CONTROL DATA
CORPORATION

CONTROL DATA®
8092
TeleProgrammer

PROGRAMMING REFERENCE MANUAL

CONTROL DATA
CORPORATION

CONTROL DATA®
8092
TeleProgrammer

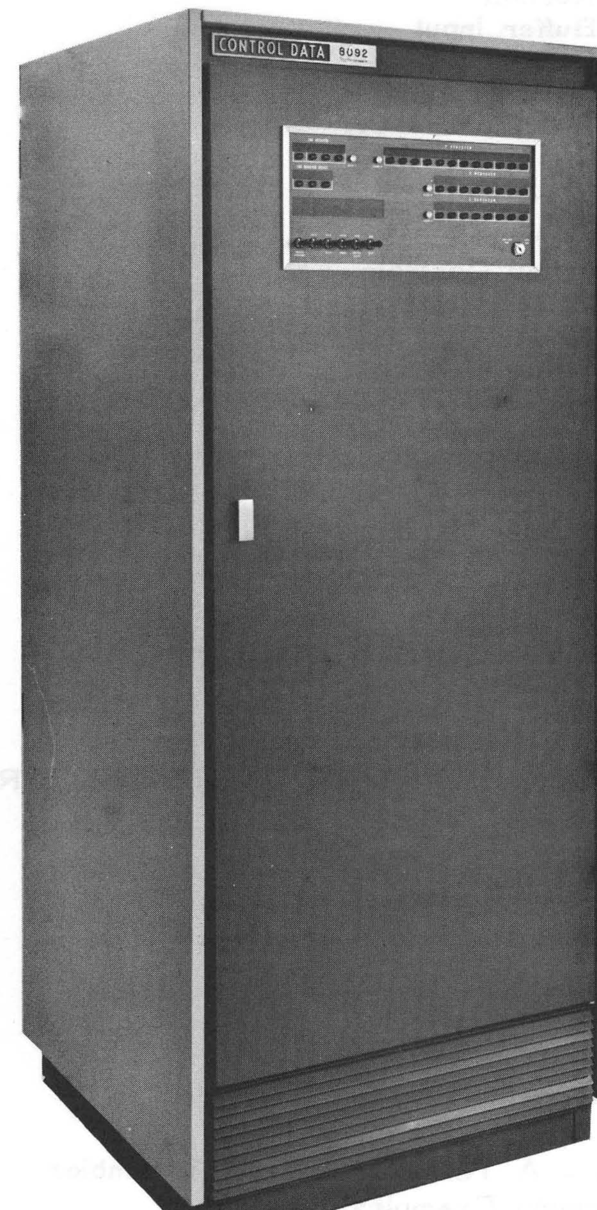
PROGRAMMING REFERENCE MANUAL

Pub No. 368 107 01

RECORD OF REVISIONS		
Rev.	Date	Notes
A	8/24/64	Reprint with revision.
Change 1	5/21/65	Revised to reflect ECOs 1163, 1198,
		1268, 1291, 1334, 1343, 1354, 1401,
		1402, 1403, 1411, 1431, 1470, 1505,
		and 1585. (Revision packet)
Change 2	8/25/66	Revised to reflect ECO DP2193.
		(Revision packet)
	5/15/67	Reprint (includes change 1 and 2).

CONTENTS

CHAPTER ONE - PROGRAMMING



8092 TeleProgrammer

CHAPTER ONE

PROGRAMMING

GENERAL CHARACTERISTICS

The CONTROL DATA* 8092 TeleProgrammer is a highly flexible and versatile stored program processor specially designed as a high speed buffer memory system for use in a variety of data communication applications.

Among the more important features are the following:

- stored program
- parallel mode of operation
- 8-bit word length
- 2048 words of core storage - 4096 (optional)
- 1 Direct I/O Channel (8 bits)
- 1 Buffer I/O Channel (8 bits)
- versatile instruction repertoire of 42 instructions
- 3 Auxiliary Tag registers of 4 bits each
- indirect and direct addressing and modification
- interrupts
- 12 bit external function address codes
- 7 internal program registers
- physical size: height, 68 inches; width, 34 inches; depth, 30 inches
- storage reference cycle time of 4 microseconds
- The ability to use the OSAS or OSAS-A assembler for those who have a 160 or 160-A computer.

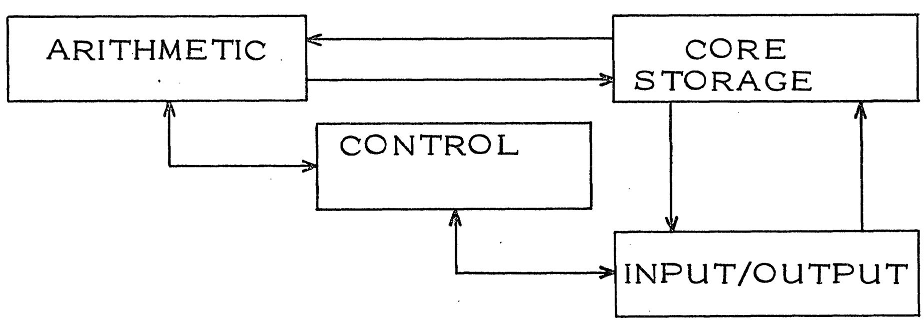
---

* Registered Trademark of Control Data Corporation

THE CENTRAL PROCESSOR

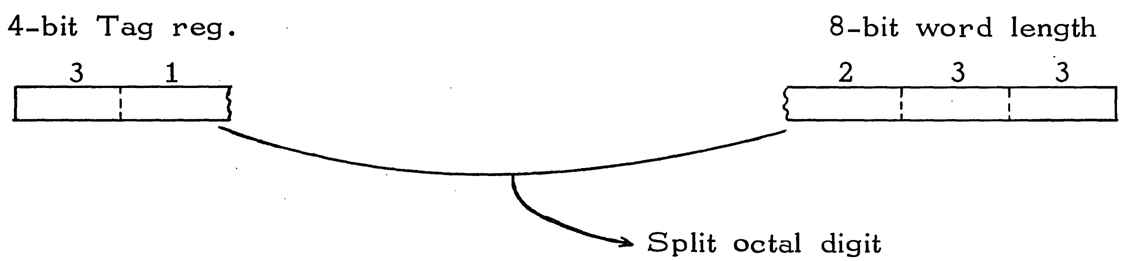
The TeleProgrammer is a parallel, single address electronic data processor. Operations are controlled by an internally stored program located in sequential addresses. The storage cycle time is 4 microseconds. The basic memory may be expanded from 2048 words to 4096 words. Each internal core word contains 8 bits. Instructions are executed in <u>one</u> to <u>four</u> storage cycle times; with times varying from 4 to 16 microseconds. The average instruction time is approximately 10 microseconds.

The Block Diagram indicates the principal functional divisions



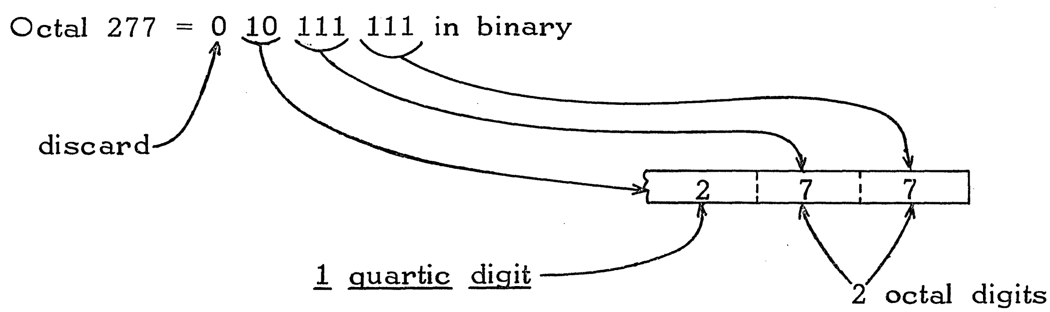
BASIC CONCEPTS IN PROGRAMMING THE TELEPROGRAMMER

The TeleProgrammer has some unique features for programming. Most of these center around the word length of 8 bits. In order to carry addresses for 4096 words, 12 bits are required ($2^{12} = 4096$, where highest address is $2^{12} - 1$). To provide for 12 bits, the TeleProgrammer makes use of three 4-bit Tag registers (Tag registers 1, 2, and 3). The carry over from 8 bits to 4 additional bits, in the Tag register, causes a split in the second octal digit from the left. This is indicated below:



In this manual, the 8-bit word length will be represented as two full <u>octal</u> digits and one <u>quartic</u> digit (the leftmost 2 bits). The Tag registers will be generally represented as shown above, with one full <u>octal</u> digit (on the left) and a <u>single bit</u> (0 or 1) on the right. The jagged ( ) ends of the registers indicate the split octal digit.

In addition, this manual will refer to numbers of "three octal digits" being contained in the 8-bit word length. Actually, this is physically impossible, since three octal digits occupy 9 bits and there are only 8 bits in the TeleProgrammer word. However, what is meant here, is that the <u>leftmost bit</u> of the three digit octal number is to be discarded. For example, show the octal number, 277, in a TeleProgrammer word.



This convention of representing the contents of the 8-bit words will be used many times in this manual. Looking at the above 9-bit configuration, one can see that to discard the leftmost bit, it must be zero. This means that the highest quartic digit of the word is 3. This, in turn, indicates the maximum "octal" of three digits which can be expressed in the 8-bit word length; --it is 377. The octal range 000 through 377 is equivalent to 256 registers. Since each Tag register holds 4 bits, there are 16 possible configurations for the 4 bits (0000 through 1111). Thus, 16 times 256 = 4096 total registers available.

WORD FORMAT

People who work with computers are generally acquainted with the term, "octal". It is the number base associated with three bits --which in turn, provides eight possible number states (zero through seven). Since the 8092 TeleProgrammer has an 8-bit word length and partitioning by three bits over the complete word is inefficient; the number base of four with partitioning by two bits is used for the upper two bits of the word. The number base of four, is referred to, in this manual, as QUARTIC. Keep in mind, that only the upper two bits of the word length is expressed in Quartic. The lower six bits are expressed by two octal digits. The upper QUARTIC digit is represented by bits, as shown below:

Bits	The Upper QUARTIC Digit
00	0
01	1
10	2
11	3

The CONTROL DATA 8092 TeleProgrammer word contains 8 binary digits. These are shown below with the least significant bit (b_0) on the right.

Single Word Binary Format

8 bits

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0

Any binary digit above can be represented by any combinations of ones or zeros. Although the 8092 operates in binary, it is more efficient to consider the word format as containing 2 octal and 1 Quartic digits, as shown below:

Single Word Format With Two Octals and One Quartic

8 bits

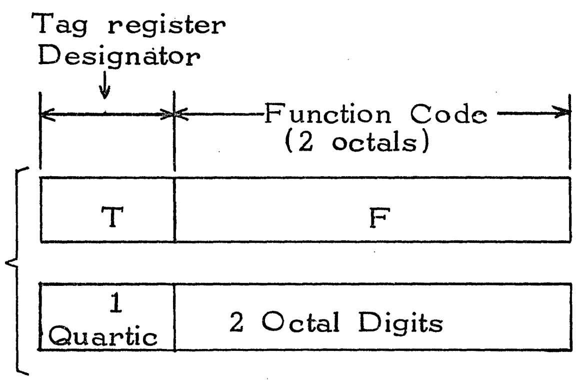
C_3	C_2	C_1
Quartic Digit (0,1,2,or 3)	Octal Digit (0 thru 7)	Octal Digit (0 thru 7)

INSTRUCTION WORD FORMAT

The TeleProgrammer operates on a two word instructional set. Most instructions are contained in a set of two sequential storage locations. The first word contains the Function Code, in the lower 2 octal digits, and the Tag register designator, T, in the upper quartic digit. The second word of the instructional set holds: an operand of 2 octals and 1 quartic, or a partial address of 2 octals and 1 quartic. Three modes of operation are possible in the 8092; NO ADDRESS MODE, MEMORY ADDRESS MODE, and INDIRECT ADDRESS MODE. Examples are shown below:

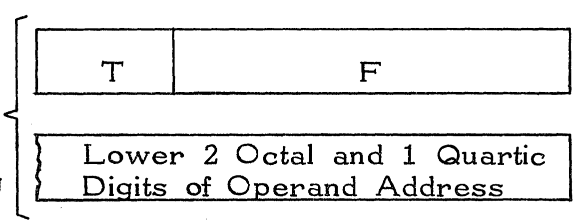
NO ADDRESS MODE

Where T = 0, since there is no Auxiliary Tag register used in this mode. The operand must contain 3 digits in the octal range of 000 thru 377.



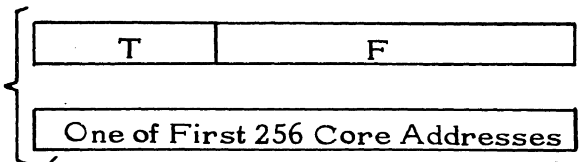
MEMORY ADDRESS MODE

Where T can equal 0, 1, 2, or 3. The lower 8 bits of the operand address appear in the second word and the upper 4 bits of the operand address appear in the Auxiliary Tag register designated by T. If T=0, the address of the operand is fully contained in the second word of the instructional set.



INDIRECT ADDRESS MODE

Where T can equal 0, 1, 2, or 3. At one of the first 256 core locations, given in the second word, is the lower 8 bits of the operand address. The upper 4 bits of this operand address will be found in the Auxiliary Tag register indicated by T.



(Octal range of these addresses is: 000 thru 377)

Examples of the Three Operational Modes

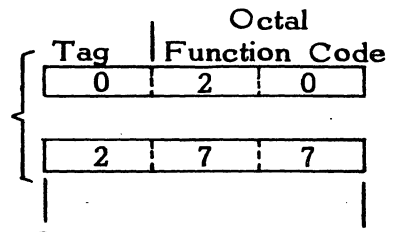
Example 1.

Put the octal number, 277, into the A register.

Solution:

Since no Auxiliary Tag register is involved, T = 0. The octal code for "LOAD A" in this mode is 20; thus F = 20. The octal operand, 277, is placed in the second set as 2 octals (77), and 1 quartic (2).

NO ADDRESS MODE



Operand of 3 digits (2 octals and 1 quartic)

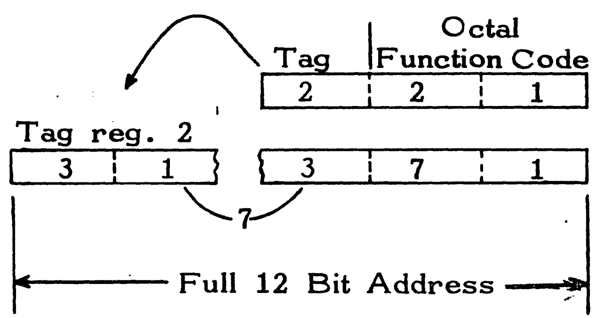
Example 2.

Load the contents of octal address, 3771, into the A register .

Solution:

The Tag, 2, indicates Auxiliary Tag register 2 holds the upper 2 quartic digits of the address whose lower 8 bits are given in the second instruction word. Note, octal 3771 is contained in the designated Tag register and the second word of the instruction set.

MEMORY ADDRESS MODE



Note: The quartic and 1 bit fit together to form octal, 7, the second digit of the address.

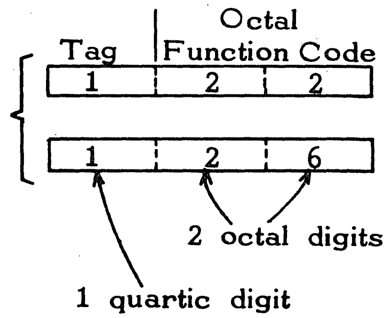
Example 3.

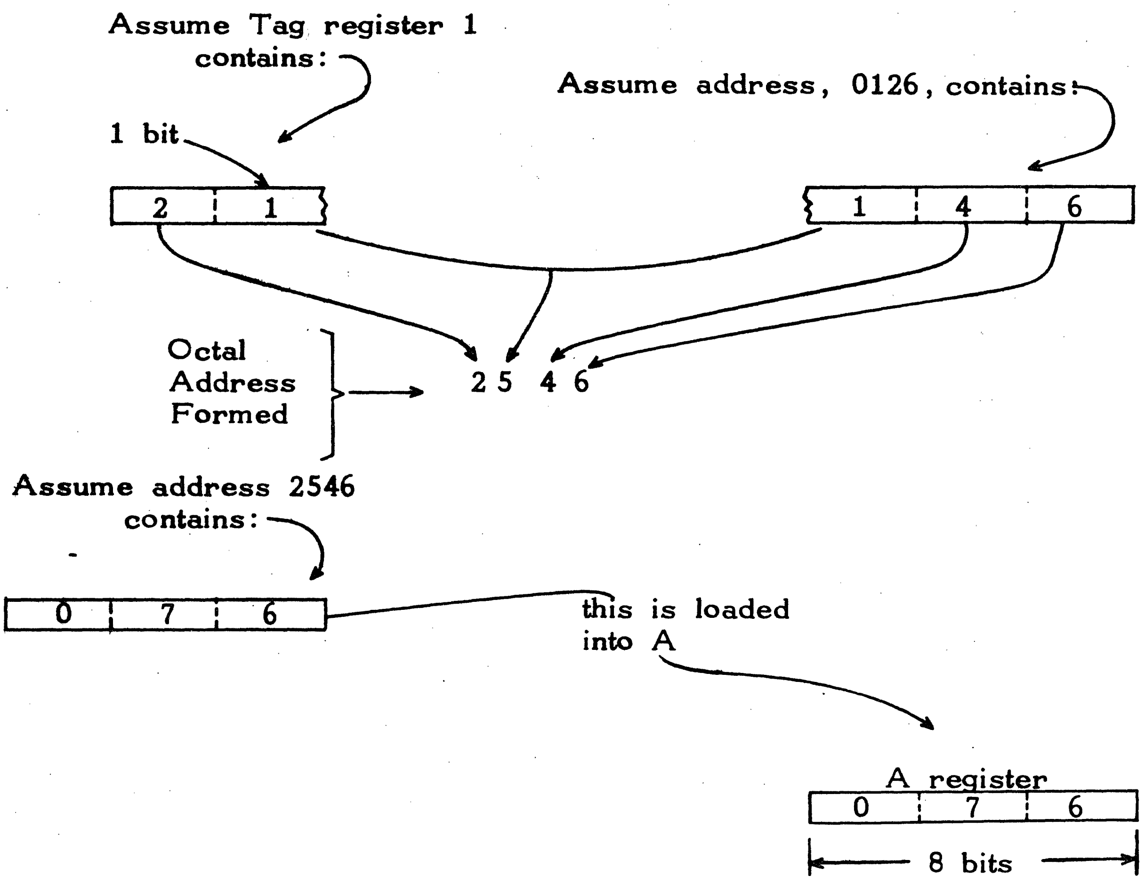
Load the operand whose complete address is in address 0126 and Tag reg. 1.

Solution:

Octal 126 is the address given in the second word. At this address, 0126, the lower 8 bits of the location of the operand are placed. The upper 4 bits of the operand location are placed in Auxiliary Tag register, 1, indicated by the Tag designator of the first word. Continuing this example, assume address 0126 and Tag register 1 contain the quantities shown below, show what finally is loaded into A.

INDIRECT ADDRESS MODE





THE 8092 TELEPROGRAMMER
INSTRUCTION REPERTOIRE

Functions	Rel. Code	Octal Code	Cycle Time *
LOADS:			
Load A (No.)	LDN	20	2
Load A (Mem.)	LDM	21	3
Load A (Ind.)	LDI	22	4
Load Comp. (Mem.)	LCM	25	3
Load Comp. (Ind.)	LCI	26	4
Tag Reg. to A	TTA	03	1
Clear A	CLA	03***	1
BER to A	BER	06	1
STORES:			
Store A (Mem.)	STM	41	3
Store A (Ind.)	STI	42	4
A to Tag Reg.	ATT	02	1
A to BER	ABR	04	1/2
A to BXR	ABX	05	1/2
JUMPS: ****			
If A = 0	ZJP	60	2
If A ≠ 0	NZP	61	2
If A ≥ 0	PJP	62	2
If A < 0	NJP	63	2
Unconditional	UJP	64	2
Do Nothing	DON	02***	1
SHIFTS:			
A Left 1 bit	SHA	01	1

Functions	Rel. Code	Octal Code	Cycle Time *
ARITHMETICS:			
Add (No. Adr.)	ADN	30	2
Add (Mem.)	ADM	31	3
Add (Ind.)	ADI	32	4
Subtract (No.)	SBN	34	2
Subtract (Mem.)	SBM	35	3
Subtract (ind.)	SBI	36	4
Replace Add (Mem.)	RAM	51	4
Replace Add 1	RAO	55	4
LOGICALS:			
Log. Prod. (No.)	LPN	10	2
Log. Prod. (Mem.)	LPM	11	3
Log. Prod. (Ind.)	LPI	12	4
Log. Sum (No.)	LSN	14	2
Log. Sum (Mem.)	LSM	15	3
Log. Sum (Ind.)	LSI	16	4
IN-OUT:			
Input Normal	INN	72	**
Output Normal	OUT	73	**
Input Buffer	IBI	70	1/2
Output Buffer	IBO	71	1/2
Input to A	INA	76	2
Output No. Adr.	OTN	74	2
CONTROLS:			
Ext. Function	EXF	75	3
Clear Interrupt	CIL	13	1
Clear Buffer	CBC	07	1
Error Stop	ERR	00	-
Halt	HLT	77	1

* Cycle Times; each cycle = 4 microseconds.
** 3 + 2(X = 1) + terminate time. Where X = No. of words.
*** No tag should be referenced.
**** Jump cycle time is 1 cycle, if jump is not made.

