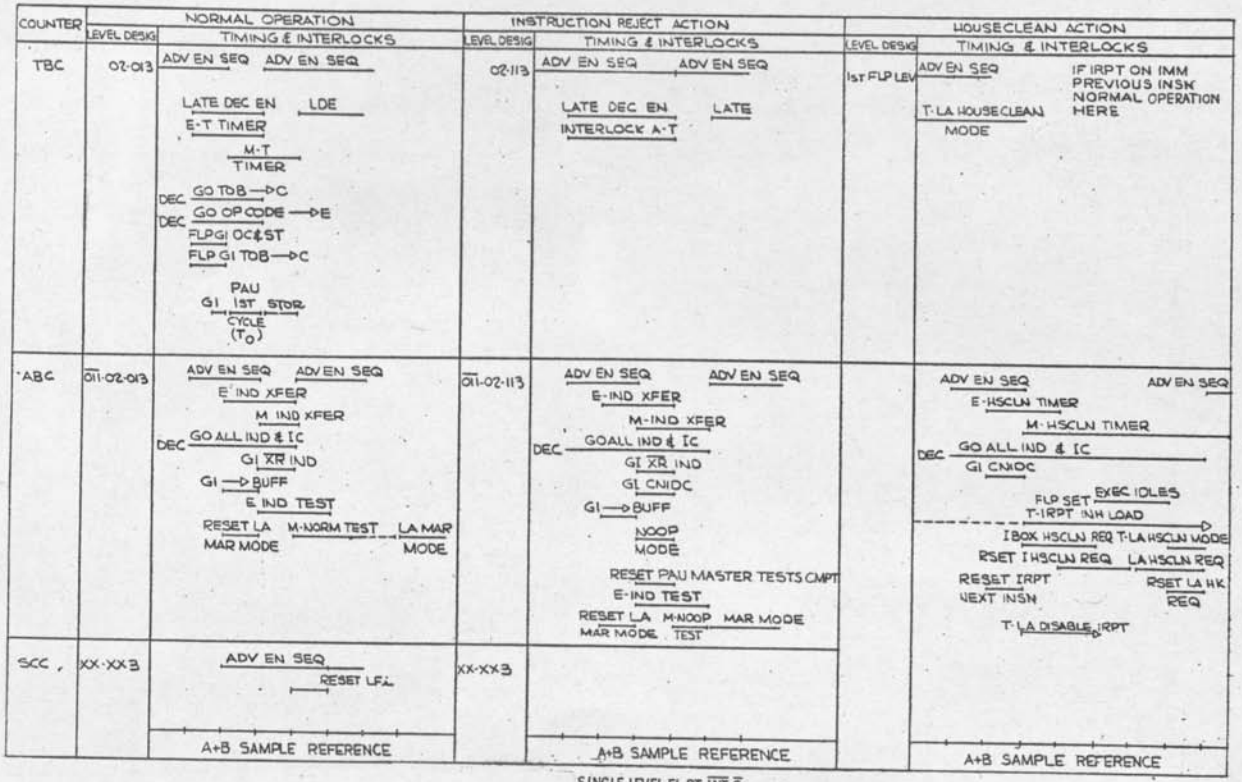
Regardless of the type of recovery, the reset accom - plished with Houseclean Over Timer essentially allows normal lookahead action to continue when the Instruction Unit resumes the loading process.

9.0.00 MAINTENANCE MODE

Each of the four lookahead levels has a disconnect switch associated with it. The switches are physically located on the 7030 maintenance console. When any of the switches are turned on, the level to which to switch is associated is "disconnected" from the system. The action performed by the switch is a blocking function of all operations within the level. If any level is disconnected the sequencing counters at the level are all unconditionally allowed to step to the next level and all operations in the level are blocked. Effectively, the 7030 system operates without the level. By using the switches, any level or combination of levels can be disconnected. If all four switches are on, the system does not operate because there is in effect no lookahead. If trouble is expected in any one level, the other three switches can be turned on and effectively limit lookahead to a one level operation which can be easily checked.

In addition to the disconnect switches lookahead OCC action is also altered when the maintenance console panel keys are addressed as an operand in the maintenance mode of operation. The fetch, controlled by the BCU, returns the data with no associated ECC bits to lookahead. To identify this type of operation, the I unit codes the preparation tag bits by setting internal, not LF and not LC. The storage select results in internal, LF and not LC. The OCC signal to the I checker to check ECC is suppressed and another signal substituted to merely generate parity and residue. Finally, the operand check E and M forms the normal coding of such a level as not internal, LF and LC by

resetting the internal tag bit as well as setting LC. For consistency the internal tag bit is always reset by operand check E and M.

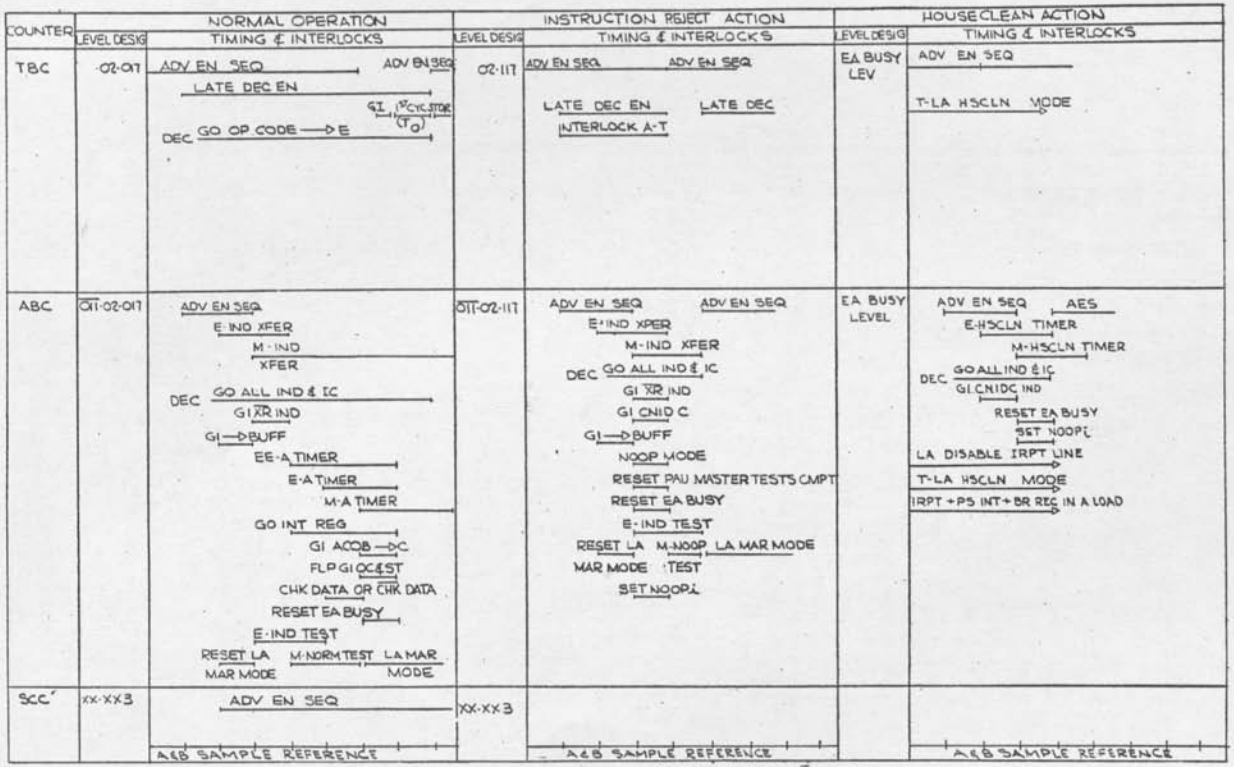


SINGLE LEVEL FL PT INT'S

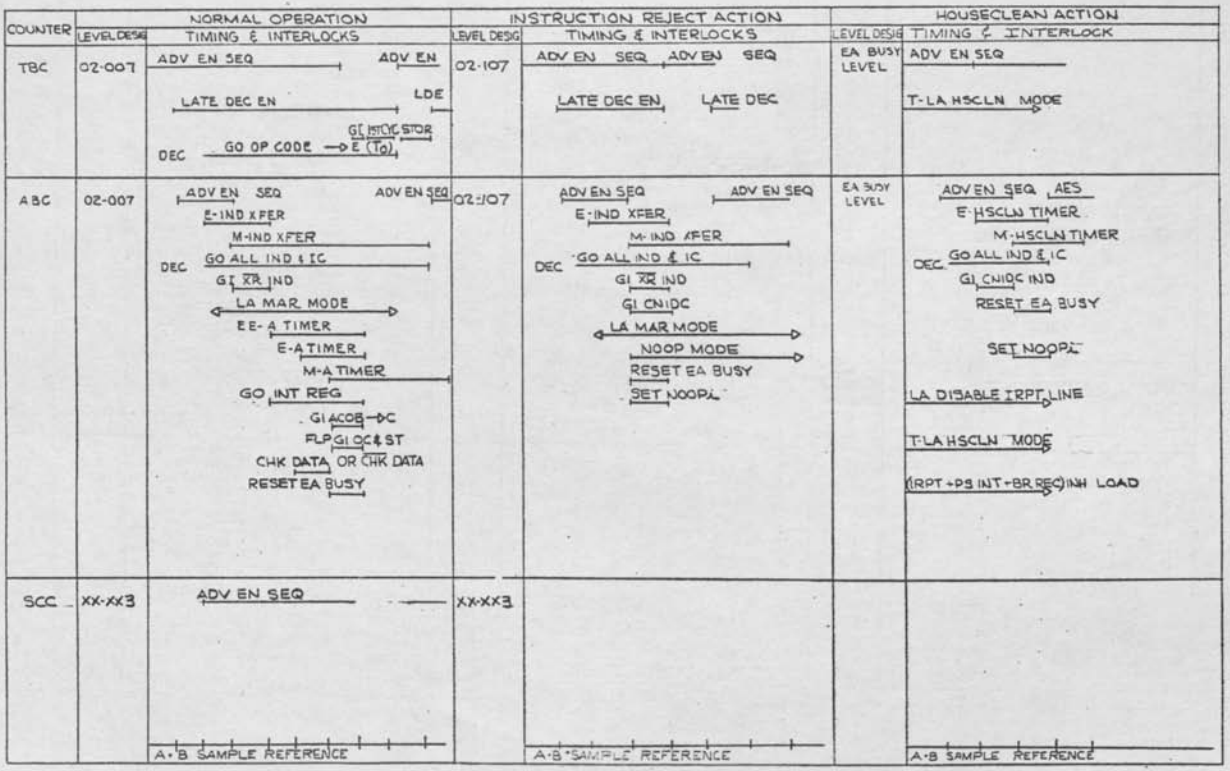
88

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMER & INTERLOCKS
TBC	02-003	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE DEC E-T TIMER M-T TIMER DEC GO TOB → C DEC GO OP CODE → E FLP GI OC↑CT FLP GI TOB↑C GI 1ST CYC MEM (GO)	02-103	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE DEC	1st FLP LEV	ADV EN SEQ T-LA HOUSECLEAN MODE
ABC	02-003	ADV EN SEQ — ABC ADV E IND X FER M IND X FER DEC GO ALL IND & IC GI $\overline{XR}$ IND LA MAR MODE	02-103	ADV EN SEQ — ADV EN SEQ E-IND X FER M-IND X FER DEC GO ALL IND & IC GI XR IND GI CNIOC LA MAC MODE NOOP MODE		ADV EN SEQ — AN EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIOC IND FLP SET — EXEC IDLES T-IRPT INH LOAD I BOX HSCLN REQ — T-LA HSCLN MODE RESET I HSCLN REQ — LA HSCLN REQ RESET IRPT NEXT INSTR — RESET LA HK REQ T-LA DISABLE IRPT
SCC	XX-XX3	ADV EN SEQ — AES RESET LFL	XX-XX3			
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

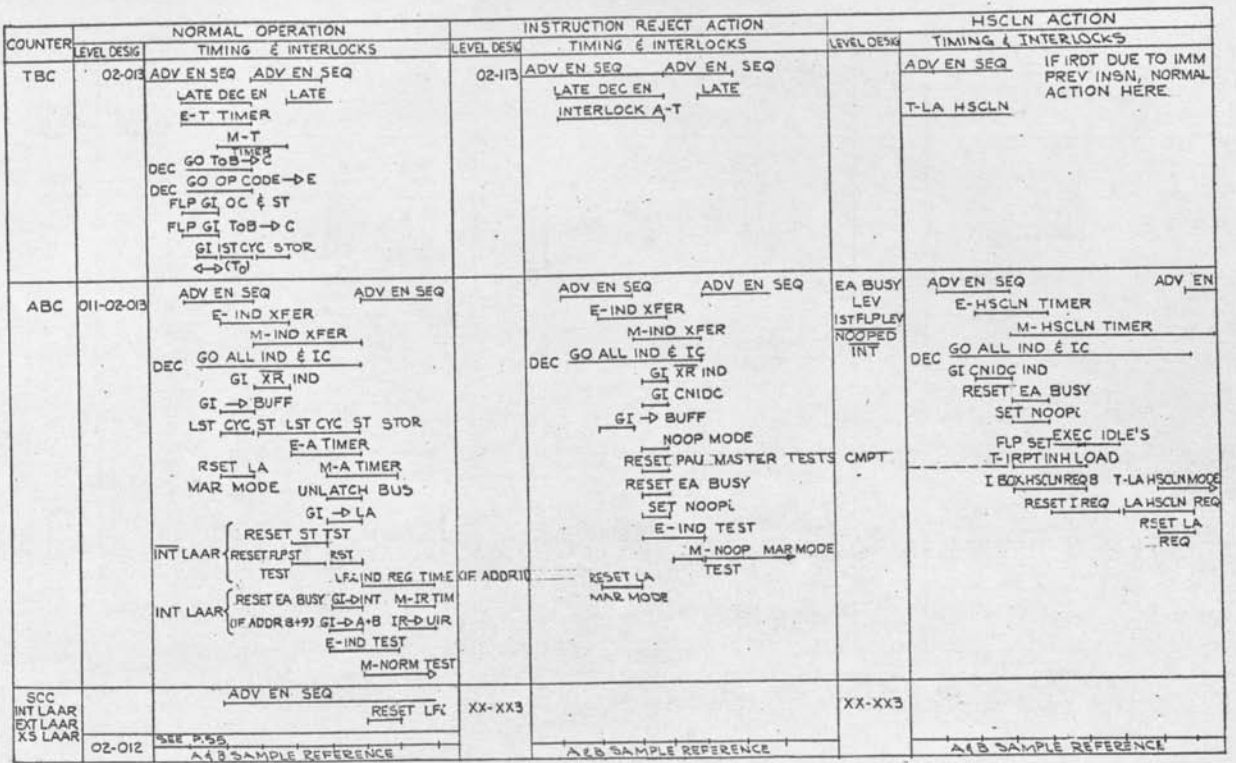
FL PT 1st LEV INT



SINGLE LEVEL FLP INTS



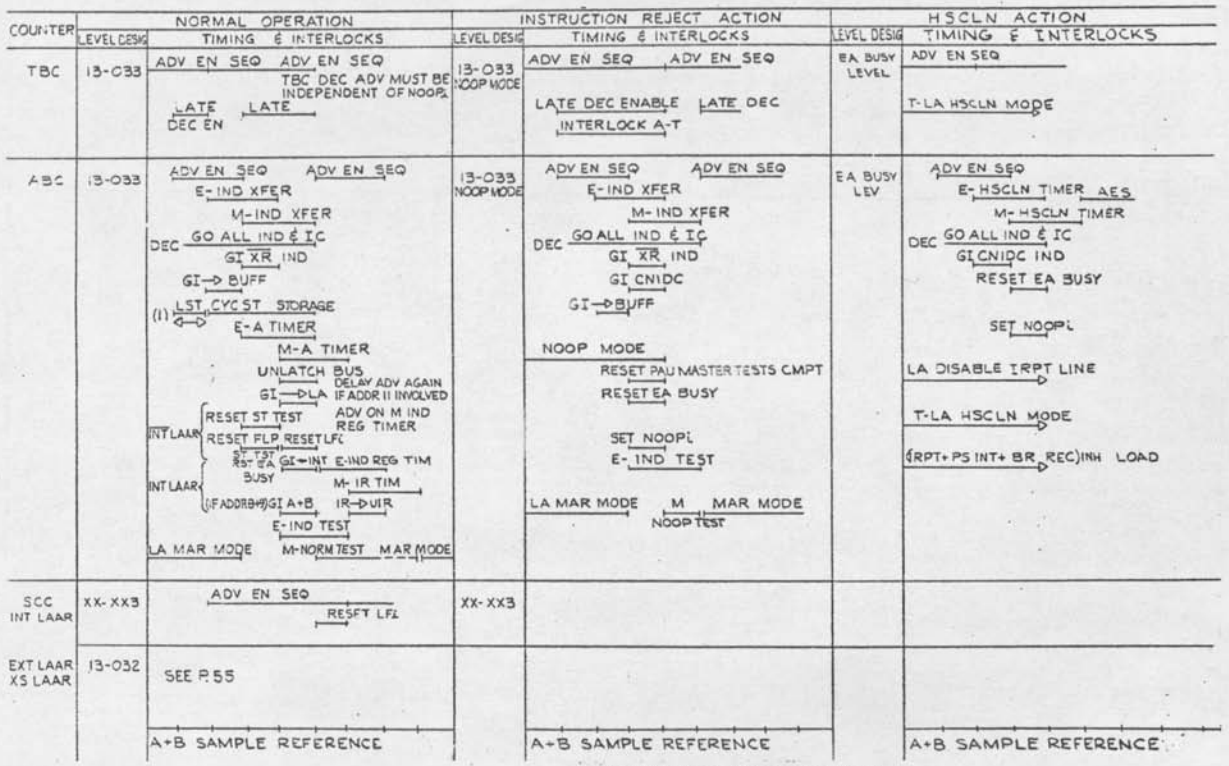
FL PT 1st LEVEL INT



SINGLE LEVEL FLP STORE

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HSCLN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	11-033	ADV EN SEQ — ADV EN SEQ LATE DEC EN E-T TIMER M-T TIMER DECODE GO TO B → C FLP CONT FLP GI TO B-C GI INTM CYCLE STOR	11-133 11-033 NOOP MODE	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE INTERLOCK A-T	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE
ABC	11-033	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI $\overline{XR}$ IND GI → B BUFF E-IND TEST MAR MODE — M-NORM TEST	11-133 11-033 NOOP MODE	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC GI → B BUFF NOOP MODE FLP INSN REJ RESET PAU MASTER TESTS CMPT E-IND TEST LA MAR MODE — M-NOOP MAR MODE TEST	NO SPEC ACTION	ADV EN SEQ — ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT & PS INT & BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ RESET LFL				
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

FLP MPYC OPERAND

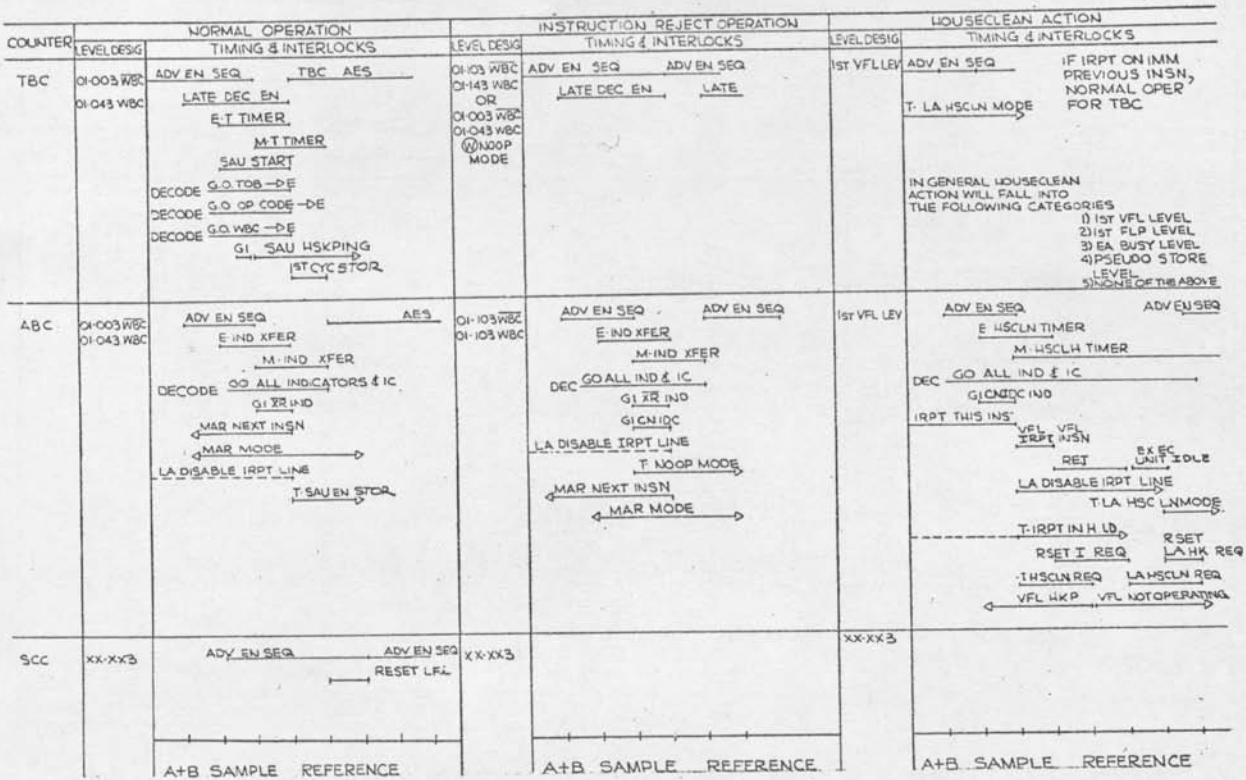


FLP 2nd LEV STORE BUS NORMAL TERMINATION

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEANING ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCK
TBC	13-033	ADV EN SEQ ADV EN SEQ TBC DEC ADV MUST BE INDEPENDENT ON NOOP'S LATE DEC EN	13-033 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC ENABLE LATE DEC INTERLOCK A-T	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	13-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI-> BUFF (2) NO ST DATA STOR E A TIMER M A TIMER RESET EA BUSY E-IND TEST LA MAR MODE M-NORM TEST NOOP1	13-033 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI-> BUFF NOOP MODE RESET PAU MASTER TESTS CMPT RESET EA BUSY SET NOOP1 E IND TEST LA MAR MODE M- MAR MODE NOOP TEST	EA BUSY LEV	ADV EN SEQ AES E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOP1 LA DISABLE IRPT LINE T=LA HSCLN MODE (IRPT+PS INT+BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ RST LF A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

