SUPPLEMENT
to the
INSTRUCTION BOOK
for

NAVY MODEL CXMX EQUIPMENT

A Guide to the Basic Arithmetic
and
Arithmetic Circuits

ENGINEERING RESEARCH ASSOCIATES
DIVISION OF REMINGTON RAND, INC.
1902 WEST MINNEHAHA AVENUE
ST. PAUL W4, MINNESOTA

BUREAU OF SHIPS NAVY DEPARTMENT

Contracts: NObsr 42001 & 63010
Task: 13 *Approved by BuShips: 13 May 1953*

LIST OF EFFECTIVE PAGES

Title Page
A thru B
i thru iii
1 thru 58
1 thru 27 (Appendix 1)
1 thru 11 (Appendix 2)

RECORD OF CORRECTIONS MADE

CHANGE NO.	DATE	SIGNATURE OF OFFICER MAKING CORRECTION

TABLE OF CONTENTS

P[illegible] 1177

LIST OF ILLUSTRATIONS

FOREWORD

The purpose of this guide is to provide both a summary of binary arithmetic fundamentals and sufficient information to understand the arithmetic functioning of the Navy Model CXMX Equipment.

To provide this information the guide is divided as follows:

SECTION I - FUNDAMENTALS

SECTION II - RULES OF COMPUTATION FOR THE NAVY MODEL CXMX EQUIPMENT

SECTION III - ARITHMETIC AS USED IN THE NAVY MODEL CXMX EQUIPMENT

SECTION IV - ARITHMETIC CIRCUITRY OF THE NAVY MODEL CXMX EQUIPMENT
BIBLIOGRAPHY
APPENDIXED MATERIAL

Section I deals with the fundamentals of binary arithmetic. Section II explains in detail the methods used in the CXMX Equipment, while Section III is a discussion of logical steps taken within the CXMX Equipment to perform the various arithmetic operations. Section IV discusses the basic arithmetic circuits and their inherent properties. The Appendix is composed of papers which rigorously consider the arithmetic proofs. Examination of these papers is left to the discretion of the reader since an understanding of their content is not a prerequisite to an understanding of the arithmetic operations of the CXMX Equipment. The bibliography is a listing of current literature containing further details concerning the material discussed in this guide.

I. FUNDAMENTALS

A. NUMBER SYSTEMS - A number system may be defined by two principal characteristics: its radix and its modulus.

(1) Radix - The radix, or base, of a system is a number equal to the number of unique symbols used to represent all the discrete numerical values or quantities encompassed by the system. For example, a system with Radix Ten, encompassing an infinite number of values or quantities may use no more than ten unique symbols to represent any particular value.

(2) Modulus - The modulus of a system is the number of discrete quantities which the system can distinguish uniquely. An example of system modulus is found in the twelve-hour system (Modulus Twelve) of telling time. Seven o'clock is indicated by the number 7 whether it be morning or evening.

Several number systems, more specifically, systems of numerical notation, are commonly used in digital computers. All of these have the common function of identifying quantities or magnitudes. Although in different systems the methods of notation vary widely, the values expressed are identical. It is important to understand a few of these systems and their interrelationships in order to understand the functioning of digital computers.

(1) Decimal System - The system with Radix Ten mentioned above is the commonly used decimal system. It contains ten unique characters, 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9, representing, respectively, ten successive quantities. The use of decimal notation has obvious advantages: its symbols and arithmetic processes are clearly understood, and conversion, either mental or physical is unnecessary for utilizing data produced by a decimal computer.

(2) Binary System - The binary system of numerical representation has a radix of two; that is, the system uses only two unique characters, commonly 0 and 1, to represent quantities or magnitudes. For quantities greater than one, the two binary characters are repeated in a manner to be described in Paragraph B. This system of recording numerical values has particular advantages for use in digital computers since the digits may be conveniently represented electrically and physically by two states; i.e., "on" and "off", "signal" and "no signal", etc.

(3) Octal System - The octal system, Radix Eight, contains eight unique elements which are usually represented by the Arabic numerals 0 through 7. It is extremely convenient to represent large binary numbers in octal notation, and examples of this conversion are given in Section C below.

B. REPRESENTATION OF VALUES IN A NUMBER SYSTEM - Any quantity, Q , may be expressed in completely general terms as follows:

$$Q = A_n r^n + A_{n-1} r^{n-1} + A_{n-2} r^{n-2} + \ldots + A_2 r^2 + A_1 r^1 + A_0 r^0 \quad (1)$$

where $A_n, A_{n-1}, \ldots, A_1, A_0$ is a set of coefficients whose largest value is one less than the radix of the system,

and $r^n, r^{n-1}, \ldots, r^1, r^0$ is a set of decreasing powers of the radix of the system, limited in magnitude only by the number of terms required to express Q.

If the radix of the numbering system has been established, the quantity Q may be expressed in "shorthand" notation by listing only the coefficients of the various powers of the radix in descending order of the powers as follows:

$$A_n\ A_{n-1}\ A_{n-2} \ldots A_2\ A_1\ A_0$$

(1) Representation of Values in Decimal Notation - Values are expressed in decimal (Radix Ten) notation in accordance with the following rule:

$$Q = A_n \times 10^n + A_{n-1} \times 10^{n-1} + \ldots + A_1 \times 10^1 + A_0 \times 10^0 \quad (2)$$

Thus, the quantity one hundred thirty-seven is indicated by the decimal digits 137, and in terms of the general equation:

$$Q = 137 = 1 \times 10^2 + 3 \times 10^1 + 7 \times 10^0 \quad (3)$$

(2) Representation of Values in Binary Notation - Values are expressed in binary (Radix Two) notation in accordance with the following similar rule:

$$Q = A_n \times 2^n + A_{n-1} \times 2^{n-1} + \ldots + A_1 \times 2^1 + A_0 \times 2^0 \quad (4)$$

Thus, the decimal number 137 is represented in binary notation as

$$Q = \underset{\text{DECIMAL}}{137} = \underset{\text{BINARY}}{1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0} \quad (5)$$

With the radix two understood, the binary representation becomes simply

DECIMAL		BINARY
137	=	10001001

(6)

The method used to determine the coefficients for this binary number will be shown in the next paragraph. At present, only the numerical equivalence of the two notations will be shown. This equivalence is seen by converting all terms of Equation (5) to their decimal equivalent and summing:

$$137 = 128 + 0 + 0 + 0 + 8 + 0 + 0 + 1 \qquad (7)$$

(3) Representation of Values in Ternary Notation - Similar use of the general equation yields the following ternary expression of the same decimal number ($_{137}$).

