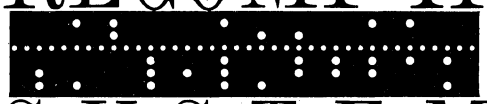
RECOMP II

SYSTEM SERVICE MANUAL

Autonetics Industrial Products
A DIVISION OF NORTH AMERICAN AVIATION, INC. 3584 WILSHIRE BOULEVARD, LOS ANGELES 5, CALIFORNIA

RECOMP II

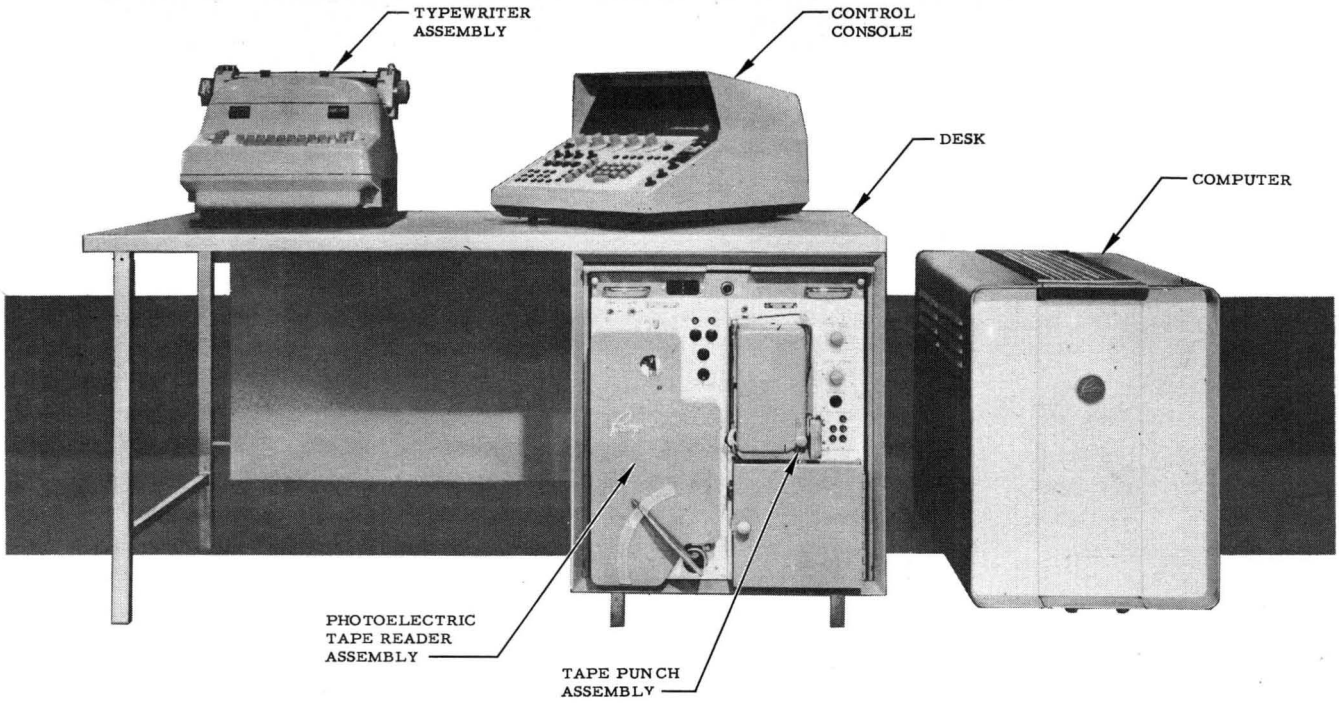


SYSTEM SERVICE MANUAL

1 FEBRUARY 1960

Publication 508-T-6

Autonetics Industrial Products
A DIVISION OF NORTH AMERICAN AVIATION, INC. 3584 WILSHIRE BOULEVARD, LOS ANGELES 5, CALIFORNIA



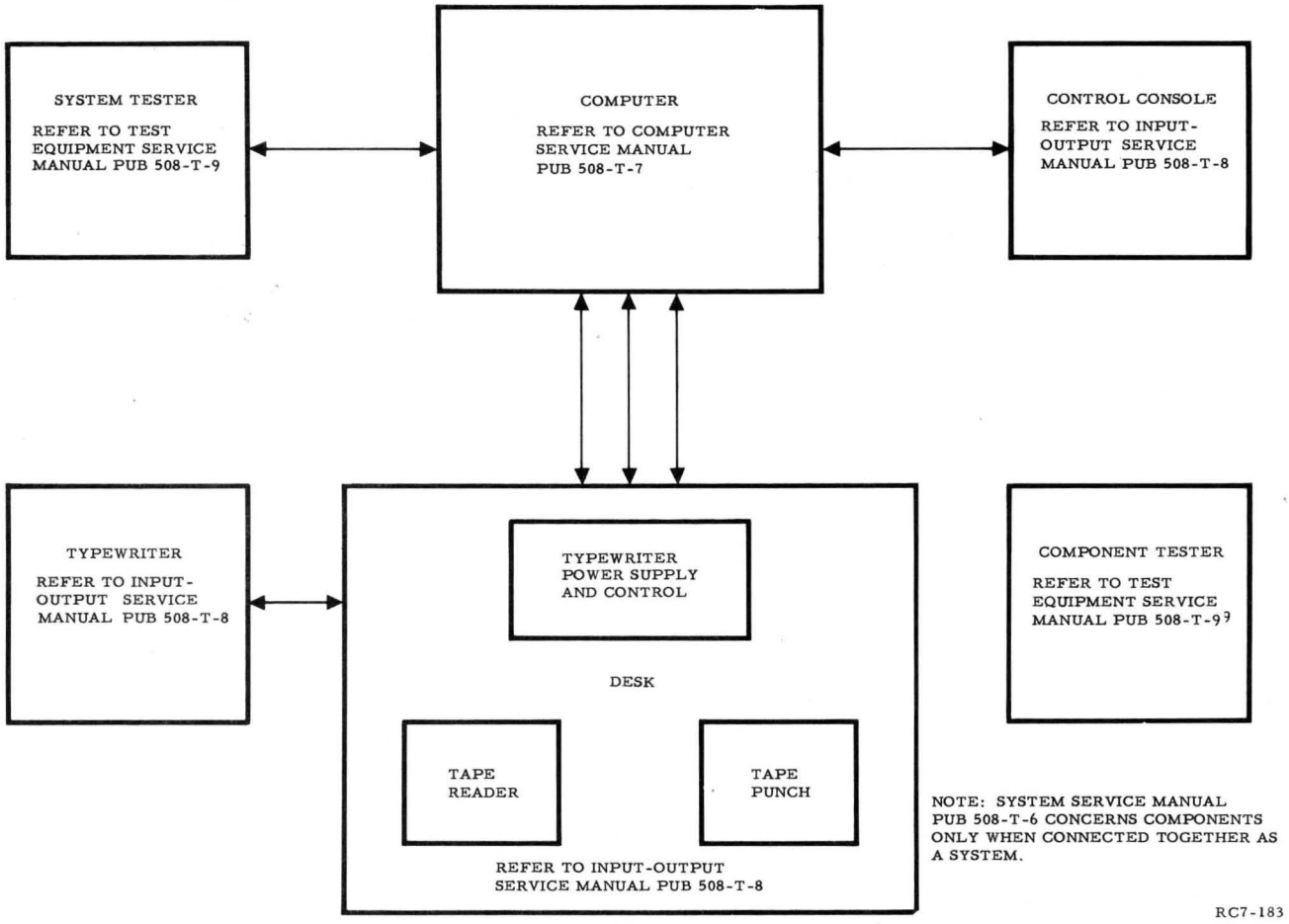


Figure 1. RECOMP II System Block Diagram

PREFACE

This manual is one of a set of five published by Autonetics to provide maintenance information on the RECOMP II computer system.

System Service Manual: Publication No. 508-T-6
Computer Service Manual: Publication No. 508-T-7
Input-Output Service Manual: Publication No. 508-T-8
Test Equipment Service Manual: Publication No. 508-T-9
System Reference Schematics: Publication No. 508-T-11

The block diagram of figure 1 illustrates each system component and shows its applicable service manual. In brief, the scope of each RECOMP II service manual is as follows:

System Service Manual: This manual describes the general concept of RECOMP II maintenance and provides instructions for checkout of the computer system using the system tester in both static and dynamic test modes. The goal of these system test procedures is to isolate malfunctions to a specific operational area or system component.

Computer Service Manual: This manual describes the operational components located in the computer assembly, including memory unit, power circuits, and signal circuits. It also provides maintenance instructions and adjustments for computer components, and gives a detailed set of test procedures for computer circuit boards using the component tester.

Input-Output Service Manual: This manual describes the operational characteristics of RECOMP II input-output equipment (control console, typewriter, tape reader, and tape punch). It also provides maintenance and test instructions for these input-output devices and the associated desk assembly.

Test Equipment Service Manual: This manual describes the functional characteristics of the system tester and component tester. It also provides maintenance and test information for these two RECOMP II test equipments.

System Reference Schematics: This publication provides a set of schematics, assembly drawings, wiring charts, and signal charts for each system component; i.e., computer, desk, control console, tape reader, typewriter, and tape punch.

CONTENTS

ILLUSTRATIONS

ILLUSTRATIONS (Continued)

TABLES

This page intentionally left blank

INTRODUCTION

Computing system maintenance comprises two general types: system maintenance and component maintenance. System maintenance is primarily concerned with checking out the operational status of each functional system component in terms of system operation. Component maintenance principally consists of preventive maintenance, and replacement of malfunctioning electronic and electromechanical parts of the system. The procedures of system maintenance are thus primarily trouble-isolation procedures. That is, whenever faulty operation occurs, the trouble is traced to a particular system component by using the system checkout procedures. If, for example, the trouble should be in the main computer circuitry, the trouble is traced to a particular functional circuit or group of circuits within the computer. Once the trouble has been identified with a particular functional area of the computer system, the technician refers to the applicable component service manual for (1) more specific maintenance procedures to isolate trouble to particular electrical or mechanical components and (2) instructions to repair or replace the defective components. In this system service manual, system maintenance is divided into three phases: Preliminary Checkout, Operational Checkout, and Logic Checkout.

PRELIMINARY CHECKOUT

Preliminary checkout procedures provide instructions for checking electrical and operational conditions that must be satisfied before the more complex operational steps can be performed. For example, all d-c power supply voltage levels and load characteristics are checked in preliminary checkout. The ability of the control console to enter information and to read out information from the computer is also checked. All preliminary checkout procedures are limited to those that can be performed relatively quickly and without referring to logic interpretation.

OPERATIONAL CHECKOUT

Operational checkout procedures provide the means for checking out the ability of the computer to perform various commands and functions. Short routines are entered from the control console (or the tape reader or typewriter) and the computer then is set to perform these

routines. Results are monitored on the visual readout of the control console and/or an oscilloscope. If the routine fails, the computer has incorrectly executed a command or malfunctioned in another manner. The maintenance engineer can then, through use of the logic descriptions, determine the circuits involved in the malfunction.

LOGIC CHECKOUT

Logic checkout involves determination of the characteristics of each function in terms of the logic equations which represent the detailed operational design of the computer. From this, the maintenance engineer can then readily determine the circuits and components involved. In most cases, logic checkout procedures isolate trouble to particular circuit boards, which can then be checked out in detail on the component tester and replaced.

MAINTENANCE AIDS

The RECOMP II system (see figure 1) has numerous facilities which aid in checking out the system, troubleshooting, and performing preventive maintenance. Most important of these is the System Tester, a separate self-contained unit which provides facilities for testing and monitoring the computing system in both dynamic and static modes of operation. This unit is especially useful in conducting preventive maintenance tests under marginal conditions. Other system checkout aids--all integral parts of the system--are the control console, tape reader, tape punch, and electric typewriter. All of these units are used both with and without the system tester, as checkout and troubleshooting may require.

The control console contains virtually all of the operating switches and indicators used in directing the activities of the computing system. Consequently, it is employed extensively in almost all checkout and troubleshooting procedures. The control console keyboard is used to set up memory locations and to enter commands and data into the computer. Readout switches and the visual readout panel are utilized in checking the contents of registers and memory locations. Error indicators, memory location indicators, and several operating switches aid in determining the exact location of a malfunction. The control console also contains several other checkout and troubleshooting facilities. These items, located on the rear of the console, consist of (1) power supply voltmeter and range switch which facilitates checking the secondary d-c voltages and (2) memory and register test jacks which permit direct connection of an oscilloscope to check memory and register contents. Procedures in which the control console is used in checkout, troubleshooting, and preventive maintenance of the RECOMP II system are presented later. All persons associated with maintenance of the RECOMP II system should be thoroughly familiar with the functions of all console controls and indicators before performing any work on the system. A description of the controls and indicators and their functions is given in the RECOMP II Operating Manual.

Principal use of the tape reader, tape punch, and typewriter in checkout, troubleshooting, and preventive maintenance is in the insertion and recording of data. Use of these units in this work is presented later. Operating procedures for these units and their controls are given in the Operating Manual.

Uses of the system tester are given later in this section. System tester controls are described in the Test Equipment Service Manual. Persons utilizing the system tester should completely understand its various modes of operation before using it.

CONTROL CONSOLE

Most of the controls for directing computer activities during checkout, trouble-shooting, and preventive maintenance are located on the control console (see figure 2).

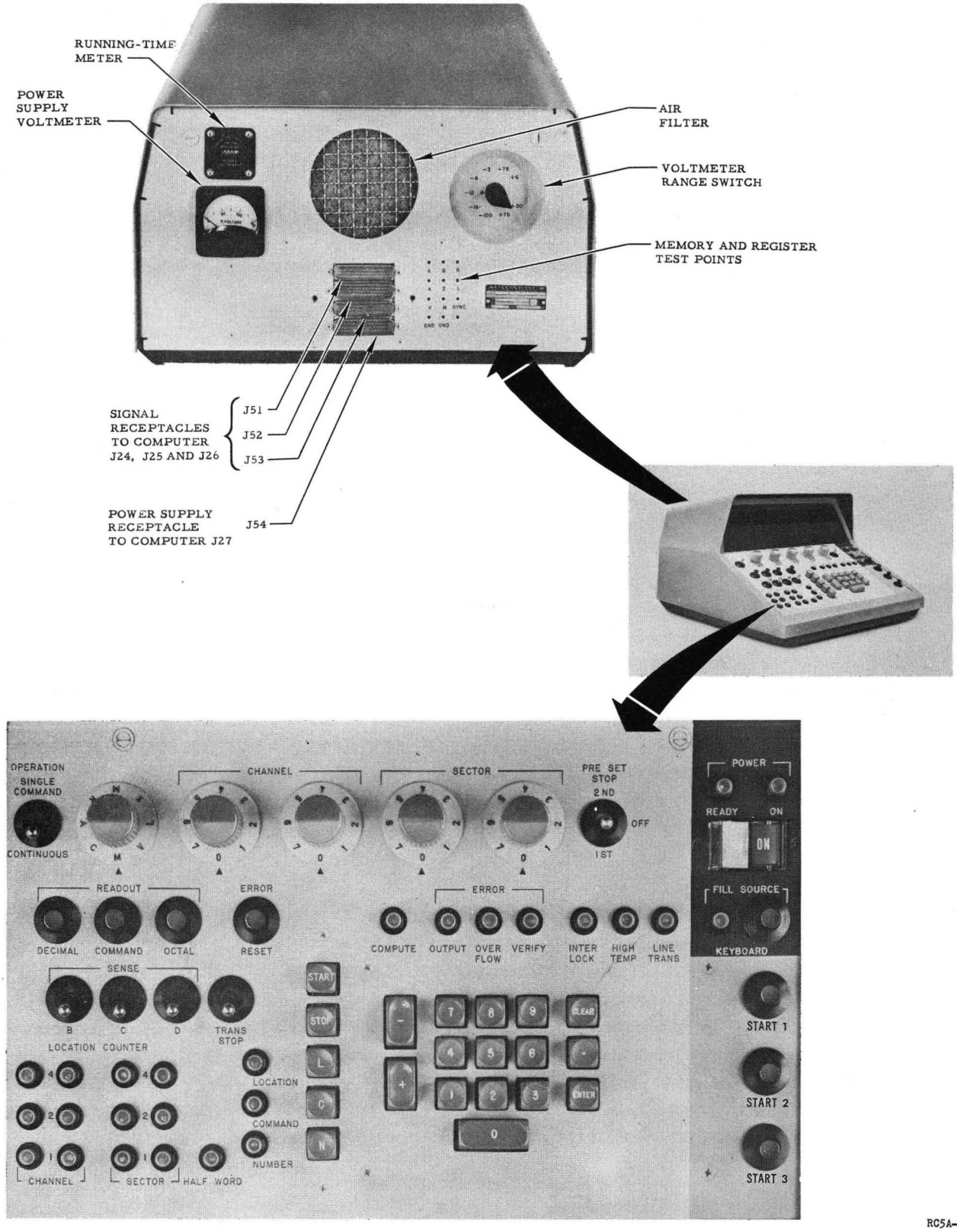
Control of Internal Operations

Principal use of the control console is in the control of computer internal operations. In addition to the controls for starting, stopping, and applying power to the computer, the console contains the switches and indicators used in directing the system in the various modes of operation. The Start switches can be used in several different ways during checkout and maintenance work. The regular Start switch is utilized (1) to start the computer to execute instructions, as when running a test routine, and (2) in conjunction with the single command operation switch to cause the computer to execute one instruction at a time and stop automatically. The single-command operating mode is especially helpful when isolating a malfunction. If the computer is not executing a test program correctly, the instruction causing the difficulty can be determined by displaying the contents of registers and memory locations on the visual readout following execution of each command.

Start 1, Start 2, and Start 3 switches start the computer on execution of a sequence of instructions beginning at memory location 0001, 0002, and 0003, respectively. They are used mainly with some test routines having the initial command in one of these memory locations. All Start switches initiate the Compute state which is indicated by the Compute neon indicator in the center of the console.

Four possible methods of stopping the computer provide additional flexibility in controlling internal operations during checkout and maintenance. These are the regular Stop button, the Transfer Stop switch, the Pre-set Stop switch, and the Error Reset button.

The Transfer Stop and Pre-set Stop switches are of particular value in locating malfunctions. When placed in the Up position, the Transfer Stop switch halts computer operation after interpretation of any transfer instruction even though the condition necessary for transfer of



RC5A-110

Figure 2. Control Console Front View

control was not met. The location of the command containing the transfer instruction is placed in the right half of the X-register from where it can be displayed on the console visual radout. This enables checking if the location of a transfer instruction in a test program is incorrect because of improper sequence or a memory read-write malfunction. The Pre-Set Stop switch, when placed in the 1st or 2nd position, stops the computer after execution of the command in the left or right half-word position, respectively, of the full-word location as set on the Channel and Sector Selectors. This permits running a test routine until it reaches the location in which the malfunction is known or believed to exist. Contents of the memory location of the last instruction executed may then be displayed on the visual readout. The location counter will display the location of the next instruction to be executed.

Other switches and indicators used in controlling internal operations include the Sense switches, Error indicators and Error Reset button, and the Fill Source button. The three Sense switches operate in conjunction with the Conditional Transfer instructions TSB, TSC, and TSD to enable manual direction of the computer along various routes of a stored program.

Error Reset permits resumption of operation following a stop caused by an output, overflow, or verify error. However, the Error indicators, are of greater aid than the Error Reset button in checkout and maintenance. The Verify Error neon indicates a discrepancy between the contents of a tape and the information originally stored in memory from that tape. Usually a Verify error indicates faulty recording of the information in memory. It may also, however, serve as a means of checking the accuracy of the tape reading process if known tape content is read in and then displayed from memory on the console visual readout. The Verify mode of operation (see Operating manual) is helpful in locating a memory write malfunction because the location counter will indicate the next memory address that is to be verified; i. e., the malfunction is at the address which is one sector less than shown by the location counter. An output error is indicated when a difference occurs between the code sent from the computer and that typed. By comparing the information at the memory location indicated by the location counter with that shown on the output register on the punch, the character being typed correctly can be determined. The Overflow Error indicator lights whenever a number exceeds the capacity of the A-register (see Operating manual). An overflow condition usually indicates an error in the program but may also signify improper command execution. All three error lights are extinguished by depressing the Error Reset button.

Environmental conditions affecting proper computer operation are indicated by three warning neon indicator lights. These conditions are open circuit panels, indicated by the Interlock neon; excessively high internal temperatures, indicated by the High Temperature neon; and a line voltage drop or increase, indicated by the Line Transient neon. The computer may be operated with one or both hinged door circuit panels open; however, internal operating temperatures may become sufficiently high to cause the computer to halt. A high temperature indication will precede a halt. Excessively high temperature inside the computer may turn off the computer almost immediately or after a long delay, depending on the rate of temperature rise. The abnormally high temperature may have originated in the power supply or memory. A line voltage drop below 108 volts or above 130 volts will cause the Line Transient neon to light. Usually a line voltage change sufficient to light the indicator will be of concern only to the operator. This is because and execessive voltage change during computing may alter information in the registers or rapid access loops. Information in main memory is unaffected by line voltage variations other than high transients.

Main use of the Fill Source button in checkout and maintenance is to set up the computer to receive instructions and data from the console keyboard. The Fill Source indicator lights when the Fill Source button is depressed. It turns off automatically when the Fill switch on the tape reader or typewriter is depressed, when an input command is executed, or when the computer enters the Compute state.

Isolation of Malfunctions

The location counter indicator in the lower left corner of the console panel is used extensively in maintenance work. The location counter indicator consists of 13 neon indicators and displays: (1) the address of the location in which the data is being stored or selected for comparison in the fill or verify mode (2) the address of the location of the next instruction to be executed when operating in the compute state. Instances in which the location counter indicator can aid in isolating a malfunction include: (1) location of the instruction on which a command execution malfunction occurred, (2) location of the instruction to which a program will transfer, (3) visual verification of the location in which the information is being placed or verified, and (4) visual observation of memory access and program execution sequences during single-step operation.

Input of Commands and Data

The control panel keyboard is the principal medium for entering commands and data during checkout and preventive maintenance. It is also used for setting the location counter. All types of information except alphanumeric can be entered from the console keyboard. Maintenance work in which the keyboard is utilized include: (1) modification of test routines to alter the course of operations, (2) entry of test data into memory, (3) insertion of commands to check correct execution, (4) entry of data into memory to ascertain proper functioning of output equipment, (5) entry of data in memory to verify accuracy of write and read processes in the channel in which a malfunction occurred, and (6) entry of information into channels when making memory and register gain and other adjustments. The switches identified by L, C, and N--and their indicators--are considered part of the keyboard. Their main use is to set the computer to receive numerical entries in location, command, or number format, respectively. Procedures for entering information in these formats are given later in this section.

Readout of Memory and Register Contents

The visual readout (see figure 3) is of considerable assistance in maintenance by enabling: (1) visual inspection of the information in main memory channels, rapid access loops, and registers, (2) verification of data before entry, (3) isolation of a malfunction in the write, read, and command execution processes, (4) display of results of a command execution or other operation, (5) a check of information entered or transmitted from the computer against that in memory, (6) a check on accuracy of the computations performed, and (7) determination of proper execution of transfer commands. Frequently, the visual readout is more helpful in isolating malfunctions when it is used with the single command operating mode. Commands and data entered from the keyboard are displayed on the visual readout automatically upon depression of a button. Display of information in the computer is achieved by execution of the Display command or manually. The latter is readily performed with a row of rotary memory and register location switches and three readout buttons. Contents of all main memory and rapid access loop locations and all registers except the B-register can be displayed in binary coded decimal, octal, and command formats. Procedures for obtaining readout manually are presented on pages 24 through 28 under Computer Control in the Operating manual, Pub. 512-E-3.

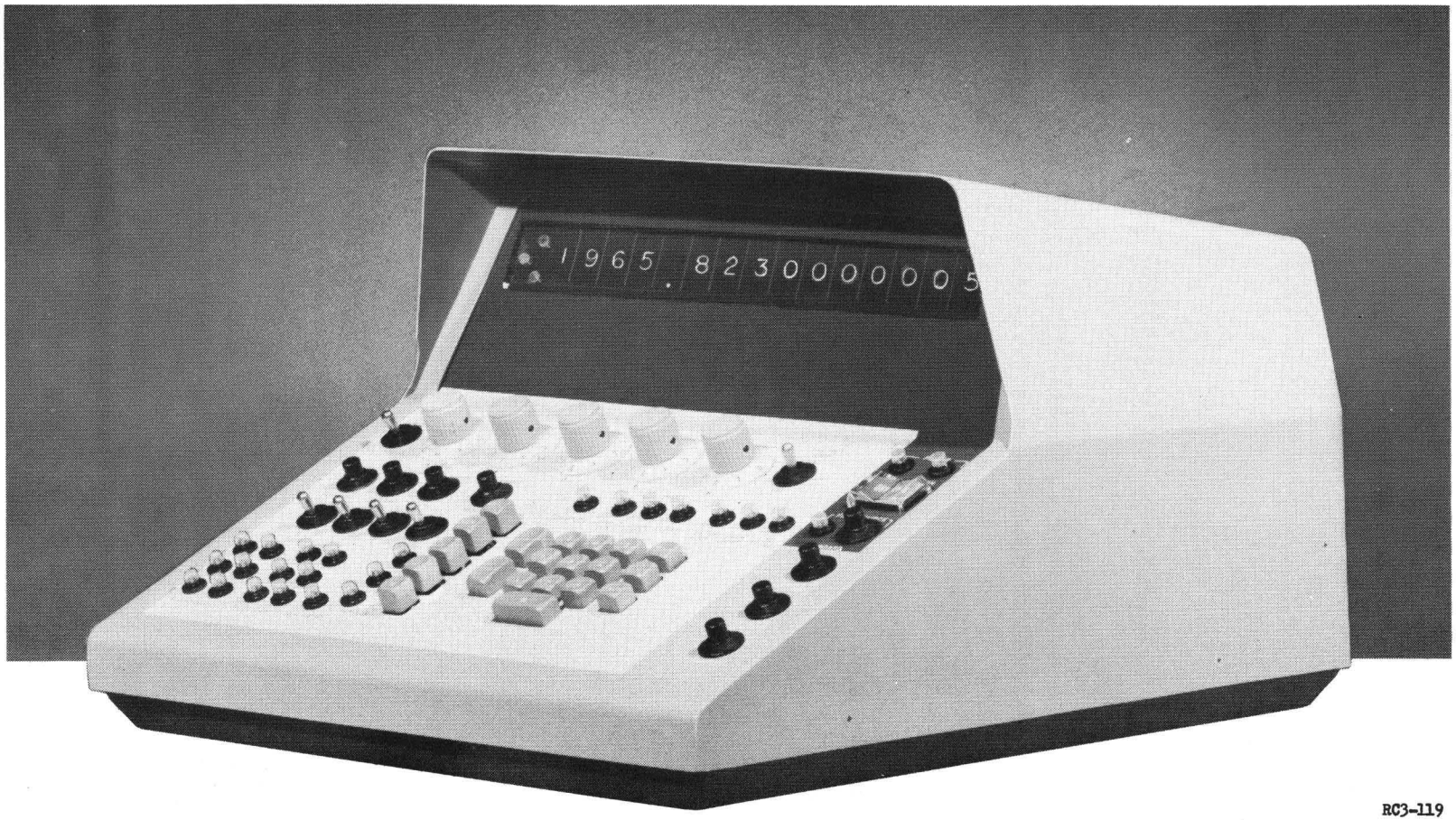


Figure 3. Visual Readout with Display of Numbers

Monitoring of Internal Operations

Facilities for monitoring internal computer operations on the rear of the console (see figure 4) consist of test receptacles and a power supply voltmeter range switch and percentage indicator. The test receptacles enable examination of the contents of registers, loops, and main memory locations with the use of an oscilloscope. This aids in determining if the contents of the memory or registers change, if the writing and reading processes are functioning properly, and if the signals are of proper amplitude . Contents of registers and loops can be displayed in continuous shape; contents of main memory locations can either be viewed as the contents pass through the circuitry or in continuous shape by placing the contents of a main memory location into a register or loop. A synchronization receptacle provides a synchronization signal for the oscilloscope.

All secondary d-c voltage levels of the computer and control console can be conveniently checked with the power supply voltmeter range switch and percentage indicator. The levels are tested by turning the range switch to each position and observing percentage reading on the indicator.

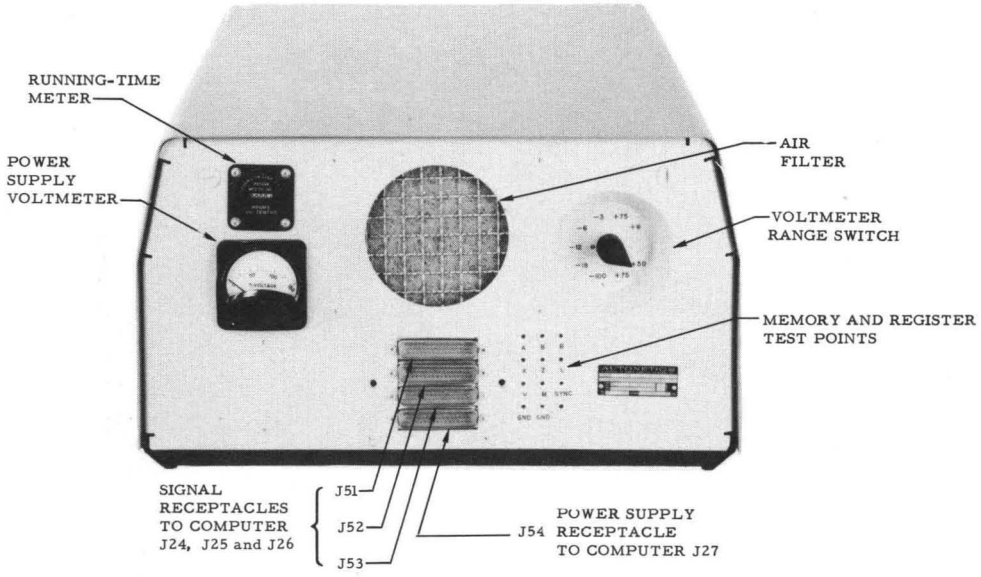


Figure 4. Rear of Console

PAPER TAPE READER

Use of the tape reader (see figure 5) in checkout and maintenance consists of entering test routines into the computer and verifying the recording of information into memory. The latter function is performed by comparing the contents of memory against that on the tape with which it had been entered. Verification thus serves as a rapid check of correct recording in memory. Operation of the tape reader in both the Fill and Verify modes is described later.

ELECTRIC TYPEWRITER

The electric typewriter (see figure 6) is especially useful in obtaining a printout of the contents of memory for comparison with information entered or for examination to detect program errors and machine malfunctions. The typewriter is also used frequently during checkout and trouble-shooting to enter information into the computer and, in conjunction with the paper tape punch, to prepare test tapes. Because information can be entered into the computer in the same format as from the tape reader and console keyboard, the typewriter can be utilized to enter commands and data when either of the other units is inoperative. When the tape reader is inoperative, the typewriter also is the only

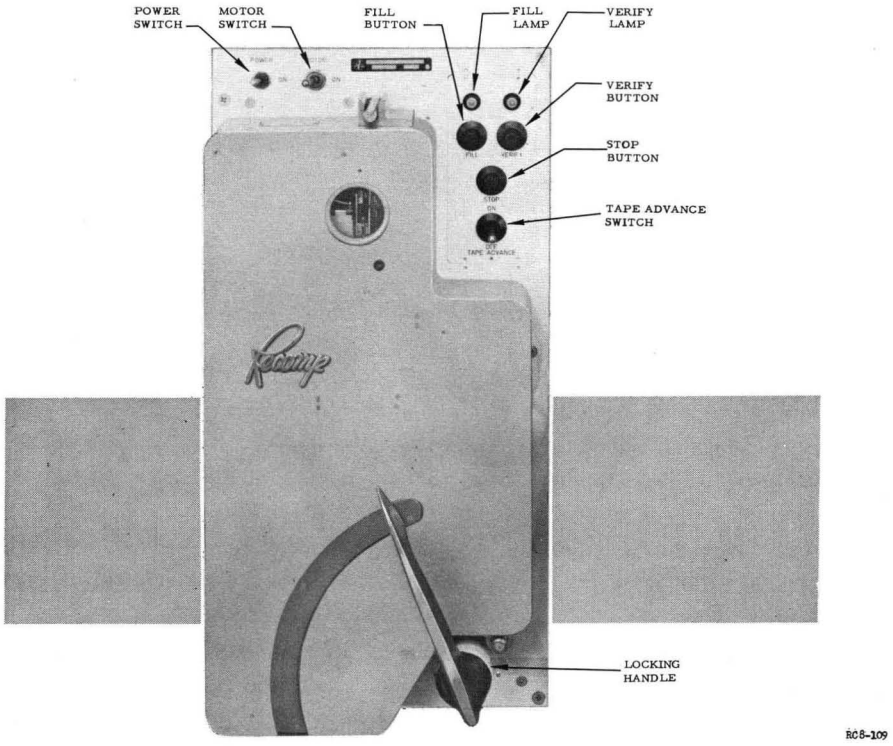


Figure 5. Tape Reader

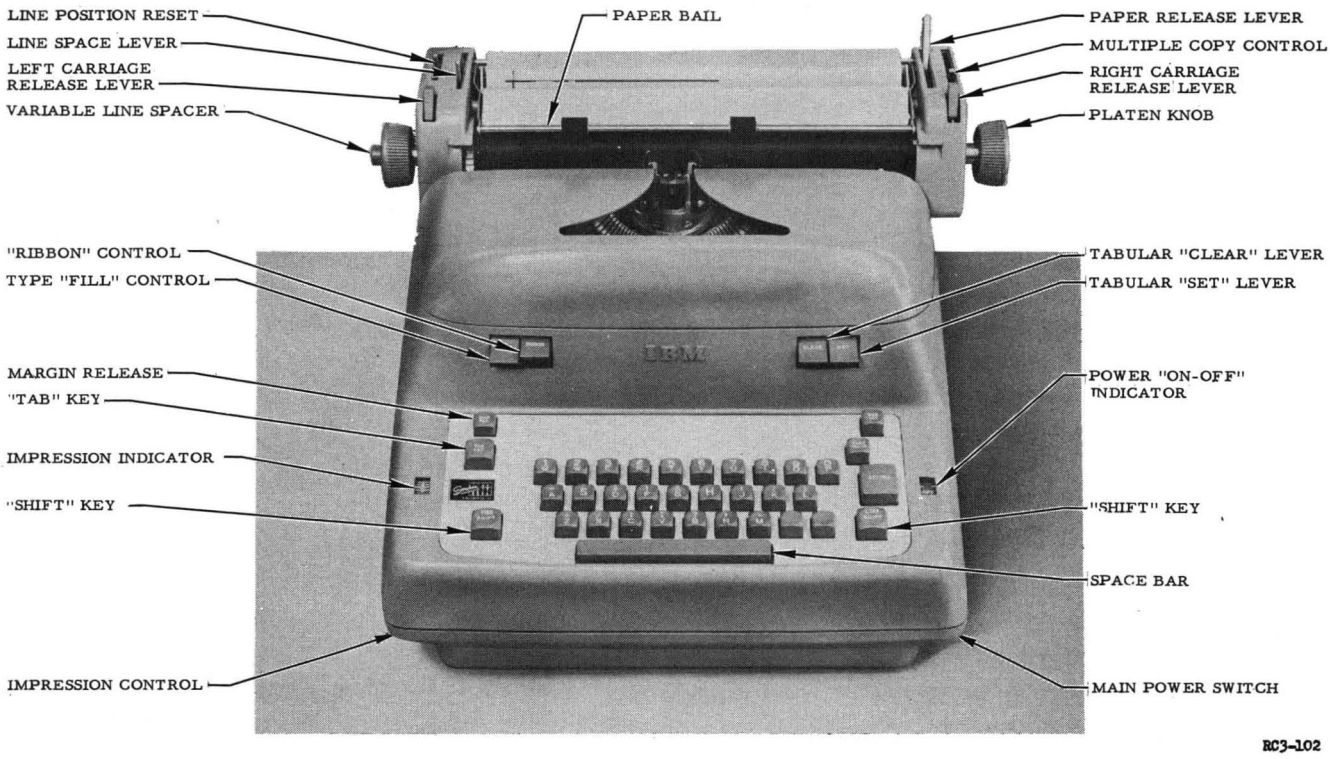


Figure 6. Typewriter Front View

means of entering or receiving alphanumeric information to or from the computer. Several typewriter control codes can also be used to direct computer operations. These control codes, procedures for entering information into the computer with the typewriter, and use of the typewriter in preparation of test tapes are presented later in this section. Typewriter output operation is controlled by program instructions TYC, TYW, PTC, and PTW. Use of these instructions is outlined in the Operating manual.

PAPER TAPE PUNCH

Main uses of the paper tape punch (see figure 7) in checkout and maintenance are the preparation of test tapes and the output of information when the typewriter and visual readout are malfunctioning. The punch is also used occasionally with the tape reader in test routines to check correct operation of the tape reader, tape punch, and typewriter. The output register, located on the punch control panel, aids in determining if the typewriter is recording correctly. If the Output Error indicator on the console is on, the output register will indicate the character mistyped. Comparison between the character displayed on the output register and that typed will help to isolate the malfunction. A check to ascertain if the register displays correctly can be made by comparing the display on the output register with the contents of the memory location being transferred to the punch. Loading of paper tape into the punch is described on the inside of the cover of the unit; use of the punch controls and punch instructions is explained later in this section. Computer controlled punch operation is governed by program instructions PNC, PNW, PTC, and PTW. Their use is described in the Operating manual.

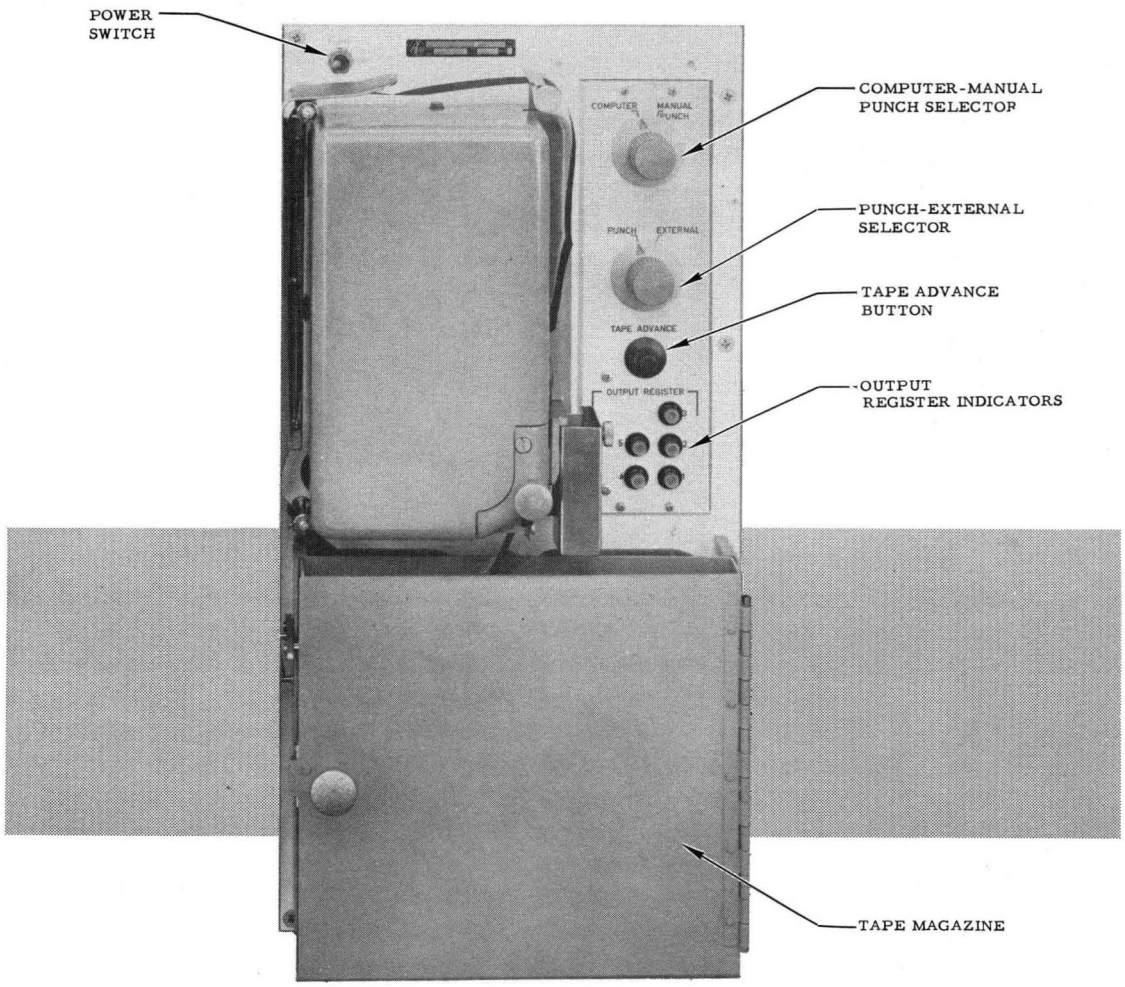


Figure 7. Tape Punch

SYSTEM TESTER

Uses of System Tester

Main purpose of the System Tester (see figure 8) is to simplify trouble-shooting. Specific uses include (1) monitoring the computer flip-flops, including the read and write amplifier flip-flops; (2) controlling the computer flip-flops to determine if each is functioning properly and to static-check the logic; (3) checking operation of gates to determine if operating properly marginally, or if inoperative; (4) controlling a number of flip-flops under dynamic conditions to aid in isolating a malfunction; and (5) applying marginal conditions to the computer to locate incipient malfunctions. Procedures for operating the system tester in these uses are given in the System Checkout Procedures section of this manual.

Uses of System Tester Control

System tester controls consist of two power switches, the flip-flop set-reset switches and indicators, a static-dynamic mode switch, a marginal check switch, and a one-shot clock switch. Except for the power switches, all are used to establish various conditions within the

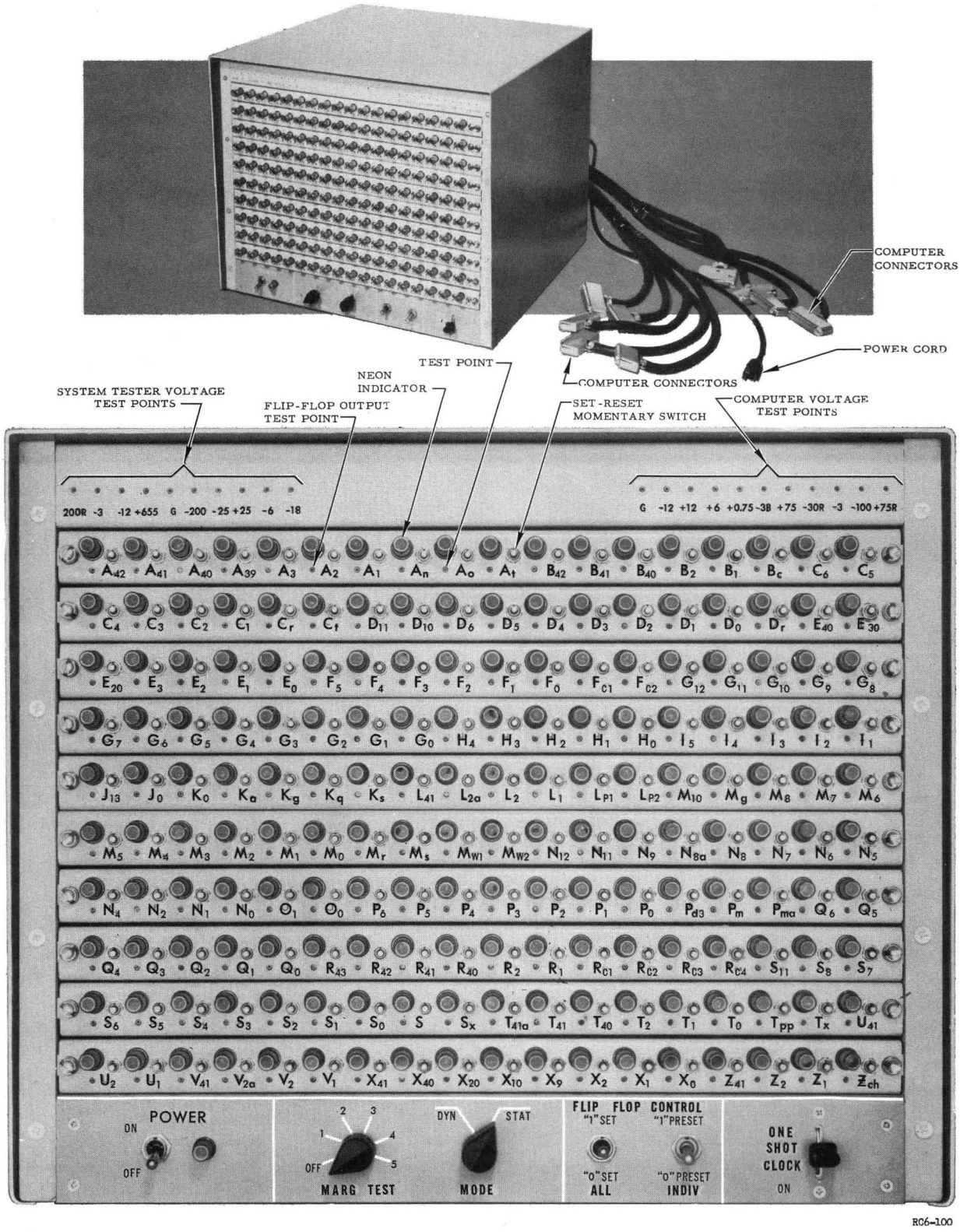


Figure 8. System Tester

tester that will cause the computer to react to desired external control rather than normal internal operation. Connecting the system tester does not, however, affect the normal dynamic characteristics of the computer nor alter the normal sequence of operations except when a change is initiated externally through the system tester buttons. Functions of these controls and the computer servicing operations in which they are used are given in the following paragraphs.

Application of Power

The power controls apply power to the tester. The Power switch applies 60-cycle power to the power supply and internal circuits of the tester and the Neon Power switch applies d-c power to the tester panel neons. A neon indicator above each of these switches indicates proper application of power within the tester.

Setting and Resetting of Flip-Flops

Set-Reset controls consist of a 1-Set All– 0-Set All switch, a Preset 1–Preset 0 switch, and individual microswitches for each flip-flop. These controls permit resetting the flip-flops (terms), including those in the read and write amplifiers, to 0 or 1 states. The 1-Set All–0-Set All switch enables setting all flip-flops simultaneously to one state or the other; the Preset 1–Preset 0 in conjunction with the individual microswitches enables setting the individual flip-flops to one state or the other. The Set-Reset controls are used in determining proper switching of the flip-flops and in locating malfunctioning flip-flops. Instances in which these controls are used for isolation of malfunction are (1) improper execution of commands, (2) weak read and write signals, (3) incorrect writing on and reading from memory, (4) incorrect memory location selection, and (5) broken wiring or wiring errors.

Checkout in Static or Dynamic Modes

The Static-Dynamic mode switch enables checking the logic through testing gates in the static mode and through observing indications of tester lights in the dynamic mode. In some procedures the Static-Dynamic switch is used in conjunction with the One-Shot Clock switch, in others with the Marginal Check switch.

With the Static-Dynamic switch in the Static position, any logic gate or gates set up can be triggered with the One-Shot Clock switch. The Static position turns off the computer clock and leaves it in the true state. When the One-Shot Clock switch is depressed, the tester forces the clock line to a false level by grounding it. This initiates a positive

step function suitable for triggering the flip-flops to statically check the logic gates one at a time. It is helpful in isolating almost every malfunction of an electronic nature because gates are used in all logic and many other operations, such as console keyboard entry and visual readout.

During the dynamic testing mode, normal computer operation is monitored by the system tester. When computation ceases, the states of all flip-flops are displayed. The malfunctions can then be isolated by the utilization of the computer logic with the necessary understanding of computer operation. The system tester does not effect the operation of the computer during normal operation.

Operation Under Marginal Conditions

Use of the Static-Dynamic switch in conjunction with the Marginal Check switch is of greater assistance in checkout and preventive maintenance than trouble-shooting. The Marginal Check switch can be set to change various voltage levels within the computer. These changed levels can be applied in either the static or dynamic mode according to the setting of the Static-Dynamic switch. The -6 and -18 voltages are lowered in either the static or dynamic mode; the +6 and -12 voltages are raised in the static mode and raised and lowered in the dynamic mode. When the marginal condition is imposed in the dynamic mode, the clock is also advanced and retarded on successive clock pulses. Two jitter flip-flops generate a square wave of one-forth the clock rate for modulating the +6 and -12 voltages. The frequency and the rise and fall times of these waves do not cause transient effects within the computer.

Imposing marginal conditions is especially useful in locating incipient malfunctions. Application of a marginal condition in the static mode permits testing of a specific component in the dynamic mode to continuous operation of the computer on a given program. Components and circuits which function satisfactorily under normal operating conditions will fail when marginal conditions are imposed; thus the static mode enables locating and replacing failing components before a malfunction occurs. Under marginal conditions in the dynamic mode, if the computer handles a program or routine correctly, it usually is operating satisfactorily. The marginal check is helpful in trouble-shooting through aiding in isolating a malfunction that is occurring intermittently.

Flip-flop indicators on the System Tester enable identification of a malfunctioning flip-flop. When a computer flip-flop is in the true state the appropriate neon is illuminated; when in the false state it is extinguished. There is one neon for each flip-flop, including each read and write amplifier flip-flop. Proper operation of the neons themselves can be easily confirmed by placing the 1-Set All-0-Set All switch in the 1-Set All position and noting if all neons are lighted.

The control unit network board mounts the marginal test condition potentiometers. This board is shown in Fig 9.

The adjustments available to the computer operator for establishing marginal test conditions are listed in Table 1. The marginal test conditions available in a computer are shown in Table 2. The voltage levels for marginal test conditions may be adjusted by the use of potentiometers located on the control unit network board. The designations of marginal tests voltage conditions and adjustment potentiometers are listed in Table 3.

Table 1. Adjustments Used To Establish Marginal Test Conditions

-18 v	Static margin = R12
-6 v	Static margin = R14
-12 v	Static margin = R5
+6 v	Static margin = R13
-12	Dynamic margin = R10
+6	Dynamic margin = R40

R12, R14, R5, R3, R10, R40 located on marginal test board of System Tester.

Clock Jitter width = R4 on clock board No. 1 in computer.

Table 2. Marginal Test Conditions

Switch Setting		Action				
Mode	Marginal Test	Voltage Change				Clock Jitter
S_1	S_2	-6	+6	-12	-18	
Static	OFF	None	None	None	None	None
Static	No. 1	None	None	None	Lowered by 10%	None
Static	No. 2	Lowered by 10%	None	None	Lowered by 10%	None
Static	No. 3	Lowered by 10%	None	Raised by 10%	Lowered by 10%	None
Static	No. 4	Lowered by 10%	Raised by 10%	Raised by 10%	Lowered by 10%	None
Static	No. 5	Lowered by 10%	Raised by 10%	Raised by 10%	Lowered by 10%	None
Dynamic	OFF	None	None	None	None	None
Dynamic	No. 1	None	None	None	Lowered b6 10%	None

Table 2. (Continued)

Switch Setting		Action				
Mode	Marginal Test	Voltage Change				Clock Jitter
S_1	S_2	-6	+6	-12	-18	
Dynamic	No. 2	Lowered by 10%	None	None	Lowered by 10%	None
Dynamic	No. 3	Lowered by 10%	None	Jitter ±10%	Lowered by 10%	None
Dynamic	No. 4	Lowered by 10%	Jitter ±10%	Jitter ±10%	Lowered by 10%	None
Dynamic	No. 5	Lowered by 10%	Jitter ±10%	Jitter ±10%	Lowered by 10%	Jitter ±0.5 usec

Table 3. Designations of Potentiometers Used for Establishing Marginal Test Conditions

Adjustment	Voltage	Remarks
R33	Line Trans	High Limit
R34	+50	Adjusted before adjustment of R47
R35	-12	
R36	±6	Adjust for mean value
R37	+0.75	
R38	-3	
R39	-18	
R40	-100	Adjustment dependent on voltage level
R42	Line Trans	Low Limit
R47	+75	Adjusted after adjustment of R34

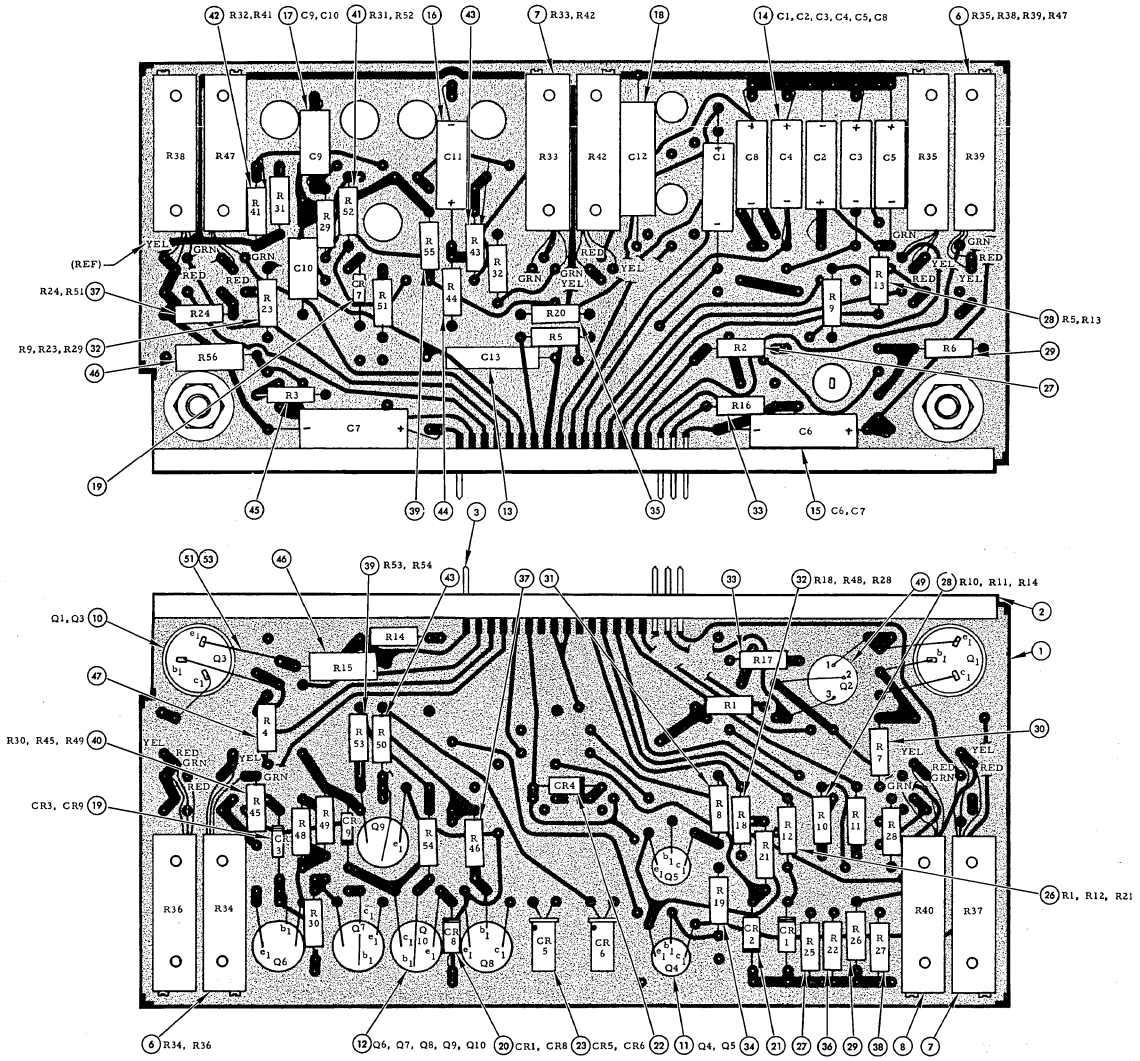


Figure 9. Control Unit Network Board

SYSTEM OPERATION AND CONTROL

Checkout, trouble-shooting, and preventive maintenance is much more easily and quickly accomplished when operation and control of the system is thoroughly understood. Procedures for operating the computer and its associated input-output units are given on the following pages, but principally as a brief, convenient reference. Detailed operating procedures are presented in the Operating manual. Procedures given here are presented in the order in which they will usually be used; exceptions are covered in the tests in which the variation in operating procedure is required.

Computer Operating Procedure

To apply power to computer:

Depress Power On button on control console (Power On indicator should light immediately; Ready indicator should light approximately 45 seconds after power turn-on).

To set Location counter from console keyboard:

1. If Fill Source indicator is extinguished, depress Fill Source button.
2. If Compute, Output Error, Overflow Error, and Verify Error indicators are lit, depress Error Reset button.
3. Depress Location button (Location indicator should light).
4. Depress in proper sequence the keyboard buttons corresponding to the location in which it is desired to enter information, consisting of four octal digits for the channels and sectors desired and one binary digit to indicate the first or second half of the sector (a zero for first and a one for second half, usually a zero). (Corresponding numerals should be displayed on the visual readout; if a wrong character has been keyed depress Clear button and repeat this step from its beginning.)

5. Depress Enter button (location will be entered into computer, visual readout will be cleared, and location counter indicator should display location set up -- location counter indicator is read by adding numbers beside lighted neons).

To enter commands from console keyboard:

1. Set location counter.
2. Depress Command button (Command indicator will light).
3. Depress in proper sequence the keyboard buttons for the pair of commands desired, consisting for each command of a sign, two octal digits for the operation code, four octal digits representing the address or other information, one binary digit (zero or one) for half-word sector indication or other information (the various meanings of sign, address, and half-word indicators are explained in the Operating manual); corresponding characters are displayed on the visual readout as the buttons for the command are depressed, if a wrong character has been keyed depress Clear button and repeat this step from its beginning.
4. Depress Enter button (pair of commands will be entered into location desired, visual readout will be cleared, and location counter will be advanced to next location).

5. To enter additional commands in consecutive locations repeat steps 3 and 4, in non-consecutive locations steps 1 through 4.

To enter mixed numbers from console keyboard.

1. Set location counter.
2. Depress Number button (Number indicator will light).
3. To enter whole-number portion, depress plus or minus sign followed by up to 11 (numeral) keys representing decimal digits (corresponding numerals are displayed on visual readout as numeral key is depressed, if wrong character has been keyed depress Clear button and repeat this step from its beginning).
4. Depress decimal point key (whole number should be entered into location desired and location counter should be advanced to next location; if it is desired to clear visual readout before entering fractional portion depress Clear key after decimal point key).
5. To enter fractional portion, depress up to 11 numeral keys representing decimal digits--if Clear or Enter key had been depressed following entry of whole number, also depress sign and decimal point keys before numeral keys (corresponding character are displayed on visual readout as keys are depressed, if wrong character has been keyed, depress Clear button and repeat this step from its beginning.

6. Depress Enter key (fractional number will be entered into location following whole number, visual readout will be cleared, and location counter will be advanced to next location).

7. To enter additional mixed numbers in consecutive locations repeat steps 3 through 6; in nonconsecutive locations, steps 1 through 6.

To enter whole numbers only:

1. Set location counter.
2. Depress Number button (Number indicator will light).
3. Depress plus or minus sign followed by up to 11 numeral keys representing decimal digits (corresponding numerals are displayed on visual readout as numeral key is depressed, if wrong character has been keyed depress Clear button and repeat this step from its beginning).
4. Depress decimal point key (whole number will be entered into location desired and location counter should be advanced to next location).
5. Depress Enter key (visual readout will be cleared).
6. To enter additional whole numbers only in consecutive locations repeat steps 3 through 6; in non-consecutive locations, steps 1 through 6.

To enter fractional numbers only:

1. Set location counter as described previously.
2. Depress Number button (Number indicator will light).
3. Depress plus or minus sign key, followed by decimal point key and up to 11 numeral keys representing decimal digits (corresponding numerals are displayed on visual readout as numeral key is depressed, if wrong character has been keyed depress Clear button and repeat this step from its beginning).
4. Depress Enter key (fractional number will be entered into location desired, visual readout will be cleared, and location counter will be advanced to next location).
5. To enter additional fractional numbers in consecutive locations repeat steps 3 and 4; in non-consecutive locations, steps 1 through 4.

To start computer (unless specified otherwise in test):

1. Check Ready indicator (should be lighted; if extinguished, turn on power).
2. Check Compute, Output Error, Overflow Error, and Verify Error indicators (should be extinguished; if lighted depress Error Reset button).

3. Check Fill Source indicator (should be lighted, if extinguished, depress Fill Source button).

4. Place Operation switch in Continuous position.

5. Place Preset Stop switch in Off position.

6. Place Sense Switches B, C, and D down.

7. Place Transfer Stop switch down.

8. Set Location Counter (using steps 3 through 5 of procedure) to location at which program or routine starts.

9. Depress Start button (Compute indicator should light).

To stop computer, unless halted by error or programed stop, depress Stop button (Compute indicator should extinguish).

To display contents of registers and rapid access loops on visual readout:

1. Check Ready indicator (should be lighted, if extinguished, turn on power).

2. Check Compute indicator (should be extinguished; if lighted, depress Stop button).

3. Check Error (Output, Overflow, and Verify) indicators (should be extinguished; if lighted, depress Error Reset button).

4. To read out registers, rotate location selector knob until desired register is indicated by arrow below knob and depress readout button corresponding to display format desired; to read out rapid access loops, rotate location selector knob until L or V loop is indicated, set right Sector selector knob to any number 0 through 7 according to last octal digit of loop address to be displayed, depress readout button corresponding to display format desired. (For characteristics of display formats, see Operating manual).

5. To display additional registers or loop locations repeat step 4.

To display contents of main memory locations on visual readout:

1. Repeat steps 1 through 3 of register and loop display procedures.

2. Rotate location selector knob until M is indicated by arrow below knob, set both Channel and Sector selector knobs to channel and sector locations, respectively, to be displayed, then depress readout button corresponding to display format desired. (For characteristics of display formats, see Operating manual.)

3. To display additional main memory locations, set Channel and Sector selector knobs to location to be displayed and depress readout button of display format desired.

Typewriter Operating Procedure

Typewriter operating procedures are the same for preparation of paper tape with control codes as for direct input and computer control except as indicated.

To apply power to typewriter (computer power need be turned on only if typewriter is to be used for input and output), turn switch on circuit breaker at front of desk and switch typewriter on (if typewriter is to be used for tape preparation, power switch on paper tape punch must also be turned on).

To prepare typewriter for operation:

1. Check for proper setting of tab stops, line space lever, paper guides, margin stops, paper release lever, and tab override switch (see Input-Output Service Manual or User's Service Guide for identification and description of controls). Carriage setting limits are scale 5 on left and scale 95 on right.
2. Place Computer-Manual Punch switch on paper tape punch in Computer position (if typewriter is to be used for tape preparation, place Computer-Manual Punch switch in Manual Punch position).

To set location counter from typewriter:

1. If Compute indicator is lighted, depress Stop Button on control console.
2. If Output Error, Overflow Error, and Verify Error indicators are lighted, depress Error Reset button on control console.
3. Depress Fill key on typewriter.
4. Depress Letters Shift key.
5. Depress L key (location indicator on control console should light).
6. Depress Figures Shift key.
7. Depress in proper sequence the numeral keys corresponding to the location in which it is desired to enter information, consisting of four octal digits for the channels and sectors desired and one binary digit to indicate the first or second half of the sector (a zero for first and a one for second half, usually a zero). (Numerals are not displayed on the control console visual readout but can be checked on typewritten copy before entry; if a wrong character has been typed, depress X key on typewriter and repeat this step from its beginning).

8. Depress Carriage Return (location will be entered into computer and location counter will indicate location set up -- location counter indicator is read by adding numbers beside lighted neons).

To enter commands from typewriter:

1. Prepare typewriter for operation.
2. Set location counter.
3. Make certain Fill key on typewriter is depressed.
4. Depress Letters Shift key.
5. Depress C key (Command indicator on control console should light).
6. Depress Figures Shift key.
7. Depress in proper sequence the typewriter keys necessary for the pair of commands desired, consisting for each command of a sign, two octal digits for the operation code, four octal digits representing the address or other information, one binary digit (zero or one) for half-word sector indication or other information (the various meanings of sign, address, and half-word indicators are explained in the Operating manual); characters are not displayed on the console visual readout but can be checked on the typewritten copy before entry, if a wrong character has been typed depress X key on typewriter and repeat this step from its beginning.

8. Depress Carriage Return (pair of commands will be entered into location desired and location counter will be advanced to next location).
9. To enter Additional commands in consecutive locations repeat steps 7 and 8; in non-consecutive locations, steps 2 through 8.

To enter mixed numbers from typewriter:

1. Prepare typewriter for operation.
2. Set location counter.
3. Make certain Fill key on typewriter is depressed.

4. Depress Letters Shift key.
5. Depress N key (Number indicator on control console will light).
6. Depress Figures Shift key.
7. To enter whole-number portion, depress plus or minus sign followed by up to 11 numeral keys representing decimal digits (numerals are not displayed on control console visual readout but may be checked on typewritten copy before entry, if a wrong character has been typed depress X key on typewriter and repeat this step from its beginning).

8. Depress decimal point key on typewriter (whole number should be entered into location desired and location counter should be advanced to next location).

9. To enter fractional-number portion, depress up to 11 numeral keys representing decimal digits (numerals are not displayed on control console visual readout but can be checked on typewritten copy before entry, if a wrong character has been typed depress X key on typewriter and repeat this step from its beginning).

10. Depress Carriage Return (fractional number should be entered into location following whole number and location counter should be advanced to next location).

11. To enter additional mixed numbers in consecutive locations repeat steps 7 through 10, in non-consecutive locations steps 2 through 10.

To enter whole numbers only:

1. Prepare typewriter for operation.
2. Set location counter.
3. Make certain Fill key on typewriter is depressed.

4. Depress Letters Shift key.
5. Depress N key (number indicator on control console should light).
6. Depress Figures Shift key.
7. Depress plus or minus sign followed by up to 11 numeral keys representing decimal digits (numerals are not displayed on control console visual readout but may be checked on typewritten copy before entry; if a wrong character has been typed depress X key and repeat this step from its beginning).
8. Depress decimal point key on typewriter (whole number should be entered into location desired and location counter advanced to next location).
9. Depress Carriage Return.
10. To enter additional whole numbers only in consecutive locations repeat steps 7 through 9; in non-consecutive locations, steps 2 through 9.

To enter fractional numbers only:

1. Prepare tyepwriter for operation.
2. Set location counter.
3. Make certain Fill key on typewriter is depressed.

4. Depress Letter Shift key.
5. Depress N key (number indicator on control console should light).

6. Depress Figure Shift key.

7. Depress plus or minus sign key followed by decimal point key and up to 11 numeral keys representing decimal digits (numerals are not displayed on control console visual readout but may be checked on typewritten copy before entry; if a wrong character has been typed, depress X key and repeat this step from its beginning).

8. Depress Carriage Return (fractional number should be entered into location desired and location counter advanced to next location).

9. To enter additional fractional numbers only in consecutive locations, repeat steps 7 and 8; in non-consecutive locations, steps 2 through 8.

To enter alphanumeric information:

1. Prepare typewriter for operation.
2. Set location counter.
3. Make certain Fill key on typewriter is depressed.
4. Depress F key.
5. Depress eight alphanumeric character keys (characters are not displayed on control console visual readout but may be checked on typewritten copy before entry.

6. Depress Carriage Return (unless format change is desired in which case depress C as the ninth character, then Carriage Return).

7. To enter additional alphanumeric words in consecutive locations repeat steps 5 and 6; in non-consecutive locations steps 2 through 6.

To start Compute mode from typewriter:

1. Check Ready indicator on control console (should be lighted; if extinguished, turn on power).

2. Check Compute indicator (should be extinguished; if lighted, depress Stop button on control console).

3. Check Error (Output, Overflow, and Verify) indicators (should be extinguished; if lighted, depress Error Reset button on control console).

4. Place Operation, Preset Stop, Transfer Stop switches, and Sense Switches B, C, and D in desired positions.

5. Set location counter.

6. Depress S key on typewriter (make certain Fill key on typewriter is depressed before depressing S key; Compute indicator on control console should light).

To halt compute mode from typewriter (unless halted by error condition or programed stop) can not be done without causing an output error.

NOTE

S code alone at end of information on tape will initiate computer operation; from location set in location counter H code alone following information on tape halts tape reader.
In all modes of tape preparation, information on the tape must consist of a minimum of seven characters between Carriage Return and/or Decimal Point codes which enter the information in memory, and between Carriage Return and/or Decimal Point codes and any of the control codes, such as Halt or Start. This procedure ensures proper recording of the information in memory. A blank may be substituted for any of the seven characters.

When entering a mixed number or a number consisting of only a fraction, a minimum of five tape sprocket holes must be punched between the Carriage Return code and the next number. The additional sprocket holes ensure sufficient time to convert the fractions to their internal representation. A special key on the typewriter punches these sprocket holes, one sprocket hole is punched each time the blank key is depressed.

A leader at least two folds in length should be prepared at the beginning of a tape by depressing the Tape Advance button on the punch control panel.

To operate in Fill mode:

1. Turn canister locking handle counterclockwise to unlocked position (down).
2. Pull canister off pins.
3. Place folded tape in canister on spring clips so that beginning of tape extends downward from carrying spring clips.
4. Thread tape around pins on bottom of inside of canister, up right side (when facing open canister with its top up), around pins on top of inside of canister, and down through guide jaws in center of canister (draw tape taut and sufficiently far through guides to enable refolding procedure of read tape to begin properly; ends of tape may be spliced to form continuous loop if desired; if a loop is formed, the tape can be advanced to the starting point from any place on the tape by depressing the Tape Advance button on the tape reader panel).
5. Replace canister on tape reader over pins.
6. Turn canister locking handle clockwise to locked position (up).
7. Depress keyboard fill source button, if necessary, to light the keyboard fill source light.
8. Depress Fill button on tape reader panel (tape will feed through reader until (1) end of tape passes through read head, (2) the reader detects an S or H code, or (3) the Stop button on the tape reader panel is depressed).

To operate in Verify mode:

1. Place tape in canister as outlined in steps 1 through 6 under Fill mode procedure.
2. Depress keyboard fill source button, if necessary, to light the keyboard fill source light.
3. Depress Verify button on tape reader panel (tape will feed through reader until (1) end of tape passes through read head, (2) a Verify error is detected, or (3) the Stop button on the tape reader panel is depressed).

To splice tape, when repairing or forming loops:

1. When repairing tape, repunch deleted information on new tape, preceding and following repunched section with at least two inches of "code delete" code (splice can be in middle of word without disturbing program; this step unnecessary when forming loop if leaders are long enough).

2. Cut one section of tape between 2" and 6-1/2" from last fold (tape should be cut squarely through a sprocket hole rather than on diagonal).

3. Place uncut section under cut section, maintaining 8-1/2" ± 1/32" between folds, and mark uncut section with pencil (using cut section as template).

4. Cut uncut section on a sprocket hole such that overlap will contain two complete sprocket holes when splice is completed.

5. Clean tape ends with acetone to remove excess oil and dirt.

6. Apply household cement or plastic cement on both surfaces to be joined and press firmly together (sprocket holes must coincide and edges of tape must be parallel; thickness of splice must not be greater than 0.010" at any point).

7. Remove excess cement from sprocket and information holes with pointed instrument.

8. Allow splice to dry at least 10 minutes before using tape.

Paper Tape Punch Operating Procedure

Punch operating procedures are the same for preparation of tape manually from the typewriter as for direct output from the computer, except that for tape preparation the Computer-Manual Punch switch is placed in Manual Punch position and for output it is placed in Computer position.

To apply power to punch, computer power need not be turned on unless punch will be used for output, turn switch on circuit breaker at front of desk and switch for the punch on.

To prepare punch for operation:

1. Load punch with paper tape as directed on rear of tape punch front and rear canister doors.

2. Place Computer-Manual Punch switch in proper position according to function to be performed.

3. Place Punch-External switch in Punch position.

4. Depress Tape Advance button to punch leader of sprocket holes only (leader should be sufficiently long to enable at least one fold in tape to lie properly in lower left corner of front canister. During operation tape feeds and stops automatically under both computer and typewriter control).

5. After punching of tape is completed, depress Tape Advance button to punch leader of sufficient length to enable at least one fold of tape to lie properly in lower left corner of front canister.

6. Tear off tape and remove from canister as directed on canister door.

NOTE

It is wise to punch an H (halt code) at the end of the tape, so the photoreader machine will halt when the tape is read into the machine. This is done by switching the computer-manual switch to the manual position and depressing the H key on the typewriter. The computer-manual switch should be returned to the computer position.

LOGIC DESCRIPTION

GENERAL

The technical world is witnessing a revolution in the field of mathematical computation that is rapidly and radically affecting all of man's science, technology, and mechanics. The great advance in technology is akin to the industrial revolution of the 18th and 19th centuries. This technical revolution was initiated by the recent introduction of mechanical computers. Because of the electrical nature of present-day electronic computers, these machines can process in a few milliseconds involved mathematical problems so lengthy and complicated as to be heretofore entirely beyond the scope of human operators. A wide and ever-increasing variety of mathematical problems is now subject to quick and accurate resolution, the results of which are directly applicable to the physical sciences. With the advent of the electronic computer, man has achieved an advantage over his environment that will increasingly enable him to understand and control his environment.

Basic Operation

The RECOMP II in its basic operation performs in the simplest mathematical processes. A reasonably complicated numerical calculation is broken down into its basic elements and the resulting primary computations are performed by computer internal logic. These computations could be accomplished as well by any literate person; the one element that enables RECOMP II to out-calculate any human operator is its speed. In one eight-hour period a modern digital computer is capable of performing a block of calculations that could not be solved by the effort of all living mathematicians and their descendants all similarly trained in many thousands of years.

Computer Analysis

At first glance a computer is a very complex device. To simplify an analysis, figuratively divide the concept into two parts: one consists of its components, mechanical and electronic; the other treats of its functions. The first is made up mostly of commonly known electronic elements such as resistors, condensers, diodes, and transistors, all fabricated on plug-in printed circuit boards; specialized elements

include nixie neon tubes for readout display, a rotating magnetic oxide-coated memory disk, read and write heads that function in conjunction with the memory disk, and input-output equipment such as the electric typewriter, the tape punch, the tape-reader, and the control console. Other manuals treat the specific physical components of the computer. The explanation that follows will be principally concerned with the functional concept of the computer, namely, the function of its electronic and mechanical elements in an integrated system capable of handling mathematical problems and computing required results.

Classes of Operations

Any operation that can be reduced to simple logic or arithmetic can be handled by the RECOMP II. There does not seem to be any theoretical limit to the types of problems that it can handle. Although it is sometimes difficult to reduce a given problem to its primary elements for insertion into the computer, problems have been found to fall into distinct classes to which solution methods have been previously worked out (frequently by the use of the computer itself) and recorded in specific computer programs. The work of analyzing a complex problem may be frequently reduced to identifying it within its class, after which an operation is performed by the application of an available program or combination of programs. The latter is the work of the programer who usually operates the computer, prepares the programs, and resolves the problems.

Because a digital computer is an information processing device, processing is the only operation it can perform. It cannot create any information. It may transform the input information into a more useful form or it may decide the mode of process operation, basing its decision upon the result of intermediate calculations; but the information that permits the machine to make such a decision is the work of the programer and must be programed into the machine at the outset of the computation.

The input to be processed may be the statement of a mathematical problem, rules governing the mathematical operations to be performed (commands), and the data to be operated upon. The digital computer is able to handle as wide a variety of problems as the programer is able to prepare for computer processing, but in every case the computer simply processes the input data according to a programed format specified by the operator and in a manner compatible with its internal logic circuitry. The computer makes use of more than one source of information during the processing; the input data, the list of operating instructions, and any intermediate commands for changing processing procedure based

upon results of calculations. Mathematical processes are partly built into the wiring of the computer as inalterable mathematical rules and partly entered as instructions for a particular problem. Information processing is performed by rearranging information from one relationship at the input to another at the output, not only in mathematical configuration but also in the dimension of time. In presenting a problem to the computer, a program or list of instructions is entered and subsequently the data on which the operations are to be performed. The computer must be able to remember the instructions until it receives the data, and because most of the operations cannot be performed simultaneously a continued use of memory or information storage is required.

Information Switching and Information Storage

Two functions are basic to the computer; information switching and information storage. Switching is the mathematical arrangement of information within the computer; information storage is the timed sequence handling of the information. Switching and storage in the computer are closely related and, in many functions, interrelated and interdependent. Most of the computer switching devices (flip-flops, gates) contain elements of memory or of information storage, and most of the storage devices (loops, memory disks, delay flip-flops) require switching mechanisms for access. Most switching and storage devices are binary in physical nature; ON or OFF. They exhibit two stable states. Consequently the computer is based on the binary system of numbers one and zero.

Digits

A digital computer uses digits or numbers. The basic computational process of RECOMP II operates through the use of binary digits, one and zero. These two digits arranged in an ordered configuration stand for all numbers, signs, factors, and commands necessary to the solution of any given problem. In computer terminology, digits refer to numbers, whether binary, octal, decimal, or of other mathematical description. In binary notation, a single binary digit is known as a bit. A group of digits (specified number) is known as a word. A group of several digits or bits, whether representing numerals or alphabetical letters, is also known as a character. A word may consist of several characters.

Subassemblies

In examining the basic mechanisms involved in a computer the machine and the computational process were viewed wholly. To analyze

the computer in detail, it is desirable to group the switching and storage elements into subassemblies which perform specialized functions in terms of the flow of information through the machine. First, the programer (operator) must get information into and out of the computer. This requires input and output devices which operate through input and output channels and incorporate both storage and switching functions. Input devices consist of typewriter, tape reader, keyboard; output devices consist of typewriter, tape punch, Nixie readout. Second, the operation to be performed on the input data is either logical or arithmetic and requires a logical and arithmetical unit (which is referred to in RECOMP II as a register). Third, instructions and intermediate data are stored in a storage device or memory. In RECOMP II, the storage device is a rotating magnetic oxide-coated disk upon the surface of which is magnetically inscribed the required digital information to be electronically recalled as required. Fourth, a control unit is required in computer operation. Control largely involves switching, but elements of timing are also necessary. The control portion of RECOMP II contains counters to keep track of the computational steps being performed, registers to direct the information along the proper logical paths, and a timing device known as the clock to synchronize the entire operation of the computer.

Programing

Programing is the fifth element of the computer concept. Programing is the direction of the control unit by the operator. Fixed programing is the part of the program built into the circuitry of the computer and available in modes to the programer. Once started along a certain control path, the computer will proceed through a fixed series of operations to a final result, processing the information fed into the machine in a rigid and predetermined manner. By selecting various fixed modes of control, the programer is able to cause some of the program steps to sample intermediate data, to transfer control to selected control points, to skip steps in programing if certain calculations occur, to stop the program, and to store in memory or in recirculation loops. The machine simply operates as directed and uses the result of its computations to perform certain other operations prescribed in advance. The term "programing" refers to those phases of the planning of a problem that consist of recording the steps required, the intermediate sampling of data desired, and the way in which the solution steps are to be altered as a result of this sampling.

Stored Program

The stored program consists of a coded notation of instructions in acceptable form for entry into the computer and which is entered with

the data alternately and in a programed sequence in such a manner as to operate upon the data to resolve the problem and produce an output in desired form.

Instructions

An instruction consists of two parts; an operation part which instructs the computer what to do, and an address part which designates the specific portion of memory in which to store information or from which to read information. The instruction contains the address code of the word upon which an operation is to be performed rather than the word itself. This simplifies the substitution of data, permitting the stored data to be changed without in any way changing the instructions and vice versa. Computers constructed on this principle are known as **single-address machines. RECOMP II is a single address machine. RECOMP II does not require a program counter since it takes instructions in the order in which they are listed unless a transfer of control is specified as an instruction.**

There are a number of kinds of operation that RECOMP II can perform (the list of operations is available in a separate operating manual). They are coded for convenience in entering them into the computer and are referred to as commands. They may be classified into several logical categories: (a) instructions relating to input and output devices, (b) instructions enabling the operator to store information in memory or to recall it as desired, (c) mathematical instructions such as to add, subtract, multiply, and divide, (d) control transfer operations and halt commands. The stored program method of operation used in RECOMP II provides an ability to operate on the instructions as though they were ordinary data. This means that the entire course of a computation can be altered including the operations themselves, the choice of data upon which the operations are to be performed, and the location in memory at which the results are to be stored. These changes can all be done on the basis of the results obtained in the course of the calculations to resolve the problem.

The programer need not write out every operation that he wishes the machine to perform. He may work out procedures for portions of the problem and then arrange a master program employing these 'subprograms'; the machine modifies and adapts these subprograms as required during various stages of problem solution.

Summary

Programing in its advanced form is certainly an art. Much of the apparent complexity of the computer resides in the program rather than the machine itself. A large number of basic and specialized programs have been worked out for RECOMP and new programs are constantly being devised for application in the ever-increasing variety of fields.

Programing has been briefly explained in this introduction because of a need to know some of its basic concepts in an approach to the understanding of the logic elements of the computer. Conversely, an intimate knowledge of the logic elements of the computer will broaden the programer's ability to use the computer as a useful tool to its fullest capacity.

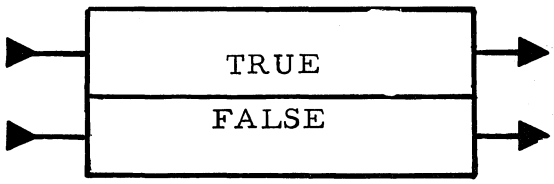
LOGIC EQUATION ANALYSIS

This section contains an analysis of the logic of RECOMP II in all modes of operation. It includes an analysis of computer timing, control, and information handling including input/output functions, and an examination of the system tester as it is utilized to determine the functional condition of the logical elements within the computer. Simplified data-flow diagrams are provided to facilitate a ready understanding of the logic functions; engineering drawings and schematics are available in Manual 508-T-11 for detailed reference to the circuitry and for point-to-point signal tracing.

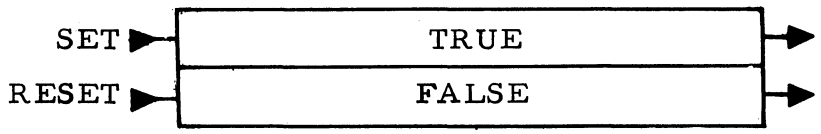
The logical design of RECOMP II was achieved entirely by application of Boolean algebra; in this section all functional computer processes will be explained on the basis of this technique. For those who are familiar with this notation, a recognition of the notational style is all that is required to understand the equations that follow; for those who are not accustomed to Boolean representation of logic circuitry, and for clarification of the style of notation used in this manual, the following general conditions are expressed:

Flip-Flops

Flip-flops are bistable electronic switches whose two stable states are designated TRUE and FALSE:



A flip-flop is considered set to a true state and reset to a false state:



Each flip-flop is symbolized by a letter of the alphabet, for example, the sum flip-flop, S. Flip-flops are more specifically identified by a modifying subscript as in the case of one term of the input/output register, F_5. The true and false states of a flip-flop are described respectively by the upper-case letter (with modifying subscript) as in the example F_5 and its prime notation, F_5' (F-five-prime).

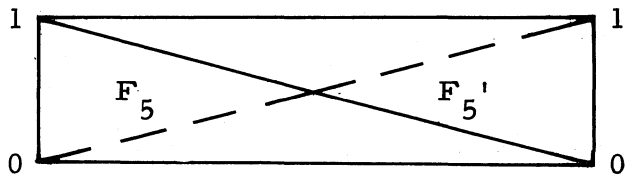
F_5	TRUE
F_5'	FALSE

When the flip-flop is set to the true or false state, it is identified by a function symbol specified by lower-case letters preceded by a "one" or "zero" written as follows:

$$1^{f}_{5}$$

$$0^{f}_{5}$$

When relating the flip-flop state to the binary number for which it stands (one or zero), it may be useful to make an analogy to a child's teeter-totter: when the flip-flop is set to the true state and the F_5 "end" is up, then $F_5 = 1$; at the same time the F_5' "end" is down and $F_5' = 0$. Conversely, when the flip-flop is reset to the false state and the F_5' "end" is up, then $F_5' = 1$; at the same time the F_5 "end" is down and $F_5 = 0$. By application of simple logic, for a given flip-flop state (true or false), when one term of the flip-flop equals one, the other must equal zero; change the state of the flip-flop and the terms exchange values -- the first term becomes equal to zero and the second term becomes equal to one.



The designation $1^{f}5$ translates in linguistic form to "one-set-F-five"; $0^{f}5$ translates to "zero-set-F-five". Specified input conditions are required to effect a one-set or zero-set operation, and these conditions are represented by equations as in the examples that follow:

$1^{f}5 = K_5 I_4 T_{pp}$ translates to: one-set F-five by K-five and I-four and T-p-p

$0^{f}5 = Z_1' N_6$ translates to: zero-set F-five by Z-one-prime and N-six.

There may be a single specified input condition to effect the setting or resetting of a given flip-flop, but there are usually multiple conditions deriving from various computer functions. These multiple conditions are expressed by either the logic "and" sign (·) or the logic "or" sign (+), or both in combination, and may be found for any particular function in the listing of logic equations included in this manual. The logic "and" sign (·) is not written between the terms of a conditional equation but is understood:

$1^{f}5 = K_5 \cdot I_4 \cdot T_{pp}$ is written $1^{f}5 = K_5 I_4 T_{pp}$

Amplifying the previous example:

$$1^{f}5 = K_5 I_4 T_{pp} + Z_1 N_6 + F_4' F_3' F_1 C_t + (\text{etc})$$

$$0^{f}5 = Z_1' N_6 + N_{12} + P_{d3} T_{40} + (\text{etc})$$ (from Appendix II, Logic Equations)

This is read as one-set F-five by K-five and I-four and T_{pp} OR Z-one and N-six OR F-four-prime and F-three-prime and F-one and C-t OR (etc).

The step beyond linguistically forming the equational conditions of the computer logic is to interpret the symbols of the functions: F_5 is one of five input-output register flip-flops; K_5 is one of the input channel switches; I_4 is a flip-flop that signifies one of the five operational states of the computer; T_{pp} is the tape-punch-pulse flip-flop. In a final interpretation of the equation $1^{f}5 = K_5 I_4 T_{pp}$, the number five flip-flop (F_5) of the input-output register is one-set by the TRUE state of the number five input channel switch (K_5) when the computer is in I_4 state of operation and the tape-punch-pulse flip-flop is also in the TRUE state.

An interpretation and brief analysis of the conditional terms in the above examples as well as for all RECOMP flip-flops, registers, counters, etc, may be found in Appendix III, Circuit Board Charts.

Timing

The computer functions in a serial progression and the functions are initiated by a series of timing pulses. All flip-flop state changes are "triggered" by these pulses called "CLOCK" pulses. The timing pulses occur at regular intervals (t_1, t_2, t_3.....to t_{41}) and are known as "bit" times written as the equational term "C". An equational condition coincident with a clock pulse (C) changes a flip-flop to the state indicated for the following bit time (t + 1). All equations have as a final term, whether written or not, the clock term (C): $1^f5 = K_5 I_4 T_{pp} C$. This form of computer timing is known as synchronous timing.

Gates

The decision elements in the computer are comprised of diode "gates." Two types of diode gates operate to interpret data and transmit control signals: these are described as "AND" and "OR" gates. An AND gate requires all of its inputs to be true in order for its output to be true. An "OR" gate requires at least one of its inputs to be true in order for its output to be true. Diode AND gates contain anywhere from two to nine input terms (including the enabling clock pulse). As many as twenty OR gates are used to trigger any one flip-flop. An AND gate is indicated by the symbol  . An OR gate is described by the symbol  containing an interior plus sign.

Boolean Algebra

For ease in understanding simplifications of Boolean equations, several useful relationships are presented. Truth tables, following, serve to verify (a) through (k). The fundamental laws (l) through (p) are stated here without proof.

(a) A0 = 0
(b) A + 0 = A
(c) A1 = A
(d) A + 1 = 1
(e) AA = A
(f) AA' = 0
(g) A + A' = 1

(h) A + AB = A
(i) A + A'B = A + B
(j) (AB)' = A' + B' } DeMorgan's Theorem
(k) (A + B)' = A'B'
(l) AB = BA } Commutative Law
(m) A + B = B + A
(n) A (BC) = AB(C) = ABC } Associative Law
(o) A + (B + C) = (A + B) + C = A + B + C
(p) A(B + C) = AB + AC Distributive Law

A	A'	(a) A0	(b) A + 0	(c) A1	(d) A + 1	(e) AA	(f) AA'	(g) A + A'
0	1	0	0	0	1	0	0	1
1	0	0	1	1	1	1	0	1

					(h)		(i)		(j)		(k)	
A	B	A'	B'	AB	A + AB	A'B	A + B	A + A'B	(AB)'	A' + B'	(A + B)'	A'B'
0	0	1	1	0	0	0	0	0	1	1	1	1
0	1	1	0	0	0	1	1	1	1	1	0	0
1	0	0	1	0	1	0	1	1	1	1	0	1
1	1	0	0	1	1	0	1	1	0	0	0	0

When two Boolean functions, A and B, are so related that A is true whenever B is true, though not necessarily conversely, then we say that A is included in B, written $A \subset B$. If also $B \subset A$, then obviously A = B. From the preceding truth table, we see that $AC \subset AB$.

Example of Logical Equations: To illustrate the method of deducing a set of logical equations, let us design a simple 3-stage binary counter that increases by "one" whenever a clock pulse is received up to a count of 7, resets itself to zero and repeats the process, ad infinitum. Table 4 shows the truth table for such a counter, where flip-flops X_3, X_2 and X_1 contain the bits in descending order of significance.

Table 4. Truth Table for 3-Stage Pulse Counter

Time t			Time (t + 1)		
X_3	X_2	X_1	X_3	X_2	X_1
0	0	0	0	0	1
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	1	0	1
1	0	1	1	1	0
1	1	0	1	1	1
1	1	1	0	0	0

It is observed from table 4 that if $X_1 = 0$ at time t, then $X_1 = 1$ at time t + 1. In other words X_1 is set to the true state whenever it is in the false state, X_1', and a clock pulse, C, is received, or

$$1^{X}1 = X_1'C$$

Similarly, X_1 is set to the false state whenever it is in the true state, X_1, and a clock pulse is received, or

$$0^{X}1 = X_1C$$

Further, X_2 is changed to the true state when it is false and X_1 is true and a clock pulse is received, or

$$1^{X}2 = X_2'X_1C$$

while X_2 is changed to false when both itself and X_1 are true and a clock pulse is received, or

$$0^{X}2 = X_2X_1C$$

Finally, X_3 changes to the opposite state whenever both X_2 and X_1 are true and a clock pulse is received. The complete set of logical equations for this counter are therefore:

X_1	$1^x1 = X_1'C$
	$0^x1 = X_1C$
X_2	$1^x2 = X_2'X_1C$
	$0^x2 = X_2X_1C$
X_3	$1^x3 = X_3'X_2X_1C$
	$0^x3 = X_3X_2X_1C$

It may be noted that any stage in a binary counter changes state whenever all less significant stages are true, or

X_n	$1^xn = X_n'X_{n-1}X_{n-2}\cdots X_2X_1C$
	$0^xn = X_nX_{n-1}X_{n-2}\cdots X_2X_1C$

This means simply that adding a 1 changes the 0 in binary 0111111 to 1 or the leading 1 in 111111 to 0.

Summary

This, then, is the step-by-step analysis the engineer or technician will follow in reading RECOMP II logic equations. He may devise his own personal mental shortcuts and abbreviations in analyzing the equational notations, and as his familiarity with the whole system increases, the symbols will come to represent the elements of the computer circuitry, operating conditions, and logic resolution. When a competent RECOMP II maintenance engineer or technician is referred to the term I_1, he immediately understands the reference to be one of the operational states of the computer, the first step in a master sequence control which involves, among other things, a selection of the required command pair from memory and the gating of the command pair into the Z-register. There are a large number of flip-flops (approximately 180), gates (more than 10,000), and numerous other elements and functions designated by letters and subscripts connected with the computer logic. Because of the complexity of their interrelationships and interdependence, a thorough and intimate knowledge of the majority of the symbols and even of a large number of the more common multiple-symbol configurations used as conditional terms in the logic equations is desirable in order that they may be readily identified on sight. Working daily with the equations will lead to an understanding more easily than may be at first anticipated. It has been

found by an ever-increasing number of engineers and technicians that the logic notational form is by far the least complex and the most articulate method of logic circuit description. A compendium of logic equations representing all logic functions within the computer will be found in Appendices I and II of this manual. Further knowledge of Boolean algebra and boolean relationships may be found in Montgomery Phister's LOGICAL DESIGN OF DIGITAL COMPUTERS published by John Wiley & Sons, Inc, New York and London.

COMPUTER CHARACTERISTICS

RECOMP II is a general-purpose digital computer that performs all internal operations in the binary number system. The memory is a rotating magnetic disk, and input equipment consists of a photoelectric paper tape reader, an electric typewriter, and a control unit keyboard. Output information may be punched on paper tape, typed, or displayed on the control panel visual readout. All computer circuits are transistorized and diodes are used for logic gating.

Memory

The magnetic disk memory unit provides basic timing channels, arithmetic registers, rapid access loops, and information storage channels.

Information Storage

Sixty-three channels, each containing 64 words of information and one channel containing 48 words comprise the 4080-word addressable main storage. This portion of the memory permits nonvolatile information storage. The magnetic oxide-coated disk surface retains stored information until erasure is desired, even during power interruptions or power-off condition. Words in memory are addressed by four octal digits from 0000 to 7757. The first two digits identify the memory channel; the last two denote the sector or word within the channel. The average access time to stored information is one-half a disk revolution or approximately 9.0 milliseconds.

Rapid Access Loops. In order to provide rapid access to stored information, two high-speed recirculation loops, L and V, of eight words each are available in programing. Loop addresses are indicated from octal 7760 to 7767 and from 7770 to 7777, respectively. The least

significant digit specifies which of the eight words in a loop is addressed. Access to loop information averages 0.95 millisecond. Information contained in the loops is volatile, subject to erasure during power interruption or power-off condition.

Arithmetic Register. Five recirculating loops or registers, each containing one word, are located on the memory disk. The A-register or accumulator stores the result of any arithmetic operation. The R-register stores the remainder in division, the least significant half of a double-length product or dividend, or the exponent of a floating-point number. The Z-register contains the command pair being obeyed at a given time. The B-register holds the number whose address is found in the command. The R-register holds the exponent of a floating-point number. The B and Z-registers are not directly accessible to the programer.

Timing Channels. Two permanently-recorded tracks on the disk provide basic timing controls for the computer operation. The clock channel contains a sine wave with a nominal frequency of 151 kc, at the nominal motor speed of 3450 rpm. After wave shaping this results in 2624 equally-spaced pulses which control the triggering of logic gates on the inputs to the flip-flops. Clock pulses are denoted by the letter C in the logic equations. The origin channel has a single pulse that serves as a reference for the nonvolatile storage by resetting counters used in memory addressing.

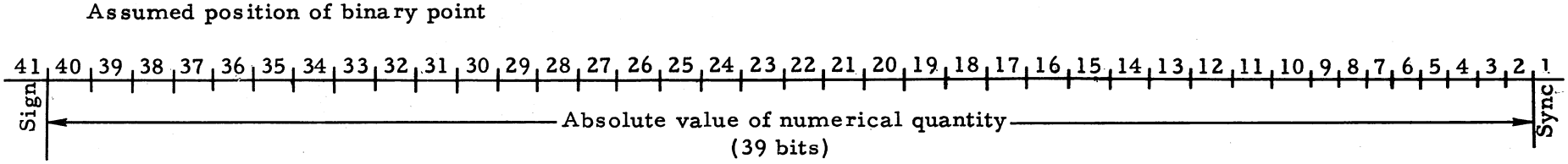
Word Format

A word consists of 41 bits (binary digits) which may represent either numerical quantities to be operated upon or computer commands that specify the operations to be performed. The right-most bit is used for memory synchronization and is not available in programing. The word formats for numerical quantities and commands are shown in figure 10.

Numbers. If a word contains a numerical quantity, the left-most bit stores the sign (1 if plus, 0 if minus) followed by the absolute value of the number as shown in figure 10. Normally, the computer assumes the binary point to be between the sign and the most significant bit (i.e., numbers are considered less than "one" in absolute value).

Commands. Single-address commands are stored two to a word. Each command has a sign bit, an operation code of two octal digits (6 bits), and an address containing four octal and one binary digit (13 bits) as shown in figure 10. The sign associated with a command has no effect on its execution except when sense switches are used. The operation code determines the arithmetic or logic operation to be performed. The address portion is the memory location of an operand for arithmetic commands, the number of binary shifts for a

NUMERICAL QUANTITY



COMMAND PAIR

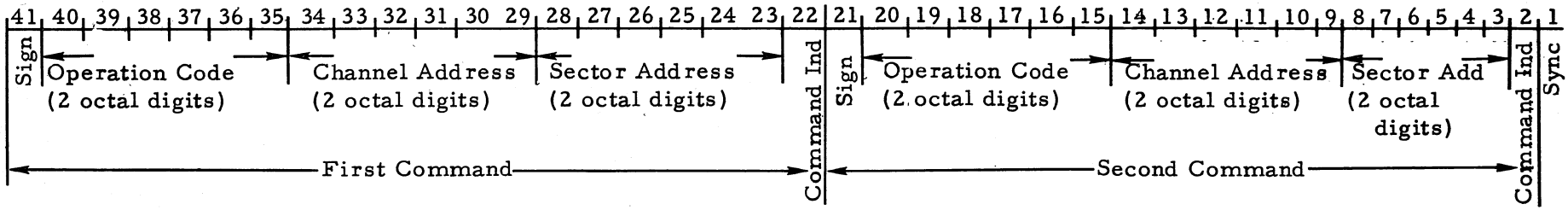


Figure 10. Word Format for Numerical Quantities and Commands

shift command, the octal code for punching and printing, or it has no significance at all. For arithmetic commands, the four octal address digits refer to the memory channel and sector of the operand. Depending on whether it is a 0 or 1, the additional bit specifies the first or second command in a word. This command indicator bit is significant only in the address of transfer-of-control and partial substitution commands.

Command List

The commands available in RECOMP are shown in table 5 with their octal operation codes, execution times, and explanatory notes.

Table 5. RECOMP II Commands

Word Times	Numerical Code	Operation Symbol	Operation
Fixed-Point Arithmetic			
1 + T	00	CLA w	Clear and add. (w) → A
2 + T	01	ADD w	Add. (w) + (A) → A
1 + T	02	CLS w	Clear and subtract. - (w) → A
2 + T	03	SUB w	Subtract. (A) - (w) → A
40 + T	13	MPR w	Multiply and round. (w) X (A) → A
40 + T	11	MPY w	Multiply (unrounded). (w) X (A) → A, R
42 + T	21	DSR w	Divide single length and round. (A)/(w) → A
41 + T	20	DSL w	Divide single length (unrounded). (A)/(w) → A, R (rem)
42 + T	23	DVR w	Divide (double length) and round. (A, R)/(w) → A
41 + T	22	DIV w	Divide (double length unrounded). (A, R)/(w) → A, R (rem)
1 + T	60	STO w	Store. (A) → w
*	42	STA w	Store address. $(A)_a \rightarrow w_a$
2 + n	40	ARS n	Accumulator right shift. Shift (A) n right, sign untouched.
2 + n	41	ALS n	Accumulator left shift. Shift (A) n places to the left, sign untouched.
42 + T	25	SQR w	Square root. $\sqrt{(w)} \rightarrow A$.

*Main Memory: 32 + T; Rapid Access Storage: 4 or 8 + T_{write}

Word Times	Numerical Code	Operation Symbol	Operation
Floating-Point Arithmetic			
1 + T	30	FCA w	Floating clear and add (w, w + 1)→ A, R.
5 + D + T	04	FAD w	Floating add. (w, w + 1) + (A, R)→ A, R.
1 + T	34	FCS w	Floating clear and subtract. - (w, w + 1)→A, R.
5 + D + T	06	FSB w	Floating subtract. (A, R) - (w, w + 1)→ A, R.
46 + D + T	07	FMP w	Floating multiply. (w, w + 1) X (A, R)→ A, R.
47 + D + T	05	FDV w	Floating divide. (A, R)/(w, w +1)→ A, R.
1 + T	35	FST w	Floating store. (A, R) → (w, w + 1).
43 + S + T	44	FSQ w	Floating square root. $\sqrt{(w, w + 1)}$→ A, R.
2 + n	45	FNM	Floating normalize. (A, R) normalized.
Input - Output			
	71	RDY w	Read from Y reader. Location counter-set to w.
	73	RDZ w	Read from Z reader. Location counter-set to w.
1 + 2N*	36	DIS w	Display. Display (w) on display register.
	74	PNC n	Punch character. Punch character corresponding to n.
	72	TYC n	Type character. Type character corresponding to n.
	76	PTC n	Punch and type character. Punch and type character corresponding to n.
	14	PNW w	Punch word. Punch (w).
	12	TYW w	Type word. Type (w).
	16	PTW w	Punch and type word. Punch and type (w).

*N = Number of positions on the display register (excluding initial sign) activated by instructions.

Word Times	Numerical Code	Operation Symbol	Operation
Block Transfers			
7 + T	64	CTL w	Copy to L loop. (w) to (w + 7) → 7760 to 7767.
7 + T	65	CFL w	Copy from L loop. (7760) to (7767) → w to w + 7.
7 + T	66	CTV w	Copy to V loop. (w) to (w + 7) → 7770 to 7777.
7 + T	67	CFV w	Copy from V loop. (7770) to (7777) → w to w - 7.
Control			
3	50	TZE w	Transfer on zero. If (A) = ± 0, transfer to w.
3	52	TPL w	Transfer on plus. If $(A)_s$ = +, transfer to w.
3	51	TMI w	Transfer on minus. If $(A)_s$ = -, transfer to w.
3	53	TOV w	Transfer on overflow. If overflow occurs, transfer to w.
3	57	TRA w	Transfer (unconditional). Transfer to w.
3	54	TSB w	Transfer on switch B. If on, transfer.
3	55	TSC w	Transfer on switch C. If on, transfer.
3	56	TSD w	Transfer on switch D. If on, transfer.
3	77	HTR w	Halt and transfer. Stop computer and reset location counter to w.
Miscellaneous			
1 + T	33	EXT w	Extract. Erase all bits in A except those in which corresponding bits in w are "ones".
2	43	XAR	Exchange A and R. Interchange (A) and (R).
1 + T	15	SAX	Store and exchange A and X. (A) → w, interchange (A) and (X).