FLP 2nd LEV STORE BUS NO STORE TO DZ IND

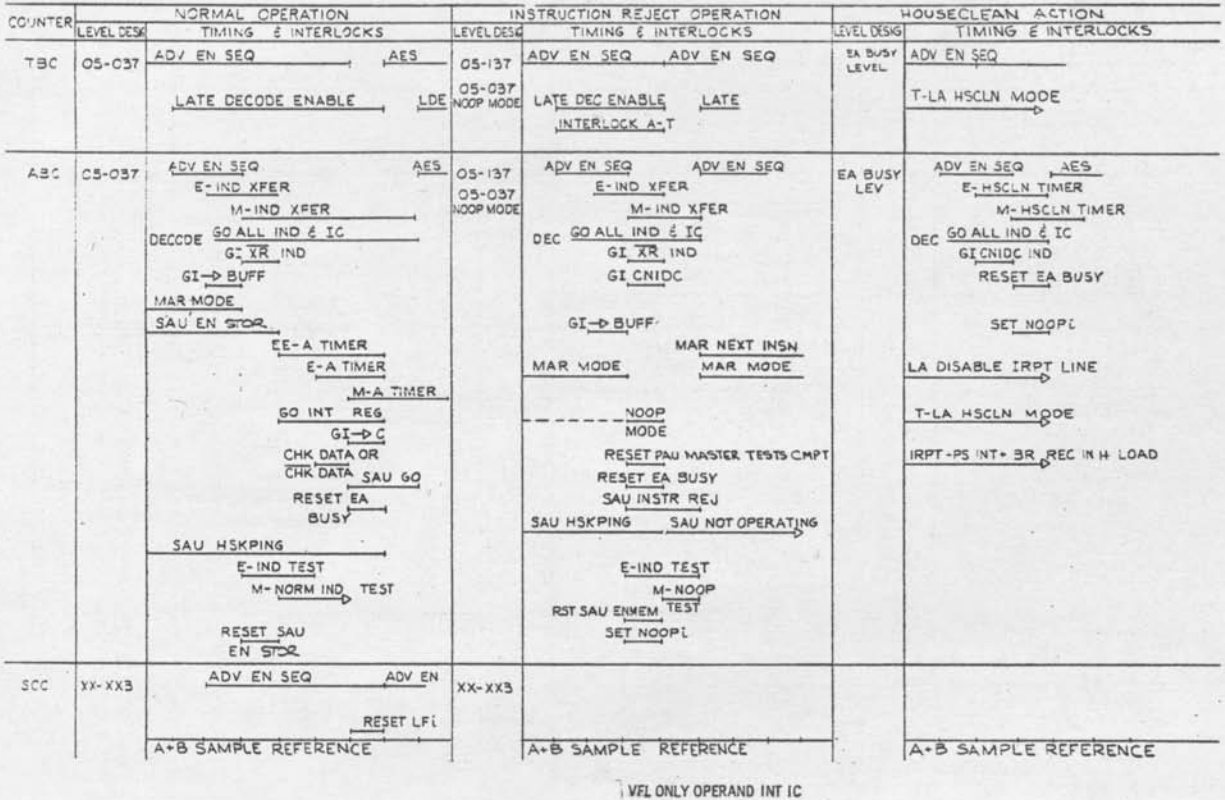
68

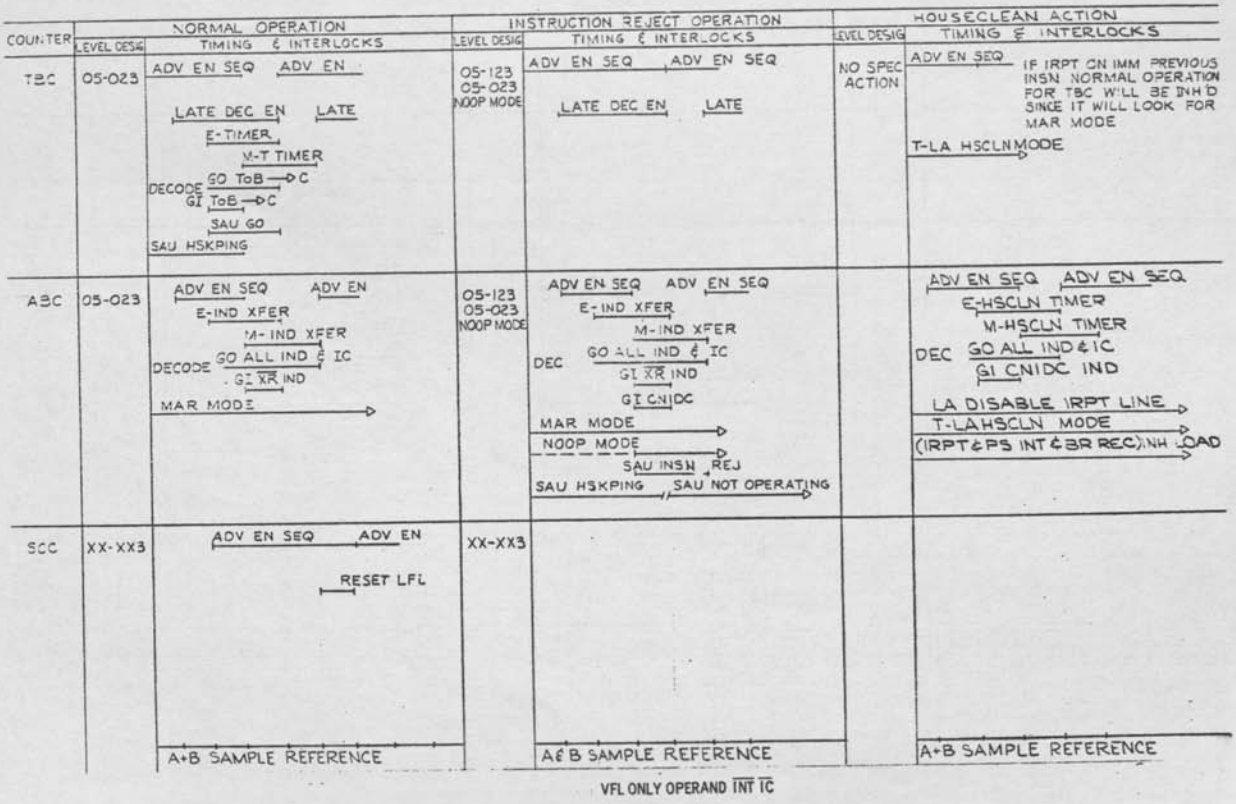


LEVEL FUNCTIONS VFL OP CODE LEVEL

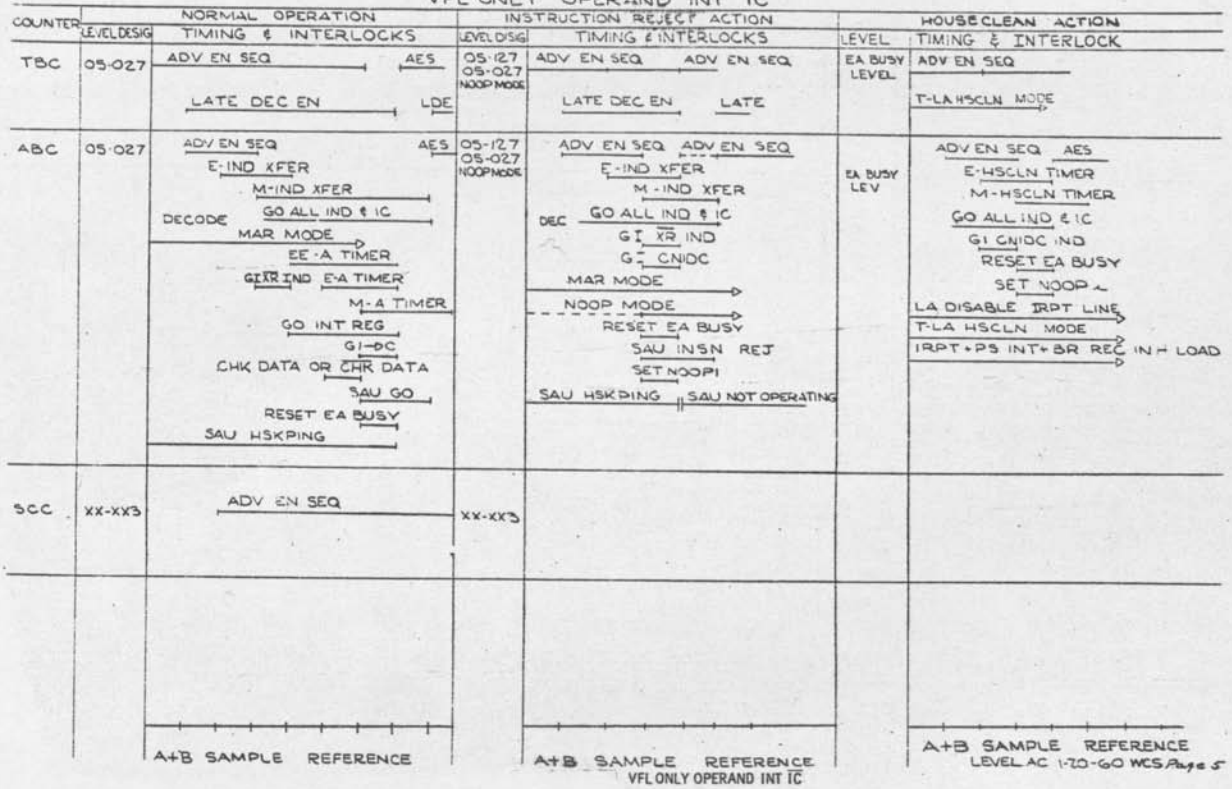
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT OPERATION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	05-033	ADV EN SEQ ADV EN SEQ LATE DEC EN E-T TIMER M-T TIMER DECODE GO TOB → C GI TOB → C VFL GO VFL HSKEEPING	05-133 05-033 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE INTERLOCK A-T	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE IF IRPT ON IMM PREVIOUS INSN NORMAL OPERATION FOR TBC WILL BE INHIBITED SINCE IT MUST LOOK FOR MAR MODE
ABC	05-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND GI BUFF MAR MADE E-IND TEST M-NORM IND TEST SAU EN STOR RESET SAU EN STOR	05-133 05-033 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XPER M-IND XFER DECODE GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF MAR NEXT INSN MAR MODE MAR MODE T-NOOP MODE RESET PAU MASTER TESTS CMPT E-IND TEST SAU EN MEM M-NOOP TEST RESET SAU EN STOR SAU INSTR REJ SAU HSKPING SAU NOT OPERATING		ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DECODE GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT+PS INT+BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ ADV EN RESET LFL A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE

VFL ONLY OPERAND $\overline{\text{INT}}$ IC'


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
05-037
ADV EN SEQ
AES
LATE DECODE ENABLE
LDE
05-137
05-037
NOOP MODE
ADV EN SEQ
ADV EN SEQ
LATE DEC ENABLE
LATE
INTERLOCK A-T
EA BUSY LEVEL
ADV EN SEQ
T-LA HSCLN MODE
ABC
05-037
ADV EN SEQ
AES
E-IND XFER
M-IND XFER
DECCDE
GO ALL IND & IC
GI XR IND
GI → BUFF
MAR MODE
SAU EN STDR
EE-A TIMER
E-A TIMER
M-A TIMER
GO INT REG
GI → C
CHK DATA OR
CHK DATA
SAU GO
RESET EA
BUSY
SAU HSKPING
E-IND TEST
M-NORM IND TEST
RESET SAU
EN STDR
05-137
05-037
NOOP MODE
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC
GO ALL IND & IC
GI XR IND
GI CNIDC
GI → BUFF
MAR NEXT INSN
MAR MODE
MAR MODE
NOOP
MODE
RESET PAU MASTER TESTS CMPT
RESET EA BUSY
SAU INSTR REJ
SAU HSKPING
SAU NOT OPERATING
E-IND TEST
M-NOOP
RST SAU ENMEM TEST
SET NOOPI
EA BUSY LEV
ADV EN SEQ
AES
E-HSCLN TIMER
M-HSCLN TIMER
DEC
GO ALL IND & IC
GI CNIDC IND
RESET EA BUSY
SET NOOPI
LA DISABLE IRPT LINE
T-LA HSCLN MODE
IRPT-PS INT+ BR REC IN H LOAD
SCC
XX-XX3
ADV EN SEQ
ADV EN
RESET LFI
A+B SAMPLE REFERENCE
XX-XX3
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
VFL ONLY OPERAND INT IC


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
05-023
ADV EN SEQ
ADV EN
LATE DEC EN
LATE
E-TIMER
M-T TIMER
DECODE GO ToB → C
GI ToB → C
SAU GO
SAU HSKPING
05-123
05-023
NOOP MODE
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
NO SPEC ACTION
ADV EN SEQ
IF IRPT ON IMM PREVIOUS INSN NORMAL OPERATION FOR TBC WILL BE INH'D SINCE IT WILL LOOK FOR MAR MODE
T-LA HSCLN MODE
ABC
05-023
ADV EN SEQ
ADV EN
E-IND XFER
M-IND XFER
DECODE GO ALL IND & IC
GI XR IND
MAR MODE
05-123
05-023
NOOP MODE
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
GI XR IND
GI CNIDC
MAR MODE
NOOP MODE
SAU INSN REJ
SAU HSKPING
SAU NOT OPERATING
ADV EN SEQ
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC GO ALL IND & IC
GI CNIDC IND
LA DISABLE IRPT LINE
T-LA HSCLN MODE
(IRPT & PS INT & BR REC) INH LOAD
SCC
XX-XX3
ADV EN SEQ
ADV EN
RESET LFL
XX-XX3
A+B SAMPLE REFERENCE
A & B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
VFL ONLY OPERAND INT TC

VFL ONLY OPERAND INT $\overline{IC}$

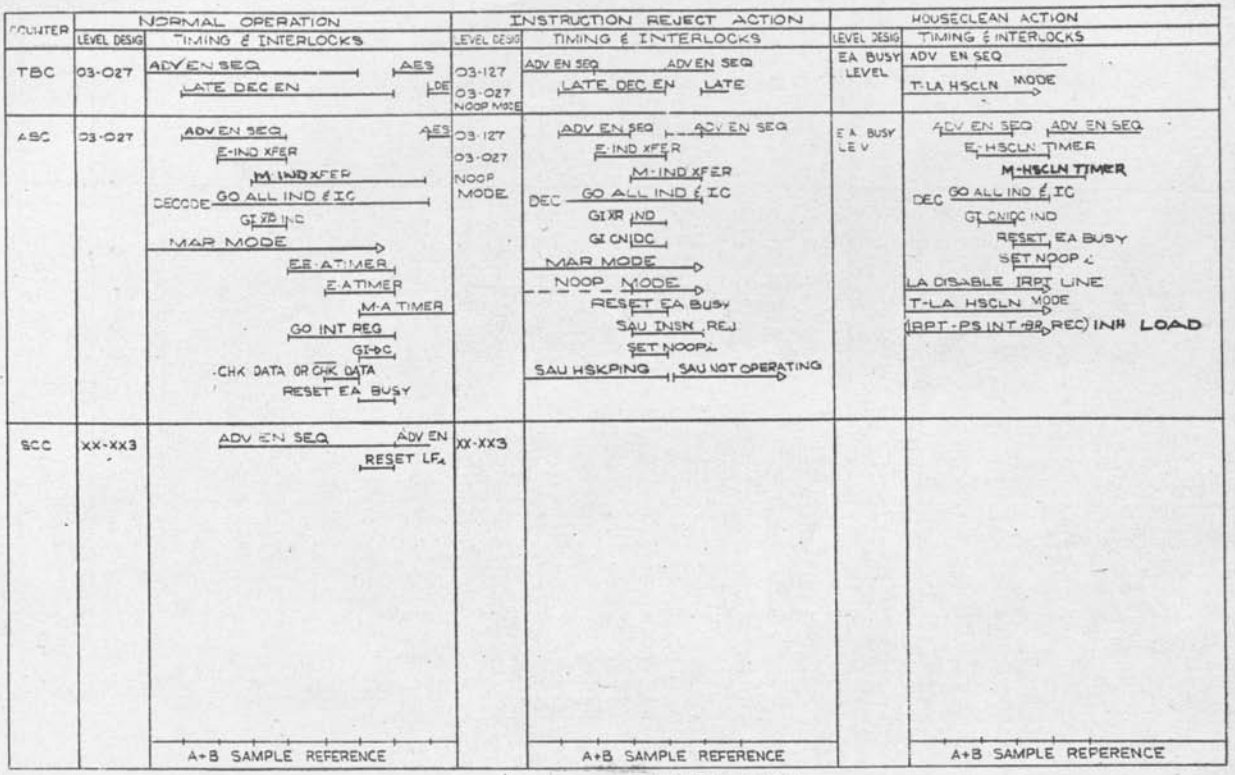


VFL ONLY OPERAND INT $\overline{IC}$

LEVEL AC 1-20-60 WCS Page 5

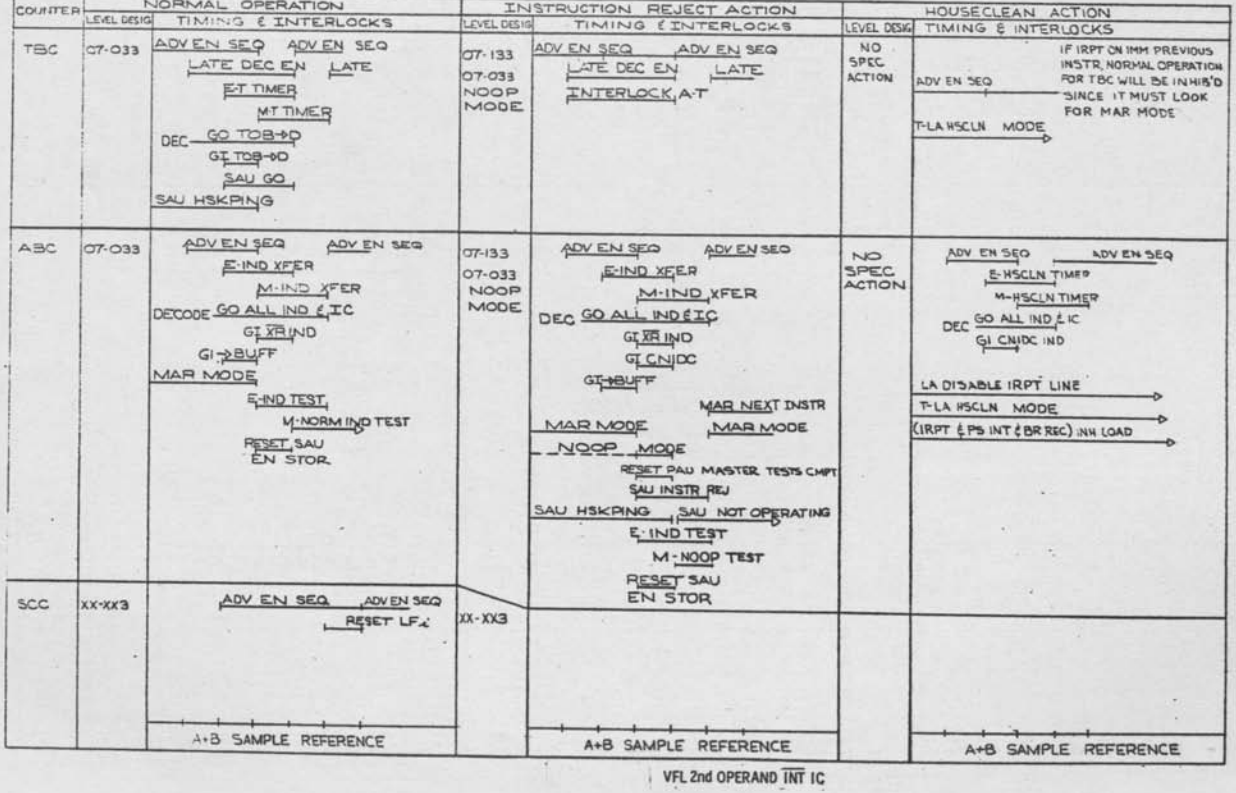
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	03-023	ADV EN SEQ — ADV EN SEQ LATE DECEN — LDE E-T TIMER M-T TIMER DEC GO TOB → DC GI TOB → DC	03-123 03-023 NOOP MODE	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE IF IRPT ON IMM PREVIOUS INST NORMAL OPERATION FOR TBC WILL BE INHB SINCE IT MUST LOOK FOR MAR MODE
ABC	03-023	ADV EN SEQ — ADV EN E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND MAR MODE	03-123 03-023 NOOP MODE	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC MAR MODE NOOP MODE SAU INSTR REJ SAU HSKPING SAU NOT OPERATING		ADV EN SEQ — ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT & PS INT & BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ — ADV EN SEQ RESET LFL	XX-XX3			
		A+B SAMPLE REFERENCE		A+B SAMPE REFERENCE		A+B SAMPLE REFERENCE

VFL 1st OPERAND INT



VFL 1st OPERAND INT

96



VFL 2nd OPERAND INT IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	07-037	ADV EN SEQ — AES LATE DEC EN — LDE	07-137 07-037 NOOP MODE	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE INTERLOCK A-T	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	07-037	ADV EN SEQ — AES E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND GI BUFF MAR MODE EE-A TIMER E-A TIMER M-A TIMER GO INT REG GI→DO CHK DATA OR CHR DATA SAU GO RESET EA BUSY SAU HSKPING E-IND TEST M-NORMAL IND TEST RESET SAU EN STOR	07-137 07-037 NOOP MODE	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI→DBUFF MAR NEXT INSTR MAR MODE — MAR MODE NOOP MODE RESET SAU MASTER TESTS CMPT RESET EA BUSY SAU INSTR REJ SAU HSKPING — SAU NOT OPERATING E-IND TEST M-NOOP MODE SET NOOP RESET SAU EN STOR	EA BUSY LEV	ADV EN SEQ — AES E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOP A DISABLE IRPT LINE T-LA HSCLN MODE (IRPT+PS INT+BR REC) NH LOAD
SCC	XX-XX3	ADV EN SEQ — ADV EN SEQ RESET LF2	XX-XX3			
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL 2nd OPERAND INT IC

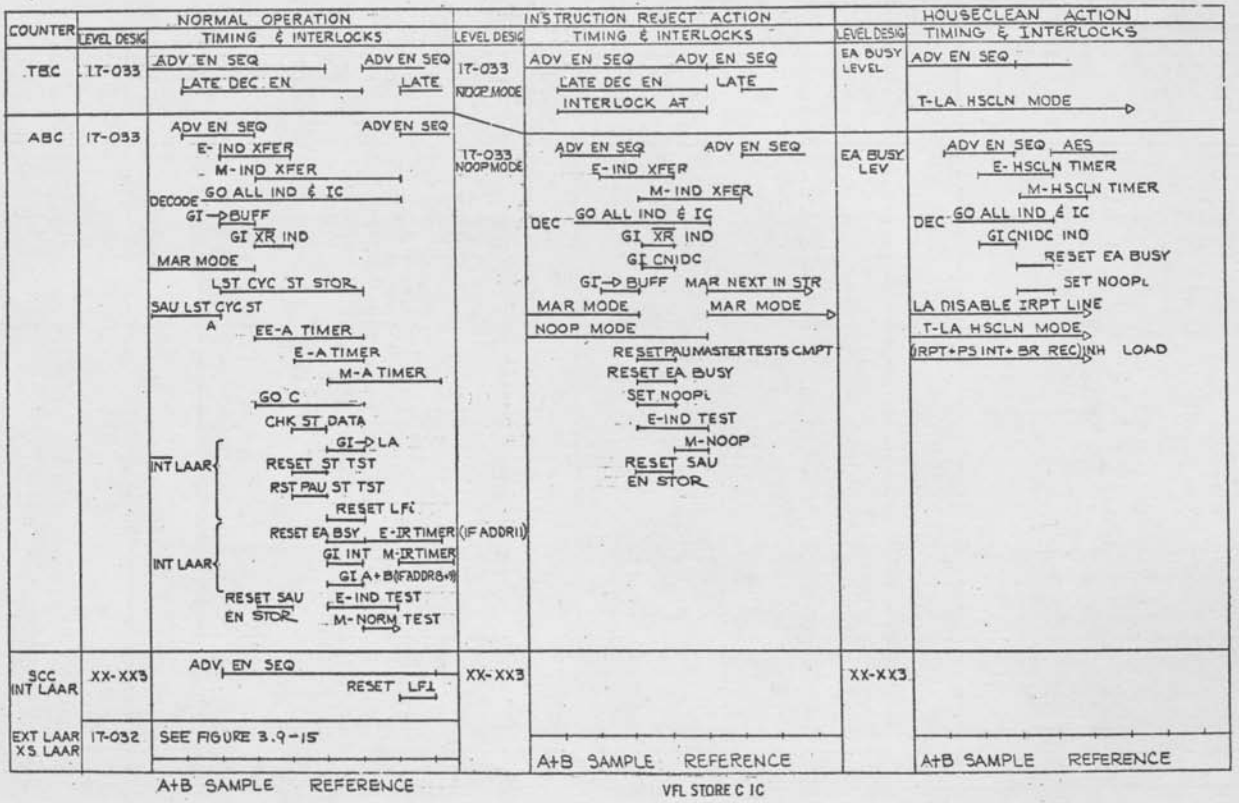
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	07-023	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC E-T TIMER M-T TIMER DECODE GO TOB→DP GI→DD SAU GO SAU HSKPING	07-123 07-023 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE
ABC	07-023	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND MAR MODE	07-123 07-023 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC MAR MODE NOOP MODE SAU INSTR REJ SAU HSKPING SAU NOT OPER	NO SPEC ACTION	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT & PS INT & BR REQ) INH LOAD
SCC	XX-XX3	ADV EN SEQ ADV EN SEQ RESET LFL	XX-XX3			
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL 2nd OPERAND INT IC

