IBM 1401 DATA PROCESSING SYSTEM

INSTRUCTION LOGIC

Thomas M. Mierswa
Eastern Region Systems Dept.
November 1960

TABLE OF CONTENTS

INTRODUCTION

The purpose of these notes is to provide a simplified version of 1401 machine logic for IBM Field Systems personnel. A knowledge of these principles will aid in understanding more fully the functions that each instruction causes the machine to perform. This knowledge, it is hoped, will also stimulate ideas for better programming which will both reduce machine time and conserve memory space.

These notes were compiled from the Customer Engineering Manual and from the Logic Flow Diagrams used by Customer Engineers. No effort has been made to present the exact sequence of events within a machine cycle because a knowledge of partial cycle operation would be of no practical knowledge to the Field Systems Representative.

Only the instructions in the basic 1401 card system are presented.

In the diagrams and figures included in these notes the following abbreviations are used:

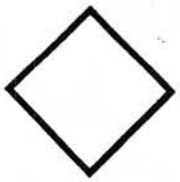
A-REG	-	A-Register
A-STAR	-	A-Storage Address Register
B-REG	-	B-Register
B-STAR	-	B-Storage Address Register
C. B.	-	Circuit breaker
Character	-	All bits including check bit, zone bits, digit bits and WM bit
C. L.	-	Console light
Digit	-	The 1, 2, 4, 8 bits and C-bit as required
E-Phase	-	The machine cycles required to execute an instruction
I-OP	-	Instruction Operation cycle during which the instruction operation code is accessed.
I-Phase	-	The machine cycles required to access the entire instruction
I-STAR	-	Instruction Storage Address Register
pos.	-	position
p. s.	-	program skip latch set if the normal sequential access of instructions is changed.
R. B.	-	Read back into storage from B-register. The entire character including the word mark bit is regenerated in the storage location indicated in the storage address register.
R. I.	-	Read into some register from storage
R. S.	-	Reverse scan where storage positions are accessed from a lower address (high order position) to a higher address (low order position) as opposed to a forward scan where storage is accessed from a higher address (low order position) to a lower address (high order position).

		When reverse scan is begun, the high order position is readdressed by not resetting the Storage Address Register.
SAR	-	Storage Address Register which controls the current storage location to be accessed.
S. F.	-	Standard form. This applies to sign indication. A field is considered plus if it has any zone combination other than a "B-bit" alone but a plus sign in standard form is an "A B bit" combination.
trig.	-	trigger
WM	-	Word mark
WO	-	Without
Z. S.	-	Zero Suppress latch

Positive logic only is shown in the diagrams. For example, if a STAR is to be modified it is so shown but if it is not to be modified the step is omitted rather than shown to "not happen". If a latch or trigger is turned on it will remain on until it is shown to have been turned off or is reset on the I-OP cycle of the next instruction. For example, if A-cycles eliminate is turned on, all subsequent A-cycles will be eliminated until the next I-OP cycle or the machine is told to start an A-cycle.

The following symbols are used in the diagrams and have meanings as indicated.

- machine operation



- machine interrogation

- machine cycle indication

- connector

DATA FLOW

Figure 1 is a schematic of the overall flow of information through the 1401. A general description of each component follows.

Core Storage Unit

The core storage unit is the "center" of all data flow in the 1401 system. The over-all objective of the system is to receive data from cards in the card read-punch, process this data, and send the resultant data to the punch or printer. The core storage unit is part of all these functions. Each character of information enters or leaves the storage unit in BCD form.

Information is read out of storage during the early part of a cycle. Readout is actually accomplished by setting all the cores to zero. A core set at "one" will, when it flips from one to zero during readout, induce a voltage on one of the wires running through the center of the core. This is recognized as a bit.

Information is read into storage during the later half of a cycle. When information, that is readout of a storage location, is to be retained in the particular location, it is "transferred" from one of the registers back into the same cores that it was read out of. This happens on the later half of the same cycle.

Data Lines

Data flow paths shown as single lines are actually eight lines, one for each BCD "bit value", plus one additional line for word marks. The lines leading to the inhibit drive are called inhibit lines, and are so named because they will prevent, or inhibit, the setting of cores unless "activated" by a bit of information.

Information to storage is through the inhibit drive while information from storage is through the B-register.

A- and B-Registers

The A- and B-registers are single-character storage devices used for storing the specific characters being treated. To process any information from the storage unit, it must first be brought to one or both of these registers. For example, in certain operations involving two data fields, an A-field character is stored in the A-register (through the B-register), and then a B-field character is stored in the B-register. Characters thus stored may be added, subtracted, compared, or other-wise treated. When necessary, each of these registers can transfer its character "back" to storage, preventing the loss of the information in the storage unit. This is necessary because the cores of a position are all set to zero when the particular location is read out.

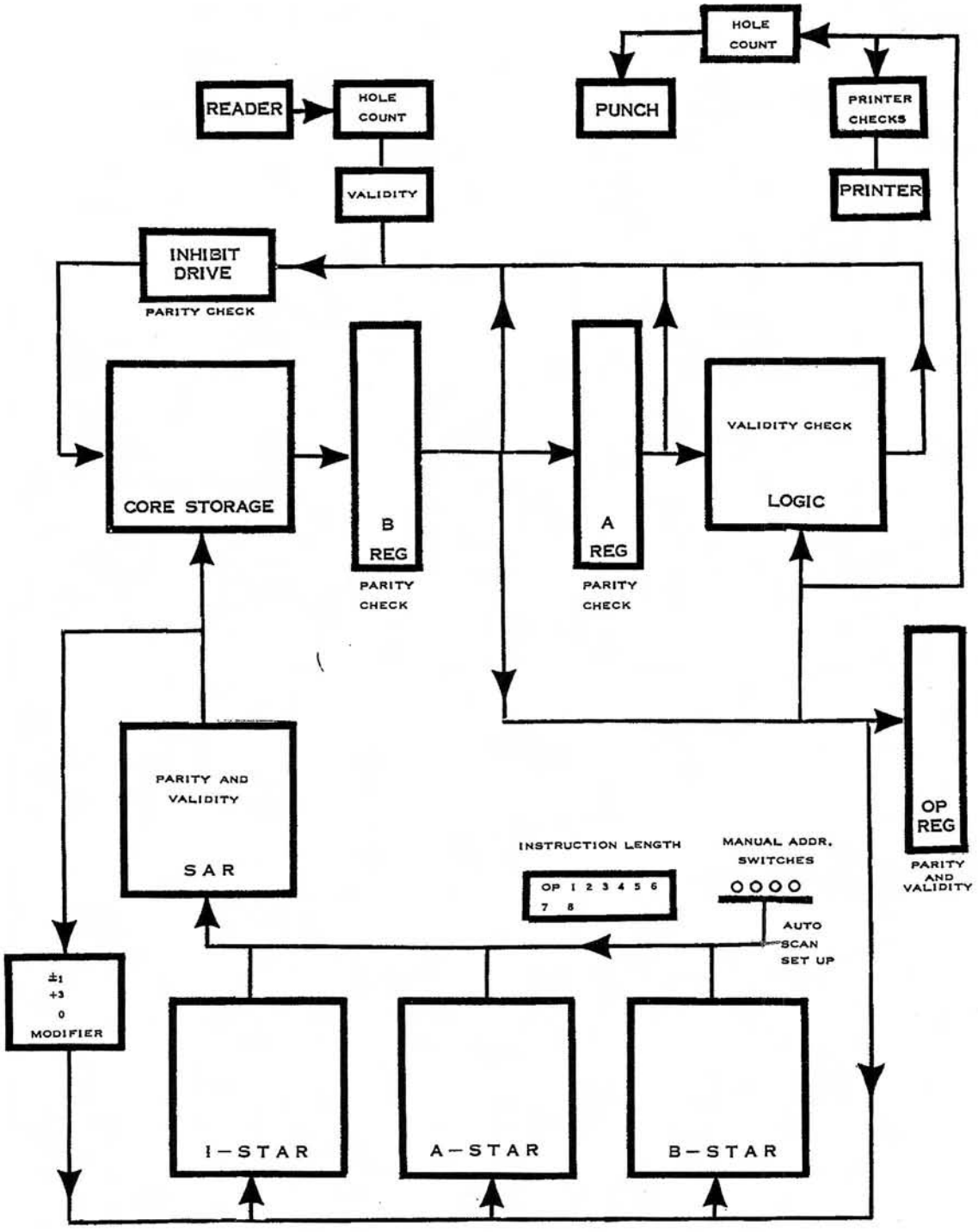


FIGURE I - DATA FLOW AND CHECKING FEATURES

Input/Output

Holes in cards read at the card reader cause corresponding BCD characters to be applied to the inhibit lines for storage in the storage unit. To print or punch data, the storage unit supplies each character of information through the B-register. The circuits are controlled so that this data, in BCD form, is translated to a form and timing relationship compatible with the appropriate output device.

Operation Register

The operation register is used for storing an op code for the duration of that operation. This register consists of seven latches (A, B, C, 1, 2, 4, 8) which are capable of storing the op code bit configurations. Any latches that are set in the op register are decoded into a single line, named for the operation, which controls the system during the operation.

I-, A-, and B- Address Registers

The 1401 uses three, 3-position address registers. These address registers are referred to as STARS (Storage Address Registers). Each STAR contains the following latches:

1. Five in the units and tens position (8, 4, 2, 1, C)

2. Six in the hundreds position (8, 4, 2, 1, C, A). A B-latch is used in the hundreds position if a 4k storage unit is used.

The I-STAR controls the system during I-phase. On I-phase, the I-STAR contains the address of the storage location to be read-out on the next I-cycle.

On E-phase, the A-STAR contains the address of the storage location to be read-out on the next A-cycle. The B-STAR contains the address of the storage location to be read-out on the B-cycle. All three STARS operate on the following sequence on any particular cycle:

1. Reset the units position and read in the units position.

2. Reset the tens position and read in the tens position.

3. Reset the hundreds position and read in the hundreds position.

4. Read out all positions, in parallel to the storage address register.

No checking is performed in the I-, A-, or B-STARS. Checking takes place in other circuitry after the STAR is read-out.

Instruction Length

The 1401 cycle control keeps track of the I-phase cycles by triggers. The instruction length determines the number of I-phase cycles. There can be as few as one I-phase cycle or as many as nine. Except for the "set word mark" instruction the word mark of the next instruction stops I-phase. The "set word mark" instruction causes I-phase to cease on I-6 cycle regardless of a following word mark. A blank in storage can also stop I-phase for the unconditional branch instruction.

CONTROL

In a stored program computer, such as the 1401, the computer must distinguish between instructions and data. The cycle-control circuits are the control center of the 1401. All central processing is regulated by cycle control. It controls the change from I-Phase to E-Phase and vice versa. During I-Phase the cycle control circuitry must direct the successive I-cycles and during E-Phase it provides A and B cycle control.

Address Register Modification

Because each storage cycle requires scanning at a new storage address, the address in either the I-, A-, or B-Storage address register is modified during the storage cycle in which it was used. During I-phase, instruction words are scanned from high-order to low-order position under control of the I-storage address register. This register increases its value by one for each I-cycle. During most E-phase operations, treatment of an A-and B- field requires the A- and B-storage address registers to decrease in value by one for each A- and B-cycle, respectively. Operations involving the printer require an address to be increased by three for each storage cycle, while at other times it is not necessary to modify an address at all.

The I-, A-, or B- storage address register controlling storage addressing is "updated" during the same cycle in which it is used by a modified address received from the modifier. The modifier output is obtained by adding ±1 or +3 to the address received from the storage address register. This register always contains the address just used. Therefore, the address it contains will always be the one to be modified.

Modification During I-Phase

During I-cycles, the I-storage address register reads into the storage address register, which in turn selects the storage position containing the desired instruction word character. In most I-phase operation, the I-address register must be modified by +1 during each I-cycle, except the last I-cycle. Modification must not occur on the last I-cycle, because this is the cycle in which the Op Code and word-mark of the next instruction word, in sequence, appears in the B-register - the word-mark which ends the I-phase. The first cycle (I-Op) of the next I-phase begins at the same storage address scanned on the last cycle of the previous I-phase, so that the OP Code may again be brought to the B-register. At this time it is also stored in the OP-register.

For example, assume a section of the stored program contains the following instruction words at the addresses shown below each character:

	wm			wm						
Instruction word:	OP	-	d	OP	-	A	-	A	-	A
Address:	541		542	543		544		545		546
Cycle:	I-op		I-1	I-2						
				I-op		I-1		I-2		I-3

The I-address register will be modified by +1 for each I-cycle except 1-2, because storage address 543 must be treated both at I-2 time of the first I-phase shown, and at I-op time of the following I-phase.

Although normal I-phase operation consists of scanning instruction word characters in sequence, flexibility of the 1401 is increased by two methods of completely changing the I-address (contained in the I-storage address register) at any time: one under program control, and one by "manual" means. The program control method uses the A-address (in the A-storage address register) as the address of the next desired instruction word. In this case, the storage address for the next I-op cycle is taken from the A-address register. During I-op cycle, the modifier receives this address from the storage address register in the normal manner, adds +1, and reads into the I-address register. During cycle I-1, addressing occurs as usual, under control of the I-storage address register which now contains the new I-address, already modified by +1.

The "manual" method of entering a new I-address is by means of the manual address switches (Figure 1). These switches allow the operator to set the desired address and store it in the I-address register (or the A- and B-address registers) from the console.

Modification During E-Phase

During E-phase, the storage address register selects the A- or B-field position to be treated, by addresses received from either the A- or B-storage address registers. Because an A-field character is addressed during an A-cycle, the A-address register reads into the storage address register, and is modified, during A-cycles. The B-register reads into the storage address register, and is modified during B-cycles. Modification of the A- or B-address register is done during the cycle in which the register is used, and by either a ±1 or +3, depending on the operation being performed. Most E-phase operations require forward scanning, which requires modification of the A-or B-storage address registers by -1 for each position scanned. Other E-phase operations require that the data word in the B-field be reverse scanned; the B-address register will be modified by +1 for each character position scanned.

Input/output operations of the card read-punch or the printer always use certain areas of storage. Addresses of the high-order positions of these areas are supplied to the storage address register by the auto scan setup (Figure 1), which stores an address directly in the storage address register on the first cycle of E-phase.

INSTRUCTIONS: I-PHASE

The treatment of data words, or other functions of the 1401 system, is controlled by instruction words. A "copy" of the instruction word about to control the system is first scanned from the storage unit, through the B-register, and placed in special instruction word registers. Once stored in these registers, the instruction word can satisfy its major objective: to initiate a function (the OP Code) of the instruction upon the data fields specified by the body of the instruction word.

I-Phase

The proper instruction word is read out of the storage unit and into the proper registers during I-phase. At the same time it is also transferred back into storage unit for later use.

I-Phase is broken down into 11.5μ sec storage cycles called I-cycles. A total of nine I-cycles is possible, however, the exact number of I-cycles taken during any I-phase is dependent upon the length of the instruction word. One I-cycle is required for each character in the instruction word plus one additional I-cycle which is required to recognize the end of an instruction word (the next instruction word "word mark"). The "set word mark" instruction is an exception to this rule because I-phase is automatically ended after I-6 cycle.

The following narrative and flow diagrams describe cycle by cycle I-Phase. The I-Phase process differs for some instructions and will be discussed later as those particular instructions are explained.

I. The cycle control unit signals that the last execute cycle has been taken and it is time to read a new instruction:

   A. Turn on the I-cycle latch (I-phase)
   B. Turn on the first I-cycle trigger (I-op)
   C. Transfer the address of the next instruction from the I-address register to the storage address register. The I-address register remembers the proper address to start at, from the previous I-phase (the instructions are in sequence). The storage address register activates the lines to cause readout of a particular location in the storage unit.

II. During the 11.5μ sec I-Op cycle:

A. The operation code (add, subtract, etc.) is read into the B-register from the storage position.
B. The Op Code is transferred from the B-register back into the storage location.
C. The Op Code is gated into the op register from the B-register. (The Op code must be accompanied by a word mark).
D. The address in the storage address register is increased by +1 (if it was 501 it would be increased to 502) and transferred into the I-address register for use on the next I-cycle.
E. The I-op trigger also lights the console Instruction Length Light, "I-OP."

III. At the end of the I-Op cycle, the I-Op trigger turns off and the I-1 trigger turns on. During I-1 cycle:

A. The increased address from the I-address register is transferred into the storage address register. (This is the address of the next character in the instruction word).
B. The contents of that storage unit location are read into the B-register. (This would normally be the high order position of the A data field address.)
C. The content of the B-register is transferred back into the same storage location.
D. The content of the B-register is also gated into the hundreds/thousands position of the A- and B-address registers. (The A- and B-address registers will contain the addresses of the A and B data fields at the end of I-phase.)
E. Increase the address in the storage address register by +1 and gate it into the I-address register.
F. The I-1 trigger also lights the instruction length light 1.

IV. At the end of the I-1 cycle the I-1 trigger turns off and the I-2 trigger turns on. During I-2 cycle:

A. The address in the I-storage address register (I-STAR) is gated into the storage address register.
B. The content of that storage location is read into the B-register. (This is normally the tens position of the A-field data address.)
C. The content of the B-register is:

    1. transferred back into storage
    2. gated into the tens position of the A and B storage address registers (A- and B-STARS)

D. Increase the address in the storage address register by +1 and gate it into the I-storage address register (I-STAR).
E. The instruction length light "2" is turned on.

V. At the end of I-2 time, the I-2 trigger is turned off and the I-3 trigger is turned on. During I-3 time the same procedure is followed as in I-2 time except the content of the B-register is gated into the units position of the A- and B-STARS.

VI. I-4 time is the same as I-3 time except that this time the content of the B-register is gated into the hundreds/thousands position of the B-STAR only. The transfer of the A-data-address in the A-STAR was completed on the last I-cycle.

VII. I-5 time is similar to I-4 time except that the B-register contents are gated into the tens position of the B-STAR.

VIII. I-6 time is similar to I-5 time except that the B-register content is now gated into the units position of the B-STAR. The transfer of the B-data field address into the B-STAR is now complete.

IX. I-7 time is different from the others. If the character read out of storage on this cycle is accompanied by a "word mark", I-phase ends and "execute phase" starts. If the character read out on this cycle has no word mark, it is assumed to be a "d" character and it is gated into the A-register where it is retained for further analysis.

X. I-8 time occurs provided there was not a word mark at I-7 time. The character read out of storage on this cycle should be accompanied by a word mark (the word mark of the next op code).

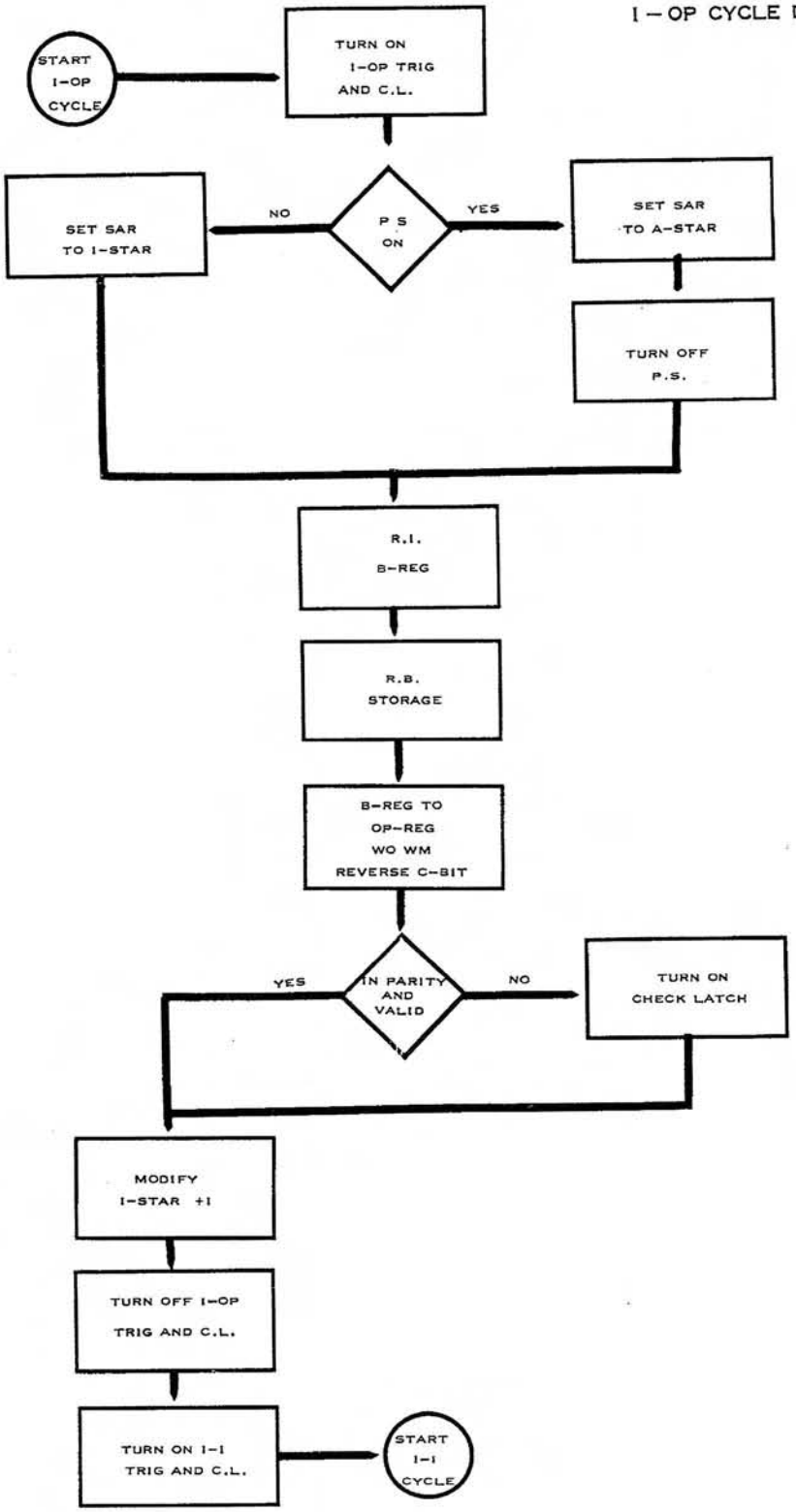
A word mark recognized during I-1, I-2, I-4, I-5, I-7 or I-8 would also end I-phase and start E-phase.

When I-phase is complete, the OP Code will have been stored in the OP-register; the A-address (if any) in the A-address register; the B-address (if any) in the B-address register; and the d-character (if any) in the A-register.

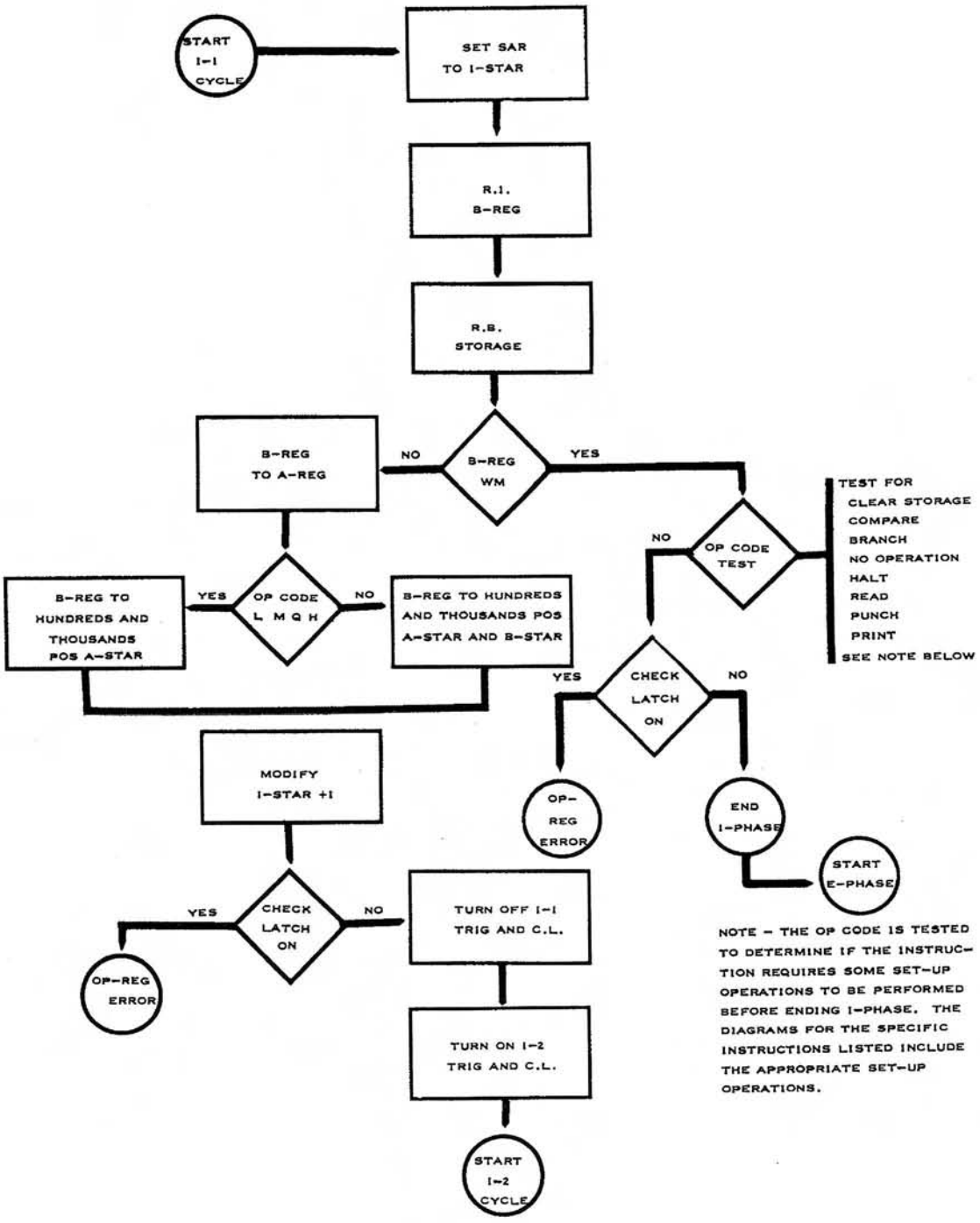
Recognition of d-Character

During I-phase, a d-character accompanying an instruction word must be stored in the A-register if it is to control the system's operation. As each character of an instruction word is scanned from storage to the B-register and appropriate address register, it is also stored in the A-register -- even though the character may not be a true d-character. The A-register is reset at the beginning of every I-cycle so that it may receive the next character of the instruction word. When the word mark of the next OP Code appears in the B-register, the normal reset and read-in of the A-register is prevented -- saving the last character of the instruction word about to control the system in the A-register, whether it is a d-character or not.