DESCRIPTION AND EXAMPLES OF INSTRUCTIONS

LOAD Instructions

Seven LOAD instruction are available. These are:

LDN	-	LOAD A (No Address Mode)
LDM	-	LOAD A (Memory Address Mode)
LDI	-	LOAD A (Indirect Address Mode)
LCM	-	LOAD Complement to A (Memory Address Mode)
LCI	-	LOAD Complement to A (Indirect Address Mode)
TTA	-	Tag Register Contents to A
BER	-	Contents of BER Register to A

LDN - (20) - LOAD A (No Address) 2 Cycles

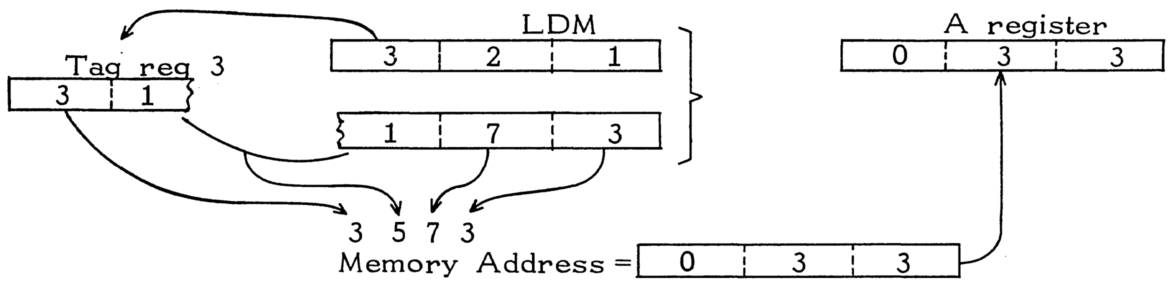
Load the A register with the contents of the second word of the instructional set. Octal numbers 000 through 377 can be entered into A by this instruction.

Example: Put the octal number, 177, into A

LDM - (21) - LOAD A (Memory) 3 Cycles

Load the A register with the contents of the memory address whose lower eight bits are given in the second instruction word and whose upper four bits are contained in the designated Auxiliary Tag register.

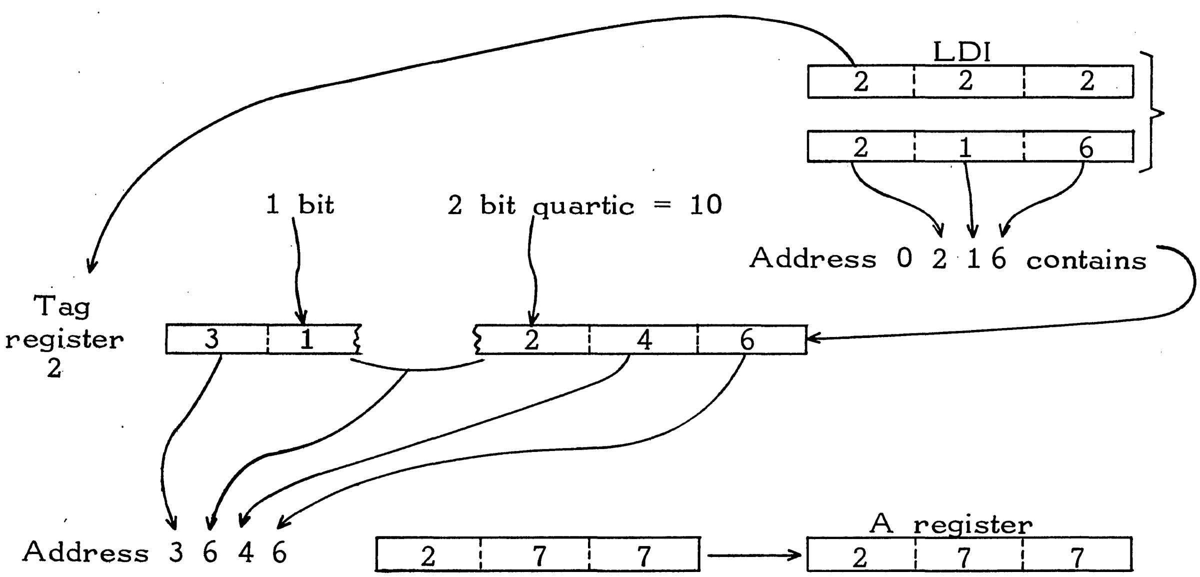
Example: Assume memory address 3573 (in octal) contains the octal quantity, 033. Load this into A .



LDI - (22) - LOAD A (Indirect) 4 Cycles

Load A with the contents of the address whose lower 8 bits are contained in one of the first 256 (decimal) addresses, and whose upper 4 bits are contained in a designated Auxiliary Tag register. The location in the core (one of the first 256 decimal addresses) is given in the second instruction word. The Auxiliary Tag register is indicated in the first word.

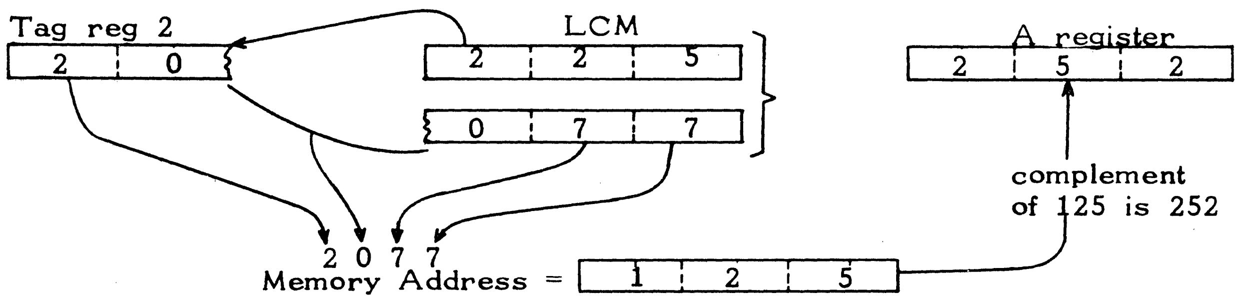
Example: Assume octal address, 3646, contains the octal number, 277. Load this number to A, using the indirect mode via octal address 0216.



LCM - (25) - Load Complement to A (Memory) 3 Cycles

Load the A register with the complement of the contents of the memory address whose lower 8 bits are given in the second instruction word and whose upper 4 bits are contained in the designated Auxiliary Tag register.

Example: Assume memory address 2077 (in octal) contains the octal quantity, 125. Load the complement of this quantity into A.

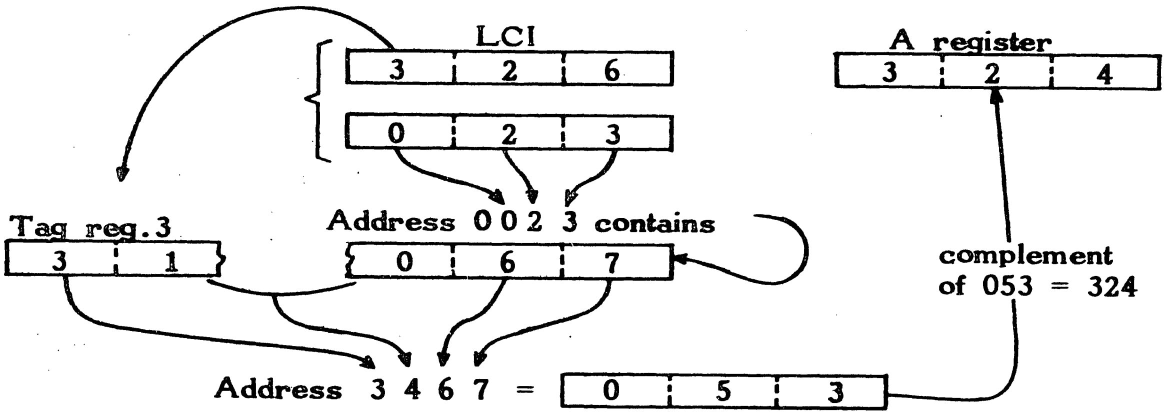


($\bar{\text{Note}}$: quartic complements are "three's complement". Thus, the complement of the quartic digit, 1, above is 2; whereas, the complements of the octal digits 2 and 5 are respectively 5 and 2.)

LCI - (26) - Load Complement to A (Indirect) 4 Cycles

Load A with the complement of the contents of the address whose lower 8 bits are contained in one of the first 256 (decimal) addresses and whose upper 4 bits are contained in the designated Auxiliary Tag register. The location in the core (one of the first 256 decimal addresses) is given in the second instruction word. The Auxiliary Tag register is indicated in the first word.

Example: Assume octal address 3467 contains the octal number, 053. Load the complement of this number into A, using the indirect mode and via octal location 0023.

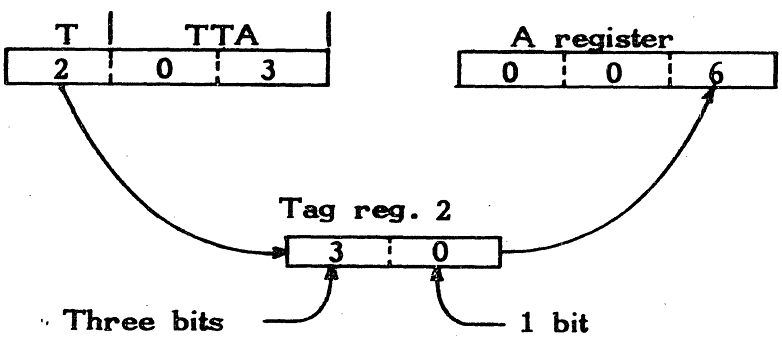


Note: The 1 bit of Tag register 3 and the quartic digit at address 023, form the bits, 100, which gives the octal digit, 4. Also note the complement of the quartic digit, 0, at address 3467 is equal to 3; whereas the complements of the octal digits 5 and 3 are respectively equal to 2 and 4

TTA - (03) - Tag Register to A 1 Cycle

Load the contents of the designated Auxiliary Tag register into the A register. Pack zero's in upper 4 bits.

Example: Load contents of Tag register, 2, into A.



Note: The four bits of the Tag register are: 0110. When packed to the right of A, they give the following: 00 000 110 = 006.

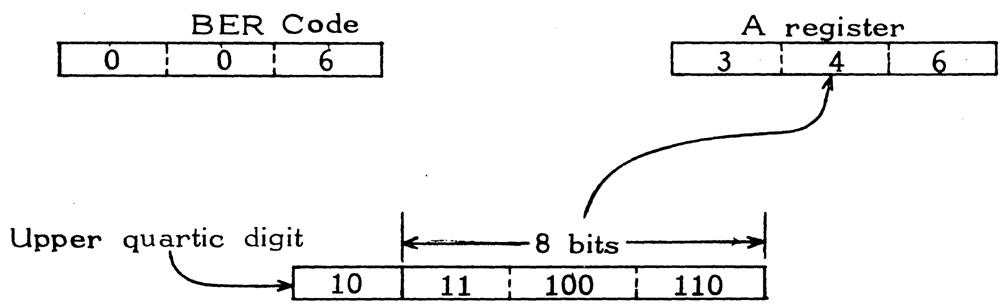
CLA - (03) - Clear A 1 Cycle

This instruction is the same as the preceding instruction (TTA) except that the TAG is not referenced. The A register is therefore cleared (all zeros).

BER - (06) - Buffer Entrance Register to A   1 Cycle

Load the A register with the lower 8 bits of the Buffer Entrance register.

Example: Load Buffer Entrance register into A



Buffer Entrance Register (10 bits)
(Shown in bit format)

Note: On this instruction, the lower 8 bits (1 quartic and 2 octals) are transferred into the A register. The upper 2 bits (1 quartic digit) are not transferred. On the reverse transfer (A to BER), the right 2 bits of Tag register 3 are sent to the upper 2 bit locations of BER. This is explained in detail in the ABR instruction.

STORE Instructions

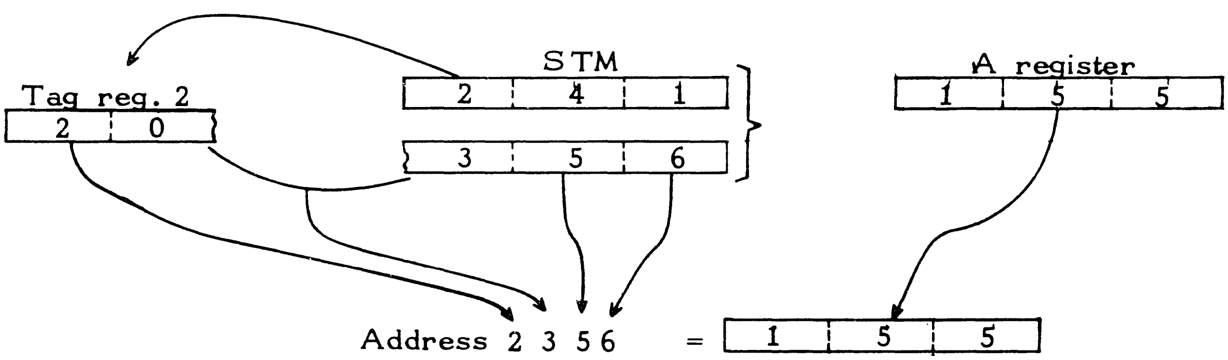
Five STORE instructions are available; these are:

STM	-	STORE A (Memory Address Mode)
STI	-	STORE A (Indirect Address Mode)
ATT	-	A to Tag Register
ABR	-	A to Buffer Entrance Register
ABX	-	A to Buffer Exit Register

STM - (41) - STORE A (Memory Mode) 3 Cycles

Store the contents of the A register into the location whose address is equivalent to the combined contents of the designated Tag register and the second word of the instruction set.

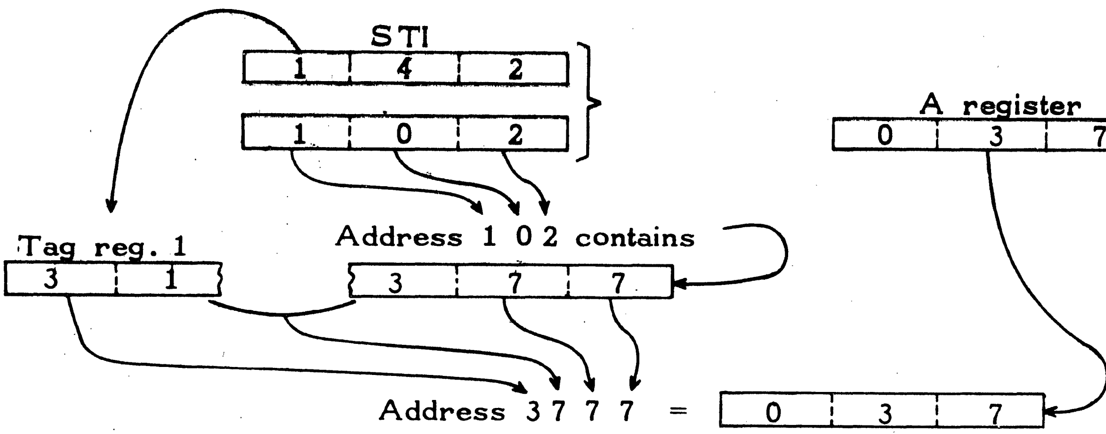
Example: Assume A contains the octal number, 155. Store this number at octal address, 2356.



STI - (42) - STORE A (Indirect Mode) 4 Cycles

Store the contents of the A register into the location whose address is equivalent to the combined contents of the designated Tag register and the contents of one of the first 256 decimal core registers. The exact location of one of these 256 registers is given, through its address, in the second instruction word.

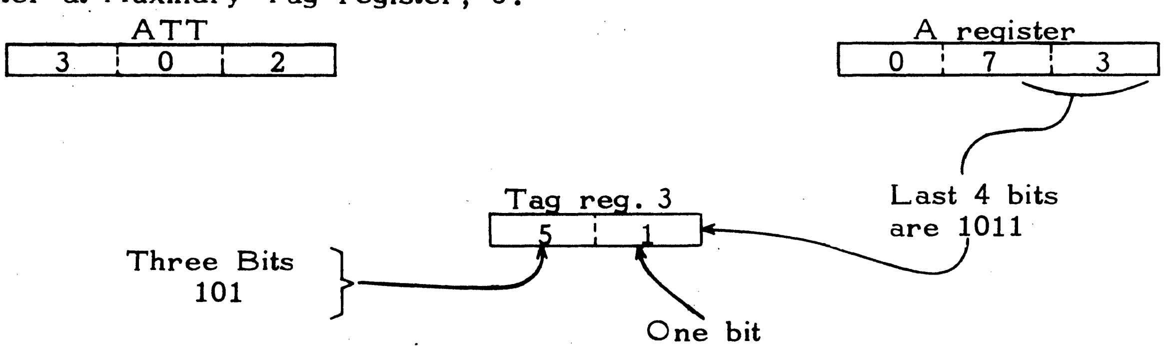
Example: Assume the A register contains the octal number, 037. Store this number in octal address, 3777, by using octal location 0102, and the indirect mode.



ATT - (02) - A to Tag Register 1 Cycle

Transfer the lower 4 bits of the A register into the designated Auxiliary Tag register.

Example: Assume the A register contains the octal number, 073. Store the A register at Auxiliary Tag register, 3.



DON - (02) - Do Nothing 1 Cycle

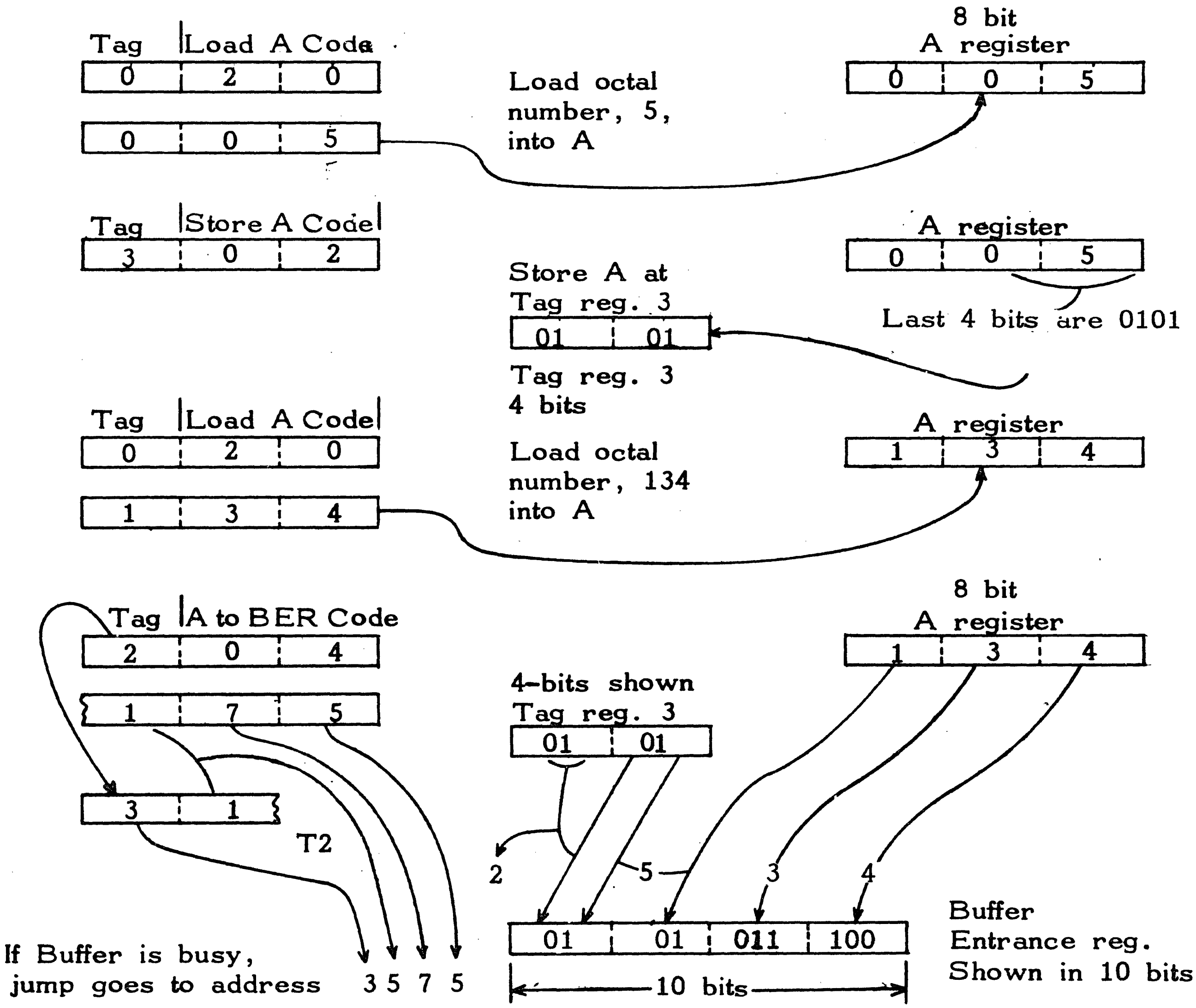
This instruction is the same as the preceding instruction (ATT), except that the tag is not referenced. This instruction has no operation. Control goes to the next instruction set.

ABR - (04) - A to Buffer Entrance Register 1 Cycle, 2 Cycles if jump is made.

