AN ABSTRACT OF THE THESIS OF

IZAAK LEO UNGER for the Ph. D. in Mathematics
(Name) (Degree) (Major)

Date thesis is presented Jan 14, 1965

Title ELEMENTS OF A FAST DECIMAL ARITHMETIC UNIT

Abstract approved Redacted for Privacy
(Major professor)

A systematic and rigorous derivation of the Boolean functions that represent the three operations of the ring of integers in the 1-2-4-5 code is developed from their corresponding tables. The same is done for numerical complementation of a number. The equations of the latter are combined with those for addition to illustrate a representative calculating section of a practical arithmetic unit. The addition equations are also combined with the subtraction equations to form a so-called adder-subtractor which is shown to be faster in operation than a corresponding so-called adder-complementor that uses numerical complementation for subtraction. The amount of physical devices for such a unit, judged from the equations, turns out to be significantly less than twice the amount required for just an adder. For the adder-subtractor Boolean functions to determine the eventual sign of a sum or of a difference of two integers of arbitrary signs are developed as well.

The equations of the multiplier represent a scheme of

multiplication suggested in "Synthesis of Electronic Computing and Control Circuits" (The Annals of the Computation Laboratory of Harvard University, v. 27, p. 198). When compared to those of the adder they indicate that the amount of physical devices that is required to mechanize them is comparable to that required by the adder. In reality these quantities depend on the number of digits these units are designed to process simultaneously.

Simplifications of some of the derived equations are carried out by the Veitch diagram for six independent Boolean variables, by Samson and Mills' or Quine's consensus and prime implicant scheme and by, what shall be called for the lack of an existing term, a judicious observation method.

ELEMENTS OF
A FAST DECIMAL ARITHMETIC UNIT

by

IZAAK LEO UNGER

A THESIS

submitted to

OREGON STATE UNIVERSITY

in partial fulfillment of
the requirements for the
degree of

DOCTOR OF PHILOSOPHY

June 1965

APPROVED:

Redacted for Privacy

Professor of Mathematics

In Charge of Major

Redacted for Privacy

Head of Department of Mathematics

Redacted for Privacy

Dean of Graduate School

Date thesis is presented Jan 14, 1965

Typed by Lucinda M. Nyberg

PREFACE

In the voluminous digital computer literature of today it is hard to find a down-to-earth guide to the design of the most important part of an automatic calculating machine. Most books on the subject deal in generalities. Some elaborate on certain unique features of specific computers, some may even discuss at length selected functional components of a computer. The majority, however, seems to devote little space to the procedure of translating arithmetic operations that one would like a computer to perform, or other operations, for that matter, into equivalent Boolean functions which in turn could be simulated by physical devices. In terms of textbook terminology this can perhaps be described as the lack of an "example" to the reader who has been showered with "theory". The absence of such examples can readily be justified by their exhorbitant lengths as is evidenced by this treatment which is only part of such an example. The justification for such a partial treatment is length again. However, as it deals with the constituents of the most important part of a digital computer, namely the arithmetic unit, it is hoped that the short treatment will not appear to be incongruous with some of the objectives of this thesis.

Another objective was the desire to treat the development of the Boolean computing functions methodically and rigorously. The first part of this objective is accomplished by starting out with the tables of the three operations of the ring of integers. (A numerical complementation table is also introduced in order to derive the corresponding Boolean functions.) The second part of this objective is achieved by a strict adherence to the rules of Boolean algebra once a Boolean expression has been derived and by otherwise justifying the steps taken to arrive at some conclusions regarding ordinary arithmetic or algebra.

The motivation for the writing of this thesis was not entirely the reason cited above. The latter offered an appropriate setting for the exposition of an idea for a fast multiplier in a binary coded decimal computer. It was originally conceived while working on the Alwac 800 arithmetic unit. However, a search in the literature prior to the preparation of the manuscript revealed that such a scheme had already been proposed, ([10], p. 198). On the other hand no evidence could be found that such a scheme had actually ever been implemented. Most computers execute repeated addition for the multiplication of two numbers. The Mark II used a combination of decimal shifts, repeated addition and subtraction, in order to perform the multiplication operation, ([9], p. 26). The Mark I built up the nine multiples of the multiplicand and used them as needed to

add them up to obtain the product, ([8], p. 22). A search in the literature failed to reveal other than an indication ([10], p. 198) that direct multiplication of a multi-digit multiplicand by a single digit multiplier may have been realized in a computer in the past. Thus, although such a scheme of multiplication may very well have been implemented no published record thereof could be found. Therefore, making the details of such a method of digital computer multiplication available to the interested is considered sufficient justification of part of this thesis.

A binary code for the decimal representation of the digits had to be chosen in order to carry out these "examples". The association with the Alwac 800 computer has naturally influenced the choice of this code in this thesis. Since the transition to another code is a perfunctory operation the choice of a particular one should not be considered as detracting from the generality of the treatment in this respect. This code is 1-2-4-5.

The parts of the arithmetic unit to be taken up here are the adder, subtractor, multiplier, complementor and counter. A complementor in a computer with a subtractor would be redundant, but for the sake of achieving reasonable completeness of the treatment of an arithmetic unit it will be given equal attention to that of the other parts. The Boolean equations of the adder and complementor to be presented are those employed in the Alwac 800 and were originally

CONTENTS

DEFINITIONS OF SYMBOLS

a	numerical variable ranging over 0 - 4; element of a ring
A	numerical augend or subtrahend
A_e	Boolean variable representing even parity of A if $A_e = 1$
A_i	Boolean variable representing augend, subtrahend, multiplicand and complementand digit weighted numerically at $1 = 1, 2, 4, 5$
a_5	numerical variable ranging over 0, 5
b	element of a ring
B_n	denary borrow of 1 by the n-th subtrahend digits (from the (n+1)-st subtrahend digit
B_q	quinary borrow by a minuend digit modulo 5 from a corresponding subtrahend digit modulo 5
$\overline{C}_j$	multiplication carry weighted at j x 10, j = 1 - 8
C_n	denary carry of 1 from the n-th digit pair in a sum
C_q	quinary carry from an augend digit modulo 5 and an addend digit modulo 5
C_t	denary carry of 1 in a counter by increments of 1
$(C*B)_n$	$C_n Q_a + B_n Q'_a$
D	number represented by the first m subsubtractors
D_i	Boolean polynomial in A_i, I_1 and B_q or in G_j representing partial or total difference digit constituent weighted at $i = 1, 2, 4, 5$
$D_{0,n}$	$D'_1 D'_2 D'_4 D'_5 = 1$ of the n-th partial difference digit
e	digit indicated by the extra digit subsubtractor

n digit ordinal of a number counted from the least significant one

N complementand*; minuend

N_q quinary carry modulo 10 from a multiplicand digit modulo 5 and a multiplier digit modulo 5

P number with respect to which a numerical complement is to be formed

P_i Boolean polynomial in A_i, I_i and N_q or in M_j representing partial product digit constituent weighted at $i = 1, 2, 4, 5$

R_i Boolean general variable weighted at numerically at $i = 1, 2, 4, 5$ for $R_i = 1$ and at 0 for $R_i = 0$

Q_a adder-subtractor operation corresponding to arithmetic addition of non-negative integers

Q'_a adder-subtractor operation corresponding to arithmetic subtraction of non-negative integers

Q_c adder-complementor operation corresponding to complementing numerically each addend digit with respect to 9

Q'_c adder-complementor operation corresponding to addition of non-negative integers

S_i Boolean polynomial in A_i, I_i and C_q or in H_j or in L_j representing partial or total sum digit constituent weighted at $i = 1, 2, 4, 5$

$S_{9,n}$ $S_4 S_5 = 1$ of the n-th partial sum digit

$(S*D)_i$ $S_i Q_a + D_i Q'_a$

V_n denary ("standing on 0's") eventual borrow of 1 to be subtracted from the n-th partial difference digit

---

*This term denotes a number to be numerically complemented with respect to some given number.

example the relation of less than or equal to was always interpreted as set proper or improper inclusion. The reason for all this is probably "first love". The cup and cap signs for the Boolean operations of join (symbolic logic 'disjunction', set 'union', computer engineering 'or') and meet (symbolic logic 'conjunction', set 'intersection', computer engineering 'and'), which are becoming standard in Boolean algebra as they have become in set theory, have been replaced by the plus sign and no sign at all, (juxtaposition)*, respectively. This is partly due to typewriter expedience and partly to time and space savings. However, in this manner the sufficient similarity between the operational rules of Boolean algebra and those of arithmetic enables one to capitalize on something that has become second nature. The complement of a Boolean variable or function is denoted by an apostrophe. The verbs 'to join' and 'to meet' have been employed to describe the action or process of forming joins and unions, respectively.

Striking examples of the use of point set techniques in Boolean algebra are the Venn and Veitch diagrams. Again the reader is expected to have a mastery of them, at least for four variables. The Veitch diagram in five or more independent variables requires a

---

*Matrices over a Boolean algebra ('switching' or 'Boolean' matrices) may require two types of 'multiplication'. In such cases the cup between and juxtaposition of matrices are used to denote different operations ([5], p. 148).

rule in addition to those for four independent variables or less. It is pointed out and justified in appendix A where the Veitch diagram is used to simplify some Boolean expressions.

When reference is made to a "Boolean variable" it is customary to consider both the variable and its dual. This practice has been adhered to. But for the sake of emphasis occasionally both the variable and its complement are redundantly mentioned in the same context.

The term "generated" will be encountered every once and a while. Here it is used both in the electrical engineering and mathematical senses. In the sense of the former its mathematical meaning is "becomes Boolean 1". The precise meaning will be clear from the context. Other terms from electrical and computer engineering have also been employed. They are defined at the time when they are introduced, and whenever possible, in mathematical terminology. Sometimes computer engineering terms and phrases are injected in parentheses as paraphrases of Boolean algebra terminology. This has already been done in a previous paragraph.

The letters A_i and I_i, $i = 1, 2, 4, 5$, have been chosen as the variables of the functions that represent all independent numbers in arithmetic operations. More specifically the I_i are variables of the functions that denote addends, minuends, multipliers and the A_i serve as the variables of the functions that indicate augends,

subtrahends, multiplicands and complementands. The last word is believed to have been coined here and denotes a number that is being numerically complemented. (See chapter IV.) Such complementation is not considered as one of the arithmetic operations but is very important in digital computer technology. Other symbols are introduced as the need for them arises.

Conjunction is used in two ostensibly different ways. One is in the synthesis of the code and the other is in the execution of arithmetic and numerical complementation oprerations. This may seem inconsistent but is not really so as it is possible to interpret conjunction in such a way so as to admit both concepts. The case of addition is particularly easy. This is due to the definition of the code which makes it additive in nature, i. e., the number associated with a monomial of weighted variables is the sum of the weights of the variables whose Boolean value is 1. Therefore there is no difficulty in interpreting the value associated with a meet of two monomials as the sum of the weights of the variables of both whose Boolean value is 1. For the other arithmetic operations such a "natural" extension of meaning does not seem to be possible with the chosen code. However, conjunction of particular values of some variables can be interpreted as a predetermined result in accordance to some rules. This interpretation makes conjunction independent of arithmetic notions and in particular independent of arithmatic operations. There is no risk

of confusing the various arithmetic operations as it will be clear in the context which particular one is under consideration. In practice this risk does not exist at all as every operation is made to be executed only in the presence of a particular variable (signal) that represents that particular operation. This variable thus identifies it.

In general no short cuts have been taken to develop the Boolean expressions of the computing functions. Except at the end a step by step procedure is maintained in the derivation of the equations from the arithmatic definitions to the Boolean representations. Another exception to this, which, however, prevails through most chapters is in the simplification of the functions. But these are filled in for the adder in appendix A in two ways. A simplification of a function by Quine 's prime implicants ([2] , chapter IV) is carried out at the end of chapter V.

CHAPTER I

THE CODE

The reason for employing the 1-2-4-5 code in the present exposition is given in the preface. No attempt to justify this choice otherwise will be made as here the code is only a means to an end, a notation by which to illustrate a technique. The use of any other code would serve the purpose just as well. Different codes have different peculiarities and features which make one more suitable for a particular computer with otherwise fixed design considerations than another. For example, one code may afford more evenly distributed electrical loads on signal sources than another; in one code complementation of a number with respect to a fixed quantity may be simpler to carry out with some computing circuits than in another. Comparisons of decimal codes can be found in the literature, [16]. However, some justification for the choice of the decimal representation of the computing elements will be made and something in the defense of the decimal computer in general will be said.

As this thesis is intended to be an example of the design of fast calculating elements it is only proper to consider the fastest physical devices available for such a purpose. As matters stand today these are the two-state devices. Ten-state or decimal devices, such as ring counters, are no match for them for speed of operation.

Consequently at present it would be inconceivable to construct a fast (relative to present day standards) calculator out of anything but two-state devices. These are "naturals" for binary arithmetic. However, there is less to learn from the mechanization of a binary arithmetic unit with two-state devices than from the representation of a decimal arithmetic unit with the same devices. (Equivalently the design of a decimal arithmetic unit with ten-state devices would not hold much interest.) In other words, with two-state devices the design of a decimal arithmetic unit is more difficult than that of a binary one. Therefore once such an example is mastered the passage to the design of a binary arithmetic unit becomes somewhat of a formality.

As is well known, from the viewpoint of the number of components required to handle a given order of magnitude of numbers a decimal computer is not as economical or as efficient as a binary one. The reason, of course, is that four variables (bits) in a binary computer can represent a hexadecimal digit whereas in a decimal computer they can only describe numbers from 0 to 9. At first glance it may seem that about 35% of a decimal computer's number magnitude handling capability is wasted. Fortunately this is not the case because of what is known as redundancy states or Boolean constants and absorption states. This will become evident presently. Moreover, assuming that input and output data to be fed to and from the computer is to be in decimal notation, a decimal computer does not

require decimal to binary and binary to decimal converters. This fact should not be overlooked in evaluating the efficiency and the construction economy of decimal versus binary computers. For practical purposes the time required for the transformation can be neglected if the converters are constructed of components which operate much faster than the input and output equipment, usually the slowest link in a computing system. However, for a precise comparison the time necessary for the transformation of scale of notation should be taken into account.

For some purposes, such as accounting ("business") applications in which relatively few internal computations are performed a decimal computer is preferable to a binary. Such a computer is relatively simple in its structure and the radix conversion equipment can become a significant portion of its number of components. On the other hand, calculators intended primarily for relatively elaborate ("scientific") computations which therefore require complex structures a binary computer is more advantageous. Thus it is hard to pass absolute judgment on the merits of a decimal versus a binary computer, and vice versa. Each is optimum in certain areas of application.

The 1-2-4-5 code consists of four Boolean variables (bits) that are weighted, as the name may suggest, by the integers, 1, 2, 4, 5 when the variables have an arbitrary but fixed Boolean value. The

other Boolean value is made to correspond to the number 0. Thus this code has somewhat of an additive feature. Customarily the numerical 0 is associated with the Boolean 0 and the Boolean 1 is taken to represent 1, 2, 4, 5 by the variable (bit) under consideration. This convention will be adopted here. Physically the two Boolean values are usually represented by two distinct voltages which may be arbitrary, but which are often governed by the physical devices employed to mechanize the computing functions. Using R_i, i = 1, 2, 4, 5, to represent a general Boolean variable this code is defined as follows.

number	Boolean representation
0	$R_1' R_2' R_4' R_5'$
1	$R_1 R_2' R_4' R_5'$
2	$R_1' R_2 R_4' R_5'$
3	$R_1 R_2 R_4' R_5'$
4	$R_1' R_2' R_4 R_5'$
5	$R_1' R_2' R_4' R_5$
6	$R_1 R_2' R_4' R_5$
7	$R_1' R_2 R_4' R_5$
8	$R_1 R_2 R_4' R_5$
9	$R_1' R_2' R_4 R_5$

Each digit is thus represented by a monomial in four variables.

Since in four variables there are six monomials besides those listed above, namely,

$$R_1 R_2' R_4 R_5, \qquad R_1' R_2 R_4 R_5, \qquad R_1 R_2 R_4 R_5,$$
$$R_1 R_2' R_4 R_5', \qquad R_1' R_2 R_4 R_5', \qquad R_1 R_2 R_4 R_5',$$

they are excluded from the code and their presence is considered the result of an error. But for the sake of completeness they can be admitted provided they are assigned a Boolean constant which would be the equivalent to their (permanent) absence. By the choice of the Boolean values made for the code (0 for absence and 1 for presence) this value should be 0. Therefore let

$$R_1 R_2' R_4 R_5 = \cdots = R_1 R_2 R_4 R_5' = 0. \quad (1)$$

These six equations can be used to simplify monomials in the code itself as well as Boolean polynomials in this code.

By suitable arrangement of the equations of (1) one obtains

$$R_1 R_2' R_4 R_5 + R_1 R_2' R_4 R_5' = R_1 R_2' R_4 = 0$$
$$R_1' R_2 R_4 R_5 + R_1' R_2 R_4 R_5' = R_1' R_2 R_4 = 0$$
$$R_1 R_2 R_4 R_5 + R_1 R_2 R_4 R_5 = R_1 R_2 R_4 = 0.$$

The joins of the right sides of the first with the third and the second with the third of these equations yield

$$R_1 R_4 = R_2 R_4 = 0. \quad (2)$$

By De Morgan's rules this means that

$$R_1' + R_4' = R_2' + R_4' = 1 \qquad (3)$$

Forming the meets of these constants with R_4 one obtains

$$R_1' R_4 = R_4, \quad R_2' R_4 = R_4. \qquad (4)$$

and the meet of these two equations results in

$$R_1' R_2' R_4 = R_4. \qquad (5)$$

From (3) it also follows that

$$R_1 R_4' = R_1, \quad R_2 R_4' = R_2. \qquad (6)$$

The last three equations are very important in reducing the number of components that mechanize the equations. It was these relations which were alluded to in the assertion that less than 35% of the components are wasted in a decimal computer (p. 7), without even considering the converters between the two scales of notation. For if it were necessary to represent numbers in the hexadecimal scale with the four available variables (bits) the reduced representations (4) and (5) could not be tolerated. (These relations can also be interpreted as

$$R_4 \leq R_1', \quad R_4 \leq R_2', \quad R_4 \leq R_1' R_2',$$
$$R_1 \leq R_4', \quad R_2 \leq R_4', \quad R_1 R_2 \leq R_4'.)$$

With these contractions the 1-2-4-5 code can be formulated as

follows.

number	Boolean representation		
0	$R_1' R_2' R_4' R_5'$	$= J_o$	
1	$R_1 R_2' R_5'$	$= J_1$	
2	$R_1' R_2 R_5'$	$= J_2$	
3	$R_1 R_2 R_5'$	$= J_3$	
4	$R_4 R_5'$	$= J_4$	(7)
5	$R_1' R_2' R_4' R_5$	$= J_5$	
6	$R_1 R_2' R_5$	$= J_6$	
7	$R_1' R_2 R_5$	$= J_7$	
8	$R_1 R_2 R_5$	$= J_8$	
9	$R_4 R_5$	$= J_9$	

The key to the derivation of the Boolean computing functions for the arithmetic operations is the representation of the variables R_i in terms of the monomials J_j (cf. [3] chapter XI, section 5). The contractions derived above present somewhat of a difficulty in obtaining such a representation. The task, however, is not insurmountable. Using conditional inequality and equality wherever prima facie applicable it is easy to deduce from (7) that

$$R_1 = J_1 + J_3 + J_6 + J_8 \quad \text{(a)} \qquad R_1' \leq J_0 + J_2 + J_5 + J_7 \quad \text{(e)}$$
$$R_2 = J_2 + J_3 + J_7 + J_8 \quad \text{(b)} \qquad R_2' \leq J_0 + J_1 + J_5 + J_6 \quad \text{(f)} \quad (8)$$
$$R_4 = J_4 + J_9 \quad \text{(c)} \qquad R_4' \leq J_0 + J_5 \quad \text{(g)}$$
$$R_5 \leq J_5 + J_6 + J_7 + J_8 + J_9 \quad \text{(d)} \qquad R_5' \leq J_0 + J_1 + J_2 + J_3 + J_4 \quad \text{(h)}$$

Equations (8) (a), (b), (c) present no problem. The right side of (8) (d) will be shown now to be equal to R_5. By (7) it is clearly equal to

$$(R_1' R_2' R_4' + R_1 R_2' + R_1' R_2 + R_1 R_2 + R_4)R_5$$

so it is sufficient to prove that the expression in the parentheses is equal to 1. In view of (6) this expression is equal to

$$\begin{aligned} & R_1' R_2' R_4' + R_1 R_2' R_4' + R_1' R_2 R_4' + R_1 R_2 R_4' + R_4 \\ = \; & (R_1' R_2' + R_1 R_2' + R_1' R_2 + R_1 R_2)R_4' + R_4 \\ = \; & R_4' + R_4 = 1 \end{aligned}$$

Thus the inequality sign in (8) (d) can be changed to an equality sign. Again with the aid of (6) the right side of (8) (e) is readily found to be

$$R_1' R_2' R_4' + R_1' R_2 = R_1' R_4'. \qquad (9)$$

Since by (4) $R_4 = R_1' R_4$ and by (8) (c) $R_4 = J_4 + J_9$

one obtains

$$J_4 + J_9 = R_1' R_4.$$

Joining this expression with (8) (e) through (9) yields

$$R_1' = J_0 + J_2 + J_4 + J_5 + J_7 + J_9$$

Note that this is precisely the complement of (8) (a) with respect to

$\sum_{j=0}^{9} J_j$. * By similar reasoning it can be shown that in order to obtain equality in (8) (f), (g) it is necessary to join to their right sides the expressions $(J_4 + J_9)$ and $(J_1 + J_2 + J_3 + J_6 + J_7 + J_8)$, respectively. The inequality in (8) (h) is only apparent. The proof is analogous to that of (8) (d) or is immediate by complements. Thus the representation of the four variables R_i as joins of their monomials (7) is

$$
\begin{aligned}
R_1 &= J_1 + J_3 + J_6 + J_8 \\
R_2 &= J_2 + J_3 + J_7 + J_8 \\
R_4 &= J_4 + J_9 \\
R_5 &= J_5 + J_6 + J_7 + J_8 + J_9 \\
R'_1 &= J_0 + J_2 + J_4 + J_5 + J_7 + J_9 \\
R'_2 &= J_0 + J_1 + J_4 + J_5 + J_6 + J_9 \\
R'_4 &= J_0 + J_1 + J_2 + J_5 + J_6 + J_7 + J_9 \\
R'_5 &= J_0 + J_1 + J_2 + J_3 + J_4
\end{aligned}
\qquad (10)
$$

This representation is essentially the disjunctive canonical form of the R_i in terms of themselves. Its difference from the latter is due to the contractions (4), (5) and (6). It can be cast in the disjunctive canonical form by joining to the right sides of (10) suitable

---

* From (7) and (1) it follows immediately that $\sum_{j=0}^{9} J_j = 1$.

expressions from (1). Therefore, in what follows, it will be referred to as the 1-2-4-5 disjunctive canonical form. These forms, especially the first four, will be used repeatedly in the subsequent development.

The most outstanding feature of the 1-2-4-5 code is perhaps the independence of the R_5 variable from the other three. This imparts to the code some sort of a congruence property in the sense that two digits that are congruent modulo 5 have the same representation in terms of the "first" three variables (bits) and thus differ only in the value of R_5. But otherwise every admissible meet (intersection) of the variables R_1, R_2, R_4 occurs both with R_5 and R_5' in the representation of the digits 0 through 9. This property greatly simplifies the derivation of the Boolean functions to represent the various arithmetic operations. For in order to express any variable corresponding to R_1, R_2, R_4 or their complements in the 1-2-4-5 disjunctive canonical form it is sufficient to consider only the five monomials corresponding to J_0 through J_4 and disregard the variable that corresponds to R_5, or to consider the five monomials that correspond to J_5 through J_9 and ignore the variable that corresponds to R_5'. This will become clearer in the following chapters.

CHAPTER II

ADDITION

The most frequently used operation in digital computers is probably addition. It may therefore seem natural to start the derivation of the logic equations with the development of the adder. Since for utmost speed the latter must be a "parallel" ("simultaneous") device it is convenient to derive its Boolean functions in two principal steps. The first is the development of the equations for the addition of two single numbers modulo 10. The second is the derivation of a group of equations to interconnect as many sets of the previous equations as the number of digits desired to incorporate in the adder. In other words, the second set of equations would correspond to the generation of carries and carries due to the first carries.

As pointed out in chapter I (p. 15) one of the advantages of the 1-2-4-5 code is the pseudo congruence representation of the digits 0 through 4 and 5 through 9 modulo 5. This effectively makes the highest weighted variable (bit) a carry and thus lends itself to arithmetic modulo 5. Taking advantage of this the first principal step above can further be divided into two parts, one for digits not exceeding 4 and the other for the ciphers 5 through 9.

Consider then the addition table of two single digit numbers both ranging from 0 to 4 in least non-negative representatives modulo 5.

+	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

TABLE I

Here the heavy segments inside the table indicate that the sums under and to the right of them are associated with carries modulo 5. Denoting by I_i, A_i and S_i, $i = 1, 2, 4$, the weighted variables (bits) of the augend, the addend and the sum, respectively, in an analogous manner to that used for the variables R_i in chapter I the above table in terms of the Boolean monomials becomes

+	$A'_1\ A'_2\ A'_4$	$A_1\ A'_2$	$A'_1\ A_2$	$A_1\ A_2$	A_4
$I'_1\ I'_2\ I'_4$	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$	$S'_1\ S_2$	$S_1\ S_2$	S_4
$I_1\ I'_2$	$S_1\ S'_2$	$S'_1\ S_2$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S'_4$
$I'_1\ I_2$	$S'_1\ S_2$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$
$I_1\ I_2$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$	$S'_1\ S_2$
I_4	S_4	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$	$S'_1\ S_2$	$S_1\ S_2$

TABLE II

These tables do not define addition modulo 10, but with slight

modification can be made to do so. If C_q denote the quinary carry i. e., a variable weighted at 5 (in modulo 10 arithmetic) and C_q represent 0 then Table II, with an inversion of the order of the I_i monomials, for the sake of diagonal uniformity, can be written in the following form.

+	$A'_1\ A'_2\ A'_4$	$A_1\ A'_2$	$A'_1\ A_2$	$A_1\ A_2$	A_4
I_4	H_4	H_5	H_6	H_7	H_8
$I_1\ I_2$	H_3	H_4	H_5	H_6	H_7
$I_1\ I_2$	H_2	H_3	H_4	H_5	H_6
$I_1\ I_2$	H_1	H_2	H_3	H_4	H_5
$I_1\ I_2\ I_4$	H_0	H_1	H_2	H_3	H_4

TABLE III

where

$$
\begin{aligned}
H_0 &= S'_1\ S'_2\ S'_4\ C'_q \\
H_1 &= S_1\ S'_2\ C'_q \\
H_2 &= S'_1\ S_2\ C'_q \\
H_3 &= S_1\ S_2\ C'_q \\
H_4 &= S_4\ C'_q \\
H_5 &= S'_1\ S'_2\ S'_4\ C_q \\
H_6 &= S_1\ S'_2\ C_q \\
H_7 &= S'_1\ S_2\ C_q \\
H_8 &= S_1\ S_2\ C_q
\end{aligned}
\qquad (11)
$$

Thus C_q and C'_q can be regarded to represent the sum variables S_5 and S'_5 weighted at 5 and at 0, respectively, in the sums covered by tables II and III. As there are sums (modulo 10) of two single digit numbers that are not covered by these tables and yet require S_5 and S'_5 the latter two variables cannot be admitted in (11). In other words, (complete) expressions for S_5 and for S'_5 could not be obtained from their 1-2-4-5 disjunctive canonical forms based on (11). Put still another way $C_q \leq S_5$ (cf. (16)). Besides affording economy in writing the auxiliary variable C_q may also offer guidance for the construction and grouping of some physical devices.