Page 10

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	07-027	ADV EN SEQ — AES LATE DEC EN	07-127 07-027 NOOP MODE	ADV EN SEQ - ADV EN SEQ LATE DEC EN — LATE	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	07-027	ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI $\overline{XR}$ IND MAR MODE EE-A-TIMER E-A TIMER M-A TIMER GO INT REG CHK DATA CHR DATA GI→D RESET EA BUSY SAU GO SAU HSKPING	07-127 07-027 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC MAR MODE NOOP MODE SAU INSTR REJ RESET EA BUSY SET NOOPL SAU HSKPING — SAU NOT OPERATING	EA BUSY LEV	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOPL LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT+ PS INT+ BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL 2nd OPERAND INT $\overline{IC}$

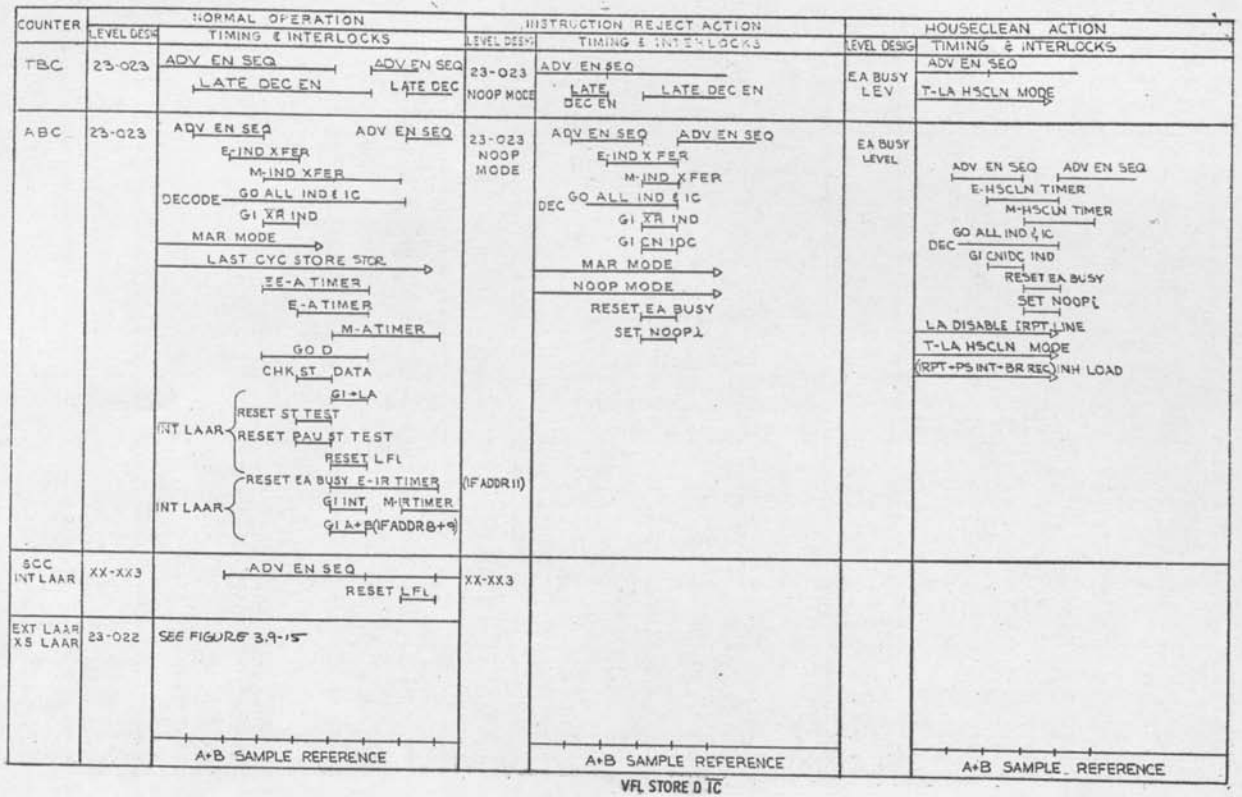

COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT ACTION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
17-033
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
17-033
NOOP MODE
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
INTERLOCK AT
EA BUSY LEVEL
ADV EN SEQ
T-LA HSCLN MODE
ABC
17-033
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DECODE
GO ALL IND & IC
GI → BUFF
GI XR IND
MAR MODE
LST CYC ST STOR
SAU LST CYC ST A
EE-A TIMER
E-A TIMER
M-A TIMER
GO C
CHK ST DATA
GI → LA
INT LAAR
RESET ST TST
RST PAU ST TST
RESET LFL
RESET EA BSY
E-IR TIMER
(IF ADDR1)
INT LAAR
GI INT M-IR TIMER
GI A+B(IFADDR8+9)
RESET SAU EN STOR
E-IND TEST
M-NORM TEST
17-033
NOOP MODE
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC
GO ALL IND & IC
GI XR IND
GI CNIDC
GI → BUFF
MAR NEXT IN STR
MAR MODE
MAR MODE
NOOP MODE
RESET PAU MASTER TESTS CMPT
RESET EA BUSY
SET NOOPL
E-IND TEST
M-NOOP
RESET SAU EN STOR
EA BUSY LEV
ADV EN SEQ AES
E-HSCLN TIMER
M-HSCLN TIMER
DEC
GO ALL IND & IC
GI CNIDC IND
RESET EA BUSY
SET NOOPL
LA DISABLE IRPT LINE
T-LA HSCLN MODE
(IRPT+PS INT+ BR REC) INH LOAD
SCC INT LAAR
XX-XX3
ADV EN SEQ
RESET LFL
XX-XX3
XX-XX3
EXT LAAR XS LAAR
17-032
SEE FIGURE 3.9-15
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
VFL STORE C IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	17-023	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE	17-023 NOOP MODE	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE DEC EN	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	17-023	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DECODE — GO ALL IND & IC GI $\overline{XR}$ IND MAR MODE STOR LST CYC STD — EE-A TIMER A — E-A TIMER M-A TIMER GO C CH ST DATA GI→LA $\overline{INT}$ LAAR — RESET ST. TEST RST PAU ST TEST RESET LFL RESET EA BUSY E-IR TIMER INT LAAR — GI INT M-IR TIMER GI A+B (IF ADDR B+9) IF BB, ONE OF 3 RESPONSES SET Ⓜ LST CYC ST (SEE P.26)	17-023 NOOP MODE (FADDR11)	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC — GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC MAR MODE NOOP MODE RESET EA BUSY SET NOOPL	EA BUSY LEV	ADV EN SEQ — AES E-HSCLN TIMER M-HSCLN TIMER DEC — GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOPL LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT+PS INT+BR REC) INH LOAD
SCC INT LAAR	XX-XX3	ADV EN SEQ — ADV EN SEQ RESET LFL	XX-XX3		XX-XX3	
EXT LAAR XS LAAR	17-022	SEE FIGURE 3.9-15 A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL STORE C $\overline{IC}$

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	23-033	ADV EN SEQ — ADV EN SEQ LATE DECODE EN — LATE DEC EN	23-033 NOOP MODE	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE INTERLOCK A-T	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	23-033	ADV EN SEQ — ADV EN SEQ E-IND XFER M-INDXFER DECODE — GO ALL IND & IC GI XR IND GI → BUFF MAR MODE LST CYC ST STOR EE-A TIMER E-A TIMER M-A TIMER GO D CHK ST DATA INT LAAR: GI → LA RESET ST TEST RST PAU ST TEST RESET LFL INT LAAR: RESET EA BUSY E-IR TIMER GI INT M-IR TIMER GI A+B (IF ADDR B+9) E-IND TEST RESET SAU EN STOR — M-NORM TEST	23-033 NOOP MODE (IF ADDR I)	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER DEC — GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF — MAR NEXT INSTR MAR MODE — MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT RESET EA BUSY SET NOOPL E-IND TEST M-NOOP TEST RESET SAU EN STOR	EA BUSY LEV	ADV EN SEQ — ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC — GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOPL LA DISABLE IRPT LINE T-LA HSCLN MODE (RPT+PS INT+BR REC) INH LOAD
SCC INT LAAR	XX-XX3	ADV EN SEQ RESET LFL A+B SAMPLE REFERENCE	XX-XX3		XX-XX3	
EXT LAAR IND LAAR	23-032	SEE FIGURE 3.9-15		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL STORE D IC


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT ACTION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
23-023
ADV EN SEQ
LATE DEC EN
ADV EN SEQ
LATE DEC
23-023
NOOP MODE
ADV EN SEQ
LATE DEC EN
LATE DEC EN
EA BUSY LEV
ADV EN SEQ
T-LA HSCLN MODE
ABC
23-023
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DECODE
GO ALL IND & IC
GI XR IND
MAR MODE
LAST CYC STORE STOR
EE-A TIMER
E-A TIMER
M-A TIMER
GO D
CHK ST DATA
GI→LA
INT LAAR
RESET ST TEST
RESET PAU ST TEST
RESET LFL
INT LAAR
RESET EA BUSY E-IR TIMER
GI INT M-IR TIMER
GI A+B(IFADDR B+9)
23-023
NOOP
MODE
(IFADDR 11)
ADV EN SEQ
ADV EN SEQ
E-IND X FER
M-IND XFER
DEC
GO ALL IND & IC
GI XR IND
GI CN IDC
MAR MODE
NOOP MODE
RESET EA BUSY
SET NOOP↓
EA BUSY LEVEL
ADV EN SEQ
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC
GO ALL IND & IC
GI CNIDC IND
RESET EA BUSY
SET NOOP↓
LA DISABLE IRPT LINE
T-LA HSCLN MODE
(RPT+PS INT+BR REQ) INH LOAD
SCC INT LAAR
XX-XX3
ADV EN SEQ
RESET LFL
XX-XX3
EXT LAAR XS LAAR
23-022
SEE FIGURE 3.9-15
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
VFL STORE D IC

COUNTER	NORMAL OPERATION: LEVEL DESIG	NORMAL OPERATION: TIMING INTERLOCKS	INSTRUCTION FAULT ACTION: LEVEL DESIG	INSTRUCTION FAULT ACTION: TIMING INTERLOCKS	HOUSECLEAN ACTION: LEVEL DESIG	HOUSECLEAN ACTION: TIMING & INTERLOCKS
TBC	21-033	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC EN	21-033 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN, LATE INTERLOCK A-T	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	21-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND GI → BUFF MAR MODE 1ST CYC STOR STORE A E-A TIMER M-A TIMER UNLATCH BUS INT LAAR { GI LA; RESET ST TEST; RESET RST IFL PAU ST TST INT LAAR { E-I R TIMER (IF ADDR 11); STEA BUSY GI INT, M-IR TIMER; GI A+B (IF ADDR 8+9) E-IND TEST M-NORM TEST RESET SAU EN STOR	21-033 NOOP MODE	ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF MAR NEXT INSTR MAR MODE MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT RESET EA BUSY SET NOOPi E-IND TEST E-NOOP TEST RESET SAU EN STOR	EA BUSY LEVEL	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOPi LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT+PS INT+BR REC) INH LOAD
SCC INT LAAR	XX-XX3	ADV EN SEQ RESET LFi				
EXT LAAR XS LAAR	21-032	SEE FIGURE 3.9-15				
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

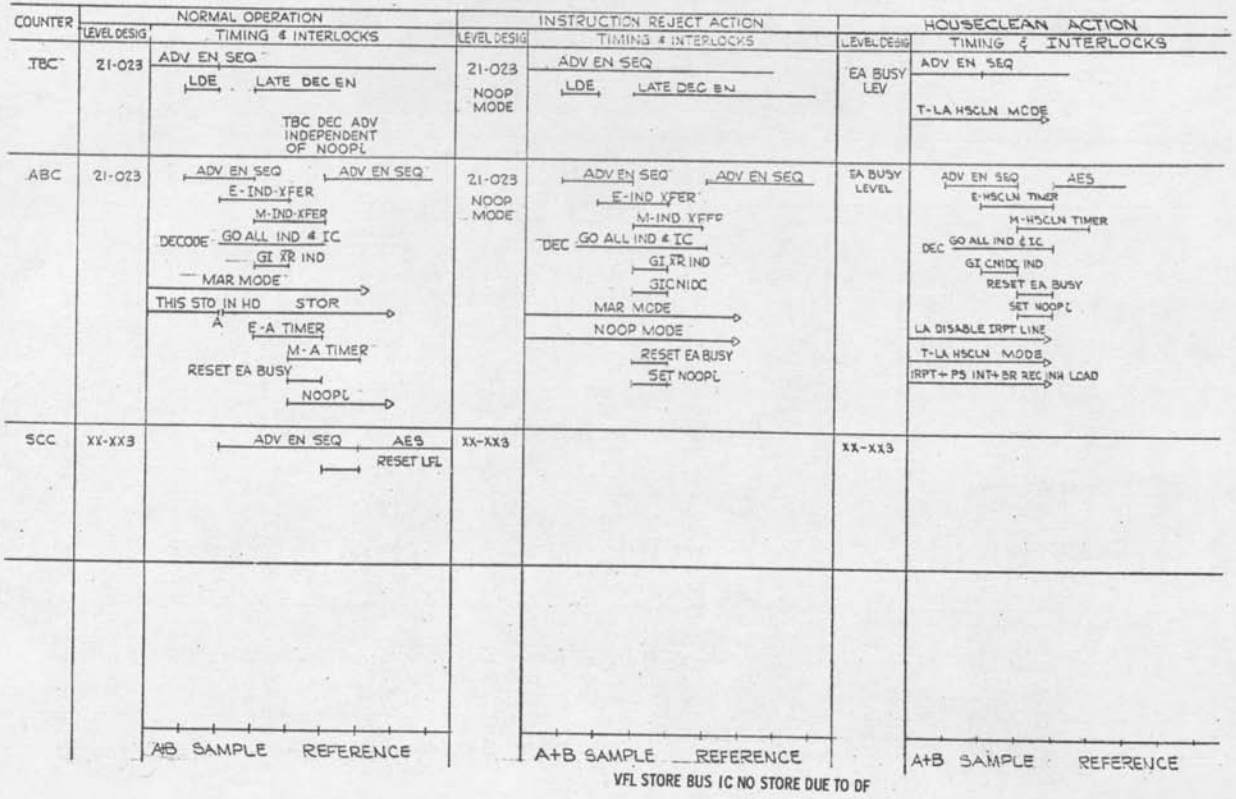
VFL STORE BUS IC NORMAL TERMINATION

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	21-033	ADV EN SEQ ADV EN SEQ TBC DEC ADV REGARDLESS OF NOOP LATE LATE DEC EN	21-033 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE INTERLOCK A-T	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	21-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND GI → BUFF MAR MODE STOR THIS STORE INHD E-A TIMER M-A TIMER RESET EA BUSY E-IND TEST M-NORM TEST NOOP↓ RESET SAU EN STOR		ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF MAR NEXT INSTR MAR MODE MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT RESET EA BUSY SET NOOP↓ E-IND TEST M-NOOP TEST RESET SAU EN STOR	EA BUSY LEVEL	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOP↓ LA DISABLE IRPT LINE T-LA HSCLN MODE (RPT+PS INT+ BR REC) INH LOAD
SCC	XX-XX3	ADV EN SEQ ADV EN SEQ RESET LFL A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE

VFL STORE BUS $\overline{IC}$ NO STORE DUE TO DZ

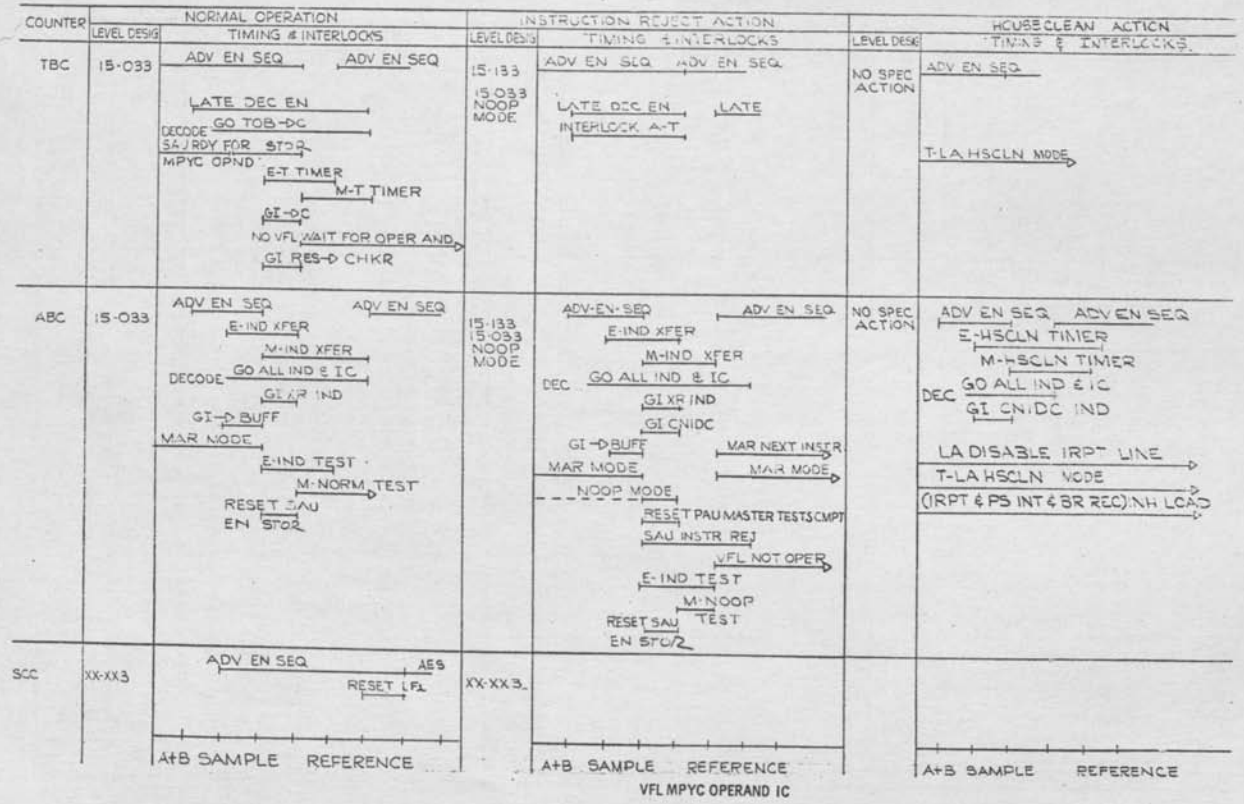
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	21-023	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC EN	21-023 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DC EN LATE DEC EN	EA BUSY LEVEL	ADV EN SEQ T-LA HSCLN MODE
ABC	21-023	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI XR IND MAR MODE (1) STOR LAST CYC STORE E-A TIMER M-A TIMER UNLATCH BUS GI-LA INT LAAR: RESET ST TEST; RST PAU ST RESET LFL; TST E-IR TIMER(IFADDR11) INT LAAR: RESET EA BUSY GI INT M-IR TIMER; GI→A+B(IFADDR 8+9)	21-023 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC MAR MODE NOOP MODE RESET EA BUSY SET NOOPL	EA BUSY LEV	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOPL LA DISABLE IRPT LINE T-LA HSCLN MODE (RPT+PS INT+BR REC) INH LOAD
SCC INT LAAR	XX-XX3	ADV EN SEQ RESET LFL	XX-XX3			
EXT LAAR XS LAAR	21-022	SEE FIGURE 3.9-15				
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL STORE BUS $\overline{IC}$ NORMAL TERMINATION



VFL STORE BUS IC NO STORE DUE TO DF

801


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT ACTION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
15-033
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
DECODE GO TO B-DC
SA / RDY FOR STOR
MPYC OPND
E-T TIMER
M-T TIMER
GI-DC
NO VFL WAIT FOR OPER AND
GI RES-D CHKR
15-133
15-033 NOOP MODE
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
INTERLOCK A-T
NO SPEC ACTION
ADV EN SEQ
T-LA HSCLN MODE
ABC
15-033
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DECODE GO ALL IND & IC
CLR IR IND
GI-D BUFF
MAR MODE
E-IND TEST
M-NORM TEST
RESET SAU EN STOR
15-133
15-033 NOOP MODE
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
GI XR IND
GI CNIDC
GI-D BUFF
MAR NEXT INSTR
MAR MODE
MAR MODE
NOOP MODE
RESET PAU MASTER TESTS CMPT
SAU INSTR REJ
VFL NOT OPER
E-IND TEST
M-NOOP TEST
RESET SAU EN STOR
NO SPEC ACTION
ADV EN SEQ
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC GO ALL IND & IC
GI CNIDC IND
LA DISABLE IRPT LINE
T-LA HSCLN MODE
(IRPT & PS INT & BR REC) INH LOAD
SCC
XX-XX3
ADV EN SEQ
AES
RESET LFL
XX-XX3
A+B SAMPLE
REFERENCE
A+B SAMPLE
REFERENCE
A+B SAMPLE
REFERENCE
VFL MPYC OPERAND IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		INTERRUPT ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	15-023	ADV EN SEQ; ADV EN SEQ; LATE DEC EN; LATE DEC; DEC; GO TOB → C; SAU RDY FOR MPYC OPER AND; STOR; E-T TIMER; M-T TIMER; GI → C; NO SAU WAIT FOR OPERAND; GI RES → CHKR	15-123 15-023 NOOP MODE	ADV EN SEQ; ADV EN SEQ; LATE DEC EN; LATE DEC EN	NO SPEC ACTION	ADV EN SEQ; T-LA HSCLN MODE
ABC	15-023	ADV EN SEQ; ADV EN SEQ; E-IND XFER; M-IND XFER; DECODE; GO ALL IND & IC; GI $\overline{XR}$ IND; MAR MODE	15-123 15-123 NOOP MODE	ADV EN SEQ; ADV EN SEQ; E-IND XFER; M-IND XFER; DEC; GO ALL IND & IC; GI $\overline{XR}$ IND; GI CNIPC; MAR MODE; NOOP MODE; SAU INSTR REJ; SAU NOT OPERATING	NO SPEC ACTION	ADV EN SEQ; ADV EN SEQ; E-HSCLN TIMER; M-HSCLN TIMER; DEC; GO ALL IND & IC; GI CNIDC IND; LA DISABLE IRPT LINE; T-LA HSCLN MODE; (IRPT & PS INT & BR REC) INH LOAD
SEE	XX-XX3	ADV EN SEQ; RESET LFL; A+B SAMPLE REFERENCE	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

