TRW-130 *AN/UYK-1*

DATA PROCESSING SYSTEM

TRW-130 *AN/UYK-1*

MACHINE REFERENCE MANUAL

M250-2U19 REV FEB 63

RW *DIVISION*

8433 FALLBROOK AVENUE • CANOGA PARK, CALIFORNIA • DIAMOND 6-6000

TABLE OF CONTENTS

TABLE OF CONTENTS (Cont.)

TABLE OF CONTENTS (Cont.)

LIST OF ILLUSTRATIONS

LIST OF TABLES

LIST OF TABLES (Cont.)

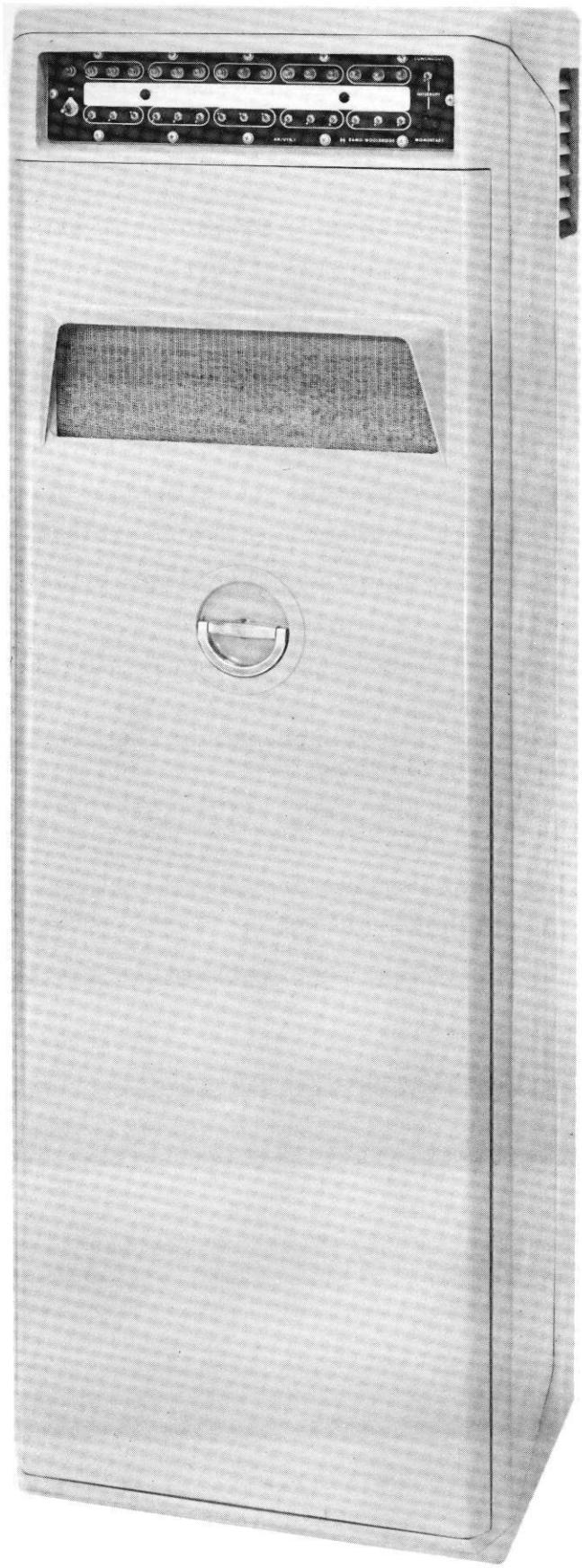


Figure 1-1. AN/UYK-1 (TRW-130) Computer

CHAPTER I

INTRODUCTION TO SYSTEM

1.0 SYSTEM CHARACTERISTICS

The AN/UYK-1 computer (figure 1-1) is designed to meet the computation and data processing requirements of a shipboard digital computer. Its foremost characteristics are:

(a) Ruggedness. It can operate in adverse environments aboard ship or on land.

(b) Small size. The entire computer, without dismantling, can be loaded into, and maneuvered within, confined quarters.

(c) Compatibility with NTDS. It can communicate with the Navy Tactical Data System (NTDS) peripheral equipment.

(d) Inexpensiveness. It is economically justified even for small tasks.

The system is organized around the arithmetic and input-output control elements, which can be connected to a variety of input and output devices, and NTDS devices. The keynote of this system is its adaptability in a wide range of problems.

Stored program control is implemented in a manner which permits the user to create the instruction repertoire most useful for the task at hand. In the interests of hardware simplicity, versatility, operating reliability, and economy, the AN/UYK-1 computer employs a practical variation of the stored-logic computer control concept. Logic building block commands, referred to as logands, can be arranged in ordered sequences and loaded by the user directly into the computer's working storage. Any type of operation or conventional subroutine function can be represented by such a sequence of logands. These logical sequences are

known as logic programs, or lograms. Once the desired lograms are written and loaded into the computer, they define its instruction list. The AN/UYK-1 computer is so designed that operational control flows automatically from logram to logram without the time loss customarily associated with conventional interpretively controlled equipment. Thus, the characteristic command form of any computer can be used to instruct the AN/UYK-1 if the user so desires. Once he becomes proficient, he can augment or otherwise alter his instruction set at will to fit the programming requirements at hand.

An AN/UYK-1 computing system may be increased to an intermediate equipment capacity via a selection of internal storage and input-output devices. The internal magnetic core storage of the computer may be expanded from a minimum of 8,192 to a maximum of 32,768 words. The choice of peripheral units for a standard system (in addition to NTDS devices) includes a high-speed paper tape reader and punch, a teletype printer, and one or more magnetic tape units. A teletype printer serves as the minimum input-output device and becomes the communication link between operator and machine.

1.1 GENERAL DESCRIPTION

The AN/UYK-1 computer system consists of the central computer and various input-output devices which are NTDS-compatible, plus other standard peripheral devices. A block diagram of a typical AN/UYK-1 system is shown in figure 1-2.

1.1.1 The Computing Element

The AN/UYK-1 computer is a binary-parallel-word machine with a versatile stored-program-command repertoire which facilitates both numerical computation and alphanumerical character-manipulation. Its basic internal storage consists of 8,192 words of magnetic core storage. Each word contains 15 binary bits (plus a parity bit) and thus may contain any unsigned number between zero and 32,768. In its internal operation, the AN/UYK-1 uses two's complement arithmetic. Since the

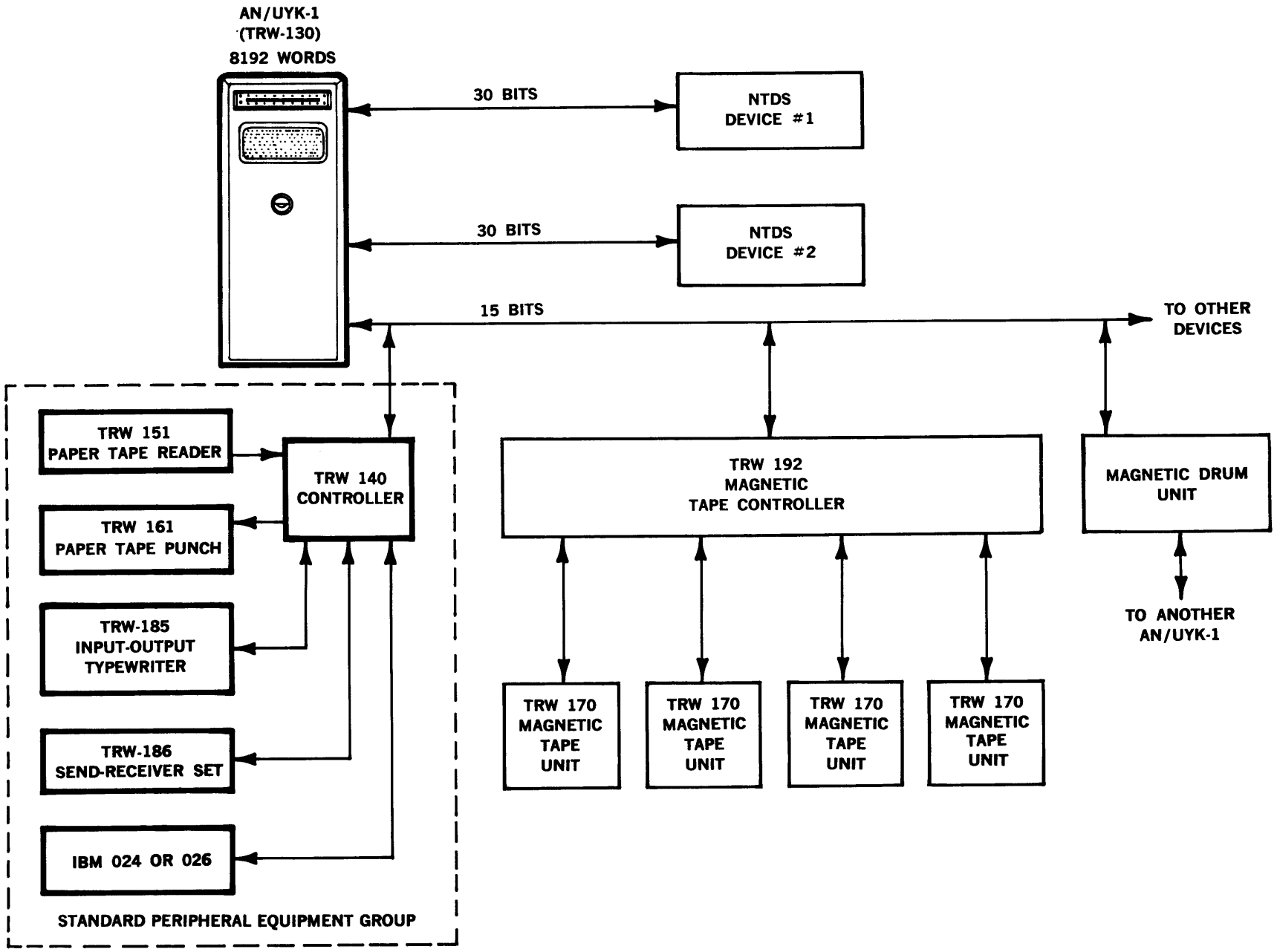
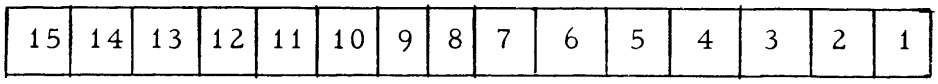


Figure 1-2. AN/UYK-1 (TRW-130) Electronic Data Processing System.

internal code of the machine is binary, the machine can be programmed as if a word contained any type of information. For example, each word may be considered to contain the binary coded equivalent of two alphanumerical characters, a signed numerical quantity between minus 16,384 and plus 16,383, an address, an instruction, or an arbitrarily coded bit pattern representing the status of physical control devices. The bits of a word are numbered from right to left, 1–15 as:



Bit 1 is the low-order bit, and bit 15 is the high-order bit, or sign bit.

The command structure of the computer is an adaptation of the stored-logic principle. Several thousand combinative commands can be used by a programmer to directly control the machine in its own language. The computer may be readily programmed in terms of instructions arbitrarily specified by the user to fit his problem requirements and programming preferences. An instruction is implemented by writing a corresponding logram in machine language, loading the logram into the machine, and referring to that logram in writing a program whenever the specified action is desired. The AN/UYK-1 is designed to run interpretively without the need for a time-consuming interpretive control routine. A counter is merely passed from logram to logram as the machine sequences through the programmer's instructions. Each logram uses this counter to find its data, and advances the counter by the number of data locations accessed by it. The last data location designates the location of the first logand of the next logram. The computer is delivered with a Logand Assembly program to facilitate the construction of customized instruction repertoires. The AN/UYK-1 programmer is therefore free to choose the type and number of instructions he wants to use, write the lograms to implement each of these instructions, program his problem in terms of his chosen symbolic macro-instructions, and assemble his data for interpretation by the machine as it processes his problem.

1.1.2 Input-Output Control Element

The AN/UYK-1 input-output control element selects the type of devices to be communicated with (NTDS or peripheral) by the selection of the proper cables, selects a particular device, and initiates the transfer of data to and from the computing element and the various input-output devices.

The input-output control can also respond to interrupts from the I/O devices. This allows control timing of the data flow to maintain the maximum speed when several devices are in operation. Effectively, the input-output control element communicates to the computing element that a device is prepared to receive or transmit data.

The AN/UYK-1 communicates with external devices by means of three sets of cables, each consisting of an input and an output cable. Two of these cables are reserved for NTDS devices and the third is reserved for peripheral devices that may be connected either to the input or the output cable or to both. The devices connected to the third cable can be disconnected internally from this cable set so that other devices can be connected.

CHAPTER II

SYSTEM ORGANIZATION AND DESCRIPTION

2.0 SYSTEM ORGANIZATION

This chapter contains descriptions of the organization of the AN/UYK-1 computer and of its basic functions. An understanding of this information at the programmer level is a prerequisite to facile communication with the system in its own language. The actual command structure of the machine is described in Chapter III in terms of the fundamental operations presented in the following sections. The programmer must avoid thinking that he must program in terms of such basic functions. The advantage gained of virtually complete order code flexibility would be more than offset by the tedium of coding at such a basic level. Instead, most of the basic function sequences which are performed of necessity during problem coding are made available as programmer options with each computer command. The combinative possibilities of the AN/UYK-1's order code are so large that direct memorization is impractical, if not impossible. Once the system's design is understood, however, its inherent logical power is readily exploitable by programmers.

2.1 STORED-LOGIC ADAPTATION

Control of the AN/UYK-1 computer is based upon an adaptation of the stored-logic concept. Much of the wired control logic which has heretofore mechanized machine-language instructions is omitted in the stricter form of the stored-logic concept. Instead, only the microcommand logic is wired into the hardware. The logic for any type of instruction above the microcommand level is then stored in the computer's memory as a microprogram. Such a design permits the instantaneous logical organization of the computer to be specified by the programmer as a part of his problem-solving code. The academically powerful and flexible stored-logic design concept has long been advocated as the answer to the many programming limitations imposed by fixed computer instruction repertoires.

The AN/UYK-1 computer is controlled by combinations of logical commands, or LOGANDS. The most common microcommand sequences necessary to maintain programming control are wired into the computer as implicit operations. These include such functions as incrementing the command address in anticipation of the need to obtain the next command once the current command has been executed, keeping track of the current operand address, regenerating a core storage location after data readout, clearing a core storage location before write-in, and making register exchanges to preserve data during normal command and operand address updating. The implicit operation sequence corresponding to the command type and form of storage reference used is augmented by the explicit microoperations corresponding to the particular command to form the total of microcommands implemented by that command. Wide latitude and flexibility result from the combinative possibilities inherent in this command organization without the tedium and storage dedication of the direct microprogramming approach.

2.2 AN/UYK-1 COMPUTER

A block diagram of the AN/UYK-1 computer is shown in figure 2-1.

The principal elements of the computer are discussed in the following sections. These elements are: the memory; the six 15-bit flip-flop registers L, E, A, P, M, and T; the Adder (AD); the Address Counter (AC); the Carry Indicator (C); and the Overflow Indicator (OV).

2.2.1 Memory

The basic internal memory of the AN/UYK-1 computer consists of an 8192 15-bit-word, random access, magnetic core unit. The memory can be expanded to 32,768 words. Readout is destructive and, therefore, each readout must be followed by a write-in operation to preserve the accessed storage cell's contents. Times for read and write operations are each 3 microseconds, resulting in a 6-microsecond memory cycle.

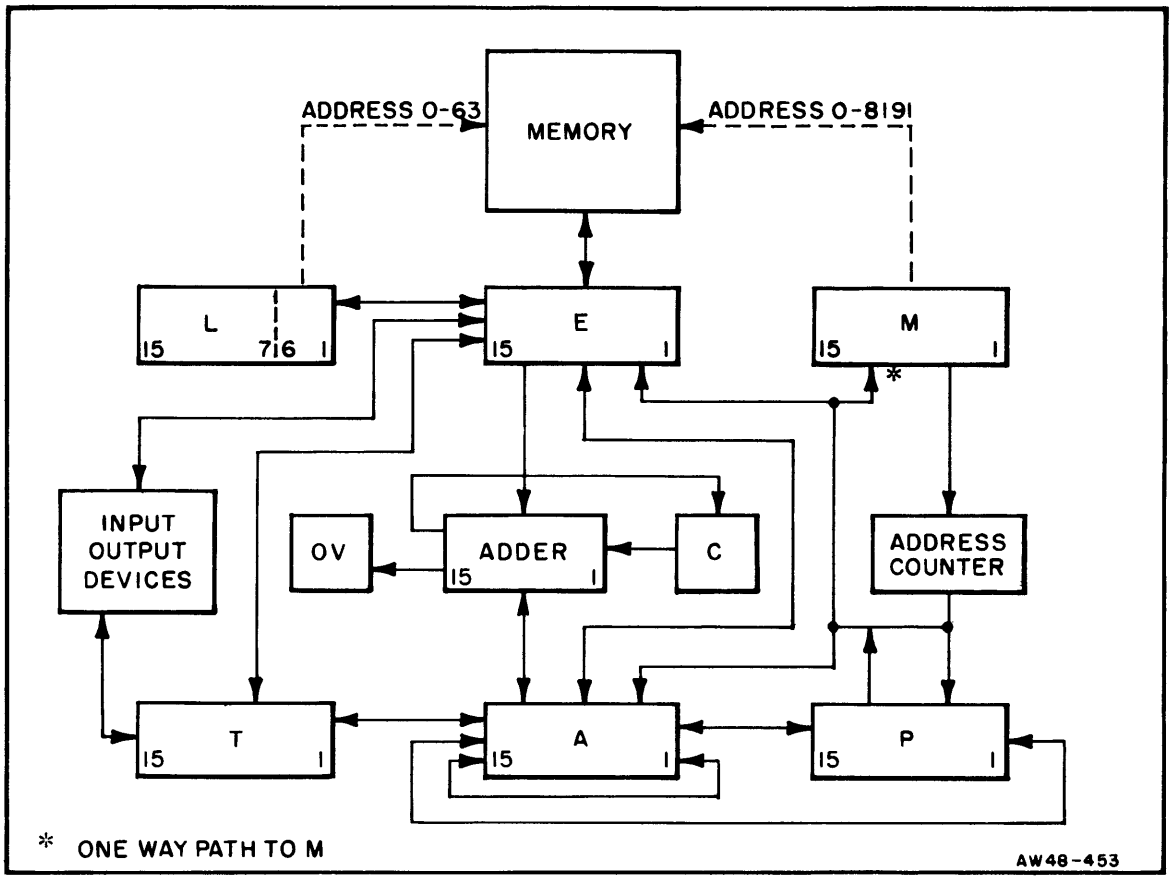


Figure 2-1. AN/UYK-1 Computer Block Diagram

2.2.2 Logand Register (L Register)

The logand register is a 15-bit flip-flop register that holds logands while they are being interpreted. It accepts 15-bit operands from the E register only. The low-order six bits of the L register may be used to address memory, thus providing access to memory locations 0 through 63 (the "scratchpad"). The low-order four bits of the L register may be zero-tested or decremented by unity.

2.2.3 Exchange Register (E Register)

The exchange register is a 15-bit flip-flop register used principally for holding 15-bit operands being read from or written into memory. The E register accepts 15-bit operands from the L register, the P register, the A register, the T register, and the M register via the address counter. The E register serves also as an exchange register for the input/output control unit. In this capacity, it can accept 15-bit operands in parallel;

or it can accept partial operands and subsequently parallel-transfer its contents in preparation for the transmission of 30-bit operands.

2.2.4 Arithmetic Register (A Register)

The arithmetic register is a 15-bit flip-flop register that holds the results of the addition operation and some logical operations, the most significant part of the product in the multiply operation, and the remainder after division. It can be shifted left or right, either alone or coupled to the P register. It accepts 15-bit operands from the E register, the M register (via the address counter), the P register, the T register, and the adder.

2.2.5 Program Address Register (P Register)

The P register is a 15-bit flip-flop register which nominally holds operand addresses. It is also used as an extension of the A register for shifting, multiply, and divide operations. It holds the least significant part of the product in multiplication and the quotient after division. It can be shifted left or right while coupled to the A register. It accepts 15-bit operands from the E register, the M register (via the address counter), and the A register.

2.2.6 Memory Address Register (M Register)

The memory address register is a 15-bit flip-flop register used principally to address any of 32,768 locations in memory. It also serves as the logand sequence counter. At the end of each logand execution it contains the address of the next logand to be executed. The M register accepts 15-bit operands from the E register, the P register, the A register, and the address counter.

2.2.7 Input-Output Register (T Register)

The T register is a 15-bit flip-flop register used principally to buffer data between the computer and the output devices. It can transmit to such devices up to 15 bits in parallel while the rest of the computer

proceeds independently. It may also be used as an auxiliary storage register. It accepts 15-bit operands from the E register and the A register.

2.2.8 Adder

The adder is a 15-position parallel full adder that is used to form sums and to compare 15-bit operands. Each adder position produces a sum digit and a carry digit from the corresponding digits of the A register, the E register and the carry from the preceding adder position.

The carry input to adder position 1 is obtained from the carry indicator.

2.2.9 Address Counter

The address counter is a 15-position parallel half-adder that increments addresses by unity. Each position of the address counter has two inputs and two outputs. The inputs are:

(a) The bit from the corresponding position of the M register.

(b) The carry from the next-least-significant position of the address counter.

The outputs are:

(a) The sum bit.

(b) The carry to the next-most-significant address counter position.

The sum bit is the least-significant bit of the binary sum of the two inputs and the carry is the most-significant bit of this sum.

Depending on the result desired, the carry input c_0 to the first position of the address counter is set to zero or to one on the clock pulse preceding the one on which a transfer out of the address counter is desired.

If it is set to zero, the sum digits transferred out of the counter are identical to the input digits from M. If it is set to one, the sum is a number that is one greater than the number in M. In either case, the carry output from the 15th position of the address counter is lost. If M initially holds all ones, and c_0 is set to one, the sum is all zeros.

2.2.10 Carry Indicator (C)

The carry indicator (C) is a one-position indicator that furnishes the carry input to the low-order position of the adder. It is set to zero by the Add Single, Add Least and Complement Clear commands, and to one by the Complement Set command. In addition, it is automatically set (either to zero or to one) by c_{15} of the adder, as the result of any addition operation.

2.2.11 Overflow Indicator (OV)

The overflow indicator is a one-position indicator that holds the indication of overflow in the addition of signed operands. It is set to one by an Add Single or Add Most command when the sign bits of the addend and augend (e_{15} and a_{15}) are the same but the sign bit of the sum (s_{15}) is different.

The overflow indicator is set to zero by any conditional logand which designates the overflow indicator as a condition.

2.3 AN/UYK-1 INPUT/OUTPUT CONTROL UNIT

The input/output unit is an integral part of the AN/UYK-1 computer. Sufficient logic is contained in the I/O portion of the computer to permit communication with various external devices with no intermediate control logic. The elements of the input/output control unit are shown in figure 2-2.

The computer communicates with the external devices via three sets of cables, each of which contains an input and an output channel. Two of the sets, A and B, are reserved for NTDS devices, which require that 30 bits of parallel data be transferred. The 30 bits of data are transferred

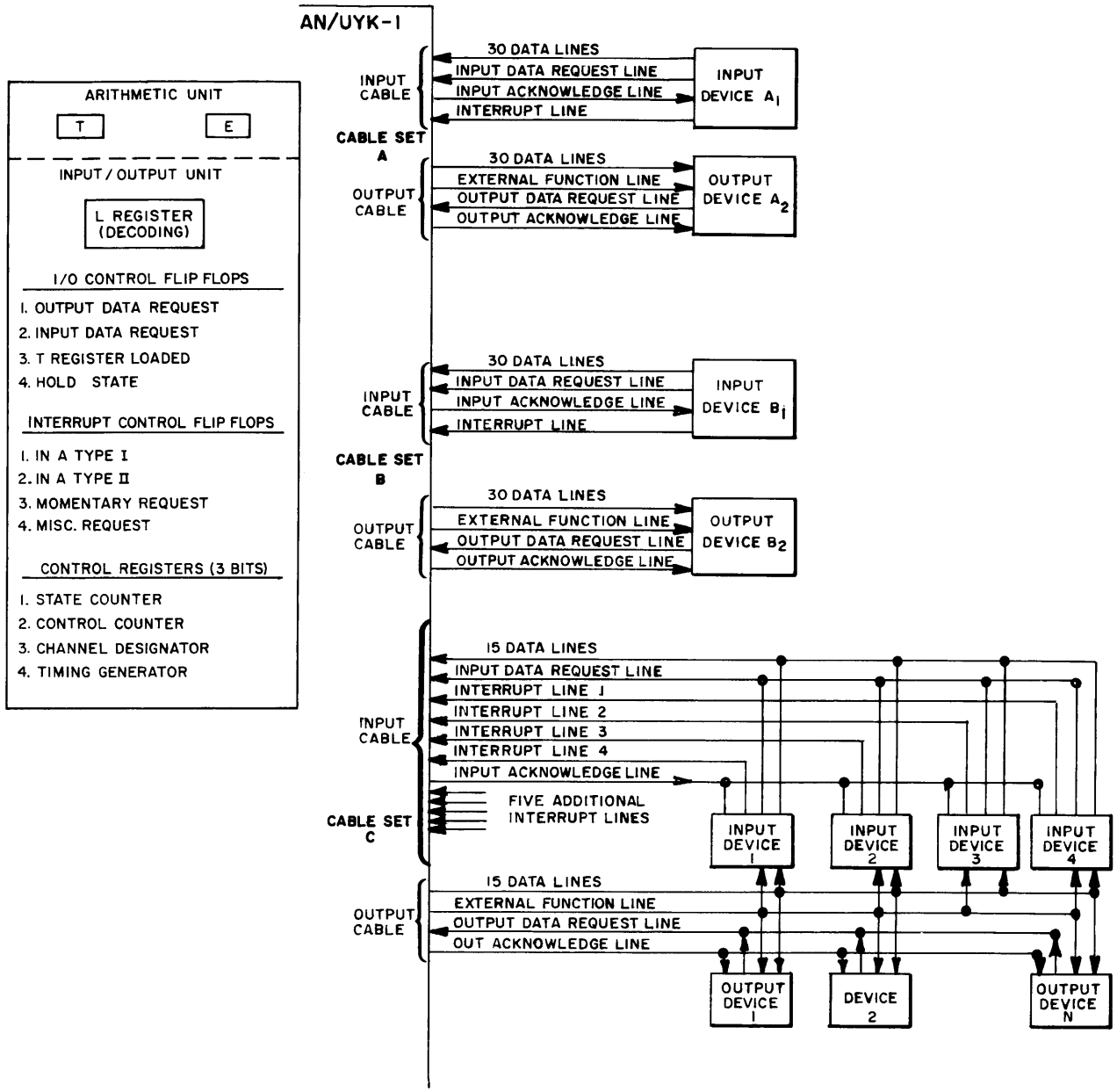


Figure 2-2. AN/UYK-1 Input-Output Diagram

via the 15-bit T and E registers. The third cable set, C, is used for standard peripheral devices such as a Teletype, a high-speed paper tape punch, and a punched-tape reader, etc. The 15-bit output data line in cable C is accompanied by three control lines: the output data request line, the output acknowledge line, and the external function line. The 15-bit input data line in cable C has eleven associated control lines: the input data request line, the input acknowledge line, and nine interrupt lines. Cable C always inputs to E and outputs from T.

The selection of cables is done with control function (CF) logands. The selection of a particular device is done with an external function (EF) logand and the transfer of data is accomplished with word or block input/output logands (WI, WO, BI, and BO).