Since the equivalent of arithmetic addition is to be performed when and only when combinations of A_i's and I_i's are present* in symbolic logic terms this can be represented by conjunction between

---

*This is usually not the case in practical computers. There an additional variable (control signal) is required for executing an arithmetic operation. This is particularly true when certain physical components are used for more than one arithmetic operation, or if it is desired to control the timing of the operation. Here such a variable has been omitted since in its present form the adder stands by itself. Such a signal is introduced in chapters IV and V where the adder is combined with a complementor and a subtractor, respectively.

them. Hence

$$
\begin{aligned}
H_0 &= A_1' A_2' A_4' I_1' I_2' I_4' \\
H_1 &= A_1 A_2' I_1' I_2' I_4' + A_1' A_2' A_4' I_1 I_2' \\
H_2 &= A_1' A_2 I_1' I_2' I_4' + A_1 A_2' I_1 I_2' + A_1' A_2' A_4' I_1' I_2 \\
H_3 &= A_1 A_2 I_1' I_2' I_4' + A_1' A_2 I_1 I_2' + A_1 A_2' I_1' I_2 + A_1' A_2' A_4' I_1 I_2 \\
H_4 &= A_4 I_1' I_2' I_4' + A_1 A_2 I_1 I_2' + A_1' A_2 I_1' I_2 + A_1 A_2' I_1 I_2 \\
&\quad + A_1' A_2' A_4' I_4 \qquad (12) \\
H_5 &= A_4 I_1 I_2' + A_1 A_2 I_1' I_2 + A_1' A_2 I_1 I_2 + A_1 A_2' I_4 \\
H_6 &= A_4 I_1' I_2 + A_1 A_2 I_1 I_2 + A_1' A_2 I_4 \\
H_7 &= A_4 I_1 I_2 + A_1 A_2 I_4 \\
H_8 &= A_4 I_4
\end{aligned}
$$

The 1-2-4-5 disjunctive canonical forms for S_1, S_2, S_4 and C_q and their complements follow at once from (11), in the same way as (10) with a minor difference. Thus

$$
\begin{aligned}
S_1 &= H_1 + H_3 + H_6 + H_8 & S_1' &= H_0 + H_2 + H_4 + H_5 + H_7 \\
S_2 &= H_2 + H_3 + H_7 + H_8 & S_2' &= H_0 + H_1 + H_4 + H_5 + H_6 \\
S_4 &= H_4 & S_4' &= H_0 + H_1 + H_2 + H_3 + H_5 + H_6 \\
& & &\quad + H_7 + H_8 \qquad (13) \\
C_q &= H_5 + H_6 + H_7 + H_8 & C_q &= H_0 + H_1 + H_2 + H_3 + H_4
\end{aligned}
$$

Equations (13) differ in form from (10) by the absence of H_9. According to (11) H_9 should be $S_4 C_q$. But if this were a non-constant

function it would mean that the sum of two digits ranging from 0 through 4 could be 9. Since this is impossible it is necessary to set $H_9 = S_4 C_q = 0$. (Cf. (1).) (In order to justify the equality signs of some of the equations of (13) $S_4 C_q$ can formally be joined to their right hand sides and thus facilitate the proofs. The case of S_4 follows immediately.) Thus by (12) and (13)

$$S_1 = A_1 A_2' \overset{①}{I_1'} I_2' I_4' + A_1' A_2' A_4' I_1 I_2' + A_1 A_2 \overset{②}{I_1'} I_2' I_4' + A_1' \overset{③}{A_2} I_1 I_2'$$
$$+ A_1 A_2' I_1' I_2 + A_1' A_2' A_4' I_1 I_2 + A_4 I_1' I_2 + A_1 A_2 I_1 I_2$$
$$+ A_1' \overset{④}{A_2} I_4 + A_4 I_4$$

$$= A_1 I_1' I_2' I_4' + A_1' A_2' A_4' I_1 + A_1' A_2 I_1 I_2' + A_1 A_2' I_1' I_2 I_4$$
$$+ A_1' A_2 I_4 + A_4 I_1' I_2 + A_1 A_2 I_1 I_2 + A_4 I_4$$

$$S_2 = A_1' A_2 \overset{①}{I_1'} I_2' I_4' + A_1 A_2' I_1 I_2' + A_1' A_2' \overset{②}{A_4'} I_1' I_2 + A_1 A_2 \overset{③}{I_1'} I_2' I_4'$$
$$+ A_1' A_2 I_1 I_2' + A_1 A_2' I_1' I_2 + A_1' A_2' \overset{④}{A_4'} I_1 I_2 + A_1 A_2 I_4$$
$$+ A_4 I_4 \qquad (14)$$

$$= A_1' A_2 I_2' I_4' + A_2' A_4' I_1' I_2 + A_1 A_2 I_1' I_2' + A_1' A_2' I_1 I_2$$
$$+ A_1 A_2' I_1 I_2' + A_4 I_4$$

$$S_4 = A_4 \overset{①}{I_1'} I_2' I_4' + A_1' A_2' A_4' I_4 + A_1 A_2 I_1 I_2' + A_1 A_2' I_1 I_2$$
$$+ A_1' A_2 I_1' I_2$$

$$C_q = A_4 I_1 I_2' + A_1 A_2 I_1' I_2 + A_1' \overset{④}{A_2} I_1 I_2 + A_1 \overset{③}{A_2'} I_4 + A_4 I_1' I_2$$
$$+ A_1 A_2 I_1 I_2 + A_1' \overset{②}{A_2} I_4 + A_4 I_1 I_2 + A_1 \overset{①}{A_2} I_4 + A_4 I_4$$

$$= A_4 I_1 + A_1 I_4 + A_4 I_2 + A_2 I_4 + A_1 A_2 I_2 + A_2 I_1 I_2 + A_4 I_4$$

Intuitively this set of equations can be interpreted to indicate some of the alternate circumstances in terms of the augend and addend variables under which particular S_i and C_q should be generated. To obtain the duals S_i' and C_q' either the right hand sides of (14) can be complemented (inverted) by De Morgan's rules or they can be derived directly from the right hand set of (13). The circled numbers over some of the terms should be ignored for the time being. Their significance will become apparent in the development of the adder-subtractor in chapter V.

Since the term C_q' is required for the generation of S_5, as will shortly become evident, it is given hereunder ---

$$C_q' = A_1' A_2' A_4' + I_1' I_2' I_4' + A_4' I_2' I_4' + A_2' A_4' I_4' + A_1' A_2 I_1' I_2 \quad (15).$$

Justification of this equation as well as the simplifications of (14) are to be found in appendix A.

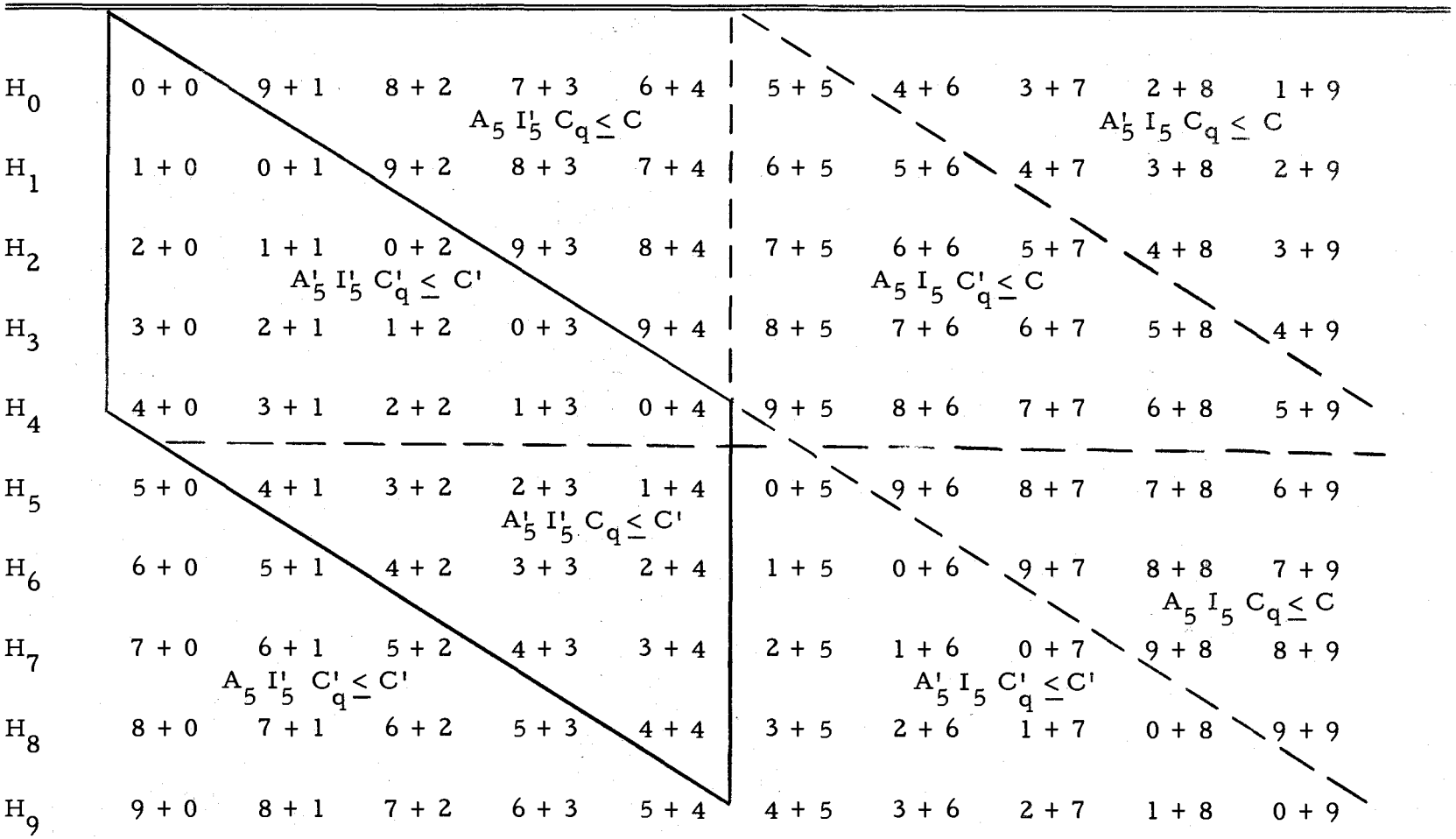
Although physical devices that perform the operation of Boolean complementation (inversion) are available in practically all computers it may not be feasible to employ them for such a purpose. This is very likely to be the case in a parallel (simultaneous) adder in which speed of operation is of prime importance. A physical

device, regardless of how fast it may act, requires a non-zero time period in which to perform its function. Then, compared to the length of time its associates take to complete their tasks, it may seem to be rather slow in its performance. Specifically with the case on hand it is desired to form all S_i simultaneously. This means that unless the delay of the generation of C'_q is negligible as far as the physical producers of S_5 are concerned they must be supplied with signals derived from the A_i and the I_i sources directly at the instants the latter appear for processing as they are given in equation (15).

Before deriving the equation for S_5 let the interpretation of the H_j be generalized. Instead of indicating by its subscripts the total sum of two digits neither of which exceeds 4 let H_j denote now the least significant digit in the sum of any two decimal digits. This amounts to interpreting now the H_j modulo 10. Since previously no sum exceeded one decimal digit the new definition is consistent with the former.

Consider now table IV (p. 24) which contains all 100 permutations of the augends and of the addends that can occur in the sum of two decimal digits. It is partioned into eight triangular parts. Each is characterized by a combination of the variables A_5, I_5, C_q and their complements. These triangles also contain the variable C, the denary carry. In regard to the former variables the word "characterized" was not chosen accidentally as the designation of these triangles

TABLE IV SINGLE DECIMAL DIGIT ADDITION TABLE



Sums above and below the main diagonal line require or do not require two decimal digits for their representations. Their Boolean characteristics are C and C', respectively.

by Boolean monomials in these three variables is unique. (In fact it is equivalent to defining them by a necessary and sufficient condition.) Since there are only eight distinct monomials in three independent Boolean variables verification is trivial. Thus the sums in some particular triangle have the property that they are generated by augends and addends that have or do not have the 5 weighted variable (bit) in their representations and that at the same time produce or do not produce a quinary carry from their variables that are weighted by 4 or less.

Posing now the rhetorical question, "Under what circumstances should an S_5 be generated?" the answer can be found by inspection of table IV, "Whenever the summands are characterized by an odd number of 5 weighted variables A_5, I_5 and C_q.". Thus directly from this table it is found that

$$S_5 = A'_5 I'_5 C_q + A_5 I_5 C_q + A_5 I'_5 C'_q + A'_5 I_5 C'_q \qquad (16).$$

This equation can be derived in a systematic way from 10 x 10 versions of tables II and III (p. 17, 18) based on table IV (p. 24) by using the 1-2-4-5 disjunctive canonical form for S_5 that is equivalent to R_5 in (10). The expressions obtained would not contain C_q and C'_q but their forms in terms of the variables A_i, I_i, $i = 1, 2, 4$. Doing this seems to be less elegant than what has been done. This would have required to simplify a function of eight variables. Also note that S_5 depends

not only on I_5 and on I_5' but on A_i, I_i, i = 1, 2, 4, (through C_q) as well.

Referring back to table IV it will be noted that the variables A_5, I_5, C_q (and their complements) are connected with C (and its complement) not by conjunction but rather by inequality (implication). For although combinations of these four variables through conjunction would be unique for each triangle the set of them would fail to be complete. In other words the sixteen possible monomials in these four variables would not be exhausted by the eight triangles. The reason is that the variables A_5, I_5, C_q and C are dependent and thus must be connected by implication which in Boolean algebra is a generalization of equality*. This gives immediately the polynomial

$$\begin{aligned} C &= A_5 I_5' C_q + A_5' I_5 C_q + A_5 I_5 C_q' + A_5 I_5 C_q \\ &= A_5 I_5 + A_5 C_q + I_5 C_q. \end{aligned} \tag{17}$$

The simplification of this equation will be justified in appendix A. Its complement, directly from the table or by De Morgan's rules, is

$$C' = A_5' I_5' + A_5' C_q' + I_5' C_q'. \tag{18}$$

Since the sum of two digits may also require two digits for its representation the sum of two m-digit numbers may need m + 1 digits

---

*However, since $A_5' I_5' C_c' \leq C'$ it follows that $A_5' I_5' C_q' C' = A_5' I_5' C_q'$. Analogous relations to this hold for the other seven monomials of table IV.

to denote it. To prevent the possible loss of the most significant digit of a sum a subadder extra over the number of independent digits in the augends and in the addends that the computer is intended to accept is incorporated. As the (m + 1)-st addend can only be 1 or 0 the mechanics of this extra digit subadder can be highly simplified and cost economy realized. Its capability can be limited to adding an addend of 1 to an arbitrary decimal single digit augend. In a sense it is a dynamic counter by 1 because whenever 1 is not being added to the augend 0 is. (Cf. chapter VII.) Since the addend is to assume only the numerical values 0 and 1 its Boolean representation requires only one variable (bit). With this simplification the addition tables for this subadder corresponding to tables I and III (p. 17 and 18) but calculated modulo 10 are given hereunder.

+	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	1	2	3	4	5	6	7	8	9	0

TABLE V

The analog of table IV and the equivalents of equations (11) are combined in table VII. In these C_q and C'_q have been replaced by S_5 and S'_5 as the expressions for the former in this case, as is evident from the table, are very simple and thus are not worthwhile deriving separately. This, by the way, indicates that the expressions for C_q

+	$A'_1A'_2A'_4A'_5$	$A_1A'_2A'_5$	$A'_1A_2A'_5$	$A_1A_2A'_5$	$A_4A'_5$	$A'_1A'_2A'_4A_5$	$A_1A'_2A_5$	$A'_1A_2A_5$	$A_1A_2A_5$	A_4A_5
I'_1	H_0	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8	H_9
I_1	H_1	H_2	H_3	H_4	H_5	H_6	H_7	H_8	H_9	H_0

TABLE VI

and C'_q in the other subadders did not have to be derived at all and that S_5 could have been expressed directly in terms of A_i and I_i. (Cf. the discussion subsequent to equation (16) p. 25.) Note that in table VII and in the subsequent equations the S_i are different functions from those denoted by the same symbols in table II, and equations (11) and (14). The same is true of the H_j in tables VI, VII and those in table III (11), (12), (13).

$S'_1 S'_2 S'_4 S'_5$	=	H_0	0 + 0
$S_1 S'_2 S'_5$	=	H_1	1 + 0
$S'_1 S_2 S'_5$	=	H_2	2 + 0
$S_1 S_2 S'_5$	=	H_3	3 + 0
$S_4 S'_5$	=	H_4	4 + 0
$S'_1 S'_2 S'_4 S_5$	=	H_5	5 + 0
$S_1 S'_2 S_5$	=	H_6	6 + 0
$S'_1 S_2 S_5$	=	H_7	7 + 0
$S_1 S_2 S_5$	=	H_8	8 + 0
$S_4 S_5$	=	H_9	9 + 0

$A_5 C_q \leq C$

9 + 1	
0 + 1	
1 + 1	
2 + 1	
3 + 1	
4 + 1	
5 + 1	
6 + 1	
7 + 1	
8 + 1	

$A'_5 C'_q \leq C'$

$A_5 C'_q \leq C'$

$A'_5 C_q \leq C'$

TABLE VII

From tables VI and VII it is easily deduced that

$$S_1 = H_1 + H_3 + H_6 + H_8$$

$$= A'_1 A'_2 A'_4 I_1 + A_1 A'_2 I'_1 + A'_1 A_2 I + A_1 A_2 I'_1$$

$$= A_1' A_4' I_1 + A_1 I_1'$$

$$S_2 = H_2 + H_3 + H_7 + H_8$$

$$= A_1 A_2' I_1 + A_1' A_2 I_1' + A_1 A_2 I_1' + A_1' A_2 I_1$$

$$= A_1' A_2 + A_1 A_2' + A_2 I_1'$$

$$S_4 = H_4 + H_9 \qquad (19)$$

$$= A_1 A_2 I_1 + A_4 I_1'$$

$$S_5 = H_5 + H_6 + H_7 + H_8 + H_9$$

$$= A_4 A_5' I_1 + A_1' A_2' A_4' A_5 + A_1 A_2' A_5 + A_1' A_2 A_5$$

$$+ A_1 A_2 A_5 + A_4 A_5 I_1'$$

$$= A_4 A_5' I_1 + A_4' A_5 + A_5 I_1'$$

In forming the first three of these equations A_5 and A_5' were ignored as both occur with the same monomials in A_i, $i = 1, 2, 4$, and in I_1. This is discussed in chapter I. (Cf. also p. 15). For a similar reason I_1 and I_1' were left out from four of the terms in the second line of the equation for S_5. The inter-digit carry from this subadder, which would indicate overflow, is

$$C = C_{m+1} = A_4 A_5 I_1. \qquad (20)$$

$C_{m+1} = 1$ is only a sufficient condition for indicating overflow as a carry to the digit pair beyond the $(m+1)$-st (for which no subadder is available) can be generated in digit pairs behind it.

However, when K_{m+2}, to be defined towards the end of this chapter, becomes 1 it is necessary and sufficient to show overflow. This function depends on C_{m+1} as well as on other variables. $K_{m+2} = 1$ usually serves as an alarm signal to guard against meaningless results that may arise from overflow. The number that is present with K_{m+2} when it becomes 1 is still meaningful and may have as many as m + 2 significant digits the most significant one being 1. However, subsequent numbers that may result from additional operations on an augend associated with $K_{m+2} = 1$ may be meaningless. For if these operations cause another K_{m+2} to be formed there may be no way of knowing that it was generated more than once and thus the representation of the sum in the adder would not be the desired one.

Before proceeding further something should be pointed out about the notation used so far. To be rigorous the variables of the subadders should have been labeled so as to indicate to which digit of the complete adder they pertain, for logically all variables are in general distinct. This will be done in the derivation of the carry equations. In them the distinction between the digits of the partial sums is essential. Here it is not. The subadders of all but the extra digit are identical in form. For the sake of making the notation as little cumbersome as possible not only has there been made no distinction among the variables of the "ordinary" subadders but even

between them and those of the extra digit. There is very little risk of misinterpretation as the text makes the distinction infallible.

This disposes of the derivation of the logic equations for two digit subadder, a carry generator and a somewhat modified counter (see chapter VII). A representation of the two digit adder in terms of physical devices is given symbolically in appendix B.

In order to acquire some insight into the mechanics of the equations of a subadder developed so far an example of partial addition (without carries) will be considered. It should be understood that when the computer is "on", that is, when it is in a state ready to execute operations or in the process of carrying them out the complements of those Boolean variables which have the value 1 are 0, and vice versa. In practice 0 and 1 are represented by two distinct electrical signals (in present day computers) or by mechanical positions of electro-mechanical devices (in relay computers) or by a combination of both. Strictly speaking when the computer is "on" addition is constantly being performed in the adder even when seemingly none should be. This is not evident from the S_i equations but would be apparent from their complements had they been given. Thus when the adder is ostensibly not doing anything it does add 0 to 0. The need for generating the S_i depends on the physical construction of the computer, namely, whether there are some other variables which depend on them. For if the Boolean 1 of an S_i is

indicated by the presence of some signal the latter's absence indicates unequivocally that $S_i' = 1$. (In order to be able to distinguish n identically looking objects it is sufficient to tag only n - 1 of them with distinct marks. The absence of a mark on one distinguishes it from the rest.) This however could not be allowed with the A_i and the I_i variables that determine the S_i and the S_i' as both are also functions of the A_i' and of the I_i'. For example if A_i' were never generated physically the "and gate" (physical representation of a Boolean monomial) $A_i' I_i$ would always remain "low" (Boolean 0) and the "or gate" (physical representation of a Boolean polynomial) $A_i' + I_i$ would never become "high" (Boolean 1) unless I_i did.

Suppose now that the sum of

$$\begin{array}{r} 7\ 6\ 8 \\ +\ 9\ 2\ 4 \\ \hline \end{array}$$

is to be formed in the adder where 0 is being added to 0, i.e., $A_i = I_i = 0$, $i = 1, 2, 4, 5$, in all subadders and consequently $S_i = 0$ ($S_i' = 1$) therein. For the most significant digits 7 and 9 A_2, A_5, I_4, I_5 become 1, A_2', A_5', I_4', I_5' become 0 and A_1', A_4', I_1', I_2' remain 1. In the simplified right hand sides of S_1 and of C_q (p. 21) $A_1'\ A_2\ I_4$ and $A_2\ I_4$, respectively, become 1. C_q' goes to 0 because all its monomials either contain A_2' or I_4' or both. S_5 becomes 1 as a result of A_5 and of $I_5\ C_q$. Finally C assumes the value 1 through $A_5\ I_5$. There

is no change in the value of S_2, which is 0, because although A_2 becomes 1 A_1 in the term $A_1 A_2 I'_1 I'_2$ remains 0 and so does A_4 in the monomial $A_4 I_4$. Similarly S_4 stays at 0 as A_1 in $A_1 A_2 I_1 I'_2$, I_2 in $A'_1 A_2 I'_1 I_2$ remain 0 and A'_2 in $A'_1 A'_2 A'_4 I_4$ becomes 0. Thus the introduction of the equivalent of 7 in the augend variables and 9 in the addend variables produces 1's only for S_1, S_5, C and leaves S'_2, S'_4 unaltered at 1. This is the equivalent of 6 and a carry of 1. In the next digit pair 1's are assumed by A_1, A_5, I_2. Consequently, in the expressions for S_1 and for S_5 $A_1 A'_2 I'_1 I'_2$ and $A'_2 A'_4 I'_1 I_2$ become 1, respectively. S_4 undergoes no change since $A_1 A'_2 I_1 I_2$ continues to be 0 and neither does C_q as none of its monomials can become 1 from A_1 and from I_2 alone. Thus C'_q remains at 1 leaving S'_5 at the same value. These two together with the 1 of A_5 set $A_5 I'_5 C'_q$, and hence, S_5 to 1. C does not change value as C_q and I_5 remain at 0. Thus the equivalent of 6 + 2 produces $S_1 = S_2 = S'_4 = S_5 = 1$, which, in decoded form is 8. In the least significant digits $A_1 = A_2 = A_5 = I_4 = 1$ and $A'_4 = I'_1 = 1'_2 = I'_5 = 1$ holds. In the polynomials for S_2, C_q and C, respectively, $A_1 A_2 I'_1 I'_2$, $A_4 I_4$ and $A_5 C_q$ revert to 1. Nothing changes in the expressions for S_1, S_4, S_5. The result is then the equivalent of 2 and a carry of 1.

To complete the adder it is necessary to develop the expressions for the inter-digit carries. In order to appreciate the equations to be derived a small digession from the subject will be made.

In the decimal scale of notation a single digit can only denote a number from 0 through 9. Numbers greater than 9 require more than one digit for their representation. The sum of two single digit numbers may require a representation by more than one digit. This happens when the addition generates a carry, which is the second (most significant) digit of the sum. When adding two multi-digit numbers digit by digit carries that are no longer part of the digit representation of the sum may also be produced. These carries must be added to the addend or the augend digits next to the ones which generated them in the first place. When this is done they and the properly corresponding digits of the summands may produce carries again, and these have to be handled in a similar fashion with respect to the following digits of the numbers that are being added.

It is clear that the greater two digits are, the more likely is their addition to produce a carry. Since the largest number a digit may symbolize is 9 the "worst" possible situation for the generation of a carry is the condition in which the addend and the augend both consist entirely of 9's. On a digit by digit consideration the first sum is 18 so that a carry of 1 ensues. The latter added to the next digit pair results in 19 with a carry of 1. Similar results follow for the rest of the digit pairs. It thus becomes evident that the magnitude of any carry in an addition operation of two numbers cannot exceed 1. If it is admitted that the absence of a carry is equivalent to

the presence of a carry of value 0 then the above can be summarized by saying that in the decimal scale of notation, addition produces carries which are numerically either 0 or 1.

In "pencil and paper" addition it is customary to add the first two corresponding digits by order of significance of the augend and of the addend, mark below them and in line with them the first digit of the sum obtained, add up the next two corresponding digits and augment the sum by the carry (zero included) from the preceding two digits, write down the first digit of the augmented sum below and in line with the second pair of corresponding digits, then proceed similarly with the rest of the digits. This procedure, however, is not mandatory. For, instead, it is possible to write down the actual sum of any pair of corresponding digits under order of significance in the addend and in the augend by marking the least significant digit directly below the digits of interest and the most significant digit, the carry, one digit space to the left and below the first digit of the sum. In other words, the sum of two corresponding digits of the augend and of the addend, a two digit number itself, would be written out in full, not horizontally, however, but rather along a $+45^{\circ}$ line. The next sum of a pair of digits would be similarly treated and written down one digit space to the left of its predecessors but along the same horizontal lines. Thus two new numbers would be obtained - the upper one, consisting of any one of the ten digits 0 through 9, and the

lower one that would consist only of 0's and 1's. The sum of these new numbers would represent the originally sought result. To obtain it their addition would be necessary. This could be done in the same way the originally given numbers would be treated. However, the second addition may also produce carries. For if two corresponding digits in the original addend and augend summed to 9 and if two immediately preceding ones generated a carry of 1 the second addition for these would result in 10. The procedure could be repeated again, treating the two new numbers as originals, until a stage would be reached where all carry values would be 0. Such a stage must necessarily be reached in a finite number of steps. For at any stage the sum of the upper and lower numbers is equal to the finite sum of the original augend and addend. Since the newly obtained augends increase monotonically with every stage of addition the addend must reduce to 0 in a finite number of steps.

The above two methods are equivalent insofar as obtaining the correct result is concerned. Either one could be adapted to a computer but neither is suitable as they stand, since a better one is to be described*. When it is considered that in the "paper and pencil" method the immediate addition of a carry would take as long as the

---

*This method is known in computer engineering as "standing on nines carries", as "look ahead carries' and as "simultaneous carries". It can be characterized by its avoidance of adding a carry to a digit of value 9.

addition of two corresponding digits, and that the addition of the next pair of digits could not be executed until after the carry had been added to the next corresponding digits, it becomes apparent that such a procedure is wasteful of time and is not suitable for a fast adder. The other method, though possibly shorter, is still not as fast as it can be, as a modification thereof, to be described, is faster.

Consider the case which leads to the generation of a carry in the second addition stage. Since the carries are at most 1 it is clear that the other corresponding digit in the upper number has to be 9. In terms of the digits of the original augend and addend this can be stated that a carry is generated in the second addition whenever corresponding digits produce a carry and the sum of the next two corresponding digits has a 9 in the least significant digit. For example, let it be required to add 4 6 7 9 4 8 to 5 2 5 0 6 5. Adding the two by writing out the carries

4 6 7 9 4 8	
	original numbers
5 2 5 0 6 5	
9 8 2 9 0 3	first upper number
0 0 1 0 1 1	first lower number (carries)
0 9 9 2 0 1 3	second upper number
0 0 0 0 1 0 0	second lower number (carries)
0 0 9 9 3 0 1 3	

It is thus seen that a carry of 1 was generated in the second lower number only because there was a carry of 1 from the two digits which were followed by another pair whose sum was 9. Note also that the latter carry was eventually added to the pair of digits immediately following the ones whose sum was 9. It is now possible to develop a rule to anticipate the generation of such "secondary" carries. To this end some new symbols will be defined.