Symbol

w = memory location whose address is w
(w) = contents of w
A = A-register
(A) = contents of the A-register
R = R-register
(R) = contents of the R-register
X = X-register
(X) = contents of the X-register
w_a = address portion of memory location w
$(A)_a$ = address portion of the contents of the A-register
$(A)_s$ = sign portion of the contents of the A-register
T = access time
D = difference in exponent parts of two floating-point numbers
n = number of shifts required by instruction

Timing Information

Main memory: 33.5 word times, average access time

Rapid Access Storage

Read: 3.5 word times average access time
Write: 5.5 word times average access time

1 word time: 0.27 millisecond

Basic Computer Operation

The functional operation of the computer may be regarded as shown in figure 11. The solid lines represent information flow and the dashed lines indicate functional control. Commands selected from memory control information flow and arithmetic processes.

Location Counter

The location counter is a 13-stage binary counter, designated G_{12}, $G_{11}, \ldots . G_0$ which stores the memory address of the command. During execution of a command, the location counter is increased by "one" to the address of the following command. This normal sequencing of the location counter is broken only where a transfer-of-control command is to be obeyed. The location counter is set initially when filling the computer.

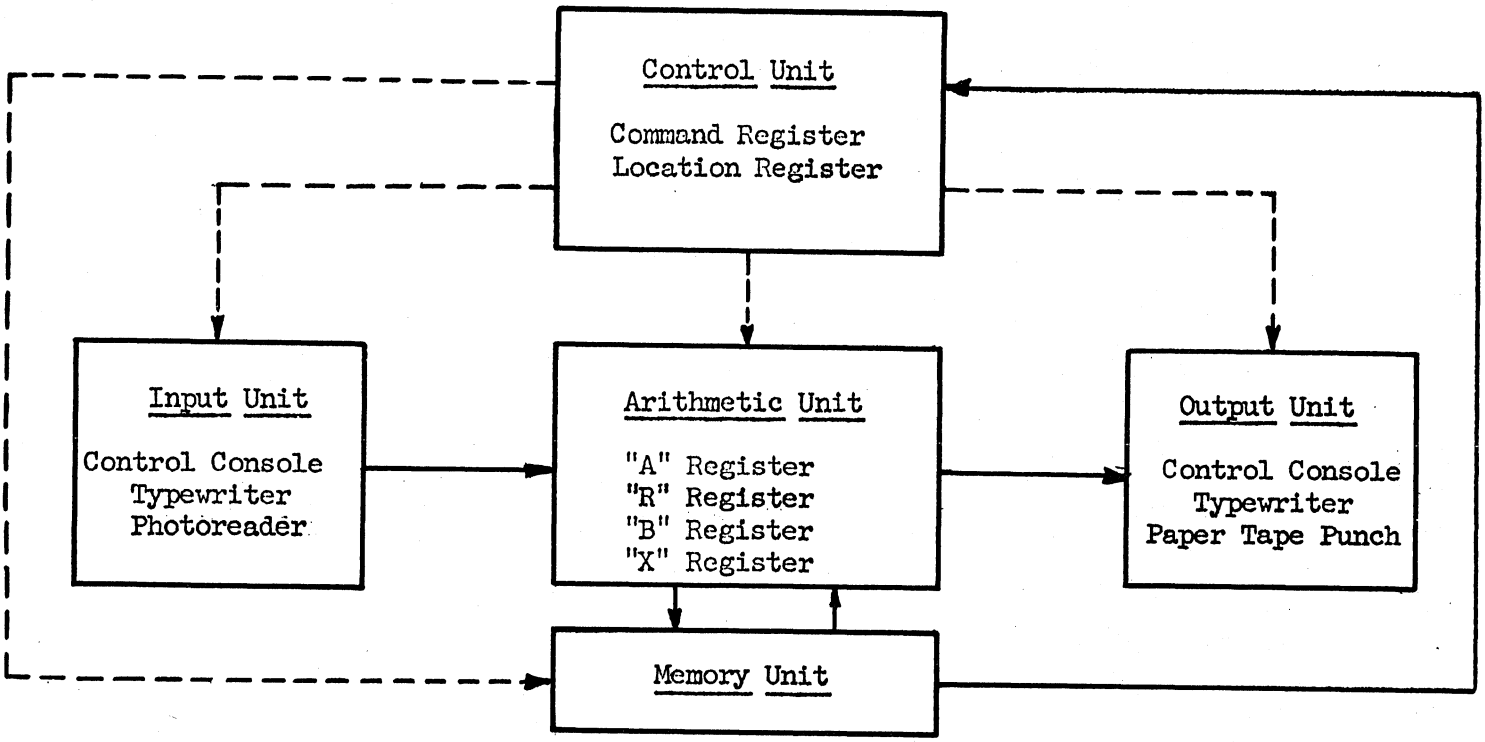


Figure 11. Functional Block Diagram of RECOMP II

Master Sequence Control

Normal computer operation requires that three different functions take place successively (see figure 12). First, the word containing the desired command is located and transferred from the memory to the Z-register. This is called command selection, and is identified with the true state of the timing flip-flop I_1. Since I_1 involves the selection of a full word from memory, this function is not repeated for the second command in a pair.

During the second function, command interpretation, the word referred to by the address of the command is located and transferred from memory to the B-register. I_2 is the flip-flop associated with command interpretation. Third, the command is executed whereby the required arithmetic or logical operation is performed. Flip-flop I_3 represents command execution. During execution of each command the location counter (G) is augmented to its succeeding value unless a transfer-of-control command is encountered. The master sequence control cycle is then repeated for the new command pair and so on until a stop command or error turns

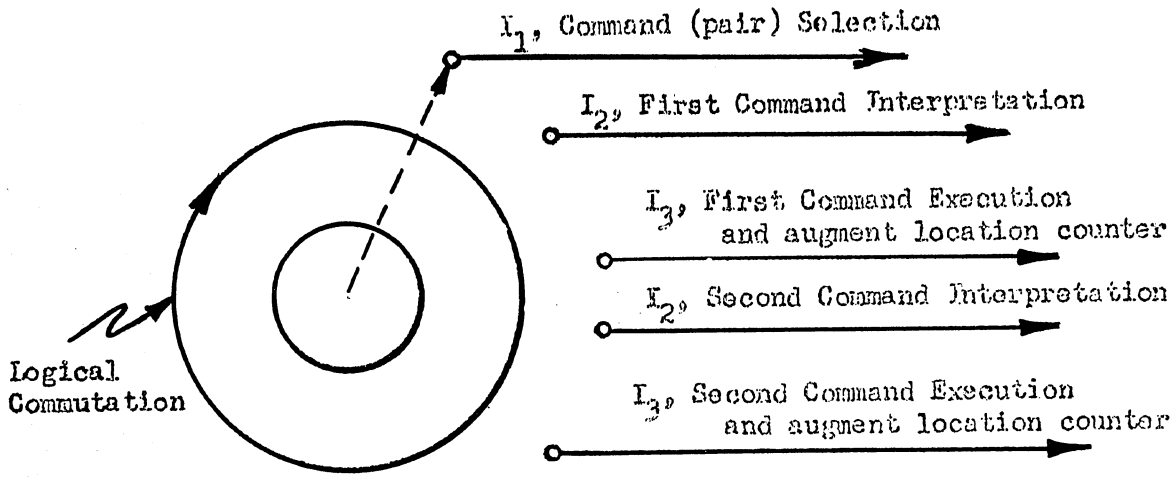


Figure 12. RECOMP Master Sequence Control Cycle

RECOMP to the noncompute state I_4. The master control sequence is initiated and the COMPUTE indicator is lighted by the control panel START button or the "S" character from the typewriter or the tape reader.

Noncompute Modes

Computer operation as described is suspended whenever flip-flop I_4 becomes true. During the I_4 state, however, any of the following input-output functions may be performed.

Fill Computer

Filling commands and numbers into memory and setting the command counter is accomplished through operation of any one of the input devices; keyboard, tape reader, or typewriter.

Verify Input

Verification of internally stored information may be achieved by comparison with the orginal input tapes. This function is performed automatically by pressing the VERIFY button on the tape reader and reading the tape.

Readout

Visual readout of information stored in any memory position or arithmetic register is possible in command, octal, or decimal format. The three possible binary groupings which allow these formats are initiated by the three readout buttons marked DECIMAL, OCTAL, and COMMAND.

Timing Source

The memory clock channel furnishes clock pulses necessary for all flip-flop triggering. Binary digit and sector counters and special timing flip-flops maintain the proper memory-referencing index.

Origin Flip-Flop

The single pulse on the origin channel is sensed by a read head and amplified by the origin pulse amplifier, X_{or}. X_{or} sets the origin flip-flop, X_o, which provides counter and timing flip-flop synchronization. After being set to the true state by the origin pulse, X_o resets itself to the false state during the bit time immediately following:

X_o

$1^{x}o = X_{or}C$
$0^{x}o = X'_{or}C$

Young Digit Counter

The Young digit counter provides the basic bit time and interval information for the computer. The terms P_2, P_3, P_4, P_5 operate as a shift counter; the terms P_1 and P_6 operate as conditional elements of and in conjunction with the counter. Its specific design goal is minimal logic, including the logic for associated timing flip-flops.

The memory reading and writing processes are serial operations, proceeding from least to most significant bit. The Young digit counter identifies each of the 41 bits within a given word and furnishes appropriate signals for control and arithmetic operations. The Young digit counter consisting of flip-flops P_1 through P_6, counts clock pulses modulo 41.

The counting cycle is set to zero (bit time 1) by X_o or on completion an entire counting cycle. Logical equations for the Young digit counter are shown on the next page.

P_1	$1^P1 = P_5{}'P_1{}'X_o{}'C + P_6{}'P_5P_4{}'P_3{}'X_o{}'C$
	$0^P1 = P_5{}'P_1{}'C + X_oC$
P_2	$1^P2 = P_1X_o{}'C$
	$0^P2 = P_1{}'C + X_oC$
P_3	$1^P3 = P_2C$
	$0^P3 = P_2{}'C$
P_4	$1^P4 = P_3X_o{}'C$
	$0^P4 = P_3{}'C + X_oC$
P_5	$1^P5 = P_4X_o{}'C$
	$0^P5 = P_4{}'C + X_oC$
P_6	$1^P6 = P_6{}'P_5P_4{}'P_3{}'P_2{}'C$
	$0^P6 = T_{41a}C$

COUNTER LOGIC

It is observed in the logic for P_1 (figure 13) that $P_5{}'P_1C$ one-sets P_1 at the proper points except for T_{22}. The X_o flip-flop signifies the memory reset point or origin. $X_o{}'$ allows the counter to count except when being reset. $P_6{}'P_5P_4{}'P_3{}'X_o{}'C$ provides one-set of P_1 at $T_{21}C$.

$P_5{}'P_1C$ zero-sets P_1 at the proper times and X_oC zero-sets P_1 for primary synchronization.

The logic for P_2 shows that it copies P_1 except at the origin.

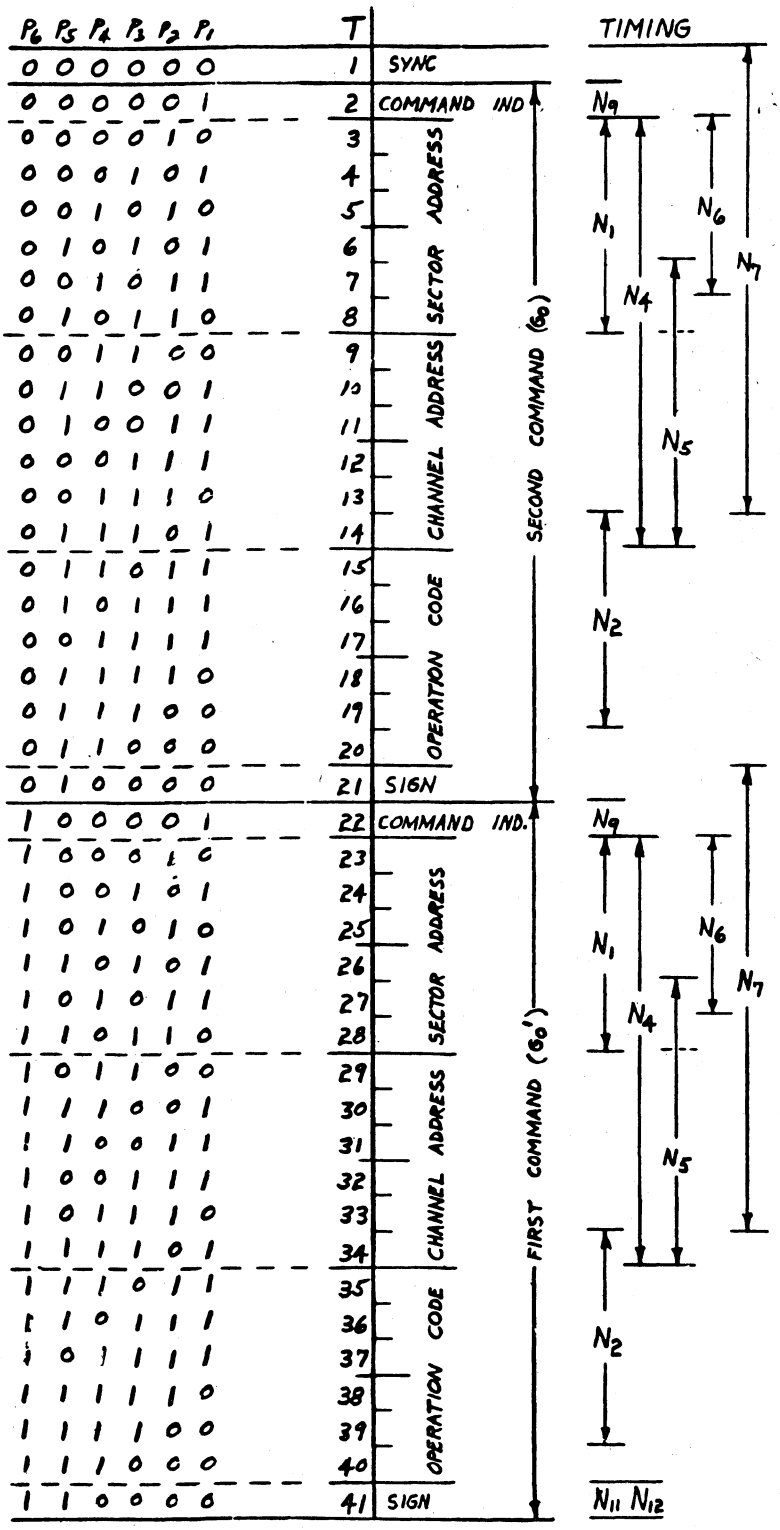


Figure 13. RECOMP II Digit Counter

The logic for P_3 indicates a normal left shift. However, a question may arise about the validity of omitting the X_oC term in the zero-set logic. This will be considered subsequently. The logic for P_4 and P_5 indicates normal left shifting.

Observe in the logic for P_6 that at the end of each word time P_6 is zero-set by $T_{41}C$. When the counter is first turned on, P_6 may either be a "one" or a "zero". After a maximum of one word time, P_6 will be forced back to the proper sequence by $T_{41}C$.

At this point, P_3 and P_6 logic will be considered in greater detail. Note that the logic for both of these flip-flops does not contain the X_oC term for resetting the counter. Therefore, the random settings of these flip-flops, when the computer is first turned on, must be analyzed. The cases possible at T_1 (after the first origin pulse, X_oC) are:

	P_6	P_5	P_4	P_3	P_2	P_1
CASE I	0	0	0	0	0	0
CASE II	0	0	0	1	0	0
CASE III	1	0	0	0	0	0
CASE IV	1	0	0	1	0	0

Case I - This is the proper array T_1 so no further discussion of this case is required.

Case II - The following truth table is constructed using the counter logic equations:

	P_6	P_5	P_4	P_3	P_2	P_1
T_1	0	0	0	1	0	0
T_2	0	0	1	0	0	1
T_3	0	1	0	0	1	0
T_4	0	0	0	1	0	1

At the count of 4 the counter is back in step.

Case III

	P_6	P_5	P_4	P_3	P_2	P_1
T_1	1	0	0	0	0	0
T_2	1	0	0	0	0	1
T_3	1	0	0	0	1	0
T_4	1	0	0	1	0	1
T_5	1	0	1	0	1	0

The first time the incorrect value of P_6 affects the picture is at $T_{21}C$. Here P_1 does not become one-set as is normal

	P_6	P_5	P_4	P_3	P_2	P_1	
T_{21}	1	1	0	0	0	0	
T_{22}	1	0	0	0	0	0	
T_{23}	1	0	0	0	0	1	Legitimate array for T_{22}

From this point on, the counter proceeds normally except that it is one count behind the actual bit time (as referenced to X_o). At actual T_{40}, the array is:

	P_6	P_5	P_4	P_3	P_2	P_1
T_{40}	1	1	1	1	0	0

This array appears to the logic as T_{39}. The T_{40} flip-flop then is turned on ($1^T40 = P6\ P_5\ P_3\ P_2\ 'P_1\ 'C$).
The array at actual T_{41} is:

	P_6	P_5	P_4	P_3	P_2	P_1
T_{41}	1	1	1	0	0	0

and at actual T_1 (X_o causes zero-set of P_1, P_2, P_4, P_5)

	P_6	P_5	P_4	P_3	P_2	P_1
T_1	1	1	0	0	0	0

Since the T_{41} C reset term becomes true one clock later than usual ($1^T41 = T_{40}$ C), P_6 is set at actual T_1 C.

	P_6	P_5	P_4	P_3	P_2	P_1
T_2	0	0	0	0	0	0

The counter is still one count behind and will remain so until the origin (X_o) again comes true (one disk revolution later). At actual T_{41} of the 64th word, the array is 1 1 1 0 0 0. The origin now zero-sets P_1, P_2, P_4 and P_5; giving 1 0 0 0 0 0 at actual T_1. The T_{41} flip-flop now is true and at actual T_2, P_6 is zero-set giving 0 0 0 0 0 1. The counter is now in step. It therefore requires over one disk revolution to synchronize the counter for Case III.

Case IV

	P_6	P_5	P_4	P_3	P_2	P_1
T_1	1	0	0	1	0	0
T_2	1	0	1	0	0	1
T_3	1	1	0	0	1	0
T_4	1	0	0	1	0	0
T_5	1	0	1	0	0	1

Notice that in Case IV the above "illegitimate" arrays form a closed recycling configuration of three terms (1 0 0 1 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 0 0 1 0 0 ..... etc.). This loop can be broken only by X_o C at a time when 1 0 0 1 0 0 is not present. Table 6 illustrates the process of synchronization:

Table 6. Synchronization Process

		P_6	P_5	P_4	P_3	P_2	P_1	
S_0	T_1	1	0	0	1	0	0	
	T_{41}	1	0	1	0	0	1	*
S_1	T_1	1	1	0	0	1	0	S_i = sector number
	T_{41}	1	0	0	1	0	0	
S_2	T_1	1	0	1	0	0	1	
	T_{41}	1	1	0	0	1	0	
S_{63}	T_1	1	0	0	1	0	0	**
	T_{41}	1	0	1	0	0	1	
S_0	T_1	1	0	0	0	0	0	***

* $\frac{\text{41 bits per word}}{\text{3 bits per cycle}}$ = 13-2/3 cycles per 41 bits. Therefore, the second array of the three is found at T_{41}.

** $\frac{\text{64 words per revolution}}{\text{3 words per cycle}}$ = 21-1/3 cycles per revolution. Therefore, the array 1 0 0 1 0 0 occurs at $T_1 S_{63}$.

*** P_1, P_2, P_4, and P_5 are zero-set at T_{41} C of S_{63} by X_o.

At $T_1 S_o$ (after the second X_o) Case IV reduces to Case III. Since Case III requires over one disk revolution to provide synchronization, the total synchronization time for Case IV is over two disk revolutions.

Special Digit Times

Particular counts of the digit counter are required for numerous control and arithmetic functions in the computer logic. It is economical to create special timing flip-flops in such cases rather than to form lengthy "AND" gates using the P flip-flops. In particular, signals are required at digit times 40, 41, 1 and 2, therefore three flip-flops are provided. Flip-flop T_{40} is set to 1 by the count of 39 and is therefore true during time 40 at which time it resets itself to 0 at T_{41}.

T_{40}

$1^t40 = P_6 P_5 P_3 P_2' P_1' C$
$0^t40 = T_{40} C$

T_{41} is turned on by T_{40}, T_1 by T_{41}, and T_2 by T_1. Each flip-flop resets itself to zero one bit-time after being turned on.

T_{41}

$1^t41 = T_{40} C$
$0^t41 = T_{41} C$

T_{41a}

$1^t41a = T_{40} C$
$0^t41a = T_{41a} C$

T_1

$1^t1 = T_{41} C$
$0^t1 = T_1 C$

T_2

$1^t2 = T_1 C$
$0^t2 = T_2 C$

Timing flip-flop N_9 is true for a single bit time of each ~~word~~ COMMAND

N_9

$1^n9 = G_0 T_1 C + G_0' P_6' P_5 P_4' P_3' P_2' C$
$0^n9 = N_9 C$

If the right-hand command is being effected, G_0 is true, N_9 is set at T_1 and becomes true during bit time 2. If the left-hand command is being effected, G_0 is false, N_9 is set at bit time 21 and true during bit time 22. N_9 is used to turn on several time interval flip-flops such as N_1, N_4 and N_6.

Sector Counter

The sector counter consisting of flip-flops S_1 through S_6 indicates the sector address of the next word to pass under a given read head. Since the information channels on the memory contain 64 word-sectors, the sector counter increases by 1 during each word time from 0 to 63. Thus the sector counter changes by 1 during the same time that the digit counter sweeps through its entire 41-bit count. Initial zero reference is accomplished by gate X_oC on the zero input to each S flip-flop.

Associated with the sector counter is a sector address time flip-flop, N_1, which is true only while the sector address (bits 3 through 8, or 23 through 28) of the current command is being read (see figure 10). Accordingly, N_1 is turned on by N_9 and turned off 6 bit times later.

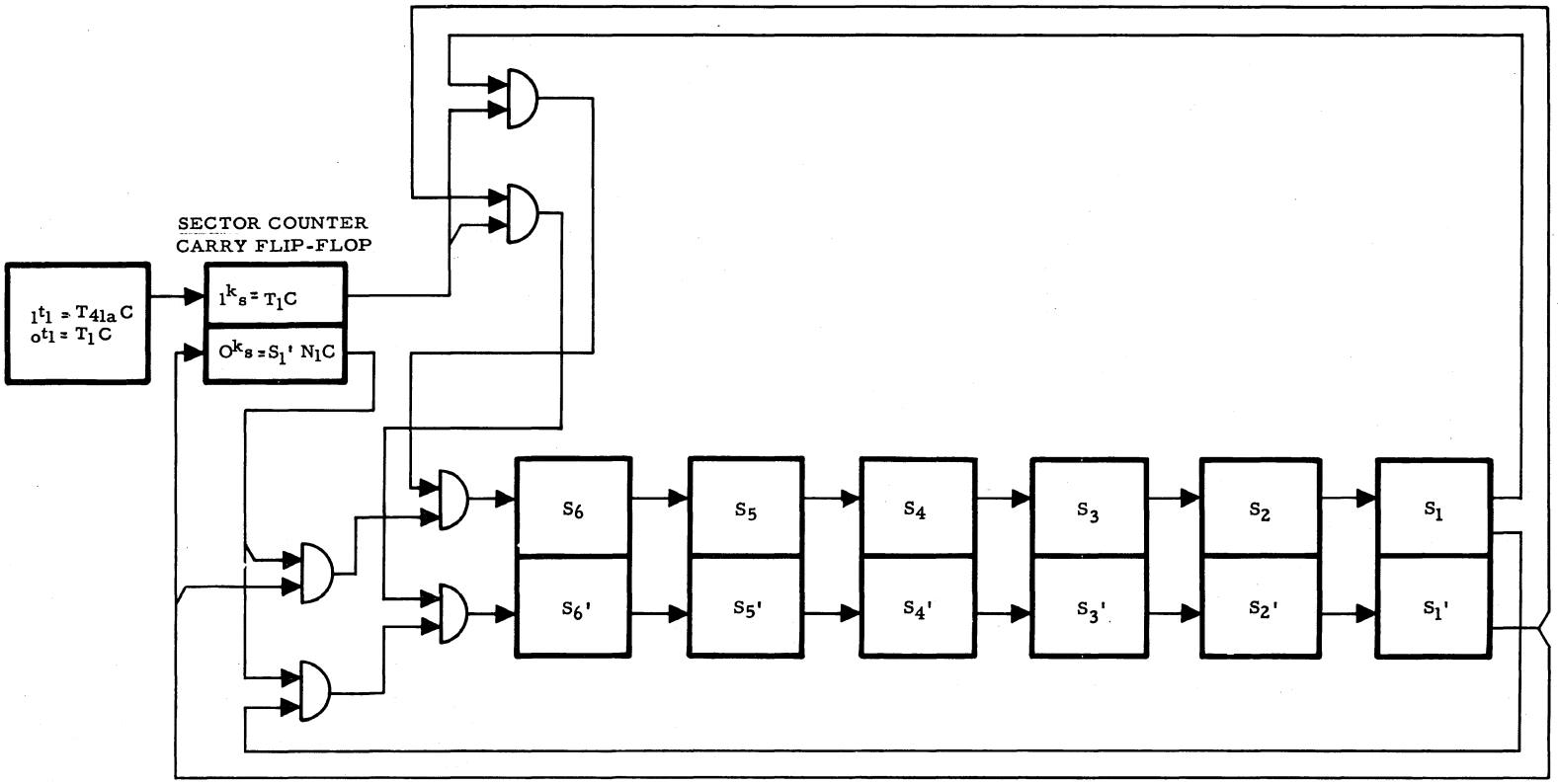
N_1	$1^n1 = N_9C$
	$0^n1 = P_2 P_3 C$

During sector address times (bits 3 through 8 and 23 through 28), both P_2 and P_3 are true only at bit time 8 and at bit time 28. Additional P diodes in these terms are therefore redundant.

Figure 14 illustrates the process by which the sector counter is augmented.

First, the increment "one" to be added to the S-counter sets the carry flip-flop K_s at T_1 of each word time. Next, cyclic right-shifting of the S-counter occurs during the six bit times of N_1. During the end-around shift, if K_s is false, the state of S_1 is copied in S_6; if K_s is true, S_1 is reversed in S_6. K_s is one-set when a zero appears in S_1 and remains in the false state for the duration of the word time. For example, figure15 shows how the sector counter and related flip-flops behave when the sector count is advanced from 19 to 20. G_0 is assumed true. If G_0 is false, N_9 is true at bit time 22 instead of T_2, and N_1 is true from 23 to 28 instead of 3 through 8. Although K_s is again set at T_1, whatever changes formerly occurred in the sector counter and K_s during times 23 through 29.

K_s	$1^{k_s} = T_1C$
	$0^{k_s} = S_1'N_1C$
S_1	$1^{s}1 = S_2N_1C$
	$0^{s}1 = S_2'N_1C + X_oC$
S_2	$1^{s}2 = S_3N_1C$
	$0^{s}2 = S_3'N_1C + X_oC$
S_3	$1^{s}3 = S_4N_1C$
	$0^{s}3 = S_4'N_1C + X_oC$
S_4	$1^{s}4 = S_5N_1C$
	$0^{s}4 = S_5'N_1C + X_oC$
S_5	$1^{s}5 = S_6N_1C$
	$0^{s}5 = S_6'N_1C + X_oC$
S_6	$1^{s}6 = S_1K_s'N_1C + S_1'K_sN_1C$
	$0^{s}6 = S_1'K_s'N_1C + S_1K_sN_1C + X_oC$



RC8-126

Figure 14. Sector Counter Augmentation

Bit Time	T_1	N_9	N_1	K_s	S_6	S_5	S_4	S_3	S_2	S_1	
1	1	0	0	0	0	1	0	0	1	1	$= 23_8 = 19_{10}$
2	0	1	0	1	0	1	0	0	1	1	
3	0	0	1	1	0	1	0	0	1	1	
4	0	0	1	1	0	0	1	0	0	1	
5	0	0	1	1	0	0	0	1	0	0	
6	0	0	1	0	1	0	0	0	1	0	
7	0	0	1	0	0	1	0	0	0	1	
8	0	0	1	0	1	0	1	0	0	0	
9	0	0	0	0	0	1	0	1	0	0	
10	0	0	0	0	0	1	0	1	0	0	
.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	.	.	
41	0	0	0	0	0	1	0	1	0	0	$= 24_8 = 20_{10}$

Figure 15. Example of Sector Counting

Such a counter is sometimes called a one-input adder because it performs ordinary serial addition with only one operand, the other being assumed zero and the carry stage initially set to one.

COMMAND SELECTION I_1

The first step, I_1, in the master sequence of normal computation involves selection of a command pair from memory and then gating it into the Z-register. Command selection is identified by flip-flop I, being one-set. The location counter, flip-flops G_1 through G_{12}, specifies the memory location of the desired command pair. G_7 through G_{12} determine the memory channel and G_1 through G_6 determine the sector to be read in that channel.

Recirculation of Location Counter

The 12-bit G-counter experiences right-shifting during I_1 from bit times 3 through 14, or 23 through 34 (depending on whether the right-or left-hand command is being executed). N_4 furnishes the necessary timing.

N_4	
	$1^n4 = I_1 N_9 C$
	$0^n4 = P_5 P_4 P_3 C$

The terms in 0^n4 are true at bit times 14 and 34 respectively and at no other time during the intervals 3 through 14 and 23 through 34 (see figure 13). Recirculation logic for the G-counter is shown in figure 16.

The expression $K_g{}'F_0{}'U_1$ in the final G_{12} logic is always true during command selection, I_1, in the compute mode, $I_4{}'$. Reasons for this expression will be presented in later sections of this manual.

Channel Selection

The channel address originally contained in G_7 through G_{12} can be read from G_1 during bit times 9 through 14 or 29 through 34 as the G-counter shifts. A "channel address time" flip-flop N_5 defines the time interval.

N_5	
	$1^n5 = (N_1 P_5) D_0{}' D_{10}{}' D_{11}{}' R_{cl}{}' I_3{}' C$
	$0^n5 = P_5 P_4 P_3 C$

Examination of figure 13 will show that the terms in parentheses are true at bit times 6 and 26 respectively while the terms in $_0n_5$ can turn N_5 off only at 14 and 34 respectively. The expression outside the parentheses in $_1n_5$ is true during I_1 except for the last word time.

The channel register, flip-flops C_1 through C_6, receives the channel address bits as follows: C_6 copies G_1 during N_5 while the C-register is shifted right.

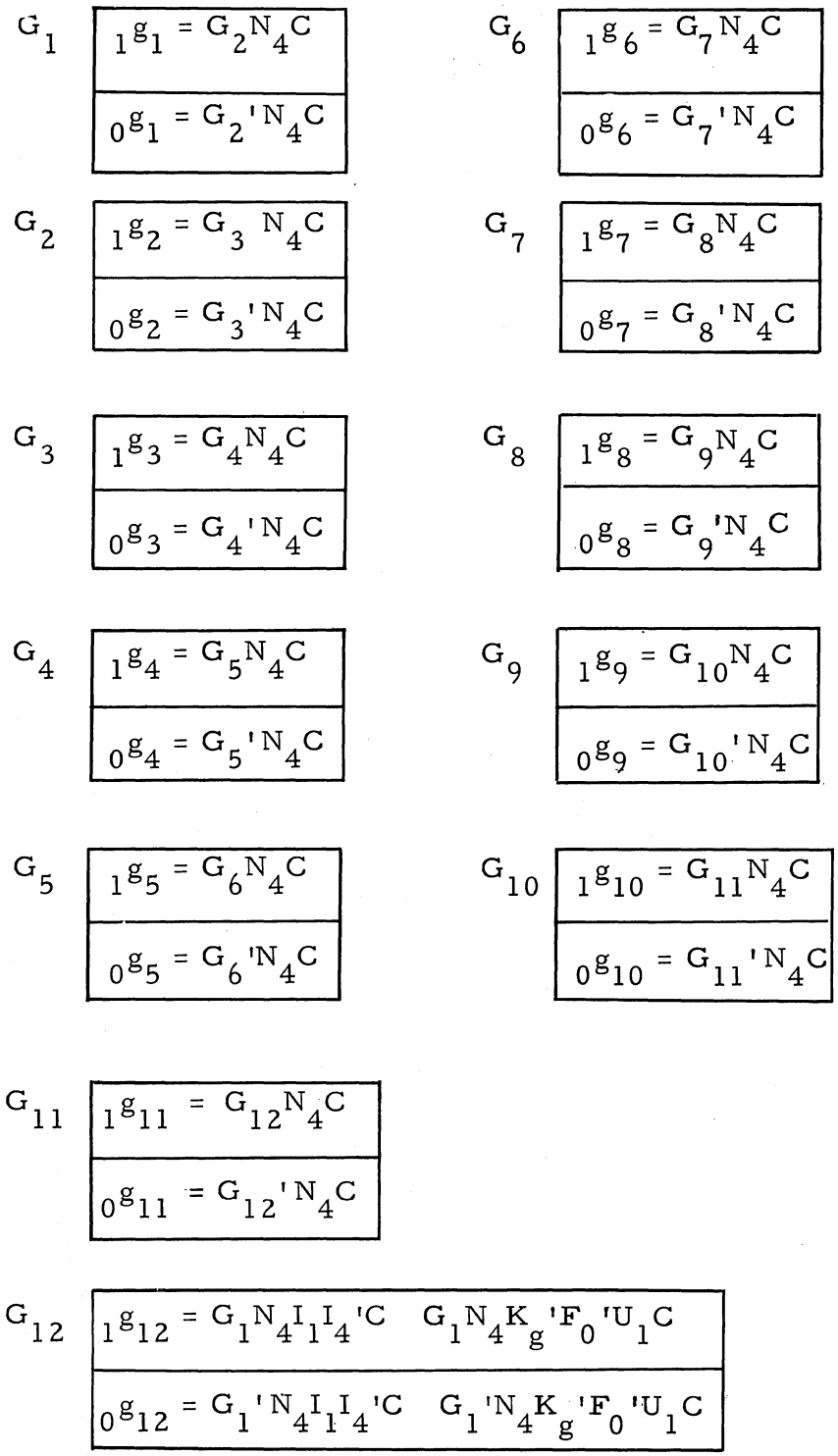


Figure 16. Recirculation Logic for the G Counter

C_1	${}_1c_1 = C_2N_5C$ ${}_0c_1 = C_2'N_5C$	C_4	${}_1c_4 = C_5N_5C$ ${}_0c_4 = C_5'N_5C$
C_2	${}_1c_2 = C_3N_5C$ ${}_0c_2 = C_3'N_5C$	C_5	${}_1c_5 = C_6N_5C$ ${}_0c_5 = C_6'N_5C$
C_3	${}_1c_3 = C_4N_5C$ ${}_0c_3 = C_4'N_5C$	C_6	${}_1c_6 = G_1I_1N_5C$ ${}_0c_6 = G_1'I_1N_5C$

Information from eight of the 64 memory read amplifiers is gated into the memory read power amplifiers M_{ra}, M_{rb}, M_{rc}, M_{rd}, M_{re}, M_{rf}, M_{rg}, M_{rh}, as determined by the three least significant channel bits in C_1 through C_3. (Clock pulses are not required here because $M_{ra, b, c, d, e, f, g, h}$ are amplifiers, not flip-flops. As a result, there is no time delay in this transfer of information.)

M_{ra}

$$
\begin{aligned}
{}_1m_{ra} = {} & (C_3'C_2'C_1')M_{ro} + (C_3'C_2'C_1)M_{r1} + (C_3'C_2C_1')M_{r2} + \\
& (C_3'C_2C_1)M_{r3} + (C_3C_2'C_1')M_{r4} + (C_3C_2'C_1)M_{r5} + \\
& (C_3C_2C_1')M_{r6} + (C_3C_2C_1)M_{r7}
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{ra} = {} & (C_3'C_2'C_1')M_{ro}' + (C_3'C_2'C_1)M_{r1}' + (C_3'C_2C_1')M_{r2}' + \\
& (C_3'C_2C_1)M_{r3}' + (C_3C_2'C_1')M_{r4}' + (C_3C_2'C_1)M_{r5}' + \\
& (C_3C_2C_1')M_{r6}' + (C_3C_2C_1)M_{r7}'
\end{aligned}
$$

M_{rb}

$$
\begin{aligned}
{}_1m_{rb} = {} & (C_3'C_2'C_1')M_{r8} + (C_3'C_2'C_1)M_{r9} + (C_3'C_2C_1')M_{r10} + \\
& (C_3'C_2C_1)M_{r11} + (C_3C_2'C_1')M_{r12} + (C_3C_2'C_1)M_{r13} + \\
& (C_3C_2C_1')M_{r14} + (C_3C_2C_1)M_{r15}
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rb} = {} & (C_3'C_2'C_1')M_{r8} + (C_3'C_2'C_1)M_{r9}' + (C_3'C_2C_1')M_{r10}' + \\
& (C_3'C_2C_1)M_{r11}' + (C_3C_2'C_1')M_{r12}' + (C_3C_2'C_1)M_{r13}' + \\
& (C_3C_2C_1')M_{r14}' + (C_3C_2C_1)M_{r15}'
\end{aligned}
$$

M_{rc}

$$
\begin{aligned}
{}_1m_{rc} = {} & (C_3'C_2'C_1')M_{r16} + (C_3'C_2'C_1)M_{r17} + (C_3'C_2C_1')M_{r18} + \\
& (C_3'C_2C_1)M_{r19} + (C_3C_2'C_1')M_{r20} + (C_3C_2'C_1)M_{r21} + \\
& (C_3C_2C_1')M_{r22} + (C_3C_2C_1)M_{r23}
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rc} = {} & (C_3'C_2'C_1')M_{r16}' + (C_3'C_2'C_1)M_{r17}' + (C_3'C_2C_1')M_{r18}' + \\
& (C_3'C_2C_1)M_{r19}' + (C_3C_2'C_1')M_{r20}' + (C_3C_2'C_1)M_{r21}' + \\
& (C_3C_2C_1')M_{r22}' + (C_3C_2C_1)M_{r23}'
\end{aligned}
$$

M_{rd}

$$
\begin{aligned}
{}_1m_{rd} = {} & (C_3'C_2'C_1')M_{r24} + (C_3'C_2'C_1)M_{r25} + (C_3'C_2C_1')M_{r26} + \\
& (C_3'C_2C_1)M_{r27} + (C_3C_2'C_1')M_{r28} + (C_3C_2'C_1)M_{r29} + \\
& (C_3C_2C_1')M_{r30} + (C_3C_2C_1)M_{r31}
\end{aligned}
$$

$$0^{m}_{rd} = (C_3'C_2'C_1')Mr_{24}' + (C_3'C_2'C_1)Mr_{25}' + (C_3'C_2C_1')Mr_{26}' + (C_3'C_2C_1)Mr_{27}' + (C_3C_2'C_1')Mr_{28}' + (C_3C_2'C_1)Mr_{29}' + (C_3C_2C_1')Mr_{30}' + (C_3C_2C_1)Mr_{31}'$$

M_{re}

$$1^{m}_{re} = (C_3'C_2'C_1')Mr_{32} + (C_3'C_2'C_1)Mr_{33} + (C_3'C_2C_1')Mr_{34} + (C_3'C_2C_1)Mr_{35} + (C_3C_2'C_1')Mr_{36} + (C_3C_2'C_1)Mr_{37} + (C_3C_2C_1')Mr_{38} + (C_3C_2C_1)Mr_{39}$$

$$0^{m}_{re} = (C_3'C_2'C_1')Mr_{32}' + (C_3'C_2'C_1)Mr_{33}' + (C_3'C_2C_1')Mr_{34}' + (C_3'C_2C_1)Mr_{35}' + (C_3C_2'C_1')Mr_{36}' + (C_3C_2'C_1)Mr_{37}' + (C_3C_2C_1')Mr_{38}' + (C_3C_2C_1)Mr_{39}'$$

M_{rf}

$$1^{m}_{rf} = (C_3'C_2'C_1')Mr_{40} + (C_3'C_2'C_1)Mr_{41} + (C_3'C_2C_1')Mr_{42} + (C_3'C_2C_1)Mr_{43} + (C_3C_2'C_1')Mr_{44} + (C_3C_2'C_1)Mr_{45} + (C_3C_2C_1')Mr_{46} + (C_3C_2C_1)Mr_{47}$$

$$0^{m}_{rf} = (C_3'C_2'C_1')Mr_{40}' + (C_3'C_2'C_1)Mr_{41}' + (C_3'C_2C_1')Mr_{42}' + (C_3'C_2C_1)Mr_{43}' + (C_3C_2'C_1')Mr_{44}' + (C_3C_2'C_1)Mr_{45}' + (C_3C_2C_1')Mr_{46}' + (C_3C_2C_1)Mr_{47}'$$

M_{rg}

$$1m_{rg} = (C_3'C_2'C_1')Mr_{48} + (C_3'C_2'C_1)Mr_{49} + (C_3'C_2C_1')Mr_{50} + (C_3'C_2C_1)Mr_{51} + (C_3C_2'C_1')Mr_{52} + (C_3C_2'C_1)Mr_{53} + (C_3C_2C_1')Mr_{54} + (C_3C_2C_1)Mr_{55}$$

$$0m_{rg} = (C_3'C_2'C_1')Mr_{48}' + (C_3'C_2'C_1)Mr_{49}' + (C_3'C_2C_1')Mr_{50}' + (C_3'C_2C_1)Mr_{51}' + (C_3C_2'C_1')Mr_{52}' + (C_3C_2'C_1)Mr_{53}' + (C_3C_2C_1')Mr_{54}' + (C_3C_2C_1)Mr_{55}'$$

M_{rh}

$$1m_{rh} = (C_3'C_2'C_1')Mr_{56} + (C_3'C_2'C_1)Mr_{57} + (C_3'C_2C_1')Mr_{58} + (C_3'C_2C_1)Mr_{59} + (C_3C_2'C_1')Mr_{60} + (C_3C_2'C_1)Mr_{61} + (C_3C_2C_1')Mr_{62} + (C_3C_2C_1)Mr_{63}$$

$$0m_{rh} = (C_3'C_2'C_1')Mr_{56}' + (C_3'C_2'C_1)Mr_{57}' + (C_3'C_2C_1')Mr_{58}' + (C_3'C_2C_1)Mr_{59}' + (C_3C_2'C_1')Mr_{60}' + (C_3C_2'C_1)Mr_{61}' + (C_3C_2C_1')Mr_{62}' + (C_3C_2C_1)Mr_{63}'$$

Careful study of the memory read power amplifier logic reveals that when the least significant octal channel address digit is n, we may write

M_{ra}

$$1m_{ra} = M_{rn}$$
$$0m_{ra} = M_{rn}'$$

M_{rb}

$$1m_{rb} = M_{r(n+8)}$$
$$0m_{rb} = M_{r(n+8)}'$$

M_{rc}

$$1m_{rc} = M_{r(n+16)}$$
$$0m_{rc} = M_{r(n+16)}'$$

M_{rd}

$$1m_{rd} = M_{r(n+24)}$$
$$0m_{rd} = M_{r(n+24)}'$$

M_{re}

$$1m_{re} = M_{r(n+32)}$$

$$0m_{re} = M_{r(n+32)}'$$

M_{rg}

$$1m_{rg} = M_{r(n+48)}$$

$$0m_{rg} = M_{r(n+48)}'$$

M_{rf}

$$1m_{rf} = M_{r(n+40)}$$

$$0m_{rf} = M_{r(n+40)}'$$

M_{rh}

$$1m_{rh} = M_{r(n+56)}$$

$$0m_{rh} = M_{r(n+56)}'$$

where $0 < n \leq 7$.

Information from only one of the eight read power amplifiers is copied into the memory flip-flop M_r according to the states of C_6, C_5, and C_4.

M_r

$$1m_r = C_6'C_5'C_4'M_{ra} + C_6'C_5'C_4M_{rb} + C_6'C_5C_4'M_{rc} + C_6'C_5C_4M_{rd} + C_6C_5'C_4'M_{re} + C_6C_5'C_4M_{rf} + C_6C_5C_4'M_{rg} + C_6C_5C_4M_{rh}$$

$$0m_r = C_6'C_5'C_4'M_{ra}' + C_6'C_5'C_4M_{rb}' + C_6'C_5C_4'M_{rc}' + C_6'C_5C_4M_{rd}' + C_6C_5'C_4'M_{re}' + C_6C_5'C_4M_{rf}' + C_6C_5C_4'M_{rg}' + C_6C_5C_4M_{rh}'$$

Thus, for a given binary configuration in C_1 through C_6, M_r will accept the information read from only one memory channel. For example if the channel address of a command is 45, $\overset{C_6}{1}\ \overset{C_5}{0}\ \overset{C_4}{0}\ \overset{C_3}{1}\ \overset{C_2}{0}\ \overset{C_1}{1}$ in binary, then $C_3\ C_2'\ C_1$ becomes true and M_{r5}, M_{r13}, M_{r21}, M_{r29}, M_{r37}, M_{r45}, M_{r53}, and M_{r61} are copied into M_{ra}, M_{rb}, M_{rc}, M_{rd}, M_{re}, M_{rf}, M_{rg}, and M_{rh} respectively. Since $C_6\ C_5'\ C_4'$ is also true, M_{rf} $(= M_{r45})$ is copied into M_r as required.

Sector Selection

In order to locate the sector identified in G_1 through G_6, it is necessary to wait for the sector counter S_1 through S_6 to reach agreement with G_1 through G_6, respectively. Obviously, such agreement must occur within a disk revolution since this is the duration of a complete sector count (0 through 63). Since the S counter is right-shifted through S_1 during N_1 (bit times 3 through 8 or 23 through 28) while the G-counter is similarly shifted through G_1 during N_4 (bit times 3 through 14 or 23 through 34), it is clear that S_1 and G_1 will contain corresponding bits during the 6-bit interval N_1.(See figure 17.)

S_o is the memory read selection flip-flop; it is set to zero at every T_{41} pulse. During N_1, flip-flops G_1 and S_1 are sampled for disagreement, recognizable by their having opposite states. Any such disagreement between G_i and S_i ($i = 1, 2, \ldots, 6$) is used to turn S_o on

S_o

$$1^{s}_{o} = (G_1 S_1' + G_1' S_1) I_1 N_1 C \quad G_1 S_1' I_2' I_3' N_1 C + G_1' S_1 I_2' I_3' N_1 C$$

$$0^{s}_{o} = T_{41} C$$

Where $I_2' I_3'$ in the final S_o logic includes I_1. Thus, if S_o has not been turned on by the end of the word time, T_{41}, agreement must have been attained and the desired sector has been located.

Command Gating

The memory read gating flip-flop, D_o, is turned on if S_o is true at T_{41}. D_o resets itself to zero at the next T_{41}.

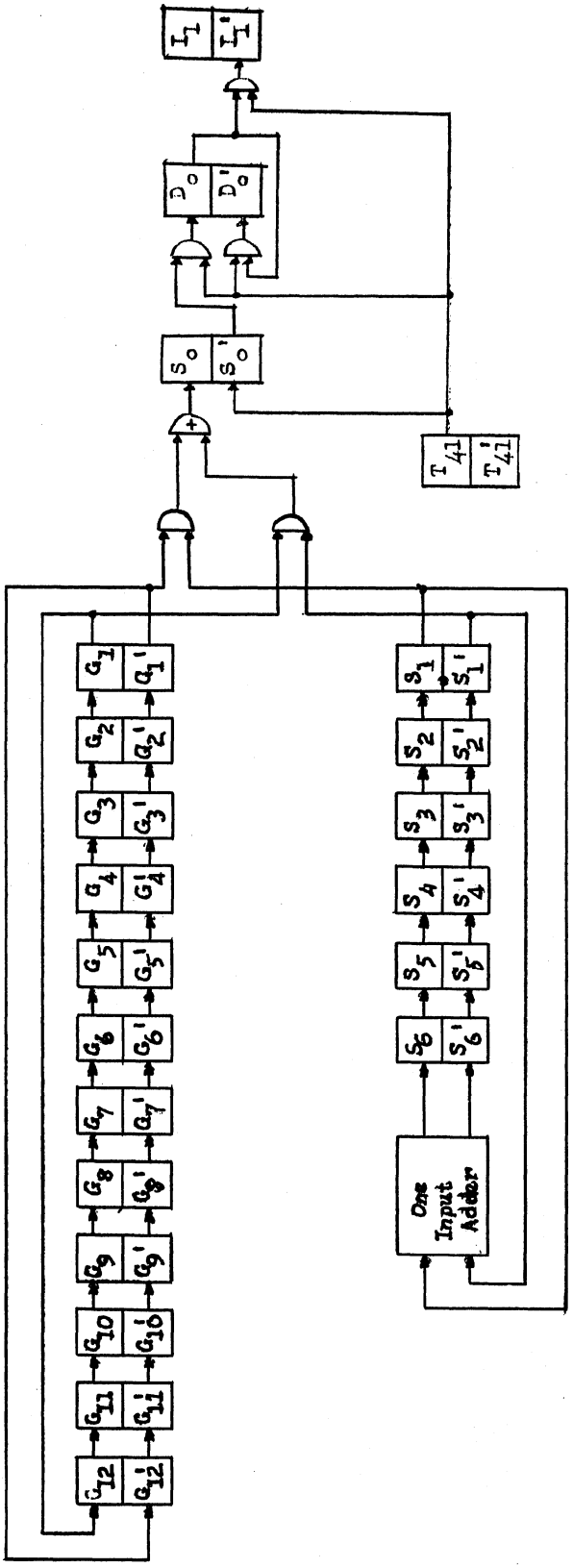


Figure 17. Sector Selection During I_1

D_0

$1^{d}_{o} = S_o {}'L_{p1} {}'I_1 I_4 {}'T_{41} C$
$0^{d}_{o} = D_o T_{41} C$

where $L_{P1}{}'$ refers to a nonvolitile channel. D_o is true for one complete word time (bit times 1 through 41) during which time it gates a command pair from memory into the Z-register.

Z_{41}

$1^{z}_{41} = M_r D_o I_1 C$
$0^{z}_{41} = M_r {}'D_o I_1 C$

where flip-flop Z_{41} is the transmitting end of the 41-bit command register. Since a word is read from memory serially, starting with the least significant bit, it is copied into Z_{41} in the same sequence. Thus, Z_{41} will contain the most significant (sign) bit immediately upon completion of the gating operation, i.e., during $D_o {}'T_1$. In order to conserve equipment in the mechanization of the 41-bit Z-register, only the Z_1, Z_2 and Z_{41} positions are stored in flip-flops. The 38 intervening Z-bits are stored in a recirculating loop on the memory disk. Right-shifting of the Z-register is accomplished by the physical rotation of the disk past stationary reading and writing heads. (See figure 18.) Information from Z_{41} is transferred with no clocked delay to the Z-channel write amplifier.

Z_w

$1^{z}_{w} = Z_{41}$
$0^{z}_{w} = Z_{41}{}'$

The output of Z_w activates the Z-channel write head which magnetizes in time sequence the portion of the disk passing under it. A read head located 38-1/2 bits in the direction of rotation from the write head is energized in time sequence by the same locally magnetized areas passing under it, 38-1/2 bit times later. A Z-channel read amplifier, Z_r, improves the read head signal and is clocked into flip-flop Z_2.

Z_2

$1^{z}_{2} = Z_r C$
$0^{z}_{2} = Z_r {}'C$

Information from Z_2 is shifted into Z_1

Z_1

$_1z_1 = Z_2C$
$_0z_1 = Z_2'C$

When a command pair has been written in the Z-register, the function of command selection is completed. Consequently both I_1 and the gating flip-flop D_o are turned off simultaneously.

$_0i_1 = I_1D_oT_{41}C$

Until I_1 is reestablished, the selected command pair in the Z-register is available for sampling and interpreting. This is easily accomplished by continuously rewriting back on the disk the information that has just been read from it.

$_1z_{41} = Z_1I_1'C$
$_0z_{41} = Z_1'I_1'C$

This one-word recirculation loop allows all the bits in the Z-register to be observed sequentially (in flip-flop Z_1, for example) during each one-word time cycle. In discussing the Z-register small letters z_i represent the information bits in the Z-channel with subscripts corresponding to the bit positions in a word. Illustrated is the quiescent condition existing between consecutive clock pulses $T_{41}C$ and T_1C (i.e., flip-flop T_1 is in the true state). Coincidentally with the clock pulse that set T_1 (i.e. $T_{41}C$), the Z-channel read head sensed the magnetic pattern of z_2 on the disk. At the same time that z_2 was read, it was amplified by Z_r and clocked into flip-flop Z_2. Simultaneously, the previously sensed z_1 bit was clocked from Z_2 into the Z_1 flip-flop, while the z_{41} bit was clocked from Z_1 or M_r into Z_{41} to be written via Z_w onto the disk.

Since the memory read flip-flop M_r receives its pulse train from whatever channel is indicated by the C-register, it is imperative that flip-flops C_1 through C_6 remain static during memory gating. Flip-flop

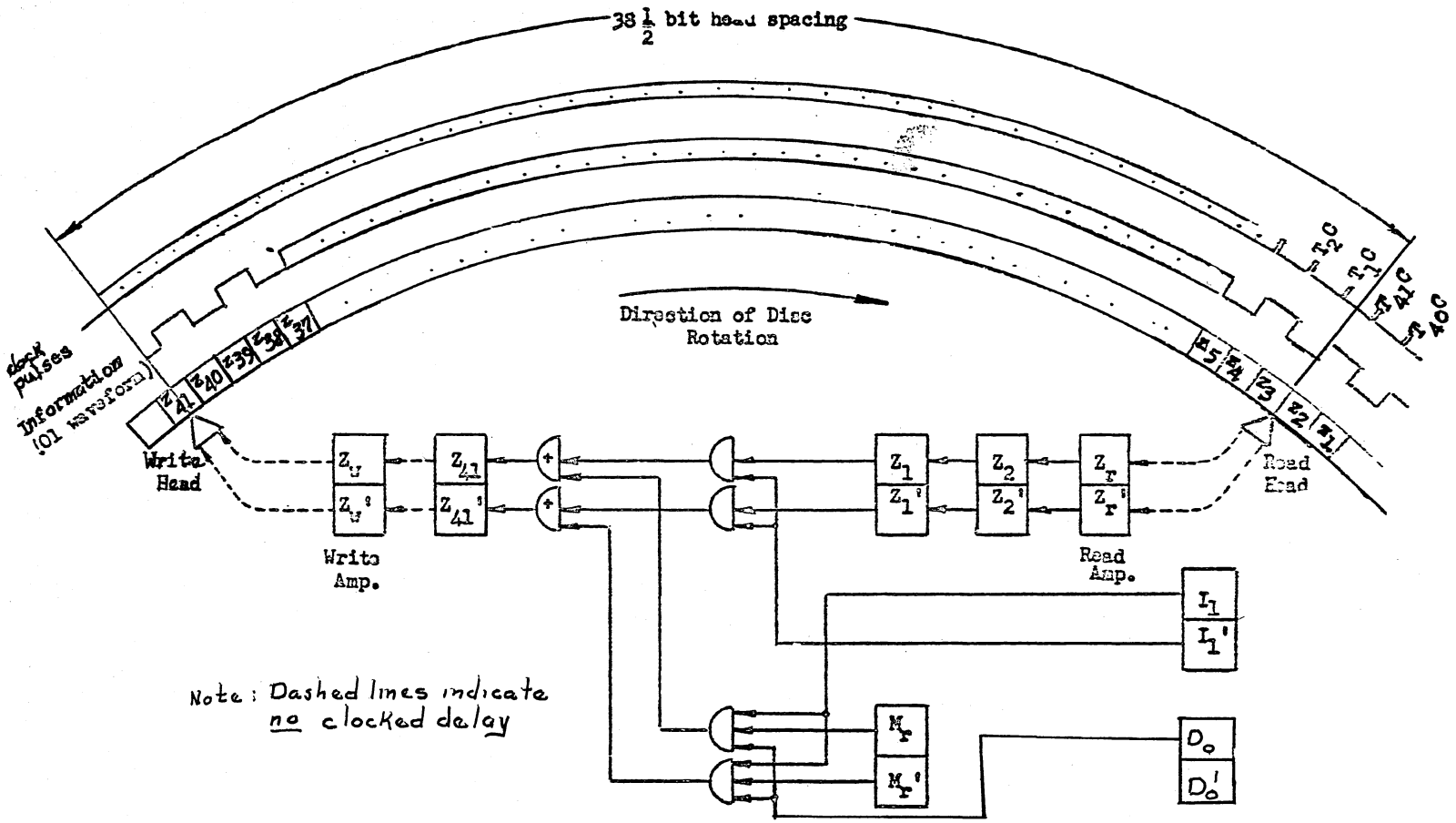


Figure 18. Operation of Z Register

N_5 must be true in order for shifting to occur in the C-register. Therefore N_5 is inhibited from coming true during memory read gating, D_o, as follows.

$$1^{n_5} = D_o' (\ldots)$$

Command Gating from Loops L and V

Both 8-word recirculation loops may contain numerical data and/or commands. If a command to be executed is stored on a loop, the G-counter will contain its loop address. As always, during I_1 the channel address bits are shifted into the C-register and the sector address bits are sampled for coincidence with the S-counter. However, the interpretation of the loop address differs from that of the information channel address as described below.

Loop Selection

Loop addresses are octal 7760 through 7767 for the L loop, and octal 7770 through 7777 for the V loop. In binary notation this would be 1 1 1 1 1 1 1 1 0 0 0 0 - 1 1 1 1 1 1 1 1 0 1 1 1 for the L loop, and 1 1 1 1 1 1 1 1 1 0 0 0 -- 1 1 1 1 1 1 1 1 1 1 1 1 for the V loop. Distinction between loop and main memory is made by L_{p1}, the loop-memory selection flip-flop.

L_{p1}

$1^{1}_{p1} = D_o' D_{11}' I_3' R_{c1}' T_1 C$
$0^{1}_{p1} = G_1' I_1 N_5 C$

Notice that L_{p1} is one-set at T_1 time of I_3'. This would make L_{p1} one-set during T_1 time of the first word time of I_1. Also observe the eight most significant bits of loop address are all ones (the loop address being present in the G-register). This indicates that the occurrence of a zero in G_1 during N_5 time (recall that the G-register circulates during this time T_7 through T_{14} or T_{27} through T_{34}) would indicate that a loop was not desired, and the logic would zero-set L_{p1}. Since there is no logic to one-set L_{p1}' again, L_{p1}' would indicate main memory. On the other hand, if L_{p1} were not zero-set a loop address would be indicated, i.e., L_{p1} would remain one-set.

If a loop is selected, it then remains to determine which loop. L_{p2}, the L-V selection flip-flop,

L_{p2}

$$1^{1}_{p2} = G_1 N_1 P_5 {}'I_1 C$$

$$0^{1}_{p2} = G_1 {}'N_1 P_5 I_1 C$$

is designed to determine this; L_{p2} zero-set indicates the L loop and L_{p2} one-set indicates the V loop. In the binary notation for the loop address that the 4th least significant bit is zero for the L loop and a one for the V loop. This bit is called the loop indicator bit. At T_6 or T_{26} time this bit appears in G_1 of the G-register, and is indicated in the logic by the appearance of $N_1 P_5$. If G_1 is one-set at T_6 or T_{26} time, L_{p2} is one-set during the remainder of N_1. If G_1 is zero-set at T_6 or T_{26} time, L_{p2} is zero-set during the remainder of N_1. $P_2{}'$ is included in the one-set logic for L_{p2} to keep L_{p2} one-set or zero-set during the remainder of N_1.

Loop Design

A detailed description of the high-speed loops is desirable at this point. Since the two loops are identical, the description applies to either L or V. For convenience, however, reference will be made to the L loop.

Loop Recirculation

As seen in figure 19, information on the loop is preserved by recirculation around a path that includes a 325-1/2-bit portion of a memory channel, associated read and write heads and amplifiers, L_{r8} and L_w, and three flip-flops, L_2, L_1 and L_{41}.

The complete recirculation path stores 328 bits which is exactly equal to eight 41-bit words. This closed path is interrupted (between L_1 and L_{41}) only when new information is to be written on the loop; it will be fully described in a later section. When a word is to be read from the loop, as during I_1, it is necessary only to wait until it circulates around to a position that is accessible for gating out. With this method a waiting period of from one to eight word times may be required for access to a random word on the loop.

L_1

$1^{l}_{1} = L_2C$
$0^{l}_{1} = L_2'C$

L_{2a}

$1^{l}_{2a} = L_{r4}$
$0^{l}_{2a} = L_{r4}'$

L_{2a} is the L-loop 4-word read flip-flop.

L_2

$1^{l}_{2} = L_{r8}C$
$0^{l}_{2} = L_{r8}'C$

L_2 is the L-loop 8-word read flip-flop

L_w

$1^{l}_{w} = L_{41}$
$0^{l}_{w} = L_{41}'$

Similar logic applies to the V loop.

V_1

$1^{v}_{1} = V_2C$
$0^{v}_{1} = V_2'C$

V_{2a}

$1^{v}_{2a} = V_{r4}$
$0^{v}_{2a} = V_{r4}'$

V_2

$1^{v}_{2} = V_{r8}C$
$0^{v}_{2} = V_{r8}'C$

V_{2a} is the V-loop 4-word read flip-flop

V_2 is the V-loop 8-word read flip-flop

V_w

$1^{v}_{w} = V_{41}$
$0^{v}_{w} = V_{41}'$

Supplementary Reading Station

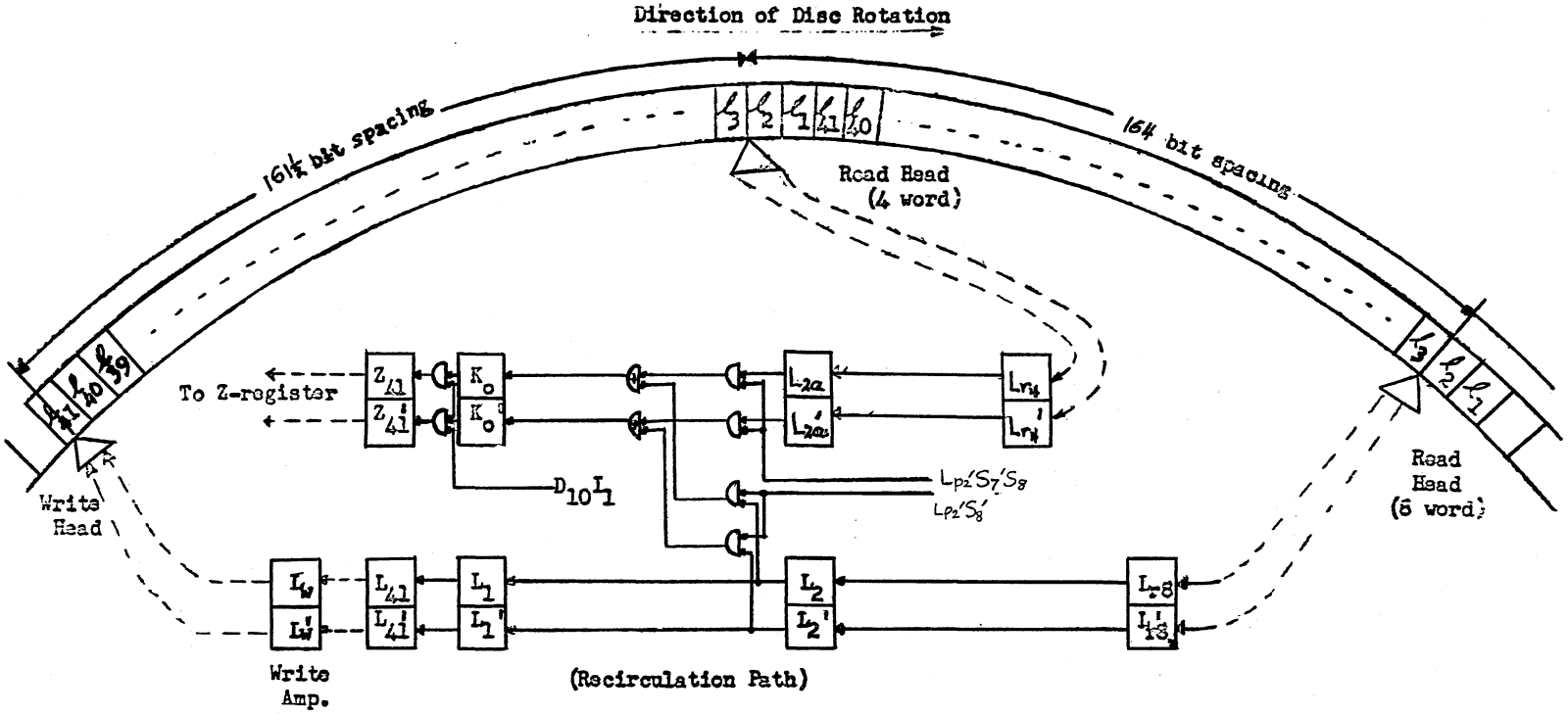
Reduction of access time is achieved by placement of an additional read head within the loop approximately midway between the other heads. With two reading stations four words apart in an 8-word loop, the maximum time before a random word passes under a read head is clearly four word times. Use of supplementary read heads and associated amplifiers L_{r4} and V_{r4} thus improves reading access time by a factor of two. Determining from which head and the exact time to read information from the loop is covered in the next section.

Loop Sector Selection

Since each loop contains eight word sectors, a 3-stage binary counter is needed for sector addressing. Happily, the three least significant stages of the S-counter, S_3, S_2 and S_1, serve this purpose well. This loop sector counter counts eight word times from binary 000 to 111, resets to 000, and repeats indefinitely. The three most significant sector bits are ignored in loop addresses, e.g., sectors 010011 and 111011 both refer to loop sector 3. If the loop sector count is binary 001 when a given word is being written on the loop, it will be 101 when the same word is passing under the 4-word read head, and it will have recycled back to 001 when the word is passing under the 8-word read head. Since the loop sector counter must agree with the corresponding address bits of a given word when it is written on the loop, the same agreement must therefore exist when the word is read from the 8-word read head. It follows that when the desired word is read from the 4-word read head, the loop sector counter must differ by 4 from the corresponding address bits. Conveniently, decimal 4 is binary 100 which, added or subtracted from a binary address, has no effect on the two least significant bits and reverses the next (see table 7).

Table 7. Sector Counting in 8-Word Loops

Loop Sector Address of Word Under:		
Write Head	4-Word Read Head	8-Word Read Head
000	100	000
001	101	001
010	110	010
011	111	011
100	000	100
101	001	101
110	010	110
111	011	111



(Dashed lines indicate no clocked delay)

Figure 19. Operation of 8-Word Loop During I_1

It may be recalled that flip-flop S_o is turned on during I_1N_1 whenever the sector and address bits shifted into S_1 and G_1 disagree. Thus, if S_o is still false after three bits have been sampled (at bit time 6 or 26), then flip-flop S_8 is not set, which indicates that the desired word must be gated from the 8-word read head.

S_8

$$_1s_8 = S_oD_{10}'I_1N_1P_2' \subset S_oD_{10}'M_3'N_1P_2'C$$

$$_0s_8 = S_7T_{41}C + I_1D_{10}T_{41}C$$

Figure 13 reveals that N_1P_2' can occur at either bit time 6 or 26 depending on the state of G_o. S_8 is reset to zero at T_{41}. Also, if S_o is false after only two bits have been sampled (at bit time 5 or 25), flip-flop S_7 is not set.

S_7

$$_1s_7 = S_oD_{10}'I_1N_1P_4P_1'C \subset S_oD_{10}'N_1M_3'P_4P_1'C$$

$$_0s_7 = T_{41a}C$$

Figure 13 shows $N_1P_4P_1'$ is true only at bit time 5 or 25. S_7 is zero-set at every T_{41}. At the end of a word time, if S_7 is false and S_8 is true, then the S- and G-counters agree in the two least significant positions and disagree in the next, indicating that the desired word must be gated from the 4-word read head. Information from either head is copied into a common flip-flop, K_o (figure 19).

K_o

$$_1k_o = L_{2a}L_{p2}'S_7'S_8C + L_2L_{p2}'S_8'C$$

$$_0k_o = L_{2a}'L_{p2}'S_7'S_8C + L_2'L_{p2}'S_8'C$$

where L_{p2}' identifies the L-loop. The use of L_2 instead of L_{r8} is necessitated by circuit restrictions that prevent power amplifiers from driving more than one gate; also L_{2a} is used instead of L_{r4} for the same reason. If $L_{p2} = 1$, the V-loop read heads affect K_o in the same manner.

$$ {}_1k_o = V_{2a} L_{p2} S_7' S_8 C + V_2 L_{p2} S_8' C $$

$$ {}_0k_o = V_{2a}' L_{p2} S_7' S_8 C + V_2' L_{p2} S_8' C $$

Command Gating from Loop

For a loop address, L_{p1} is true, and S_7' indicates that the desired word will appear under one of the read head during the next word time. Consequently, the loop gating flip-flop D_{10} is turned on at T_{41} and remains on for a full word time.

D_{10}

$$ {}_1d_{10} = S_7' L_{P1} D_{10}' I_1 T_{41} C \leftarrow S_7' L_{P1} D_{10}' I_3' (D_1' + D_3' + D_4') T_{41} C $$

$$ {}_0d_{10} = D_{10} T_{41} C $$

During $D_{10} I_1$, the command pair read from the loop into K_o is gated into the Z-register as required.

$$ {}_1z_{41} = K_o D_{10} I_1 C $$

$$ {}_0z_{41} = K_o' D_{10} I_1 C $$

One additional precaution is necessary. During gating it is imperative that K_o continue to copy from the same head. This requires that flip-flops $L_{p2} S_7$ and S_8 not change state while D_{10} is true. L_{p2} is held fixed by L-V indicator bit in the G-register which does not change during I_1.

When reading from the 8-word head, S_8 must remain off and is therefore inhibited from coming on by D_{10}' in ${}_1s_8$. When reading from the 4-word head, S_7 must remain off and is therefore inhibited from coming on by D_{10}' in ${}_1s_7$, while S_8 must remain on and is therefore inhibited from going off by S_7 in ${}_0s_8$. Finally, D_{10}' is inserted in ${}_1d_{10}$ to prevent D_{10} itself from being pulsed on both sides at the conclusion of the gating operation $D_{10} T_{41}$. Command selection I_1 is terminated when gating into the Z-register is completed.

$$ {}_0i_1 = I_1 D_{10} T_{41a} C $$

Command selection, I_1, is concluded and command interpretation, I_2, is initiated when a command pair has been gated into the Z-register from either an information channel by D_0 or a high-speed loop by D_{10}.

I_2

$$1^{i}2 = I_1 D_o T_{41a} C + I_1 I_4' D_{10} T_{41a} C$$

The computer's primary function during I_2 is to select from memory the operand necessary for the execution of arithmetic commands. However, for certain nonarithmetic commands not requiring an operand, this function is superfluous. The command's operation code determines what is accomplished during I_2 and later during I_3.

Operation Code Register

The operation code bits in a command pair (figure10) occupy positions 35 through 40 for the first command and positions 15 through 20 for the second command. During I_2 the selected command pair recirculates in the Z-register so that the least significant bit z_1 is in flip-flop Z_1 during bit time 1; the next bit z_2 is in Z_1 during bit time 2, and in general the n^{th} bit is in Z_1 during bit time n. Recirculation is accomplished by

$$1^{z}41 = Z_1 I_2 C \subset Z_1 I_1' C$$

$$0^{z}41 = Z_1' I_2 C \subset Z_1' I_1' C$$

The operation code bits are available in flip-flop Z_1 during bit times 15 through 20 or 35 through 40; or one bit time earlier in Z_2 they are available during bit times 14 through 19 or 34 through 39. The latter timing is provided by the operation code time flip-flop N_2.

N_2

$_1n_2 = I_2 I_4 ' N_5 P_4 P_2 P_1 ' C$
$_0n_2 = P_2 ' P_1 ' C$

where N_5 is true during the same interval (bit times 9 through 14 or 29 through 34) in I_2 as in I_1

$_1n_5 = I_2 (\ldots) \subset I_3 ' (\ldots)$

Examination of figure 13 shows that the three terms in $_1n_2$ can become true only at bit times 13 or 33, making N_2 true starting with bit times 14 or 34 respectively. $P_2 ' P_1 '$ in $_0n_2$ resets N_2 to zero at bit times 19 or 39. The choice of interval depends on which command is being interpreted. For the left-hand command, $G_o ' = 1$ and N_2 is true during 34 through 39; for the right-hand command, $G_o = 1$ and N_2 is true during 14 through 19. During N_2 the operation code bits are shifted from Z_2 into the operation code register, flip-flops D_1 through D_6

D_1

$_1d_1 = D_2 N_2 C$
$_0d_1 = D_2 ' N_2 C$

D_2

$_1d_2 = D_3 N_2 C$
$_0d_2 = D_3 ' N_2 C$

D_3

$_1d_3 = D_4 N_2 C$
$_0d_3 = D_4 ' N_2 C$

D_4

$_1d_4 = D_5 N_2 C$
$_0d_4 = D_5 ' N_2 C$

D_5

$_1d_5 = D_6 N_2 C$
$_0d_5 = D_6 ' N_2 C$

D_6

$_1d_6 = Z_2 N_2 C$
$_0d_6 = Z_2 ' N_2 C$

It should be noted that N_2 always becomes true during the first word time of I_2 and gates the operation code into the D-register in sufficient time for sampling at the first $I_2 T_{40}$ time.

Types I_2

Sampling of the operation code bits stored in the D-register determines the logical sequence to be followed during I_2. For this purpose the computer commands may be categorized into three major types as follows.

No Memory Reference Required (Type 1)

Commands in this category do not require selection of an operand from memory. For all type-1 operations, the state change flip-flop N_{11} is set at T_{40} and reset at T_{41}.

N_{11}

1^{n}_{11} = Type-1 operation codes $T_{40}C$
$0^{n}_{11} = T_{41}C$

N_{11} being true during T_{41} then terminates I_2 at the end of the first word time

$$0^{i}_{2} = I_2 N_{11} C$$

The I_2 diode in this gate prevents the I_2 flip-flop from being pulsed on both sides when it is set at the end of I_1 since N_{11} could be true at that time.

Referring to the Operation Code Matrix, figure 20, it will be observed that type 1 includes all transfer of control commands (first octal digit = 5), input-output and stop commands (first octal digit = 7), shift and normalize commands (octal codes 40, 41, 45), R-A interchange command (octal code 43) and transfer-to-memory command (octal code 60), and transfer to memory command with X, A interchange (octal code 15). All commands whose first octal code digit is 5 or 7 are covered by the combination D_6D_4. Operation codes 40, 41 and 60 are included in $D_6D_3'D_2'$, operation code 43 and 45 $D_6D_5'D_1$ and operation code 15 in $D_5'D_4D_3D_1T_{40}$. Substituting these gates in the logical equation for N_{11} yields

$$1^{n}_{11} = D_6D_4T_{40}C + D_6D_3'D_2'T_{40}C + D_6D_5'D_1T_{40}C + D_5'D_3D_4D_1T_{40}C$$

	$D_3'D_2'D_1'$	$D_3'D_2'D_1$	$D_3'D_2D_1'$	$D_3'D_2D_1$	$D_3D_2'D_1'$	$D_3D_2'D_1$	$D_3D_2D_1'$	$D_3D_2D_1$
$D_6'D_5'D_4'$	CLA 00 M_0	ADD 01 M_0	CLS 02 M_0	SUB 03 M_0	FAD 04 $M_8M_4(M_0)$	FDV 05 $M_9M_4(M_{10}M_2)$	FSB 06 $M_8M_4(M_0)$	FMP 07 $M_9M_4(M_{10}M_1)$
$D_6'D_5'D_4$	10 M_7	MPY 11 M_1	TYW 12 M_7	MPR 13 M_1	PNW 14 M_7	SAX 15 $M_1(M_6)$	PTW 16 M_7	17 M_1
$D_6'D_5D_4'$	DSL 20 M_2	DSR 21 M_2	DIV 22 M_2	DVR 23 M_2	24 M_2	SQR 25 M_2	26 M_2	27 M_2
$D_6'D_5D_4$	FCA 30 M_3	31 M_3	READOUT 32 BUTTON M_3	EXT 33 M_3	FCS 34 M_3	FST 35 M_3	DIS 36 M_3	37 M_3
$D_6D_5'D_4'$	SHR 40 M_4	SHL 41 M_4	STA 42 $M_4(M_6)$	XAR 43 M_4	FSQ 44 $M_8M_4(M_2)$	FNM 45 M_4	46 M_8M_4	47 M_4
$D_6D_5'D_4$	TZE 50 M_5	TMI 51 M_5	TPL 52 M_5	TOV 53 M_5	TSB 54 M_5	TSC 55 M_5	TSD 56 M_5	TRA 57 M_5
$D_6D_5D_4'$	STO 60 M_6	61 M_6	62 M_6	63 M_6	CTL 64 M_6	CFL 65 M_6	CTV 66 M_6	CFV 67 M_6
$D_6D_5D_4$	FNC 70 M_7	RDY 71 M_7	TYC 72 M_7	RDZ 73 M_7	PNC 74 M_7	75 M_7	PTC 76 M_7	HLT 77 M_7

Figure 20. Operation Code Matrix

Memory Referencing

The address portions of all commands except type 1 refer to memory or loop locations. As during I_1, the C-register receives the channel address bits which select the desired channel for gating, while the S-counter is compared for coincidence with the sector address bits to ascertain the proper gating time. The procedure during I_2 differs from that during I_1 in that the address bits are initially stored in the recirculating Z-register instead of the G-counter. The channel address occupies bit positions 9 through 14 or 29 through 34 and is therefore available in flip-flop Z_1 during the corresponding bit times. As before N_5 provides the necessary timing while information is shifted into the C-register after being copied in C_6.

N_5 provides the necessary timing while information is shifted into the C-register after being copied in C_6.

$$1^{c_i} = C_{i+1} N_5 C$$

$$0^{c_i} = C_{i+1}' N_5 C \qquad i = 1, 2, 3, 4, 5.$$

$$1^{c_6} = Z_1 I_2 N_5 C$$

$$0^{c_6} = Z_1' I_2 N_5 C$$

The sector address is available in Z_1 during bit times 3 through 8 or 23 through 28 for comparison with the corresponding S-counter bits. As in I_1, S_o remains false only when agreement exists during N_1.

$$1^{s_o} = (Z_1 S_1' + Z_1' S_1) I_2 N_1 C \subset Z_1 S_1' I_1' I_4' N_1 C + Z_1' S_1 I_1' I_4' N_1 C$$

$$0^{s_o} = T_{41} C$$

For loop addresses, flip-flops S_7, S_8 and D_{10} are affected by the same logic as during I_1.

Number Selection Without Gating (Type 2)

For commands in this category, I_2 is terminated when the addressee has been located but not yet gated out of the memory. This occurs when the sector address of the command agrees with the sector count. As previously mentioned, this condition is indicated by the false state of the applicable sector selection flip-flop (e.g. S_o) at T_{41}. Gating associated with type 2 commands is performed during I_3.

Figure 20 shows that type-2 commands include the 8-word transfers to or from the L and V loops (octal codes 64 through 67) and the 2-word transfer-to-memory (octal code 35). Of the five, only the CTL and CTV commands (codes 64 and 66) require reading from memory. $D_6D_5D_1'$ is one of several combinations that covers these codes. Inclusion of type-1 codes in the combination causes no difficulty since such codes terminate I_2 after the first word time. N_{11} is set at T_{40} when the desired sector is ready for gating from memory.

$$1^n_{11} = S_o'D_6D_5D_1'T_{40}C$$

As with type-1 commands, N_{11} immediately terminates I_2 at T_{41}.

The remaining type-2 commands (octal codes 35, 65, 67) require writing on the memory. S_{11} is the sector selection flip-flop used for memory writing and corresponds to S_o for memory reading.

The nonvolatile memory channels utilize separate read and write heads spaced 180 $\widehat{\text{deg}}$ or 32 sectors apart. Therefore the sector addresses of words passing under the read and write heads differ by 32 (= binary 100000). It follows that the binary addresses agree in all but the most significant bit. Since the S-counter is synchronized to indicate the addresses of words under the read-head, words under the write head may be identified by the sector counter with its most significant bit reversed. S_{11}, initially zero, is turned on if the desired sector address in G or Z and the S-counter disagree in any of the first five bits or agree in the sixth.

S_{11}

$$ {}_1s_{11} = S_o N_1 C + S_o{}' P_6 P_5 P_4 $$

$$ {}_0s_{11} = T_{41} C $$

Thus if S_o is still false after five bits have been sampled (at bit time 8 or 28) and becomes true after the sixth bit has been sampled (at bit time 9 or 29), then S_{11} is not set, indicating that the desired sector is the next to appear under the write head. If any of the above conditions is not met, S_{11} is set, indicating that the desired sector is not about to appear under the write head at this time. S_{11} is reset to zero every T_{41}.

Returning to the type-2 commands that require memory writing, clearly S_{11}' and an appropriate D-register combination are sufficient to terminate I_2. Because this group of commands includes codes 35, 65, and 67, figure 20 reveals $D_6 D_1$ and $D_4 D_3 D_1$ as the configurations. Thus

$$ {}_1n_{11} = S_{11}{}' D_4 D_3 D_1 T_{40} C + S_{11}{}' D_6 D_1 T_{40} C $$

One additional possiblity must be considered. The 2-word transfer-to-memory command, FST (octal code 35), could contain a loop address. Since S_8 is the sector selection flip-flop associated with loop writing and $D_4 D_3 D_1$ covers octal code 35, we may write

$$ {}_1n_{11} = S_8{}' L_{p1} D_4 D_3 D_1 T_{40} C $$

where L_{p1} indicates a loop address.

Number Selection with Gating (Type 3)

This category includes all commands not covered by types 1 and 2. Here the operand is located and gated out of memory during I_2. As during I_1, the applicable sector selection flip-flop (S_o or S_7) indicates by its false state at T_{41} the correct time to turn on the memory or loop gating flip-flop (D_0 or D_{10}).

$1^{d}_{o} = S_o' L_{p1}' I_2 T_{41} C \subset S_o' L_{p1}' I_1 I_4' T_{41} C$
$1^{d}_{10} = S_7' L_{p1} I_2 T_{41} C \subset S_7 L_{p1} I_3' T_{41} C$

Instead of being gated into the Z-register as during I_1, the addressee is gated into the B-register during I_2. This is a one-word recirculating register consisting of four flip-flops (B_1, B_2, B_{40}, B_{41}) and a portion of the disk storing the remaining 38 bits between the B read and write heads. Thus the B-register differs from the Z-register only in storing one more bit in a flip-flop and therefore one less bit on the disk. Information from a memory channel or loop is gated into B_{41} as follows:

B_{41}

$1^{b}_{41} = M_r D_0 I_2 C + K_0 D_{10} I_2 C$
$0^{b}_{41} = M_r' D_0 I_2 C + K_0' D_{10} I_2 C$

B_{40} copies B_{41}.

B_{40}

$1^{b}_{40} = B_{41} I_2 C \subset B_{41} M_3' M_0' C$
$0^{b}_{40} = B_{41}' I_2 C \subset B_{41} M_3' M_0' C$

where M_o' in the final logic is always true in I_2. B_{40} is connected directly to the B-channel write amplifier, B_w.

B_w

$1^{b}_{w} = B_{40}$
$0^{b}_{w} = B_{40'}$

At the reading end of the B-channel, the read power amplifier B_r feeds into B_2 which in turn shifts into B_1.

B_2

$1^{b}_{2} = B_r C$
$0^{b}_{2} = B_r' C$

B_1

$$ {}_1b_1 = B_2 I_2 C \subset B_2 M_{10}{}' N_7{}' E_{30}{}' C $$

$$ {}_0b_1 = B_2 I_2 C \subset B_2{}' M_{10}{}' N_7{}' E_{20}{}' E_{30}{}' C $$

where $E_{20}{}' E_{30}{}'$ is true except when filling the computer.

Having gated the desired number into the B-register, D_0 or D_{10} is reset.

$$ {}_0d_0 = D_0 T_{41} C $$

$$ {}_0d_{10} = D_{10} T_{41} C $$

At the same time I_2 is terminated by N_{11}.

$$ {}_1n_{11} = D_0 T_{40} C + D_{10} T_{40} C $$

$$ {}_0\bar{i}_2 = I_2 N_{11} C $$

COMMAND EXECUTION, I_3

During command execution, I_3, the arithmetic and logical operations specified by the commands are actually carried out. I_3 is initiated at the same time I_2 is terminated.

$$ {}_1i_3 = J_0{}' I_2 N_{11} C $$

where J_0, the overflow flip-flop, is normally false.

Functional Groups

The computer's commands separate roughly into eight functional groups that are identified by flip-flops M_i ($i = 0, 1, \ldots, 7$) where i is the most significant octal digit of the operation code (see figure 20). Three additional flip-flops, M_{8-10}, specify floating point operations. The D-register, having been filled with the operation code bits during I_2, is used to set the appropriate M flip-flop simultaneously with the start of I_3.

M_0 $_1m_0 = (D_6{}'D_5{}'D_4{}'D_3{}') J_0{}'I_2N_{11}C$

M_1 $_1m_1 = (D_6{}'D_5{}'D_4) D_1J_0{}'I_2N_{11}C$

M_2 $_1m_2 = (D_6{}'D_5D_4{}') J_0{}'I_2I_4{}'N_{11}C$

M_3 $_1m_3 = (D_6{}'D_5D_4) J_0{}'I_2N_{11}C$

M_4 $_1m_4 = (D_6D_5{}'D_4{}') J_0{}'I_2N_{11}C$

M_5 $_1m_5 = (D_6D_5{}'D_4) J_0{}'I_2N_{11}C$

M_6 $_1m_6 = (D_6D_5D_4{}') J_0{}'I_2N_{11}C$

M_7 $_1m_7 = (D_6D_5D_4) J_0{}'I_2N_{11}C$

M_8 $_1m_8 = (D_5{}'D_4{}'D_3D_1{}') I_2N_{11}J_0{}'C$

M_9 $_1m_9 = (D_6{}'D_5{}'D_4{}'D_3D_1) I_2N_{11}J_0{}'C$

M_{10} $_1m_{10} = M_9M_4U_2T_{41a}C$

I_3 Timing

Most of the commands require a fixed number of word times for their execution during I_3. It is necessary therefore to keep count of these word times in order to terminate the operations at the proper time. A computation cycle counter consists of flip-flops Q_1 through Q_6

and an associated carry flip-flop, Kq, to form a one-input adder which counts word times in the same manner as the sector counter. The Q-counter counts only during I_3.

Q_1

$$_1q_1 = Q_2N_1M_8' + M_8X_{41}T_1'C$$

$$_0q_1 = Q_2'N_1M_8' + I_5T_{41}C + M_8T_1C$$

Q_2

$$_1q_2 = Q_3N_1M_8' + Q_1'M_8T_{40}C + X_{41}M_8U_1'T_1C$$

$$_0q_2 = Q_3'N_1M_8' + I_5T_{41}C$$

Q_3

$$_1q_3 = Q_4N_1M_8' + M_8I_5'U_1'C_2'T_{40}C$$

$$_0q_3 = Q_4'N_1M_8' + U_1T_{41}C + M_{10}T_{41}C$$

Q_4

$$_1q_4 = Q_5N_1M_8' + M_8M_0'I_5'U_1T_{41}C$$

$$_0q_4 = Q_5'N_1M_8' + M_8Q_5'U_2T_{41}C + M_8U_1'C_1'T_{41}C$$

Q_5

$$_1q_5 = Q_6N_1M_8' + M_8D_6'X_2'R_2U_2C + M_8D_6'U_2X_2'Q_5'T_{40}C$$

$$_0q_5 = Q_6'N_1M_8' + M_8X_2R_2'U_2C + M_8U_2R_2'Q_5T_{40}C$$

Counter Operation During $M_8'M_0'M_4'M_{10}' + M_8'M_0'M_9'$ Commands

Inspection of the logic equations for Q_1 through Q_5 indicates that the usual one-input adder shift logic is present except for M_8-type commands. M_8 commands are FAD, FSB, and FSQ. These commands use some of the Q flip-flops for various tests required by the floating-point mechanization. The addition of M_8 to gates for the logic of Q_1 through Q_5 will be discussed later with the floating commands.

Inspection of the logic for Q_6 reveals that this flip-flop has the usual one-input adder recirculation logic (similar to the logic for S_6 of the sector counter; plus the additional terms $P_m C + I_3'C$ in the zero-set logic. The $P_m C$ term is involved in the tape punch output mode, therefore we will neglect its presence for compute operation. $I_3'C$ determines that Q_6 through Q_1 will be in their zero-set states by the start of I_3. The I_3 terms in the one-set logic determine how long the counter will count. This, in turn, depends on the word times required by the command in I_3.

The logic for the carry flip-flop K_q indicates that the carry will one-set at T_1 (similar to K_s;) for $M_0'M_4'M_{10}' + M_0'M_9'$. These gates are to exclude operation of the counter during the portions of the floating-point instructions that require the Q flip-flops for testing. The zero-set logic for K_q is the usual one-input adder logic.

U_{41}, U_2, and U_1 Flip-Flops (Neglecting FDV, FMP, FAD, FSB, FSQ)

At the beginning of I_3 the U_{41}, and U_2, flip flops are in their zero-set state by the logic:

$$_0U_{41} = U_{41}T_{41a}C$$

$$_0U_2 = U_2T_{41}C$$

The U_1 flip-flop is in its one-set state by the beginning of I_3 by the logic:

$$_1U_1 = I_3'C + M_0U_2T_{41a}C$$

Table 8 gives the counting sequence.

Table 8. Counting Sequence

TIME	LOGIC INVOLVED	U_{41}	U_2	U_1
T_1I_3 Word 1	${}_0U_{41} = U_{41}T_{41a}C$ ${}_0U_2 = U_2T_{41}C$ ${}_1U_1 = I_3'C +$	0	0	1
T_1I_3 Word 2	${}_1U_2 = A_t'I_3U_1T_{41}I_5'N_{12}'C$ ${}_0U_1 = I_3T_{41a}U_1I_5'C$	0	1	0
T_1I_3 Word 3	${}_0U_2 = U_2T_{41}C \quad {}_1U_1 = M_0U_2T_{41a}C$	0	0	0, 1
T_1I_3 Word 41	${}_1U_{41} = M_2D_1'Q_6Q_4Q_1'T_{41a}C +$ For DSL and DIV	1	0	0
T_1I_3 Word 42	${}_1U_{41} = M_2D_1Q_6Q_4Q_1T_{41a}C +$ For DSR, DVR, SQR	1	0	0
T_1I_3'	${}_0U_{41} = U_{41}T_{41a}C$ ${}_1U_1 = I_3'C$	0	0	0
T_2I_3'	${}_1U_1 = I_3'C +$	0	0	1

Q_6

${}_1q_6 = Q_1K_q'N_1I_3P_m'C + Q_1'K_qN_1I_3P_m'C$
${}_0q_6 = Q_1'K_q'N_1C + Q_1K_qN_1C + P_mC + M_8C + I_3'C$

K_q

${}_1k_q = T_1M_0'M_4'M_{10}'C + T_1M_0'M_9'C$
${}_0k_q = Q_1'N_1C$

Inasmuch as frequent reference must be made to the first and second word times in I_3, special timing flip-flops, designated U_1 and U_2, help to simplify the logic associated with such times. U_1 is true during I_3' and the first word time in I_3, while U_2 is true only during the second word time in I_3. U_{41} is true during the last word time which may be 41 or 42 in I_3 for certain type-3 commands.

U_1	$1^{u}_{1} = I_3{}'C + M_8U_1{}'C_1{}'T_{41a}C + M_oU_2T_{41a}C + M_9M_1{}'M_2{}'U_2T_{41a}C + M_9B_{42}Q_6T_{41a}C$
	$0^{u}_{1} = I_3U_1I_5{}'T_{41a}C$

U_2	$1^{u}_{2} = I_3U_1I_5{}'N_{12}{}'A_t{}'T_{41}C$
	$0^{u}_{2} = U_2T_{41}C$

U_{41}	$1^{u}_{41} = M_2D_1{}'Q_6Q_4Q_1{}'T_{41a}C + M_2D_1Q_6Q_4Q_1T_{41a}C$
	$0^{u}_{41} = U_{41}T_{41a}C$

Accumulator (A) and Remainder (R) Registers

The A- and R-registers are available for use by the programer and, together with the B-register, hold operands and intermediate and final results of all arithmetic operations performed by RECOMP. Like the B- and Z-registers, they are one-word recirculating loops which store part of their information in flip-flops and the rest on the disk. The A-register has six flip-flops, A_1, A_2, A_3, A_{39}, A_{40} and A_{41}, plus a 35-bit portion of the disk, while the R-register has four flip-flops, R_1, R_2, R_{40} and R_{41}, plus a 37-bit portion of the disk. Results produced by execution of a given command must be preserved until needed as operands for the next computation. This is accomplished if the A- and R-registers exhibit normal recirculation during I_3.

In the A-register, A_1 copies A_2 except at T_{41} when A_1 is set to zero. During T_{41}, A_2 contains the synchronizing bit which must be zero for certain operations.

A_1	$1^{a}_{1} = A_2I_3{}'T_{41}{}'(E_{20}{}'E_{30}{}'E_{40}{}')C$
	$0^{a}_{1} = A_2{}'I_3{}'E_{30}{}'C + I_2T_{41}C$

where the E terms can be true only when filling the computer. A_2 in turn copies A_3.

A_2

$1^{a}{}_{2} = A_3 I_3 {}' E_{20} {}' E_{40} {}' C$
$0^{a}{}_{2} = A_3 {}' I_3 {}' E_{20} {}' E_{40} {}' C$

A_3 receives its information from the A-channel read power amplifier, A_r.

A_3

$1^{a}{}_{3} = A_r C$
$0^{a}{}_{3} = A_r {}' C$

On the other side of the A-channel, the write amplifier, A_w, is directly connected to A_{39}.

A_w

$1^{a}{}_{w} = A_{39}$
$0^{a}{}_{w} = A_{39} {}'$

which copies A_{40}

A_{39}

$1^{a}{}_{39} = A_{40} U_1 + T_1 + M_1 {}') C$
$0^{a}{}_{39} = A_{40} {}' U_1 + T_1 + M_1 {}') C$

which in turn copies A_{41}

A_{40}

$1^{a}{}_{40} = A_{41} I_3 {}' C \subset A_{41} M_0 {}' M_1 {}' U_{41} {}' C$
$0^{a}{}_{40} = A_{41} {}' I_3 {}' C \subset A_{41} {}' M_0 {}' M_1 {}' U_{41} {}' C$

Finally A_{41} copies A_1, closing the loop.

A_{41}

$1^{a}_{41} = A_1 I_3 {}' P_0 {}' C$
$0^{a}_{41} = A_1 {}' I_3 {}' P_0 {}' C$

P_0 is true only during filling.

During recirculation in the R-register, R_1 copies R_2 which receives information from the R-channel read power amplifier, R_r.

R_1

$1^{r}_{1} = R_2 I_3 {}' P_0 {}' C$
$0^{r}_{1} = R_2 {}' I_3 {}' P_0 {}' C$

R_2

$1^{r}_{2} = R_r C$
$0^{r}_{2} = R_r {}' C$

The R-channel write amplifier, R_w, receives its signal from R_{40}

R_w

$1^{r}_{w} = R_{40}$
$0^{r}_{w} = R_{40}{}'$

which copies R_{41}.

R_{40}

$1^{r}_{40} = R_{41} E_{30} {}' U_{41} {}' C$
$0^{r}_{40} = R_{41} {}' E_{30} {}' U_{41} {}' C$

R_{41} copying R_1 completes the recirculation.

R_{41}

$1^{r}_{41} = R_1 I_3 {}' P_0 {}' C$
$0^{r}_{41} = R_1 {}' I_3 {}' P_0 {}' C$

Clear-Add (CLA) and Clear-Subtract (CLS) Commands

The command CLA causes the word in memory location m to be transferred to the A-register. The command CLS differs only in that the sign of the word in m is reversed in the transfer to the A-register. The combination M_0D_1' is true for both commands, with D_2' or D_2 added to specify the CLA or CLS commands respectively.

Since these commands belong to I_2 type-3, the word in memory location m is stored in the B-register during I_2. It suffices therefore during I_3 to shift the B-register contents into the A-register, taking required action on the algebraic sign. This procedure is illustrated for the CLA-CLS commands by figure 21. The B-register is shifted right and the sign bit in B_{41} is duplicated in B_{40} for CLA command and reversed in B_{40} for CLS command.

For CLA

$1^b_{40} = B_{41}D_3'D_2'D_1'C$
$0^b_{40} = B_{41}'D_3'D_2'D_1'C$

For CLS

$1^b_{40} = B_{41}'M_0D_2D_1'C$
$0^b_{40} = B_{41}M_0D_2D_1'C$

The rest of the B-register follows the normal right shift.

B_w

$1^b_w = B_{40}$
$0^b_w = B_{40}'$

B_2	$_1b_2 = B_rC$
	$_0b_2 = B_r'C$

B_1	$_1b_1 = B_2M_0C \subset B_2M_{10}'N_7'E_{30}'C$
	$_0b_1 = B_2'M_0C \subset B_2M_{10}'N_7'E_{20}'E_{30}'C$

Everything from the B-register including the sign is copied from B_1 into A_{41} and on through the A-register.

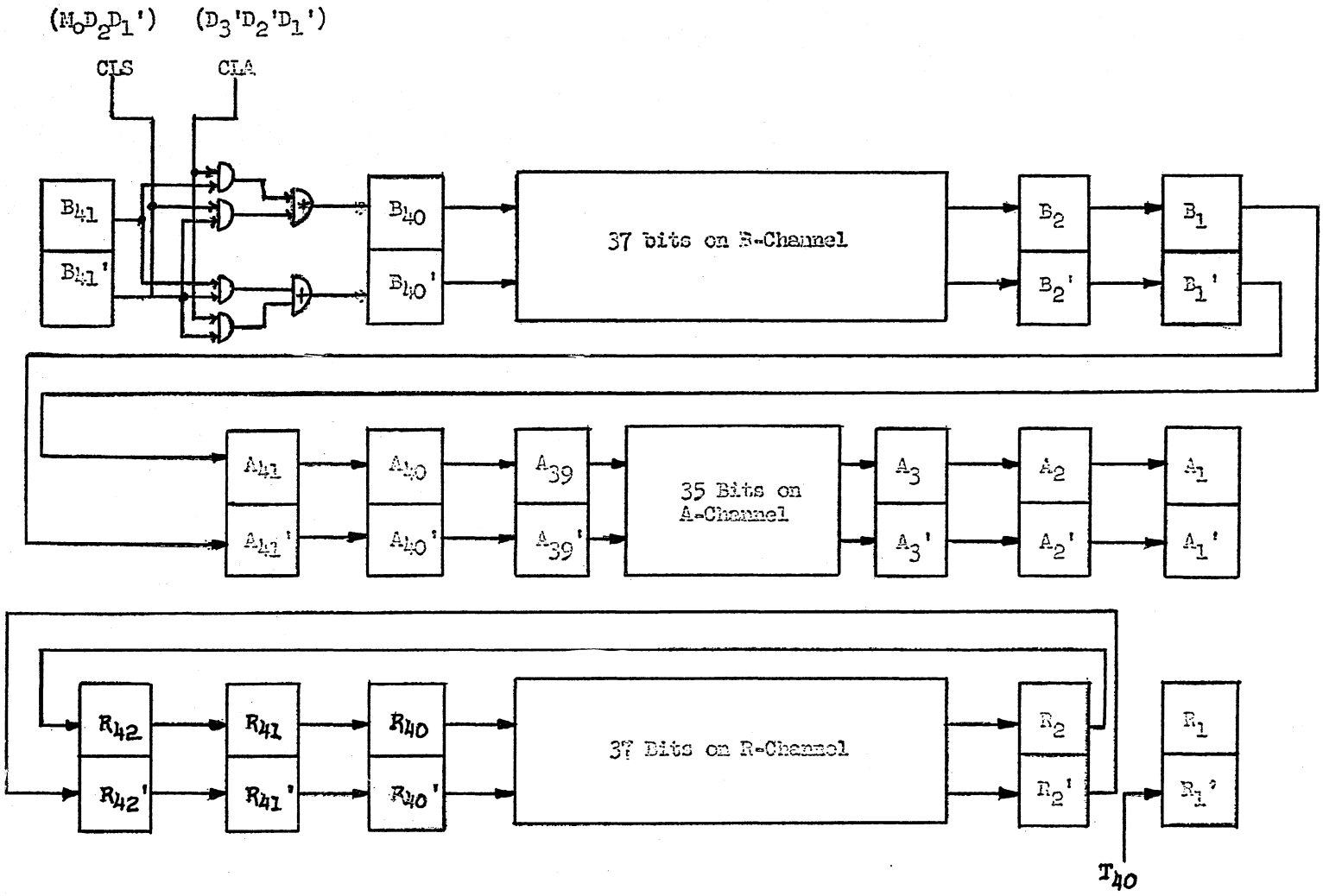


Figure 21. Operation of CLA-CLS Commands

A_{41}

$1^{a}_{41} = B_1 M_0 D_1 'C$
$0^{a}_{41} = B_1 ' M_0 D_1 'C$

A_{w}

$1^{a}_{w} = A_{39}$
$0^{a}_{w} = A_{39}'$

A_{40}

$1^{a}_{40} = A_{41} M_0 U_1 C$
$0^{a}_{40} = A_{41} ' M_0 U_1 C$

A_{3}

$1^{a}_{3} = A_r C$
$0^{a}_{3} = A_r 'C$

A_{39}

$1^{a}_{39} = A_{40} (U_1 + T_1 + M_1^{*}) C$
$0^{a}_{39} = A_{40} '(U_1 + T_1 + M_1^{*}) C$

A_{1}

$1^{a}_{1} = A_2 I_3 M_2 ' M_{10} ' T_{41} 'C$
$0^{a}_{1} = A_2 ' I_3 M_2 ' M_{10} 'C$

B_{40} receives the sign bit stored in B_{41} while the entire A-register contents are shifted-right one more bit. For a CLA command, B_{41} is duplicated in B_{40}.