Interrupts to the computer are controlled by the I/O unit. The AN/UYK-1 can respond to two types of interrupts: type I and type II. Type I interrupts are generated by the request signals associated with the selected cables and I/O devices. Type I interrupts are inhibited only if the computer is in a type I interrupt or is executing a word or block logand. Type II interrupts are generated from either external or internal interrupt signals. A type II interrupt inhibits other type II interrupts but does not inhibit type I interrupts.

2.4 COMPUTER CONTROL PANELS

The design of the control panels of the AN/UYK-1 computer is functional, simple, and compact. Two panels facilitate both the normal operational functions and the routine maintenance control functions. Figure 2-3 illustrates the operational control panel and figure 2-4, the maintenance control panel. The former provides the operator with the status display and mode control facilities needed during normal program-controlled operation. The latter augments these facilities with the additional man-machine communication controls required during abnormal operation or routine maintenance periods.

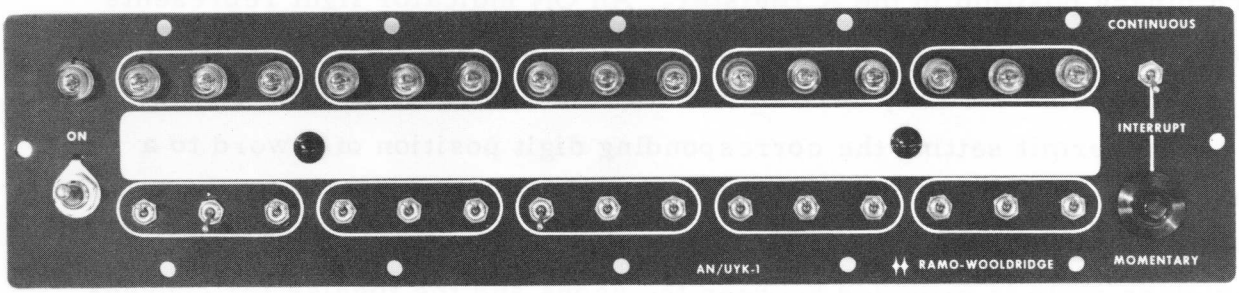


Figure 2-3. Operational Control Panel

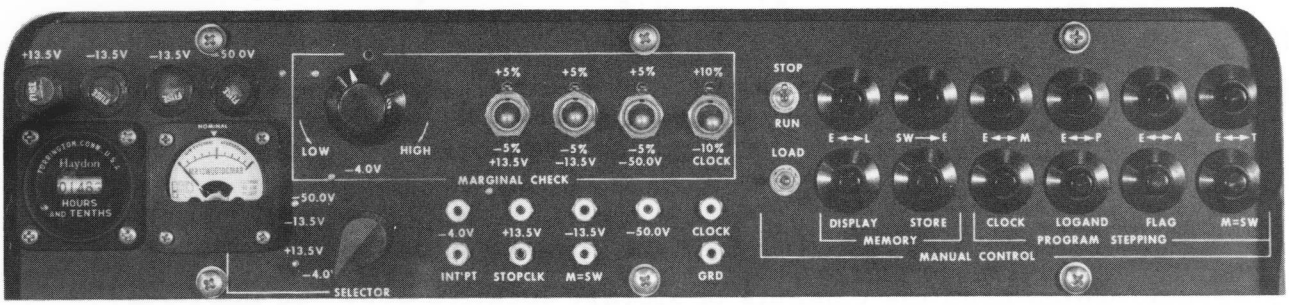


Figure 2-4. Maintenance Control Panel

2.4.1 Operational Control Panel

The control panel contains 15 neon indicator lights, which display each binary position in the A register. An ON indicator light represents a binary one in that digit position of the register, and an OFF indicator light represents a binary zero. Beneath each indicator light is a toggle switch to permit setting the corresponding digit position of a word to a binary one or zero. A toggle in the UP position represents a binary one and a toggle in the DOWN position represents a binary zero. A frosted plexiglass strip is provided between the neons and the toggles for convenience in the penciled labeling or temporary name strip identification of particular bit groupings. The specific functions of individual switches and indicators on both panels are explained in the following sections.

2.4.1.1 ON-OFF SWITCH

Turning this switch on supplies ac power to all power supplies and fans. A neon indicator light directly above this switch glows when power is on.

2.4.1.2 MOMENTARY INTERRUPT PUSHBUTTON

Pressing this pushbutton causes the computer to execute a type II interrupt, which uses the contents of storage location 00001 as the interrupt control address.

2.4.1.3 CONTINUOUS-INTERRUPT SWITCH

Setting this toggle switch in the UP position causes the computer to execute a type II interrupt at the completion of each logand. Once the interrupt has taken place, further interrupts are inhibited until the interrupt condition has been terminated.

2.4.2 Maintenance Control Panel

The twelve pushbuttons and the two toggle switches on the right side of the maintenance panel provide both the operator and the maintenance

Table 2-a. Wired-In Bootstrap Routine

Cell	Contents	
0	-	Noise
1	00002	Pulled out and entered in M by interrupt logand.
2	EF/DM/10	External function which access cell 00003 for paper tape reader connect command. Paper tape reader connect command is transmitted to reader.
3	40001	Paper tape reader connect command
4	WI/DL/01	Input one word, DL option, accessing cell 00001. This and the following two characters from the tape are assembled into a 15-bit word, as are every three characters thereafter, with the last character of a group going into the most-significant position in the word. The character read in is always in the least-significant five bits of the 15-bit word.
5	LA/DL/01	Load A with the input character, DL option, cell 00001.
6	SO/DM/D/R05	Shift AP right, open, five bits. This shifts the input character into the P register. The ten most-significant bits of each input word are discarded.
7	WI/DL/01	Input one word, DL, cell 00001.
10	LA/DL/01	Load A, DL, cell 00001.
11	SO/DM/D/R05	Shift AP right, open, five positions.
12	WI/DL/01	Input one word, DL, cell 00001.
13	LA/DL/01	Load A, DL, cell 00001.
14	SO/DM/D/R05	Shift AP right, open, five bits. This completes the assembly of the first complete 15-bit word.
15	SE/IL/17	Store E, IL option. This places the assembled word, which is in P at the beginning of this logand into the cell whose address is in cell 17 (i.e., cell 21 initially).
16	SP/DM/C/AP	Store P in cell 00017, DM option. This places the storage address for the next assembled input word in cell 00017. Secondary command: exchange A and P.

Table 2-a. Wired-In Bootstrap Routine (Cont.)

Cell	Contents	
17	00021	Initially contains 00021. This number is incremented by one every time it is used.
20	BR/IM/NQ	Branch on (A) ≠ (00021)' to (00022), IM option. If the branch condition is satisfied, the next logand is executed from the cell 00004.
21	—	Initially contains noise, Thereafter, it contains the first assembled word from the tape, which is the complement of the address of the last cell to be loaded with the bootstrap routine.
22	00004	The second word assembled from the tape must repeat this word.
23	—	This and the cells following are loaded with the new program from the paper tape reader.

engineer with direct control over the computer's register contents. The computer must be stopped before the pushbuttons become effective. The combined facilities of these controls plus those of the control panel permit complete register and memory cell loading and display under manual control. The left half of this panel contains the test inputs, the test outputs and the hardware control features necessary for equipment malfunction diagnosis and maintenance.

2.4.2.1 STOP-RUN TOGGLE SWITCH

When this switch is set in the RUN position, the computer is under complete program control and the manual controls are ineffective. When the switch is in the STOP position, the computer is under joint program and manual control. The manual controls are explained in 2.4.2.5. After start-up or after the switch is thrown from RUN to STOP, the computer stops after clock one of the logand.

2.4.2.2 LOAD TOGGLE SWITCH (SPRING RETURN)

Setting this spring toggle to the UP position while the computer is stopped energizes storage locations 00000 through 00022_8 with a punched tape bootstrap loading routine. Storage location 00002 contains the first logand of the loader to be executed. Table 2-a (pages 2-11a and 2-11b, gives the details of the wired-in Bootstrap loader routine.

2.4.2.3 MEMORY CONTROL BUTTONS

Pressing the STORE button stores the contents of the E register in the memory location addressed by the contents of the M register.

Pressing the DISPLAY button places in the E register the contents of the memory location addressed by the M register.

In both instances, the M register is incremented by unity.

2.4.2.4 MANUAL TRANSFER BUTTONS

The action of each button is described below in terms of the register transfers described in Section 2.5.2.

Pressing the E⟷L button initiates an *ele* register transfer.

Pressing the SW ⟷E button causes the settings of the 15 toggle switches to be entered into the E register.

Pressing the E ⟷M button initiates an *eme* register transfer.

Pressing the E ⟷P button initiates an *epe* register transfer.

Pressing the E ⟷A button initiates an *eae* register transfer.

Pressing the E ⟷T button initiates an *ete* register transfer.

2.4.2.5 PROGRAM-STEPPING BUTTONS

When the computer is in the Stop mode and is not operating, it may be started in one of four manual modes.

The Clock mode causes the computer to execute one logand clock cycle and stop.

The Logand mode causes the computer to operate through clock 1 of the next logand and then stop.*

The Flag mode causes the computer to operate until a one in bit 6 of a BR or an SK logand is encountered; then the computer operates through clock 1 of the next logand.*

---

*The E register contains the next logand to be executed, and the M register contains the address plus one of that logand.

The M = SW mode causes the computer to operate until the contents of the M register at the end of a logand cycle is equal to the setting of the fifteen toggle switches and operates through clock 1 of the next logand. *

2.4.2.6 RUNNING-TIME METER

This meter indicates the total time that power has been applied to the entire machine.

2.4.2.7 MILLIAMMETER

This meter measures the current flow in the four principal power supplies. A precision wire-wound resistor for each voltage is mounted behind the panel. Scale of 0 to 1.5 on the meter is interpreted as 0 to 150 percent of nominal.

2.4.2.8 SELECTOR SWITCH

This switch selects the voltage levels to be measured by the milliammeter.

2.4.2.9 MARGINAL CHECK PROVISIONS

Four toggle switches independently raise or lower each of the three main dc voltages by 5 percent and the clock by 10 percent as an aid in isolating intermittent troubles or in predicting future ones. A potentiometer is provided for continuous variation of the critical logical reference voltage (-4 volts).

2.4.2.10 VOLTAGE AND CLOCK TEST POINTS

These test points may be used to connect power to auxiliary test devices, to check the clock pulse width and amplitude, and to check power supply voltages.

2.4.2.11 JACKS FOR EXTERNAL TEST DEVICES

The M = SW jack supplies a signal to external devices when the contents of the M register are equal to the setting of the loading switches. An

*See footnote page 2-12

external device can gate the computer's clock pulse amplifier by connection to the STOPCLK jack. It can electrically interrupt the computer through connection to the INT'PT jack.

2.5 BASIC COMPUTER OPERATIONS

2.5.1 General

The AN/UYK-1 computer is a synchronous device driven by a 333-kilocycle clock. Clock pulses thus occur every 3 microseconds. On each clock pulse, certain basic operations are initiated. Operations initiated on one pulse are carried out in the 3-microsecond interval that follows, and are completed by the time the next pulse occurs.

The purpose of this section is to describe the basic operational logic of the AN/UYK-1. The manner in which these operations are combined into meaningful sequences is described in the next chapter.

2.5.2 Register Transfers

The design of the AN/UYK-1 registers provides for the replacement, on every clock pulse, of the contents of each register by some 15-bit quantity. If the contents of a register are to be "held" until the next clock pulse, the output of that register is simply returned to its input, as illustrated in figure 2-5.

If the contents of one register are to be copied into another register, the output of the first is switched to the input of the second, as well as being returned to the input of the first. This type of transfer is illustrated in figure 2-6.

The interchange of the contents of two registers on a single clock pulse is accomplished by switching the output of the first to the input of the second, and the output of the second to the input of the first, as illustrated in figure 2-7.

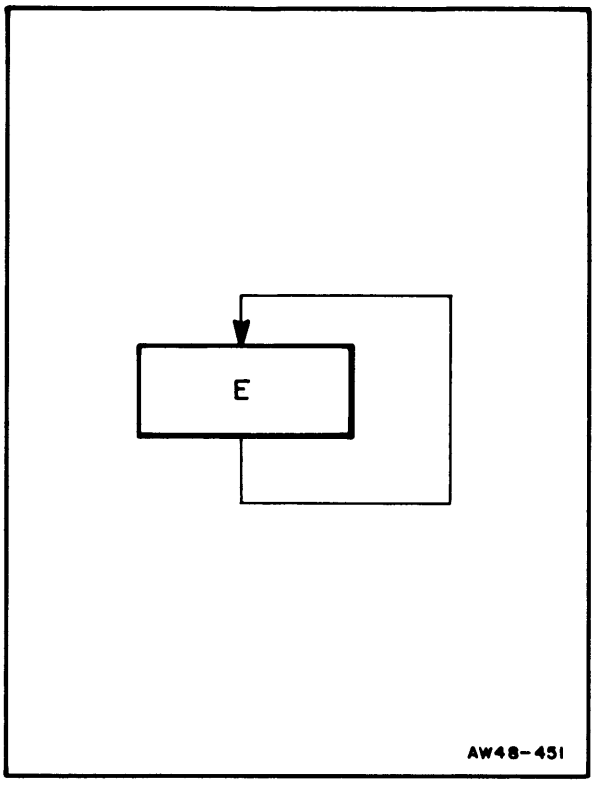


Figure 2-5. The Transfer ee

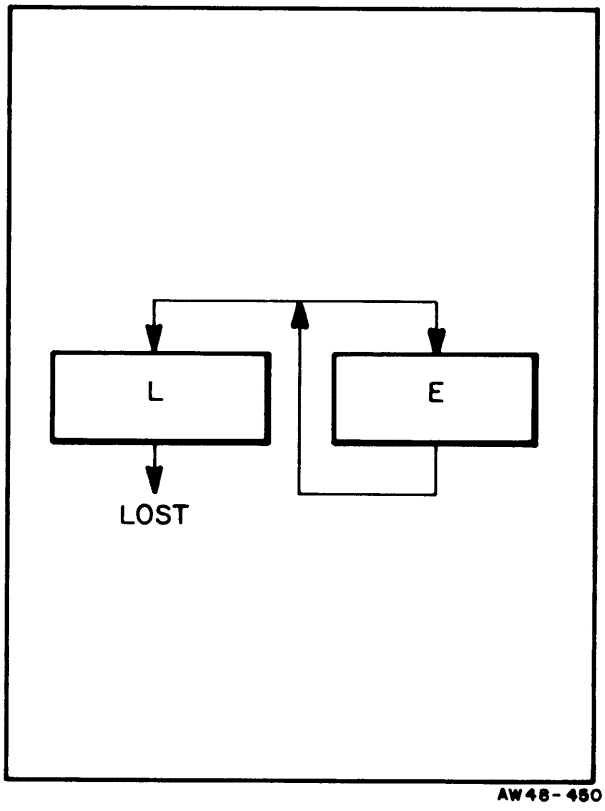


Figure 2-6. The Transfer el

Similar techniques are used for transfers involving more than two registers. For example, the transfer apea is accomplished by switching the output of A to the input of P, the output of P to the input of E, and the output of E to the input of A, as in figure 2-8.

All transfers from the M register are made via the address counter. This provides the option of transferring the contents of M as is, or of transferring the contents of M counted by one. In the former case, the carry input to the low-order position of the address counter is set to zero on the clock pulse preceding the transfer operation, so that the output of the address counter is identical to the input, i.e., identical to (M). In the latter case, the initial carry input is set to one, so that the output of the address counter is equal to (M) + 1. These two cases are illustrated in figures 2-9 and 2-10. Since only the M register is input to the address counter, only quantities in M can be incremented.

Table 2-1 defines the register transfers that have been implemented in the AN/UYK-1. Each transfer operation is identified by a string of lower-case letters that identify the registers involved in the operation, and that are arranged in such a manner that the output of the register identified by one letter is the input to the register identified by the following letter. If the input to a register is not otherwise specified, it is assumed to be identical with the output of that register. If the output of a register is not designated as input to any register, it is assumed to be lost. For example, in the transfer el, the output of E is switched to the input of L; the input to E, not being otherwise specified, is assumed to be the same as the output of E; and the output of L is lost.

The two possible transfers out of the M register, namely, (M) and (M) + 1, are denoted by the letters m and mi, respectively. These symbols imply the setting of the address counter carry input, on the previous pulse.

The effect of an operation initiated on clock pulse n is defined by showing the contents of the L, E, A, P, M, and T registers on clock pulse n + 1, in terms of the contents of these registers on clock pulse n. The former are assumed to be ℓ, e, a, p, m, and t, respectively.

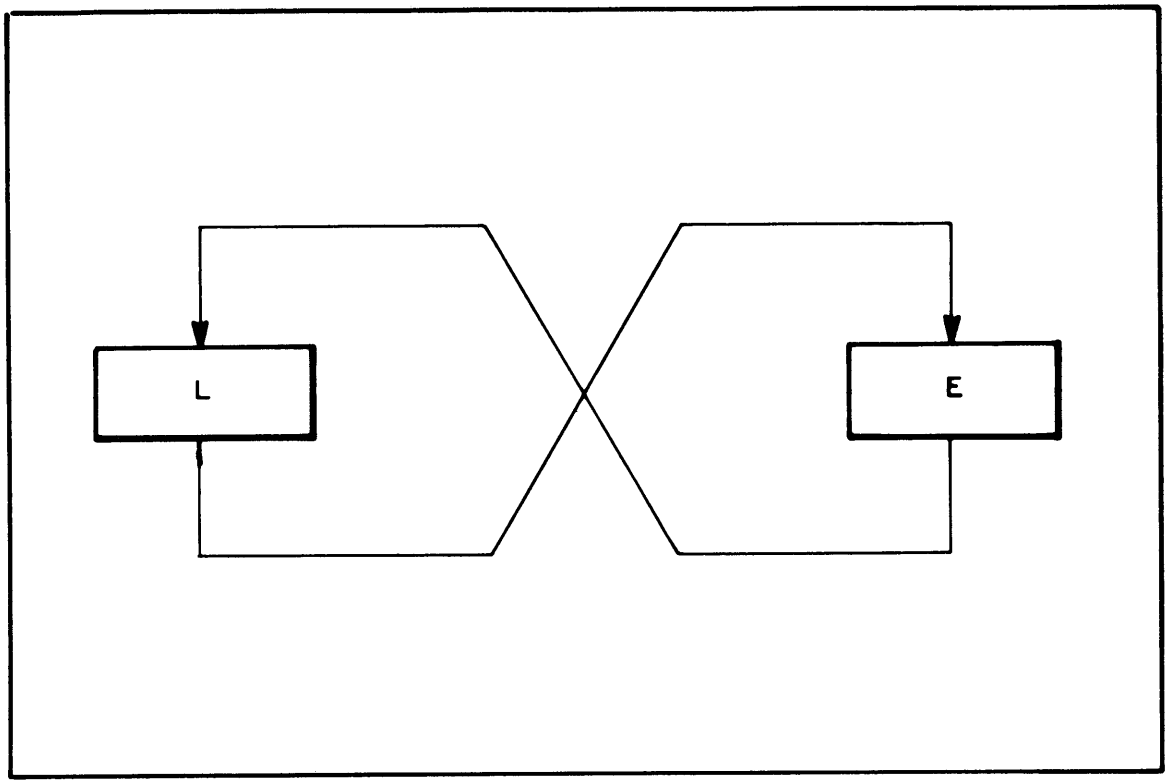


Figure 2-7. The Transfer e*l*e

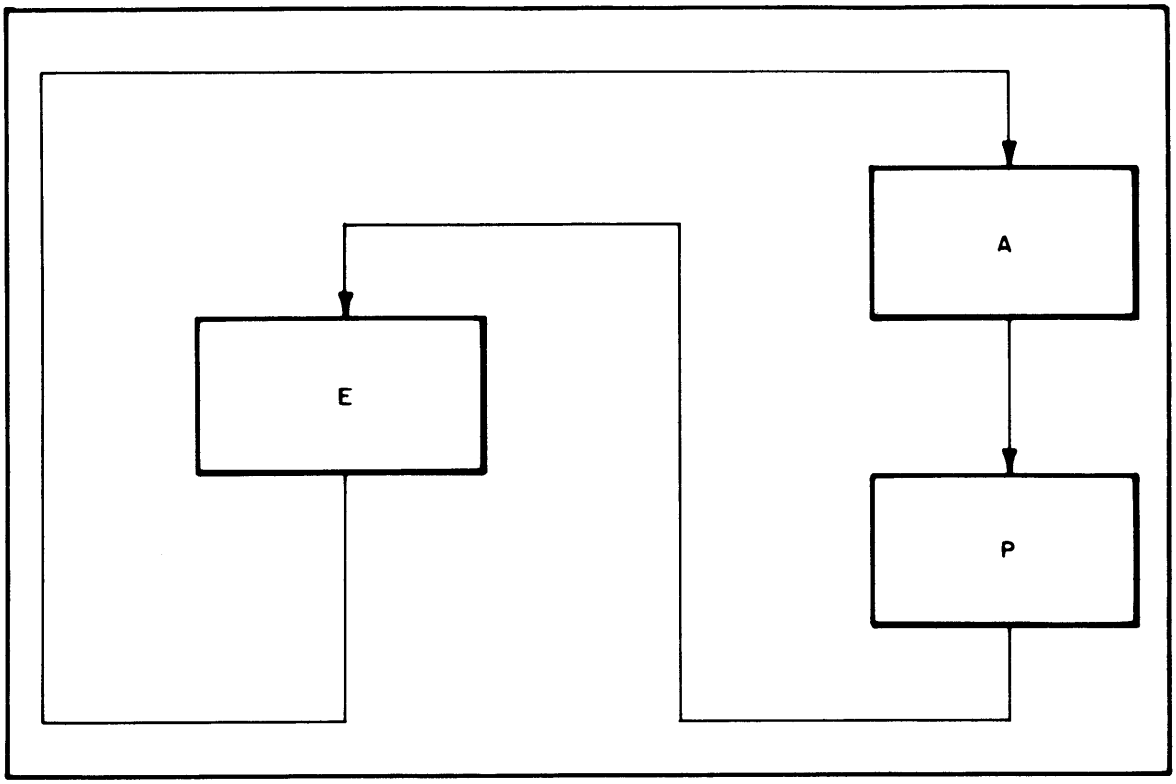


Figure 2-8. The Transfer apea

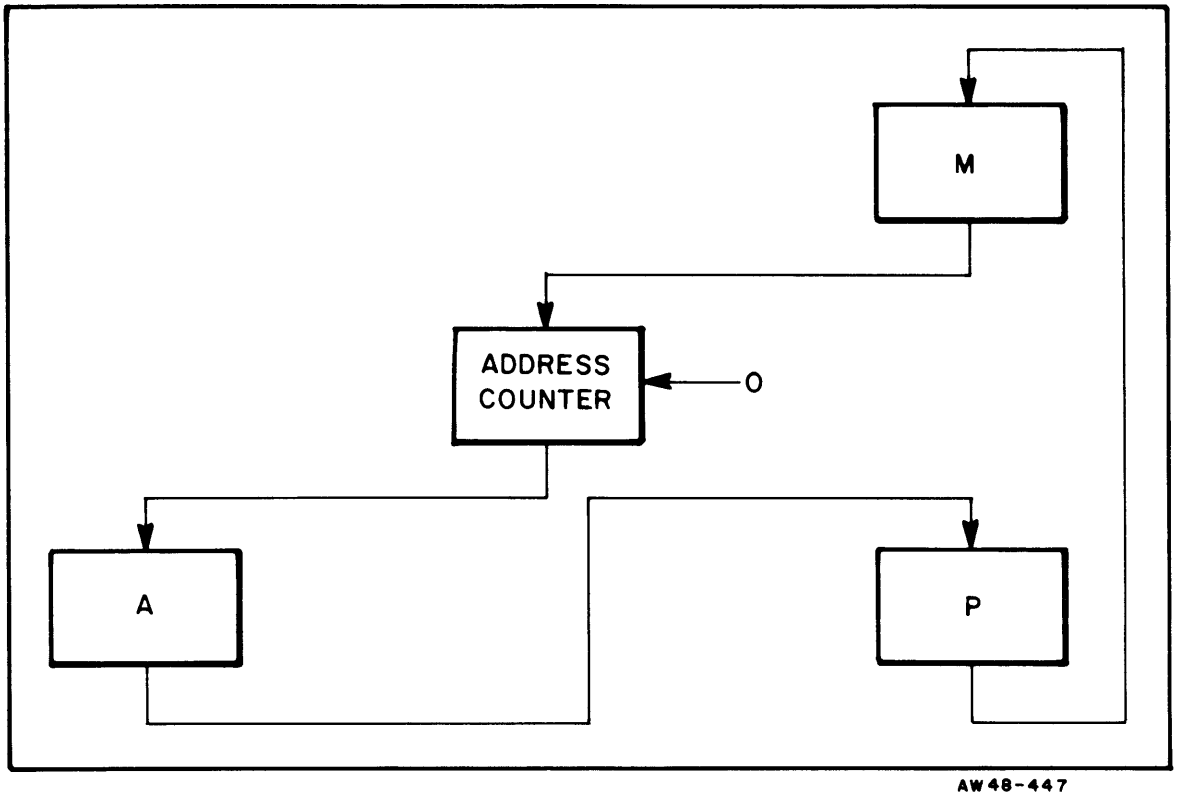


Figure 2-9. The Transfer pmap

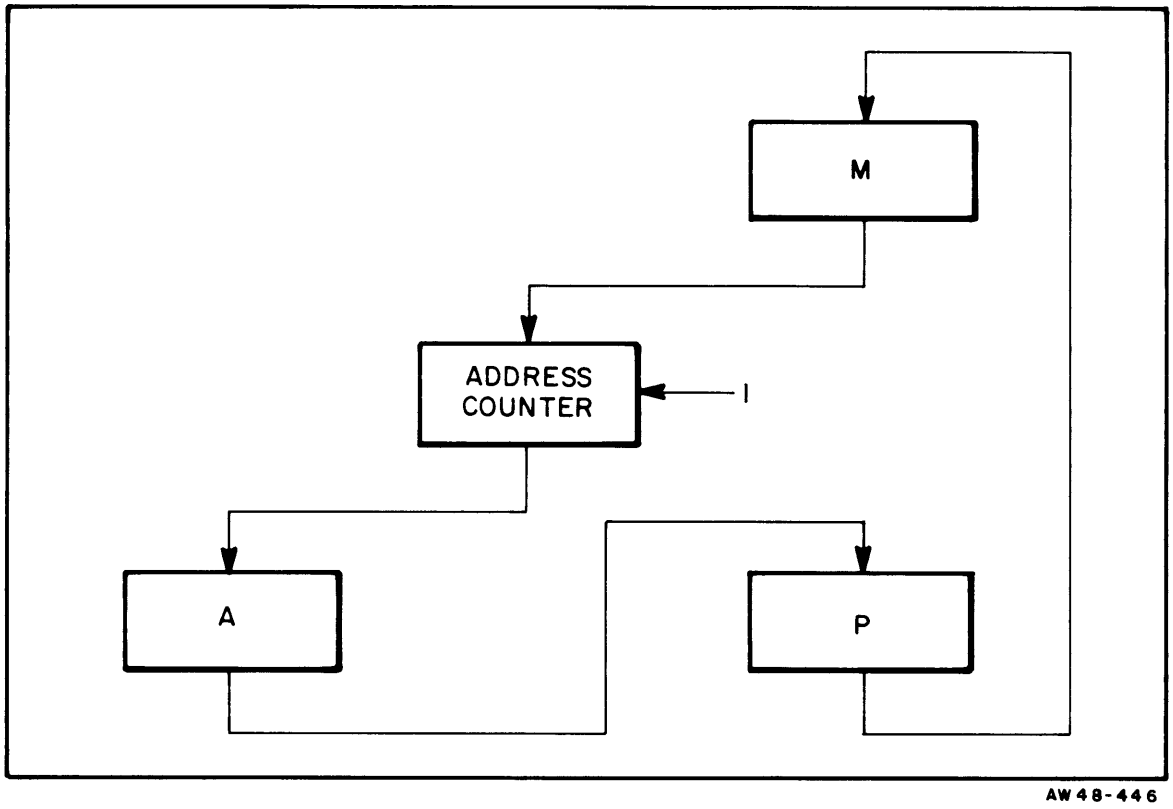


Figure 2-10. The Transfer pmiap

Table 2-1. Register Transfer Operations

Operation (Clock Pulse n)	Register Contents (Clock Pulse n+1)					
	L	E	A	P	M	T
ℓℓ, ee, mm, pp, aa, tt	ℓ	e	a	p	m	t
mim					m+1	
eℓ	e					
em					e	
ep				e		
ea			e			
et						e
eℓe	e	ℓ				
eme		m			e	
epe		p		e		
eae		a	e			
ete		t				e
te		t				
ae		a				
pe		p				
ata			t			a
apa			p	a		
apea		p	e	a		
pmp				m	p	
pmip				m+1	p	
emp				m	e	
emip				m+1	e	
empe		p		m	e	
emipe		p		m+1	e	
emia			m+1		e	
emiae		a	m+1		e	
pmap			m	a	p	
pmiap			m+1	a	p	
amipa			p	m+1	a	

2.5.3 Reading from Memory

The operation of the core memory is organized into two possible 6-microsecond cycles: read or write. During a read cycle, a word is read from core memory and placed in the E register. Within this same read cycle, the word is rewritten in core memory. Since the readout portion of the read cycle requires only 3 microseconds, the word is available at the completion of the readout for transfer to other registers.