Let C_n and C'_n be the Boolean variables representing carries of 1 and of 0, respectively, from the sum of the n-th digit pair in the augend and in the addend, counted from the least significant one. Since in addition these are the only carries this representation is valid. Furthermore let $S_{4,n}$ and $S_{5,n}$ denote S_4 and S_5, respectively, in the n-th digit subadder, i.e., in the first upper number (partial sum) generated by the n-th digit pair. For convenience let $S_{4,n} S_{5,n} = S_{9,n}$. (This should not suggest that the latter function must be generated separately in the adder. The symbol is introduced here merely for notational convenience.)

Now whenever C_n is generated, $n = 1, 2, \ldots, m$ or $n = 1, 2, \ldots, m + 1$*, it must be "added" to the $(n + 1)$-st digit pair sum to complete the addition operation. Therefore let K_{n+1} be a function which

---

*The range of n is discussed later. In the immediately following discussion of the carries it will be assumed that $n = 1, 2, \ldots, m$ or $n = 2, 3, \ldots, m + 1$, depending on the indices used in the formulas.

indicates that a carry of 1 should be added to the (n + 1)-st digit pair sum. It is clear that K_{n+1} must be generated whenever C_n is (or is potentially present). Another occasion for the generation of K_{n+1} is the case in which C_n is not generated (that is, when C'_n is generated) but rather both C_{n-1} and $S_{9,n}$ are. Symbolically then

$$K_{n+1} \leq C_n + S_{9,n} C'_n C_{n-1} \qquad n = 1, 2, \ldots, m \qquad (21)$$
$$= C_n + S_{9,n} C_{n-1}$$

where it is to be assumed that $C_0 = 0$*.

K_{n+1} is also to be generated when the sum of the (n-1)-st digit pair as well as that of the n-th pair are both 9, that is, both $S_{9,n}$ and $S_{9,n-1}$ are simultaneously generated and the carry C_{n-2} from the (n-2)-nd digit pair is also generated. For C_{n-2} "added" to $S_{9,n-1}$ produces C_{n-1} and the latter combined with $S_{9,n}$ generates C_n which in turn produces K_{n+1}. Thus there exists another case to generate K_{n+1}. Symbolically now

$$K_{n+1} \leq C_n + S_{9,n} C_{n-1} + S_{9,n} S_{9,n} C_{n-2} \qquad (22)$$
$$n = 1, 2, \ldots, m$$

where it is to be assumed that $C_0 = C_{-1} = 0$. Continuing along this

---

*It will be shown later that $C_0 = 1$ may have a useful application.

line of reasoning (informal induction in disguise) it is not difficult to deduce that K_{n+1} should be generated when either one of the following monomials

$$C_n,\ S_{9,\,n}\,C_{n-1},\ S_{9,\,n}\,S_{9,\,n-1}\,C_{n-2},\ \cdots,$$
$$S_{9,\,n}\,S_{9,\,n-1}\,S_{9,\,n-3}\cdots S_{9,\,2}\,S_{9,1}\,C_0$$

is generated. Here $C_k = 0$ for $k \measuredangle 0$. Hence K_{n+1} should be less than or equal to the join of these expressions, that is,

$$K_{n+1} \leq C_n + \sum_{j=1}^{n} \prod_{i=1}^{j} S_{9,\,n+1-i}\ C_{n-j} \qquad n = 1, 2, \ldots, m$$

or

$$K_n \leq C_{n-1} + \sum_{j=1}^{n-1} \prod_{i=1}^{j} S_{9,\,n-i}\ C_{n-1-j} \qquad n = 2, 3, \ldots, m+1$$

The inequalities in the last two expressions can be reversed, for if C_n is generated, or both $S_{9,\,n}$ and C_{n-1} ($S_{9,n}\,C_{n-1}$) are generated, or all three $S_{9,\,n-1}$ and $S_{9,\,n-2}$ and C_{n-2} ($S_{9,\,n}\,S_{9,\,n-1}\,C_{n-2}$) are generated*, etc., a carry of 1 must be added to the (n-1)-st pair sum, that is, K_{n+1} must be generated. Hence

---

*A little reflection will show that only one monomial in the join representing K_{n+1} can be generated at a time.

$$K_{n+1} \geq C_n + \sum_{j=1}^{n} \prod_{i=1}^{j} S_{9,n+1-i} \; C_{n-j} \qquad n = 1, 2, \ldots, m$$

or

$$K_n \geq C_{n-1} + \sum_{j=1}^{n-1} \prod_{i=1}^{j} S_{9,n-i} \; C_{n-1-j} \qquad n = 2, 3, \ldots, m+1$$

Therefore

$$K_{n+1} = C_n + \sum_{j=1}^{n} \prod_{i=1}^{j} S_{9,n+1-i} \; C_{n-j} \qquad n = 1, 2, \ldots, m \qquad (23)$$

or

$$K_n = C_{n-1} + \sum_{j=1}^{n-1} \prod_{i=1}^{j} S_{9,n-i} \; C_{n-1-j} \qquad n = 2, 3, \ldots, m+1 \qquad (24)$$

The formal proof by induction on the number of digits n follows now. For n = 1 (23) reduces to

$$\begin{aligned} K_2 &= C_1 + \sum_{j=1}^{1} \prod_{i=1}^{1} S_{9,2-i} \; C_{1-j} \\ &= C_1 + \prod_{i=1}^{1} S_{9,2-i} \; C_0 \\ &= C_1 + S_{9,1} \; C_0 \\ &= C_1 \end{aligned}$$

which is certainly true for a single digit adder. Assuming the validity of (24) for a fixed n (i. e., for the first n-1 subadders) suppose that the n-th digit pair gives rise to a carry C_n. Then, as noted above, K_{n+1} should be generated. Furthermore if both K_n and $S_{9,n}$

are generated (simultaneously) K_{n+1} must also be generated*. Then

$$K_{n+1} \geq C_n + S_{9,n} K_n. \tag{25}$$

Conversely, if K_{n+1} is generated then it is either due to C_n alone or due to the generation of both $S_{9,n}$ and K_n ($S_{9,n} K_n$). This means that

$$K_{n+1} \leq C_n + S_{9,n} K_n.$$

Therefore

$$\begin{aligned} K_{n+1} &= C_n + S_{9,n} K_n \qquad (26) \\ &= C_n + S_{9,n} \left[C_{n-1} + \sum_{j=1}^{n-1} \prod_{i=1}^{j} S_{9,n-i} C_{n-1-j} \right] \\ &\quad C_n + \sum_{i=1}^{n} \prod_{j=1}^{j} S_{9,n+1-i} C_{n=j} \end{aligned}$$

which is (23) (with n fixed).

For the sake of as much simplicity as possible when the S_i ($S_{i,n}$) and the K_n are generated as a partial sum and as eventual carries they can immediately be converted to corresponding A_i and I_1 (with $I_2 = I_4 = I_5 = 0$), respectively, in order to make it possible to treat the partial sum as an augend and the number consisting of the carries as an addend. A complete addition operation would consist then

---

*Note that K_n is not a function of $S_{9,n}$ but does depend on $S_{9,k}$ $k < n$.

of two addition steps. These, however, would not be entirely identical. Whereas the carries C_n that are generated during the first step of the operation are retained to form together with the $S_{9,n}$ the K_n, those C_n that would result from the second addition in the operation would either be suppressed or ignored. These secondary carries are superfluous as the K_n anticipate all of them at the same time, that is, they anticipate the carries generated by themselves.

To generate K_{m+2}, where m is the number of "ordinary" (p. 26) subadders in the adder, may be fairly costly in terms of physical components. It may even be inadvisable to obtain it from (26) since the $S_{9,m}$ signal source may already be sufficiently loaded in supplying signal power to produce other monomials (cf. (27)). At best it is preferable to drain as little power from the signal sources as possible. This can be helped in the case of K_{m+2} at the expense of some time which may not be detrimental to the operation of the computer at all. More specifically, this can be done if the information conveyed by K_{m+2} can be utilized at the time when the total sum of two numbers (addition of the carries to the partial sum) is obtained just as well as when the partial sum is formed.

By (25) K_{m+2} is generated whenever C_{m+1} is and the latter by (20) is equal to $A_4 A_5 I_1$ of the (m+1)-st (extra) digit subadder. Thus

$$K_{m+2} \geq C_{m+1} = A_4 A_5 I_1$$

and the generation of this term which serves only as an alarm need not involve $S_{9,m}$. (It is though a function of $S_{9,m+1}$.) However, unlike the I_1 in the preceding subadders the I_1 in the above inequality is not an independent variable since it is formed from K_{m+1} (not before the formation of the partial sum). Therefore if the timing scheme of the computer is such that K_{m+2} is usable at the instant when the carries are added to the partial sum then the highest integral value n need assume in the expressions (21)- (24) and (26), including some of the unnumbered ones between them is as indicated alongside them. However, if overflow must be detected earlier in the addition operation then n must be allowed to be as high as m + 1 or m + 2, according as the count starts from 1 or 2, respectively.

The recursive relation (26) among the K_n which was obtained in the course of the proof of (23) holds more than mere academic interest. For through its utilization economy in the construction of the complete adder can be realized. Some reliability may also be gained. In order to appreciate this some of the K_n in terms of the C_n and the $S_{9,n}$ are listed below. They will also aid in illustrating the principle of these equations by examples to follow immediately after them. (The first of these equations is not derivable from (24) and is thus separately defined. It is required in the application of C_0.)

$$K_1 = C_0$$

$$
\begin{aligned}
K_2 &= C_1 + S_{9,1}C_0 \\
K_3 &= C_2 + S_{9,2}C_1 + S_{9,2}S_{9,1}C_0 \\
K_4 &= C_3 + S_{9,3}C_2 + S_{9,3}S_{9,2}C_1 + S_{9,3}S_{9,2}S_{9,1}C_0 \\
K_5 &= C_4 + S_{9,4}C_3 + S_{9,4}S_{9,3}C_2 + S_{9,4}S_{9,3}S_{9,2}C_1 \\
&\quad + S_{9,4}S_{9,3}S_{9,2}S_{9,1}C_0
\end{aligned}
\qquad (27)
$$

Consider now the example of page 38. At the stage of the first upper number (assuming that $C_0 = 0$)

$S_{9,1} = 0$	$C_1 = 1$		$K_2 = 1$	(due to C_1)
$S_{9,2} = 0$	$C_2 = 1$		$K_3 = 1$	(due to C_2)
$S_{9,3} = 1$	$C_3 = 0$	whence	$K_4 = 1$	(due to $S_{9,3}C_2$)
$S_{9,4} = 0$	$C_4 = 1$	by (25)	$K_5 = 1$	(due to C_4)
$S_{9,5} = 0$	$C_5 = 0$		$K_6 = 0$	not present in (27)
$S_{9,6} = 1$	$C_6 = 0$		$K_7 = 0$	not present in (27)

Consequently the carry addend (the first lower number) is 0 0 1 1 1 1 and the sum should be obtained, by ignoring now carries C_n, from

$$
\begin{array}{r}
9\,8\,2\,9\,0\,3 \\
+\,0\,0\,1\,1\,1\,1 \\
\hline
9\,9\,3\,0\,1\,3
\end{array}
$$

which is the correct result.

As another example consider a somewhat "extreme" case

$$\begin{array}{r} 9\,9\,9\,9\,0 \\ +\quad\quad 1 \\ \hline \end{array}$$

Thus

$S_{9,1} = 0 \qquad C_1 = 1 \qquad K_2 = 1$ (due to C_1)

$S_{9,2} = 1 \qquad C_2 = 0 \qquad K_3 = 1$ (due to $S_{9,2}C_1$)

$S_{9,3} = 1 \qquad C_3 = 0 \qquad K_4 = 1$ (due to $S_{9,3}S_{9,2}C_1$)

$S_{9,4} = 1 \qquad C_4 = 0 \qquad K_5 = 1$ (due to $S_{9,4}S_{9,3}S_{9,2}C_1$)

$S_{9,5} = 1 \qquad C_5 = 0 \qquad K_6 = 1$ (due to $S_{9,5}S_{9,4}S_{9,3}S_{9,2}C_1$).

The numbers to be added in the second stage line up as

$$\begin{array}{r} 9\,9\,9\,9\,0 \\ +\,1\,1\,1\,1\,1 \\ \hline \end{array}$$

Adding and ignoring the carries the correct result is obtained.

Returning now to (26) it should be pointed out that most decimal computers are designed to accept from the outside numbers of up to ten digits on which to operate. (Some may even have been made for twelve digit numbers.) With this in mind it is easy to convince oneself that the Boolean polynomial for K_{11}, or for K_{12}, when an extra digit subadder is incorporated, becomes "monstrous". To supply signal power from the various C_n and $S_{9,n}$ sources that is, from the various $S_{4,n}$ and $S_{5,n}$, sources, especially $S_{9,10}$ or $S_{9,11}$, can become a formidable engineering task and of best is inadvisable. The

reason is the higher vulnerability of high power signal generators operating at "computer speeds" as compared to those on which only moderate power demands are placed. This contention should be interpreted in the light of the available physical devices and the prevailing state of technology. However, to this date low power sources have proven to be more reliable than high ones. High loads of signal power by numerous power sinks also increase the distributed capacitance presented to the power sources. The internal impedances of the high power sources do not decrease fast enough to make this capacitance have comparable effects on the shape of the electrical signals to those which high impedance (low power) sources have on loads with less capacitance. Signal shape deterioration is inevitably attended with unreliable switching. In contrast, when formula (26) is used the signal power demand on each $S_{9,n}$ source is only for a single K_n generator. The latter is usually a device with a power gain of its own and thus can supply the signal power needs of a single K_{n+1} generator. Similar considerations are applicable to the C_n generators. Although the power these are required to furnish is far less than their K_n associates, as is easily evident from equations (27), it may still be so large that avoiding it is advisable.

For reference purposes the complements of (23), (24), (26) and (27), in this order, are listed below.

$$K'_{n+1} = C'_n \prod_{j=1}^{n} \sum_{i=1}^{j} (S'_{9,n+1-i} + C'_{n-j}) \qquad n = 1, 2, \ldots, m$$

$$K'_n = C'_n \prod_{j=1}^{n-1} \sum_{i=1}^{j} (S'_{9,n-i} + C'_{n-1-j})$$

$$K'_{n+1} = C'_n (S'_{9,n} + K'_n)$$

$$K'_1 = C'_0$$

$$K'_2 = C'_1 S'_{9,1} + C'_1 C_0$$

$$K'_3 = C'_2 S'_{9,2} + C'_2 C'_1 S'_{9,1} + C'_2 C'_1 C'_0$$

$$K'_4 = C'_3 S'_{9,3} + C'_3 C'_2 S'_{9,2} + C'_3 C'_2 C'_1 S'_{9,1} + C'_3 C'_2 C'_1 C'_0$$

$$K'_5 = C'_4 S'_{9,4} + C'_4 C'_3 S'_{9,3} + C'_4 C'_3 C'_2 S'_{9,2} + C'_4 C'_3 C'_2 C'_1 S'_{9,1} + C'_4 C'_3 C'_2 C'_1 C'_0$$

In terms of the signal power handling capabilities the last set of equations indicates that the C'_n take the place of the $S_{9,n}$.

A few pages back it was indicated that the term C_0 had an application. Indeed to justify its presence some functional usefulness should be found for it, since as already seen it can be dispensed with without apparently affecting the operation of the computer. This was essentially done in the preceding discussion of the carry equations. C_0 was permanently assigned the universal bound 0. In this way, for example, K_2 became C_1 rather than being $C_1 + S_{9,1} C_0$. Similarly the "last" term of every K_n, as given in (27), actually vanished.

Suppose, however, that it were desired to add 1 to the augend already present in the adder without going through the normal

sequence of steps of an addition operation in the computer. This problem is posed on the premise that something stood to be gained by such a detour, especially time, and that it would be used frequently enough to warrant the extra building blocks that are required. Indeed, that time would be saved follows immediately by realizing that this 1 is introduced at a stage of the addition operation which follows the entering of the number via the I_i terms after which the carry propagation would necessarily ensue. Thus, the time of some phase of the addition operation would be saved. But there is more to it. For at the same time the devices representing the I_i variables are also available to accept information through the normal sequence of addition events should it be desired to add some number to the adder contents immediately following or even somewhat before the addition of this 1 through C_0. The addition of just 1 is often required when the adder contents (number) is rounded off by shifting or truncation. By a criterion set by the magnitude of the highest significant digit lost in the shifting operation 1 can be conveniently and "swiftly" added to the number left in the adder through C_0. Another use for C_0 is in the formation of the so-called tens coplement of a number, the complement or the additive inverse with respect to the modulus 10^m or 10^{m+1}, where m has the same meaning as defined before. This is done by forming the complement of the number with respect to $10^m - 1$ or $10^{m+1} - 1$ and then adding 1 to it through C_0. This use

of C_0 is very frequent and numerous in a computer without a subtractor as subtraction in it is performed through complementation. Moreover division is performed as a series of subtractions in any digital computer so that the incorporation of C_0 can be easily justified. The details of the complementation operation are the subject of chapter IV.

CHAPTER III

SUBTRACTION

Equations of a subtractor as an independent entity can be developed in almost strict analogy to those of the adder. The major difference in the derivations of the two sets of equations would be the set of premises from which they individually evolve, namely, an addition table versus a subtraction table. However, a problem, and one of a practical nature, that did not arise in the adder does come up here. This is the determination of the algebraic sign of the difference. For the adder it was tacitly assumed that only non-negative integers would be operated on but even such an assumption here does not relieve the problem. The reason can be traced back to the lack of closure of the non-negative integers under subtraction or to, what is equivalent, the non-commutativity of the latter operation*.

*This is easily demonstrated by imposing closure on the non-negative integers by defining subtraction for any two elements x, y therefrom as

$$x - y = |x - y| ,$$

where the interpretation of the absolute value is to be made in the usual way. Then

$$y - x = |y - x|$$

and thus commutativity follows from

$$x - y = |x - y| = |y - x| = y - x.$$

Conversely, if the last equation holds for all non-negative integers then in particular, for non-zero u, v such that $u \neq v$ it follows that

$$u - v = v - u.$$

Since one of the differences must be non-negative the same must be true of the other. This argument applies of course to the non-negative real numbers as well.

An algebraic sign with a number is extra information about the latter and thus requires extra means of representation over an unsigned number, that is, a number whose sign is assumed to be constant. As there are only two algebraic signs one Boolean variable is sufficient to represent both. Clearly the sign of the difference of two positive integers depends on their relative magnitudes and although it may be desirable to represent negative numbers in the usual way by the equivalent (in the code) of absolute values accompanied by the equivalent of a negative sign this is not always practical in a computer. To do this would either require a comparison of the absolute values of the subtrahend and of the minuend or the conversion of the difference to this form were it not so subsequent to the subtraction operation. Either way may require an additional subtraction operation and thus is wasteful of time. Besides there is hardly any practical advantage to have negative numbers represented in this form inside the computer. Instead they can be represented as least positive* representatives modulo 10^{m+1} accompanied by the proper value of the sign function. Such a scheme is compatible with the computer

---

*In ordinary (modulo 0) arithmetic of the ring of integers a negative number can be characterized as the unique additive inverse of a positive number of equal absolute value. In non-zero modular arithmetic of the integers a negative number can no longer be characterized in this way since "positive" integers are also additive inverses of other "positive integers.

organization which after all is restricted to a finite number of digits and thus performs arithmetic in a ring of integers whose characteristic is 10^{m+1}. This is clearly demonstrated in the adder of the preceding chapter wherein an accumulated sum that exceeds $10^{m+1} - 1$ is transformed into its least positive representative modulo 10^{m+1}. Moreover, positive representatives of negative (modulo 0) numbers can be subjected to all three ring operations to obtain isomorphic results to those that would be obtained had some or all of the negative operands been represented in the form of absolute values with negative signs. As final results to be presented for visual inspection they can be converted to the prevalent representation. This makes subtraction completely analogous to addition both in method and in time, and thus is faster than subtraction by complementation. The latter is the subject of the next chapter.

In view of the modular nature of the subtractor the subtraction table according to which it is to be constructed has to be in modular arithmetic. This means that negative differences modulo 0 must be represented by least non-negative additive inverses modulo 10 accompanied by a label to indicate that the number under consideration is not to be taken as positive. With this in mind the analog of table I (p. 17) becomes

-	0	1	2	3	4
0	0	1	2	3	4
1	4	0	1	2	3
2	3	4	0	1	2
3	2	3	4	0	1
4	1	2	3	4	0

TABLE VIII

where the top row represents the subtrahends and the leftmost column contains the minuends. Note that in this table as in table I the arithmetic is modulo 5. It will be converted to modulo 10 later, in table X through the subscripts of the G_i.

It is evident that the differences below and to the right of the heavy line segments represent negative numbers. They can be characterized by borrows which of course correspond to the carries in addition.

In order to construct the analog of table II (p. 17) it would be desirable to use for subtraction letters that are different from the ones in table II to designate the subtrahends, the minuends, the differences and the borrows. However, comparisons of pairs of equations from (14), (15) and (16) and their correspondents (29) and (30), to be derived from table IX below and from table XI (p. 59), are intended. Eventual amalgamation of the two sets of equations will

also be attempted. All this will be easier if the letters representing the independent (input) variables used in the adder equations for the Boolean monomials are retained. Consequently the subtrahend will be expressed in A_i's and the minuends in I_i's. Differences, however, will be expressed in the suggestive D_i's, representing the same numerical values as their corresponding R_i's on page 9. Thus table VIII in terms of Boolean monomials takes the following form.

-	$A'_1\ A'_2\ A'_4$	$A_1\ A'_2$	$A'_1\ A_2$	$A_1\ A_2$	A_4
$I'_1\ I'_2\ I'_4$	$D'_1\ D'_2\ D'_4$	$D_1\ D'_2$	$D'_1\ D_2$	$D_1\ D_2$	D_4
$I_1\ I'_2$	D_4	$D'_1\ D'_2\ D'_4$	$D_1\ D'_2$	$D'_1\ D_2$	$D_1\ D_2$
$I_1\ I'_2$	$D_1\ D_2$	D_4	$D'_1\ D'_2\ D'_4$	$D_1\ D'_2$	$D'_1\ D_2$
$I_1\ I_2$	$D'_1\ D_2$	$D_1\ D_2$	D_4	$D'_1\ D'_2\ D'_4$	$D_1\ D'_4$
I_4	$D_1\ D'_2$	$D'_1\ D_2$	$D_1\ D_2$	D_4	$D'_1\ D'_2\ D'_4$

TABLE IX

The note made on page 17 in connection with table II has a corresponding meaning here. Table IX does not define subtraction modulo 10 either. Whereas in table II the reason was some "wrong" absolute values for the sum the reason here is the "wrong" algebraic signs for some of the differences, if, according to common practice, the absence of an algebraic sign or a representative thereof, shall mean a positive sign. To overcome this shortcoming table X and

equations (28) are introduced.

-	$A'_1\ A'_2\ A'_4$	$A_1\ A'_2$	$A'_1\ A_2$	$A_1\ A_2$	A_4
$I'_1\ I'_2\ I'_4$	G_0	G_1	G_2	G_3	G_4
$I_1\ I'_2$	G_9	G_0	G_1	G_2	G_3
$I'_1\ I_2$	G_8	G_9	G_0	G_1	G_2
$I_1\ I_2$	G_7	G_8	G_9	G_0	G_1
I_4	G_6	G_7	G_8	G_9	G_0

TABLE X

$$
\begin{aligned}
G_0 &= D'_1\ D'_2\ D'_4\ B'_q \\
G_1 &= D_1\ D'_2\ B'_q \\
G_2 &= D'_1\ D_2\ B'_q \\
G_3 &= D_1\ D_2\ B'_q \\
G_4 &= D_4\ B'_q \\
G_6 &= D_1\ D'_2\ B_q \\
G_7 &= D'_1\ D_2\ B_q \\
G_8 &= D_1\ D_2\ B_q \\
G_9 &= D_4\ B_q \ .
\end{aligned}
\qquad (28)
$$

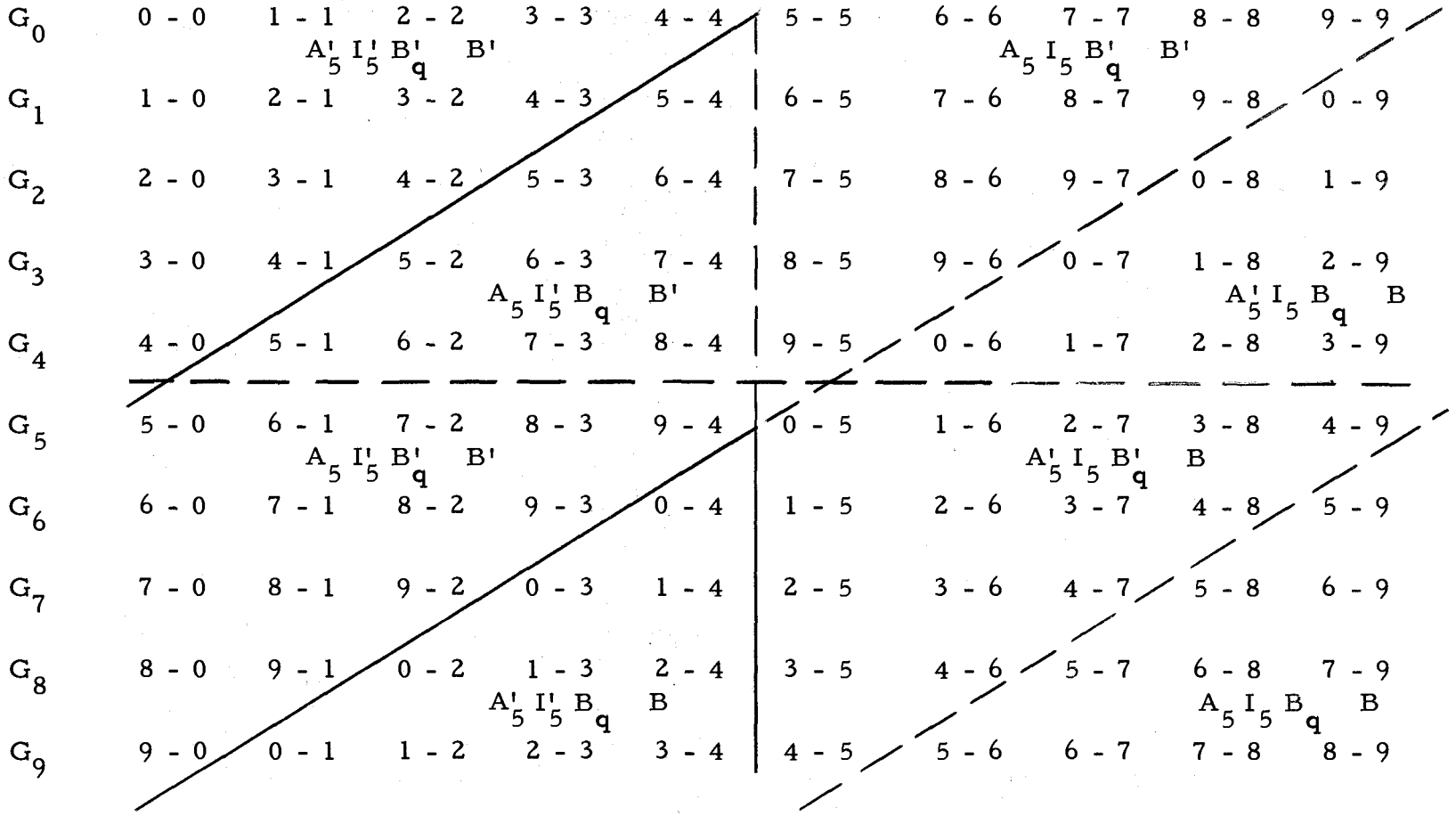
It is easy to see from table X that an equation of (28) in which B_q occurs represents a negative (modulo 0) difference whereas one involving B'_q corresponds to a non-negative (modulo 0) difference. Then in order to interpret these results modulo 0, in view of the

definition made for the D_i, B_q and B'_q must be made to represent the numerical values of -5 and 0, respectively. (The notation of (28) is thus both additive and subtractive.) However, these results can still be interpreted correctly modulo 10 according to the subscripts of the G_i, and in fact it is sufficient to do so as a single digit in the decimal scale cannot represent an integer that is greater than 9.