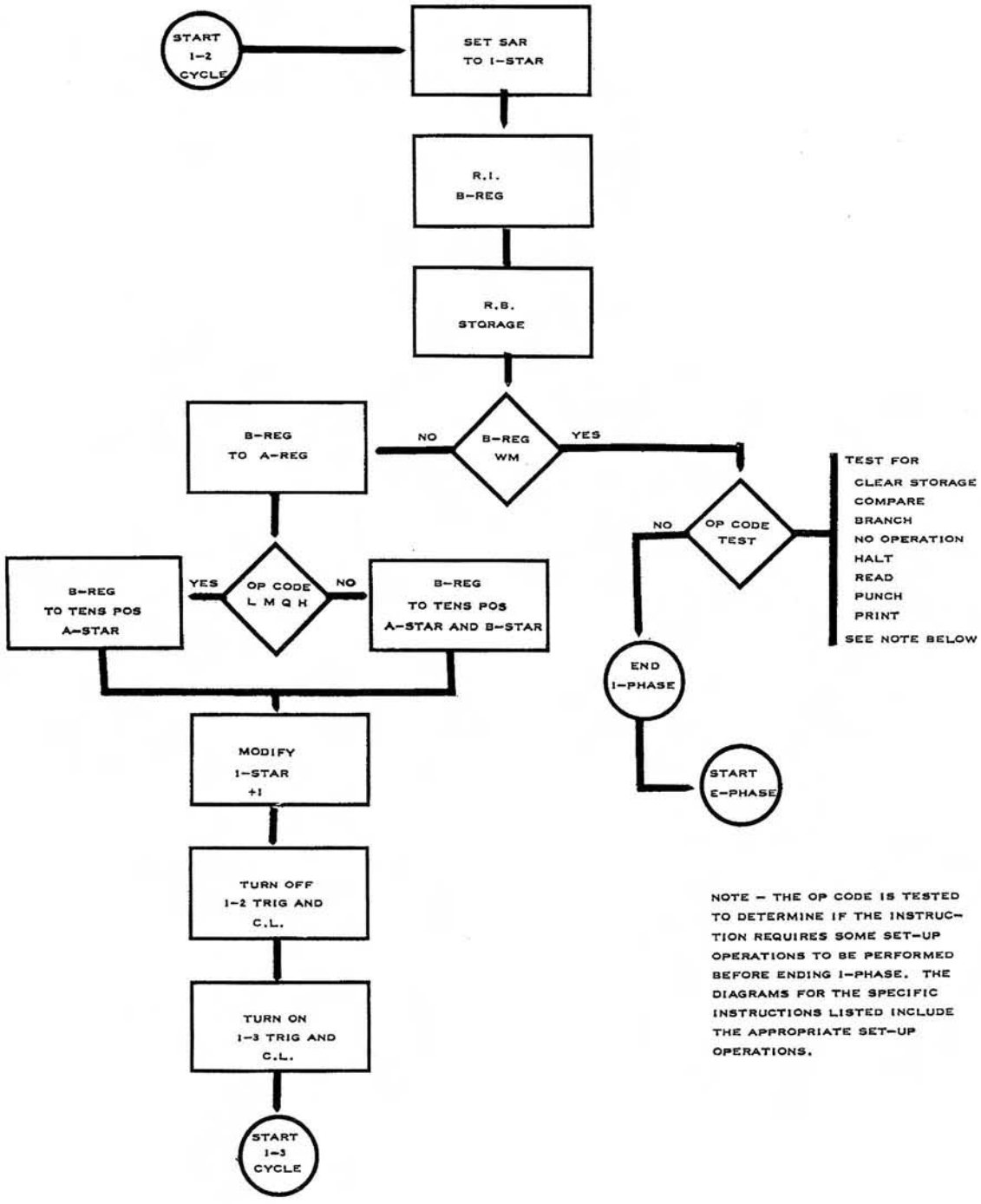
I – OP CYCLE DIAGRAM



I – I CYCLE DIAGRAM

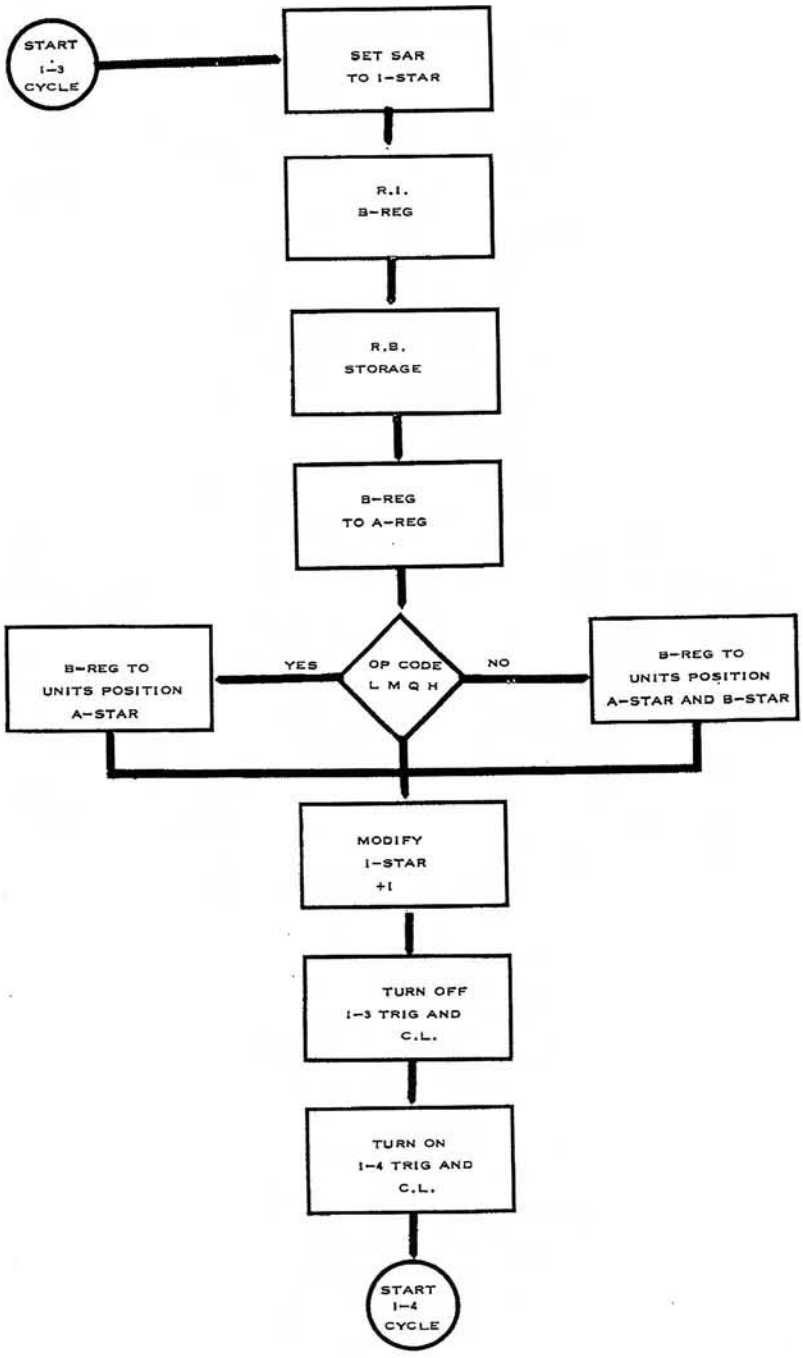


I – 2 CYCLE DIAGRAM

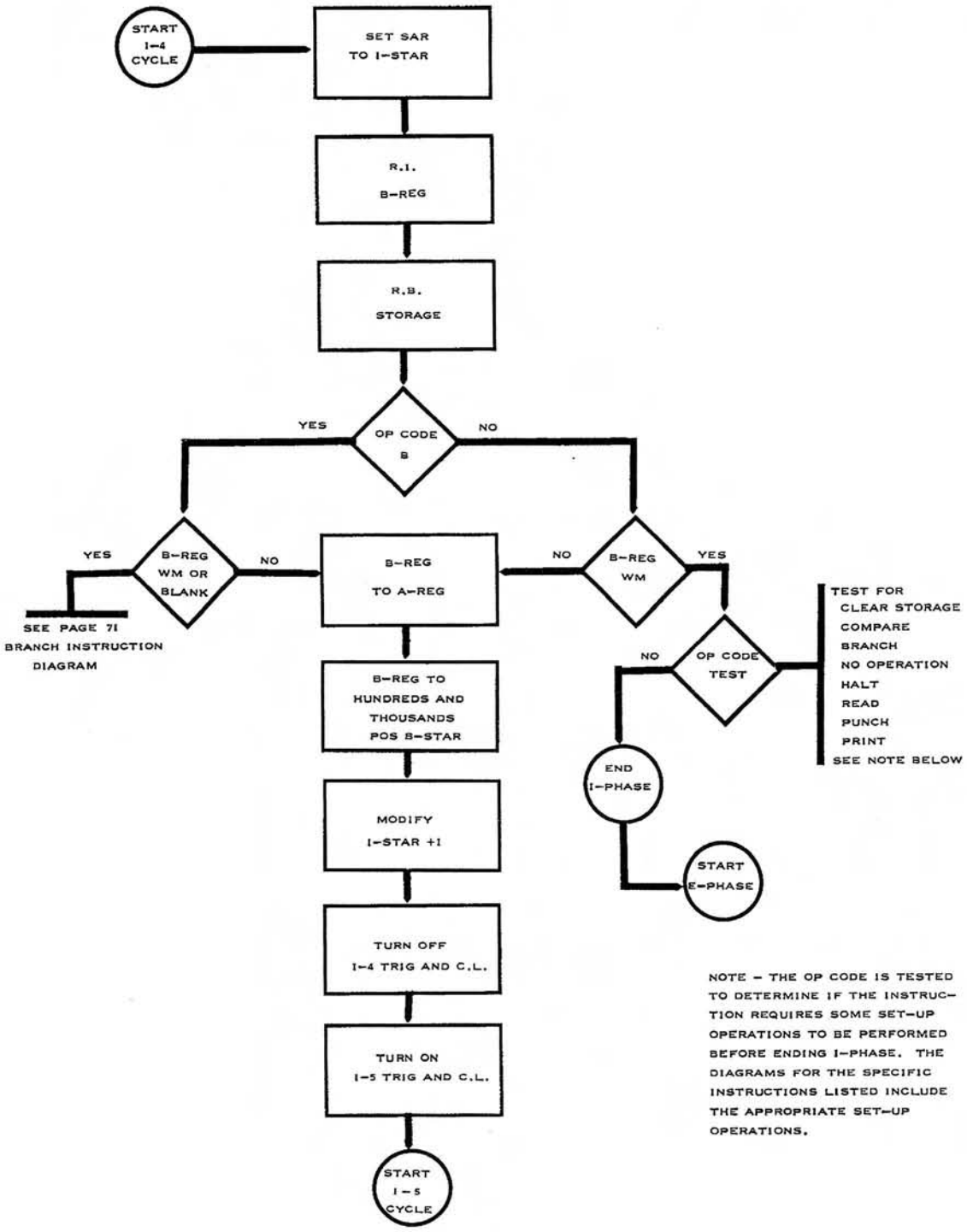


NOTE – THE OP CODE IS TESTED TO DETERMINE IF THE INSTRUCTION REQUIRES SOME SET-UP OPERATIONS TO BE PERFORMED BEFORE ENDING I-PHASE. THE DIAGRAMS FOR THE SPECIFIC INSTRUCTIONS LISTED INCLUDE THE APPROPRIATE SET-UP OPERATIONS.

1 – 3 CYCLE DIAGRAM

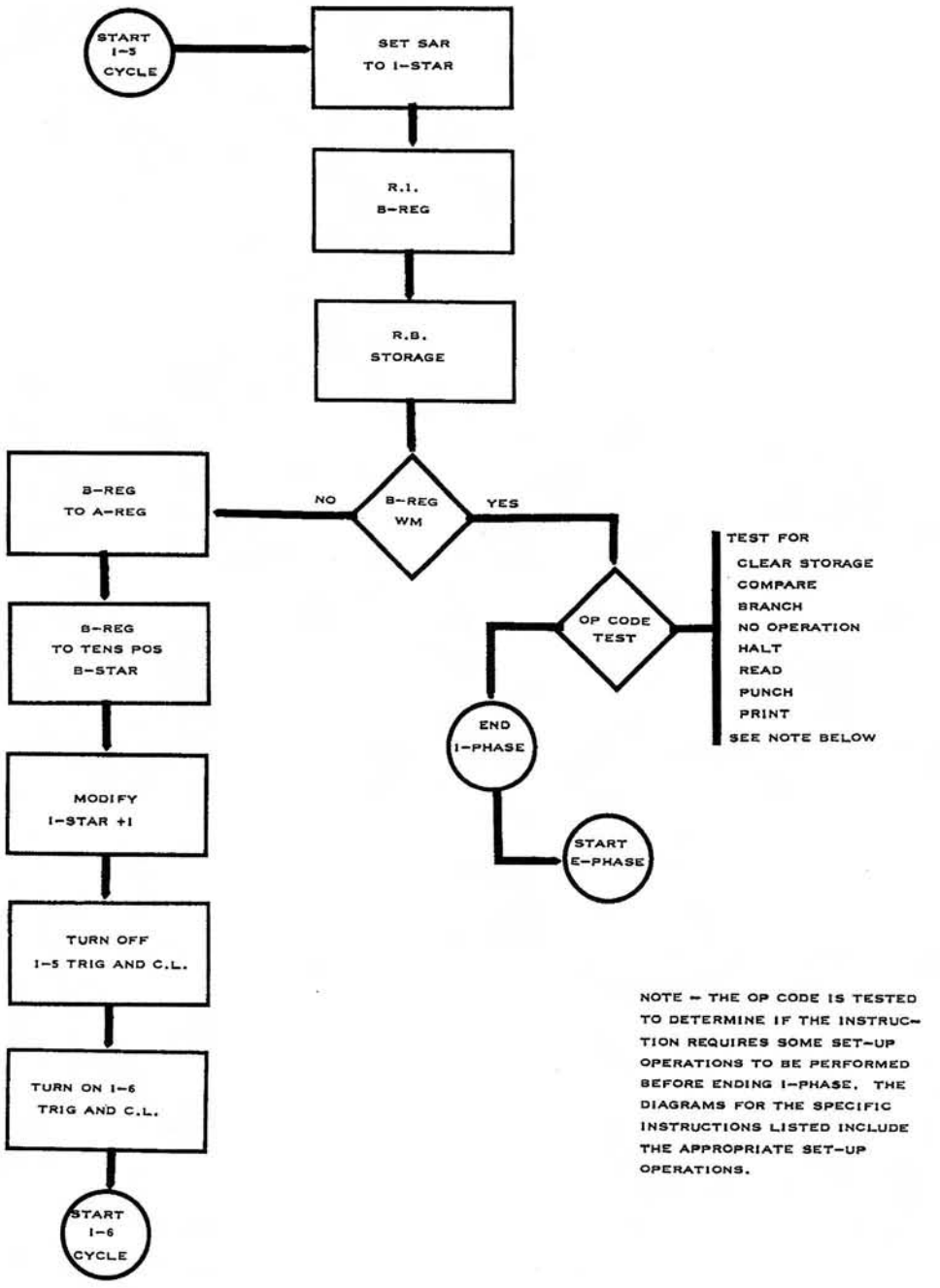


I – 4 CYCLE DIAGRAM



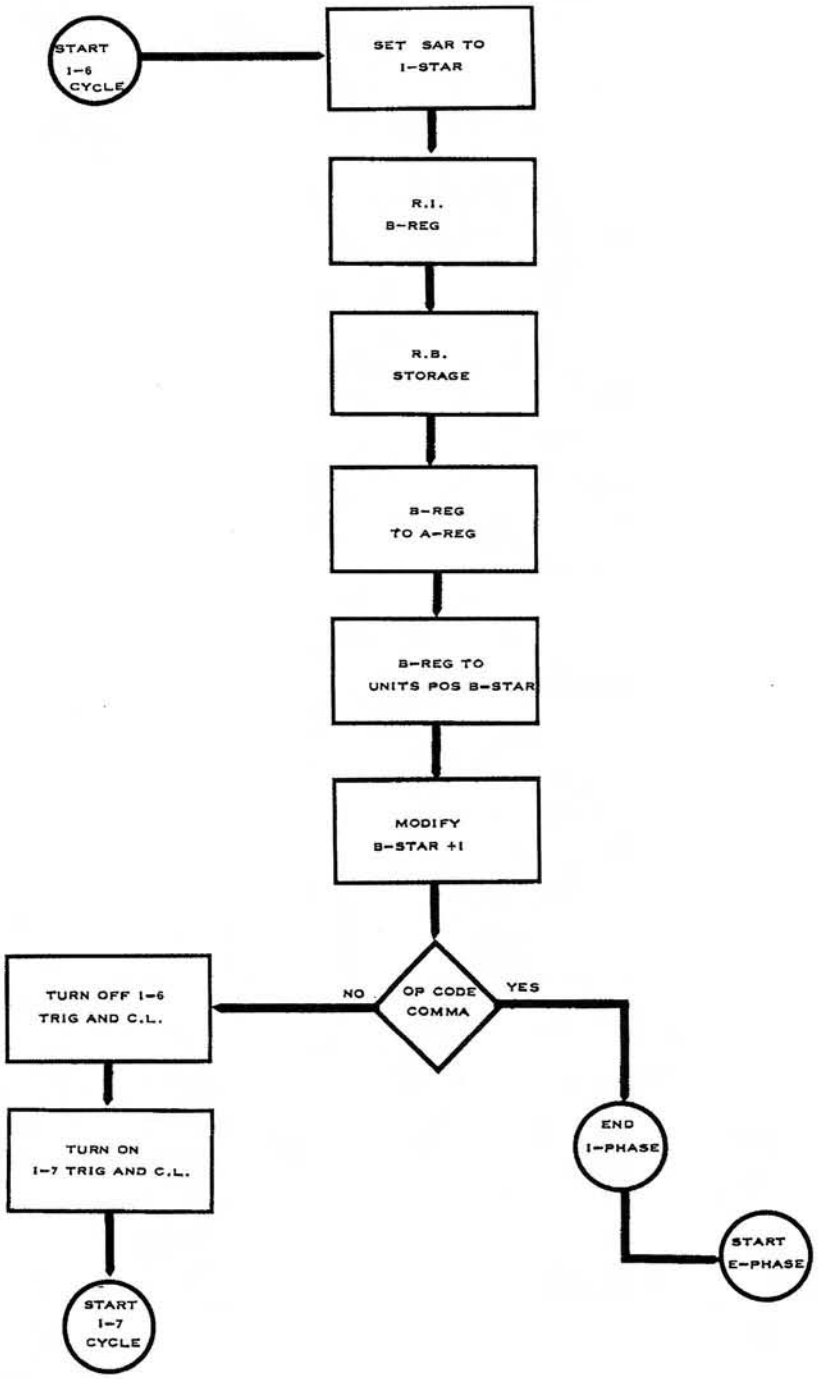
NOTE – THE OP CODE IS TESTED TO DETERMINE IF THE INSTRUCTION REQUIRES SOME SET–UP OPERATIONS TO BE PERFORMED BEFORE ENDING I–PHASE. THE DIAGRAMS FOR THE SPECIFIC INSTRUCTIONS LISTED INCLUDE THE APPROPRIATE SET–UP OPERATIONS.

I – 5 CYCLE DIAGRAM

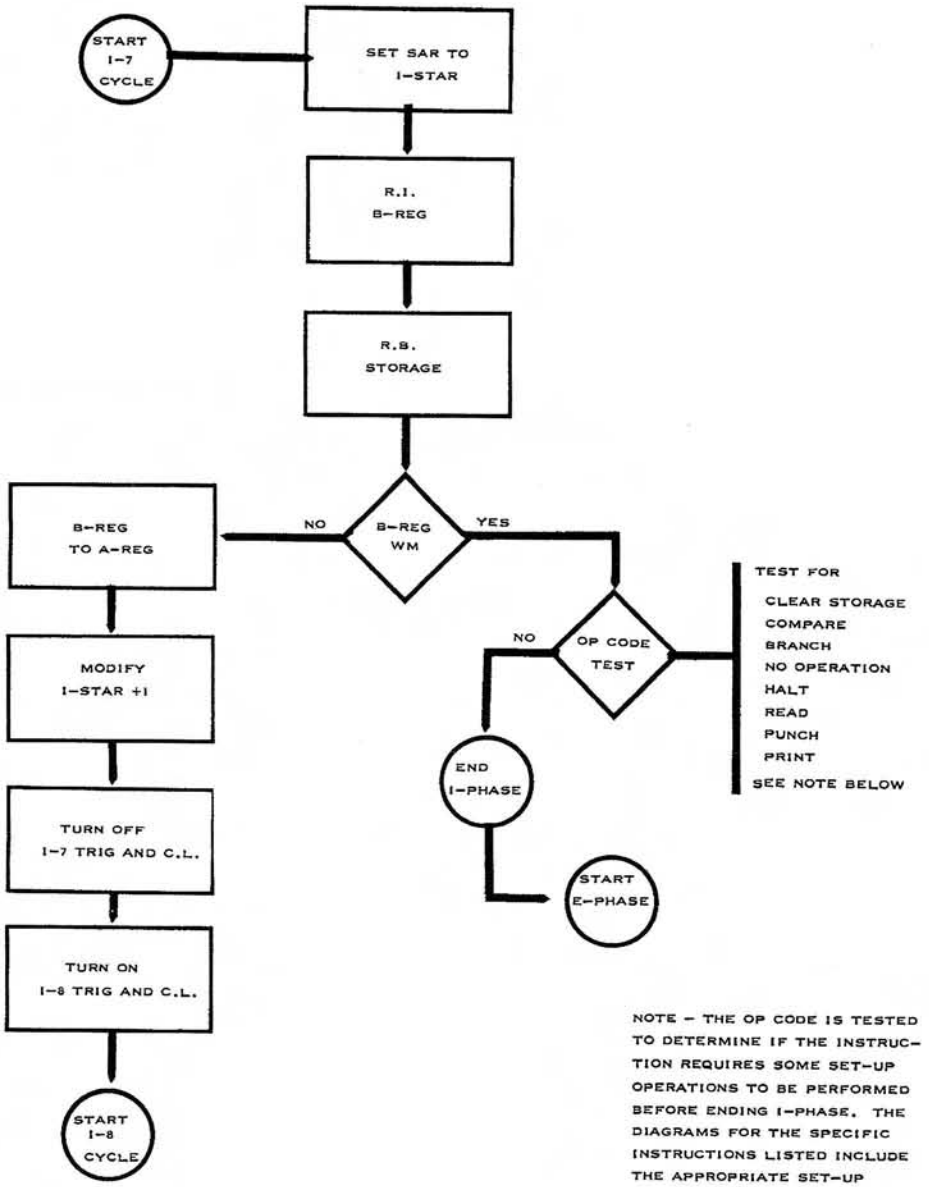


NOTE – THE OP CODE IS TESTED TO DETERMINE IF THE INSTRUCTION REQUIRES SOME SET-UP OPERATIONS TO BE PERFORMED BEFORE ENDING I-PHASE. THE DIAGRAMS FOR THE SPECIFIC INSTRUCTIONS LISTED INCLUDE THE APPROPRIATE SET-UP OPERATIONS.

I – 6 CYCLE DIAGRAM

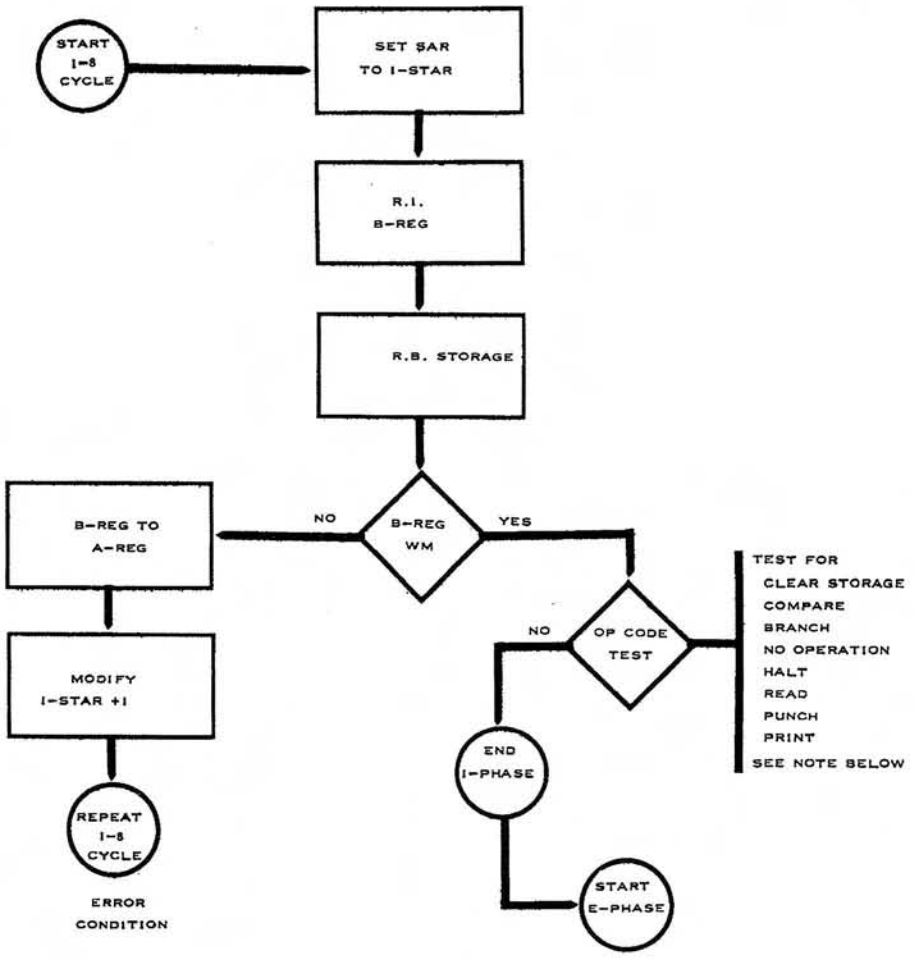


I – 7 CYCLE DIAGRAM



NOTE – THE OP CODE IS TESTED TO DETERMINE IF THE INSTRUCTION REQUIRES SOME SET-UP OPERATIONS TO BE PERFORMED BEFORE ENDING I-PHASE. THE DIAGRAMS FOR THE SPECIFIC INSTRUCTIONS LISTED INCLUDE THE APPROPRIATE SET-UP OPERATIONS.

I – 8 CYCLE DIAGRAM



NOTE – THE OP CODE IS TESTED TO DETERMINE IF THE INSTRUCTION REQUIRES SOME SET-UP OPERATIONS TO BE PERFORMED BEFORE ENDING I-PHASE. THE DIAGRAMS FOR THE SPECIFIC INSTRUCTIONS LISTED INCLUDE THE APPROPRIATE SET-UP OPERATIONS.

INSTRUCTIONS: E-PHASE

The end of an I-phase usually signals the beginning of an E-phase, during which data characters are transferred to the B-register in an 11 1/2 microsecond storage cycle. If the instruction word stored in the OP register calls for data processing of both an A-field and a B-field, an alternate sequence of addressing occurs: first from the A-address register for an A-field character; then from the B-address register for a B-field character. Because data words are scanned from low-order to high-order positions, data addresses stored in the A- and B-storage address registers must decrease by one for each storage cycle.

A- and B-Cycles

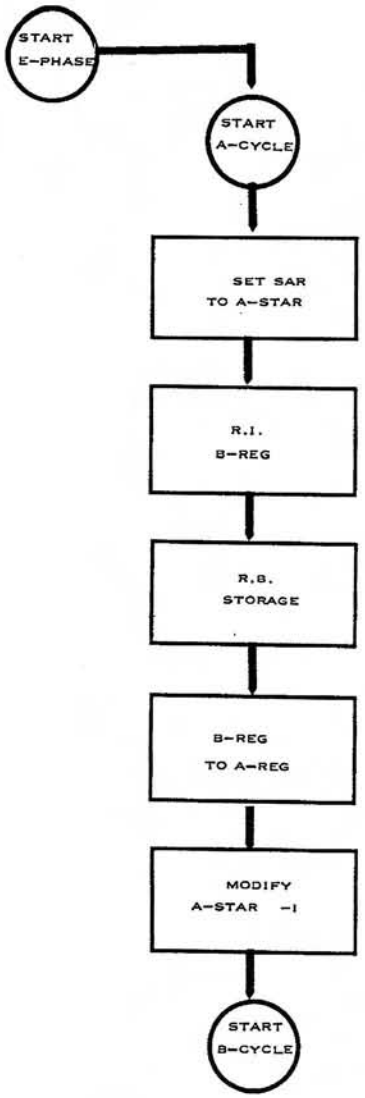
An 11 1/2 microsecond storage cycle in which the A-address register addresses storage is called an A- cycle. The B-address register addresses storage during a B-cycle. E-phase operation may consist of various combinations of A- and B- cycles, depending upon the operation called for by the instruction word scanned during the previous I-phase.

Terminating E-Phase

E-phase usually ends when a word mark is sensed in the B-register -- signifying that the end of the word in storage has been encountered. Some operations involving two data fields stop only when a A-field word mark appears; some only when a B-field word mark appears; and some when either A- or B-field word mark appears. This is easily controlled by conditioning the ending of E-phase with the type of cycle (A-cycle or B-cycle) in which a B-register word mark is sensed. For example, if an operation is to stop only on a B-field word mark, a word mark appearing in the B-register during an A-cycle is ignored. The operation stops only when a word mark appears in the B-register during a B-cycle.

The first E-Phase Diagram is shown on the opposite page. It has been called the Common A-cycle Diagram because many 1401 instructions include this type of A-cycle in their execution. It is set forth once here and reference is made to it for the move, move digit, move zone, load, compare, move and zero suppress, edit, reset add, reset subtract, true add, and complement add instruction.

COMMON A — CYCLE DIAGRAM



THIS TYPE OF A-CYCLE IS COMMON TO MANY 1401 INSTRUCTIONS. IT WILL NOT BE REPEATED FOR EACH ONE BUT REFERENCE WILL BE MADE TO THIS PAGE FOR THE MOVE, MOVE DIGIT, MOVE ZONE, LOAD, COMPARE, MOVE AND ZERO SUPPRESS, EDIT, RESET ADD, RESET SUBTRACT, TRUE ADD, AND COMPLEMENT ADD INSTRUCTIONS.

Set Word Mark (, AAA BBB)

This instruction causes a word mark to be set at both specified addresses without disturbing the data in those locations.

The character in the storage location is brought into the B-register and is transferred from the B-register back into storage along with a word mark. To maintain odd-bit parity, a check bit must be added or removed at this time (because adding the word mark changes the parity).

The execute portion of the operation starts with an A-cycle. In the A-cycle the following action takes place:

1. The A-data field address is gated into the storage address register from the A-STAR.
2. The character from the A-field is read into the B-register. It is also gated into the A-register on A-cycles, but it serves no functional purpose for this operation; it is easier to allow it than to prevent it.
3. The B-register (A-field data) is transferred back into storage.
4. The word mark inhibit drive line is activated. This adds a word mark to the A-field character.
5. The check-bit status is reversed. If it had no check bit, one is added, etc.

Next, a B-cycle occurs in which the same operation is performed on the storage location specified by the B-address.

Set Word Mark (, AAA)

Operation under control of this instruction is identical to that of the set-word mark instruction just described but with one exception, a word mark is set at the same storage location twice. This action is due to a characteristic of I-phase operation; the A-data address of most instruction words is stored in both the A- and B-address registers during cycles I-1, I-2, and I-3. Therefore, when E-phase for a single-address instruction begins, the A-data address is in both the A- and B-STARS: thus, a word mark is set in the same location two times, once during the A-cycle and once during the B-cycle. Again, this is not necessary, but it is easier to allow it than to prevent it.

Clear Word Mark (¤ AAA BBB)

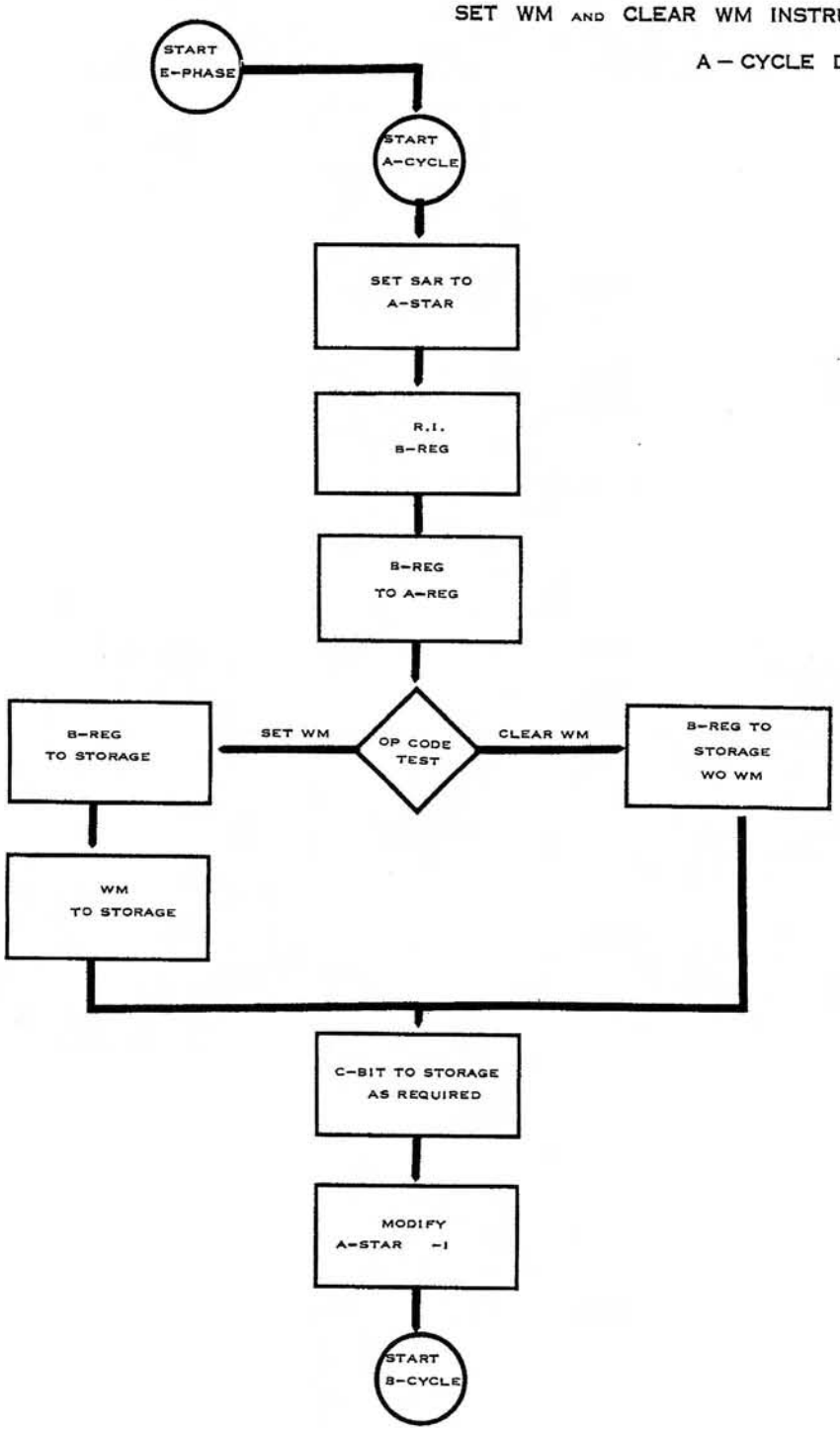
This operation is the same as the set-word mark operation, previously described, except that word marks are eliminated from the address specified. This is accomplished by blocking the word mark inhibit drive line which prevents the transfer of the word mark when the rest of the character is transferred back into storage.

Clear Word Mark (¤ AAA)

This operation is the same as that for the single-address set-word mark instruction, just described, except that the word mark is eliminated from the specified address. Also see Clear Word Mark with two addresses specified.