Transfer the contents of A to the lower 8 bit positions of the Buffer Entrance register. The rightmost 2 bits of Tag register 3 become the 9th and 10th bits of the Buffer Entrance Register (BER); the upper two bits of Tag register 3 are referenced for bits 11 and 12 of BER. If the buffer is busy, a jump occurs to the combined address contained in the second word of the instruction set and the designated Tag register. If the buffer is not busy, control goes to the next instructional set.

Example: Assume one wants to effectively enter a starting octal address of 2534 into the Buffer Entrance register. Shown are the program steps involved.

To effectively enter a starting address, 2534 into BER



Since BER is a 10-bit register, there is not room for the full 12-bit address. The upper 2 bits (1 quartic) are obtained by referencing the left 2 bits of Tag register 3. In the above example, the left 2 bits of Tag register 3 and the left-most bit of BER give the octal digit, 2.

ABX - (05) - A to Buffer Exit Register 1 cycle, 2 cycles if jump is made.

Transfer the contents of A to the lower 8 bits of the Buffer Exit register (BXR). The right quartic digit (2 bits) of Tag register 3 fills the 2 upper bits of BXR. The instruction is used to store the terminating address for buffer transfers. The left quartic digit of Tag register 3 is referenced by the TeleProgrammer to determine the highest order 2 bits of the address. For a Buffer Input instruction, enter the LWA + 1, and for a Buffer Output instruction, enter the LWA + 2.

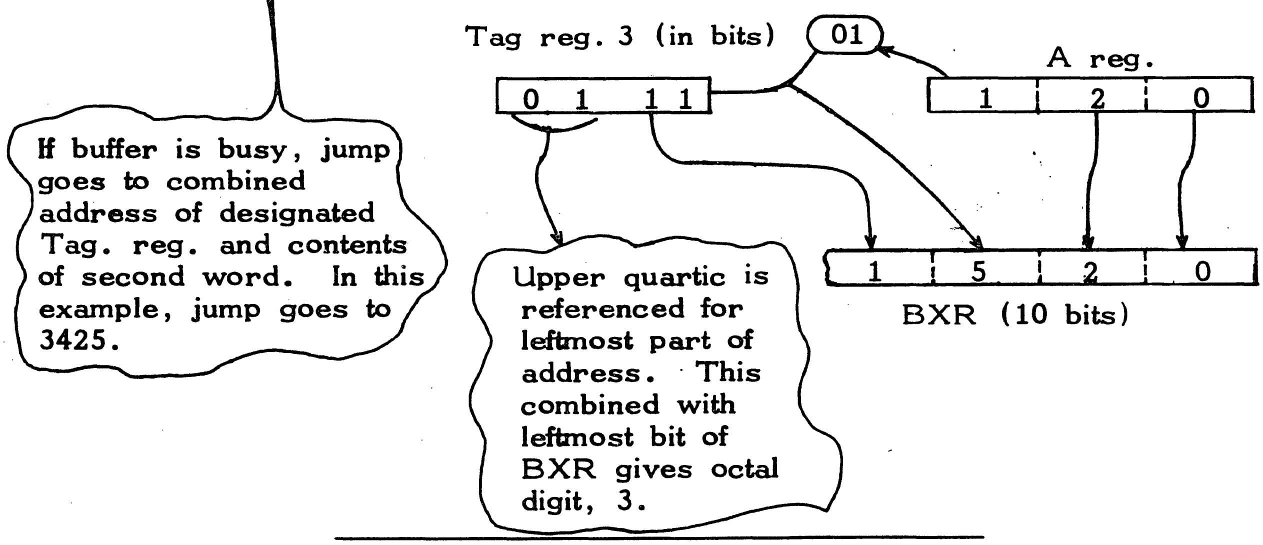
If the buffer is busy a jump occurs to the combined address contained in the designated Tag register of the first word and the contents of the second word of the instruction set. If not busy, control continues to the next instruction set in sequence.

NOTE: The above concept may be clearer, if it is remembered that 12 bits, rather than 8 bits, are required to cover the whole possible address range of 4096 registers. As a consequence, it must be possible to perform buffer operations covering the complete address range. To accomplish this, the BER or BXR (of 10 bits) uses the 8 bits of the instruction operand, 2 bits from TAG register 3 (the lowest order 2 bits). By referencing the highest order 2 bits of Tag register 3, the full 12 bits are available.

The use of a 10 bit BER and BXR allows a maximum buffer operation of 2000_8 words. The first word address and last word address must be identical in the highest order 2 bits. The highest order 2 bits of Tag 3 must not be altered during buffer operations.

Example: Show a program which places octal address, 3520 into BXR; if the buffer is busy, wait until it is not busy.

Location of Instruction	Instructions	Explanation of Action which Occurs	
3420 3421	020 007	Load A with the octal number, 007.	007 ⟶ A
3422	302	Store lower 4 bits of A at Tag register, 3.	Bits, 0111, go to Tag reg. 3
3423 3424	020 120	Load A with the octal number, 120.	120 ⟶ A
3425 3426	305 025	A goes to bits 1 thru 8 of BXR, right quartic (3) of Tag reg. 3 goes to bits 9 and 10 of BXR. Left quartic digit of Tag reg. 3 is <u>referenced</u> for upper 2 bits of address.	



JUMP INSTRUCTION

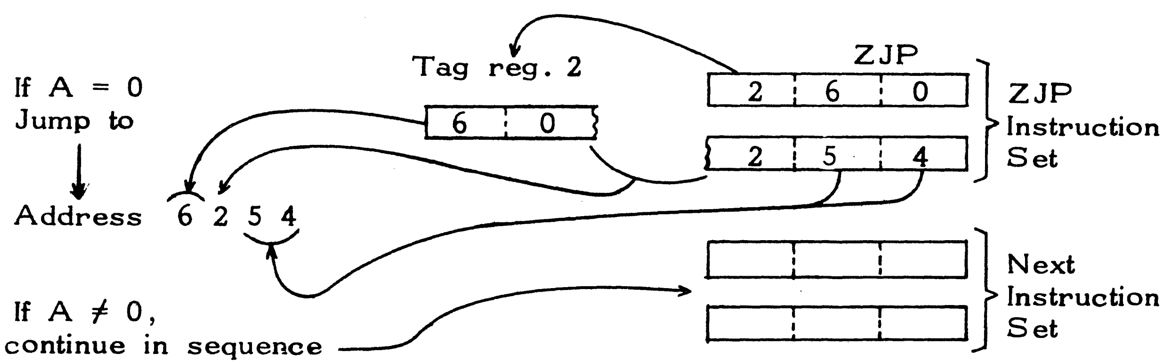
Five JUMP instructions are available, they are:

ZJP - JUMP, if contents of A = 0
NZP - JUMP, if contents of A ≠ 0
PJP - JUMP, if contents of A ≥ 0 (positive)
NJP - JUMP, if contents of A < 0 (negative)
UJP - Unconditional JUMP

ZJP - (60) - Zero JUMP 2 Cycles if jump is made; otherwise, 1.

If the contents of A equals zero, jump to the combined address contained in the designated Tag register and the second word of the instruction set. If the contents of A are not zero, continue in sequence with next set of instructions.

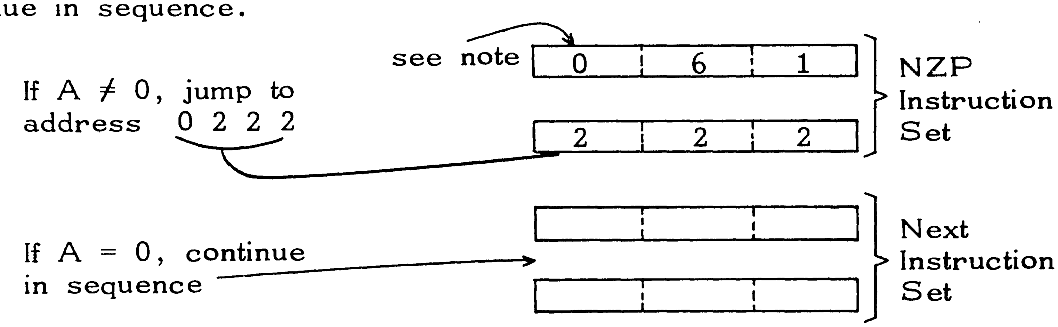
Example: Test A for zero, and jump to octal address, 6254, if A is zero; otherwise continue.



NZP - (61) - Not Zero JUMP 2 Cycles if jump is made; otherwise, 1.

If contents of A are not zero, jump to the combined address contained in the designated Tag register and the second word of the instruction set. If the contents of A are zero, continue in sequence with the next set of instructions.

Example: Test A, and if not zero, jump to octal address, 0222. If zero, continue in sequence.

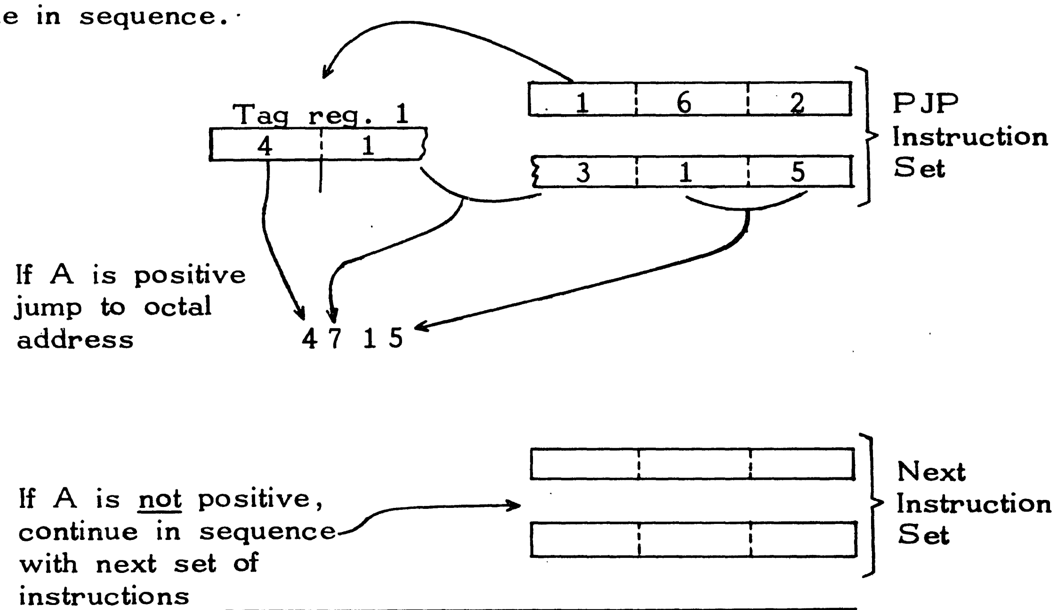


Note: Since the complete jump address can be expressed in 8 bits, no Tag register is required. Thus, the Tag designation = 0, in the first instruction word.

<u>PJP - (62) - Positive JUMP</u> 2 Cycles if jump is made; otherwise, 1.

If the contents of A are positive (equal or greater than zero), jump to the combined address contained in the designated Tag register and the second word of the instruction set. If the contents of A are not positive, continue in sequence. (If leftmost bit = 0, contents of A are positive.)

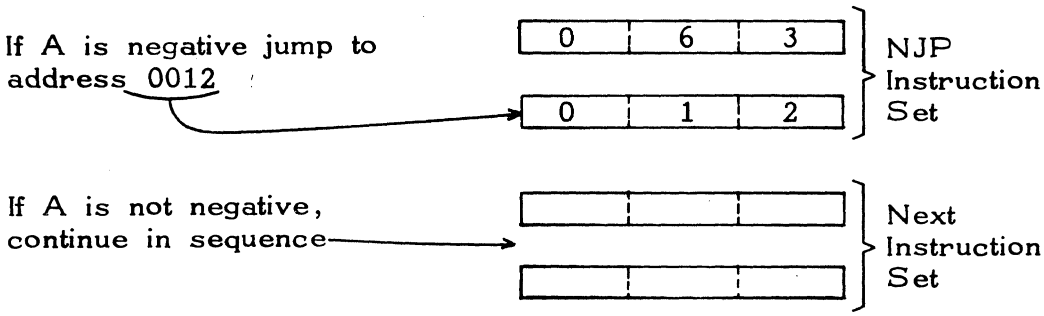
Example: Test A, and if positive, jump to octal address 4715. Otherwise, continue in sequence.



NJP - (63) - Negative JUMP   2 Cycles if jump is made; otherwise, 1.

If the contents of A are negative, jump to the combined address contained in the designated Tag register and the second word of the instruction set. If the contents of A are not negative, continue in sequence with the next set of instructions.

Example: Test A, and if negative, jump to octal address, 0012. If not negative, continue in sequence.

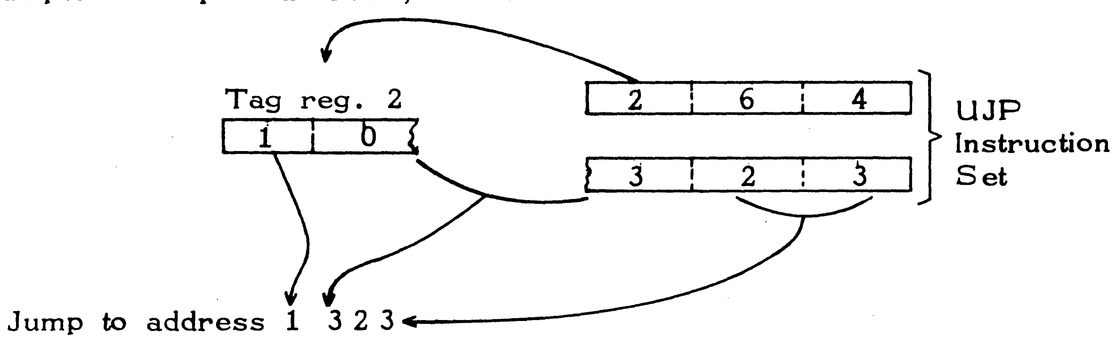


Since significant portion of the address can be contained in 8 bits, no Tag register is required and thus Tag designation of first instruction word is zero.

UJP - (64) - Unconditional JUMP   2 Cycles

Jump to the combined address contained in the designated Tag register and the second word of the instruction set.

Example: Jump to address, 1323.



SHIFT INSTRUCTION

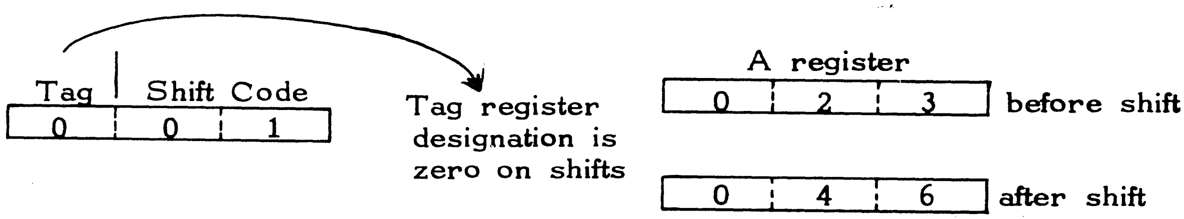
One shift instruction is available:

SHA = SHIFT A LEFT ONE BIT

SHA - (01) - Shift A Left 1   1 Cycle

Shift the contents of A left--end around--1 bit position. Bits coming off the left end of the A register enter the lowest bit position on the right end of the register.

Example: Assume A contains the octal number 023. Multiply the contents of A by 2, using the shift instruction.



Note: One shift instruction is required to shift A one place (1 bit) to the left. Each left shift is equivalent to one multiplication by 2. To shift 5 bits left, it is necessary to give 5 shift instructions, or loop through the single shift instruction 5 times.

ARITHMETIC INSTRUCTIONS

There are eight Arithmetic instructions: three adds, three subtracts, and two replace adds. These are:

ADN - ADD (No Address)
ADM - ADD (Memory Address)
ADI - ADD (Indirect Address)

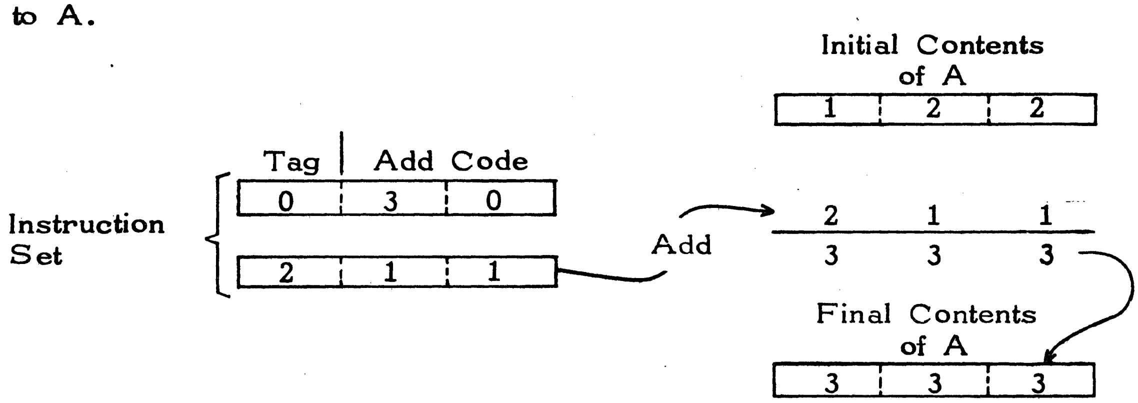
SBM - SUBTRACT (No Address)
SBM - SUBTRACT (Memory Address)
SBI - SUBTRACT (Indirect Address)

RAM - REPLACE ADD (Memory Address)
RAO - REPLACE ADD ONE (Memory Address)

<u>ADN - (30) - ADD (No Address)</u> 2 Cycles

Add to the A register the 8 bit number given in the second word of the instruction set. The sum is left in A.

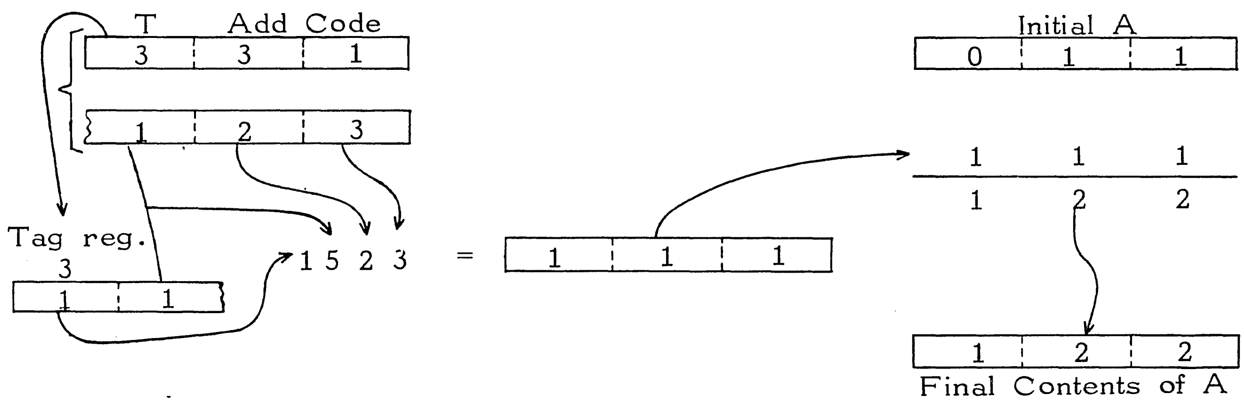
Example: Assume A contains the octal number, 122. Add the octal number, 211, to A.



<u>ADM - (31) - ADD (Memory Address)</u> 3 Cycles

Add to A the contents of the combined address given in the designated Tag register and the second word of the instruction set.

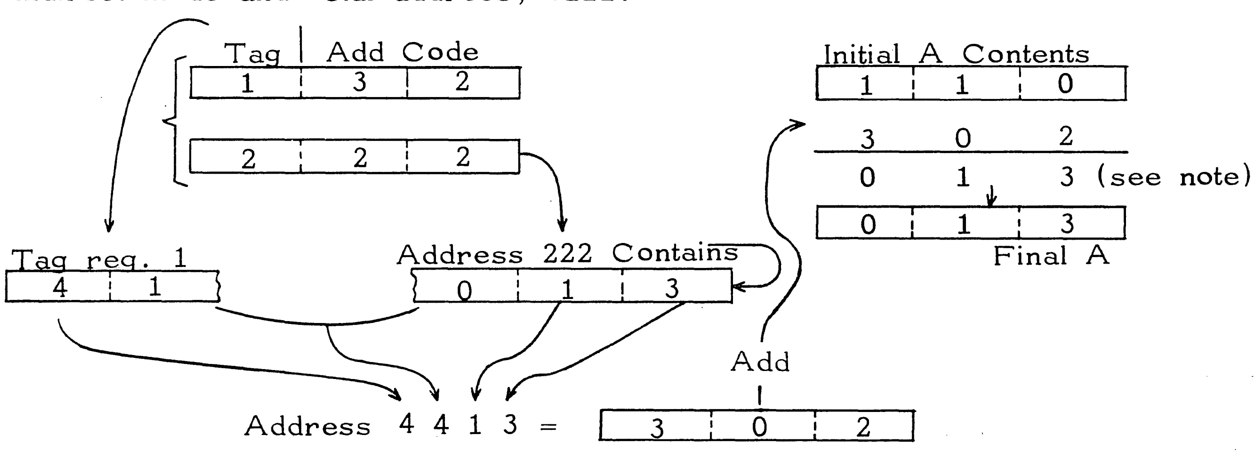
Example: Assume A contains the octal, 011. Add the contents of address 1523 to A. (Assume contents of address 1523 are 111.)



ADI - (32) - ADD (Indirect Address) 4 Cycles

Add to A the contents of the combined address contained in the designated Tag register and one of the first 256 decimal locations indicated in the second word of the instruction set.