G_5 is absent from (28) and is thus the counterpart of H_9 missing in (11). The reason here is that the difference of two integers ranging from 0 through 4 cannot vanish and require at the same time a quinary borrow. Thus if it is desired to "complete" (28) it is necessary to set $G_5 = D'_1 D'_2 D'_4 B_q = 0$.

Table X also represents the two triangular parts of table XI (p. 59) characterized by $A'_5 I'_5 B'_q$ and $A'_5 I'_5 B_q$. The corresponding triangular parts of table IV (p. 24) which table III represents can be given the same characterization if B_q is substituted for C_q.

TABLE XI. SINGLE DECIMAL DIGIT SUBTRACTION TABLE.



Differences above and below the main line diagonal are non-negative and positive, respectively. In terms of a Boolean variable they can be characterized by B and B', respectively.

Proceeding further as in the adder

$$G_0 = A_1' A_2' A_4' I_1' I_2' I_4' + A_1 A_2' I_1 I_2' + A_1 A_2 I_1 I_2 + A_4 I_4$$

$$G_1 = A_1 A_2' I_1' I_2' I_4' + A_1' A_2 I_1 I_2 + A_1 A_2 I_1' I_2 + A_4 I_1 I_2$$

$$G_2 = A_1' A_2 I_1' I_2' I_4' + A_1 A_2 I_1 I_2 + A_4 I_1' I_2$$

$$G_3 = A_1 A_2 I_1' I_2' I_4' + A_4 I_1 I_2'$$

$$G_4 = A_4 I_1' I_2' I_4'$$

$$G_6 = A_1' A_2' A_4' I_4$$

$$G_7 = A_1' A_2' A_4' I_1 I_2 + A_1 A_2' I_4$$

$$G_8 = A_1' A_2' A_4' I_1' I_2 + A_1 A_2' I_1 I_2 + A_1' A_2 I_4$$

$$G_9 = A_1' A_2' A_4' I_1 I_4' + A_1 A_2' I_1' I_2 + A_1' A_2 I_1 I_2 + A_1 A_2 I_4$$

Hence the 1-2-4-5 disjunctive canonical forms for D_i, $i = 1, 2, 4$, and B_q are

$$D_1 = G_1 + G_3 + G_6 + G_8 \qquad D_1' = G_0 + G_2 + G_4 + G_7 + G_9$$

$$D_2 = G_2 + G_3 + G_7 + G_8 \qquad D_2' = G_0 + G_1 + G_4 + G_6 + G_9$$

$$D_4 = G_4 + G_9 \qquad D_4' = G_0 + G_1 + G_2 + G_3 + G_6 + G_7 + G_8$$

$$B_q = G_6 + G_7 + G_8 + G_9 \qquad B_q' = G_0 + G_1 + G_2 + G_3 + G_4$$

and

$$D_1 = A_1 A_2' \textcircled{1} I_1' I_2' I_4' + A_1' A_2 \textcircled{3} I_1 I_2' + A_1 A_2 I_1' I_2 + A_4 I_1 I_2$$
$$+ A_1 \textcircled{2} A_2 I_1' I_2' I_4' + A_4 I_1 I_2' + A_1' A_2' A_4' I_4 + A_1' A_2' A_4' I_1' I_2$$
$$+ A_1 A_2' I_1 I_2 + A_1' \textcircled{4} A_2 I_4$$

$$= A_1 I_1' I_2' I_4' + A_1 A_2 I_1' I_4' + A_1' A_2 I_1 I_2' + A_1 A_2' I_1 I_2$$
$$+ A_4 I_1 + A_1' A_4' I_4 + A_1' A_2' A_4' I_1' I_2$$

$$D_2 = A_1' A_2 \overset{1}{I_1'} I_2' I_4' + A_1 A_2 I_1 I_2' + A_4 I_1' I_2 + A_1 A_2 \overset{3}{I_1'} I_2' I_4'$$
$$+ A_4 I_1 I_2' + A_1' A_2' \overset{4}{A_4'} I_1 I_2 + A_1 A_2' I_4 + A_1' A_2' \overset{2}{A_4'} I_1' I_2$$
$$+ A_1 A_2' I_1 I_2 + A_1' A_2 I_4$$

$$= A_1 A_2 I_2' I_4' + A_1' A_2 I_1' I_2' + A_1 A_2' I_4 + A_2' A_4' I_1 I_2$$
$$+ A_1' A_2' I_1' I_2 + A_4 I_1 I_2'$$

$$D_4 = A_4 I_1' \overset{1}{I_2'} I_4' + A_1' A_2' A_4' I_1 I_2' + A_1 A_2' I_1' I_2' + A_1' A_2 I_1 I_2$$
$$+ A_1 A_2 I_4$$

$$B_q = A_1' A_2' A_4' I_4 + A_1' A_2' A_4' I_1 I_2 + A_1 \overset{3}{A_2'} I_4 + A_1' A_2' A_4' I_1' I_2$$
$$+ A_1 A_2' I_1 I_2 + A_1' \overset{2}{A_2} I_4 + A_1' A_1' A_2' I_1 I_2' + A_1 A_2' I_1' I_2'$$
$$+ A_1' A_2 \overset{4}{I_1} I_2 + A_1 \overset{1}{A_2} I_4$$

$$= A_1' A_2' A_4' I_1 + A_2' A_4' I_2 + A_4' I_4 + A_1' A_2 I_1 I_2$$

$$B_q' = A_1' A_2' A_4' I_1' I_2' I_4' + A_1 A_2' I_1 I_2' + A_1 A_2 I_1 I_2 + A_4 I_4$$
$$+ A_1 A_2' I_1' I_2' I_4' + A_1' A_2 I_1 I_2' + A_1 A_2 I_1' I_2 + A_1 I_1 I_2$$
$$+ A_1' A_2 I_1' I_2' I_4' + A_1 A_2 I_1 I_2' + A_4 I_1' I_2 + A_1 A_2 I_1' I_2' I_4'$$
$$+ A_4 I_1 I_2' + A_4 I_1' I_2' I_4'$$

$$= I_1' I_2' I_4' + A_1 I_1 I_2' + A_2 I_1 I_2' + A_4 + A_1 A_2 I_2$$
$$+ A_1 A_2 I_1' I_2.$$

Applying the same generalization to the G_i as that made for the H_i on page 23 it follows from the fourth equation of (10) (p. 14) that

$$\begin{aligned} D_5 &= G_5 + G_6 + G_7 + G_8 + G_9 \\ &= G_6 + G_7 + G_8 + G_9. \end{aligned}$$

However, by applying the same reasoning to table XI as that used to derive the equation for S_5 on page 25 one obtains immediately

$$D_5 = A_5 I_5 B_q + A_5 I'_5 B'_q + A'_5 I'_5 B_q + A'_5 I_5 B'_q. \quad (30)$$

Note that as in the case of S_5 the monomials of this equation consist only of variables whose weights in absolute value are 5.

From table XI again, after some reflection, it is seen that the inter-digit (denary) borrow is

$$\begin{aligned} B &= A'_5 I_5 B_q + A'_5 I_5 B'_q + A'_5 I'_5 B_q + A_5 I_5 B_q \\ &= A'_5 I_5 + I_5 B_q + A'_5 B_q \end{aligned} \quad (31)$$

and

$$\begin{aligned} B' &= A'_5 I'_5 B'_q + A_5 I_5 B'_q + A_5 I'_5 B_q + A_5 I'_5 B'_q \\ &= A_5 I'_5 + I'_5 B'_q + A_5 B'_q \end{aligned} \quad (32)$$

As in the case of the complete adder an extra digit is desirable in the complete subtractor. Here, however, its function is to generate the algebraic sign of the difference as well as to indicate extra magnitude. Although the number of significant digits in the difference

of two integers cannot exceed the largest number of significant digits of either the subtrahend or of the minuend there may be occasions where successive subtractions of numbers from immediately formed differences may generate results whose numbers of digits exceeds m. Again as in the adder this extra digit can only change by 1 (representing a borrow) in any single subtraction operation and depends indirectly on the borrow B_m of the most significant digit in the subtrahend and directly on the eventual carry V_{m+1} which is discussed later. Thus in (33) I_1 can be taken as V_{m+1}. This simplifies the mechanics of this extra sub-subtractor whose equations are derived hereunder with the aid of tables XII - XIV in complete analogy to that of the adder (p. 27) without much further comment. Here too it should be realized that the D_i and the G_j are not the same functions as those denoted by these symbols on pages 56 - 62 (Cf. p. 29). For example, according to table XIV below $G_5 \neq 0$.

-	0	1	2	3	4	5	6	7	8	9
0	0	1	2	3	4	5	6	7	8	9
1	9	0	1	2	3	4	5	6	7	8

TABLE XII

-	$A'_1A'_2A'_4A'_5$	$A_1A'_2A'_5$	$A'_1A_2A'_5$	$A_1A_2A'_5$	$A_4A'_5$	$A'_1A'_2A'_4A'_5$	$A_1A'_2A_5$	$A'_1A_2A_5$	$A_1A_2A_5$	A_4A_5
I'_1	G_0	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8	G_9
I_1	G_9	G_0	G_1	G_2	G_3	G_4	G_5	G_6	G_7	G_8

TABLE XIII

$D'_1 D'_2 D'_4 D'_5 = G_0$	0 - 0		1 - 1
$D_1 D'_2 D'_5 = G_1$	1 - 0		2 - 1
$D'_1 D_2 D'_5 = G_2$	2 - 0		3 - 1
	$A'_5 B' \leq B'$		
$D_1 D_2 D'_5 = G_3$	3 - 0		4 - 1
$D_4 D'_5 = G_4$	4 - 0		5 - 1
			$A_5 B' \leq B'$
$D'_1 D'_2 D'_4 D_5 = G_5$	5 - 0		6 - 1
$D_1 D'_2 D_5 = G_6$	6 - 0		7 - 1
	$A_5 B' \leq B'$		
$D'_1 D_2 D_5 = G_7$	7 - 0		8 - 1
$D_1 D_2 D_5 = G_8$	8 - 0		9 - 1
$D_4 D_5 = G_9$	9 - 0		0 - 1
			$A'_5 B \leq B$

TABLE XIV

$$
\begin{aligned}
D_1 &= G_1 + G_3 + G_6 + G_8 \\
&= A_1 A'_2 I'_1 + A'_1 A_2 I_1 + A_1 A_2 I'_1 + A_4 I_1 \\
&= A_1 I'_1 + A'_1 A_2 I_1 + A_4 I_1
\end{aligned}
$$

$$
\begin{aligned}
D_2 &= G_2 + G_3 + G_7 + G_8 \\
&= A'_1 A_2 I'_1 + A_1 A_2 I_1 + A_1 A_2 I'_1 + A_4 I_1 \\
&= A_2 I'_1 + A_1 A_2 + A_4 I_1
\end{aligned}
\qquad (33)
$$

$$
\begin{aligned}
D_4 &= G_4 + G_9 \\
&= A_4 I'_1 + A'_1 A'_2 A'_4 I_1
\end{aligned}
$$

$$
D_5 = G_5 + G_6 + G_7 + G_8 + G_9
$$

$$= A_1' A_2' A_4' A_5 I_1' + A_1 A_2' A_5 I_1' + A_1' A_2 A_5 I_1' + A_1 A_2 A_5 I_1'$$
$$+ A_4 A_5 I_1' + A_1 A_2' A_5 I_1 + A_1' A_2 A_5 I_1 + A_1 A_2 A_5 I_1$$
$$+ A_4 A_5 I_1 + A_1' A_2' A_4' A_5' I_1$$

$$= A_5 I_1' + A_1 A_5 I_1 + A_2 A_5 I_1 + A_4 A_5 I_1 + A_1' A_2' A_4' A_5' I_1$$

$$B = B_{m+1} = A_1' A_2' A_4' A_5 I_1 \leq V_{m+2} \qquad (34)$$

Note that in the first three equations A_5 and A_5' were ignored for the purpose of simplification (cf. p. 30 and p. 15).

In the adder the equations for the "standing on nines carries" were developed (p. 37). The analogous equations here will be termed "standing on zeros borrows"*. They can be described as equations which form borrows of 1 to be subtracted from the digits of a partial difference# of two numbers without effectively generating (new) borrows to be subtracted from the newly formed difference. They are functions of the partial difference and of the borrows generated in the process of its formation. With such a facility a subtraction operation can be completed in two steps. The first is the simultaneous

---

*To the best of knowledge this phrase has been coined here in obvious analogy to "standing on nines carries".

#Here partial difference means the number consisting of digits that are obtained by subtracting (according to order of magnitude) minuend digits from corresponding subtrahend digits modulo 10.

formation of the partial difference and a somewhat specialized minuend in the sense that it is a number that consists of 0's and 1's only. The second is the subtraction of this minuend from the partial difference and ignoring the borrows that may be generated in the course of the second digit by digit subtraction.

For the sake of convenience let $D_{o,n}$ denote $D'_1\ D'_2\ D'_4\ D'_5$ in the n-th digit subtractor counted from the least significant one. (This is purely a symbolic abbreviation and it need not have a physical representative in the subtractor (cf. p. 39). Let B_n be a borrow of 1 by the n-th digit (pair) from the (n+1)-st digit of the subtrahend. If $B_1 = 1$ and the second partial difference digit is zero ($D_{o,2} = 1$) then it is necessary to borrow from the third digit of the partial difference. Let this borrow be denoted by V_3. In general let V_n denote a borrow from the n-th partial difference digit in distinction to B_n which is a borrow by the n-th partial difference digit (pair). Returning to V_3 and the case under consideration it may also be necessary to generate it when $B_2 = 1$. Since there are no other occasions to borrow from the third partial difference it follows that

$$V_3 = B_2 + D_{o,2}\,B_1. \qquad (35)$$

If $B_1 = 1$ and both the second and the third partial difference digits are zero ($D_{o,2} = D_{o,3} = 0$) then it is necessary to borrow from the fourth one. The same should be done when B_2 is generated and the

third digit of the partial difference is zero (i. e., when $B_2 = 1$ and $D_{0,3} = 1$), or when just B_3 is generated. Then

$$V_4 \geq B_3 + D_{0,3} B_2 + D_{0,3} D_{0,2} B_1.$$

Since these are all the instances in which V_4 should be generated the above inequality (implication) can be changed into an equality so that

$$V_4 = B_3 + D_{0,3} B_2 + D_{0,3} D_{0,2} B_1.$$

Generalizing this equation by such reasoning (which is quite similar to that used for the carries in addition) it follows that

$$\begin{aligned} V_{n+1} &= B_n + D_{0,n} B_{n-1} + \ldots + D_{0,n} D_{0,n-1} D_{0,n-2} \cdots D_{0,3} D_{0,2} B_1 \\ &= B_n + \sum_{j=1}^{n-1} \prod_{i=1}^{j} D_{0,n+1-i} B_{n-j} \qquad n = 2, 3, \ldots, m \end{aligned} \tag{36}$$

For $n = 1$ it is necessary to define V_n separately, that is, $V_2 = B_1$. This is plausible inasmuch as when B_1 is generated it is necessary to subtract a 1 from the second partial difference digit so that $V_2 \geq B_1$. Conversely, if a borrow of 1 is to subtracted from the second partial difference digit it must be because B_1 has been generated. Therefore $V_2 \leq B_1$. (The considerations for the range of n given to the n associated with the carries in addition apply to this one also.)

To prove (36) let $n = 2$. Then it becomes

$$V_3 = B_2 + \sum_{j=1}^{1} \prod_{i=1}^{1} D_{0,3-i} B_{2-j}$$

$$= B_2 + D_{0,2} B_1,$$

which is (35). Assuming the validity of (36) for some fixed n suppose that the (n+1)-st digit pair causes a borrow B_{n+1} to be generated. Then it is necessary to produce V_{n+2} as a borrow to be subtracted from the (n+2)-nd partial difference digit* so that

$$V_{n+2} \geq B_{n+1}.$$

However, it is necessary to do the same thing if for some reason V_{n+1} is generated (it is necessary to subtract a borrow from the (n-1)-st digit) and the (n+1)-st partial difference digit is zero ($D_{0,n+1} = 1$). Therefore

$$V_{n+2} \geq D_{0,n-1} V_{n+1}$$

so that

$$V_{n+2} \geq B_{n+1} + D_{0,n+1} V_{n+1}.$$

Conversely, a borrow of 1 to be subtracted from the (n+2)-nd partial difference digit (V_{n+2}) is either due to a borrow originated by the

---

*Whether the (n+2)-nd partial difference digit is zero or not is immaterial at this point as the object is to define V_{n+2}.

(n+1)-st digit pair (B_{n+1}) or to a borrow generated by lower order digit pairs that should be subtracted from the (n+1)-st partial difference digit (V_{n+1}) but which is zero ($D_{0,n+1}$). Hence

$$V_{n+2} \leq B_{n+1} + D_{0,n+1} V_{n+1}$$

so that

$$V_{n+2} = B_{n+1} + D_{0,n+1} V_{n+1}.$$

Substituting the expression for V_{n+1} in (36) in this equation one obtains

$$V_{n+2} = B_{n+1} + D_{0,n+1} B_n + \sum_{j=1}^{n-1} \prod_{i=1}^{j} D_{0,n+1-i} B_{n-j}$$

$$= B_{n+1} + \sum_{j=1}^{n} \prod_{i=1}^{j} D_{0,n+2-i} B_{n+1-j}, \qquad n = 2, 3, \ldots, m$$

Making the transformation n+1 $\longrightarrow$ n in this equation (36) follows immediately.

It is possible to formulate the borrow equations (36) in terms of a "phantom" borrow B_0 in complete analogy to C_0 in the adder. The way to do this should be obvious from a comparison of the join indices in (23) and (36). However it is not as easy to find as useful an application for B_0 as it is for C_0. Aside from subtracting a 1 from a number in the subtractor "in a hurry" it could also be used in a very convenient way to turn all digits in the subtractor to 9's after first

turning them to 0's. The latter facility can safely be said to exist in every calculating device so that it would not have to be installed specifically for that purpose.

For easy reference (36) is written out hereunder for a few values of n.

$$
\begin{aligned}
V_2 &= B_1 \\
V_3 &= B_2 + D_{0,2}B_1 \\
V_4 &= B_3 + D_{0,3}B_2 + D_{0,3}D_{0,2}B_1 \\
V_5 &= B_4 + D_{0,4}B_3 + D_{0,4}D_{0,3}B_2 + D_{0,4}D_{0,3}D_{0,2}B_1 \\
V_6 &= B_5 + D_{0,5}B_4 + D_{0,5}D_{0,4}B_3 + D_{0,5}D_{0,4}D_{0,3}B_2 \\
&\quad + D_{0,5}D_{0,4}D_{0,3}D_{0,2}B_{0,1}
\end{aligned}
\tag{37}
$$

This completes the treatment of the subtractor except for the determination of the sign of a difference. After considering two examples equations to define this algebraic sign will be derived.

Examples

Suppose that m = 3 and that an extra digit subsubtractor is incorporated in the complete subtractor. Let it be required to compute the difference.

6 0 3

- 5 7 4 .

It should be realized that when these numbers are entered to be

operated on the subtractor is cleared (that is to say, all A_i are made to be 0 and consequently all $D_{0,n}$ are 1.) Recall that, as in the adder, when the subtractor is ostensibly not doing anything it does subtract the numerical zero from whatever number may happen to be stored in it. Consequently all I_i are also 0 just before the time the numbers are entered for processing. Then the A_i and the I_i are encoded in accordance with the subtrahend and the minuend, respectively. For the given numbers the result is then the following.

third digit		second digit		first digit	
$A_1 = 1$	$I_1 = 0$	$A_1 = 0$	$I_1 = 0$	$A_1 = 1$	$I_1 = 0$
$A_2 = 0$	$I_2 = 0$	$A_2 = 0$	$I_2 = 1$	$A_2 = 1$	$I_2 = 0$
$A_4 = 0$	$I_4 = 0$	$A_4 = 0$	$I_4 = 0$	$A_4 = 0$	$I_4 = 1$
$A_5 = 1$	$I_5 = 1$	$A_5 = 0$	$I_5 = 1$	$A_5 = 0$	$I_5 = 0$.

Then from (29) - (31)

$D_1 = A_1 A_2' I_1' I_2' I_4 = 1$	$D_1 = A_1' A_2' A_4' I_1' I_2 = 1$	$D_1 = 0$
$D_2 = 0$	$D_2 = A_1' A_2' I_1' I_2 = 1$	$D_2 = 0$
$D_4 = 0$	$D_4 = 0$	$D_4 = A_1 A_4 I_4 = 1$
$D_5 = 0$	$D_5 = 0$	$D_5 = 0$
$B_q = 0$	$B_q = 0$	$B_q = A_1 A_2 I_4 = 1$
$B_3 = 0$	$B_2 = A_5' I_5 = 1$	$B_1 = A' B_q = 1$
$D_{0,3} = 0$	$D_{0,2} = 0$	$D_{0,1} = 0$

According to the values of the D_i listed above the partial difference is 1 3 9. This shows that the necessary D_i are generated. It is a matter of checking the monomials in (29) - (31) to verify that no other D_i becomes 1. Referring now to (37) it is easy to deduce that

$$V_3 = B_2 = 1 \qquad V_2 = B_1 = 1$$

and $V_4 = 0$. Thus the carry number is 0 1 1 and should be subtracted from the partial difference as

$$\begin{array}{r} 1\ 3\ 9 \\ -\ 0\ 1\ 1 \\ \hline \end{array}$$

to yield 0 0 2 9 where the last 0 appears in the extra digit subsubtractor.

As another example consider a modification of the numbers in the previous one with the most significant digits of both the subtrahend and of the minuend zero. Then for the first two pairs of digits the A_i, I_i, B_q's, B_n, D_i ($D_{0,n}$) remain as before. For the third pair

$$\begin{array}{llll} A_1 = 0 & I_1 = 0 & D_1 = 0 & B_q = 0 \\ A_2 = 0 & I_2 = 0 & D_2 = 0 & B_3 = 0 \\ A_4 = 0 & I_4 = 0 & D_4 = 0 & D_{0,3} = 1 \\ A_5 = 0 & I_5 = 0 & D_5 = 0 & \end{array}$$

By (37)

$$V_4 = D_{0,n}B_2 = 1 \qquad V_3 = B_2 = 1 \qquad V_2 = B_1 = 1$$

and the second subtraction is performed as

$$\begin{array}{r} 0\ 3\ 9 \\ -\ 1\ 1\ 1 \\ \hline \end{array}$$

and yields 9 2 9 in the first three subsubtractors and 9 in the extra one. The latter is the result of

$$A_1 = A_2 = A_4 = A_5 = 0 \qquad I_1 = V_4 = 1$$

and by (33)

$$D_1 = D_2 = 0 \qquad D_4 = A'_1 A'_2 A'_4 I_1 = 1 \qquad D_5 = A'_1 A'_2 A'_4 A'_5 I_1 = 1.$$

Thus the result of the subtraction operation in the four subsubtractors is 9 9 2 9. Clearly the desired difference is a negative number and its absolute value can be obtained by subtracting the indicated result from 10^5. The fact that the extra subsubtractor did perform a non-trivial subtraction can be taken as the indication of a negative difference. However this operation is a result of $V_4 = 1$ so the latter can also be taken as the mark of the negative nature of the total difference.

It is clear that this contention can be generalized to the (m+1)-st

subsubtractor as the number of digits in the minuend cannot exceed m and the subtraction of a carry from the n-th partial difference digit is equivalent to subtracting 10^{n-1} from the partial difference. Hence the negative sign can be determined by the generation of V_{m+1}. (Note, however, that B_m is not necessary for this purpose, as the above example shows.) It can also be determined from V_{m+2}. The choice may be a matter of preference or expedience such as discussed in connection with raising the alarm with K_{m+2} or with C_{m+1} (p. 44). The problem of the alarm here is slightly more complicated than in addition. It will be taken up shortly. In the meantime let Z be the variable that denotes the algebraic sign and be such that Z = 0 is equivalent to + and Z = 1 is equivalent to -. Taking V_{m+1} as the negative sign generator then

$$V_{m+1} \leq Z. \tag{38}$$

Since negative numbers can also originate in other places, such as in the input source, the inequality sign in the above equation must be left. Using V_{m+2} as the indicator then by (34)

$$B_{m+1} \leq Z. \tag{39}$$

If (38) were used then it would be necessary to ensure that when V_{m+1} is generated for the second time, the value of Z would not change as the difference would still be negative but greater in

absolute value. This can be accomplished by a suitable two state device which once its state has been changed by a signal on some input terminal cannot be changed again by a similar signal on the same terminal. This type of device is readily available. Such a precaution would not have to be taken if equation (39) were used, as at the second time B_{m+1} would be generated the alarm would have to be set off. This is elucidated in the next two paragraphs.

Suppose that a subtraction operation is initiated with a cleared subtractor (the subtrahend and the minuend are zero) on two numbers whose number of digits does not exceed m. Let the difference of these be negative. Then V_{m+1} will be generated and the extra subsubtractor having originally represented 0 will turn to 9. (Subtraction of digit pairs is performed modulo 10 in terms of the least non-negative representatives.) At the same time or shortly thereafter B_{m+1} will be generated so that whether (38) or (39) is used, in any case, Z will become 1. Now if D is the number indicated by the first m subsubtractors then the desired difference will be

$$D - 10^m.$$

Suppose further that from this negative difference another number of m digits or less and one whose absolute value is greater than that of D is entered for subtraction. (The original difference is now as the subtrahend.) Since V_{m+1} will be generated again the extra digit

subsubtractor will turn to 8. However B_{m+1} will not be generated this time as $9 > 1$. Denoting by D again the number indicated by the first m subsubtractors in the second operation the desired difference this time is clearly

$$D - 2 \cdot 10^m.$$

In general, if the digit indicated by the extra subsubtractor is e then the desired difference will be

$$D - (10 - e)\, 10^m$$

provided $e \neq 0$ and B_{m+1} has been generated only once. For if e does vanish and B_{m+1} has not been generated at all then the desired difference is non-negative and in fact is equal to D, whereas the last expression would necessarily be negative since $D + e \cdot 10^m$ has at most m+1 digits. If B_{m+1} is generated more than once and the exact number is not known then the results calculated from the above formula are meaningless unless they are considered modulo 10^{m+1}. To keep count of the number of times B_{m+1} is generated requires a counter which is equivalent to another extra subsubtractor. However, a counter of 2 is available and it is Z. Therefore the alarm should be set off whenever both B_{m+1} and Z are equal to 1.

A somewhat simpler way than this and one that is closer to that used for the alarm in the adder is to substitute a subadder for the

subsubtractor of the extra digit. Then the alarm could be set off by the carry generated by it. The essential difference between the two methods is that with an extra subsubtractor B_{m+1} is generated the first time the difference becomes negative whereas with a subadder C_{m+1} would be generated after V_{m+1} had been generated ten times. The interpretation of the number represented by such a modified subtractor would also be simplified. For then it would be necessary to subtract a multiple equal to the number represented by the subadder of 10^m from the number represented by the m subsubtractors. Thus with the notation as above the last equation would become

$$D - e \cdot 10^m$$

and there would be no restriction on e.

CHAPTER IV

COMPLEMENTATION AND ADDITION - COMPLEMENTATION

As far as it could have been ascertained present day decimal computers have no subtractors. Their operations are performed by complementors which in a sense are equivalent to subtractors. This is possible since complementation is essentially a subtractive operation. * The reason, as usual, is the saving of monetary cost (in terms of components), which, however, sacrifices speed of operation. Since subtraction by complementation still requires addition, complementors are usually integrated with adders. Of course complementors could also be incorporated in subtractors to perform addition. This, however, is not done as it is generally agreed that the most frequent arithmetic operation in digital computers is addition and this would require more time in most computers than it would with direct addition.