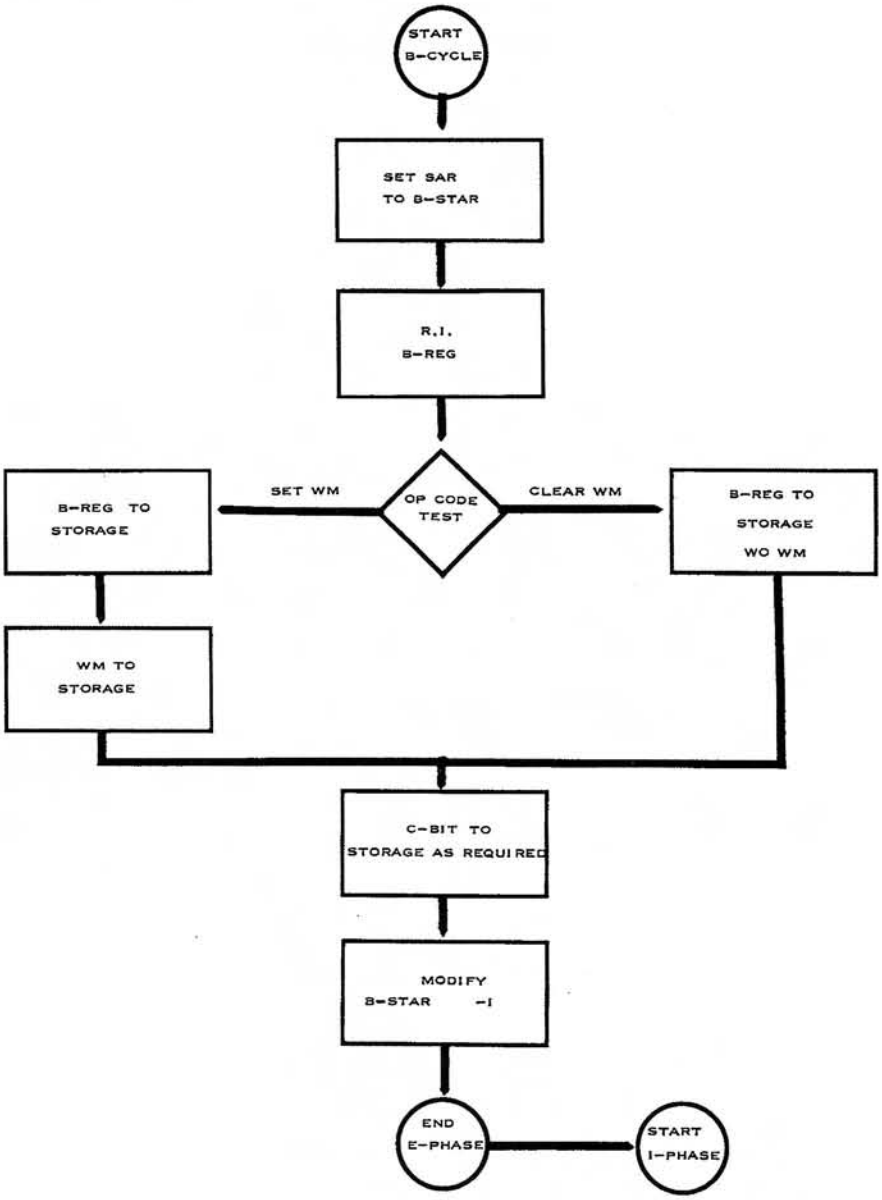
SET WM AND CLEAR WM INSTRUCTIONS

A – CYCLE DIAGRAM



SET WM AND CLEAR WM INSTRUCTIONS

B — CYCLE DIAGRAM



Move Digit, Move Zone & Move

Move Digit (D AAA BBB)

This instruction causes the numeric portion (8-4-2-1 bits) of the single character at the A-address to be transferred to the B-address. The entire A-field along with the zone portion (A-B bits) of the B-field remains unchanged. The word mark status of both fields remains unchanged.

The move-digit operation consists of one A-cycle and one B-cycle. The following action takes place on the A-cycle:

1. The address in the A-STAR is gated into the storage address register.
2. The entire content of the A-address location is read out of storage and into both the A- and B-registers.
3. The entire character is transferred back into the same storage location (A-field) from the B-register.

On the B-cycle:

1. The B-field address is gated into the storage address register from the B-STAR.
2. The entire content of the B-field is read out of storage and into the B-register only; the A-register continues to hold the character from the A-field.
3. The zone portion only of the B-register is transferred back into the B-field storage location.
4. The digit portion only of the A-register is transferred into the B-field storage location. Items 3 and 4 occur simultaneously.

Move Zone (Y AAA BBB)

This operation is the same as the preceding move digit operation except for the B-cycle. On the B-cycle the zone portion of the A-register is transferred to the B-field together with the digit portion of the B-register. The word mark status of both fields remains unchanged.

Move (M AAA BBB)

This instruction causes the data in the A-field to be stored in the B-field. Both the word mark status of either field and the data in the A-field remain unchanged on move operations.

Move is accomplished by a series of alternate A-and B-cycles. The Move operation is to bring A-field characters to the A-register on A-cycles, and to store them in the B-field on B-cycles. To move a "copy" of each character from the A-field to the B-field requires one A-cycle and one B-cycle. The operation begins with an A-cycle.

First A-cycle:

1. The A-field address is gated into the storage address register from the A-STAR.
2. The A-field character is read into both the A- and B-registers.
3. The A-field character is transferred back into storage from the B-register.
4. The A-field address, located in the storage address register, is reduced by one and gated back into the A-STAR.

On the first B-cycle:

1. The B-field address is gated into the storage address register.
2. The B-field character is read into the B-register. (The A-register retains the A-field character.)
3. The contents of the A-register is transferred into the B-field storage location.
4. The B-field address, located in the storage address register, is reduced by one and gated back into the B-STAR.

This process is repeated during each succeeding A- and B-cycle, addressing the next high-order position in the two fields until a word mark is detected; the move operation then stops.

Only one of the fields needs a defining word-mark if both fields are the same length. If the fields are of different lengths, the first word-mark encountered defines the length of both fields and stops the operation. If an A-field word mark is sensed first, the corresponding A-field character is stored in the B-field before the operation stops.

Move (M AAA)

This operation is the same as the move operation just described. The B-field address is the address that was left in the B-STAR upon completion of the last instruction. This operation should not be performed following a branch operation because the B-STAR will be blank.

During move, load, or store code operations, the normal B-STAR read-in circuit is blocked for I-cycles 1, 2, and 3. This permits sequential storing of information from fields in different locations.

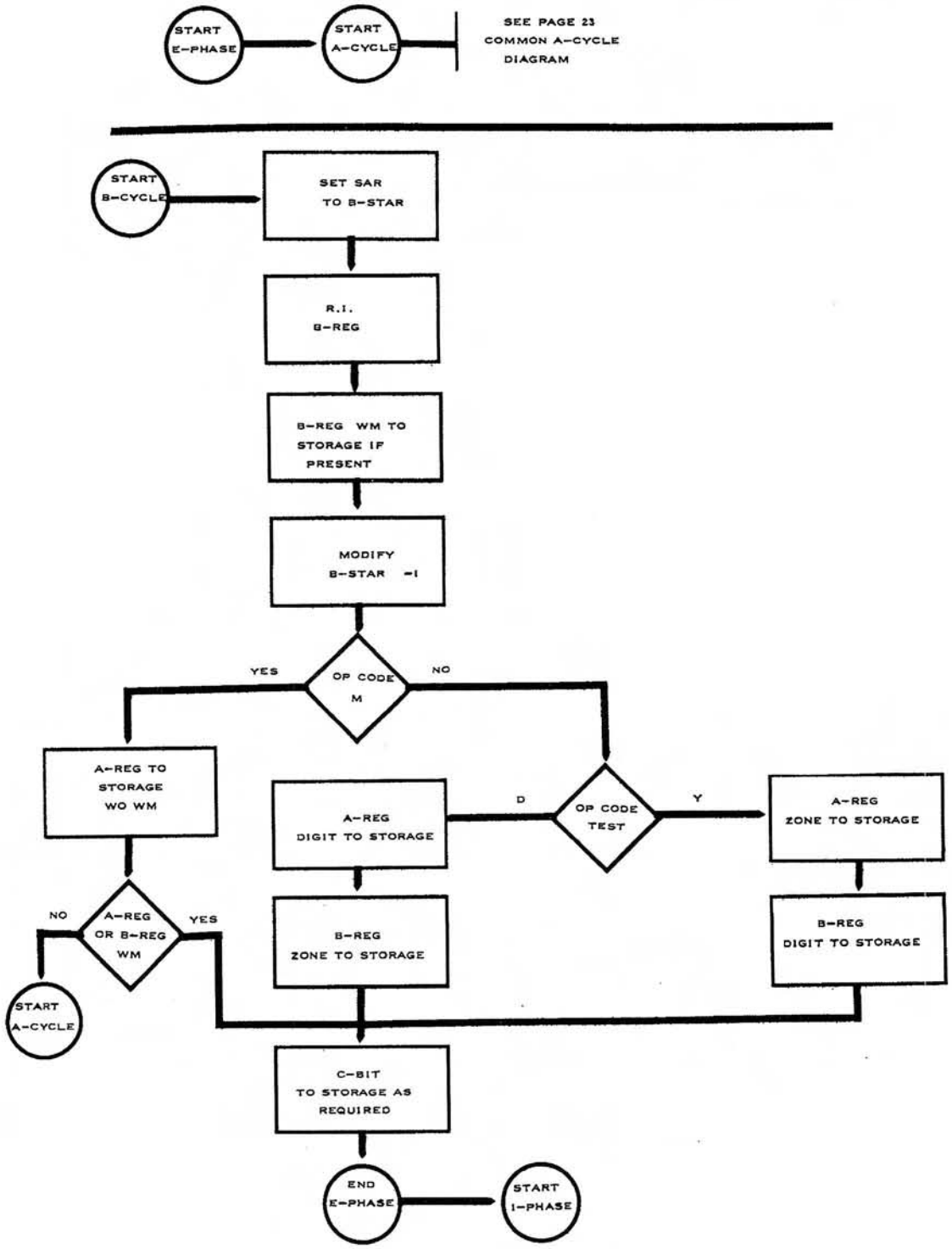
Example: M 135 850 M 315 M 468 (Assume all fields to be 5 character position fields.)

First Move Operation: M135 850
Locations 131 - 135 are moved to 846-850. At the end of the execute phase the B-STAR contains 845.

Second Move Operation: M315
Locations 311 to 315 are moved to 841-845. At the end of E-phase the B-STAR contains 840.

Third Move Operation: M468
Locations 464-468 are moved to 836-840.

MOVE . MOVE DIGIT AND MOVE ZONE INSTRUCTIONS DIAGRAM



Load

Load (L AAA BBB)

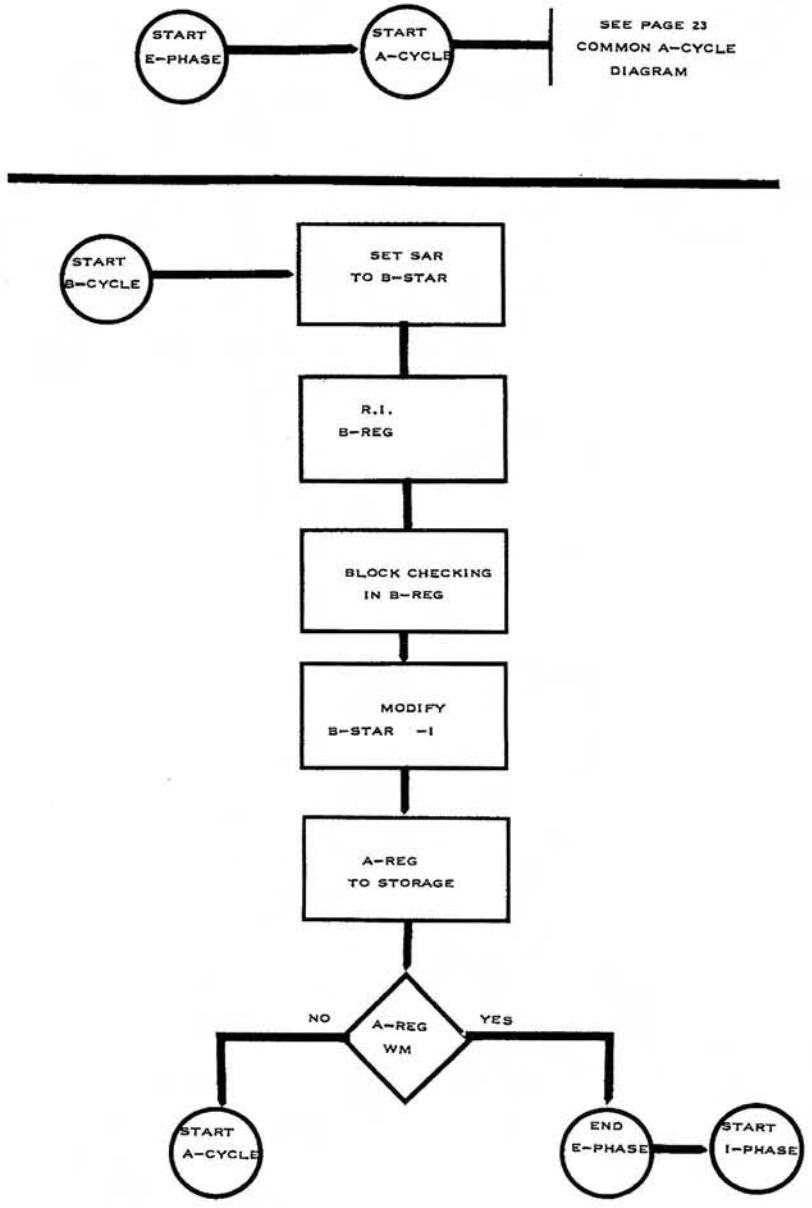
This operation is similar to the move instruction just described except the length of the word in the A-field must be defined by a word mark. The word-mark of the A-field is transferred to the B-field, and all B-field word marks up to this newly moved word mark are cleared. The A-field word mark stops the operation.

The load instruction is commonly used to load data into the printer or punch output areas of storage, or to move data or instructions from the card input area of storage to another location.

Load (L AAA)

This operation is the same as the Load operation just described. The B-STAR contains, and therefore the B-address is, the address left in the B-STAR upon completion of the last operation. This action is described in Move M AAA. This instruction cannot be used following a branch operation because the B-register is reset to blanks.

LOAD INSTRUCTION DIAGRAM



Clear, Clear and Branch

Clear (/ AAA)

This instruction clears all data and word marks from the A-address to the 00 address of that hundreds group in the storage unit. For example, if the A-address specified is 588, positions 588 "down" through 500 are cleared; if the A-address specified is 702, positions 702, 701 and 700 are cleared.

When E-phase begins, the A-address will be in both the A- and B-STARS. Because the clear operation consists of a series of B-cycles, all A-cycles are eliminated, and addressing is done from the B-STAR. On each cycle, normal transfer to the storage unit from the B-register is prevented. This action leaves all storage positions addressed set to blanks. Check bits are added to maintain odd bit parity.

The operation stops when the hundreds position of the B-STAR is reduced by one, indicating, for example, that the B-STAR address has changed from 500 to 499. Location 499 is not cleared because the address is actually reduced to 499 during the cycle that the storage address register contains 500.

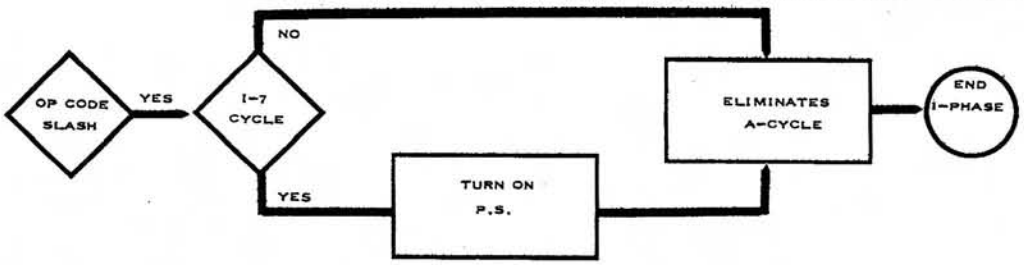
Clear and Branch (/ III BBB)

This operation is the same as the Clear operation just described with the following exception: Upon completion of the execute-phase of this operation, the next instruction is taken from the address specified in the A-STAR (Contains III) instead of the I-STAR.

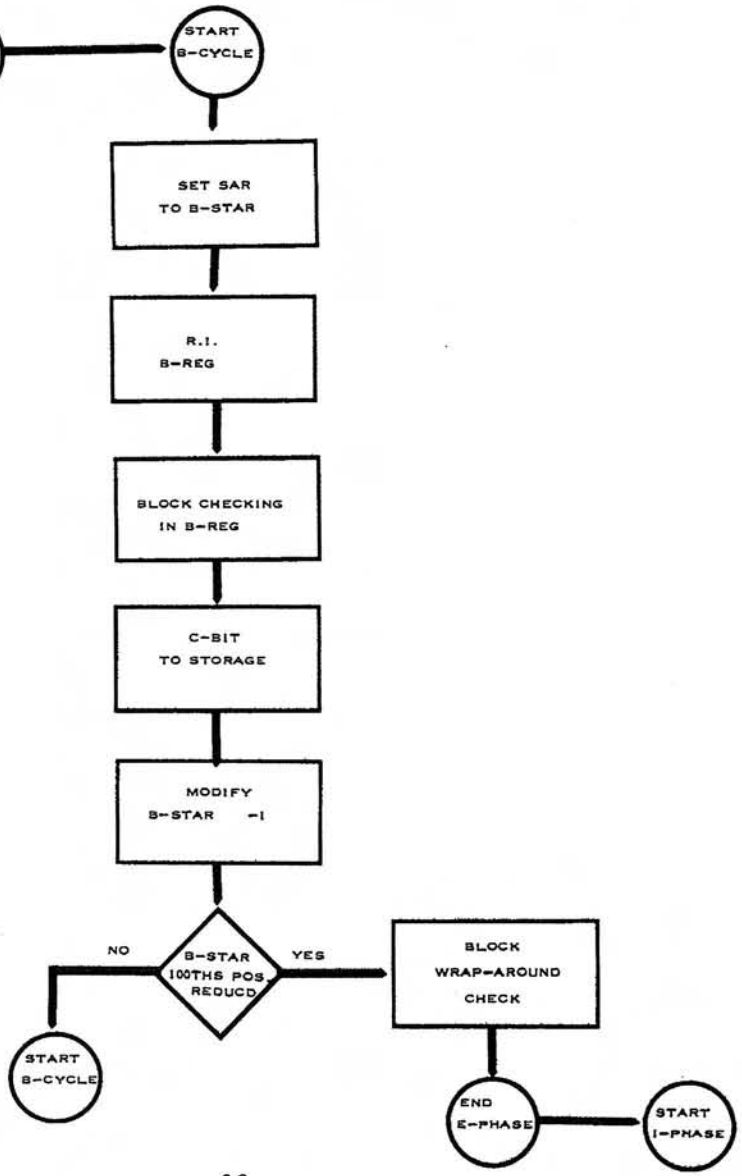
On normal clear operations (no branch) the I-phase is completed at 1-4 time when the word mark of the next op code is detected. On clear and branch operations, the I-phase is completed after 1-7 time.

Whenever the system detects the coincidence of 1-7 time and a clear operation, circuits are set up to read into the storage address register from the A-STAR instead of the I-STAR on the next I-phase. The actual transfer takes place in the next I-phase, after the execute-phase of the clear operation is completed. Once the new address (A-STAR) is in the storage address register, it is increased by one and then gated into the I-STAR. The remaining I-cycle addressing is accomplished in the normal manner as previously described.

CLEAR STORAGE AND CLEAR STORAGE AND BRANCH INSTRUCTIONS DIAGRAM



DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE.



Compare

Compare (C AAA BBB)

This operation compares the A-field with the B-field. The result, either equal or unequal, is stored in a latch for future reference by a branch operation (B III s or B III /, branch equal compare, or, branch unequal compare).

The following conditions cause an unequal compare:

1. The B-field is longer than the A-field.
2. The bit combinations of the two fields are not the same.

The two fields must have the same bit configuration in the positions compared for an equal compare result. If the A-field is longer than the B-field, the comparison stops at the end of the B-field.

The execute-phase of compare operations consists of alternate A- and B-cycles. The following action takes place on the first A-cycle:

1. The A-STAR is gated into the storage address register.
2. This address is reduced by one and stored back into the A-STAR.
3. The units position of the A -field is read out of storage and into both the A- and B-registers.
4. It is also put back into the same storage location by transferring the content of the B-register back to storage.

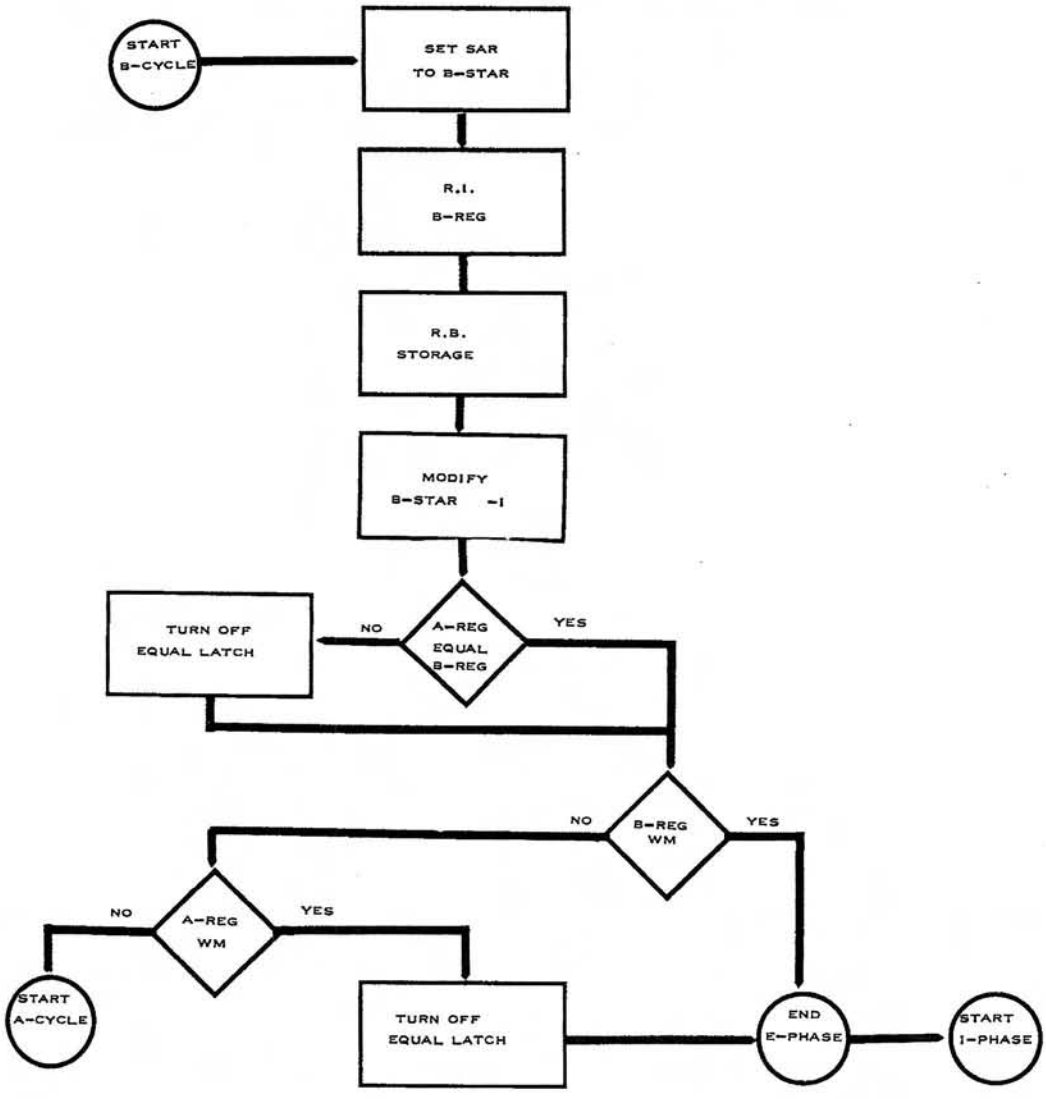
On the first B-cycle:

1. The B-STAR is gated into the storage address register.
2. This address is reduced by one and stored back into the B-STAR.
3. The units position of the B-field is read out of storage and into the B-register.
4. It is also transferred back into storage from the B-register.
5. The A- and B-register contents are compared. If they are equal, the equal compare latch is left on. Any unequal compare (units, tens, hundreds, positions, etc.) turns the latch off. Once it is turned off, it remains off to indicate an unequal comparison of the two fields.

COMPARE INSTRUCTION DIAGRAM

DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE.

SEE PAGE 23
COMMON A-CYCLE
DIAGRAM



Move Zero Suppress

Move Zero Suppress (Z AAA BBB)

This operation is the same as a Move operation with the following exceptions:

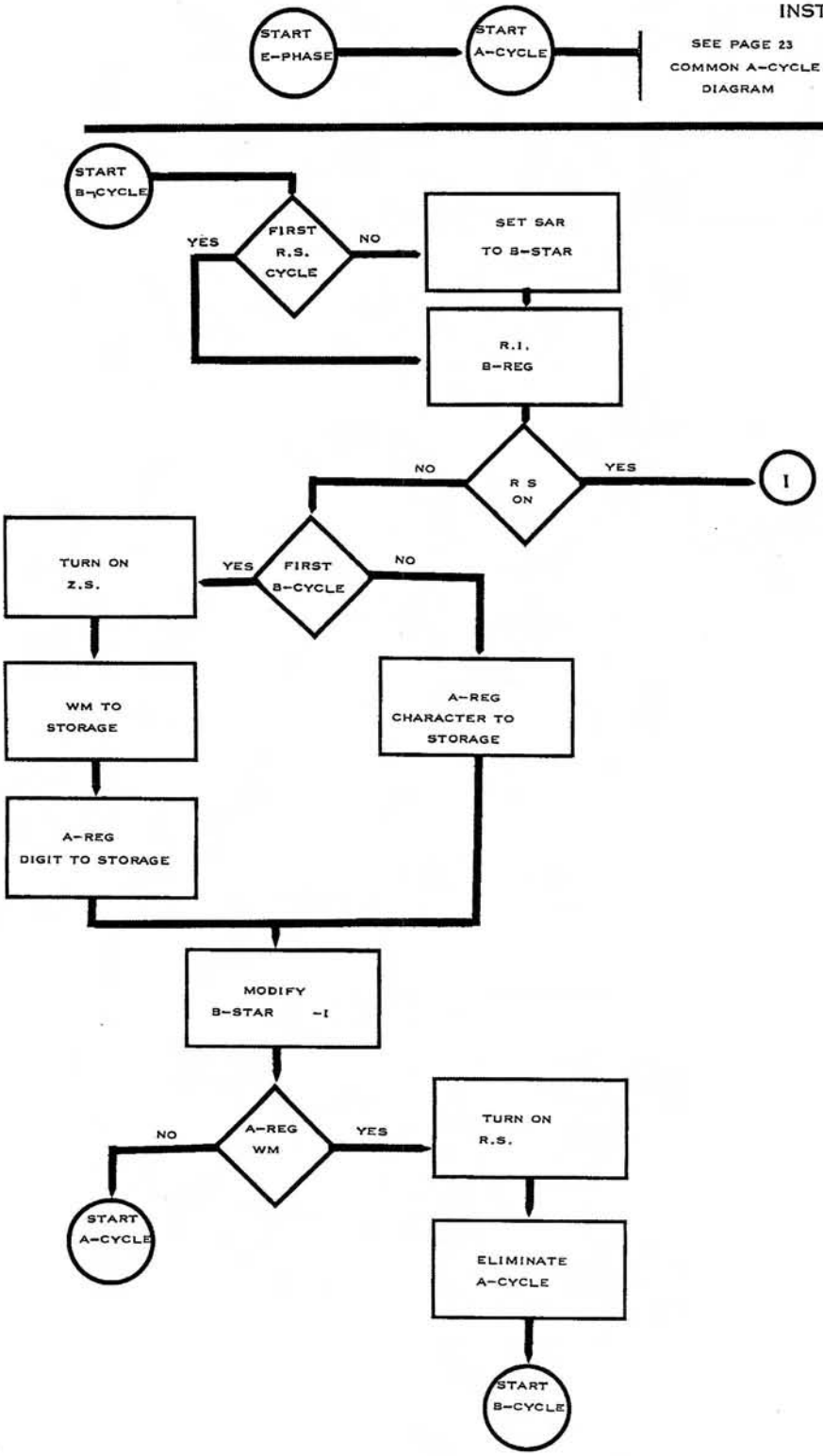
1. Zeros, dashes, and commas to the left of the highest order significant digit are eliminated.
2. Zone bits in the units position are eliminated.
3. The A-field must contain the defining word mark.

When a move-zero-suppress operation begins, the system sets a word-mark in the units position of the B-field. Zero-suppress-condition is also set on. Next, a copy of the entire A-field word is moved to the B-field, stopping when the A-field word mark is sensed. The digit portion only of the units position character is moved. Its zone is stripped.

Zero suppression now starts. The first cycle of zero suppression is one in which the B-field position just addressed is readdressed -- this time to eliminate any zero or punctuation present. This first zero-suppression cycle is called the first-edit-readdress cycle. During the first-edit readdress cycle, the B-address is prepared for reverse scanning by being modified with a +1. This causes addressing from the high-order position to the units position.

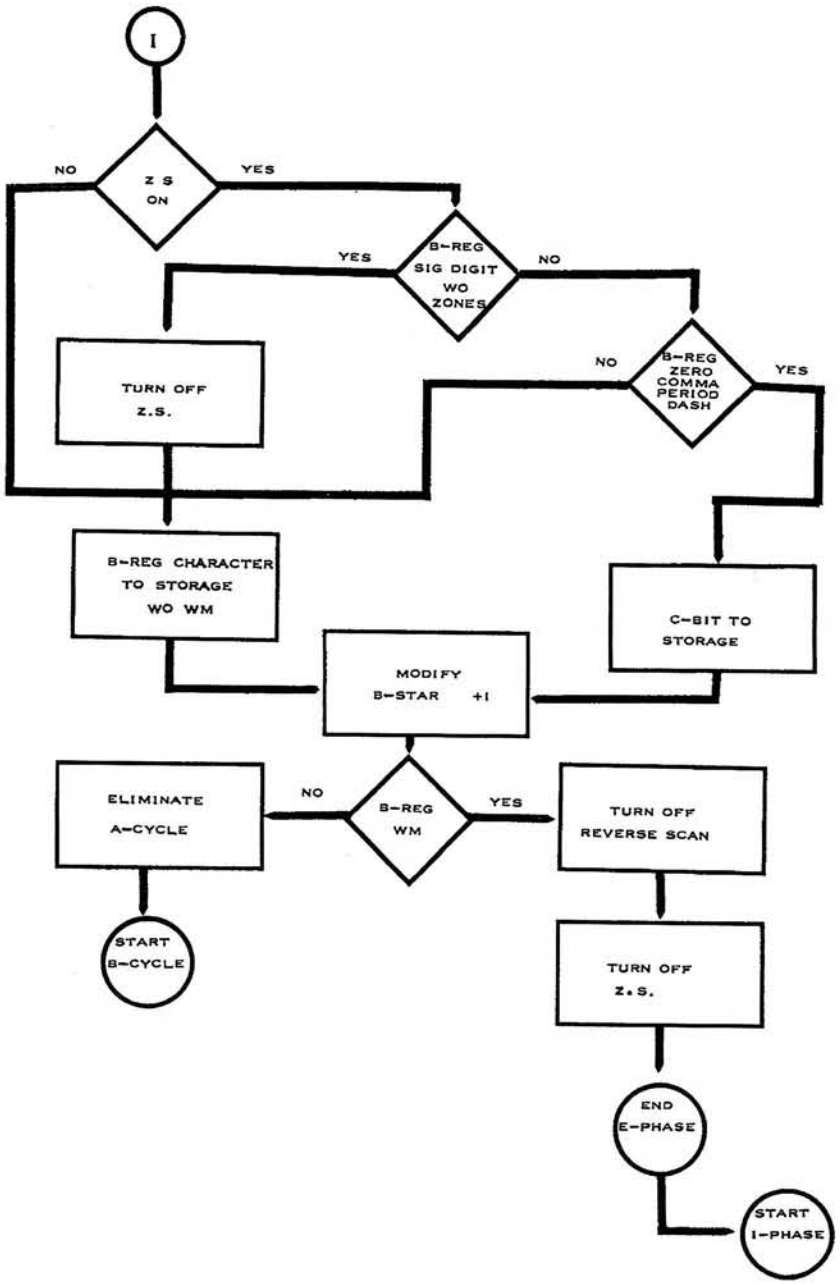
On the reverse scan, B-field characters are read into the B-register in a series of B-cycles; all A-cycles are eliminated. Zeros or punctuation (, . or -) are not transferred back into the B-field. When the first numeric digit appears in the B-register, the zero-suppress condition is set off allowing remaining zeros and punctuation to be transferred back to the B-field.