	DECIMAL		TERNARY	
Q	$= 137$	$=$	$1 \times 3^4 + 2 \times 3^3 + 0 \times 3^2 + 0 \times 3^1 + 2 \times 3^0$	(8)

DECIMAL		TERNARY	
137	$=$	12002	(9)

The numerical equivalence is shown converting all terms of Equation (8) to their decimal equivalent and summing:

$$137 = 81 + 54 + 0 + 0 + 2 + 0$$

C. CONVERSION FROM SYSTEM TO SYSTEM - In the general case, conversion from one system (radix p) to another system (radix r) is performed as follows:

(1) Determine the highest power n to which the radix r may be raised and yield a positive remainder when (Q_p) is divided by (r^n).

(2) Subtract r^n as many times as possible and still leave a positive remainder less than r^n. The integer representing the number of times that r^n was subtracted is the coefficient of r^n.

(3) Determine the next highest power, k, to which r may be raised and have r^k less than or equal to the remainder from step (2). The integer which represents the number of times r^k is subtracted is the coefficient of r^k. This procedure is repeated until r^0 has been subtracted leaving a zero remainder.

(4) The coefficients of all powers not used are zero, and must be represented in their proper order in the final result.

An example using the above rules is shown below using the general form of the equation for number representation.

$$Q_p = A_n r^n + A_{n-1} r^{n-1} + \ldots + A_0 r^0$$

n is determined by division as indicated in rule (1) above. Then r^n is subtracted as many times as possible (A_n times in this example).

PX 71177

$$Q_p - A_n r^n = Q_{p-1}$$

From the remainder, Q_{p-1}, of the preceding step, r^{n-1} is subtracted A_{n-1} times if r^{n-1} is less than or equal to Q_{p-1}.

$$Q_{p-1} - A_{n-1} r^{n-1} = Q_{p-2}$$

This process is repeated until the lowest order term is reached, and

$$Q_0 - A_0 r^0 = R$$

Note that the remainder must, for the lowest order term be 0. Since r^0 equals 1, A_0 must equal Q_0.

The coefficients ($A_n, A_{n-1}, \ldots A_0$) of the terms of the radix (r) are the representation of the quantity (Q_p) in the new system, as shown below.

RADIX p		RADIX r	
Q_p	=	$A_n A_{n-1} \ldots A_0$	(11)

The use of these rules is illustrated in the following numerical example.

Express decimal 137 as an octal number.

8^2 (or 64) is the greatest power of 8 that divides into 137, so the expression will be in the form

DECIMAL		OCTAL
$Q = 137$	=	$A_2 8^2 + A_1 8^1 + A_0 8^0$

Subtract 64 from 137 as many times as possible and still leave a positive remainder less than 64.

$137 - 64 = 73$ the remainder is still greater than 64
$73 - 64 = 9$ the remainder is less than 64

64 was subtracted twice, so the first octal digit (A_2) is 2. The remainder 9 is greater than 8^1 so 8 is subtracted until a positive remainder less than 8 is left.

$9 - 8 = 1$ the remainder is less than 8

8 was subtracted once, so the second octal digit (A_1) is 1. The remainder 1 is equal to 8^0 so 1 is subtracted.

$1 - 1 = 0$ the remainder is 0

1 was subtracted once, so the third octal digit (A_0) is 1. Therefore, upon collecting the octal coefficients,

DECIMAL		OCTAL		OCTAL
137	=	$A_2A_1A_0$	=	211

Conversion from one non-decimal system to another non-decimal system can often best be performed by using the more familiar decimal system as an intermediate step. For example, translation of the binary number 10001001 into the ternary system may be performed as follows:

BINARY		DECIMAL		DECIMAL
10001001	=	128 + 8 + 1	=	137

DECIMAL		DECIMAL
137	=	81 + 54 + 2

TERNARY

$$= 1 \times 3^4 + 2 \times 3^3 + 0 \times 3^2 + 0 \times 3^1 + 2 \times 3^0$$

TERNARY

$$= 12002$$

Binary-to-Octal Conversion - The conversion process from binary-to-octal and octal-to-binary digits can be shown as follows:

BINARY

$$Q = 10001001 \tag{12}$$

Expanded, (12) becomes

$$Q = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \tag{13}$$

but

$$2^6 = 8^2$$
$$2^3 = 8^1$$
$$2^0 = 8^0 = 1$$

and from (13), converting the non-zero digits,

$$1 \times 2^0 = 1 \times 8^0 \tag{14}$$
$$1 \times 2^3 = 1 \times 8^1 \tag{15}$$
$$1 \times 2^7 = 2 \times 2^6 = 2 \times 8^2 \tag{16}$$

PX 71177

~~PX 20961~~

The expressions to the right of the equality signs in (14), (15), and (16) are the ascending powers of the octal expansion derived from the binary example, so

$$\underline{\text{OCTAL}}$$
$$Q = 2 \times 8^2 + 1 \times 8^1 + 1 \times 8^0 \qquad (17)$$

and with the radix (8) understood,

$$\underline{\text{OCTAL}}$$
$$Q = 211 \qquad (18)$$

Octal-to-Binary Conversion

$$\underline{\text{OCTAL}}$$
$$Q = 211 \qquad (19)$$

Expanded (19) becomes

$$Q = 2 \times 8^2 + 1 \times 8^1 + 1 \times 8^0 \qquad (20)$$

but

$$1 \times 8^0 = 1 \times 2^0$$
$$1 \times 8^1 = 1 \times 2^3$$
$$2 \times 8^2 = 2 \times 2^6 = 1 \times 2^7$$

The coefficients of all the intervening binary powers are zero, so,

$$\underline{\text{BINARY}}$$
$$Q = 1 \times 2^7 + 0 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 \qquad (21)$$

and with the radix (2) understood, (21) becomes

$$\underline{\text{BINARY}}$$
$$Q = 10001001 \qquad (22)$$

Combining Equations (18) and (22)

$$Q = \underset{\downarrow}{10}\ \underset{\downarrow}{001}\ \underset{\downarrow}{001} \quad \text{BINARY} \qquad (23)$$
$$Q = 2\ 1\ 1 \quad \text{OCTAL} \qquad (24)$$

Equations (23) and (24) show that three ascending, consecutive binary digits can be represented by a single octal digit, and, conversely, an octal digit can be represented by three consecutive binary digits.

PX 71177

PX 29861

D. LIMITATIONS ON SIZE OF NUMBERS - In manual computation, operations may be performed on a quantity, regardless of its size, provided there is sufficient time and paper. This fact is also true, at least in theory, of computations performed by automatic calculators. In practice, however, the size of the number which a computer can handle at any one time is limited by the modulus of the system.

Consider the familiar desk calculator having 10 decimal digit capacity. The machine can express 10^{10} different numbers. It cannot distinguish 10^{10} from the number 0 or distinguish between two numbers which differ by integral multiples of 10^{10}. A simpler example of this limitation is found in the odometer used to record mileage in an automobile. Since this instrument generally operates with modulus 10^5, it cannot distinguish between two numbers which differ by integral multiples of 100,000 miles.