Example: Assume A contains octal number, 110. Assume octal address, 4413 contains 302. Add the contents of address 4413 to A, by using the indirect mode and octal address, 0222.

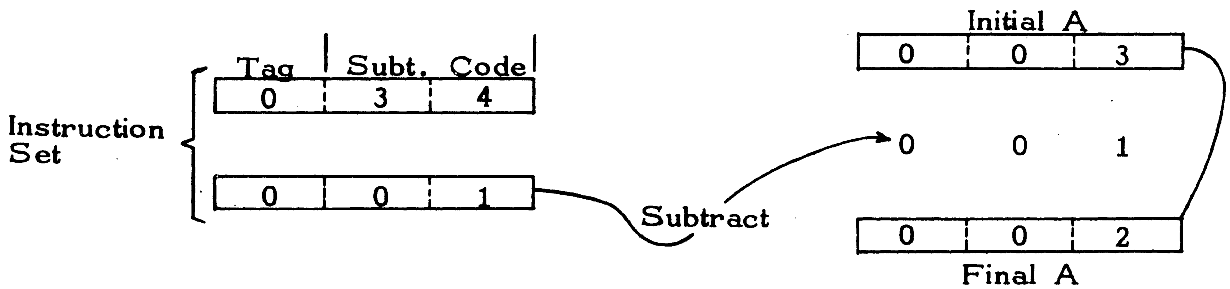


Note: The addition of 1 and 3 in the rightmost quartic digits overflows the register and the carry over (1) is added to the rightmost digit.

SBN - (34) - SUBTRACT (No Address) 2 Cycles

Subtract from the A register, the number contained in the second word of the instruction set. The difference is left in A register.

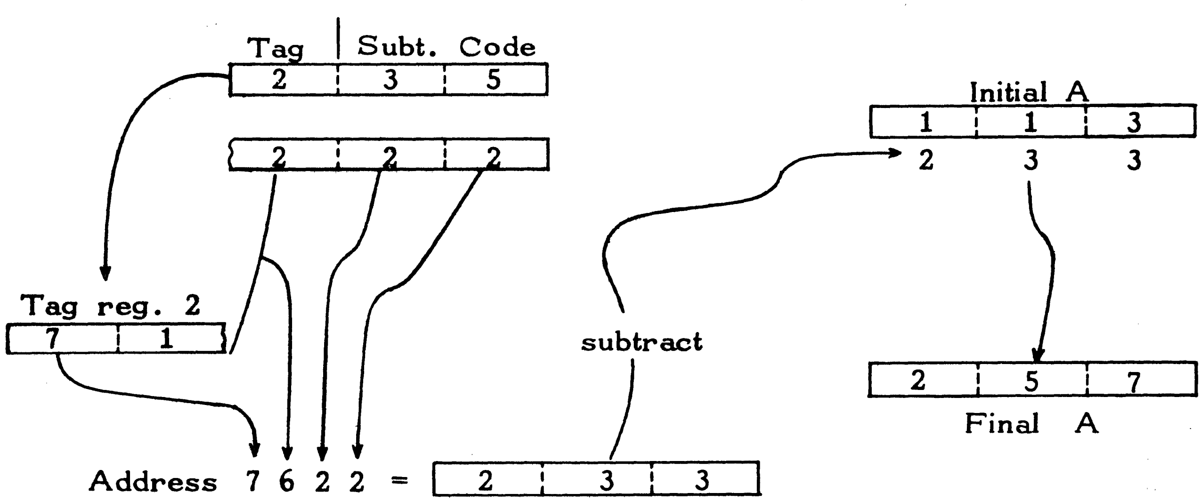
Example: Assume A contains 003. Subtract 001.



SBM - (35) - SUBTRACT (Memory Address) 3 Cycles

Subtract from the contents of A, the contents of the combined address contained in the designated Tag register and the second word of the instruction set.

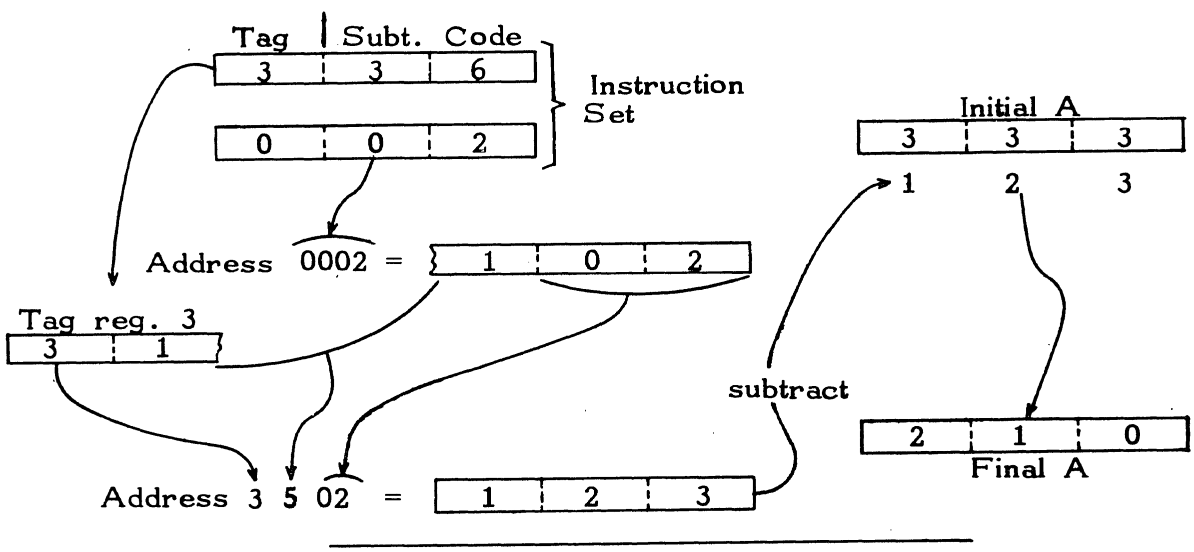
Example: Assume A contains the octal, 113. Assume address, 7622 contains 233. Subtract the contents of address 7622 from A.



SBI - (36) - SUBTRACT (Indirect Address) 4 Cycles

Subtract from the contents of A, the contents of the combined address contained in the designated Tag register and the location of one of the first 256 decimal registers, indicated by the second word of the instruction set.

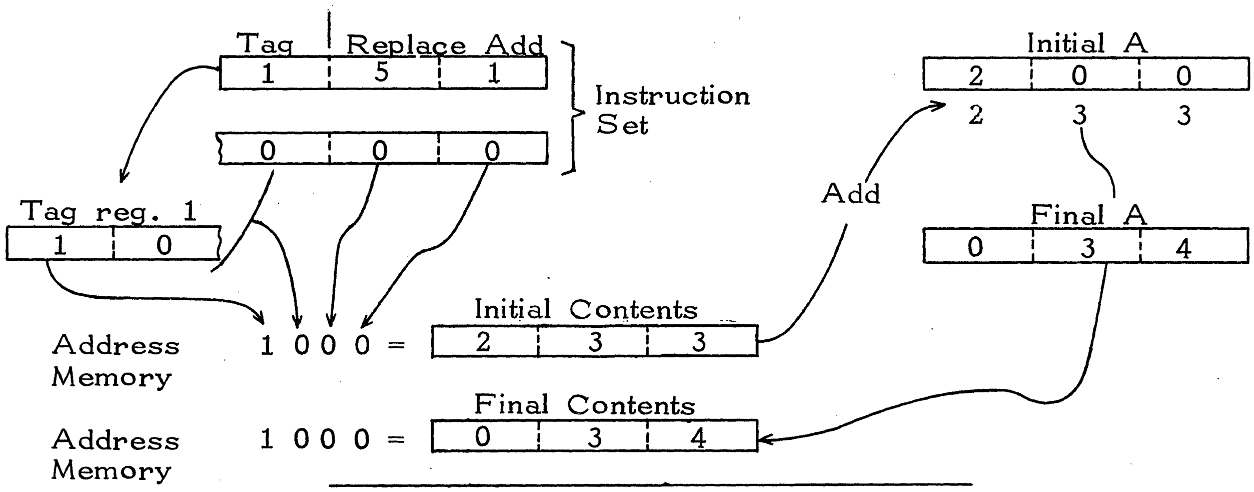
Example: Assume A contains the octal, 333. Assume address 3502 contains the octal number, 123. Reduce A by the contents of address 3502, using indirect mode and octal address, 0002.



RAM - (51) - REPLACE ADD (Memory Address) 4 Cycles

Add the contents of the A register to the contents of the memory address formed by the contents of the designated Tag register and the second word of the instruction set. The sum thus formed, remains in A, and replaces the initial contents of the memory address.

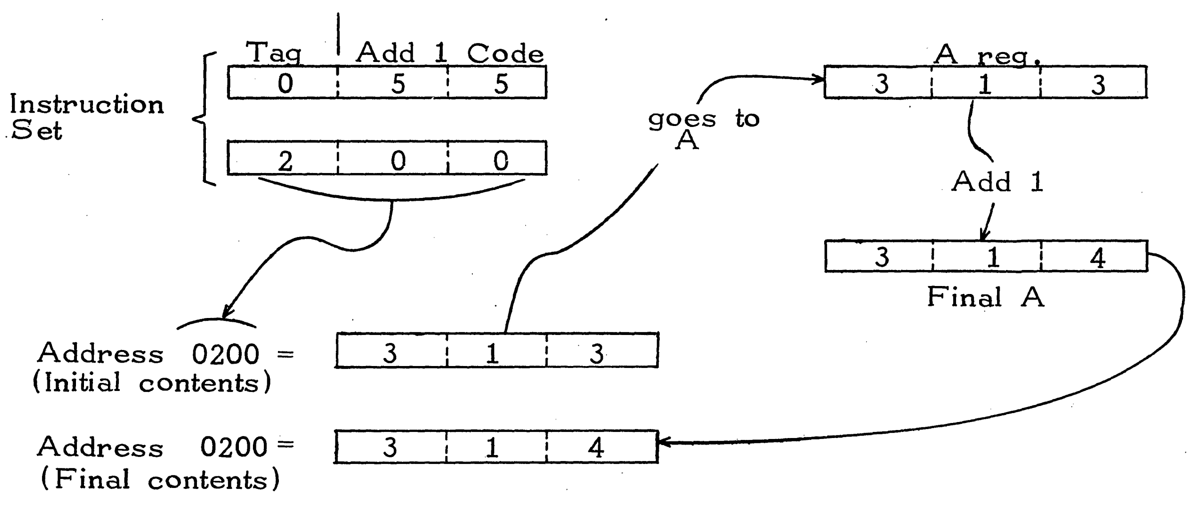
Example: Assume A contains the octal number, 200. Assume address 1000 contains the octal number, 233. Increase the contents of address 1000 by the contents of A.



RAO - (55) - REPLACE ADD ONE 4 Cycles

Add 1 to the contents of the memory address indicated by the combined contents of the designated Tag register and the second word of the instruction set. This sum is performed in A and remains in A at the end of the instruction.

Example: Add 1 to the contents of memory address, 0200.



LOGICAL INSTRUCTIONS

There are six Logical instructions: three of which are Logical products; three are Logical sums. These are:

LPN	-	LOGICAL PRODUCT (No Address)
LPM	-	LOGICAL PRODUCT (Memory Address)
LPI	-	LOGICAL PRODUCT (Indirect Address)
LSN	-	LOGICAL SUM (No Address)
LSM	-	LOGICAL SUM (Memory Address)
LSI	-	LOGICAL SUM (Indirect Address)

Logical Product is defined as a "bit by bit" multiply which observes the following rules:

1 times 0 = 0
0 times 0 = 0
0 times 1 = 0
1 times 1 = 1

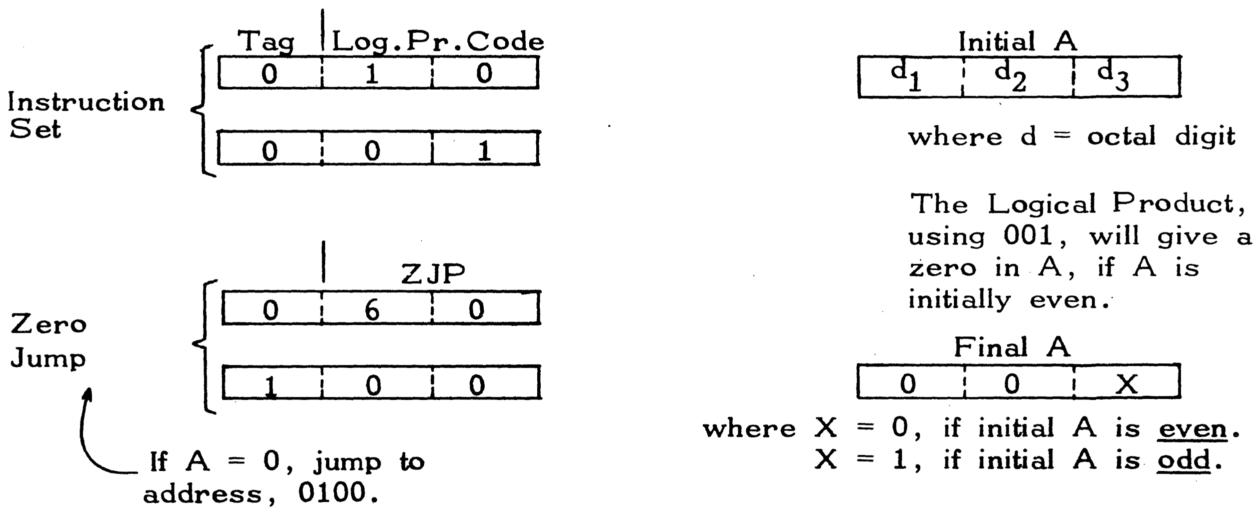
Logical Sum is a "bit by bit" sum without "carries" which observe the following rules:

1 + 0 = 1
0 + 1 = 1
0 + 0 = 0
1 + 1 = 0

LPN - (10) - LOGICAL PRODUCT (No Address) 2 Cycles

Form in A the Logical Product of the **contents of** A **and the contents of the** second word of the instruction set.

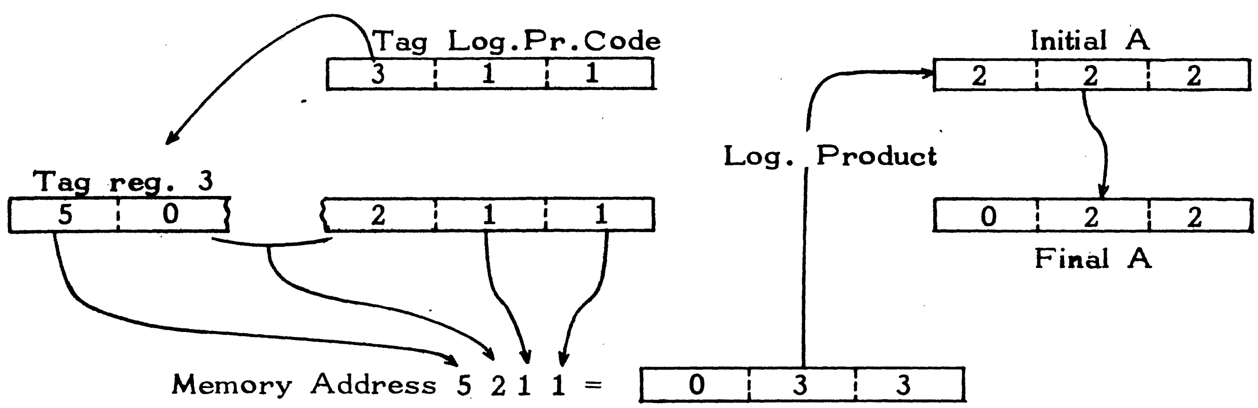
Example: Test A for "even". If even, jump to octal address, 0100.



LPM - (11) - LOGICAL PRODUCT (Memory Address) 3 Cycles

Form in A, the Logical Product of the contents of A and the contents of the memory location whose address is the combined contents of the designated Tag register, and the second word of the instruction set. The initial contents of the memory location remains unchanged.

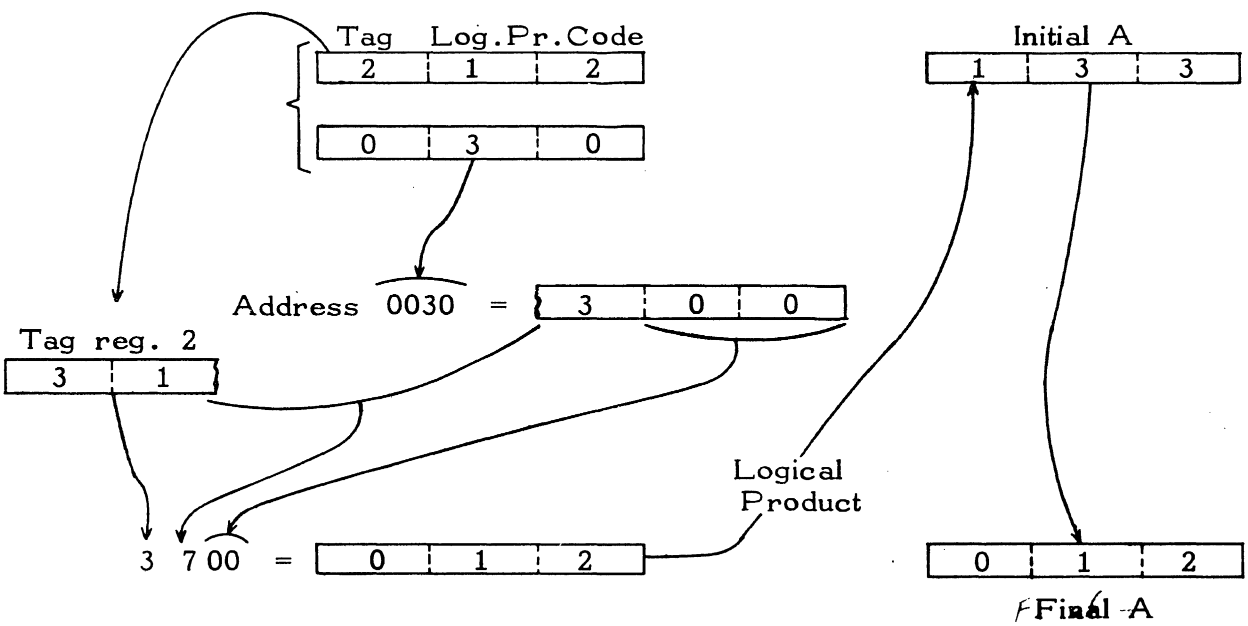
Example: Assume A contains the octal, 222. Assume memory address, 5211, contains 033. Form the Logical Product in A.



LPI - (12) - LOGICAL PRODUCT (Indirect Address) 4 Cycles

Form in A the Logical Product of the contents of A and the contents of the memory location whose address is the combined contents of the designated Tag register and the contents of one of the first 256 decimal locations. The address of this decimal location is given in the second word of the instruction set. The initial contents of the memory location remain unchanged.

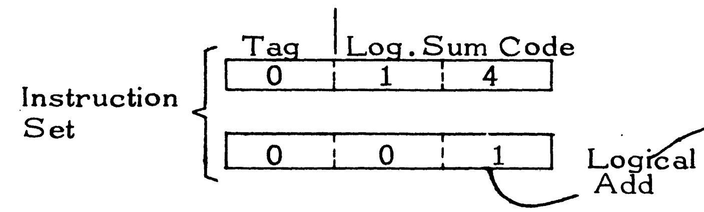
Example: Use the indirect mode to form the Logical Product of A and memory location 3700. Use octal location, 0030 in the process. Assume initial contents of A and location 3700 are respectively: 133 and 012.

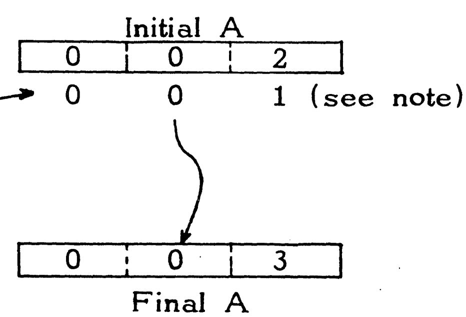


LSN - (14) - LOGICAL SUM (No Address) 2 Cycles

Form in A the Logical Sum of the contents of A and the second word of the instruction set.

Example: Assume A contains octal number, 002. Set A to 003.





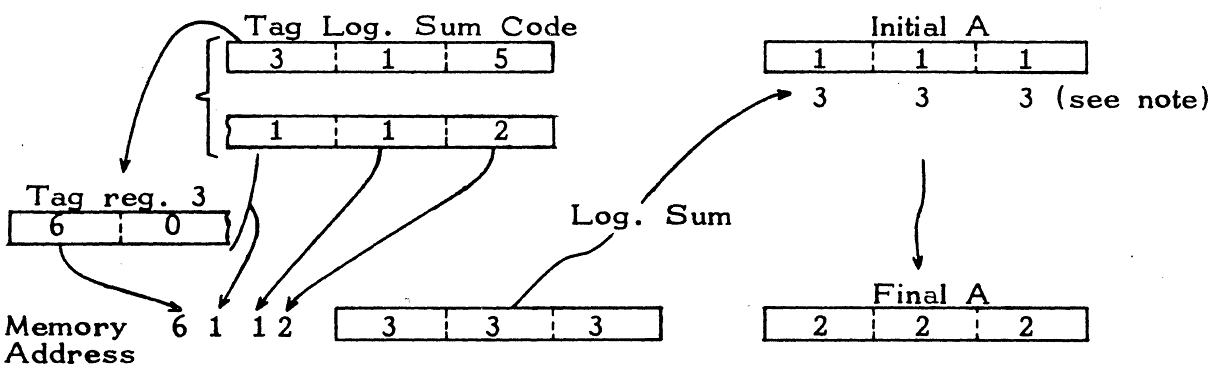
Note: The bit-by-bit Logical Add of the above example is:

002	=	00 000 010
001	=	00 000 001
Logical Sum	=	00 000 011 = 003

LSM - (15) - LOGICAL SUM (Memory Address) 3 Cycles

Form in A the Logical Sum of the contents of A and the contents of the memory location whose combined address is given in the designated Tag register and the second word of the instruction set.