$1^{b}_{40} = B_{41}$ (CLA command) $C = B_{41} D_3 ' D_2 ' D_1 'C$
$0^{b}_{40} = B_{41}'$ (CLA command) $C = B_{41} ' D_3 ' D_2 ' D_1 'C$

whereas for a CLS command, B_{41} is reversed in B_{40}

$1^{b}_{40} = B_{41}$ (CLS command) $C = B_{41} ' M_0 D_2 D_1 'C$
$0^{b}_{40} = B_{41}'$ (CLS command) $C = B_{41} M_0 D_2 D_1 'C$

The R-register recirculates the information. Because R_1 is used as a gate, the recirculation takes place from R_2 to R_{42}.

R_{42}

$1^{r}_{42} = R_2 M_0 C$
$0^{r}_{42} = R_2' M_0 C$

R_{41}

$1^{r}_{41} = R_{42} M_0 Q_3' C$
$0^{r}_{41} = R_{42}' M_0 Q_3' C$

R_{40}

$1^{r}_{40} = R_{41} M_0 C \subset R_{41} E_{30}' U_{41}' C$
$0^{r}_{40} = R_{41}' M_0 C \subset R_{41}' E_{30}' U_{41}' C$

R_w

$1^{r}_{w} = R_{40}$
$0^{r}_{w} = R_{40}'$

R_2

$1^{r}_{2} = R_r C$
$0^{r}_{2} = R_r' C$

The R-register contents at the end of U_1 will be identical with that of the R-register one word earlier. Also, to force a zero sync bit in R_1,

$$0^{r}_{1} = M_0 T_{40} C$$

The functions of the CLA and CLS commands are completed in a single word time, therefore the state change flip-flop N_{12} is set at T_{40} for one bit time and terminates I_3 and M_0 at T_{41}

N_{12}

$1^{n}_{12} = M_0 D_1' T_{40} C$
$0^{n}_{12} = T_{41a} C$
$0^{i}_{3} = N_{12} C$
$0^{m}_{0} = N_{12} C$

It will be observed that N_{12} is used to terminate I_3 in the same manner that N_{11} was used to terminate I_2.

Add (ADD) and Subtract (SUB) Commands

The command ADD causes the word in memory location m to be algebraically added to the word in the A-register. The command SUB causes the word in m to be algebraically subtracted from the word in the A-register. The R-register is unaffected by either command. M_0D_1 identifies this pair of commands, with D_2 being false for ADD and true for SUB (see figure 20). Being type-3 commands, the word in m is gated into the B-register during I_2. It is therefore necessary, during I_3, for the B-register contents to be added to or subtracted from the A-register contents and the result stored back in A.

The question of whether to perform absolute addition or subtraction depends not only upon the nature of the command but upon the algebraic signs of the operands as well. Addition of absolute values is required for ADD commands if the operands have the same sign and for SUB commands if the operands have opposite signs. Subtraction of absolute values is indicated for SUB commands if the operands have the same sign and for ADD commands if the operands have opposite signs. Flip-flop E_o indicates whether absolute addition or subtraction is required: E_o indicates addition; E_o' indicates subtraction. The signs of the operands are in flip-flops A_{41} and B_{41} at T_1.

E_o

$_1e_o = N_5C$
$_0e_o = A_{41}B_{41}\overbrace{'M_0D_2}^{\text{Add}}{}'U_1T_1C + A_{41}'B_{41}M_0D_2'U_1T_1C +$ $A_{41}B_{41}\underbrace{M_0D_2}_{\text{Sub}}C + A_{41}'B_{41}'M_0D_2C$

Serial Binary Adder

It is well at this point to consider the characteristics of serial binary addition and subtraction of absolute values.

Serial addition means simply adding one digit position at a time starting from the least significant end and working toward the most significant end, each time recording the sum digit and temporarily storing the carry digit for use at the next digit time. It may be noted that this is the usual pencil-and-paper method of addition.

Assume the addend and augend are initially stored in the A- and B-registers, and during addition these registers are shifted-right so that flip-flops A_1 and B_1 contain corresponding bits at any given time. The bits in A_1, B_1, and the carry from the previous bit time in flip-flop K_a, determine the sum bit, S, and new carry bit. Figure 22 lists the eight possibilities that may occur.

Inputs, Time t			Outputs, Time (t + 1)	
A_1	B_1	K_a	S	K_a
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Figure 22. Truth Table for Binary Adder

Inspection of figure 22 yields the following logic for the sum flip-flop, S,

S

$$_1s = A_1{}'B_1{}'K_aC + A_1{}'B_1K_a{}'C + A_1B_1{}'K_a{}'C + A_1B_1K_aC$$

$$_0s = A_1{}'B_1{}'K_a{}'C + A_1{}'B_1K_aC + A_1B_1{}'K_aC + A_1B_1K_a{}'C$$

Note in figure 22 that the carry flip-flop, K_a, changes from 0 to 1 only when A_1 and B_1 are both true, and from 1 to 0 only when A_1 and B_1 are both false, or

K_a

$_1k_a = A_1B_1C$
$_0k_a = A_1'B_1'C$

For other combinations of A_1 and B_1, K_a remains in the same state. K_a must always be zero when the addition is started.

If the number in the B-register is to be subtracted from the number in the A-register, the truth table in figure 23 applies. Note that the "difference" column is identical in all cases with the S column in the adder, which means that the S logic can also be used to generate the difference when performing subtraction. K_a, which now contains the "borrow" bit, changes from 0 to 1 only when $A_1'B_1$ is true and from 1 to 0 only when A_1B_1' is true. Thus the K_a logic may be expanded to cover both absolute addition and subtraction.

Inputs, time t			Outputs, time (t + 1)	
A_1	B_1	K_a	Difference	K_a
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Figure 23. Truth Table for Subtractor

K_a

$_1k_a = E_0A_1B_1C + E_0'A_1'B_1C$
$_0k_a = E_0A_1'B_1'C + E_0'A_1B_1'C$

(Carry borrow logic)

Suppose we desire a more versatile logic system capable of addition and subtraction involving either the A- and B-registers or the A- and R-registers. This is exactly the requirement in RECOMP where the operands, normally in A and B, occur in A and R when converting decimal to binary during fill. Instead of duplicating equipment with two

adders, the one described above may be modified to perform both jobs by utilizing R_1 and B_1 as control flip-flops to gate the B- and R-registers respectively into the adder. It is necessary merely to replace B_1 by the "AND" combination B_1R_1 in the logic for S and K_a above. B_1' then becomes $(B_1R_1)' = B_1' + R_1'$ (by DeMorgan's Theorem). The final logic then becomes

(adder-subtractor logic)

S

$$_1s = A_1'B_1'K_aC + A_1'R_1'K_aC + A_1'B_1R_1K_a'C + A_1B_1'K_a'C + A_1R_1'K_a'C + A_1B_1R_1K_aC$$

$$_0s = A_1'B_1'K_a'C + A_1'R_1'K_a'C + A_1'B_1R_1K_aC + A_1B_1'K_aC + A_1R_1'K_aC + A_1B_1R_1K_a'C$$

K_a

$$_1k_a = E_oA_1B_1R_1M_4'T_{41}'C + E_o'A_1'B_1R_1M_4'T_{41}'C$$

$$_0k_a = E_oA_1'B_1'M_4'T_1'C + E_oA_1'R_1'M_4'C + E_o'A_1B_1'M_4'C + E_o'A_1R_1'M_4'C + T_{41}C$$

where $T_{41}C$ in the $_ok_a$ logic and T_{41}' in the $_1k_a$ terms ensure an initial zero carry. Also, M_4' is obviously true for add-subtract commands, and T_1' in the $_0k_a$ term is needed for divide commands and causes no difficulty here since only the sync (zero) bits are sampled at T_1. If we set $R_1 = 1$ in this logic it reduces to the simple A + B adder previously considered. Similarly $B_1 = 1$ yields the logic for an A + R adder. It may be noted further that if $R_1 = 0$, the sum logic becomes

$$_1s = A_1B_1'K_a'C + A_1K_a'C + A_1'B_1'K_aC + A_1'K_aC$$

$$_0s = A_1'B_1'K_a'C + A_1'K_a'C + A_1B_1'K_aC + A_1K_aC$$

which may be further reduced to

$1^{s} = A_1 K_a {}'C + A_1 {}'K_a C$
$0^{s} = A_1 {}'K_a {}'C + A_1 K_a C$

If $K_a = 0$ also, S will copy A_1. Thus, depending on whether $R_1 = 1$ or 0, S will generate the sum A + B or copy A, a fact that will prove extremely useful in multiplication.

First Computation Cycle, U_1

Having mastered the fundamental serial adder-subtractor, we are now in a position to consider the detailed I_3 logic associated with the ADD and SUB commands. At the start of I_3, the A- and B-registers contain the operands in their normal positions, $E_0 = 1$ indicating absolute addition, and $K_a = 0$.

$1^{e}{}_0 = N_5 C$
$0^{k}{}_a = T_{41} C$

Figure 24 illustrates the events that transpire during U_1 for ADD commands. The diagram can also apply to SUB commands by interchanging the B_{41} output lines. The first clock pulse, $U_1 T_1 C$, sets $R_1 = 1$ for gating into the adder, sets $A_{41} = 0$ for clearing the A-register to zero, and sets $E_0 = 0$ if absolute subtraction is required.

$1^{r}{}_1 = M_0 T_1 C$
$0^{a}{}_{41} = M_0 U_1 T_1 C$

Operands in both the A- and B-registers are shifted-right during U_1.

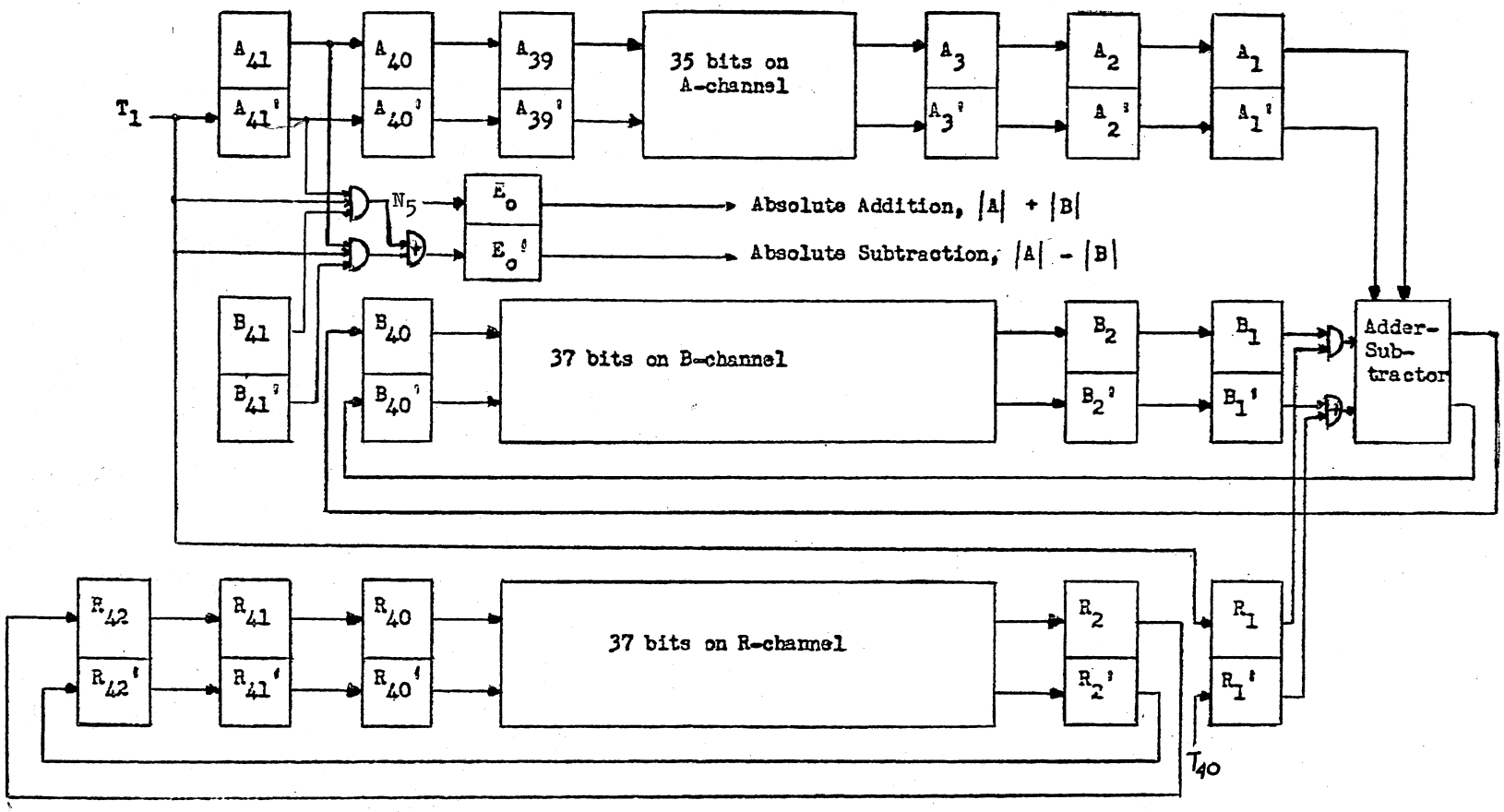


Figure 24. Operation of ADD Command During U_1

A_{40}

$1^{a}_{40} = A_{41}M_0U_1C$
$0^{a}_{40} = A_{41}'M_0U_1C$

A_1

$1^{a}_{1} = A_2M_0C \subset A_2I_3M_{10}'M_2'T_{41}'C$
$0^{a}_{1} = A_2'M_0C \subset A_2'I_3M_{10}'M_2'T_{41}'C$

A_{39}

$1^{a}_{39} = A_{40}M_1'C$
$0^{a}_{39} = A_{40}'M_1'C$

B_w

$1^{b}_{w} = B_{40}$
$0^{b}_{w} = B_{40}'$

A_w

$1^{a}_{w} = A_{39}$
$0^{a}_{w} = A_{39}'$

B_2

$1^{b}_{2} = B_rC$
$0^{b}_{2} = B_r'C$

A_3

$1^{a}_{3} = A_rC$
$0^{a}_{3} = A_r'C$

B_1

$1^{b}_{1} = B_2M_0C \subset B_2M_{10}'N_7'E_{30}'C$
$0^{b}_{1} = B_2'M_0C \subset B_2'M_{10}'N_7'E_{20}'E_{30}'C$

A_2

$1^{a}_{2} = A_3M_0C \subset A_3I_3M_2'C$
$0^{a}_{2} = A_3'M_0C \subset A_3'I_3M_2'C$

R_1 being true during bit times 2, 3, . . . , 40, the significant operand bits a_2, a_3, ... a_{40}, and b_2, b_3, . . . , b_{40}, appear successively in flip-flops A_1 and B_1 respectively for gating into the serial adder-subtractor. The sum or difference is generated serially in flip-flop S and then re-written back in B_{40}.

$1^{b}_{40} = SM_0D_1C$
$0^{b}_{40} = S'M_0D_1C$

The sum digits are now shifted through the B-register by the logic for B_w, B_2 and B_1 above. Note that the sum is written in B_{40} rather than

B_{41}, because the S flip-flop in the adder introduces a delay of one bit time. This delay must be compensated for by shortening the B-register by one flip-flop, B_{41}. (S may be considered as replacing B_{41} in the 41-bit B-register).

The zero written in flip-flop A_{41} at T_1 is propagated down through the A-register by the regular shift logic above, and the A-register is cleared since R_1 is being used for gating, The contents of the R-register are preserved by recirculation from R_2 to R_{42} instead of the normal R_1 to R_{41}.

R_{42}

$$1^{r}_{42} = R_2 M_0 C$$

$$0^{r}_{42} = R_2' M_0 C$$

R_{41}

$$1^{r}_{41} = R_{42} M_0 Q_3' C$$

$$0^{r}_{41} = R_{42}' M_0 Q_3' C$$

R_{40}

$$1^{r}_{40} = R_{41} M_0 C \subset R_{41} E_{30}' U_{41}' C$$

$$0^{r}_{40} = R_{41}' M_0 C \subset R_{41}' E_{30}' U_{41}' C$$

R_w

$$1^{r}_{w} = R_{40}$$

$$0^{r}_{w} = R_{40}'$$

R_2 | $1^r2 = R_rC$ | $0^r2 = R_r'C$

The most significant non-sign bits, a_{40} and b_{40}, are gated into the adder-subtractor at $T_{40}C$, and R_1 is turned off at this time.

$$0^r1 = M_0T_{40}C$$

At T_{41}, $R_1 = 0$ reduces the sum logic to

$$1^s = (A_1K_a' + A_1'K_a)\,T_{41}C$$

$$0^s = (A_1'K_a' + A_1K_a)\,T_{41}C$$

at which time A_1 contains the sign of A, and K_a the carry or borrow bit resulting from the addition or subtraction of a_{40} and b_{40}.

Summarizing the situation at the end of the first computation cycle, U_1, the A-register has been cleared to zero, the B-register contains $|A| \pm |B|$ and the R-register is unchanged. Interpretation of the new B-register contents in terms of S, K_a and E_0 occurs in the second computation cycle, U_2.

Second Computation Cycle, U_2

Depending upon the relative signs and magnitudes of the original operands, four distinct cases will be considered during U_2. These are summarized in figure 25 and described in detail below. Figure 26 illustrates the general flow of information during U_2.

Case I: A and B have the same sign (i.e., $|A| + |B|$ is performed during U_1) and $|A| + |B| < 1$.

Since RECOMP considers the binary point between the sign and most significant bit, $|A| + |B| < 1$ means that there is no carry bit left over from the addition of a_{40} and b_{40} so that $K_a = 0$ at U_1T_{41} at which time S copies the sign of A from flip-flop A_1. Obviously in Case I, both A and the sum A + B have the same sign which, if positive (binary 1), sets flip-flop A_{41} at the next bit time.

$$1^a41 = SM_0J_0'U_2T_1C$$

where $J_0 = 0$ for Case I. If the sign is negative (binary 0), no logic is needed since A_{41} was zeroed in U_1.

During U_2 the zero contents of the A-register and the sum $|A| + |B|$ in the B-register are added, since E_0 remains true, and the resulting sum, again $|A| + |B|$, is written back in the A-register.

A_{40}

$$1^a40 = SM_0U_2C$$

$$0^a40 = S'M_0U_2C$$

Writing occurs in A_{40} instead of A_{41} to compensate for the one-bit delay in the adder. Logic for shifting the A- and B-registers and recirculating the R-register is the same during U_2 as U_1. R_1 is also set for gating B into the adder as in U_1.

Case II: A and B have the same sign (i.e., $|A| + |B|$ is performed during U_1) and $|A| + |B| \geq 1$.

In this case, $|A| + |B| \geq 1$ means that a "one" carry bit is left in K_a at U_1T_{41} from the addition of the most significant bits a_{40} and b_{40}. The overflow flip-flop, J_0, is set and activates the neon overflow light on the computer's control panel whenever such an addition overflow occurs.

J_o

$$1^j0 = E_0K_aM_0U_1T_{41}D_1C$$

The overflow carry bit is the integer part of $|A| + |B|$ and has the numerical value ± 1, while the quantity remaining in the B-register at the end of U_1 is the fractional part $|A| + |B| - 1$. With E_0 remaining on during U_2, this B-register content is added to the A-register's zero contents and the sum is written back in A.

Since $K_a = 1$ and $R_1 = 0$ at U_1T_{41}, the sign of A is reversed in flip-flop S. But the sign of the result should agree with the sign of A in this case. Therefore, at the next bit time, U_2T_1, another sign reversal occurs in the transfer from S to A_{41}.

$$1^a41 = S'M_0J_0U_2T_1C$$

where J_0 identifies Case II. Again, no logic is needed for the minus sign since A_{41} was already zero.

Case III: A and B have opposite signs (i.e., $|A| - |B|$ is performed during U_1) and $|A| \geq |B|$.

During U_1 the smaller absolute value was subtracted from the larger leaving no overborrow in K_a at U_1T_{41}. Hence, there is no sign reversal in S nor does J_0 get turned on, and the Case I term

$$1^a41 = SM_0J_0\ 'U_2T_1C$$

copies the sign of A as required for Case III.

During U_1 the correct result, $|A| - |B|$, was produced in the B-register while the A-register was cleared to zero. Thus the adder must be set for addition in U_2,

$$1^e0 = K_a\ 'M_0\ T_{41a}C$$

From the above it is clear that Case III requires identically the same logic as Case I during U_2.

Case IV: A and B have opposite signs (i.e., $|A| - |B|$ is performed during U_1) and $|A| < |B|$.

During U_1 the larger absolute value was subtracted from the smaller leaving an overborrow in K_a at U_1T_{41}. Thus the contents of the B-register are actually the result of subtracting $|B|$ from $1 + |A|$, or A - B + 1 as indicated in figure 25. The A-register is cleared to zero during U_1 as usual. If we repeat absolute subtraction during U_2, where again the larger absolute value is in the B-register (A - B + 1 = 0), an overborrow recurs. As seen during U_1, this is equivalent to subtracting the B-register contents from the A-register contents augmented by 1. Thus the U_2 result, written back in the A-register, becomes (1 + 0) - (A - B + 1) = B - A as desired. With $K_a = 1$ at U_1T_{41}, E_o remains false during U_2, thereby assuring subtraction.

For Case IV, clearly the sign of the original B-register contents should prevail. At U_1T_{41} the sign of A was reversed in S because of $K_a = 1$. Also, J_0 is not set since $E_0 = 0$ for Case IV. Therefore at U_2T_1 the reversed sign of A is copied into A_{41} by

$$1^a41 = SM_0J_0\ 'U_2T_1C$$

Since A and B originally had opposite signs, this reversed sign of A is the correct sign of B exactly as required.

Case	Signs of A & B	Relative Magnitudes	During U_1					During U_2					
			A-reg.	E_o set for	B-reg.	S generates	K_a at T_{A1}	A-reg.	E_o set for	B-reg.	S generates	J_o	Final Sign
I	Same	$\|A\|+\|B\|<1$	A	+	B	$\|A\|+\|B\|$.	0	0	+	$\|A\|+\|B\|$	$\|A\|+\|B\|$	0	Same as A
II	Same	$\|A\|+\|B\|\geq 1$	A	+	B	$\|A\|+\|B\|-1$	1	0	+	$\|A\|+\|B\|-1$	$\|A\|+\|B\|-1$	1	Same as A
III	Opposite	$\|A\|\geq\|B\|$	A	-	B	$\|A\|-\|B\|$	0	0	+	$\|A\|-\|B\|$	$\|A\|-\|B\|$	0	Same as A
IV	Opposite	$\|A\|<\|B\|$	A	-	B	$\|A\|-\|B\|+1$	1	0	-	$\|A\|-\|B\|+1$	$\|B\|-\|A\|$	0	Same as B

Figure 25. Summary of ADD and SUB Command Cases

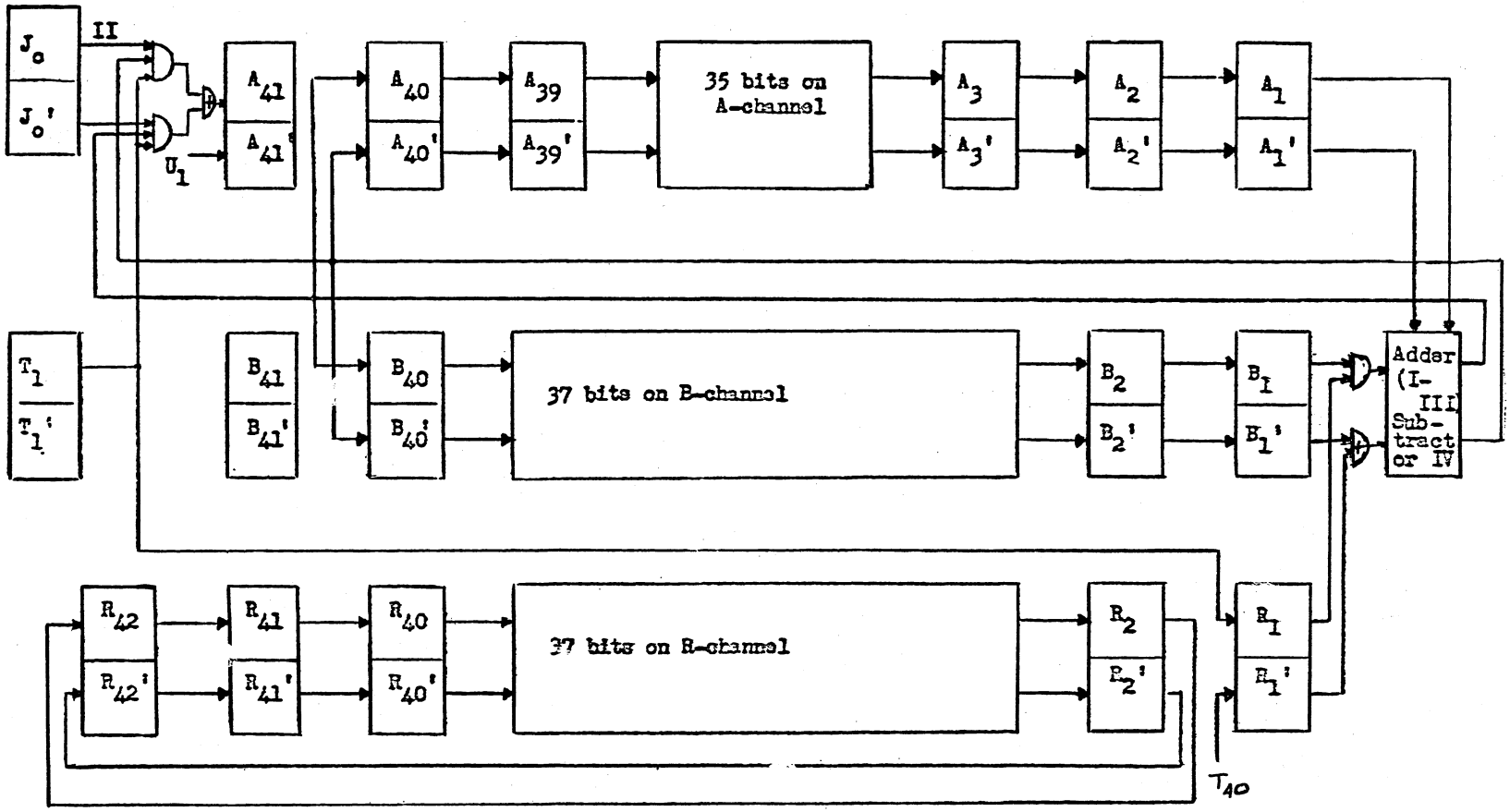


Figure 26. Operation of ADD and SUB Commands During U_2

CASE I: Example of ADD Command with Absolute Addition Required

(ADD)	+13	(ADD)	+.1101
	+ 2		+.0010
	+15		+.1111

		E_o	K_a	S	A 6 5 4 3 2 1	B 6 5 4 3 2 1	R 1
	T_1	1	0	X	+11010	+00100	X
	T_2	1	0	0	0+1101	+X0010	1
U_1	T_3	1	0	1	00+110	+0X001	1
	T_4	1	0	1	000+11	+10X00	1
	T_5	1	0	1	0000+1	+110X0	1
	T_6	1	0	1	00000+	+1110X	0
	T_1	1	0	÷	000000	+11110	0
	T_2	1	0	0	+÷0000	+÷1111	1
U_2	T_3	1	0	1	+0÷000	+0÷111	1
	T_4	1	0	1	+10÷00	+10÷11	1
	T_5	1	0	1	+110÷0	+110÷1	1
	T_6	1	0	1	+1110+	+1110+	0
	T_1	1	0	X	+11110	+11110	0

Notes:
R_6 through R_2 irrelevant

X--Irrelevant bit

0--Means flip-flop zero-set by logic term

B_{41} Remains static in I_3

Other terms in A & B are standard re-circulating logic.

Important Logic:

$1^e0 = N_5C$

$1^r1 = M_0T_1C$

$0^r1 = M_0T_{40}C$

$1^a41 = SM_0J_0{}'U_2T_1C$

$0^a41 = M_0U_1T_1C$

$1^b41 = SM_0D_1C$

$0^b41 = S'M_0D_1C$

$1^a40 = SM_0U_2C$

$0^a40 = S'M_0U_2C$

$0^ka = T_{41}C$

$1^s = (A_1K_a{}' + A_1{}'K_a)\,T_{41}C$

$0^s = (A_1{}'K_a{}' + A_1K_a)\,T_{41}C$

CASE III: Example of ADD Command with Absolute Subtraction Required

(ADD)	+13	(ADD)	+.1101
	- 2		-.0010
	+11		+.1011

		E_o	K_a	S	A 654321	B 654321	R 1
	T_1	1	0	X	+11010	-00100	X
	T_2	0	0	0	0+1101	-X0010	1
	T_3	0	0	1	00+110	-0X001	1
U_1	T_4	0	1	1	000+11	-10X00	1
	T_5	0	0	0	0000+1	-110X0	1
	T_6	0	0	1	00000+	-0110X	0
	T_1	1	0	1	000000	-10110	0
	T_2	1	0	0	++0000	-+1011	1
U_2	T_3	1	0	1	+0+000	-0+101	1
	T_4	1	0	1	+10+00	-10+10	1
	T_5	1	0	0	+110+0	-110+1	1
	T_6	1	0	1	+0110+	-0110+	0
	T_1	1	0	X	+10110	-10110	0

Notes:
R_6 -R_2 irrelevant

X--Irrelevant bit

0--Means flip-flop zero-set by logic term

B_{41} remains static in I_3

Other terms in A & B are standard re-circulating logic

Important Logic:

$$1^e_0 = N_5C + K_a{}'M_0\,T_{41a}\,C$$
$$0^e_0 = A_{41}B_{41}{}'M_0D_2{}'U_1T_1C$$

$$1^r_1 = M_0T_1C$$
$$0^r_1 = M_0T_{40}C$$

$$1^a_{41} = SM_0J_0{}'U_2T_1C$$
$$0^a_{41} = M_0U_1T_1C$$

$$1^b_{40} = SM_0D_1C$$
$$0^b_{40} = S'M_0D_1C$$

$$1^a_{40} = SM_0U_2C$$
$$0^a_{40} = S'M_0U_2C$$

$$1^s = (A_1K_a{}' + A_1{}'K_a)\,T_{41}C$$
$$0^s = (A_1{}'K_a{}' + A_1K_a)\,T_{41}C$$

$$0_{ka} = T_{41}C$$

CASE IV: Example of SUB Command with Absolute Subtraction Required

(Subtract) +2 (Subtract) +.0010
+3 +.0011
-1 -.0001

		E_o	K_a	S	A 6 5 4 3 2 1	B 6 5 4 3 2 1	R 1
	T_1	1	0	X	+00100	+00110	X
	T_2	0	0	0	0+0010	+X0011	1
	T_3	0	1	1	00+001	+0X001	1
U_1	T_4	0	1	1	000+00	+10X00	1
	T_5	0	1	1	0000+0	+110X0	1
	T_6	0	1	1	00000+	+1110X	0
	T_1	0	0	0	000000	+11110	0
	T_2	0	0	0	-00000	+01111	1
U_2	T_3	0	1	1	-00000	+00111	1
	T_4	0	1	0	-10000	+10011	1
	T_5	0	1	0	-01000	+01001	1
	T_6	0	1	0	-00100	+00100	0
	T_1	0	0	X	-00010	+00100	0

Notes:
R_6-R_2 irrelevant

X--irrelevant bits

0--means flip-flop zero-set by logic term

B_{41} remains static in I_3

Other terms in A & B are standard re-circulating logic

Important Logic:

$1^e0 = N_5\ C$

$0^e0 = A_{41}B_{41}M_0D_2C$

$1^r1 = M_0T_1C$

$0^r1 = M_0T_{40}C$

$1^a41 = SM_0J_0{}'U_2T_1C$

$0^a = M_0U_1T_1C$

$1^{b40} = SM_0D_1C$

$0^{b40} = S'M_0\ D_1C$

$1^{a40} = SM_0U_2C$

$0^{a40} = S'M_0U_2C$

$1^s = (A_1K_a{}' + A_1{}'K_a)\ T_{41}C$

$0^s = (A_1{}'K_a{}' + A_1K_a)\ T_{41}C$

$0^{ka} = T_{41}C$

The ADD and SUB commands having been completed at the end of U_2, N_{12} is set which terminates I_3.

$$1^n 12 = M_0 M_8 {}' U_2 T_{40} C$$

$$0^i 3 = N_{12} C$$

Multiply Unrounded (MPY) and Multiply Round (MPR) Commands

The command MPY causes the word in memory location m to be multiplied by the word in the A-register to obtain a double-length (78-bit) product whose most- and least-significant halves are stored in the A-register and R-register, respectively. MPR performs the same operation and rounds-off the product to the portion in the A-register, i. e., 39 binary places. Flip-flop M_1 identifies the multiply commands (figure 20) with D_2' and D_2 indicating the MPY and MPR commands respectively.

Several things may be noted from the preceding example. A 1 or 0 multiplier bit, times the multiplicand, yields the multiplicand itself or a string of 0's respectively. This means that a partial product is formed by either adding or not adding the shifted multiplicand to the previous partial product. In RECOMP this addition is performed by the same adder that is used for the ADD and SUB commands. The option of adding or not adding the multiplicand, stored in the B-register, to the partial product, stored in the A-register, is provided by flip-flop R_1 which gates B_1 into the adder. R_1 must therefore contain the applicable multiplier bit. This is accomplished by initially placing the multiplier in the R-register and subsequently shifting it one bit to the right before performing each addition.

Note also that the length of the partial product starts at one word and increases by one or two bits for each multiplier bit until it becomes two words long as the final product. This appears to indicate that another arithmetic register is required. Fortunately, once a multiplier bit has performed its function, it is no longer needed and may therefore be sloughed off to make room for an additional partial product bit. Thus, the next least significant partial product bit is right-shifted out of A and into R each time a multiplier bit is discarded from R until the final double-length product occupies the entire A- and R-registers as indicated in figure 27.

Furthermore, shifting the A-register one bit to the right is equivalent to shifting the B-register one bit to the left when adding A and B. The first method has the advantage of being a natural outcome

of the process mentioned in the preceding paragraph and at the same time simplifies the B-register logic to that of normal recirculation.

Serial Binary Multiplier

Multiplication of binary numbers follows the same pattern as multiplication of decimal numbers as demonstrated in the examples of figure 27. Starting at the least significant end, digits of the multiplicand are successively multiplied by the least significant digit of the multiplier and the result recorded. The process is repeated using each digit of the multiplier in turn, with the result at each step being shifted-left corresponding to the position of the multiplier digit. When all multiplier digits have been exhausted, the results are added together to obtain the final product.

Ignoring signs for the moment, consider the operation in terms of the computer. The addition process that yields the final product would appear to require an adder having 39 inputs, not to mention temporary storage for 39 words. This problem is overcome by performing the addition as the multiplication proceeds, obtaining a new partial product after each multiplier bit is used up. In figure 27 for example the least significant multiplier bit (1) times the multiplicand (=1010) yields the first partial product (=1010). The second least significant multiplier bit (0) times the multiplicand (=1010) yields the quantity 0000 which is shifted one bit left and added to the first partial product to obtain the second partial product (=01010). In general the n^{th} least significant multiplier bit times the multiplicand is shifted and added to the $(n-1)^{th}$ partial product to obtain the n^{th} partial product. When finally n becomes equal to the number of bits in the multiplier, the n^{th} partial product is the final product and the multiplication process is completed. In the example shown, the multiplier (=1101) has four bits; hence the fourth partial product is the final product.

First Computation Cycle, U_1

Multiplication being I_2 type-3 commands, the multiplicand was read from memory and stored in the B-register during I_2, which is exactly where it is required for the operation. Before starting the multiplication process, the multiplier must be transferred from the A- to the R-register, and the A-register must be cleared to zero before accumulating any partial products. These preliminary operations are performed during the first computation cycle, U_1, as illustrated in figure 28. The A-register is cleared by setting A_{41} to zero at T_1 and shifting right. Simultaneously the A-register contents are transferred serially to the R-register by having R_{41} copy A_2 while the R-register is likewise shifted right.

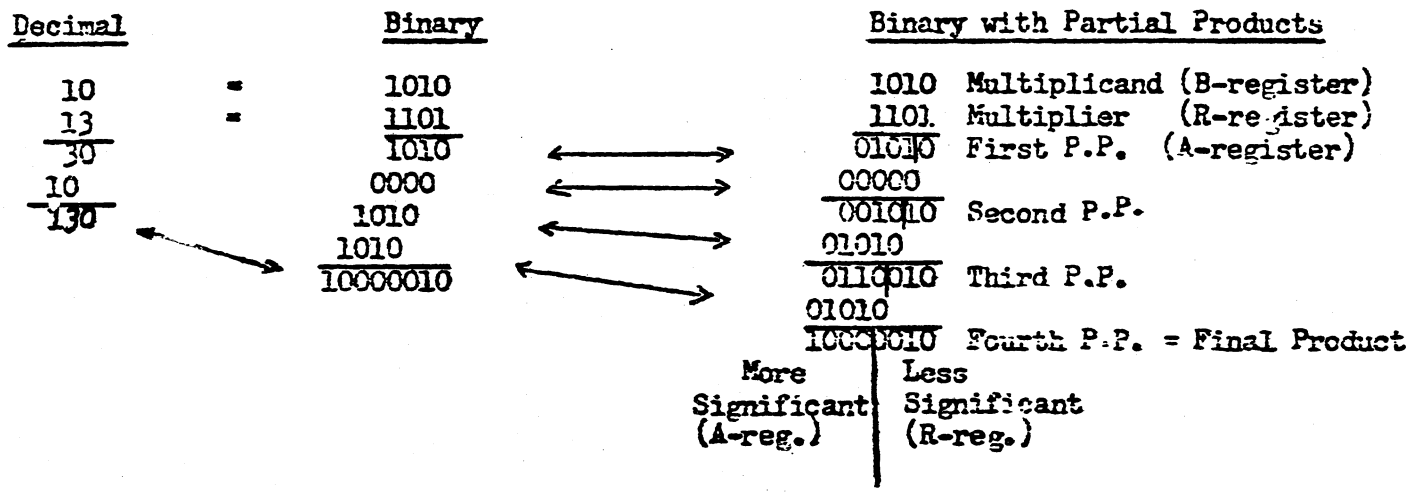


Figure 27. Multiplication Example

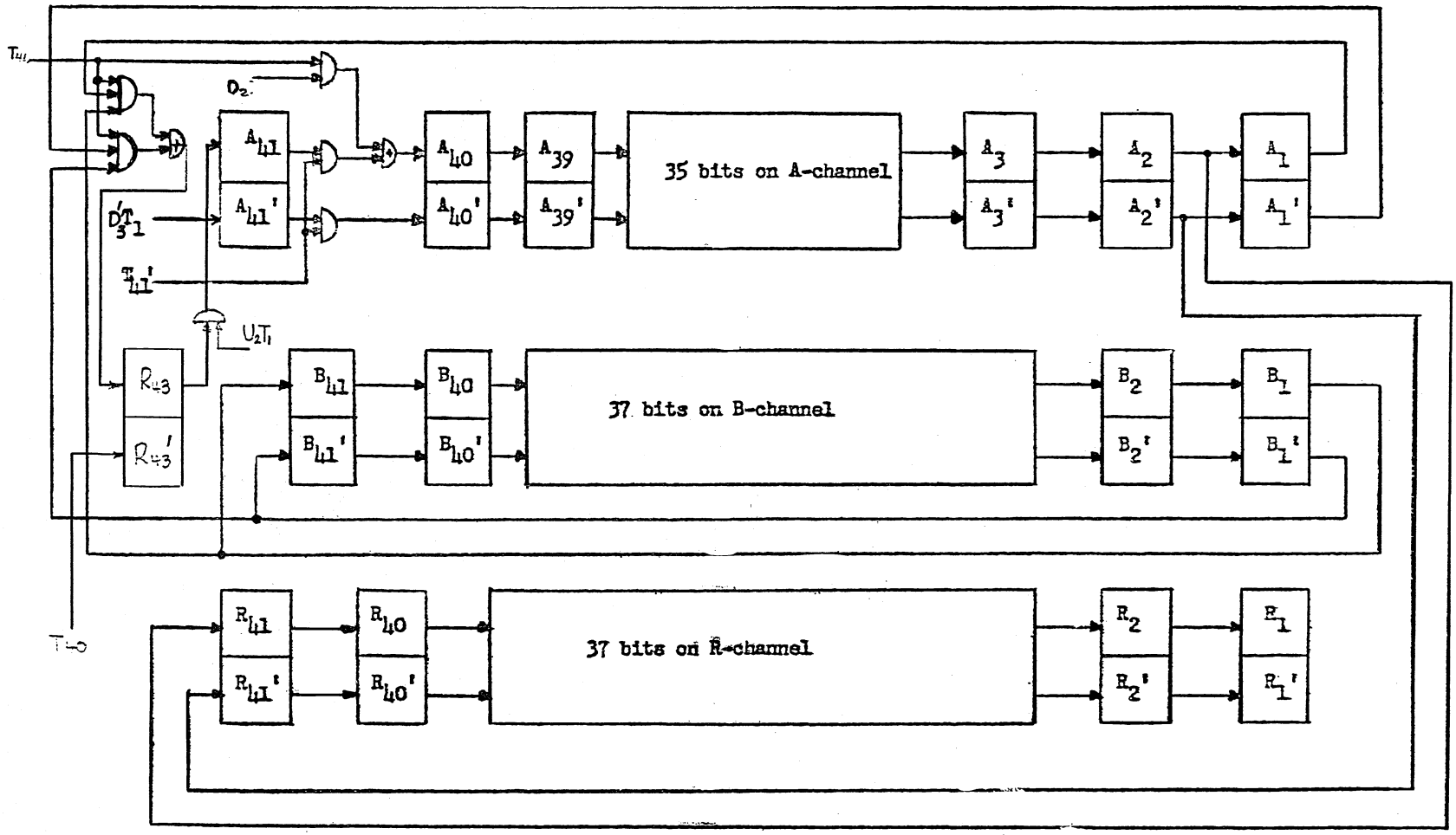


Figure 28. Multiplication During U_1

	$0^{a}41 = M_1 D_3 \, 'T_1 C$
A_{40}	$1^{a}40 = A_{41} M_1 U_1 C$
	$0^{a}40 = A_{41} \, 'M_1 T_{41} \, 'C$
A_{39}	$1^{a}39 = A_{40} U_1 C$
	$0^{a}39 = A_{40} \, 'U_1 C$
A_2	$1^{a}2 = A_3 M_1 C \subset A_3 I_3 M_2 \, 'C$
	$0^{a}2 = A_3 \, 'M_1 C \subset A_3 \, 'I_3 M_2 \, 'C$
R_{41}	$1^{r}41 = A_2 M_1 D_3 \, 'C$
	$0^{r}41 = A_2 \, 'M_1 D_3 \, 'C$
R_{40}	$1^{r}40 = R_{41} M_1 C \subset R_{41} E_{30} \, 'U_{41} \, 'C$
	$0^{r}40 = R_{41} \, 'M_1 C \subset R_{41} \, 'E_{30} \, 'U_{41} \, 'C$
R_1	$1^{r}1 = R_2 M_1 U_1 C$
	$0^{r}1 = R_2 \, 'M_1 U_1 C$

Note that A_1 was by-passed in the preceding process. Consequently, at the end of U_1 the entire multiplier will be displaced one bit to the right in R with respect to its original orientation in A, i.e., the least significant bit a_2 will be in R_1, a_3 in R_2, ..., the sign bit a_{41} in R_{40}, and zero in R_{41}. Obviously the original contents of R are destroyed during U_1.

The multiplicand in the B-register is preserved by recirculation.

B_{41}	$1^{b}41 = B_1 M_1 C \subset B_1 T_x \, 'M_0 \, 'M_8 \, 'M_{10} \, 'I_3 C$
	$0^{b}41 = B_1 \, 'M_1 C \subset B_1 \, 'T_x \, 'M_0 \, 'M_8 \, 'M_{10} \, 'I_3 C$
B_{40}	$1^{b}40 = B_{41} M_1 C \subset B_{41} M_0 \, 'M_3 \, 'C$
	$0^{b}40 = B_{41} \, 'M_1 C \subset B_{41} \, 'M_0 \, 'M_3 \, 'C$

B_1

$$1^b1 = B_2M_1C \subset B_2M_{10}'N_7'E_{30}'C$$

$$0^b1 = B_2'M_1C \subset B_2'M_{10}'N_7'E_{20}'E_{30}'C$$

The multiplication example of figure 27 is detailed in figure 29 to show exactly what takes place in the computer. For convenience, this 6-bit example has been chosen for illustrative purposes in lieu of a standard 41-bit RECOMP number. As a result, T_6 in the example corresponds to T_{41} in RECOMP, etc. During U_1, note that zero is set into the leading A-register bit, A_6, at T_1 (thereby making $A_6 = 0$ during T_2) and is shifted down through A. In the mean-time the multiplier bits are shifted from A_2 to R_6 and down through R while the original R contents, indicated by X's, are dumped off the right end of R. The multiplicand is recirculated by B_6 copying B_1 and shifting.

Several other functions are performed during U_1. The algebraic sign of the product is determined and appropriately stored. The signs of the multiplier and multiplicand, initially in A_{41} and B_{41} respectively, are sampled in A_{A1} and B_1 at T_{41}. If the signs are the same, a plus sign is indicated and R_{43} is turned on.

$$1^r_{43} = A_1B_1I_3U_1T_x'M_4'T_{41a}C + A_1'B_1'I_3U_1T_x'M_4'T_{41a}C \quad (M_4'T_{41a} = M_{41})$$

If the signs are opposite, a minus sign is indicated. No logic is necessary for R_{43} since it is already off due to

$$0^r43 = T_{40}C$$

At T_1 in the second word time (U_2) A_{41} is turned on (for a plus sign) by R_{43} true.

$$1^a41 = R_{43}M_1U_2T_1C$$

For a minus sign, A_{41} was turned off at T_1 in U_1 for clearing the A-register and just remains off.

$$0^a41 = M_1D_3'T_1C$$

At first observation, it would appear that A_{41} is hit on both sides at U_2T_1 time. However, this is prevented by turning D_3 on at T_{41} of U_1.

$$1^d3 = M_1T_{41}C$$

Thus, D_3 is on for the remainder of multiplication.

		K_a	S	A	R	B	R	Notes
				6 5 4 3 2 1	6 5 4 3 2 1	6 5 4 3 2 1	43	
U_1	T_1	not used during U_1		- 1 1 0 1 0	x x x x x x	- 1 0 1 0 0	X	x's represent irrelevant information.
	T_2			0 - 1 1 0 1	1 x x x x x	0 - 1 0 1 0	X	
	T_3			0 0 - 1 1 0	0 1 x x x x	0 0 - 1 0 1	X	
	T_4			0 0 0 - 1 1	1 0 1 x x x	1 0 0 - 1 0	X	0 means flip-flop zero-set by logic term.
	T_5			0 0 0 0 - 1	1 1 0 1 x x	0 1 0 0 - 1	X	
	T_6			0 0 0 0 0 -	x 1 1 0 1 x	1 0 1 0 0 -	0	
U_1'	T_1	0	x	0 0 0 0 0 0	0 x 1 1 0 1	- 1 0 1 0 0	1	* 0 for MPY, 1 for MPR
	T_2	0	0	+ x 0 0 0 0	0 0 x 1 1 1"	0 - 1 0 1 0		" indicates flip-flop remains in previous state
	T_3	0	0	+ x 0 0 0 0	1 0 0 x 1 1	0 0 - 1 0 1		
	T_4	0	1	+ x 0 0 0 0	1 1 0 0 x 1	1 0 0 - 1 0		
	T_5	0	0	+ x 1 0 0 0	x 1 1 0 0 1	0 1 0 0 - 1		
	T_6	0	1	+ x 0 1 0 0	0 x 1 1 0 1	1 0 1 0 0 -	0	Information is normally shifted one bit right unless arrows indicated otherwise.
	T_1	0	x	+ 0 1 0 1 0	+ 0 x 1 1 0	- 1 0 1 0 0		
	T_2	0	0	+ x 0 1 0 1	1 + 0 x 1 0"	0 - 1 0 1 0		
	T_3	0	1	+ x 0 0 1 0	1 1 + 0 x 0	0 0 - 1 0 1		
	T_4	0	0	+ x 1 0 0 1	x 1 1 + 0 0	1 0 0 - 1 0		
	T_5	0	1	+ x 0 1 0 0	0 x 1 1 + 0	0 1 0 0 - 1		
	T_6	0	0	+ x 1 0 1 0	1 0 x 1 1 0	1 0 1 0 0 -		
	T_1	0	x	+ 0 0 1 0 0	+ 1 0 x 1 1	- 1 0 1 0 0		
	T_2	0	0	+ x 0 0 1 0	1 + 1 0 x 1"	0 - 1 0 1 0		
	T_3	0	0	+ x 0 0 0 1	x 1 + 1 0 1	0 0 - 1 0 1		
	T_4	1	0	+ x 0 0 0 0	0 x 1 + 1 1	1 0 0 - 1 0		
	T_5	0	1	+ x 0 0 0 0	1 0 x 1 + 1	0 1 0 0 - 1		
	T_6	0	1	+ x 1 0 0 0	0 1 0 x 1 1	1 0 1 0 0 -		
	T_1	0	x	+ 0 1 1 0 0	+ 0 1 0 x 1	- 1 0 1 0 0		
	T_2	0	0	+ x 0 1 1 0	x + 0 1 0 1"	0 - 1 0 1 0		
	T_3	0	0	+ x 0 0 1 1	0 x + 0 1 1	0 0 - 1 0 1		
	T_4	1	0	+ x 0 0 0 1	1 0 x + 0 1	1 0 0 - 1 0		
	T_5	1	0	+ x 0 0 0 0	0 1 0 x + 1	0 1 0 0 - 1		
	T_6	1	0	+ x 0 0 0 0	0 0 1 0 x 1	1 0 1 0 0 -		
I_3'	T_1	0	x	+ 1 0 0 0 0	+ 0 0 1 0 x	- 1 0 1 0 0		

Figure 29. RECOMP Multiplication Process (MPY)

Rounding the final product to 39 bits is accomplished by adding binary "1" in the next bit position to the right, which happens to be the most significant bit of the first partial product. The example of figure 27 illustrates this fact wherein the most significant bit of the first partial product appears in the same vertical column as the most significant bit of the less significant half of the final double-length product. Clearly, the first partial product will be properly augmented for round-off if the multiplication process starts (at U_2T_1) with a "1" in A_{40}. Thus

$$1^a40 = M_1D_2U_1T_{41}C$$

where D_2 identifies the MPR command. For the MPY command, A_{40} remains zero as copied from A_{41}, the previous bit time.

Subsequent Computation Cycles, U_1'

Having completed the necessary preparations in U_1, we are now ready for the multiplication process which is shown in figure 30. Each of the 39 non-sign multiplier bits calls for a serial addition of the multiplicand into the previous partial product. Since each such addition requires one word time, the entire multiplication, including U_1, requires 40 word times.

Because signs are handled separately in U_1, one word time per addition suffices since addition of absolute values (Case I) is forced by

$$1^e0 = N_5C$$

At the start of U_2 the least significant (non-sync) multiplier bit is in R_1 and it must remain in R_1 during the entire word time in order to gate B_1 into the adder. The rest of R is preserved by recirculation from R_2 to R_{41} except at T_{40} and T_{41}.

R_{41}	$1^r41 = R_2M_1B_{42}'U_1'T_{40}'T_{41a}'C$
	$0^r41 = R_2'M_1B_{42}'U_1'T_{40}'T_{41a}'C$
R_{40}	$1^r40 = R_{41}M_1C \leftarrow R_{41}E_{30}'U_{41}'C$
	$0^r40 = R_{41}'M_1C \leftarrow R_{41}'E_{30}'U_{41}'C$

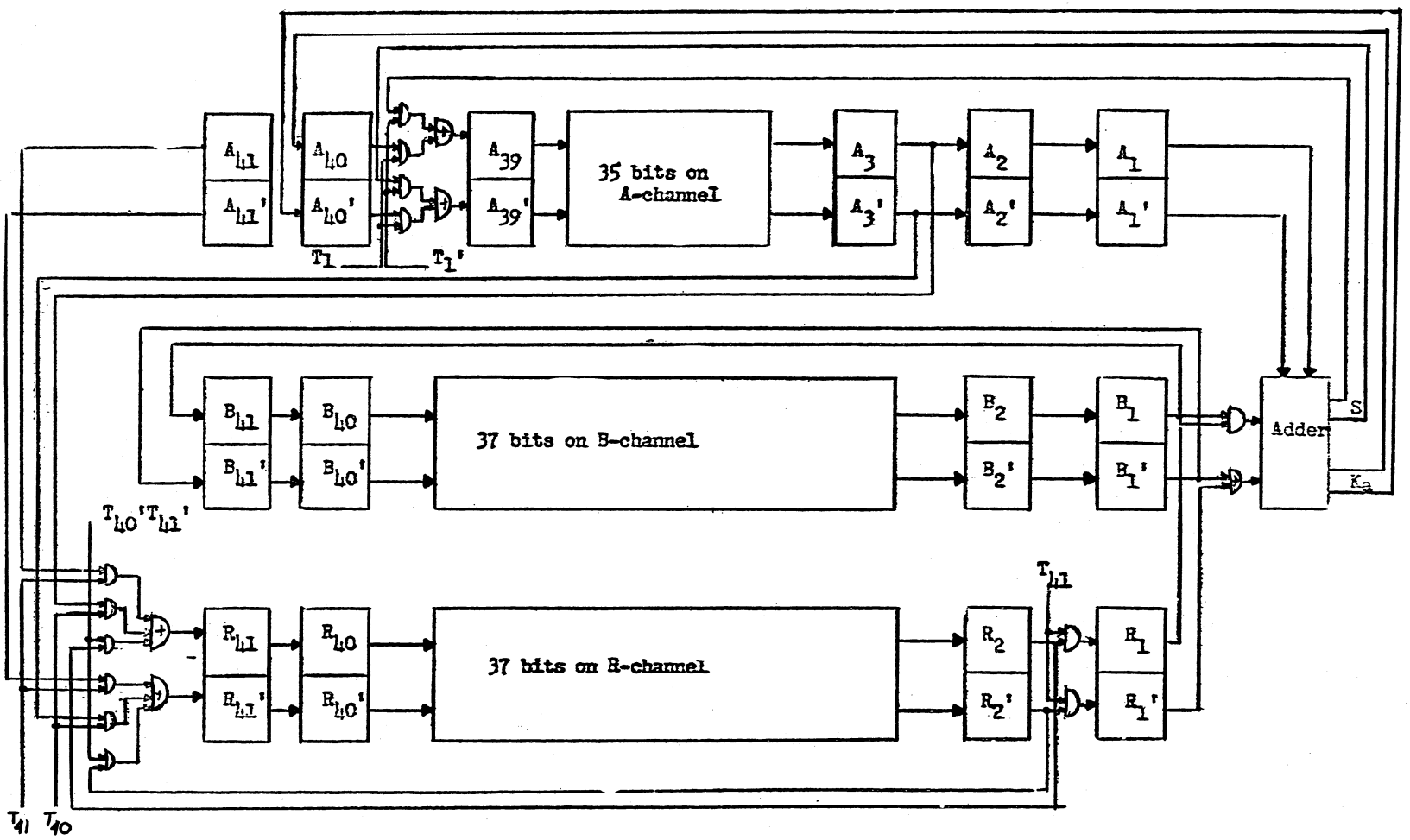


Figure 30. Multiplication During U_1'

At T_{40}, R_{41} receives the next least significant partial product bit from the A-register.

$1^r41 = A_3M_1B_{42}'T_{40}C$
$0^r41 = A_3'M_1B_{42}'T_{40}C$

The reader may convince himself that A_3 does contain the proper bit by referring to the example of figure 29. During any $U_1'T_1$, the least significant A-register bit of the previous partial product is in A_2. Proceeding in time sequence, the same bit is shifted into A_1 at T_1C, and is gated along with B_1 into S at T_2C, copied into A_4 at T_3C, shifted into A_3 at T_4C, and finally copied into R_6 at T_5C. A_4 in the example corresponds to A_{39} in RECOMP, so that the subject bit is copied into A_{39} at T_3C, shifted into A_{38} at T_4C, A_{37} at T_5C, ..., A_4 at $T_{38}C$, A_3 at $T_{39}C$, and copied into R_{41} at $T_{40}C$ as indicated in the logic.

At T_{41}, R_{41} picks up the sign of the product.

R_{41}	$1^r41 = A_{41}M_1B_{42}'T_{41a}C$
	$0^r41 = A_{41}'M_1B_{42}'T_{41a}C$

The next least significant multiplier bit, originating in R_2 during $U_1'T_1$, is copied into R_{41} and shifted down through R to return to flip-flop R_2 during T_{41}. At T_{41} it is shifted into R_1 in preparation for gating the multiplicand into the adder during the next word time.

R_1	$1^r1 = R_2M_1T_{41a}C$
	$0^r1 = R_2'M_1T_{41a}C$

Each new multiplier bit in R_1 is used to gate the entire multiplicand in B into the adder. Consequently the B-register exhibits normal recirculation during each word time in U_1' by means of the same logic as during U_1.

Logic for the sum and carry flip-flops is identically that used for addition of absolute values, where $E_0 = 1$. Flip-flop S generates a new partial product during each U_1' word time, which is the sum of the previous partial product and either the multiplicand or zero. This sum is rewritten back in A and at the same time shifted one bit to the right as required. The shifting is accomplished by having A_{39}, rather than A_{40}, copy S, thereby shortening the A-register by one bit.

A_{39}

$1^a39 = SM_1U_1'T_1'C$
$0^a39 = S'M_1U_1'T_1'C$

At T_1, S contains irrelevant information. However, the most significant bit of the new partial product is in A_{40} at T_1 and is therefore prefixed to the partial product by means of

$1^a39 = A_{40}T_1C$
$0^a39 = A_{40}'T_1C$

This left-most partial product bit was generated at T_{40} of the previous word time as the carry bit resulting from the addition of the most significant bits in A and B and was copied into A_{40} at T_{41} by means of

$1^a40 = K_aM_1U_1'C$
$0^a40 = K_a'M_1U_1'C$

Here T_{41} need not be specified since A_{40} is not used at $U_1'T_{41}'$. Each new partial product thus formed is shifted down through the A-register including flip-flop A_1 and reentered into the adder. Thus A_3 reads the disk and in turn is copied into A_2 and A_1.

$1^a2 = A_3M_1C \subset A_3I_3M_2'C$
$0^a2 = A_3'M_1C \subset A_3'I_3M_2'C$

$1^a1 = A_2M_1C \subset A_2I_3M_2'M_{10}'T_{41}'C$
$0^a1 = A_2'M_1C \subset A_2'I_3M_2'M_{10}'C + M_1T_{41}C$

At the end of 39 word times in U_1' (corresponding to the 39 multiplier bits), the 39th partial product is the final product thus completing the multiplication process. For MPY commands the double-length product is in the A- and R-registers and for MPR commands the rounded product is in the A-register. Both registers contain the sign of the product. Including U_1, a total of 40 word times elapse during I_3 so that the Q-counter finally contains a count of 40 which is used to turn on N_{12} which terminates I_3.

$1^n12 = M_1M_9'Q_6Q_4T_{40}C$
$0^i3 = N_{12}C$

Division (DSL, DIV, DSR, DVR) Commands

Four separate division commands may be executed in RECOMP. DSL causes division of a dividend A in the A-register by a divisor B in memory location m to produce a quotient Q in the A-register and a remainder R in the R-register. Thus,

$$\frac{A}{B} = Q + \frac{2^{-39}R}{B} \quad \text{where } |R| < |B| \text{ and A and R have the same}$$

sign. Those conditions uniquely determine Q. Retaining the sign of the dividend A in the remainder R facilitates programing of multiple precision division. If the binary point is assumed at the left of the most significant bit, then $|A|$ and $|B|$ are both <1, and if also $|A| < |B|$, then $|Q| < 1$.

If a double-length (78-bit) dividend is available whose more and less significant halves are in the A- and R-registers respectively, then the command DIV causes division as described above.

The commands DSR and DVR perform the same operations as DSL and DIV respectively and also include final rounding of the quotient to 39 binary places.

One may think of the binary points as being at the right of a 78-bit dividend and a 39-bit divisor. Then the equation above becomes

$$\frac{2^{78}A}{2^{39}B} = 2^{39}Q + \frac{R}{B}$$

and, the binary point is now considered at the right of the quotient.

Figure 20 shows flip-flop M_2 true for all divide commands. Further, D_2 and D_2' specify division with double and single-length dividends respectively, while D_1 and D_1' distinguish between the rounded and unrounded divide commands respectively.

Serial Binary Divider

Division in binary, as in decimal, may be performed by successive subtraction of the divisor, properly positioned, from the full and reduced dividends.

In the decimal example of figure 31, the two-digit divisor, 11, is first subtracted from the two most significant dividend bits, 14, in order to obtain the most significant (tens) digit of the quotient. Clearly, this quotient digit, 1, is the number of such subtractions possible before an overborrow occurs, as shown in the example. When an overborrow occurs, the reduced dividend, 037, is restored by adding the divisor, 11, back in again. Next, the divisor is shifted one digit right and the same process is repeated to obtain the next most significant (units) digit, 3. This being the least significant digit, the entire quotient, 13, has now been found and the final reduced dividend, 004, is the remainder. It should be noted that nine subtractions are the maximum required to obtain a given quotient digit.

The same operation is performed in binary as illustrated in figure 28. Quotient bits obtained in descending order of significance are 0 or 1 according to whether subtraction of the divisor did or did not cause an overborrow. If an overborrow occurs, restoration of the reduced dividend is again achieved by adding back the divisor. Note that only a single subtraction is required to determine a given quotient bit in binary as against nine in decimal. It appears that this method requires as many subtractions as there are quotient bits and as many additions as there are zero quotient bits.

An improvement of the previous method yields the so-called Von Neumann division that is used in RECOMP and illustrated in figure 28. In the previous method an overborrow requires addition to restore the reduced dividend, followed by subtraction of the right-shifted divisor. Clearly, this two-step operation is equivalent to a single addition of the shifted divisor, (i.e., $+D - \frac{1}{2}D = +\frac{1}{2}D$). In the Von Neumann process, this technique is used to obtain each quotient bit as a result of a single addition or subtraction. Depending upon the operation performed and the existence of an overborrow or overcarry, four distinct cases must be considered. They are summarized in the table below. Verification of the tabulated results is left as an exercise for the reader.

If		Then	
Current operation is	and caused	Quotient bit is	and next operation is
Subtraction	No overborrow	1	Subtraction
Subtraction	Overborrow	0	Addition
Addition	No overcarry	0	Addition
Addition	Overcarry	1	Subtraction

Rules for Von Neumann Division

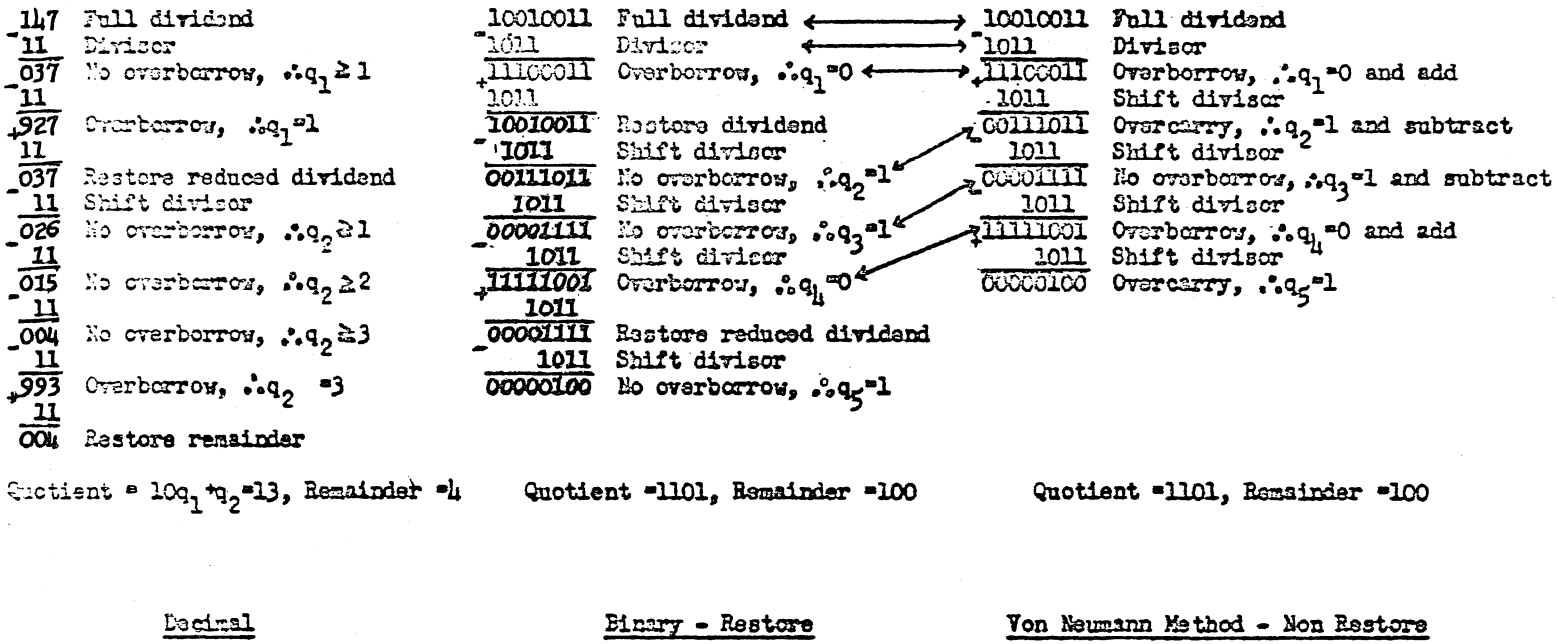


Figure 31. Division Example

These rules may be applied to the example in figure 31.

In order to perform this division in RECOMP, several things must be considered. A possible double-length dividend must be provided at the start. The A- and R-registers fulfill this requirement. Also, the computer must be capable of performing both addition and subtraction of the divisor in B upon the full or reduced dividend in A. The adder-subtractor logic previously described is used, with flip-flop E_0 true for addition and false for subtraction. Note from the example of figure 31 that the divisor is shifted right for each operation. In the computer it is more convenient to maintain the original orientation of the divisor by recirculating B, and left-shift the reduced dividend in A and R instead. As each new quotient bit is determined, it is written in the R-register. Simultaneously, another dividend bit (if double length) is transferred from R to A while the loading A-register bit, no longer needed, is dropped.

Division during U_1

At the start of I_3 the divisor is stored in the B-register after having been transferred from the memory during I_2. The dividend stands in the A-register or the A- and R-registers depending upon whether a single or double-length dividend is specified (figure 32).

During U_1 the divisor in B is subtracted from the dividend in A using the normal subtract logic. Hence, E_0 is reset for subtraction

$$0^{e}_{0} = M_2 U_1 C$$

and R_1 is set to gate B into the subtractor

$$1^{r}_{1} = M_2 U_1 C$$

The difference, A-B, is written back in A and shifted one bit left because of the delay in S.

$$1^{a}_{41} = S M_2 T_x' U_{41}' T_1' C$$

$$0^{a}_{41} = S' M_2 T_x' U_{41}' C$$

where U_{41} identifies only the final word time of division and, therefore,

$$U_1 \subset U_{41}'$$

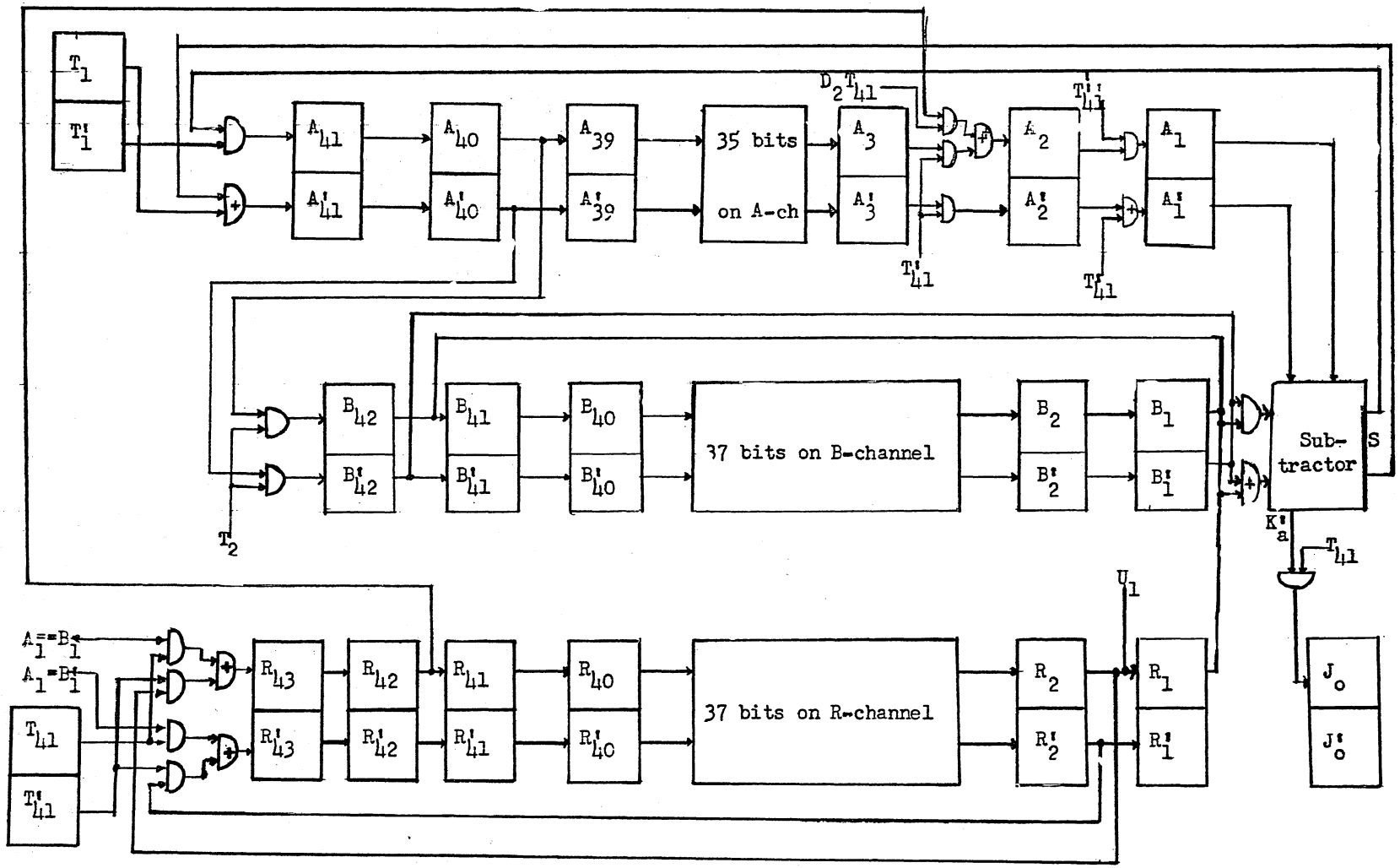


Figure 32. Division during U_1

At T_1, A_{41} is set to zero for the proper sync bit.

$$0^{a}_{41} = M_2 E_{20}' U_{41}' T_1 C$$

If the divisor is less than or equal to the dividend, no overborrow occurs at T_{40} and the overflow flip-flop J_0 is turned on at T_{41} to indicate improper division and possibly stop computation.

$$1^{j}_{0} = M_2 K_a' T_x' U_1 T_{41} C$$

Left-shifting of the R-register is accomplished using flip-flops R_{43} and R_{42} but not R_1.

$$1^{r}_{43} = R_2 T_x' T_{40}' T_{41a}' C$$

$$0^{r}_{43} = R_2' T_{41a}' C + T_{40} C$$

$$1^{r}_{42} = R_{43} M_2 C$$

$$0^{r}_{42} = R_{43}' M_2 C$$

$$1^{r}_{41} = R_{42} M_2 U_{41}' C$$

$$0^{r}_{41} = R_{42}' M_2 U_{41}' C$$

At $U_1 T_{41}$, R_{43} receives the sign of the quotient, plus if $A_1 = B_1$. If minus ($A_1 = B_1'$), no logic is needed since R_{43} has already been zero-set at T_{40} time.

$$1^{r}_{43} = A_1 B_1 I_3 U_1 T_x' M_4' T_{41a} C + A_1' B_1' I_3 U_1 M_4' T_{41a} C \quad (M_4' T_{41a} = M_{41})$$

$$0^{r}_{43} = T_{40} C$$

The sign of the dividend is in A_{41} at T_1 and, therefore, in A_{40} at T_2. It is copied at this time into B_{42} and held there.

$$1^{b}_{42} = A_{40} M_2 M_9' U_1 T_2 C$$

$$0^{b}_{42} = N_{11}' I_3' T_{41} C$$

No logic is needed for a minus sign since B_{42} is set to zero previously. The B-register exhibits normal recirculation throughout the entire division process.

$$ {}_1b_{41} = B_1M_2C \subset B_1T_x'M_0'M_8'M_{10}'I_3C $$

$$ {}_0b_{41} = B_1'M_2C \subset B_1'T_x'M_0'M_8'M_{10}'I_3C $$

Division in U_1' and U_{41}

The same logic operations are performed each word time between the first, U_1, and the last, U_{41}, and then are essentially the mechanization of the Von Neumann process. The special logic for U_1' and U_{41}' follow and are tabulated along with logic for U_1.

Division Logic

U_1, Unrounded and Rounded, Double Length, (D_2)

$$ {}_1e_0 = N_5C $$

$$ {}_0e_0 = M_2U_1C $$

$$ {}_0k_a = T_{41}C $$

$$ {}_0b_{42} = N_{11}'I_3'T_{41}C $$

$$ {}_0a_{41} = M_2E_{20}'U_{41}'T_1C $$

$$ {}_1r_{43} = R_2T_x'T_{40}'T_{41a}'C + A_1B_1I_3U_1T_x'M_4'T_{41a}C + A_1'B_1'I_3U_1M_4'T_{41a}C $$

$$ {}_0r_{43} = R_2'T_{41a}'C + T_{40}C $$

$$ {}_1a_{41} = SM_2T_x'U_{41}'T_1'C $$

$$ {}_0a_{41} = S'M_2T_x'U_{41}'C $$

$_{1}a_{2} = R_{42}M_{2}D_{2}U_{41}'T_{41}C$ If single length is desired, delete. A_2 remains zero for D_2'
$_{0}a_{2} = A_{3}'M_{2}T_{41}'C$

$_{1}r_{1} = M_{2}U_{1}C$

$_{1}j_{0} = K_{a}'M_{2}U_{1}T_{x}'T_{41}C$

$U_1'U_{41}'$, Unrounded and Rounded, Double Length, (D_2)

$_{0}k_{a} = T_{41}C$

$_{1}e_{0} = M_{2}T_{x}'U_{2}T_{1}C + A_{1}'K_{a}M_{2}U_{1}'T_{41a}C$
$_{0}e_{0} = A_{1}K_{a}M_{2}T_{41a}C$

$_{0}a_{41} = M_{2}E_{20}'U_{41}'T_{1}C$

$_{1}r_{43} = R_{2}T_{x}'T_{40}'T_{41a}'C + R_{2}U_{1}'T_{41a}'C + A_{1}K_{a}U_{1}'M_{4}'T_{41a}C$ $+ E_{0}'K_{a}'U_{1}'M_{4}'T_{41a}C$
$_{0}r_{43} = R_{2}'T_{41a}'C + M_{1}'E_{0}K_{a}'T_{41a}C + A_{1}'K_{a}U_{1}'T_{41a}C + T_{40}C$

$_{1}a_{41} = SM_{2}T_{x}'U_{41}'T_{1}'C$
$_{0}a_{41} = S'M_{2}T_{x}'U_{41}'C$

$_{1}a_{2} = R_{42}M_{2}D_{2}U_{41}'T_{41}C$ If single length is desired, delete. A_2 remains zero for D_2'
$_{0}a_{2} = A_{3}'M_{2}T_{41}'C$

Note: 41st word time rounded same as $U_1'U_{41}'$

U_{41}, Rounded, (D_1)

$$ {}_1u_{41} = M_2 D_1 Q_6 Q_4 Q_1 T_{41a} C $$

$$ {}_1e_0 = U_{41} C $$

$$ {}_0r_1 = D_1 U_{41} C + U_{41} T_{41a} C $$

$$ {}_1k_a = R_{43} D_1 U_{41} T_1 C $$

$$ {}_0k_a = E_0 A_1' B_1' M_4' T_1' C + E_0 A_1' R_1' M_4' C $$

$$ {}_1a_{41} = R_{41} U_{41} T_1 C $$

$$ {}_0a_{41} = R_{41}' U_{41} T_1 C $$

Sign

$$ {}_1a_{40} = S D_1 U_{41} C $$

$$ {}_1a_1 = R_2 D_1 U_{41} T_{41}' C $$

$$ {}_0a_1 = R_2' D_1 U_{41} C + U_{41} T_{41} C $$

$$ {}_1r_{41} = B_{42} T_x' U_{41} M_9' C $$

$$ {}_0r_{41} = B_{42}' T_x' U_{41} M_9' C $$

$$ {}_1s = A_1' R_1' K_a C $$

$$ {}_0s = A_1 R_1' K_a C $$

$$ {}_1j_0 = K_a D_1 U_{41} T_{41} C $$

Note: Check carefully S and K_a logic for entire word time.

Unrounded, (D_1')

$$ {}_1u_{41} = M_2 D_1' Q_6 Q_4 Q_1' T_{41a} C $$

$$ {}_0r_1 = U_{41} T_{41a} C + E_0' U_{41} C $$

$$ {}_1a_{41} = R_{41} U_{41} T_1 C + R_{42} D_1' U_{41} C $$

$$ {}_0a_{41} = R_{41}' U_{41} T_1 C + R_{42}' D_1' U_{41} C $$

$$ {}_1r_{43} = R_2 T_x' T_{40}' T_{41a}' C + R_2 U_1' T_{41a}' C $$

$$ {}_1r_{40} = S U_{41} T_x' M_9' C $$

$$ {}_0r_{40} = S' U_{41} T_x' M_9' C $$

$$ {}_1a_1 = A_3 D_1' U_{41} C $$

$$ {}_0a_1 = U_{41} T_{41} C + A_3' D_1' U_{41} C $$

$$ {}_1e_0 = U_{41} C $$

$$ {}_1a_2 = A_3 U_{41} C $$

$$ {}_0a_2 = A_3' U_{41} C $$

$$ {}_1r_{41} = B_{42} T_x' U_{41} M_9' C $$

$$ {}_0r_{41} = B_{42}' T_x' U_{41} M_9' C $$

Square Root

The basis for finding arithmetic square root is illustrated in the following example:

Example 1: Find $\sqrt{X}$

1. Estimate R_1 and obtain a remainder $A_1 = X - R_1^2$.
2. Estimate an increment h_1 such that $h_1 (2R_1 + h_1) \leq A_1$.
3. $A_2 = A_1 - h_1 (2R_1 + h_1)$

   $= X - R_1^2 - 2R_1 h_1 - h_1^2$

   $= X - (R_1 + h_1)^2 = X - R_2^2$
4. In general, after i steps $A_{i+1} = A_i - h_i (2R_i + h_i)$.

Decimal Example

Find $\sqrt{.063504}$

$.063504 \;/\; .252$

```
      04
.45 | 0235
     0225
  .502 | 1004
       | 1004
```

$R_1 = .2$ $\quad A_1 = .023504$

$h_1 = .05$ $\quad A_2 = .001004$

$R_2 = .25$ $\quad A_3 = .000000$

$h_2 = .002$

$R_3 = .252$

The same procedure still holds in binary arithmetic with certain simplifications: (a) considering only proper fractions, $h_i = 2^{-i-1}$; (b) establishing this as a fixed routine, 1 will always be tried as a "next" bit in R_i. If it does not work, it is necessary to restore the previous remainder and put a 0 as the next digit of R_i before continuing. This provides a "restoring" square root procedure. The general step is provided by:

$$A_{i+1} = A_i - h_i (2R_i + h_i)$$

$$= A_i - 2^{-i-1} (2R_i + 2^{i-1})$$

$$= A_i - 2^{-i} (R_i + 2^{-i-2})$$

If A_{i+1} is negative, $R_{i+1} = R_i$, add back the subtracted quantity to restore A_i and then proceed to step i+1.

$$A_{i+1} = A_i - 2^{-(i+1)}(R_i + 2^{-(i+1)-2})$$

$$= A_i - 2^{-i-1} R_i - 2^{-2i-4}$$

$$= A_i - 2(2^{-i-1} R_i) + 2^{-i-1} R_i - 4(2^{-2i-4}) + 3(2^{-2i-4})$$

$$= A_i - 2^{-i} R_i - 4(2^{-2i-4}) + 2^{-i-1} R_i + 3(2^{-2i-4})$$

$$= \left[A_i - 2^{-i}(R_i + 2^{-i-2})\right] + 2^{-i-1}(R_i + 3(2^{-i-3})$$

The quantity in brackets is the partial remainder before restoring; a nonrestoring procedure is given by the last term.

If $A_i \geq 0$, $A_{i+1} = A_i - 2^{-i}(R_i + 2^{-i-2})$, and

if $A_i < 0$, $A_{i+1} = A_i + 2^{-i}(R_i + (3)\,2^{-i-2})$

Binary Example (negatives expressed by true binary compliments)

Find $\sqrt{.011001}$

```
.011001    | .101
 01
------
 0010
  101
------
110101
 1011
------
000000
```

$A_1 > 0$, $R_1 = .1$

$A_2 < 0$, $R_2 = .10$

$A_3 \leq 0$, $R_3 = .101$

Example 2: Find $\sqrt{\frac{81}{256}} = \sqrt{.01010001}$

Sign of remainder	.01 01 00 01	partial root
	01	
(+)	00 01 00 01	.1
	01 01	
(-)	11 00 00 01	.10
	00 10 11	
(-)	11 10 11 01	.100
	00 01 00 11	
(+)	00 00 00 00	.1001 = $\frac{9}{16}$ = answer

Let R_i = Partial root at end of i^{th} step
Let A_i = Remainder at end of i^{th} step

Rules of Operation:

If A_i is +, 1 has just been inserted as right-most digit of R_i.
If A_i is -, 0 has just been inserted as right-most digit of R_i.
If A_i is +, next step is to subtract $2^{-i} \left\{ \left[(R_i) + 1 \right] \left[2^{-(i+2)} \right] \right\}$
If A_i is -, next step is to add $2^{-i} \left\{ \left[(R_i) + 3 \right] \left[2^{-(i+2)} \right] \right\}$

Mechanization of Square Root

The following description of the mechanization can be easily followed if the operation is traced through the example on the following page. The primed headings indicate extensions of normal registers. The primed entries in the R-register indicate the partial root.

Three registers are used, A, B, and R. Each is extended by two flip-flops over its normal length.

During the first word time, the operand, initially in A, is shifted into B. R is cleared. In the table on the following page, arrows indicate recirculation paths, and underlines indicate the insertion of bits from logic, not recirculation.

At the last bit time of each word and the first bit time of the next word (bit times 5 and 1 of the example), two bits are shifted from B into the right-hand bit position of A (from B6 to A1). These two bits are in each case the two "next" bits not yet operated upon. At the end of each word time, the right-hand bit of R (R7) is set to 1 and the new bit of the partial root is inserted in the left-hand bit position (R6). At the next bit time, the right-hand bit of R is set to 0 if A_i is positive (i.e., if A1 is 0) and to 1 if A_i is negative (i.e., A1 is 1). During every word time after the first, B recirculates. Its 2 additional bits cause a left-shift of two bits each word time.

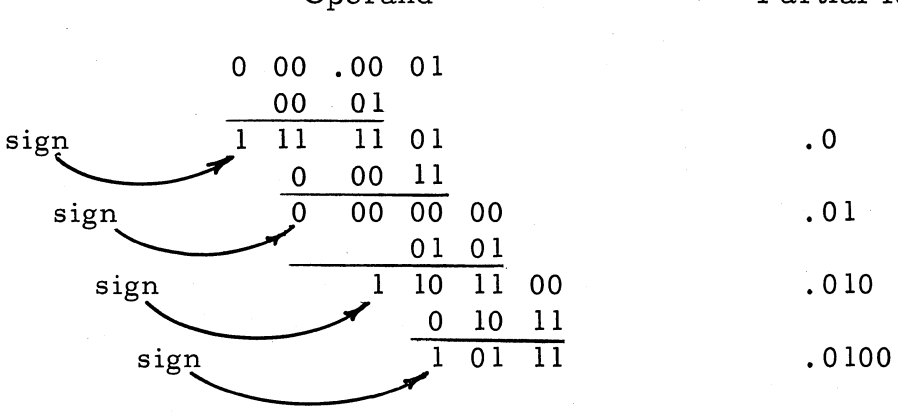
The contents of the right-hand bit of the R-register is subtracted from (or added to) the right-hand bit of A, according to the sign of A_i and the rules of the preceding section. The difference (or sum) is entered into the left-hand bit of the A-register.

R has a complicated recirculation pattern. The bit in the next-to-right-hand bit position shifts into the left-hand bit position (R1 to R6), then shifts one to the right (R5), and then goes two places: the next position to the right (R4) and also the right-hand bit position (R7). In this way the partial root is built up in R.

The square root process consists of three parts: a shift from A into B (one word time), operation (for an n-bit word this takes n word times), and a shift of the root from R into A (one word time).

During this last word time the root is shifted into A in such a manner that it is in the "correct" position for further use elsewhere (in the example R6 shifts into A5).

Example 3: Find $\sqrt{\frac{1}{16}} = \frac{1}{4}$ ($\sqrt{.0001} = .0100$).



Definitions of Floating-Point Operations

Given a pair of floating-point numbers having the form $x_1 = m_1 \cdot 2^{e_1}$ and $x_2 = m_2 \cdot 2^{e_2}$, form their sum, difference, product, and quotient as follows.

$$x_1 \pm x_2 = (m_1 \pm m_2 \cdot 2^{e_2 - e_1})\ 2^{e_1}, \text{ assuming } e_1 \geq e_2$$

$$x_1 x_2 = m_1 m_2 \cdot 2^{e_1 + e_2}$$

$$\frac{x_1}{x_2} = \frac{m_1}{m_2} \cdot 2^{e_1 - e_2}$$

Floating-Point Addition and Subtraction

Before m_1 and m_2 can be added or subtracted, they must be scaled to the same exponent. This is accomplished by denormalizing the number having the smaller exponent until the two exponents are equal (i.e., shifting m_i right and augmenting e_i by the number of such shifts). In the example above, e_2 was assumed $\leq e_1$ and, therefore, x_2 was denormalized to

$$x_2 = m_3 \cdot 2^{e_1}$$

where $m_3 = m_2 \cdot 2^{e_2 - e_1}$

Procedure

The following must be accomplished in the order mentioned.

1. The four arguments m_1, e_1, m_2, e_2 must be properly stored in arithmetic registers.
2. Exponents e_1 and e_2 must be compared to determine which is smaller (assume $e_2 \leq e_1$ here).
3. If x_2 has the smaller exponent, it must be denormalized by right-shifting m_2 and augmenting e_2 according to the following:

   a. If $e_2 \geq 0$, perform the addition $/e_2/ + 1$ and assign a plus sign to the result.
   b. If $e_2 < 0$, perform the subtraction $/e_2/ - 1$ and assign a minus sign to the result.

4. Comparison of e_1 and e_2 occurs at each step during denormalization until $e_1 = e_2$ at which time denormalization is concluded.
5. Addition or subtraction of m_1 and m_2 is performed according to the operation code of the floating-point command.
6. Normalization of resulting sum or difference occurs to insure

$$1/2 \leq /m_1 \pm m_2/ < 1$$

Additional details that must be considered are:

1. If m_1 or m_2 is initially zero or becomes zero as a result of denormalization, the denormalization process must be concluded and addition or subtraction begun.

2. If the result of the addition or subtraction (paragraph 5, preceding page) is zero, the final normalization process must be omitted.

3. To avoid possible overflow from addition, both x_1 and x_2 are subjected to an initial one-bit denormalization so that

$$/m_1' \pm m_2'/ < 1/2$$

where $m_1' = 1/2m_1$ and $m_2' = 1/2m_2$.

Operation Identification

The floating-point add (FAD) and floating-point subtract (FSB) commands have operations codes of 04 and 06, respectively. Both commands are identified with a new flip-flop M_8 which is one-set simultaneously with I_3. Floating-point multiply (FMP) and floating-point divide (FDV) commands have operation codes of 07 and 05, respectively, and are identified with another flip-flop designated M_9.

Operand Working Storage

Before any floating-point command is to be executed, the first operand x_1 must appear with its mantissa m_1 in the A register and its exponent e_1 in the R register. The second operand, x_2, exists initially in two consecutive memory locations, the first location being given by the address of the floating-point command. Thus, during I_2, the first word of x_2, which is m_2, is gated from memory into the B-register. The second word, e_2, is gated during the next word time I_3U_1 into the X-register. *

Sequence Control

Events occurring during the execution of the FAD and FSB commands are controlled by several flip-flops as indicated in figure 33.

*A new one-word recirculating register

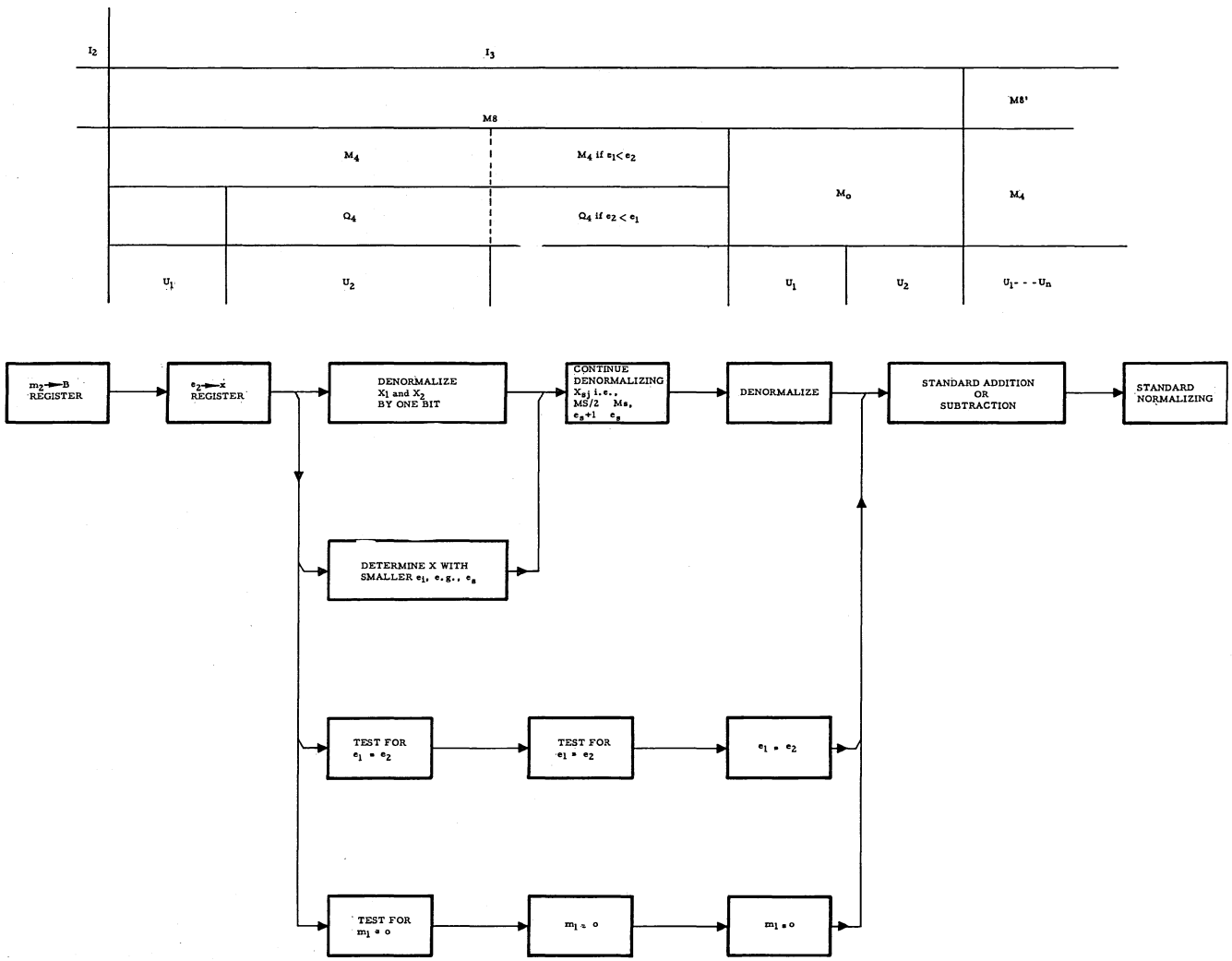


Figure 33. Floating Point Add-Subtract Sequence

Logic for FAD and FSB Commands

A detailed description of the changes in RECOMP logic necessitated by the FAD and FSB commands follows. Octal operation codes 04, 06, 44, and 46 set flip-flop M_8 at the end of I_2.

$$1^{m}_{8} = D_5'D_4'D_3D_1'I_2N_{11}J_0'C$$

M_4 is also set by the FAD and FSB commands codes by

$$1^{m}_{4} = D_5'D_4'D_3I_2N_{11}J_0'C$$

while M_0 must be inhibited by D_3' in

$$1^{m}_{0} = D_6'D_5'D_4'D_3'I_2N_{11}J_0'C$$

With M_4D_3 now true, terms required for the FNM command (code 45) must be inhibited by $M_8'M_9'$.

$$1^{h}_{0} = M_4M_8'M_9'D_3U_1T_1C$$

$$1^{n}_{12} = A_1M_4M_8'M_9'D_3U_1T_{40}C + A_{42}M_4M_8'M_9'D_3U_1'T_{40}C$$

Also, since M_4D_2 is for FSB as well as STA commands M_6 and D_2' must be inhibited to prevent STO code from being activated at end of U_1. D_3' (or M_8') may be used.

$$1^{m}_{6} = M_4D_3'D_2D_1'T_{41a}C$$

$$0^{d}_{2} = M_4D_3'T_{41a}C$$

$$0^{m}_{4} = M_4D_3'D_2T_{41}C$$

To prevent STA logic from affecting FSB command, D_3' (or M_8') must be inserted in the following

$$1^{n}_{7} = A_{42}'M_4D_3'P_6'P_5P_3'P_1'C$$

$1^{a}_{41} = A_1 M_4 D_3' D_2 D_1' C$
$0^{a}_{41} = A_1' M_4 D_3' D_2 D_1' C$

During the first word time of M_8, the A, B, and R-registers must recirculate m_1, m_2, and e_1, respectively.

For A-register recirculation, add

$1^{a}_{41} = A_1 M_4 D_3 U_1 C$
$0^{a}_{41} = A_1' M_4 D_3 U_1 C$

For B-register recirculation, add gates

$1^{b}_{41} = B_1 M_8 M_0' Q_4' C$
$0^{b}_{41} = B_1' M_8 M_0' Q_4' C$

where Q_4' is certainly true during the first word time.

For R-register recirculation during the first word time:

$1^{r}_{41} = R_1 K_a' M_4 D_3 T_{41a}' C + E_0 M_4 D_3 T_{41a} C$
$0^{r}_{41} = R_1' K_a' M_4 D_3 T_{41a}' C + E_0' M_4 D_3 T_{41a} C$

where E_0 contains the sign of the R contents.

During the first word time the X-register receives the exponent e_2 from memory or high-speed loop

X_{41}

$1^{x}_{41} = M_r L_{p1}' M_4 U_1 C + K_0 L_{p1} M_4 U_1 C$
$0^{x}_{41} = M_r' L_{p1}' M_4 U_1 Q_2' C + K_0' L_{p1} M_4 U_1 Q_2' C$

X_{40}

$1^{x}_{40} = X_{41}M_{10}{}'C$
$0^{x}_{40} = X_{41}{}'M_{10}{}'C$

X_{w}

$1^{x}_{w} = X_{40}$
$0^{x}_{w} = X_{40}{}'$

X_{2}

$1^{x}_{2} = X_{r}$
$0^{x}_{2} = X_{r}{}'$

X_{1}

$1^{x}_{1} = X_{2}C$
$0^{x}_{1} = X_{2}{}'C$

During the second word time of M_8, both operands are denormalized by one bit in order to avoid overflow from addition of mantissas. Denormalization of x_1 is accomplished by shifting the A contents one bit to generate $m_1/2$, and augment the R contents by one to generate e_1+1:

$$x_1 = m_1 \cdot 2^{e_1} = m_1/2 \cdot 2^{e_1+1}$$

Right-shifting of the A-register as exists for the ARS command (code 40) may be extended to the FAD and FSB commands merely if $D_2{}'$ is omitted from:

$1^{a}_{41} = A_2M_4D_1{}'U_1{}'T_1{}'T_{40}{}'T_{41a}{}'C$
$0^{a}_{41} = A_2{}'M_4D_1{}'U_1{}'T_{41a}{}'C$

and preserve the sign by replacing $D_2{}'$ with $U_1{}'$ in:

$1^{a}_{41} = A_1M_4U_1{}'T_{41a}C$
$0^{a}_{41} = A_1{}'M_4U_1{}'T_{41a}C$

and remove D_2' from:

$$ {}_0a_{41} = M_4 D_1' T_1 C + M_4 D_1' U_1' T_{40} C $$

No difficulty arises with the STA command since code 42 exists during U_1 only.

Augmentation of the R-register requires sampling to determine whether $e_1 \geq$ or < 0. If $e_1 \geq 0$, $/e_1/$ must be increased by one and if $e_1 < 0$, $/e_1/$ must be decreased by one.

Q_0 samples e_1 for zero as it shifts through the R-register. Q_0 is set at T_1 and reset if a "one" bit is sensed in R_{41}.

$$ {}_1q_0 = D_3 T_1 C $$

$$ {}_0q_0 = R_{41} D_3 T_1' T_2' C $$

If Q_0 is still true at T_{41}, all bits in R except possibly the sign and most significant bit are zero. It is unreasonable to assume the most significant bit "one" since this would mean:

$$ x_1 \geq 2^{(2^{38}-1)} $$

which is larger than the number of grains of sand that would fill the universe.

The states of R_{41} and Q_0 at the end of a word time depend upon the exponent e_1 and are, therefore, used to set E_0 for augmenting or reducing this exponent.

$$ {}_1e_0 = R_{41} M_4 D_3 T_1 C + Q_0 M_8 M_0' T_{40} C $$

$$ {}_0e_0 = R_{41}' M_4 D_3 T_1 C $$

E_0 determines whether the R-register acts as a one-input adder or subtractor. If the exponent is positive, $R_{41} = 1$ at T_1 and if the exponent is zero $Q_0 = 1$ at T_{40}. In either case, E_0 is set and one input addition occurs because of

$$ {}_0k_a = E_0 R_1' M_4 D_1' T_1' C $$

(i.e., the carry is turned off by the first zero bit in R_1). If the exponent is negative, R_{41}' turns off E_0 at T_1 which causes subtraction by means of:

$$ {}_0k_a = E_0' R_1 M_4 D_1' T_1' C $$

The use of T_{40} in the second gate in ${}_1e_0$ above is necessary to obtain a plus sign (since R_{41} copies the sign from E_0 at T_{41}) when the exponent is zero. If this were not done, E_0 would get zero-set at the next T_1 by R_{41}' which would result in subtraction instead of addition and the exponent would oscillate between -0 and +1 instead of going from -1 to +0 to +1 to +2, etc.

Gates in ${}_0k_a$ used for normalization are inhibited at this time by D_1 in:

$$ {}_0k_a = E_0 R_1 M_4 D_1 T_1' C + E_0' R_1' M_4 D_1 T_1' C $$

Recirculation of R during execution of ARS, ALS, and STA commands is assured by D_3' in:

$$ {}_1r_{41} = R_1 M_4 D_3' D_1' C + R_1 M_4 D_3' D_2' C $$

$$ {}_0r_{41} = R_1' M_4 D_3' D_1' C + R_1' M_4 D_3' D_2' C $$

In order to denormalize the floating-point operand in the B and X registers a number of flip-flops corresponding to Q_0, E_0, and K_a are needed. Since the Q counter is not needed for FAD and FSB, and the C-register is used only for gating e_2 during the first word time, twelve flip-flops are available for time sharing. In order to avoid conflict with their previous functions, M_8' is placed in each existing gate in Q_1 through Q_5 (Q_6 is not used). No difficulty occurs from the C flip-flops since their normal shift logic requires N_5 which is inhibited during I_3. Hence, add M_8' to:

$$ {}_1q_i = Q_{i+1} N_1 M_8' C $$

$$ {}_0q_i = Q_{i+1}' N_1 M_8' C $$

for i = 1, 2, 3, 4, 5.

Let Q_4 be true during the time that denormalization of the operand in B and X must occur, just as M_4 is true as long as the operand in A and R requires denormalization. During the second word time, U_2, both M_4 and Q_4 must be true along with M_8.

$$ {}_1q_4 = M_8 M_0' I_5' U_1 T_{41} C $$

During $M_8 Q_4$ the mantissa in B is shifted right.

$$ {}_1b_{41} = B_2 M_8 Q_4 T_1' T_{40}' T_{41a}' C + B_1 M_8 Q_4 T_{41a} C $$

$$ {}_0b_{41} = B_2' M_8 Q_4 T_{41a}' C + B_1' M_8 Q_4 T_{41a} C + M_8 Q_4 T_1 C + M_8 Q_4 T_{40} C $$

Let Q_1 sample the exponent in X for zero (corresponds to Q_0 for R). It is zeroed at T_1 and turned on if a one is sensed in X_{41}.

$$ {}_1q_1 = M_8 X_{41} T_1' C $$

$$ {}_0q_1 = M_8 T_1 C $$

Let Q_2 indicate whether the absolute value of the exponent in X is to be increased or decreased (corresponds to E_0 for R). If the exponent is positive or zero, one-input addition must occur in X; if it is negative and not zero, subtraction must occur.

Since Q_2 is initially zero-set at the beginning of I_3 and there is not a requirement to re-zero-set Q_2 if it becomes one-set, no additional logic is needed for zero-setting Q_2.