The complement of a number is not unique. It depends on what is equivalent to the universal set. This can be an arbitrary (integral

---

*In Boolean algebra as well as in set theory complementation is regarded as a unary operation. Strictly speaking it is a binary operation since two sets are involved in the formation of a complement-the complementand and the universal set. The operation is the formation of the difference between the two. Munroe in [12] uses a minus sign in front of the symbol of a set to designate its complement which suggests the difference operation.

in this case) number P but for one sort of convenience it should be such that $P - N \geq 0$ where N is the largest number that is ever expected to be complemented. P is usually taken to be 10^{m+1} or $10^{m+1}-1$ where m+1* is the number of digits in N. (Cf. p. 26) When the complement is formed with respect to 10^{m+1} or $10^{m+1}-1$ the so-called "ten's complement" or "nine's complement", respectively, are obtained. The motivation for both terms should be clear. The convenience in connection with $P - N \geq 0$ is the avoidance of a Boolean function (bit) to indicate negative numbers that may result from the complementation operation.# Once this is accepted it follows that $P \geq 10^{m+1}-1$. But there is another type of convenience to be considered. Both the easiest and the fastest complement to form is the one with respect to $10^{m+1}-1$. The reason is that each digit of the complementand can individually be complemented with respect to 9. This makes it possible to construct every digit complementor, or that part which is incorporated into every subadder, in identical form. On the other hand to obtain the complement with respect to a number larger than $10^{m+1}-1$ in the same length of time as with respect to $10^{m+1}-1$ the number of components required may be

---

*m +1 rather than m is used so that the latter be consistent with the definition given to this symbol before.

#Such negative numbers should not be confused with ordinary negative numbers. The modulus of the former is some positive integer whereas that of the latter is 0.

unmanageable from a practical view point. Such a task would be comparable to that of obtaining the sum of two numbers in the same time interval that it takes to calculate their partial sum as described in chapter II in connection with the complete adder. In other words, there would be no need to add carries. However, if complementation with respect to $P \geq 10^{m+1}-1$ is to be in two steps the mechanics become relatively simple. For then it is possible to complement with respect to $10^{m+1}-1$ and then add the appropriate but fixed number to the intermediate complement that would yield the desired result. When the complement with respect to 10^{m+1} is desired, complementation with respect to $10^{m+1}-1$ is carried out first and then 1 can be added to the intermediate result via C_0 discussed in chapter II p. 49 , or by some other means. The first is by far the simplest of the second steps in this type of complementation with the available facilities described in that chapter, namely, the "standing on nines carries". As already pointed out other numbers can be added to the "nine's complement" to obtain complements with respect to other numbers. Then, however, not only the technical task may be more difficult than that for the "ten's complement" but it may also take longer. For by adding 1 via C_0 to any number, and to the "nine's complement" in particular, no carry can be generated and thus the addition operation is completed with this step. Adding an arbitrary number to the "nine's complement", including 1, not through C_0 is

the same as an arbitrary addition operation and thus also requires the carry addition time. With C_0 the first steps of the addition operation are avoided.

Since the "nine's complement" and the "ten's complement" are the most frequently employed in decimal computers the following considerations are given for both of them. Suppose it is desired to form the difference M - N where M and N are non-negative integers. It is equal to

$$10^{m+1} - [(10^{m+1} - M) + N] = (10^{m+1} - 1) - [(10^{m+1} - 1 - M) + N] \quad (40)$$

The sequence of operations, for either complement, is the following. The complement of the subtrahend M is formed, to it the minuend N is added and then the sum is complemented. The result is the absolute value of the difference. No algebraic sign can be deduced from it without detecting something in the process. Since $10^{m+1} - M > 0$ and $10^{m+1} - 1 - M \geqslant 0$ it follows that the quantities in the brackets of (40) are non-negative. Consequently the difference is negative if and only if $(10^{m+1} - M) + N > 10^{m+1}$. This strict inequality is not easy to determine. However, the conditional inequality $(10^{m+1} - M) + N \geqslant 10^m$ is easy to detect. For the latter exists whenever C_{m+1} of the extra digit of the adder (p. 30) is not zero. (Note the identical roles of the extra digits of the adder and of the subtractor, p. 63.) However, when $(10^{m+1} - M + N = 10^{m+1}$ then M = N and the difference is 0 so the

algebraic sign is immaterial. This may account for a "negative 0". Similarly when $(10^{m+1}-1-M)+N \leq 10^{m+1}-1$ $M-N \geq 0$ and when $(10^{m+1}-1-M)+N > 10^{m+1}-1$ $M-N<0$ so again a negative difference is indicated by the absence of a zero in the extra digit subadder. In this case, however, $M-N=0$ will not be accompanied by a negative sign. (It should be realized that even with complements with respect to 10^{m+1} 0 need not necessarily be accompanied by a negative sign. By a suitable arrangement it can be suppressed.) Other schemes of subtraction through completation that involve only one complementation and one addition operation but which do not always yield the absolute value of the difference and the negative sign are described in [13].

Before combining the adder with a complementor the equations characterizing the latter alone will be derived. This will convey a clearer picture of the mechanics of numerical complementation than the one that could be obtained from the equations of the combination. The latter, as may have already been inferred, is closer to the realistic situation to decimal computers that use the operations of addition and complementation than are separate units. Here a combination should be construed not as a mere juxtaposition of two independent units, the adder and the complementor, but rather as a unit in which some physical devices are shared by both of them, that is, part of the physical devices in the unit perform both addition and

complementation.

It should be clear that complementation in a computer may not be desired at all times but only under prescribed conditions. In such cases if the complementor is designed always to complement, numbers not to be complemented must be made to bypass it. An equivalent way of achieving the same goal is to pass all numbers through the complementor and make it also capable of no complementation. In other words, the complementor must also be able to reproduce the number that is presented to it. In either case the choice of complementation and no complementation must be controlled by some function or operator. The second method is more practical and more instructive than the first and, for that reason, will be developed. The operator of complementation will be denoted by Q_c and, of course, Q'_c will denote no complementation. Later the meaning of Q'_c will be modified.

The starting point for representing the numerical complements in terms of Boolean expressions can, as in the previous cases, be a numerical complement table and the code (7) on page 12. In view of the aforesaid about the numbers with respect to which to complement, the complements to be derived will be with respect to 9. They are listed in table XV and need no further comment. The 'no complement' in the table are of course the numbers themselves. Their Boolean equivalents are given immediately thereafter. There A_i's

are used for both the complementand and the 'no complementand' and S_i's are used for the complements and the 'no complements', as the case may be. The A_i's are met (intersected) with the two operators defined above. The reason for the choice of these letters is to facilitate the amalgamation of the complementor equations to be derived shortly with those of the adder on pages 21- 22, 25.

number	complement
0	9
1	8
2	7
3	6
4	5
5	4
6	3
7	2
8	1
9	0

TABLE XV

$$A'_1\ A'_2\ A'_4\ A'_5\ Q_c \leq S_4\ S_5 \qquad A'_1\ A'_2\ A'_4\ A'_5\ Q'_c \leq S'_1\ S'_2\ S'_4\ S'_5$$

$$A_1\ A'_2\ A'_5\ Q_c \leq S_1\ S_2\ S_5 \qquad A_1\ A'_2\ A'_5\ Q'_c \leq S_1\ S'_2\ S'_5$$

$$A'_1\ A_2\ A'_5\ Q_c \leq S'_1\ S_2\ S_5 \qquad A'_1\ A_2\ A'_5\ Q'_c \leq S'_1\ S_2\ S'_5$$

$$
\begin{array}{llll}
A_1 A_2 A_5' Q_c \leq S_1 S_2' S_5 & & A_1 A_2 A_5' Q_c' \leq S_1 S_2 S_5' \\
A_4 A_5' Q_c \leq S_1' S_2' S_4' S_5 & & A_4 A_5' \leq S_4 S_5' \\
A_1' A_2' A_4' A_5 Q_c \leq S_4 S_5 & & A_1' A_2' A_4' A_5 Q_c' \leq S_1' S_2' S_4' S_5 \\
A_1 A_2' A_5 Q_c \leq S_1 S_2 S_5' & & A_1 A_2' A_5 Q_c' \leq S_1 S_2' S_5 \\
A_1' A_2 A_5 Q_c \leq S_1' S_2 S_5' & & A_1' A_2 A_5 Q_c' \leq S_1' S_2 S_5 \\
A_1 A_2 A_5 Q_c \leq S_1 S_2' S_5' & & A_1 A_2 A_5 Q_c' \leq S_1 S_2 S_5 \\
A_4 A_5 Q_c \leq S_1' S_2' S_4' S_5' & & A_4 A_5 Q_c' \leq S_4 S_5
\end{array}
\tag{41}
$$

The reason for the inequality (implication) signs (rather than equality) is that a monomial in the S_i is determined either by complementation or by no complementation. Hence the equalities are

$$
\begin{array}{llll}
A_4 A_5 Q_c + A_1' A_2' A_4' A_5' Q_c' = S_1' S_2' S_4' S_5' = L_0 \\
A_1 A_2 A_5 Q_c + A_1 A_2' A_5' Q_c' = S_1 S_2' S_5' = L_1 \\
A_1' A_2 A_5 Q_c + A_1' A_2 A_5' Q_c' = S_1' S_2 S_5' = L_2 \\
A_1 A_2' A_5 Q_c + A_1 A_2 A_5' Q_c' = S_1 S_2 S_5' = L_3 \\
A_1' A_2' A_4' A_5 Q_c + A_4 A_5 Q_c' = S_4 S_5' = L_4 \\
A_4 A_5' Q_c + A_1' A_2' A_4' A_5 Q_c' = S_1' S_2' S_4' S_5 = L_5 \\
A_1 A_2 A_5' Q_c + A_1 A_2' A_5 Q_c' = S_1 S_2' S_5 = L_6 \\
A_1' A_2 A_5' Q_c + A_1' A_2 A_5 Q_c' = S_1' S_2 S_5 = L_7 \\
A_1 A_2' A_5' Q_c + A_1 A_2 A_5 Q_c' = S_1 S_2 S_5 = L_8 \\
A_1' A_2' A_4' A_5' Q_c + A_4 A_5 Q_c' = S_4 S_5 = L_9
\end{array}
\tag{42}
$$

Forming the 1-2-4-5 disjunctive canonical forms for these S_i (cf. (8),

p. 12 and the subsequent paragraph)

$$S_1 = L_1 + L_3 + L_6 + L_8 \qquad S'_1 = L_0 + L_2 + L_5 + L_7$$

$$S_2 = L_2 + L_3 + L_7 + L_8 \qquad S'_2 = L_0 + L_1 + L_5 + L_6 \qquad (43)$$

$$S_4 = L_4 + L_9 \qquad S'_4 = L_0 + L_1 + L_2 + L_3 + L_5 + L_6 + L_7 + L_8$$

$$S_5 = L_5 + L_6 + L_7 + L_8 + L_9 \qquad S'_5 = L_0 + L_1 + L_2 + L_3 + L_4$$

Hence, by (42) and (43)

$$S_1 = A_1 A_2 A_5 Q_c + A_1 A'_2 A'_5 Q'_c + A_1 A'_2 A'_5 Q_c + A_1 A_2 A'_5 Q'_c + A_1 A_2 A'_5 Q_c + A_1 A'_2 A_5 Q'_c + A_1 A'_2 A'_5 Q_c + A_1 A_2 A_5 Q'_c = A_1$$

$$S_2 = A'_1 A_2 A_5 Q_c + A'_1 A_2 A'_5 Q'_c + A_1 A'_2 A_5 Q_c + A_1 A_2 A'_5 Q'_c + A'_1 A_2 A'_5 Q_c + A'_1 A_2 A_5 Q'_c + A_1 A'_2 A'_5 Q_c + A_1 A_2 A_4 Q'_c = A'_1 A_2 + A_1 A'_2 Q_c + A_2 Q'_c \qquad (44)$$

$$S_4 = A'_1 A'_2 A'_4 A_5 Q_c + A_4 A_5 Q'_c + A'_1 A'_2 A'_4 A'_5 Q_c + A_4 A_5 Q'_c = A'_1 A'_2 A'_4 Q_c + A_4 Q'_c$$

$$S_5 = A_4 A'_5 Q_c + A'_1 A'_2 A'_4 A_5 Q'_c + A_1 A_2 A'_5 Q_c + A_1 A'_2 A_5 Q'_c + A'_1 A_2 A'_5 Q_c + A'_1 A_2 A_5 Q'_c + A_1 A'_2 A'_5 Q_c + A_1 A_2 A_5 Q'_c + A'_1 A'_2 A'_4 A'_5 Q_c + A_4 A_5 Q'_c = A'_5 Q_c + A_5 Q'_c$$

(The independence of the 5 weighted variable from the other three is also evident here.)

To combine addition and complementation as operations on the A_i which become in these operations the augend and the complementand, respectively, it is necessary to make the variables of the two operations span the universal set (space). For whenever complementation is not to be performed addition is (including addition of 0) and vice versa. However, the join of all ten addend functions (taking into account (2), (4) and (5)),

$$I'_1 I'_2 I'_4 I'_5 + I_1 I'_2 I'_5 + I'_1 I_2 I'_5 + I_1 I_2 I'_5 + I_4 I'_5 + I'_1 I'_2 I'_4 I_5 + I_1 I'_2 I_5 + I'_1 I_2 I_5 + I_1 I_2 I_5 + I_4 I_5 = 1 \tag{45}$$

is a tautology. Consequently complementation must somehow be fitted into this universal set, that is, it must be made a subset thereof. It certainly could not be made to take place during non-trivial addition; but it could be carried out when zero is being added to the augend (now regarded as an operand) as this is equivalent to not operating on it at all and so is "free" to be operated on by anything, and in particular is "free" to be complemented (numerically) by Q_c. Since non-trivial addition is indicated by the presence of at least one of the variables I_1, I_2, I_4, I_5 and implies no complementation it follows that

$$I_1, I_2, I_4, I_5 \leq Q'_c \tag{46}$$

whence

$$I_i\, Q_c = I_2\, Q_c = I_4\, Q_c = I_5\, Q_c = 0 .$$

As complementation is to be executed only during trivial addition which is characterized by the presence of $I_1'\ I_2'\ I_4'\ I_5'$ it follows that

$$Q_c \leq I_1'\ I_2'\ I_4'\ I_5' \tag{47}$$

(This could have been derived immediately from (46).) In view of this the complementation table in terms of the Boolean variables should be stated as

$$\begin{aligned}
A_1'\, A_2'\, A_4'\, A_5'\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_4\, S_5 \\
A_1\, A_2'\, A_5'\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1\, S_2\, S_5 \\
A_1'\, A_2\, A_5'\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1'\, S_2\, S_5 \\
A_1\, A_2\, A_5'\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1\, S_2'\, S_5 \\
A_4\, A_5'\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1'\, S_2'\, S_4'\, S_5 \\
A_1'\, A_2'\, A_4'\, A_5\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_4\, S_5' \\
A_1\, A_2'\, A_5\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1\, S_2\, S_5' \\
A_1'\, A_2\, A_5\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1'\, S_2\, S_5' \\
A_1\, A_2\, A_5\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1\, S_2'\, S_5' \\
A_4\, A_5\, I_1'\, I_2'\, I_4'\, I_5'\, Q_c &\leq S_1'\, S_2'\, S_4'\, S_5'
\end{aligned} \tag{48}$$

Inequality signs are used for the same reason given in connection with (41). But by (47) (48) reduces to the left "half" of (42).

Now just as every monomial in the left half of (42) was met (intersected) with Q_c every monomial in equations (14) and (16) (p. 21,25) should be met with Q'_c. Each would become then the equivalent of the right side of (42) from the standpoint of no complementation. But in view of (46) all monomials of these equations except those not involving I_1, I_2, I_4 and I_5 would reduce to what they are shown to be.

All that is left now in order to combine the addition and complementation equations is to form the joins of the 1-2-4-5 disjunctive canonical forms of their respective S_i. For addition these are of course (14) and (16) met (intersected) with Q'_c. Those for complementation can now be picked out from (44). They are the joins of the monomials met with Q_c. Thus

$$\begin{aligned} S_1 = {} & A_1 I'_1 I'_2 I'_4 I'_5 Q'_c + A'_1 A'_2 A'_4 I_1 + A'_1 A_2 I_1 I'_2 + A'_1 A_2 I_4 \\ & + A_1 A'_2 I'_1 I_2 + A_4 I'_1 I_2 + A_1 A_2 I_1 I_2 + A_4 I_4 \\ & + A_1 I'_1 I'_2 I'_4 I'_5 Q_c \end{aligned}$$

$$\begin{aligned} = {} & A_1 I'_1 I'_2 I'_4 + A'_1 A'_2 A'_4 I_1 + A'_1 A_2 I_1 I'_2 + A'_1 A_2 I_4 \\ & + A_1 A'_2 I'_1 I_2 + A_4 I'_1 I_2 + A_1 A_2 I_1 I_2 + A_4 I_4 \end{aligned} \qquad (49)$$

This is the same equation as the first one in (14). It indicates that A_1 is unaffected by complementation (in the 1-2-4-5 code). This is also evident from the first equation of (44).

$$S_2 = A_1' A_2 I_2' I_4 Q_c' + A_1 A_2' I_1 I_2' + A_2' A_4' I_1' I_2 + A_1 A_2 I_1' I_2' Q_c'$$
$$+ A_1' A_2' I_1 I_2 + A_4 I_4 + A_1 A_2' Q_c + A_1' A_2 Q_c$$

$$= A_1' A_2 I_2' I_4 + A_1 A_2' I_1 I_2' + A_2' A_4' I_1' I_2 + A_1 A_2 I_1' I_2' Q_c$$
$$+ A_1' A_2' I_1 I_2 + A_4 I_4 + A_1 A_2' Q_c \qquad (50)$$

$$S_4 = A_4 I_1' I_2' I_4' Q_c' + A_1 A_2 I_1 I_2' + A_1' A_2 I_1' I_2 + A_1 A_2' I_1' I_2$$
$$+ A_1' A_2' A_4' I_4 + A_1' A_2' A_4' Q_c \qquad (51)$$

The formation of the equation for S_5 may require some elucidation as the left side of (16) (p. 21) consists of monomials that contain C_q and C_q'. Since every term of C_q involves some I_i it follows by (46) that $C_q \leq Q_c'$ and hence $Q_c \leq C_q'$. Therefore the left side of (16) met (intersected) with Q_c' becomes

$$A_5' I_5' C_q + A_5 I_5 C_q + A_5 I_5' C_q' Q_c' + A_5' I_5 C_q'$$

This expression joined (united) with $A_5' Q_c$ from the fourth equation of (44) yields

$$S_5 = A_5' I_5' C_q + A_5 I_5 C_q + A_5 I_5' C_q' Q_c' + A_5' I_5 C_q' + A_5' Q_c. \qquad (52)$$

It has been intimated that the equations for the combined adder and complementor could be derived at once. This will not be done here as the results will be the same as (49)-(52) but the first (and essential) step will be indicated. This is the formation of a combined

addition and complementation table as shown below. The procedure from this table to the detailed equations should be clear by now.

	$A'_1\ A'_2\ A'_4$	$A_1\ A'_2$	$A'_1\ A_2$	$A_1\ A_2$	A_4
$I'_1\ I'_2\ I'_4\ Q_c$	S_4	$S_1\ S_2$	$S'_1\ S_2$	$S_1\ S'_2$	$S'_1\ S'_2\ S'_4$
$I'_1\ I'_2\ I'_4\ Q'_c$	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$	$S'_1\ S_2$	$S_1\ S_2$	S_4
$I_1\ I'_2\ Q'_c$	$S_1\ S_2$	$S'_1\ S_2$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S_4$
$I'_1\ I_2\ Q'_c$	$S'_1\ S_2$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$
$I_1\ I_2\ Q'_c$	$S_1\ S_2$	S_4	$S'_1\ S'_2\ S_4$	$S_1\ S'_2$	$S'_1\ S_2$
$I_4\ Q'_c$	S_4	$S'_1\ S'_2\ S'_4$	$S_1\ S'_2$	$S'_1\ S_2$	$S_1\ S_2$

TABLE XVI

For emphasis the monomials in the first column of table XVI have not been simplified by (46). Finally it should be noted that the arithmetic in this table is still modulo 5.

CHAPTER V

ADDITION - SUBTRACTION

A comparison of (14) and (16) (p.21,25) with (49)-(52) shows little difference between the equations for the individual S_i of the two sets. This is probably the strongest reason for employing complementation for subtraction. However, as has already been pointed out, this relative simplicity of numerical complementation is paid for by the relative slowness of subtraction as compared to addition. When speed of operation is of prime importance and the cost of components is of secondary importance an adder and a subtractor can be combined into one unit to produce a computer that can be appreciably faster in its operation than one with an adder-complementor combination. Such a computer is best utilized when a large number of its arithmetic operations is division which is actually a series of subtractions. Again a combination adder-subtractor should be understood in the sense that was pointed out for the adder-complementor combination on page 83. Clearly a computer with such a combination would normally not require facilities for numerical complementation and when a cost comparison is made between it and one which does contain a complementor this fact should not be overlooked, as minor as one may deem it to be. Moreover it should be realized that the actual increase in the number of components is far less than

what may appear from the equations to be derived as compared with (13, (14) and (15). The reason is that these equations represent only the "and gates" (electrical representations of Boolean monomial or meet) and only the number of these need be increased over those in the adder alone. However, the number of flip flops (electrical representations of Boolean functional values) need not be changed.

Let Q_a denote the addition operator in the adder-subtractor combination. (Cf. p. 84.) Assuming that if addition in the unit is not being performed then subtraction is, it follows that the subtraction operator is Q'_a. (In the case of trivial operations (cf. p. 32, 88) in the combination it is immaterial which one of the two operators is 1 as then neither changes the contents (the number in terms of the A_i) that may be present in the adder-subtractor.) Thus the right sides of equations (14) and (16) should be met (intersected) with Q_a and the right sides of (29) and (30) should be met with Q'_a. The corresponding results according to the subscript i should then be joined (united) to form a single right side. To indicate that these right sides are the joins of modified monomials that define S_i and D_i they will be denoted by the suggestive symbol $(S*D)_i$. (The asterisk between the S and the D is to avoid possible misinterpretation of meet of the two functions if nothing is inserted between the two letters.)

It will be noticed that four monomials in the unsimplified right sides of S_1 and of D_1 are identical. They are distinguished by the

circled numbers (1) through (4) above them. Since these monomials are to be met with Q_a in S_1 and with Q'_a in D_1 it follows that they can be left as they are ($Q_a + Q'_a = 1$) as they are common to both addition and subtraction. Hence using the unsimplified forms one obtains

$$\begin{aligned}(S*D)_1 &= A'_1\ A'_2\ A'_4\ I_1\ I'_2\ Q_a + A_1\ A'_2\ I'_1\ I_2\ Q_a + A'_1\ A'_2\ A'_4\ I_1\ I_2\ Q_a \\ &+ A_4\ I'_1\ I'_2\ Q_a + A_1\ A_2\ I_1\ I_2\ Q_a + A_4\ I_4\ Q_a + A_1\ A_2\ I'_1\ I_2\ Q'_a \\ &+ A_4\ I_1\ I_2\ Q'_a + A_4\ I_1\ I'_2\ Q'_a + A'_1\ A'_2\ A'_4\ I_4\ Q'_a \\ &+ A'_1\ A'_2\ A'_4\ I'_1\ I_2\ Q'_a + A_1\ A'_2\ I_1\ I_2\ Q'_a + A_1\ A'_2\ I'_1\ I'_2\ I'_4 \\ &+ A_1\ A_2\ I'_1\ I'_2\ I'_4 + A'_1\ A_2\ I_1\ I'_2 + A'_1\ A_2\ I_4 \\ \\ &= A'_1\ A'_2\ A'_4\ I_1\ Q_a + A_1\ A'_2\ I'_1\ I_2\ Q_a + A_4\ I'_1\ I_2\ Q_a \qquad (53) \\ &+ A_1\ A_2\ I_1\ I_2\ Q_a + A_4\ I_4\ Q_a + A_1\ A_2\ I'_1\ I_2\ Q'_a + A_4\ I_1\ Q'_a \\ &+ A'_1\ A'_2\ A'_4\ I_4\ Q'_a + A'_1\ A'_2\ A'_4\ I'_1\ I_2\ Q'_a + A_1\ A'_2\ I_1\ I_2\ Q'_a \\ &+ A_1\ I'_1\ I'_2\ I'_4 + A'_1\ A_2\ I_1\ I'_2 + A'_1\ A_2\ I_4\end{aligned}$$

The term $A_1\ A_2\ I'_1\ I_2\ Q'_a$ can be replaced by $A_1\ A_2\ I'_1\ I'_4\ Q_a'$ (appendix A, p. 139). When this is done the simplified right side of this equation becomes an immediate result of the simplified right sides of the equations of S_1 and of D_1.

The unsimplified expressions for S_2 and for D_2 also have four monomials in common marked (1) through (4). If they are to be used to form the equation for $(S*D)_2$ the remaining six monomials

from S_2 and the same number from D_2 must be included. This would make the polynomial for $(S*D)_2$ consist of sixteen monomials. The simplified right side of S_2 has six terms. The same is true of D_2. All twelve terms are distinct. But even so the use of them would result in a polynomial of only twelve terms. Since the second alternative is prima facie more desirable (there may be electrical problems to be reckoned with) it should be

$$(S*D)_2 = A_1' A_2 I_2' I_4' Q_a + A_1 A_2' I_1 I_2' Q_a + A_2' A_4' I_1' I_2 Q_a$$
$$+ A_1 A_2 I_1' I_2' Q_a + A_1' A_2' I_1 I_2 Q_a + A_4 I_4 Q_a$$
$$+ A_1 A_2 I_2' I_4 Q_a' + A_1' A_2 I_1' I_2' Q_a' + A_1 A_2' I_4 Q_a'$$
$$+ A_2' A_4' I_1 I_2 Q_a' + A_1' A_2' I_1' I_2 Q_a' + A_4 I_1 I_2' Q_a' .$$

With S_4 and D_4 there is little choice and therefore

$$(S*D)_4 = A_1 A_2 I_1 I_2' Q_a + A_1' A_2 I_1' I_2 Q_a + A_1 A_2' I_1 I_2 Q_a +$$
$$+ A_1' A_2' A_4' I_4 Q_a + A_1' A_2' A_4' I_1 I_2' Q_a' + A_1 A_2' I_1' I_2 Q_a'$$
$$+ A_1' A_2 I_1 I_2 Q_a' + A_1 A_2 I_4 Q_a' + A_4 I_1' I_2' I_4' .$$

$(S*D)_5$ presents somewhat of a problem due to C_q, B_q and their complements. These variables are not independent and a comparison amongst them must of course be made in terms of A_i, I_i, i = 1, 2, 4, and their complements. Although the unsimplified forms for C_q and for B_q have four monomials in common little or no advantage can be taken of this for a reason similar to the one given in

regards to $(S*D)_2$. In this case it is fifteen versus eleven terms. The simplified versions of these functions have no terms in common. Their complements do contain two common monomials and both meet (intersect) $A_5 I'_5$ and $A'_5 I_5$. However it is doubtful whether it would be worthwhile to separate these two common terms from C'_q and from B'_q as undoubtedly the latter would be generated as individual terms anyway. Therefore

$$(S*D)_5 = A'_5 I'_5 C_q Q_a + A_5 I_5 C_q Q_a + A_5 I'_5 C'_q Q_a + A'_5 I_5 C'_q Q_a + A'_5 I'_5 B_q Q'_a + A_5 I_5 B_q Q'_a + A_5 I'_5 B'_q Q'_a + A'_5 I_5 B'_q Q'_a$$

It may be argued that since no common terms from S_2 and D_2 and from S_5 and D_5 were used to form $(S*D)_2$ and $(S*D)_5$, respectively, they in effect require the same quantity of components as the four separate functions do. This however, is not true since, as has already been indicated, the flip flops need not be duplicated.

Denoting the join (union) of the meets (intersections) $C Q_a$ and $B Q'_a$ by $(C*B)$ it follows from (17) and (31) (p. 26, 62) that

$$(C*B) = A_5 I_5 Q_a + A_5 C_q Q_a + I_5 C_q Q_a + A'_5 I_5 Q'_a + I_5 B_q Q'_a + A'_5 B_q Q'_a.$$

and similarly, from (18) and (32),

$$(C*B)' = A'_5 I'_5 Q_a + A'_5 C'_q Q_a + I'_5 C'_q Q_a + A_5 I'_5 Q'_a + I'_5 B'_q Q'_a + A_5 B'_q Q'_a.$$

H e r e again the duplication of flip flops or other delay devices should be borne in mind during a comparison.