Example: Assume A contains octal number, 111. Form in A the Logical Sum of the contents of A and the contents of memory location, 6112. Assume this location contains 333.



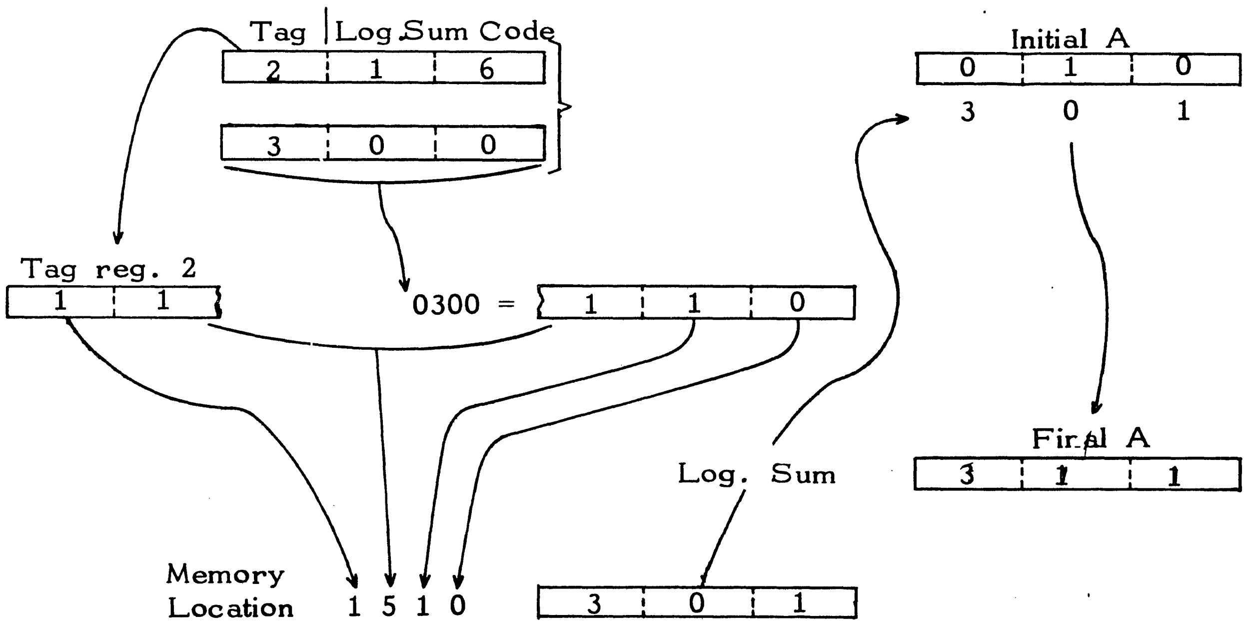
Note: The Logical Sum performed above, is shown below in bit form.

111,	=	01 001 001
333,	=	11 011 011
Logical Sum	=	10 010 010 = 222

LSI - (16) - LOGICAL SUM (Indirect Address) 4 Cycles

Form in A the Logical Sum of the contents of A and the contents of the memory location whose address is the combined contents of the designated Tag register and one of the first 256 (decimal) locations. The location of one of these 256 locations is given in the second word of the instruction set.

Example: Assume A contains 010. Assume memory location, 1510, contains 301. Using the indirect mode, and location 0300, form in A the Logical Sum of the contents of A and the contents of address 1510.



INPUT-OUTPUT INSTRUCTIONS

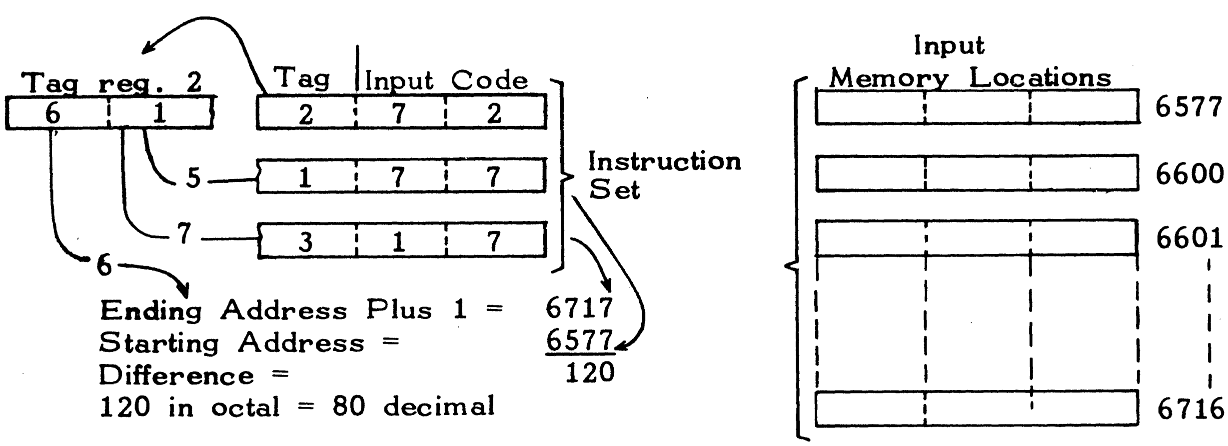
There are six instructions directly related to input-output functions. These are:

INN - INPUT NORMAL
OUT - OUTPUT NORMAL
IBI - INITIATE BUFFER INPUT
IBO - INITIATE BUFFER OUTPUT
INA - INPUT TO A
OTN - OUTPUT NO ADDRESS

<u>INN - (72) - INPUT NORMAL</u> (see p. 9 for timing)

Input a number of words to memory <u>starting</u> at the memory address contained in the designated Tag register and the second word of the instruction. The <u>ending</u> address plus 1, is contained in a third word immediately following the second word. Thus, this instruction set is composed of <u>three</u> words. (The Tag register designation indicated in the first word is automatically assigned as the Tag register designation for the ending address plus 1, in the third word.)

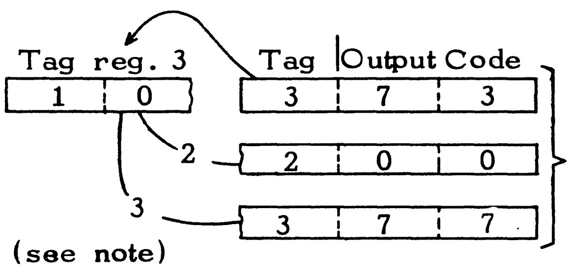
Example: Input 80 words to memory starting at octal address, 6577.



OUT - (73) - OUTPUT NORMAL

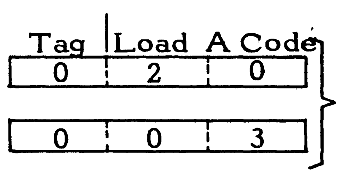
Output a number of words from memory starting at the memory address contained in the designated Tag register and the second word of the instruction set. The ending address plus 1, is contained in a third word immediately following. Thus, this instruction set is composed of three words. (The Tag register designation, indicated in the first word is automatically assigned as the Tag register designation for the ending address plus 1, in the third word.)

Example: Output 300 (decimal) words from memory, starting at octal address, 1200.

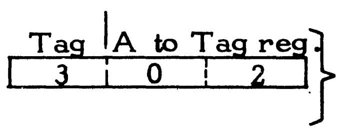


First 127 (decimal) words are outputed from octal addresses shown:

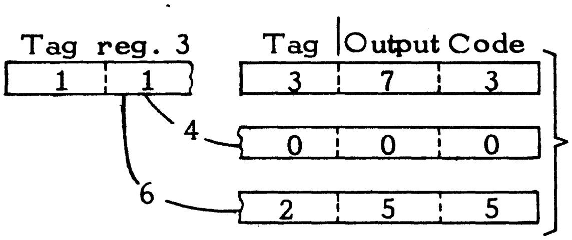
Ending Address Plus 1	=	1377
Starting Address	=	1200
Number of Words	=	177 = 127_{10}



Load A with 003.



Change Tag register, 3, by storing A at Tag register 3. Tag register 3 now contains 0011 (in bits).



Next 173 (decimal) words are outputed from octal addresses shown:

Ending Address Plus 1	=	1655
Starting Address	=	1400
Number of Words	=	255 = 173_{10}

127 + 173 = total 300 words

NOTE:

The "ending address plus 1" of 1377 above, resulted in a "gap"--that is, no output came from this register. The reason is that quartic address, 1377, falls at a "boundary address" as far as the addressing logic of the Tele-Programmer is concerned. "Boundary addresses" are those, which when incremented by 1, cause a change to occur in any one of the 4 leftmost address bits. This in turn, requires a change in the Tag register (as above). There are 16 such "boundary addresses" in the whole 4096 registers. This condition is not serious due to the following alternatives:

(a) If output follows input or vice versa such "gaps" would have existed in the identical places anyway, and thus are of no consequence.

(b) If one wishes, he can fill the gap location by loading one word into A and storing at the gap address.

(c) By effective memory allocation, boundary addresses can often be entirely avoided.

(d) Buffered operations do <u>not</u> have this situation.

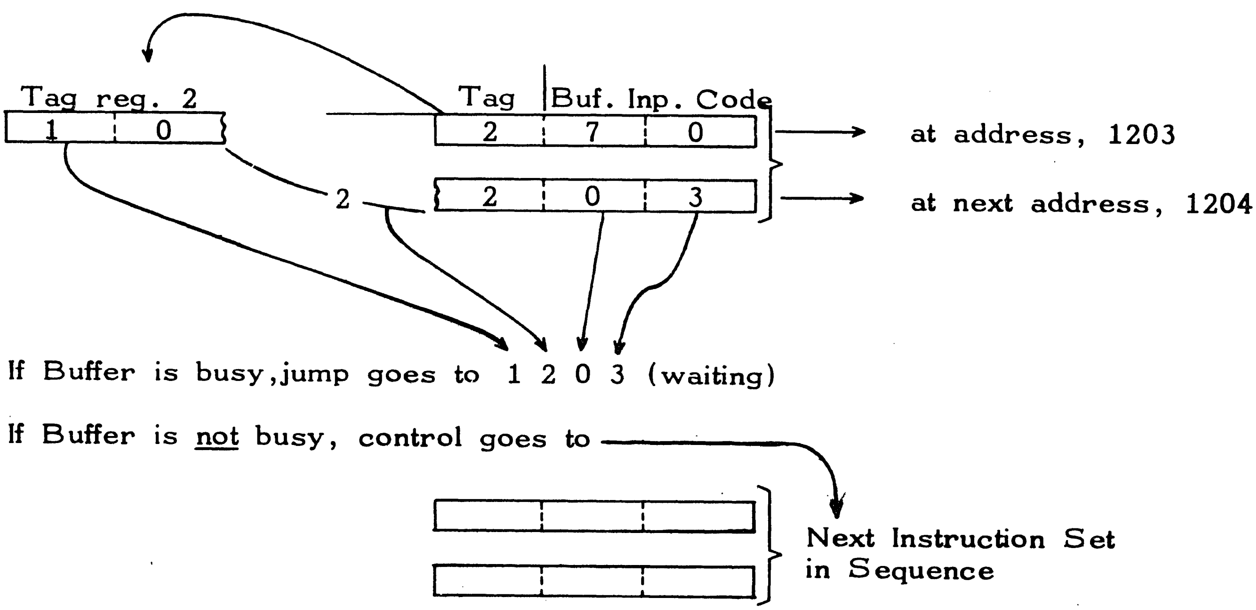
The previous example was given to indicate that a change in address which changes <u>any one</u> of the 4 leftmost bits of the 12-bit address, requires a corresponding change in the contents of the Tag register. It should be apparent, that the maximum transfer <u>without changing the Tag register</u> is 256 (decimal words.

IBI - (70) - INITIATE BUFFER INPUT 1 cycle, 2 cycles if jump is made.

Before using this instruction, the starting address of the buffer transfer is sent to BER, and the ending address plus 1 is sent to BXR (see these instructions).

This instruction initiates the input buffer cycle. If the buffer channel is not busy, control goes to the next instruction following the second word of the instruction set. If the buffer channel is busy, a jump occurs to the memory location whose combined address is contained in the designated Tag register and the second word of the instruction set.

Example: Initiate buffer input, and if busy wait until not busy. Assume the instruction is given at the location whose octal address is, 1203.

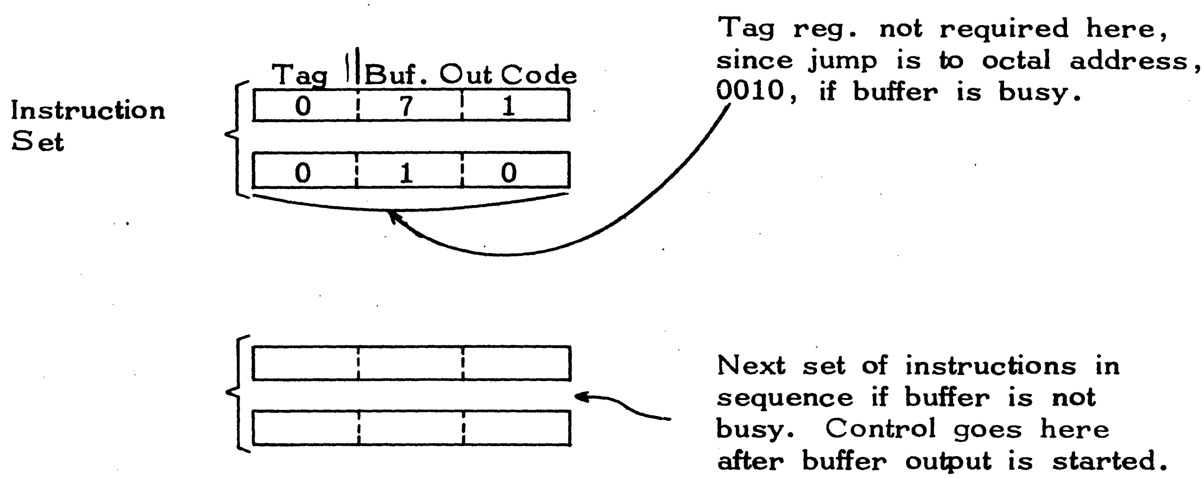


IBO - (71) - INITIATE BUFFER OUTPUT 1 cycle, 2 cycles if jump is made.

Before using this instruction, the starting address of the buffer transfer must be sent to BER, and the ending address plus 1 must be sent to BXR (see these instructions).

This instruction initiates the output buffer cycle. If the buffer channel is busy, a jump occurs to the combined memory address given in the designated Tag register and the second word of the instruction set. If the buffer channel is not busy, control goes to the next sequential instruction following the instruction set.

Example: Initiate buffer output and if busy jump to octal address 0010.



INA - (76) - INPUT TO A

This instruction inputs one word from a previously selected input device to the A register.

Example: Assume a previous instruction (see EXF) has selected the paper tape reader for input. Input one frame (one word) to A.

Tag	INA Code	
0	7	6

Note: This is a single word instruction, and the Tag register designation is always zero.

OTN - (74) - OUTPUT NO ADDRESS

This instruction outputs one word. This word is the second word of the instruction set.

Example: Assume a previous instruction has selected the Printer. Output the number 0102.

	Tag	Output Code	
Instruction Set	0	7	4
	1	0	2

Note: The Tag register designation is always zero in this instruction.

CONTROL INSTRUCTIONS

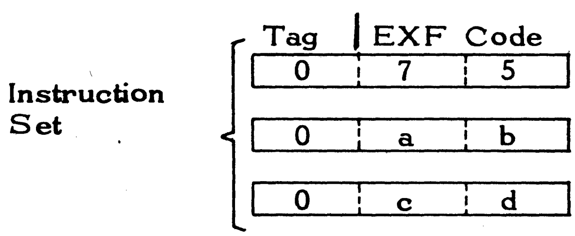
Five Control instructions are available:

EXF - EXTERNAL FUNCTION
CIL - CLEAR INTERRUPT LOCKOUT
CBC - CLEAR BUFFER CONTROLS
ERR - ERROR STOP
HLT - HALT

EXF - (75) - EXTERNAL FUNCTION

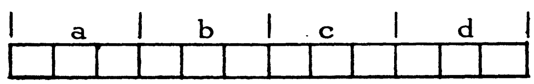
This instruction is used to select an external input or output device to communicate with the TeleProgrammer. The select function is accomplished by sending out on the output lines a 12-bit "function code". Each external device is capable of recognizing and interpreting only its own unique code. Thus, the programmer by selecting different external function codes can use this same instruction to select all external devices.

The 12-bit function code is contained in the second and third words of the three words which make up this instruction set. The format of the three words are best described by the following:



where
a b = The upper 6 bits of the function code.

c d = The lower 6 bits of the function code.



The 12-bit Function Code

Note: If the external device cannot be selected the TeleProgrammer halts.

Example: Request the status of the typewriter (ready or not ready), if busy, wait; request typewriter input; and input to A.

Address	Tag	EXF Code	
(0010) =	0	7	5
(0011) =	0	4	2
(0012) =	0	4	0

Request status of the typewriter. The octal function code for this is 42 40

Address	Tag	Input to A	
(0013) =	0	7	6

Input status response to A

Address	Tag	Non Zero Jump	
(0014) =	0	6	1
(0015) =	0	1	0

If A = 0, typewriter is ready, continue. If A $\neq$ 0, it is not ready, jump back to address 0010 (wait). (see note)

Address	Tag	EXF Code	
(0016) =	0	7	5
(0017) =	0	4	2
(0020) =	0	2	0

Request typewriter to input a character to A. The octal function code for this is: 42 20

Address	Tag	Input to A	
(0021) =	0	7	6

Input the character to A

Note: In the jump back to address 0010 above, no Tag register is required since the octal address is one whose significant bits can be expressed in 8 bits.

CIL - (13) - CLEAR INTERRUPT LOCKOUT

NOTE

A do nothing (02) instruction should be used at interrupt locations 10, 20, 30 and 40 when such interrupt levels are used; then use the 013 or 113 instruction.

This instruction clears the interrupt lockout flip flop (FF). This instruction must be programmed at the end of every routine which is initiated by the interrupt. This instruction returns control to the main program.

Example: Assume an interrupt has occurred and a routine entered. At the end of this routine show the instruction required to clear the Interrupt Lockout and return control to the Main Program.

Tag	CIL Code	
1	1	3

Note: In this instruction, the Tag designation becomes a part of the function code itself. It can only be 0 or 1. Thus, to return to main program after clearing interrupt lockout, the Tag designation must be 1. If zero, control continues in sequence.

CBC - (07) - CLEAR BUFFER CONTROLS

This instruction has the effect of sending a zero to buffer control and thus putting that device in a "ready state". If this instruction is used <u>during</u> a buffer operation, it will stop the buffer.

Example: Clear buffer control.

Tag	CBC Code	
0	0	7

A Tag register designation is ignored in this single word instruction.

Two STOPS are available; these are:

ERR = ERROR STOP
HLT = HALT STOP

ERR - (000) - ERROR STOP

This is an illegal instruction -- as such, it can be used as an Error Stop.

Example: Use the Error Stop instruction.

0	0	0	Error Stop

HLT - (77) - PROGRAM STOP

This instruction is used to bring the program to a halt.

Example: Use the STOP instruction.

Tag	Stop Code		
0	7	7	Program Stop

CHAPTER TWO

OPERATION

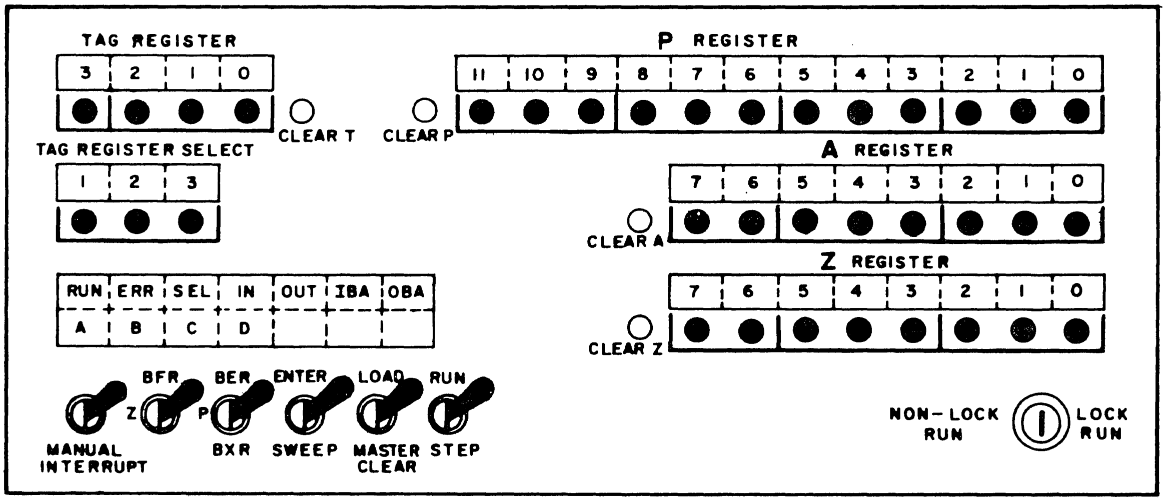


Figure 2-1 8092 Operator's Panel

TeleProgrammer OPERATOR's CONSOLE

The 8092 TeleProgrammer Operator's Panel consists of several displays and switches necessary for the operation of the TeleProgrammer. The panel (see figure 2-1) contains six display windows, six switches, and a lock switch. Four of the display windows can display in binary the contents of nine 8092 registers. Buttons beneath these displays clear and enter data into the P, A, Z, and Tag registers (the only registers into which data may be entered or cleared). A fifth window contains information as to which Tag register has been selected. The sixth window contains the operating lights which indicate the status of operation of the TeleProgrammer. At the bottom of the panel are located all the operating and mode switches. The operation of these switches is explained below:

SWITCHES

Manual Interrupt	-Momentary depression causes the TeleProgrammer to enter an interrupt routine to determine the nature of the interrupt.
BFR, Z	-This 3-position switch chooses the register that is to be displayed in the 8-bit Z register display. Up - Displays the last word processed during the last buffer operation (BFR register) Center - Shows the current contents of the Z register (Z register). Down - Not assigned.
BER, P, BXR	-This 3-position switch chooses the register to be displayed in the 12-bit P register display. Up - Displays the lowest-order 10 bits of the address of the last word transferred out, or the next word to be transferred in on the buffer channel (BER) register. Tag 3 must be referenced for the highest-order 2 bits of the address.