$1^{x}_{41} = M_8 Q_2 Q_4 T_{41} C$
$0^{x}_{41} = M_8 Q_2' Q_4 T_{41} C$

A possible difficulty arises when the exponent read from memory is -0 since in this case X_{41}' would normally copy M_r' or K_0' during $U_1 T_{41}$ while Q_2 would be set by Q_1' for exponent augmentation. During the succeeding word time then, Q_2 would be zeroed by X_{41}' at T_1 causing exponent diminution to occur. This problem is solved by using Q_2 to inhibit X_{41}' from copying the sign from memory or loop at $U_1 T_{41}$ since Q_2 true at this time indicates a zero exponent.

Thus:

$1^{x}_{41} = M_8 Q_2 Q_4 T_{41} C$
$0^{x}_{41} = M_r' L_{p1}' M_4 U_1 Q_2' C + K_0' L_{p1} M_4 U_1 Q_2' C$

Let C_4 be the carry flip-flop for the X-register one-input adder-subtractor. Since the C flip-flops are used for memory gating during U_1, time-shared gates must be inhibited during this time with U_1'. The carry is set at T_1 and turned off by X_1' when adding or X_1 when subtracting:

$1^{c}_{4} = M_8 U_1' T_1 C$
$0^{c}_{4} = M_8 Q_2 X_1' U_1' T_1' C + M_8 Q_2' X_1 U_1' T_1' C + M_8 T_{41} C$

The X-register receives the result of the one-input addition or subtraction during Q_4

$1^{x}_{41} = X_1 C_4' Q_4 T_{41}' C + X_1' M_8 C_4 Q_4 T_{41}' C$
$0^{x}_{41} = X_1' M_8 C_4' Q_4 T_{41}' C + X_1 M_8 C_4 Q_4 T_{41}' C$

In order to determine which floating-point number requires denormalization after U_2, flip-flop Q_5 samples the exponents in R and X to determine which is smaller and therefore needs additional denormalization.

$$ {}_1q_5 = M_8 D_6 {}'X_2 {}'R_2 U_2 C $$

$$ {}_0q_5 = M_8 X_2 R_2 {}'U_2 C $$

Since Q_5 must be explained at T_{41}, R_2 and X_2 must contain all bits from least significant to sign during T_1 through T_{40} which is clearly the case. Note that if R> X, $Q_5 = 1$ at T_{41} and the floating-point number in B and X requires further denormalization, whereas if R X, $Q_5 = 0$ at T_{41} and the floating-point number in A and R requires further denormalization.

Thus if $Q_5 = 0$ we turn off Q_4 so that B and X merely recirculate, and leave M_4 on:

$$ {}_0q_4 = M_8 Q_5 {}'U_2 T_{41} C $$

$$ {}_1b_{41} = B_1 M_8 Q_4 {}'M_0 {}'C $$

$$ {}_0b_{41} = B_1 {}'M_8 Q_4 {}'M_0 {}'C $$

$$ {}_1x_{41} = X_1 Q_4 {}'U_1 {}'M_{10} {}'C $$

$$ {}_0x_{41} = X_1 {}'Q_4 {}'U_1 {}'M_{10} {}'C $$

whereas if $Q_5 = 1$ we turn off M_4 so that A and R recirculate, and leave Q_4 on:

$$ {}_0m_4 = M_8 M_0 {}'Q_5 U_2 T_{41} C $$

$$ {}_1a_{41} = A_1 M_0 {}'M_1 {}'M_2 {}'M_3 {}'M_4 {}'M_{10} {}'P_0 {}'C $$

$$ {}_0a_{41} = A_1 {}'M_0 {}'M_1 {}'M_2 {}'M_3 {}'M_4 {}'M_{10} {}'P_0 {}'C $$

$$ {}_1r_{41} = R_1 M_0 {}'M_1 {}'M_2 {}'M_3 {}'M_4 {}'M_9 {}'I_3 C $$

$$ {}_0r_{41} = R_1 {}'M_0 {}'M_1 {}'M_2 {}'M_3 {}'M_4 {}'M_9 {}'I_3 C $$

After U_1, the exponents in R and X are compared for equality every word time. When agreement occurs, denormalization of both FP numbers is terminated. Flip-flop C_1 samples for agreement by being zeroed at T_{41} and turned on when a pair of corresponding bits disagree.

$_1c_1 = M_8U_1'R_{41}X_{41}'I_5'T_2'C + M_8U_1'R_{41}'X_{41}I_5'T_2'C$
$_0c_1 = M_8T_{41}C$

Note that C_1 samples R_{41} and X_{41} at T_1 for sign agreement, does not sample the sync bits at T_2, and samples the numerical bits during T_3-T_{40}. As before, if we assume the most significant bits to be zero, C_1 can not be triggered on the true side at T_{41}. One anomaly still exists, namely, when both exponents are zero, for in this case the exponent that is being increased from -1 to +0 contains a minus in the sign position until zero sensing is completed. This is overcome by turning C_1 off again at T_{40} if both exponents are zero. Thus:

$_0c_1 = M_8U_1'Q_0Q_1'T_{40}C$

Note that the two leading exponent bits are not being sampled in this situation.

When C_1' exists at T_{41}, exponents are equal and denormalization is terminated as follows.

$_0q_4 = M_8U_1'C_1'T_{41}C$
$_0m_4 = M_8M_0'U_1'C_1'T_{41}C$

Denormalization is also terminated when either mantissa is initially zero or becomes zero during denormalization. Flip-flops C_2 and C_3 sample the A and B registers respectively for zero.

$$ {}_1c_2 = A_2 M_8 U_1' C $$

$$ {}_0c_2 = M_8 T_{41} C $$

$$ {}_1c_3 = B_2 M_8 U_1' C $$

$$ {}_0c_3 = M_8 T_{41a} C $$

T_{41}' is not needed since zero sync bits were written into A_{41} and B_{41} at T_1.

Now if $C_2 = 0$ at T_{40}, A must be zero, whereas if $C_3 = 0$ at T_{40}, B must be zero. In either case C_1 is turned off which terminates denormalization.

$$ {}_0c_1 = M_8 U_1' C_2' T_{40} C + M_8 U_1' C_3' T_{40} C $$

Note that the signs appear in A_2 and B_2 at T_{40} and therefore cannot set C_2 or C_3 in time to prevent C_1 from being turned off.

Between U_2 and the end of denormalization, one of the floating-point numbers is being denormalized while the other must be preserved by recirculation. The number in A and R must recirculate during M_4' and the number in B and X during Q_4'.

$$ {}_1a_{41} = A_1 M_0' M_1' M_2' M_3' M_4' M_{10}' P_0' C $$

$$ {}_0a_{41} = A_1' M_0' M_1' M_2' M_3' M_4' M_{10}' P_0' C $$

$$ {}_1r_{41} = R_1 M_0' M_1' M_2' M_3' M_4' M_9' I_3 C $$

$$ {}_0r_{41} = R_1' M_0' M_1' M_2' M_3' M_4' M_9' I_3 C $$

$$ {}_1b_{41} = B_1 M_8 M_0' Q_4' C $$

$$ {}_0b_{41} = B_1' M_8 M_0' Q_4' C $$

$$ {}^{1}x_{41} = X_1 Q_4' U_1' M_{10}' C $$

$$ {}^{0}x_{41} = X_1' Q_4' U_1' M_{10}' C $$

Upon termination of denormalization, addition or subtraction of mantissi is initiated. M_0, U_1, and E_0 are turned on.

$$ {}^{1}m_0 = M_8 M_0' U_1' C_1' I_{15}' T_{41a} C $$

$$ {}^{1}u_1 = M_8 U_1' C_1' T_{41a} C $$

$$ {}^{1}e_0 = M_8 M_0' T_{41a} C $$

To prevent U_1 from being pulsed on the prime side we have

$$ {}^{0}u_1 = I_3 U_1 I_5' T_{41a} C $$

At this time the FAD (04) and FSB (06) codes must be made to look like ADD (01) and SUB (03). But D_6, D_5, D_4, and D_3 do not appear in any logic involving M_0, and D_2 is already set correctly for addition or subtraction. Hence it suffices only to set D_1:

$$ {}^{1}d_1 = M_8 U_1' C_1' T_{41} C $$

M_0 remains on for two word times during which the computer performs ordinary addition or subtraction as required. At the end of U_2 the sum of the mantissi is in A, so that addition or subtraction is terminated and final normalization begins. M_0 is reset

$$ {}^{0}m_0 = M_0 U_2 T_{41a} C $$

and N_{12} inhibited by M_8'

$$ {}^{1}n_{12} = M_0 M_8' U_2 T_{40} C $$

The FNM code (45) is set up.

$$ {}_1m_4 = M_8M_0U_2T_{41}C $$

$$ {}_0d_2 = M_0C $$

U_1 is set on and M_8 is terminated.

$$ {}_1u_1 = M_0U_2T_{41a}C $$

$$ {}_0m_8 = M_0U_2T_{41}C $$

From here on the computer sees an ordinary FNM command, and I_3 is terminated when normalization of the result in A is accomplished.

$$ {}_1n_{12} = A_1M_8'M_9'M_4D_3U_1T_{40}C + A_{42}M_4M_8'M_9'D_3U_1'T_{40}C $$

If the resulting mantissa in A is zero, H_0 is set and terminates the operation. Thus

$$ {}_1h_0 = M_4M_8'M_9'D_3U_1T_1C $$

$$ {}_0h_0 = A_1M_4T_1'C $$

$$ {}_1n_{12} = M_4H_0T_{40}C $$

Note that the FNM command no longer transfers control when the argument is zero but is ignored instead. Also note that the exponents in R and X will usually be equal upon termination of denormalization. Hence we may choose R in which case it is preserved by recirculation during M_0.

$$ {}_1r_{42} = R_2M_0C $$

$$ {}_0r_{42} = R_2'M_0C $$

while R_1 is used to gate B_1 into the adder.

The exception to the above (i.e., when R and X are unequal at the end of denormalization) occurs if A or B becomes zero. In this case, denormalization is immediately terminated and addition or subtraction is begun. However, the exponent that prevails in the final result must be that associated with the non-zero mantissa. Thus if B = 0, the exponent in R must remain in R, while if A = 0, the exponent in X must be transferred into R. In the latter case, C_2' will exist at T_{40} when A = 0 and this turns on Q_3 which is used for gating X into R.

$$_1q_3 = M_8 I_5 {}'U_1 {}'C_2 {}'T_{40} C$$

$$_0q_3 = U_1 T_{41} C$$

$$_1r_{41} = X_1 M_0 Q_3 C + R_{42} M_0 Q_3 {}'C$$

$$_0r_{41} = X_1 {}'M_0 Q_3 C + R_{42} {}'M_0 Q_3 {}'C$$

Q_3 being false during $M_0 U_2$, the right-hand gates preserve the exponent in R.

Another point is that the G-counter must be inhibited from counting every time $I_3 U_1$ comes true during the floating-point commands, since this occurs three times. This is solved by allowing N_4 to come true only during the last $I_3 U_1$ when M_8 is no longer on.

$$_1n_4 = I_3 M_8 {}'M_9 {}'U_1 T_2 C$$

Floating-Point Multiplication and Division

Multiplication of two floating-point numbers, x_1 and x_2, involves addition of their exponents $e_1 + e_2$ and multiplication of their mantissi $m_1 . m_2$. Division of two floating-point numbers involves subtraction of exponents $e_1 - e_2$ and division of mantissi m_1/m_2.

Procedure

The following must be accomplished in the order described.

1. The four arguments m_1, e_1, m_2, e_2 must be properly stored in arithmetic registers.

2. Exponents e_1 and e_2 must be added or subtracted as determined by the operation codes.

3. Mantissi m_1 and m_2 must be multiplied or divided as determined by the operation code.

4. Normalization of resulting floating-point product or quotient.

An additional detail to be considered is that overflow may result from division if the dividend is larger than the divisor. If it is assumed that both numbers are in normalized form initially, then a one-bit denormalization of the dividend will ensure that no division overflow occurs.

Operation Identification

Floating-point multiply (FMP) has operation code 07 and floating-point divide (FDV) has operation code 05. Both are identified with flip-flop M_9. For both operations the addition or subtraction of exponents occurs during the time M_{10} is true.

Operand Working Storage

These are identical with the registers used for the FAD and FSB commands, namely m_1 in A, e_1 in R, m_2 in B, and e_2 in X.

Sequence Control

Control of the FMP and FDV operations is as shown in figure 34 on the following page.

Logic for FMP and FDV Commands

Operation codes 05 and 07 set M_9 by means of:

$$_{1}m_{9} = D_{6}{}'D_{5}{}'D_{4}{}'D_{3}D_{12}IN_{11}J_{0}{}'C$$

M_4 is also set by:

$$_{1}m_{4} = D_{5}{}'D_{4}{}'D_{3}I_{2}N_{11}J_{0}{}'C$$

D_1 is immediately turned off by:

$$_{0}d_{1} = M_{9}M_{4}C$$

so that the XAR command (code 43) is preserved by:

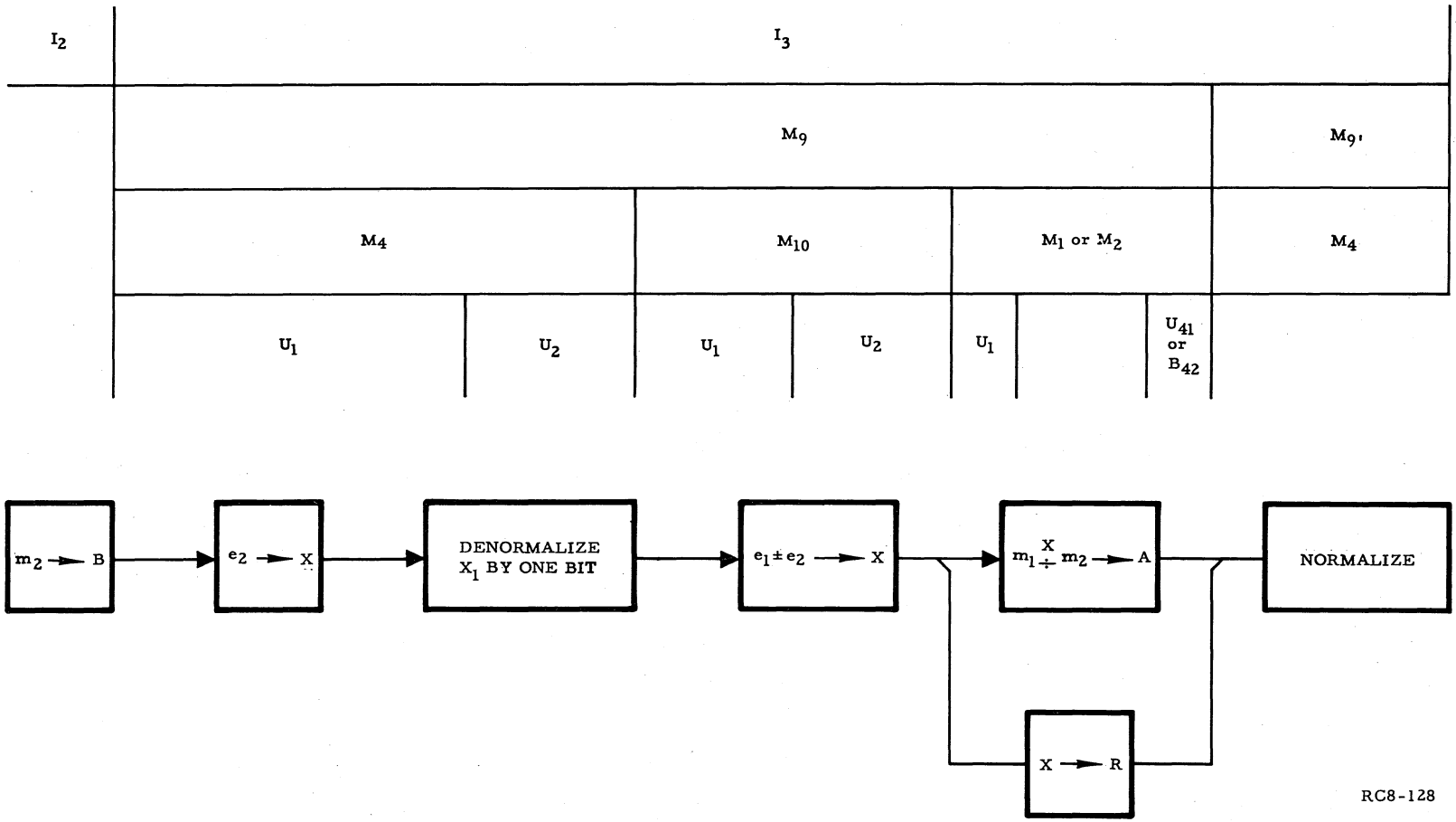


Figure 34. Floating Point Multiply-Divide Sequence

$$1^{a}_{41} = R_1 M_4 D_3 {}'D_2 D_1 C$$

$$0^{a}_{41} = R_1 {}'M_4 D_3 {}'D_2 D_1 C$$

$$1^{r}_{41} = A_1 M_4 D_3 {}'D_2 D_1 C$$

$$0^{r}_{41} = A_1 {}'M_4 D_3 {}'D_2 D_1 C$$

$$1^{n}_{12} = M_4 D_3 {}'D_2 D_1 T_{40} C$$

During U_1 the A, B, and R registers recirculate while X receives the exponent e_2 from memory:

$$1^{a}_{41} = A_1 M_4 D_3 U_1 C$$

$$0^{a}_{41} = A_1 {}'M_4 D_3 U_1 C$$

$$1^{b}_{41} = B_1 T_x {}'M_0 {}'M_8 {}'M_{10} {}'I_3 C$$

$$0^{b}_{41} = B_1 {}'T_x {}'M_0 {}'M_8 {}'M_{10} {}'I_3 C$$

$$1^{r}_{41} = R_1 K_a {}'M_4 D_3 T_{41a} {}'C + E_0 M_4 D_3 T_{41a} C$$

$$0^{r}_{41} = R_1 {}'K_a {}'M_4 D_3 T_{41a} {}'C + E_0 {}'M_4 D_3 T_{41a} C$$

$$1^{x}_{41} = M_r L_{p1} {}'M_4 U_1 C + K_0 L_{p1} M_4 U_1 C$$

$$0^{x}_{41} = M_r {}'L_{p1} {}'M_4 U_1 Q_2 {}'C + K_0 {}'L_{p1} M_4 U_1 Q_2 {}'C$$

During U_2 the floating-point number in A and R is denormalized as during FAD and FSB commands. $M_4 D_3 D_1{}'$ exists to generate ARS logic for A, and one-input add logic for R. M_4 is reset at $U_2 T_{41}$.

$$0^{m}_{4} = M_9 U_2 T_{41} C$$

B and X continue recirculation during U_2, B by the same terms as during U_1, and X by means of

$$ {}_{1}x_{41} = X_1 Q_4' U_1' M_{10}' C $$

$$ {}_{0}x_{41} = X_1' Q_4' U_1' M_{10}' C $$

Inasmuch as three arithmetic registers (A, B, and R) are required to perform the multiplication or division, it is necessary to first add or subtract exponents and store the resulting exponent in X. M_{10} identifies this operation involving the exponents. Both U_1 and E_0 must be turned on at this time.

$$ {}_{1}m_{10} = M_9 M_4 U_2 T_{41a} C $$

$$ {}_{1}u_{1} = M_9 M_1' M_2' U_2 T_{41a} C $$

$$ {}_{1}e_{0} = M_9 M_2' U_2 T_{41a} C $$

During the two word times of M_{10}, R and X must be added or subtracted. In order to utilize the existing add-subtract logic these numbers must pass through A_1 and B_1 with R_1 set for gating B_1 into the adder. This is accomplished by having A_1 copy R_2, B_1 copy X_2 and R_1 set to one. However, since the mantissi awaiting multiplication or division are in A and B, these must be preserved by recirculation from A_2 to A_{42} and from B_2 to B_{42}. The sum or difference is written into X.

$$ {}_{1}a_{1} = R_2 M_{10} C $$

$$ {}_{0}a_{1} = R_2' M_{10} C $$

$$ {}_{1}b_{1} = X_2 M_{10} C $$

$$ {}_{0}b_{1} = X_2' M_{10} C $$

$$ {}_{1}r_{1} = M_{10} T_1 C $$

$$ {}_{0}r_{1} = M_{10} T_{40} C $$

The following must be inhibited by M_{10}'.

$$1a_1 = A_2 I_3 M_2' M_{10}' T_{41}' C$$

$$0a_1 = A_2' I_3 M_2' M_{10}' C$$

Recirculation of A and B contents results from:

$$1a_{42} = A_2 M_{10} C$$

$$0a_{42} = A_2' M_{10} C$$

$$1a_{41} = A_{42} M_{10} C$$

$$0a_{41} = A_{42}' M_{10} C$$

$$1b_{42} = B_2 M_{10} T_{41}' C$$

$$0b_{42} = B_2' M_{10} C$$

$$1b_{41} = B_{42} M_{10} C$$

$$0b_{41} = B_{42}' M_{10} C$$

The logic for the addition or subtraction parallels the regular add-subtract logic. E_0, initially on, samples signs in X_{41} and R_{41} in conjunction with operation codes:

$$0e_0 = X_{41} R_{41}' M_{10} D_2 U_1 T_1 C + X_{41}' R_{41} M_{10} D_2 U_1 T_1 C$$

$$+ X_{41} R_{41} M_{10} D_2' U_1 T_1 C + X_{41}' R_{41}' M_{10} D_2' U_1 T_1 C$$

Just as the A register is cleared to zero during the first word time of M_0, so R is cleared to zero during M_{10}.

$$0^{r}_{41} = M_{10}C$$

with M_{10}' inhibiting:

$$1^{r}_{41} = M_9 M_{10}' B_{42} X_1 M_4' C$$

Note that it is not necessary to restrict the clearing to U_1 since the result here is written into X both during U_1 and U_2.

$$1^{x}_{40} = SM_{10}C$$

$$0^{x}_{40} = S'M_{10}C$$

If addition is to be performed during U_2

$$1^{e}_{0} = K_a' M_{10} T_{41a} C$$

corresponds to a similar gate during M_0. The sign is inserted into X_{41} by:

$$1^{x}_{41} = SM_{10}U_2 T_1 C$$

$$0^{x}_{41} = M_{10}U_1 C$$

M_{10} is reset at the end of U_2:

$$0^{m}_{10} = M_{10}U_2 T_{41a} C$$

From here on, X must preserve the resulting exponent by recirculation.

$$1^{x}_{41} = X_1 M_9 M_{10}' M_4' C$$

$$0^{x}_{41} = X_1' M_9 M_{10}' M_4' C$$

Depending on whether FMP (Code 07) or FDV (Code 05) is indicated, MPR (code 13) or DSR (code 21) is set up.

$$ {}_{1}m_{1} = M_{10}D_{2}U_{2}T_{41a}C $$

$$ {}_{1}m_{2} = M_{10}D_{2}'U_{2}T_{41a}C $$

D_1, having been previously turned off by M_9M_4, is turned back on.

$$ {}_{1}d_{1} = M_{10}C $$

By the end of M_{10}, four word times have elapsed in I_3 (two during M_4 and two during M_{10}) so that the Q counter now contains a count of four ($Q_3 = 1$). It is necessary to reset the Q counter to zero before starting the multiplication or division process. Thus:

$$ {}_{0}q_{3} = M_{10}T_{41}C $$

U_1 is again set by:

$$ {}_{1}u_{1} = M_{9}M_{1}'M_{2}'U_{2}T_{41a}C $$

For multiplication, E_0 is again set by:

$$ {}_{1}e_{0} = M_{9}M_{2}'U_{2}T_{41a}C $$

Both multiplication and division are performed in the same manner as for the corresponding fixed-point operations including round-off. However, during the last word time, the exponent in X is transferred into R. B_{42} identifies this last word time for both multiplication and division.

$$ {}_{1}b_{42} = M_{9}M_{1}Q_{6}Q_{3}Q_{2}Q_{1}T_{41}C + M_{9}Q_{6}Q_{4}Q_{1}T_{41}C $$

$$ {}_{0}b_{42} = M_{9}B_{42}T_{41}C $$

The gate in ${}_{0}b_{42}$ not only turns B_{42} off after one word time but also guarantees B_{42}' prior to M_9 ($M_1 + M_2$). During fixed-point division, B_{42} is used to store the sign of the remainder. However, the remainder is not used in FDV and therefore, M_9' is used to inhibit B_{42}:

$$ {}_{1}b_{42} = A_{40}M_{2}M_{9}'U_{1}T_{2}C $$

During B_{42}, R copies the exponent from X:

$$ {}_1r_{41} = X_1 M_9 M_{10}' M_4' B_{42} C $$

$$ {}_0r_{41} = X_1' M_9 M_4' B_{42} C $$

$$ {}_1r_{40} = R_{41} M_9 C $$

$$ {}_0r_{40} = R_{41}' M_9 C $$

Inhibiting diodes B_{42}' and M_9' must be included in the following gates:

$$ {}_1r_{41} = R_2 M_1 B_{42}' U_1' T_{40}' T_{41a}' C + A_3 M_1 B_{42}' T_{40} C + A_{41} M_1 B_{42}' T_{41a} C + B_{42} U_{41} M_9' T_x' C $$

$$ {}_0r_{41} = R_2' M_1 B_{42}' U_1' T_{40}' T_{41a}' C + A_3' M_1 B_{42}' T_{40} C + A_{41}' M_1 B_{42}' T_{41a} C + B_{42}' U_{41} M_9' T_x' C $$

$$ {}_1r_{40} = S U_{41} M_9' T_x' C $$

$$ {}_0r_{40} = S' U_{41} M_9' T_x' C $$

Upon completion of the multiplication or division process, the floating-point result stands in the A and R registers and it is necessary only to normalize them. Accordingly, M_1, M_2, and M_9 are terminated:

$$ {}_0m_1 = M_1 B_{42} T_{41a} C $$

$$ {}_0m_2 = U_{41} T_{41a} C $$

$$ {}_0m_9 = B_{42} Q_6 T_{41a} C $$

N_{12} is inhibited by M_9' in:

$$1^{n}_{12} = M_1 M_9' Q_6 Q_4 T_{40} C + M_9' U_{41} T_{40} C$$

in order that normalize logic may be activated. M_4 and U_1 are established by:

$$1^{m}_{4} = M_9 B_{42} Q_6 T_{41} C$$

$$1^{u}_{1} = M_9 B_{42} Q_6 T_{41a} C$$

It will be recalled that D_3 and D_1 are true at this time. If division was performed, D_2' existed (code 05) and if multiplication was performed, D_2 is turned off by:

$$0^{d}_{2} = M_1 U_2 C$$

In either case the computer sees the normalize command code and proceeds to normalize the result standing in the A and R registers. As with the FAD and FSB commands the FNM command logic terminates the FMP and FDV operations.

Both numbers and commands may be filled into any location of the main memory. Numbers are filled as mixed decimal or fractional decimal quantities. Commands are filled in the usual binary-octal "command format."

Keyboard Procedure for Entering Commands

Procedure.

Step 1 — Energize the machine as per normal operation

Step 2 — The "Ready" light indicates ON condition.

Step 3 — The "Fill Source" light indicates ready condition of console keyboard; if off, press "Fill Source" button.

Step 4 — The "Compute," "Output Error," "Overflow Error," "Verify Error," lights not illuminated; if on, press the "Error Reset" button.

Step 5 — Press the "Location" button. Light indicates the computer is ready to accept information to set the location counter.

Step 6 — Strike in the proper sequence the keyboard buttons corresponding to the desired location to be filled (2 octal digits per channel, 2 octal digits per sector, 1 bit for word half). Corresponding nixie tubes in the control panel readout indicate characters selected on keyboard. If keyboard selection in error, press the "Clear" button and start from number 6 above.

Step 7 — After the proper information has been entered into the nixie readout, press the "Enter" button. This enters the information into the computer and clears the readout. The "Location Counter" displays the location entered. The location counter is read as illustrated.

Step 8 — After checking the location counter, press the "Command" button (the command light indicates the computer will accept information in command format).

Procedure.

Step 9 Strike in proper sequence the keyboard characters required for the command desired (a sign, two octal command digits, four octal digits, one binary digit-1st half word indicator; a sign, two octal command digits, four octal digits, one binary digit-2nd half word Indicator).

Step 10 The nixie readout will display this information (check against desired entry).

Step 11 After a readout has been checked, press the "Enter" button. This enters the command pair into the selected location and clears the readout. The "Enter" button also advances the location counter in such a way that if the next consecutive memory location requires a command no further location entering is needed. Merely start with step 8 and proceed. If any other than the next location is desired, commence again from step 5.

Command Fill Example

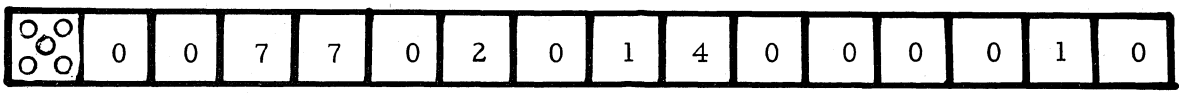
Problem: To enter the command pair +0077020 +4000010 in Location 1020

Steps 1 - 7 Set location counter to 1020.0

Step 8 "C" key

Step 9 "+" key
0077020 on keyboard
"+" key
4000010 on keyboard

Step 10 The panel display at this point would be:



Step 11 If the display indicates an error, DEPRESS "Clear" key. If display is correct, DEPRESS "Enter" key.

a. Display clears
b. Command pair is put into memory at location 1020
c. The location counter is augmented by "1.0" upon depressing "enter"

Keyboard Procedure for Entering Mixed Numbers

Procedure

Steps 1 - 7 — The same steps (1 - 7) as given in "entering commands" are used here.

Step 8 — Depress "N" button (the number light indicates number mode enabled)

Step 9 — Filling whole number portion

a. Strike sign key "+" or "-"
b. Strike up to 11 whole number digits, most significant digit first.
c. Check nixie display ("Clear" if incorrect).
d. Strike "." (decimal) key.

Step 10 — Filling fractional portion

a. Strike up to 11 digits of the fractional portion of the mixed number.
b. Check nixie display ("Clear" if incorrect).
c. Strike "Enter" key.

Example of Mixed Decimal Fill

Problem: To fill +12.75 into memory beginning at location 6010:

Steps 1 - 7 — Set location counter to 6010.0

Step 8 — "N" key

Step 9
a. "+" key
b. "1" "2" on keyboard
c. At this point the display would be:

d. Strike "." key

1. +12 is filled into location 6010 of memory as:

```
                                                              S
                                                              Y
S                                                             N
1000000000000000000000000000000000001100                     C
```

2. Location counter is automatically augmented by "1.0"

Step 10 a. Depress "7" then "5" on keyboard. At this point the display is:

b. Strike "Enter" key

1. Display is cleared
2. +.75 is filled into location 6011 of memory as:

```
                                                              S
S                                                             Y
1110000000000000000000000000000000000000                     N
                                                              C
```

3. Location counter is automatically augmented by "1.0"

Detailed analysis of Steps 1, 2, 3, 4 of II

Step 1 Energize the machine as per normal operation

With circuit breakers CB1 and CB2 closed and the POWER ON switch on the control unit depressed, a-c power is applied to the POWER ON indicator, blower, and running-time meter in the control unit, and to the full-wave bridge rectifier. The bridge supplies d-c power which energizes relay K1 and K2. Closing of K1 and K2 contacts causes a-c power to be applied to (1) the computer blowers, (2) the power transformer, (3) the memory motor, and (4) to delay relay K3. Relay K3 has a 30-second energizing time. The power supplies and all circuits deriving power from them are energized, with the exception of the main memory write switches. No current can flow to these switches because of K6 contacts 13 and 12 in open position. In addition, the computer cannot be operated because of an initial synchronization delay signal T_d generated as a result of K6 contacts 10 and 9 in open position.

Thirty seconds after power application to relay K3, it is fully energized. Closing of K3 contacts allows the memory solenoid and relay K5 to be energized. Solenoid action positions the memory disk, thereby permitting clock signals to be generated for zeroing of computer circuits. Solenoid and relay K5 current is distributed through a memory thermostat and a power supply thermostat, both of which are set to open at 120 degrees F. The memory thermostat is mounted to the memory housing and the power supply thermostat is mounted on the etched-circuit board of the -18-volt power supply.

Closing of K5 contacts 5 and 6 provides a holding circuit for the memory solenoid when the normally closed K5 contacts 1 and 3 open.

Closing of K5 contacts 12 and 13 permits power to be supplied to K4. This relay has an energizing time of 10 seconds. Zeroing of computer circuitry continues for 10 seconds, at the end of which time relay K4 is energized. Closing of K4 contacts energizes relay K6 and causes the POWER READY indicator to glow. This signifies that the computer is ready for operation, because (1) closing of relay K6 contacts 13 and 12 have applied power to the main memory write selection circuits and (2) closing of K6 contacts 10 and 9 have removed the initial synchronization delay signal.

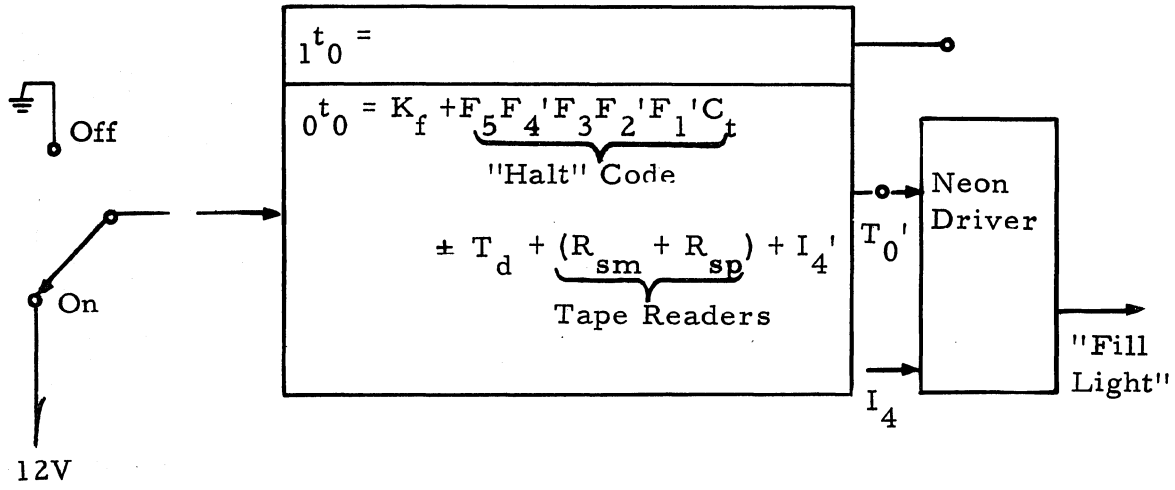
When power is removed from the computer, the memory solenoid is deactivated, thereby moving the memory disk away from the head-plate. To prevent transients from being written on the disk at this time, power is maintained to the computer secondary power supplies for approximately 0.5 second by the r-c networks across relays K1 and K2.

Step 2 Ready Light On

Step 3 The Fill Source Light On: if Off, Press the Fill Source button

Control panel fill can only be accomplished if the fill source button, on the panel, has been pressed. The signal flow is shown on the following page.

Tape Reader "ON" Flip Flop



Panel
"Fill Source"
Button

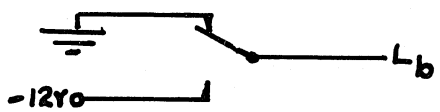
Step 4 The Compute, Output Error, Overflow Error, and Verify Error lights off; if on, press the Error Reset button. The Error Reset button zero-sets, among others, J_0 and J_{13}

$$0^{j}0 = T_d$$

$$0^{j}13 = T_d$$

Step 5 Press "Location" button. Light indicates "Locations" mode enabled.

a. Generating L_B
The "Location" button generates L_b as follows:



b. F_0 and related actions.

The generation of L_b in turn, turns on F_0 (Fill location counter flip-flop), and zero-sets 0_0 (Fill Orders flip-flop) and N_0 (fill decimal number flip-flop).

$$_1f_0 = L_b C_r' M_3' I_4 T_1 C$$

$$_0o_0 = L_b +$$

$$_0n_0 = L_b +$$

F_0, in its one-set state, is involved in the following action:

1. The Input Counter reset flip-flop, C_r, is one-set.

$$_1c_r = F_0 T_2' C$$

C_r was initially zero-set by:

$$_0c_r = F_0' 0_0' N_0' A_n' I_3' C$$

2. Since the Nixie display is involved in the location counter fill procedure, the H-counter is required. The H-counter is set to 1 (thus skipping the sign position) by:

$$_1h_1 = H_3' H_2' H_1' T_0' R_{c4}' FC$$

H_4, H_3, H_2, H_1, R_{c4}, were previously zero-set by:

$$_0h_i = U_1 C_r' A_0' C \qquad i = 1\text{-}4$$

(Where $_1u_1 = I_3' C$; $_0a_0 = I_4' + T_d$)

$$_0r_{c4} = R_{c1}' T_{40} C$$ (Where $_0r_{c1} = T_d$)

3. The input Sequence Counter (E_3, E_2, E_1) is set to "4", (1 0 0). This Counter is involved in separating the octal characters from the half-word indicator. Initially the "E" counter is "001" by:

$$ {}_{0}e_{3} = C_r {}'I_2 {}'T_1 C+ \; ; \quad {}_{0}e_{2} = C_r {}'I_2 C+ \; ; $$

$$ {}_{1}e_{1} = C_r C+ $$

It is set to "100" by:

$$ {}_{0}e_{1} = F_0 E_3 {}'T_{40} C $$

$$ {}_{1}e_{3} = F_0 E_3' E_1 R_{c1} {}'T_{40} C $$

Step 6 Strike the keyboard buttons corresponding to the desired location to be filled (2 octal digits per channel, 2 octal digits per sector and 1 bit for half-word indicator). Corresponding NIXIE tubes in the control panel readout indicate characters selected on keyboard. If keyboard selection in error, press the "clear" button and start from the beginning of Step 6.

a. TPP, Pm, P_{d3} (see timing diagram)
Striking a keyboard button causes TPP to one-set:

$$ {}_{1}TPP = TPP_k I_4 P_5' P_4' P_3' P_2' P_1 C $$

Then Pm is one-set, one word later, by:

$$ {}_{1}P_m = (TPP) J_{13}' A_n' \underbrace{P_5' P_4' P_3' P_2' P_1}_{T_2} C $$

When the keyboard button is released, TPP is zero-set: ${}_{0}TPP = (TPP_k)' (\quad)' (\quad)' \ldots$.
This, in turn, one-sets P_{d3} at T_{40}:

$$ {}_{1}P_{d3} = (TPP)' T_0' P_m T_{40} C $$

P_{d3}, in its one-set state, terminates P_m:

$$_0P_m = P_{d3}\,C$$

And finally P_{d3} is zero-set, after it has been on for 1 word time, by:

$$_0P_{d3} = P_{d3}\,T_{40}C$$

b. Filling the Input Register (F5, 4, 3, 2, 1)

1. Keyboard-to-binary code
Depressing a keyboard button generates a 5-bit code as follows:

Keyboard Character	Coding Channels K_5	K_4	K_3	K_2	K_1
∅	1	0	0	0	0
1	1	0	0	0	1
2	1	0	0	1	0
3	1	0	0	1	1
4	1	0	1	0	0
5	1	0	1	0	1
6	1	0	1	1	0
7	1	0	1	1	1
8	1	1	0	0	0
9	1	1	0	0	1
+	1	1	0	1	1
-	1	1	0	1	0
.	1	1	1	0	0
ENTER	1	1	1	1	0
CLEAR	1	1	1	0	1

2. Setting F_5, F_4, F_3, F_2, F_1

During TPP, the F's receive the signals from the Input lines K_5, K_4, K_3, K_2 and K_1.

$$_1f_i = K_i\,(TPP_5)\,I_4\,C \qquad i = 1\text{-}5$$

F's are initially in their zero-set state.

R_{c3} is one-set by P_m at this time.

c. Filling "A" Register (See figure 35)

1. Legitimate Code Flip-Flop (P_0) and Fill Orders Flip-Flop (0_1)

If the code received from the K lines is a legitimate order address character (that is, if the characters are ∅ through 7), P_0 one-sets for one word-time by:

$$ {}_1P_0 = F_5\, F_4'\, P_{d3}\, F_0\, C $$

$$ {}_0P_0 = P_0\, E_2'\, T_{41a}\, C $$

0_1 is also set to one for the octal characters:

$$ {}_1{}^0_1 = N_0'\, P_0\, E_3\, 0_1'\, F_4' + N_0'\, P_0\, E_2\, 0_1'\, F_4' $$

This, by DeMorgan's Theorem is:

$$ N_0'\, P_0\, 0_1'\, F_4'\, (E_3'\, E_2')' $$

The term $(E_3'\, E_2')'$ only allows 0_1 to be one-set at character count of 4, 5, 6 and 7. Therefore, only the channel and sector address bits are considered octal (remember "E" counter started at 100) and excludes the 1/2 word indicator.

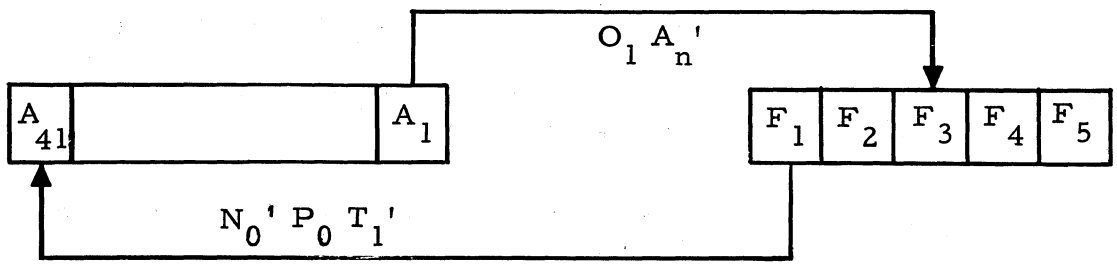
0_1 is one-set (for each octal character) for one word-time only:

$$ {}_0{}^0_1 = 0_1\, T_{41a}\, C $$

2. Shifting octal characters into "A" Register

a. Shift Diagram

For each of the first 4 characters of "A" Location (Order) counter fill, the shift flow diagram is:



0_1 is one-set during T_2 - T_{41} (for octal characters). In one-word time (T_1) therefore, the shift configuration will have left-shifted 4 bit positions. The bits originally in F_3, F_2, and F_1 will then be in A_4, A_3, and A_2 respectively. At T_2 the bits will have moved to A_3, A_2, and A_1 since A's normal right movement is not inhibited at T_1.

The left-shifting time (T_2 - T_{41}) shifts a new octal group from F_3, F_2 and F_1 to A_4, A_3, A_2 respectively while the first character moves to A_7, A_6, A_5. Again by T_2 the entire configuration moves right one-bit.

The configuration for the first four characters (octal) are summarized in the following table. The last two steps will be explained in paragraph e.

Cell Position

Time	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	
T_1											3	2	1		
T_2												3	2	1	
T_1								6	5	4	3	2	1		
T_2									6	5	4	3	2	1	
T_1					9	8	7	6	5	4	3	2	1		
T_2						9	8	7	6	5	4	3	2	1	
T_1		12	11	10	9	8	7	6	5	4	3	2	1		
T_2			12	11	10	9	8	7	6	5	4	3	2	1	
T_1	13	12	11	10	9	8	7	6	5	4	3	2	1		*
T_2		13	12	11	10	9	8	7	6	5	4	3	2	1	

*See Paragraph e.

The filled information is labeled as follows:

$$\underbrace{1\ 2\ 3,\ 4\ 5\ 6,}_{\text{CHANNEL}}\ \underbrace{7\ 8\ 9,\ 10\ 11\ 12,}_{\text{SECTOR}}\ 13\ 1/2$$

b. The logic involved is:

$$_1f_3 = A_1\ 0_1\ A_n'C +$$

$$_0f_3 = A_1'\ 0_1\ A_n'C +$$

$$_1f_2 = F_3\ 0_1 C +$$

$$_0f_2 = F_3'\ 0_1 C +$$

$$_1f_1 = F_2\ 0_1 C +$$

$$_0f_1 = F_2'\ 0_1 C +$$

$$_1a_{41} = F_1\ N_0'\ P_0\ T_1'C +$$

$$_0a_{41} = F_1'\ N_0'\ P_0\ T_1'C +$$

The rest of "A" copies right, inclusive of T_1.

c. Timing

1. H-Counter

The H-counter, as usual, counts each one-setting of N_8. N_8 is controlled by:

$$_1n_8 = (TPP)'\ R_{c3}\ T_{40}\ C +$$

$$_0n_8 = N_8\ C$$

2. E Counter

The E-counter counts the number of times P_0 is one-set:

$$ {}_1e_1 = E_1' P_0 T_{41a} C+ $$

$$ {}_0e_1 = E_1 P_0 T_{41a} C+ $$

$$ {}_1e_2 = E_2' E_1 P_0 T_{41} C+ $$

$$ {}_0e_2 = E_2 E_1 P_0 T_{41} C+ $$

$$ {}_1e_3 = E_3' E_2 E_1 P_0 T_{41} C+ $$

$$ {}_0e_3 = E_3 E_2 E_1 P_0 T_{41} C+ $$

d. Clearing F's

The F's are cleared after each character by Z_{ch} (the input Zeroing-keyboard stop flip-flop) employing the following logic:

$$ {}_0f_i = Z_{ch} I_3' C; i = 1\text{-}5 $$

Z_{ch} is set to one at the same time as the E-counter increases:

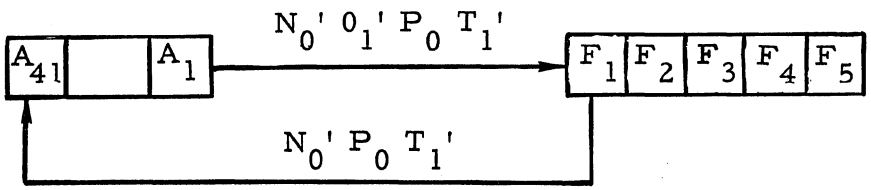
$$ {}_1Z_{ch} = N_0' P_0 T_{41} C+ $$

$$ {}_0Z_{ch} = Z_{ch} I_3' T_{41}' C+ J_{13} C $$

e. Shifting Half-Word Indicator into A-Register

The E-counter, after the 4 octal characters are filled into A, will be at a count of "000" (since it started as 100). This causes 0_1 not to be one-set at this time ($_10_1 = N_0' P_0 F_4' 01' (E_3' E_2')'$). Therefore the next character (5th) in the F-register is considered binary.

1. Shift Diagram



Since $0_1' = 1$, for the 5th character, the shifting mechanism gives a 2-bit left shift. The 1/2 word indicator will appear in position A_2 at T_1 and in position A_1 at T_2.*

2. The logic involved follows:

$$ {}_1f_1 = A_1 N_0' 0_1' P_0 T_1' C $$

$$ {}_0f_1 = A_1' N_0' 0_1' P_0 T_1' C $$

$$ {}_1a_{41} = F_1 N_0' P_0 T_1' C $$

$$ {}_0a_{41} = F_1' N_0' P_0 T_1' C $$

The rest of A copies right, inclusive of T_1.

f. Effect of "Clear" button

If at any time during the A-register fill sequence (before "enter" is depressed) the Nixie readout shows that an error has been made, then the "clear" button may be used to clear the readout and allow filling A again.

The "clear" button generates the coded character 11101 (F5, 4, 3, 2, 1). C_r is turned off by this code:

$$ {}_0c_r = A_n' F_4 F_3 P_{d3} T_2 C $$

*See Cell Position Table

C_r' resets the H-counter* to "0001" and the E-counter to "100." R_{c4} is also one-set for one word-time:

$$1^{r}_{c4} = T_0' C_r' A_0' I_4 P_{d3} F_5 C$$

$$0^{r}_{c4} = R_{c1}' T_{40} C$$

R_{c4} clears the readout.

The F-register is cleared by the usual gate:

$$0^{f}_{i} = Z_{ch} I_3 C$$

Step 7 After the proper information has been entered into the NIXIE readout, press the "enter" button. This enters the information into the computer and clears the readout. The location counter displays the next location into which information will be entered (augments one).

a. General

The "enter" button causes the following:

1. Turn C_r off at T_2.
2. Reset H-counter and E-counter.
3. Turn R_{c4} on to clear readout.
4. Turn I_1 on for one word-time.
5. Transfer A to G.
6. Light location counter control panel neons.

b. Turn C_r off

The "enter" code is 11110 ($F_{5, 4, 3, 2, 1}'$).

C_r is turned off by the same gate as in Step 6-d:

$$0^{c}_{r} = A_n' F_4 F_3 P_{d3} T_2 C$$

---

*See Step 5

c. Reset H-counter and E-counter

C_r' resets H to "0001" and E to "100."*

d. Turn R_{c4} on

R_{c4} is turned one-set (to clear the Nixie display) by:

$$1^{r}_{c4} = T_0 'C_r 'A_0 'I_4 P_{d3} F_5 C$$

e. Turn I_1 on

C_r is now turned on by:

$$1^{c}_{r} = F_0 T_2 'C$$

I_1 then is one-set for one-word time by:

$$1^{i}_{1} = F_5 F_4 F_3 F_2 \underbrace{C_r P_{d3}}_{T_{41}} N_0 ' A_n ' C^{**} \quad (P_{d3} \text{ is on by } T_{41})$$

$$0^{i}_{1} = I_1 F_0 T_{41a} C$$

f. Transfer A to G

N_4 is one-set during T_2 through T_{14} during $I_1 I_4$

$$1^{n}_{4} = F_5 F_0 I_1 T_1 C$$

$$0^{n}_{4} = P_5 P_4 P_3 C$$

During N_4 the bits passing through A_1 are copied by G_{12} and shifted into the G-counter:***

$$1^{g}_{12} = A_1 F_0 N_4 C+$$

---

*See Step 5

**C_r defines, among others, $F_0 = 1$

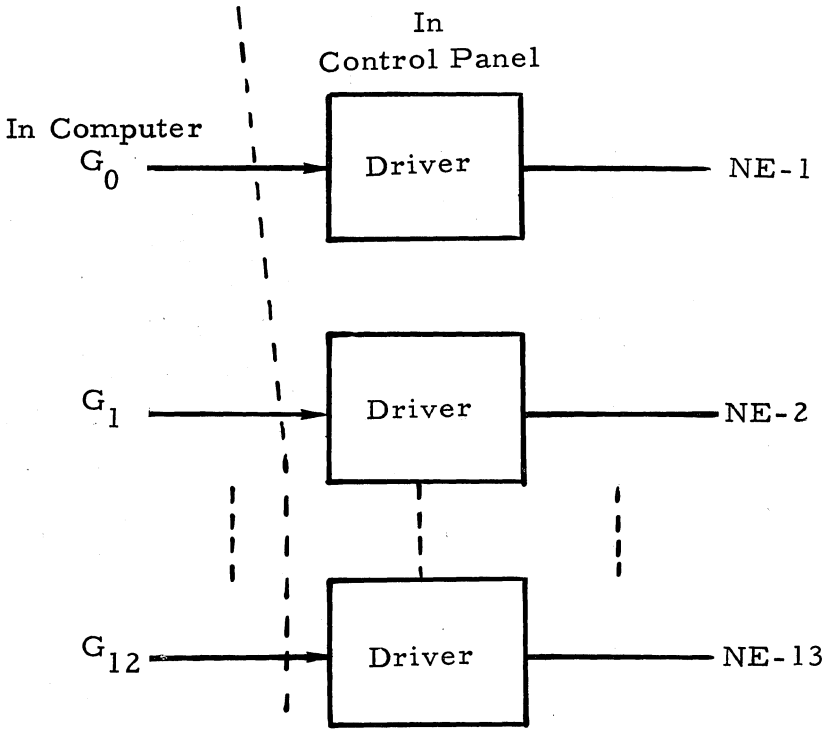
***See cell position table for initial (T_2) positions of bits in A.

$$0^{g}_{12} = A_1' F_0 N_4 C$$

$$\left.\begin{array}{l} 1^{g}_{i} = G_{(i+1)} N_4 \\ 0^{g}_{i} = G'_{(i+1)} N_4 \end{array}\right\} \quad (i = 11 - 1)$$

The location counter is now filled with the keyboard request (with the half-word indicator in G_0 and the most significant channel bit in G_{12}).

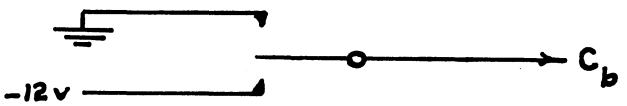
The control panel neons are driven as follows:



Step 8 After checking the location counter, press the "Command" button (the command light indicates command mode).

a. C_b Action

The command button generates the signal "C_b" as follows:



C_b itself turns 0_0 (fill orders flip-flop) on (provided C_r is off) and F_0 and N_0 off:

$$0^{c}_{r} = F_0' \, 0_0' \, N_0' \, A_n' \, I_3' C$$

$$1^{o}_{0} = C_b \, C_r' \, M_3' \, I_4 \, C$$

$$0^{f}_{0} = C_b$$

$$0^{n}_{0} = C_b$$

b. Resetting the H and E-Counters

C_r' sets the H-counter to "0" and the E-counter to "1."

$$0^{h}_{i} = U_1 \, C_r' \, A_0' \, C \qquad i = 1 - 4$$

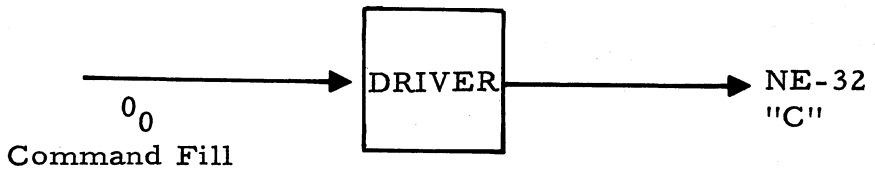
$$1^{e}_{1} = C_r' \, C$$

$$0^{e}_{2} = C_r' \, I_2' \, C$$

$$0^{e}_{3} = C_r' \, I_2' \, T_1' C$$

c. Driving the "Command" Neon

The following indicates the control panel neon indicator system:



Step 9 Strike the keyboard characters required for the command desired (sign, two octal command digits, four octal address digits, one binary digit-first half-word indicator-a sign, two octal command digits, four octal address digits, and one binary digit-2nd half-word indicator.

a. Legitimate Code Flip-Flop (P_0)

Legitimate command characters are ∅ through 7, "+" and "-". P_0 will one-set for one-word time for each of these characters only.

$$1P_0 = 0_0 \underbrace{F_5\, F_4'}_{\emptyset - 7}\, P_{d3}\, C + 0_0 \underbrace{F_5\, F_3'\, F_2}_{2, 3, +, -}\, P_{d3}\, C$$

$$0P_0 = N_0'\, P_0\, T_{41a}\, C + P_0\, E_2'\, T_{41a}\, C$$

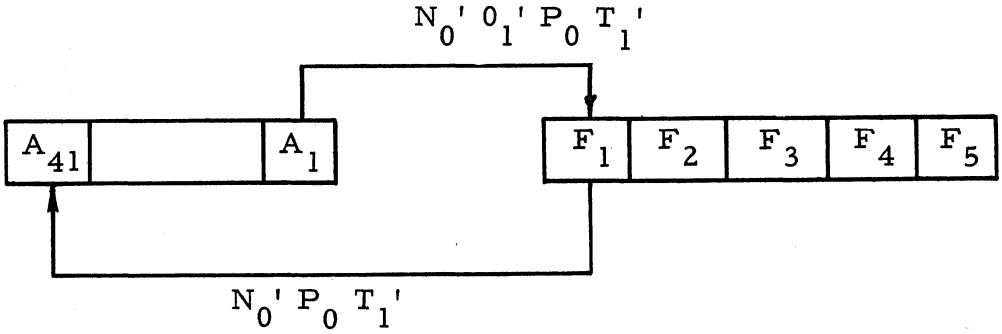
The $(N_0E_2)'$ inhibit, of the zero-set logic, does not affect 0_0 operation.

b. Putting Characters into F-Register

The process from keyboard to F-register for commands is the same as that discussed for filling the location counter.

c. Shifting the Sign from F into A

The shift mechanism for the sign is shown on the following page:



0_1 cannot become one-set (for the sign) because of the $(E_3'E_2')'$ inhibit term:*

$1^o_1 = (E_3' E_2')' F_4' 0_1' N_0' P_0 C$
$0^o_1 = 0_1 T_{41a} C$

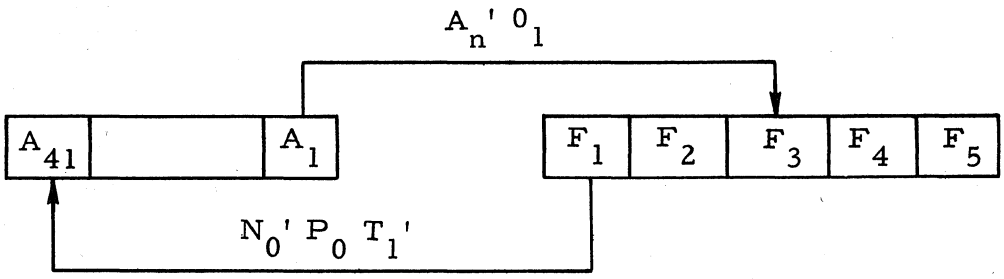
Since $T_1' P_0$ is 1 during T_2 - T_{41}, the sign bit**, originally in F_1 will be found in A_2 at T_1 after the one word time of shifting.

d. Shifting the First Octal character into "A"

0_1 is one-set (for T_2 through T_{41}) for each of the octal characters

$1^o_1 = (E_3' E_2')' F_4' 0_1' N_0' P_0 C$
$0^o_1 = 0_1 T_{41a} C$

The shifting mechanism for the octal characters follows:



*The E-counter is initially "001"

**Note from the table under Step 6, that $F_1 = 1$ for "+", $F_1 = 0$ for "-".

After the shifting time, T_2 - T_{41}, the bits originally in F_3, F_2, and F_1 will be in A_4, A_3, and A_2. The sign would, at this time, be in A_5 cell position.

e. Shifting Remaining Characters from F to A

As shown in c and d, the E-counter configuration determines the octal and binary shifts (via 0_1). The term $(E_3'E_2')'$ inhibits 0_1 at the configurations 001, and 000. These occur at the character counts of 1, 8, 9, and 16. Therefore, the signs and half-word indicators are processed by a binary shift mechanism and the remaining characters are involved in an octal shift.

Step 10 This information will be displayed in the NIXIE readout and should be checked against the desired entry.

The "Clear" button may be depressed if the readout is in error.

Step 11 After the readout has been checked, depress the "Enter" button. This enters the command pair into the selected location and clears the readout. The "Enter" button also advances the location counter.

a. General

The "Enter button causes the command pair in the A-register to be transfered into the memory location specified by the location counter. The memory search will be done during a forced I_1 state.

b. Entering I_1 I_4 State

Depression of the "Enter" button generates the code 11110 ($F_{5,4,3,2,1'}$). The I_1 flip-flop is turned one-set by:

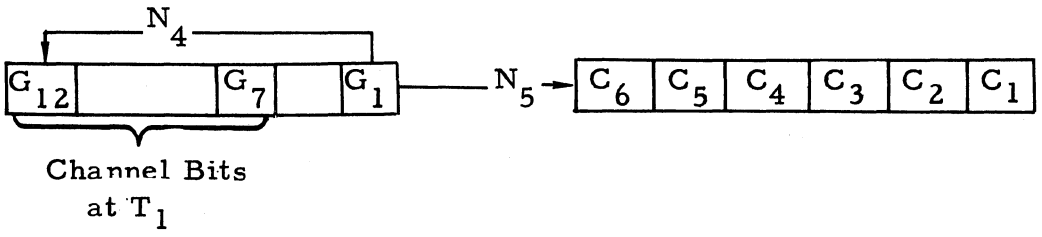
$$i_1 = F_5 F_4 F_3 F_2 C_r P_{d3} N_0' A_n' C$$

The I_1 state persists until the memory location is found and the command pair is written into it:

$$ {}^{i}_{0}1_{1} = D_{11}\, T_{40}\, C $$

c. Setting Channel Register

The channel address characters are loaded into the channel register (via C_6) from G_1 of the location counter as shown below:



N_4 is one-set for T_3 through T_{14} or T_{23} through T_{34} depending on the state of G_0.

$$ {}^{n}_{1}4 = I_1\, N_9\, F_0'\, C+ $$

$$ {}^{n}_{0}4 = P_5\, P_4\, P_3\, C+ $$

$$ {}^{n}_{1}9 = G_0\, T_1\, C+ \; G_0'\, P_6'\, P_5\, P_4'\, P_3'\, P_2'\, C \qquad T_2 \text{ or } T_{22} $$

$$ {}^{n}_{0}9 = N_9\, C \qquad (G_0)\ (G_0') $$

The channel register logic is:

$$ {}^{c}_{1}6 = G_1\, I_1\, N_5\, C+ $$

$$ {}^{c}_{0}6 = G_1'\, I_1\, N_5\, C+ $$

The rest of the channel register copies right

$$
\begin{array}{|l|}
\hline
{}_{1}c_{i} = C_{i+1}\, N_5\, C+ \\
\hline
{}_{0}c_{i} = C'_{i+1}\, N_5\, C+ \\
\hline
\end{array}
\qquad \text{where } i = 1\text{-}5
$$

d. Memory Write Gating Flip-Flop (D_{11})

D_{11} is turned one-set for the word-time of writing by:

$$
\begin{array}{|l|}
\hline
{}_{1}d_{11} = S_{11}'\, F_0'\, H_0'\, I_1\, I_4\, T_{40}\, C \\
\hline
{}_{0}d_{11} = D_{11}\, I_1\, T_{40}\, C \\
\hline
\end{array}
$$

The $S_{11}'\, T_{40}$ term indicates that the next sector is the proper sector for writing. S_{11} (memory write sector selector) remains zero-set for agreement of the least significant 5-sector address bits and disagreement in the 6th which allows for the 180-degree read-write head spacing. S_{11} thus remains zero-set if the proper sector for writing is about to come under the write head.

The logic for S_{11} monitors S_0 (memory read sector selection flip-flop) in making the write sector selection. The one-set logic for S_{11} is:

$$
{}_{1}S_{11} = S_0 \underbrace{N_1 C}_{T_3 - T_8} + S_0' \underbrace{P_6\, P_5\, P_4 C}_{T_{30}}
$$

or

$$
T_{23} - T_{28}
$$

The first gate ($S_0\, N_1$) checks for any disagreement in the first five (least significant) bits which are available (inputs) to S_0 during T_3 -T_7 (or T_{23} -T_{27}) and are seen as outputs during T_4 -T_8 (or T_{24} -T_{28}).

The second gate (S_0' P_6 P_5 P_4) checks for the required disagreement in the sixth bit. This information is available from S_0 by T_9 (or T_{29}) and is monitored at T_{30}.

The logic for S_0 itself is:

$$_1{}^{s}{}_0 = G_1 S_1' I_2' I_3' N_1 C + G_1' S_1 I_2' I_3' N_1 C +$$

$$_0{}^{s}{}_0 = T_{41} C$$

where $N_1 = 1$ at $T_3 - T_8$ (or $T_{23} - T_{28}$)

e. Writing on Memory

The information in the A-register (see Step 9) is placed in the X-register before placement in memory. This frees the A-register (while sector search is being done) for processing another input character.*

At the transfer time, I_5 is turned one-set.

$$_1{}^{i}{}_5 = F_0' T_1 I_1 P_{d3} F_4 F_3 F_1' A_n' C$$

$$_0{}^{i}{}_5 = I_5 T_{41a} C$$

During I_5 I_1, the X-register copies the A-register:

$$_1{}^{x}{}_{41} = A_1 I_1 I_5 C$$

$$_0{}^{x}{}_{41} = A_1' I_1 I_5 C$$

The main memory write flip-flops (M_{w1} and M_{w2}) copy X_2 during I_1 D_{11}.

---

*This is useful for photoreader input. The photoreader can process 400 characters a second (2.5 milliseconds per character). A disk revolution takes 17.4 milliseconds.

$$ {}_{1}m_{w1} = X_2\, I_1\, T_{41}'C + D_{11}'C $$

$$ {}_{0}m_{w1} = X_2'\, I_1\, D_{11}C + D_{11}\, T_{41}\, C $$ ---- writes 0

Synch

$$ {}_{1}m_{w2} = X_2'\, I_1\, C + T_{41}\, C $$

$$ {}_{0}m_{w2} = X_2\, I_1\, D_{11}\, T_{41}'C $$ ---- writes 1

f. Advancing the Location Counter

The "Enter" code also advances the location counter. Actually the entrance into writing process ($D_{11}\, I_1$) accomplishes the up-count.

The location counter, you will recall, is a one-input counter. The carry flip-flop for this counter (K_g) is set to "1" by:

$$ {}_{1}k_{g} = (S_0)'\, D_{11}\, I_1\, T_1\, C + T_0'\, D_{11}\, I_1\, T_1 C $$

The "1" placed in K_g will, before the next word time, increase the count held by G-counter to "m + 1.0" where "m" is the location where information was just entered.

Filling Decimal Numbers. The filling of mixed decimal numbers involves the R, B, and A-registers and also the input-output register F. For example, let us enter the mixed decimal number +12.75 starting at location "m" in memory.

1. The location counter is set to "m."
2. Pressing N sets the computer control logic for number fill.
3. Pressing "+" determines that a binary one will be entered, eventually, into the sign position of A.
4. Pressing the decimal integer "1" will:

   a. Put binary 1 (0001) into the F's
   b. Multiply the contents of R (0000) by ten (1010) and place the product (0000) in R.
   c. Add the F's (0001) to R (via A) forming the sum "0001" (one) in R

d. Multiply B (initially "1") by ten thus forming 1010 in B. This step is not required for whole-number fill.
e. Copy R into A while preserving R.

5. Pressing the decimal integer "2" will:

a. Put "2" (0010) into the F's
b. Multiply the contents of R (0001) by ten (1010) and place the product (01010) in R.
c. Add the F's (0010)(two) to R (01010), (via A) forming the sum (01100) in R.
d. Multiply B (1010) by ten $(1010)^2$ - 1100100. This is not actually required for the whole-number fill.
e. The value in R is transferred to A leaving the binary equivalent of +12 in A.

6. Pressing the "." button will:

a. Enter the contents of A (+12) into memory at location "m" (specified in 1.) as +0.....01100S.*
b. Reset B to "1." The B-register contents were not really required for the whole-number fill.
c. Augment location counter by "1."
d. Clear R.

7. Pressing the fractional decimal numbers "75" will, in like manner to steps 4 and 5, put the binary equivalent of the integer 75 into the A-register and will put the binary equivalent of 100_{10} in the B-register.

8. Pressing the enter button will cause the A value to be divided by the B value or:

$$\frac{75_{10} \longrightarrow 1001011}{100_{10} \longrightarrow 1100100}$$

```
                .11
1100100 /1001011.00
         110010 0
        0011001 00
          11001 00
```

That is, 0.11 would be found in the A-register after the conversion and also in the memory location m + 1.0. 0.11, of course, is the

*S is synch bit

binary fraction that is equivalent to a decimal 0.75. After steps 1 through 8 have been accomplished, the mixed number +12.75 would appear in memory as:

Assumed Binary Point

Location m +0000 ....01100 S

Location m + 1 +1100 ..... 0 S

Number Button N. Pressing the N-button generates a -12 signal N_b. N_b, itself, turns the fill decimal number flip-flop (N_0) one-set:

$$1^{n}{}_{0} = N_b C_r' M_3' I_4 C$$

Also, F_0 and 0_0 are turned zero-set:

$$0^{f}{}_{0} = N_b C$$

$$0^{0}{}_{0} = N_b C$$

In addition, the H-counter and E-counter are reset to "0000" and "001," respectively.

Sign Digit. Depressing the sign key will:

1. Generate TPP_k which one-sets TPP (see Step 6)
2. TPP sets P_m and P_{d3} (see Step 6)
3. P_{d3} sets P_0 for 1 word-time

Legitimate characters, for number fill, are Ø - 9, +, -, and "."

$$1^{P}{}_{0} = \underbrace{N_0 F_5 F_4' P_{d3} T_{41a} C + N_0 F_5 F_3' P_{d3} C + N_0 F_5 F_1' F_2' P_{d3} C}_{N_0 F_5 P_{d3} \left[F_4' + F_3' + F_2' F_1' \right]}$$

F_4': Ø-7

F_3': Ø-3, 8, 9, +, -

$F_2' F_1'$: Ø, 4, 8, .

$$0^{P}{}_{0} = P_0 E_2' T_{41a} C$$

4. Store sign in A_{42}

$$1^{a}_{42} = \underbrace{F_3' F_1}_{\text{covers "+"}} N_0 P_0 \underbrace{E_2' E_1}_{\text{First character}}, C \quad \text{(if sign = "+")}$$

$$0^{a}_{42} = \underbrace{F_3' F_1'}_{\text{covers "-"}} N_0 P_0 \underbrace{E_2' E_1}_{\text{First character}}, C \quad \text{(if sign = "-")}$$

5. Clear R-register

$$0^{r}_{41} = N_0 P_0 E_3' E_2' C$$

$$0^{r}_{40} = R_{41}' E_{30}' U_{41}' C$$

$$0^{r}_{1} = R_2' N_0 P_0 E_3' C$$

6. Set A_1 to zero (synch)

$$0^{a}_{1} = N_0 P_0 T_{41} E_2' C$$

7. Set B-register to

S		S
I	0....0001	Y
G		N
N		C
		H

$$1^{b}_{41} = N_0 P_0 E_3' E_2' T_2 C$$

$$0^{b}_{41} = N_0 P_0 E_3' E_2' \underbrace{P_1'}_{\text{covers } T_1,\ T_3\ldots} C$$

8. H and E-counters change to "0001" and "010" respectively

$$ {}_{0}h_{4} = U_{1} C_{r}' A_{0}' C $$

$$ {}_{0}h_{3} = U_{1} C_{r}' A_{0}' C $$

$$ {}_{0}h_{2} = U_{1} C_{r}' A_{0}' C $$

$$ {}_{1}h_{1} = H_{1}' R_{c4}' N_{8} C $$

where N_8 is set by: $ {}_{1}n_{8} = (TPP)' R_{c3} T_{40} C $

$$ {}_{0}e_{3} = E_{3} E_{2} E_{1} P_{0} T_{41} C $$

$$ {}_{1}e_{2} = E_{2}' E_{1} P_{0} T_{41} C $$

$$ {}_{0}e_{1} = E_{1} P_{0} T_{41a} C $$

9. F-Register is cleared by Z_{ch}

$$ {}_{1}z_{ch} = P_{0} E_{2}' T_{41} C $$

$$ {}_{0}f_{i} = Z_{ch} I_{3}' C \qquad i = 1\text{-}5 $$

First Decimal Character. Depressing the first decimal character (Ø - 9) starts the following actions:

1. TPP_k sets TPP
2. TPP sets P_m and P_{d3}
3. P_{d3} sets P_0 for 3 word-times

$$ {}_{1}p_{0} = N_{0} F_{5} F_{4}' P_{d3} T_{41a} C + F_{5} F_{3}' P_{d3} C + N_{0} F_{5} F_{2}' F_{1}' P_{d3} C $$

$$ {}_{0}p_{0} = P_{0} E_{2}' T_{41a} C $$

4. E-counter advanced to "001" by P_0 T_{41}

5. E_{20} turned on for 1 word-time

$$_1e_{20} = \underbrace{N_0\ F_5\ F_4'\ E_2\ P_{d3}\ C + N_0\ F_5\ F_3'\ E_2\ P_{d3}\ C}_{\text{same gates as } P_0}$$

$$_0e_{20} = E_{20}\ T_{41a}\ I_4\ C$$

6. Multiply R by Ten during E_{20}

a. General Approach

Assume that the binary number X is added to a left shifted (by two bit times) version of itself. We then have:

$X + 4X = 5X$

If the resulting sum (5X) is further left-shifted one bit, we obtain:

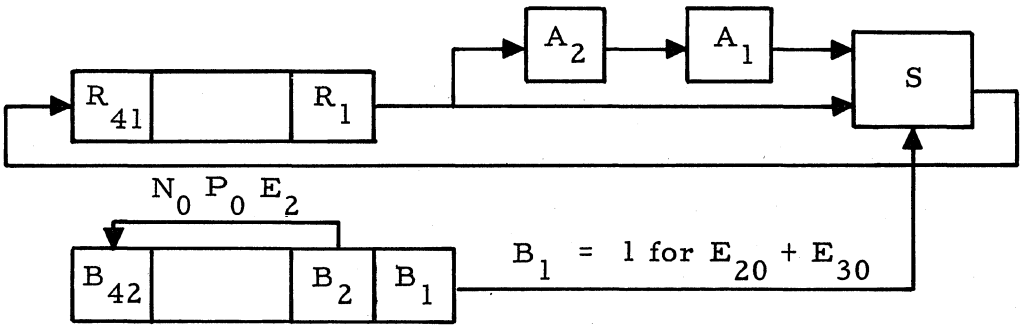
$5X + \text{delay} = 10X$

It is this approach that will be used to generate R times "ten."

b. Details

The contents of R are delayed two bit-times (before entering the sum flip-flop, S) by A_2 and A_1. This information is added to the undelayed R contents and the sum is written back into the R-register. The last one-bit delay is provided by S.

The following diagram and logic summarizes this action:



$$ {}_1b_1 = E_{20}\, C $$

$$ {}_0b_1 = E_{30}\, I_4\, T_{41a}\, C^{*} $$

$$ {}_1a_2 = R_1\, E_{20}\, I_4\, C $$

$$ {}_0a_2 = R_1'\, E_{20}\, I_4\, C $$

$$ {}_1a_1 = A_2\, N_0\, P_0\, E_1'\, T_1'\, C $$

$$ {}_0a_1 = A_2'\, I_3'\, E_{30}'\, C $$

$$ {}_1r_{41} = S\, E_{20}\, I_4\, C $$

$$ {}_0r_{41} = S'\, E_{20}\, I_4\, C $$

$$ {}_1r_1 = R_2\, N_0\, P_0\, E_2\, C $$

$$ {}_0r_1 = R_2'\, N_0\, P_0\, E_3'\, C $$

$$ {}_1b_{42} = B_2\, N_0\, P_0\, E_2\, T_{41}'\, C $$

$$ {}_0b_{42} = B_2'\, N_0\, P_0\, E_2\, C $$

$$ {}_1s = A_1\, B_1\, R_1\, K_a\, C + A_1\, R_1'\, K_a'\, C + A_1'\, B_1\, R_1\, K_a'\, C + A_1'\, R_1'\, K_a\, C $$

$$ {}_0s = A_1'\, R_1'\, K_a'\, C + A_1'\, B_1\, R_1\, K_a\, C + A_1\, R_1'\, K_a\, C + A_1\, B_1\, R_1\, K_a'\, C $$

7. Adding F to R during E_{30}

   E_{30} is one-set for 1 word-time following E_{20}

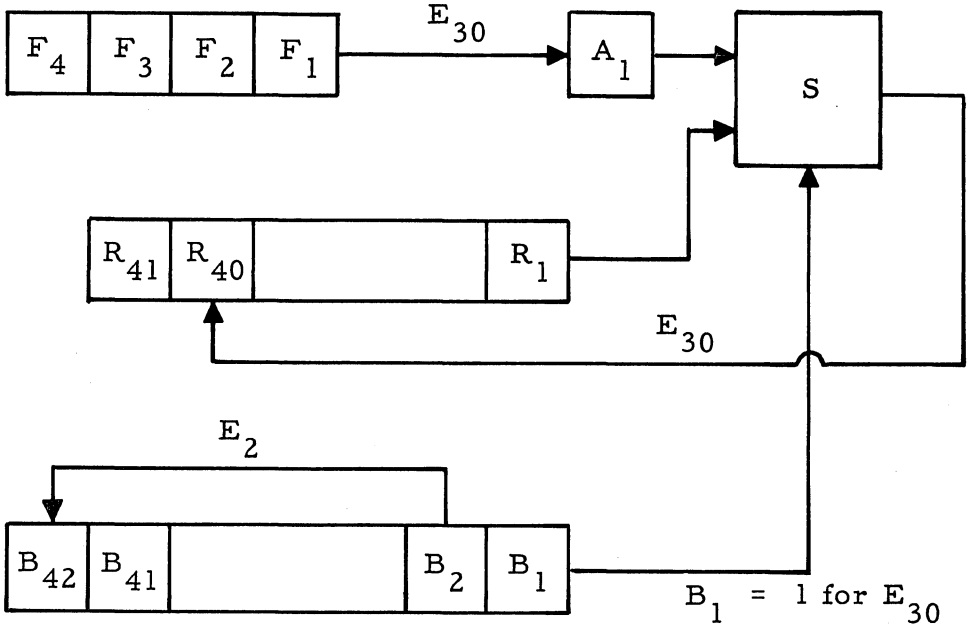
$$ {}_1e_{30} = E_{20}\, N_0\, T_{41a}\, C $$

$$ {}_0e_{30} = I_4\, E_{30}\, T_{41a}\, C $$

---

*E_{30} logic is discussed in operation 7.

During E_{30}, the F-register (input digit) is added via the A-register to the R-register as follows:



The logic involved is:

$$1a_1 = F_1 E_{30} C$$

$$0a_1 = F_1' E_{30} C$$

$$1r_{40} = S E_{30} I_4 C$$

$$0r_{40} = S' E_{30} I_4 C$$

$$1b_{42} = B_2 N_0 P_0 E_2 T_{41}' C$$

$$0b_{42} = B_2' N_0 P_0 E_2 C$$

$$1s = A_1 B_1 R_1 K_a C + A_1 R_1' K_a' C + A_1' B_1 R_1 K_a' C + A_1' R_1' K_a C$$

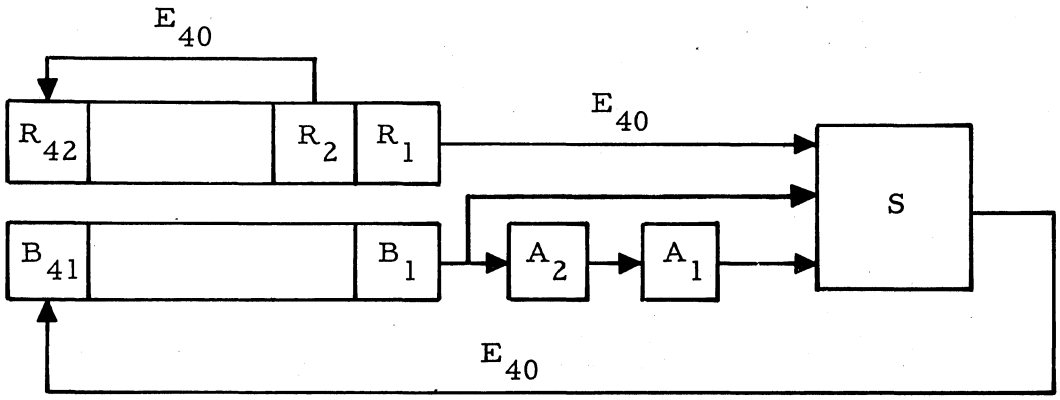
$$0s = A_1' R_1' K_a' C + A_1' B_1 R_1 K_a C + A_1 R_1' K_a C + A_1 B_1 R_1 K_a' C$$

Also the F-register is being cleared during E_{30}:

$$0^{f}_{4} = E_{30}\, C$$

8. Multiplying B by "ten" during E_{40}

A similar process, to that described in 6, is used during E_{40} to multiply the contents of B by "ten."



The logic is:

$$1^{e}_{40} = E_{30}\, N_0\, T_{41a}\, C$$

$$0^{e}_{40} = E_{40}\, T_{41a}\, C$$

$$1^{a}_{1} = A_2\, N_0\, P_0\, E_1'\, T_1'\, C$$

$$0^{a}_{1} = A_2'\, I_3'\, E_{30}'\, C + N_0\, P_0\, E_2'\, T_{41}\, C_{(Synch)}$$

$$1^{a}_{2} = B_1\, E_{40}\, T_{41}'\, C$$

$$0^{a}_{2} = B_1'\, E_{40}\, T_{41}'\, C + A_3'\, E_{40}\, T_{41}\, C$$

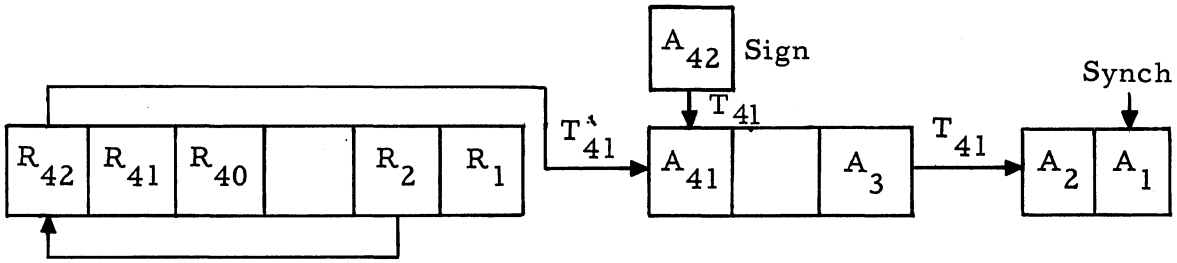
$$1^{b}_{41} = S\, E_{40}\, C$$

$$0^{b}_{41} = S'\, E_{40}\, T_{41a}'\, C$$

$$1^{r}_{42} = R_2\, E_{40}\, C$$

$$0^{r}_{42} = R_2'\, E_{40}\, C$$

9. Copying R into A during E_{40}



The logic involved is:

$$1^{a}_{41} = R_{42}\, E_{40}\, T_{41a}'C + A_{42}\, E_{40}\, T_{41a}\, C$$

$$0^{a}_{41} = R_{42}'\, E_{40}\, T_{41a}'C + A_{42}'\, E_{40}\, T_{41a}\, C$$

$$1^{a}_{2} = A_3\, E_{40}\, T_{41}\, C$$

$$0^{a}_{2} = A_3'\, E_{40}\, T_{41}\, C$$

$$0^{a}_{1} = \underbrace{N_0\, P_0\, T_{41}\, E_2}_{\text{(Synch)}}\, C$$

$$1^{r}_{42} = R_2\, E_{40}\, C$$

$$0^{r}_{42} = R_{2'}\, E_{40} C$$

$$1^{r}_{41} = R_{42}\, E_{40} C$$

$$0^{r}_{41} = R_{42'} E_{40} C$$

$$1^{r}_{40} = R_{41}\, E_{30}'U_{41}'C$$

$$0^{r}_{40} = R_{41}' E_{30}' U_{41}' C$$

$$0^{r}_{1} = E_{40} T_{41a} C_{(Synch)}$$

10. Other operations of E_{40}

It is unknown (at this time), to the computer, whether or not the number in the A-register is to be filled as a fraction or a a whole number. In order to be ready for a possible fractional entry, a "+" sign is forced into the B_{41} flip-flop as follows:

$$1^{b}_{41} = E_{40} T_{41a} C$$

At $E_2' T_{41}$, P_0 is terminated and A, B, and R recirculate

$$0^{p}_{0} = P_0 E_2' T_{41a} C$$

The E-counter is reset to "010" at the end of E_{40}:

$$0^{e}_{1} = N_0 E_3 E_1 C$$

$$1^{e}_{2} = N_0 E_3 E_1 C$$

$$0^{e}_{3} = N_0 E_3 E_1 C$$

The computer is now ready for the ".", "Enter" process, or another decimal digit.

VISUAL READOUT (D_6 'D_5 D_4 D_3 'D_2 D_1')

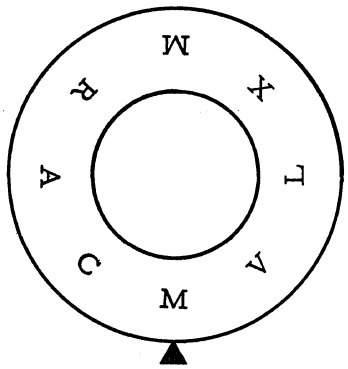
Displays Available

The following may be selected for visual display on the control panel NIXIE register:

Contents of:	Format:
A-Register	BCD, Command, or Octal
R-Register	BCD, Command, or Octal
X-Register	BCD, Command, or Octal
C-Register	BCD, Command, or Octal
Main Memory Location (M)	BCD, Command, or Octal
L - Loop Location (L)	BCD, Command, or Octal
V - Loop Location (V)	BCD, Command, or Octal

Register Selector Switch

The Register Selector switch, located on the control panel, is an 8-position switch with markings as follows:



This switch alone determines the source of the display for the A, R, X, or C positions. The two M positions and the L or V positions require, in addition, the address of the location to be displayed. The address information is set into the control panel via the two 8-position channel selection rotary switches and/or the two 8-position sector selection rotary switches.

DISPLAY COMMAND (DIS, D_6 'D_5 D_4 D_3 D_2 D_1')

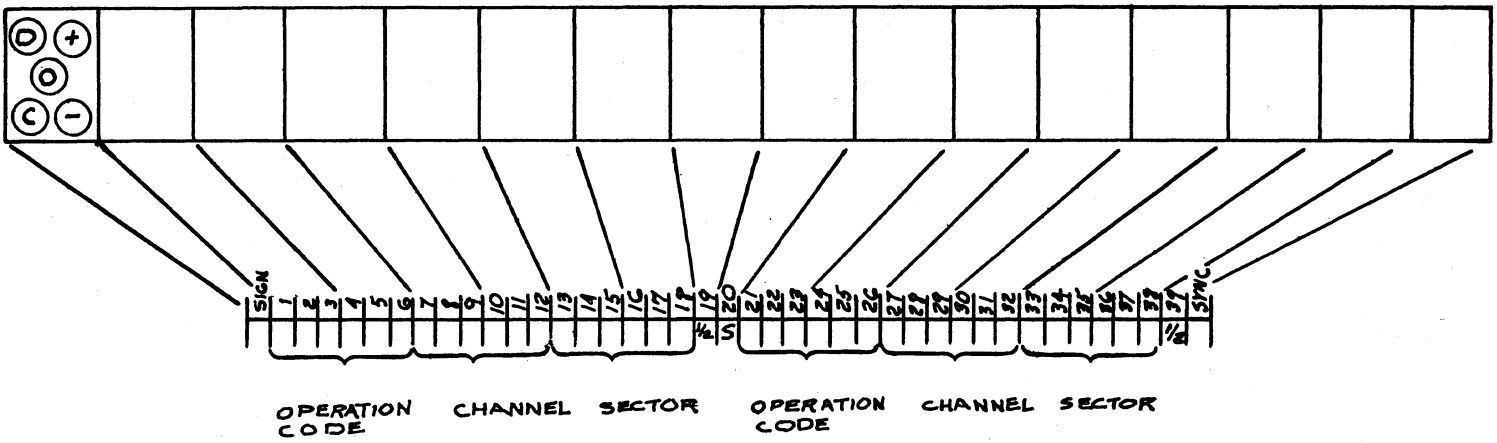
Displays Available

The display command can specify two readout formats; "command" and BCD. If the half-word indicator of the command is a "0", a command format is specified. If the half-word indicator is a "1", a decimal (BCD) format is specified.

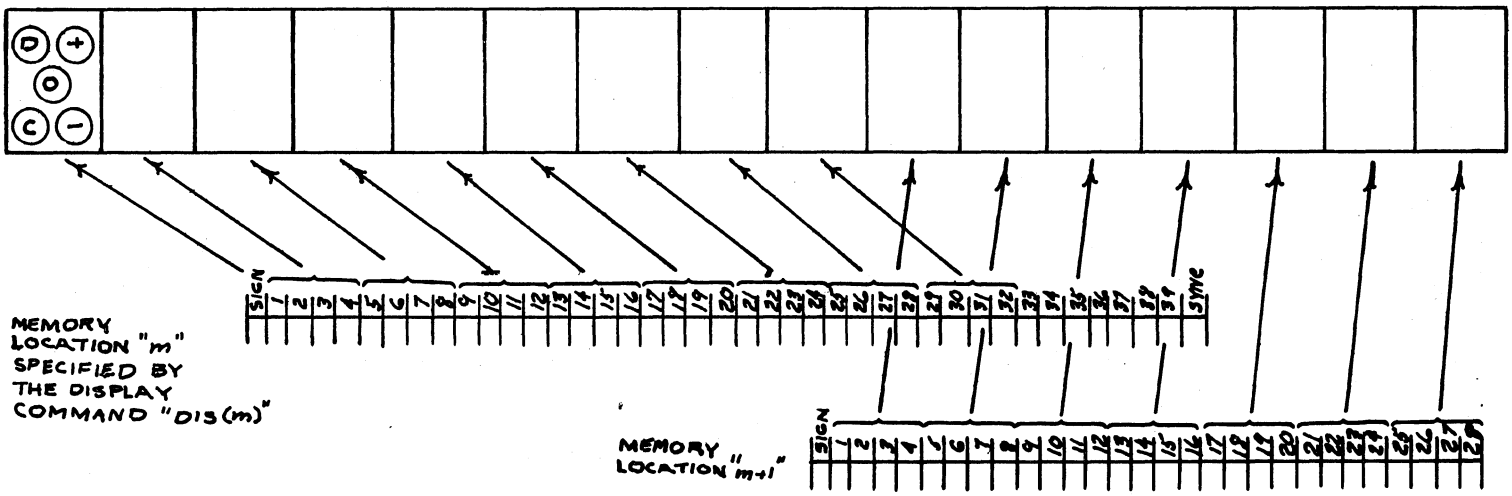
The remainder of the "DIS" commands address gives the memory location of the information to be displayed.

Format Details

Command. The command format is as follows:

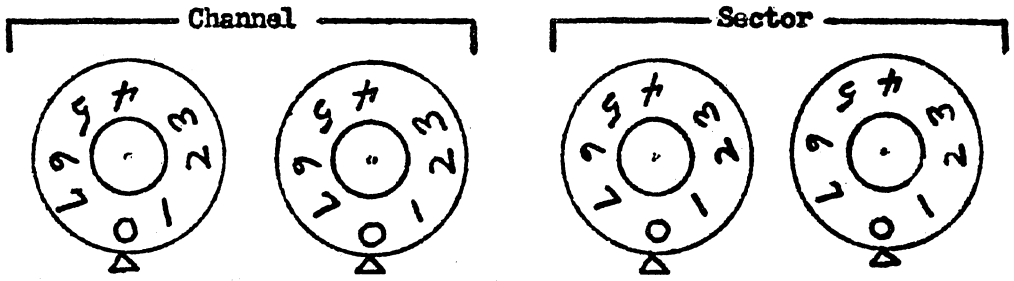


Decimal. The decimal format is as follows:



Channel and Sector Selector Switches

The figure following is of the channel and sector selector switches:



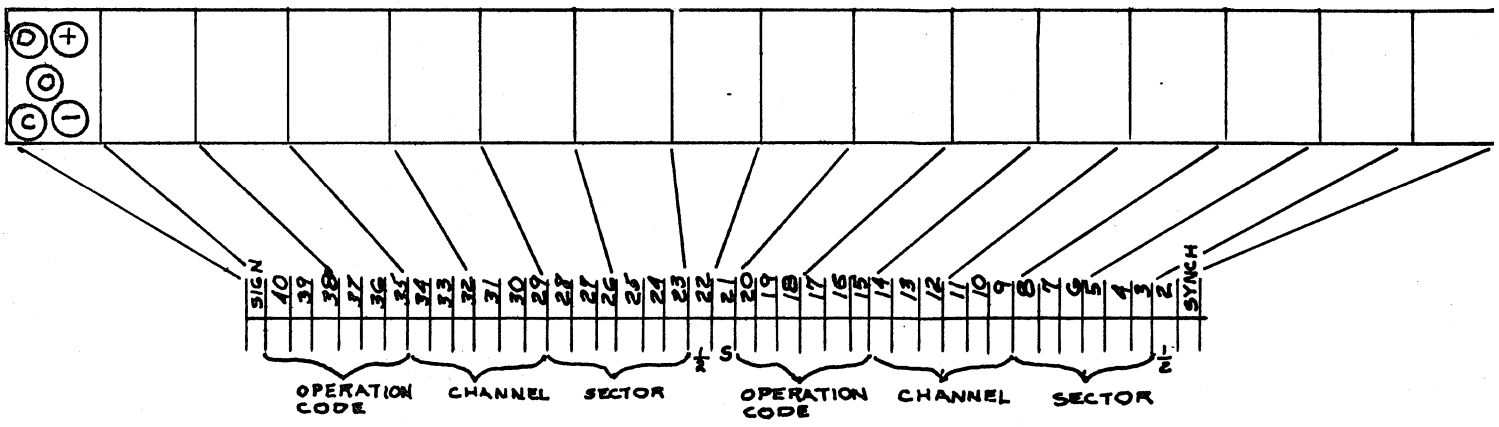
For the M positions of the register selector switch, all four channel and sector selector switches must be set to determine the required main memory location for display. For the L or V positions of the register selector switch, only the least significant sector selector switch need to be set.

Readout Format Buttons

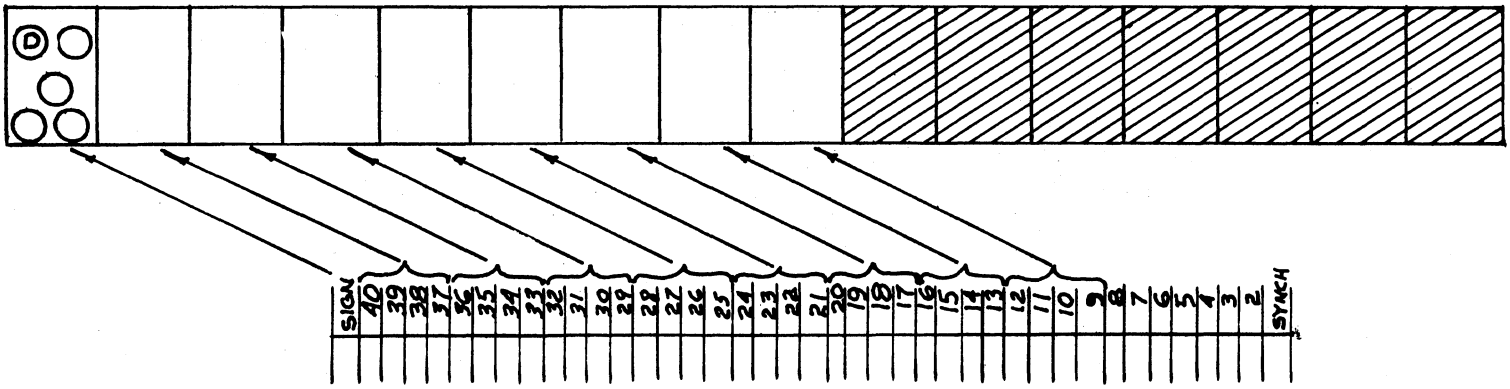
The three readout format buttons, Decimal, Command, and Octal, determine the manual display format.

Format Details

Command. The Command format is as follows:



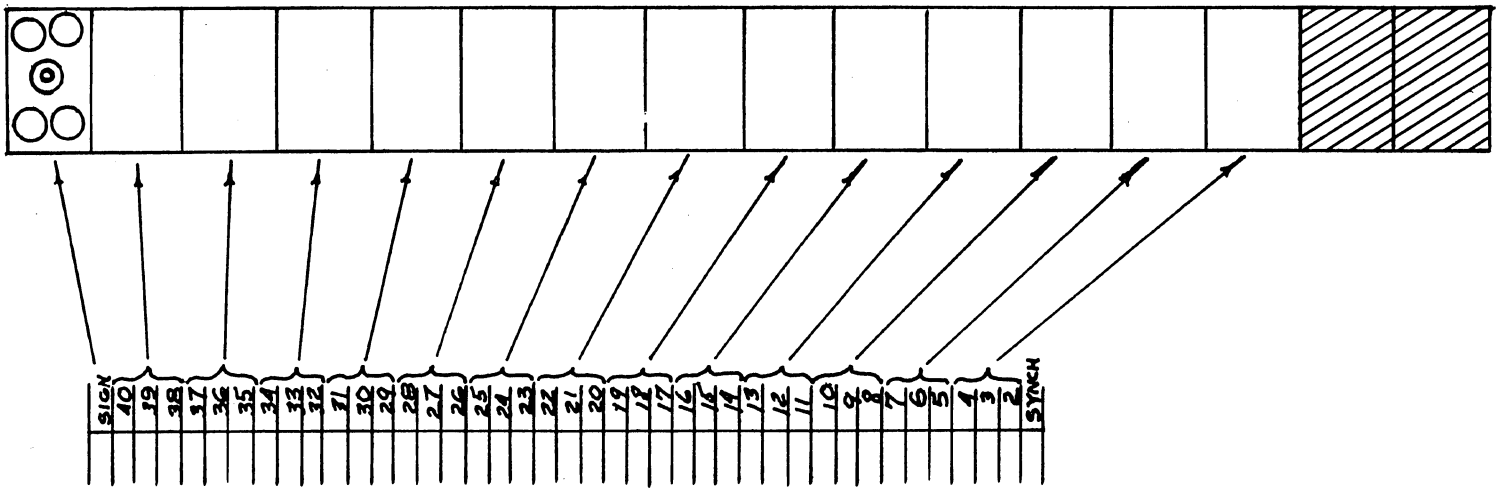
Decimal (BCD). The decimal format is as follows:



The BCD scheme is as follows:

Binary	Displayed
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	Not Used
1011	Not Used
1100	"."
1101	"Blank"
1110	Not Used
1111	"Terminate"

Octal. The octal format is as follows:



Setting the Operation Code Register to $D_6' D_5 D_4 D_3' D_2 D_1'$

Depressing any one of the three readout buttons causes R_b to become true. This, in turn, causes the R_{c1} (readout timing) flip-flop to become one-set if it is I_4.

$$1^{r}_{c1} = R_b I_4 C$$

The operation code register is forced to the manual readout array $D_6' D_5 D_4 D_3' D_2 D_1'$ by R_{c1} and I_4.

$$0^{d}_{6} = R_{c1} C$$

$$1^{d}_{5} = I_4 C$$

$$1^{d}_{4} = R_{c1} C$$

$$_0d_3 = I_4 C$$

$$_1d_2 = R_{c1} C$$

$$_0d_1 = R_{c1} C$$

Filling the B-Register with Manual Display Information

From Main Memory or High-Speed Loops

1. The computer monitors the control panel channel selector switches for use in displaying main memory or high-speed loop locations, the control panel channel selection information is placed in the channel register.

The channel register is first cleared of old information by:

$$_0c_i = R_b I_4 C +$$

where i = 1-6

The two 8-position channel selector switches each use three contacts ($J_{7, 8, 9}$ or $J_{10, 11, 12}$) per position for this purpose. Therefore any arbitrary channel setting will generate six signals. For example if the switches are set to channel "25_8" then the "J" signal would be:

$J_{12}\ J_{11}\ J_{10}\ J_9\ J_8\ J_7$

0 1 0 1 0 1

where: 1 is -12.5 volts and 0 is 0 volt

The state of the J signals determine the new setting of the channel register as follows:

$$1^{c}_{i} = J_{i+6} R_{c1} D_3' C+$$

where i = 1-6

2. Monitoring the control panel sector selector switches. In order to display a particular location of the main memory, or the L and V loops, a sector search must be made using, in part, the two sector selector switches.

Before the search can be made the system must go to the $I_2 I_4$ state:

$$1^{i}_{2} = R_{c1} R_b' I_2' T_{41a} C+$$

The sector selector switches monitor the prime (or unprime) side of the sector counter flip-flops ($S_6 \ldots . S_1$) dependent on the sector count placed into the two switches. For example, if the sector selector switches (contacts J_6, 5, 4, 3, 2, 1) specify sector "32_8" the sector counter array monitored is S_6', S_5, S_4, S_3', S_2, S_1'. When the sector counter reaches the count of 011010 (this array is in the sector counter by T_1 of sector 31_8) each J_6, 5, 4, 3, 2, 1 contact monitors -12.5 volts from the S outputs. These agreement signals are used to zero-set S_0 at T_2. If S_0 is in its zero-set state by $I_2 T_{41}$, sector agreement has taken place and the next sector to pass the main memory read heads is the one requested by the control panel sector selector switches.

$$1^{s}_{0} = I_2 R_{c1} T_1 C+$$

$$0^{s}_{0} = E_3 J_3 J_2 J_1 R_{c1} I_2 T_2 C+ \; T_{41} C$$

The E_3 signal provides the J_6, J_5 and J_4 signals:

$$1^{e}_{3} = J_6 J_5 J_4 I_2 R_{c1} T_1 C+$$

$$0^{e}_{3} = I_2 R_{c1} T_{41} C+$$

For main memory reading, D_0 is one-set by:

$$1^{d}_{0} = D_3' S_0' L_{p1}' I_2 T_{41} C+ \; D_1' S_0' L_{p1}' I_2 T_{41} C+$$

For high-speed loop reading, D_{10} is one-set by:

$$ {}_{1}d_{10} = S_7' \, L_{p1} \, D_{10}' \, I_3' \, D_3' \, T_{41} C + S_7' \, L_{p1} \, D_{10}' \, I_3' \, D_1' \, T_{41} C $$

3. Gating information into B-register. The B-register copies the main memory read flip-flop (M_r) if D_0 is one-set or the loop read flip-flop (K_0) if D_{10} is one-set:

$$ {}_{1}b_{41} = M_r \, D_0 \, I_2 C + K_0 \, D_{10} \, I_2 C $$

$$ {}_{0}b_{41} = M_r' \, D_0 \, I_2 C + K_0' \, D_{10} \, I_2 C $$

b_{40} copies b_{41} etc.

After one word-time, D_0 and D_{10} are zero-set by:

$$ {}_{0}d_{0} = D_0 \, T_{41} C $$

$$ {}_{0}d_{10} = D_{10} \, T_{41} C $$

...and the state change flip-flop, N_{11}, is one-set for one bit time by:

$$ {}_{1}n_{11} = D_0 \, T_{40} C + D_{10} \, T_{40} C $$

$$ {}_{0}n_{11} = T_{41a} C $$

The I_2 state is then terminated by:

$$ {}_{0}i_{2} = I_2 \, N_{11} C $$

From One-Word Registers

1. The register selection switch. The one-word registers A, R, C, or X may also be displayed by the proper placement of the register selector. No memory search is required for this display.