It is important to understand clearly the distinction between the symbolic representation of a number and the number itself. In the example of the desk calculator cited above, 10^{10} different representations of numbers are possible. These could represent the numbers from 0 to $10^{10}-1$ inclusive or they could be any other set of 10^{10} values. If it is desired to represent both positive and negative numbers in the machine, then these 10^{10} different representations could be used to represent 5×10^9 positive numbers and 5×10^9 negative numbers. The range of numbers covered would then be -5,000,000,000 through +4,999,999,999. In this case 0 is considered a positive number.

E. REPRESENTATION OF NEGATIVE VALUES

(1) IN DECIMAL NOTATION - In manual computations, negative numbers are indicated by the absolute value preceded by a minus sign. This method of negative number indication is, in general, not suitable for calculators. Consider the case of the desk calculator handling a series of subtractions. In performing subtraction, the desk calculator uses a reversing gear and runs the register wheels backwards. Assume that a positive number is initially entered into the machine, and that the subtraction process is continued enough times for the value in the register to go through zero and become negative. It will then be noted that the numbers shown by the register wheels continue to decrease with each subtraction, whereas in the usual manual notation the numbers would be preceded by a minus sign and begin to increase. The solution to this apparent discrepancy depends upon recognition of the fact that the machine is of limited capacity and, as pointed out above, can express only 10^{10} different numbers. The range of indication is 0 to $10^{10}-1$. Any quantity outside this range of indication is represented in the machine by the quantity plus or minus an integral multiple of 10^{10}. The required integral multiple is that necessary to make the indicated number lie between the aforementioned limits of 0 and $10^{10}-1$. For example, minus 137 is represented as $-137 + 10^{10}$ or 9999999863.

The number 9999999863, obtained by subtracting 137 from 10^{10}, is called the ten's complement of 137 with respect to 10^{10}. Note that the process used is equivalent to subtracting the rightmost non-zero digit from 10 and all other digits from 9, or that all digits are subtracted from 9 and then a 1 is added.

Two examples of the ten's complement with respect to 10^8 obtained by subtraction from and addition of 1 are shown on the following page.

99999999		99999999	
00000137	SUBTRACTION	00001370	SUBTRACTION
99999862		99998629	
1	ADD	1	ADD
99999863	COMPLEMENT	99998630	COMPLEMENT

The method of negative number representation described above, used in desk calculators, is not directly applicable to large scale automatic calculators. In the first place, most high speed devices cannot be made to run backwards conveniently or economically. In the second place, a complement system (system of representing negative values) which requires special treatment of the rightmost non-zero digit or an addition process for each complement formation is undesirable.

To avoid the first mentioned difficulty (the necessity for operating the computing mechanisms backward) subtractions may be performed by the simple rule of elementary algebra: change the sign of the subtrahend (that is, complement) and add.

To avoid the second difficulty, the nine's complement system may be used. The nine's complement of a decimal number is obtained simply by subtracting each digit from 9 (treating all digits alike).

Considering that the basic computing or counting mechanism is decimal in nature, each position of the mechanism has ten stable states. Upon reaching the tenth state and the addition of one more unit, the original counter reverts to its zero state and a carry is generated which is transmitted by the mechanism to the next position. The principles outlined above are the same for negative numbers except that a different method of notation is used.

The usual method of notation uses symbols which progress consecutively from 0 ...00 to 9 ...99 . The addition of 1 more to this progression results in 0 ...00, which is the original state. Figure 1, an example of this notation, is a 100 position counting wheel. Each count advances it successively one step to maximum of 99. One more count returns the counter to 00.

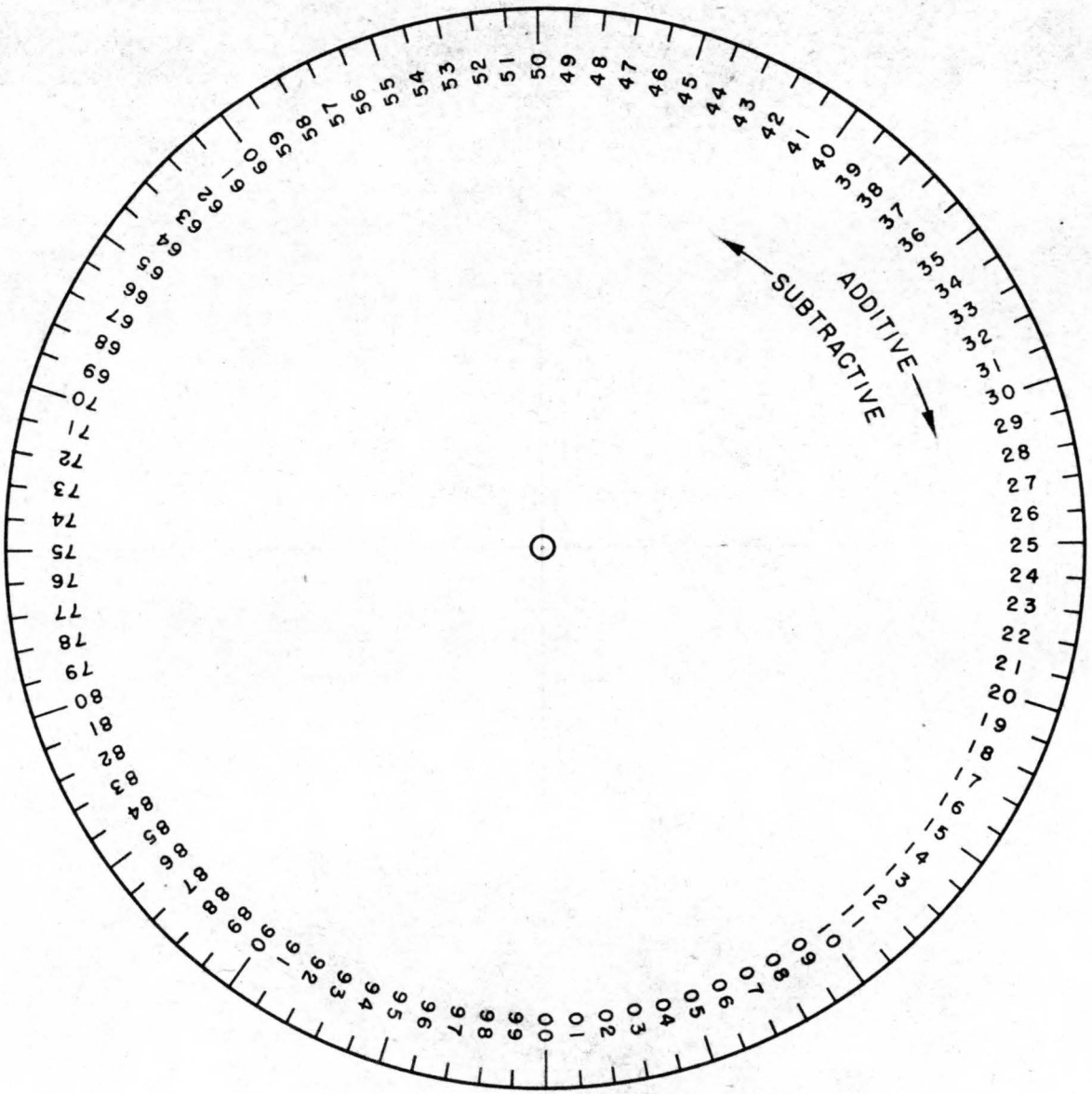
Such a wheel is useful in representing a ten's complement system of negative numbers as described in previous paragraphs because it shows the result of the subtraction directly with no need for correction. As an example, the complement of 1 can be found by setting the wheel 00 on the arrow and turning the wheel subtractively 1 place, the complement appearing as 99.