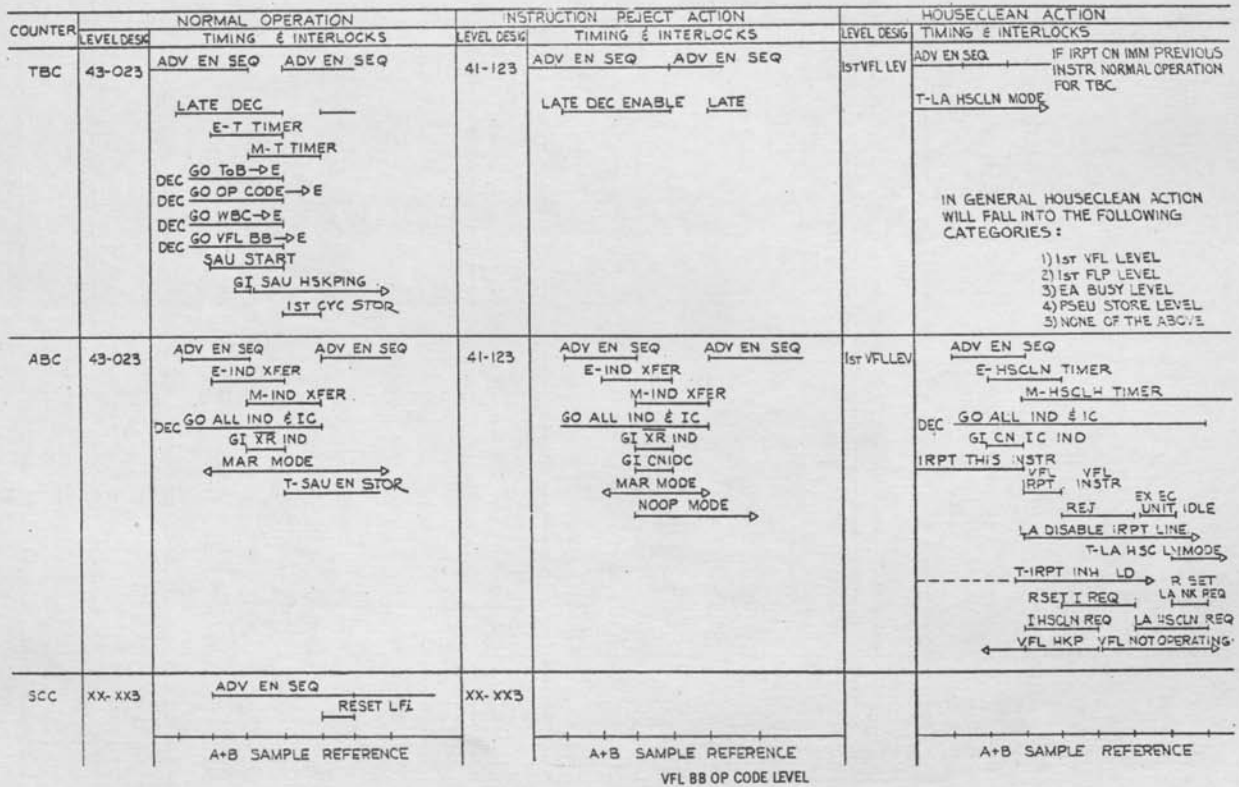
VFL MPYC OPERAND $\overline{IC}$

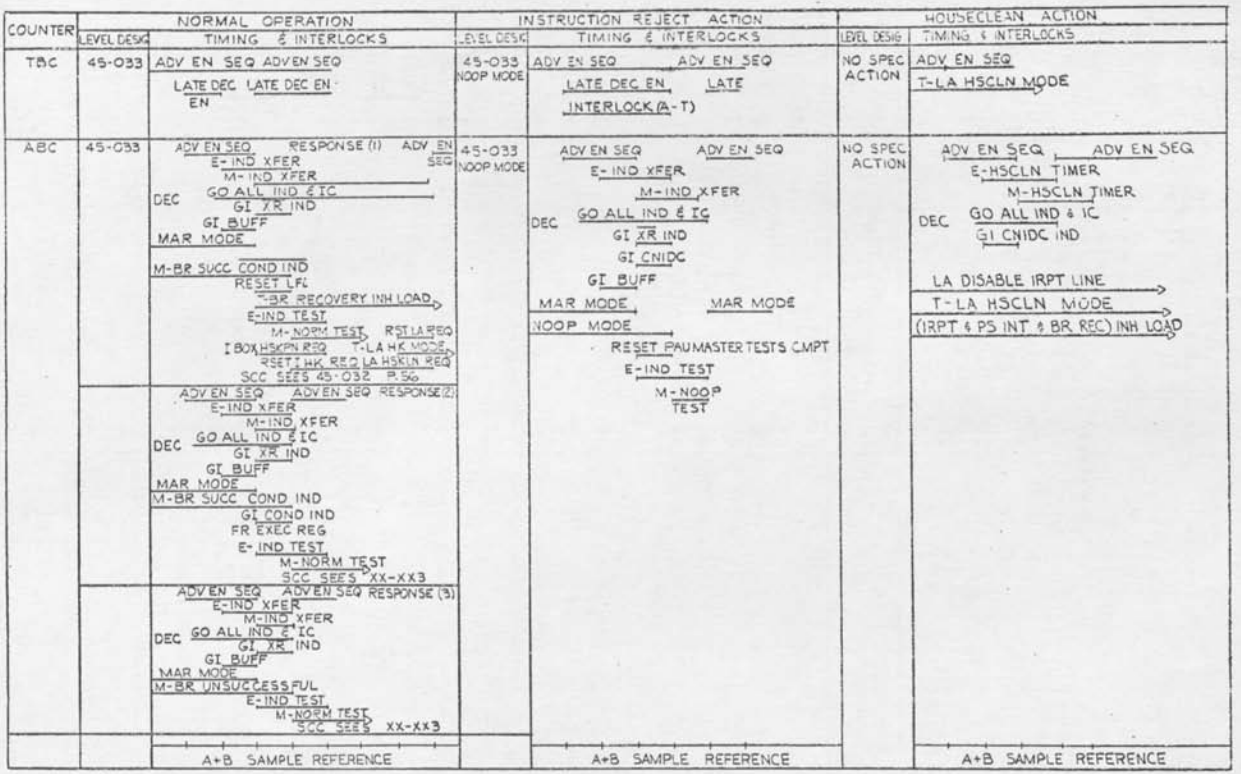
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		INTERRUPT ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	04-033	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC EN	04-033 NOOP MODE	ADV EN SEQ AES LATE DEC EN INTERLOCK A-T	PSEUDO STORE	ADV EN SEQ T-LA HSCLN MODE
ABC	04-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DECODE GO ALL IND & IC GI $\overline{XR}$ IND GI → BUFF GI XR IND LST CYC ST STOR MAR MODE E-IND TEST M-NORM TEST RESET SAU EN STOR	04-033 NOOP MODE	ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC GI BUFF MAR NEXT INSTR MAR MODE MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT SET PS INT INH LOAD RESET LFL E-IND TEST M-NOOP TEST I HSCLN REQ T-LA HSCLN MODE RST I REQ LA IIK REQ RESET SAU EN STOR RST LA REQ	PSEUDO STORE	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET LFL LA DISABLE IRPT LINE T-IRPT INH LOAD T-LA HSCLN MODE
SCC	XX-XX3	ADV EN SEQ RESET LFL A+B SAMPLE REFERENCE	04-032	SEE FIGURE 3.11-1 A+B SAMPLE REFERENCE	04-032	SEE FIGURE 3.11-1 A+B SAMPLE REFERENCE

NORMAL CODE FOR VFL PX LEVEL

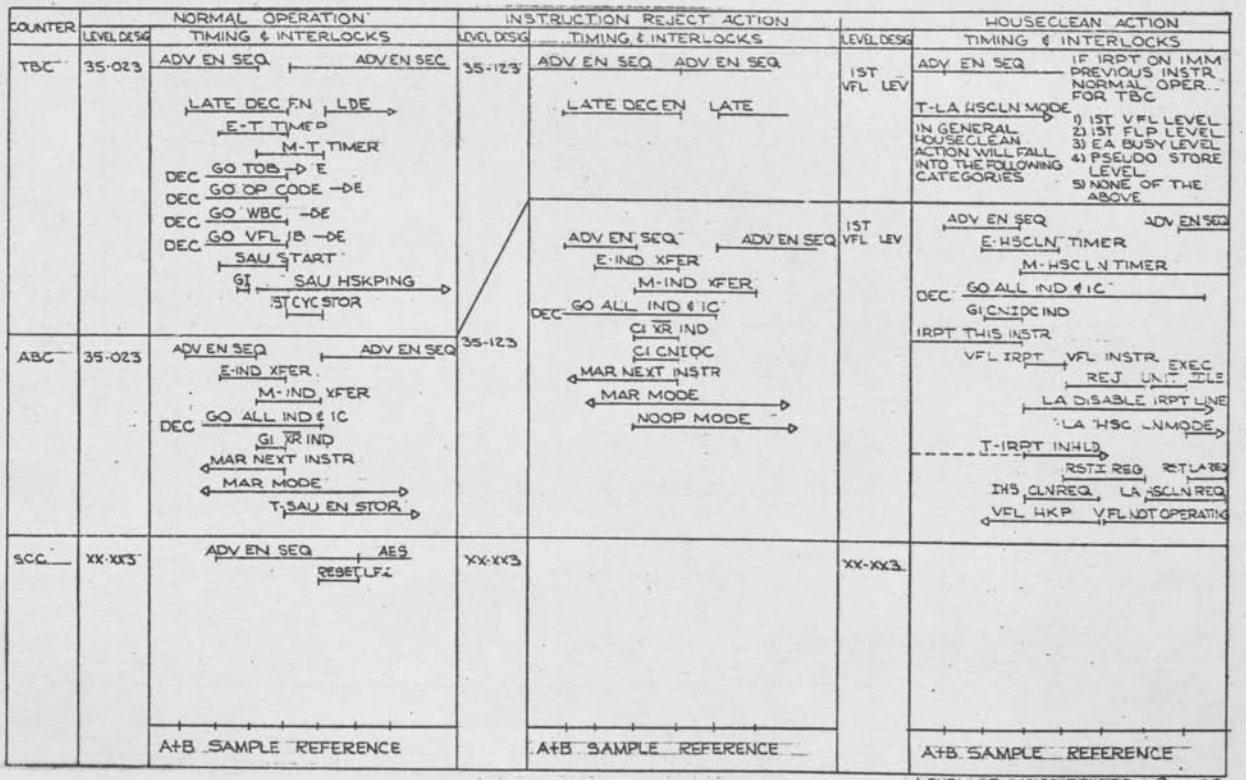
COUNTER	NORMAL OPERATION LEVEL DESIG	NORMAL OPERATION TIMING & INTERLOCKS	INSTRUCTION REJECT ACTION LEVEL DESIG	INSTRUCTION REJECT ACTION TIMING & INTERLOCKS	HOUSECLEAN ACTION LEVEL DESIG	HOUSECLEAN ACTION TIMING & INTERLOCKS
TBC	06-033	ADV EN SEQ LATE DEC ENABLE	04-033 NOOP MODE	ADV EN SEQ LATE DEC EN INTERLOCK A-T	PSEUDO STORE	ADV EN SEQ T-LA HSCLN MODE
ABC	06-033	ADV EN SEQ E-IND XFER M-IND XFER DEL GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF LST CYC STO2 MAR MODE SET PS INT INH LOAD RESET LF1 I BOX HSCLN REQ T-LA-HSCLN RST I REQ LA HSCLN REQ E-IND-TEST RST LA REQ M-NORMAL TEST RESET SAU EN STOR	04-033 NOOP MODE	ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & TC GI XR IND GI CNIDC GI → BUFF, MAR NEXT INSTR. MAR MODE MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT SET PS INT INH LOAD RESET LF1 I BOX HSCLN REQ T-LA-HSCLN MODE RST I REQ LA HKLN REQ E-IND TEST RST LA REQ M-NOOP TEST RESET SAU EN STOR	PSEUDO STORE	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET LF1 LA DISABLE IRPT LINE T-IRPT INH LOAD TA-LA HSCLN MODE
SCC	06-032	SEE FIGURE 3.11-1 A+B SAMPLE REFERENCE	06-032	SEE FIGURE 3.11-1 A+B SAMPLE REFERENCE	06-032	SEE FIGURE 3.11-1 A+B SAMPLE REFERENCE

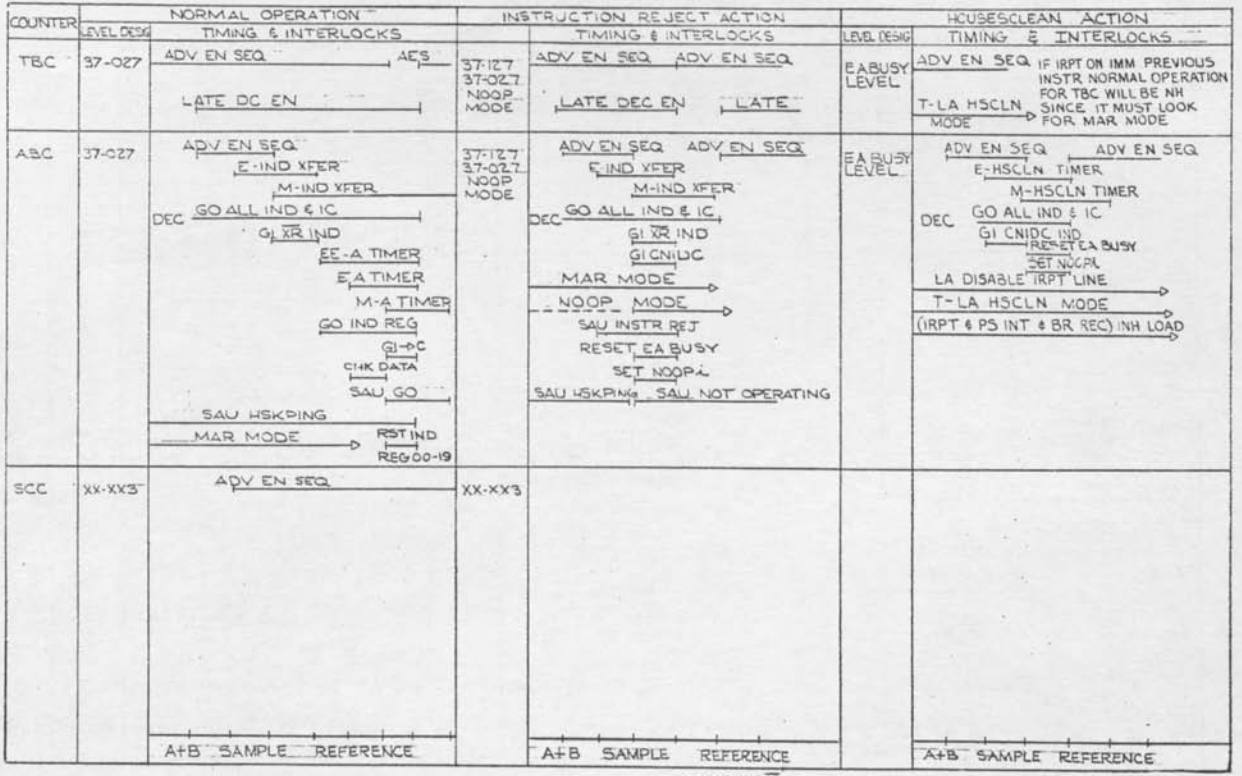
NO-OP CODE FOR VFL PX LEVEL


COUNTER
NORMAL OPERATION
LEVEL DESG
TIMING & INTERLOCKS
INSTRUCTION REJECT ACTION
LEVEL DESG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESG
TIMING & INTERLOCKS
TBC
43-023
ADV EN SEQ
ADV EN SEQ
LATE DEC
E-T TIMER
M-T TIMER
DEC GO ToB→E
DEC GO OP CODE→E
DEC GO WBC→E
DEC GO VFL BB→E
SAU START
GI SAU HSKPING
1ST CYC STOR
41-123
ADV EN SEQ
ADV EN SEQ
LATE DEC ENABLE
LATE
1ST VFL LEV
ADV EN SEQ
IF IRPT ON IMM PREVIOUS INSTR NORMAL OPERATION FOR TBC
T-LA HSCLN MODE
IN GENERAL HOUSECLEAN ACTION WILL FALL INTO THE FOLLOWING CATEGORIES:
1) 1ST VFL LEVEL
2) 1ST FLP LEVEL
3) EA BUSY LEVEL
4) PSEU STORE LEVEL
5) NONE OF THE ABOVE
ABC
43-023
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
GI XR IND
MAR MODE
T-SAU EN STOR
41-123
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
GO ALL IND & IC
GI XR IND
GI CNIDC
MAR MODE
NOOP MODE
1ST VFLLEV
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC GO ALL IND & IC
GI CN IC IND
IRPT THIS INSTR
VFL IRPT
VFL INSTR
EX EC UNIT
REJ
IDLE
LA DISABLE IRPT LINE
T-LA HSC LN MODE
T-IRPT INH LD
RESET
RSET I REQ
LA NK REQ
I HSCLN REQ
LA HSCLN REQ
VFL HKP
VFL NOT OPERATING
SCC
XX-XX3
ADV EN SEQ
RESET LFL
A+B SAMPLE REFERENCE
XX-XX3
A+B SAMPLE REFERENCE
VFL BB OP CODE LEVEL
A+B SAMPLE REFERENCE



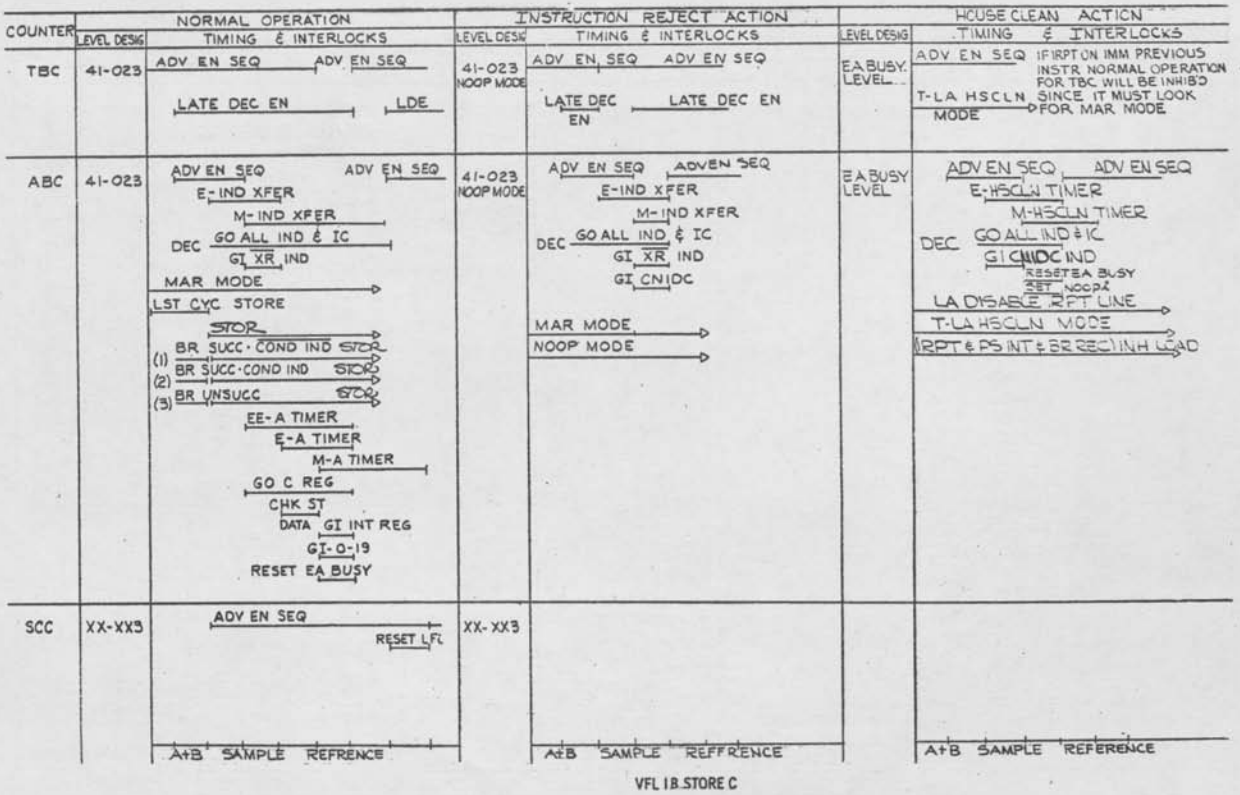
VFL I8 OR B8 RECOVERY LEVEL


COUNTER
NORMAL OPERATION
LEVEL DESG
TIMING & INTERLOCKS
INSTRUCTION REJECT ACTION
LEVEL DESG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESG
TIMING & INTERLOCKS
TBC
35-023
ADV EN SEQ
ADV EN SEC
LATE DEC FN
LDE
E-T TIMER
M-T TIMER
DEC GO TOB → E
DEC GO OP CODE → DE
DEC GO WBC → DE
DEC GO VFL B → DE
SAU START
GI SAU HSKPING
1ST CYC STOR
35-123
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
1ST VFL LEV
ADV EN SEQ
IF IRPT ON IMM PREVIOUS INSTR NORMAL OPER. FOR TBC
T-LA HSCLN MODE
IN GENERAL HOUSECLEAN ACTION WILL FALL INTO THE FOLLOWING CATEGORIES
1) 1ST VFL LEVEL
2) 1ST FLP LEVEL
3) EA BUSY LEVEL
4) PSEUDO STORE LEVEL
5) NONE OF THE ABOVE
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
CI XR IND
CI CNIOC
MAR NEXT INSTR
MAR MODE
NOOP MODE
1ST VFL LEV
ADV EN SEQ
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC GO ALL IND & IC
GI CNIOC IND
IRPT THIS INSTR
ABC
35-023
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
GI XR IND
MAR NEXT INSTR
MAR MODE
T-SAU EN STOR
35-123
VFL IRPT
VFL INSTR EXEC
REJ UNIT TILE
LA DISABLE IRPT LINE
LA HSC LN MODE
T-IRPT INHLD
RSTI REQ
RST LA REQ
IHS CLNREQ
LA HSCLN REQ
VFL HKP
VFL NOT OPERATING
SCC
XX-XX3
ADV EN SEQ
AES
RESET LFL
XX-XX3
XX-XX3
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
VFL OP CODE LEVEL INDICATOR BRANCH
LEVEL AB, 10-27-59 WSC



VFL 1B OPERAND INT IC

911



VFL I B STORE C

(1 UNIT WAITING FOR LA RESPONSE 1) RESUME - STICA OR 2) RESUME - NO STICA)

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	45-023	ADV EN SEQ ADV EN SEQ LATE DEC EN LDE	45-023 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN LDE	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN
ABC	45-023	ADV EN SEQ ADV EN SEQ RESPONSE (1) E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND MAR MODE M-BR SUCC COND IND RESUME-STICA	45-023 NOOP MODE	ADV EN SEQ AES E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC MAR MODE NOOP MODE RESUME-NO STICA		ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN (IRPT & PS INT & BR REC) INH
		ADV EN SEQ ADV EN SEQ RESPONSE (2) E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND MAR MODE M-BR SUCC COND IND RESUME-NO STICA GI COND IND FR EXEC REG				
		ADV EN SEQ ADV EN SEQ RESPONSE (3) E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND MAR MODE M-BR UNSUCCESSFULL RESUME-NO STICA				
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

VFL STICA-IB RECOVERY LEVEL

Page 39

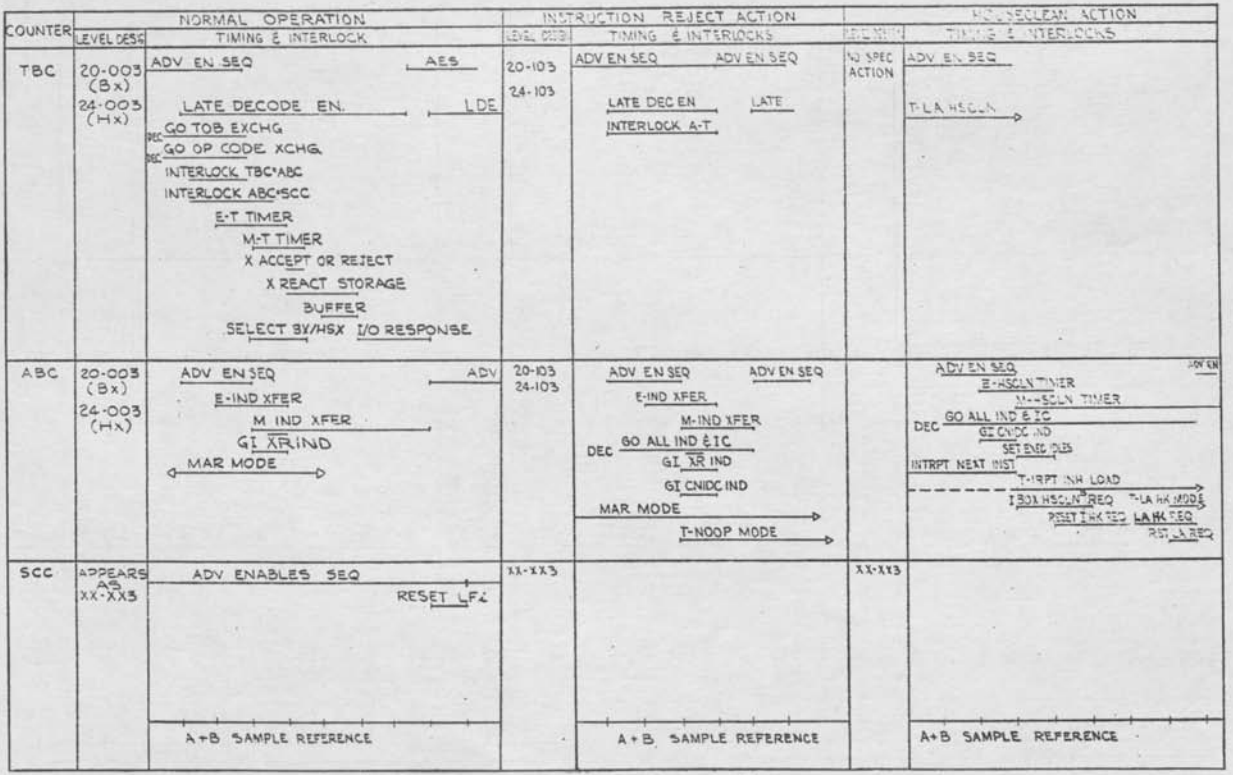
811

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	47-033	ADV EN SEQ ADV EN SEQ LATE DEC LATE DEC EN EN	47-033 NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC ENABLE LATE	NO SPEC ACTION	ADV EN SEQ IF IRPT ON IMM PREVIOUS INSTR, NORMAL OPERATION FOR TBC WILL BE INH'D T-LA HSCLN SINCE IT MUST LOOK FOR MAR MODE MODE
ABC	47-033	ADV EN SEQ RESPONSE (1) E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI BUFF MAR MODE M-BR SUCC. COND IND E-IND TEST M-NORMAL TEST	47-033 NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO IND & IC GI XR IND GI CNIDC GI BUFF MAR MODE NOOP MODE RESET PAU MASTER TESTS CMPT E-IND TEST M-NOOP TEST	NO SPEC ACTION	ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT & PS INT & BR REC) INH LOAD
		ADV EN SEQ ADV EN SEQ RESPONSE(2) E-IND XFER M-IND XFER GO ALL IND & IC DEC GI XR IND GI CNIDC SCC SEES XX-XX3 GI BUFF MAR MODE M-BR SUCC COND IND GI COND IND E-IND TEST M-NORM TEST				
		ADV EN SEQ AES RESPONSE (3) E-IND XFER M-IND XFER DEC GO ALL IND & IC SCC SEES XX-XX3 GI XR IND GI BUFF MAR MODE M-BR UNSUCCESS E-IND TEST M-NORM TEST	XX-XX3			
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

NO-OP CODE FOR 1B OR BB RECOVERY LEVEL.