Two basic operations are provided for reading from memory; read under L control (rdℓ) and read under M control (rdm). Both operations cause a 15-bit operand to be transferred from a specified memory location to the E register.

In read under L control, the memory address of the operand to be read is held in the low-order six bits of the L register. The remaining bits of the L register are interpreted as zeros for purposes of controlling this operation. This operation is therefore capable of reading only from memory locations 0 through 63.

In read under M control, the memory address of the operand to be read is held in the M register. Since the M register is 15 bits long, this operation is capable of reading from memory locations 0 through 32,767.

In both rdℓ and rdm, the operand address is placed in either the L or the M register by a register transfer operation. This transfer may be initiated on the same pulse that initiates the read operation, or on some earlier pulse.

In the interval following the pulse that initiates a read operation, an operand is transferred from the specified memory location to the E register. Thus, the previous contents of E are destroyed.

2.5.4 Writing to Memory

Only one word may be placed in core memory during a single write cycle. The write cycle is started with a readout portion as performed in the read cycle, with one exception: the sense windings are disconnected from the E register for the entire write cycle. The symbolic notation for this operation is cll and clm. Therefore, the selected core cell is cleared and the original contents are not placed in the E register. This allows for transfer into the E register at any time during the readout portion of the cycle for regeneration during the write portion of the cycle.

As is the case in read operations, two basic operations are provided for writing to memory: write under L control (wrℓ) and write under M control (wrm).

The memory location to be affected by the write operations is specified exactly as in the reading and clearing operations. The wrℓ operation can affect only memory locations 0 through 63, whereas the wrm operation can affect any memory location.

In both the wrℓ and wrm operations, the operand address is placed in the L or M register by a register transfer operation which is initiated on an earlier pulse. However, the operand to be written to memory may be transferred into the E register either on or before this pulse.

2.5.5 Register Shifting

Figure 2-11 shows schematically the basic shift operations which have been implemented in the AN/UYK-1. Five of these operations (sosℓ, sosr, scsℓ, snsr) involve the shifting of the A register; five (sodℓ, sodr, scdℓ, scrd, sndr) involve the shifting of the A and P registers combined; and the remaining three (sdiv, smsℓ, smuℓ) involve the shifting of the A and P registers, together with the introduction of external quantities.

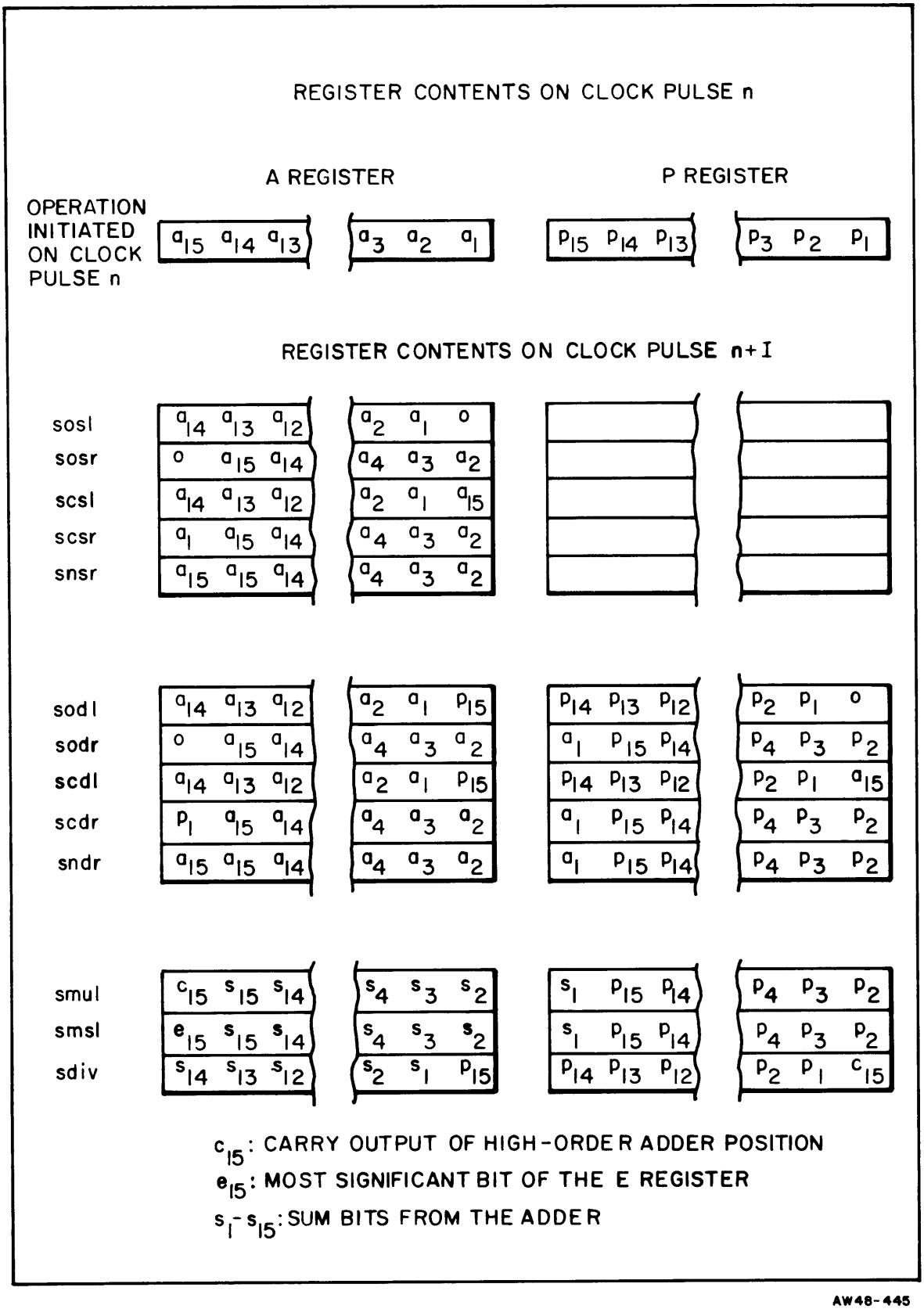


Figure 2-11. Register Shifting Operations

2.5.5.1 A REGISTER SHIFTS

Three types of shift operations are provided: open, closed, and numeric. With the open and closed types, two directions of shift are provided, left and right; the numeric shifts right only for five shift operations.

In the open-shift operations (sosℓ, sosr), the quantity in A is treated as a 15-bit logical quantity. In shifting left (sosℓ), bit i is transferred into bit position i+1, i = 1, 2, ... 14; a zero is transferred into A_1; and the bit in A_{15} is lost. In shifting right (sosr), bit i goes into bit position i-1, i = 2, 3. . . . . 15; A_{15} is filled with a zero; and the bit in A_1 is lost.

The closed-shift operations (scsℓ, scsr) are identical with their open-shift counterparts, except that the bit that would be lost in an open shift is now transferred into the bit position at the other end of the register. Thus, in shifting left (scsℓ), the bit in A_{15} is transferred to A_1; in shifting right (scsr), the bit in A_1 is transferred to A_{15}.

In the numeric-shift operation, the quantity in A is treated as a 15-bit signed twos-complement number, with the sign bit in A_{15}. In shifting right (snsr), bit i goes into bit position i-1, i = 2, 3, ... 14; the bit in A_{15} is transferred to A_{14} (i.e., the sign is propagated to the right); the bit in A_1 is lost; and the bit in A_{15} again is not changed.

2.5.5.2 A AND P REGISTER SHIFTS

In correspondence with the five operations for shifting the A register, five operations are provided for shifting the A and P registers combined. The two open-shift operations (sodℓ, sodr) shift the contents of A and P as a 30-bit logical quantity. In the closed-shift operations (scdℓ, scdr), the A and P registers are similarly coupled, and the bit shifted out of one register is reintroduced at the opposite end of the other register. The numeric-shift operation (sndr) is similar to snsr, with the P register acting as an extension of the A register.

The ten basic shift operations described above implement the Shift Open, Shift Closed, Shift Numeric, and Shift Floating commands.

2.5.5.3 A AND P REGISTER SHIFTS WITH EXTERNAL QUANTITIES

Three basic shift operations, smul, smsl, and sdiv, are provided to implement multiplication and division, respectively. The smul operation is similar to the sodr (shift open double right) operation, except for the manner in which P_{15} and A_i through A_{15} are filled: P_{15}, A_1, A_2, . . . A_{14} are filled with the sum bits from Adder positions AD_1, AD_2, ... AD_{15}, respectively; and A_{15} is filled from the carry output C_{15} of the Adder.

The smsl operation is similar to the sndr (shift numeric double right) operation. It is the same as smul except in the manner in which A_{15} is filled. For smsl, A_{15} is filled from E_{15} retaining the sign of the multiplicand.

The sdiv operation is similar to the sodl (shift open double left) operation, except for the filling of A_2, A_3, ... A_{15} and P_1: A_2, A_3, . . . A_{15} are filled with the sum digits from Adder position AD_1, AD_2, ... AD_{14}, respectively; and P_1 is filled with C_{15}.

2.5.6 Register Combination

2.5.6.1 ADDITION

The addition of two 15-bit operands is accomplished by the basic operation add. This operation causes the sum digit outputs of the adder, s_1, s_2, ... s_{15}, to be transferred to the corresponding positions of the A register, destroying the previous contents of the A register. In addition, the add operation causes the carry indicator (C) to be set to the value of c_{15}, the carry output from the high-order adder position; i.e., the carry indicator is always set or cleared as a result of an Add operation.

As described in paragraph 2.2.8, the three inputs to each adder position consist of the corresponding bit of the A register, the corresponding bit of the E register, and the carry bit from the next-least-significant adder position. For the first adder position, the carry input is taken from

the carry indicator. If, on the clock pulse that initiates the add operation, the A register contains a, the E register contains e, and the carry indicator contains c; then on the following clock pulse:

(a) The A register contains the least-significant 15 bits of the sum a + e + c, where a and e are interpreted as 15-bit positive integers.

(b) The carry indicator C contains the 16th bit of this sum.

(c) The E register is not affected.

Example 1: - (Note: The register contents are given in octal notation).

		Register Contents at Next Clock Pulse		
Clock Pulse	Operations	E	A	C
n		01234	03210	1
n+1	add		04445	0

Example 2:

		Register Contents at Next Clock Pulse		
Clock Pulse	Operations	E	A	C
n		54321	65432	0
n+1	add		41753	1

2.5.6.2 LOGICAL COMBINATION

Five basic operations are provided for forming a logical combination of register contents:

2.5.6.2.1 Extract to A (exa)

The logical product of the i^{th} bit of E and of the complement of the i^{th} bit of A is transferred to the i^{th} bit of A, $i = 1, 2, \ldots 15$. Symbolically, this operation is expressed: $(E) \cdot (A)'$ replaces (A).

Example 3:

Clock Pulse	Operations	E	A
n		01234	03210
n+1	exa		00024

2.5.6.2.2 Merge to A (mga)

The logical sum of the i^{th} bit of A and the i^{th} bit of E is transferred to the i^{th} bit of A, i = 1, 2, ... 15. Symbolically, this operation is (E) $\vee$ (A) replaces (A).

Example 4:

Clock Pulse	Operations	E	A
n		01234	03210
n+1	mga		03234

2.5.6.2.3 Extract to E (exe)

The logical product of the i^{th} bit of A and the i^{th} bit of E is transferred to the i^{th} bit of E, i = 1, 2, ... 15. Symbolically, this operation is (E) $\cdot$ (A) replaces (E).

Example 5:

Clock Pulse	Operations	E	A
n		01234	03210
n+1	exe	01210	

3.5.6.2.4 Merge to E (mge)

The logical sums of the i^{th} bit of A and the i^{th} bit of E is transferred to the i^{th} bit of E, i = 1, 2, ... 15. Symbolically, this operation is (E) $\vee$ (A) replaces (E).

Example 6:

Clock Pulse	Operations	E	A
n		01234	03210
n+1	mge	03234	

2.5.6.2.5 Extract to A and E (exae)

This operation is the combination of extract to A (exa) and extract to E (exe). Symbolically, it is (E) · $(A)'$ replaces (A) and at the same time (E) · (A) replaces (E).

Example 7:

Clock Pulse	Operations	E	A
n		01234	03210
n+1	exae	01210	00024

2.5.6.2.6 Complement A (cpa)

The complement of the i^{th} bit of A is transferred to the i^{th} bit of A, i = 1, 2, ... 15. $(A)'$ replaces (A).

Example 8:

Clock Pulse	Operations	A
n		03210
n+1	cpa	74567

2.5.7 Register and Indicator Setting

A number of basic operations are provided for setting register or indicator contents to specific binary values. These operations are:

2.5.7.1 CLEAR E REGISTER (ze)

The digit zero is transferred into each bit position of the E register.

2.5.7.2 CLEAR CARRY INDICATOR (cc)

The digit zero is transferred into the carry indicator.

2.5.7.3 SET CARRY INDICATOR (sc)

The digit one is transferred into the carry indicator.

2.5.7.4 DECREMENT L REGISTER (lil)

The low-order four-bits of the L register are decremented by unity.

2.5.7.5 SET L REGISTER ODD (odl)

The low-order bit of the L register is set to one, effecting an increment.

CHAPTER III

COMPUTER COMMAND STRUCTURE

3.1 LOGANDS

Programmed control of the AN/UYK-1 is achieved through elementary instructions called logands (for logical commands), which are stored in memory. Logands are each 15 bits in length and occupy a single location in memory. They are read from memory, interpreted in the E and L registers, and executed to carry out the operations called for. Logands may be stored anywhere in memory and are normally executed from sequential clusters.

3.2 LOGAND EXECUTION

A logand is executed in a sequence of clock pulses called a logand cycle. On each pulse of a cycle, one or more basic operations (of the type described in chapter 2) are initiated and completed in the 3-microsecond interval separating clock pulses. The number of clock pulses in a logand cycle, and the basic operation initiated on each, are determined by the coding in the logand.

The clock pulses in a logand cycle are arbitrarily numbered 1, 2, 3, etc. On clock pulse 1 of every logand cycle, the logand to be executed is held in the E register. The address of the memory location from which the logand was read is held in the M register. The remaining registers are as left by the previous logand. The initial configuration is illustrated in figure 3-1, where the letters ℓ, (m), a, p, m and t denote the clock 1 contents of the L, E, A, P, M and T registers respectively.

CLOCK	BASIC OPERATIONS	L	E	A	P	M	T
		ℓ	(m)	a	p	m	t

Figure 3-1. Initial Configuration for Logand Cycle

To illustrate the structure of a logand cycle: consider a logand whose function is to read a 15-bit operand from the memory location whose address is held in the P register, and place this operand in the A register. This particular logand requires a four-clock-pulse logand cycle. The basic operations initiated on each clock pulse and their effect on the registers are illustrated in figure 3-2.

CLOCK	BASIC OPERATIONS	L	E	A	P	M	T	REMARKS
		ℓ	(m)	a	p	m	t	
1	wrm							Regenerate logand (m) back in memory cell m
2	pmip, eℓ, rdm	(m)	(p)		m+1	p		Read from memory cell P into E
3	ea, wrm			(p)				Make specified transfer regenerate operand
4	pmip, eℓ, rdm	(p)	(m+1)	(p)	p+1	m+1	t	Read next logand from cell m+1

Figure 3-2. Illustrative Logand Cycle

On clock pulse 1, the operation wrm is initiated to write the logand in E back into memory location m; i. e., regenerate m.

On clock pulse 2, the address in P is transferred into the M register, so that the operand at this address may be read from memory. At the same time, the logand sequence address m is incremented by one (in anticipation of addressing the next logand) and transferred to P. The logand in E is transferred to L for interpretation throughout the remainder of the cycle and the contents of memory location p (the operand) are read into E.

On clock pulse 3, the operation wrm is again initiated, this time to regenerate memory location p. At the same time, the operand is transferred from E to A by the operation ea.

On clock pulse 4, the operand address in P is incremented by one, and the contents of M and P are again interchanged. The logand in L is no longer functional, so the contents of E (in this case, the operand) are transferred to L for possible use in the next logand cycle. Finally, a read operation under control of M is initiated. Since M now contains the incremented logand sequence address m + 1, the next logand in sequence is read into the E register, thus establishing the initial configuration for the next logand.

3.3 LOGAND STRUCTURE

The basic operations initiated in executing a logand are determined by the coding in the logand itself. In order to simplify the information required to do this, logands have been divided into fields. The general logand format is illustrated in figure 3-3.

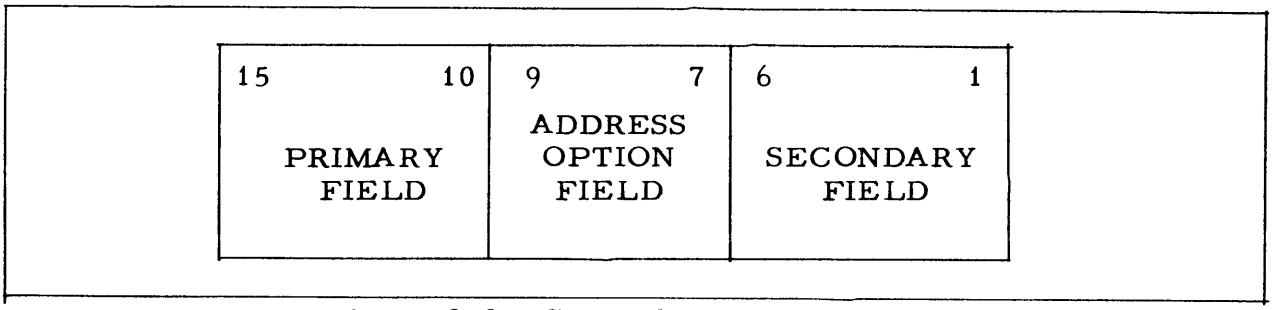


Figure 3-3. General Logand Format

The primary field (bit positions 10-15) is used to hold the primary command of the logand. Primary commands fall into two general classes: (1) regular primary commands, which have octal codes 40 through 77; and (2) special primary commands, which have octal codes 00 through 37. Each class determines the interpretation to be given to the remainder of the logand. Therefore, each class is described separately in the sections that follow. Logands containing regular primary commands are regular logands, and those containing special primary commands are special logands.

3.4 REGULAR LOGANDS

Logands containing regular primary commands have one of the two formats illustrated in figure 3-4.

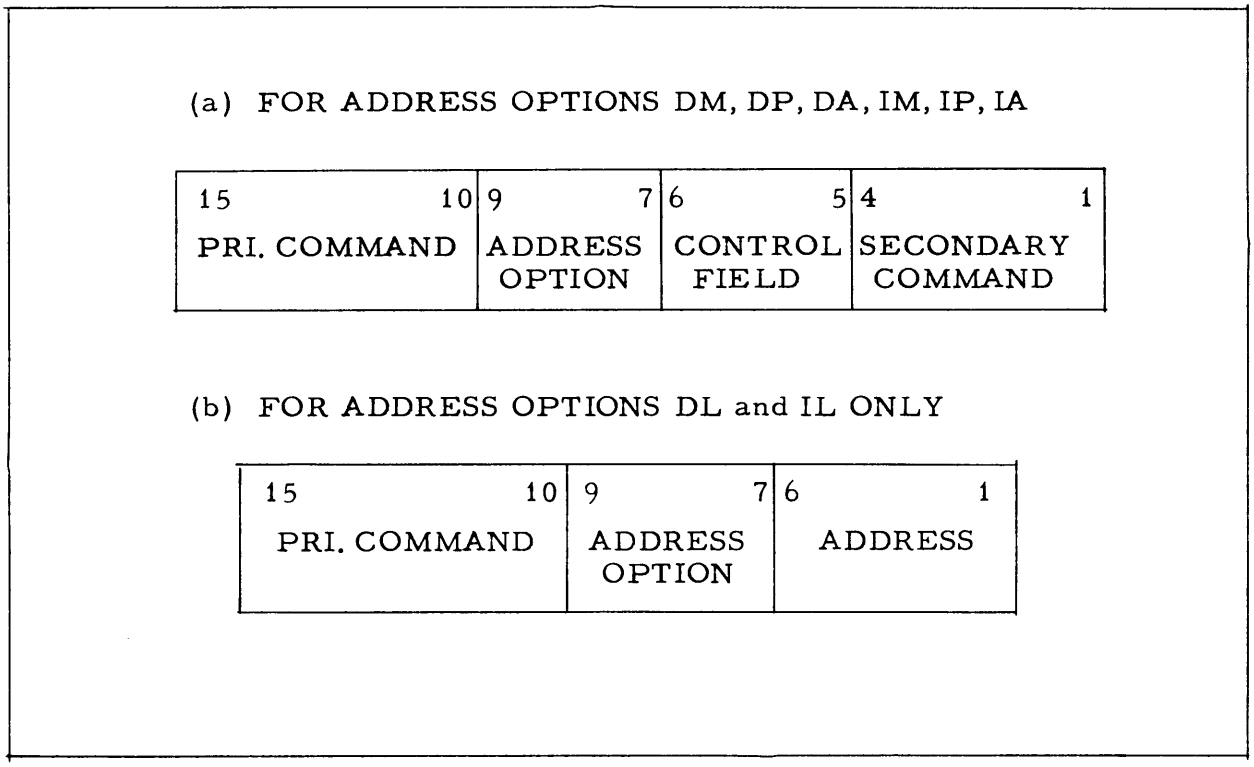


Figure 3-4. Regular Logand Formats

The elements of these formats are described in the following section.

3.4.1 Address Option

The address option (AO) is a three-bit code that determines the source of the address to be used in addressing operands in memory. Such an address is referred to as a target address. The following eight addressing options are provided:

Octal Code	Symbolic Code	Name
0	DM	Direct M
1	DL	Direct Logand
2	DP	Direct P
3	DA	Direct A
4	IM	Indirect M
5	IL	Indirect Logand
6	IP	Indirect P
7	IA	Indirect A

3.4.1.1 DIRECT LOGAND (DL) AND INDIRECT LOGAND (IL) ADDRESS OPTIONS

In the DL address option, the target address is taken from the low-order six bits of the logand being executed. This option provides a means for addressing the first 64 (0-63) locations in memory directly from the logand. The 64 locations addressable in this manner are referred to as scratchpad, since they are useful for storage of intermediate results. Scratchpad is an integral part of the 8192-word memory and is identical to the rest of the core except for its intended use.

In the IL address option, the target address is obtained from the scratchpad location called for in bits one to six of the logand. Since scratchpad words are 15 bits in length, this option may be used to address indirectly any location in memory.

Regular logands containing the DL or IL address option have format (b) in figure 3-4.

3.4.1.2 DIRECT M (DM) AND INDIRECT M (IM) ADDRESS OPTIONS

The DM address option results in the target address being taken from the M register. At the time memory is accessed, the M register contains an address which is one greater than that of the logand currently being executed. Thus, since the current logand is in location m, the target address is m + 1.

In the IM address option, the target address is the contents of the memory location immediately following the logand. Thus, if the current logand is in location m, the target address is (m + 1), where the parentheses denote "contents of".

The DM address option is used to read operands directly from, or write operands directly into, the logand sequence. The IM address option is used to read operands from, or write operands to, memory locations whose addresses have been imbedded in the logand sequence. In both options, provision is made for automatic skipping over the imbedded operand or address in executing the logand sequence.

3.4.1.3 DIRECT P (DP) AND INDIRECT P (IP) ADDRESS OPTIONS

In the DP address option, the target address is taken from the P register. Thus, if the P register contains p, an operand is read from or written into memory location p.

In the IP address option, the target address is taken from the memory location whose address is held in P. Thus, an operand is read from or written into memory location (p).

3.4.1.4 DIRECT A (DA) AND INDIRECT A (IA) ADDRESS OPTIONS

These options are analogous to DP and IP respectively, except that the target address, or the address of the target address, is taken from the A register.

Regular logands containing the DM, IM, DP, IP, DA, and IA address options have format (a) in figure 3-4.

3.4.2 Control Field

The control field is a two-bit code which has special meaning when format (a), figure 3-4, is used for a logand. Its effect is determined by the type of logand, regular or special.

3.4.2.1 REGULAR LOGAND

When a regular logand is used, the control field controls the accessing of memory and the incrementing of the target address. If the control field =0, memory is accessed, either to read an operand from memory or to write an operand to memory, and the target address is incremented. If the control field =1, memory accessing takes place but incrementing of the target address is inhibited. If the control field =2, no memory accessing takes place and the target address is incremented. If the control field =3, both accessing and incrementing are inhibited.

The control field for regular logands has been assigned the following symbolic codes:

Binary Code	Symbolic Code
00 = 0	C (Count and Access)
01 = 1	H (Access and Hold Count)
10 = 2	N (Count and No Access)
11 = 3	B (Both - Hold Count and No Access)

3.4.2.2 SPECIAL LOGANDS

When the special logands for branching and shifting are used, the control field has a different function.

It provides a means of stopping the computer on a branch-type command when the computer is in the manual mode. This would require that the control field = 2; i. e., bit six is a one. This condition has been assigned a symbolic code F. During normal operation, bit 6 of the control field is zero, or symbolically, blank.

In the shift-type operations, the control field contains the coding of the direction of a shift and the register or registers involved. The following table gives the necessary information:

Binary Code	Symbolic Code	Function
00 = 0	SR	A right (Single)
01 = 1	SL	A left (Single)
10 = 2	DR	A and P right (Double)
11 = 3	DL	A and P left (Double)

3.4.3 Primary Commands

The primary command (PC) in a regular logand serves two purposes: (a) to select, in conjunction with the address option and the control field, a set of implicit basic operations to be carried out in the logand cycle; and (b) to select one or more explicit basic operations to be initiated on one or more of the clock pulses of the cycle. These terms are defined in the sections that follow.