In the nine's complement system, the complement of 1 is 98. Adding 1 to this number yields 99, which number, therefore, is a representation of zero. Subtracting 1 from 01 yields 00. Thus zero has two symbols in this system.

Because zero is represented by two unique numbers, the modulus of the system is reduced from 10^n to $10^n - 1$.

Consider again a 100 position wheel (Figure 1). It is desired to operate this wheel in the nine's complement system (as though it had 99 positions). The numbers 99 and 00 therefore both represent 0. Adding 1 to this system moves the wheel to the next position in a clockwise direction. Subtracting 1 from the

~~PX 29861~~ PX 71177



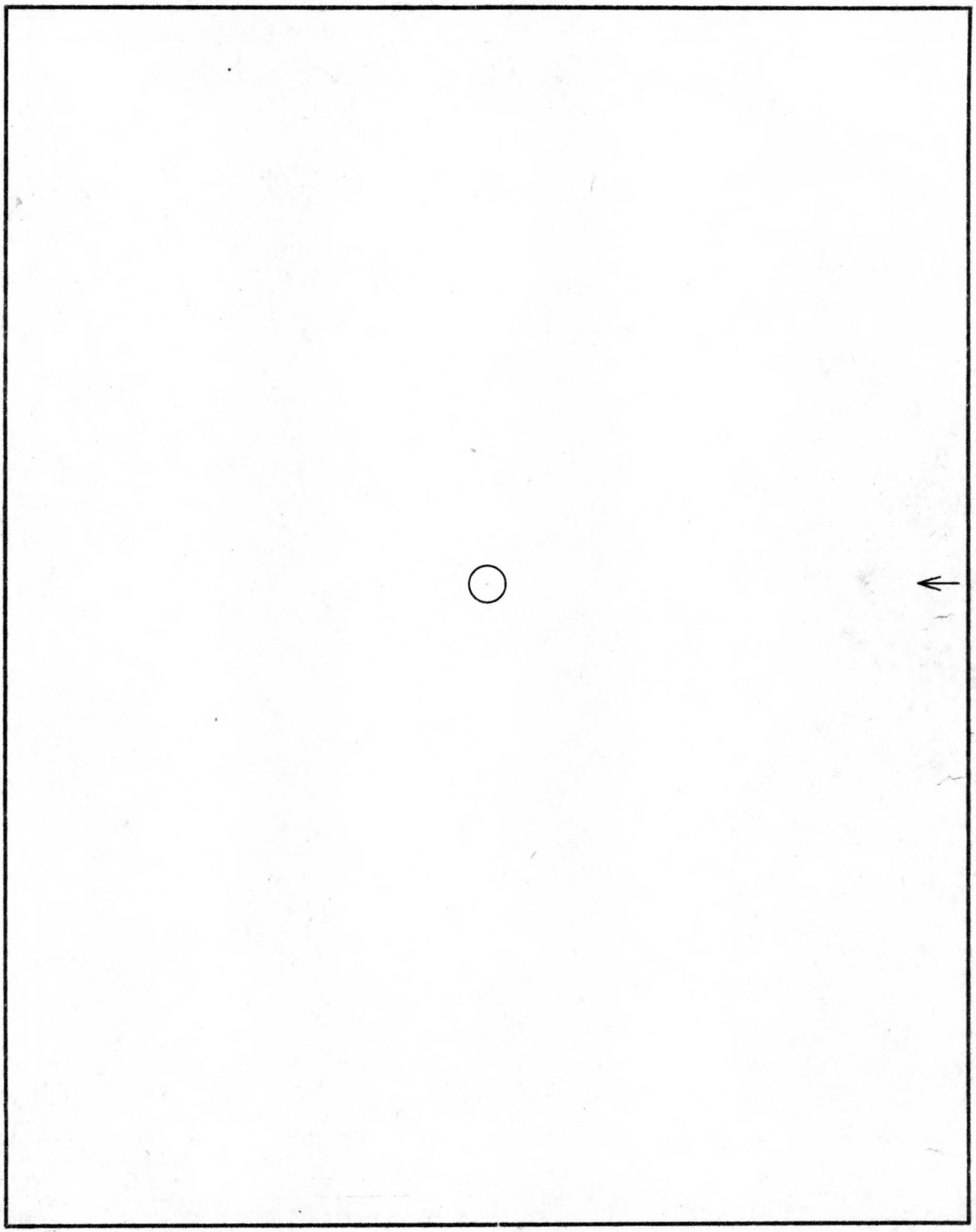


FIGURE I COUNTING WHEEL

system moves the wheel to the next position in counterclockwise direction. Adding 1 to 98 should move the wheel to zero and adding 1 to zero should move the wheel to 01 This may be accomplished if in the transition from 98 to zero the wheel is moved an extra position. If the wheel is considered additive (clockwise motion) the progression of numbers increases from the lower numbers to 99 (zero) at which point an extra step is required to move the wheel to the 01 position. In other words, in an additive nine's complement device zero is represented by 99. In a subtractive nine's complement device (counterclockwise motion) the numbers proceed from the higher order numbers to 00 with the subtraction of 1 transferring the wheel from 0 to 98 . Therefore, in such a subtractive device, zero is represented by 00 . The modulus of the system is reduced from 100 to 99 by use of this extra corrective step. The correction supplied at the transition point is known as end around carry for the additive wheel and end around borrow for the subtractive wheel.

End-around carry means that whenever the number in the system passes through zero in the additive sense a quantity 1 is automatically added to the right hand digit of the number to correct for the transition point. This quantity 1 is the digit which would ordinarily appear in the next left column if another column were available.