MOVE AND ZERO SUPPRESS
INSTRUCTION DIAGRAM



MOVE AND ZERO SUPPRESS

INSTRUCTION DIAGRAM CONTINUED



Edit Operations

Editing, as used in the 1401 system, is the modification of a data word with dollar sign, comma, decimal, credit (CR), minus (dash), asterisk, or other characters or symbols. When printed, the edited data word will have its characters separated by this punctuation. IBM 1401 editing also includes automatic suppression of zeros and punctuation to the left of the high-order significant digit.

There is only one edit instruction, E (A) (B). The A-field contains the original data word. The B-field contains a control word, consisting only of the desired punctuation symbols, separated by positions reserved for data characters. In a 1401 with standard editing ability, these "reserved" character positions may be blank, or may contain zeros. If a zero is used in the control word, zero suppression is automatically included in the edit operation.

When the edit operation is complete, the original data word remains undisturbed in the A-field. The resultant edited word appears in the B-field, containing the data characters "copied" from the A-field, interspersed with the original control word punctuation.

An edit operation not involving zero suppression stops when the B-field word mark is sensed in the B-register. Unlike most other operations, the B-field word mark is cleared on an edit operation. If an A-field word mark is encountered first, the operation continues, but all remaining B-field commas or zeros are set to blanks.

Edit with Basic Punctuation

An example of editing with the basic punctuation symbols follows. The control word in this example has been condensed for simplicity; a more realistic control word might be $bbb, bbb. bb**.

A-field (data word):	395
B-field before edit (control word):	$b, b, b. b**
B-field after edit:	$bb3, 9. 5**

The control word is addressed on B-cycles, because it is in the B-field. When a control word character should be retained in the resultant edited word, the B-register digit and zone is transferred back to the B-field. When a control word position reserved for a data character is encountered, the system must be prepared to send an A-field character to take its place. This is done by starting the edit operation with an A-cycle, in which an A-field character is stored in the A-register, where it "waits" for a blank position to be sensed in the B-field. Next, a B-cycle occurs, and a control word character is read into the B-register. If this character is retained in the resultant edited word, the B-register digit and zone is transferred back to the B-field, leaving the A-register character undisturbed. However, if the B-register contains a blank, the A-register is stored in the B-field.

Storage to the B-field always occurs on a B-cycle, no matter which register supplies the character to be stored. Any B-cycle, in which the B-register character (digit and zone) is transferred back to the B-field, is followed immediately by another B-cycle. The A-cycle, which normally would follow a B-cycle, is eliminated. This prevents another A-field character from being read into the A-register before the "waiting" character is used. An A-cycle is allowed only after an A-register character has been stored in the B-field.

Blank Space Insertion

The ampersand (&) is used in the control word to create blank spaces in the edited word, as follows:

A-field (data word):	395
B-field before edit (control word):	$bb, b. b&&*&*
B-field after edit:	$b3, 9. 5b b *b *
Results, if printed:	$ 3, 9. 5 * *

When sensed in the B-register on a B-cycle, the ampersand creates a blank space in the edited word by preventing both the B-register transfer to storage, and the A-register read into storage. Note that an ampersand in the A-register, originating from the A-field, is treated as any other character. For example:

A-field (data word):	&395
B-field before edit (control word):	$b b, b. b&&*&*
B-field after edit:	$&3, 9. 5b b *b *
Results, if printed:	$&3, 9. 5 * *

Extra Character Insertion

Any alphabetic, numeric, or special character (except those having special meaning in editing control) may be inserted in the body portion of the control word. For example:

A-field (data word):	012175
B-field before edit (control word):	bbb#&TYPE&C-R&&@&&$b. bb
B-field after edit:	012#b TYPEb C-Rb b@b b$1. 75
Results, if printed	012# TYPE C-R @ $1. 75

Credit or Minus Symbol

The letters C and R, or the minus (dash) symbol may be stored in the control word to the right of the low-order blank position to indicate on a printed report the sign of the data word involved. If to the right of the data positions, these characters will remain in the resultant edited word only if the sign of the data word is minus. (A minus data word has a B-bit over its units position in the A-field.) For example:

```
                                                          (-)
A-field (data word):                                    2 3 1 7 5
B-field before edit (control word):     bb&TYPE&C-R&&$b. bb&CR&*
B-field after edit:                     23b TYPEc C-Rb b$1. 75b CRb*
Results, if printed:                    23  TYPE  C-R   $1. 75  CR *
                              OR                          (+)
A-field (data word):                                    2 3 1 7 5
B-field before edit, (control word):    bb&TYPE&C-R&&$b. bb&CR&*
B-field after edit:                     23b TYPEb C-Rb b$1. 75b b b b*
Results, if printed:                    23  TYPE  C-R   $1. 75      *
```

In the preceding examples, note that the CR to the right of the data was transferred to the B-field, but conditional upon the sign of the data word; while the C-R of TYPE C-R was transferred unconditionally to the B-field. This control is attained by using the body and status conditions similar to that of the move and zero-suppress operation discussed in a previous section. All portions to the left and to the right of the data characters are considered the status portions. The system begins an edit operation in the status condition, and will not change to the body condition until the first A-register (data) character is stored in the B-field. The system again changes to the status condition if the last A-register (data) character is encountered before the end of the control word is reached. The A-field word-mark identifies the last A-field character.

Thus, C, R, and - are transferred unconditionally back to storage if they are in the body portion of the control word, and are transferred conditionally upon the sign of the A-register if they are in the status portion of the control word.

Edit with Zero Suppression

The basic punctuation process occurs as described above but in addition, zero suppression automatically changes zeros, commas, decimals, or dashes to blanks in all positions to the left of the highest-order significant digit in the edited word. The programmer calls for zero suppression by storing a zero, instead of a blank, in the control word. The position in which this control word zero is stored defines the extreme right limit of zero suppression. Zero suppression begins after a word is completely edited, and is accomplished by reverse scanning, in which the system "goes back" to eliminate zeros and punctuation:

```
A-field                                                 000501
B-field before edit, (control word):                    $b, bb0. bb*
B-field after edit, before zero suppression:            $0, 005. 01*
B-field after reverse scan (zero suppression):          $bbbb5. 01*
Results, if printed:                                    $     5. 01*
```

The control word zero, sensed during the normal forward scan, signals the system that zero suppression is to take place. A zero-suppress condition is set in the system at this time to be used later. Also, a word mark is set in the B-field position originally occupied by the control word zero to indicate the right-most limit of zero suppression. Normal editing continues until the high order B-field word mark is encountered. This word mark normally stops the edit operation, but because the zero suppress condition is on, the system is conditioned for reverse scanning.

All zero-suppression cycles are B-cycles. During the first cycle of zero suppression the B-field high-order position (just addressed) is re-addressed -- this time to eliminate any zero present. This first zero-suppression cycle is called the first-edit re-address cycle. During the first-edit re-address cycle, the B-address register is modified by a +1.

On the reverse scan, beginning with the first-edit re-address cycle, B-field characters are read into the B-register. Zeros or punctuation (, . or -) are not transferred to the B-field. When the first numeric digit appears in the B-register, the zero suppress condition is set off, allowing remaining zeros and punctuation to be regenerated to the B-field.

If the sequence of numeric digits is interrupted by an alphabetic or special character (other than, . or -), zero suppression is turned back on until another significant digit is encountered. When the first word mark is sensed on the reverse scan, the operation ends.

During reverse scanning in the following example, the zero-suppress condition is on from the high-order position of the edited word to the digit 1; off until the P appears in the B-register; on until the digit 3 appears in the B-register. The operation ends when the WM is sensed in the B-field.

		(-)
	A-field	0 0120035 60
B-field before edit (control word):		$bbb. bb&PLUS&$bb0. bb&-*
B-field after edit, before reverse scan:		$001. 20b PLUSb $035. 60b -*
B-field after reverse scan:		$bb1. 20b PLUSb $035. 60b -*
Results, if printed:		$ 1.20 PLUS $ 35.60 -*

Summary of Control Word Characters

b(blank) Replaced with the character from the corresponding position of the A-field.

0 (zero) Used for zero suppression. It is replaced with a corresponding character from the A-field. It also sets the zero-suppress condition on for later use, and causes a word mark to be set in the B-field to define the extreme right limit of zero suppression.

. (decimal) Undisturbed in the punctuated B-field after edit. It is eliminated on a zero-suppress operation if to the left of the high-order significant digit.

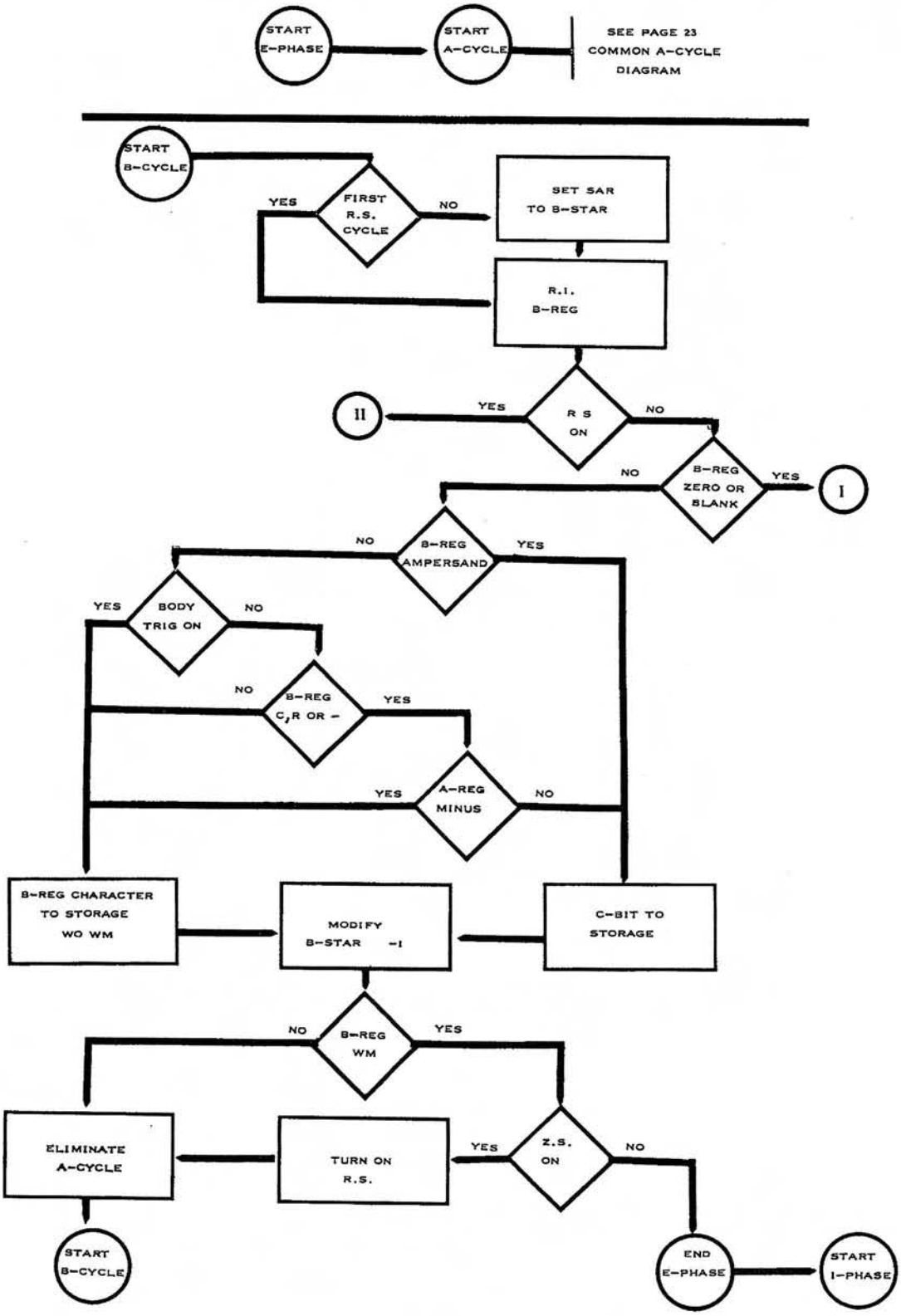
, (comma) Undisturbed in the punctuated B-field after edit. It is eliminated on a zero-suppress operation if to the left of the highest-order significant digit.

C and R (credit) If in the status portion of the control word, these characters are transferred from the B-register to the B-field when the A-field sign is minus. In the body portion of the control word these characters are transferred unconditionally.

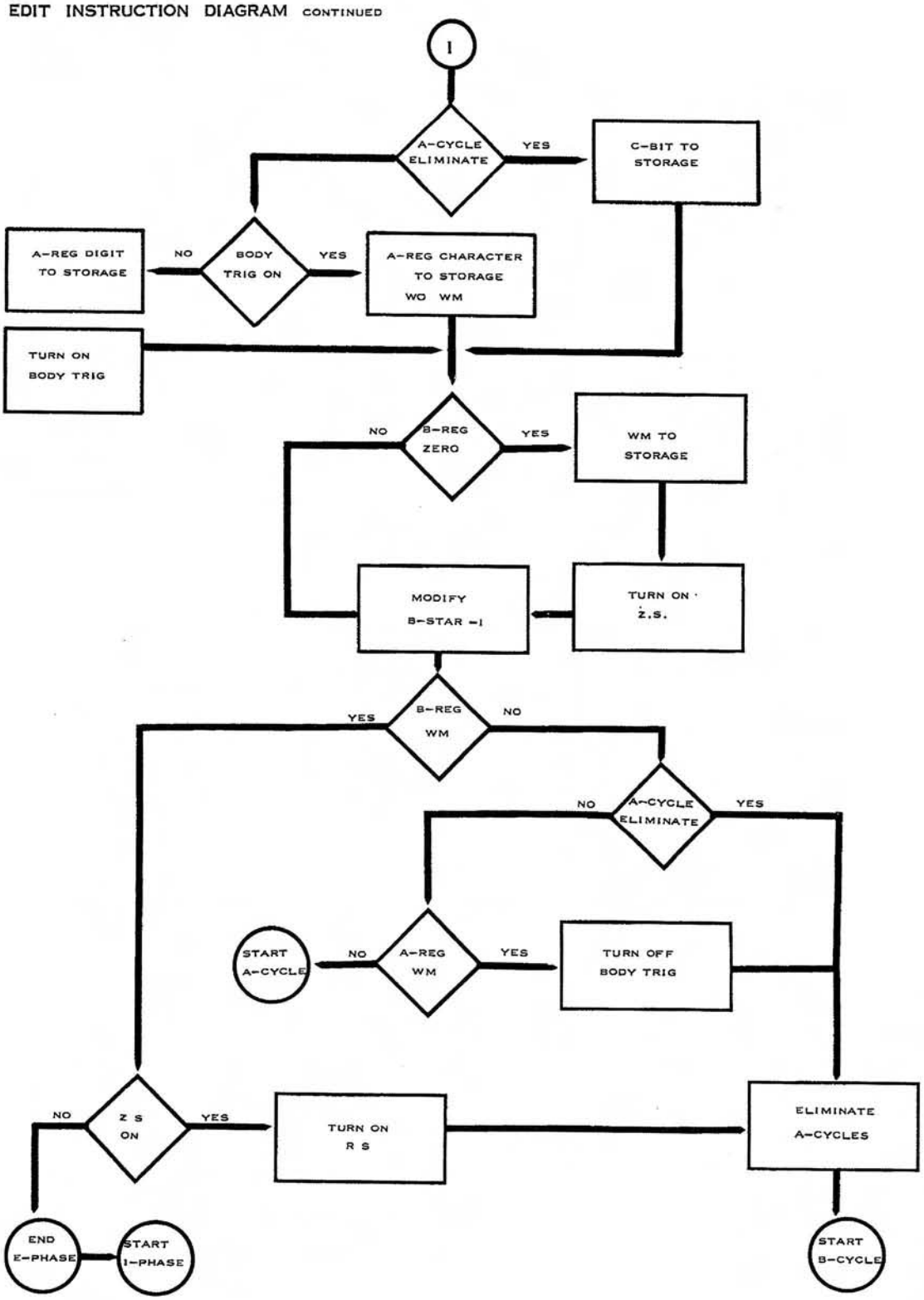
- (minus, or dash) Performs the same as C and R if it is in the status portion of the control word. This character in the body portion is transferred unconditionally. During zero suppression, it will be eliminated if to the left of the high-order significant digit.

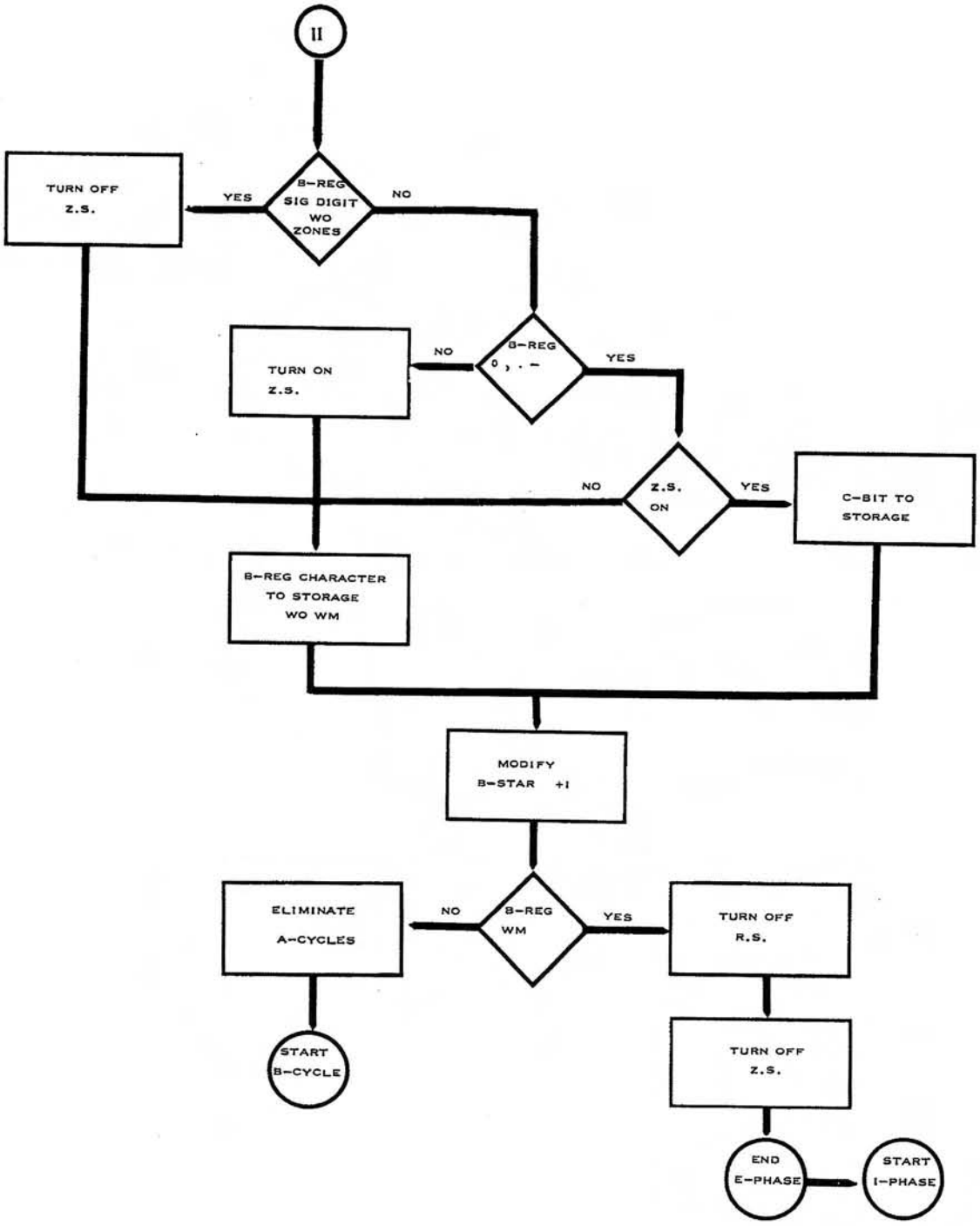
& (ampersand) Causes a blank space in the edited B-field.

EDIT INSTRUCTION DIAGRAM



EDIT INSTRUCTION DIAGRAM CONTINUED




II
B-REG SIG DIGIT WO ZONES
YES
NO
TURN OFF Z.S.
B-REG 0) . -
NO
YES
TURN ON Z.S.
Z.S. ON
NO
YES
C-BIT TO STORAGE
B-REG CHARACTER TO STORAGE WO WM
MODIFY B-STAR +1
B-REG WM
NO
YES
ELIMINATE A-CYCLES
TURN OFF R.S.
START B-CYCLE
TURN OFF Z.S.
END E-PHASE
START I-PHASE

Reset Add, Reset Subtract

Reset Add ($\underline{\overset{+}{0}}$ AAA BBB)

The reset-add operation reads the B-field out of storage (reset) and replaces it (add) with numeric data transferred from the A-field. If the A-field is shorter than the B-field, the data in the high order B-field positions is replaced by zeros. The resultant B-field sign is the same as the A-field sign, but is in standard form.

The qui-binary adder is not used on a reset add operation. The objective is accomplished by using the A-register, the B-register, and the zone-bit generator. On an A-cycle, the A-field data is read into the A-register and transferred back to the A-field. On the following B-cycle, the corresponding B-field position is read into the B-register, and the data in the A-register is transferred to the B-field. The zone-bit generator produces the resultant B-field sign in standard form on the first B-Cycle. The zone-bit generator uses the zone bit input from the A-register on the first B-cycle and the operation code, to determine whether to produce a standard plus or minus sign. If the A-field is shorter than the B-field, the A-register is reset and set to zero on each following B-cycle, beginning with the second B-cycle after the A-field WM. The A-field does not require a WM unless it is shorter than the B-field. The zero is transferred to the B-field, thus replacing the high-order B-field positions with zeros.

The A-field is not changed by the operation. The resultant B-field contains the same numeric data as the A-field, but contains zeros in the high-order positions if the B-field is longer. Zone bits are present in only the units position of the resultant B-field to indicate the sign. Zone bits in any other A-field positions are not transferred to the B-field.

Address modification of the A-STAR and B-STAR by minus one allows each field position to be addressed. Alternate A- and B-cycles continue, allowing A-field data to be transferred to the B-field. The operation ends when a WM is sensed in the B-register on a B-cycle.

Reset Add ($\underline{\overset{+}{0}}$ AAA)

The reset-add operation code is not commonly used with only an A-address. However, two functions may be accomplished by $\underline{\overset{+}{0}}$ AAA and are discussed here. One function is to set an A-bit and B-bit in the units position of a field which does not have any zone bits assigned. The units zone bits may be useful on certain program branch routines. However, this function can be accomplished in less processing time by use of a move-zone instruction.

The second function that may be accomplished is the stripping of zone bits from all field positions except the units. However, on arithmetic operations, zone bits are usually present in only the high-order field position (overflow) and the low-order field position (sign). Therefore, an individual position may be stripped of its zone in less processing time by other instruction codes.

A reset-add operation, with only the A-field specified is effectively $\underline{\overset{+}{0}}$ AAA AAA. The execution of the instruction is performed like a normal reset-add operation except that the A-field is accessed on both A & B cycles.

Reset Subtract ($\underline{\bar{0}}$ AAA BBB)

A reset-subtract operation with an A-field and B-field address is essentially the same as the reset-add operation ($\underline{\overset{+}{0}}$ AAA BBB) previously discussed. The operation objective is the same. The B-field data is read out of storage (reset) and replaced with data transferred from the A-field. If the A-field is shorter than the B-field, the high-order B-field positions are replaced with zeros. The resultant B-field sign is opposite to the A-field sign, and is in standard form.

The zone-bit generator produces the resultant sign in standard form on the first B-cycle. The qui-binary adder is not required on reset-subtract operations.

As explained in the reset-add operation, A- and B-cycles are used to accomplish the operation. The resultant B-field sign is produced on the first B-cycle by the zone-bit generator which uses the zone-bit input from the A-register and the operation code to determine whether to produce a standard plus or minus sign.

Address modification of each address register by minus one allows the 1401 to forward scan and address each A- and B-field position until a B-field WM is sensed. The operation ends on the B-cycle in which a WM is sensed in the B-register.

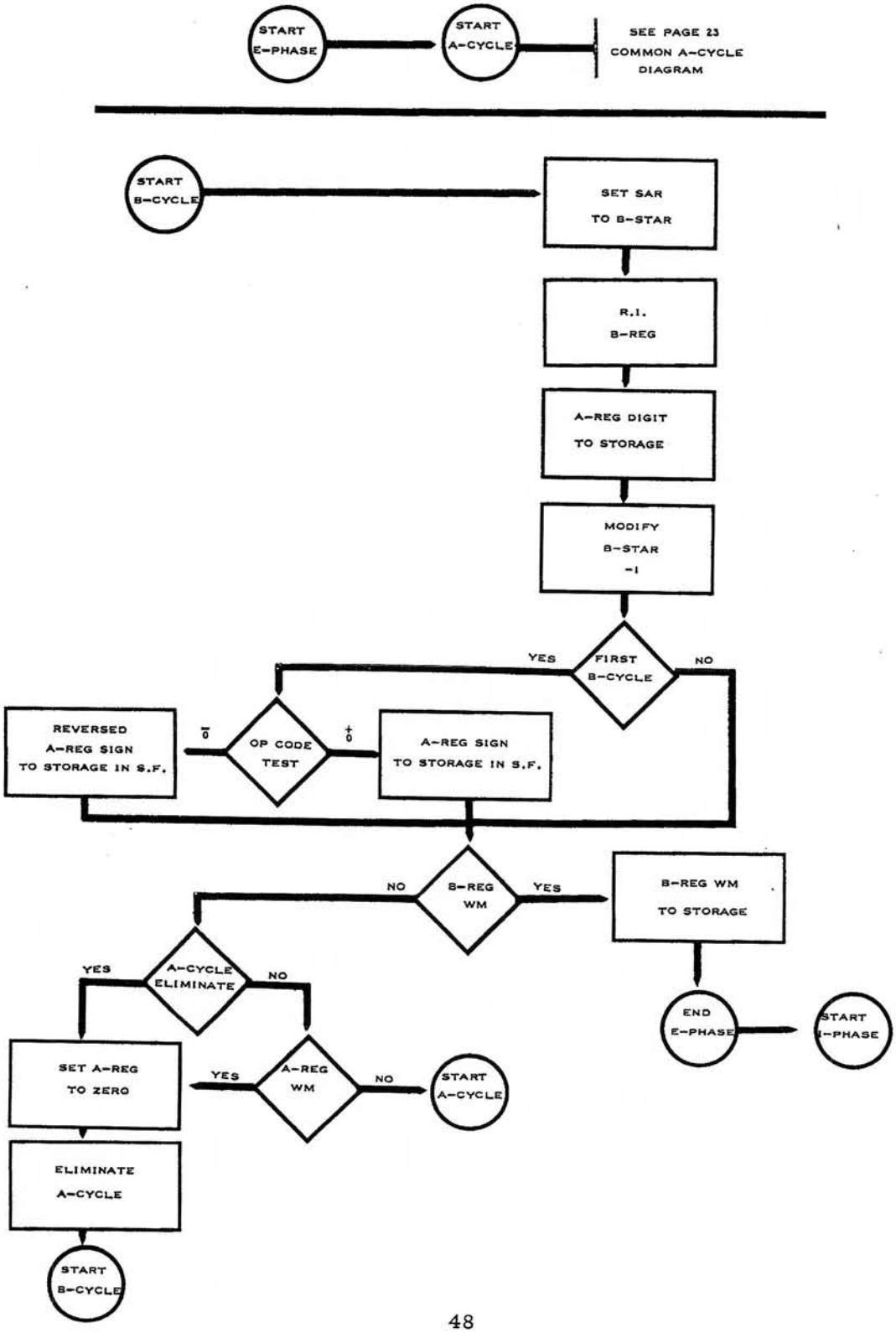
A short A-field requires an A-field WM. This causes the high-order positions of the B-field to be replaced with zeros.

Reset Subtract ($\underline{\bar{0}}$ AAA)

The reset-subtract operation code used with only an A-field can accomplish two functions. The most important function is to change the sign of the field addressed. It may also be used to strip zone bits from all except the units position of a field. The operation $\underline{\bar{0}}$ AAA is basically the same as $\underline{\overset{+}{0}}$ AAA previously discussed. Details of operation which are the same are not repeated here.

A reset-subtract operation, with only the A-field specified is effectively $\underline{\bar{0}}$ AAA AAA. The execution of this instruction is performed like a normal reset-subtract operation except that the A-field is accessed on both A & B cycles.

RESET ADD AND RESET SUBTRACT INSTRUCTIONS DIAGRAM



Add & Subtract

Of the standard arithmetic operation codes, only add and subtract codes require an adder. In the 1401, a qui-binary adder is used to perform either true-add or complement-add operations. It is a single-digit position adder which can combine two separate single-digit inputs with a carry or no-carry condition. The output is a single-digit position resultant which is stored in the resultant field (B-field), and an adder carry or no-adder carry which is stored in the arithmetic unit to be combined with the next digit position.

The digit inputs to the qui-binary adder come from the A-register and the B-register. Therefore, two fields may be added together by reading one field into the A-register and one field into the B-register, digit by digit beginning with the units position. The A-field enters the qui-binary adder from the A-register, and the B-field enters the qui-binary adder from the B-register.

The qui-binary adder output for each digit position is determined by combining the inputs from the A-register and B-register with a carry or no-carry condition. The digit output of the qui-binary adder is stored in the resultant B-field if the proper control gates are activated, and the adder carry or no-adder carry is stored in the arithmetic unit.

The qui-binary adder produces an output which is displayed on the 1401 console whenever there is information in the A-register and B-register. However, the output is not allowed to read into storage unless the proper control gates are activated by the operation controls.

Operations are performed in the qui-binary adder using a qui-binary code. The outputs of the A-register and B-register are in BCD code and must be translated to qui-binary code. An A-register translator and B-register translator are used to change the qui-binary adder inputs from BCD code to qui-binary code. An output translator is used to change the resultant of the qui-binary adder from qui-binary code back to BCD code.

Subtraction (add or subtract operation codes) is accomplished in the 1401 by complement-adding the input from the A-field and true-adding the input from the B-field. The input from the A-register is switched with true or complement controls to allow either a true or complement input to the qui-binary adder. The input from the B-register is always a true input. Because complement-add may occur on either add or subtract codes, the first A- and B-cycle of these codes must be used to analyze the operation code, the A-field sign, and the B-field sign to determine whether the operation should be a true-add (true input from A-register) or complement-add operation (complement input from A-register).

On true-add arithmetic operations, the sign of the resultant is the same as the B-field sign and in the same form. For complement-add operations, the resultant sign is produced by a zone-bit generator as determined by the operation conditions. The sign output of the zone-bit generator is always AB for a plus sign and B for a minus sign.

Zone bits in the high-order position of a field are normally used to indicate the number of overflows which have occurred in a field. An overflow is a carry out of the high-order position of the resultant B-field on only arithmetic true-add operations.

Overflows (zone bits) over the A-field digit may also be added to the high order (WM position) of the B-field on only true add operations and only on the cycle during which the B-field WM is detected. On reset add, reset subtract, and complement-add operations, overflow zone bits in the B-field are destroyed and high-order adder carries are not recognized as overflows. A-field overflow zone bits are not destroyed on any arithmetic operation (unless the A-STAR and B-STAR addresses are the same on subtract, and complement-add operations.)

Overflow bits may be used in programming in several ways. The 1401 can make logical decisions by interrogating the overflow condition (B III d, where Z is the d character) or interrogating the overflow bits (B III BBB d, where B is the d character). Address modification can also be accomplished, as a B-field containing 547 with an indication of three overflows (AB-bit) corresponds to a three character address of EA7 or actual address of 3547.

Because the qui-binary adder does not have the ability to add zone bits, a zone adder is required. The zone adder can combine A-register zone bits with B-register zone bits and an adder carry (overflow) or no-adder carry (no overflow) to produce a zone-adder output. The zone adder is activated and gated back to storage on only true-add operations, on the B-cycle in which the B-field WM is recognized. Therefore, overflow bits in the A-field may be true-added to overflow bits in the B-field, but only on the B-cycle during which the B-field WM is detected.