3.4.3.1 SELECTION OF IMPLICIT OPERATION

Implicit operations form the skeleton of the logand cycle. The selection of a set of implicit operations for a particular logand cycle is based on three factors: (1) the address option; (2) the control field; and (3) the primary command type, of which there are two: load and store. Load-type primary commands (octal codes 44-47, 54-77) cause an operand to be read from memory into the E register during the operand access portion of the cycle; store-type primary commands (octal codes 40-43, 50-53) cause the content of E to be written into memory. In both types, the target address to be used in accessing memory and the counting of the target address are determined by the address option and the control field, respectively.

Since there are eight address options, two count options (except for DL, IL), two access options (except for DL, IL), and two primary command types, a total of 52 sets of implicit basic operations are used for regular

logand cycles. These sets are defined in table 3-1, which shows the basic operations comprising each set, and the clock pulses on which they are initiated. The following conclusions may be drawn from the table.

(1) The use of a direct addressing option (DL, DM, DP and DA) results in a four-clock-pulse cycle, and therefore requires 12 microseconds. The indirect address options (IL, IM, IP and IA) require six clock pulses or 18 microseconds.

(2) The first clock pulse of every cycle is used to regenerate, by means of the wrm operation, the logand read from memory on the last clock pulse of the preceding cycle. The logand is transferred to L on clock pulse 2 by the operation el or ele. On the last clock pulse, the contents of E are saved in L, and the next logand is read into E under M control.

(3) In the direct address option cycles, the loading and storing of operands is accomplished on clock pulses 2 and 3. For load-type commands, the sequences rdm, wrm, and rdl, wrl are used; for store-type commands, the sequences clm, wrm, and cll, wrl are used

For count options in which the control field = 1 or 3, the incrementing of the address on clock pulse 4 is inhibited; when control field = 2 or 3, the memory access operations at clock pulses 2 and 3 are inhibited.

(4) In the indirect address option cycles, clock pulses 2 and 3 are used for reading the target address into the E register. Clock pulses 4 and 5 of such cycles are then used to load or store operands at this address for the respective type primary commands.

For the control field = 1 or 3, the incrementing of the target address on clock pulse 6 is inhibited; for control field = 2 or 3, the memory access operations on clock pulses 4 and 5 are inhibited.

Table 3-1. Implicit Operations for Regular Logand Cycles

Address Option / PC Type	Clock	Control Field Coding (Bits Five and Six)			
		C = 00	H = 01	N = 10	B = 11
DL Load	1	wrm			
	2	mim, el, rdl			
	3	wrl			
	4	el, rdm			
DL Store	1	wrm			
	2	mim, ele, cll			
	3	wrl			
	4	el, rdm			
DM Load	1	wrm	wrm	wrm	wrm
	2	mim, el, rdm	mim, el, rdm	mim, ele	mim, ele
	3	wrm	wrm		
	4	mim, el, rdm	el, rdm	mim, el, rdm	el, rdm
DM Store	1	wrm	wrm	wrm	wrm
	2	mim, ele, clm	mim, ele, clm	mim, ele	mim, ele
	3	wrm	wrm		
	4	mim, el, rdm	el, rdm	mim, el, rdm	el, rdm

Table 3-1. Implicit Operations for Regular Logand Cycles (Cont.)

Address Option / PC Type	Clock	Control Field Coding (Bits Five and Six)			
		C = 00	H = 01	N = 10	B = 11
	1	wrm	wrm	wrm	wrm
DP	2	pmip, el, rdm	pmip, el, rdm	pmip, ele	pmip, ele
Load	3	wrm	wrm		
	4	pmip, el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm
	1	wrm	wrm	wrm	wrm
DP	2	pmip, ele, clm	pmip, ele, clm	pmip, ele	pmip, ele
Store	3	wrm	wrm		
	4	pmip, el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm
	1	wrm	wrm	wrm	wrm
DA	2	amipa, el, rdm	amipa, el, rdm	amipa, ele	amipa, ele
Load	3	wrm	wrm		
	4	pmip , el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm
	1	wrm	wrm	wrm	wrm
DA	2	amipa, ele, clm	amipa, ele, clm	amipa, ele	amipa, ele
Store	3	wrm	wrm		
	4	pmip, el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm

Table 3-1. Implicit Operations for Regular Logand Cycles (Cont.)

Address Option / PC Type	Clock	Control Field Coding (Bits Five and Six)			
		C = 00	H = 01	N = 10	B = 11
	1	wrm			
IL	2	mim, el, rdl			
Load	3	wrl			
	4	emp, rdm			
	5	wrm			
	6	pmip, el, rdm			
	1	wrm			
IL	2	mim, el, rdl			
Store	3	wrl			
	4	empe, clm			
	5	wrm			
	6	pmip, el, rdm			
	1	wrm	wrm	wrm	wrm
IM	2	mim, el, rdm	mim, el, rdm	mim, el, rdm	mim, el, rdm
Load	3	wrm	wrm	wrm	wrm
	4	emip, rdm	emip, rdm	zemip	zemip
	5	wrm	wrm		
	6	pmip, el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm

Address Option / PC Type	Clock	Control Field Coding (Bits Five and Six)			
		C = 00	H = 01	N = 10	B = 11
	1	wrm	wrm	wrm	wrm
IM	2	mim, el, rdm	mim, el, rdm	mim, el, rdm	mim, el, rdm
Store	3	wrm	wrm	wrm	wrm
	4	emipe, clm	emipe, clm	emipe	emipe
	5	wrm	wrm		
	6	pmip, el, rdm	pmp, el, rdm	pmip, el, rdm	pmp, el, rdm
	1	wrm	wrm	wrm	wrm
IP	2	pmip, el, rdm	pmip, el, rdm	pmip, el, rdm	pmip, el, rdm
Load	3	wrm	wrm	wrm	wrm
	4	emia, rdm	emia, rdm	zemia	zemia
	5	wrm	wrm		
	6	pmiap, el, rdm	pmap, el, rdm	pmiap, el, rdm	pmap, el, rdm
	1	wrm	wrm	wrm	wrm
IP	2	pmip, el, rdm	pmip, el, rdm	pmip, el, rdm	pmip, el, rdm
Store	3	wrm	wrm	wrm	wrm
	4	emiae, clm	emiae, clm	emiae	emiae
	5	wrm	wrm		
	6	pmiap, el, rdm	pmap, el, rdm	pmiap, el, rdm	pmap, el, rdm

Table 3-1. Implicit Operations for Regular Logand Cycles (Cont.)

Address Option / PC Type	Clock	Control Field Coding (Bits Five and Six)			
		C = 00	H = 01	N = 10	B = 11
	1	wrm	wrm	wrm	wrm
IA	2	amipa, el, rdm	amipa, el, rdm	amipa, el, rdm	amipa, el, rdm
Load	3	wrm	wrm	wrm	wrm
	4	emia, rdm	emia, rdm	zemia	zemia
	5	wrm	wrm		
	6	pmiap, el, rdm	pmap, el, rdm	pmiap, el, rdm	pmap, el, rdm
	1	wrm	wrm	wrm	wrm
IA	2	amipa, el, rdm	amipa, el, rdm	amipa, el, rdm	amipa, el, rdm
Store	3	wrm	wrm	wrm	wrm
	4	emiae, clm	emiae, clm	emiae	emiae
	5	wrm	wrm		
	6	pmiap, el, rdm	pmap, el, rdm	pmiap, el, rdm	pmap, el, rdm

3.4.3.2 SELECTION OF EXPLICIT OPERATIONS

In addition to a set of implicit operations, the primary command also selects one or more explicit basic operations. These operations are usually initiated on clock pulse 3 of a direct address option cycle, or on clock pulse 5 of an indirect address option cycle. Unlike the selection of implicit operations, which is based on the primary command type, the selection of explicit operations is determined by the particular primary command.

Table 3-2 lists the regular primary commands, by type. Each command is defined by the explicit basic operation it selects, and the clock pulse or pulses on which the operation is initiated. Definitions of the basic operations are given in Chapter 2.

Address options IA and IP select the implicit operations ma or mia at clock 6. If an explicit secondary operation ea, mga or exa is selected, the explicit operation inhibits the ma or mia transfer. For example, if the explicit operation ea is selected, the resulting combined register transfer becomes eapm.

Table 3-2. Regular Primary Command List

A. Load-Type Commands			Direct Address Option		Indirect Address Option	
Octal Code	Sym Code	Name	Clock 3	Clock 4	Clock 5	Clock 6
60	NO	No Operation	ee		ee	
75	LA	Load A	ea		ea	
62	LP	Load P	ep		ep	
70	LT	Load T	et		et	
72	LM	Load M	em		em	
55	RA	Replace A	eae		eae	
47	RP	Replace P	epe		epe	
45	RT	Replace T	ete		ete	
57	RM	Replace M	eme		eme	
66	AP	Exchange A&P	apa		apa	
64	AT	Exchange A&T	ata		ata	
44	ZE	Clear E	ze		ze	
76	XA	Extract to A	exa		exa	
74	MA	Merge to A	mga		mga	
46	XE	Extract to E	exe		exe	
54	ME	Merge to E	mge		mge	
56	DX	Double Extract	exae		exae	
73*	AS	Add Single	cc	add**	cc	add**
63*	AL	Add Least	cc	add	cc	add
61*	AI	Add Intermediate		add		add
71*	AM	Add Most		add**		add**
65	CS	Complement Set	cpa, sc		cpa, sc	
67	CC	Complement Clear	cpa, cc		cpa, cc	
77	CH	Complement Hold	cpa		cpa	

*secondary command is ignored

**overflow indicator set to 1 if overflow

B. Store-Type Commands			Direct Address Option		Indirect Address Option	
Octal Code	Sym Code	Name	Clock 3	Clock 4	Clock 5	Clock 6
52	SE	Store E	ee		ee	
50	SA	Store A	ae		ae	
42	SP	Store P	pe		pe	
40	ST	Store T	te		te	
51	HA	Hold A	eae		eae	
43	HP	Hold P	epe		epe	
41	HT	Hold T	ete		ete	
53	HM	Hold M	eme		eme	

3.4.4 Secondary Commands

If the DL or IL address option is used in a regular logand, the secondary field of the logand (bit positions one to six)are interpreted as a scratchpad address. If any other address option is used (DM, DP, DA, IM, IP or IA), bit positions one to four of the logand are generally used for a secondary command. Like the primary command, the secondary command selects one or more explicit basic operations. These operations are generally initiated on the clock pulse following the explicit operation selected by the primary command; i. e., on clock pulse 4 of a direct address option cycle or on clock pulse 6 of the indirect address option cycle.

Table 3-3 lists the secondary commands. Each command is defined by the explicit basic operation it selects, and the clock pulse or pulses on which these operations are initiated.

3.4.5 Construction of Regular Logand Cycles

The effect of the coding in a particular logand may be determined by a clock-by-clock listing of the basic operations, both implicit and explicit, which the coding selects. To illustrate, consider the following logand:

Primary Command:	LT (Load T)
Address Option:	DP (Direct P)
Control Field:	C (Normal access and count)
Secondary Command:	CC (Complement Clear)

The primary command LT is a load-type command. A load-type primary command, together with the address option DP and the control field C, selects the following set of implicit operations (from table 3-1):

Clock	Basic Operations
1	wrm
2	pmip, eℓ, rdm
3	wrm
4	pmip, eℓ, rdm

Table 3-3. Secondary Command List

A. Load-Type Commands			Direct Address Option		Indirect Address Option	
Octal Code	Sym Code	Name	Clock 3	Clock 4	Clock 5	Clock 6
00	NO	No Operation				
15	LA	Load A		ea		ea
02	LP	Load P		ep		ep
10	LT	Load T		et		et
12	LM	Load M		em		em
06	AP	Exchange A&P		apa		apa
04	AT	Exchange A&T		ata		ata
16	XA	Extract to A		exa		exa
14	MA	Merge to A		mga		mga
13	AS	Add Single	cc	add*	cc	add*
03	AL	Add Least	cc	add	cc	add
01	AI	Add Intermediate		add		add
11	AM	Add Most		add*		add*
05	CS	Complement Set		cpa, sc		cpa, sc
07	CC	Complement Clear		cpa, cc		cpa, cc
17	CH	Complement Hold		cpa		cpa

*Overflow indicator set to 1 if overflow

Now, the primary command LT, according to table 3-2, selects the explicit operation et for initiation on clock pulse 3. The secondary command CC, by table 3-3, selects the operations cpa, cc, for initiation on clock pulse 4. The complete set of operations is then as follows:

Clock	Implicit Operation	Explicit Operation
1	wrm	
2	pmip, el, rdm	
3	wrm	et
4	pmip, el, rdm	cpa, cc

The effect of these operations is shown below:

Table 3-1. Implicit Operations for Regular Logand Cycles (Cont.)

Note: Where inputs to a register are not indicated, the contents of that register are assumed to be "held."

Clock	Basic Operations		L	E	A	P	M	T
	Implicit	Explicit	ℓ	(m)	a	p	m	t
1	wrm							
2	pmip, eℓ, rdm		(m)	(p)		m+1	p	
3	wrm	et						(p)
4	pmip, eℓ, rdm	cpa, cc	(p)	(m+1)	a′	p+1	m+1	

The net result of the logand is to load the operand (p) into the L and T registers; increment the operand address p in P; complement the quantity in A; clear the carry indicator; increment the logand sequence address in M; and read the next logand into E.

Since a regular logand cycle is composed of both implicit and explicit operations, the possibility exists that a conflict will arise between an implicit and an explicit operation. These conflicts are described below.

(a) Restriction on use of Primary Command LM, RM, and HM

For the DM, DP, and DA address options with the control field = C or H, the clock 3 implicit operation is wrm. Similarly, for all the indirect address options (with the control field = C or H), the clock pulse 5 implicit operation is wrm. If a transfer into the M register is attempted on the same clock pulse on which the wrm operation is initiated, the results will be unpredictable. For this reason, the primary commands LM, RM, and HM should be used only with the address option DL, or when the control field is N or B for the other address options.

(b) Restriction on use of Secondary Commands LP, LM and AP

Address options DP, DA, and IM select the implicit operation pmip (or pmp), and address options IP, IA select the implicit pmiap (or pmap) on clock pulse 4 for the direct options and on clock pulse 6 for the indirect options. If a transfer into P is attempted on these respective clock pulses

by means of the secondary commands LP (Load P) or AP (Exchange A and P), the P register receives the logical sum (OR) of the two inputs.

The LM is allowed as a secondary command only if the DM (with the control field = H or B) addressing option is used.

(c) Restriction on the use of Primary Command CS with Secondary Commands AS or AL

The AS (add single) or AL (add least) commands are unique in that the explicit operations they select are initiated on two successive clock pulses.

If AS or AL is used as a secondary command, and the command CS (complement and set carry indicator) is used as a primary command, the explicit cc (clear carry indicator) selected by the former conflicts with the explicit operation sc (set carry indicator) selected by the latter. Since the result is unpredictable, this combination should not be used.

(d) Restriction on the use of any Secondary Command with Primary Commands AS, AL, AI or AM

The explicit operation add, which is selected by the primary commands AS, AL, AI, and AM, is deferred until the last clock pulse of the cycle in order to allow time for the adder output to become stable. For reasons associated with this fact, any explicit operation which would normally result from a secondary command cannot be initiated. The coding of the secondary command, for this case, is therefore immaterial.

3.5 SPECIAL LOGANDS

The format of logands containing special primary commands is, in general, different for each special primary command. The special primary commands are listed in table 3-4, and are described in the sections that follow.

Table 3-4. Special Primary Command List

Octal Code	Symbolic Code	Primary Command
20	BR	Branch
24	SK	Skip
11	SO	Shift Open
03	SC	Shift Closed
13	NR	Numeric Right
01	FL	Float Left
07	RC	Repeat Count
15	MP	Multiply
17	MS	Multiply Signed
05	DV	Divide
32	MV	Move
36	TB	Table Search
30	MH	Match
34	SR	Sort
25	CF	Control Function
21	EF	External Function
27	WI	Word Input
23	WO	Word Output
37	BI	Block Input
33	BO	Block Output
00	IT	Interrupt
10	TM	Terminate Interrupt

3.5.1 Conditions for Conditional Special Logands

Logands containing the special primary commands BR (branch), SK (skip), TB (table search), MH (match), MV (move), and SR (sort) are called conditional logands since their outcome (in general) is dependent on the contents of certain registers and indicators. Bit positions one to five of these logands are used to specify one of a number of conditions to be tested for in determining the outcome of the logand. These conditions are summarized in table 4-21, and described below.

Unconditional (UN): This condition is always satisfied. It may be used, for example, with the BR command to cause an unconditional jump.

Never (NV): This condition is never satisfied. It may be used, for example, in BI and BO logands to insure that the logands are terminated by an implied condition.

A Odd (AD): Satisfied if the low-order bit in A is one.

A Negative (AN): Satisfied if the high-order bit in A is one.

A Positive (AP): Satisfied if the high-order bit in A is zero.

A Zero (AZ): Satisfied if every bit in A is zero.

Carry Indicator On (CY): Satisfied if the carry indicator contains one.

Overflow Indicator On (OV): Satisfied if the overflow indicator contains one. The overflow is reset to zero after testing.

Memory Parity Error Indicator On (PY): Satisfied if the parity error indicator contains one.

E Negative (EN): Satisfied if the high-order bit in E is a one.

E and A Equal (EQ): Satisfied if the number in A is the one's complement of the number in E, and if the carry indicator is zero.

E and A Not Equal (NQ): Satisfied if the number in A is not the one's complement of the number in E, and if the carry indicator is zero.

Numeric High (NH): Satisfied if the one's complement of the number in the A register is algebraically greater than or is equal to the number in the E register, and if the carry indicator is zero. If the carry indicator is one, it is satisfied only if $(A)' > (E)$.

Numeric Low (NL): Satisfied if the one's complement of the number in A is algebraically less than the number in the E register and the carry indicator is zero. If the carry indicator is one, it is satisfied if $(A)' \leqslant (E)$.

Configuration	Satisfies
$a_{15} = e_{15} = 0$	NL
$a_{15} = e_{15} = 1$	NH
$a_{15} \neq e_{15}$ and carry out of $AD_{15} = 1$	NL
$a_{15} \neq e_{15}$ and carry-out of $AD_{15} = 0$	NH

T Register in Use (TL): Satisfied if the T register is being used by the input/output unit.

3.5.2 Branch (BR)

The BR command provides a means for transferring control to the logand in an arbitrary memory location if a specified condition is met; or if this condition is not met, of continuing with the next logand in sequence. Logands containing the BR command have the following format:

15 10	9 7	6	5 1
BR	AO	FLG STOP	CONDITION

Address options DM, DP, DA, IM, IP, or IA may be used with the BR command. The effect of each is as follows:

If a direct address option (DM, DP, or DA) is used, the branch address (i. e., the address of the logand to which control is to be transferred if the condition is met) is read from the location specified by the address option. With the DM option, the branch address is read from location m+ 1, where m is the location of the BR logand. With the DP and DA options, the branch address is read from locations p and a, respectively, the initial contents of the P and A registers. Only test conditions not involving the E register should be used with the direct address options. When the test for a specified condition is made, the A register contains a, the initial contents of A.

If an indirect address option (IM, IP, IA) is used, two memory accesses are made: the first, to read a "comparand" for use in testing the specified condition, the second to read the branch address. In the IM option, the comparand is read from location m + 1 and the branch address from location m + 2, where m is the location of the BR logand. In the IP and IA options, the comparand is read from location p or a, respectively, where p or a are the initial contents of P and A; and the branch address is read from location m + 1. Any of the test conditions are allowable with the indirect address option. When the test for the specified condition is made, the A register contains a, the initial contents of A; and the E register contains the comparand read from memory.

Bit six of the BR logand holds a flag code. If bit six is one (symbolic F), and the computer is running in the flag search mode, (see description of control panel), the computer stops after executing the BR logand. If bit six is zero, or if the computer is not running in the flag search mode, the computer proceeds.

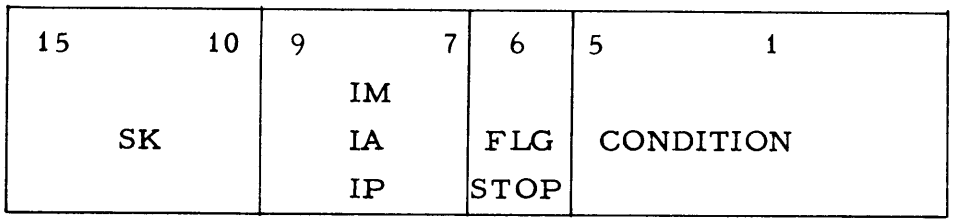
The BR logand cycles for the permissible address option are shown in table 3-5. With the direct address option, the BR logand requires 12 microseconds; with the indirect address option, it requires 18 microseconds.

Example: The following (equivalent) logand sequences compare the contents of memory location p to the constant 10, and branch to Y if (p) < 10:

Location	Contents	Location	Contents
X :	LA/DP/C/CC	X :	LA/DM/C/CS
X + 1 :	BR/IM/ /NL	X + 1 :	00012_8
X + 2 :	00012_8	X + 2 :	BR/IP/ /NH
X + 3 :	Y	X + 3 :	Y

3.5.3 Skip (SK)

The SK command provides a means for conditionally skipping a logand in the logand sequence being executed. Logands containing the SK command have the following format:



The SK command may be used with address options IM, IP, and IA only. In each option, a "comparand" is read from memory for use in testing the specified condition. In the IM option, the comparand is read from location m + 1, where m is the location of the SK logand. If the condition is satisfied, the next logand is taken from location m + 3; if the condition is not satisfied, the next logand is taken from m + 2. In the IP and IA options, the comparand is read from location p or a, respectively, where p and a are the initial contents of P and A. If the condition is satisfied, the next logand is taken from location m + 2; otherwise, it is taken from m + 1.

The flag code in position six of the SK logand has the same effect as in the BR logand.

Table 3-5. BR Logand Cycles

Address Option	Clock	Operations	L	E	A	P	M	T	
DM	1	wrm	ℓ	(m)	a	p	m	t	
	2	mim, eℓ, rdm	(m)	(m+1)			m+1		
	3	wrm, test							
	4	mim, eℓ, rdm	(m+1)	(m+2)			m+2		Not Satisfied
		em, eℓ, rdm	(m+1)	((m+1))			(m+1)		Satisfied
DP	1	wrm	ℓ	(m)	a	p	m	t	
	2	pmip, eℓ, rdm	(m)	(p)		m+1	p		
	3	wrm, test							
	4	pmip, eℓ, rdm	(p)	(m+1)		p+1	m+1		Not Satisfied
		emip, eℓ, rdm	(p)	((p))		p+1	(p)		Satisfied
DA	1	wrm	ℓ	(m)	a	p	m	t	
	2	amipa, eℓ, rdm	(m)	(a)	p	m+1	a		
	3	wrm, test							
	4	pmip, eℓ, rdm	(a)	(m+1)		a+1	m+1		Not Satisfied
		emip, eℓ, rdm	(a)	((a))		a+1	(a)		Satisfied
IM	1	wrm	ℓ	(m)	a	p	m	t	
	2	mim, eℓ, rdm	(m)	(m+1)			m+1		
	3	wrm, test							
	4	mim, rdm		(m+2)			m+2		
	5	wrm							
	6	mim, eℓ, rdm	(m+2)	(m+3)			m+3		Not Satisfied
		em, eℓ, rdm	(m+2)	((m+2))			(m+2)		Satisfied
IP	1	wrm	ℓ	(m)	a	p	m	t	
	2	pmip, eℓ, rdm	(m)	(p)		m+1	p		
	3	wrm, test							
	4	pmip, rdm		(m+1)		p+1	m+1		
	5	wrm							
	6	mim, eℓ, rdm	(m+1)	(m+2)			m+2		Not Satisfied
		em, eℓ, rdm	(m+1)	((m+1))			(m+1)		Satisfied

Table 3-5. BR Logand Cycles (Cont.)

Address Option	Clock	Operations	L	E	A	P	M	T	
	1	wrm	ℓ	(m)	a	p	m	t	
	2	amipa, eℓ, rdm	(m)	(a)	p	m+1	a		
	3	wrm, test							
IA	4	pmip, rdm		(m+1)		a+1	m+1		
	5	wrm							
	6	mim, eℓ, rdm	(m+1)	(m+2)			m+2		Not Satisfied
		em, eℓ, rdm	(m+1)	((m+1))			(m+1)		Satisfied

test: test for specified condition

Any test condition may be specified in the SK logand. At the time the test is made, the A register contains a, the initial contents of A; and the E register contains the comparand read from memory.

The SK logand cycles for the permissible address options are shown in table 3-6. The logand requires 18 microseconds.

Example: The following logand sequence places the positive absolute value of the signed operand at location p in the A register:

Loc. Logand	(p) is +						(p) is -					
	L	E	A	P	M	T	L	E	A	P	M	T
x: ZE/DM/B/LA	0	(x+1)	0	p	x+1	t	0	(x+1)	0	p	x+1	t
x+1: SK/IP/ / NL	(p)	(x+3)	0	p+1	x+3	t	(p)	(x+2)	0	p+1	x+2	t
x+2: HA/DM/B/CS							0	(x+3)	(p)	p+1	x+3	t
x+3: AM/DM/B/NO	(p)	(x+4)	(p)	p+1	x+4	t	0	(x+4)	-(p)	p+1	x+4	t

Table 3-6. SK Logand Cycles

Address Option	Clock	Operations	L	E	A	P	M	T	
IM	1	wrm	ℓ	(m)	a	p	m	t	
	2	mim, eℓ, rdm	(m)	(m+ 1)			m+ 1		
	3	wrm							
	4	mim, test					m+ 2		
	5								
	6	eℓ, rdm	(m+ 1)	(m+ 2)			m+ 2		Not Satisfied
		mim, eℓ, rdm	(m+ 1)	(m+ 3)			m+3		Satisfied
IP	1	wrm	ℓ	(m)	a	p	m	t	
	2	pmip, eℓ, rdm	(m)	(p)		m+1	p		
	3	wrm							
	4	pmip, test				p+ 1	m+ 1		
	5								
	6	eℓ, rdm	(p)	(m+ 1)			m+ 1		Not Satisfied
		mim, eℓ, rdm	(p)	(m+ 2)			m+ 2		Satisfied
IA	1	wrm	ℓ	(m)	a	p	m	t	
	2	amipa, eℓ, rdm	(m)	(a)	p	m+1	a		
	3	wrm							
	4	pmip, test				a+ 1	m+1		
	5								
	6	eℓ, rdm	(a)	(m+ 1)			m+1		Not Satisfied
		mim, eℓ, rdm	(a)	(m+2)			m+ 2		Satisfied

3.5.4 Shift Open (SO)

The command causes the contents of the A register, or the A and P registers combined, to be shifted a specified number of places to the left or right.

Logands containing the SO command have the following format:

15 10	9 7	6 5	4 1
SO	DM	Control Field	No. of Shifts

Control field: SR = 00 = A right
SL = 01 = A left
DR = 10 = A and P right; 30 bit register
DL = 11 = A and P left; 30 bit register

With the SR control field option, bits shifted out of A_1 are lost, and zeros are introduced into A_{15}. With the SL option, bits shifted out of A_{15} are lost, and zeros are introduced into A_1.

With the DR control field option, bits shifted out of P_1 are lost; bits from A_1 enter P_{15}; and zeros are entered in A_{15}. With the DL option, bits shifted out of A_{15} are lost; bits from P_{15} go into A_1; and zeros enter P_1.