End-around-borrow means that whenever the number in the system passes through zero in the subtractive sense, a quantity 1 is automatically subtracted from the right hand digit of the number to correct for the transition point. This quantity 1 is the digit which would ordinarily be borrowed from the next left column if another column were available. Because of the absence of another left column, the quantity 1 is borrowed from the right hand column.

The two zero possibilities are not really a disadvantage because one possibility is always automatically excluded by the system of arithmetic followed by the machine. Computers may be designed basically additive or basically subtractive. In a machine which adds, the number 000.... 0 never results from an arithmetic operation and in one which subtracts, the number 999..... 9 never results from an arithmetic operation.

(2) IN BINARY NOTATION - The remarks dealing with negative number representation in machines using decimal notation may be applied to negative number representation in machines using binary notation.

In binary notation, there are two systems of negative representation which are similar to the ten's and nine's complement systems. These are the two's complement, which is analogous to the ten's complement, and the one's complement, which is analogous to the nine's complement, systems.

Consider Figure 2 which is a tabulation of the sixteen decimal values from 0 through 15 and the corresponding binary representations.

Note that a total of 2^4 or 16 , values can be expressed and the decimal range of indication is from 0 through 15 . The modulus of this system is 2^4 and the binary numbers listed may be considered a closed loop with 0000 following 1111.

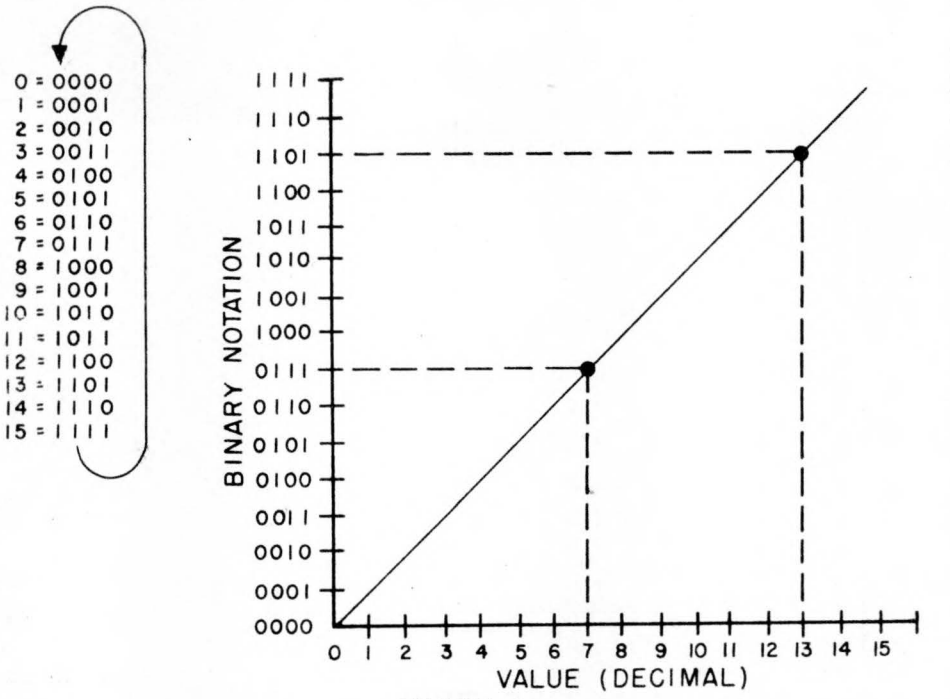


FIGURE 2
DECIMAL VS. BINARY POSITIVE VALUES

In the two's complement system, negative values are formed in a manner similar to that used in forming ten's complements. Figure 3 shows a system having a modulus of sixteen as used in Figure 2, but having an equal number of positive and negative values. Zero is considered to be a positive value. The negative values are formed by subtraction of the absolute value from 2^4. The values -8 through +7 are now represented by the same 16 binary values used in Figure 2. For example:

$$-5 = 2^4 - |5|$$

$$= 10000 - 0101$$

$$= 1011$$

PX 71177

~~PX 29061~~

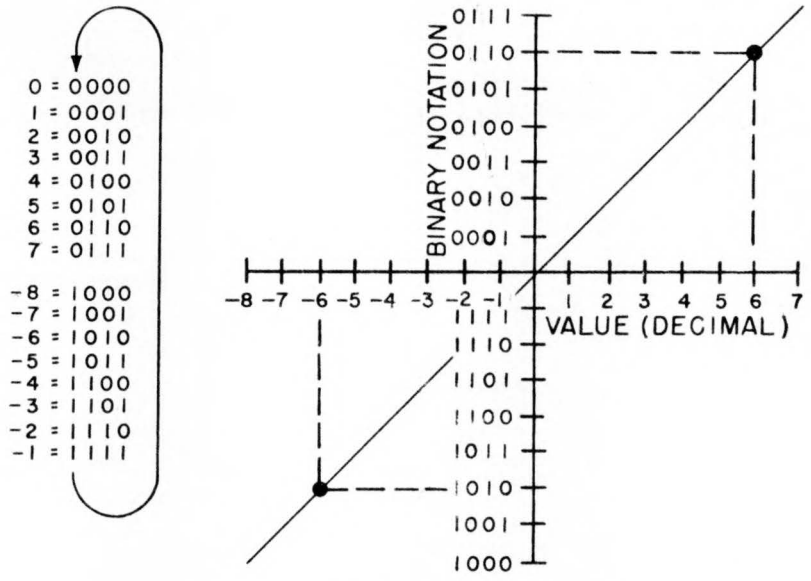


FIGURE 3
DECIMAL VS. BINARY, (TWO'S COMPLEMENT REPRESENTATION)

Note that the closed loop is still maintained and that 0000 follows 1111 . Note also that all negative values have a 1 as the leftmost digit.

The two's complement system possesses the same disadvantages for machine adaption as the ten's complement system; the rightmost non-zero digit must be treated individually. The one's complement system presents the same advantages as the nine's complement system; all digits are treated alike.

Figure 4 depicts the 15 decimal values which may be represented by 4 binary digits, plotted against their one's complement representations.