2. Gating information into the B-register. For the one-word registers, B copies the register specified by register selection switch signals A_d, R_d, Z_d, or X_d.

$$ {}_1b_{41} = A_1 A_d I_2 R_{c1} C + X_1 X_d I_2 R_{c1} C + R_1 R_d I_2 R_{c1} C + Z_1 Z_d I_2 R_{c1} C $$

$$ {}_0b_{41} = A_1' A_d I_2 R_{c1} C + X_1' X_d I_2 R_{c1} C + R_1' R_d I_2 R_{c1} C + Z_1' Z_d I_2 R_{c1} C $$

The I_2 state is terminated by:

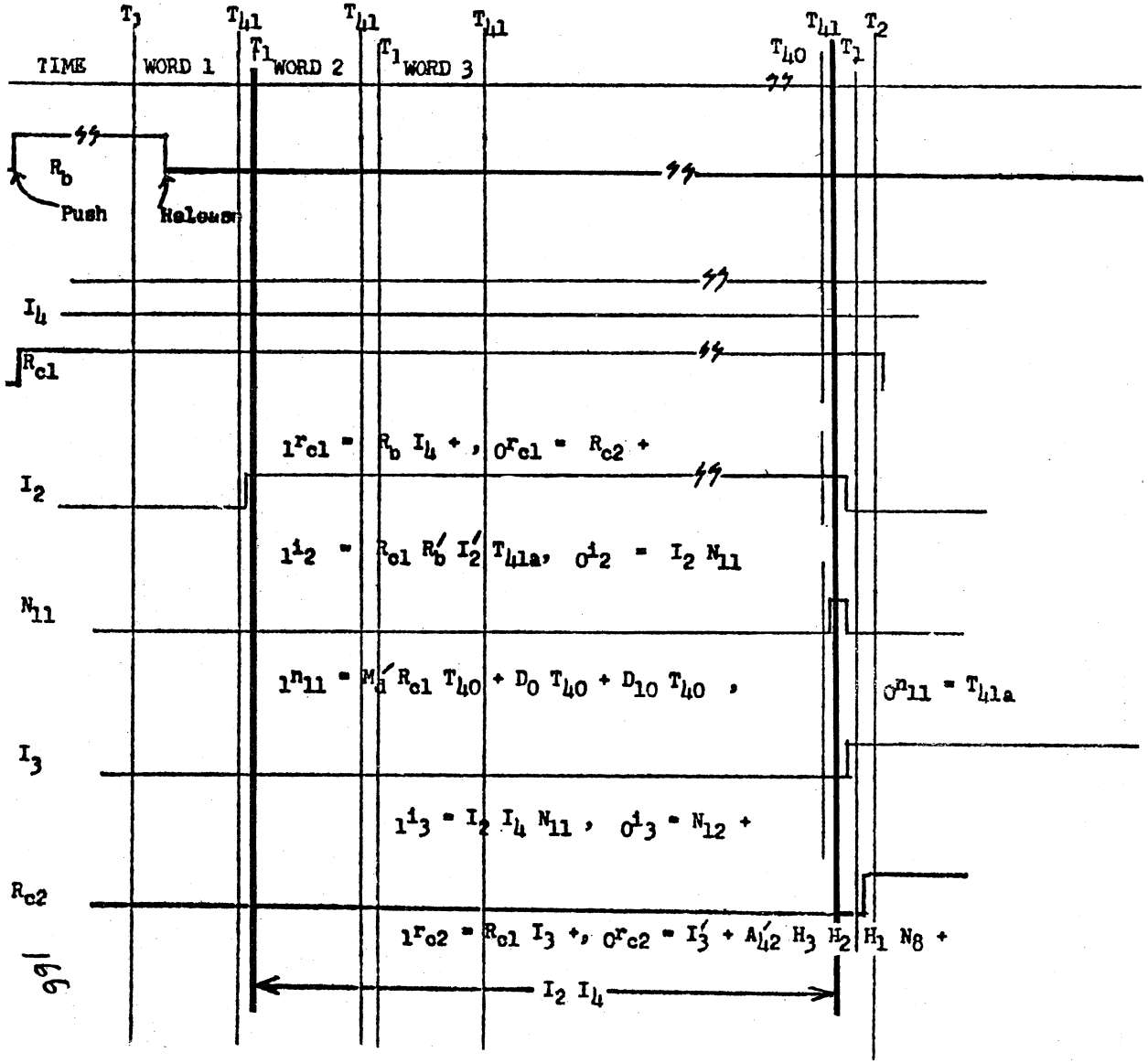
$$ {}_1n_{11} = M_d' R_{c1} T_{40} C $$ (M_d is main memory or loop signal from register selector)

$$ {}_0n_{11} = T_{41a} C $$

$$ {}_0i_2 = I_2 N_{11} C $$

Summary of I_2 I_4 State

During the I_2 I_4 state the requested display information is gated into the B-register. After the gating is finished, the I_2 flip-flop is zero-set and the I_3 flip-flop one-set. We thus transfer to the I_3 I_4 state. The following timing diagram summarizes the sequencing of the I_2 I_4 state.



Timing Diagram For $I_2\ I_4$ State For Manual Readout

Transferring the "B" Register to the "F" Register

General

Now that the requested readout information is in the "B" register, the I_3 I_4 state is started by:

$$1^{i}_{3} = I_2\ I_4\ N_{11}\ C$$

During I_3 I_4 the information contained in "B" will be shifted character-wise into the output register (F1, 2, 3, 4) in the requested octal, B.C.D., or command format. The format request is made via the control panel "readout" buttons.

Format Control

The format request, from the momentary "readout" format buttons, (or the "DIS" command) is stored in flip-flops F_{c1} and F_{c2}. The following table shows the possible arrays before I_3 I_4:

	F_{c2}	F_{c1}	Format Requested
Manual Readout	1	1	Octal (0_f = 1)
	0	0	B.C.D. (D_f = 1)
	1	0	Command (C_f = 1)
Programed Readout	0	0	Programed Readout in B.C.D. (1/2 word indicator was "1")
	1	0	Programed Readout in Command (1/2 word indicator was "0")

The logic for the manual readout follows:

$$1^{f}_{c1} = 0_f\ C$$

$$0^{f}_{c1} = C_f\ C + I'_4\ R_{c2}\ U_1\ C$$

$$1^{f}_{c2} = 0_f\ I_4\ C + C_f\ I_4\ C + A'_{42}\ T_x\ D'_1\ C$$

$$0^{f}_{c2} = D_f\ I_4\ C + A_{42}\ T_x\ D'_1\ C$$

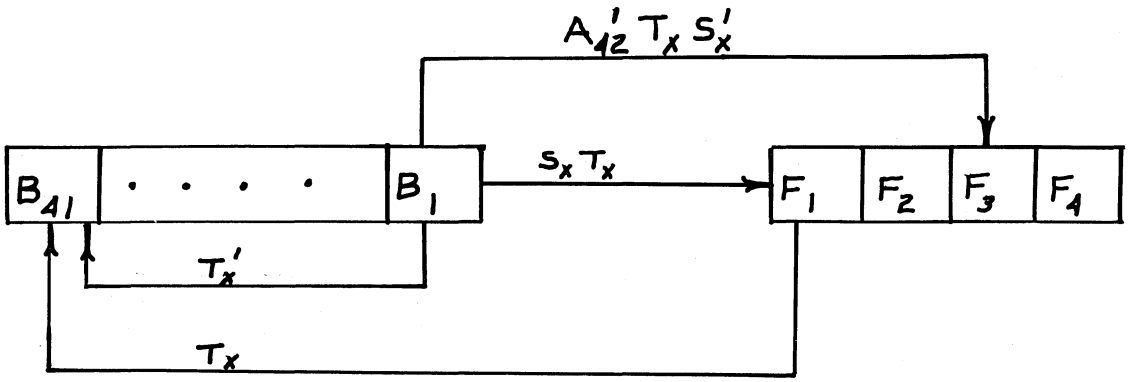
NOTE

A_{42} logic is involved in programed readout and B.C.D. manual readout.

Octal Format

"B" and "F" Configuration

In order to shift octal format characters into the "F" register, the following shift mechanism is used:



When $T_x = 0$, the "B" register recirculates. When $T_x = 1$, two lengthened loops are possible under the control of S_x. After 41 clocks, the previous contents of "B" will be shifted 1-bit left if $T_x S_x = 1$ or will be shifted 3-bits left if $T_x S'_x = 1$. The "F" and "B" logic follow:

$1^f4 =$ $0^f4 =$	not involved

$1^f3 = B_1\ A'_{42}\ S'_x\ T_x\ M'_2\ N'_{12}\ C$
$0^f3 = B'_1\ A'_{42}\ T_x\ M'_2 C + M_3\ N_8\ C$

$1^f2 = F_3\ E'_{30}\ T_x\ N'_{12}\ C$
$0^f2 = F'_3\ T_x C + M_3\ N_8\ C$

$1^{f}1 = F_2\ S'_x\ T_x\ N'_{12}C + B_1\ S_x\ T_x\ N'_{12}\ C$
$0^{f}1 = F'_2\ S'_x\ T_xC + B'_1\ S_x\ T_xC + M_3\ N_8\ C$

$1^{b}41 = F_1\ R'_{c2}\ T_xC + B_1\ T'_x\ M'_0\ M'_8\ M'_{10}\ I_3\ C$
$0^{b}41 = F'_1\ R'_{c2}\ T_xC + B'_1\ T'_x\ M'_0\ M'_8\ M'_{10}\ I_3\ C$

The rest of "B" copies left to right.

NOTE

A_{42} is one-set for B.C.D. format only:

$1^{a}42 = F'_{c2}\ D_4\ I_4\ C$
$0^{a}42 = H'_0\ I_2\ T_1\ C$

1. General. The requirements for T_x and S_x (for octal readout) are summarized in the following table:

H Counter	Word Time of $I_3 I_4$	Array Required ($A_{42} = 1$)	Purpose
0	1	$T = 0$	Recirculate "B"
	2	$S_x\ T_x = 1$	To shift sign bit into F_1
	3	$T_x = 0$	Recirculate "B" while sign is copied into sign NEON
1	4	$S'_x\ T_x = 1$	Shift first octal character into F_3, F_2, F_1
	5	$T_x = 0$	Recirculate "B" while first octal character is copied into most significant NIXIE

H Counter	Word Time of I_3 I_4	Array Required (A_{42} = 1)	Purpose
2	6	$S_x' T_x = 1$	Shift second octal character into F_3, F_2, F_1
	7	$T_x = 0$	Etc.
		Etc.	Etc.
13	28	$S_x' T_x = 1$	Shift last octal character into F_3, F_2, F_1
	29	$T_x = 0$	Recirculate "B" while 13th octal bit is copied into 13th (left to right) NIXIE. 14 and 15 are blank.
14	30	$S_x' T_x = 1$	Shift garbage into F_3, F_2, F_1

2. "H" Counter (readout control flip-flop's $H_{4,3,2,1}$)

The number of characters shifted through the output register is stored in the "H" counter. For octal format, 14 characters are displayed.

The counter determines the character count by summing the N_8 flip-flop outputs. N_8 is dependent on the R_{c3} flip-flop which, in turn, monitors T_x. These operations follow.

The "H" counter is an 8, 4, 2, 1 binary counter. The logic is listed below:

$1^h1 = H_1' R_{c4}' N_8 C$
$0^h1 = H_1 N_8 C + U_1 C_r' A_0' C$
$1^h2 = H_2' H_1 N_8 C$
$0^h2 = H_2 H_1 N_8 C + U_1 C_r' A_0' C$

$1^{h}3 = H_3' H_2 H_1 N_8 C$
$0^{h}3 = H_3 H_2 H_1 N_8 C + U_1 C_r' A_0' C$

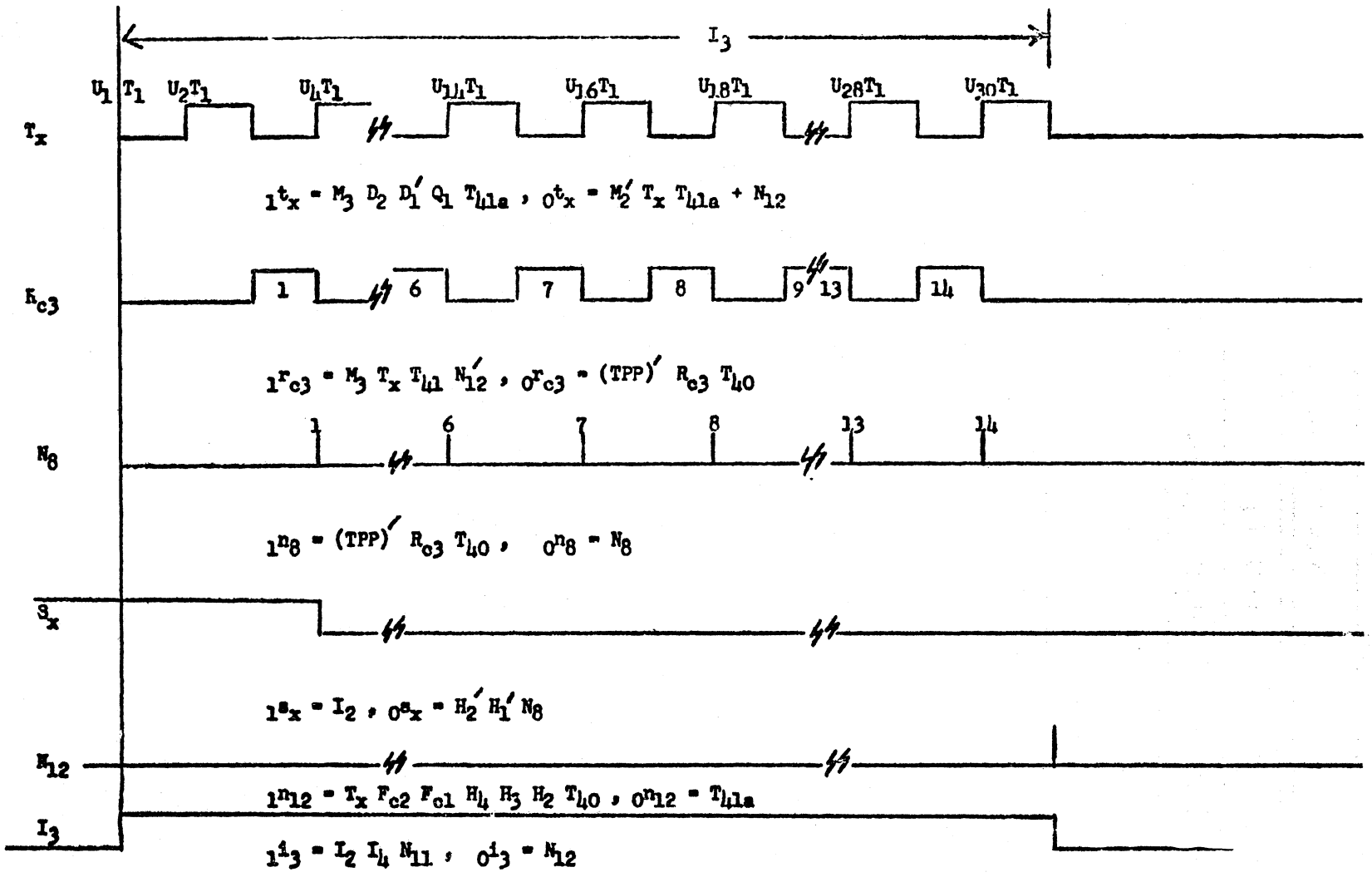
$1^{h}4 = H_4' H_3 H_2 H_1 N_8 C$
$0^{h}4 = H_4 H_3 H_2 H_1 N_8 C + N_{12} C + U_1 C_r' A_0' C$

$1^{n}8 = (TPP)' R_{c3} T_{40} C$
$0^{n}8 = N_8 C$

The following table shows the counting process:

H_4	H_3	H_2	H_1	
0	0	0	0	Cleared by $U_1 C_r' A_0' (T_1$ of $U_1)$
0	0	0	1	First N_8 (T_{41} of U_3)
0	0	1	0	Second N_8 (T_{41} of U_5)
0	0	1	1	Third N_8 (T_{41} of U_7)
.	.	.	.	
.	.	.	.	
.	.	.	.	
X	X	X	X	R_{c3}' Terminates Counting

The timing chart on the following page shows the interrelation of the T_x, R_{c3}, N_8, S_x, N_{12} and I_3 flip-flops:



Timing Diagram For I_3 I_4 State Of "Octal" Manual Readout

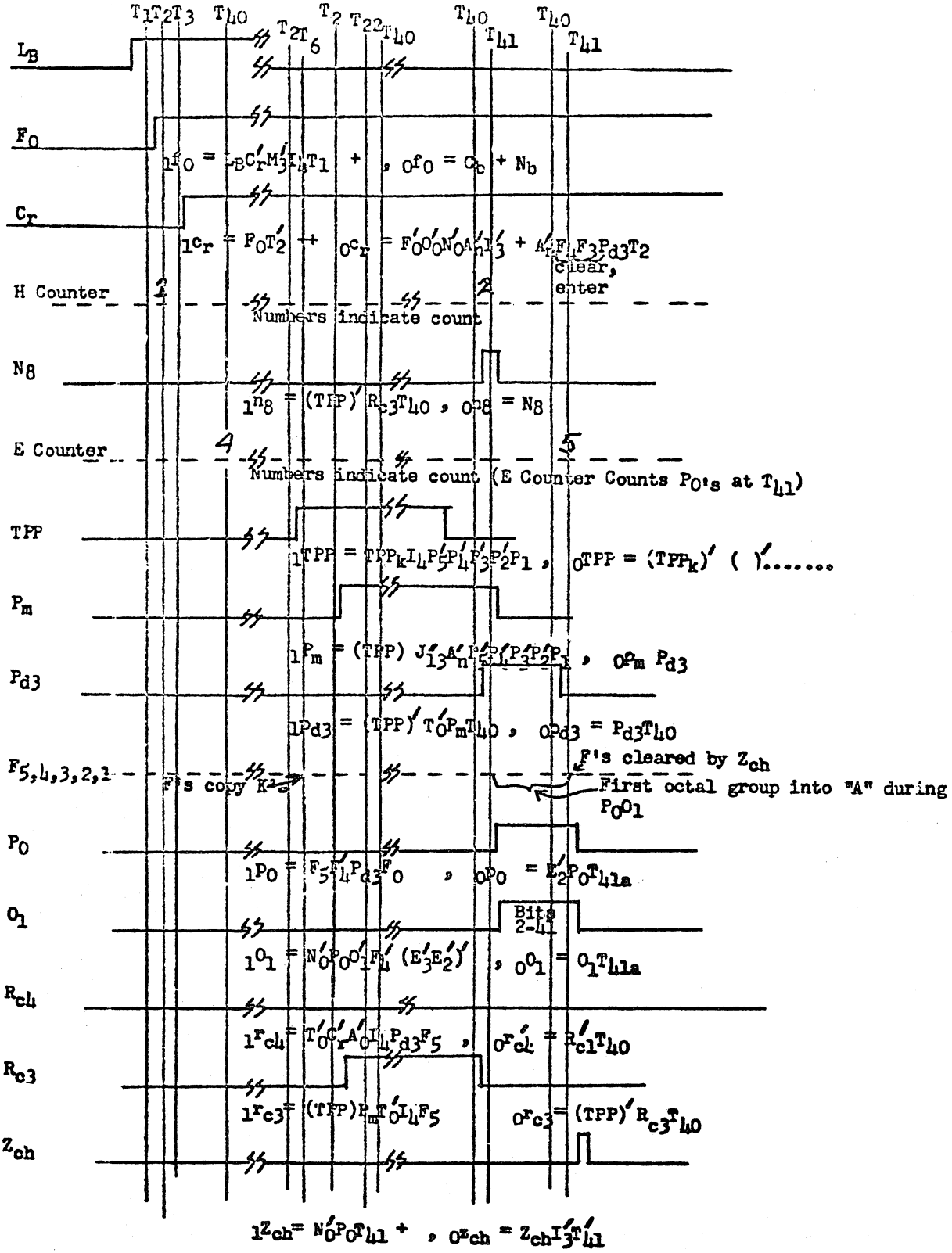


Figure 35. Filling the first Octal Character into "A"
(The other characters are handled similarly)

Command Format

"B" and "F" Configuration

The same loops used in the octal readout are used for command readout with, however, differences in logic of S_x and N_{12} that provide the variance in format. (See "B" and "F" configuration)

1. General. The following table lists the requirements for S_x and T_x to provide command format:

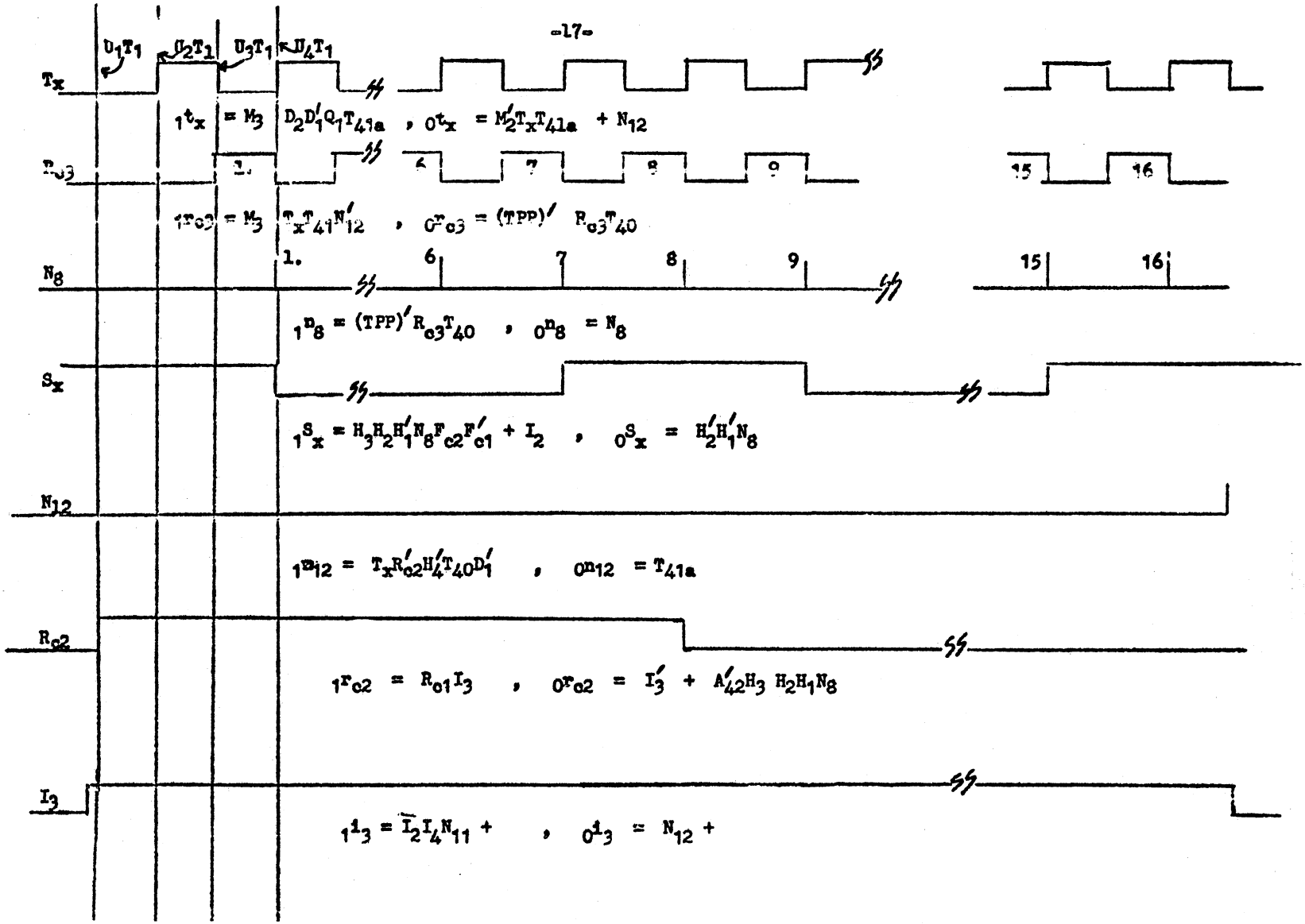
"H" Counter	Word-Time of I_3	Array Required ($A_{42'} = 1$)	Purpose
0	1	$T_x = 0$	Recirculate "B"
	2	$A_{42'} S_x T_x = 1$	Shift Sign into F_1
	3	$T_x = 0$	Copy sign into sign neon
1	4	$A_{42'} S_{x'} T_x = 1$	Shift first octal group into F's
	5	$T_x = 0$	Copy first octal group into NIXIE
2	6	$A_{42'} S_x T_{x'} = 1$	Shift second octal group into F's
	7	$T_x = 0$	Copy second octal group into NIXIE
3	8	$A_{42'} S_{x'} T_x = 1$	Shift third octal group into F's
	9	$T_x = 0$	Copy third octal group into NIXIE
4	10	$A_{42'} S_{x'} T_x = 1$	Shift fourth octal group into F's
	11	$T_x = 0$	Copy fourth octal group into NIXIE
5	12	$A_{42'} S_{x'} T_x = 1$	Shift fifth octal group into F's

"H" Counter	Word-Time of I_3	Array Required ($A_{42'} = 1$)	Purpose
	13	$T_x = 0$	Copy fifth octal group into NIXIE
6	14	$A_{42'} S_{x'} T_x = 1$	Shift sixth octal group into F's
	15	$T_x = 0$	Copy sixth octal group into NIXIE
7	16	$S_x T_x = 1$	Shift 1/2 word indicator into F's
	17	$T_x = 0$	Copy 1/2 word indicator into NIXIE
8	18	$S_x T_x = 1$	Shift sign of second command into F's
	19	$T_x = 0$	Copy sign into NIXIE
15	31	$S_x T_x = 1$	Shift 1/2 word indicator into F's
	32	$T_x = 0$	Copy 1/2 word indicator into NIXIE
16	33	$S_x T_x = 1$	Shift garbage into F's

"H" Counter and Timing Diagram. The "H" counter operates identically to its operation during "octal" readout. See Octal Format-2. "H" counter.

The following timing chart shows the interrelation of the T_x, R_{c3}, N_8, R_{c2}, S_x, and N_{12} flip-flops. Note that only the logic for S_x and N_{12} is changed from that used in the octal format case.

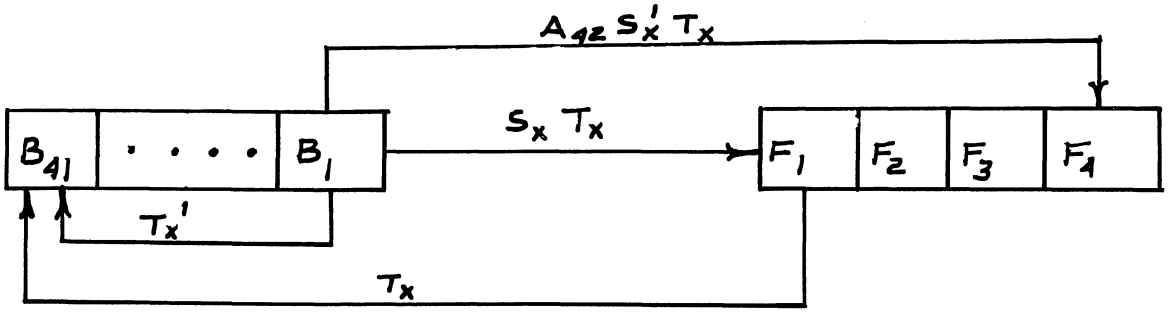
$1^{S}x = H_3\ H_2\ H_1'N_8\ F_{c2}\ F_{c1}'C+I_2C$
$0^{S}x = H_2'\ H_1'\ N_8\ C$
$1^{n}12 = T_x\ R_{c2}'\ H_4'\ T_{40}\ D_1'\ C$
$0^{n}12 = T_{41a}\ C$



Timing Diagram For I_3 I_4 State of Command Format Manual Readout

"B" and "F" Configuration

The loops used for B. C. D. format are as follows:



NOTE

A_{42} is one-set only for B. C. D.,

$$1^{a}42 = F_{c2}' D_4 I_4 C$$

1. General. The table on the following page lists the requirements for S_x and T_x to provide B. C. D. readout.

"H" Counter	Word-Time of I_3	Array Required ($A_{42} = 1$)	Purpose
0	1	$T_x = 0$	Recirculate "B"
	2	$S_x T_x = 1$	Shift sign into F_1
	3	$T_x = 0$	Copy sign into NEON
1	4	$S_x' T_x = 1$	Shift 4 bits (B. C. D. character) into F's
	5	$T_x = 0$	Copy character into NIXIE
8	18	$S_x' T_x = 1$	Shift last B.C. D. to "F" and one-set "B" register
	19	$T_x = 0$	Copy into NIXIE
9	20	$S_x' T_x = 1$	Shift terminate code into F's

2. "H" Counter and Timing Diagram

a. General. The "H" counter operates in an identical manner to that of "octal" or "command" readout. See Octal Format-2. "H" counter.

The following Timing Chart shows the interrelation of T_x, R_{c3}, N_8, N_{12}, R_{c1}, R_{c2} and I_3 flip-flops.

b. Details of manual B. C. D. readout termination. In order to terminate the display after 9 characters (counting sign), the terminate code (1111) is internally forced to appear as the last (tenth) character shifted into the "F" register. This is accomplished as follows:

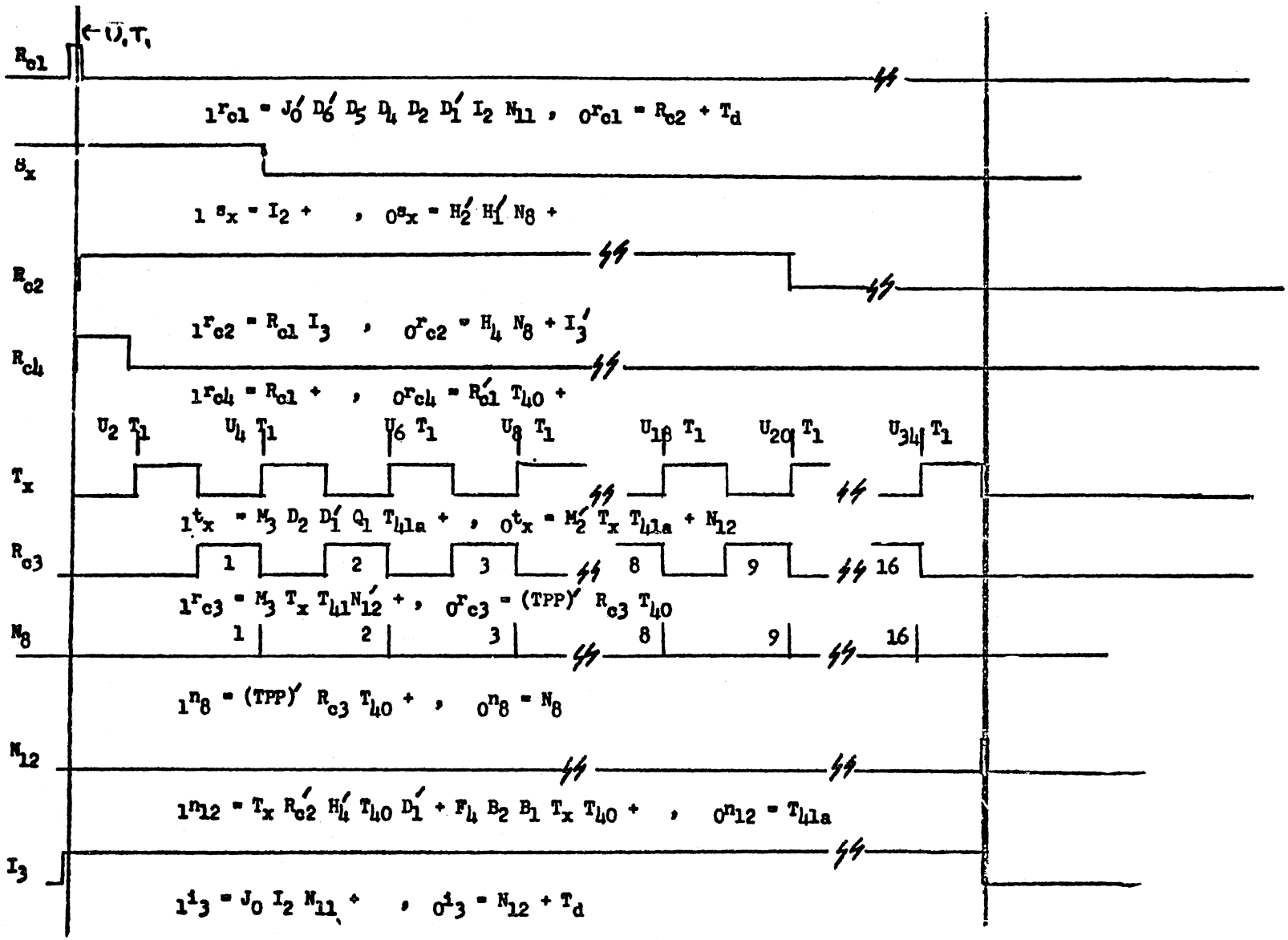
B_{42} is one-set; $_1b_{42} = \boxed{R_{c1}\ I_4 C}$

and is not allowed to zero-set during I_4; $_0b_{42} = I_4'\ R_1'\ R_{c2}$.

B_{41} is allowed (for B. C. D. only) to copy B_{42} after the eighth character

$$\boxed{{}_1b_{41} = B_{42}\ A_{42}\ R_{c2}\ H_4\ T_x C}$$

$$\boxed{{}_0b_{41} = B_{42}'\ A_{42}\ R_{c2}\ H_4\ T_x C}$$



I_3 State For PROGRAMED B. C. D. Readout ("DIS")

After one word-time of B's shifting, the "1" copied into B_{41} has one-set the entire B-register. The ninth character was shifted into F_4, F_3, F_2, F_1 during this word-time. The ninth character is read out during the next word-time (9th R_{c3} pedestal). During the last word-time of I_3, the terminate code (111X) is shifted into the "F" register. The code is monitored by N_{12} two clocks before, i.e., T_{40}, the shift is completed. At this time, the leading three coding bits are in F_4, B_2, and B_1. Thus, N_{12} is one-set by:

$$1^{n}_{12} = F_4 B_2 B_1 T_x T_{40} C$$

in turn, I_3 is terminated by N_{12} as follows:

$$0^{i}_{3} = N_{12} C$$

I_3 may also be terminated earlier by an actual (not forced by logic) terminate character (see table on previous page).

It must be emphasized that the internal binary array in locations "m" and "m + 1" had to be previously processed by a programed routine in order to provide a useful readout. One does not read out the true decimal equivalent of a register(s)(or location(s))contents without first converting the contents via a routine. This is also true for manual readout.

By the use of the "Decimal Point" code, the "Blank" code and the "Terminate" code, several numbers may be displayed at the same time by means of judicious programing.

SETTING THE CODE OPERATION REGISTER TO D_6 D_5 D_4 D_3 D_2 D_1'

The operation code register is set during I_2 by the "DIS" command in exactly the same manner as in any command. See 9.1 of the "Description of the Logical Design of the RECOMP II."

FILLING THE B AND R REGISTERS

Filling the "B" Register with Information Specified by "m" of "DIS (m)"

The display command, "DIS", is an I_2, Type-3 and is handled, during I_2, in essentially the normal manner. See 9.2.2.2 of "Description of the Logical Design of the RECOMP II." The Timing Diagram on page 230 gives the special readout timing.

Filling the "R" Register with the "m + 1" Information for BCD Readout

One characteristic difference between manual and programed readout is the use of two memory locations for programed B.C.D. readout. During I_3 the second word is placed in the "R" register by:

$$1^{r}_{41} = M_r L_{p1}' D_6' D_4 I_3 I_4' D_1' U_1 C + K_0 L_{p1} D_6' D_4 I_3 I_4' D_1' U_1 C$$

$$0^{r}_{41} = M_r' L_{p1}' D_6' D_4 I_3 I_4' D_1' U_1 C + K_0' L_{p1} D_6' D_4 I_3 I_4' D_1' U_1 C$$

The remainder of "R" shifts so that at $I_3\ U_2\ T_1$ the "R" register has the contents of location "m + 1".

Transferring the "B" Register to the "F" Register

General

Now that the requested readout information is in the "B" (and "R" for B.C.D.) register, the I_3 state is started by:

$$1^{i}_{3} = J_0' I_2 N_{11} C$$

During I_3, the information contained in "B" (and "R" for B.C.D.) will be shifted character-wise into the output register ($F_{4, 3, 2, 1}$) in the format requested by the command's 1/2-word indicator bit; 1 for B.C.D., 0 for command.

Format Control

The format request is stored in the F_{c1} and F_{c2} flip flops as follows:

F_{c2}	F_{c1}	Format
0	0	B.C.D.
1	0	Command

The logic involved follows:

$$0^{f}_{c1} = A_{42}\ T_x\ D_1'\ C + I_4'\ R_{c2}\ U_1\ C$$

$$1^{f}_{c2} = A_{42}'\ T_x\ D_1'\ C$$

$$0^{f}_{c2} = A_{42}\ T_x\ D_1'$$

$$1^{a}_{42} = Z_1\ I_2\ I_4'\ N_9\ C$$

$$0^{a}_{42} = H_0'\ I_2\ T_1\ C$$

The Z_1 flip-flop at T_2 (or T_{22}) of I_2 contains the half-word indicator of the second command (or first). If the indicator is a "1", A_{42} is set to one and, in turn, F_{c1} and F_{c2} are zero-set. If the indicator is zero, A_{42} is left in its zero-set state.

Command Format

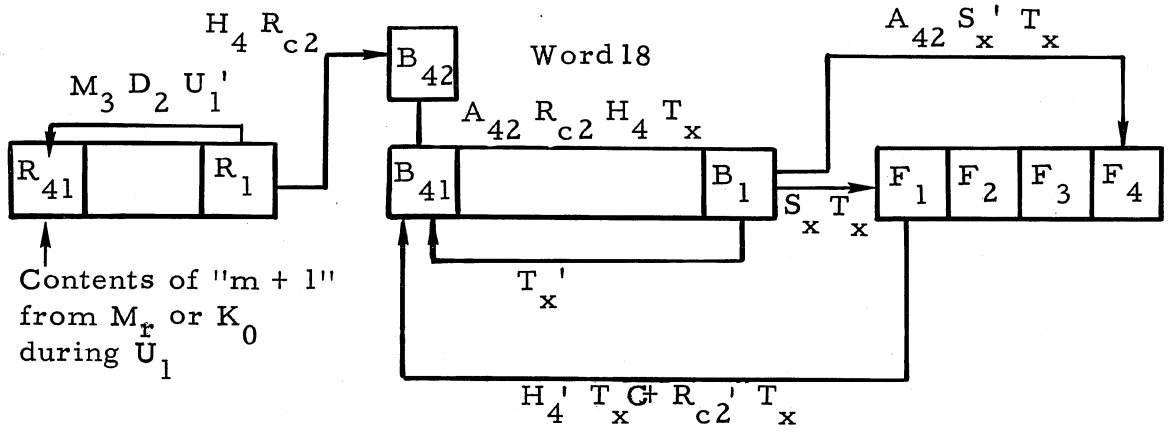
During I_3, the logic and operation for programed command readout is the same as for manual readout.

BCD Format

B, F and R configuration

During I_3, sixteen characters (including sign) are read out. Nine originate in location "m" and seven in location "m + 1".

The loops used for programed B.C.D. format are as follows:



The exact logic required, to mechanize the above diagram, follows:

$$ {}^{1}r_{41} = M_r\ L_{p1}'\ D_6'\ D_4\ I_3\ I_4'\ D_1'\ U_1 C + K_0\ L_{p1}\ D_6'\ D_4\ I_3\ I_4'\ D_1'\ U_1 C + R_1\ M_3\ D_2\ U_1'\ C $$

$$ {}^{0}r_{41} = M_r'\ L_{p1}'\ D_6'\ D_4\ I_3\ I_4'\ D_1'\ U_1 C + K_0'\ L_{p1}\ D_6'\ D_4\ I_3\ I_4'\ D_1'\ U_1 C + R_1'\ M_3\ D_2\ U_1'\ C $$

$$ {}^{1}b_{42} = R_1\ H_4\ R_{c2}\ N_{12}'\ C $$

$$ {}^{0}b_{42} = R_1'\ I_4'\ R_{c2}\ C $$

$$ {}^{1}b_{41} = B_1\ T_x'\ M_0'\ M_8'\ M_{10}'\ I_3 C + F_1\ H_4'\ T_x C + F_1\ R_{c2}'\ T_x C + B_{42}\ A_{42}\ R_{c2}\ H_4\ T_x\ C $$

$$ {}^{0}b_{41} = B_1'\ T_x'\ M_0'\ M_8'\ M_{10}'\ I_3 C + F_1'\ H_4'\ T_x C + F_1'\ R_{c2}'\ T_x C + B_{42}'\ A_{42}\ R_{c2}\ H_4\ T_x\ C $$

$$ {}^{1}f_4 = B_1\ A_{42}\ M_2'\ N_{12}'\ S_x'\ T_x\ C $$

$$ {}^{0}f_4 = B_1'\ A_{42}\ T_x\ C $$

$$ {}^{1}f_1 = B_1\ S_x\ T_x\ N_{12}'\ C + F_2\ S_x'\ T_x\ N_{12}'\ C $$

$$ {}^{0}f_1 = B_1'\ S_x\ T_x C + F_2'\ S_x'\ T_x\ C $$

1. General. The following table lists the requirements for S_x and T_x to provide B.C.D. programed readout.

"H" Counter	Word-Time of I_3	Array Required ($A_{42} = 1$)	Purpose
0	1	$T_x = 0$	Recirculate "B"
	2	$S_x T_x = 1$	Shift sign into F_1
	3	$T_x = 0$	Copy sign into NEON
1	4	$S_x' T_x = 1$	Shift 4 bits into F's
	5	$T_x = 0$	Copy character into NIXIE
8	18	$S_x' T_x = 1$	Shift last B.C.D. character (of "m") into F's while "B" accepts "R's" "m + 1" information via B_{42}
	19	$T_x = 0$	Copy last character (of "m") into NIXIE
9	20	$S_x' T_x = 1$	Shift first B.C.D. character (of "m + 1") into F's. (sign was lost in B_{42})
	33	$T_x = 0$	Copy last B.C.D. character into NIXIE
16	34	$S_x' T_x = 0$	Shift garbage into F's

The timing chart on the following page shows the interrelation of R_{c1}, R_{c2}, R_{c3}, S_x, T_x, N_8, N_{12} during I_3 for programed B.C.D.

PUNCH AND TYPEWRITER OUTPUT

Punch Character (PNC), Type Character (TYC), Punch and Type Character (PTC) Commands

General

The one-character output commands all are similar in operation, therefore, they will be discussed as a unit.

"PNC" (D_6 D_5 D_4 D_3 D_2' D_1'). A single character is punched in a 5- bit teletype code corresponding to the least significant five address bits (not counting 1/2 word indicator).

"TYC" (D_6 D_5 D_4 D_3' D_2 D_1'). The "Type Character" command substitutes typing for punching. The actual typed character depends on whether "LTRS." shift or "FIGS." shift has been previously called for.

"PTC" (D_6 D_5 D_4 D_3 D_2 D_1'). The "Punch and Type Character" command causes both punching and typing to occur.

"PNC" Operation

Setting M_7. After I_1, the command pair is in the Z-register and during I_2 the operation code is placed in the operation code register (D_6 D_5 D_4 D_3 D_2 D_1).

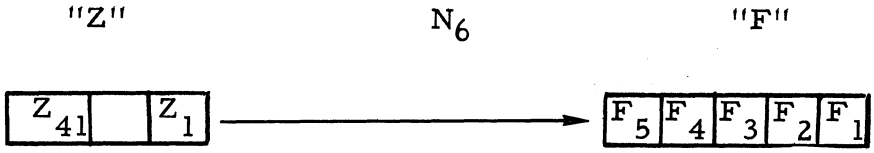
M_7 is then set by:

$$1^{m_7} = J_0' D_6 D_5 D_4 I_2 N_{11} C$$

M_7 is zero-set while the actual punching is being done.

$$0^{m_7} = TPP_p X_{10}' C$$

Z and F Configuration. The flow of the code bits from Z to F during word 1 of I_3, is shown on the following page:



The logic involved follows:

$1^{f_5} = Z_1 N_6 C$
$0^{f_5} = Z_1' N_6 C$

$1^{f_i} = F_{i+1} N_6 C$
$0^{f_i} = F_i' + 1 N_6 C$

Where i = 1-4

N_6 determines bit-times T_3 - T_7 (or T_{23} - T_{27}) which are the required code bit positions of the "PNC" command.

$1^{n_6} = M_7 D_6 D_1' U_1 N_9 C$
$0^{n_6} = P_2 P_1 C$

"Punch" and "Tape Punch Pulse" Flip-Flops. The punch flip-flop one-sets by:

$1^{x_9} = M_7 D_3 X_{10}' Q_5 (TPP)' T_{41} C$

Q_5 T_{41} determines a delay of 16 word-times in I_3.

The term (TPP)' inhibits X_9 while the Tape-Punch-Pulse flip-flop is one-set. TPP is one-set (for punching) by the closure of a contact (TPP_p) during the actual mechanical punching motion.

$1^{TPP} = (TPP_p) X_{10}' C$

X_{10}' is necessary for the "PTC" command.

TPP is zero-set at the termination of the punch motion by the TPP_p contact opening.

$$0TPP = (TPP_{rm})' (TPP_k)' (TPP_t)' (TPP_p)' (TPP_{rp})' [S_{u1}' S_{u2}' S_{u3}' S_c']$$

The other terms were already in their prime states. The definitions of these terms follow:

TPP_{rm}	Tape punch pulse for mechanical reader
TPP_k	Tape punch pulse for keyboard
TPP_t	Tape punch pulse for typewriter
TPP_p	Tape punch pulse for punch
TPP_{rp}	Tape punch pulse for photo-reader
S_{u1}	"Set up" button #1
S_{u2}	"Set up" button #2
S_{u3}	"Set up" button #3
S_c	"Start Compute" button

Delay Flip-Flops P_m and P_{d3}. Delay flip-flops will be as follows:

1. P_m: The P_m flip-flop resets the Q-counter by forcing a zero into Q_6 at T_2C.

$$0q_6 = P_m C$$

$$1p_m = (TPP) J_{13}' A_n' P_5' P_4' P_3' P_2' P_1$$

$P_5' P_4' P_3' P_2' P_1$ determines T_2

J_{13}' indicates no ECHO check error

TPP indicates punch is in motion

$$0p_m = P_{d3}C + J_{13}C$$

2. P_{d3}: The P_{d3} flip-flop indicates termination of the command and clears the output register (F's).

$1^{P}_{d3} = (TPP)' \; T_0' \; P_m \; T_{40} \; C$	Where $T_0' \triangleq$ Tape Reader Not on
$0^{P}_{d3} = P_{d3} \; T_{40} \; C$	
$0^{f}_{5} = P_{d3} \; T_{40} \; C$	
$0^{f}_{4} = D_5 \; P_0' \; P_{d3} \; T_{40} \; C$	Where $P_0' \; P_{d3}$ limits coverage to non-fill use of P_{d3}
$0^{f}_{3} = D_5 \; P_0' \; P_{d3} \; T_{40} \; C$	
$0^{f}_{2} = D_5 \; P_0' \; P_{d3} \; T_{40} \; C$	
$0^{f}_{1} = D_5 \; P_0' \; P_{d3} \; T_{40} \; C$	

3. ECHO Checking: The output error flip-flop (J_{13}) will one-set and cause its light to flash if the requested punching does not take place. ECHO Checking contacts are on each punch. The contact signals (E_p's) are compared with the original code in F by J_{13}.

$1^{j}_{13} = F_1 \; E_{p1}' \; T_p C + F_1' \; E_{p1} \; T_p C + F_2 \; E_{p2}' \; T_p C + F_2' \; E_{p2} \; T_p C + F_3 \; E_{p3}' \; T_p C$ $+ F_3' \; E_{p3} \; T_p C + F_4 \; E_{p4}' \; T_p C + F_4' \; E_{p4} \; T_p C + F_5 \; E_{p5}' \; T_p C + F_5' \; E_{p5} \; T_p C$
$0^{j}_{13} = T_d \; C$

"TYC" Operation

Setting M_7. The same one-set logic is used as for "PNC". See "PNC" operation.

The zero-set logic is:

$0^{m}_{7} = T_{bs} \; J_{13}' \; C$	where "T_t" is the type ECHO timing pulse

Z and F Configuration. See "PNC" operation.

"Type" and "Tape Punch Pulse" Flip-Flops. X_{10} indicates a typing operation. Its logic follows:

$$_1x_{10} = M_7\ D_2\ D_1'\ Q_5\ (TPP)'\ T_{41}\ C$$

$$_0x_{10} = (TPP_t)\ G+\ T_d\ C$$

As for X_9, the Q_5 term allows a delay (in I_3) of 16 word-times; the TPP_t signal is true during the actual typing motion. The logic for the TPP signal is:

$$_1TPP = TPP_t\ X_9'\ I_4'\ C$$

$_0$TPP = same as for punching. See "PNC" operation.

Delay flip-flops P_m and P_{d3}. See "PNC" operation.

Echo Checking. The error checking logic for typing is:

$$_1j_{13} = F_1\ E_{t1}'\ T_tC+\ F_1'\ K_{1k}\ T_tC+F_2\ E_{t2}'\ T_tC+F_2'\ K_{2k}\ T_tC+F_3\ E_{t3}'\ T_t\ C$$

$$+F_3'\ K_{3k}\ T_tC+F_4\ E_{t4}'\ T_tC+F_4'\ K_{4k}\ T_tC+F_5\ E_{t5}'\ T_tC+F_5'\ K_{5k}\ T_t\ C$$

The E_t's are the typing ECHO checking normally closed contacts. The typing ECHO checking normally open contacts (K_k's) receive their nomenclature from their use during keyboard or typewriter fill state. T_t is the type ECHO timing pulse. The zero-set logic for J_{13} is:

$$_0j_{13} = T_dC+\ S_{u1}C$$

"PTC" Operation

Setting M_7. The one-set logic for M_7 is identical to that of "TYC" or "PNC." See "PNC" operation.

The zero-set logic for M_7 is:

$$_0m_7 = T_{bs}\ J_{13}'C \pm TPP_pX_{10}'C$$

Z and F Configuration. See "PNC" operation.

"Punch," "Type," and "Tape Punch Pulse" Flip-Flops. For the "PTC" command TPP is controlled by the slower of the two output devices. The sequence, if the punch is faster than the typewriter, is:

1. X_9 and X_{10} are one-set

$$\boxed{{}_1X_9 = M_7\, D_3\, X_{10}'\, Q_5\, (TPP)'\, T_{41}\, C}$$

$$\boxed{{}_1X_{10} = M_7\, D_2\, D_1'\, Q_5\, (TPP)'\, T_{41}\, C}$$

2. TPP_p goes true

3. X_9 zero-sets

$$\boxed{{}_0X_9 = (TPP_p)\, C}$$

4. TPP_t goes true

5. X_{10} zero-sets

$$\boxed{{}_0X_{10} = (TPP_t)\, C}$$

6. TPP flip-flop one-sets

$$\boxed{{}_1TPP = (TPP_t)\, X_9'\, I_4'\, C}$$

7. TPP_p goes false

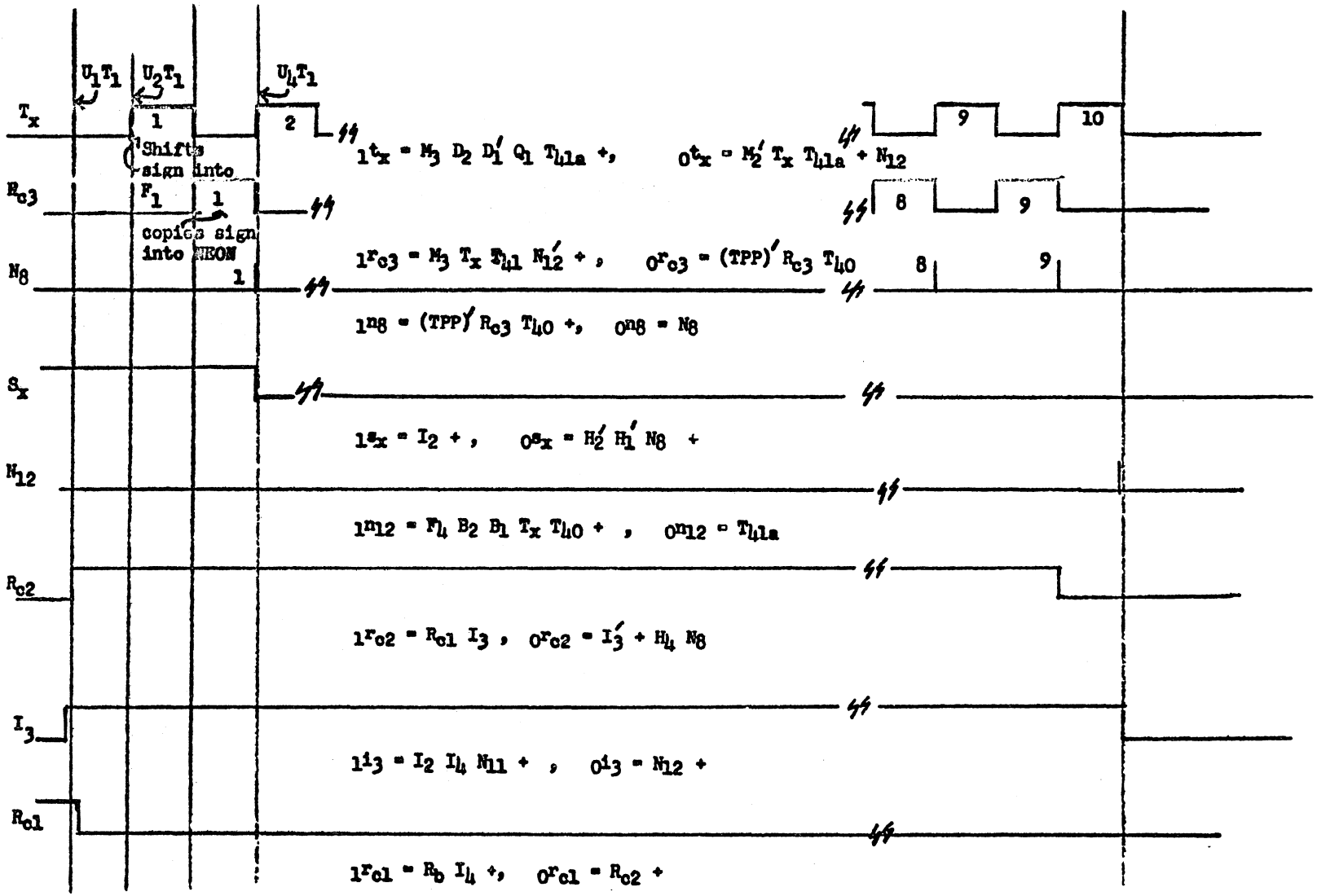
8. TPP_t goes false

9. TPP zero-sets

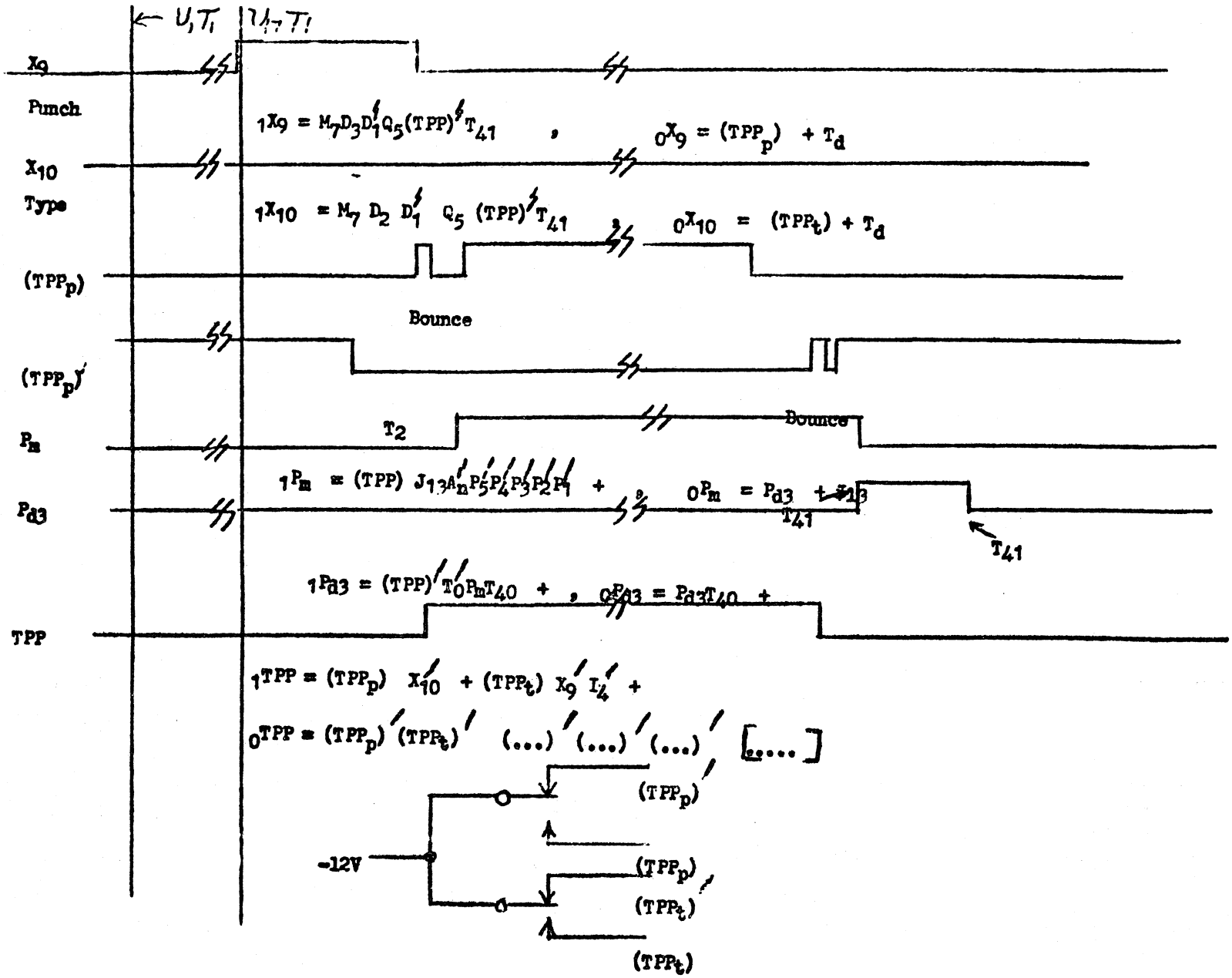
$${}_0TPP = (TPP_p)'\, (TPP_t)' \ldots\ldots$$

Delay Flip-Flops, P_m and P_{d3}. See "PNC" operation.

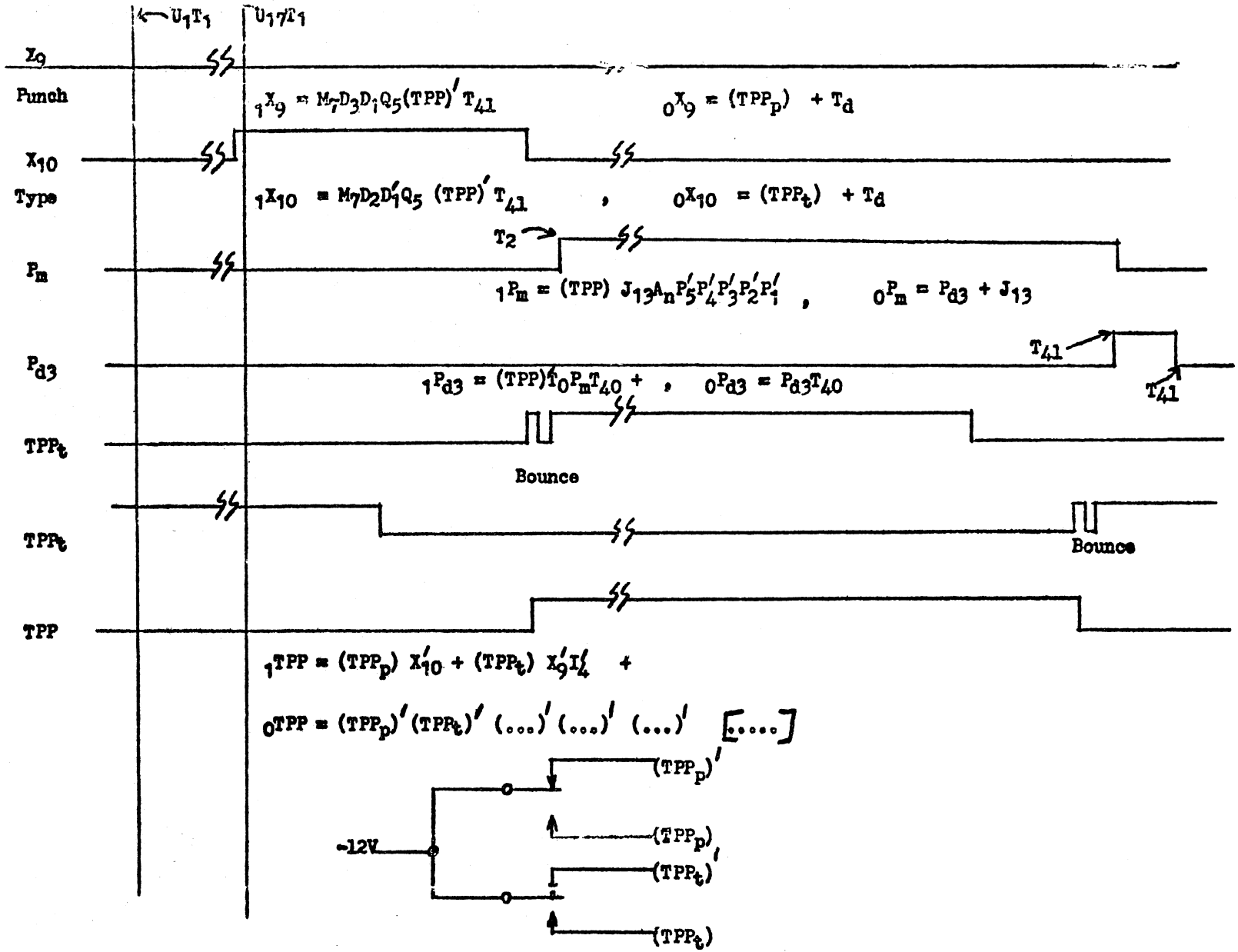
Echo Checking. Errors in either typing or punching will cause the output error light to flash and the computer to stop.



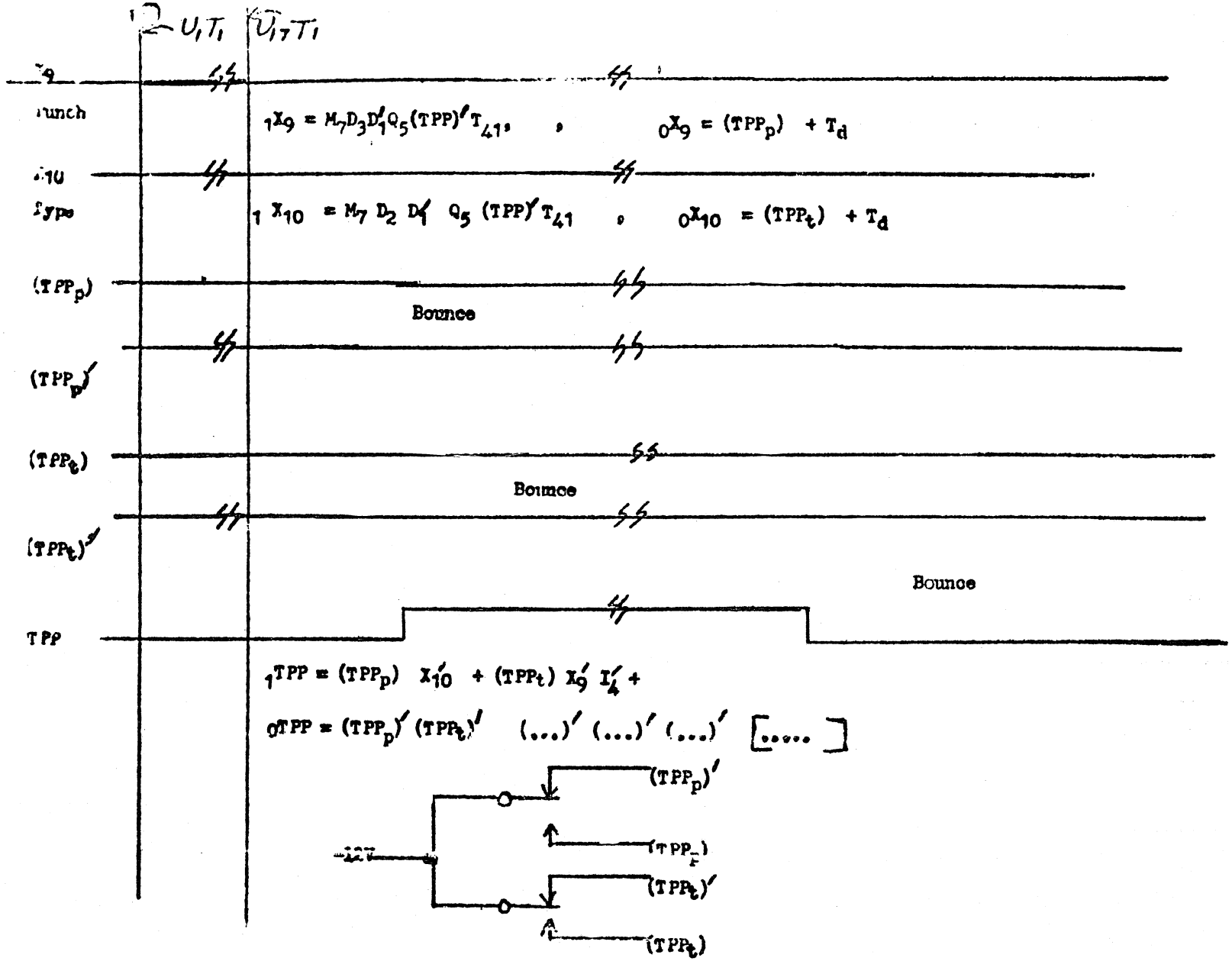
$I_3\ I_4$ State For Manual, BCD Format, Readout



Timing of "PNC" Command



Timing of "TYC" Command



Timing of "TYC" Command

PUNCH WORD (PNW), TYPE WORD (TYW), AND PUNCH AND TYPE WORD (PTW) COMMANDS

General

In order to provide compatibility between the 8, 4, 2, 1 binary internal code and the teletype code used to drive the punch and typewriter, the output register must have binary-to-teletype conversion logic.

In other respects, the PNW, TYW, and PTW commands are similar to the Display Command ("DIS"). An "ECHO" check is made on each character.

Punch Word, "PNW" (D_6' D_5' D_4 D_3 D_2' D_1'). The punch word command, "PNW (m)", punches the requested information in the format specified by the half-word indicator, 1 for BCD, 0 for Command.

1. Command Format. Sixteen 5-bit characters are punched to represent the usual command format (sign, 6-octal, 1-binary, 1-binary, 6-octal, 1-binary).

2. Decimal (BCD) Format. The partial contents of memory positions, "m" and "m + 1" are punched out in BCD if the command half-word indicator is "1."

Up to sixteen 5-bit characters may be punched; 1 binary, and 15 BCD. The information from "m" and "m + 1 is apportioned in the same manner as for the "DIS" command (a leading "+" sign will not be punched; a leading "-" sign will be punched).

Type Word, "TYW" (D_6' D_5' D_4 D_3' D_2 D_1')

The same formats as above are available for typewriting.

Punch and Type Word, "PTW" (D_6' D_5' D_4 D_3 D_2 D_1')

The same formats as above are available for punching and typing

"PNW" Command

I_1 State. As for any command, the command pair is gated into the Z register during I_1.

I_2 State. The "PNW" command is an I_2, Type 3. Therefore, the address "m" of the command is located during I_2 and the contents of "m" are gated into the B-register. The operation code bits are also placed into the operation code register (D_{6-1}) during I_2.

I_3 State. During I_3 U_1 (for BCD) the contents of "m + 1" are gated into R. Also during I_3, the information to be punched out is shifted, character by character into the F-register (very much the same as for the "DIS" command). However, a binary-to-teletype code conversion must be accomplished on each character (after it is in the F-register) before the punch may be allowed to monitor F. This added step was not required for the Display command.

1. I_3 and N_{12} Flip-Flops. I_3 is turned one-set by the usual logic:

$$1^{i}_{3} = J_0' I_2 N_{11} C$$

After the command is completed, I_3 is zero-set by:

$$0^{i}_{3} = N_{12} C$$

...where N_{12}'s logic is:

$$1^{n}_{12} = \underbrace{T_x R_{c2}' H_4' T_{40} D_1'}_{\text{Command Format}} C + \underbrace{F_4 B_2 B_1 T_x T_{40}}_{\text{BCD Format}} C$$

$$0^{n}_{12} = T_{41a}$$

2. M_7 Flip-Flop. In order to punch, type, or punch and type a word, the basic character operation is repeated a number of times dependent on the format.

M_7 is cycled for each character as follows:

$$1^{m}_{7} = \underbrace{J_0' D_6' D_5' D_4 D_1' I_2 N_{11}}_{\substack{\text{Initial} \\ \text{One-Setting}}} C + \underbrace{D_5' T_x T_{41a} N_{12}' C}_{\substack{\text{Subsequent} \\ \text{One-Settings}}}$$

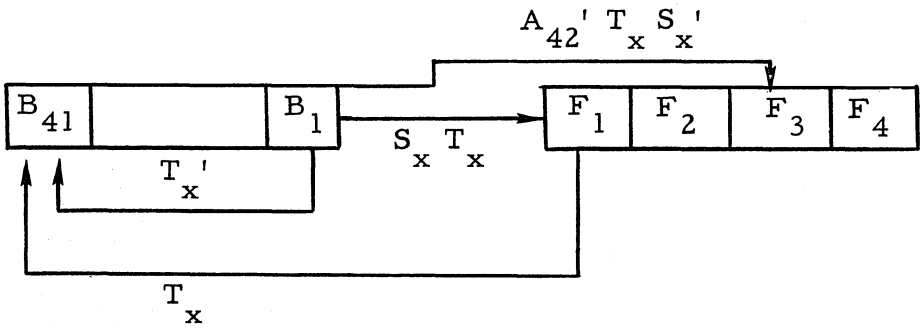
After the punching, typing, or punching and typing actually begins, the M_7 flip-flop zero-sets.

$$0^{m}_{7} = \underbrace{T_{bs} J_{13}' C}_{\text{"TYW" or "PTW"}} + \underbrace{TPP_{p} X_{10}'}_{\text{"PNW" or "PTW"}} +$$

3. A-Register. During I_3 of the PNW Command, the A-register recirculates.

$1^{a}_{41} = A_1 M_0' M_1' M_2' M_3' M_4' M_{10}' P_0' C$
$0^{a}_{41} = A_1' M_0' M_1' M_2' M_3' M_4' M_{10}' P_0' C$

4. Shifting B into F. Command Format is as follows:



R recirculates by:

$1^{r}_{41} = R_1 M_0' M_1' M_2' M_3' M_4' M_9' I_3 C$
$0^{r}_{41} = R_1' M_0' M_1' M_2' M_3' M_4' M_9' I_3 C$

B_{41}'s logic is:

$1^{b}_{41} = B_1 T_x' M_0' M_8' M_{10}' I_3 C+ F_1 R_{c2}' T_x C+ F_1 H_4' T_x C$
$0^{b}_{41} = B_1' T_x' M_0' M_8' M_{10}' I_3 C+ F_1' R_{c2}' T_x C+ F_1' H_4' T_x C$

The F-Register logic is:

$_1f_1 = F_2 S_x' T_x N_{12}'C+ B_1 S_x T_x N_{12}' C$
$_0f_1 = F_2' S_x' T_x C+ B_1' S_x T_x C+ D_5' P_{d3} T_{41} C$

$_1f_2 = F_3 E_{30}' T_x N_{12}' C$
$_0f_2 = F_3' T_x C+ D_5' P_{d3} T_{41} C$

$_1f_3 = B_1 A_{42}' S_x' T_x M_2' N_{12}' C$
$_0f_3 = B_1' A_{42}' T_x M_2'C+ D_5' P_{d3} T_{41} C$

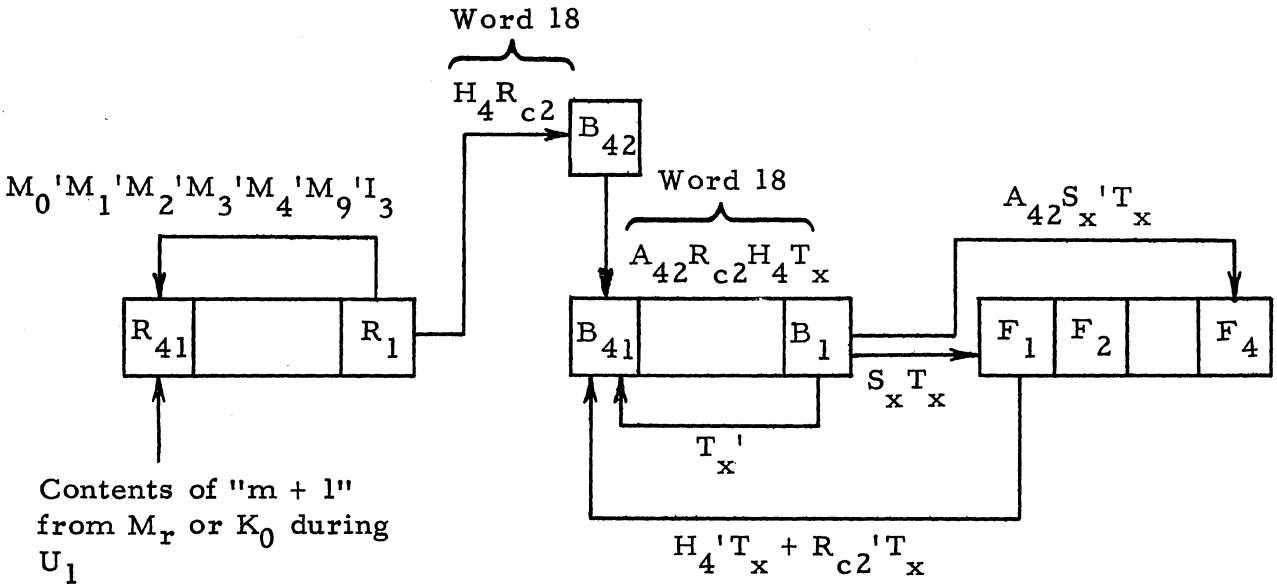
$_1f_4$ = Not involved in shifting
$_0f_4$ = Not involved in shifting

A_{42}'s logic is:

$_1a_{42}$ = Won't one-set for "command" format
$_0a_{42} = H_0' I_2 T_1 C$

R_{c2}'s logic is:

$_1R_{c2} = M_7 D_6' U_1 C$
$_0R_{c2} = A_{42}' H_3 H_2 H_1 N_8 C+ I_3'C$



The logic involved is:

$$1^r_{41} = M_r\, L_{p1}'\, D_6'\, D_4\, I_3\, I_4'\, D_1'\, U_1 C + K_0\, L_{p1}\, D_6'\, D_4\, I_3\, I_4'\, D_1'\, U_1 C$$

$$+ R_1\, M_0'\, M_1'\, M_2'\, M_3'\, M_4'\, M_9'\, I_3\, C$$

$$ {}_{0}r_{41} = M_r' L_{p1}' D_6' D_4 I_3 I_4' D_1' U_1 C + K_0' L_{p1} D_6' D_4 I_3 I_4' D_1' U_1 C + R_1' M_0' M_1' M_2' M_3' M_4' M_9' I_3 C $$

$$ {}_{1}b_{42} = R_1 H_4 R_{c2} N_{12}' C $$

$$ {}_{0}b_{42} = R_1' I_4' R_{c2} C $$

$$ {}_{1}b_{41} = B_1 T_x' M_0' M_8' M_{10}' I_3 C + F_1 H_4' T_x C + F_1 R_{c2}' T_x C + B_{42} A_{42} R_{c2} H_4 T_x C $$

$$ {}_{0}b_{41} = B_1' T_x' M_0' M_8' M_{10}' I_3 C + F_1' H_4' T_x C + F_1' R_{c2}' T_x C + B_{42}' A_{42} R_{c2} H_4 T_x C $$

$$ {}_{1}f_4 = B_1 A_{42} M_2' N_{12}' S_x' T_x C $$

$$ {}_{0}f_4 = B_1' A_{42} T_x C $$

$$ {}_{1}f_1 = B_1 S_x T_x N_{12}' C + F_2 S_x' T_x N_{12}' C $$

$$ {}_{0}f_1 = B_1' S_x T_x C + F_2' S_x' T_x C $$

$$ {}_{1}R_{c2} = M_7 D_6' U_1 C $$

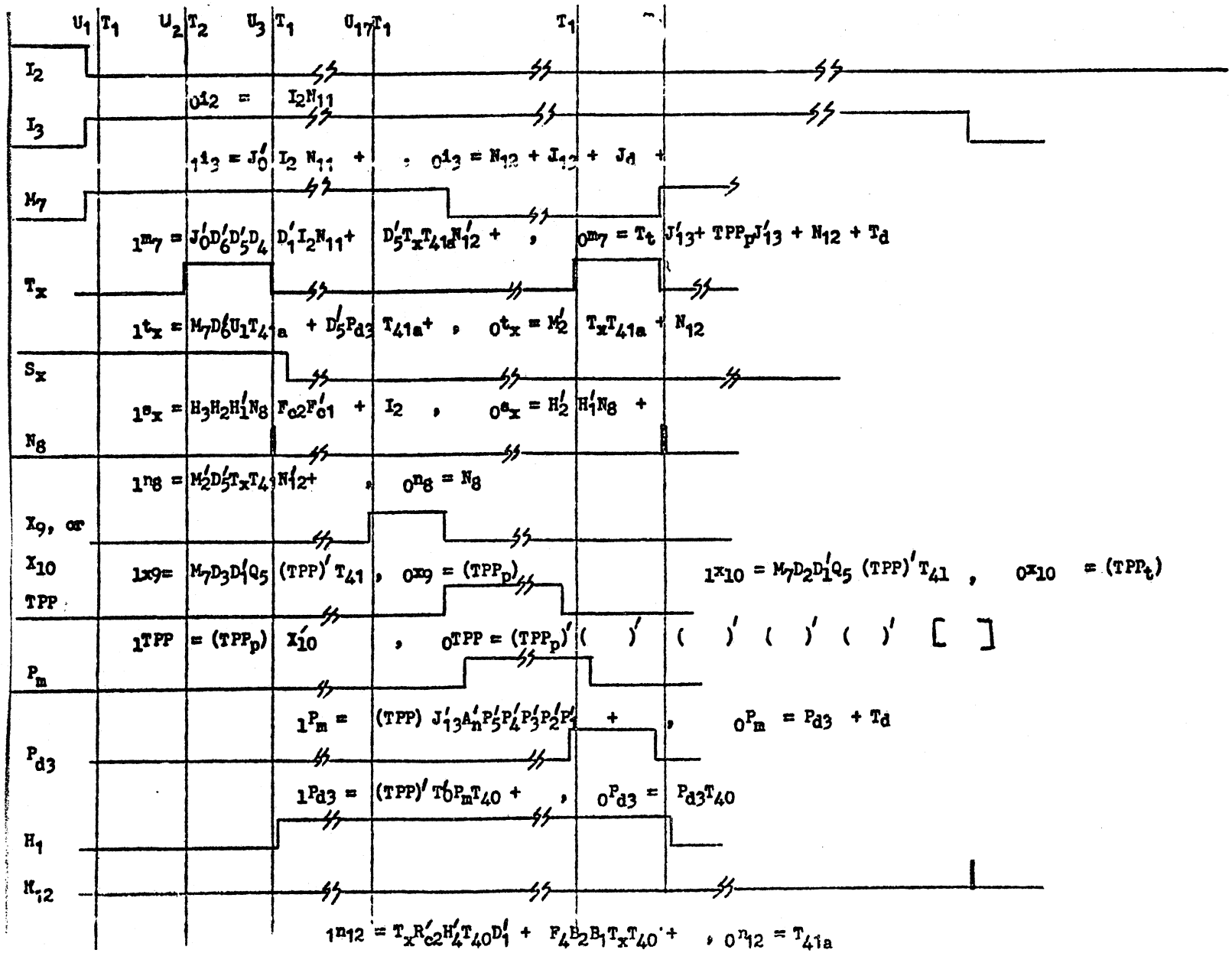
$$ {}_{0}R_{c2} = H_4 N_8 C + I_3' C $$

5. Timing Logic. The following are timing logic operations:

a. H-Counter: The H-Counter is initially cleared by:

$$ {}_{0}h_i = U_1 C_r' A_0' + \qquad i = 1\text{-}4 $$

The H-Counter here will keep track of the number of times T_x becomes one-set (via N_8).



PNW, or TYW Commands (See "Display Command" passout for format timing)

b. S_x: S_x is one-set by:

$$ {}_1S_x = I_2 C+ \underbrace{H_3 H_2 H_1' N_8 F_{c2} F_{c1}}\ C $$

Command Format

S_x will be one-set during the times that binary characters are being shifted into the F-register.

S_x is Zero-set by:

$$ {}_0S_x = H_2' H_1' N_8 C $$

c. T_x: T_x is one-set by:

$$ {}_1t_x = \underbrace{M_7 D_6' U_1 T_{41a}} C+ \underbrace{D_5' P_{d3} T_{41a}} C $$

Initially Subsequent Characters

...and zero-set after 1 word-time by:

$$ {}_0t_x = M_2' T_x T_{41a} C $$

6. Sign Conversion. The following are sign conversion operations:

Command Format. In command format the sign (either "+" or "-") is a legitimate output character. Before conversion, the F-register will contain one of the following:

F_1	F_2	F_3	F_4	F_5	
1	0	0	0	0	for "+"
0	0	0	0	0	for "-"

The conversion should derive one of the following:

F_1	F_2	F_3	F_4	F_5	
1	0	0	0	1	for "+"
1	1	0	0	0	for "-"

For the "+" conversion, only the F_5 flip-flop must contain added logic. The following logic of F_5 covers the sign position.

$$ {}_1f_5 = F_2' F_1 N_8 C $$

N_8 here is a conversion timing flip-flop:

$$ {}_1n_8 = M_2' D_5' T_x T_{41} N_{12}' C $$

$$ {}_0n_8 = N_8 C $$

For the "-" conversion, the bits in F_2 and F_1 require modification.

$$ {}_1f_2 = F_3' F_2' F_1' D_5' N_8 C + F_4' F_2' F_1' D_5' N_8 C $$

$$ {}_1f_1 = D_5' S_x H_1' N_8 C $$

For BCD Format. In BCD, the sign is legitimate only if it is "-". "+" signs are not punched out. This allows the programer the ability to program a BCD output longer than 15 characters without the usually meaningless inclusion of sign characters at the start of each word of output.

The "+" sign is used to generate a "FIGS" shift code and, therefore, negates the need of a separate "FIGS" shift before the "+PNW" command.

Before conversion, the "F" register contains one of the following:

F_1	F_2	F_3	F_4	F_5	
1	0	0	0	0	for "+"
0	0	0	0	0	for "-"

The conversion should derive one of the following:

F_1	F_2	F_3	F_4	F_5	
1	1	0	1	1	for "+"
1	1	0	0	0	for "-"

The logic required for the "+" to "FIGS" conversion is:

$$_1f_4 = \underbrace{A_{42}}_{\text{BCD}}\ \underbrace{S_x}_{\text{Sign Position is +}}\ F_1\ \underbrace{N_{8a}}_{\text{Same as } N_8}\ C$$

$$_1f_2 = A_{42}\ S_x\ F_1\ N_{8a}\ C$$

$$_1f_1 = D_5'\ S_x\ H_1'\ N_8\ C$$

Same gate as for "command"

The "-" conversion is identical to that for "command" format:

$$_1f_2 = F_3'\ F_2'\ F_1'\ D_5'\ N_8 + F_4'\ F_2'\ F_1'\ D_5'\ N_8\ C$$

$$_1f_1 = D_5'\ S_x\ H_1'\ N_8\ C$$

"PNW", "TYW", and "PTW" Output Conversion

Binary-to-Teletype

Character (In "FIGS" Shift)	Binary (Before Conversion) F_5	F_4	F_3	F_2	F_1	Teletype (After Conversion) F_5	F_4	F_3	F_2	F_1	
∅	X	0	0	0	0	1	0	1	1	0	
1	X	0	0	0	1	1	0	1	1	1	
2	X	0	0	1	0	1	0	0	1	1	
3	X	0	0	1	1	0	0	0	0	1	
4	X	0	1	0	0	0	1	0	1	0	
5	X	0	1	0	1	1	0	0	0	0	
6	X	0	1	1	0	1	0	1	0	1	
7	X	0	1	1	1	0	0	1	1	1	
8	X	1	0	0	0	0	0	1	1	0	
9	X	1	0	0	1	1	1	0	0	0	
"."	X	1	1	0	0	1	1	1	0	0	
Carriage Return	X	1	0	1	1	0	1	0	0	0	
						*1	1	0	1	1	(BCD)
"+"	X	0	0	0	1	**1	0	0	0	1	(Command)
"-"	X	0	0	0	0	0	0	0	1	1	
"Space"	X	1	0	1	0	0	0	1	0	0	
No Output	X	1	1	1	1	Causes Termination of I_3					

X Will always be "0" initially and will not be affected by the shifting operations of "F"

* Causes "FIGS" shift

** Causes "+" shift

7. More Code Conversion. Code conversion examples follow:

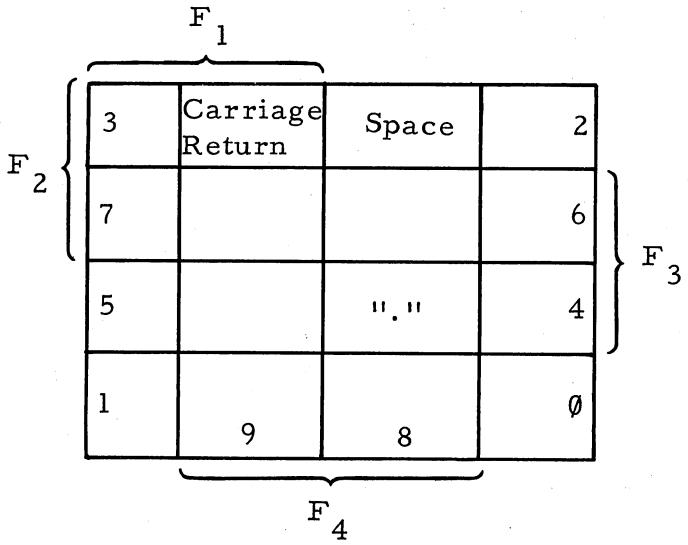
The Problem

Of course the sign is not the only character that requires conversion for the "PNW", "TYW," and "PTW" commands.

Each character on the figure on the preceding page needs conversion. To derive the required logic, recourse to Veitch diagram techniques will be made.

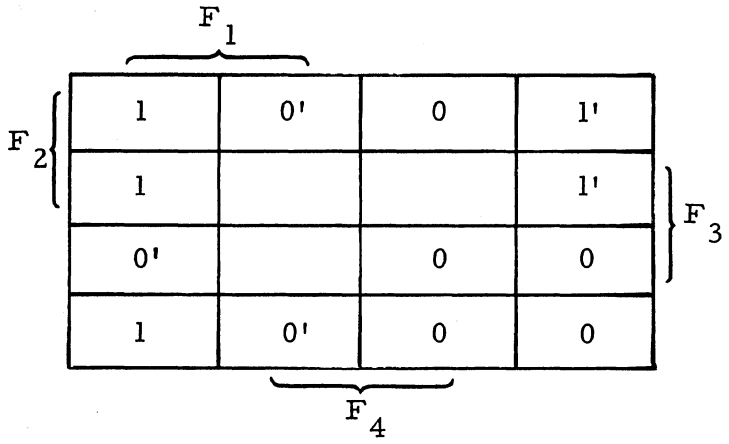
F_1 Logic

First a four-variable Veitch diagram is constructed locating each character by its binary code:



Secondly, another four-variable Veitch is constructed using the following rules of procedure:

1. Place a 1' for each character (as defined on the last Veitch diagram) that requires F_1 to be one-set when going from binary to teletype coding.
2. Place a "1" for each character that may be one-set in going from binary to teletype coding.
3. Place a 0' for each character that requires F_1 to be zero-set.
4. Place a "0" for each character that F_1 may be zero-set.



The total conversion logic is shown below with the logic covering the primed 1's and 0's underlined.

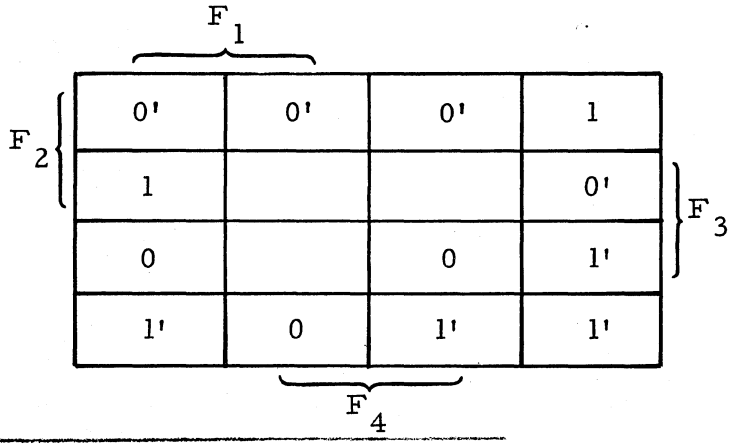
$$ {}_1f_1 = \underline{F'_4 F_2 N_8 D'_5 C}^{*} + \underbrace{\underline{D'_5 H'_1 S_x N_8}\ C}_{\text{for sign conversion}} $$

The zero-set logic is:

$$ {}_0f_1 = \underline{F_3 F'_2 N_{8a} C} + \underline{F_4 N_{8a} C} + \underbrace{D'_5 P_{d3} T_{41} C + N_{12} C}_{\text{Clearing}} $$

F_2 Logic

The same process, as in "b", is used for F_2 logic:



*N_8 (and N_{8a}) here is a conversion timing flip-flop for outputs

$$1f_2 = \underbrace{F'_4F'_3F'_2N_8D'_5H_1C + F'_4F'_3F'_2N_8D'_5S'_xC}_{(H'_1S_x)' \text{ excludes sign positions}}$$

$$+ F'_3F'_2F'_1N_8D'_5C + F'_4F'_2F'_1N_8D'_5C$$

$$0f_2 = F'_3F_2F_1N_{8a}C + F_3F_2F'_1N_{8a}C + F_4F_2N_{8a}C$$

$$+ \underbrace{D'_5P_{d3}T_{41}C + N_{12}C}_{\text{Clearing}}$$

F_3 Logic

F_2 \ F_1	0	0	1′	0
	1			1
	0′		1	0′
	1	0	1′	1′

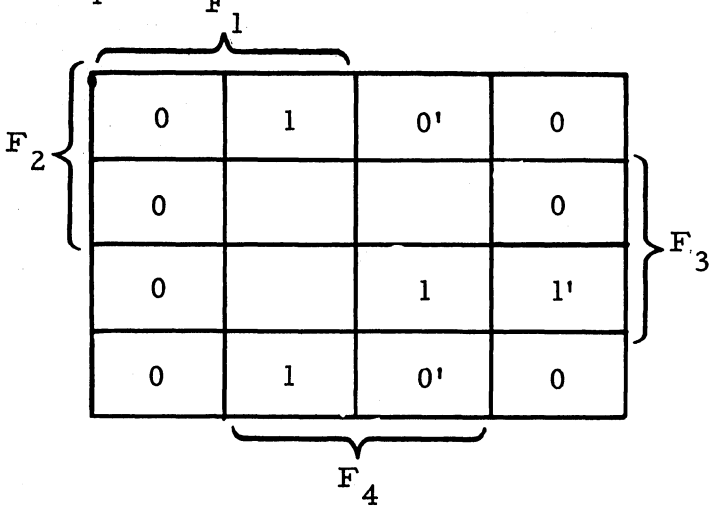
F_3 (1, 1, 0′, 1′)

F_4 (0, 0, 1′, 0)

$$1f_3 = \underbrace{F'_4F'_3F'_2N_8D'_5H_1C + F'_4F'_3F'_2N_8D'_5S'_xC}_{(H'_1S_x)' \text{ excludes sign positions}} + F_4F'_1N_8D'_5C$$

$$0f_3 = F'_4F_3F'_2N_{8a}C + \underbrace{D'_5P_{d3}T_{41}C}_{\text{Clear}}$$

F_4 Logic



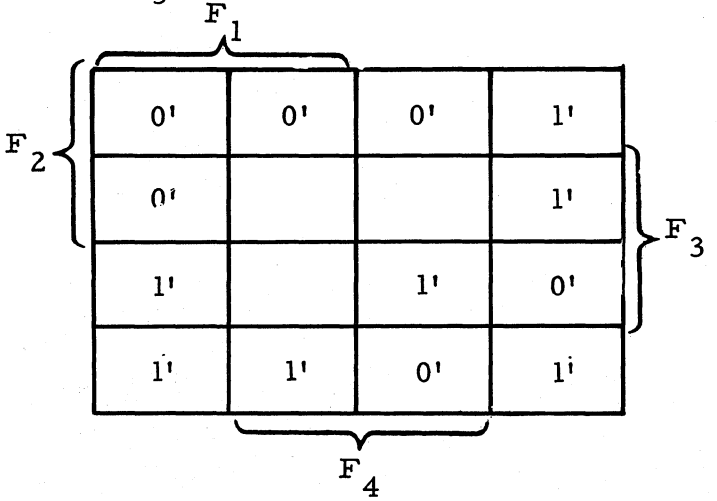
$$1^{f}_{4} = F_3 F'_2 F'_1 N_8 D'_5 C + A_{42} S_x F_1 N_{8a} C$$

$$0^{f}_{4} = F'_3 F'_1 N_{8a} C + D'_5 P_{d3} T_{41} C + N_{12} C$$

Clearing

F_5 Logic

The F_5 flip-flop is, for the "PNW", "TYW", and "PTW" commands, in its zero-set state at the beginning of I_3 and remains so (untouched by octal, binary, or B. C. D. shifts from "B" to "F") before conversion. During binary-to-teletype conversion, F_5 may be set to 1 dependent on the character. Therefore, the following Veitch diagram assumes that F_5 is "0" at the start of the conversion.



$$ {}_1f_5 = \underbrace{F'_4F'_3F'_2N_8H_1C + F'_4F'_3F'_2N_8S'_xC}_{(H'_1S_x)' \text{ Excludes signs}} $$

$$ + F'_4F_2F'_1N_8C + F'_2F_1N_8C + F_4F_3N_8C $$

$$ {}_0f_5 = \underbrace{P_{d3}T_{40}C + D'_5P_{d3}T_{41}C + N_{12}C}_{\text{Clearing}} $$

SYSTEM CHECKOUT PROCEDURES

GENERAL

Many procedures can be devised for checking or troubleshooting a computing system. But selecting the appropriate procedure can save much time in diagnosing a malfunction and isolating it to the faulty component.

The procedures given in this section, when used wherever possible inconjunction with test routines, will enable detection and isolation of most system malfunctions relatively quickly and easily. Procedures for correcting the malfunctions, if not presented in this section, are given in the Computer Service Manual and Input-Output Service Manual.

COMPUTER

Careful anaysis of the indications of trouble will save time in isolating a computer malfunction, particularly when more than one unit or function is affected. At times an error in the program or routines may be causing the difficulty, though the computer may appear to be at fault. In addition to thorough analysis of the indications of trouble, the proper troubleshooting sequence may also save considerable time. Generally, however, the first steps will include checking the voltage levels and the controls, and running appropriate test routines.

Procedures for checking the voltage levels are given in the Computer Service Manual, for initially testing operation of the controls and running test routines in this and section , respectively, of this manual. These procedures should at least isolate the malfunction to a particular unit.

If the malfunction is in the computer or circuit portions of the other units, most rapid isolation can usually be achieved by using the system tester. This equipment enables flip-flop checks, static and dynamic checks of logic, set-reset control of flip-flops, and application of marginal conditions to voltage levels and clock.

Before making any circuit tests, however, the person performing them should familiarize himself with the circuit characteristics and functions by studying the logic data-flow diagrams and accompanying logic charts in this manual, and the circuit schematic and wire listings

in the System Reference Schematics. For example, if the computer fails to execute a command, reference to the diagrams and charts mentioned will aid in isolating the difficulty by enabling testing of the circuitry used by that command. When making tests, the following precautions should be observed:

CAUTION

Boards containing transistors should never be pulled out while power is on. (Logic boards may be removed with power on.) Power to the computer should never be removed except with the switch at the control console. Metal objects, such as a screwdriver, should never be used on boards or wiring while power is on.

The computer should never be moved while the memory disk is rotating; allow at least 15 minutes from power turnoff for complete memory deceleration before moving computer.

The following paragraphs outline checkout and troubleshooting procedures on the computer, some using the system tester as indicated (a functional description of the system tester and procedures for operating it are presented in the Test Equipment Service Manual).

Power Turn-On Check

1. Connect variac into outlet and adjust its output to 115 volts as monitored on voltmeter.
2. Connect computer power cable (attached to desk) to variac.
3. Depress Power On-Off switch on control console to On position and note the time elapsed until the Ready indicator lights (Ready indicator should light in 40 ± 10 seconds; Power indicator should light immediately upon depressing Power button).
4. Recheck line voltage with voltmeter to reassure that line voltage is 115 volts.

Power Supply Checks

The following procedures ascertain if power supply voltages are correct and tolerances are within percentage limits.

1. Attach digital voltmeter and note power supply voltages are monitored at System Tester test points (see table following for normal power supply levels and Computer Service Manual for adjustment procedures).

2. Check voltage percent levels at power supply voltmeter on rear of control console by turning voltmeter range switch to each of the positions and observing the reading at each position. Levels must read 100 ± 5 percent and the -100 voltage which should read within ±10 percent; percent voltage levels may be adjusted at the appropriate potentiometer settings on the control panel network board as outlined below:

Table 9. Power Supply Voltages

Normal Power Supply And Ripple Values

Level	Value	Adjustable	Tolerance (plus/minus)	Ripple p-p (less than)
-18	-18	yes	2.5%	90 mv
-12	-12	yes	2.5%	60 mv
-6	-6	yes	2.5%	90 mv
+6	+6	yes	2.5%	30 mv
+75	+75	yes	2.5%	400 mv
-3	-2.5	no	0.5V	
+.75	+.75	no	0.15V	
-3B	-2.5	no	0.5V	
+75R	+35	no	10.0V	
-100	-100	no	10.0V	3.5 V

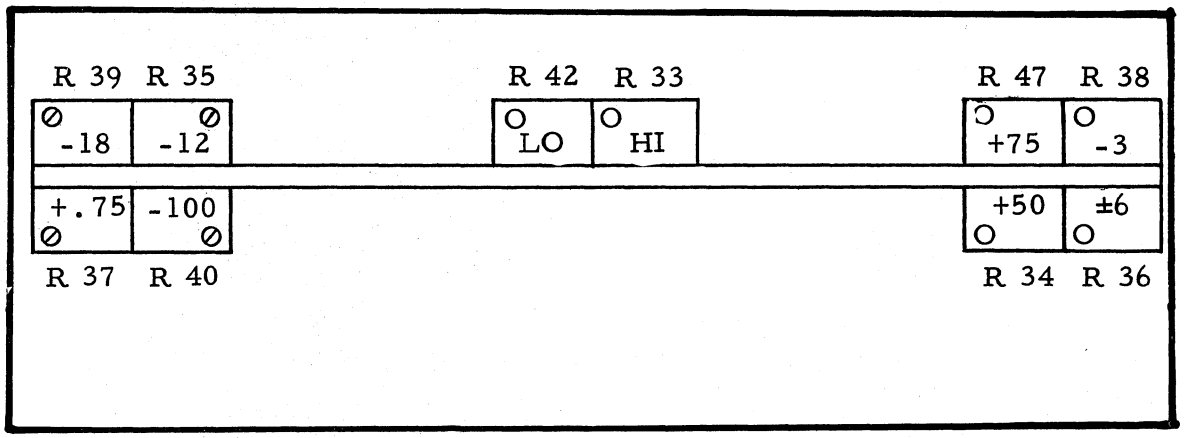


Figure 36. Control Panel Network Board

Location of Potentiometers On Control Panel Network Board

Adjustment	Voltage	Remarks
R 33	Line Trans	High Limit
R 34	+50	Adjusted before adjustment of R 47
R 35	-12	
R 36	±6	Adjust for mean value
R 37	+.75	
R 38	-3	
R 39	-18	
R 40	-100	Adjustment dependent on voltage level
R 42	Line Trans	Low Limit
R 47	+75	Adjusted after adjustment of R 34

3. Check power supply ripple values with oscilloscope at System Tester test points, with the System Tester Mode switch in static position (see Normal Power Supply Table 8 for proper values).

4. Adjust variac until system is supplied with 108 volts ac as monitored on voltmeter.

5. Turn control pots on control panel network board (R33 and R42) counterclockwise all the way (see figure 36 for location of pots).

6. Depress Error Reset button on control console (Line Transient indicator on control console should extinguish).

7. Slowly turn R42 pot adjustment clockwise until Line Transient indicator lights.

8. Check voltmeter on rear of control console (reading should be ±3 percent of values given above.

9. Adjust variac up to 125 volts ac as monitored on voltmeter.

10. Slowly turn R 33 pot adjustment clockwise until Line Transient indicator lights.

11. Check voltmeter on rear of control console (reading should be ±3 percent of values given in table 8.

12. Readjust variac to 115 volts ac.

Static Computer Flip-Flop Check

This procedure checks for faulty flip-flops and whether they will toggle individually. The procedure can be varied in troubleshooting to the extent that only those flip-flops associated with the suspected malfunction, as indicated by the logic, need be individually one-and zero-set.