COUNTER	LEVEL DESG	NORMAL OPERATION TIMING & INTERLOCKS
SCC	45-032 T-LA HSCLM	ADV EN SEQ ADV EN SEQ LATE DEC EN E-LA-I TIMER M-LA-I TIMER FOR NO INDEX XFER GO LAICIB GI-Y RESET LF CND MK FR LA PAR ERR
		A+B SAMPLE REFERENCE

VFL 1B OR BB SCC BR RECOVERY


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCK
INSTRUCTION REJECT ACTION
LEVEL DESIG
TIMING & INTERLOCKS
HOUSECLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
20-003 (Bx)
24-003 (Hx)
ADV EN SEQ
AES
LATE DECODE EN.
LDE
GO TOB EXCHG
GO OP CODE XCHG
INTERLOCK TBC*ABC
INTERLOCK ABC*SCC
E-T TIMER
M-T TIMER
X ACCEPT OR REJECT
X REACT STORAGE
BUFFER
SELECT BX/HSX I/O RESPONSE
20-103
24-103
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
LATE
INTERLOCK A-T
NO SPEC ACTION
ADV EN SEQ
T-LA HSCLN
ABC
20-003 (Bx)
24-003 (Hx)
ADV EN SEQ
ADV
E-IND XFER
M IND XFER
GI XR IND
MAR MODE
20-103
24-103
ADV EN SEQ
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND & IC
GI XR IND
GI CNIDC IND
MAR MODE
T-NOOP MODE
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC GO ALL IND & IC
GI CNIDC IND
SET ENBL OLES
INTRPT NEXT INST
T-IRPT INH LOAD
BX HSCLN REQ
T-LA HK MODE
RESET INH REQ
LA HK REQ
RST LA REQ
SCC
APPEARS AS XX-XX3
ADV ENABLES SEQ
RESET LF4
XX-XX3
XX-XX3
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE

I-O LEVEL IC

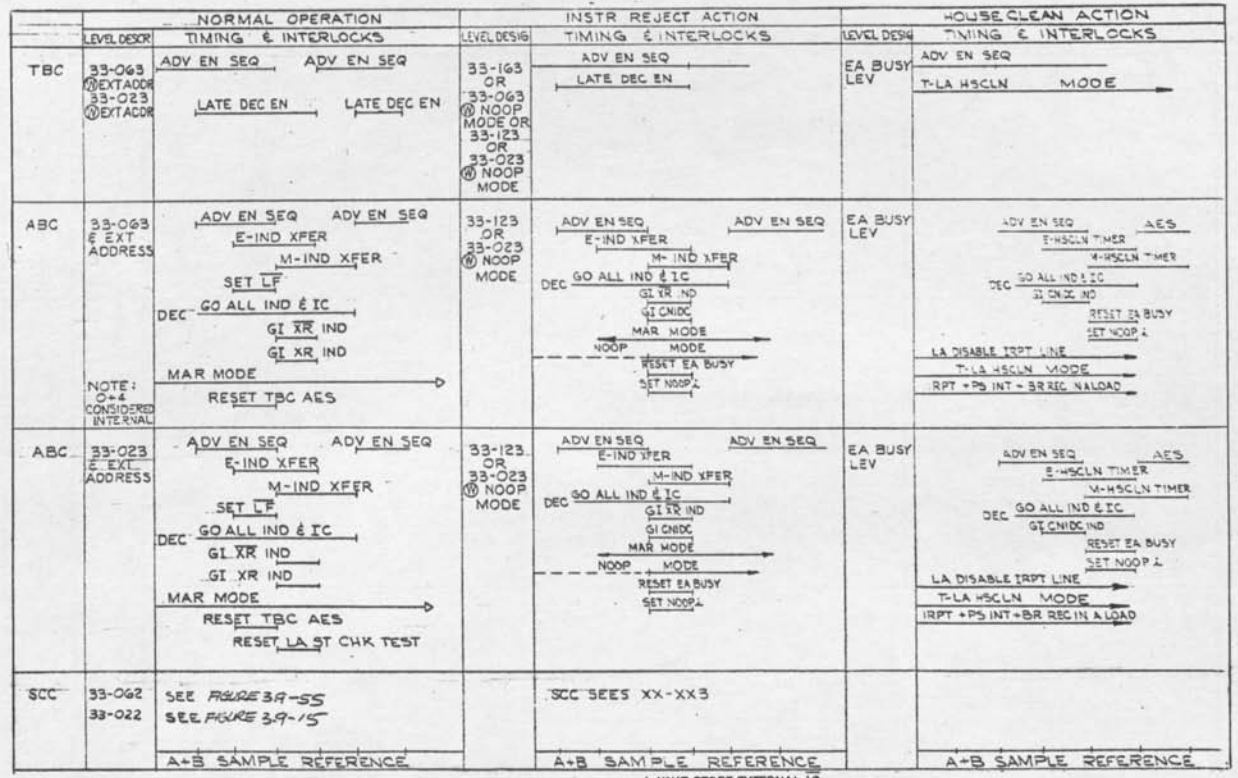
COUNTER	NORMAL OPERATION		INSTR REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	25-033	ADV EN SEQ — ADV LATE DEC EN — LDE	25-033 Ⓝ NOOP MODE OR 25-133	ADV EN SEQ LDE — LDE INTERLOCK A-T	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE →
ABC	25-033	ADV EN SEQ — ADV EN SEQ E-XFER IND M-XFER IND DEC GO ALL IND & IC GI $\overline{XR}$ IND GI BUFF MAR MODE — MAR MODE RESET PAU MASTER TESTS RESET TBC AES E-IND TEST M-NOOP TEST	25-133 OR 25-033 Ⓝ NOOP MODE CMPT	ADV EN SEQ — ADV EN SEQ E-XFER IND M-XFER IND DEC GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC GI BUFF MAR MODE — MAR MODE RESET PAU MASTER TESTS & CMPT NOOP MODE E-IND TEST M-NOOP TEST		ADV EN SEQ — ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE → T-LA HSCLN MODE → (IRPT & PS INT & BR RES) NH LOAD →
SCC	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

I-O 2nd LEV $\overline{CCW}$ IC

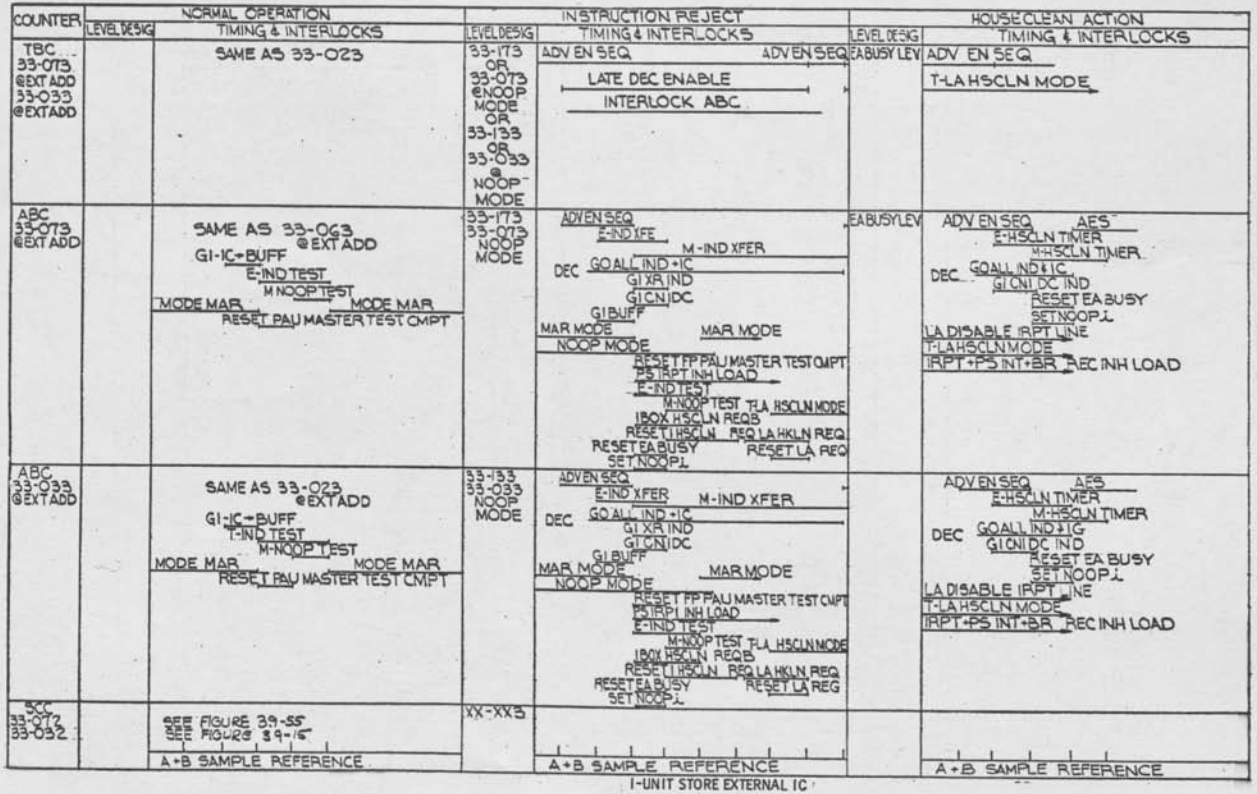
122

COUNTER	NORMAL OPERATION		INSTR. REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	65-033	ADV EN SEQ ADV LATE DEC EN LDE	65-133 OR 65-033 @ NOOP MODE	ADV EN SEQ LDE LDE INTERLOCK A-T	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN MODE
ABC	65-033	ADV EN SEQ ADV EN SEQ E-XFER IND M-XFER IND DEC GO ALL IND & IC GI XR IND GI BUFF MAR MODE MAR MODE RESET PAU MASTER TESTS CMPT RESET TBC AES E-IND TEST M-NOOP TEST	65-133 OR 65-033 @ NOOP MODE	ADV EN SEQ ADV EN SEQ E-XFER IND M-XFER IND DEC GO ALL IND & IC GI XR IND GI CNIDC GI BUFF MAR MODE MAR MODE RESET PAU MASTER TESTS CMPT NOOP MODE E-IND TEST M-NOOP TEST		ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT&PS INT& BR REC) INH LOAD
SCC	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

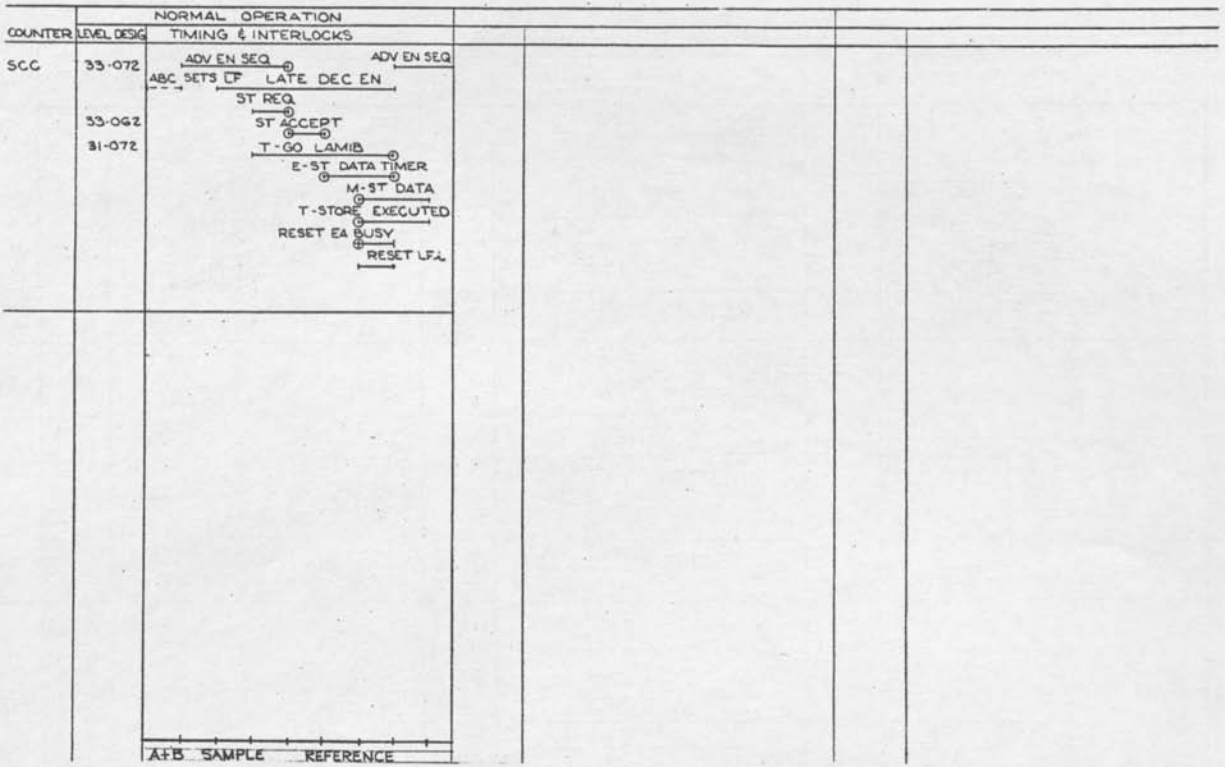
I-O 2nd LEV CCW IC



I-UNIT STORE EXTERNAL IC

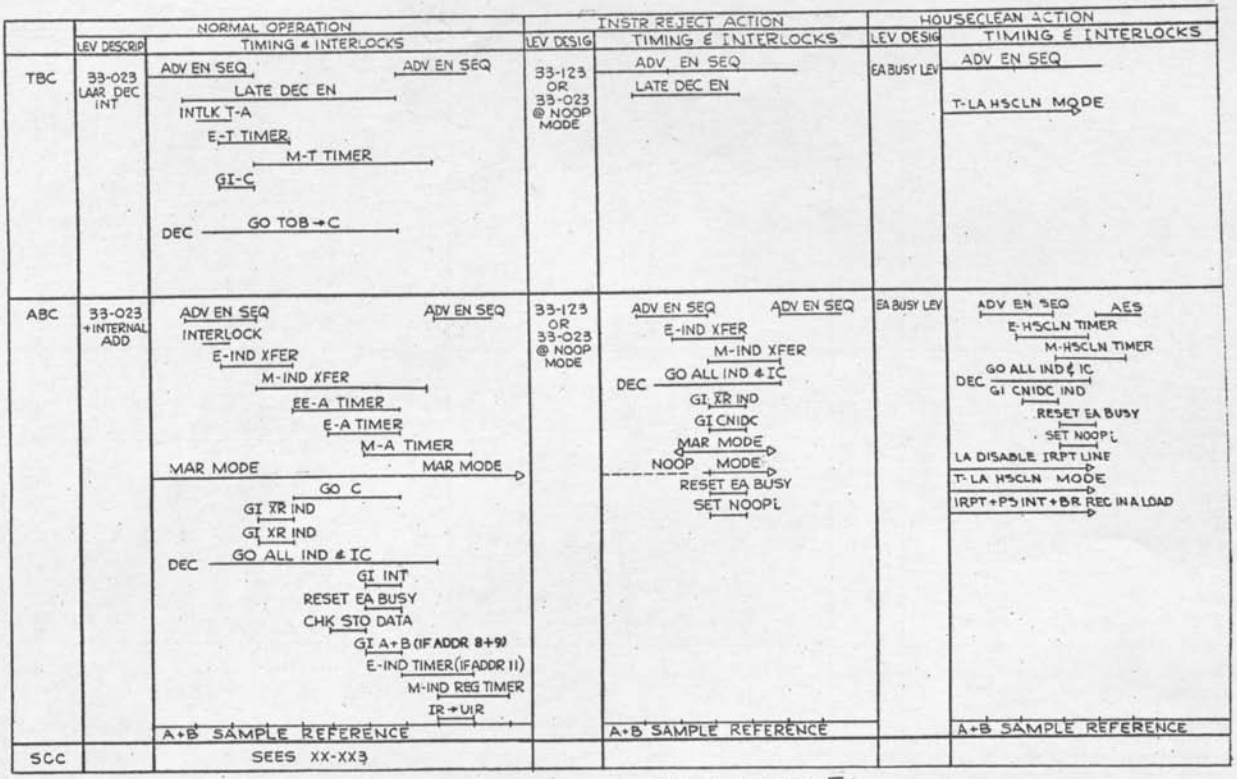


I-UNIT STORE EXTERNAL IC

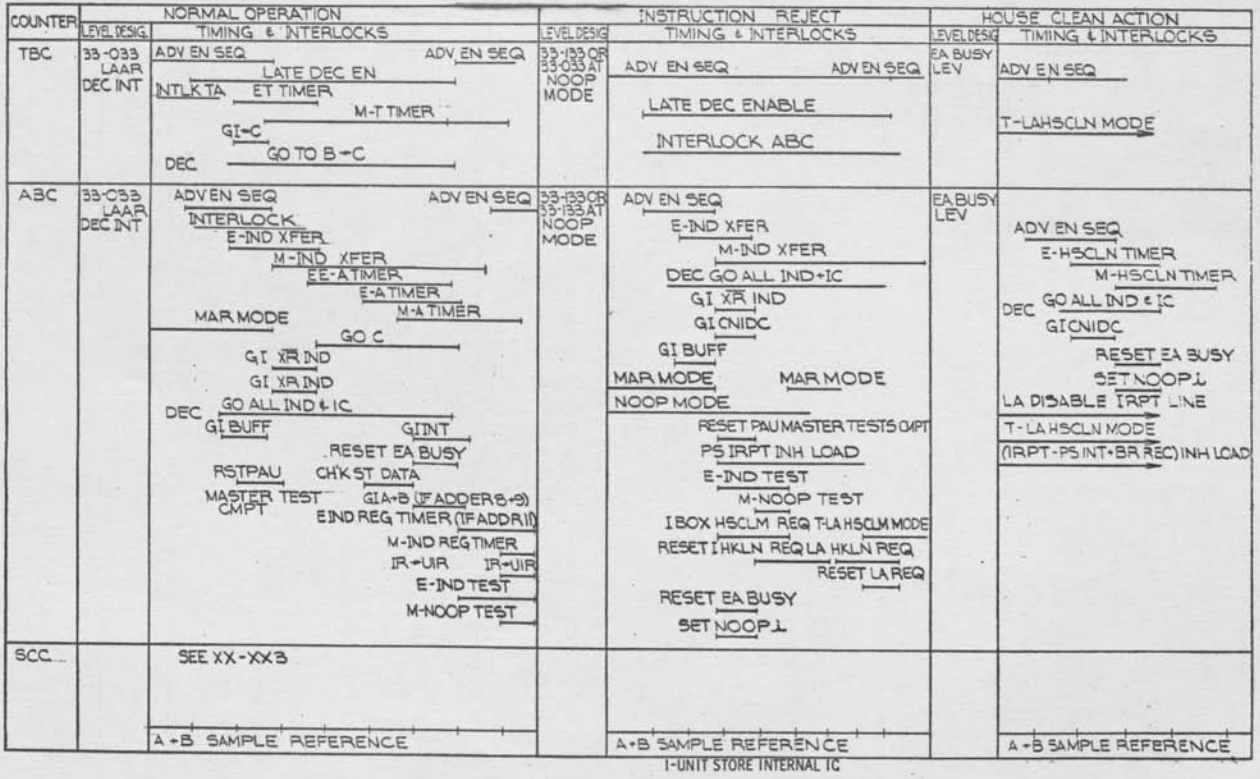

NORMAL OPERATION
COUNTER
LEVEL DESIG
TIMING & INTERLOCKS
SCC
33-072
ADV EN SEQ
ADV EN SEQ
ABC SETS CP
LATE DEC EN
ST REQ
33-062
ST ACCEPT
31-072
T-GO LAMIB
E-ST DATA TIMER
M-ST DATA
T-STORE EXECUTED
RESET EA BUSY
RESET LFL
A+B SAMPLE
REFERENCE
I-UNIT STORE EXT