The "standing on nines carries" and the "standing on zeros borrows" have nothing in common. Consequently the formation of the expressions (K*V) needs no further elaboration. Note however that

$$\begin{aligned} (K*V)_n &= K_n Q_a + V_n Q'_a \\ (K*V)'_n &= K'_n Q_a + V'_n Q'_a \end{aligned} \qquad (54)$$

In the developments of the adder and of the subtractor it was tacitly assumed that the Boolean functions on which these devices operate represent non-negative numbers. For the subtractor it became necessary to establish a criterion for recognizing negative numbers. This was done by associating with the difference the variable Z whose value was determined by that criterion (p. 75). However, no change in the arguments would have been necessary had it been assumed that the operands represented non-positive numbers. For let A and I be the absolute values of the numbers on whose Boolean representations the adder or the subtractor operates. Then the operations of the adder and of the subtractor as considered in chapters II and III correspond to

$$+ A \pm (+ I) = + (A \pm I)$$

whereas if non-negative numbers were assumed this would have

corresponded to

$$-A \pm (-I) = -(A \pm I)$$

and in the second case a criterion for a positive number would have been necessary. Clearly this argument also applies to the adder-subtractor.

In practice, however, operands representing distinct signs must be admitted. This creates the problem of adapting the present adder-subtractor to process meaningfully numbers of different signs for sums and differences. To this end some new symbols will be introduced.

Let Z_a and Z_i be the Boolean variables representing the signs of the numbers associated with the A_i and the I_i, respectively, such that their values for denoting positive and negative signs are the same as those adopted for Z (p. 75). (Note that Z may become a Z_a or a Z_i, depending on how the number associated with it is used.) Furthermore let F and F' denote the arithmetic operations of subtraction and addition, respectively, of the arbitrary numbers that are to be processed by the adder-subtractor. At this point it becomes desirable to bear in mind the logical distinction between the last pair of symbols and the pair Q_a and Q'_a. Therefore the first will be referred to as (representing) arithmetic operations and the latter as processing operations. If A and I are taken as above then all processing

requirements of the adder-subtractor can be represented by the eight cases that arise from all possible choices of signs in the expression

$$\pm A \textcircled{\pm} \pm I \tag{55}$$

where the circled signs are to be represented by F. The desired number and sign for a particular choice of signs in the above equation can now be well defined for the adder-subtractor in terms of the A_i, I_i, Z_a, Z_i and F.

Realizing that Q_a and Q'_a effectively produce absolute values which correspond to those of A + I and A - I, respectively, it follows from (55) and the variables defined for its sign that

$$\begin{aligned} Q_a &= Z'_a Z'_i F' + Z_a Z_i F' + Z_a Z'_i F + Z'_a Z_i F \\ Q'_a &= Z'_a Z'_i F + Z_a Z_i F + Z_a Z'_i F' + Z'_a Z_i F' . \end{aligned} \tag{56}$$

Equality signs are assumed since the only cases for Q_a and Q'_a to be admitted are those that arise from (55). The processing operation is thus uniquely determined from the desired arithmetic operation on the pair of numbers and their respective algebraic signs. Recalling that A and I are non-negative integers corresponding to the operands of the adder-subtractor it follows from the foregoing that under the operation of Q_a the combinations of signs in (55) is such that the equivalent of arithmetic addition is performed on numbers of like signs. Since the sign associated with A is not affected by

the sign denoting the arithmetic operation, as that of I is, it follows that in this case the (eventual) sign of the result can be determined from Z_a, and in fact is equal to the one Z_a represents. It is also the algebraic "product of signs"*, the "product" of those represented by Z_i and F individually. Hence, since this is certainly true when Q_a operates but may not be so when Q'_a does, it follows that

$$Z \geq Z_a Q_a \qquad \text{and} \qquad Z' \geq Z'_a Q_a .$$

On substituting in these inequalities the expression for Q_a from (56) they become

$$Z \geq Z_a Z_i F' + Z_a Z'_i F \quad \text{and} \quad Z' \geq Z'_a Z_i F' + Z'_a Z_i F. \tag{57}$$

It is somewhat more difficult to establish the eventual sign for the number resulting from an operation of Q'_a. The trichotomy of the real numbers can be enlisted, that is, one can examine which of the three arithmetical conditions

$$A < I \qquad A = I \qquad A > I \tag{58}$$

---

*The product of signs is given by table XVII. This multiplication is isomorphic to addition modulo 2 that is shown in table XVIII. The former table uses the fact that in a ring (-a)(-b) = a b. This is not as trivial a result as it may seem (cf. [3] p. 5).

x	+	-
+	+	-
-	-	+

TABLE XVII

+	0	1
0	0	0
1	1	0

TABLE XVIII

prevails during an operation of Q'_a. Now when Q'_a operates the equivalent of subtraction is performed on numbers of like signs also but since this arithmetic operation is not commutative (cf. p. 52) the eventual sign depends on the conditions described by (58). These can be determined with the aid of the variable $(K*V)'_{m+1}$ which, when Q'_a operates, is, by (54), equivalent to V'_{m+1}.

If A = I, the sign is immaterial. If A < I (and Q'_a operates) then $(K*V)_{m+1} = 1$. Hence if the numbers and the desired arithmetic operation on them can be reduced in (55) to + (A - I) the result should be negative and if it can be reduced to - (A - I) then it should be non-negative. But clearly these can be deduced from the signs of A. Hence

$$Z \geq Z'_a (K*V)_{m+1} Q'_a \quad \text{and} \quad Z' \geq Z_a (K*V)_{m+1} Q'_a.$$

By similar reasoning, if A > I then $(K*V)'_{m+1} Q'_a = 1$ and

$$Z \geq Z_a (K*V)'_{m+1} Q'_a \quad \text{and} \quad Z' \geq Z'_a (K*V)_{m+1} Q'_a.$$

Then, on resorting to (54),

$$Z \geq Z'_a (K*V)_{m+1} Q'_a + Z_a (K*V)'_{m+1} Q'_a = Z'_a V_{m+1} Q'_a + Z_a V'_{m+1} Q'_a$$

$$Z' \geq Z (K*V)_{m+1} Q'_a + Z'(K*V)'_{m+1} Q'_a = Z_a V_{m+1} Q'_a + Z'_a V'_{m+1} Q'_a$$

and on substituting in these expressions for Q'_a from (56)

$$Z \geq Z'_a Z'_i F V'_{m+1} + Z'_a Z_i F' V'_{m+1} + Z_a Z_i F V'_{m+1} + Z_a Z'_i F' V'_{m+1}$$

$$Z' \geq Z_a Z_i F V_{m-1} + Z_a Z'_i F' V_{m-1} + Z'_a Z'_i F V'_{m-1} + Z'_a Z_i F' V'_{m+1}. \quad (59)$$

Before combining (57) with (59) into two expressions for Z and for Z', note that the following consensuses can be formed in the join of their respective right hand expressions. (Deletions at this stage are impossible.)

	$Z_a Z_i V'_{m+1}$	$Z_a Z'_i V'_{m+1}$	with respect to F
Z:	$Z_i F' V_{m+1}$	$Z'_i F V_{m+1}$	with respect to Z_a
	$Z_a F' V'_{m+1}$	$Z_a F V'_{m+1}$	with respect to Z_i
	$Z'_a Z'_i V'_{m+1}$	$Z'_a Z'_i V'_{m+1}$	with respect to F
Z':	$Z'_i F' V_{m+1}$	$Z_i F V_{m+1}$	with respect to Z_a
	$Z'_a F' V'_{m+1}$	$Z'_a F V'_{m+1}$	with respect to Z_i

The individual meets of these consensuses simplify to

$$\begin{aligned} Z: &\quad Z_a V'_{m+1} + Z_i F' V_{m+1} + Z'_i F V_{m+1} \\ Z': &\quad Z'_a V'_{m+1} + Z'_i F' V_{m+1} + Z_i F V_{m+1} \end{aligned} \quad (60)$$

and in (60)

$$
\begin{aligned}
&Z_a\ V_{m+1} \text{ deletes } Z_a\ Z_i\ F\ \ V_{m+1} \text{ and } Z_a\ Z_i'\ F'\ V_{m+1}' \\
Z:\ &Z_i\ F'\ V_{m+1} \text{ deletes } Z_a'\ Z_i\ V_{m+1} \\
&Z_i'\ F\ \ V_{m+1} \text{ deletes } Z_a'\ Z_i'\ F\ \ V_{m+1}
\end{aligned}
$$

$$
\begin{aligned}
&Z_a'\ V_{m+1}' \text{ deletes } Z_a'\ Z_i\ F\ \ V_{m+1}' \text{ and } Z_a'\ Z_i\ F'\ V_{m+1}' \\
Z':\ &Z_i'\ F'\ V_{m+1} \text{ deletes } Z_a\ Z_i'\ F'\ V_{m+1} \\
&Z_i\ F\ \ V_{m+1} \text{ deletes } Z_a\ Z_i\ F\ \ V_{m+1}
\end{aligned}
$$

Hence the pairs of the respective right hand sides for Z and for Z' of (57) and of (59) simplify to the corresponding right hand sides of (57) and of (60). Thus

$$
\begin{aligned}
Z &= Z_a\ Z_i\ F' + Z_a\ Z_i'\ F\ \ + Z_a\ V_{m+1}' + Z_i\ F'\ V_{m+1} \\
&\quad + Z_i'\ F\ \ V_{m+1} \qquad\qquad (61) \\
Z' &= Z_a'\ Z_i'\ F' + Z_a'\ Z_i\ F\ \ + Z_a'\ V_{m+1} + Z_i'\ F'\ V_{m+1} \\
&\quad + Z_i\ F\ \ V_{m+1}
\end{aligned}
$$

and equality signs are justified by the fact that (57) and (59) together include all cases for Z and for Z'. In these polynomials

$$
\begin{aligned}
Z:\ &Z_a\ Z_i\ F' \text{ is the consensus of } Z_a\ V_{m+1}' \text{ and } Z_i\ F'\ V_{m+1} \\
&Z_a\ Z_i'\ F \text{ is the consensus of } Z_a\ V_{m+1}' \text{ and } Z_i\ F\ V_{m+1}
\end{aligned}
$$

Z': $Z'_a\ Z_i\ F'$ is the consensus of $Z'_a\ V'_{m+1}$ and $Z_i\ F'\ V_{m+1}$

$Z'_a\ Z_i\ F$ is the consensus of $Z'_a\ V'_{m+1}$ and $Z_i\ F\ V_{m+1}$

Hence (61) can further be simplified (see (70) p. 130) to

$$\begin{aligned} Z &= Z_a\ V'_{m+1} + Z_i\ F'\ V_{m+1} + Z'_i\ F\ V_{m+1} \\ Z' &= Z'_a\ V_{m+1} + Z'_i\ F'\ V_{m+1} + Z_i\ F\ V_{m+1}\ . \end{aligned} \tag{62}$$

CHAPTER VI

MULTIPLICATION

One of the characteristics of the ring of integers is that a natural multiple of one of its elements is indistinguishable from the product of that element and some other one. This makes it possible to reduce multiplication to a series of additions and turns it into a special case of addition. (In general rings these two operations are logically distinct.) This property of the integers has been utilized in many digital computers and it is safe to assume that all decimal computers constructed in the past fifteen years did take advantage of this peculiarity of the integers to carry out multiplication. When this is done the multiplicand is effectively added to zero and to the resulting sums a number of times equal to the multiplier. Actually the number of such additions is far less than this and is at most equal to a ninth multiple of the number of digits in the multiplier. The reduction comes about by utilizing the positional significance of the digits in a radix notation, namely, by shifting the multiples of the multiplicand according to the place of the digit in the multiplier as they are added together. The actual number of additions is equal to the sum of the digits in the multiplier. This method requires a counter to control the number of additions in correspondence to the individual digits of the multiplier. (Chapter VII is the subject of a counter.)

It is possible to devise a multiplication scheme that is analogous to the addition and subtraction operations in the sense that it is performed in a digit by digit manner and which is on the average considerably faster than the one just discussed. It still requires additions but their number is equal to twice the number of digits in the multiplier rather than their sum. This is true if only one adder is used with the multiplier. If two adders are employed then, of course, the same number of additions is still required but the time to complete the entire multiplication operation can be reduced by a factor of two as compared to the time when only one is used (cf. [10] p. 199). This, in terms of time, reduces effectively the number of additions to the number of digits in the multiplier. Such a scheme is a modification of "paper and pencil" multiplication. There the partial products of the multiplicand by the multiplier digits are written out in a staggered array and subsequently added together to form the total product. The partial products are formed by referring to a memorized 10 x 10 decimal multiplication table. The least significant digit of the product of a multiplier digit by the first multiplicand digit is written down and the most significant digit is mentally retained and added to the product of the second multiplicand digit by the previous multiplier digit. The least significant digit of this sum is written down to the left of the previously written down digit and the most significant one is mentally retained again. This is repeated until all

multiplicant digits are exhausted. The same process is applied to the remaining multiplier digits until they are exhausted. The partial products are then properly added to form the total product. This is illustrated by the subsequent example on the left.

In the scheme to be described the least significant digits of the products of successive multiplicand digits by a multiplier digit are assembled as one number. The same is done with the most significant digits (the carries). The two numbers in the first of such pairs are then properly added with regard to the significance of their digits and a partial product of "paper and pencil" multiplication is obtained. To this partial product the numbers of the next pair are added in some suitable order and with proper regard of digit significance. This results in the sum of the first two partial products of "paper and pencil" multiplication. To this sum the numbers of the next pair is added until all multiplier digits are exhausted. This is illustrated by the subsequent example on the right.

```
  3 6 5
x   7 4
-------
  1 4 6 0
+ 2 5 5 5
---------
  2 7 0 1 0
```

```
  3 6 5
x   7 4
-------
    2 4 0
  1 2 2
  1 2 5
+ 2 4 3
---------
  2 7 0 1 0
```

It is worthwhile to point out in passing that the reason that addition, and hence multiplication, can be carried out in a digit by digit manner, with proper adjustments for carries, is primarily due to the additive nature of the decimal scale of notation. (This is true of course in any scale of notation with a radix.) Were integers, for example, represented by products of prime powers addition would be much harder than it is. On the other hand multiplication would be much easier than it is now as such a notation is multiplicative in nature. (Cf., however, the paragraph on division in [1], p. 93.)

In order to develop the Boolean functions two multiplication tables will be used. One is the ordinary 10 x 10 multiplication table. In it products which have the same carry (most significant digit) are blocked off in heavy line segments. This is table XIX. The second, table XXI, is arranged according to the same least significant digits of the products. It is in modulo 10 arithmetic and is analogous to table IV (p. 24) and table XI (p. 59) for addition and subtraction, respectively. The reason that for those operations the analog of table XVII was dispensed with, although it certainly exists, is that the carries and the borrows had only two values - 0 and 1. In multiplication the carries range from 0 through 8. The left part of table XXI corresponds to table II (p. 17) and to table IX (p. 56) which were the starting points for the derivation of the addition and subtraction equations, respectively. As a 5 x 5 array in modulo 5

arithmetic it is given in table XX.

x	0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9
2	0	2	4	6	8	10	12	14	16	18
3	0	3	6	9	12	15	18	21	24	27
4	0	4	8	12	16	20	24	28	32	36
5	0	5	10	15	20	25	30	35	40	45
6	0	6	12	16	24	30	36	42	48	54
7	0	7	14	21	28	35	42	49	56	63
8	0	8	16	24	32	40	48	56	64	72
9	0	9	18	27	36	45	54	63	72	81

TABLE XIX

x	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

TABLE XX

Let the Boolean variables of the multiplicand and the multiplier

TABLE XXI. SINGLE DIGIT MULTIPLICATION TABLE.

M_0	0 x 0	0 x 1	0 x 2	0 x 3	0 x 4	0 x 5	0 x 6	0 x 7	0 x 8	0 x 9	9 x 0	8 x 0	7 x 0	6 x 0	5 x 0
	1 x 0	2 x 0	3 x 0	4 x 0		2 x 5	4 x 5	6 x 5	8 x 5	5 x 8	6 x 8	5 x 4	5 x 2		
M_1		1 x 1						3 x 7	9 x 9	7 x 3					
M_2	1 x 2	3 x 4	4 x 3	2 x 1		2 x 6	7 x 6	4 x 8	9 x 8	8 x 9	8 x 2	7 x 6	6 x 2		
M_3		1 x 3	3 x 1						7 x 9	9 x 7					
M_4	1 x 4	2 x 2	4 x 1			2 x 7	3 x 8	4 x 6	6 x 9	8 x 8	9 x 6	6 x 4	8 x 3	7 x 2	
M_5						1 x 5	3 x 5	7 x 5	9 x 5	5 x 5	5 x 9	5 x 7	5 x 3	5 x 1	
M_6	2 x 3	4 x 4	3 x 2			1 x 6	2 x 8	4 x 9	7 x 8	6 x 6	8 x 7	9 x 4	8 x 2	6 x 1	
M_7								1 x 7	3 x 9	9 x 3	7 x 1				
M_8		2 x 4	4 x 2			1 x 8	2 x 9	3 x 6	4 x 7	6 x 8	8 x 6	7 x 4	6 x 3	9 x 2	8 x 1
M_9			3 x 3						1 x 9	7 x 7	9 x 1				

The M_j are to be interpreted modulo 10.

be denoted by A_i and by I_i, respectively. For mnemonic reasons P_i will denote the product variables and the quinary carry will be designated N_q. Now the numbers represented by the M_j are interpreted in table XX modulo 10. However, interpreting them in equations (63) modulo 5 (cf. p. 19) it follows that

$$\begin{aligned} M_0 &= P_1' P_2' P_4' N_q' \\ M_1 &= P_1 P_2' N_q' \\ M_2 &= P_1' P_2 N_q' \\ M_3 &= P_1 P_2 N_q' \\ M_4 &= P_4 N_q' \\ M_6 &= P_1 P_2' N_q \\ M_8 &= P_1 P_2 N_q \\ M_9 &= P_4 N_q \end{aligned} \tag{63}$$

and $P_1' P_2' P_4' N_q = P_1' P_2 N_q = 0$. In other words, there is no pair of (non-negative) digits each less than 5 whose product is 5 or 7.

The Boolean form of table XX is easily recognized to be

x	$A_1' A_2' A_4'$	$A_1 A_2'$	$A_1' A_2$	$A_1 A_2$	A_4
$I_1' I_2' I_4'$	M_0	M_0	M_0	M_0	M_0
$I_1 I_2'$	M_0	M_1	M_2	M_3	M_4
$I_1' I_2$	M_0	M_2	M_4	M_6	M_8
$I_1 I_2$	M_0	M_3	M_6	M_9	M_2
I_4	M_0	M_4	M_8	M_2	M_6

TABLE XXII

Hence by (63) and (10)

$$P_1 = A_1 A_2' I_1 I_2' + A_1 A_2' I_1 I_2 + A_1 A_2 I_1 I_2' + A_1' A_2 I_1 I_2 + A_4 I_4 + A_1 A_2 I_1' I_2 + A_4 I_1' I_2 + A_1' A_2 I_4$$

$$= A_1 A_2' I_1 + A_1 I_1 I_2' + A_1' A_2 I_1 I_2 + A_1 A_2 I_1' I_2 + A_4 I_1' I_2 + A_1' A_2 I_4 + A_4 I_4$$

$$P_2 = A_1 A_2' I_1' I_2 + A_1 A_2 I_4 + A_1' A_2 I_1 I_2' + A_4 I_1 I_2 + A_1 A_2' I_1 I_2 + A_1 A_2 I_1 I_2' + A_1' A_2 I_4 + A_4 I_1' I_2$$

$$= A_1 A_2' I_2 + A_2 I_1 I_2' + A_2 I_4 + A_4 I_2$$

$$P_4 = A_1 A_2' I_4 + A_4 I_1 I_2' + A_1' A_2 I_1' I_2 + A_1 A_2 I_1 I_2$$

$$N_q = A_1' A_2 I_1 I_2 + A_1' A_2 I_4 + A_1 A_2 I_1 I_2 + A_4 I_4 + A_4 I_1' I_2 + A_1 A_2 I_1' I_2$$

$$= A_2 I_1 I_2 + A_1 A_2 I_2 + A_4 I_1' I_2 + A_1' A_2 I_4 + A_4 I_4$$

$$N_q' = A_4 I_1' I_2' I_4' + A_1 A_2 I_1' I_2' I_4' + A_1' A_2 I_1' I_2' I_4' + A_1 A_2' I_1' I_2' I_4' + A_1' A_2' A_4' I_1' I_2' I_4' + A_1' A_2' A_4' I_1 I_2' + A_1' A_2' A_4' I_1' I_2 + A_1' A_2' A_4' I_1 I_2 + A_1' A_2' A_4' I_4 + A_4 I_1 I_2' + A_1 A_2 I_1 I_2' + A_1' A_2 I_1 I_2' + A_1 A_2' I_1 I_2' + A_1 A_2' I_1' I_2 + A_1 A_2' I_1 I_2 + A_1 A_2' I_4 + A_1 A_2 I_4 + A_1' A_2 I_1' I_2 + A_4 I_1 I_2 \quad (64)$$

$$= A_2' A_4' I_1' I_4' + A_1' A_2 I_1' I_2$$

The expressions for P_5 are not as easy to form as those for S_5 and for D_5. The reason is that the variables A_5, I_5 and N_q are insufficient by themselves to determine it. For example, both in 3 x 6 and in 3 x 7 $A_5 = 0$, $I_5 = 1$, $N_q = 0$ but in the first product $P_5 = 1$ while in the second $P_5 = 0$. A "brute force" way to define P_5 would be to pick out from table XIX the products which require P_5 for the representation of their least significant digits and form the join (union) of their factors in Boolean form. Numerically these products can be characterized as those numbers in the table that can be represented as a sum of an odd multiple of 5 and a number that does exceed 5. However, a more elegant way is the following.

Let a_5 and i_5 be the numerical analogs of A_5 and of I_5, respectively, that is, variables whose range is (0, 5). Similarly let the range of a and of i be (0, 1, 2, 3, 4). From the right hand side of the arithmetic equality

$$(a_5 + a)(i_5 + i) = a_5\, i_5 + a_5\, i + a\, i_5 + a\, i$$

it is easy to see that the least significant digit of this sum is greater or equal to 5 under the following conditions.

$$\begin{aligned} &a_5 = 5, \quad i_5 = 5, \quad a\, i < 5, \quad a + i \equiv 0 \ (2) \\ &a_5 = 5, \quad i_5 = 5, \quad a\, i \geq 5, \quad a + i \equiv 1 \ (2) \\ &a_5 = 0, \quad i_5 = 5, \quad a\, i \geq 5, \quad a \equiv 0 \ (2) \end{aligned} \tag{65}$$

$$a_5 = 0, \quad i_5 = 5, \quad a\,i < 5, \quad a \equiv 1\ (2)$$

$$a_5 = 5, \quad i_5 = 0, \quad a\,i \geq 5, \quad i \equiv 0\ (2)$$

$$a_5 = 5, \quad i_5 = 0, \quad a\,i < 5, \quad i \equiv 1\ (2)$$

$$a_5 = 0, \quad i_5 = 0, \quad a\,i \geq 5$$

That these exhaust all conditions for which the assertion is made can be verified by examining the complementary ones. There is only a total of $2^5 = 32$ cases to be considered, as becomes evident from the next paragraph* The task then is not too cumbersome.

Let the parity of a be denoted by the Boolean variable A_e, $A_e = 1$ when a is even. Let I_e be similarly defined for i. The arithmetic predicates $a\,i \geq 5$ and $a\,i < 5$ have already been defined by N_q. Since the even and odd states of $a + i$ depend on the individual states of a and i, the former states can be expressed in terms of A_e and I_e. Then the even state of $a + i$ is expressed as $A_e I_e + A'_e I'_e$. With these newly introduced symbols the Boolean form of (65) becomes

$$\begin{array}{c} A_5 I_5 (A_e I_e + A'_e I'_e)\ N'_q \\ A_5 I_5 (A_e I'_e + A'_e I_e)\ N_q \\ A'_5 I_5 A_e N_q \\ A'_5 I_5 A'_e N'_q \\ A_5 I'_5 I_e N_q \\ A_5 I'_5 I'_e N'_q \\ A'_5 I'_5 N_q \end{array} \qquad (66)$$

*The state of $a + i$ being even or odd should not be taken as an independent Boolean variable.

(In each of these monomials the number of variables which represent a 5 is odd.) Since in the five integers (0, 1, 2, 3, 4) 0, 2, 4 are even it follows that

$$A_e = A'_1 A'_2 A_4 + A'_1 A_2 + A_4$$
$$= A'_1$$
$$A'_e = A_1,$$

and similarly

$$I_e = I'_1$$
$$I'_e = I_1.$$

Hence, by (66) with these relationships introduced in the expressions

$$P_5 = A_5 I_5 (A_e I_e + A'_e I'_e) N'_q + A_5 I_5 (A_e I'_e + A'_e I_e) N_q$$
$$+ A'_5 I_5 A_e N_q + A'_5 I_5 A'_e N'_q + A_5 I'_5 I_e N_q + A_5 I'_5 I'_e N'_q$$
$$+ A'_5 I'_5 N_q$$

$$= A_1 A_5 I'_1 N_q + A'_1 I_1 I_5 N_q + A'_5 I'_5 N_q + A'_1 A'_5 N_q$$
$$+ I'_1 I'_5 N_q + A'_1 A_5 I'_1 I_5 N'_q + A_1 A_5 I_1 I_5 N'_q + A_1 A'_5 I_5 N'_q$$
$$+ A_5 I_1 I'_5 N'_q \qquad (67)$$

This equation can also be cast in a symmetrical form which brings to mind those for S_5 and for D_5, namely,

$$P_5 = (A'_1\, A_5\, I'_1\, I_5 + A_1\, I_1\, I_5 + A_1\, A'_5\, I_5 + A_5\, I_1\, I'_5)'\, N_q$$
$$+ (A'_1\, A_5\, I'_1\, I_5 + A_1\, I_1\, I_5 + A_1\, A'_5\, I_5 + A_5\, I_1\, I'_5)\, N'_q$$