1. If not done previously, check to see if voltage levels are correct.
2. Connect System Tester and turn Marginal Test switch to Off position and Mode switch to Static position.
3. If not already done, apply power to computer at control console.
4. Place One-Set–Zero-Set-All switch in One-Set position and observe if all neon indicators on System Tester light (if a neon does not light, assuring that the neon isn't malfunctioning, replace the flip-flop concerned and repeat this step; if the flip-flop is not at fault check the Set lines within the computer).
5. Depress the One-Shot Clock button and observe if all neons remain lighted (if a neon extinguishes replace the flip-flop concerned and repeat this step).
6. Place All switch in Zero-Set position and observe if all neon indicators extinguish (if a neon remains lighted, replace the flip-flop concerned and repeat this step; if the flip-flop is not at fault check the set lines within the computer).
7. Depress the One-Shot Clock button and observe if any neons light (if a neon lights, replace the flip-flop concerned).
8. Place Individual One-Set–Zero-Set switch to 1-Preset position.
9. Depress microswitch of each flip-flop (term) on System Tester, beginning with A_{42} and ending with Z_{ch} and observe if all flip-flops come true by noting if all neon indicators light (indicator should remain lighted); if a neon extinguishes it indicates one of three possible conditions:

   a. The flip-flop has 0-set itself, indicating a faulty board.
   b. Another flip-flop has triggered the flip-flop indicated by the neon that extinguished, in which case, if (a) is eliminated as the cause set all flip-flops to 0-set state, then 1-set the flip-flop concerned and again 1-set each flip-flop individually, meanwhile observing when the neon for the flip-flop concerned extinguishes. When this neon extinguishes check for the possibility of an improper interconnection between the flip-flops concerned.
   c. The neon has burned out.

(Procedures for removing circuit boards are given in the Computer Service Manual).

10. After all neons remain lighted from performing Steps 8 and 9, place Individual switch to 0-Preset position.

11. Depress microswitch of each flip-flop on System Tester, beginning with A_{42} and ending with Z_{ch}, and observe if all flip-flops go false by noting if its neon idicator extinguishes (indicator should remain off); if a neon lights, it indicates one of three possible conditions:

a. The flip-flop has 1-set itself, indicating a faulty board.
b. Another flip-flop has triggered the flip-flop that lighted, in which case, if (a) is eliminated as the cause set all flip-flops to 1-set state, then 0-set the flip-flop concerned and again 0-set each flip-flop individually, meanwhile observing when the neon for the flip-flop concerned lights. When this neon lights, check for the possibility of an improper interconnection between the flip-flops concerned.

(Procedures for removing circuit boards are given in the Computer Service Manual).

12. Upon completion of Static Computer Flip-Flop Check, turn off power to computer at control console.

One Shot of Logic Check

This check can be used to detect and correct major wiring errors and broken wires. The procedure is:

1. Insert specially wired logic board and attach meter.
2. Zero-set all flip-flops with System tester and set up logical proposition to be tested using System tester with S1 in static position (observe if gate goes true as last term in proposition is set).
3. Check if each term can cause output to go false if it is not as stated in proposition under test.
4. After all of proposition has been checked, ohm out wiring between gate and set inputs to the flip-flop.

Example: one-shot $1^{m}_{4} = J_0 'D_6 D_5 'D_4 'I_2 N_{11}$ 120-9

Zero-set all

Turn D_6 on, (true) J_0 is already off (false)
Turn I_2 on, $D_5 D_4$ are already off (false)
Turn N_{11} on, gate should go true.
Turn J_0 on, gate goes false.

Turn J_0 off, gate goes true.
Turn D_6 off, gate goes false.
Turn D_6 on, gate goes true.
Turn D_5 on, gate goes false.
Turn D_5 off, gate goes true.
Turn D_4 on, gate goes false.
Turn D_4 off, gate goes true.
Turn I_2 off, gate goes false.
Turn I_2 on, gate goes true.
Turn N_{11} off, gate goes false.
Turn N_{11} on, gate goes true.

Gate is now checked.

Manual Input-Output (Console Keyboard and Visual Readout) Check

Procedures for checking proper operation of the console keyboard and visual readout are given later in this section.

Clock Signal Check

CAUTION

The computer should not be moved while the memory disk is rotating; allow at least 15 minutes from power turnoff for complete memory deceleration before moving computer.

NOTE

See figure 40 for location of computer connectors.

To Check Clock Track Preamplifier Signal

1. Monitor preamplifier signal with oscilloscope at connector J525-11 of computer or through monitor box plugged into J525 (amplitude should be 4.5 ± 0.5 volts peak-to-peak).
2. If necessary, adjust at potentiometer on preamplifier plugged into connectors J439 and J539 of computer.

To Check Computer Clock Duty Cycle

1. Measure clock signal output with oscilloscope at J339-36 and 339-37 (final signal should be as shown in figure 37 or from computer

manual; necessary adjustment should be made according to procedures in the Computer Service Manual).

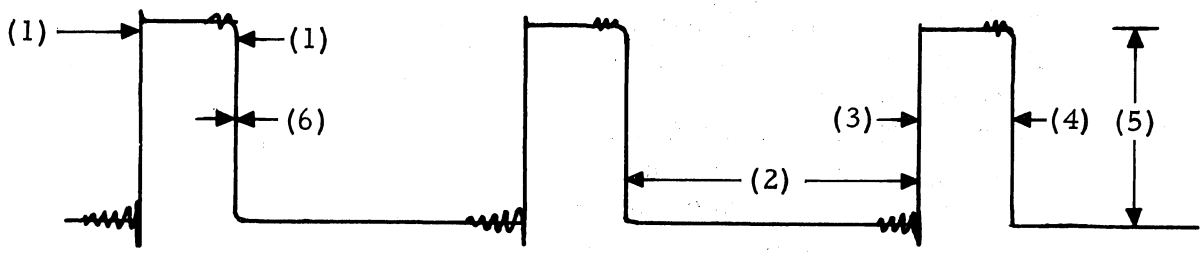


Figure 37. Clock Signal Waveform

(1) 1.75 ±0.15 usec (measured at 10% point of wave shape)

(2) 4.62 ±0.5 usec (measured at 10% point of wave shape)

(3) Rise Time: less than 0.5 usec

(4) Fall Time: less than 2 usec

(5) Amplitude: 0 ±0.5V to -6.5 ±0.5V

(6) Jitter: 0.5 usec

Origin Pulse Check

CAUTION

The computer should never be moved while the memory disk is rotating; allow at least 15 minutes from point of turnoff for complete memory deceleration before moving computer.

NOTE

See figure 40 for location of connectors.

To check origin pulse amplitude:

1. Monitor pulse with oscilloscope at connector J525-10 of computer or through monitor box plugged into J525 (origin pulse occurs once every disk revolution; amplitude should be 4.5 ±0.5 volts peak-to-peak).
2. If necessary, adjust potentiometer on preamplifier board plugged into connectors J438 and J538 of computer.

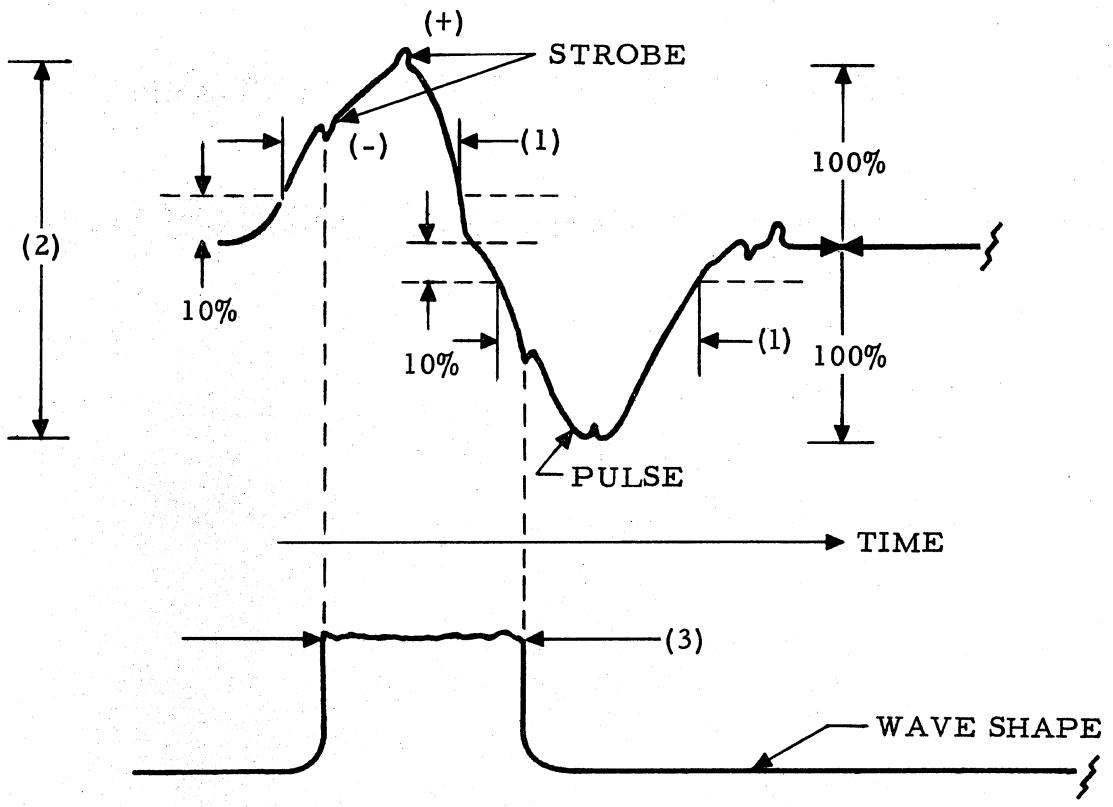
To check origin pulse shape:

1. With oscilloscope, check origin waveshape as monitored at System Tester X_o flip-flop output (origin waveshape is available once every disk revolution; wave shape should be as shown in figure 38).

2. If wave shape does not conform to standards given in figure below, memory should be returned to factory for adjustment.

To check origin pulse for noise and crosstalk:

Check for noise and crosstalk (incombination) on base line of origin track on oscilloscope (should be less than 500 millivolts).



(1) Pulse Width at 10 percent of magnitude must be less than 12 usec.
(2) Amplitude, excluding strobe, 4.5 ±0.5 volts.
(3) Pulse Width of Origin Wave shape: 6.37 usec ±0.65 usec (1 clock time measured at top of waveshape).

Figure 38. Origin Pulse and Waveshape

Sector Counter Signal Check

This procedure checks proper counting of the Sector Counter.

1. Connect System Tester.
2. Place Static-Dynamic switch in Static position.
3. Place One-Set—Zero-Set-All switch in 0-set position (check that all neon indicators are extinguished).
4. Set up any sector configuration (except 00_8 or 77_8) on flip-flops S_6 through S_0 by depressing individual microswitches to achieve binary indication of desired sector.
5. One-set G_0 and T_1 flip-flops (respective indicators should light).
6. Depress One-Shot Clock button (K_s and N_9 indicators should light).
7. Depress One-Shot Clock button (N_1 indicator should light).
8. Depress One-Shot Clock button six times (indicators S_6 through S_1 should light and the preceding one extinguish in succession).
9. Depress One-Shot Clock button (N_1 indicator should extinguish and sector counter should advance one to next sector from that set up in step 4).

Digit Counter Check

This procedure checks correct counting of the Digit Counter.

1. Connect System Tester.
2. Place Static-Dynamic switch in Static position.
3. Place One-Set—Zero-Set-All switch in 0-set position (check that all neon indicators are extinguished).
4. Depress One-Shot Clock button 41 times; indicators should light as follows: T_1 indicator lights on first depression of One-Shot Clock button; then digit (bit) counter indicators P_6 through P_1 should show binary representation of the decimal quantity signifying the number of times the One-Shot Clock button has been depressed through the 39th time; then T_{40} indicator lights on the 40th depression of the One-Shot Clock button; then both T_{41} indicators light on the 41st depression of the One-Shot Clock button.

Memory Signal Check

The following procedures enable checks of amplitude, pulsewidth, noise and crosstalk of all memory channels (additional procedures are given in the Computer Service Manual).

CAUTION

The computer should never be moved while the memory disk is rotating, allow at least 15 minutes from power turnoff for complete memory deceleration before moving computer.

NOTE

See figure 40 for location of connectors.

1. Set location counter to 0000.0.
2. Fill all of channel 00 with the pattern +2514340-7407601.
3. Connect test point at D_0 to ground at System Tester.
4. Select readout location 0077 with Channel and Sector Selector switches on control console.
5. Depress Command Readout button on control console.
6. Observe amplitude with oscilloscope at Connector J550-1 of computer (amplitude excluding strobe should be 4.5 ±0.5 volts average peak-to-peak).
7. If necessary, adjust amplitude at read switch plugged into J342 by adjusting the potentiometers (counting downward from the top beginning with 0).
8. Check pulse width at 10 percent of its magnitude (width should be less than 12 microseconds).
9. Disconnect ground from D_0 flip-flop and fill all of channel 00 with -000000000000000.
10. Reconnect D_0 to ground and select readout location 0077 as in Step 4.
11. Observe noise and crosstalk (in combination) with oscilloscope (should be less than 500 millivolts).
12. Disconnect ground from D_0.
13. Enter and verify Fast Memory Fill and One-Channel Zero routine with tape reader (see Test Routine section).
14. Place Sense Switch D up.
15. Depress Start 1 button (memory will load with +2514340-7407601 pattern; routine will halt automatically).
16. Reconnect D_0 to ground.
17. Select readout location 0177 with Channel and Sector Selector switches and depress Command Readout button.
18. Repeat steps 6 through 8.
19. Disconnect ground from D_0 and depress Start 2 button (computer is in a "copy from Y reader" state).
20. Depress numbers 01 on console keyboard and channel 01 will be filled with zero in all sectors through 0177.

21. After routine halts, reconnect D_o to ground and repeat step 11.

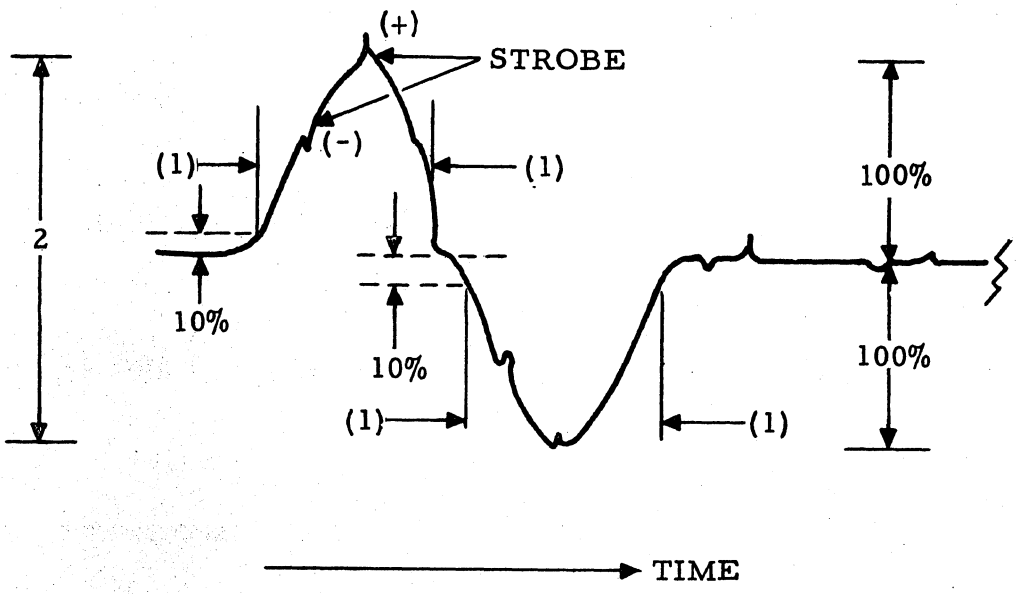
22. Repeat steps 12, 15, 16, 17, 18, 19, 20, and 21 except as follows:

When Testing Channel	Select Readout Location	And Depress Keyboard Numbers
02	0277	02
03	0377	03
04	0477	04
05	0577	05
06	0677	06
07	0777	07

23. To test remaining channels repeat steps 12, 15, 16, 17, 18, 19, 20, and 21 except as follows:

When Testing Channels	Observe Amplitude At	Adjust Amplitude At	Select Corresponding Readout Locations	And Depress Corresponding Keyboard Numbers
10 through 17	J550-2	J343	1077, 1177, etc through 1777	10 through 17
20 through 27	J550-3	J344	2077, 2177, etc through 2777	20 through 27
30 through 37	J550-4	J345	3077, 3177, etc through 3777	30 through 37
40 through 47	J550-5	J346	4077, 4177, etc through 4777	40 through 47
50 through 57	J550-6	J347	5077, 5177, etc through 5777	50 through 57
60 through 67	J550-7	J348	6077, 6177, etc through 6777	60 through 67

When Testing Channels	Observe Amplitude At	Adjust Amplitude At	Select Corresponding Readout Locations	And Depress Corresponding Keyboard Numbers
70 through 77	J550-8	J349	7077, 7177, etc through 7777	70 through 77



(1) Pulse width of positive pulse and negative pulse measured individually at 10 percent of its magnitude shall be less than:

10 usec, for loops and registers
12 usec, for main memory

(2) Amplitude, excluding strobe:

4.5 ±0.5 V, main memory
5.0 ±0.5 V, loops and registers

Figure 39. Strobe Waveshape for Normal Pulse

Loop and Register Signal Check

The following procedures enable checks of amplitude, pulse width, noise and crosstalk of all loops and registers (additional procedures are given in the Computer Service Manual).

CAUTION

The computer should never be moved while the memory disk is rotating; allow at least 15 minutes from power turnoff for complete memory deceleration before moving computer.

NOTE

See figure 40 for location of connectors.

1. Enter and verify Loop Load routine with tape reader, typewriter, or console keyboard.
2. Depress Start 1 button and allow computer to operate until all loops and registers (except X register) are filled (when computer halts, location counter indicator should read 2525.0).
3. Observe amplitude of loops L and V and registers A, B, R, and Z (C) with oscilloscope at connectors indicated below (amplitude excluding strobe should be 5 ±0.5 volts peak-to-peak).

Register or Loop	Connector and Pin Number		Read Amplifier Adjustment Location
A	J525-1		J529
B	J525-2	or	J530
R	J525-3	through	J531
X	J525-4	monitor	J532
Z (C)	J525-5	box	J533
L2a*	J525-6	plugged	J534
L2*	J525-7	into	J535
V2a*	J525-8	J525	J536
V2*	J525-9		J537

*L and V loops each contain two read heads which transmit signals to separate read amplifiers.

4. If necessary, adjust amplitude at read amplifier board plugged into locations indicated in table in step 3.

5. Depress Command button followed by Enter button (this fills X register).

6. Repeat steps 3 and 4 on X register.

7. Zero A register by zero-setting A2 flip-flop with System Tester.

8. Observe noise and crosstalk (in combination) with oscilloscope (should be less than 500 millivolts).

9. If noise and crosstalk exceed 500 millivolts zero-set each register or loop in turn to determine if it is being picked up from any of the other registers or loops; this is accomplished by observing on the oscilloscope if there is a decrease in noise and crosstalk as the zero setting indicates need for head adjustment).

10. Depress Start 1 button to refill registers and loops with pattern in Loop Load routine.

11. Repeat steps 7 through 10 until all registers and loops have been checked for noise and crosstalk, except in step 7 zero-set a different register or loop each time (X register is filled as in Step 5; step 10 is not repeated after the last loop-and-register check has been made).

12. Depress Start 1 button and allow all loops and registers to fill with pattern in Loop Load routine.

13. Fill X register as in step 5.

14. Check pulse width of each register and loop one at a time at 10 percent of its magnitude (width should be less than 10 usec).

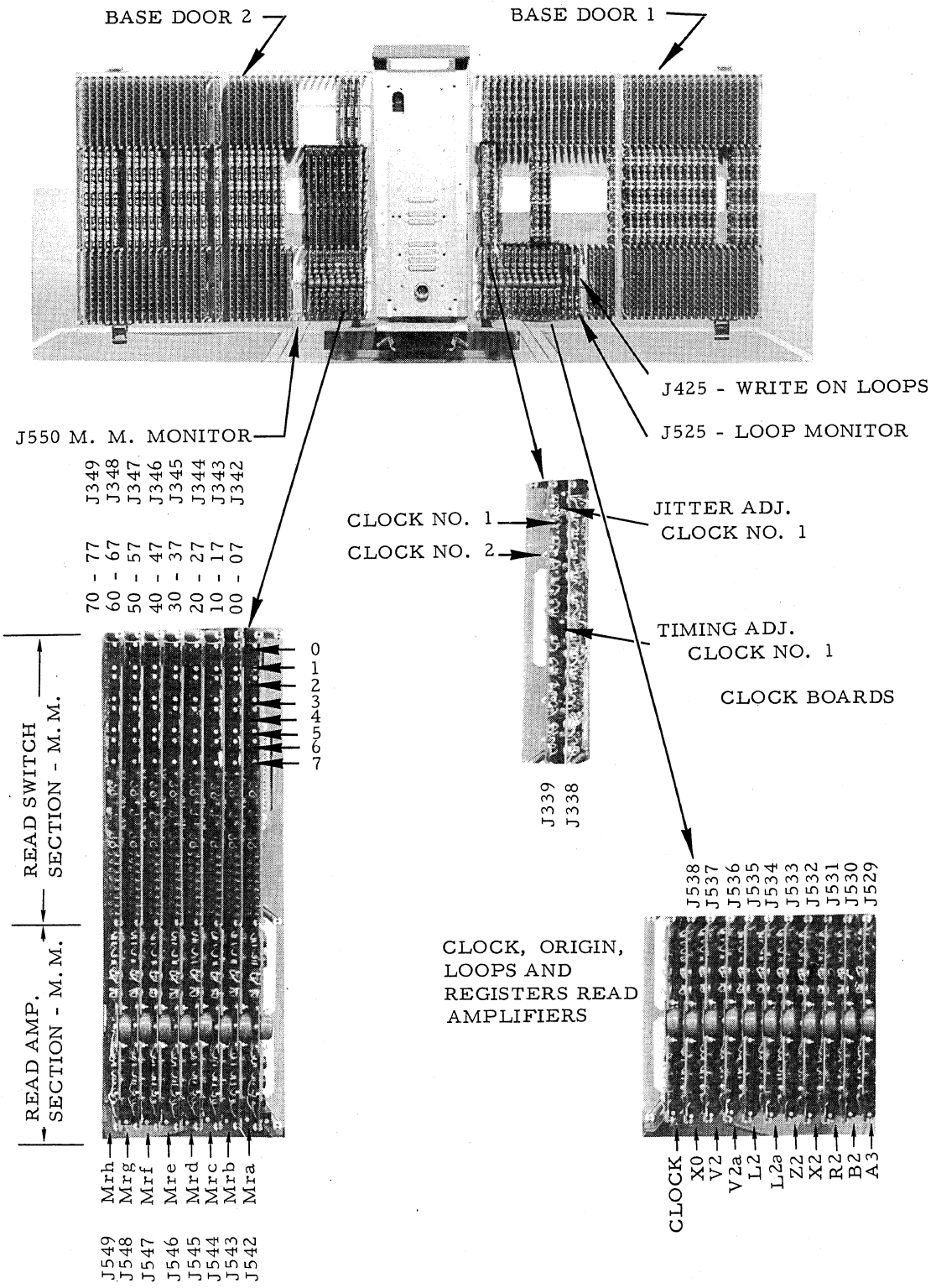


Figure 40. Location of Computer Connectors Used in Signal Checks

System Check Under Marginal Conditions

Adjustments used to establish marginal test conditions of table are as follows:

-18V	Static margin = R12
-6V	Static margin = R14
-12V	Static margin = R5
+6V	Static margin = R13
-12V	Dynamic margin = R10
+6V	Dynamic margin = R40

R12, R14, R5, R3, R10, R40 located on marginal test board of System Tester.

Clock jitter width = R4 on clock board No. 1 in computer.

Checking the computer under marginal conditions with the System Tester enables the location of beginning malfunctions. Both static and dynamic marginal conditions can be established in varying degrees. In the static mode, the System Tester disables the computer clock and the clock pulse is then initiated at will by depressing the One-Shot Clock switch. The dynamic mode enables running routines under computer clock control, but also with varying marginal conditions. Procedures are as follows:

1. Establish the marginal conditions desired by placing the Static-Dynamic Mode and Marginal Test switches on the System Tester to the appropriate position (see table 9).
2. Establish the marginal conditions necessary for the desired mode and degree of marginal condition by adjusting the potentiometers as specified above.
3. Make the operational check desired.
4. To test computer under most severe marginal conditions:

    a. Place Static-Dynamic switch in Dynamic position.
    b. Place Marginal Test switch on position five.
    c. Enter and verify any test routine.
    d. Place Operation switch in Single Step or Continuous position as desired.

Switch Setting		Action				
Mode	Marginal Test	Voltage Change				Clock Jitter
S_1	S_2	-6	+6	-12	-18	
Static	OFF	None	None	None	None	None
Static	No. 1	None	None	None	Lowered by 10%	None
Static	No. 2	Lowered by 10%	None	None	Lowered by 10%	None
Static	No. 3	Lowered by 10%	None	Raised by 10%	Lowered by 10%	None
Static	No. 4	Lowered by 10%	Raised by 10%	Raised by 10%	Lowered by 10%	None
Static	No. 5	Lowered by 10%	Raised by 10%	Raised by 10%	Lowered by 10%	None
Dynamic	OFF	None	None	None	None	None
Dynamic	No. 1	None	None	None	Lowered by 10%	None
Dynamic	No. 2	Lowered by 10%	None	None	Lowered by 10%	None
Dynamic	No. 3	Lowered by 10%	None	Jitter ±10%	Lowered by 10%	None
Dynamic	No. 4	Lowered by 10%	Jitter ±10%	Jitter ±10%	Lowered by 10%	None
Dynamic	No. 5	Lowered by 10%	Jitter ±10%	Jitter ±10%	Lowered by 10%	Jitter ±0.5 usec.

Table 9. Marginal Test Conditions

e. Start computer (if computer halts at any place in the routine other than where so directed, the fault will most likely be a component or circuit failure caused by the marginal operating condition).
f. Remove power from computer and replace component or circuit board that failed.
g. Rerun routine used in step c to assure satisfactory operation under this marginal condition.

CONTROL CONSOLE

Normally the control console requires relatively little maintenance, consisting principally of replacing indicator neons and visual readout NIXIE tubes. All controls and indicators, however, should be checked periodically to detect incipient malfunctions if possible and ensure proper operation. Troubleshooting of control console malfunctions will usually involve the visual readout controls and NIXIE tube.

Procedures for checking all controls and indicators are given on the following pages. If a real or beginning malfunction is detected during this preliminary checkout, it can be usually isolated by testing the connections and circuits involved. Detailed procedures for maintenance of the control console are presented in the Input-Output Service Manual.

Controls and Indicators

When power to the computer has been turned on, the neon indicators on the sloping portion of the control panel should initially display the following characteristics. Any indication contrary to that given below should be noted as it will aid in locating a malfunction, if one exists.

Neon	Condition
ON	should be a steady glow
READY	should be a steady glow
FILL SOURCE KEYBOARD	should be a steady glow; if not, depress FILL SOURCE KEYBOARD button.
HIGH TEMP	should be off
INTER-LOCK	should be off
VERIFY ERROR	should be off; if on, depress ERROR RESET button

Neon	Condition
OVERFLOW ERROR	should be off; if on, depress ERROR RESET button
OUTPUT ERROR	should be off; if on, depress ERROR RESET button
COMPUTE	should be off

Fill Modes Controls and Indicators

Checking the fill mode indicators achieves two objectives: (1) a test on correct operation of the keys involved, and (2) verification of correct operation of the indicators. Procedures for checking the controls and indicators used in entering information in command and number fill modes and in setting the location counter are:

Number Fill Mode. Depress "N" key; NUMBER neon should light. Depress "+" key; "D" and "+" neons to left of visual readout should light. Depress CLEAR key to extinguish "D" and "+" neons.

Depress "-" key; "D" and "-" neons to left of visual readout should light.

Command Fill Mode. Depress "C" key; command neon should light.

Depress "+" neons to left of visual readout should light. Depress CLEAR key to extinguish "C" and "+" neons.

Depress "-" key; "C" and "-" neons to left of visual readout should light. Depress CLEAR key to extinguish "C" and "-" neons.

Location Selection. Depress LOCATION key; LOCATION neon and "0" neon to left of visual readout should light. Depress CLEAR key; "0" neon should remain lighted. Select location 00001 by depressing corresponding number keys and ENTER; all LOCATION COUNTER neons should be off except HALF-WORD neon.

Continue check as follows: select location 11110 and ENTER; only number "1" neons in both CHANNEL and SECTOR sections of LOCATION COUNTER should light. Proceed with check by selecting locations 22220, 33330, 44440, 55550, 66660, and 77770, noting LOCATION COUNTER display after each selection by counting (octally) the numbers represented by the neons. For example, when location 55550 is selected, neons 4 and 1 should be lighted in both positions of each of the channel and sector portions of the location counter indicator.

Visual Readout Nixie Tubes and Decimal Number Keys

While the preceding Location Selection check will display the numbers entered on the visual readout, the NIXIE should be further tested. To conduct this further check, depress number 1 key at the keyboard 15 times to fill all 15 NIXIEs, observing that each in turn displays a complete configuration of the number entered. Depress Clear key to clear NIXIEs and enter number 2 in the same manner and making the same observation. Continue by displaying all numbers, including zero and decimal point, in this manner. A test on correct operation of the NIXIE under computer control is given in Section V; procedure for checking operation of the console keyboard is presented later in this section.

Keyboard Fill Source Button and Indicator

Proper functioning of the Keyboard Fill Source button and its indicator is best confirmed by checking them with the System tester. The procedure is:

1. Turn "T_o" flip-flop on System Tester on; Fill Source neon should extinguish.
2. Depress Fill Source button; neon should relight.

Error and Environment Indicators

Because these indicators are the principal sources of information regarding proper operation, they should be tested frequently to assure correct operation. Tests on the Verify, Output, and Overflow Error indicators are performed with the System Tester. The procedure on each test is:

Verify Error Indicator.

1. Set "H_o" and "J_o" on System tester to true (1-set) states; Verify Error neon should flash.
2. Set "H_o" and "J_o" to false (0-set).

Output Error Indicator.

1. Set "J13" to true (1-set); Output Error neon should flash.

Overflow Error Indicator

2. Set "J13" to false (0-set); "J_o" should be true and Overflow neon should flash.
3. Set "J_o" to false (0-set).

Interlock Indicator.

1. Release both doors on computer sufficiently to energize Interlock neon.
2. Push in one of interlock switches on computer door (Interlock neon should remain flashing).
3. Release interlock switch and push in other switch (Interlock neon should remain flashing).
4. Close and latch both doors (Interlock neon should extinguish).

Line Transient.

1. Connect and Adjust Variac until system is supplied with 108 V ac as monitored on voltmeter (Line Transient neon should light).
2. Depress Error Reset button (Line Transient neon should extinguish momentarily).
3. Adjust Variac to 115 V ac (Line Transient neon should extinguish).
4. Adjust Variac until system is supplied with 125 V ac as monitored on voltmeter (Line Transient neon should light).
5. Depress Error Reset button (Line Transient neon should extinguish momentarily).
6. Adjust Variac to 115 V ac (Line Transient neon should extinguish).

Compute Indicator and Start and Stop Buttons

Proper operation of the Compute indicator can be determined by running a known program already stored in memory or the following short routine:

```
L 00000 ENTER

C    +0000030 +0100040 ENTER      (L00000)
     +6000030 +3600030 ENTER      (L00010)
     +570000000000000  ENTER      (L00020)
N    +0. ENTER                    (L00030)
     +2. ENTER                    (L00040)
```

Depress START 2 button; COMPUTE neon should light and FILL SOURCE neon should extinguish. (The LOCATION COUNTER will display a continuous random lighting during the computation; all other neons should be extinguished except READY and ON. The visual readout will continuously display the changing addition as the computation proceeds.) Depress STOP button; computer will stop computing and COMPUTE neon will extinguish and FILL SOURCE neon will light. (The LOCATION COUNTER neons will display the location the Computer halted when computation was stopped.)

Start 1, Start 2, and Start 3 Buttons Proper functioning of these buttons can be verified by the following procedure (correct operation of the keyboard in the various fill modes should be determined first to ensure proper entry):

1. Enter the following routine:

Location	Commands
0001.0	+0000000 +0000000
0002.0	+0000000 +0000000
0003.0	+0000000 +0000000

2. Place the Operation switch in Single-Command position.
3. Depress Start 1 button (location counter should read 0001.1).
4. Depress Start 2 button (location counter should read 0002.1).
5. Depress Start 3 button (location counter should read 0003.1).

Octal, Decimal, and Command Readout Buttons

The three readout buttons can be checked by the following procedure (the test can also serve as a check on proper functioning of the rotary switches by following the procedures outlined under comments):

1. Enter the following information:

Location	Information
0000.0	+2443360 -4700760

2. Readout memory location 0000.0 with command button; visual readout should display +244336004700760
3. Readout memory location 0000.0 with octal button; visual readout should display +2443361160174
4. Readout memory location 0000.0 with decimal button; visual readout should display +523789.0

Comments

Operation of the rotary switches can be checked by placing the information in other main memory locations and by transferring it to the loops and registers with the appropriate commands after entry and prior to readout. The information can be copied into any loop location by executing the Copy to L Loop and Copy to V Loop instructions. Execution of a Clear and Add command places the information in the Accumulator and Command register; execution of a Clear and Add command followed by the Exchange A and R instruction places it in the R

register; execution of a Clear and Add command followed by a Store A and Exchange A and X instruction places it in the X register.

Single Command Switch

Correct operation of the computer in Single Command operation can be ascertained with the following procedure:

1. Enter the following information:

Location	Commands
0001.0	+0000000 +0000000
0002.0	+0000000 +0000000
0003.0	+0000000 +0000000

2. Place Operation switch in Single-Command position.
3. Depress Start 1 button; location counter should read 0001.1.
4. Depress Start button; location counter should read 0002.0.
   Depress Start button; location counter should read 0002.1.
   Depress Start button; location counter should read 0003.0.
   Depress Start button; location counter should read 0003.1.

Preset Stop Switch

The Preset Stop switch can be tested with the following procedure:

1. Enter the following routine:

Location	Commands
0001.0	+5700100 -0000000
0010.0	+5700201 -0000000
0020.0	+0000000 +5700010

2. Depress Start 1 button.
3. Set Channel and Sector Selector switches to 0010 and place Preset Stop switch in 1st position to stop the computer after executing the first command in a word; routine should halt and location counter should read 0020.1.
4. Place Preset Stop switch in Off position.
5. Depress Start 1 button.
6. Set Channel and Sector Selector switches to 0020 and place Preset Stop switch in 2nd position to stop the computer after executing the second command in a word; routine should halt and location counter should read 0001.0.

7. Place Preset Stop switch in Off position.
8. Depress Start 1 button.
9. Set Channel and Sector Selector switches to 0001 and place Preset Stop switch in 2nd position; computer should continue operating.
10. Set Channel and Sector Selector switches to 0020 and place Preset Stop switch in 1st position; computer should continue operating.

Transfer Stop Switch

Proper operation of the Transfer Stop switch can be determined by the following procedure:

1. Enter the following routine:

Location	Commands
0001.0	+5700011 +5400020
0002.0	+5500021 +5600011

2. Place Transfer Stop switch up (on).
3. Depress Start 1 button; location counter should read 0001.1.
4. Depress Start button; location counter should read 0002.0.
5. Depress Start button; location counter should read 0002.1.
6. Depress Start button; location counter should read 0001.1.

Sense Switches B, C, D

Correct operation in both obeying and ignoring the settings of Sense Switches B, C, and D can be determined with the routine and procedures given below (the routine is changed as indicated according to the switch being tested; the procedures remain the same, however, except as stated):

1. Enter one of the following routines, according to the switch being tested (memory locations are the same for each routine):

Location	Commands Switch B	Commands Switch C	Commands Switch D
0001.0	+5400051 +7700051	+5500051 +7700051	+5600051 +7700051
0002.0	+7700000 +5400040	+7700000 +5500040	+7700000 +5600040
0003.0	+7700040 +7700000	+7700040 +7700000	+7700040 +7700000
0004.0	+5400010 +7700010	+5500010 +7700010	+5600010 +7700010
0005.0	+7700000 +5400021	+7700000 +5500021	+7700000 +5600021
0006.0	+7700021 -0000000	+7700021 -0000000	+7700021 -0000000

2. Place Sense Switch B (C or D) up (tests if sense switch transfer instruction is ignored).

3. Place Operation switch in Continuous position.

4. Depress Start 1 button (computer should halt).

5. Read location counter and C register (readouts should be as follows):

Location Counter	C Register Switch B	C Register Switch C	C Register Switch D
0005.1	+540005117700051	+550005117700051	+560005117700051

6. Depress Start button; when computer halts, read out location counter and C register (readouts should be as follows):

Location Counter	C Register Switch B	C Register Switch C	C Register Switch D
0002.1	+770002100000000	+770002100000000	+770002100000000

7. Start button; when computer halts, read out location counter and C register (readouts should be as follows):

Location Counter	C Register Switch B	C Register Switch C	C Register Switch D
0004.0	+770004017700000	+770004017700000	+770004017700000

8. Depress Start button; when computer halts, read out location counter and C register (readouts should be as follows):

Location Counter	C Register Switch B	C Register Switch C	C Register Switch D
0001.0	+540001017700010	+550001017700010	+560001017700010

9. Place Sense Switch B (C or D) down (tests if sense switch transfer instruction is followed).

10. Depress Start 1 button; routine should run continuously, after it has run for a couple of minutes, depress Stop button.

11. Place Transfer Stop switch up (on).

12. Depress Start 1 button; when computer halts, read out location counter and X register (readouts should be as follows):

Location Counter	X Register Switch B	X Register Switch C	X Register Switch D
0005.1	+000000000000010	+000000000000010	+000000000000010

13. Set location counter to 0002.1.

14. Depress Start button; when computer halts, read out location counter and X register (readouts should be as follows):

Location Counter	X Register Switch B	X Register Switch C	X Register Switch D
0004.0	+000000000000021	+000000000000021	+000000000000021

Typewriter-Actuated Control and Indicators

Most control operations carried on from the control console can also be conducted from the typewriter. The console indicators, however, provide the indication of the control activities performed from the typewriter and the internal computer operations. Proper operation of the typewriter as a control unit as well as correct indication by the console neons can be ascertained by performing the following check procedures:

Turn Compute-Manual switch on punch control panel to Computer position, then depress Fill lever on typewriter; Fill Source neon on control console should extinguish.

Depress Letters key and L key on typewriter; Location neon on control console should light (the Location neon should remain lighted when the Figures key and numeral keys on the typewriter selecting a location are depressed).

Depress Return key on typewriter (this sets the location counter); the Location Counter lights should remain lighted.

Depress Letters key and C key on typewriter; Command neon on control should light (the Command neon should remain lighted when the Figures number keys representing a pair of commands, and the Return keys are depressed; the Location Counter neons should step up to show the next location).

Depress the Letters key and N key; Number neon on control console should light (the Number neon should remain lighted when the Figures, plus, number keys representing a number, decimal point, and the Return keys on the typewriter are depressed; the Location Counter neons should step up to the next location when the decimal point and Return keys are depressed during mixed number input and when the Return key is depressed on integer or fraction input).

If a routine is stored in the computer, set the Location Counter to the location at which the routine begins and depress the S key on the typewriter; the Compute neon on the control console should light and remain so until the computer stops or is halted.

To stop computation manually from the typewriter, depress the H key; the Compute neon on the control console should extinguish.

INPUT-OUTPUT UNITS

Typewriter

When checking or troubleshooting the typewriter, proper functioning of controls should be determined before attempting to locate the cause of incorrect operation, unless the area of malfunction is definitely established from a report of the difficulty.

Proper functioning of the various typewriter controls can be readily checked by operating them and observing the corresponding action. All controls and their functions are given in the Input-Output Service Manual; observing the performance of the functions will serve as a satisfactory preliminary check on operation of the controls and usually isolate the malfunction. More extensive checking, however, may be necessary on such malfunctions as weak impression, improper tab operation or carriage action, and main power faults. Weak impression may also be caused by weak drive output in the decoder and mechanical drive units, in which case all circuits and components associated with typing under computer control may require checking. Improper tab operation or carriage return may require checking the circuits carrying the signals from the computer, signal decoding, and the mechanical action involved in tab and carriage return operations. A routine to test typing with tab operation under computer control is given in the Test Routine Section. Main power faults will necessitate testing of the power supply circuits for faulty connections and components.

Incorrect entry and recording of information, or non-entry and non-recording, can usually be traced to malfunctions in the coder or decoder assemblies, respectively. The source of the trouble may, however, also be located in the power assembly supplying mechanical drive to the coder and decoder or in the circuitry transmitting the signals to and from the computer. Procedures for isolating these malfunctions are given in the Input-Output Service Manual.

Preliminary checking of the typewriter for correct entry of information into the computer involves principally a comparison of the characters recorded from the computer with those entered. This can be accomplished by running the test routine given in Section . Correct entry of numbers in command and number formats can also be verified by displaying the contents of a memory location into which this information has been entered from the typewriter. Proper input of alphanumeric

characters can be verified by entering the information and then recording it with the typewriter or punch under computer control, or by entering information in a specific sequence and then reading it out in command format. A procedure for manually entering alphanumeric characters from the typewriter and reading them out in command format on the visual readout is outlined below. A routine for testing alphanumeric input from the typewriter under computer control is given in Section Before checking for accuracy of input, proper operation of the typewriter in the various input modes should be confirmed by performing the steps outlined under Typewriter Actuated Indications and Control.

Depress Fill key on typewriter, then the characters L 00100, followed by Carriage Return.

Depress the following keys in the sequence indicated:

RYRYRYRY	Carriage Return
01234567	Carriage Return
89ZAMXLC	Carriage Return
NSHDFGKV	Carriage Return

Read out the following memory locations on the visual readout, using the Command readout button (the display should read as indicated):

Location	Information Displayed
0010.0	-525252 1 0 525252 1
0011.0	+327460 1 0 520523 1
0012.0	-330421 1 1
0013.0	-605504 1 0 672377 0

Initial checkout of the typewriter for correct recording of output consists mainly of comparing the information transmitted from the computer with that typed or typewriter action performed. This is most easily accomplished with test routines in which the output as it should be recorded is known. These test routines are given in the Test Routine Section. Correct output can also be checked by entering information in a specific memory location and then entering and executing one of the Type commands. These diagnostic checks are especially useful when the source of the malfunction is in the computer output circuitry rather than the typewriter. Malfunctions of incorrect decoding and recording by the typewriter are usually detected by the echo-checking circuits. These circuits verify the output with the signals sent from the computer. Any discrepancy causes the computer and typewriter to halt and the Output Error indicator on the control console to light. The character

recorded incorrectly can be determined by comparing its configuration with that displayed by the output register indicator on the punch. Output errors can be isolated by troubleshooting computer and typewriter circuits and components associated with this function.

Console Keyboard and Visual Readout

Console Keyboard

Should preliminary checkout of the various control console indicators and controls indicate a malfunction in the keyboard, the fault can be further isolated by the following procedure:

1. Enter the following routine from the console keyboard (correct entry of each pair of commands can be verified by reading out the Accumulator before depressing the Enter button and the memory location into which it was placed after depressing the Enter button):

Location	Commands
0001.0	+3600050 +7700020
0002.0	+3600060 +7700030
0003.0	+3600070 +7700040
0004.0	+3600100 +7700010

2. Enter the following numbers in Number Fill mode (correct keying of numbers can be verified by observing display on visual read-out before depressing Clear or Enter button):

Location	Numbers	
0005.0	+8192	Depress Clear button
0006.0	-00048828125	Depress Enter button
0007.0	+15722656250	Depress Enter button
0010.0	-590864	Depress Enter button

3. Depress Start 1 button.

4. Read-out location 0002.0; display should be +00 00000 0 01 00000 (in command format).

5. Depress Start button.

6. Read out location 0003.0; display (in command format) should be -00 02000 0 00 00000.

7. Depress Start button.

8. Read out location 0004.0, display (in command format) should be +12 04000 0 00 00000.

9. Depress Start button.

10. Read out location 0001.0; display (in command format) should be 00 00000 1 10 10100.

(The routine should repeat if the Start button continues to be depressed).

Visual Readout

If the control console checkout procedures indicate a malfunction in the visual readout, the fault can be further isolated by running the NIXIE Tube Test routine given in the Test Routine section. This test should define the tube assembly or its associated circuitry causing the difficulty. The exact source of the malfunction can then be determined by examining the circuit connections and components used by the malfunctioning tube assembly. Neon indicator faults can be readily traced by examining these circuits. If a fault within the computer appears to be the cause of the malfunction, it can be confirmed by entering known information into a specific memory location and then reading it out on both the visual readout and typewriter or punch.

Photoelectric Paper Tape Reader

Unless a malfunction in the paper tape reader is definitely known from an explanation of the trouble, it is often to advantage to first check the d-c voltages, then the controls and mechanical operation. Application of power to the reader upon turning on the Power On-Off circuit breaker and Motor switches can usually be checked simply by observing if the capstan rotates. If the capstan is not rotating, the trouble can be isolated by checking for circuit continuity. If the capstan rotates, and d-c voltages are correct, the exciter lamps and the photodiode window on the reader head should be checked to determine that the lamps are not blackened or burned out and that the window is clean. Also check the AGC board signals and adjustments. Proper functioning of the remaining switches and indicators can be determined with the following procedures:

Fill Button and Indicator

Insert any program tape into reader and depress Fill button. If tape fails to pass through reader head, check idler roller for extreme wear, switch and circuits for faulty connections and components, and insufficient pressure against capstan drum. The Fill indicator should light upon depressing the Fill button and extinguish when tape movement stops.

Verify Button and Indicator

Enter any program tape through reader, then re-enter same tape but by depressing Verify instead of Fill button. If tape fails to pass through reader, check switch and circuits for faulty connections and components. The Verify indicator should light upon depressing the Verify button and extinguish when tape movement stops.

Stop Switch

Enter any program tape through reader; before tape reaches end depress Stop switch. If tape fails to halt, check switch and circuit connections and control components.

Tape Advance Switch

Insert endless program tape in reader and depress Fill switch. After a short length of tape has passed through the reader, depress Stop switch, then the Tape Advance switch. If the tape fails to start at any point other than the beginning of the tape and to stop at slit at starting point, check switch and circuits for faulty connections and components.

Mechanical actions of the tape reader requiring checkout are usually indicated by faulty feeding of tape past the reader head. Indications include tape slippage on starting tape movement and excessive tape movement after a stop has been initiated. Cause of excessive tape slippage on starting can be determined by checking the idler roller for excessive wear, the solenoid engaging the idler roller for malfunction, the tape pressure spring for weakness, and the movement of the locking handle for faulty closing. If the tape only accelerates slowly, the drive belt between motor and capstan should be checked for slippage. Excessive tape movement after a stop has been given can be diagnosed by checking the solenoid engaging the idler roller for abnormal delay in action, and the brake for faulty braking action.

Incorrect reading by the tape reader can best be confirmed and the cause isolated by running test routines. The routines testing operation and correct reading of the tape reader are given in the Test Routine section. Correct reading may also be ascertained by entering a tape of known content, then displaying the contents of the memory locations into which the information was entered and comparing the readout with the information in the tape. If recording in memory is faulty, one work at a time may be entered and displayed in the accumulator.

Paper Tape Punch

Most punch malfunctions are attributable to failures of mechanical parts or wear to a degree requiring readjustment. But unless these conditions are definitely known to be the cause of the malfunction, it is advisable to first check the controls and test-run a tape to note any improper operation.

Proper application of power to the punch upon turning on the Power On-Off circuit breaker and Motor switches can be readily ascertained. If the punch remains inoperative, checking the power supply circuits will locate the fault. Proper functioning of the other switches and indicators can be determined by following the procedures given below:

Computer-Manual Punch Switch

In Computer position, place information in memory, then attempt to punch it into a blank tape with any of the Punch instructions. To test Manual position, strike a key on the typewriter. Though failure of the punch to respond in either instance may be caused by a malfunction elsewhere, the switch circuits should be tested before conducting other checkout or troubleshooting procedures. (See also Punch-External Switch below.)

Punch-External Switch

Unless the system includes a plotter or other external output device, only the Punch position need be checked. This switch should be in the Punch position when the Computer-Manual Punch Switch is tested Because failure of the punch to operate may be caused by malfunction in either switch, the Punch position should be tested simultaneously with the two positions of the Computer-Manual Punch switch.

Tape Advance Switch

Place a section of blank tape in punch and depress Tape Advance button. If tape fails to advance, check connections and circuits associated with tape advance process.

Output Register Indicator

Because this indicator must always display the correct code transmitted from the computer to an output unit, it should be checked frequently to assure correct functioning. To test the indicator, enter a program which will transmit a series of the same character to the

punch at a time, the first series consisting of a character with a hole punched in only one channel, the second series of a character punching in only another channel, and continuing until a series has been punched in all channels. (This procedure may be repeated, if desired, with a program containing series of characters punching in combinations of channels.) Observe the output register indicator while the punch is recording, if further checking is necessary to determine correct output, visually inspect tape punched. If the indicator malfunctions, inspect neons, connections, and circuits to output register. If further checking is necessary, use the System Tester to monitor the output register.

Tape punch mechanical malfunctions are usually due to misadjustment caused by usage and consequent wear and misalignment. Procedures for disassembly and adjustment are given in the Input-Output Service Manual. Indications of maladjustment include improper tape feeding, poor quality punching, and double punching. A test to confirm existence of double punching is given in the test routines. Incorrect punching, if not definitely known, can best be detected and the cause located by running the test routines presented in the Test Routine section. Malfunctions can be further isolated and the cause determined by checking circuits and components; procedures for accomplishing this are given in the Input-Output manual.

TAPE PREPARATION PROCESS

With Computer Off

Determining correct operation of the typewriter and tape punch as a tape preparation system can be accomplished with the following procedure:

1. Turn on power at desk circuit breaker switch, typewriter, and tape punch.
2. Place blank tape in punch and turn Compute-Manual switch to Manual position.
3. Depress each typewriter character and function key having a teletype code equivalent in Letters Shift and Figure Shift modes (any sequence of key depression is satisfactory but one should be selected that will permit easy comparison between character typed and punched).
4. Visually inspect tape to determine if character configurations punched correspond to character typed and if configuration punched is legitimate code.

If the test above indicates a malfunction, the source can be isolated by following the checkout and troubleshooting procedures for each unit and by checking the connections, circuits, and components associated with the tape preparation process.

With Computer Operating

Proper functioning of the typewriter and tape punch in tape preparation while the computer is operating can be determined with the following procedure (perform check with computer off as given above before testing with computer operating):

1. Enter any routine that will operate in a loop and start computer.
2. After computer has operated for a few seconds, depress Stop button.
3. Turn Compute-Manual switch on tape punch to Manual position.
4. Depress Start button on computer.
5. Turn Power switch on punch to on.
6. Depress Tape Advance button on punch and hold for a few seconds (computer operation should not be affected).
7. Depress following keys on typewriter in succession: FIGS 1 2 3 4 5 6 7 8 9 0 - s $! & HALT ' () + / : ; ? , . LINEFEED BLANK TAB RETURN (computer operation should not be affected).
8. Depress Letters Shift key followed by keys listed in Step 7 except FIGS (computer operation should not be affected).
9. Turn Punch-External switch to External position.
10. Depress typewriter keys as in Steps 7 and 8 (punch should not operate and computer operation should not be affected.

If computer operation should be affected, check circuits from typewriter to computer for wrong or faulty connections and malfunctions.

TEST ROUTINES

The test routines are a quick, effective means of determining proper operation of all or a part of a computing system. Frequently, it also can aid materially in locating a malfunction, particularly when the routine provides indication of the area of malfunction or records information which can be compared with known, correct results.

Most test routines check only a portion of a system. This is because (1) a routine testing operation of a system completely would require considerable time to run and (2) usually the unit at fault is known. Thus, while a few routines check the system extensively, the majority test one unit such as the memory or several units in combination such as the typewriter and tape punch.

Selecting the most appropriate routine to locate a malfunction requires a complete understanding of the theory of operation of the system. For example, if the punch records the wrong character the error could be caused by faulty execution of the Punch Character command rather than the punch recording incorrectly. Consequently the Type Character, Type/Punch Character, and Punch Character test routines might locate the difficulty more quickly than the Input-Output and Memory Test routine.

Interpreting a routine halt caused by a computer malfunction necessitates detailed knowledge of the system logic. When the logic is understood, the malfunction can usually be readily isolated to a particular circuit or component, particularly if the System tester is used in conjunction with the test routine to check the flip-flops and gates involved in the halt.

The test routines in this section, together with those among the tests in Section IV, enable a relatively easy check of the entire system both in preventive maintenance and trouble shooting. Test tapes are available for all except the very short routines. These, as well as the long routines if the tape reader is inoperative, can be readily entered from the typewriter or console keyboard. Operating instructions are included with each routine.

COMMAND TEST ROUTINE

Purpose

To check execution of all computer commands under marginal or normal operating conditions. (Operation as described in Steps 1 through 5 below tests all commands except input-output and HTR, TMI, TPL, TOV, TSB, TSC, and TSD; operation as described in Steps 6 through 8 tests the transfer commands mentioned.)

Operating Instructions

1. If routine is to be run under marginal conditions, connect System Tester and set marginal test switch on System Tester to 5; if routine is to be run under normal operating conditions, operate computer without System Tester connected.
2. Enter and verify Command Test routine with tape reader.
3. Place all Sense switches on Control Console in down position.
4. Depress Start 1 button. (Visual readout will display on right side of readout the number of times the routine has run successfully (see comments following 8).
5. After routine has run the desired number of times, depress Stop button.
6. Set location counter to 2077.0
7. Depress Start button. (Routine should halt at location 2101.1 and display +0000000-0000510.)
8. Continue as indicated below:

Operation	Correct Result
Depress Start button	Routine halts at location 2103.1 and displays +0000000-0000530
Depress Start button	Routine halts at location 2104.0 and displays +0000000-0000540
Depress Start button	Routine halts at location 2104.1 and displays +0000000-0000550

Operation	Correct Result
Depress Start button	Routine halts at location 2105.0 and displays +0000000-0000560
Depress Start button	Routine halts at location 2102.0 and displays +0000000-0000520
Place Sense Switch B up; depress Start button	Routine halts at location 2103.1 and displays +0000000-0000530
Keep Sense Switch B up; depress Start button	Routine halts at location 2104.1 and displays +0000000-0000550
Keep Sense Switch B up; depress Start button	Routine halts at location 2105.0 and displays +0000000-0000560
Place Sense Switches B and C up; depress Start button	Routine halts at location 2101.1 and displays +0000000-0000510
Keep Sense Switches B and C up; depress Start button	Routine halts at location 2103.1 and displays +0000000-0000530
Keep Sense Switches B and C up; depress Start button	Routine halts at location 2105.0 and displays +0000000-0000560
Place Sense Switches B, C, and D up; depress Start button	Routine halts at location 2102.0 and displays +0000000-0000520
Keep Sense Switches B, C, and D up; depress Start button	Routine halts at location 2103.1 and displays +0000000-0000530
Keep Sense Switches B, C, and D up; depress Start button	Routine halts at location 2101.1 and displays +0000000-0000510

COMMENTS

If the routine halts, the location counter will be reset to the location from which the transfer command came. The command on which the malfunction occurred will be found two locations prior to that indicated; this command can be displayed on the visual readout by selecting the proper location with the rotary selector switches and depressing the command readout button. This routine can be used in its entirety or in two separate portions: Steps 1, 2, 3, 6, 7, and 8.

The command test routine printout is shown on the subsequent pages.

```
L00000  +2525250+2525250
    1.  +0003000+6004000
    2.  +0305000+5000031
    3.  +5702100+0003000
    4.  +0103000+6004010
    5.  +0305010+5000061
    6.  +5702100+0203000
    7.  +6004020+0305020
L00100  +5000110+5702100
    1.  +3003000+0403000
    2.  +3504030+0305030
    3.  +5000140+5702100
    4.  +0004040+0305040
    5.  +5000160+5702100
    6.  +3003000+0503020
    7.  +3504050+0305050
L00200  +5000210+5702100
    1.  +0004060+0305060
    2.  +5000230+5702100
    3.  +3403000+0603000
    4.  +3504070+0305070
    5.  +5000260+5702100
    6.  +0004100+0305100
    7.  +5000300+5702100
L00300  +3003000+0703170
    1.  +3504110+0305110
    2.  +5000330+5702100
    3.  +0004120+0305120
    4.  +5000350+5702100
    5.  +0003000+1103000
    6.  +3504130+0305130
    7.  +5000400+5702100
L00400  +0004140+0305140
    1.  +5000420+5702100
    2.  +0003000+1303000
    3.  +6004150+0305150
    4.  +5000450+5702100
    5.  +1504160+6004170
    6.  +0004160+0305160
    7.  +5000500+5702100
L00500  +0004170+0305170
    1.  +5000520+5702100
    2.  +0003010+2003000
    3.  +6004200+0305200
    4.  +5000550+5702100
    5.  +0003010+2103000
    6.  +3504210+0305210
    7.  +5000600+5702100

L00600  +0004220+0305220
    1.  +5000620+5702100
    2.  +0003010+2203000
    3.  +3504230+0305230
    4.  +5000650+5702100
    5.  +0004240+0305240
    6.  +5000670+5702100
    7.  +0003010+2303000
L00700  +3504250+0305250
    1.  +5000720+5702100
    2.  +0004260+0305260
    3.  +5000740+5702100
    4.  +0003200+2503200
    5.  +6004270+0305270
    6.  +5000770+5702100
    7.  +0003000+3303000
L01000  +6004300+0305300
    1.  +5001020+5702100
    2.  +0003000+4000170
    3.  +6004310+0305310
    4.  +5001050+5702100
    5.  +0003000+4100170
    6.  +6004320+0305320
    7.  +5001100+5702100
L01100  +1504330+6004340
    1.  +0305340+5001121
    2.  +5702100+0004330
    3.  +0305330+5001141
    4.  +5702100+3003000
    5.  +3504350+4300000
    6.  +3504370+0305370
    7.  +5001200+5702100
L01200  +0004350+0305350
    1.  +5001220+5702100
    2.  +0004360+0305360
    3.  +5001240+5702100
    4.  +0004400+0305400
    5.  +5001260+5702100
    6.  +4403220+3504410
    7.  +0305410+5001301
L01300  +5702100+0004420
    1.  +0305420+5001321
    2.  +5702100+3003000
    3.  +4500000+3504430
    4.  +0305430+5001351
    5.  +5702100+0004440
    6.  +0305440+5001371
    7.  +5702100+6403400
```

```
L01400  +6503500+0003500
     1. +0303400+5001421
     2. +5702100+0003510
     3. +0303410+5001441
     4. +5702100+0003520
     5. +0303420+5001461
     6. +5702100+0003530
     7. +0303430+5001501
L01500  +5702100+0003540
     1. +0303440+5001521
     2. +5702100+0003550
     3. +0303450+5001541
     4. +5702100+0003560
     5. +0303460+5001561
     6. +5702100+0003570
     7. +0303470+5001601
L01600  +5702100+6603600
     1. +6703700+0003700
     2. +0303600+5001631
     3. +5702100+0003710
     4. +0303610+5001651
     5. +5702100+0003720
     6. +0303620+5001671
     7. +5702100+0003730
L01700  +0303630+5001711
     1. +5702100+0003740
     2. +0303640+5001731
     3. +5702100+0003750
     4. +0303650+5001751
     5. +5702100+0003760
     6. +0303660+5001771
     7. +5702100+0003770
L02000  +0303670+5002011
     1. +5702100+0002040
     2. +0102050+6002040
     3. +3602040+5702160
     4. -0000000-0000000
     5. +0000000-0000010
     6. +0000000-0000350
     7. +0000000-0000370
L02100  +1502240+6002250
     1. +4300000+6002260
     2. +0002250+4202151
     3. +4000010+4100010
     4. +4202141+3600000
     5. +5300000+7700000
     6. +6402300+6504000
     7. +6504100+6504200
L02200  +6504300+6504400
     1. +6504500+6504600
     2. +6504700+6510500
     3. +6503700+5700010
     4. +6067400-0210300
     5. +0000000-0000321
     6. +0000000-1161671
     7. -0000000-0000000
L02300  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02400  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02500  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02600  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02700  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
```

Command Test Routine Printout (Cont)

```
L03000  +0012250+6567651
    1.  +0000000-0001000
    2.  +6010000+3610000
    3.  +0000000+5710000
    4.  +0000010+0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L03100  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  +0010170-0000000
L03200  +0000000-1161000
    1.  -5263431+0370170
    2.  +2420000-0000000
    3.  +0000000-0000120
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L03300  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L03400  +1235210-7325010
    1.  +4502640+7124761
    2.  +7563241+6321761
    3.  +6574130-5746210
    4.  +1524361-3736351
    5.  +3514340+5321721
    6.  +6401650-7503450
    7.  +6324511+6417510
L03500  +1235210-7325010
    1.  +4502640+7124761
    2.  +7563241+6321761
    3.  +6574130-5746210
    4.  +1524361-3736351
    5.  +3514340+5321721
    6.  +6401650-7503450
    7.  +6324511+6417510
L03600  -3134650+3464310
    1.  +4642711-4752631
    2.  +4565540+6654450
    3.  +1233211-3321121
    4.  +7533571+3357751
    5.  +1533511+3351151
    6.  +6244260-4426620
    7.  +7511571+1157751
L03700  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L04000  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L04100  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L04200  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
L04300  -0000000-0000000
    1.  -0000000-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
```

```
L04400  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L04500  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L04600  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L04700  -0000000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L05000  +0012250+6567651
     1. +0024521+5357530
     2. -0012250+6567651
     3. +5125651+3724000
     4. +0000000-0000741
     5. +6676471-4340000
     6. -0000000+5707040
     7. -5125651+3724000
L05100  +0000000-0000741
     1. +5243261+4000000
     2. +0000000-1161671
     3. +0000011-5321471
     4. +2321130-5123341
     5. +0000011-5321471
     6. +0000000-0000000
     7. +0000000-0000440

L05200  +0000000-0614231
     1. +0000000-0614240
     2. +0000000-0614240
     3. +0000000-0614231
     4. +0010200+2306551
     5. +0000000-0614241
     6. +0000000-0614241
     7. +0001440-0000000
L05300  +0012250+6567651
     1. +0000000-0512561
     2. +5353571-5400000
     3. +0000000-0000000
     4. +0000000-0001070
     5. +0012250+6567651
     6. +0000000-0001000
     7. +0000000-0001000
L05400  +0012250+6567651
     1. +4400000-0000000
     2. +0000000-0000050
     3. +5125651+3726000
     4. +0000000-0000740
     5. -0000000-0000000
     6. -0000000-0000000
     7. +7022440+0374220

L20770  +3021100+4300000
L21000  +3521100+3621100
     1. +5121200+5221210
     2. +0021120+0121130
     3. +5321220+5421230
     4. +5521240+5621250
     5. +5720770-0000000

L21100  +0000000-0000000
     1. -0000000-0000000
     2. +7000000-0000000
     3. +7000000-0000000

L21200  +3621300+7721011
     1. +3621310+7721020
     2. +3621320+7721031
     3. +3621330+7721040
     4. +3621340+7721041
     5. +3621350+7721050

L21300  +0000000-0000510
     1. +0000000-0000520
     2. +0000000-0000530
     3. +0000000-0000540
     4. +0000000-0000550
     5. +0000000-0000560
```

TRANSFER COMMANDS TEST ROUTINE

Purpose

To test correct operation of computer in following and ignoring all transfer commands.

Operating Instructions

1. Enter and verify Transfer Commands Test routine with tape reader.
2. Place Operation switch in Continuous position.
3. Place Sense Switch B up and Sense Switches C and D down.
4. Depress Start 3 button (first routine loop stops automatically; loop can be repeated by again depressing Start 3 button - see Comments).
5. Place Sense Switches B, C, and D down.
6. Depress Start button (second routine loop routine runs continuously for a period of time, then automatically enters third routine loop; third loop runs continuously for a period of time, then halts automatically - see Comments).
7. Place Sense Switches B, C, and D up.
8. Depress Start button (fourth routine loop runs continuously for a period of time, then automatically enters fifth routine loop; after routine has run in fifth loop for a short while, halt computer by depressing Stop button - see Comments).

Comments

First routine loop tests the HTR command; computer should halt at location 0003.0 and display seven decimal points on visual readout; error in operation is indicated by display on visual readout of +0000000-0000040 or any operating sequence other than as specified.

Second routine loop tests TRA command; visual readout display should continuously change, starting at +400000-0000000 and counting down to +0010000-0000000; error is indicated by any operating sequence other than as specified or any of the halts and displays as given below:

Halt at Location	Display
0011.1	+0000000-0000141
0015.1	+0000000-0000151
0020.0	+0000000-0000200
0023.0	+0000000-0000230

Third Routine Loop Tests The Following:

Transfer on TSB, TSC, TSD with sense switches in down position.
Transfer on TMI with a negative number in the accumulator.
Transfer ignored on TPL with a negative number in the accumulator
Transfer ignored on TOV.
Transfer to Trapping Mode on a negative command.

Visual readout display should continuously change, starting at -40000000-0000000 and counting down to -0010000-0000000; after displaying last number computer should halt at location 0041.0; error is indicated by any operating sequence other than as specified or any of the halts and displays as given below:

Halt at Location	Display	Significance
0027.1	+0001000+7700271	Ignoring TSB command instead of executing it
0031.0	+0001000+7700310	Ignoring TSC command instead of executing it
0032.1	+0001000+7700321	Ignoring TSD command instead of executing it
0036.1	+0001000+7700361	Ignoring TMI command instead of executing it
0035.1	+0001000+7700351	Executing TPL command instead of ignoring it
0035.0	+0001000+7700350	Executing TOV command instead of ignoring it
0026.1	+0000000-0000261	Ignoring negative sign of command instead of entering Trapping Mode

Fourth Routine Loop Tests The Following:

Ignoring TSB, TSC, TSD with sense switches in up position.
Ignoring TMI and TZE with a positive, non-zero number inaccumulator
Transfer on TPL with a positive, non-zero number in accumulator.

Visual readout display should continuously change, starting at -0000000-0000000 and counting to +7770000-0000000; after displaying last number, computer should automatically enter fifth routine loop; error is indicated by any operating sequence other than as specified or any of the halts and displays as given below:

Halt at Location	Display	Significance
0041.0	+0001000+7700410	Executing TSB command instead of ignoring it
0041.1	+0001000+7700411	Executing TSC command instead of ignoring it
0042.0	+0001000+7700420	Executing TSD command instead of ignoring it
0043.0	+0001000+7700430	Executing TMI command instead of ignoring it
0045.0	+0001000+7700450	Executing TZE command instead of ignoring it
0045.1	+0001000+7700451	Ignoring TPL command instead of executing it

Fifth Routine Loop Tests The Following:

Transfer on TOV when overflow occurs.
Transfer on TZE with a zero in the accumulator.

Overflow error indicator on control console should flash on and off; visual readout should display a positive zero; error is indicated by any operating sequence other than as specified or any of the halts and displays as given below:

Halt at Location	Display	Significance
0051.0	+0001000+7700510	Ignoring TOV command instead of executing it
0053.1	+0001000+7700531	Ignoring TZE command instead of executing it

Transfer Commands Test Routine Printout

```
L00000  +1501000+4200071
     1. +0300600+5000060
     2. +3600070+5700070
     3. +5400100+3600611
     4. +7700030+3600620
     5. +7700030+7700030
     6. +0001000+5700271
     7. +0001000+7700000
L00100  +0000650+6001010
     1. +0001010+5700130
     2. +3600720+7700111
     3. +1501000+0100640
     4. +4200151+3601010
     5. +0001000+5700171
     6. +3600730+7700151
     7. +7700151+0300630
L00200  +5700211+3600740
     1. +7700200+6001010
     2. +5000270+3300360
     3. +5700110+3600750
     4. +7700230+0000000
     5. -0000000-0000000
     6. +0200650-3600600
     7. +5700260+5400310
L00300  +1501000+5700001
     1. +5500321+1501000
     2. +5700001+5600340
     3. +1501000+5700001
     4. +3601000+0100630
     5. +5300000+5200000
     6. +5000401+5100261
     7. +1501000+5700001
L00400  +7700371+7700410
     1. +5400000+5500000
     2. +5600000+0100630
     3. +5300471+5100000
     4. +3001030+3601030
     5. +5000000+5200410
     6. +1501000+5700001
     7. -0000000+0000760
L00500  +0100650+0100650
     1. +5300521+1501000
     2. +5700001+6001040
     3. +3601040+5000500
     4. +1501000+5700001
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L00600  +0000000-0000261
     1. +6314630-6317000
     2. +0000000-0000040
     3. +0010000-0000000
     4. +0000000-0000060
     5. +4000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L00700  -0000000-0000000
     1. -0000000-0000000
     2. +0000000-0000111
     3. +0000000-0000151
     4. +0000000-0000200
     5. +0000000-0000230
     6. +0000000-0000000
     7. -0000000-0000000
```

GENERAL MACHINE TEST ROUTINE I

PURPOSE

To test for proper execution of miscellaneous commands in RECOMP II instruction repertoire, particularly the logic commands.

OPERATING INSTRUCTIONS

1. Enter and verify General Machine Test Routine 1 with tape reader.
2. Prepare typewriter for operation.
3. Depress Start 3 button (computer should operate about ten minutes, then halt at location 0004.0 - see Comments).
4. When last display is observed, depress Stop button on control console.

COMMENTS

Following numbers should be displayed in order as computer executes program:

+2525250+2525250
-0025250-0025250
+0000000-0000000
+0000000-0001650
+2525250+2525250
+2525250+2525240
+2525250+2525200
etc, losing one bit at a time and ending at
+0000000-0000000

Error is indicated by typeout of first location of the test which failed or any operating sequence other than as specified.

The General Machine Test Routine I Printout is shown on the subsequent pages.

3½ pages here

```
L00000  +1503770+3303700
    1.  +0303710+5200031
    2.  +0103710+5703000
    3.  +5700040+7700031
    4.  +0002700+6004000
    5.  +0004000+6004010
    6.  +4000000+0304010
    7.  +4000000+3604010
L00100  +5000110-3000000
    1.  +0000050+0100150
    2.  +6000050+4200061
    3.  +4200071+0300160
    4.  +5100050+5700200
    5.  +0000010-0000010
    6.  +0004771-0000000
    7.  -0000000-0000000
L00200  +0002700+6004000
    1.  +0004000+1504010
    2.  +1500330+0304010
    3.  +5000241-3000000
    4.  +4000000+3604010
    5.  +0000210+0100310
    6.  +6000210+4200221
    7.  +4200241+0300320
L00300  +5100210+5700400
    1.  +0000010-0000010
    2.  +0004770-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0077771-0077771
L00400  +3300360+6004000
    1.  +0000400+0100550
    2.  +4200401+0300570
    3.  +5100400+0002700
    4.  +3300370+6004000
    5.  +0004000+4204011
    6.  +4204010+0304011
    7.  +5000500-3000000
L00500  +3604010+0000460
    1.  +4200450+0100550
    2.  +6000460+4200500
    3.  +4200451+0300560
    4.  +5100450+5700600
    5.  +0000010-0000010
    6.  +4204770-0000000
    7.  +3300370+6004770

L00600  +4000000+6402700
    1.  +0077600+0302700
    2.  +5000630-3000000
    3.  +0000610+0100730
    4.  +6000610+0300740
    5.  +5100610+4000000
    6.  +0000750+6000610
    7.  +6504000+6404000
L00700  +0000670+0100760
    1.  +6000670+0300770
    2.  +5100610+5701000
    3.  +0000010-0000010
    4.  +0077700-0000000
    5.  +0077600+0302700
    6.  +0000100-0000100
    7.  +6512000-0000000
L01000  +4000000+6602700
    1.  +0077700+0302700
    2.  +5001030-3000000
    3.  +0001010+0101130
    4.  +6001010+0301140
    5.  +5101010+4000000
    6.  +0001150+6001010
    7.  +6704000+6604000
L01100  +0001070+0101160
    1.  +6001070+0301170
    2.  +5101010+5701200
    3.  +0000010-0000010
    4.  +0100000-0000000
    5.  +0077700+0302700
    6.  +0000100-0000100
    7.  +6712000-0000000
L01200  +3002700+3504000
    1.  +3004000+0302700
    2.  +5001230-3000000
    3.  +4300000+0302710
    4.  +5001250-3000000
    5.  +0001210+0101300
    6.  +6001210+0301310
    7.  +5101210+5701400
L01300  +0000020-0000000
    1.  +3004770-0000000
    2.  -0000000-0000000
    3.  -0000000-0000000
    4.  -0000000-0000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
```

```
L01400  +3002700+4300000
     1. +4300000+0302700
     2. +5001430-3000000
     3. +4300000+0302710
     4. +5001450-3000000
     5. +6004000+3604000
     6. +0001510+0301520
     7. +6001510+5201400
L01500  +0001520+5701650
     1. +7700000-0000000
     2. +0100000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L01600  +1577700+6004000
     1. +0301660+5001621
     2. -3000000+3604000
     3. +0001670+0301700
     4. +6001670+4000000
     5. +5201600+5702000
     6. +0000000-0001650
     7. +7700000-0000000
L01700  +0100000-0000000
     1. -0000000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02000  +0002070+3302700
     1. +3302700+0302700
     2. +5002030-3000000
     3. +3302700+5002041
     4. -3000000+0002100
     5. +0302110+6002100
     6. +5202000+5702200
     7. +7777771+7777771
L02100  +7700000-0000000
     1. +0100000-0000000
     2. -0000000-0000000
     3. -0000000-0000000
     4. -0000000-0000000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02200  +0002700+3302420
     1. +6004000+0002700
     2. +4000010+4100010
     3. +0304000+5002241
     4. -3000000+3604000
     5. +0002400+0302430
     6. +6002400+5202211
     7. +0002410+6002400
L02300  +0002220+0102440
     1. +6002220+6002340
     2. +0302450+5202500
     3. +0002420+4000000
     4. +4000010+4100010
     5. +6002420+5702200
     6. -0000000-0000000
     7. -0000000-0000000
L02400  +7700000-0000000
     1. +7700000-0000000
     2. +7777771+7777770
     3. +0100000-0000000
     4. +0000010-0000010
     5. +4000540-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L02500  +3002700+4500000
     1. +3504000+4000010
     2. +4500000+4300000
     3. +0304010+0102650
     4. +5002550-3000000
     5. +4300000+0304000
     6. +4000000+4000010
     7. +5002600-3000000
L02600  +0002510+0102660
     1. +6002510+4202561
     2. +4000010+4202651
     3. +4100010+0302670
     4. +5102500+7700040
     5. +0000000-0000001
     6. +0000000-0000010
     7. +3504000+4000470
L02700  +2525250+2525250
     1. +2525250+2525250
     2. +2525250+2525250
     3. +2525250+2525250
     4. +2525250+2525250
     5. +2525250+2525250
     6. +2525250+2525250
     7. +2525250+2525250
```

```
L03000  +4000030+0103720
    1.  +4203041+0003730
    2.  +5703200+7200060
    3.  +7200140+7200040
    4.  +4000000+0000000
    5.  +5703200+0003040
    6.  +0103740+4203041
    7.  +3303740+5003101
L03100  +5703041+0003750
    1.  +5703200+7200160
    2.  +7200040+0003041
    3.  +4000000+0103760
    4.  +4203151+4000000
    5.  +4000000+1213361
    6.  +7703160-0000000
    7.  +7703160-0000000
L03200  +1503770+0103610
    1.  +4203321+0003770
    2.  +4000120+4100010
    3.  +4203250+3303600
    4.  +4100050+4000000
    5.  +7200040+5003261
    6.  +5703230+0003770
    7.  +4100010+4203311
L03300  +4100170+4203310
    1.  +7202221+7211300
    2.  +4000000+5703111
    3.  +6003210+6403021
    4.  +0003330+0103430
    5.  +6003330+5703111
    6.  +3677600+7703000
    7.  +3677600+7703021
L03400  +0000010-0000010
    1.  +0077071-0077071
    2.  +0000010-0000000
    3.  +0000000-0000010
    4.  +6004160+5703171
    5.  +3677610+7703060
    6.  +3604200+7703000
    7.  +0077610+5777610
L03500  +0077071-0077071
    1.  +6004100+0077620
    2.  +6004110+0077630
    3.  +6004120+0077640
    4.  +6004130+0077650
    5.  +6004140+0077660
    6.  +6004150+0077670
    7.  +6004160+5703171
```

```
L03600  +0000001+7777771
    1.  +0000000-0000001
    2.  +6004100+0077620
    3.  +6004110+0077630
    4.  +6004120+0077640
    5.  +6004130+0077650
    6.  +6004140+0077660
    7.  +6004150+0077670
L03700  +0000000-0077600
    1.  +0000000-0002410
    2.  +0000000-0013000
    3.  -0370250-5302420
    4.  +0000000-0000010
    5.  -0200221+0044540
    6.  +0000000-0000261
    7.  -0200221+0044540
```

```
L13000  -0054140-2000000
    1.  -0000000-0000000
    2.  -0050761-2054140
    3.  -0120220-0000000
    4.  -0054011-2000000
    5.  -0000000-0000000
    6.  -2164110+5547620
    7.  -0163310-2000000
L13100  -0164170+5547620
    1.  -0163370-2000000
    2.  -0151300-2000000
    3.  -0000000-0000000
    4.  -0350650-2000000
    5.  -0000000-0000000
    6.  -0050761-2017270
    7.  -0240660+5011000
L13200  -0017300-2000000
    1.  -0000000-0000000
    2.  -0032421+5547620
    3.  -0034421-2000000
    4.  -0153160-2000000
    5.  -0000000-0000000
    6.  -0000000-0000000
    7.  -0000000-0000000
```

General Machine Test Routine I Printout (Cont)

```
L13300  +0000230-0360000
     1. -0000000-0000000
     2. +0002030-0360000
     3. -0000000-0000000
     4. +0004030-0360000
     5. -0000000-0000000
     6. +0006030-0360000
     7. -0000000-0000000
L13400  +0020030-0360000
     1. -0000000-0000000
     2. +0022030-0360000
     3. -0000000-0000000
     4. +0024030-0360000
     5. -0000000-0000000
     6. +0026030-0360000
     7. -0000000-0000000
L13500  +0040030-0360000
     1. -0000000-0000000
     2. +0042030-0360000
     3. -0000000-0000000
     4. +0045030-0360000
     5. -0000000-0000000
     6. -0000000-0000000
     7. -0000000-0000000
L13600  -0000000-0000000
```

GENERAL MACHINE TEST II

Purpose

To test for proper operation of all arithmetical commands.

Operating Instructions

Explanations of the arithmetic routines located in the memory are discussed in the following table.

GENERAL MACHINE TEST II

Sense Switch	Remarks
B	Up; display.
B	Down; type.
C	Up; leapfrog.
C	Down; New constants after running 12 tests.

Location	Remarks
00000	Beginning of type-out linkage.
00010	Initializing: depress START 1.
00020 through 00070	Type-out linkage.
00200	Pseudo-random constants generator: Constants are generated by routine in location 0034-0045, and modified and stored by location 0012-0031, given proper format to prevent overflow, i.e., constants used in square root tests are positive.
00320	If SENSE switch "B" is down, constants will be typed out. (L00331 is exit.)
00340	Constants generator: constants are stored as follows: Fixed point, for tests other than square root (test 8), L7770 and 7771 (i.e. two constants).

	Floating point, except square root: L7772 and 7773 (mantissa and exponent, first constant); L7774 and 7775 (mantissa and exponent, second constant). Square root, always positive. Fixed point, L7654; floating point, L7655 and 7656.
00460 through 00570 and 00740 through 00760	Types out constants in command format and initializes the alphanumeric type-out loop for labeling constants.
00600 through 00730	BCD labels used to identify tests in error type-out or display.
01000 through 01070	Test 1: Note that tests are "leapfrogged" through the memory, so that this test will subsequently be located at location 02000, 03000, 04000, etc, to 74000, afterwhich it will be returned to 01000.
01100 through 01170	Test 2: Also note that in all fixed point tests the accumulator should contain a + or - zero immediately before the command " +5077000." If not, the program jumps to 77030 and types or displays the error.
01200 through 01270	Test 3
01300 through 01370	Test 4
01400 through 01470	Test 5
01500 through 01570	Test 6
01600	Instruction beginning at L76520 will cause a jump to Test 7, which begins at sector 00 of the next channel.
01610 through 01760	Contains labels to be typed out in alpha for the constants generator routine.

02000 through 02070	Test 7
02100 through 02170	Test 8
02200 through 02270	Test 9. Tests 9 through 12 are floating point tests. In these cases it is possible to obtain a "1" in bit position 1, instead of a zero, due to normalization of round-off. If this occurs, the program will test the mantissa; anything but a normalized "1" will cause indication of a mantissa error. If it is a normalized "1," the exponent will be tested. It should differ from the exponent of the first floating point constant by exactly 45_8 or 37_{10} as a result of denormalization and normalization. Different values will cause the indication of floating point exponent errors.
02300 through 02370	Test 10
02400 through 02470	Test 11
02500 through 02570	Test 12
02600	Jump to L7600, which links to the constants generator when SENSE switch "C" is down and to the leapfrog loop when that switch is up.
76000	If "C" is up, jump to constants generator. If not, continue through leapfrog.

This page intentionally left blank

General Machine Test Routine II Printout

```
L00000  +1577670+5700020
     1. +5700111-0000000
     2. +6577600+6400000
     3. +5777631+4277651
     4. +3300050+5077651
     5. +4277660+0000001
     6. +4000240+4277261
     7. +0077650+5777240
L00100  +3377540+0376710
     1. +4200331+5700340
     2. +6077700+5700340
     3. +6077710+0177700
     4. +5300150+5700160
     5. +0377710+6077710
     6. +5700340+4500000
     7. +6077720+5700340
L00200  +3376640+6077730
     1. +5700340+4500000
     2. +6077740+5700340
     3. +3376640+6077750
     4. +5700340+6076540
     5. +5200261+0276540
     6. +6076540+5700340
     7. +6076550+5200310
L00300  +0276550+6076550
     1. +0077730+6076560
     2. +5400460+7200370
     3. +7200100+5701000
     4. +1577650+0100770
     5. +4200451+0076570
     6. +1176600+3376610
     7. +6076620+0076600
L00400  +6076570+4300000
     1. +4100250+0176620
     2. +6077630+6076600
     3. +3376650+5000450
     4. +0077630+5700451
     5. +0277630+5700311
     6. -3001610-3001630
     7. +1277700-3001650
L00500  +1277710-3001670
     1. -3001710+1277720
     2. -3001730+1277730
     3. -3001670-3001710
     4. +1277740-3001730
     5. +1277750-3001750
     6. -3001630+1276540
     7. -3001670+5700740

L00600  +5241571+0000000
     1. +5242571+0000000
     2. +5243571+0000000
     3. +5244571+0000000
     4. +5245571+0000000
     5. +5246571+0000000
     6. +5247571+0000000
     7. +5250571+0000000
L00700  +5251571+0000000
     1. +5020571+0000000
     2. +5021571+0000000
     3. +5022571+0000000
     4. -3001710+1276550
     5. -3001730+1276560
     6. +7200370+5700330
     7. +0000000-0000001
L01000  +0077700+0177710
     1. +0377710+0377700
     2. +5077000+6077600
     3. +0000600+5777030
     4. +0010000+0010000
     5. +0010000+0010000
     6. +0000600+5777030
     7. +0000600+5777030
L01100  +0277700+0177700
     1. +5077000+6077600
     2. +0000610+5777030
     3. +0000610+5777030
     4. +0000610+5777030
     5. +0000610+5777030
     6. +0000610+5777030
     7. +0000610+5777030
L01200  +0077700+1177710
     1. +6077760+2277710
     2. +0377700+5077000
     3. +6077600+0000620
     4. +5777030-0000000
     5. +5777030-0000000
     6. +5777030-0000000
     7. +5777030-0000000
L01300  +0077700+1377710
     1. +0377760+3376670
     2. +5077000+6077600
     3. +0000630+5777030
     4. +0000630+5777030
     5. +0000630+5777030
     6. +0000630+5777030
     7. +0000630+5777030
```

General Machine Test Routine II Printout (Cont)

```
L01400  +0077700+1177710
    1.  +2377710+0377700
    2.  +5077000+6077600
    3.  +0000640+5777030
    4.  +0000640+5777030
    5.  +0000640+5777030
    6.  +0000640+5777030
    7.  +0000640+5777030
L01500  +0077700+1177710
    1.  +2077710+6077770
    2.  +3076670+0077760
    3.  +2277710+0377770
    4.  +5077000+6077600
    5.  +0000650+5777030
    6.  +0000650+5777030
    7.  +0000650+5777030
L01600  +5776520-0000000
    1.  -0000171-7303021
    2.  -0000100-1544021
    3.  -0000040-6467220
    4.  -0000130+0332040
    5.  -0000171-4332040
    6.  -0000000-0000000
    7.  -0000171-4154500
L01700  -0000020+3206640
    1.  -0000171+6033151
    2.  -0000040-0000000
    3.  -0000171-4332171
    4.  -0000001+6666640
    5.  -0000171-4055620
    6.  -0000050+4304020
    7.  +5777000+0000000
L02000  +0077700+1177710
    1.  +2177710+6077770
    2.  +3076670+0077760
    3.  +2377710+0377770
    4.  +5077000+6077600
    5.  +0000660+5777030
    6.  +0000600+5777030
    7.  +0000600+5777030
L02100  +2576540+6077770
    1.  +1377770+0376540
    2.  +3376670+5077000
    3.  +6077600+0000670
    4.  +5777030+0000000
    5.  +0000700+5777340
    6.  +0000700+5777340
    7.  +0000700+5777340
L02200  +3077720+0477720
    1.  +0677720+0677720
    2.  +5077000+3577620
    3.  +0000700+5777340
    4.  +0000700+5777340
    5.  +0000700+5777340
    6.  +0000700+5777340
    7.  +0000700+5777340
L02300  +3477720+0477720
    1.  +5077000+3577620
    2.  +0000710+5777340
    3.  +0000710+5777340
    4.  +0000710+5777340
    5.  +0000710+5777340
    6.  +0000710+5777340
    7.  +0000710+5777340
L02400  +3077720+0777740
    1.  +0577740+0677720
    2.  +5077000+3577620
    3.  +0000720+5777340
    4.  +0000720+5777340
    5.  +0000720+5777340
    6.  +0000720+5777340
    7.  +0000720+5777340
L02500  +4476550+3577760
    1.  +0777760+0676550
    2.  +5077000+3577620
    3.  +0000730+5777340
    4.  +0000730+5777340
    5.  +0000730+5777340
    6.  +0000730+5777340
    7.  +0000730+5777340
L02600  +5776000-0000000
    1.  -0000171+6033020
    2.  -0000000-0000000
    3.  -0000171-0755420
    4.  -0000000-0000000
    5.  +4614010+2446550
    6.  +6272540+6063460
    7.  +0370730+0104620
L02700  +0513711-5406270
    1.  +0672420-7263110
    2.  +1116021+7104130
    3.  +1422561-4132710
    4.  +2030760-5443660
    5.  +2566500-7332350
    6.  +3510651-4443210
    7.  +1666341+3331530
```

```
L76000  +1577600+5500100
     1. +0076070+3377540
     2. +0377550+5176070
     3. +0076070+0376660
     4. +4276071+0076130
     5. +0376660+4276131
     6. +0277560+6077570
     7. +6402000+6503000
L76100  +0076070+0176450
     1. +6076070+3376720
     2. +5076130+5776070
     3. +6401000+6502000
     4. +0076130+0176450
     5. +6076130+3376720
     6. +5076170+5776130
     7. +0077570+5276240
L76200  +0076070+0376660
     1. +4276070+0076130
     2. +0376660+4276130
     3. +0077560+6077570
     4. +0076130+3376460
     5. +4000240+4200331
     6. +0076130+3376470
     7. +4100010+6077630
L76300  +0076130+3376500
     1. +0177630+0176510
     2. +6077630+0076070
     3. +3376470+4100010
     4. +6077640+0076070
     5. +3376500+0177640
     6. +0176510+6077640
     7. +3677631+5476410
L76400  +7700331+5702000
     1. -3076730+1277631
     2. -3076750+1277641
     3. +5700321-0000000
     4. +4276401+5776401
     5. +0000100-0000100
     6. +0077000-0000000
     7. +0070000-0000000
L76500  +0007000-0000000
     1. +5000531-0000000
     2. +1577600+3377540
     3. +0176710+5776440
     4. +6723150+5377721
     5. +5522311-6663471
     6. -0000000-0000020
     7. +5547170+1644760
L76600  +0050040-5327741
     1. +0000000+7777771
     2. +0000000-5327741
     3. +3105231-0024020
     4. +0000000-0000371
     5. +0000100-0000000
     6. +0074000-0074000
     7. +7777771+7777770
```

```
L76700  +0000000-0000000
     1. +0000000-0001000
     2. +0000700-0000000
     3. -0000171-7240660
     4. -0000060-0626620
     5. -0000171+0301070
     6. -0000120-1541000
     7. +6457521+1601530
L77000  +1577770+3377130
     1. +0176450+4277021
     2. +3677770+5702500
     3. +1577610+3377130
     4. +0176450+4277111
     5. +5477140+0077610
     6. +0177120+6077610
     7. +3677611+7777100
L77100  +3677600+7777111
     1. +7200100+5777500
     2. +1000030+6317000
     3. +0000000-0077700
     4. -3077170-3077210
     5. +1277611+1277600
     6. +7200370+5777110
     7. -0000171-0522540
L77200  -0000050-2063020
     1. -0000100-0454020
     2. +0000371+7777771
     3. +0000000+5700001
     4. +6477600+5777250
     5. +0100770+4277231
     6. +0000000+3000000
     7. +4100010+4277300
L77300  +7200331+4100050
     1. +3377220+5077321
     2. +5777271+4300001
     3. +5077231+5777270
     4. +1577660+3377130
     5. +0176450+4277501
     6. +3077620+3377470
     7. +5077420+6077600
L77400  +5077420+6077600
     1. +0077660+5777030
     2. +4300000+0177520
     3. +0377730+5077501
     4. +6077600+0077530
     5. +5477530+0077660
     6. +5777030-0000000
     7. +3777771+7777771
L77500  +4000000+5702300
     1. -3002610+5777410
     2. +0000000-0000221
     3. -3002630+5777451
     4. +0000000-0077000
     5. +0000000-0076000
     6. +0075000-0075000
     7. +7777771+7777771
```

TYPE CHARACTER AND PUNCH/TYPE CHARACTER TEST ROUTINE

Purpose

To check proper operation of typewriter and punch under computer control with Type Character, and Punch and Type Character commands.

Operating Instructions

1. Fill and verify Fill Memory with Random Numbers routine.
2. Set location counter to 7757.0 and depress Start button and allow routine entered in Step 1 to run until it stops.
3. Fill and verify Type Character and Punch/Type Character Test routine.
4. Place Sense Switch B up.
5. Prepare typewriter for operation and depress Letters Shift key on typewriter.
6. Turn Computer-Manual switch on punch control panel to Computer position.
7. Depress Start 1 button (unless there is a malfunction, the computer halts automatically after the typewriter has typed and performed all functions from code 00 through 37_8, with 16 operations of each character in Letters Shift mode).
8. Depress Figures Shift key on typewriter..
9. Depress Start 1 button (unless there is a malfunction, the computer halts automatically after the typewriter has typed and performed all functions from code 00 through 37_8, with 16 operations of each code, in Figures Shift mode).
10. Prepare punch for operation.
11. Place Sense Switch C up.
12. Depress Start 2 button (unless there is a malfunction, the computer halts after the typewriter and punch have recorded all characters and functions from code 00 through 37_8, with 16 operations of each code).
13. Remove punched tape from punch and visually check to confirm that all codes have been punched correctly and in proper sequence.

Comments

To cause the typewriter to continue typing, place Sense Switch B down; to cause the typewriter and punch to continue typing and punching, place Sense Switch C down.

The octal character of each character is displayed on the visual readout as the character is typed and punched.

```
L00010  +5710440-0000000
     2. +5710510-0000000

L10000  +7200000+7200000
     1. +7200000+7200000
     2. +7200000+7200000
     3. +7200000+7200000
     4. +7200000+7200000
     5. +7200000+7200000
     6. +7200000+7200000
     7. +7200000+7200000
L10100  +7200100+0010370
     1. +6010000+6010010
     2. +6010020+6010030
     3. +6010040+6010050
     4. +6010060+6010070
     5. +0110400+6010370
     6. +3610000+5410000  "B" UP
     7. +5710620-0000000  TO HALT
L10200  +7600000+7600000
     1. +7600000+7600000
     2. +7600000+7600000
     3. +7600000+7600000
     4. +7600000+7600000
     5. +7600000+7600000
     6. +7600000+7600000
     7. +7600000+7600000
L10300  +7200100+0010410
     1. +6010200+6010210
     2. +6010220+6010230
     3. +6010240+6010250
     4. +6010260+6010270
     5. +0110400+6010410
     6. +3610200+5710660
     7. +7200010+7200010
L10400  +0000010-0000010
     1. +7600010+7600010
     2. +7200000+7200000
     3. +7600000+7600000
     4. +0010420+6010000
     5. +6010010+6010020
     6. +6010030+6010040
     7. +6010050+6010060

L10500  +6010070+5710560
     1. +0010430+6010200
     2. +6010210+6010220
     3. +6010230+6010240
     4. +6010250+6010260
     5. +6010270+5710600
     6. +0010400+4210370
     7. +4210371+5710000
L10600  +0010400+4210410
     1. +4210411+5710200
     2. +0010000+0310650
     3. +5010640+5710000
     4. +7700010-0000000
     5. +7200400+7200400
     6. +5510200+5300000
     7. +0010200+0310720
L10700  +5010710+5710200
     1. +7700020-0000000
     2. +7600400+7600400

L 77570  ≠64 ,00≠6677700
L 77600  ≠5477640≠5577670
     1. ≠7200000≠0077610
     2. ≠0177720≠4277610
     3. ≠3677610≠5777610
     4. ≠7600000≠0077640
     5. ≠0177720≠4277640
     6. ≠3677640≠5777640
     7. ≠7400000≠0077670
L77700  ≠0177720≠4277670
     1. ≠3677670≠5777670
     2. ≠0000010000000000
```

Fill Memory With Random Numbers Test Routine Printout

```
L77570  +6477600+6677700        START: L 77570
L77600  +0077710+1177700
    1.  +4100010+6077710
    2.  +4300000+6001010
    3.  +0077620+0177670
    4.  +6077620+3677620
    5.  +0077620+0377730
    6.  +5077740+5777600
    7.  +0000000-0000010
L77700  -5252521-5252521
    1.  -5260110-0664160
    2.  +2525250-0000000
    3.  +4300000+6077560
    4.  +7777600-0000000
```

TYPE CHARACTER, TYPE/PUNCH CHARACTER, AND PUNCH CHARACTER CONSECUTIVELY TEST ROUTINE

Purpose

To check operation of the typewriter and punch under computer control with Type Character, Type and Punch Character, and Punch Character commands.

Operating Instructions

1. Fill and verify the Type Character, Type and Punch Character, and Punch Character Consecutively Test routine.
2. Prepare typewriter for operation and depress Letters Shift key on typewriter.
3. Place Sense Switches B and C up (causes typing only).
4. Set location counter to 7757.0.
5. Depress Start button (unless there is a malfunction, the routine will continue to run until stopped manually; the routine should be allowed to run at least one minute).
6. Prepare punch for operation.
7. Set location counter to 7760.0.
8. Place Sense Switch B down (causes typing and punching).
9. Depress Start button (unless there is a malfunction, the routine will run until stopped manually; the routine should be allowed to run at least one minute).
10. Remove punched tape from punch and visually inspect to determine if punched correctly (see Comments below).
11. Place Sense Switches B up and C down (causes punching only).
12. Set location counter to 7760.0.
13. Depress Start button (unless there is a malfunction, the routine will run until stopped manually; the routine should be allowed to run at least one minute).
14. Remove punched tape from punch and visually inspect to determine if punched correctly (see Comments below).

Comments

On type only, the typewriter should record the following:

EA SIU
DRJNFCKTZLWHYPQOBG. /;EA SIU
DRJNFCKTZLWHYPQOBG. /;EA SIU
etc.

On punch and type, the same will be recorded as on type only (the binary configurations in the tape corresponding to the octal numbers 00, 01, 02, etc. through 36, except 33 and 37 which are not punched).

On punch only, the same will be recorded by the punch as on punch and type.

Type, Punch/Type, Punch Characters Consecutively
Test Routine Printout

L 77570	+6477600+6677700
L 77600	+5477640+5577670
1.	+7200000+0077610
2.	+0177720+4277610
3.	+3677610+5777610
4.	+7600000+0077640
5.	+0177720+4277640
6.	+3677640+5777640
7.	+7400000+0077670
L 77700	+0177720+4277670
1.	+3677670+5777670
2.	+000001000000000

TAB TEST AND TYPE WORD WITH TAB TEST ROUTINE

Purpose

To check the tab operation of the typewriter under computer control with the Type Word command.

Operating Instructions

1. Prepare typewriter for operation and set Tab Over-ride switch on typewriter to permit tabulating.
2. Set tabs on typewriter to cause three tab operations, the first of 30 to 40 spaces from the left-hand margin and the remaining two any place before the right-hand margin and carriage return.
3. Enter and verify Tab Test and Type Word with Tab Test routine.
4. Depress Start 3 button.
5. After typewriter has completed approximately 12 operations of three tabs and a carriage return each, depress Stop button on computer.

6. Set tabs on typewriter to cause three tabulations across the page, each sufficiently long to contain one computer word in command format (16 characters).

7. Depress Start 2 button.

8. After several minutes of operations, depress Stop button.

Comments

In Step 7 typewriter records whatever is in memory, beginning with channel 00, in three columns across the page, then performs a carriage return and repeats the procedure, each time from different memory locations.

Tab Test And Type Word With Tab Test Routine Printout

```
LOC. 1100: START TAB TEST

LOC. 12000: START TYPE WORD
            WITH TAB.

L00020  +5712000-0000000
    3.  +5711000-0000000

L11000  +7200330+7200100
    1.  +7200100+7200100
    2.  +7200370+7200100
    3.  +5711000-0000000

L12000  +7200330+1200000
    1.  +7200100+1200010
    2.  +7200100+1200020
    3.  +7200370+7200100
    4.  +0012000+0112110
    5.  +4212001+0012010
    6.  +0112110+4212011
    7.  +0012020+0112110
L12100  +4212021+5712120
    1.  +0000000-0000030
    2.  +3612020+5712000
```

ALPHA DUMP TEST ROUTINE

Purpose

To check operation of the tape punch by recording the contents of specified locations in memory in alphanumeric format under computer control.

Operating Instructions

1. Enter and verify Alpha Dump Test routine.
2. Set location counter to 5000.0.

3. Enter following command from control console keyboard:

    +0000000+0050000 or +00XXXX0+00YYYY0
    where X = beginning location from which dump is to be made
    Y = location (plus 1) at which dump is to end
    (X and Y locations can be 0000 to 5000_8 and 5042_8 to 7756_8).

4. Turn Compute-Manual switch on punch to Manual position.
5. Depress Tape Advance button on punch until at least 18 inches of leader have been punched.
6. Typewriter punch the following:

    L 00000 Carriage Return
    F

7. Turn Compute-Manual switch to Compute position.
8. Check to make certain location counter is set at 5001.0.
9. If display is desired, place Sense Switch C up (see Comments below).
10. Depress Start button (routine runs until it halts automatically).
11. Depress Tape Advance button to run off at least 12 inches of leader and remove tape from punch.
12. Near end of punched information, cut out 15 sprocket holes with small knife.
13. Glue ends of tape together, with three sprocket holes over lapping.
14. Place tape in reader and depress Verify button on reader.
15. Set location counter to 2000.0.
16. Fill location 2000.0 with zeros from console keyboard.
17. Re-run tape in Verify mode (Verification process should stop and Verify Error neon on control console should light when verification attempt fails at location 2000.0).

Comments

Console readout display indicates location being dumped in first address portion of readout panel.

START L50010 Alpha Dump Test Routine

IN L50000 FILL THE FOLLOWING:

+00XXXX0+00YYYY0

X = START DUMP AT LOCATION

Y = END DUMP LOCATION PLUS 1

SENSE SWITCH C:

UP = DISPLAY L7760

DOWN = NO DISPLAY

```
L50000  +0050000+0050410
     1. +0050000+4250100
     2. +4100240+4250070
     3. +6450100+6350200
     4. +5777600+0077600
     5. +0350070+5050340
     6. +5577600+5750330
     7. +0050410+5050320
L50100  +0050000+6050320
     1. +4000160+5177621
     2. +0150310+4277700
     3. +4277731+0050320
     4. +4100010+4277730
     5. +4277761+4000050
     6. +4277720+4277751
     7. +4000050+4277710
L50200  +7400000+4277741
     1. +7400000+0077600
     2. +7400000+0150300
     3. +7400000+7400000
     4. +4277600+7400000
     5. +5300200+7400000
     6. +5300010+7400000
     7. +7400100+5750041
L50300  +0000010-0000000
     1. +0000200-0000000
     2. +7400000+7750000
     3. +3677600+5777600
     4. +7400000+7400000
     5. +7400000+7400000
     6. +7400000+7400000
     7. +7400000+7400000
L50400  +7400000+7750000
```

COMMAND MEMORY DUMP (PUNCH) TEST ROUTINE

Purpose

To check operation of punch by recording contents of memory on paper tape in command format under computer control.

Operating Instructions

1. Enter Command Memory Dump (Punch) Test routine.
2. Set location counter to 7757.0.
3. Depress Start button (routine will halt computer after three channels of information have been punched).
4. Remove punched tape and verify through reader.

Command Memory Dump (Punch) Test Routine Printout

```
L77570  +6477600+6677700
L77600  +7400220+7400260
     1. +7400260+7400260
     2. +7400260+7400260
     3. +7400100+7400160
     4. +1400000+7400100
     5. +0077640+0177700
     6. +4277640+0377710
     7. +5077720+5777640
L77700  +0000010-0000000
     1. +1403000+7400100
     2. +7400000+7400000
     3. +7400000+7400000
     4. +7400000+7400000
     5. +7400000+7400000
     6. +7400000+7400000
L77770  +7400240+7777500
```

ENTER NUMBERS THROUGH PHOTOREADER TEST ROUTINE

Purpose

To check proper entry of numbers with paper tape reader and correct type-out of the same numbers in both command and decimal formats.