Center - Displays the address of the current instruction (P register).

Down - Displays the lowest-order 10 bits of the LWA + 1 of the buffer area (BXR) register. Tag 3 must be referenced for the highest-order 2 bits of the address.

ENTER/SWEEP

-Sweep is used to display the contents of core storage locations. Enter is used for entering information into core storage from the console.

LOAD/MASTER CLEAR

- LOAD position allows specially prepared paper tapes to be read into storage by the paper tape reader.

Master CLEAR performs a TeleProgrammer clear which:

a. Clears the registers
b. Clears the control flip-flops
c. Clears all waiting interrupts and removes interrupt lockout.

Note: The master clear does not alter core storage.

RUN/STEP

Up - In RUN position, a program is executed at high speed starting at the location specified by the P register.

Center - Center position stops the computer program. If the switch is in RUN and an ERR or HLT instruction is executed, the switch must be returned to neutral and then placed in RUN to continue computation.

Down - In STEP position, one storage cycle of an instruction is executed each time the switch is set; a program may be executed one instruction at a time for debugging.

NON-RUN LOCK
RUN LOCK

In the Lock position all other switches are disabled and the TeleProgrammer is locked in the RUN position.

In the non-lock position, the console switches are enabled and the TeleProgrammer programs can be operated and modified from the console.

DISPLAYS

Z REGISTER

-This display known as the Z register group displays the Z and BFR registers in accordance with the setting of the BFR, Z switch.

A Register

-Displays the current contents of the A register.

P Register -This display known as the P register group displays the BER, P, and BXR registers in accordance with the setting of the BER, P, BXR switch.

TAG REGISTER -This display indicates the Tag register currently being referenced by an instruction. The contents of any Tag register may be displayed by depressing one of the buttons directly below the select indicators. Depressing one of the select buttons also enables the Tag registers to be manually set or cleared.

STATUS INDICATORS

RUN -Indicates that the TeleProgrammer is in RUN status. This does not necessarily indicate that instructions are being executed.

ERR -Indicates that a timing fault has occurred.

SEL -Displayed each time an EXF instruction is executed; remains until selection is completed. A constant display of SEL with no apparent input/output action usually indicates the TeleProgrammer has attempted an illegal selection.

<u>IN</u>	-Displayed during all normal input operations. A constant display of IN with no apparent input action usually indicates that input was attempted without proper unit selection. IN is also displayed when the TeleProgrammer is waiting for an external device to supply data.
<u>OUT</u>	Displayed during all normal output operations. A constant display of OUT with no apparent output action usually indicates that output was attempted without proper unit selection.
<u>IBA</u>	Displayed during all buffer input operations.
<u>OBA</u>	Displayed during all buffer output operations.
<u>A, B, C, or D</u>	Indicates which storage reference cycle will be executed at the next operation of the Run/Step switch. When a master clear is performed, D is displayed indicating that the next operation to be executed, when the Run/Step switch is operated, will be to fetch the instruction from memory at the address indicated by the P register.

STARTING THE 8092 TeleProgrammer

1) Be sure the TeleProgrammer is plugged into proper power source and room temperature is within the prescribed limits.
2) Turn on the cabinet power, then turn on the power supply.
3) Master clear by momentarily pressing Load/Clear switch to Clear.
4) When the ERR light goes out, the TeleProgrammer is ready to operate. If repeated master clears do not turn the Red ERR light off, turn off the 8092 and call maintenance.

LOADING A PROGRAM OR DATA

Paper Tape Load Format

1) Master Clear
2) Turn on reader
3) Insert paper tape in reader
4) Set P to starting location
5) Set Load/Clear switch to LOAD
6) Set Run/Step switch to RUN. Paper tape will load and TeleProgrammer will stop.

ENTERING DATA FROM THE TeleProgrammer CONSOLE

1) Master clear. Set Enter/Sweep switch to ENTER.
2) Set P to location into which data is to be entered.
3) Enter one word of data into the A register.
4) Set Run/Step switch to STEP, once. At this point A is clear and the data word is in storage and in Z.
5) If data is to be entered into consecutive locations, go to step 3 and P will be advanced by one on step 4. If data is to be entered into non-consecutive locations, clear P. Go to step 2.

EXAMINING THE STORAGE CONTENTS

1) Master clear. Set Enter/Sweep switch on SWEEP

2) Set P to location to be examined.

3) Press Run/Step switch to STEP, once. The contents of the location specified by P will appear in Z.

4) To examine consecutive locations, go to step 3 and P will be advanced by one on step 3. To examine non-consecutive location, clear P, go to step 2.

CHAPTER THREE

A BRIEF LOGICAL DESCRIPTION OF THE TELEPROGRAMMER

Input/Output Section

The input/output (I/O) Section contains one normal (or direct) channel and one buffered channel. Each channel can communicate with five units of peripheral equipment.

The normal channel communicates with external equipment under program control only. There are no provisions for the external equipment to initiate a data transfer except under program control.

The buffered channel communicates with external equipment asynchronously to the main program. It can transfer data in one direction only until changed by program control. In other words, the buffer channel can input to main storage or output from main storage while not under main program control. However, it cannot input and output alternately without having been so instructed by the main program. Once an input buffering or an output buffering operation is initiated, it continues until completed or until cleared by the main program.

The buffer I/O channel has three registers associated with it. These are the Buffer Entrance Register (BER register), the Buffer Exit Register (BXR register), and the Buffer Data Register (BFR register). The BER register holds the buffer starting address and is advanced by one for each buffer cycle. The BXR register holds the buffer ending address, and when BER = BXR the buffer operation is complete. The BFR register holds the input or output word for transfer to or from external equipment.

The buffer channel may also be used as a normal channel whenever the buffer is not busy.

Interface control is maintained by the control section on a Ready-Resume basis. Within the control section is a separate buffer control section which controls the buffering operations on the same Ready-Resume basis.

Program Step

A program step in the TeleProgrammer is one storage reference cycle. Normally the steps proceed as: 1) Read instruction into control section, 2) Read address of operand (one or two steps) and 3) Perform indicated instruction.

The TeleProgrammer instruction is basically a 2-word instruction contained in 2 sequential storage locations. The first word of the instruction contains: the instruction in the lower 6 bits and the tag bits (TAG register reference bits) in the upper 2 bits. The second word of the instruction contains: the operand (no address mode), the lower 8 bits of a 12-bit address (memory address mode) or the address of one of the first 256 storage locations (indirect address mode).

Arithmetic Section

The arithmetic section of the TeleProgrammer consists of 3 registers and a borrow pyramid. The three registers are the A register, the A' register (which is the accumulator register) and the Z register.

All arithmetic functions (add, subtract and logical operations) are performed by the borrow pyramid which is integrated with the A' register. Inputs to the borrow pyramid are from the A register and the Z register.

Shifting of the A register is accomplished via the borrow pyramid and is confined to a left shift one bit, in the TeleProgrammer. This shifting is a circular shift where the highest order bit is shifted into the lowest order bit position.

The borrow pyramid forms the results of arithmetic operations in a subtractive manner; so that, addition is performed by complementing the Z register and subtracting. Subtraction is a direct process, and logical operations are performed similarly to addition.

Interrupt

The interrupt feature gives the TeleProgrammer four unique interrupt levels which can be utilized in the programming of the TeleProgrammer. The four interrupt levels in order of priority are:

1) Manual Interrupt 10
2) Buffer Interrupt 20
3) External Interrupt 30
4) External Interrupt 40

Recognition of an interrupt by the TeleProgrammer forces the TeleProgrammer to start an Interrupt recognition routine which starts at memory location 10 or 20 or 30 or 40 depending on the interrupt activated.

Interruption of the main program can only occur on a 'D' cycle and the occurrence of an interrupt causes the TeleProgrammer to store the address at which it was interrupted and jump to locations 10 or 20 or 30 or 40. At these locations must be the start of a routine which determines the nature of the interrupt. At the end of this routine must be a Clear-Interrupt Lock-out instruction which causes the TeleProgrammer to jump back into the main program at the same address it was at when interrupted. If the interrupt feature is to be used, memory locations 10, 20, 30 or 40 should not be used for the main program or storage.

<u>Interface Control</u>

At the interface of the TeleProgrammer are the same basic control lines and data lines as in the CONTROL DATA 160-A computer.

Storage Section

The storage section of the TeleProgrammer is a high-speed magnetic-core storage system providing non-volatile, random-access storage for 2048 or 4096 8-bit words. Transfer of the words into and out of storage is under control of the control section. For each storage reference cycle,the program-address register (P register) is advanced by 1 to form the address of the next storage location. This address is then entered into the storage access register (S register) where it is translated to a unique selection of one vertical line and one horizontal line selecting 1 core in each of the 8 planes.

After translation, the selected lines are pulsed simultaneously by Read/Write drivers to give a coincident current through the selected core. Normally this would write a "1" in the selected core. If that plane is "inhibited" however, an "0" will be written in that core. The inhibit effectively cancels the effect of the vertical write pulse so that only a half-write current will exist in the core.

Storage Sequence

The storage sequence is divided into four basic portions which accomplish the Read/Write control of the storage section. Every storage sequence is as follows:

1) Divert - select one of eight vertical and one of eight horizontal lines from the Read/Write drivers.

2) Read - select one of eight vertical and one of eight horizontal Read/Write drivers in the read mode which drives the core to its "0" state.

3) Inhibit - cancel the effect of the write pulse and allow the core to remain in the "0" state.

4) Write - select the same Read/Write driver as on read, and drive the core to its "1" state if the inhibit pulse is absent.

There are two basic registers associated directly with the storage section. These are the storage address register (S register) and the transfer register (Z register).

The S register is a 12-bit register that holds the storage address during the storage reference cycle. This register has storage capabilities only and is set from the P register, the Buffer Entrance Register (BER register), the A register or the Z register, depending on the instruction being performed.

The Z register is the main transfer and data handling register in the Tele-Programmer. All outputs from the core storage enter the Z or BFR registers, and all inputs to the core storage come from the Z or BFR registers. The Z register also has inputs from the A register, the normal input channel and the buffer input channel in the normal mode. Outputs from the Z register feed the borrow pyramid, the S register, the F register, the normal output channel, and the buffer output channel in the normal mode.

Control Section

The control section of the TeleProgrammer consists of the timing controls, the function translation, and the TAG registers.

Timing Controls

Timing of the operations of the TeleProgrammer is controlled by the timing chain and the primary timing controls. The timing chain is an 8-stage ring counter which recirculates three times for every storage reference cycle to produce a chain of 24, successive, unique pulses. A resynchronizing circuit is employed to insure the timing chain starting on the same clock phase each storage reference cycle.

Function Translation

The instruction control or function control of the TeleProgrammer is achieved by the F register and the function translators. The F register is translated to determine the control and data transfer sequence for any given instruction. The translators are carefully integrated with the timing controls to insure proper operation of the TeleProgrammer.

Address-Tag Registers

Also in the control section are three address-Tag registers each of 4 bits length. These registers are referenced by the tag bits of the instruction word (which are also translated).

The Tag registers are capable of modification at any point in the program from the A register. The upper two bits of the Tag register 3 are used as the buffer channel Tag register. Also, if Tag register 0 is referenced, the address will automatically be one of the first 256 storage locations since Tag register 0 is non-existent.

APPENDIX

APPENDIX A

TOSAS -- A TELEPROGRAMMER ASSEMBLER

Preface

The TeleProgrammer is easily programmed in machine language using the previous descriptions and references of this manual. Those Control Data Customers and analysts who have recourse to either a 160 or 160-A Computer, can also use "TOSAS" - the TeleProgrammer One Sixty Assembler System. TOSAS is easily implemented by adopting very slight modifications of the OSAS or OSAS-A assembly language. These modifications are described in this Appendix. Full descriptive manuals of OSAS or OSAS-A are available and can be obtained by writing to:

Industrial Data Processing Division
Control Data Corporation
9549 Penn Ave. South
Minneapolis, Minnesota

Description of TOSAS

TOSAS uses all the rules of OSAS or OSAS-A. With the exception of minor changes in the coding forms used, along with the adoption of one or two limitations, the two assemblers are the same. The differences are listed in detail below and followed with an example program to indicate the changes.

Providing for Different Function Codes:

The 160 and 160-A Computers employ 6-bit function codes. The TeleProgrammer also uses 6-bit function codes. However, the octal codes are different. To overcome this difference, TOSAS requires a "function Code Identification Listing" as part of the TOSAS program. This identification simply lists the mnemonic codes of the TeleProgrammer under "LOCATION" (cols. 2 - 8 in OSAS coding form); the pseudo OP Code, EQU, under "OP" (cols. 10 - 13); and the TeleProgrammer octal Function Codes under "ADDITIVE" (cols. 23 - 29). "COMMENTS" can appear as usual. A sample of identification listing is shown on page A-6. (The octal addresses in the leftmost column on page A-6 were assigned by the assembler for the problem of which this listing is an example.

Use of "CON"

In OSAS, the pseudo OP, "CON" is used to set aside the first 64 registers (octal address, 0000 through 0077) for constants which follow the code, "CON". In TOSAS, this pseudo OP can be used exactly the same way. However, the TeleProgrammer provides for 256 low core address (octal addresses 0000 through 0377). This means that if the programmer desires to reserve low core area beyond the first 64 locations, he must use separate symbolic tags under "LOCATION" preceeded by the pseudo OP, "PRG" or by using the EQU pseudo OP as shown.

Example: Assume one wants to store octal constants: 5, 27, 31 and LG at respective octal locations 0076, 0077, 0100 and 0101.

LOCATION	OP	ADDRESS	ADDITIVE	COMMENTS
	CON	76		
			5	
			27	
	PRG	100		
			31	
LG	EQU	101		

Size of Numerics Under ADDITIVE Column

Since the TeleProgrammer involves an 8-bit word length instead of 12 bits, the size of the octal numbers (quantities and addresses) must not exceed 8 bits. Thus, the effective range is 000 through 377. In addition, 100, 200, or 300, must appear in the ADDITIVE column opposite the mnemonic code to indicate respectively the use of Tag registers 1, 2, or 3. (See examples.)

Difference in Coding

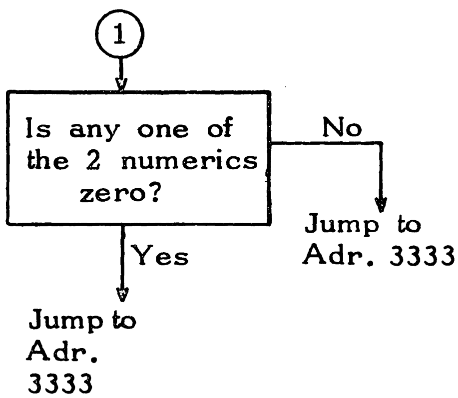
The same coding forms, as used in OSAS or OSAS-A, can be used in TOSAS. The difference is in the placement of mnemonic OP codes. In TOSAS, all TeleProgrammer mnemonic OP codes are placed under the "ADDRESS" column rather than the "OP" column. Symbolic addresses can be placed under the mnemonic, in the "ADDRESS" column, or in the "ADDITIVE" column. Octal numerics or decimal numerics (with letter "D") can be placed under "ADDITIVE". Also, 100, 200, or 300 must appear in the ADDITIVE column directly opposite the mnemonic code of the ADDRESS column to respectively indicate use of Tag register 1, 2, or 3. Thus in the example on the next page the third line of coding LDM 200
106

indicates that Tag register 2 is used with LDM, and contains the upper 4 bits of address 106 which follows.

To aid the reader in the TOSAS concept, a sample example is programmed, using TOSAS. A few sheets of the routine, on OSAS coding form, and the corresponding TOSAS listing are shown on pages A-7 through A-12. (Since this example used the lower 256 core memory locations, the reader should note that the use of 100, 200, or 300 opposite the mnemonic codes is reserved for the occasions when jumps or references beyond the first 256 registers are made. In such instances, Tag registers are required.

Example: Assume octal address 3106 and 3107 contain unknown numerics. If any one of these numerics is zero, jump to address, 3255; otherwise, jump to T1 (where T1 = Adr. 3333).

Flow Chart



TeleProgrammer Octal Code

0LDN 0 06 2ATT	Store octal, 3, in three left bits of Tag Reg. 2
2LDM 1 06	Contents of Adr. 3106 to A
2ZJP 2 55	If zero, jump to Adr. 3255
2LDM 1 07	Contents of Adr. 3107 to A
2ZJP 2 55	If zero, jump to Adr. 3255
2UJP 3 3 3	Jump to Adr. 3333

TOSAS Coding (using OSAP Coding Form)

LOCATION	OP	ADDRESS	ADDITIVE	COMMENTS
		LDN	6	Octal 3 to Tag Reg. 2
		ATT	200	
		LDM	200 106	Contents of Adr. 3106 to A
		ZJP	200 255	If zero, jump to 3255
		LDM	200 107	Contents of Adr. 3107 to A
		ZJP	200 255	If zero, to Adr. 3255
		UJP T1*		Jump to T1

* Where T1 must be previously identified in the program by a statement such as:

LOCATION	OP	ADDRESS
T1	EQU	3333

IDENTIFICATION LISTINGS

	Location	OP	Additive	Comments
0000	ERR	EQU	0	ERROR STOP
0001	SHA	EQU	1	SHIFT A LEFT ONE BIT
0002	ATT	EQU	2	A TO TAG REGISTER
0003	TTA	EQU	3	TAG REGISTER CONTENTS TO A
0004	ABR	EQU	4	A TO BUFFER ENTRANCE REGISTER
0005	ABX	EQU	5	A TO BUFFER EXIST REGISTER
0006	BER	EQU	6	CONTENTS OF BER REGISTER TO A
0007	CBC	EQU	7	CLEAR BUFFER CONTROLS
0010	LPN	EQU	10	LOGICAL PRODUCT NO ADDRESS
0011	LPM	EQU	11	LOGICAL PRODUCT MEMORY ADDRESS
0012	LPI	EQU	12	LOGICAL PRODUCT INDIRECT ADDRESS
0013	CIL	EQU	13	CLEAR INTERRUPT LOCKOUT
0014	LSN	EQU	14	LOGICAL SUM NO ADDRESS
0015	LSM	EQU	15	LOGICAL SUM MEMORY ADDRESS
0016	LSI	EQU	16	LOGICAL SUM INDIRECT ADDRESS
0020	LDN	EQU	20	LOAD A NO ADDRESS MODE
0021	LDM	EQU	21	LOAD A MEMORY
0022	LDI	EQU	22	LOAD A INDIRECT
0025	LCM	EQU	25	LOAD COMPLEMENT TO A MEMORY
0026	LCI	EQU	26	LOAD COMPLEMENT TO A INDIRECT
0030	ADN	EQU	30	ADD NO ADDRESS
0031	ADM	EQU	31	ADD MEMORY ADDRESS
0032	ADI	EQU	32	ADD INDIRECT ADDRESS
0034	SBN	EQU	34	SUBTRACT NO ADDRESS
0035	SBM	EQU	35	SUBTRACT MEMORY ADDRESS
0036	SBI	EQU	36	SUBTRACT INDIRECT ADDRESS
0041	STM	EQU	41	STORE A MEMORY
0042	STI	EQU	42	STORE A INDIRECT
0051	RAM	EQU	51	REPLACE ADD MEMORY ADDRESS
0055	RAO	EQU	55	REPLACE ADD ONE MEMORY ADDRESS
0060	ZJB	EQU	60	JUMP, IF CONTENTS OF A = 0
0061	NZP	EQU	61	JUMP, IF CONTENTS OF A (0
0062	PJP	EQU	62	JUMP, IF CONTENTS OF A 0 POSITIVE
0063	NJP	EQU	63	JUMP, IF CONTENTS OF A 0 NEGATIVE
0064	UJP	EQU	64	UNCONDITIONAL JUMP
0070	IBI	EQU	70	INITIATE BUFFER INPUT
0071	IBO	EQU	71	INITIATE BUFFER OUTPUT
0072	INN	EQU	72	INPUT NORMAL
0073	OUT	EQU	73	OUTPUT NORMAL
0074	OTN	EQU	74	OUTPUT, NO ADDRESS
0075	EXF	EQU	75	EXTERNAL FUNCTION
0076	INA	EQU	76	INPUT TO A
0077	HLT	EQU	77	HALT
0000		END		COMPLETE ASSEMBLY