The multiplication carries $\overline{C}_j$ weighted at j x 10, j = 1, 2, ... 8, are readily determined from the blocked off portions of table XIX. Thus

$$\overline{C}_1 = A'_1\, A_2\, A'_5\, I'_1\, I'_2\, I'_4\, I_5 + A'_1\, A_2\, A'_5\, I_1\, I'_2\, I_5 + A'_1\, A_2\, A'_5\, I'_1\, I_2\, I_5$$
$$+ A'_1\, A_2\, A'_5\, I_1\, I_2\, I_5 + A'_1\, A_2\, A'_5\, I_4\, I_5 + A_1\, A_2\, A'_5\, I_4\, I'_5$$
$$+ A_1\, A_2\, A'_5\, I'_1\, I'_2\, I'_4\, I_5 + A_1\, A_2\, A'_5\, I_1\, I'_2\, I_5 + A_4\, A'_5\, I_1\, I_2\, I'_5$$
$$+ A_4\, A'_5\, I_4\, I'_5 + A'_1\, A'_2\, A'_4\, A_5\, I'_1\, I_2\, I'_5 + A'_1\, A'_2\, A'_4\, A_5\, I_1\, I_2\, I'_5$$
$$+ A_1\, A'_2\, A_5\, I_1\, I_2\, I'_5 + A'_1\, A_2\, A_5\, I'_1\, I_2\, I'_5 + A_1\, A_2\, A_5\, I'_1\, I_2\, I'_5$$
$$+ A_4\, A_5\, I'_1\, I_2\, I'_5$$

$$= A'_1\, A_2\, A'_5\, I_5 + A_2\, A'_5\, I'_2\, I'_4\, I_5 + A_1\, A_2\, A'_5\, I_4\, I'_5 + A_4\, A'_5\, I_4\, I'_5$$
$$+ A_4\, A'_5\, I_1\, I_2\, I'_5 + A'_2\, A'_4\, A_5\, I_2\, I'_5 + A_5\, I'_1\, I_2\, I'_5$$

$$\overline{C}_2 = A_1\, A_2\, A'_5\, I'_1\, I_2\, I_5 + A_1\, A_2\, A'_5\, I_1\, I_2\, I_5 + A_1\, A_2\, A'_5\, I_4\, I_5$$
$$+ A_4\, A'_5\, I'_1\, I'_2\, I_4\, I_5 + A_4\, A'_5\, I_1\, I'_2\, I_5 + A_4\, A'_5\, I'_1\, I_2\, I_5$$
$$+ A'_1\, A'_2\, A'_4\, A_5\, I_4\, I'_5 + A'_1\, A'_2\, A'_4\, A_5\, I'_1\, I'_2\, I'_4\, I_5 + A_1\, A'_2\, A_5\, I_4\, I'_5$$
$$+ A'_1\, A_2\, A_5\, I_1\, I_2\, I'_5 + A'_1\, A_2\, A_5\, I_4\, I'_5 + A_1\, A_2\, A_5\, I_1\, I_2\, I'_5$$
$$+ A_4\, A_5\, I_1\, I_2\, I'_5$$

$$= A_1\, A_2\, A'_5\, I_4\, I_5 + A_1\, A_2\, A'_5\, I_2\, I_5 + A_4\, A'_5\, I'_2\, I'_4\, I_5$$
$$+ A_4\, A'_5\, I'_1\, I'_4\, I_5 + A'_1\, A'_2\, A'_4\, A_5\, I'_1\, I'_2\, I'_4\, I_5 + A_4\, A_5\, I_1\, I_2\, I'_5$$

$$+ A_2 A_5 I_1 I_2 I_5' + A_2' A_4' A_5 I_4 I_5 + A_1' A_4' A_5 I_4 I_5'$$

$$\bar{C}_3 = A_4 A_5' I_1 I_2 I_5 + A_4 A_5' I_4 I_5 + A_1' A_2' A_4' A_5 I_1 I_2' I_5$$
$$+ A_1' A_2' A_4' A_5 I_1' I_2 I_5 + A_1 A_2' A_5 I_1' I_2' I_4' I_5 + A_1 A_2' A_5 I_1 I_2' I_5$$
$$+ A_1' A_2 A_5 I_1' I_2' I_4' I_5 + A_1 A_2 A_5 I_4 I_5' + A_4 A_5 I_4 I_5'$$

$$= A_4 A_5' I_1 I_2 I_5 + A_4 A_5' I_4 I_5 + A_1' A_2' A_4' A_5 I_1 I_2' I_5$$
$$+ A_1' A_2' A_4' A_5 I_1' I_2 I_5 + A_1 A_2' A_5 I_2' I_4' I_5$$
$$- A_1' A_2 A_5 I_1' I_2' I_4' I_5 + A_4 A_5 I_4 I_5' + A_1 A_2 A_5 I_4 I_5'$$

$$\bar{C}_4 = A_1' A_2' A_4' A_5 I_1 I_2 I_5 + A_1' A_2' A_4' A_5 I_4 I_5 + A_1 A_2' A_5 I_1' I_2 I_5$$
$$+ A_1 A_2' A_5 I_1 I_2 I_5 + A_1' A_2 A_5 I_1 I_2' I_5 + A_1' A_2 A_5 I_1' I_2 I_5$$
$$+ A_1 A_2 A_5 I_1' I_2' I_4' I_5 + A_1 A_2 A_5 I_1 I_2' I_5 + A_4 A_5 I_1' I_2' I_4' I_5$$

$$= A_2' A_4' A_5 I_1 I_2 I_5 + A_1' A_2' A_4' A_5 I_4 I_5 + A_1 A_2' A_5 I_2 I_5$$
$$+ A_1' A_2 A_5 I_1' I_2 I_5 + A_4 A_5 I_1' I_2' I_4' I_5 + A_1 A_2 A_5 I_2' I_4' I_5$$
$$+ A_2 A_5 I_1 I_2' I_5$$

$$\bar{C}_5 = A_1 A_2' A_5 I_4 I_5 + A_1' A_2 A_5 I_1 I_2 I_5 + A_1 A_2 A_5 I_1' I_2 I_5$$
$$+ A_4 A_5 I_1 I_2' I_5$$

$$\bar{C}_6 = A_1' A_2 A_5 I_4 I_5 + A_1 A_2 A_5 I_1 I_2 I_5 + A_4 A_5 I_1' I_2 I_5$$

$$\bar{C}_7 = A_1 A_2 A_5 I_4 I_5 + A_4 A_5 I_1 I_2 I_5$$

$$\bar{C}_8 = A_4 A_5 I_4 I_5$$

The last equations need not have been minimized as in their present form they are unusable in the computer. This is because the numbers they represent are not in the chosen computer language - the code. The multiplicity of 10 is no hindrance. The problem is that some of the most significant digits of these numbers are not in the 1-2-4-5 code. Hence, in order to be able to add the numbers they represent to the other partial products it is necessary, in computer engineering terminology, to decode them. This, however, is nothing more than the formation of Boolean weighted variable in the given code according to (10) (p. 14), their 1-2-4-5 disjunctive canonical forms. (The simplification of the above expressions can be justified by the subsequent facilitation of minimization it affords.) Then, denoting these variables by F_i,

$$\begin{aligned} F_1 &= \overline{C}_1 + \overline{C}_3 + \overline{C}_6 + \overline{C}_8 & F_4 &= \overline{C}_4 \\ F_2 &= \overline{C}_2 + \overline{C}_3 + \overline{C}_7 + \overline{C}_8 & F_5 &= \overline{C}_5 + \overline{C}_6 + \overline{C}_7 + \overline{C}_8 \end{aligned} \tag{68}$$

Since there is no 90 weighted carry $\overline{C}_9 = 0$.

Making the substitutions in (68) one obtains

$$\begin{aligned} F_1 = {} & A_1' A_2 A_5' I_5 + A_2 A_5' I_2' I_4' I_5 + A_1 A_2 A_5' I_4 I_5' + A_4 A_5' I_4 I_5' \\ & + A_4 A_5' I_1 I_2 I_5' + A_2' A_4 A_5 I_2 I_5' + A_5 I_1' I_2 I_5' \\ & + A_4 A_5' I_1 I_2 I_5 + A_4 A_5' I_4 I_5 + A_2' A_4' A_5 I_1 I_2' I_5' \\ & + A_1' A_2' A_4' A_5 I_1' I_2 I_5 + A_1 A_2' A_5 I_2' I_4' I_5 \end{aligned}$$

$$+ A_1' A_2 A_5 I_1' I_2' I_4' I_5 + A_4 A_5 I_4 I_5' + A_1 A_2 A_5 I_4 I_5'$$
$$+ A_1 A_2 A_5 I_1 I_2 I_5 + A_4 A_5 I_1' I_2 I_5 + A_4 A_5 I_4 I_5$$

$$= A_1' A_2 A_5' I_5 + A_5 I_1' I_2 I_5' + A_2 A_5' I_2' I_4' I_5 + A_2' A_4' A_5 I_2 I_5'$$
$$+ A_1' A_2 I_1' I_2' I_5 + A_1' A_2' A_5 I_1' I_2 + A_4 A_5' I_1 I_2 + A_1 A_2 I_4 I_5'$$
$$+ A_1 A_2' A_5 I_2' I_4' I_5 + A_2' A_4' A_5 I_1 I_2' I_5 + A_1 A_2 A_5 I_1 I_2 I_5$$
$$+ A_4 I_5$$

$$F_2 = A_1 A_2 A_5' I_4 I_5 + A_1 A_2 A_5' I_2 I_5 + A_4 A_5' I_1' I_4' I_5$$
$$+ A_4 A_5' I_1' I_4' I_5 + A_1' A_2' A_4' A_5' I_1' I_2' I_4' I_5 + A_4 A_5 I_1 I_2 I_5'$$
$$+ A_2 A_5 I_1 I_2 I_5' + A_2' A_4' A_5 I_4 I_5' + A_1' A_4' A_5 I_4 I_5'$$
$$+ A_4 A_5' I_1 I_2 I_5 + A_4 A_5' I_4 I_5 + A_2' A_4' A_5 I_1 I_2' I_5$$
$$+ A_1' A_2' A_4' A_5 I_1' I_2 I_5 + A_1 A_2' A_5 I_2' I_4' I_5 + A_1' A_2 A_5 I_1' I_2' I_4' I_5$$
$$+ A_4 A_5 I_4 I_5' + A_1 A_2 A_5 I_4 I_5' + A_1 A_2 A_5 I_4 I_5$$
$$+ A_4 A_5 I_1 I_2 I_5 + A_4 A_5 I_4 I_5$$

$$= A_1 A_2 I_4 I_5 + A_4 A_5 I_1 I_2 + A_1 A_2 A_5' I_2 I_5 + A_2 A_5 I_1 I_2 I_5'$$
$$+ A_1' A_2' A_4' A_5 I_1' I_2 I_5 + A_1' A_2 A_5 I_1' I_2' I_4' I_5 + A_5 I_4 I_5'$$
$$+ A_4 A_5' I_5 + A_4 I_1 I_2 I_5 + A_1 A_2 A_5 I_4 + A_2' A_4' A_5 I_2' I_4' I_5$$

$$F_4 = A_2' A_4' A_5 I_1 I_2 I_5 + A_1 A_2 A_5 I_2' I_4' I_5 + A_1 A_2' A_5 I_2 I_5$$
$$+ A_2 A_5 I_1 I_2' I_5 + A_1' A_2' A_4' A_5 I_4 I_5 + A_4 A_5 I_1' I_2' I_4' I_5$$
$$+ A_1' A_2 A_5 I_1' I_2 I_5$$

$$F_5 = A_1 A_2' A_5 I_4 I_5 + A_1' A_2 A_5 I_1 I_2 I_5 + A_1 A_2 A_5 I_1' I_2 I_5$$
$$+ A_4 A_5 I_1 I_2' I_5 + A_1' A_2 A_5 I_4 I_5 + A_1 A_2 A_5 I_1' I_2 I_5$$
$$+ A_4 A_5 I_1 I_2 I_5 + A_1 A_2 A_5 I_4 I_5 + A_4 A_5 I_1 I_2 I_5$$
$$+ A_4 A_5 I_4 I_5$$

$$= A_2 A_5 I_1 I_2 I_5 + A_1 A_2 A_5 I_2 I_5 + A_1 A_5 I_4 I_5 + A_4 A_5 I_1 I_5$$
$$+ A_2 A_5 I_4 I_5 + A_4 A_5 I_2 I_5 + A_4 A_5 I_4 I_5$$

The simplified expressions of these equations have been arranged in symmetric pairs and symmetric singletons with respect to the A_i and the I_i. The same has been done with equations (64) and (67). The existence of this symmetry should not be surprising inasmuch as multiplication in the ring of integers is commutative.

Such symmetry is also true of addition for the same reason but it does not hold for subtraction. In (67) the term $A_1 A_5 F_1 I_5 N_q'$ could have been abbreviated either to $A_1 I_1 I_5 N_q'$ or to $A_1 A_5 I_1 N_q'$ since $A_1 A_5 I_1 I_5 N_q' + A_1 A_5' I_5 N_q' = A_1 I_1 I_5 N_q'$ and $A_1 A_5 I_1 I_5 N_q' + A_5 I_1 I_5' N_q' = A_1 A_5 I_1 N_q'$ (see the end of appendix A). However, for the sake of symmetry it was left as a five variable monomial. In practice there may be an advantage to choose a particular one of the three.

In comparing the adder equations (14) - (16) (p.21-25) with (64) and (67) of the multiplier it is seen that they are comparable in the number of components required to "mechanize" the least significant

digits is about the same for both units. A comparison of the carry digit equations is not easy. The form of the multiplication carry equation is fixed whereas those for addition depend on the number of the digits the adder is designed to accept. There is probably a number of digits for which the two are comparable. However, with a "sufficiently large" number of digits the variables in the adder carry equations outnumber those in the multiplier.

CHAPTER VII

COUNTING

It has been pointed out in the previous chapter (p. 106) that a counter is required when multiplication is carried out purely as a series of additions. A counter is also necessary for division as this operation in digital computers can only be performed as a series of subtractions. There it keeps track of the number of times the divisor is subtracted from the dividend.

A counter is a special case of an adder or of a subtractor in which the augend or the minuend is arbitrary and the addend or the subtrahend, respectively, is always 1. Counters can also be made to count by increments of arbitrary but fixed integers. However, there is little or no point in making such a counter since a one to one correspondence can always be set up between any set of distinct integers and the first non-negative ones, and in particular between the latter and a set that is a fixed multiple of each of them.

The use of a forward or a backward counter is a matter of convenience. When the first type is employed the number accumulated in the counter is usually compared with another one (like a digit in a multiplier or in a divisor) and coincidence serves to terminate an operation in progress or to initiate a new, or do both. With the second type of counter a number is set into it before an operation

begins and zero serves as the control signal.

The equations about to be developed are those for the forward counter as it probably is the type most frequently used and is to have a capacity of 10. The variables for the number present in the counter will be denoted by A_i's and the operator that increases that number by 1 will be denoted by I_1. The variables to represent the result of the operation will be designated by T_j Transforming table XXII to these variables

operand	result
0	1
1	2
2	3
3	4
4	5
5	6
6	7
7	8
8	9
9	0

TABLE XXIII

$$T_1 = A_1' A_2' A_4' A_5' I_1 + A_1' A_2 A_5' I_1 + A_1' A_2' A_5' I_1 + A_1' A_2 A_5 I_1$$

$$= A_1' A_2' I_1$$

$$T_2 = A_1 A_2' A_5' I_1 + A_1' A_2 A_5' I_1 + A_1 A_2' A_5 I_1 + A_1' A_2 A_5 I_1$$
$$= A_1 A_2' I_1 + A_1' A_2 I_1 \qquad (69)$$

$$T_4 = A_1 A_2 A_5' I_1 + A_1 A_2 A_5 I_1$$
$$= A_1 A_2 I_1$$

$$T_5 = A_4 A_5' I_1 + A_1' A_2' A_4' A_5 I_1 + A_1 A_2' A_5 I_1 + A_1' A_2 A_5 I_1$$
$$+ A_1 A_2 A_4 I_1$$
$$= A_4 A_5' I_1 + A_4' A_5 I_1$$

The carry, after the number in the counter turns to zero, is given by

$$C_t = A_4 A_5 I_1.$$

By right these equations should be obtainable from (19) (p. 30) as a special case when $I_1' = 0$. All of (19) (including the carry) except the equation for S_2 seem to support this contention. The reason is the simplification of S_2 by the presence of I_1'. Indeed this equation could have been left in such a form so as to yield to this claim. By the same token (18) should also be obtainable from (14) - (16) by setting I_1', I_2, I_4, I_5, to 0 (and of course I_2', I_4' and I_5' to 1) and substituting the resulting expressions for C_q and C_q' in that for S_5.

The extra digit subadder represented by (19) and the counter under discussion differ only by the ability of the former to "count by

zero" so to speak. Effectively this does not make them different but it may seem that they are not only technically different but logically as well. However, the second difference is only apparent and stems from the physical representation of the (numerical) zero of the addend. Some devices that represent ("mechanize") Boolean functions require that both the variable and its complement be supplied to them in some physical form. This means that for each Boolean variable there must be two physical channels in only one of which a specified signal exists at any time. Other devices can dispense with, say, the complement (signal) which is then represented by the absence of the variable signal. The first type of devices are referred to in electrical engineering as "double ended". Multivibrators and flip flops are representatives. The second type is known as "single ended", and require only one channel per variable. Ratchets that impart motions to gears in mechanical calculators are an example of the second type. No movement of the ratchet is equivalent to, say in the adder, the 0 addend. In other words, there is no need for a second moving ratchet to represent 0.

REFERENCES

1 Arden, Bruce W. An introduction to digital computing. Reading, Addison-Wesley, 1963. 389 p.

2 Bartree, Thomas C., Irwin L. Lebow and Irving S. Reed. Theory and design of digital machines. New York, McGraw-Hill, 1962. 324 p.

3 Birkhoff, Garrett and Saunders MacLane. A survey of modern algebra. Rev. ed. New York, MacMillan, 1958. 472 p.

4 Boole, George. An investigation of the laws of thought. New York, Dover, 1953. 472 p.

5 Flegg, H. Graham. Boolean algebra and its application. New York, Wiley, 1964. 261 p.

6 Goodstein, R. L. Boolean algebra. New York, MacMillan, 1963. 140 p.

7 Halmos, P. R. Lectures in Boolean algebra. Princeton, Van Nostrand, 1963. 147 p.

8 Harvard University. Computation Laboratory. A manual of operation for the automatic sequence controlled calculator. Cambridge, Harvard University Press, 1946. 561 p. (The Annals of the Computation Laboratory of Harvard University Vol. 1)

9 Harvard University. Computation Laboratory. Description of a relay calculator. Cambridge, Harvard University Press 1949. 366 p. (The Annals of the Computation Laboratory of Harvard University Vol. 24)

10 Harvard University. Computation Laboratory. Synthesis of electronic computing and control circuits. Cambridge, Harvard University Press, 1951. 278 p. (The Annals of the Computation Laboratory of Harvard University Vol. 27)

11 Johnson, Richard E. First course in abstract algebra. New York, Prentice-Hall, 1953. 257 p.

12 Munroe, Marshall E. Introduction to measure and integration. Reading, Addison-Wesley, 1953. 310 p.

13 Richards, R. K. Arithmetic operations in digital computers. Princeton, Van Nostrand, 1955. 397 p.

14 Stone, M. H. The theory of representations of Boolean algebras. Transactions of the Mathematical Society 40:37-111. 1936.

15 Veitch, E. W. A chart method for simplifying truth functions. Proceedings of the Association of Computing Machinery 127-133. May 2-3, 1952.*

16 White, G. S. Coded decimal number systems for digital computers. Proceedings of the Institute of Radio Engineers 41: 1450-1452. 1953.

---

*See note on p. 130.

APPENDICES

APPENDIX A

Most of the functions derived in the main text were formed directly from tables and except at the end of chapter V their expressions were immediately equated to simplified versions. No justifications were given for the simplifications as it was not desired to digress from the main vein of the subject. Part of this appendix is intended to fill in the steps that would formally allow to deduce the last lines of the adder equations from the first ones. After following these simplifications no difficulty should be encountered in justifying the remainder of the equations of the other subunits.

The problem of simplifying Boolean functions is always a fascinating one since generally no unique solution exists. Various techniques to minimize such expressions by given criteria have been developed. The best known is perhaps the technique by prime implicants. A description of it, as well as a host of references to other methods, can be found in chapter 4 of [2] . Here, however, no formal procedure is intended to be followed as the object is only to prove the validity of the simplified expressions. Recognition by inspection of patterns of terms that lend themselves to simplification will be mostly relied on.

Besides the analytical technique there is also a graphical method

known as the Veitch diagram [15] .* It is an extension of the Venn diagram for functions of four or more independent variables. As may be suspected the difficulty in using the diagram increases with the number of variables. However it has the distinction of requiring the least space on paper of perhaps all other known techniques. Instructions for the use of the Veitch diagram can be found in [13] , page 66, where it is essentially discussed for functions of four variables.** Since the expressions to be simplified here usually involve six independent variables and there is a rather important condition omitted in [13] , probably because it is automatically satisfied for four variables, the simplifications will also be carried out with the aid of the Veitch diagram.

Before proceding with the proofs the following two relations will be established.

$$X\,W + Y\,W' \geq X\,Y^{\#} \qquad (70)$$

$$X\,Y + X' = X' + Y \qquad (71)$$

---

*A copy of this article could not be obtained for purusal by the time the manuscript of this thesis was submitted for typing. A reply from the ACM to a query stated that it had never been published. This, however, is doubtful.

**This was found to be the case in a half dozen of texts which took up the Veitch diagram.

#$X\,Y$ is known as the consensus of $X\,W$ and $Y\,W'$. (67) is not used in this appendix but was used for obtaining (62) on page 105.

(70) follows from $X\,W > X\,Y\,W$, $Y\,W' > X\,Y\,W'$ and the formation of the join of the two expressions. (71) follows from

$$X\,Y + X' = X\,Y + X'\,Y + X'\,Y'$$

$$X' + Y = X'\,Y + X'\,Y' + X\,Y + X'\,Y$$

$$= X\,Y + X'\,Y + X'\,Y'$$

Starting with the first lines of the S_i

$$S_1 = A_1\,A_2'\,I_1'\,I_2'\,I_4' + A_1'\,A_2'\,A_4'\,I_1\,I_2' + A_1\,A_2\,I_1'\,I_2'\,I_4' + A_1'\,A_2\,I_1'\,I_2$$
$$+ A_1'\,A_2'\,A_4'\,I_1\,I_2 + A_4\,I_1'\,I_2 + A_1\,A_2\,I_1\,I_2 + A_1'\,A_2\,I_4$$
$$+ A_4\,I_4$$

$$= (A_1\,A_2' + A_1\,A_2)\,I_1'\,I_2'\,I_4' + A_1'\,A_2'\,A_4'\,(I_1'\,I_2 + I_1\,I_2)$$
$$+ A_1'\,A_2\,I_1'\,I_2 + A_4\,I_1'\,I_2 + A_1'\,A_2\,I_1\,I_2 + A_1\,A_2\,I_4 + A_4\,I_4$$

$$= A_1\,I_1'\,I_2'\,I_4' + A_1'\,A_2'\,A_4'\,I_1 + A_1'\,A_2\,I_1\,I_2' + A_1\,A_2'\,I_1'\,I_2$$
$$+ A_1'\,A_2\,I_4 + A_4\,I_1'\,I_2 + A_1\,A_2\,I_1\,I_2 + A_4\,I_4$$

$$S_2 = A_1'\,A_2\,I_1'\,I_2'\,I_4' + A_1\,A_2'\,I_1\,I_2' + A_1'\,A_2'\,A_4'\,I_1'\,I_2 + A_1\,A_2\,I_1'\,I_2'\,I_4'$$
$$+ A_1'\,A_2\,I_1\,I_2 + A_1\,A_2'\,I_1'\,I_2 + A_1'\,A_2'\,A_4'\,I_1\,I_2 + A_4\,I_1\,I_2$$
$$+ A_1\,A_2\,I_4 + A_4\,I_4$$

$$= A_1'\,A_2\,(I_1'\,I_4' + I_4)I_2' + A_1\,A_2'\,I_1\,I_2' + A_1\,A_2(I_1'\,I_2'\,I_4' + I_4)$$
$$+ A_2'(A_1'\,A_4' + A_4)I_1'\,I_2 + (A_1'\,A_2'\,A_4' + A_4)I_1'\,I_2 + A_4\,I_4$$

$$= A_1'\,A_2\,I_2'\,I_4' + A_2'\,A_4'\,I_1'\,I_2 + A_1\,A_2\,I_1'\,I_2' + A_1'\,A_2'\,I_1\,I_2$$

$$+ A_1 A_2' I_1 I_2' + A_4 I_4$$

The third line is obtained by applying (71) to the expressions in the parentheses and expanding the results. Since S_4 was not simplified there is nothing to do about it.

$$C_q = A_4 I_1 I_2' + A_1 A_2 I_1' I_2 + A_1' A_2 I_1 I_2 + A_1 A_2' I_4 + A_4 I_1' I_2$$
$$+ A_1 A_2 I_1 I_2 + A_1' A_2 I_4 + A_4 I_1 I_2 + A_1 A_2 I_4 + A_4 I_4$$

$$= A_4(I_1 I_2' + I_1' I_2 + I_1 I_2) + A_2(A_1 I_1' + A_1' I_1 + A_1 I_1)I_2$$
$$+ (A_1 A_2' + A_1' A_2 + A_1 A_2)I_4 + A_4 I_4$$

$$= A_4(I_1 I_2' + I_2) + A_2(A_1 I_1' + I_1)I_4 + (A_1 A_2' + A_2)I_4$$
$$+ A_4 I_4$$

$$= A_4 I_1 + A_1 I_4 + A_4 I_2 + A_2 I_4 + A_1 A_2 I_2 + A_2 I_1 I_2$$
$$+ A_4 I_4$$

Here again the fourth line was obtained from the third by applying (71) to it and expanding the meets.

The expression defining C_q' was not formulated in the text so it shall be done here. By (12) and (13)

$$C_q' = A_1' A_2' A_4' I_1' I_2' I_4' + A_1 A_2' I_1' I_2' I_4' + A_1' A_2' A_4' I_1 I_2'$$
$$+ A_1' A_2 I_1' I_2' I_4' + A_1 A_2' I_1 I_2' + A_1' A_2' A_4' I_1' I_2$$
$$+ A_1 A_2 I_1' I_2' I_4' + A_1' A_2 I_1 I_2' + A_1 A_2' I_1' I_2 + A_1' A_2' A_4' I_1 I_2$$

$$+ A_4 I_1' I_2' I_4' + A_1 A_2 I_1 I_2' + A_1' A_2 I_1' I_2 + A_1 A_2' I_1 I_2$$
$$+ A_1' A_2' A_4' I_4$$

$$C_q' = A_1' A_2' A_4'(I_1' I_2' I_4' + I_1 I_2' + I_1' I_2 + I_1 I_2 + I_4)$$
$$+ (A_1' A_2' A_4' + A_1 A_2' + A_1' A_2 + A_1 A_2 + I_4) I_1' I_2' I_4'$$
$$+ A_2 I_1 I_2' + A_1' A_2 I_1' I_2 + A_1 A_2' I_2 + A_1 A_2' I_1 I_2'$$

$$= A_1' A_2' A_4' + I_1' I_2' I_4' + I_1 I_2' A_2 + A_1' A_2 I_1' I_2 + A_1 A_2' I_2$$
$$+ A_1 A_2' I_1 I_2'$$

$$= A_1' A_2' A_4' + I_1' I_2' I_4' + A_2 A_4' I_1 I_2' I_4' + A_1 A_2' A_4' I_2 I_4'$$
$$+ A_1 A_2' A_4' I_1 I_2' I_4' + A_1' A_2 I_1' I_2$$

$$= A_2' A_4'(A_1' + A_1 I_1 I_2' I_4') + (A_2 A_4' I_1 + I_1')I_2' I_4' + A_1 A_2' A_4' I_2 I_4'$$
$$+ A_1' A_2 I_1' I_2$$

$$= A_2' A_4'(A_1' + A_1 I_2 I_4') + (A_2' A_4' I_1 + I_1')I_2' I_4' + A_2 A_4' I_2' I_4'$$
$$+ A_1' A_2 I_1' I_2$$

$$= A_1' A_2' A_4' + I_1' I_2' I_4' + A_2' A_4' I_2 I_4' + A_2' A_4' I_2' I_4'$$
$$+ A_2 A_4' I_2' I_4' + A_1' A_2 I_1' I_2$$

$$= A_1' A_2' A_4' + I_1' I_2' I_4' + A_2' A_4'(I_2' + I_2) I_4' + A_4'(A_2' + A_2) I_2' I_4'$$
$$+ A_1' A_2 I_1' I_2$$

$$= A_1' A_2' A_4' + I_1' I_2' I_4' + A_4' I_2' I_4' + A_2' A_4' I_4' + A_1' A_2 I_1' I_2$$

which is (15) on page 22.

The third line is obtained from the second by the tautology (45) on page 88. The fourth follows from the third by (4) and (5) on page 11. The rest follows by applications of (71). This expression, as already stated on page 22, can also be obtained by applying De Morgan's rules to the fourth equation of (14). In computer engineering terminology this would be referred to as an inversion of C_q. For the purpose of verification and illustration this will be done to (15) to obtain the fourth equation of (14) on page 21. Starting with (15) or the last line of the equation above

$$\left(C'_q\right)' = [\, A_2\, A'_4(A'_1 + I'_4) + (A'_4 + I'_1)I'_2\, I'_4 + A'_1\, A_2\, I'_1\, I_2]'$$

$$= [\, A_2 + A_4 + (A'_1 + I'_4)']\, [(A'_4 + I'_1)' + I_2 + I_4)]$$
$$[\, A_1 + A'_2 + I_1 + I'_2]$$

$$= (A_2 + A_4 + A_1\, I_4)(A_4\, I_1 + I_2 + I_4)(A_1 + A'_2 + I_1 + I'_2)$$

$$= (A_1\, A_2\, I_2 + A_2\, I_1\, I_2 + A_1\, A_2\, I_4 + A_2\, I'_2\, I_4 + A'_2\, A_4\, I_1$$
$$+ A_4\, I_1 + A'_2\, A_4\, I_2 + A'_2\, A_4\, I_4 + A_4\, I'_2\, I_4 + A_1\, I_4 + A_1\, A'_2 I_4$$
$$+ A_1\, I'_2\, I_4$$

$$= A_4\, I_1 + A_1\, I_4 + A_4\, I_2 + A_2\, I_4 + A_1\, A_2\, I_2 + A_2\, I_1\, I_2$$
$$+ A_4\, I_4$$

S_5 requires no treatment but would have had its expression been

derived from a 10 x 10 table without the auxiliary variable C_q.