Bits 1-4 of the logand hold the number (n) of places to be shifted, $0 \leq n \leq 15$. If n = 0, no shifting results.

The SO logand cycles are shown in table 3-7. The single-position shift operation initiated on clock 3a is determined by the control field coding as follows:

Control Field	Shift Operation
SR	sosr
SL	sosℓ
DR	sodr
DL	sodℓ

The SO logand requires 12 + 3n microseconds, where n is the number of places shifted.

Table 3-7. SO Logand Cycles

Address Option	Clock	Operations	L	E	A	P	M	T
	1	wrm	ℓ	(m)	a	p	m	t
	2	mim, eℓe	(m)	ℓ	a	p	m+1	t
	3	ℓiℓ, test			as			
DM	3a	ℓiℓ, test			shifted			
		sosℓ/sosr/sodℓ/sodr						
	.	.						
	.	.						
	3a	ℓiℓ, test						
		sosℓ/sosr/sodℓ/sodr						
	4	eℓ, rdm	ℓ	(m+1)			m+1	t

3.5.5 Shift Closed (SC)

This command is identical to the SO (shift open) with the following exceptions:

(a) In a single-length shift (Control Field = SR or SL), bits shifted out of one end of the A register are returned to the opposite end of the register. With a control field of SL (A left), bits shifted out of A_{15} are entered into A_1; with an SR option, bits shifted out of A_1 are entered into A_{15}.

(b) In a double length shift left (Control Field = DL), bits shifted out of A_{15} are entered into P_1. In a double length shift right (Control Field = DR), bits shifted out of P_1 are entered into A_{15}.

The format of the SC logand is the same as that for the SO logand. They differ only in the single-position shift operation (table 3-7) initiated on clock 3a:

CF	Shift Operation
SR	scsr
SL	scsℓ
DR	scdr
DL	scdℓ

The SC logand requires 12 + 3n microseconds, where n is the number of places shifted.

3.5.6 Numeric Right Shift (NR)

This command causes the contents of the A register, or the A and P registers combined, to be shifted a specified number of places to the right. The command is identical to the SO (Control Field = SR or DR) command, except that the bit in A_{15} is propagated throughout the shift in order to preserve the numerical format of the quantity being shifted.

The format of the NR logand is the same as that of the SO logand. The NR logand cycle is identical to the SO cycle (table 3-7) except for the single-position shift operation on clock 3a:

Control Field	Shift Operation
SR	snsr
DR	sndr

The NR logand requires 12 + 3n microseconds, where n is the number of places shifted.

3.5.7 Float Left (FL)

This command causes the contents of the A register, or the A and P registers combined, to be shifted left until a specified number of positions have been shifted; or until the bits in A register positions 14 and 15 are

different from each other, indicating that the number in A has been "normalized". The format of the FL logand, and the type of shifting employed, is identical to that of the SO (Control Field = SL or DL) logand.

At the end of the FL logand cycle, positions 1-15 of the L register hold; eg, FL/DM/SL/n - k, where n is the number of shift positions specified in positions 1-4 of the logand, and k is the number of positions actually shifted. If n = 0 or if the quantity in A is normalized at the outset, no shifting will occur.

The FL logand cycle is shown in table 3-8. The single-position shift operation initiated on clock 3a is determined by the control field code, as follows:

Control Field	Shift Operation
SL	sosℓ
DL	sodℓ

The FL logand requires 12 + 3k microseconds, where k is the number of places actually shifted.

Table 3-8. FL Logand Cycles

Address Option	Clock	Operations	L	E	A	P	M	T
DM	1	wrm	ℓ	(m)	a	p	m	t
	2	mim, eℓe	(m)	ℓ			m+1	
	3a	test ℓe, ℓiℓ, sosℓ/sodℓ	(m)-1	(m)	as shifted			
	.							
	.							
	3a	test ℓe, ℓiℓ, sosℓ/sodℓ	(m)-k	(m)-k-1				
	3a	test, ℓe, ℓiℓ	(m)-k-1	(m)-k				
	4	eℓ, rdm	(m)-k	(m+1)				

3.5.8 Repeat Count (RC)

The RC logand is used to increment the contents of the A, P, or M register by a specified number. The RC logand has the following format:

15 10	9 7	6 5	4 1
RC	AO	SR	n

Address options DM, DA, or DP may be used with this command. The contents of M, A, or P are incremented by unity n times, respectively. The contents of the M register at clock 3a are counted by the command mim n times under control of bits one-four of the L register.

The control field must be SR = 00 at all times for the RC, or shifting of the A register may take place. The logand cycles for the RC logand are shown in table 3-9.

The RC logand requires 12+3n microseconds, where n is the number of counts.

Table 3-9. RC Logand Cycles

Address Option	Clock	Operations	L	E	A	P	M	T
	1	wrm	ℓ	(m)	a	p	m	t
	2	mim, ℓe	(m)	ℓ			m+1	
DM	3	test, $\ell i\ell$	(m)-1				m+1	
	3a	test, $\ell i\ell$, mim	(m)-2				m+2	
	.							
	3a	test, $\ell i\ell$, mim	(m)-n-1				m+n+1	
	4	eℓ, rdm	ℓ	(m+n+1)	a	p	m+n+1	t

Table 3-9. RC Logand Cycles (Cont.)

Address Option	Clock	Operations	L	E	A	P	M	T
DP	1	wrm	ℓ	(m)	a	p	m	t
	2	pmip, $e\ell$ e	(m)	ℓ		m+1	p	
	3	test, $\ell i\ell$	(m)-1				p	
	3a	test, $\ell i\ell$, mim	(m)-2				p+1	
	.							
	3a	test, $\ell i\ell$, mim	(m)-n-1		a		p+n.	
	4	pmp, $e\ell$, rdm	ℓ	(m+1)		p+n	m+1	t
DA	1	wrm	ℓ	(m)	a	p	m	t
	2	amipa, $e\ell$ e	(m)	ℓ	p	m+1	a	
	3	test, $\ell i\ell$	(m)-1				a	
	3a	test, $\ell i\ell$, mim	(m)-2				a+1	
	3a	test, $\ell i\ell$, mim	(m)-n-1				a+n	
	4	pmp, $e\ell$, rdm	ℓ	(m+1)	p	a+n	m+1	t

3.5.9 Multiply (MP)

The MP command is used to form the arithmetic product of two operands. Logands containing the MP command have the following format:

15 10	9 7	6 5	4 1
MP	DM	DR	n

The address option must be DM. The control field must contain DR (binary 10). The carry indicator must initially be set to zero.

The MP logand forms a (15 + n) - bit quantity r, which is numerically equivalent to:

$$r = \ell \times p_n + a$$

where

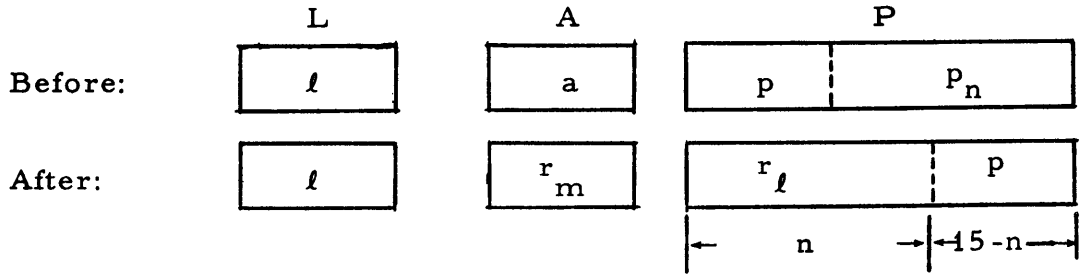
ℓ = initial contents of the L register, interpreted as a 15-bit positive multiplicand,

a = initial contents of the A register, interpreted as a 15-bit positive integer,

p_n = initial contents of the n low-order positions of the P register, interpreted as an n-bit positive multiplier,

n = contents of positions one-four of the logand, $0 \leq n \leq 15$

At the end of the logand, the 15 most-significant bits of r are held in the A register, while the n least-significant bits of r are held in the n high-order positions of the P register. The (15 - n) low-order positions of P hold the initial contents of the (15 - n) high-order positions of P.



If n = 0, no multiplication takes place and the contents of L, A, and P remain unchanged.

The MP logand cycle is shown in table 3-10. The single-position shift operation initiated on clock pulse 3a is a function of the low-order bit of the P register on that clock pulse. If $p_1 = 0$ (corresponding to a zero multiplier digit), the operation sodr is initiated. If $p_1 = 1$, the operation smuℓ is initiated. (See figure 2-11, page 2-22)

The MP logand requires 12 + 3n microseconds, where n is the integer in positions one-four of the logand.

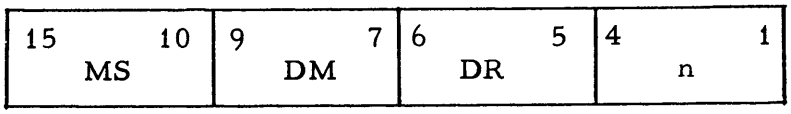
Table 3-10. MP Logand Cycle

Clock	Operations	L	E	A	P	M	T
1	wrm	ℓ	(m)	a	p	m	t
2	mim, eℓe	(m)	ℓ			m+1	
3	test						
3a	ℓiℓ, test, sodr/smuℓ	(m)-1		as shifted			
.							
.							
.							
3a	ℓiℓ, test, sodr/smuℓ	(m)-n					
4	eℓ, rdm	ℓ	(m+1)				

Test: If the four low-order positions of L contain zero, initiate the clock four operations; otherwise, repeat clock 3a.

3.5.10 Multiply Signed (MS)

The MS command is used to form the arithmetic product of two operands where at least one of them is signed; only one of them can be negative. Logands containing the MS command have the following format:



The address option must be DM. The control field must contain DR (binary 10). The carry indicator C must initially be cleared to zero.

The MS logand forms a (15 + n)-bit quantity r which is numerically equivalent to:

$$r = \ell \times p_n + a$$

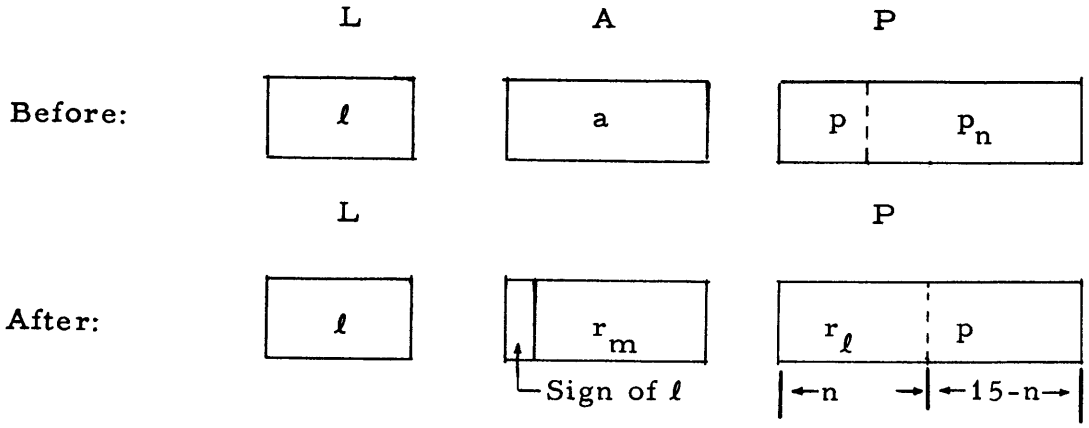
where:

ℓ = initial contents of the L register, interpreted as a 15-bit signed multiplicand

a = initial contents of the A register, interpreted as a 14-bit signed integer

p_n = initial contents of the n low-order positions of the P register, interpreted as an n-bit multiplier

n = contents of positions one to four of the logand, where $0 \leq n \leq 15$



If n = 0, no multiplication takes place and the contents of L, A, and P remain unchanged.

The MS logand cycles are shown in table 3-11. The single-position shift operation initiated on clock pulse 3a is a function of the low-order bit of the P register on that clock pulse. If $p_1 = 0$ (corresponding to a zero multiplier digit), the operation sndr is initiated. If $p_1 = 1$, the operation smsℓ is initiated. (See figure 2-11, page 2-22)

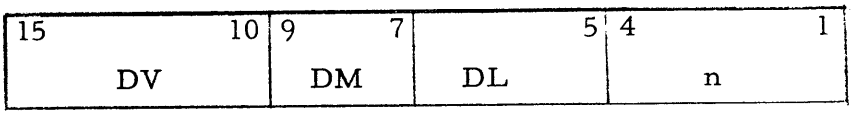
The MS logand requires 12 + 3n microseconds, where n is the integer in positions one to four of the logand.

Table 3-11. MS Logand Cycles

Clock	Operation	L	E	A	P	M	T
1	wrm	ℓ	(m)	a	p	m	t
2	mim, eℓe	(m)	ℓ			m+1	
3	test						
3a	ℓiℓ, test, sndr/smsℓ	(m)-1		(as			
.				shifted)			
.				↓	↓		
.							
3a	ℓiℓ, test, sndr/smsℓ	(m)-n					
4	eℓ, rdm	ℓ	(m+1)				

3.5.11 Divide (DV)

The DV command is used to form the quotient of two positive operands. Logand containing the DV command have the following format:



The address option must be DM. The control field must contain DL (binary 11). Before executing the DV logand, the carry indicator must be set to zero. The overflow indicator may be set by the DV logand (see below), hence it should also be set to zero before executing the logand if it is to be tested afterward.

The DV logand forms an (n+1)-bit quantity which is numerically equivalent to

$$q = (a + p) \div (-\ell)$$

where

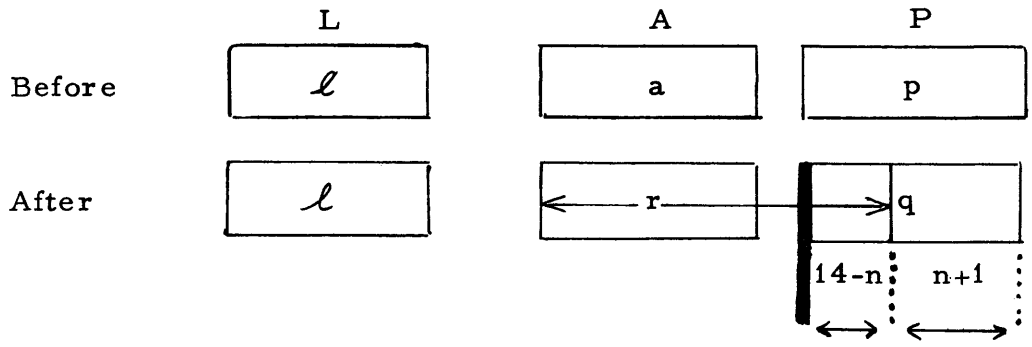
a = initial contents of the P register, viewed as a 14-bit positive integer (i.e., a_{15} must be zero);

p = initial contents of the P register, viewed as a 15-bit positive fraction;

ℓ = initial contents of the L register, a 14-bit negative integer in two's complement form (i.e., ℓ_{15} must be 1); and

n = contents of positions one to four of the logand, where $0 \leq n \leq 15$.

At the end of the logand, q is held in the low order n+1 positions of the P register, with binary point between positions n and n+1. The A register and the high-order 14-n positions of the P register hold the unsigned undivided remainder r, with the same scaling as the dividend. (If n = 15, the high-order bit of q is in the low-order position of the A register, and the remainder is in positions 15-2 of the A register.)



The significance of the result of the DV logand depends on the relative magnitudes of the dividend and divisor, as follows:

(a) $(a+p) < (-\ell)$: The result q will be a fraction; i.e., the high order bit of q will be zero.

(b) $(-\ell) \leq (a+p) < 2(-\ell)$: The result q will be a number greater than or equal to 1 but less than 2. The overflow indicator will be set, even though the result may be meaningful.

(c) $(a+p) \geq 2(-\ell)$: The result is not meaningful. The overflow indicator is set.

The DV logand cycle is shown in Table 3-12. The single position shift operation initiated on clock pulse 3a is a function of the carry output C_{15} from the high-order adder position on that clock pulse. If $C_{15}=0$ (indicating that the current remainder is smaller than the divisor), the operation sodℓ is initiated (see Figure 2-11, page 2-22). If $C_{15}=1$ (indicating that the current remainder is larger than the divisor), the operation sdiv is initiated.

The DV logand requires 12 + 3n microseconds, where n is the contents of positions 1-4 of the logand.

Table 3-12. DV Logand Cycles

Clock	Operations	L	E	A	P	M	T
1	wrm	ℓ	(m)	a	p	m	t
2	mim, eℓe	(m)	ℓ			m+1	
3a	ℓiℓ						
	test, sodℓ/sdiv			(as			
3a	ℓiℓ			shifted)			
	test, sodℓ/sdiv						
.							
.							
.							
3a	ℓiℓ						
	test, sodℓ/sdiv						
4	eℓ, rdm	ℓ	(m+1)				

Test: If the four low-order positions of L contain zero, initiate clock 4 operations; otherwise, repeat clock 3a.

3.5.12 Move (MV)

This command causes a block of words to be moved from one set of consecutive memory locations to a second set of memory locations. The format of the MV logand is as follows:

15 10	9 7	6	5 1
MV	IP	–	CONDITION

Address option IP must be used; the conditions allowed for termination are NV, EN, EQ, NQ, NH, and NL. Octal codes for these conditions are given in Table 4-21, page 4-33.

Before executing the MV logand, the A register must be loaded with the starting address of the sequential words to be moved; the P register must be loaded with the address of the first location to receive the moved words; the T register must be loaded with the one's complement of a + n, where n is the number of words to be moved. The carry must also be cleared to zero.

Cells a through a+ (n - 1) are moved if terminated on the address; however, if a condition other than NV is used, termination may occur before if (a + i) compares with t, according to the condition.

The execution time of this logand is 12 microseconds for each word moved, plus an 18-microsecond overhead for the logand. The MV logand cycle is shown in table 3-13.

3.5.13 Table Search (TB)

The TB command is used to compare a given "search" word with each word in a block of consecutive memory locations, until a specified test condition is met or until a specified number of words have been compared. Logands containing the TB command have the following format:

15 10	9 7	6	5 1
TB	IP	-	CONDITION

Address option IP must be used. Bit positions one to five of the logand may contain EQ, NQ, NL, or NH as a test condition. Octal codes for these conditions are given in Table 4-21, page 4-33.

Before executing the TB logand, the A, P, and T registers must be loaded as follows: The A register must contain the address a of the first word in the block to be searched; the T register must contain the one's complement of the search word; and the P register must contain the one's complement of a + n, where n is the number of words in the block. In addition, the carry indicator must be set to zero.

In execution of the TB logand, the search word is compared in turn to the contents of memory locations a, a+1, a+2, etc., with one of two conditions satisfied:

(a) The specified test condition is satisfied by the contents of a + k - 1, k = 1, 2, . . . n. When the test for this condition is made, the A register contains the complemented search word (i.e., the initial contents of T); and the E register contains (a+k). If the logand terminates

Table 3-13. MV Logand Cycles

Clock	Operation	L	E	A	P	M	T	Remarks
1	wrm	ℓ	(m)	a	p	m	t	
2	pmiap, eℓe	(m)	ℓ	m+1	a	p		
3	ata			t			m+1	
4	pmp, rdm		(a)		p	a		
5	wrm							
6	epmiae		t	a+1	(a)	p		
	cℓm							
3	apea, test,		(a)	t	a+1			(a)——→(p)
	wrm							
4	pmip, rdm, test		(a+1)		p+1	a+1		
5	wrm							
6	epmiae		t	a+2	(a+1)	p+1		
	cℓm							
.	.		.	.	.	.	.	
.	.		.	.	.	.	.	
.	.		.	.	.	.	.	
3	test, apea,		(a+n-1)	t	a+n	p+n-1		(a+n-1)→(p+n-1)
	wrm							
4	pmip, rdm, test		(a+n)		p+n	a+n		
5	wrm, ata			m+1			t	
6	ama, eℓ, rdm	(a+n)	(m+1)	a+n	p+n	m+1	t	

under this condition, the overflow indicator is unchanged. The address a + k is left in A, and the contents of a + k are left in L.

(b) The address of the word to be compared next equals a + n. Under this condition, the overflow indicator is set to one. The A register holds a + n, and the L register holds (a + n).

The TB logand cycle is shown in table 3-14. It should be noted that the setting of the carry indicator is not changed during the cycle, and that its initial setting (0), in general, affects the test for the specified condition.

The TB logand cycle requires 18 + 12n microseconds, where n is the number of table words compared before the logand is terminated.

3.5.14 Match (MH)

The MH command is used to compare corresponding words in two blocks of consecutive memory locations until a specified test condition is satisfied. Logands containing the MH command have the following format:

15 10	9 7	6	5 1
MH	IP	-	CONDITION

The address option IP must be used. Bit positions one to five of the logand may contain any of the following test conditions: EQ, NQ, NH or NL. Octal codes for these conditions are given in Table 4-21, page 4-33.

Before executing the MH logand, the A register must be loaded with the address a of the first word of the first block; and the P register must be loaded with the address p, which is one less than the first word of the second block. In execution of the logand, the contents of location a are compared with the contents of location p + 1, the contents of a + 1 with the contents of p + 2, etc., until the specified test condition is satisfied. When the test for this condition is made, the A register

Table 3-14. TB Logand Cycle

Clock	Operation	L	E	A	P	M	T	Remarks
1	wrm	ℓ	(m)	a	p	m	t	
2	pmiap, eℓe	(m)	ℓ	m+1	a	p		
3	ata			t			m+1	
4	pmp, rdm		(a)		p	a		(a) →(E)
5	wrm							
6	test, epe, amia		p	a+1	(a)	t		
3	test, emape		(a)	t	a+1	p		
4	pmp, rdm,test		(a+1)		p	a+1		(a+1) →(E)
5	wrm							
6	epe, amia		p	a+2	(a+1)	t		
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
3	test, emape		(a+n-1)	t	a+n	p		
4	pmp, rdm, test		(a+n)		p	a+n		
5	wrm, ata			m+1			t	
6	ama, eℓ, rdm	(a+n)	(m+1)	a+n	p	m+1	t	

contains the one's complement of (a + k) and the E register contains (p + k + 1). If the test is satisfied by the word-pair (a + k - 1) and (p + k), then at the termination of the logand, P contains p + k + 1; A contains a + k; T contains the one's complement of (a + k - 1); and L contains (a + k - 1).

The MH logand cycles are shown in table 3-15. The Carry Indicator is not affected by the cycle, and its initial setting (which may be either zero or one) in general affects the test for the specified condition.

The MH logand requires 18 + 12k microseconds, where k is the number of word-pairs compared before the test condition is satisfied.

3.5.15 Sort (SR)

The SR logand is used to compare a sequence of words in a block to a word in memory. It also compares each word in the sequence to the succeeding word in the sequence. Logands containing SR have the following format:

15 10	9 7	6	5 1
SR	IP	-	CONDITION

The address option IP must be used. Bit positions one to five of the logand may contain any of the following test conditions: EQ, NQ, NH or NL. Octal codes for these conditions are given in Table 4-21, page 4-33.

Before execution of the SR logand, the A register must be loaded with the address of the first word in the block; the P register must be loaded with the address of the word in memory to be compared. In execution of the logand, the contents of location a are compared first with the contents of location p, and then with the contents of location a + 1;the contents of a + 1 are compared with the contents of p, and then a + 2, etc., until the specified condition is satisfied. When the test for the condition is made, A contains (a + k)', and E contains first (p) and then (a + k + 1)).

Table 3-15. MH Logand Cycles

Clock	Operation	L	E	A	P	M	T
1	wrm	*l*	(m)	a	p	m	t
2	apmia, e*l*, rdm	(m)	(p)	m+1	a	p	
3	wrm, ata			t			m+1
4	pmip, rdm		(a)		p+1	a	
5	wrm						
6	pmip, ea, rdm		(p+1)	(a)	a+1	p+1	
3	wrm, cpa,			(a)'			
4	pmip, rdm, test		(a+1)		p+2	a+1	
5	wrm						
6	pmip, ea, rdm		(p+2)	(a+1)	a+2	p+2	
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
3	wrm, cpa,		(p+n)	(a+n-1)'	a+n	p+n	
4	pmip, rdm, test		(a+n)		p+n+1	a+n	
5	wrm, ata			m+1			(a+n-1)'
6	ama, e*l*, rdm	(a+n)	(m+1)	a+n	p+n+1	m+1	(a+n-1)'

At the termination of the logand, P contains p; A contains a + n; T contains*(a + n - 1) ; and L contains (a + n).

*If the termination was caused by the (a + k) and (a + k + 1) comparison, the overflow indicator is set to one and T contains (a + k + 1). If the comparison of (a + k) and (p) causes the termination, the overflow indicator is unchanged, and T contains (a + k)'.

The SR logand cycles are shown in table 3-16. The carry indicator is not affected by the cycle; its initial setting, in general, affects the test for the specified condition.

The SR logand requires 18 + 12n microseconds, where n is the number of p comparisons made before termination.

Table 3-16. SR Logand Cycles

Clock	Operation	L	E	A	P	M	T
1	wrm	ℓ	(m)	a	p	m	t
2	apmia, eℓ, rdm	(m)	(p)	m+1	a	p	
3	wrm, ata			t			m+1
4	pmp, rdm		(a)		p	a	
5	wrm						
6	pmip, ea, rdm, test		(p)	(a)	a+1	p	
3	wrm, cpa			(a)'			
4	pmp, rdm, test		(a+1)		p	a+1	
5	wrm						
6	pmip, ea, rdm, test		(p)	(a+1)	a+2	p	
.	.			.			
.	.			.			
.	.			.			
3	wrm, cpa		(p)	(a+n-1)'	a+n	p	
4	pmp, rdm, test		(a+n)		p	a+n	
5	wrm, ata			m+1			*(a+n-1)
6	ama, eℓ, rdm	(a+n)	(m+1)	a+n	p	m+1	
	*See written description.						

3.5.16 Control Function (CF)

The Control Function Command is used to

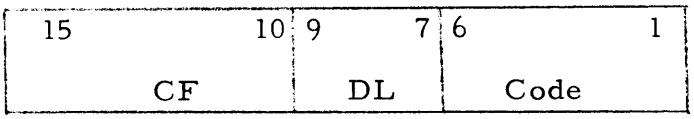
(1) select cable set combinations preparatory to issuing commands to and exchanging data with input/output devices;

(2) control the TRW-173 Buffer channel

(3) input status information; and

(4) reset certain indicators

Logands containing the CF command have the following format;



DL is the only address option allowed with the CF logand. The code in the low-order six bits of the logand indicates the particular function to be performed by the CF logand.

The CF logand cycle has the same effect on the registers as a NØ regular primary command used with a direct address option. Execution time for the CF logand is 12 μsec, except when used to input an interrupt status word from a selected cable. In this case the execution time is 27 μs.

3.5.16.1 Selection of Cable Set

Three sets of input/output cables, designated A, B, C, may be used to connect the TRW-130 computer to external devices. Cables A and B are designed for connection to NTDS devices. Cable C may be used to link the computer to a variety of peripheral devices.

Each cable set contains an input cable and an output cable. An input cable consists of

(a) Input Data Lines, over which data is transmitted to the computer. Input cables in cable sets A and B have 30 input data lines, while the input cable in set C has 15 input data lines.

(b) Interrupt Line, by which a peripheral device can interrupt the TRW-130. An interrupt causes the TRW-130 to stop executing its current program and instead execute an interrupt subroutine. Cables A-in and B-in each contain one interrupt line. Cable C-in contains nine interrupt lines.