Operating Instructions

1. Fill and verify Enter Numbers Through Photoreader Test routine.
2. Fill and verify Type Word (Command and Decimal) Test routine.
3. Depress Start button (typewriter should type out results as listed below).

```
L70000

N+12345678901.
+98765432109.
+23456789.
+144.
+275
-.52678975
+654389.
+2
-76555
+1.
+2.
+12.
-123.
+123.
+1234.
+12345.
+0.
```

TYPED RESULTS OF WHICH SHOULD BE:

```
 +0133761+4070321          05
+1337670-0712261           2.
+0000130-5730121           0002.                .80000000
+0000000-0001100           00000001.666666
+2146311-3146311           4666666666.                    16
-4155561+0554071           -86.                           16.000013
+0000000+1770321           000013
+1463141+1463150           33333333.3
-6077540+2665350           -.3
+0000000-0000001           0000000000000000
+0000000-0000010           0000000000000000
+0000000-0000060           0000000000000000
-0000000-0000751           -0000000000000000
+0000000-0000751           0000000000000000
+0000000-0011510           0000000900000006
+0000000-0140341           0000006000000000
```

DOUBLE-PUNCHING TEST ROUTINE

Purpose

To test operation of punch against double-punching with Punch Character and Punch and Type Character commands.

Operating Instructions

1. Enter Double-Punching Test routine from control console keyboard:

```
L  40000 Enter
C  +7400000+7400370 Enter
   +5740000+0000000 Enter
```

2. Set location counter to 40000.
3. Depress Start button.
4. After routine has run approximately three minutes, depress Stop button.
5. Inspect tape for double punches (see Comments below).
6. Change routine as follows:

```
L  40000 Enter
C  +7600000+7600370 Enter
```

7. Depress Start button.
8. After routine has run approximately three minutes, depress Stop button.
9. Inspect tape for double punches (see Comments below).

Comments

With both the punch character and punch and type character commands, the punch should record a blank code (sprocket hole only) followed by a Letters Shift code (sprocket hole and five character code holes), then continue to repeat this configuration (blank, Letters Shift, blank, Letters Shift, etc.).

INPUT-OUTPUT AND MEMORY TEST ROUTINE

Purpose

To (1) check operation of paper tape reader, tape punch, and typewriter, and (2) simultaneously check the memory for incorrect recording and reading and memory changing information.

Operating Instructions

1. Enter and verify Input-Output and Memory Test routine.
2. Punch leader sufficiently long to loop back to and insert in photoreader (leave tape slack).
3. Turn Compute-Manual switch on punch to Compute position.
4. Set location counter to 2516.0.
5. Depress Start button (routine runs until malfunction occurs or it has operated through entire memory).

Comments

Routine causes punching leader of first 17 words of routine; after these have been punched and typed, routine causes reading and verification of one word at a time, each reading and verification being followed by typing and punching of another word.

During the typing and punching of the first 17 words, the visual readout displays the location being recorded on the right side of the readout; when reading and verification commences, the visual readout displays the location just verified in the first half command portion of the readout. If the routine stops after reading a word and the Compute neon is extinguished, either (1) the reader has failed in verifying the information punched, or (2) the reader has malfunctioned by permitting too many locations to pass through the reader, or (3) the reader did not read the proper character. If the routine halts and the Compute neon remains lighted, the typewriter or tape punch malfunctioned.

Input-Output and Memory Test Routine Printout

```
L00010  +5765160-0000000

L65000  +7200330+7200100
    1.  +7400160+1600000
    2.  +7400050+0065040
    3.  +7400000+0165030
    4.  +5365030+0065010
    5.  +0165140+4265011
    6.  +3665010+3365150
    7.  +5065001+7365100

L65100  +0300000+5065111
    1.  +7765111+0065100
    2.  +0165140+4265100
    3.  +3665100+5765001
    4.  +0000010-0000010
    5.  +0000000-0077600
    6.  +0065200+4265011
    7.  +4265100+5765000
L65200  +0000000-0000000
```

Type Word: Command and Decimal Routine Printout

```
L77570  +6477600+6677700
L77600  +7200330+1260000
    1.  +7200100+1260001
    2.  +7200370+7200100
    3.  +0077600+0177700
    4.  +4277601+0077610
    5.  +0177700+4277611
    6.  +0377710+5077720
    7.  +5777600-0000000
L77700  +0000000-0000010
    1.  +7200100+1260201
    2.  +7777600-0000000
```

MEMORY TEST ROUTINE

Purpose

To check accuracy of writing and reading processes by writing various patterns in each sector and verifying the readout from each.

Operating Instructions

1. Enter and verify Memory Test routine.
2. Place Sense Switch B up (causes typewriter to print Memory OK. if routine runs successfully).
3. Depress Start 1 button (unless there is a malfunction, routine will halt automatically).

Comments

Malfunction causes computer to stop after displaying number of faulty location on console visual readout.

Memory Test Routine Printout

```
L00000
C+3600620+7700101
+0000600+6000620
+0000620+4200030
+6077170+0100610
+6000620+0300630
+5100020+0000620
+0300610+4200070
+0300170+5000101
-0000000+0000620
+0300610+6000620
+3600070+5200061
+4100500+6000620
+6400200+6600300
+5777600-0000000
+2525250-0000000
+0000170-0000000
+6701000+0077600
+6701000+0077600
+6701000+0077600
+0000620+0100610
+6000620+0300640
+5000400+5700140
+6777600+0077600
+0000100-0000000
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
-5263431+0370170
+6400400+6600500
+3677630+5777620
+3077750+3501000
+0001000+0301010
+5077650+7777650
+5277660+7777660
+0077620+0177740
+4277621+0077630
+0177740+6077630
+0377770+3677630
+5177620+5400651
+7200370+5700660
+0000020-0000020
+2514340-7407601
+2514340-7407601
+0077560+0377570
+0000170-0000000
+0001000-0000000
+0006000-0000000
+0100170-0000000
+0006000-0000000
+3677630+7700130
+7200340+7200010
+7200340+7200300
+7200120+7200250
+7200040+7200300
+7200170+7200330
+7200340+7200100
+5700130-0000000
```

NIXIE TUBE TEST ROUTINE

Purpose

To check accuracy of readout of the Nixie tubes under computer control with the Display command.

Operating Instructions

1. Set location counter to 7757.0.
2. Enter and verify Nixie Tube Test routine.
3. Place Transfer Stop switch up.
4. Set location counter to 7757.0.
5. Depress Start button once (all Nixie tubes should display numeral 0).
6. Depress Start button twice (all Nixie tubes should display numeral 1).
7. Repeat Step 6 to display, in order, each numeral through 9.
8. Depress Start button six times (all decimal point indicators should light).
9. Place Transfer Stop switch down.
10. Depress Stop button once after routine has run approximately one minute.

Comments

Permitting routine to run continously checks whether continuous operation will cause an indicator to malfunction after warm-up.

Nixie Tube Test Routine Printout

```
L 7757.0    +6477600+0000000
  7760.0    +4100500+6077650
  7761.0    +6077660+3677651
  7762.0    +0077650+0177640
  7763.0    +5377600+5777601
  7764.0    +0421040+0421040
```

READ TAPE TEST ROUTINE

Purpose

To determine correct reading of tape under computer control with the RDZ command.

Operating Instructions

1. Prepare a tape as follows:

   a. C+7654321+7654321 Carriage Return
      -6543211+7351240 Carriage Return
   b. 12 sprocket holes (for sufficient memory access time)
   c. N +8192.
   d. 12 sprocket holes (for suffiecient memory access time)
   e. L 00011 Carriage Return
   f. S
   g. 12 sprocket holes (for sufficient tape leader to allow proper starting and stopping of reader)
   h. FABCDEFGH Carriage Return
   i. IJKLMNOPQ Carriage Return
   j. 12 sprocket holes (for sufficient memory access time)
   k. L 00111 Carriage Return
   l. S

2. Enter the program given in the routine printout via the tape reader, typewriter, or console keyboard.
3. Splice tape prepared in (1) above to form a continuous loop sufficiently long to easily fit around reader head; place loop to read start of tape first.
4. Place Operation switch in Continuous position.
5. Depress Start 1 button (first length of tape should feed, then stop and immediately resume).
6. If tape runs through successfully several times, depress Halt button.

Comments

Computer should not halt nor display during test; if either occurs, the information read improperly can be determined by comparing the contents of memory locations 0050, 0051, 0052, 0060, and 0061 with the information on the tape.

Read Tape Test Routine Printout

```
L00010  +7300500+0000500
     2. +0300170+5000040
     3. +3600500+7700040
     4. +0000510+0300200
     5. +5000061+3600510
     6. +7700061+0000520
     7. +0300210+5000110
L00100  +3600520+7700110
     1. +7300600+0000600
     2. +0300220+5000140
     3. +3600600+7700140
     4. +0000610+0300230
     5. +5000161+3600610
     6. +7700010+5700010
     7. +7654321+7654321
L00200  -6543211+7351240
     1. +0000000-0100000
     2. -1713441-0556520
     3. -3133710+6146130
```

COMPUTER-CONTROLLED TYPEWRITER INPUT TEST ROUTINE

Purpose

To check proper operation of typewriter on computer-controlled input with the RDY command.

Operating Instructions

1. Enter the program given in the routine printout via the tape reader or control console.
2. Place Operation switch in Continuous position.
3. Depress Start 1 button (compute indicator should flash on and off).

4. Depress the following keys on the typewriter:

a. C+7654321+7654321 Carriage Return
b. -6543211+7351240 Carriage Return
c. N+8192.
d. L00011 Carriage Return
e. S (Compute indicator should flash on and off; if visual readout displays anything, typewriter has entered information improperly and should be checked.)

5. After Compute indicator extinguished and if no display was obtained, depress the following keys on the typewriter:

a. FABCDEFGH Carriage Return
b. IJKLMNOPQ Carriage Return
c. L00111 Carriage Return
d. S (Compute indicator should flash on and off; if visual readout displays anything, typewriter has entered improperly and should be checked.)

Computer-Controlled Typewriter Input Test Routine Printout

```
00010    +7100500+0000500
00020    +0300170+5000040
00030    +3600500+7700041
00040    +0000510+0300200
00050    +5000061+3600510
00060    +7700061+0000520
00070    +0300210+5000110
00100    +3600520+7700110
00110    +7100600+0000600
00120    +0300220+5000140
00130    +3600600+7700140
00140    +0000610+0300230
00150    +5000161+3600610
00160    +7700010+5700010
00170    +7654321+7654321
00200    -6543211+7351240
00210    +0000000-0100000
00220    -1713441-0556520
00230    -3133710+6146130
```

COMPUTER-CONTROLLED TYPEWRITER ALPHANUMERIC INPUT TEST ROUTINE

Purpose

To check accuracy of input of alphanumeric information from typewriter under computer control with the RDY command.

Operating Instructions

1. Enter the routine given in the routine printout with the tape reader or from the control console.
2. Depress Start 1 button (routine should halt immediately).

3. Depress following typewriter keys (DO NOT DEPRESS TYPEWRITER FILL SWITCH):

Carriage Return Carriage Return Carriage Return Letter Shift THIS (pause)

Space IS space RECO (pause: Place Sense Switch B up)

MP Period Carriage Return Blank Blank Blank Blank (After depressing the last Blank the computer should immediately begin typing the message entered: THIS IS RECOMP. and then Carriage Return).

4. Depress Halt button on control console.

Computer-Controlled Typewriter Alphanumeric Input Test Routine

```
L03060  +6403100+4100500
     7. +5777600-0000000
L03100  +7177600+6020000
     1. +0077600+0177630
     2. +6077600+5777600
     3. +0000000-0000010

L50000  +0020000-0000000
     1. +0050000+4250100
     2. +6450100+6650200
     3. +5777600-0000000
     4. +5777600-0000000
     5. +5777600-0000000
     6. +5777600-0000000
     7. +5777600-0000000
L50100  +0020000+6050320
     1. +4000160+5177621
     2. +0150310+4277700
     3. +4277731+0050320
     4. +4100010+4277730
     5. +4277761+4000050
     6. +4277720+4277751
     7. +4000050+4277741

L50200  +7200000+4277710
     1. +7200000+0077600
     2. +7200000+0150300
     3. +7200000+7200000
     4. +4277600+7200000
     5. +4100000+7200000
     6. +3677600+7200000
     7. +5477600+5750330
L50300  +0000010-0000000
     1. +0000200-0000000
     2. -1737040+7400000
     3. +0077600+4250340
     4. +0020150+5050360
     5. +5777600-0000000
     6. +7750010-0000000
```

PUNCH WORD TEST ROUTINE

PURPOSE

To test operation of punch under computer control with PNW command in command and binary-command and binary-coded-decimal formats.

OPERATING INSTRUCTIONS

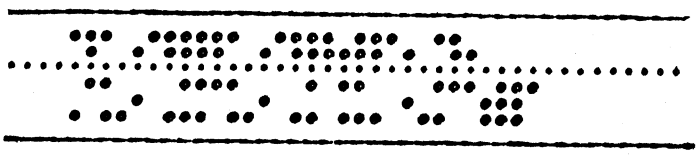
1. Enter the following routine with tape reader, typewriter, or console keyboard:

Location	Commands
0001.0	+7200330+1400030
0002.0	+1400041+5700010
0003.0	+7654321+7654321
0004.0	-0022150-2547000
0005.0	+4234671-7400000

2. Prepare punch for operation.
3. Place Operation switch in continuous position.
4. Depress Start 1 button (see Comments).
5. After routine has run desired number of times, depress stop button.

Comments

The following tape should be punched each time the routine runs:

Start of Tape 

PUNCH AND TYPE WORD TEST ROUTINE

Purpose

To test operation of typewriter and punch under computer control with PTW command in command and binary-coded-decimal formats.

OPERATING INSTRUCTIONS

1. Enter the following routine with the tape reader, typewriter, or console keyboard:

Location	Commands
0001.0	+7200370+7200100
0002.0	+7200330+1600040
0003.0	+1600051+5700010
0004.0	+7654321+7654321
0005.0	-0022150-2547000
0006.0	+4234021-4760000

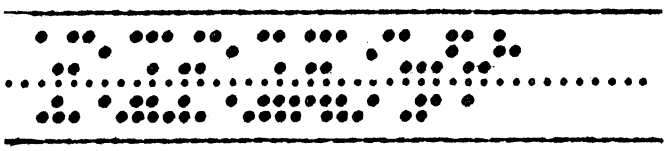
2. Prepare typewriter and punch for operation.
3. Place Operation switch in Continuous position.
4. Depress Start 1 button (see Comments):
5. After routine has run desired number of times depress Stop button.

Comments

The following information should be typed each time the routine runs:

+7654321+7654321-0123456789.0

The following tape should be punched each time the routine runs:

Start of Tape 

TRAPPING MODE TEST ROUTINE

Purpose

To test correct operation of the Trapping Mode from various memory locations.

Operating Instructions

1. Enter the following routine with the tape reader, typewriter, or console keyboard:

Location	Commands
0000.0	+7700000-0000000 The above command in location 0000.0 must always accompany either of the commands below entered from any other location
0001.0	+0000000-0000000
0002.0	-0000000+0000000
Any other	The commands at 0001.0 or 0002.0

2. Place Operation switch in Continuous position.
3. Depress Start 1 button (computer should halt and location counter should read 0000.0).
4. Depress Start 2 button (computer should halt and location counter should read 0000.0).
5. If any other location is selected, set location counter to location in which it is desired to place the pair of commands at 0001.0 or 0002,0 and enter the pair chosen.
6. Set location counter to location chosen in Step 5.
7. Depress Start button to start routine at location chosen in Step 5. (computer should halt and location counter should read 0000.0).

LOOP LOAD ROUTINE

Purpose

To fill L and V loops and A, B, C, and R registers with pattern in memory locations 0003.0 through 0013.0 of this routine for checking loops and registers for amplitude, crosstalk and noise, and pulse width.

Operating Instructions

1. Enter and verify Loop Load routine with tape reader, typewriter, or console keyboard.
2. Depress Start 1 button (computer halts automatically).

Loop Load Routine Printout

```
L00010  +6400030+6600030
    2.  +3000030+5700030
    3.  +7725250-0005250
    4.  +7725250-0005250
    5.  +7725250-0005250
    6.  +7725250-0005250
    7.  +7725250-0005250
L00100  +7725250-0005250
    1.  +7725250-0005250
    2.  +7725250-0005250
    3.  +7725250-0005250
```

FAST MEMORY FILL AND ONE CHANNEL ZERO ROUTINE

Purpose

To fill main memory with pattern in memory locations 0040.0 through 0047.0 of this routine and load any channel selected (except 00) with all zeros for checking memory for amplitude, crosstalk and noise, and pulse width.

Operating Instructions

See instructions for checking main memory in Section IV.

```
L00000  +5600020+7700010
     1. +5700260-0000 00
     2. +7177610+4100120
     3. +4200101+4100500
     4. +7177610+3300220
     5. +4100070+0300100
     6. +4200101+0300210
     7. +4200161+3600100
L00100  +0000140+6000000
     1. +0000100+0100150
     2. +4200101+0000100
     3. +5700170+0000000
     4. -0000000-0000000
     5. +0000000-0000010
     6. +0000140+6000000
     7. +0300160+5000201
L00200  +5700100+7700010
     1. +0000000-0001000
     2. +0000000-0000031
     3. +5777030+0077700
     4. +1177720+2377720
     5. +0377700+5077000
     6. +6400300+6600400
     7. +5777600-0000000
L00300  +0077670+6077610
     1. +6701000+0177660
     2. +6077610+0377650
     3. +5000000+0077610
     4. +5777610-0000000
     5. +6777600+0177660
     6. +0000100-0000000
     7. +6701000+0177660
L00400  +2514340-7407601
     1. +2514340-7407601
     2. +2514340-7407601
     3. +2514340-7407601
     4. +2514340-7407601
     5. +2514340-7407601
     6. +2514340-7407601
     7. +2514340-7407601
L00500  +1277710-3001670
     1. -3001710+1277720
     2. -3001730+1277730
     3. -3001670-3001710
     4. +1277740-3001730
     5. +1277750-3001750
     6. -3001630+1276540
     7. -3001670+5700740
L00600  +5241571+0000000
     1. +5242571+0000000
     2. +5243571+0000000
     3. +5244571+0000000
     4. +5245571+0000000
     5. +5246571+0000000
     6. +5247571+0000000
     7. +5250571+0000000
L00700  +5251571+0000000
     1. +5020571+0000000
     2. +5021571+0000000
     3. +5022571+0000000
     4. -3001710+1276550
     5. -3001730+1276560
     6. +7200370+5700330
     7. +0000000-0000001
L01000  +6157741+7556460
```

TO ZERO A CHANNEL:

DEPRESS START 2. STRIKE TWO KEYBOARD CHARACTERS OF THE CHANNEL DESIRED TO BE ZEROED. (FROM CHANNEL 0100 AND UP.)

TO LOAD MEMORY WITH PATTERN:

DEPRESS START 1 AND MEMORY WILL LOAD WITH PATTERN FOUND IN LOCATIONS 00400 THRU 00470.

APPENDIX I. DEFINITION OF LOGIC TERMS

FUNCTIONAL COMPONENT LISTING

Code	Name
E 0	Add-Subtract Indicator Flip-Flop
E 1-3	Input Sequence Counter
E 20	"Number Fill" State Flip-Flop
E 30	"Number Fill" State Flip-Flop
E 40	"Number Fill" State Flip-Flop
E p1-p5 & E'p1-p5	"Punch" Echo Signals
E t1-t5 & E't1-t5	"Type" Echo Signals
F 0	Fill Order Counter Flip-Flop
F 1	Input-Output Register
F 2	Input-Output Register
F 3	Input-Output Register
F 4	Input-Output Register
F 5	Input-Output Register
F c1, c2	Format Control Flip-Flop
G 0	Half-Word Indication Flip-Flop
G 1, 2	Location Counter Logic
G 3-10	Location Counter Logic
G 11, 12	Location Counter Logic
H 0	Transfer of Control-Verify Input Flip-Flop
H 1-4	Readout Control
H c	"Stop Compute" Button
I 1	State Flip-Flop
I 2, 3	State Flip-Flop
I 4	State Flip-Flop
I 5	Input Buffer Fill Flip-Flop
J 0	Overflow and Error Flip-Flop
J 1-6	Sector Agreement Signals for Readout
J 7-12	Channel Set Signals for Readout
J 13	Output Error Flip-Flop
K 0	Loop Read Flip-Flop
K 1-5	Input Channel Switches 1 through 5
K a	Computation Carry Flip-Flop
K f	Keyboard Fill Button
K g	Order Counter Carry Flip-Flop
K p5	Photoreader Input Channels 1 thru 5
K q	Computation Cycle Counter Carry Flip-Flop
K s	Sector Counter Carry Flip-Flop

Code	Name
L 1-2	L-Memory Loop
L 2a	L-Loop 4-Word Read Flip-Flop
L 41	L-Memory Loop
L b	Fill "Location" Button
L p1	Loop-Memory Selection Flip-Flop
L p2	L-V Loop Selection Flip-Flop
L r4	L-Loop Read Amplifier Output from 4-word Read Head
L r8	L-Loop Read Amplifier Output from 8-word Read Head
L w	L-Loop Write Amplifier Input
M 0-2	Operation-Type Flip-Flops
M 3-5	Operation-Type Flip-Flops
M 6-10	Operation-Type Flip-Flops
M d	Display Selection Switch Setting for Memory
M r	Memory Read Flip-Flop
M ra	Memory Read Power Amplifier
M rb	Memory Read Power Amplifier
M rc	Memory Read Power Amplifier
M rd	Memory Read Power Amplifier
M re	Memory Read Power Amplifier
M rf	Memory Read Power Amplifier
M rg	Memory Read Power Amplifier
M rh	Memory Read Power Amplifier
M r0-r63	Memory Read Amplifier Outputs from Channels 0 to 63 respectively
M s	Memory Sync Flip-Flop
M w1	Main Memory Write Flip-Flop
M w2	Main Memory Write Flip-Flop
N 0	Fill Decimal Number Flip-Flop
N 1	Sector Counter Timing Flip-Flop
N 2	Operation Code Timing Flip-Flop
N 4	Location Counter Timing Flip-Flop
N 5	Channel Register Timing Flip-Flop
N 6	Output Timing Flip-Flop
N 7	PS Order Flip-Flop
N 8	Readout Control
N 8a	Readout Timing Flip-Flop
N 9	Sector Counter Timing Flip-Flop
N 11	State Change Flip-Flop
N 12	State Change Flip-Flop
N b	Fill "Numbers" Button

Code	Name
O 0	Fill Orders Flip-Flop
O 1	Fill Orders Flip-Flop
O f	Octal Setting of Format Switch
P 0	Legitimate Code Flip-Flop
P 1-6	Digit Counter
P d3	Delay Flip-Flop
P m	Delay Flip-Flop
P ma	Delay Flip-Flop
Q 0	Computation Cycle Counter
Q 1-6	Computation Cycle Counter
R 1-2	R-Register
R 40	R-Register
R 41	R-Register
R 42	R-Register
R 43	R-Register
R b	Readout Activation Button
R c1-c4	Readout Timing Flip-Flops
R d	Display Selection Switch Setting for R-Register
R gm	Mechanical Tape Reader On Button
R gp	Photoelectric Tape Reader On Button
R sm	Mechanical Tape Reader Off Button
R sp	Photoelectric Reader Off Button
R r	R-Register Read Amplifier Output
R w	R-Register Write Amplifier Inputs
S	Sum Flip-Flop
S 0	Memory Read Sector Selection Flip-Flop
S 1-6	Sector Counter
S 7	Loop Read Sector Selection Flip-Flop
S 8	Loop Write Sector Selection Flip-Flop
S 11	Memory Write Sector Selection Flip-Flop
S c	"Start Compute" Button
S p	Preset Stop Switch On
S sB	Sense Switch for TSB Command
S sC	Sense Switch for TSC Command
S sD	Sense Switch for TSD Command
S x	Readout Timing Flip-Flop
S u1-3	Set-up Buttons 1-3
S0	"Single Order" Positions of Operation Switch
S0'	"Continuous" Position of Operation Switch

Code	Name
T 0	Tape Reader On Flip-Flop
T 1	Sync Bit Time Flip-Flop
T 2	Least significant bit time flip-flop
T 40	Most significant bit time flip-flop
T 41, 41a	Sign Bit Time Flip-Flops
T d	Initial Time Delay and Error Reset Button
T g	Typewriter On Button
T s	Typewriter Off Button
T p	"Punch" Echo Timing Pulse
T t	"Type" Echo Timing Pulse
T x	Readout Timing Flip-Flop
TPP	Tape Punch Pulse Flip-Flop
TPP k	Tape Punch Pulse for Keyboard
TPP p	Tape Punch Pulse for Punch
TPP rm	Tape Punch Pulse for Mechanical Reader
TPP rp	Tape Punch Pulse for Photoelectric Reader
TPP t	Tape Punch Pulse for Typewriter
TPP 5	Input Sampling Signal = (TPP) P_5
U 1-2	Computation Word Time Flip-Flops
U 41	Last Divide Word Time Flip-Flop
V 1-2	V-Memory Loop
V 2a	V-loop 4-word Read Flip-Flop
V 41	V-Memory Loop
V gm	Mechanical Tape Reader "Verify" Button
V gp	Photoelectric Reader "Verify" Button
V r4	V-Loop Read Amplifier Output from 4-word Read Head
V r8	V-Loop Read Amplifier Output from 8-word Read Head
V w	V-Loop Write Amplifier Input
W 0-63	Memory Write Amplifiers
X 0	Origin Flip-Flop
X 1, 2	X-Register
X 9	Punch Flip-Flop
X 10	Type Flip-Flop
X 20	Blinker Flip-Flop
X 40	X-Register
X 41	X-Register
X or	Origin Pulse Amplifier

Code	Name
X d	X-Register Display Selection Switch Setting
X r	X-Register Read Amplifier Output
X w	X-Register Write Amplifier Input
Y 0-12	"Preset Stop" Bit Agreement Signals
Z 1, 2	Order Register
Z 41	Order Register
Z ch	Input Zeroing-Keyboard Stop Flip-Flop
Z d	Display Selection Switch Setting for Instruction Register
Z r	Z-Register Read Amplifier Output
Z w	Z-Register Write Amplifier Input

DEFINITION OF FUNCTIONAL COMPONENTS

A_0 Inhibit Fill Flip-Flop:

(a) Used in decimal fill to prevent decimal point from being interpreted as an ENTER code following a plus or minus sign:*

$$\boxed{1^{i}1 = F_4\, F_3\, F_2'\, F_1'\, N_0\, A_0'\, P_{d3}\, T_{41a}\, C}$$

(b) Prevents an ENTER code after a decimal point from turning on I_5 a second time. I_5 is first turned on by a decimal point code, and places the number in the X-register for storage in memory. It is the habit of programers to hit the ENTER button at the end of a fill process. The ENTER button would ordinarily turn on I_5 a second time if not prevented by A_0. Turning on I_5 a second time would put random information into the X-register instead of the number to be filled.

*Decimal point normally ENTERS in decimal fill but will only enter if N_0 is true (fill decimal number flip-flop). A plus or minus sign turns on A_0. A_0 prevents I_1 from turning on. Therefore A_0 prevents the decimal point code from entering information into the memory when the decimal point code follows a plus or minus sign.

$A_{1, 2, 3, 39, 40, 41}$

These six flip-flops form the two ends of the accumulator register; the remaining 35 bits of information which constitute a word are stored on the memory disk. With appropriate timing this register shifts from the high-numbered to low-numbered end. Information is entered into the accumulator register via A_{41} and the output is taken from A_1. This register is used to hold intermediate and final results of all arithmetic operations.

A_{42} Logical Function of Flip-Flops and Registers

A flip-flop used as an auxiliary to the A-register and utilized for certain operations as follows:

During left-shift operation it is added to the A-register:

$$1^{a}_{42} = A_1 M_4 D_2' T_{41a}' C$$

During floating multiply while adding exponents, A_2 recirculates to A_{42}:

$$1^{a}_{42} = A_2 M_{10} C$$

$$0^{a}_{42} = A_2' M_{10} C$$

During square root operation it copies the sum flip-flop:

$$1^{a}_{42} = S M_2 C$$

Stores sign bit in decimal fill:

$$1^{a}_{42} = F_3' F_1 N_0 P_0 E_2' E_1 C$$

Store address; sets N_7 to select first or second address to be stored; A_{42}, RIGHT, second address; A_{42}' LEFT, first address. copies G_o information from Z-register:

$$1^{a}_{42} = Z_1 I_2 I_4' N_9 C$$

Determines decimal command readout.

A_n Delay Flip-Flop (Alphanumeric Input)

Used in alphanumeric fill for read-Y (typewriter) and read-Z (photo-reader) instructions:

Allows alphanumeric input 1 to 8 five-bit characters from keyboard:

$$1^{a}_{n} = M_7 D_6 D_3' D_1 L_{p1} C$$

A_n is also one-set by an F code from typewriter or tape reader when alphanumeric fill is desired from this source:

$$1^{a}_{n} = F_5' F_4 F_3 F_2' F_1 C_t C$$

A_t Delay Flip-Flop

This flip-flop is one-set for alphanumeric fill. After 8 characters have been struck, it remains one-set while the format is changed to command fill in order to enter the information (the alphanumeric flip-flop, A_n, is zero-set to receive the enter code). This flip-flop then functions to reset the input format to alphanumeric.

$B_{1, 2, 40, 41}$

These four flip-flops form the two ends of the B-register while the remaining 37 bits of information which constitute a word are stored on the memory disk. With appropriate timing this register shifts one bit at a time from the high-numbered end to the low-numbered end. Information is entered into this register via B_{41} and the output is taken from B_1. This register is used to hold the operand and/or intermediate results of arithmetic operations.

B_{42}

This flip-flop is used in division to hold the sign of the quotient. It is used as an extension of the B-register during floating-multiply, floating-division, and square root.

B_c Photoreader Clutch-Brake Flip-Flop

This flip-flop is used to energize and de-energize the solenoid on the photoreader; hence the use of this flip-flop is to engage the clutch or apply the brake to the tape in the photoreader.

C_1 - C_6 Channel Register Flip-Flops

This register consists of a series of flip-flops. Its purpose is to indicate which one of the 64 channels in memory to read from or write into.

C_j Clock Jitter Flip-Flop

This flip-flop is used to delay the clock pulse by as much as 25 percent of the clock pulse width during marginal testing. The effect of this flip-flop is to provide an uneven clock pulse rate for the marginal testing.

C_r Input Counter Reset Flip-Flop

This flip-flop is used to set the H-counter to a count of 0001 and the E-counter to a count of 100 during input and output operation.

C_t Code Conversion Time Flip-Flop

Because the typewriter and photoreader accept standard teletype code, it is necessary to convert the teletype code to the binary representation which is used internally in this machine. This flip-flop provides the one-bit timing for the necessary conversion.

D_0 Memory Read Gating Flip-Flop

This flip-flop is used to gate a command pair from memory into the Z-register. D_0 is one-set if S_0' is true at T_{41}. D_0 remains on for one complete word time (bit times 1 through 41) after which T_{41} resets it to zero.

D_1 - D_6 Operational Code Register

This register consists of a series of flip-flops D_1 - D_6. Its purpose is to hold the command that is to be executed. By means of the binary configuration in this register, gates are set up in the computer to allow the operation desired.

D_{10} Loop Gating Flip-Flop

D_{10} allows the command pair read from the loop into K_0 to be gated into the Z-register as required. D_0 is one-set for a loop address if S_7' indicates that the desired word will appear under one of the read heads during the next word time. D_{10} remains on for one complete word time (bit times 1-41) at which time T_{41} resets it to zero.

D_{11} Memory Write Gating Flip-Flop

This flip-flop is used to gate information into the memory write flip-flops. From there it enters the memory write amplifiers and is written into memory. D_{11} comes on only when sector agreement has occurred and stays on for one word time after which T_{40} resets it to zero.

E_0 Add-Subtract Indicator Flip-Flop

This flip-flop is used to establish whether absolute addition or absolute subtraction is required when addition or subtraction is to be performed; that is, whether to add or subtract the two quantities algebraically.

E_{1-3} Input Sequence Counter

This counter counts the number of times the P_0 flip-flop is one-set, hence this counter counts the number of legitimate characters being entered.

E_{20} Number Fill State Flip-Flop

During number fill it becomes necessary to multiply the contents of the R-register by ten. This flip-flop provides the necessary timing to accomplish this multiplication.

E_{30} Number Fill State Flip-Flop

This flip-flop provides the timing logic for adding the contents of the F-register to the contents of the R-register during number fill. This flip-flop also provides the timing for clearing the F-register.

E_{40} Number Fill State Flip-Flop

This flip-flop provides the timing logic for multiplying the contents of the B-register by ten during number fill. This flip-flop also includes the timing for copying the contents of the R-register into the A-register. Another function of this flip-flop is to reset the E-counter to 010.

F_0 Fill Order Counter Flip-Flop

This flip-flop is used to establish the input format to receive a location and set the location counter.

F_1 - F_5 Input-Output Register

These five flip-flops constitute the input-output register.

(a) During input mode three sources provide information: typewriter , photoreader , and keyboard . The first two devices generate a teletype or Baudot code (see figure) which is accepted

by the input register flip-flops F_1, F_2, F_3, F_4, F_5. They are set in accordance with the Baudot code sent to them. At a certain bit time later the state of these flip-flops is changed by the following equations to effect a translation to a corresponding binary code used internally in the computer.

$$1f_1 = F_5 F_3' F_2' C_t C$$

$$0f_1 = F_3' F_2 C_t + F_4' F_3 F_2' C C_t$$

$$1f_2 = F_5' F_2' C C_t + F_4' F_2' F_1 C_t C$$

$$0f_2 = F_2 F_1' C C_t + F_5 F_3 F_2 C_t C + F_4 F_1 C_t C$$

$$1f_3 = F_5' F_4 C C_t + F_4' F_5 F_2' F_1' C_t C$$

$$0f_3 = F_5 F_2 C C_t + F_5' F_4' F_1' C_t C$$

$$1f_4 = F_5 F_3' F_2' F_1 C C_t + F_5' F_4' F_1' C C_t + F_5' F_4' F_3' F_2 C_t C$$

$$0f_4 = F_4 F_2 C_t C$$

$$1f_5 = F_4' F_3' F_1 C C_t + F_4' F_3 F_2 C_t C + F_5' F_4 F_3' F_1' C_t C$$

$$0f_5 = F_4 F_3' F_1 C C_t + F_4 F_3 F_2 C C_t + F_4' F_3 F_2' F_1' C_t C + F_5 F_3' F_2 F_1' C_t C$$

The translation process from Baudot to binary is accomplished by specifically turning on or off F flip-flops as required for the translation with a configuration of the F flip-flops initially set by the teletype code input in combination with C_t (code conversion time flip-flop) (figure). In terms of the computer time base the F flip-flops are first set by the incoming information in Baudot code.

The F flip-flops are again set in a translation process to a binary configuration corresponding to the particular Baudot character (for internal logic of the computer which functions by use of the binary code).

F_{c1}, F_{c2} format control flip-flop

The four possible combinations of these two flip-flops is used to determine the format of the information to be output.

G_0 Half-Word Indicator Flip-Flop

This flip-flop is used to specify the first or second command depending on whether it is zero-set or one-set, respectively. When referring to a location which contains data (in contrast to a command), a zero is used to indicate the data is in command format and a one is used to indicate the data is in BCD format and occupies the location specified plus the following sector (two successive words).

G_1 - G_{12} Location Counter

The location counter must contain the desired memory location to read from or write into. G_7 - G_{12} determines the memory channel; G_1 - G_6 determines the sector.

H_0 Transfer of Control-Verify Input Flip-Flop

During the verify mode of operation this flip-flop is initially set to zero and then a comparison is made between the information previously recorded in memory and the information currently being read by the photoreader. Disagreement results in setting this flip-flop which activates the verify error light. This flip-flop is also used to reset the location counter (K_g and G_0) when a transfer of control is indicated.

H_{1-4} Readout Control Flip-Flop

This four-flip-flop counter counts the number of characters shifted through the output register by counting the times the N_8 flip-flop is one-set (which is once per character).

I_1 Computer State Flip-Flop

This flip-flop is utilized for timing. During the time I_1 is true, the desired command pair is located and transferred from the memory to the Z-register. Since a full word was taken from memory, this function is not necessary for the second command in a pair.

I_2 Computer State Flip-Flop

This flip-flop is utilized for timing. During the time I_2 is true, the word information referred to by the address of the command is located and transferred from memory to the B-register.

I_3 Computer State Flip-Flop

This flip-flop is utilized for timing. During the time I_3 is true, the operation designated by the command is executed. Also at this time the location counter is augmented to its succeeding value unless a transfer-of-control command is encountered.

I_4 Computer State Flip-Flop

This flip-flop is utilized for timing. I_4 becomes true if the computer encounters a stop command or error. Therefore, I_4 is known as the noncompute flip-flop. During I_4 input-output operations such as computer fill, verify, or readout may be performed.

I_5 Computer State Flip-Flop

This flip-flop is utilized for timing. When a word has been filled into the accumulator and the ENTER button is depressed, I_5 becomes true and allows A_1 through A_{41} to be copied into X_1 through X_{41}, and then the word enters the memory.

During floating square root, I_5 becomes true if the exponent does not have to be denormalized. If I_5 does not become true, then the exponent needs denormalizing.

J_0 Overflow Error Flip-Flop

This flip-flop is one-set when a number exceeding the capacity of the machine is called for; that is, when an attempt is made to enter too large a number or when the results of an arithmetic operation exceed absolute one. This flip-flop activates the overflow error light.

J_{13} Output Error Flip-Flop

During output operations the computer directs a signal to the output device which in turn transmits to the computer a signal signifying the character being output. If the two signals are not identical, this flip-flop is set to activate the output error light.

K_0 Loop Read Flip-Flop

This flip-flop copies bit by bit the information being gated from the loop memory and acts as a buffer between the loop memory and the register to which the information is being transferred.

K_a Computation Carry Flip-Flop

This flip-flop is used in the logic to determine whether a carry or borrow arithmetic condition has been made in addition and subtraction.

K_g Carry Flip-Flop for Location Counter

This flip-flop comes on at bit time 2 and stays on until a zero is shifted into G_1 during N_4 timing. This is done to provide the one input to the counter, counting it up to the next location.

K_q Carry Flip-Flop for Computation Cycle Counter

This flip-flop injects the "one" which makes the Q-counter a one-input adder. It serves also to inject the zero to the Q-counter the first word time following the execution of a command or arithmetic operation.

K_s Carry Flip-Flop for Sector Counter

This flip-flop comes on at bit time 2 and stays on until a zero is shifted into S_1 during N_1 timing. This is done to provide the one input to the sector counter.

$L_{1, 2, 41}$ L-Loop Memory Flip-Flops

These flip-flops are used to delay the information by three bit times in the recirculation path to compensate for the difference in spacing between the write head and the two read heads. L_2 acts as a buffer between the memory and the K_0 flip-flop.

L_{2a} L-Loop 4-Word Read Flip-Flop

This flip-flop is used as a buffer between the four-word read head and the K_0 flip-flop.

L_{p1} Loop-Memory Selection Flip-Flop

This flip-flop selects whether the information will be gated out of the main memory or from one of the high-speed loops.

L_{p2} L-V-Loop Selection Flip-Flop

If the high-speed loop memory is indicated, this flip-flop is used to determine whether the L-loop or V-loop is indicated.

M_0 Operation-Type Flip-Flop

This flip-flop is used to specify the 00, 01, 02, and 03 code commands. It is used to decrease the number of inputs to "and" gates in the logic.

M_1 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 10 through 17 inclusive.

M_2 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 20 through 27 inclusive.

M_3 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 30 through 37 inclusive.

M_4 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 40 through 47 inclusive.

M_5 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 50 through 57 inclusive.

M_6 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 60 through 67.

M_7 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 70 through 77.

M_8 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 04, 06, 44, and 46.

M_9 Operation-Type Flip-Flop

This flip-flop is used to specify the command codes 05 and 07.

M_{10} Operation-Type Flip-Flop

This flip-flop identifies the operation of adding or subtracting the exponents and storing the resultant exponent in the X-register during floating multiplication and division. This operation takes place in two word times.

M_r Memory Read Flip-Flop

This flip-flop is used to copy the information being gated from main memory. It acts as a buffer between the main memory and the register to which the information is being directed.

M_{w1} Main Memory Write Flip-Flop

This flip-flop is used to write a binary "0" in main memory when it is zero-set.

M_{w2} Main Memory Write Flip-Flop

This flip-flop is used to write a binary "1" in main memory when it is zero-set.

N_0 Fill Decimal Number Flip-Flop

This flip-flop establishes the input format so that the computer will accept a decimal integer and transform it to a four-place binary equivalent.

N_1 Sector Counter Timing Flip-Flop

This flip-flop comes on at bit times 3 or 23 and stays on through bit times 8 or 28 respectively, depending on the half-word indicated. During this time it allows the sector counter to shift and augment one count. It is also a timing gate interval for N5 and sector selector flip-flops.

N_2 Operation Code Timing Flip-Flop

This flip-flop comes on at bit times 14 or 34 and stays on through bit times 19 or 39 respectively, depending on the half-word indicated. During these times it allows the operational code register to receive new information from the order register and to shift out the old.

N_4 Location Counter Timing Flip-Flop

This flip-flop comes on at bit times 3 or 23 and stays on through bit times 14 or 34 respectively depending on the half-word indicated. During these times it allows shifting and recirculating of the location counter.

N_5 Channel Register Timing Flip-Flop

This flip-flop comes on at bit times 7 or 27 and stays on through bit times 14 or 34 respectively, depending on the half-word indicated. During this time it allows the channel register to receive new information from the location counter and to shift out the old. It is also a gate for the operation code timing flip-flop N_2.

N_6 Output Timing Flip-Flop

This flip-flop comes on at bit times 3 or 23 and stays on through bit times 7 or 27 respectively, depending on the half-word indicated. During these times it allows shifting of the input and output register (F_1 - F_5).

N_7 Partial Substitution Order Flip-Flop

This flip-flop is used for the three one-word store commands, STO (store word), STA (store address), and SAX (store contents of the accumulator and exchange contents of the A- and X-registers). The use of this flip-flop is to provide the timing necessary to gate the information from A_2 to B_1 from which it is gated into memory.

N_8 Readout Control Flip-Flop

This flip-flop provides conversion timing in the output register to convert from the internal representation of the computer to the teletype code. In turn, this flip-flop provides the input to the H-counter; that is, the H-counter counts the number of times the N_8 flip-flop is one-set.

N_{8a} Readout Timing Flip-Flop

This flip-flop is similar to N_8 except that it is used in BCD format output. (In BCD, the sign is legitimate only if it is minus. Plus signs are not punched or typed. The plus sign is used to generate a figs shift code.)

N_9 Sector Counter Timing Flip-Flop (Intermediate Timing Flip-Flop)

This flip-flop is turned on at bit times 2 or 22 depending on which half of the word is indicated. It is used as a timing gate interval for N_1, N_4, and N_6 timing flip-flops.

N_{11} State Change Flip-Flop

This flip-flop is used to terminate the I_2, command interpretation state. The logic one-sets this flip-flop when the necessary interpretation of the command to be executed has been completed. This flip-flop becomes one-set during the last bit time of a word time. The following word time, it zero-sets both itself and the I_2 flip-flop which terminates the command interpretation state.

N_{12} State Change Flip-Flop

This flip-flop is used to terminate the I_3, command execution state. During the last bit time of the last word time in the execution of an arithmetic operation or command, this flip-flop is one-set. The following bit time, as with N_{11}, it zero-sets both itself and the I_3 flip-flop simultaneously which terminates the command execution state.

O_0 Fill Orders (Command Fill) Flip-Flop

This flip-flop is used in conjunction with logic to establish the command format-type of input; that is, the first character which is entered is treated as the sign which is a binary bit, the next six characters are converted to octal, the next character is treated as the half-word indicator and fills as a binary bit, the next character is treated as the sign of the second command and enters as a binary bit, the next six characters are filled as octal information, and finally the last character is the half-word indicator of the second command filling the last binary bit.

O_1 Fill Orders Flip-Flop

When this flip-flop is one-set, the input information accepted by the computer is an octal character. When this flip-flop is zero-set, the computer accepts only binary information.

P_0 Legitimate Code Flip-Flop

This flip-flop is set for one word time if the character being input is a legitimate character for the format selected.

P_1 - P_6 Young Digit Counter

This counter is used to count the number of binary bits in each word. This counter identifies each of the 41 bits within a given word and furnishes appropriate signals for timing control.

P_{d3} Delay Flip-Flop

This flip-flop provides a delay for determining whether the character being input is legitimate. During output operation this flip-flop indicates termination of the command and clears the F (output) register.

P_m Delay Flip-Flop

This flip-flop is used to reset the Q-counter to zero during input-output operation, other than alphanumeric.

P_{ma} Delay Flip-Flop

Operation of this flip-flop is similar to that of P_m except this flip-flop is used during alphanumeric input-output.

Q_0 Computation Cycle Counter Flip-Flop

This flip-flop is used during floating addition and subtraction to determine the sign of the exponent and whether or not the exponent is zero.

Q_1 - Q_6 Computation Cycle Counter Flip-Flop

These six flip-flops constitute a one-input adder which counts the word times during the execution of commands in order that the operation being performed will be terminated at the proper time.

$R_{1, 2, 40, 41, 42, 43}$ R-Register Flip-Flops

These flip-flops form the two ends of the R-register while the remainder of the register is contained on the memory disk. R_{42} and R_{43} are not normally part of this register, but act to extend the register under special conditions.

R_{c1-c4} Readout Timing Flip-Flops

R_{c1} is used to initiate a readout. R_{c2} is used in timing the shifting of information from the R-register and the B-register. R_{c3} is used in shifting the information into the nixies. R_{c4} is used to clear the previous display in the nixies.

S Sum-Difference Flip-Flop

During addition and subtraction, this flip-flop creates the sum or difference.

S_0 Memory Read Sector Selection Flip-Flop

This flip-flop is one-set, if during sector comparison, sector agreement is not reached. At the end of each word, T_{41} zero-sets S_0. Therefore, if the next word provides us with sector agreement, S_0 remains zero-set. This provides D_0 or K_0 with the necessary terms to start gating information from memory.

S_1 - S_6 Sector Counter

This counter is used to count the number of word sectors of which there are 64 per revolution, always indicating the sector address of the next word to pass under a given read head. Such a counter is sometimes called a one-input adder.

S_7 - S_8 Loop Read Sector Selection Flip-Flop

These two flip-flops are used to determine whether the 4-word or 8-word read head is to be used for reading from the loops. The configuration of these flip-flops is dependent upon the sector comparison.

S_{11} Memory Write Sector Selection Flip-Flop

This flip-flop is one-set, if during sector comparison, sector agreement is not reached. At the end of each word T_{41} zero-sets S_{11}.

Therefore, if the next word provides sector agreement, S_{11} remains zero-set, indicating that the desired sector is the next to appear under the write head.

S_x Readout Timing Flip-Flop

This flip-flop is used to determine whether to shift a binary bit (when it is one-set) or whether to shift an octal character (when it is zero-set) during output operation.

T_0 Tape Reader "On" Flip-Flop

This flip-flop is used to indicate to the computer that the photoelectric tape reader is turned on.

$T_{1, 2, 40, 41, 41a}$ Digit Time Flip-Flop

These flip-flops are used throughout the computer logic as a timing reference. The digit counter could be used, but to save on logic terms the T flip-flops were used. They provide a term to represent the beginning (T_1) and a term to represent the end (T_{41}) of each word.

T_x Readout Timing Flip-Flop

During the time this flip-flop is one-set, information is shifted into the F (output) register. When this flip-flop is false during output, information is copied into the nixies.

TPP Tape Punch Pulse Flip-Flop

This flip-flop is used to indicate to the computer that either the punch or typewriter is in the process of punching or typing.

U_1, U_2, U_{41} Computation Cycle Time Flip-Flops

During the execution of commands certain word times appear in numerous logic gates. U_1 and U_2 represent the first and second word times, respectively, of the execution of commands because of the importance of these word times. U_{41} is used to represent the last word time during the execution of a command.

$V_{1, 2, 2a, 41}$ V-Loop Flip-Flops

Identical to the L-flip-flops of the same number

X_0 Origin Flip-Flop

This flip-flop is used to initialize the Young digit counter and the sector counter. It also provides us with a reference point on the memory disk. $_1X_0 = X_{0r}$

X_1 - X_{41} Register

This register has several uses. First, it is used as a buffer register for information coming from an input device. Second, it is used for the exponent portion of a floating-point number when the number is coming from memory.

X_9 Punch Flip-Flop

This flip-flop is for initializing. When the punch coding solenoids have copied the output register (F_1 - F_5) X_9 becomes true. This activates the punch clutch magnet and initializes the mechanics of punching.

X_{10} Type Flip-Flop

This flip-flop is for initializing. When the typewriter coding solenoids have copied the output register (F_1 - F_5) X_{10} becomes true. This activates the typewriter clutch magnet and initializes the mechanics of typing.

X_{20} Blinker Flip-Flop

This flip-flop is used to blink the error-trouble indicator lights.

X_{0r} Origin Pulse Amplifier

X_{or} is the amplifier used to amplify the origin pulse from the memory disk, and supply a signal to the origin flip-flop X_0.

Z_1 - Z_{41} Order Register

This register is comprised of 41 bit positions, 38 of which are contained on the memory disk. The other 3 are held in flip-flop's Z_1, Z_2, and Z_{41}. A read and write head is used as a recirculating register. Its purpose is to excerpt from memory the selected command word.

Z_{ch} Input Zeroing-Keyboard Stop Flip-Flop

This flip-flop is used to zero the F output register after each character is entered. This flip-flop is one-set by the E-counter in order to accomplish this purpose.

APPENDIX II. LOGIC EQUATIONS

INDEX FOR RECOMP II LOGICAL EQUATIONS

This page intentionally left blank

RECOMP II LOGIC EQUATIONS

CIRCUIT BOARD LOCATION

ORIGIN FLIP-FLOPS

X_0 $\quad {}_1X_0 = X_{0r}$

$\quad {}_0X_0 = X_{0r}'$

X_{0r} ORIGIN PULSE AMPLIFIER

YOUNG DIGIT COUNTER

P_1 $\quad {}_1P_1 = P_5' P_1' X_0'$ 463-10

$\quad + P_6' P_5 P_4' P_3' X_0'$ 563-9

$\quad {}_0P_1 = P_5' P_1$ 463-34

$\quad + X_0$ 463-34

P_2 $\quad {}_1P_2 = P_1 X_0'$ 461-18

$\quad {}_0P_2 = P_1'$ 461-19

$\quad + X_0$ 460-19

YOUNG DIGIT COUNTER (Cont)

P_3	${}_1P_3 = P_2$	264-19
	${}_0P_3 = P_2'$	263-19
P_4	${}_1P_4 = P_3 X_0'$	265-18
	${}_0P_4 = P_3'$	263-34
	$+ X_0$	263-34
P_5	${}_1P_5 = P_4 X_0'$	261-18
	${}_0P_5 = P_4'$	260-34
	$+ X_0$	260-34
P_6	${}_1P_6 = P_6' P_5 P_4' P_3' P_2'$	159-9
	${}_0P_6 = T_{41a}$	259-19

CIRCUIT BOARD
LOCATION

SECTOR COUNTER

S_1	$_1s_1 = S_2 N_1$	402-18
	$_0s_1 = S_2' N_1$	202-18
	$+ X_0$	202-19
S_2	$_1s_2 = S_3 N_1$	203-18
	$_0s_2 = S_3' N_1$	403-34
	$+ X_0$	403-34
S_3	$_1s_3 = S_4 N_1$	404-18
	$_0s_3 = S_4' N_1$	204-18
	$+ X_0$	204-19

CIRCUIT BOARD LOCATION

SECTOR COUNTER (Cont)

S_4	$_1S_4 = S_5 N_1$	405-18
	$_0S_4 = S_5' N_1$	206-34
	$+ X_0$	206-34
S_5	$_1S_5 = S_6 N_1$	208-18
	$_0S_5 = S_6' N_1$	408-18
	$+ X_0$	409-19
S_6	$_1S_6 = S_1 K_S' N_1$	222-12
	$+ S_1' K_S N_1$	222-12
	$_0S_6 = S_1' K_S' N_1$	418-12
	$+ S_1 K_S N_1$	418-12
	$+ X_0$	420-19

CIRCUIT BOARD LOCATION

SECTOR COUNTER (Cont)

K_S	${}_1k_S = T_1$	210-19
	${}_0k_S = S_1' N_1$	210-18

SECTOR COUNTER TIMING FLIP-FLOPS

N_1	${}_1n_1 = N_9$	411-19
	${}_0n_1 = P_2 P_3$	410-18
N_9	${}_1n_9 = G_0 T_1$	209-18
	$+ G_0' P_5 P_4' P_3' P_2' P_6'$	109-9
	${}_0n_9 = N_9$	208-19

CIRCUIT BOARD
LOCATION

CHANNEL REGISTER

C_1	${}_1c_1 = C_2 N_5$	206-12
	$+ J_7 R_{C1} D_3'$	206-12
	$+ M_8 U_1' R_{41} X_{41}' I_5' T_2'$	103-9
	$+ M_8 U_1' R_{41}' X_{41} T_2' I_5'$	104-9
	${}_0c_1 = C_2' N_5$	406-34
	$+ R_b I_4$	406-34
	$+ M_8 T_{41}$	405-27
	$+ M_8 U_1' C_2' T_{40}$	405-27
	$+ M_8 U_1' C_3' T_{40}$	406-24
	$+ M_8 U_1' Q_0 Q_1' T_{40}$	506-9

CIRCUIT BOARD LOCATION

CHANNEL REGISTER (Cont)

C_2	${}_1c_2 = C_3 N_5$	415-12
	$+ J_8 R_{C1} D_3'$	415-12
	$+ A_2 M_8 U_1'$	415-10
	$+ M_0$	416-18
	${}_0c_2 = C_3' N_5$	217-18
	$+ R_b I_4$	216-34
	$+ M_8 T_{41}$	216-34
C_3	${}_1c_3 = C_4 N_5$	467-12
	$+ J_9 R_{C1} D_3'$	467-12
	$+ B_2 M_8 U_1'$	467-10
	${}_0c_3 = C_4' N_5$	267-34
	$+ R_b I_4$	267-34
	$+ M_8 T_{41a}$	267-18

CHANNEL REGISTER (Cont)

C_4	$_1c_4 = C_5 N_5$	263-10
	$+ J_{10} R_{c1} D_3'$	264-12
	$+ M_8 U_1' T_1$	264-12
	$_0c_4 = C_5' N_5$	264-34
	$+ R_b I_4$	264-34
	$+ M_8 Q_2 X_1' T_1' U_1'$	164-25
	$+ M_8 Q_2' X_1 T_1' U_1'$	164-25
	$+ M_8 T_{41}$	268-34
C_5	$_1c_5 = C_6 N_5$	477-18
	$+ J_{11} R_{c1} D_3'$	477-10
	$_0c_5 = C_6' N_5$	277-34
	$+ R_b I_4$	277-34

CIRCUIT BOARD LOCATION

CHANNEL REGISTER (Cont)

C_6	${}_1c_6 = G_1 I_1 N_5$	265-12
	$+ Z_1 I_2 N_5$	265-12
	$+ J_{12} R_{C1} D_3'$	264-10
	${}_0c_6 = G_1' I_1 N_5$	464-12
	$+ Z_1' I_2 N_5$	464-12
	$+ R_b I_4$	464-10

CHANNEL REGISTER TIMING FLIP-FLOPS

N_5	${}_1n_5 = D_0' D_{10}' D_{11}' R_{C1}' I_3' N_1 P_5$	176-18
	${}_0n_5 = P_5 P_4 P_3$	276-10

CIRCUIT BOARD
LOCATION

OPERATION CODE REGISTER

D_1	$_1d_1 = D_2\ N_2$	214-34
	$+ N_0\ C_R'$	214-34
	$+ M_8\ U_1'\ C_1'\ T_{41}$	215-27
	$+ M_{10}$	214-19
	$+ I_5\ I_3\ T_{41}$	215-27
	$+ B_{42}\ I_2\ I_4'\ T_{40}$	411-24
	$_0d_1 = D_2'\ N_2$	413-34
	$+ R_{C1}$	412-34
	$+ N_{12}$	412-34
	$+ M_9\ M_4$	413-34
	$+ M_6\ D_6'$	411-34

OPERATION CODE REGISTER (Cont)

D_2	$_1d_2 = D_3\ N_2$	478-34
	$+ R_{C1}$	478-34
	$+ B_{42}\ I_4'\ I_2\ T_{40}$	568-25
	$_0d_2 = D_3'\ N_2$	477-34
	$+ M_4\ D_3'\ T_{41a}$	475-27
	$+ N_0\ C_r'$	477-34
	$+ M_0$	475-27
	$+ M_1\ U_2$	476-10

CIRCUIT BOARD
LOCATION

OPERATION CODE REGISTER (Cont)

D_3	${}_1d_3 = D_4 N_2$	211-18
	$+ M_3 D_2'$	213-18
	$+ J_0 I_2 I_4' T_{40}$	211-24
	$+ B_{42} I_4' I_2$	209-24
	$+ U_2 N_4 M_1'$	212-24
	$+ M_1 T_{41}$	209-34
	${}_0d_3 = D_4' N_2$	212-34
	$+ I_4$	212-34
	$+ M_{10} D_2 U_2 T_{41}$	213-24
	$+ M_6 D_6'$	213-10
D_4	${}_1d_4 = D_5 N_2$	277-18
	$+ R_{C1}$	277-19
	$+ B_{42} I_2 T_{40}$	277-10
	${}_0d_4 = D_5' N_2$	479-18
	$+ R_{C1}' M_3' I_4$	479-18

OPERATION CODE REGISTER (Cont)

D_5	$_1d_5 = D_6\ N_2\ B_{42}'$	266-12
	$+ I_4$	266-12
	$+ M_2\ T_X$	264-18
	$_0d_5 = D_6'\ N_2$	466-18
	$+ B_{42}\ I_4'\ I_2$	566-9
D_6	$_1d_6 = Z_2\ N_2$	474-18
	$+ B_{42}\ I_4'\ I_2\ T_{40}$	468-27
	$_0d_6 = Z_2'\ N_2$	475-34
	$+ I_4\ P_{d3}$	475-34
	$+ R_{c1}$	475-19
	$+ N_{12}$	475-19
	$+ I_5\ T_{41a}$	475-18

CIRCUIT BOARD
LOCATION

OPERATION CODE TIMING FLIP-FLOPS

N_2	${}_1n_2 = I_2\, I_4'\, N_5\, P_4\, P_2\, P_1'$	554-9
	${}_0n_2 = P_2'\, P_1'$	454-24

ORDER REGISTER

Z_1	${}_1z_1 = Z_2\, M_5'$	468-34
	$+ Z_2\, B_{42}'$	468-34
	${}_0z_1 = Z_2'$	279-19
	$+ M_5\, B_{42}$	276-18
Z_2	${}_1z_2 = Z_r$	533
	${}_0z_2 = Z_r'$	533

Z_r, Z_r' } Z CHANNEL READ POWER AMPLIFIER OUTPUTS

Z_w ${}_1z_w = Z_{41}$, ${}_0z_w' = Z_{41}'$ } Z CHANNEL WRITE AMPLIFIER INPUTS

CIRCUIT BOARD LOCATION

ORDER REGISTER (Cont)

Z_{41}	$_1z_{41} = M_r D_0 I_1$	230-12
	$+ Z_1 I_1'$	230-18
	$+ K_0 D_{10} I_1$	230-12
	$_0z_{41} = M_r' D_0 I_1$	231-12
	$+ Z_1' I_1'$	231-18
	$+ K_0' D_{10} I_1$	231-12

COMPUTATION CYCLE COUNTER

Q_1	$_1q_1 = Q_2 N_1 M_8'$	412-12
	$+ M_8 X_{41} T_1'$	412-12
	$_0q_1 = Q_2' N_1 M_8'$	413-12
	$+ I_5 T_{41}$	412-10
	$+ M_8 T_1$	413-12

CIRCUIT BOARD
LOCATION

COMPUTATION CYCLE COUNTER (Cont)

Q_2	$_1q_2 = Q_3 N_1 M_8'$	417-10
	$+ Q_1' M_8 T_{40}$	418-27
	$+ X_{41} M_8 U_1' T_1$	418-27
	$_0q_2 = Q_3' N_1 M_8'$	418-10
	$+ I_5 T_{41}$	420-18
Q_3	$_1q_3 = Q_4 N_1 M_8'$	115-25
	$+ M_8 I_5' U_1' C_2' T_{40}$	115-25
	$_0q_3 = Q_4' N_1 M_8'$	212-10
	$+ U_1 T_{41}$	213-12
	$(+ M_{10} T_{41})^*$	213-12

CIRCUIT BOARD
LOCATION

COMPUTATION CYCLE COUNTER (Cont)

Q_4	$1q_4 = Q_5 N_1 M_8'$	505-25
	$+ M_8 M_0' I_5' U_1 T_{41}$	505-25
	$0q_4 = Q_5' N_1 M_8'$	204-10
	$+ M_8 Q_5' U_2 T_{41}$	104-25
	$+ M_8 U_1' C_1' T_{41}$	104-25

NOTE: * Gates In Parentheses Are Redundant

Q_5	$1q_5 = Q_6 N_1 M_8'$	515-25
	$+ M_8 D_6' X_2' R_2 U_2$	515-25
	$+ M_8 D_6' U_2 X_2' Q_5' T_{40}$	511-18
	$0q_5 = Q_6' N_1 M_8'$	214-10
	$+ M_8 X_2 R_2' U_2$	214-24
	$+ M_8 U_2 R_2' Q_5 T_{40}$	106-18

COMPUTATION CYCLE COUNTER (Cont)

Q_6	$_1q_6 = Q_1 K_q' I_3 P_m' N_1$	172-25
	$+ Q_1' K_q I_3 P_m' N_1$	172-25
	$_0q_6 = Q_1' K_q' N_1$	272-12
	$+ Q_1 K_q N_1$	272-12
	$+ I_3'$	272-34
	$+ P_m$	272-34
	$+ M_8$	271-19
K_q	$_1k_q = T_1 M_0' M_4' M_{10}'$	457-27
	$+ T_1 M_0' M_9'$	457-27
	$_0k_q = Q_1' N_1$	457-18

CIRCUIT BOARD
LOCATION

COMPUTATION CYCLE COUNTER (Cont)

Q_0 $\quad 1q_0 = Q_1 Z_1' I_4' D_3' N_1$ 157-25

$+ Q_1' Z_1 I_4' D_3' N_1$ 157-25

$+ D_3 T_1$ 257-10

$+ R_{gP} I_4$ 257-27

$+ V_{gP} I_4$ 257-27

$+ M_7 D_2$ 258-18

$0q_0 = I_4' T_{41a}$ 256-34

$+ R_{41} D_3 T_1' T_2'$ 257-24

$+ (R_{gm} + T_g) I_4$ 256-12

$+ V_{gm} I_4$ 256-12

$+ M_7 D_2'$ 256-34

CIRCUIT BOARD
LOCATION

ORDER COUNTER LOGIC

G_1	$1g_1 = G_2\ N_4$	263-18
	$+ F_5'\ F_4'\ F_2\ F_1\ I_4\ P_{d3}\ T_{41a}\ T_o'$	163-18
	$0g_1 = G_2'\ N_4$	463-18
G_2	$1g_2 = G_3\ N_4$	259-10
	$+ F_5'\ F_4'\ F_3'\ F_2\ I_4\ P_{d3}\ T_{41a}\ T_o'$	159-18
	$0g_2 = G_3'\ N_4$	460-10
G_3	$1g_3 = G_4\ N_4$	273-18
	$0g_3 = G_4'\ N_4$	473-18
G_4	$1g_4 = G_5\ N_4$	272-18
	$0g_4 = G_5'\ N_4$	472-18
G_5	$1g_5 = G_6\ N_4$	271-18
	$0g_5 = G_6'\ N_4$	471-18

CIRCUIT BOARD LOCATION

ORDER COUNTER LOGIC (Cont)

G_6	$_1g_6 = G_7 N_4$	270-18
	$_0g_6 = G_7' N_4$	470-18
G_7	$_1g_7 = G_8 N_4$	269-18
	$_0g_7 = G_8' N_4$	469-18
G_8	$_1g_8 = G_9 N_4$	453-18
	$_0g_8 = G_9' N_4$	452-18
G_9	$_1g_9 = G_{10} N_4$	260-18
	$_0g_9 = G_{10}' N_4$	460-18
G_{10}	$_1g_{10} = G_{11} N_4$	259-18
	$_0g_{10} = G_{11}' N_4$	459-18
G_{11}	$_1g_{11} = G_{12} N_4$	254-18
	$_0g_{11} = G_{12}' N_4$	253-18

ORDER COUNTER LOGIC (Cont)

CIRCUIT BOARD LOCATION

G_{12} $_1g_{12} = G_1 K_g' F_o' U_1 N_4 TPP'$ 520-18

$+ G_1' K_g U_1 N_4$ 420-27

$+ Z_1 U_2 N_4$ 420-27

$+ A_1 F_o N_4$ 419-10

$+ G_1 K_g' F_o' U_1 N_4 S_{ub}'$ 521-18

$_0g_{12} = G_1' K_g' F_o' U_1 N_4$ 118-9

$+ G_1 K_g U_1 N_4$ 219-27

$+ Z_1' U_2 N_4$ 219-27

$+ A_1' F_o N_4$ 219-10

$+ I_4 (S_{u1} + S_{u2} + S_{u3})$ 220-18

K_g $_1k_g =$ $D_{11} I_1 T_1$ 558-25

$+ G_o I_3 I_4' U_1 T_1$ 558-25

$+ M_2' J_o' H_o I_2 I_4 T_1$ 558-9

$+ (T_o' D_{11} I_1 T_1)$ 458-24

$_0k_g = G_1' N_4$ 258-34

$+ T_{41a}$ 258-34

ORDER COUNTER LOGIC (Cont)

N_4	$_1n_4 = H_0\ I_4'\ U_2\ N_9$	518-25
	$+ I_3\ M_8'\ M_9'\ U_1\ T_2$	519-18
	$+ F_5\ F_0\ I_1\ T_1$	419-27
	$+ M_2'\ H_0\ I_2\ I_4\ T_2$	518-25
	$+ I_1\ N_9\ F_0'$	419-27
	$+ I_4(S_{u1} + S_{u2} + S_{u3})$	419-18
	$+ M_7\ M_9\ T_2$	416-10
	$_0n_4 = P_5\ P_4\ P_3$	216-10

CIRCUIT BOARD
LOCATION

SECTOR SELECTION FLIP-FLOPS

S_0	$_1S_0 = G_1 S_1' I_2' I_3' N_1$	117-25
	$+ G_1' S_1 I_2' I_3' N_1$	118-25
	$+ Z_1 S_1' I_1' I_4' N_1$	117-25
	$+ Z_1' S_1 I_1' I_4' N_1$	118-25
	$+ I_2 R_{C1} T_1$	217-10
	$_0S_0 = T_{41}$	218-19
	$+ E_3 J_3 J_2 J_1 R_{C1} I_2 T_2$	117-18
S_7	$_1S_7 = S_0 D_{10}' M_3' N_1 P_4 P_1'$	173-9
	$+ I_4 I_2'$	273-10
	$+ D_6 D_5 I_2 T_{40}$	273-24
	$_0S_7 = T_{41a}$	273-19

CIRCUIT BOARD LOCATION

SECTOR SELECTION FLIP-FLOPS (Cont)

S_8	${}_1S_8 = S_0\, D_{10}'\, M_3'\, N_1\, P_2'$		117-9
	${}_0S_8 = S_7\, T_{41}$		416-12
	$+ I_1\, D_{10}\, T_{41}$		416-12
	$+ N_{12}$		416-19
S_{11}	${}_1S_{11} = S_0\, N_1$		506-25
	$+ S_0'\, P_6\, P_5\, P_4$		506-25
	$+ J_0$		405-19
	${}_0S_{11} = T_{41}$		205-19

MEMORY READ GATING FLIP-FLOPS

D_0	${}_1d_0 = S_0'\, L_{P1}'\, I_1\, I_4'\, T_{41}$	513-25
	$+ D_3'\, S_0'\, L_{P1}'\, I_2\, T_{41}$	513-25
	$+ D_1'\, S_0'\, L_{P1}'\, I_2\, T_{41}$	514-25
	$+ S_0'\, L_{P1}'\, H_0\, I_1\, T_{41}$	514-25
	$+ D_6'\, D_4'\, S_0'\, L_{P1}'\, I_2\, T_{41}$	513-9
	${}_0d_0 = D_0\, T_{41}$	414-18

CIRCUIT BOARD LOCATION

LOOP GATING FLIP-FLOPS

D_{10} $\quad {}_1d_{10} = S_7' L_{P1} D_{10}' I_3' D_3' T_{41}$ 503-18

$+ S_8' L_{P1} D_{10}' M_6 D_3' T_{41}$ 504-18

$+ S_7' L_{P1} D_{10}' I_3' D_1' T_{41}$ 505-18

$+ S_7' L_{P1} D_{10}' I_3' D_4' T_{41}$ 505-9

$+ I_3 M_6' D_5' T_1$ 403-24

${}_0d_{10} = D_{10} T_{41}$ 403-18

MEMORY WRITE GATING FLIP-FLOPS

D_{11} $\quad {}_1d_{11} = S_{11}' F_0' H_0' I_1 I_4 T_{40}$ 116-9

$+ S_{11}' D_6' D_5 D_4 D_3 D_1 I_2 T_{40}$ 115-18

$+ S_{11}' M_6 D_3' T_{40}$ 212-24

$+ S_{11}' H_0' D_6 D_5 D_4' D_1 I_2 T_{40}$ 116-18

${}_0d_{11} = D_{11} I_1 T_{40}$ 217-27

$+ D_{11} M_6 D_3' T_{40}$ 217-27

$+ M_3 U_2 T_{40}$ 217-12

$+ D_3 Q_4 T_{40}$ 217-12

$+ T_d$ 217-19

CIRCUIT BOARD
LOCATION

OUTPUT TIMING FLIP-FLOPS

N_6	${}_1n_6 = M_7 D_1' U_1 N_9 D_6$	520-9
	${}_0n_6 = P_2 P_1$	421-18

PUNCH FLIP-FLOPS

X_9	${}_1x_9 = M_7 D_3 X_{10}' Q_5 (TPP)' T_{41}$	105-9
	${}_0x_9 = (TPP_p)$	205-34
	$+ T_d$	205-34

TYPE FLIP-FLOPS

X_{10}	${}_1x_{10} = M_7 D_2 D_1' Q_5 (TPP)' T_{41}$	507-9
	${}_0x_{10} = (TPP_t)$	409-34
	$+ T_d$	409-34

CIRCUIT BOARD LOCATION

DIGIT TIME FLIP-FLOPS

T_1	${}_1t_1 = T_{41a}$	267-19
	${}_0t_1 = T_1$	467-19
T_2	${}_1t_2 = T_1$	478-18
	${}_0t_2 = T_2$	478-19
T_{40}	${}_1t_{40} = P_6\ P_5\ P_3\ P_2'\ P_1'$	564-9
	${}_0t_{40} = T_{40}$	463-19
T_{41}	${}_1t_{41} = T_{40}$	415-19
	${}_0t_{41} = T_{41}$	215-18
T_{41a}	${}_1t_{41a} = T_{40}$	266-19
	${}_0t_{41a} = T_{41a}$	265-19

CIRCUIT BOARD LOCATION

COMPUTATION CYCLE TIME FLIP-FLOPS

U_1	${}_1u_1 = I_3'$	272-19
	$+ M_8 U_1' C_1' T_{41a}$	273-27
	$+ M_0 U_2 T_{41a}$	273-27
	$+ M_9 M_1' M_2' U_2 T_{41a}$	173-25
	$+ M_9 B_{42} Q_6 T_{41a}$	173-25
	${}_0u_1 = I_3 U_1 I_5' T_{41a}$	272-24
U_2	${}_1u_2 = I_3 U_1 T_{41} I_5' N_{12}' A_t'$	508-9
	${}_0u_2 = U_2 T_{41}$	409-18
U_{41}	${}_1u_{41} = M_2 D_1' Q_6 Q_4 Q_1' T_{41a}$	576-18
	$+ M_2 D_1 Q_6 Q_4 Q_1 T_{41a}$	576-9
	${}_0u_{41} = U_{41} T_{41a}$	476-18

CIRCUIT BOARD LOCATION

OPERATION TYPE FLIP-FLOPS

M_0	${}_1m_0 = J_0' D_6' D_5' D_4' D_3' I_2 N_{11}$	151-18
	$+ M_8 U_1' C_1' T_{41a} I_5' M_0'$	151-9
	${}_0m_0 = N_{12}$	452-19
	$+ T_d$	452-12
	$+ M_0 U_2 T_{41a}$	452-12
M_1	${}_1m_1 = J_0' D_6' D_5' D_4 D_1 I_2 N_{11}$	167-18
	$+ M_{10} D_2 U_2 T_{41a}$	267-24
	${}_0m_1 = N_{12}$	267-12
	$+ T_d$	267-12
	$+ M_1 B_{42} T_{41a}$	267-10
	$+ N_7 T_{41a}$	263-18

CIRCUIT BOARD
LOCATION

OPERATION TYPE FLIP-FLOPS (Cont)

M_2	${}_1m_2 = J_0' D_6' D_5 D_4' I_2 I_4' N_{11}$	554-18
	$+ F_5 F_4 F_3 F_2 A_0' N_0 P_{d3} T_{41a}$	553-18
	$+ M_{10} D_2' U_2 T_{41a}$	455-27
	$+ I_5 I_3 T_{41a}$	455-27
	${}_0m_2 = T_d$	455-34
	$+ U_{41} T_{41a}$	455-34
M_3	${}_1m_3 = J_0' D_6' D_5 D_4 I_2 N_{11}$	513-18
	$+ R_{C1} I_2 N_{11}$	413-10
	${}_0m_3 = N_{12}$	213-34
	$+ T_d$	213-34

CIRCUIT BOARD LOCATION

OPERATION TYPE FLIP-FLOPS (Cont)

M_4	${}_1m_4 = J_0' D_6 D_5' D_4' I_2 N_{11}$	120-9
	$+ D_5' D_4' D_3 I_2 N_{11} J_0'$	121-9
	$+ M_0 M_8 U_2 T_{41}$	220-24
	$+ M_9 B_{42} Q_6 T_{41}$	219-24
	${}_0m_4 = N_{12}$	222-34
	$+ M_4 D_2 D_3' T_{41}$	222-27
	$+ T_d$	222-34
	$+ M_8 M_0' Q_5 U_2 T_{41}$	119-18
	$+ M_8 M_0' U_1' C_1' T_{41}$	122-9
	$+ M_9 U_2 T_{41}$	222-10
	$+ I_5 T_{41}$	222-27
M_5	${}_1m_5 = J_0' D_6 D_5' D_4 I_2 N_{11}$	169-18
	${}_0m_5 = N_{12}$	270-19
	$+ T_d$	269-19
	$+ D_3' U_2 T_{40}$	274-10

OPERATION TYPE FLIP-FLOPS (Cont)

M_6	$_1m_6 = J_0' D_6 D_5 D_4' I_2 N_{11}$	167-9
	$+ M_4 D_3' D_2 D_1' T_{41a}$	166-9
	$+ M_1 N_7 T_{41a}$	268-10
	$_0m_6 = N_{12}$	467-34
	$+ I_4$	467-34
M_7	$_1m_7 = J_0' D_6 D_5 D_4 I_2 N_{11}$	154-9
	$+ J_0' D_6' D_5' D_4 D_1' I_2 N_{11}$	154-18
	$+ D_5' T_X T_{41a} N_{12}'$	253-24
	$+ J_{13} I_4'$	251-18
	$_0m_7 = N_{12}$	255-34
	$+ I_4$	254-34
	$+(TBS)\ J_{13}'$	254-34
	$+ TPP_P\ X_{10}'$	255-34
M_8	$_1m_8 = D_5' D_3 D_4' D_1' I_2 N_{11} J_0'$	514-18
	$_0m_8 = M_0 U_2 T_{41}$	414-12
	$+ I_5 T_{41}$	414-12
	$+ T_d$	422-19

OPERATION TYPE FLIP-FLOPS (Cont)

M_9	${}_1m_9 = D_6' D_5' D_3 D_1 I_2 N_{11} J_0' D_4'$	578-18
	$+ M_7 D_6' U_1$	478-10
	${}_0m_9 = B_{42} Q_6 T_{41a}$	475-12
	$+ M_7 U_2$	475-12
	$+ T_d$	472-19
M_{10}	${}_1m_{10} = M_9 M_4 U_2 T_{41a}$	459-24
	${}_0m_{10} = M_{10} U_2 T_{41a}$	459-10
	$+ T_d$	459-19

CIRCUIT BOARD
LOCATION

A REGISTER

$A_1 \quad {}_1a_1 = A_2 I_3' E_{20}' E_{30}' E_{40}' T_{41}'$ 502-9

$+ A_2 I_3 M_2' M_{10}' T_{41}'$ 503-25

$+ A_2 M_2 E_{30}' U_{41}'$ 502-25

$+ A_3 D_1' U_{41}$ 402-27

$+ R_2 D_1 U_{41} T_{41}'$ 402-27

$+ A_2 N_0 P_0 E_1' T_1'$ 502-25

$+ R_2 M_{10}$ 402-12

$+ F_1 E_{30}$ 402-12

0^a1 (Next Page)

A REGISTER (Cont)

$$
\begin{aligned}
{}_{0}a_1 = {} & A_2' I_3' E_{30}' && 202\text{-}27 \\
& + A_2' I_3 M_2' M_{10}' && 202\text{-}24 \\
& + A_2' M_2 E_{30}' U_{41}' && 202\text{-}27 \\
& + A_3' D_1' U_{41} && 202\text{-}10 \\
& + R_2' D_1 U_{41} && 102\text{-}25 \\
& + N_0 P_0 T_{41} E_2' && 102\text{-}25 \\
& + F_1' E_{30} && 203\text{-}34 \\
& + I_2 T_{41} && 203\text{-}34 \\
& + R_2' M_{10} && 202\text{-}34 \\
& + U_{41} T_{41} && 202\text{-}34 \\
& + M_1 T_{41} && 205\text{-}18
\end{aligned}
$$

CIRCUIT BOARD LOCATION

A REGISTER (Cont)

A_2	$_1a_2 = A_3 I_3' E_{20}' E_{40}'$	416-27
	$+ A_3 I_3 M_2'$	416-27
	$+ A_3 M_2 T_{41}'$	417-27
	$+ R_{42} M_2 D_2 U_{41}' T_{41}$	517-9
	$+ A_3 U_{41}$	417-18
	$+ R_1 E_{20} I_4$	417-12
	$+ B_1 E_{40} T_{41}'$	417-27
	$+ A_3 E_{40} T_{41}$	417-12
	$_0a_2 = A_3' I_3' E_{20}' E_{40}'$	218-27
	$+ A_3' I_3 M_2'$	218-27
	$+ A_3' M_2 T_{41}'$	219-12
	$+ A_3' U_{41}$	218-18
	$+ R_1' E_{20} I_4$	219-12
	$+ B_1' E_{40} T_{41}'$	218-12
	$+ A_3' E_{40} T_{41}$	218-12

A REGISTER (Cont)

A_3 $\quad {}_1a_3 = A_r$ 529

$\quad {}_0a_3 = A_r'$ 529

A_r, A_r' } A CHANNEL READ POWER AMPLIFIER OUTPUTS

A_w $\quad {}_1a_w = A_{39}$, $\quad {}_0a_w = A_{39}'$ } A CHANNEL WRITE AMPLIFIER INPUTS

A_{39} $\quad {}_1a_{39} = A_{40} M_1'$ 226-34

$\quad + A_{40} U_1$ 226-34

$\quad + A_{40} T_1$ 226-18

$\quad + S M_1 U_1' T_1'$ 129-9

$\quad {}_0a_{39} = A_{40}' M_1'$ 225-34

$\quad + A_{40}' U_1$ 225-34

$\quad + A_{40}' T_1$ 225-18

$\quad + S' M_1 U_1' T_1'$ 225-24

CIRCUIT BOARD LOCATION

A REGISTER (Cont)

A_{40}	$_1a_{40} = A_{41} M_0' M_1' U_{41}'$	419-24
	$+ A_{41} M_0 U_1$	419-12
	$+ S M_0 U_2$	419-12
	$+ A_{41} M_1 U_1$	420-12
	$+ M_1 D_2 U_1 T_{41}$	420-24
	$+ K_a M_1 U_1'$	420-12
	$+ A_{41} D_1' U_{41}$	421-12
	$+ S D_1 U_{41}$	421-12
	$_0a_{40} = A_{41}' M_0' M_1' U_{41}'$	220-27
	$+ A_{41}' M_0 U_1$	220-27
	$+ S' M_0 U_2$	220-10
	$+ A_{41}' M_1 T_{41}'$	220-12
	$+ K_a' M_1 U_1'$	220-12
	$+ A_{41}' D_1' U_{41}$	221-12
	$+ S' D_1 U_{41}$	221-12
	$+ A_{41}' M_1 D_3 U_1$	123-18

A REGISTER (Cont)

A_{41}	$_1a_{41} =$	$A_1 I_3' P_0'$	279-27
		$+ B_1 M_0 D_1'$	277-27
		$+ S M_0 J_0' U_2 T_1$	179-25
		$+ S' M_0 J_0 U_2 T_1$	179-25
		$+ R_{43} M_1 U_2 T_1$	278-27
		$+ S M_2 T_X' U_{41}' T_1'$	177-25
		$+ R_{41} U_{41} T_1$	278-27
		$+ R_{42} D_1' U_{41}$	279-12
		$+ A_1 B_1 M_3 D_3' D_1$	177-25
		$+ B_1 M_3 D_2' D_1'$	277-27
		$+ R_1 M_3 D_3 D_1$	279-24
		$+ A_1 M_3 D_2 D_1'$	279-27
		$+ A_1 M_4 D_2' U_1$	278-24
		$+ A_1 M_4 U_1' T_{41a}$	276-27

$_1a_{41}$ (Cont Next Page)

A REGISTER (Cont)

$1^{a}41$ (Cont)

$+ A_2 M_4 D_1' U_1' T_1' T_{40}' T_{41a}'$	178-18
$+ A_{42} M_4 D_1 U_1' T_{41a}'$	179-9
$+ A_1 M_4 D_3' D_2 D_1'$	178-25
$+ R_1 M_4 D_3' D_2 D_1$	178-25
$+ U_{41} T_X$	279-34
$+ M_1 D_3 U_1 X_1$	275-24
$+ F_1 N_0' P_0 T_1'$	277-24
$+ R_{42} E_{40} T_{41a}'$	279-12
$+ A_{42} E_{40} T_{41a}$	276-27
$+ A_{42} M_2 T_X U_1' U_{41}'$	176-25
$+ A_1 M_4 D_3 U_1$	176-25
$+ A_{42} M_{10}$	279-18
$+ A_1 M_0' M_1' M_2' M_3' M_4' M_{10}' P_0'$	179-18

$0^{a}41$ (Next Page)

A REGISTER (Cont)

$$
\begin{aligned}
{}_0a_{41} = {} & A_1' I_3' P_0' && 479\text{-}12 \\
& + B_1' M_0 D_1' && 479\text{-}27 \\
& + M_2 E_{20}' U_{41}' T_1 && 478\text{-}27 \\
& + S' M_2 T_X' U_{41}' && 578\text{-}25 \\
& + R_{41}' U_{41} T_1 && 479\text{-}12 \\
& + R_{42}' D_1' U_{41} && 478\text{-}12 \\
& + B_1' M_3 D_1 D_3' && 479\text{-}27 \\
& + R_1' M_3 D_3 D_1 && 577\text{-}25 \\
& + A_1' M_3 D_2 && 477\text{-}27 \\
& + A_1' M_4 D_2' U_1 && 477\text{-}27 \\
& + A_1' M_4 U_1' T_{41a} && 476\text{-}27 \\
& + M_4 D_1' T_1 && 478\text{-}12 \\
& + U_1' M_4 D_1' T_{40} && 577\text{-}25 \\
& + A_2' M_4 D_1' U_1' T_{41a}' && 578\text{-}25 \\
& + A_4' M_4 D_1 U_1' T_{41a}' && 579\text{-}1
\end{aligned}
$$

${}_0{}^a41$ (Cont Next Page)

CIRCUIT BOARD LOCATION

A REGISTER (Cont)

$0^{a}41$ (Cont)

$+ A_1' M_4 D_3' D_2 D_1'$	579-25
$+ R_1' M_4 D_3' D_2 D_1$	579-25
$+ F_1' N_0' P_0 T_1'$	479-24
$+ R_{42}' E_{40} T_{41a}'$	477-12
$+ A_{42}' E_{40} T_{41a}$	476-27
$+ M_1 U_1 X_1'$	471-12
$+ A_1' M_4 D_3 U_1$	478-24
$+ M_0 U_1 T_1$	478-27
$+ M_1 D_3' T_1$	477-12
$+ A_{42}' M_{10}$	479-34
$+ A_1' M_0' M_1' M_2' M_3' M_4' M_{10}' P_0'$	579-18
$+ A_{42}' M_2 T_X U_{41}'$	477-24
$+ B_1' M_3 D_2' D_1'$	476-24
$+ N_0 P_0 E_2' E_3'$	475-24

CIRCUIT BOARD LOCATION

A REGISTER (Cont)

A_{42} $_1a_{42} = A_1 M_4 D_2' T_{41a}'$ 158-25

$+ F_3' F_1 N_0 P_0 E_2' E_1$ 158-18

$+ Z_1 I_2 I_4' N_9$ 158-25

$+ A_2 M_{10}$ 258-12

$+ S M_2$ 258-12

$+ F_{c2}' D_4 I_4$ 258-10

$_0a_{42} = A_1' M_4 D_2'$ 458-12

$+ M_4 T_{41a}$ 458-34

$+ F_3' F_1' N_0 P_0 E_2' E_1$ 558-18

$+ H_0' I_2 T_1$ 458-12

$+ A_2' M_{10}$ 458-34

$+ S' M_2$ 458-18

CIRCUIT BOARD LOCATION

B REGISTER

B_1	${}_1b_1 = B_2\, M_{10}'\, N_7'\, E_{30}'$	178-9
	$+ E_{20}$	278-19
	$+ X_2\, M_{10}$	278-34
	$+ A_2\, N_7$	278-34
	${}_0b_1 = B_2'\, M_{10}'\, N_7'\, E_{20}'\, E_{30}'$	177-9
	$+ E_{30}\, I_4\, T_{41a}$	277-12
	$+ X_2'\, M_{10}$	277-12
	$+ A_2'\, N_7$	278-18
B_2	${}_1b_2 = B_r$	530
	${}_0b_2 = B_r'$	530

B_r, B_r' } B CHANNEL READ POWER AMPLIFIER OUTPUTS

B REGISTER (Cont)

B_W $_1b_W = B_{40}$
$_0b_W = B_{40}'$
} B CHANNEL WRITE AMPLIFIER INPUTS

B_{40} $_1b_{40} = B_{41} M_0' M_3'$ 224-27

$+ S M_0 D_1$ 225-27

$+ B_{41} D_3' D_2' D_1'$ 224-27

$+ B_{41}' M_0 D_2 D_1'$ 225-27

$+ B_{41} M_3 D_2$ 226-27

$+ B_{41}' M_3 D_3 D_2'$ 226-27

$_0b_{40} = B_{41}' M_0' M_3'$ 227-27

$+ S' M_0 D_1$ 224-24

$+ B_{41}' D_3' D_2' D_1'$ 227-27

$+ B_{41} M_0 D_2 D_1'$ 125-18

$+ B_{41}' M_3 D_2$ 228-27

$+ B_{41} M_3 D_3 D_2'$ 228-27

B REGISTER (Cont)

B_{41} $_1b_{41} = M_R D_0 I_2$ 260-27

$+ K_0 D_{10} I_2$ 261-27

$+ B_1 T_X' M_0' M_8' M_{10}' I_3$ 161-9

$+ T_X' B_1 R_{C1}' I_4 P_0'$ 161-25

$+ N_0 P_0 E_3' E_2' T_2$ 159-25

$+ B_{42} N_0 P_0 E_2$ 159-25

$+ S E_{40}$ 261-34

$+ E_{40} T_{41a}$ 261-34

$+ A_1 A_d I_2 R_{C1}$ 261-24

$+ X_1 X_d I_2 R_{C1}$ 260-24

$+ R_1 R_d I_2 R_{C1}$ 260-27

$+ Z_1 Z_d I_2 R_{C1}$ 261-27

$+ F_1 H_4' T_X$ 261-12

$+ F_1 R_{C2}' T_X$ 261-12

$_1b_{41}$ (Cont Next Page)

B REGISTER (Cont)

$_1b_{41}$ (Cont)	$+ B_{42} M_{10}$	261-10
	$+ B_{42} A_{42} R_{C2} H_4 T_X$	161-25
	$+ B_1 M_8 Q_4' M_0'$	160-25
	$+ B_2 M_8 Q_4 T_1' T_{40}' T_{41a}'$	160-9
	$+ B_1 M_8 Q_4 T_{41a}$	160-25
$_0b_{41} =$	$M_R' D_0 I_2$	460-27
	$+ K_0' D_{10} I_2$	459-27
	$+ B_1' M_0' M_8' M_{10}' I_3 T_X'$	561-9
	$+ B_1' R_{C1}' I_4 P_0' T_X'$	561-25
	$+ N_0 P_0 E_3' E_2' P_1'$	561-25
	$+ B_{42}' N_0 P_0 E_2$	461-24
	$+ S' E_{40} T_{41a}'$	461-10
	$+ X_1' X_d I_2 R_{C1}$	459-27
	$+ A_1' A_d I_2 R_{C1}$	560-25

$_0b_{41}$ (Cont Next Page)

CIRCUIT BOARD
LOCATION

B REGISTER (Cont)

$0^{b}41$ (Cont)

$+ R_1' R_d I_2 R_{c1}$	460-27
$+ Z_1' Z_d I_2 R_{c1}$	460-24
$+ F_1' H_4' T_x$	461-12
$+ F_1' R_{c2}' T_x$	461-12
$+ B_{42}' A_{42} R_{c2} H_4 T_x$	560-25
$+ B_1' M_8 Q_4' M_0'$	559-25
$+ B_2' M_8 Q_4 T_{41a}'$	559-25
$+ B_1' M_8 Q_4 T_{41a}$	461-27
$+ M_8 Q_4 T_1$	461-27
$+ M_8 Q_4 T_{40}$	460-12
$+ B_{42}' M_{10}$	460-12

B REGISTER (Cont)

B_{42} $_{1}b_{42} = A_{40}\ M_2\ M_4'\ U_1\ T_2$ 109-25

$+ R_1\ H_4\ R_{C2}\ N_{12}'$ 208-24

$+ Z_1'\ G_0\ I_2\ P_6'\ P_5\ P_4'\ P_3'\ P_2'$ 110-18

$+ Z_{41}'\ G_0'\ T_1\ I_2\ H_0'$ 109-25

$+ B_2\ N_0\ P_0\ E_2\ T_{41}'$ 110-25

$+ I_4\ R_{C1}$ 210-34

$+ B_2\ M_{10}\ T_{41}'$ 110-25

$+ M_1\ M_4\ Q_6\ Q_3\ Q_2\ Q_1\ T_{41}$ 109-18

$+ M_4\ Q_6\ Q_4\ Q_1\ T_{41}$ 110-9

$+ M_1\ D_3\ U_1$ 210-24

$0^{b}42$ (Next Page)

B REGISTER (Cont)

$$
\begin{aligned}
{}_{0}b_{42} = {} & I_4' R_1' R_{C2} && 510\text{-}25 \\
& + B_2' N_0 P_0 E_2 && 510\text{-}25 \\
& + B_2' M_{10} && 412\text{-}18 \\
& + M_9 B_{42} T_{41} && 410\text{-}12 \\
& + N_{11}' I_3' T_{41} && 410\text{-}12 \\
& + N_{12} && 412\text{-}19 \\
& + M_7 U_1 && 405\text{-}12
\end{aligned}
$$

CIRCUIT BOARD
LOCATION

R REGISTER

R_1	$_1r_1 = R_2 I_3' P_0'$	269-27
	$+ M_0 T_1$	269-34
	$+ R_2 M_1 U_1$	273-12
	$+ R_2 M_1 T_{41a}$	270-27
	$+ M_2 U_1$	270-34
	$+ R_2 M_0' M_1' M_2' M_{10}' I_3$	170-18
	$+ R_2 N_0 P_0 E_2$	269-27
	$+ E_{40} T_1$	269-34
	$+ M_{10} T_1$	270-34
	$+ M_2 T_X U_{41}' T_{41a}$	270-27
	$+ R_{42} M_2 U_{41}' T_X T_1'$	170-9

0^r1 (Next Page)

CIRCUIT BOARD LOCATION

R REGISTER (Cont)

$_0r_1 = R_2' I_3' P_0'$	470-27
$+ M_0 T_{40}$	470-34
$+ M_1 R_2' T_{41a}$	469-27
$+ D_1 U_{41}$	470-34
$+ R_2' M_1 U_1$	468-10
$+ E_0' U_{41}$	469-34
$+ U_{41} T_{41a}$	469-27
$+ R_{42}' M_2 T_X T_1' T_{41a}'$	570-9
$+ R_2' M_0' M_1' M_2' M_{10}' I_3$	571-4
$+ R_2' N_0 P_0 E_3'$	470-27
$+ E_{40} T_{41a}$	469-34
$+ M_{10} T_{40}$	470-10
$+ E_0' M_2 T_X T_1$	470-24

R_2	$_1r_2 = R_r$	531
	$_0r_2 = R_r'$	531

R REGISTER (Cont)

R_r, $R_{r'}$ } R CHANNEL READ POWER AMPLIFIER OUTPUTS

R_w $\quad {}_1r_w = R_{40}$, $\quad {}_0r_w = R_{40}'$ } R CHANNEL WRITE AMPLIFIER INPUTS

R_{40} $\quad {}_1r_{40} = R_{41}\, E_{30}'\, U_{41}'$ 229-10

$+ S\, U_{41}\, T_x'\, M_9'$ 127-25

$+ S\, E_{30}\, I_4$ 127-25

$+ R_{41}\, T_x$ 229-34

$+ R_{41}\, M_9$ 229-34

${}_0r_{40} = R_{41}'\, E_{30}'\, U_{41}'$ 228-10

$+ S'\, U_{41}\, T_x'\, M_9'$ 126-25

$+ S'\, E_{30}\, I_4$ 126-25

$+ R_{41}'\, T_x$ 228-18

$+ R_{41}'\, M_9$ 229-18

CIRCUIT BOARD
LOCATION

R REGISTER (Cont)

R_{41}	$_1r_{41} = R_1 I_3' P_0'$	267-27
	$+ R_{42} M_0 Q_3'$	164-9
	$+ A_2 M_1 D_3'$	265-27
	$+ R_2 M_1 B_{42}' U_1' T_{40}' T_{41a}$	165-9
	$+ A_3 M_1 B_{42}' T_{40}$	265-27
	$+ A_{41} M_1 B_{42}' T_{41a}$	167-25
	$+ R_{42} M_2 U_{41}'$	163-9
	$+ B_{42} T_X' U_{41} M_9'$	167-25
	$+ R_1 M_3 D_3'$	266-27
	$+ M_R L_{P1}' D_6' D_4 I_3 I_4' D_1' U_1$	166-18
	$+ K_0 L_{P1} D_6' D_4 I_3 I_4' D_1' U_1$	165-18
	$+ A_1 M_3 D_3 D_1$	264-24
	$+ R_1 M_3 D_2 U_1'$	266-27
	$+ R_1 M_4 D_3' D_2'$	264-27
	$+ R_1 M_4 D_3' D_1'$	267-27

$_1r_{41}$ (Cont Next Page)

R REGISTER (Cont)

1^r41 (Cont)	$+ A_1 M_4 D_3' D_2 D_1$	169-9
	$+ R_1 K_a' M_4 D_3 T_{41a}$	166-25
	$+ R_1' K_a M_4 D_3 T_{41a}'$	166-25
	$+ E_0 M_4 D_3 T_{41a}$	265-24
	$+ S E_{20} I_4$	266-10
	$+ R_{42} E_{40}$	266-18
	$(+ R_1 M_3 I_4)$	264-27
	$+ X_1 M_0 Q_3$	263-27
	$+ M_9 M_{10}' B_{42} X_1 M_4'$	165-25
	$+ R_1 M_0' M_1' M_2' M_3' M_4' M_9' I_3$	164-18
	$+ X_2 T_X U_{41} T_{40}' T_{41a}'$	165-25
	$+ X_1 T_X U_{41} T_{41a}$	263-27
	$+ R_1 M_1 D_3 U_1$	263-24

0^r41 (Next Page)

R REGISTER (Cont)

$$0r_{41} = R_1' I_3' P_0'$$ 467-27

$$+ R_{42}' M_0 Q_3'$$ 465-10

$$+ A_2' M_1 D_3'$$ 466-27

$$+ R_2' M_1 B_{42}' U_1' T_{40}' T_{41a}'$$ 567-9

$$+ A_3' M_1 B_{42}' T_{40}$$ 466-27

$$+ M_{10}$$ 465-19

$$+ A_{41}' M_1 B_{42}' T_{41a}$$ 566-25

$$+ R_{42}' M_2 U_{41}'$$ 465-12

$$+ B_{42}' T_x' U_{41} M_9'$$ 566-25

$$+ R_1' M_3 D_3'$$ 465-27

$$+ M_r' L_{P1}' D_6' D_4 I_3 I_4' D_1' U_1$$ 565-18

$$+ K_0' L_{P1} D_6' D_4 I_3 I_4' D_1' U_1$$ 566-18

$$+ A_1' M_3 D_3 D_1$$ 465-27

$$+ R_1' M_3 D_2 U_1'$$ 467-27

$$+ R_1' M_4 D_3' D_2'$$ 464-27

0^r41 (Cont Next Page)

R REGISTER (Cont)

0^r41 (Cont)

$+ X_1' M_0 Q_3$	463-27
$+ R_1' M_4 D_3' D_1'$	564-25
$+ A_1' M_4 D_3' D_2 D_1$	564-25
$+ R_1' K_a' M_4 D_3 T_{41a}$	565-25
$+ R_1 K_a M_4 D_3 T_{41a}'$	565-25
$+ E_0' M_4 D_3 T_{41a}$	466-24
$+ N_0 P_0 E_3' E_2'$	465-24
$+ S' E_{20} I_4$	465-12
$+ R_{42}' E_{40}$	465-18
$(+ R_1' M_3 I_4)$	464-27
$+ M_9 B_{42} X_1' M_4'$	464-24
$+ R_1' M_0' M_1' M_2' M_3' M_4' M_9' I_3$	567-18
$+ X_2' T_X U_{41} T_{41a}'$	463-24
$+ X_1' T_X U_{41} T_{41a}$	463-27
$+ R_1' M_1 D_3 U_1$	467-24

R REGISTER (Cont)

R 12	$1r_{42} = R_2 M_0$	476-12
	$+ R_{43} M_2$	476-34
	$+ R_2 E_{40}$	476-34
	$0r_{42} = R_2' M_0$	276-12
	$+ R_{43}' M_2$	276-34
	$+ R_2' E_{40}$	276-34
R_{43}	$1r_{43} = R_2 T_x' T_{40}' T_{41a}'$	271-27
	$+ A_1 B_1 I_3 \underline{M_4'^{(M41)} T_{41a}} T_x' U_1$	171-9
	$+ A_1' B_1' I_3 \underline{M_4'^{(M41)} T_{41a}} U_1$	171-25
	$+ E_0' K_a' U_1' \underline{M_4'^{(M41)} T_{41a}}$	171-25
	$+ A_1 K_a U_1' \underline{M_4'^{(M41)} T_{41a}}$	271-24
	$+ R_2 U_1' T_{41a}'$	271-27
	$0r_{43} = R_2' T_{41a}'$	471-10
	$+ M_1' E_0 K_a' T_{41a}$	471-24
	$+ A_1' K_a U_1' T_{41a}$	471-27
	$+ T_{40}$	471-19

CIRCUIT BOARD
LOCATION

X REGISTER

X_1	${}_1X_1 = X_2$	476-19
	${}_0X_1 = X_2'$	477-19
X_2	${}_1X_2 = X_R$	532
	${}_0X_2 = X_R'$	532

X_R, X_R' } X CHANNEL READ POWER AMPLIFIER OUTPUTS

X_W	${}_1X_W = X_{40}$	} X CHANNEL WRITE AMPLIFIER INPUTS
	${}_0X_W = X_{40}'$	

X_{40}	${}_1X_{40} = X_{41}\ M_{10}'$	228-34
	$+ S\ M_{10}$	228-34
	${}_0X_{40} = X_{41}'\ M_{10}'$	227-34
	$+ S'\ M_{10}$	227-34

CIRCUIT BOARD
LOCATION

X REGISTER (Cont)

X_{41} $_1X_{41} = M_R\ L_{P1}'\ M_4\ U_1$ 519-25

$+ K_0\ L_{P1}\ M_4\ U_1$ 520-25

$+ X_1\ M_8\ C_4'\ Q_4\ T_{41}'$ 519-25

$+ X_1'\ M_8\ C_4\ Q_4\ T_{41}'$ 520-25

$+ M_8\ Q_2\ Q_4\ T_{41}$ 521-25

$+ X_1\ Q_4'\ U_1'\ M_{10}'$ 421-24

$+ X_1\ M_9\ M_{10}'\ M_4'$ 422-24

$+ S\ M_{10}\ U_2\ T_1$ 423-24

$+ A_1\ I_1\ I_5$ 421-10

$+ X_1\ M_8'\ M_9'\ I_5'\ U_1'$ 521-25

$+ M_4\ U_1\ Q_2$ 423-10

$+ G_1\ M_5\ U_1\ N_4$ 413-24

$+ G_0\ M_5\ T_2$ 414-24

$+ X_1\ D_5\ I_5'$ 414-10

$_1X_{41}$ (Cont Next Page)

X REGISTER (Cont)