For the Veitch diagram treatment let the monomials of the expressions to be simplified be considered as numbered in the order that they appear. These numbers should not be confused with the circled ones appearing above some of the terms of the S_i. To eliminate possible confusion the proposed numbers will not be marked down and are expected to be determined by actual count from left to right whenever necessary. These numbers will be entered in the diagram* to indicate not only that the square in which one appears has a representative in the expression to be simplified but will also show at a glance which one it is. Some squares will contain more than one numeral and will thus indicate that more than one monomial in the expression is represented by the particular square. These numbers should also not be mixed up with the numbers to be used to locate a particular square in the diagram. There will be $2^6 = 64$ such numbers as the expressions to be simplified are functions of six independent variables. The diagrams can therefore be made squares themselves. Following Richards [13] the locating numbers will run from left to right and down. This is illustrated in diagram 1. Rows and columns will be identified as they are in matrices. (Due to space problems the aforementioned squares appear as rectangles.)

---

*Richards in [13] uses 1's for these proposed numerals.

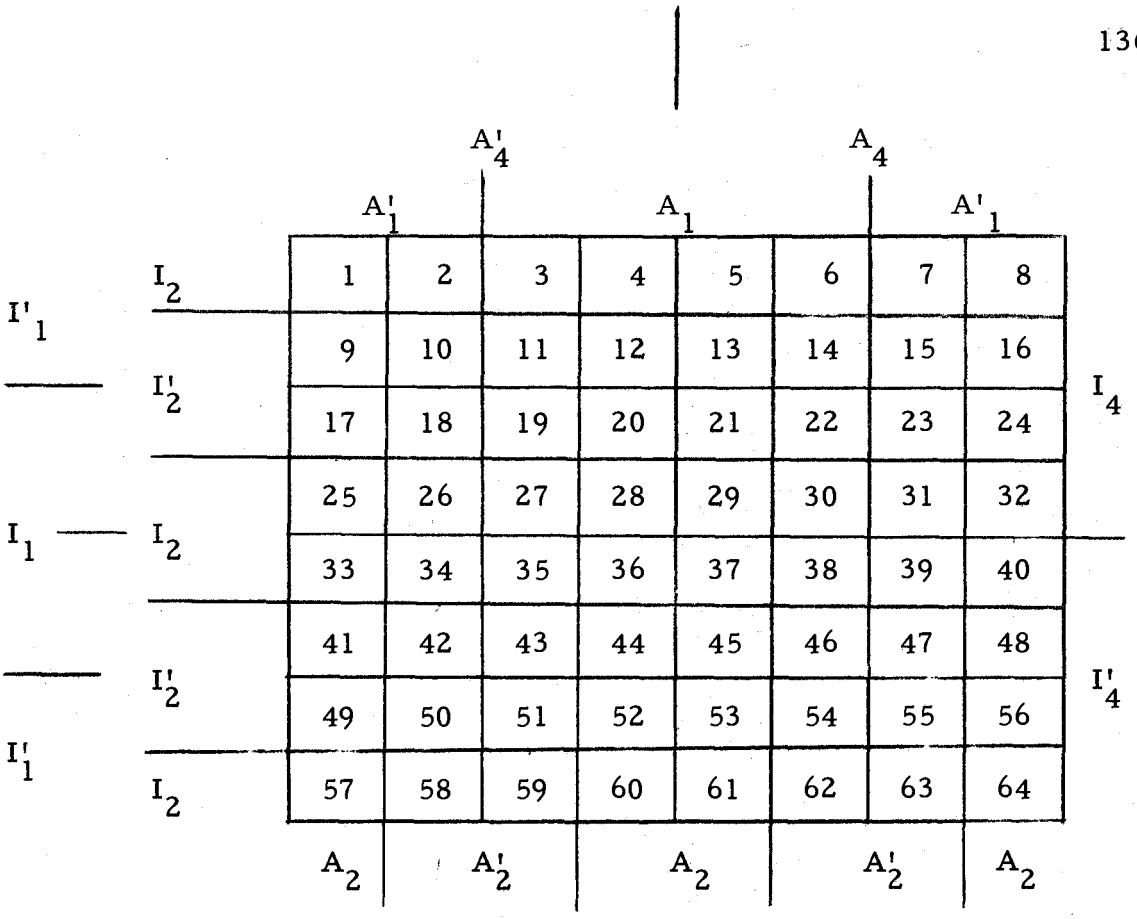


DIAGRAM 1

Since inadmissible meets of variables (cf. p10) can be entered in the diagram all monomials of less than six variables, either due to a "don't care" condition or to inadmissibility, will be made to take up more than one square. The number of squares a monomial should occupy is equal to 2^{r-p} where r is the number of independent variables in the expression to be simplified and p is the number of them that occur in the monomial.

In entering a monomial in the diagram it is convenient to consider first that variable in it for which the diagram or "set", to borrow terms from topology, is connected. The variables to which

such sets are assigned are, of course, arbitrary. Here A_4 and I_4 have been chosen for them. A_1 and I_1 almost have the same distinction. Their sets can be made to be connected by picturing the opposite sides of the diagram to be joined.

Starting with the first monomial $A_1\ A_2'\ I_1'\ I_2'\ I_4'$ in the unsimplified expression for S_1 it becomes evident that for it all 1's will be confined to the lower half of the diagram. Because of I_1' they are further narrowed down to the confines of rows 7 and 8, and in view of I_2' they must be limited to row 7. A_1 makes the restriction to columns 3 - 6, and A_2' pin points squares 51 and 54. In the same way the next monomial $A_1'\ A_2'\ A_4'\ I_1\ I_2'$ occupies squares 18 and 42 marked with 2's. The complete markings are given in diagram 2.

		A_4'			A_4			
		A_1'		A_1			A_1'	
I_1'	I_2	9		5	7;10	5;7;10	7;10	7;9;10
	I_2'	9			10	10	10	9;10
		4;9	2		10	10	10	4;9;10
I_1	I_2	9	6	8	8;10	10	10	9
			6	8	8			
	I_2'	4	2					4
I_1'				1	3	3	1	
	I_2			5	7	5;7	7	7
		A_2	A_2'		A_2		A_2'	A_2

I_4 — I_4'

DIAGRAM 2 (S_1)

The upper right hand quarter of the diagram is the largest single block of marked squares whose number is a power of 2. It cannot be combined with any other squares even if those are at opposite sides of it, the upper half of column 1 and the right hand half of row 8. The number of squares to be represented by a single monomial must be a power of 2 and $2^4 + 2^2 = 20$ is not. Hence the term $A_4 I_4$ becomes part of the simplified expression. Another term comes from the four center squares marked 8 (one is also marked 10). This term is $A_1 A_2 I_1 I_2$. It and the previous term have actually undergone no changes from the forms in which they appeared in the original expression. Another one like this can be formed from the upper halves of the first column and the last column to yield $A'_1 A_2 I_4$. That these terms would not differ from their predecessors can be recognized by the numbers in the squares of the combinations - they all contain 9's and so must have come from the same original term. However, a new term can be obtained from squares 18, 26, 34, 42 marked 2, 6, 6, 2, respectively. It is $A'_1 A'_2 A'_4 I_1$. Squares 18, 26 could also have been combined with squares 31, 39 to yield $A'_1 A'_2 I_1 I_4$. This was not done in order not to make the expression simplified by the diagram different from the one already obtained. This disposes of the upper half of the diagram except for square 3 marked 5. This one can be combined with squares 6, 59, 62. That such a combination can be formed follows from the fact that the squares contain the

number 5 and thus stem from the same monomial. But this could also be done without this information. For according to the rules for the Veitch diagram (in [13]) squares 3 and 59 can be combined to yield $A_1 A_2' A_4' I_1' I_2$. Similarly squares 6 and 62 produce $A_1 A_2' A_4 I_1' I_2$. The join of these two results in $A_1 A_2' I_1' I_2$.

Generally when two combinations are symmetrically located about subsets (including the null set - a dividing line) between them they can be combined themselves. For in such cases the two individual combinations differ only in the value of the variable that corresponds to the set of the centrally located dividing line.* (Symmetry about a row or a column is impossible as all subsets consist of an even number of squares.) The monomial $A_1' A_2 A_4' I_1 I_2' = A_1' A_2 I_1 I_2'$, squares 17, 24, 41, 48, undergoes no change. The same applies to $A_4 I_1' I_2$. But squares 51, 52, 53, 54 contain different numbers, namely, 1, 3, 3, 1, respectively, and so lead to $A_1 I_1' I_2' I_4'$. The simplification can be finished by combining squares 17, 24, 41, 48 containing the numeral 4 and squares 61, 62, 63, 64 containing the numeral 7 to obtain $A_1' A_2 A_4' I_1 I_2' = A_1' A_2 I_1 I_2'$ and $A_4 I_1' I_2 I_4' = A_4 I_1' I_2$, respectively. These are also original terms.

The next three diagrams, 3, 4, and 5, represent S_2, C_q and C_q', respectively. The procedure to carry out the simplifications

---

*This is the omitted rule alluded to in the third paragraph of this appendix.

should be clear now.

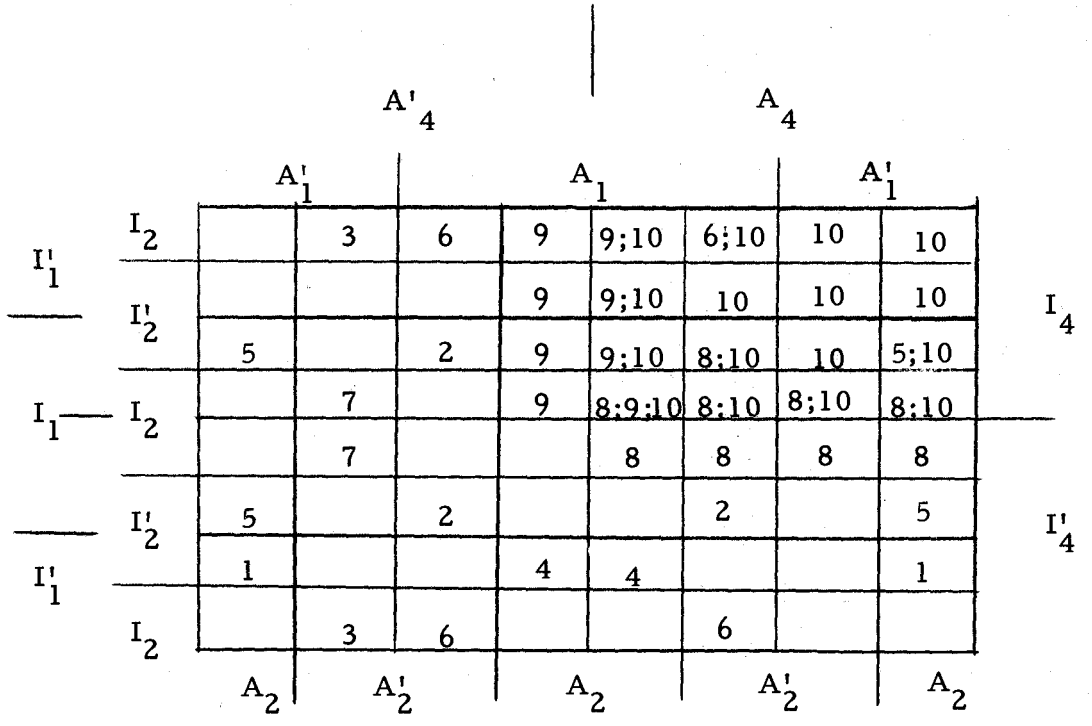


DIAGRAM 3 (S_2)

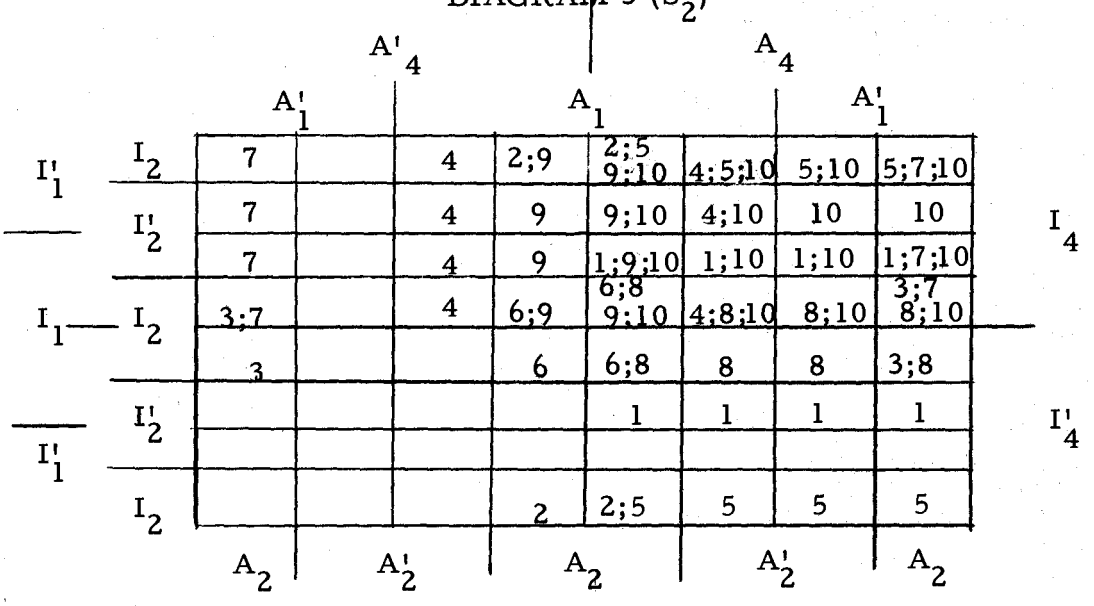


DIAGRAM 4 (C_q)

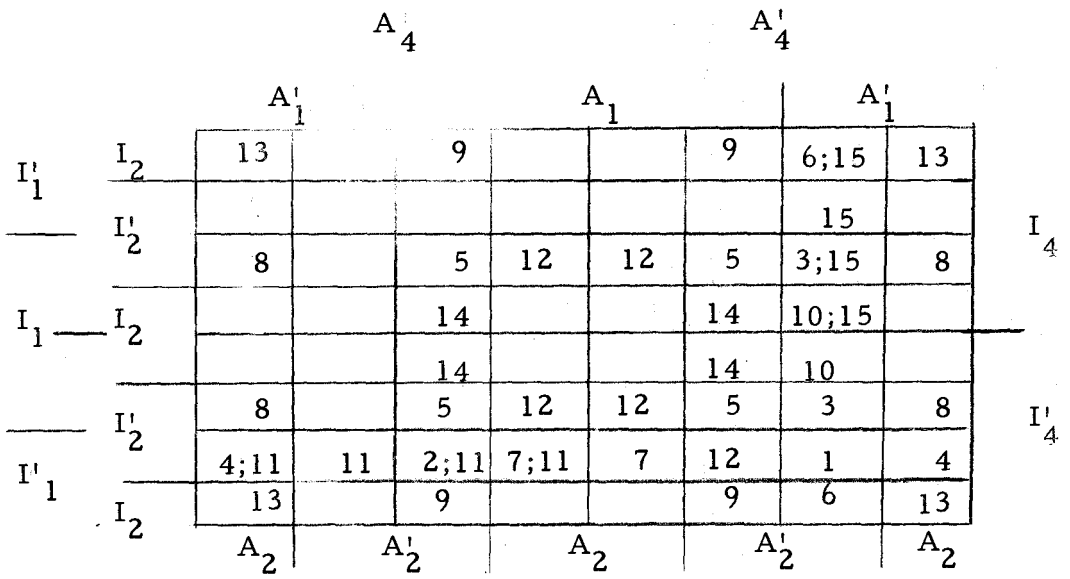


DIAGRAM 5 (C'_q)

The prevalence of terms of two and of three variables in the simplified expression of C_q can be traced to the "density" of marked squares in diagram 4. The expression for C'_q consists "almost" of only two variable terms. If it were not for the vacant squares 14 and 42 these would be $A'_2\ A'_4$ and $I'_2\ I'_4$.

On page 95 it was asserted that the term $A_1\ A_2\ I'_1\ I_2\ Q'_a$ could be replaced by $A_1\ A_2\ I'_1\ I'_4\ Q'_a$ in equation (67). The proof is very simple. Due to the presence of $A_1\ I'_1\ I'_2\ I'_4$ in this equation it follows that

$$A_1\ I'_1\ I'_2\ I'_4 + A_1\ A_2\ I'_1\ I_2\ Q'_q = A_1\ I'_1\ I'_2\ I'_4 + A_1\ A_2\ I'_1\ I_2\ I'_4\ Q'_a$$
$$\leq A_1\ I'_1\ I'_2\ I'_4 + A_1\ A_2\ I'_1\ I'_4\ Q'_a$$

$$= A_1 I_1' I_2' I_4' + A_1 A_2 I_1' I_2 I_4' Q_a'$$

$$+ A_1 A_2 I_1' I_2' I_4' Q_a'$$

$$= A_1 I_1' I_2' I_4' + A_1 A_2 I_1' I_2' I_4' Q_a'$$

$$= A_1 I_1' I_2' I_4' + A_1 A_2 I_1' I_2 O_a'$$

This proves the assertion.

APPENDIX B

In computer engineering practice the equations ("switching functions") developed in the text are also represented graphically as they are mechanized. This makes the symbolic representation closer to the physical one and thus affords ease of orientation in the equipment. The following is such an example for the adder equations. Since not only are expressions of Boolean functions not unique but their mechanizations is multi-valued as well this example should not be construed as the only one possible. In fact it is an oversimplification of what would actually be found in a computer. All control signals have been omitted.

The constituents of the representation are four distinct symbols that denote and gates, or gates, amplifiers and inverters. These devices perform operations that are equivalent to Boolean operations. The and gate corresponds to the operation of meet and the or gate is the physical realization of the join. These can be active (transistors, vacuum tubes) or passive (diodes) devices. An amplifier in computer engineering has a different meaning from the one associated with it in electronics. For the present purpose an amplifier will denote a device whose transfer function (disregarding time and scale considerations) is the identity. An inverter, also known as an inhibiter, is a device whose transfer function is the complement. (The

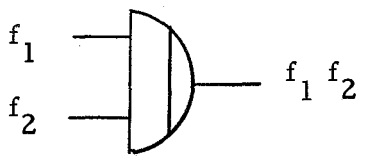
latter device comes closer to the amplifier in electronics than the former.)

Besides the equivalent of addition the combinations of these four devices to be given here are also capable of storing information (equivalents of the numbers of the augend or of the sum). These storage devices are essentially multivibrators. For each variable (bit) there is a regenerative closed loop (positive feedback) through the gates. Under ideal conditions the information is preserved (recirculated) indefinitely unless loops are opened. The A_i and the S_i are generated by the amplifiers. The A'_1 are generated by the inverters. The I_i and their complements are supposed to be supplied to the gates from sources that are outside the adder. Although connections by lines are not shown between the various points marked by A_i and A'_i for the same i it is to be understood that such connections exist. Interconnecting lines have been omitted to avoid cluttering the diagrams.

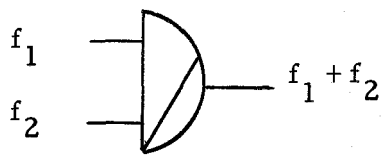
According to these diagrams there is no way to introduce the A_i by themselves but rather through the I_i. In other words the augend and the addend must be introduced one after the other. The first number introduced (through the I_i) is added to 0 and becomes the augend when some of the A_i are generated. The second number introduced becomes the addend and is added to the prevailing number in the adder. Equations (14) - (16) thus represent these diagrams

when the A_i are already present (recirculating) in the adder. By a suitable modification it is possible to introduce both the addend and the augend simultaneously.

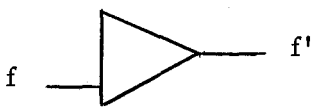
The symbols for the devices are given below.



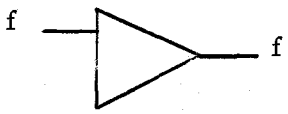
and gate



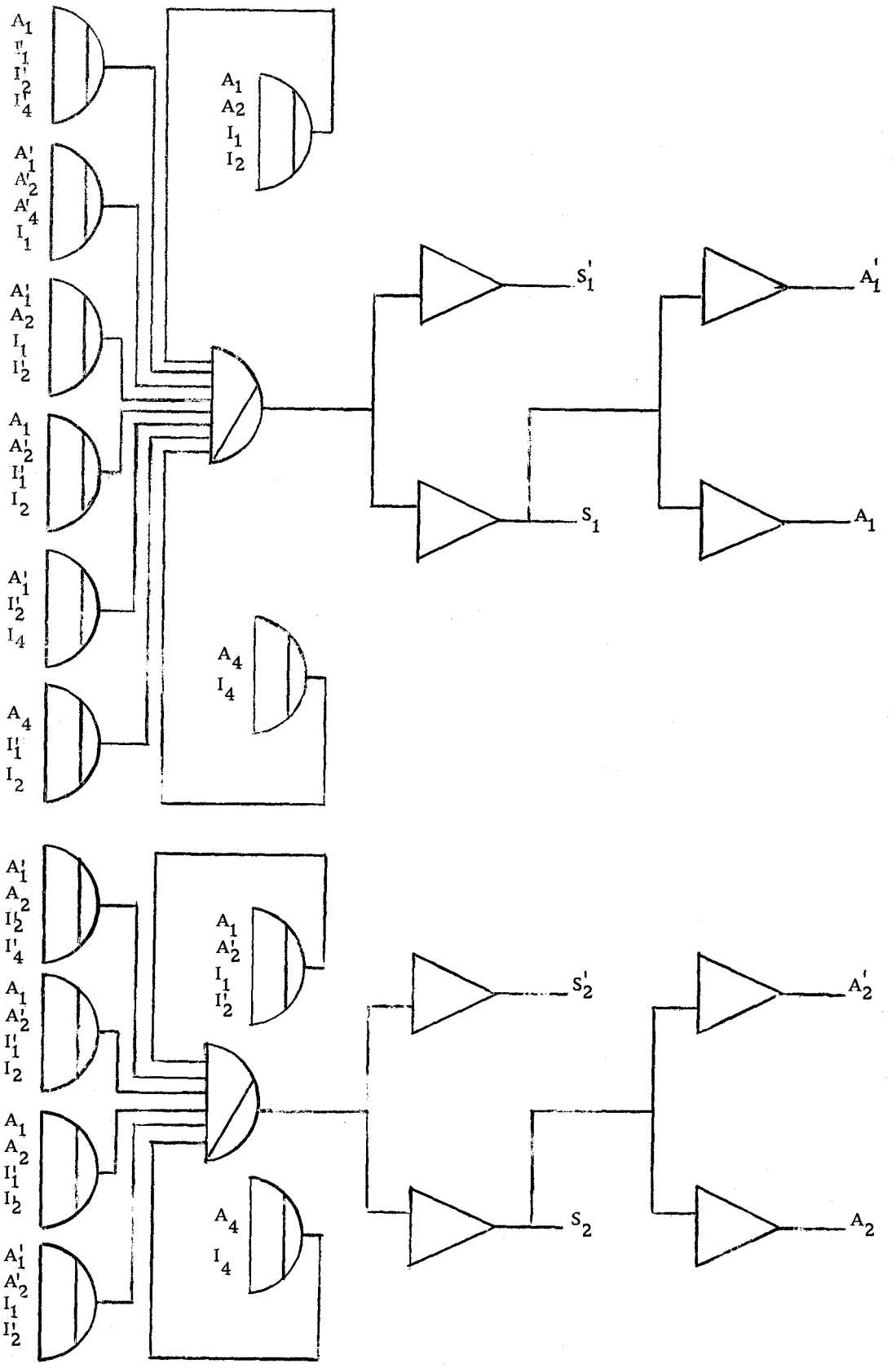
or gate

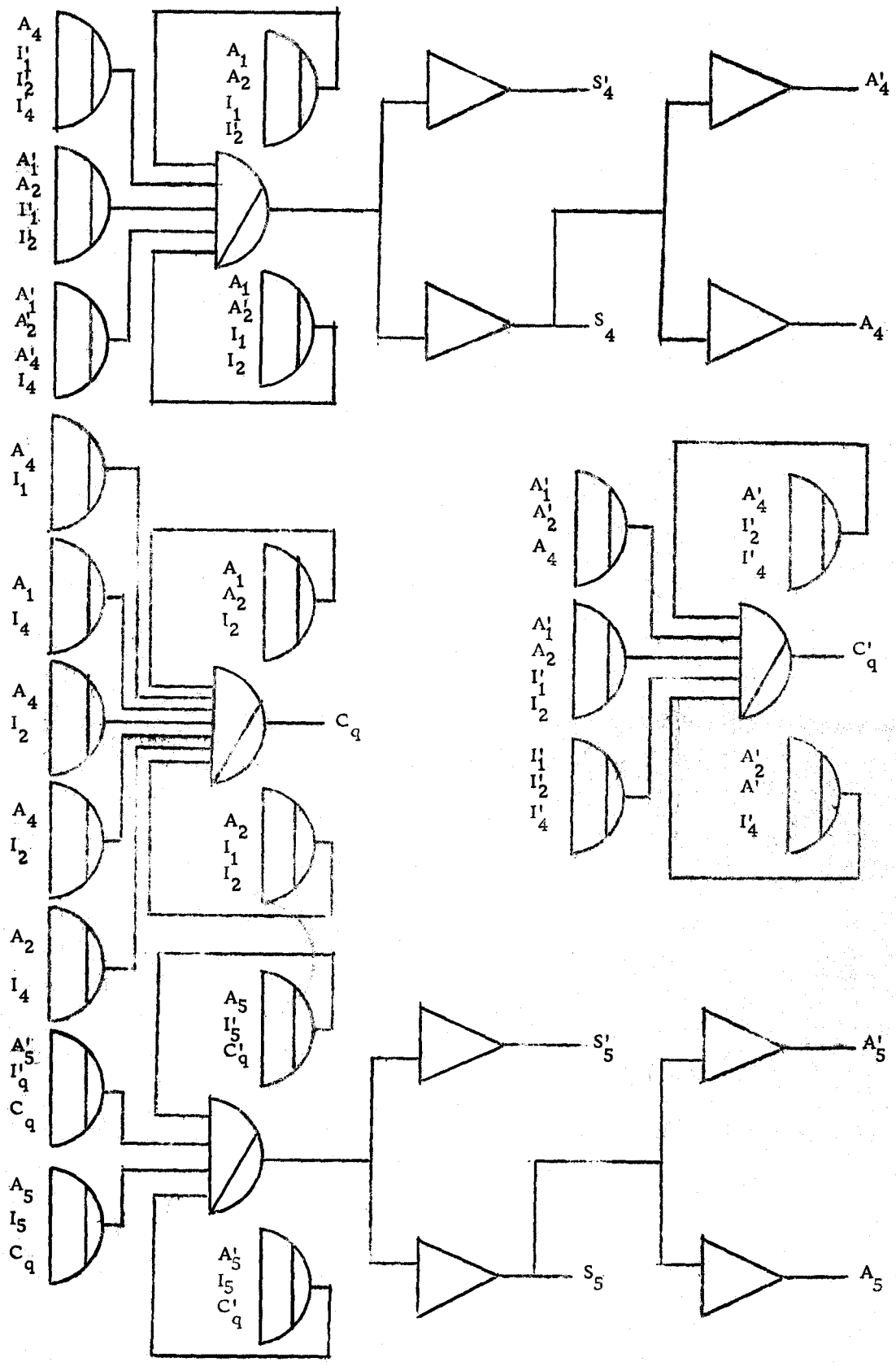


inverter



amplifier


A1
I'1
I'2
I'4
A'1
A'2
A'4
I1
A'1
A2
I1
I'2
A1
A'2
I'1
I2
A'1
I'2
I4
A4
I'1
I2
A1
A2
I1
I2
A4
I4
S'1
S1
A'1
A1
A'1
A'2
I1
I'2
A1
A'2
I'1
I2
A1
A2
I'1
I2
A'1
A'2
I1
I'2
A4
I4
S'2
S2
A'2
A2


A4
I'1
I'2
I4
A1
A2
I1
I'2
A'1
A2
I'1
I'2
A1
A'2
I1
I2
A'1
A'2
A'4
I4
S'4
S4
A'4
A4
A4
I1
A1
I4
A4
I2
A4
I2
A2
I4
A'5
I'q
Cq
A5
I5
Cq
A1
A2
I2
Cq
A2
I1
I'2
A5
I'5
C'q
A'5
I5
C'q
S'5
S5
A'5
A5
A'1
A'2
A4
A'1
A2
I'1
I'2
I'1
I'2
I'4
A'4
I'2
I'4
C'q
A'2
A'
I'4