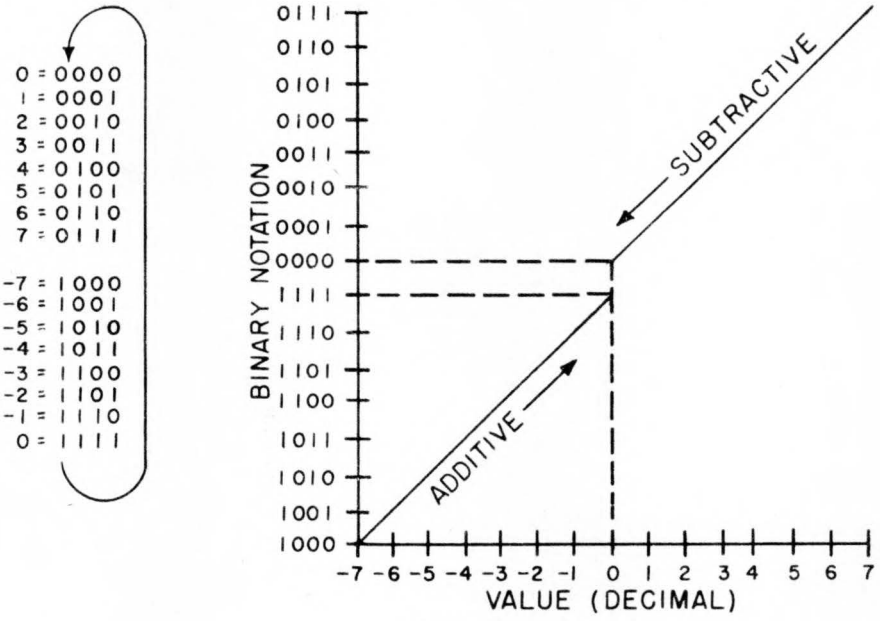


FIGURE 4
DECIMAL VS. BINARY (ONE'S COMPLEMENT REPRESENTATION)

Note that the same closed loop is used but that two values for zero are possible. The modulus of the system has been reduced from 2^4 to 2^4-1.

The negative values in Figure 4 are obtained by subtracting the absolute value from 2^4-1. For example:

$$\begin{aligned} -5 &= 2^4 - 1 - |5| \\ &= 1111 - 0101 \\ &= 1010 \end{aligned}$$

Note the important convenience of the one's complement system. The complement is formed simply by changing 1's to 0's and 0's to 1's. This is illustrated in the tabulation below. Note also that all negative numbers have a 1 in the leftmost place, as they did in the two's complement system.

0 = 0000	−0 = 1111
1 = 0001	−1 = 1110
2 = 0010	−2 = 1101
3 = 0011	−3 = 1100
4 = 0100	−4 = 1011
5 = 0101	−5 = 1010
6 = 0110	−6 = 1001
7 = 0111	−7 = 1000

Note that it is a property of the number system that odd numbers have opposite high and low order digit values. For positive odd numbers the leftmost digit is zero and the rightmost is 1. For negative odd numbers the leftmost is 1 and the rightmost 0.

In an additive system zero is always represented as 111....1 (negative zero since the leftmost digit is 1) and in a subtractive system as 000.....0 (positive zero). In Figure 4 the vertical step from 1111 through 0000 is analogous to notching the wheel of Figure 1 an extra position.

The foregoing discussion shows that the one's complement of a number can be formed in a register which does not have the property of carry or borrow; whereas the two's complement of a number can be formed only in a register having carry or borrow properties.

F. ADDITION - Consider the simple problem of adding the decimal digits 8 and 6: the sum is 14.

The problem may be stated formally as:

$$8 + 6 = x \tag{25}$$

Which may be rewritten as:

$$8 + 1 + 1 + 1 + 1 + 1 + 1 = x \tag{26}$$

The problem has been simplified to a counting operation where the initial quantity is given as the starting point and a limit of 6 is set to the number of additional unit counts that are to be made. Note that the completion of the first addition, in Equation (26), exhausts the unique symbols available in the decimal system. In order to represent larger quantities, a digit, called the carry, is provided in the next column to the left. This carry indicates that one whole group of ten is to be included in the accumulation (sum). The right-hand digit represents the additional quantity of less than ten which completes the total. Had the total count increased until the number of whole groups of ten had exceeded nine, the total would also have been expressed in whole groups of hundreds. This is merely a description of Equation (1), page 2.

The process of addition, then, may be reduced to counting in an orderly and cyclic fashion; and each time the cycle is completed a carry is provided. This is a general rule of addition of quantities and is not restricted to the decimal system of notation used in the example.

Addition in the binary system is accomplished using the general rule given above. Since only two digits are involved, it is only necessary to remember four digit combinations. These are:

$$0 + 0 = 0$$
$$0 + 1 = 1$$
$$1 + 0 = 1$$
$$1 + 1 = 0 + \text{A CARRY OF 1}$$

The addition of 00010011 to 00110101 may be analyzed as follows:

COLUMN	128's	64's	32's	16's	8's	4's	2's	1's	DECIMAL
	0	0	1	1	0	1	0	1	= 53
	0	0	0	1	0	0	1	1	= 19
	0	1	0	0	1	0	0	0	= 72

1's	1 + 1 =	0 (AND A CARRY OF 1)
2's	0 + 1 = 1 (PLUS THE 1'S CARRY)	= 0 (AND A CARRY OF 1)
4's	1 + 0 = 1 (PLUS THE 2'S CARRY)	= 0 (AND A CARRY OF 1)
8's	0 + 0 = 0 (PLUS THE 4'S CARRY)	= 1
16's	1 + 1 =	0 (AND A CARRY OF 1)
32's	1 + 0 = 1 (PLUS THE 16'S CARRY)	= 0 (AND A CARRY OF 1)
64's	0 + 0 = 0 (PLUS THE 32'S CARRY)	= 1
128's	0 + 0 =	0

CHECK: 01001000 = 72

In calculating equipment, the addition of more than two values is done as repeated additions to an accumulating total. In other words, if the sum of values, A, B, C, D, and E is desired, B is first added to A, then C is added to A+B, then D to A+B+C, and finally E is added to A+B+C+D. Such additions may be performed, manually, in the following manner. Starting from right to left, count the 1's in a column, convert the decimal total to binary, place the rightmost binary digit under the column and carry the remaining digits to the next columns.

Example:

Add, 0111, 0101, 0100, 0001, AND 0110

COLUMN	128's	64's	32's	16's	8's	4's	2's	1's		DECIMAL
	0	0	0	0	0	1	1	1	=	7
	0	0	0	0	0	1	0	1	=	5
	0	0	0	0	0	1	0	0	=	4
	0	0	0	0	0	0	0	1	=	1
	0	0	0	0	0	1	1	0	=	6
	0	0	0	1	0	1	1	1	=	23

1's	1 + 1 + 1	= (BINARY 0011)	= 1 (AND A CARRY OF 1)
2's	1 + 1 + (1's CARRY)	= (BINARY 0011)	= 1 (AND A CARRY OF 1)
4's	1 + 1 + 1 + 1 + (2's CARRY)	= (BINARY 0101)	= 1 (AND A CARRY OF 10)
8's	0 + (4's CARRY)	= (BINARY 0010)	= 0 (AND A CARRY OF 1)
16's	0 + (8's CARRY)	= (BINARY 0001)	= 1
32's			= 0
64's			= 0
128's			= 0

CHECK: 00010111 = 23

G. SUBTRACTION - The introductory remarks of Paragraph (F) Addition, apply, in general, to subtraction. However, in subtraction, the counting process is reversed (negative).