(c) Input Data Request Line, by which a peripheral device signals the TRW-130 that it has a data word on the data lines for input.

(d) Input Acknowledge Line, by which the TRW-130 signals a peripheral device that it has taken data or has executed an interrupt.

An output cable consists of

(a) Output Data Lines, over which data is transmitted from the computer. Output cables in cable sets A and B have 30 output data lines, while the output cable in set C has 15 output data lines.

(b) Output Data Request Line, by which a peripheral device signals the TRW-130 that it is ready to accept a word of data.

(c) Output Acknowledge Line, by which the TRW-130 signals a peripheral device that a word of data is on the output lines.

(d) External Function Line, by which the TRW-130 signals a peripheral device that a word is on the output lines, which is to be interpreted as an external function code (a command or control function, as opposed to data).

Although a number of peripheral devices may be connected to the TRW-130, only two communication channels - one input and one output - may be active at any given time. The codes used to select cable combinations with the CF command are:

Octal	Decimal	Input Cable	Output Cable
10 or 11	8 or 9	C-in (15 bits)	C-out (15 bits)
12	10	C-in (15 bits)	B-out (30 bits)
13	11	C-in (15 bits)	A-out (30 bits)
14	12	B-in (30 bits)	C-out (15 bits)
15	13	A-in (30 bits)	C-out (15 bits)
16	14	B-in (30 bits)	B-out (30 bits)
17	15	A-in (30 bits)	A-out (30 bits)

3.5.16.2 Control of Buffer Channel

The TRW-173 Buffer Channel, Cable D, is an intermediary between the TRW-130 computer and fast peripheral devices. The CF command can exercise direct control over the Buffer Channel, using these codes:

Octal	Decimal	Command
00	0	Terminate Computer Mode
01	1	Establish Computer Mode
02	2	Start Buffer Output
03	3	Start Buffer Input
04	4	Stop Buffer
07	7	EF to Cable D (wait)
31	25	EFE Buffer Transfer
32	26	FQR Buffer Transfer
33	27	RQF Buffer Transfer
42	34	EF to Cable D (NTDS)

The Buffer Unit can be commanded to take over many of the input/output request and acknowledge communications that would otherwise be handled by the computer. The Buffer Channel makes actual data transfers to and from the computer by means of a cycle-stealing technique: that is, during any computer cycle except read or write, the buffer may access computer memory to input or take for output the data it has been instructed to process. It bypasses the computer's input/output section and reduces loss of computation time during its asynchronous input/output operations to only 9 μsec per word.

The Buffer Channel contains three 15-bit registers:

F register: An alternate for the computer's E register, to hold the current word being transferred.

R register: An alternate for the computer's M register, to hold the address of the next word to be transferred.

Q register: A length register, to hold the word indicating the total number of words to be transferred.

The programmer writes the instructions to set up the F, R and Q registers, designating the transfer as input or output. The last instruction initiates the transfer and releases as the Buffer Unit from computer control. The Buffer Channel then operates under its own control. A detailed description of the Buffer Unit is given in the TRW-173 Buffer Unit Reference Manual.

3.5.16.3 Input Status Words

The following CF codes are used to input status words:

Octal	Decimal	Command
21	17	Input Interrupt Status Word to E register
22	18	Input the 15 Control Panel switches (toggle) into E register.
40	32	Input a 15-bit Interrupt Code word from Cable C-in into E register.
41	33	Input a 15-bit Interrupt Code word from Cable D-in into E register. (Buffer Channel)
43	35	Input a 30-bit Interrupt Code word from Cable A-in into the E and T registers. The most significant word is in E.
46	38	Input a 30-bit interrupt code word from Cable B-in into the E and T registers. The most significant word is in E.

Interrupt Status Word Each bit of the Interrupt Status word is assigned a unique meaning. The 13 interrupt lines identified are the nine lines from Cable C, one line from Cable A, one line from Cable B, and the two lines from the control panel switches, MOMENTARY INTERRUPT and CONTINUOUS INTERRUPT. Variable assignments may be made according to the types of devices used with the system. A typical assignment of Interrupt Status word identifiers is:

Bit	Meaning
15	Unavailable
14	TRW-173 Buffer
13	Unused
12	Momentary Interrupt Switch
11	Cable A
10	Cable B
9	Cable C - line 1
8	Cable C - line 2
7	Cable C - line 3
6	Cable C - line 4
5	TRW-80 Display System
4	TRW-192/170 Magnetic Tapes
3	Teletype
2	Flexowriter
1	Continuous Interrupt Switch

Interrupt Code Word The interrupt code word indicates the device on the predetermined interrupt line that caused the interrupt. If a given device can have more than one reason for interrupting, this code word also gives the specific reason.

The exact bit meaning for the interrupt code word depends upon the interrupting device, and is specified at installation.

3.5.16.4 Reset Indicators

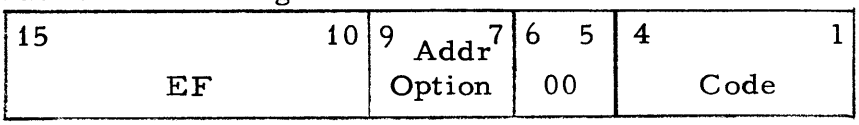
The parity error indicator and the Momentary Interrupt indicator are cleared by CF logands with the following codes:

Octal	Decimal	Meaning
20	16	Reset Memory Parity Error Indicator
24	20	Reset Momentary Interrupt Indicator.

3.5.17 External Function (EF)

The EF command causes command words, called external functions, to be sent to peripheral devices.*

The format for the EF logand is



The allowable address options are DM,DP, DA, IM, IP, IA. Bits 5 and 6 (the Control Field) are always zero. The contents of the low-order four bits of the EF logand determine the output cable on which the EF command will be sent, and turns on the External Function line to indicate that the word on the output data lines is to be interpreted as an EF command.

The EF logand cycle has the same effect on the registers as a NO (No operation) regular primary command. Execution time for the EF logand is 12 μsec. or 18 μsec., depending upon the address option with which it is used.

---

* It is possible to transmit an EF code to a device which has not been selected by a CF logand.

The EF logand codes are as follows:

Decimal	Octal	Interpretation by Input-Output Section
1	01	First EF logand to send command on Cable A-out (T contains most significant 15 bits of 30-bit word)
3	03	Second EF logand to send command on Cable A-out (E contains least significant 15 bits of 30-bit word)
4	04	First EF logand to send command on Cable B-out (T contains most significant 15 bits of 30-bit word)
6	06	Second EF logand to send command on Cable B-out (E contains least significant 15 bits of 30-bit word)
8	10	EF logand to send command on Cable C-out (T contains the 15-bit word)

The EF logand completes the connection of a peripheral device on the selected out-cable, and signals the device that a command is about to be transmitted.

Devices using cable C must be capable of connecting and disconnecting, since more than one device may use cable C. A device is said to be "connected", or "true", if it

(a) controls the data request line, and

(b) is responsive to the acknowledge line;

otherwise, it is said to be "disconnected."

A command word is used with the EF logand to specify which device is to be made "true", or connected, and whether it is to serve as an input or as an output device. Actions other than connect or disconnect can also be controlled by the EF command word. These may be coded uniquely for each device in bits 12 through 9. (See Magnetic Tape Manual, M250-2U 44).

The convention has been established that any external function signal in which bit 15 is a 1 is interpreted as a command to connect or disconnect an input device. A 1 in bit 14 is interpreted as a command to connect or disconnect an output device.

In general, the format of the command word is as follows:

	Connect Command			Select Unit	Select Device
1 in Bit	15	14	13	12 - - - - - - 8	7 - - - - - - - - 1
	Connect Input Device	Connect Output Device	Connect Mag. T Cntrllr	(Mag. Tapes)	(or designate action; i.e. for mag. tapes)

For example, assume the assignment of the following command word codes:

Octal Code	Meaning
20000	Disconnect all Cable C output devices
20001	Connect paper tape punch (as an output device)
20002	Connect teletype(as an output device)
40001	Connect paper tape reader (as an input device)
60000	Disconnect all Cable C (input and output) devices
60001	Connect paper tape reader (input) and paper tape punch (output)
60002	Connect typewriter (as both input and output device)

Prior to initiating data transfer between an external device and the computer, the following conditions must be satisfied:

1. The computer must be connected to the appropriate input and output cables by a CF logand.
2. The external device must be connected to an input and/or output cable. This is done with an EF command, if the device is one capable of "connecting" and "disconnecting".
3. Commands must be issued by the computer program to control the actual transfer of information between the peripheral device and the computer.

Information may be transferred as single words, or in blocks of words.

3.5.18 Word Input (WI) and Word Output (WO)

The WI and WO commands transfer single (15-bit) words from and to the computer and the external devices. The format of the WI and WO logands is as follows:

15 — 10	9 — 7	6 — 1
WI WO	Addr. Option	NO or S

All address options are allowed. The secondary command (for address options other than DL or IL) is ignored. With DL and IL address options, bits 1-6 designate a scratchpad memory address where data is to be stored or to be extracted.

3.5.18.1 WI Logand Cycle

The WI logand cycle has the same effect as the primary command NO (No Operation) except for the following:

(a) The quantity in E following clock pulse two or four of the logand cycle (for direct and indirect address options, respectively) is replaced by a 15-bit quantity supplied by the I/O control unit. The computer idles until this has been done, then initiates clock three and four operations (or five and six, with indirect address options).

(b) The E register is the primary input register, and always receives an input word from the connected input data lines. The 15-bit word is stored in memory at the address designated.

Two WI logands in succession are required to transmit a 30-bit message. The message is received in the E and T registers in parallel, the T register receiving the least significant word.

WI execution time:

15-bit word (Cable C)	30-bit word (Cables A and B)
30 μsec per word	30 μsec per word.

3.5.18.2 Data Input Sequence

The following sequence occurs during a data input:

(a) The peripheral device places a data word on the lines (30 bits or less, using A-in or B-in; 15 bits or less, using C-in)

(b) The peripheral device makes its input data request line true to indicate that data is on the lines. Once initiated, this signal and the data must be maintained until acknowledged by the TRW-130.

(c) The TRW-130 detects an input data request at its own convenience.

(d) After detection of the data request, the TRW-130 samples the data lines.

(e) TRW-130 makes the input acknowledge line of the appropriate cable true.

(f) The peripheral device detects an input acknowledge.

(g) The peripheral device drops the input data request.

(h) In the case of the C-in cable, the TRW-130 drops the input acknowledge signal following removal of the input data request.

3.5.18.3 WO Logand Cycle

The WO logand cycle has the same effect as the primary command NO (No Operation) except for the following:

(a) The quantity in E following clock pulse two (or four) is transferred to the I/O control unit, via the T register, and the E register is held. The computer idles until this has been done, and then initiates the clock three and four operations (or clock five and six, with indirect address.)

(b) The T register is the primary output register, and always receives the output word from computer memory.

Two WO logands in succession are required to transmit a 30-bit message. The message is transmitted in parallel from the E and T registers, the T register holding the most significant word, the E register holding the least significant word.

NOTE: During WO commands to Cable C-out connected devices, the T register holds the output word until data transmission is completed. An attempt

during this time to execute a command (other than an input-output command) which alters the contents of the T register, may affect the output data word.

A BR or SK logand with the condition parameter TL (T Register in Use) will protect the contents of the T register until transmission is completed.

WO execution times:

15-bit word (Cable C)	30-bit word (Cables A and B)
33 μsec per word	48 μsec per word

3.5.18.4 Data Output Sequence

The following sequence occurs during data output:

(a) The peripheral device makes its output data request line true to indicate that it is prepared to accept data. This signal, once initiated, must be maintained until acknowledged.

(b) TRW-130 detects an output data request at its convenience.

(c) TRW-130 places a data word on its output lines.

(d) After the data word is placed on the lines, the corresponding output acknowledge line is made true. For cable A-out or B-out, this signal is maintained for 15 microseconds. For Cable C-out, this signal is maintained until the data request is made false.

(e) The peripheral device detects an output acknowledge.

(f) The peripheral device samples the data lines.

(g) The peripheral device may drop the output data request at any time following detection of the output acknowledge.

(h) In the case of cable C-out, the TRW-130 drops the output acknowledge signal following removal of the output data request.

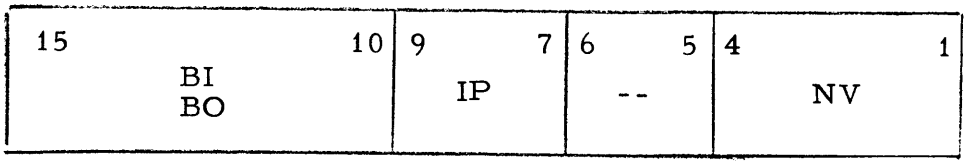
NOTE: This sequence is repeated for each data word transferred.

3.5.19 Block Input (BI) and Block Output (BO)

The BI and BO logands are used to transfer blocks of 15-bit words from external devices to computer memory, and from the computer to external devices, respectively.

The input or output cable must have been previously selected by a CF logand, and the input or output device designated by an EF logand, together with the appropriate device code when necessary.

The BI and BO command logands have the following format:



The address option is always IP, and the condition is always NV (Never). The BI or BO logand must be preceded by logands which

(a) load A register with a, the starting address of input or output data; and

(b) load P register with p*, the ending address of input or output data.

Thus:

```
LA/DM/NO
OCT XXXX (starting address)
LP/DM/NO
OCT XXXX (ending address)
BO/IP/NV
```

would set up a block output transfer.

The transfer is terminated when the last address to be read into or from is equal to the one's complement of the P register.

---

* If carry indicator zero: with one's complement of a+n-1
If carry indicator one : with one's complement of a+n

where n is the number of words to input or output.

The BI and BO logand cycles are shown in table 3-17. The cycles are the same, except for the disposition of the quantity in the E register at clock pulse five:

BI quantity in E is replaced or modified by the I/O control unit.

BO quantity in E is output by the I/O control unit through the T register.

NOTE: During BO commands to Cable C-out connected devices, the T register holds the output word until data transmission is complete. An attempt during this time to execute a command (other than an input-output command) which alters the contents of the T register, may affect the output data word.

A BR or SK logand with the condition parameter TL (T Register in Use) will protect the contents of the T register until transmission is completed.

Minimum execution time for BI or BO logands is 18 + 12n μsec, where n is the number of words transferred.

Table 3-17. BI and BO Logand Cycles

Clock	Operation	L	E	A	P	M	T
1	wrm	ℓ	(m)	a	p	m	t
2	apmia, eℓe	(m)	ℓ	m+1	a	p	
3							
4	pmp, rdm		(a)		p	a	
5	wrm						
6	amia, epe		p	a+1	(a)	m+1	
3	emape, test		(a)	m+1	a+1	p	
4	pmp, rdm		(a+1)		p	a+1	
5	wrm						
6	amia, epe		p	a+2	(a+1)	m+1	
.	.			.			
.	.			.			
.	.			.			
3	emape		(a+n-2)	m+1	a+n-1	p	
4	pmp, rdm		(a+n-1)		p	a+n-1	
5	wrm						
6	amia, eℓ, rdm	(a+n-1)	(m+1)	a+n	p	m+1	t

3.5.20 Interrupt (IT)

The interrupt (IT) logand causes the normal sequencing of logand execution to be altered. The IT logand is normally supplied automatically in response to interrupt signals from input/output devices or certain computer indicators, but may also appear explicitly in a stored program.

3.5.20.1 Externally-Initiated Interrupts

The computer can respond to three types of interrupt signals which are generated by input/output devices or certain computer indicators:

Type I Input Interrupt: A device which has been commanded to transmit data to the computer sets the input data request line TRUE whenever it is ready to transfer a 15- or 30-bit word to the computer. If the computer is not executing a CF, TM (Terminate Interrupt), WI, or BI logand, and if the Type I Interrupt Indicator is set to 0 (zero), the Type I Interrupt Indicator will be set to 1, and the computer will be interrupted.

If the computer is executing one of the indicated logands, the computer is allowed to finish the logand cycle and start the next logand cycle; if the input data request line is still TRUE, another attempt is made to interrupt the computer. If the Type I Interrupt Indicator is set to 1, the computer cannot be interrupted by a TRUE input data request line until the indicator has been set to 0 (zero) by the TM logand.

Type I Output Interrupt: A device which has been commanded to receive data from the computer sets the output data request line TRUE whenever it is ready to receive a 15- or 30-bit word from the computer. The conditions under which the computer will be interrupted are the same as for TRUE input data request lines (see Type I Input Interrupt above).

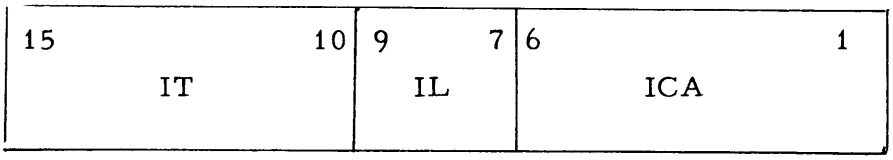
Type II Interrupt: Type II interrupt signals are generated by peripheral devices and certain computer indicators independently of the computer program. Cable sets A and B and the Buffer Channel

each have one interrupt line for this purpose, while cable set C has 9 interrupt lines which may be assigned to the various devices on this cable. In addition, the setting of the MOMENTARY INTERRUPT indicator (by depressing the MOMENTARY INTERRUPT pushbutton on the control panel) and the setting of the CONTINUOUS INTERRUPT indicator (by setting the CONTINUOUS INTERRUPT switch on the control panel) result in the generation of a Type II interrupt signal.
If the computer is not executing a CF or TM logand, and if the Type II Interrupt Indicator is set to 0, the Type II Interrupt Indicator will be set to 1 and the computer will be interrupted.

If the computer is executing a CF or TM logand, the computer is allowed to finish the logand and start the next logand.

If the Type II interrupt signal is still present, another attempt is made to interrupt the computer. If the Type II Interrupt Indicator is set to 1, the computer cannot be interrupted by a Type II interrupt signal until the indicator has been set to 0 (by the TM logand).

When an interrupt signal is received and the appropriate Interrupt Indicator is set, the computer completes the current logand cycle in the normal manner, with the following exception: on the last clock pulse of the cycle, the operation rdm (normally used to read the next logand) is suppressed. Instead, the I/O control unit transfers into the E register an interrupt logand with the following format:



The address option is always IL. The secondary field contains an interrupt control address (ICA) which is one of three scratchpad addresses, depending on the type of interrupt signal received:

TYPE I INPUT INTERRUPT	ICA = 6
TYPE I OUTPUT INTERRUPT	ICA = 4
TYPE II INTERRUPT	ICA = 0

The computer then executes the IT logand as follows:

(a) The contents of L at the beginning of the logand are stored in memory location ICA (0, 4, or 6).

(b) The contents of M at the beginning of the logand (i.e. the address of the logand that would have been executed next in the absence of the interrupt signal) are interchanged with the contents of memory location ICA + 1 (1, 5, or 7). As a result, the next logand is taken from the address held initially in ICA + 1.

In order that meaningful operations ensue from an interrupt, scratchpad location ICA + 1 must be loaded, before the interrupt, with the address of the first logand of an interrupt subroutine.

If the interrupt is of Type II, the interrupt subroutine will normally determine the cause of interrupt by testing the Interrupt Status Word (section 3.5.16.3) and take whatever action is required.

If the interrupt was due to the setting of the MOMENTARY INTERRUPT indicator, the indicator must be reset (section 3.5.16.4) before the interrupt is terminated in order to avoid another momentary interrupt.

An interrupt is terminated by executing the TM logand. This logand resets the Type I or Type II Interrupt Indicator, and allows future interrupts of the corresponding type.

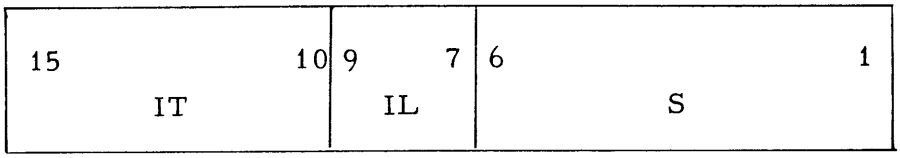
When the input/output unit must choose between two or three interrupts, the following priorities apply:

TYPE I interrupt with input channel address
TYPE I interrupt with output channel address
TYPE II interrupts

The IT logand cycle for externally-initiated interrupts is shown in Table 3-18(a). The cycle requires 18 microseconds.

3.5.20.2 Program-Initiated Interrupts

When the IT logand is used in a program, it must have the following format:



Address option IL must be used. The secondary field may contain any even scratchpad address, S. The effect of the logand is as follows:

(a) The contents of L at the beginning of the logand are stored in memory location S.

(b) The contents of M at the beginning of the logand are incremented by one and interchanged with the contents of memory location S+1. As a result, the next logand is taken from the address held initially in S+1.

Table 3-18. IT Logand Cycles

(a) I/O-Initiated Interrupt

Clock	Operation	L	E	A	P	M	T	Memory
1		ℓ	IT/IL/ICA	a	p	m	t	
2	eℓe, cℓℓ	IT/IL/ICA	ℓ					0 → (ICA)
3	wrℓ							ℓ → (ICA)
4	odℓ, rdℓ	ICA+1	(ICA+1)					(ICA+1) → (E)
5	eme, wrℓ		m			(ICA+1)		m → (ICA+1)
6	eℓ, rdm	m	((ICA+1))			(ICA+1)		

(b) Programmer-Initiated Interrupt

Clock	Operation	L	E	A	P	M	T	Memory
1	wrm		IT/IL/S	a	p	m	t	
2	eℓe, mim, cℓℓ	IT/IL/S	ℓ			m+1		0 → (S)
3	wrℓ							ℓ → (S)
4	odℓ, rdℓ	S+1	(S+1)					(S+1) → (E)
5	eme, wrℓ		m+1			(S+1)		m+1 → (S+1)
6	eℓ, rdm	m+1	((S+1))			(S+1)		

Before executing the IT logand, location S+1 must be loaded with the address to which control is to be transferred.

The principal use of the program-initiated interrupt is to inhibit externally initiated interrupts. If the address S has a zero in bit position 3, the Type II Interrupt Indicator will be set to 1, thus inhibiting Type II interrupts. If the address has a 1 in bit position 3, the Type I Interrupt Indicator will be set to 1, thus inhibiting Type I Input interrupts and Type I Output interrupts. Thus, the logands

IT/IL/20	010000
IT/IL/30	011000

will inhibit Type II interrupts, while the logands

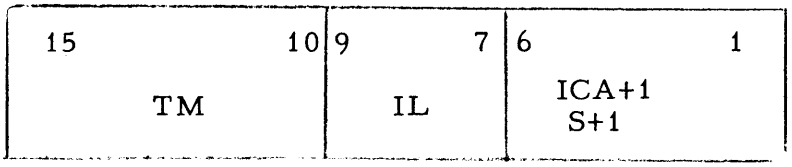
IT/IL/24	010100
IT/IL/26	010110

will inhibit Type I interrupts.

The program-initiated IT logand cycle is shown in Table 3-18 (b). The cycle requires 18 microseconds.

3.5.21 Terminate Interrupt (TM)

The Terminate Interrupt (TM) logand is used to return program control to the logand sequence that would have ensued if an interrupt had not occurred. The TM logand has this format:



The address option is always IL. Bits 6 to 1 contain the scratchpad address which was used in the IT logand initiating the interrupt subroutine. The effect of the TM logand is then as follows:

(a) The contents of the M register are incremented by one and exchanged with the contents of memory location ICA+1 (or S+1).

(b) The contents of memory location ICA (or S) are placed in the L register.

(c) The next logand is taken from the address initially held in ICA+1 (or S+1), i.e. from the address of the next sequential

logand of the main program, which was interrupted by the IT logand.

Since the TM logand stores the address of the next logand in sequence in memory location ICA+1 or S+1, the address of the first logand of the interrupt subroutine R may be restored to ICA+1 or S+1 by execution of the TM logand from location R-1.

In other words, the TM logand must immediately precede the first logand of an interrupt subroutine.

The TM logand cycles are shown in table 3-19. The logand requires 18 microseconds.

Table 3-19 TM Logand Cycle

(Illustrated for $(m)_{1-6}$ = ICA+1)

Clock	Operations	L	E	A	P	M	T	Memory
1	wrm	ℓ	(m)	a	p	m	t	
2	mim, eℓ, rdℓ	(m)	(ICA+1)			m+1		0→(ICA+1)
3	eme, wrℓ		m+1			(ICA+1)		m+1→(ICA+1)
4	ℓiℓ, rdℓ		(ICA)					0→(ICA)
5	wrℓ							(ICA)→(ICA)
6	eℓ, rdm	(ICA)	((ICA+1))			(ICA+1)		

CHAPTER IV

SUMMARY
TABLES OF BASIC OPERATIONS

4.1 INTRODUCTION

The material presented in the earlier chapters of this manual is sufficient to enable a programmer to write meaningful sequences of logands. However, the programmer will not normally be concerned with the details of logand cycles, once he understands them, but only with the final results produced by these cycles. For this purpose, the programmer can refer rapidly to tables which define the final results of logand cycles as a function of the coding in the logand. These tables are divided into three groups:

(a) Regular Primary Commands

(b) Special Primary Commands

(c) Secondary Commands

4.2 REGULAR PRIMARY COMMANDS

Tables 4-1 through 4-16 describe the effect of each of the 32 regular primary commands in combination with each allowable option. The effect of the control field on the address options is shown in tables 4-15 and 4-16. The effect is described by showing the contents of registers L, E, A, P, M and T (and the changes to memory, if any) at two points in the logand cycle, as follows:

(1) as of the next-to-last clock pulse of the cycle (clock pulse three for direct options and clock pulse five for indirect address options)

(2) as of the last clock pulse of the cycle. This partitioning of the cycle is required by the fact that the secondary command in general affects the register contents on the last clock

pulse. For this reason, the cycles shown assume a secondary command of NO (no operation).

Register contents are shown as functions of an initial set of register contents which are assumed to be ℓ, (m), a, p, m, and t for the L, E, A, P, M and T registers, respectively. A register's contents are shown wherever they are defined by the given coding, and are left blank where they are not so defined. An asterisk denotes that the stated contents apply only if a conflicting secondary command is not given. The rules for resolving such conflicts are given in Chapter III.

4.3 SPECIAL PRIMARY COMMANDS

The effect of special logand cycles is described by showing the register contents and the changes to memory at the end of the logand cycle. In addition, a brief description is given of selected special commands. Initial register contents are again assumed to be ℓ, (m), a, p, m and t. The condition variables and the detailed I/O information are also described in paragraph 4.6.

4.4 SECONDARY COMMANDS

The effect of a secondary command is described in table 4-17 by a summary of the basic operations selected by the command. This table is essentially the same as table 3-3, Secondary Command List, in Chapter III.

4.5 USE OF TABLES

To illustrate the use of the tables: suppose a logand sequence must be constructed which moves the negative (two's complement) of a signed number in memory location SAM to the pseudo accumulator ACC* in scratchpad; that is

$$-(SAM) \rightarrow (ACC)$$

*ACC refers to scratchpad address $AL

Assume that the address SAM is initially in the P register; that ACC is addressed by the L register; and that the first logand of the required sequence is located in memory location X. This initial configuration may be displayed as follows:

LOC	LOGAND				L	E	A	P	M	T
	PC	AO	CF	SC/ADDRESS	Q1	(X)	Q2	SAM	X	Q3

The initial contents of L, A, and T are arbitrary and are given arbitrary symbols Q_1, Q_2 and Q_3.