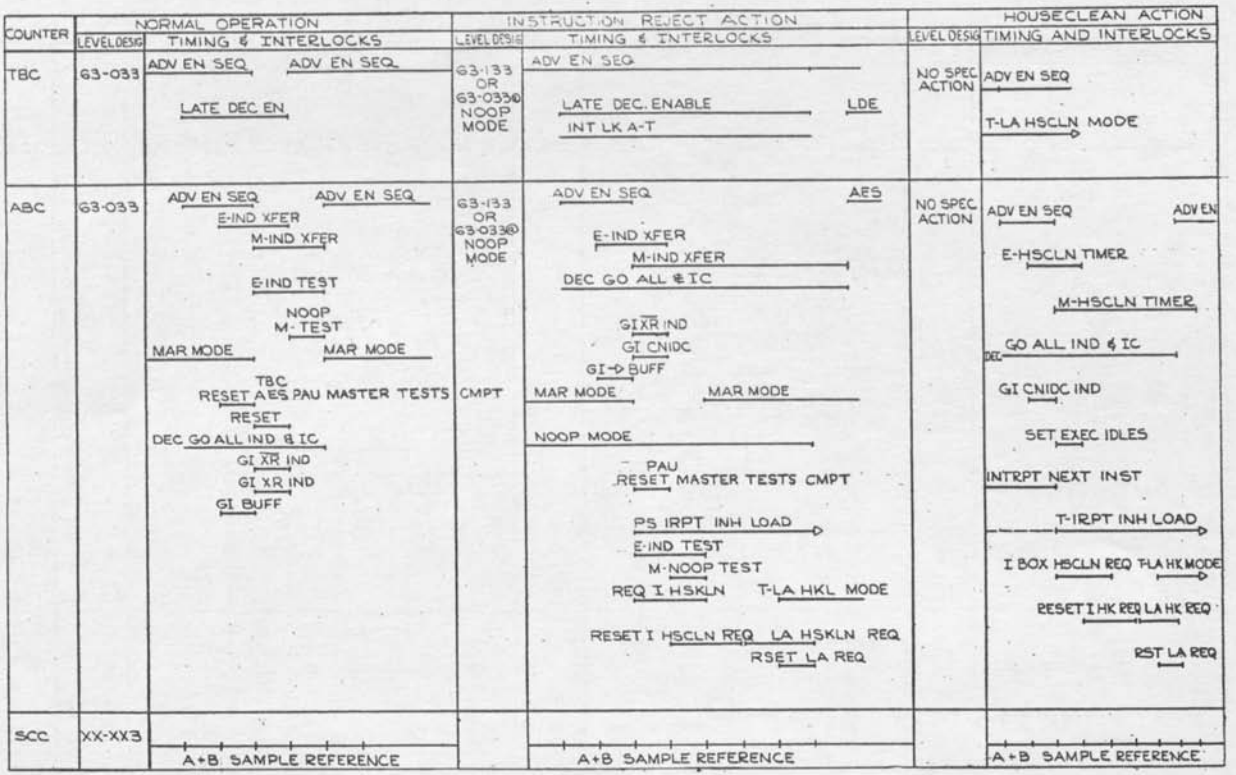
COUNTER	NORMAL OPERATION		INSTRUCTION REJECT		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS				
TBC	31-073	ADV EN SEQ — ADV EN SEQ LATE DEC EN — LATE DEC EN		DOES NOT APPEAR		DOES NOT APPEAR
ABC	31-073	ADV EN SEQ — ADV EN SEQ E-IND XFER M-IND XFER SET LF E-IND TEST M-NORM TEST MAR MODE RESET TBC AES DEC GO ALL IND EIC GI XR IND GI BUFF RESET SAU EN STOR		DOES NOT APPEAR		DOES NOT APPEAR
SCC	31-072	SEE FIGURE 3.9-55 A+B SAMPLE REFERENCE				

STICA-BIN STORE (ASSUMED EXTERNAL) IC



I-UNIT STORE INTERNAL $\overline{IC}$


COUNTER
NORMAL OPERATION
LEVEL DESIG
TIMING & INTERLOCKS
INSTRUCTION REJECT
LEVEL DESIG
TIMING & INTERLOCKS
HOUSE CLEAN ACTION
LEVEL DESIG
TIMING & INTERLOCKS
TBC
33-033 LAAR DEC INT
ADV EN SEQ
ADV EN SEQ
LATE DEC EN
INTLK TA
ET TIMER
M-T TIMER
GI→C
GO TO B→C
DEC
33-133 OR 33-033 AT NOOP MODE
ADV EN SEQ
ADV EN SEQ
LATE DEC ENABLE
INTERLOCK ABC
EA BUSY LEV
ADV EN SEQ
T-LAHSCLN MODE
ABC
33-033 LAAR DEC INT
ADV EN SEQ
ADV EN SEQ
INTERLOCK
E-IND XFER
M-IND XFER
EE-A TIMER
E-A TIMER
M-A TIMER
MAR MODE
GO C
GI XR IND
GI XR IND
DEC
GO ALL IND & IC
GI BUFF
GIINT
RESET EA BUSY
RSTPAU
CHK ST DATA
MASTER TEST CMPT
GIA→B (IF ADDER 8→9)
E IND REG TIMER (IF ADDR I)
M-IND REG TIMER
IR→UIR
IR→UIR
E-IND TEST
M-NOOP TEST
33-133 OR 33-133 AT NOOP MODE
ADV EN SEQ
E-IND XFER
M-IND XFER
DEC GO ALL IND→IC
GI XR IND
GI CNIDC
GI BUFF
MAR MODE
MAR MODE
NOOP MODE
RESET PAU MASTER TESTS CMPT
PS IRPT INH LOAD
E-IND TEST
M-NOOP TEST
I BOX HSCLM REQ T-LA HSCLM MODE
RESET I HKLN REQ LA HKLN REQ
RESET LA REQ
RESET EA BUSY
SET NOOP ⊥
EA BUSY LEV
ADV EN SEQ
E-HSCLN TIMER
M-HSCLN TIMER
DEC
GO ALL IND & IC
GI CNIDC
RESET EA BUSY
SET NOOP ⊥
LA DISABLE IRPT LINE
T-LA HSCLN MODE
(IRPT-PS INT+BR REC) INH LOAD
SCC
SEE XX-XX3
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
A+B SAMPLE REFERENCE
I-UNIT STORE INTERNAL TC



I-UNIT INDICATOR TRANSFER IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	02-023	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC		DOES NOT APPEAR	PS EUDO STORE	ADV EN SEQ T-LA HSCLN IF IRPT ON IMM PREVIOUS INSTR, TBC WAITS FOR ABC TO RESET TC WHICH NEVER COMES
ABC	02-023	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI XR IND RESET TBC AES		DOES NOT APPEAR		ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET LFL SET EXEC IDLES
SCC	XX-XX3	ADV EN SEQ RESET LFL				T-IRPT INH LOAD I BOX HSKLN REQ B T-LA HK MODE RST I HK REQ LA HK REQ RST LA REQ
		A+B SAMPLE REFERENCE				A+B SAMPLE REFERENCE

1-UNIT PSEUDO STORE IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT OPERATION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	02-033	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE DEC WAIT FOR ABC TO RESET FLP TC	02-133 02-133 NOOP MODE	ADV EN SEQ LATE DEC ENABLE INTERLOCK A-T	PSEUDO STORE	ADV EN SEQ T-LA HSCLN IF IRPT ON IMM PREVIOUS INSTR, TBC WAITS FOR ABC TO RESET TC WHICH NEVER COMES
ABC	02-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI XR IND GI → BUFF RESET PAU MASTER TESTS E-IND TEST M-NOOP TEST MAR MODE MAR MODE RESET TBC AES	02-133 02-033 NOOP MODE CMPT	ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI XR IND GI CNIDC GI → BUFF MAR MODE NOOP MODE RESET PAU MASTER TEST CMPT P5 IRPT INH LOAD RESET LFL T-BOX HSKLN REQ B LA HK MODE RESET I HK REQ LA HK REQ E-IND TEST RST LA REQ M NOOP TEST SCC SEE FIGURE 3.11-1		ADV EN SEQ ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET LFL SET EXEC IDLES T-IRPT INH LOAD T BOX HSKLN REQ B TLA HK MODE RST I HK REQ LA HK REQ RST LA REQ
SCC	XX-XX3	ADV EN SEQ RESET LFL				
		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE

I-UNIT PSEUDO STORE IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSEHOLD ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	51-027	ADV EN SEQ — ADV EN SEQ LATE DEC EN		DOES NOT APPEAR	EA BUSY LEV	ADV EN SEQ T-LA HSCLN MODE
ABC	51-027	ADV EN SEQ E-IND XFER M-IND DEC GO ALL IND & IC GI XR IND EE-A TIMER E-A TIMER M-A TIMER GO INT REG MAR MODE GI → LA RST EA BUSY CHK DATA OR CHK DATA RESET LFL		DOES NOT APPEAR	EA BUSY LEVEL	ADV EN SEQ AES E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND RESET EA BUSY SET NOOP i LA DISABLE IRPT LINE T-LA HSCLN MODE (IRPT→PS INT→BR REG) INH LOAD
SCC	51-026	ADV EN SEQ — ADV EN SEQ ABC RESETS LFL LATE DEC EN E-LA I TIMER M-LA-I FOR NO INDEX XFER GO LAICIB→Y GI → Y RESET LFL GI LA PAR ERR TO I BOX COND MK IND FR LA PAR ERR A+B SAMPLE REFERENCE		DOES NOT APPEAR		A+B SAMPLE REFERENCE

I-UNIT INT FETCH $\overline{IC}$

COUNTER	NORMAL OPERATION		INSTR REJ ACTION		RECOVERY ACTION	
	LEV DESIG		LEV DESIG			
TBC	75-023	SAME AS 01-003, PLUS DEC GO IRPT BIT TO EXEC REG	75-123	WILL NOT APPEAR		WILL NOT APPEAR
ABC	75-023	SAME AS 01-003 PLUS INHIBIT RESET T-LA DISABLE IRPT LINE	75-123	WILL NOT APPEAR		WILL NOT APPEAR
SEE	XX-XX3					

IRPT BIN OP CODE LEVEL FOR IRPT BIT RESET

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIGN	TIMING & INTERLOCKS	LEVEL DESIGN	TIMING & INTERLOCKS
TBC	00-023-1	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE TBC WAIT FOR E-M OF IND XFER		WILL NOT APPEAR	NO SPEC ACTION	ADV EN SEQ T-LA HSCLN
ABC	00-023-1	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC MAR MODE GI $\overline{XR}$ IND RST TBC/AES		WILL NOT APPEAR		ADV EN SEQ ADV EN E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC GI CNIDC IND SET EXEC IDLES INTRPT NEXT INST T-IRPT INH LOAD I BOX HSCLN^{B} REQ T-LA HK MODE RESET I HK REQ LA HK REQ RST LA REQ
SCC	XX-XX3	ADV EN SEQ RESET LFL A+B SAMPLE REFERENCE			XX-XX3	A+B SAMPLE REFERENCE

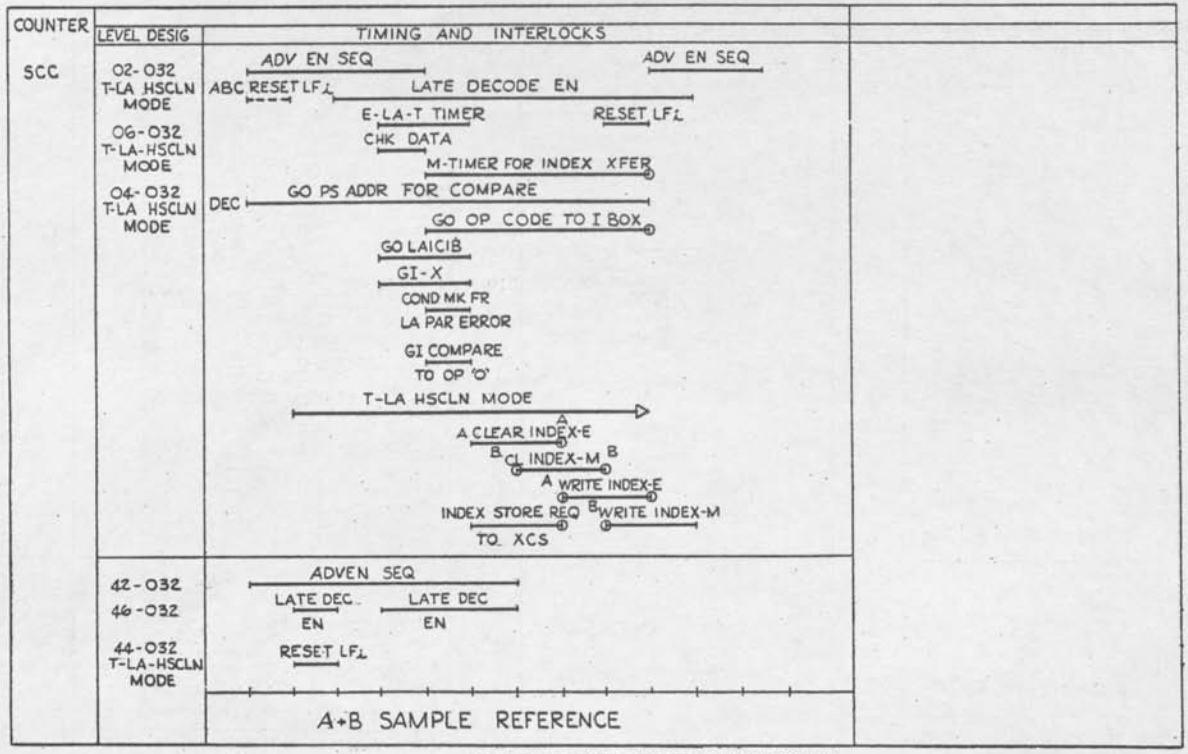
INTERRUPT TEST LEVEL $\overline{IC}$

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS
TBC	00-033-1	ADV EN SEQ, ADV EN SEQ LATE DEC ENABLE		WILL NOT APPEAR	NO SPEC ACTION	ADV EN SEQ T-LA HK MODE
ABC		ADV EN SEQ, ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC MAR MODE, MAR MODE GI XR IND E-IND TEST GI BUFF, M-NOOP TEST RESET DAU MASTER TESTS CMPT RST TBC AES		WILL NOT APPEAR		ADV EN SEQ, ADV EN SEQ E-HSCLN TIMER M-HSCLN TIMER DEC GO ALL IND & IC SET EX IDLES GI CNIDC INTRPT NEXT INST T-IRPT IN H LOAD I BOX HK REQ B, T-LA HK MODE RST I HK REQ, LA HK REQ RST LA HK REQ
SCC	XX-XX3	A+B SAMPLE REFERENCE			XX-XX3	A+B SAMPLE REFERENCE

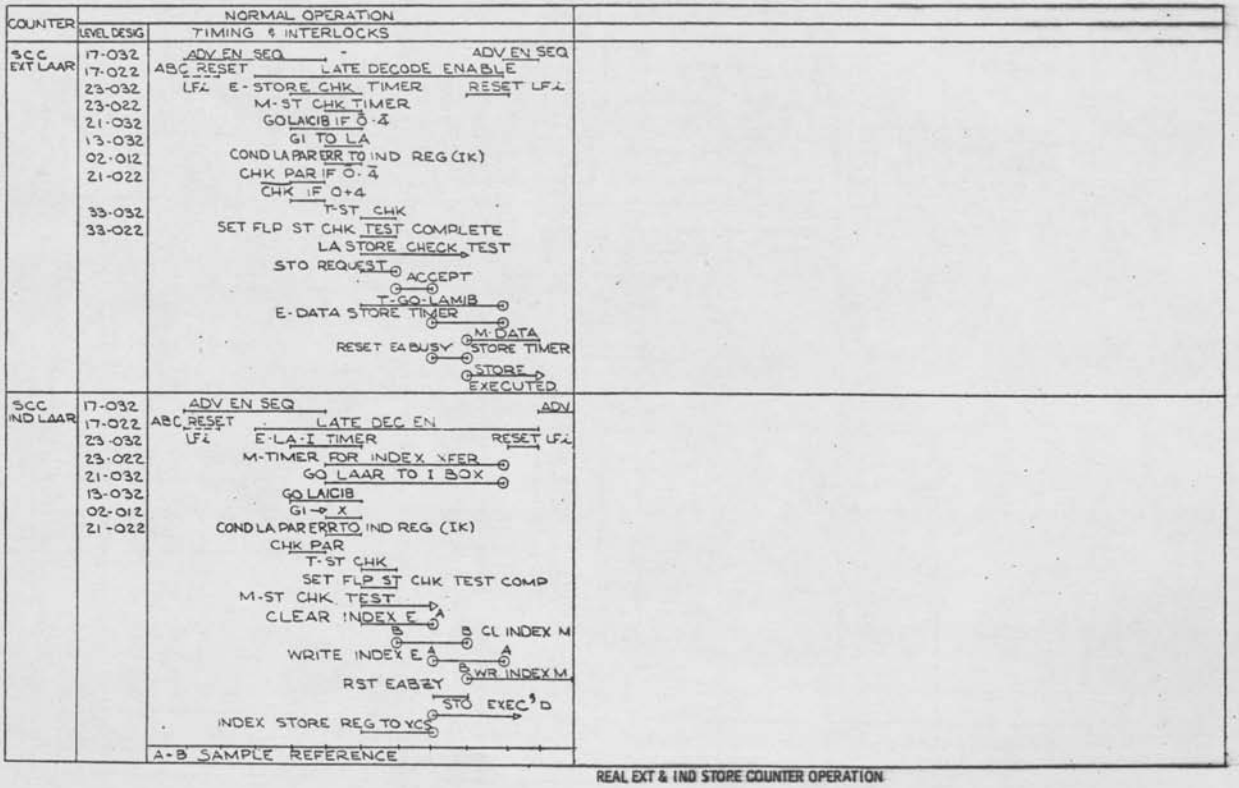
INTERRUPT TEST LEVEL IC

COUNTER	NORMAL OPERATION		INSTRUCTION REJECT ACTION		HOUSECLEAN ACTION	
	LEVEL DESIG	TIMING & INTERLOCKS	LEVEL DESIG	TIMING & INTERLOCKS		
TBC	57-033	ADV EN SEQ ADV EN SEQ LATE DEC EN LATE	57-033 Ⓦ NOOP MODE	ADV EN SEQ ADV EN SEQ LATE DEC EN LDE		DOES NOT APPEAR
ABC	57-033	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER E-IND TEST NORMAL M TEST MAR MODE RESET TBC AB5 DEC GO ALL IND & IC GI $\overline{XR}$ IND GI BUFF RESET SAU EN STOR	57-033 Ⓦ NOOP MODE	ADV EN SEQ ADV EN SEQ E-IND XFER M-IND XFER DEC GO ALL IND & IC GI $\overline{XR}$ IND GI CNIDC GI BUFF E-IND TEST M-NOOP TEST MAR MODE NOOP MODE RESET SAU EN STOR RST PAU MASTER TESTS CMPT		DOES NOT APPEAR
SCC	XX-XX3	A+B SAMPLE REFERENCE		A+B SAMPLE REFERENCE		

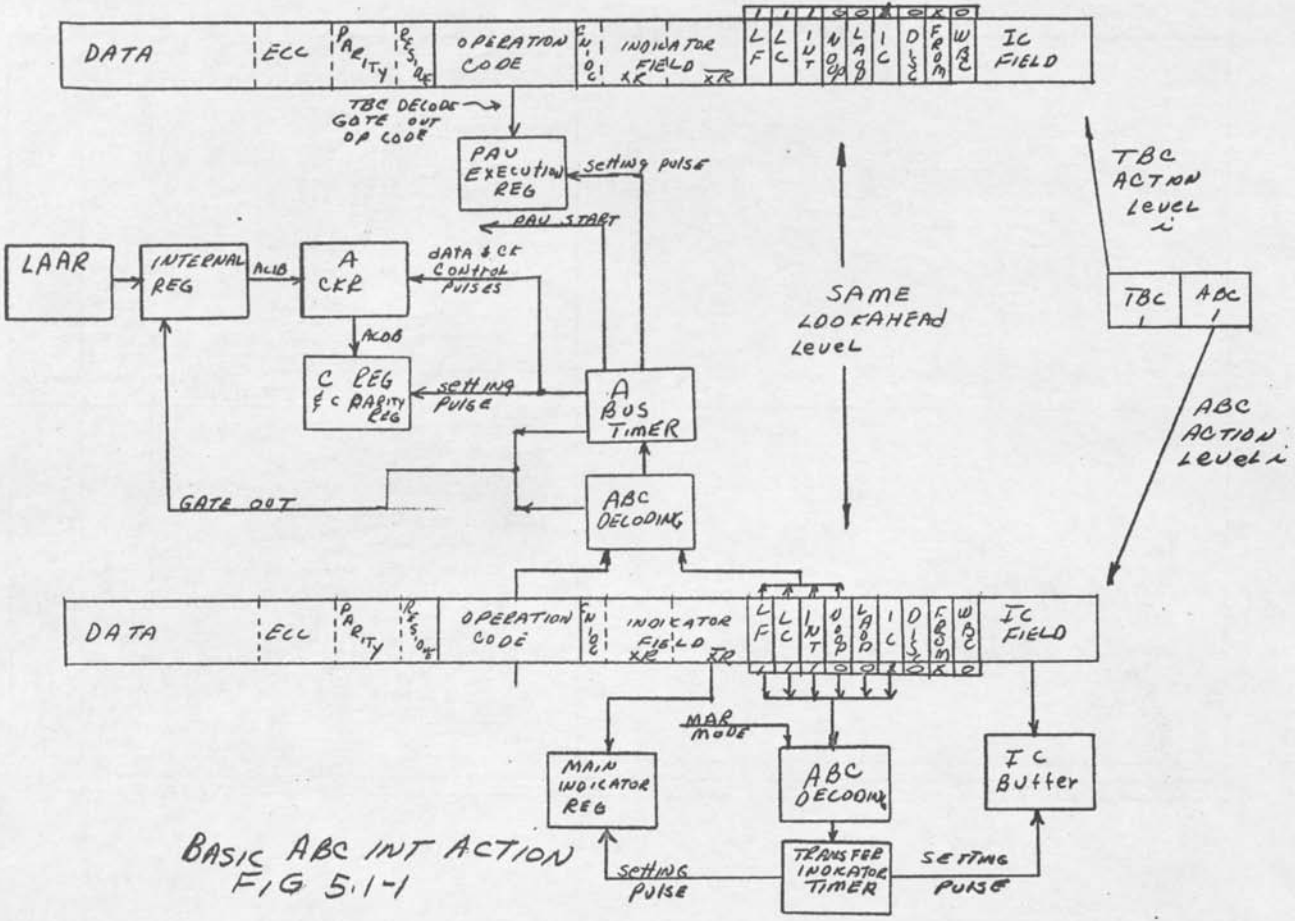
INDICATOR TRANSFER IC FOR STICA BIN



SCC PSEUDO-STORE RETURN DURING HOUSE CLEANING



REAL EXT & IND STORE COUNTER OPERATION



BASIC ABC INT ACTION
FIG 5.1-1

	LINE ABC DECODED	OP CODE AND TAGS																		AND'ED WITH	OTHER COND OR'ED
		0	5	6	7	8	9	LAOP	No Op	No Op Mode	INT	LC	LF		DISC	WBC	LAAR Int	TIC		
1	1st Lev VFL No Op							1	0	0	0										3, 6
2	2nd Lev I/O		1	0	1	0			1	0	0										
3	BB Op Cd No Op	1	0	0	0	1		1	1	0	0										
4	BIN Fetch Lev	0	1	1	1	1		1	1	0	0										
5	BIN Intrpt Lev	1	1	1	1	0		1	1												
6	BIN Op Cd No Op		1	1	1	0		1	1	0	0										
7	BIN Store Level	1	0	0	0	0		1	1	0	0										
8	Cond ABC Dec Br Recovery																	1			
9	Cond ABC Dec No Op Cd Rec Lev						1														
10	Cond ABC Set and No Op	1	0	0	1																
11	Cond ABC Set Cond Ind							1	1		0									10 and Branch Successful Condition Ind	
12	Cond ABC Dec TBC Mult Adv	0	0	1	1	1	1	1			0										7, 32, Int ABC
	Cond ABC Dec TBC Mult Adv	0	1	0	0	1	1	1			0										7, 32, Int ABC
13	Cond SALU MAR	0	0	0	1	0	1	1													4,
	Cond SALU MAR	0	0	0	1	1	1	1													
14	Data Ret On House-lean						1	0	1												
	Data Ret On Housecleon					1		0	1												
15	Del ABC Adv									0	0	1									7, 32, 48, 49, 36, 30, 50
	Del ABC Adv	0	0	1	0	1	1	1	0	0											
	Del ABC Adv		0	1	1	1	0	0	0	0											
16	Del Ind Test	0	0	1	0	1	1	1	0	0											50, 7, 32, 48, 49
	Del Ind Test		0	1	1	1	0	0	0	0											
17	FLP 1st Level					1	0	0													
18	FLP Int					1	0	0	0	0	1										
19	FLP Store	0	0	1	0	1	1	1	0	0											
	FLP Store		0	1	1	1	0	0	0	0											
20	GI D	0	0	0	1	1	1	1												Note: VFL 2nd Operand	
21	I Box Ext Store	0	1	1	0	1	1	1	1	0	0									No LAAR Dec Int Add and No LAAR Dec Index Add	
22	I Box Int Fetch	1	0	1	0	0	1	1	1	0	0										
23	I Box ECC Ext St	0	1	1	0	1	1	1	1	0	0					0				No LAAR Dec Int Add and No LAAR Dec Index Add	
24	I Box Pseudo St		0	0	0	1	0	1	0	0	0										
25	I Box Store No Op	0	1	1	0	1	1	1	1	1	1										
26	I Ex Ind Transfer Only	1	1	0	0	1	1	1	1	0	0										
27	Ind Transfer STICA BIN	1	0	1	1	1	1	1	1	0	0										
28	Inhibit GI C On Int ABC	0	0	0	1	1	1	1	1												
29	Inhibit Set Idles							1	0	0	0										3, 6
	Inhibit Set Idles						1	0	0	0	0										
30	Initiate Branch Rec	1	0	0	1	0	1	1	1	0	0									Branch Successful Not Condition Ind and 8	
31	Initiate Pseudo Int		0	0	1			0	1		1										25, 39, 38
	Initiate Pseudo Int		0	0	1	1	0	1			0										
32	Int I Box Store	0	1	1	0	1	1	1	1	0	0									LAAR Dec Int Add	
33	Interrupt Test Level		0	0	0	0	0	1	0	0											
34	LA Only MAR Next Inst	0	1	1	0	1	1	1	1	0	0										33, 24, 26, 2, 22
35	Level Loaded											1	1	0						ABC I	
36	No Op Code PX NO No Op		0	0	1	1	0	1		0											
37	No Op Coded REC Level						1	1		0								0		9 and 10 Branch Successful Not Cond Ind	
38	No Op I Box Pseudo Store		0	0	0	1	0	0	1	1											
39	No Op I Execute Ind Transfer	1	1	0	0	1	1	1	1	1											
40	Normal PX No No Op Mode		0	0	1	0	0	1		0											24, 26, 32, 21, 38
41	Reset TBC AES	0	1	1	0	1	1	1	0	0										No LAAR Dec Int Add and No LAAR Dec Index Add	24, 27, 26, 45, 33, 2,
42	Resume Branch STICA EM	1	0	0	1		1	1		0								0		Branch Successful Not Condition Ind	
43	Resume No Branch STICA EM	1	0	0	1		1	1										0		No Branch Successful Not Condition Ind	
44	Set a Return Ind XE		0	0	1		0	1	1												30, 45, 21,
	Set a Return Ind XE		0	0	1	1	0	1		0											
45	STICA BIN Store	0	1	1	0	0	1	1													
46	Store A Store B	0	0	1	0	1	1	1	0	0											50, 7, 48, 49
	Store A Store B		0	1	1	1	0	0	0	0											
47	Store Arith Bus	0	0	1	0	1	1	1	0	0											50,
	Store Arith Bus		0	1	1	1	0	0	0	0											
48	Store C	0	1	1	1	1	1			0											
49	Store D	0	1	0	0	1	1	1		0											
50	VFL Store Bus	0	0	1	1	0	1	1		0											