For example, an A-field with an indication of one overflow (A-bit) true added to B-field (of the same length) with an indication of two overflows (B-bit), produces a resultant B-field with an indication of three overflows (AB-bit). If an overflow had occurred as the fields were true-added, the resultant B-field overflow indication would be four or zero overflows (no overflow zone bits). The maximum indication of overflows is three, after which the indication sequence repeats. A-field or B-field word-marks are always transferred back to storage from the B-register on all standard arithmetic operations.

Add or subtract operation codes are performed in the 1401 by either a true-add or complement-add operation. To determine whether an operation should be a true-add or complement-add operation, the first A- and B-cycles of add or subtract codes are used to analyze the sign in the A-register (from the A-field), the sign in the B-register (from the B-field), and the operation sign (add code is plus and subtract code is minus).

If the number of minus signs is odd, the operation is a complement-add, and the true-complement controls are set to complement. This allows the input from the A-register to enter the qui-binary adder in complement form and develop a true resultant. The input from the B-register is always a true input. Figure 2 shows an analysis of possible combinations of codes and signs, and also shows the type of operation and the resultant sign.

A-REGISTER Zone bits	Sign	B-REGISTER Zone bits	Sign	Opr. Code	True or Compl Add	B-field > A-field Resultant Zone bits	B-field Sign	B-field < A-field Resultant Zone bits	B-field Sign
A . B	+	A. B	+	A	TA	A. B	+	A. B	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. B	+	A. $\bar{B}$	+	A	TA	A. $\bar{B}$	+	A. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. B	+	$\bar{A}$. $\bar{B}$	+	A	TA	$\bar{A}$. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. $\bar{B}$	+	A. B	+	A	TA	A. B	+	A. B	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. $\bar{B}$	+	A. $\bar{B}$	+	A	TA	A. $\bar{B}$	+	A. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+	A	TA	$\bar{A}$. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
$\bar{A}$. $\bar{B}$	+	A. B	+	A	TA	A. B	+	A. B	+
				S	CA	A. B	+	$\bar{A}$. B	-
$\bar{A}$. $\bar{B}$.	+	A. $\bar{B}$	+	A	TA	A. $\bar{B}$	+	A. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
$\bar{A}$. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+	A	TA	$\bar{A}$. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+
				S	CA	A. B	+	$\bar{A}$. B	-
A. B	+	$\bar{A}$. B	-	A	CA	$\bar{A}$. B	-	A. B	+
				S	TA	$\bar{A}$. B	-	$\bar{A}$. B	-
A. $\bar{B}$	+	$\bar{A}$. B	-	A	CA	$\bar{A}$. B	-	A. B	+
				S	TA	$\bar{A}$. B	-	$\bar{A}$. B	-
$\bar{A}$. $\bar{B}$	+	$\bar{A}$. B	-	A	CA	$\bar{A}$. B	-	A. B	+
				S	TA	$\bar{A}$. B	-	$\bar{A}$. B	-
$\bar{A}$. B	-	A. B	+	A	CA	A. B	+	$\bar{A}$. B	-
				S	TA	A. B	+	A. B	+
$\bar{A}$. B	-	A. $\bar{B}$	+	A	CA	A. B	+	$\bar{A}$. B	-
				S	TA	A. $\bar{B}$	+	A. $\bar{B}$	+
$\bar{A}$. B	-	$\bar{A}$. $\bar{B}$	+	A	CA	A. B	+	$\bar{A}$. B	-
				S	TA	$\bar{A}$. $\bar{B}$	+	$\bar{A}$. $\bar{B}$	+
$\bar{A}$. B	-	$\bar{A}$. B	-	A	TA	$\bar{A}$. B	-	$\bar{A}$. B	-
				S	CA	$\bar{A}$. B	-	A. B	+

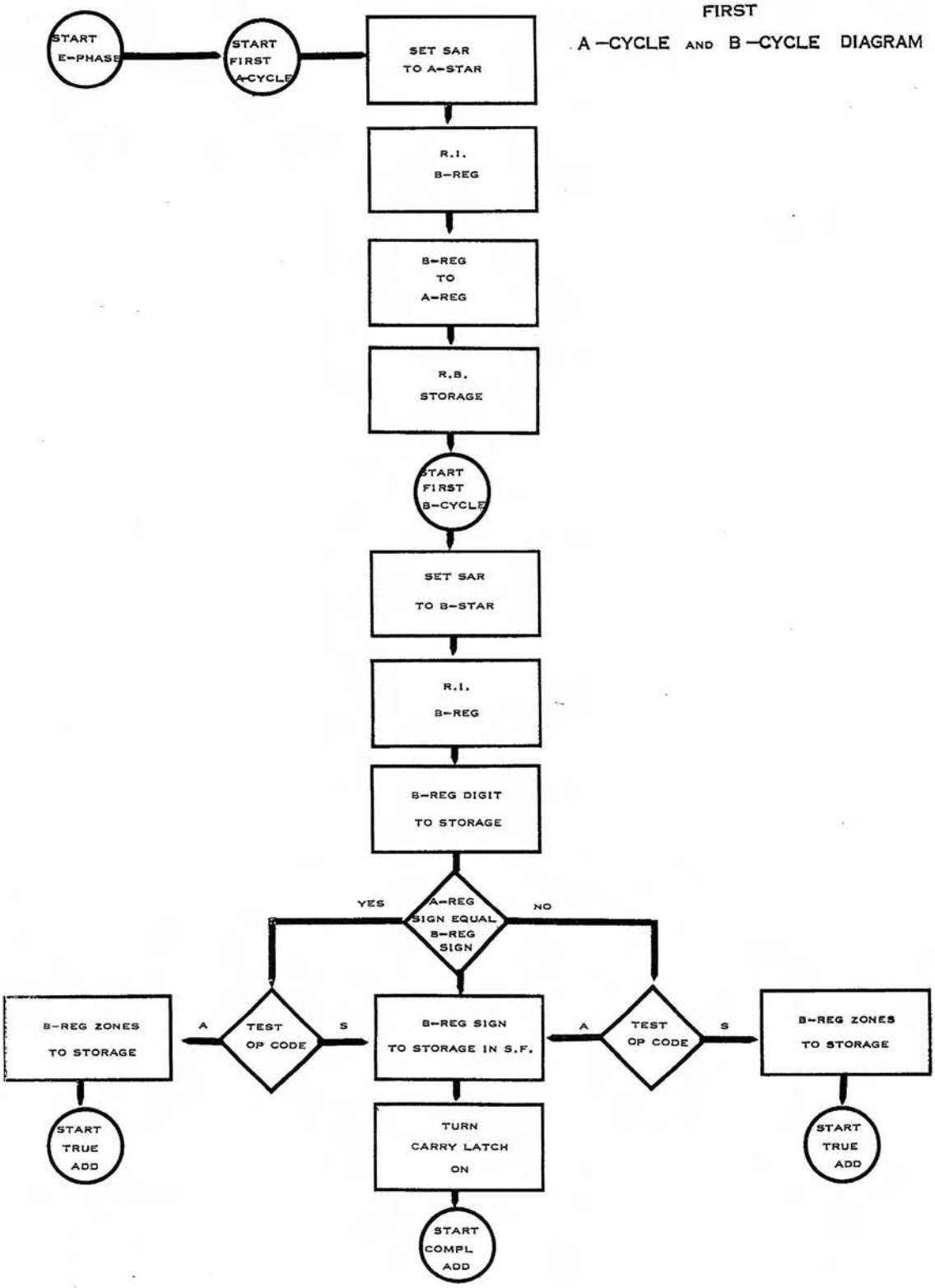
FIGURE 2 Add and Subtract Operation Codes -- Operation and Sign Analysis
(A dash over the zone indicates that the bit is not present)

Although a qui-binary adder output is available on the first A- and B-cycles of add or subtract codes, no adding actually occurs because the output is not allowed to go to storage. This requires that the units positions of the A- and B-fields be addressed a second time. Therefore, on the first A- and B-cycle, address modification is prevented so that the units addresses are in the A-STAR and B-STAR for the second A- and B-cycle. Minus one address modification begins on the second A-cycle.

ADD AND SUBTRACT INSTRUCTIONS

FIRST

A-CYCLE AND B-CYCLE DIAGRAM



True Add Operation (A AAA BBB) or (S AA BBB)

This operation begins with the second A-cycle of E-phase of either an add or subtract operation based on the decision during the first B-cycle just described.

On the second A-cycle, the units position of the A-field is again read into the B-register, transferred back to the A-field, and read into the A-register. On the second B-cycle, the units position of the B-field is read into the B-register. The sign (units zone bits) is transferred from the B-register to the units position of the B-field together with the numeric output of the qui-binary adder.

The qui-binary adder output on a B-cycle is the resultant developed by combining the true inputs from the A-register and B-register with a carry or no-carry condition. The input from the A-register is in true form for true-add operations. The input from each register is translated from BCD code to qui-binary code by the respective translators. The resultant is developed and validity checked in qui-binary code, and then translated back to BCD code by an output translator, and gated to the resultant B-field.

If the addition results in an adder carry, it is stored in the arithmetic unit to be combined with the inputs from the A- and B-register for the next digit position on the next B-cycle.

Alternate A- and B-cycles continue, and address modification by minus one allows each digit position to be addressed in turn. On each A-cycle, A-field data is read into the B-register, transferred back to the A-field, and read into the A-register. On each following B-cycle, the B-field data is read into the B-register. The contents of the A-register is combined in the qui-binary adder with the contents of the B-register and a carry or no-carry condition. A qui-binary adder output is developed which is gated to the B-field on each B-cycle to become part of the resultant B-field. Hence, the resultant B-field is a sum of the original A- and B-fields.

The A-field does not require a WM, unless it is shorter than the B-field. An A-field WM causes elimination of further A-cycles. Beginning with the second B-cycle after the A-field WM, the A-register is reset and set to zero. Therefore, on succeeding B-cycles, the B-field data is read into the B-register and combined in the qui-binary adder with the zero from the A-register and a carry or no-carry condition. The data in the high-order resultant B-field positions is the data in the original high-order B-field positions plus any carries that are developed. The true-add operation always ends on the B-cycle in which a WM is sensed in the B-register. The A-field is not changed by the operation.

B-field zone bits in other than the units (sign) position and word-mark (overflow) position are destroyed. The qui-binary adder cannot accept zone bits, and the zone adder is active on only the WM B-cycle. The resultant sign is the same as the original B-field sign and has the same form.

On the B-cycle in which the B-field WM is read into the B-register, the zone adder is active. A-register zone bits (from the A-field) are combined in the zone adder with B-register zone bits (from the B-field) and an adder or no-adder carry. The zone-adder output is gated to the resultant B-field on the WM B-cycle. Therefore, A-field overflow bits can be added to B-field overflow bits on true-add operations. The zone adder is not activated if the B-field WM is detected during the second B-cycle. In other words, the zone bits are not added in a one digit add operation.

For example, if a short A-field with an indication of two overflows (B-bit) is true-added to a B-field with an overflow (adder carry) occurs, the resultant B-field contains an indication of only one overflow (A-bit). The A-field overflow zone bits are not added, so only the overflow that occurred on the true-add operation is indicated.

If both fields are the same length, then an A-field with an indication of one overflow (A-bit), true-added to a B-field with an indication of two overflows (B-bit) produces a resultant B-field with an indication of three overflows (AB-bit). However, if an overflow occurs on the true-add operation, the resultant B-field then contains an indication of four or zero overflows (no zone bits). The A-field overflow bits are not destroyed on true-add operations (A AAA BBB).

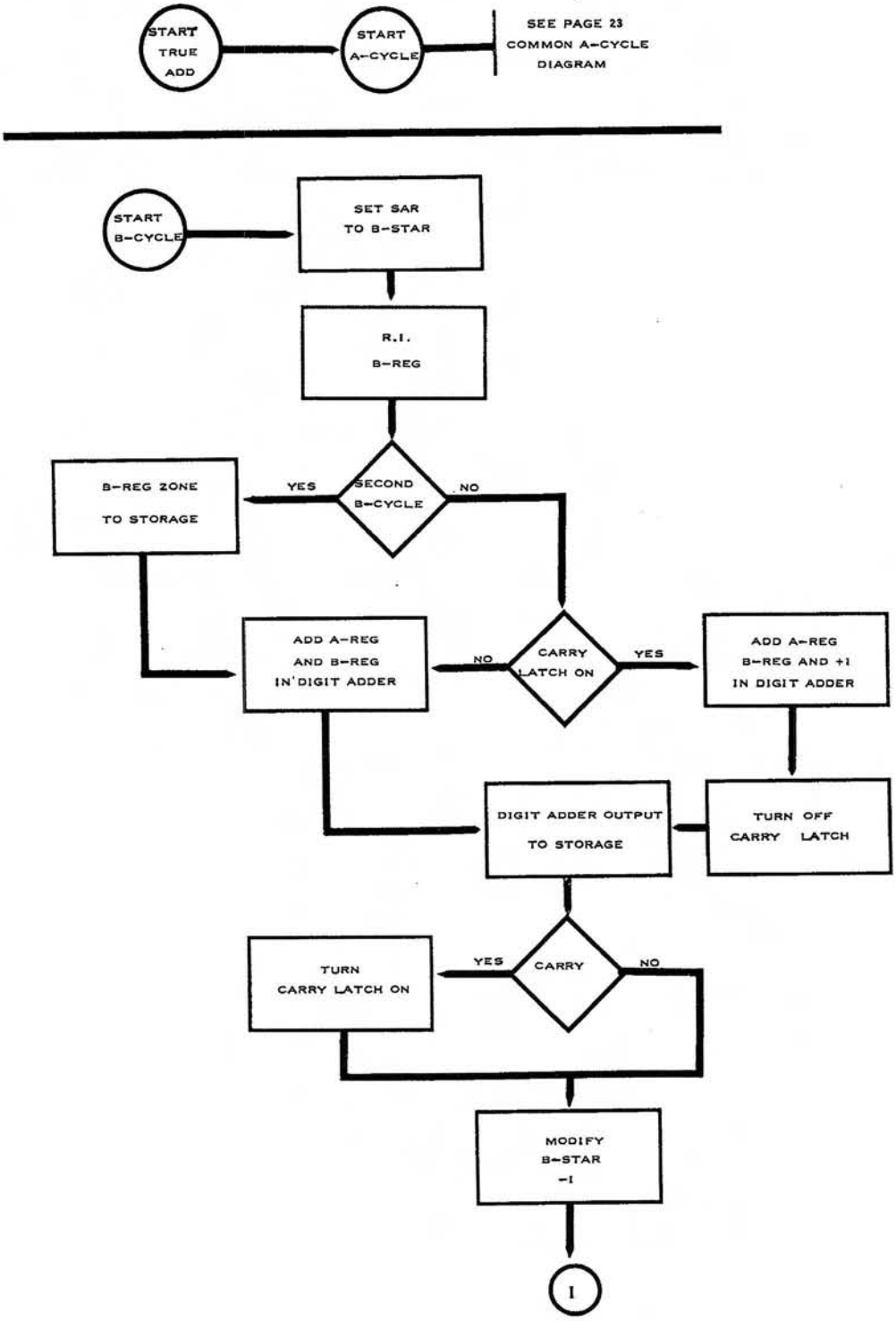
True-Add Operations (A AAA)

An add operation code with only the A-field specified adds the A-field data to itself. A AAA is always a true-add operation, and the effect of the operation is to double the A-field. The resultant A-field sign is the same as the original A-field sign and has the same form. Overflow bits are added so that the resultant A-field overflow indication is doubled. Details of operation which have been previously explained for true-add operations will be mentioned but not repeated in detail in this section.

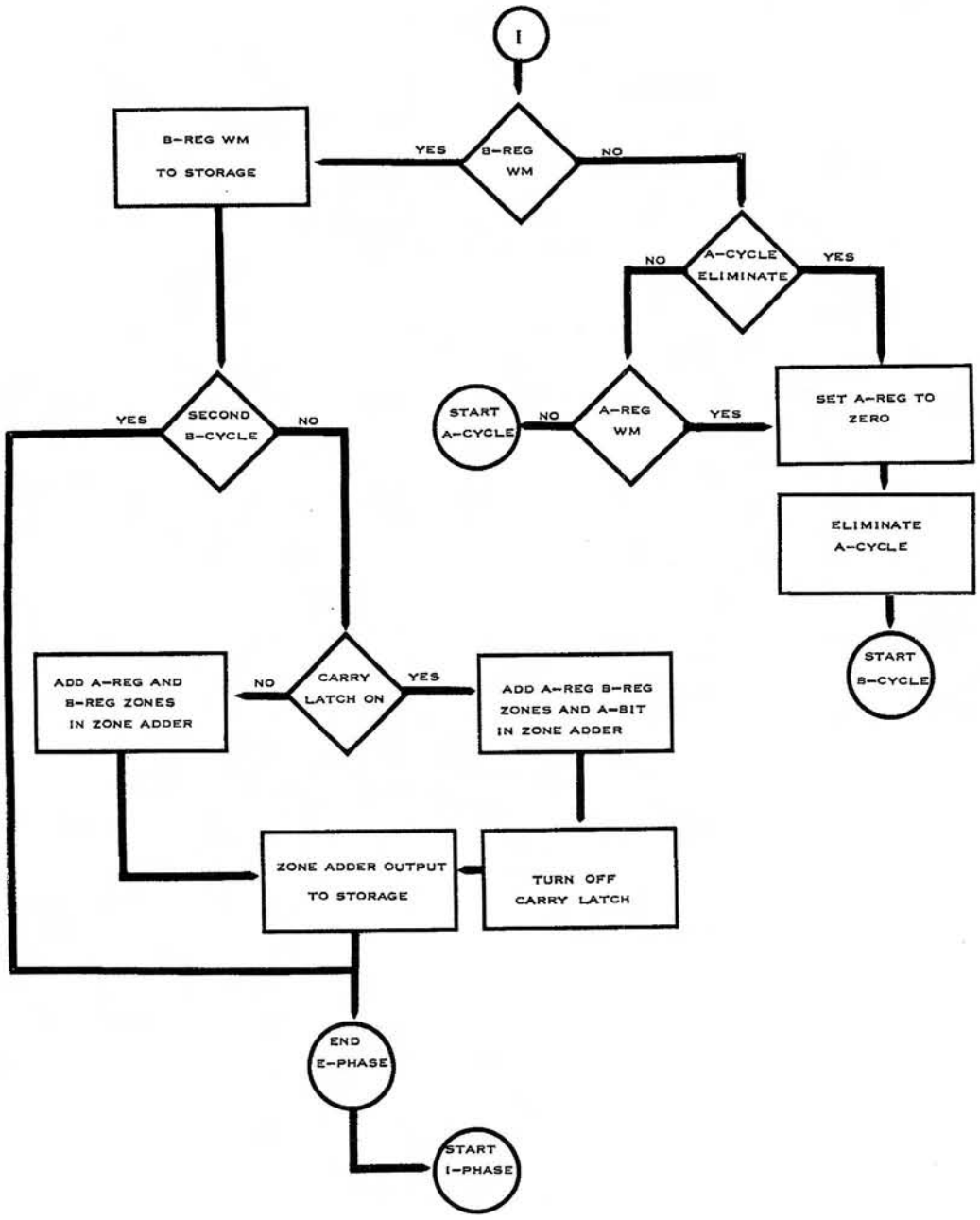
With only the A-field specified, the effective instruction is A AAA AAA. Alternate A- and B-cycles are used to accomplish the operation. Address modification of each address register is by minus one, and each A-field position is addressed twice because the A-field address is in both the A-STAR and B-STAR.

The operation A AAA is basically the same as A AAA BBB just discussed under true-add operations. The difference is that the information from storage always comes from the same field. Therefore, each A-field position is added to itself, and the resultant A-field is twice the original A-field. Otherwise, the cycle-by-cycle details of operation are the same as previously described under true-add operations.

TRUE ADD OPERATION DIAGRAM



TRUE ADD OPERATION DIAGRAM CONTINUED



Complement Add Operations (A AAA BBB) or (S AAA BBB)

Add or subtract codes are performed in the 1401 by either a true -add or complement-add operation as discussed previously.

On the first A- and B-cycle of any complement-add operation, the zone-bit generator produces a first resultant B-field sign, which is the same as the original B-field sign, but is in standard form. If the first resultant B-field developed is in true-form, the first resultant B-field sign is the final resultant B-field sign and the complement-add operation (no recomplement) ends.

If the first resultant B-field developed is in complement form, a recomplement operation is initiated to convert the complement result to true form and change the first resultant B-field sign to a final resultant B-field sign.

On the first B-cycle of the second complement-add (recomplement operation), the zone-bit generator produces a sign which is opposite to the first resultant B-field sign, and also in standard form. Figure 3 shows the resultant B-field signs for complement-add operations. (See Page 62)

On complement-add operations, the input from the A-register enters the quibinary adder in complement form and is combined with the true input from the B-register and a carry or no-carry condition.

Because the complement input from the A-register is a 9's complement, provision is made to have the carry condition on (in the arithmetic unit) when the units positions are combined. The units carry condition must be forced, on the first B-cycle of complement-add operation to develop the correct resultant. For example:

B-reg contains + 65;	
true input from B-reg	65
A-reg contains -50;	
compl input from A-reg	49
Carry condition on	1
	15 and adder carry

NOTE: The high-order adder carry is not recognized as an overflow on complement-add operations.

Any complement-add operation may also require a recomplement operation. If the absolute value (disregard sign) of the B-field is greater than the absolute value of the A-field, a high-order carry occurs. This indicates that the first resultant B-field is the final resultant B-field in true form, and no recomplement is required.

If the absolute value of the B-field is less than the absolute value of the A-field, no high-order adder carry occurs. This indicates that the first resultant B-field is in complement form, and a recomplement operation is necessary to convert the first resultant B-field to a final resultant B-field in true form, and change the sign.

A complement-add operation subtracts algebraically the A-field data from the B-field data. The resultant is developed in the qui-binary adder and stored in the resultant B-field in true form. On complement-add operations (no recomplement), the resultant B-field sign is the same as the original B-field sign and is in standard form.

On a complement-add operation (with recomplement), the first resultant B-field sign is also the same as the B-field sign but is in standard form. During the recomplement operation, the sign is changed, and the final resultant B-field sign is opposite to the first resultant B-field sign and is also in standard form.

On the second A-cycle, minus one address modification of the A-STAR and B-STAR begins. The units position of the A-field is read into the B-register, transferred back to the units position of the A-field, and read into the A-register. On the second B-cycle, the units position of the B-field, including the sign bits produced by the zone-bit generator, is read into the B-register. The sign in the B-register is transferred back to the units position of the B-field, together with the output from the qui-binary adder.

The qui-binary adder output in the second B-cycle is the result of combining the true input from the B-register with the complement input from the A-register and a carry condition, which was forced on the first B-cycle.

For all other positions, the carry or no-carry condition is set by an adder carry or no-adder carry. If an adder carry occurs in any position, the arithmetic unit stores the carry and combines it with the A-register and B-register inputs from the next digit position on the following B-cycle.

Alternate A- and B-cycles are continued, using minus one address modification of each address register to allow each field position to be addressed in turn on a forward scan.

On each following A-cycle, an A-field position is read into the B-register, transferred back to the A-field, and read into the A-register. On each following B-cycle, the corresponding B-field position is read into the B-register. The output developed in the qui-binary adder is gated to the resultant B-field.

Zone bits are not allowed to be transferred back to the B-field except in the sign position on complement-add operations. Zone bits in the other B-field positions are lost because the zone adder is not activated on any arithmetic operation except true-add. A-field zone bits are not changed.

The operation continues until a B-field WM is sensed. On complement-add operations, an adder carry in the B-cycle in which the B-register WM is sensed, indicates that recomplementing is not required (the absolute value of the B-field was larger than the absolute value of the A-field), and the operation ends.

A carry from the high-order position will always occur on complement-add operations (no recomplement). Therefore, overflows are not meaningful on complement-add, and high order adder carries are not recognized as overflows. For example, a B-field with an indication of two overflows (B-bit) before any complement-add operation, will have an indication of zero overflows (no zone bits) when the operation is complete. A short A-field requires a WM. As discussed in other arithmetic operations, the A-field WM causes elimination of further A-cycles. Beginning with the second B-cycle after the A-field WM, the A-register is reset and set to zero. Therefore, as each high-order B-field position is read into the B-register, it is combined in the qui-binary adder with a nine (9's complement to zero) from the A-register and a carry or no-carry condition. The complement-add operation (no recomplement) ends on the cycle that the B-field WM is read into the B-register.

If on the B-cycle in which the B-register WM is sensed there is no adder carry the complement-add operation does not end. A no adder carry at this time indicates that the absolute value of the B-field is less than the absolute value of the A-field.

Therefore, the first resultant B-field is in complement form and must be changed back to true form by recomplementing. The normal E-phase to I-phase change is prevented, and a reverse scan takes place prior to the actual recomplement operation.

The recomplement operation, as the name implies, is actually a second complement-add operation which converts the first resultant B-field to true form, and also changes the first resultant B-field sign to the final resultant B-field sign.

After the first complement-add operation is complete, the A- and B-STARS contain the addresses of the field positions to the left of the high-order A- and B-field positions. Because the A-field data is not required in the recomplement operation, A-cycles are eliminated for the remainder of the operation, and the A-STAR address remains the same. However, it is necessary to change the address in the B-STAR back to the units position address of the first resultant B-field to perform the second complement-add. This is done in the basic 1401 by taking a series of B-cycles in reverse scan, using plus one address modification.

During the reverse scan, the B-field data is read out of storage and transferred back to the B-field without being changed. The reverse scan continues until the units position of the B-field is indicated by a B-bit set in the beginning of the complement-add operation (AB-bit for + or B-bit for -).

To convert the first resultant B-field to true form, each first resultant B-field position is addressed twice on the second forward scan. The first time each B-field position is addressed, the data is read into the B-register and A-register, and a zero is read back into the B-field position to replace the data just read out. The second time the same B-field position is addressed (B-readdress-cycle), the zero is read into the B-register. The complement input from the A-register and the true zero input from the B-register is combined in the qui-binary adder with a carry or no-carry condition. The qui-binary adder output in true form is gated back to storage on the B-readdress-cycle.

On the B-readdress cycle that the units position of the first resultant B-field is converted to true form (controlled as first B-cycle of second forward scan), the carry condition must be on to develop the correct qui-binary adder output. The carry condition is forced at the end of the reverse scan. For all other positions, the adder carries in the qui-binary adder set the carry condition. The B-register sign is also changed on the same B-readdress cycle (controlled as first B-cycle of second forward scan).

B-readdress cycles are accomplished by preventing the reset and set of the storage address register. Address modification is not prevented, so that the modified address is present in the B-STAR, but is not read into the storage address register. Therefore, on the following B-cycle (B-readdress cycle) the same position of core storage is read out because the storage address register contents are the same.

B-readdress cycles are used on the recomplement operation as explained, and are also used to change from forward scan (minus one address modification) to reverse scan (plus one address modification); and from reverse scan to forward scan. A problem exists in changing address modification of the B-field from minus one to plus one, or vice versa. For example, if the 1401 is in forward scan, the reverse scan signal occurs late in the last forward B-cycle. Therefore, address modification by minus one has already been initiated. However, the storage address register can be preventing from resetting. Therefore, the following cycle is a B-readdress cycle in which plus one address modification begins. This allows the scan to advance in a reverse direction. The following cycles are B-cycles with plus one address modification which allow the 1401 to continue scanning in a reverse direction until a forward scan signal occurs.

The actual recomplement operation occurs on the second forward scan, and the complement-add operation (with recomplement) ends when the B-field WM is sensed in the B-register on the last B-readdress cycle of the second forward scan.

Units Position of	Zone bits	Sign	Zone bits	Sign	Zone bits	Sign	Zone bits	Sign
Original B-field	$\bar{A} \cdot \bar{B}$	+	$A \cdot \bar{B}$	+	$A \cdot B$	+	$\bar{A} \cdot B$	-
1st Resultant B-field (All compl-add operations) or Final Resultant B-field (Compl-add-no recompl)	$A \cdot B$	+	$A \cdot B$	+	$A \cdot B$	+	$\bar{A} \cdot B$	-
Final Resultant B-field (Compl-add with recomp)	$\bar{A} \cdot B$	-	$\bar{A} \cdot B$	-	$\bar{A} \cdot B$	-	$A \cdot B$	+

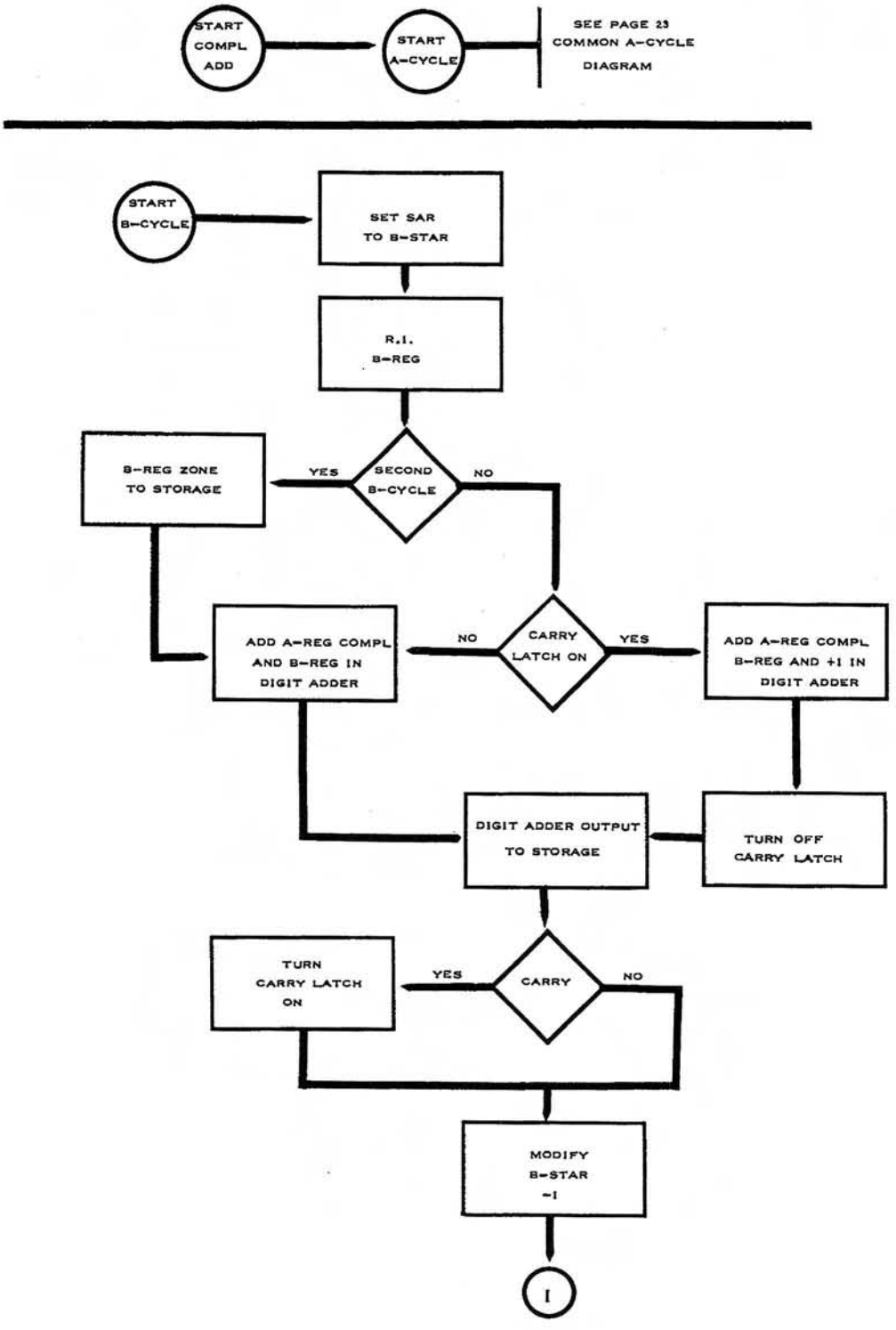
FIGURE 3 Resultant B-field signs for Complement-Add Operation
(A dash over the zone bit indicates that the bit is not present)

Complement Add Operations (S AAA)

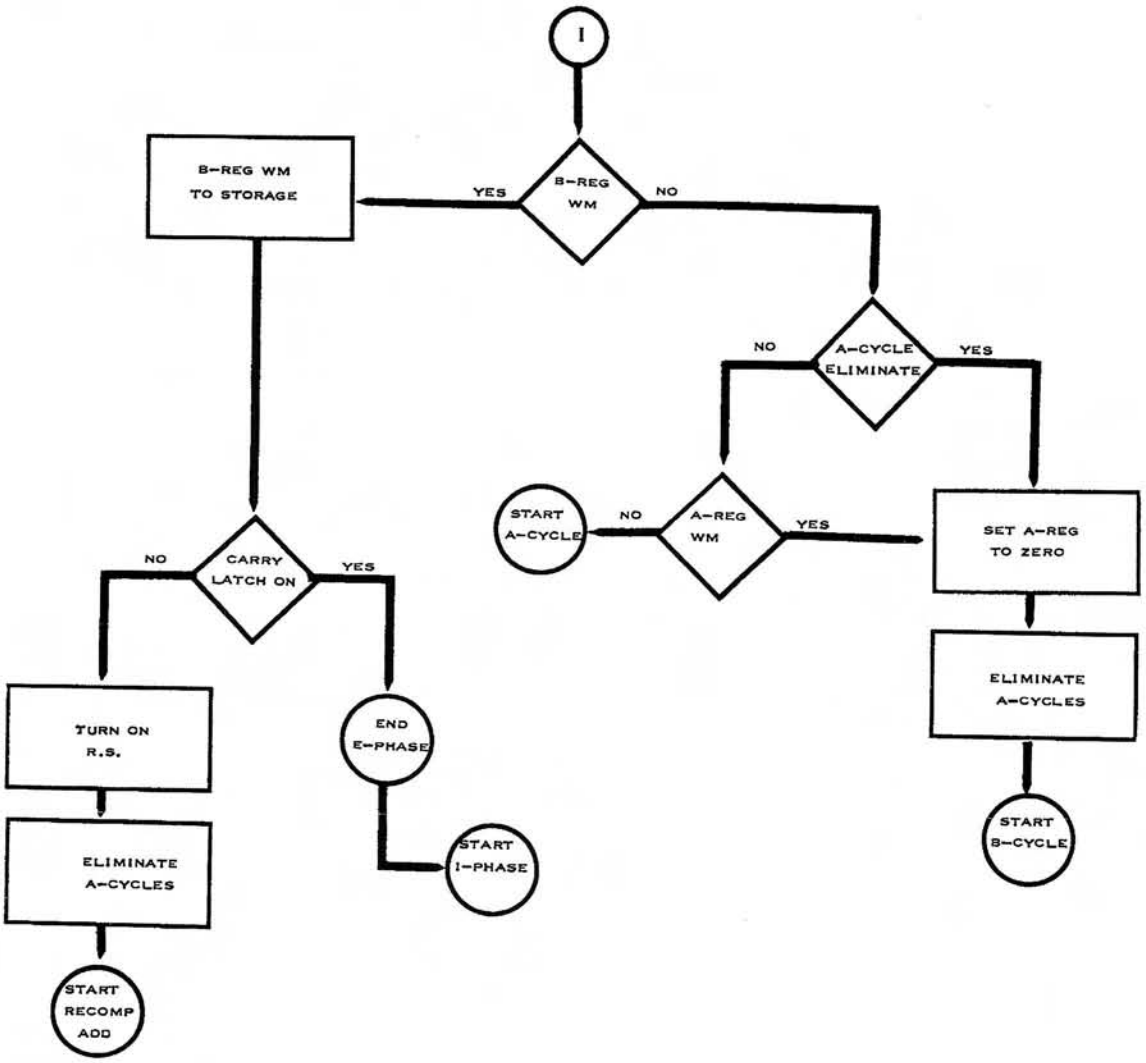
The objective of a complement-add operation with only the A-field specified, is to subtract the A-field data from itself. S AAA is always a complement-add (no recomplement) operation. The resultant A-field sign is the same as the original A-field sign, but is in standard form. A-field zone bits in other than the units (sign) position are destroyed. The A-field must have a WM.