The chosen sequence must perform the following steps:

(1) Bring the contents of SAM into the A register.

(2) Complement the A register and set the carry indicator to 1.

(3) Add zero to A (to form the two's complement).

(4) Store the contents of A in ACC.

Step (1) may be accomplished with the primary command LA (Load A), the address option DP (Direct P), and the count field C (normal access). Table 4-1 shows that this combination produces the following results:

	L	E	A	P	M	T
LA DP C	(m)	(p)	(p)	m+1	p	t
	(p)	(m+1)		p+1*	m+1	

Since p is equivalent to SAM, the A register now contains (SAM). Step 2 of the procedure can now be implemented with the secondary command CS (complement set), which has the following effect:

				L	E	A	P	M	T	
LA	DP	C	CS	(m)	(p)	(p)	m+1	p	t	
				(p)	(m+1)	(p)'	p+1	m+1	t	1 → (c)

Note that P and T are not affected by the secondary command.

Substituting the equivalents

$$m = x$$

$$p = SAM$$

into the general results, we get the result of the first logand:

	Logand				L	E	A	P	M	T	
	PC	AO	CF	SC/ADDRESS	Q_1	(X)	Q_2	SAM	X	Q_3	
X	LA	DP	C	CS	(SAM)	(X+1)	(SAM)'	SAM+1	X+1	Q_3	1 → (c)

Step (3) of the procedure may now be carried out by the primary command ZE (clear E), the secondary command AI (add intermediate word), the address option DM (direct M), and count field B. With the aforementioned method, this combination has the effect shown in tables 4-12 and 4-14.

				L	E	A	P	M	T
ZE	DM	B	AI	(m)	O	a	p	m+1	t
				O	(m+2)	a+o+	(c)p	m+1	t

Making the substitutions

$m = x + 1$

$a = (SAM)'$

$p = SAM + 1$

we get the results of the second logand:

Logand					L	E	A	P	M	T	
LOC	PC	AO	CF	SC/ADDRESS	Q_1	(x)	Q_2	SAM	X	Q_3	
X	LA	DP	C	CS	(SAM)	(x+1)	SAM'	SAM+1	X+1	Q_3	1→(c)
X+1	ZE	DM	B	AI	O	(x+2)	-(SAM)	SAM+1	X+2	Q_3	

The final step of the procedure is carried out by the primary command SA (store A), address option DL (direct L), and count field C. Table 4-7 shows that this combination produces the following results:

				L	E	A	P	M	T	
SA	DL	C	ACC	(m)	a	a	p	m+1	t	$a \rightarrow (L_{1-6})$
				a	(m+1)					

ACC in this case is an address in memory. Substituting

$m = x + 2$

$a = -(SAM)$

$p = SAM + 1$

the results of the logand are as follows:

	LOGAND	L	E	A	P	M	T	
LOC	PC AO CF SC/ADDRESS	Q_1	(X)	Q_2	SAM	X	Q_3	
X	LA DP C CS	(SAM)	(x+1)	(SAM)'	SAM+1	X+1	Q_3	1 → (c)
X+1	ZE DM B AI	O	(x+2)	-(SAM)	SAM+1	X+2	Q_3	
X+2	SA DL C ACC	-(SAM)	(x+3)	-(SAM)	SAM+1	X+3	Q_3	-(SAM)→(ACC)

Table 4-1. Primary Commands NO (No Operation) and LA (Load A) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
NO	DL		S	(m)	(S)	a	p	m+1	t	
				(S)	(m+1)	a	p	m+1	t	
NO	DM	C		(m)	(m+1)	a	p	m+1	t	
				(m+1)	(m+2)			m+2		
NO	DP	C		(m)	(p)	a	m+1	p	t	
				(p)	(m+1)		p+1	m+1		
NO	DA	C		(m)	(a)	p	m+1	a	t	
				(a)	(m+1)		a+1	m+1		
NO	IL		S	(m)	((S))	a	m+1	(S)	t	
				((S))	(m+1)	a	(S)+1	m+1	t	
NO	IM	C		(m)	((m+1))	a	m+2	(m+1)	t	
				((m+1))	(m+2)		(m+1)+1	m+2		
NO	IP	C		(m)	((p))	p+1	m+1	(p)	t	
				((p))	(m+1)	(p)+1*	p+1	m+1		
NO	IA	C		(m)	((a))	a+1	m+1	(a)	t	
				((a))	(m+1)	(a)+1*	a+1	m+1		

*Applies only if secondary command does not conflict.

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
LA	DL		S	(m)	(S)	(S)	p	m+1	t	
				(S)	(m+1)	(S)	p	m+1	t	
LA	DM	C		(m)	(m+1)	(m+1)	p	m+1	t	
				(m+1)	(m+2)			m+2		
LA	DP	C		(m)	(p)	(p)	m+1	p	t	
				(p)	(m+1)		p+1	m+1		
LA	DA	C		(m)	(a)	(a)	m+1	a	t	
				(a)	(m+1)		a+1	m+1		
LA	IL		S	(m)	((S))	((S))	m+1	(S)	t	
				((S))	(m+1)	((S))	(S)+1	m+1	t	
LA	IM	C		(m)	((m+1))	((m+1))	m+2	(m+1)	t	
				((m+1))	(m+2)		(m+1)+1	m+2		
LA	IP	C		(m)	((p))	((p))	m+1	(p)	t	
				((p))	(m+1)	(p)+1*	((p))	m+1		
LA	IA	C		(m)	((a))	((a))	m+1	(a)	t	
				((a))	(m+1)	(a)+1*	((a))	m+1		

Table 4-2. Primary Commands LP (Load P) and LT (Load T) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
LP	DL		S	(m) (S)	(S) (m+1)	a a	(S) (S)	m+1 m+1	t t	
LP	DM	C		(m) (m+1)	(m+1) (m+2)	a	(m+1)	m+1 m+2	t	
LP	DP	C		(m) (p)	(p) ((p))	a	(p) p+1	p (p)	t	
LP	DA	C		(m) (a)	(a) ((a))	p	(a) a+1	a (a)	t	
LP	IL		S	(m) ((S))	((S)) (((S)))	a a	((S)) (S)+1	(S) ((S))	t t	
LP	IM	C		(m) ((m+1))	((m+1)) (((m+1)))	a	((m+1)) (m+1)+1	(m+1) ((m+1))	t	
LP	IP	C		(m) ((p))	((p)) (((p)))	p+1 (p)+1*	((p)) p+1	(p) ((p))	t	
LP	IA	C		(m) ((a))	((a)) (((a)))	a+1 (a)+1*	((a)) a+1	(a) ((a))	t	

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
LT	DL		S	(m) (S)	(S) (m+1)	a a	p p	m+1 m+1	(S) (S)	
LT	DM	C		(m) (m+1)	(m+1) (m+2)	a	p	m+1 m+2	(m+1)	
LT	DP	C		(m) (p)	(p) (m+1)	a	m+1 p+1	p m+1	(p)	
LT	DA	C		(m) (a)	(a) (m+1)	p	m+1 a+1	a m+1	(a)	
LT	IL		S	(m) ((S))	((S)) (m+1)	a a	m+1 (S)+1	(S) m+1	((S)) ((S))	
LT	IM	C		(m) ((m+1))	((m+1)) (m+2)	a	m+1 (m+1)+1	(m+1) m+2	((m+1))	
LT	IP	C		(m) ((p))	((p)) (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	((p))	
LT	IA	C		(m) ((a))	((a)) (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	((a))	

Table 4-3. Primary Commands LM (Load M) and RA (Replace A) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
LM	DL		S	(m)	(S)	a	p	(S)	t	
				(S)	((S))	a	p	(S)	t	
LM	DM	C				NOT ALLOWED				
LM	DP	C				NOT ALLOWED				
LM	DA	C				NOT ALLOWED				
LM	IL		S			NOT ALLOWED				
LM	IM	C				NOT ALLOWED				
LM	IP	C				NOT ALLOWED				
LM	IA	C				NOT ALLOWED				

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
RA	DL		S	(m)	a	(S)	p	m+1	t	a → (S)
				a	(m+1)	(S)	p	m+1	t	
RA	DM	C		(m)	a	(m+1)	p	m+1	t	a → (m+1)
				a	(m+2)			m+2		
RA	DP	C		(m)	a	(p)	m+1	p	t	a → (p)
				a	(m+1)		p+1	m+1		
RA	DA	C		(m)	p	(a)	m+1	a	t	p → (a)
				p	(m+1)		a+1	m+1		
RA	IL		S	(m)	a	((S))	m+1	(S)	t	a → ((S))
				a	(m+1)	((S))	(S)+1	m+1	t	
RA	IM	C		(m)	a	((m+1))	m+2	(m+1)	t	a → ((m+1))
				a	(m+2)		(m+1)+1	m+2		
RA	IP	C		(m)	p+1	((p))	m+1	(p)	t	p+1 → ((p))
				p+1	(m+1)	(p)+1*	((p))	m+1		
RA	IA	C		(m)	a+1	((a))	m+1	(a)	t	a+1 → ((a))
				a+1	(m+1)	(a)+1*	((a))	m+1		

Table 4-4. Primary Commands RP (Replace P) and RT (Replace T) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
RP	DL		S	(m)	p	a	(S)	m+1	t	p → (S)
				p	(m+1)	a	(S)	m+1	t	
RP	DM	C		(m)	p	a	(m+1)	m+1	t	p → (m+1)
				p	(m+2)			m+2		
RP	DP	C		(m)	m+1	a	(p)	p	t	m+1 → (p)
				m+1	((p))		p+1	(p)		
RP	DA	C		(m)	m+1	p	(a)	a	t	m+1 → (a)
				m+1	((a))		a+1	(a)		
RP	IL		S	(m)	m+1	a	((S))	(S)	t	m+1 → ((S))
				m+1	(((S)))	a	(S)+1	((S))	t	
RP	IM	C		(m)	m+2	a	((m+1))	(m+1)	t	m+2 → ((m+1))
				m+2	(((m+1)))		(m+1)+1	((m+1))		
RP	IP	C		(m)	m+1	p+1	((p))	(p)	t	m+1 → ((p))
				m+1	(((p)))	(p)+1*	p+1	((p))		
RP	IA	C		(m)	m+1	a+1	((a))	(a)	t	m+1 → ((a))
				m+1	(((a)))	(a)+1*	a+1	((a))		

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
RT	DL		S	(m)	t	a	p	m+1	(S)	t → (S)
				t	(m+1)	a	p	m+1	(S)	
RT	DM	C		(m)	t	a	p	m+1	(m+1)	t → (m+1)
				t	(m+2)			m+2		
RT	DP	C		(m)	t	a	m+1	p	(p)	t → (p)
				t	(m+1)		p+1	m+1		
RT	DA	C		(m)	t	p	m+1	a	(a)	t → (a)
				t	(m+1)		a+1	m+1		
RT	IL		S	(m)	t	a	m+1	(S)	((S))	t → ((S))
				t	(m+1)	a	(S)+1	m+1	((S))	
RT	IM	C		(m)	t	a	m+2	(m+1)	((m+1))	t → ((m+1))
				t	(m+2)		(m+1)+1	m+2		
RT	IP	C		(m)	t	p+1	m+1	(p)	((p))	t → ((p))
				t	(m+1)	(p)+1*	p+1	m+1		
RT	IA	C		(m)	t	a+1	m+1	(a)	((a))	t → ((a))
				t	(m+1)	(a)+1*	a+1	m+1		

Table 4-5. Primary Commands RM (Replace M) and AP (Exchange A and P) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
RM	DL		S	(m)	m+1	a	p	(S)	t	m+1 → (S)
				m+1	((S))	a	p	(S)	t	
RM	DM	C				NOT ALLOWED				
RM	DP	C				NOT ALLOWED				
RM	DA	C				NOT ALLOWED				
RM	IL		S			NOT ALLOWED				
RM	IM	C				NOT ALLOWED				
RM	IP	C				NOT ALLOWED				
RM	IA	C				NOT ALLOWED				

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
AP	DL		S	(m)	(S)	p	a	m+1	t	
				(S)	(m+1)	p	a	m+1	t	
AP	DM	C		(m)	(m+1)	p	a	m+1	t	
				(m+1)	(m+2)			m+2		
AP	DP	C		(m)	(p)	m+1	a	p	t	
				(p)	(a)		p+1	a		
AP	DA	C		(m)	(a)	m+1	p	a	t	
				(a)	(p)		a+1	p		
AP	IL		S	(m)	((S))	m+1	a	(S)	t	
				((S))	(a)	m+1	(S)+1	a	t	
AP	IM	C		(m)	((m+1))	m+2	a	(m+1)	t	
				((m+1))	(a)		(m+1)+1	a		
AP	IP	C		(m)	((p))	m+1	p+1	(p)	t	
				((p))	(p+1)	(p)+1*	m+1	p+1		
AP	IA	C		(m)	((a))	m+1	a+1	(a)	t	
				((a))	(a+1)	(a)+1*	m+1	a+1		

Table 4-6. Primary Commands AT (Exchange A and T) and SE (Store E) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
AT	DL		S	(m) (S)	(S) (m+1)	t t	p p	m+1 m+1	a a	
AT	DM	C		(m) (m+1)	(m+1) (m+2)	t	p	m+1 m+2	a	
AT	DP	C		(m) (p)	(p) (m+1)	t	m+1 p+1	p m+1	a	
AT	DA	C		(m) (a)	(a) (m+1)	t	m+1 a	a m+1	p	
AT	IL		S	(m) ((S))	((S)) (m+1)	t t	m+1 (S)+1	(S) m+1	a a	
AT	IM	C		(m) ((m+1))	((m+1)) (m+2)	t	m+2 (m+1)+1	(m+1) m+2	a	
AT	IP	C		(m) ((p))	((p)) (m+1)	t (p)+1*	m+1 t	(p) m+1	p+1	
AT	IA	C		(m) ((a))	((a)) m+1	t (a)+1*	m+1 t	(a) m+1	a+1	

Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
SE	DL		S	(m) ℓ	ℓ (m+1)	a a	p p	m+1 m+1	t t	ℓ → (S)
SE	DM	C		(m) ℓ	ℓ (m+2)	a	p	m+1 m+2	t	ℓ → (m+1)
SE	DP	C		(m) ℓ	ℓ (m+1)	a	m+1 p+1	p m+1	t	ℓ → (p)
SE	DA	C		(m) ℓ	ℓ (m+1)	p	m+1 a+1	a m+1	t	ℓ → (a)
SE	IL		S	(m) p	p (m+1)	a a	m+1 (S)+1	(S) m+1	t t	p → ((S))
SE	IM	C		(m) p	p (m+2)	a	m+2 (m+1)+1	(m+1) m+2	t	p → ((m+1))
SE	IP	C		(m) a	a (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	t	a → ((p))
SE	IA	C		(m) p	p (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	t	p → ((a))

Table 4-7. Primary Commands SA (Store A) and SP (Store P) Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
SA	DL		S	(m) a	a (m+1)	a a	p p	m+1 m+1	t	a → (S)
SA	DM	C		(m) a	a (m+2)	a	p	m+1 m+2	t	a → (m+1)
SA	DP	C		(m) a	a (m+1)	a	m+1 p+1	p m+1	t	a → (p)
SA	DA	C		(m) p	p (m+1)	p	m+1 a+1	a m+1	t	p → (a)
SA	IL		S	(m) a	a (m+1)	a a	m+1 (S)+1	(S) m+1	t t	a → ((S))
SA	IM	C		(m) a	a (m+2)	a	m+2 (m+1)+1	(m+1) m+2	t	a → ((m+1))
SA	IP	C		(m) p+1	p+1 (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	t	p+1 → ((p))
SA	IA	C		(m) a+1	a+1 (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	t	a+1 → ((a))

Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
SP	DL		S	(m) p	p (m+1)	a a	p p	m+1 m+1	t t	p → (S)
SP	DM	C		(m) p	p (m+2)	a	p	m+1 m+2	t	p → (m+1)
SP	DP	C		(m) m+1	m+1 (m+1)	a	m+1 p+1	p m+1	t	m+1 → (p)
SP	DA	C		(m) m+1	m+1 (m+1)	p	m+1 a+1	a m+1	t	m+1 → (a)
SP	IL		S	(m) m+1	m+1 (m+1)	a a	m+1 (S)+1	(S) m+1	t t	m+1 → ((S))
SP	IM	C		(m) m+2	m+2 (m+2)	a	m+2 (m+1)+1	(m+1) m+2	t	m+2 → ((m+1))
SP	IP	C		(m) m+1	m+1 (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	t	m+1 → ((p))
SP	IA	C		(m) m+1	m+1 (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	t	m+1 → ((a))

Table 4-8. Primary Commands ST (Store T) and HM (Hold M) Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
ST	DL		S	(m) t	t (m+1)	a a	p p	m+1 m+1	t t	t → (S)
ST	DM	C		(m) t	t (m+2)	a	p	m+1 m+2	t	t → (m+1)
ST	DP	C		(m) t	t (m+1)	a	m+1 p+1	p m+1	t	t → (p)
ST	DA	C		(m) t	t (m+1)	p	m+1 a+1	a m+1	t	t → (a)
ST	IL		S	(m) t	t (m+1)	a a	m+1 (S)+1	(S) m+1	t t	t → ((S))
ST	IM	C		(m) t	t (m+2)	a	m+2 (m+1)+1	(m+1) m+2	t	t → ((m+1))
ST	IP	C		(m) t	t (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	t	t → ((p))
ST	IA	C		(m) t	t (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	t	t → ((a))

Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
HM	DL		S	(m) m+1	m+1 (ℓ)	a a	p p	ℓ ℓ	t t	m+1 → (S)
HM	DM	C				NOT ALLOWED				
HM	DP	C				NOT ALLOWED				
HM	DA	C				NOT ALLOWED				
HM	IL		S			NOT ALLOWED				
HM	IM	C				NOT ALLOWED				
HM	IP	C				NOT ALLOWED				
HM	IA	C				NOT ALLOWED				

Table 4-9. Primary Commands HA (Hold A) and HP (Hold P) Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
HA	DL		S	(m)	a	ℓ	p	m+1	t	a → (S)
				a	(m+1)	ℓ	p	m+1	t	
HA	DM	C		(m)	a	ℓ	p	m+1	t	a → (m+1)
				a	(m+2)			m+2		
HA	DP	C		(m)	a	ℓ	m+1	p	t	a → (p)
				a	(m+1)		p+1	m+1		
HA	DA	C		(m)	p	ℓ	m+1	a	t	p → (a)
				p	(m+1)		a+1	m+1		
HA	IL		S	(m)	a	p	m+1	(S)	t	a → ((S))
				a	(m+1)	p	(S)+1	m+1	t	
HA	IM	C		(m)	a	p	m+2	(m+1)	t	a → ((m+1))
				a	(m+2)		(m+1)+1	m+2		
HA	IP	C		(m)	p+1	a	m+1	(p)	t	p+1 → ((p))
				p+1	(m+1)	(p)+1*	a	m+1		
HA	IA	C		(m)	a+1	p	m+1	(a)	t	a+1 → ((a))
				a+1	(m+1)	(a)+1*	p	m+1		

Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
HP	DL		S	(m)	p	a	ℓ	m+1	t	p → (S)
				p	(m+1)	a	ℓ	m+1	t	
HP	DM	C		(m)	p	a	ℓ	m+1	t	p → (m+1)
				p	(m+2)			m+2		
HP	DP	C		(m)	m+1	a	ℓ	p	t	m+1 → (p)
				m+1	(ℓ)		p+1	ℓ		
HP	DA	C		(m)	m+1	p	ℓ	a	t	m+1 → (a)
				m+1	(ℓ)		a+1	ℓ		
HP	IL		S	(m)	m+1	a	p	(S)	t	m+1 → ((S))
				m+1	(p)	a	(S)+1	p	t	
HP	IM	C		(m)	m+2	a	p	(m+1)	t	m+2 → ((m+1))
				m+2	(p)		(m+1)+1	p		
HP	IP	C		(m)	m+1	p+1	a	(p)	t	m+1 → ((p))
				m+1	(a)	(p)+1*	p+1	a		
HP	IA	C		(m)	m+1	a+1	p	(a)	t	m+1 → ((a))
				m+1	(p)	(a)+1*	a+1	p		

Table 4-10. Primary Command HT (Hold T) Type: Store

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
HT	DL		S	(m) t	t (m+1)	a a	p p	m+1 m+1	ℓ ℓ	t → (S)
HT	DM	C		(m) t	t (m+2)	a	p	m+1 m+2	ℓ	t → (m+1)
HT	DP	C		(m) t	t (m+1)	a	m+1 p+1	p m+1	ℓ	t → (p)
HT	DA	C		(m) t	t (m+1)	p	m+1 a+1	a m+1	ℓ	t → (a)
HT	IL			(m) t	t (m+1)	a a	m+1 (S)+1	(S) m+1	p p	t → ((S))
HT	IM	C		(m) t	t (m+2)	a	m+2 (m+1)+1	(m+1) m+2	p	t → ((m+1))
HT	IP	C		(m) t	t (m+1)	p+1 (p)+1*	m+1 p+1	(p) m+1	a	t → ((p))
HT	IA	C		(m) t	t (m+1)	a+1 (a)+1*	m+1 a+1	(a) m+1	p	t → ((a))

Table 4-11. Primary Commands XA,MA,CS,CC, and CH Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
	DL		S	(m)	(S)	f	p	m+1	t	(E) = (S)
				(S)	(m+1)	f	p	m+1	t	(A) = a
	DM	C		(m)	(m+1)	f	p	m+1	t	(E) = (m+1)
				(m+1)	(m+2)			m+2		(A) = a
	DP	C		(m)	(p)	f	m+1	p	t	(E) = (p)
				(p)	(m+1)		p+1	m+1		(A) = a
	DA	C		(m)	(a)	f	m+1	a	t	(E) = (a)
				(a)	(m+1)		a+1	m+1		(A) = p
	IL		S	(m)	((S))	f	m+1	(S)	t	(E) = ((S))
				((S))	(m+1)	f	(S)+1	m+1	t	(A) = a
	IM	C		(m)	((m+1))	f	m+2	(m+1)	t	(E) = ((m+1))
				((m+1))	(m+2)		(m+1)+1	m+2		(A) = a
	IP	C		(m)	((p))	f	m+1	(p)	t	(E) = ((p))
				((p))	(m+1)	(p)+1*	f	m+1		(A) = p+1
	LA	C		(m)	((a))	f	m+1	(a)	t	(E) = ((a))
				((a))	(m+1)	(a)+1*	f	m+1		(A) = a+1

Notes:

Primary Command		f	Remarks
XA	(Extract to A)	(E) • (A)'	
MA	(Merge to A)	(E) V (A)	
CS	(Complement and Set)	(A)'	Carry Indicator Set to 1
CC	(Complement and Clear)	(A)'	" " " " 0
CH	(Complement and Hold)	(A)'	" " is held

Table 4-12. Primary Commands XE, ME, and ZE

Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
	DL		S	(m)	f	a	p	m+1	t	f → (S)
				f	(m+1)	a	p	m+1	t	(E) = (S)
	DM	C		(m)	f	a	p	m+1	t	f → (m+1)
				f	(m+2)			m+2		(E) = (m+1)
	DP	C		(m)	f	a	m+1	p	t	f → (p)
				f	(m+1)		p+1	m+1		(E) = (p)
	DA	C		(m)	f	p	m+1	a	t	f → (a)
				f	(m+1)		a+1	m+1		(E) = (a)
	IL		S	(m)	f	a	m+1	(S)	t	f → ((S))
				f	(m+1)	a	(S)+1	m+1	t	(E) = ((S))
	IM	C		(m)	f	a	m+2	(m+1)	t	f → ((m+1))
				f	(m+2)		(m+1)+1	m+2		(E) = ((m+1))
	IP	C		(m)	f	p+1	m+1	(p)	t	f → ((p))
				f	(m+1)	(P)+1*	p+1	m+1		(E) = ((p))
	IA	C		(m)	f	a+1	m+1	(a)	t	f → ((a))
				f	(m+1)	(a)+1*	a+1	m+1		(E) = ((a))

Notes:

Primary Command		F
XE	(Extract to E)	(E).(A)
ME	(Merge to E)	(E) V (A)
ZE	(Clear E)	0

Result of F is seen at 3 or 5 time

Table 4-13. Primary Command DX (Double Extract) Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
DX	DL		S	(m)	f	g	p	m+1	t	
				f	(m+1)	g	p	m+1	t	
DX	DM	C		(m)	f	g	p	m+1	t	
				f	(m+2)			m+2		
DX	DP	C		(m)	f	g	m+1	p	t	
				f	(m+1)		p+1	m+1		
DX	DA	C		(m)	f	g	m+1	a	t	
				f	(m+1)		a+1	m+1		
DX	IL		S	(m)	f	g	m+1	(S)	t	
				f	(m+1)	g	(S)+1	m+1	t	
DX	IM	C		(m)	f	g	m+2	(m+1)	t	
				f	(m+2)		(m+1)+1	m+2		
DX	IP	C		(m)	f	g	m+1	(p)	t	
				f	(m+1)	(p)+1*	g	m+1		
DX	IA	C		(m)	f	g	m+1	(a)	t	
				f	(m+1)	(a)+1*	g	m+1		

Note

Function:

F = (E) · (A) , g = (E) · (A)'

Table 4-14. Primary Commands AS, AL, AI, and AM Type: Load

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/Remarks
	DL		S	(m)	(S)	a	p	m+1		
				(S)	(m+1)	f	p	m+1	t	(E) = (S)
	DM	C	NO	(m)	(m+1)	a	p	m+1	t	
				(m+1)	(m+2)	f	p	m+2	t	(E) = (m+1)
	DP	C	NO	(m)	(p)	a	m+1	p	t	
				(p)	(m+1)	f	p+1	m+1	t	(E) = (p)
	DA	C	NO	(m)	(a)	p	m+1	a	t	(A) = p
				(a)	(m+1)	f	a+1	m+1	t	(E) = (a)
	IL		S	(m)	((S))	a	m+1	(S)	t	
				((S))	(m+1)	f	(S)+1	m+1	t	(E) = ((S))
	IM	C	NO	(m)	((m+1))	a	m+2	(m+1)	t	
				((m+1))	(m+2)	f	(m+1)+1	m+2	t	(E) = ((m+1))
	IP	C	NO	(m)	((p))	p+1	m+1	(p)	t	(A) = p+1
				((p))	(m+1)	f	p+1	m+1	t	(E) = ((p))
	IA	C	NO	(m)	((a))	a+1	m+1	(a)	t	(A) = a+1
				((a))	(m+1)	f	a+1	m+1	t	(E) = ((a))

Notes:

Primary Command	F	Remarks
AS (Add Single Word)	(A) + (E)	Overflow Possible
AL (Add Least Significant Word)	(A) + (E)	
AI (Add Intermediate Word)	(A) + (E) + C	
AM (Add Most Significant Word)	(A) + (E) + C	Overflow Possible

Table 4-15. Primary Command NO (No Operation) Type: Load - Hold Count

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
NO	DM	H		(m) (m+1)	(m+1) (m+1)	a	p	m+1 m+1	t	M not counted at 4 time
NO	DP	H		(m) (p)	(p) (m+1)	a	m+1 p	p m+1	t	p not counted at 4 time
NO	DA	H		(m) (a)	(a) (m+1)	p	m+1 a	a m+1	t	a not counted at 4 time
NO	IM	H		(m) ((m+1))	((m+1)) (m+2)	a	m+2 (m+1)	(m+1) m+2	t	(m+1) not counted at 6 time
NO	IP	H		(m) ((p))	((p)) (m+1)	p+1 (p)*	m+1 p+1	(p) m+1	t	(p) not counted at 6 time
NO	IA	H		(m) ((a))	((a)) (m+1)	a+1 (a)*	m+1 a+1	(a) m+1	t	(a) not counted at 6 time

Type: No Access-Count

PC	AO	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
NO	DM	N		(m) *l*	*l* (m+2)	a	p	m+1 m+2	t	No access at 2 time
NO	DP	N		(m) *l*	*l* (m+1)	a	m+1 p+1	p m+1	t	No access at 2 time
NO	DA	N		(m) *l*	*l* (m+1)	p	m+1 a+1	a m+1	t	No access at 2 time
NO	IM	N		(m) (m+1)	(m+1) (m+2)	a	m+2 (m+1)+1	(m+1) m+2	t	No access at 4 time
NO	IP	N		(m) (p)	(p) (m+1)	p+1 (p)+1	m+1 p+1	(p) m+1	t	No access at 4 time
NO	IA	N		(m) (a)	(a) (m+1)	a+1 (a)+1	m+1 a+1	(a) m+1	t	No access at 4 time

Table 4-16. Primary Command NO (No Operation) Type: Hold Count - No Access

PC	AP	CF	SC/ ADD	L	E	A	P	M	T	Memory/ Remarks
NO	DM	B		(m) **ℓ**	**ℓ** (m+1)	**a**	p	m+1 m+1	t	No count at 4 No access at 2
NO	DP	B		(m) **ℓ**	**ℓ** (m+1)	**a**	m+1 p	p m+1	t	" "
NO	DA	B		(m) **ℓ**	**ℓ** (m+1)	p	m+1 **a**	**a** m+1	t	" "
NO	IM	B		(m) (m+1)	(m+1) (m+2)	**a**	m+2 (m+1)	(m+1) m+2	t	No count at 6 No access at 4
NO	IP	B		(m) (p)	(p) (m+1)	p+1 (p)*	m+1 p+1	(p) m+1	t	" "
NO	IA	B		(m) (a)	(a) (m+1)	**a**+1 (a)*	m+1 **a**+1	(**a**) m+1	t	" "

Table 4-17. Secondary Commands

Octal Code L_{1-4}	Symbolic Code	Operations	
		Clock Pulse Three or Five	Clock Pulse Four or Six
00	NO	NONE	e → e
15	LA	NONE	e → a
02	LP	NONE	e → p
10	LT	NONE	e → t
12	LM	NONE	e → m
06	AP	NONE	a ↔ p
04	AT	NONE	a ↔ t
16	XA	NONE	e·a' → a
14	MA	NONE	e v a → a
13	AS	0 → (C)	a + e → a overflow enabled
03	AL	0 → (C)	a + e → a
01	AI	NONE	a + e + (C) → a
11	AM	NONE	a + e + (C) → a overflow enabled
05	CS	NONE	a' → a, 1 → (C)
07	CC	NONE	a' → a, 0 → (C)
17	CH	NONE	a' → a

4.6 NOTES ON USE OF SPECIAL PRIMARY COMMANDS

Multiply - MP

The E register contains the multiplicand, the A register contains the initial partial/product and the P register initially contains the multiplier.