Like binary addition, binary subtraction follows the pattern of the corresponding decimal operation. Since only two digits are involved, the results of only 4 combinations need be remembered. These are:

1 - 1 = 0
0 - 0 = 0
1 - 0 = 1
0 - 1 = 1 WITH A BORROW DEMANDED FROM THE NEXT - LEFT COLUMN

PX 71177

~~PX 29861~~

As an example, subtract 00010011 from 00111010

COLUMN	128's	64's	32's	16's	8's	4's	2's	1's	DECIMAL
	0	0	1	1	1	0	1	0	= 58
	0	0	0	1	0	0	1	1	= 19
	0	0	1	0	0	1	1	1	= 39

1's 0 - 1 = 1 (AND A BORROW OF 1)
2's 1 - 1 = 0 (MINUS THE 1'S BORROW) = 1 (AND A BORROW OF 1)
4's 0 - 0 = 0 (MINUS THE 2'S BORROW) = 1 (AND A BORROW OF 1)
8's 1 - 0 = 1 (MINUS THE 4'S BORROW) = 0
16's 1 - 1 = 0
32's 1 - 0 = 1
64's 0 - 0 = 0
128's 0 - 0 = 0

CHECK: 00100111 = 39

H. MULTIPLICATION

(1) GENERAL - Regardless of the notation used, (decimal, ternary, binary, etc.) multiplication is always done using the same three operations. Conventionally, these are:

a) Form the partial product of the multiplicand and the least significant multiplier digit.

b) Form the partial product of the multiplicand and the next most significant multiplier digit and shift the entry of partial product left one place. Repeat for each multiplier digit.

c) Form product by adding partial products.

A decimal example of this multiplication is shown as follows:

Multiply 257 by 127:

```
  257
  127
 1799
  514
 257
32639
```

Note that this is the conventional method of multiplication. The process of (a) starting with the least significant digit and (b) shifting the partial products left may be reversed to that of starting with the most significant digit of the multiplier and shifting the multiplicand to the right. A decimal example of this shift reversal is given below since some machines operate in this manner.

PX 71177

Multiply 257 by 127 :

$$\begin{array}{r} 257 \\ \underline{127} \\ 257 \\ 514 \\ 1799 \\ \hline 32639 \end{array}$$

(2) BINARY MULTIPLICATION - In performing a binary multiplication the same pattern is used. However, there are no multiplication tables to learn. If the multiplier contains a 1, the multiplicand is added. If the multiplier contains a 0, the multiplicand is not added. A shift to the left occurs after the inspection of each multiplier digit. As an example, multiply 00111010 by 00101101 .

				128's	64's	32's	16's	8's	4's	2's	1's	COLUMN
				0	0	1	1	1	0	1	0	MULTIPLICAND
				0	0	1	0	1	1	0	1	MULTIPLIER
				0	0	1	1	1	0	1	0	1'S COL. MULTIPLIER IS 1, ADD, SHIFT
			-	-	-	-	-	-	-	-	-	2'S COL. MULTIPLIER IS 0, NO ADD, SHIFT
		0	0	1	1	1	0	1	0			4'S COL. MULTIPLIER IS 1, ADD, SHIFT
	0	0	0	1	1	1	0	1	0			8'S COL. MULTIPLIER IS 1, ADD, SHIFT
-	-	-	-	-	-	-	-	-				16'S COL. MULTIPLIER IS 0, NO ADD, SHIFT
0	0	1	1	1	0	1	0					32'S COL. MULTIPLIER IS 1, ADD
0	1	0	1	0	0	0	1	1	0	0	1	0	PRODUCT

CHECK: 01010000110010 = 2610
58 x 45 = 2610

I. DIVISION

(1) DECIMAL RESTORING AND NON-RESTORING - Manual decimal division is conventionally done by what is known as the Restoring method. In this familiar method, the divisor is subtracted repeatedly from the dividend until a change of sign of the remainder occurs. The dividend is then added (restored) one time, to insure a positive remainder. The divisor is shifted to the right and the process repeated.

RESTORING

```
        0 6 5
   ┌──────────
25 ) 1 6 3 4
     0 0          0 x 25
     1 6 3
     1 5 0        6 x 25
       1 3 4
       1 2 5      5 x 25
           9      REMAINDER
```

In the first example above, 25 is subtracted from 16, to yield a -9. The divisor 25 is immediately added back to the dividend and 0 is entered as the first quotient digit. In the next step, the seventh trial subtraction of 25 from 163 creates a negative remainder so the dividend is added once and the figure 6 entered as the next digit of the quotient. The process continues in the familiar manner.

Another method, known as Non-Restoring Division, employs a slightly different technique. First, as in restoring division the divisor is subtracted repeatedly from the dividend until a negative remainder is produced. Second, the divisor is shifted and added until a positive remainder occurs. Next, the divisor is shifted and subtracted until a negative remainder again appears. This process of subtract, shift; add, shift; subtract, etc., continues until the divisor has been shifted the entire length of the dividend as in restoring division.

NON-RESTORING

```
        +1-4+6
   ┌──────────
25 ) 1 6 3 4
     -2 5          1 x 25
     -0 9 0
        3
     -0 8 7
     +1 0 0        4 x 25
     +0 1 3 4
     -0 1 5 0      6 x 25
     -     1 6     SINCE REMAINDER IS
           2 5     NEGATIVE, ADD DIVISOR
             9     AND SUBTRACT 1 FROM
                   QUOTIENT
```

CHECK: + 100 − 40 + 6 − 1 = 65

In the example of non-restoring division given above, the divisor 25 is subtracted from 16 to yield -9. This process of obtaining alternate negative and positive remainders is the reason for the name non-restoring, i.e., the negative remainders are not restored to positive remainders as was done in restoring division. Note that the next digit of the dividend, 3, is added to -90 to yield -87. The first digit of the quotient is +1. The divisor is next added to -87 a sufficient number of times to create a positive remainder. The next digit of

the quotient is therefore -4. The process alternates between subtraction and addition changing with each change of sign in the remainder. The quotient of +1, -4, +6 is evaluated as $100 - 40 + 6 - 16/25$ or $66 - 16/25$.

An answer identical to that obtained with restoring division is obtained by always correcting to a positive remainder. To obtain a positive remainder the answer is expressed as $65 + 9/25$. The correction is needed only for negative remainders and is obtained simply by adding the divisor to the remainder and reducing the quotient by 1.

A further discussion of restoring and non-restoring division will be found in the appendixed "Notes on Machine Division."