FIGURE 5.2-1. ABC DECODING

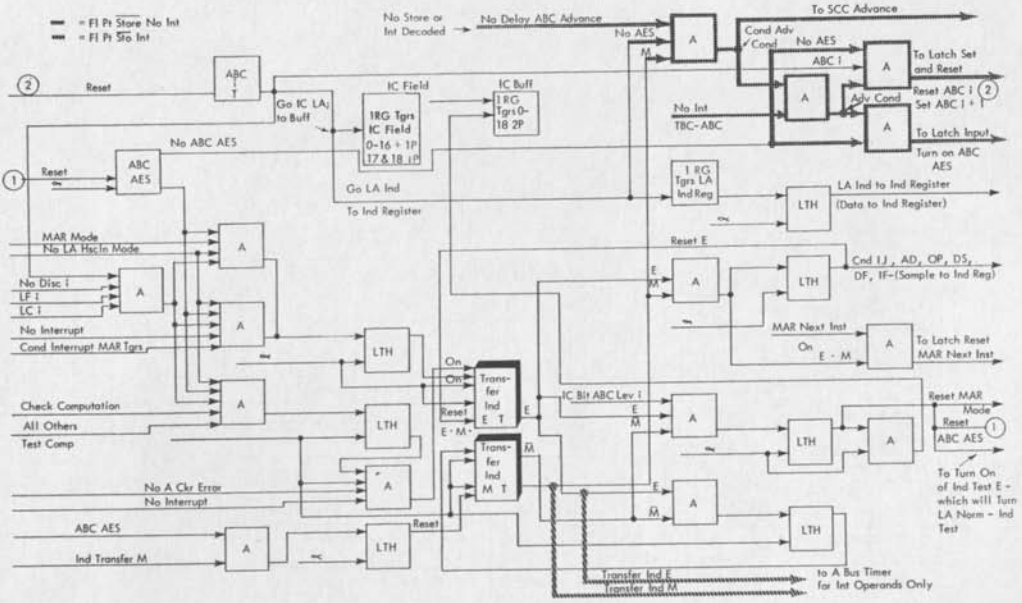


FIGURE 5.2-2. ABC ACTION FLP (NO STORE - NO INTERNAL

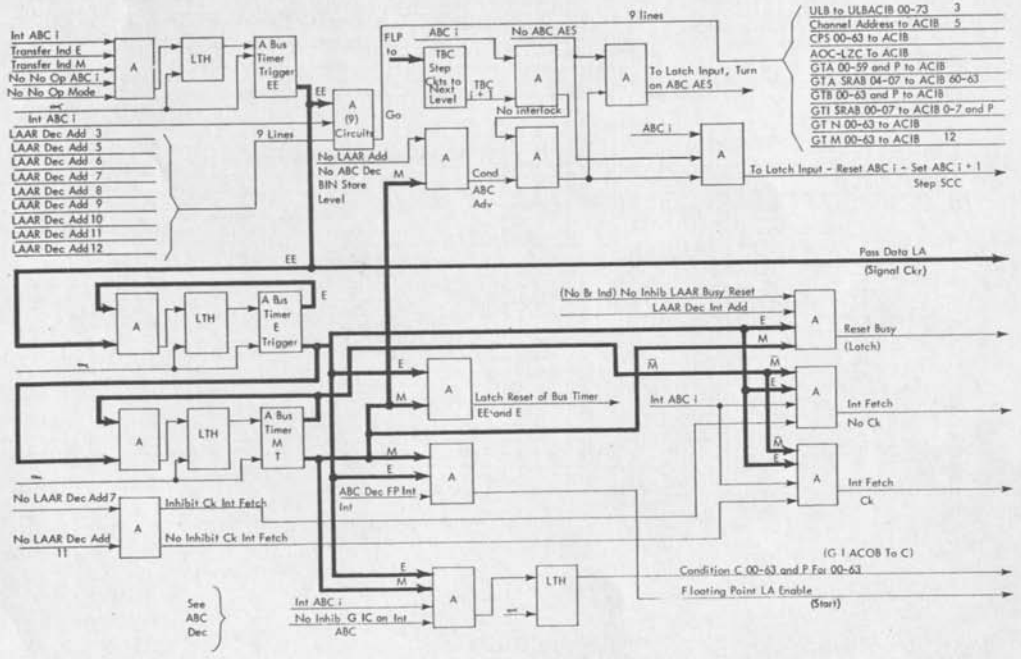


FIGURE 5.3-1. INTERNAL OPERAND FETCH AND COUNTER CONTROL

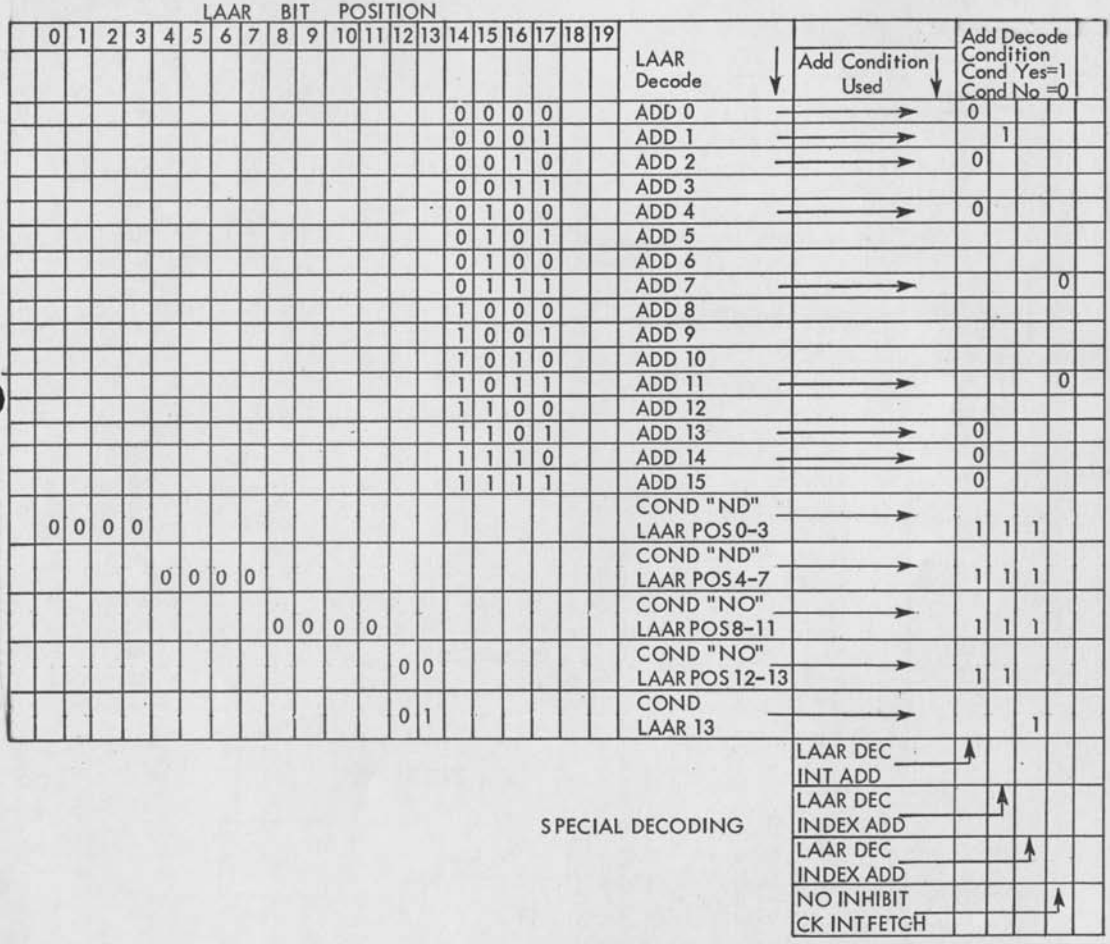
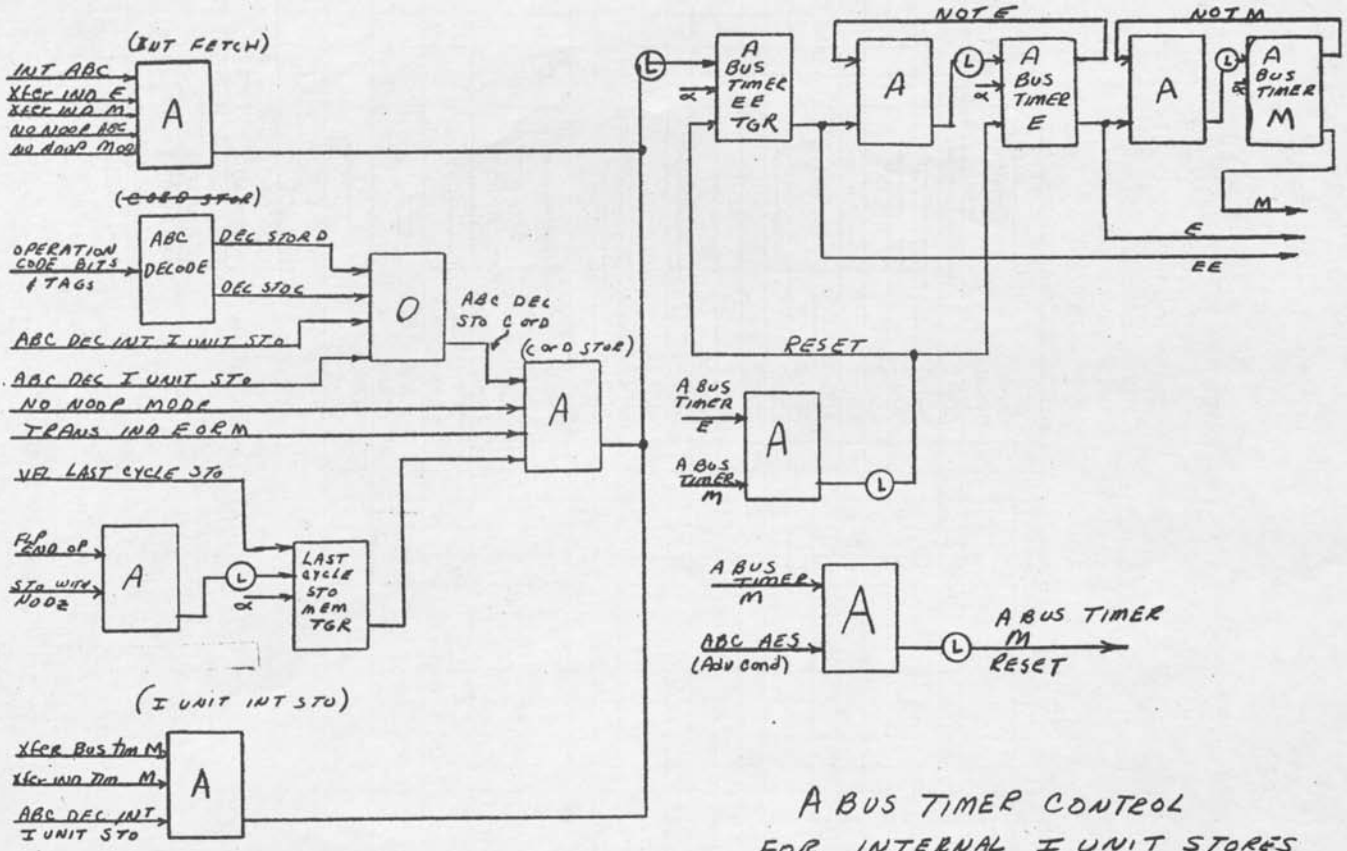


FIGURE 5.3-2. LAAR DECODING



A BUS TIMER FIG 5.4-1

A BUS TIMER CONTROL FOR INTERNAL I UNIT STORES AND STORES FROM THE C OR D REG (NORM VFL STOR LEVELS)

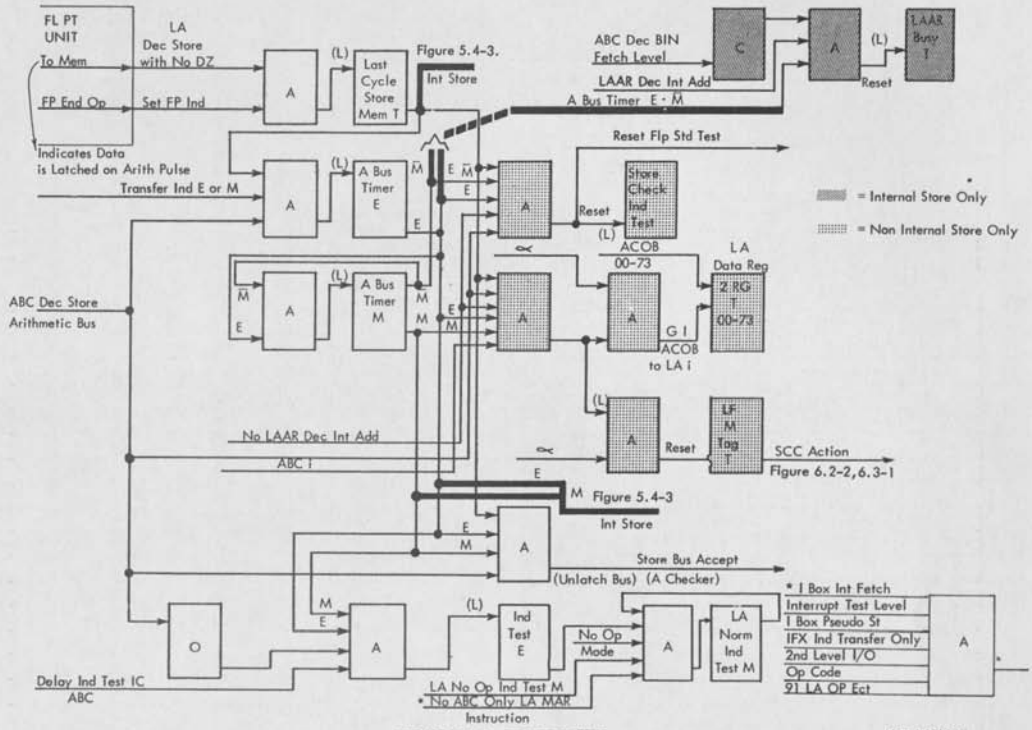


FIGURE 5.4-2. FLP STORE INT

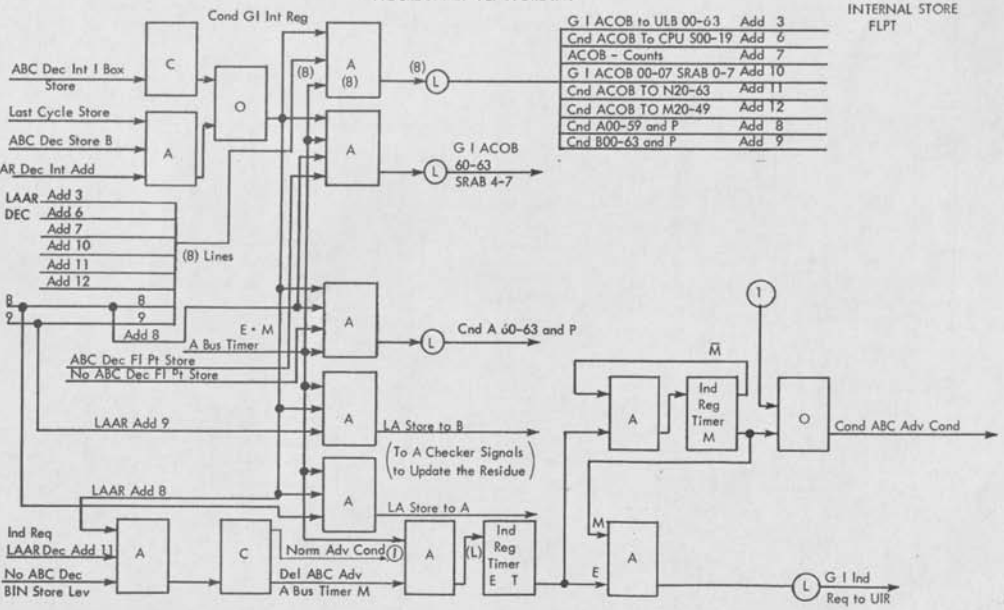
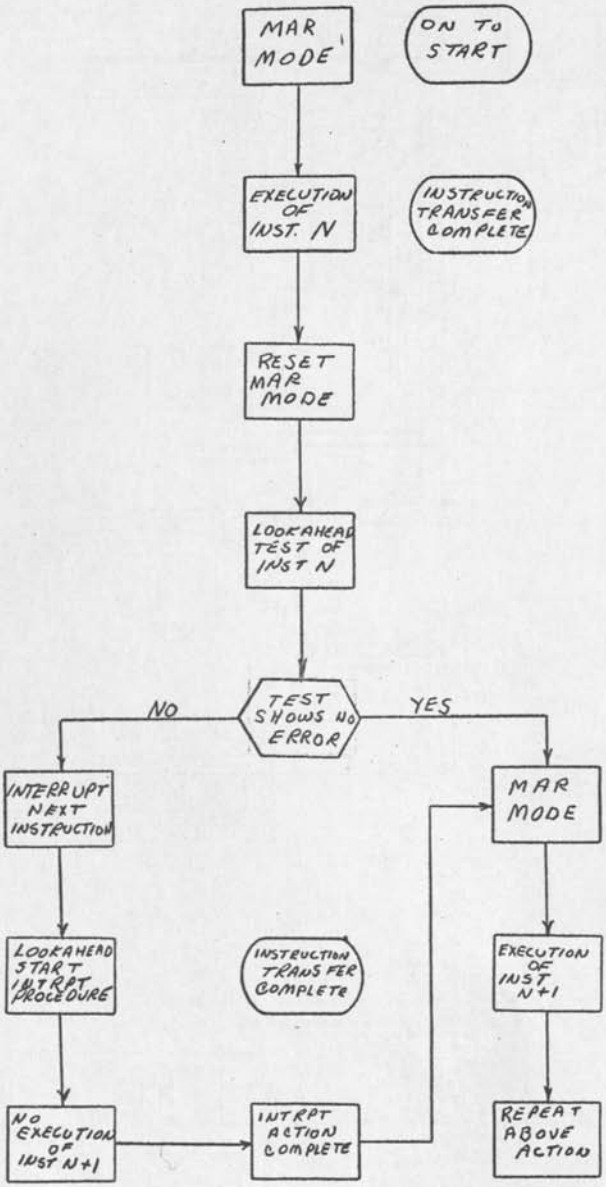