CONTROL DATA

OSAS/OSAS-A CODING FORM

A TOSAS Assembly Program
(Part of "XLATE" Routine)

PAGE NO. ____
DATE ____
PROGRAMMER ____

LOCATION	OP	ADDRESS	ADDITIVE	COMMENTS
INPUTA		LDM		ROUTINE M/T INPUT A
			FLAG1	
		ZJB		TEST FLAG IF 1st TIME
			INPUT1	NO JUMP
		LDN		
			O	
		STM		
			FLAG1	
		EXF		T/H STATUS
			11	RS1
			02	
		INA		
		SBN		TEST IF T/H ON
			41	LØAD PØINT AND CØDE
		NZP		
			INPUT2	NOT READY
INPUT1		EXC		T/H STATUS
			11	RS1
			02	
		INA		
		LPN		SAVE BUSY BIT
			202	
		NZP		
			INPUT1	TRY AGAIN
INPUT6		EXF		
			11	READ BCD LOW
			32	
		INN	300	
			FWA	
			LWA	
INPUT3		EXF		TAKE STATUS ØF
			11	READ

FORM 136e – REPLACES 136a&b   Use shaded columns only if first symbol character is + or -, or for 4 character OP code

CONTROL DATA

OSAS/OSAS-A CODING FORM

PAGE NO. ______
DATE ______
PROGRAMMER ______

LOCATION	OP	ADDRESS	ADDITIVE	COMMENTS
			02	
		INA		
		STM		STORE STATUS
			TEM2	
		LPN		
			200	
		NZP		
			INPUT2	
		LDM		LOAD STATUS
			TEM2	BACK IN
		LPN		
			24	SAVE EØF & PARITY
		ZJB	100	
			WORK	JUMP TO WORK ROUTINE
		LPN		
			04	SAVE PARITY BIT
		NZP		
			INPUT4	
		UJP	300	JUMP TO EØM ROUTINE
			EØM	
INPUT2		LPN		SET RETURN
			63	JUMP FOR
		STM	300	LOCAL TROUBLE
			RETURN	
		LDN		
			INPUT1	
		STM	300	
			JUMP	
		UJP	300	JUMP TO LOCAL TROUBLE
			LOTROU	ROUTINE
INPUT4		EXF		PARITY ON READ
			11	

FORM 138e — REPLACES 138a&b Use shaded columns only if first symbol character is + or -, or for 4 character OP code

CONTROL DATA CORPORATION

OSAS/OSAS-A CODING FORM

PAGE NO. ______
DATE ______
PROGRAMMER ______

2 LOCATION	10 OP	15 ADDRESS	23 ADDITIVE	31 COMMENTS
			25	BACK SPACE ØNE RECORD
		LDM		
			TRY3	
		NJP		
			INPUT5	
		SHA		
		STM		STØRE BACK AT
			TRY3	TRY 3
		UJP		
			INPUT6	READ AGAIN
INPUT5		LDN		
			021	SET TRY 3
		STM		
			TRY3	
		LDN		
			063	
		STM	300	
			RETUR1	
		LDN		
			INPUT6	
		STM	300	
			JUMP1	
		UJP	300	
			TIME	
	PRG		400	

FORM 138e — REPLACES 138a&b Use shaded columns only if first symbol character is + or -, or for 4 character OP code

Listing From Previous TeleProgrammer "XLATE" Routine

0130	0021	INPUTA	LDM		ROUTINE M/T INPUTA
0131	0060			FLAG1	
0132	0060		ZJB		TEST FLAG IF 1ST TIMF
0133	0173			INPUT1	NO JUMP
0134	0020		LDN		
0135	0000			0	
0136	0041		STM		
0137	0060			FLAG1	
0140	0075	INPUT7	EXF		TAKE STATUS OF TAPE
0141	0011			11	
0142	0002			2	
0143	0076		INA		
0144	0041		STM		SAVE STATUS
0145	0066			TEM4	
0146	0010		LPN		SAVE NOT READY BIT
0147	0002			2	
0150	0061		NZP		TAPE NOT READY
0151	0237			INPUT2	
0152	0021		LDM		BRING STATUS BACK
0153	0066			TEM4	
0154	0010		LPN		IS IT BUSY
0155	0200			200	
0156	0061		NZP		YES TRY AGAIN
0157	0140			INPUT7	
0160	0021		LDM		
0161	0066		TEM4		BRING STATUS BACK
0162	0034		SBN		CHECK LOAD POINT
0163	0041			41	
0164	0060		ZJB		GO ON TO READY
0165	0203			INPUT6	
0166	0075		EXF		REWIND TAPE TO LOAD POINT
0167	0011			11	
0170	0022			22	
0171	0064		UJP		GO TAKE STATUS AGAIN
0172	0140			INPUT7	
0173	0075	INPUT1	EXF		T/H STATUS
0174	0011			11	RS1
0175	0002			2	
0176	0076		INA		
0177	0010		LPN		SAVE BUSY BIT
0200	0202			202	
0201	0061		NZP		
0202	0173			INPUT1	TRY AGAIN

0203	0075	INPUT6	EXF		
0204	0011			11	
0205	0032			32	READ BCD LOW
0206	0372		INN	300	
0207	0000			0	FWA
0210	0120			120	LWA
0211	0075	INPUT3	EXF		TAKE STATUS OF
0212	0011			11	READ
0213	0002			2	
0214	0076		INA		
0215	0041		STM		STORE STATUS
0216	0064			TEM2	
0217	0010		LPN		
0220	0200			200	
0221	0061		NZP		
0222	0211			INPUT3	
0223	0021		LDM		LOAD STATUS
0224	0064			TEM2	BACK IN
0225	0010		LPN		
0226	0024			24	SAVE EOF AND PARITY
0227	0160		ZJB	100	
0230	1213			WORK	JUMP TO WORK ROUTINE
0231	0010		LPN		
0232	0004			4	SAVE PARITY BIT
0233	0061		NZP		
0234	0251			INPUT4	
0235	0364		UJP	300	JUMP TO EOM ROUTINE
0236	0676			EOM	
0237	0020	INPUT2	LDN		SET RETURN
0240	0063			63	JUMP FOR
0241	0341		STM	300	LOCAL TROUBLE
0242	0652			RETURN	
0243	0020		LDN		
0244	0173			INPUT1	
0245	0341		STM	300	
0246	0653			JUMP	
0247	0364		UJP	300	JUMP TO LOCAL TROUBLE
0250	0636			LOTROU	ROUTINE
0251	0075	INPUT4	EXF		PARITY ON READ
0252	0011			11	
0253	0025			25	BACK SPACE ONE RECORD
0254	0021		LDM		
0255	0115			TRY3	
0256	0063		NJP		
0257	0265			INPUT5	
0260	0001		SHA		
0261	0041		STM		STORE BACK AT
0262	0115			TRY3	TRY3

TeleProgrammer "XLATE" Routine Listing (Con't)

0263	0064		UJP	
0264	0173			INPUT1 READ AGAIN
0265	0020	INPUT5	LDN	
0266	0021			21
0267	0041		STM	SET TRY 3
0270	0115			TRY3
0271	0020		LDN	
0272	0063			63
0273	0341		STM	300
0274	0672			RETUR1
0275	0020		LDN	
0276	0173			INPUT1
0277	0341		STM	300
0300	0673			JUMP1
0301	0364		UJP	300
0302	0656			TIME

The previous listing is <u>part</u> of a Magnetic Tape input routine. As such, the reader should be aware of the fact that several symbolic tags are used (for example, FLAG 1, TEM 2, TEM 4, WORK, etc.) which were identified by "EQU" statements on other parts of the total program. Likewise, notes, page number, etc. refer to the total program from which the example was taken.

APPENDIX B

PROGRAMMING EXAMPLES

Example 1
Servicing the Interface

A small message switching system is composed of ten full duplex lines operating at rates of 100 words per minute. Each time the input interface is serviced (once each 100 milliseconds) each of the ten input terminal units (TTU) supplies one 8 bit character, where each character contains 7 bits of data and 1 parity bit).

The octal select codes of the ten TTU units are identical to the octal memory addresses that are used to store the inputs. These addresses are:

0420	0425
0421	0426
0422	0427
0423	0430
0424	0431

Each time the input interface is serviced, <u>one</u> character from each of the ten TTU locations is read into a corresponding Raw Data Register (RDR) in the TeleProgrammer memory. Assuming the Raw Data registers start at octal address, 0600, the program follows:

Program Location	Instructions Tag	Codes	Cycles	Action Performed
0460 0461	0 0	LDN 01	2	Load A with 1
0462	1	ATT	1	Bits, 001, go to Tag reg. 1
0463 0464 0465	0 0 0	EXF 04 20	3	Select the TTU, starting with the first TTU.
0466	0	INA	2	Input the character from the selected TTU to the A register.
0467 0470	1 2	STM 00	3	Store character in A at desired memory location
0471 0472	1 0	RAO 70	4	Add 1 to memory location where characters are stored.
0473 0474	1 0	RAO 65	4	Add 1 to TTU select address. This also tests last TTU address.
0475 0476	0 0	SBN 32	2	Subtract one more than number of lines being serviced.
0477 0500	1 0	NJP 63	2	If not last servicing, jump back to service next TTU.
0501	Continue			

* Note: The above instructions contain mnemonic function codes in order to indicate the type of instruction being performed. Before program execution, these must be replaced by their equivalent numeric codes.

Example 2
Assuring Transmission Validity

Several techniques have evolved to assure message content validity. One such technique is a form of the Fire code which is described in the following problem. By this method, specific words of the data to be transmitted are added into eight "Check Sum" (S) words. After computing each of the eight sum words, at both origin and destination locations, comparisons of the corresponding sums indicate message validity. This technique provides the advantage of being able to use all bits of a message character as information bits. Thus the presence of a parity bit is not mandatory. However, the presence of a parity bit does not affect or degrade the method.

Assume a block of 240 words of 1 character per word is to be transmitted. This block is preceded by an 8 word header, and followed by 8 Check Sum words. Using a Fire code, the data in the header and information portions are to be checked through comparisons of the accumulated sums in the 8 Check Sum (S) words. The accumulated sums of the sum words are determined by the following algorithm:

$$S_1 = W_i + W_j + W_k + \text{----------} + W_n$$

$$S_2 = W_{i+1} + W_{j+1} + W_{k+1} + \text{----} + W_{n+1}$$

$$S_3 = W_{i+2} + W_{j+2} + W_{k+2} + \text{----} + W_{n+2}$$

$$S_8 = W_{i+7} + W_{j+7} + W_{k+7} \text{------} + W_{n+7}$$

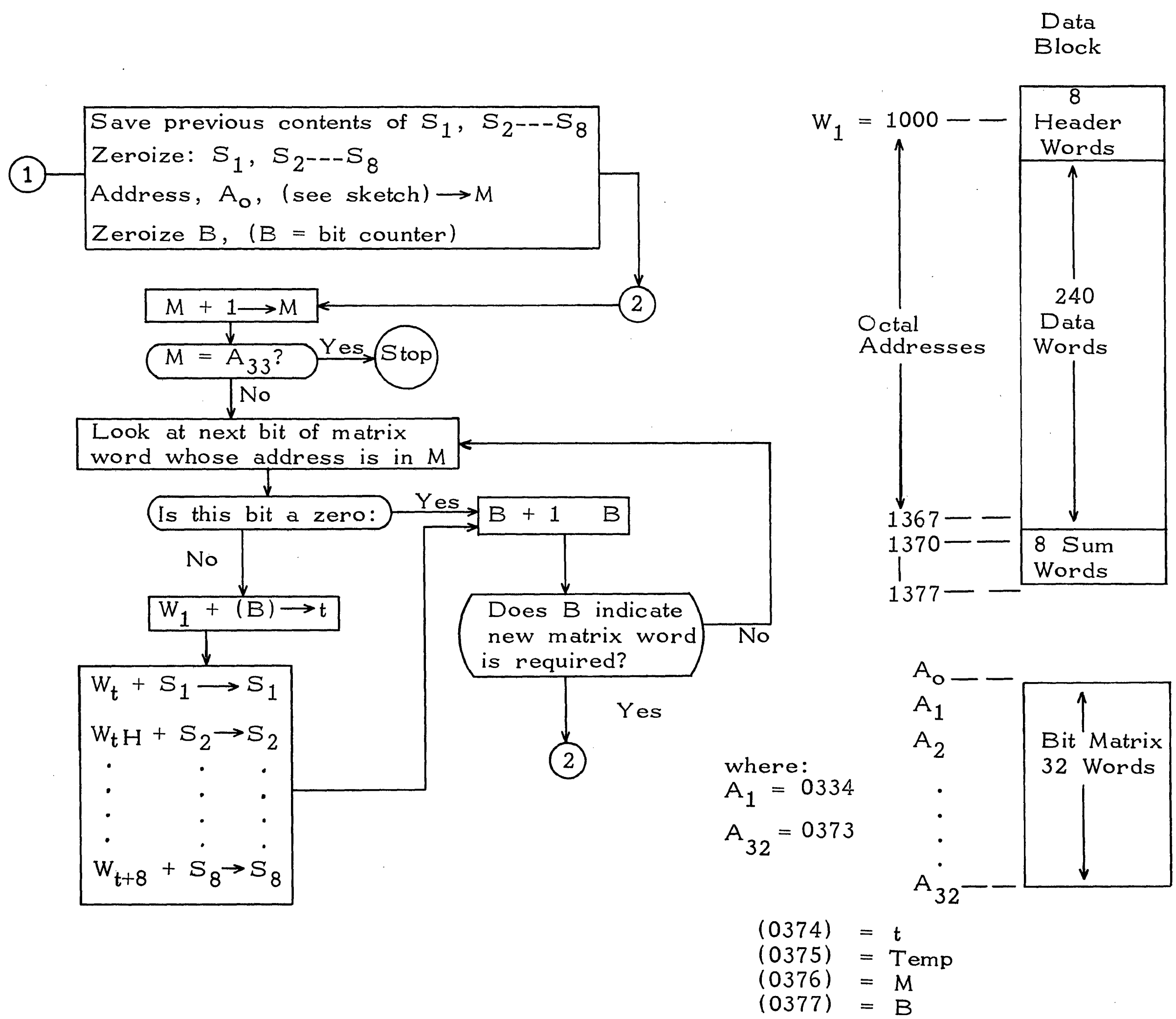
where i,j,k, --- n are the computer addresses containing the data words which are to be transmitted.

W = data words, thus W_j = data word at address j

S_1, S_2, --- S_8 = Check Sum words

In order to implement this method, a matrix of 32 words is used. Each of the 32 words contains 8 bits which indicate the computer addresses of the words which are to be added in the first Check Sum word, S_1. Thus the bit locations within the matrix indicate: i, j, k, ---n addresses of the preceding algorithm.

The technique is indicated by the flow chart and diagram of TeleProgrammer areas below:



THE PROGRAM

Location	T	Code	Cycles	Action Performed
2000	0	LDN	1	Preset Tag register 1 to 0010
2001	0	02		(in bits)
2002	1	ATT	1	
2003	1	LDM	3	
2004	3	70		Save the contents of the check sum
2005	0	STM	3	words by storing initial contents at
2006	3	24		following addresses:
2007	2	RAO	4	$S_1 \longrightarrow 0324$
2010	0	04		
2011	2	RAO	4	$S_2 \longrightarrow 0325$
2012	0	06		
2013	0	SBN	2	
2014	3	34		
2015	2	NZP	2	$S_8 \longrightarrow 0333$
2016	0	03		
2017	0	LDN	2	
2020	0	00		This loop presets each of the eight
2021	1	STM	3	check sum words to zero.
2022	3	70		
2023	2	RAO	4	
2024	0	22		
2025	2	NZP	2	
2026	0	17		
2027	0	STM	3	Set Bit Counter, B, to zero.
2030	3	77		(B is located at octal address, 377.)

THE PROGRAM

Location	T	Code	Cycles	Action Performed
2031	0	LDN	2	Store first address minus 1 of Bit
2032	3	33		Matrix at M; where M is at address
2033	0	STM	3	0376.
2034	3	76		
2035	0	RAO	4	M + 1 $\longrightarrow$ M
2036	3	76		
2037	0	SBN	2	Test address at M, and jump if
2040	3	74		last Matrix Word has been serviced.
2041	2	ZJP	2	
2042	1	12		
2043	0	LDI	4	Load next Matrix Word into A
2044	3	76		
2045	0	SHA	1	Shift Matrix Word left 1
2046	0	STM	3	Store shifted Matrix Word at tempo-
2047	3	75		rary register (at address 0375).
2050	0	LPN	2	Look at what previously had been
2051	0	001		the leftmost bit of Matrix Word.
2052	2	NZP	2	If bit $\neq$ 0, jump; otherwise, con-
2053	0	66		tinue.
2054	0	RAO	4	B + 1 $\longrightarrow$ B (Increase bit count
2055	3	77		by 1.)
2056	0	LPN	2	Look at last three bit positions of
2057	0	07		count at B.

THE PROGRAM

Location	T	Code	Cycles	Action Performed
2060 2061	2 0	ZJP 35	2	If next Matrix Word is required, jump. Otherwise, continue.
2062 2063 2064 2065	0 3 2 0	LDM 75 UJP 45	3 2	Return the current Matrix Word to A and jump back to look at next bit.
2066 2067 2070 2071	0 3 0 3	LDM 77 STM 74	3 3	Bit Count $\rightarrow$ t, where t is at address 0374.
2072 2073 2074 2075	1 3 1 3	LDI 74 RAD 70	4 3	$W_t + S_1 \longrightarrow S_1$
2076 2077 2100 2101	0 3 2 0	RAO 74 RAO 75	4 3	Update parameters in above equation.
2102 2103	2 0	NZP 72	2	Loop back to location, 2072, if not zero.
2104 2105 2106 2107	1 3 1 3	LDI 74 RAD 77	4 4	Store into last Check Sum Word at address 0777.
2110 2111	2 0	UJP 54	2	Loop back for next iteration.
2112	0	HLT	1	Stop

TABLE OF POWERS OF TWO

2^n	n	2^{-n}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5
4 294 967 296	32	0.000 000 000 232 830 643 653 869 628 906 25
8 589 934 592	33	0.000 000 000 116 415 321 826 934 814 453 125
17 179 869 184	34	0.000 000 000 058 207 660 913 467 407 226 562 5
34 359 738 368	35	0.000 000 000 029 103 830 456 733 703 613 281 25
68 719 476 736	36	0.000 000 000 014 551 915 228 366 851 806 640 625
137 438 953 472	37	0.000 000 000 007 275 957 614 183 425 903 320 312 5
274 877 906 944	38	0.000 000 000 003 637 978 807 091 712 951 660 156 25
549 755 813 888	39	0.000 000 000 001 818 989 403 545 856 475 830 078 125

OCTAL-DECIMAL INTEGER CONVERSION TABLE

0000 to 0777 (Octal) — 0000 to 0511 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

1000 to 1777 (Octal) — 0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0612	0613	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

OCTAL-DECIMAL INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2100	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279

	0	1	2	3	4	5	6	7
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1519	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

2000 to 2777 (Octal) — 1024 to 1535 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

3000 to 3777 (Octal) — 1536 to 2047 (Decimal)

OCTAL-DECIMAL INTEGER CONVERSION TABLE

4000 to 4777 (Octal) — 2048 to 2559 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

5000 to 5777 (Octal) — 2560 to 3071 (Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

OCTAL-DECIMAL INTEGER CONVERSION TABLE

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3291	3292	3293	3294	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3308	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

6000 to 6777 (Octal) — 3072 to 3583 (Decimal)

Octal	Decimal
10000 -	4096
20000 -	8192
30000 -	12288
40000 -	16384
50000 -	20480
60000 -	24576
70000 -	28672

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3496	3497	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	4022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

7000 to 7777 (Octal) — 3584 to 4095 (Decimal)