$1^{X}41$ (Cont)

$+ A_1 M_1 D_3 U_1$ 412-24

$+ X_1 T_X$ 414-34

$+ M_5 T_{41}$ 419-34

$+ X_1 I_3' I_5'$ 410-24

$_0X_{41} = M_{21}' L_{P1}' M_4 U_1 Q_2'$ 121-25

$+ K_0' L_{P1} M_4 U_1 Q_2'$ 120-25

$+ X_1' M_8 C_4' Q_4 T_{41}'$ 121-25

$+ X_1 M_8 C_4 Q_4 T_{41}'$ 120-25

$+ M_8 Q_2' Q_4 T_{41}$ 222-24

$+ X_1' Q_4' U_1' M_{10}'$ 221-24

$+ X_1' M_9 M_{10}' M_4'$ 122-25

$+ M_{10} U_1$ 222-18

$+ A_1' I_1 I_5$ 221-10

$+ X_1' M_8' M_9' I_5' U_1'$ 122-25

$0^{X}41$ (Cont Next Page)

CIRCUIT BOARD LOCATION

X REGISTER (Cont)

$0^{x}41$ (Cont)	$+ G_1' N_4 U_1 M_5$	221-27
	$+ M_5 G_0' T_2$	221-27
	$+ X_1' D_5 I_5'$	223-10
	$+ A_1' M_1 D_3 U_1$	223-27
	$+ X_1' T_X$	223-18
	$+ M_5 U_1 N_4' P_2$	223-24
	$+ X_1' I_3' I_5'$	123-9

L MEMORY LOOP

L_1	$_1\ell_1 = L_2$	215-19
	$_0\ell_1 = L_2'$	216-19
L_2	$_1\ell_2 = L_{r8}$	535
	$_0\ell_2 = L_{r8}'$	535
L_{2a}	$_1\ell_{2a} = L_{r4}$	
	$_0\ell_{2a} = L_{r4}'$	

L_{r8}, $L_{r8'}$ } L Channel Read Power Amplifier Outputs From 8-word Read Head

L_{r4}, $L_{r4'}$ } L Channel Read Power Amplifier Outputs From 4-word Read Head

L_w $1^1w = L_{41}$, $0^1w = L_{41'}$ } L Channel Write Amplifier Inputs

L_{41} $1^1_{41} = M_R M_6 D_3 D_2' D_1' T_1'$ 128-9

$+ A_1 M_3 D_3 D_1 L_{P1} L_{P2}' T_1'$ 127-18

$+ B_1 M_6 D_3' L_{P2}' D_{10}$ 126-18

$+ L_1 M_3' M_6'$ 228-12

$+ L_1 M_3 D_1'$ 228-12

$+ L_1 D_2$ 224-18

$+ L_1 D_1 L_{P1}'$ 227-12

$+ L_1 D_1 L_{P2}$ 226-12

$+ L_1 D_3' D_{10}'$ 227-12

$+ L_1 D_3' L_{P2}$ 226-12

0^141 (Next Page)

CIRCUIT BOARD LOCATION

L MEMORY LOOP (Cont)

$$
\begin{aligned}
o\ell_{41} = {} & M_R' M_6 D_3 D_2' D_1' && 124\text{-}25 \\
& + A_1' M_3 D_3 D_1 L_{P1} L_{P2}' && 124\text{-}18 \\
& + B_1' M_6 D_3' L_{P2}' D_{10} && 124\text{-}25 \\
& + L_1' M_3' M_6' && 125\text{-}25 \\
& + L_1' M_3 D_1' && 125\text{-}25 \\
& + L_1' D_2 && 224\text{-}34 \\
& + L_1' D_1 L_{P1}' && 225\text{-}12 \\
& + L_1' D_1 L_{P2} && 224\text{-}12 \\
& + L_1' D_3' D_{10}' && 225\text{-}12 \\
& + L_1' D_3' L_{P2} && 224\text{-}12 \\
& + T_1 && 224\text{-}34
\end{aligned}
$$

L MEMORY LOOP (Cont)

L_{P1}	${}_1\ell_{P1} = D_0' D_{11}' I_3' R_{C1}' T_1$		570-25
	$+ (V_d + L_d) I_2 R_{C1}$		570-25
	${}_0\ell_{P1} = G_1' I_1 N_5$		270-10
	$+ Z_1' I_1' N_5$		270-12
	$+ M_d I_2 R_{C1}$		270-12
L_{P2}	${}_1\ell_{P2} = G_1 N_1 P_5 P_2' I_1$		112-9
	$+ Z_1 N_1 P_5 P_2' I_1' I_4'$		112-18
	$+ V_d I_4$		212-18
	${}_0\ell_{P2} = G_1' N_1 P_5 I_1$		512-25
	$+ Z_1' N_1 P_5 I_1' I_4'$		512-25
	$+ L_d I_4$		413-18

CIRCUIT BOARD
LOCATION

V MEMORY LOOP

V_1	$_1V_1 = V_2$	413-19
	$_0V_1 = V_2'$	414-19
V_2	$_1V_2 = V_{r8}$	537
	$_0V_2 = V_{r8}'$	537
V_{2a}	$_1V_{2a} = V_{r4}$	
	$_0V_{2a} = V_{r4}'$	

V_{r8}, V_{r8}' } V CHANNEL READ POWER AMPLIFIER OUTPUTS FROM 8-WORD READ HEAD

V_{r4}, V_{r4}' } V CHANNEL READ POWER AMPLIFIER OUTPUTS FROM 4-WORD READ HEAD

V_w $_1V_w = V_{41}$, $_0V_w = V_{41}'$ } V CHANNEL WRITE AMPLIFIER INPUTS

V MEMORY LOOP (Cont)

V_{41} $\quad {}_{1}V_{41} = M_{\Lambda}\, M_6\, D_2\, D_1'\, T_1'$ 129-25

$+ A_1\, M_3\, D_3\, D_1\, L_{P1}\, L_{P2}\, T_1'$ 131-18

$+ B_1\, M_6\, D_3'\, L_{P2}\, D_{10}$ 129-25

$+ V_1\, M_3'\, M_6'$ 229-27

$+ V_1\, M_3\, D_1'$ 229-27

$+ V_1\, M_6\, D_3\, D_2'$ 128-25

$+ V_1\, D_1\, L_{P1}'$ 229-12

$+ V_1\, D_1\, L_{P2}'$ 229-12

$+ V_1\, D_3'\, D_{10}'$ 128-25

$+ V_1\, D_3'\, L_{P2}'$ 231-10

$0^{V}41$ (Next Page)

CIRCUIT BOARD LOCATION

V MEMORY LOOP (Cont)

$$
\begin{aligned}
{}_{0}V_{41} = {} & M_{R}' M_6 D_2 D_1' && 130\text{-}25 \\
& + A_1' M_3 D_3 L_{P1} L_{P2} D_1 && 130\text{-}9 \\
& + B_1' M_6 D_3' L_{P2} D_{10} && 131\text{-}9 \\
& + V_1' M_3' M_6' && 130\text{-}25 \\
& + V_1' M_3 D_1' && 131\text{-}25 \\
& + V_1' M_6 D_3 D_2' && 131\text{-}25 \\
& + V_1' D_1 L_{P1}' && 230\text{-}27 \\
& + V_1' D_1 L_{P2}' && 230\text{-}27 \\
& + V_1' D_3' D_{10}' && 231\text{-}27 \\
& + V_1' D_3' L_{P2}' && 231\text{-}27 \\
& + T_1 && 231\text{-}19
\end{aligned}
$$

V MEMORY LOOP (Cont)

K_0	$_1k_0 = L_{2a}\, L_{P2}'\, S_7'\, S_8$	163-25
	$+ L_2\, L_{P2}'\, S_8'$	263-12
	$+ V_{2a}\, L_{P2}\, S_7'\, S_8$	163-25
	$+ V_2\, L_{P2}\, S_8'$	263-12
	$_0k_0 = L_{2a}'\, L_{P2}'\, S_7'\, S_8$	563-25
	$+ L_2'\, L_{P2}'\, S_8'$	463-12
	$+ V_{2a}'\, L_{P2}\, S_7'\, S_8$	563-25
	$+ V_2'\, L_{P2}\, S_8'$	463-12

CIRCUIT BOARD
LOCATION

STATE FLIP-FLOPS

I_1	${}_1i_1 = Z_{CH}' G_0 N_{12}$	472-10
	$+ Z_{CH}' M_5 N_{12}$	472-12
	$+ N_0 N_{12}$	472-12
	$+ F_4 F_3 F_2' F_1' N_0 A_0' P_{d3} T_4$	571-18
	$+ F_5 F_4 F_3 F_2 C_R P_{d3} N_0' A_n'$	573-18
	$+ E_3 E_2 E_1' A_n A_t' P_0 T_{41}$	570-18
	$+ F_5' F_4' F_2 F_0 I_4 P_{d3}$	572-9
	$+ F_5' F_4' F_2 I_4 G_0' A_n' P_{d3}$	572-18
	${}_0i_1 = I_1 D_0 T_{41a}$	272-10
	$+ I_1 D_{10} T_{41a}$	272-27
	$+ D_{11} T_{40}$	273-34
	$+ I_1 F_0 T_{41a}$	272-27
	$+ T_d$	273-34

STATE FLIP-FLOPS (Cont)

I_2	${}_1i_2 = Z_{CH}\, G_o'\, M_5'\, N_o'\, N_{12}$	569-9
	$+ I_1\, D_0\, T_{41a}$	569-25
	$+ I_1\, I_4'\, D_{10}\, T_{41a}$	469-24
	$+ R_{c1}\, R_b'\, I_2'\, T_{41a}$	569-25
	$+ F_5'\, F_4'\, F_2\, I_4\, F_o'\, G_o\, A_n'\, P_{d3}$	569-18
	${}_0i_2 = I_2\, N_{11}$	469-12
	$+ T_d$	468-19
	$+ I_2\, H_o\, T_{41}$	469-12
I_3	${}_1i_3 = J_o'\, I_2\, N_{11}$	202-12
	$+ I_2\, I_4\, N_{11}$	202-12
	$+ F_5\, F_4\, F_3\, F_2\, N_o\, A_o'\, P_{d3}\, T_{41}$	102-18
	${}_0i_3 = N_{12}$	402-34
	$+ M_7\, D_1\, U_2\, T_{41}$	402-24
	$(+ M_2'\, H_o\, I_3\, I_4\, T_{41})$	503-9
	$+ T_d$	402-34
	$+ J_{13}$	402-19

CIRCUIT BOARD
LOCATION

STATE FLIP-FLOPS (Cont)

I_4 $_1i_4 = M_7 D_1 U_2 T_{41}$ 405-24

$+ Z_{CH} N_{12}$ 404-12

$+ J_o I_2 N_{11}$ 404-12

$+ T_d$ 405-34

$+ J_{13}$ 405-34

$_oi_4 = F_5' F_4' F_2 A_n' P_{d3}$ 102-9

$+ E_3 E_2 E_1' A_n A_t' P_o T_{41}$ 103-18

CIRCUIT BOARD
LOCATION

STATE CHANGE FLIP-FLOPS

N_{11} $_{1}n_{11} = B_{42}\, T_{40}\, I_4'$ 468-12

$+ D_0\, T_{40}$ 473-10

$+ D_{10}\, T_{40}$ 473-12

$+ S_0'\, D_6\, D_5\, D_1'\, T_{40}$ 573-25

$+ S_8'\, L_{P1}\, D_4\, D_3\, D_1\, T_{40}$ 573-9

$+ S_{11}'\, D_4\, D_3\, D_1\, T_{40}$ 573-25

$+ D_6\, D_3'\, D_2'\, T_{40}$ 473-24

$+ D_6\, D_5'\, D_1\, T_{40}$ 472-27

$+ D_6\, D_4\, T_{40}$ 472-27

$+ J_0\, T_{40}\, I_4'$ 468-12

$+ M_d'\, R_{C1}\, T_{40}$ 473-27

$+ S_{11}'\, D_6\, D_1\, T_{40}$ 473-27

$+ D_5'\, D_4\, D_3\, T_{40}\, D_1$ 574-18 473-19

$_{0}n_{11} = T_{41a}$

CIRCUIT BOARD LOCATION

STATE CHANGE FLIP-FLOPS (Cont)

N_{12} $_{1}n_{12} = M_0 D_1' T_{40}$ 275-27

$+ M_0 M_8' U_2 T_{40}$ 275-27

$+ M_1 M_9' Q_6 Q_4 T_{40}$ 175-25

$+ U_{41} M_9' T_{40}$ 474-10

$+ M_3 D_1 D_3' T_{40}$ 274-24

$+ T_X R_{C2}' H_4' T_{40} D_1'$ 175-9

$+ M_3 D_1 U_2 T_{40}$ 474-24

$+ T_X F_{C2} F_{C1} H_4 H_3 H_2 T_{40}$ 174-18

$+ F_4 B_2 B_1 T_X T_{40}$ 175-25

$+ Q_0' M_4 D_3' D_2' T_{40}$ 575-25

$+ M_4 D_2 D_1 T_{40} D_3'$ 575-25

$+ A_1 M_8' M_9' M_4 D_3 U_1 T_{40}$ 175-18

$+ A_{42} M_4 M_8' M_9' D_3 U_1' T_{40}$ 575-18

$+ (H_0 M_4 T_{40})$ 474-27

$+ M_5 U_2 T_{40}$ 274-27

$_{1}n_{12}$ (Cont Next Page)

STATE CHANGE FLIP-FLOPS (Cont)

1^n12 (Cont)	$+ M_6 D_3 Q_4 T_{40}$	474-27
	$+ T_{40} S_{sc} M_5 D_2' D_1 D_3$	575-9
	$+ M_6 D_3' D_{10} T_{40}$	274-27
	$+ M_6 D_3' D_{11} T_{40}$	574-25
	$+ I_4' I_3 D_6 P_{d3} T_{40}$	574-25
	$+ T_{40} S_{sd} M_5 D_2 D_1' D_3$	574-9
	$+ T_{40} S_{sb} M_5 D_2' D_1' D_3$	174-9
	$+ M_7 D_3' D_2' D_1' T_{40}$	174-25
	$+ M_3 D_2' D_1' T_{40}$	174-25
0^n12	$= T_{41a}$	275-19

CIRCUIT BOARD
LOCATION

SUM FLIP-FLOPS

$S_{1S} = A_1 B_1 R_1 K_a$		412-27
$+ A_1 B_1' K_a'$		412-27
$+ A_1 R_1' K_a'$		413-27
$+ A_1' B_1 R_1 K_a'$		413-27
$+ A_1' B_1' K_a$		414-27
$+ A_1' R_1' K_2$		414-27
$_{0}S = A_1' B_1' K_a'$		214-27
$+ A_1' R_1' K_a'$		214-27
$+ A_1' B_1 R_1 K_a$		212-27
$+ A_1 B_1' K_a$		212-27
$+ A_1 R_1' K_a$		213-27
$+ A_1 B_1 R_1 K_a'$		213-27

CIRCUIT BOARD
LOCATION

COMPUTATION CARRY FLIP-FLOPS

K_a

$$_1k_a = E_0 A_1 B_1 R_1 M_4' T_{41}'$$ 512-18

$$+ E_0' A_1' B_1 R_1 M_4' T_{41}'$$ 512-9

$$+ R_{43} D_1 U_{41} T_1$$ 411-27

$$+ M_4 U_1' T_1$$ 411-27

$$_0k_a = E_0 A_1' B_1' M_4' T_1'$$ 111-25

$$+ E_0 A_1' R_1' M_4'$$ 114-25

$$+ E_0' A_1 B_1' M_4'$$ 113-25

$$+ E_0' A_1 R_1' M_4'$$ 114-25

$$+ E_0 R_1 M_4 D_1 T_1'$$ 112-25

$$+ E_0' R_1' M_4 D_1 T_1'$$ 111-25

$$+ T_{41}$$ 212-19

$$+ E_0 R_1' M_4 D_1' T_1'$$ 113-25

$$+ E_0' R_1 M_4 D_1' T_1'$$ 112-25

CIRCUIT BOARD
LOCATION

ADD-SUBTRACT INDICATOR FLIP-FLOPS

E_0 $1e_0 = K'_a M_0 T_{41a}$ 155-25

$+ A'_1 K_a M_2 U'_1 T_{41a}$ 156-25

$+ M_2 T_x' U_2 T_1$ 155-25

$+ Q_0 M_9 M_4 T_{40}$ 258-24

$+ U_{41}$ 256-19

$+ R_{41} M_4 D_3 T_1$ 156-25

$+ N_5$ 255-19

$+ Q_0 M_8 M'_0 T_{40}$ 256-24

$+ M_9 M'_2 U_2 T_{41a}$ 255-24

$+ K'_a M_{10} T_{41a}$ 254-10

$+ M_8 M'_0 T_{41a}$ 252-10

0^e0 (Next Page)

CIRCUIT BOARD LOCATION

ADD-SUBTRACT INDICATOR FLIP-FLOPS (Cont)

$_0e_0 = A_{41}\, B_{41}'\, M_0\, D_2'\, U_1\, T_1$	155-18
$+ A_{41}'\, B_{41}\, M_0\, D_2'\, U_1\, T_1$	555-9
$+ A_{41}\, B_{41}\, M_0\, D_2$	556-9
$+ A_{41}'\, B_{41}'\, M_0\, D_2$	155-9
$+ M_2\, U_1$	456-18
$+ A_1\, K_a\, M_2\, T_{41a}$	455-24
$+ R_{41}'\, M_4\, D_3\, T_1$	556-25
$+ Q_0\, M_4\, D_3\, D_1\, T_{40}$	556-25
$+ X_{41}\, R_{41}'\, M_{10}\, D_2\, U_1\, T_1$	555-18
$+ X_{41}'\, R_{41}\, M_{10}\, D_2\, U_1\, T_1$	156-9
$+ X_{41}\, R_{41}\, M_{10}\, D_2'\, U_1\, T_1$	157-9
$+ X_{41}'\, R_{41}'\, M_{10}\, D_2'\, U_1\, T_1$	158-9
$+ F_a$	257-19

CIRCUIT BOARD
LOCATION

TRANSFER OF CONTROL-VERIFY INPUT FLIP-FLOPS

H_o $_1h_o = M_4\ M_8\ M_9\ D_3\ U_1\ T_1$ 557-18

$+ A_{41}'\ M_5\ D_2'\ U_1\ T_1$ 557-9

$+ A_{41}\ M_5\ U_1\ D_1'\ T_1$ 557-25

$+ (J_o\ M_5\ D_2\ D_1\ T_1)$ 557-25

$+ V_{gm}\ I_4$ 458-27

$+ M_7\ D_1\ T_2$ 455-12

$+ M_5\ D_3\ U_1$ 455-12

$+ V_{gp}\ I_4$ 458-27

$_0h_o = A_1\ M_4\ T_1'$ 257-12

$+ A_1\ M_5\ D_3'\ D_2'\ D_1'\ T_1'\ T_{41a}'$ 157-18

$+ J_0'\ T_0'\ I_4$ 257-12

$+ I_3'\ I_4'$ 257-18

$+ R_{c1}\ J_0'$ 257-34

$+ U_2\ N_4$ 257-34

$+ A_n\ A_t'$ 252-18

$+ N_{12}\ I_4'$ 259-34

CIRCUIT BOARD
LOCATION

ERROR FLIP-FLOPS

J_0 $_1j_0 = D_1 E_0 K_a M_0 U_1 T_{41}$ 518-9

$+ K_a' M_2 U_1 T_X' T_{41}$ 516-25

$+ K_a D_1 U_{41} T_{41}$ 516-25

$+ X_2 Z_2' H_0 I_2 T_{41}'$ 517-25

$+ X_2' Z_2 H_0 I_2$ 517-25

$+ R_b T_0$ 418-18

$+ M_2 E_{30} U_1 F_3'$ 417-24

$+ J_{13}$ 418-19

$+ E_{40} T_{41} S$ 424-10

$_0j_0 = T_d$ 417-19

$+ D_6 D_5' D_4 D_3' D_2 D_1 I_2 T_{40}$ 518-18

CIRCUIT BOARD
LOCATION

READER DELAY FLIP-FLOPS

D_R $_1d_R = F_5' F_4 F_3' F_2' F_0' A_0' P_m I_1$ 173-18

$+ F_5 F_3 F_2' F_1' A_0' P_m I_1$ 172-18

$+ F_5' F_4' F_3 F_2' F_1 P_m I_1$ 171-18

$+ M_2 N_{12}'$ 271-34

$+ (R_{sm} + R_{sp}) T_{PP5}$ 271-34

$+ N_{11} J_0$ 214-18

$+ E_3 E_2 E_1' P_{ma} A_t' P_{d3}$ 172-9

$_0d_R = (R_{gm} + T_g)$ 473-34

$+ V_{gm}$ 473-34

$+ N_{12}$ 472-34

$+ R_{gp}$ 471-34

$+ V_{gp}$ 471-34

$+ K_g I_4$ 472-34

$+ T_0'$ 470-19

CIRCUIT BOARD LOCATION

OUTPUT ERROR FLIP-FLOPS

J_{13} $_1j_{13} = F_1 E_{t1}' T_t$ 251-27

$+ F_1' K_{1K} T_t$ 251-27

$+ F_2 E_{t2}' T_t$ 255-27

$+ F_2' K_{2K} T_t$ 255-27

$+ F_3 E_{t3}' T_t$ 255-12

$+ F_3' K_{3K} T_t$ 254-12

$+ F_4 E_{t4}' T_t$ 254-12

$+ F_4' K_{4K} T_t$ 254-27

$+ F_5 E_{t5}' T_t$ 254-27

$+ F_5' K_{5K} T_t$ 255-12

$+ F_1 E_{P1}' T_P$ 253-12

$+ F_1' E_{P1} T_P$ 253-12

$+ F_2 E_{P2}' T_P$ 253-27

$+ F_2' E_{P2} T_P$ 253-27

$+ F_3 E_{P3}' T_P$ 252-12

$_1j_{13}$ (Cont Next Page)

CIRCUIT BOARD LOCATION

OUTPUT ERROR FLIP-FLOPS (Cont)

$1^{j}13$ (Cont)

$$+ F_3' E_{P3} T_P \qquad 252\text{-}12$$

$$+ F_4 E_{P4}' T_P \qquad 252\text{-}27$$

$$+ F_4' E_{P4} T_P \qquad 252\text{-}27$$

$$+ F_5 E_{P5}' T_P \qquad 251\text{-}12$$

$$+ F_5' E_{P5} T_P \qquad 251\text{-}12$$

$$o_{j13} = T_d \qquad 254\text{-}19$$

$$+ S_{u1} \qquad 253\text{-}19$$

CIRCUIT BOARD LOCATION

HALF-WORD INDICATOR FLIP-FLOPS

G_0 $\quad {}_1g_0 = G_0' I_4' N_{12}$ 216-27

$+ Z_1' H_0 I_4' U_2 N_9$ 114-9

$+ G_0' M_7 I_4$ 216-27

$+ G_1 F_0 N_4$ 215-10

${}_0g_0 = G_0 I_4' N_{12}$ 215-12

$+ Z_1 H_0 I_4' U_2 T_2$ 115-9

$+ G_0 M_7 I_4$ 215-12

$+ G_1' F_0 N_4$ 216-12

$+ I_4 (S_{u1} + S_{u2} + S_{u3})$ 216-12

PS ORDER FLIP-FLOPS

N_7 $\quad {}_1n_7 = A_{42} I_2 N_{11} D_6 D_3'$ 119-9

$+ A_{42}' M_4 D_3' P_6' P_5 P_3' P_1'$ 120-18

$+ I_2 N_{11} D_6 D_5$ 218-24

$+ I_2 N_{11} D_4 D_3 D_1$ 123-25

${}_0n_7 = N_7 T_{41}$ 218-10

$+ P_4 P_3 P_2 M_4$ 216-24

CIRCUIT BOARD
LOCATION

READOUT CONTROL

H_1	$_1h_1 = H_1' R_{C4}' N_8$	411-10
	$+ R_{C4}' H_1' H_2' H_3' F_0 T_0'$	510-9
	$_0h_1 = H_1 N_8$	411-12
	$+ U_1 C_R' A_0'$	411-12
H_2	$_1h_2 = H_2' H_1 N_8$	278-10
	$_0h_2 = H_2 H_1 N_8$	278-12
	$+ U_1 C_R' A_0'$	278-12
H_3	$_1h_3 = H_3' H_2 H_1 N_8$	205-24
	$_0h_3 = H_3 H_2 H_1 N_8$	105-25
	$+ U_1 C_R' A_0'$	105-25
H_4	$_1h_4 = H_4' H_3 H_2 H_1 N_8$	511-9
	$_0h_4 = H_4 H_3 H_2 H_1 N_8$	511-25
	$+ U_1 C_R' A_0'$	511-25
	$+ N_{12}$	406-19

READOUT CONTROL (Cont)

N_8	$_1n_8 = M_2' D_5' T_X T_{41} N_{12}'$	509-9
	$+ (TPP)' R_{C3} T_{40}$	410-10
	$_0n_8 = N_8$	411-18

READOUT TIMING

R_{C1}	$_1r_{C1} = J_0' D_6' D_5 D_4 D_2 D_1' I_2 N_{11}$	516-18
	$+ R_b I_4$	415-18
	$_0r_{C1} = R_{C2}$	415-34
	$+ T_d$	415-34
R_{C2}	$_1r_{C2} = R_{C1} I_3$	214-12
	$+ M_7 D_6' U_1$	214-12
	$_0r_{C2} = H_4 N_8$	215-34
	$+ I_3'$	215-34
	$+ A_{42}' H_3 H_2 H_1 N_8$	113-9

READOUT TIMING (Cont)

R_{C3}	$_1r_{C3} = M_3\, T_X\, T_{41}\, N_{12}'$	103-25
	$+(TPP)\, P_m\, T_0'\, I_4\, F_5$	103-25
	$_0r_{C3} = (TPP)'\, R_{C3}\, T_{40}$	205-10
R_{C4}	$_1r_{C4} = R_{C1}$	219-19
	$+ T_0'\, C_r'\, A_0'\, I_4\, P_{d3}\, F_5$	118-18
	$_0r_{C4} = R_{C1}'\, T_{40}$	221-18
T_X	$_1t_X = M_3\, D_2\, D_1'\, Q_1\, T_{41a}$	553-9
	$+ M_7\, D_6'\, U_1\, T_{41a}$	453-27
	$+ D_5'\, P_{d3}\, T_{41a}$	453-10
	$+ D_6'\, D_5\, D_4'\, D_3\, I_2\, N_{11}\, J_0'$	552-18
	$+ I_5\, I_3\, T_{41a}$	453-27
	$_0t_X = M_2'\, T_X\, T_{41a}$	453-12
	$+ N_{12}$	453-12

CIRCUIT BOARD
LOCATION

READOUT TIMING (Cont)

N_{8A} $_1n_{8a} = M_2' D_5' T_X T_{41} N_{12}$ 176-9

$_0n_{8a} = N_{8a}$ 276-19

S_X $_1s_X = H_3 H_2 H_1' N_8 F_{c2} F_{c1}'$ 111-9

$+ I_2$ 211-19

$_0s_X = H_2' H_1' N_8$ 209-12

$+ M_2$ 209-12

TAPE PUNCH PULSE FLIP-FLOPS

TPP $_1t_{PP} = (TPP_P)\ X_{10}'$ 457-12

$+ T_0 Q_0 P_5' P_4' P_3' P_2' P_1$ 559-18

$+ (S_u + TPP)\ I_4 P_5' P_4' P_3' P_2' P_1$ 556-18

$+ T_{PPt} X_9' I_4'$ 457-12

$_0t_{PP} = (TPP_{rm})'(TPP_K)'(TPP_t)'(TPP_P)' \quad [S_{u1}' S_{u2}' S_{u3}' S_c']\ (TPP_{rP})'$ 156-18

CIRCUIT BOARD LOCATION

DELAY FLIP-FLOPS

P_m	$_1P_m = (TPP)\ J_{13}'\ A_n'\ P_5'\ P_4'\ P_3'\ P_2'\ P_1$	564-18
	$_oP_m = P_{d3}$	466-19
	$+ J_{13}$	466-34
	$+ C_t$	466-34
P_{d3}	$_1P_{d3} = (TPP)'\ T_o'\ P_m\ T_{40}$	418-24
	$+ C_t$	419-19
	$+ (TPP)'\ P_{ma}\ T_{40}$	415-24
	$_oP_{d3} = P_{d3}\ T_{40}$	219-18
X_{20}	$_1x_{20} = X_{20}'\ X_o$	256-18
	$_ox_{20} = X_{20}\ X_o$	255-18
A_n	$_1a_n = M_7\ D_6\ D_3'\ D_1\ L_{P1}$	577-9
	$+ F_5'\ F_4\ F_3\ F_2'\ F_1\ C_t$	578-9
	$+ A_t\ I_5$	474-12
	$_oa_n = M_7'\ T_o'$	275-18
	$+ E_3'\ E_2'\ E_1'\ A_t\ P_o\ T_{41}$	177-18
	$+ T_d$	274-19

DELAY FLIP-FLOPS (Cont)

A_t $_1a_t = (A_n\ P_m)$ 455-18
$+ F_5'\ F_4\ F_3\ F_2'\ F_1\ C_T$ 552-9
$_0a_t = A_n'\ P_{d3}\ T_{40}$ 455-10
$+ T_d$ 457-19

P_{ma} $_1P_{ma} = (TPP)\ A_n\ P_5'\ P_4'\ P_3'\ P_2'\ P_1$ 577-18

$_0P_{ma} = P_{d3}$ 474-34
$+ F_5'\ F_4'\ F_3'\ F_2'\ F_1'\ A_t'\ Q_0$ 563-18

INPUT BUFFER FILL FLIP-FLOPS

I_5 $_1i_5 = F_0'\ T_1\ I_1\ P_{d3}\ F_4\ F_3\ F_1'\ A_N'$ 568-18
$+ M_3'\ I_4\ N_{12}$ 472-24
$+ M_R'\ L_{P1}'\ M_8\ D_6\ T_2$ 572-25
$+ K_n'\ L_{P1}\ M_8\ D_6\ T_2$ 572-25
$+ M_8\ D_6\ I_5'\ T_{41a}$ 571-25

$_0i_5 = I_5\ T_{41a}$ 271-10

CIRCUIT BOARD
LOCATION

FILL DECIMAL NUMBER FLIP-FLOPS

N_o $_1n_o = F_5' F_4 F_3 F_2' F_1' C_t$ 516-9

$+ N_b C_r' M_3' I_4$ 416-24

$_on_o = L_b$ 416-34

$+ F_5 F_4' F_3' F_2 F_1' T_o P_m$ 515-18

$+ C_b$ 416-34

$+ F_5' F_4 F_3 F_2 F_1' P_m$ 515-9

$+ I_4'$ 417-34

$+ M_3$ 417-34

$+ F_5' F_4 F_3 F_2' F_1 P_m$ 519-9

CIRCUIT BOARD
LOCATION

FILL ORDERS FLIP-FLOPS

O_0	$_0O_0 = F_5' F_4 F_3 F_2 F_1' C_t$	560-9
	$+ C_b C_n' M_3' I_4$	559-9
	$+ A_t A_n'$	464-18
	$_{00}O_0 = L_b$	461-34
	$+ F_5 F_4' F_3' F_2 P_m T_0 F_1'$	561-18
	$+ N_L$	461-34
	$+ F_5' F_4 F_3 F_2' P_m$	560-18
	$+ I_4'$	460-34
	$+ M_3$	460-34
	$+ A_t I_5$	459-34
O_1	$_1O_1 = N_0' P_0 E_3 O_1' F_4'$	574-25
	$+ O_1' N_0' P_0 E_2 F_4'$	574-25
	$+ A_n P_0 O_1'$	475-10
	$_0O_1 = O_1 T_{41C}$	276-24

CIRCUIT BOARD
LOCATION

FILL ORDER COUNTER FLIP-FLOP

F_0	$_1f_o = F_5\ F_4'\ F_3'\ F_2\ F_1'\ C_t$	152-18
	$+ M_7\ I_4\ A_n'$	254-24
	$+ L_b\ C_R'\ M_3'\ I_4\ T_1$	152-9
	$+ E_0'\ M_8\ N_{12}$	253-10
	$+ M_5\ N_{12}\ Z_{CH}$	251-10
	$_0f_o = F_5'\ F_4\ F_3\ P_m$	452-27
	$+ N_b$	452-34
	$+ I_4'\ N_{12}'$	453-34
	$+ M_3\ N_{12}'$	453-34
	$+ (S_{u1} + S_{u2} + S_{u3})$	453-19
	$+ C_b$	452-34

CIRCUIT BOARD
LOCATION

LEGITIMATE CODE FLIP-FLOP

P_o	$_1P_o = F_o\, F_5\, F_4'\, P_{d3}$	270-24
	$+ O_o\, F_5\, F_4'\, P_{d3}$	169-25
	$+ O_o\, F_5\, F_3'\, F_2\, P_{d3}$	170-25
	$+ N_o\, F_5\, F_4'\, P_{d3}\, T_{41a}$	169-25
	$+ N_o\, F_5\, F_3'\, P_{d3}$	269-24
	$+ P_{d3}\, F_1'\, F_2'\, F_5\, N_o$	170-25
	$+ A_n\, P_{d3}\, P_{ma}$	271-12
	$_0P_o = N_o'\, P_o\, T_{41a}$	470-12
	$+ P_o\, E_2'\, T_{41a}$	470-12

INPUT-OUTPUT REGISTER

F_1 $_1f_1 = K_1\ I_4\ \underline{(TPP)P_5}$ 408-12

$+ F_5\ F_3'\ F_2'\ C_t$ 507-25

$+ A_1\ N_0'\ O_1'\ P_0\ T_1'$ 508-18

$+ F_2\ O_1$ 408-34

$+ F_2\ E_{30}$ 407-27

$+ B_1\ S_x\ T_x\ N_{12}'$ 507-25

$+ F_2\ S_x'\ T_x\ N_{12}'$ 408-27

$+ F_2\ N_6$ 408-27

$+ D_5'\ S_x\ N_8\ H_1'$ 407-24

$+ F_4'\ F_2\ N_8\ D_5'$ 407-27

$+ S_{u1}\ I_4$ 408-10

$+ S_{u3}\ I_4$ 407-18

0^{f_1} (Next Page)

INPUT-OUTPUT REGISTER (Cont)

$_{0}f_1 = F_3' F_2 C_t$	207-27
$+ F_4' F_3 F_2' C_t$	207-27
$+ A_1' N_0' O_1' P_0 T_1'$	107-9
$+ F_2' O_1$	208-12
$+ F_2' E_{30}$	208-12
$+ B_1' S_x T_x$	208-10
$+ F_2' S_x' T_x$	209-27
$+ F_2' N_6$	207-34
$+ Z_{CH} I_3'$	207-34
$+ M_3 N_8$	208-34
$+ D_5 P_0' P_{d3} T_{40}$	208-27
$+ N_{12}$	207-19
$+ D_5' P_{d3} T_{41}$	208-27
$+ F_3 F_2' N_{8a}$	209-27
$+ F_4 N_{8a}$	208-34

INPUT-OUTPUT REGISTER (Cont)

F_2 $\quad {}_1f_2 = K_2\ I_4\ \underline{(TPP)P_5}$ 403-12

$+ F_5'\ F_2'\ C_t$ 403-27

$+ F_4'\ F_2'\ F_1\ C_t$ 403-27

$+ F_3\ O_1$ 404-34

$+ F_3\ E_{30}\ M_2'$ 404-27

$+ F_3\ E_{30}'\ T_x\ N_{12}'$ 404-27

$+ F_3\ N_6$ 404-34

$+ S_c\ I_4$ 403-12

$+ F_4'\ F_3'\ F_2'\ N_8\ D_5'\ H_1$ 502-18

$+ F_3'\ F_1'\ N_8\ D_5'\ F_2'$ 504-25

$+ F_4'\ F_2'\ F_1'\ N_8\ D_5'$ 506-18

$+ S_x'\ F_4'\ F_3'\ F_2'\ N_8\ D_5'$ 504-9

$+ I_4\ (S_{u1} + S_{u2} + S_{u3})$ 504-25

$+ N_{8a}\ A_{42}\ S_x\ F_1$ 404-24

0^f2 (Next Page)

CIRCUIT BOARD LOCATION

INPUT-OUTPUT REGISTER (Cont)

$of_2 = F_2\ F_1'\ C_t$	203-27
$+ F_4\ F_1\ C_t$	204-27
$+ F_5\ F_3\ F_2\ C_t$	204-27
$+ F_3'\ O_1$	204-34
$+ F_3'\ E_{30}$	204-34
$+ F_3'\ T_X$	204-12
$+ F_3'\ N_6$	204-12
$+ N_{12}$	203-19
$+ Z_{CH}\ I_3'$	205-12
$+ M_3\ N_8$	206-10
$+ D_5\ P_0'\ P_{d3}\ T_{40}$	205-27
$+ D_5'\ P_{d3}\ T_{41}$	205-27
$+ F_3'\ F_2\ F_1\ N_{8a}$	203-27
$+ F_3\ F_2\ F_1'\ N_{8a}$	206-27
$+ M_2\ E_{30}$	205-12
$+ N_{8a}\ F_2\ F_4$	206-27

INPUT-OUTPUT REGISTER (Cont)

F_3	${}_1f_3 = K_3\ I_4\ \underline{(TPP)\ P_5}$	210-12
	$+ S_c\ I_4$	210-12
	$+ F_5'\ F_4\ C_t$	211-27
	$+ F_4'\ F_5\ F_2'\ F_1'\ C_t$	108-25
	$+ A_1\ A_n'\ O_1$	211-12
	$+ F_4\ E_{30}$	206-18
	$+ B_2\ M_2\ T_x\ N_{12}'$	108-25
	$+ B_1\ A_{42}'\ S_x'\ T_x\ M_2'\ N_{12}'$	108-18
	$+ F_4\ A_{42}\ T_x\ N_{12}'$	210-27
	$+ F_4'\ F_3'\ F_2'\ N_8\ D_5'\ H_1$	108-9
	$+ F_4\ N_6$	210-27
	$+ F_4'\ F_3'\ F_2'\ N_8\ S_x'\ D_5'$	107-18
	$+ F_4\ F_1'\ N_8\ D_5'$	211-27
	$+ S_{u1}\ I_4$	209-10
	$+ F_4\ A_n\ O_1$	211-12

${}_0f_3$ (Next Page)

CIRCUIT BOARD LOCATION

INPUT-OUTPUT REGISTER (Cont)

$$_{0}f_3 = F_5\ F_2\ C_t$$ 409-27

$$+ F_5'\ F_4'\ F_1'\ C_t$$ 409-27

$$+ A_1'\ A_n'\ O_1$$ 409-12

$$+ F_4'\ E_{30}\ N_0$$ 409-10

$$+ B_1'\ A_{42}'\ T_X\ M_2'$$ 508-25

$$+ N_{12}$$ 410-19

$$+ F_4'\ A_{42}\ M_2'\ T_X$$ 508-25

$$+ F_4'\ N_6$$ 409-12

$$+ Z_{CH}\ I_3'$$ 410-34

$$+ M_3\ N_8$$ 410-34

$$+ D_5\ P_0'\ P_{d3}\ T_{40}$$ 410-27

$$+ D_5'\ P_{d3}\ T_{41}$$ 410-27

$$+ F_4'\ F_3\ F_2'\ N_{8a}$$ 509-25

$$+ B_2'\ M_2\ T_X$$ 509-25

$$+ F_4'\ A_n\ O_1$$ 409-10

CIRCUIT BOARD
LOCATION

INPUT-OUTPUT REGISTER (Cont)

F_4	$_1f_4 = K_4\ I_4\ \underline{(TPP)\ P_5}$	207-12
	$+ F_5\ F_3'\ F_2'\ F_1\ C_t$	106-25
	$+ F_5'\ F_4'\ F_1'\ C_t$	107-25
	$+ F_5'\ F_4'\ F_3'\ F_2\ C_t$	106-25
	$+ B_1\ A_{42}\ M_2'\ N_{12}'\ S_X'\ T_X$	106-9
	$+ F_5\ N_6$	207-12
	$+ F_3\ F_2'\ F_1'\ N_8\ D_5'$	107-25
	$+ F_5\ O_1\ A_N$	207-10
	$+ N_{8a}\ A_{42}\ S_X\ F_1$	206-24

$0^{f}4$ (Next Page)

INPUT-OUTPUT REGISTER (Cont)

$$of_4 = F_4\ F_2\ C_t \qquad 407\text{-}12$$

$$+ E_{30} \qquad 407\text{-}19$$

$$+ B_1'\ A_{42}\ T_x \qquad 406\text{-}12$$

$$+ F_5'\ N_6 \qquad 407\text{-}34$$

$$+ Z_{CH}\ I_3' \qquad 407\text{-}34$$

$$+ M_3\ N_8 \qquad 407\text{-}12$$

$$+ D_5\ P_o'\ P_{d3}\ T_{40} \qquad 406\text{-}27$$

$$+ D_5'\ P_{d3}\ T_{41} \qquad 406\text{-}27$$

$$+ N_{12} \qquad 408\text{-}19$$

$$+ F_3'\ F_1'\ N_{8a} \qquad 406\text{-}12$$

$$+ F_5'\ O_1 \qquad 406\text{-}18$$

$$+ A_o\ P_{d3}\ F_3\ T_o \qquad 408\text{-}24$$

CIRCUIT BOARD
LOCATION

INPUT-OUTPUT REGISTER (Cont)

F_5 $\quad {}_1f_5 = K_5\ I_4\ \underline{(TPP)\ P_5}$ 423-12

$+ F_4'\ F_3'\ F_1\ C_t$ 423-27

$+ F_4'\ F_3\ F_2\ C_t$ 422-27

$+ F_5'\ F_4\ F_3'\ F_1'\ C_t$ 523-9

$+ Z_1\ N_6$ 423-34

$+ F_4'\ F_3'\ F_2'\ N_8\ H_1$ 523-25

$+ F_2\ F_1'\ N_8\ F_4'$ 424-24

$+ F_2'\ F_1\ N_8$ 423-27

$+ F_4\ F_3\ N_8$ 422-27

$+ F_4'\ F_3'\ F_2'\ N_8\ S_x'$ 523-25

$+ A_1\ O_1$ 423-34

0^{f5} (Next Page)

CIRCUIT BOARD
LOCATION

INPUT-OUTPUT REGISTER (Cont)

$$
\begin{aligned}
{}_{o}f_5 = {} & F_4\, F_3'\, F_1\, C_t && 521\text{-}9 \\
& + F_4\, F_3\, F_2\, C_t && 522\text{-}9 \\
& + F_4'\, F_3\, F_2'\, F_1'\, C_t && 522\text{-}25 \\
& + F_5\, F_3'\, F_2\, F_1'\, C_t && 522\text{-}25 \\
& + Z_1'\, N_6 && 422\text{-}34 \\
& + N_{12} && 423\text{-}19 \\
& + P_{d3}\, T_{40} && 421\text{-}27 \\
& + Z_{CH}\, I_3' && 422\text{-}34 \\
& + D_5'\, P_{d3}\, T_{41} && 421\text{-}27 \\
& + A_1'\, O_1\, A_N && 522\text{-}18
\end{aligned}
$$

CIRCUIT BOARD
LOCATION

FORMAT CONTROL FLIP-FLOP

F_{C1} $\quad {}_1f_{c1} = O_f$ 403-19

$${}_0f_{c1} = A_{42}\, T_x\, D_1' + C_f + N_0\, R_{c4}' + I_4'\, R_{c2}\, U_1$$

404-10
404-19
405-10
402-10

F_{C2} $\quad {}_1f_{c2} = A_{42}'\, T_x\, D_1' + O_f\, I_4 + C_f\, I_4$

456-10
456-34
456-34

$${}_0f_{c2} = A_{42}\, T_x\, D_1' + D_f\, I_a$$

456-12
456-12

CIRCUIT BOARD
LOCATION

INPUT ZEROING-KEYBOARD STOP FLIP-FLOP

Z_{CH} $_1Z_{CH} = N_0' P_0 T_{41}$ 211-10

$+ P_0 E_2' T_{41}$ 212-12

$+ M_3 I_4$ 211-34

$+ T_d I_4$ 211-34

$+ (SO + H_c) I_4' T_{41}$ 212-12

$+ E_3 Y_8 Y_1 Y_0 I_3 U_1 T_2 Y_2$ 111-18

$+ M_5 S_{sa}$ 214-18

$_0Z_{CH} = J_{13}$ 209-19

$+ Z_{CH} I_3' T_{41}'$ 210-10

CIRCUIT BOARD LOCATION

INPUT SEQUENCE COUNTER

E_1	${}_1e_1 = E_1' P_0 T_{41a}$	469-10
	$+ C_R'$	469-19
	$+ A_0 P_{d3} F_4 F_3$	468-24
	${}_0e_1 = E_1 P_0 T_{41a}$	269-10
	$+ N_0 E_3 E_1$	269-12
	$+ F_0 E_3' T_{40}$	269-12
	$+ Z_2 N_9 A_n U_2$	266-24
E_2	${}_1e_2 = E_2' E_1 P_0 T_{41}$	415-27
	$+ N_0 E_3 E_1$	415-27
	$+ S_P Y_{12} Y_{11} Y_{10} Y_9 A_n' I_4' T_{41}$	517-18
	$+ Z_2' A_n U_2 N_1 P_4' P_3'$	514-9
	${}_0e_2 = E_2 E_1 P_0 T_{41}$	116-25
	$+ C_R' I_2'$	216-18
	$+ P_{d3} N_0 F_3 F_4 F_5$	116-25

CIRCUIT BOARD
LOCATION

INPUT SEQUENCE COUNTER (Cont)

E_3 $_1e_3 = E_3' E_2 E_1 P_0 T_{41}$ 119-25

$+ F_0 E_3' T_{40} E_1 R_{C1}'$ 119-25

$+ E_2 Y_7 Y_6 Y_5 Y_4 Y_3 T_1 I_4'$ 121-18

$+ J_6 J_5 J_4 I_2 R_{C1} T_1$ 122-18

$+ Z_2' A_n U_2 N_1 P_5' P_3$ 114-18

—

$_0e_3 = E_3 E_2 E_1 P_0 T_{41}$ 523-18

$+ N_0 E_3 E_1$ 422-10

$+ I_2 R_{C1} T_{41}$ 422-12

$+ C_R' I_2' T_1'$ 422-12

CIRCUIT BOARD LOCATION

"NUMBER FILL" STATE FLIP-FLOPS

E_{20}	$_1e_{20} = N_0 F_5 F_4' E_2 P_{d3}$	567-25
	$+ N_0 F_5 F_3' E_2 P_{d3}$	567-25
	$+ M_2 T_X U_{41}'$	466-10
	$_0e_{20} = E_{20} T_{41a} I_4$	466-12
	$+ N_{12}$	466-12
E_{30}	$_1e_{30} = E_{20} T_{41a} N_0$	259-27
	$+ M_2 E_{20} U_{41}' T_{40}$	259-27
	$_0e_{30} = I_4 E_{30} T_{41a}$	259-12
	$+ M_2 T_1$	259-12
	$+ Q_6 Q_4 Q_1 T_{41a}$	259-24
E_{40}	$_1e_{40} = E_{30} N_0 T_{41a}$	256-10
	$_0e_{40} = E_{40} T_{41a}$	255-10

CIRCUIT BOARD LOCATION

INPUT COUNTER RESET FLIP-FLOPS

C_r $\quad {}_1c_r = F_o\ T_2'$ 454-18

$+ O_o\ T_2'$ 454-34

$+ N_o\ T_2'$ 454-34

$+ A_n\ U_2$ 454-12

$+ A_n\ A_t$ 454-12

${}_0c_r = A_N'\ F_4\ F_3\ P_{d3}\ T_2$ 555-25

$+ F_o'\ O_o'\ N_o'\ A_n'\ I_3'$ 555-25

$+ R_b$ 454-19

$+ N_{12}$ 455-19

INHIBIT FILL FLIP-FLOP

A_o $\quad {}_1a_o = N_o\ F_5\ F_4\ F_3\ F_2'\ F_1'\ P_{d3}\ T_1$ 161-18

$+ F_5\ F_4\ F_3'\ F_2\ N_o\ P_{d3}\ T_1$ 160-18

${}_0a_o = F_5\ P_{d3}\ T_{41a}$ 260-10

$+ I_4'$ 260-19

$+ R_b$ 261-19

$+ T_1$ 258-19

CIRCUIT BOARD LOCATION

TAPE READER "ON" FLIP-FLOPS

T_o	$_1t_o = (R_{gm} + T_g)\, I_4$	266-34
	$+ V_{gm}\, I_4$	266-34
	$+ M_7\, D_3'\, I_4$	265-10
	$+ R_{gp}\, I_4$	265-34
	$+ V_{gp}\, I_4$	265-34
	$_0t_o = K_f$	464-34
	$+ F_5\, F_4'\, F_3\, F_2'\, F_1'\, C_t$	565-9
	$+ I_4'$	464-19
	$+ T_d$	464-34
	$+ (R_{sm} + R_{sp})$	467-18

PHOTOREADER CLUTCH-BRAKE FLIP-FLOPS

B_c	$_1b_c = T_o\, Q_o\, D_r'$	420-10
	$_0b_c = T_o'$	221-19
	$+ Q_o'$	221-34
	$+ D_r$	221-34

CIRCUIT BOARD LOCATION

CODE CONVERSION TIME FLIP-FLOP

C_t	${}_1c_t = (TPP)'\ T_0\ P_m\ P_6\ P_3\ P_5\ P_2'\ P_1'$	113-18
	${}_0c_t = C_t$	213-19

MAIN MEMORY WRITE FLIP-FLOPS

M_{W1}	${}_1m_{w1} = A_2\ M_3\ T_{41}'$	226-10
	$+ B_2\ M_6\ D_3'\ T_{41}'$	227-24
	$+ L_2\ M_6\ D_2'\ D_1\ T_{41}'$	126-9
	$+ V_2\ M_6\ D_2\ D_1\ T_{41}'$	127-9
	$+ X_2\ I_1\ T_{41}'$	227-10
	$+ D_{11}'$	227-19
	${}_0m_{w1} = A_2'\ M_3\ D_{11}$	224-10
	$+ B_2'\ M_6\ D_3'\ D_{11}$	226-24
	$+ L_2'\ M_6\ D_2'\ D_1\ D_{11}$	124-9
	$+ V_2'\ M_6\ D_2\ D_1\ D_{11}$	125-9
	$+ X_2'\ I_1\ D_{11}$	225-10
	$+ D_{11}\ T_{41}$	227-18

MAIN MEMORY WRITE FLIP-FLOPS (Cont)

M_{W2} $_{1}m_{w2} = A_2' M_3$ 230-34

$+ B_2' M_6 D_3'$ 230-10

$+ L_2' M_6 D_2' D_1$ 228-24

$+ V_2' M_6 D_2 D_1$ 229-24

$+ X_2' I_1$ 230-34

$+ T_{41}$ 230-19

$_{0}m_{w2} = A_2 M_3 D_{11} T_{41}'$ 230-24

$+ B_2 M_6 D_3' D_{11} T_{41}'$ 130-18

$+ L_2 M_6 D_2' D_1 D_{11} T_{41}'$ 128-18

$+ V_2 M_6 D_2 D_1 D_{11} T_{41}'$ 129-18

$+ X_2 I_1 D_{11} T_{41}'$ 231-24

MEMORY WRITE AMPLIFIERS

$$ {}_1W_0 = (C_6' C_5' C_4' C_3' C_2' C_1') M_{W2}' $$

$$ {}_0W_0 = (\quad " \quad) M_{W1}' $$

$$ {}_1W_1 = (C_6' C_5' C_4' C_3' C_2' C_1) M_{W2}' $$

$$ {}_0W_1 = (\quad " \quad) M_{W1}' $$

$$ {}_1W_2 = (C_6' C_5' C_4' C_3' C_2 C_1') M_{W2}' $$

$$ {}_0W_2 = (\quad " \quad) M_{W1}' $$

$$ {}_1W_3 = (C_6' C_5' C_4' C_3' C_2 C_1) M_{W2}' $$

$$ {}_0W_3 = (\quad " \quad) M_{W1}' $$

$$ {}_1W_4 = (C_6' C_5' C_4' C_3 C_2' C_1') M_{W2}' $$

$$ {}_0W_4 = (\quad " \quad) M_{W1}' $$

$$ {}_1W_5 = (C_6' C_5' C_4' C_3 C_2' C_1) M_{W2}' $$

$$ {}_0W_5 = (\quad " \quad) M_{W1}' $$

MEMORY WRITE AMPLIFIERS (Cont)

$$_{i}W_{6} = (C_6' C_5' C_4' C_3 C_2 C_1') M_{W2}'$$

$$_{o}W_{6} = (\quad " \quad) M_{W1}'$$

$$_{i}W_{7} = (C_6' C_5' C_4' C_3 C_2 C_1) M_{W2}'$$

$$_{o}W_{7} = (\quad " \quad) M_{W1}'$$

$$_{i}W_{8} = (C_6' C_5' C_4 C_3' C_2' C_1') M_{W2}'$$

$$_{o}W_{8} = (\quad " \quad) M_{W1}'$$

$$_{i}W_{9} = (C_6' C_5' C_4 C_3' C_2' C_1) M_{W2}'$$

$$_{o}W_{9} = (\quad " \quad) M_{W1}'$$

$$_{i}W_{10} = (C_6' C_5' C_4 C_3' C_2 C_1') M_{W2}'$$

$$_{o}W_{10} = (\quad " \quad) M_{W1}'$$

$$_{i}W_{11} = (C_6' C_5' C_4 C_3' C_2 C_1) M_{W2}'$$

$$_{o}W_{11} = (\quad " \quad) M_{W1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{12} = (C_6' C_5' C_4 C_3 C_2' C_1') M_{w2}'$$

$$_0W_{12} = (\quad " \quad) M_{w1}'$$

$$_1W_{13} = (C_6' C_5' C_4 C_3 C_2' C_1) M_{w2}'$$

$$_0W_{13} = (\quad " \quad) M_{w1}'$$

$$_1W_{14} = (C_6' C_5' C_4 C_3 C_2 C_1') M_{w2}'$$

$$_0W_{14} = (\quad " \quad) M_{w1}'$$

$$_1W_{15} = (C_6' C_5' C_4 C_3 C_2 C_1) M_{w2}'$$

$$_0W_{15} = (\quad " \quad) M_{w1}'$$

$$_1W_{16} = (C_6' C_5 C_4' C_3' C_2' C_1') M_{w2}'$$

$$_0W_{16} = (\quad " \quad) M_{w1}'$$

$$_1W_{17} = (C_6' C_5 C_4' C_3' C_2' C_1) M_{w2}'$$

$$_0W_{17} = (\quad " \quad) M_{w1}'$$

$$_1W_{18} = (C_6' C_5 C_4' C_3' C_2 C_1') M_{w2}'$$
$$_0W_{18} = (\quad " \quad) M_{w1}'$$

$$_1W_{19} = (C_6' C_5 C_4' C_3' C_2 C_1) M_{w2}'$$
$$_0W_{19} = (\quad " \quad) M_{w1}'$$

$$_1W_{20} = (C_6' C_5 C_4' C_3 C_2' C_1') M_{w2}'$$
$$_0W_{20} = (\quad " \quad) M_{w1}'$$

$$_1W_{21} = (C_6' C_5 C_4' C_3 C_2' C_1) M_{w2}'$$
$$_0W_{21} = (\quad " \quad) M_{w1}'$$

$$_1W_{22} = (C_6' C_5 C_4' C_3 C_2 C_1') M_{w2}'$$
$$_0W_{22} = (\quad " \quad) M_{w1}'$$

$$_1W_{23} = (C_6' C_5 C_4' C_3 C_2 C_1) M_{w2}'$$
$$_0W_{23} = (\quad " \quad) M_{w1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{24} = (C_6' C_5 C_4 C_3' C_2' C_1') M_{w2}'$$

$$_0W_{24} = (\quad " \quad) M_{w1}'$$

$$_1W_{25} = (C_6' C_5 C_4 C_3' C_2' C_1) M_{w2}'$$

$$_0W_{25} = (\quad " \quad) M_{w1}'$$

$$_1W_{26} = (C_6' C_5 C_4 C_3' C_2 C_1') M_{w2}'$$

$$_0W_{26} = (\quad " \quad) M_{w1}'$$

$$_1W_{27} = (C_6' C_5 C_4 C_3' C_2 C_1) M_{w2}'$$

$$_0W_{27} = (\quad " \quad) M_{w1}'$$

$$_1W_{28} = (C_6' C_5 C_4 C_3 C_2' C_1') M_{w2}'$$

$$_0W_{28} = (\quad " \quad) M_{w1}'$$

$$_1W_{29} = (C_6' C_5 C_4 C_3 C_2' C_1) M_{w2}$$

$$_0W_{29} = (\quad " \quad) M_{w1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{30} = (C_6' C_5 C_4 C_3 C_2 C_1') M_{W2}'$$
$$_0W_{30} = (\quad " \quad) M_{W1}'$$

$$_1W_{31} = (C_6' C_5 C_4 C_3 C_2 C_1) M_{W2}'$$
$$_0W_{31} = (\quad " \quad) M_{W1}'$$

$$_1W_{32} = (C_6 C_5' C_4' C_3' C_2' C_1') M_{W2}'$$
$$_0W_{32} = (\quad " \quad) M_{W1}'$$

$$_1W_{33} = (C_6 C_5' C_4' C_3' C_2' C_1) M_{W2}'$$
$$_0W_{33} = (\quad " \quad) M_{W1}'$$

$$_1W_{34} = (C_6 C_5' C_4' C_3' C_2 C_1') M_{W2}'$$
$$_0W_{34} = (\quad " \quad) M_{W1}'$$

$$_1W_{35} = (C_6 C_5' C_4' C_3' C_2 C_1) M_{W2}'$$
$$_0W_{35} = (\quad " \quad) M_{W1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{36} = (C_6\, C_5'\, C_4'\, C_3\, C_2'\, C_1')\, M_{W2}'$$

$$_0W_{36} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{37} = (C_6\, C_5'\, C_4'\, C_3\, C_2'\, C_1)\, M_{W2}'$$

$$_0W_{37} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{38} = (C_6\, C_5'\, C_4'\, C_3\, C_2\, C_1')\, M_{W2}$$

$$_0W_{38} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{39} = (C_6\, C_5'\, C_4'\, C_3\, C_2\, C_1)\, M_{W2}'$$

$$_0W_{39} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{40} = (C_6\, C_5'\, C_4\, C_3'\, C_2'\, C_1')\, M_{W2}'$$

$$_0W_{40} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{41} = (C_6\, C_5'\, C_4\, C_3'\, C_2'\, C_1)\, M_{W2}'$$

$$_0W_{41} = (\quad\text{"}\quad)\, M_{W1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{42} = (C_6 C_5' C_4 C_3' C_2 C_1') M_{W2}'$$

$$_0W_{42} = (\quad " \quad) M_{W1}'$$

$$_1W_{43} = (C_6 C_5' C_4 C_3' C_2 C_1) M_{W2}'$$

$$_0W_{43} = (\quad " \quad) M_{W1}'$$

$$_1W_{44} = (C_6 C_5' C_4 C_3 C_2' C_1') M_{W2}'$$

$$_0W_{44} = (\quad " \quad) M_{W1}'$$

$$_1W_{45} = (C_6 C_5' C_4 C_3 C_2' C_1) M_{W2}'$$

$$_0W_{45} = (\quad " \quad) M_{W1}'$$

$$_1W_{46} = (C_6 C_5' C_4 C_3 C_2 C_1') M_{W2}'$$

$$_0W_{46} = (\quad " \quad) M_{W1}'$$

$$_1W_{47} = (C_6 C_5' C_4 C_3 C_2 C_1) M_{W2}'$$

$$_0W_{47} = (\quad " \quad) M_{W1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{48} = (C_6 C_5 C_4' C_3' C_2' C_1') M_{W2}'$$

$$_0W_{48} = (\quad " \quad) M_{W1}'$$

$$_1W_{49} = (C_6 C_5 C_4' C_3' C_2' C_1) M_{W2}'$$

$$_0W_{49} = (\quad " \quad) M_{W1}'$$

$$_1W_{50} = (C_6 C_5 C_4' C_3' C_2 C_1') M_{W2}'$$

$$_0W_{50} = (\quad " \quad) M_{W1}'$$

$$_1W_{51} = (C_6 C_5 C_4' C_3' C_2 C_1) M_{W2}'$$

$$_0W_{51} = (\quad " \quad) M_{W1}'$$

$$_1W_{52} = (C_6 C_5 C_4' C_3 C_2' C_1') M_{W2}'$$

$$_0W_{52} = (\quad " \quad) M_{W1}'$$

$$_1W_{53} = (C_6 C_5 C_4' C_3 C_2' C_1) M_{W2}'$$

$$_0W_{53} = (\quad " \quad) M_{W1}'$$

MEMORY WRITE AMPLIFIERS (Cont)

$$_1W_{54} = (C_6\, C_5\, C_4'\, C_3\, C_2\, C_1')\, M_{W2}'$$

$$_0W_{54} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{55} = (C_6\, C_5\, C_4'\, C_3\, C_2\, C_1)\, M_{W2}$$

$$_0W_{55} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{56} = (C_6\, C_5\, C_4\, C_3'\, C_2'\, C_1')\, M_{W2}'$$

$$_0W_{56} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{57} = (C_6\, C_5\, C_4\, C_3'\, C_2'\, C_1)\, M_{W2}'$$

$$_0W_{57} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{58} = (C_6\, C_5\, C_4\, C_3'\, C_2\, C_1')\, M_{W2}'$$

$$_0W_{58} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{59} = (C_6\, C_5\, C_4\, C_3'\, C_2\, C_1)\, M_{W2}'$$

$$_0W_{59} = (\quad\text{"}\quad)\, M_{W1}'$$

$$_1W_{60} = (C_6 C_5 C_4 C_3 C_2' C_1') M_{W2}'$$
$$_0W_{60} = (\quad " \quad) M_{W1}'$$

$$_1W_{61} = (C_6 C_5 C_4 C_3 C_2' C_1) M_{W2}'$$
$$_0W_{61} = (\quad " \quad) M_{W1}'$$

$$_1W_{62} = (C_6 C_5 C_4 C_3 C_2 C_1') M_{W2}'$$
$$_0W_{62} = (\quad " \quad) M_{W1}'$$

$$_1W_{63} = (C_6 C_5 C_4 C_3 C_2 C_1) M_{W2}'$$
$$_0W_{63} = (\quad " \quad) M_{W1}'$$

MEMORY READ POWER AMPLIFIERS

M_{ra}

$$
\begin{aligned}
{}_{i}m_{ra} = [&(C_3' C_2' C_1') M_{r0} + (C_3' C_2' C_1) M_{r1} + (C_3' C_2 C_1') M_{r2} \\
&+ (C_3' C_2 C_1) M_{r3} + (C_3 C_2' C_1') M_{r4} + (C_3 C_2' C_1) M_{r5} \\
&+ (C_3 C_2 C_1') M_{r6} + (C_3 C_2 C_1) M_{r7}]\, M_s' \\
{}_{o}m_{ra} = [&(C_3' C_2' C_1') M_{r0}' + (C_3' C_2' C_1) M_{r1}' + (C_3' C_2 C_1') M_{r2}' \\
&+ (C_3' C_2 C_1) M_{r3}' + (C_3 C_2' C_1') M_{r4}' + (C_3 C_2' C_1) M_{r5}' \\
&+ (C_3 C_2 C_1') M_{r6}' + (C_3 C_2 C_1) M_{r7}' + M_s]
\end{aligned}
$$

M_{rb}

$$
\begin{aligned}
{}_{i}m_{rb} = [&(C_3' C_2' C_1') M_{r8} + (C_3' C_2' C_1) M_{r9} + (C_3' C_2 C_1') M_{r10} \\
&+ (C_3' C_2 C_1) M_{r11} + (C_3 C_2' C_1') M_{r12} + (C_3 C_2' C_1) M_{r13} \\
&+ (C_3 C_2 C_1') M_{r14} + (C_3 C_2 C_1) M_{r15}]\, M_s' \\
{}_{o}m_{rb} = [&(C_3' C_2' C_1') M_{r8}' + (C_3' C_2' C_1) M_{r9}' + (C_3' C_2 C_1') M_{r10}' \\
&+ (C_3' C_2 C_1) M_{r11}' + (C_3 C_2' C_1') M_{r12}' + (C_3 C_2' C_1) M_{r13}' \\
&+ (C_3 C_2 C_1') M_{r14}' + (C_3 C_2 C_1) M_{r15}' + M_s]
\end{aligned}
$$

M_{rc}

$$
\begin{aligned}
{}_1m_{rc} = [&(C_3' C_2' C_1')M_{r16} + (C_3' C_2' C_1)M_{r17} + (C_3' C_2 C_1')M_{r18} \\
&+ (C_3' C_2 C_1)M_{r19} + (C_3 C_2' C_1')M_{r20} + (C_3 C_2' C_1)M_{r21} \\
&+ (C_3 C_2 C_1')M_{r22} + (C_3 C_2 C_1)M_{r23}] M_s'
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rc} = [&(C_3' C_2' C_1')M_{r16} + (C_3' C_2' C_1)M_{r17}' + (C_3' C_2 C_1')M_{r18}' \\
&+ (C_3' C_2 C_1)M_{r19}' + (C_3 C_2' C_1')M_{r20}' + (C_3 C_2' C_1)M_{r21}' \\
&+ (C_3 C_2 C_1')M_{r22}' + (C_3 C_2 C_1)M_{r23}' + M_s]
\end{aligned}
$$

M_{rd}

$$
\begin{aligned}
{}_1m_{rd} = [&(C_3' C_2' C_1')M_{r24} + (C_3' C_2' C_1)M_{r25} + (C_3' C_2 C_1')M_{r26} \\
&+ (C_3' C_2 C_1)M_{r27} + (C_3 C_2' C_1')M_{r28} + (C_3 C_2' C_1)M_{r29} \\
&+ (C_3 C_2 C_1')M_{r30} + (C_3 C_2 C_1)M_{r31}] M_s'
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rd} = [&(C_3' C_2' C_1')M_{r24}' (C_3' C_2' C_1)M_{r25}' + (C_3' C_2 C_1')M_{r26} \\
&+ (C_3' C_2 C_1)M_{r27}' + (C_3 C_2' C_1')M_{r28}' + (C_3 C_2' C_1)M_{r29} \\
&+ (C_3 C_2 C_1')M_{r30}' + (C_3 C_2 C_1)M_{r31}' + M_s]
\end{aligned}
$$

M_{re}

$$
\begin{aligned}
{}_1m_{re} = [&(C_3' C_2' C_1')M_{r32} + (C_3' C_2' C_1)M_{r33} + (C_3' C_2 C_1')M_{r34} \\
&+ (C_3' C_2 C_1)M_{r35} + (C_3 C_2' C_1')M_{r36} + (C_3 C_2' C_1)M_{r37} \\
&+ (C_3 C_2 C_1')M_{r38} + (C_3 C_2 C_1)M_{r39}]\, M_s' \\
{}_0m_{re} = [&(C_3' C_2' C_1')M_{r32}' + (C_3' C_2' C_1)M_{r33}' + (C_3' C_2 C_1')M_{r34}' \\
&+ (C_3' C_2 C_1)M_{r35}' + (C_3 C_2' C_1')M_{r36}' + (C_3 C_2' C_1)M_{r37}' \\
&+ (C_3 C_2 C_1')M_{r38}' + (C_3 C_2 C_1)M_{r39}' + M_s]
\end{aligned}
$$

M_{rf}

$$
\begin{aligned}
{}_1m_{rf} = [&(C_3' C_2' C_1')M_{r40} + (C_3' C_2' C_1)M_{r41} + (C_3' C_2 C_1')M_{r42} \\
&+ (C_3' C_2 C_1)M_{r43} + (C_3 C_2' C_1')M_{r44} + (C_3 C_2' C_1)M_{r45} \\
&+ (C_3 C_2 C_1')M_{r46} + (C_3 C_2 C_1)M_{r47}]\, M_s' \\
{}_0m_{rf} = [&(C_3' C_2' C_1')M_{r40}' + (C_3' C_2' C_1)M_{r41}' + (C_3' C_2 C_1')M_{r42}' \\
&+ (C_3' C_2 C_1)M_{r43}' + (C_3 C_2' C_1')M_{r44}' + (C_3 C_2' C_1)M_{r45}' \\
&+ (C_3 C_2 C_1')M_{r46}' + (C_3 C_2 C_1)M_{r47}' + M_s]
\end{aligned}
$$

M_{rg}

$$
\begin{aligned}
{}_1m_{rg} = [&(C_3' C_2' C_1')M_{r48} + (C_3' C_2' C_1)M_{r49} + (C_3' C_2 C_1')M_{r50} \\
&+ (C_3' C_2 C_1)M_{r51} + (C_3 C_2' C_1')M_{r52} + (C_3 C_2' C_1)M_{r53} \\
&+ (C_3 C_2 C_1')M_{r54} + (C_3 C_2 C_1)M_{r55}]M_5'
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rg} = [&(C_3' C_2' C_1')M_{r48}' + (C_3' C_2' C_1)M_{r49}' + (C_3' C_2 C_1')M_{r50}' \\
&+ (C_3' C_2 C_1)M_{r51}' + (C_3 C_2' C_1')M_{r52}' + (C_3 C_2' C_1)M_{r53}' \\
&+ (C_3 C_2 C_1')M_{r54}' + (C_3 C_2 C_1)M_{r55}' + M_5]
\end{aligned}
$$

M_{rh}

$$
\begin{aligned}
{}_1m_{rh} = [&(C_3' C_2' C_1')M_{r56} + (C_3' C_2' C_1)M_{r57} + (C_3' C_2 C_1')M_{r58} \\
&+ (C_3' C_2 C_1)M_{r59} + (C_3 C_2' C_1')M_{r60} + (C_3 C_2' C_1)M_{r61} \\
&+ (C_3 C_2 C_1')M_{r62} + (C_3 C_2 C_1)M_{r63}]M_5'
\end{aligned}
$$

$$
\begin{aligned}
{}_0m_{rh} = [&(C_3' C_2' C_1')M_{r56}' + (C_3' C_2' C_1)M_{r57}' + (C_3' C_2 C_1')M_{r58}' \\
&+ (C_3' C_2 C_1)M_{r59}' + (C_3 C_2' C_1')M_{r60}' + (C_3 C_2' C_1)M_{r61}' \\
&+ (C_3 C_2 C_1')M_{r62}' + (C_3 C_2 C_1)M_{r63}' + M_5]
\end{aligned}
$$

CIRCUIT BOARD LOCATION

MEMORY SYNCHRONIZATION FLIP-FLOP

M_S $_1m_s = T_{40}$ 458-19

$_0m_s = T_{41a}$ 456-19

CLOCK JITTER FLIP-FLOP

C_j $_1c_j = C_j'$ 421-19

$_0c_j = C_j$ 220-19

MEMORY READ FLIP-FLOP

M_R $_1m_R = C_6' C_5' C_4' M_{Ra}$ 154-25

$+ C_6' C_5' C_4 M_{Rb}$ 154-25

$+ C_6' C_5 C_4' M_{Rc}$ 153-25

$+ C_6 C_5 C_4 M_{Rd}$ 153-25

$+ C_6 C_5' C_4' M_{Re}$ 152-25

$+ C_6 C_5' C_4 M_{Rf}$ 152-25

$+ C_6 C_5 C_4' M_{Rg}$ 252-24

$+ C_6 C_5 C_4 M_{Rh}$ 153-9

$_1m_r$ (Next Page)

MEMORY READ FLIP-FLOP (Cont)

$$\circ M_R = C_6' C_5' C_4' M_{Ra}' \qquad 554\text{-}25$$

$$+ C_6' C_5' C_4 M_{Rb}' \qquad 554\text{-}25$$

$$+ C_6' C_5 C_4' M_{Rc}' \qquad 552\text{-}9$$

$$+ C_6' C_5 C_4 M_{Rd}' \qquad 552\text{-}25$$

$$+ C_6 C_5' C_4' M_{Re}' \qquad 553\text{-}25$$

$$+ C_6 C_5' C_4 M_{Rf}' \qquad 553\text{-}25$$

$$+ C_6 C_5 C_4' M_{Rg}' \qquad 452\text{-}24$$

$$+ C_6 C_5 C_4 M_{Rh}' \qquad 453\text{-}24$$

APPENDIX III. CIRCUIT BOARD CHARTS

COMPUTER CONNECTOR SIGNAL CHART

The computer connector signal chart represents each cable receptacle, plug receptacle, and terminal board located in the computer assembly, excluding the signal circuit board receptacles and the power circuit board receptacles in the power supply assembly. The signal found at each pin of these receptacles is identified. (See figure .)

Each square represents a pin or a jack on the receptacle, and the letter designation or the combination letter and number designation within individual squares represents the signal or voltage found at that pin or jack. These designations are those used for the computer logic equations, or are defined in the computer wire list.

CIRCUIT BOARD CONNECTOR SIGNAL CHARTS

Six circuit board connector signal charts are required to represent each plug-in circuit board receptacle mounted on the two side castings of the computer assembly. Again, the signal charts also identify each signal found at each pin or jack of these receptacles, with individual squares representing individual pins or jacks. Each connector signal chart illustrates one complete row of 40 connectors. There are six such rows of connectors, three on each side casting; thus, there are six charts. (See figures 41 through 47.)

Two of the rows of connectors on each side casting are made up of double-receptacle connectors, and one is made up of single-receptacle connectors. Therefore, there are actually five rows of receptacles on each side: number designations 101-140 through 501-540 (side No. 1) and 141-180 through 541-580 (side No. 2). Thus, a 3-digit number specifies exactly one receptacle. For example, receptacle 129 specifies the top receptacle in the twenty-ninth position of side No. 1.

The layout of each signal chart is discussed in the following paragraphs.

100-200 SERIES-SIDE NO. 1

This chart is arranged to facilitate signal checking of the logic network boards plugged into receptacles 102 and 202 through 131 and 231. All the pins on the particular board are represented by one horizontal row of squares. The pin numbers for each logic network board are printed at the top of the chart and grouped into individual gates on the boards. The output pin numbers are raised above the input pin numbers for each gate. The number at the top of each gate grouping indicates the number of diodes used in that gate.

The second part of this chart illustrates the signals found at each pin of the eight write switches plugged into side No. 1.

300 SERIES-SIDE NO. 1

This chart is arranged to facilitate signal checking of the flip-flop boards plugged into side No. 1. As for all other signal charts, one horizontal row of squares represents all the pins on a particular circuit board. The pin numbers along the top of the chart are grouped into four sets of input-output signals (four flip-flops on a board) and one set of voltages. Also the signal found at each pin of the write amplifier circuit boards and the clock power amplifier boards is shown.

400-500 SERIES-SIDE NO. 1

This chart is arranged similarly to the 100-200 series chart for side No. 1. This arrangement facilitates signal checking of the logic network boards plugged into 402 and 502 through 424 and 524. Also, the signals found at each pin of 3 computer network boards and the 11 nonmain memory read amplifier boards are illustrated.

100-200 SERIES-SIDE NO. 2

This chart is arranged similarly to the 100-200 series chart for side No. 1. This arrangement facilitates signal checking of the logic network boards plugged into 151 and 251 through 179 and 279. Also, the signals found at each pin of the other three computer network boards are illustrated.

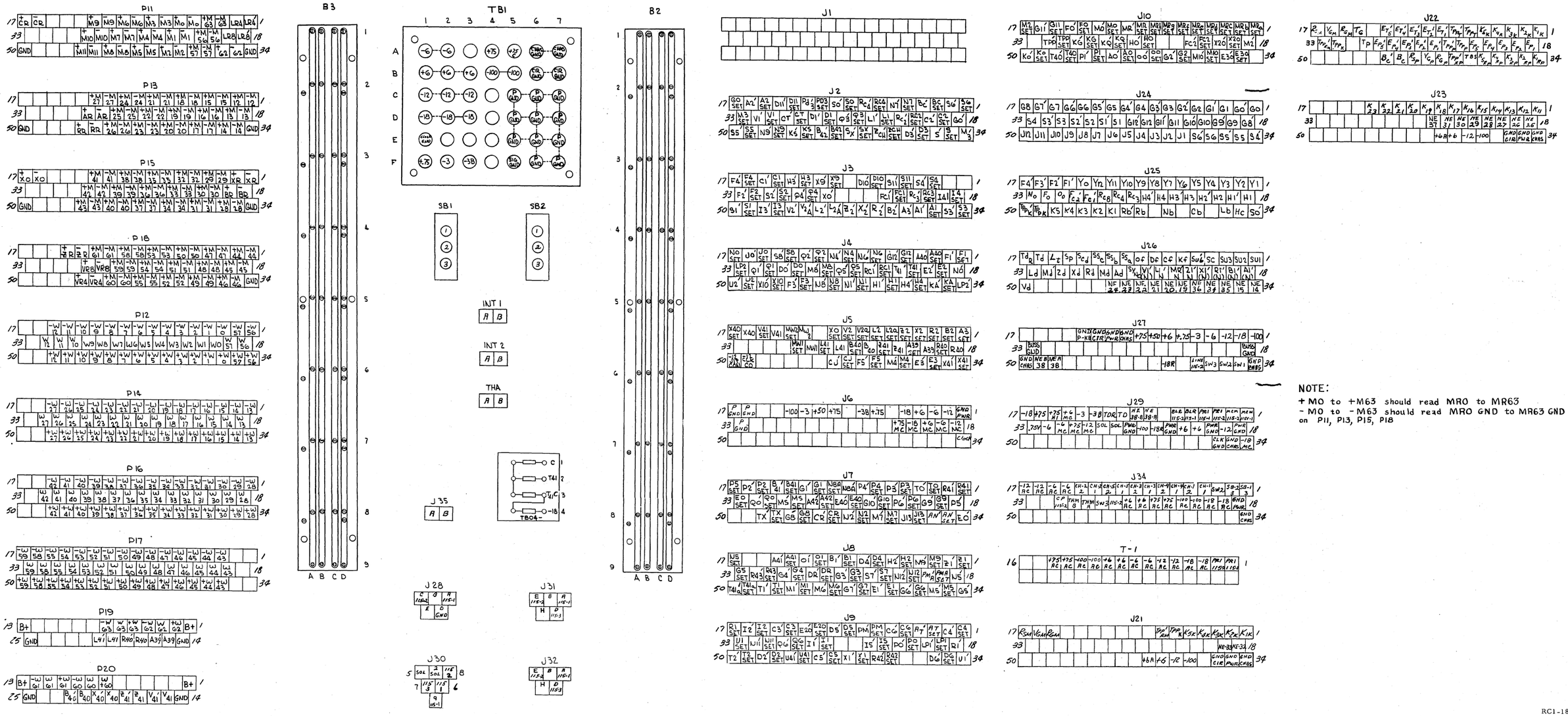
300 SERIES-SIDE NO. 2

This chart is arranged similarly to the 300 series chart for side No. 1. This arrangement facilitates signal checking of the flip-flop

boards plugged into side No. 2. Also, the signals found at each pin of the eight read switching circuit boards are illustrated.

400-500 SERIES-SIDE NO. 2

This chart is arranged similarly to the 100-200 series chart for side No. 1. This arrangement facilitates signal checking of the logic network boards plugged into 452 and 552 through 479 and 579 (except that receptacles 462 and 562 comprise a spare connector).



NOTE:

+MO to +M63 should read MRO to MR63
-MO to -M63 should read MRO GND to MR63 GND on P11, P13, P15, P18

RC1-182 A

Figure 41 . Computer Connector Signal Chart

(101 THROUGH 140) ← → (201 THROUGH 240) 100-200 PLUG SERIES SIDE *1

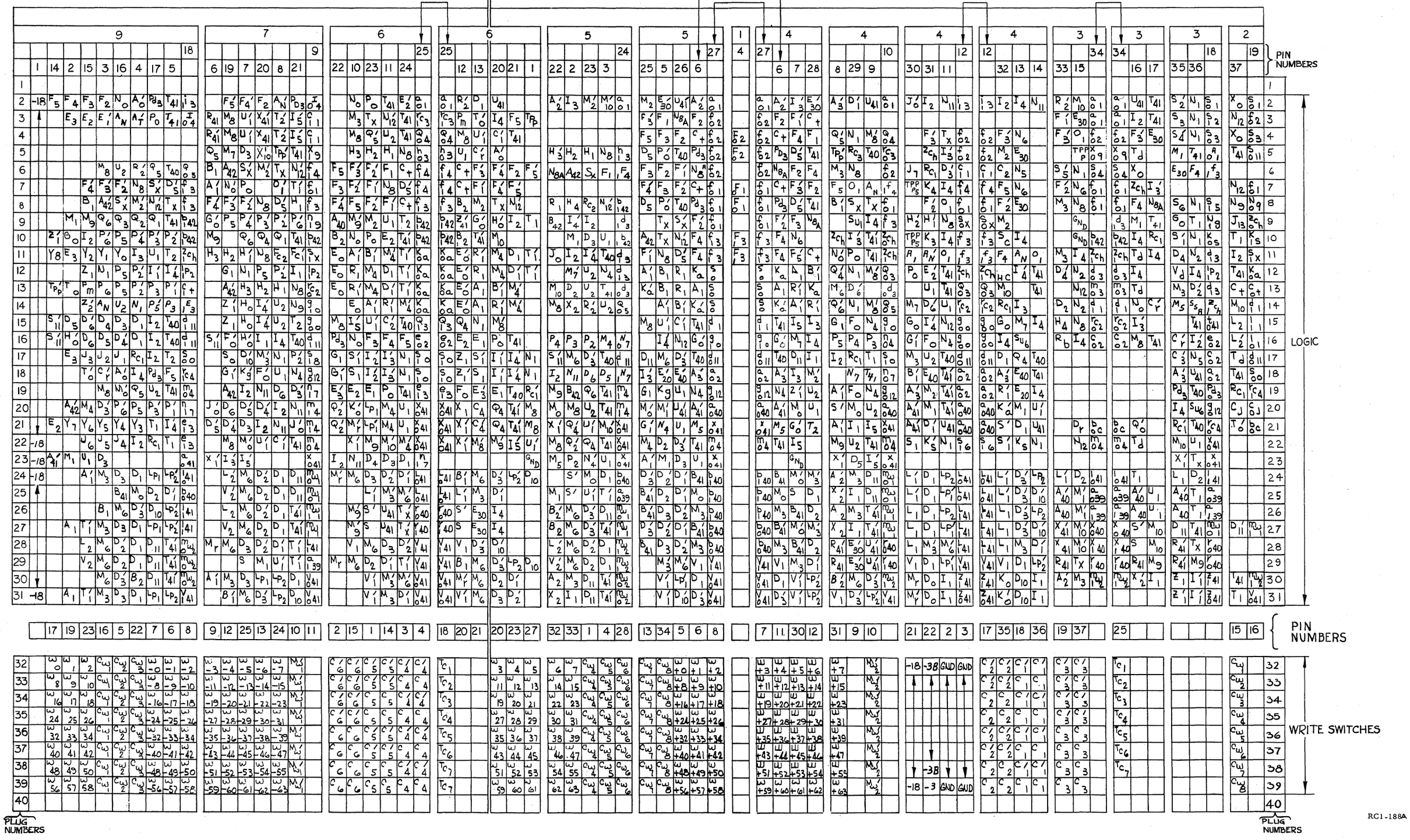


Figure 42. Circuit Board Connector Signal Chart No. 1

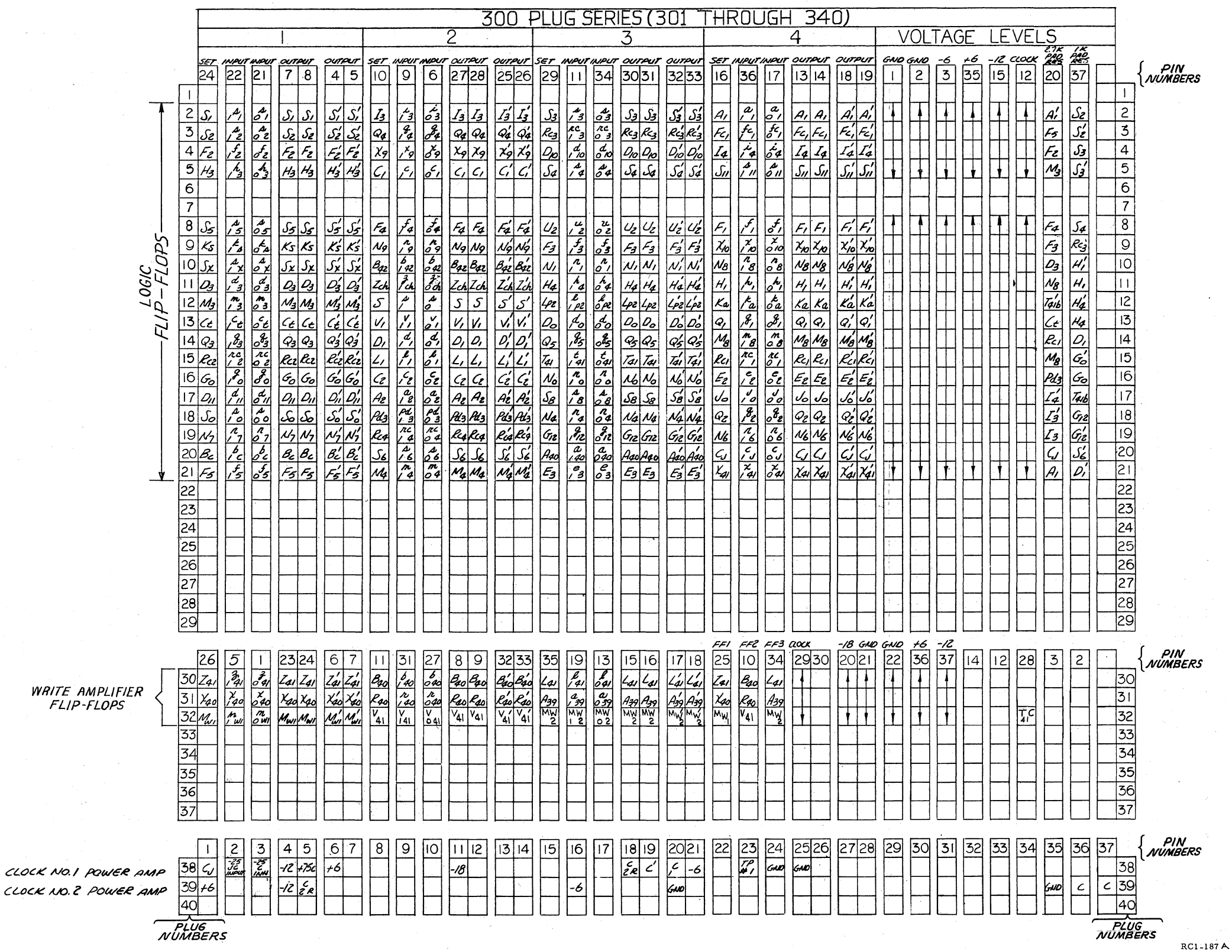


Figure 43. Circuit Board Connector Signal Chart No. 2

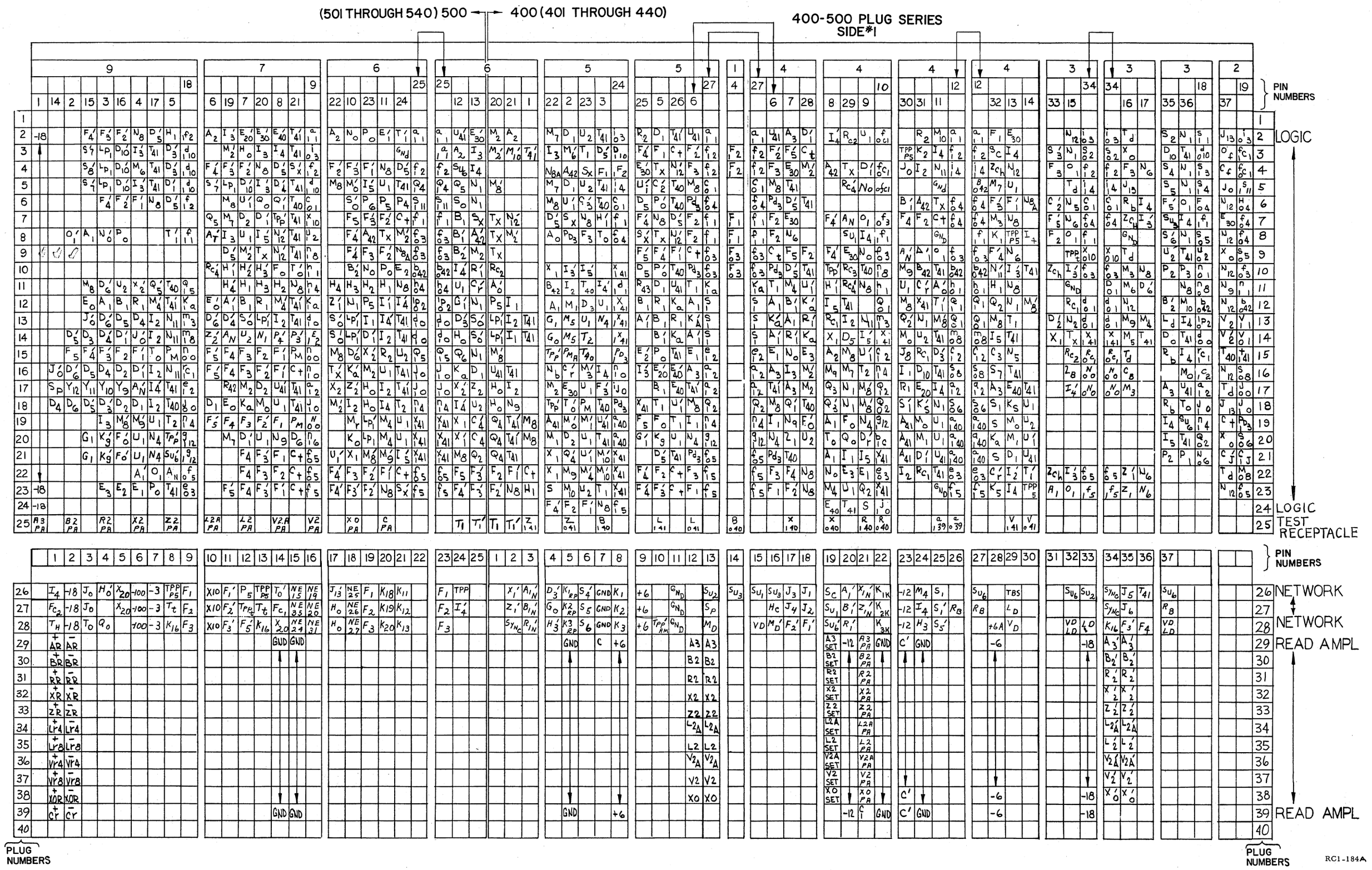


Figure 44. Circuit Board Connector Signal Chart No. 3

100-200 PLUG SERIES - SIDE #2

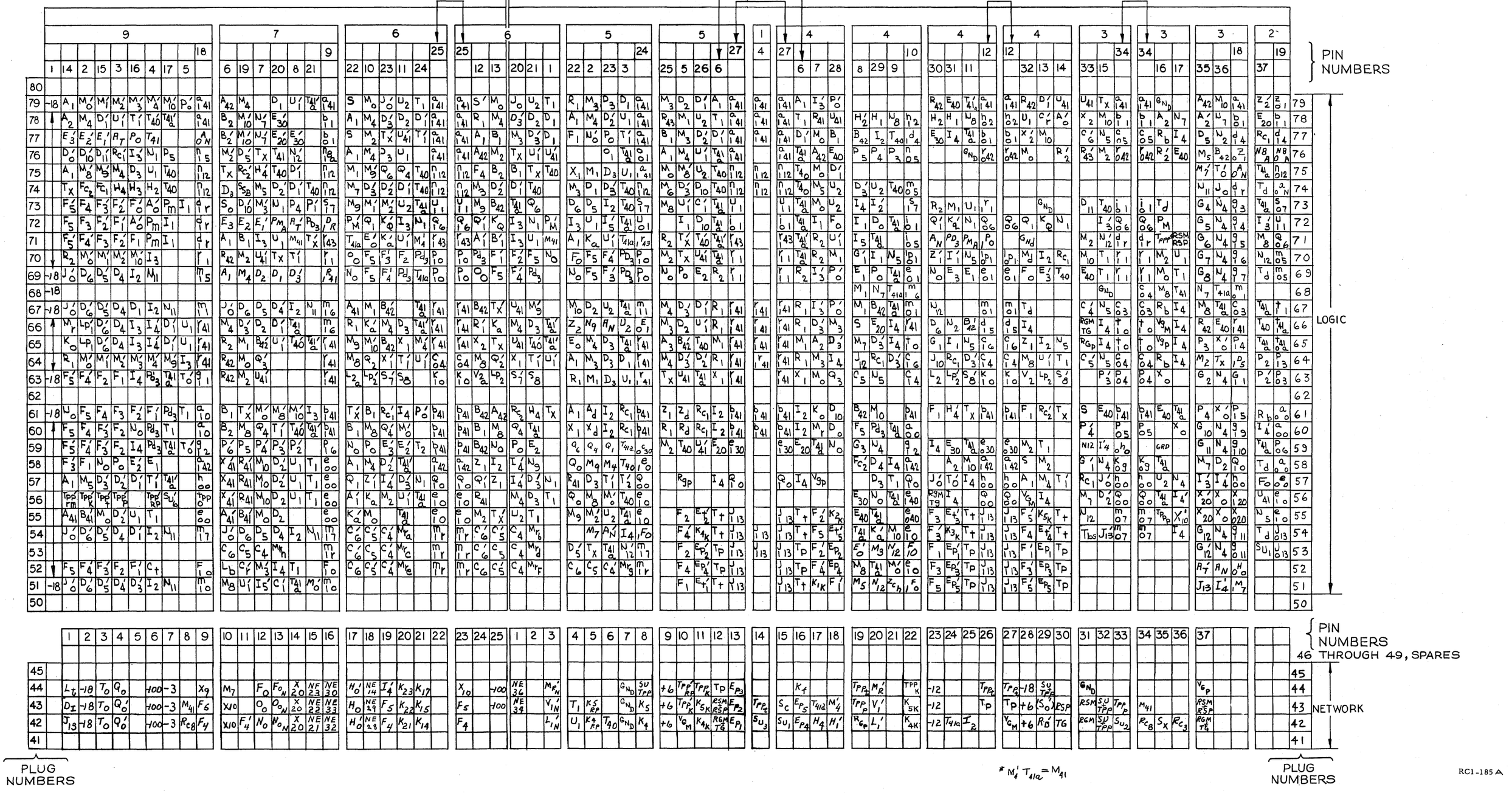


Figure 45. Circuit Board Connector Signal Chart No. 4

300 SERIES SIDE # 2
(341 THROUGH 380)

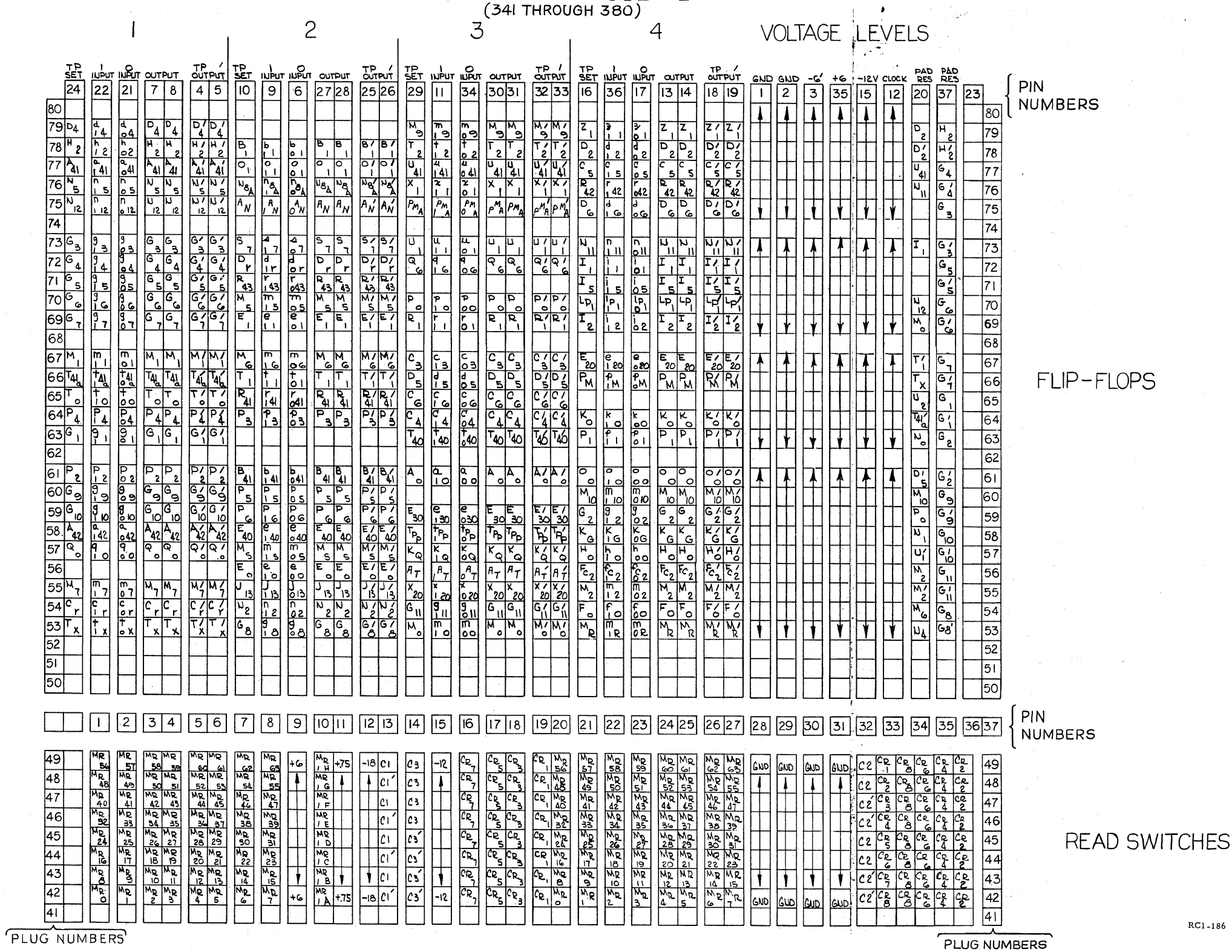


Figure 46. Circuit Board Connector Signal Chart No. 5

400-500 PLUG SERIES SIDE #2

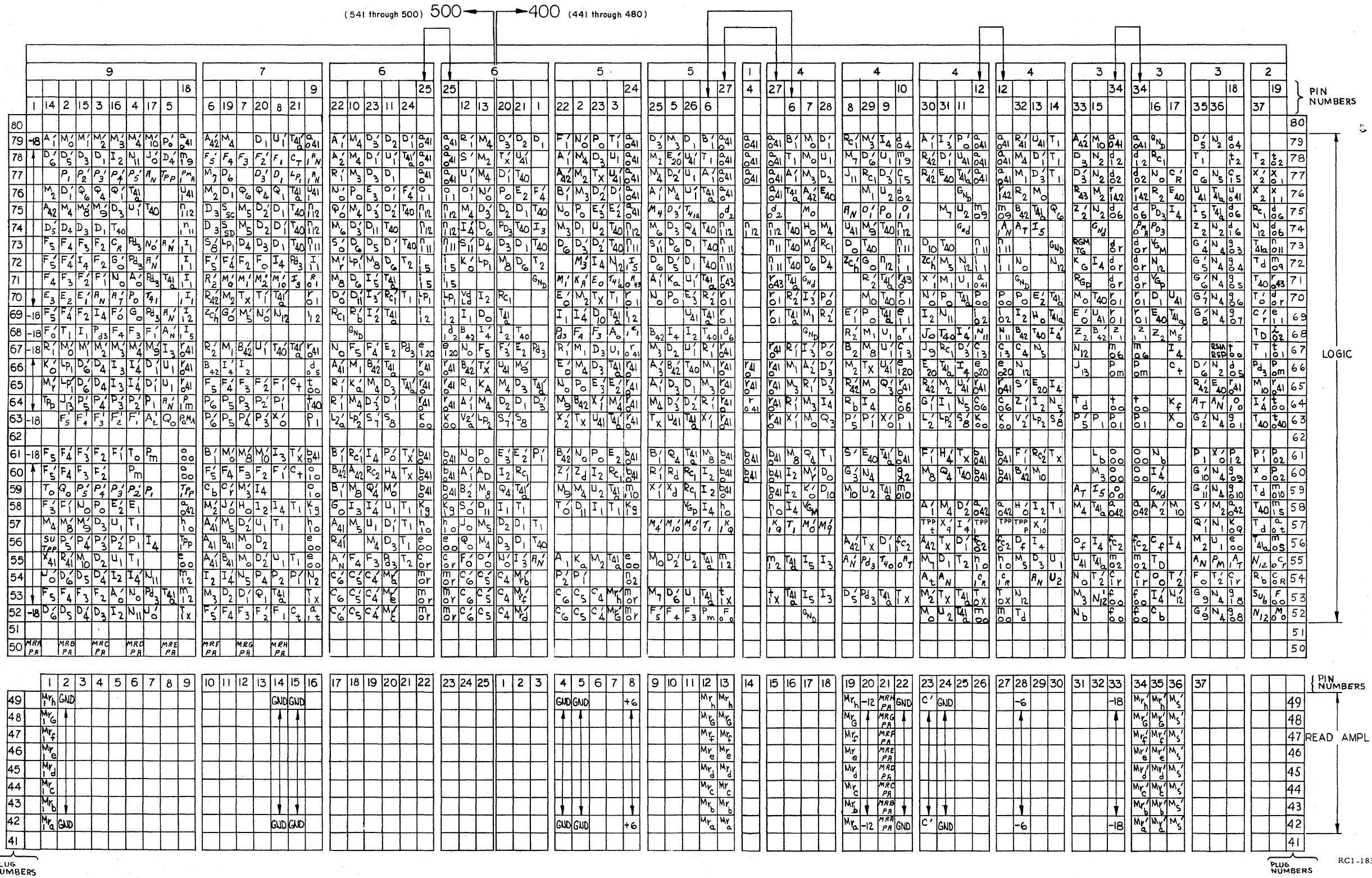


Figure 47. Circuit Board Connector Signal Chart No. 6