(2) BINARY RESTORING - Restoring binary division is done in the same fashion as restoring decimal division and is illustrated in the following example.

Divide 01110101 by 10101

```
          00000101     QUOTIENT
10101   )01110101
          10101        11101 > 10101, ∴ ENTER 1 IN QUOTIENT AND
          --------     SUBTRACT.
          0100001      10000 < 10101, ∴ ENTER 0 IN QUOTIENT, DO
                       NOT SUBTRACT, AND BRING DOWN 1 FROM DIVIDEND.
            10101      100001 > 10101, ∴ ENTER 1 IN QUOTIENT AND
          --------     SUBTRACT.
            01100      REMAINDER

   101 = 5
  1100 = 12 OR                    CHECK:    5
5 WITH REMAINDER OF 12                    21 )117
                                             105
                                             ---
                                              12
```

(3) BINARY NON-RESTORING - Non-restoring division in the binary system is accomplished by a series of approximations which approach the quotient in powers of two. An example and explanation of this type of division is given below. A graphical representation of the same example is given in Figure 5.

PX 71177

DECIMAL	BINARY
$\frac{36}{4} = 9$	$\frac{100100}{100} = 1001$

```
                +-+--
100    100 ) 0100100
              1000000
             -011100     REMAINDER
              100000
             +00100      REMAINDER
               10000
             -01100      REMAINDER
                1000
              -100       REMAINDER
                 100
                   0     REMAINDER
```

APPROXIMATIONS

```
+10000
 -1000
  +100
   -10
    -1
  1001    QUOTIENT
```

In making the first approximation, it is convenient to select an estimated quotient which will ensure a negative remainder; therefore an approximation is selected which, when multiplied by the divisor, will produce a product exceeding the dividend. When the divisor in the example (100) is multiplied by the first approximation (10000) the product is (1000000). This product is subtracted from the dividend in the conventional manner and a negative remainder occurs indicating that the first approximation is too large. The negative sign of the remainder means that a positive overcast has occurred, and a + sign is placed in the appropriate place in the quotient as a shorthand, space saving notation for this fact. The approximation (10000) is placed in the column to the right of the example for ease in computation. The remainder then becomes the new dividend and the process is repeated. This time the approximation is (1000) and upon subtraction of the product of the divisor and the approximation a positive remainder (100) is produced. The positive remainder indicates that a negative overcast has occurred; therefore a - sign is placed in the appropriate place in the quotient and the approximation (1000) takes its place in the column to the right of the example. This process outlined above is repeated as shown in the example until a remainder less than the divisor occurs. The quotient is then obtained by summing the approximations and adding the final remainder. In the event the remainder is negative, the quotient must be reduced by one and the divisor must be added to the remainder. This will result in the desired positive remainder since, for a complete division, the remainder may never exceed the divisor in absolute magnitude.

PX 71177

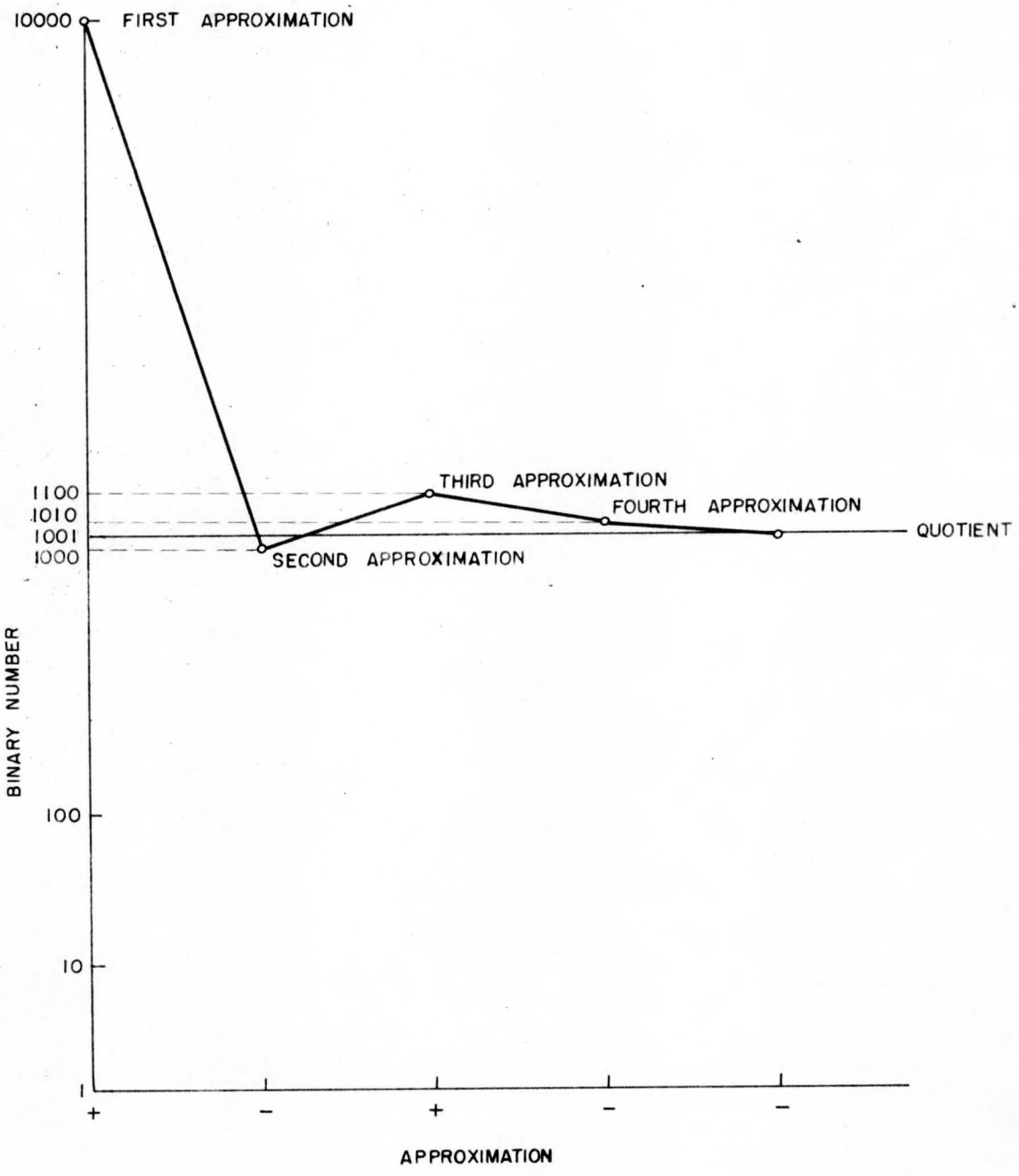


Figure 5. Graphical Representation of Non-Restoring Binary Division