FIGURE 5.4-3. FLP STORE INT



MAR MODE SIMPLIFIED ACTION

FIG 5.7-1

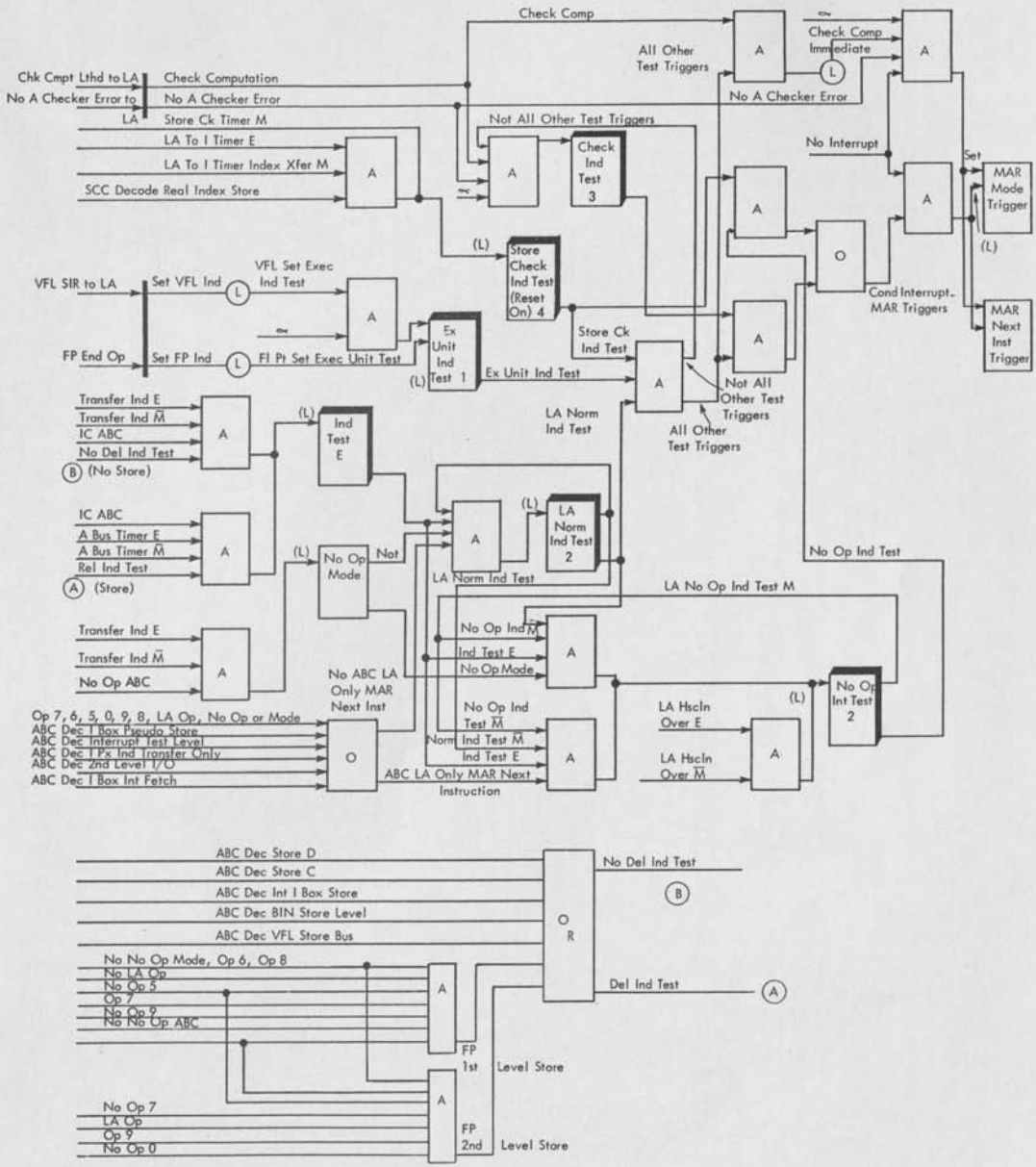


FIGURE 5.7-2. MAR MAR MODE TEST

SCC DECODE	OP CODE AND TAGS																AND'ED WITH	OTHER OR COND
	0	5	6	7	8	9	LAOP	LAOP	No Op Mode	INT	LC	LF	Disc	WBC	LAAR Int	IC		
Adv Illegal Pseudo Store	1					0	1										SCC Late Dec Enable	
GI To Y Prep	1					1	1			1								
GI To Y Prep	1		0			1												
I Box Int Fetch	1					1	1			1								
Index Transfer	0					1								0				
Index Transfer							0											
Index Transfer	0					0	1											
Index Transfer Prep	0					1								0				
Index Transfer Prep							0											
Index Transfer Prep	0					0	1											
LA To I No Real Store	0					0	1											
LA To I No Real Store	1		0			1												
Real Index Store	0					1								0				
Real Index Store							0											
Restore Pseudo Store	0					0	1											
Store Check Required	0					1								0				
Store Check Required							0											

FIGURE 6.2-1. SCC DECODING

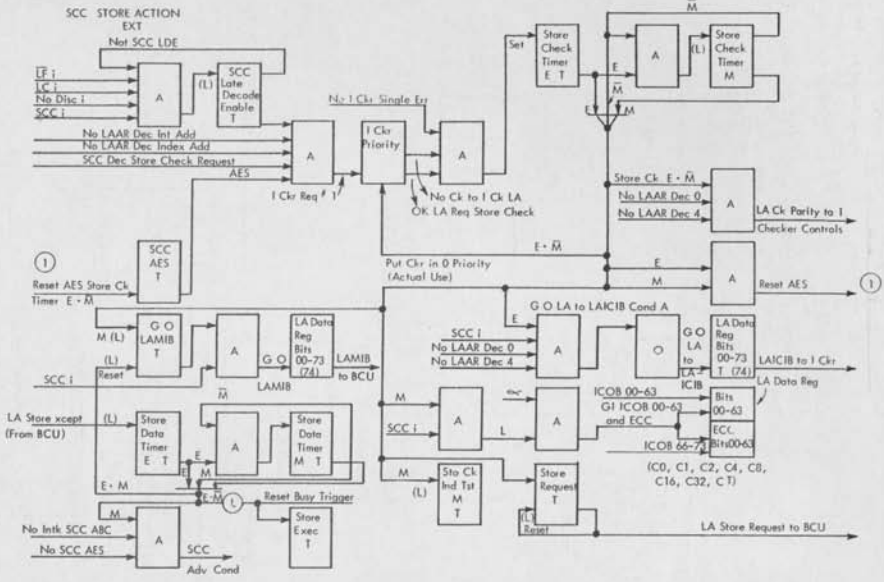


FIGURE 6.2-2. SCC STORE ACTION EXT

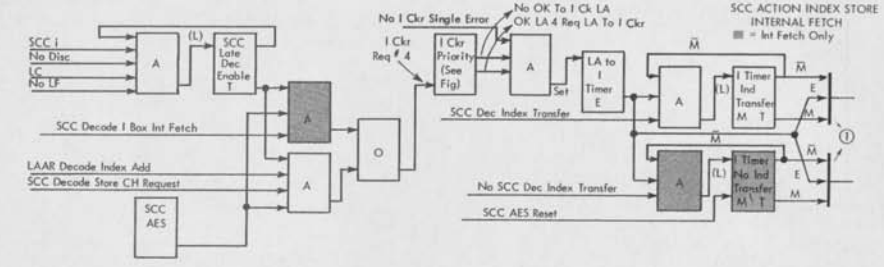


FIGURE 6.3-1. SCC ACTION INTERNAL FETCH

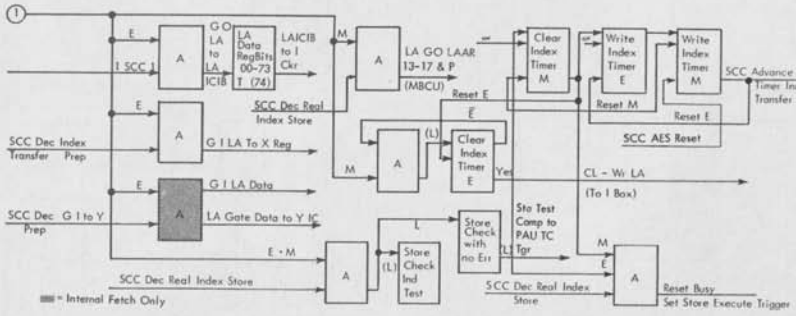
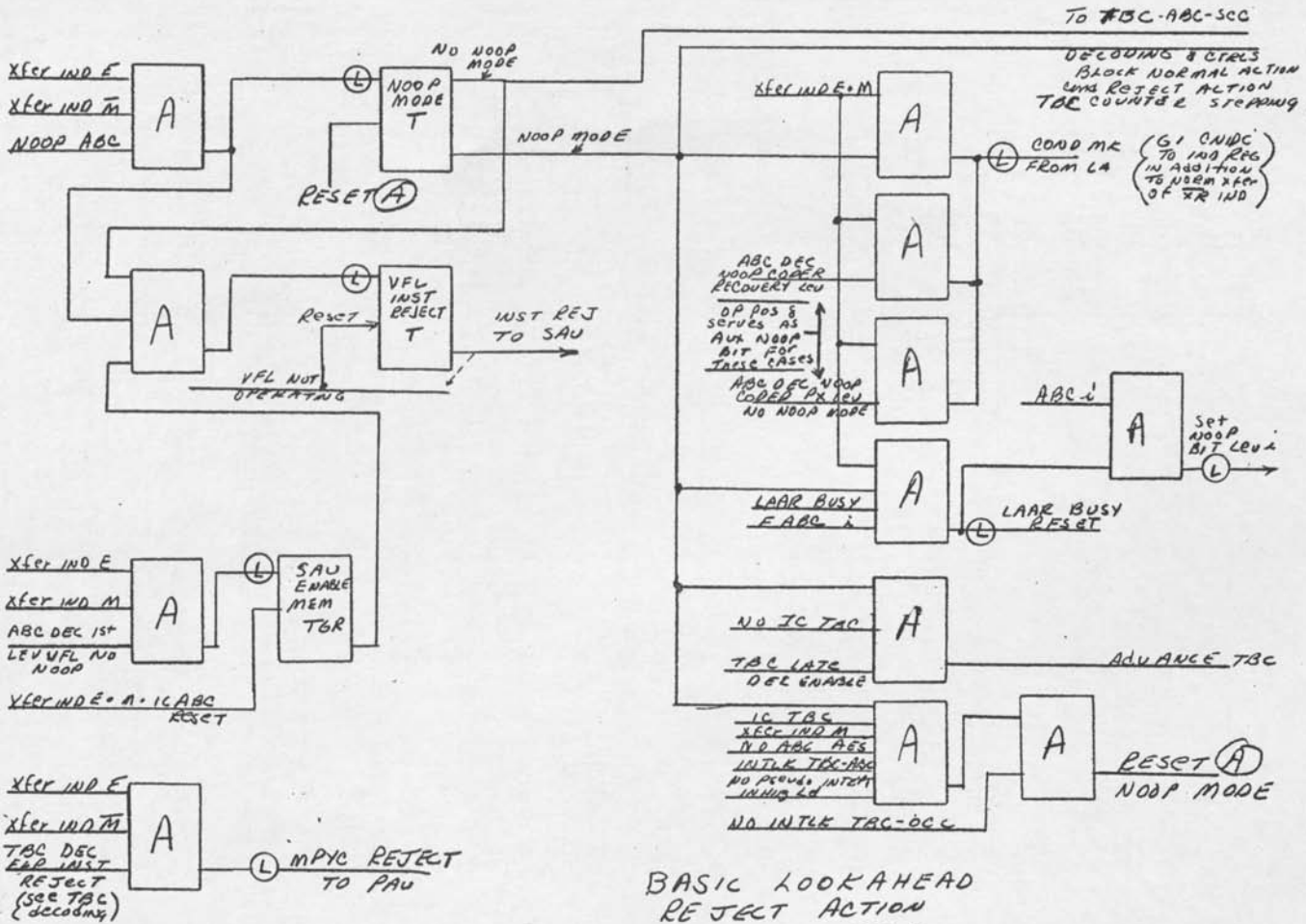
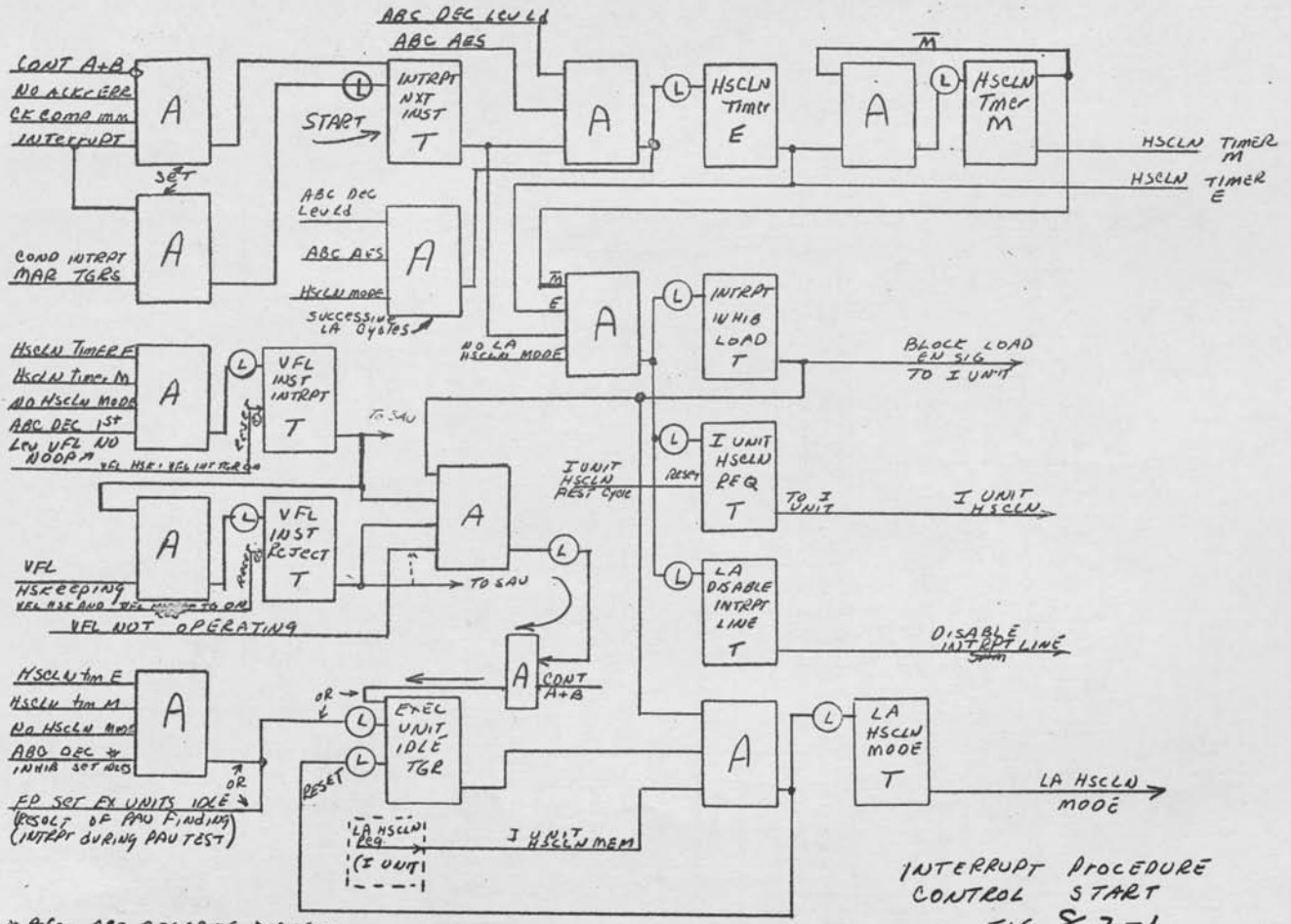


FIGURE 6.3-2. SCC ACTION STORE TO I UNIT



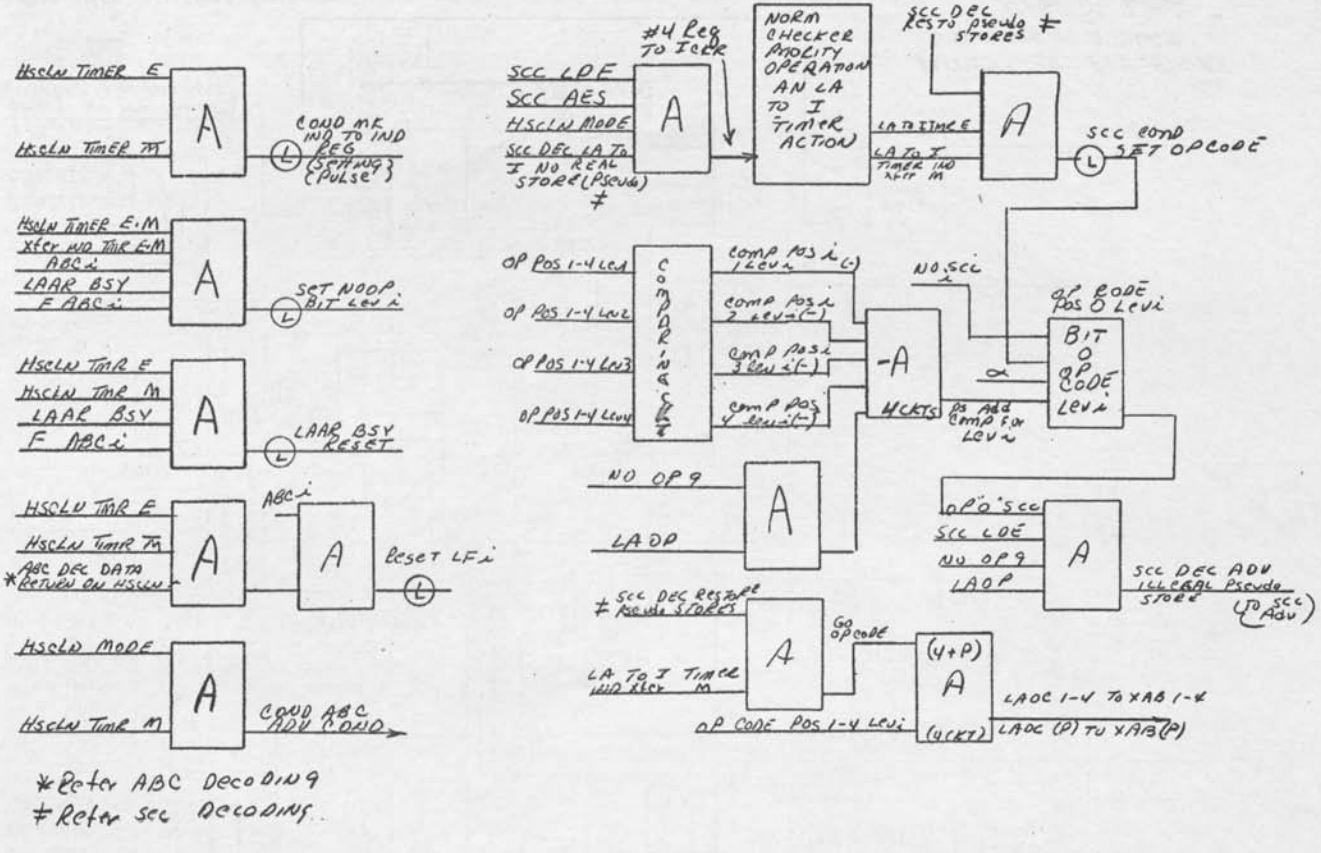
BASIC LOOKAHEAD REJECT ACTION
FIGURE 7.2-1



* Refer ABC DECODING DIAGRAM

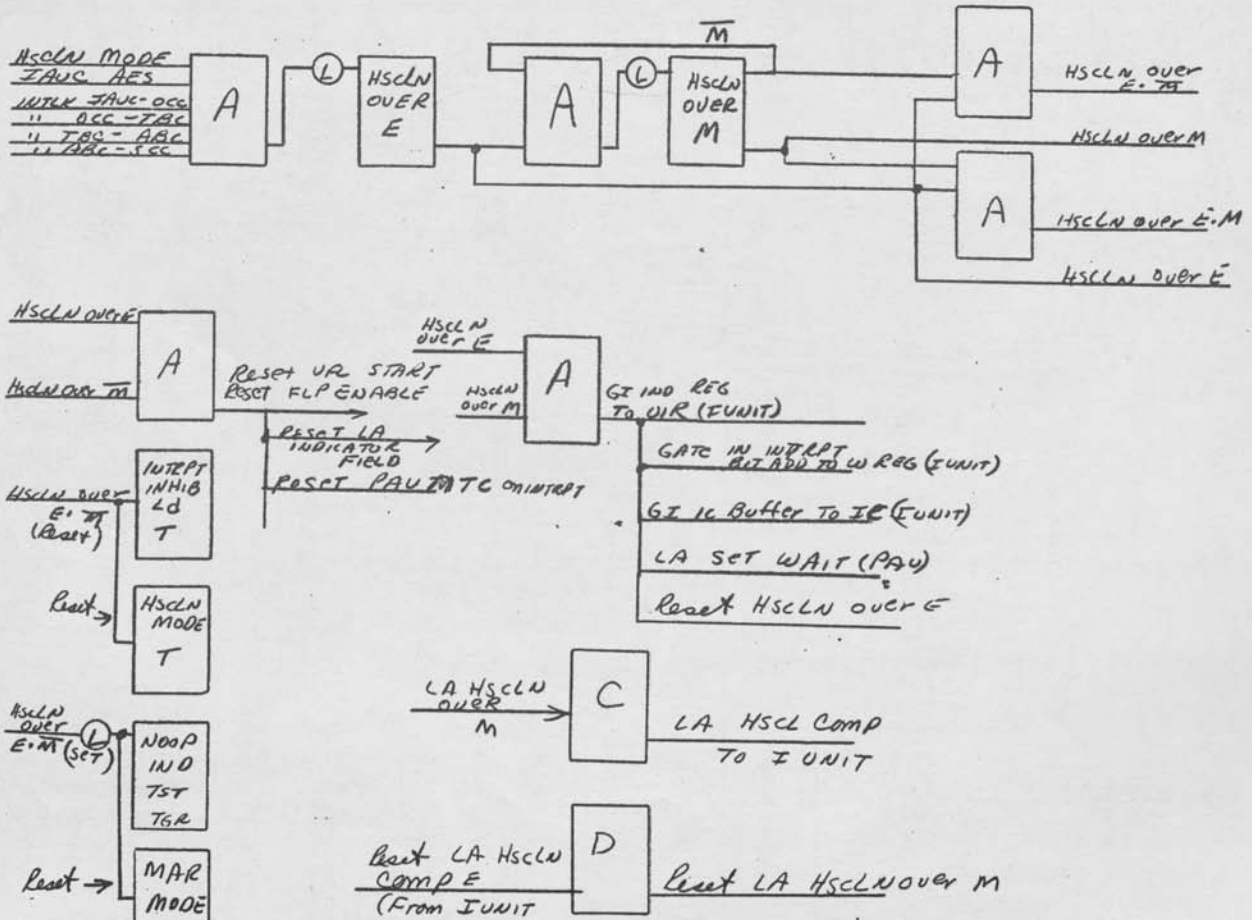
INTERRUPT PROCEDURE CONTROL START

FIG 8.2-1

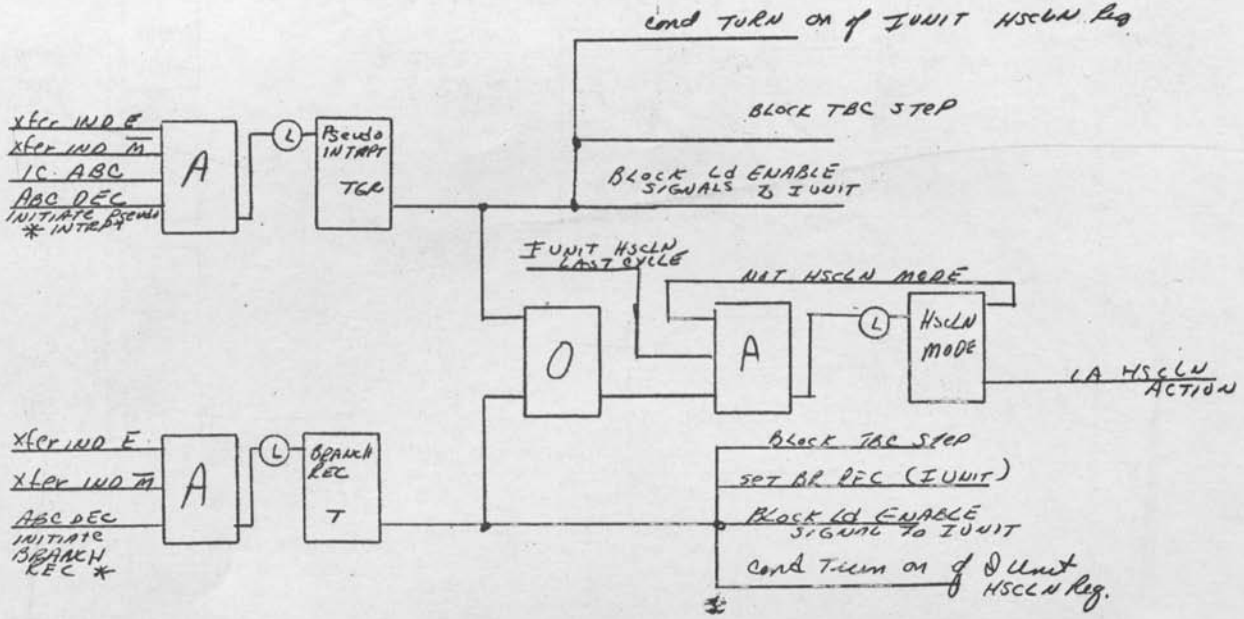


HOUSECLEAN TIMER ACTION

FIGURE 8.2-2



LA HSCLN OVER FIGURE 8.2-3



* Refer ABC DECODING DIAGRAM

PSEUDO INTERRUPT - BRANCH RECOVERY START

FIG 8.3-1