The final product is in A and the high order n bits of P.

Multiply - MS

Same as for MP except that the sign of the multiplicand is propagated in A.

Divide - DV

The E register contains the negative of a positive divisor; the A register contains the most significant part of the positive dividend; and P contains the least significant part of the dividend.

The quotient is in P and the remainder scaled 2^n is in A and the high order 15-n bits of P.

Move - MV

This logand moves a block of sequential words from a to p. The operation is terminated when (a+n) is equal to the one's complement of t. To transfer n words, set t = (a+n)', and the cells a through a + (n-1) are moved.

For SA's other than NV, a decision to terminate can be made by the EN, EQ, NQ, NH, and NL conditionals.

Match - MH

This logand compares two sequences or words starting with a and p+1. It terminates only when the desired comparison is made. Available comparison criteria are (a+k) EQ, NQ, NH, or NL to (p+k+1).

Table Look-Up - TB

This logand compares a sequence of words starting at a to a number in T. The operation terminates when the desired comparison is achieved or if a+n is equal to the one's complement of p. If termination is due to address equality, the overflow indicator is set to 1; if due to a data comparison the indicator is unchanged. Available comparisons are EQ, NQ, NL, or NH.

Sort - SR

This logand compares a sequence starting with a to a word at p. It also compares each word in the sequence to each succeeding word in the sequence. It can terminate only on the conditions EQ, NQ, NH, or NL. If it terminates on the (a+k) and (a+k-1) comparison, the overflow indicator is set. If it terminates with an (a+k) and (p) comparison, the indicator is left as is.

Control Function - CF

This logand deals with three functions:

(a) Channel selections

(b) Interrupt control

(c) Control panel input

The octal coding of bits 1 through 6 is:

Octal Code	Function
10 or 11	Select cable C-in and C-out.
12	Select cable C-in and B-out.
13	Select cable C-in and A-out.
14	Select cable B-in and C-out.
15	Select cable A-in and C-out.
16	Select cable B-in and B-out.
17	Select cable A-in and A-out.
20	Clear parity error indicator.
21	Input interrupt status word to the E register.
22	Input control panel switches to the E register.
24	Reset momentary interrupt flip-flop.
40	Input a 15-bit interrupt status word from cable C-in to the E register.
43	Input a 30-bit interrupt status word from cable A-in to the E and T registers.
46	Input a 30-bit interrupt status word from cable B-in to the E and T registers.

The coding of the type II input interrupt status word (octal code 21) is as follows:

Bit 1 - The continuous interrupt switch is in the INTERRUPT position.

Bits 2 through 9 - The devices connected to cable C-in.

Bit 10 - The device connected to B-in.

Bit 11 - The device connected to A-in.

Bit 12 - The momentary interrupt button has been pressed.

Bit 13 - The devices connected to cable C-in.

Bits 14 and 15 - Not assigned

External Function - EF

This logand provides for connecting the computer to a device which is connected to a previously selected cable. The coded command word is read into the E register (unless it is a no-access type) and decoded by the I/O unit.

Octal Code	I/O Interpretation
0 1	1st EF logand to cable A-out - Sends most significant 15 bits of a 30-bit word.
0 3	2nd EF logand to cable A-out - Sends least significant 15 bits of a 30-bit word.
0 4	1st EF logand to cable B-out.
0 6	2nd EF logand to cable B-out.
1 0	EF logand to send 15 bits to cable C-out.

It is necessary that two EF logands be executed in succession when cable A or B is used. The first EF logand transmits the most-significant 15 bits of the command code.

Block Input - BI and Block Output - BO

These logands input or output sequentially, starting with a, and terminate when the next address equals the one's complement of p. NV is the conditional code.

Word Input WI and Word Output WO

These logands input or output a word according to the address option. The register results are the same as for a NO (no operation) primary command. In the case of a 30-bit word transfer, the I/O unit performs additional E to T transfers and stores the 30-bit input word according to the address option. The secondary command is always NO.

Interrupt - IT

This logand provides for exit to a specific core location under program or I/O control.

The logand is stored in cell 1 for type II interrupts, in cell 5 for type I output data request interrupts, and in cell 7 for type I input data request interrupts. When called on to execute, the contents of E are stored in 0, 4, or 6, respectively, and the contents of M are stored in 1, 5, or 7, respectively.

Terminate - TM

This logand provides for resuming operation from the point of interrupt. The logand should be executed from R-1 (where R is the first logand in the interrupt routine) and should have an address of X+ 1 in the address field which had R as its contents before the interrupt.

Basic Shifts, Branch and Skip

See Chapter III for additional information.

Table 4-18. Special Primary Commands BR (Branch) and SK (Skip)

PC	AO	CF	SC/ L5-1	L	E	A	P	M	T	Remarks		Exec. Time (μs)
BR	DM	* F	(Condition)	(m+1)	(m+2)	a	p	m+2	t	Condition Not Satisfied	(m+1)= Branch Address	12
				(m+1)	((m+1))	a	p	(m+1)	t	Condition Satisfied		
BR	DP	F	(Condition)	(p)	(m+1)	a	p+1	m+1	t	Condition Not Satisfied	(p)= Branch Address	12
				(p)	((p))	a	p+1	(p)	t	Condition Satisfied		
BR	DA	F	(Condition)	(a)	(m+1)	p	a+1	m+1	t	Condition Not Satisfied	(a) = Branch Address	12
				(a)	((a))	p	a+1	(a)	t	Condition Satisfied		
BR	IM	F	(Condition	(m+2)	(m+3)	a	p	m+3	t	Condition Not Satisfied	(m+1)= E comperand	18
				(m+2)	((m+2))	a	p	(m+2)	t	Condition Satisfied	(m+2)= Branch Address	
BR	IP	F	(Condition)	(m+1)	(m+2)	a	p+1	m+2	t	Condition Not Satisfied	(p)= E Comperand	18
				(m+1)	((m+1))	a	p+1	(m+1)	t	Condition Satisfied	(m+1)=Branch Address	
BR	IA	F	(Condition)	(m+1)	(m+2)	p	a+1	m+2	t	Condition Not Satisfied	(a)= E Comperand	18
				(m+1)	((m+1))	p	a+1	(m+1)	t	Condition Satisfied	(m+1)=Branch Address	
SK	IM	F	(Condition)	(m+1)	m+2)	a	p	m+2	t	Condition Not Satisfied	(m+1)=E Comperand	18
				(m+1)	(m+3)	a	p	m+3	t	Condition Satisfied		
SK	IP	F	(Condition)	(p)	(m+1)	a	p+1	m+1	t	Condition Not Satisfied	(p)=E Comperand	18
				(p)	(m+2)	a	p+1	m+2	t	Condition Satisfied		
SK	IA	F	(Condition)	(a)	(m+1)	p	a+1	m+1	t	Condition Not Satisfied	(a)= E Comperand	18
				(a)	(m+2)	p	a+1	m+2	t	Condition Satisfied		

* blank = 0 = Normal

F = 1 = Flag Stop

Table 4-19. Special Primary Commands SO, SC, NR, FL, RC, MP, MS, and DV

PC	AO	CF	L4-1	L	E	A	P	M	T	Remarks	Exec. Time (μs)
SO	DM	*	n	ℓ	(m+1)	(As Shifted)		m+1	t	A or AP shifted open left or right n places	12+3n
SC	DM	*	n	ℓ	(m+1)	(As Shifted)		m+1	t	A or AP shifted closed left or right n places	12+3n
NR	DM DL	SR DR	n	ℓ	(m+1)	(As Shifted)		m+1	t	A or AP shifted numeric right n places	12+3n
FL	DM	SL DL	n	(m)-k	(m+1)	(As Shifted)		m+1	t	A or AP Floated Left k places	12+3n
MP	DM	DR	n	ℓ	(m+1)	(As shifted)		m+1	t	Product in A and high order n bits of P	12+3n
MS	DM	DR	n	ℓ	(m+1)	(As Shifted)		m+1	t	Product in A and high order n bits of P	12+3n
DV	DM	DL	n	ℓ	(m+1)	(As Shifted)		m+1	t	Quotient in low order n bits of P; re-remainder in A, high order 15-n bits of P.	12+3n
RC	DM	SR	n	ℓ	(m+n+1)	a	p	m+n+1	t	m+1 is incremented n times	12+3n
RC	DP	SR	n	ℓ	(m+1)	a	p+n	m+1	t	p is incremented n times	12+3n
RC	DA	SR	n	ℓ	(m+1)	p	a+n	m+1	t	a is incremented n times and ends up in P, p ends up in A	12+3n

* The four control field shift options SR, SL, DR and DL are meaningful.

Table 4-20. Primary Commands: MV, MH, TB, SR, CF, EF, WI, WO, BI, BO, IT, and TM

PC	AO	CF	- 1	L	E	A	P	M	T	Remarks	Exec. Time μs
MV	IP	-	NV,EN,EQ, NQ,NH,NL	(a+ n)	(m+1)	a+ n	p+ n	m+1	t	N words moved from a to p unless condition terminates	18+ 12n
MH	IP	-	NQ,EQ,NH, NL	(a+ n)	(m+1)	a+(n-1)	p+n	m+1	(a+n-1)'	Comparison of two tables starting at a and p+1, terminates on condition satisfied only.	30+ 12n
TB	IP	-	NQ,EQ,NH, NL	(a+n)	(m+1)	a+n	p	m+1	t	Compares a table at a with t, terminates with a=p' or on a condition. Overflow set if address termination.	30+ 12n
SR	IP	-	NQ,EQ,NH, NL	(a+n)	(m+1)	a+n	p	m+1	(a+n-1)'	Compares a table at a with word at p and with each word in the a table. Terminates with (a+i) compares with (p) or with (a+i)and(a+i-1). Overflow set with the latter.	30+ 12n
CF	DL	Code		ℓ	(m+1)	a	p	m+1	t	The I/O unit interprets directly the SA field of the logand	27 Min.
EF	*	Code								(E) Transmitted to I/O control unit as a command	12 or 18 Min.
WI	ANY	-	NO	Effect identical to primary command NO (No Operation)						(E) Replaced or modified by I/O control unit	12 or 18 Min.
WO	ANY	-	NO							(E) Transmitted to I/O control unit	12 or 18 Min
BI	IP	-	NV	(a+ n-1)	(m+1	a+n	p	m+1	t	Inputs sequentially starting at a. Terminates when next address equals p or (a+n)'	18 Min.
BO	IP	-	NV	(a+ n-1)	(m+1)	a+n	p	m+1	t	Outputs sequentially starting at a	18 Min
IT	IL	X		M	((X+1))	a	p	(x+1)	t	ℓ → (X), m → (X+1), the logand is forced by the I/O unit, because of an I/O controlled interrupt	18

Table 4-20. Primary Commands: MV, MH, TB, SR, CF, EF, WI, WO, BI, BO, IT, and TM (Cont.)

PC	AO	CF	L6 - 1	L	E	A	P	M	T	Remarks	Exec. Time µs
It	IL	-	X	m+1	((x+1))	a	p	(x+1)	t	$\ell \rightarrow$ (X), m+1 → (X+1), X is supplied by the program, under programmer control	18
TM	IL	-	X	(X)	((X+1))	a	p	(x+1)	t	Executed from (X)-1, returns registers to state before IT took place	18

*Any option except DL or IL

Table 4-21. Condition Parameters

Condition Parameter Code	Octal Equivalents		Condition Variables
	BR SK	TB MV MH SR	
UN	20		Unconditional, condition always satisfied.
AD	21		Least-significant bit in the A register is one; i.e., A is odd.
PY	25		A parity error has occurred since this condition was last tested.
AZ	24		A register all zeros.
NV	23	03	Never, condition never satisfied.
EQ	26	06	The number in the A register is the one's complement of the number in the E register. The carry flip-flop must be cleared by a complement-clear command before using this condition.
NQ	27	07	The number in the A register is not the one's complement of the number in the E register. The carry flip-flop must be cleared by a complement-clear command before using this condition.
EN	30	10	The most significant bit in the E register is one.
OV	31		An arithmetic overflow has occurred since this condition was last tested.
CY	32		Carry flip-flop holds a one.
TL	33		The T register is being used by the input/output unit.
AN	34		The most-significant bit in the A register is one.

Table 4-21. Condition Parameters (Cont.)

Condition Parameter Code	Octal Equivalents		Condition Variables
	BR SK	TB MV MH SR	
AP	35		The most-significant bit in the A register is zero.
NH	36	16	The one's complement of the number in the A register is algebraically greater than or is equal to, the number in the E register. *
NL	37	17	The one's complement of the number in the A register is algebraically less than the number in the E register. *

*The conditions described are correct if the carry flip-flop is off. If it is on, the equality of E/ with the one's complement of A/ satisfies condition 37, 17 instead of 36, 16. The numbers being compared by these two conditions are assumed, if the most significant bit is one, to be negative in two's-complement form.

Thompson Ramo Wooldridge Inc.

TRW-130 AN/UYK-1

DIGITAL COMPUTER

CF LOGAND CODES

	Oct*	Dec
Terminate Computer Mode	00	0
Establish Computer Mode	01	1
Start Buffer Output	02	2
Start Buffer Input	03	3
Stop Buffer	04	4
EF to Cable "D" (wait)	07	7
Select C IN - C OUT	10	8
Select C IN - B OUT	12	10
Select C IN - A OUT	13	11
Select B IN - C OUT	14	12
Select A IN - C OUT	15	13
Select B IN - B OUT	16	14
Select A IN - A OUT	17	15
Reset Memory Parity Error	20	16
Interrupt Status Word In	21	17
Input Toggles	22	18
Reset Momentary Interrupt	24	20
EFE Buffer Transfer	31	25
FQR Buffer Transfer	32	26
RQF Buffer Transfer	33	27
C Interrupt Code Word In	40	32
"D" Interrupt Word In	41	33
EF to Cable "D" (NTDS)	42	34
A Interrupt Code Word In	43	35
B Interrupt Code Word In	46	38

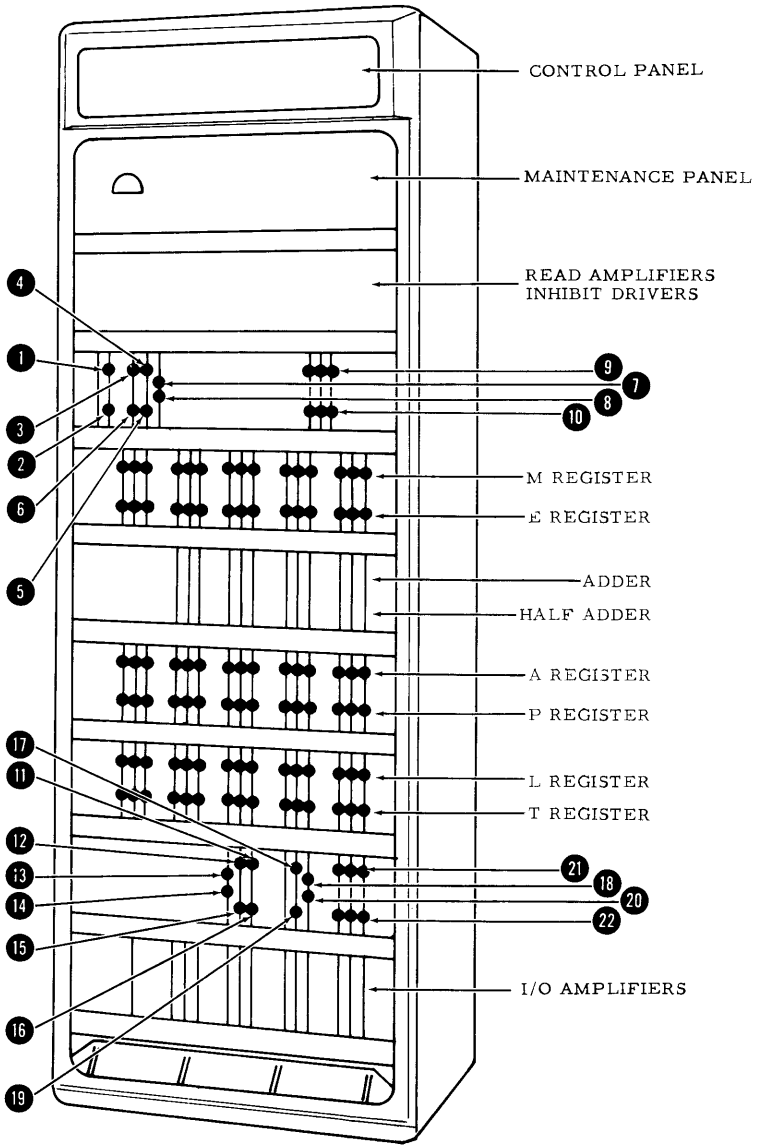
*Codes written for the assembler must be in decimal.

EF LOGAND CODES

I/O Interpretation	Oct Code
Alert Cable A-OUT for command word (most significant 15 bits of 30 bit word)	01
Alert Cable A-OUT for command word (least significant 15 bits of 30 bit word)	03
Alert Cable B-OUT for command word	04
Alert Cable B-OUT for command word	06
Alert Cable C-OUT for command word	10

INTERRUPT STATUS WORD

	Bit
(Unavailable)	15
TRW-173 Buffer	14
(Unused)	13
Momentary Interrupt Switch	12
Cable A	11
Cable B	10
(Unused - Cable C)	9
(Unused - Cable C)	8
(Unused - Cable C)	7
(Unused - Cable C)	6
TRW-80 Display System	5
192/170 Magnetic Tapes	4
Teletype (only)	3
Flexowriter	2
Continuous Interrupt Switch	1



Thompson Ramo Wooldridge Inc.

TRW-130 AN/UYK-1

DIGITAL COMPUTER

NEON INDICATORS

1 PARITY BIT
2 PARITY ERROR
3 CARRY INDICATOR
4 RDM (RDL)
5 CONDITION NOT SATISFIED
6 OVERFLOW
7 AC CARRY
8 POWER ON
9 STATE COUNTER REGISTER
10 CONTROL COUNTER REGISTER

11 COMPUTER IN TYPE II INTERRUPT CONDITION
12 MOMENTARY (PANEL) INTERRUPT
13 MISCELLANEOUS INTERRUPT ON LINE
14 HOLD COUNT
15 T REGISTER LOADED
16 COMPUTER IN TYPE I INTERRUPT CONDITION

OUTPUT DATA REQUEST
17 COMPUTER READY
18 REQUEST ON LINE

INPUT DATA REQUEST
19 COMPUTER READY
20 REQUEST ON LINE

21 CHANNEL DESIGNATOR

In	Out	If A/B
1=A/B 0= C	1=A/B 0= C	1 = A 0 = B

22 IN/OUT TIMING

Normal

TRW-130 *AN/UYK-1*
Thompson Ramo Wooldridge Inc. **DIGITAL COMPUTER**

REGULAR PRIMARY COMMAND LIST

Octal Code	Sym Code	Name
		Load Commands
44	ZE	Clear E
45	RT	Replace T
46	XE	Extract to E
47	RP	Replace P
54	ME	Merge to E
55	RA	Replace A
56	DX	Double Extract
57	RM	Replace M
60	NO	No Operation
61*	AI	Add Intermediate
62	LP	Load P
63*	AL	Add Least
64	AT	Exchange A and T
65	CS	Complement Set
66	AP	Exchange A and P
67	CC	Complement Clear
70	LT	Load T
71*+	AM	Add Most
72	LM	Load M
73*+	AS	Add Single
74	MA	Merge to A
75	LA	Load A
76	XA	Extract to A
77	CH	Complement Hold
		Store Commands
40	ST	Store T
41	HT	Hold T
42	SP	Store P
43	HP	Hold P
50	SA	Store A
51	HA	Hold A
52	SE	Store E
53	HM	Hold M

Alphabetical	
AI	61*
AL	63*
AM	71*+
AP	66
AS	73*+
AT	64
CC	67
CH	77
CS	65
DX	56
HA	51
HM	53
HP	43
HT	41
LA	75
LM	72
LP	62
LT	70
MA	74
ME	54
NO	60
RA	55
RM	57
RP	47
RT	45
SA	50
SE	52
SP	42
ST	40
XA	76
XE	46
ZE	44

* Secondary command will be ignored
\+ Overflow indicator set to 1 if overflow occurs

ADDRESS OPTIONS

Octal Code	Sym Code	Name
0	DM	Direct M
1	DL	Direct L
2	DP	Direct P
3	DA	Direct A
4	IM	Indirect M
5	IL	Indirect L
6	IP	Indirect P
7	IA	Indirect A

SPECIAL PRIMARY COMMAND LIST

Octal Code	Sym Code	Primary Command
00	IT	Interrupt
01	FL	Float Left
03	SC	Shift Closed
05	DV	Divide
07	RC	Repeat Count
10	TM	Terminate Interrupt
11	SO	Shift Open
13	NR	Numeric Right
15	MP	Multiply
17	MS	Multiply Signed
20	BR	Branch
21	EF	External Function
23	WO	Word Output
24	SK	Skip
25	CF	Control Function
27	WI	Word Input
30	MH	Match
32	MV	Move
33	BO	Block Output
34	SR	Sort
36	TB	Table Search
37	BI	Block Input

Alphabetical	
BI	37
BO	33
BR	20
CF	25
DV	05
EF	21
FL	01
IT	00
MH	30
MP	15
MS	17
MV	32
NR	13
RC	07
SC	03
SK	24
SO	11
SR	34
TB	36
TM	10
WI	27
WO	23

SECONDARY COMMAND LIST

Octal Code	Sym Code	Command
00	NO	No Operation
01	AI	Add Intermediate
02	LP	Load P
03	AL	Add Least
04	AT	Exchange A and T
05	CS	Complement Set
06	AP	Exchange A and P
07	CC	Complement Clear
10	LT	Load T
11	AM	Add Most
12	LM	Load M
13	AS	Add Single
14	MA	Merge to A
15	LA	Load A
16	XA	Extract to A
17	CH	Complement Hold

Alphabetical	
AI	01
AL	03
AM	11
AP	06
AS	13
AT	04
CC	07
CH	17
CS	05
LA	15
LM	12
LP	02
LT	10
MA	14
NO	00
XA	16

CONTROL FIELD OPTIONS

Regular Logand			Shift Logand		
Biny Code	Sym Code	Function	Biny Code	Sym Code	Function
00=0	C	Count, Access	00=0	SR	Shift A Right
01=1	H	Hold Count, Access	01=1	SL	Shift A Left
10=2	N	Count, No Access	10=2	DR	Shift A & P Right
11=3	B	Both - Hold Count, No Access	11=3	DL	Shift A & P Left

TRW-130 *AN/UYK-1*
Thompson Ramo Wooldridge Inc. **DIGITAL COMPUTER**

LOGAND FORMATS

L ADDRESS OPTIONS

15 ... 10	9 ... 7	6 ... 1
Reg Pri Command WI WO	DL or IL	Scratchpad Address
IT TM	IL	
CF	DL	Code

M, A, P ADDRESS OPTIONS

15 ... 10	9 ... 7	6 ... 5	4 ... 1
Reg Pri Command	Any	Cntrl Fld	Sec Cmnd
SO SC NR FL MP MS DV	DM		n
RC	Direct	SR	
WI, WO	Any		NO
MV TB MH SR BI BO	IP		Condition
EF	Any		Code
BR		F*	Condition
SK	Indirect		

* FLAG Stop. Optional.

CONDITIONS

Sym Code	Octal Code BR SK	Octal Code TB MH MV SR	Condition
UN	20		Unconditional
AD	21		A Odd
NV	23	03	Never
AZ	24		A Zero
PY	25		Parity Indicator On
EQ*	26	06	E and A Equal
NQ*	27	07	E and A Not Equal
EN	30	10	E Negative
OV	31		Overflow Indicator On
CY	32		Carry Indicator On
TL	33		T Register in Use
AN	34		A Negative
AP	35		A Positive
NH*	36	16	Numeric High
NL*	37	17	Numeric Low

*The A register must be loaded with one's complement of number to be tested, before the conditional logand is given.

ALPHABETICAL INDEX

(To chapters 1 through 4)

- A -

ALPHABETICAL INDEX (Cont.)

- A -

- B -

- B -

- C -

- C -

- D -

- E -

- E -

- F -

- H -

- I -

- L -

ALPHABETICAL INDEX (Cont.)

- L -

- M -

- M -

- N -

- O -

ALPHABETICAL INDEX (Cont.)

- P -

- P -

ALPHABETICAL INDEX (Cont.)

- P -

- P -

ALPHABETICAL INDEX (Cont.)

- P -

- R -

- S -

- S -

- S -

- T -

ALPHABETICAL INDEX (Cont.)

- U -

- W -

- X -

- Z -