The net effect of S AAA is to set the A-field to zero and assign zone bits to the units position of the A-field to represent the sign. Details of this operation are basically the same as complement-add operations (no recomplement) covered for A AAA BBB or S AAA BBB, and are not repeated here.

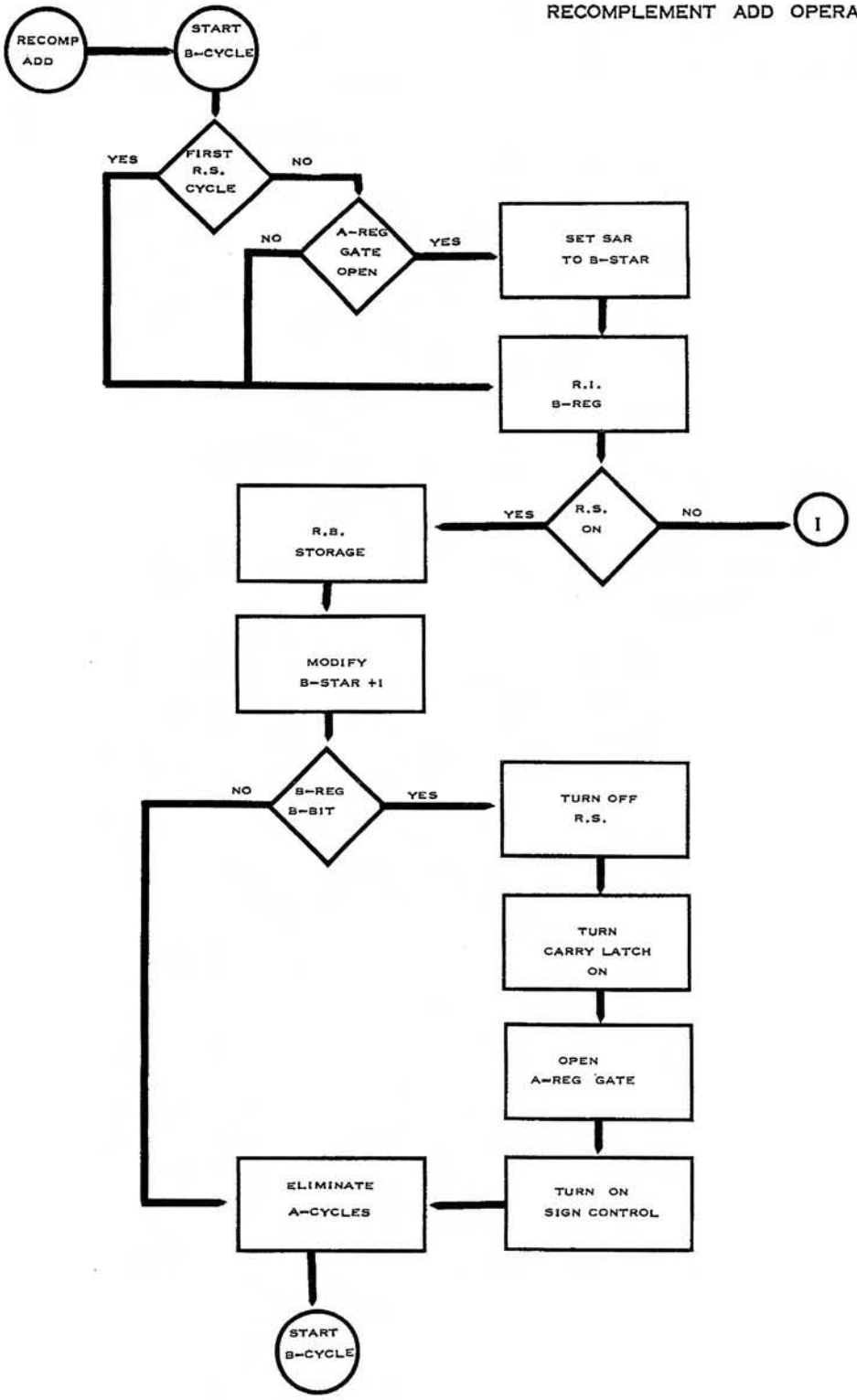
COMPLEMENT ADD OPERATION DIAGRAM

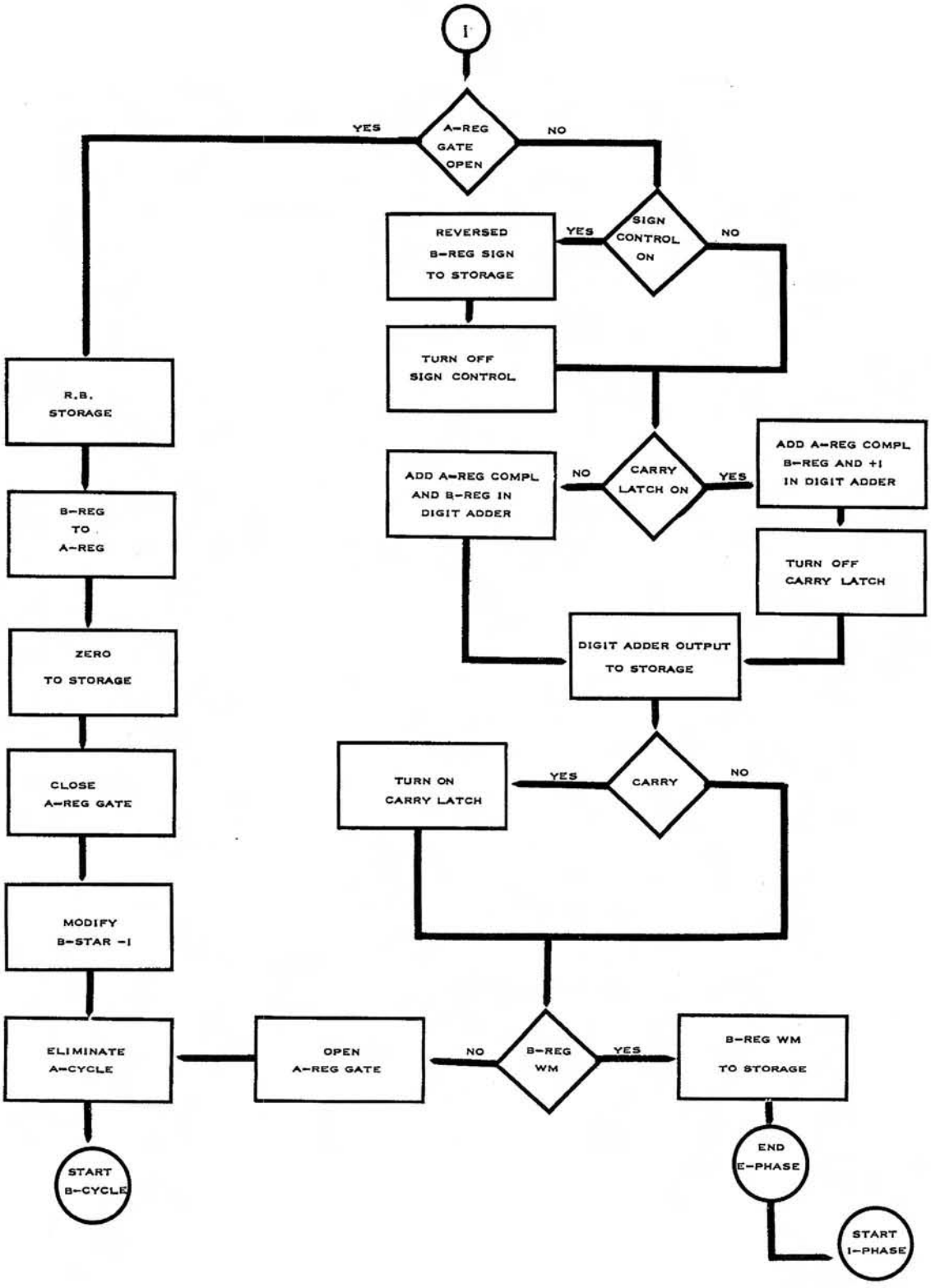


COMPLEMENT ADD OPERATION DIAGRAM CONTINUED



RECOMPLEMENT ADD OPERATION DIAGRAM




I
A-REG GATE OPEN
YES
NO
REVERSED B-REG SIGN TO STORAGE
SIGN CONTROL ON
YES
NO
TURN OFF SIGN CONTROL
R.B. STORAGE
B-REG TO A-REG
ZERO TO STORAGE
CLOSE A-REG GATE
MODIFY B-STAR -1
ELIMINATE A-CYCLE
START B-CYCLE
ADD A-REG COMPL AND B-REG IN DIGIT ADDER
CARRY LATCH ON
NO
YES
ADD A-REG COMPL B-REG AND +1 IN DIGIT ADDER
TURN OFF CARRY LATCH
DIGIT ADDER OUTPUT TO STORAGE
TURN ON CARRY LATCH
CARRY
YES
NO
OPEN A-REG GATE
B-REG WM
NO
YES
B-REG WM TO STORAGE
END E-PHASE
START I-PHASE

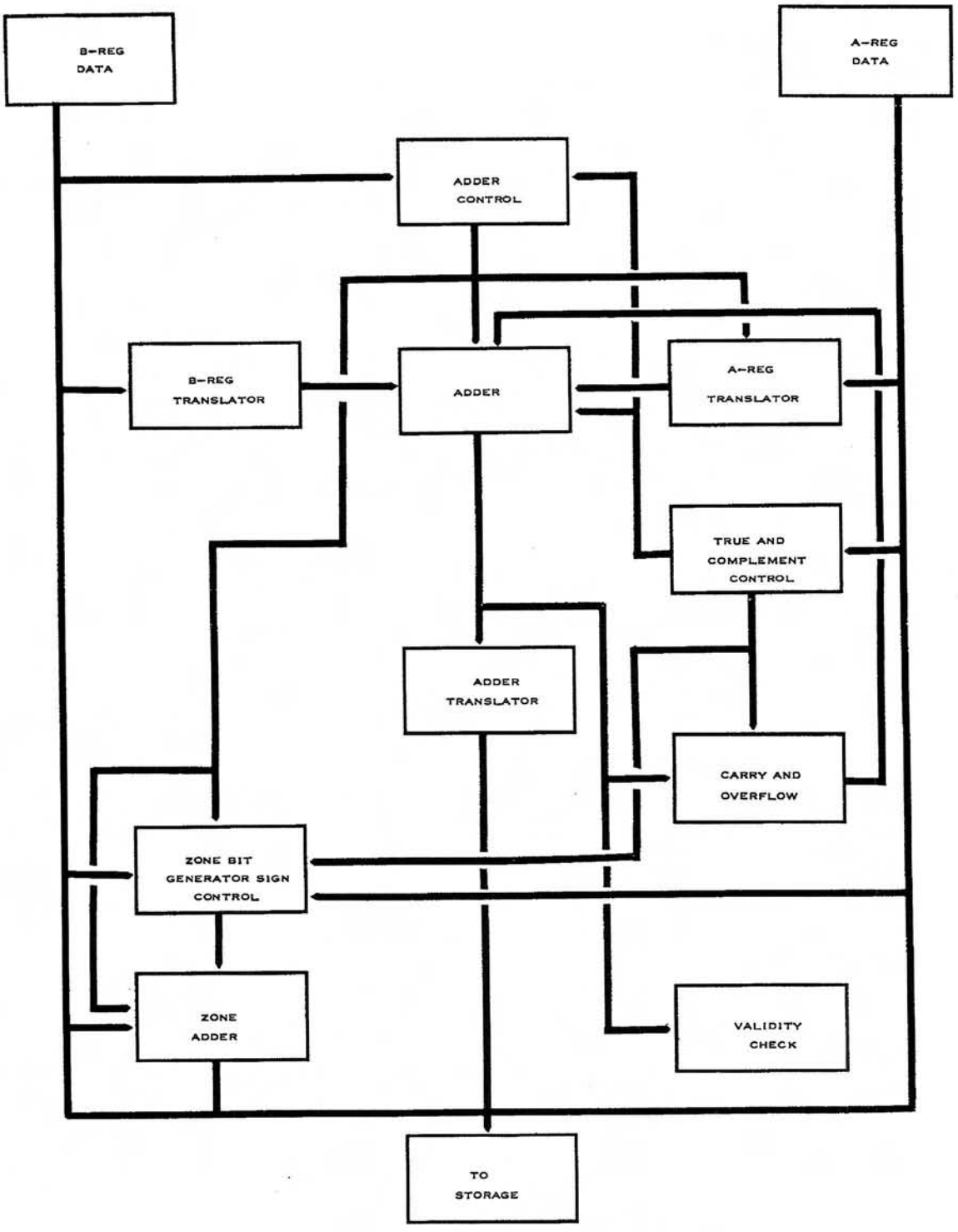


FIGURE 4 - GENERAL ARITHMETIC FLOW

Branch, Test and Branch, Test Character and Branch, Test Zone or WM and Branch

The execution of these instructions occurs partially during the I-phase. Therefore the narrative and Logic Flow Diagrams include discussion of I-Phase as well as E-Phase.

Branch (B III)

This op code directs the system to the address specified by (III) for its next instruction. The decision to branch unconditionally is made a I-4 time by the presence of a B-reg blank or WM.

A branch unconditional operation is accomplished as follows:

1. I-op, I-1, I-2, and I-3 cycles occur in the normal manner. During these cycles the branch-op-code is stored in the op register and the (III) address is stored in the A- and B-STARS.

2. At I-4 time, either a gated WM or a B-register blank causes the following actions:

   A. Sets up EXECUTE ELIMINATE. This eliminates execute cycles and causes the system to stay in I-phase.

   B. Sets the PROGRAM SKIP condition. This causes the system to skip the following instructions in the normal sequence by setting up circuits to gate the A-STAR (contains the next instruction) instead of the I-STAR, into the storage address register.

   C. Restarts I-phase with an I-op cycle.

Test and Branch (B III d)

This instruction causes a branch operation if certain conditions are satisfied; the d-character specifies the conditions to be tested. If the condition tested is not satisfied, the next instruction in sequence is performed.

The various d-characters are reviewed below:

d	Branch on:	d	Branch on:	d	Branch on:	d	Branch on:
bl	Unconditional	A	Sense Switch A "Last Card" Switch	K	End of Reel * ‡	/	Unequal Compare B ≠ A
9	Carr. Chan. #9	B	Sense Switch B *	L	Tape Channel Transmission Error* ‡	S	Equal Compare B = A *
@	Carr. Chan #12	C	Sense Switch C *			T	Low Compare B < A *
		D	Sense Switch D *	$\overset{+}{o}$	Reader Error if I/O Check Stop Switch OFF ‡	U	High Compare B > A *
		E	Sense Switch E *	$\overset{-}{o}$	Punch Error if I/O Check Stop Switch OFF ‡		
		F	Sense Switch F *	+	Printer Error if I/O Check Stop Switch OFF ‡	Z	Overflow ‡
		G	Sense Switch G *			%	Processing Check with Process Check Switch OFF ‡

* optional

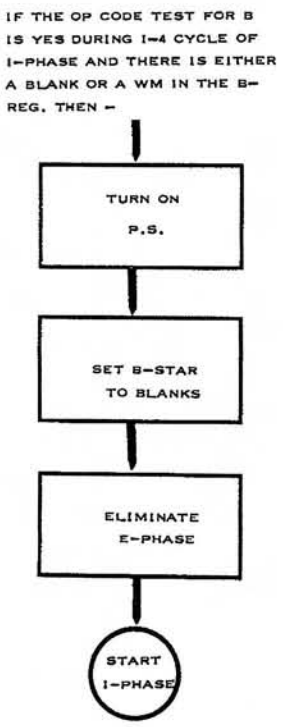
‡Condition reset by a Test and Branch Instruction

The operation is performed as follows:

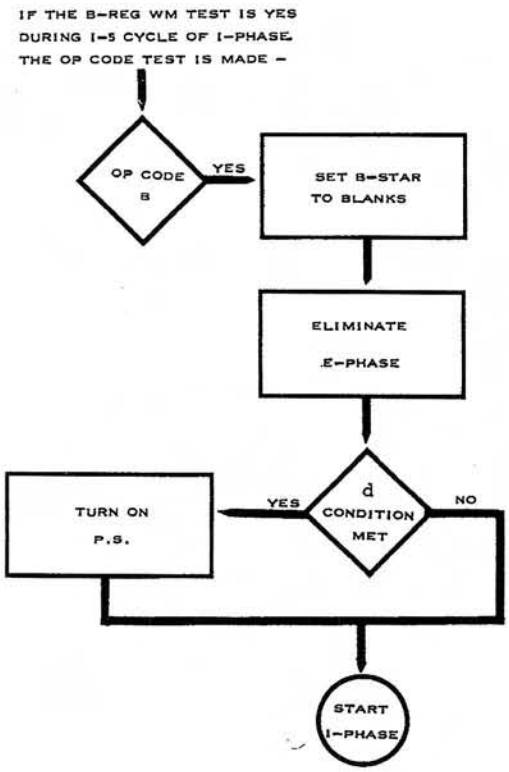
1. I-op, I-1, I-2, and I-3 occur the same way as previously described.
2. At I-4 time the d-character is read into the B-register, the A-register, and the hundreds thousands position of the B-STAR. It serves no useful purpose in the B-STAR; it's easier to let it read in than to prevent it.
3. A B-register WM at I-5 time sets up conditions, within the system, to test for the condition specified by the d-character which is in the A-register.
4. Branch operation and I-5 together also develop execute-eliminate so that the system stays in I-phase.
5. If the test is successful, program skip causes a branch function; if not successful, the program skip is not activated and the system takes the next instruction address from the I-STAR.

BRANCH AND TEST AND BRANCH INSTRUCTIONS DIAGRAMS

BRANCH



TEST AND BRANCH



Test Character and Branch (B III BBB d)

This is a single-character test instruction; it tests the character at the B-address for the same bit configuration as the d-character. I-phase is accomplished in the same manner as just described with the following exceptions:

1. The character in the B-address is tested for the same bit configuration as the d-character itself.
2. The nature of this instruction requires the comparison of the d-character (in the A-reg) to the character specified by the B-address. An E-phase, consisting of one B-cycle, is required to move the B-field character into the B-register so that the comparison can be made.

    a. A B-register WM, which occurs at I-ring 8-time, causes the system to go into E-phase.
    b. Branch operation develops eliminate A-cycle so that the E-phase starts with a B-cycle.
    c. The storage location specified by the B-field address is read into the B-register.
    d. The A- and B-registers are compared.
    e. If equal, they set PROGRAM SKIP; this causes a branch operation as was previously described. If unequal, the next instruction in sequence is performed.
    f. A branch op-code also sets up ONE-CHARACTER I-E CHANGE; this puts the system back in I-phase after one B-cycle.

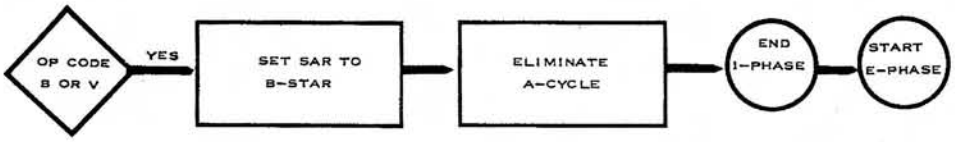
Test Zone or WM and Branch (V III BBB d)

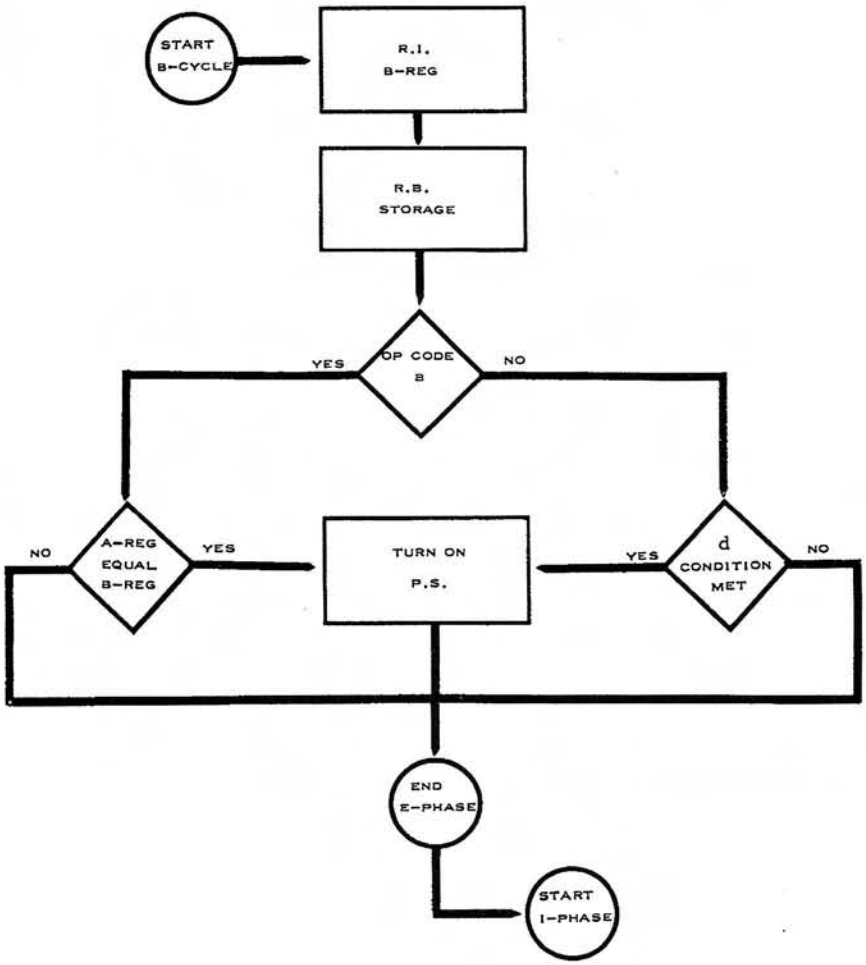
This is also a single-character test instruction; it tests the character located in the (BBB) address for the condition specified by the d-character, and branches if the condition is met. I-phase is the same as for a test and branch operation as was just described. The d-characters and their meanings are as follows:

d-character	Branch to (III) if address (BBB) contains
1	Word-Mark
2	No Zone
B	12-Zone
K	11-Zone
S	Zero Zone
3	Either a WM, or no Zone
C	Either a WM, or a 12-Zone
L	Either a WM, or an 11-Zone
T	Either a WM, or a Zero Zone

TEST CHARACTER AND BRANCH AND TEST ZONE OR WM AND BRANCH INSTRUCTIONS DIAGRAMS

DURING I-S CYCLE WHEN THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE.





No OP

No OP (N)

The op code does nothing. It can be substituted for the operation code of any instruction in a program routine to make the particular instruction ineffective; thus, an instruction can be eliminated without reworking the program routine. It is accomplished by eliminating all execute cycles for the particular instruction.

Stop and Stop and Branch

Stop (.)

The stop instruction causes the system to stop at I-1 time and turns on the light within the stop key. Depression of the start key following a stop operation causes the system to start at the next instruction in sequence unless the operation is a stop and branch operation.

The actual stop occurs when the WM of the next instruction in sequence is detected at I-1 time. If the stop operation is followed by blanks, the system will stop at I-4 time and treat it as a stop and branch operation.

Stop and Branch (. III)

Stop and Branch instructions cause the system to stop at I-4 time with the stop key light turned on. The new instruction address will be in the A-STAR.

Depression of the start key following a stop and branch operation causes the system to start at the instruction located at the new address. The actual operation is accomplished as follows:

1. I-op, I-1, I-2, and I-3 occur as normal I-phase cycles; the op code is read into the op register, and the new instruction address (III) is read into the A-Star.

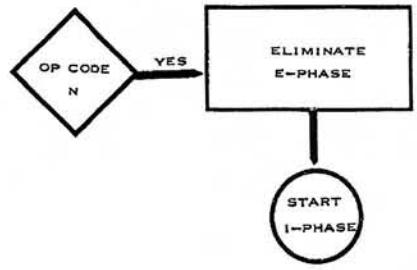
2. At I-4 time a blank or WM in the B-register activates the circuits to cause an unconditional branch operation, in addition to stopping the system. When the start key is depressed, the instruction address is taken from the A-STAR instead of the I-STAR.

If I-phase continues into I-5 cycle, the branch is eliminated so that when the start key is pressed, SAR is set to I-STAR.

NO OPERATION AND STOP INSTRUCTIONS DIAGRAMS

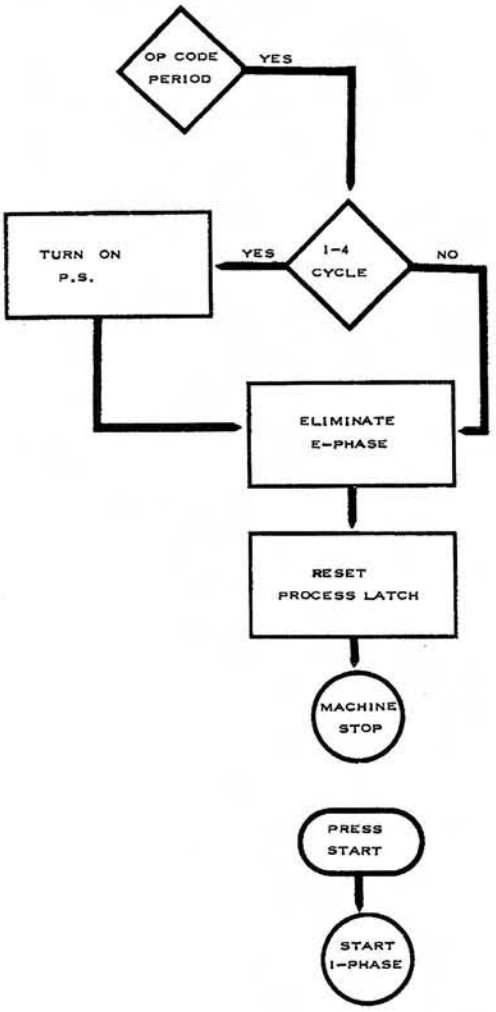
NO OPERATION

DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE -



STOP

DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE-



Read (1)

The major objectives of this op code are:

1. Initiate a read-feed cycle in the IBM 1402 Card Read Punch.
2. Set up circuitry in the 1401 which allows the information read from the card at the first reading brushes to condition the hole count check planes.
3. Set up circuitry in the 1401 which allows the information read from the card at second reading, to enter the read area of main storage (locations 001-080) in BCD form, and to condition the check planes so that a hole count check may be completed.

The cards move through the read feed, 9-edge first, face down. The cards passing the first and second read brushes are read into two separate 80-core extensions of main core storage. These extensions of storage are called row-bit cores. The 80-first read brushes are connected to the read one row-bit cores. The 80-second read brushes are connected to the read two row-bit cores. In each case, brush 1 connects to row-bit core one, brush 2 to row-bit core two and brush 80 to row-bit core 80.

The information in the card is read by the brushes, parallel by digit and cycle point by cycle point. When a punch in a card column is read, the row-bit core associated with that card column will be "flipped." A row-bit core is flipped each time there is a coincidence of a punch in a card column and an impulse (circuit breaker impulse).

As the impulse from the circuit breakers break for each digit time, a read scan condition is set up. The main purpose of a read scan is to transfer the row-bit core information into the read area of main storage in BCD form.

Each read scan is a series of B-cycles in which main storage locations 001 through 080 are addressed serially by the storage address register. Each time main storage is addressed, row-bit cores are also addressed. If main storage location 003 is addressed, row-bit core 3 is also addressed in each row-bit core plane.

On each B-cycle of a read scan, if an output is received from a read one row-bit core, a core in the check planes is caused to change status. The check planes are used to perform a hole count check. The check for this card will be completed when this same card is read at the second reading station.

When an output is received from a read two row-bit core, it is sent to the check planes to complete the hole count check for this card, at the same time the read two-row bit core output is presented to the read encoder. At the read encoder, the row-bit core output and a read scan condition allows the BCD coded character stored in A-register to be gated into the read area of main storage. This procedure is repeated for each read scan, at each digit time of the 1402 read feed.