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000	.000000	.100	.125000	.200	.250000	.300	.375000
.001	.001953	.101	.126953	.201	.251953	.301	.376953
.002	.003906	.102	.128906	.202	.253906	.302	.378906
.003	.005859	.103	.130859	.203	.255859	.303	.380859
.004	.007812	.104	.132812	.204	.257812	.304	.382812
.005	.009765	.105	.134765	.205	.259765	.305	.384765
.006	.011718	.106	.136718	.206	.261718	.306	.386718
.007	.013671	.107	.138671	.207	.263671	.307	.388671
.010	.015625	.110	.140625	.210	.265625	.310	.390625
.011	.017578	.111	.142578	.211	.267578	.311	.392578
.012	.019531	.112	.144531	.212	.269531	.312	.394531
.013	.021484	.113	.146484	.213	.271484	.313	.396484
.014	.023437	.114	.148437	.214	.273437	.314	.398437
.015	.025390	.115	.150390	.215	.275390	.315	.400390
.016	.027343	.116	.152343	.216	.277343	.316	.402343
.017	.029296	.117	.154296	.217	.279296	.317	.404296
.020	.031250	.120	.156250	.220	.281250	.320	.406250
.021	.033203	.121	.158203	.221	.283203	.321	.408203
.022	.035156	.122	.160156	.222	.285156	.322	.410156
.023	.037109	.123	.162109	.223	.287109	.323	.412109
.024	.039062	.124	.164062	.224	.289062	.324	.414062
.025	.041015	.125	.166015	.225	.291015	.325	.416015
.026	.042968	.126	.167968	.226	.292968	.326	.417968
.027	.044921	.127	.169921	.227	.294921	.327	.419921
.030	.046875	.130	.171875	.230	.296875	.330	.421875
.031	.048828	.131	.173828	.231	.298828	.331	.423828
.032	.050781	.132	.175781	.232	.300781	.332	.425781
.033	.052734	.133	.177734	.233	.302734	.333	.427734
.034	.054687	.134	.179687	.234	.304687	.334	.429687
.035	.056640	.135	.181640	.235	.306640	.335	.431640
.036	.058593	.136	.183593	.236	.308593	.336	.433593
.037	.060546	.137	.185546	.237	.310546	.337	.435546
.040	.062500	.140	.187500	.240	.312500	.340	.437500
.041	.064453	.141	.189453	.241	.314453	.341	.439453
.042	.066406	.142	.191406	.242	.316406	.342	.441406
.043	.068359	.143	.193359	.243	.318359	.343	.443359
.044	.070312	.144	.195312	.244	.320312	.344	.445312
.045	.072265	.145	.197265	.245	.322265	.345	.447265
.046	.074218	.146	.199218	.246	.324218	.346	.449218
.047	.076171	.147	.201171	.247	.326171	.347	.451171
.050	.078125	.150	.203125	.250	.328125	.350	.453125
.051	.080078	.151	.205078	.251	.330078	.351	.455078
.052	.082031	.152	.207031	.252	.332031	.352	.457031
.053	.083984	.153	.208984	.253	.333984	.353	.458984
.054	.085937	.154	.210937	.254	.335937	.354	.460937
.055	.087890	.155	.212890	.255	.337890	.355	.462890
.056	.089843	.156	.214843	.256	.339843	.356	.464843
.057	.091796	.157	.216796	.257	.341796	.357	.466796
.060	.093750	.160	.218750	.260	.343750	.360	.468750
.061	.095703	.161	.220703	.261	.345703	.361	.470703
.062	.097656	.162	.222656	.262	.347656	.362	.472656
.063	.099609	.163	.224609	.263	.349609	.363	.474609
.064	.101562	.164	.226562	.264	.351562	.364	.476562
.065	.103515	.165	.228515	.265	.353515	.365	.478515
.066	.105468	.166	.230468	.266	.355468	.366	.480468
.067	.107421	.167	.232421	.267	.357421	.367	.482421
.070	.109375	.170	.234375	.270	.359375	.370	.484375
.071	.111328	.171	.236328	.271	.361328	.371	.486328
.072	.113281	.172	.238281	.272	.363281	.372	.488281
.073	.115234	.173	.240234	.273	.365234	.373	.490234
.074	.117187	.174	.242187	.274	.367187	.374	.492187
.075	.119140	.175	.244140	.275	.369140	.375	.494140
.076	.121093	.176	.246093	.276	.371093	.376	.496093
.077	.123046	.177	.248046	.277	.373046	.377	.498046

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000000	.000000	.000100	.000244	.000200	.000488	.000300	.000732
.000001	.000003	.000101	.000247	.000201	.000492	.000301	.000736
.000002	.000007	.000102	.000251	.000202	.000495	.000302	.000740
.000003	.000011	.000103	.000255	.000203	.000499	.000303	.000743
.000004	.000015	.000104	.000259	.000204	.000503	.000304	.000747
.000005	.000019	.000105	.000263	.000205	.000507	.000305	.000751
.000006	.000022	.000106	.000267	.000206	.000511	.000306	.000755
.000007	.000026	.000107	.000270	.000207	.000514	.000307	.000759
.000010	.000030	.000110	.000274	.000210	.000518	.000310	.000762
.000011	.000034	.000111	.000278	.000211	.000522	.000311	.000766
.000012	.000038	.000112	.000282	.000212	.000526	.000312	.000770
.000013	.000041	.000113	.000286	.000213	.000530	.000313	.000774
.000014	.000045	.000114	.000289	.000214	.000534	.000314	.000778
.000015	.000049	.000115	.000293	.000215	.000537	.000315	.000782
.000016	.000053	.000116	.000297	.000216	.000541	.000316	.000785
.000017	.000057	.000117	.000301	.000217	.000545	.000317	.000789
.000020	.000061	.000120	.000305	.000220	.000549	.000320	.000793
.000021	.000064	.000121	.000308	.000221	.000553	.000321	.000797
.000022	.000068	.000122	.000312	.000222	.000556	.000322	.000801
.000023	.000072	.000123	.000316	.000223	.000560	.000323	.000805
.000024	.000076	.000124	.000320	.000224	.000564	.000324	.000808
.000025	.000080	.000125	.000324	.000225	.000568	.000325	.000812
.000026	.000083	.000126	.000328	.000226	.000572	.000326	.000816
.000027	.000087	.000127	.000331	.000227	.000576	.000327	.000820
.000030	.000091	.000130	.000335	.000230	.000579	.000330	.000823
.000031	.000095	.000131	.000339	.000231	.000583	.000331	.000827
.000032	.000099	.000132	.000343	.000232	.000587	.000332	.000831
.000033	.000102	.000133	.000347	.000233	.000591	.000333	.000835
.000034	.000106	.000134	.000350	.000234	.000595	.000334	.000839
.000035	.000110	.000135	.000354	.000235	.000598	.000335	.000843
.000036	.000114	.000136	.000358	.000236	.000602	.000336	.000846
.000037	.000118	.000137	.000362	.000237	.000606	.000337	.000850
.000040	.000122	.000140	.000366	.000240	.000610	.000340	.000854
.000041	.000125	.000141	.000370	.000241	.000614	.000341	.000858
.000042	.000129	.000142	.000373	.000242	.000617	.000342	.000862
.000043	.000133	.000143	.000377	.000243	.000621	.000343	.000865
.000044	.000137	.000144	.000381	.000244	.000625	.000344	.000869
.000045	.000141	.000145	.000385	.000245	.000629	.000345	.000873
.000046	.000144	.000146	.000389	.000246	.000633	.000346	.000877
.000047	.000148	.000147	.000392	.000247	.000637	.000347	.000881
.000050	.000152	.000150	.000396	.000250	.000640	.000350	.000885
.000051	.000156	.000151	.000400	.000251	.000644	.000351	.000888
.000052	.000160	.000152	.000404	.000252	.000648	.000352	.000892
.000053	.000164	.000153	.000408	.000253	.000652	.000353	.000896
.000054	.000167	.000154	.000411	.000254	.000656	.000354	.000900
.000055	.000171	.000155	.000415	.000255	.000659	.000355	.000904
.000056	.000175	.000156	.000419	.000256	.000663	.000356	.000907
.000057	.000179	.000157	.000423	.000257	.000667	.000357	.000911
.000060	.000183	.000160	.000427	.000260	.000671	.000360	.000915
.000061	.000186	.000161	.000431	.000261	.000675	.000361	.000919
.000062	.000190	.000162	.000434	.000262	.000679	.000362	.000923
.000063	.000194	.000163	.000438	.000263	.000682	.000363	.000926
.000064	.000198	.000164	.000442	.000264	.000686	.000364	.000930
.000065	.000202	.000165	.000446	.000265	.000690	.000365	.000934
.000066	.000205	.000166	.000450	.000266	.000694	.000366	.000938
.000067	.000209	.000167	.000453	.000267	.000698	.000367	.000942
.000070	.000213	.000170	.000457	.000270	.000701	.000370	.000946
.000071	.000217	.000171	.000461	.000271	.000705	.000371	.000949
.000072	.000221	.000172	.000465	.000272	.000709	.000372	.000953
.000073	.000225	.000173	.000469	.000273	.000713	.000373	.000957
.000074	.000228	.000174	.000473	.000274	.000717	.000374	.000961
.000075	.000232	.000175	.000476	.000275	.000720	.000375	.000965
.000076	.000236	.000176	.000480	.000276	.000724	.000376	.000968
.000077	.000240	.000177	.000484	.000277	.000728	.000377	.000972

OCTAL-DECIMAL FRACTION CONVERSION TABLE

OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.	OCTAL	DEC.
.000400	.000976	.000500	.001220	.000600	.001464	.000700	.001708
.000401	.000980	.000501	.001224	.000601	.001468	.000701	.001712
.000402	.000984	.000502	.001228	.000602	.001472	.000702	.001716
.000403	.000988	.000503	.001232	.000603	.001476	.000703	.001720
.000404	.000991	.000504	.001235	.000604	.001480	.000704	.001724
.000405	.000995	.000505	.001239	.000605	.001483	.000705	.001728
.000406	.000999	.000506	.001243	.000606	.001487	.000706	.001731
.000407	.001003	.000507	.001247	.000607	.001491	.000707	.001735
.000410	.001007	.000510	.001251	.000610	.001495	.000710	.001739
.000411	.001010	.000511	.001255	.000611	.001499	.000711	.001743
.000412	.001014	.000512	.001258	.000612	.001502	.000712	.001747
.000413	.001018	.000513	.001262	.000613	.001506	.000713	.001750
.000414	.001022	.000514	.001266	.000614	.001510	.000714	.001754
.000415	.001026	.000515	.001270	.000615	.001514	.000715	.001758
.000416	.001029	.000516	.001274	.000616	.001518	.000716	.001762
.000417	.001033	.000517	.001277	.000617	.001522	.000717	.001766
.000420	.001037	.000520	.001281	.000620	.001525	.000720	.001770
.000421	.001041	.000521	.001285	.000621	.001529	.000721	.001773
.000422	.001045	.000522	.001289	.000622	.001533	.000722	.001777
.000423	.001049	.000523	.001293	.000623	.001537	.000723	.001781
.000424	.001052	.000524	.001296	.000624	.001541	.000724	.001785
.000425	.001056	.000525	.001300	.000625	.001544	.000725	.001789
.000426	.001060	.000526	.001304	.000626	.001548	.000726	.001792
.000427	.001064	.000527	.001308	.000627	.001552	.000727	.001796
.000430	.001068	.000530	.001312	.000630	.001556	.000730	.001800
.000431	.001071	.000531	.001316	.000631	.001560	.000731	.001804
.000432	.001075	.000532	.001319	.000632	.001564	.000732	.001808
.000433	.001079	.000533	.001323	.000633	.001567	.000733	.001811
.000434	.001083	.000534	.001327	.000634	.001571	.000734	.001815
.000435	.001087	.000535	.001331	.000635	.001575	.000735	.001819
.000436	.001091	.000536	.001335	.000636	.001579	.000736	.001823
.000437	.001094	.000537	.001338	.000637	.001583	.000737	.001827
.000440	.001098	.000540	.001342	.000640	.001586	.000740	.001831
.000441	.001102	.000541	.001346	.000641	.001590	.000741	.001834
.000442	.001106	.000542	.001350	.000642	.001594	.000742	.001838
.000443	.001110	.000543	.001354	.000643	.001598	.000743	.001842
.000444	.001113	.000544	.001358	.000644	.001602	.000744	.001846
.000445	.001117	.000545	.001361	.000645	.001605	.000745	.001850
.000446	.001121	.000546	.001365	.000646	.001609	.000746	.001853
.000447	.001125	.000547	.001369	.000647	.001613	.000747	.001857
.000450	.001129	.000550	.001373	.000650	.001617	.000750	.001861
.000451	.001132	.000551	.001377	.000651	.001621	.000751	.001865
.000452	.001136	.000552	.001380	.000652	.001625	.000752	.001869
.000453	.001140	.000553	.001384	.000653	.001628	.000753	.001873
.000454	.001144	.000554	.001388	.000654	.001632	.000754	.001876
.000455	.001148	.000555	.001392	.000655	.001636	.000755	.001880
.000456	.001152	.000556	.001396	.000656	.001640	.000756	.001884
.000457	.001155	.000557	.001399	.000657	.001644	.000757	.001888
.000460	.001159	.000560	.001403	.000660	.001647	.000760	.001892
.000461	.001163	.000561	.001407	.000661	.001651	.000761	.001895
.000462	.001167	.000562	.001411	.000662	.001655	.000762	.001899
.000463	.001171	.000563	.001415	.000663	.001659	.000763	.001903
.000464	.001174	.000564	.001419	.000664	.001663	.000764	.001907
.000465	.001178	.000565	.001422	.000665	.001667	.000765	.001911
.000466	.001182	.000566	.001426	.000666	.001670	.000766	.001914
.000467	.001186	.000567	.001430	.000667	.001674	.000767	.001918
.000470	.001190	.000570	.001434	.000670	.001678	.000770	.001922
.000471	.001194	.000571	.001438	.000671	.001682	.000771	.001926
.000472	.001197	.000572	.001441	.000672	.001686	.000772	.001930
.000473	.001201	.000573	.001445	.000673	.001689	.000773	.001934
.000474	.001205	.000574	.001449	.000674	.001693	.000774	.001937
.000475	.001209	.000575	.001453	.000675	.001697	.000775	.001941
.000476	.001213	.000576	.001457	.000676	.001701	.000776	.001945
.000477	.001216	.000577	.001461	.000677	.001705	.000777	.001949

GLOSSARY

GLOSSARY OF TeleProgramming TERMS

The following glossary gives the meaning of terms that are used in a relatively specialized sense in this manual.

ADDER — In general, a device used to add two quantities. Specifically, the borrow structure in the subject computer.

ADDRESS — The number designating a storage location; also the storage location itself.

- **NO ADDRESS MODE** — The TeleProgrammer permits the performance of arithmetic and logical operations by an 8-bit constant associated with the instruction and using the memory location immediately following the instruction as an 8-bit operand.

- **MEMORY ADDRESS MODE** — A mode of addressing wherein an 8-bit operand in any storage location is addressed by the memory location (immediately following the instruction) and the contents of the Tag register as referenced by T.

- **INDIRECT ADDRESS MODE** — Instructions employing indirect addressing use the memory location immediately following the instruction to refer to one of the first 256 storage locations. The contents of this location are used along with the contents of the Tag register to form the address of the operand.

BIT — Binary digit; may be either "1" or "0".

BORROW — In a subtractive counter or accumulator, a signal indicating that in stage n, a "1" was subtracted from a "0".

BUFFER — Noun: A device in which data are stored temporarily in the course of transmission from one point to another. Verb: To store data temporarily.

BUFFERED INPUT/OUTPUT	A term indicating that the computer may carry on high speed computation at the same time it is exchanging data with a peripheral device. In the TeleProgrammer, this term must be distinguished from normal I/O, during which the computer cannot engage in computation.
CARRY	In an additive counter or accumulator, a signal indicating that in stage n, a "1" was added to a "1".
CHANNEL	A transmission path that connects the computer to a given external equipment.
CLEAR	A command that removes a quantity from a register by placing every stage in the "0" state.
COMMAND	A signal that performs a unit operation, such as transmitting contents of one register to another, shifting a register, setting a flip-flop.
COMPLEMENT	Noun: See One's Complement to Two's Complement. Verb: A command which produces the one's complement of a given quantity.
CONTENT	The quantity or word held in a register or storage location.
CORE	A small ferromagnetic toroid used as the bistable device for storing a bit in a memory plane.
COUNTER	A register with provisions for increasing or decreasing its content by 1 upon receiving the appropriate command.
END-AROUND BORROW	A borrow that is generated in the highest order of an accumulator or counter, and is sent directly to the lowest order stage.
ENTER	To manually place in a register a quantity that is not from storage. In the TeleProgrammer, quantities may be entered in only the Tag A, P, and Z registers.
FUNCTION CODE	The lower 2 octal digits of the first word in the instruction set.

INPUT DISCONNECT	During an input instruction, a signal sent to the computer by the external device to indicate that the device has completed all available transmissions to the computer.
INPUT REQUEST	A request, by the computer, for information from an external device. Occurs during input instruction only. (See Resume.)
INTERRUPT	A signal (or class thereof) which, when received and recognized by the computer, forces the computer to forestall its current operation and jump to a subroutine, the starting address of which is determined by the class of the interrupt. A subroutine may have any number of options. It may merely stop the computer, it may determine the nature of the interrupt in order to take corrective measures, or it may return the computer to another phase of the main program.
JUMP	An instruction that jumps from one sequence of instructions to a second, and makes no preparation for returning to the first sequence.
LOAD	To place a quantity from storage in the A register.
LOCKOUT	Any function (usually of machine logic) that inhibits an action which would normally occur were the lockout not imposed.
LOGICAL PRODUCT	In Boolean algebra, the AND function of several terms. The product is "1" only when all the terms are "1"; otherwise it is "0". Sometimes referred to as the result of "bit-by-bit" multiplication.
LOGICAL SUM	In Boolean algebra, the OR function of several terms. The sum is "1" when any or all of the terms are "1"; it is "0" only when all are "0".
MASK	In the information of the logical products of two quantities, one of them may be used as a mask for the other. The mask determines what part of the other quantity is to be considered. Wherever the mask is "0", that part of the other quantity is cleared, but wherever the mask is a "1", the other quantity is left unaltered.

MASTER CLEAR (MC)	A general command produced by placing the Load/Clear switch in the down (CLEAR) position. An MC clears all of the crucial registers and control FFs to prepare for a new mode of operation.
MODULUS	An integer which describes certain arithmetic characteristics of registers, especially counters and accumulators, within a digital computer. The modulus of a device is defined by r^n for an open ended device and r^n-1 for a closed (end-around) device, where r is the base of the number system used and n is the number of digit positions (stages) in the device. Generally, devices with modulus r^n use two's complement arithmetic procedures, and devices with modulus r^n-1 use one's complement procedures.
ONE'S COMPLEMENT	With reference to a binary number, that number which results from subtracting each bit of the given number from the bit "1". A negative number is expressed by the one's complement of the corresponding positive number.
OPERAND	The quantity specified by the 8 bits of the second word of the instruction set. This quantity is operated upon in the execution of the instruction.
OPERATION CODE	The lower 2 octal digits of the first word in the instruction set also called Function Code and identified by the letter, F. After the code is translated, it conditions the computer for execution of the specified instruction.
OVERFLOW	The condition in which the capacity of a register is exceeded.
PARTIAL ADD	An addition without carries. Accomplished by toggling each bit of the augend where the corresponding bit of addend is a "1".
PROGRAM	A precise sequence of instructions that accomplishes a computer routine; a plan for the solution of a problem.

QUARTIC	A number system with a base of four. These numbers are normally partitioned into groups of two for ease of reading.
READ	To place a quantity from a storage location into a register. The quantity in storage remains unchanged.
READY	The input/output control signal sent by either the computer or an external equipment to alert the device that is to receive a transmission. The ready signal indicates that the word or character has been transmitted.
RELATIVE ADDRESSING	A mode of addressing wherein the address of the operand is determined by adding (or subtracting) the contents of the execution address portion of the instruction word to (or from) the instruction address.
REPLACE	In the title of an instruction, the result of the execution of the instruction is stored in the location from which the initial operand was obtained.
RESUME	The output control signal sent by an external equipment to indicate that it is prepared to receive another word or character. The resume signal is thus a request for data. (See Input Request.)
ROUTINE	The sequence of operations which the computer performs under the direction of a program.
SHIFT	To move the bits of a quantity right or left.
SIGN BIT	The bit in the highest-order stage of the register (in registers where a quantity is treated as signed by use of one's complement notation). If the bit is "1", the quantity is negative; if the bit is "0", the quantity is positive.
SIGN EXTENSION	The duplication of the sign bit in the highest-order stages of a register.

STATUS — 1) The condition of an external device, as reflected in the response given to a status request interrogation by the computer.
2) The condition of the computer as shown by the Status Mode indicator on the console. May variously indicate what it is presently doing, why it stopped, or what it will do when it next starts.

TRANSMISSION FORCED — A transmission where both set and clear inputs, only one of which will be a "1", are simultaneously gated into a FF which has not been cleared previously.

TRANSLATION — An indication of the content of a group of bit registers. A complete translation gives the exact content, while a partial translation indicates only that the content is within certain limits.

TWO's COMPLEMENT — That number which results from subtracting each bit of a number from "0". The two's complement may be formed by complementing each bit of the given number and then adding one to the result, performing the required carries.

WORD — A unit of information which has been coded for use in the computer as a series of bits. The normal word length is 8 bits.

WRITE — To enter a quantity into a storage location.

COMMENT SHEET

MANUAL TITLE 8092 TeleProgrammer

Programming Reference Manual

PUBLICATION NO. 36810701 REVISION

FROM: NAME:

BUSINESS ADDRESS:

COMMENTS:

This form is not intended to be used as an order blank. Your evaluation of this manual will be welcomed by Control Data Corporation. Any errors, suggested additions or deletions, or general comments may be made below. Please include page number references and fill in publication revision level as shown by the last entry on the Record of Revision page at the front of the manual. Customer engineers are urged to use the TAR.

CUT ALONG LINE

AA3419 REV. 11/69 PRINTED IN U.S.A.

NO POSTAGE STAMP NECESSARY IF MAILED IN U. S. A.

FOLD ON DOTTED LINES AND STAPLE

STAPLE

STAPLE

FOLD

FOLD

FIRST CLASS
PERMIT NO. 8241

MINNEAPOLIS, MINN.

BUSINESS REPLY MAIL
NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

POSTAGE WILL BE PAID BY

CONTROL DATA CORPORATION
Technical Publications
4201 N. Lexington Ave.
St. Paul, Minnesota 55112

CUT ALONG LINE

FOLD

FOLD

CONTROL DATA

1/2
3/4
1
1-1/4

>>> CUT OUT FOR USE AS LOOSE-LEAF BINDER TITLE TAB

CONTROL DATA
CORPORATION
8100 34th AVE. SO., MINNEAPOLIS, MINN. 55440

LITHO IN U.S.A.