As the card progresses through the read feed, the A-register will contain the digit time (in BCD form) of the read feed. When the A-register contains 8-1-C, the read feed is at a 9-digit time; a 4-2-1 in the A-register indicates the read feed is at 7-digit time; and an A-B-C in the A-register indicates that the read feed is at 12-digit time.

A 9-punch in the card, read at second reading, will set a row-bit core. Durin g the read scan for 9-digit time, an 8-1-C will be gated into the read area of main storage when the row-bit core is addressed by the storage address register. All digits will be handled in a similar manner, on their respective read scans.

Each time the storage address register addresses a position in main storage, the character in this position is read out into the B-register. On each B-cycle, after the read scan for 9-digit time, the B-register character is always transferred back to main storage. This procedure allows the characters in the B- and A-register to be combined into one BCD coded character on any read scan after the one for 9-digit time. The combining of the B- and A-register into one character will enable the 1401 to accept alphabetic and spe cial character card punching.

After the read scan for 12-digit time has been completed, the 1401 will start the next I-phase.

The operation of this instruction is executed as follows:

When a read op code is read into the op register, a read-feed cycle will be initiated. The 1401 will be interlocked until the impulse circuit breakers break for nine digit time. After the impulse circuit breakers break for nine digit time, the first of 12 read scans will be set-up. A read scan will be set up each time the impulse circuit breakers break.

The read-scan condition will start cycle control which will produce B-cycles. The first three B-cycles of read scan 1 are used to:

1. Reset storage location 000 to zero. This location will be used to store the read-feed digit time, in BCD form, for each read scan.
2. Complement-add the contents of the A-register to the contents of the B-register. The result will be the read-feed digit time, in BCD form, which will be stored in location 000.

3. Transfer the contents of storage location 000 to the A-register and transfer the contents of the B-register back to storage location 000.

On each read scan after read scan 1 only two B-cycles are need to:

1. Complement-add the contents of the A-register to the contents of the B-register. The result will be the read-feed digit time, in BCD form, which will be stored in location 000.
2. Transfer the contents of storage location 000 to the A-register and transfer the contents of the B-register back to storage location 000.

When the A-register contains the correct digit time of the read feed, a read scan of row-bit cores will begin. Each row-bit core is scanned serially, starting with the core specified by the address in the storage-address register. To control the read scan, the address register is modified by a +1 on each B-cycle of each read scan.

On each B-cycle of a read scan if an output is sensed from a read two row-bit core, it is presented to the read encoder. The row-bit output at the read encoder will allow the contents of the A-register to be gated back to the storage location specified by the storage-address register. On each read scan the procedure is executed each time a read two row-bit output is received at the read encoder.

During read scan 1, the contents of the B-register are not allowed to transfer back to storage. Therefore, the read area of main storage will be cleared of any previous information on read scan 1. After read scan 1, the B-register is transferred back to storage on each B-cycle of a read scan.

Each scan will end when the storage-address register reaches address 080. The last address line will be activated at this time ending the read scan. A new read scan will be started when the impulse CB's break for the next digit time of the read feed.

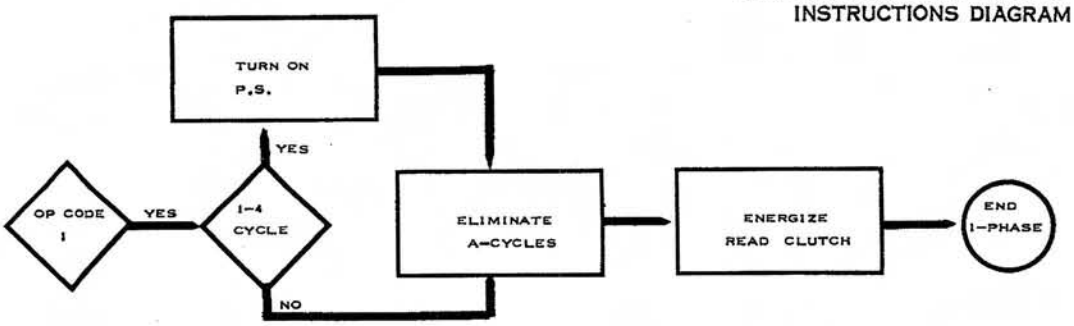
When all twelve read scans are completed, the READ SCANS COMPLETE TRIGGER will come on. This trigger will turn on the ALL-SCANS-COMPLETE TRIGGER, which will enable the 1401 to start into the next I-phase.

Read and Branch (1 III)

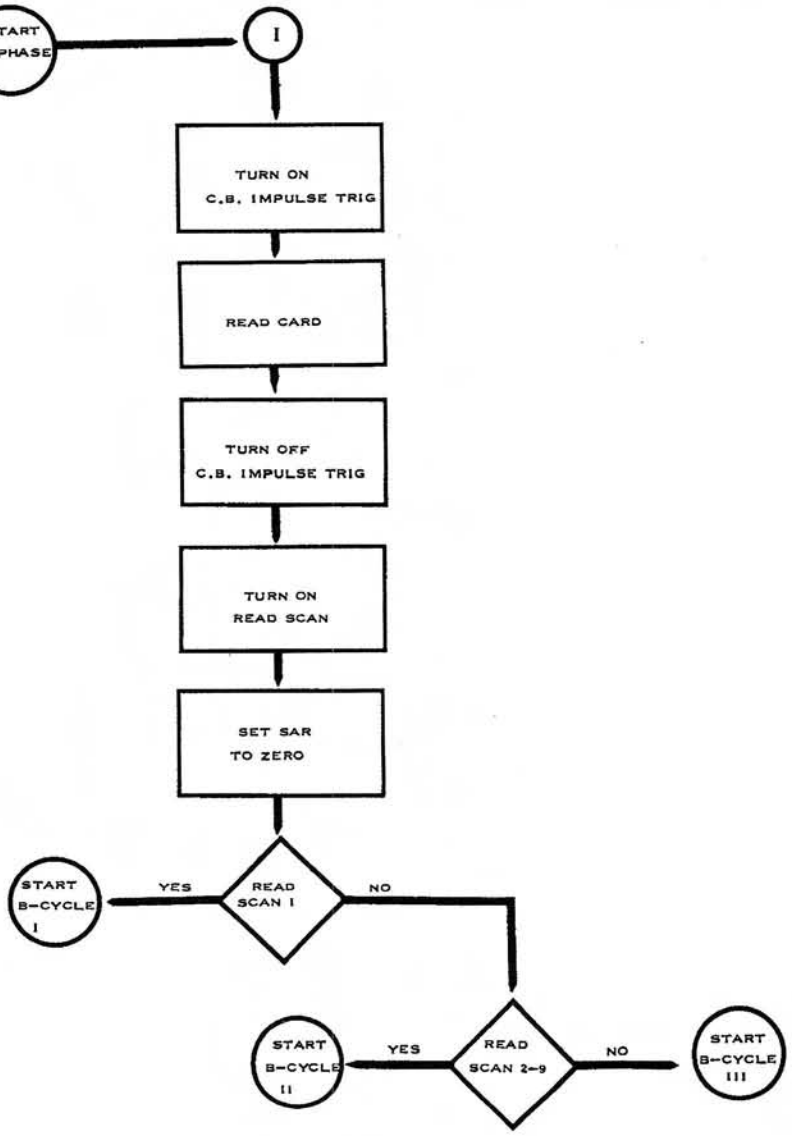
This instruction will cause the card to be read in the normal manner. After the card has been read by the 1402 Card Read Punch, the 1401 program will branch unconditionally.

Card reading takes place on E-phase. When E-phase is completed the 1401 starts an I-phase. At the beginning of I-phase, the A-address register is gated into the storage address register. Therefore, the next instruction will be taken from the address specified by the A-address register instead of from the address specified by the instruction address register. This causes a branch in the program.

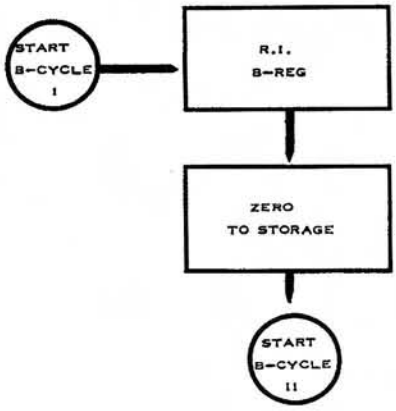
READ AND READ AND BRANCH INSTRUCTIONS DIAGRAM

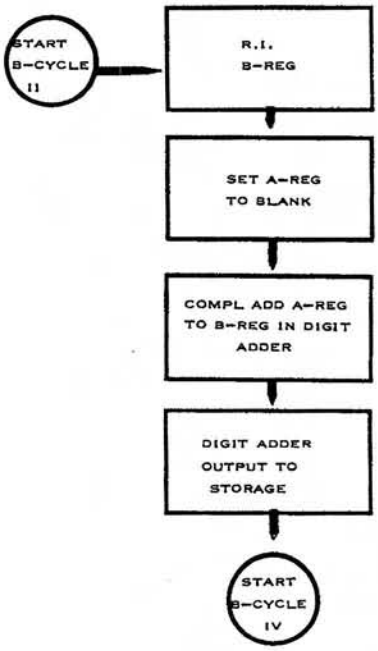


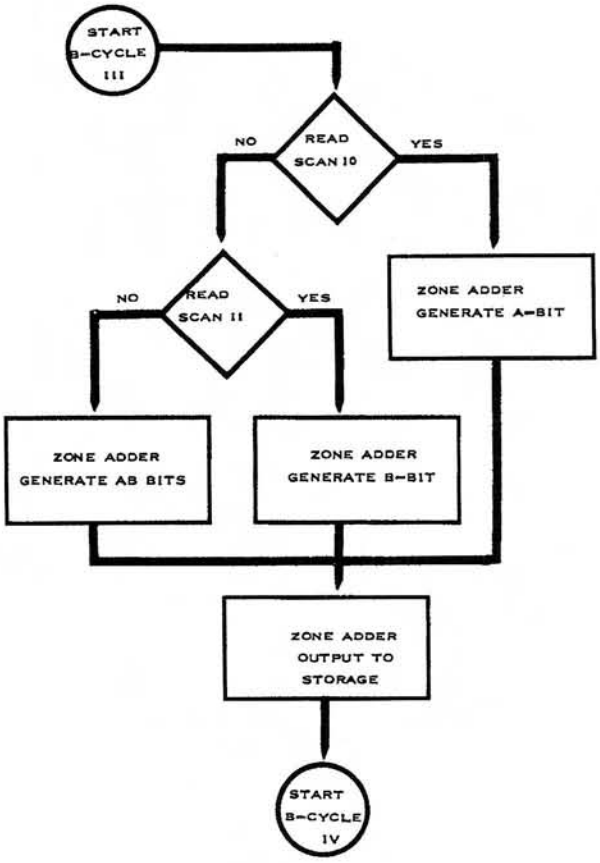
DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE.

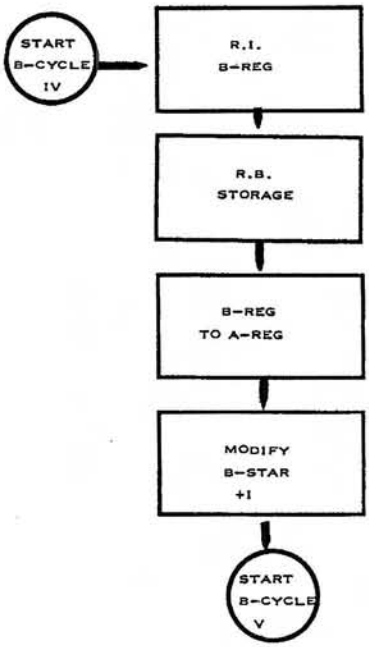


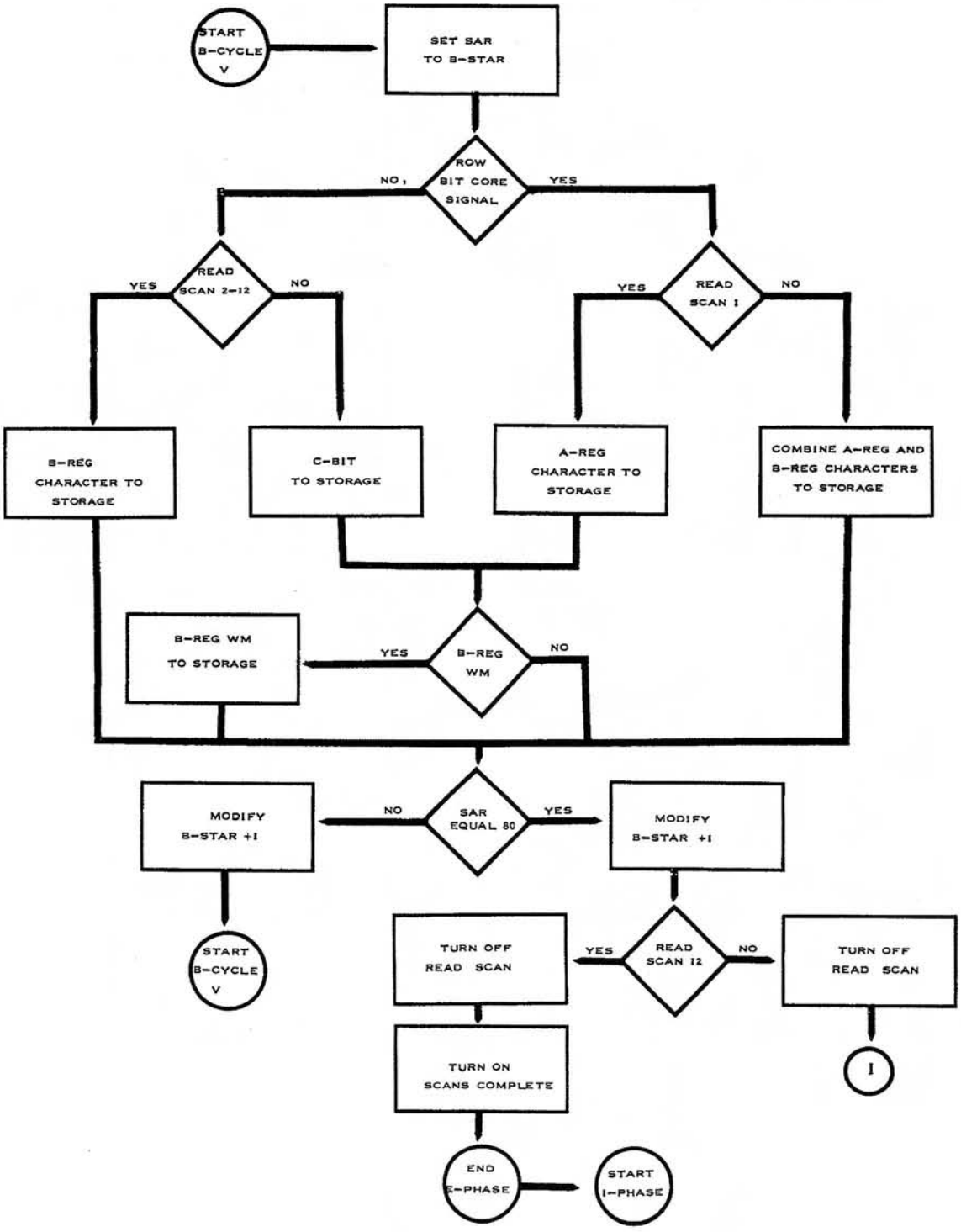
READ AND READ AND BRANCH INSTRUCTIONS DIAGRAM CONTINUED










START B-CYCLE V
SET SAR TO B-STAR
ROW BIT CORE SIGNAL
NO
YES
READ SCAN 2-12
YES
NO
READ SCAN 1
YES
NO
B-REG CHARACTER TO STORAGE
C-BIT TO STORAGE
A-REG CHARACTER TO STORAGE
COMBINE A-REG AND B-REG CHARACTERS TO STORAGE
B-REG WM TO STORAGE
YES
B-REG WM
NO
MODIFY B-STAR +1
NO
SAR EQUAL 80
YES
MODIFY B-STAR +1
START B-CYCLE V
TURN OFF READ SCAN
YES
READ SCAN 12
NO
TURN OFF READ SCAN
TURN ON SCANS COMPLETE
I
END E-PHASE
START I-PHASE

Punch (4)

The major objectives of this op code are:

1. Initiate a punch-feed cycle in the 1402 Card Read Punch.
2. Set up circuitry in the 1401 which allows the card passing the punch unit to be punched with the information contained in the punch area of main storage (locations 101-180). At the same time, condition the check plane cores so that a hole count check may be performed.
3. Set up circuitry in the 1401 which allows the information read from the card at the punch check brushes to condition the check plane cores so that a hole count check may be completed.

The cards move through the punch feed 12-edge first, face down. Before each digit punching time, a punch scan condition is set up by punch scan circuit breakers. A punch scan will be set up for each digit punching time by these circuit breakers.

A punch scan is a series of B-cycles, in which main storage locations 100 through 180 are addressed serially by the storage address register. On each B-cycle of a punch scan, the character in the storage location being addressed is read into the B-register. The B-register character is presented to punch decode.

At punch decode, the B-register character is compared against the A-register character. (The character contained in the A-register is the digit punching time of the punch feed, in BCD form.) If the two characters are equal, circuitry is set up to cause this character to be punched in the card and also to condition a check plane core. When the A- and B-register contain A, B, C, circuitry will be set up to cause a 12 to be punched in the card, if both registers contain 4, 2, 1, a 7 will be punched in the card. All punching of the card takes place in this manner.

The card column that is punched is determined by the units and tens position of the storage address register. If the storage address register contained 124, card column 24 would be punched.

At the same time that a card is being punched, another card is being read at the punch check station. The information read from this card is read into the punch check row-bit cores, parallel by digit, cycle point by cycle point. On each punch scan the row-bit core information will condition the check planes so that a hole count check may be completed for this card.

When card punching and reading are completed, the next I-phase will start in the 1401.

The operation of this instruction is executed as follows:

When a punch op code is read into the op register, a punch feed is initiated. The 1401 is interlocked until the punch scan circuit breakers make. When these Circuit breakers make, the first of 13 punch scans is set up. Twelve punch scans are required to complete card punching. The thirteenth punch scan is used to complete checking of the card read at the punch check brushes. Checking is explained in Checking of these notes.

A punch-scan condition will set up cycle control to produce a series of B-cycles for each punch scan.

On punch scan one, three B-cycles are needed to

1. reset storage location 100 to zero. This location will be used to store the digit time of the punch feed, in BCD form, for each punch scan.
2. produce the correct digit time of the punch feed and store it in storage location 100.
3. transfer the contents of storage location 100 to the A-register and transfer the contents of the B-register back to storage location 100.

On each punch scan, after punch scan one, only two B-cycles are needed to:

1. produce the correct digit time of the punch feed and store it in storage location 100.
2. transfer the contents of storage location 100 to the A-register and transfer the contents of the B-register back to storage location 100.

When the A-register contains the correct digit time of the punch feed, a punch scan of storage locations 101 through 180 will begin. Each storage location is addressed on a B-cycle, and read into the B-register.

The B-register character is presented to punch decode. At punch decode the B-register and A-register characters are compared to determine if this digit is to be punched in this card column. If the two characters are equal, circuitry is set up to energize one of the 80 punch magnet drivers (one for each punch magnet).

The punch magnet driver energized is determined by the units and tens position of the storage-address register.

When the storage-address register reaches address 180, the last address line will be activated. The last address line will end the punch scan for each digit time.

When a punch scan has been completed, for a particular digit time, the punch magnet drivers will energize their respective punch magnets under control of punch circuit breakers. After the punch magnets have been energized, the mechanical action of the punch unit will punch the holes in the card.

At the completion of card punching for each digit time, the punch magnet drivers are reset by circuit breakers. After reset, a new punch scan is started for the next digit time in sequence.

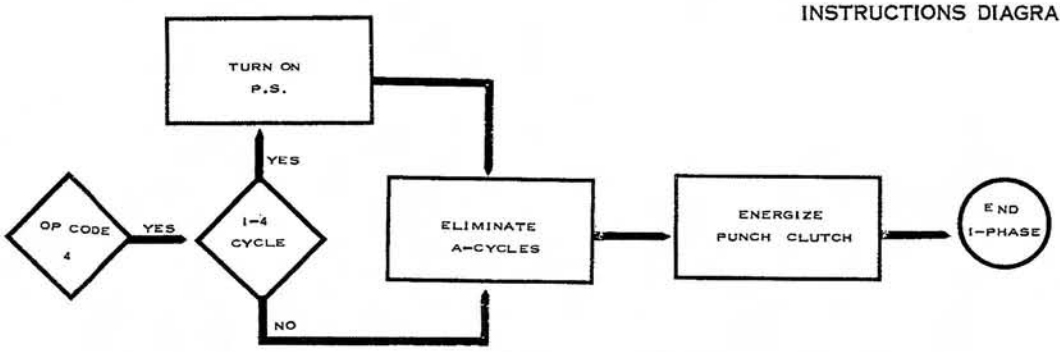
When the 13-punch scans have been completed, the all-scans-complete-trigger will come on. This trigger will enable the 1401 to start into the next I-phase.

Punch and Branch (4 III)

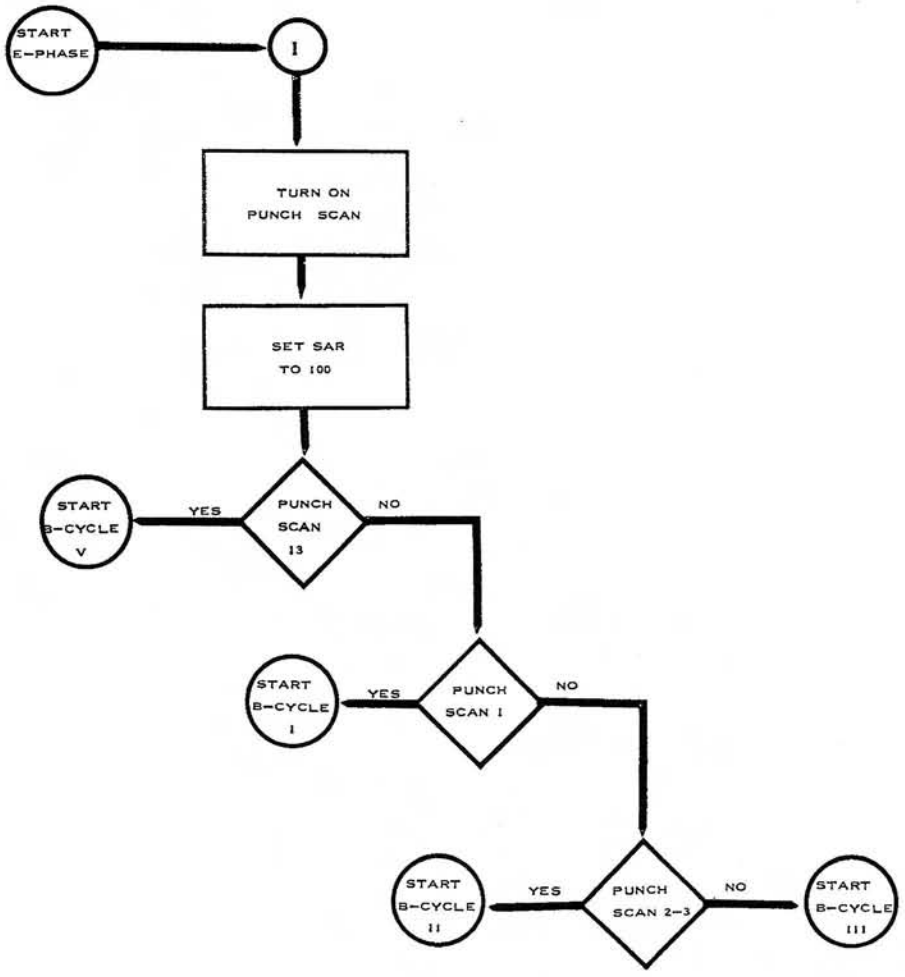
This instruction will cause the card to be punched in the normal manner. After the card has been punched by the 1402 Card Read Punch, the 1401 program will branch unconditionally.

Card punching takes place on E-phase. When E-phase is completed, the 1401 starts an I-phase. At the beginning of the I-phase, the A-address register is gated into the storage address register. Therefore, the next instruction will be taken from the address specified by the A-address register instead of from the address specified by the instruction address register. This will cause a branch in the normal sequential program.

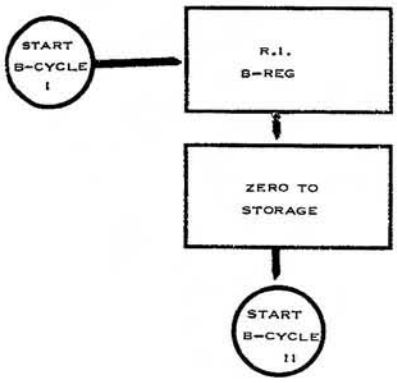
PUNCH AND PUNCH AND BRANCH
INSTRUCTIONS DIAGRAM

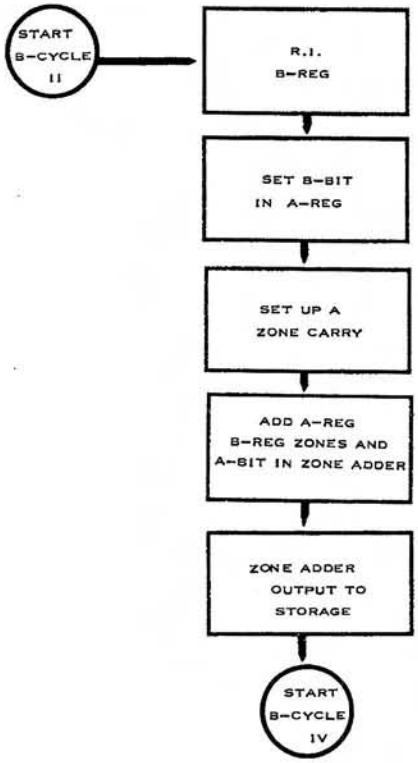


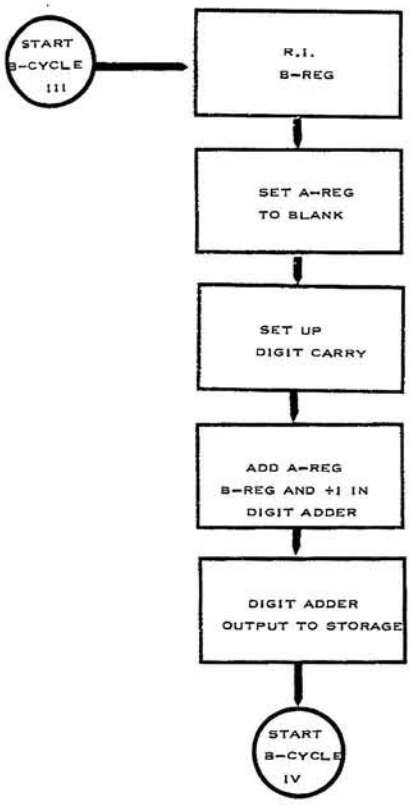
DURING THE I-PHASE CYCLE THAT THE B-REG WM TEST IS YES, THE OP CODE TEST IS MADE.

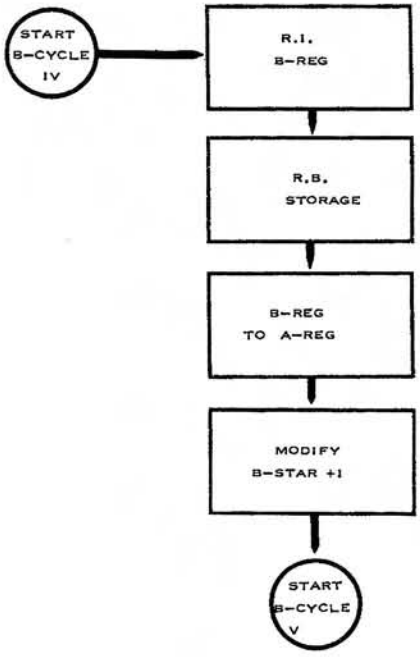


PUNCH AND PUNCH AND BRANCH
INSTRUCTIONS DIAGRAM CONTINUED

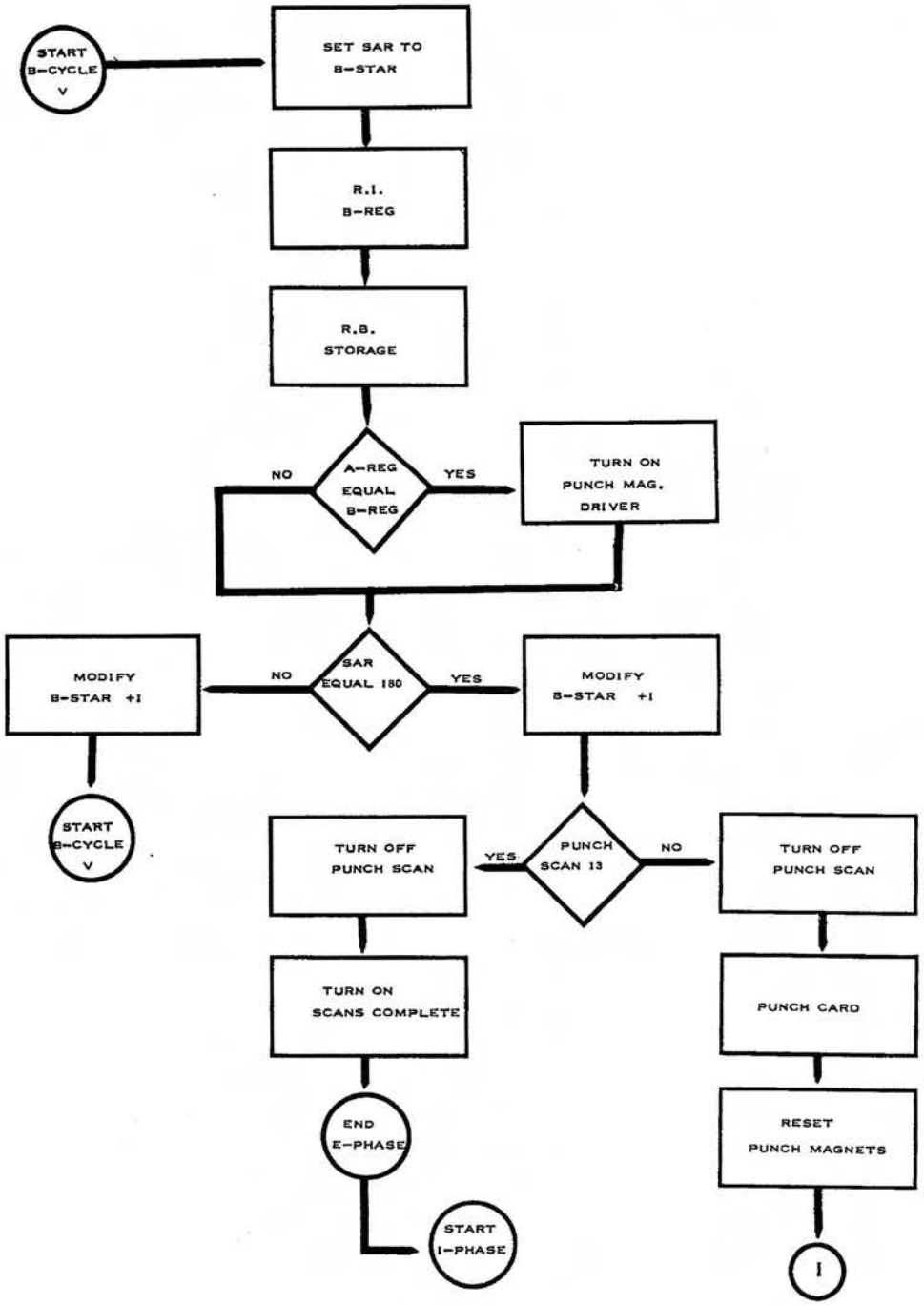








PUNCH AND PUNCH AND BRANCH
INSTRUCTIONS DIAGRAM CONTINUED



In core storage, positions 201 through 322 are the printer output area. With the 1403 Model 1, (100 print positions) storage positions 201 through 300 are used. Four check planes are associated with the printer output area. One core in each of the check planes corresponds to each address in the printer output area.

In any instruction associated with printing, one of two kinds of printing will occur. The kind of printing that occurs is determined by whether or not a lozenge (⌑) is included within the instruction as a d-character. When a lozenge does not occur as a d-character, normal printing takes place; when a lozenge does occur, word-mark printing takes place. Normal printing causes all printable characters in the printer output area to print out on one line. (The carriage automatically single spaces if no "AFTER PRINT" carriage operation is set up.) Word mark printing causes a normal printing to take place followed by a printing on the next line of a 1 for each storage location in the printer output area that contains a word mark. Nothing else prints on this line. (Carriage operation is unaffected by the kind of printing that takes place.)

The mechanical units required to do the actual printing are a type array and a hammer unit. The type array spans the entire length of the print line. The hammers force the paper and ribbon against this unit to cause printing. There is one hammer for each print position.

The type array contains five identical alphabets of 48 characters each, including numbers and special characters. Within each alphabet, the characters are grouped in sequence by the modified BCD character bit values. This arrangement facilitates identification of characters as the type becomes aligned with the various print positions.

Within the array there are two type characters per slug, and the slugs are fastened to a steel tape to form an endless chain one character high and 240 characters long. The type and tape move continuously along the print line from the highest numbered print position (132nd) toward the lowest numbered print position (1st). The type is exposed to the hammers along the print line.

Printing takes place serially, one character and one print position followed by another character and another print position. The sequence in which characters become aligned to the hammers is as follows: hammer one followed by every third hammer (4, 7 - 130); two, followed by every third hammer (5, 8-131); three, followed by every third hammer (6, 9-132). This sequence is repeated 48 times to print one line. It insures that every hammer has been exposed to one complete alphabet.

Thus, there are 6336 possible print selection times. These are called options to print. During each of these times, core storage is addressed to determine the character contained in the storage position corresponding to the hammer. Of these options to print, one and only one is used to select each print position in which a character is to be printed. This use is called the allocation of a print

time to a print position and causes the hammer to fire. It occurs when the character desired to be printed is aligned for printing in the selected print position.

The time required for the type to move one-half of a print-span is called a subscan. The time required for three subscans is called a print scan. The first subscan starts when a type is aligned with the first print position. The second subscan starts when a type is aligned with the second print position. The third subscan starts when a type is aligned with the third print position. A print scan consists of a first, second, and third subscan.

During each subscan, one third of the hammers, each in progressive sequence, is optioned to some character in the type array at 11 μs intervals. At the end of a print scan, each of the hammers will have been optioned to a character in the type array.

At the end of 48 print scans, all of the hammers will have been optioned to a complete alphabet. Hence, 48 print scans are required to print a line.

Printing is initiated only at the beginning of a first subscan. During printing, the subscans are identified, and the print scans are counted.

For the printing rate of 600 lines per minute, a subscan is 555 μs, a print scan is 1665 μs, and a print line is approximately 80 ms. At this printing speed the type is moving at 90.3 inches per second. During a subscan, 484 μs are required to provide an option to print for the 44 positions scanned (44 times 11 equals 484). A clock is started at the beginning of each subscan. It is stopped in each subscan after all 44 positions scanned (for 132 position print line) have had an option to print.

To be able to print intelligible information, three things must be known:

1. The specific character within the type array which is aligned for printing in a print position
2. The specific print position with which this character is aligned
3. The character in storage desired to be printed in this position.

Through control and comparing circuitry, the knowledge of these three things enable the printer to print the desired character in the desired print position.

Three triggers, one ring, and two counters are used to determine which character (within the type array) is aligned with some print position. The three triggers are the print subscan (PSS) trigger, the home trigger, and the run trigger. The ring is the print subscan (PSS) ring. The counters are the print subscan (PSS) counter and the compare counter. These units are discussed in order below.

PSS Ring

The pss ring identifies which of the first three print positions is the one that has a character, in the type array, in position to be printed in it at the beginning of each subscan. When ring position three is on at the beginning of a subscan, print position one will be the first position that has an option to print.

The PSS ring in the 1401 is reset to ring position 3 when the run trigger is off. After the run trigger comes on, ring position 3 causes the pss counter to regress to one when the first home pulse occurs. The chain and pss counter are then synchronized, and the PSS counter will continue to operate until the run trigger is turned off.

PSS Counter

1. Serves as a register that can identify any one of 48 characters used within the type array.

2. Indicates the first character that is eligible to be printed at the beginning of a subscan in print position one, two, or three.

3. Is composed of six triggers, one for each bit value of the modified BCD code.

4. Changes its value (and the character it identifies when a PSS pulse occurs).

5. Is reset with the 2-bit trigger on and all other bit triggers off. (Reset occurs because the run trigger is off.)

6. Regresses (subtracts) when PSS ring position three is on, and a PSS pulse occurs.

7. Progresses (adds) when PSS ring position three is off and a PSS pulse occurs.

8. Counts in a binary manner.

9. Contains a numeric sum (sum of 1-bit, 2-bit, 4-bit, and 8-bit triggers that are on) which is never less than one nor greater than 12.

The PSS ctr counts in a binary fashion. It can add (progress), and it can subtract (regress). The operation of the counter is controlled so that the sum of its numeric bits is never greater than 12, nor less than one. The counter identifies, by bit values, the character in the type array that is in position to be printed in one of the first three print positions at the start of a subscan.

If either PSS ring position one or position two is on when a PSS pulse occurs, the PSS counter will add. If PSS ring position three is on when a PSS pulse occurs, the PSS counter will subtract. The PSS counter changes its numeric

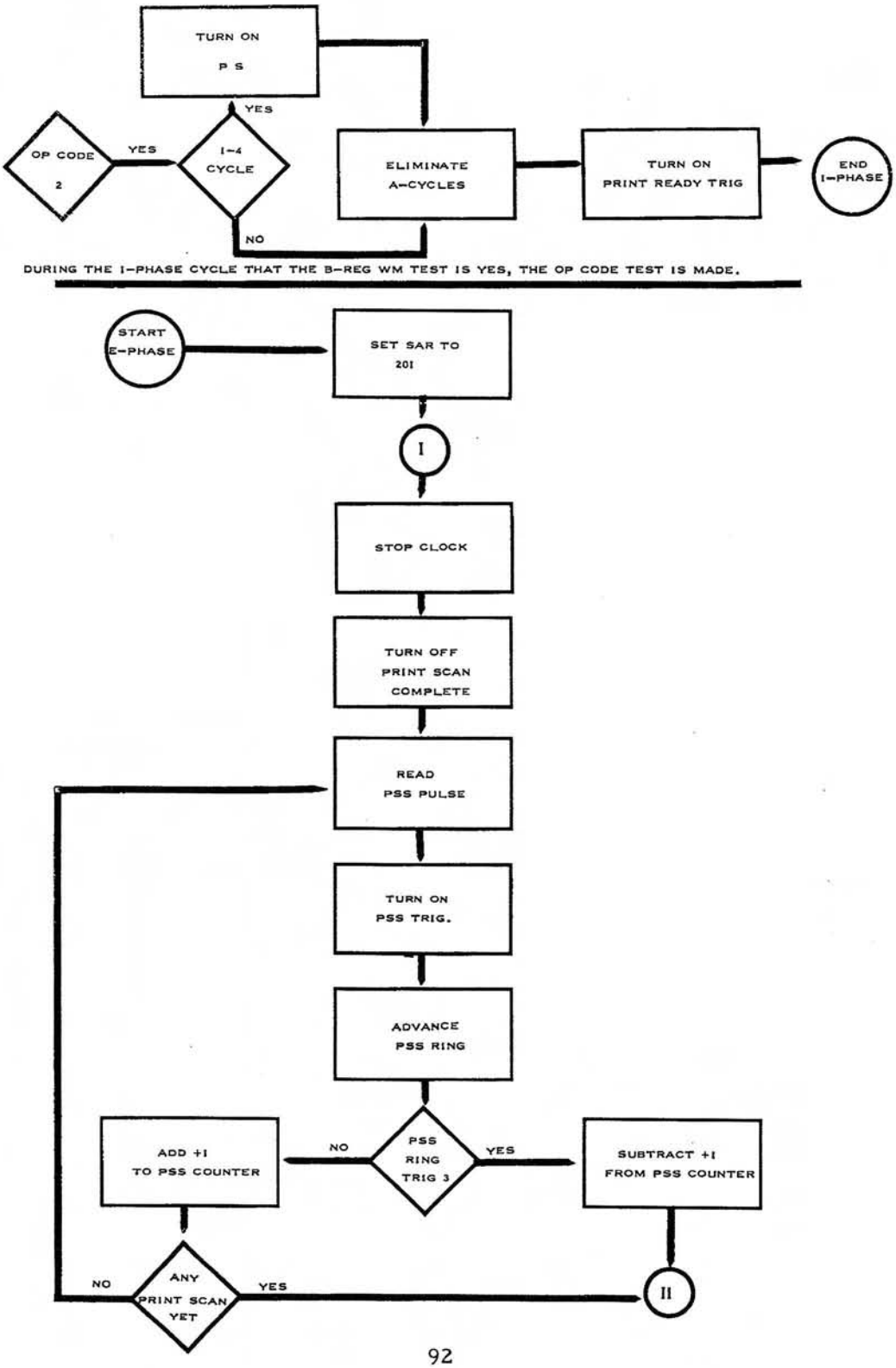
bit values by one except when it is progressing or regressing into a different character grouping as defined by the A- and B-bits. As it progresses from one grouping to another, it goes from 12 to one. As it regresses from one grouping to another, it goes from one to twelve.

Compare Counter

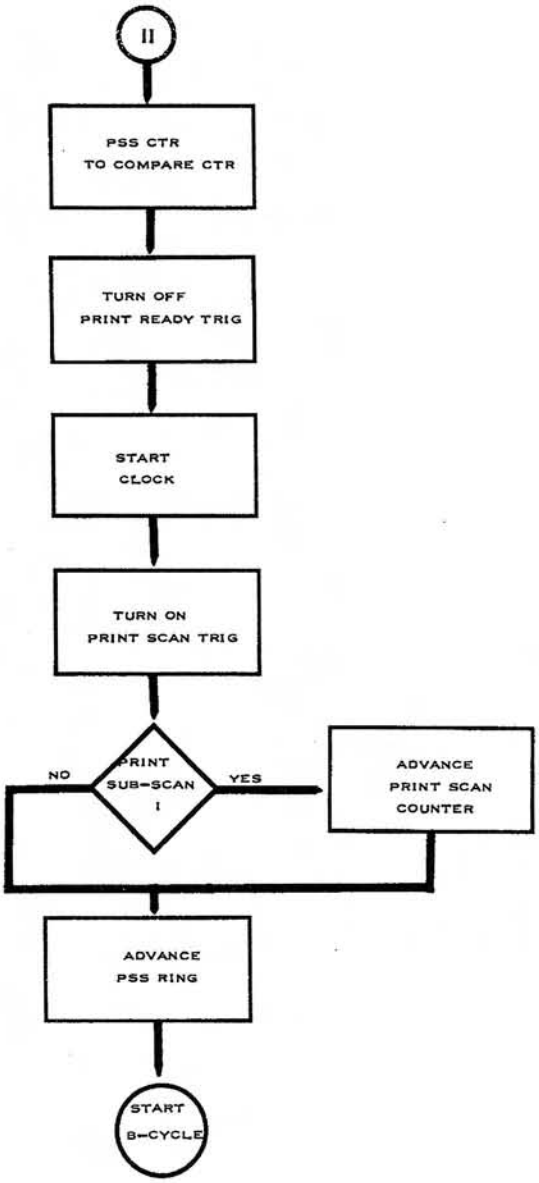
The compare counter value identifies which character in the type array is in position to be printed.

After the leading edge of the PSS pulse steps the pss counter to the next character, the condition of the bit triggers in the PSS counter are set into the bit triggers of the compare counter by collector pullover. The PSS pulse gates this transfer. The compare counter has no 1-bit trigger. Therefore, the 1-bit trigger of the PSS counter must be used together with the compare counter to identify odd-bit characters (characters having a 1-bit). If the printer is in a print operation, the PSS pulse will end when the PSS trigger is turned off during the first B-cycle of the subscan. During the following B-cycles of the subscan, the compare counter advances by a bit value of two.

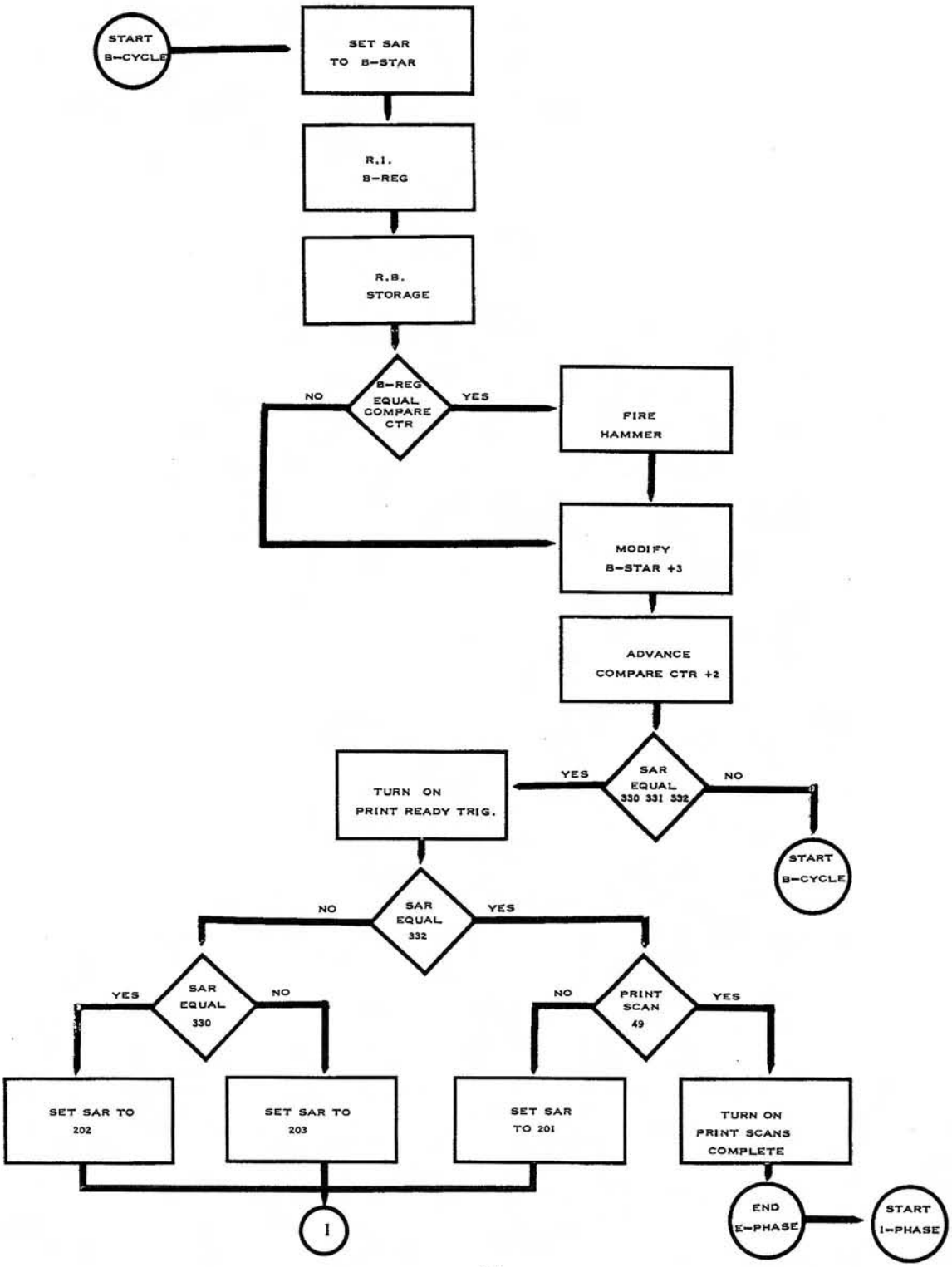
PRINT AND PRINT AND BRANCH
INSTRUCTIONS DIAGRAM



PRINT AND PRINT AND BRANCH
INSTRUCTIONS DIAGRAM CONTINUED



PRINT AND PRINT AND BRANCH
INSTRUCTIONS DIAGRAM CONTINUED



CHECKING FEATURES

1401 checks:

Parity Check

Parity is the even or odd bit count of a particular group of bits. If the bit count is odd, it is odd parity; if even, it is even parity.

The 1401 is an odd parity system. Characters read into the system or generated in the system that have even parity are changed to odd parity by adding a check bit; therefore, all characters should have odd parity.

Parity checking is performed at various points in the system (Figure 1) to determine the condition of the character. Even parity indicates a malfunction and stops the system.

Validity Check

Of the hundreds of bit configurations possible among a choice of eight bits per character position, few more than 50 are "recognizable" characters. A bit configuration is considered to be invalid if it falls outside the realm of real characters. For example, an invalid character containing a C, A, B, 1, 2, 4 and 8 bit would pass a parity check, but would not pass a validity check. Characters are checked for validity when read into storage from the read side of the card read-punch. Characters read out of the storage address register are also checked for validity as well as the output of the logic unit (adder) and the op register.

Valid Address Check

The validity of the address is dependent on the storage capacity of the system. Any 3-digit address, including the hundreds position zone, is valid for a 4K system, while only addresses between 0000 and 1399 are valid for systems with 1.4K storage.

Close examination of the addresses involved with the 4K storage system shows that there is no such thing as an invalid address. The possible bit combinations of the hundreds position(zones AB, A only, B only, and neither A nor B, along with any valid digit), are all valid as they address locations between 0000 and 3999.

Examination of the hundreds position of the invalid addresses involved with the 2K storage system (2000 to 3999) shows that they all have one thing in common, a B-bit. Because there is no B-bit output from the storage address register on systems with less than 4K storage, the B-bit will be lost and a parity check error will result.

The same invalid addresses (2000 to 3999) are detected the same way on systems with 1.4K storage. In addition, check circuits to detect addresses between 1400 and 1999 are provided. It is accomplished by switching a 4-bit and an A-bit

together (1400 and 1799) or an 8-bit and an A-bit together (1800 to 1999) at hundreds time. Thus it can be seen that any invalid address will stop the machine.

Wrap - Around Checking

Wrap - around describes the process of modifying the highest address of a system by plus one or the lowest address by minus one. For example, on a 1.4K storage system, modifying 1399 by a plus one to obtain 0000 or modifying 0000 by minus one to obtain 1399 is referred to as wrap-around.

If wrap-around is detected at any time other than a storage scan operation or a clear operation, it will set the storage-address register check latch which will stop the machine at the end of the next cycle.

Wrap-around is detected by switching a C,A,8, 2-bit with modify by minus one and tens borrow or carry (this detects 0000 by minus 1). Also, for 1.4K storage capacity systems, by switching an A, 2, 1-bit combination with modify by plus one and tens borrow or carry (1399 + 1).

1402 Checks:

Read Hole-Count Check

The read hole-count check detects any possible reading errors that have occurred in the 1402 Card Read Punch.

The hole-count check is a comparison of the holes read, in each card column, at the first reading station against the holes read from each card column, of the same card, at the second reading station. If the comparison is unequal, a read check error will be signaled and the 1401 stopped.

The hole-count check is performed in four check planes consisting of 80 cores each. The check for any one card is performed in two of these check planes.

The check plane cores are read into on a read scan. Each check plane core is addressed serially by the storage-address register, simultaneously with the cores in the read area of main storage and the read row-bit cores.

Punch Hole-Count Check

The punch hole-count check is to detect any possible punching errors that may have occurred in the 1402 Card Read Punch. The hole-count check is a comparison of the holes set up to punch in each card column at the punch station against the holes read from each card column, of the same card, at the punch check station. If the comparison is unequal, a punch check error will be signaled and the 1401 and 1402 stopped.

The hole-count check is performed in four check planes consisting of 80 cores each. The check for any one card is performed in two of these check planes.

The check plane cores are read into on a punch scan. Each check plane core is addressed serially by the storage-address register, simultaneously with the cores in the punch area of main storage and the punch check row-bit cores. With this addressing arrangement, each check plane core is associated with a card column. If location 101 in main storage is being addressed on a punch scan, core 1 in each check plane and core 1 in the punch check row-bit core plane will also be addressed.

The upper check plane cores record when the first hole is set up to be punched in each card column or when the first hole is read from each card column at the punch check brushes. The lower check plane cores keep an odd or even count of holes set up to punch in each card column or holes read from each card column at the punch check brushes.

Read Validity

As the information to the 1401 storage is gated on to the inhibit lines, it is checked to insure that all characters being sent to storage are valid characters. Any invalid character will end up in storage;however, the validity check circuits will cause the 1401 and the card reader to stop at the completion of the card read operation and turn on the validity check light in the card reader.

Feeding Failure

Card feeding failures are checked for by card levers within the 1402 system. Card jams activate a line to the process unit called RD JAM or PUNCH JAM which block the read and punch clutch triggers and therefore prevent another clutch operation of the unit until the jam is cleared.

The jam lines are also activated when the clutch receives an impulse but fails to cause a feed cycle.

Punch Parity

The information to be punched is parity-checked in the B-register. An even-bit configuration will turn on the punch error stop latch in addition to the B-register check latch. The main reasons for the additional check is to provide circuits for the following:

1. Stop the 1402 at the end of that punch cycle.
2. Light the punch indicator on the 1401 console to indicate an error during punching.
3. Indicate in which phase of a combination op the parity error occurs.

1403 Checks:

Parity Check

Parity Checking is accomplished as the information to be printed is read from storage into the B-register. An even-bit configuration turns on the B-register check latch as described in B-register. The machine will stop at the end of the print operation.

Type Synchronization Check

This check insures that the 1401 is in synchronism with the print chain on the 1403. The home trigger, which is turned on when the character 1 on the print chain is in position to be printed by the number 1 hammer, is checked against the PSS ring and the print compare counter to insure that they are in synchronism. Any discrepancy during a print operation will turn on the Home-Error-Trigger and stop the machine with the sync-check light on. If the error is detected prior to the first subscan, the machine will be stopped before the actual print operation. Otherwise, the machine will print the line of information, even though it may be incorrect, before the system is stopped. An important point to notice is that the Home-Error-Trigger will not turn on unless a print operation is taking place.

When the system is stopped because of a home error, the following lights will be turned on:

1. PRINTER on the 1401 console
2. SYNC CHECK on the 1403

Hammer Fire - Print-Compare Check

This circuit checks for two conditions:

1. The hammer failed to fire when it was told to. (Because it had already fired once during this print operation, or because of a failure to receive a fire impulse.)

2. The hammer fired when it should not have fired. (Received an erroneous impulse.)

An error caused by one of these failure stops the machine at the end of the print operation in which it is detected. The PRINTER light on the 1401 console and the PRINT-CHECK light on the 1403 will be turned on.

At the same time that the failure is detected, a core, corresponding to the address being printed at the time of the failure, is set in the print error storage. This core remembers the location of the error and will indicate it on a storage scan operation.

The actual hammer fire-print compare failure is detected by the use of two additional core storage planes (planes 10 and 11) which are addressed simultaneously with main storage. One core, the hammer fire core, is reset to 1 each time that the particular location is addressed; if the hammer fires, it is set back to zero. The other core, called the print-compare core, is reset to zero each time that that location is addressed provided there is no print compare equal for that cycle. A print-compare-equal sets it to 1. Thus, it can be seen that on a no-print-compare cycle, the hammer plane core is reset to 1 and remains there while the print compare core is reset to 0. On a print-compare cycle the hammer plane core is reset to 1 and then set to 0 with the hammer response signal which indicates that the hammer has fired, while the print compare core is set to 1 by the print-compare equal. In either case, the cores were opposite when they were tested (the test occurs at read-out time when the location is addressed during the next print scan).

If, when tested, and it is tested on every print scan except the first one, both core sense lines are alike, an error is indicated. That means that a 1 in both cores, which indicates that the hammer failed to fire, or a 0 in both cores, which indicates that the hammer received an erroneous fire impulse, turns on the print check latch.

Print Line Complete

In addition to the print-compare cores, the hammer-fire cores, and the error-storage cores, which check for various print errors; another core plane, called print-line-complete, is provided to determine whether each storage position containing printable information actually had an opportunity to print. If a storage location containing information to be printed does not get a print-compare equal some time during the print operation, the print check latch is turned on, and the machine stops at the end of the operation with a print check error indicated. As in the case of hammer-fire failures, a core is set in the error storage plane corresponding to the particular storage location involved. A storage-scan operation, following the error stop, will indicate the address involved in the failure.

Each individual core in the print line complete plane corresponds to an individual storage location used for printing. At the beginning of a print operation, they all contain logical zeros. At the end of the print operation, (print scan 49) when the Print Line Complete cores are tested, they all should contain ones unless a print line complete check error has occurred.

The core for any position is set to 1 as follows:

1. A blank or an ineligible character (one not available on the chain).
2. A print compare signal.

The blank (C bit only) or ineligible character (any character with an 8, 4, and 1 or 8, 4 and 2-bit conbination) sets the core to 1 for that position during the first print scan. It will be regenerated on the following print scans and be retained in the core until after they are tested (print scan 49). A print-compare signal sets the core to 1 and signifies that that particular location has had an opportunity to print. Summing it up: Each core is immediately set to 1 unless it has a printable character. The remaining cores are set to 1 whenever the printable character gets a chance to print (print compare). Any core not set to 1 at the end of the print operation indicates that that position had a character to print but did not have an opportunity to do so as indicated by the failure to receive a print compare signal.

The print line complete circuitry also checks for a print-compare signal in conjunction with a blank or an ineligible character during print scans one through 48; such a condition sets an error-storage core to 1 and turns on the print-check latch.

Print Error Storage

The print error storage area is provided to remember in which position an error has occurred. These cores correspond to, and are addressed simultaneously with, the print area core of main storage. They are set to one during any print scan, except print scan 1, the 1401 will stop at the completion of the print operation with the PRINTER light on the console and the PRINT-CHECK light on the printer turned on (assuming that the I/O check stop switch is on).

The actual setting of the error storage cores is accomplished by activating the inhibit line for plane 12. This line is activated by the following:

1. By sensing a 1 in the hammer-fire core plane (plane 10) along with a 1 in the print-compare core plane (plane 11).
2. By sensing a 0 in the hammer-fire core plane along with a 0 in the print-compare core plane.
3. By sensing a PL complete check error.
4. By sensing a 1 in the error storage core. This regenerates the 1 once the core has been set.

The core that is set to 1 causes the system to stop at that location on a storage scan operation. This provides an indication as to the location that is involved in the failure.

The cores are reset to 0 on print scan 1. They can be set to 1 on any of the remaining 48 print scans. If a core is set to 1, the 1 will regenerate during the remaining print scans. It will not be reset to 0 again until the first print scan of a print operation.

Effects of Error on System - Method of Reset

Process Unit

Figure 5 shows the various error conditions in the process unit. It shows the lights that are on, etc. The conditions shown occur when the machine is in RUN mode and not involved with an input-output operation.

Reader Punch Unit

Figure 6 lists the various error conditions relative to the reader/punch unit. Notice that the I/O check stop switch is on.

When the I/O check stop is off, there are two additional methods to reset the error latches:

1. The I/O check reset switch located on the auxiliary console (I/O check stop switch must be off).
2. A branch code, B (III) (d), that checks for a read or punch error.

Printer

An error in the print operation will cause the 1401 to stop upon completion of the operation provided the I/O check stop switch on the 1401 console is on stop. Figure 7 lists the various error conditions and their effects.

If the I/O check stop switch is off, the various error lights will turn on but the system will not stop except for home errors. Under this condition, the error latches can be reset as follows:

1. By executing a B (III) (+).
2. By operating the I/O check reset key on the aux console.

Process Unit Error Conditions

Unit	Type of Error	Latch Involved	Machine Stops (Process ck stops "on")	Storage Adr Reg Contains	Lights On When Stopped	Reset by	Remarks
A Reg	Parity	A Reg Chk latch	End of Next Cycle (B cycle)	"B" Addr	Process A Reg chk Reset	Check Reset Key	Contents of A Reg at the time of error will be still on display.
B Reg	Parity	B Reg chk latch	End of Cycle in which error is detected	Address of location that was read into "B" register	Process B Reg chk Reset	Check Reset Key	Contents of B Reg at the time the error is detected will remain on display in B register.
Arith	Validity	Arith chk latch	End of following cycle	Normally 1 less than the location that the resultant is in.	Process Logic Chk Reset	Check Reset Key	Adr Reg will indicate one less than the location that the resultant is read into except for: 1. When the error is detected in the last cycle of the first forward scan on a recomplement operation when it will indicate the same location or 2. It will indicate one more than the location the resultant is read into on a reverse scan operation. The bit combination which causes the error will be in the storage unit and not on display under "Logic" (remember that it is in qui-binary form when checked and goes through the translator before going into storage.)

(continued on next page)

(cont'd)

Unit	Type of Error	Latch Involved	Machine Stops (Process ck stops "ON")	Storage Adr Reg Contains	Lights on when stopped	Reset by	Remarks
Inhibit Switching	Parity	Inhibit Check Latch	End of following cycle	Dependent upon operation being performed	Process Storage Chk Reset	Check Reset Key	This type error indicates that an even bit configuration has been read into storage.
Op Reg	Validity and Parity	Op Reg Check	End of cycle in which error is detected	Dependent upon type of operation being performed and phase	Process Op Reg Chk Reset	Check Reset Key	The check latch will not turn on during I Op Cycle
Storage Address Register	Parity and Validity	Stor Addr Reg Error Latch	End of cycle in which error is detected	Bit combination that caused error	Process Storage Addr Chk Reset	Check Reset Key	The error check is made after the address is serialized. An error could be caused by a fault in serializing.
	Wrap Around	Stor Addr Reg Error Latch	End of following cycle	Dependent upon operation being performed and modification	Process Storage Address Key Check reset	Check Reset	Can be modified by +1 or by -1.

NOTE: If any of the above errors occur during an input/output operation, the system will complete the particular operation involved before stopping.

FIGURE 5 Process Unit Error Conditions

Unit	Error	Latch Involved	Machine Stops I/O Check Switch "On"	Lights On When Stopped	Reset by	Remarks
Reader	Read Check	Read Check	At the end of the feed cycle	Read Check (1402) Read (Process)	Check Reset Key on the 1402	Cards must be run out before check reset key becomes effective.
Reader	Validity	Validity	"	Validity (1402) Read (Process	Check Reset Key on the 1402	Cards must be run out before the check reset key is effective.
	NOTE: Also, if the invalid combination causes incorrect parity:					
				Storage (Process) Process (Process) Check Reset (Process)	Check Reset on the Process Unit	
Reader	Jam	Picks R:4	At the end of the feed cycle	Reader Stop (1402) Read (Process Unit)	Check Reset Key (1402)	Cards must be run out before the reset key is effective.
Punch	Punch Check	Punch Check	At the end of the feed cycle	Punch Check (1402) Punch (Process)	Check Reset Key (1402)	"
Punch	Parity	Punch Stop B Reg Check	"	Punch (Process) Progress (") B Reg (") Check Reset (Process)	Check Reset Key (1401)	
Punch	Jam	R-31	"	Punch Stop (1402) Punch (Process	Check Reset Key (1402)	Cards must be run out before the reset key is effective.

FIGURE 6 IBM 1402 Error Conditions

Error	Latch	1401 Stops (I/O Chk Stop Switch) On	Lights On: Process Unit	Lights On: Printer	Reset By	Remarks
Parity	B Reg Check	Upon completion of Print Operation	B Reg Process		Check Reset (Process out)	
Type Sync	Home Error	"	Printer	Sync Check	Check Reset (Printer)	
Hammer Fire	Print Check	"	Printer	Print Check	Same	Sets Error Storage
Print Line Complete	"	"	"	"	"	"

FIGURE 7 IBM 1403 Error Conditions